EVPN Features

This chapter describes how to configure Layer 2 (L2) Ethernet VPN (EVPN) features on the Cisco ASR 9000 Series Aggregation Services Routers supporting Cisco IOS XR software.

- **EVPN Overview**, on page 2
- **EVPN Operation**, on page 2
- **EVPN Route Types**, on page 3
- **Configure EVPN L2 Bridging Service**, on page 4
- **EVPN Software MAC Learning**, on page 6
- **EVPN VXLAN Layer 2 Data Center Interconnect Gateway**, on page 16
- **Configure EVVPN VXLAN Layer 2 Data Center Interconnect Gateway**, on page 19
- **Configure L2 EVVPN Address Family under BGP Routing Process**, on page 19
- **Configure the Routing Sessions Between the DCI and ToR**, on page 20
- **Configure BGP session for remote DCI Connectivity**, on page 22
- **Configure Network Virtualization Endpoint (NVE) Interface**, on page 24
- **Configure a Bridge Domain**, on page 27
- **Configure BGP Route Targets Import/Export Rules**, on page 28
- **Configure Ethernet Segment Identifier**, on page 30
- **Configure ICCP Group**, on page 32
- **Enable Flow-based Load Balancing**, on page 33
- **Example: All-Active Multi Homing with Anycast VTEP IP Address Configuration**, on page 34
- **Example: All-Active Multi Homing with Unique VTEP IP Address Configuration**, on page 35
- **EVPN MPLS Seamless Integration with VPLS**, on page 36
- **EVPN Single-Active Multi-Homing**, on page 48
- **Virtual Ethernet Segment (vES)**, on page 57
- **EVPN Anycast Gateway All-Active Static Pseudowire**, on page 63
- **CFM Support for EVPN**, on page 69
- **EVPN Multiple Services per Ethernet Segment**, on page 69
- **EVPN VXLAN Ingress Replication**, on page 73
- **EVPN Core Isolation Protection**, on page 83
- **EVPN Routing Policy**, on page 85
EVPN Overview

Ethernet VPN (EVPN) is a next generation solution that provide 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, PE's 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 multi-homing with per-flow load balancing.

The EVPN control-plane MAC learning has the following benefits:

- Eliminate flood and learn mechanism
- Fast-reroute, resiliency, and faster reconvergence when link to dual-homed server fails
- Enables load balancing of traffic to and from CEs that are multihomed to multiple PEs

The following EVPN modes are supported:

- Single homing - This enables you connect a customer edge (CE) device to one provider edge (PE) device.
- Multihoming - This enables you to connect a customer edge (CE) device to two or more provider edge (PE) devices to provide 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.
  - Active-Active - In active-active mode, all the PEs attached to the particular Ethernet-Segment is allowed to forward traffic to and from that 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 PE's flood list associated with an EVI.

- **Ethernet segment reachability**: In multi-home 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 (multi-homing) 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.
EVPN can operate in single homing or dual homing mode. Consider single homing scenario, when EVPN is enabled on PE, routes are 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 type-2 routes to other PEs. MAC routes are advertised to the other PEs using EVPN type-2 routes. In multi-homing scenarios Type 1, 3 and 4 are advertised to discover other PEs and their redundancy modes (single active or active-active). Use of Type-1 route 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. Type-4 route 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 is associated MPLS label. This EVPN MPLS label is used later by remote PEs when sending traffic destined to the advertised MAC address.

**EVPN Route Types**

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

**Table 1: EVPN Route Types**

<table>
<thead>
<tr>
<th>Route Type</th>
<th>Name</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Ethernet Auto-Discovery (AD) Route</td>
<td>Few routes sent per ES, carry the list of EVIs that belong to ES</td>
</tr>
<tr>
<td>2</td>
<td>MAC/IP Advertisement Route</td>
<td>Advertise MAC, address reachability, advertise IP/MAC binding</td>
</tr>
<tr>
<td>3</td>
<td>Inclusive Multicast Ethernet Tag Route</td>
<td>Multicast Tunnel End point discovery</td>
</tr>
<tr>
<td>4</td>
<td>Ethernet Segment Route</td>
<td>Redundancy group discovery, DF election</td>
</tr>
</tbody>
</table>
**Route Type 1: Ethernet Auto-Discovery (AD) Route**

The Ethernet (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.

**Route Type 2: MAC/IP Advertisement Route**

The host's IP and MAC addresses are advertised to the peers within NRLI. 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.

---

**Configure EVPN L2 Bridging Service**

Perform the following steps to configure EVPN L2 bridging service.

**SUMMARY STEPS**

1. configure
2. l2vpn
3. bridge-group bridge-group-name
4. bridge-domain bridge-domain-name
5. interface GigabitEthernet GigabitEthernet Interface Instance
6. evi ethernet vpn id
7. exit
8. exit
9. bridge-domain bridge-domain-name
10. interface GigabitEthernet GigabitEthernet Interface Instance
11. evi ethernet vpn id
12. Use the commit or end command.

**DETAILED STEPS**

**Step 1**

configure

**Example:**

```
RP/0/RSP0/CPU0:router# configure
```

Enters the global configuration mode.

**Step 2**

l2vpn
Example:
RP/0/RSP0/CPU0:router(config)# l2vpn
Enters the l2vpn configuration mode.

Step 3  bridge group  bridge-group-name

Example:
RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group 1
Enters the bridge group configuration mode.

Step 4  bridge-domain  bridge-domain-name

Example:
RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain 1-1
Enters the bridge domain configuration mode.

Step 5  interface GigabitEthernet  GigabitEthernet Interface Instance

Example:
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# interface GigabitEthernet 0/0/0/1.1
Enters interface configuration mode.

Step 6  evi  ethernet vpn id

Example:
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)# evi 1
Creates the ethernet VPN ID.

Step 7  exit

Example:
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac-evi)# exit
Exits the current configuration mode.

Step 8  exit

Example:
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# exit
Exits the current configuration mode.

Step 9  bridge-domain  bridge-domain-name

Example:
RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain 1-2
Enters the bridge domain configuration mode.

Step 10  interface GigabitEthernet  GigabitEthernet Interface Instance

Example:
EVPN Software MAC Learning

MAC learning is the method of learning the MAC addresses of all devices available in a VLAN. The MAC addresses learned on one device needs to be learned or distributed on the other devices in a VLAN. EVPN Native with 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.

Figure 2: EVPN Native with 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 DC1 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 DCI3 as a static remote MAC address.

Software and Hardware Support

The EVPN Native with Software MAC Learning feature is supported on Cisco ASR 9000 Series Routers that support Cisco IOS XR software and Cisco IOS XR 64-bit.

Configure EVPN Native with Software MAC Learning

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

/* 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-12vpn-bd)# bridge-domain EVPN_2001
RP/0/RSP0/CPU0:router(config-12vpn-bd)# interface TenGigE0/4/0/10.2001
RP/0/RSP0/CPU0:router(config-12vpn-bd)# interface BundleEther 20.2001
RP/0/RSP0/CPU0:router(config-12vpn-bd)# storm-control broadcast pps 10000
RP/0/RSP0/CPU0:router(config-12vpn-bd)# neighbor 20.20.20.20 pw-id 1020001
RP/0/RSP0/CPU0:router(config-12vpn-bd)# evi 2001
RP/0/RSP0/CPU0:router(config-12vpn-bd)# exit
RP/0/RSP0/CPU0:router(config-12vpn-bd)# exit
RP/0/RSP0/CPU0:router(config-l2vpn-bd)# exit

/* Configure advertisement of MAC routes, suppress unknown unicast, disable the control word, */
/* configure the flow label, configure BGP route-exchange using RT. */
RP/0/RSP0/CPU0:router(config)# evpn
RP/0/RSP0/CPU0:router(config-evpn)# evi 2001
/* Use the advertise-mac command to control the advertisement of MAC routes through BGP to
other neighbors. */
RP/0/RSP0/CPU0:router(config-evpn-evi)# advertise-mac
/* Use the unknown-unicast-suppress command to prevent unknown unicast traffic from going
to the MPLS core */
/* and then to all remote PE bridge-ports. */
RP/0/RSP0/CPU0:router(config-evpn-evi)# unknown-unicast-suppress
/* Use the control-word-disable command to prevent the control word from being sent */
/* in the packet that is sent to MPLS core. The control word functionality is enabled by
default. */
RP/0/RSP0/CPU0:router(config-evpn-evi)# control-word-disable
/* Use the load-balance flow label static command to add additional flow label header to
the packet */
/* that is sent to MPLS core. The loadbalance flow functionality is disabled by default. */
RP/0/RSP0/CPU0:router(config-evpn-evi)# load-balance flow label static
/* Perform the following steps to configure BGP route-exchange using RT */
RP/0/RSP0/CPU0:router(config-evpn-evi)# bgp
RP/0/RSP0/CPU0:router(config-evpn-evi)# route-target import 200:101
RP/0/RSP0/CPU0:router(config-evpn-evi)# route-target export 200:101
/* Configure address family session in BGP. */
RP/0/RSP0/CPU0:router(config)# configure
RP/0/RSP0/CPU0:router(config)# router bgp 200
RP/0/RSP0/CPU0:router(config-bgp)# bgp router-id 40.40.40.40
RP/0/RSP0/CPU0:router(config-bgp)# address-family 12vpn evpn
RP/0/RSP0/CPU0:router(config-bgp)# neighbor 10.10.10.10
Supported Modes for EVPN Native with Software MAC Learning

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

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

Single Home Device or Single Home Network

The following section describes how you can configure EVPN Native with Software MAC Learning feature in single home device or single home network:

*Figure 3: Single Home Device or Single Home Network (SHD/SHN)*

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

/* 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 BundleEther1.2001
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 40.40.40.40
RP/0/RSP0/CPU0:router#(config-bgp)# address-family l2vpn evpn
RP/0/RSP0/CPU0:router#(config-bgp)# neighbor 10.10.10.10
```
Running Configuration

l2vpn
bridge group EVPN_ALL_ACTIVE
    bridge-domain EVPN_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

<table>
<thead>
<tr>
<th>Ethernet Segment Id</th>
<th>Interface</th>
<th>Nexthops</th>
</tr>
</thead>
<tbody>
<tr>
<td>N/A</td>
<td>Te0/4/0/10</td>
<td>20.20.20.20</td>
</tr>
</tbody>
</table>

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:
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:

```plaintext
/* 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 BundleEther1.2001
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
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)# router-id 209.165.200.227
RP/0/RSP0/CPU0:router(config-bgp)# address-family l2vpn evpn
RP/0/RSP0/CPU0:router(config-bgp-nbr)# 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 Link Aggregation Control Protocol (LACP) bundle. */

RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface Bundle-Ether1 300
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-Ether1.2001 l2transport
RP/0/RSP0/CPU0:router(config-if)# encapsulation dot1aq 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**

```
12vpn
bridge group EVPN_ALL_ACTIVE
 bridge-domain EVPN_2001
 interface Bundle-Ether1.2001
   !
   evi 2001
   !
 evpn
 evi 2001
   !
   advertise-mac
   !
 interface bundle-ether1
   ethernet-segment
     identifier type 0 01.11.00.00.00.00.00.00.01
     !
     !
 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-Ether1.2001 l2transport
 encapsulation dot1aq 2001
 rewrite ingress tag pop 1 symmetric
   !
```

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

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

<table>
<thead>
<tr>
<th>Ethernet Segment Id</th>
<th>Interface</th>
<th>Nexthops</th>
</tr>
</thead>
<tbody>
<tr>
<td>0100.211b.fce5.df00.0b00</td>
<td>BE11</td>
<td>10.10.10.10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>209.165.201.1</td>
</tr>
</tbody>
</table>

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

---

**Dual Home Device—Single-Active Load Balancing**

The following section describes how you can configure EVPN Native with Software MAC Learning feature in dual home device in single-active load balancing mode:

*Figure 5: Dual Home Device (DHD)—Single-Active Load Balancing*

Single-active load balancing also is known as Active/Active per Service (AApS).

Identical ESI are configured on both EVPN PEs. In the CE, separate bundles or independent physical interfaces are configured towards two EVPN PEs. In this mode, the MAC address that is learnt is stored on both PE1 and PE2. Only one PE can forward traffic within the EVI at a given time.

**Configure EVPN in Dual Home Device—Single-Active Mode**

```
/* Configure bridge domain. */

RP/0/RSP0/CPU0:router(config)# l2vpn
```
EVPN Features

Configure EVPN in Dual Home Device—Single-Active Mode

RP/0/RSP0/CPU0:router(config-12vpn)\# bridge group EVPN_ALL_ACTIVE
RP/0/RSP0/CPU0:router(config-12vpn-bg)\# bridge-domain EVPN_2001
RP/0/RSP0/CPU0:router(config-12vpn-bg-bd)\# interface BundleEther1.2001
RP/0/RSP0/CPU0:router(config-12vpn-bg-bd)\# evi 2001

\/* Configure VLAN Header Rewrite (Single-tagged sub-interface).*/
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# interface bundle-Ether1.21 l2transport
RP/0/RSP0/CPU0:router(config)# lACP switchover suppress-flaps 300
RP/0/RSP0/CPU0:router(config)# exit
RP/0/RSP0/CPU0:router# interface Bundle-Ether1.2001 l2transport
RP/0/RSP0/CPU0:router(config)# encapsulation dot1q 10
RP/0/RSP0/CPU0:router(config)# rewrite ingress tag pop 1 symmetric

\/* 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)# advertise-mac

\/* Configure load balancing. */
RP/0/RSP0/CPU0:router(config)# evpn
RP/0/RSP0/CPU0:router(config-evpn)# evi 2001
RP/0/RSP0/CPU0:router(config-evpn)# 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)# load-balancing-mode single-active
RP/0/RSP0/CPU0:router(config-evpn-ac-es)# identifier type 0 12.12.00.00.00.00.00.00.02
RP/0/RSP0/CPU0:router(config-evpn-ac-es)# bgp route-target 1212.0000.0002

\/* 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-router)# bgp router-id 40.40.40.40
RP/0/RSP0/CPU0:router(config-router)# address-family l2vpn evpn
RP/0/RSP0/CPU0:router(config-router)# neighbor 10.10.10.10
RP/0/RSP0/CPU0:router(config-router)# remote-as 200
RP/0/RSP0/CPU0:router(config-router)# description MPLSFACING-PEER
RP/0/RSP0/CPU0:router(config-router)# update-source Loopback 0
RP/0/RSP0/CPU0:router(config-router)# address-family l2vpn evpn

Verification
Verify EVPN in dual home devices in Single-Active mode.

RP/0/RSP0/CPU0:router# show evpn ethernet-segment int bundleEther 21 carving detail

... Ethernet Segment Id Interface Nexthops
----- --------------- --------
0012.1200.0000.0000.0002 BE21 10.10.10.10 30.30.30.30

ESI type : 0
Value : 12.1200.0000.0000.0002
ES Import RT : 1212.0000.0000 (from ESI)
Source MAC : 0000.0000.0000 (N/A)

Topology :
- Operational : MHN
- Configured : Single-active (AAPS)
Primary Services : Auto-selection
Secondary Services: Auto-selection

Service Carving Results:
- Forwarders : 2
- Elected : 1
- EVI E : 500
- Not Elected : 1
- EVI NE : 501

Verify EVPN Native with Software MAC Learning

Verify the packet drop statistics.

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 EVFNs:
- 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 875013022336
    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

<table>
<thead>
<tr>
<th>Neighbor IP</th>
<th>vpn-id</th>
</tr>
</thead>
<tbody>
<tr>
<td>20.20.20.20</td>
<td>2001</td>
</tr>
<tr>
<td>30.30.30.30</td>
<td>2001</td>
</tr>
</tbody>
</table>

Verify the BGP L2VPN EVPN summary.

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

<table>
<thead>
<tr>
<th>Neighbor</th>
<th>Spk</th>
<th>AS</th>
<th>MsgRcvd</th>
<th>MsgSent</th>
<th>TblVer</th>
<th>InQ</th>
<th>OutQ</th>
<th>Up/Down</th>
<th>St/PfxRcd</th>
</tr>
</thead>
<tbody>
<tr>
<td>20.20.20.20</td>
<td>0</td>
<td>200</td>
<td>216739</td>
<td>229871</td>
<td>200781341</td>
<td>0</td>
<td>0</td>
<td>3d00h</td>
<td>348032</td>
</tr>
</tbody>
</table>
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/RSP0/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
0000.2001.5555 dynamic Te0/0/0/2/0.2001 N/A 11 Jan 14:37:22
N/A <-- local dynamic
00bb.2001.0001 dynamic Te0/0/0/2/0.2001 N/A 11 Jan 14:37:22
N/A
0000.2001.1111 EVPN BD id: 1110 N/A N/A
N/A <-- remote static
00a9.2002.0001 EVPN BD id: 1110 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
EVI MAC address IP address Nexthop Label
---- -------------- ------- -----
2001 00a9.2002.0001 :: 10.10.10.10 34226 <-- Remote MAC
2001 00a9.2002.0001 :: 30.30.30.30 34202
2001 0000.2001.5555 20.1.5.55 TenGigE0/0/0/2/0.2001 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 :: 30.30.30.30 34202

EVPN Features
Multi-paths Resolved: True --- aliasing to two remote PE with All-Active load balancing

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
*>[2][0][48][00a9.2002.00be][0]/104 10.10.10.10 100
* i 30.30.30.30 100 0 i
```

EVPN VXLAN Layer 2 Data Center Interconnect Gateway

The Cisco ASR 9000 Series Routers serve as a Data Center Interconnect (DCI) Layer 2 gateway to provide Layer 2 connectivity between EVPN VXLAN based data centers, over a MPLS-based L2VPN network. The data centers are connected through the intermediate service provider network. The EVPN VXLAN enabled data centers use EVPN control plane for distributing Layer 2 forwarding information from one data center to another data center. This feature provides redundancy, resiliency, and ease of provisioning.

The EVPN VXLAN layer 2 DCI gateway feature supports these functions:

- VXLAN access for single homing
- VXLAN access for all-active multi homing with anycast VXLAN Terminal EndPoint (VTEP) IP address
- VXLAN access for all-active multi homing with unique VTEP IP address
- EVPN ESI Multipath with VXLAN encapsulation

All-Active Multi Homing with Anycast VTEP IP Address

The DCIs use the same anycast VTEP IP address for all-active multi-homing with anycast VTEP IP address. Consider the following topology where Top of Racks (ToRs) are connected to the DCIs using multiple paths. The traffic passes from ToRs to DCIs through multiple physical paths and uses anycast IP address for load balancing. DCI1 and DCI2 advertise MAC routes to ToRs using the same anycast IP address as that of the next-hop. So, the ToR sends the traffic to the same anycast IP address of the DCIs, and uses IGP ECMP for load balancing. A virtual PortChannel (vPC) allows ToR1 and ToR2 to have the same IP configuration. ToR1 and ToR2 advertise MAC routes to DCIs using the same IP address as that of the next-hop. So, the DCI sends the traffic to the same IP address of the ToRs, and uses IGP ECMP for load balancing. The DCI sends the traffic to the remote data center through MPLS forwarding.
All-Active Multi Homing with Unique VTEP IP Address

The DCIs do not share anycast VTEP IP address for all-active multi homing with unique VTEP IP address. Each DCI uses a unique VTEP IP address. Consider the following topology where ToR receives the MAC routes from DCIs. Each MAC route has a unique next-hop. Because both DCI1 and DCI2 advertise routes for the same MAC with different next-hops, ToR has two equal cost next-hops for the same MAC. When ToR sends the traffic to the MAC, ToR load balances the traffic on both next-hops.

EVPN ESI Multipath for VxLAN - EVI Based Load balancing

The EVPN Ethernet Segment Identifier (ESI) Multipath feature supports multi-path traffic to active-active dual-homed TORs and DCIs to provide redundant connectivity within the data center. ESI multi paths are discovered by the ASR9k DCI router through EVPN signalling. The path selection is based on Ethernet Segment Identifier (ESI) and EVPN instance (EVI). To resolve paths for MAC routes received, use Ethernet A-D routes per ES (ES-EAD) and Ethernet A-D routes per EVI (EVI-EAD) as specified in RFC 7432.

Consider the following topology where DCIs receive the MAC routes from ToRs and each MAC route has a next-hop for each ToR. Similarly, DCIs advertise MAC routes with different next-hops to ToRs. When DCI sends the traffic to VM, which is behind a pair of ToRs, there are two paths (ToR) for every MAC. The DCI load balances the traffic on the two paths. The selection of path is based on EVI. For example, DCI1 and DCI2 selects ToR1 for all traffic destined to the MAC address learnt on EVI1; DCI1 and DCI2 selects ToR2 for all traffic destined to the MAC address learnt on EVI2.
The EVPN Ethernet Segment Identifier (ESI) Multipath for VxLAN feature supports flow-based load balancing to forward the traffic between Top of Racks (ToRs) and Data Center Interconnect (DCI), and between the source and remote DCIs. A flow is identified either by the source and destination IP address of the traffic, or the source and destination MAC address of the traffic.

In Release 6.2.1, the default load balancing mode is flow-based. You can change the load balancing mode based on per EVI. See Configure Network Virtualization Endpoint (NVE) Interface, on page 24 task to change the load balancing mode based on per EVI.

In Release 6.1.2, only per EVI-based load balancing was supported. Starting from Release 6.2.1, both flow-based load balancing and per EVI based load balancing are supported. The following table shows the support matrix:

Table 2: Support Matrix for EVPN ESI Multipath for VxLAN Load Balancing

<table>
<thead>
<tr>
<th>Line Card</th>
<th>Release 6.1.2</th>
<th>Release 6.2.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASR 9000 Enhanced Ethernet Line Card</td>
<td>Supports only per EVI-based load balancing</td>
<td>Supports only per EVI-based load balancing</td>
</tr>
</tbody>
</table>

The unknown unicast flooding on traffic received from VxLAN segment is supported. In Release 6.2.1, by default, the unknown unicast flooding on traffic received from VxLAN segment is enabled. To disable the unknown unicast flooding, use the `suppress-unknown-unicast-flooding` command. See Configure Network Virtualization Endpoint (NVE) Interface, on page 24 task to disable unknown unicast flooding on traffic received from VxLAN segment.

In Release 6.1.2, by default, the unknown unicast flooding on traffic received from VxLAN segment is disabled.
Table 3: Support Matrix for Unknown Unicast Flooding

<table>
<thead>
<tr>
<th>Release</th>
<th>Unknown Unicast Flooding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release 6.1.2</td>
<td>The unknown unicast flooding on traffic received from VxLAN segment is disabled.</td>
</tr>
<tr>
<td>Release 6.2.1</td>
<td>The unknown unicast flooding on traffic received from VxLAN segment is enabled. To disable, use the suppress-unknown-unicast-flooding command.</td>
</tr>
</tbody>
</table>

Configure EVPN VXLAN Layer 2 Data Center Interconnect Gateway

Perform the following tasks to configure EVPN VXLAN Layer 2 Data Center Interconnect Gateway.

If you want to configure EVPN ESI Multipath feature, do not configure anycast IP address, the remaining configuration tasks remain the same.

Configure L2 EVPN Address Family under BGP Routing Process

Perform this task to enable EVPN address family under BGP routing process.

**SUMMARY STEPS**

1. configure
2. router bgp <asn_id>
3. nsr
4. bgp graceful-restart
5. bgp router-id <ip-address>
6. address-family l2vpn evpn
7. Use the commit or end command.

**DETAILED STEPS**

**Step 1**

configure

Example:

```
RP/0/RSP0/CPU0:router# configure
```

Enters the global configuration mode.

**Step 2**

router bgp <asn_id>

Example:

```
RP/0/RSP0/CPU0:router(config)# router bgp 100
```

Specifies the BGP AS number and enters the BGP configuration mode, allowing you to configure the BGP routing process.
Configure the Routing Sessions Between the DCI and ToR

Perform this task to configure the routing sessions between the DCI and ToR.

**SUMMARY STEPS**

1. `configure`
2. `router bgp asn_id`
3. `neighbor ip-address`
4. `remote-as autonomous-system-number`
5. `ebgp-multihop maximum hop count`
6. `update-source loopback`
7. address-family l2vpn evpn
8. import stitching-rt reoriginate
9. route-policy route-policy-name in
10. encapsulation-type type
11. route-policy route-policy-name out
12. advertise l2vpn evpn re-originated stitching-rt
13. Use the commit or end command.

DETAILED STEPS

Step 1 configure
Example:
RP/0/RSP0/CPU0:router# configure
Enters the global configuration mode.

Step 2 router bgp asn_id
Example:
RP/0/RSP0/CPU0:router(config)# router bgp 100
Specifies the BGP AS number and enters the BGP configuration mode, allowing you to configure the BGP routing process.

Step 3 neighbor ip-address
Example:
RP/0/RSP0/CPU0:router(config-bgp)# neighbor 209.165.200.225
Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address 209.165.200.225 as a BGP peer.

Step 4 remote-as autonomous-system-number
Example:
RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 2000
Creates a neighbor and assigns it a remote autonomous system number.

Step 5 ebgp-multihop maximum hop count
Example:
RP/0/RSP0/CPU0:router(config-bgp-nbr)# ebgp-multihop 255
Enables multihop peerings with external BGP neighbors.

Step 6 update-source loopback
Example:
RP/0/RSP0/CPU0:router(config-bgp-nbr)# update-source loopback1
Allows BGP sessions to use the primary IP address from a particular interface as the local address.

Step 7 address-family l2vpn evpn
Example:
RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family l2vpn evpn
Configures EVPN address family.

**Step 8** import stitching-rt reoriginate

**Example:**
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# import stitching-rt reoriginate

Enables import of routing information from BGP EVPN NLRIs that has route target identifier matching the stitching route target identifier and exports this routing information after re-origination to the L2VPN BGP neighbor.

**Step 9** route-policy route-policy-name in

**Example:**
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# route-policy pass-all in

Applies the route policy to inbound unicast routes.

**Step 10** encapsulation-type type

**Example:**
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# encapsulation-type vxlan

Configures VXLAN as encapsulation type.

**Step 11** route-policy route-policy-name out

**Example:**
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# route-policy pass-all out

Applies the route policy to outbound unicast routes.

**Step 12** advertise l2vpn evpn re-originated stitching-rt

**Example:**
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# advertise l2vpn evpn re-originated stitching-rt

Configures advertisement of L2VPN EVPN routes to be received from the L2VPN BGP neighbor.

**Step 13** Use the commit or end command.

**commit** - Saves the configuration changes and remains within the configuration session.

**end** - Prompts user to take one of these actions:

- **Yes** - Saves configuration changes and exits the configuration session.
- **No** - Exits the configuration session without committing the configuration changes.
- **Cancel** - Remains in the configuration mode, without committing the configuration changes.

---

**Configure BGP session for remote DCI Connectivity**

Perform this task to configure BGP session for remote DCI connectivity.
SUMMARY STEPS

1. configure
2. router bgp asn_id
3. neighbor ip-address
4. remote-as autonomous-system-number
5. update-source loopback
6. address-family l2vpn evpn
7. import re-originate stitching-rt
8. advertise l2vpn evpn re-originated
9. Use the commit or end command.

DETAILED STEPS

Step 1 configure

Example:
RP/0/RSP0/CPU0:router# configure

Enters the global configuration mode.

Step 2 router bgp asn_id

Example:
RP/0/RSP0/CPU0:router(config)# router bgp 200

Specifies the BGP AS number and enters the BGP configuration mode, allowing you to configure the BGP routing process.

Step 3 neighbor ip-address

Example:
RP/0/RSP0/CPU0:router(config-bgp)# neighbor 209.165.201.1

Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address 209.165.201.1 as a BGP peer.

Step 4 remote-as autonomous-system-number

Example:
RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 100

Creates a neighbor and assigns it a remote autonomous system number.

Step 5 update-source loopback

Example:
RP/0/RSP0/CPU0:router(config-bgp-nbr)# update-source loopback2

Allows BGP sessions to use the primary IP address from a particular interface as the local address.

Step 6 address-family l2vpn evpn

Example:
RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family l2vpn evpn
Configures EVPN address family.

**Step 7** import re-originate stitching-rt

*Example:*

```
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# import re-originate stitching-rt
```

Enables import of routing information from BGP EVPN NLRIs that have route target identifier matching the stitching route target identifier, and exports this routing information after re-origination to the L2VPN BGP neighbor.

**Step 8** advertise l2vpn evpn re-originated

*Example:*

```
RP/0/RSP0/CPU0:router(config-bgp-nbr-af)# advertise l2vpn evpn re-originated
```

Configures the advertisement of L2VPN EVPN routes to be received from the L2VPN BGP neighbor.

**Step 9** Use the `commit` or `end` command.

- **commit** - Saves the configuration changes and remains within the configuration session.
- **end** - Prompts user to take one of these actions:
  - **Yes** - Saves configuration changes and exits the configuration session.
  - **No** - Exits the configuration session without committing the configuration changes.
  - **Cancel** - Remains in the configuration mode, without committing the configuration changes.

### Configure Network Virtualization Endpoint (NVE) Interface

Perform this task to create an NVE interface and configure it as a VXLAN Tunnel EndPoint (VTEP) for VxLAN.

**SUMMARY STEPS**

1. configure
2. interface nve nve-identifier
3. source-interface loopback loopback-interface-identifier
4. anycast source-interface loopback loopback-interface-identifier
5. redundancy
6. backbone vxlans
7. iccp group group number
8. exit
9. backbone mpls
10. iccp group group number
11. exit
12. exit
13. member vni vni_number
14. load-balance per-evi
### DETAILED STEPS

**Step 1**
configure

**Example:**
```
RP/0/RSP0/CPU0:router# configure
```
Enters the global configuration mode.

**Step 2**
interface nve nve-identifier

**Example:**
```
RP/0/RSP0/CPU0:router(config)# interface nve 1
```
Creates the NVE interface and enters the NVE interface configuration sub-mode.

**Step 3**
source-interface loopback loopback-interface-identifier

**Example:**
```
RP/0/RSP0/CPU0:router(config-if)# source-interface loopback 1
```
Sets a loopback interface as the source interface for the VTEP.

**Step 4**
anycast source-interface loopback loopback-interface-identifier

**Example:**
```
RP/0/RSP0/CPU0:router(config-if)# anycast source-interface loopback 1
```
Configures anycast mode parameters and source interface for the anycast mode. Anycast IP address is used for BGP next hop on the fabric side. If you want to configure the ESI multipath feature, do not configure anycast IP address.

**Step 5**
redundancy

**Example:**
```
RP/0/RSP0/CPU0:router(config-if)# redundancy
```
Configures the redundancy path.

**Step 6**
backbone vxlan

**Example:**
```
RP/0/RSP0/CPU0:router(config-nve-red)# backbone vxlan
```
Configures Inter-Chassis Communication Protocol (ICCP) VXLAN backbone.

**Step 7**
iccp group group number

**Example:**
```
RP/0/RSP0/CPU0:router(config-nve-red-backbone-vxlan)# iccp group 11
```
Configures the ICCP group number.
Configure Network Virtualization Endpoint (NVE) Interface

Step 8  
exit

Example:
RP/0/RSP0/CPU0:router(config-nve-red-backbone-vxlan)# exit

Exits the backbone-vxlan submode and returns to redundancy submode.

Step 9  
backbone mpls

Example:
RP/0/RSP0/CPU0:router(config-nve-red)# backbone mpls

Configures ICCP MPLS backbone.

Step 10  
iccp group group number

Example:
RP/0/RSP0/CPU0:router(config-nve-red-backbone-mpls)# iccp group 12

Configures ICCP group number for MPLS backbone.

Step 11  
exit

Example:
RP/0/RSP0/CPU0:router(config-nve-red-backbone-mpls)# exit

Exits the backbone-mpls submode and returns to redundancy submode.

Step 12  
exit

Example:
RP/0/RSP0/CPU0:router(config-nve-red)# exit

Exits the redundancy submode and returns to interface submode.

Step 13  
member vni vni_number

Example:
RP/0/RSP0/CPU0:router(config-nve)# member vni 1

Associates a single VxLAN with the NVE interface using the VxLAN Network Identifier (VNI) and specifies a multicast address associated with this VNI.

Step 14  
load-balance per-evi

Example:
RP/0/RSP0/CPU0:router(config-nve-vni)# load-balance per-evi

Configures per-evi load balance mode (default is per-flow).

Step 15  
suppress-unknown-unicast-flooding

Example:
RP/0/RSP0/CPU0:router(config-nve-vni)# suppress-unknown-unicast-flooding

Configures the suppression of unknown unicast flooding.

Step 16  
mcast-group ip_address

Example:
RP/0/RSP0/CPU0:router(config-nve-vni)# mcast-group 209.165.202.129
Specifies a multicast address associated with the VNI.

Step 17  
**host-reachability protocol** protocol

**Example:**
RP/0/RSP0/CPU0:router(config-nve-vni)# host-reachability protocol bgp
Configures the BGP control protocol for VxLAN tunnel endpoint reachability.

Step 18  
Use the **commit** or **end** command

**commit** - Saves the configuration changes and remains within the configuration session.

**end** - Prompts user to take one of these actions:
- **Yes** - Saves configuration changes and exits the configuration session.
- **No** - Exits the configuration session without committing the configuration changes.
- **Cancel** - Remains in the configuration mode, without committing the configuration changes.

---

**Configure a Bridge Domain**

Perform the following steps to configure the bridge domain on the DCI Gateway.

**SUMMARY STEPS**

1. **configure**
2. **l2vpn**
3. **bridge group** bridge-group-name
4. **bridge-domain** bridge-domain-name
5. **evi** ethernet vpn id
6. **exit**
7. **member vni** vxlan-id
8. Use the **commit** or **end** command.

**DETAILED STEPS**

**Step 1**  
**configure**

**Example:**
RP/0/RSP0/CPU0:router# configure
Enters the global configuration mode.

**Step 2**  
**l2vpn**

**Example:**
RP/0/RSP0/CPU0:router(config)# l2vpn
Enters the l2vpn configuration mode.

**Step 3**  
**bridge group** *bridge-group-name*  
**Example:**  
RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group bg1  
Enters the bridge group configuration mode.

**Step 4**  
**bridge-domain** *bridge-domain-name*  
**Example:**  
RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain bd1  
Enters the bridge domain configuration mode.

**Step 5**  
**evi ethernet vpn id**  
**Example:**  
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# evi 1  
Creates the ethernet VPN ID.

**Step 6**  
**exit**  
**Example:**  
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-evi)# exit  
Exits the EVI configuration mode and returns to bridge domain configuration mode.

**Step 7**  
**member vni** *vxlan-id*  
**Example:**  
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# member vni 1  
Associates a member VNI with the bridge domain.

**Step 8**  
Use the **commit** or **end** command.  
**commit**  - Saves the configuration changes and remains within the configuration session.  
**end**  - Prompts user to take one of these actions:  
  - **Yes**  - Saves configuration changes and exits the configuration session.  
  - **No**  - Exits the configuration session without committing the configuration changes.  
  - **Cancel**  - Remains in the configuration mode, without committing the configuration changes.

---

**Configure BGP Route Targets Import/Export Rules**

By default, these parameters are auto-derived from the DCI's configuration:

- Route Distinguisher (RD) for global Ethernet Segment table
  
Default: Auto-generated RD based on loopback IP address
- EVI's BGP Route Distinguisher (RD)
  Default: Auto-generated RD based on loopback IP address
- EVI's BGP Route Target. Default: Auto-generated RT based on EVI ID

Perform this task to overwrite the auto-generated BGP RD/RT values and define route targets to be used for import and export of forwarding information.

**SUMMARY STEPS**

1. configure
2. evpn
3. bgp
4. rd { 2-byte as_number | 4-byte as_number | IP_address | none } : { nn }
5. exit
6. evi evi_id
7. bgp
8. route-target import { 2-byte as_number | 4-byte as_number | IP_address | none } : { nn } [stitching]
9. route-target export { 2-byte as_number | 4-byte as_number | IP_address | none } : { nn } [stitching]
10. Use the commit or end command.

**DETAILED STEPS**

**Step 1**
configure

*Example:*
RP/0/RSP0/CPU0:router# configure

Enters the global configuration mode.

**Step 2**
evpn

*Example:*
RP/0/RSP0/CPU0:router(config)# evpn

Enters EVPN configuration mode.

**Step 3**
bgp

*Example:*
RP/0/RSP0/CPU0:router(config-evpn)# bgp

Enters EVPN BGP configuration mode and configures static BGP settings for the Ethernet Segment ES:GLOBAL EVI, which is used for handling ES routes.

**Step 4**
 rd { 2-byte as_number | 4-byte as_number | IP_address | none } : { nn }

*Example:*
RP/0/RSP0/CPU0:router(config-evpn-bgp)# rd 200:50

Configures the route distinguisher.

**Step 5**
exit
Configure Ethernet Segment Identifier

Perform this task to configure Ethernet Segment Identifier (ESI).

SUMMARY STEPS

1. configure
2. evpn
3. interface nve nve-identifier
4. ethernet-segment
5. identifier type esi-type esi-identifier
6. bgp route-target route target value
7. Use the commit or end command

DETAILED STEPS

Step 1 configure
Example:
RP/0/RSP0/CPU0:router# configure
Enters the global configuration mode.

Step 2 evpn
Example:
RP/0/RSP0/CPU0:router# evpn
Enters EVPN configuration mode.

Step 3 interface nve nve-identifier
Example:
RP/0/RSP0/CPU0:router(config-evpn)# interface nve 1
Creates the NVE interface and enters the NVE interface configuration sub-mode.

Step 4 ethernet-segment
Example:
RP/0/RSP0/CPU0:router(config-evpn-ac)# ethernet-segment
Enters the EVPN ethernet-segment configuration mode.

Step 5 identifier type esi-type esi-identifier
Example:
RP/0/RSP0/CPU0:router(config-evpn-ac-es)# identifier type 20 88.00.00.00.00.00.00.00.01
Configures Ethernet Segment Identifier.

Step 6 bgp route-target route target value
Example:
RP/0/RSP0/CPU0:router(config-evpn-ac-es)# bgp route-target 8888.0000.0001
Configures the BGP import route-target for the Ethernet-Segment.

Step 7 Use the commit or end command
commit - Saves the configuration changes and remains within the configuration session.
dead - Prompts user to take one of these actions:
Configure ICCP Group

Perform this task to configure Inter Chassis Communication Protocol (ICCP) parameters.

Configure ICCP group for core interface tracking. If all interfaces are down, the DCI is isolated from the core/fabric network. The associated nve interface is brought down, and BGP NLRIs are withdrawn.

SUMMARY STEPS

1. configure
2. redundancy
3. iccp group group number
4. mode singleton
5. backbone
6. interface GigabitEthernet GigabitEthernet Interface Instance
7. Use the commit or end command

DETAILED STEPS

Step 1  configure
Example:
RP/0/RSP0/CPU0:router# configure
Enters the global configuration mode.

Step 2  redundancy
Example:
RP/0/RSP0/CPU0:router(config)# redundancy
Enters redundancy configuration mode.

Step 3  iccp group group number
Example:
RP/0/RSP0/CPU0:router(config-redundancy)# iccp group 11
Configures ICCP group number.

Step 4  mode singleton
Example:
RP/0/RSP0/CPU0:router(config-redundancy-iccp-group)# mode singleton
Enables to run the group in singleton mode.

**Step 5**

**backbone**

**Example:**

```
RP/0/RSP0/CPU0:router(config-redundancy-iccp-group)# backbone
```

Configures ICCP backbone interface.

**Step 6**

**interface GigabitEthernet** _GigabitEthernet Interface Instance_

**Example:**

```
RP/0/RSP0/CPU0:router(config-redundancy-group-iccp-backbone)# interface GigabitEthernet 0/2/0/12
```

Configures GigabitEthernet interface.

**Step 7**

Use the **commit** or **end** command

- **commit** - Saves the configuration changes and remains within the configuration session.
- **end** - Prompts user to take one of these actions:
  - **Yes** - Saves configuration changes and exits the configuration session.
  - **No** - Exits the configuration session without committing the configuration changes.
  - **Cancel** - Remains in the configuration mode, without committing the configuration changes.

---

### Enable Flow-based Load Balancing

Perform this task to enable flow-based load balancing.

**SUMMARY STEPS**

1. configure
2. l2vpn
3. load-balancing flow {src-dst-mac | src-dst-ip}
4. Use the **commit** or **end** command.

**DETAILED STEPS**

**Step 1**

**configure**

**Example:**

```
RP/0/RSP0/CPU0:router# configure
```

Enters the Global Configuration mode.

**Step 2**

**l2vpn**

**Example:**
Example: All-Active Multi Homing with Anycast VTEP IP Address Configuration

The following example shows the all-active multi homing with anycast VTEP IP address configuration:

```plaintext
interface nve1
source-interface loopback1
anycast source-interface loopback2
member vni 5100
mcast-address 239.1.1.1
host-reachability protocol bgp
!
evpn
evi 10
bgp
route-target import 100:10
route-target import 200:5100 stitching
route-target export 200:5100 stitching
!
l2vpn
bridge group DCI
bridge-domain V1
  evpn evi 10
    member vni 5100
!
router bgp 100
bgp router-id 209.165.200.226
address-family l2vpn evpn

!
neighbor 209.165.201.2
```

Step 3 load-balancing flow {src-dst-mac | src-dst-ip}

Example:

```plaintext
RP/0/RSP0/CPU0:router(config-l2vpn)# load-balancing flow src-dst-ip
```

Enables flow-based load balancing.

Step 4 Use the **commit** or **end** command.

**commit** - Saves the configuration changes and remains within the configuration session.

**end** - Prompts user to take one of these actions:

- **Yes** - Saves configuration changes and exits the configuration session.
- **No** - Exits the configuration session without committing the configuration changes.
- **Cancel** - Remains in the configuration mode, without committing the configuration changes.
remote-as 100
description core-facing
update-source Loopback1
address-family l2vpn evpn
  import re-originate stitching-rt
  advertise l2vpn evpn re-originated
! neighbor 209.165.202.130
  remote-as 200
ebgp-multihop 255
  update-source Loopback1
  address-family l2vpn evpn
    import stitching-rt re-originate
    route-policy passall in
    encapsulation-type vxlan
    route-policy passall out
    advertise l2vpn evpn re-originated stitching-rt
!

Example: All-Active Multi Homing with Unique VTEP IP Address Configuration

The following example shows the all-active multi homing with unique VTEP IP address configuration:

```
interface nve1
  source-interface loopback1
  member vni 5100
    mcast-address 239.1.1.1
    host-reachabilty protocol bgp
! evpn
evi 10
  bgp
    route-target import 100:10
    route-target import 200:5100 stitching
    route-target export 200:5100 stitching
! l2vpn
  bridge group DCI
  bridge-domain V1
    evpn evi 10
      member vni 5100
! router bgp 100
  bgp router-id 209.165.200.226
  address-family l2vpn evpn
! neighbor 209.165.201.2
  remote-as 100
  description core-facing
  update-source Loopback1
  address-family l2vpn evpn
    import re-originate stitching-rt
    multipath
    advertise l2vpn evpn re-originated
! neighbor 209.165.202.130
  remote-as 200
```
EVPN MPLS Seamless Integration with VPLS

VPLS is a widely-deployed L2VPN technology. As service providers are looking to adopt EVPN on their existing VPLS networks, it is required to provide a mechanism by which EVPN can be introduced without a software upgrade. The EVPN MPLS Seamless Integration with VPLS feature allows EVPN service introduced gradually in the network on a few PE nodes at a time. It eliminates the need to network wide software upgrade at the same time. This feature allows a VPLS service migrated to EVPN service. This feature allows for staged migration where new EVPN sites can be provisioned on existing VPLS enabled PEs. This feature also allows for the co-existence of PE nodes running EVPN and VPLS for the same VPN instance. This allows VPLS or legacy network to be upgraded to the next generation EVPN network without service disruption.

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

EVPN Features
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 up. 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 43 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.

SUMMARY STEPS

1. `configure`
2. `router bgp asn_id`
3. `nsr`
4. `bgp graceful-restart`
5. `bgp router-id ip-address`
6. `address-family l2vpn evpn`
7. `exit`
8. neighbor ip-address
9. remote-as autonomous-system-number
10. update-source loopback
11. address-family l2vpn evpn
12. Use the commit or end command.

DETAILED STEPS

Step 1 configure
Example:
RP/0/RSP0/CPU0:router# configure
Enters the global configuration mode.

Step 2 router bgp asn_id
Example:
RP/0/RSP0/CPU0:router(config)# router bgp 65530
Specifies the BGP AS number and enters the BGP configuration mode, allowing you to configure the BGP routing process.

Step 3 nsr
Example:
RP/0/RSP0/CPU0:router(config-bgp)# nsr
Enables non-stop routing.

Step 4 bgp graceful-restart
Example:
RP/0/RSP0/CPU0:router(config-bgp)# bgp graceful-restart
Enables graceful restart on the router.

Step 5 bgp router-id ip-address
Example:
RP/0/RSP0/CPU0:router(config-bgp)# bgp router-id 200.0.1.1
Configures the router with a specified router ID.

Step 6 address-family l2vpn evpn
Example:
RP/0/RSP0/CPU0:router(config-bgp)# address-family l2vpn evpn
Enables EVPN address family globally under BGP routing process and enters EVPN address family configuration submode.

Step 7 exit
Example:
RP/0/RSP0/CPU0:router(config-bgp-sf)# exit
Exits the current configuration mode.

**Step 8**

`neighbor ip-address`

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp)# neighbor 200.0.4.1
```

Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address 200.0.4.1 as a BGP peer.

**Step 9**

`remote-as autonomous-system-number`

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-nbr)# remote-as 65530
```

Creates a neighbor and assigns it a remote autonomous system number.

**Step 10**

`update-source loopback`

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-nbr)# update-source Loopback0
```

Allows BGP sessions to use the primary IP address from a particular interface as the local address.

**Step 11**

`address-family l2vpn evpn`

**Example:**

```
RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family l2vpn evpn
```

Enables EVPN address family globally under BGP routing process and enters EVPN address family configuration submode.

**Step 12**

Use the **commit** or **end** command.

- **commit** - Saves the configuration changes and remains within the configuration session.
- **end** - Prompts user to take one of these actions:
  - **Yes** - Saves configuration changes and exits the configuration session.
  - **No** - Exits the configuration session without committing the configuration changes.
  - **Cancel** - Remains in the configuration mode, without committing the configuration changes.

---

**Configure EVI and Corresponding BGP Route Targets 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.

**SUMMARY STEPS**

1. **configure**
2. **evpn**
3. **evi evi_id**
4. **bgp**
5. **table-policy policy name**
Configure EVI and Corresponding BGP Route Targets under EVPN Configuration Mode

6. **route-target import** \{ 2-byte as_number | 4-byte as_number | IP_address | none \} : \{ nn \}

7. **route-target export** \{ 2-byte as_number | 4-byte as_number | IP_address | none \} : \{ nn \}

8. **exit**

9. **advertise-mac**

10. Use the **commit** or **end** command.

**DETAILED STEPS**

**Step 1**

configure

**Example:**

RP/0/RSP0/CPU0:router# configure

Enters the global configuration mode.

**Step 2**

**evpn**

**Example:**

RP/0/RSP0/CPU0:router(config)# evpn

Enters EVPN configuration mode.

**Step 3**

**evi evi_id**

**Example:**

RP/0/RSP0/CPU0:router(config-evpn)# evi 1

Configures Ethernet VPN ID.

The EVI ID range is from 1 to 65534.

**Step 4**

**bgp**

**Example:**

RP/0/RSP0/CPU0:router(config-evpn-evi)# bgp

Enters the BGP configuration mode for the specific EVI.

**Step 5**

**table-policy policy name**

**Example:**

RP/0/RSP0/CPU0:router(config-evpn-evi-bgp)# table-policy spp-basic-6

Configures policy for installation of forwarding data to L2FIB.

The EVI ID range is from 1 to 65534.

**Step 6**

**route-target import** \{ 2-byte as_number | 4-byte as_number | IP_address | none \} : \{ nn \}

**Example:**

RP/0/RSP0/CPU0:router(config-evpn-evi-bgp)# route-target import 100:6005

Configures importing of routes from the L2 EVPN BGP NLRI that have the matching route-target value.

**Step 7**

**route-target export** \{ 2-byte as_number | 4-byte as_number | IP_address | none \} : \{ nn \}

**Example:**

RP/0/RSP0/CPU0:router(config-evpn-evi-bgp)# route-target export 100:6005
**EVPN Features**

Example: EVI Configuration under EVPN Configuration-mode

Every participating EVPN instances are identified by EVI_ID. EVI_ID must be defined under EVPN configuration mode as shown below.

```
EVPN
  Evi <VPN ID>
    Bgp
      RD <>
      RT <>
  !
  advertise-mac
```

**Configure EVI under a Bridge Domain**

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

**SUMMARY STEPS**

1. `configure`
2. `l2vpn`
3. `bridge group bridge group name`
4. `bridge-domain bridge-domain name`
5. `interface type interface-path-id`
6. `exit`
Configure EVI under a Bridge Domain

**DETAILED STEPS**

**Step 1**

configure

*Example:*

```
RP/0/RSP0/CPU0:router# configure
```

Enters the global configuration mode.

**Step 2**

l2vpn

*Example:*

```
RP/0/RSP0/CPU0:router(config)# l2vpn
```

Enters the L2VPN configuration mode.

**Step 3**

bridge group *bridge group name*

*Example:*

```
RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group bg1
```

Creates a bridge group so that it can contain bridge domains and then assigns network interfaces to the bridge domain.

**Step 4**

bridge-domain *bridge-domain name*

*Example:*

```
RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain bd1
```

Establishes a bridge domain and enters L2VPN bridge group bridge domain configuration mode.

**Step 5**

interface *type interface-path-id*

*Example:*

```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# interface GigabitEthernet0/2/0/0.1
```

Enters interface configuration mode and adds an interface to a bridge domain that allows packets to be forwarded and received from other interfaces that are part of the same bridge domain.

**Step 6**

exit

*Example:*

```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)# exit
```

Exits the current configuration mode.
Step 7  
\textbf{vfi }\{ \textit{vfi name} \}

\textbf{Example:}

```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd)# vfi v1
```

Configures virtual forwarding interface (VFI) parameters and enters L2VPN bridge group bridge domain VFI configuration mode.

Step 8  
\textbf{neighbor }\{ \textit{A.B.C.D} \} \{ \textit{pw-id value} \}

\textbf{Example:}

```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi)# neighbor 10.1.1.2 pw-id 1000
```

Adds an access pseudowire port to a bridge domain or a pseudowire to a bridge virtual forwarding interface (VFI).

- Use the \textit{A.B.C.D} argument to specify the IP address of the cross-connect peer.
- Use the \textit{pw-id} keyword to configure the pseudowire ID and ID value. The range is 1 to 4294967295.

Step 9  
\textbf{mpls static label local }\textit{label} \textbf{remote }\textit{label}

\textbf{Example:}

```
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-vfi-pw)# mpls static label local 20001 remote 10001
```

Configures the MPLS static local label to associate a remote label with a pseudowire or any other bridge interface.

Step 10  
Use the \textbf{commit} or \textbf{end} command.

- \textbf{commit} - Saves the configuration changes and remains within the configuration session.
- \textbf{end} - Prompts user to take one of these actions:
  - \textbf{Yes} - Saves configuration changes and exits the configuration session.
  - \textbf{No} - Exits the configuration session without committing the configuration changes.
  - \textbf{Cancel} - Remains in the configuration mode, without committing the configuration changes.

---

\textbf{EVI Configuration under L2VPN Bridge-Domain}

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

\textbf{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
```
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 <VPN-ID>

AutoDiscovery BGP and LDP signaling based VPLS

l2vpn
bridge group bg1
bridge-domain bd-1-3
  interface GigabitEthernet0/2/0/0.3
  vfi vfi-1-3
    vpn-id 3
    autodiscovery bgp
    rd 101:3
    route-target 65530:300
    signaling-protocol ldp
    vpls-id 65530:3
  
  evi <VPN-ID>

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 <VPN-ID>
Verify EVPN Configuration

Verify EVPN configuration and MAC advertisement.

Verify EVPN status, AC status, and VFI status

```
RP/0/# 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.

```
RP/0/# 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.

```
RP/0/# show bgp rt l2vpn evpn
Mon Feb 20 21:06:18.882 EST
EVTCOMM IMP/EXP
```
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>
```

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

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>
```

<table>
<thead>
<tr>
<th>Mac Address Type</th>
<th>Learned from/Filtered on</th>
<th>LC learned Resync Age/Last Change Mapped</th>
</tr>
</thead>
<tbody>
<tr>
<td>0033.0000.0001</td>
<td>EVPN BD id: 0</td>
<td>N/A N/A N/A</td>
</tr>
<tr>
<td>0033.0000.0002</td>
<td>EVPN BD id: 1</td>
<td>N/A N/A N/A</td>
</tr>
<tr>
<td>0033.0000.0003</td>
<td>EVPN BD id: 2</td>
<td>N/A N/A N/A</td>
</tr>
<tr>
<td>0033.0000.0004</td>
<td>EVPN BD id: 3</td>
<td>N/A N/A N/A</td>
</tr>
</tbody>
</table>

Verify EVPN MAC routes pertaining to specific VPN instance.

```
RP/0/# show evpn evi vpn-id 1 mac
Mon Feb 20 21:36:23.574 EST
EVI MAC address IP address Nexthop
---------- -------------- ------------------------
1 0033.0000.0001 :: 200.0.1.1/45106
```

Verify L2 routing.

```
RP/0/# 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 MAC routes pertaining to specific VPN instance.

```
RP/0/# show evpn evi vpn-id 1 mac
Mon Feb 20 21:36:23.574 EST
EVI MAC address IP address Nexthop
---------- -------------- ------------------------
1 0033.0000.0001 :: 200.0.1.1/45106
```

Verify L2 routing.

```
RP/0/# 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 MAC routes pertaining to specific VPN instance.

```
RP/0/# show evpn evi vpn-id 1 mac
Mon Feb 20 21:36:23.574 EST
EVI MAC address IP address Nexthop
---------- -------------- ------------------------
1 0033.0000.0001 :: 200.0.1.1/45106
```

Verify L2 routing.

```
RP/0/# 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 MAC routes pertaining to specific VPN instance.

```
RP/0/# show evpn evi vpn-id 1 mac
Mon Feb 20 21:36:23.574 EST
EVI MAC address IP address Nexthop
---------- -------------- ------------------------
1 0033.0000.0001 :: 200.0.1.1/45106
```

Verify L2 routing.

```
RP/0/# 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 MAC routes pertaining to specific VPN instance.

```
RP/0/# show evpn evi vpn-id 1 mac
Mon Feb 20 21:36:23.574 EST
EVI MAC address IP address Nexthop
---------- -------------- ------------------------
1 0033.0000.0001 :: 200.0.1.1/45106
```

Verify L2 routing.

```
RP/0/# 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 MAC routes pertaining to specific VPN instance.

```
RP/0/# show evpn evi vpn-id 1 mac
Mon Feb 20 21:36:23.574 EST
EVI MAC address IP address Nexthop
---------- -------------- ------------------------
1 0033.0000.0001 :: 200.0.1.1/45106
```

Verify L2 routing.
Verify EVPN route-type 2 routes.

RP/0#/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
      100 0 i
Route Distinguisher: 200.0.1.1:2
   *>i[2][0][48][0033.0000.0002][0]/104
      100 0 i
Route Distinguisher: 200.0.1.1:3
   *>i[2][0][48][0033.0000.0003][0]/104
      100 0 i
Route Distinguisher: 200.0.1.1:4
   *>i[2][0][48][0033.0000.0004][0]/104
      100 0 i
Route Distinguisher: 200.0.1.1:1 (default for vrf bd-1-1)
   *>i[2][0][48][0033.0000.0001][0]/104
      100 0 i
Route Distinguisher: 200.0.1.1:2 (default for vrf bd-1-2)
   *>i[2][0][48][0033.0000.0002][0]/104
      100 0 i
Route Distinguisher: 200.0.1.1:3 (default for vrf bd-1-3)
   *>i[2][0][48][0033.0000.0003][0]/104
      100 0 i
Route Distinguisher: 200.0.1.1:4 (default for vrf bd-1-4)
   *>i[2][0][48][0033.0000.0004][0]/104
      100 0 i

Processed 8 prefixes, 8 paths

Verify inclusive multicast routes and route-type 3 routes.

RP/0#/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
EVPN Single-Active Multi-Homing

The EVPN Single-Active Multi-Homing feature supports single-active redundancy mode. In single-active mode, the PE nodes locally connected to an Ethernet Segment load balance traffic to and from the Ethernet Segment based on EVPN service instance (EVI). Within an EVPN service instance, only one PE forwards traffic to and from the Ethernet Segment.

**Figure 10: EVPN: Single-Active Multi-Homing**

Here is a topology in which CE1 is multihomed to PE1 and PE2. PE1 and PE2 are connected to PE3 through MPLS core. CE3 is connected to PE3 through an Ethernet interface bundle. PE1 and PE2 advertise Type 4 routes, and then do designated forwarder (DF) election. The non-DF blocks the traffic in both the directions in single-active mode.

Consider a traffic flow from CE1 to CE2. CE1 sends an address resolution protocol (ARP) broadcast request to both PE1 and PE2. If PE1 is the designated forwarder for the EVI, PE1 forwards the ARP request from CE1. PE2 drops the traffic from CE1. Thereafter, all the unicast traffic is sent through PE1. PE2 will be
stand-by or blocked. Traffic is not sent over this path. PE1 advertises MAC to PE3. PE3 always sends and receives traffic through PE1. PE3 sends the traffic to CE2 over Ethernet interface bundle.

**Configure EVPN Single-Active Multi-Homing**

Perform the following tasks on PE1 and PE2 to configure EVPN Single-Active Multi-Homing feature:

**Configuring EVPN Ethernet Segment**

Perform this task to configure the EVPN Ethernet segment.

**SUMMARY STEPS**

1. configure
2. evpn
3. (Optional) timers
4. (Optional) peering seconds
5. (Optional) recovery seconds
6. exit
7. interface Bundle-Ether bundle-id
8. ethernet-segment
9. identifier type esi-type esi-identifier
10. load-balancing-mode single-active
11. bgp route-target ipv4/v6-address
12. (Optional) service-carving manual primary {isid} secondary {isid}
13. exit
14. exit
15. (Optional) mac-flush mvrp
16. (Optional) timers
17. (Optional) peering seconds
18. (Optional) recovery seconds
19. Use the commit or end command.

**DETAILED STEPS**

**Step 1**

configure

Example:

```
RP/0/RSP0/CPU0:router# configure
```

Enters the Global Configuration mode.

**Step 2**

evpn

Example:
RP/0/RSP0/CPU0:router(config)# evpn

Enters EVPN configuration mode.

**Step 3**

(Optional) **timers**

*Example:*

RP/0/RSP0/CPU0:router(config-evpn)# timers

Configures global EVPN timers.

**Step 4**

(Optional) **peering seconds**

*Example:*

RP/0/RSP0/CPU0:router(config-evpn-timers)# peering 15

Configures the global peering timer. Default is 3 seconds. Range is 0 to 300 seconds.

**Step 5**

(Optional) **recovery seconds**

*Example:*

RP/0/RSP0/CPU0:router(config-evpn-timers)# recovery 30

Configures the global recovery timer. Default is 30 seconds. Range is 20 to 3600 seconds.

**Step 6**

**exit**

*Example:*

RP/0/RSP0/CPU0:router(config-evpn-timers)# exit

Exits the current configuration mode.

**Step 7**

**interface Bundle-Ether bundle-id**

*Example:*

RP/0/RSP0/CPU0:router(config-evpn)# interface Bundle-Ether1

Enters bundle interface configuration mode.

**Step 8**

**ethernet-segment**

*Example:*

RP/0/RSP0/CPU0:router(config-evpn-ac)# ethernet-segment

Enters the EVPN ethernet-segment configuration mode.

**Step 9**

**identifier type esi-type esi-identifier**
Example:

RP/0/RSP0/CPU0:router(config-evpn-ac-es)# identifier type 0 40.00.00.00.00.00.00.00.01

Configures the Ethernet segment identifier (ESI) of an interface.

**Step 10**

**load-balancing-mode single-active**

Example:

RP/0/RSP0/CPU0:router(config-evpn-ac-es)# load-balancing-mode single-active

Specifies the load balancing mode.

**Step 11**

**bgp route-target ipv4/v6-address**

Example:

RP/0/RSP0/CPU0:router(config-evpn-ac-es)# bgp route-target 4000.0000.0001

Configures the BGP Import Route-Target for the Ethernet-Segment.

**Step 12**

(Optional) **service-carving manual primary {isid} secondary {isid}**

Example:

RP/0/RSP0/CPU0:router(config-evpn-ac-es)# service-carving manual primary 100 secondary 200

Specifies a list of service identifiers (isid) as active and standby services. The isid range is from 256 to 16777216.

**Step 13**

**exit**

Example:

RP/0/RSP0/CPU0:router(config-evpn-ac-es-man)# exit

Exits the current configuration mode.

**Step 14**

**exit**

Example:

RP/0/RSP0/CPU0:router(config-evpn-ac-es)# exit

Exits the current configuration mode.

**Step 15**

(Optional) **mac-flush mvrp**

Example:

RP/0/RSP0/CPU0:router(config-evpn-ac)# mac-flush mvrp

Specifies MAC flush mode for this Ethernet Segment.
Configure EVPN Service Instance (EVI) Parameters

Perform this task to define EVPN service instance (EVI) parameters.

**SUMMARY STEPS**

1. configure
2. evpn
3. evi evi_id
4. bgp
5. (Optional) rd { 2-byte as_number | 4-byte as_number | IP_address | none } : { nn }
6. (Optional) route-target import { 2-byte as_number | 4-byte as_number | IP_address | none } : { nn }
7. (Optional) route-target export { 2-byte as_number | 4-byte as_number | IP_address | none } : { nn }
8. exit
9. advertise-mac

---

**Step 16**
(Optional) timers

*Example:*

```
RP/0/RSP0/CPU0:router(config-evpn-ac)# timers
```

Configures per Ethernet segment timers.

**Step 17**
(Optional) peering seconds

*Example:*

```
RP/0/RSP0/CPU0:router(config-evpn-ac-timers)# peering 15
```

Configures the interface specific peering timer. Default is 3 seconds. Range is 0 to 300 seconds.

**Step 18**
(Optional) recovery seconds

*Example:*

```
RP/0/RSP0/CPU0:router(config-evpn-ac-timers)# recovery 30
```

Configures the interface specific recovery timer. Default is 30 seconds. Range is 20 to 3600 seconds.

**Step 19**
Use the `commit` or `end` command.

- **commit** - Saves the configuration changes and remains within the configuration session.
- **end** - Prompts user to take one of these actions:
  - **Yes** - Saves configuration changes and exits the configuration session.
  - **No** - Exits the configuration session without committing the configuration changes.
  - **Cancel** - Remains in the configuration mode, without committing the configuration changes.
10. Use the **commit** or **end** command.

**DETAILED STEPS**

**Step 1** configure

Example:

```
RP/0/RSP0/CPU0:router# configure
```

Enters the global configuration mode.

**Step 2** evpn

Example:

```
RP/0/RSP0/CPU0:router(config)# evpn
```

Enters EVPN configuration mode.

**Step 3** evi **evi_id**

Example:

```
RP/0/RSP0/CPU0:router(config-evpn)# evi 6005
```

Configures Ethernet VPN ID.

The EVI ID range is from 1 to 65534.

**Step 4** bgp

Example:

```
RP/0/RSP0/CPU0:router(config-evpn-evi)# bgp
```

Enters the BGP configuration mode for the specific EVI.

**Step 5** (Optional) **rd** \{ **2-byte as_number** | **4-byte as_number** | **IP_address** | **none** \} : \{ **nn** \}

Example:

```
RP/0/RSP0/CPU0:router(config-evpn-evi-bgp)# rd 200:50
```

Configures the route distinguisher.

**Step 6** (Optional) **route-target import** \{ **2-byte as_number** | **4-byte as_number** | **IP_address** | **none** \} : \{ **nn** \}

Example:

```
RP/0/RSP0/CPU0:router(config-evpn-evi-bgp)# route-target import 100:6005
```

Configures importing of routes from the L2 EVPN BGP NLRI that have the matching route-target value.

**Step 7** (Optional) **route-target export** \{ **2-byte as_number** | **4-byte as_number** | **IP_address** | **none** \} : \{ **nn** \}

Example:

```
RP/0/RSP0/CPU0:router(config-evpn-evi-bgp)# route-target export 100:6005
```

Configures exporting of routes to the L2 EVPN BGP NLRIs and assigns the specified route-target identifiers to the BGP EVPN NLRIs.

**Step 8** exit
Configure Layer 2 Interface

Perform this task to define Layer 2 interface.

SUMMARY STEPS

1. configure
2. interface bundle-ether instance.subinterface l2transport
3. (Optional) no shut
4. encapsulation dot1q vlan-id
5. (Optional) rewrite tag pop dot1q vlan-id symmetric
6. Use the commit or end command.

DETAILED STEPS

Step 1 configure

Example:
RP/0/RSP0/CPU0:router# configure

Enters the global configuration mode.

Step 2 interface bundle-ether instance.subinterface l2transport

Example:
RP/0/RSP0/CPU0:router(config)# interface bundle-ether2.1 l2transport

Configures the bundle ethernet interface and enables Layer 2 transport mode on the bundle ethernet interface.

Example:

Step 9 advertise-mac

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

Advertises the MAC route.

Step 10 Use the commit or end command.

commit - Saves the configuration changes and remains within the configuration session.
end - Prompts user to take one of these actions:
• Yes - Saves configuration changes and exits the configuration session.
• No - Exits the configuration session without committing the configuration changes.
• Cancel - Remains in the configuration mode, without committing the configuration changes.
Configure a Bridge Domain

Perform the following steps to configure the bridge domain on PE1 and PE2.

SUMMARY STEPS

1. configure
2. l2vpn
3. bridge group bridge-group-name
4. bridge-domain bridge-domain-name
5. interface Bundle-Ether bundle-id
6. evi ethernet vpn id
7. Use the commit or end command.

Step 3  (Optional) no shut

**Example:**

RP/0/RSP0/CPU0:router(config-subif-l2)# no shut

If a link is in the down state, bring it up. The no shut command returns the link to an up or down state depending on the configuration and state of the link.

Step 4  encapsulation dot1q vlan-id

**Example:**

RP/0/RSP0/CPU0:router(config-subif-l2)# encapsulation dot1q 1

Assigns a VLAN attachment circuit to the subinterface.

Step 5  (Optional) rewrite tag pop dot1q vlan-id symmetric

**Example:**

RP/0/RSP0/CPU0:router(config-subif-l2)# rewrite ingress tag pop 1 symmetric

Specifies the encapsulation adjustment that is to be performed on the frame ingress to the service instance.

Step 6  Use the commit or end command.

**commit** - Saves the configuration changes and remains within the configuration session.

**end** - Prompts user to take one of these actions:

- **Yes** - Saves configuration changes and exits the configuration session.
- **No** - Exits the configuration session without committing the configuration changes.
- **Cancel** - Remains in the configuration mode, without committing the configuration changes.
DETAILED STEPS

**Step 1**  
**configure**  
**Example:**  
RP/0/RSP0/CPU0:router# configure  
Enters the global configuration mode.

**Step 2**  
**l2vpn**  
**Example:**  
RP/0/RSP0/CPU0:router(config)# l2vpn  
Enters the l2vpn configuration mode.

**Step 3**  
**bridge group bridge-group-name**  
**Example:**  
RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group 6005  
Enters the bridge group configuration mode.

**Step 4**  
**bridge-domain bridge-domain-name**  
**Example:**  
RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain 6005  
Enters the bridge domain configuration mode.

**Step 5**  
**interface Bundle-Ether bundle-id**  
**Example:**  
RP/0/RSP0/CPU0:router(config-evpn)# interface Bundle-Ether2.1  
Enters bundle interface configuration mode.

**Step 6**  
**evi ethernet vpn id**  
**Example:**  
RP/0/RSP0/CPU0:router(config-l2vpn-bg-bd-ac)# evi 6005  
Creates the ethernet VPN ID.

**Step 7**  
Use the **commit** or **end** command.  
**commit** - Saves the configuration changes and remains within the configuration session.  
**end** - Prompts user to take one of these actions:  
- **Yes** - Saves configuration changes and exits the configuration session.  
- **No** - Exits the configuration session without committing the configuration changes.  
- **Cancel** - Remains in the configuration mode, without committing the configuration changes.
Virtual Ethernet Segment (vES)

Traditionally, multi-homing access to EVPN bridge is through bundle Ethernet connection or a physical Ethernet connection. The Virtual Ethernet Segment (vES) allows a Customer Edge (CE) to access EVPN bridge through MPLS network. The logical connection between CE and EVPN provider edge (PE) is a pseudowire (PW). Using vES you can connect VxLAN EVPN-based data center and a legacy data center through PW based virtual circuit.

The VxLAN EVPN-based data centers and legacy data centers are interconnected through access pseudowire (PW), access virtual forwarding instance (VFI), or both. One vES is created for each access PW and one vES is created per access VFI. This feature supports only single-active mode.

Use access VFI for connecting multiple sites in a mesh topology. Use access PW for connecting few sites in hub and spoke topology.

Consider the topology where EVPN data centers are connected to legacy data centers through access PW or access VFI on a single Ethernet segment, which is vES.

Consider a traffic flow from CE2 to PE3. CE2 sends the traffic to DCI1 or DCI2 through EVPN VxLAN. DCI1 and DCI2 are connected to PE3 through access PW on a single Ethernet segment. DCI1 and DCI2 advertise Type 4 routes, and then do designated forwarder (DF) election. The non-DF blocks the traffic on that particular Ethernet segment. Both DCI1 and DCI2 can do the DF election. DCI1 and DCI2 perform DF election after they discover each other. Either one of them can be a DF and other a non-DF. The traffic is forwarded through the DF. The non-DF path is in stand-by mode. DF election is used to prevent traffic loop. DCI1 or DCI2 sends the traffic to PE3.

Consider a traffic flow from CE2 to PE1 and PE2. CE2 sends the traffic to DCI1 or DCI2 through EVPN VxLAN. DCI1 and DCI2 are connected to PE1 and PE2 through access VFI. DCI1 and DCI2 are connected to PE1 and PE2 through access VFI on a single Ethernet segment. DCI1 or DCI2 sends the traffic to PE1 and PE2. DCI1 and DCI2 advertise Type 4 routes, and then do designated forwarder (DF) election. The non-DF
blocks the traffic on that particular Ethernet segment. Both DCI1 and DCI2 can do the DF election. DCI1 and DCI2 perform DF election after they discover each other. Either one of them can be a DF and other a non-DF. The traffic is forwarded through the DF. The non-DF path is in stand-by mode. DF election is used to prevent traffic loop. DCI1 or DCI2 sends the traffic to PE3.

Interoperability Between VxLAN and vES

When all-active VxLAN and single-active vES are integrated together, some traffic may take non-optimal path. Consider a traffic flow from CE2 to PE1. VxLAN is in all-active mode and vES is in single active mode. CE2 sends the traffic to ToR1, and ToR1 sends the traffic to DCI1 and DCI2. Both DCI1 and DCI2 can receive the traffic from VxLAN because it is in all-active mode. But, either DCI1 or DCI2 (which is a DF) can forward the traffic through vES. If DCI1 is a non-DF, the traffic is sent from DCI2 to PE1.

Limitations

The vES feature is supported with the following limitations:

- Core isolation is not supported for vES. MPLS core network must be always up and vES redundant peers must be able to exchange type 4 routes while vES is in operation.
- Only targeted LDP pseudowire is supported.
- Interoperability between VxLAN and classic VFI (legacy L2VPN) is not supported.
- Backup PW is not supported with vES.
- PW-status must be supported and enabled on both sides of PW.
- Up to 400 unique RTs are supported for each ESI. However, multiple ESI can share same the RT. Hence, this does not restrict the number of vES.

Configure Virtual Ethernet Segment (vES)

The following sections describe how to configure access PW and access VFI.

Configure Access PW

This section describes how you can configure access PW.

/* Configure DCI1 */
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# l2vpn
RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group bg1
RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain bd1
RP/0/RSP0/CPU0:router(config-bd)# neighbor 70.70.70.70 pw-id 17300001
RP/0/RSP0/CPU0:router(config-bd-pw)# evi 1
RP/0/RSP0/CPU0:router(config-bd-pw-evi)# member vni 10001

/* Configure EVPN */
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# evpn
RP/0/RSP0/CPU0:router(config-evpn)# virtual neighbor 70.70.70.70 pw-id 17300001
RP/0/RSP0/CPU0:router(config-evpn-ac-pw)# ethernet-segment
RP/0/RSP0/CPU0:router(config-evpn-ac-pw-es)# identifier type 0 12.12.00.00.01.00.00.03
RP/0/RSP0/CPU0:router(config-evpn-ac-pw-es)# bgp route-target 1212.8888.0003
RP/0/RSP0/CPU0:router(config-evpn-ac-pw-es)# exit
RP/0/RSP0/CPU0:router(config-evpn-ac-pw)# timers peering 15
RP/0/RSP0/CPU0:router(config-evpn-ac-pw-timers)# commit

/* Configure DCI2 */
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# l2vpn
RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group bg1
RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain bd1
RP/0/RSP0/CPU0:router(config-bg-bd)# neighbor 70.70.70.70 pw-id 27300001
RP/0/RSP0/CPU0:router(config-bg-bd-pw)# evi 1
RP/0/RSP0/CPU0:router(config-bg-bd-pw-evi)# member vni 10001

/* Configure EVPN */
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# evpn
RP/0/RSP0/CPU0:router(config-evpn)# virtual neighbor 70.70.70.70 pw-id 27300001
RP/0/RSP0/CPU0:router(config-evpn-ac-pw)# ethernet-segment
RP/0/RSP0/CPU0:router(config-evpn-ac-pw-es)# identifier type 0 12.12.00.00.00.01.00.00.03
RP/0/RSP0/CPU0:router(config-evpn-ac-pw-es)# bgp route-target 1212.8888.0003
RP/0/RSP0/CPU0:router(config-evpn-ac-pw-es)# exit
RP/0/RSP0/CPU0:router(config-evpn-ac-pw)# timers peering 15
RP/0/RSP0/CPU0:router(config-evpn-ac-pw-timers)# commit

/* Configure PE3 */
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# l2vpn
RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group 73
RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain 73-1
RP/0/RSP0/CPU0:router(config-bg-bd)# neighbor 10.10.10.10 pw-id 17300001
RP/0/RSP0/CPU0:router(config-bg-bd-pw)# neighbor 20.20.20.20 pw-id 27300001
RP/0/RSP0/CPU0:router(config-bg-bd-pw)# commit

Running Configuration - Access PW

This section shows access PW running configuration.

/* On DCI1 */
!
configure
l2vpn
bridge group bg1
bridge-domain bd1
neighbor 70.70.70.70 pw-id 17300001
evi 1
member vni 10001
!
evpn
virtual neighbor 70.70.70.70 pw-id 17300001
ethernet-segment
identifier type 0 12.12.00.00.00.01.00.00.03
bgp route-target 1212.8888.0003
!
timers peering 15
!
/* On DCI2 */
!
configure
l2vpn
bridge group bg1
bridge-domain bd1
neighbor 70.70.70.70 pw-id 27300001
evi 1
Configure Access VFI

This section describes how you can configure access VFI. RTs must match on the redundant DCIs that are connected to the same Ethernet segment.

/* Configure DCI1 */
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# l2vpn
RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group bg1
RP/0/RSP0/CPU0:router(config-l2vpn)# bridge-domain bd1
RP/0/RSP0/CPU0:router(config-l2vpn)# access-vfi ac-vfi-1
RP/0/RSP0/CPU0:router(config-l2vpn)# neighbor 70.70.70.70 pw-id 17300001
RP/0/RSP0/CPU0:router(config-l2vpn)# neighbor 80.80.80.80 pw-id 18100005
RP/0/RSP0/CPU0:router(config-l2vpn)# exit
RP/0/RSP0/CPU0:router(config)# evpn
RP/0/RSP0/CPU0:router(config-evpn)# virtual vfi ac-vfi-1
RP/0/RSP0/CPU0:router(config-evpn)# ethernet-segment
RP/0/RSP0/CPU0:router(config-evpn-ac-vfi)# identifier type 0 12.12.00.00.00.01.00.00.01
RP/0/RSP0/CPU0:router(config-evpn-ac-vfi)# bgp route-target 1212.0005.0001
RP/0/RSP0/CPU0:router(config-evpn-ac-vfi)# exit
RP/0/RSP0/CPU0:router(config-evpn-ac-vfi)# timers peering 15
RP/0/RSP0/CPU0:router(config-evpn-ac-vfi)# exit
RP/0/RSP0/CPU0:router(config-evpn-ac-vfi)# ethernet-segment
RP/0/RSP0/CPU0:router(config-evpn-ac-vfi)# identifier type 0 12.12.00.00.05.00.00.00.03
RP/0/RSP0/CPU0:router(config-evpn-ac-vfi)# bgp route-target 1212.0005.0003
RP/0/RSP0/CPU0:router(config-evpn-ac-vfi)# commit

/* Configure DCI2 */
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# l2vpn
RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group bg1
RP/0/RSP0/CPU0:router(config-l2vpn)# bridge-domain bd1
RP/0/RSP0/CPU0:router(config-l2vpn)# access-vfi ac-vfi-1
RP/0/RSP0/CPU0:router(config-l2vpn)# neighbor 70.70.70.70 pw-id 27300001

Configure Access VFI
/* Configure EVVPN */
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# evpn
RP/0/RSP0/CPU0:router(config-evpn)# virtual vfi ac-vfi-1
RP/0/RSP0/CPU0:router(config-evpn-ac-vfi-es)# identifier type 0 12.12.00.00.00.01.00.00.01
RP/0/RSP0/CPU0:router(config-evpn-ac-vfi-es)# bgp route-target 1212.0005.0001
RP/0/RSP0/CPU0:router(config-evpn-ac-vfi)# timers peering 15
RP/0/RSP0/CPU0:router(config-evpn-ac-vfi-es)# identifier type 0 12.12.00.00.00.05.00.00.03
RP/0/RSP0/CPU0:router(config-evpn-ac-vfi-es)# bgp route-target 1212.0005.0003
RP/0/RSP0/CPU0:router(config-evpn-ac-vfi-es)# commit

/* Configure PE1 */
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# l2vpn
RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group 71
RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain bd1
RP/0/RSP0/CPU0:router(config-l2vpn-bd)# vfi vfi-71-1
RP/0/RSP0/CPU0:router(config-l2vpn-bd-vfi)# neighbor 10.10.10.10 pw-id 17100005
RP/0/RSP0/CPU0:router(config-l2vpn-bd-vfi-pw)# neighbor 20.20.20.20 pw-id 27100005
RP/0/RSP0/CPU0:router(config-l2vpn-bd-vfi-pw)# neighbor 80.80.80.80 pw-id 78100005
RP/0/RSP0/CPU0:router(config-l2vpn-bd-vfi-pw)# commit

/* Configure PE2 */
RP/0/RSP0/CPU0:router# configure
RP/0/RSP0/CPU0:router(config)# l2vpn
RP/0/RSP0/CPU0:router(config-l2vpn)# bridge group 71
RP/0/RSP0/CPU0:router(config-l2vpn-bg)# bridge-domain bd1
RP/0/RSP0/CPU0:router(config-l2vpn-bd)# vfi vfi-71-1
RP/0/RSP0/CPU0:router(config-l2vpn-bd-vfi)# neighbor 10.10.10.10 pw-id 18100005
RP/0/RSP0/CPU0:router(config-l2vpn-bd-vfi-pw)# neighbor 20.20.20.20 pw-id 28100005
RP/0/RSP0/CPU0:router(config-l2vpn-bd-vfi-pw)# neighbor 70.70.70.70 pw-id 78100005
RP/0/RSP0/CPU0:router(config-l2vpn-bd-vfi-pw)# commit

Running Configuration - Access VFI

This section shows access VFI running configuration.

/* On DC1 */

configure
l2vpn
bridge group bg1
bridge-domain bd1
access-vfi ac-vfi-1
neighbor 70.70.70.70 pw-id 17100005
evi 1
member vni 10001

evpn
virtual vfi ac-vfi-1
ethernet-segment
identifier type 0 12.12.00.00.00.01.00.00.01
bgp route-target 1212.0005.0001
!
timers peering 15
!
!
ethernet-segment
   identifier type 0 12.12.00.00.05.00.00.00.03
   bgp route-target 1212.0005.0003
!

/* On DCI2 */
!
configure
l2vpn
   bridge group bg1
   bridge-domain bd1
      access-vfi ac-vfi-1
      neighbor 70.70.70.70 pw-id 27100005
      neighbor 80.80.80.80 pw-id 28100005
      evi 1
      member vni 10001
!
evpn
   virtual vfi ac-vfi-1
   ethernet-segment
      identifier type 0 12.12.00.00.00.01.00.00.01
      bgp route-target 1212.0005.0001
!
      timers peering 15
!
!
ethernet-segment
   identifier type 0 12.12.00.00.05.00.00.00.03
   bgp route-target 1212.0005.0003
!

/* On PE1 */
!
configure
l2vpn
   bridge group bg71
   bridge-domain bd71-1
      neighbor 10.10.10.10 pw-id 17100005
!
      neighbor 20.20.20.20 pw-id 27100005
!
      neighbor 80.80.80.80 pw-id 78100005
!

/* On PE2 */
!
configure
l2vpn
   bridge group bg71
   bridge-domain bd71-1
EVPN Anycast Gateway All-Active Static Pseudowire

The EVPN Anycast Gateway All-active Static Pseudowire (PW) feature enables all-active multi-homing support for static PWs. When static PWs are configured, it overrides the default behavior of single-active, and the node becomes all-active per flow (AApF).

Configure EVPN Anycast All-active Static Pseudowire

Consider a traffic flow from CE1 to CE2. CE1 sends the traffic to PE1 or PE2. PE1 and PE2 are connected to CE1 through static PW. CE1 sends the traffic to the PEs using the same anycast IP address, and uses IGP ECMP for load balancing. Anycast PWs are static. You can configure an ESI per static PW. PE1 and PE2 forward the traffic based on the type of traffic.

Consider PE1 to be a DF and PE2 a non-DF. When a Broadcast, Unknown unicast and Multicast (BUM) traffic is sent from CE1 to PE1 or PE2. PE1 sends traffic to all other nodes towards the core side, including PE2. However, PE2 drops the traffic as it is a non-DF. Similarly, PE2 sends traffic to all other nodes towards the core side, including PE1. However, PE1 drops the traffic as it is coming from a non-DF node. PE1 or PE2 sends the traffic to CE2 through MPLS core.

When BUM traffic is sent from the core side, that is from PE3 or PE4 to CE1. PE3 or PE4 sends the traffic to PE1 and PE2. PE1 forwards the traffic to CE1. PE2 drops the packets as it is a non-DF.

When unicast traffic is sent from CE1 to PE1 and PE2, both PE1 and PE2 forward the traffic to the core. When unicast traffic is sent from PE3 or PE4 to CE1, both PE1 and PE2 send the traffic to CE1.

Configure Static PW

This section describes how you can configure static PW.

/* Configure PE1 */
RP/0/RSP0/CP00:router# configure
RP/0/RSP0/CP00:router(config)# l2vpn
RP/0/RSP0/CP00:router(config-l2vpn)# bridge group bg1
RP/0/RSP0/CP00:router(config-l2vpn-bg)# bridge-domain bd1
RP/0/RSP0/CP00:router(config-bg-bd)# neighbor 10.0.0.1 pw-id 17300001
RP/0/RSP0/CP00:router(config-bg-bd-pw)# mpls static label local 1000 remote 2000

/* Configure EVPN */
Running Configuration

This section shows static PW running configuration.

/* On PE1 */
!
configure
l2vpn
bridge group bg1
bridge-domain bd1
    neighbor 10.0.0.1 pw-id 17300001
    mpls static label local 1000 remote 2000
!
evpn
    virtual neighbor 10.0.0.1 pw-id 17300001
    ethernet-segment
        identifier type 0 14.14.00.00.01.00.00.03
!
/* On PE2 */
!
configure
l2vpn
bridge group bg1
bridge-domain bd1
    neighbor 10.0.0.1 pw-id 17300001
    mpls static label local 1000 remote 2000
!
EVN Features
evpn
  virtual neighbor 10.0.0.1 pw-id 17300001
  ethernet-segment
    identifier type 0 14.14.00.00.01.00.00.03
!

/* On CE1 */
!
configure
l2vpn
  bridge group bg73
  bridge-domain bd73-1
    neighbor 209.165.200.225 pw-id 17300001
    mpls static label local 2000 remote 1000
!

**Verification**

The outputs in this section show the number of static PWs configured on CE1, PE1, and PE2 and the configuration details of their neighbors.

/* CE1 static PW configuration details */

RP/0/RSP0/CPU0# show l2vpn bridge-domain bd-name bd-73-1
Fri Aug 11 12:36:12.732 EDT
Legend: pp = Partially Programmed.
  Bridge group: bg73, bridge-domain: bd-73-1, id: 3, 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: 0, PWs: 1 (1 up), PBBs: 0 (0 up), VNIs: 0 (0 up)
  List of ACs:
    BE7301.1, state: up, Static MAC addresses: 0
  List of Access PWs:
    Neighbor 128.0.0.19 pw-id 17300001, state: up, Static MAC addresses: 0
  List of VFIs:

RP/0/RSP0/CPU0# show l2vpn bridge-domain bd-name bd-73-1 detail
Fri Aug 11 12:36:27.136 EDT
Number of groups: 2, bridge-domains: 8000, Up: 8000, Shutdown: 0, Partially-programmed: 0
Default: 8000, pbb-edge: 0, pbb-core: 0
Number of ACs: 8000 Up: 8000, Down: 0, Partially-programmed: 0
Number of PWs: 12001 Up: 12000, Down: 1, Standby: 0, Partially-programmed: 0
Number of P2MP PWs: 0, Up: 0, Down: 0, other-state: 0
Number of VNIs: 0, Up: 0, Down: 0, Unresolved: 0
Coupled state: disabled
VINE state: Default
MAC learning: enabled
MAC withdraw: enabled
  MAC withdraw for Access PW: enabled
  MAC withdraw sent on: bridge port down (legacy)
  MAC withdraw relaying (access to access): disabled
Flooding:
  Broadcast & Multicast: enabled
  Unknown unicast: enabled
  MAC aging time: 300 s, Type: inactivity
  MAC limit: 4000, Action: none, Notification: syslog
MAC limit reached: no
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: 4
Filter MAC addresses:
P2MP PW: disabled
Create time: 08/08/2017 17:19:31 (2d19h ago)
No status change since creation
ACs: 1 (1 up), VFIs: 0, PWs: 1 (1 up), PBBs: 0 (0 up), VNIs: 0 (0 up)
List of ACs:
AC: Bundle-Ether7301.1, state is up
Type VLAN; Num Ranges: 1
Rewrite Tags: []
VLAN ranges: [1, 1]
MTU 8986; XC ID 0xc0003e82; interworking none
MAC learning: enabled
Flooding:
  Broadcast & Multicast: enabled
  Unknown unicast: enabled
MAC aging time: 300 s, Type: inactivity
MAC limit: 4000, Action: none, Notification: syslog
MAC limit reached: no
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: bridge-domain policer
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:
PW: neighbor 128.0.0.19, PW ID 17300001, state is up
PW class not set, XC ID 0xa0000013
Encapsulation MPLS, protocol none
Source address 10.0.0.1
PW type Ethernet, control word disabled, interworking none
PW backup disable delay 0 sec
Sequencing not set

MPLS  Local  Remote
Label 2000 1000
Interface Access PW
MTU 1500
Control word disabled
PW type Ethernet
VCCV CV type 0x2
  (LSP ping verification)
VCCV CC type 0x6
  (router alert label)
  (TTL expiry)

MIB cpwVcIndex: 2684354579
Create time: 08/08/2017 17:19:33 (2d19h ago)
Last time status changed: 11/08/2017 11:39:50 (00:56:46 ago)
MAC withdraw messages: sent 0, received 0
Forward-class: 0
Static MAC addresses:
  Statistics:
    packets: received 0 (unicast 0), sent 0
    bytes: received 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
MAC learning: enabled
Flooding:
  Broadcast & Multicast: enabled
  Unknown unicast: enabled
MAC aging time: 300 s, Type: inactivity
MAC limit: 4000, Action: none, Notification: syslog
MAC limit reached: no
MAC port down flush: enabled
MAC Secure: disabled, Logging: disabled
Split Horizon Group: none
DHCPv4 Snooping: disabled
DHCPv4 Snooping profile: none
IGMP Snooping: disabled
IGMP Snooping profile: none
MLD Snooping profile: none
Storm Control: bridge-domain policer
List of VFIs:
  List of Access VFIs:
/* PE1 static PW configuration details */

RP/0/RSP0/CPU0#router-PE1#show evpn ethernet-segment esi 0 14.14.00.00.00.01.00.00.03 carving
detail
Fri Aug 11 12:47:30.981 EDT
Legend:
A  - Load-balancing mode and Access Protection incompatible,
B  - No Forwarders EVPN-enabled,
C  - Backbone Source MAC missing (PBB-EVPN),
RT - ES-Import Route Target missing,
E  - ESI missing,
H  - Interface handle missing,
I  - Name (Interface or Virtual Access) missing,
M  - Interface in Down state,
O  - BGP End of Download missing,
P  - Interface already Access Protected,
Pf - Interface forced single-homed,
R  - BGP RID not received,
S  - Interface in redundancy standby state,
X  - ESI-extracted MAC Conflict
SHG - No local split-horizon-group label allocated

<table>
<thead>
<tr>
<th>Ethernet Segment Id</th>
<th>Interface</th>
<th>Nexthops</th>
</tr>
</thead>
<tbody>
<tr>
<td>0014.1400.0000.0100.0003</td>
<td>PW:10.0.0.1,17300001</td>
<td>128.0.0.1, 128.0.0.2</td>
</tr>
</tbody>
</table>

ES to BGP Gates : Ready
ES to L2FIB Gates : Ready
Virtual Access :
  Name : PW_10.0.0.1_17300001
  State : Up
  Num FW Up : 1
  ESI type : 0
  Value : 14.1400.0000.0100.0003
  ES Import RT : 1414.0001.0003 (from ESI)
  Source MAC : 0000.0000.0000 (N/A)
  Topology :
     Operational : MH
     Configured : All-active (AApF)
Primary Services : Auto-selection
Secondary Services: Auto-selection
Service Carving Results:
  Forwards : 1
  Permanent : 0
  Elected : 1
  EVI E : 1
  Not Elected : 0
MAC Flushing mode : Invalid
Peering timer : 3 sec [not running]
Recovery timer : 30 sec [not running]
Carving timer : 0 sec [not running]
Local SHG label : 32096
Remote SHG labels : 1
  32096 : nexthop 128.0.0.1

/* PE2 static PW configuration details */

RP/0/RSP0/CPU0# show evpn ethernet-segment esi 0014.1400.0000.0100.0003 carving detail

Legend:
A - Load-balancing mode and Access Protection incompatible,
B - No Forwards EVPN-enabled,
C - Backbone Source MAC missing (PBB-EVPN),
RT - ES-Import Route Target missing,
E - ESI missing,
H - Interface handle missing,
I - Name (Interface or Virtual Access) missing,
M - Interface in Down state,
O - BGP End of Download missing,
P - Interface already Access Protected,
Pf - Interface forced single-homed,
R - BGP RID not received,
S - Interface in redundancy standby state,
X - ESI-extracted MAC Conflict
SHG - No local split-horizon-group label allocated

<table>
<thead>
<tr>
<th>Ethernet Segment Id</th>
<th>Interface</th>
<th>Nexthops</th>
</tr>
</thead>
<tbody>
<tr>
<td>0014.1400.0000.0100.0003</td>
<td>PW:10.0.0.1,17300001</td>
<td>128.0.0.2, 128.0.0.1</td>
</tr>
</tbody>
</table>

ES to BGP Gates : Ready
ES to L2FIB Gates : Ready
Virtual Access :
  Name : PW_10.0.0.1_17300001
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 Cisco ASR 9000 Series Aggregation Services Router Interface and Hardware Component Configuration Guide.

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.

- **EVPN-VPWS Xconnect service.** Only all-active multihoming is supported.
  
  For more information, see *EVPN Virtual Private Wire Service (VPWS)* chapter in *L2VPN and Ethernet Services Configuration Guide for Cisco ASR 9000 Series Routers*.

- **Native EVPN with Integrated Routing and Bridging (IRB) on a single ES.** Both single-active and all-active multihoming modes are supported. However, both single-active and all-active multihoming cannot be configured on a single ES. You can configure either single-active or all-active multihoming mode on a single ES. But, they can coexist.
  
  For more information, see *Configure EVPN IRB* chapter in *L2VPN and Ethernet Services Configuration Guide for Cisco ASR 9000 Series Routers*.

- **Native EVPN.** Both single-active and all-active multihoming modes are supported. However, both single-active and all-active multihoming cannot be configured on a single ES. You can configure either single-active or all-active multihoming mode on a single ES. But, they can coexist.
  
  For more information see, *EVPN Features* chapter in *L2VPN and Ethernet Services Configuration Guide for Cisco ASR 9000 Series Routers*.

### 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.

```plain
/* 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
Route(config)# l2vpn
Route(config-l2vpn)# xconnect group xg22001
Route(config-l2vpn-xc)# p2p evpn-vpws-mclag-22001
Route(config-l2vpn-xc-p2p)# interface Bundle-Ether22001.11
Route(config-l2vpn-xc-p2p)# neighbor evpn evi 22101 target 220101 source 220301
Route(config-l2vpn-xc-p2p)# commit
Route(config-l2vpn-xc-p2p)# 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.ee
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)# exit
Router (config-evpn-evi)# 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)# 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)# advertise-mac
Router (config-evpn-evi)# exit
Router (config-evpn-evi)# 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 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
en encapsulation dot1q 1 second-dot1q 21
rewrite ingress tag pop 2 symmetric

! 12vpn
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
!
verification

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

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 | Segment 1 | Segment 2
------------------------ | ------------- | -------------
\[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 xconnect group

EVPN VXLAN Ingress Replication

The EVPN VXLAN Ingress Replication feature enables the VXLAN tunnel endpoint (VTEP) to exchange local and remote VTEP IP addresses on the Virtual Network Identifier (VNI) in order to create the ingress replication list. This enables VTEPs to send and receive broadcast, unknown unicast and multicast (BUM) traffic for the VNI. These IP addresses are exchanged between VTEPs through the BGP EVPN control plane using EVPN Route Type 3. This feature enables in reduced traffic flooding, increased load sharing at VTEP, faster convergence during link and device failures, and simplified data center automation.

The VXLAN imposition node maintains a list of remote VTEP nodes that serve the same tenant VNI. Each copy of VXLAN packet is sent to the destination VTEP through underlay L3 unicast transport. EVPN Route Type 3 which is an inclusive multicast route, is used to build a replication list of VXLAN data plane VTEPs. The imposition node replicates BUM traffic for each remote VTEP node discovered by this route. Each copy of VXLAN is sent to destination VTEP through underlay L3 unicast transport. The ASR 9000 router is a DC edge router, which works as DCI gateway by stitching two MP-BGP control planes, one on the DC side, and the other on the MPLS WAN side.

Following are the use cases of this feature:

- Single Homing VXLAN L2 gateway
• Anycast VXLAN L2 gateway
• All-active multihoming VXLAN L2 gateway

**Single Homing VXLAN L2 GW**

Consider a topology of single homing L2 gateway between DC and WAN. In this topology, ASR 9000 router is the DCI PE router. The L2 gateway on the PE is a bridge which forwards L2 frames between VXLAN DC and MPLS WAN. DC fabric devices, such as leaf and spine nodes, do not run IP multicast protocols, such as PIM-SM. All L2 BUM traffic between Nexus 9000 router and ASR 9000 router is forwarded through ingress replication at VXLAN imposition node.

*Figure 12: Single Homing VXLAN L2 GW*

A tenant VNI is enabled on all the four Nexus 9000 leaf nodes and one ASR 9000 border leaf node for L2VPN service. An IP host in DC1 initiates a communication to another IP host in DC2. The first ARP request goes from DC1 to DC2. Nexus 9000 router receives the ARP first, and uses ingress replication approach to flood the frame to other leaf nodes in DC1. One copy arrives on border leaf node ASR 9000. ASR 9000 performs L2 gateway operation. It replicates traffic using per EVI replication list at MPLS WAN side. One copy is sent to DC2. The other to DC3.

In the reverse direction, when an IP host in DC2 initiates a communication with an IP host in DC1, an ARP request arrives at ASR 9000 DCI PE from WAN. ASR 9000 performs L2 gateway operation using per VNI ingress replication list for VXLAN. A total of four copies are created. Each copy is sent to one Nexus 9000 leaf node. Nexus 9000 leaf nodes that are configured as DFs forward the traffic to IP hosts on VMs.
Anycast VXLAN L2 Gateway

Anycast VXLAN L2 gateway requires multihoming gateway nodes to use a common VTEP IP address. Gateway nodes in the same DC advertise the common VTEP IP in all EVPN routes from type 2 to type 5. Nexus 9000 leaf nodes in the DC considers only one border leaf VTEP located on multiple physical gateway nodes. Each Nexus 9000 router forwards traffic to the nearest gateway node through IGP routing.

Among multihoming DCI gateway nodes, an EVPN Ethernet segment is created on VXLAN facing NVE interface. One of the nodes is elected as DF for a tenant VNI. The DF node floods BUM traffic from WAN to DC. All DCI PE nodes discover each other through EVPN inclusive multicast routes advertised through WAN.

Consider a topology of anycast VXLAN L2 gateway between DC and WAN. In this topology, both ASR 9000 PE nodes share the same source VTEP IP address (VTEP5). Nexus 9000 router runs in vPC mode. ASR 9000 nodes advertise inclusive multicast routes using VTEP5 IP address. Nexus 9000 leaf nodes discover only one VTEP hosted by two ASR 9000 nodes.

Figure 13: Anycast VXLAN L2 Gateway
When the Nexus 9000 router in DC1 receives BUM traffic from local IP host, it sends one copy to VTEP5. IGP routing in underlay transport chooses the nearest ASR 9000 router as the destination. After ASR 9000 router receives the L2 frame, it replicates it to MPLS WAN side. Three copies are sent to WAN. One arrives on peer ASR 9000 router in the same DC. The copy is dropped on peer PE using Ethernet Segment Split-Horizon feature.

In the direction from DC2 and DC3 to DC1, both ASR 9000 DCI PE nodes receive the same BUM traffic from MPLS WAN. The DF PE for the tenant VNI forwards traffic to DC1. Non-DF PE drops BUM traffic from WAN.

All-Active Multihoming VXLAN L2 Gateway

Consider a topology of all-active multihoming VXLAN L2 gateway where all leaf nodes, including Nexus 9000 node and ASR 9000 node, each has an unique VTEP IP address. Each Nexus 9000 leaf node creates EVPN Ethernet segment (ES1 and ES2) for dual-homed VM server. ASR 9000 border leaf nodes create an Ethernet Segment (ES3) for VXLAN facing NVE interface. Since every leaf node advertises inclusive multicast route using its local VTEP IP, ASR 9000 node receives four routes from Nexus 9000 node. The per VNI ingress replication list includes four remote VTEP (VTEP1 to VTEP4). Every Nexus 9000 node receives two routes from ASR 9000 gateway nodes. It sends BUM traffic to both ASR 9000 nodes. To prevent traffic duplication, only one of the ASR 9000 nodes can accept VXLAN traffic from Nexus 9000 leaf using DF. DF election is done at per tenant VNI level. One half of the VNIs elect top PE as DF. The other half elect bottom PE. DF PE accepts traffic both from DC and WAN. Non-DF drops traffic from DC and WAN.
Figure 14: All-Active Multihoming VXLAN L2 Gateway

BUM traffic from DC1 arrives at Nexus 9000 leaf first. Nexus 9000 replicates the traffic to two ASR 9000 DCI nodes. DF DCI nodes flood traffic to WAN. Non-DF node drops traffic from DC fabric. Traffic flooded to WAN goes to DC2 and DC3. One copy comes back to DC1 through bottom DCI node. The bottom DCI node compares the split horizon label in the received MPLS packet and drops the packet.

In the reverse direction, when the traffic flows from DC2 and DC3, towards DC1, arrives at both top and bottom DCI nodes. The bottom DCI which is a non-DF drops traffic. The top DCI which is a DF, forwards four copies to remote leaf nodes. The Nexus 9000 leaf nodes forward traffic to an IP host.

Configure EVPN VXLAN Ingress Replication

Perform the following tasks to configure EVPN VXLAN Ingress Replication feature:

- Configure DCI
- Configure ToR
/* DCI Configuration */
/* Configure Network Virtualization Endpoint (NVE) Interface */
Router# configure
Router(config)# interface nve 40
Router(config-if)# member vni 40002
Router(config-if)# host-reachability protocol bgp
Router(config-if)# source-interface loopback 40
Router(config-if)# anycast source-interface Loopback41
Router(config-if)# ingress-replication protocol bgp
Router(config-if)# commit
/* Configure a Bridge Domain */
Router# configure
Router(config)# l2vpn
Router(config-l2vpn)# bridge group bg1
Router(config-l2vpn-bg)# bridge-domain bd2
Router(config-l2vpn-bg-bd)# evi 40
Router(config-l2vpn-bg-bd-evi)# exit
Router(config-l2vpn-bg-bd-vni)# member vni 40002
Router(config-l2vpn-bg-bd-vni)# commit
/* Configure Ethernet Segment Identifier */
Router# configure
Router(config)# evpn
Router(config-evpn)# interface nve 40
Router(config-evpn-ac)# ethernet-segment
Router(config-evpn-ac-es)# identifier type 0 28.28.28.00.00.40.00.00.13
Router(config-evpn-ac-es)# bgp route-target 200:40000 stitching
Router(config-evpn-ac-es)# commit
/* Configure the routing sessions between the DCI and ToR */
Router# configure
Router(config)# route bgp 100
Router(config-bgp)# bgp router-id 192.168.0.4
Router(config-bgp)# address-family l2vpn evpn
Router(config-bgp-af)# exit
Router(config-bgp)# neighbor 15.15.15.5 ------> ToR ebgp neighbour
Router(config-bgp-nbr)# remote-as 200
Router(config-bgp-nbr)# ebgp-multihop 255
Router(config-bgp-nbr)# address-family l2vpn evpn
Router(config-bgp-nbr-af)# import stitching-rt reoriginate
Router(config-bgp-nbr-af)# route-policy pass-all in
Router(config-bgp-nbr-af)# encapsulation-type vxlan
Router(config-bgp-nbr-af)# route-policy pass-all out
Router(config-bgp-nbr-af)# advertise l2vpn evpn re-originated stitching-rt
Router(config-bgp-nbr-af)# commit
Router(config-bgp-nbr-af)# exit
Router(config-bgp)# neighbor 192.168.0.2 ------> DCI BGP neighbour
Router(config-bgp-nbr)# remote-as 100
Router(config-bgp-nbr)# update-source Loopback0
Router(config-bgp-nbr)# address-family l2vpn evpn
Router(config-bgp-nbr-af)# import stitching-rt reoriginate
Router(config-bgp-nbr-af)# advertise l2vpn evpn re-originated stitching-rt
Router(config-bgp-nbr-af)# commit
/* ToR Configuration */

/* Configure Network Virtualization Endpoint (NVE) Interface */

Router# configure
Router(config)# interface nve 40
Router(config-if)# member vni 40002
Router(config-if)# host-reachability protocol bgp
Router(config-if)# source-interface loopback 40
Router(config-if)# anycast source-interface Loopback41
Router(config-if)# ingress-replication protocol bgp
Router(config-if)# commit

/* Configure RD and Route Targets for VXLAN Bridging */

Router# configure
Router(config)# evpn
Router(config-evpn)# router bgp
Router(config-evpn-bgp)# rd auto
Router(config-evpn-bgp)# route-target import auto
Router(config-evpn-bgp)# route-target import 200:40000
Router(config-evpn-bgp)# route-target export 200:40000
Router(config-evpn-bgp)# commit

/* Configure the routing sessions between the ToR and DCI */

Router# configure
Router(config)# router bgp 200
Router(config-bgp)# bgp router-id 10.5.41.41
Router(config-bgp)# address-family l2vpn evpn
Router(config-bgp-af)# maximum-paths 8
Router(config-bgp-af)# maximum-paths ibgp 8
Router(config-bgp-af)# exit

Router(config-bgp)# 192.168.0.4 -------> DCI neighbour: ebgp
Router(config-bgp-nbr)# remote-as 100
Router(config-bgp-nbr)# update-source Loopback0
Router(config-bgp-nbr)# ebgp-multihop 255
Router(config-bgp-nbr)# address-family ipv4 unicast
Router(config-bgp-nbr-af)# address-family l2vpn evpn
Router(config-bgp-nbr-af)# send-community extended
Router(config-bgp-nbr-af)# route-map passall in
Router(config-bgp-nbr-af)# route-map IR-test out
Router(config-bgp-nbr-af)# exit

Router(config-bgp)# neighbor 192.168.0.2 -------> VXLAN neighbour
Router(config-bgp-nbr)# remote-as 200
Router(config-bgp-nbr)# update-source Loopback0
Router(config-bgp-nbr)# address-family l2vpn evpn
Router(config-bgp-nbr-af)# send-community extended
Router(config-bgp-nbr-af)# commit

Running Configuration

/* DCI Configuration */

interface nve40

member vni 40002

host-reachability protocol bgp
Running Configuration

source-interface Loopback40
anycast source-interface Loopback41
ingress-replication protocol bgp

l2vpn
bridge group bg1
bridge-domain bd2
evi 40
member vni 40002

evpn
interface nve 40
ethernet-segment
  identifier type 0 28.28.28.00.00.40.00.00.13
  bgp route-target 200:40000 stitching

evpn evi 40
  bgp route-target 200:40000 stitching
router bgp 100
  bgp router-id 192.168.0.4
  address-family l2vpn evpn
  neighbor 15.15.15.5 -------> TOR ebgp neighbor
    remote-as 200
    ebgp-multihop 255
    address-family l2vpn evpn
      import stitching-rt re-originate
      route-policy pass-all in
      encapsulation-type vxlan
      route-policy pass-all out
      next-hop-self
      advertise l2vpn evpn re-originated stitching-rt

neighbor 192.168.0.2 ---------> DCI BGP neighbor
  remote-as 100
  update-source Loopback0
  address-family l2vpn evpn
  import re-originate stitching-rt
  advertise l2vpn evpn re-originated

/* ToR Configuration */

interface nve 40
member vni 40002
host-reachability protocol bgp
source-interface loopback 40
anycast source-interface Loopback41
ingress-replication protocol bgp

evpn
router bgp
  rd auto
  route-target import auto
  route-target import 200:40000
  route-target export 200:40000

router bgp 200
  bgp router-id 10.5.41.41
  address-family l2vpn evpn
  maximum-paths 8
  maximum-paths ibgp 8

  neighbor 192.168.0.4 ------------> DCI neighbour: ebgp
remote-as 100
update-source loopback0
ebgp-multihop 255
address-family ipv4 unicast
address-family l2vpn evpn
send-community extended
route-map passall in
route-map IR-test out

neighbor 192.168.0.6 ------> VXLAN neighbour
remote-as 200
update-source loopback0
address-family l2vpn evpn
send-community both

Verification

Verify that you have configured EVPN VXLAN Ingress Replication feature successfully.

DC3# show evpn evi vpn-id 40 inclusive-multicast detail
Ethernet Tag: 0, Originating IP: 192.168.0.2, vpn-id: 40
   Nexthop: 192.168.0.2
   Label : 24004
   Source : Remote
   Encap : MPLS
Ethernet Tag: 0, Originating IP: 192.168.0.3, vpn-id: 40
   Nexthop: 192.168.0.3
   Label : 24003
   Source : Remote
   Encap : MPLS
Ethernet Tag: 0, Originating IP: 192.168.0.4, vpn-id: 40
   Nexthop: ::
   Label : 24001
   Source : Local
   Encap : MPLS

DC2# show evpn ethernet-segment interface nve 40 detail

Ethernet Segment Id Interface Nexthops
------------------------------- ------------------- ---------------------
0028.2828.0000.4000.0013 nv40 128.0.0.1
128.0.0.2

ES to BGP Gates : Ready
ES to L2FIB Gates : Ready
Main port :
   Interface name : nv40
   Interface MAC : 0000.0000.0000
   IfHandle : 0x000003e960
   State : Up
   Redundancy : Not Defined
ESI type : 0
   Value : 28.2828.0000.4000.0013
ES Import RT : 2828.2800.0040 (from ESI)
Source MAC : 0000.0000.0000 (N/A)
Topology :
   Operational : MNH
   Configured : All-active (AApF) (default)
Primary Services : Auto-selection
Secondary Services: Auto-selection
Service Carving Results:
Forwarders : 4000
Permanent : 0
Elected : 2000
Not Elected : 2000
MAC Flushing mode : Invalid
Peering timer : 30 sec [not running]
Recovery timer : 30 sec [not running]
Carving timer : 0 sec [not running]
Local SHG label : 38029
Remote SHG labels : 1
46029 : nexthop 128.0.0.1

DCI# show l2vpn forwarding protection main-interface nve 40 location 0/2/CPU0
Main Interface ID Instance State
------------------------------------------
nve40 0 FORWARDING
nve40 1 FORWARDING
nve40 2 PE2CEBLOCK
nve40 3 FORWARDING
nve40 4 PE2CEBLOCK
nve40 5 FORWARDING
nve40 6 PE2CEBLOCK
nve40 7 FORWARDING
nve40 8 PE2CEBLOCK
nve40 9 FORWARDING
nve40 10 PE2CEBLOCK
nve40 11 FORWARDING
nve40 12 PE2CEBLOCK
nve40 13 FORWARDING
nve40 14 PE2CEBLOCK

DC3# show evpn evi vpn-id 40 inclusive-multicast detail
Ethernet Tag: 0, Originating IP: 10.4.41.41, vpn-id: 40
  Nexthop: ::
  Label : 40000
  Source : Local
  Encap : VXLAN
Ethernet Tag: 0, Originating IP: 10.5.41.41, vpn-id: 40
  Nexthop: 10.5.41.41
  Label : 40000
  Source : Remote
  Encap : VXLAN
Ethernet Tag: 0, Originating IP: 10.6.41.41, vpn-id: 40
  Nexthop: 10.6.41.41
  Label : 40000
  Source : Remote
  Encap : VXLAN

DC3# show l2vpn forwarding bridge-domain evpn inclusive-multicast location 0/0/CPU0
Bridge-Domain Name BD-ID XCID Next Hop Label/VNI
------------------------------------------
l2cp-ir:l2cp-40 1 0xffff01002 192.168.0.2 192.168.0.3 24004 24003 ;; MPLS-side
l2cp-ir:l2cp-40 1 0xffff01002 192.168.0.2 192.168.0.3 24004 24003 ;; VXLAN side

EVPN Features
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.

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.

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.

• EVPN core facing interfaces must be physical or bundle main interfaces only. Sub-interfaces are not supported.

```
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
!
```

**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

**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 (1.1.1.1:0) [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 (1.1.1.2:0) [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)                 |
+---------------------------------------------+   |
```

**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 (1.1.1.2:0) [and/or evpn-route-type is 2] [and/or esi in (0000.0000.0000.0000.0000)] [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**

```
rout-policy evpn-policy
  if rd in (1.1.1.1:300) [and/or evpn-route-type is 3] [and/or etag is 0] [and/or evpn-originator in (1.1.1.1)] 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:
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

```bash
route-policy evpn-policy
   if rd in (1.1.1.1:0) [and/or evpn-route-type is 4] [and/or esi in (00a1.a2a3.a4a5.a6a7.a8a9)] [and/or evpn-originator in (1.1.1.1)] then
      set ..
   endif
end-policy
```

Route Type 5: IP Prefix Route

An IP Prefix Route type specific EVPN NLRI consists of the following fields:
NLRI Format: Route-type 5:

<table>
<thead>
<tr>
<th>Route Type (1 octet)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Length (1 octet)</td>
</tr>
<tr>
<td>RD (8 octets)</td>
</tr>
<tr>
<td>Ethernet Segment ID (10 octets)</td>
</tr>
<tr>
<td>Ethernet Tag ID (4 octets)</td>
</tr>
<tr>
<td>IP Address Length (1 octet)</td>
</tr>
<tr>
<td>IP Address (4 or 16 octets)</td>
</tr>
<tr>
<td>GW IP Address (4 or 16 octets)</td>
</tr>
<tr>
<td>MPLS Label (3 octets)</td>
</tr>
</tbody>
</table>

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
del-set
**mac-set**

The mac-set specifies one or more MAC addresses.

**Example**

```plaintext
cmac-set mac_address_set
1234.2345.6789,
2345.3456.7890
end-set
```

**esi-set**

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

**Example**

```plaintext
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**

```plaintext
etag-set evpn etag_set
10000,
20000
end-set
```

**EVPN Attributes and Operators**

This table summarizes the EVPN attributes and operators per attach points.
### Configure EVPN RPL Feature

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

```bash
/* Configuring a mac-set and refering 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
```

Table 4: EVPN Attributes and Operators

<table>
<thead>
<tr>
<th>Attach Point</th>
<th>Attribute</th>
<th>Match</th>
<th>Attribute-Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>neighbor-in</td>
<td>destination</td>
<td>in</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td>rd</td>
<td>in</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td>evpn-route-type</td>
<td>is</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td>esi</td>
<td>in</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>etag</td>
<td>in</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>mac</td>
<td>in</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>evpn-originator</td>
<td>in</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td>evpn-gateway</td>
<td>in</td>
<td>—</td>
</tr>
<tr>
<td>neighbor-out</td>
<td>destination</td>
<td>in</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td>rd</td>
<td>in</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td>evpn-route-type</td>
<td>is</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td>esi</td>
<td>in</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>etag</td>
<td>in</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>mac</td>
<td>in</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>evpn-originator</td>
<td>in</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td>evpn-gateway</td>
<td>in</td>
<td>—</td>
</tr>
</tbody>
</table>
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-af)# address-family ipv4 unicast
Router(config-bgp-nbr-af)# route-policy policy_use_pass_mac_set in
Router(config-bgp-nbr-af)# commit

/* Configuring a esi-set and refering 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-af)# 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 refering 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 ipv4 unicast
!
neighbor 10.0.0.10
remote-as 8
address-family ipv4 unicast
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
  ad34.123.1222.ffff.44ff,
ad34.123.1222.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:*1) 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
! 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
  else
    pass
  endif
end-policy
!
drop
elseif evpn-route-type is 5 and esi in evpn_esi_set1 then
  drop
else
  pass
endif
end-policy
!