Configuring EVPN VXLAN Integrated Routing and Bridging

Information About EVPN VXLAN Integrated Routing and Bridging

EVPN VXLAN integrated routing and bridging (IRB) allows the VTEPs or leaf switches in an EVPN VXLAN network to perform both bridging and routing. IRB allows the VTEPs to forward both Layer 2 or bridged and Layer 3 or routed traffic. A VTEP performs bridging when it forwards traffic to the same subnet. Similarly, a VTEP performs routing when it forwards traffic to a different subnet. The VTEPs in the network forward traffic to each other through the VXLAN gateways. BGP EVPN VXLAN implements IRB in two ways:

- Asymmetric IRB
- Symmetric IRB

Asymmetric IRB

In asymmetric IRB, the ingress VTEP performs both bridging and routing whereas the egress VTEP performs only bridging. A packet first moves through a MAC VRF followed by an IP VRF on the network virtualisation endpoint (NVE) of the ingress VTEP. It then moves only through a MAC VRF on the NVE of the egress VTEP. The NVE of the ingress VTEP manages all the packet processing associated with intersubnet forwarding semantics.

The return traffic during asymmetric IRB goes through a different virtual network instance (VNI) compared to the source traffic. Asymmetric IRB needs the source and destination VNIs to be associated with both the ingress and egress VTEPs.

Symmetric IRB

In symmetric IRB, both the ingress and egress VTEPs perform both bridging and routing. A packet first moves through a MAC VRF followed by an IP VRF on the NVE of the ingress VTEP. It then moves through an IP VRF followed by a MAC VRF on the NVE of the egress VTEP. The NVEs of ingress and egress VTEPs equally share all the packet processing associated with intersubnet forwarding semantics.

In symmetric IRB, you are required to define only the VNIs of locally attached endpoints on the ingress and egress VTEPs. Symmetric IRB offers better scalability in terms of the number of VNIs that a BGP EVPN VXLAN fabric supports.

The following figure shows the implementation of symmetric IRB and the movement of traffic in an EVPN VXLAN network:
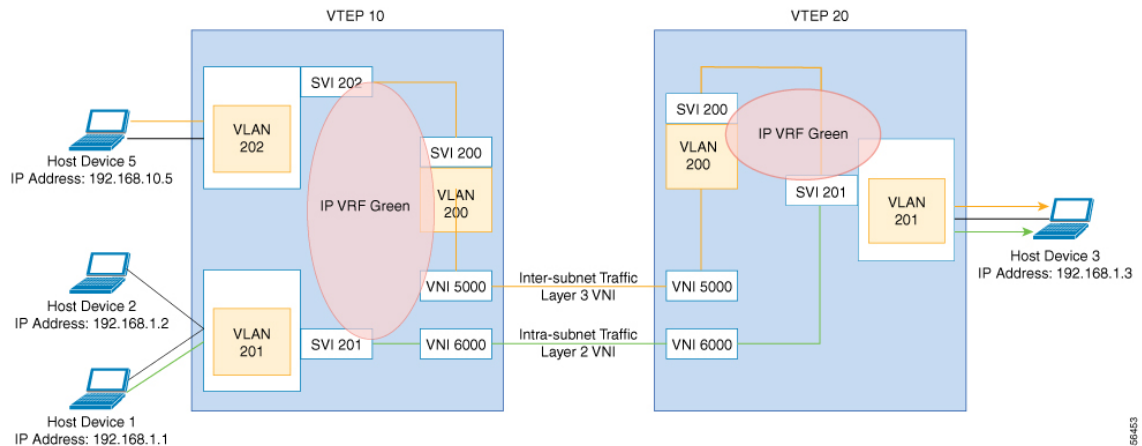
EVPN VXLAN Distributed Anycast Gateway

Distributed anycast gateway is a default gateway addressing mechanism in a BGP EVPN VXLAN fabric. The feature enables the use of the same gateway IP and MAC address across all the VETPs in an EVPN VXLAN network. This ensures that every VTEP functions as the default gateway for the workloads directly connected to it. The feature facilitates flexible workload placement, host mobility, and optimal traffic forwarding across the BGP EVPN VXLAN fabric.

The scenario shown in the following figure depicts a distributed gateway. Subnet 1 contains two leaf switches, leaf switch 1 and leaf switch 2, acting together as a distributed default gateway for VLAN 10. Host device 1 is connected to leaf switch 1 and needs to send traffic to host device 3, which is in a different subnet. When host device 1 tries to send traffic outside of subnet 1, the traffic goes through the configured gateway on leaf switch 1. Host device 1 registers the Address Resolution Protocol (ARP) entries of the gateway VLAN MAC and IP addresses on leaf switch 1.
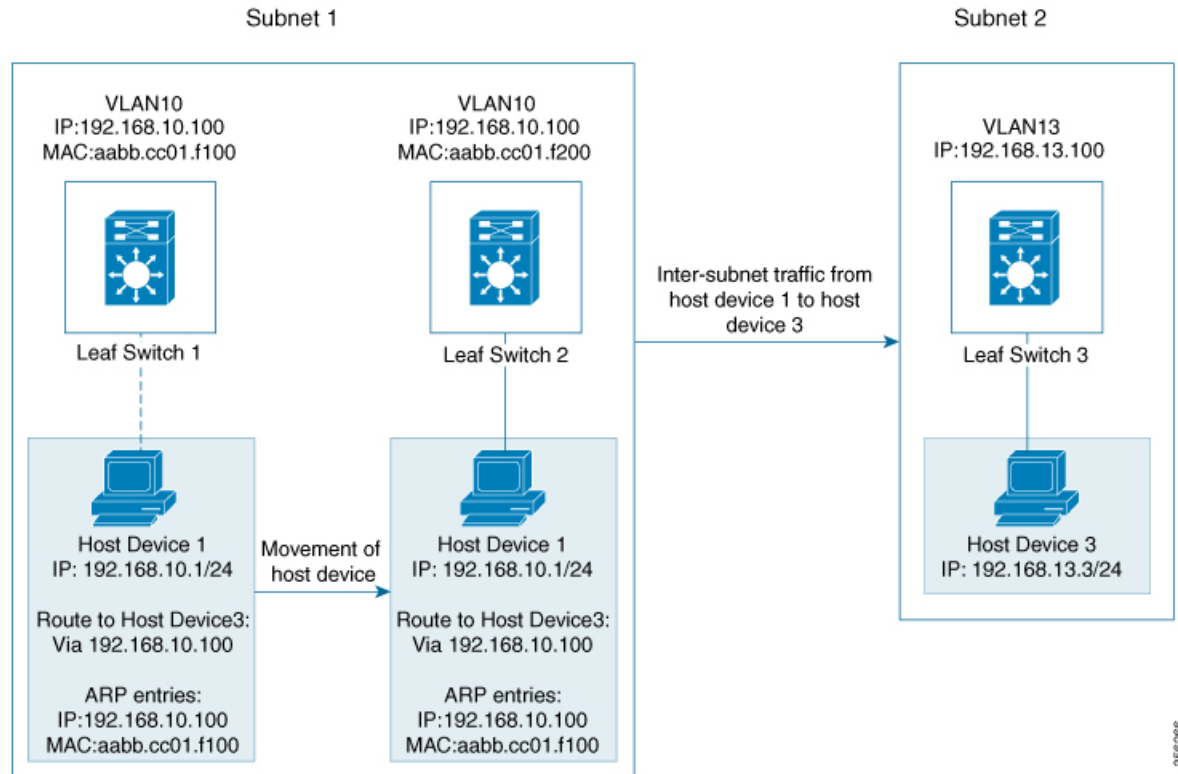
When multiple VETPs act together as one single distributed default gateway for the same VLAN, the VLAN IP address remains the same across all of them. This IP address becomes the gateway IP address for any host device in the VLAN that tries to reach an IP address outside its subnet. But, each VTEP retains its own MAC address.

In the preceding figure, consider the scenario where host device 1 moves from leaf switch 1 to leaf switch 2. The host device remains within the same network and still maintains the same ARP entries for gateway MAC and IP addresses. But the MAC addresses of the VLAN interfaces on leaf switch 2 and leaf switch 1 are different. This results in a MAC address mismatch between the ARP entry and the VLAN on leaf switch 2. As a result, any traffic that host device 1 tries to send outside of Subnet 1 is either lost or continuously flooded as unknown unicast. EVPN VXLAN distributed anycast gateway feature prevents this traffic loss by ensuring that all the VTEPs have the same gateway MAC and IP addresses.

There are two ways to maintain the same MAC address across all VTEPs and configure distributed anycast gateway:

- Manual MAC address configuration

- MAC aliasing

Manual MAC Address Configuration

Manual MAC address configuration is the conventional method of enabling distributed anycast gateway in an EVPN VXLAN network. In this method, you manually configure the same MAC address on the Layer 2

VNI VLAN SVI on all the VTEPs in the network. You must configure the same MAC address on all the VTEPs in the same Layer 2 VNI.

Note The VLAN SVIs on all the leaf switches must already share the same gateway IP address.

In the Figure 1: Distributed Gateway Topology, on page 3 image, to enable distributed anycast gateway in subnet 1, configure the same MAC address on leaf switch 1 and leaf switch 2. This ensures that the ARP entries of gateway MAC and IP addresses on host device 1 match with the MAC and IP addresses of both leaf switch 1 and leaf switch 2.

MAC Aliasing

MAC aliasing for distributed anycast gateway removes the need to configure the same MAC address explicitly on the VLAN interfaces of every VTEP. MAC aliasing allows the VTEPs to advertise their VLAN MAC addresses as the gateway MAC addresses to all the other VTEPs in the network. The VTEPs in the network store the advertised MAC address as a gateway MAC address provided their VLAN IP address matches with the gateway IP address.

In the Figure 1: Distributed Gateway Topology, on page 3 image, consider the scenario where MAC aliasing is enabled in subnet 1. Leaf switch 1 and leaf switch 2 advertise their MAC addresses to each other as gateway MAC addresses. This allows leaf switch 2 to recognize the MAC address in the ARP entry of host device 1 as a gateway MAC address. It allows host device 1 to send traffic outside of subnet 1 even though its VLAN MAC address does not match with the ARP entry.

MAC aliasing in an EVPN VXLAN network is configured by enabling the default gateway advertisement on all the VTEPs.

How to Configure EVPN VXLAN Integrated Routing and Bridging

To configure EVPN VXLAN IRB, you need to configure EVPN VXLAN Layer 2 and Layer 3 overlay networks, and enable the gateways in the VXLAN network.

To enable IRB in a VXLAN network using distributed anycast gateway, perform the following set of procedures:

- Configure Layer 2 VPN EVPN on the VTEPs.

  Enable distributed anycast gateway for the VXLAN network when you configure Layer 2 VPN.

- Configure the core-facing and access-facing VLANs on the VTEPs.

- Configure switch virtual interface (SVI) for the core-facing VLAN on the VTEPs.

- Configure SVI for the access-facing VLAN on the VTEPs.

- Configure the IP VRF on the VTEPs.

- Configure the Loopback interface on the VTEPs.

- Configure the Network Virtualization Endpoint (NVE) interface on the VTEPs.

- Configure BGP with EVPN address family on the VTEPs.

Configuring Layer 2 VPN EVPN on a VTEP

See Configuring Layer 2 VPN EVPN on a VTEP for detailed steps.

Configuring IP VRF on VTEP

See Configuring an IP VRF on a VTEP for detailed steps.

Configuring Core-facing and Access-facing VLANs on a VTEP

To configure the core-facing and access-facing VLANs on a VTEP and enable IRB in the EVPN VXLAN network, perform the following steps:

Procedure

	Command or Action	**Purpose**
Step 1	**enable** **Example:** Device> **enable**	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** Device# **configure terminal**	Enters global configuration mode.
Step 3	**vlan configuration** *vlan-id* **Example:** Device(config)# **vlan configuration 201**	Enters VLAN feature configuration mode for the specified VLAN interface.
Step 4	**member evpn-instance** *evpn-instance-id* **vni** *l2-vni-number* **Example:** Device(config-vlan)# **member evpn-instance 1 vni 6000**	Adds EVPN instance as a member of the VLAN configuration. The VNI here is used as a Layer 2 VNI.
Step 5	**exit** **Example:** Device(config-vlan)# **exit**	Returns to global configuration mode.
Step 6	**vlan configuration** *vlan-id* **Example:** Device(config)# **vlan configuration 202**	Enters VLAN feature configuration mode for the specified VLAN interface.
Step 7	**member evpn-instance** *evpn-instance-id* **vni** *l2-vni-number* **Example:**	Adds EVPN instance as a member of the VLAN configuration. The VNI here is used as a Layer 2 VNI.

	Command or Action	Purpose
	Device(config-vlan)# **member evpn-instance 2 vni 7000**	
Step 8	**exit** **Example:** Device(config-vlan)# **exit**	Returns to global configuration mode.
Step 9	**vlan configuration** *vlan-id* **Example:** Device(config)# **vlan configuration 200**	Enters VLAN feature configuration mode for the specified VLAN interface.
Step 10	**member vni** *l3-vni-number* **Example:** Device(config-vlan)# **member vni 5000**	Adds EVPN instance as a member of the VLAN configuration. The VNI here is used as a Layer 3 VNI.
Step 11	**exit** **Example:** Device(config-vlan)# **exit**	Returns to global configuration mode.
Step 12	**end** **Example:** Device(config-vlan)# **end**	Returns to privileged EXEC mode.

Configuring Switch Virtual Interface for the Core-facing VLAN on a VTEP

To configure an SVI for the core-facing VLAN on a VTEP, perform the following steps:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** Device> **enable**	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** Device# **configure terminal**	Enters global configuration mode.
Step 3	**interface vlan** *vlan-id* **Example:** Device(config)# **interface vlan 200**	Enters interface configuration mode for the specified VLAN.
Step 4	**vrf forwarding** *vrf-name* **Example:**	Configures the SVI for the VLAN.

	Command or Action	Purpose
	Device(config-if)# **vrf forwarding Green**	
Step 5	**ip unnumbered***Loopback-interface* **Example:** Device(config-if)# **ip unnumbered Loopback0**	Enables IP processing on the Loopback interface without assigning an explicit IP address to the interface.
Step 6	**no autostate** **Example:** Device(config-if)# **no autostate**	Disables autostate on the interface. In EVPN deployments, once a VLAN is used for a core-facing SVI, it should not be allowed in any trunk. For a core-facing SVI to function properly, the **no autostate** command must be configured under the SVI.
Step 7	**end** **Example:** Device(config-if)# **end**	Returns to privileged EXEC mode.

Configuring Switch Virtual Interface for the Access-facing VLANs on a VTEP

To configure SVIs for the access-facing VLANs on a VTEP, perform the following steps:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** Device> **enable**	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** Device# **configure terminal**	Enters global configuration mode.
Step 3	**interface vlan** *vlan-id* **Example:** Device(config)# **interface vlan 202**	Enters interface configuration mode for the specified VLAN.
Step 4	**vrf forwarding** *vrf-name* **Example:** Device(config-if)# **vrf forwarding Green**	Configures the SVI for the VLAN.
Step 5	**ip address** *gateway-ip-address* **Example:** Device(config-if)# **ip address 192.168.10.1 255.255.255.0**	Configures the gateway IP address for the access SVI. Configure the same gateway IP address for this SVI on all the other VTEPs.

	Command or Action	**Purpose**
Step 6	**mac-address** *mac-address-value* **Example:** Device(config-if)# **mac-address aabb.cc01.f100**	(Optional) Manually sets the MAC address for the VLAN interface. To configure distributed anycast gateway in a VXLAN network using manual MAC configuration, configure the same MAC address on the corresponding Layer 2 VNI SVIs on all the VTEPs in a VXLAN network.
Step 7	**end** **Example:** Device(config-if)# **end**	Returns to privileged EXEC mode.

Configuring the Loopback Interface on a VTEP

See Configuring the Loopback Interface on a VTEP for detailed steps.

Configuring the NVE Interface on a VTEP

To add Layer 2 and Layer 3 VNI members to the NVE interface of a VTEP, perform the following steps:

Procedure

	Command or Action	**Purpose**
Step 1	**enable** **Example:** Device> **enable**	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** Device# **configure terminal**	Enters global configuration mode.
Step 3	**interface** *nve-interface-id* **Example:** Device(config)# **interface nve1**	Defines the interface to be configured as a trunk, and enters interface configuration mode.
Step 4	**no ip address** **Example:** Device(config-if)# **no ip address**	Disables IP processing on the interface by removing its IP address.
Step 5	**source-interface** *loopback-interface-id* **Example:** Device(config-if)# **source-interface loopback0**	Sets the IP address of the specified loopback interface as the source IP address.

	Command or Action	**Purpose**
Step 6	**host-reachability protocol bgp** **Example:** Device(config-if)# **host-reachability protocol bgp**	Configures BGP as the host-reachability protocol on the interface.
Step 7	**member vni** *layer2-vni-id* {**ingress-replication** | **mcast-group** *multicast-group-address* **Example:** Device(config-if)# **member vni 6000 mcast-group 227.0.0.1** Device(config-if)# **member vni 7000 mcast-group 227.0.0.1**	Associates the Layer 2 VNI member with the NVE. The specified replication type must match the replication type that is configured globally or for the specific EVPN instance. Use **mcast-group** keyword for static replication and **ingress-replication** keyword for ingress replication.
Step 8	**member vni** *layer3-vni-id* **vrf** *vrf-name* **Example:** Device(config-if)# **member vni 5000 vrf Green**	Associates the Layer 3 VNI member with the NVE.
Step 9	**end** **Example:** Device(config-if)# **end**	Returns to privileged EXEC mode.

Configuring BGP with EVPN and VRF Address Families on a VTEP

To configure BGP on a VTEP with EVPN and VRF address families and a spine switch as the neighbor, perform these steps:

Procedure

|	**Command or Action**	**Purpose**
Step 1	**enable** **Example:** Device> **enable**	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** Device# **configure terminal**	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** Device(config)# **router bgp 1**	Enables a BGP routing process, assigns it an autonomous system number, and enters router configuration mode.

	Command or Action	Purpose
Step 4	**bgp log-neighbor-changes** **Example:** Device(config-router)# **bgp log-neighbor-changes**	(Optional) Enables the generation of logging messages when the status of a BGP neighbor changes. For more information, see *Configuring BGP* module of the *IP Routing Configuration Guide*.
Step 5	**bgp update-delay** *time-period* **Example:** Device(config-router)# **bgp update-delay 1**	(Optional) Sets the maximum initial delay period before sending the first update. For more information, see *Configuring BGP* module of the *IP Routing Configuration Guide*.
Step 6	**bgp graceful-restart** **Example:** Device(config-router)# **bgp graceful-restart**	(Optional) Enables the BGP graceful restart capability for all BGP neighbors. For more information, see *Configuring BGP* module of the *IP Routing Configuration Guide*.
Step 7	**no bgp default ipv4-unicast** **Example:** Device(config-router)# **no bgp default ipv4-unicast**	(Optional) Disables default IPv4 unicast address family for BGP peering session establishment. For more information, see *Configuring BGP* module of the *IP Routing Configuration Guide*.
Step 8	**neighbor** *ip-address* **remote-as** *number* **Example:** Device(config-router)# **neighbor 10.11.11.11 remote-as 1**	Defines multiprotocol-BGP neighbors. Under each neighbor, define the Layer 2 Virtual Private Network (L2VPN) EVPN configuration. Use the IP address of the spine switch as the neighbor IP address.
Step 9	**neighbor** {*ip-address* | *group-name*} **update-source** *interface* **Example:** Device(config-router)# **neighbor 10.11.11.11 update-source Loopback0**	Configures update source. Update source can be configured per neighbor or per peer-group. Use the IP address of the spine switch as the neighbor IP address.
Step 10	**address-family l2vpn evpn** **Example:** Device(config-router)# **address-family l2vpn evpn**	Specifies the L2VPN address family and enters address family configuration mode.
Step 11	**neighbor** *ip-address* **activate** **Example:** Device(config-router-af)# **neighbor 10.11.11.11 activate**	Enables the exchange information from a BGP neighbor. Use the IP address of the spine switch as the neighbor IP address.
Step 12	**neighbor** *ip-address* **send-community** [**both** | **extended** | **standard**]	Specifies the communities attribute sent to a BGP neighbor.

	Command or Action	Purpose
	Example: Device(config-router-af)# **neighbor 10.11.11.11 send-community both**	Use the IP address of the spine switch as the neighbor IP address.
Step 13	**exit-address-family** **Example:** Device(config-router-af)# **exit-address-family**	Exits address family configuration mode and returns to router configuration mode.
Step 14	**address-family ipv4 vrf** *vrf-name* **Example:** Device(config-router)# **address-family ipv4 vrf green**	Specifies the IPv4 address family and enters address family configuration mode.
Step 15	**advertise l2vpn evpn** **Example:** Device(config-router-af)# **advertise l2vpn evpn**	Advertises Layer 2 VPN EVPN routes within a tenant VRF in an EVPN VXLAN fabric.
Step 16	**redistribute connected** **Example:** Device(config-router-af)# **redistribute connected**	Redistributes connected routes to BGP.
Step 17	**redistribute static** **Example:** Device(config-router-af)# **redistribute static**	Redistributes static routes to BGP.
Step 18	**exit-address-family** **Example:** Device(config-router-af)# **exit-address-family**	Exits address family configuration mode and returns to router configuration mode.
Step 19	**address-family ipv6 vrf** *vrf-name* **Example:** Device(config-router)# **address-family ipv6 vrf green**	Specifies the IPv6 address family and enters address family configuration mode.
Step 20	**advertise l2vpn evpn** **Example:** Device(config-router-af)# **advertise l2vpn evpn**	Advertises Layer 2 VPN EVPN routes within a tenant VRF in an EVPN VXLAN fabric.
Step 21	**redistribute connected** **Example:**	Redistributes connected routes to BGP.

	Command or Action	Purpose
	Device(config-router-af)# **redistribute connected**	
Step 22	**redistribute static** **Example:** Device(config-router-af)# **redistribute static**	Redistributes static routes to BGP.
Step 23	**exit-address-family** **Example:** Device(config-router-af)# **exit-address-family**	Exits address family configuration mode and returns to router configuration mode.
Step 24	**end** **Example:** Device(config-router)# **end**	Returns to privileged EXEC mode.

Configuration Examples for EVPN VXLAN Integrated Routing and Bridging

This section provides an example to show how to enable EVPN VXLAN IRB using distributed anycast gateway. The following example shows a sample configuration for a VXLAN network with 2 VTEPs. VTEP 1 and VTEP 2 are connected to perform integrated routing and bridging.
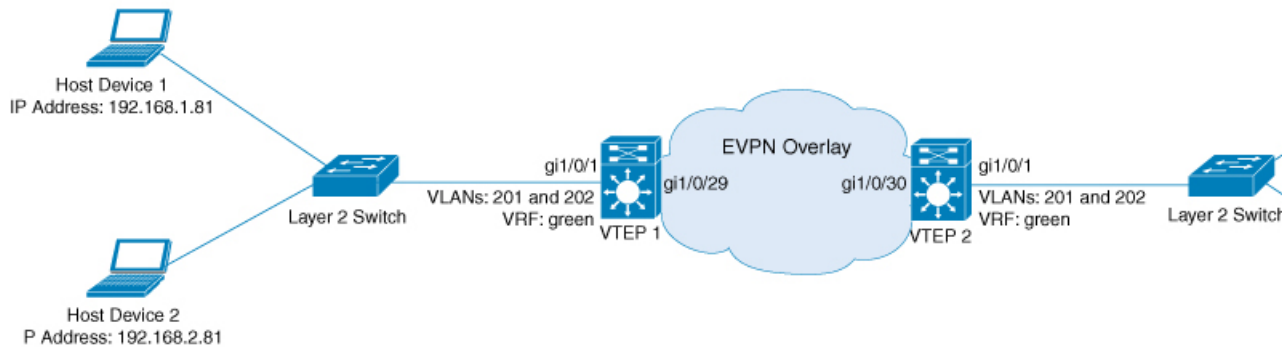
Table 1: Configuration Example for a VXLAN Network with Two VTEPs Connected to Perform Integrated Routing and Bridging Using Distributed Anycast Gateway

VTEP 1	VTEP 2

VTEP 1	VTEP 2

<pre>VTEP1# show running-config
!
hostname VTEP1
!
vrf definition green
 rd 103:2
 !
 address-family ipv4
  route-target export 103:2
  route-target import 104:2
  route-target export 103:2 stitching
  route-target import 104:2 stitching
 exit-address-family
 !
 address-family ipv6
  route-target export 103:2
  route-target import 104:2
  route-target export 103:2 stitching
  route-target import 104:2 stitching
 exit-address-family
!
ip routing
ip multicast-routing
ipv6 unicast-routing
!
!
l2vpn evpn
 replication-type static
 router-id Loopback0
 default-gateway advertise
!
l2vpn evpn instance 1 vlan-based
 encapsulation vxlan
!
l2vpn evpn instance 2 vlan-based
 encapsulation vxlan
!
!
system mtu 9150
!
vlan configuration 200
 member vni 5000
vlan configuration 201
 member evpn-instance 1 vni 6000
vlan configuration 202
 member evpn-instance 2 vni 7000
!
!
interface Loopback0
 ip address 10.1.1.10 255.255.255.255
 ip pim sparse-mode
!
interface Loopback13
description demo only (for rt5 distribution)
 vrf forwarding green
 ip address 10.1.13.13 255.255.255.0
!
interface GigabitEthernet1/0/1
description access-facing-interface
 switchport trunk allowed vlan 201,202
 switchport mode trunk
!</pre>
	<pre>VTEP2# show running-config
	!
	hostname VTEP2
	!
	vrf definition green
	 rd 104:2
	 !
	 address-family ipv4
	  route-target export 104:2
	  route-target import 103:2
	  route-target export 104:2 stitching
	  route-target import 103:2 stitching
	 exit-address-family
	 !
	 address-family ipv6
	  route-target export 104:2
	  route-target import 103:2
	  route-target export 104:2 stitching
	  route-target import 103:2 stitching
	 exit-address-family
	!
	ip routing
	ip multicast-routing
	ipv6 unicast-routing
	!
	!
	l2vpn evpn
	 replication-type static
	 router-id Loopback0
	 default-gateway advertise
	!
	l2vpn evpn instance 1 vlan-based
	 encapsulation vxlan
	!
	l2vpn evpn instance 2 vlan-based
	 encapsulation vxlan
	!
	!
	system mtu 9150
	!
	vlan configuration 200
	 member vni 5000
	vlan configuration 201
	 member evpn-instance 1 vni 6000
	vlan configuration 202
	 member evpn-instance 2 vni 7000
	!
	!
	interface Loopback0
	 ip address 10.2.2.20 255.255.255.255
	 ip pim sparse-mode
	!
	interface Loopback14
	 description demo only (for rt5 distribution)
	 vrf forwarding green
	 ip address 10.1.14.14 255.255.255.0
	!
	interface GigabitEthernet1/0/1
	description access-facing-interface
	 switchport trunk allowed vlan 201,202
	 switchport mode trunk
	!</pre>

VTEP 1	VTEP 2
| ```
!
interface GigabitEthernet1/0/29
 description core-underlay-interface
 no switchport
 ip address 172.16.1.29 255.255.255.0
 ip pim sparse-mode
!
!
interface Vlan200
 description core svi for l3vni
 vrf forwarding green
 ip unnumbered Loopback0
 ipv6 enable
 no autostate
!
interface Vlan201
 description vni 6000 default-gateway
 vrf forwarding green
 ip address 192.168.1.201 255.255.255.0
 ipv6 address 2001:DB8:201::201/64
 ipv6 enable
!
interface Vlan202
 description vni 7000 default-gateway
 vrf forwarding green
 ip address 192.168.2.202 255.255.255.0
 ipv6 address 2001:DB8:202::202/64
 ipv6 enable
!
!
interface nve10
 no ip address
 source-interface Loopback0
 host-reachability protocol bgp
 member vni 6000 mcast-group 232.1.1.1
 member vni 5000 vrf green
 member vni 7000 mcast-group 232.1.1.1
!
router ospf 1
 router-id 10.1.1.10
 network 10.1.1.0 0.0.0.255 area 0
 network 172.16.1.0 0.0.0.255 area 0
!
router bgp 10
 bgp router-id interface Loopback0
 bgp log-neighbor-changes
 bgp update-delay 1
 no bgp default ipv4-unicast
 neighbor 10.2.2.20 remote-as 10
 neighbor 10.2.2.20 update-source Loopback0
 !
 address-family ipv4
 exit-address-family
 !
 address-family l2vpn evpn
  neighbor 10.2.2.20 activate
  neighbor 10.2.2.20 send-community both
 exit-address-family
 !
 address-family ipv4 vrf green
  advertise l2vpn evpn
  redistribute connected
``` | ```
!
interface GigabitEthernet1/0/30
 description core-underlay-interface
 no switchport
 ip address 172.16.1.30 255.255.255.0
 ip pim sparse-mode
!
!
interface Vlan200
 description core svi for l3vni
 vrf forwarding green
 ip unnumbered Loopback0
 ipv6 enable
 no autostate
!
interface Vlan201
 description vni 6000 default-gateway
 vrf forwarding green
 ip address 192.168.1.201 255.255.255.0
 ipv6 address 2001:DB8:201::201/64
 ipv6 enable
!
interface Vlan202
 description vni 7000 default-gateway
 vrf forwarding green
 ip address 192.168.2.202 255.255.255.0
 ipv6 address 2001:DB8:202::202/64
 ipv6 enable
!
!
interface nve10
 no ip address
 source-interface Loopback0
 host-reachability protocol bgp
 member vni 6000 mcast-group 232.1.1.1
 member vni 7000 mcast-group 232.1.1.1
 member vni 5000 vrf green
!
router ospf 1
 router-id 10.2.2.20
 network 10.2.2.0 0.0.0.255 area 0
 network 172.16.1.0 0.0.0.255 area 0
!
router bgp 10
 bgp router-id interface Loopback0
 bgp log-neighbor-changes
 bgp update-delay 1
 no bgp default ipv4-unicast
 neighbor 10.1.1.10 remote-as 10
 neighbor 10.1.1.10 update-source Loopback0
 !
 address-family ipv4
 exit-address-family
 !
 address-family l2vpn evpn
  neighbor 10.1.1.10 activate
  neighbor 10.1.1.10 send-community both
 exit-address-family
 !
 address-family ipv4 vrf green
  advertise l2vpn evpn
  redistribute connected
``` |

VTEP 1	VTEP 2

<pre> redistribute static
 exit-address-family
 !
 address-family ipv6 vrf green
  redistribute connected
  redistribute static
  advertise l2vpn evpn
 exit-address-family
!
ip pim rp-address 10.1.1.10
!
end</pre>
	<pre> redistribute static
	 exit-address-family
	 !
	 address-family ipv6 vrf green
	  redistribute connected
	  redistribute static
	  advertise l2vpn evpn
	 exit-address-family
	!
	ip pim rp-address 10.1.1.10
	!
	end</pre>

The following examples provide outputs for **show** commands on VTEP 1 and VTEP 2 in the topology configured above:

show nve peers

VTEP 1

The following example shows the output for the **show nve peers** command on VTEP 1:

```
VTEP1# show nve peers
Interface  VNI      Type Peer-IP          RMAC/Num_RTs    eVNI       state flags UP time
nve10      5000     L3CP 10.2.2.20        380e.4d9b.6a4a 5000        UP   A/M/4 01:33:41
nve10      5000     L3CP 10.2.2.20        380e.4d9b.6a4a 5000        UP   A/-/6 00:43:38
nve10      6000     L2CP 10.2.2.20        5               6000        UP   N/A   01:33:41
nve10      7000     L2CP 10.2.2.20        6               7000        UP   N/A   01:33:41
```

VTEP 2

The following example shows the output for the **show nve peers** command on VTEP 2:

```
VTEP2# show nve peers
Interface  VNI      Type Peer-IP          RMAC/Num_RTs    eVNI       state flags UP time
nve10      5000     L3CP 10.1.1.10        a0f8.4910.bce2 5000        UP   A/M/4 01:33:55
nve10      5000     L3CP 10.1.1.10        a0f8.4910.bce2 5000        UP   A/-/6 01:14:23
nve10      6000     L2CP 10.1.1.10        7               6000        UP   N/A   01:33:55
nve10      7000     L2CP 10.1.1.10        6               7000        UP   N/A   01:33:55
```

show l2vpn evpn peers vxlan

VTEP 1

The following example shows the output for the **show l2vpn evpn peers vxlan** command on VTEP 1:

```
VTEP1# l2vpn evpn peers vxlan
Interface VNI      Peer-IP                    Num routes eVNI     UP time
--------- -------- ----------------------- ---------- -------- --------
nve10     6000     10.2.2.20                  5          6000     01:34:50
nve10     7000     10.2.2.20                  6          7000     01:34:50
```

VTEP 2

The following example shows the output for the **show l2vpn evpn peers vxlan** command on VTEP 2:

```
VTEP2# show l2vpn evpn peers vxlan
Interface VNI      Peer-IP                    Num routes eVNI     UP time
--------- -------- ----------------------- ---------- -------- --------
nve10     6000     10.1.1.10                  7          6000     01:35:23
nve10     7000     10.1.1.10                  6          7000     01:35:23
```

show l2vpn evpn evi evpn-instance detail

VTEP 1

The following example shows the output for the **show l2vpn evpn evi** *evpn-instance* **detail** command on VTEP 1:

```
VTEP1# show l2vpn evpn evi 1 detail
EVPN instance:     1 (VLAN Based)
 RD:               10.1.1.10:1 (auto)
 Import-RTs:       10:1
 Export-RTs:       10:1
 Per-EVI Label:    none
 State:            Established
 Replication Type: Static (global)
 Encapsulation:    vxlan
 IP Local Learn:   Enable (global)
 Vlan:             201
   Ethernet-Tag:   0
   State:          Established
   Core If:        Vlan200
   Access If:      Vlan201
   NVE If:         nve10
   RMAC:           a0f8.4910.bce2
   Core Vlan:      200
   L2 VNI:         6000
   L3 VNI:         5000
   VTEP IP:        10.1.1.10
   MCAST IP:       232.1.1.1
   VRF:            green
   IPv4 IRB:       Enabled
   IPv6 IRB:       Enabled
   Pseudoports:
     GigabitEthernet1/0/1 service instance 201
```

VTEP 2

The following example shows the output for the **show l2vpn evpn evi** *evpn-instance* **detail** command on VTEP 2:

```
VTEP2# show l2vpn evpn evi 1 detail
EVPN instance:    1 (VLAN Based)
 RD:              10.2.2.20:1 (auto)
 Import-RTs:      10:1
 Export-RTs:      10:1
 Per-EVI Label:   none
 State:           Established
 Replication Type: Static (global)
 Encapsulation:   vxlan
 IP Local Learn:  Enable (global)
 Vlan:            201
   Ethernet-Tag:  0
   State:         Established
   Core If:       Vlan200
   Access If:     Vlan201
   NVE If:        nve10
   RMAC:          380e.4d9b.6a4a
   Core Vlan:     200
   L2 VNI:        6000
   L3 VNI:        5000
   VTEP IP:       10.2.2.20
   MCAST IP:      232.1.1.1
   VRF:           green
   IPv4 IRB:      Enabled
   IPv6 IRB:      Enabled
   Pseudoports:
     GigabitEthernet1/0/1 service instance 201
```

show l2vpn evpn default-gateway

VTEP 1

The following example shows the output for the **show l2vpn evpn default-gateway** command on VTEP 1:

```
VTEP1# show l2vpn evpn default-gateway
Valid Default Gateway Address   EVI   VLAN  MAC Address    Source
----- ------------------------- ----- ----- -------------- --------------------
  Y   192.168.1.201             1     201   a0f8.4910.bccc Vl201
  Y   192.168.1.201             1     201   380e.4d9b.6a48 10.2.2.20
  Y   2001:DB8:201::201         1     201   a0f8.4910.bccc Vl201
  Y   2001:DB8:201::201         1     201   380e.4d9b.6a48 10.2.2.20
  Y   192.168.2.202             2     202   a0f8.4910.bcc2 Vl202
  Y   192.168.2.202             2     202   380e.4d9b.6a42 10.2.2.20
  Y   2001:DB8:202::202         2     202   a0f8.4910.bcc2 Vl202
  Y   2001:DB8:202::202         2     202   380e.4d9b.6a42 10.2.2.20
```

VTEP 2

The following example shows the output for the **show l2vpn evpn default-gateway** command on VTEP 2:

```
VTEP2# show l2vpn evpn default-gateway
Valid Default Gateway Address    EVI   VLAN  MAC Address    Source
----- ------------------------ ----- ----- ------------- --------------------
  Y   192.168.1.201                1   201  380e.4d9b.6a48 Vl201
  Y   192.168.1.201                1   201  a0f8.4910.bccc 10.1.1.10
  Y   2001:DB8:201::201            1   201  380e.4d9b.6a48 Vl201
  Y   2001:DB8:201::201            1   201  a0f8.4910.bccc 10.1.1.10
  Y   192.168.2.202                2   202  380e.4d9b.6a42 Vl202
  Y   192.168.2.202                2   202  a0f8.4910.bcc2 10.1.1.10
  Y   2001:DB8:202::202            2   202  380e.4d9b.6a42 Vl202
  Y   2001:DB8:202::202            2   202  a0f8.4910.bcc2 10.1.1.10
```

show bgp l2vpn evpn all

VTEP 1

The following example shows the output for the **show bgp l2vpn evpn all** command on VTEP 1:

```
VTEP1# show bgp l2vpn evpn all
BGP table version is 705, local router ID is 10.1.1.10
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
              t secondary path, L long-lived-stale,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 10.1.1.10:1
 *>i [2][10.1.1.10:1][0][48][0018736C56C3][0][*]/20
                   10.2.2.20                0    100      0 ?
 *>i [2][10.1.1.10:1][0][48][0018736C56C3][32][192.168.1.89]/24
                   10.2.2.20                0    100      0 ?
 *>  [2][10.1.1.10:1][0][48][0059DC50AE01][0][*]/20
                   ::                                 32768 ?
 *>  [2][10.1.1.10:1][0][48][0059DC50AE4C][0][*]/20
                   ::                                 32768 ?
 *>  [2][10.1.1.10:1][0][48][0059DC50AE4C][32][192.168.1.81]/24
                   ::                                 32768 ?
 *>  [2][10.1.1.10:1][0][48][0059DC50AE4C][128][2001:DB8:201::81]/36
                   ::                                 32768 ?
 *>  [2][10.1.1.10:1][0][48][0059DC50AE4C][128][FE80::259:DCFF:FE50:AE4C]/36
                   ::                                 32768 ?
 *>i [2][10.1.1.10:1][0][48][380E4D9B6A48][32][192.168.1.201]/24
                   10.2.2.20                0    100      0 ?
 *>i [2][10.1.1.10:1][0][48][380E4D9B6A48][128][2001:DB8:201::201]/36
                   10.2.2.20                0    100      0 ?
 *>  [2][10.1.1.10:1][0][48][A0F84910BCCC][32][192.168.1.201]/24
                   ::                                 32768 ?
 *>  [2][10.1.1.10:1][0][48][A0F84910BCCC][128][2001:DB8:201::201]/36
                   ::                                 32768 ?
Route Distinguisher: 10.1.1.10:2
 *>i [2][10.1.1.10:2][0][48][0018736C5681][0][*]/20
                   10.2.2.20                0    100      0 ?
 *>i [2][10.1.1.10:2][0][48][0018736C56C2][0][*]/20
                   10.2.2.20                0    100      0 ?
 *>i [2][10.1.1.10:2][0][48][0018736C56C2][32][192.168.2.89]/24
                   10.2.2.20                0    100      0 ?
 *>  [2][10.1.1.10:2][0][48][0059DC50AE01][0][*]/20
                   ::                                 32768 ?
 *>  [2][10.1.1.10:2][0][48][0059DC50AE42][0][*]/20
```

```
                            ::                                  32768 ?
 *>   [2][10.1.1.10:2][0][48][0059DC50AE42][32][192.168.2.81]/24
                            ::                                  32768 ?
 *>i  [2][10.1.1.10:2][0][48][380E4D9B6A42][32][192.168.2.202]/24
                            10.2.2.20             0   100      0 ?
 *>i  [2][10.1.1.10:2][0][48][380E4D9B6A42][128][2001:DB8:202::202]/36
                            10.2.2.20             0   100      0 ?
 *>   [2][10.1.1.10:2][0][48][A0F84910BCC2][32][192.168.2.202]/24
                            ::                                  32768 ?
 *>   [2][10.1.1.10:2][0][48][A0F84910BCC2][128][2001:DB8:202::202]/36
                            ::                                  32768 ?
Route Distinguisher: 10.2.2.20:1
 *>i  [2][10.2.2.20:1][0][48][0018736C56C3][0][*]/20
                            10.2.2.20             0   100      0 ?
 *>i  [2][10.2.2.20:1][0][48][0018736C56C3][32][192.168.1.89]/24
                            10.2.2.20             0   100      0 ?
 *>i  [2][10.2.2.20:1][0][48][380E4D9B6A48][32][192.168.1.201]/24
                            10.2.2.20             0   100      0 ?
 *>i  [2][10.2.2.20:1][0][48][380E4D9B6A48][128][2001:DB8:201::201]/36
                            10.2.2.20             0   100      0 ?
Route Distinguisher: 10.2.2.20:2
 *>i  [2][10.2.2.20:2][0][48][0018736C5681][0][*]/20
                            10.2.2.20             0   100      0 ?
 *>i  [2][10.2.2.20:2][0][48][0018736C56C2][0][*]/20
                            10.2.2.20             0   100      0 ?
 *>i  [2][10.2.2.20:2][0][48][0018736C56C2][32][192.168.2.89]/24
                            10.2.2.20             0   100      0 ?
 *>i  [2][10.2.2.20:2][0][48][380E4D9B6A42][32][192.168.2.202]/24
                            10.2.2.20             0   100      0 ?
 *>i  [2][10.2.2.20:2][0][48][380E4D9B6A42][128][2001:DB8:202::202]/36
                            10.2.2.20             0   100      0 ?
Route Distinguisher: 103:2 (default for vrf green)
 *>   [5][103:2][0][24][10.1.13.0]/17
                            0.0.0.0               0        32768 ?
 *>   [5][103:2][0][24][192.168.1.0]/17
                            0.0.0.0               0        32768 ?
 *>   [5][103:2][0][24][192.168.2.0]/17
                            0.0.0.0               0        32768 ?
 *>   [5][103:2][0][64][2001:DB8:201::]/29
                            ::                    0        32768 ?
 *>   [5][103:2][0][64][2001:DB8:202::]/29
                            ::                    0        32768 ?
Route Distinguisher: 104:2
 *>i  [5][104:2][0][24][10.1.14.0]/17
                            10.2.2.20             0   100      0 ?
 *>i  [5][104:2][0][24][192.168.1.0]/17
                            10.2.2.20             0   100      0 ?
 *>i  [5][104:2][0][24][192.168.2.0]/17
                            10.2.2.20             0   100      0 ?
 *>i  [5][104:2][0][64][2001:DB8:201::]/29
                            10.2.2.20             0   100      0 ?
 *>i  [5][104:2][0][64][2001:DB8:202::]/29
                            10.2.2.20             0   100      0 ?
```

VTEP 2

The following example shows the output for the **show bgp l2vpn evpn all** command on VTEP 2:

```
VTEP2# show bgp l2vpn evpn all
BGP table version is 584, local router ID is 10.2.2.20
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
```

```
                 x best-external, a additional-path, c RIB-compressed,
                 t secondary path, L long-lived-stale,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

      Network          Next Hop         Metric LocPrf Weight Path
Route Distinguisher: 10.1.1.10:1
 *>i  [2][10.1.1.10:1][0][48][0059DC50AE01][0][*]/20
                       10.1.1.10               0    100      0 ?
 *>i  [2][10.1.1.10:1][0][48][0059DC50AE4C][0][*]/20
                       10.1.1.10               0    100      0 ?
 *>i  [2][10.1.1.10:1][0][48][0059DC50AE4C][32][192.168.1.81]/24
                       10.1.1.10               0    100      0 ?
 *>i  [2][10.1.1.10:1][0][48][0059DC50AE4C][128][2001:DB8:201::81]/36
                       10.1.1.10               0    100      0 ?
 *>i  [2][10.1.1.10:1][0][48][0059DC50AE4C][128][FE80::259:DCFF:FE50:AE4C]/36
                       10.1.1.10               0    100      0 ?
 *>i  [2][10.1.1.10:1][0][48][A0F84910BCCC][32][192.168.1.201]/24
                       10.1.1.10               0    100      0 ?
 *>i  [2][10.1.1.10:1][0][48][A0F84910BCCC][128][2001:DB8:201::201]/36
                       10.1.1.10               0    100      0 ?
Route Distinguisher: 10.1.1.10:2
 *>i  [2][10.1.1.10:2][0][48][0059DC50AE01][0][*]/20
                       10.1.1.10               0    100      0 ?
 *>i  [2][10.1.1.10:2][0][48][0059DC50AE42][0][*]/20
                       10.1.1.10               0    100      0 ?
 *>i  [2][10.1.1.10:2][0][48][0059DC50AE42][32][192.168.2.81]/24
                       10.1.1.10               0    100      0 ?
 *>i  [2][10.1.1.10:2][0][48][A0F84910BCC2][32][192.168.2.202]/24
                       10.1.1.10               0    100      0 ?
 *>i  [2][10.1.1.10:2][0][48][A0F84910BCC2][128][2001:DB8:202::202]/36
                       10.1.1.10               0    100      0 ?
Route Distinguisher: 10.2.2.20:1
 *>   [2][10.2.2.20:1][0][48][0018736C56C3][0][*]/20
                       ::                           32768 ?
 *>   [2][10.2.2.20:1][0][48][0018736C56C3][32][192.168.1.89]/24
                       ::                           32768 ?
 *>i  [2][10.2.2.20:1][0][48][0059DC50AE01][0][*]/20
                       10.1.1.10               0    100      0 ?
 *>i  [2][10.2.2.20:1][0][48][0059DC50AE4C][0][*]/20
                       10.1.1.10               0    100      0 ?
 *>i  [2][10.2.2.20:1][0][48][0059DC50AE4C][32][192.168.1.81]/24
                       10.1.1.10               0    100      0 ?
 *>i  [2][10.2.2.20:1][0][48][0059DC50AE4C][128][2001:DB8:201::81]/36
                       10.1.1.10               0    100      0 ?
 *>i  [2][10.2.2.20:1][0][48][0059DC50AE4C][128][FE80::259:DCFF:FE50:AE4C]/36
                       10.1.1.10               0    100      0 ?
 *>   [2][10.2.2.20:1][0][48][380E4D9B6A48][32][192.168.1.201]/24
                       ::                           32768 ?
 *>   [2][10.2.2.20:1][0][48][380E4D9B6A48][128][2001:DB8:201::201]/36
                       ::                           32768 ?
 *>i  [2][10.2.2.20:1][0][48][A0F84910BCCC][32][192.168.1.201]/24
                       10.1.1.10               0    100      0 ?
 *>i  [2][10.2.2.20:1][0][48][A0F84910BCCC][128][2001:DB8:201::201]/36
                       10.1.1.10               0    100      0 ?
Route Distinguisher: 10.2.2.20:2
 *>   [2][10.2.2.20:2][0][48][0018736C5681][0][*]/20
                       ::                           32768 ?
 *>   [2][10.2.2.20:2][0][48][0018736C56C2][0][*]/20
                       ::                           32768 ?
 *>   [2][10.2.2.20:2][0][48][0018736C56C2][32][192.168.2.89]/24
                       ::                           32768 ?
 *>i  [2][10.2.2.20:2][0][48][0059DC50AE01][0][*]/20
                       10.1.1.10               0    100      0 ?
```

```
 *>i  [2][10.2.2.20:2][0][48][0059DC50AE42][0][*]/20
                       10.1.1.10                0    100       0 ?
 *>i  [2][10.2.2.20:2][0][48][0059DC50AE42][32][192.168.2.81]/24
                       10.1.1.10                0    100       0 ?
 *>   [2][10.2.2.20:2][0][48][380E4D9B6A42][32][192.168.2.202]/24
                       ::                            32768 ?
 *>   [2][10.2.2.20:2][0][48][380E4D9B6A42][128][2001:DB8:202::202]/36
                       ::                            32768 ?
 *>i  [2][10.2.2.20:2][0][48][A0F84910BCC2][32][192.168.2.202]/24
                       10.1.1.10                0    100       0 ?
 *>i  [2][10.2.2.20:2][0][48][A0F84910BCC2][128][2001:DB8:202::202]/36
                       10.1.1.10                0    100       0 ?
Route Distinguisher: 103:2
 *>i  [5][103:2][0][24][10.1.13.0]/17
                       10.1.1.10                0    100       0 ?
 *>i  [5][103:2][0][24][192.168.1.0]/17
                       10.1.1.10                0    100       0 ?
 *>i  [5][103:2][0][24][192.168.2.0]/17
                       10.1.1.10                0    100       0 ?
 *>i  [5][103:2][0][64][2001:DB8:201::]/29
                       10.1.1.10                0    100       0 ?
 *>i  [5][103:2][0][64][2001:DB8:202::]/29
                       10.1.1.10                0    100       0 ?
Route Distinguisher: 104:2 (default for vrf green)
 *>   [5][104:2][0][24][10.1.14.0]/17
                       0.0.0.0                  0         32768 ?
 *>   [5][104:2][0][24][192.168.1.0]/17
                       0.0.0.0                  0         32768 ?
 *>   [5][104:2][0][24][192.168.2.0]/17
                       0.0.0.0                  0         32768 ?
 *>   [5][104:2][0][64][2001:DB8:201::]/29
                       ::                       0         32768 ?
 *>   [5][104:2][0][64][2001:DB8:202::]/29
                       ::                       0         32768 ?
```

show ip route vrf green

VTEP 1

The following example shows the output for the **show ip route vrf** *vrf-name* command on VTEP 1:

```
VTEP1# show ip route vrf green
Routing Table: green
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, m - OMP
       n - NAT, Ni - NAT inside, No - NAT outside, Nd - NAT DIA
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       H - NHRP, G - NHRP registered, g - NHRP registration summary
       o - ODR, P - periodic downloaded static route, l - LISP
       a - application route
       + - replicated route, % - next hop override, p - overrides from PfR

Gateway of last resort is not set

      10.0.0.0/8 is variably subnetted, 3 subnets, 2 masks
C        10.1.13.0/24 is directly connected, Loopback13
L        10.1.13.13/32 is directly connected, Loopback13
B        10.1.14.0/24 [200/0] via 10.2.2.20, 01:30:02, Vlan200
      192.168.1.0/24 is variably subnetted, 3 subnets, 2 masks
```

```
C          192.168.1.0/24 is directly connected, Vlan201
B          192.168.1.89/32 [200/0] via 10.2.2.20, 00:04:05, Vlan200
L          192.168.1.201/32 is directly connected, Vlan201
        192.168.2.0/24 is variably subnetted, 3 subnets, 2 masks
C          192.168.2.0/24 is directly connected, Vlan202
B          192.168.2.89/32 [200/0] via 10.2.2.20, 00:04:10, Vlan200
L          192.168.2.202/32 is directly connected, Vlan202
```

VTEP 2

The following example shows the output for the **show ip route vrf** *vrf-name* command on VTEP 2:

```
VTEP2# show ip route vrf green
Routing Table: green
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, m - OMP
       n - NAT, Ni - NAT inside, No - NAT outside, Nd - NAT DIA
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       H - NHRP, G - NHRP registered, g - NHRP registration summary
       o - ODR, P - periodic downloaded static route, l - LISP
       a - application route
       + - replicated route, % - next hop override, p - overrides from PfR

Gateway of last resort is not set

        10.0.0.0/8 is variably subnetted, 3 subnets, 2 masks
B          10.1.13.0/24 [200/0] via 10.1.1.10, 01:31:17, Vlan200
C          10.1.14.0/24 is directly connected, Loopback14
L          10.1.14.14/32 is directly connected, Loopback14
        192.168.1.0/24 is variably subnetted, 3 subnets, 2 masks
C          192.168.1.0/24 is directly connected, Vlan201
B          192.168.1.81/32 [200/0] via 10.1.1.10, 01:39:53, Vlan200
L          192.168.1.201/32 is directly connected, Vlan201
        192.168.2.0/24 is variably subnetted, 3 subnets, 2 masks
C          192.168.2.0/24 is directly connected, Vlan202
B          192.168.2.81/32 [200/0] via 10.1.1.10, 01:39:30, Vlan200
L          192.168.2.202/32 is directly connected, Vlan202
```

show platform software fed switch active matm mactable vlan

VTEP 1

The following examples show the output for the **show platform software fed switch active matm mactable vlan** *vlan-id* command on VTEP 1:

Note The MAC address of the peer's core SVI interface must be present in the core VLAN.

```
VTEP1# show platform software fed switch active matm macTable vlan 200
VLAN   MAC                Type  Seq#   EC_Bi  Flags  machandle        siHandle
       riHandle          diHandle           *a_time *e_time  ports

---------------------------------------------------------------------------------------
200    a0f8.4910.bce2     0x8002    0  19880    64  0x7f5d8503fd48   0x7f5d852b6d28
```

```
        0x0                  0x5234                    0         0  Vlan200

200    380e.4d9b.6a4a   0x1000001    0    0    64  0x7f5d855bfaa8    0x7f5d852aca68
       0x7f5d851c7078   0x0                            0         0  RLOC 10.2.2.20 adj_id 126


Total Mac number of addresses:: 2


VTEP1# show platform software fed switch active matm macTable vlan 201
VLAN   MAC               Type Seq#   EC_Bi  Flags  machandle         siHandle
       riHandle          diHandle           *a_time  *e_time  ports
_____
201    00aa.00bb.00cc   0x8002    0  42949   64  0x7f5d85007b88    0x7f5d852b6d28
       0x0               0x0                    0         0  Vlan201

201    0059.dc50.ae01     0x1    9      0    0  0x7f5d852abaf8    0x7f5d85035248
       0x0               0x7f5d8517eae8       300         9  GigabitEthernet1/0/1

201    a0f8.4910.bccc   0x8002    0  19880   64  0x7f5d852ad618    0x7f5d852b6d28
       0x0               0x5234                 0         9  Vlan201

201    0059.dc50.ae4c     0x1   16      0    0  0x7f5d855b3ff8    0x7f5d855a2858
       0x0               0x7f5d8517eae8       300        95  GigabitEthernet1/0/1

201    380e.4d9b.6a48   0x8002    0      0   64  0x7f5d84fbf948    0x7f5d852b6d28
       0x0               0x5234                 0        95  Vlan201

201    0018.736c.56c3   0x1000001  0    0   64  0x7f5d855c8268    0x7f5d852368b8
       0x7f5d855c6098   0x0                    0        95  RLOC 10.2.2.20 adj_id 36


Total Mac number of addresses:: 6


VTEP1# show platform software fed switch active matm macTable vlan 202
VLAN   MAC               Type Seq#   EC_Bi  Flags  machandle         siHandle
       riHandle          diHandle           *a_time  *e_time  ports
_____
202    a0f8.4910.bcc2   0x8002    0  19880   64  0x7f5d8503d288    0x7f5d852b6d28
       0x0               0x0                    0         0  Vlan202

202    0059.dc50.ae01     0x1   10      0    0  0x7f5d852ac8b8    0x7f5d852ac668
       0x0               0x7f5d8517eae8       300        15  GigabitEthernet1/0/1

202    0018.736c.5681   0x1000001  0    0   64  0x7f5d855ba7a8    0x7f5d855b0c58
       0x7f5d8518dea8   0x0                    0        15  RLOC 10.2.2.20 adj_id 125

202    0059.dc50.ae42     0x1   17      0    0  0x7f5d8518e848    0x7f5d855a5258
       0x0               0x7f5d8517eae8       300       225  GigabitEthernet1/0/1

202    380e.4d9b.6a42   0x8002    0      0   64  0x7f5d855a59a8    0x7f5d852b6d28
       0x0               0x5234                 0       225  Vlan202

202    0018.736c.56c2   0x1000001  0    0   64  0x7f5d8523d2b8    0x7f5d855b0c58
       0x7f5d8518dea8   0x0                    0       225  RLOC 10.2.2.20 adj_id 125


Total Mac number of addresses:: 6
```

VTEP 2

The following examples show the output for the **show platform software fed switch active matm mactable vlan** *vlan-id* command on VTEP 2:

Note The MAC address of the peer's core SVI interface must be present in the core VLAN.

```
VTEP2# show platform software fed switch active matm macTable vlan 200
VLAN   MAC                 Type   Seq#   EC_Bi  Flags  machandle         siHandle
       riHandle            diHandle         *a_time  *e_time  ports
_____

200    380e.4d9b.6a4a      0x8002    0    128     64   0x7fa88557f3a8    0x7fa885574e38
       0x0                 0x5174               0       0  Vlan200

200    a0f8.4910.bce2   0x1000001    0      0     64   0x7fa8859a3d38    0x7fa885947ba8
       0x7fa88598bfb8      0x0                  0       0  RLOC 10.1.1.10 adj_id 155


Total Mac number of addresses:: 2


VTEP2# show platform software fed switch active matm macTable vlan 201
VLAN   MAC                 Type   Seq#   EC_Bi  Flags  machandle         siHandle
       riHandle            diHandle         *a_time  *e_time  ports
_____

201    380e.4d9b.6a48      0x8002    0  42949     64   0x7fa885970018    0x7fa885574e38
       0x0                 0x5174               0       0  Vlan201

201    0059.dc50.ae01   0x1000001    0      0     64   0x7fa8849e1be8    0x7fa88598da48
       0x7fa88598e1f8      0x0                  0       0  RLOC 10.1.1.10 adj_id 153

201    0059.dc50.ae4c   0x1000001    0      0     64   0x7fa885993e68    0x7fa88598da48
       0x7fa88598e1f8      0x0                  0       0  RLOC 10.1.1.10 adj_id 153

201    a0f8.4910.bccc      0x8002    0      0     64   0x7fa8859acc48    0x7fa885574e38
       0x0                 0x5174               0       0  Vlan201

201    0018.736c.56c3        0x1   68      0      0   0x7fa8859d3908    0x7fa88599e108
       0x0                 0x7fa884f079d8     300     247  GigabitEthernet1/0/1


Total Mac number of addresses:: 5


VTEP2# show platform software fed switch active matm macTable vlan 202
VLAN   MAC                 Type   Seq#   EC_Bi  Flags  machandle         siHandle
       riHandle            diHandle         *a_time  *e_time  ports
_____

202    380e.4d9b.6a42      0x8002    0  19018     64   0x7fa885994cd8    0x7fa885574e38
       0x0                 0x5174               0       0  Vlan202

202    0018.736c.5681        0x1    9      0      0   0x7fa88599c4e8    0x7fa88599c218
       0x0                 0x7fa884f079d8     300       7  GigabitEthernet1/0/1

202    0059.dc50.ae01   0x1000001    0      0     64   0x7fa8859a3098    0x7fa8859a2dc8
       0x7fa88599ee48      0x0                  0       7  RLOC 10.1.1.10 adj_id 154
```

```
202    0059.dc50.ae42    0x1000001      0       0      64  0x7fa8849e6b78      0x7fa8859a2dc8
       0x7fa88599ee48      0x0                             0       7  RLOC 10.1.1.10 adj_id 154

202    a0f8.4910.bcc2      0x8002       0       0      64  0x7fa88594ddb8      0x7fa885574e38
       0x0                 0x5174                          0       7  Vlan202

202    0018.736c.56c2        0x1       67       0       0  0x7fa8859d3488      0x7fa8859834f8
       0x0                 0x7fa884f079d8                300     267  GigabitEthernet1/0/1


Total Mac number of addresses:: 6
```

Verifying EVPN VXLAN Integrated Routing and Bridging

The following sections provide information about how to verify EVPN VXLAN integrated routing and bridging:

Verifying EVPN VXLAN Layer 2 Overlay Network

See Verifying EVPN VXLAN Layer 2 Overlay Network for the list of commands.

Verifying EVPN VXLAN Layer 3 Overlay Network

See Verifying EVPN VXLAN Layer 3 Overlay Network for the list of commands.

Verifying Distributed Anycast Gateway

The following table lists the **show** commands that are used to verify distributed anycast gateway:

Table 2: Commands to Verify Distributed Anycast Gateway

Command	Purpose
show l2vpn evpn default-gateway	Displays the default gateway database.
show l2vpn l2route default-gateway	Displays the list of sent or received default gateway routes.
show mac address-table	Displays the list of MAC addresses received in default gateway routes that are installed as static MAC addresses for an SVI interface.