Configuring VXLAN BGP EVPN

The following sections provide information about configuring VXLAN BGP EVPN:

Guidelines and Limitations for VXLAN BGP EVPN

The following are the limitations for Virtual Extensible LAN (VXLAN) Border Gateway Protocol (BGP) Ethernet VPN (EVPN) has the following:

- Multicast over VXLAN is currently not supported.
- show commands with the keyword **internal** are not supported.
- For EBGP, it is recommended to use a single overlay EBGP EVPN session between loopbacks.
- Bind NVE to a loopback address that is separate from other loopback addresses that are required by Layer 3 protocols. A best practice is to use a dedicated loopback address for VXLAN.
- VXLAN BGP EVPN does not support an NVE interface in a non-default VRF.
- It is recommended to configure a single BGP session over the loopback for an overlay BGP session.
- The VXLAN UDP port number is used for VXLAN encapsulation. It complies with IETF standards and is not configurable.
- VXLAN BGP EVPN currently supports only leaf switch functionality. Spine switch functionality is not supported.
- Support is not available for any integrated underlay technologies such as route-reflector, or anycast rendezvous point , or Multicast Source Discovery Protocol (MSDP) endezvous point.
- Border leaf functionality and interworking between BGP EVPN and traditional Layer 3 and Layer 2 overlay networks are not supported.
- Auto route-distinguisher and auto route-target for IP VRF is not supported

- Centralized Gateway for Layer 2 VXLAN network identifier (L2VNI) is not supported.
- BGP EVPN Network Virtualization Overlay MIB is not supported.
- 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.

Information About VXLAN BGP EVPN

VXLAN is a MAC in IP/UDP overlay that allows layer 2 segments to be stretched across an IP core. All the benefits of layer 3 topologies are thereby available with VXLAN. The encapsulation and decapsulation of VXLAN headers is handled by a functionality embedded in VXLAN Tunnel End Points (VTEPs). VTEPs themselves could be implemented in software or a hardware form-factor.

VXLAN natively operates on a flood-n-learn mechanism where BU (Broadcast, Unknown Unicast) traffic and Layer 2 Multicast traffic in a given VXLAN network is sent over the IP core to every VTEP that has membership in that network. IP multicast is used to send traffic over the network.The receiving VTEPs decapsulate the packet, and based on the inner frame perform layer-2 MAC learning. The inner SMAC is learnt against the outer Source IP Address (SIP) corresponding to the source VTEP. In this way, reverse traffic can be unicasted toward the previously learnt end host.

Motivations for using an overlay architecture include:

- Scalability — VXLAN provides Layer-2 connectivity that allows the infrastructure that can scale to 16 million tenant networks. It overcomes the 4094-segment limitation of VLANs. This is necessary to address today's multi-tenant cloud requirements.
- Flexibility — VXLAN allows workloads to be placed anywhere, along with the traffic separation required in a multi-tenant environment. The traffic separation is done using network segmentation (segment IDs or virtual network identifiers [VNIs]).Workloads for a tenant can be distributed across different physical devices (since workloads are added as the need arises, into available server space) but the workloads are identified by the same layer 2 or layer 3 VNI as the case may be.
- Mobility — VMs can be moved from one data center location to another without updating spine switch tables. This is because entities within the same tenant network in a VXLAN/EVPN fabric setup retain the same segment ID, regardless of their location.

One of the biggest limitations of VXLAN flood-n-learn is the inherent flooding that is required ensuring that learning happens at the VTEPs. In a traditional deployment, a layer-2 segment is represented with a VLAN that comprises a broadcast domain, which also scopes BU traffic. With VXLAN, now the layer-2 segment spans a much larger boundary across an IP core where floods are translated to IP multicast (or HER). Consequently, the flood-n-learn based scheme presents serious scale challenges especially as the number of end hosts go up. This is addressed via learning using a control-plane for distribution of end host addresses. The control plane of choice is BGP EVPN.

Considerations for VXLAN BGP EVPN deployment

The following considerations need to be taken into account for VXLAN BGP EVPN deployment:

- A loopback address is required when using the source-interface config command. The loopback address represents the local VTEP IP.
- To establish IP multicast routing in the core, IP multicast configuration, PIM configuration, and RP configuration are required.
- VTEP to VTEP unicast reachability can be configured through any IGP/BGP protocol.
- If the anycast gateway feature is enabled for a specific VNI, then the anycast gateway feature must be enabled on all VTEPs that have that VNI configured. Having the anycast gateway feature configured on only some of the VTEPs enabled for a specific VNI is not supported.
- It is a requirement when changing the primary or secondary IP address of the NVE source interfaces to shut the NVE interface before changing the IP address.
- As a best practice, the RP for the multicast group should be configured only on the spine layer. Usethe anycast RP for RP load balancing and redundancy.
- Every tenant VRF needs a VRF overlay, VLAN and SVI for VXLAN routing.
- The following considerations need to be taken into account with eBGP use case:
    - Manual configuration of the Route Targets (RT) is required. RT must be matching between the VTEPs for a given EVPN instance (EVI).
    - The **retain route-target all** BGP knob must be enabled on the Spine nodes under BGP routing process
    - The **set ip next-hop unchanged** BGP knob must be enabled on Spine nodes to set next hop for EVPN routes to the proper VTEP node.
    - Peering between VTEPs can be achieved to multiple Spine nodes to achieve redundancy.
- Ensure the following to create a proper VLAN database:
    - The route targets with eBGP EVPN VxLAN design model cannot be auto generated like in iBGP/IGP model, hence they need to be manually configured for each EVPN instance (EVI) and should be matching for a given EVI. Failure to manually configure route target will result in loss of connectivity and improper operation due to routes not being installed.
    - To ensure proper operation of EVPN VXLAN, assign the vlan first as an access interface to create the vlan and store it in the vlan.dat file. For a trunk interface, trying to create a SVI before creating the vlan in VLAN.dat will put the SVI in a down state.
- In case of a scoped configuration, not all L2 VNIs need to be enabled on all VTEP switches. They will only be enabled as needed on a given VTEP.
- Route Distinguishers (RD) need to be unique per IP VRF (L3 VNI). Route Targets (RT) must match for a given IP VRF (L3 VNI) . There is no auto-generation neither for RD or RT for the case of IP VRF (L3 VNI).
- All VTEP switches need not be configured with same L2 VNIs unless in the scoped configuration. Access VLANs are the VLANs connected to hosts. Access SVIs must have an IP address with the same subnet as the hosts the VLAN is connected to. For AnyCast Gateway support, Access SVIs of the same VLAN should have the same IP and MAC addresses in all VTEPs.

- It is important to configure additional L3 VNIs on all VTEP nodes where Inter-VxLAN communication is needed.

Network considerations for VXLAN deployments

The following network consideration need to be taken into account for VXLAN deployments:

MTU Size in the Transport Network

Due to the MAC-to-UDP encapsulation, VXLAN introduces 50-byte overhead to the original frames. Therefore, the maximum transmission unit (MTU) in the transport network needs to be increased by50 bytes. If the overlays use a 1500-byteMTU, the transport network needs to be configured to accommodate 1550-byte packets at a minimum. Jumbo-frame support in the transport network is required if the overlay applications tend to use larger frame sizes than 1500 bytes.

ECMP and LACP Hashing Algorithms in the Transport Network

Switches introduce a level of entropy in the source UDP port for ECMP and LACP hashing in the transport network. As a way to augment this implementation, the transport network uses an ECMP or LACP hashing algorithm that takes the UDP source port as an input for hashing, which achieves the best load-sharing results for VXLANencapsulated traffic.

Multicast Group Scaling

The VXLAN implementation uses multicast tunnels for broadcast, unknown unicast, and multicast traffic forwarding. Ideally, one VXLAN segment mapping to one IP multicast group is the way to provide the optimal multicast forwarding. It is possible, however, to have multiple VXLAN segments share a single IP multicast group in the core network. VXLAN can support up to 16 million logical Layer 2 segments, using the 24-bit VNID field in the header. With one-to-one mapping between VXLAN segments and IP multicast groups, an increase in the number of VXLAN segments causes a parallel increase in the required multicast address space and the amount of forwarding states on the core network devices. At some point, multicast scalability in the transport network can become a concern. In this case, mapping multiple VXLAN segments to a single multicast group can help conserve multicast control plane resources on the core devices and achieve the desired VXLAN scalability. However, this mapping comes at the cost of suboptimal multicast forwarding. Packets forwarded to the multicast group for one tenant are now sent to the VTEPs of other tenants that are sharing the same multicast group. This causes inefficient utilization of multicast data plane resources. Therefore, this solution is a trade-off between control plane scalability and data plane efficiency.

Despite the suboptimal multicast replication and forwarding, having multiple-tenant VXLAN networks to share a multicast group does not bring any implications to the Layer 2 isolation between the tenant networks. After receiving an encapsulated packet fromthe multicast group, a VTEP checks and validates the VNID in the VXLAN header of the packet. The VTEP discards the packet if the VNID is unknown to it. Only when the VNID matches one of the VTEP's local VXLAN VNIDs, does it forward the packet to that VXLAN segment.Other tenant networks will not receive the packet. Thus, the segregation between VXLAN segments is not compromised.

Considerations for the Transport Network

The following considerations need to be taken into account for the configuration of the transport network:

- On the VTEP device:

- Enable and configure IP multicast.
- Create and configure a loopback interface with a /32 IP address.
- Enable IP multicast on the loopback interface.
- Advertise the loopback interface /32 addresses through the routing protocol (static route) that runs in the transport network.
- Enable IP multicast on the uplink outgoing physical interface.

- Throughout the transport network:
  - Enable and configure IP multicast.

How to Configure VXLAN BGP EVPN

This section provides information about how to configure VXLAN BGP EVPN:

Configuring Bridged VXLAN BGP EVPN using Underlay Multicast

Perform the following set of procedures to configure bridged VXLAN BGP EVPN using underlay multicast:

1. Configure the spine switch for underlay transport between the spine switch and the leaf switches.
2. Configure the leaf switches for underlay transport between the spine switch and the leaf switches.
3. Configure BGP with EVPN address family on the spine switch.
4. Configure BGP with EVPN address family on the leaf switches.
5. Add a VXLAN network identifier node to the Network Virtualization Endpoint (NVE) interface on the leaf switches.
6. Configure L2VPN EVPN on the leaf switches.
7. Configure an EVPN instance on the access-facing VLAN of the leaf switches.

Configuring Underlay Transport (Unicast and Multicast) between the VTEPs and the Spines

Follow these steps to configure underlay transport on the Spine:

Note This configuration is applicable to Cisco Nexus Series Switches and is not applicable to Cisco Catalyst 9000 Family Switches.

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip pim rp-address** *rp-address* **group-list** *prefix* **Example:** `Device(config)# ip pim rp-address 100.1.1.1 group-list 239.0.0.0/8`	Configures a PIM static route processor (RP) address for a multicast group range and specifies a group range for a static RP.
Step 4	**ip pim rp-candidate loopback** *if_number* **group-list** *prefix* **Example:** `Device(config)# ip pim rp-candidate loopback1 group-list 239.0.0.0/8`	Configures a PIM address as a RP candidate. Specifies the loopkback interface. Specifies a group range handled by the RP.
Step 5	**ip pim ssm range** *groups* **Example:** `Device(config)# ip pim ssm range 232.0.0.0/8`	Configures a group range for SSM.
Step 6	**ip pim anycast-rp** *rp-address anycast-rp-peer-address* **Example:** `Device(config)# ip pim anycast-rp 100.1.1.1 10.1.1.1`	Configures PIM Anycast-RP peer for the specified Anycast-RP address.
Step 7	**interface loopback** *number* **Example:** `Device(config)# interface loopback0`	Creates a loopback interface and enters interface configuration mode.
Step 8	**ip address** *ip address* **Example:** `Device(config-if)# ip address 10.1.1.1/32`	Defines the IP address for an interface.
Step 9	**ip pim sparse-mode** **Example:** `Device(config-if)# ip pim sparse-mode`	Enables Protocol Independent Multicast (PIM) sparse mode on an interface.
Step 10	**exit** **Example:** `Device(config-if)# exit`	Exits the interface configuration mode

	Command or Action	Purpose
Step 11	**interface port-channel** *channel-number* **Example:** `Device(config)# interface port-channel1`	Specifies the port-channel interface to configure, and enters the interface configuration mode.
Step 12	**mtu** *bytes* **Example:** `Device(config-if)# mtu 9198`	Sets the interface MTU size.
Step 13	**medium p2p** **Example:** `Device(config-if)# medium p2p`	Configures the interface medium as point to point.
Step 14	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 10.10.1.1/30`	Defines the IP address for an interface.
Step 15	**ip pim sparse-mode** **Example:** `Device(config-if)# ip pim sparse-mode`	Enables Protocol Independent Multicast (PIM) sparse mode on an interface.
Step 16	**exit** **Example:** `Device(config-if)# exit`	Exits the interface configuration mode.

Configuring the VTEP

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	**ip multicast-routing** **Example:** `Device(config)# ip multicast-routing`	Enables IP multicast routing.

	Command or Action	Purpose
Step 4	**ip pim rp-address** *rp-address* **Example:** `Device(config)# ip pim rp-address 100.1.1.1`	Configures a PIM static route processor (RP) address for a multicast group range. The rp address used in this step should be the same one used on the spine.
Step 5	**ip routing** **Example:** `Device(config)# ip routing`	Enables routing on the switch. Even if IP routing was previously enabled, this step ensures that it is activated.
Step 6	**interface loopback** *number* **Example:** `Device(config)# interface Loopback0`	Creates a loopback interface and enters interface configuration mode. This loopback interface is assinged to the NVE interface.
Step 7	**ip address** *ip address* **Example:** `Device(config-if)# ip address 10.11.11.11 255.255.255.255`	Defines the IP address for an interface.
Step 8	**ip pim sparse-mode** **Example:** `Device(config-if)# ip pim sparse-mode`	Enables Protocol Independent Multicast (PIM) sparse mode on an interface.
Step 9	**exit** **Example:** `Device(config-if)# exit`	Exits the interface configuration mode
Step 10	**interface loopback** *number* **Example:** `Device(config)# interface Loopback2`	Creates a loopback interface and enters interface configuration mode. This loopback interface is assinged to the L3 VNI.
Step 11	**ip vrf forwarding** *vrf name* **Example:** `Device(config-if)# vrf forwarding tenant_1`	Associates the VRF with the Layer 3 interface.
Step 12	**ip address** *ip address* **Example:** `Device(config-if)# ip address 11.11.11.11 255.255.255.255`	Defines the IP address for an interface.
Step 13	**exit** **Example:** `Device(config-if)# exit`	Exits the interface configuration mode

	Command or Action	Purpose
Step 14	**interface tengigabitethernet** *slot/port* **Example:** `Device(config)# interface TenGigabitEthernet1/1/2`	Selects the port to configure.
Step 15	**no switchport** **Example:** `Device(config-if)# no switchport`	Makes the interface Layer 3 capable.
Step 16	**no ip address** **Example:** `Device(config-if)# no ip address`	Disables IP processing on a particular interface.
Step 17	**channel-group** *number* **Example:** `Device(config-if)# channel-group 1 mode active`	Assigns and configure a physical interface to an EtherChannel.
Step 18	**exit** **Example:** `Device(config-if)# exit`	Exits the interface configuration mode

Configuring eBGP on the Spine

Follow these steps to configure eBGP with EVPN address family on the Spine:

Note This configuration is applicable to Cisco Nexus Series Switches and is not applicable to Cisco Catalyst 9000 Family Switches.

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	**ip prefix-list** *name* [**seq** *number*] {**permit** | **deny**} *prefix* [**eq** *length*] | [**ge** *length*] | [**le** *length*]]	Creates a prefix list to match IP packets or routes against.

	Command or Action	Purpose
	Example: `Device(config)# ip prefix-list lo_prefix seq 5 permit 0.0.0.0/0 le 32`	
Step 4	**route-map name** {**permit** | **deny**} [*sequence-number*] **Example:** `Device(config)# route-map NH-UNCHANGED permit 10`	Creates the route map entry. Enters route-map configuration mode.
Step 5	**set ip next-hop unchanged** **Example:** `Device(config-route-map)# set ip next-hop unchanged`	Defines the route-map and applies outbound policy for neighbour.
Step 6	**exit** **Example:** `Device(config-route-map)# exit`	Exits the route-map configuration mode
Step 7	**route-map name** {**permit** | **deny**} [**sequence number**] **Example:** `Device(config)# route-map any_prefix permit 10`	Creates the route map entry. Enters route-map configuration mode.
Step 8	**match ip address prefix-list** *name* [*name*] **Example:** `Device(config-route-map)# match ip address prefix-list lo_prefix`	Matches against one or more ip address prefix lists.
Step 9	**exit** **Example:** `Device(config-route-map)# exit`	Exits the route-map configuration mode
Step 10	**router bgp** *number* **Example:** `Device(config)# router bgp 1`	Configures BGP.
Step 11	**router id** {*router id*} **Example:** `Device(config-router)# router-id 10.1.1.1`	Specifies a fixed router ID in the router configuration mode.
Step 12	**bgp log-neighbor-changes** **Example:** `Device(config-router)# log-neighbor-changes`	Enables the generation of logging messages generated when the status of a BGP neighbor changes.

	Command or Action	Purpose
Step 13	**address-family ipv4 unicast** **Example:** `Device(config-router)# address-family ipv4 unicast`	Enters address family configuration mode and Specifies IP Version 4 unicast address prefixes.
Step 14	**redistribute direct** [**route-map** *map-name*] **Example:** `Device(config-router-af)# redistribute direct route-map any_prefix`	Distributes routes that are directly connected on an interface.
Step 15	**exit** **Example:** `Device(config-router-af)# exit`	Exits the address family configuration mode
Step 16	**address-family l2vpn evpn** **Example:** `Device(config-router)# address-family l2vpn evpn`	Specifies the L2VPN address family and enters address family configuration mode. The evpn keyword specifies that EVPN endpoint provisioning information is to be distributed to BGP peers.
Step 17	**nexthop route-map** *name* **Example:** `Device(config-router-af)# nexthop route-map NH-UNCHANGED`	Specifies that Border Gateway Protocol (BGP) routes are resolved using only the next hops that have routes that match specific characteristics.
Step 18	**retain route-target all** **Example:** `Device(config-router-af)# retain route-target all`	Accepts received updates with specified route targets.
Step 19	**exit** **Example:** `Device(config-router-af)# exit`	Exits the address family configuration mode
Step 20	**neighbor** *vtep1 loopback address* **remote-as** *number* **Example:** `Device(config-router)# neighbor 10.11.11.11 remote-as 2`	Adds an entry to the BGP or multiprotocol BGP neighbor table in the router configuration mode.
Step 21	**neighbor** *ip-address* **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 10.11.11.11 update-source loopback0`	Allows BGP sessions to use any operational interface for TCP connections.

	Command or Action	Purpose
Step 22	**neighbor** {*ip address* | *peer-group-name*} **ebgp-multihop** [**ttl**] **Example:** `Device(config-router)# neighbor 10.11.11.11 ebgp-multihop 10`	Allows BGP connections to external peers on networks that are not directly connected.
Step 23	**address-family ipv4 unicast** **Example:** `Device(config-router)# address-family ipv4 unicast`	Enters address family configuration mode and Specifies IP Version 4 unicast address prefixes.
Step 24	**neighbor** {*ip address* | *peer-group-name*} **send-community both** **Example:** `Device(config-router-af)# neighbor 10.11.11.11 send-community both`	Specifies both standard and extended communities attribute should be sent to a BGP neighbour.
Step 25	**soft-reconfiguration inbound** **Example:** `Device(config-router-af)# soft-reconfiguration inbound`	Configures the switch software to start storing BGP peer updates.
Step 26	**exit** **Example:** `Device(config-router-af)# exit`	Exits the address family configuration mode
Step 27	**address-family l2vpn evpn** **Example:** `Device(config-router)# address-family l2vpn evpn`	Specifies the L2VPN address family and enters address family configuration mode. The evpn keyword specifies that EVPN endpoint provisioning information is to be distributed to BGP peers.
Step 28	**neighbor** {*ip address* | *peer-group-name*} **send-community both** **Example:** `Device(config-router-af)# neighbor 10.11.11.11 send-community both`	Specifies both standard and extended communities attribute should be sent to a BGP neighbour.
Step 29	**neighbor** {*ip address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**} **Example:** `Device(config-router-af)# neighbor 10.11.11.11 route-map NH-UNCHANGED out`	Applies the inbound route map to routes received from the specified neighbor, or applies an outbound route map to routes advertised to the specified neighbor.
Step 30	**exit** **Example:** `Device(config-router-af)# exit`	Exits the address family configuration mode

Configuring eBGP on the Leaf Switch

To configure eBGP with EVPN address family on the leaf switch, perform this procedure:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *number* **Example:** `Device(config)# router bgp 2`	Configures BGP.
Step 4	**bgp router-id interface** *loopback-address* **Example:** `Device(config-router)# bgp router-id interface Loopback0`	Specifies loopback address as router address.
Step 5	**bgp log-neighbor-changes** **Example:** `Device(config-router)# bgp log-neighbor-changes`	Enables the generation of logging messages generated when the status of a BGP neighbor changes.
Step 6	**bgp graceful-restart** **Example:** `Device(config-router)# bgp graceful-restart`	Enables the BGP graceful restart capability for a BGP neighbor.
Step 7	**neighbor** *ip-address***remote-as** *number* **Example:** `Device(config-router)# neighbor 10.1.1.1 remote-as 1`	Defines MP-BGP neighbors. Under each neighbor define l2vpn evpn.
Step 8	**neighbor** {*ip address* | *peer-group-name*} **ebgp-multihop** [**ttl**] **Example:** `Device(config-router)# neighbor 10.1.1.1 ebgp-multihop 10`	Allows BGP connections to external peers on networks that are not directly connected.

	Command or Action	Purpose
Step 9	**neighbor** {*ip address* | *group-name*} **update-source** *interface* **Example:** `Device(config-router)# neighbor 10.1.1.1 update-source Loopback0`	Configures update source. Update source can be configured per neighbor or per peer-group
Step 10	**address-family ipv4** **Example:** `Device(config-router)# address-family ipv4`	Enters address family configuration mode.
Step 11	**redistribute connected** **Example:** `Device(config-router-af)# redistribute connected`	Redistributes connected routes from another routing protocol.
Step 12	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 10.1.1.1 activate`	Enables the exchange information from a bgp neighbor
Step 13	**exit** **Example:** `Device(config-router-af)# exit-address-family`	Exits the address family configuration mode
Step 14	**address-family l2vpn evpn** **Example:** `Device(config-router)# address-family l2vpn evpn`	Specifies the L2VPN address family and enters address family configuration mode.
Step 15	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 10.1.1.1 activate`	Enables the exchange information from a bgp neighbor
Step 16	**neighbor** *ip-address* **send-community both** **Example:** `Device(config-router-af)# neighbor 10.1.1.1 send-community both`	Specifies the communities attribute sent to a bgp neighbor
Step 17	**maximum-paths** *number-of-paths* **Example:** `Device(config-router-af)# maximum-paths 2`	Controls the maximum number of parallel routes an IP routing protocol can support.

	Command or Action	Purpose
Step 18	**exit** **Example:** `Device(config-router-af)# exit-address-family`	Exits the address family configuration mode
Step 19	**address-family ipv4 vrf** *vrf-name* **Example:** `Device(config-router)# address-family ipv4 vrf tenant_1`	Specifies the name of the VRF instance to associate with subsequent address family configuration mode commands.
Step 20	**advertise l2vpn evpn** **Example:** `Device(config-router-af)# advertise l2vpn evpn`	Advertises (L2VPN) EVPN routes within a tenant VRF in a VXLAN EVPN fabric.
Step 21	**redistribute connected** **Example:** `Device(config-router-af)# redistribute connected`	Redistributes connected routes from another routing protocol.
Step 22	**exit** **Example:** `Device(config-router-af)# exit-address-family`	Exits the address family configuration mode

Configuring the NVE Interface and VNIs

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* **Example:** `Device(config)# interface nve1`	Configures the NVE interface.
Step 4	**no ip address** **Example:**	Disables IP processing on the interface.

	Command or Action	Purpose
	`Device(config-if)# no ip address`	
Step 5	**source-interface loopback** *number* **Example:** `Device(config-if)# source-interface Loopback1`	Creates a loopback interface. **Note** This interface will be a different loopback from the loopback interface used for underlay.
Step 6	**host-reachability protocol bgp** **Example:** `Device(config-if)# host-reachability protocol bgp`	Defines BGP as the mechanism for host reachability advertisement.
Step 7	**member vni** *vni* **associate-vrf** **Example:** `Device(config-if)# member vni 11001 mcast-group 239.0.1.1`	Adds Layer-3 VNIs, one per tenant VRF, to the overlay. **Note** Required for VXLAN routing only.
Step 8	**member vni** *vni* **mcast-group** *address* **Example:** `Device(config-if)# member vni 900001 vrf tenant_1`	Adds Layer 2 VNIs to the tunnel interface and assigns a multicast group to the VNIs.

Configuring L2VPN EVPN on Leaf Switch

To configure L2VPN EVPN on a leaf switch, perform this procedure:

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	**l2vpn evpn** **Example:** `Device(config)# l2vpn evpn`	Enters L2VPN configuration mode
Step 4	**replication-type** [**static** | **ingress**] **Example:** `Device(config-l2vpn)# replication-type static`	Suppresses use of Inclusive Multicast Ethernet Tag (IMET) routes. IP Multicast is used for BUM traffic.

	Command or Action	Purpose
Step 5	**router-id loopback** *number* **Example:** `Device(config-l2vpn)# router-id Loopback1`	Specifies the interface that will supply the IP addresses to be used in auto-generating route distinguishers.
Step 6	**exit** **Example:** `Device(config-l2vpn)# exit`	Exits the L2VPN configuration.
Step 7	**l2vpn evpn instance** *instance-number* **vlan-based** **Example:** `Device(config)# l2vpn evpn instance 1 vlan-based`	Configures VLAN based EVI in the L2VPN configuration mode. This command is optional if the route targets or the route distinguishers are not needed to be configured manually.
Step 8	**encapsulation vxlan** **Example:** `Device(config-l2vpn)# encapsulation vxlan`	Defines the encapsulation format as VXLAN
Step 9	**route-target export** *route-target-id* **Example:** `Device(config-l2vpn)# route-target export 2:1`	Configures BGP route exchange.
Step 10	**route-target import** *route-target-id* **Example:** `Device(config-l2vpn)# route-target import 2:1`	Configures BGP route exchange.
Step 11	**no auto-route-target** **Example:** `Device(config-l2vpn)# no auto-route-target`	Removes the automatically generated route-targets.
Step 12	**exit** **Example:** `Device(config-l2vpn)# exit`	Exits the L2VPN configuration.
Step 13	**vlan configuration** *vlan-id* **Example:** `Device(config)# vlan configuration 11`	Enters the vlan feature configuration mode.
Step 14	**member evpn-instance** *evpn-instance-number* **vni** *vni-number* **Example:**	Configures the evpn vxlan vni instance.

	Command or Action	Purpose
	`Device(config-vlan)# member evpn-instance 1 vni 11001`	

Configuring access customer facing VLAN VTEP

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 gigabitethernet** *slot/port* **Example:** `Device(config)# interface GigabitEthernet1/0/11`	Enters the interface configuration mode on the Gigabit Ethernet interface.
Step 4	**switchport access vlan** *vlan-id* **Example:** `Device(config-if)# switchport access vlan 11`	Sets the access VLAN when the interface is in access mode.
Step 5	**switchport mode access** **Example:** `Device(config-if)# switchport mode access`	Sets the interface as a nontrunking nontagged single-VLAN Ethernet interface.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Exits the interface configuration mode.
Step 7	**interface gigabitethernet** *slot/port* **Example:** `Device(config)# interface TenGigabitEthernet1/1/7`	Enters the interface configuration mode on the Gigabit Ethernet interface.
Step 8	**switchport trunk allowed vlan** *vlan_list* **Example:** `Device(config-if)# switchport trunk allowed vlan 11-210,901-905`	Configures the VLAN ids of the allowed VLANs for the interface.

	Command or Action	Purpose
Step 9	**switchport mode trunk** **Example:** `Device(config-if)# switchport mode trunk`	Sets the interface as an Ethernet trunk port.

Configuring IP VRF on Leaf Switch for Inter-VxLAN routing

To configure IP virtual routing and forwarding (VRF) on the leaf switch for inter-VXLAN routing, perform this procedure:

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	**vrf definition** *vrf-name* **Example:** `Device(config)# vrf definition tenant_1`	Configures a VRF routing-table instance and enters VRF configuration mode.
Step 4	**rd** *route-distinguisher* **Example:** `Device(config-vrf)# rd 1:1`	Creates routing and forwarding tables for a VRF.
Step 5	**address-family ipv4** **Example:** `Device(config-vrf)# address-family ipv4`	Enters address family configuration mode.
Step 6	**route-target export** *route-target-id* **Example:** `Device(config-vrf-af)# route-target export 1:1`	Creates a list of export route target communities for the specified VRF.
Step 7	**route-target import** *route-target-id* **Example:** `Device(config-vrf-af)# route-target import 1:1`	Creates a list of import route target communities for the specified VRF.
Step 8	**route-target import** *route-target-id* **stitching** **Example:**	Configures importing of VXLAN route target communities to the VRF.

	Command or Action	Purpose
	`Device(config-vrf-af)# route-target import 1:1 stitching`	
Step 9	**route-target export** *route-target-id* **stitching** **Example:** `Device(config-vrf-af)# route-target export 1:1 stitching`	Configures exporting of VXLAN route target communities from the VRF.
Step 10	**exit-address-family** **Example:** `Device(config-vrf-af)# exit-address-family`	Exits address-family configuration mode.

Verifying the VXLAN BGP EVPN Configuration

Command	Purpose
show nve vni	Displays VNIs associated in the NVE.
show ip mroute	Displays multicast routing table information.
show ip mfib	Displays forwarding entries and interfaces in the IPv4 Multicast Forwarding Information Base (MFIB).
show ip pim neighbors	Displays PIM neighbour table.
show ip pim tunnel	Displays information about the PIM register encapsulation and decapsulation tunnels on an interface.
show ip pim rp	Displays mapping information for the RP.
show l2vpn evpn evi [*evpn-id* | *all*]	Displays detailed information for a particular EVI or all EVIs.
show mac address-table vlan vlan id	Displays information for a specific VLAN.
show l2route evpn mac [**all** | **evi** *vlan-id*]	Displays MAC and IP address information learnt by the switch in the EVPN control plane.
show bgp l2vpn evpn	Displays BGP information for L2VPN-EVPN address family.
show ip vrf *vrf-name*	Displays a summary of all VRFs present on the current router and their associated route-distinguishers and interface(s).
show bgp vpnv4 unicast vrf *vrf-name*	Displays VPNv4 routes from BGP table for a specific vrf.
show ip route vrf *vrf-name*	Displays the IP routing table associated with a specific VRF.

Command	Purpose
show l2vpn evpn mac	Displays the MAC address database for Layer 2 EVPN.
show l2vpn evpn mac ip	Displays the IP address database for Layer 2 EVPN.
show l2route evpn mac ip	Displays MAC IP routes.

Note Although the **show ip bgp** command is available for verifying a BGP configuration, as a best practice, it is preferable to use the**show bgp** command instead.

Configuration Examples for VXLAN BGP EVPN (EBGP)

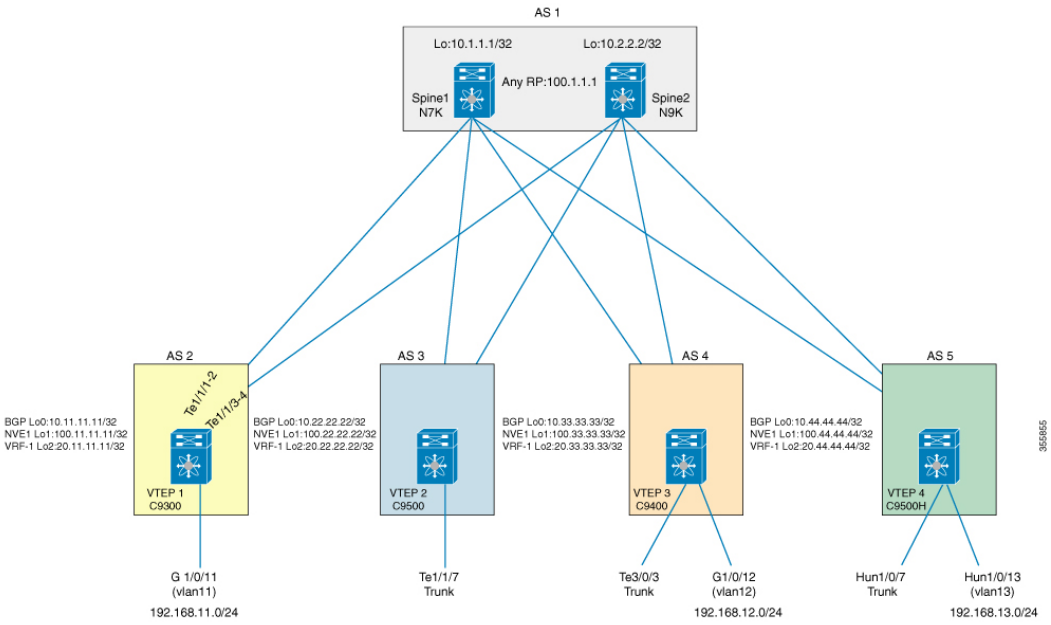
The following sections provide various configuration examples for VXLAN BGP EVPN:

Configuration Examples for VxLAN BGP EVPN in Bridge Mode

The following section provides various examples for VxLAN BGP EVPN in bridge mode:

Example: Configuring eBGP Multi-AS EVPN VxLAN design model

Figure 1: shows the topology used in the eBGP Multi –AS design model



Example: Configuring Underlay Transport (Unicast and Multicast) Between all the VTEPs and the Spine(s):

Example

eBGP peering between the spine and the VTEPs requires IP connectivity. This can be achieved by using static routes to reach loopback addresses between VTEPs and spines.

Configuring the spine

Note The following Spine configuration is applicable to Cisco Nexus Series Switches and is not applicable to Cisco Catalyst 9000 Family Switches.

```
Device(config)# ip pim rp-address 100.1.1.1 group-list 239.0.0.0/8
Device(config)# ip pim rp-candidate loopback1 group-list 239.0.0.0/8
Device(config)# ip pim anycast-rp 100.1.1.1 10.1.1.1
Device(config)# ip pim anycast-rp 100.1.1.1 10.2.2.2
!
Device(config)# interface loopback0
Device(config-if)# ip address 10.1.1.1/32
Device(config-if)# ip pim sparse-mode
!
Device(config)# interface loopback1
Device(config-if)# ip address 100.1.1.1/32
Device(config-if)# ip pim sparse-mode
!
Device(config)# interface port-channel1
Device(config-if)# mtu 9198
Device(config-if)# medium p2p
Device(config-if)# ip address 10.10.1.1/30
Device(config-if)# ip pim sparse-mode
!

Device(config)# interface port-channel2
Device(config-if)# mtu 9198
Device(config-if)# medium p2p
Device(config-if)# ip address 10.10.2.1/30
Device(config-if)# ip pim sparse-mode
!
Device(config)# interface port-channel3
Device(config-if)# mtu 9198
Device(config-if)# medium p2p
Device(config-if)# ip address 10.10.3.1/30
Device(config-if)# ip pim sparse-mode
```

Configuring the VTEP

```
Device(config)# ip multicast-routing
Device(config)# ip pim rp-address 100.1.1.1
!
Device(config)# ip routing
!
Device(config)# interface Loopback0
Device(config-if)# ip address 10.11.11.11 255.255.255.255
Device(config-if)# ip pim sparse-mode
Device(config-if)# exit
!
```

```
Device(config)# interface Loopback1
Device(config-if)# ip address 100.11.11.11 255.255.255.255
Device(config-if)# ip pim sparse-mode
Device(config-if)# exit
!
Device(config)# interface Loopback2
Device(config-if)# vrf forwarding tenant_1
Device(config-if)# ip address 11.11.11.11 255.255.255.255
Device(config-if)# exit
!
Device(config)# interface Port-channel1
Device(config-if)# no switchport
Device(config-if)# ip address 10.10.1.2 255.255.255.252
Device(config-if)# ip pim sparse-mode
Device(config-if)# exit
!
Device(config)# interface Port-channel11
Device(config-if)# no switchport
Device(config-if)# ip address 20.20.1.2 255.255.255.252
Device(config-if)# ip pim sparse-mode
Device(config-if)# exit
!
Device(config)# interface TenGigabitEthernet1/1/2
Device(config-if)# no switchport
Device(config-if)# no ip address
Device(config-if)# channel-group 1 mode active
Device(config-if)# exit
!
Device(config)# interface TenGigabitEthernet1/1/3
Device(config-if)# no switchport
Device(config-if)# no ip address
Device(config-if)# channel-group 11 mode active
```

Example: Configuring eBGP with EVPN Address Family Between the Spine(s) and VTEPs:

Example

The following example shows how to configure the spine

Note The following Spine configuration is applicable to Cisco Nexus Series Switches and is not applicable to Cisco Catalyst 9000 Family Switches.

```
Device(config)# ip prefix-list lo_prefix seq 5 permit 0.0.0.0/0 le 32
Device(config)# route-map NH-UNCHANGED permit 10
Device(config-route-map)# set ip next-hop unchanged
Device(config-route-map)# exit
Device(config)# route-map any_prefix permit 10
Device(config-route-map)# match ip address prefix-list lo_prefix
Device(config-route-map)# exit
!
Device(config)# router bgp 1
Device(config-router)# router-id 10.1.1.1
Device(config-router)# log-neighbor-changes
Device(config-router)# address-family ipv4 unicast
Device(config-router-af)# redistribute direct route-map any_prefix
Device(config-router-af)# exit
```

```
Device(config-router)# address-family l2vpn evpn
Device(config-router-af)# nexthop route-map NH-UNCHANGED
Device(config-router-af)# retain route-target all
Device(config-router-af)# exit
!
Device(config-router)# neighbor 10.11.11.11 remote-as 2
Device(config-router)# neighbor 10.11.11.11 update-source loopback0
Device(config-router)# neighbor 10.11.11.11 ebgp-multihop 10
Device(config-router)# address-family ipv4 unicast
Device(config-router-af)# neighbor 10.11.11.11 send-community both
Device(config-router-af)# soft-reconfiguration inbound
Device(config-router-af)# exit
Device(config-router)# address-family l2vpn evpn
Device(config-router-af)# neighbor 10.11.11.11 send-community both
Device(config-router-af)# neighbor 10.11.11.11 route-map NH-UNCHANGED out
```

The following example shows how to configure the VTEP

```
Device(config)# router bgp 2
Device(config-router)# bgp router-id interface Loopback0
Device(config-router)# bgp log-neighbor-changes
Device(config-router)# bgp graceful-restart
Device(config-router)# neighbor 10.1.1.1 remote-as 1
Device(config-router)# neighbor 10.1.1.1 ebgp-multihop 10
Device(config-router)# neighbor 10.1.1.1 update-source Loopback0
!
Device(config-router)# address-family ipv4
Device(config-router-af)# redistribute connected
Device(config-router-af)# neighbor 10.1.1.1 activate
Device(config-router-af)# exit-address-family
!
Device(config-router)# address-family l2vpn evpn
Device(config-router-af)# neighbor 10.1.1.1 activate
Device(config-router-af)# neighbor 10.1.1.1 send-community both
Device(config-router-af)# maximum-paths 2
Device(config-router-af)# exit-address-family
!
Device(config-router)# address-family ipv4 vrf tenant_1
Device(config-router-af)# advertise l2vpn evpn
Device(config-router-af)# redistribute connected
Device(config-router-af)# exit-address-family
```

Example: Configuring NVE on all VTEPs

Example

The following example shows how to configure the VTEP

```
Device(config)# interface nve1
Device(config-if)# no ip address
Device(config-if)# source-interface Loopback1
Device(config-if)# host-reachability protocol bgp
Device(config-if)# member vni 11001 mcast-group 239.0.1.1
Device(config-if)# member vni 11002 mcast-group 239.0.1.1
Device(config-if)# member vni 900001 vrf tenant_1
```

Example: Configuring L2VPN EVPN on VTEPs

Example

The following example shows how to configure the VTEP

```
Device(config)# l2vpn evpn
Device(config-l2vpn)# replication-type static
Device(config-l2vpn)# router-id Loopback1
!
Device(config)# l2vpn evpn instance 1 vlan-based
Device(config-l2vpn)# encapsulation vxlan
Device(config-l2vpn)# route-target export 2:1
Device(config-l2vpn)# route-target import 2:1
Device(config-l2vpn)# no auto-route-target
!
Device(config)# l2vpn evpn instance 2 vlan-based
Device(config-l2vpn)# encapsulation vxlan
Device(config-l2vpn)# route-target export 2:2
Device(config-l2vpn)# route-target import 2:2
Device(config-l2vpn)# no auto-route-target
```

Example: Configuring Access Customer Facing VLAN VTEPs

Example

The following example shows how to configure the VTEP

```
Device(config)# interface GigabitEthernet1/0/11
Device(config-if)# switchport access vlan 11
Device(config-if)# switchport mode access
!
Device(config)# interface TenGigabitEthernet1/1/7
Device(config-if)# switchport trunk allowed vlan 11-210,901-905
Device(config-if)# switchport mode trunk
```

Example: Configuring Additional VNI, EVI and VLAN on VTEPs

Example

```
Device(config)# vlan 4000
Device(config-vlan)# state active
Device(config)# vlan configuration 4000
Device(config-vlan)# member evpn-instance 20000
```

Example

The following example shows how to configure the VTEP

```
Device(config)# vlan 11
Device(config-vlan)# state active
Device(config)# vlan 12
Device(config-vlan)# state active
Device(config)# vlan 901
Device(config-vlan)# state active
```

```
!
Device(config)# vlan configuration 11
Device(config-vlan)#  member evpn-instance 1 vni 11001
!
Device(config)# vlan configuration 12
Device(config-vlan)# member evpn-instance 2 vni 11002
!

Device(config)# vlan configuration 901
Device(config-vlan)# member vni 900001
!
Device(config)# interface Vlan901
description connected to vni_900001
Device(config-if)# vrf forwarding tenant_1
Device(config-if)# ip unnumbered Loopback2
!
Device(config)# interface nve1
Device(config-if)# no ip address
Device(config-if)# source-interface Loopback1
Device(config-if)# host-reachability protocol bgp
Device(config-if)# member vni 11001 mcast-group 239.0.1.1
Device(config-if)# member vni 11002 mcast-group 239.0.1.1
Device(config-if)# member vni 900001 vrf tenant_1
```

Example: Configuring IP VRF on VTEPs for Inter-VxLAN routing

Example

The following example shows how to configure the VTEP

```
Device(config)# vrf definition tenant_1
Device(config-vrf)# rd 1:1
!
Device(config-vrf)# address-family ipv4
Device(config-vrf-af)# route-target export 1:1
Device(config-vrf-af)# route-target import 1:1
Device(config-vrf-af)# route-target export 1:1 stitching
Device(config-vrf-af)# route-target import 1:1 stitching
Device(config-vrf-af)# exit-address-family
```

Example: Configuring Access VLAN Interfaces (SVIs) on VTEPs

Example

The following example shows how to configure the VTEP

```
Device(config)# interface Vlan11
description vni_11001
mac-address 0001.0001.0001
Device(config-if)# vrf forwarding tenant_1
Device(config-if)# ip address 192.168.1.254 255.255.255.0
Device(config-if)# exit
Device(config)# interface Vlan12
description vni_11002
mac-address 0001.0001.0001
Device(config-if)# vrf forwarding tenant_1
Device(config-if)# ip address 192.168.2.254 255.255.255.0
Device(config-if)# exit
```

Example: Configuring Additional L3-VNI in NVE interfaces

Example

The following example shows how to configure the VTEP

```
Device(config)# interface nve1
Device(config-if)# no ip address
Device(config-if)# source-interface Loopback1
Device(config-if)# host-reachability protocol bgp
Device(config-if)# member vni 11001 mcast-group 239.0.1.1
Device(config-if)# member vni 11002 mcast-group 239.0.1.1
Device(config-if)# member vni 900001 vrf tenant_1
```

Example: Configuring Core-facing VLANs and VLAN Interfaces

Example

The following example shows how to configure the VTEP

```
Device(config)# vlan configuration 901
Device(config-vlan)# member vni 900001
Device(config)# exit
!
Device(config)# interface Vlan901
description connected to vni_900001
Device(config-if)# vrf forwarding tenant_1
Device(config-if)# ip unnumbered Loopback2
```

Example: Configuring iBGP/IGP EVPN VxLAN Design Model

Example

Configuring the spine:

Note The following Spine configuration is applicable to Cisco Nexus Series Switches and is not applicable to Cisco Catalyst 9000 Family Switches.

```
Device(config)# feature-set fabric
Device(config)# hostname spine-1
!
Device(config)# feature telnet
Device(config)# feature scp-server
Device(config)# feature fabric forwarding
Device(config)# nv overlay evpn
Device(config)# feature ospf
Device(config)# feature bgp
Device(config)# feature pim
Device(config)# feature ipp
Device(config)# feature isis
Device(config)# feature fabric multicast
Device(config)# feature interface-vlan
Device(config)# feature lldp
```

```
Device(config)# feature fabric access
Device(config)# feature nv overlay
Device(config)# feature nxapi
!

Device(config)# ip pim rp-address 4.5.4.5 group-list 224.0.0.0/4
!

Device(config)# vlan 1
!

Device(config)# interface Vlan1
!

Device(config)# interface Ethernet1/1 ip address 10.14.1.4/24
Device(config-if)# ip router ospf 1 area 0.0.0.0
Device(config-if)# ip pim sparse-mode
Device(config-if)# no shutdown
!
Device(config)# interface loopback0
Device(config-if)# ip address 4.4.4.4/32
Device(config-if)# ip router ospf 1 area 0.0.0.0
Device(config-if)# ip pim sparse-mode
Device(config-if)# interface loopback1
Device(config-if)# ip address 4.5.4.5/32
Device(config-if)# ip router ospf 1 area 0.0.0.0
Device(config-if)# ip pim sparse-mode
!
Device(config)# router ospf 1
Device(config-router)# router-id 4.4.4.4
!
Device(config)# router bgp 100
Device(config-router)# router-id 4.4.4.4
Device(config-router)# address-family l2vpn evpn
Device(config-router-af)# neighbor 1.1.1.1 remote-as 100
Device(config-router-af)# update-source loopback0
Device(config-router-af)# address-family ipv4 unicast
Device(config-router-af)# send-community both
Device(config-router-af)# route-reflector-client
Device(config-router-af)# address-family l2vpn evpn
Device(config-router-af)# send-community both
Device(config-router-af)# route-reflector-client
```

Configuring the VTEP

```
Device(config)# vrf definition l3vni50000
Device(config-vrf)# rd 101:1
!
Device(config-vrf)# address-family ipv4
Device(config-vrf-af)# route-target export 100:1 stitching
Device(config-vrf-af)# route-target import 100:1 stitching
Device(config)# exit-address-family
!
Device(config)# ip multicast-routing
Device(config)# ip pim rp-address 4.5.4.5
!
Device(config)# l2vpn evpn
Device(config-l2vpn)# replication-type static
Device(config-l2vpn)# exit

!
Device(config)# vlan 10
Device(config-vlan)# State active
```

```
Device(config-vlan)# exit
Device(config)# vlan 11
Device(config-vlan)# State active
Device(config-vlan)# exit
Device(config)# vlan 501
Device(config-vlan)# state active
Device(config-vlan)# exit
!
Device(config)# vlan configuration 10
Device(config-vlan)# member evpn-instance 10 vni 100010
Device(config-vlan)# exit
!
Device(config)# vlan configuration 11
Device(config-vlan)# member evpn-instance 11 vni 100011
Device(config-vlan)# exit
!
Device(config)# vlan configuration 501
Device(config-vlan)# member vni 50000
Device(config-vlan)# exit
!
Device(config)# interface Loopback0
Device(config-if)# ip address 1.1.1.1 255.255.255.255
Device(config-if)# ip pim sparse-mode
Device(config-if)# ip ospf 1 area 0
Device(config-if)# exit
!
Device(config)# interface GigabitEthernet1/0/1
Device(config-if)# switchport mode trunk
Device(config-if)# exit
!
Device(config)# interface GigabitEthernet1/0/2
Device(config-if)# switchport access vlan 10
Device(config-if)# switchport mode access
Device(config-if)# exit
!
Device(config)# interface TenGigabitEthernet3/0/1
description To Spine1
Device(config-if)# no switchport
Device(config-if)# ip address 10.14.1.1 255.255.255.0
Device(config-if))# ip pim sparse-mode
Device(config-if)# ip ospf 1 area 0
Device(config-if)# exit
!
Device(config)# interface TenGigabitEthernet3/0/2
description To Spine1
Device(config-if)# no switchport
Device(config-if)# ip address 10.15.1.1 255.255.255.0
Device(config-if)# ip pim sparse-mode
Device(config-if)# ip ospf 1 area 0
Device(config-if)# exit
!
Device(config)# interface Vlan10
description connected to 100010
Device(config-if)# mac-address 0001.0001.0001
Device(config-if)# vrf forwarding l3vni50000
Device(config-if)# ip address 192.168.10.1 255.255.255.0
Device(config-if)# exit
!
Device(config)# interface Vlan11
description connected to 100011
Device(config-if)# mac-address 0001.0001.0001
Device(config-if)# vrf forwarding l3vni50000
Device(config-if)# ip address 192.168.11.1 255.255.255.0
```

```
Device(config-if)# exit
!
Device(config)# interface Vlan501
description connected to 50000
Device(config-if)# vrf forwarding l3vni50000
Device(config-if)# ip unnumbered Loopback0
Device(config-if)# exit
!
Device(config)# router ospf 1
Device(config-router)# router-id 1.1.1.1
Device(config-router)# nsr
Device(config-router)# exit
!
Device(config)# router bgp 100
Device(config-router)# bgp router-id 1.1.1.1
Device(config-router)# bgp log-neighbor-changes
Device(config-router)# bgp graceful-restart
Device(config-router)# neighbor 4.4.4.4 remote-as 100
Device(config-router)# neighbor 4.4.4.4 update-source Loopback0
!
Device(config-router)# address-family ipv4
Device(config-router-af)# redistribute connected
Device(config-router-af)# neighbor 4.4.4.4 activate
Device(config-router-af)# exit-address-family
!
Device(config-router)# address-family l2vpn evpn
Device(config-router-af)# neighbor 4.4.4.4 activate
Device(config-router-af)# neighbor 4.4.4.4 send-community both
Device(config-router-af))# exit-address-family
!
Device(config)# address-family ipv4 vrf l3vni50000
Device(config-vrf-af)# advertise l2vpn evpn
Device(config-vrf-af)# redistribute connected
Device(config-vrf-af)# exit-address-family
Device(config-vrf)# exit
!
Device(config)# interface nve1
Device(config-if)# no ip address
Device(config-if)# source-interface Loopback0
Device(config-if)# host-reachability protocol bgp
Device(config-if)# member vni 100010 mcast-group 227.0.0.1
Device(config-if)# member vni 100011 mcast-group 227.0.0.1
Device(config-if)# member vni 50000 vrf l3vni50000
```

Example: Verifying L2/L3 VNI in NVE

Example

The following example is a sample output of the **show nve vni** command

```
Device# show nve vni

Interface VNI Multicast-group VNI state Mode VLAN cfg vrf
nve1 60519 233.1.1.19 Up L2CP 519 CLI N/A
nve1 60518 233.1.1.18 Up L2CP 518 CLI N/A
```

Example: Verifying Multicast in Multicast Routing Table

Example

The following example is a sample output of the **show ip mroute** command

```
Device# show ip mroute
IP Multicast Routing Table
Outgoing interface flags: H - Hardware switched, A - Assert winner, p - PIM Join
Timers: Uptime/Expires
Interface state: Interface, Next-Hop or VCD, State/Mode
(*, 239.0.1.1), 5d16h/stopped, RP 100.1.1.1, flags: SJCFx
Incoming interface: Port-channel11, RPF nbr 20.20.1.1
Outgoing interface list:
Tunnel0, Forward/Sparse-Dense, 5d16h/00:01:17
!
(100.11.11.11, 239.0.1.1), 00:02:18/00:00:41, flags: FTx
Incoming interface: Loopback1, RPF nbr 0.0.0.0, Registering
Outgoing interface list:
Port-channel11, Forward/Sparse, 00:02:18/00:03:14
```

Example

The following example is a sample output of the **show ip mfib** command

```
Device# show ip mfib
Forwarding Counts: Pkt Count/Pkts per second/Avg Pkt Size/Kbits per second
Other counts: Total/RPF failed/Other drops
I/O Item Counts: FS Pkt Count/PS Pkt Count
Default
(*,224.0.0.0/4) Flags: C HW
SW Forwarding: 0/0/0/0, Other: 0/0/0
HW Forwarding: 0/0/0/0, Other: 0/0/0
(*,224.0.1.40) Flags: C HW
SW Forwarding: 0/0/0/0, Other: 0/0/0
HW Forwarding: 0/0/0/0, Other: 0/0/0
Port-channel11 Flags: A NS
Loopback0 Flags: F IC NS
Pkts: 0/0
(*,239.0.1.1) Flags: C HW
SW Forwarding: 0/0/0/0, Other: 0/0/0
HW Forwarding: 187/0/190/0, Other: 0/0/0
Port-channel11 Flags: A NS
Tunnel0, VXLAN Decap Flags: F NS
Pkts: 0/0
(100.11.11.11,239.0.1.1) Flags: HW
SW Forwarding: 0/0/0/0, Other: 0/0/0
HW Forwarding: 0/0/0/0, Other: 0/0/0
Null0 Flags: A NS
Port-channel11 Flags: F NS
Pkts: 0/0
Tunnel1 Flags: F
Pkts: 0/0
```

Example

The following example is a sample output of the **show ip pim neighbors** command

```
Device# show ip pim neighbors
PIM Neighbor Table
Mode: B - Bidir Capable, DR - Designated Router, N - Default DR Priority,
P - Proxy Capable, S - State Refresh Capable, G - GenID Capable,
L - DR Load-balancing Capable
Neighbor Interface Uptime/Expires Ver DR
Address Prio/Mode
10.10.1.1 Port-channel1 5d16h/00:01:40 v2 1 / G
20.20.1.1 Port-channel11 5d16h/00:01:20 v2 1 / G
```

Example

The following example is a sample output of the **show ip pim tunnel** command

```
Device# show ip pim tunnel
Tunnel1*
Type : PIM Encap
RP : 100.1.1.1
Source : 20.20.1.2
State : UP
Last event : Created (5d16h)
# sh ip pim rp
Group: 239.0.1.1, RP: 100.1.1.1, uptime 5d16h, expires never
```

Example: Verifying EVPN Instance in EVPN Manager

Example

The following example is a sample output of the **show l2vpn evpn evi** command

```
Device# show l2vpn evpn evi 1 detail
EVPN instance: 1 (VLAN Based)
RD: 100.11.11.11:1 (auto)
Import-RTs: 2:1
Export-RTs: 2:1
Per-EVI Label: none
State: Established
Encapsulation: vxlan
Vlan: 11
Ethernet-Tag: 0
State: Established
Core If: Vlan901
Access If: Vlan11
RMAC: ec1d.8b75.eac8
Core Vlan: 901
L2 VNI: 11001
L3 VNI: 900001
VTEP IP: 100.11.11.11
MCAST IP: 239.0.1.1
VRF: tenant_1
Pseudoports:
TenGigabitEthernet1/1/7 service instance 11
```

Example: Verifying MAC Table

Example

The following example is a sample output of the **show mac address-table vlan** command

```
Device# show mac address-table vlan 11
Mac Address Table
-------------------------------------------
Vlan Mac Address Type Ports
---- ----------- -------- -----
11 0001.0001.0001 STATIC Vl11 ---------□ SVI mac for Anycast Gateway
11 0011.0011.0005 DYNAMIC Te1/1/7--------□ dynamically learned
Total Mac Addresses for this criterion: 2
```

Example: Verifying MAC entries in EVPN Manager

Example

The following example is a sample output of the **show l2vpn evpn mac** command

```
Device# show l2vpn evpn mac
MAC Address EVI VLAN ESI Ether Tag Next Hop
-------------- ----- ----- ------------------------ ---------- ---------------
0011.0011.00c9 1 11 0000.0000.0000.0000.0000 0 Te1/1/7:11
0012.0012.0001 1 11 0000.0000.0000.0000.0000 0 100.22.22.22
0013.0013.0001 1 11 0000.0000.0000.0000.0000 0 100.33.33.33
0014.0014.0001 1 11 0000.0000.0000.0000.0000 0 100.44.44.44
```

Example: Verifying MAC routes in BGP

Example

The following example is a sample output of the **show bgp l2vpn evpn evi** command

```
Device# show bgp l2vpn evpn evi 1
BGP table version is 654847, local router ID is 10.11.11.11
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: 100.11.11.11:1
*> [2][100.11.11.11:1][0][48][0011001100C9][0][*]/20
:: 32768 ?
*> [2][100.11.11.11:1][0][48][001200120001][0][*]/20
100.22.22.22 0 1 3 ?
*> [2][100.11.11.11:1][0][48][001200120001][32][192.168.1.2]/24
100.22.22.22 0 1 3 ?
*> [2][100.11.11.11:1][0][48][001300130001][0][*]/20
100.33.33.33 0 1 4 ?
*> [2][100.11.11.11:1][0][48][001300130001][32][192.168.1.3]/24
100.33.33.33 0 1 4 ?
*> [2][100.11.11.11:1][0][48][001400140001][0][*]/20
100.44.44.44 0 1 4 ?
```

```
*> [2][100.11.11.11:1][0][48][001400140001][32][192.168.1.4]/24
100.44.44.44 0 1 4 ?
```

Example: Verifying MAC routes in Layer 2 Routing Information Base

Example

The following example is a sample output of the **show l2route evpn mac** command

```
Device# show l2route evpn mac
EVI ETag Prod Mac Address Next Hop(s) Seq Number
----- ---------- ----- -------------- ---------------------------- ----------
1 0 BGP 0012.0012.0001 V:11001 100.22.22.22 0
1 0 BGP 0013.0013.0001 V:11001 100.33.33.33 0
1 0 BGP 0014.0014.0001 V:11001 100.44.44.44 0
1 0 L2VPN 0011.0011.00c9 Te1/1/7:11 0
```

Example: Verifying IP VRF with all SVIs

Example

The following example is a sample output of the **show ip vrf** command

```
Device# show ip vrf
Name                              Default RD            Interfaces
Mgmt-vrf                          <not set>             Gi0/0
tenant_1                          1:1                         Lo2
                                                              Vl11
                                                              Vl12
```

Example: Verifying MAC/IP entries in MAC VRFs (EVIs)

Example

The following example is a sample output of the **show bgp l2vpn evpn evi** command

```
Device# show bgp l2vpn evpn evi 1 route-type 2
BGP routing table entry for [2][100.11.11.11:1][0][48][0011001100C9][32][10.0.0.2]/24,
version 7
Paths: (1 available, best #1, table evi_1)
  Advertised to update-groups:
     1
  Refresh Epoch 1
  Local
    :: (via default) from 0.0.0.0 (10.11.11.11)
      Origin incomplete, localpref 100, weight 32768, valid, sourced, local, best
      EVPN ESI: 00000000000000000000, Label1 11001-□ L2 VNI
      Extended Community: RT:2:1 ENCAP:8
      Local irb vxlan vtep:
        vrf:tenant_1, l3-vni:900001--------□ IP VRF and L3 VNI
        local router mac:EC1D.8B75.EAC8
        core-irb interface:Vlan901----□ core SVI
        vtep-ip:100.11.11.11
      rx pathid: 0, tx pathid: 0x0
```

Example: Verifying Remote MAC/IP and IP Prefix routes in L3VNI (IP VRF)

Example

The following example is a sample output of the **show bgp vpnv4 unicast vrf** command

```
Device# show bgp vpnv4 unicast vrf tenant_1
BGP table version is 8583, local router ID is 10.11.11.11
     Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 1:1 (default for vrf tenant_1)
AF-Private Import to Address-Family: L2VPN E-VPN, Pfx Count/Limit: 11/1000
 *>   11.11.11.11/32   0.0.0.0                  0         32768 ?
 *>   11.22.22.22/32   100.22.22.22                           0 1 3 ?
 *>   11.33.33.33/32   100.33.33.33                           0 1 4 ?
 *>   11.44.44.44/32   100.44.44.44                           0 1 4 ?
 *    192.168.1.0         100.44.44.44                           0 1 4 ?
*                               100.33.33.33                           0 1 4 ?
 *                               100.22.22.22                           0 1 3 ?
 *>                               0.0.0.0                  0         32768 ?
 *>   192.168.1.2/32   100.22.22.22                           0 1 3 ?
 *>   192.168.1.3/32   100.33.33.33                           0 1 4 ?
 *>   192.168.1.4/32   100.44.44.44                           0 1 4 ?
 *    192.168.2.0          100.44.44.44                           0 1 4 ?
 *                                 100.33.33.33                           0 1 4 ?
 *                                 100.22.22.22                           0 1 3 ?
 *>                                 0.0.0.0                  0         32768 ?
```

Example: Verifying IP routes are installed in L3 VNI (IP VRF)

Example

The following example is a sample output of the **show ip route vrf** command:

```
Device# show ip route vrf tenant_1
Routing Table: tenant_1
Gateway of last resort is not set

      11.0.0.0/32 is subnetted, 3 subnets
C        11.11.11.11 is directly connected, Loopback2
B        11.22.22.22 [20/0] via 100.22.22.22, 00:13:21, Vlan901
B        11.33.33.33 [20/0] via 100.33.33.33, 00:13:21, Vlan901
B        11.44.44.44 [20/0] via 100.44.44.44, 00:12:51, Vlan901
      192.168.1.0/24 is variably subnetted, 4 subnets, 2 masks
C        192.168.1.0/24 is directly connected, Vlan11
B        192.168.1.3/32 [20/0] via 100.33.33.33, 16:26:48, Vlan901
B        192.168.1.4/32 [20/0] via 100.44.44.44, 2d19h, Vlan901
L        192.168.1.254/32 is directly connected, Vlan11
      192.168.2.0/24 is variably subnetted, 4 subnets, 2 masks
C        192.168.2.0/24 is directly connected, Vlan12
B        192.168.2.3/32 [20/0] via 100.33.33.33, 02:52:20, Vlan901
B        192.168.2.4/32 [20/0] via 100.44.44.44, 2d19h, Vlan901
L        192.168.2.254/32 is directly connected, Vlan12
      192.168.3.0/24 is variably subnetted, 4 subnets, 2 masks
C        192.168.3.0/24 is directly connected, Vlan13
B        192.168.3.3/32 [20/0] via 100.33.33.33, 2d19h, Vlan901
```

Example: Verifying MAC/IP entries in EVPN Manager

Example

The following example is a sample output of the **show l2vpn evpn mac** command:

```
Device# show l2vpn evpn mac ip

IP Address EVI VLAN MAC Address Next Hop(s)
------------------------ ----- ----- -------------- -------------------------
10.0.0.1 1 11 0011.0011.00c9 Te1/1/7:11
10.0.0.2 1 11 0012.0012.0001 100.22.22.22
```

Example: Verifying MAC/IP routes in Layer 2 Routing Information Base

Example

The following example is a sample output of the **show l2route evpn mac** command:

```
Device# show l2route evpn mac ip

EVI ETag Prod Mac Address Host IP Next Hop(s)
----- ---------- ----- -------------- --------------- --------------------------
1 0 BGP 0012.0012.0001 10.0.0.2 V:11001 100.22.22.22
1 0 L2VPN 0011.0011.00c9 10.0.0.1 Te1/1/7:11
```

Feature History and Information for VXLAN BGP EVPN

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Table 1: Feature History for VXLAN BGP EVPN

Feature Name	Release	Feature Information
Configuring VXLAN BGP EVPN	Cisco IOS XE Fuji 16.9.1	VXLAN is a MAC in IP/UDP overlay that allows layer 2 segments to be stretched across an IP core. VXLAN EVPN BGP operates in Bridged and Routed modes. The feature was introduced with IPv4 and IPv6 support in Bridged mode and with IPv4 support in Routed mode.

Feature Name	Release	Feature Information
Configuring VXLAN BGP EVPN	Cisco IOS XE Gibraltar 16.10.1	VXLAN is a MAC in IP/UDP overlay that allows layer 2 segments to be stretched across an IP core. VXLAN EVPN BGP operates in Bridged and Routed modes. This feature was implemented on Cisco Catalyst 9500-High Performance Series Switches.
EVPN VXLAN Ingress Replication	Cisco IOS XE Gibraltar 16.11.1	Enables forwarding of broadcast, unknown unicast, and multicast (BUM) traffic to the relevant recipients in a network. IR is a unicast approach to handling multi-destination traffic, and involves an ingress device replicating every BUM packet and then sending it as a separate unicast to remote egress devices.
VXLAN BGP EVPN Support on Cisco StackWise Virtual	Cisco IOS XE Gibraltar 16.12.1	Support was introduced for VXLAN BGP EVPN feature on switches with Cisco StackWise Virtual configured.