VXLANs

This chapter describes how to identify and resolve problems that might occur when implementing Virtual Extensible Local Area Networks (VXLANs) and includes the following sections:

Information About VXLANs

This section includes the following topics:

Overview

A Virtual Extensible LAN creates LAN segments by using an overlay approach with MAC-in-UDP encapsulation and a 24-bit segment identifier in the form of a VXLAN ID. The encapsulation carries the original Layer 2 frame from the virtual machine (VM) that is encapsulated from within the Virtual Ethernet Module (VEM). Each VEM is assigned an IP address that is used as the source IP address when encapsulated MAC frames are sent over the network. You can have multiple VTEPs per VEM that are used as sources for this encapsulated traffic. The encapsulation carries the VXLAN identifier used to

scope the MAC address of the payload frame. The VXLAN ID to which a VM belongs is indicated within the port profile configuration of the vNIC and is applied when the VM connects to the network. A VXLAN supports three different modes for broadcast, multicast, and MAC distribution mode transport:

- Multicast Mode—A VXLAN uses an IP multicast network to send broadcast, multicast, and unknown unicast flood frames. When a VM joins a VXLAN segment, the server joins a multicast group. Broadcast traffic from the VM is encapsulated and is sent using the multicast outer destination IP address to all the servers in the same multicast group. Subsequent unicast packets are encapsulated and unicast directly to the destination server without a multicast IP address.
- Unicast-only Mode—A VXLAN uses each VEM's single unicast IP address as the destination IP address to send broadcast, multicast, and unknown unicast flood frames. Broadcast traffic from the VM is replicated to each VEM by encapsulating it with a VXLAN header and the designated IP address as the outer destination IP address.
- MAC Distribution Mode (supported only in unicast mode)—In this mode, the unknown unicast flooding is reduced because the Virtual Supervisor Module (VSM) learns all the MAC addresses from the VEMs in all VXLANs and distributes those MAC addresses with VXLAN Tunnel Endpoint (VTEP) IP mappings to other VEMs.

The VXLAN creates LAN segments by using an overlay approach with MAC in IP encapsulation.

VXLAN Tunnel EndPoint

Each VEM requires at least one IP/MAC pair to terminate VXLAN packets. This IP/MAC address pair is known as the VXLAN Tunnel End Point (VTEP) IP/MAC addresses. The VEM supports IPv4 addressing for this purpose. The IP/MAC address that the VTEP uses is configured when you enter the **capability vxlan** command. You can have a maximum of four VTEPs in a single VEM.

One VTEP per VXLAN segment is designated to receive all broadcast, multicast, and unknown unicast flood traffic for the VEM.

When encapsulated traffic is destined to a VEM that is connected to a different subnet, the VEM does not use the VMware host routing table. Instead, the VTEPs initiate the Address Resolution Protocol (ARP) for remote VEM IP addresses. If the VTEPs in the different VEMs are in different subnets, you must configure the upstream router to respond by using the Proxy ARP.

VXLAN Gateway

VXLAN termination (encapsulation and decapsulation) is supported on virtual switches. As a result, the only endpoints that can connect into VXLANs are VMs that are connected to a virtual switch. Physical servers cannot be in VXLANs and routers or services that have traditional VLAN interfaces cannot be used by VXLAN networks. The only way that VXLANs can currently interconnect with traditional VLANs is through VM-based software routers.

Note Starting with Release 5.2(1)SV3(1.15), Cisco Nexus 1000V for VMware vSphere does not support the VXLAN Gateway feature.

VXLAN Trunks

A VXLAN trunk allows you to trunk multiple VXLANs on a single virtual Ethernet interface. To achieve this configuration, you must encapsulate a VXLAN-VLAN mapping on the virtual Ethernet interface.

VXLAN-VLAN mappings are configured through the Virtual Suervisor Module (VSM) and must always be a 1:1 mapping for each Layer 2 domain. VXLAN-VLAN mappings are applied on a virtual Ethernet interface using a port profile. A single port profile can support multiple VLAN-VXLAN mappings.

VXLAN Border Gateway Protocol Control Plane

The Border Gateway Protocol (BGP) control plane enables the Cisco Nexus 1000V to exchange the VXLAN information collected on the VSM-VTEP flood list across VSMs. The Cisco Nexus 1000V supports BGP peering between 16 VSMs to allow VXLAN segments to reach across servers. BGP runs on the VSM and can exchange VXLAN information with the BGP on any other Cisco Nexus 1000V. The Cisco Nexus 1000V can also be used as a route reflector to exchange a VTEP list between VSMs.

This feature extends the unicast-only mode to a multi-VSM environment using a L2VPN EVPN address family. The VTEP information is not exchanged with the VSMs that are running the old version. They will continue to work in multicast mode (VXLAN 1.0) or unicast-only mode in a single Cisco Nexus 1000V (VXLAN 1.5).

BGP Commands

This example shows how to enable BGP:

```
switch# configure terminal
switch(config)# feature bgp
Cisco Nexus 1000V VXLAN Configuration Guide, Release 5.2(1)SV3(1.1)
24 OL-31596-01
Configuring BGP Control Plane
Configuring BGP
switch(config)# interface control0
switch(config-if)# ip address 14.17.199.1/24
switch(config-if)# vrf context default
switch(config-if)# ip route 0.0.0.0/0 14.17.199.254
switch(config-if)# exit
switch(config)# show feature
Feature Name Instance State
---------------------------- -------- --------
bgp
```

This example shows how to enable BGP with the l2vpn evpn address family:

```
switch# configure terminal
switch(config)# router bgp 64496
switch(config-router)# router-id 192.169.67.11
switch(config-router)# address-family l2vpn evpn
switch(config-router-af)# copy running-config startup-config
```

This example shows how to configure a BGP peer:

```
switch# configure terminal
switch(config)# router bgp 64496
switch(config-router)# neighbor 192.0.2.1 remote-as 64497
switch(config-router)# password password1
switch(config-router-neighbor)# description Peer Router B
switch(config-router-neighbor)# address-family l2vpn evpn
```

```
switch(config-router-neighbor-af)# send-community extended
switch(config-router-neighbor-af)# copy running-config startup-config
```

This example shows how to configure a BGP peer-session template and apply it to a BGP peer:

```
switch# configure terminal
switch(config)# router bgp 65536
switch(config-router)# template peer-session BaseSession
switch(config-router-stmp)# timers 30 90
switch(config-router-stmp)# exit
switch(config-router)# neighbor 192.168.1.2 remote-as 65536
switch(config-router-neighbor)# inherit peer-session BaseSession
switch(config-router-neighbor)# description Peer Router A
switch(config-router-neighbor)# copy running-config startup-config
```

This example shows how to configure a BGP peer-policy template and apply it to a BGP peer:

```
switch# configure terminal
switch(config)# router bgp 65536
switch(config-router)# template peer-session BasePolicy
switch(config-router-ptmp)# maximum-prefix 20
switch(config-router-ptmp)# exit
switch(config-router)# neighbor 192.168.1.1 remote-as 65536
switch(config-router-neighbor)# address-family l2vpn evpn
switch(config-router-neighbor-af)# inherit peer-policy BasePolicy
switch(config-router-neighbor-af)# copy running-config startup-config
```

This example shows how to configure a BGP peer template and apply it to a BGP peer:

```
switch# configure terminal
switch(config)# router bgp 65536
switch(config-router)# template peer BasePeer
switch(config-router-neighbor)# inherit peer-session BaseSession
switch(config-router-neighbor-af)# inherit peer-policy BasePolicy 1
switch(config-router-neighbor-af)# exita
switch(config-router-neighbor)# exit
switch(config-router)# neighbor 192.168.1.2 remote-as 65536
switch(config-router-neighbor)# inherit peer BasePeer
switch(config-router-neighbor)# copy running-config startup-config
```

This example shows how to display the BGP sessions:

```
vsm# show bgp session
Total peers 1, established peers 1
ASN 65000
VRF default, local ASN 65000
peers 1, established peers 1, local router-id 1.1.1.1
State: I-Idle, A-Active, O-Open, E-Established, C-Closing, S-Shutdown
Neighbor ASN Flaps LastUpDn|LastRead|LastWrit St Port(L/R) Notif(S/R)
14.17.199.2 65000 0 00:04:05|00:00:04|00:00:04 E 61467/179 0/0
```

This example shows how to display the VTEPs that are learned through the BGP:

```
vsm# show bgp l2vpn evpn
BGP routing table information for VRF default, address family L2VPN EVPN
BGP table version is 10, local router ID is 172.23.181.67
Status: s-suppressed, x-deleted, S-stale, d-dampened, h-history, *-valid, >-best
Path type: i-internal, e-external, c-confed, l-local, a-aggregate, r-redist
Origin codes: i - IGP, e - EGP, ? - incomplete, | - multipath
Network Next Hop Metric LocPrf Weight Path
Route Distinguisher: 172.23.181.67:5000 (EVI 5000) # RD = <Router-id>:<segment-id>
*>l[3]:[5000]:[4]:[192.168.69.3]/88 #Local VTEP 192.168.69.3
0.0.0.0 100 32768 i
```

```
*>i[3]:[5000]:[4]:[192.168.69.104]/88 #VTEP 192.168.69.104 that are learned from peer
172.23.181.68
172.23.181.68 100 0 i
```

This example shows how to display the detailed output for a specific segment ID or RD:

```
vsm# show bgp l2vpn evpn rd 172.23.181.67:5000
BGP routing table information for VRF default, address family L2VPN EVPN
BGP table version is 10, local router ID is 172.23.181.67
Status: s-suppressed, x-deleted, S-stale, d-dampened, h-history, *-valid, >-best
Path type: i-internal, e-external, c-confed, l-local, a-aggregate, r-redist
Origin codes: i - IGP, e - EGP, ? - incomplete, | - multipath
Network Next Hop Metric LocPrf Weight Path
Route Distinguisher: 172.23.181.67:5000 (EVI 5000)
BGP routing table entry for [3]:[5000]:[4]:[192.168.69.3]/88, version 4
Paths: (1 available, best #1)
Flags: (0x00000a) on xmit-list, is not in l2rib/evpn
Path type: local, path is valid, is best path
AS-Path: NONE, path locally originated
0.0.0.0 (metric 0) from 0.0.0.0 (0.0.0.0)
Origin IGP, MED not set, localpref 100, weight 32768
Extcommunity:
RT:1:5000
Advertised to peers:
172.23.181.68
BGP routing table entry for [3]:[5000]:[4]:[192.168.69.104]/88, version 10
Paths: (1 available, best #1)
Flags: (0x00001a) on xmit-list, is in l2rib/evpn
Path type: internal, path is valid, is best path
Imported from 172.23.181.68:5000:[3]:[5000]:[4]:[192.168.69.104]/88
AS-Path: NONE, path sourced internal to AS
172.23.181.68 (metric 0) from 172.23.181.68 (172.23.181.68)
Origin IGP, MED not set, localpref 100, weight 0
Extcommunity:
RT:1:5000
Not advertised to any peer
```

This example shows how to display the BGP convergence time:

```
switch3# show bgp convergence
Global settings:
BGP start time 00:01:06
Config processing completed 00:00:08 after start
BGP out of wait mode 00:00:30 after start

Information for VRF default
Initial-bestpath timeout: 300 sec, configured 0 sec
First peer up 00:00:29 after start
Bestpath timer not running

   IPv4 Unicast:
   First bestpath signalled 00:00:08 after start
   First bestpath completed 00:00:08 after start

   L2VPN EVPN:
   First bestpath signalled 00:00:30 after start
   First bestpath completed 00:00:30 after start
</>
```

This example shows how to display the VTEP list for a specific VXLAN segment ID or all segments:

```
vsm# show bgp l2vpn evpn evi all VTEP
BGP routing table information for VRF default, address family L2VPN EVPN
BGP table version is 17, local router ID is 192.168.66.10
Status: s-suppressed, x-deleted, S-stale, d-dampened, h-history, *-valid, >-best
```

```
Path type: i-internal, e-external, c-confed, l-local, a-aggregate, r-redist
Origin codes: i - IGP, e - EGP, ? - incomplete, | - multipath
Network Next Hop Metric LocPrf Weight Path
Route Distinguisher: 192.168.66.10:5000 (EVI 5000)
*>i66.100.0.1 192.168.66.100 100 0 i
*>l192.168.69.101 0.0.0.0 100 32768 i
*>i192.168.69.201 192.168.67.10 100 0 i
```

This example shows how to display the VTEP list for a specific VXLAN segment ID or all segments:

```
BGP routing table information for VRF default, address family L2VPN EVPN
Route Distinguisher: 192.168.66.10:5000 (EVI 5000)
BGP routing table entry for [3]:[5000]:[4]:[192.168.69.101]/88, version 2
Paths: (1 available, best #1)
Flags: (0x00000a) on xmit-list, is not in l2rib/evpn
Path type: local, path is valid, is best path
AS-Path: NONE, path locally originated
0.0.0.0 (metric 0) from 0.0.0.0 (0.0.0.0)
Origin IGP, MED not set, localpref 100, weight 32768
Extcommunity:
RT:1:5000
Advertised to peers:
192.168.66.100 192.168.67.10
```

This example shows how to display the bridge domain-to-VTEP mappings that are maintained by the VSM and are pushed to all VEMs:

```
switch# show bridge-domain VTEPs
D: Designated VTEP I:Forwarding Publish Incapable VTEP
Note: (*) Denotes active gateway module
Bridge-domain: vxlan-5000
VTEP Table Version: 13
Port Module VTEP-IP Address VTEP-Flags
------------------------------------------------------------------------------
Veth5 3 192.168.69.101 (D)
Remote - 66.100.0.1 (DI)
Remote - 192.168.69.201 (DI)
```

This example shows how to display the BGP evpn summary:

```
switch# show bgp l2vpn evpn neighbors 192.168.65.10
BGP summary information for VRF default, address family L2VPN EVPN
BGP router identifier 192.168.67.10, local AS number 1
BGP table version is 6, L2VPN EVPN config peers 2, capable peers 1
3 network entries and 3 paths using 348 bytes of memory
BGP attribute entries [2/264], BGP AS path entries [0/0]
BGP community entries [0/0], BGP clusterlist entries [0/0]
Neighbor V AS MsgRcvd MsgSent TblVer InQ OutQ Up/Down State/PfxRcd
192.168.65.10 4 1 4011 4013 6 0 0 2d18h 1
```

This example shows how to display the detailed state for a neighbor:

```
switch# show bgp l2vpn evpn neighbors 192.168.65.10
BGP neighbor is 192.168.65.10, remote AS 1, ibgp link, Peer index 1
Inherits peer configuration from peer-template vxlan
BGP version 4, remote router ID 192.168.65.10
BGP state = Established, up for 2d18h
TCP MD5 authentication is enabled
Last read 00:00:24, hold time = 180, keepalive interval is 60 seconds
Last written 00:00:59, keepalive timer expiry due 0.819374
Received 4006 messages, 0 notifications, 0 bytes in queue
Sent 4008 messages, 0 notifications, 0 bytes in queue
Connections established 1, dropped 0
```

```
Last reset by us 2d18h, due to session closed
Last reset by peer never, due to No error

Neighbor capabilities:
Dynamic capability: advertised (mp, refresh, gr) received (mp, refresh, gr)
Dynamic capability (old): advertised received
Route refresh capability (new): advertised received
Route refresh capability (old): advertised received
4-Byte AS capability: advertised received
Address family L2VPN EVPN: advertised received
Graceful Restart capability: advertised received

Graceful Restart Parameters:
Address families advertised to peer:
L2VPN EVPN
Address families received from peer:
L2VPN EVPN
Forwarding state preserved by peer for:
Restart time advertised to peer: 120 seconds
Stale time for routes advertised by peer: 300 seconds
Restart time advertised by peer: 120 seconds

Message statistics:
Sent Rcvd
Opens: 2 1
Notifications: 0 0
Updates: 3 2
Keepalives: 4003 4003
Route Refresh: 0 0
Capability: 0 0
Total: 4008 4006
Total bytes: 76196 76167
Bytes in queue: 0 0

For address family: L2VPN EVPN
BGP table version 6, neighbor version 6
1 accepted paths consume 60 bytes of memory
1 sent paths
Extended community attribute sent to this neighbor
Third-party Nexthop will not be computed.

Local host: 192.168.67.10, Local port: 179
Foreign host: 192.168.65.10, Foreign port: 58283
fd = 39
```

This example shows how to display the detailed state for a VXLAN segment (5000 in this case)

```
switch# show bgp internal evi 5000
BGP L2VPN/EVPN RD Information for 192.168.67.10:5000
VNI ID : 5000 (evi_5000)
#Prefixes Local/BRIB : 1 / 2
BGP EVI Information for evi_5000
EVI ID : 5000 (evi_5000)
RD : 192.168.67.10:5000
Prefixes (local/total) : 1/2
Delete pending : 0
Import pending : 0
Import in progress : 0
Export RTs : 1
Export RT list : 1:5000
Export RT chg/chg-pending : 0/0
Import RTs : 1
Import RT list : 1:5000
```

```
Import RT chg/chg-pending : 0/0
```

A few additional commands are as follows:

- **show bgp event-history msgs**
- **show bgp event-history events**

Multi-MAC Capability

You can use multi-MAC addresses to mark a virtual Ethernet interface as capable of sourcing packets from multiple MAC addresses. For example, you can use this feature if you have a virtual Ethernet port and you have enabled VXLAN trunking on it and the VM that is connected to the port bridges packets that are sourced from multiple MAC addresses.

By using this feature, you can easily identify multi-MAC capable ports and handle live migration scenarios correctly for those ports.

Fragmentation

The VXLAN encapsulation overhead is 50 bytes. To prevent performance degradation due to fragmentation, the entire interconnection infrastructure between all VEMs exchange VXLAN packets must be configured to carry 50 bytes more than what the VM VNICs are configured to send. For example, if the default VNIC configuration is 1500 bytes, you must configure the VEM uplink port profile, upstream physical switch port, interswitch links, and any routers to carry a maximum transmission unit (MTU) of at least 1550 bytes. If that is not possible, we recommend that the MTU within the guest VMs you configure to be smaller by 50 bytes.

If you do not configure a smaller MTU, the VEM attempts to notify the VM if it performs Path MTU (PMTU) Discovery. If the VM does not send packets with a smaller MTU, the VM fragments the IP packets. Fragmentation occurs only at the IP layer. If the VM sends a frame that is too large, the frame is to be dropped after VXLAN encapsulation and if the frame does not contain an IP packet.

Scalability

Maximum Number of VXLANs

The Cisco Nexus 1000V supports a total of 4000 and 6144 bridge domains.

```
VSM-DAOX(config-port-prof-srv)# show resource-availability vlan

Maximum number of user VLANs supported: 4093
Number of user VLANs created : 3968
Total number of available user VLANs : 125
Note: Total number of available user VLANs additionally depend on number of
bridge-domains under usage. Please verify the usage of bridge-domains too.

VSM-DAOX(config-port-prof-srv)# show resource-availability bridge-domain
Maximum number of bridge-domains per DVS: 6144
Number of bridge-domains currently created: 5004
Number of bridge-domains available*: 1140
* available bridge-domains do not account for created VLANs
```

Supported Features

This section includes the following topics:

Jumbo Frames

Jumbo frames are supported by the Cisco Nexus 1000V if there is space on the frame to accommodate the VXLAN encapsulation overhead of at least 50 bytes, and the physical switch/router infrastructure has the capability to transport these jumbo-sized IP packets.

Disabling the VXLAN Feature Globally

As a safety precaution, do not use the no feature segmentation command if there are any ports associated with a VXLAN port profile. You must remove all associations before you can disable this feature. You can use the no feature segmentation command to remove all the VXLAN bridge domain configurations on the Cisco Nexus 1000V.

VXLAN Troubleshooting Commands

Use the following commands to display VXLAN attributes.

This section contains the following topics:

VSM Commands

You can use the commands in this section to troubleshoot problems related to the VSM.

Command	Purpose
show system internal seg_bd info segment 10000	Displays the ports belonging to a specific segment. See Example 25-1 on page 25-10
show system internal seg_bd info port vethernet 1	Displays the vEthernet bridge domain configuration. See Example 25-2 on page 25-10
show system internal seg_bd info port ifindex 0x1c000050	Displays the vEthernet bridge configuration with ifindex as an argument. See Example 25-3 on page 25-10
show system internal seg_bd info port_count	Displays the total number of bridge domain ports. See Example 25-4 on page 25-10

Command	Purpose
show system internal seg_bd info bd vxlan-home	Displays the bridge domain internal configuration See Example 25-5 on page 25-10
show system internal seg_bd info port	Displays the VXLAN vEthernet information. See Example 25-6 on page 25-10

Example 25-1 **show system internal seg_bd info segment 10000**

```
switch(config)# show system internal seg_bd info segment 10000
Bridge-domain: A
Port Count: 11
Veth1
Veth2
Veth3
```

Example 25-2 show system internal seg_bd info port vethernet 1

```
switch(config)# show system internal seg_bd info port vethernet 1
Bridge-domain: A
segment_id = 10000
Group IP: 225.1.1.1
```

Example 25-3 show system internal seg_bd info port ifindex 0x1c000050

```
switch(config)# show system internal seg_bd info port ifindex 0x1c000050
Bridge-domain: A
segment_id = 10000
Group IP: 225.1.1.1
```

Example 25-4 show system internal seg_bd info port_count

```
switch(config)# show system internal seg_bd info port_count
Number of ports: 11
```

Example 25-5 show system internal seg_bd info bd vxlan-home

```
switch(config)# show system internal seg_bd info bd vxlan-home

Bridge-domain vxlan-home (2 ports in all)
Segment ID: 5555 (Manual/Active)
Group IP: 235.5.5.5
State: UP               Mac learning: Enabled
is_bd_created: Yes
current state: SEG_BD_FSM_ST_READY
pending_delete: 0
port_count: 2
action: 4
hwbd: 28
pa_count: 0
Veth2, Veth5
switch(config)#
```

Example 25-6 show system internal seg_bd info port

```
switch# show system internal seg_bd info port
if_index = <0x1c000010>
```

```
Bridge-domain vxlan-pepsi
rid = 216172786878513168
swbd = 4098

if_index = <0x1c000040>
Bridge-domain vxlan-pepsi
rid = 216172786878513216
swbd = 4098

switch#
```

VXLAN Gateway Commands

Note Starting with Release 5.2(1)SV3(1.15), Cisco Nexus 1000V for VMware vSphere does not support the VXLAN Gateway feature.

To display VXLAN Gateway information that is attached to the VSM:

```
switch# show module vem
Mod  Ports  Module-Type                      Model               Status
---  -----  -------------------------------  ------------------  ------------
3    7      Virtual Service Module           VXLAN Gateway       ok
```

To display VXLAN Gateway information that is not attached to the VSM:

```
VXLANGW# attach vem
VXLANGW(vem-attach)# ?
  vemcmd     Execute vem command
  vemdpa     Execute vemdpa command
  vemdpalog  Execute vemdpalog command
  vemlog     Execute vemlog command
  vempkt     Execute vempkt command
  vemset     Execute vemset command
switch(vem-attach)#
```

To display VXLAN Gateway mappings:

```
VXGW-switch(vem-attach)# vemcmd show vxlan-gw-mappings
VLAN   Segment   NumProbes     State
--------------------------------------
1821     9001        3         Active
1822     9002        3         Active
Linux(debug)#
Linux(debug)#
Linux(debug)# vemcmd show vxlan
LTL     VSM Port      IP       Seconds since Last   Vem Port
                    Netmask    IGMP Query Received
                    Gateway
(* = IGMP Join Interface/Designated VTEP)
------------------------------------------------------------
 20        Veth7  17.17.19.111        33        vxlannic0         *
                255.255.255.0
                   17.17.19.1
```

To display VXLAN Gateway statistics:

```
switch(vem-attach)# vemcmd show vxlan-stats
```

```
  LTL  Ucast   Mcast/Repl   Ucast   Mcast    Total
       Encaps  Encaps       Decaps  Decaps   Drops
   17    8717          173    8334       0     242
switch(vem-attach)#

switch(vem-attach)# vemcmd show vxlan-stats ltl 17
VXLAN Port Stats for LTL 17
Unicast Encapsulations: 8756
Multicast Encapsulations/HeadEnd Replications: 173
Unicast Decapsulations: 8372
Multicast Decapsulations: 0
IP Pre-fragmentations: 0
TSO Processed Packets: 0
ICMP Pkt Too Big msgs from upstream: 0
ICMP Pkt Too Big msgs sent to VM: 0
Packets generated by Head End Replication: 172
```

To display the VXLAN Gateway packet path:

```
switch(vem-attach)# vemlog show all
```

To display the bridge-domain configuration on the VSM:

```
switch# show bridge-domain
Note - This command is common for both gateway and VEM.

Global Configuration:
Mode: Unicast-only
MAC Distribution: Disable
Note - If you have enabled MAC distribution, the above command will display Enable.
Bridge-domain segment-cisco (3 ports in all)
Segment ID: 9001 (Manual/Active)
Mode: Unicast-only (default)
MAC Distribution: Disable (default)
Group IP: NULL
State: UP               Mac learning: Enabled
Veth2, Veth3, Veth5
```

To display the bridge-domain vsteps on the VSM:

```
switch# show bridge-domain VTEPs

D: Designated VTEP     I:Forwarding Publish Incapable VTEP

Note: (*) Denotes active gateway module

Bridge-domain: bd-1776
VTEP Table Version: 40
Port         Module   VTEP-IP Address   VTEP-Flags
-------------------------------------------------------------------------------
Veth11       4        172.172.0.134     (D)
Veth66       5        172.172.0.145     (D)
Veth67       5        172.172.1.145

-------------------------------------------------------------------------------
Interface       Module  Serv Inst  Vlan  BD-Name
-------------------------------------------------------------------------------
Ethernet8/1     8       1          1821  vxlan-7001
                8       2          1822  vxlan-7002
                8       3          1823  vxlan-7003
                8       4          1824  vxlan-7004
                8       5          1825  vxlan-7005
switch# sh module VTEPs
```

```
D: Designated VTEP     I:Forwarding Publish Incapable VTEP
A: Active VTEP has duplicates     S: Suppressed duplicate VTEP

Note: (*) Denotes active gateway module

Module   Port         VTEP-IP Address   VTEP-Flags
---------------------------------------------------------------------------
3        Veth2        172.172.1.143     (D)
3        Veth3        172.172.2.143
3        Veth4        172.172.3.143
3        Veth9        172.172.5.143
4        Veth11       172.172.0.134     (D)
Veth68       5        172.172.2.145
Veth69       5        172.172.4.145
```

To display the VLAN-VXLAN mappings programmed on the VSM:

```
switch# show bridge-domain mapping
```

To display the interfaces on the VSM:

```
switch# show module VTEP
```

To display the the bridge domain-to-VTEP mappings that are maintained by the VSM and are pushed to all VEMs:

```
switch# show bridge-domain VTEP
```

To displays the MACs learned on the VSM through VEM distribution:

```
switch# show bridge-domain mac

Bridge-domain: segment-cisco
MAC TABLE Version: 1
Note: You can compare with VEM output using the echo show vxlan version-table command.
MAC Address      Module     Port        VTEP-IP Address  VM-IP Address
---------------------------------------------------------------------------
0050.5683.014e   5          Veth5       10.106.199.117   -
0050.5683.0160   4          Veth2       10.106.199.116   -
0050.5683.0161   4          Veth3       10.106.199.116   -
```

To verify the port configuration on the VSM:

```
switch# show int switchport | begin Vethernet2
Name: Vethernet2
  Switchport: Enabled
  Switchport Monitor: Not enabled
  Operational Mode: access
  Access Mode VLAN: 0 (none)
  Access BD name: segment-cisco
```

To verify the VTEP distribution on the VSM:

```
switch# show bridge-domain segment-cisco VTEPs

D: Designated VTEP     I:Forwarding Publish Incapable VTEP

Bridge-domain: segment-cisco
VTEP Table Version: 2
Note: You can compare the VTEP table version with the echo show vxlan version-table on VEM.
Ifindex     Module  VTEP-IP Address
---------------------------------------------------------------------------
Veth4         4     10.106.199.116(D)
Veth1         5     10.106.199.117(D)
```

```
switch#
```

To verify VXLAN mac-distribution:

```
VSM-DAOX# show bridge-domain mac
Bridge-domain: vxlan6227
MAC Table Count: 1 Next Update Count: 0 Last Update Count: 0
MAC Table Version: 1
MAC Address Module Port VTEP-IP Address
--------------------------------------------------------------------------------
0050.5683.3269 8 Veth1502 192.168.10.6

VSM-DAOX# show bridge-domain vxlan6067 mac
Bridge-domain: vxlan6067
MAC Table Count: 77 Next Update Count: 0 Last Update Count: 0
MAC Table Version: 1
MAC Address Module Port VTEP-IP Address
--------------------------------------------------------------------------------
0050.5683.4c88 6 Veth108 192.168.10.13
0050.5691.01d6 3 Veth177 192.168.10.27
0050.5691.0549 3 Veth695 192.168.10.27
```

Additional **show** commands:

```
show platform fwm errors

show platform fwm info (VTEP | trace | error history)

show platform fwm info error history

show platform fwm event-history msgs

show platform fwm info vlan (all|swbd)
```

VEM Commands

To verify VXLAN vEthernet programming:

```
~ # vemcmd show port segments
                          Native  Seg
  LTL   VSM Port  Mode    SegID   State
   50      Veth5   A       5555   FWD
   51      Veth9   A       8888   FWD
~ #
```

To verify VXLAN VTEP programming:

```
~ # vemcmd show vxlan interfaces
LTL           IP       Seconds since Last
                       IGMP Query Received
(* Interface on which IGMP Joins are sent)
-----------------------------------------
 49        10.3.3.3        50         *
 52        10.3.3.6        50
~ #
Use "vemcmd show port vlans" to verify that the VTEPs are in the correct transport VLAN.
```

To verify bridge domain creation on the VEM:

```
~ # vemcmd show bd  bd-name vxlan-home
```

```
BD 31, vdc 1, segment id 5555, segment group IP 235.5.5.5, swbd 4098, 1 ports,
"vxlan-home"
Portlist:
     50  RedHat_VM1.eth0

~ #
```

To verify remote IP learning:

```
~ # vemcmd show l2 bd-name vxlan-home
Bridge domain   31 brtmax 4096, brtcnt 2, timeout 300
Segment ID 5555, swbd 4098, "vxlan-home"
Flags:  P - PVLAN  S - Secure  D - Drop
       Type          MAC Address   LTL   timeout   Flags    PVLAN    Remote IP
    Dynamic   00:50:56:ad:71:4e   305         2                      10.3.3.100
     Static   00:50:56:85:01:5b    50         0                       0.0.0.0

~ #
```

To display statistics:

```
~ # vemcmd show vxlan-stats
  LTL  Ucast   Mcast   Ucast   Mcast    Total
       Encaps  Encaps  Decaps  Decaps   Drops
   49       5   14265       4      15       0
   50       6   14261       4      15     213
   51       1      15       0       0      10
   52       0      11       0       0      15

~ #
```

To display detailed per-port statistics for a VXLAN vEthernet/VTEP:

```
~ # vemcmd show vxlan-stats ltl 51
```

To display detailed per-port-per-bridge domain statistics for a VXLAN VTEP for all bridge domains:

```
~ # vemcmd show vxlan-stats ltl <vxlan_VTEP_ltl> bd-all
```

To display detailed per-port-per-bridge domain statistics for a VXLAN VTEP for a specified bridge domain:

```
~ # vemcmd show vxlan-stats ltl vxlan_VTEP_ltl bd-name bd-name
```

To verify the bridge-domain configuration on the VEM:

```
switch# vemcmd show bd bd-name segment-cisco
Note - Use the module command to check the details of VEM and gateway on the VSM.

BD 26, vdc 1, segment id 9001, segment group IP 0.0.0.0, swbd 4102, 2 ports,
"segment-cisco"
Segment Mode: Unicast
Note: If MAC distribution is enabled, the above command will displays Segment moode as
Unicast MAC distribution
VTEP DSN: 1 , MAC DSN: 1
Note: You can check the VTEP and MAC download sequence numbers using the vemcmd show
vxlan-VTEPs and vemcmd show l2 bd bd-name commands.
Portlist:
     53  RedHat_VM1_112.eth4
     54  RedHat_VM1_112.eth5
~ #
```

To display the MAC address table that shows the MAC addresses delivered by the VSM:

```
switch# vemcmd show l2 bd-name segment-cisco
```

```
Bridge domain   26 brtmax 4096, brtcnt 3, timeout 300
Segment ID 9001, swbd 4102, "segment-cisco"
Flags:  P - PVLAN  S - Secure  D - Drop
       Type         MAC Address   LTL   timeout   Flags    PVLAN    Remote IP    DSN
    SwInsta   00:50:56:83:01:4e   561         0               10.106.199.117   1
     Static   00:50:56:83:01:61    54         0                    0.0.0.0   1
     Static   00:50:56:83:01:60    53         0                    0.0.0.0   1

switch#
```

Displays the port configuration on the VEM:

```
switch# vemcmd show port
  LTL   VSM Port  Admin Link  State  PC-LTL  SGID  Vem Port  Type
   17     Eth4/1     UP   UP    F/B*    561     0    vmnic0
   49              DOWN   UP    BLK       0         RedHat_VM1_112 ethernet7
   50      Veth8   DOWN   UP    BLK       0         RedHat_VM1_112.eth8
   51      Veth4     UP   UP    FWD       0     0       vmk1  VXLAN
   52              DOWN   UP    BLK       0         RedHat_VM1_112.eth6
   53      Veth2     UP   UP    FWD       0         RedHat_VM1_112.eth4
   54      Veth3     UP   UP    FWD       0         RedHat_VM1_112.eth5
  561        Po2     UP   UP    F/B*      0
```

Displays the VTEP distribution on the VEM:

```
switch# vemcmd show vxlan-VTEPs
Bridge-Domain: segment-cisco Segment ID: 9001
Designated Remote VTEP IPs (*=forwarding publish incapable):
10.106.199.117(DSN: 1),
Note: You can compare the download sequence number against the VTEP download sequence
number using the vemcmnd show bd bd-name.
```

Displays if the MAC address table displays the remote IP learning in the segment-cisco bridge domain:

```
switch# vemcmd show l2 bd-name segment-cisco
Note - Use the module command to check the details of VEM and gateway on the VSM.

Bridge domain   26 brtmax 4096, brtcnt 3, timeout 300
Segment ID 9001, swbd 4102, "segment-cisco"
Flags:  P - PVLAN  S - Secure  D - Drop
       Type         MAC Address   LTL   timeout   Flags    PVLAN    Remote IP    DSN
    Dynamic   00:50:56:83:01:4e   561         1               10.106.199.117   0
     Static   00:50:56:83:01:61    54         0                    0.0.0.0   0
     Static   00:50:56:83:01:60    53         0                    0.0.0.0   0
```

To display the VLAN-VXLAN mappings programmed on the VEM:

```
switch# vemcmd show vlan-vxlan mapping
Note - Use the module command to check the details of VEM and gateway on the VSM.
```

To display the multi-MAC capable interfaces on the VEM:

```
Note - Use the module command to check the details of VEM and gateway on the VSM.
switch# vemcmd show multi-mac-capable interfaces
```

VEM Packet Path Debugging

Use the following commands to debug VXLAN traffic from a VM on VEM1 to a VM on VEM2.

- VEM1: Verify that packets are coming into the switch from the segment vEthernet.

  vempkt capture ingress ltl *vxlan_veth*

- VEM1: Verify VXLAN ecapsulation.

```
vemlog debug sflisp all
vemlog debug sfvnsegment all
```

- VEM1: Verify that the remote IP address is learned:

```
vemcmd show l2 bd-name segbdname
```

  If the remote IP is not learned, packets are sent multicast encapsulated.

- VEM1: Verify encapsulated packets go out or a uplink.

  Use the **vemcmd show vxlan-encap ltl** *ltl* command to find out which uplink is being used.

```
vempkt capture egress ltl uplink
```

- VEM1: Look at statistics for any failures.

```
vemcmd show vxlan-stats all
vemcmd show vxlan-stats ltl veth/vxlanVTEP
```

- VEM2: Verify encapsulated packets are arriving on the uplink.

```
vempkt capture ingress ltl uplink
```

- VEM2: Verify VXLAN decapsulation.

```
vemlog debug sflisp all
vemlog debug sfvnsegment all
```

- VEM2: Verify decapsulated packets go out on a VXLAN vEthernet interface.

```
vempkt capture egress ltl vxlan_veth
```

- VEM2: Look at statistics for any failures:

```
vemcmd show vxlan-stats all
vemcmd show vxlan-stats ltl veth/vxlanVTEP
```

Use the following commands to debug the VXLAN packet path:

```
switch# module vem 4 execute vemlog debug vssnet all
switch# module vem 4 execute vemlog debug sfsched all
switch# module vem 4 execute vemlog debug sfport all
switch# module vem 4 execute vemlog debug sflisp all
switch# module vem 4 execute vemlog debug sfvnsegment all
```

Use the following commands to debug the VXLAN packet path from the VSM:

```
switch# module vem 4 execute vemdpalog debug if_bridge_rt all
switch# module vem 4 execute vemdpalog debug sfbd all
switch# module vem 4 execute vemdpalog debug sf_dp_threads all
switch# module vem 4 execute vemdpalog debug sfl2agent all
switch# module vem 4 execute vemlog debug sfporttable all
```

You can view the output for all the above logs by using the **module vem 4 execute vemlog show all** command.

VEM Multicast Debugging

Use the following command to debug VEM multicast.

- IGMP state on the VEM:

```
vemcmd show igmp vxlan_transport_vlan detail
```

Note This command does not show any output for the segment multicast groups. To save multicast table space, segment groups are not tracked by IGMP snooping on the VEM.

- IGMP queries:

Use the **vemcmd show vxlan interfaces** command to verify that IGMP queries are being received.

- IGMP joins from VTEP:

Use the **vempkt capture ingress ltl** *first_vxlan_VTEP_ltl* command to see if the VMware stack is sending joins.

Use the **vempkt capture egress ltl** *uplink_ltl* command to see if the joins are being sent out to the upstream switch.

VXLAN Data Path Debugging

Use the commands listed in this section to troubleshot VXLAN problems.

This section contains the following topics:

vemlog Debugging

To debug the bridge domain setup or configuration, use the following command:

```
vemlog debug sfbd all
```

To debug the port configuration/CBL/vEthernet LTL pinning, use the following command:

```
vemlog debug sfporttable all
```

(for encapsulated/decapsulated setup and decisions)

```
vemlog debug sfvnsegment all
```

To debug for actual packet editing, VXLAN interface handling, and multicast handling, use the following command:

```
vemlog debug sflisp all
```

To debug multicast joins or leaves on the DPA socket, use the following command:

```
echo "debug dpa_allplatform all" > /tmp/dpafifo
```

To debug the bridge domain configuration, use the following command:

```
echo “debug sfl2agent all” > /tmp/dpafifo
```

To debug the port configuration, use the following command:

```
echo "debug sfportagent all" > /tmp/dpafifo
```

To debug hitless reconnect (HR) for capability l2-lisp, use the following command:

```
echo "debug sfportl2lisp_cache all" > /tmp/dpafifo
```

To debug CBL programming.

```
echo "debug sfpixmagent all" > /tmp/dpafifo
```

To debug a VXLAN agent that interacts with the VSM, use the following command:

```
echo "debug sfvxlanagent all" > /tmp/dpafifo
```

To check the VTEP and MAC addresses, use the following command:

```
Jul  1 10:18:20.852679: num_update_timer_interval 3232890
Jul  1 10:18:20.852716: total_report_vsm_count 0
Jul  1 10:18:20.852739: num_act_VTEP_xfaces 1
Jul  1 10:18:20.853108: DPA READY = TRUE
Jul  1 10:18:20.853223: num_swbds 0
Jul  1 10:18:20.853244: missing_swbds 0
Jul  1 10:18:20.853344: get_VTEP_txnid 3
Jul  1 10:18:20.853372: get_mac_txnid 5
Jul  1 10:18:20.853475: vxlan_retry_intvl 600
Jul  1 10:18:20.853602: get_mac_sent = TRUE
Jul  1 10:18:20.853622: get_VTEP_sent = TRUE
Jul  1 10:18:20.853733: VTEP_mismatch_syslog_sent = FALSE
Jul  1 10:18:20.853843: mac_mismatch_syslog_sent = FALSE
Jul  1 10:18:20.853863: delete_notif_rx(Pending MAC deletes) = FALSE
Jul  1 10:18:20.853876: update timer ticks that pending deletes not sent 0
Jul  1 10:18:20.853890: VxLAN update timer state: 1
Jul  1 10:18:20.853906: VSM connected: FALSE
Jul  1 10:18:20.854021: Last retry slot 0 MAC 00:00:00:00:00:00
Jul  1 10:18:20.854132: Last delete slot 0 MAC 00:00:00:00:00:00
Hash      SWBD      VTEP-Ver     MAC-Ver   Created on DP   Need version check
  0       4096         23           0          1           0
  1       4097         23          24          1           0
  2       4098          0           0          0           0
Note: You can compare the MAC version output on the VSM using the show bridge-domain mac
command and VTEP version output on the VSM using the show bridge-domain VTEP command.
```

To check the MAC addresses to be distributed on the VSM, use the following command:

```
Flags:  R - Report to VSM  I - VSM Informed
Add: MAC to be distributed
Delete: MAC to be un-distributed
Stale - Stale entry in VSM
No VTEP - NO VTEP. Entry to be removed from VSM
Wait - Wait for Attach from VSM
BD       MAC Address       if-index         Id      VTEP        Flags  VM IP-cnt   Retry
4097   00:50:56:97:15:dc   0x1c000480  3218000   172.172.0.134   I(Add)        0       1
4097   00:50:56:97:06:89   0x1c0004b0  3218000   172.172.0.134   I(Add)        0       1
```

Vempkt

Vempkt has been enhanced to display the VLAN/SegmentID. Use vempkt to trace the packet path through the VEM.

- Encapsulated: Capture ingress on Seg-VEth LTL – Egress on uplink
- Decapsulated: Capture ingress on uplink – Egress on Seg-VEth LTL

Statistics

To display a summary of per-port statistics, use the following command:

vemcmd show vxlan-stats

To display detailed per-port statistics for VXLAN VTEP, use the following command:

vemcmd show vxlan-stats ltl *vxlan_VTEP_ltl*

To display detailed per-port statistics for the vEthernet interface in a VXLAN, use the following command:

vemcmd show vxlan-stats ltl *vxlan_veth_ltl*

To display detailed per-port-per-bridge domain statistics for a VXLAN VTEP for all bridge domains, use the following command:

vemcmd show vxlan-stats ltl *vxlan_VTEP_ltl* **bd-all**

To display detailed per-port-per-bridge domain statistics for a VXLAN VTEP for the specified bridge domain, use the following command:

vemcmd show vxlan-stats ltl *vxlan_VTEP_ltl* **bd-name** *bd-name*

To display which VXLAN VTEP is used for encapsulation and subsequent pinning to the uplink port channel for static MAC addresses learned on port, use the following command:

vemcmd show vxlan-encap ltl *vxlan_veth_ltl*

To display which VXLAN VTEP is used for encapsulation and subsequent pinning to the uplink port channel, use the following command:

vemcmd show vxlan-encap mac *vxlan_vm_mac*

show Commands

Command	Result
vemcmd show vxlan interfaces	Displays the VXLAN encapsulated interfaces.
vemcmd show port vlans	Checks the port programming and CBL state for the bridge domain.
vemcmd show bd	Displays the bridge domain segmentId/group/list of ports.
vemcmd show bd bd-name *bd-name-string*	Displays one segment bridge domain.
vemcmd show l2 all	Displays the remote IP being learned.
vemcmd show l2 bd-name *bd-name-string*	Displays the Layer 2 table for one segment bridge domain.
vemcmd show arp all	Displays the IP-MAC mapping for the outer encapsulated header.