Configuring Cross Connect

This chapter contains the following sections:

About VXLAN Cross Connect

This feature provides point-to-point tunneling of data and control packet from one VTEP to another. Every attachment circuit will be part of a unique provider VNI. BGP EVPN signaling will discover these end-points based on how the provider VNI is stretched in the fabric. All inner customer .1q tags will be preserved, as is, and packets will be encapsulated in the provider VNI at the encapsulation VTEP. On the decapsulation end-point, the provider VNI will forward the packet to its attachment circuit while preserving all customer .1q tags in the packets.

Note Cross Connect and xconnect are synonymous.

Beginning with Cisco NX-OS Release 9.2(3), support added for vPC Fabric Peering.

VXLAN Cross Connect enables VXLAN point-to-point functionality on the following switches:

- Cisco Nexus 9332PQ
- Cisco Nexus 9336C-FX2

- Cisco Nexus 9372PX
- Cisco Nexus 9372PX-E
- Cisco Nexus 9372TX
- Cisco Nexus 9372TX-E
- Cisco Nexus 93120TX
- Cisco Nexus 93108TC-EX
- Cisco Nexus 93108TC-FX
- Cisco Nexus 93180LC-EX
- Cisco Nexus 93180YC-EX
- Cisco Nexus 93180YC-FX
- Cisco Nexus 93240YC-FX2

VXLAN Cross Connect enables tunneling of all control frames (CDP, LLDP, LACP, STP, BFD, and PAGP) and data across the VXLAN cloud.

Guidelines and Limitations for VXLAN Cross Connect

VXLAN Cross Connect has the following guidelines and limitations:

- When an upgrade is performed non-disruptively from Cisco NX-OS Release 7.0(3)I7(4) to Cisco NX-OS Release 9.2(x) code, and if a VLAN is created and configured as xconnect, you must enter the **copy running-config startup-config** command and reload the switch. If the box was upgraded disruptively to Cisco NX-OS Release 9.2(x) code, a reload is not needed on configuring a VLAN as xconnect.
- MAC learning will be disabled on the xconnect VNIs and none of the host MAC will be learned on the tunnel access ports.
- Only supported on a BGP EVPN topology.
- LACP bundling of attachment circuits is not supported.
- Only one attachment circuit can be configured for a provider VNI on a given VTEP.
- A VNI can only be stretched in a point-to-point fashion. Point-to-multipoint is not supported.
- SVI on an xconnect VLAN is not supported.
- ARP suppression is not supported on an xconnect VLAN VNI.
- Xconnect is not supported on the following switches:
    - Cisco Nexus 9504
    - Cisco Nexus 9508
    - Cisco Nexus 9516

- Scale of xconnect VLANs depends on the number of ports available on the switch. Every xconnect VLAN can tunnel all 4k customer VLANs.
- Xconnect or Crossconnect feature on vpc-vtep needs backup-svi as native VLAN on the vPC peer-link.
- Make sure that the NGOAM xconnect hb-interval is set to 5000 milliseconds on all VTEPs before attempting ISSU/patch activation to avoid link flaps.
- Before activating the patch for the cfs process, you must move the NGOAM xconnect hb-interval to the maximum value of 5000 milliseconds. This prevents interface flaps during the patch activation.
- The vPC orphan tunneled port per VNI should be either on the vPC primary switch or secondary switch, but not both.
- Configuring a static MAC on xconnect tunnel interfaces is not supported.
- xconnect is not supported on FEX ports.
- On vpc-vtep, spanning tree must be disabled on both vPC peers for xconnect VLANs.
- Xconnect access ports need to be flapped after disabling NGOAM on all the VTEPs.
- After deleting and adding a VLAN, or removing xconnect from a VLAN, physical ports need to be flapped with NGOAM.
- VXLAN Cross Connect is not supported as part of multi-site solution.

Configuring VXLAN Cross Connect

This procedure describes how to configure the VXLAN Cross Connect feature.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	**vlan** *vlan-id* **Example:** `switch(config)# vlan 10`	Specifies VLAN.
Step 3	**vn-segment** *vnid* **Example:** `switch(config-vlan)# vn-segment 10010`	Specifies VXLAN VNID (Virtual Network Identifier).
Step 4	**xconnect** **Example:** `switch(config-vlan)# xconnect`	Defines the provider VLAN with the attached VNI to be in cross connect mode.

	Command or Action	Purpose
Step 5	**exit** **Example:** `switch(config-vlan)# exit`	Exits command mode.
Step 6	**interface** *type port* **Example:** `switch(config)# interface ethernet 1/1`	Enters interface configuration mode.
Step 7	**switchport mode dot1q-tunnel** **Example:** `switch(config-if)# switchport mode dot1q-tunnel`	Creates a 802.1q tunnel on the port. The port will do down and reinitialize (port flap) when the interface mode is changed. BPDU filtering is enabled and CDP is disabled on tunnel interfaces.
Step 8	**switchport access vlan** *vlan-id* **Example:** `switch(config-if)# switchport access vlan 10`	Sets the interface access VLAN.
Step 9	**exit** **Example:** `switch(config-vlan)# exit`	Exits command mode.

Example

This example shows how to configure VXLAN Cross Connect.

```
switch# configure terminal
switch(config)# vlan 10
switch(config)# vn-segment 10010
switch(config)# xconnect
switch(config)# vlan 20
switch(config)# vn-segment 10020
switch(config)# xconnect
switch(config)# vlan 30
switch(config)# vn-segment 10030
switch(config)# xconnect
```

This example shows how to configure access ports:

```
switch# configure terminal
switch(config)# interface ethernet1/1
switch(config-if)# switchport mode dot1q-tunnel
switch(config-if)# switchport access vlan 10
switch(config-if)# exit
switch(config)# interface ethernet1/2
switch(config-if)# switchport mode dot1q-tunnel
switch(config-if)# switchport access vlan 20
switch(config-if)# exit
switch(config)# interface ethernet1/3
switch(config-if)# switchport mode dot1q-tunnel
switch(config-if)# switchport access vlan 30
```

Verifying VXLAN Cross Connect Configuration

To display the status for the VXLAN Cross Connect configuration, enter one of the following commands:

Table 1: Display VXLAN Cross Connect Information

Command	Purpose
show running-config vlan *session-num*	Displays VLAN information.
show nve vni	Displays VXLAN VNI status.
show nve vni *session-num*	Displays VXLAN VNI status per VNI.

Example of the **show run vlan 503** command:

```
switch(config)# sh run vlan 503

!Command: show running-config vlan 503
!Running configuration last done at: Mon Jul  9 13:46:03 2018
!Time: Tue Jul 10 14:12:04 2018

version 9.2(1) Bios:version 07.64
vlan 503
vlan 503
  vn-segment 5503
  xconnect
```

Example of the **show nve vni 5503** command:

```
switch(config)# sh nve vni 5503
Codes: CP - Control Plane        DP - Data Plane
       UC - Unconfigured         SA - Suppress ARP
       SU - Suppress Unknown Unicast
Interface VNI      Multicast-group   State Mode Type [BD/VRF]      Flags
--------- -------- ----------------- ----- ---- ------------------ -----
nve1      5503     225.5.0.3         Up    CP   L2 [503]           SA       Xconn
```

Example of the **show nve vni** command:

```
switch(config)# sh nve vni
Codes: CP - Control Plane        DP - Data Plane
       UC - Unconfigured         SA - Suppress ARP
       SU - Suppress Unknown Unicast
Interface VNI      Multicast-group   State Mode Type [BD/VRF]      Flags
--------- -------- ----------------- ----- ---- ------------------ -----
nve1      5501     225.5.0.1         Up    CP   L2 [501]           SA
nve1      5502     225.5.0.2         Up    CP   L2 [502]           SA
nve1      5503     225.5.0.3         Up    CP   L2 [503]           SA       Xconn
nve1      5504     UnicastBGP        Up    CP   L2 [504]           SA       Xconn
nve1      5505     225.5.0.5         Up    CP   L2 [505]           SA       Xconn
nve1      5506     UnicastBGP        Up    CP   L2 [506]           SA       Xconn
nve1      5507     225.5.0.7         Up    CP   L2 [507]           SA       Xconn
nve1      5510     225.5.0.10        Up    CP   L2 [510]           SA       Xconn
nve1      5511     225.5.0.11        Up    CP   L2 [511]           SA       Xconn
```

```
nve1      5512     225.5.0.12        Up    CP   L2 [512]          SA        Xconn
nve1      5513     UnicastBGP        Up    CP   L2 [513]          SA        Xconn
nve1      5514     225.5.0.14        Up    CP   L2 [514]          SA        Xconn
nve1      5515     UnicastBGP        Up    CP   L2 [515]          SA        Xconn
nve1      5516     UnicastBGP        Up    CP   L2 [516]          SA        Xconn
nve1      5517     UnicastBGP        Up    CP   L2 [517]          SA        Xconn
nve1      5518     UnicastBGP        Up    CP   L2 [518]          SA        Xconn
```

Configuring NGOAM for VXLAN Cross Connect

This procedure describes how to configure NGOAM for VXLAN Cross Connect.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**feature ngoam** **Example:** `switch(config)# feature ngoam`	Enters the NGOAM feature.
Step 3	**ngoam install acl** **Example:** `switch(config)# ngoam install acl`	Installs NGOAM Access Control List (ACL).
Step 4	(Optional) **ngoam xconnect hb-interval** *interval* **Example:** `switch(config)# ngoam xconnect hb-interval 5000`	Configures the heart beat interval. Range of *interval* is 150 to 5000. The default value is 190.

Verifying NGOAM for VXLAN Cross Connect

To display the NGOAM status for the VXLAN Cross Connect configuration, enter one of the following commands:

Table 2: Display VXLAN Cross Connect Information

Command	Purpose
show ngoam xconnect session all	Displays the summary of xconnect sessions.
show ngoam xconnect session *session-num*	Displays detailed xconnect information for the session.

Example of the **show ngoam xconnect session all** command:

```
switch(config)# sh ngoam xconnect session all
```

```
States: LD = Local interface down, RD = Remote interface Down
          HB = Heartbeat lost, DB = Database/Routes not present
          * - Showing Vpc-peer interface info
Vlan          Peer-ip/vni      XC-State       Local-if/State    Rmt-if/State
===============================================================================
507       6.6.6.6 /  5507      Active        Eth1/7 / UP        Eth1/5 / UP
508       7.7.7.7 /  5508      Active        Eth1/8 / UP        Eth1/5 / UP
509       7.7.7.7 /  5509      Active        Eth1/9 / UP        Eth1/9 / UP
510       6.6.6.6 /  5510      Active         Po303 / UP         Po103 / UP
513       6.6.6.6 /  5513      Active        Eth1/6 / UP        Eth1/8 / UP
```

Example of the **show ngoam xconnect session 507** command:

```
switch(config)# sh ngoam xconnect session 507
Vlan ID: 507
Peer IP: 6.6.6.6  VNI : 5507
State: Active
Last state update: 07/09/2018 13:47:03.849
Local interface: Eth1/7  State:  UP
Local vpc interface Unknown  State:  DOWN
Remote interface: Eth1/5  State:  UP
Remote vpc interface: Unknown  State:  DOWN
switch(config)#
```

NGOAM Authentication

NGOAM provides the interface statistics in the pathtrace response. NGOAM authenticates the pathtrace requests to provide the statistics by using the HMAC MD5 authentication mechanism.

NGOAM authentication validates the pathtrace requests before providing the interface statistics. NGOAM authentication takes effect only for the pathtrace requests with **req-stats** option. All the other commands are not affected with the authentication configuration. If NGOAM authentication key is configured on the requesting node, NGOAM runs the MD5 algorithm using this key to generate the 16-bit MD5 digest. This digest is encoded as type-length-value (TLV) in the pathtrace request messages.

When the pathtrace request is received, NGOAM checks for the **req-stats** option and the local NGOAM authentication key. If the local NGOAM authentication key is present, it runs MD5 using the local key on the request to generate the MD5 digest. If both digests match, it includes the interface statistics. If both digests do not match, it sends only the interface names. If an NGOAM request comes with the MD5 digest but no local authentication key is configured, it ignores the digest and sends all the interface statistics. To secure an entire network, configure the authentication key on all nodes.

To configure the NGOAM authentication key, use the **ngoam authentication-key** <*key*> CLI command. Use the **show running-config ngoam** CLI command to display the authentication key.

```
switch# show running-config ngoam
!Time: Tue Mar 28 18:21:50 2017
version 7.0(3)I6(1)
feature ngoam
ngoam profile 1
  oam-channel 2
ngoam profile 3
ngoam install acl
ngoam authentication-key 987601ABCDEF
```

In the following example, the same authentication key is configured on the requesting switch and the responding switch.

```
switch# pathtrace nve ip 12.0.22.1 profile 1 vni 31000 req-stats ver
Path trace Request to peer ip 12.0.22.1 source ip 11.0.22.1
Hop   Code   ReplyIP   IngressI/f  EgressI/f   State
====================================================
  1 !Reply from 55.55.55.2, Eth5/7/1  Eth5/7/2  UP / UP
  Input Stats: PktRate:0 ByteRate:0 Load:0 Bytes:339573434 unicast:14657 mcast:307581
bcast:67 discards:0 errors:3 unknown:0 bandwidth:42949672970000000
Output Stats: PktRate:0 ByteRate:0 load:0 bytes:237399176 unicast:2929 mcast:535710
bcast:10408 discards:0 errors:0 bandwidth:42949672970000000
  2 !Reply from 12.0.22.1, Eth1/7  Unknown  UP / DOWN
  Input Stats: PktRate:0 ByteRate:0 Load:0 Bytes:4213416 unicast:275 mcast:4366 bcast:3
discards:0 errors:0 unknown:0 bandwidth:42949672970000000
switch# conf t
switch(config)# no ngoam authentication-key 123456789
switch(config)# end
```

In the following example, an authentication key is not configured on the requesting switch. Therefore, the responding switch does not send any interface statistics. The intermediate node does not have any authentication key configured and it always replies with the interface statistics.

```
switch# pathtrace nve ip 12.0.22.1 profile 1 vni 31000 req-stats ver
Path trace Request to peer ip 12.0.22.1 source ip 11.0.22.1
Sender handle: 10
Hop   Code   ReplyIP   IngressI/f  EgressI/f   State
====================================================
  1 !Reply from 55.55.55.2, Eth5/7/1  Eth5/7/2  UP / UP
  Input Stats: PktRate:0 ByteRate:0 Load:0 Bytes:339580108 unicast:14658 mcast:307587
bcast:67 discards:0 errors:3 unknown:0 bandwidth:42949672970000000
Output Stats: PktRate:0 ByteRate:0 load:0 bytes:237405790 unicast:2929 mcast:535716
bcast:10408 discards:0 errors:0 bandwidth:42949672970000000
  2 !Reply from 12.0.22.1, Eth1/17  Unknown  UP / DOWN
```

Guidelines and Limitations for Q-in-VNI

Q-in-VNI has the following guidelines and limitations:

- Q-in-VNI and selective Q-in-VNI are supported with VXLAN Flood and Learn with Ingress Replication and VXLAN EVPN with Ingress Replication.
- Q-in-VNI, selective Q-in-VNI, and QinQ-QinVNI are not supported with the multicast underlay on Cisco Nexus 9000-EX platform switches.
- The **system dot1q-tunnel transit** command is required when running this feature on vPC VTEPs.
- Port VLAN mapping and Q-in-VNI cannot coexist on the same port.
- Port VLAN mapping and Q-in-VNI cannot coexist on a switch if the **system dot1q-tunnel transit** command is enabled.
- For proper operation during L3 uplink failure scenarios on vPC VTEPs, configure a backup SVI and enter the **system nve infra-vlans** *backup-svi-vlan* command. On Cisco Nexus 9000-EX platform switches, the backup SVI VLAN needs to be the native VLAN on the peer-link.

- Q-in-VNI only supports VXLAN bridging. It does not support VXLAN routing.
- The dot1q tunnel mode does not support ALE ports on Cisco Nexus 9300 Series and Cisco Nexus 9500 platform switches.
- Q-in-VNI does not support FEX.
- When configuring access ports and trunk ports for Cisco Nexus 9000 Series switches with a Network Forwarding Engine (NFE) or a Leaf Spine Engine (LSE), you can have access ports, trunk ports, and dot1q ports on different interfaces on the same switch.
- You cannot have the same VLAN configured for both dot1q and trunk ports/access ports.
- Disable ARP suppression on the provider VNI for ARP traffic originated from a customer VLAN in order to flow.

  ```
  switch(config)# interface nve 1
  switch(config-if-nve)# member VNI 10000011
  switch(config-if-nve-vni)# no suppress-arp
  ```

- Cisco Nexus 9300 platform switches support single tag. You can enable it by entering the **no overlay-encapsulation vxlan-with-tag** command for the NVE interface:

  ```
  switch(config)# interface nve 1
  switch(config-if-nve)# no overlay-encapsulation vxlan-with-tag
  switch# show run int nve 1

  !Command: show running-config interface nve1
  !Time: Wed Jul 20 23:26:25 2016

  version 7.0(3u)I4(2u)

  interface nve1
    no shutdown
    source-interface loopback0
    host-reachability protocol bgp
    member vni 900001 associate-vrf
    member vni 2000980
      mcast-group 225.4.0.1
  ```

- Cisco Nexus 9500 platform switches do not support single tag. They support only double tag.
- Cisco Nexus 9300-EX platform switches do not support double tag. They support only single tag.
- Cisco Nexus 9300-EX platform switches do not support traffic between ports configured for Q-in-VNI and ports configured for trunk.
- Q-in-VNI cannot coexist with a VTEP that has Layer 3 subinterfaces configured.
- When VLAN1 is configured as the native VLAN with selective Q-in-VNI with the multiple provider tag, traffic on the native VLAN gets dropped. Do not configure VLAN1 as the native VLAN when the port is configured with selective Q-in-VNI. When VLAN1 is configured as a customer VLAN, the traffic on VLAN1 gets dropped.
- The base port mode must be a dot1q tunnel port with an access VLAN configured.
- VNI mapping is required for the access VLAN on the port.
- If you have Q-in-VNI on one Cisco Nexus 9300-EX Series switch VTEP and trunk on another Cisco Nexus 9300-EX Series switch VTEP, the bidirectional traffic will not be sent between the two ports.

- Cisco Nexus 9300-EX Series of switches performing VXLAN and Q-in-Q, a mix of provider interface and VXLAN uplinks is not considered. The VXLAN uplinks have to be separated from the Q-in-Q provider or customer interface.

  For vPC use cases, the following considerations must be made when VXLAN and Q-in-Q are used on the same switch.

  - The vPC peer-link has to be specifically configured as a provider interface to ensure orphan-to-orphan port communication. In these cases, the traffic is sent with two IEEE 802.1q tags (double dot1q tagging). The inner dot1q is the customer VLAN ID while the outer dot1q is the provider VLAN ID (access VLAN).
  - The vPC peer-link is used as backup path for the VXLAN encapsulated traffic in the case of an uplink failure. In Q-in-Q, the vPC peer-link also acts as the provider interface (orphan-to-orphan port communication). In this combination, use the native VLAN as the backup VLAN for traffic to handle uplink failure scenarios. Also make sure the backup VLAN is configured as a system infra VLAN (system nve infra-vlans).
- Q-in-VNI does not support vPC Fabric Peering.
- BPDU filter is required for Selective Q-in-VNI, as we do not support tunneling STP BPDU.

Configuring Q-in-VNI

Using Q-in-VNI provides a way for you to segregate traffic by mapping to a specific port. In a multi-tenant environment, you can specify a port to a tenant and send/receive packets over the VXLAN overlay.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**interface** *type port*	Enters interface configuration mode.
Step 3	**switchport mode dot1q-tunnel**	Creates a 802.1Q tunnel on the port.
Step 4	**switchport access vlan** *vlan-id*	Specifies the port assigned to a VLAN.
Step 5	**spanning-tree bpdufilter enable**	Enables BPDU Filtering for the specified spanning tree edge interface. By default, BPDU Filtering is disabled.

Example

The following is an example of configuring Q-in-VNI:

```
switch# config terminal
switch(config)# interface ethernet 1/4
switch(config-if)# switchport mode dot1q-tunnel
switch(config-if)# switchport access vlan 10
switch(config-if)# spanning-tree bpdufilter enable
switch(config-if)#
```

Configuring Selective Q-in-VNI

Selective Q-in-VNI is a VXLAN tunneling feature that allows a user specific range of customer VLANs on a port to be associated with one specific provider VLAN. Packets that come in with a VLAN tag that matches any of the configured customer VLANs on the port are tunneled across the VXLAN fabric using the properties of the service provider VNI. The VXLAN encapsulated packet carries the customer VLAN tag as part of the L2 header of the inner packet.

The packets that come in with a VLAN tag that is not present in the range of the configured customer VLANs on a selective Q-in-VNI configured port are dropped. This includes the packets that come in with a VLAN tag that matches the native VLAN on the port. Packets coming untagged or with a native VLAN tag are L3 routed using the native VLAN's SVI that is configured on the selective Q-in-VNI port (no VXLAN).

See the following guidelines for selective Q-in-VNI:

- Selective Q-in-VNI is supported on both vPC and non-vPC ports on Cisco Nexus 9300-EX and 9300-FX/FXP/FX2 platform switches. This feature is not supported on Cisco Nexus 9200 and 9300 platform switches.
- Selective Q-in-VNI does not support vPC Fabric Peering.
- Configuring selective Q-in-VNI on one VTEP and configuring plain Q-in-VNI on the VXLAN peer is supported. Configuring one port with selective Q-in-VNI and the other port with plain Q-in-VNI on the same switch is supported.
- Selective Q-in-VNI is an ingress VLAN tag-policing feature. Only ingress VLAN tag policing is performed with respect to the selective Q-in-VNI configured range.

  For example, selective Q-in-VNI customer VLAN range of 100-200 is configured on VTEP1 and customer VLAN range of 200-300 is configured on VTEP2. When traffic with VLAN tag of 175 is sent from VTEP1 to VTEP2, the traffic is accepted on VTEP1, since the VLAN is in the configured range and it is forwarded to the VTEP2. On VTEP2, even though VLAN tag 175 is not part of the configured range, the packet egresses out of the selective Q-in-VNI port. If a packet is sent with VLAN tag 300 from VTEP1, it is dropped because 300 is not in VTEP1's selective Q-in-VNI configured range.
- Port VLAN mapping and selective Q-in-VNI cannot coexist on the same port.
- Port VLAN mapping and selective Q-in-VNI cannot coexist on a switch if the **system dot1q-tunnel transit** command is enabled.
- Configure the **system dot1q-tunnel transit** command on vPC switches with selective Q-in-VNI configurations. This command is required to retain the inner Q-tag as the packet goes over the vPC peer link when one of the vPC peers has an orphan port. With this CLI configuration, the **vlan dot1Q tag native** functionality does not work.
- The native VLAN configured on the selective Q-in-VNI port cannot be a part of the customer VLAN range. If the native VLAN is part of the customer VLAN range, the configuration is rejected.

  The provider VLAN can overlap with the customer VLAN range. For example, **switchport vlan mapping 100-1000 dot1q-tunnel 200**.

- By default, the native VLAN on any port is VLAN 1. If VLAN 1 is configured as part of the customer VLAN range using the **switchport vlan mapping** <*range*>**dot1q-tunnel** <*sp-vlan*> CLI command, the traffic with customer VLAN 1 is not carried over as VLAN 1 is the native VLAN on the port. If customer wants VLAN 1 traffic to be carried over the VXLAN cloud, they should configure a dummy native VLAN on the port whose value is outside the customer VLAN range.
- To remove some VLANs or a range of VLANs from the configured switchport VLAN mapping range on the selective Q-in-VNI port, use the **no** form of the **switchport vlan mapping** <*range*>**dot1q-tunnel** <*sp-vlan*> command.

  For example, VLAN 100-1000 is configured on the port. To remove VLAN 200-300 from the configured range, use the **no switchport vlan mapping** <*200-300*> **dot1q-tunnel** <*sp-vlan*> command.

```
interface Ethernet1/32
  switchport
  switchport mode trunk
  switchport trunk native vlan 4049
  switchport vlan mapping 100-1000 dot1q-tunnel 21
  switchport trunk allowed vlan 21,4049
  spanning-tree bpdufilter enable
  no shutdown

switch(config-if)# no sw vlan mapp 200-300 dot1q-tunnel 21
switch(config-if)# sh run int e 1/32

version 7.0(3)I5(2)

interface Ethernet1/32
  switchport
  switchport mode trunk
  switchport trunk native vlan 4049
  switchport vlan mapping 100-199,301-1000 dot1q-tunnel 21
  switchport trunk allowed vlan 21,4049
  spanning-tree bpdufilter enable
  no shutdown
```

See the following configuration examples.

- See the following example for the provider VLAN configuration:

```
vlan 50
 vn-segment 10050
```

- See the following example for configuring VXLAN Flood and Learn with Ingress Replication:

```
member vni 10050
    ingress-replication protocol static
      peer-ip 100.1.1.3
      peer-ip 100.1.1.5
      peer-ip 100.1.1.10
```

- See the following example for the interface nve configuration:

```
interface nve1
 no shutdown
 source-interface loopback0  member vni 10050
mcast-group 230.1.1.1
```

- See the following example for configuring an SVI in the native VLAN to routed traffic.

```
vlan 150
interface vlan150
 no shutdown
 ip address 150.1.150.6/24
 ip pim sparse-mode
```

- See the following example for configuring selective Q-in-VNI on a port. In this example, native VLAN 150 is used for routing the untagged packets. Customer VLANs 200-700 are carried across the dot1q tunnel. The native VLAN 150 and the provider VLAN 50 are the only VLANs allowed.

```
switch# config terminal
switch(config)#interface Ethernet 1/31
switch(config-if)#switchport
switch(config-if)#switchport mode trunk
switch(config-if)#switchport trunk native vlan 150
switch(config-if)#switchport vlan mapping 200-700 dot1q-tunnel 50
switch(config-if)#switchport trunk allowed vlan 50,150
switch(config-if)#no shutdown
```

- Disable ARP suppression on the provider VNI for ARP traffic originated from a customer VLAN in order to flow.

```
switch(config)# interface nve 1
switch(config-if-nve)# member VNI 10000011
 switch(config-if-nve-vni)# no suppress-arp
```

Configuring Q-in-VNI with LACP Tunneling

Q-in-VNI can be configured to tunnel LACP packets.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**interface** *type port*	Enters interface configuration mode.
Step 3	**switchport mode dot1q-tunnel**	Enables dot1q-tunnel mode.
Step 4	**switchport access vlan** *vlan-id*	Specifies the port assigned to a VLAN.
Step 5	**interface nve** *x*	Creates a VXLAN overlay interface that terminates VXLAN tunnels.
Step 6	**overlay-encapsulation vxlan-with-tag tunnel-control-frames lacp**	Enables Q-in-VNI for LACP tunneling.

	Command or Action	Purpose
		Note Use this form of the command for NX-OS 7.0(3)I3(1) and later releases. For NX-OS 7.0(3)I2(2) and earlier releases, use the **overlay-encapsulation vxlan-with-tag tunnel-control-frames** command.

Example

- The following is an example of configuring a Q-in-VNI for LACP tunneling (NX-OS 7.0(3)I2(2) and earlier releases):

```
switch# config terminal
switch(config)# interface ethernet 1/4
switch(config-if)# switchport mode dot1q-tunnel
switch(config-if)# switchport access vlan 10
switch(config-if)# spanning-tree bpdufilter enable
switch(config-if)# interface nve1
switch(config-if)# overlay-encapsulation vxlan-with-tag tunnel-control-frames
```

Note

- STP is disabled on VNI mapped VLANs.
- No spanning-tree VLAN <> on the VTEP.
- No MAC address-table notification for mac-move.
- As a best practice, configure a fast LACP rate on the interface where the LACP port is configured. Otherwise the convergence time is approximately 90 seconds.

- The following is an example of configuring a Q-in-VNI for LACP tunneling (NX-OS 7.0(3)I3(1) and later releases):

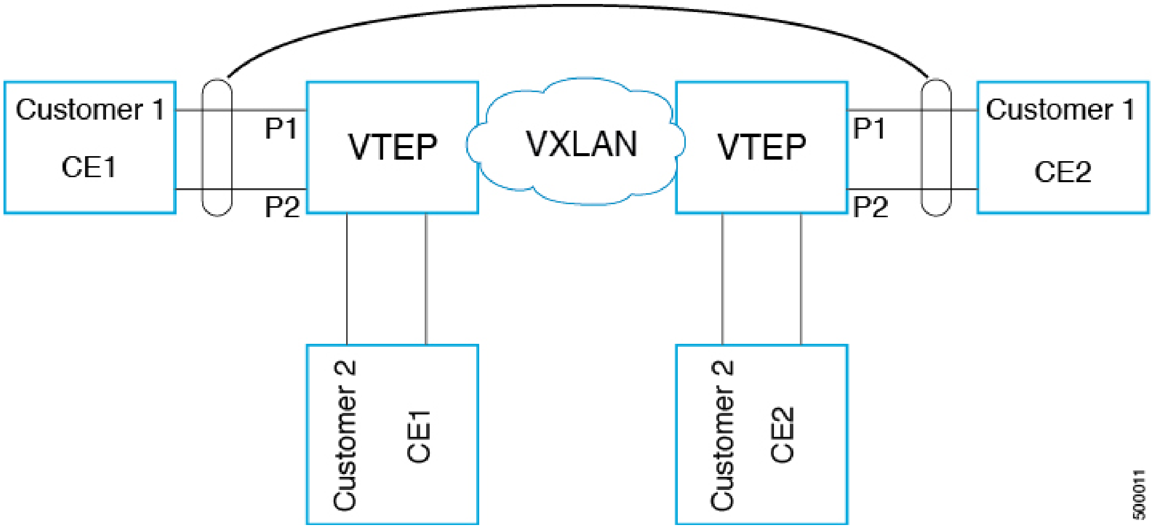
```
switch# config terminal
switch(config)# interface ethernet 1/4
switch(config-if)# switchport mode dot1q-tunnel
switch(config-if)# switchport access vlan 10
switch(config-if)# spanning-tree bpdufilter enable
switch(config-if)# interface nve1
switch(config-if)# overlay-encapsulation vxlan-with-tag tunnel-control-frames lacp
```

Note

- STP is disabled on VNI mapped VLANs.
- No spanning-tree VLAN <> on the VTEP.
- No MAC address-table notification for mac-move.
- As a best practice, configure a fast LACP rate on the interface where the LACP port is configured. Otherwise the convergence time is approximately 90 seconds.

- The following is an example topology that pins each port of a port-channel pair to a unique VM. The port-channel is stretched from the CE perspective. There is no port-channel on VTEP. The traffic on P1 of CE1 transits to P1 of CE2 using Q-in-VNI.

Figure 1: LACP Tunneling Over VXLAN P2P Tunnels



Note

- Q-in-VNI can be configured to tunnel LACP packets. (Able to provide port-channel connectivity across data-centers.)
    - Gives impression of L1 connectivity and co-location across data-centers.
    - Exactly two sites. Traffic coming from P1 of CE1 goes out of P1 of CE2. If P1 of CE1 goes down, LACP provides coverage (over time) to redirect traffic to P2.
- Uses static ingress replication with VXLAN with flood and learn. Each port of the port channel is configured with Q-in-VNI. There are multiple VNIs for each member of a port-channel and each port is pinned to specific VNI.
    - To avoid saturating the MAC, you should turn off/disable learning of VLANS.
- Configuring Q-in-VNI to tunnel LACP packets is not supported for VXLAN EVPN.
- The number of port-channel members supported is the number of ports supported by the VTEP.

Selective Q-in-VNI with Multiple Provider VLANs

About Selective Q-in-VNI with Multiple Provider VLANs

Selective Q-in-VNI with multiple provider VLANs is a VXLAN tunneling feature. This feature allows a user specific range of customer VLANs on a port to be associated with one specific provider VLAN. It also enables you to have multiple customer-VLAN to provider-VLAN mappings on a port. Packets that come in with a VLAN tag which matches any of the configured customer VLANs on the port are tunneled across the VXLAN fabric using the properties of the service provider VNI. The VXLAN encapsulated packet carries the customer VLAN tag as part of the Layer 2 header of the inner packet.

Guidelines and Limitations for Selective Q-in-VNI with Multiple Provider VLANs

Selective Q-in-VNI with multiple provider VLANs has the following guidelines and limitations:

- All the existing guidelines and limitations for Selective Q-in-VNI apply.
- This feature is supported with VXLAN BGP EVPN IR mode only.
- When enabling multiple provider VLANs on a vPC port channel, make sure that the configuration is consistent across the vPC peers.
- Port VLAN mapping and selective Q-in-VNI cannot coexist on the same port.
- Port VLAN mapping and selective Q-in-VNI cannot coexist on a switch if the **system dot1q-tunnel transit** command is enabled.
- The **system dot1q-tunnel transit** command is required when using this feature on vPC VTEPs.

- For proper operation during Layer 3 uplink failure scenarios on vPC VTEPs, configure the backup SVI and enter the **system nve infra-vlans** *backup-svi-vlan* command. On Cisco Nexus 9000-EX platform switches, the backup SVI VLAN must be the native VLAN on the peer-link.
- As a best practice, do not allow provider VLANs on a regular trunk.
- We recommend not creating or allowing customer VLANs on the switch where customer-VLAN to provider-VLAN mapping is configured.
- We do not support specific native VLAN configuration when the **switchport vlan mapping all dot1q-tunnel** command is entered.
- Selective Q-in-VNI with a multiple provider tag does not support vPC Fabric Peering.
- Disable ARP suppression on the provider VNI for ARP traffic originated from a customer VLAN in order to flow.

```
switch(config)# interface nve 1
switch(config-if-nve)# member VNI 10000011
switch(config-if-nve-vni)# no suppress-arp
```

- All incoming traffic should be tagged when the interface is configured with the **switchport vlan mapping all dot1q-tunnel** command.

Configuring Selective Q-in-VNI with Multiple Provider VLANs

You can configure selective Q-in-VNI with multiple provider VLANs.

Before you begin

You must configure provider VLANs and associate the VLAN to a vn-segment.

Procedure

Step 1 Enter global configuration mode.

```
switch# configure terminal
```

Step 2 Configure Layer 2 VLANs and associate them to a vn-segment.

```
switch(config)# vlan 10
vn-segment 10000010
switch(config)# vlan 20
 vn-segment 10000020
```

Step 3 Enter interface configuration mode where the traffic comes in with a dot1Q VLAN tag.

```
switch(config)# interf port-channel 10
switch(config-if)# switchport
switch(config-if)# switchport mode trunk
switch(config-if)# switchport trunk native vlan 3962
switch(config-if)# switchport vlan mapping 2-400 dot1q-tunnel 10
switch(config-if)# switchport vlan mapping 401-800 dot1q-tunnel 20
switch(config-if)# switchport vlan mapping 801-1200 dot1q-tunnel 30
switch(config-if)# switchport vlan mapping 1201-1600 dot1q-tunnel 40
switch(config-if)# switchport vlan mapping 1601-2000 dot1q-tunnel 50
switch(config-if)# switchport vlan mapping 2001-2400 dot1q-tunnel 60
switch(config-if)# switchport vlan mapping 2401-2800 dot1q-tunnel 70
switch(config-if)# switchport vlan mapping 2801-3200 dot1q-tunnel 80
switch(config-if)# switchport vlan mapping 3201-3600 dot1q-tunnel 90
switch(config-if)# switchport vlan mapping 3601-3960 dot1q-tunnel 100
switch(config-if)# switchport trunk allowed vlan 10,20,30,40,50,60,70,80,90,100,3961-3967
```

Example

This example shows how to configure Selective Qinvni with multiple provider VLANs:

```
switch# show run vlan 121
vlan 121
vlan 121
  vn-segment 10000021

switch#
switch# sh run interf port-channel 5

interface port-channel5
  description VPC PO
  switchport
  switchport mode trunk
  switchport trunk native vlan 504
  switchport vlan mapping 11 dot1q-tunnel 111
  switchport vlan mapping 12 dot1q-tunnel 112
  switchport vlan mapping 13 dot1q-tunnel 113
  switchport vlan mapping 14 dot1q-tunnel 114
  switchport vlan mapping 15 dot1q-tunnel 115
  switchport vlan mapping 16 dot1q-tunnel 116
  switchport vlan mapping 17 dot1q-tunnel 117
  switchport vlan mapping 18 dot1q-tunnel 118
  switchport vlan mapping 19 dot1q-tunnel 119
  switchport vlan mapping 20 dot1q-tunnel 120
  switchport trunk allowed vlan 111-120,500-505
  vpc 5

switch#

switch# sh spanning-tree vlan 111

VLAN0111
  Spanning tree enabled protocol rstp
  Root ID    Priority    32879
             Address     7079.b3cf.956d
             This bridge is the root
             Hello Time  2  sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority    32879  (priority 32768 sys-id-ext 111)
             Address     7079.b3cf.956d
             Hello Time  2  sec  Max Age 20 sec  Forward Delay 15 sec

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Po1              Desg FWD 1         128.4096 (vPC peer-link) Network P2p
Po5              Desg FWD 1         128.4100 (vPC) P2p
Eth1/7/2         Desg FWD 10        128.26   P2p

switch#


switch# sh vlan internal info mapping | b Po5
  ifindex Po5(0x16000004)
  vlan mapping enabled: TRUE
  vlan translation mapping information (count=10):
    Original Vlan       Translated Vlan
    -------------       ---------------
    11                   111
    12                   112
    13                   113
    14                   114
    15                   115
    16                   116
    17                   117
    18                   118
    19                   119
    20                   120
switch#


switch# sh consistency-checker vxlan selective-qinvni interface port-channel 5
Performing port specific checks for intf port-channel5
Port specific selective QinVNI checks for interface port-channel5 : PASS
Performing port specific checks for intf port-channel5
Port specific selective QinVNI checks for interface port-channel5 : PASS

switch#
```

Configuring QinQ-QinVNI

Overview for QinQ-QinVNI

- QinQ-QinVNI is a VXLAN tunneling feature that allows you to configure a trunk port as a multi-tag port to preserve the customer VLANs that are carried across the network.
- On a port that is configured as multi-tag, packets are expected with multiple-tags or at least one tag. When multi-tag packets ingress on this port, the outer-most or first tag is treated as provider-tag or provider-vlan. The remaining tags are treated as customer-tag or customer-vlan.
- This feature is supported on both vPC and non-vPC ports.
- Ensure that the **switchport trunk allow-multi-tag** command is configured on both of the vPC-peers. It is a type 1 consistency check.
- This feature is supported with VXLAN Flood and Learn and VXLAN EVPN.

Guidelines and Limitations for QinQ-QinVNI

QinQ-QinVNI has the following guidelines and limitations:

- This feature is supported on the Cisco Nexus 9300-FX/FX2 switches.
- This feature supports vPC Fabric Peering, beginning with Cisco NX-OS Release 9.2(3). For more information, see the Configuring vPC Fabric Peering chapter.
- On a multi-tag port, provider VLANs must be a part of the port. They are used to derive the VNI for that packet.

- Untagged packets are associated with the native VLAN. If the native VLAN is not configured, the packet is associated with the default VLAN (VLAN 1).
- Packets coming in with an outermost VLAN tag (provider-vlan), not present in the range of allowed VLANs on a multi-tag port, are dropped.
- Packets coming in with an outermost VLAN tag (provider-vlan) tag matching the native VLAN are routed or bridged in the native VLAN's domain.
- This feature is supported with VXLAN bridging. It does not support VXLAN routing.
- Multicast data traffic with more than two Q-Tags in not supported when snooping in enabled on the VXLAN VLAN.
- You need at least one multi-tag trunk port allowing the provider VLANs in **up** state on both the vPC peers. Otherwise, traffic traversing via the peer-link for these provider VLANs will not carry all inner C-Tags.

Configuring QinQ-QinVNI

Note You can also carry native VLAN (untagged traffic) on the same multi-tag trunk port.

The native VLAN on a multi-tag port cannot be configured as a provider VLAN on another multi-tag port or a dot1q enabled port on the same switch.

The **allow-multi-tag** command is allowed only on a trunk port. It is not available on access or dot1q ports.

The **allow-multi-tag** command is not allowed on Peer Link ports. Port channel with multi-tag enabled must not be configured as a vPC peer-link.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	**interface ethernet** *slot/port* **Example:** `switch(config)# interface ethernet1/7`	Specifies the interface that you are configuring.
Step 3	**switchport** **Example:** `switch(config-inf)# switchport`	Configures it as a Layer 2 port.
Step 4	**switchport mode trunk** **Example:** `switch(config-inf)# switchport mode trunk`	Sets the interface as a Layer 2 trunk port.

	Command or Action	Purpose
Step 5	**switchport trunk native vlan** *vlan-id* **Example:** `switch(config-inf)#` **`switchport trunk native vlan 30`**	Sets the native VLAN for the 802.1Q trunk. Valid values are from 1 to 4094. The default value is VLAN1.
Step 6	**switchport trunk allowed vlan** *vlan-list* **Example:** `switch(config-inf)#` **`switchport trunk allowed vlan 10,20,30`**	Sets the allowed VLANs for the trunk interface. The default is to allow all VLANs on the trunk interface: 1 to 3967 and 4048 to 4094. VLANs 3968 to 4047 are the default VLANs reserved for internal use by default.
Step 7	**switchport trunk allow-multi-tag** **Example:** `switch(config-inf)#` **`switchport trunk allow-multi-tag`**	Sets the allowed VLANs as the provider VLANs excluding the native VLAN. In the following example, VLANs 10 and 20 are provider VLANs and can carry multiple Inner Q-tags. Native VLAN 30 will not carry inner Q-tags.

Example

```
interface Ethernet1/7
switchport
switchport mode trunk
switchport trunk native vlan 30
switchport trunk allow-multi-tag
switchport trunk allowed vlan 10,20,30
no shutdown
```

Removing a VNI

Use this procedure to remove a VNI.

Procedure

Step 1 Remove the VNI under NVE.

Step 2 Remove the VRF from BGP (applicable when decommissioning for Layer 3 VNI).

Step 3 Delete the SVI.

Step 4 Delete the VLAN and VNI.