Introduction to the VXLAN Lab

Welcome

Introduction to the VXLAN Lab (LTRDCN-1469)

Welcome to the VXLAN Lab (LTRDCN-1469) at Cisco Live!

The VXLAN Lab (LTRDCN-1469) is presented by:

Name	Department	Role	Email
Shyla Karimanye	CX.SVS	Customer Delivery Engineering,Technical Leader	skariman@cisco.com
Rohith Ramannagari	CX.SVS	Customer Delivery Engineering,Technical Leader	roramann@cisco.com
Ambrish Singh	Data Center Switching	Technical Marketing Engineer,Technical Leader	ambsingh@cisco.com

You have entered into the lab manual for **pod {{spod}}**. If this pod number does not match the pod number written on the pod assignment document given to you by one of the proctors, please click here and choose the correct pod number.

To proceed to the next section of the lab, please click on the "Next" button on the bottom right

corner of this page. To go back a page, click the "Back" button of the lower left corner of any page.

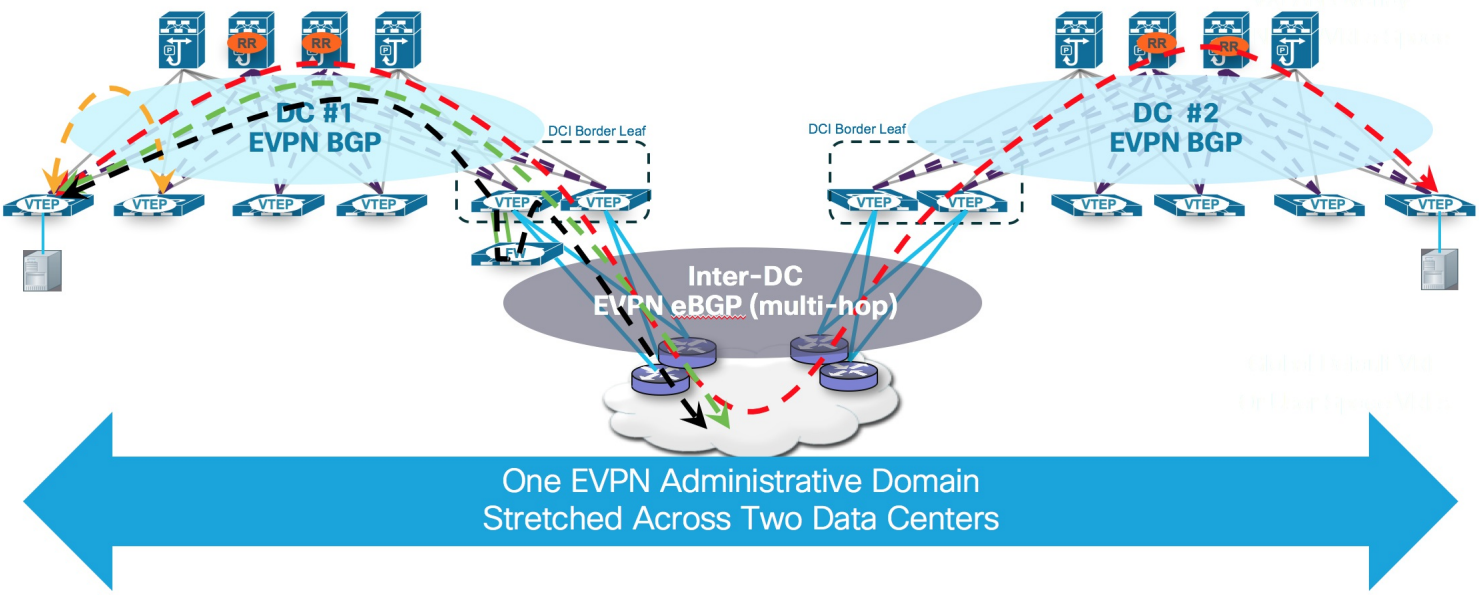
Introduction

Introduction to the VXLAN Lab (LTRDCN-1469)

In the current datacenter, server virtualization has placed increased demands on the physical network, not only for large scale of IP/MAC but also for VM mobility across the network. In addition to this, each tenant now wants to have their own dedicated network where they can easily expand their VLANs anywhere in the network over the cloud. With the increased IP/MAC and VLAN scale, expanding layer 2 across multiple pods is also challenging in large datacenters. The challenges described above lead to the creation of a virtualized overlay network known as Virtual Extensible LAN (VXLAN) to carry IP traffic from individual hosts, in an encapsulated format, over a logical "tunnel".

This lab session will demonstrate deploying VXLAN (Virtual Extensible LAN) in a data center using hardware VXLAN endpoints (VTEPs) with the Nexus 9000 series platform. This lab will be focused on multiple scenarios using the hardware VTEPs to help you implement the VXLAN solution in your network. The following topology, shown below, depicts the high level design that we cover in our lab. It should help you understand multiple solutions available for VXLAN and its use cases.

We will be using Nexus 9ks as hardware VTEPs and running code that includes BGP EVPN for VXLAN control plane learning and Ingress Replication for BUM (broadcast, unicast, multicast) Traffic. In addition, this lab will cover how to extend Layer 2 networks across two datacenters by using a Multipod VXLAN fabric.

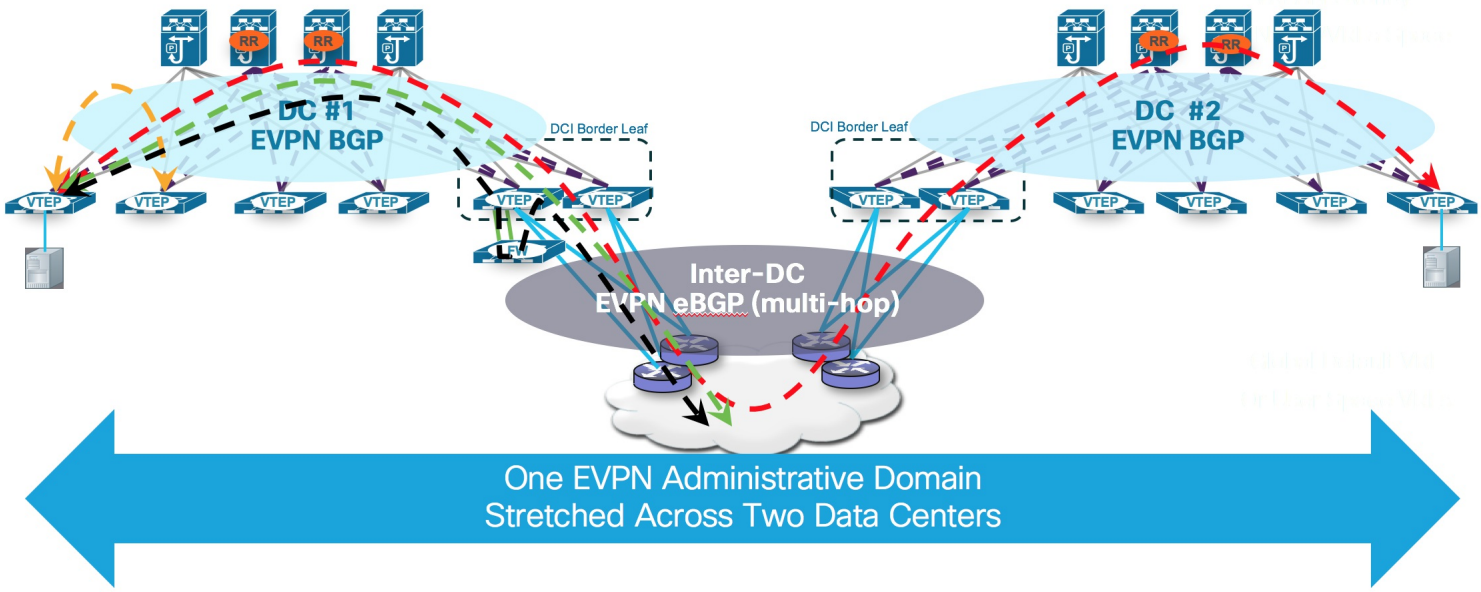

DC #1
EVPN BGP
DCI Border Leaf
DCI Border Leaf
DC #2
EVPN BGP
RR
VTEP
FW
Inter-DC
EVPN eBGP (multi-hop)
One EVPN Administrative Domain
Stretched Across Two Data Centers

VXLAN Lab Use Cases

Use Case 1

Use Case 1

Datacenter 1 - Network Overlay (Hardware Based VTEPs) for VXLAN Bridging/Routing and Connecting to an Outside Network



In this use case, each tenant will extend their layer 2 network across a layer 3 underlay network using VXLAN. For example, a host that is part of Room A will be able to create a virtual layer 2

overlay network, using any hardware capable VTEPs (such as the Nexus 3100 or Nexus 9300), to a host in Room B as well as able to route between VXLANs. This example is illustrated in the figure above with orange lines. Host A can also be able to communicate with a host outside of the network (on the internet for example). This example is illustrated in the figure above with green lines.

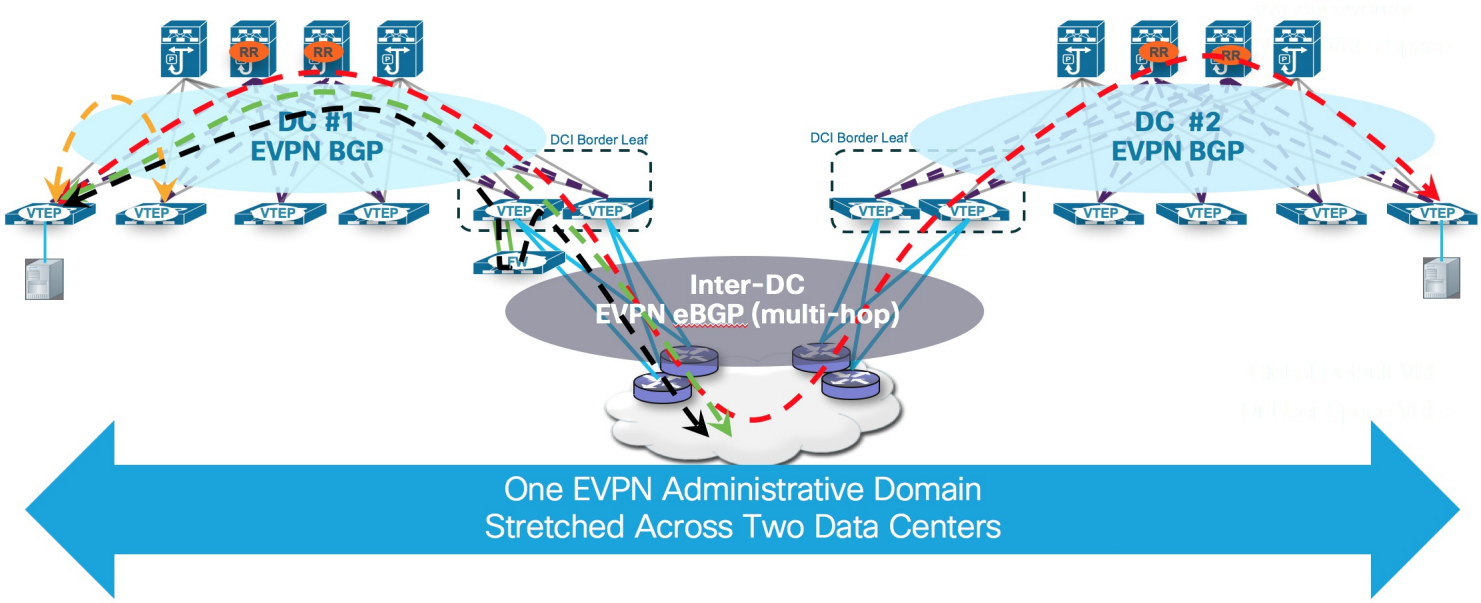
In this use case, the student will be able to configure VXLAN in the following scenarios.

1. Bridging a VXLAN across two leaf nodes connected through spine- Room A and Room B
2. Routing a VXLAN across two leaf nodes connected through spine using Anycast Gateways - Room A and Room B
3. Connecting the Datacenter to an Outside Network using VXLAN Routing

Use Case 2

Use Case 2

Connecting the Datacenter to an Outside Network through a Firewall using VXLAN Routing



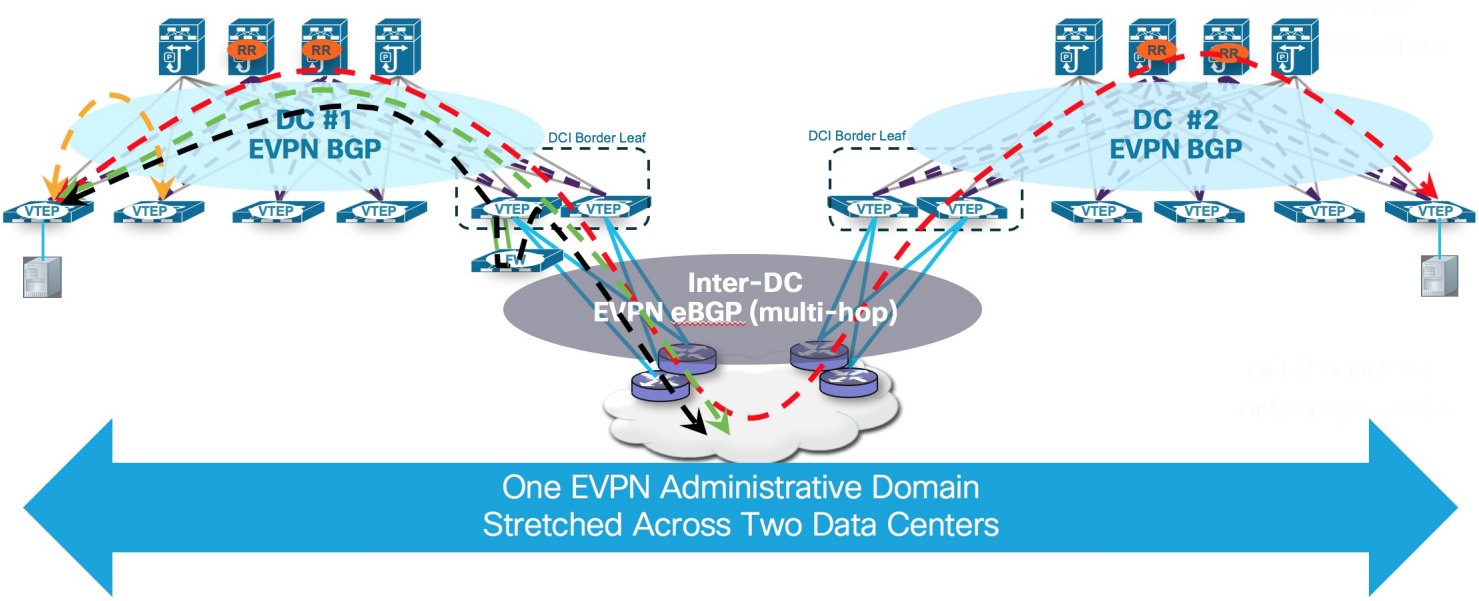
In this use case, we will be creating trusted and non-trusted zones and forcing traffic through the firewall. The firewall is this case is in transport mode.

As shown the diagram in the figure above, the black lines depict how the host on Room A will be able to ping the Internet host but, in this case, traffic is forced through the Transparent firewall.

Use Case 3

Use Case 3

VXLAN Multipod (Fabric)



Important!

This use case is optional.

The VXLAN fabric can be extended between data centers by using VXLAN Multipod. VXLAN can be extended across a LAN or WAN. This approach treats multiple data center fabrics as one single large administrative domain, which simplifies the operational aspects of the deployment. It can be deployed between any pods, whether they are in the same room in the same physical

data center or in separate data center locations.

Another way to extend the datacenter is to have separate fabrics with their own administrative domains. Then, data center interconnect (DCI) protocols such as OTV can be used to connect remote data centers. This lab only focuses on VXLAN multipod with a single administrator domain.

In this use case, we will extend one of the VLANs to the remote DataCenter B/POD 2 as shown using red lines in the figure above. A host in Room A in DataCenter A/POD 1 will be able to ping the host in Room C in DataCenter B/POD 2 . For simplicity purposes, we have one spine and one compute leaf in DataCenter B. In this use case, a host that is attached to DataCenter B can not only be part of the same VLAN/VXLAN as is present in DataCenter A, but it can also access any host in DataCenter A that are members of any other VXLAN/VLANs.

Lab Flow

Lab Flow

Students will first work through use case 1 and then use case 2. At that point, students will need to decide if they want to do use case 3. There may not be enough time during the lab to do all three use cases. If students desire to do all three but do not have time, they can request access to the lab after the session is finished. To request access, either ask a proctor or email us using the contact information on the welcome page.

Lab Information

Lab Prerequisites

Lab Prerequisites

- Familiarity with Cisco NX-OS
- Knowledge of unicast routing and switching concepts
- Knowledge of BGP EVPN Concepts

Lab Manual

Lab Manual

There are several types of content in this manual. The following section is intended to show you how to use the manual.

Anything shown under a “Configuration sample” heading is not meant to be executed by the user. It is simply showing the configuration that was used to preconfigure that particular part of the lab. For example:

Configuration sample:

```
VXLAN-L11(config)# feature vn-segment-vlan-based
VXLAN-L11(config)# feature nv overlay
VXLAN-L11(config)# feature interface-vlan
```

Anything shown in a grey box is meant to be executed by the user. For example:

```
VXLAN-L13(config)# evpn
VXLAN-L13(config-evpn)# vni 42{{spod}}52 l2
VXLAN-L13(config-evpn-evi)# rd auto
VXLAN-L13(config-evpn-evi)# route-target import 2:42{{spod}}52
VXLAN-L13(config-evpn-evi)# route-target export 2:42{{spod}}52

VXLAN-L13(config)# show run interface vlan {{spod}}52

!Command: show running-config interface Vlan{{spod}}52
!Time: Sun May 10 00:47:53 2015

version 7.0(3)I1(3)

interface Vlan{{spod}}52 ← Anycast SVI
  no shutdown
  vrf member POD{{spod}}
  ip address 10.{{spod}}.52.1/24
  fabric forwarding mode anycast-gateway
<SNIP>
```

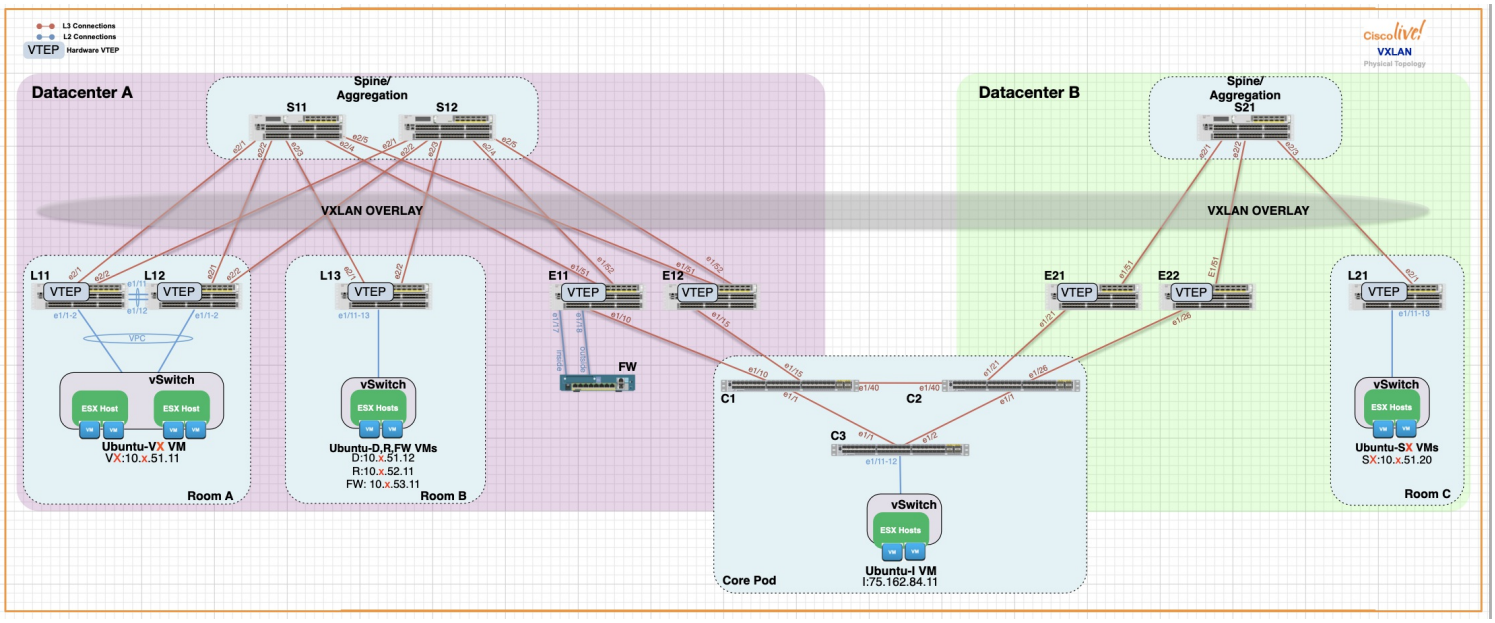
In the box above, you will notice 4 different colors:

- **Blue** text: This shows which commands students should execute on the NXOS command line.
- **Gold** text: Gives more details about console output text.
- Green highlight: This signifies something that we're calling out attention to. Usually, the green highlighting is showing you what you just configured in a show command output.
- Red highlight: This shows where some output was removed for brevity.

Lab Topology

Lab Topology

The full physical lab topology is shown below.



Lab Layout

Lab Layout

Hardware VXLAN Layout

Datacenter 1 / Pod 1

- OSPF area 0.0.0.0 is running throughout Datacenter 1 as the underlay protocol
- IBGP is configured for the overlay protocol. BGP EVPN is also used. BGP route reflectors are configured on the spine Nexus 9000s in Datacenter A
- IBGP is configured on the Nexus 9300 leaf switches for BGP EVPN. This provides a path for VXLAN control plane traffic. Ingress replication is used for BUM (Broadcast, Unknown Unicast and Multicast) traffic.
- Each Nexus 9300 switch will be configured as hardware VXLAN end point (VTEP).
- Each Nexus 9300 switch will be configured with VLANs connecting to ESX hosts for VM traffic.
- A firewall is connected to one of the Nexus 9300 edge switches.

Datacenter 2 / Pod 2

- EBGP is configured as the underlay protocol in Datacenter B.
- “next hop unchanged” and “retain route targets” are configured to extend the EVPN routes.
- Ingress replication is used for BUM (Broadcast, Unknown Unicast and Multicast) traffic.

> **Note!**
>
> To help student’s understanding of the implementation of VXLAN, we are keeping the lab design simple to keep our focus on the configuration and troubleshooting of VXLAN.

Lab Pods

Lab Pods

Each participant will be assigned one pod. Your pod is pod {{spod}}.

Each pod consists of (refer to the figure on the Lab Topology pages for Pod A/B/C definitions):

Hardware Based VXLAN

- One Linux Virtual Machine (VM) connected to Leaf L11/L12 designated as Ubuntu-VX (your pod's is Ubuntu-V{{spod}})
- One Linux Virtual Machine (VM) connected to Leaf L13 for Bridging designated as Ubuntu-D
- One Linux Virtual Machine (VM) connected to Leaf L13 for Routing designated as Ubuntu-R
- One Linux Virtual Machine (VM) connected to Leaf L13 for Routing with a Firewall designated as Ubuntu-FWX (your pod's is Ubuntu-FW{{spod}})
- One Linux Virtual Machine (VM) connected to Leaf L21 for Bridging and Routing in the VXLAN Multipod fabric designated as Ubuntu-SX (your pod's is Ubuntu-S{{spod}})
- One Linux Virtual Machine (VM) on the Internet/Outside designated as Ubuntu-I

Access to the Lab

Access to the Lab

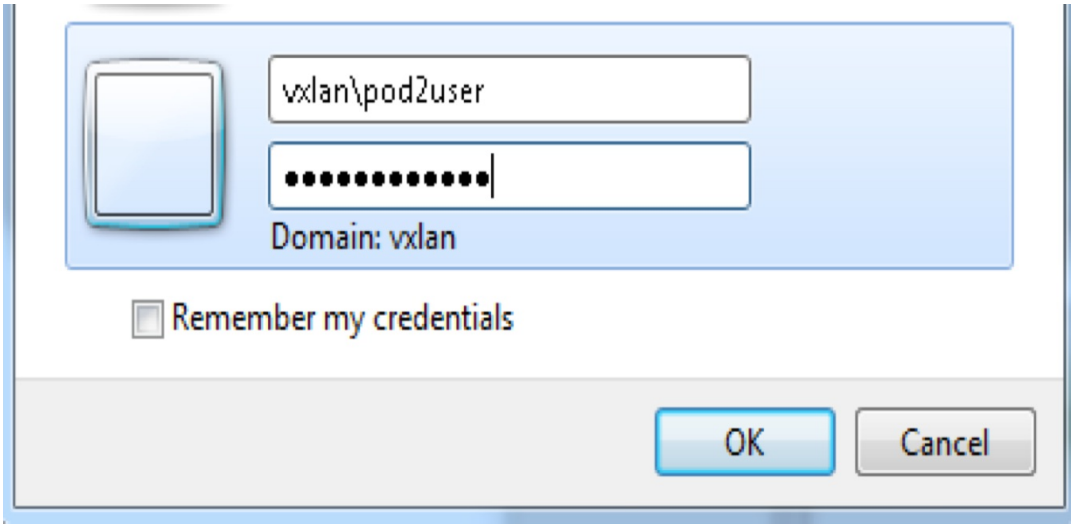
Step 1: Open Remote Desktop Connection from your computer
Step 2: For the Computer name field, enter: vxlanlab.ciscolive.com:3390

Step 3: Enter the appropriate username and password as shown below, and click OK. The username is based on your pod number.

> **Note!**
>
> The domain is vxlan and should be entered with the username as shown below.

> **Note!**
>
> In the screenshot below, pod 2 is used as an example. Change “pod2user” to your pod's username ("pod{{spod}}user").

Device	Username	Password
Remote Desktop	vxlan\pod{{spod}}user	given by Proctor



Step 4: If presented with a security warning click YES to accept the security certificate identity.

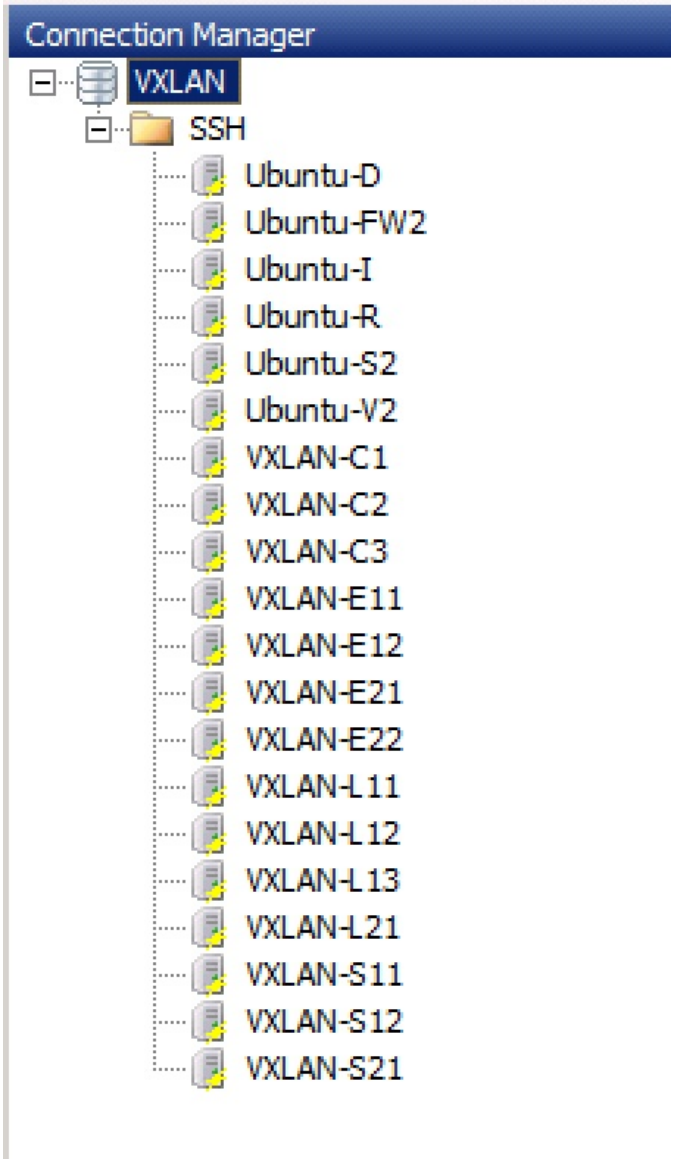
Step 5: After the login is complete, open a PuTTY connection by double-clicking on the puttycm icon on the desktop.

Step 6: In the Putty Connection Manager application, double-click on your switches to launch the appropriate SSH sessions. Putty should automatically log you into each switch/VM. If it does not login, credentials are shown below:

Device	Username	Password
Switches	pod{{spod}}user	given by Proctor
Virtual Machines	vxlan\pod{{spod}}user	given by Proctor

> **Note!**
>
> In the screenshot below, pod 2 is used as an example. You should see devices relevant to your pod (pod {{spod}}).

After you log in and launch PuttyCM, you should see the following list of devices:


Connection Manager
VXLAN
SSH
Ubuntu-D
Ubuntu-FW2
Ubuntu-I
Ubuntu-R
Ubuntu-S2
Ubuntu-V2
VXLAN-C1
VXLAN-C2
VXLAN-C3
VXLAN-E11
VXLAN-E12
VXLAN-E21
VXLAN-E22
VXLAN-L11
VXLAN-L12
VXLAN-L13
VXLAN-L21
VXLAN-S11
VXLAN-S12
VXLAN-S21

Use Case 1

Introduction

Use Case 1

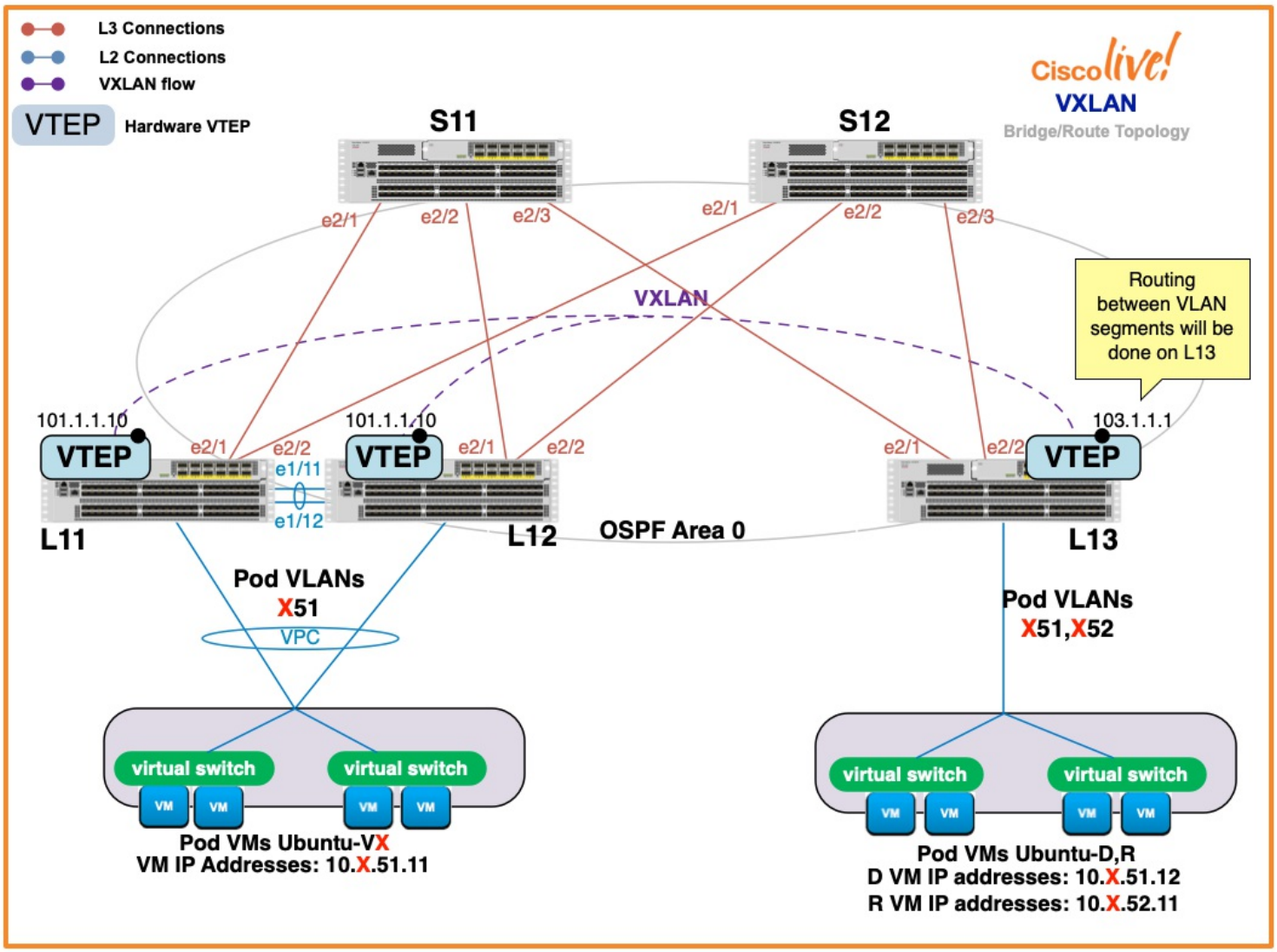
Datacenter 1 - Network Overlay (Hardware Based VTEPs) for VXLAN Bridging/Routing and Connecting to an Outside Network

The hardware based VXLAN solution in this lab is configured on shared Nexus 9300 switches (two in a VPC pair in Room A), a single leaf in Room B, and a pair of Border Leaf and Edge Routers for external Connectivity. Each Nexus 9300 switch in Room A and B is connected to end devices (VMs) on classic VLAN segments. All Nexus 9300 switches are connected over a layer 3 ECMP network via Nexus 9000 spine switches. Each Nexus 9300 switch will be configured as a VXLAN gateway, as well as a VXLAN router, and will connect the VLAN and VXLAN segments. In addition, the Border Leafs will be configured to allow communication with a device outside of the Datacenter (in the simulated “internet”).

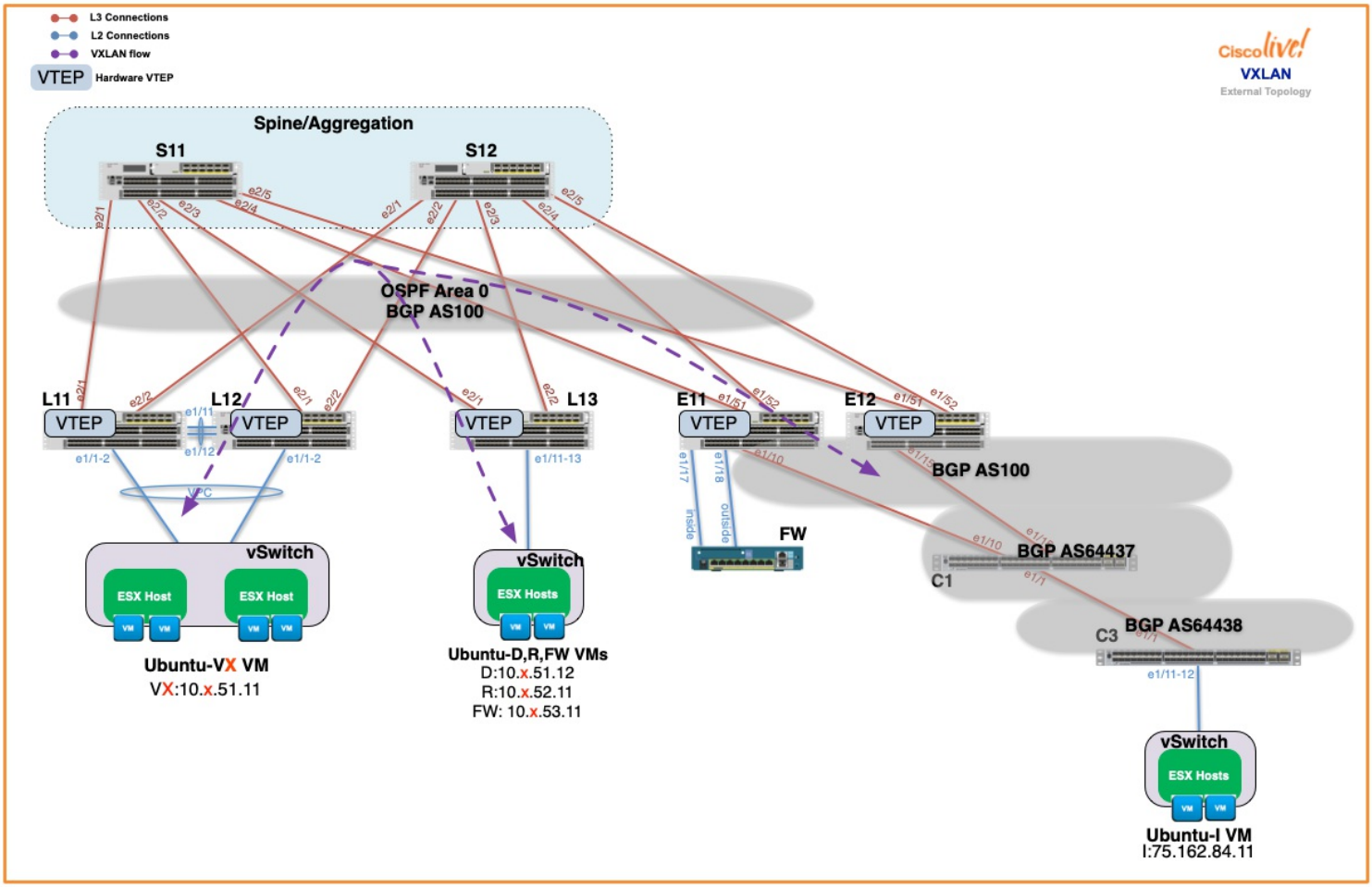
In this lab, IBGP is configured to provide BGP EVPN for VXLAN control plane packets. Instead of using multicast, we will be using ingress replication for the BUM (broadcast, unknown unicast, multicast) traffic.

The hardware based VXLAN routing/bridging solution lab is setup based on the following network topology (this is an excerpt of the master physical topology shown earlier – only a portion of

datacenter 1 is included):



The hardware based external VXLAN solution lab (a host in the datacenter communicating with a host outside of the datacenter [on the internet, for example]) is setup based on the following network topology (this is an excerpt of the master physical topology shown earlier – only datacenter A and the external/firewall portion is included):


L3 Connections
L2 Connections
VXLAN flow
VTEP Hardware VTEP
Cisco live!
VXLAN
External Topology
Spine/Aggregation
S11
S12
OSPF Area 0
BGP AS100
L11
L12
L13
E11
E12
VTEP
vPC
BGP AS100
FW
inside
outside
BGP AS64437
C1
C3
BGP AS64438
vSwitch
ESX Host
ESX Hosts
VM
Ubuntu-VX VM
VX:10.x.51.11
Ubuntu-D,R,FW VMs
D:10.x.51.12
R:10.x.52.11
FW: 10.x.53.11
Ubuntu-I VM
I:75.162.84.11

Base Network Configuration Verification

Use Case 1

Base Network Configuration Verification

> **Note!**
> All users are sharing hardware. Each switch can support only a single VTEP at this point. The underlay network is preconfigured but you will be able to verify the configuration of the entire underlay network before configuring VXLAN.

In this lab setup, the underlay network is pre-configured between the Nexus 9300 leaf switches and the Nexus 9300 spine switches. Prior to configuring the VXLAN components we will run basic verification of the OSPF underlay and BGP configurations.

Verify Network Configuration

The following section lists the preconfigured parts of the network we will verify to make sure the network is ready for VXLAN deployment.

1. Verify OSPF is configured as the underlay protocol.
2. Verify BGP is configured with Route Reflectors on the Spines. BGP will be used for VXLAN control plane.
3. Verify the VPC configuration on L11 and L12. Verify VPC is up by verifying the VPC peer link is up and the VPC port-channels are up.

Verify the Underlay Network

Follow the following steps to verify OSPF is configured correctly in the underlay and BGP is enabled as the overlay in the network. We will use leaf L11 and spine S11 for verification but the

rest of the leafs are configured identically.

Important!

Only execute commands shown in the boxes below. Any commands shown under “Configuration Sample” are just an example. They are not meant to be executed.

Step 1: Enable OSPF in the network. All of the leaf and edge switches are part of area 0.0.0.0.

Configuration sample:

```
VXLAN-L11(config)# feature ospf
VXLAN-L11(config)# router ospf 1
VXLAN-L11(config-router)# router-id 100.1.1.2
VXLAN-L11(config-router)# exit
VXLAN-L11(config)# interface ethernet 2/1
VXLAN-L11(config-if)# ip router ospf 1 area 0.0.0.0
VXLAN-L11(config-if)# interface ethernet 2/2
VXLAN-L11(config-if)# ip router ospf 1 area 0.0.0.0
VXLAN-L11(config-if)# interface loopback 0
VXLAN-L11(config-if)# ip router ospf 1 area 0.0.0.0
VXLAN-L11(config-if)# interface loopback 1
VXLAN-L11(config-if)# ip router ospf 1 area 0.0.0.0
VXLAN-L11(config-if)# interface loopback 3
VXLAN-L11(config-if)# ip router ospf 1 area 0.0.0.0
```

Step 2: Verify OSPF is enabled in the network.

```
VXLAN-L11# show run ospf

!Command: show running-config ospf
!Time: Mon May 28 20:46:08 2018

version 7.0(3)I4(7)
feature ospf

router ospf 1
  router-id 100.1.1.2

interface loopback0
  ip router ospf 1 area 0.0.0.0

interface loopback1
  ip router ospf 1 area 0.0.0.0

interface loopback3
  ip router ospf 1 area 0.0.0.0

interface Ethernet2/1
  ip router ospf 1 area 0.0.0.0

interface Ethernet2/2
  ip router ospf 1 area 0.0.0.0
```

```
VXLAN-S11# show run ospf

!Command: show running-config ospf
!Time: Mon May 28 20:46:08 2018

version 7.0(3)I4(7)
feature ospf

router ospf 1
  router-id 201.1.1.1

interface loopback3
  ip router ospf 1 area 0.0.0.0

interface loopback4
  ip router ospf 1 area 0.0.0.0

interface Ethernet2/1
  ip router ospf 1 area 0.0.0.0

interface Ethernet2/2
  ip router ospf 1 area 0.0.0.0

interface Ethernet2/3
  ip router ospf 1 area 0.0.0.0

interface Ethernet2/4
  ip router ospf 1 area 0.0.0.0

interface Ethernet2/5
  ip router ospf 1 area 0.0.0.0

interface Ethernet2/6
  ip router ospf 1 area 0.0.0.0
```

Step 3: Verify IBGP on each leaf. IBGP will be used to exchange new MAC addresses learned on each leaf to all of the other leafs. A neighbor statement must be added for each leaf in the topology and L2VPN EVPN must be enabled for each neighbor. In addition, a source interface of loopback3 is used to keep the IP schema consistent. The current lab design uses spine switches S11 and S12 as route reflectors to avoid having a full mesh IBGP peering.

Configuration sample:

```
VXLAN-L11(config)# router bgp 100
VXLAN-L11(config-router)# address-family ipv4 unicast
VXLAN-L11(config-router-af)# network 101.1.1.1/32
VXLAN-L11(config-router-af)# neighbor 200.200.200.2 remote-as 100
VXLAN-L11(config-router-neighbor)# update-source loopback 3
VXLAN-L11(config-router-neighbor)# address-family ipv4 unicast
VXLAN-L11(config-router-neighbor-af)# send-community both
VXLAN-L11(config-router-neighbor-af)# address-family l2vpn evpn
VXLAN-L11(config-router-neighbor-af)# send-community both
VXLAN-L11(config-router-neighbor-af)# neighbor 200.200.200.3 remote-as 100
VXLAN-L11(config-router-neighbor)# update-source loopback 3
VXLAN-L11(config-router-neighbor)# address-family ipv4 unicast
VXLAN-L11(config-router-neighbor-af)# send-community both
VXLAN-L11(config-router-neighbor-af)# address-family l2vpn evpn
VXLAN-L11(config-router-neighbor-af)# send-community both
```

Note!

The two IBGP neighbors are spine switches S11 and S12 configured as Route Reflectors with loopback IPs 200.200.200.2 and 200.200.200.3, respectively.

```
VXLAN-L11# show run bgp

!Command: show running-config bgp
!Time: Mon May 28 20:46:08 2018

version 7.0(3)I4(7)
feature bgp

router bgp 100
  address-family ipv4 unicast
    network 101.1.1.1/32
    network 101.1.1.10/32
  neighbor 200.200.200.2 remote-as 100
    update-source loopback3
    address-family ipv4 unicast
      send-community both
    address-family l2vpn evpn
      send-community both
  neighbor 200.200.200.3 remote-as 100
    update-source loopback3
    address-family ipv4 unicast
      send-community both
    address-family l2vpn evpn
      send-community both
```

```
VXLAN-S11# show run bgp

!Command: show running-config bgp
!Time: Mon May 28 20:46:08 2018

version 7.0(3)I4(7)
feature bgp

router bgp 100
  address-family ipv4 unicast
  template peer OVERLAY
    remote-as 100
    update-source loopback4
    address-family ipv4 unicast
      send-community both
      route-reflector-client
      soft-reconfiguration inbound always
    address-family l2vpn evpn
      send-community both
      route-reflector-client
  neighbor 101.1.1.3
    inherit peer OVERLAY
  neighbor 102.1.1.3
    inherit peer OVERLAY
  neighbor 103.1.1.3
    inherit peer OVERLAY
  neighbor 104.1.1.3
    inherit peer OVERLAY
  neighbor 105.1.1.3
    inherit peer OVERLAY
```

```
VXLAN-S12# show run bgp

!Command: show running-config bgp
!Time: Mon May 28 20:46:08 2018

version 7.0(3)I4(7)
feature bgp

router bgp 100
  address-family ipv4 unicast
  template peer OVERLAY
    remote-as 100
    update-source loopback4
    address-family ipv4 unicast
      send-community both
      route-reflector-client
      soft-reconfiguration inbound always
    address-family l2vpn evpn
      send-community both
      route-reflector-client
  neighbor 101.1.1.3
    inherit peer OVERLAY
  neighbor 102.1.1.3
    inherit peer OVERLAY
  neighbor 103.1.1.3
    inherit peer OVERLAY
  neighbor 104.1.1.3
    inherit peer OVERLAY
  neighbor 105.1.1.3
    inherit peer OVERLAY
```

Step 4: If you would like, verify the BGP and OSPF configurations on rest of the switches.

Verify VPC Configuration

Leaf L11 and Leaf L12 are setup in a VPC pair to provide redundancy. Prior to configuring the VXLAN components, we will run basic verification of the VPC configuration.

Important!

Only execute commands shown in the boxes below. Any commands shown under “Configuration Sample” are just an example. They are not meant to be executed.

This section will verify the VPC setup between Leaf L11 and Leaf L12.

Note!

When using VXLAN in a VPC environment, VPC peers must have identical VXLAN configuration.

Step 1: Verify the peer keep-alive is up on both switches and verify that the neighbor’s management address is that of the proper switch (Leaf L11 IP address is 192.168.11.21 and Leaf L12 IP address is 192.168.11.22).

```
VXLAN-L11# show vpc peer-keepalive

vPC keep-alive status           : peer is alive
--Peer is alive for             : (14122) seconds, (4) msec
--Send status                   : Success
--Last send at                  : 2018.05.08 02:16:54 975 ms
--Sent on interface             : mgmt0
--Receive status                : Success
--Last receive at               : 2018.05.08 02:16:54 975 ms
--Received on interface         : mgmt0
--Last update from peer         : (0) seconds, (422) msec

vPC Keep-alive parameters
--Destination                   : 192.168.11.22
--Keepalive interval            : 1000 msec
--Keepalive timeout             : 5 seconds
--Keepalive hold timeout        : 3 seconds
--Keepalive vrf                 : management
--Keepalive udp port            : 3200
--Keepalive tos                 : 192
```

```
VXLAN-L12# show vpc peer-keepalive

vPC keep-alive status           : peer is alive
--Peer is alive for             : (14237) seconds, (407) msec
--Send status                   : Success
--Last send at                  : 2018.05.08 02:23:49 228 ms
--Sent on interface             : mgmt0
--Receive status                : Success
--Last receive at               : 2018.05.08 02:23:49 228 ms
--Received on interface         : mgmt0
--Last update from peer         : (0) seconds, (250) msec

vPC Keep-alive parameters
--Destination                   : 192.168.11.21
--Keepalive interval            : 1000 msec
--Keepalive timeout             : 5 seconds
--Keepalive hold timeout        : 3 seconds
--Keepalive vrf                 : management
--Keepalive udp port            : 3200
--Keepalive tos                 : 192
```

Step 2: Verify that the VPC is up. Also verify that peer-link and the VPC's are up and allow the lab VLANs (X00 and X51 on the peer-link and X51 on the VPCs).

```
VXLAN-L11# show vpc brief
Legend:
                (*) - local vPC is down, forwarding via vPC peer-link

vPC domain id                     : 1
Peer status                       : peer adjacency formed ok
vPC keep-alive status             : peer is alive
Configuration consistency status  : success
Per-vlan consistency status       : success
Type-2 consistency status         : success
vPC role                          : primary
Number of vPCs configured         : 2
Peer Gateway                      : Enabled
Dual-active excluded VLANs        : -
Graceful Consistency Check        : Enabled
Auto-recovery status              : Disabled
Delay-restore status              : Timer is off.(timeout = 30s)
Delay-restore SVI status          : Timer is off.(timeout = 10s)

vPC Peer-link status
---------------------------------------------------------------------
id   Port   Status Active vlans
--   ----   ------ --------------------------------------------------
1    Po1    up     <SNIP>

vPC status
----------------------------------------------------------------------------
id   Port   Status Consistency Reason                     Active vlans
--   ----   ------ ----------- ------                     ------------
11   Po11   up     success     success                    <SNIP>
12   Po12   up     success     success                    <SNIP>
```

```
VXLAN-L12# show vpc brief
Legend:
                (*) - local vPC is down, forwarding via vPC peer-link

vPC domain id                     : 1
Peer status                       : peer adjacency formed ok
vPC keep-alive status             : peer is alive
Configuration consistency status  : success
Per-vlan consistency status       : success
Type-2 consistency status         : success
vPC role                          : primary
Number of vPCs configured         : 2
Peer Gateway                      : Enabled
Dual-active excluded VLANs        : -
Graceful Consistency Check        : Enabled
Auto-recovery status              : Disabled
Delay-restore status              : Timer is off.(timeout = 30s)
Delay-restore SVI status          : Timer is off.(timeout = 10s)

vPC Peer-link status
---------------------------------------------------------------------
id   Port   Status Active vlans
--   ----   ------ --------------------------------------------------
1    Po1    up     <SNIP>

vPC status
----------------------------------------------------------------------------
id   Port   Status Consistency Reason                     Active vlans
--   ----   ------ ----------- ------                     ------------
11   Po11   up     success     success                    <SNIP>
12   Po12   up     success     success                    <SNIP>
```

Note!

You might notice VPC inconsistency in the VPC output, ignore this at this point. Also there will be other students who might be on different speed or different part of the lab using their VLANs and can caused these messages.

Nexus 9300 VXLAN Configuration

Use Case 1

Nexus 9300 VXLAN Configuration

VXLAN configuration involves creating the hardware VTEP, mapping VLANs to VXLAN VNIDs, and configuring the VNIDs for ingress replication for broadcast, unknown unicast, and multicast traffic. In the following sections, we will:

Note!

Since the Nexus 9000s supports a single VTEP, the VTEP will be preconfigured in this lab. Participants will verify the preconfigured VTEP configuration and then add VLAN to VXLAN mappings for their pod.

1. Verify VXLAN features are enabled on the switches.
2. Verify the loopback 0 interface is configured. This will be source or destination IP address of the VXLAN tunnel. It has OSPF running to make sure it is reachable across the network.
3. Verify the VTEP interface (NVE [Network Virtualization Endpoint] interface) has been configured.
4. Verify the VTEP is configured to use the loopback 0 interface as the VTEP source interface.
5. Configure POD VRF.
6. Configure POD VLANs:
    - VLAN {{spod}}00 (for L3 VNI),
    - VLAN {{spod}}51 (for VXLAN Bridging traffic),
    - VLAN {{spod}}52 (to demonstrate VXLAN routing capabilities),
    - VLAN {{spod}}53/{{spod}}54 (to demonstrate VXLAN routing capabilities through a firewall).

7. Configure VLAN to VXLAN VNID mapping under VLAN configuration.
8. Configure data VXLAN VNIDs under NVE configuration for ingress replication BUM traffic.
9. Configure IBGP underlay. Add each spine as IBGP peers so that each host can provide updates to its neighbors directly.

> **Note!**
>
> In this portion of the lab, we used OSPF for the underlay network and IBGP for the overlay network. We could also use EBGP or IBGP full mesh if the spine hardware supports BGP with EVPN (for example, the Nexus 9000 and Nexus 7000 platforms).

10. Configure EVPN so that any L2 host MACs that are learned on Leaf L11 or Leaf L12 will be propagated to Leaf L13 and Leaf E11 and vice-versa.
11. Configure BGP to include EVPN POD VRF configuration to extend route reachability.

> **Note!**
>
> Configure interface VLANs for VLAN {{spod}}00, {{spod}}51 and {{spod}}52 and configure the anycast gateway across all the leafs.

12. Configure interface VLAN for VLAN {{spod}}54.

There are three features that must be enabled to configure the Nexus 9300 switches to act as a VXLAN gateway: the **vn-segment-vlan-based** feature, the **nv overlay** feature and the **interface-vlan** feature.

Verify VXLAN Configuration

Step 1: Verify VXLAN features are configured and enabled

Configuration sample:

```
VXLAN-L11(config)# feature vn-segment-vlan-based
VXLAN-L11(config)# feature nv overlay
VXLAN-L11(config)# feature interface-vlan
```

Step 2: Globally enable the VLAN/VXLAN routing capability

Configuration sample:

```
VXLAN-L11(config)# nv overlay evpn
```

> **Note!**
>
> Run the following commands on L11, L12, L13, E11, and E12. The output on all of them should be the same except for LACP and VPC which is required only on L11 and L12, as indicated on the lab topology.

```
VXLAN-L11# show running | include feature
feature ospf
feature bgp
feature interface-vlan
feature vn-segment-vlan-based ← Allows VLANs to be mapped to VXLAN VNIDs
feature lacp
feature vpc
feature nv overlay ← Allows creation of NVE interface.
VXLAN-L11# show feature | inc vnseg
vnseg_vlan             1          enabled
VXLAN-L11# show feature | inc nve
nve                    1          enabled
VXLAN-L11# show running | inc "overlay evpn"
nv overlay evpn ← Enables the EVPN capability for the NV Overlay.
```

Step 3: Verify loopback interfaces are created on leaf L11 and L12. The loopback is assigned to the NVE interface. Since we are using a VPC setup on L11 and L12, we need a single IP address for the NVE to bind to that is common between both switches. We will create this using an identical secondary IP address. All packets sent by the NVE will be sourced from the secondary loopback interface IP address to avoid duplication.

> **Note!**
>
> The secondary IP address is only needed if VPC is in use.

Configuration sample:

```
VXLAN-L11(config)# interface loopback0
VXLAN-L11(config-if)# description loopback for NVE
VXLAN-L11(config-if)# ip address 101.1.1.1/32
VXLAN-L11(config-if)# ip address 101.1.1.10/32 secondary
VXLAN-L11(config-if)# ip router ospf 100 area 0.0.0.0
```

```
VXLAN-L12(config)# interface loopback0
VXLAN-L12(config-if)# description loopback for NVE
VXLAN-L12(config-if)# ip address 102.1.1.1/32
VXLAN-L12(config-if)# ip address 101.1.1.10/32 secondary
VXLAN-L12(config-if)# ip router ospf 100 area 0.0.0.0
```

```
VXLAN-L11# show run interface loopback 0

!Command: show running-config interface loopback0
!Time: Mon May 28 21:01:41 2018

version 7.0(3)I4(7)

interface loopback0
  description loopback for NVE
  ip address 101.1.1.1/32
  ip address 101.1.1.10/32 secondary
  ip router ospf 1 area 0.0.0.0
```

```
VXLAN-L12# show run interface loopback 0

!Command: show running-config interface loopback0
!Time: Mon May 28 21:01:41 2018

version 7.0(3)I4(7)

interface loopback0
  description loopback for NVE
  ip address 102.1.1.1/32
  ip address 101.1.1.10/32 secondary
  ip router ospf 1 area 0.0.0.0
```

Step 4: Verify the loopback interfaces are created on L13, E11 and E12.

```
VLAN-L13# show run interface loopback 0

!Command: show running-config interface loopback0
!Time: Mon May 28 21:01:41 2018

version 7.0(3)I4(7)

interface loopback0
  description loopback for NVE
  ip address 103.1.1.1/32
  ip router ospf 1 area 0.0.0.0
```

```
VXLAN-E11# show run interface loopback 0

!Command: show running-config interface loopback0
!Time: Mon May 28 21:01:41 2018

version 7.0(3)I4(7)

interface loopback0
  description loopback for NVE
  ip address 104.1.1.1/32
  ip router ospf 1 area 0.0.0.0
```

```
VXLAN-E12# show run interface loopback 0

!Command: show running-config interface loopback0
!Time: Mon May 28 21:01:41 2018

version 7.0(3)I4(7)

interface loopback0
  description loopback for NVE
  ip address 105.1.1.1/32
  ip router ospf 1 area 0.0.0.0
```

Step 5: Verify VTEP (NVE) configuration on the leafs. The NVE interface configuration includes a source command to identify the source loopback interface, the mode of the VTEP (NVE) (in this case is the BGP EVPN) for host learning, and a number of member commands that map VXLAN VNIDs to multicast groups and configure any specific features used. The member configuration will be done later in this module. In this step we are verifying that the NVE has been created and is configured with the correct source loopback interface.

> **Note!**
>
> There may be some additional configuration present in the following show commands if other students have already moved past this point and started to configure their PODs.

> **Important!**
>
> Run the commands in the following two steps on L11, L12 and L13. The output on all of them should be the same.

Configuration sample:

```
VXLAN-L11(config)# interface nve 1
VXLAN-L11(config-if-nve)# source-interface loopback 0
VXLAN-L11(config-if-nve)# host-reachability protocol bgp
```

> **Note!**
>
> Note: The “host-reachability protocol bgp” command is required in order to use BGP EVPN control plane for VXLAN traffic.

```
VXLAN-L11(config)# show run interface nve 1

!Command: show running-config interface nve1
!Time: Mon May 28 21:01:41 2018

version 7.0(3)I4(7)

interface nve1
  no shutdown
  source-interface loopback0
  host-reachability protocol bgp
```

VXLAN Bridging and Routing Configuration

Each POD has four Linux hosts that will be used to test the hardware VXLAN configuration.

One host (Ubuntu-V{{spod}}) is connected to a VLAN{{spod}}51 on the L11/L12 VPC pair. One host (Ubuntu-D) is connected to a VLAN{{spod}}51 on L13. Both of these VMs will communicate on the same VXLAN.

The third host (Ubuntu-R) is connected to a different VLAN{{spod}}52 on L13. Ubuntu-V{{spod}} and Ubuntu-R will communicate across different VXLANs using VXLAN routing anycast gateway.

In addition, Ubuntu-I resides outside the datacenter, connected to C3.

You will be adding configuration to map the VLAN to a VXLAN VNI as well as configuring the anycast gateway functionality across the leafs. The next steps will be to map the POD data (L2) VNI, under the NVE configuration, for your POD and create three VLANs: VLAN {{spod}}00 for BGP EVPN control plane traffic and two data plane VLANs/VXLANs: {{spod}}51 for intra-VLAN traffic (which Ubuntu-V{{spod}} and Ubuntu-D will reside on) and {{spod}}52 for inter-VLAN routing (which Ubuntu-R will reside on). Lastly, you will configure the border leaf switches to connect the datacenter to an outside network.

Note!

There may be some additional configuration present in the following show commands if other students have already moved past this point and started to configure their PODs.

Important!

Issue the commands in the following steps on L11, L12, and L13.

In this module, we are demonstrating how to configure the network for VXLAN bridging and routing.

Step 1: Open putty sessions to your POD VMs and verify you are unable to ping between hosts across the pods.

Ubuntu-V{{spod}}

```
VXLAN\POD{{spod}}user@ubuntu-v{{spod}}:~$ ping 10.{{spod}}.51.12 ← ping host Ubuntu-D
PING 10.{{spod}}.51.12 (10.{{spod}}.51.12) 56(84) bytes of data.
From 10.{{spod}}.51.11 icmp_seq=1 Destination Host Unreachable
From 10.{{spod}}.51.11 icmp_seq=2 Destination Host Unreachable
From 10.{{spod}}.51.11 icmp_seq=3 Destination Host Unreachable
^C
--- 10.{{spod}}.51.12 ping statistics ---
4 packets transmitted, 0 received, +3 errors, 100% packet loss, time 3005ms

VXLAN\POD{{spod}}user@ubuntu-v{{spod}}:~$ ping 10.{{spod}}.52.11 ← ping host Ubuntu-R
PING 10.{{spod}}.52.11 (10.{{spod}}.52.11) 56(84) bytes of data.
From 10.{{spod}}.51.11 icmp_seq=1 Destination Host Unreachable
From 10.{{spod}}.51.11 icmp_seq=2 Destination Host Unreachable
From 10.{{spod}}.51.11 icmp_seq=3 Destination Host Unreachable
^C
--- 10.{{spod}}.52.11 ping statistics ---
4 packets transmitted, 0 received, +3 errors, 100% packet loss, time 3001ms
```

Step 2: Add VLAN to VNID mapping on each leaf (Note: since Leaf L11 and Leaf L12 are VPC devices, the configuration has to be identical across both devices. However, this configuration does not have to match on L13). The value specified with the **vn-segment** keyword is the VNID value sent in the VXLAN header.

VLAN numbers will be: {{spod}}00, {{spod}}51 and will map to VXLAN VNIs: 4{{spod}}00, 42{{spod}}51 respectively.

```
VXLAN-L11(config)# vlan {{spod}}00
VXLAN-L11(config-vlan)# vn-segment 4{{spod}}00
VXLAN-L11(config-vlan)# vlan {{spod}}51
VXLAN-L11(config-vlan)# vn-segment 42{{spod}}51
VXLAN-L11(config-vlan)# exit

VXLAN-L11# show run vlan {{spod}}00,{{spod}}51

!Command: show running-config vlan {{spod}}00, {{spod}}51
!Time: Mon May 28 21:01:41 2018

version 7.0(3)I4(7)
vlan {{spod}}00,{{spod}}51
vlan {{spod}}00
  vn-segment 4{{spod}}00
vlan {{spod}}51
  vn-segment 42{{spod}}51
```

Step 3: For each VNID used for control plane traffic ({{spod}}00), associate the POD VRF. Under each member data VNI, configure ARP suppression. Configure the VNID for ingress replication on each leaf.

Note!

ARP suppression is highly recommended. It will help to minimize the amount of flooding across the VXLAN fabric. This link provides more information regarding ARP suppression.

Note!

Ingress Replication will help customers who don't want to enable multicast in their network. Ingress replication uses unicast instead of multicast for BUM traffic.

```
VXLAN-L11(config)# interface nve 1
VXLAN-L11(config-if-nve)# member vni 4{{spod}}00 associate-vrf
VXLAN-L11(config-if-nve)# member vni 42{{spod}}51
VXLAN-L11(config-if-nve-vni)# ingress-replication protocol bgp
VXLAN-L11(config-if-nve-vni)# suppress-arp

VXLAN-L11(config-if-nve)# show run interface nve 1
!Command: show running-config interface nve1
!Time: Mon May 28 21:01:41 2018

version 7.0(3)I4(7)

interface nve1
  no shutdown
  source-interface loopback0
  host-reachability protocol bgp
  member vni 4{{spod}}00 associate-vrf
  member vni 42{{spod}}51
     suppress-arp
     ingress-replication protocol bgp
```

Step 4: Create a VRF for the POD, associate the control plane VLAN {{spod}}00 VNID to the VRF on each leaf and specify the EVPN session used with route targets. This is only needed for L3 VNI that is needed for VXLAN routing.

```
VXLAN-L11(config)# vrf context POD{{spod}}
VXLAN-L11(config-vrf)# vni 4{{spod}}00
VXLAN-L11(config-vrf)# rd auto
VXLAN-L11(config-vrf)# address-family ipv4 unicast
VXLAN-L11(config-vrf-af-ipv4)# route-target import 65000:{{spod}}
VXLAN-L11(config-vrf-af-ipv4)# route-target import 65000:{{spod}} evpn
VXLAN-L11(config-vrf-af-ipv4)# route-target export 65000:{{spod}}
VXLAN-L11(config-vrf-af-ipv4)# route-target export 65000:{{spod}} evpn

VXLAN-L11(config-vrf-af-ipv4)# show run vrf POD{{spod}}
!Command: show running-config vrf POD{{spod}}
!Time: Mon May 28 21:01:41 2018

version 7.0(3)I4(7)

<SNIP>
vrf context POD{{spod}}
   vni 4{{spod}}00
   rd auto
   address-family ipv4 unicast
     route-target import 65000:{{spod}}
     route-target import 65000:{{spod}} evpn
     route-target export 65000:{{spod}}
     route-target export 65000:{{spod}} evpn
```

Step 5: Import/export the data VLAN VNIDs (VLAN {{spod}}51) into/out of EVPN for host MAC learning.

```
VXLAN-L11(config)# evpn
VXLAN-L11(config-evpn)# vni 42{{spod}}51 l2
VXLAN-L11(config-evpn-evi)# rd auto
VXLAN-L11(config-evpn-evi)# route-target import {{spod}}:42{{spod}}51
VXLAN-L11(config-evpn-evi)# route-target export {{spod}}:42{{spod}}51
```

Step 6: Configure BGP to include your POD's VRF so that any MACs learned on the local switch will be propagated to the other VTEPs and their respective switches.

```
VXLAN-L11(config)# router bgp 100
VXLAN-L11(config-router)# vrf POD{{spod}}
VXLAN-L11(config-router-vrf)# address-family ipv4 unicast
VXLAN-L11(config-router-vrf-af)# advertise l2vpn evpn
VXLAN-L11(config-router-vrf-af)# show run bgp

!Command: show running-config bgp
!Time: Mon May 28 21:01:41 2018

version 7.0(3)I4(7)
feature bgp

router bgp 100
  address-family ipv4 unicast
    network 101.1.1.1/32
    network 101.1.1.10/32
  neighbor 200.200.200.2 remote-as 100
    update-source loopback3
    address-family ipv4 unicast
      send-community both
    address-family l2vpn evpn
      send-community both
  neighbor 200.200.200.3 remote-as 100
    update-source loopback3
    address-family ipv4 unicast
      send-community both
    address-family l2vpn evpn
      send-community both
  vrf POD{{spod}}
    address-family ipv4 unicast
      advertise l2vpn evpn
<SNIP>
evpn
vni 42{{spod}}51 l2
     rd auto
     route-target import {{spod}}:42{{spod}}51
     route-target export {{spod}}:42{{spod}}51
<SNIP>
```

Step 7: Configure the interface VLANs (SVI) for the control and data VLANs to enable VXLAN routing using an anycast gateway. Make sure the IP of the gateway is the same across all the leafs to provide the anycast gateway functionality.

> ## Note!
>
> In order for anycast gateway functionality to work, the MAC address must be specified and must be the same across all three switches. This is accomplished using the “fabric forwarding anycast-gateway-mac 0002.0002.0002” command. This command has already been preconfigured for you.

Note!

Once you configure L11, you will notice VPC inconsistency in the VPC output, ignore this at this point. Also there will be other students who might be on different speed or different part of the lab using their VLANs and can cause these messages.

```
VXLAN-L11(config)# interface vlan {{spod}}00
VXLAN-L11(config-if)# vrf member POD{{spod}}
VXLAN-L11(config-if)# ip forward
VXLAN-L11(config-if)# no shutdown
VXLAN-L11(config-if)# interface Vlan{{spod}}51
VXLAN-L11(config-if)# vrf member POD{{spod}}
VXLAN-L11(config-if)# ip address 10.{{spod}}.51.1/24
VXLAN-L11(config-if)# no shutdown
VXLAN-L11(config-if)# fabric forwarding mode anycast-gateway
VXLAN-L11(config-if)# show run int vlan{{spod}}00,vlan{{spod}}51

!Command: show running-config interface Vlan{{spod}}00, Vlan{{spod}}51
!Time: Mon May 28 21:01:41 2018

version 7.0(3)I4(7)
interface Vlan{{spod}}00
  no shutdown
  vrf member POD{{spod}}

interface Vlan{{spod}}51
  no shutdown
  vrf member POD{{spod}}
  ip address 10.{{spod}}.51.1/24
  fabric forwarding mode anycast-gateway
```

Step 8: You should have configured all of the above commands on L11, L12, and L13. If you haven't configured all three switches, there is a file in the folder entitled "POD{{spod}}_Configs" on your desktop called "L11_L12_L13_pod{{spod}}". Copy paste all of the configuration in that file into any device that you didn't configure. This configuration is identical to that of L11 as overlay configuration for configuring VLANs {{spod}}50 and {{spod}}51 will be same across all the switches.

Important!

It is vital that you do not skip this step!

Step 9: Configure Leaf L13 with VLAN {{spod}}52 / VXLAN VNI 42{{spod}}52. Ubuntu-R is a member of this VLAN. In order for Ubuntu-V{{spod}} to communicate with Ubuntu-R, traffic must be routed from VXLAN 42{{spod}}51 to 42{{spod}}52.

Important!

Up until this point, leaf L13 configuration has been the same as L11 and L12. Only the below configuration for VLAN{{spod}}52 / VNI 52{{spod}}52 is different. **Configure this step only on L13.**

```
VXLAN-L13(config)# vlan {{spod}}52
VXLAN-L13(config-vlan)# vn-segment 42{{spod}}52

VXLAN-L13(config)# interface vlan {{spod}}52
VXLAN-L13(config-if)# no shutdown
VXLAN-L13(config-if)# vrf member POD{{spod}}
VXLAN-L13(config-if)# ip address 10.{{spod}}.52.1/24
VXLAN-L13(config-if)# fabric forwarding mode anycast-gateway

VXLAN-L13(config)# int nve1
VXLAN-L13(config-if-nve)# member vni 42{{spod}}52
VXLAN-L13(config-if-nve-vni)# ingress-replication protocol bgp
VXLAN-L13(config-if-nve-vni)# suppress-arp

VXLAN-L13(config)# evpn
VXLAN-L13(config-evpn)# vni 42{{spod}}52 l2
VXLAN-L13(config-evpn-evi)# rd auto
VXLAN-L13(config-evpn-evi)# route-target import {{spod}}:42{{spod}}52
VXLAN-L13(config-evpn-evi)# route-target export {{spod}}:42{{spod}}52

VXLAN-L13(config)# show run interface vlan {{spod}}52

!Command: show running-config interface Vlan{{spod}}52
!Time: Mon May 28 21:01:41 2018

version 7.0(3)I4(7)

interface Vlan{{spod}}52
  no shutdown
  vrf member POD{{spod}}
  ip address 10.{{spod}}.52.1/24
  fabric forwarding mode anycast-gateway
```

Step 10: Verify your VNI is in an up state under the NVE on each of the leafs.

```
VXLAN-L11# show nve vni
Codes: CP - Control Plane        DP - Data Plane
       UC - Unconfigured         SA - Suppress ARP

Interface VNI      Multicast-group   State Mode Type [BD/VRF]      Flags
--------- -------- ----------------- ----- ---- ------------------ -----
nve1      4{{spod}}00     n/a               Up    CP   L3 [POD{{spod}}]
nve1      42{{spod}}51    Unicast BGP       Up    CP   L2 [{{spod}}51]           SA
```

```
VXLAN-L12# show nve vni
Codes: CP - Control Plane        DP - Data Plane
       UC - Unconfigured         SA - Suppress ARP

Interface VNI      Multicast-group   State Mode Type [BD/VRF]      Flags
--------- -------- ----------------- ----- ---- ------------------ -----
nve1      4{{spod}}00     n/a               Up    CP   L3 [POD{{spod}}]
nve1      42{{spod}}51    Unicast BGP       Up    CP   L2 [{{spod}}51]           SA
```

```
VXLAN-L13# sh nve vni
Codes: CP - Control Plane        DP - Data Plane
       UC - Unconfigured         SA - Suppress ARP

Interface VNI      Multicast-group   State Mode Type [BD/VRF]      Flags
--------- -------- ----------------- ----- ---- ------------------ -----
nve1      4{{spod}}00    n/a               Up   CP   L3 [POD{{spod}}]
nve1      42{{spod}}51   UnicastBGP        Up   CP   L2 [{{spod}}51]            SA
nve1      42{{spod}}52   UnicastBGP        Up   CP   L2 [{{spod}}52]            SA
```

Step 11: Use ping to test connectivity between your POD VMs (V{{spod}} and D).

```
VXLAN\POD{{spod}}user@ubuntu-v{{spod}}:~$ ping 10.{{spod}}.51.12
PING 10.{{spod}}.51.12 (10.{{spod}}.51.12) 56(84) bytes of data.
64 bytes from 10.{{spod}}.51.12: icmp_req=1 ttl=64 time=0.410 ms
64 bytes from 10.{{spod}}.51.12: icmp_req=2 ttl=64 time=0.352 ms
64 bytes from 10.{{spod}}.51.12: icmp_req=3 ttl=64 time=0.379 ms
64 bytes from 10.{{spod}}.51.12: icmp_req=4 ttl=64 time=0.300 ms
^C
--- 10.{{spod}}.51.12 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3000ms
```

Step 12: Use ping to test connectivity between your POD VMs (V{{spod}} and R). Note that this ping is being routed between VXLANs.

```
VXLAN\POD{{spod}}user@ubuntu-v{{spod}}:~$ ping 10.{{spod}}.52.11
PING 10.{{spod}}.52.11 (10.{{spod}}.52.11) 56(84) bytes of data.
64 bytes from 10.{{spod}}.52.11: icmp_req=1 ttl=62 time=2.46 ms
64 bytes from 10.{{spod}}.52.11: icmp_req=2 ttl=62 time=0.424 ms
64 bytes from 10.{{spod}}.52.11: icmp_req=3 ttl=62 time=0.471 ms
64 bytes from 10.{{spod}}.52.11: icmp_req=4 ttl=62 time=0.513 ms
^C
--- 10.{{spod}}.52.11 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3001ms
```

Step 13: Check the MAC address for the Ubuntu-V{{spod}} host.

> **Note!**
>
> In the following steps, the MAC addresses shown are for reference only! They may differ for your pod.

```
VXLAN\POD{{spod}}user@ubuntu-v{{spod}}:~$ ifconfig eth1
eth1      Link encap:Ethernet  HWaddr 00:50:56:82:ad:66
          inet addr:10.{{spod}}.51.11  Bcast:10.{{spod}}.51.255  Mask:255.255.255.0
          inet6 addr: fe80::250:56ff:fe82:ad66/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3441 errors:0 dropped:0 overruns:0 frame:0
          TX packets:610049 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:345064 (345.0 KB)  TX bytes:26742917 (26.7 MB)
```

Step 14: Check the MAC address for the Ubuntu-D host.

```
VXLAN\POD{{spod}}user@ubuntu-d:~$ ifconfig eth1.{{spod}}51
eth1.251  Link encap:Ethernet  HWaddr 00:50:56:82:cf:76
          inet addr:10.{{spod}}.51.12  Bcast:10.{{spod}}.51.255  Mask:255.255.255.0
          inet6 addr: fe80::250:56ff:fe82:cf76/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1723 errors:0 dropped:15 overruns:0 frame:0
          TX packets:3777 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:166252 (166.2 KB)  TX bytes:751986 (751.9 KB)
```

Step 15: Check the MAC address for the Ubuntu-R host.

```
VXLAN\POD{{spod}}user@ubuntu-r:~$ ifconfig eth1.{{spod}}52
eth1.252  Link encap:Ethernet  HWaddr 00:50:56:82:c4:fc
          inet addr:10.{{spod}}.52.11  Bcast:10.{{spod}}.52.255  Mask:255.255.255.0
          inet6 addr: fe80::250:56ff:fe82:c4fc/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1287 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4155 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:94390 (94.3 KB)  TX bytes:826697 (826.6 KB)
```

Step 16: Check the MAC address table on Leaf L11 and L12

> ## Note!
>
> The MAC address-table on each N9K switch should show entries for your hosts. One entry will be learned via the L2 trunk port, from the host connected to this switch, and the other entry will be via the NVE interface, from hosts reached over the VTEP.

```
VXLAN-L11# show mac address-table vlan {{spod}}51
Legend:
        * - primary entry, G - Gateway MAC, (R) - Routed MAC, O - Overlay MAC
        age - seconds since last seen,+ - primary entry using vPC Peer-Link,
        (T) - True, (F) - False, C - ControlPlane MAC
   VLAN     MAC Address      Type      age     Secure NTFY Ports
---------+-----------------+--------+---------+------+----+------------------
+  {{spod}}51     0050.5682.ad66   dynamic  0         F      F    Po11 ← V{{spod}} host le
C  {{spod}}51     0050.5682.cf76   dynamic  0         F      F    nve1(103.1.1.1) ← D host
G  {{spod}}51     f8c2.8887.d36f   static   -         F      F    vPC Peer-Link(R)   over
G  {{spod}}51     f8c2.8887.d3df   static   -         F      F    sup-eth1(R)
```

```
VXLAN-L12# show mac address-table vlan {{spod}}51
Legend:
        * - primary entry, G - Gateway MAC, (R) - Routed MAC, O - Overlay MAC
        age - seconds since last seen,+ - primary entry using vPC Peer-Link,
        (T) - True, (F) - False, C - ControlPlane MAC
   VLAN     MAC Address      Type      age     Secure NTFY Ports
---------+-----------------+--------+---------+------+----+------------------
+  {{spod}}51     0050.5682.ad66   dynamic  0         F      F    Po11 ← V{{spod}} host on
C  {{spod}}51     0050.5682.cf76   dynamic  0         F      F    nve1(103.1.1.1) ← D host
G  {{spod}}51     f8c2.8887.d36f   static   -         F      F    sup-eth1(R)        over
G  {{spod}}51     f8c2.8887.d3df   static   -         F      F    vPC Peer-Link(R)
```

Step 17: Check the MAC address table on Leaf L13

```
VLAN-L13# show mac address-table vlan {{spod}}51
Legend:
        * - primary entry, G - Gateway MAC, (R) - Routed MAC, O - Overlay MAC
        age - seconds since last seen,+ - primary entry using vPC Peer-Link,
        (T) - True, (F) - False, C - ControlPlane MAC
   VLAN     MAC Address      Type      age     Secure NTFY Ports
---------+-----------------+--------+---------+------+----+------------------
*  {{spod}}51     0050.5682.cf76   dynamic  0         F      F    Eth1/13 ← D host on loca
C  {{spod}}51     0050.5682.ad66   dynamic  0         F      F    nve1(101.1.1.10)← V{{spo
G  {{spod}}51     58f3.9ca3.4603   static   -         F      F    sup-eth1(R)        over

VLAN-L13# show mac address-table vlan {{spod}}52
Legend:
        * - primary entry, G - Gateway MAC, (R) - Routed MAC, O - Overlay MAC
        age - seconds since last seen,+ - primary entry using vPC Peer-Link,
        (T) - True, (F) - False, C - ControlPlane MAC
   VLAN     MAC Address      Type      age     Secure NTFY Ports
---------+-----------------+--------+---------+------+----+------------------
*  {{spod}}52     0050.5682.c4fc   dynamic  0         F      F    Eth1/1 ← R host on local
G  {{spod}}52     58f3.9ca3.4603   static   -         F      F    sup-eth1(R)
```

Step 18: Verify the L2 routing for the EVPN routes for your POD VNIs.

```
VXLAN-L11# show l2route evpn mac-ip all
Topology ID Mac Address    Prod Host IP                                 Next Hop (s)
----------- -------------- ---- --------------------------------------- --------------
{{spod}}51         0050.5682.ad66 HMM  10.{{spod}}.51.11                            N/A
{{spod}}51         0050.5682.cf76 BGP  10.{{spod}}.51.12                            103.
```

```
VXLAN-L12# show l2route evpn mac-ip all
Topology ID Mac Address    Prod Host IP                                 Next Hop (s)
----------- -------------- ---- --------------------------------------- --------------
{{spod}}51         0050.5682.ad66 HMM  10.{{spod}}.51.11                            N/A
{{spod}}51         0050.5682.cf76 BGP  10.{{spod}}.51.12                            103.
```

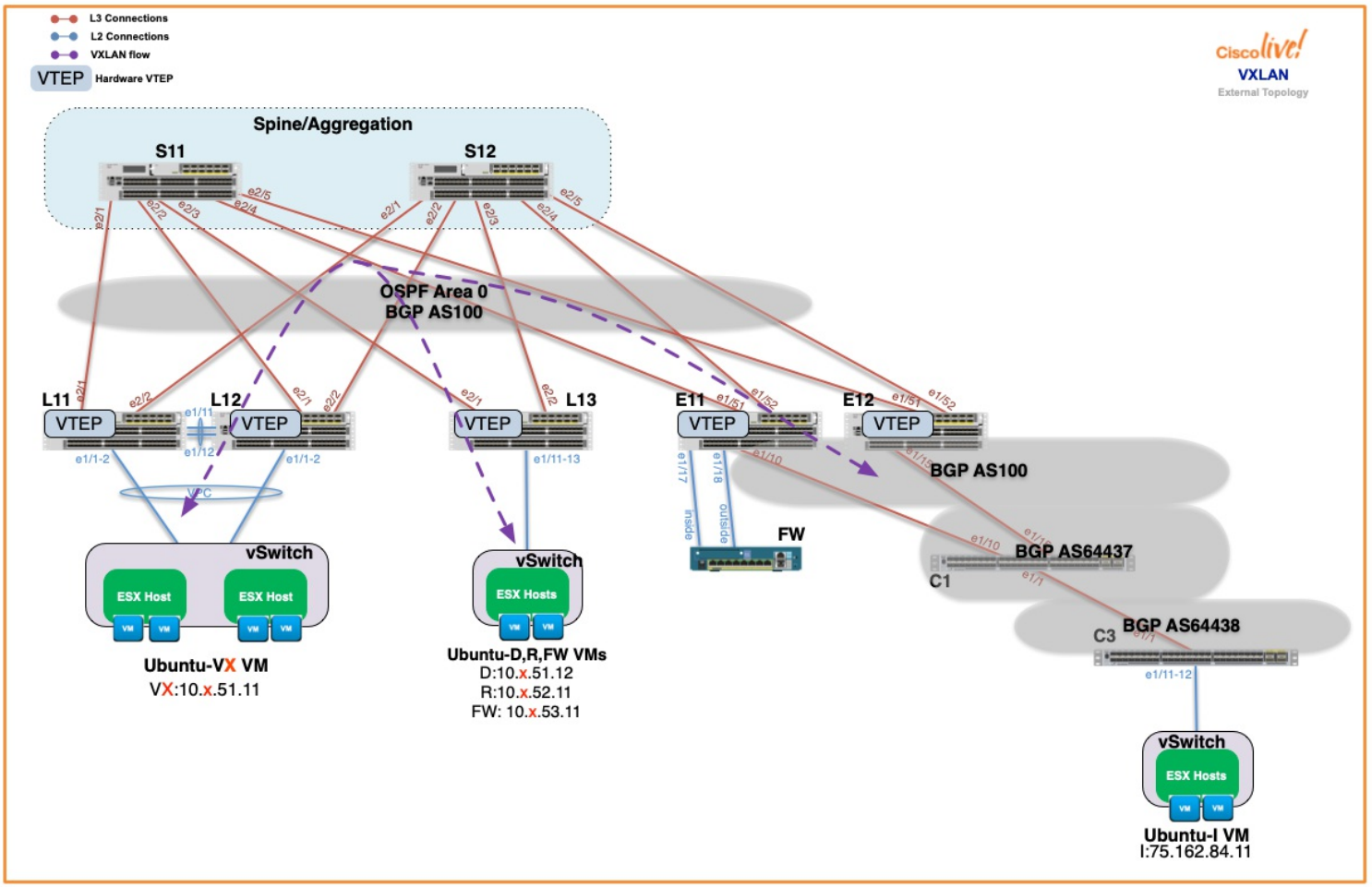
```
VLAN-L13# show l2route evpn mac-ip all
Topology ID Mac Address    Prod Host IP                                 Next Hop (s)
----------- -------------- ---- --------------------------------------- --------------
{{spod}}51         0050.5682.cf76 HMM  10.{{spod}}.51.12                            N/A
{{spod}}52         0050.5682.c4fc HMM  10.{{spod}}.52.11                            N/A
{{spod}}51         0050.5682.ad66 BGP  10.{{spod}}.51.11                            101.
                                                                                    loop
```

Border Leaf Configuration

Use Case 1

Border Leaf Configuration – External Network Communication

This module is based off of the topology shown below. It demonstrates the VXLAN host's connectivity with external networks through the Border Leaf / Edge Switches, E11 and E12.



One host (Ubuntu-V{{spod}}) is connected to a VLAN on the L11/L12 VPC pair and the external “internet” host (Ubuntu-I) is connected to Internet core router C3. To facilitate this communication, the Border Leaf switches (referred to from here on as edge switches), E11 and E12 need to be configured with VTEPs. The VXLAN traffic will terminate at E11 and E12 and then regular (non-VXLAN) traffic for each of the PODs will be sent through core C1 on the respective customer VRFs and then eventually routed to C3 through BGP.

Step 1: Open putty sessions to your POD VMs and verify you are unable to ping from Ubuntu V{{spod}} to Ubuntu I

Ubuntu-V{{spod}}

```
VXLAN\POD{{spod}}user@ubuntu-v{{spod}}:~$ ping 75.162.84.11 ← ping external host Ubuntu-I
PING 75.162.84.11 (75.162.84.11) 56(84) bytes of data.
^C
--- 75.162.84.11 ping statistics ---
6 packets transmitted, 0 received, 100% packet loss, time 5040ms
```

Step 2: Verify there is no route from each Leaf to the “Internet” VM in your POD’s VRF.

```
VXLAN-L11# show ip route 75.162.84.11 vrf POD{{spod}}
IP Route Table for VRF POD{{spod}}"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>

Route not found
```

Step 3: Add VLAN to VNID mapping for the L3 VNI on E11. The VLAN number will be {{spod}}00 and will map to VXLAN VNI 4{{spod}}00.

> **Note!**
>
> There are two edge switches (E11 and E12). To save time, you will only issue the below commands on E11. Then, you will copy paste the entire set of commands for E12 on to E12 using the provided configuration file (this file is located on your desktop).

```
VXLAN-E11(config)# vlan {{spod}}00
VXLAN-E11(config-vlan)# vn-segment 4{{spod}}00
VXLAN-E11(config-vlan)# exit

VXLAN-E11(config)# show run vlan {{spod}}00

!Command: show running-config vlan {{spod}}00
!Time: Mon May 28 21:35:38 2018

version 7.0(3)I4(7)
vlan {{spod}}00
vlan {{spod}}00
  vn-segment 4{{spod}}00
```

Step 4: On the NVE configuration for E11, associate the POD VRF to the the Layer 3 VNID used for control plane traffic ({{spod}}00).

```
VXLAN-E11(config)# interface nve1
VXLAN-E11(config-if-nve)# member vni 4{{spod}}00 associate-vrf
VXLAN-E11(config-if-nve)# show run interface nve1

!Command: show running-config interface nve1
!Time: Mon May 28 21:35:38 2018

version 7.0(3)I4(7)

interface nve1
  no shutdown
  source-interface loopback0
  host-reachability protocol bgp
  member vni 4{{spod}}00 associate-vrf
```

Step 5: Create the VRF for the POD, associate the control plane VLAN {{spod}}00 VNID to the VRF and specify the EVPN session used (auto selection).

```
VXLAN-E11(config)# vrf context POD{{spod}}
VXLAN-E11(config-vrf)# vni 4{{spod}}00
VXLAN-E11(config-vrf)# rd auto
VXLAN-E11(config-vrf)# address-family ipv4 unicast
VXLAN-E11(config-vrf-af-ipv4)# route-target import 65000:{{spod}}
VXLAN-E11(config-vrf-af-ipv4)# route-target import 65000:{{spod}} evpn
VXLAN-E11(config-vrf-af-ipv4)# route-target export 65000:{{spod}}
VXLAN-E11(config-vrf-af-ipv4)# route-target export 65000:{{spod}} evpn
VXLAN-E11(config-vrf-af-ipv4)# show run vrf POD{{spod}}
!Command: show running-config vrf POD{{spod}}
!Time: Mon May 28 21:35:38 2018

version 7.0(3)I4(7)
<SNIP>
vrf context POD{{spod}}
   vni 4{{spod}}00
   rd auto
   address-family ipv4 unicast
     route-target import 65000:{{spod}}
     route-target import 65000:{{spod}} evpn
     route-target export 65000:{{spod}}
     route-target export 65000:{{spod}} evpn
```

Note!

Since no compute servers/hosts are attached to the edge routers, there is no need to have any other VXLANs defined on E11/E12 (for example, VXLAN {{spod}}51 and {{spod}}52). All the host's information can be carried by the L3VNI VXLAN/VLAN that is associated with the VRF.

Step 6: Configure BGP to include your POD's VRF so that any MACs learned on the local switch will be propagated to the other VTEPs and their respective switches.

```
VXLAN-E11(config)# router bgp 100
VXLAN-E11(config-router)# vrf POD{{spod}}
VXLAN-E11(config-router-vrf)# address-family ipv4 unicast
VXLAN-E11(config-router-vrf-af)# advertise l2vpn evpn

VXLAN-E11# show run bgp
!Command: show running-config bgp
!Time: Mon May 28 21:35:38 2018

version 7.0(3)I4(7)
feature bgp

router bgp 100
  address-family ipv4 unicast
<SNIP>
  vrf POD{{spod}}
    address-family ipv4 unicast
      advertise l2vpn evpn
<SNIP>
```

Step 7: Configure the interface VLAN (SVI) for the control VLAN on the POD{{spod}} VRF.

```
VXLAN-E11(config)# interface vlan {{spod}}00
VXLAN-E11(config-if)# vrf member POD{{spod}}
VXLAN-E11(config-if)# ip forward
VXLAN-E11(config-if)# no shutdown
VXLAN-E11(config)# show run int vlan{{spod}}00

!Command: show running-config interface Vlan{{spod}}00
!Time: Mon May 28 21:35:38 2018

version 7.0(3)I4(7)

interface Vlan{{spod}}00
  no shutdown
  vrf member POD{{spod}}
  ip forward
```

Step 8: Verify your control VNI (L3 VNI) is in an UP state under the NVE

```
VXLAN-E11(config)# show nve vni
Codes: CP - Control Plane        DP - Data Plane
       UC - Unconfigured         SA - Suppress ARP

Interface VNI      Multicast-group   State Mode Type [BD/VRF]      Flags
--------- -------- ----------------- ----- ---- ------------------ -----
nve1      4{{spod}}00     n/a               Up    CP   L3 [POD{{spod}}]
```

Step 9: Configure sub-interfaces on E11 in the POD{{spod}} VRF on the links between E11 and Core C1.

```
VXLAN-E11(config)# interface ethernet 1/10.{{spod}}
VXLAN-E11(config-subif)# encapsulation dot1Q 10{{spod}}
VXLAN-E11(config-subif)# vrf member POD{{spod}}
VXLAN-E11(config-subif)# ip address 172.11.{{spod}}.1/30
VXLAN-E11(config-subif)# no shutdown
VXLAN-E11(config-subif)# show run interface ethernet 1/10.{{spod}}

!Command: show running-config interface Ethernet1/10.{{spod}}
!Time: Mon May 28 21:35:38 2018

version 7.0(3)I4(7)

interface Ethernet1/10.{{spod}}
  encapsulation dot1q 10{{spod}}
  vrf member POD{{spod}}
  ip address 172.11.{{spod}}.1/30
  no shutdown
```

Step 10: Configure sub-interfaces on C1 in the default VRF on the links between C1 and Edge E11 and E12.

```
VXLAN-C1(config)# interface ethernet 1/10.{{spod}}
VXLAN-C1(config-subif)# encapsulation dot1Q 10{{spod}}
VXLAN-C1(config-subif)# ip address 172.11.{{spod}}.2/30
VXLAN-C1(config-subif)# interface ethernet 1/15.{{spod}}
VXLAN-C1(config-subif)# encapsulation dot1Q 15{{spod}}
VXLAN-C1(config-subif)# ip address 172.12.{{spod}}.2/30
VXLAN-C1(config-subif)# show run interface ethernet 1/10.{{spod}}, ethernet 1/15.{{spod}}

!Command: show running-config interface Ethernet1/10.{{spod}}, Ethernet1/15.{{spod}}
!Time: Mon May 28 21:35:38 2018

version 6.0(2)U3(7)

interface Ethernet1/10.{{spod}}
  encapsulation dot1q 10{{spod}}
  ip address 172.11.{{spod}}.2/30

interface Ethernet1/15.{{spod}}
  encapsulation dot1q 15{{spod}}
  ip address 172.12.{{spod}}.2/30
```

Step 11: On E11, define BGP peering towards C1 to connect the tenants to the outside world.

```
VXLAN-E11(config)# router bgp 100
VXLAN-E11(config-router)# vrf POD{{spod}}
VXLAN-E11(config-router-vrf)# address-family ipv4 unicast
VXLAN-E11(config-router-vrf-af)# advertise l2vpn evpn
VXLAN-E11(config-router-vrf-neighbor-af)# neighbor 172.11.{{spod}}.2 remote-as 64437
VXLAN-E11(config-router-vrf-neighbor)# address-family ipv4 unicast
VXLAN-E11(config-router-vrf-neighbor-af)# show run bgp

!Command: show running-config bgp
!Time: Mon May 28 21:35:38 2018

version 7.0(3)I4(7)
feature bgp

router bgp 100
<SNIP>
vrf POD{{spod}}
    address-family ipv4 unicast
      advertise l2vpn evpn
    neighbor 172.11.{{spod}}.2 remote-as 64437
      address-family ipv4 unicast
<SNIP>
```

Step 12: Copy paste the provided file on your desktop in the “POD{{spod}}_Configs“ to E12. This file is titled “E12_pod{{spod}}”. The only difference in the configuration are the following sections:

- IP address and description for VRF subinterface to core C1
- BGP neighbor statement

> **Important!**
>
> It is vital that you do not skip this step!

Step 13: Verify that the BGP sessions for POD{{spod}} VRF are not yet in the “established” state on E11.

```
VXLAN-E11# show bgp sessions vrf POD{{spod}}
Total peers 6, established peers 4
ASN 100
VRF POD{{spod}}, local ASN 100
peers 2, established peers 0, local router-id 172.10.2.1
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)
172.11.{{spod}}.2      64437  0     00:11:07|never   |never    I   0/0          0/0
```

Step 14: On C1, define BGP peering towards E11 and E12. As we are leaking all the routes to the default VRF on C1, we need to make sure we configure a route map on C1 to make it only advertise the internet subnet and stop it from leaking all host routes from the default VRF into other VRFs. This is accomplished via an ip prefix list.

Note!

After completing this step, the POD Ubuntu-V{{spod}} VM (VXLAN domain) should be able to reach the external VM (non VXLAN domain) as one of the redundant paths to the C3 core (E11<>C1<>C3) will have been established.

```
VXLAN-C1(config)# router bgp 64437
VXLAN-C1(config-router)# neighbor 172.11.{{spod}}.1 remote-as 100
VXLAN-C1(config-router-neighbor)# address-family ipv4 unicast
VXLAN-C1(config-router-vrf-neighbor-af)# route-map PermitIRoute out
VXLAN-C1(config-router-vrf-neighbor-af)# neighbor 172.12.{{spod}}.1 remote-as 100
VXLAN-C1(config-router-vrf-neighbor)# address-family ipv4 unicast
VXLAN-C1(config-router-vrf-neighbor-af)# route-map PermitIRoute out
VXLAN-C1(config-router-vrf-neighbor-af)# show run bgp

!Command: show running-config bgp
!Time: Mon May 28 21:35:38 2018

version 6.0(2)U3(7)
feature bgp

router bgp 64437
<SNIP>
neighbor 172.11.{{spod}}.1 remote-as 100
     address-family ipv4 unicast
       route-map PermitIRoute out
neighbor 172.12.{{spod}}.1 remote-as 100
     address-family ipv4 unicast
       route-map PermitIRoute out
neighbor 192.168.6.14 remote-as 64438
     address-family ipv4 unicast
       route-map Route_Filtering_to_C3 out

<SNIP>
VXLAN-C1(config-router-vrf-neighbor-af)# show route-map PermitIRoute
route-map PermitIRoute, permit, sequence 10
  Match clauses:
    ip address prefix-lists: PermitIRoute
  Set clauses:
VXLAN-C1(config-router-vrf-neighbor-af)# show ip prefix-list PermitIRoute
ip prefix-list FilterOverlay: 2 entries
   seq 10 deny 10.0.0.0/8 le 32 ← POD routes
   seq 50 permit 0.0.0.0/0 le 32
ip prefix-list Filter_C1Routes_ExceptLo3_To_C3: 19 entries
   seq 1 deny 101.1.1.1/32 ← Underlay routes
   seq 2 deny 102.1.1.1/32
<SNIP>
   seq 50 permit 0.0.0.0/0 le 32
ip prefix-list PermitIRoute: 1 entries
   seq 10 permit 75.162.84.0/24 ← Internet routes
```

Step 15: Verify that there is now a route to the “internet” VM on Leaf L11 to which the Ubuntu-VX is connected.

```
VXLAN-L11# show ip route 75.162.84.11 vrf POD{{spod}}
IP Route Table for VRF POD{{spod}}"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>

75.162.84.0/24, ubest/mbest: 1/0
    *via 104.1.1.1%default, [200/0], 00:01:51, bgp-100, internal, tag 64437 (evpn)segid: 4
     ↑ Route to Ubuntu I is learned via VTEP of E11
```

Step 16: Use ping to test connectivity between POD VM and external VM (V{{spod}} and I).

```
VXLAN\POD{{spod}}user@ubuntu-v{{spod}}:~$ ping 75.162.84.11 ← ping external host Ubuntu-I
PING 75.162.84.11 (75.162.84.11) 56(84) bytes of data.
64 bytes from 75.162.84.11: icmp_req=1 ttl=60 time=0.367 ms
64 bytes from 75.162.84.11: icmp_req=2 ttl=60 time=0.409 ms
64 bytes from 75.162.84.11: icmp_req=3 ttl=60 time=0.436 ms
64 bytes from 75.162.84.11: icmp_req=4 ttl=60 time=0.445 ms
^C
--- 75.162.84.11 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3000ms
```

Step 17: Verify that all BGP sessions for your POD are ‘Established’

```
VXLAN-E11# show bgp sessions vrf POD{{spod}}
Total peers 37, established peers 37
 ASN 100
 VRF POD{{spod}}, local ASN 100
 peers 1, established peers 1, local router-id 172.11.2.1
 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)
 172.11.{{spod}}.2       64437 0     2d02h   |00:00:57|00:00:24 E   179/46043      0/0
```

```
VXLAN-E12# show bgp sessions vrf POD{{spod}}
Total peers 37, established peers 37
 ASN 100
 VRF POD{{spod}}, local ASN 100
 peers 1, established peers 1, local router-id 172.12.2.1
 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)
 172.12.{{spod}}.2       64437 0     2d02h   |00:00:34|00:00:03 E   179/46345      0/0
```

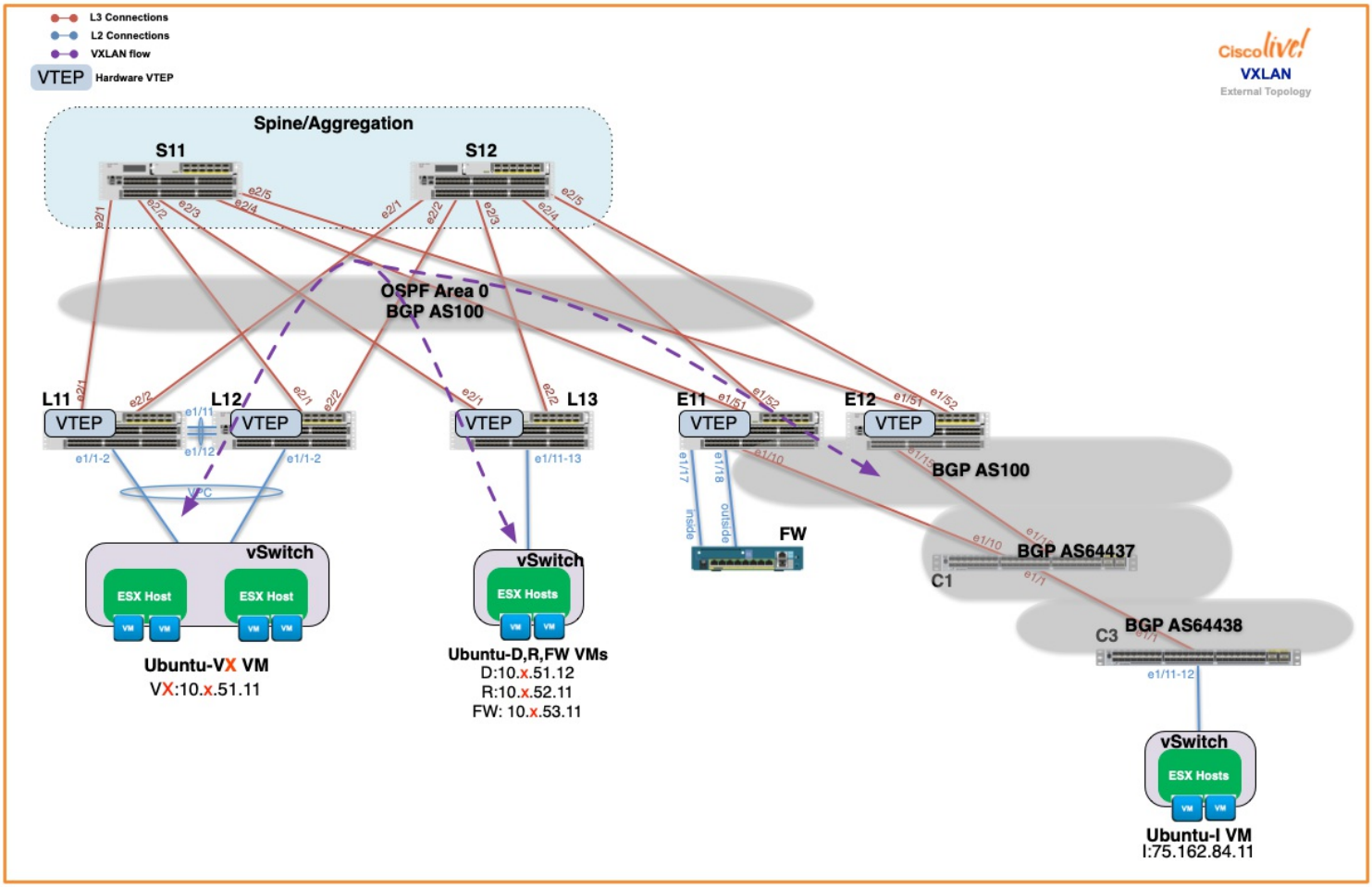
Use Case 2

Introduction

Use Case 2

VXLAN Border Leaf Configuration - External Network Communication Through A Firewall

This module is based of the topology shown in the figure below. It demonstrates the VXLAN host’s connectivity with external networks through the border leaf/edge switch E11, through a firewall.



One host (Ubuntu-FW{{spod}}) is connected to a VLAN ({{spod}}53) on L13 and the external “internet” host (Ubuntu-I) is connected to Internet core router C3. The firewall is hanging off of E11 and is in transparent mode. To facilitate this communication, the border leaf switch (referred to from here on as edge switch), E11 needs to be configured with a VTEP. The VXLAN traffic will terminate at E11 and then regular (non-VXLAN) traffic for each of the PODs will be sent through the firewall, be switched onto another VLAN ({{spod}}54), where an SVI is present, and then proceed to core C1 and C2 on the respective customer VRFs and then eventually routed to C3 through BGP. In our use case, we are using E11 as an edge and service leaf.

Nexus 9300 VXLAN Configuration

Use Case 2

Nexus 9300 VXLAN Configuration

Note!

All of the routing needed for this section was already configured in the previous section, so it will not be covered in this section. The L3 VNI was configured in the previous section as well.

Step 1: Open putty sessions to your POD VMs and verify you are unable to ping from Ubuntu FW{{spod}} to Ubuntu I

Ubuntu-FW{{spod}}

```
VXLAN\POD{{spod}}user@ubuntu-FW{{spod}}:~$ ping 75.162.84.11 ← ping external host Ubuntu-I
PING 75.162.84.11 (75.162.84.11) 56(84) bytes of data.
^C
--- 75.162.84.11 ping statistics ---
6 packets transmitted, 0 received, 100% packet loss, time 5040ms
```

Step 2: Configure edge E11 with VLAN {{spod}}53 / VXLAN VNI 42{{spod}}53.

```
VXLAN-E11(config)# vlan {{spod}}53
VXLAN-E11(config-vlan)# vn-segment 42{{spod}}53
VXLAN-E11(config-vlan)# exit

VXLAN-E11(config)# show run vlan {{spod}}53

!Command: show running-config vlan {{spod}}53
!Time: Mon May 28 22:27:59 2018

version 7.0(3)I4(7)
vlan {{spod}}53
vlan {{spod}}53
  vn-segment 42{{spod}}53
```

Step 3: Configure leaf L13 with VLAN {{spod}}53 / VXLAN VNI 42{{spod}}53.

```
VXLAN-L13(config)# vlan {{spod}}53
VXLAN-L13(config-vlan)# vn-segment 42{{spod}}53
VXLAN-L13(config-vlan)# exit

VXLAN-L13(config)# show run vlan {{spod}}53

!Command: show running-config vlan {{spod}}53
!Time: Mon May 28 22:27:59 2018

version 7.0(3)I4(7)
vlan {{spod}}53
vlan {{spod}}53
  vn-segment 42{{spod}}53
```

Step 4: Configure the interface VLAN (SVI) for the gateway of Ubuntu-FW{{spod}} on edge E11.

> **Note!**
>
> Traffic from Ubuntu-FW{{spod}} will be placed on VLAN {{spod}}53 on L13. From there, it will traverse VXLAN 42{{spod}}53 which terminates on E11 as VLAN {{spod}}53. The “inside” interface of the firewall is a member of VLAN {{spod}}53. The traffic will proceed to the firewall where it will be switched to the “outside” interface (via an allow any any statement), which is a member of VLAN {{spod}}54. This step configures the SVI for VLAN {{spod}}54, which is the SVI which will respond to Ubuntu-FW{{spod}} when it ARPs.

```
VXLAN-E11(config)# vlan {{spod}}54
VXLAN-E11(config-vlan)# interface vlan {{spod}}54
VXLAN-E11(config-if)# vrf member POD{{spod}}
VXLAN-E11(config-if)# ip address 10.{{spod}}.53.1/24
VXLAN-E11(config-if)# no shutdown
VXLAN-E11(config)# show run int vlan{{spod}}54

!Command: show running-config interface Vlan{{spod}}54
!Time: Mon May 28 22:27:59 2018

version 7.0(3)I4(7)

interface Vlan{{spod}}54
  no shutdown
  vrf member POD{{spod}}
  ip address 10.{{spod}}.53.1/24
```

Step 5: On the NVE configuration for E11, configure the Layer 3 VNID to use ingress replication.

```
VXLAN-E11(config)# interface nve1
VXLAN-E11(config-if-nve)# member vni 42{{spod}}53
VXLAN-E11(config-if-nve-vni)# ingress-replication protocol bgp
VXLAN-E11(config-if-nve)# show run interface nve1

!Command: show running-config interface nve1
!Time: Mon May 28 22:27:59 2018

version 7.0(3)I4(7)

interface nve1
  no shutdown
  source-interface loopback0
  host-reachability protocol bgp
  redundancy-group
  member vni 4{{spod}}00 associate-vrf
  member vni 42{{spod}}53
    ingress-replication protocol bgp
```

Step 6: On the NVE configuration for L13, configure the layer 3 VNID to use ingress replication.

```
VXLAN-L13(config)# interface nve1
VXLAN-L13(config-if-nve)# member vni 42{{spod}}53
VXLAN-L13(config-if-nve-vni)# ingress-replication protocol bgp
VXLAN-L13(config-if-nve)# show run interface nve1

!Command: show running-config interface nve1
!Time: Mon May 28 22:27:59 2018

version 7.0(3)I4(7)

interface nve1
  no shutdown
  source-interface loopback0
  host-reachability protocol bgp
  member vni 4{{spod}}00 associate-vrf
  member vni 42{{spod}}53
    ingress-replication protocol bgp
```

Step 7: Import/export the data VLAN VNIDs (VLAN {{spod}}53) into/out of EVPN for host MAC learning on E11.

```
VXLAN-E11(config)# evpn
VXLAN-E11(config-evpn)# vni 42{{spod}}53 l2
VXLAN-E11(config-evpn-evi)# rd auto
VXLAN-E11(config-evpn-evi)# route-target both auto
```

Step 8: Configure BGP to include your POD's VLAN {{spod}}54 subnet so downstream devices know how to route traffic back to devices in your POD.

```
VXLAN-E11(config)# router bgp 100
VXLAN-E11(config-router)# vrf POD{{spod}}
VXLAN-E11(config-router-vrf)# address-family ipv4 unicast
VXLAN-E11(config-router-vrf-af)# network 10.{{spod}}.53.0/24
VXLAN-E11(config-router-vrf-af)# show run bgp

!Command: show running-config bgp
!Time: Mon May 28 22:27:59 2018

version 7.0(3)I4(7)
feature bgp

router bgp 100
<SNIP>
  neighbor 200.200.200.2 remote-as 100
    update-source loopback3
    address-family ipv4 unicast
      send-community both
      next-hop-self
    address-family l2vpn evpn
      send-community both
  neighbor 200.200.200.3 remote-as 100
    update-source loopback3
    address-family ipv4 unicast
      send-community both
      next-hop-self
    address-family l2vpn evpn
      send-community both
  vrf POD{{spod}}
    address-family ipv4 unicast
      advertise l2vpn evpn
      network 10.{{spod}}.53.0/24
<SNIP>
evpn
  vni 42{{spod}}53 l2
    rd auto
    route-target import auto
    route-target export auto
<SNIP>
```

Step 9: Import/export the data VLAN VNIDs (VLAN {{spod}}53) into/out of EVPN for host MAC learning on L13.

```
VXLAN-L13(config)# evpn
VXLAN-L13(config-evpn)# vni 42{{spod}}53 l2
VXLAN-L13(config-evpn-evi)# rd auto
VXLAN-L13(config-evpn-evi)# route-target both auto
VXLAN-L13(config-evpn-evi)# show run bgp

!Command: show running-config bgp
!Time: Mon May 28 22:27:59 2018

version 7.0(3)I4(7)
feature bgp

<SNIP>
evpn
  vni 42{{spod}}53 l2
    rd auto
    route-target import auto
    route-target export auto
<SNIP>
```

Traffic Flow Verification

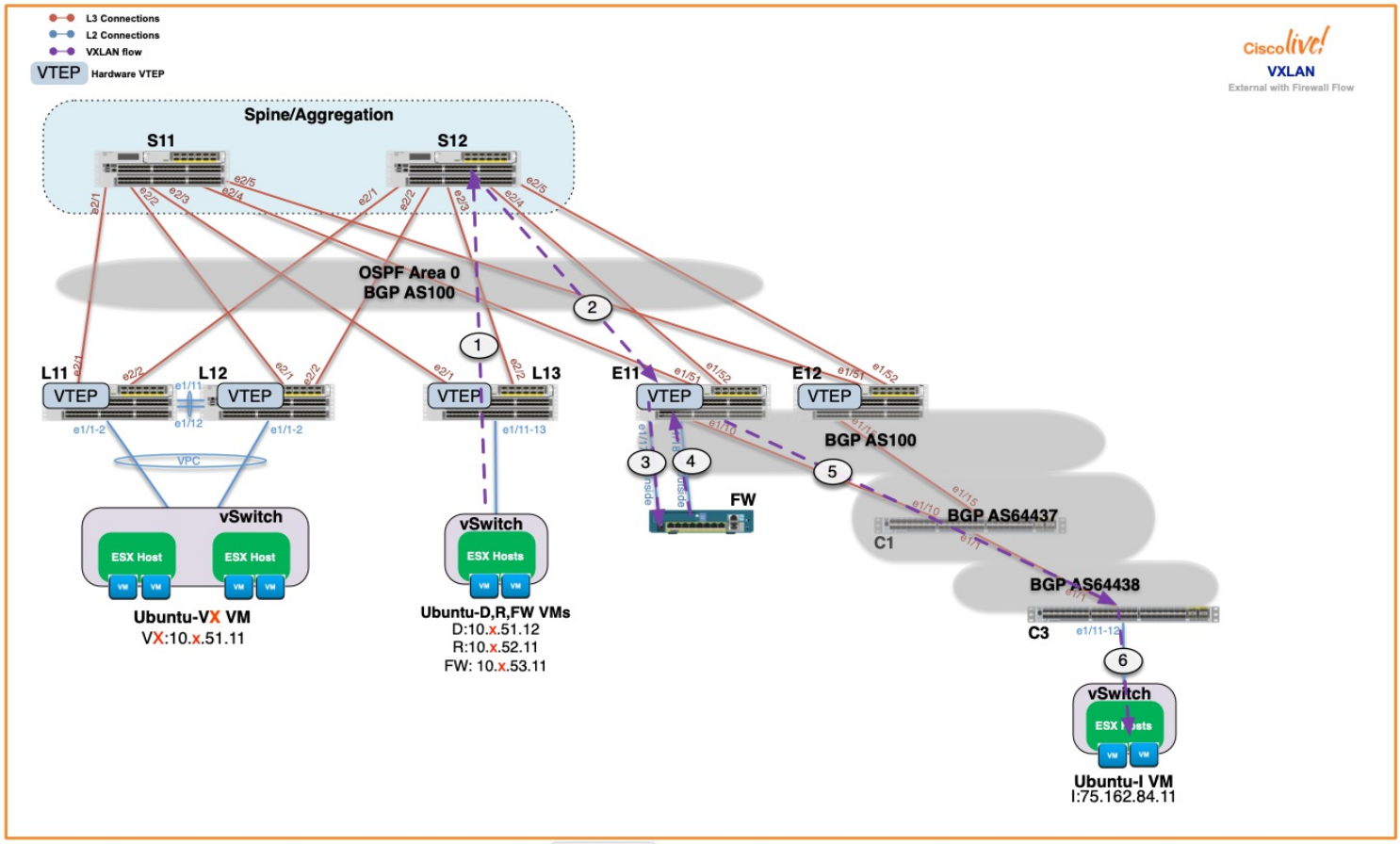
Use Case 2

Traffic Flow Verification

Step 1: Use ping to test connectivity between POD VM and external VM (FW{{spod}} and I).

```
VXLAN\POD{{spod}}user@ubuntu-FW{{spod}}:~$ ping 75.162.84.11 ← ping external host Ubuntu-I
PING 75.162.84.11 (75.162.84.11) 56(84) bytes of data.
64 bytes from 75.162.84.11: icmp_req=1 ttl=60 time=0.367 ms
64 bytes from 75.162.84.11: icmp_req=2 ttl=60 time=0.409 ms
64 bytes from 75.162.84.11: icmp_req=3 ttl=60 time=0.436 ms
64 bytes from 75.162.84.11: icmp_req=4 ttl=60 time=0.445 ms
^C
--- 75.162.84.11 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3000ms
```

In summary, traffic is now flowing from VM Ubuntu-FW{{spod}} to the internet IP address via the firewall. Please refer to the below figure for a representation of the traffic flow:



1. Traffic is placed on VLAN {{spod}}53 by Ubuntu-FW{{spod}}. It is encapsulated in VXLAN 42{{spod}}53 and sent onwards to the spines.
2. Traffic is switched on the spines and sent towards E11 where it is decapsulated from VXLAN 42{{spod}}53 and placed on VLAN {{spod}}53.
3. Traffic travels from E11 to the firewall's inside interface. Once at the firewall, traffic matches the "allow any any" statement and is placed onto the firewall's outside interface, which is a member of VLAN {{spod}}54.
4. Traffic travels from the firewall's outside interface to interface VLAN {{spod}}54 (10.{{spod}}.53.1) on E11.
5. Traffic is routed from interface VLAN {{spod}}54 to the core switches.
6. Traffic is routed from the core switches to the internet.

Use Case 1 and Use Case 2 Summary

This module demonstrated using VXLAN between a hardware VTEP (N9000) and another hardware VTEP (N9000) using the EVPN BGP control plane method. The VXLAN gateway extended the layer 2 network over a layer 3 transport network. Connectivity across the VXLAN overlay was tested from hosts connected to Layer 2 VLAN segments configured on different switches that were separated by a layer 3 boundary. Connectivity was tested for hosts on the same VXLANs (VXLAN bridging), on different VXLANs (VXLAN routing) and on hosts outside of

the datacenter (external VXLAN), both with and without a firewall.

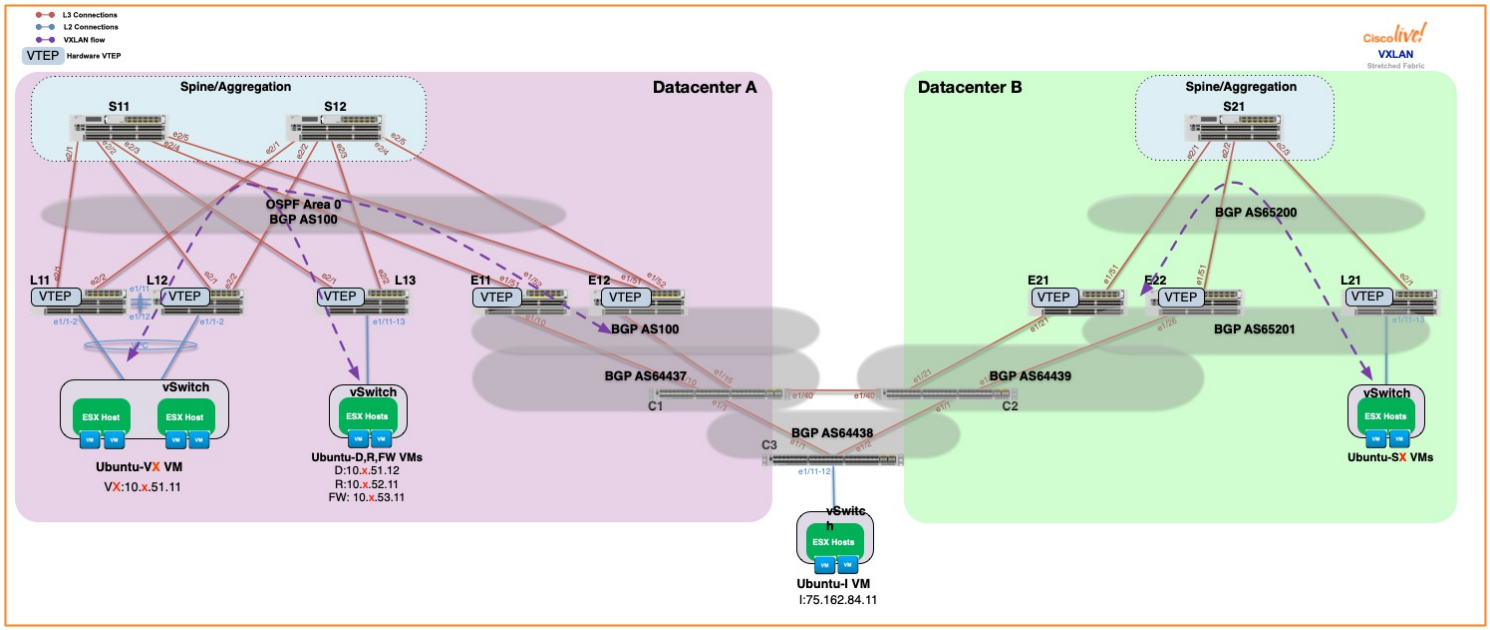
Use Case 3

Introduction

Use Case 3

Multipod Fabric - VXLAN Across Two Data Centers

This module is based off of the topology shown in the figure below. In this scenario we will be making the VXLAN fabric look like one big fabric so that the VLANs/Layer 2 domains that are present in DataCenterA will be Multipod to DataCenterB.



As was discussed earlier, DC A is using OSPF for the underlay and IBGP for the overlay. DC B, however, will be using EBGP for both the underlay and overlay to demonstrate another way to configure VXLAN.

The POD configurations are the same, regardless of underlay technology, so deploying tenants in either data center is seamless.

A VM called Ubuntu-S{{spod}} is connected to leaf L21. It will be able to ping hosts Ubuntu-V{{spod}}, Ubuntu-D and Ubuntu-R to show that it can not only reach bridged networks, but also routed networks in DC A. In addition, the Ubuntu- S{{spod}} host will be able to access the external Internet Ubuntu-I host.

Underlay Configuration Verification

Use Case 3

Underlay Configuration Verification

As mentioned earlier, DC2 is running EBGP for both the underlay and the overlay. The following section will highlight the relevant configurations.

Note!

Since all PODs are sharing hardware, the underlay configuration is already pre-configured, just like in DC1.

Important!

Only execute commands shown in the boxes, below. Any commands shown under “Configuration Sample” are just an example. They are not meant to be executed.

Step 1: Place an IP address on each interface that is connecting to a leaf switch. Since we have only one spine (S21) in this part of the DC, there are only three interfaces which are connected to L21, E21 and E22.

Configuration sample:

```
VXLAN-S21(config)# interface ethernet 2/1
VXLAN-S21(config-if)# no switchport
VXLAN-S21(config-if)# ip address 192.168.3.5/30
VXLAN-S21(config-if)# no shutdown

VXLAN-S21(config)# interface ethernet 2/2
VXLAN-S21(config-if)# no switchport
VXLAN-S21(config-if)# ip address 192.168.3.9/30
VXLAN-S21(config-if)# no shutdown

VXLAN-S21(config)# interface ethernet 2/3
VXLAN-S21(config-if)# no switchport
VXLAN-S21(config-if)# ip address 192.168.3.1/30
VXLAN-S21(config-if)# no shutdown
```

Step 2: Configure the EBGP session on spine S21 to each leaf and enable EVPN. In addition, disable AS peer checks to each neighbor tag. Make sure "NextHopUnchanged" is configured so that the routes leaving the spine won't change the VTEPs address. In addition, we need to flag the neighbors to disable the peer AS checks because all of the leafs are in the same AS.

Configuration sample:

```
VXLAN-S21(config)# route-map NextHopUnchanged permit 10
VXLAN-S21(config-route-map)# set ip next-hop unchanged
VXLAN-S21(config)# router bgp 65200
VXLAN-S21(config-router)# address-family l2vpn evpn
VXLAN-S21(config-router-af)# nexthop route-map NextHopUnchanged
VXLAN-S21(config-router-af)# retain route-target all
VXLAN-S21(config-router-af)# neighbor 192.168.3.2 remote-as 65201
VXLAN-S21(config-router-neighbor)# address-family ipv4 unicast
VXLAN-S21(config-router-neighbor-af)# disable-peer-as-check
VXLAN-S21(config-router-neighbor-af)# send-community
VXLAN-S21(config-router-neighbor-af)# exit
VXLAN-S21(config-router-neighbor)# address-family ipv4 unicast
VXLAN-S21(config-router-neighbor-af)# disable-peer-as-check
VXLAN-S21(config-router-neighbor-af)# send-community both
VXLAN-S21(config-router-neighbor-af)# address-family l2vpn evpn
VXLAN-S21(config-router-neighbor-af)# disable-peer-as-check
VXLAN-S21(config-router-neighbor-af)# send-community extended
VXLAN-S21(config-router-neighbor-af)# route-map NextHopUnchanged out
```

> **Note!**
>
> The spine is in one AS and the leafs are in another. EBGP is running between the leafs and the spines.

Step 3: On each leaf and edge switch (L21, E21 and E22), configure each EBGP session to the spine. Configure the "AllowAS In" tag to make sure that the routes with the same AS numbers are allowed to be programmed from the spines.

Configuration sample:

```
VXLAN-L21(config-router-af)# neighbor 192.168.3.1 remote-as 65200
VXLAN-L21(config-router-neighbor)# address-family ipv4 unicast
VXLAN-L21(config-router-neighbor-af)# allowas-in
VXLAN-L21(config-router-neighbor-af)# send-community both
VXLAN-L21(config-router-neighbor-af)# address-family l2vpn evpn
VXLAN-L21(config-router-neighbor-af)# allowas-in
VXLAN-L21(config-router-neighbor-af)# send-community both
```

Step 4: Verify BGP configuration on L21 and ensure BGP is up and routes are learned on L21

```
VXLAN-L21# sh run bgp

 !Command: show running-config bgp
 !Time: Mon May 28 22:16:56 2018

 version 7.0(3)I4(7)
 feature bgp

 router bgp 65201
   address-family ipv4 unicast
     network 111.1.1.1/32
     network 111.1.1.2/32
     network 111.1.1.3/32
   neighbor 192.168.3.1 remote-as 65200
     address-family ipv4 unicast
       allowas-in
       send-community both
     address-family l2vpn evpn
       allowas-in
       send-community both

VXLAN-L21# sh ip route bgp
 IP Route Table for VRF "default"
 '*' denotes best ucast next-hop
 '**' denotes best mcast next-hop
 '[x/y]' denotes [preference/metric]
 '%<string>' in via output denotes VRF <string>

 75.162.84.0/24, ubest/mbest: 1/0
     *via 192.168.3.1, [20/0], 2d03h, bgp-65201, external, tag 65200
 101.1.1.1/32, ubest/mbest: 1/0
     *via 192.168.3.1, [20/0], 2d03h, bgp-65201, external, tag 65200
 101.1.1.10/32, ubest/mbest: 1/0
     *via 192.168.3.1, [20/0], 2d03h, bgp-65201, external, tag 65200
 102.1.1.1/32, ubest/mbest: 1/0
     *via 192.168.3.1, [20/0], 2d03h, bgp-65201, external, tag 65200
 103.1.1.1/32, ubest/mbest: 1/0
     *via 192.168.3.1, [20/0], 2d03h, bgp-65201, external, tag 65200
 104.1.1.1/32, ubest/mbest: 1/0
     *via 192.168.3.1, [20/0], 2d03h, bgp-65201, external, tag 65200
 104.1.1.2/32, ubest/mbest: 1/0
     *via 192.168.3.1, [20/0], 2d03h, bgp-65201, external, tag 65200
 104.1.1.3/32, ubest/mbest: 1/0
     *via 192.168.3.1, [20/0], 2d03h, bgp-65201, external, tag 65200
 105.1.1.1/32, ubest/mbest: 1/0
     *via 192.168.3.1, [20/0], 2d03h, bgp-65201, external, tag 65200
 105.1.1.2/32, ubest/mbest: 1/0
     *via 192.168.3.1, [20/0], 2d03h, bgp-65201, external, tag 65200
 105.1.1.3/32, ubest/mbest: 1/0
     *via 192.168.3.1, [20/0], 2d03h, bgp-65201, external, tag 65200
 106.1.1.3/32, ubest/mbest: 1/0
     *via 192.168.3.1, [20/0], 2d03h, bgp-65201, external, tag 65200
```

```
107.1.1.3/32, ubest/mbest: 1/0
    *via 192.168.3.1, [20/0], 2d03h, bgp-65201, external, tag 65200
108.1.1.3/32, ubest/mbest: 1/0
    *via 192.168.3.1, [20/0], 2d03h, bgp-65201, external, tag 65200
109.1.1.1/32, ubest/mbest: 1/0
    *via 192.168.3.1, [20/0], 2d03h, bgp-65201, external, tag 65200
109.1.1.2/32, ubest/mbest: 1/0
    *via 192.168.3.1, [20/0], 2d03h, bgp-65201, external, tag 65200
109.1.1.3/32, ubest/mbest: 1/0
    *via 192.168.3.1, [20/0], 2d03h, bgp-65201, external, tag 65200
110.1.1.1/32, ubest/mbest: 1/0
    *via 192.168.3.1, [20/0], 2d03h, bgp-65201, external, tag 65200
110.1.1.2/32, ubest/mbest: 1/0
    *via 192.168.3.1, [20/0], 2d03h, bgp-65201, external, tag 65200
110.1.1.3/32, ubest/mbest: 1/0
    *via 192.168.3.1, [20/0], 2d03h, bgp-65201, external, tag 65200
200.200.200.5/32, ubest/mbest: 1/0
    *via 192.168.3.1, [20/0], 2d03h, bgp-65201, external, tag 65200
```

Step 5: On the edge router E21 and E22, configure the EBGP session to connect the underlay to the external router. This router also provides external connectivity to the internet as well as routes to connect the DC edges together.

> **Note!**
>
> In the configuration sample below, only E21 is shown. However, E21 and E22 are configured the same way.

Configuration sample:

```
VXLAN-E21(config)# router bgp 65201
VXLAN-E21(config-router)# neighbor 192.168.3.5 remote-as 65200
VXLAN-E21(config-router-neighbor)# address-family ipv4 unicast
VXLAN-E21(config-router-neighbor-af)# allow as-in
VXLAN-E21(config-router-neighbor-af)# send-community both
VXLAN-E21(config-router-neighbor-af)# address-family l2vpn evpn
VXLAN-E21(config-router-neighbor-af)# allowas-in
VXLAN-E21(config-router-neighbor-af)# send-community both
VXLAN-E21(config-router-neighbor-af)# route-map NextHopUnchanged out
VXLAN-E21(config-router-neighbor-af)# neighbor 192.168.150.6 remote-as 64439
VXLAN-E21(config-router-neighbor)# address-family ipv4 unicast
VXLAN-E21(config-router-neighbor-af)# route-map Route_Filtering_to_C1-C2 out
```

Step 6: On edge switches E21, E22, E11, and E12 configure EBGP multihop neighbor commands. This will advertise all the routes from DC1 (including hosts and underlay routes) to DC2.

> **Note!**
>
> In the configuration sample below, only E21 is shown. However, E21, E22, E11, and E12 are configured the same way.

Note!

To stretch the fabric across two Datacenters, we need to use EGP multihop between the spines and between the edge switches. This will create one big Multipod fabric.

Configuration sample:

```
VXLAN-E21(config)# router bgp 65201
VXLAN-E21(config-router)# address-family ipv4 unicast
VXLAN-E21(config-router-af)# network 109.1.1.1/32
VXLAN-E21(config-router-af)# network 109.1.1.2/32
VXLAN-E21(config-router-af)# network 109.1.1.3/32
VXLAN-E21(config-router-af)# maximum-paths 6
VXLAN-E21(config-router-af)# address-family l2vpn evpn
VXLAN-E21(config-router-af)# nexthop route-map NextHopUnchanged
VXLAN-E21(config-router-af)# retain route-target all
VXLAN-E21(config-router-af)# neighbor 104.1.1.3 remote-as 100
VXLAN-E21(config-router-neighbor)# update-source loopback3
VXLAN-E21(config-router-neighbor)# ebgp-multihop 5
VXLAN-E21(config-router-neighbor)# address-family ipv4 unicast
VXLAN-E21(config-router-neighbor-af)# send-community both
VXLAN-E21(config-router-neighbor-af)# route-map Lo3Filter_to_PeerDC_EdgeRouters out
VXLAN-E21(config-router-neighbor-af)# soft-reconfiguration inbound always
VXLAN-E21(config-router-neighbor-af)# address-family l2vpn evpn
VXLAN-E21(config-router-neighbor-af)# send-community both
VXLAN-E21(config-router-neighbor-af)# route-map NextHopUnchanged out
VXLAN-E21(config-router-neighbor-af)# neighbor 105.1.1.3 remote-as 100
VXLAN-E21(config-router-neighbor)# update-source loopback3
VXLAN-E21(config-router-neighbor)# ebgp-multihop 5
VXLAN-E21(config-router-neighbor)# address-family ipv4 unicast
VXLAN-E21(config-router-neighbor-af)# send-community both
VXLAN-E21(config-router-neighbor-af)# route-map Lo3Filter_to_PeerDC_EdgeRouters out
VXLAN-E21(config-router-neighbor-af)# soft-reconfiguration inbound always
VXLAN-E21(config-router-neighbor-af)# address-family l2vpn evpn
VXLAN-E21(config-router-neighbor-af)# send-community both
VXLAN-E21(config-router-neighbor-af)# route-map NextHopUnchanged out
```

Step 7: Verify BGP multihop configuration on E11, E12, E21, and E22.

Note!

As is shown below, all edge switches should be configured to retain route targets and “NextHop Unchanged” should be configured to make sure all the EVPN routes are advertised with the right VTEP addresses.

Note!

A template could easily be used in the configuration below to allow for easier configuration of a large number of neighbors.

```
VXLAN-E11# show run bgp
 !Command: show running-config bgp
 !Time: Mon May 28 22:16:56 2018

 version 7.0(3)I4(7)
 feature bgp

 router bgp 100
 <SNIP>
   address-family l2vpn evpn
     nexthop route-map NextHopUnchanged
     retain route-target all
   neighbor 109.1.1.3 remote-as 65201
     update-source loopback3
     ebgp-multihop 5
     address-family ipv4 unicast
       send-community both
       route-map Lo3Filter_to_PeerDC_EdgeRouters out
       soft-reconfiguration inbound always
     address-family l2vpn evpn
       send-community both
       route-map NextHopUnchanged out
   neighbor 110.1.1.3 remote-as 65201
     update-source loopback3
     ebgp-multihop 5
     address-family ipv4 unicast
       send-community both
       route-map Lo3Filter_to_PeerDC_EdgeRouters out
       soft-reconfiguration inbound always
     address-family l2vpn evpn
       send-community both
       route-map NextHopUnchanged out
<SNIP>
```

```
VXLAN-E12# show run bgp
!Command: show running-config bgp
 !Time: Mon May 28 22:16:56 2018

 version 7.0(3)I4(7)
 feature bgp

 router bgp 100
 <SNIP>
   address-family l2vpn evpn
     nexthop route-map NextHopUnchanged
     retain route-target all
   neighbor 109.1.1.3 remote-as 65201
     update-source loopback3
     ebgp-multihop 5
     address-family ipv4 unicast
       send-community both
       route-map Lo3Filter_to_PeerDC_EdgeRouters out
       soft-reconfiguration inbound always
     address-family l2vpn evpn
       send-community both
       route-map NextHopUnchanged out
   neighbor 110.1.1.3 remote-as 65201
     update-source loopback3
     ebgp-multihop 5
     address-family ipv4 unicast
       send-community both
       route-map Lo3Filter_to_PeerDC_EdgeRouters out
       soft-reconfiguration inbound always
     address-family l2vpn evpn
       send-community both
       route-map NextHopUnchanged out
  <SNIP>
```

```
VXLAN-E21# show run bgp
 !Command: show running-config bgp
 !Time: Mon May 28 22:16:56 2018

 version 7.0(3)I4(7)
 feature bgp

 router bgp 65201
 <SNIP>
   address-family l2vpn evpn
     nexthop route-map NextHopUnchanged
     retain route-target all
   neighbor 104.1.1.3 remote-as 100
     update-source loopback3
     ebgp-multihop 5
     address-family ipv4 unicast
       send-community both
       route-map Lo3Filter_to_PeerDC_EdgeRouters out
       soft-reconfiguration inbound always
     address-family l2vpn evpn
       send-community both
       route-map NextHopUnchanged out
   neighbor 105.1.1.3 remote-as 100
     update-source loopback3
     ebgp-multihop 5
     address-family ipv4 unicast
       send-community both
       route-map Lo3Filter_to_PeerDC_EdgeRouters out
       soft-reconfiguration inbound always
     address-family l2vpn evpn
       send-community both
       route-map NextHopUnchanged out
  <SNIP>
```

```
VXLAN-E22# show run bgp
 !Command: show running-config bgp
 !Time: Mon May 28 22:16:56 2018

 version 7.0(3)I4(7)
 feature bgp

 router bgp 65201
 <SNIP>
   address-family l2vpn evpn
     nexthop route-map NextHopUnchanged
     retain route-target all
   neighbor 104.1.1.3 remote-as 100
     update-source loopback3
     ebgp-multihop 5
     address-family ipv4 unicast
       send-community both
       route-map Lo3Filter_to_PeerDC_EdgeRouters out
       soft-reconfiguration inbound always
     address-family l2vpn evpn
       send-community both
       route-map NextHopUnchanged out
   neighbor 105.1.1.3 remote-as 100
     update-source loopback3
     ebgp-multihop 5
     address-family ipv4 unicast
       send-community both
       route-map Lo3Filter_to_PeerDC_EdgeRouters out
       soft-reconfiguration inbound always
     address-family l2vpn evpn
       send-community both
       route-map NextHopUnchanged out
<SNIP>
```

Step 8: Verify BGP multihop neighbors on E11, E12, E21, and E22.

Note!

Each edge switch will only have 2 multihop neighbors. Only neighbors in the remote datacenter will be multihop. Devices in the local datacenter will peer through the local spine switches.

```
VXLAN-E11# show ip bgp neighbors
<SNIP>
 BGP neighbor is 109.1.1.3,  remote AS 65201, ebgp link,  Peer index 1
   BGP version 4, remote router ID 109.1.1.1
   BGP state = Established, up for 2d08h
   Using loopback3 as update source for this peer
   External BGP peer might be upto 5 hops away
   Last read 00:00:44, hold time = 180, keepalive interval is 60 seconds
   Last written 00:00:34, keepalive timer expiry due 00:00:25
   Received 6136 messages, 0 notifications, 0 bytes in queue
   Sent 4084 messages, 0 notifications, 0 bytes in queue
   Connections established 1, dropped 0
   Last reset by us never, due to No error
   Last reset by peer never, due to No error
 <SNIP>
 BGP neighbor is 110.1.1.3,  remote AS 65201, ebgp link,  Peer index 2
   BGP version 4, remote router ID 110.1.1.1
   BGP state = Established, up for 2d08h
   Using loopback3 as update source for this peer
   External BGP peer might be upto 5 hops away
   Last read 00:00:45, hold time = 180, keepalive interval is 60 seconds
   Last written 00:00:34, keepalive timer expiry due 00:00:25
   Received 6060 messages, 0 notifications, 0 bytes in queue
   Sent 4087 messages, 0 notifications, 0 bytes in queue
   Connections established 1, dropped 0
   Last reset by us never, due to No error
   Last reset by peer never, due to No error
<SNIP>
```

```
VXLAN-E12# show ip bgp neighbors
<SNIP>
 BGP neighbor is 109.1.1.3,  remote AS 65201, ebgp link,  Peer index 1
   BGP version 4, remote router ID 109.1.1.1
   BGP state = Established, up for 2d08h
   Using loopback3 as update source for this peer
   External BGP peer might be upto 5 hops away
   Last read 00:00:46, hold time = 180, keepalive interval is 60 seconds
   Last written 00:00:35, keepalive timer expiry due 00:00:24
   Received 3654 messages, 0 notifications, 0 bytes in queue
   Sent 3983 messages, 0 notifications, 0 bytes in queue
   Connections established 1, dropped 0
   Last reset by us never, due to No error
   Last reset by peer never, due to No error
 <SNIP>
 BGP neighbor is 110.1.1.3,  remote AS 65201, ebgp link,  Peer index 2
   BGP version 4, remote router ID 110.1.1.1
   BGP state = Established, up for 2d08h
   Using loopback3 as update source for this peer
   External BGP peer might be upto 5 hops away
   Last read 00:00:47, hold time = 180, keepalive interval is 60 seconds
   Last written 00:00:35, keepalive timer expiry due 00:00:24
   Received 3573 messages, 0 notifications, 0 bytes in queue
   Sent 3983 messages, 0 notifications, 0 bytes in queue
   Connections established 1, dropped 0
   Last reset by us never, due to No error
   Last reset by peer never, due to No error
<SNIP>
```

```
VXLAN-E21# show ip bgp neighbors
<SNIP>
 BGP neighbor is 104.1.1.3,  remote AS 100, ebgp link,  Peer index 1
   BGP version 4, remote router ID 104.1.1.1
   BGP state = Established, up for 2d09h
   Using loopback3 as update source for this peer
   External BGP peer might be upto 5 hops away
   Last read 00:00:40, hold time = 180, keepalive interval is 60 seconds
   Last written 00:00:50, keepalive timer expiry due 00:00:09
   Received 4108 messages, 0 notifications, 0 bytes in queue
   Sent 6160 messages, 0 notifications, 0 bytes in queue
   Connections established 1, dropped 0
   Last reset by us never, due to No error
   Last reset by peer never, due to No error
<SNIP>
BGP neighbor is 105.1.1.3,  remote AS 100, ebgp link,  Peer index 2
   BGP version 4, remote router ID 105.1.1.1
   BGP state = Established, up for 2d09h
   Using loopback3 as update source for this peer
   External BGP peer might be upto 5 hops away
   Last read 00:00:40, hold time = 180, keepalive interval is 60 seconds
   Last written 00:00:50, keepalive timer expiry due 00:00:09
   Received 3986 messages, 0 notifications, 0 bytes in queue
   Sent 3657 messages, 0 notifications, 0 bytes in queue
   Connections established 1, dropped 0
   Last reset by us never, due to No error
   Last reset by peer never, due to No error
<SNIP>
```

```
VXLAN-E22# show ip bgp neighbors
<SNIP>
 BGP neighbor is 104.1.1.3,  remote AS 100, ebgp link,  Peer index 1
   BGP version 4, remote router ID 104.1.1.1
   BGP state = Established, up for 2d09h
   Using loopback3 as update source for this peer
   External BGP peer might be upto 5 hops away
   Last read 00:00:29, hold time = 180, keepalive interval is 60 seconds
   Last written 00:00:39, keepalive timer expiry due 00:00:20
   Received 4113 messages, 0 notifications, 0 bytes in queue
   Sent 6086 messages, 0 notifications, 0 bytes in queue
   Connections established 1, dropped 0
   Last reset by us never, due to No error
   Last reset by peer never, due to No error
<SNIP>
BGP neighbor is 105.1.1.3,  remote AS 100, ebgp link,  Peer index 2
   BGP version 4, remote router ID 105.1.1.1
   BGP state = Established, up for 2d09h
   Using loopback3 as update source for this peer
   External BGP peer might be upto 5 hops away
   Last read 00:00:28, hold time = 180, keepalive interval is 60 seconds
   Last written 00:00:39, keepalive timer expiry due 00:00:20
   Received 3988 messages, 0 notifications, 0 bytes in queue
   Sent 3578 messages, 0 notifications, 0 bytes in queue
   Connections established 1, dropped 0
   Last reset by us never, due to No error
   Last reset by peer never, due to No error
<SNIP>
```

Pod and Overlay Configuration

Use Case 3

Pod and Overlay Configuration

In this section, each POD user will configure their PODs as if they are deploying a tenant. This is similar to what we did earlier in DC 1.

Step 1: Open putty sessions to your POD VMs and verify you are unable to ping from Ubuntu-S{{spod}} to the Ubuntu-I or Ubuntu-V{{spod}} VMs.

Ubuntu-S{{spod}} VM

```
VXLAN\POD{{spod}}user@ubuntu-S{{spod}}:~$ ping 75.162.84.11 ← ping external host Ubuntu-I
PING 75.162.84.11 (75.162.84.11) 56(84) bytes of data.
^C
--- 75.162.84.11 ping statistics ---
6 packets transmitted, 0 received, 100% packet loss, time 5040ms
VXLAN\POD{{spod}}user@ubuntu-S{{spod}}:~$ ping 10.{{spod}}.51.11 ← ping host ubuntu-V{{spo
PING 10.{{spod}}.51.11 (10.{{spod}}.51.11) 56(84) bytes of data.
^C
--- 10.{{spod}}.51.11 ping statistics ---
6 packets transmitted, 0 received, 100% packet loss, time 5040ms
```

Step 2: Add VLAN to VNID mapping for the L3 VNI on E21. The VLAN number will be {{spod}}00 and will map to VXLAN VNI 4{{spod}}00. Remember: we don't need to define VXLAN {{spod}}51 and {{spod}}52 on the edge routers, as VLAN {{spod}}00 will be carrying all the host routes for all the VLANs in the tenant.

> **Note!**
>
> There are two edge switches (E21 and E22). To save time, you will only issue the below commands on E21. Then, you will copy paste the entire set of commands for E22 on to E22 using the provided configuration file (this file is located on your desktop).

```
VXLAN-E21(config)# vlan {{spod}}00
VXLAN-E21(config-vlan)# vn-segment 4{{spod}}00
VXLAN-E21(config-vlan)# exit
VXLAN-E21(config)# show run vlan {{spod}}00

!Command: show running-config vlan {{spod}}00
!Time: Mon May 28 22:48:50 2018

version 7.0(3)I4(7)
vlan {{spod}}00
vlan {{spod}}00
  vn-segment 4{{spod}}00
```

Step 3: On the NVE configuration for E21, associate the POD VRF to the the Layer 3 VNID used for control plane traffic ({{spod}}00).

```
VXLAN-E21(config)# interface nve1
VXLAN-E21(config-if-nve)# member vni 4{{spod}}00 associate-vrf
VXLAN-E21(config-if-nve)# show run interface nve1

!Command: show running-config interface nve1
!Time: Mon May 28 22:48:50 2018

version 7.0(3)I4(7)

interface nve1
  no shutdown
  source-interface loopback0
  host-reachability protocol bgp
  member vni 4{{spod}}00 associate-vrf
```

Step 4: Create the VRF for the POD, associate the control plane VLAN {{spod}}00 VNID to the VRF and specify the EVPN session used (auto selection).

```
VXLAN-E21(config)# vrf context POD{{spod}}
VXLAN-E21(config-vrf)# vni 4{{spod}}00
VXLAN-E21(config-vrf)# rd auto
VXLAN-E21(config-vrf)# address-family ipv4 unicast
VXLAN-E21(config-vrf-af-ipv4)# route-target import 65000:{{spod}}
VXLAN-E21(config-vrf-af-ipv4)# route-target import 65000:{{spod}} evpn
VXLAN-E21(config-vrf-af-ipv4)# route-target export 65000:{{spod}}
VXLAN-E21(config-vrf-af-ipv4)# route-target export 65000:{{spod}} evpn
VXLAN-E21(config-vrf-af-ipv4)# show run vrf POD{{spod}}
!Command: show running-config vrf POD{{spod}}
!Time: Mon May 28 22:48:50 2018

version 7.0(3)I4(7)
vrf context POD{{spod}}
<SNIP>
   vni 4{{spod}}00
   rd auto
   address-family ipv4 unicast
     route-target import 65000:{{spod}}
     route-target import 65000:{{spod}} evpn
     route-target export 65000:{{spod}}
     route-target export 65000:{{spod}} evpn
```

Step 5: Configure the interface VLAN (SVI) for the control VLAN on the POD{{spod}} VRF.

```
VXLAN-E21(config)# interface vlan {{spod}}00
VXLAN-E21(config-if)# vrf member POD{{spod}}
VXLAN-E21(config-if)# ip forward
VXLAN-E21(config-if)# no shutdown
VXLAN-E21(config)# show run int vlan{{spod}}00

!Command: show running-config interface Vlan{{spod}}00
!Time: Mon May 28 22:48:50 2018

version 7.0(3)I4(7)

interface Vlan{{spod}}00
  no shutdown
  vrf member POD{{spod}}
  ip forward
```

Step 6: Verify your control VNI (L3 VNI) is in an up state under the NVE

```
VXLAN-E21(config)# show nve vni
Codes: CP - Control Plane        DP - Data Plane
       UC - Unconfigured         SA - Suppress ARP

Interface VNI      Multicast-group   State Mode Type [BD/VRF]      Flags
--------- -------- ----------------- ----- ---- ------------------ -----
nve1      4{{spod}}00     n/a               Up    CP   L3 [POD{{spod}}]
```

Step 7: Copy paste the provided file on your desktop into E22. This file is titled “E22”.

> **Important!**
>
> It is vital that you do not skip this step!

Step 8: Configure switch L21 just like the other switches in DC1 were configured (note that this switch is not VPC). Add VLAN to VNID mappings. The value specified with the vn-segment keyword is the VNID value sent in the VXLAN header. Extend VLAN {{spod}}51 and assign it to the same VXLAN ID as we did in the first DC.

```
VXLAN-L21(config)# vlan {{spod}}00
VXLAN-L21(config-vlan)# vn-segment 4{{spod}}00
VXLAN-L21(config-vlan)# vlan {{spod}}51
VXLAN-L21(config-vlan)# vn-segment 42{{spod}}51
VXLAN-L21(config-vlan)# exit
VXLAN-L21# show run vlan {{spod}}00,{{spod}}51

!Command: show running-config vlan {{spod}}00, {{spod}}51
!Time: Mon May 28 22:48:50 2018

version 7.0(3)I4(7)
vlan {{spod}}00,{{spod}}51
vlan {{spod}}00
  vn-segment 4{{spod}}00
vlan {{spod}}51
  vn-segment 42{{spod}}51
```

Step 9: Configure the interface VLAN (SVI) for the gateway of Ubuntu-S{{spod}} on L21.

```
VXLAN-L21(config)# interface Vlan{{spod}}00
VXLAN-L21(config-if)# no shutdown
VXLAN-L21(config-if)# vrf member POD{{spod}}
VXLAN-L21(config)# interface vlan {{spod}}51
VXLAN-L21(config-if)# vrf member POD{{spod}}
VXLAN-L21(config-if)# ip address 10.{{spod}}.51.1/24
VXLAN-L21(config-if)# no shutdown
VXLAN-L21(config-if)# fabric forwarding mode anycast-gateway
VXLAN-L21# show run int vlan {{spod}}00, vlan {{spod}}51

 !Command: show running-config interface Vlan{{spod}}00, interface Vlan{{spod}}51
 !Time: Mon May 28 22:48:50 2018

 version 7.0(3)I4(7)

 interface Vlan{{spod}}00
   no shutdown
   vrf member POD{{spod}}
   ip forward

 interface Vlan{{spod}}51
   no shutdown
   vrf member POD{{spod}}
   ip address 10.{{spod}}.51.1/24
   fabric forwarding mode anycast-gateway
```

Step 10: On the NVE configuration for L21, associate the Layer 3 VNI and Layer 2 VNI as we did in DC1 on L13.

```
VXLAN-L21(config)# int nve1
VXLAN-L21(config-if-nve)# member vni 4{{spod}}00 associate-vrf
VXLAN-L21(config-if-nve)# member vni 42{{spod}}51
VXLAN-L21(config-if-nve-vni)# ingress-replication protocol bgp
VXLAN-L21(config-if-nve-vni)# suppress-arp

VXLAN-L21(config)# vrf context POD{{spod}}
VXLAN-L21(config-vrf)# vni 4{{spod}}00
VXLAN-L21(config-vrf)# rd auto
VXLAN-L21(config-vrf)# address-family ipv4 unicast
VXLAN-L21(config-vrf-af-ipv4)# route-target import 65000:{{spod}}
VXLAN-L21(config-vrf-af-ipv4)# route-target import 65000:{{spod}} evpn
VXLAN-L21(config-vrf-af-ipv4)# route-target export 65000:{{spod}}
VXLAN-L21(config-vrf-af-ipv4)# route-target export 65000:{{spod}} evpn

VXLAN-L21# show run int nve1
!Command: show running-config interface nve1
!Time: Mon May 28 22:48:50 2018

version 7.0(3)I4(7)

 interface nve1
   no shutdown
   source-interface loopback0
   host-reachability protocol bgp
   member vni 4{{spod}}00 associate-vrf
   member vni 42{{spod}}51
     suppress-arp
     ingress-replication protocol bgp
<SNIP>

VXLAN-L21# show run vrf POD{{spod}}
!Command: show running-config vrf POD{{spod}}
!Time: Mon May 28 22:48:50 2018

version 7.0(3)I4(7)

 vrf context POD{{spod}}
 <SNIP>
   vni 4{{spod}}00
   rd auto
   address-family ipv4 unicast
     route-target import 65000:{{spod}}
     route-target import 65000:{{spod}} evpn
     route-target export 65000:{{spod}}
     route-target export 65000:{{spod}} evpn
<SNIP>
```

Step 11: Import/export the data VLAN VNIDs /L2 VNI (VLAN {{spod}}51) into/out of EVPN for host MAC learning on L21.

```
VXLAN-L21(config)# evpn
VXLAN-L21(config-evpn)# vni 42{{spod}}51 l2
VXLAN-L21(config-evpn-evi)# rd auto
VXLAN-L21(config-evpn-evi)# route-target import {{spod}}:42{{spod}}51
VXLAN-L21(config-evpn-evi)# route-target export {{spod}}:42{{spod}}51
```

Step 12: Configure BGP on L21 to advertise the VRF POD{{spod}} routes.

```
VXLAN-L21(config)# router bgp 65201
VXLAN-L21(config-router)# vrf POD{{spod}}
VXLAN-L21(config-router-vrf)# address-family ipv4 unicast
VXLAN-L21(config-router-vrf-af)# advertise l2vpn evpn

VXLAN-L21# show run bgp
!Command: show running-config bgp
!Time: Mon May 28 22:48:50 2018

version 7.0(3)I4(7)
 feature bgp

 router bgp 65201
   address-family ipv4 unicast
     network 111.1.1.1/32
     network 111.1.1.2/32
     network 111.1.1.3/32
   neighbor 192.168.3.1 remote-as 65200
     address-family ipv4 unicast
       allowas-in
       send-community both
     address-family l2vpn evpn
       allowas-in
       send-community both
   vrf POD{{spod}}
     address-family ipv4 unicast
       advertise l2vpn evpn
<SNIP>
 evpn
   vni 42{{spod}}51 l2
     rd auto
     route-target import {{spod}}:42{{spod}}51
     route-target export {{spod}}:42{{spod}}51
<SNIP>
```

Step 13: Configure the EBGP session for your POD VRF on E21 to leak the routes to the default VRF to provide external connectivity to DC2.

```
VXLAN-E21(config)# interface Ethernet1/21.{{spod}}
VXLAN-E21(config-subif)# description VRFsubif_to_C2_POD{{spod}}
VXLAN-E21(config-subif)# encapsulation dot1q 21{{spod}}
VXLAN-E21(config-subif)# vrf member POD{{spod}}
VXLAN-E21(config-subif)# ip address 172.21.{{spod}}.1/30
VXLAN-E21(config-subif)# no shutdown
VXLAN-E21(config-subif)# exit
VXLAN-E21(config)# router bgp 65201
VXLAN-E21(config-router)# vrf POD{{spod}}
VXLAN-E21(config-router-vrf)# address-family ipv4 unicast
VXLAN-E21(config-router-vrf-af)# advertise l2vpn evpn
VXLAN-E21(config-router-vrf-af)# neighbor 172.21.{{spod}}.2 remote-as 64439
VXLAN-E21(config-router-vrf-neighbor)# address-family ipv4 unicast

VXLAN-E21# show run vrf POD{{spod}}

!Command: show running-config vrf POD{{spod}}
!Time: Mon May 28 22:48:50 2018

version 7.0(3)I4(7)
 <SNIP>
 interface Ethernet1/21.{{spod}}
   vrf member POD{{spod}}
 <SNIP>
 router bgp 65201
   vrf POD{{spod}}
     address-family ipv4 unicast
       advertise l2vpn evpn
     neighbor 172.21.{{spod}}.2
       remote-as 64439
       address-family ipv4 unicast

VXLAN-E21# show ip interface brief vrf POD{{spod}}
 IP Interface Status for VRF "POD{{spod}}"(4)
 Interface            IP Address      Interface Status
 Vlan{{spod}}00              forward-enabled protocol-up/link-up/admin-up
 Eth1/21.{{spod}}            172.21.{{spod}}.1      protocol-up/link-up/admin-up

VXLAN-E21# show bgp sessions vrf POD{{spod}}
 Total peers 36, established peers 36
 ASN 65201
 VRF POD{{spod}}, local ASN 65201
 peers 1, established peers 1, local router-id 172.21.{{spod}}.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)
 172.21.{{spod}}.2       64439 0     2d04h   |00:00:42|00:00:18 E   18816/179        0/0
```

Step 14: Configure the EBGP session for your POD VRF on E22 to leak the routes to the default VRF to provide external connectivity to DC2.

```
VXLAN-E22(config)# interface Ethernet1/26.{{spod}}
VXLAN-E22(config-subif)# description VRFsubif_to_C2_POD{{spod}}
VXLAN-E22(config-subif)# encapsulation dot1q 26{{spod}}
VXLAN-E22(config-subif)# vrf member POD{{spod}}
VXLAN-E22(config-subif)# ip address 172.22.{{spod}}.1/30
VXLAN-E22(config-subif)# no shutdown
VXLAN-E22(config-subif)# router bgp 65201
VXLAN-E22(config-router)# vrf POD{{spod}}
VXLAN-E22(config-router-vrf)# address-family ipv4 unicast
VXLAN-E22(config-router-vrf-af)# advertise l2vpn evpn
VXLAN-E22(config-router-vrf-af)# neighbor 172.22.{{spod}}.2 remote-as 64439
VXLAN-E22(config-router-vrf-neighbor)# address-family ipv4 unicast

VXLAN-E22# show run vrf POD{{spod}}

!Command: show running-config vrf POD{{spod}}
!Time: Mon May 28 22:48:50 2018

version 7.0(3)I4(7)
 <SNIP>
 interface Ethernet1/26.{{spod}}
   vrf member POD{{spod}}
 <SNIP>
 router bgp 65201
   vrf POD{{spod}}
     address-family ipv4 unicast
       advertise l2vpn evpn
     neighbor 172.22.{{spod}}.2
       remote-as 64439
       address-family ipv4 unicast

VXLAN-E22# show ip interface brief vrf POD{{spod}}
 IP Interface Status for VRF "POD{{spod}}"(4)
 Interface            IP Address      Interface Status
 Vlan{{spod}}00              forward-enabled protocol-up/link-up/admin-up
 Eth1/26.{{spod}}            172.22.{{spod}}.1      protocol-up/link-up/admin-up

VXLAN-E22# sh bgp sessions vrf POD{{spod}}
 Total peers 36, established peers 36
 ASN 65201
 VRF POD{{spod}}, local ASN 65201
 peers 1, established peers 1, local router-id 172.22.{{spod}}.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)
 172.22.{{spod}}.2      64439 0     2d04h   |00:00:52|00:00:29 E   50514/179        0/0
```

Traffic Flow Verification

Use Case 3

Traffic Flow Verification

Step 1: Use ping to test connectivity between the Ubuntu-S{{spod}} VM and Ubuntu-V{{spod}} and Ubuntu-I VMs.

```
VXLAN\POD{{spod}}user@ubuntu-S{{spod}}:~$ ping 10.{{spod}}.51.11 ← ping external host Ubun
PING 75.162.84.11 (10.2.51.11) 56(84) bytes of data.
64 bytes from 10.{{spod}}.51.11: icmp_req=1 ttl=60 time=0.367 ms
64 bytes from 10.{{spod}}.51.11: icmp_req=2 ttl=60 time=0.409 ms
64 bytes from 10.{{spod}}.51.11: icmp_req=3 ttl=60 time=0.436 ms
64 bytes from 10.{{spod}}.51.11: icmp_req=4 ttl=60 time=0.445 ms
^C
--- 10.{{spod}}.51.11 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3000ms

VXLAN\POD{{spod}}user@ubuntu-S{{spod}}:~$ ping 75.162.84.11 ← ping external host Ubuntu-I
PING 75.162.84.11 (75.162.84.11) 56(84) bytes of data.
64 bytes from 75.162.84.11: icmp_req=1 ttl=60 time=0.367 ms
64 bytes from 75.162.84.11: icmp_req=2 ttl=60 time=0.409 ms
64 bytes from 75.162.84.11: icmp_req=3 ttl=60 time=0.436 ms
64 bytes from 75.162.84.11: icmp_req=4 ttl=60 time=0.445 ms
^C
--- 75.162.84.11 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3000ms
```

Step 2: Confirm that all of the leaf switches (L11, L12, L13, L21) are all NVE peers of each other.

```
VXLAN-L11# show nve peers
Interface Peer-IP          State LearnType Uptime   Router-Mac
--------- ---------------  ----- --------- -------- -----------------
nve1      103.1.1.1        Up    CP        3d14h    58f3.9ca3.4603
nve1      104.1.1.1        Up    CP        3d14h    6412.2575.1431
nve1      105.1.1.1        Up    CP        3d14h    6412.2575.0641
nve1      109.1.1.1        Up    CP        3d14h    7426.ac76.6c1f
nve1      110.1.1.1        Up    CP        3d14h    6412.2575.13c1
nve1      111.1.1.1        Up    CP        3d14h    74a0.2fdf.0b73
```

```
VXLAN-L12# show nve peers
Interface Peer-IP          State LearnType Uptime   Router-Mac
--------- ---------------  ----- --------- -------- -----------------
nve1      103.1.1.1        Up    CP        3d14h    58f3.9ca3.4603
nve1      104.1.1.1        Up    CP        3d14h    6412.2575.1431
nve1      105.1.1.1        Up    CP        3d14h    6412.2575.0641
nve1      109.1.1.1        Up    CP        3d14h    7426.ac76.6c1f
nve1      110.1.1.1        Up    CP        3d14h    6412.2575.13c1
nve1      111.1.1.1        Up    CP        3d14h    74a0.2fdf.0b73
```

```
VXLAN-L13# show nve peers
Interface Peer-IP          State LearnType Uptime   Router-Mac
--------- ---------------  ----- --------- -------- -----------------
nve1      101.1.1.10       Up    CP        3d14h    f8c2.8887.d36f
nve1      104.1.1.1        Up    CP        3d14h    6412.2575.1431
nve1      105.1.1.1        Up    CP        3d14h    6412.2575.0641
nve1      109.1.1.1        Up    CP        3d14h    7426.ac76.6c1f
nve1      110.1.1.1        Up    CP        3d14h    6412.2575.13c1
nve1      111.1.1.1        Up    CP        3d14h    74a0.2fdf.0b73
```

```
VXLAN-L21# show nve peers
Interface Peer-IP          State LearnType Uptime   Router-Mac
--------- ---------------  ----- --------- -------- -----------------
nve1      101.1.1.10       Up    CP        3d14h    f8c2.8887.d36f
nve1      103.1.1.1        Up    CP        3d14h    58f3.9ca3.4603
nve1      104.1.1.1        Up    CP        3d14h    6412.2575.1431
nve1      105.1.1.1        Up    CP        3d14h    6412.2575.0641
nve1      109.1.1.1        Up    CP        3d14h    7426.ac76.6c1f
nve1      110.1.1.1        Up    CP        3d14h    6412.2575.13c1
```

Step 3: Verify that the switches are using ingress replication for BUM traffic.

```
VXLAN-L11# show nve vni ingress-replication 42{{spod}}51
Interface VNI      Replication List  Source  Up Time
--------- -------- ----------------- ------- -------
nve1      42{{spod}}51    103.1.1.1         IMET    2d00h
                  111.1.1.1         IMET    1d19h
```

```
VXLAN-L12# show nve vni ingress-replication 42{{spod}}51
Interface VNI      Replication List  Source  Up Time
--------- -------- ----------------- ------- -------
nve1      42{{spod}}51    103.1.1.1         IMET    2d00h
                  111.1.1.1         IMET    1d19h
```

```
VXLAN-L13# show nve vni ingress-replication 42{{spod}}51
Interface VNI      Replication List  Source  Up Time
--------- -------- ----------------- ------- -------
nve1      42{{spod}}51    101.1.1.10        IMET    2d00h
                  111.1.1.1         IMET    1d19h
```

```
VXLAN-L21# show nve vni ingress-replication 42{{spod}}51
Interface VNI      Replication List  Source  Up Time
--------- -------- ----------------- ------- -------
nve1      42{{spod}}51    101.1.1.10        IMET    1d19h
                  103.1.1.1         IMET    1d19h
```

Step 4: Verify that the switches are using ARP suppression.

```
VXLAN-L11# show ip arp suppression-cache summary

IP ARP suppression-cache Summary
Remote               :39
Local                :25
Total                :64

VXLAN-L11# show ip arp suppression-cache statistics

ARP packet statistics for suppression-cache

Suppressed:
Total 93176, Requests 70159, Requests on L2 34, Gratuitous 22983, Gratuitous on L2 0
Forwarded :
Total: 880
 L3 mode :       Requests 420, Replies 45
                 Request on core port 420, Reply on core port 45
                 Dropped 0
 L2 mode :       Requests 409, Replies 6
                 Request on core port 409, Reply on core port 2
                 Dropped 0
Received:
Total: 165747
 L3 mode:        Requests 142156, Replies 161
                 Local Request 71577, Local Responses 116
                 Gratuitous 22983, Dropped 0
 L2 mode :       Requests 443, Replies 4
                 Gratuitous 0, Dropped 0
ARP suppression-cache Local entry statistics
Adds 274117, Deletes 0
```

```
VXLAN-L12# show ip arp suppression-cache summary

IP ARP suppression-cache Summary
Remote               :28
Local                :17
Total                :45

VXLAN-L12# show ip arp suppression-cache statistics

ARP packet statistics for suppression-cache
Suppressed:
Total 44878, Requests 29767, Requests on L2 0, Gratuitous 15111, Gratuitous on L2 0
Forwarded :
Total: 129
 L3 mode :       Requests 119, Replies 10
                 Request on core port 119, Reply on core port 10
                 Dropped 0
 L2 mode :       Requests 0, Replies 0
                 Request on core port 0, Reply on core port 0
                 Dropped 0
Received:
Total: 67325
 L3 mode:        Requests 52057, Replies 157
                 Local Request 22171, Local Responses 147
                 Gratuitous 15111, Dropped 0
 L2 mode :       Requests 0, Replies 0
                 Gratuitous 0, Dropped 0
ARP suppression-cache Local entry statistics
Adds 80550, Deletes 1
```

```
VXLAN-L13# show ip arp suppression-cache summary

IP ARP suppression-cache Summary
Remote               :39
Local                :49
Total                :88

VXLAN-L13# show ip arp suppression-cache statistics

ARP packet statistics for suppression-cache

Suppressed:
Total 154003, Requests 153904, Requests on L2 99, Gratuitous 0, Gratuitous on L2 0
Forwarded :
Total: 9107
 L3 mode :      Requests 2492, Replies 1
                Request on core port 2492, Reply on core port 1
                Dropped 0
 L2 mode :      Requests 6612, Replies 2
                Request on core port 6612, Reply on core port 1
                Dropped 0
Received:
Total: 301614
 L3 mode:       Requests 294744, Replies 158
                Local Request 138348, Local Responses 157
                Gratuitous 0, Dropped 0
 L2 mode :      Requests 6711, Replies 1
                Gratuitous 0, Dropped 0
ARP suppression-cache Local entry statistics
Adds 301614, Deletes 0
```

```
VXLAN-L21# show ip arp suppression-cache summary

IP ARP suppression-cache Summary
Remote               :28
Local                :14
Total                :42

VXLAN-L21# show ip arp suppression-cache statistics

ARP packet statistics for suppression-cache
Suppressed:
Total 3439, Requests 3439, Requests on L2 0, Gratuitous 0, Gratuitous on L2 0
Forwarded :
Total: 106
 L3 mode :      Requests 90, Replies 2
                Request on core port 90, Reply on core port 2
                Dropped 0
 L2 mode :      Requests 14, Replies 0
                Request on core port 14, Reply on core port 0
                Dropped 0

Received:
Total: 5366
 L3 mode:       Requests 3559, Replies 1793
                Local Request 30, Local Responses 1791
                Gratuitous 0, Dropped 0
 L2 mode :      Requests 14, Replies 0
                Gratuitous 0, Dropped 0
ARP suppression-cache Local entry statistics
Adds 5366, Deletes 0
```

Summary

In summary, traffic is now flowing from VM Ubuntu-S{{spod}} to the internet IP address and to Ubuntu-V{{spod}} via the Multipod fabric. All of the leaf switches are NVE peers of each other, so they are all in one big fabric.

Summary and Survey

Summary

This lab session demonstrated deploying VXLAN (Virtual Extensible LAN) in a data center using hardware VXLAN endpoints (VTEPs) with the Nexus 9000 series platform. The lab focused on multiple scenarios using the hardware VTEPs. You should now have the skills to implement a VXLAN solution in your network.

If you did not have time to complete all three sections of the lab, we have granted extra time to finish your lab. You can access your pod anytime for the next 7 days. If you would like to redo a portion of the lab later, please request for pod clean up. To request that, either ask a proctor or email us using the contact information on the welcome page.

Session Evaluations

We sincerely hope you enjoyed the lab. **Your feedback is invaluable to us**. Feedback from students who take the lab are what help us to shape the lab for students at future events and the scores are used as a major determining factor when deciding which labs will be allowed to present at future events. **Please complete the surveys via CiscoLive.com/online or in the mobile app.**