

# BGP-EVPN VxLAN Lab - Part 1 - Intra Vlan Routing

## CSE PRACTICALS

visit <http://www.csepracticals.com> 25+ Courses on System Programming and Network Development

Udemy Link : <https://www.udemy.com/user/abhishek-sagar-8/>

---

## Syllabus

[Chapter 1 Intra-Vlan Routing Lab \(this \)](#)

[Chapter 2 BGP-EVPN Control Plane & Route Advertisement](#)

[Chapter 3 Inter-VNI Routing](#)

[Chapter 4 External Connectivity](#)

---

## Introduction

This is a detailed GNS3 LAB in which **BGP-EVPN VxLAN** is Implemented on Cisco NXOSv L3 Switches. We Explain the Concept involved in BGP-EVPN VxLAN, show the various output, and packet captures, and explain the data plane operations.

Why VxLAN is introduced, and what problem it solves comes under theory. You are recommended to cover the ***Why VxLAN part from standard books or otherwise.***

**Software Used:** GNS3 version 2.2.39 ( *You can use latest available on GNS3 website* )

**Virtualization:** VMWare workstation PRO ( *You can use the latest available on the VMWare website.* VMWare WS PRO is a paid software, go for *Work Station Player* instead which is free )

**System:** My system has 64 GB RAM. At least 16 GB RAM is recommended to run this LAB.

**Cisco NXOSv Image:** [NXOSv9k-93000v-10.1.1.1](#) (This is L3 Switch)

A Great Resource of GNS3 Image Collection is [here](#)

The NXOSv Image that I used in this lab is [here](#) .

For this LAB you are recommended to use same version of NXOSv Image as mine to avoid any discrepancy during lab time.

**Books:** We use two Books as a reference :

[Building Data Centers with VxLAN BGP EVPN \( Cisco Book \)](#)

[Virtual Extensible LAN - VxLA A Practical Guide](#) ( *Strongly Recommend this one , I followed this book Most of the time* )

[Troubleshooting PPT](#) ( *Somebody created this nice PPT on VxLAN* )

---

## Pre-Requisites

To do this Tutorial You must know the following :

A solid understanding of ARP and PING functionality is essential.

Proficiency in L2 switching and L3 routing for IPV4 packets is crucial.

Basic knowledge of multicast protocols such as PIM and IGMP is necessary.

Experience with network simulation tools like GNS3 or Eve-NG is beneficial, especially for those new to these platforms.

Familiarity with any routing protocol, including OSPF, ISIS, or RIP, is advantageous.

Having a willingness to learn and the ability to ask questions promptly will greatly aid in your professional development.

---

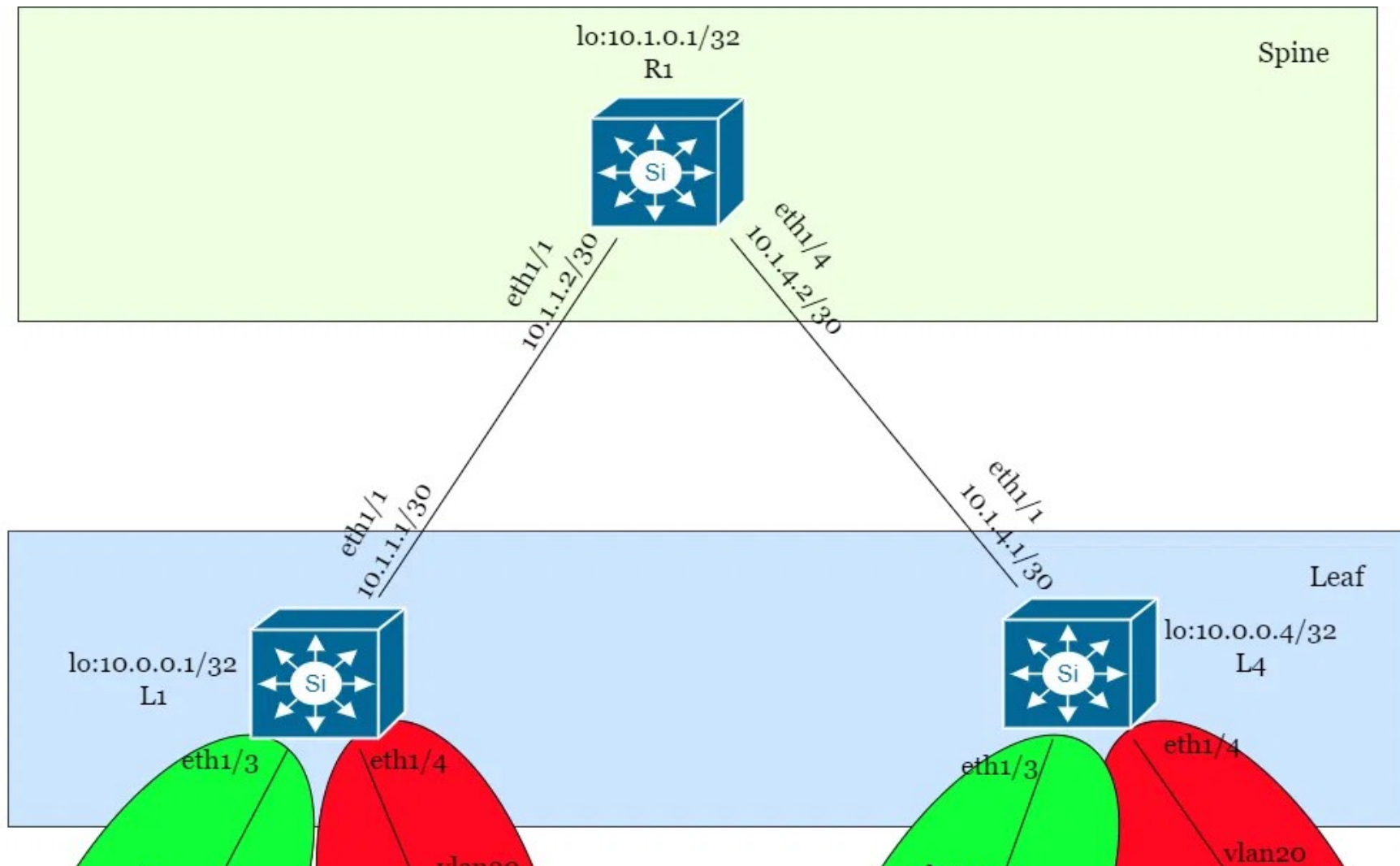
## Topology

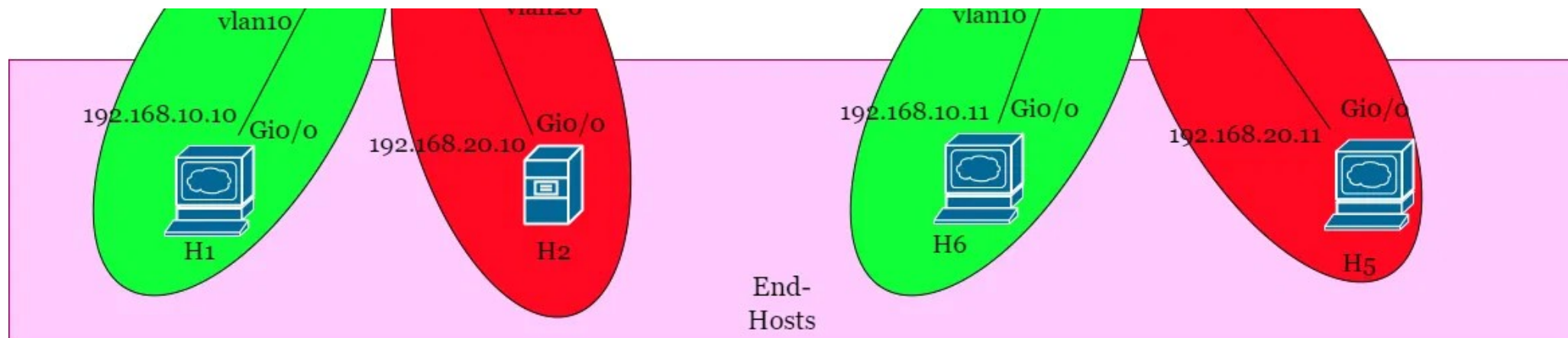
We will construct the exactly below topology.

You are strongly recommended to establish the connections and name the interfaces exactly in the same way as is done in the topology, so that our discussion during LABs aligns well.

Configure the same IP Addresses as shown in the topology. Do not Deviate.

The Topology is Layered - **Spine layer**, **Leaf Layer** and **End host Layer** as you can see in the Diagram.





## Lab Topology

At Minimum, you must deploy **R1, L1,L4, H1, H2, H5, H6** nodes. We will not use the other Nodes in this tutorial - L2 and L3, H3, H4 but you are free to extend the topology as you wish to experiment more.

All Spine and Leaf Nodes are the same NXOSv L3 Switches. So, Total 3 NXOSv nodes you need to deploy - R1, L1, and L4.

End-Hosts could be any node that supports minimum Network functionality like ping, ARP, etc. You can use VPCS hosts (come inbuilt with GNS3, no need to install anything). In my lab, I prefer to use Cisco ASAv Firewall Images. The end host Image type doesn't matter.

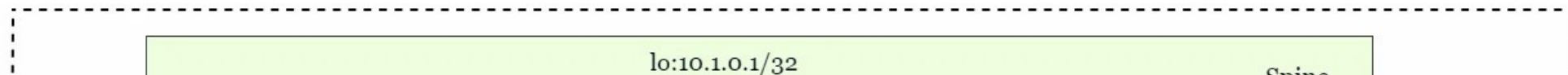
End Hosts H1 and H2 are sitting behind Leaf node L1

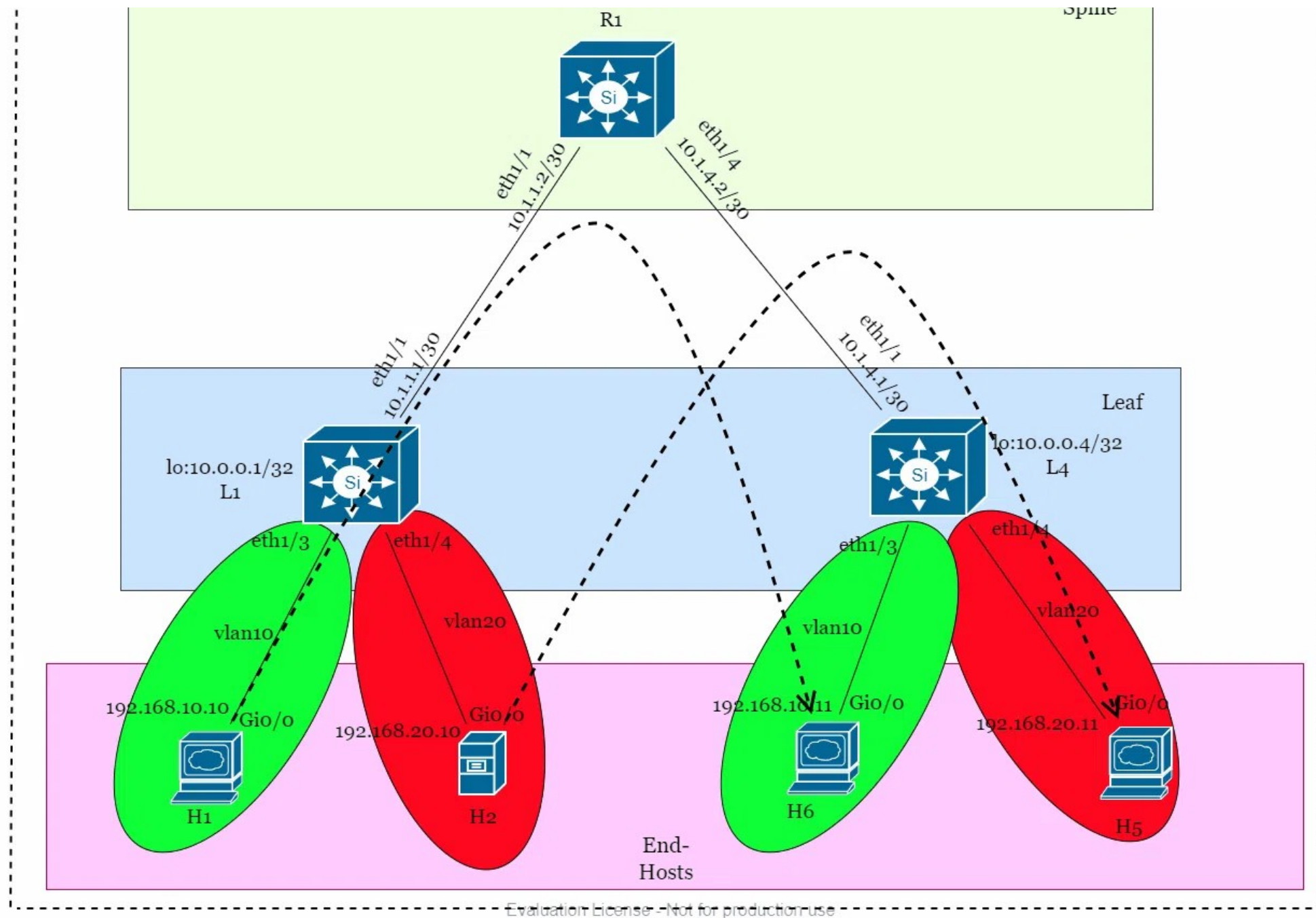
End Hosts H5 and H6 are sitting behind Leaf node L4

## End Goal of the LAB

To put it in a few words, our Goal is to :

*Make ping 192.168.10.11 success from H1 node ( i.e. ping H6 from H1 ) - This covers intra-vlan routing. So, we want to ping from a machine in vlan 10 to another machine in vlan 10, but both vlan 10s are separated by VxLAN network/fabric*





## Communication Over VxLAN Network ( Intra-Vlan )

---

### Topology Base Configuration ( Under Lay Configuration )

All the links between Spine R1 and Leafs L1 and L4 are Layer 3 Links.

We need to deploy Unicast Configuration and Multicast Configuration to start with.

### Unicast Configuration

( to be done on Spine and Leafs )

The objective of unicast configuration is to simply setup IP reachability among all spine and leaf nodes in the network.

We need to configure IP address on all interfaces connecting Spines and Leafs

We need to configure loopback address 0 on all nodes with /32 loopback address

Deploy Routing protocol ( OSPF, ISIS ) between spine and Leafs so that every node is reachable to every other node

### Configs :

Spine R1	Leaf L1	Leaf L4
<pre>conf hostname spine1 #boot nxos nxos.10.1.1.bin feature ospf router ospf SI-Underlay router-id 10.1.0.1 end  conf interface loopback 0 no sh ip address 10.1.0.1/32 ip router ospf SI-Underlay area 0</pre>	<pre>conf hostname leaf1 #boot nxos nxos.10.1.1.bin feature ospf router ospf SI-Underlay router-id 10.0.0.1 end  conf interface loopback 0 no sh ip address 10.0.0.1/32 ip router ospf SI-Underlay area 0</pre>	<pre>conf hostname leaf4 #boot nxos nxos.10.1.1.bin feature ospf router ospf SI-Underlay router-id 10.0.0.4 end  conf interface loopback 0 no sh ip address 10.0.0.4/32 ip router ospf SI-Underlay area 0</pre>

<pre> ip ospf network point-to-point end  conf interface ethernet 1/1 description to-leaf1-eth1/1 no sh no sw ip address 10.1.1.2/30 ip router ospf SI-Underlay area 0 ip ospf network point-to-point mtu 9000  interface ethernet 1/2 description to-leaf2-eth1/1 no sw ip address 10.1.2.2/30 ip router ospf SI-Underlay area 0 ip ospf network point-to-point mtu 9000 no sh  interface ethernet 1/3 description to-leaf3-eth1/1 no sw ip address 10.1.3.2/30 ip router ospf SI-Underlay area 0 ip ospf network point-to-point mtu 9000 no sh  interface ethernet 1/4 description to-leaf4-eth1/1 no sw ip address 10.1.4.2/30 ip router ospf SI-Underlay area 0 ip ospf network point-to-point mtu 9000 no sh end </pre>	<pre> ip ospf network point-to-point end  conf interface ethernet 1/1 description to-spine1-eth1/0 no sh no sw ip address 10.1.1.1/30 ip router ospf SI-Underlay area 0 ip ospf network point-to-point mtu 9000  interface ethernet 1/2 description to-spine2-eth1/0 no sw ip address 10.2.1.1/30 ip router ospf SI-Underlay area 0 ip ospf network point-to-point mtu 9000 no sh end </pre>	<pre> ip ospf network point-to-point end  conf interface ethernet 1/1 description to-spine1-eth1/3 no sh no sw ip address 10.1.4.1/30 ip router ospf SI-Underlay area 0 ip ospf network point-to-point mtu 9000  interface ethernet 1/2 description to-spine2-eth1/3 no sw ip address 10.2.4.1/30 ip router ospf SI-Underlay area 0 ip ospf network point-to-point mtu 9000 no sh end </pre>
--	--	--

## Verification :

Ping loopback address from Any node to any other node ( Amongst R1 , L1 and L4 ). This completes the unicast Routing protocol Deployment.



## Multicast Configuration

( to be done on Spine and Leafs )

The objective of multicast configuration is to efficiently distribute the BUM (unknown Broadcast, Unicast, or Multicast) traffic

Enable PIM Sparse mode on all interfaces connecting spines and leafs

Configure Spine the Rendezvous Point ( RP )

Make Leafs ( L1 and L4) the LHR ( Last hop Router ) for Mcast Group 239.1.1.1 and 239.2.2.2

### Configs :

Spine R1	Leaf L1	Leaf L4
<pre> conf feature pim  ip pim rp-address 10.100.100.100 ip pim anycast-rp 10.100.100.100 10.1.0.1 ip pim anycast-rp 10.100.100.100 10.1.0.2  interface loopback 0 ip pim sparse-mode  interface loopback 1 ip pim sparse-mode description anycast-rp ip address 10.100.100.100/32 no sh ip router ospf SI-Underlay area 0  interface ethernet 1/1 ip pim sparse-mode  interface ethernet 1/2 ip pim sparse-mode  interface ethernet 1/3 ip pim sparse-mode  interface ethernet 1/4 </pre>	<pre> conf feature pim  ip pim rp-address 10.100.100.100  interface loopback 0 ip pim sparse-mode  interface ethernet 1/1 ip pim sparse-mode  interface ethernet 1/2 ip pim sparse-mode end </pre>	<pre> conf feature pim  ip pim rp-address 10.100.100.100  interface loopback 0 ip pim sparse-mode  interface ethernet 1/1 ip pim sparse-mode  interface ethernet 1/2 ip pim sparse-mode end </pre>



```
ip pim sparse-mode
end
```

### Verification :

Leaf L1 : ping multicast 239.1.1.1 interface ethernet 1/1

```
leaf1# ping multicast 239.1.1.1 interface ethernet 1/1
PING 239.1.1.1 (239.1.1.1): 56 data bytes
Request 0 timed out
64 bytes from 10.1.4.1: icmp_seq=1 ttl=253 time=25.798 ms
64 bytes from 10.1.4.1: icmp_seq=2 ttl=253 time=26.01 ms
64 bytes from 10.1.4.1: icmp_seq=3 ttl=253 time=30.025 ms
64 bytes from 10.1.4.1: icmp_seq=4 ttl=253 time=20.754 ms
--- 239.1.1.1 ping multicast statistics ---
5 packets transmitted,
From member 10.1.4.1: 4 packets received, 20.00% packet loss
--- in total, 1 group member responded ---
```

Leaf L1 : ping multicast 239.2.2.2 interface ethernet 1/1

```
leaf1# ping multicast 239.1.1.1 interface ethernet 1/1
PING 239.1.1.1 (239.1.1.1): 56 data bytes
Request 0 timed out
```

```
64 bytes from 10.1.4.1: icmp_seq=1 ttl=253 time=25.798 ms
64 bytes from 10.1.4.1: icmp_seq=2 ttl=253 time=26.01 ms
64 bytes from 10.1.4.1: icmp_seq=3 ttl=253 time=30.025 ms
64 bytes from 10.1.4.1: icmp_seq=4 ttl=253 time=20.754 ms
--- 239.1.1.1 ping multicast statistics ---
5 packets transmitted,
From member 10.1.4.1: 4 packets received, 20.00% packet loss
--- in total, 1 group member responded ---
```

Leaf L4 : ping multicast 239.1.1.1 interface ethernet 1/1

```
leaf4# ping multicast 239.1.1.1 interface ethernet 1/1
PING 239.1.1.1 (239.1.1.1): 56 data bytes
Request 0 timed out
64 bytes from 10.1.1.1: icmp_seq=1 ttl=253 time=59.807 ms
64 bytes from 10.1.1.1: icmp_seq=2 ttl=253 time=14.429 ms
64 bytes from 10.1.1.1: icmp_seq=3 ttl=253 time=12.342 ms
64 bytes from 10.1.1.1: icmp_seq=4 ttl=253 time=11.522 ms
--- 239.1.1.1 ping multicast statistics ---
5 packets transmitted,
From member 10.1.1.1: 4 packets received, 20.00% packet loss
```

```
|--- in total, 1 group member responded ---  
Leaf L4 : ping multicast 239.2.2.2 interface ethernet 1/1  
leaf4# ping multicast 239.2.2.2 interface ethernet 1/1  
PING 239.2.2.2 (239.2.2.2): 56 data bytes  
Request 0 timed out  
64 bytes from 10.1.1.1: icmp_seq=1 ttl=253 time=61.027 ms  
64 bytes from 10.1.1.1: icmp_seq=2 ttl=253 time=12.879 ms  
64 bytes from 10.1.1.1: icmp_seq=3 ttl=253 time=25.466 ms  
64 bytes from 10.1.1.1: icmp_seq=4 ttl=253 time=29.776 ms  
--- 239.2.2.2 ping multicast statistics ---  
5 packets transmitted,  
From member 10.1.1.1: 4 packets received, 20.00% packet loss  
--- in total, 1 group member responded ---  
leaf4#
```

Checking Multicast Routes on Leaf L1, L4 and Spine R1

---

## VxLAN Configuration ( Overlay Configuration )

Now, We need to do basic VxLAN configuration. VxLAN Configuration needs to be done only on LEAF Switches ( L1 and L4 )

After this VxLAN configuration, End Hosts in same Vlan must be able to ping each other.

## Configs:

Leaf L1	Leaf L4
<pre> conf feature interface-vlan feature nv overlay feature vn-segment-vlan-based  interface nve 1 no sh source loopback 0 member vni 5010 mcast-group 239.1.1.1 member vni 5020 mcast-group 239.2.2.2  vlan 10 vn-segment 5010  vlan 20 vn-segment 5020  interface ethernet 1/3 sw mode access sw access vlan 10  interface ethernet 1/4 sw mode access sw access vlan 20 end                     </pre>	<pre> conf feature interface-vlan feature nv overlay feature vn-segment-vlan-based  interface nve 1 no sh source loopback 0 member vni 5010 mcast-group 239.1.1.1 member vni 5020 mcast-group 239.2.2.2  vlan 10 vn-segment 5010  vlan 20 vn-segment 5020  interface ethernet 1/3 sw mode access sw access vlan 10  interface ethernet 1/4 sw mode access sw access vlan 20 end                     </pre>

## Verification :

From Host H1, ping 192.168.10.11

From Host H2, ping 192.168.20.11

From Host H5, ping 192.168.20.10

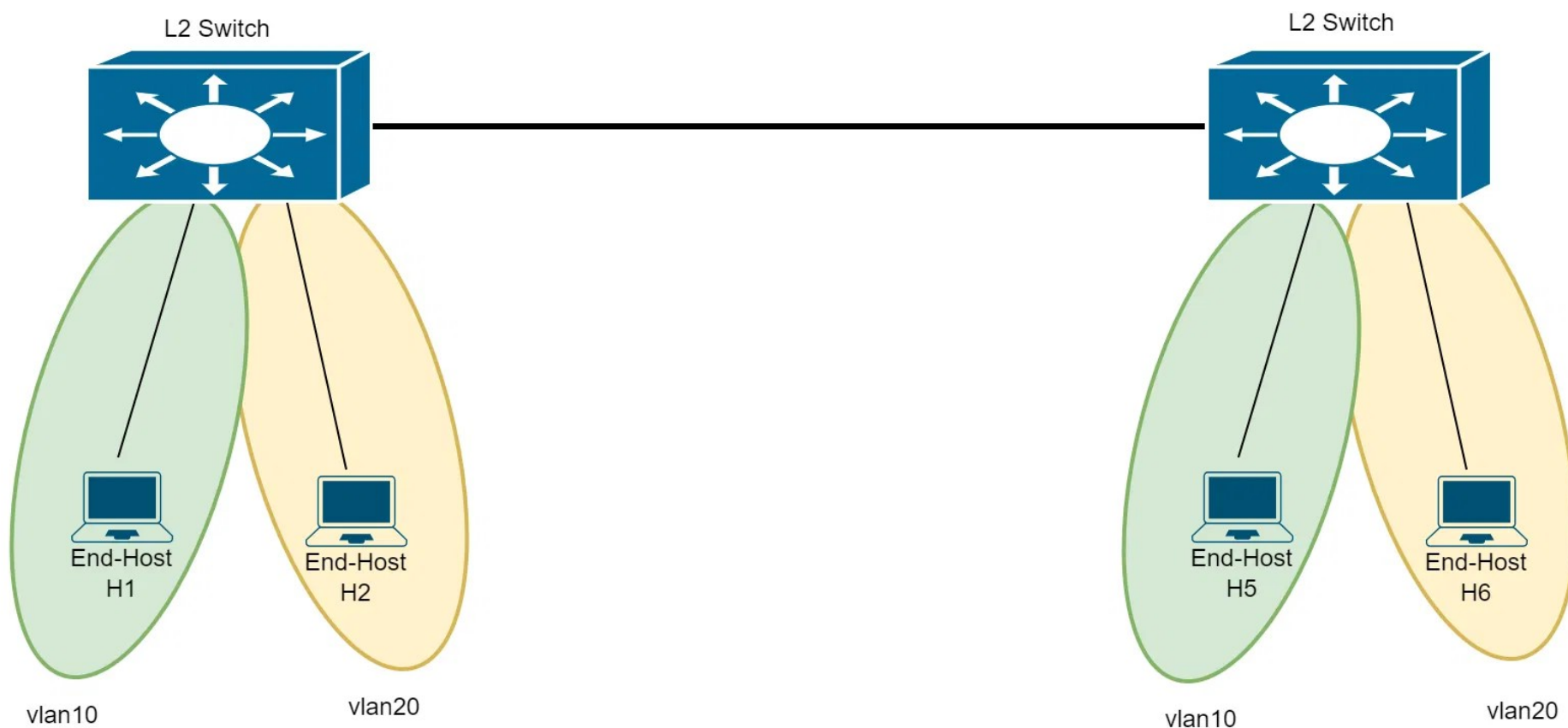
From Host H6, ping 192.168.10.10

If all Pings are Successful, it means, our VxLAN deployment is successful. So, far, our VxLAN deployment supports only Intra-Vlan Routing (

communication within same Vlan ). You can realize that VLANs have been extended i.e. same vlan , say, vlan 10, is now present behind multiple Leafs - L1 and L4. Same is true for vlan 20.

Also, BUM traffic is handled through multicast ( as opposed to **ingress replication** which is not an efficient way of handling BUM traffic )

If there are 1000 leafs ( also called VTEPs ( Virtual Tunneling End Points ) ) in the Data Center, there can be 1000 vlan-10 segments, 1 sitting behind each leaf. All hosts in those Vlan segments can seamlessly communicate as if they are present in the same traditional vlan segment tied to same L2VNI.



## Traditional Switching

---

### Data Plane Operations ( Intra Vlan Routing )

In this section, we will understand step by step how Intra-vlan communication is achieved in VxLAN-based networks.

Let, say, Host H1 ping Host H6

H1 : ping 192.168.10.11

Let us see what are the operations involved in making the above ping work.

*Suggestion: Please, stick with L2 switching and L3 routing fundamentals to understand the data-plane operations!*

### Initial State Inspection

Let us see the state of Leafs/VTEPs L1 and L4 before User pings

The tables involved are :

**Mac address Table on Leafs ( show mac address table )**

**Multicast Routes on Leafs and Spines ( show ip mroute )**

**ARP Table on Hosts ( show arp )**

L3 Unicast Routing Table is not involved since we are doing **intra-vlan communication** here.

Mac Addresses :

Mac H1 Gi 0/0 - 0c:08:4e:22:00:01

Mac H2 Gi 0/0 - 0c:fb:bb:4c:00:01

Mac H5 Gi 0/0 - 0c:b8:f4:23:00:01

Mac H6 Gi 0/0 - 0c:8a:0b:b7:00:01

## MAC Address Table on Leafs

Leaf L1	Leaf L2
<pre>leaf1# show mac address-table 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, ~ - vsan   VLAN    MAC Address      Type      age      Secure NTFY Ports -----+-----+-----+-----+-----+-----+----- G    -    0ca5.0000.1b08    static    -          F      F      sup-eth1(R) leaf1#</pre> <p>Mac-Addr_table-L1-Initial</p>	<pre>leaf4# show mac address-table 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, ~ - vsan   VLAN    MAC Address      Type      age      Secure NTFY Ports -----+-----+-----+-----+-----+-----+----- G    -    0c48.0000.1b08    static    -          F      F      sup-eth1(R) leaf4#</pre> <p>Mac-Addr_table-L4-Initial</p>
<p>Empty Table</p> <p>The only entry is the mac address which is called Router mac, which we will discuss later</p>	<p>Empty Table</p> <p>The only entry is the mac address which is called Router mac, which we will discuss later</p>

## Multicast Routes

### Multicast Routes on Leaf L1

Leaf L1	Explanation
---------	-------------



```
leaf1# show ip mroute
IP Multicast Routing Table for VRF "default"

(*, 232.0.0.0/8), uptime: 00:01:24, pim ip
Incoming interface: Null, RPF nbr: 0.0.0.0
Outgoing interface list: (count: 0)

(*, 239.1.1.1/32), uptime: 00:01:24, nve pim ip
Incoming interface: Ethernet1/1, RPF nbr: 10.1.1.2
Outgoing interface list: (count: 1)
nve1, uptime: 00:01:24, nve

(10.0.0.1/32, 239.1.1.1/32), uptime: 00:01:24, nve mrrib pim ip
Incoming interface: loopback0, RPF nbr: 10.0.0.1
Outgoing interface list: (count: 1)
Ethernet1/1, uptime: 00:00:41, pim

(*, 239.2.2.2/32), uptime: 00:01:24, nve pim ip
Incoming interface: Ethernet1/1, RPF nbr: 10.1.1.2
Outgoing interface list: (count: 1)
nve1, uptime: 00:01:24, nve

(10.0.0.1/32, 239.2.2.2/32), uptime: 00:01:24, nve mrrib pim ip
Incoming interface: loopback0, RPF nbr: 10.0.0.1
Outgoing interface list: (count: 1)
Ethernet1/1, uptime: 00:00:40, pim

leaf1#
```

### L1-Mroute-Initial

Leaf is an Mcast LHR ( Last Hop Router ) for Groups 239.1.1.1 and 239.2.2.2

Therefore, two  $*$ ,  $G$  routes -  $*$ , 239.1.1.1 and  $*$ , 239.2.2.2 are created.

Note, that outgoing interface of these routes is nve1 which will decap the multicast packets.

These  $*$ ,  $G$  routes are responsible to process the ingress multicast packets.

```
interface nve 1
no sh
source loopback 0
member vni 5010 mcast-group 239.1.1.1
member vni 5020 mcast-group 239.2.2.2
```

Source address of the loo interface is **10.0.0.1**. Using this address as Source, L1 send dummy register packet to RP to complete the FHR (First hop Router) registration process proactively ( without actual app data involved ). It does this for both Groups , hence two more S,G routes are created on leaf

10.0.0.1/32 , 239.1.1.1 and 10.0.0.1/32 , 239.2.2.2 . These routes enable routing the locally-generated outbound multicast traffic from FHR to RP. Note that OIF of these routes is an interface towards RP. As a Part of registration process, these S = 10.0.0.1/32, G = 239.1.1.1 | 239.2.2.2 routes are created on Spine R1 also.

Thus  $*$ ,  $G$  routes enable processing of ingress mcast traffic

S,G routes enable routing of egress mcast traffic.

PIM Register packet send by Leaf L1 to RP for Group 239.1.1.1

*Note : A similar pkt is sent by Leaf L1 to RP for Group 239.2.2.2 also.*

```

> Frame 29: 62 bytes on wire (496 bits), 62 bytes captured (496 bits) on interface -, id 0
> Ethernet II, Src: 8c:a5:00:00:1b:00 (8c:a5:00:00:1b:00), Dst: 8c:a3:00:00:1b:00 (8c:a3:00:00:1b:00)
> Destination: 8c:a3:00:00:1b:00 (8c:a3:00:00:1b:00)
> Source: 8c:a5:00:00:1b:00 (8c:a5:00:00:1b:00)
Type: IPv4 (0x0000)
> Internet Protocol Version 4, Src: 10.0.0.1, Dst: 10.100.100.100 unicast pkt
0100 .... = Version: 4
... 0101 = Header Length: 20 bytes (5)
> Differentiated Services Field: 0xc0 (DSCP: CS6, ECN: Not-ECT)
Total Length: 48
Identification: 0xe095 (57493)
> 0000 .... = Flags: 0x0
... 0 0000 0000 0000 = Fragment Offset: 0
Time to Live: 255
Protocol: PIM (103)
Header Checksum: 0x61a0 [validation disabled]
[Header checksum status: Unverified]
Source Address: 10.0.0.1
Destination Address: 10.100.100.100
> Protocol Independent Multicast
0010 .... = Version: 2
... 0001 = Type: Register (1)
Reserved byte(s): 00
Checksum: 0x9eff [correct]
[Checksum Status: Good]
> PIM Options
> Internet Protocol Version 4, Src: 10.0.0.1, Dst: 239.1.1.1
0100 .... = Version: 4
... 0101 = Header Length: 20 bytes (5)
> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 20
Identification: 0x0000 (0)
> 0000 .... = Flags: 0x0
... 0 0000 0000 0000 = Fragment Offset: 0
Time to Live: 255
Protocol: PIM (103)
Header Checksum: 0xc17f [validation disabled]
[Header checksum status: Unverified]
Source Address: 10.0.0.1
Destination Address: 239.1.1.1

```

Encapsulated Mcast  
Pkt without any  
appin payload

Similarly, Leaf L4 will also have 4 Mcast routes in its MFIB. Two are \*, **G = 239.1.1.1 | 239.2.2.2.2** and two for **S = 10.0.0.4** , **G = 239.1.1.1 | 239.2.2.2**. Same explanation applies here.

## Multicast Routes on Leaf L4

Leaf L4	Explanation
<pre> leaf4# show ip mroute IP Multicast Routing Table for VRF "default"  (*, 232.0.0.0/0), uptime: 01:55:50, pim ip   Incoming interface: Null1, RPF nbr: 0.0.0.0   Outgoing interface list: (count: 0)  (*, 239.1.1.1/32), uptime: 01:55:44, nve pim ip   Incoming interface: Ethernet1/1, RPF nbr: 10.1.4.2   Outgoing interface list: (count: 1)     nve1, uptime: 01:55:44, nve  (10.0.0.4/32, 239.1.1.1/32), uptime: 01:55:44, nve mrib pim ip   Incoming interface: loopback0, RPF nbr: 10.0.0.4   Outgoing interface list: (count: 1)     Ethernet1/1, uptime: 00:07:20, pim  (*, 239.2.2.2/32), uptime: 01:55:44, nve pim ip   Incoming interface: Ethernet1/1, RPF nbr: 10.1.4.2   Outgoing interface list: (count: 1)     nve1, uptime: 01:55:44, nve  (10.0.0.4/32, 239.2.2.2/32), uptime: 01:55:44, nve mrib pim ip   Incoming interface: loopback0, RPF nbr: 10.0.0.4   Outgoing interface list: (count: 1)     Ethernet1/1, uptime: 00:07:20, pim </pre>	<p>Leaf is an Mcast LHR ( Last Hop Router ) for Groups 239.1.1.1 and 239.2.2.2</p> <p>Therefore, two : , <i>G routes</i> - *, <i>239.1.1.1</i> and *, <i>239.2.2.2</i> are created.</p> <p>Note, that outgoing interface of these routes is nve1 which will decap the multicast packets.</p> <p>These *, G routes are responsible to process the ingress multicast packets.</p> <pre> interface nve 1 no sh source loopback 0 member vni 5010 mcast-group 239.1.1.1 member vni 5020 mcast-group 239.2.2.2 </pre> <p>Source address of the loo interface is <b>10.0.0.4</b>. Using this address as Source, L4 send dummy register packet to RP to complete the FHR (First hop Router) registration process proactively ( without actual app data involved ). It does this for both Groups , hence two more S,G routes are created on leaf</p> <p>10.0.0.4/32 , 239.1.1.1 and 10.0.0.4/32 , 239.2.2.2 . These routes enable routing the locally-generated outbound</p>

L4-Mroute-Initial

	<p>multicast traffic from FHR to RP. Note that OIF of these routes is an interface towards RP. As a Part of the registration process, these S = 10.0.0.4/32, G = 239.1.1.1   239.2.2.2 routes are created on Spine R1 also.</p> <p>Thus *, G routes enable processing of ingress mcast traffic</p> <p>S,G routes enable routing of egress mcast traffic.</p>
--	--

#### Multicast Routes on spine R1

Spine R1	Explanation
----------	-------------

```

spinel# show ip mroute
IP Multicast Routing Table for VRF "default"

(*, 232.0.0.0/8), uptime: 00:22:44, pim ip
  Incoming interface: Null, RPF nbr: 0.0.0.0
  Outgoing interface list: (count: 0)

(*, 239.1.1.1/32), uptime: 00:12:28, pim ip
  Incoming interface: loopback1, RPF nbr: 10.100.100.100
  Outgoing interface list: (count: 2)
    Ethernet1/1, uptime: 00:11:16, pim
    Ethernet1/4, uptime: 00:12:27, pim

(10.0.0.1/32, 239.1.1.1/32), uptime: 00:12:29, pim mrib ip
  Incoming interface: Ethernet1/1, RPF nbr: 10.1.1.1, internal
  Outgoing interface list: (count: 2)
    Ethernet1/1, uptime: 00:10:26, pim, (RPF)
    Ethernet1/4, uptime: 00:12:27, pim

(10.0.0.4/32, 239.1.1.1/32), uptime: 00:11:37, pim mrib ip
  Incoming interface: Ethernet1/4, RPF nbr: 10.1.4.1, internal
  Outgoing interface list: (count: 2)
    Ethernet1/4, uptime: 00:10:36, pim, (RPF)
    Ethernet1/1, uptime: 00:11:16, pim

(*, 239.2.2.2/32), uptime: 00:12:29, pim ip
  Incoming interface: loopback1, RPF nbr: 10.100.100.100
  Outgoing interface list: (count: 2)
    Ethernet1/1, uptime: 00:11:16, pim
    Ethernet1/4, uptime: 00:12:27, pim

(10.0.0.1/32, 239.2.2.2/32), uptime: 00:12:28, pim mrib ip
  Incoming interface: Ethernet1/1, RPF nbr: 10.1.1.1, internal
  Outgoing interface list: (count: 2)
    Ethernet1/1, uptime: 00:10:25, pim, (RPF)
    Ethernet1/4, uptime: 00:12:27, pim

(10.0.0.4/32, 239.2.2.2/32), uptime: 00:11:36, pim mrib ip
  Incoming interface: Ethernet1/4, RPF nbr: 10.1.4.1, internal
  Outgoing interface list: (count: 2)
    Ethernet1/4, uptime: 00:10:35, pim, (RPF)
    Ethernet1/1, uptime: 00:11:16, pim

spinel#

```

R1-Mroute-Initial

Spine R1 receives , *G PIM Join for Group G = 239.1.1.1 and 239.2.2.2 from L1 and L2 on interface eth1/1 and eth1/4 respectively. Therefore, R1 creates two ,G routes :*

\*, G = 239.1.1.1 with OIF = eth1/1 and eth1/4.

\*, G = 239.2.2.2 with OIF = eth1/1 and eth1/4.

As a part of PIM registration process, Spine R1 also creates four S,G routes :

S = 10.0.0.1 , G = 239.1.1.1

S = 10.0.0.1 , G = 239.2.2.2

S = 10.0.0.4 , G = 239.1.1.1

S = 10.0.0.4 , G = 239.1.1.1

Using these S,G routes , Spine R1 receives Mcast traffic and forward them to respise outgoing interfaces.

## ping 192.168.10.11 on H1

### ARP Broadcast Request Packet

H1 checks its local ARP cache to find the mac address for 192.68.10.11 but doesn't find it ( since its a first ping )

H1 launches ARP-Broadcast Request pkt ( P1 ), since the destination is in the same subnet as Source Wireshark Capture P1.

```

> Frame 1: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface -, id 0
▼ Ethernet II, Src: 0c:08:4e:22:00:01 (0c:08:4e:22:00:01), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
  > Destination: Broadcast (ff:ff:ff:ff:ff:ff)
  > Source: 0c:08:4e:22:00:01 (0c:08:4e:22:00:01)
  Type: ARP (0x0806)
  Padding: 00000000000000000000000000000000
▼ Address Resolution Protocol (request)
  Hardware type: Ethernet (1)
  Protocol type: IPv4 (0x0800)
  Hardware size: 6
  Protocol size: 4
  Opcode: request (1)
  Sender MAC address: 0c:08:4e:22:00:01 (0c:08:4e:22:00:01)
  Sender IP address: 192.168.10.10
  Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
  Target IP address: 192.168.10.11

```

## Packet P1

P1 is received by L1 on port eth1/3 which is in vlan 10 access mode

P1 gets tagged with 802.1Q hdr with vlan 10

P1 is handled by Data Link Layer ( L2 Layer ).

L1 does Mac Learning

```
leaf1# show mac address-table
```

Legend:

\* - primary entry, G - Gateway MAC, (R) - Routed MAC, O - Overlay MAC

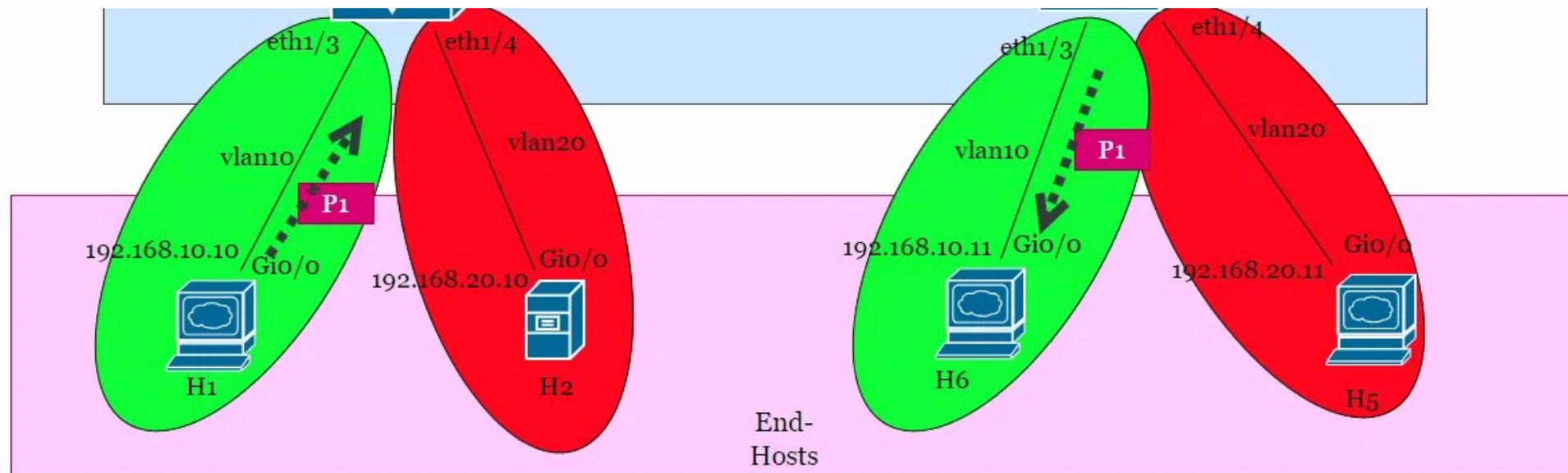
age - seconds since last seen,+ - primary entry using vPC Peer-Link,

(T) - True, (F) - False, C - ControlPlane MAC, ~ - vsan

VLAN	MAC	Address	Type	age	Secure	NTFY	Ports
------	-----	---------	------	-----	--------	------	-------







### Packet Flow - Intra Vlan Routing

**Data-Plane on Leaf maintains a table as below. keyed by vlan , keyed by vni ( for reverse mapping ).**

L1 : show nve internal platform interface nve 1 detail

Sw	BD	Vni	State	Intf	Type	Vrf-ID	Notified
10	5010	NONE	nve1	CP	0	No	
20	5020	NONE	nve1	CP	0	No	

**The above table's first entry says - Any packet that arrives on vlan 10 on Leaf L1 will be sent out of interface nve1 , using VNID as 5010**

**Any pkt sent out of interface nve1 shall be encapsulated as VxLAN pkt. The destination IP Address of the outer IP hdr of the pkt is not known from this table, it has to be provided as an additional parameter by the entity that is trying to use nve1 as an egress interface**

**Configuration on Leaf L1 responsible for building the above table. The same table exists on leaf L4.**



```

interface nve 1
    no sh
    source loopback 0
    member vni 5010 mcast-group 239.1.1.1
    member vni 5020 mcast-group 239.2.2.2
    vlan 10 vn-segment 5010
    vlan 20 vn-segment 5020

```

As a result of the Mac-table look-up, L1 Prepares the Pkt **P2** with the following fields ( Adding additional Headers in Green ).

L1 looks Up the Table which has vlan-vni mapping and knows that for vlan 10, VNI to be used is **5010**. Great !

The Destination IP Address to be used is **239.1.1.1** ( obtained from step 7 )

The source Address to be used is an address of the interface which is taken as a source of nve 1 through the configuration below, which is **10.0.0.1**.

```

interface nve 1
no sh
source loopback 0

```

Dst Mac	Src Mac	Src Ip	Dst Ip	Protocol	UDP Hdr	VxLAN Hdr	ARP-Msg
01:00:5e:01:01:01	Mac(L1-eth1/1)	10.0.0.1	239.1.1.1	17 ( UDP)	dst port = 4789 src port = <random>	5010	<pkt 1>

### Packet P2


Since the above pkt P2 is a multicast Packet, L1 forwards it to R1 using matching mcast route **10.0.0.1/32, 239.1.1.1**. We can verify this using **CLI show ip mroute** on L1. In this case, L1 is acting as Mcast First Hop Router. Dst Mac **01:00:5e:01:01:01** is obviously mcast mac address which is derived from Group address. Below is the Wireshark capture of P2. Ping region shows the encapsulated ARP Broadcast Request packet

which is pkt p1.

```

> Frame 426: 110 bytes on wire (880 bits), 110 bytes captured (880 bits) on interface -, id 0
▼ Ethernet II, Src: 0c:a5:00:00:1b:08 (0c:a5:00:00:1b:08), Dst: IPv4mcast_01:01:01 (01:00:5e:01:01:01)
  > Destination: IPv4mcast_01:01:01 (01:00:5e:01:01:01)
  > Source: 0c:a5:00:00:1b:08 (0c:a5:00:00:1b:08)
    Type: IPv4 (0x0800)
▼ Internet Protocol Version 4, Src: 10.0.0.1, Dst: 239.1.1.1
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 96
    Identification: 0xd113 (53523)
  > 000. .... = Flags: 0x0
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 254
    Protocol: UDP (17)
    Header Checksum: 0xf175 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.0.0.1
    Destination Address: 239.1.1.1
▼ User Datagram Protocol, Src Port: 54517, Dst Port: 4789
  Source Port: 54517
  Destination Port: 4789
  Length: 76
  > Checksum: 0x0000 [zero-value ignored]
    [Stream index: 2]
  > [Timestamps]
    UDP payload (68 bytes)
▼ Virtual eXtensible Local Area Network
  > Flags: 0x0800, VXLAN Network ID (VNI)
    Group Policy ID: 0
    VXLAN Network Identifier (VNI): 5010
    Reserved: 0
▼ Ethernet II, Src: 0c:08:4e:22:00:01 (0c:08:4e:22:00:01), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
  > Destination: Broadcast (ff:ff:ff:ff:ff:ff)
  > Source: 0c:08:4e:22:00:01 (0c:08:4e:22:00:01)
    Type: ARP (0x0806)
    Trailer: 00000000000000000000000000000000
  Encapsulated ARP Broadcast Pkt P1
▼ Address Resolution Protocol (request)
  Hardware type: Ethernet (1)

```

 *Random Number  
used for ECMP load  
balancing*

```

Protocol type: IPv4 (0x0800)
Hardware size: 6
Protocol size: 4
Opcode: request (1)
Sender MAC address: 0c:08:4e:22:00:01 (0c:08:4e:22:00:01)
Sender IP address: 192.168.10.10
Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
Target IP address: 192.168.10.11

```

### intra-vlan-routinig-pkt-p2

When spine R1 receives the Mcast Pkt P2 , it forwards the Multicast packet P2 to Leaf L4 using multicast route entry ( 10.0.0.1, 239.1.1.1 ) . The pkt content is unchanged except the Src mac address of the pkt is updated to MAc (R1-eth1/4) which is not worth discussion.

Finally, L4 receives Packet P2. P2 is a multicast packet. So, P2 is processed by Multicast Stack. L4 can process this paket only and only if it has a Multicast route in MFIB. Matching Multicast Route is **S, G, IIF = (\*, 239.1.1.1, eth1/1 )**. This route route was created through configuration.

```
interface nve 1
```

```
no sh
```

```
source loopback 0
```

```
member vni 5010 mcast-group 239.1.1.1 <<<< This config is saying that leaf is Mcast LHR for Group 239.1.1.1 and OIF is nve1
```

```
member vni 5020 mcast-group 239.2.2.2 <<<< This config is saying that leaf is Mcast LHR for Group 239.2.2.2 and OIF is nve1
```

But what L4 will do with this packet and how?

If you notice in the above step, the OIF of the multicast route is **nve1** interface. The nve1 interface when used as OIF of mcast route is set with **the DECAP** flag. The mcast route shall push the Packet P2 to OIF nve1. When the OIF interface receives the multicast packet, it *decapsulates the packet rather than traditional mcast forwarding behavior*. Hence. nve1 decap the packet (remove Outermost IP Hdr) and detects that the next header followed is the UDP header which has a destination port = 4789, a special value which implies that next following header is VxLAN

header, hence, process the pkt next as per VxLAN module logic.

The reception of VxLAN headered packet on NVE interface nve1 on L4 switch helps L4 switch to learn about NVE peer 10.0.0.1.

```
leaf4# show nve peers detail
```

Details of nve Peers:

-----

Peer-Ip: 10.0.0.1

NVE Interface : nve1

Peer State : Up

Peer Uptime : 00:06:04

Router-Mac : n/a

Peer First VNI : 5010

Time since Create : 00:06:05

Configured VNIs : 5010,5020

Provision State : peer-add-complete

Learnt CP VNIs : 

vni assignment mode : SYMMETRIC

Peer Location : N/A

```
leaf1# show nve peers
```

```
leaf1#
```

Since L1 has not yet seen any kind of packet from L4, therefore, L1 do not yet know about the existence of 10.0.0.4 ( L4 ) as an NVE peer.

Out IP Hdr, UDP Hdr, and VxLAN hdr of the pkt P2 are removed, and Inner Packet is obtained which was nothing but pkt p1 by L4.

Using vlan--vni mapping table, L4 maps VNI 5010 encoded in VxLAN hdr back to vlan 10 ( reverse mapping ). It tags the ethernet of the packet P1 with 802.1Q VLAN header with VLAN-id = 10.

Following unprocessed bytes in the packet is the ethernet hdr of the packet P1. L4 processes the Ethernet hdr of the Packet P1. Be noted, that ethernet Hdr is processed by Data Link Layer. In other words, the Mac address table is used to process the packet based on ethernet header information. Now, Using the Dst Mac address of the packet P1 which is **ff:ff:ff:ff:ff:ff**, L4 floods the packet in out of all ports that are in VLAN 10.

In addition, Mac Learning is also performed. The following entry is inserted into the Mac address table of Leaf L4

```
leaf4# show mac address-table
```

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, ~ - vsan

VLAN MAC Address Type age Secure NTFY Ports

```
-----+-----+-----+-----+-----+-----+-----
```

```
* 10 0c08.4e22.0001 dynamic 0 F F nve1(10.0.0.1) <<<< Entry learned through Mac Learning using ARP-B pkt P1
```

```
* 10 0c8a.0bb7.0001 dynamic 0 F F Eth1/3 <<<< Learned through Mac Learning but by ARP-Reply pkt, Explained further
```

```
G - 0c48.0000.1b08 static - F F sup-eth1(R)
```

```
leaf4#
```

Packet P1 is received by Host H6 now as it is the only host in VLAN 10. Note that, when the packet was pushed out of interface eth1/3 by L4, it got

untagged since eth1/3 is an access port.

H6 receives the untagged ARP-Broadcast Packet P1. H6 is not even aware the packet has traversed the Spine or contains any traces that the packet has traveled the VxLAN network. H6 views H1 as if H1 is present in its Local VLAN 10.

H6 processes ARP-Broadcast packet, does ARP learning and is willing to send ARP reply back.

*This completes the packet journey of the ARP-Broadcast packet from Host H1 to Host H6 through VxLAN fabric. Note that, this ARP-Broadcast request helps Leafs to learn about hosts (local and remote) as it created mac address table entry in mac address table of Leaf L1 and Leaf L4. Leafs also get to know about their NVE peers (show nve peers) due to exchange of traffic.*

---

## ARP Reply

Host H6 Receives the ARP broadcast packet, it populate its ARP cache with the mac address of host H1.

H6 replies with ARP reply msg

### ARP reply Gen by H6

```
> Frame 9: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface -, id 0
✓ Ethernet II, Src: 0c:8a:0b:b7:00:01 (0c:8a:0b:b7:00:01), Dst: 0c:08:4e:22:00:01 (0c:08:4e:22:00:01)
  > Destination: 0c:08:4e:22:00:01 (0c:08:4e:22:00:01)
  > Source: 0c:8a:0b:b7:00:01 (0c:8a:0b:b7:00:01)
    Type: ARP (0x0806)
    Padding: 00000000000000000000000000000000
  ✓ Address Resolution Protocol (reply)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: reply (2)
    Sender MAC address: 0c:8a:0b:b7:00:01 (0c:8a:0b:b7:00:01)
    Sender IP address: 192.168.10.11
    Target MAC address: 0c:08:4e:22:00:01 (0c:08:4e:22:00:01)
    Target IP address: 192.168.10.10
```

This ARP reply msg when reaches the L4 VTEP on interface eth1/3, gets tagged with vlan 802.1q hdr vlan id = 10.

L4 data link layer handles this pkt now. Look Up the mac address table with key: vlan = 10, mac = <dst mac = 0c:08:4e:22:00:01 > in the packet

```
leaf4# show mac address-table
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, ~ - vsan
VLAN MAC Address Type age Secure NTFY Ports
-----+-----+-----+-----+-----+-----+-----
* 10 0c08.4e22.0001 dynamic 0 F F nve1(10.0.0.1) << matching entry
* 10 0c8a.0bb7.0001 dynamic 0 F F Eth1/3
G - 0c48.0000.1b08 static - F F sup-eth1(R)
leaf4#
```

The lookup result was the entry which was populated by ARP-Broadcast request. This entry dictates L4 to send the packet out of interface nve1 to VTEP 10.0.0.1. VNI to be used it 5010 ( converted from vlan 10 using vni-to-vlan mapping table ).

**The Encapsulated ARP reply pkt generated by L4**



```

> Frame 10: 110 bytes on wire (880 bits), 110 bytes captured (880 bits) on interface -, id 0
▼ Ethernet II, Src: 0c:48:00:00:1b:08 (0c:48:00:00:1b:08), Dst: 0c:a3:00:00:1b:08 (0c:a3:00:00:1b:08)
  > Destination: 0c:a3:00:00:1b:08 (0c:a3:00:00:1b:08)
  > Source: 0c:48:00:00:1b:08 (0c:48:00:00:1b:08)
  Type: IPv4 (0x0800)
▼ Internet Protocol Version 4, Src: 10.0.0.4, Dst: 10.0.0.1
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 96
  Identification: 0x0000 (0)
  > 000. .... = Flags: 0x0
  ...0 0000 0000 0000 = Fragment Offset: 0
  Time to Live: 254
  Protocol: UDP (17)
  Header Checksum: 0xa888 [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 10.0.0.4
  Destination Address: 10.0.0.1
▼ User Datagram Protocol, Src Port: 54428, Dst Port: 4789
  Source Port: 54428
  Destination Port: 4789
  Length: 76
  > Checksum: 0x0000 [zero-value ignored]
  [Stream index: 3]
  > [Timestamps]
  UDP payload (68 bytes)
▼ Virtual eXtensible Local Area Network
  > Flags: 0x0800, VXLAN Network ID (VNI)
  Group Policy ID: 0
  VXLAN Network Identifier (VNI): 5010
  Reserved: 0
▼ Ethernet II, Src: 0c:8a:0b:b7:00:01 (0c:8a:0b:b7:00:01), Dst: 0c:08:4e:22:00:01 (0c:08:4e:22:00:01)
  > Destination: 0c:08:4e:22:00:01 (0c:08:4e:22:00:01)
  > Source: 0c:8a:0b:b7:00:01 (0c:8a:0b:b7:00:01)
  Type: ARP (0x0806)
  Trailer: 00000000000000000000000000000000
  Address Resolution Protocol (reply)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6

```

Encapsulated

ARP reply

```

Hardware size: 0
Protocol size: 4
Opcode: reply (2)
Sender MAC address: 0c:8a:0b:b7:00:01 (0c:8a:0b:b7:00:01)
Sender IP address: 192.168.10.11
Target MAC address: 0c:08:4e:22:00:01 (0c:08:4e:22:00:01)
Target IP address: 192.168.10.10

```

This ARP reply packet is unicast packet, as opposed to ARP-Broadcast packet which was multicast packet. The underlay unicast routing takes care to unicast this packet to its intended destination 10.0.0.1 from 10.0.0.4.

When the ARP reply arrives on L4 , the decapsulation procedure applies.

Layer 2 handling: The dst mac address in the outer most ethernet hdr of the packet matches with the mac address of physical interface eth1/1, pkt is promoted to Layer 3

The dst ip address in the outermost ip hdr of the pkt is the local address of L1, pkt is destined to self ( for L1 ), promoted to the transport layer as protocol value in IP hdr was 17 ( UDP )

UDP dst port is 4789, which means pkt is to be handled by VxLAN application module. and next hdr in the packet is VxLAN hdr

Now that, VTEP1 has received VxLAN encapsulated packet from remote machine whose IP address was 10.0.0.4, it learn the remote machine as its nve peer.

```
leaf1# show nve peers detail
```

```
Details of nve Peers:
```

```
-----
Peer-Ip: 10.0.0.4
```

```
NVE Interface : nve1
```

```
Peer State : Up
```

```
Peer Uptime : 00:52:37
```

```

Router-Mac : n/a
Peer First VNI : 5010
Time since Create : 00:52:37
Configured VNIs : 5010,5020
Provision State : peer-add-complete
Learnt CP VNIs :
vni assignment mode : SYMMETRIC
Peer Location : N/A

```

VNI is extracted from VxLAN hdr which is 5010 and is mapped back to vlan using vni-vlan mapping table

Following VxLAN hdr is always ethernet hdr , therefore, pkt is handled again by Data link layer

Mac address table is lookup using key: vni-to-vlan(vni(VxLAN Hdr)), dst mac which is 10, MAC(H1)

```
leaf1# show mac address-table
```

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, ~ - vsan

VLAN MAC Address Type age Secure NTFY Ports

```

-----+-----+-----+-----+-----+-----+-----
* 10 0c08.4e22.0001 dynamic 0 F F Eth1/3 << Matching entry

```

```
G - 0ca5.0000.1b08 static - F F sup-eth1(R)
leaf1#
```

The entry found dictates VTEP L1 to switch ARP reply out of local port eth1/3. Remember this mac entry was populated by ARP broadcast request.

In addition, as usual, L1 performs Mac learning and inserts a new entry in mac address table

```
leaf1# show mac address-table

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, ~ - vsan

VLAN MAC Address Type age Secure NTFY Ports
-----+-----+-----+-----+-----+-----+-----
* 10 0c08.4e22.0001 dynamic 0 F F Eth1/3
* 10 0c8a.0bb7.0001 dynamic 0 F F nve1(10.0.0.4) <<< Learning Remote MACs
G - 0ca5.0000.1b08 static - F F sup-eth1(R)
leaf1#
```

ARP reply is eventually processed by Host H1, it populates its ARP cache. This completes the journey of the ARP reply pkt from H6 to H1.

---

## ICMP PING

H1 has resolved the ARP successfully for 192.168.10.11, and now it can manufacture ICMP packets. The ICMP (request or reply) packets takes

exactly the same procedure as ARP-reply pkt tool from H6 to H1 ( since ARP-reply and ICMP both are unicast packets ). Here I explain the procedure how ICMP ping echo request will go from H1 to H6 and leave ICMP echo reply tracing back from H6 to H1 as homework for you.

User issue ping 192.168.10.11 on H1. H1 detects he is pinging somebody in its local subnet.

H1 looks up ARP cache for 192.168.10.11 and finds the MAC entry MAC(H6) ( assuming ARP has been resolved using above procedure )

H1 send out ICMP echo request towards its default gateway

L1 receives ICMP echo request packet. The packet gets tagged with vlan id = 10.

Mac Learning - Entry is refreshed in mac table based on vlan, src mac in the packet as a key.

Mac forwarding - using vlan and destination mac as a key, MAC table lookup is performed.

```
leaf1# show mac address-table
```

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, ~ - vsan

VLAN MAC Address Type age Secure NTFY Ports

```
-----+-----+-----+-----+-----+-----+-----
```

```
* 10 0c08.4e22.0001 dynamic 0 F F Eth1/3 << refreshed
```

```
* 10 0c8a.0bb7.0001 dynamic 0 F F nve1(10.0.0.4) << used for mac forwarding
```

```
G - 0ca5.0000.1b08 static - F F sup-eth1(R)
```

```
leaf1#
```

Packet is pushed out of interface nve1 logically. nve1→encapsulate\_and\_send (10.0.0.4 , 5010) the procedure is executed. Below is the

Sireshawk snapshot of the packet. Pink region represents the innermost ICMP payload. You can see the stack of headers are as expected.

**Packet when pushed out of L1 interface eth1/1**



```

> Frame 38: 164 bytes on wire (1312 bits), 164 bytes captured (1312 bits) on interface -, id 0
▼ Ethernet II, Src: 0c:a3:00:00:1b:08 (0c:a3:00:00:1b:08), Dst: 0c:48:00:00:1b:08 (0c:48:00:00:1b:08)
  > Destination: 0c:48:00:00:1b:08 (0c:48:00:00:1b:08)
  > Source: 0c:a3:00:00:1b:08 (0c:a3:00:00:1b:08)
  Type: IPv4 (0x0800)
▼ Internet Protocol Version 4, Src: 10.0.0.1, Dst: 10.0.0.4
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 150
  Identification: 0x0000 (0)
  > 000. .... = Flags: 0x0
  ...0 0000 0000 0000 = Fragment Offset: 0
  Time to Live: 253
  Protocol: UDP (17)
  Header Checksum: 0xa952 [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 10.0.0.1
  Destination Address: 10.0.0.4
▼ User Datagram Protocol, Src Port: 50170, Dst Port: 4789
  Source Port: 50170
  Destination Port: 4789
  Length: 130
  > Checksum: 0x0000 [zero-value ignored]
  [Stream index: 2]
  > [Timestamps]
  UDP payload (122 bytes)
▼ Virtual eXtensible Local Area Network
  > Flags: 0x0800, VXLAN Network ID (VNI)
  Group Policy ID: 0
  VXLAN Network Identifier (VNI): 5010
  Reserved: 0
▼ Ethernet II, Src: 0c:08:4e:22:00:01 (0c:08:4e:22:00:01), Dst: 0c:8a:0b:b7:00:01 (0c:8a:0b:b7:00:01)
  > Destination: 0c:8a:0b:b7:00:01 (0c:8a:0b:b7:00:01)
  > Source: 0c:08:4e:22:00:01 (0c:08:4e:22:00:01)
  Type: IPv4 (0x0800)
▼ Internet Protocol Version 4, Src: 192.168.10.10, Dst: 192.168.10.11
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 100

```



```

-----
Identification: 0x3406 (13318)
> 000. .... = Flags: 0x0
...0 0000 0000 0000 = Fragment Offset: 0
Time to Live: 255
Protocol: ICMP (1)
Header Checksum: 0xf22c [validation disabled]
[Header checksum status: Unverified]
Source Address: 192.168.10.10
Destination Address: 192.168.10.11
> Internet Control Message Protocol

```

ICMP packet is received by L4. The decapsulation procedure applies.

Layer 2 handling : The dst mac address in the outer most ethernet hdr of the packet matches with the mac address of physical interface eth1/1, pkt is promoted to Layer 3

The dst ip address in the outermost ip hdr of the pkt is the local address of L1, pkt is destined to self ( for L1 ), promoted to Transport layer ( UDP ) as protocol value in IP hdr was 17 ( UDP )

UDP dst port is 4789, which means pkt is to be handled by VxLAN application module. and next hdr in packet is VxLAN hdr

VNI is extracted from VxLAN hdr which is 5010 and is mapped back to vlan using vni-vlan mapping table

Following VxLAN hdr is always ethernet hdr , therefore, pkt is handled again by Data link layer

Mac address table is lookup using using key : vni-to-vlan(vni(VxLAN Hdr))), dst mac which is 10, MAC(H6).

```
leaf4# show mac address-table
```

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, ~ - vsan

```

VLAN MAC Address Type age Secure NTFY Ports
-----+-----+-----+-----+-----+-----+-----
* 10 0c08.4e22.0001 dynamic 0 F F nve1(10.0.0.1) << refreshed
* 10 0c8a.0bb7.0001 dynamic 0 F F Eth1/3 << Matching entry
G - 0c48.0000.1b08 static - F F sup-eth1(R)
leaf4#

```

The entry found dictates VTEP L4 to switch the ICMP echo request packet out of local port eth1/3.

ICMP echo request is received by H6 and processed.

## NVE Interface

Now, Having studied and understood the basics of VxLAN Data Plane Operations involved, We can now discuss the NVE interfaces. The NVE interface is used to encapsulate the ethernet packet with VxLAN hdr as well as to decapsulate the packet arrived from VxLAN fabric on a VTEP.

NVE interface encapsulation algorithm is a function of two parameters - the remote VTEP end point IP Address and VNI id.

Below captures the pseudocode regarding how NVE interface encapsulate and forward the packet on sending VTEP as well as how NVE interface decapsulates and process the packet on Recipient VTEP.

Encapsulation	Decapsulation
<pre> nve-&gt;encapsulate_and_send (remote_vtep_ip , VNI) . Attach VxLAN Hdr with VNI value . Attach UDP hdr with src port = &lt;random no&gt; and dst port = 4789 . Attach IP Hdr . . src ip address = ip address of nve interface </pre>	<pre> nve-&gt;decapsulate_and_process (pkt) . assert ( dst_ip (pkt) == ( ip_addr(nve) or mcast group address config on nve); . src_vtep_ip = srp_ip (pkt); . Remove outer IP Hdr , next hdr = UDP ( because protocol field in outer IP hdr = 17 ) </pre>

```

. . dst ip address = remote_vtep_ip
. . protocol = 17
. Attach Ethernet Hdr
. . dst mac address = mac address of nexthop to remote_vtep_ip
. . src mac address = mac address of outgoing interface towards
remote_vtep_ip
. . type = 0x0800
. Route packet to remote_vtep_ip
Note : Above pseudocode attach headers to the packer from inward
towards outward.

```

```

. Remove UDP hdr , next hdr = VxLAN Hdr ( because dst port no in
UDP hdr = 4789 )
. assert (pkt_type (pkt) == ethernet_pkt );
. mac_learning (vni_to_vlan (vni (VxLAN Hdr)), src_mac(pkt), nve
(src_vtep_ip));
. l2_entry = lookup_mac_table (vlan_to_vni (vni (VLAN Hdr),
dst_mac(pkt));
. assert (l2_entry);
. if l2_entry OIF = local port
. . switch packet to local port
. else if l2_entry OIF = L3 vlan with IP routing capabilities
. . subject packet to L3 routing based on dst_ip (pkt) in a VRF context
Note : Above Pseudocode process the headers from outward towards
inwards

```

## Conclusion

We demonstrated how Hosts present across Vxlan fabric in the same vlan can communicate. While multicast takes responsibility for distributing BUM traffic efficiently across interested VTEPs, the routing of packet between hosts of the same vlan across VxLAN fabric involved only mac address table - Mac Learning and Forwarding. Hosts, despite being separated by L3 VxLAN network, logically view each other as if present in same vlan / network. But how hosts in different Vlans can communicate with each other - this we will cover in Inter-Vlan Routing Lab.

visit : <http://www.csepracticals.com> for more courses and offers.

**Abhishek Sagar**