

VXLAN (Virtual Extensible LAN) คืออะไร?

VXLAN เป็นเทคโนโลยีที่ใช้สำหรับสร้างเครือข่ายเสมือน (Overlay Network) บนเครือข่าย IP แบบเดิม โดย VXLAN ช่วยขยายจำนวน VLAN ที่สามารถใช้ได้จากเดิมที่จำกัดเพียง 4096 VLAN (802.1Q) ไปเป็น 16 ล้าน (2^{24}) ผ่านการใช้ VXLAN Network Identifier (VNI) ซึ่งมีความยาว 24 บิต

วิธีการทำงานของ VXLAN

VXLAN ใช้หลักการของการสร้าง Overlay Network โดยมีส่วนประกอบสำคัญคือ

1. **VXLAN Tunnel Endpoints (VTEP)** – อุปกรณ์ที่ทำหน้าที่แปลงและห่อหุ้ม (Encapsulate) เฟรม Ethernet ไปเป็นแพ็กเก็ต VXLAN ก่อนส่งข้ามเครือข่าย
 2. **VXLAN Encapsulation** – เฟรม Ethernet จะถูกห่อหุ้มด้วยหัว VXLAN และโปรโตคอล UDP ซึ่งถูกส่งผ่านเครือข่าย IP (Underlay Network)
 3. **VXLAN VNI (VXLAN Network Identifier)** – ใช้แทน VLAN ID เพื่อระบุเครือข่ายเสมือน
 4. **Multicast หรือ Unicast สำหรับ Flood & Learn** – ใช้สำหรับกระจายข้อมูลไปยังปลายทางที่เกี่ยวข้อง

ໜີ້ຕະລາງທຳງວາເຊລົງ XVI AN:

1. อุปกรณ์ต้นทางส่งเฟรม Ethernet ไปยัง VTEP
 2. VTEP ห่อหุ้ม (Encapsulate) เฟรมดังกล่าวด้วยส่วนหัว VXLAN และส่งไปยังปลายทางผ่านเครือข่าย IP
 3. VTEP ปลายทางถอด (Decapsulate) หัว VXLAN และส่งเฟรม Ethernet ไปยังปลายทาง

ข้อดีของ VXLAN

- ✓ รองรับจำนวนเครือข่ายมากขึ้น – ขยายขีดจำกัดจาก 4096 VLAN ไปถึง 16 ล้าน VXLAN
 - ✓ สามารถทำงานข้ามศูนย์ข้อมูล (Data Center Interconnect - DCI) – VXLAN ใช้ IP เป็น Underlay ทำให้สามารถขยายเครือข่ายไปยังหลายศูนย์ข้อมูล
 - ✓ รองรับการทำงานแบบ Multi-Tenancy – รองรับการแยกเครือข่ายของผู้ใช้หลายกลุ่ม
 - ✓ ทำงานบนโครงสร้าง IP เดิม – ใช้ IP ในการส่งข้อมูล จึงสามารถใช้กับเครือข่ายปัจจุบันได้โดยไม่ต้องเปลี่ยนโครงสร้างมากข้อเสียของ VXLAN

 เพิ่ม Overhead – VXLAN ห่อหุ้มแพ็กเก็ต Ethernet ด้วยหัว VXLAN และ UDP ทำให้มีขนาดแพ็กเก็ตใหญ่ขึ้น

 ต้องการการตั้งค่าที่ซับซ้อน – ต้องกำหนดค่า VTEP และจัดการการเรียนรู้เส้นทาง (Flood & Learn)

 อุปกรณ์ต้องรองรับ VXLAN – อุปกรณ์เครือข่ายบางรุ่นอาจไม่รองรับ VXLAN หรือรองรับเฉพาะในระดับซอฟต์แวร์

สรุป

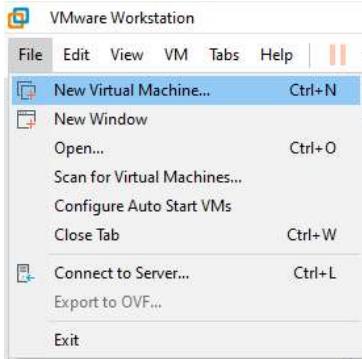
VXLAN เป็นเทคโนโลยีที่ช่วยขยายขีดจำกัดของ VLAN และทำให้สามารถสร้างเครือข่ายเสมือนในศูนย์ข้อมูลและบนเครือข่าย IP ได้อย่างยืดหยุ่น โดยมีข้อดีเรื่องการรองรับจำนวนเครือข่ายมากขึ้นและทำงานบนโครงสร้าง IP เดิม แต่ก็มีข้อเสียด้าน Overhead และความซับซ้อนในการตั้งค่า

Vxlan

โปรแกรมที่ต้องติดตั้ง
Hypervisor ในที่นี้มีเลือกเป็น Vmware Workstation
MobaXterm ใช้สำหรับ SSH และ FTP ไฟล์
Eve-ng.iso > <https://www.eve-ng.net/index.php/download/#DL-COMM>
Image eve ที่ใช้สร้าง Lab นี้ Cisco Nexus 9000v
https://mega.nz/file/8d11kaAI#_Ovf_1yl4cxCPfwyQ_J41bB8pedRPZL3l8K0mxSExA

ทำการตั้งค่า โปรแกรม Vmware Workstation

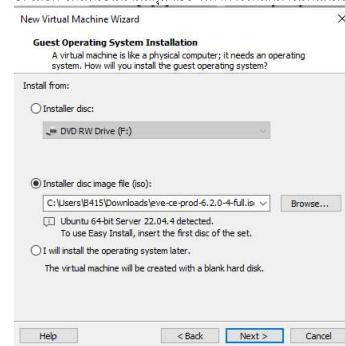
1. New Virtual Machine หรือ กด Ctrl+N



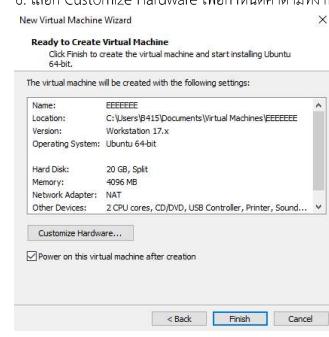
2. เลือก Typical จากนั้นกด Next



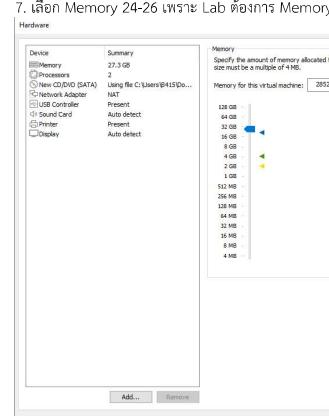
3. เลือกไฟล์ที่เป็นนามสกุล .iso ที่ทำการโหลดมาในที่นี้แล้วให้ eve-ce-prod-6.2.0-4-full.iso จากนั้นกด Next



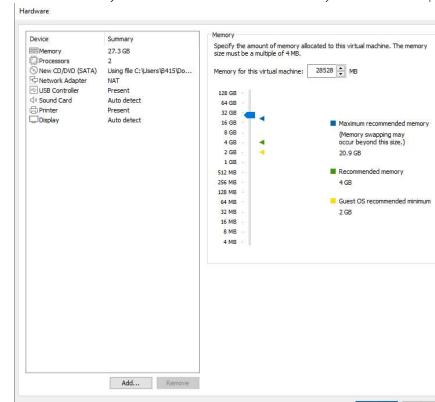
4. ตั้งชื่อ VM ของรากจากนั้นกด Next



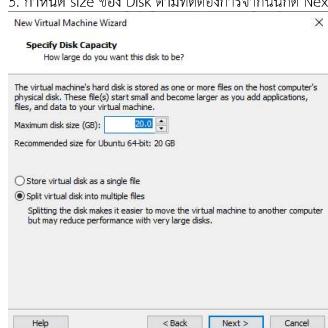
6. เลือก Customize Hardware เพื่อกำหนດค่าตามที่เราต้องการ



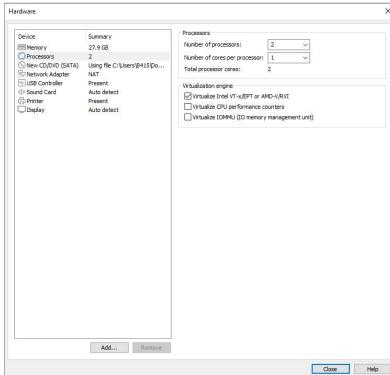
7. เลือก Memory 24-26 เพราะ Lab ต้องการ Memory ที่เยอะแบบนี้ๆ



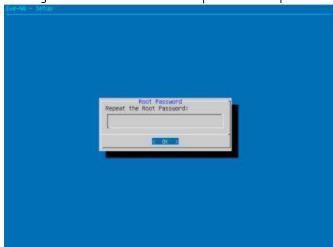
5. กำหนด size ของ Disk ตามที่ต้องการจากนั้นกด Next



8. เลือก Processors จากนั้นเลือก Virtualize Intel VT-x/EPT or AMD-V/RVI สุดท้ายกด Close



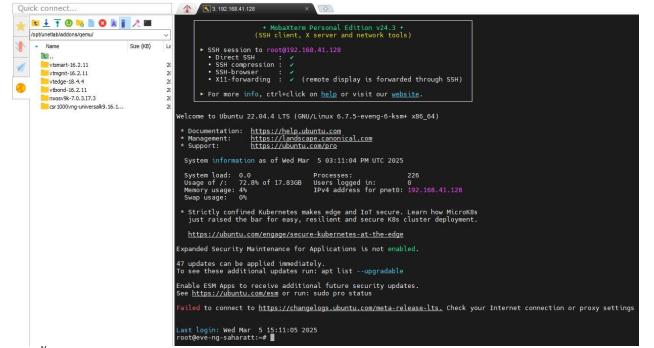
เมื่อกำหนนค่าเครื่องหมายแล้วปุ่มกด Eve ให้
Eve-ng จะให้ทำการกำหนดค่าต่างๆ โดยจะให้ใส่ password โดยจะมาใส่เป็น eve เพื่อง่ายต่อการใช้



เมื่อติดตั้งเสร็จจะขึ้นหน้าแบบนี้ และจะให้ ip เพื่อเข้าสู่ eve ผ่านเว็บเบราว์เซอร์มาอย่างเดียวคือ 192.168.41.128

```
System information as of Wed Mar 5 03:11:04 PM UTC 2025
System load: 0.0 Processes: 226
Usage of /: 72.8% of 17.83GB Users logged in: 0
Memory usage: 4% IPv4 address for pnet0: 192.168.41.128
Swap usage: 0%
```

ทำการอัพโหลดไฟล์ image ของอุปกรณ์ที่ path /opt/unetlab/addons/qemu/
จากนั้น ใช้คำสั่ง /opt/unetlab/wrappers/unl_wrapper -a fixpermissions เพื่อให้สามารถใช้ image ของ
อุปกรณ์ได้



จากนั้นเปิด browser ใส่ ip ของ eve-ng (192.168.41.128) ในหน้า browser มี username: admin และ
password: eve (ห้ามหักดิบ)

Unicast Configuration

1. เป้าหมายของการตั้งค่า (Objective)

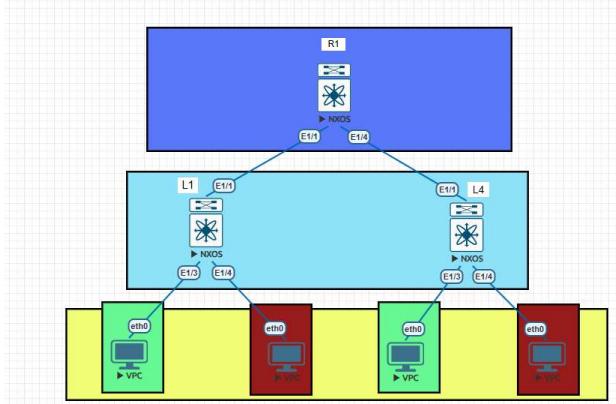
- เป้าหมายหลักของการตั้งค่า Unicast Routing คือ ทำให้ทุกอุปกรณ์ในเครือข่าย Spine-Leaf ติดต่อกัน ได้ผ่าน IP (IP Reachability)
- เครือข่าย Spine-Leaf เป็นสถาปัตยกรรมเครือข่ายที่มีการเชื่อมต่อ Spine Switches กับ Leaf Switches
- การตั้งค่า Unicast Routing เป็นชั้นฐานที่สำคัญก่อนนำไปใช้งานเทคโนโลยีอื่น ๆ เช่น VXLAN Overlay

2. กำหนดค่า IP บนอินเทอร์เฟซ (Configure IP address on all interfaces connecting Spines and Leafs)

- ทุกอุปกรณ์ที่ต้องต่อระหว่าง Spine และ Leaf ต้องมี IP Address
- โดยปกติจะใช้ /30 Subnet สำหรับการเชื่อมต่อแบบ Point-to-Point



ทำการสร้าง Topology



- ตัวอย่าง interface ethernet 1/1


```
description to-spine1
ip address 10.1.1.1/30
```
- การกำหนดค่า IP นี้ช่วยให้สามารถตั้งค่า Routing Protocol เช่น OSPF หรือ ISIS เพื่อแลกเปลี่ยนข้อมูล
ผ่านทาง

3. ตั้งค่า Loopback Address บนอุปกรณ์ทั้งหมด (Configure Loopback Address 0 on all nodes with /32 loopback address)

- ทุกอุปกรณ์จะมี Loopback 0 Address ซึ่งเป็น /32 IP Address
- Loopback Address จะเป็น Router-ID ใน Routing Protocol เช่น OSPF หรือ ISIS
- ตัวอย่างการตั้งค่า Loopback


```
interface loopback 0
ip address 10.0.0.1/32
```
- Loopback นี้ต้องไม่ซ้ำกับสถานะของ Physical Interface ที่ให้เป็นจุดอ้างอิงที่เสถียร

4. ติดตั้ง Routing Protocol (Deploy Routing Protocol OSPF/ISIS)

- ต้องใช้ Routing Protocol เช่น OSPF หรือ ISIS เพื่อทำให้ทุกๆปุ่มนั้น รู้จักเส้นทางของกันและกัน เมื่อใช้ OSPF
 - router ospf Si-Underlay
 - router-id 10.1.0.1
 - network 10.1.0.1/32 area 0
- OSPF ซ่อนการจ่ายเส้นทาง (Route Advertisement) เพื่อให้ Spine และ Leaf สามารถแลกเปลี่ยนข้อมูลเครือข่ายได้

5. Config

Spine R1	Leaf L1	Leaf L4
<pre>conf hostname spine1 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 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 sh ip address 10.1.2.2/30 ip router ospf Si-Underlay area 0 ip ospf network point-to-point mtu 9000 interface ethernet 1/3 description to-leaf3-eth1/1 no sh 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 sh 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 copy run start</pre>	<pre>conf hostname leaf1 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 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 sh no sw ip address 10.1.2.1/30 ip router ospf Si-Underlay area 0 ip ospf network point-to-point mtu 9000 interface ethernet 1/3 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 no sh end copy run start</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 ip ospf network point-to-point end conf interface ethernet 1/1 description to-spine1-eth1/3 no sh no sw ip address 10.2.4.1/30 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 sh 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 copy run start</pre>

6. Routing Table

R1

```
Spine1# show ip route
IP Route Table for VRF "default"
* denotes best cost next-hop
*** denotes best most cost next-hop
[<x/y>] denotes [preference/metric]
*<string> in via output denotes VRF <string>
10.0.0.1/32, ubest/mbest: 1/0
  *via 10.1.4.4, Eth1/4, [110/41], 17:55:37, ospf-SI-Underlay, intra
10.0.0.4/32, ubest/mbest: 1/0, attached
  *via 10.1.4.4, Eth1/4, [110/41], 17:55:19, ospf-SI-Underlay, intra
10.1.0.1/32, ubest/mbest: 2/0, attached
  *via 10.1.0.1, Eth1/1, [0/0], 17:57:24, local
  *via 10.1.0.1, Lco, [0/0], 17:57:24, direct
10.1.1.0/30, ubest/mbest: 1/0, attached
  *via 10.1.1.2, Eth1/1, [0/0], 17:57:18, direct
10.1.2.0/30, ubest/mbest: 1/0, attached
  *via 10.1.2.2, Eth1/1, [0/0], 17:57:18, local
10.1.2.2/32, ubest/mbest: 1/0, attached
  *via 10.1.2.2, Eth1/2, [0/0], 17:57:11, direct
10.1.3.0/30, ubest/mbest: 1/0, attached
  *via 10.1.3.2, Eth1/3, [0/0], 17:57:06, direct
10.1.3.2/32, ubest/mbest: 1/0, attached
  *via 10.1.3.2, Eth1/3, [0/0], 17:57:06, local
10.1.4.0/30, ubest/mbest: 1/0, attached
  *via 10.1.4.2, Eth1/4, [0/0], 17:56:17, direct
10.1.4.2/32, ubest/mbest: 1/0, attached
  *via 10.1.4.2, Eth1/4, [110/80], 17:55:57, ospf-SI-Underlay, intra
10.2.4.0/30, ubest/mbest: 1/0
  *via 10.3.4.1, Eth1/4, [110/80], 17:55:19, ospf-SI-Underlay, intra
```

L1

```
leaf1# show ip route
IP Route Table for VRF "default"
* denotes best cost next-hop
*** denotes best most cost next-hop
[<x/y>] denotes [preference/metric]
*<string> in via output denotes VRF <string>
```

```
10.0.0.1/32, ubest/mbest: 2/0, attached
  *via 10.0.0.1, Lco, [0/0], 17:56:16, local
  *via 10.0.0.1, Lco, [0/0], 17:56:16, direct
10.0.0.4/32, ubest/mbest: 1/0
  *via 10.1.1.2, Eth1/1, [110/81], 17:55:38, ospf-SI-Underlay, intra
```

```
10.1.0.1/32, ubest/mbest: 1/0, attached
  *via 10.1.1.1, Eth1/1, [0/0], 17:56:15, direct
10.1.1.0/30, ubest/mbest: 1/0, attached
  *via 10.1.1.1, Eth1/1, [0/0], 17:56:15, local
10.1.2.0/30, ubest/mbest: 1/0
  *via 10.1.1.2, Eth1/1, [110/80], 17:55:57, ospf-SI-Underlay, intra
10.1.3.0/30, ubest/mbest: 1/0
  *via 10.1.1.2, Eth1/1, [110/80], 17:55:57, ospf-SI-Underlay, intra
```

```
10.1.4.0/30, ubest/mbest: 1/0
  *via 10.1.1.2, Eth1/1, [110/80], 17:55:57, ospf-SI-Underlay, intra
10.2.1.0/30, ubest/mbest: 1/0, attached
  *via 10.2.1.1, Eth1/2, [0/0], 17:56:14, direct
10.2.1.1/32, ubest/mbest: 1/0, attached
  *via 10.2.1.1, Eth1/2, [0/0], 17:56:14, local
10.2.4.0/30, ubest/mbest: 1/0
  *via 10.1.1.2, Eth1/1, [110/120], 17:55:38, ospf-SI-Underlay, intra
```

```
L4
```

```
leaf4# show ip route
IP Route Table for VRF "default"
* denotes best cost next-hop
*** denotes best most cost next-hop
[<x/y>] denotes [preference/metric]
*<string> in via output denotes VRF <string>
```

```
10.0.0.1/32, ubest/mbest: 2/0, attached
  *via 10.0.0.1, Lco, [0/0], 17:56:16, local
  *via 10.0.0.1, Lco, [0/0], 17:56:16, direct
10.0.0.4/32, ubest/mbest: 1/0
  *via 10.1.1.2, Eth1/1, [110/81], 17:55:38, ospf-SI-Underlay, intra
```

```
10.1.0.1/32, ubest/mbest: 1/0, attached
  *via 10.1.1.1, Eth1/1, [0/0], 17:56:15, direct
10.1.1.0/30, ubest/mbest: 1/0, attached
  *via 10.1.1.1, Eth1/1, [0/0], 17:56:15, local
10.1.2.0/30, ubest/mbest: 1/0
  *via 10.1.1.2, Eth1/1, [110/80], 17:55:57, ospf-SI-Underlay, intra
10.1.3.0/30, ubest/mbest: 1/0
  *via 10.1.1.2, Eth1/1, [110/80], 17:55:57, ospf-SI-Underlay, intra
```

```
10.1.4.0/30, ubest/mbest: 1/0
  *via 10.1.1.2, Eth1/1, [110/80], 17:55:57, ospf-SI-Underlay, intra
10.2.1.0/30, ubest/mbest: 1/0, attached
  *via 10.2.1.1, Eth1/2, [0/0], 17:56:14, direct
10.2.1.1/32, ubest/mbest: 1/0, attached
  *via 10.2.1.1, Eth1/2, [0/0], 17:56:14, local
10.2.4.0/30, ubest/mbest: 1/0
  *via 10.1.1.2, Eth1/1, [110/120], 17:55:38, ospf-SI-Underlay, intra
```

```
l4# 
l4# show ip route
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>
10.0.0.1/32, ubest/mbest: 1/0
  *via 10.0.0.1, Eth1/1, [110/81], 17:56:23, ospf-SI-Underlay, intra
10.0.0.4/32, ubest/mbest: 1/0, attached
  *via 10.0.0.4, L00, [0/0], 17:56:43, local
  *via 10.0.0.4, L00, [0/0], 17:56:43, direct
10.1.0.1/32, ubest/mbest: 1/0
  *via 10.1.1.4.2, Eth1/1, [110/41], 17:56:23, ospf-SI-Underlay, intra
10.1.1.0/30, ubest/mbest: 1/0
  *via 10.1.1.4.2, Eth1/1, [110/80], 17:56:23, ospf-SI-Underlay, intra
10.1.4.0/30, ubest/mbest: 1/0, attached
  *via 10.1.4.1, Eth1/1, [0/0], 17:56:42, direct
10.1.4.1/32, ubest/mbest: 1/0, attached
  *via 10.1.4.1, Eth1/1, [0/0], 17:56:42, local
10.2.1.0/30, ubest/mbest: 1/0
  *via 10.1.4.2, Eth1/1, [110/120], 17:56:23, ospf-SI-Underlay, intra
10.2.4.0/30, ubest/mbest: 1/0, attached
  *via 10.2.4.1, Eth1/2, [0/0], 17:56:40, direct
10.2.4.1/32, ubest/mbest: 1/0, attached
  *via 10.2.4.1, Eth1/2, [0/0], 17:56:40, local
```

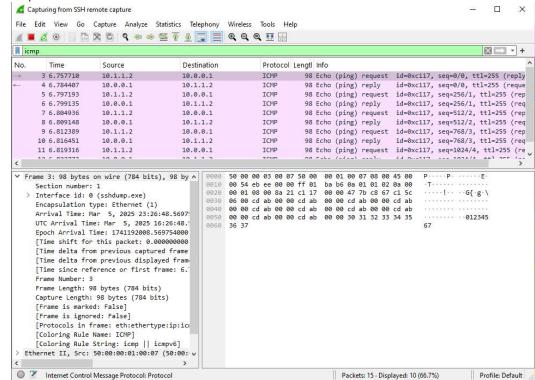
ทดสอบการ Ping Loopback

R1 to L1

```
spine# ping 10.0.0.1
PING 10.0.0.1 (10.0.0.1): 56 data bytes
64 bytes from 10.0.0.1: icmp_seq=0 ttl=254 time=6.603 ms
64 bytes from 10.0.0.1: icmp_seq=1 ttl=254 time=6.598 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=254 time=5.587 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=254 time=5.105 ms
64 bytes from 10.0.0.1: icmp_seq=4 ttl=254 time=4.286 ms

--- 10.0.0.1 ping statistics ---
5 packets transmitted, 5 packets received, 0.00% packet loss
round-trip min/avg/max = 4.286/5.635/6.603 ms
```

Capture



อธิบายตัวอย่างการ Capture

Frame

98 bytes on wire (784 bits), 98 bytes captured (784 bits)

→ พื้นที่เก็บข้อมูล 98 ไบต์ และถูกจับมาทั้งหมด

Interface: sshdump.exe, id 0

→ กำลังจับแพ็กเก็ตจากอินเทอร์เฟซ sshdump.exe

Ethernet II (เลเยอร์ 2 - Data Link Layer)

Src (MAC Source): 50:00:00:00:01:00:07

→ อุปกรณ์ที่ส่งแพ็กเก็ต

Dst (MAC Destination): 50:00:00:03:00:07

→ อุปกรณ์ปลายทางที่รับแพ็กเก็ต

IPv4 (เลเยอร์ 3 - Network Layer)

Src (IP Source): 10.1.1.2

→ IP ของเครื่องที่ส่งแพ็กเก็ต

Dst (IP Destination): 10.0.0.1

→ IP ของเครื่องปลายทาง

ICMP (เลเยอร์ 4 - Transport Layer - Internet Control Message Protocol)

→ โปรดติดต่อเพื่อขอความคุณสูง เช่น Ping (Echo Request/Reply)

วิเคราะห์แพ็กเก็ต

1. แพ็กเก็ตที่ส่ง ICMP (Internet Control Message Protocol) ซึ่งปกติจะใช้สำหรับ Ping

2. และตรวจสอบว่าอุปกรณ์ 10.1.1.2 กำลัง Ping ไปที่ 10.0.0.1

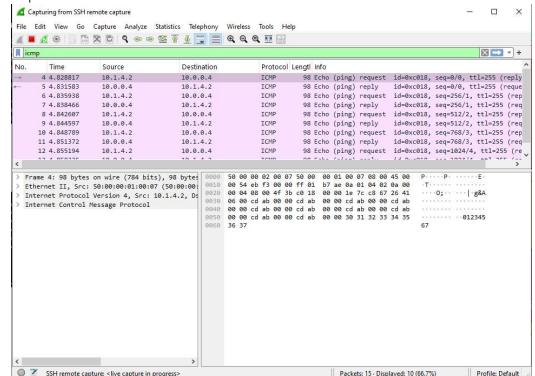
3. MAC Address ที่เห็นเป็นของอุปกรณ์ที่ใช้ VxLAN

R1 to L4

```
spine# ping 10.0.0.4
PING 10.0.0.4 (10.0.0.4): 56 data bytes
64 bytes from 10.0.0.4: icmp_seq=0 ttl=254 time=6.135 ms
64 bytes from 10.0.0.4: icmp_seq=1 ttl=254 time=6.81 ms
64 bytes from 10.0.0.4: icmp_seq=2 ttl=254 time=5.884 ms
64 bytes from 10.0.0.4: icmp_seq=3 ttl=254 time=8.83 ms
64 bytes from 10.0.0.4: icmp_seq=4 ttl=254 time=4.615 ms

--- 10.0.0.4 ping statistics ---
5 packets transmitted, 5 packets received, 0.00% packet loss
round-trip min/avg/max = 4.615/6.454/8.83 ms
```

Capture

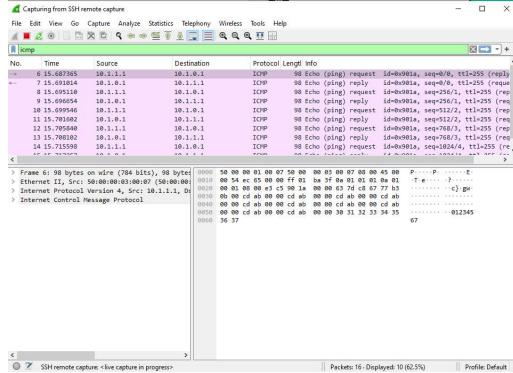


L1 to R1

```
leaf1# ping 10.1.0.1
PING 10.1.0.1 (10.1.0.1): 56 data bytes
64 bytes from 10.1.0.1: icmp_seq=0 ttl=254 time=8.261 ms
64 bytes from 10.1.0.1: icmp_seq=1 ttl=254 time=5.615 ms
64 bytes from 10.1.0.1: icmp_seq=2 ttl=254 time=7 ms
64 bytes from 10.1.0.1: icmp_seq=3 ttl=254 time=5.082 ms
64 bytes from 10.1.0.1: icmp_seq=4 ttl=254 time=5.49 ms

--- 10.1.0.1 ping statistics ---
5 packets transmitted, 5 packets received, 0.00% packet loss
round-trip min/avg/max = 5.082/6.289/8.261 ms
```

Capture

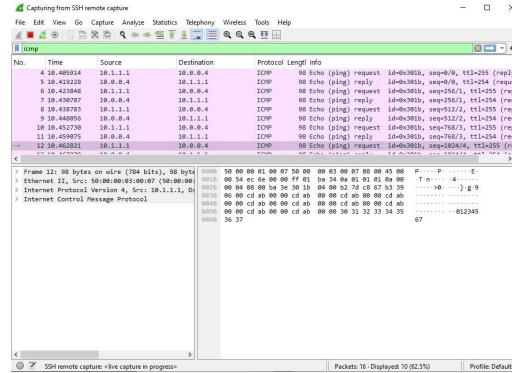


L1 to L4

```
leaf1# ping 10.0.0.4
PING 10.0.0.4 (10.0.0.4): 56 data bytes
64 bytes from 10.0.0.4: icmp_seq=0 ttl=253 time=19.51 ms
64 bytes from 10.0.0.4: icmp_seq=1 ttl=253 time=14.023 ms
64 bytes from 10.0.0.4: icmp_seq=2 ttl=253 time=10.517 ms
64 bytes from 10.0.0.4: icmp_seq=3 ttl=253 time=13.928 ms
64 bytes from 10.0.0.4: icmp_seq=4 ttl=253 time=15.985 ms

--- 10.0.0.4 ping statistics ---
5 packets transmitted, 5 packets received, 0.00% packet loss
round-trip min/avg/max = 10.517/14.792/19.51 ms
```

Capture

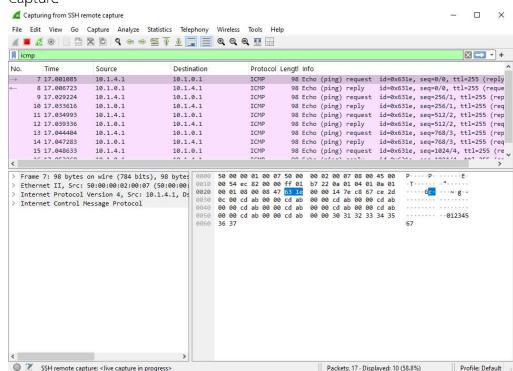


L4 to R1

```
leaf4# ping 10.1.0.1
PING 10.1.0.1 (10.1.0.1): 56 data bytes
64 bytes from 10.1.0.1: icmp_seq=0 ttl=254 time=5.884 ms
64 bytes from 10.1.0.1: icmp_seq=1 ttl=254 time=6.079 ms
64 bytes from 10.1.0.1: icmp_seq=2 ttl=254 time=3.331 ms
64 bytes from 10.1.0.1: icmp_seq=3 ttl=254 time=8.21 ms
64 bytes from 10.1.0.1: icmp_seq=4 ttl=254 time=3.856 ms

--- 10.1.0.1 ping statistics ---
5 packets transmitted, 5 packets received, 0.00% packet loss
round-trip min/avg/max = 3.331/5.472/8.21 ms
```

Capture

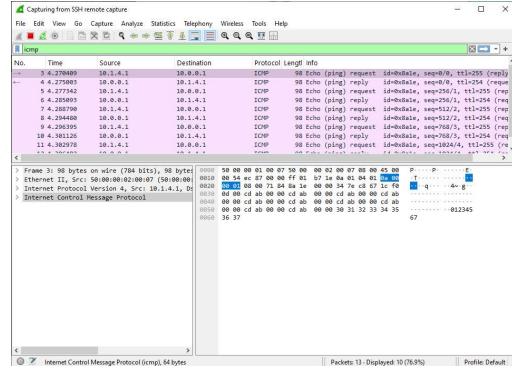


L4 to L1

```
leaf4# ping 10.0.0.1
PING 10.0.0.1 (10.0.0.1): 56 data bytes
64 bytes from 10.0.0.1: icmp_seq=0 ttl=253 time=14.837 ms
64 bytes from 10.0.0.1: icmp_seq=1 ttl=253 time=13.532 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=253 time=8.127 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=253 time=9.547 ms
64 bytes from 10.0.0.1: icmp_seq=4 ttl=253 time=8.8 ms

--- 10.0.0.1 ping statistics ---
5 packets transmitted, 5 packets received, 0.00% packet loss
round-trip min/avg/max = 8.127/10.968/14.837 ms
```

Capture



Multicast Configuration**1. เป้าหมายของงานตั้งค่า (Objective)**

การตั้งค่าที่คุณกล่าวมีวัตถุประสงค์หลักในการกระจายข้อมูล BUM traffic (Unknown, Broadcast, Unicast, Multicast) อย่างมีประสิทธิภาพในเครือข่าย multicast ซึ่งจะช่วยลดความหนาแน่นของสื่อสารข้อมูล และทำให้การกระจายข้อมูล multicast ในเครือข่ายได้ดีขึ้นเมื่อประสิทธิภาพมากขึ้น

2. Enable PIM Sparse mode บน Spine และ Leaf:

PIM (Protocol Independent Multicast) เป็นโปรโตคอลที่ใช้สำหรับการกระจายข้อมูล multicast โดยเป็นขั้นตอนไปโดยค่ากำหนดเดียว (routing protocol) ยืนยัน

Sparse Mode (PIM-SM) ให้มีความต้องการกระจายข้อมูล multicast ในบางพื้นที่หรือบางส่วนของเครือข่ายท่านั้น

การเปิดใช้ PIM Sparse Mode บน Spine และ Leaf อินเตอร์เฟสจะช่วยให้การกระจายข้อมูล multicast ทำงานได้ดีขึ้นเมื่อต้องการสื่อสารในเครือข่าย multicast ที่ไม่ใช่ทั่วไป

Rendezvous Point (RP) ที่กำหนด

3. Configure Spine เป็น Rendezvous Point (RP)

Rendezvous Point (RP) คือจุดที่ทุกหัวท่อเป็นจุดรวมสัญญาณสำหรับข้อมูล multicast ซึ่งหัวท่อที่ไม่สามารถสื่อสาร multicast ไปยังกลุ่มที่ต้องการได้จะส่งสู่ RP

ใน PIM Sparse Mode, RP จะเป็นจุดที่แรกที่แหล่งส่งข้อมูล multicast ติดต่อเพื่อขอให้ข้อมูลสู่ RP และจะจ่ายไปยังสมาชิกในกลุ่ม multicast

การตั้งค่า Spine เป็น RP จะช่วยให้การกระจายข้อมูล multicast บน Spine-to-Leaf network สามารถทำได้อย่างมีประสิทธิภาพและสามารถควบคุมได้

4. กำหนดให้ Leaf 1 (L1) และ Leaf 4 (L4) เป็น Last Hop Routers (LHR) สำหรับ Multicast Group**239.1.1.1 และ 239.2.2.2:**

Last Hop Router (LHR) คือเราเตอร์ที่อยู่ใกล้สุดกับผู้ใช้งานสำหรับปลายทางในกลุ่ม multicast

การตั้งค่า LHR ให้กับ Leaf 1 และ Leaf 4 ที่ทำให้ Leaf routers รับข้อมูล multicast โดยตรงจาก RP และทำหน้าที่กระจายข้อมูลนี้ไปยังผู้ใช้งานภายในเครือข่ายของพวงชนฯ

กลุ่ม multicast ที่สำคัญ เช่น 239.1.1.1 และ 239.2.2.2 จะต้องได้รับการกำหนดค่าจาก LHRs (Leaf routers) เพื่อให้ข้อมูลกระจายไปยังอุปกรณ์ปลายทางอย่างถูกต้อง

Config

Spine R1	Leaf L1	Leaf L4
<pre>conf feature pim ip pim rp-address 10.100.100.100 ip pim anycast-rp 10.100.100.10 10.1.0.1 ip pim anycast-rp 10.100.100.10 10.1.0.2 interface loopback 0 ip pim sparse-mode interface ethernet 1/1 ip pim sparse-mode interface loopback 1 ip pim sparse-mode description anycast-rp ip address 10.100.100.10/32 no sh ip router ospf 1 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 ip pim sparse-mode end copy run start</pre>	<pre>conf feature pim ip pim rp-address 10.100.100.100 interface loopback 0 ip pim sparse-mode interface ethernet 1/2 ip pim sparse-mode end copy run start</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 copy run start</pre>

การตรวจสอบ (Verification)**L1**

```
show ip mroute
leaf1# show ip mroute
IP Multicast Routing Table for VRF "default"
(*, 239.0.0.0/8), uptime: 00:14:16, pim ip
Incoming interface: Null, RPF nbr: 0.0.0.0
Outgoing interface list: (count: 0)

(*, 239.1.1.1/32), uptime: 00:03:11, nve ip pim
Incoming interface: Ethernet1/1, RPF nbr: 10.1.1.2
Outgoing interface list: (count: 1)
    nve1, uptime: 00:03:11, nve

(10.0.0.1/32, 239.1.1.1/32), uptime: 00:03:11, nve mrib ip pim
Incoming interface: loopback0, RPF nbr: 10.0.0.1
Outgoing interface list: (count: 1)
    Ethernet1/1, uptime: 00:02:13, pim

(10.1.1.1/32, 239.1.1.1/32), uptime: 00:03:11, pim mrib ip
Incoming interface: Ethernet1/1, RPF nbr: 10.1.1.1
Outgoing interface list: (count: 1)
    Ethernet1/1, uptime: 00:03:07, pim, (RPF)

(*, 239.2.2.2/32), uptime: 00:03:11, nve ip pim
Incoming interface: Ethernet1/1, RPF nbr: 10.1.1.2
Outgoing interface list: (count: 1)
    nve1, uptime: 00:03:11, nve

(10.0.0.1/32, 239.2.2.2/32), uptime: 00:03:11, nve mrib ip pim
Incoming interface: loopback0, RPF nbr: 10.0.0.1
Outgoing interface list: (count: 1)
    Ethernet1/1, uptime: 00:02:12, pim

(10.1.1.1/32, 239.2.2.2/32), uptime: 00:03:11, pim mrib ip
Incoming interface: Ethernet1/1, RPF nbr: 10.1.1.1
Outgoing interface list: (count: 1)
    Ethernet1/1, uptime: 00:03:06, pim, (RPF)
```

Leaf1 รับข้อมูล multicast จาก Ethernet1/1 และ loopback0 และส่งข้อมูลไปยัง nve1 สำหรับกลุ่ม 239.1.1.1 และ 239.2.2.2

L4

```
show ip mroute
leaf4# show ip mroute
IP Multicast Routing Table for VRF "default"
(*, 239.0.0.0/8), uptime: 00:14:13, pim ip
Incoming interface: Null, RPF nbr: 0.0.0.0
Outgoing interface list: (count: 0)

(*, 239.1.1.1/32), uptime: 00:03:33, nve ip pim
Incoming interface: Ethernet1/1, RPF nbr: 10.1.4.2
Outgoing interface list: (count: 1)
    nve1, uptime: 00:03:33, nve

(10.0.0.4/32, 239.1.1.1/32), uptime: 00:03:33, nve mrib ip pim
Incoming interface: loopback0, RPF nbr: 10.0.0.4
Outgoing interface list: (count: 1)
    Ethernet1/1, uptime: 00:03:10, pim

(10.1.1.1/32, 239.1.1.1/32), uptime: 00:03:33, nve mrib ip pim
Incoming interface: Ethernet1/1, RPF nbr: 10.1.4.2
Outgoing interface list: (count: 1)
    nve1, uptime: 00:03:33, mrib

(10.1.4.1/32, 239.1.1.1/32), uptime: 00:03:27, ip mrib pim
Incoming interface: Ethernet1/1, RPF nbr: 10.1.4.1
Outgoing interface list: (count: 1)
    Ethernet1/1, uptime: 00:03:27, mrib

(*, 239.2.2.2/32), uptime: 00:03:32, nve ip pim
Incoming interface: Ethernet1/1, RPF nbr: 10.1.4.2
Outgoing interface list: (count: 1)
    nve1, uptime: 00:03:32, nve

(10.0.0.4/32, 239.2.2.2/32), uptime: 00:03:32, nve mrib ip pim
Incoming interface: loopback0, RPF nbr: 10.0.0.4
Outgoing interface list: (count: 1)
    Ethernet1/1, uptime: 00:03:09, pim

(10.1.4.1/32, 239.2.2.2/32), uptime: 00:03:36, pim ip
Incoming interface: Ethernet1/1, RPF nbr: 10.1.4.1
Outgoing interface list: (count: 1)
    Ethernet1/1, uptime: 00:03:36, pim, (RPF)
```

Leaf4 รับข้อมูล multicast จาก Ethernet1/1 และ loopback0 และส่งข้อมูลไปยัง nve1 สำหรับกลุ่ม 239.1.1.1 และ 239.2.2.2

R1

show ip mroute

```
spine1# show ip mroute
IP Multicast Routing Table for VRF "default"
(*, 232.0.0.0/8), uptime: 00:16:05, pim ip
Incoming interface: Null, RPF nbr: 0.0.0.0
Outgoing interface list: (count: 0)

(*, 239.1.1.1/32), uptime: 00:04:42, pim ip
Incoming interface: loopback1, RPF nbr: 10.100.100.100
Outgoing interface list: (count: 2)
    Ethernet1/4, uptime: 00:04:38, pim
    Ethernet1/1, uptime: 00:04:42, pim

(10.0.0.1/32, 239.1.1.1/32), uptime: 00:03:48, pim mrrib ip
Incoming interface: Ethernet1/4, RPF nbr: 10.1.1.1, internal
Outgoing interface list: (count: 1)
    Ethernet1/4, uptime: 00:03:43, pim

(10.0.0.4/32, 239.1.1.1/32), uptime: 00:04:15, pim mrrib ip
Incoming interface: Ethernet1/4, RPF nbr: 10.1.4.1, internal
Outgoing interface list: (count: 2)
    Ethernet1/4, uptime: 00:04:15, pim, (RPF)
    Ethernet1/1, uptime: 00:04:15, pim

(10.1.1.1/32, 239.1.1.1/32), uptime: 00:12:34, ip pim
Incoming interface: Ethernet1/1, RPF nbr: 10.1.1.1, internal
Outgoing interface list: (count: 1)
    Ethernet1/4, uptime: 00:04:37, pim

(*, 239.2.2.2/32), uptime: 00:04:45, pim ip
Incoming interface: loopback1, RPF nbr: 10.100.100.100
Outgoing interface list: (count: 2)
    Ethernet1/4, uptime: 00:04:37, pim
    Ethernet1/1, uptime: 00:04:42, pim

(10.0.0.1/32, 239.2.2.2/32), uptime: 00:04:42, pim mrrib ip
Incoming interface: Ethernet1/1, RPF nbr: 10.1.1.1, internal
Outgoing interface list: (count: 1)
    Ethernet1/4, uptime: 00:04:42, pim

(10.0.0.4/32, 239.2.2.2/32), uptime: 00:04:14, pim mrrib ip
Incoming interface: Ethernet1/4, RPF nbr: 10.1.4.1, internal
Outgoing interface list: (count: 1)
    Ethernet1/4, uptime: 00:04:14, pim, (RPF)
    Ethernet1/1, uptime: 00:04:14, pim

(10.1.1.1/32, 239.2.2.2/32), uptime: 00:11:23, ip pim
Incoming interface: Ethernet1/1, RPF nbr: 10.1.1.1
Outgoing interface list: (count: 1)
    Ethernet1/4, uptime: 00:04:37, pim
```

Spine1 ให้รับข้อมูล multicast จาก loopback1 และ interface ต่างๆ เช่น Ethernet1/1 และ Ethernet1/4 และส่งข้อมูลไปยัง interface ต่างๆ ที่เกี่ยวข้องในการการส่งข้อมูล multicast เช่น Ethernet1/1 และ Ethernet1/4

show ip pim rp

```
spine1# show ip pim rp
PIM RP Status Information for VRF "default"
BSR disabled
Auto-RP disabled
BSR RP Candidate policy: None
BSR RP policy: None
Auto-RP Announce policy: None
Auto-RP Discovery policy: None

Anycast-RP 10.100.100.100 members:
    10.1.0.1* 10.1.0.2

RP: 10.100.100.100*, (0),
uptime: 00:16:41 priority: 255,
RP-source: (local),
group ranges:
    224.0.0.0/4
spine1# show ip pim rp
PIM RP Status Information for VRF "default"
BSR disabled
Auto-RP disabled
BSR RP Candidate policy: None
BSR RP policy: None
Auto-RP Announce policy: None
Auto-RP Discovery policy: None

Anycast-RP 10.100.100.100 members:
    10.1.0.1* 10.1.0.2

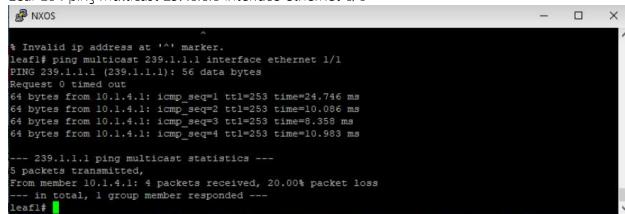
RP: 10.100.100.100*, (0),
uptime: 00:16:41 priority: 255,
RP-source: (local),
group ranges:
    224.0.0.0/4
```

- Spine1 ใช้ Anycast-RP โดยมี IP address 10.100.100.100 เป็น RP หลัก

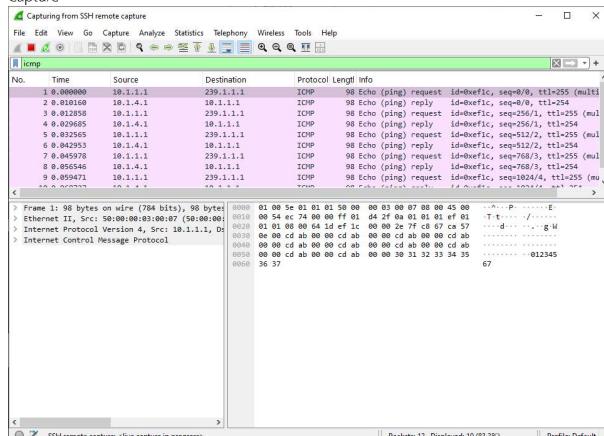
- Anycast-RP คือการที่ RP ที่แข็งแกร่งอย่างมาก เครื่อง (สมาร์ท) เช่น 10.1.0.1 และ 10.1.0.2 เพื่อให้ความสามารถในการ failover หากเครื่องหนึ่งไม่สามารถทำงานได้

- RP ที่ใช้งานในระบบเป็น RP ที่ถูกตั้งโดยอุปกรณ์เอง (local) และครอบคลุมกิจกรรม multicast ในช่วง 224.0.0.0/4

Leaf L1 : ping multicast 239.1.1.1 interface ethernet 1/1



Capture



อธิบายตัวอย่างการ Capture

Frame 1

98 bytes on wire (784 bits), 98 bytes captured (784 bits)

→ ขนาดของแพ็คเก็ตต่อ 98 ไบต์

Interface: sshdump.exe, id 0

→ จับแพ็คเก็ตผ่าน SSH Remote Capture

Ethernet II (Layer 2 - Data Link Layer)

Src (MAC Source): 50:00:00:03:00:07

→ อุปกรณ์ที่ส่งแพ็คเก็ต

Dst (MAC Destination): 01:00:5e:01:01:01 (IPv4 Multicast Address)

→ เป็น Multicast MAC Address ที่แปลงมาจาก IPv4 Multicast 239.1.1.1

IPv4 (Layer 3 - Network Layer)

Src (IP Source): 10.1.1.1

→ อุปกรณ์ที่ส่งแพ็คเก็ต

Dst (IP Destination): 239.1.1.1 (Multicast Group)

→ เป็น Multicast IP Address

ICMP (Layer 4 - Transport Layer)

→ เป็น Internet Control Message Protocol (ICMP) ซึ่งอาจเป็น

ICMP Echo Request (Ping) ไปยัง Multicast Group (239.1.1.1)

ICMP Message ซึ่งมีผลลัพธ์ของการควบคุมเครือข่าย

วิเคราะห์แพ็กเก็ต

- แพ็กเก็ตที่เป็น Multicast ICMP Message ซึ่งส่งจาก 10.1.1.1 → 239.1.1.1
- MAC Destination (01:00:5e:01:01:01) เป็น Multicast MAC ที่แปลงมาจาก 239.1.1.1
MAC Multicast ใน IPv4 ใช้รูปแบบ 01:00:5e:xx:xxxx
- ค่า 01:01:01 คือส่วนท้ายของ 239.1.1.1 (Group Address)
- เป็น ICMP Ping ไปยัง Multicast Group เพื่อทดสอบว่าอุปกรณ์ใดในเครือข่ายที่เป็นสมาชิกของ 239.1.1.1

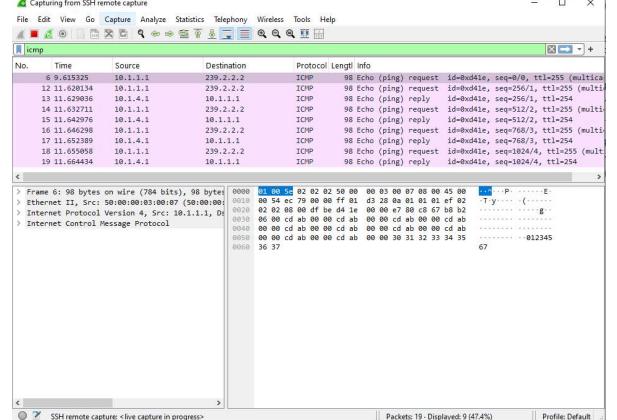
Leaf L1 : ping multicast 239.2.2.2 interface ethernet 1/1

```

NXOS
From member 10.1.4.1: 4 packets received, 20.00% packet loss
--- in total, 1 group member responded ---
leaf# 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 10.1.4.1: icmp_seq=1 ttl=253 time=13.35 ms
64 bytes from 10.1.4.1: icmp_seq=2 ttl=253 time=13.36 ms
64 bytes from 10.1.4.1: icmp_seq=3 ttl=253 time=13.37 ms
64 bytes from 10.1.4.1: icmp_seq=4 ttl=253 time=13.50 ms
--- 239.2.2.2 ping multicast statistics ---
5 packets transmitted, 5 packets received, 20.00% packet loss
From member 10.1.4.1: 4 packets received, 20.00% packet loss
--- in total, 1 group member responded ---
leaf# 

```

Capture



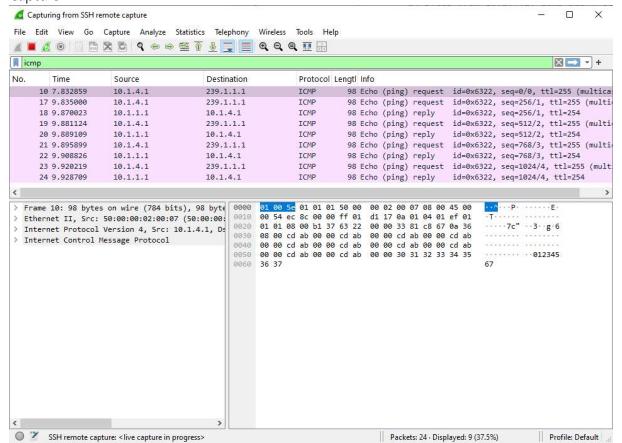
Leaf L4 : ping multicast 239.1.1.1 interface ethernet 1/1

```

NXOS
5 packets transmitted, 5 packets received, 0.00% packet loss
Round-trip min/avg/max = 9.127/10.068/14.837 ms
leaf# 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=14.665 ms
64 bytes from 10.1.1.1: icmp_seq=2 ttl=253 time=13.429 ms
64 bytes from 10.1.1.1: icmp_seq=3 ttl=253 time=10.052 ms
64 bytes from 10.1.1.1: icmp_seq=4 ttl=253 time=15.392 ms
--- 239.1.1.1 ping multicast statistics ---
5 packets transmitted, 4 packets received, 20.00% packet loss
--- in total, 1 group member responded ---
leaf# 

```

Capture



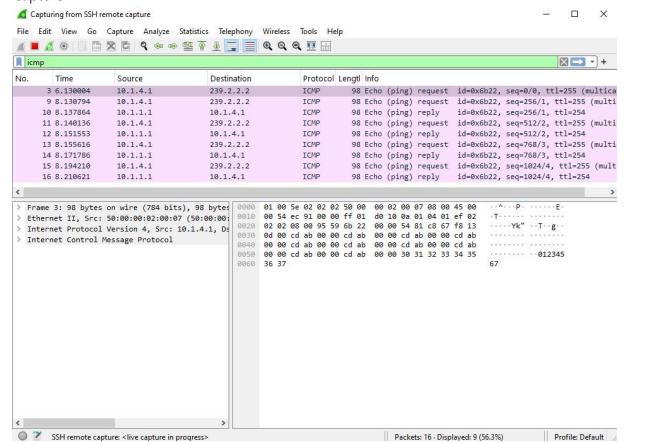
Leaf L4 : ping multicast 239.2.2.2 interface ethernet 1/1

```

NXOS
From member 10.1.1.1: 4 packets received, 20.00% packet loss
--- in total, 1 group member responded ---
leaf# 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=30.174 ms
64 bytes from 10.1.1.1: icmp_seq=2 ttl=253 time=13.626 ms
64 bytes from 10.1.1.1: icmp_seq=3 ttl=253 time=8.777 ms
64 bytes from 10.1.1.1: icmp_seq=4 ttl=253 time=11.875 ms
--- 239.2.2.2 ping multicast statistics ---
5 packets transmitted, 5 packets received, 20.00% packet loss
--- in total, 1 group member responded ---
leaf# 

```

Capture



VxLAN Configuration (Overlay Configuration)

ตอนนี้เราจำเป็นต้องทำการตั้งค่า VXLAN เนื่องด้วย ซึ่งการตั้งค่า VXLAN นี้จะต้องท่าน LEAF Switches เพื่อท่าน (L1 และ L4) หลังจากการตั้งค่า VXLAN เริ่มจะสามารถ End Hosts ที่อยู่ใน VLAN เดียวกัน ทั้งหมดสามารถ Ping ทางกันได้

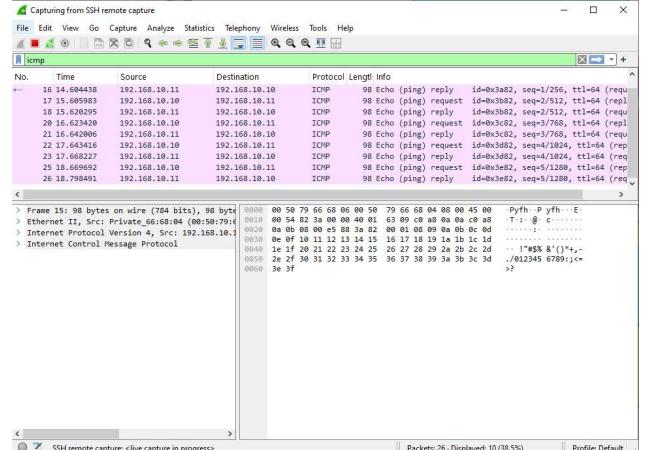
Leaf L1	Leaf L4
conf	conf
feature interface-vlan	feature interface-vlan
feature nv overlay	feature nv overlay
feature vn-segment-vlan-based	feature vn-segment-vlan-based
interface nve 1	interface nve 1
no sh	no sh
source loopback 0	source loopback 0
member vni 5010 mcast-group 239.1.1.1	member vni 5010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

การตรวจสอบ (Verification)

From Host H1, ping 192.168.10.11

```
Host> ping 192.168.10.11
84 bytes from 192.168.10.11 icmp_seq=1 ttl=64 time=26.917 ms
84 bytes from 192.168.10.11 icmp_seq=2 ttl=64 time=19.680 ms
84 bytes from 192.168.10.11 icmp_seq=3 ttl=64 time=17.572 ms
84 bytes from 192.168.10.11 icmp_seq=4 ttl=64 time=22.145 ms
84 bytes from 192.168.10.11 icmp_seq=5 ttl=64 time=21.738 ms
```

Capture



อธิบายตัวอย่างการ Capture

Frame

98 bytes on wire (784 bits), 98 bytes captured (784 bits)
→ ขนาดของแพ็คเก็ตคือ 98 ในที่

Interface: sshdump.exe, id 0
→ จับแพ็คเก็ตผ่าน SSH Remote Capture

Ethernet II (Layer 2 - Data Link Layer)

Src (MAC Source): 00:50:79:66:68:06
→ อุปกรณ์ที่ส่งแพ็คเก็ต มี MAC Address 00:50:79:66:68:06

Dst (MAC Destination): 00:50:79:66:68:04
→ อุปกรณ์ที่รับแพ็คเก็ต มี MAC Address 00:50:79:66:68:04

IPv4 (Layer 3 - Network Layer)

Src (IP Source): 192.168.10.11
→ อุปกรณ์ที่ส่งแพ็คเก็ต มี IP Address 192.168.10.11

Dst (IP Destination): 192.168.10.10
→ อุปกรณ์ที่รับแพ็คเก็ต มี IP Address 192.168.10.10

ICMP (Layer 4 - Transport Layer)

Protocol: ICMP (Internet Control Message Protocol)
→ แสดงว่าแพ็คเก็ตที่ส่ง ICMP Request (Ping) หรือ ICMP Error Message ที่ส่งจาก 192.168.10.11 ไปยัง 192.168.10.10

Spine R1

```
conf # เพิ่งโหลด configuration เพื่อกำหนดค่าทั่วๆ
hostname spine1 # ตั้งชื่ออุปกรณ์เป็น "spine1"
#boot nxos nxos.10.1.1.bin # (ถูกคอมมอนต์ออก) ใช้เพื่อกำหนดไฟล์ระบบปฏิบัติการที่ต้องบุก
feature ospf # เปิดใช้งานไปได้โดย OSPF บนอุปกรณ์
router ospf SI-Underlay # สร้างกระบวนการ OSPF และตั้งชื่อเป็น "SI-Underlay"
router-id 10.1.0.1 # กำหนด Router ID ของ OSPF เป็น 10.1.0.1
end # ออกจากโหมด configuration
conf # เพิ่งโหลด configuration
interface loopback 0 # เท้าไปกำหนดค่าของ Loopback 0
no sh # เปิดใช้งานอินเทอร์เฟซ (no shutdown)
ip address 10.1.0.1/32 # ตั้งค่า IP Address ให้ Loopback เป็น 10.1.0.1/32
ip router ospf SI-Underlay area 0 # กำหนดให้ Loopback 0 อยู่ใน OSPF Area 0
ip ospf network point-to-point # กำหนดว่า Loopback ใช้โหมด OSPF เป็น Point-to-Point
end # ออกจากโหมด configuration
conf # เพิ่งโหลด configuration
interface ethernet 1/1 # เท้าไปกำหนดค่าของ Ethernet 1/1
description to-leaf1-eth1/1 # ใส่คำอธิบายว่าเชื่อมต่อไปยัง Leaf1, Port 1/1
no sh # เปิดใช้งานอินเทอร์เฟซ (no shutdown)
no sw # ปิดโหมด switchport (กำหนดให้เป็น Layer 3)
ip address 10.1.1.2/30 # ตั้งค่า IP Address เป็น 10.1.1.2/30
ip router ospf SI-Underlay area 0 # เพิ่มอินเทอร์เฟซนี้ใน OSPF Area 0
ip ospf network point-to-point # ตั้งค่า OSPF Network Type เป็น Point-to-Point
mtu 9000 # กำหนดค่า MTU เป็น 9000 (Jumbo Frame)
interface ethernet 1/2 # เท้าไปกำหนดค่าของ Ethernet 1/2
description to-leaf2-eth1/1 # ใส่คำอธิบายว่าเชื่อมต่อไปยัง Leaf2, Port 1/1
no sw # ปิดโหมด switchport (กำหนดให้เป็น Layer 3)
ip address 10.1.2.2/30 # ตั้งค่า IP Address เป็น 10.1.2.2/30
ip router ospf SI-Underlay area 0 # เพิ่มอินเทอร์เฟซนี้ใน OSPF Area 0
ip ospf network point-to-point # ตั้งค่า OSPF Network Type เป็น Point-to-Point
mtu 9000 # กำหนดค่า MTU เป็น 9000 (Jumbo Frame)
no sh # เปิดใช้งานอินเทอร์เฟซ (no shutdown)
```

```
interface ethernet 1/3 # เท้าไปกำหนดค่าของ Ethernet 1/3
description to-leaf3-eth1/1 # ใส่คำอธิบายว่าเชื่อมต่อไปยัง Leaf3, Port 1/1
no sw # ปิดโหมด switchport (กำหนดให้เป็น Layer 3)
ip address 10.1.3.2/30 # ตั้งค่า IP Address เป็น 10.1.3.2/30
ip router ospf SI-Underlay area 0 # เพิ่มอินเทอร์เฟซนี้ใน OSPF Area 0
ip ospf network point-to-point # ตั้งค่า OSPF Network Type เป็น Point-to-Point
mtu 9000 # กำหนดค่า MTU เป็น 9000 (Jumbo Frame)
no sh # เปิดใช้งานอินเทอร์เฟซ (no shutdown)
interface ethernet 1/4 # เท้าไปกำหนดค่าของ Ethernet 1/4
description to-leaf4-eth1/1 # ใส่คำอธิบายว่าเชื่อมต่อไปยัง Leaf4, Port 1/1
no sw # ปิดโหมด switchport (กำหนดให้เป็น Layer 3)
ip address 10.1.4.2/30 # ตั้งค่า IP Address เป็น 10.1.4.2/30
ip router ospf SI-Underlay area 0 # เพิ่มอินเทอร์เฟซนี้ใน OSPF Area 0
ip ospf network point-to-point # ตั้งค่า OSPF Network Type เป็น Point-to-Point
mtu 9000 # กำหนดค่า MTU เป็น 9000 (Jumbo Frame)
no sh # เปิดใช้งานอินเทอร์เฟซ (no shutdown)
end # ออกจากโหมด configuration
```

รวม ๆ เล้า คำสั่งพากนี้ใช้อะไร?

คำสั่งทั้งหมดนี้เป็น การกำหนดค่าอุปกรณ์ Cisco Nexus (NX-OS) ที่กำหนดให้เป็น Spine Switch ใน Data Center ซึ่งใช้ OSPF (Open Shortest Path First) เป็นโปรดcols สำหรับ Underlay Network ในโครงสร้าง Spine-Leaf

คำสั่งพวกนี้คืออะไร?

- สุปรายุค:
- อุปกรณ์นี้คือ "Spine Switch" ใน Spine-Leaf Data Center Network
- ใช้ OSPF เป็น Routing Protocol เพื่อเชื่อมต่อระหว่าง Spine และ Leaf
- ทำให้แต่ละ Leaf เชื่อมต่อถึงกันผ่าน Spine โดยมีการ Routing อัตโนมัติ
- เพิ่มประสิทธิภาพด้วย Jumbo Frame (MTU 9000)
- ใช้ Subnet /30 สำหรับ Point-to-Point Links เพื่อประยุต IP

สรุปสั้น ๆ:

คำสั่งนี้ใช้ในการตั้งค่า Spine Switch ที่รับ OSPF สำหรับ Spine-Leaf Network ใน Data Center โดยมี Layer 3 Routing, MTU 9000 และการใช้ Loopback สำหรับ Router ID

Leaf L1

```
conf # เพิ่งโหลด configuration เพื่อกำหนดค่าทั่วๆ
hostname leaf1 # ตั้งชื่ออุปกรณ์เป็น "leaf1"
#boot nxos nxos.10.1.1.bin # (ถูกคอมมอนต์ออก) ใช้กำหนด NX-OS เวอร์ชันที่ต้องการบุก
feature ospf # เปิดใช้งานไฟเซอร์ OSPF
router ospf SI-Underlay # สร้าง OSPF process ชื่อ "SI-Underlay"
router-id 10.0.0.1 # กำหนด Router ID ของอุปกรณ์เป็น 10.0.0.1
end # ออกจากโหมด configuration
conf # เพิ่งโหลด configuration
interface loopback 0 # เท้าไปกำหนดค่าของ Loopback 0
no sh # เปิดใช้งานอินเทอร์เฟซ (no shutdown)
ip address 10.0.0.1/32 # ตั้งค่า IP Address เป็น 10.0.0.1/32
ip router ospf SI-Underlay area 0 # กำหนดให้ Loopback 0 อยู่ใน OSPF Area 0
ip ospf network point-to-point # ตั้งค่า OSPF Network Type เป็น Point-to-Point
end # ออกจากโหมด configuration
conf # เพิ่งโหลด configuration
interface ethernet 1/1 # เท้าไปกำหนดค่าของ Ethernet 1/1
description to-spine1-eth1/0 # ใส่คำอธิบายว่าเชื่อมต่อไปยัง Spine1, Port 1/0
no sh # เปิดใช้งานอินเทอร์เฟซ (no shutdown)
no sw # ปิดโหมด switchport (กำหนดให้เป็น Layer 3)
ip address 10.1.1.1/30 # ตั้งค่า IP Address เป็น 10.1.1.1/30
ip router ospf SI-Underlay area 0 # เพิ่มอินเทอร์เฟซนี้ใน OSPF Area 0
ip ospf network point-to-point # ตั้งค่า OSPF Network Type เป็น Point-to-Point
mtu 9000 # กำหนดค่า MTU เป็น 9000 (Jumbo Frame)
interface ethernet 1/2 # เท้าไปกำหนดค่าของ Ethernet 1/2
description to-spine2-eth1/0 # ใส่คำอธิบายว่าเชื่อมต่อไปยัง Spine2, Port 1/0
no sw # ปิดโหมด switchport (กำหนดให้เป็น Layer 3)
ip address 10.2.1.1/30 # ตั้งค่า IP Address เป็น 10.2.1.1/30
ip router ospf SI-Underlay area 0 # เพิ่มอินเทอร์เฟซนี้ใน OSPF Area 0
ip ospf network point-to-point # ตั้งค่า OSPF Network Type เป็น Point-to-Point
mtu 9000 # กำหนดค่า MTU เป็น 9000 (Jumbo Frame)
no sh # เปิดใช้งานอินเทอร์เฟซ (no shutdown)
end # ออกจากโหมด configuration
```

สรุป:

- Ethernet 1/1 เพื่อมต่อไปยัง Spine1 ด้วย IP 10.1.1.1/30
- Ethernet 1/2 เพื่อมต่อไปยัง Spine2 ด้วย IP 10.2.1.1/30
- ปิด switchport เพื่อใช้ Layer 3 Routing
- MTU 9000 ใช้สำหรับ Jumbo Frame

สรุปรวม:

คำสั่งนี้ตั้งค่า Leaf Switch ที่เชื่อมกับ Spine Switch โดยใช้ OSPF สำหรับ Routing ในเครือข่ายแบบ Spine-Leaf

- ใช้ OSPF เพื่อเชื่อมกับ Spine1 และ Spine2
- ใช้ OSPF ใน Area 0
- รองรับ Jumbo Frame (MTU 9000)
- ใช้ Loopback 0 เป็น Router ID

Leaf L4

```
conf # เพิ่งโหลด configuration เพื่อตั้งค่าอุปกรณ์
hostname leaf4 # ตั้งชื่ออุปกรณ์เป็น "leaf4"
#boot nxos nxos.10.1.1.bin # (ถูกคอมมอนต์ออก) ใช้กำหนด NX-OS เวอร์ชันที่ต้องการบุก
feature ospf # เปิดใช้งานไฟเซอร์ OSPF
router ospf SI-Underlay # สร้าง OSPF process ชื่อ "SI-Underlay"
router-id 10.0.0.4 # กำหนด Router ID เป็น 10.0.0.4
end # ออกจากโหมด configuration
conf # เพิ่งโหลด configuration
interface loopback 0 # เลือกอินเทอร์เฟซ Loopback0
no sh # เปิดใช้งานอินเทอร์เฟซ (no shutdown)
ip address 10.0.0.4/32 # ตั้งค่า IP เป็น 10.0.0.4/32
ip router ospf SI-Underlay area 0 # เพิ่มอินเทอร์เฟซนี้ใน OSPF Area 0
ip ospf network point-to-point # ตั้งค่า OSPF Network Type เป็น Point-to-Point
end # ออกจากโหมด configuration
conf # เพิ่งโหลด configuration
interface ethernet 1/1 # เท้าไปกำหนดค่าของ Ethernet 1/1
description to-spine1-eth1/3 # ใส่คำอธิบายว่าเชื่อมต่อไปยัง Spine1, Port 1/3
no sh # เปิดใช้งานอินเทอร์เฟซ (no shutdown)
no sw # ปิดโหมด switchport (Layer 3 Mode)
ip address 10.1.4.1/30 # ตั้งค่า IP Address เป็น 10.1.4.1/30
ip router ospf SI-Underlay area 0 # เพิ่มอินเทอร์เฟซนี้ใน OSPF Area 0
```

```

ip ospf network point-to-point # ตั้งค่า OSPF Network Type เป็น Point-to-Point
mtu 9000 # ตั้งค่า MTU เป็น 9000 (รองรับ Jumbo Frame)
interface ethernet 1/2 # เพิ่มเป็นก้านต่อของ Ethernet 1/2
description to-spine2-eth1/3 # ใส่คำอธิบายว่าชื่อต่อไปยัง Spine2, Port 1/3
no shutdown # ปิดโหมด switchport (Layer 3 Mode)
ip address 10.2.4.1/30 # ตั้งค่า IP Address เป็น 10.2.4.1/30
ip router ospf SI-Underlay area 0 # เพิ่มอินเทอร์เฟซนี้เข้า OSPF Area 0
ip ospf network point-to-point # ตั้งค่า OSPF Network Type เป็น Point-to-Point
mtu 9000 # ตั้งค่า MTU เป็น 9000 (รองรับ Jumbo Frame)
no shutdown # เปิดให้งานอินเทอร์เฟซ (no shutdown)
end # ออกจากโหมด configuration

```

สรุป:

- Ethernet 1/1 เพื่อมัน Spinne1 (10.1.4.1/30)
- Ethernet 1/2 เพื่อมัน Spinne2 (10.2.4.1/30)
- Layer 3 (ปิด Switchport Mode)
- ใช้ OSPF ใน Area 0
- รองรับ Jumbo Frame (MTU 9000)

สรุปรวม

คำสั่งนี้ ตั้งค่า Leaf Switch (Leaf4) ที่เชื่อมต่อ กับ Spine Switch โดยใช้ OSPF เป็น Routing Protocol

- Leaf4 เพื่อมัน Spinne1 และ Spinne2
- ใช้ OSPF ใน Area 0
- รองรับ Jumbo Frame (MTU 9000)
- ใช้ Loopback0 เป็น Router ID

สรุปรวม: คำสั่งของ Spinne1, Leaf1 และ Leaf4 ใช้ทำอะไร?

คำสั่งทั้งหมดได้รับ ตั้งค่าเครือข่ายแบบ Spine-Leaf โดยใช้ OSPF เป็น Routing Protocol เพื่อให้ Switch ในเครือข่ายสามารถถูกต่อ กับ Layer 3 (IP Routing แทนที่จะใช้ Switching แบบเดิม) ประโยชน์ของการตั้งค่านี้

- ใช้ Layer 3 Routing แทน Switching → ลดปัญหา Broadcast Storm
- ใช้ OSPF เพื่อแลกเปลี่ยนเส้นทางอัตโนมัติ → รองรับการขยายเครือข่าย
- รองรับ MTU 9000 (Jumbo Frames) → เพิ่มประสิทธิภาพในการส่งข้อมูล
- ใช้ Network Type: Point-to-Point → ลด LSA ที่ไม่จำเป็น

- เปิดใช้งาน Multicast Routing บนทุกพอร์ตที่เกี่ยวข้อง
- ใช้ OSPF เพื่อให้ RP สามารถเข้าถึงได้จากทุกอุปกรณ์
- ◆ สัน្តิคือ: คำสั่งเหล่านี้ช่วยให้เครือข่ายสามารถรองรับ Multicast Traffic ได้โดยใช้ Anycast RP และ OSPF ⚡

Leaf L1

```

conf # เข้าโหมด configuration
feature pim # เปิดใช้งานเพื่อจัด PIM เพื่อรองรับการทำงานของ Multicast
ip pim rp-address 10.100.100.100 # ตั้งค่า RP (Rendezvous Point) ให้เป็น 10.100.100.100
interface loopback 0 # กำหนดค่า Interface Loopback0
ip pim sparse-mode # เปิดใช้ PIM Sparse Mode บน Loopback0 เพื่อรองรับ Multicast
interface ethernet 1/1 # กำหนดค่า Interface Ethernet 1/1
ip pim sparse-mode # เปิดใช้ PIM
interface ethernet 1/2 # กำหนดค่า Interface Ethernet 1/2
ip pim sparse-mode # เปิดใช้ PIM Sparse Mode บนพอร์ตนี้เพื่อรองรับ Multicast
end # ออกจากโหมด configuration และบันทึกการตั้งค่า

```

สรุป: คำสั่งนี้ใช้ทำอะไร?

- เปิดใช้งาน PIM สำหรับ Multicast Routing
- ตั้งค่า RP ที่อยู่ 10.100.100.100 เพื่อเป็นศูนย์กลางการรับส่ง Multicast
- เปิดใช้งาน PIM Sparse Mode บน Loopback0 และ Ethernet Interfaces
- ช่วยให้สามารถส่งข้อมูล Multicast ในเครือข่ายที่ใช้ PIM ได้อย่างมีประสิทธิภาพ

- ◆ สัน្តิคือ: คำสั่งเหล่านี้ช่วยให้ Switch/Router รองรับ Multicast Traffic ผ่าน PIM Sparse Mode โดยมี RP เป็นศูนย์กลาง ⚡

Leaf L4

```

conf # เข้าโหมด configuration
feature pim # เปิดใช้งานเพื่อจัด PIM เพื่อรองรับ Multicast Routing
ip pim rp-address 10.100.100.100 # กำหนด RP (Rendezvous Point) ที่จะใช้เป็นศูนย์กลางสำหรับ Multicast Group
interface loopback 0 # กำหนดค่า Interface Loopback0
ip pim sparse-mode # เปิดใช้ PIM Sparse Mode บน Loopback0 เพื่อรองรับ Multicast
interface ethernet 1/1 # กำหนดค่า Interface Ethernet 1/1
ip pim sparse-mode # เปิดใช้ PIM Sparse Mode บนพอร์ตนี้เพื่อรองรับ Multicast
interface ethernet 1/2 # กำหนดค่า Interface Ethernet 1/2

```

สรุป:

การใช้ Unicast Routing ใน การตั้งค่า Spine R1, Leaf L1, และ Leaf L4 ผ่าน OSPF มีวัตถุประสงค์ใน การกำหนดเส้นทางที่ดีที่สุดสำหรับส่งข้อมูลจาก Router ไปยัง Router หรือจาก Switch ไปยัง Server ในเครือข่ายที่ไม่มี การใช้ OSPF ช่วยให้สามารถจัดการเส้นทางได้อย่างมีประสิทธิภาพ, รองรับการเดินทางเครือข่าย, และมีความเสถียรในระยะยาว.

Multicast Configuration

Spine R1

```

conf # เข้าโหมด configuration
feature pim # เปิดใช้งานเพื่อจัด PIM (Protocol Independent Multicast) เพื่อให้สามารถส่งข้อมูล Multicast ได้
ip pim rp-address 10.100.100.100 # กำหนด RP(Rendezvous Point) ที่จะใช้ใน Multicast Routing ให้เป็น 10.100.100.100
ip pim anycast-rp 10.100.100.100 10.1.0.1 # กำหนด AnycastRP ให้เป็น IP 10.1.0.1 เป็นหนึ่งใน RP ที่รองรับ 10.100.100.100
ip pim anycast-rp 10.100.100.100 10.1.0.2 # กำหนด Anycast RP ให้เป็น IP 10.1.0.2 เป็นอีก RP ที่รองรับ 10.100.100.100
interface loopback 0 # กำหนดค่า Loopback Interface 0 ซึ่งมีที่ใช้เป็น Router ID
ip pim sparse-mode # เปิดใช้ PIM Sparse Mode บน Loopback0 เพื่อให้รองรับ Multicast
interface loopback 1 # กำหนดค่า Loopback Interface 1 ซึ่งจะใช้เป็น Anycast RP
ip pim sparse-mode # เปิดใช้ PIM Sparse Mode บน Loopback1 เพื่อให้รองรับ Multicast
description anycast-rp # กำหนดค่าอินบoks ให้ Interface นี้เป็น Anycast RP
ip address 10.100.100.100/32 # ตั้งค่า IP Address ให้ Loopback1 เป็น 10.100.100.100 (ที่เป็น RP Address)
no shutdown # เปิดให้งาน Interface (No Shutdown)
ip router ospf SI-Underlay area 0 # โฆษณา IP ของ Loopback1 บน OSPF Area 0 เพื่อให้ Switch ยื่นสามารถเข้าถึง RP
interface ethernet 1/1 # กำหนดค่า Interface Ethernet 1/1
ip pim sparse-mode # เปิดใช้ PIM Sparse Mode บนพอร์ตนี้เพื่อให้รองรับ Multicast
interface ethernet 1/2 # กำหนดค่า Interface Ethernet 1/2
ip pim sparse-mode # เปิดใช้ PIM Sparse Mode บนพอร์ตนี้เพื่อให้รองรับ Multicast
interface ethernet 1/3 # กำหนดค่า Interface Ethernet 1/3
ip pim sparse-mode # เปิดใช้ PIM Sparse Mode บนพอร์ตนี้เพื่อให้รองรับ Multicast
interface ethernet 1/4 # กำหนดค่า Interface Ethernet 1/4
ip pim sparse-mode # เปิดใช้ PIM Sparse Mode บนพอร์ตนี้เพื่อให้รองรับ Multicast

```

สรุป: คำสั่งนี้ใช้ทำอะไร?

- เปิดใช้งาน PIM (Multicast Routing Protocol)
- ตั้งค่า Anycast RP ให้ทำงานร่วมกันเพื่อความทนทาน

เปิดใช้งาน Multicast Routing บนทุกพอร์ตที่เกี่ยวข้อง

ตั้งค่า OSPF เพื่อให้ RP สามารถเข้าถึงได้จากทุกอุปกรณ์

◆ สัน្តิคือ: คำสั่งเหล่านี้ช่วยให้เครือข่ายสามารถรองรับ Multicast Traffic ได้โดยใช้ Anycast RP และ OSPF ⚡

สรุป: คำสั่งนี้ใช้ทำอะไร?

- เปิดใช้งาน PIM สำหรับ Multicast Routing
- ตั้งค่า RP ที่อยู่ 10.100.100.100 เพื่อเป็นศูนย์กลางการรับส่ง Multicast
- เปิดใช้งาน PIM Sparse Mode บน Loopback0 และ Ethernet Interfaces
- ช่วยให้สามารถส่งข้อมูล Multicast ในเครือข่ายที่ใช้ PIM ได้อย่างมีประสิทธิภาพ

◆ สัน្តิคือ: คำสั่งเหล่านี้ช่วยให้ Switch/Router รองรับ Multicast Traffic ผ่าน PIM Sparse Mode โดยมี RP เป็นศูนย์กลาง ⚡

สรุปวัตถุประสงค์ทั้งหมด

คำสั่งเหล่านี้ใช้ในการตั้งค่า Multicast Routing โดยใช้ PIM Sparse Mode และการกำหนด RP เพื่อช่วยในการกระจายข้อมูล Multicast ไปยังทุกในเครือข่าย. การใช้ RP ทำให้สามารถลดภาระและเส้นทางการส่งข้อมูล Multicast ในปัจจุบัน ให้ดีขึ้น ผู้ใช้สามารถเข้าถึงข้อมูลที่ต้องการได้อย่างมีประสิทธิภาพ.

ท่ามถือใช้ Multicast Routing?

- Multicast Routing เป็นการส่งข้อมูลไปยังหลายๆ อุปกรณ์ (Receiver) พร้อมกัน โดยไม่ต้องส่งข้อมูลไปยังแต่ละอุปกรณ์ แยกกัน ที่ใช้ประยุกต์แบบเครือข่ายและทรัพยากร.
- PIM Sparse Mode เป็น moden ที่เหมาะสมสำหรับเครือข่ายที่มีผู้รับ Multicast ไม่มาก เนื่องจากนั้นใช้ RP เพื่อจัดการการส่งข้อมูลที่ต้องการมีประสิทธิภาพ.

VXLAN

Leaf L1

```

conf # เข้าโหมดการกำหนดค่า (Configuration mode) ของอุปกรณ์
feature interface-vlan # เปิดใช้งานเพื่อจัด VLAN interface ซึ่งจะให้สามารถกำหนดค่า VLAN interfaces ได้
feature nv overlay # เปิดใช้งานเพื่อจัด NV overlay ซึ่งรองรับการใช้งาน VXLAN ซึ่งเป็นรูปแบบของเครือข่าย overlay
feature vn-segment-vlan-based # เปิดใช้งานเพื่อจัด VN segment แบบใช้ VLAN ซึ่งชื่อในไฟ VLAN กับ VN segments สำหรับ VXLAN
interface nve 1 # กำหนดค่า NVE (Network Virtualization Edge) interface หมายเลข 1 สำหรับการ encapsulate VXLAN
no shutdown # ปิดการทํางานในสถานะ shutdown ของ interface นี้ (เปิดใช้งาน interface)
source loopback 0 # ให้ Loopback interface หมายเลข 0 เป็นแหล่งที่มาของออดิโอและเสียง VXLAN
member vni 5010 mcast-group 239.1.1.1 # กำหนดค่า VXLAN Network Identifier (VNI) 5010 และกลุ่ม multicast สำหรับ VNI นี้เป็น 239.1.1.1

```

```

member vni 5020 mcast-group 239.2.2.2 # กำหนดค่า VNI 5020 และกลุ่ม multicast สำหรับ VNI นี้เป็น 239.2.2.2
vlan 10 # กำหนด VLAN หมายเลข 10
vn-segment 5010 # เพิ่มโง่ VLAN 10 กับ VN Segment หมายเลข 5010 สำหรับการใช้งาน VXLAN
vlan 20 # กำหนด VLAN หมายเลข 20
vn-segment 5020 # เพิ่มโง่ VLAN 20 กับ VN Segment หมายเลข 5020 สำหรับการใช้งาน VXLAN
interface ethernet 1/3 # กำหนดค่า interface Ethernet หมายเลข 1/3
sw mode access # กำหนดโหมดของ interface นี้เป็น Access (โหมดที่ใช้งานกับอุปกรณ์ที่เป็นส่วนหนึ่งของ VLAN)
sw access vlan 10 # กำหนดให้ interface นี้เชื่อมโยงกับ VLAN 10
interface ethernet 1/4 # กำหนดค่า interface Ethernet หมายเลข 1/4
sw mode access # กำหนดโหมดของ interface นี้เป็น Access
sw access vlan 20 # กำหนดให้ interface นี้เชื่อมโยงกับ VLAN 20
end # ออกจากโหมดการกำหนดค่า

```

คำสั่งที่ได้มานั้นใช้สำหรับการตั้งค่า **VXLAN (Virtual Extensible LAN)** บนอุปกรณ์เครือข่ายเพื่อสร้างเครือข่ายเสมือน (Virtual Network) โดยสามารถแยก VLAN อื่นๆ ออกจากกันโดยไม่ต้องเปลี่ยน IP หรือชื่อเครือข่ายเดียวกัน ด้วยการใช้ **VNI (VXLAN Network Identifier)** และ **multicast** สำหรับการสื่อสารระหว่างอุปกรณ์ในแหล่งเดียวกัน.

สรุปการใช้งาน:

- สร้าง VXLAN: ใช้ nve interface, กำหนด vni และ mcast-group เพื่อกำหนด encapsulate ข้อมูลใน VXLAN.
- กำหนด VLANs: เพิ่มโง่และ VLAN กับ VN segment สำหรับการทำมาาร์ชร่วมกับ VXLAN.

3. ตั้งค่า Interfaces: กำหนดโหมด access สำหรับ interfaces เพื่อเชื่อมต่อ กับ VLAN ที่กำหนด.

โดยรวมแล้ว การกำหนดค่าด้านบนจะให้ความสามารถสำหรับการตั้งค่าและจัดการเครือข่ายเสมือนหลายๆ เครือข่ายได้ภายในเครือข่ายเดียว, หมายความว่าทุกๆ VLAN จะสามารถสื่อสารกันได้โดยไม่ต้องเปลี่ยน IP หรือชื่อเครือข่าย.

Leaf L4

```

conf # เข้าสู่โหมดการกำหนดค่า (Configuration mode) ของอุปกรณ์
feature interface-vlan # เปิดใช้งานฟีเจอร์ VLAN interface ซึ่งช่วยให้สามารถกำหนดค่า VLAN interfaces ได้
feature nv overlay # เปิดใช้งานฟีเจอร์ NV overlay ซึ่งรองรับการใช้งาน VXLAN สำหรับการสร้างเครือข่ายเสมือน
feature vr-segment-vlan-based # เปิดใช้งานฟีเจอร์ VN segment แบบใช้ VLAN ซึ่งเพิ่มโง่ VLAN กับ VN segments
สำหรับ VXLAN

interface nve 1 # กำหนดค่า NVE (Network Virtualization Edge) interface หมายเลข 1 สำหรับการ encapsulate
VXLAN

no sh # ปิดการทำงานในสถานะ shutdown ของ interface นี้ (ปิดใช้งาน interface)

source loopback 0 # ใช้ Loopback interface หมายเลข 0 เป็นแหล่งที่มาของยอดเดรสสำหรับการ encapsulate VXLAN

```

เพื่อใส่คำสั่งทั้งหมดใน Spine R1, Leaf L1, และ Leaf L4 จะเกิดการตั้งค่าต่าง ๆ ที่เกี่ยวข้องกับการทำงานของ VXLAN (Virtual Extensible LAN) และ Multicast Routing โดยเฉพาะการสร้าง Overlay Network ที่ทำให้สามารถถ่ายทอดข้อมูลไปยัง Data Center ได้โดยไม่จำเป็นต้องปรับเปลี่ยนโครงสร้างเครือข่ายหลัก (Underlay Network) มากนัก การทำงานที่เกิดขึ้นจะมีดังนี้:

1. การตั้งค่า VXLAN

- Leaf L1 และ Leaf L4 จะทำงานเป็น VXLAN Gateway โดยใช้ NVE (Network Virtualization Edge) interface เพื่อยื่มต่อ VXLAN Overlay Network และ Underlay Network.
- ในค่าสั่งที่ให้ไป:
 - VLAN 10 จะถูกแยกกับ VNI 5010 และ VLAN 20 จะถูกแยกกับ VNI 5020.
 - ทั้งสอง Leaf switches จะใช้ Multicast Groups (239.1.1 และ 239.2.2.2) สำหรับการส่งข้อมูลระหว่าง VXLAN segments ที่กำหนดไว้.
 - Source IP Address สำหรับ VXLAN encapsulation จะถูกกำหนดเป็น Loopback 0.

2. การใช้งาน Multicast Routing

- Multicast Group ที่ตั้งค่าใน ip pim rp-address และ ip pim anycast-rp ให้เพื่อส่งข้อมูลในรูปแบบ Multicast ระหว่าง Leaf และ Spine switches.
- การตั้งค่า PIM (Protocol Independent Multicast) จะช่วยให้การส่งข้อมูลใน VXLAN Overlay network สามารถทำได้โดยไม่ต้องใช้ Unicast Routing ในการจัดการข้อมูล, ทั้งนี้ให้การกระจายข้อมูลเป็นไปอย่างมีประสิทธิภาพ มากกว่าในเครือข่ายที่มีหลายอุปกรณ์.

3. การตั้งค่า VLAN กับ VXLAN

- กำหนดค่า sw mode access และ sw access vlan ไม้แต่ละ interface จะกำหนดให้ Leaf L1 และ Leaf L4 เชื่อมต่อ กับอุปกรณ์ใน VLAN 10 และ VLAN 20 ผ่าน Ethernet interfaces ที่กำหนด.
- VLAN 10 จะใช้ VNI 5010 และ VLAN 20 จะใช้ VNI 5020 เพื่อสร้าง VXLAN Overlay network ที่สามารถขยายไปได้ไกลขึ้น.

ผลลัพธ์ที่เกิดขึ้น:

- Overlay Network ที่ทำงานผ่าน VXLAN จะเชื่อมต่อ Leaf switches กับ Spine switches.
- Multicast Routing จะช่วยให้การกระจายข้อมูลระหว่าง VXLAN segments เป็นไปอย่างมีประสิทธิภาพ.
- การตั้งค่า VLAN 10 และ VLAN 20 กับ VXLAN ผ่าน NVE interfaces จะช่วยให้การส่งข้อมูลใน VXLAN Overlay Network สามารถเกิดขึ้นได้ใน Data Center ที่มีการใช้งาน Multicast.

สรุป:

การใส่คำสั่งที่ได้ใน Spine R1, Leaf L1, และ Leaf L4 จะทำให้เกิดการตั้งค่า VXLAN Overlay Network ที่ใช้ Multicast เพื่อ กระจายข้อมูลระหว่าง VXLAN segments. นอกจากนี้ยังช่วยให้สามารถถ่ายทอดข้อมูลไปยัง Data Center ได้โดยไม่จำเป็นต้อง เปลี่ยนแปลงโครงสร้างของ Underlay Network.

```

member vni 5010 mcast-group 239.1.1.1 # กำหนดค่า VXLAN Network Identifier (VNI) 5010 และกลุ่ม multicast สำหรับ VNI นี้เป็น 239.1.1.1
member vni 5020 mcast-group 239.2.2.2 # กำหนดค่า VNI 5020 และกลุ่ม multicast สำหรับ VNI นี้เป็น 239.2.2.2
vlan 10 # กำหนด VLAN หมายเลข 10
vn-segment 5010 # เพิ่มโง่ VLAN 10 กับ VN Segment หมายเลข 5010 สำหรับการใช้งาน VXLAN
vlan 20 # กำหนด VLAN หมายเลข 20
vn-segment 5020 # เพิ่มโง่ VLAN 20 กับ VN Segment หมายเลข 5020 สำหรับการใช้งาน VXLAN
interface ethernet 1/3 # กำหนดค่า interface Ethernet หมายเลข 1/3
sw mode access # กำหนดโหมดของ interface นี้เป็น Access (โหมดที่ใช้งานกับอุปกรณ์ที่เป็นส่วนหนึ่งของ VLAN)
sw access vlan 10 # กำหนดให้ interface นี้เชื่อมโยงกับ VLAN 10
interface ethernet 1/4 # กำหนดค่า interface Ethernet หมายเลข 1/4
sw mode access # กำหนดโหมดของ interface นี้เป็น Access
sw access vlan 20 # กำหนดให้ interface นี้เชื่อมโยงกับ VLAN 20
end # ออกจากโหมดการกำหนดค่า

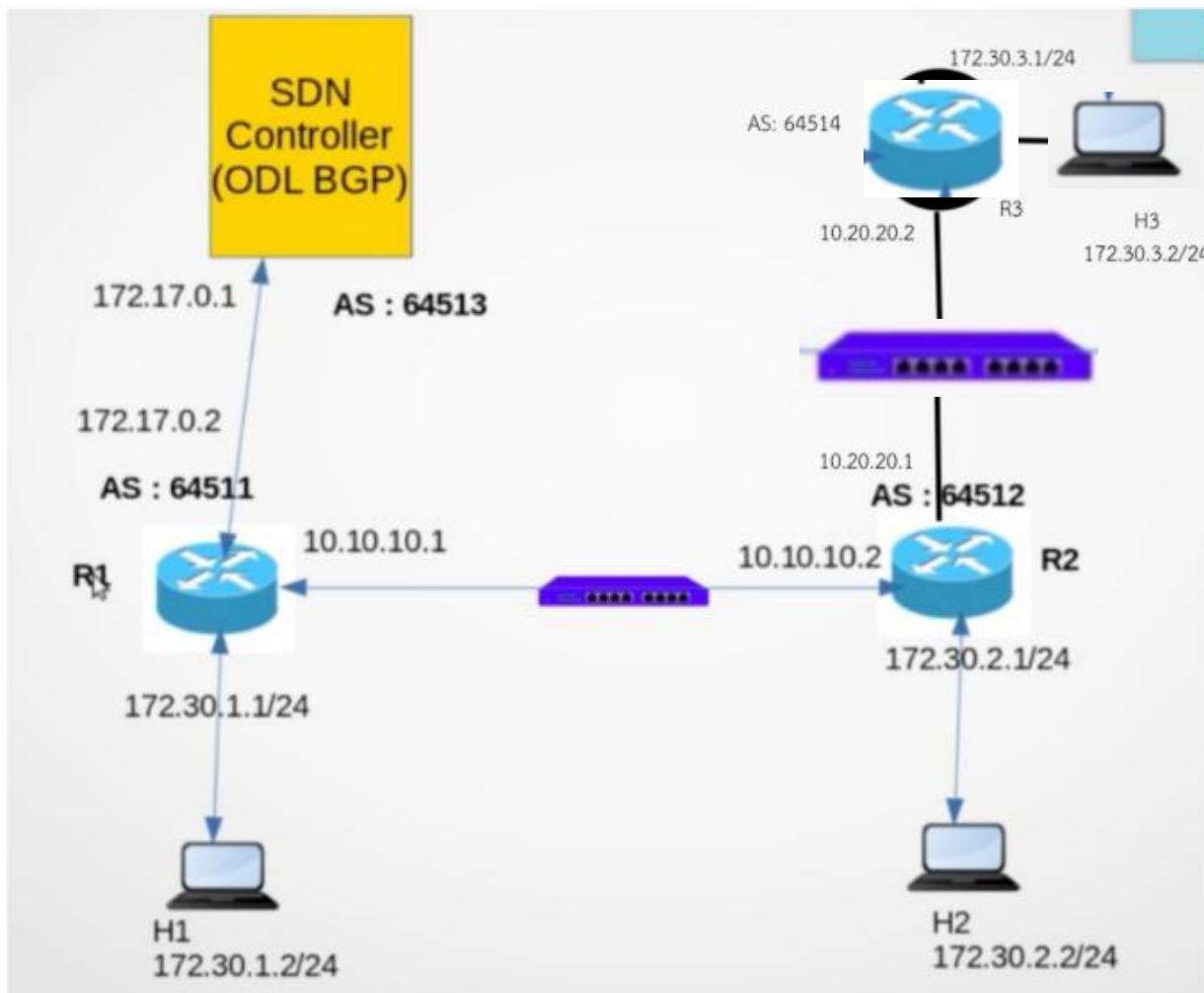
```

สรุปการใช้งาน:

คำสั่งที่ได้ในการตั้งค่า VXLAN (Virtual Extensible LAN) บนอุปกรณ์เครือข่ายเพื่อให้สามารถสร้างและจัดการเครือข่ายเสมือน (Virtual Network) โดยการเพิ่มโง่ VLAN และ VNI (VXLAN Network Identifier) ให้สามารถทำงานร่วมกันได้ตามกำหนด.

- ใช้ nve interface สำหรับการ encapsulate ข้อมูลใน VXLAN.
- ตั้งค่า VLAN 10 และ VLAN 20 และเพิ่มโง่กับ VN segments (5010 และ 5020) สำหรับการใช้งาน VXLAN.
- กำหนด interfaces Ethernet 1/3 และ 1/4 ให้เพิ่มโง่กับ VLAN 10 และ VLAN 20 ตามลำดับในโหมด Access. การตั้งค่านี้จะมีผลสำหรับการใช้งานในศูนย์ข้อมูลหรือเครือข่ายที่ต้องการสร้างเครือข่ายเสมือนหลายๆ เครือข่ายบนเครือข่าย IP เดียวทั้งนั้น.

ODL BGPS



Topo.py

```
#!/usr/bin/python
from containernet.cli import CLI
from containernet.link import TCLink
from containernet.net import Containernet
from mininet.node import Controller
from mininet.log import info, setLogLevel

# topo diagram
# h1----r1---s3----r2-----h2
#           |
#           s4-----r3-----h3

def topology():

    "Create a network with some docker containers acting as hosts."

    net = Containernet(controller=Controller)

    info('*** Adding switch\n')
    r1 = net.addDocker('r1', ip='172.30.1.1/24', dimage="knet/urouter:1.4")
    r2 = net.addDocker('r2', ip='172.30.2.1/24', dimage="knet/urouter:1.4")
    r3 = net.addDocker('r3', ip='172.30.3.1/24', dimage="knet/urouter:1.4")
    h1 = net.addDocker('h1', ip='172.30.1.2/24', defaultRoute='via 172.30.1.1', dimage="knet/host-ubuntu:1-2")
    h2 = net.addDocker('h2', ip='172.30.2.2/24', defaultRoute='via 172.30.2.1', dimage="knet/host-ubuntu:1-2")
    h3 = net.addDocker('h3', ip='172.30.3.2/24', defaultRoute='via 172.30.3.1', dimage="knet/host-ubuntu:1-2")

    s3 = net.addSwitch('s3', failMode='standalone')
    s4 = net.addSwitch('s4', failMode='standalone')

    info('*** Creating links\n')

    net.addLink(h1, r1)
    net.addLink(h2, r2)
    net.addLink(h3, r3)

    net.addLink(r1, s3, params1={"ip": "10.10.10.1/24"})
    net.addLink(r2, s3, params1={"ip": "10.10.10.2/24"})
    net.addLink(r2, s4, params1={"ip": "10.20.20.1/24"})
    net.addLink(r3, s4, params1={"ip": "10.20.20.2/24"})

    info('*** Starting network\n')
    net.start()

    #copy the bird config files
    s3.cmd("sudo docker cp r1.conf mn.r1:/etc/bird.conf")
    s3.cmd("sudo docker cp r2.conf mn.r2:/etc/bird.conf")
    s4.cmd("sudo docker cp r2.conf mn.r2:/etc/bird.conf")
    s4.cmd("sudo docker cp r3.conf mn.r3:/etc/bird.conf")
    r1.cmd("bird -c /etc/bird.conf")
    r2.cmd("bird -c /etc/bird.conf")
    r3.cmd("bird -c /etc/bird.conf")

    info('*** Running CLI\n')
    CLI(net)
    info('*** Stopping network')
    net.stop()

if __name__ == '__main__':
    setLogLevel('info')
    topology()
```

Bgp_neighbor.xml

```
bgp_neighbor.xml
1 <neighbor xmlns="urn:opendaylight:params:xml:ns:yang:bgp:openconfig-extensions">
2   <neighbor-address>172.17.0.2</neighbor-address>
3   <timers>
4     <config>
5       <hold-time>90</hold-time>
6       <connect-retry>10</connect-retry>
7     </config>
8   </timers>
9   <transport>
10    <config>
11      <remote-port>179</remote-port>
12      <passive-mode>false</passive-mode>
13    </config>
14  </transport>
15  <config>
16    <peer-type>EXTERNAL</peer-type>
17    <peer-as>64511</peer-as>
18  </config>
19  <afi-safis>
20    <afi-safi>
21      <afi-safi-name xmlns:x="http://openconfig.net/yang/bgp-types">x:IPV4-UNICAST</afi-safi-name>
22    </afi-safi>
23    <afi-safi>
24      <afi-safi-name xmlns:x="http://openconfig.net/yang/bgp-types">x:IPV6-UNICAST</afi-safi-name>
25    </afi-safi>
26  </afi-safis>
27 </neighbor>
```

Bgp_router.xml

```
bgp_router.xml
1 <protocol xmlns="http://openconfig.net/yang/network-instance">
2   <name>bgp-odl-router</name>
3   <identifier xmlns:x="http://openconfig.net/yang/policy-types">x:BGP</identifier>
4   <bgp xmlns="urn:opendaylight:params:xml:ns:yang:bgp:openconfig-extensions">
5     <global>
6       <config>
7         <router-id>172.17.0.1</router-id>
8         <as>64513</as>
9       </config>
10      <afi-safis>
11        <afi-safi>
12          <afi-safi-name xmlns:x="http://openconfig.net/yang/bgp-types">x:IPV4-UNICAST</afi-safi-name>
13        </afi-safi>
14        <afi-safi>
15          <afi-safi-name xmlns:x="http://openconfig.net/yang/bgp-types">x:IPV6-UNICAST</afi-safi-name>
16        </afi-safi>
17        </afi-safis>
18      </global>
19    </bgp>
20 </protocol>
```

R1.conf

```
r1.conf
1 log "/var/log/bird.log" all;
2 debug protocols all
3 router id 10.10.10.1;
4 protocol direct {
5     interface "*";
6 }
7 protocol kernel {
8     learn;
9     scan time 20;
10    export all;
11    import all;
12 }
13 protocol device {
14     scan time 10;
15 }
16 #BGP Configuration
17 protocol bgp R2{
18     export all;
19     import all;
20     local as 64511;
21     neighbor 10.10.10.2 as 64512;
22 }
23 protocol bgp OD1{
24     export all;
25     import all;
26     local as 64511;
27     neighbor 172.17.0.1 as 64513;
28 }
```

R2.conf

```
r2.conf
1 log "/var/log/bird.log" all;
2 debug protocols all
3 router id 10.10.10.2;
4 protocol direct {
5     interface "*";
6 }
7 protocol kernel {
8     learn;
9     scan time 20;
10    export all;
11    import all;
12 }
13 protocol device {
14     scan time 10;
15 }
16 #BGP Configuration
17 protocol bgp R1{
18     export all;
19     import all;
20     local as 64512;
21     neighbor 10.10.10.1 as 64511;
22 }
23 protocol bgp R3{
24     export all;
25     import all;
26     local as 64512;
27     neighbor 10.20.20.2 as 64514;
28 }
```

R3.conf

```
r3.conf
1 log "/var/log/bird.log" all;
2 debug protocols all
3 router id 10.20.20.2;
4 protocol direct {
5     interface "*";
6 }
7 protocol kernel {
8     learn;
9     scan time 20;
10    export all;
11    import all;
12 }
13 protocol device {
14     scan time 10;
15 }
16 #BGP Configuration
17 protocol bgp R1{
18     export all;
19     import all;
20     local as 64514;
21     neighbor 10.20.20.1 as 64512;
22 }
23 protocol bgp R2{
24     export all;
25     import all;
26     local as 64514;
27     neighbor 10.20.20.1 as 64512;
28 }
```

```
sudo python3 topo1.py
```

R1

```
containernet>r1 birdc show protocols
```

```
containernet>r1 ip route
```

R2

```
containernet>r2 birdc show protocols
```

```
containernet>r2 ip route
```

Create the ODL BGP Instance

```
curl -v --user "admin":"admin" -H "Accept: application/xml" -H "Content-Type: application/xml" -X POST http://localhost:8181/restconf/config/openconfig-network-instance:network-instances/network-instance/global-bgp/openconfig-network-instance:protocols/ -d @bgp_router.xml
```

verification

```
curl -v --user "admin":"admin" -H "Accept: application/xml" -H "Content-Type: application/xml" -X GET http://localhost:8181/restconf/operational/bgp-rib:bgp-rib/ | xmllint --format -
```

create bgp neighbor

```
curl -v --user "admin":"admin" -H "Accept: application/xml" -H "Content-Type: application/xml" -X POST http://localhost:8181/restconf/config/openconfig-network-instance:network-instances/network-instance/global-bgp/openconfig-network-instance:protocols/protocol/openconfig-policy-types:BGP/bgp-odl-router/bgp/neighbors/ -d @bgp_neighbor.xml
```

verify bgp neighbor/route details

```
curl -v --user "admin":"admin" -H "Accept: application/xml" -H "Content-Type: application/xml" -X GET http://localhost:8181/restconf/operational/bgp-rib:bgp-rib/bgp-odl-router/ | xmllint --format -
```

verify

R1

```
containernet>r1 birdc show protocols
```

```
containernet>r1 ip route
```

R2

```
containernet>r2 birdc show protocols
```

```
containernet>r2 ip route
```

Destroy the test อันนี้ไม่ต้องทำมันคือลบข้อมูลทั้งหมดที่ทำไป

```
curl -v --user "admin":"admin" -H "Accept: application/xml" -H "Content-Type: application/xml" -X DELETE http://localhost:8181/restconf/config/openconfig-network-instance:network-instances/network-instance/global-bgp/openconfig-network-instance:protocols/ -d @bgp_router.xml
```

Add VXLAN PORT

```
{  
    "network-topology:termination-point": [  
        {  
            "ovsdb:options": [  
                {  
                    "ovsdb:option": "remote_ip",  
                    "ovsdb:value" : "10.10.14.11"  
                }  
            ],  
            "ovsdb:name": "vxlanport1",  
            "ovsdb:interface-type": "ovsdb:interface-type-vxlan",  
            "tp-id": "vxlanport1",  
            "vlan-tag": "1",  
            "trunks": [  
                {  
                    "trunk": "5"  
                }  
            ],  
            "vlan-mode": "access"  
        }  
    ]  
}
```

```
curl --user "admin":"admin" -H "Accept: application/json" -H "Content-Type: application/json" -X PUT  
http://localhost:8181/restconf/config/network-topology:network-  
topology/topology/ovsdb:1/node/ovsdb:HOST1%2Fbridge%2Fs1/termination-point/vxlanport1 -d  
@vxlanport1.json
```

curl -L -o Bigkuma.zip

https://github.com/thanormsaksudsee/SDN_Final/archive/refs/heads/main.zip

ลิ้ง Vxlan

<https://www.csepracticals.com/blog/bgp-evpn-vxlan-lab-part1>