

CUMULUS LINUX TRAINING: LAB GUIDE

Scope

This workbook covers configuration of network protocols on Cumulus Linux Network Operating System.

Audience

This workbook is intended for Technical Training students.

Objectives

By the end of this workbook, students will be able to:

- Configure basic switch functions with Cumulus Linux
- Configure layer 2 and layer 3 protocols with Cumulus Linux
- Verify configuration and connectivity
- Monitor and troubleshoot networking related connectivity issues

Overview

Each student will be using the Cumulus Air © platform, exercises in this workbook on a group of devices (servers and switches).

Notice

Please follow the instructions below carefully to successfully complete the practice.

If you encounter technical issues, please contact the Nvidia Networking Academy team: nbu-academy-support@nvidia.com

Release Date

Revision 1.6 – September 2020

Good Luck,

Nvidia Networking Academy team

TABLE OF CONTENTS

Prerequisites and Guidelines	3
Nvidia Networking Academy Lab Servers and Switches	4
 PRACTICE UNITS	
Practice 1: Using the NCLU	6
Practice 2: Basic Switch Functions	10
Practice 3: VLANs and Trunking	16
Practice 4: Configuring MLAG and VRR	21
Practice 5: Configuring BGP Unnumbered	35
Practice 6: Configuring VXLAN with EVPN	43
Practice 7: Configuring Distributed Asymmetric VXLAN Routing	47

PREREQUISITES AND GUIDELINES

Please perform and review the following steps before you start:

1. Enter the Cumulus Air web page, and click the “GET STARTED” button.
<https://air.cumulusnetworks.com/Login>
 If you are already signed in, use your credentials to login.
 to sign up for the first time, click “[Register](#)” and fill out your details.
 Once completed, a confirmation email will be sent, open it to activate your new account.
2. Once you are logged in, you will reach the “Cumulus In The Cloud” dashboard.
 Click on the “Academy ILT” label, and wait for the switches and servers to reload.
3. Wait for the devices to load and follow the instructions in this lab manual.

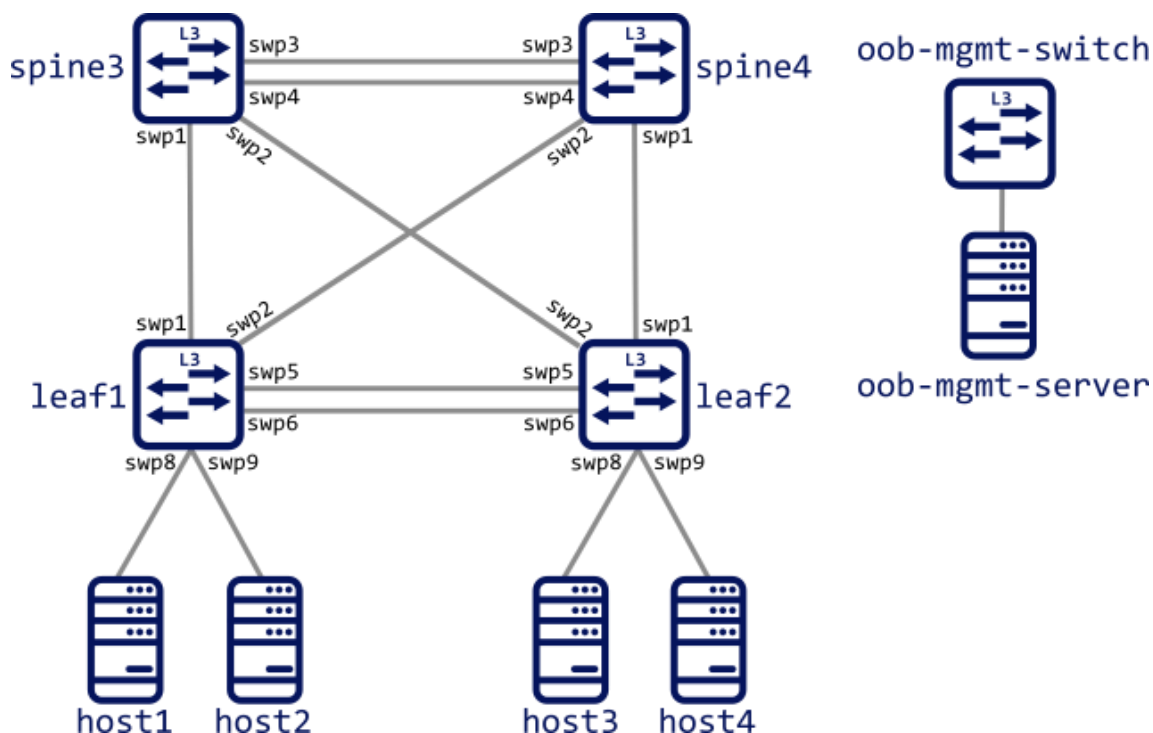
Academy ILT

LAUNCH

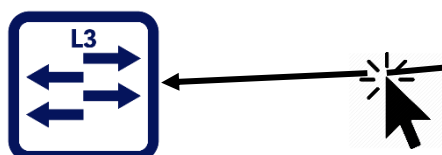
✓ Loaded

CUMULUS AIR SERVERS AND SWITCHES ACCESS

- The training Lab is organized in the following topology:

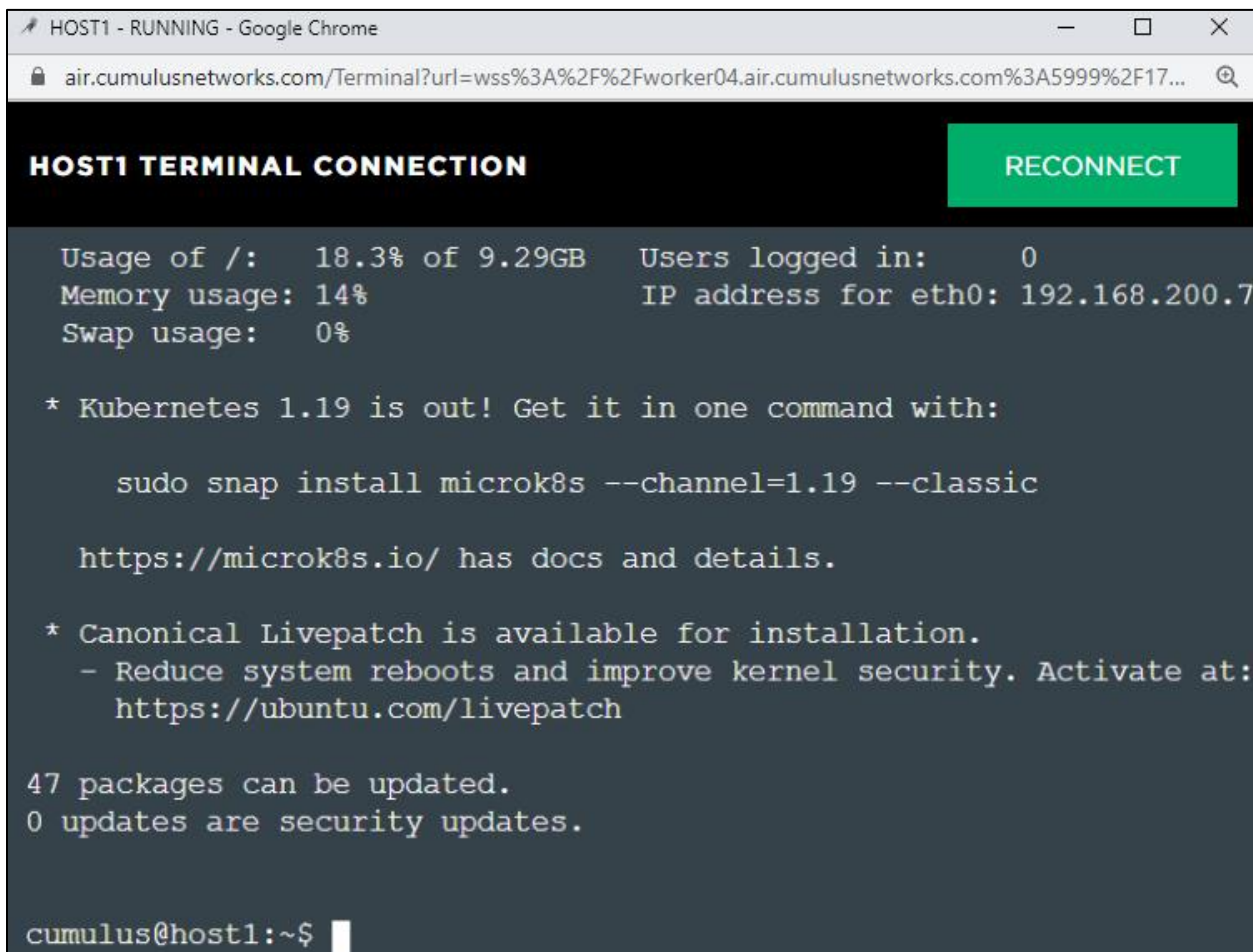


- To log in to a device, first make sure it's running, then connect to it by either clicking its name on the NODES panel or it's icon on the diagram



oob-mgmt-server Status: RUNNING	CPU: 1 Memory: 1024 MB	Storage: 10 GB ✓ Actions
oob-mgmt-switch Status: RUNNING	CPU: 1 Memory: 1024 MB	Storage: 10 GB ✓ Actions
host2 Status: RUNNING	CPU: 1 Memory: 1024 MB	Storage: 10 GB ✓ Actions
leaf2 Status: RUNNING	CPU: 1 Memory: 1024 MB	Storage: 10 GB ✓ Actions
host3 Status: RUNNING	CPU: 1 Memory: 1024 MB	Storage: 10 GB ✓ Actions
host1 Status: RUNNING	CPU: 1 Memory: 1024 MB	Storage: 10 GB ✓ Actions
spine3 Status: RUNNING	CPU: 1 Memory: 1024 MB	Storage: 10 GB ✓ Actions
leaf1 Status: RUNNING	CPU: 1 Memory: 1024 MB	Storage: 10 GB ✓ Actions
host4 Status: RUNNING	CPU: 1 Memory: 1024 MB	Storage: 10 GB ✓ Actions
spine4 Status: RUNNING	CPU: 1 Memory: 1024 MB	Storage: 10 GB ✓ Actions

3. When the login prompt appears, enter the default Cumulus Linux username –
“**cumulus**”
4. When the password prompt appears, enter the default Cumulus Linux password – “**Academy123**” and press Enter.
5. You should now be prompted with the server name. This indicates that you have successfully accessed the server/switch.



```

HOST1 - RUNNING - Google Chrome
air.cumulusnetworks.com/Terminal?url=wss%3A%2F%2Fworker04.air.cumulusnetworks.com%3A5999%2F17...

HOST1 TERMINAL CONNECTION RECONNECT

Usage of /: 18.3% of 9.29GB   Users logged in: 0
Memory usage: 14%           IP address for eth0: 192.168.200.7
Swap usage: 0%

* Kubernetes 1.19 is out! Get it in one command with:

    sudo snap install microk8s --channel=1.19 --classic

https://microk8s.io/ has docs and details.

* Canonical Livepatch is available for installation.
  - Reduce system reboots and improve kernel security. Activate at:
    https://ubuntu.com/livepatch

47 packages can be updated.
0 updates are security updates.

cumulus@host1:~$

```

PRACTICE 1: USING THE NCLU

Practice objectives:

In this practice session you will become familiar with the Cumulus Linux NCLU

- You will use '**net add**' and '**net del**' commands to change the configuration
- You will use '**net commit**' command to apply configuration changes
- You will use '**net show**' commands to validate the configuration
- Last, you will use '**net rollback**' command to roll back to a previous configuration

Task 1: Retrieve system information

a. Identify the switch platform and image:

h **system**

```
cumulus@leaf1:~$ net show system
Hostname..... leaf1
Build..... Cumulus Linux 3.7.12
Uptime..... 12:53:54.640000

Model..... Cumulus VX
Memory..... 929MBVendor
Name..... Cumulus Networks
Part Number..... 3.7.12
Base MAC Address. 44:38:39:00:00:1E
Serial Number.... 44:38:39:00:00:1e
Product Name..... VX
```

Task 2: Change the configuration with NCLU

- a. Access the switch that was assigned to you and reset configuration:

```
# net del all
# net commit
```

```
cumulus@leaf1:~$ net del all
cumulus@leaf1:~$ net commit
```

- b. Bring up the switch ports:

```
# net add interface swp1-16
```

```
cumulus@leaf1:~$ net add interface swp1-16
cumulus@leaf1:~$ net commit
```

- c. Add an alias to each of the interfaces which are part of the lab topology. The alias should describe the device which is connected to the interface.

For example, on switch leaf1 named leaf1:

- Interface swp8 is connected to server host1
- Interface swp1 is connected to switch spine3

```
# net add interface <INTERFACE> alias <TEXT>
```

```
cumulus@leaf1:~$ net add interface swp8 alias Connected to
host1:Eth2
```

```
cumulus@leaf1:~$ net add interface swp1 alias Connected to
spine3:swp1
```

- d. View the changes in the commit buffer:

```
# net pending
```

```
cumulus@leaf1:~$ net pending
--- /etc/network/interfaces      2018-03-25 12:07:38.286992779 +0000
+++ /var/run/nclu/iface/interfaces.tmp 2018-03-25 13:37:54.614290080 +0000
@@ -6,35 +6,37 @@

<output omitted>

auto swp1
iface swp1
+   alias Connected to spine3:swp1

<output omitted>
```

- e. Commit the changes with a custom description:

```
# net commit description <TEXT>
```

```
cumulus@leaf1:~$ net commit description Practice-1

<output omitted>

auto swp1
iface swp1
+   alias Connected to spine3b:swp1

<output omitted>

net add/del commands since the last 'net commit'
=====
```

User	Timestamp	Command
cumulus	2018-03-25 13:36:26.822328	net add interface swp8 alias Connected to host1:Eth2
cumulus	2018-03-25 13:36:42.782146	net add interface swp1 alias Connected to spine3:swp1
cumulus	2018-03-25 14:27:52.833075	net commit description Practice-1

Task 3: Rollback the configuration

- a. View the NCLU commit history:

```
# net show commit history
```

```
cumulus@leaf1:~$ net show commit history
#   Date                               Description
---  -
166 Mon 02 Apr 2018 08:11:40 AM UTC  nclu 'net commit' (user cumulus)
168 Mon 02 Apr 2018 08:13:38 AM UTC  nclu 'net commit' (user cumulus)
170 Mon 02 Apr 2018 08:14:26 AM UTC  nclu Practice-1
```

- b. Rollback to the last commit:

```
# net rollback last
```

```
cumulus@leaf1:~$ net rollback last
```

- c. Verify rollback was applied successfully:

```
# net show configuration
```

```
cumulus@leaf1:~$ net show configuration
<output omitted>
interface swp1
interface swp8
```

**** Please note:**

- At this point all configuration is deleted because the configuration has been reverted to a point before any configuration existed.
- Alternatively, you can roll back to any commit by referencing the unique commit number or description.

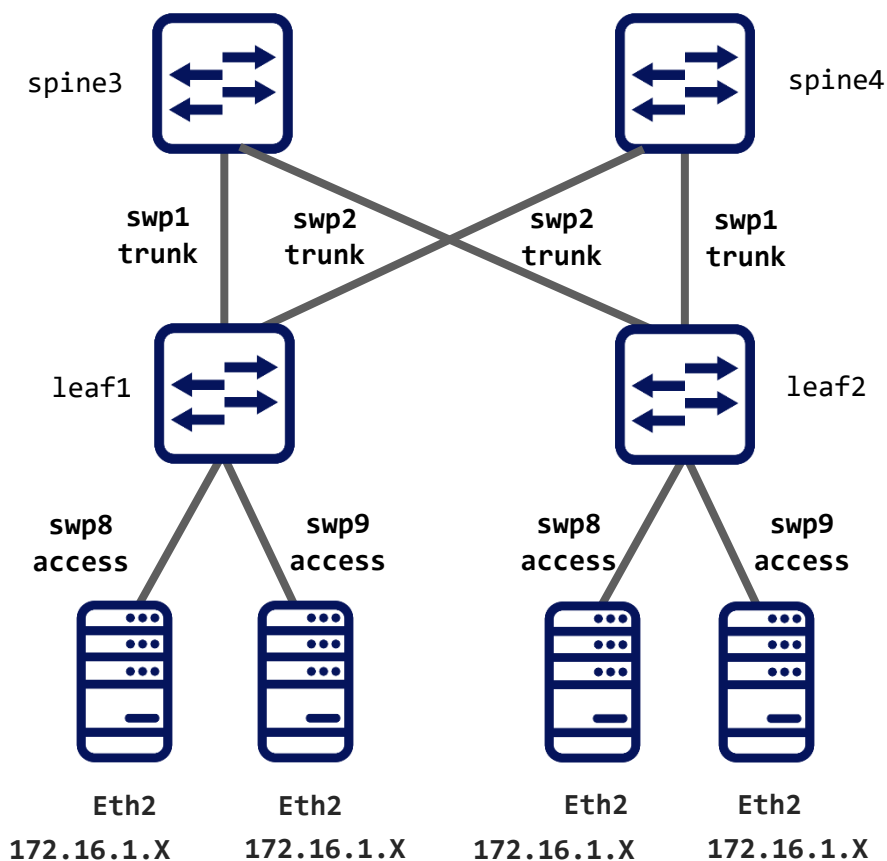
PRACTICE 2: BASIC SWITCH FUNCTIONS

Practice objectives:

In this practice session you will verify IP connectivity between servers in the lab.

- You will configure the servers IP settings – an IP address, a subnet mask and a default gateway.
- You will configure a bridge on each of the Cumulus Linux switches in your group, and add switch ports to the bridge.
- You will use **'ping'** utility to verify communication between servers in your group.
- Last, you will observe how the switch forwarding database – the MAC address table – is built and maintained.

Topology used in this practice:



Task 1: Configure servers hostname and IP settings

- a. Access the server that was assigned to you and check IP settings of interface 'eth2':

ifconfig <dev>

```
cumulus@ubuntu:~$ ifconfig eth2
eth2: flags=4098<BROADCAST,MULTICAST> mtu 1500
    ether 44:38:39:00:00:11 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

- b. Configure server hostname, IP address and a subnet mask for interface 'eth2'.

If the interface's IP address is not as listed in the table below, use the following table for IP address assignment. Use /24 as the subnet mask.

Practice Lab Servers Properties

Server	'eth2' IP Address
host1	172.16.1.1
host2	172.16.1.2
host3	172.16.1.3
host4	172.16.1.4

- Clear existing IP configuration:

ifconfig <dev> 0.0.0.0

- Configure an IP address and a subnet mask:

ifconfig <dev> IP/MASK

```
cumulus@host1:~$ sudo ifconfig eth2 0.0.0.0
cumulus@host1:~$ sudo ifconfig eth2 172.16.1.1/24
cumulus@host1:~$ ifconfig eth2
eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.16.1.1 netmask 255.255.255.0 broadcast 172.16.1.255
    inet6 fe80::4638:39ff:fe00:11 prefixlen 64 scopeid 0x20<link>
    ether 44:38:39:00:00:11 txqueuelen 1000 (Ethernet)
    RX packets 3 bytes 180 (180.0 B)
```

- c. Verify a static route entry to 172.16.0.0/16 via interface 'eth2':

route

```
cumulus@host1:~$ route
Kernel IP routing table
Destination      Gateway          Genmask          Flags Metric Ref    Use
Iface
default          _gateway        0.0.0.0          UG    0      0      0
eth0
192.168.200.0    0.0.0.0         255.255.255.0    U     0      0      0
eth0
```

- If the static route is missing, add it:

ip route add <ADDRESS/MASK> dev <DEV>

```
cumulus@host1:~$ sudo ip route add 172.16.0.0/16 dev eth2
```

```
cumulus@host1:~$ route
Kernel IP routing table
Destination      Gateway          Genmask          Flags Metric Ref    Use Iface
default          _gateway        0.0.0.0          UG    0      0      0 eth0
172.16.0.0       0.0.0.0         255.255.0.0      U     0      0      0 eth2
172.16.1.0       0.0.0.0         255.255.255.0    U     0      0      0 eth2
192.168.200.0    0.0.0.0         255.255.255.0    U     0      0      0 eth0
```

Task 2: Configure a bridge

- f. Access the switch that was assigned to you and reset configuration:

```
# net del all
# net commit
```

```
cumulus@leaf1:~$ net del all
cumulus@leaf1:~$ net commit
```

- a. Create a bridge and set the inter switch links (swp1 and swp2) as trunk ports:

```
# net add bridge bridge ports <PORTS>
```

```
cumulus@leaf1:~$ net add bridge bridge ports swp1-2
```

- b. Set the host-facing ports, swp8 and swp9 on the leaf switches, as access ports in VLAN 1:

```
# net add interface <PORTS> bridge access <VLAN-ID>
```

```
cumulus@leaf1:~$ net add bridge bridge ports swp8-9
cumulus@leaf1:~$ net add interface swp8-9 bridge access 1
```

- c. Commit changes:

```
net commit
```

```
cumulus@leaf1:~$ net commit
```

- d. Verify configuration:

```
net show configuration
```

```
cumulus@leaf1:~$ net show configuration
```

```
interface swp8
  bridge-access 1
```

```
interface swp9
  bridge-access 1
```

```
interface bridge
  bridge-ports swp1 swp2 swp8 swp9
  bridge-vids 1
  bridge-vlan-aware yes
```

- e. Verify interfaces status:
net show interface

```
cumulus@leaf1:~$ net show interface
```

Name	Master	Speed	MTU	Mode	Remote Host	Remote Port	Summary
-----	-----	-----	-----	-----	-----	-----	-----
<output omitted>							
UP swp1	bridge	100G	1500	Access/L2	spine4	swp1	
UP swp2	bridge	100G	1500	Access/L2	spine3	swp2	
<output omitted>							
UP swp8	bridge	100G	1500	Access/L2	host1	ec:0d:9a:46:9e:b5	
UP swp9	bridge	100G	1500	Access/L2			
UP bridge		N/A	1500	Bridge/L2			

**** Please note:**

- Even though swp1-2 were configured in 'Trunk' mode, they are displayed in the output in 'Access' mode. The reason is that currently only VLAN 1 is configured and no frames are sent tagged over those ports.
Once you configure additional VLANs, those ports will be displayed in 'Trunk' mode indicating they are tagging frames.

Task 3: Observe a switch's forwarding database

- a. Identify the MAC addresses of all four servers in your group.

```
cumulus@host1:~$ ifconfig eth2
eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.16.1.1 netmask 255.255.255.0 broadcast 172.16.1.255
    inet6 fe80::4638:39ff:fe00:11 prefixlen 64 scopeid 0x20<link>
    ether 44:38:39:00:00:11 txqueuelen 1000 (Ethernet)
    RX packets 3 bytes 180 (180.0 B)
    RX errors 0 dropped 3 overruns 0 frame 0
    TX packets 18 bytes 2371 (2.3 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Fill in the table with the MAC addresses of 'eth2' interfaces of the servers in your group:

Server	MAC address of interface 'eth2'

- b. Use 'ping' from one server to another server in the lab.
For example ping from server **host1** to server **host3**.

```
cumulus@host1:~$ ping 172.16.1.3
PING 172.16.1.6 (172.16.1.6) 56(84) bytes of data.
64 bytes from 172.16.1.6: icmp_seq=1 ttl=64 time=0.318 ms
64 bytes from 172.16.1.6: icmp_seq=2 ttl=64 time=0.111 ms
```

- c. Display the switch's MAC address table. Identify on which switch ports the servers' MAC addresses were learned?

net show bridge macs

```
cumulus@leaf2:~$ net show bridge macs
```

VLAN	Master	Interface	MAC	TunnelDest	State	Flags	LastSeen
untagged	bridge	swp8	00:8a:07:cf:6a:c4				00:14:02
untagged	bridge	swp9	00:8a:07:cf:6a:e4				00:14:02

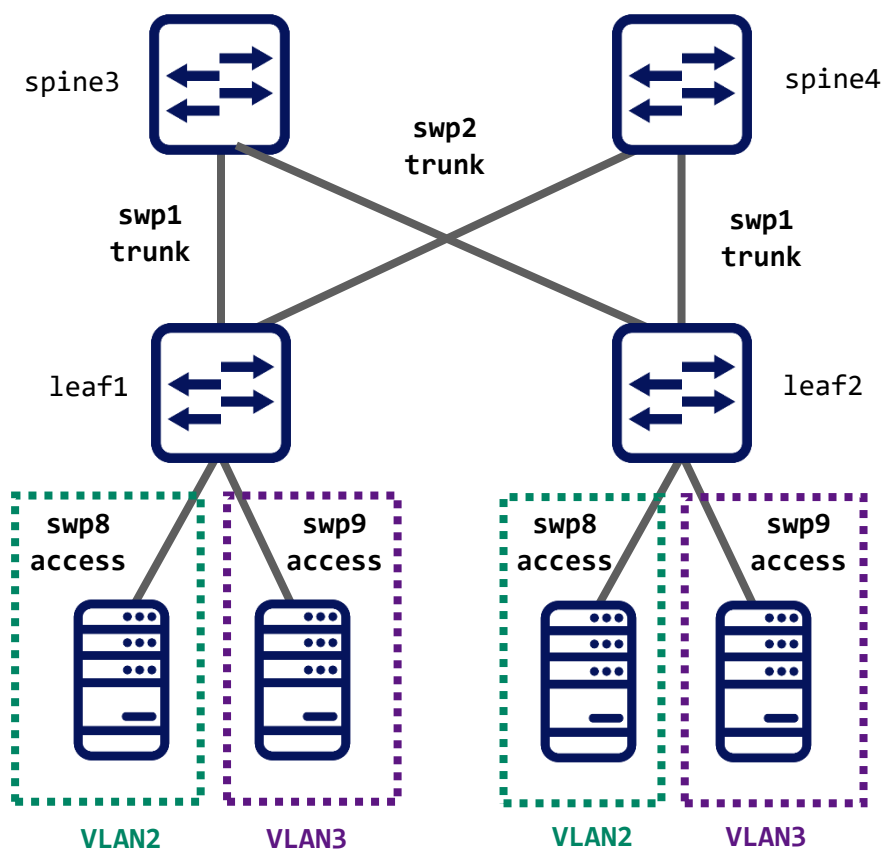
PRACTICE 3: VLANs AND TRUNKING

Practice objectives:

In this practice session you will configure and verify VLANs and trunking:

- You will configure two new VLANs and assign switch ports connected to servers to the configured VLANs.
- You will configure SVIs to allow inter-VLAN communication.

Topology used in this lab:



Task 1: Configuring VLANs and trunking

- a. Access the switch that was assigned to you and reset configuration:

```
# net del all
# net commit
```

```
cumulus@leaf1:~$ net del all
cumulus@leaf1:~$ net commit
```

- b. Create a bridge and set the inter switch links, **swp1** and **swp2**, as trunk ports:

```
# net add bridge bridge ports <PORTS>
```

```
cumulus@leaf1:~$ net add bridge bridge ports swp1-2
```

- c. Add VLANs 2-3 to the bridge:

```
cumulus@leaf1:~$ net add bridge bridge vids 2,3
```

- d. Set the host-facing ports, swp8 and swp9 on the leaf switches, as access ports and associates them to the appropriate VLAN: interface **swp8** in **VLAN2** and interface **swp9** in **VLAN3**

```
# net add interface <PORTS> bridge access <VLAN-ID>
```

```
cumulus@leaf1:~$ net add bridge bridge ports swp8-9
cumulus@leaf1:~$ net add interface swp8 bridge access 2
cumulus@leaf1:~$ net add interface swp9 bridge access 3
```

- e. Commit changes:

```
cumulus@leaf1:~$ net commit
```

- f. Verify configuration:

```
cumulus@leaf1:~$ net show configuration
```

```
interface swp8
  bridge-access 2
```

```
interface swp9
  bridge-access 3
```

```
interface bridge
  bridge-ports swp1 swp2 swp8 swp9
  bridge-vids 2-3
  bridge-vlan-aware yes
```

g. Verify VLANs configuration:

net show bridge vlan

```
cumulus@leaf1:~$ net show bridge vlan
```

Interface	VLAN	Flags
-----	-----	-----
swp1	1	PVID, Egress Untagged
	2-3	
swp2	1	PVID, Egress Untagged
	2-3	
swp8	2	PVID, Egress Untagged
swp9	3	PVID, Egress Untagged

**** Please note:**

- Access ports are shown with a single line representing the VLAN associated to the port
- Trunk ports are shown with multiple lines representing the VLANs associated with the trunk port

Task 2: Servers' IP settings

Access the server that was assigned to you, and configure an IP address for interface '**eth2**' according your group's subnet (see tables below).

Servers in the same VLAN will be configured with IP addresses in the same subnet, hence they will be able to communicate over the layer 2 network.

VLAN ID	Server	'eth2' IP Address
VLAN 2	host1	172.16.2.18/24
VLAN 3	host2	172.16.3.19/24
VLAN 2	host3	172.16.2.28/24
VLAN 3	host4	172.16.3.29/24

a. Configure the server's IP address and subnet mask:

- Clear existing IP configuration:
ifconfig eth2 0.0.0.0
- Configure an IP address and a subnet mask:
ifconfig eth2 IP_ADDRESS/SUBNET_MASK
- Verify IP configuration:

ifconfig eth2

```
cumulus@host1:~$ sudo ifconfig eth2 0.0.0.0
cumulus@host1:~$ sudo ifconfig eth2 172.16.2.18/24
cumulus@host1:~$ ifconfig eth2
eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.16.2.18 netmask 255.255.255.0 broadcast 172.16.2.255
    inet6 fe80::4638:39ff:fe00:11 prefixlen 64 scopeid 0x20<link>
    ether 44:38:39:00:00:11 txqueuelen 1000 (Ethernet)
    RX packets 3 bytes 180 (180.0 B)
    RX errors 0 dropped 3 overruns 0 frame 0
    TX packets 43 bytes 7456 (7.4 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

- b. Verify a static route entry to network 172.16.0.0/16 via the default gateway's address. If the entry is missing, add it.

** Please note:

- Make sure to use the default gateway address in the following table

VLAN ID	Default gateway address
vlan 2	172.16.2.254/24
vlan 3	172.16.3.254/24

ip route add <NET_ADDRESS/SUBNET_MASK> dev <DEV> via IP_ADDRESS

```
cumulus@host1:~$ sudo ip route add 172.16.0.0/16 dev eth2 via 172.16.2.254
cumulus@host1:~$ route
Kernel IP routing table
Destination        Gateway            Genmask           Flags Metric Ref    Use Iface
default            _gateway          0.0.0.0           UG    0      0      0 eth0
172.16.0.0         172.16.2.254     255.255.0.0       UG    0      0      0 eth2
172.16.2.0         0.0.0.0          255.255.255.0     U      0      0      0 eth2
192.168.200.0      0.0.0.0          255.255.255.0     U      0      0      0 eth0
```

- c. Use 'ping' to check communication between servers in the same VLAN.
For example server **host1** and **host3**.

```
[cumulus@host1 ~]# ping 172.16.2.28
PING 172.16.2.28 (172.16.2.28) 56(84) bytes of data.
64 bytes from 172.16.2.28: icmp_seq=1 ttl=64 time=0.147 ms
64 bytes from 172.16.2.28: icmp_seq=2 ttl=64 time=0.135 ms
```

Task 3: Configuring SVIs for inter-VLAN routing

- a. On switch **spine3** configure two SVIs (Switch VLAN Interfaces) that will be used for routing between VLAN2 and VLAN3:
 - Interface **vlan2** will serve as the default gateway for **VLAN2**
 - Interface **vlan3** will serve as the default gateway for **VLAN3**

Use the following table for IP address assignment:

Interface vlan 2	172.16.2.254/24
Interface vlan 3	172.16.3.254/24

net add vlan 2 ip address 172.16.2.254/24

```
[cumulus@spine3:~$ net add vlan 2 ip address 172.16.2.254/24
[cumulus@spine3:~$ net add vlan 3 ip address 172.16.3.254/24
[cumulus@spine3:~$ net commit
```

- b. Use **'ping'** and **'traceroute'** utilities to verify communication between hosts in different VLANs.

```
[cumulus@host1 ~]# ping 172.16.3.19
PING 172.16.3.19 (172.16.3.19) 56(84) bytes of data.
64 bytes from 172.16.3.19: icmp_seq=1 ttl=63 time=0.155 ms
```

```
[cumulus@host1 ~]# traceroute 172.16.3.19
traceroute to 172.16.23.19 (172.16.3.19), 30 hops max, 60 byte packets
 1  172.16.2.254 (172.16.2.254)  0.387 ms  0.394 ms  0.535 ms
```

For example, ping from server **'host1'** (in VLAN 2) to server **'host2'** (in VLAN3).

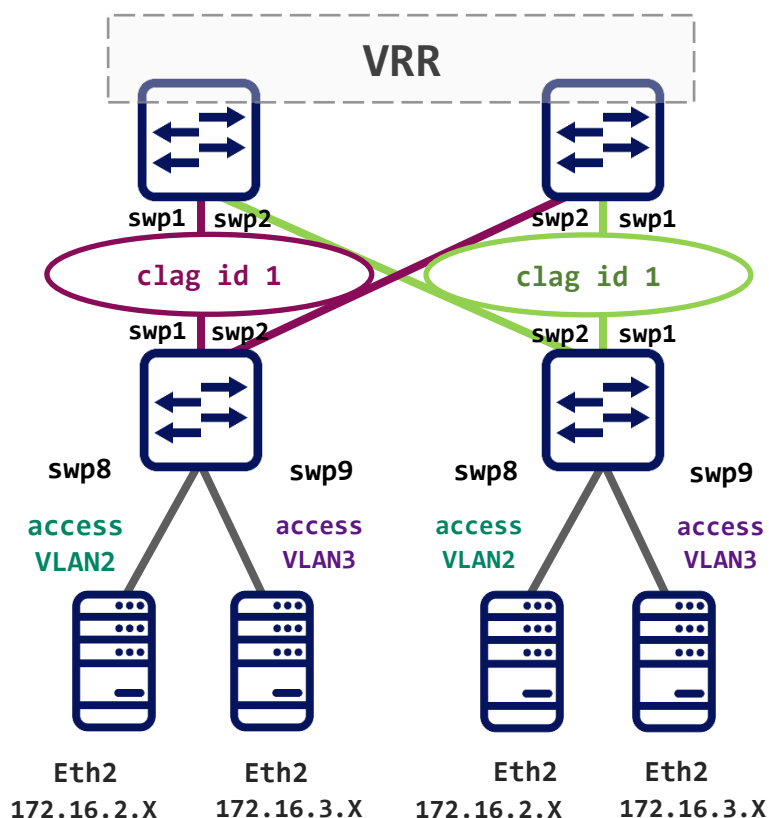
PRACTICE 4: CONFIGURING MLAG AND VRR

Practice objectives:

In this practice session you will configure the spine switches, spine3 and spine4, with MLAG and VRR towards the leaf switches.

- MLAG will make the spine switches to look and behave like a single Layer 2 switch towards the Layer 2 network.
- VRR will make the spine switches to look and behave like a single router providing default gateway redundancy.
- Interface 'clag 1' will aggregate swp1 on spine3 and swp2 on spine4 connected to switch leaf1.
Switch leaf1 will be configured with a regular LAG.
- Interface 'clag 2' will aggregate swp2 on spine3 and swp1 on spine4 connected to switch leaf2.
Switch leaf2 will be configured with a regular LAG.

Topology used in this lab:



Task 1: LAG configuration – leaf switches

- a. Access the switch that was assigned to you and reset configuration:

```
# net del all
# net commit
```

```
cumulus@leaf1:~$ net del all
cumulus@leaf1:~$ net commit
```

- b. On the leaf switches **leaf1** and **leaf2**: create a bond named '**BOND-TO-SPINES**', where bonds slaves are interfaces **swp1** and **swp2**.

```
# net add bond <BOND-NAME> bond slaves <INTERFACES>
```

```
cumulus@leaf1:~$ net add bond BOND-TO-SPINES bond slaves swp1-2
```

- c. Verify bonds configuration:

```
cumulus@leaf1:~$ net show interface bonds
```

	Name	Speed	MTU	Mode	Summary
DN	BOND-TO-SPINES	N/A	1500	802.3ad	Bond Members: swp1(UP), swp2(UP)

**** Please note:**

- The bond interface is down because its peer (the MLAG switches) was not configured yet.

- d. Create a bridge and add the bond interface, **BOND-TO-SPINES**, and the host facing interfaces, **swp8** and **swp9**, to the bridge:

```
cumulus@leaf1:~$ net add bridge bridge ports swp8-9,BOND-TO-SPINES
```

- e. Configure VLANs 2-3 and associate **swp8** to **VLAN2** and **swp9** to **VLAN3**:

```
cumulus@leaf1:~$ net add
cumulus@leaf1:~$ net add interface swp8 bridge access 2
cumulus@leaf1:~$ net add interface swp9 bridge access 3
cumulus@leaf1:~$ net commit
```

**** Please note:**

- Switch **leaf2** should be configured similarly.

Task 2: Configuring MLAG – spine switches

- Access the switch that was assigned to you and reset configuration:

```
# net del all
# net commit
```

```
cumulus@spine3:~$ net del all
cumulus@spine3:~$ net commit
```

- Write down, on a side note, the IP addresses on the switches management ports, **eth0**. Those IP addresses will be configured as the MLAG backup-IPs.

```
cumulus@spine3:~$ net show interface eth0
```

Name	MAC	Speed	MTU	Mode
UP eth0	ec:0d:9a:38:dd:72	1G	1500	Mgmt

```
IP Details
-----
IP: 10.143.33.187/24
```

```
cumulus@spine4:~$ net show interface eth0
```

Name	MAC	Speed	MTU	Mode
UP eth0	ec:0d:9a:38:df:3a	1G	1500	Mgmt

```
IP Details
-----
IP: 10.143.33.188/24
```

- Configure the spine switches, **spine3** and **spine4**, as MLAG peers.

```
# net add clag peer \
    sys-mac <MAC> \
    interface <PEERLINK-INTERFACES> \
    <ROLE> \
    backup-ip <IP>
```

** Please note:

- Interfaces **swp3** and **swp4** will be configured as the MLAG **peerlink**.
- Switch **spine3** is configured as the MLAG **primary** and **spine4** as the **secondary**.
- On switch **spine3** use **spine4's** IP as the backup-IP and vice-versa.

```
cumulus@spine3:~$ net add clag peer \  
    sys-mac 44:38:39:FF:00:01 \  
    interface swp3-4 \  
    primary \  
    backup-ip 10.143.33.188  
  
cumulus@spine3:~$ net commit
```

```
cumulus@spine4:~$ net add clag peer \  
    sys-mac 44:38:39:FF:00:01 \  
    interface swp3-4 \  
    secondary \  
    backup-ip 10.143.33.187  
  
cumulus@spine4:~$ net commit
```

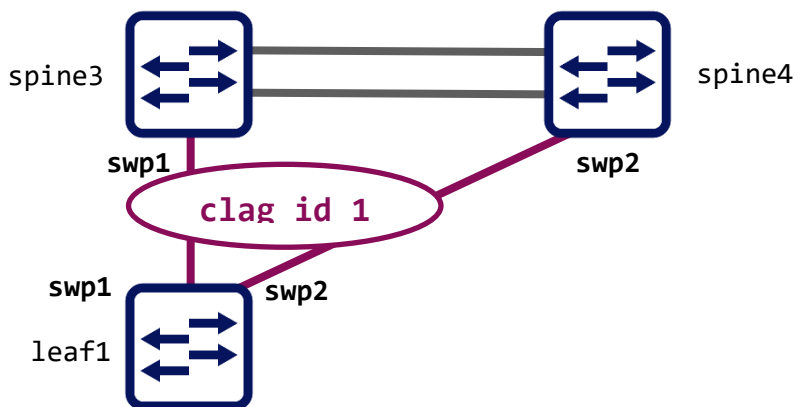
c. Configure bridge settings – VLANs and STP priority:

```
cumulus@spine3:~$ net add vlan 2-3  
cumulus@spine3:~$ net add bridge stp treeprio 4096  
cumulus@spine3:~$ net commit
```

```
cumulus@spine4:~$ net add vlan 2-3  
cumulus@spine4:~$ net add bridge stp treeprio 4096  
cumulus@spine4:~$ net commit
```


Task 3: Configuring CLAG interfaces - spine switches

- a. Configure two CLAG *interfaces* on each of the spine switches.
 - Interface '**clag 1**' will aggregate **swp1** on **spine3** and **swp2** on **spine4** which are connected to switch **leaf1**

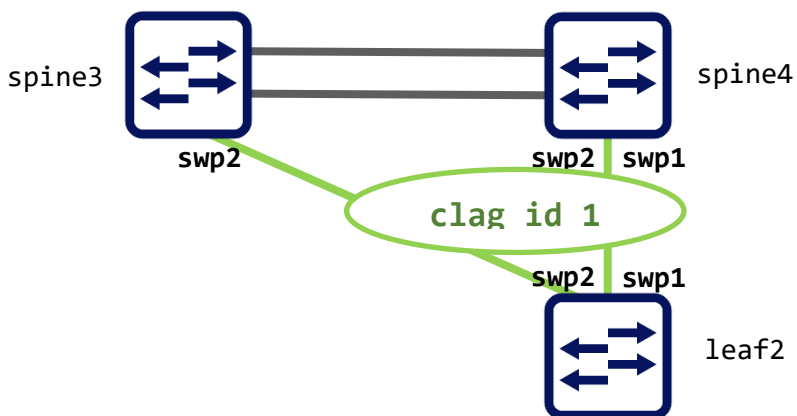


```
# net add clag port bond <BONDNAME> interface <INTERFACE> clag-id <ID>
```

```
cumulus@spine3:~$ net add clag port bond LEAF1 interface swp1 clag-id 1
cumulus@spine3:~$ net commit
```

```
cumulus@spine4:~$ net add clag port bond LEAF1 interface swp2 clag-id 1
cumulus@spine4:~$ net commit
```

- Interface '**clag 2**' will aggregate **swp2** on **spine3** and **swp1** on **spine4** which are connected to switch **leaf2**.



```
cumulus@spine3:~$ net add clag port bond LEAF2 interface swp2 clag-id 2
cumulus@spine3:~$ net commit
```

```
cumulus@spine4:~$ net add clag port bond LEAF2 interface swp1 clag-id 2
cumulus@spine4:~$ net commit
```

b. View MLAG resulting configuration:

```
cumulus@spine3:~$ net show configuration
```

```
interface LEAF1
  bond-slaves swp1
  clag-id 1

interface LEAF2
  bond-slaves swp2
  clag-id 2

interface bridge
  bridge-ports peerlink LEAF1 LEAF2
  bridge-vlan-aware yes

interface peerlink
  bond-slaves swp3 swp4

interface peerlink.4094
  address 169.254.1.1/30
  clagd-backup-ip 10.143.33.188
  clagd-peer-ip 169.254.1.2
  clagd-priority 1000
  clagd-sys-mac 44:38:39:FF:00:01
```

```
cumulus@spine4:~$ net show configuration
```

```
interface LEAF1
  bond-slaves swp2
  clag-id 1

interface LEAF2
  bond-slaves swp1
  clag-id 2

interface bridge
  bridge-ports peerlink LEAF1 LEAF2
  bridge-vlan-aware yes

interface peerlink
  bond-slaves swp3 swp4

interface peerlink.4094
  address 169.254.1.2/30
  clagd-backup-ip 10.143.33.187
  clagd-peer-ip 169.254.1.1
  clagd-priority 2000
  clagd-sys-mac 44:38:39:FF:00:01
```

c. Verify that MLAG protocol is up:

net show clag

```
cumulus@spine3:~$ net show clag
```

The peer is alive

```
Our Priority, ID, and Role: 1000 24:8a:07:cf:6a:50 primary
Peer Priority, ID, and Role: 2000 24:8a:07:cf:6d:d0 secondary
Peer Interface and IP: peerlink.4094 169.254.1.2
Backup IP: 10.143.33.188 (active)
System MAC: 44:38:39:ff:00:01
```

CLAG Interfaces

Our Interface	Peer Interface	CLAG Id	Conflicts	Proto-Down Reason
-----	-----	-----	-----	-----
LEAF1	LEAF1	1	-	-
LEAF2	LEAF2	2	-	-

Task 4: Servers' IP settings

Access the server that was assigned to you, and configure an IP address for interface 'eth2' (see tables below).

VLAN ID	Server	'eth2' IP Address
VLAN 2	host1	172.16.2.18/24
VLAN 3	host2	172.16.3.19/24
VLAN 2	host3	172.16.2.28/24
VLAN 3	host4	172.16.3.29/24

d. Configure the server's IP address and subnet mask:

- Clear existing IP configuration:
ifconfig eth2 0.0.0.0
- Configure an IP address and a subnet mask:
ifconfig eth2 IP_ADDRESS/SUBNET_MASK
- Verify IP configuration:
ifconfig eth2

```
cumulus@host1:~$ sudo ifconfig eth2 0.0.0.0
cumulus@host1:~$ sudo ifconfig eth2 172.16.2.18/24
cumulus@host1:~$ ifconfig eth2
eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.16.2.18 netmask 255.255.255.0 broadcast 172.16.2.255
    inet6 fe80::4638:39ff:fe00:11 prefixlen 64 scopeid 0x20<link>
    ether 44:38:39:00:00:11 txqueuelen 1000 (Ethernet)
    RX packets 3 bytes 180 (180.0 B)
```

e. Verify a static route entry to network 172.16.0.0/16 via the default gateway's address. If the entry is missing, add it.

**** Please note:**

- Make sure to use the following table for default gateway assignment.

VLAN ID	Default gateway address
vlan 2	172.16.2.254/24
vlan 3	172.16.3.254/24

ip route add <NET_ADDRESS/SUBNET_MASK> dev <DEV> via IP_ADDRESS

```
cumulus@host1:~$ sudo ip route add 172.16.0.0/16 dev eth2 via 172.16.2.254
cumulus@host1:~$ route
Kernel IP routing table
Destination      Gateway          Genmask          Flags Metric Ref    Use Iface
default          _gateway        0.0.0.0          UG    0      0      0 eth0
172.16.0.0       172.16.2.254    255.255.0.0      UG    0      0      0 eth2
172.16.2.0       0.0.0.0         255.255.255.0    U     0      0      0 eth2
192.168.200.0    0.0.0.0         255.255.255.0    U     0      0      0 eth0
```

- f. Use 'ping' to check communication between servers in the same VLAN.
For example server **host1** and **host3** in group B.

```
[cumulus@host1 ~]# ping 172.16.2.28
PING 172.16.2.28 (172.16.2.28) 56(84) bytes of data.
64 bytes from 172.16.2.28: icmp_seq=1 ttl=64 time=0.147 ms
64 bytes from 172.16.2.28: icmp_seq=2 ttl=64 time=0.135 ms
```

Verify that the MLAG switches have their MAC address tables synchronized.

```
cumulus@spine3:~$ net show bridge macs
```

VLAN	Master	Interface	MAC	TunnelDest	State	Flags	LastSeen
2	bridge	LEAF1	ec:0d:9a:6f:97:0b				00:23:43
2	bridge	LEAF2	ec:0d:9a:46:9e:b5				00:23:43
3	bridge	LEAF1	ec:0d:9a:6f:96:fb				00:23:43
3	bridge	LEAF2	ec:0d:9a:46:9f:8d				00:23:43

```
cumulus@spine4:~$ net show bridge macs
```

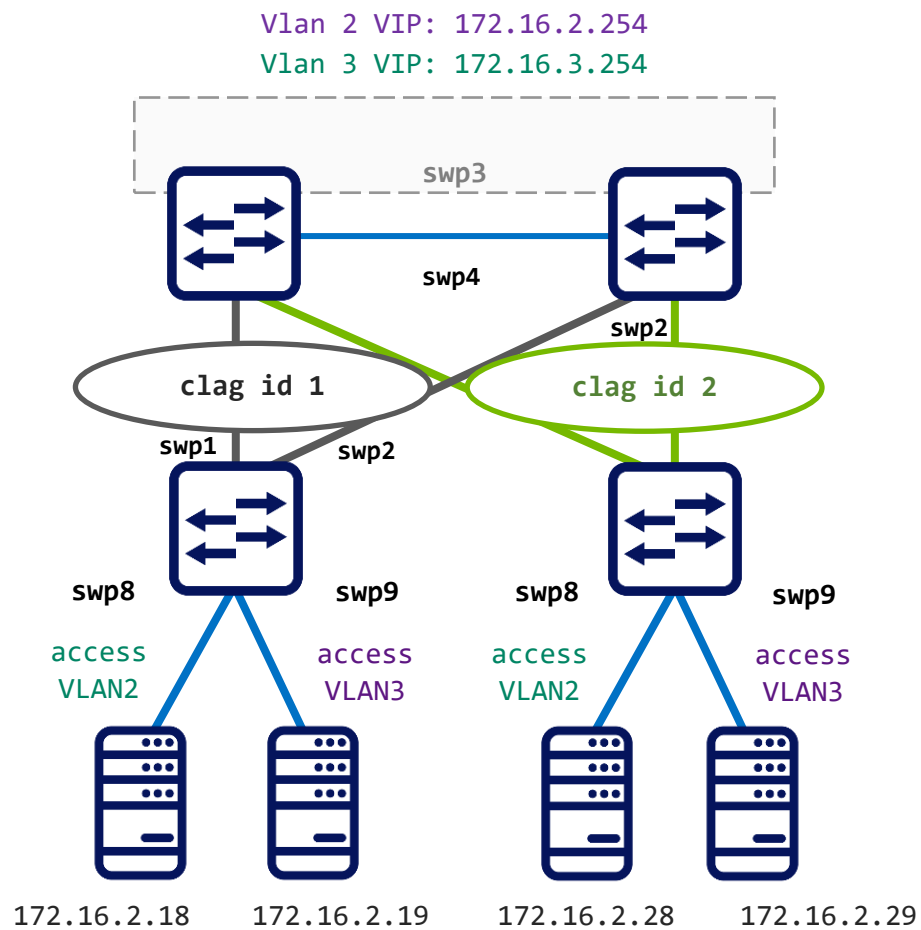
VLAN	Master	Interface	MAC	TunnelDest	State	Flags	LastSeen
2	bridge	LEAF1	ec:0d:9a:6f:97:0b				00:02:14
2	bridge	LEAF2	ec:0d:9a:46:9e:b5				00:02:14
3	bridge	LEAF1	ec:0d:9a:6f:96:fb				00:02:14
3	bridge	LEAF2	ec:0d:9a:46:9f:8d				00:02:14

**** Please note:**

- Once the MLAG configuration is completed, the spine switches appear as a single Layer 2 switch towards to Layer 2 network. Hence, they are capable to provide an efficient load balancing and better network utilization for the layer 2 network.
- In the following task additionally, the spine switches will be configured as VRR routers. Thus, they will provide default gateway redundancy and efficient load balancing towards the layer 3 network.

Task 5: Configuring VRR

Topology used in this task:



- a. Configure two SVIs (Switch Virtual Interfaces) on each of the spine switches, interface vlan 2 and interface vlan 3.
 - Configure an IP for each of the vlan interfaces.
 - Configure a VIP (Virtual IP) and a VMAC (Virtual MAC).
 - Use the following table for address assignment.
 - Use /24 as the subnet mask.

Switch spine3

VLAN	SVI	VIP	VMAC
vlan 2	172.16.2.252/24	172.16.2.254/24	00:00:5e:00:01:02
vlan 3	172.16.3.252/24	172.16.3.254/24	00:00:5e:00:01:03

Switch spine4

VLAN	SVI	VIP	VMAC
vlan 2	172.16.2.253/24	172.16.2.254/24	00:00:5e:00:01:02
vlan 3	172.16.3.253/24	172.16.3.254/24	00:00:5e:00:01:03

```
# net add vlan <VLAN_ID> ip address <IP_ADDRESS/SUBNET_MASK>
```

```
# net add vlan <VLAN_ID> ip address-virtual <VMAC> <VIP/SUBNET_MASK>
```

```
cumulus@spine3:~$ net add vlan 2 ip address 172.16.2.252/24
cumulus@spine3:~$ net add vlan 2 ip address-virtual 00:00:5e:00:01:02 172.16.2.254/24
cumulus@spine3:~$ net add vlan 3 ip address 172.16.3.252/24
cumulus@spine3:~$ net add vlan 3 ip address-virtual 00:00:5e:00:01:03 172.16.3.254/24
cumulus@spine3:~$ net commit
```

```
cumulus@spine4:~$ net add vlan 2 ip address 172.16.2.253/24
cumulus@spine4:~$ net add vlan 2 ip address-virtual 00:00:5e:00:01:02 172.16.2.254/24
cumulus@spine4:~$ net add vlan 3 ip address 172.16.3.253/24
cumulus@spine4:~$ net add vlan 3 ip address-virtual 00:00:5e:00:01:03 172.16.3.254/24
cumulus@spine4:~$ net commit
```


b. Use show commands to verify VRR configuration:

```
cumulus@spine3:~$ net show interface
```

State	Name	Spd	MTU	Mode	LLDP	Summary
UP	vlan2	N/A	1500	Interface/L3		IP: 172.16.2.252/24
UP	vlan2-v0	N/A	1500	Interface/L3		IP: 172.16.2.254/24
UP	vlan3	N/A	1500	Interface/L3		IP: 172.16.3.252/24
UP	vlan3-v0	N/A	1500	Interface/L3		IP: 172.16.3.254/24

```
cumulus@spine3:~$ net show interface vlan2-v0
```

Name	MAC	Speed	MTU	Mode
UP vlan2-v0	00:00:5e:00:01:02	N/A	1500	Interface/L3

IP Details

IP: 172.16.2.254/24

IP Neighbor(ARP) Entries: 0

c. Verify VRR operation.

Use **'ping'** and **'traceroute'** utilities to verify communication between hosts in different VLANs.

For example, in group B ping from server **'host1'** (in VLAN 2) to server **'host2'** (in VLAN3).

```
[cumulus@host1 ~]# ping 172.16.3.19
```

```
PING 172.16.3.19 (172.16.3.19) 56(84) bytes of data.
```

```
64 bytes from 172.16.3.19: icmp_seq=1 ttl=63 time=0.175 ms
```

```
64 bytes from 172.16.3.19: icmp_seq=2 ttl=63 time=0.081 ms
```

```
64 bytes from 172.16.3.19: icmp_seq=3 ttl=63 time=0.103 ms
```

```
[cumulus@host1 ~]# traceroute 172.16.3.19
```

```
traceroute to 172.16.3.19 (172.16.3.19), 30 hops max, 60 byte packets
```

```
1 172.16.2.254 (172.16.2.254) 0.195 ms 0.187 ms 0.194 ms
```

```
2 172.16.3.19 (172.16.3.19) 0.109 ms 0.099 ms 0.116 ms
```

d. Verify MLAG/VRR failover.

Use continuous '**ping**' between hosts in different VLANs.

For example, in group B ping from server '**host1**' (in VLAN 2) to server '**host2**' (in VLAN 3).

While '**ping**' is running, reboot switch **spine3**. Was the traffic disrupted?

After switch **spine3** reboots, reboot switch **spine4**. Was the traffic disrupted now?

What are your conclusions regarding MLAG/VRR failover?

PRACTICE 5: CONFIGURING BGP UNNUMBERED

Practice objectives:

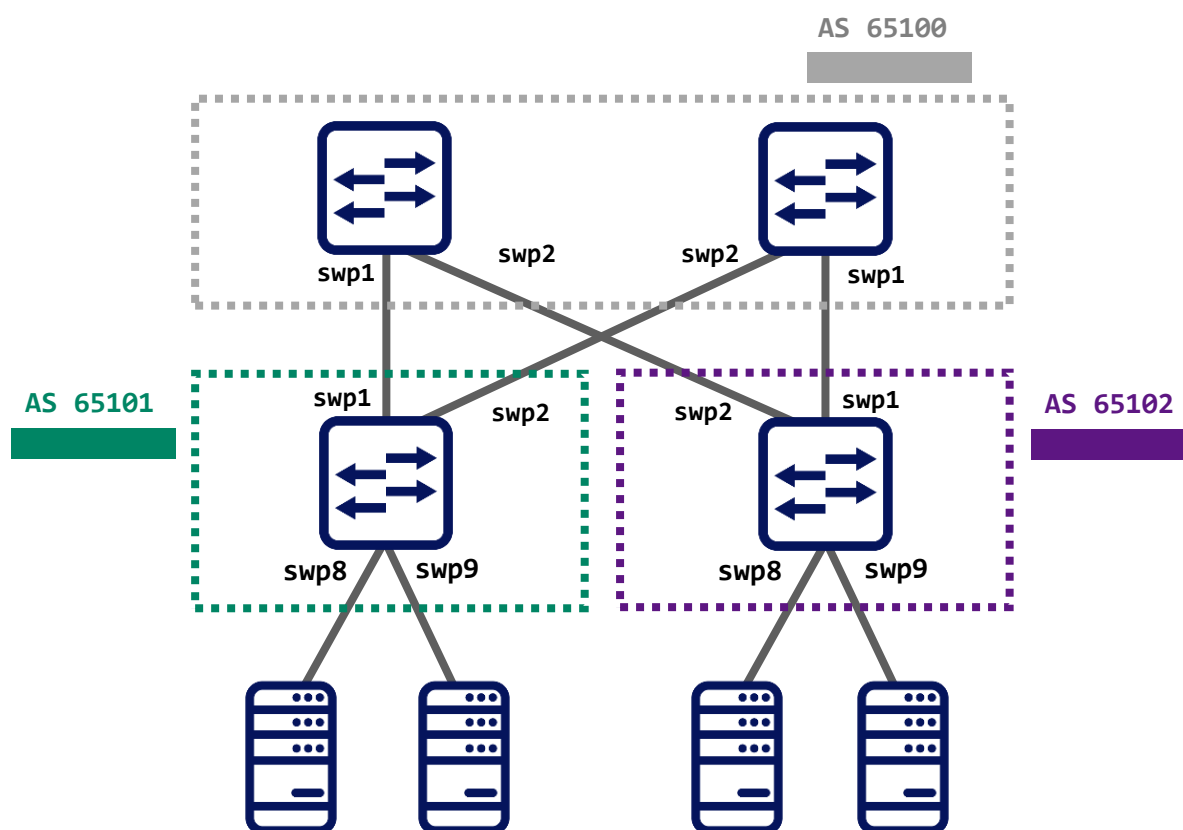
In this practice session you will configure BGP Unnumbered:

- Spine switches will be configured in the same AS and each of the leaf switches will be configured in its own AS.
- BGP unnumbered will be configured on all four switches.
- eBGP sessions will be established between the spine and leaf switches.
- Leaf switches will advertise their local IP prefixes.
- By the end of this practice session you will achieve end-to-end connectivity between all servers in your group, over the BGP Autonomous Systems.

** Please note:

- Commands are demonstrated on switches **leaf1** and **spine3** in group B, named **leaf1** and **spine3**.
You should apply similar commands on the other two switches in your group.

Topology used in this practice session:



Task 1: Starting FRR

- a. Access the switch that was assigned to you and reset configuration:

```
# net del all
# net commit
```

```
cumulus@leaf1:~$ net del all
cumulus@leaf1:~$ net commit
```

- b. Start FRR routing daemons.

Edit the `/etc/frr/daemons` file. Set to **'yes'** both **zebra** and **bgpd**.
Save and exit.

```
cumulus@leaf1:~$ sudo vi /etc/frr/daemons

zebra=yes
bgpd=yes
```

- c. Restart FRR Service.

```
cumulus@leaf1:~$ sudo systemctl restart frr
```

**** Please note:**

- **'sudo'** privileges are required for those actions.

Task 2: Configuring BGP Unnumbered

- a. Configure BGP unnumbered:

```
# net add bgp autonomous-system <LOCAL-AS>
# net add bgp neighbor <INTERFACE> interface remote-as
external
```

```
cumulus@leaf1:~$ net add bgp autonomous-system 65101
cumulus@leaf1:~$ net add bgp neighbor swp1-2 interface remote-as external
cumulus@leaf1:~$ net commit
```

```
cumulus@spine3:~$ net add bgp autonomous-system 65100
cumulus@spine3:~$ net add bgp neighbor swp1-2 interface remote-as external
cumulus@spine3:~$ net commit
```

b. Verify configuration:

```
cumulus@leaf2:~$ net show configuration

<output omitted>

interface swp1

interface swp2

router bgp 65102
  neighbor swp1 interface remote-as external
  neighbor swp2 interface remote-as external
```

c. Verify eBGP sessions were established:

Each switch should be able to see two eBGP neighbors, via interfaces **swp1** and **swp2**.

net show bgp summary

```
cumulus@leaf1:~$ net show bgp summary

show bgp ipv4 unicast summary
=====
BGP router identifier 10.143.33.185, local AS number 65101 vrf-id 0
BGP table version 0
RIB entries 0, using 0 bytes of memory
Peers 2, using 39 KiB of memory

Neighbor          V    AS MsgRcvd MsgSent  TblVer  InQ OutQ  Up/Down  State/PfxRcd
spine3(sw1) 4    65100    11      13      0    0    0 00:00:26      0
spine4(sw2) 4    65100     8       8      0    0    0 00:00:18      0

Total number of neighbors 2
```

Task 3: Configuring loopback interfaces

- a. On each of the switches configure a loopback interface and advertise its prefix in the BGP process.

Use the following table for IP address assignment:

Switch	Loopback IP address
leaf1	172.16.100.1/32
leaf2	172.16.100.2/32
spine3	172.16.100.3/32
spine4	172.16.100.4/32

```
# net add loopback lo ip address IP/MASK
# net add bgp network IP/MASK
```

```
cumulus@leaf1:~$ net add loopback lo ip address 172.16.100.1/32
cumulus@leaf1:~$ net add bgp network 172.16.100.1/32
cumulus@leaf1:~$ net commit
```

```
cumulus@spine3:~$ net add loopback lo ip address 172.16.100.3/32
cumulus@spine3:~$ net add bgp network 172.16.100.3/32
cumulus@spine3:~$ net commit
```

- b. Verify loopback prefixes are reachable:

- Each switch should have in its routing table the prefixes of the other switches loopback interfaces.

```
# net show route bgp
```

```
cumulus@leaf1:~$ net show route bgp
RIB entry for bgp
=====
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, P - PIM, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel,
       > - selected route, * - FIB route

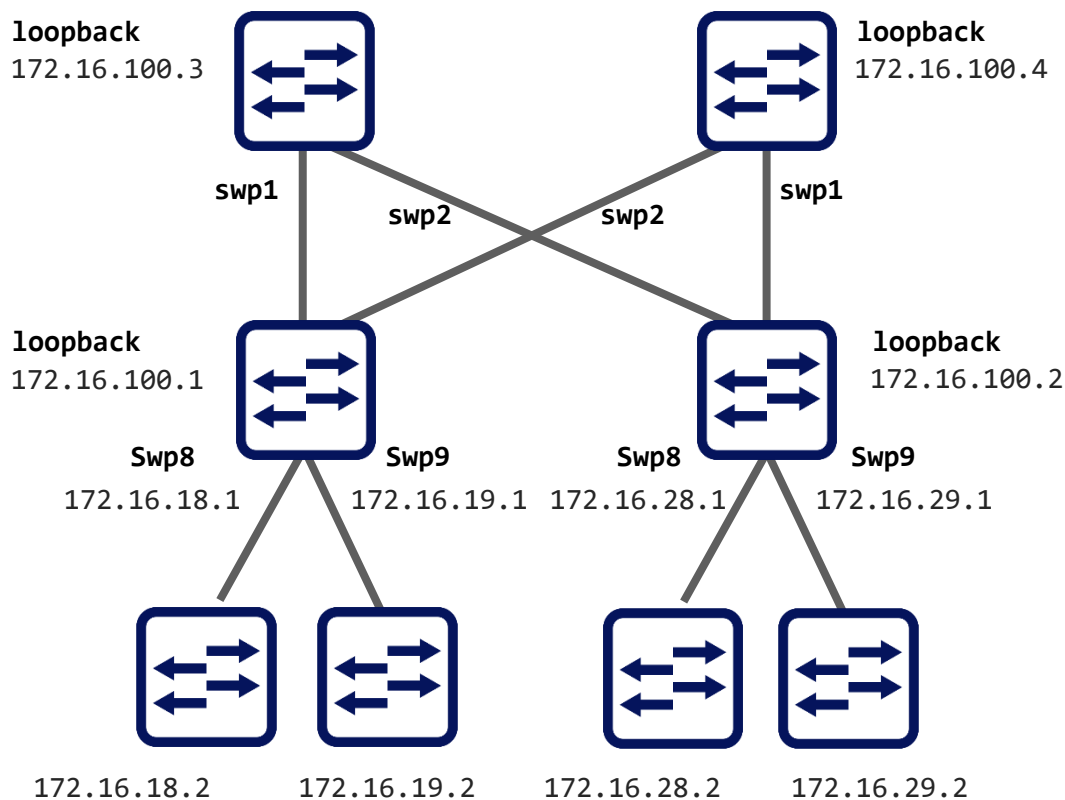
B>* 172.16.100.2/32 [20/0] via fe80::268a:7ff:febf:6a58, swp1, 00:03:57
   *                via fe80::268a:7ff:febf:6ddc, swp2, 00:03:57
B>* 172.16.100.3/32 [20/0] via fe80::268a:7ff:febf:6a58, swp1, 00:04:13
B>* 172.16.100.4/32 [20/0] via fe80::268a:7ff:febf:6ddc, swp2, 00:03:57
```

**** Please note:**

- NEXT-HOP is the neighbor's IPv6 Link Local Address
- BGP multipath is enabled

Task 4: Advertise local IP prefixes

- On the leaf switches, assign IP addresses for the host-facing interface, **swp8** and **swp9**. Advertise the IP prefixes in the BGP process. Use the IP addresses listed in the below topology. Use /24 for the subnet mask.



```
# net add interface <INTERFACE> ip add IP/MASK
```

```
# net add bgp network IP/MASK
```

```
cumulus@leaf1:~$ net add interface swp8 ip add 172.16.18.1/24
cumulus@leaf1:~$ net add interface swp9 ip add 172.16.19.1/24
cumulus@leaf1:~$ net add bgp network 172.16.18.0/24
cumulus@leaf1:~$ net add bgp network 172.16.19.0/24
cumulus@leaf1:~$ net commit
```

```
cumulus@leaf2:~$ net add interface swp8 ip add 172.16.28.1/24
cumulus@leaf2:~$ net add interface swp9 ip add 172.16.29.1/24
cumulus@leaf2:~$ net add bgp network 172.16.28.0/24
cumulus@leaf2:~$ net add bgp network 172.16.29.0/24
cumulus@leaf2:~$ net commit
```

b. Verify that the remote leaf switch has learned the IP prefixes:

- Switch **leaf1** should have in its routing table the IP prefixes advertise by switch **leaf2**:
 - 172.16.28.1/24
 - 172.16.29.0/24
- Switch **leaf2** should have in its routing table the IP prefixes advertise by switch **leaf1**:
 - 172.16.18.1/24
 - 172.16.19.0/24

net show route bgp

```
cumulus@leaf1:~$ net show route bgp
RIB entry for bgp
=====
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, P - PIM, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel,
       > - selected route, * - FIB route

B>* 172.16.28.0/24 [20/0] via fe80::268a:7ff:febf:6a58, swp1, 00:19:09
   *                  via fe80::268a:7ff:febf:6ddc, swp2, 00:19:09
B>* 172.16.29.0/24 [20/0] via fe80::268a:7ff:febf:6a58, swp1, 00:19:09
   *                  via fe80::268a:7ff:febf:6ddc, swp2, 00:19:09
<output omitted>
```

```
cumulus@leaf2:~$ net show route bgp
RIB entry for bgp
=====
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, P - PIM, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel,
       > - selected route, * - FIB route

B>* 172.16.18.0/24 [20/0] via fe80::268a:7ff:febf:6a5c, swp2, 00:21:52
   *                  via fe80::268a:7ff:febf:6dd8, swp1, 00:21:52
B>* 172.16.19.0/24 [20/0] via fe80::268a:7ff:febf:6a5c, swp2, 00:21:52
   *                  via fe80::268a:7ff:febf:6dd8, swp1, 00:21:52
<output omitted>
```


Task 5: Verify end-to-end connectivity

- a. Configure the servers IP settings. Configure an IP address and a subnet mask for interface 'eth2'. Use the following tables for IP address assignment.

Use /24 as the subnet mask.

Server	'eth2' IP Address	Next Hop to 172.16.0.0/16
host1	172.16.18.2	172.16.18.1
host2	172.16.19.2	172.16.19.1
host3	172.16.28.2	172.16.28.1
host4	172.16.29.2	172.16.29.1

- Clear existing IP configuration:

```
# ifconfig <dev> 0.0.0.0
```

- Configure an IP address and a subnet mask:

```
# ifconfig <dev> IP/MASK
```

```
cumulus@host1:~$ sudo ifconfig eth2 0.0.0.0
cumulus@host1:~$ sudo ifconfig eth2 172.16.18.2/24
cumulus@host1:~$ ifconfig eth2
eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.16.18.2 netmask 255.255.255.0 broadcast 172.16.18.255
    inet6 fe80::4638:39ff:fe00:11 prefixlen 64 scopeid 0x20<link>
    ether 44:38:39:00:00:11 txqueuelen 1000 (Ethernet)
    RX packets 3 bytes 180 (180.0 B)
```

- b. Add a route entry to 172.16.0.0/16 via interface 'eth2':

```
# ip route add <ADDRESS/MASK> dev <DEV> via <IP_ADDRESS>
```

```
cumulus@host1:~$ sudo ip route add 172.16.0.0/16 dev eth2 via 172.16.18.1
```

```
cumulus@host1:~$ route
Kernel IP routing table
Destination      Gateway          Genmask         Flags Metric Ref    Use Iface
default          _gateway        0.0.0.0         UG    0      0      0 eth0
172.16.0.0       172.16.18.1     255.255.0.0     UG    0      0      0 eth2
172.16.2.0       0.0.0.0         255.255.255.0   U     0      0      0 eth2
192.168.200.0    0.0.0.0         255.255.255.0   U     0      0      0 eth0
```

- c. Use **'ping'** and **'traceroute'** utilities to verify communication between servers in different Autonomous Systems.

For example, in group B ping from server **'host1'** (AS 65101) to server **'host3'** (in AS 65102).

```
[cumulus@host1 ~]# ping 172.16.28.2
PING 172.16.28.2 (172.16.28.2) 56(84) bytes of data.
64 bytes from 172.16.28.2: icmp_seq=1 ttl=61 time=0.211 ms
64 bytes from 172.16.28.2: icmp_seq=2 ttl=61 time=0.114 ms
```

```
[cumulus@host1 ~]# traceroute 172.16.28.2
traceroute to 172.16.28.2 (172.16.28.2), 30 hops max, 60 byte packets
 1  172.16.18.1 (172.16.18.1)  0.316 ms  0.273 ms  0.263 ms
 2  172.16.100.4 (172.16.100.4)  0.318 ms  0.293 ms  0.305 ms
 3  172.16.100.2 (172.16.100.2)  0.272 ms  0.301 ms  0.316 ms
 4  172.16.28.2 (172.16.28.2)  0.224 ms  0.187 ms  0.146 ms
```

Please note:

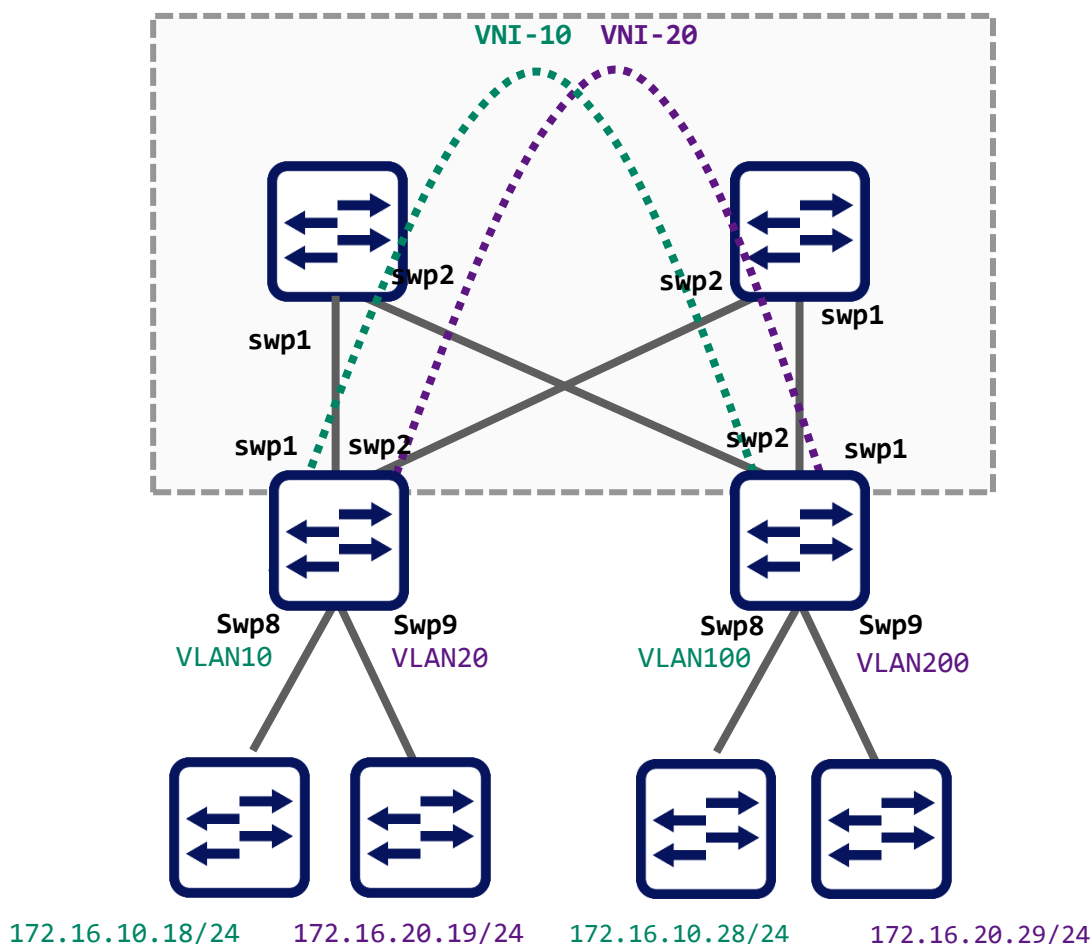
- The next practice relies on BGP Unnumbered configuration. Please make sure to save the configuration before exiting.

Practice objectives:

- The leaf switches will be configured as the VTEPs
 - EVPN will be configured as the VXLAN control plane
 - Two VXLAN Network IDs (VNIs) will be configured:
 - **VNI-10** will connect **VLAN 10** on switch **leaf1** and **VLAN 100** on switch **leaf2**.
 - **VNI-20** will connect **VLAN 20** on switch **leaf1** and **VLAN 200** on switch **leaf2**.
- Hosts in each VNI will be able to communicate in layer 2 over the underlay layer 3 network.

The configuration in this practice session relies on the previous practice.
Make sure that BGP Unnumbered is properly configured and fully operational.

Topology used in this practice session:



Task 1: Configuring servers IP settings

- a. Configure an IP address and a subnet mask for interface '**eth2**'.

Use the following tables for IP address assignment. Use /24 as the subnet mask.

Server	'eth2' IP Address
host1	172.16.10.18
host2	172.16.20.19
host3	172.16.10.28
host4	172.16.20.29

- Clear existing IP configuration:
ifconfig <dev> 0.0.0.0
- Configure an IP address and a subnet mask:
ifconfig <dev> IP/MASK

Task 2: Configuring VLANs

- a. Access the leaf switch that was assigned to you and configure the host-facing ports **swp8** and **swp9**. First clear any existing configuration, then assign the ports to the appropriate VLAN:

- Switch **leaf1** - assign swp8 to VLAN 10 and swp9 to VLAN 20
- Switch **leaf2** - assign swp8 to VLAN 100 and swp9 to VLAN 200

net add interface <INTERFACE> bridge access <VLAN>

```
cumulus@leaf1:~$ net del interface swp8-9
cumulus@leaf1:~$ net add interface swp8 bridge access 10
cumulus@leaf1:~$ net add interface swp9 bridge access 20
cumulus@leaf1:~$ net commit
```

```
cumulus@leaf2:~$ net del interface swp8-9
cumulus@leaf2:~$ net add interface swp8 bridge access 100
cumulus@leaf2:~$ net add interface swp9 bridge access 200
cumulus@leaf2:~$ net commit
```

Task 3: Configuring VNIs

- a. Access the leaf switch that was assigned to you and configure the VNIs:

- Create a VXLAN interface with a unique ID
- Map the VXLAN to a VLAN
- Configure the VXLAN's local tunnel IP (use the VTEPs loopback)

```
# net add vxlan <VXLAN-NAME> vxlan id <VXLAN-ID>
# net add vxlan <VXLAN-NAME> bridge access <VLAN-ID>
# net add vxlan <VXLAN-NAME> vxlan local-tunnelip <IP-ADDRESS>
# net commit
```

- On switch **leaf1**:
VNI-10 will be mapped to VLAN 10 and VNI-20 will be mapped to VLAN 20.
VXLAN's local tunnel IP is 172.16.100.1
- On switch **leaf2**:
VNI-10 will be mapped to VLAN 100 and VNI-20 will be mapped to VLAN 200.
VXLAN's local tunnel IP is 172.16.100.2

```
cumulus@leaf1:~$ net add vxlan VNI-10 vxlan id 10
cumulus@leaf1:~$ net add vxlan VNI-10 bridge access 10
cumulus@leaf1:~$ net add vxlan VNI-10 vxlan local-tunnelip 172.16.100.1
cumulus@leaf1:~$ net add vxlan VNI-20 vxlan id 20
cumulus@leaf1:~$ net add vxlan VNI-20 bridge access 20
cumulus@leaf1:~$ net add vxlan VNI-20 vxlan local-tunnelip 172.16.100.1
cumulus@leaf1:~$ net commit
```

```
cumulus@leaf2:~$ net add vxlan VNI-10 vxlan id 10
cumulus@leaf2:~$ net add vxlan VNI-10 bridge access 100
cumulus@leaf2:~$ net add vxlan VNI-10 vxlan local-tunnelip 172.16.100.2
cumulus@leaf2:~$ net add vxlan VNI-20 vxlan id 20
cumulus@leaf2:~$ net add vxlan VNI-20 bridge access 200
cumulus@leaf2:~$ net add vxlan VNI-20 vxlan local-tunnelip 172.16.100.2
cumulus@leaf2:~$ net commit
```

Task 4: Configuring EVPN

- Access the spine switch that was assigned and configure EVPN.
Switch **spine4** should be configured similarly.

net add bgp evpn neighbor <INTERFACE> activate

```
cumulus@c1-spine3:~$ net add bgp evpn neighbor swp1-2 activate
cumulus@c1-spine3:~$ net commit
```

- Access the leaf switch that was assigned and configure EVPN to advertise all VNIs (EVPN allows VTEPs to exchange VNI membership information).
Switch **leaf2** should be configured similarly.

net add bgp evpn neighbor <INTERFACE> activate

net add bgp evpn advertise-all-vni

```
cumulus@leaf1:~$ net add bgp evpn neighbor swp1-2 activate
cumulus@leaf1:~$ net add bgp evpn advertise-all-vni
cumulus@leaf1:~$ net commit
```

c. Verify VXLAN Information

net show evpn vni <VNI>

```
cumulus@leaf1:~$ net show evpn vni 10
VNI: 10
Type: L2
Tenant VRF: Default-IP-Routing-Table
VxLAN interface: VNI-10
VxLAN ifIndex: 21
Local VTEP IP: 172.16.100.1
Remote VTEPs for this VNI:
172.16.100.2
Number of MACs (local and remote) known for this VNI: 2
```

Task 5: Verify end-to-end communication

- Each VNI is a logical layer 2 network over the underlay layer 3 network. Logically there are no layer 3 hops between hosts in the same VNI.
- Use '**ping**' and '**traceroute**' utilities to verify end-to-end communication between hosts in a VNI. For example in Group B ping from host '**host1**' to '**host3**' that are configured in VNI-10.

```
[cumulus@host1 ~]# ping 172.16.10.28

PING 172.16.10.28 (172.16.10.28) 56(84) bytes of data.
64 bytes from 172.16.10.28: icmp_seq=1 ttl=64 time=0.221 ms
64 bytes from 172.16.10.28: icmp_seq=2 ttl=64 time=0.098 ms
```

```
[cumulus@host1 ~]# traceroute 172.16.10.28

traceroute to 172.16.10.28 (172.16.10.28), 30 hops max, 60 byte packets
 1  172.16.10.28 (172.16.10.11)  0.143 ms  0.103 ms  0.144 ms
```

** Please note:

The next practice relies on VXLAN with EVPN configuration.

Please make sure to save the configuration before exiting.

PRACTICE 7: CONFIGURING DISTRIBUTED ASYMMETRIC

Practice objectives:

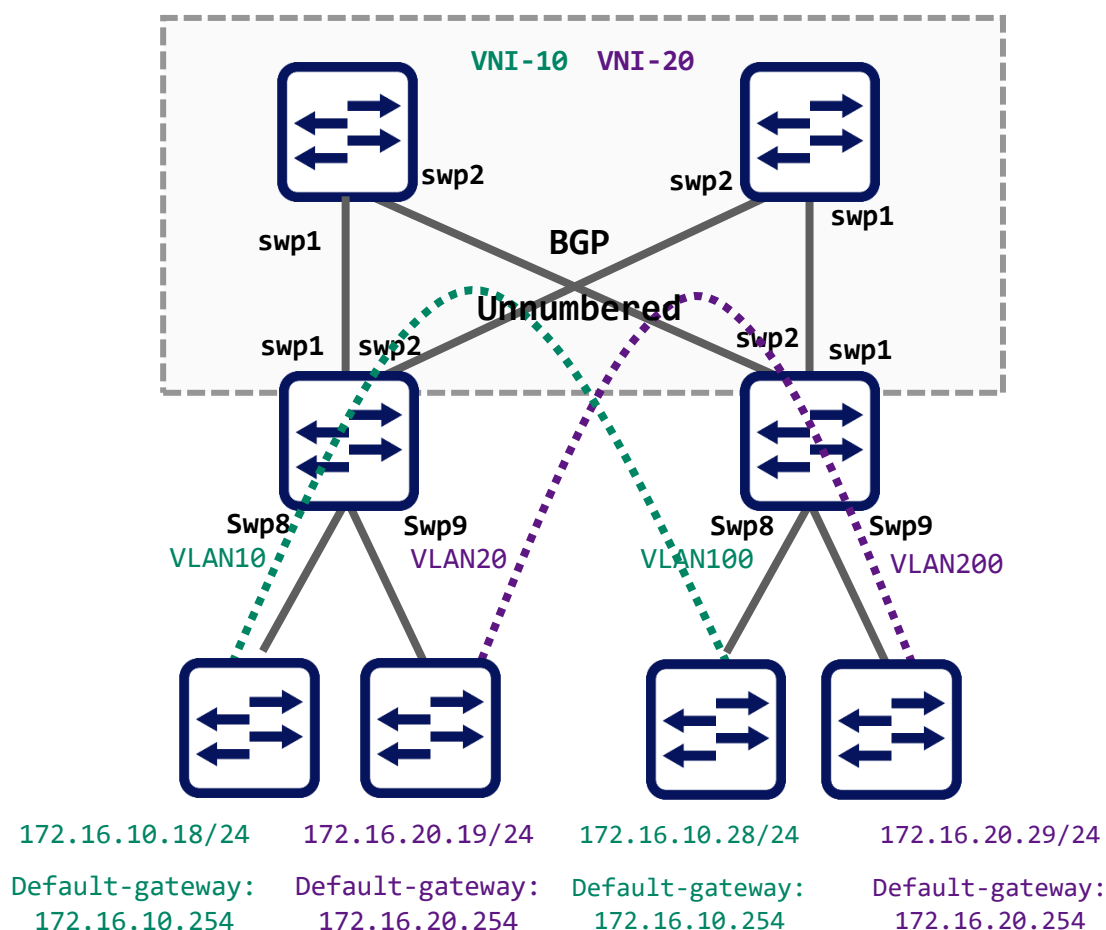
In this practice session you will configure distributed asymmetric VXLAN routing.

- VTEPs (leaf switches) will be configured as the distributed anycast default gateway, i.e., each VTEP will be configured with a VIP and VMAC for each of the subnets. Each VTEP will serve as the default gateway for its locally connected hosts.
- In asymmetric VXLAN routing only the ingress VTEP perform the routing, while the egress VTEP perform VXLAN bridging only.

** Please note:

- The configuration in this practice session relies on the previous practice. Make sure that BGP Unnumbered and VXLAN with EVPN are properly configured and fully operational.

Topology used in this practice session:



Task 1: Configuring servers IP settings

- a. Configure an IP address and a subnet mask for interface **'eth2'**.

If the interface's IP address is not as listed in the table below, use the following table for IP address assignment. Use /24 as the subnet mask.

Server	'eth2' IP Address	Default gateway
host1	172.16.10.18	172.16.10.254
host2	172.16.20.19	172.16.20.254
host3	172.16.10.28	172.16.10.254
host4	172.16.20.29	172.16.20.254

- Clear existing IP configuration:

```
# ifconfig <dev> 0.0.0.0
```

- Configure an IP address and a subnet mask:

```
# ifconfig <dev> IP/MASK
```

```
cumulus@host1:~$ sudo ifconfig eth2 172.16.10.18/24
cumulus@host1:~$ ifconfig eth2
eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 172.16.10.18 netmask 255.255.255.0  broadcast 172.16.10.255
    inet6 fe80::4638:39ff:fe00:11 prefixlen 64 scopeid 0x20<link>
    ether 44:38:39:00:00:11 txqueuelen 1000 (Ethernet)
    RX packets 3  bytes 180 (180.0 B)
    RX errors 0  dropped 3  overruns 0  frame 0
```

- b. Add a static route entry to network 172.16.0.0/16 via the default gateway's IP:

In group B for example:

- Servers 'host1' and 'host3' are in VNI-10, their default gateway is 172.16.10.254
- Servers 'host2' and 'host4' are in VNI-20, their default gateway is 172.16.20.254

```
# ip route add <ADDRESS/MASK> dev <DEV> via <IP>
```

```
[cumulus@host1 ~]# sudo ip route add 172.16.0.0/16 dev eth2 via 172.16.10.254
```

Task 2: Configuring SVIs on the VTEPs

- Access the leaf switch that was assigned to you and configure the SVIs and the anycast default gateway's VMAC and VIP.

Assign IP addresses according the following tables. Use /24 as the subnet mask.

Switch leaf1:

SVI	IP	Default gateway's VIP	Default gateway's VMAC
vlan 10	172.16.10.252	172.16.10.254	00:00:5e:00:01:01
vlan 20	172.16.20.252	172.16.20.254	00:00:5e:00:01:02

Switch leaf2:

SVI	IP	Default gateway's VIP	Default gateway's VMAC
vlan 100	172.16.10.253	172.16.10.254	00:00:5e:00:01:01
vlan 200	172.16.20.253	172.16.20.254	00:00:5e:00:01:02

```
# net add vlan <VLAN> ip address <VLAN>
```

```
# net add vlan <VLAN> ip address-virtual <VMAC> <VIP/MASK>
```

```
cumulus@leaf1:~$ net add vlan 10 ip address 172.16.10.252/24
cumulus@leaf1:~$ net add vlan 10 ip address-virtual 00:00:5e:00:01:01
172.16.10.254/24
```

```
cumulus@leaf1:~$ net add vlan 20 ip address 172.16.20.252/24
cumulus@leaf1:~$ net add vlan 20 ip address-virtual 00:00:5e:00:01:02
172.16.20.254/24
```

```
cumulus@leaf1:~$ net commit
```

```
cumulus@leaf2:~$ net add vlan 100 ip address 172.16.10.253/24
cumulus@leaf2:~$ net add vlan 100 ip address-virtual 00:00:5e:00:01:01
172.16.10.254/24
```

```
cumulus@leaf2:~$ net add vlan 200 ip address 172.16.20.253/24
cumulus@leaf2:~$ net add vlan 200 ip address-virtual 00:00:5e:00:01:02
172.16.20.254/24
```

```
cumulus@leaf2:~$ net commit
```

** Please note:

The VMAC and VIP must be identical on both VTEPs in order to implement anycast default gateway and to allow proper inter-VXLAN routing.

Task 3: Test inter-VXLAN communication

- a. Use '**ping**' and '**traceroute**' utilities to verify end-to-end communication between hosts in different VNIs.

For example in Group B ping from host '**host1**' in VNI-10 to '**host4**' in VNI-20.

In this case the egress switch **leaf1** will perform VXLAN routing between the source and destination VNIs, while the egress switch **leaf2** will bridge between destination VNI and the destination VLAN.

```
[cumulus@host1 ~]# ping 172.16.20.29
PING 172.16.20.29 (172.16.20.29) 56(84) bytes of data.
64 bytes from 172.16.20.29: icmp_seq=37 ttl=63 time=0.152 ms
64 bytes from 172.16.20.29: icmp_seq=38 ttl=63 time=0.157 ms
64 bytes from 172.16.20.29: icmp_seq=39 ttl=63 time=0.138 ms
```

```
[cumulus@host1 ~]# traceroute 172.16.20.29
traceroute to 172.16.20.29 (172.16.20.29), 30 hops max, 60 byte packets
 1  172.16.10.254 (172.16.10.254)  0.275 ms  0.248 ms  0.244 ms
 2  172.16.20.29 (172.16.20.28)  0.165 ms  0.171 ms  0.142 ms
```

PRACTICE 8: EDITING ANSIBLE INVENTORY FILE

Practice objectives:

In this practice session you perform the initial configurations required for Ansible to start working with the group servers and switches.

- You will configure **hosts** and **groups** in your Ansible hosts file.
- You will use **Ansible ping module** to validate the configuration.
- Last, you will use **Ansible Variables** to refine the hosts configuration.

**** Please note:**

- Configurations are demonstrated on group B servers and switches .
You should apply similar commands to devices in the group that was assigned to you.


Task 1: Editing the Ansible Inventory (hosts) file

- Connect to the 'oob-mgmt-server' and Use VIM, or another text editor to edit the **/etc/ansible/hosts** file:

vi /etc/ansible/hosts

```
# Default ansible hosts file
# comments begin with the '#' character
#   - Blank lines are ignored
#   - Groups of hosts are delimited by [header] elements
#   - hostnames or ip addresses are accepted
#   - A hostname/ip can be a member of multiple groups

~
~
~
```

 to exit VIM:

- Press ESC
- Type ':'
- Type "q!" to exit **without saving** or "wq" to **save and exit**

```
~
~
:wq
```

To edit the file using VIM go to insert mode by typing ‘a’
(make sure the word “—INSERT --” appears at the end of the page).

```
~
~
~
~
~
-- INSERT --
```

Task 2: Adding servers to the Inventory (hosts) file

- a. While in “INSERT” mode, add the servers host name to the hosts file.

```
# Default ansible hosts file
# comments begin with the '#' character
#   - Blank lines are ignored
#   - Groups of hosts are delimited by [header] elements
#   - hostnames or ip addresses are accepted
#   - A hostname/ip can be a member of multiple groups

host1
host2
host3
host4
~
~
~
```

**** Please note:**

- Every line that starts with ‘#’ is considered a comment and can be deleted.
- Instead of configuring each server in a different line, you can use a REGEX expression to capture all group servers in one line
host[1:4]

Task 3: Testing Ansible connectivity using the “ping” module

- Save and Exit the hosts file (type ESC, ':', 'wq' and <enter>)

```
# Default ansible hosts file
# comments begin with the '#' character
#   - Blank lines are ignored
#   - Groups of hosts are delimited by [header] elements
#   - hostnames or ip addresses are accepted
#   - A hostname/ip can be a member of multiple groups

host1
host2
host3
host4
~
~
~
:wq
```

- Validate the configuration by using the ping module

```
cumulus@oob-mgmt-server:~$ ansible host3 -m ping
host3 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": false,
  "ping": "pong"
}
cumulus@oob-mgmt-server:~$
```

```
cumulus@oob-mgmt-server:~$ ansible host4 -m ping
host4 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": false,
  "ping": "pong"
}
cumulus@oob-mgmt-server:~$
```

Task 4: Add servers to a host group

- Use VIM to edit the `/etc/ansible/hosts` file, and enter INSERT mode by typing ‘a’
vi /etc/ansible/hosts
- Add all servers to a group called “servers”

```
# Default ansible hosts file
# comments begin with the '#' character
#   - Blank lines are ignored
#   - Groups of hosts are delimited by [header] elements
#   - hostnames or ip addresses are accepted
#   - A hostname/ip can be a member of multiple groups

[servers]
host1
host2
host3
host4
~
~
~
```

- Exit VIM and use the Ansible “ping” module to test the new group configuration.

```
cumulus@oob-mgmt-server:~$ ansible servers -m ping
host3 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": false,
  "ping": "pong"
}
host4 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": false,
  "ping": "pong"
}
host1 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": false,
  "ping": "pong"
}
host2 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": false,
  "ping": "pong"
}
```

Task 5: Add Cumulus Linux switches to the inventory

- Use VIM to edit the `/etc/ansible/hosts` file, and enter INSERT mode by typing ‘a’
vi /etc/ansible/hosts
- Add the leaf switches (‘leaf1’ and ‘leaf2’) to the inventory file, also add the necessary credentials (user and password)

leaf1 ansible_user=cumulus ansible_ssh_pass=Academy123

```
# Default ansible hosts file
# comments begin with the '#' character
#   - Blank lines are ignored
#   - Groups of hosts are delimited by [header] elements
#   - hostnames or ip addresses are accepted
#   - A hostname/ip can be a member of multiple groups

[servers]
host[1:4] ansible_user=cumulus ansible_ssh_pass=Academy123

leaf1 ansible_user=cumulus ansible_ssh_pass=Academy123
leaf2 ansible_user=cumulus ansible_ssh_pass=Academy123

~
~
```

- Exit VIM and use the Ansible “ping” module to test Ansible connectivity to the switches.

```
cumulus@oob-mgmt-server:~$ ansible leaf1 -m ping
leaf1 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": false,
  "ping": "pong"
}
```

**** Please note:**

- You might encounter a warning regarding the Python interpreter on the switches, it can be ignored.
- If the ansible user or password are incorrect, you will get the following error:

```
leaf2 | UNREACHABLE! => {
  "changed": false,
  "msg": "Failed to connect to the host via ssh: Permission denied
(publickey,password).",
  "unreachable": true
}
```


Task 6: Add Cumulus Linux switches to a host group

- Use VIM to edit the `/etc/ansible/hosts` file, and enter INSERT mode by typing ‘a’
vi /etc/ansible/hosts
- Add the **leaf** switches to a group called “leaves”

```
.
.
.

[servers]
host[1:4] ansible_user=cumulus ansible_ssh_pass=Academy123

[leaves]
leaf1 ansible_user=cumulus ansible_ssh_pass=Academy123
leaf2 ansible_user=cumulus ansible_ssh_pass=Academy123
~
~
~
```

- Add the **spine** switches, same way the leaves were added.

```
.
.
.

[servers]
host[1:4] ansible_user=cumulus ansible_ssh_pass=Academy123

[leaves]
leaf1 ansible_user=cumulus ansible_ssh_pass=Academy123
leaf2 ansible_user=cumulus ansible_ssh_pass=Academy123

[spines]
spine3 ansible_user=cumulus ansible_ssh_pass=Academy123
spine4 ansible_user=cumulus ansible_ssh_pass=Academy123

~
~
~
```

- d. Add the leaves and spines to a group called “switches”.

```
.
.
.

[servers]
host[1:4] ansible_user=cumulus ansible_ssh_pass=Academy123

[leaves]
leaf1 ansible_user=cumulus ansible_ssh_pass=Academy123
leaf2 ansible_user=cumulus ansible_ssh_pass=Academy123

[spines]
spine3 ansible_user=cumulus ansible_ssh_pass=Academy123
spine4 ansible_user=cumulus ansible_ssh_pass=Academy123

[switches:children]
spines
leaves

~
~
~
```

- e. Exit VIM and use the Ansible “ping” module to test Ansible connectivity to the switches

```
cumulus@oob-mgmt-server:~$ ansible switches -m ping
leaf2 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": false,
  "ping": "pong"
}
leaf1 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": false,
  "ping": "pong"
}
spine3 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": false,
  "ping": "pong"
}
spine4 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": false,
  "ping": "pong"
}
```

Task 7: Add variables to be shared by the groups

- Add the username and password as variables, to be shared among all 'switches' group members, then delete the definitions on each switch.

```

.
.
.

[group_b_servers]
host[1:4] ansible_user=cumulus ansible_ssh_pass=Academy123

[leaves]
leaf1 ansible_user=cumulus ansible_ssh_pass=Academy123
leaf2 ansible_user=cumulus ansible_ssh_pass=Academy123

[spines]
spine3 ansible_user=cumulus ansible_ssh_pass=Academy123
spine4 ansible_user=cumulus ansible_ssh_pass=Academy123

[switches:children]
spines
leaves

[switches:vars]
ansible_user=cumulus
ansible_ssh_pass=Academy123

~
~
~

```

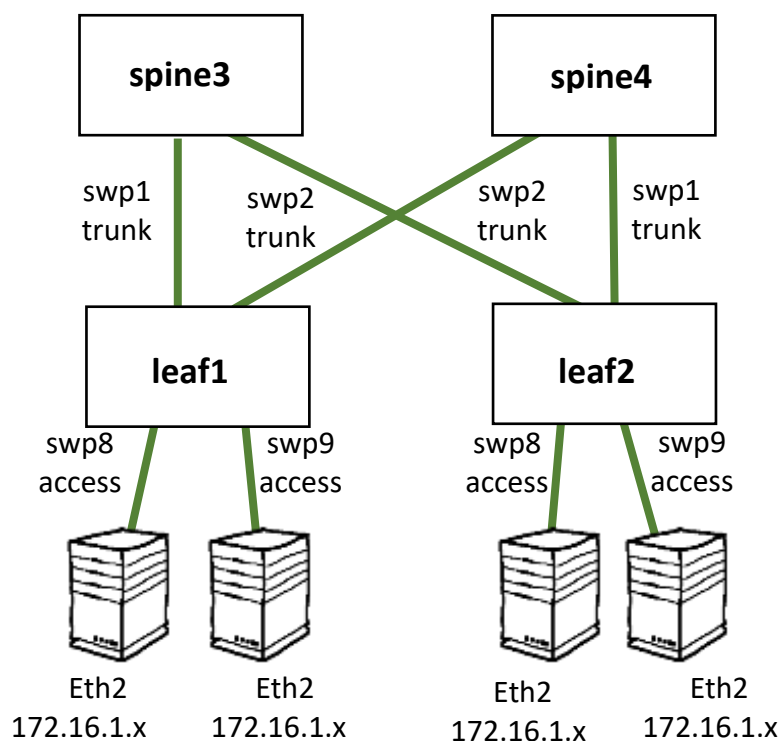
PRACTICE 9: Ansible Playbook

Practice objectives:

In this practice session you configure an Ansible playbook based on a familiar practice exercise

- You will configure the servers IP settings – an IP address, a subnet mask and a default gateway.
- You will configure a bridge on each of the Cumulus Linux switches in your group and add switch ports to the bridge.
- You will execute the playbook you wrote
- Last, you will use **'ping'** utility to verify communication between servers in your group.

Topology used in this practice:



Task 1: Create a new Ansible playbook

- a. Access the Ansible controller server that was assigned to you, and create a new yaml file under the `/etc/ansible/` directory


```
# touch /etc/ansible/labPlaybook.yaml
```

- b. Use VIM or another text editor to edit the `/etc/ansible/labPlaybook.yaml` file:

```
# vi /etc/ansible/labPlaybook.yaml
```

(the file should be empty)

- You should apply similar commands to devices in the group that was assigned to you.
- Every line that starts with '#' is considered a comment and can be deleted.

 Please note*

The tasks and commands that you will configure in the playbook are based on **Practice3** in this lab guide.

Task 2: Resetting old configurations

- a. Add a new task to the playbook, the task purpose is to clear the old configurations before we apply our own. This task should be applied on the servers group.
- b. Add two additional sub-tasks under the task you configured:
 1. Clear the current eth2 interfaces IP addresses in each server
 2. Clear static routes in each server

```
# /etc/ansible/labPlaybook.yaml

- name: Clear server's old configurations
  hosts: servers
  tasks:
    - name: Clear eth2 interface existing IP
      command: ifconfig eth2 0.0.0.0

    - name: Clear static routes for network 172.30.0.0/16
      command: ip route del 172.30.0.0/16
      ignore_errors: yes
```

**** Please note:**

- Clearing an empty routing table might cause errors, and those errors are not relevant to the playbook execution, therefore should be ignored.
Please make sure that the “**ignore_errors**” value is set to “**yes**”.

- c. Add a new task to the playbook, the task purpose is to clear the old switch configurations before we apply our own. This task should be applied on the ‘**switches**’ group.
- d. Add one sub-task under the task you configured:
 1. Use the **NCLU** module to clear all switch configurations with the **net del all** command.

```
# /etc/ansible/labPlaybook.yaml

.
.
.

- hosts: switches
  tasks:

    - name: Clear old switch configurations
      nclu:
        commands:
          - del all
        commit: true
```

Task 3: Configure switches according to the practice exercise

- a. Add a new task to the playbook, the task purpose is to apply configuration that all the switches share. This task should be applied on the **switch group**.
- b. add two sub-task under the task you configured, use **NCLU** module to apply:
 1. configure **trunks** between the switches
 2. create the **VLANs 2,3**.

```
# /etc/ansible/labPlaybook.yaml
.
.
.

- hosts: switches
  tasks:

- name: Create a bridge and set inter switch links as trunk ports
  nclu:
    commands:
      - add bridge bridge ports swp1-2
    commit: false

- name: Create VLANs
  nclu:
    commands:
      - add bridge bridge vids 2,3
    commit: true
```

- c. Add a new task to the playbook, the task purpose is to apply configuration that all **leaf** switches share. This task should be applied on the **leaves group**.
- d. add one sub-task under the task you configured, use **NCLU** module to apply:
 1. set the host-facing ports as access ports, and associate each interface to it's designated VLAN.

```
# /etc/ansible/labPlaybook.yaml
.
.
.

- hosts: leaves
  tasks:

- name: Set the host-facing ports as access ports in VLAN 1 and COMMIT
changes
  nclu:
    commands:
      - add bridge bridge ports swp8-9
      - add interface swp8 bridge access 2
      - add interface swp9 bridge access 3
    commit: true
```

Task 4: Configure servers according to the practice exercise

- a. Add 4 new tasks to the playbook, the tasks purpose is to configure the IP address on each server. Each task should be applied on **each server**.

```
# /etc/ansible/labPlaybook.yaml
.
.
.

- name: Configure host1
  hosts: host1
  tasks:
    - name: Configure IP address
      command: ifconfig eth2 172.30.12.18/24

- name: Configure host2
  hosts: host2
  tasks:
    - name: Configure IP address
      command: ifconfig eth2 172.30.13.19/24

- name: Configure host3
  hosts: host3
  tasks:
    - name: Configure IP address
      command: ifconfig eth2 172.30.12.28/24

- name: Configure host4
  hosts: host4
  tasks:
    - name: Configure IP address
      command: ifconfig eth2 172.30.13.29/24
```

- b. Add 2 new tasks to the playbook, the tasks purpose is to configure static routes for each VLAN subnet. Each task should be applied on the servers associated with the VLAN.

```
# /etc/ansible/labPlaybook.yaml
.
.
.

- name: Add static route entry for VLAN 2
  hosts: host1, host3
  tasks:
    - name: Add default gateway for 172.30.0.0/16
      command: ip route add 172.30.0.0/16 dev eth2 via 172.30.12.254

- name: Add static route entry for VLAN 3
  hosts: host2, host4
  tasks:
    - name: Add default gateway for 172.30.0.0/16
      command: ip route add 172.30.0.0/16 dev eth2 via 172.30.13.254
```


Task 4: Executing the playbook

- a. Access the Ansible controller server that was assigned to you, and make sure that the host file is configured as follows

cat /etc/ansible/hosts

```
.
.
.

[servers]
host[1:4]

[leaves]
leaf1
leaf2

[spines]
spine3
spine4

[switches:children]
spines
leaves

[switches:vars]
ansible_user=cumulus
ansible_ssh_pass=Academy123

~
~
~
```

**** Please note:**

- Configurations are demonstrated on group B servers and switches.
You should apply similar commands to devices in the group that was assigned to you.
- Every line that starts with '#' is considered a comment and can be deleted.

- b. Execute the playbook you wrote

ansible-playbook /etc/ansible/LabPlaybook.yaml

- c. Verify configuration on the servers and switches