QUICK START GUIDE

Running Multiple OcNOS® VMs in EVE-NG Quick Start Guide

August 2023

Contents

About the OcNOS VM

The OcNOS Virtual Machine (VM) from IP Infusion helps you get familiar with OcNOS. The OcNOS VM runs on a standard x86 environment. The OcNOS VM is used to validate configurations and test L2, L3, and MPLS features at your own pace, with no costs associated. Without bare metal switches, OcNOS VM can be on popular open-source software emulators EVE-NG and GNS3, and hypervisors including KVM, VirtualBox, and VMware. This document provides information on how to run OcNOS VM in the EVE-NG environment.

All basic Layer 2, Layer 3, and multicast functionality are available. MPLS support is also available, including limited support of MPLS forwarding. The OcNOS VM comes with a 365 days valid license.

The data plane forwarding functions have limited support. OcNOS VM is designed for feature testing, and not for data plane performance testing or full bandwidth traffic testing.

Benefits of the OcNOS VM

Following are benefits of OcNOS VM:

- Free
- No need to wait for the hardware
- Get familiar with OcNOS software
- Validate configurations
- Test L2, L3, and MPLS features without any risk
- Prototype network operations

Feature List

CLIs for the following features are available. The complete feature set of OcNOS is supported on hardware platforms such as the whitebox switches from Dell, Delta Agema, Edgecore, and UFISpace. For the complete feature list, please contact IP Infusion Sales.

SYSTEM FEATURES

- ARP support
- SSH/Telnet
- SNMP
- Debugging and logging
- AAA
- DHCP, DNS

LAYER-2 FEATURES

- STP/RSTP/MSTP
- BPDU Guard and Root Guard
- VLAN, Private VLAN
- LACP
- LLDP
- VLAN Interface
- QinQ
- 802.1x

LAYER-3 FEATURES

- IPv4 Routing
- VRF Support
- RIP v2, RIP NG
- BFD with BGP, OSPF, ISIS
- BGP
- OSPF v2, OSPF v3
- ISIS
- VRRP

MPLS FEATURES

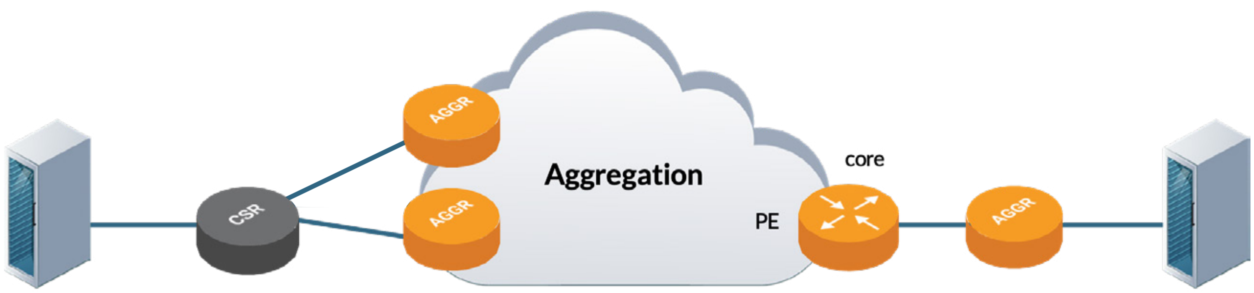
- MPLS Label Switching
- LDP and RSVP Support
- RSVP FRR
- VPLS with LDP Signaling
- VPWS with 1:1 backup support
- BGP MPLS L3VPN
- MPLS DCI using ICCP and VPLS redundancy

MULTICAST FEATURES

- IGMP
- PIM-SM/SSM/DM
- MSDP Support

Running Multiple OcNOS switches in EVE-NG

EVE-NG (Emulated Virtual Environment Next Generation) is a multi-vendor virtual network simulator. This section describes how to install EVE-NG VM in VMware hypervisor and run OcNOS VM switches and test servers in EVE-NG environment. We will create following switch topology shown below to test OcNOS L2 and L3 software features. In this example, we will test the BGP and L3 VPN feature. The following is a test topology in a EVE-NG environment.



One Cell Site Router (CSR), three Aggregation Routers (AGGR) and a core router are used in this EVE-NG test topology. Two Debian Linux servers are used in EVE-NG environment for generating the test traffic.

System Requirements for Running OcNOS VMs in EVE-NG

Following system requirements are used for running OcNOS VMs in EVE-NG. We will run EVE-NG VM in the VMware hypervisor. Following are requirements for running a EVE-NG VM:

- VMware vSphere Hypervisor (ESXi) 6.5.0 or later
- VM requirements:
  - CPU: 4 vCPUs. CPU need to support the nested VM in the ESXi server for running EVE-NG VM. Please refer to the next section for details.
  - Memory: 16 GB
  - Hard Disk: 60 GB
  - NICs: 1. Make sure there is a DHCP server on the network this NIC card is connected to.
- We will be using EVE-NG project image that contains the following VMs: five OcNOS VMs (version 6.3.0 Build 126) with BGP and L3 VPN configuration, and 2 Debian Linux Servers.

Files Provided for Running OcNOS VMs in EVE-NG

Following files are provided for running OcNOS VMs in EVE-NG: You can download these files from the following URL: https://www.ipinfusion.com/products/ocnos-vm/eve-ng/

1. ***OcNOS-SP-MPLS-x86-6.3.0-126-GA.vmdk.xz***: This is OcNOS VM image for the EVE-NG environment. OcNOS VM image file is archive compressed using XZ compression. Use Mac OS Archive Utility or 7-zip tools to uncompress the file. To uncompress the file in Linux, use the command xz -d <file_name>.xz

2. ***ocnos.yml***: This is OcNOS QEMU VM Template. You can import this template to create OcNOS VMs in EVE-NG.

3. ***OcNOS.png***: This is OcNOS switch icon.

4. ***BGP-L3VPN-switches-config.zip***: Configuration files for the topology given in this document. You can copy the configuration given in these files to corresponding OcNOS switch in EVE-NG environment.

Setup the EVE-NG Environment for Validating BGP and L3 VPN

Setting up above topology in EVE-NG for validating BGP and L3 VPN requires the following six steps:

1. Install the remote EVE-NG VM in the VMware hypervisor
2. Install EVE-NG Client Side pack that will install everything necessary for running telnet, vnc and wireshark when working on Building labs
3. Install Linux Ubuntu 21.04 Server in the EVE-NG for generating and receiving test traffic.
4. Install OcNOS VM in the EVE-NG for testing traffic
5. Set up *BGP and L3 VPN* Lab on the EVE-NG
6. Verify *BGP and L3 VPN* Lab

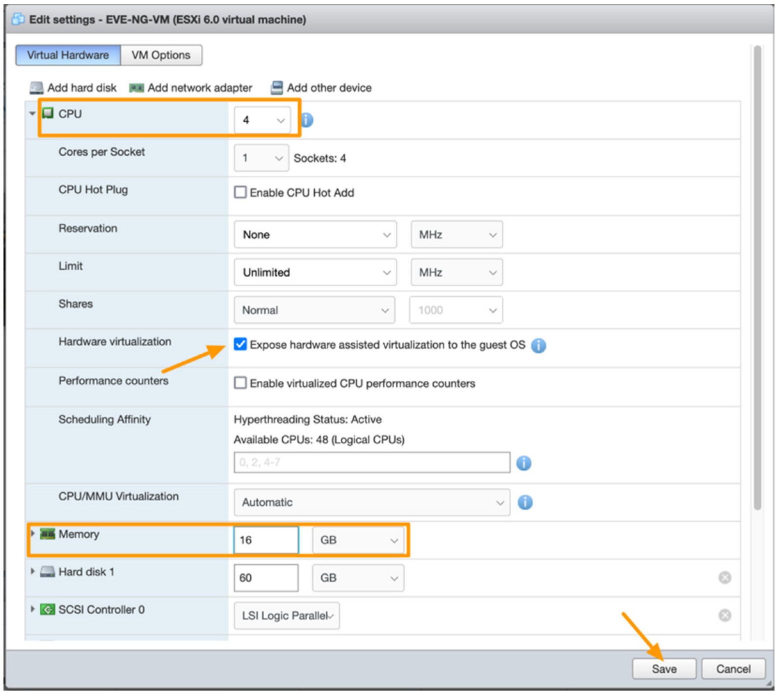
1. Install EVE-NG VM in the VMware vSphere Hypervisor

The following are steps to install a EVE-NG VM in a VMware vSphere hypervisor:

a. **Download the EVE-NG OVF Template Community Version** to run in the VMware vSphere ESXi hypervisor. In this example EVE-NG VM version 5.0.1-19 and VMware ESXi version 7.0.3 are used for testing.

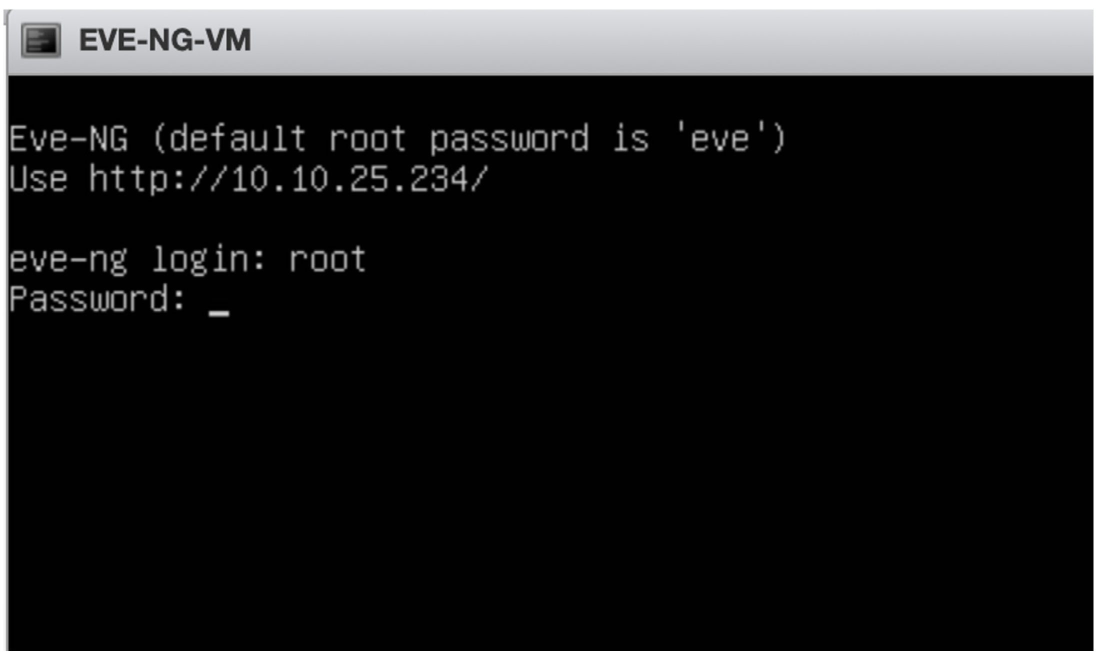
b. **Install EVE-NG VM:** Import EVE-NG OVF template to create a VM named EVE-NG-VM in ESXi server using the downloaded OVF file by following the instructions from this video. Make sure the VM Network to which *EVE-NG-VM* is connected is set to *Accept Promiscuous mode* as instructed in the video. This allows the VM to send multiple MAC addresses to the switches.

c. **Configure *EVE-NG-VM*:** After you install the EVE-NG-VM, turn off the VM power, select edit settings and expand CPU to check the nested VM support in the ESXi server. Hardware Virtualization needs to be enabled in this case as shown below.



In addition, set the Memory of the VM to 16 GB and click Save. CPU in the OVF template is set to 4 and hard disk space is set to up 60 Gb. Click *Save*.

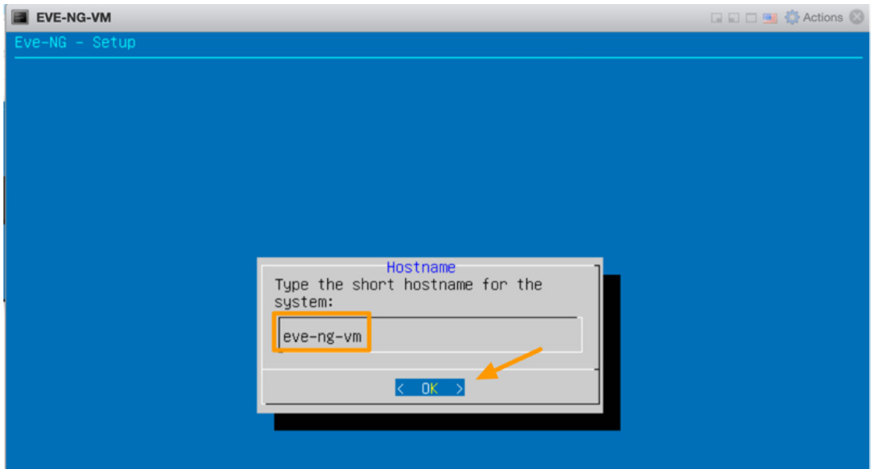
d. Power up the VM and open the VM console as shown below.



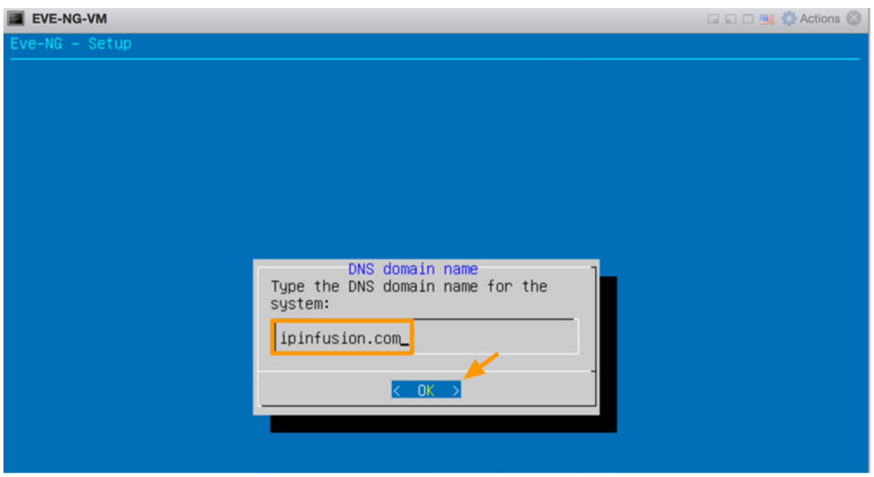
The `EVE-NG-VM` gets its IP address 10.10.25.234. The default credentials for login are also given in the console: username is *root* and password is *eve*. The Web URL to access the *EVE-NG-VM* environment is given as http://10.10.25.234.

Repeat enter root user's password again and click return. Then set new password for root user and hit return and repeat the same thing to confirm the new password set for root and hit return.

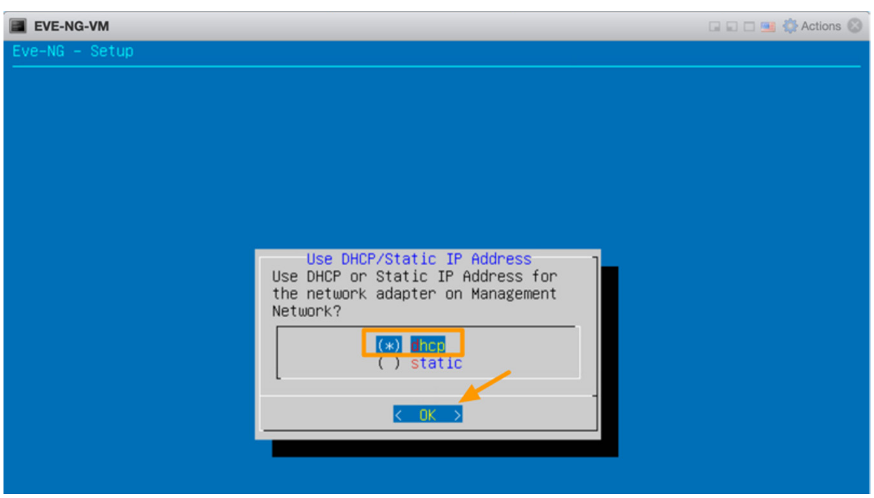
Enter the *hostname* as shown below and hit return.



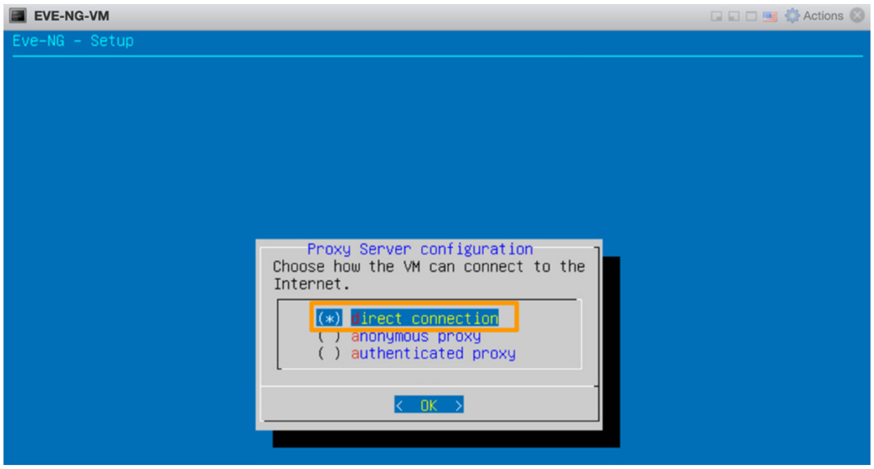
Enter the DNS domain name as shown below and hit return



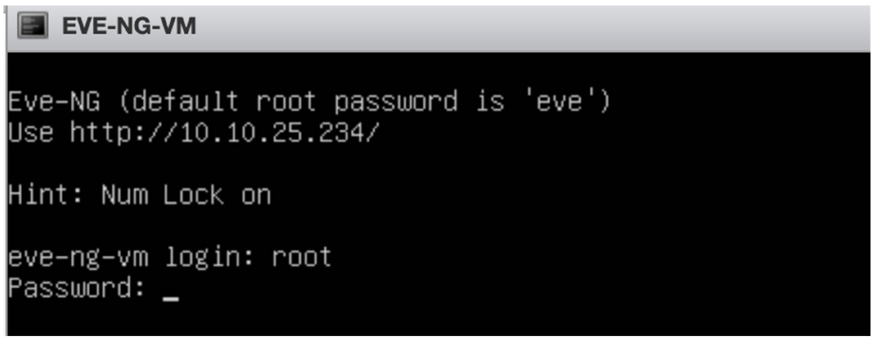
Hit return to use DHCP for getting Management IP address.



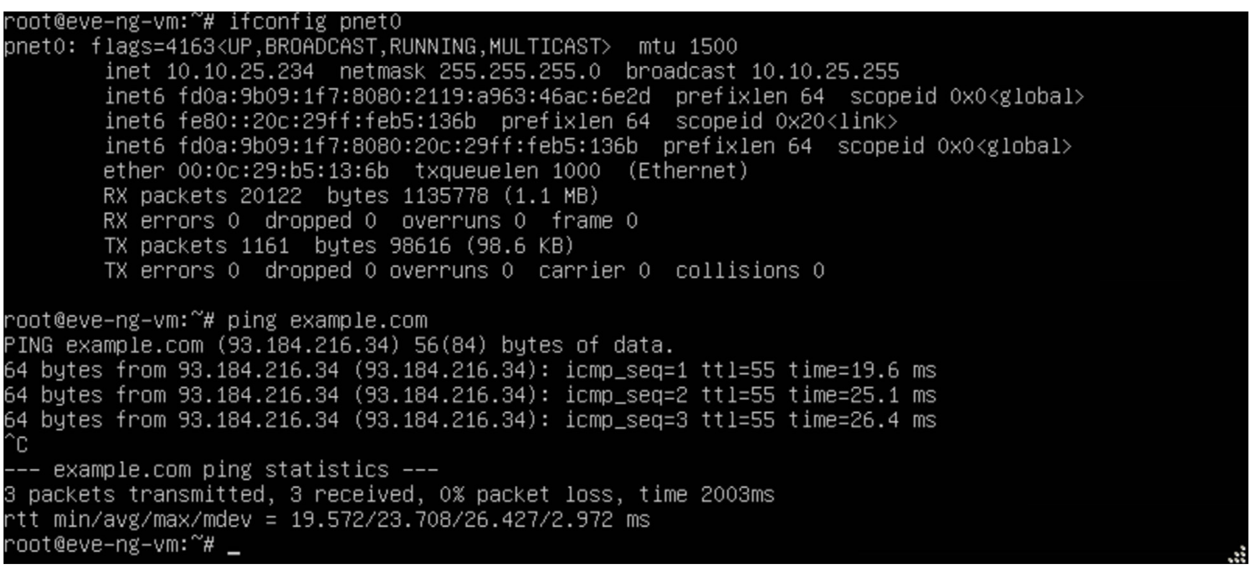
In this example we are not using any Proxy to reach the Internet. Hence we will choose direct connection and hit return.



It sets all the above configuration and reboots the *EVE-NG-VM*.



From the *EVE-NG-VM* console, enter the *username* as root and enter the newly set password earlier in this section. Verify the management IP address by executing the command *ifconfig pnet0*. You can see *EVE-NG-VM's* management IP address is 10.10.25.234. Verify whether *EVE-NG_VM* can access Internet by executing the command *ping example.com* as shown below.



Now let us access the *EVE-NG_VM* from the web browser using URL http://10.10.25.234 and verify login using default credentials: *admin/eve*

2. Install EVE-NG Client side pack

EVE-NG Client Side pack that will install everything necessary for running telnet, vnc and wireshark when working on Building labs.

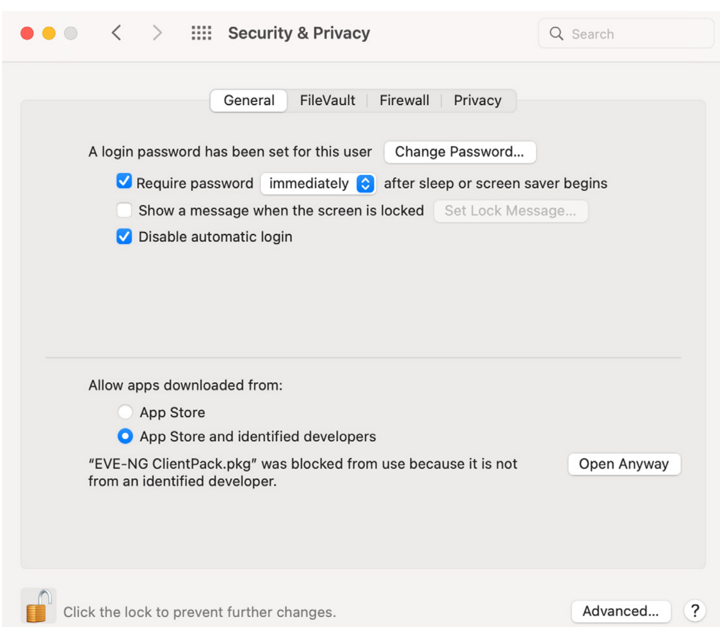
The following are steps to install a EVE-NG Client side pack:

a. Download the EVE-NG Client Side pack based on your laptop type.

   i. Windows Version

   ii. MacOS Version. In this example MacOS laptop is used.

c. Install the EVE-NG side pack on your laptop. If you get permission error on MacOS laptop, do the following: *Open System Preferences -> Security & Privacy -> General* and click *Open Anyway*.



3. Install Linux Ubuntu 21.04 Server in the EVE-NG

The following are steps to install a Linux Ubuntu 22.04 server in EVE-NG .

a. Download the Linux Ubuntu 22.04 server image from here to your laptop.

b. Copy the Linux server image from your laptop to EVE-NG VM as follows. You can also copy file using WinSCP or FileZilla:

MacBook-Pro Downloads % scp linux-ubuntu-22.04-server.tar.gz root@10.10.25.234:/opt/unetlab/addons/qemu/
root@10.10.25.234's password:
linux-ubuntu-22.04-server.tar.gz 100% 1223MB 109.3MB/s 00:11

Hardware requirement of installing Ubuntu-21.04(Linux) server:
1) Physical Device(PC/Laptop) : 8GB RAM
2) EVE-NG VM : 4GB RAM
3) CPU Processors : 2 Nos

c. Log into EVE-NG and execute following commands to install the Ubuntu Server in EVE-NG:
cd /opt/unetlab/addons/qemu/
tar xzvf linux-ubuntu-22.04-server.tar.gz
/opt/unetlab/wrappers/unl_wrapper -a fixpermissions
Verify management IP address using the following command:

root@eve-ng-vm:~# *ip addr show pnet0 | grep "scope global pnet0"*

inet 10.10.25.234/24 brd 10.10.25.255 scope global pnet0

Ubuntu 22.04 Login Credentials:
Username: *user*
Password: *Test123*

4. Install OcNOS VM in the EVE-NG

The following are steps to install the OcNOS VM in the EVE-NG :

a. Copy the *ocnos.yml* template file to EVE-NG as follows. You can also copy file using WinSCP or FileZilla.

MacBook-Pro EVE_NG % *ocnos.yml root@10.10.25.234:/opt/unetlab/html/templates/intel/*
root@10.10.25.234's password:
ocnos.yml 100% 558 11.8KB/s 00:00

b. Copy the OcNOS.png icon picture to EVE-NG as follows:

MacBook-Pro EVE_NG % *scp OcNOS.png root@10.10.25.234:/opt/unetlab/html/images/icons/*
root@10.10.25.234's password:
OcNOS.png 100% 3619 66.3KB/s 00:00

c. Copy the OcNOS VM image to EVE-NG as follows:

SSH into EVE-NG and execute following commands for copying OcNOS-VM image to the EVE-NG:
cd /opt/unetlab/addons/qemu/
mkdir ocnos-SP-MPLS-x86-6.3.0-126-GA

Please Note: Name of the directory need to start with the same name phrase associated with the Template file name. "ocnos" prefix is used in this example.

From your laptop copy the downloaded OcNOS VM image (uncomoressed version) to EVE-NG as follows:

MacBook-Pro VM % *scp OcNOS-SP-MPLS-x86-6.3.0-126-GA.qcow2*
root@10.10.25.234:/opt/unetlab/addons/qemu/OcNOS-SP-MPLS-x86-6.3.0-126-GA/
root@10.10.25.234's password:
OcNOS-SP-MPLS-x86-6.3.0-126-GA.qcow2 100% 3331MB 2.6MB/s 21:44

Once the image is copied into the folder, it must be renamed to 'virtioa.qcow2' as per EVE-NGs naming convention.

SSH into EVE-NG and execute following commands:

root@eve-ng-vm:~# *cd /opt/unetlab/addons/qemu/OcNOS-SP-MPLS-x86-6.3.0-126-GA*
root@eve-ng-vm:/opt/unetlab/addons/qemu/OcNOS-SP-MPLS-x86-6.3.0-126-GA# *mv OcNOS-SP-MPLS-x86-6.3.0-126-GA.qcow2 virtioa.qcow2*

Fix permissions with following command:

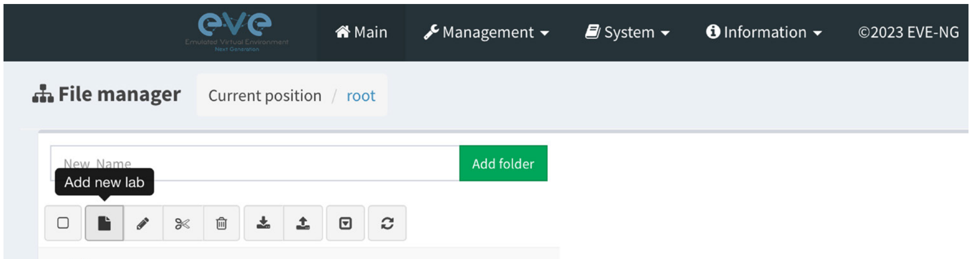
/opt/unetlab/wrappers/unl_wrapper -a fixpermissions

5. Set up BGP and L3 VPN Lab on the EVE-NG

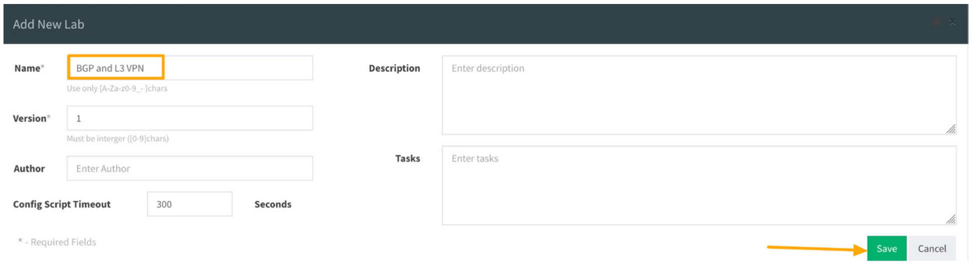
The following are steps to set up *BGP and L3 VPN Lab* in EVE-NG:

a. **Login to EVE-NG Web UI by accessing the *EVE-NG_VM* from the web browser using URL http://10.10.25.234 and verify login using default credentials: *admin/eve***

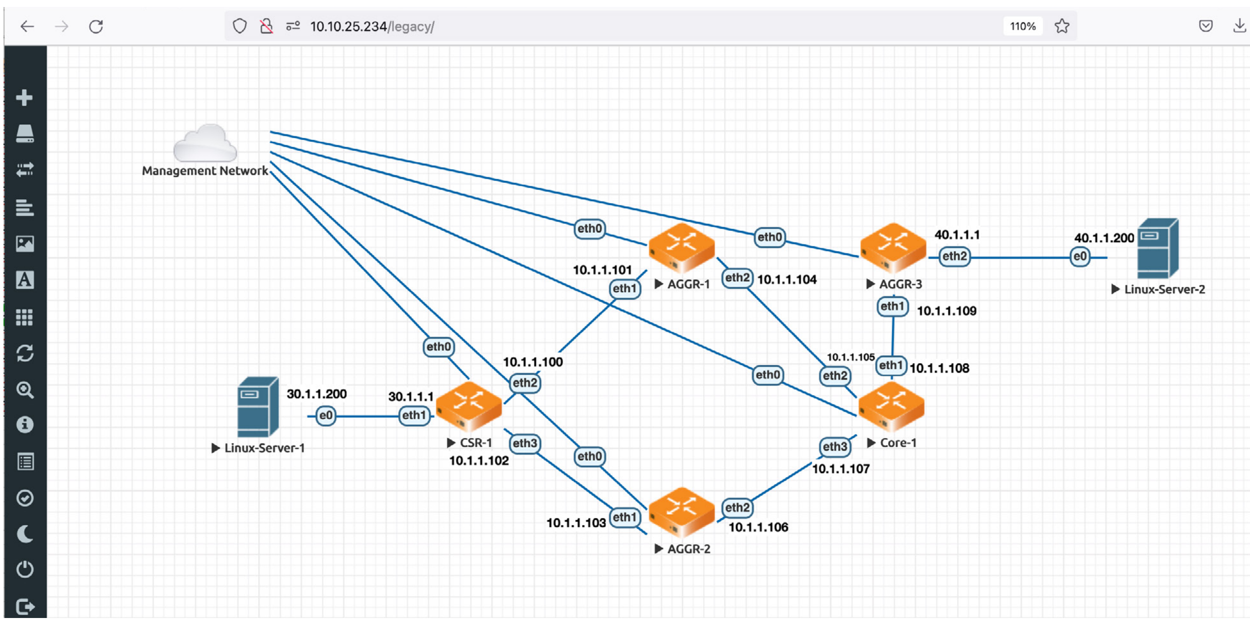
b. **Create a new lab** called *BGP and L3 VPN* as shown below.



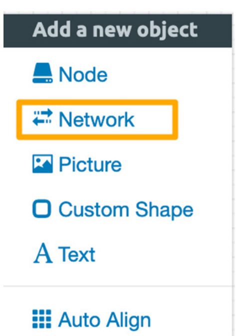
Click *Add new lab* icon enter the new lab *Name* as shown below and click Save.



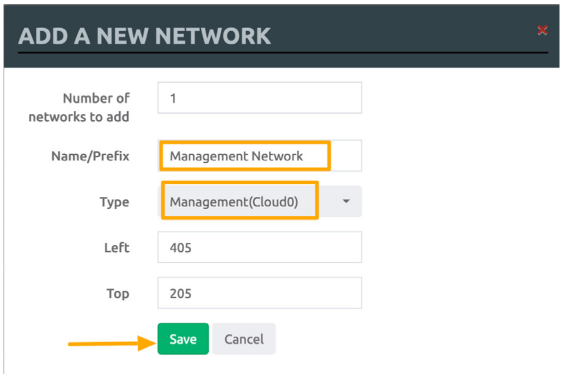
Following is the topology we are going to setup in the *BGP and L3* VPN lab.



c. **Add Management Network:** Right click on the new Lab page and select *Network* as shown below.



Enter *Name* of the network and select *Management Network Type* and click *Save* as shown below.

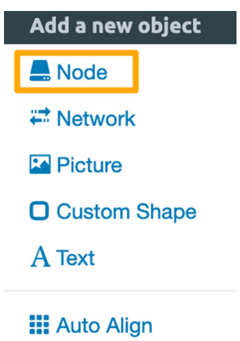


You will see Management Network (cloud) added to the lab.

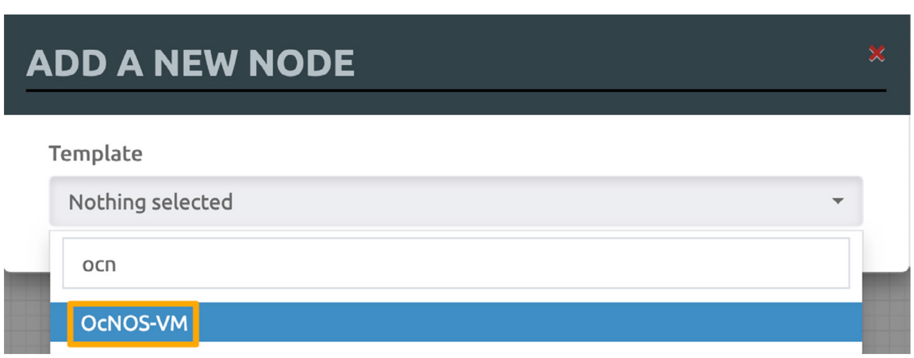
d. **Set up five OcNOS Switches as shown in the topology below:**

Following are steps to set up five OcNOS switches as part of *BGP and L3 VPN Lab* in EVE-NG.

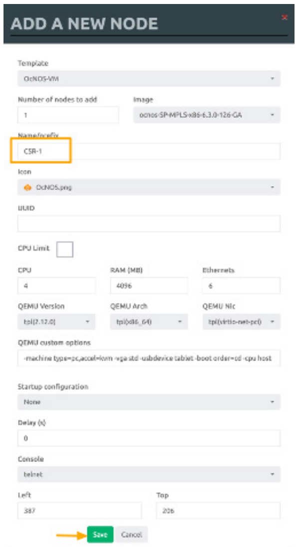
i. **Set up first OcNOS node**: Right click on the new Lab page and select Node as shown below.



Select OcNOS-VM as shown below:

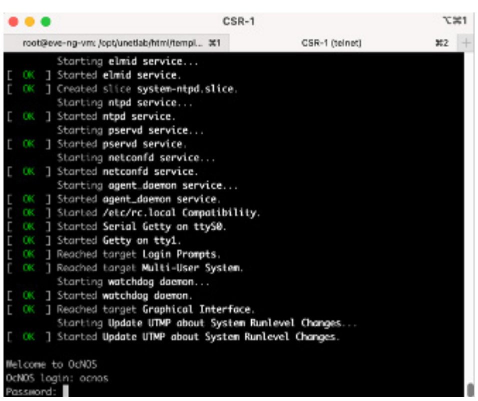


Enter the Name as *CSR-1* and click *Save*.



Right click on the CSR-1 device and click *Start* as shown below.

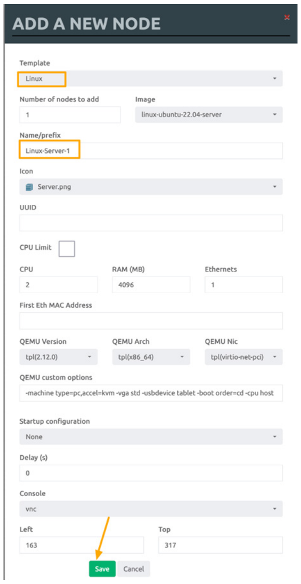
Double click on the CSR-1 device to open telnet console.



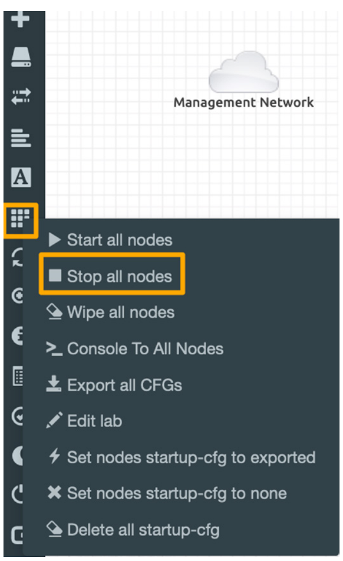
`Following are default credentials to log into the console of any of the OcNOS switches:` *`ocnos/ocnos`*

ii. **Set up four more OcNOS switches**: Setup four more OcNOS switches as shown in the Topology picture given above by repeating steps mentioned in the item (i) for creating each switch.

iii. **Set up two Linux servers**: Right click on the Lab page, select *Node*, select *Template Linux*, provide a unique server name and click *Save*. Right click on the new server and click *Start*. Double click on the one the device to open the VNC console. Following are default credentials for login: Username: *user* and Password: *Test123*. Repeat these steps to create the second Linux server.

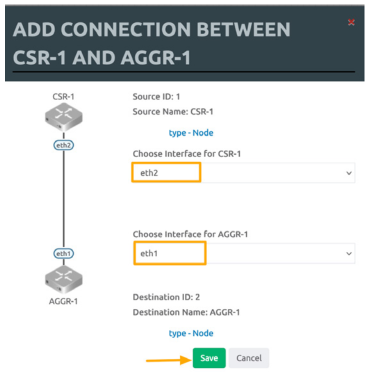


iv. **Stop all the nodes and make connections as shown above:** From the left menu click on the *More actions* and select *Stop all nodes*.

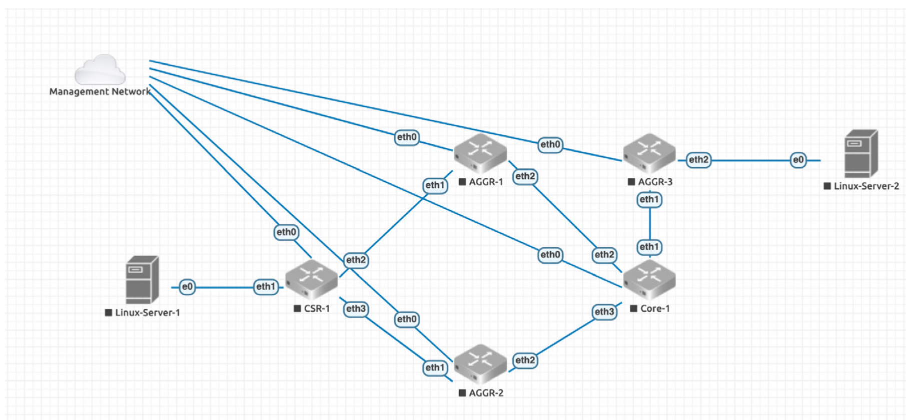


v. **Setup data plane connections between the OcNOS switches and Linux servers:** Hover over the device you want to connect, it will show a power plug sign as shown below. Right click over the Power Plug and drag it to the other device you want to connect.

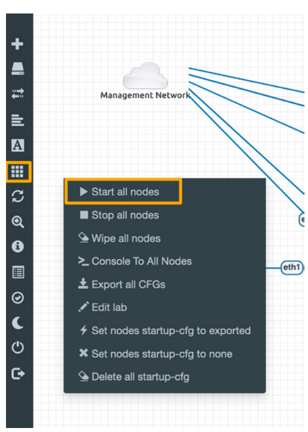
In the dialog box, select the interface you want to connect in each device and click *Save* as shown below.



Connect all the devices as shown below:



vi. **Start all devices:** From the left menu click on the *More actions* and select *Start all nodes.*



vii. **Deploy configuration in each switch each:** Double click on each switch icon to access console. Enter into enable and configuration modes. Extract switches configuration files from the ***BGP-L3VPN-switches-config.zip*** file you have downloaded earlier.

Perform the following commands on each switch after login:

```
CSR-1> en
CSR-1> conf t
```

Copy corresponding switch file configuration and paste it on the switch (for example: copy CSR-1.txt file and paste it on CSR-1) switch console in configuration mode and commit the configuration.

```
CSR-1> en
CSR-1# conf t
CSR-1(config)# <paste the config>
CSR-1(config)# commit
```

Perform the following command to copy the configuration to persistent memory in the switch.

```
CSR-1# copy running-config startup-config
Building Configuration...
[OK]
```

viii. **Configure the Linux servers to setup for Traffic Testing:** Double click on the first server to open VNC console session to the server and login to the server.

```
root@ubuntu22-server# cd /etc/netplan
root@ubuntu22-server# su
```

Enter the password and edit the following file:

```
root@ubuntu22-server# vi 00-installer-config.yaml
```

Update the content of the file as follows and save the file:

```
root@ubuntu22-server:/etc/netplan# more 00-installer-config.yaml
# This is the network config written by 'subiquity'
network:
  ethernets:
    ens3:
      dhcp4: no
      addresses: [30.1.1.200/24]
      gateway4: 30.1.1.1
  version: 2
```

```
root@ubuntu22-server# sudo netplan apply
```

Next change the name of the server as follows. Edit */etc/hostname* file using *vi* editor and change contents to *Linux-Server-1* and save the file. Reboot the server to make changes permanent.

```
root@ubuntu22-server# reboot
```

Similarly, double click on the second server to open VNC console session to the server and login to the server. Set the IP address of *ens3* interface to *40.1.1.200* using commands shown above. Set the host name of second server to Linux-Server-2. Reboot the server.

6. **Verify** BGP and L3 VPN **Lab**

We will run several commands to verify BGP and L3 VPN functionalities.

a. **Generate Test Traffic:** Log into the console of the *Linux-Server-1* and execute the following Linux shell command to send 1000 packets from the *Linux-Server-1* Server to the *Linux-Server-2* on the TEST_VRF.

```
debian@debian:~$ ping -c 1000 -i 1 40.1.1.200
PING 40.1.1.200 (40.1.1.200) 56(84) bytes of data.
64 bytes from 40.1.1.200: icmp_seq=1 ttl=63 time=4.15 ms
64 bytes from 40.1.1.200: icmp_seq=2 ttl=63 time=4.84 ms
64 bytes from 40.1.1.200: icmp_seq=3 ttl=63 time=5.45 ms
64 bytes from 40.1.1.200: icmp_seq=4 ttl=63 time=3.56 ms
64 bytes from 40.1.1.200: icmp_seq=5 ttl=63 time=3.63 ms
...
```

b. **Check summary of known neighbor:** Log into the console of the CSR-1 OcNOS virtual switch (or SSH into CSR-1) and run the following commands to verify the BGP and L3 VPN functionalities. The show clns neighbors command provides a summary of known neighbors, the connecting interface, and the state of the adjacency.

```
CSR-1#show clns neighbors

Total number of L1 adjacencies: 2
Total number of L2 adjacencies: 0
Total number of adjacencies: 2
Tag 1: VRF : default
System Id  Interface  SNPA            State     Holdtime   Type  Protocol
AGGR-1     eth2       0cc6.74db.0001  Up        6          L1    IS-IS
AGGR-2     eth3       0c2c.0e08.0001  Up        27         L1    IS-IS
```

c. **Check TEST_VRF forwarding table:** Following output shows we have path to reach the second server.

```
CSR-1# show mpls vrf-forwarding-table vrf TEST_VRF

CSR-1>CSR-1>show mpls vrf-forwarding-table vrf TEST_VRF
Owner    FEC          FTN-ID    Oper-Status   Out-Label    Tunnel-id    NHLFE-id   Out-Intf    Nexthop
BGP      40.1.1.0/24  1         Up            25600        0            5          eth2        10.1.1.5
```

Also check Incoming Label Map entries. Use the following command to view Incoming label mapping (ILM) table entries

```
CSR-1#show mpls ilm-table
Codes: > - installed ILM, * - selected ILM, p - stale ILM
       K - CLI ILM, T - MPLS-TP, s - Stitched ILM
       S - SNMP, L - LDP, R - RSVP, C - CRLDP
       B - BGP , K - CLI , V - LDP_VC, I - IGP_SHORTCUT
       O - OSPF/OSPF6 SR, i - ISIS SR, k - SR CLI
       P - SR Policy, U - unknown

Code    FEC/VRF/L2CKT    ILM-ID   In-Label  Out-Label  In-Intf   Out-Intf/VRF   Nexthop     LSP-Type
L>      10.1.1.106/31    11       24965     3          N/A       eth3           10.1.1.103  LSP_DEFAULT
L>      10.1.1.3/32      7        24961     3          N/A       eth3           10.1.1.103  LSP_DEFAULT
B>      TEST_VRF         1        24320     Nolabel    N/A       TEST_VRF       N/A         LSP_DEFAULT
L>      10.1.1.2/32      13       24967     3          N/A       eth2           10.1.1.101  LSP_DEFAULT
L>      10.1.1.104/31    14       24968     3          N/A       eth2           10.1.1.101  LSP_DEFAULT
```

d. **Check for path to AGGR-3 in MPLS forwarding Table:** Run the following command in CSR-1.

```
CSR-1#show mpls forwarding-table
Codes:    > - installed FTN, * - selected FTN, p - stale FTN,
          B - BGP FTN, K - CLI FTN, t - tunnel, P - SR Policy FTN,
          L - LDP FTN, R - RSVP-TE FTN, S - SNMP FTN, I - IGP-Shortcut,
          U - unknown FTN, O - SR-OSPF FTN, i - SR-ISIS FTN, k - SR-CLI FTN

Code  FEC          FTN-ID  Nhlfe-ID  Tunnel-id  Pri  LSP-Type     Out-Label Out-Intf ELC  Nexthop
L>    10.1.1.2/32    1       32         -       Yes  LSP_DEFAULT  3          eth2     No   10.1.1.101
L>    10.1.1.3/32    2       14         -       Yes  LSP_DEFAULT  3          eth3     No   10.1.1.103
L>    10.1.1.4/32    3       16         -       Yes  LSP_DEFAULT  24962      eth3     No   10.1.1.103
                             48         -       Yes  LSP_DEFAULT  24962      eth2     No   10.1.1.101
L>    10.1.1.5/32    4       20         -       Yes  LSP_DEFAULT  24963      eth3     No   10.1.1.103
                             49         -       Yes  LSP_DEFAULT  24963      eth2     No   10.1.1.101
L>    10.1.1.104/31  5       32         -       Yes  LSP_DEFAULT  3          eth2     No   10.1.1.101
L>    10.1.1.106/31  6       14         -       Yes  LSP_DEFAULT  3          eth3     No   10.1.1.103
L>    10.1.1.108/31  7       28         -       Yes  LSP_DEFAULT  24965      eth3     No   10.1.1.103
                             50         -       Yes  LSP_DEFAULT  24966      eth2     No   10.1.1.101
```

You can see AGGR-5 can be reached via eth2 and eth3.

e. **Check LDP sessions in CSR-1:** Execute the following CLI in CSR-1.

```
CSR-1#show ldp session
Peer IP Address      IF Name My     Role        State         KeepAlive     UpTime
10.1.1.2             eth2           Passive     OPERATIONAL   30            22:24:09
10.1.1.3             eth3           Passive O   PERATIONAL    30            22:24:09
```

f. **Check route between two Debian Servers:** Check the route from one Debian Server to other using the following command:
One server is directly connected to 30.1.1.0/24 network and other server in 40.1.1.0/24 network is accessible via BGP.

```
CSR-1#show ip route vrf TEST_VRF database
Codes: K - kernel, C - connected, S - static, R - RIP, B - BGP
       O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2,
       ia - IS-IS inter area, E - EVPN,
       v - vrf leaked
       > - selected route, * - FIB route, p - stale info

IP Route Table for VRF "TEST_VRF"
C      *> 30.1.1.0/24 is directly connected, eth1, 1d07h49m
B      *> 40.1.1.0/24 [200/0] via 10.1.1.5, 00:20:26

Gateway of last resort is not set
```

g. **Check L3VPN routes:** Use the following command to display information relating to MPLS VPN.

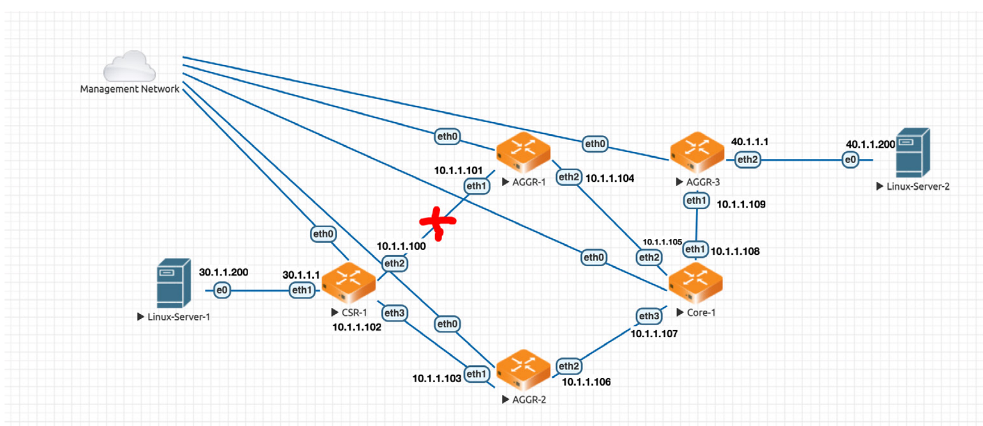
```
CSR-1#show ip bgp vpnv4 all summary
BGP router identifier 10.1.1.1, local AS number 65000
BGP table version is 9
1 BGP AS-PATH entries
0 BGP community entries

Neighbor V    AS      MsgRcv  MsgSen  TblVer  InQ     OutQ    Up/Down State/PfxRcd
10.1.1.2  4   65000 4446     4444    9       0       0       00:20:32       1
10.1.1.3  4   65000 4420     4418    9       0       0       00:20:37       1

Total number of neighbors 2

Total number of Established sessions 2
```

h. **Stop flow of traffic between CSR-1 and AGGR-1 and verify whether traffic flows from one server to the other:**

When the ICMP traffic is flowing, let us stop the traffic between the ***CSR-1*** and the ***AGGR-1***. To do this perform the following CLI commands in *CSR-1* Switch.

```
CSR-1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
CSR-1(config)#int eth2
CSR-1(config-if)#shut
CSR-1(config-if)#commit
```

This will stop the traffic flowing between *CSR-1* and *AGGR-1*. Now traffic will not go through *eth2* interface. Traffic will only go through *eth3* interface.




```
Check the traffic flow using the following command in CSR-1.
CSR-1#show mpls forwarding-table
Codes: > - installed FTN, * - selected FTN, p - stale FTN,
       B - BGP FTN, K - CLI FTN, t - tunnel, P - SR Policy FTN,
       L - LDP FTN, R - RSVP-TE FTN, S - SNMP FTN, I - IGP-Shortcut,
       U - unknown FTN, O - SR-OSPF FTN, i - SR-ISIS FTN, k - SR-CLI FTN
```

```
Code  FEC           FTN-ID      Nhlfe-ID  Tunnel-id  Pri   LSP-Type     Out-Label  Out-Intf  ELC    Nexthop
L>    10.1.1.2/32      1        10            -      Yes   LSP_DEFAULT  24961      eth3      No    10.1.1.103
L>    10.1.1.3/32      2        14            -      Yes   LSP_DEFAULT      3      eth3      No    10.1.1.103
L>    10.1.1.4/32      3        16            -      Yes   LSP_DEFAULT  24962      eth3      No    10.1.1.103
L>    10.1.1.5/32      4        20            -      Yes   LSP_DEFAULT  24963      eth3      No    10.1.1.103
L>    10.1.1.104/31    5        27            -      Yes   LSP_DEFAULT  24964      eth3      No    10.1.1.103
L>    10.1.1.106/31    6        14            -      Yes   LSP_DEFAULT      3      eth3      No    10.1.1.103
L>    10.1.1.108/31    7        28            -      Yes   LSP_DEFAULT  24965      eth3      No    10.1.1.103
```

i. Verify whether traffic can reach AGGR-3 with MPLS ping:

```
CSR-1#ping mpls ldp 10.1.1.5/32 detail
Sending 5 MPLS Echos to 10.1.1.5, timeout is 5 seconds

Codes:
'!' - Success, 'Q' - request not sent, '.' - timeout,
'x' - Retcode 0, 'M' - Malformed Request, 'm' - Errored TLV,
'N' - LBL Mapping Err, 'D' - DS Mismatch,
'U' - Unknown Interface, 'R' - Transit (LBL Switched),
'B' - IP Forwarded, 'F' No FEC Found, 'f' - FEC Mismatch,
'P' - Protocol Error, 'X' - Unknown code,
'Z' - Reverse FEC Validation Failed

Type 'Ctrl+C' to abort

!      seq_num  =    1     10.1.1.109 1.92  ms
!      seq_num  =    2     10.1.1.109 1.01  ms
!      seq_num  =    3     10.1.1.109 1.26  ms
!      seq_num  =    4     10.1.1.109 1.63  ms
!      seq_num  =    5     10.1.1.109 2.52  ms

Success Rate is 100.00 percent (5/5)
round-trip min/avg/max = 1.01/1.77/2.52
```

References

OcNOS

The following are reference materials related to OcNOS:

- OcNOS Configuration Guides

EVE-NG

The following are reference materials related to EVE-NG:

- Getting Started with EVE-NG

Appendix-A - Example BGP and L3 VPN Configuration Used in the EVE-NG Environment

The following example configurations are used in the EVE-NG environment to test BGP and L3 VPN functionality in OcNOS virtual switches.

CSR-1 Switch Configuration

The configuration used in the CSR-1 OcNOS virtual switch is given below:

```
!
no service password-encryption
!
logging console 2
logging monitor 7
logging cli
!
ip vrf management
!
ip vrf TEST_VRF
   rd 10.1.1.1:1
   route-target both 65000:1
!
hostname CSR-1
ip domain-lookup
feature telnet
feature ssh
feature rsyslog
!
router ldp
   router-id 10.1.1.1
   transport-address ipv4 10.1.1.1
!
!
interface lo
   ip address 127.0.0.1/8
   ip address 10.1.1.1/32 secondary
   ipv6 address ::1/128
   ip router isis 1
!
interface eth0
   ip vrf forwarding management
   ip address dhcp
!
interface eth1
   ip vrf forwarding TEST_VRF
   ip address 30.1.1.1/24
!
interface eth2
   ip address 10.1.1.100/31
   label-switching
   mpls ldp-igp sync isis level-1
```

```
    isis network point-to-point
    ip router isis 1
    enable-ldp ipv4
    lldp-agent
    set lldp enable txrx
    exit
!
interface eth3
    ip address 10.1.1.102/31
    label-switching
    mpls ldp-igp sync isis level-1
    isis network point-to-point
    ip router isis 1
    enable-ldp ipv4
    lldp-agent
    set lldp enable txrx
    exit
!
interface eth4
!
exit
!
router isis 1
    is-type level-1
    metric-style wide level-1
    mpls traffic-eng router-id 10.1.1.1
    mpls traffic-eng level-1
    capability cspf
    dynamic-hostname
    bfd all-interfaces
    net 49.0111.1100.0075.0001.00

!
router bgp 65000
    bgp router-id 10.1.1.1
    neighbor 10.1.1.2 remote-as 65000
    neighbor 10.1.1.3 remote-as 65000
    neighbor 10.1.1.2 update-source lo
    neighbor 10.1.1.3 update-source lo
!
address-family vpnv4 unicast
    neighbor 10.1.1.2 activate
    neighbor 10.1.1.3 activate
    exit-address-family
!
address-family ipv4 vrf TEST_VRF
redistribute connected
exit-address-family
!
line vty 0
    exec-timeout 0 0
!
!
end
```

AGGR-1 Switch Configuration

The configuration used in the AGGR-1 OcNOS virtual switch is given below:

```
!
no service password-encryption
!
logging console 2
logging monitor 7
logging cli
!
ip vrf management
!
hostname AGGR-1
!
router ldp
   router-id 10.1.1.2
   transport-address ipv4 10.1.1.2
!
!
interface lo
   ip address 127.0.0.1/8
   ip address 10.1.1.2/32 secondary
   ipv6 address ::1/128
   ip router isis 1
!
interface eth0
   ip vrf forwarding management
   ip address dhcp
!
interface eth1
   ip address 10.1.1.101/31
   label-switching
   mpls ldp-igp sync isis level-1
   isis network point-to-point
   ip router isis 1
   enable-ldp ipv4
   lldp-agent
   set lldp enable txrx
   exit
!
interface eth2
   ip address 10.1.1.104/31
   label-switching
   mpls ldp-igp sync isis level-1
   isis network point-to-point
   ip router isis 1
   enable-ldp ipv4
   lldp-agent
   set lldp enable txrx
   exit
!
interface eth3
!
interface eth4
```

```
!
exit
!
router isis 1
   is-type level-1
   metric-style wide level-1
   mpls traffic-eng router-id 10.1.1.2
   mpls traffic-eng level-1
   capability cspf
   dynamic-hostname
   bfd all-interfaces
   net 49.0111.1100.0075.0002.00
!
router bgp 65000
   no bgp inbound-route-filter
   bgp router-id 10.1.1.2
   neighbor 10.1.1.1 remote-as 65000
   neighbor 10.1.1.3 remote-as 65000
   neighbor 10.1.1.4 remote-as 65000
   neighbor 10.1.1.5 remote-as 65000
   neighbor 10.1.1.1 update-source lo
   neighbor 10.1.1.3 update-source lo
   neighbor 10.1.1.4 update-source lo
   neighbor 10.1.1.5 update-source lo
!
address-family vpnv4 unicast
   neighbor 10.1.1.1 activate
   neighbor 10.1.1.1 route-reflector-client
   neighbor 10.1.1.3 activate
   neighbor 10.1.1.4 activate
   neighbor 10.1.1.4 route-reflector-client
   neighbor 10.1.1.5 activate
   neighbor 10.1.1.5 route-reflector-client
   exit-address-family
!
line vty 0
   exec-timeout 0 0
!
!
end
```

AGGR-2 Switch Configuration

The configuration used in the AGGR-2 OcNOS virtual switch is given below:

```
!
no service password-encryption
!
logging console 2
logging monitor 7
logging cli
!
ip vrf management
!
hostname AGGR-2
!
router ldp
   router-id 10.1.1.3
   transport-address ipv4 10.1.1.3
!
!
interface lo
   ip address 127.0.0.1/8
   ip address 10.1.1.3/32 secondary
   ipv6 address ::1/128
   ip router isis 1
!
interface eth0
   ip vrf forwarding management
   ip address dhcp
!
interface eth1
   ip address 10.1.1.103/31
   label-switching
   mpls ldp-igp sync isis level-1
   isis network point-to-point
   ip router isis 1
   enable-ldp ipv4
   lldp-agent
   set lldp enable txrx
   exit
!
interface eth2
   ip address 10.1.1.106/31
   label-switching
   mpls ldp-igp sync isis level-1
   isis network point-to-point
   ip router isis 1
   enable-ldp ipv4
   lldp-agent
   set lldp enable txrx
   exit
!
interface eth3
!
```

```
interface eth4
!
exit
!
router isis 1
   is-type level-1
   metric-style wide level-1
   mpls traffic-eng router-id 10.1.1.3
   mpls traffic-eng level-1
   capability cspf dynamic-hostname
   bfd all-interfaces
   net 49.0111.1100.0075.0003.00
!
router bgp 65000
   bgp router-id 10.1.1.3
   no bgp inbound-route-filter
   neighbor 10.1.1.1 remote-as 65000
   neighbor 10.1.1.2 remote-as 65000
   neighbor 10.1.1.4 remote-as 65000
   neighbor 10.1.1.5 remote-as 65000
   neighbor 10.1.1.1 update-source lo
   neighbor 10.1.1.2 update-source lo
   neighbor 10.1.1.4 update-source lo
   neighbor 10.1.1.5 update-source lo
!
address-family vpnv4 unicast
   neighbor 10.1.1.1 activate
   neighbor 10.1.1.1 route-reflector-client
   neighbor 10.1.1.2 activate
   neighbor 10.1.1.2 route-reflector-client
   neighbor 10.1.1.4 activate
   neighbor 10.1.1.4 route-reflector-client
   neighbor 10.1.1.5 activate
   neighbor 10.1.1.5 route-reflector-client
   exit-address-family
!
line vty 0
   exec-timeout 0 0
!
!
end
```

CORE-1 Switch Configuration

The configuration used in the CORE-1 OcNOS virtual switch is given below:

```
no service password-encryption
!
logging console 2
logging monitor 7
logging cli
!
ip vrf management
!
hostname core-1
!
router ldp
   router-id 10.1.1.4
   transport-address ipv4 10.1.1.4
!
interface lo
   ip address 127.0.0.1/8
   ip address 10.1.1.4/32 secondary
   ipv6 address ::1/128
   ip router isis 1
!
interface eth0
   ip vrf forwarding management
   ip address dhcp
!
interface eth1
   ip address 10.1.1.108/31
   label-switching
   mpls ldp-igp sync isis level-1
   isis network point-to-point
   ip router isis 1
   enable-ldp ipv4
   lldp-agent
   set lldp enable txrx
   exit
!
interface eth2
   ip address 10.1.1.105/31
   label-switching
   mpls ldp-igp sync isis level-1
   isis network point-to-point
   ip router isis 1
   enable-ldp ipv4
   lldp-agent
   set lldp enable txrx
   exit
!
interface eth3
   ip address 10.1.1.107/31
   label-switching
   mpls ldp-igp sync isis level-1
   ip router isis 1
```

```
    enable-ldp ipv4
    lldp-agent
    set lldp enable txrx
    exit
!
interface eth4
!
exit
!
router isis 1
    is-type level-1
    metric-style wide level-1
    mpls traffic-eng router-id 10.1.1.4
    mpls traffic-eng level-1
    capability cspf dynamic-hostname
    bfd all-interfaces
    net 49.0111.1100.0075.0004.00
!
router bgp 65000
    bgp router-id 10.1.1.4
    neighbor 10.1.1.2 remote-as 65000
    neighbor 10.1.1.3 remote-as 65000
    neighbor 10.1.1.5 remote-as 65000
    neighbor 10.1.1.2 update-source lo
    neighbor 10.1.1.3 update-source lo
    neighbor 10.1.1.5 update-source lo
!
address-family vpnv4 unicast
    neighbor 10.1.1.2 activate
    neighbor 10.1.1.3 activate
    neighbor 10.1.1.5 activate
    exit-address-family
!
line vty 0
    exec-timeout 0 0
!
!
end
```

AGGR-3 Switch Configuration

The configuration used in the AGGR-3 OcNOS virtual switch is given below:

```
!
no service password-encryption
!
logging console 2
logging monitor 7
logging cli
!
ip vrf management
!
ip vrf TEST_VRF
   rd 10.1.1.5:1
   route-target both 65000:1
!
hostname AGGR-3
ip domain-lookup
feature telnet
feature ssh
feature rsyslog
!
router ldp
   router-id 10.1.1.5
   transport-address ipv4 10.1.1.5
!
interface lo
   ip address 127.0.0.1/8
   ip address 10.1.1.5/32 secondary
   ipv6 address ::1/128
   ip router isis 1
!
interface eth0
   ip vrf forwarding management
   ip address dhcp
!
interface eth1
   ip address 10.1.1.109/31
   label-switching
   mpls ldp-igp sync isis level-1
   isis network point-to-point
   ip router isis 1
   enable-ldp ipv4
   lldp-agent
   set lldp enable txrx
   exit
!
interface eth2
   ip vrf forwarding TEST_VRF
   ip address 40.1.1.1/24
   lldp-agent
   set lldp enable txrx
   exit
!
```

```
interface eth3
!
interface eth4
!
exit
!
router isis 1
   is-type level-1
   metric-style wide level-1
   mpls traffic-eng router-id 10.1.1.5
   mpls traffic-eng level-1
   capability cspf
   dynamic-hostname
   bfd all-interfaces
   net 49.0111.1100.0075.0005.00
!
router bgp 65000
   bgp router-id 10.1.1.5
   neighbor 10.1.1.1 remote-as 65000
   neighbor 10.1.1.2 remote-as 65000
   neighbor 10.1.1.3 remote-as 65000
   neighbor 10.1.1.4 remote-as 65000
   neighbor 10.1.1.1 update-source lo
   neighbor 10.1.1.2 update-source lo
   neighbor 10.1.1.3 update-source lo
   neighbor 10.1.1.4 update-source lo
!
address-family vpnv4 unicast
   neighbor 10.1.1.1 activate
   neighbor 10.1.1.2 activate
   neighbor 10.1.1.3 activate
   neighbor 10.1.1.4 activate
   exit-address-family
!
address-family ipv4 vrf TEST_VRF
   redistribute connected
   exit-address-family
!
line vty 0
   exec-timeout 0 0
!
!
end
```

ABOUT IP INFUSION

IP Infusion is a leading provider of open network software and solutions for carriers, service providers and data center operators. Our solutions enable network operators to disaggregate their networks to accelerate innovation, streamline operations, and reduce Total Cost of Ownership (TCO). Network OEMs may also disaggregate network devices to expedite time to market, offer comprehensive services, and achieve carrier grade robustness. IP Infusion network software platforms have a proven track record in carrier-grade open networking with over 500 customers and over 10,000 deployments. IP Infusion is headquartered in Santa Clara, Calif., and is a wholly owned and independently operated subsidiary of ACCESS CO., LTD. Additional information can be found at **http://www.ipinfusion.com**

Phone | +1-877-699-3267 **Email |** sales@ipinfusion.com **Web |** www.ipinfusion.com