

Network - Assignment

Task 1: Set up two focal VMs and assign IPs from the same network range and check if ping works.

Method 1: Static IP's Assigning

Step1: Install Two virtual Machines . In my case I have installed Linux Ubuntu Focal machines named VM1 and VM2 and set up a NAT adapter and an Internal Network adapter.

Step 2 : We need to Setup a Static IP address for the machines such that on reboot or on after any particular time. So , for that go into the `/etc/network/interfaces` file.

Use the command `sudo nano /etc/network/interfaces` or via netplan by using `sudo nano /etc/netplan/<yaml file>`

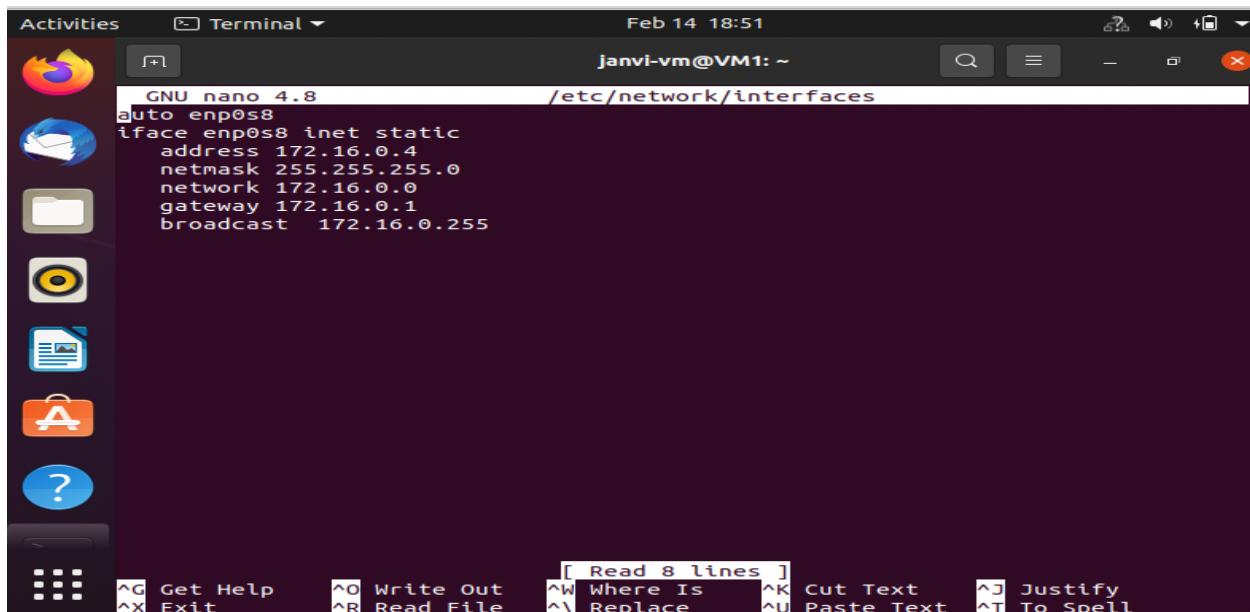
The netplan will have the configuration like

```
ethernets:  
  enp0s3:  
    addresses:  
      - 172.16.0.1/24  
    dhcp4: no
```

For network interface ,Write the configuration in that file

```
auto enp0s3  
iface enp0s3 inet static  
address <ip-address>  
netmask <subnet-mask>  
gateway <default-gateway>  
broadcast <broadcast-point>  
network <network>
```

For me it looks like :



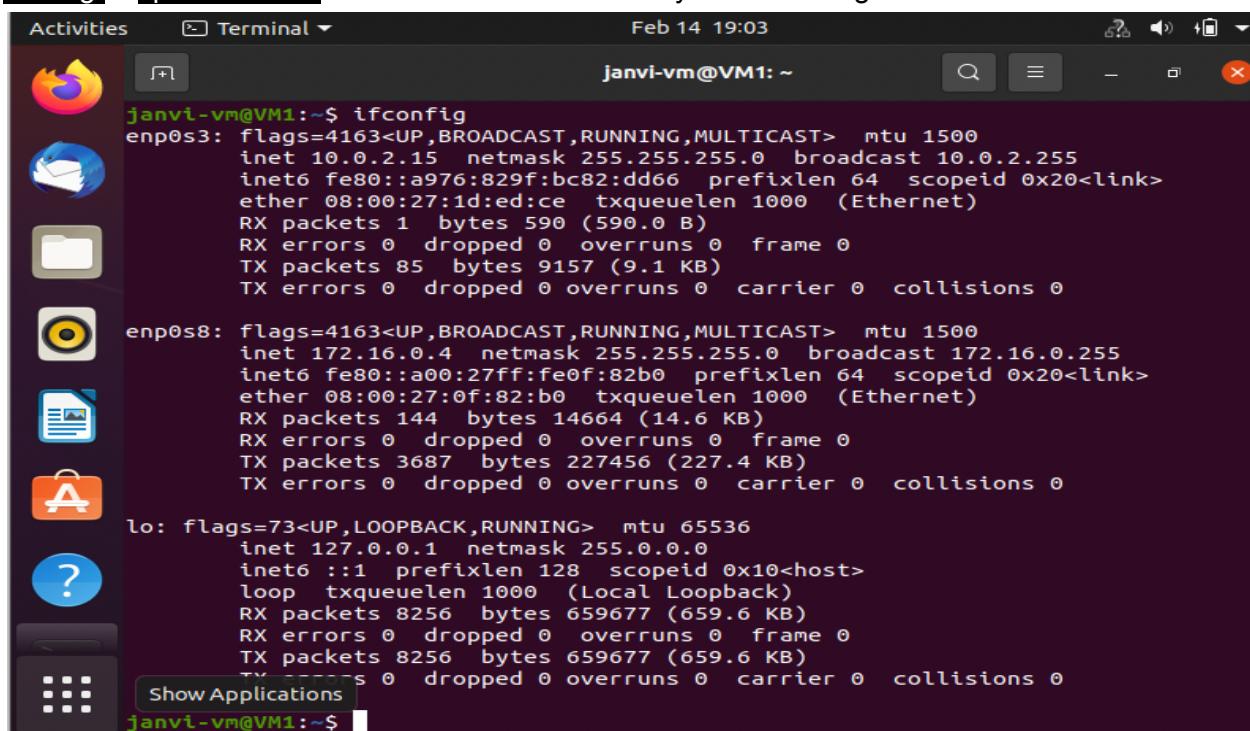
```
Activities Terminal Feb 14 18:51
janvi-vm@VM1: ~
GNU nano 4.8 /etc/network/interfaces
auto enp0s8
iface enp0s8 inet static
    address 172.16.0.4
    netmask 255.255.255.0
    network 172.16.0.0
    gateway 172.16.0.1
    broadcast 172.16.0.255

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify
^X Exit      ^R Read File  ^\ Replace ^U Paste Text ^T To Spell
```

Save and exit this file .

Step 3 : Reboot both the machines using sudo reboot.

Step 4 : After the machine is done with the reboot , Open the terminal and type in the command `ifconfig` or `ip addr show` . You'll see the IP address you have assigned to it.



```
Activities Terminal Feb 14 19:03
janvi-vm@VM1: ~
janvi-vm@VM1:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 10.0.2.15 netmask 255.255.255.0 broadcast 10.0.2.255
        inet6 fe80::a976:829f:bc82:dd66 prefixlen 64 scopeid 0x20<link>
              ether 08:00:27:1d:ed:ce txqueuelen 1000 (Ethernet)
              RX packets 1 bytes 590 (590.0 B)
              RX errors 0 dropped 0 overruns 0 frame 0
              TX packets 85 bytes 9157 (9.1 KB)
              TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 172.16.0.4 netmask 255.255.255.0 broadcast 172.16.0.255
        inet6 fe80::a00:27ff:fe0f:82b0 prefixlen 64 scopeid 0x20<link>
              ether 08:00:27:0f:82:b0 txqueuelen 1000 (Ethernet)
              RX packets 144 bytes 14664 (14.6 KB)
              RX errors 0 dropped 0 overruns 0 frame 0
              TX packets 3687 bytes 227456 (227.4 KB)
              TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
        inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
              loop txqueuelen 1000 (Local Loopback)
              RX packets 8256 bytes 659677 (659.6 KB)
              RX errors 0 dropped 0 overruns 0 frame 0
              TX packets 8256 bytes 659677 (659.6 KB)
              TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

janvi-vm@VM1:~$
```

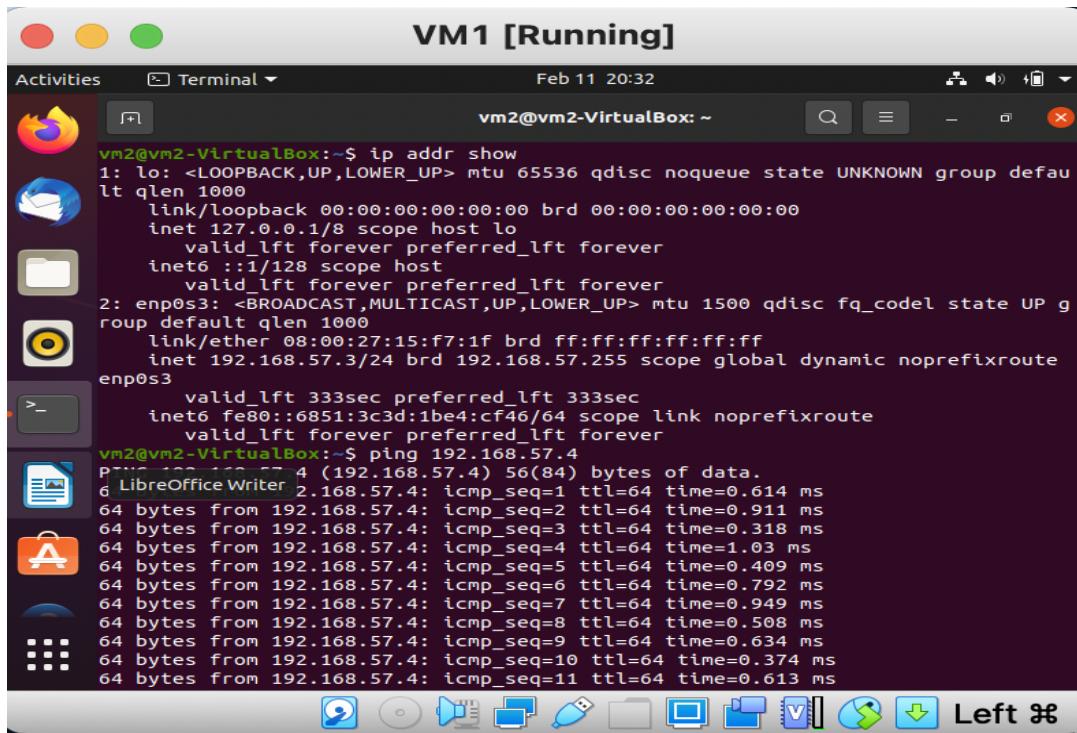
You can reboot your machine as many times as you want now . This IP address is going to remain static.

Repeat the Process for The Virtual Machine 2.

In the similar way i have created two virtual machines having nat network , provided the static IP through the netplan or network interface method and tried to reboot he machines

Now after rebooting when you'll run the command id addr show , both the machines will be having the same Ip that we provided.

You can actually see it in the below screenshots. Also , they are connecting with each other via the ping command. So the ping command working looks fine.



The screenshot shows a Linux desktop environment with a terminal window titled "VM1 [Running]". The terminal window displays the following command and its output:

```
vm2@vm2-VirtualBox:~$ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defau
lt qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP g
roup default qlen 1000
    link/ether 08:00:27:15:f7:1f brd ff:ff:ff:ff:ff:ff
    inet 192.168.57.3/24 brd 192.168.57.255 scope global dynamic noprefixroute
        enp0s3
            valid_lft 333sec preferred_lft 333sec
            inet6 fe80::6851:3c3d:1be4:cf46/64 scope link noprefixroute
                valid_lft forever preferred_lft forever
vm2@vm2-VirtualBox:~$ ping 192.168.57.4
PING 192.168.57.4 (192.168.57.4) 56(84) bytes of data.
64 bytes from 192.168.57.4: icmp_seq=1 ttl=64 time=0.614 ms
64 bytes from 192.168.57.4: icmp_seq=2 ttl=64 time=0.911 ms
64 bytes from 192.168.57.4: icmp_seq=3 ttl=64 time=0.318 ms
64 bytes from 192.168.57.4: icmp_seq=4 ttl=64 time=1.03 ms
64 bytes from 192.168.57.4: icmp_seq=5 ttl=64 time=0.409 ms
64 bytes from 192.168.57.4: icmp_seq=6 ttl=64 time=0.792 ms
64 bytes from 192.168.57.4: icmp_seq=7 ttl=64 time=0.949 ms
64 bytes from 192.168.57.4: icmp_seq=8 ttl=64 time=0.508 ms
64 bytes from 192.168.57.4: icmp_seq=9 ttl=64 time=0.634 ms
64 bytes from 192.168.57.4: icmp_seq=10 ttl=64 time=0.374 ms
64 bytes from 192.168.57.4: icmp_seq=11 ttl=64 time=0.613 ms
```

The screenshot shows a Linux desktop environment with a terminal window open. The terminal title is "VM2 [Running]" and the date and time are "Feb 11 20:31". The terminal window contains the following command-line session:

```
vm1@vm1-VirtualBox:~$ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 brd 0.0.0.0 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 brd :: scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:67:d4:35 brd ff:ff:ff:ff:ff:ff
    inet 192.168.57.4/24 brd 192.168.57.255 scope global dynamic noprefixroute
        valid_lft 406sec preferred_lft 406sec
    inet6 fe80::b48b:11d7:c950:68f9/64 brd fe80::ff:ffff:ffff:ffff:ff:ff scope link noprefixroute
        valid_lft forever preferred_lft forever
vm1@vm1-VirtualBox:~$ ping 192.168.57.3
PING 192.168.57.3 (192.168.57.3) 56(84) bytes of data.
64 bytes from 192.168.57.3: icmp_seq=1 ttl=64 time=0.462 ms
64 bytes from 192.168.57.3: icmp_seq=2 ttl=64 time=0.584 ms
64 bytes from 192.168.57.3: icmp_seq=3 ttl=64 time=0.513 ms
64 bytes from 192.168.57.3: icmp_seq=4 ttl=64 time=0.949 ms
64 bytes from 192.168.57.3: icmp_seq=5 ttl=64 time=0.431 ms
64 bytes from 192.168.57.3: icmp_seq=6 ttl=64 time=0.437 ms
64 bytes from 192.168.57.3: icmp_seq=7 ttl=64 time=0.415 ms
64 bytes from 192.168.57.3: icmp_seq=8 ttl=64 time=0.395 ms
64 bytes from 192.168.57.3: icmp_seq=9 ttl=64 time=0.559 ms
64 bytes from 192.168.57.3: icmp_seq=10 ttl=64 time=0.540 ms
64 bytes from 192.168.57.3: icmp_seq=11 ttl=64 time=0.454 ms
```

Method 3 : Manually Assigning the IP address.

In this method , Go to the Network settings of the virtual machines , go to the settings of the wired connection . There will be a connected network . Go to the settings icon of that connected network. Go to the IPV4 column there. The network is currently taking the automatic (DHCP) addresses.

You can select the manual option from there and you can assign the IP address , subnet mask and gateway to that machine.

The manual configuration page looks like this.

Cancel **Wired** Apply

Details Identity **IPv4** IPv6 Security

IPv4 Method

- Automatic (DHCP)
- Manual
- Shared to other computers

Link-Local Only

Disable

Addresses

Address	Netmask	Gateway	
172.16.0.2	255.255.255.0	172.16.0.1	

DNS

Automatic

Separate IP addresses with commas

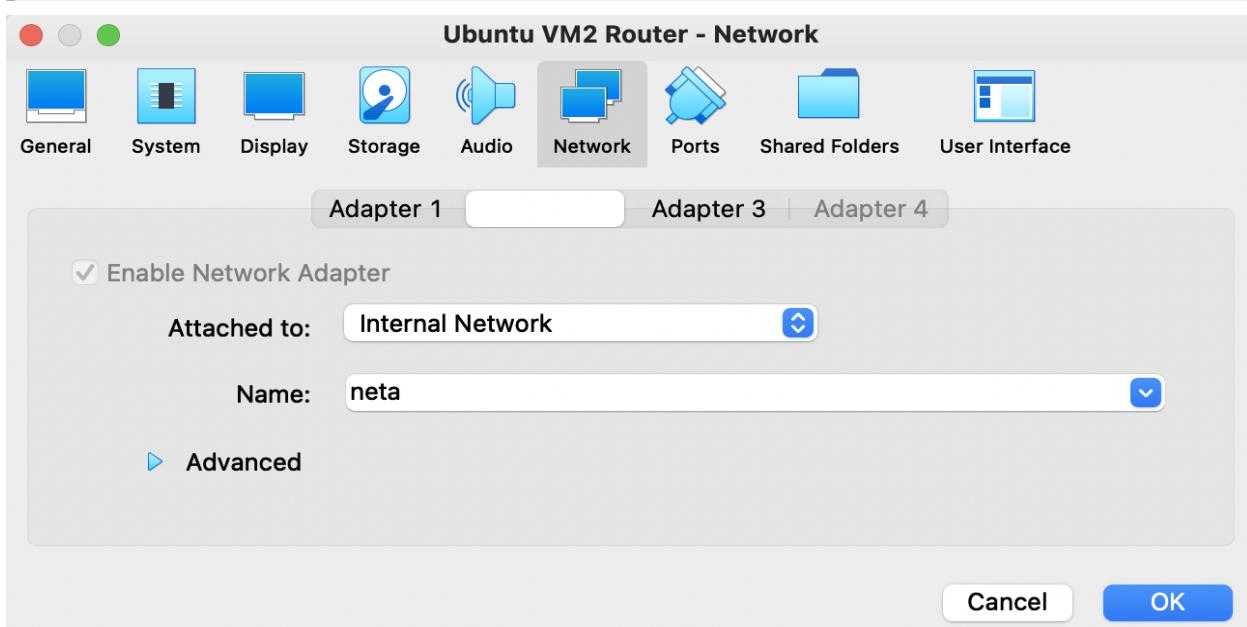
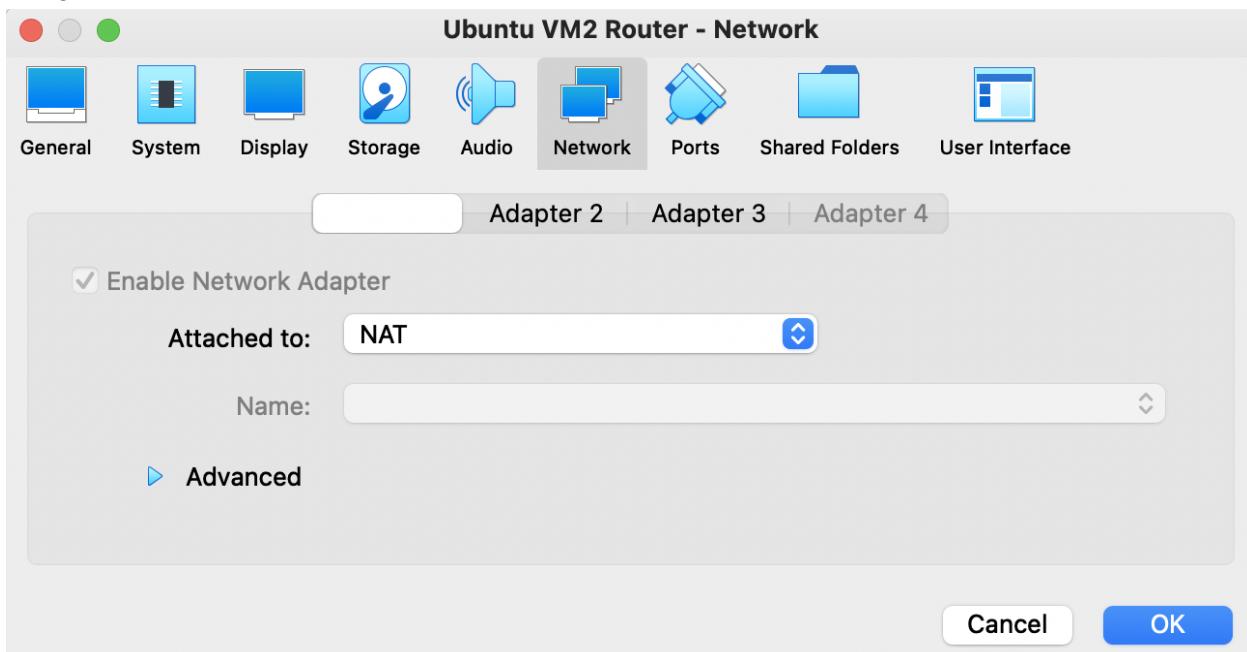
Just Keep in mind that the subnet mask and gateway will be the same. Only IP's will be changed.

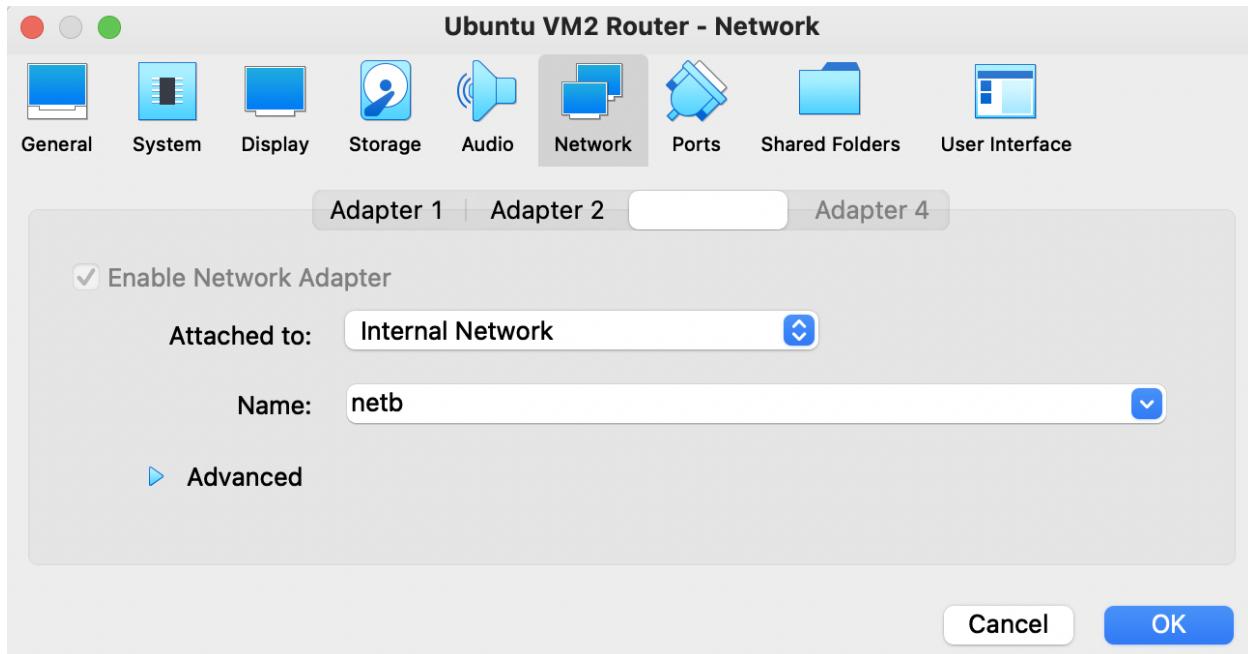
Task 2 : Add another focal VM to the setup and assign connectivity in the following order where VM2 acts like a router. VM1 <-> VM2 <-> VM3. VM1 should be from 172.16.0.0 /24 and VM3 should be from 192.168.0.0 /24. Should be done using static routing.

Step 1: Install three focal Virtual Machine's .

Step 2: To enable the connectivity in the order of VM1<-> VM2 <-> VM3 . We need to do this :
 -> The VM1 consists of two network adapters , one is NAT for the internet acces and other using an internal network . let it be named neta.
 -> The VM2 has to act like a router so it consists of three network adapters , one is NAT for the internet access and other two are the Internal networks that you have connected with VM1 and VM3.

-> The VM3 consists of two network adapters , one is NAT for the internet access and other using an internal network. Let it be named netb.

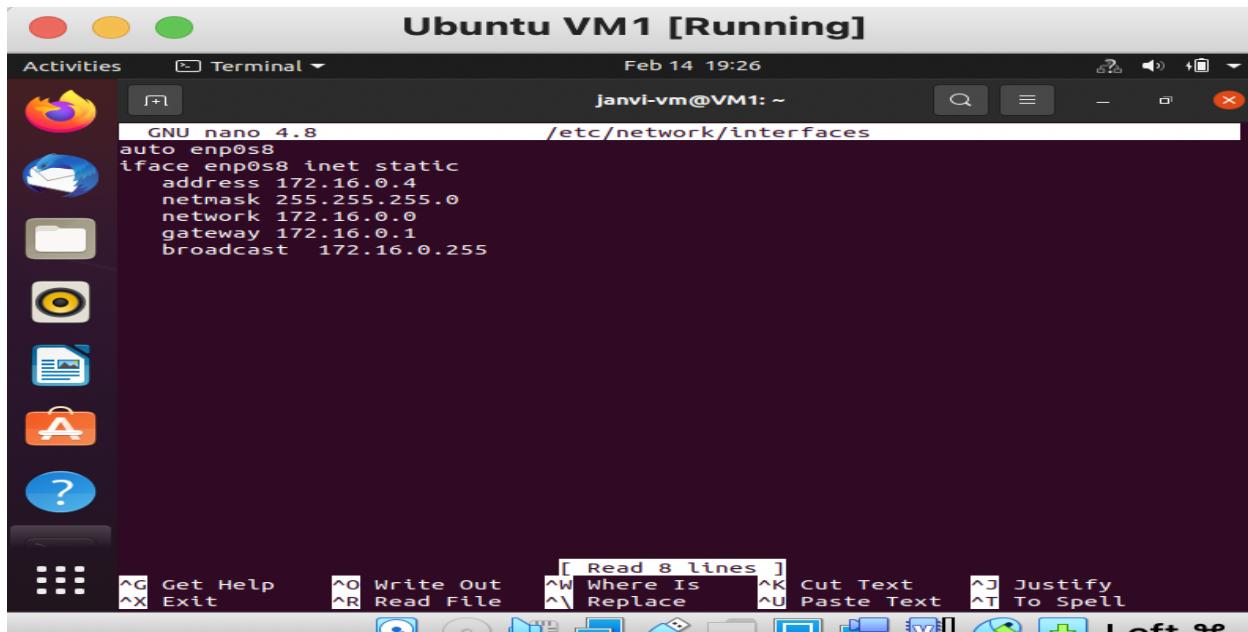




I have shows just for the VM2 above .

Step 3: Configure the Interfaces now.

-> In the VM1 Add the following to the /etc/network/interfaces file.



-> In the VM2 Add the following to the /etc/network/interfaces :

```
GNU nano 4.8          /etc/network/interfaces

#the internal interface for neta
auto enp0s8
iface enp0s8 inet static
    address 172.16.0.5
    netmask 255.255.255.0
    network 172.16.0.0
    broadcast 172.16.0.255
#the internal interface for netb
auto enp0s9
iface enp0s9 inet static
    address 192.168.0.4
    netmask 255.255.255.0
    network 192.168.0.0
    broadcast 192.168.0.255
```

-> In VM3 , Add the following in /etc/network/interfaces

```
Activities Terminal Feb 14 19:29 janvi-vm@VM3: ~
GNU nano 4.8          /etc/network/interfaces
auto enp0s8
iface enp0s8 inet static
    address 192.168.0.5
    netmask 255.255.255.0
    network 192.168.0.0
    broadcast 192.168.0.255
    gateway 192.168.0.1
```

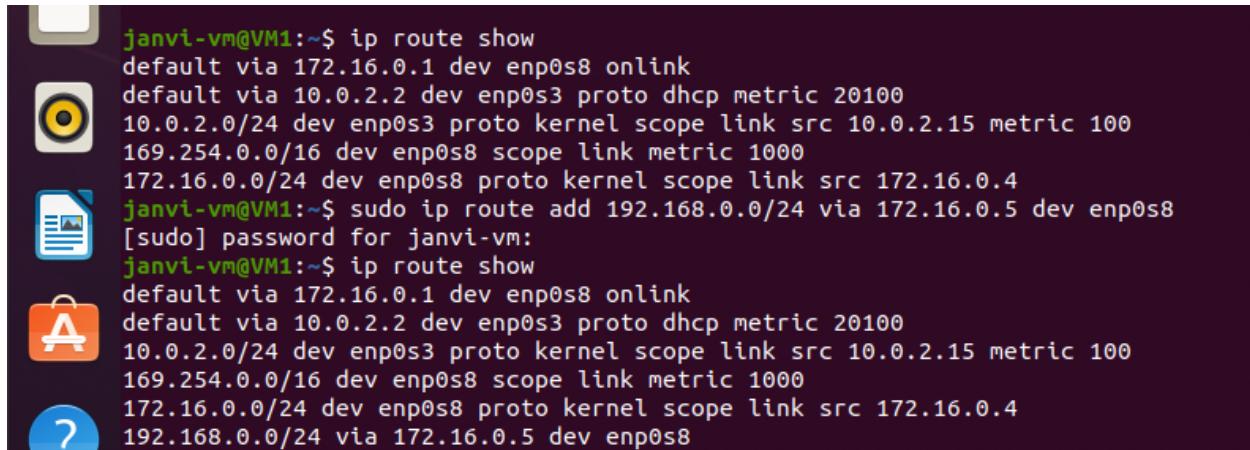
Step 4: Enable forwarding in the /etc/sysctl.conf file

You just have to uncomment the hash in the line referring to ip_forward to be :
net.ipv4.ip_forward = 1

Step 5: Reboot the VMs. Using sudo reboot

Step 5 : Setup Routing . Check the routing table through the ip route show in each of the VM. In order to set up the connectivity between these VM's. Now add static routing using the below commands in VM1 and VM3.

For VM1:



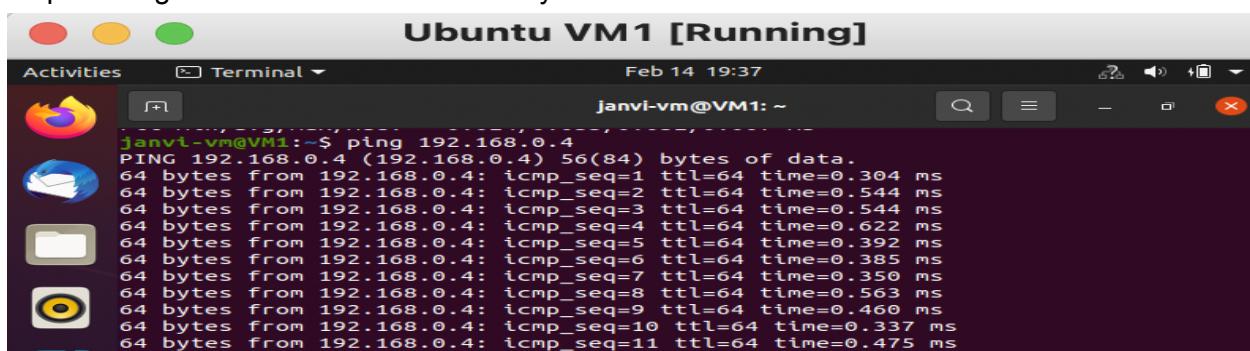
```
janvi-vm@VM1:~$ ip route show
default via 172.16.0.1 dev enp0s8 onlink
default via 10.0.2.2 dev enp0s3 proto dhcp metric 20100
10.0.2.0/24 dev enp0s3 proto kernel scope link src 10.0.2.15 metric 100
169.254.0.0/16 dev enp0s8 scope link metric 1000
172.16.0.0/24 dev enp0s8 proto kernel scope link src 172.16.0.4
janvi-vm@VM1:~$ sudo ip route add 192.168.0.0/24 via 172.16.0.5 dev enp0s8
[sudo] password for janvi-vm:
janvi-vm@VM1:~$ ip route show
default via 172.16.0.1 dev enp0s8 onlink
default via 10.0.2.2 dev enp0s3 proto dhcp metric 20100
10.0.2.0/24 dev enp0s3 proto kernel scope link src 10.0.2.15 metric 100
169.254.0.0/16 dev enp0s8 scope link metric 1000
172.16.0.0/24 dev enp0s8 proto kernel scope link src 172.16.0.4
192.168.0.0/24 via 172.16.0.5 dev enp0s8
```

For VM3:

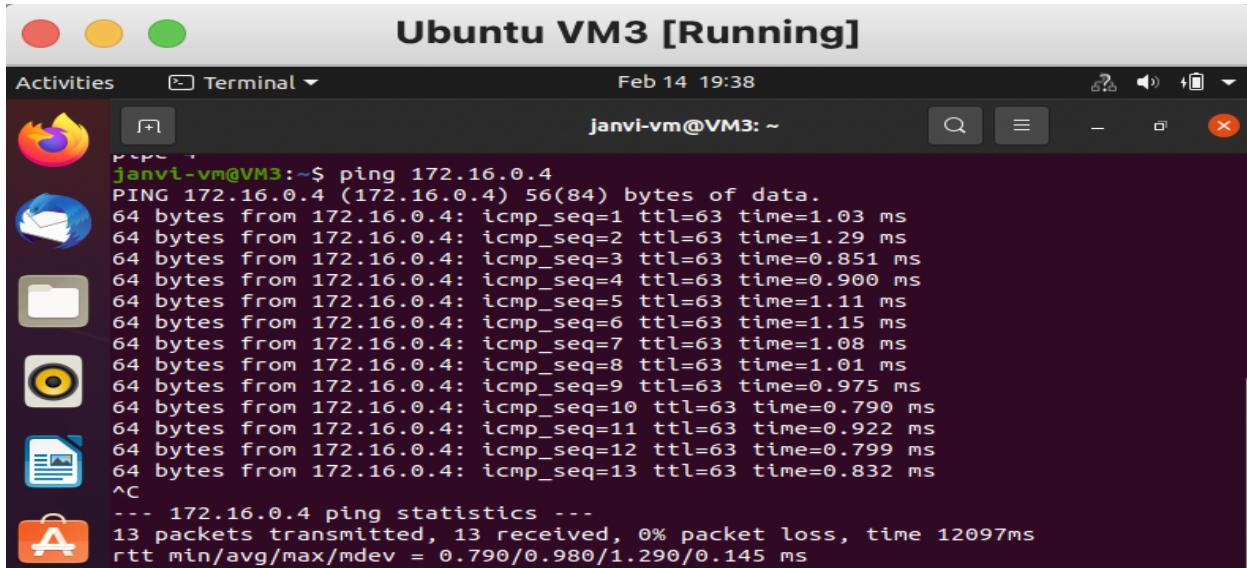


```
Activities Terminal Feb 14 19:36
janvi-vm@VM3:~$ sudo ip route add 172.16.0.0/24 via 192.168.0.4 dev enp0s8
[sudo] password for janvi-vm:
janvi-vm@VM3:~$ ip route show
default via 192.168.0.1 dev enp0s8 onlink
default via 10.0.2.2 dev enp0s3 proto dhcp metric 20100
10.0.2.0/24 dev enp0s3 proto kernel scope link src 10.0.2.15 metric 100
169.254.0.0/16 dev enp0s8 scope link metric 1000
172.16.0.0/24 via 192.168.0.4 dev enp0s8
192.168.0.0/24 dev enp0s8 proto kernel scope link src 192.168.0.5
```

Step 6 : Ping in and check the connectivity



```
Activities Terminal Feb 14 19:37
janvi-vm@VM1:~$ ping 192.168.0.4
PING 192.168.0.4 (192.168.0.4) 56(84) bytes of data.
64 bytes from 192.168.0.4: icmp_seq=1 ttl=64 time=0.304 ms
64 bytes from 192.168.0.4: icmp_seq=2 ttl=64 time=0.544 ms
64 bytes from 192.168.0.4: icmp_seq=3 ttl=64 time=0.544 ms
64 bytes from 192.168.0.4: icmp_seq=4 ttl=64 time=0.622 ms
64 bytes from 192.168.0.4: icmp_seq=5 ttl=64 time=0.392 ms
64 bytes from 192.168.0.4: icmp_seq=6 ttl=64 time=0.385 ms
64 bytes from 192.168.0.4: icmp_seq=7 ttl=64 time=0.350 ms
64 bytes from 192.168.0.4: icmp_seq=8 ttl=64 time=0.563 ms
64 bytes from 192.168.0.4: icmp_seq=9 ttl=64 time=0.460 ms
64 bytes from 192.168.0.4: icmp_seq=10 ttl=64 time=0.337 ms
64 bytes from 192.168.0.4: icmp_seq=11 ttl=64 time=0.475 ms
```



The screenshot shows a terminal window titled "Ubuntu VM3 [Running]" running on an Ubuntu desktop environment. The terminal window has a dark background and displays the output of a "ping" command. The command was run from the root prompt "janvi-vm@VM3: ~" to the IP address "172.16.0.4". The output shows 13 packets transmitted, 13 received, with 0% packet loss and a round-trip time (RTT) of 0.790/0.980/1.290/0.145 ms. The terminal window also shows various icons for desktop applications like Firefox, Nautilus, and the Dash.

```
janvi-vm@VM3:~$ ping 172.16.0.4
PING 172.16.0.4 (172.16.0.4) 56(84) bytes of data.
64 bytes from 172.16.0.4: icmp_seq=1 ttl=63 time=1.03 ms
64 bytes from 172.16.0.4: icmp_seq=2 ttl=63 time=1.29 ms
64 bytes from 172.16.0.4: icmp_seq=3 ttl=63 time=0.851 ms
64 bytes from 172.16.0.4: icmp_seq=4 ttl=63 time=0.900 ms
64 bytes from 172.16.0.4: icmp_seq=5 ttl=63 time=1.11 ms
64 bytes from 172.16.0.4: icmp_seq=6 ttl=63 time=1.15 ms
64 bytes from 172.16.0.4: icmp_seq=7 ttl=63 time=1.08 ms
64 bytes from 172.16.0.4: icmp_seq=8 ttl=63 time=1.01 ms
64 bytes from 172.16.0.4: icmp_seq=9 ttl=63 time=0.975 ms
64 bytes from 172.16.0.4: icmp_seq=10 ttl=63 time=0.790 ms
64 bytes from 172.16.0.4: icmp_seq=11 ttl=63 time=0.922 ms
64 bytes from 172.16.0.4: icmp_seq=12 ttl=63 time=0.799 ms
64 bytes from 172.16.0.4: icmp_seq=13 ttl=63 time=0.832 ms
^C
--- 172.16.0.4 ping statistics ---
13 packets transmitted, 13 received, 0% packet loss, time 12097ms
rtt min/avg/max/mdev = 0.790/0.980/1.290/0.145 ms
```

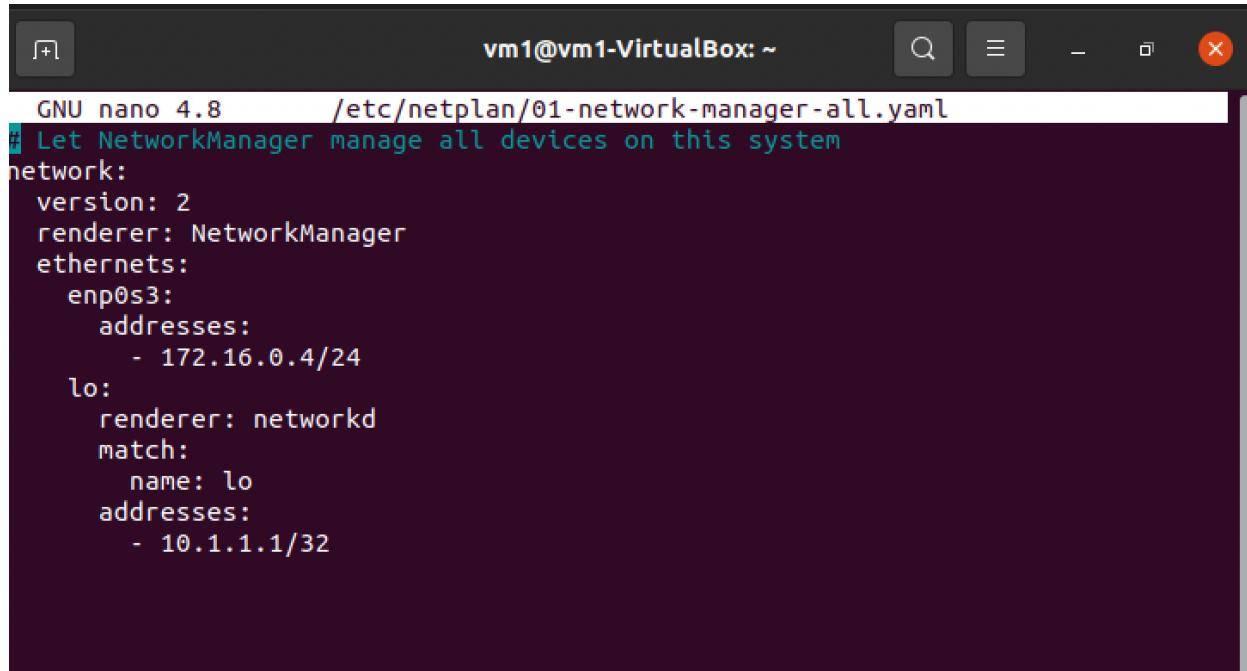
You are done. The connectivity is set up between the machines.

Task 3 : Assign IPs to loopback and recreate #1 and #2 via BGP via FRR.
VM1 lo: 10.1.1.1 /32 , VM2 lo:10.1.1.2 /32 , VM3 lo : 10.1.1.3 /32 , ping should work between the loopback addresses. The network range for BGP IPs is 172.16.0.0 /24 for both the scenarios.

Recreating one:

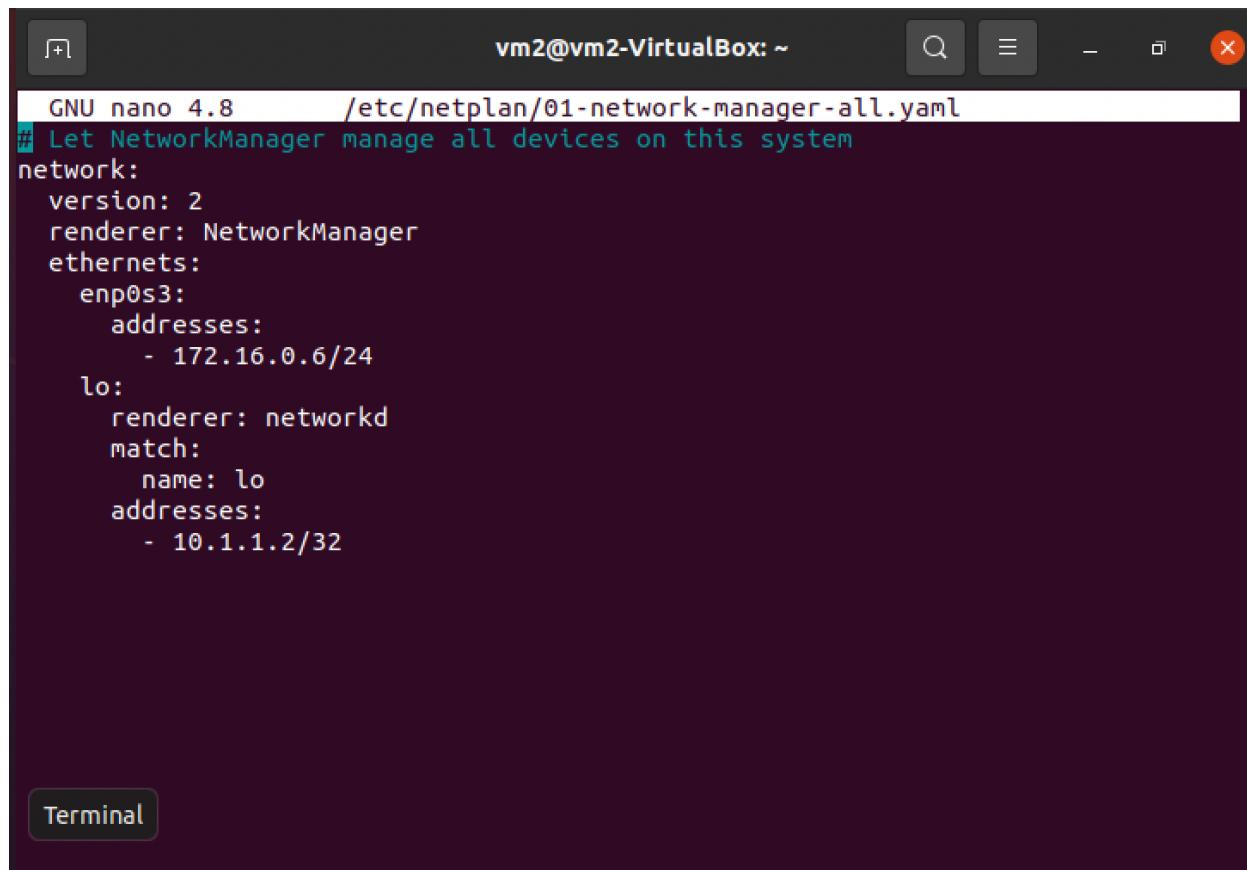
1. Setup two virtual machines and connect them to a NAT network.
2. Open the VM's and assign the IP addresses to them .

VM1:



```
GNU nano 4.8      /etc/netplan/01-network-manager-all.yaml
# Let NetworkManager manage all devices on this system
network:
  version: 2
  renderer: NetworkManager
  ethernets:
    enp0s3:
      addresses:
        - 172.16.0.4/24
    lo:
      renderer: networkd
      match:
        name: lo
      addresses:
        - 10.1.1.1/32
```

VM2:



```
GNU nano 4.8      /etc/netplan/01-network-manager-all.yaml
# Let NetworkManager manage all devices on this system
network:
  version: 2
  renderer: NetworkManager
  ethernets:
    enp0s3:
      addresses:
        - 172.16.0.6/24
    lo:
      renderer: networkd
      match:
        name: lo
      addresses:
        - 10.1.1.2/32
```

3. Install frr on both the VM's now.

```
Curl -s https://deb.frrouting.org/frr/keys.asc | sudo apt-key add -
```

```
FRRVER = "frr-stable"
```

```
Echo deb https://deb.frrouting.org/frr $(lsb_release -s -c) $FRRVER | sudo tee -a /etc/apt/sources.list.d/frr.list
```

```
sudo apt update && sudo apt install frr frr-pythontools
```

4. Enable bgp in the /etc/frr/daemons and restart the systemctl
sudo nano /etc/frr/daemons

```
GNU nano 4.8                               /etc/frr/daemons
# This file tells the frr package which daemons to start.
#
# Sample configurations for these daemons can be found in
# /usr/share/doc/frr/examples/.
#
# ATTENTION:
#
# When activating a daemon for the first time, a config file, even if it is
# empty, has to be present *and* be owned by the user and group "frr", else
# the daemon will not be started by /etc/init.d/frr. The permissions should
# be u=rw,g=r,o=.
# When using "vtysh" such a config file is also needed. It should be owned by
# group "frrvty" and set to ug=rw,o= though. Check /etc/pam.d/frr, too.
#
# The watchfrr, zebra and staticd daemons are always started.
#
bgpd=yes
ospfd=no
ospf6d=no
ripd=no
ripngd=no
isisd=no
pimd=no
ldpd=no
```

5. Enter the vtysh shell using sudo vtysh command.

6. Do the FRR configuration.

```

vm1@vm1-VirtualBox:~$ sudo systemctl restart frr
vm1@vm1-VirtualBox:~$ sudo vtysh

Hello, this is FRRouting (version 8.1).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

vm1-VirtualBox# conf t
vm1-VirtualBox(config)# router bgp 100
vm1-VirtualBox(config-router)# neighbor 172.16.0.6 remote-as 200
vm1-VirtualBox(config-router)# network 10.1.1.1/32
vm1-VirtualBox(config-router)# neighbor 172.16.0.6 prefix-list machines in
vm1-VirtualBox(config-router)# neighbor 172.16.0.6 prefix-list machines out
vm1-VirtualBox(config-router)# exit
vm1-VirtualBox(config)# ip prefix-list machines permit 0.0.0.0/0 ge 32 le 32
vm1-VirtualBox(config)# write memory
% Unknown command: write memory
vm1-VirtualBox(config)# exit
vm1-VirtualBox# write memory
Note: this version of vtysh never writes vtysh.conf
Building Configuration...
Integrated configuration saved to /etc/frr/frr.conf
[OK]
vm1-VirtualBox# sh run
Building configuration...

```

7. Do the same steps for VM2

```

GNU libtirpc 4.8                               /etc/frr/frr.conf
frr version 8.1
frr defaults traditional
hostname vm2-VirtualBox
log syslog informational
no ip forwarding
no ipv6 forwarding
service integrated-vtysh-config
!
router bgp 200
neighbor 172.16.0.4 remote-as 100
!
address-family ipv4 unicast
  network 10.1.1.2/32
  neighbor 172.16.0.4 prefix-list machines in
  neighbor 172.16.0.4 prefix-list machines out
exit-address-family
exit
!
ip prefix-list machines seq 5 permit 0.0.0.0/0 ge 32 le 32
!
```

8. Try to ping over the loopback address on both the machine now.

```
vm1@vm1-VirtualBox:~$ ip route show
10.1.1.2 nhid 10 via 172.16.0.6 dev enp0s3 proto bgp metric 20
169.254.0.0/16 dev enp0s3 scope link metric 1000
172.16.0.0/24 dev enp0s3 proto kernel scope link src 172.16.0.4 metric 100
```

```
vm2@vm2-VirtualBox:~$ ip route show
10.1.1.1 nhid 8 via 172.16.0.4 dev enp0s3 proto bgp metric 20
169.254.0.0/16 dev enp0s3 scope link metric 1000
172.16.0.0/24 dev enp0s3 proto kernel scope link src 172.16.0.6 metric 100
```

```
[root@vm1 ~]# ping 10.1.1.2
vm1@vm1-VirtualBox:~$ ping 10.1.1.2
PING 10.1.1.2 (10.1.1.2) 56(84) bytes of data.
64 bytes from 10.1.1.2: icmp_seq=1 ttl=64 time=0.338 ms
64 bytes from 10.1.1.2: icmp_seq=2 ttl=64 time=0.430 ms
64 bytes from 10.1.1.2: icmp_seq=3 ttl=64 time=0.416 ms
^X64 bytes from 10.1.1.2: icmp_seq=4 ttl=64 time=0.480 ms
^Z
[1]+  Stopped                  ping 10.1.1.2
```

```
vm2@vm2-VirtualBox:~$ ping 10.1.1.1
PING 10.1.1.1 (10.1.1.1) 56(84) bytes of data.
64 bytes from 10.1.1.1: icmp_seq=1 ttl=64 time=0.575 ms
64 bytes from 10.1.1.1: icmp_seq=2 ttl=64 time=0.393 ms
64 bytes from 10.1.1.1: icmp_seq=3 ttl=64 time=0.386 ms
64 bytes from 10.1.1.1: icmp_seq=4 ttl=64 time=0.645 ms
^Z
[ Show Applications ]          ping 10.1.1.1
```

Recreating 2:

1. Set up 3 virtual machines.
2. Install FRR and enable bgp on all the 3 virtual machines as shown above.
3. Add loopback addresses and Bgp ip's to the machines.

A screenshot of a Linux desktop environment, likely Ubuntu, showing three terminal windows side-by-side. The desktop background is dark purple with a grid pattern. The top-left icon dock contains icons for a browser, file manager, terminal, and other applications.

The first terminal window (vm1) shows the configuration for a single network interface (enp0s3) with one IPv4 address (172.16.0.4/24) and one IPv6 link-local address (10.1.1.1/32). The second terminal window (vm2) shows a more complex configuration with two network interfaces (enp0s3 and enp0s8) and their respective IPv4 and IPv6 addresses. The third terminal window (vm3) shows another configuration for a single network interface (enp0s3) with one IPv4 address (192.168.0.4/24) and one IPv6 link-local address (10.1.1.3/32).

```
GNU nano 4.8      /etc/netplan/01-network-manager-all.yaml
# Let NetworkManager manage all devices on this system
network:
  version: 2
  renderer: NetworkManager
  ethernets:
    enp0s3:
      addresses:
        - 172.16.0.4/24
    lo:
      renderer: networkd
      match:
        name: lo
      addresses:
        - 10.1.1.1/32
```

The second terminal window (vm2) shows the configuration for two network interfaces:

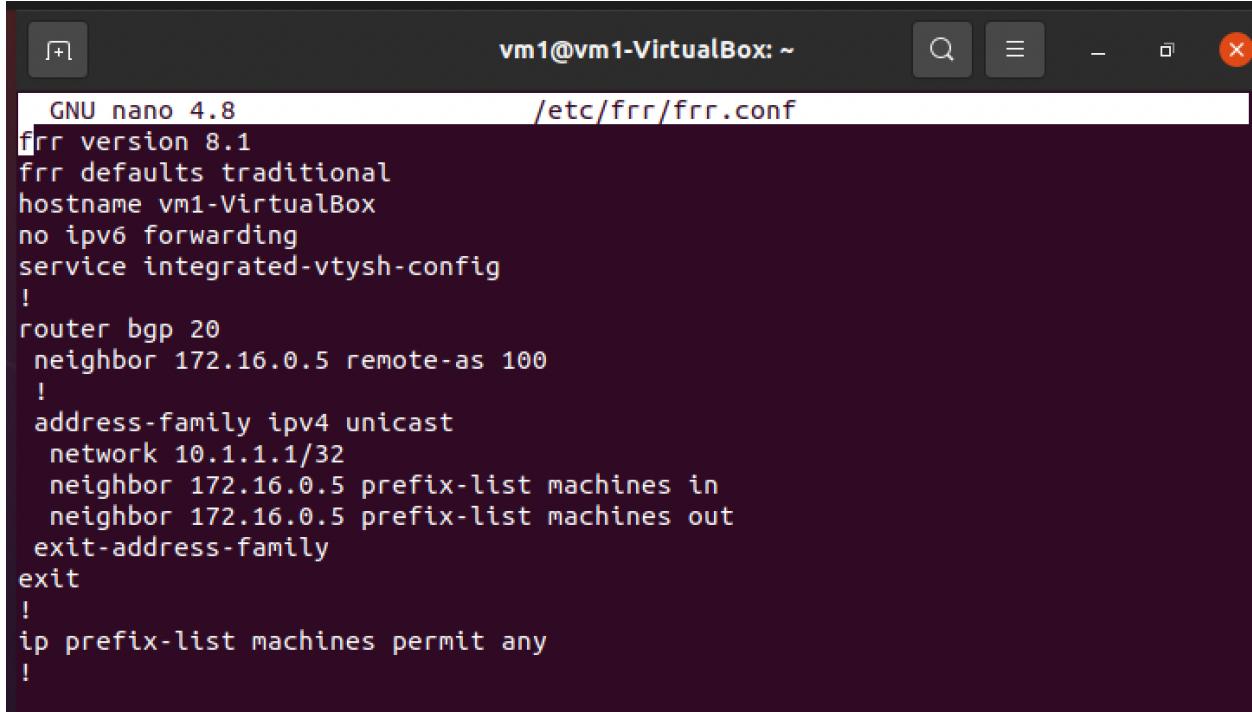
```
GNU nano 4.8      /etc/netplan/01-network-manager-all.yaml
# Let NetworkManager manage all devices on this system
network:
  version: 2
  renderer: NetworkManager
  ethernets:
    enp0s3:
      addresses:
        - 172.16.0.5/24
    enp0s8:
      addresses:
        - 192.168.0.5/24
    lo:
      renderer: networkd
      match:
        name: lo
      addresses:
        - 10.1.1.2/32
```

The third terminal window (vm3) shows the configuration for a single network interface (enp0s3) with one IPv4 address (192.168.0.4/24) and one IPv6 link-local address (10.1.1.3/32).

```
GNU nano 4.8      /etc/netplan/01-network-manager-all.yaml
# Let NetworkManager manage all devices on this system
network:
  version: 2
  renderer: NetworkManager
  ethernets:
    enp0s3:
      addresses:
        - 192.168.0.4/24
    lo:
      renderer: networkd
      match:
        name: lo
      addresses:
        - 10.1.1.3/32
```

4. Restart the frr service
5. Now configure the connection via bgp using frr with the help of vtysh shell.

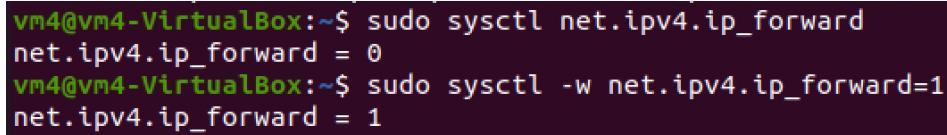
VM1:



The screenshot shows a terminal window titled "vm1@vm1-VirtualBox: ~". The title bar also displays "GNU nano 4.8 /etc/frr/frr.conf". The content of the file is as follows:

```
GNU nano 4.8          /etc/frr/frr.conf
frr version 8.1
frr defaults traditional
hostname vm1-VirtualBox
no ipv6 forwarding
service integrated-vtysh-config
!
router bgp 20
neighbor 172.16.0.5 remote-as 100
!
address-family ipv4 unicast
  network 10.1.1.1/32
  neighbor 172.16.0.5 prefix-list machines in
  neighbor 172.16.0.5 prefix-list machines out
  exit-address-family
exit
!
ip prefix-list machines permit any
!
```

VM2:



The screenshot shows a terminal window titled "vm4@vm4-VirtualBox: ~". The title bar also displays "vm4@vm4-VirtualBox: ~\$". The commands entered are:

```
vm4@vm4-VirtualBox:~$ sudo sysctl net.ipv4.ip_forward
net.ipv4.ip_forward = 0
vm4@vm4-VirtualBox:~$ sudo sysctl -w net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
```

Enable Ipv4 on the routing machine using these commands.

```
GNU nano 4.8          /etc/frr/frr.conf
frr version 8.2
frr defaults traditional
hostname vm4-VirtualBox
log syslog informational
no ip forwarding
no ipv6 forwarding
service integrated-vtysh-config
!
router bgp 100
neighbor 172.16.0.4 remote-as 20
neighbor 192.168.0.5 remote-as 10
!
address-family ipv4 unicast
network 10.1.1.2/32
neighbor 172.16.0.4 default-originate
neighbor 172.16.0.4 prefix-list machines in
neighbor 172.16.0.4 prefix-list machines out
neighbor 192.168.0.5 default-originate
neighbor 192.168.0.5 prefix-list machines in
neighbor 192.168.0.5 prefix-list machines out
redistribute connected
exit-address-family
e Terminal
!
ip prefix-list machines permit any
```

VM3:

```
GNU nano 4.8          /etc/frr/frr.conf
frr version 8.1
frr defaults traditional
hostname vm3-VirtualBox
no ipv6 forwarding
service integrated-vtysh-config
!
router bgp 10
neighbor 192.168.0.4 remote-as 100
!
address-family ipv4 unicast
network 10.1.1.3/32
neighbor 192.168.0.4 prefix-list machines in
neighbor 192.168.0.4 prefix-list machines out
exit-address-family
exit
!
ip prefix-list machines permit any
```

6. Check the configuration that if the loopback address is advertised via bgp or not using ip route show.

7. Try pinging the Vm's.

```
vm1@vm1-VirtualBox:~$ sudo systemctl restart frr
vm1@vm1-VirtualBox:~$ ip r s
default nhid 10 via 172.16.0.5 dev enp0s3 proto bgp metric 20
10.1.1.2 nhid 10 via 172.16.0.5 dev enp0s3 proto bgp metric 20
10.1.1.3 nhid 10 via 172.16.0.5 dev enp0s3 proto bgp metric 20
169.254.0.0/16 dev enp0s3 scope link metric 1000
172.16.0.0/24 dev enp0s3 proto kernel scope link src 172.16.0.4 metric 100
192.168.0.0/24 via 172.16.0.5 dev enp0s3
```

```
vm4@vm4-VirtualBox:~$ ip r s
10.1.1.1 nhid 104 via 172.16.0.4 dev enp0s3 proto bgp metric 20
10.1.1.3 nhid 102 via 192.168.0.5 dev enp0s8 proto bgp metric 20
169.254.0.0/16 dev enp0s3 scope link metric 1000
172.16.0.0/24 dev enp0s3 proto kernel scope link src 172.16.0.5 metric 100
192.168.0.0/24 dev enp0s8 proto kernel scope link src 192.168.0.4 metric 101
```

```
vm3@vm3-VirtualBox:~$ sudo systemctl restart frr
vm3@vm3-VirtualBox:~$ ip r s
default nhid 10 via 192.168.0.4 dev enp0s3 proto bgp metric 20
10.1.1.1 nhid 10 via 192.168.0.4 dev enp0s3 proto bgp metric 20
10.1.1.2 nhid 10 via 192.168.0.4 dev enp0s3 proto bgp metric 20
169.254.0.0/16 dev enp0s3 scope link metric 1000
172.16.0.0/24 via 192.168.0.4 dev enp0s3
192.168.0.0/24 dev enp0s3 proto kernel scope link src 192.168.0.5 metric 100
```

```
vm1@vm1-VirtualBox:~$ ping 10.1.1.3
PING 10.1.1.3 (10.1.1.3) 56(84) bytes of data.
64 bytes from 10.1.1.3: icmp_seq=1 ttl=63 time=0.582 ms
64 bytes from 10.1.1.3: icmp_seq=2 ttl=63 time=0.608 ms
64 bytes from 10.1.1.3: icmp_seq=3 ttl=63 time=0.826 ms
64 bytes from 10.1.1.3: icmp_seq=4 ttl=63 time=0.800 ms
64 bytes from 10.1.1.3: icmp_seq=5 ttl=63 time=0.549 ms
^Z
[4]+ Stopped ping 10.1.1.3
```

```
vm3@vm3-VirtualBox:~$ ping 10.1.1.1
PING 10.1.1.1 (10.1.1.1) 56(84) bytes of data.
64 bytes from 10.1.1.1: icmp_seq=1 ttl=63 time=0.578 ms
64 bytes from 10.1.1.1: icmp_seq=2 ttl=63 time=0.790 ms
^Z
[5]+ Stopped ping 10.1.1.1
vm3@vm3-VirtualBox:~$
```