Lab 6: Linux Networking: Internetworking and DNS overview

Parth Kalkar

Q1. You have the task to configure subinterfaces with VLANs using 2 methods:

- by using netplan
- by using iplink command

Below we provided 2 different VLANs to be configured with both mentioned methods accordingly:

- vlan 10 with ip addresses 192.168.10.10/24 and its gateway 192.168.10.1
- vlan 20 with ip addresses 192.168.20.20/24 and its gateway 192.168.20.1

For iplink command manuals you can use this link

For netplan command manuals you can use this link

Note - The above mentioned links were used for reference

a. Using iplink - We can do the following commands
    1. $ ip link add link enp2s0 name vlan10 address 192.168.10.10/24 type vlan id 10
    2. $ ip link add link enp2s0 name vlan20 address 192.168.20.20/24 type vlan id 20
b. Using Netplan - We can do the following commands
    1. Change the network-manager file with contents as shown in the picture below
    2. $ sudo netplan generate
    3. $ sudo netplan apply

We can clearly see that we successfully added VLANs, they are shown at no.4 and no.5

```
  GNU nano 4.8          01-network-manager-all.yaml
# Let NetworkManager manage all devices on this system
network:
  version: 2
  renderer: NetworkManager
  ethernets:
    enp2s0:
      dhcp4: no
      addresses: [192.168.0.0/24]
      nameservers:
        addresses: [8.8.8.8,192.168.0.109,192.168.0.105]

  vlans:
    vlan10:
      id: 10
      link: enp2s0
      addresses: [ 192.168.10.10/24 ]
      routes:
        - to: 192.168.10.10/24
          via: 192.168.10.1
          on-link: true
    vlan20:
      id: 20
      link: enp2s0
      addresses: [ 192.168.20.20/24 ]
      gateway4: 192.168.20.1
```

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mo
de DEFAULT group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: enp2s0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel
 state DOWN mode DEFAULT group default qlen 1000
    link/ether 8c:dc:d4:cf:c4:67 brd ff:ff:ff:ff:ff:ff
3: wlo1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue stat
e UP mode DORMANT group default qlen 1000
    link/ether d8:fc:93:a3:a1:d3 brd ff:ff:ff:ff:ff:ff
    altname wlp3s0
4: vlan20@enp2s0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc n
oqueue state LOWERLAYERDOWN mode DEFAULT group default qlen 1000
    link/ether 8c:dc:d4:cf:c4:67 brd ff:ff:ff:ff:ff:ff
5: vlan10@enp2s0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc n
oqueue state LOWERLAYERDOWN mode DEFAULT group default qlen 1000
    link/ether 8c:dc:d4:cf:c4:67 brd ff:ff:ff:ff:ff:ff
```

Q2. In this task you should create VM under Ubuntu OS with 3 interfaces :

a. first interface with adapter NAT

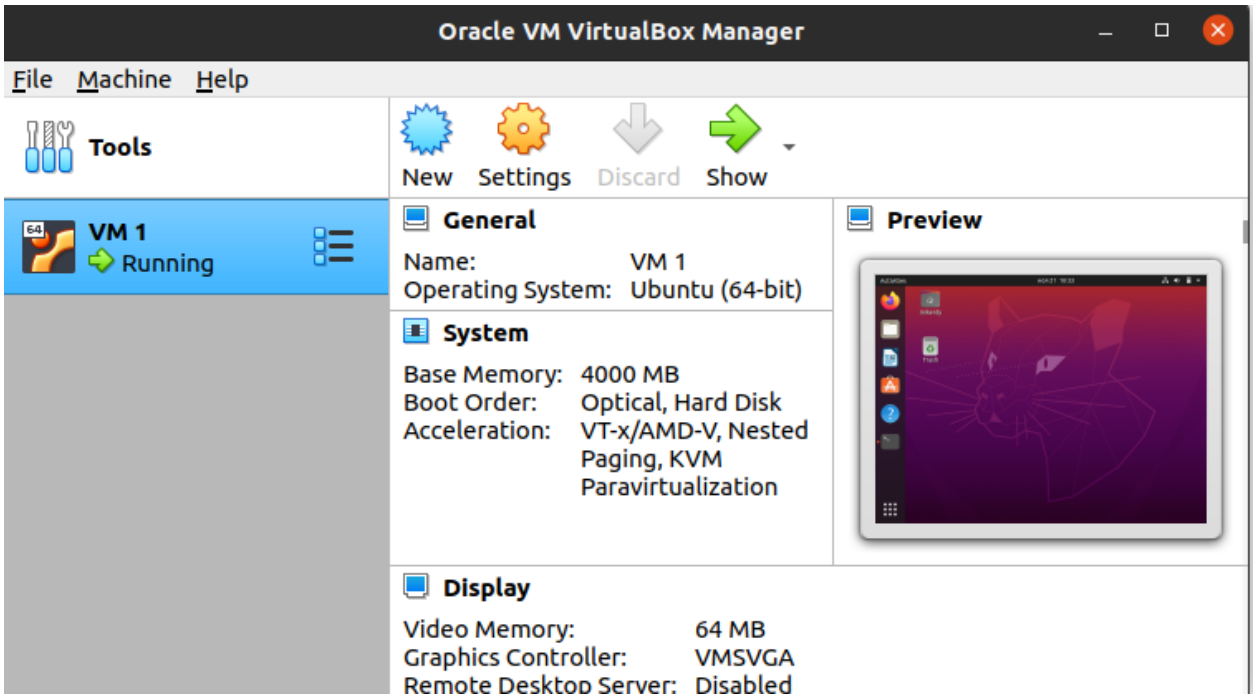
b. second interface with adapter INTERNAL NETWORK and Name=intnet (enabled promiscuous mode)
c. third interface with adapter INTERNAL NETWORK and Name=intnet2 (enabled promiscuous mode)

Then you need to make a virtual bridge to be able to connect 2 different interfaces (intnet, intnet2). In order to successfully complete the task, you need to show that now your virtual bridge (switch) has both MAC addresses in its table.

- I'm using Virtual Box with a VM using Ubuntu 20.04 LTS.

  Let's configure it with 3 interfaces:
  a. First interface with adapter NAT
  b. Second interface with adapter INTERNAL NETWORK and Name=intnet (enabled promiscuous mode)
  c. Third interface with adapter INTERNAL NETWORK and Name=intnet2 (enabled promiscuous mode)

1. The picture below shows the running virtual machine

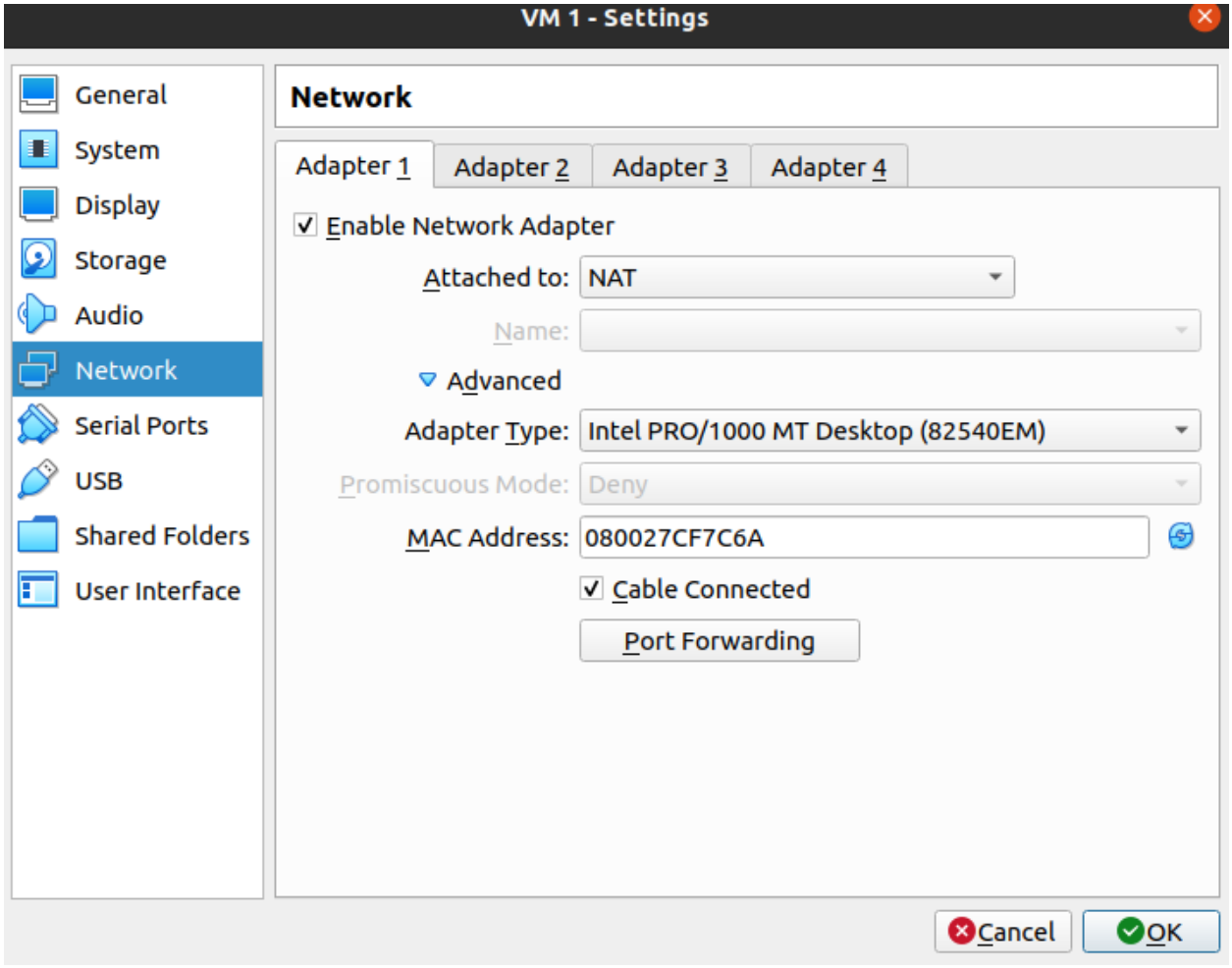


2. The picture below shows the adapter specs

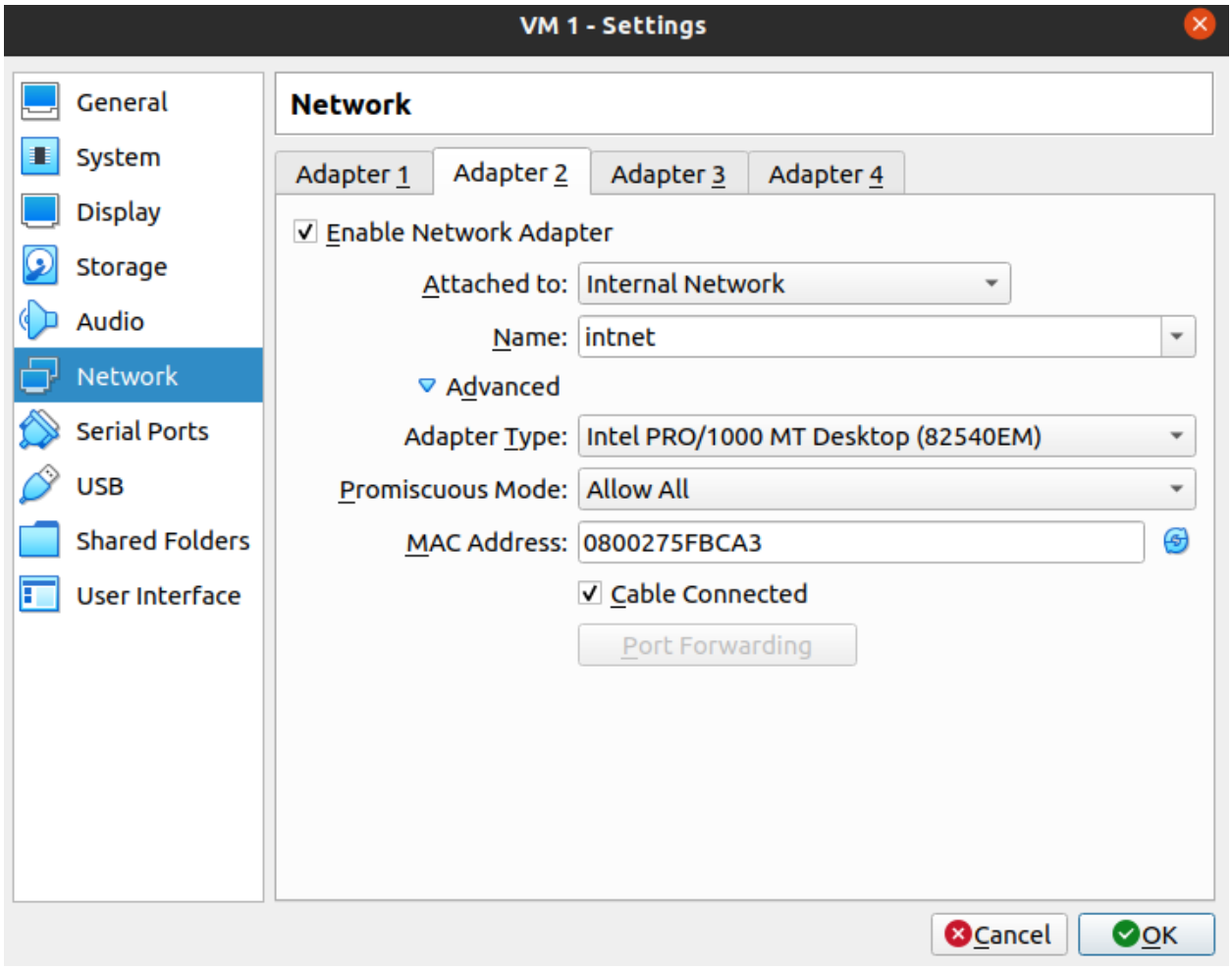
Network

Adapter 1: Intel PRO/1000 MT Desktop (NAT)
Adapter 2: Intel PRO/1000 MT Desktop (Internal Network, 'intnet')
Adapter 3: Intel PRO/1000 MT Desktop (Internal Network, 'intnet2')

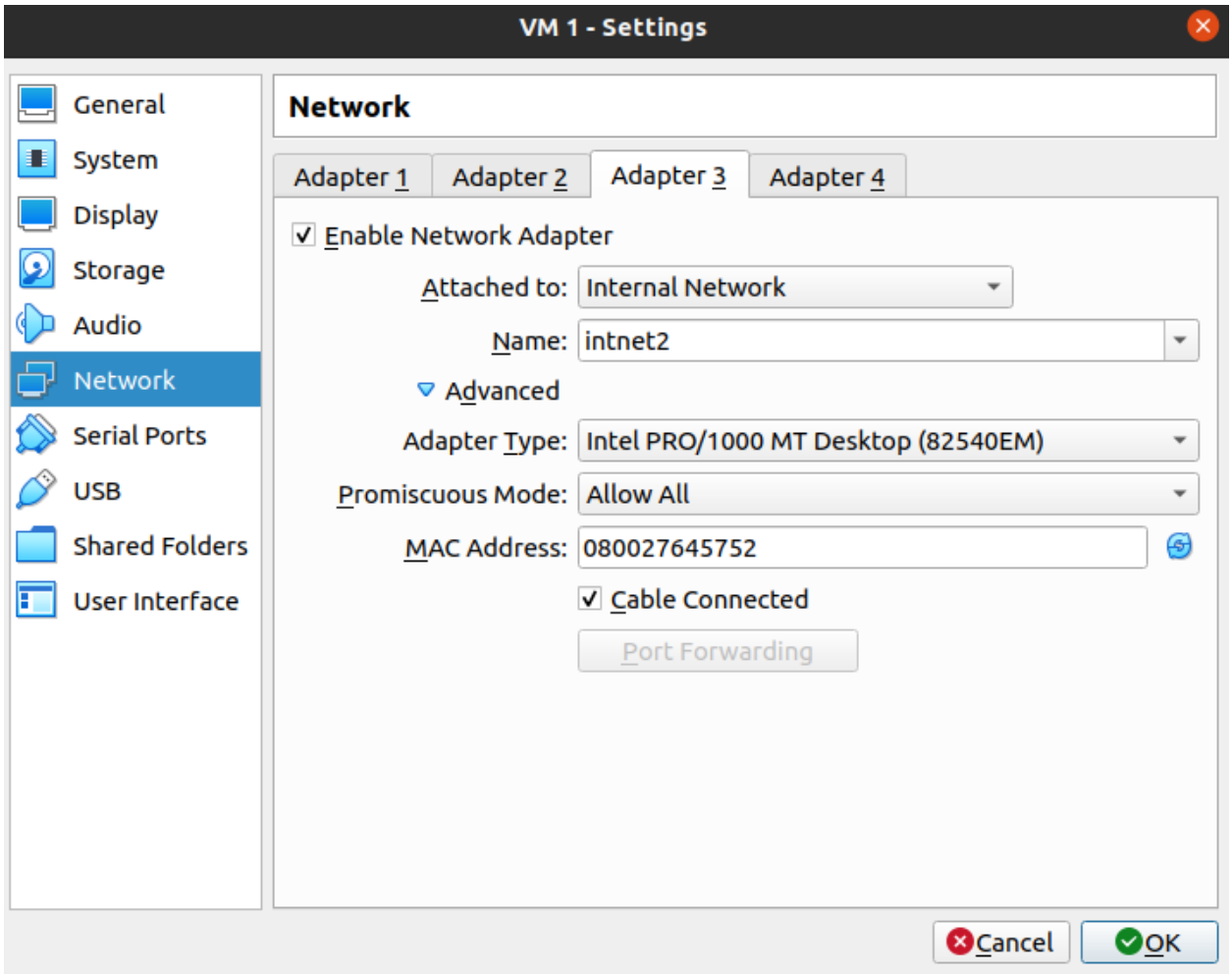
3. Configuring Interface 1 –



4. Configuring Interface 2 - using MAC address



5. Configuring Interface 3 - using MAC address



We can check our interfaces as we are already inside our VM by using the command $ ip link

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAUL
T group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP m
ode DEFAULT group default qlen 1000
    link/ether 08:00:27:cf:7c:6a brd ff:ff:ff:ff:ff:ff
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP m
ode DEFAULT group default qlen 1000
    link/ether 08:00:27:5f:bc:a3 brd ff:ff:ff:ff:ff:ff
4: enp0s9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP m
ode DEFAULT group default qlen 1000
    link/ether 08:00:27:64:57:52 brd ff:ff:ff:ff:ff:ff
```

The new internal connections are enp0s8 and enp0s9.

Now we have to create a bridge, configure them and connect them to the interfaces, to do so we can follow the following commands:

1. $ sudo ip link add bridge1 type bridge
2. $ sudo ip link set bridge1 up
3. $ sudo ip link set enp0s8 up
4. $ sudo ip link set enp0s9 up
5. $ sudo ip link set enp0s8 master bridge1
6. $ sudo ip link set enp0s9 master bridge1

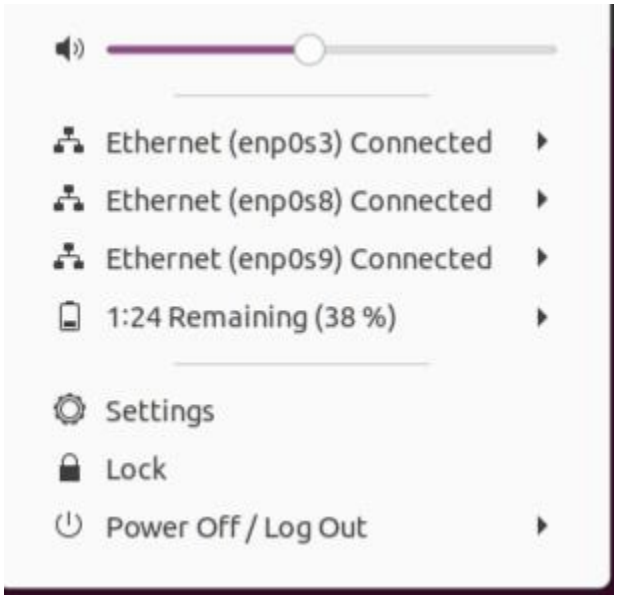
We can check if the above commands executed correctly, we can use $ bridge link

```
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 master bridge1 state forw
arding priority 32 cost 4
4: enp0s9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 master bridge1 state forw
arding priority 32 cost 4
```

Let's check the MAC addresses of our bridge to make sure that the bridge has the Interfaces, we can use $ bridge fdb

```
01:00:5e:00:00:01 dev enp0s3 self permanent
33:33:00:00:00:01 dev enp0s3 self permanent
33:33:00:00:00:fb dev enp0s3 self permanent
33:33:ff:8f:52:f9 dev enp0s3 self permanent
01:00:5e:00:00:fb dev enp0s3 self permanent
08:00:27:5f:bc:a3 dev enp0s8 vlan 1 master bridge1 permanent
08:00:27:5f:bc:a3 dev enp0s8 master bridge1 permanent
01:00:5e:00:00:01 dev enp0s8 self permanent
33:33:00:00:00:01 dev enp0s8 self permanent
08:00:27:64:57:52 dev enp0s9 vlan 1 master bridge1 permanent
08:00:27:64:57:52 dev enp0s9 master bridge1 permanent
01:00:5e:00:00:01 dev enp0s9 self permanent
33:33:00:00:00:01 dev enp0s9 self permanent
33:33:00:00:00:01 dev bridge1 self permanent
01:00:5e:00:00:6a dev bridge1 self permanent
33:33:00:00:00:6a dev bridge1 self permanent
01:00:5e:00:00:01 dev bridge1 self permanent
33:33:ff:01:bd:87 dev bridge1 self permanent
33:33:00:00:00:fb dev bridge1 self permanent
```

The MAC Addresses of our interfaces are available, the final result is shown below



Alternate Method -

1. First interface with adapter NAT

In Ubuntu 18.04, network configuration is done using the netplan utility. This utility reads yaml configuration files stored in `/etc/netplan/` and generates from them network configuration consumed by the system.

Therefore, in Ubuntu 18.04 instead of modifying `/etc/network/interfaces`, it's about creating (or modifying) a yaml configuration file. However, before doing this, it is better to check what network interfaces are available. Using $ *ip a*

I have the list of network interfaces: lo - loopback interface, *enp0s3* and *enp0s8* refer to the NAT and Host-only networks correspondingly. Now, creating a new file in `/etc/netplan/` it will provide the network configuration. Often, system administrators give this file the name `01-netcfg.yaml`.

So, applying the following command, `sudo nano /etc/netplan/01-netcfg.yaml`. Adding the following configuration:

```
network:
  version: 2
  ethernets:
    enp0s3:
      dhcp4: yes
    enp0s8:
      dhcp4: no
      addresses: [192.168.56.10/24]
```

```
    # gateway4: 192.168.56.1
```

For the *enp0s3* interface, dynamic allocation of IP addressing is enabled using the option dhcp4: yes, while for *enp0s8* we have to configure values manually.
In order to do this, we disable DHCP (dhcp4: no), set the IP address and the network mask (addresses: [192.168.56.10/24]), and possibly provide a gateway IP address (gateway4: 192.168.56.1).

Running $ *sudo netplan try*, in order to check that the configuration is correct and to apply configuration, Running $ *sudo netplan apply*

2. Second interface with adapter INTERNAL NETWORK and Name=intnet (enabled promiscuous mode) & Third interface with adapter INTERNAL NETWORK and Name=intnet2 (enabled promiscuous mode))

$ *vboxmanage dhcpserver add --netname intnet --ip 10.10.0.1 --netmask 255.255.0.0 --lowerip 10.10.10.2 --upperip 10.10.10.254 --enable*
Where

- --netname intnet: The internal network will be named intnet. This is the name you should then put inside your VM's **Network > Adapter > (Internal network) > Name** field.
- --ip 10.10.0.1: The IP address of your DHCP server inside the internal network.
- --netmask 255.255.0.0: The subnet mask.
- --lowerip 10.10.10.2: The lower bound of the IP addresses that can be assigned to network members.
- --upperip 10.10.10.254: The upper bound of the same thing.
- --enable: Enable the DHCP server.