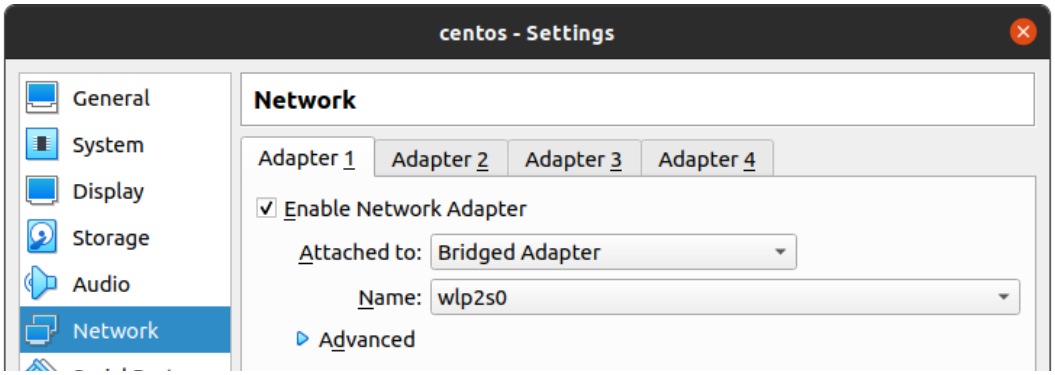
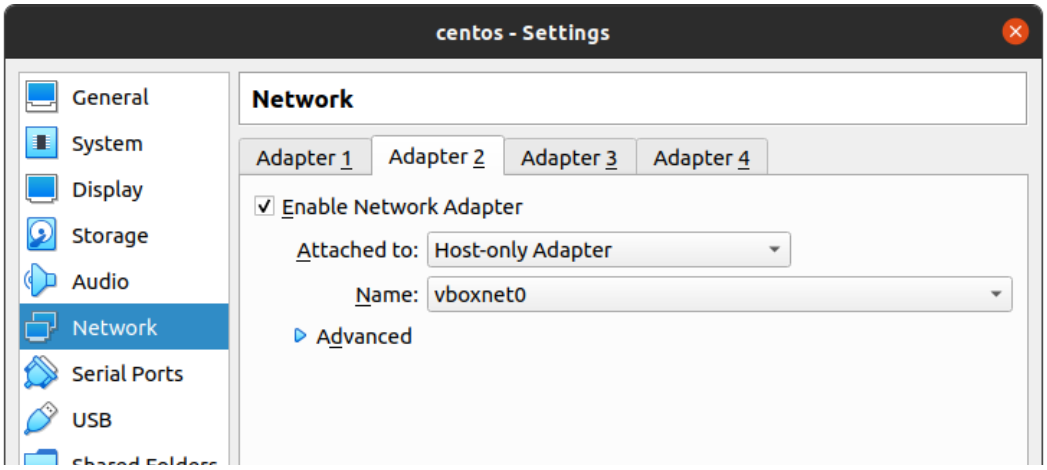
QN1: Create one vm with 2 network interfaces one should behave as WAN and another as LAN. Create another vm attaching the previously created LAN interface to it. Implement NAT in the first vm, so that the second vm can access the internet.

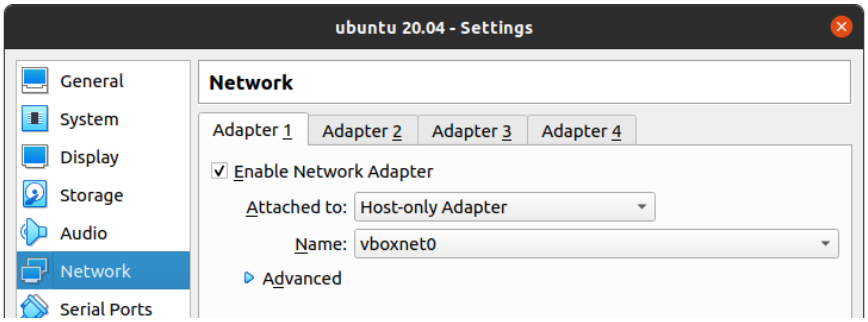
ANS:
In my virtual machine, I installed CentOS 8 (as my first VM) with two network interfaces, behaving as a WAN and a LAN.
For this, On settings of the virtual box, on the left, we can find ‘Network’,
And we can enable any number of adapters. For WAN, I used adapter1 and it is bridged as shown.



Again, adapter 2 was enabled and it was selected as Host-only adapter.



And on our Ubuntu system, which is our second VM, the same process was carried out only for Adapter 1, since It should behave as LAN only.



Then Both systems are started.

Now, to see the IP related information, we can use '***ip a***' anytime.

We tried to ping the ip 192.168.1.182 from our VM2 but all we got was 100% packet loss, and an error message of inability to connect to the host.

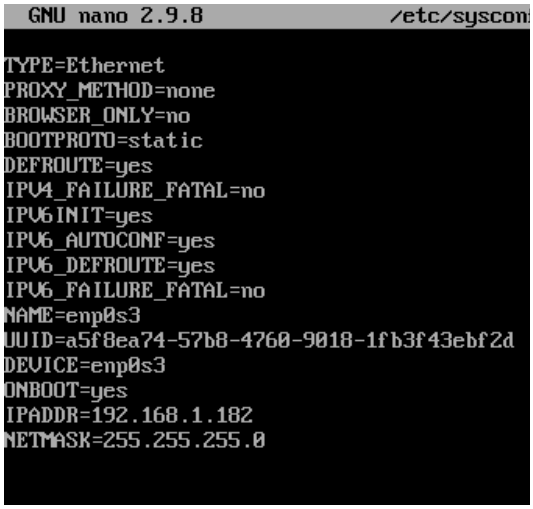
```
tom@batman:~/Desktop$ ping 192.168.1.182
PING 192.168.1.182 (192.168.1.182) 56(84) bytes of data.
^[[A^[[A^[[B^[[B^C
--- 192.168.1.182 ping statistics ---
16 packets transmitted, 0 received, 100% packet loss, time 15968ms
```

Now, at our VM1, we should edit the network scripts.
But first, we should see the name of our network adapter, we can use '***ip a***' for that.

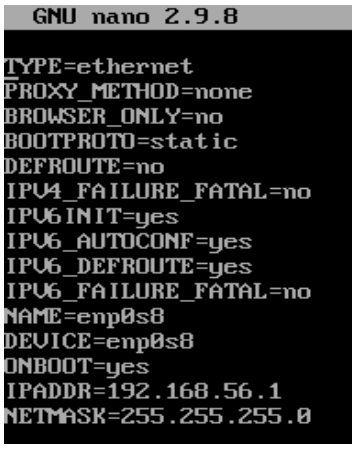
```
[root@batman bijay]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 100
0
    link/ether 08:00:27:c8:f5:ab brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.182/24 brd 192.168.1.255 scope global noprefixroute enp0s3
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fec8:f5ab/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 100
0
    link/ether 08:00:27:10:19:73 brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.1/24 brd 192.168.56.255 scope global noprefixroute enp0s8
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe10:1973/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
[root@batman bijay]# _
```

Here we can see that two network components are present as enp0s3 (ip=192.168.1.182/24) which is our router device and enp0s8(192.168.56.1/24) which is our host only device.

To edit network scripts:
'Nano /etc/sysconfig/network-scripts/ifcfg-enp0s3'
And it should be set as shown in the figure below:



It is saved and again for next device enp0s8, we can configure the file by the command:
'Nano /etc/sysconfig/network-scripts/ifcfg-enp0s8'
And setting it as:



And network was restarted using command:
'***Systemctl restart NetworkManager.service***'

Now, we should enable IP forwarding,
For that, we created a file:
'Nano /etc/sysctl.d/ip_forward.conf'
And inside that file we added just 1 line ===> net.ipv4.ip_forward=1
And saved that file.

The configuration is then activated and made permanent using command:
'Sysctl -p /etc/sysctl.d/ip_forward.conf'

```
[root@batman bijay]# sysctl -p /etc/sysctl.d/ip_forward.conf
net.ipv4.ip_forward = 1
[root@batman bijay]# _
```

Now a firewall rule is implemented as:
'Firewall-cmd --permanent --direct --passthrough ipv4 -t nat -I POSTROUTING -o enp0s3 -j MASQUERADE -s 192.168.1.0/24'

```
[root@batman bijay]# sysctl -p /etc/sysctl.d/ip_forward.conf
net.ipv4.ip_forward = 1
[root@batman bijay]# firewall-cmd --permanent --direct --passthrough ipv4 -t nat -I POSTROUTING -o e
np0s3 -j MASQUERADE -s 192.168.1.0/24
Warning: ALREADY_ENABLED: passthrough 'ipv4', '['-t', 'nat', '-I', 'POSTROUTING', '-o', 'enp0s3', '-
j', 'MASQUERADE', '-s', '192.168.1.0/24']'
success
[root@batman bijay]#
```

(*Since I was a little late to capture the screenshot the first time, I had to type that command again, therefore the 'ALREADY_ENABLED' warning is shown*)

We can also use above command by specifying the ip with name of the device as shown:

```
[root@batman bijay]# sysctl -p /etc/sysctl.d/ip_forward.conf
net.ipv4.ip_forward = 1
[root@batman bijay]# firewall-cmd --permanent --direct --passthrough ipv4 -t nat -I POSTROUTING -o e
np0s3 -j MASQUERADE -s enp0s8
success
[root@batman bijay]# systemctl restart firewall
Failed to restart firewall.service: Unit firewall.service not found.
[root@batman bijay]# systemctl restart firewalld.service
[root@batman bijay]# _
```

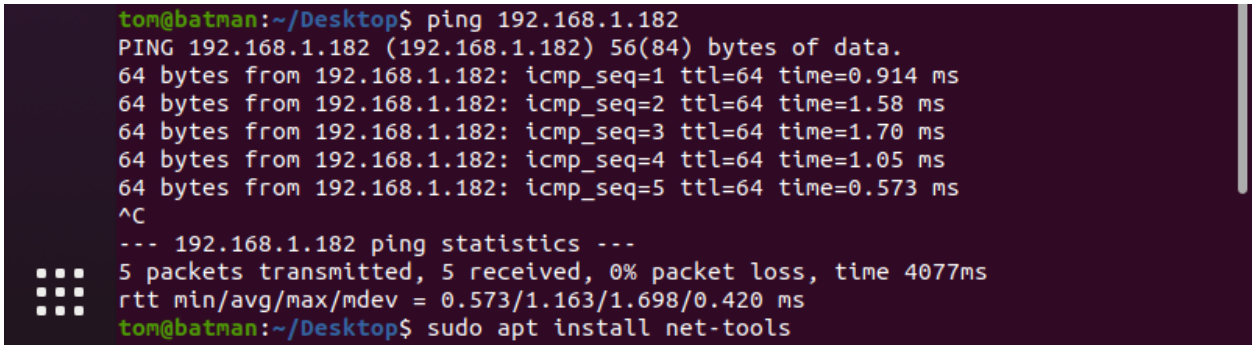
After this, the firewall is restarted using command:
'***Systemctl restart firewalld.service'***

Finally the NAT is implemented in VM1 with following commands:
'***iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE '***
iptables -A FORWARD -i eth1 -j ACCEPT '

Before all these, we couldn't ping the network, as it said 100% packet loss.

```
tom@batman:~/Desktop$ ping 192.168.1.182
PING 192.168.1.182 (192.168.1.182) 56(84) bytes of data.
^[[A^[[A^[[B^[[B^C
--- 192.168.1.182 ping statistics ---
16 packets transmitted, 0 received, 100% packet loss, time 15968ms
```

After reboot, I tried to ping this network from my VM2.



And the ping was a success.

QN1: Create a virtual machine having the os centos.

a. Install firewall in the vm(centos might have firewall installed in default).(firewalld or iptables)
b. Block certain ip range/subnet using firewalld.
c. Allow http, https and ssh connection using firewall.
d. You can add other rules as well as you prefer.

Note: The firewall rules should be saved permanently.

I have installed CentOS 8 in my Virtual Box. In CentOS, firewall is installed by default, which can be verified by the command

'***Firewall-cmd -V***'

```
[root@batman bijay]# firewall-cmd -V
0.8.2
[root@batman bijay]# _
```

And the status of firewall can be checked using the command:

'***Systemctl status firewalld***'

```
[root@batman bijay]# systemctl status firewalld
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2021-11-02 21:16:51 +0545; 1h 26min ago
     Docs: man:firewalld(1)
 Main PID: 876 (firewalld)
    Tasks: 2 (limit: 11377)
   Memory: 33.3M
   CGroup: /system.slice/firewalld.service
           └─876 /usr/libexec/platform-python -s /usr/sbin/firewalld --nofork --nopid

Nov 02 21:16:51 batman systemd[1]: Starting firewalld - dynamic firewall daemon...
Nov 02 21:16:51 batman systemd[1]: Started firewalld - dynamic firewall daemon.
Nov 02 21:16:52 batman firewalld[876]: WARNING: AllowZoneDrifting is enabled. This is considered an>
Nov 02 21:16:52 batman firewalld[876]: ERROR: '/usr/sbin/iptables-restore -w -n' failed: iptables-r>
                                       Error occurred at line: 2
                                       Try `iptables-restore -h' or 'iptables-restore --help' for m>
Nov 02 21:16:52 batman firewalld[876]: ERROR: COMMAND_FAILED: Direct: '/usr/sbin/iptables-restore ->
                                       Error occurred at line: 2
                                       Try `iptables-restore -h' or 'iptables-restore --help' for m>
Nov 02 22:27:20 batman firewalld[876]: WARNING: ALREADY_ENABLED: passthrough 'ipv4', '['-t', 'nat',>
lines 1-20/20 (END)
```

We can see the implemented firewall rules using the command:

'***Firewall-cmd --list-all***'

```
[root@batman bijay]# systemctl restart firewalld.service
[root@batman bijay]# firewall-cmd --list-all
public (active)
  target: default
  icmp-block-inversion: no
  interfaces: enp0s3 enp0s8
  sources:
  services: cockpit dhcpv6-client ssh
  ports:
  protocols:
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
```

To block certain IP or range of IP in the firewall, we can use the command to add the rich rules in firewall:
'Firewall-cmd --permanent --add-rich-rule=" rule family= 'ipv4' source address=xxx.xxx.xxx.xxx/xx reject "'
xxx.xxx.xxx.xxx/xx denotes the IP range which you want to block or reject.
To block the network, we can simply use '***block***' in place of '***reject***'.
Then we can restart the firewall and see firewall rules as shown in figure:

```
[root@batman bijay]# firewall-cmd --permanent --add-rich-rule="rule family='ipv4' source address='10
0.100.100.0/24' reject"
success
[root@batman bijay]# systemctl restart firewalld.service
[root@batman bijay]# firewall-cmd --list-all
public (active)
  target: default
  icmp-block-inversion: no
  interfaces: enp0s3 enp0s8
  sources:
  services: cockpit dhcpv6-client ssh
  ports:
  protocols:
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
        rule family="ipv4" source address="100.100.100.0/24" reject
[root@batman bijay]# _
```

Here we can see that rich rules have been updated.
Now the firewall rejects the traffic from the network 100.100.100.x.

To allow desired services(https,http and ssh) in firewall, we can use the following commands:
Firewall-cmd --add-service=https
Firewall-cmd --add-service=http
Firewall-cmd --add-service=ssh

```
[root@batman bijay]# firewall-cmd --add-service=http
success
[root@batman bijay]# firewall-cmd --add-service=https
success
[root@batman bijay]# firewall-cmd --add-service=ssh
Warning: ALREADY_ENABLED: 'ssh' already in 'public'
success
[root@batman bijay]# firewall-cmd --list-all
public (active)
  target: default
  icmp-block-inversion: no
  interfaces: enp0s3 enp0s8
  sources:
  services: cockpit dhcpv6-client http https ssh
  ports:
  protocols:
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
[root@batman bijay]#
```

We can see that the rules have been updated after allowing the services.

Also the services can be removed using the commands:
‘Firewall-cmd --remove-service=https’
‘Firewall-cmd --remove-service=http’
It will remove the previously added services http and https.