Project Report 1 - Packet Filter Firewall

Student Name: Mehran Tajbakhsh

Email: mtajbakh@asu.edu

Submission Date: 28th May, 2022

Class Name and Term: CSE548 Summer 2022

I. Project Overview

In this lab we want to implement and configure a packet filter firewall on Linux OS. We set up a network with two virtual machines. Then we set up one of them as the client machine and the other as the gateway machine (dual homed/NAT enabled). We installed a webserver on the gateway machine. In this lab we used iptables as packet filter firewall on Linux systems to control traffic between private (internal) and public (Internet) networks. We have to define the firewall rules to only allow the client machine to access our webserver and Internet through the gateway machine.

In this lab, I used the netplan tool to configure network settings in virtual machines and I used the iptables tool on Linux systems (Ubuntu) to implement a packet filter firewall.

Our goal in this lab was to set up iptables to block all traffic and only allow specific traffic to pass between clientVM and gateway/serverVM (Webserver), based on the protocol, the source, and the destination IP addresses.

II. Network Setup

I set up the network for this lab according to the steps that are mentioned in the CS-NET-00001 document. The Topology of my network is shown below. I created a network with two stations (Client and Gateway/Server). We wanted the client node to only be able to access the Internet through the Gateway/Server node, and also we wanted the Client node to be able to access the webserver that we set up on the Gateway/Server node based on the rules we defined in the firewall (iptables) on the Gateway/Server node. As shown in the network topology, the Client node works on the 10.0.2/24 network (internal Network/inet) and I assigned a static address (10.0.2.2/24) to it and the Gateway/Server node works on two network interfaces. I set up the first network interface on 10.0.2/24 network and the Gateway/Server node connected to the Client node via this interface, and I set up the second interface on the Gateway/Server node on 10.0.1/24 network that will be used to connect to the external network (Internet).

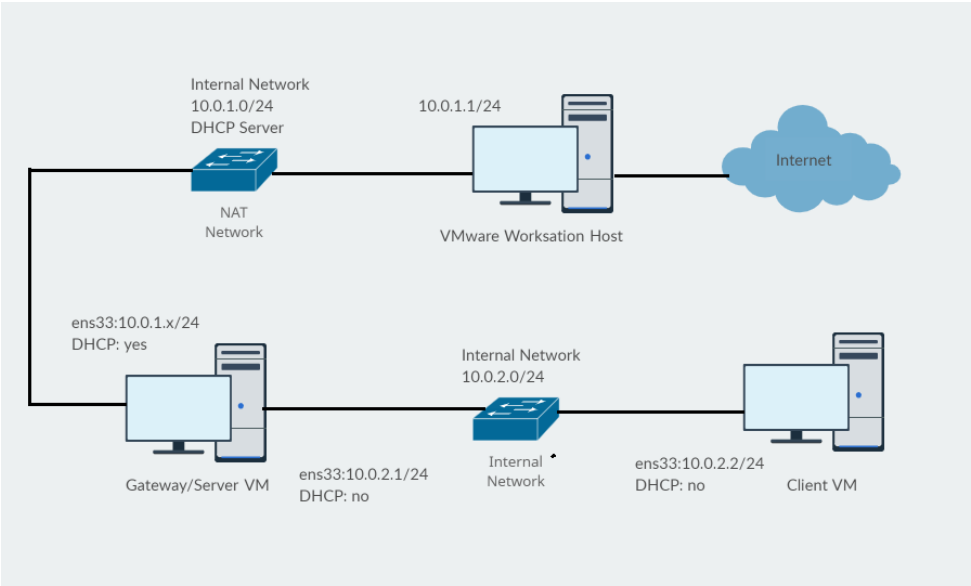

Figure-1 Network topology

For the purpose of this lab I assigned IP addresses to interface(s) in each node as follows:

Client node	Reachability
Interface: ens33 IP address: 10.0.2.2/24 Gateway: 10.0.2.1	Gateway/Server node First interface (ens33)
Gateway node	
Interface: ens33 IP address: 10.0.2.x/24 (DHCP) Gateway: - Interface: ens34: IP address: 10.0.1.1/24 Gateway:	Client Node WMware Workstation Host IP address

I used VMware workstation Pro (16.2.3 build-19376536) as a desktop hypervisor to create virtual machines. Then I created two virtual machines based on Ubuntu OS (Gateway/ServerVM and ClientVM), we can download the Ubuntu ISO image here [1]. Then I used the Ubuntu ISO image to create two virtual machines. I created the first virtual machine (ClientVM) with one network interface adapter and I created the second virtual machine (Gateway/ServerVM) with two network interface adapters. Before I started configuring network interfaces on the virtual machines, I needed to set up a host virtual network to host the machine, and the virtual machines use it to connect with each other and to the public network (Internet).
In VMware Workstation from the Edit menu choose Virtual Network Editor (Figure-2)

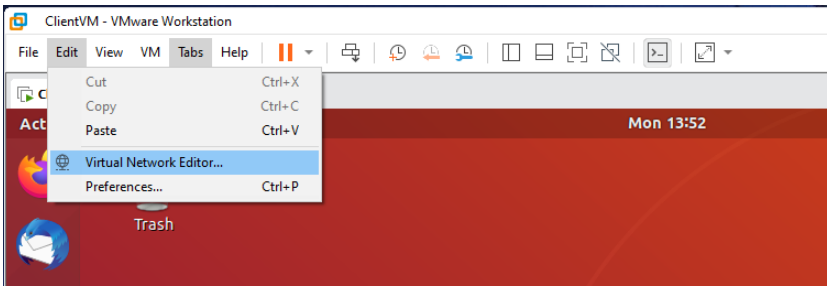

Figure-2 VMware workstation Virtual Network Editor

We need Administrator privileges to change the settings in Virtual Network Editor (Figure-3). I set up two virtual networks (VMnet1 and VMnet8) that are used to connect the virtual machines in the private (VMnet1) and public (VMnet8) networks. I used 10.0.2.0/24 address as the network address for the private network and I defined 10.0.1.0/24 as the network address to use the Gateway/ServerVM machine, the host machine, and the Internet (Figure-4).

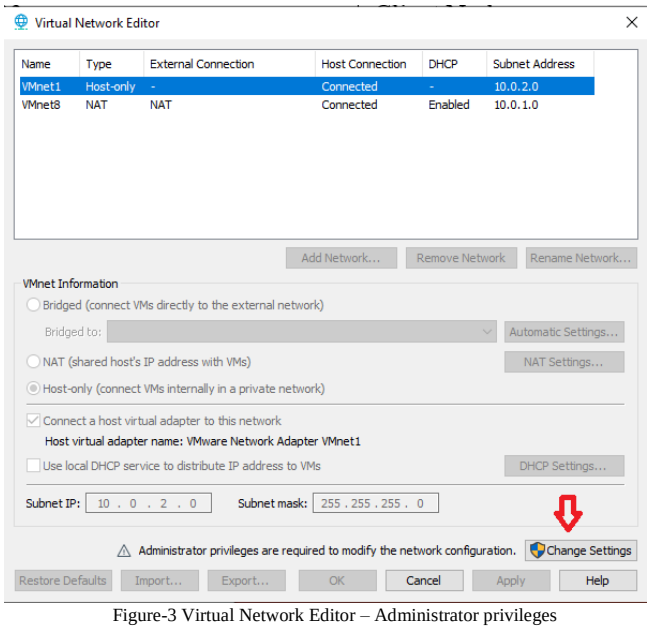

Figure-3 Virtual Network Editor – Administrator privileges

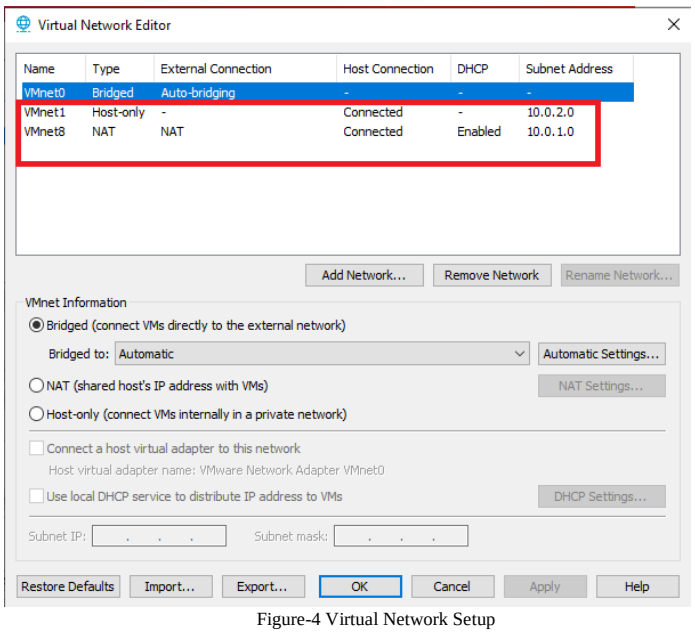

Figure-4 Virtual Network Setup

As you can see in the screenshot below, I disabled the DHCP service in the internal network (VMnet1) and I enabled and set up the DHCP service to be used on the public (Internet) network.

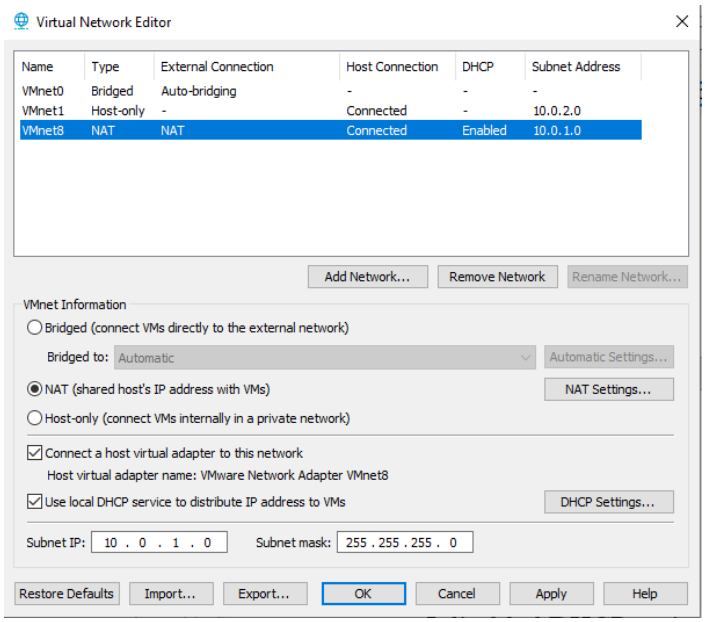

Figure-5 DHCP service setup in VMnet8 virtual network

I changed the Network Adapter settings in the ClientVM virtual machine as shown in Figure-6.

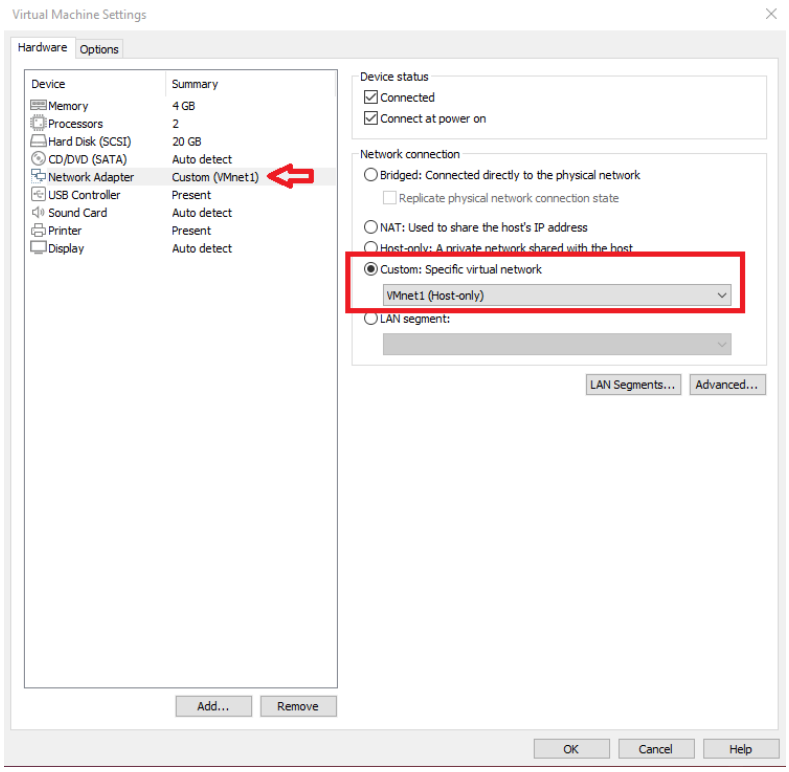

Figure-6 ClentVM network settings

I changed the network settings for two network adapters in the Gateway/ServerVM virtual machine as shown in the screenshot below (Figure7, Figure 8)

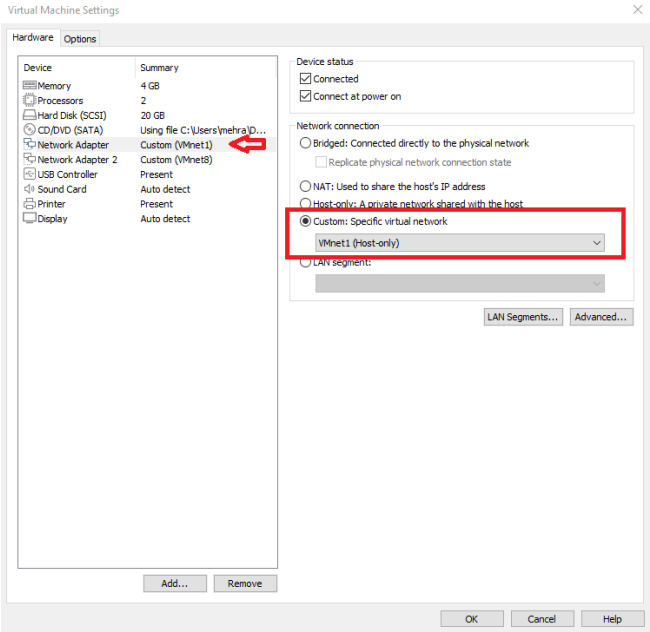

Figure-7 First network adapter settings in Gatey/ServerVM

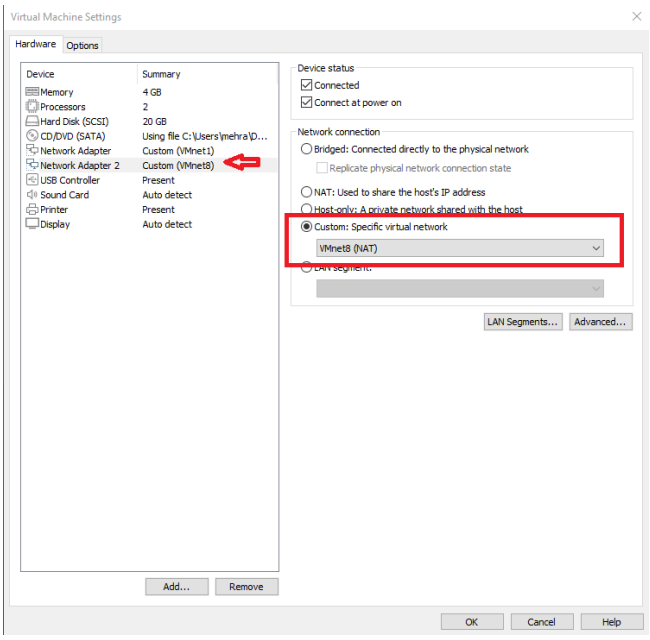

Figure-8 Second network adapter settings in Gateway/SErverVM

III. Software

- Ubuntu network command line tools (ifconfig, ping, route, …) – (sudo apt install net-tools)
- Ubuntu built-in firewall (iptables)
- Ubuntu Apache Web Server (sudo apt install apache2)
- Packet sniffing, capturing and analyzing tools (tcpdump, nmap) (sudo apt install tcpdump and sudo apt install nmap)

IV. Project Description

Before I started, I changed the background color of the terminal window in the ClientVM virtual machine to gray. This differentiates the screenshots from the two virtual machines during this lab.

A. Initial setup and connectivity tests

I used the netplan tool in the Ubuntu machines to configure the network settings in ClientVM and Gateway/ServerVM. The netplan uses a Configuration file (**01-network-manager-all.yaml**) that is located in the /etc/netplan [6]. I used a nano tool in the command line to change the content of this file in the ClientVM and Gateway/ServerVM as follows:
I used the following command in the terminal window to configure the network settings in the ClientVM virtual machine

sudo nano /etc/netplan/01-network-manager-all.yaml (Figure-9)

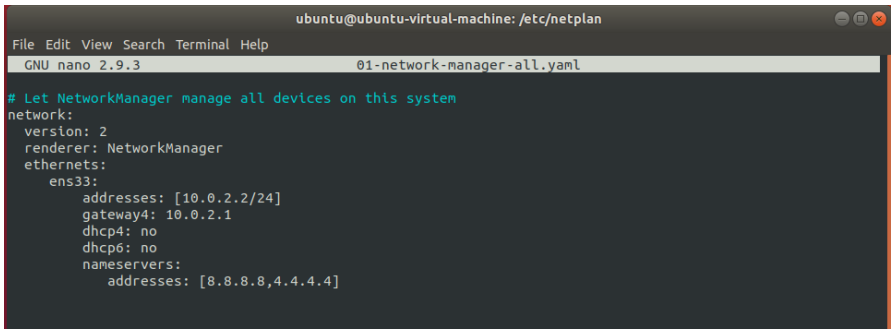

Figure-9 Edit 01-network-manager-all.yaml

Client VM:

network:
  renderer: networkd
  version: 2
  ethernets:

```
    enp0s3:
      addresses: [10.0.2.2/24]
      gateway4: 10.0.2.1
      dhcp4: no
      dhcp6: no
      nameservers:
     addresses: [8.8.8.8,4.4.4.4]
```

I used the following command in the terminal window to configure the network settings in the Gateway/ServerVM virtual machine:

sudo nano /etc/netplan/01-network-manager-all.yaml (Figure-10)

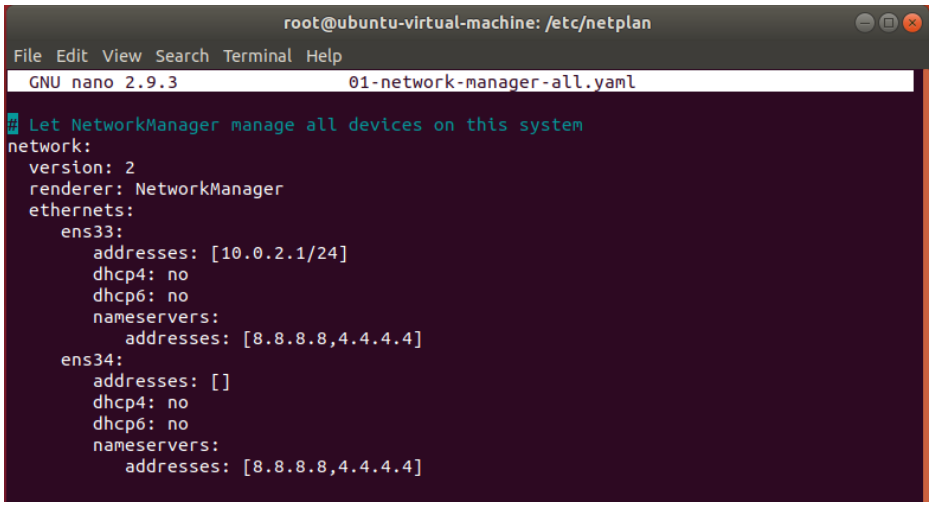

Figure-10 Edit 01-network-manager-all.yaml

Gateway/Server VM:

```
network:
  renderer: networkd
  version: 2
  ethernets:
    enp0s3:
      addresses: [10.0.2.1/24]
      dhcp4: no
      dhcp6: no
      nameservers:
        addresses: [8.8.8.8,4.4.4.4]
    enp0s8:
      addresses: []
      dhcp4: yes
      dhcp6: no
      nameservers:
     addresses: [8.8.8.8,4.4.4.4]
```

I disabled the DHCP service in the internal network (10.0.2/24) and I also used 8.8.8.8 and 4.4.4.4 (Google DNS servers) IP addresses as DNS servers for my VMs. Then I used the following command in both VMs to check the network settings:

sudo netplan try

Then I used the following command to change the network setting in both VMs:

sudo netplan apply

I used the following command in the terminal window to activate the network settings in both VMs :

sudo service networking restart

I used the ifconfig command to see and check the network settings in the Gateway/ServerVM (Figure-11):

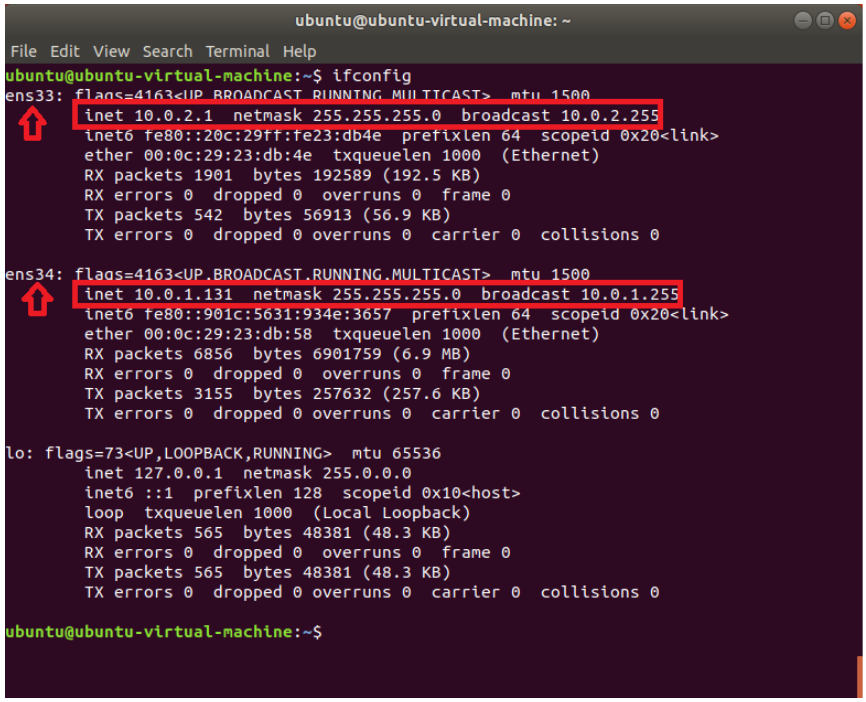

Figure-11 Gateway/Server VM network settings

B. Configuring the network and the webserver in the Gateway/Server VM:

Step 1: Install Apache Webserver

I used the following commands in the terminal window to update the system and install Apache Webserver on the Gateway/ServerVM virtual machine (Figure-23-25):

sudo apt update
sudo apt –y dist-upgrade
sudo apt install apache2

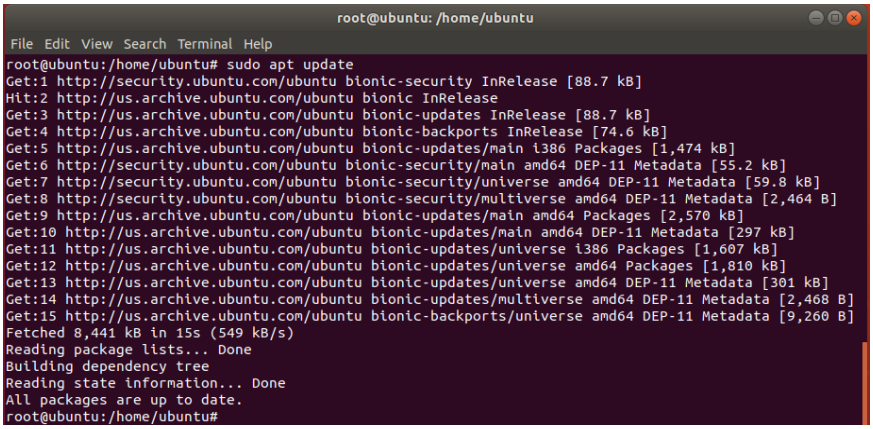

Figure-23 Updating Ubuntu

```
root@ubuntu: /home/ubuntu
File  Edit  View  Search  Terminal  Help
root@ubuntu:/home/ubuntu# sudo apt -y dist-upgrade
Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating upgrade... Done
The following package was automatically installed and is no longer required:
  linux-hwe-5.4-headers-5.4.0-42
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
root@ubuntu:/home/ubuntu#
```

Figure-24 Updating Ubuntu

```
root@ubuntu: /home/ubuntu
File  Edit  View  Search  Terminal  Help
Setting up apache2 (2.4.29-1ubuntu4.22) ...
Enabling module mpm_event.
Enabling module authz_core.
Enabling module authz_host.
Enabling module authn_core.
Enabling module auth_basic.
Enabling module access_compat.
Enabling module authn_file.
Enabling module authz_user.
Enabling module alias.
Enabling module dir.
Enabling module autoindex.
Enabling module env.
Enabling module mime.
Enabling module negotiation.
Enabling module setenvif.
Enabling module filter.
Enabling module deflate.
Enabling module status.
Enabling module reqtimeout.
Enabling conf charset.
Enabling conf localized-error-pages.
Enabling conf other-vhosts-access-log.
Enabling conf security.
Enabling conf serve-cgi-bin.
Enabling site 000-default.
Created symlink /etc/systemd/system/multi-user.target.wants/apache2.service → /lib/systemd/system/ap
ache2.service.
Created symlink /etc/systemd/system/multi-user.target.wants/apache-htcacheclean.service → /lib/syste
md/system/apache-htcacheclean.service.
Processing triggers for libc-bin (2.27-3ubuntu1.5) ...
Processing triggers for systemd (237-3ubuntu10.53) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
Processing triggers for ufw (0.36-0ubuntu0.18.04.2) ...
Processing triggers for ureadahead (0.100.0-21) ...
root@ubuntu:/home/ubuntu#
```

Figure-25 Installing Apache Webserver

Step 2: Set up virtual host and create a landing page to test the Webserver

I want to create a new VirtualHost on my webserver (my-webserver.com), so I copied of the 000-defualt.conf file in a new file (my-webserver.conf) and I made the following changes on it (Figure-26):

```
root@ubuntu-virtual-machine: /etc/apache2/sites-available
File  Edit  View  Search  Terminal  Help
  GNU nano 2.9.3                    my-webserver.conf

<VirtualHost *:80>
   ServerName my-webserver.com
   ServerAlias *.my-webserver.com

   ServerAdmin mtajbakh@asu.edu
   DocumentRoot /var/www/lab1/html

   ErrorLog ${APACHE_LOG_DIR}/error.log
   CustomLog ${APACHE_LOG_DIR}/access.log combined

</VirtualHost>
```

Figure-26 my-webserver.conf file (virtual host)

Then, I edited the file /etc/apache2/ports.conf as shown below (Figure-27), Changes this to 127.0.0.1:80 cause to webserver to listen only on loopback address, so it will not be available to the Internet.

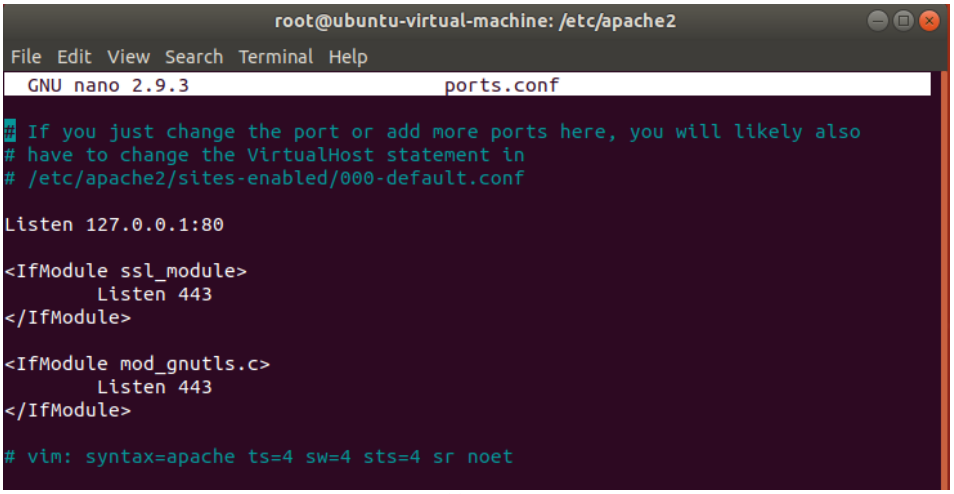

Figure-27 ports.conf file

I created folders that I need for the new VitualHost as follow:

sodu cd /var/www
sudo mkdir lab1
sudo cd lab1
sudo mkdir html

I created the file index.html in "/var/www/lab1/html" path, as shown in the following screenshot (Figure-28).

nano /var/www/lab1/html/index.html

Figure-28 Index.html file

Step 3: Activate the new virtual host
I changed Listen Directive in file /etc/apache2/ports.conf as follow:

Listen 127.0.0.1:80

Then, enable the new VirtualHost using the following command:

sudo a2ensite my-webserver.com
sudo systemctl restart apache2.service

Step 4: Test Webserver connection in the Gateway/ServerVM virtual machine:
Then I browsed the URL: http://my-webserver.com in the localhost (Gateway/ServerVM) virtual machine, I saw the following page in my browser (Figure-29):

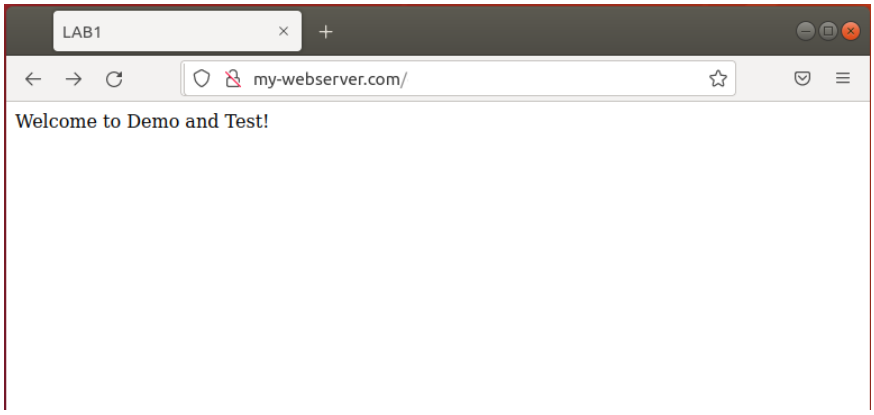

Figure-29 index.html page

C: Disable UFW (Uncomplicated Firewall) / Install required software:

In this lab we used iptables as firewall, so we needed to disable UFW on the Gateway/ServerVM virtual machine (Figure-30):

service ufw stop

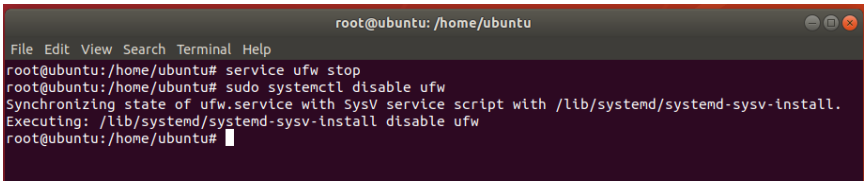

Figure-30 Disable UFW

Install nmap [5]
To install nmap use the following command (Figure-31):

sudo apt install nmap

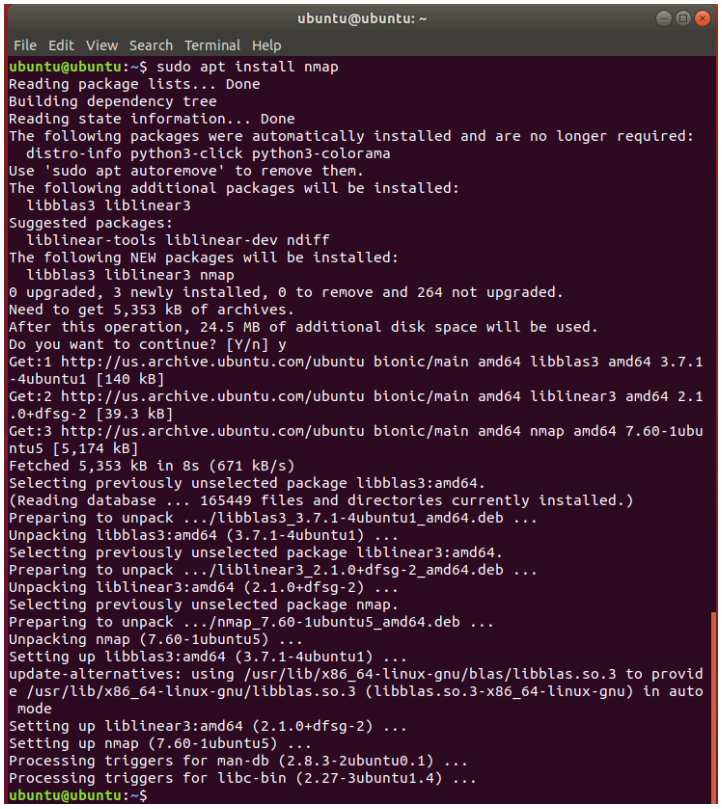

Figure-31 Install nmap

(30 points) The Gateway/Server VM should
- set up http(webpage) service to it's own IP address (with the demo page available).
- enable POSTROUTING to allow client to access outside network(8.8.8.8) and change their source IP addresses.

D. Check initial network settings in the Gateway/Server VM

As shown in the following picture, at this point nothing was configured on the Gateway/ServerVM to forward traffic (Figure-12):

cat /proc/sys/net/ipv4/ip_forward

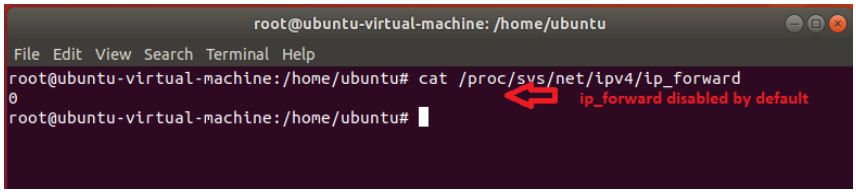

Figure-12 forward traffic setting

I checked the routing table in the Gateway/ServerVM as follows (Figure-13):

route -nv

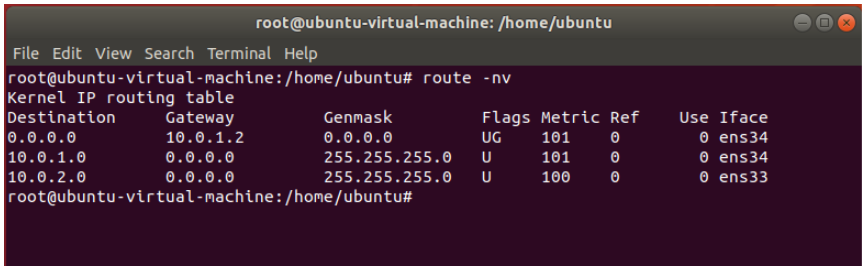

Figure-13 Gateway/Server VM routing table

I checked the iptables firewall rules at the beginning of the lab. As shown in the following screenshot, no rules had been defined (Figure-14):

sudo iptables –L –v –n

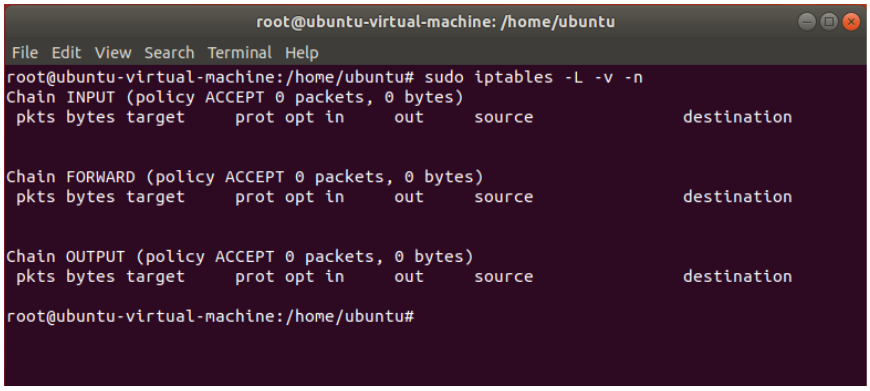

Figure-14 iptables settings

At this point I checked the Gateway/ServerVM's connection to the Internet as follows (Figure-15), and as expected we had

Internet connection in the Gateway/ServerVM virtual machine:

ping google.com

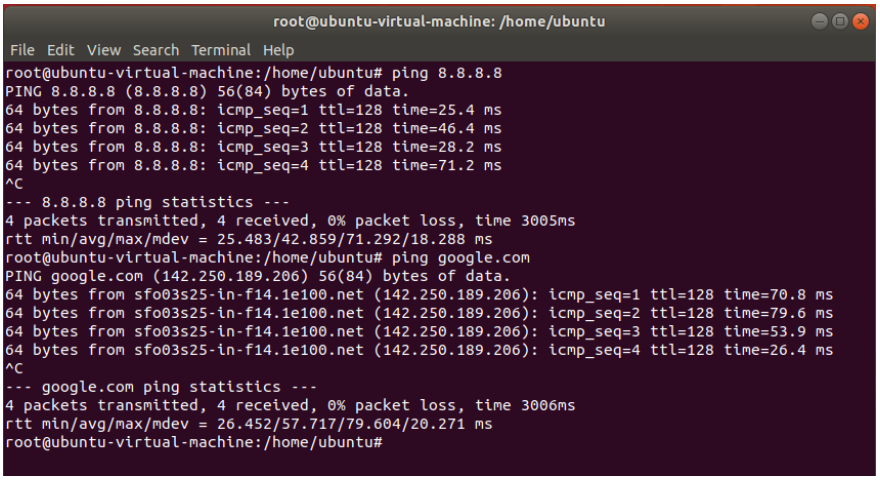

Figure-15 Gateway/Server VM check Internet connection

(30 points) The client

- can not ping the Gateway/Server VM IP address
- can access the demo webpage on Gateway/Server VM by access the IP address of Gateway/Server VM in browser (the returning page must contain ”Welcome ....”, you can also use a web browser)
- can ping 8.8.8.8.

E. Check initial network settings in the ClientVM virtual machine:

I used the ifconfig command to see and check the network settings in the ClientVM virtual machine (Figure-16):

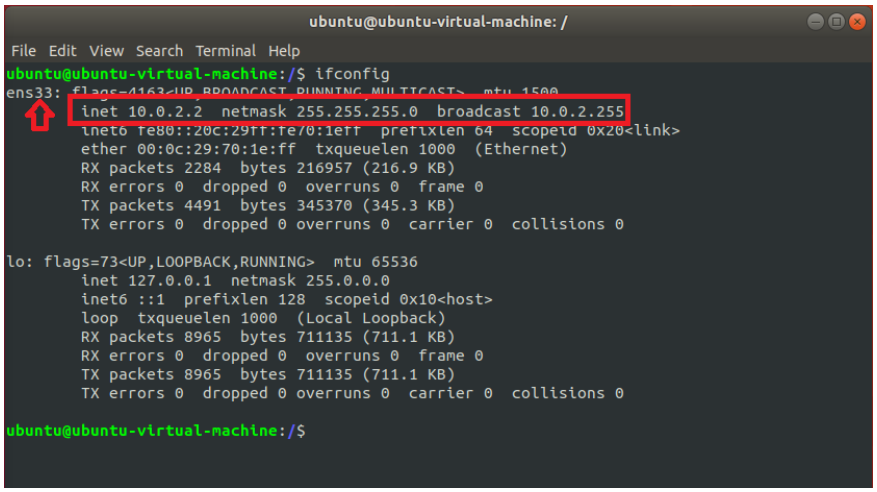

Figure-16 Client VM network settings

I checked the routing table in the ClientVM virtual machine as follows (Figure-17):

route -nv

```
ubuntu@ubuntu-virtual-machine: /
File  Edit  View  Search  Terminal  Help
ubuntu@ubuntu-virtual-machine:/$ route  -nv
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.2.1        0.0.0.0         UG    20100  0        0 ens33
10.0.2.0        0.0.0.0         255.255.255.0   U     100    0        0 ens33
ubuntu@ubuntu-virtual-machine:/$
```

Figure-17 Client VM routing table

I checked the connection between the ClientVM virtual machine and the Internet as shown in the screenshot below (Figure-18), as expected my ClientVM virtual machine was located in the internal network and only used one route to access the public network through the Gateway/ServerVM virtual machine on which IP forwarding had initially been disabled:

ping 8.8.8.8

```
ubuntu@ubuntu-virtual-machine: /
File  Edit  View  Search  Terminal  Help
ubuntu@ubuntu-virtual-machine:/$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
^C
--- 8.8.8.8 ping statistics ---
7 packets transmitted, 0 received, 100% packet loss, time 6143ms

ubuntu@ubuntu-virtual-machine:/$ ping google.com
^C
ubuntu@ubuntu-virtual-machine:/$
```

Figure-18 Check Internet connection on ClientVM virtual machine

I had a direct connection between the network adapter in theClientVM virtual machine (ens33) and the first network adapter in the Gateway/ServerVM virtual machine (ens33) and these network adapters were configured in the same network, so I checked the connectivity between them as shown in the screenshot below (Figure-19):

ping 10.0.2.1 (Gateway/ServerVM virtual machine IP address)

```
ubuntu@ubuntu-virtual-machine: /
File  Edit  View  Search  Terminal  Help
ubuntu@ubuntu-virtual-machine:/$ ping 10.0.2.1
PING 10.0.2.1 (10.0.2.1) 56(84) bytes of data.
64 bytes from 10.0.2.1: icmp_seq=1 ttl=64 time=0.575 ms
64 bytes from 10.0.2.1: icmp_seq=2 ttl=64 time=0.598 ms
64 bytes from 10.0.2.1: icmp_seq=3 ttl=64 time=0.557 ms
64 bytes from 10.0.2.1: icmp_seq=4 ttl=64 time=0.441 ms
Software from 10.0.2.1: icmp_seq=5 ttl=64 time=0.550 ms
^C
--- 10.0.2.1 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4092ms
rtt min/avg/max/mdev = 0.441/0.544/0.598/0.056 ms
ubuntu@ubuntu-virtual-machine:/$
```

Figure-19 Check connectivity between ClientVM and Gateway/ServerVM VMs

I wanted to create a path to allow the ClientVM virtual machine to access the Internet, So I Enabled packet forwarding in the Gateway/ServerVM virtual machine (Figure 20):

sudo sysctl –w net.ipv4.ip_forward=1

```
ubuntu@ubuntu-virtual-machine: ~
File  Edit  View  Search  Terminal  Help
ubuntu@ubuntu-virtual-machine:~$ sudo sysctl -w net.ipv4.ip_forward=1
[sudo] password for ubuntu:
net.ipv4.ip_forward = 1
ubuntu@ubuntu-virtual-machine:~$
```

Figure-20 Pack forwarding in Gateway/Server VM

Then I needed to enable the NAT rules on the Gateway/ServerVM virtual machine to allow our ClientVM virtual machine to connect to the Internet (Figure-21):

iptables –A FORWARD –j ACCEPT
iptables –t nat –A POSTROUTING –o enp0s8 –j MASQUERADE
iptables –t nat –L POSTROUTING –n –v –line-number

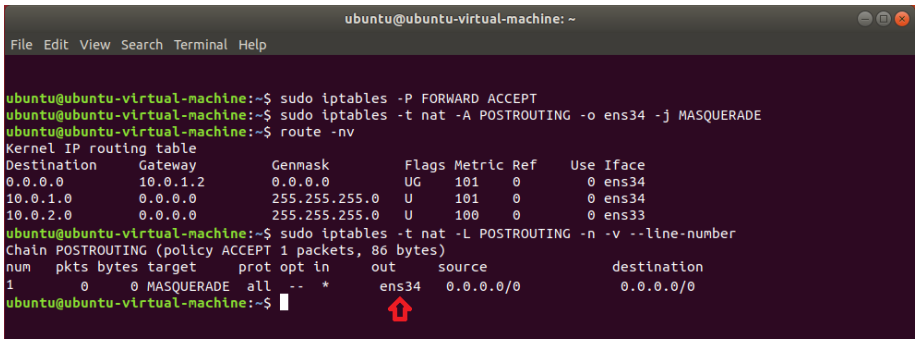

Figure-21 Enable NAT rules on Gateway/ServerVM virtual machine

Now as expected, the ClientVM virtual machine had access to the external network (Internet) as shown in the screenshot below (Figure-22):

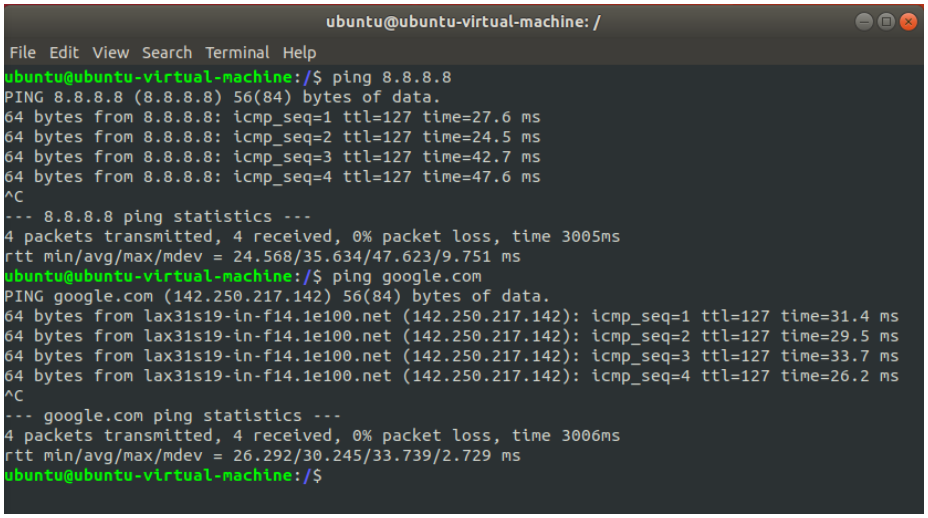

Figure-22 Check Internet reachability in the Client VM

(40 points) Additional requirements

- You should set the default firewall policy to DROP for INPUT, OUTPUT, and FORWARD chains.
- Besides the allowed network access described the above, you should not allow any other network access. Provide screenshots for the following results: On client VM:

  $ sudo nmap -sT -p- 10.0.2.x % x is the value of your Gateway/Server VM's IP address
  $ sudo nmap -sU -p- 10.0.2.x % x is the value of your Gateway/Server VM's IP address
  $ ping 8.8.8.8 % This should be working
  $ ping 8.8.4.4 % This should be not working, as you should drop all traffic that is not required in the requirement.
  $ ping 10.0.2.x % x is the value of your Gateway/Server VM's IP address, This should be not working On

Gateway/Server VM:

  $ ping localhost % This should be not working
  $ ping 10.0.2.y % y is the value of your client's IP address,this should be not working
  $ ping 8.8.8.8 % This should be not working

F. Setup the Packet Filter Firewall (iptables):

Step 1: I used the rc.firewall file that provided for this. I maked it as an executable file as show in the following screenshot (Figure 32):

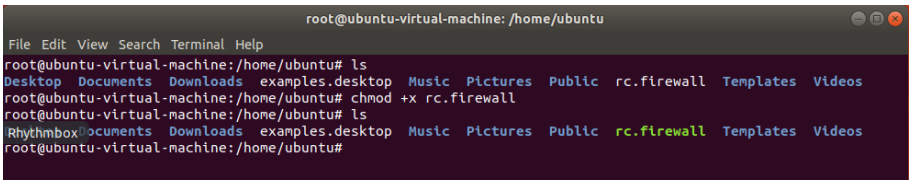

Figure-32 rc.firewall

Step 2: I used the nano tool to edit the rc.firewall (Figure 33):

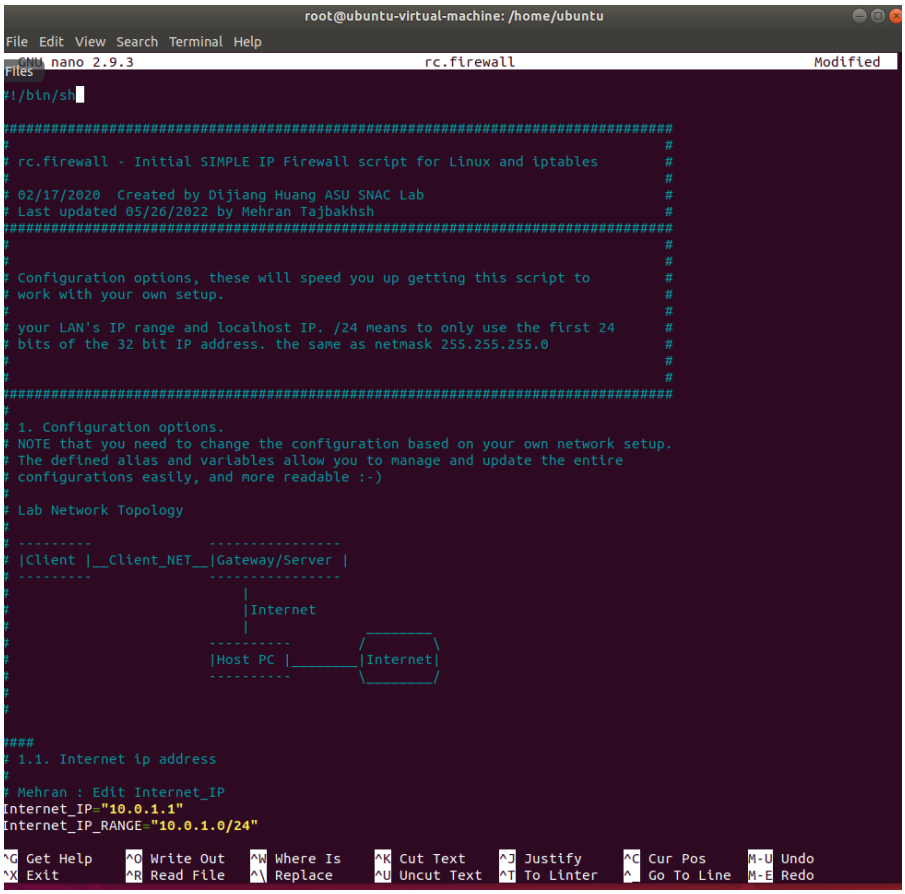

Figure-33 edit rc.firewall

Step 3: based on the lab requirements, I edited the rc.firewall to configure access policies for both VMs in the firewall [2] as shown in screenshot blow (Figure-34):

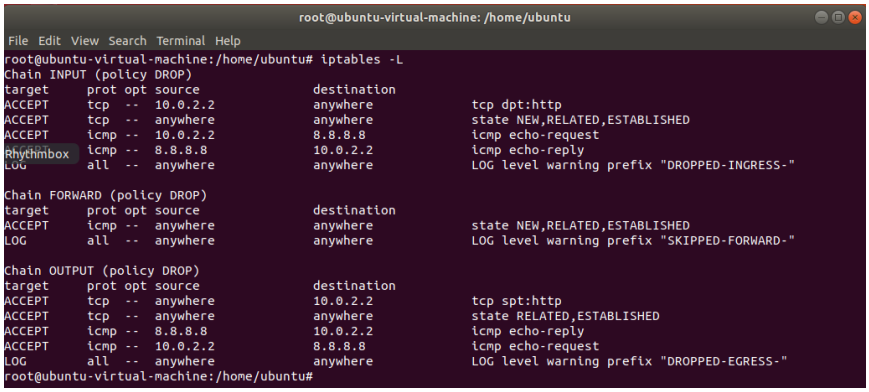

Figure-34 iptables rules

I provided the link to download the final version of the rc.firewall file in the Appendix B.
Step 4: Test the firewall configurations:
I checked the connection between the ClientVM the Gateway/SeverVM webserver, I browsed the Gateway/ServerVM's IP address in the ClientVM's browser as shown below (Figure-35):

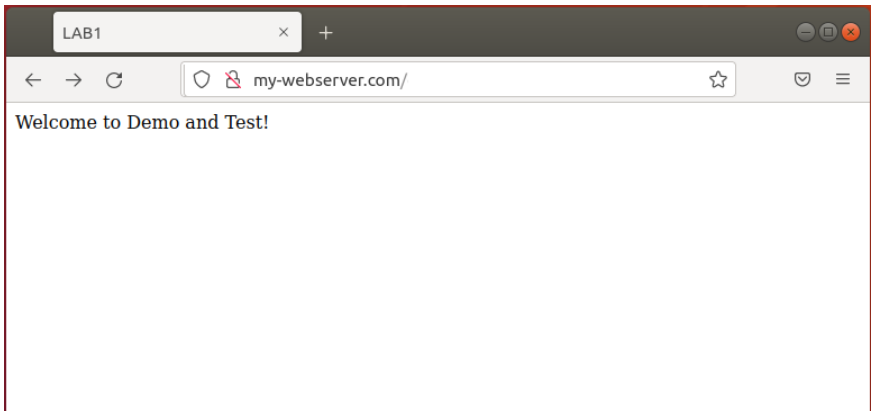

Figure-35 index.html page

```
$ sudo nmap -sT -p- 10.0.2.x % x is the value of your Gateway/Server VM's IP address
```

sudo nmap -sT -p- 10.0.2.1 % x is the value of your Gateway/Server VM's IP address

I executed this command in the ClientVM's terminal window. This command scans all (TCP) ports using TCP connect in the target machine (10.0.2.1), as shown in the following screenshot (Figure-36)

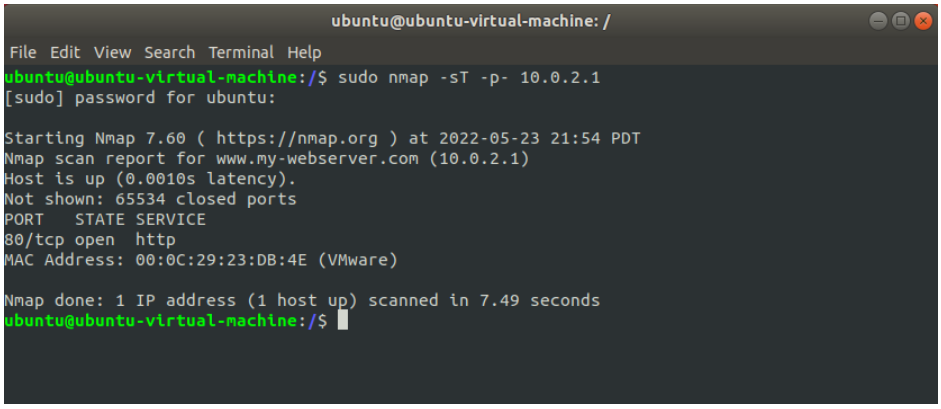

Figure-36 namp full scan TCP ports

$ sudo nmap -sU -p- 10.0.2.x % x is the value of your Gateway/Server VM's IP address

sudo nmap -sU -p- 10.0.2.1 % x is the value of your Gateway/Server VM's IP address

I executed this command in the ClientVM's terminal window. This command scans all (UDP) ports in the target machine (10.0.2.1), as shown in the following screenshot (Figure 37):

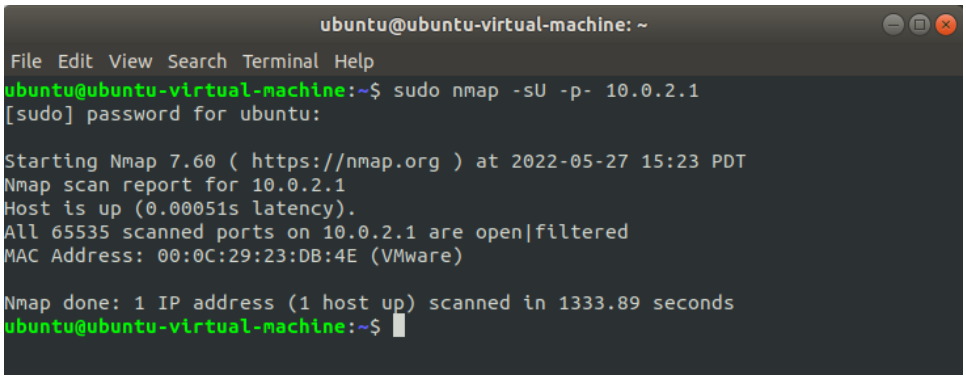

Figure-37 nmap full scan UDP ports

$ ping 8.8.8.8 % This should be working

ping 8.8.8.8 % This should be working

I executed this command in the ClientVM's terminal window. This command shows that the ClientVM virtual machine had access to the public DNS server (8.8.8.8), as shown in the following screenshot (Figure 38):

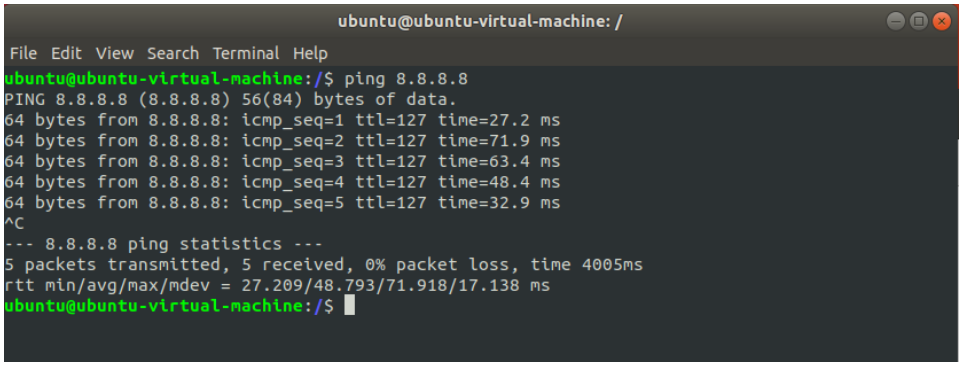

Figure-38 ping 8.8.8.8

$ ping 8.8.4.4 % This should be not working, as you should drop all traffic that is not required in the requirement.

ping 8.8.4.4 % This should be not working, as you should drop all traffic that is not required in the requirement.

I executed this command in the ClientVM's terminal window. This command shows that the ClientVM virtual machine did not have access to the address 8.8.4.4, as shown in the following screenshot (Figure 39):

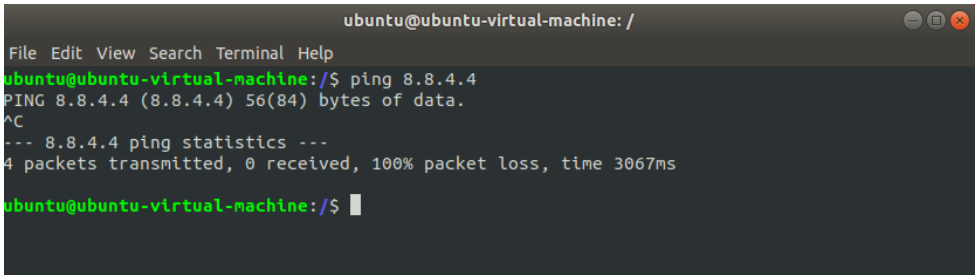

Figure-39 ping 8.8.4.4

$ ping 10.0.2.x % x is the value of your Gateway/Server VM's IP address, This should be not working On

ping 10.0.2.x % x is the value of your Gateway/Server VM's IP address, This should be not working

I executed this command in the ClientVM's terminal window. This command shows that the ClientVM virtual machine could not ping the Gateway/ServerVM virtual machine, as shown in the following screenshot (Figure 40):

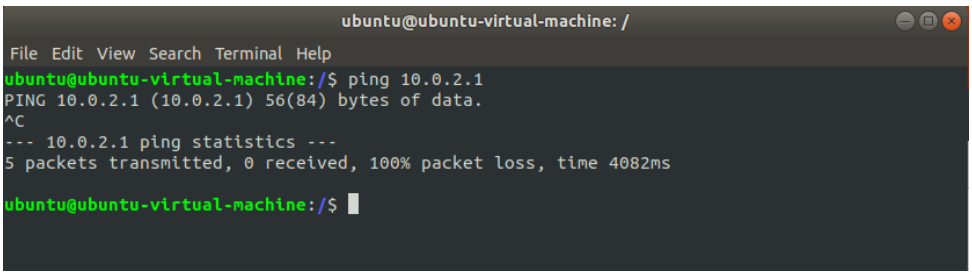

Figure-40 ping Gateway/ServerVM (10.0.2.1)

$ ping localhost % This should be not working

ping localhost % This should be not working

I executed this command in the Gateway/ServerVM's terminal window. This command shows that the loopback traffic was blocked in the Gateway/ServerVM virtual machine, as shown in the following screenshot (Figure 41):

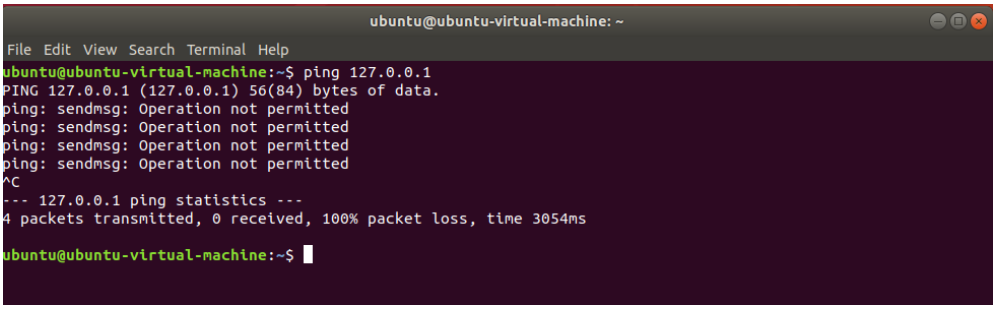

Figure-41 ping loopback (127.0.0.1)

$ ping 10.0.2.y % y is the value of your client's IP address,this should be not working $ ping 8.8.8.8 % This should be not working

ping 10.0.2.2 % y is the value of your client's IP address,this should be not working

I executed this command in the Gateway/ServerVM's terminal window. This command shows that the Gateway/ServerVM virtual machine could not ping the ClientVM virtual machine, as shown in the following screenshot (Figure 42):

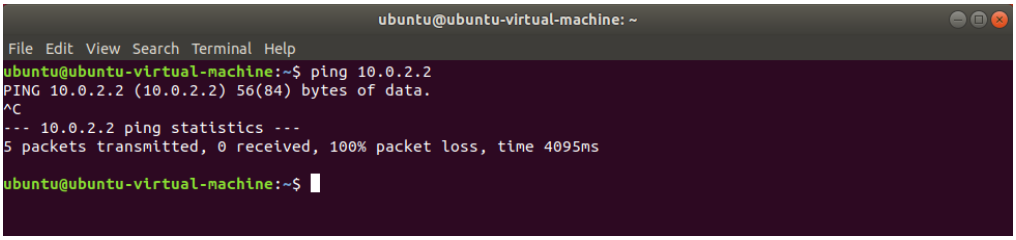

Figure-42 ping ClienVM (10.0.2.1)

$ ping 8.8.8.8 % This should be not working

ping 8.8.8.8 % This should be not working

I executed this command in the Gateway/ServerVM's terminal window. This command shows that the Gateway/ServerVM virtual machine did not have access to the public DNS server (8.8.8.8), as shown in the following screenshot (Figure 43):

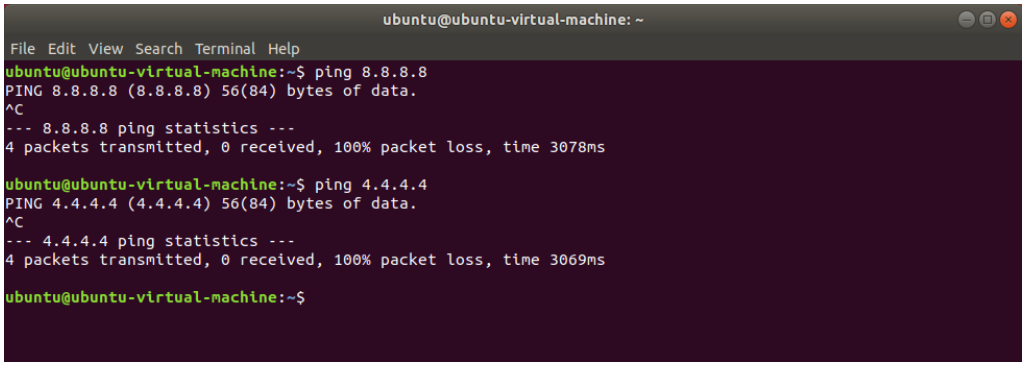

Figure-43 ping 8.8.8.8

V. Conclusion

iptables is an extremely flexible firewall utility built for Linux machines. iptables is a command-line firewall utility that uses policy chains (INPUT/OUTPUT/FORWARD) to allow or block traffic.
There are GUI alternatives to iptables like Shorewall, Firestarter, and Firewall Builder.
iptables uses logging systems to create a log file for all packets filtered by iptabels. We can enable the logging system by putting logging rules at the end of each chain. Logging systems help us monitor ingress or egress traffic to our machine and find any mistakes in iptables's rules.
We can use the following rule to enable logging for the INPUT chain:

iptables –A INPUT –j LOG

iptables logs are generated by the kernel and we can use the following command to view the log file in the Ubuntu machine:

tail –f /var/log/kern.log

When we want to define iptables's rules, it is highly recommended to follow the best practices listed below [3][4]:

- Use both whitelisting and blacklisting methodologies in firewall rule definitions to get the most benefit from each group
- Set up lo interface, because a lot of applications require access to the lo interface.
- Split complicated rules into separate chains.
- Use REJECT until you know your rules are working properly.
- Be stringent with your rules.
- Use comments for obscure rules.
- Always save your rules.

After we define our iptables's rules and are confident that these rules are working properly, we need to save our rules and make them persistent after rebooting. Firstly we need to install the "iptables-repsistent" package using the following command:

sudo apt install iptables-persistent

All the iptables rules that we already defined will be saved to the corresponding IPv4 and IPv6 files below:

/etc/iptables/rules.v4
/etc/iptables/rules.v6

To update iptables's rules and make changes permanent after rebooting use the following command:

sudo iptables- save > /etc/iptables/rules.v4
sudo iptables-save > /etc/iptables/rules.v6

VI. Appendix B: Attached files

rc.sh	https://github.com/MehranTJB/ASU-CSE5448-Advanced-Network-Security/blob/main/rc.firewall

VII. REFERENCES

[1] Ubuntu – VMware image, available at https://releases.ubuntu.com/18.04.6/?_ga=2.181101024.690550287.1653339662-1661521348.1652648283 accessed by 5/13/2022

[2] iptables rules, available at https://linuxconfig.org/how-to-make-iptables-rules-persistent-after-reboot-on-linux accessed by 5/23/2022

[3] iptables Best Practices, available at https://major.io/2010/04/12/best-practices-iptables/ accessed by 5/24/2022

[4] iptables whitelists and blacklists, available at https://nationalcybersecuritysociety.org/wp-content/uploads/2018/03/FACT-Whitelist_Blacklist-FINAL.pdf accessed by 24/5/2022

[5] nmap tutorial, available at https://nmap.org/book/port-scanning-tutorial.html accessed by 5/25/2022

[6] netplan,available at https://netplan.io/ accessed by 5/15/2022