

Student Name:
Course: CSCE 5550
Semester: Fall 2021

Firewalls and IDS Lab

Lab 4: Firewalls and IDS

In this lab, you will be using a special version of the **Ubuntu 20** VM.
Download the respective OVA file from the lab page and import it as usual.

The credentials are the same as before:

Username: sec-lab
Password: untccdc

Also, you will use the Kali 20 VM, which will be the same as in the previous labs.

The credentials are:

Username: osboxes
Password: osboxes.org

NOTE: Address all the questions (Q1, Q2, etc.) marked in bold.

When a screenshot is requested, try to fit all the results in one image.

If this is not possible, then attach multiple screenshots.

When a question is asked, e.g., “Who is an owner of the file?”, type your answer, do not simply provide a screenshot.

IMPORTANT: Before running the lab, customize the command prompt in the Ubuntu VM to show your EUID – refer to the manual provided on the lab page.

Introduction

A firewall is a software- or hardware-based network security system that controls the incoming and outgoing network traffic based on applied set of rules.

Firewalls are classified into different types: Software firewall, hardware firewall, host-based, network-based, etc. Most of the operating systems have a software firewall to protect against network threats. Hardware firewall is a dedicated piece of hardware which protects one or more networks against threats originating from the Internet. Many routers that connect networks have some firewall features built into

Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab

them. Host-based firewalls are those that are present in a single host and that only manage the incoming and outgoing traffic for that host. A software firewall within an OS is a typical host-based firewall.

General-purpose computers such as a PC or server (or even a VM) with multiple network adapters can be turned into a router or firewall by setting up suitable software on it. For example, a Linux PC with two or more network adapters can be made a firewall by installing firewall software such as **iptables**.

Some other network related terms you need to understand to do this lab are the following:

Subnet: A subnet is a logical group of IP addresses or a subdivision of an IP network. The subnet mask, 32-bit values define the range of IP address in that subnet. For nodes in two different subnets to communicate, a router has to be present between them.

Port Forwarding: Port forwarding is a way of making a computer on your home or business network accessible to computers on the Internet, even though they are behind a router. It is commonly used in gaming, security camera setup, voice over IP, and file download. After you have forwarded a port you are said to have an open port. In this lab, we will make your VM accessible to your host computer, and to the computers that belong to your local network.

UFW (Uncomplicated Firewall): This is a default firewall configuration tool for Ubuntu Linux. Developed to ease iptables firewall configuration, UFW provides a user-friendly way to create an IPv4 or IPv6 host-based firewall. By default, UFW is disabled.

Configuration of port forwarding in the VM

Let us configure the VirtualBox VM network mode as “NAT”. Your host OS (i.e., the OS of your host computer running VirtualBox) will access to the VM through the port forwarding configured in the network setting of the VM.

1. Setup Ubuntu VM network settings as internal:

In VirtualBox: Select the Ubuntu VM → Right-click → Settings → Network.

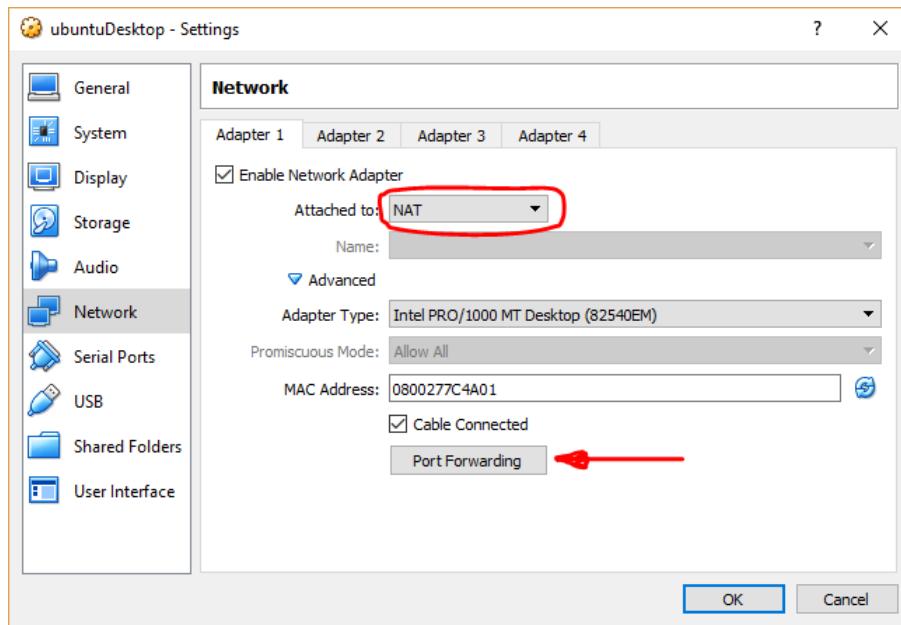
2. Under Adapter1, select “NAT” from the dropdown menu.
3. Click on Port Forwarding as shown on the below screenshot:

Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab



Then, enter the port forwarding rules as shown on the next screenshot:

Name	Protocol	Host IP	Host Port	Guest IP	Guest Port
Rule 1	TCP		2222		22
Rule 2	TCP		8080		80

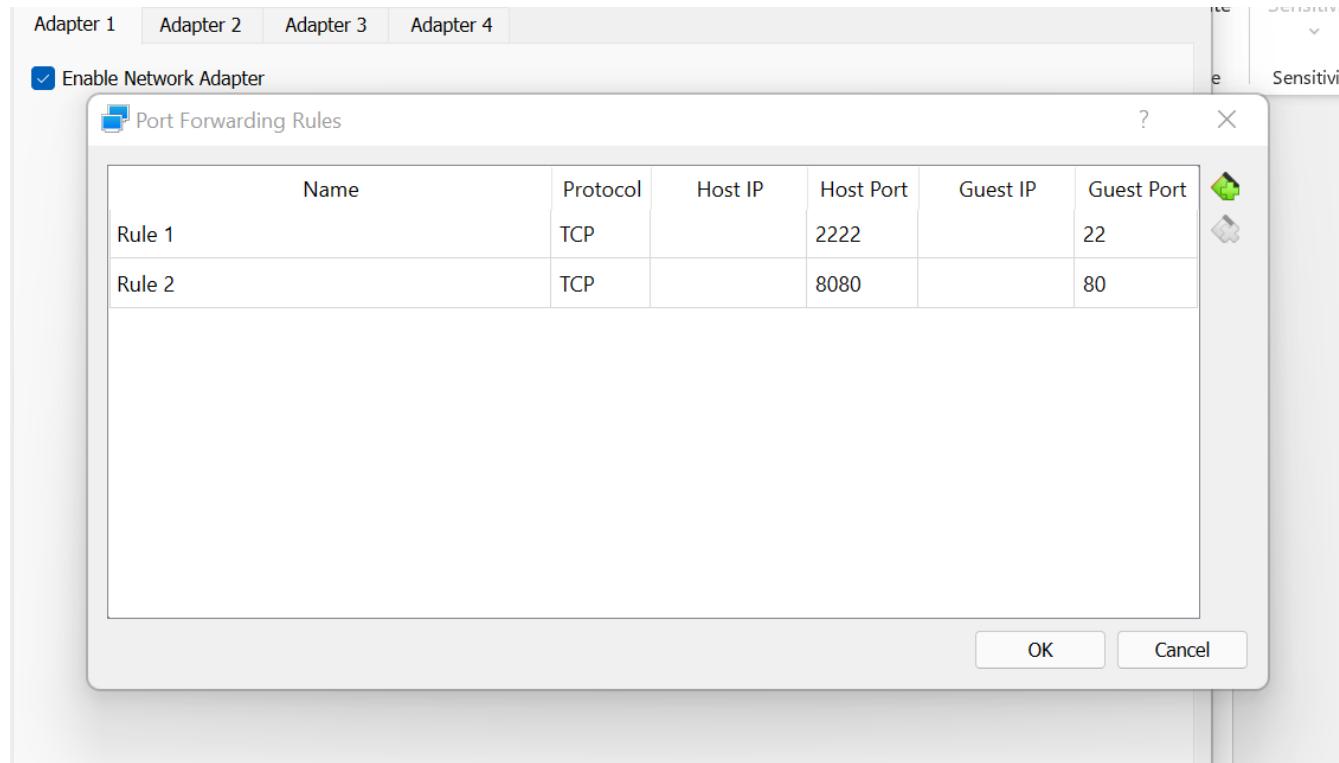
Q1: Attach a screenshot of your port forwarding rules. (It should look the same as above.)

Student Name:

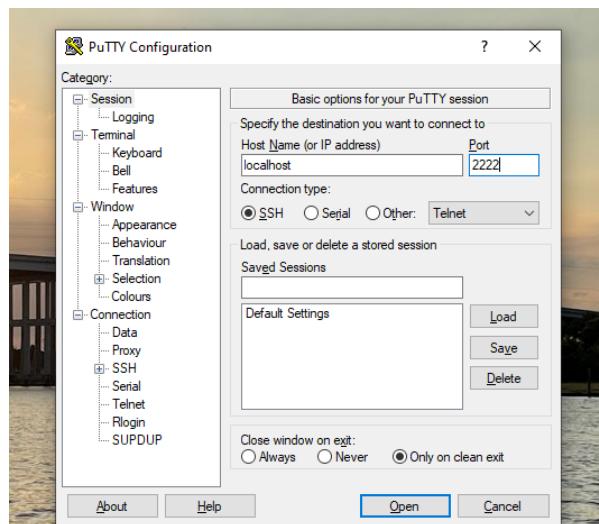
Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab



Now, let us confirm that the port forward works. If you use Windows open PuTTY on your host computer (an SSH/telnet client – install it, if you do not have it) and connect to localhost:2222.



If you use Linux/Mac OS, type “ssh sec-lab@localhost:2222” in the terminal.

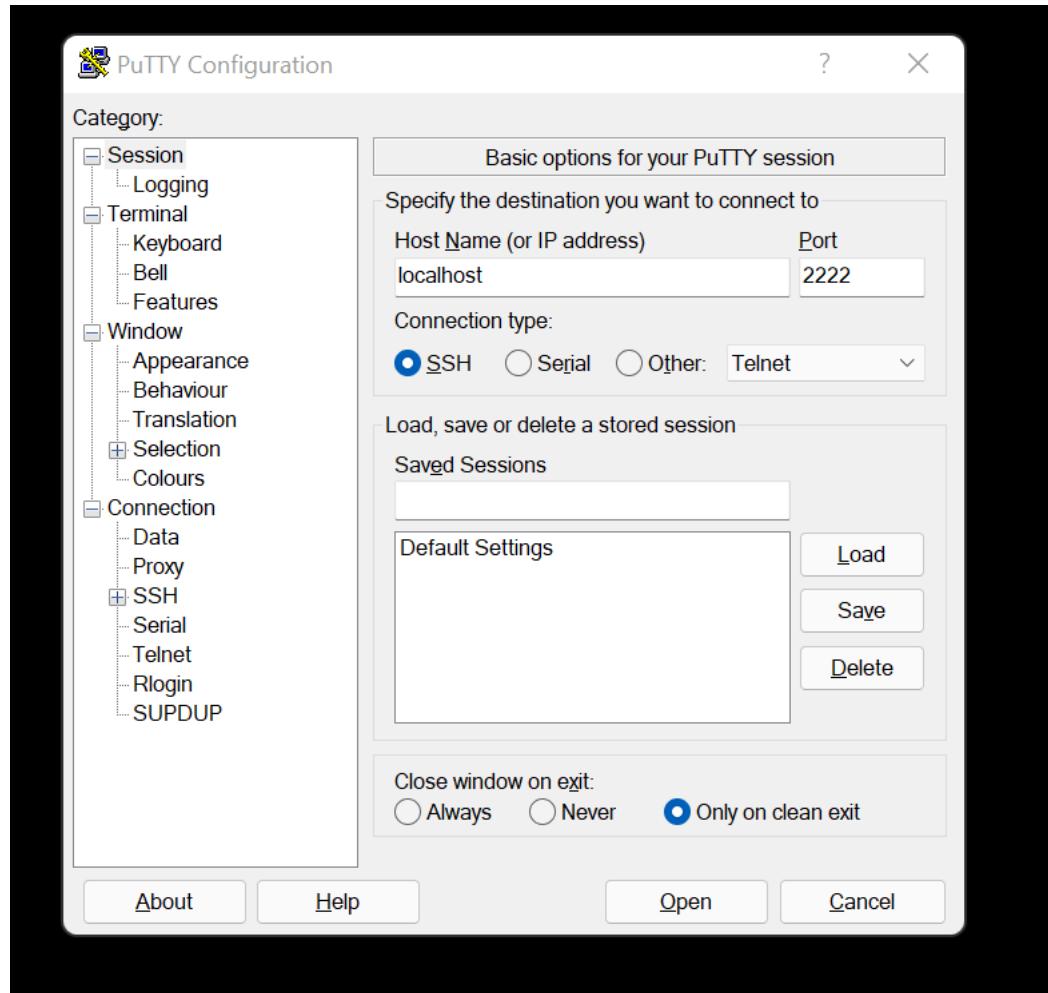
Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab

Q2: Attach a screenshot of the result.

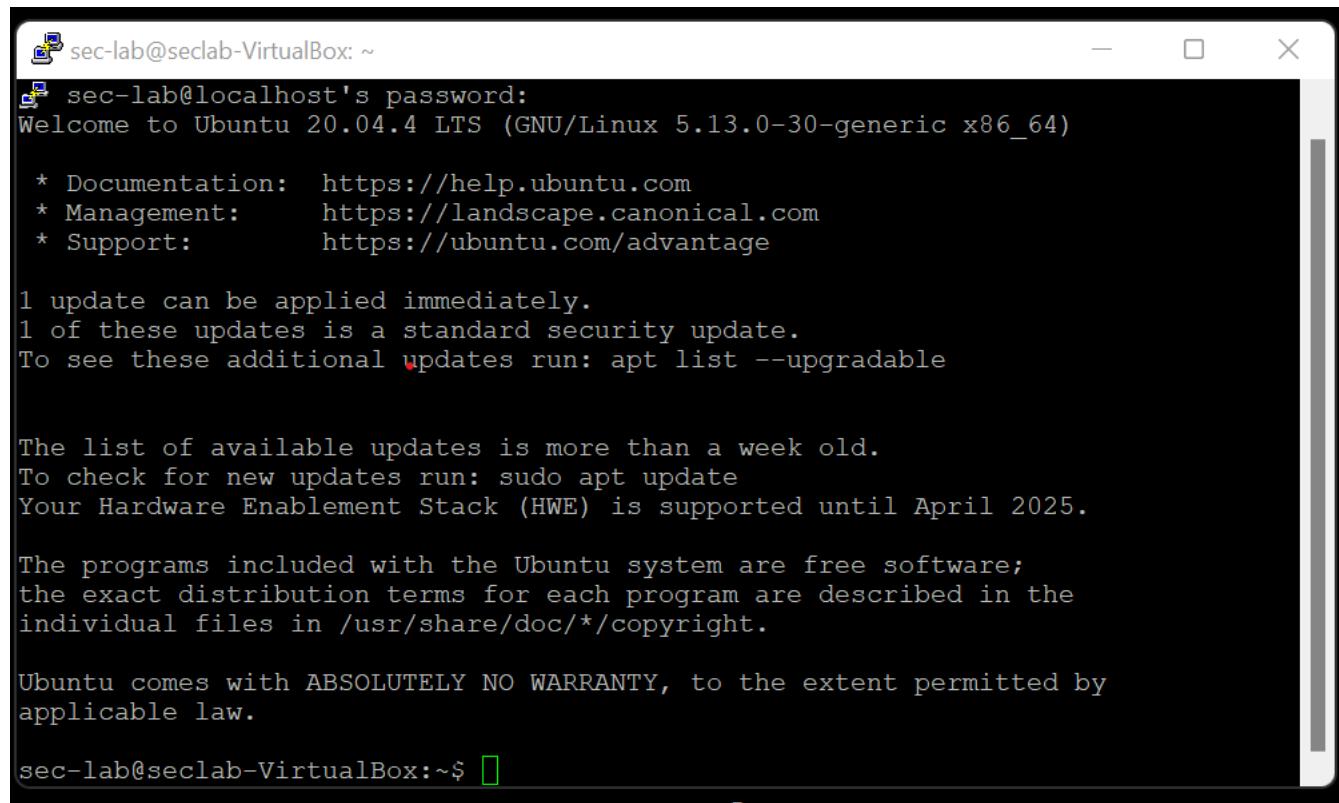


Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab



sec-lab@seclab-VirtualBox: ~

```
sec-lab@localhost's password:
Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.13.0-30-generic x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/advantage

1 update can be applied immediately.
1 of these updates is a standard security update.
To see these additional updates run: apt list --upgradable

The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Your Hardware Enablement Stack (HWE) is supported until April 2025.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

sec-lab@seclab-VirtualBox:~$
```

On your host computer, open the web browser and access “localhost:8080”.

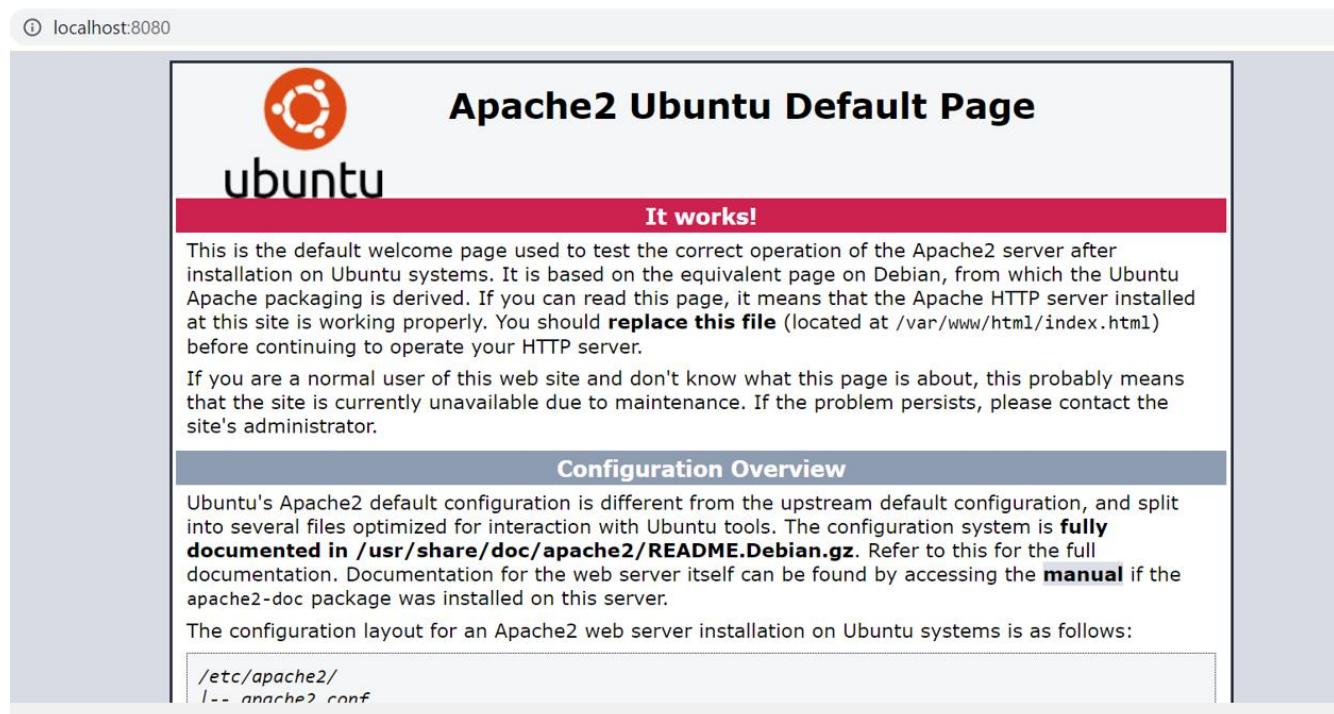
Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab

Q3: Attach a screenshot of the result.



Section 1: UFW - Uncomplicated Firewall

When UFW is activated, it uses a default set of rules (profile) that should be fine for the average home user. In short, they can be summarized as all 'incoming' is being denied, with some exceptions to make things easier for home users.

1. To turn UFW on with the default set of rules, type in the Linux terminal:
sudo ufw enable
2. To check the status of UFW:
sudo ufw status verbose

The output should resemble the following:

```
youruser@yourcomputer:~$ sudo ufw status verbose
```

[sudo] password for your user:

Status: active

Logging: on (low)

Default: deny (incoming), allow (outgoing)

Student Name:

Course: CSCE 5550

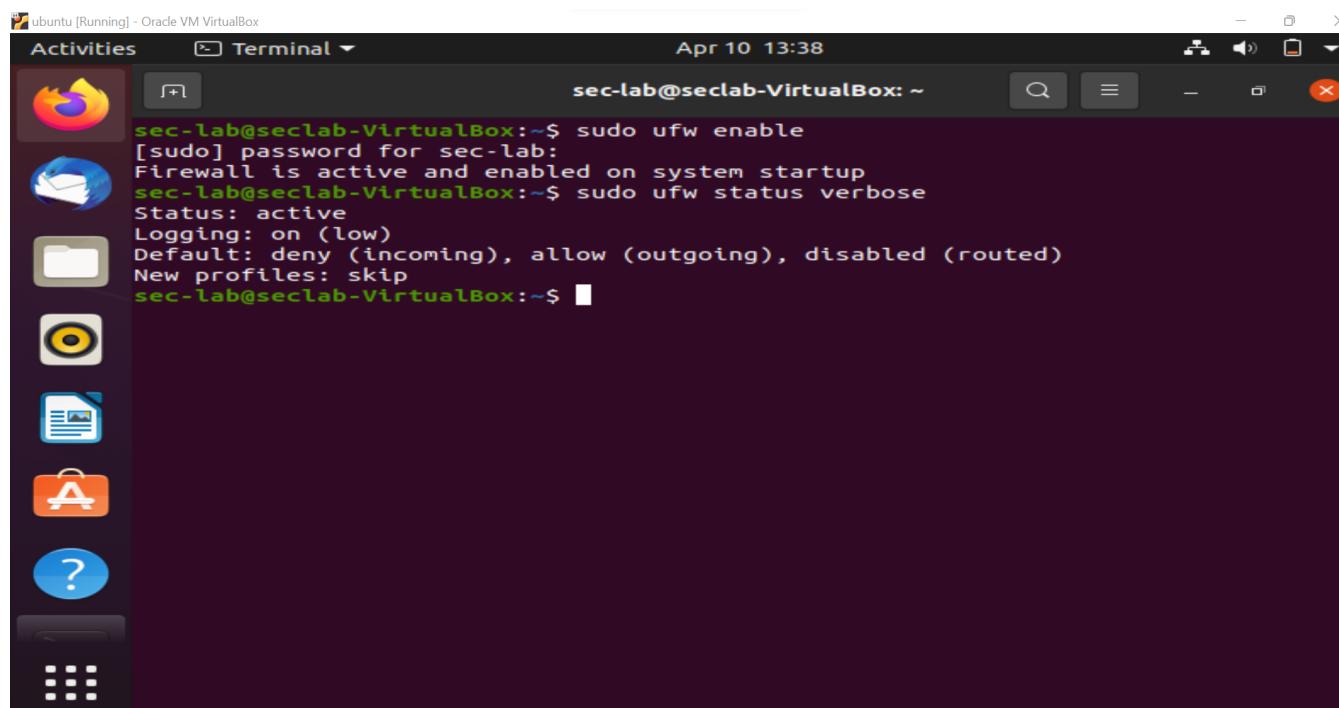
Semester: Fall 2021

Firewalls and IDS Lab

New profiles: skip

youruser@yourcomputer:~\$

Q4: Attach a screenshot of the terminal showing the status of UFW.



A screenshot of a Linux terminal window titled "Activities Terminal". The window shows the command line interface of a virtual machine named "sec-lab@seclab-VirtualBox". The terminal output is as follows:

```
sec-lab@seclab-VirtualBox:~$ sudo ufw enable
[sudo] password for sec-lab:
Firewall is active and enabled on system startup
sec-lab@seclab-VirtualBox:~$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip
sec-lab@seclab-VirtualBox:~$ █
```

3. Now, we will try to block incoming SSH packets by blocking the port using UFW.
In the Ubuntu VM, type:
`sudo ufw deny ssh`
4. From your host, try to establish an SSH connection again, as you did before Question 2.
5. Also try to connect on browser with “`http://localhost:8080`”.

Q5: Are you able to connect to your VM or get access to localhost website? Attach a screenshot and explain what happened.

Student Name:

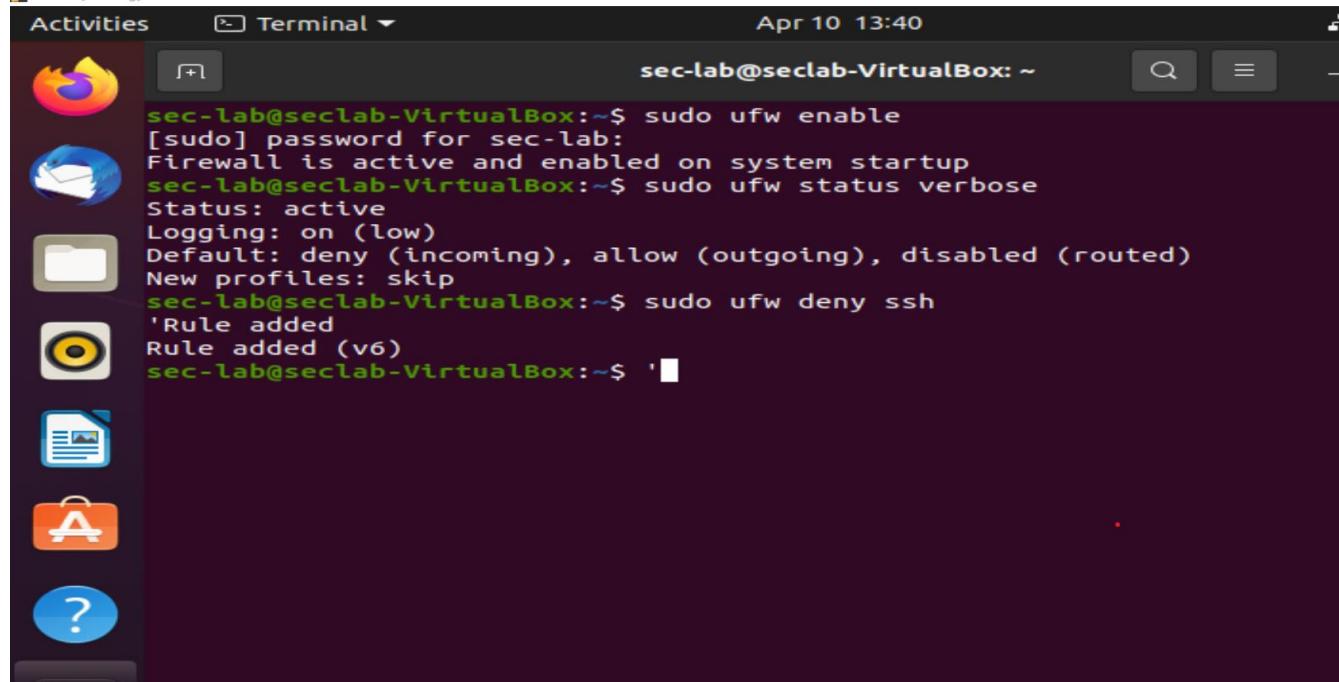
Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab

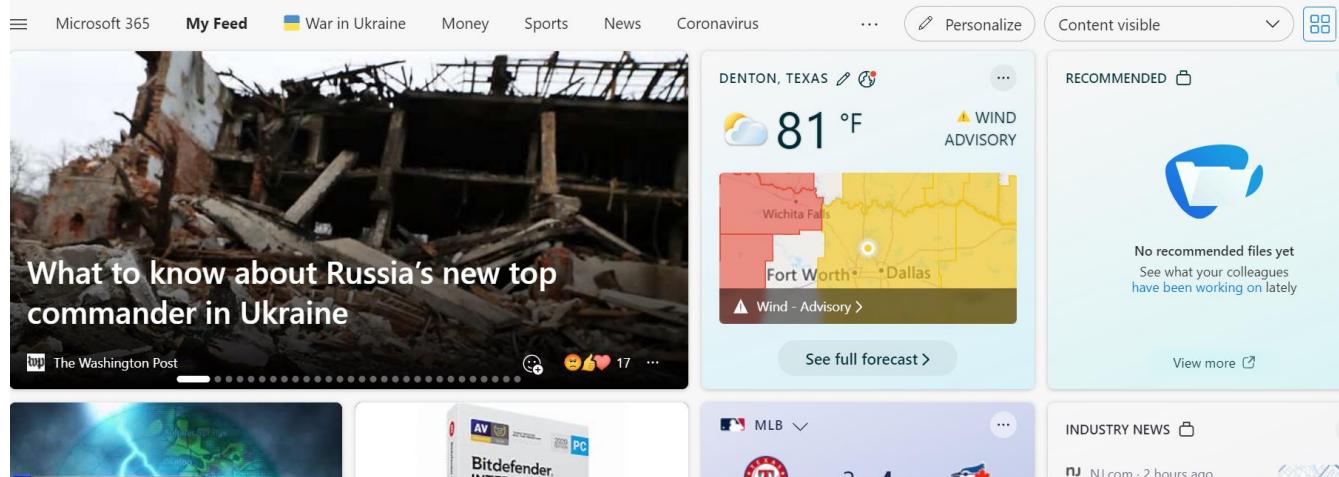
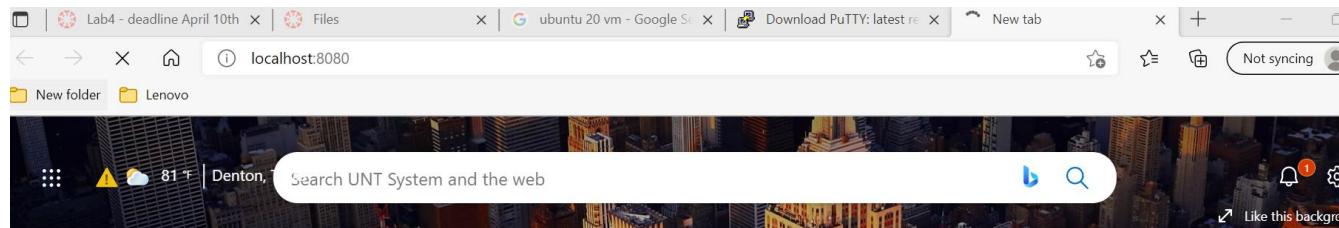
We have executed command “sudo ufw deny ssh” (denying the ssh connection). Hence, we cannot connect to localhost.

ubuntu [Running] - Oracle VM VirtualBox



A screenshot of a Linux desktop environment, likely Ubuntu, running in a virtual machine. The desktop has a dark theme with a dock on the left containing icons for various applications like a browser, file manager, and terminal. A terminal window is open at the top right, showing the command line session:

```
sec-lab@seclab-VirtualBox:~$ sudo ufw enable
[sudo] password for sec-lab:
Firewall is active and enabled on system startup
sec-lab@seclab-VirtualBox:~$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip
sec-lab@seclab-VirtualBox:~$ sudo ufw deny ssh
'Rule added
Rule added (v6)
sec-lab@seclab-VirtualBox:~$ '
```



A screenshot of a Microsoft 365 news feed. The top navigation bar includes links for "Microsoft 365", "My Feed", "War in Ukraine", "Money", "Sports", "News", "Coronavirus", "Personalize", and "Content visible". The main content area features a large image of a destroyed building, a weather forecast for Denton, Texas, and a news article from The Washington Post titled "What to know about Russia's new top commander in Ukraine". There are also sections for "RECOMMENDED" content and "INDUSTRY NEWS".

Student Name:

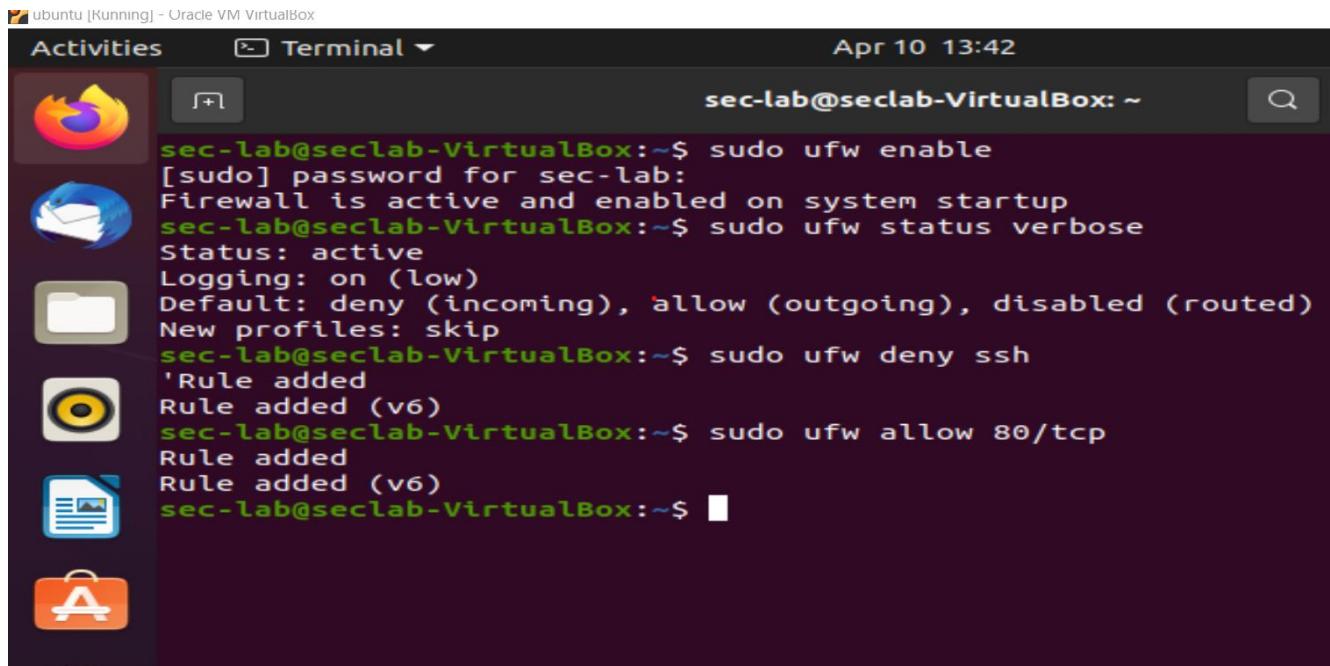
Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab

6. We are going to allow HTTP connections to the guest (Ubuntu VM acts now as a server). To allow incoming TCP packets on port 80 (http), type in the terminal of the Ubuntu VM:

```
sudo ufw allow 80/tcp
```



The screenshot shows a desktop environment with a terminal window open. The terminal window title is "Terminal" and the date and time are "Apr 10 13:42". The terminal content shows the following command history:

```
sec-lab@seclab-VirtualBox:~$ sudo ufw enable
[sudo] password for sec-lab:
Firewall is active and enabled on system startup
sec-lab@seclab-VirtualBox:~$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip
sec-lab@seclab-VirtualBox:~$ sudo ufw deny ssh
'Rule added
Rule added (v6)
sec-lab@seclab-VirtualBox:~$ sudo ufw allow 80/tcp
Rule added
Rule added (v6)
sec-lab@seclab-VirtualBox:~$ █
```

Note: We allow port 80 because it is the port of the HTTP server installed in the Ubuntu VM. The port 8080 is not used in the Ubuntu VM, it is only used for access from the host and it is redirected to port 80 of the VM.

From your host, try to access “localhost:8080” in the browser.

Q6: Attach a screenshot of the resulting browser window.

Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab

localhost:8080

Apache2 Ubuntu Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in `/usr/share/doc/apache2/README.Debian.gz`**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/  
|-- apache2.conf
```

Note: In the rest of the lab, we are going to deny outgoing connections from the Ubuntu VM.

7. Using UFW to block outgoing traffic to website “www.cnn.com”. The command syntax is as follows:

```
sudo ufw deny out from any to _ipaddress_
```

To write the above command, we need to know the IP address of the server www.cnn.com.

8. Type:

```
host -t a www.cnn.com
```

You may expect the output similar to the following:

```
www.cnn.com is an alias for turner-tls.map.fastly.net.
```

```
turner-tls.map.fastly.net has address 151.101.1.67
```

```
turner-tls.map.fastly.net has address 151.101.65.67
```

```
turner-tls.map.fastly.net has address 151.101.129.67
```

```
turner-tls.map.fastly.net has address 151.101.193.67
```

Student Name:

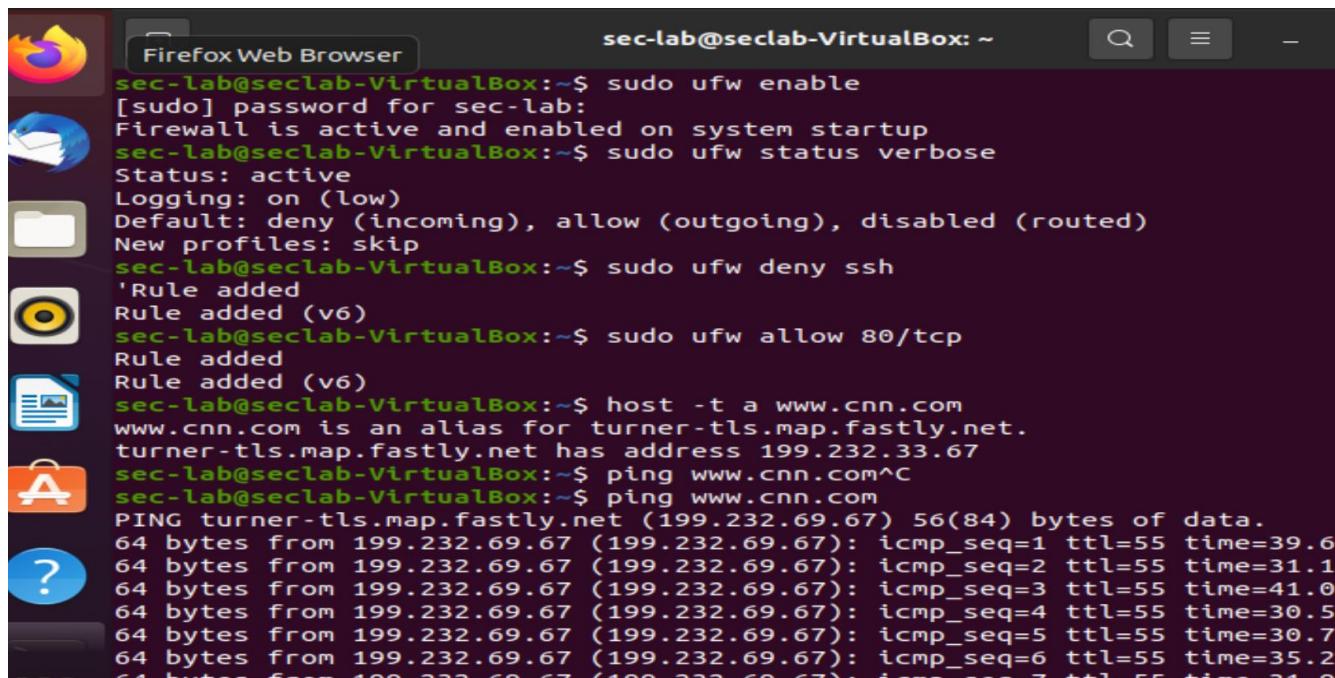
Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab

The main IP address of www.cnn.com is 151.101.1.67, but the server has alternative IP addresses 151.101.65.67, 151.101.129.67, and 151.101.193.67.

Q7: Attach a screenshot of the result.



A screenshot of a Linux terminal window titled "Firefox Web Browser". The terminal session shows the following commands and output:

```
sec-lab@seclab-VirtualBox:~$ sudo ufw enable
[sudo] password for sec-lab:
Firewall is active and enabled on system startup
sec-lab@seclab-VirtualBox:~$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip
sec-lab@seclab-VirtualBox:~$ sudo ufw deny ssh
'Rule added
Rule added (v6)
sec-lab@seclab-VirtualBox:~$ sudo ufw allow 80/tcp
Rule added
Rule added (v6)
sec-lab@seclab-VirtualBox:~$ host -t a www.cnn.com
www.cnn.com is an alias for turner-tls.map.fastly.net.
turner-tls.map.fastly.net has address 199.232.33.67
sec-lab@seclab-VirtualBox:~$ ping www.cnn.com^C
sec-lab@seclab-VirtualBox:~$ ping www.cnn.com
PING turner-tls.map.fastly.net (199.232.69.67) 56(84) bytes of data.
64 bytes from 199.232.69.67 (199.232.69.67): icmp_seq=1 ttl=55 time=39.6
64 bytes from 199.232.69.67 (199.232.69.67): icmp_seq=2 ttl=55 time=31.1
64 bytes from 199.232.69.67 (199.232.69.67): icmp_seq=3 ttl=55 time=41.0
64 bytes from 199.232.69.67 (199.232.69.67): icmp_seq=4 ttl=55 time=30.5
64 bytes from 199.232.69.67 (199.232.69.67): icmp_seq=5 ttl=55 time=30.7
64 bytes from 199.232.69.67 (199.232.69.67): icmp_seq=6 ttl=55 time=35.2
64 bytes from 199.232.69.67 (199.232.69.67): icmp_seq=7 ttl=55 time=21.9
```

Note: Alternatively, you can use the ping tool, but in this case you will only get the main IP address. Type “ping www.cnn.com” and press “Control + C”, the output will be similar to the following:

```
PING turner-tls.map.fastly.net (151.101.1.67) 56(84) bytes of data.
64 bytes from 151.101.49.67: icmp_req=1 ttl=52 time=98.8 ms
64 bytes from 151.101.49.67: icmp_req=2 ttl=52 time=28.8 ms
64 bytes from 151.101.49.67: icmp_req=3 ttl=52 time=79.2 ms
```

Test in the guest OS (Ubuntu) browser, that you are able to access the www.cnn.com website.
(Note: All tests in this lab are done in the browser of the Ubuntu VM.)

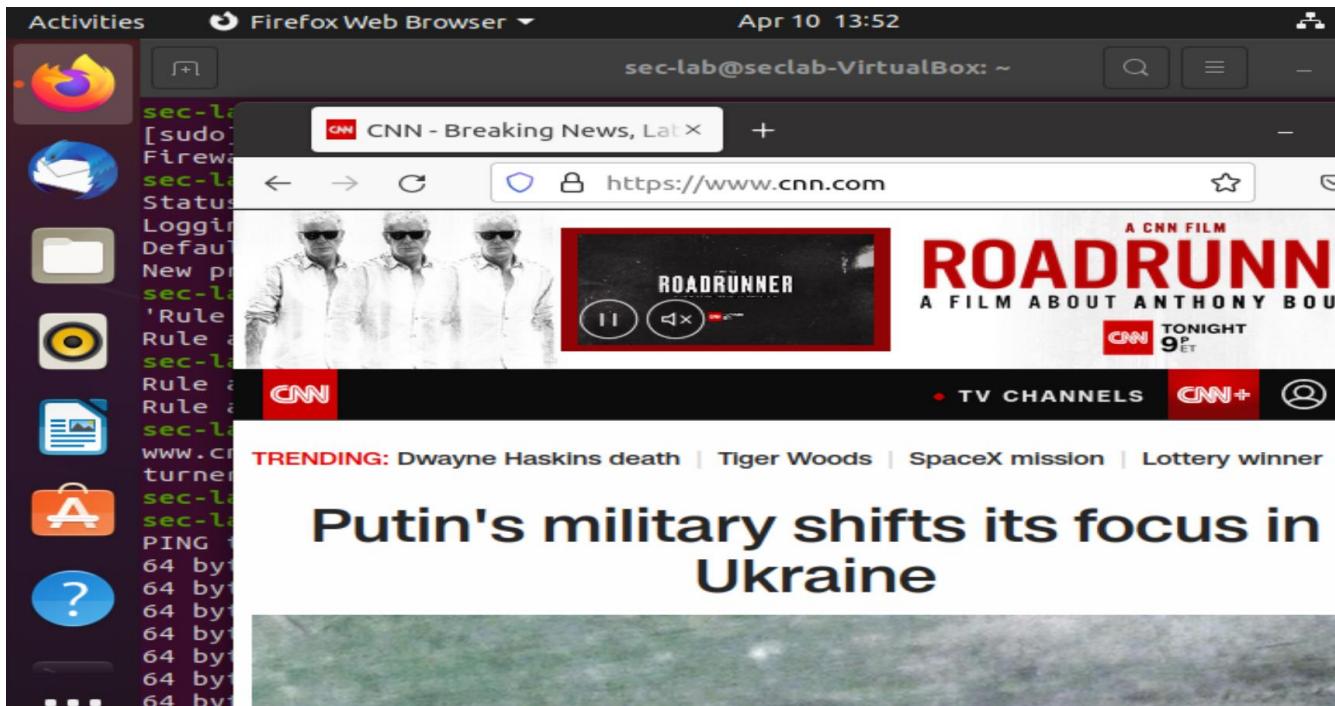
Q8: Attach a screenshot of the resulting browser window.

Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab



Now, we will run the following command:

```
sudo ufw deny out from any to 151.101.1.67
```

After that, you may not be able to access to the www.cnn.com webpage.

9. Try to access www.cnn.com in the browser.
10. If you still are able to connect, follow the steps mentioned in the note below.

Note: In order to completely deny access to the domain www.cnn.com, we need to cover all the IP addresses obtained using “host -t a www.cnn.com” along with alternative ip addresses mentioned earlier.

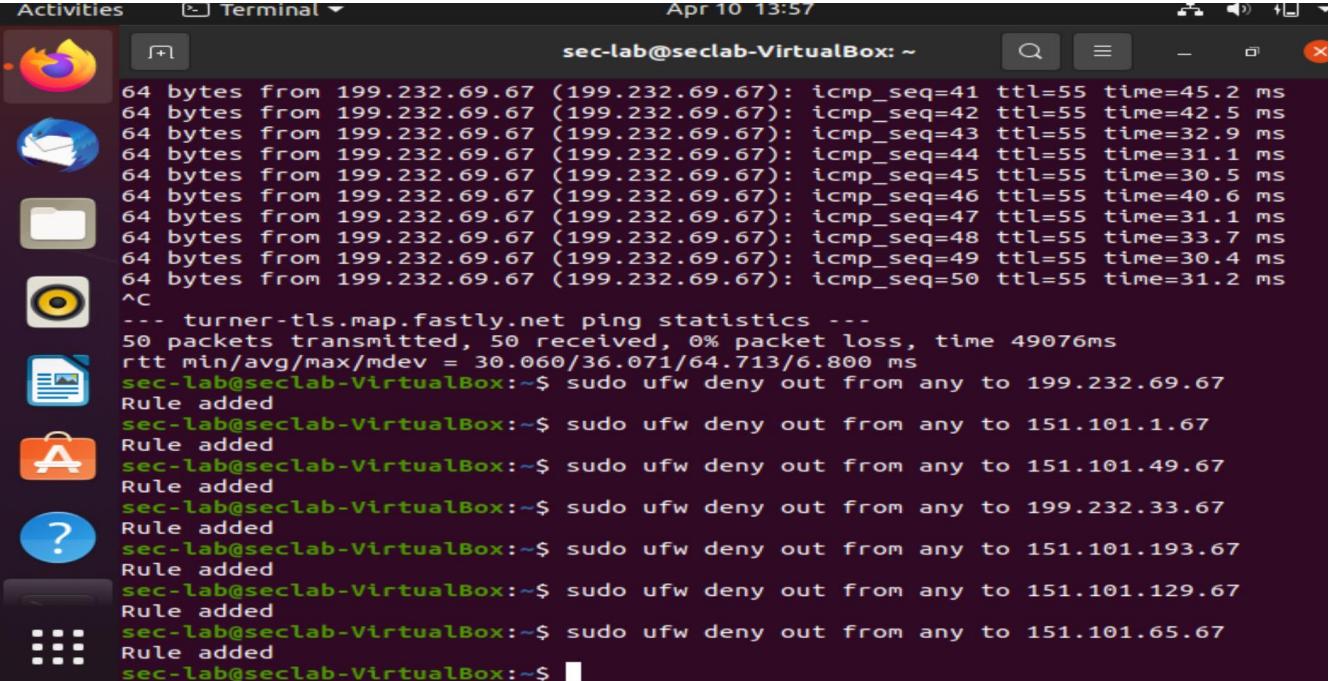
Q9: Attach a screenshot of the resulting browser window.

Student Name:

Course: CSCE 5550

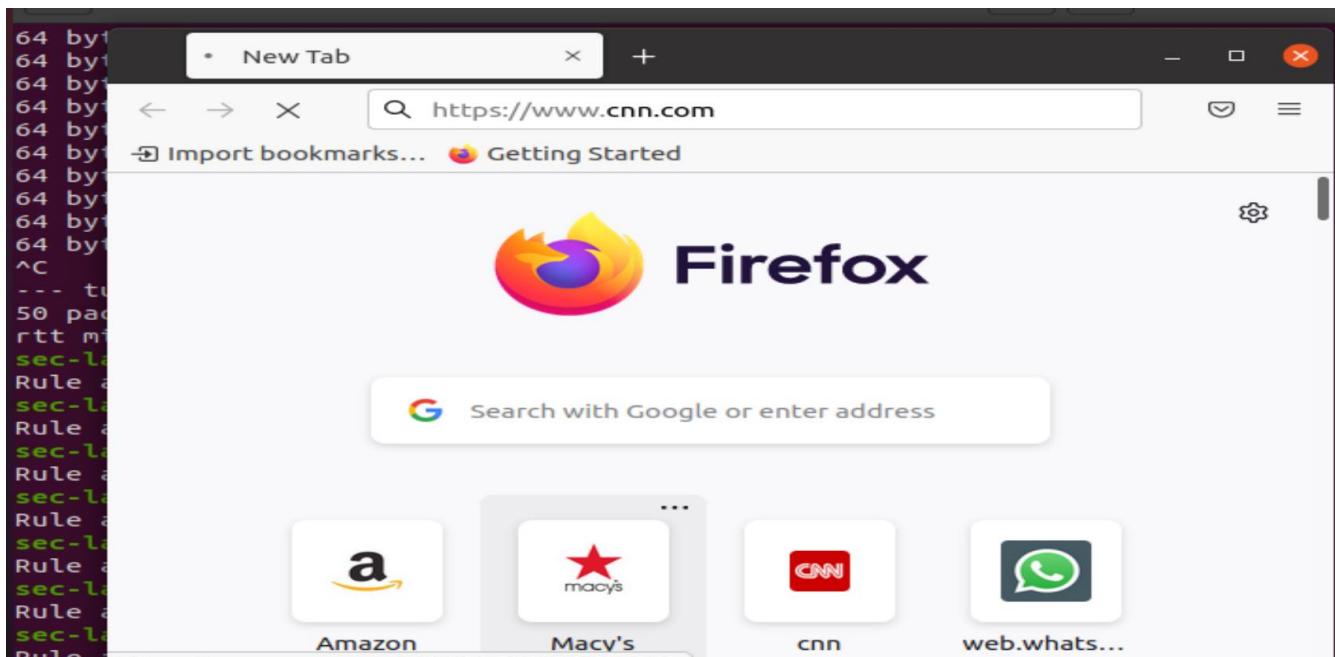
Semester: Fall 2021

Firewalls and IDS Lab



A screenshot of a Linux desktop environment. In the foreground, a terminal window titled "Activities Terminal" is open, showing a ping session to 199.232.69.67 and a series of "sudo ufw deny out" commands being run. The terminal window has a dark background with light-colored text. In the background, a file manager window is visible, showing a list of files and folders.

```
sec-lab@seclab-VirtualBox: ~
64 bytes from 199.232.69.67 (199.232.69.67): icmp_seq=41 ttl=55 time=45.2 ms
64 bytes from 199.232.69.67 (199.232.69.67): icmp_seq=42 ttl=55 time=42.5 ms
64 bytes from 199.232.69.67 (199.232.69.67): icmp_seq=43 ttl=55 time=32.9 ms
64 bytes from 199.232.69.67 (199.232.69.67): icmp_seq=44 ttl=55 time=31.1 ms
64 bytes from 199.232.69.67 (199.232.69.67): icmp_seq=45 ttl=55 time=30.5 ms
64 bytes from 199.232.69.67 (199.232.69.67): icmp_seq=46 ttl=55 time=40.6 ms
64 bytes from 199.232.69.67 (199.232.69.67): icmp_seq=47 ttl=55 time=31.1 ms
64 bytes from 199.232.69.67 (199.232.69.67): icmp_seq=48 ttl=55 time=33.7 ms
64 bytes from 199.232.69.67 (199.232.69.67): icmp_seq=49 ttl=55 time=30.4 ms
64 bytes from 199.232.69.67 (199.232.69.67): icmp_seq=50 ttl=55 time=31.2 ms
^C
--- turner-tls.map.fastly.net ping statistics ---
50 packets transmitted, 50 received, 0% packet loss, time 49076ms
rtt min/avg/max/mdev = 30.060/36.071/64.713/6.800 ms
sec-lab@seclab-VirtualBox:~$ sudo ufw deny out from any to 199.232.69.67
Rule added
sec-lab@seclab-VirtualBox:~$ sudo ufw deny out from any to 151.101.1.67
Rule added
sec-lab@seclab-VirtualBox:~$ sudo ufw deny out from any to 151.101.49.67
Rule added
sec-lab@seclab-VirtualBox:~$ sudo ufw deny out from any to 199.232.33.67
Rule added
sec-lab@seclab-VirtualBox:~$ sudo ufw deny out from any to 151.101.193.67
Rule added
sec-lab@seclab-VirtualBox:~$ sudo ufw deny out from any to 151.101.129.67
Rule added
sec-lab@seclab-VirtualBox:~$ sudo ufw deny out from any to 151.101.65.67
Rule added
sec-lab@seclab-VirtualBox:~$
```



11. Check the UFW status again:

```
sudo ufw status verbose
```

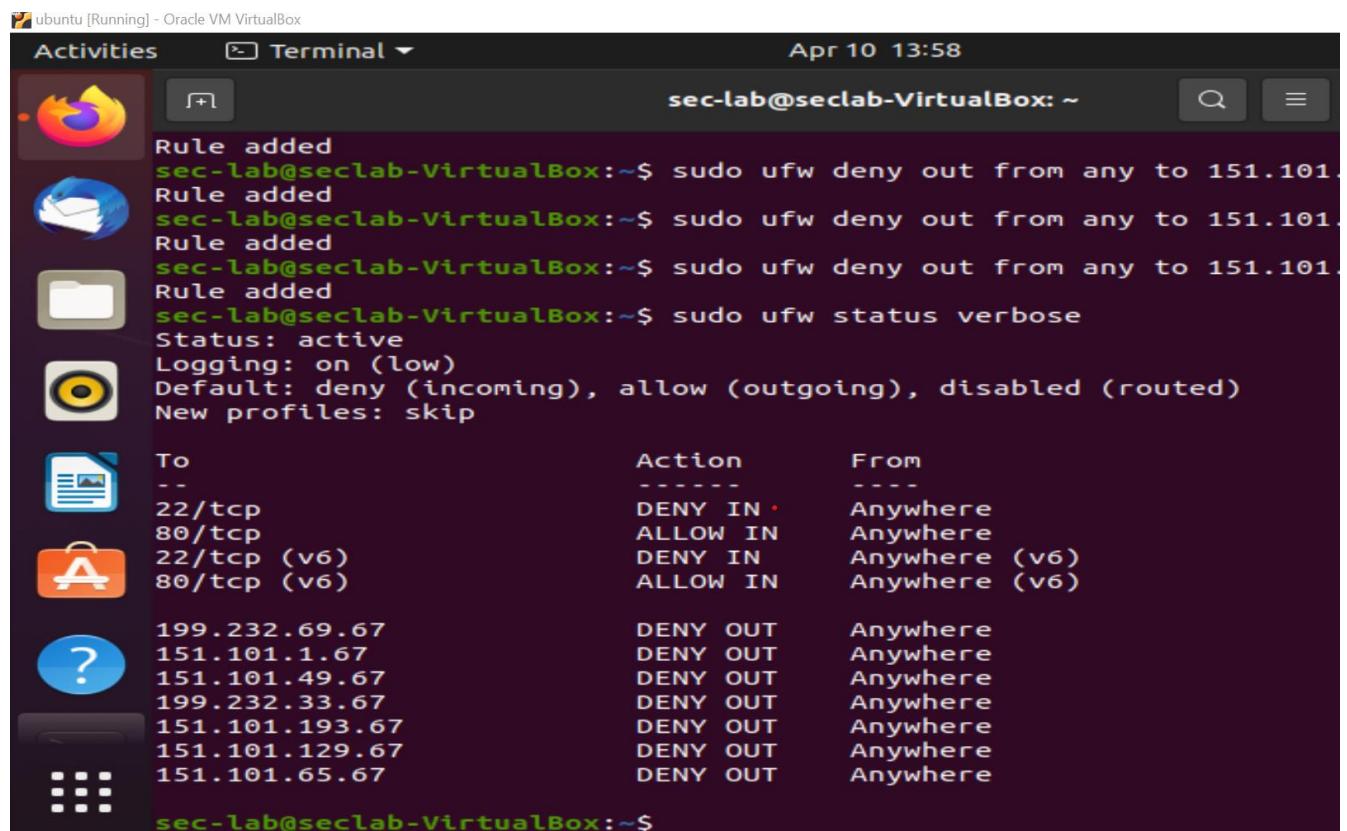
Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab

Q10: Attach a screenshot of the result.



The screenshot shows a terminal window titled "Activities" with a "Terminal" tab selected. The date and time "Apr 10 13:58" are at the top right. The command "sec-lab@seclab-VirtualBox: ~" is at the prompt. The terminal output is as follows:

```
Rule added
sec-lab@seclab-VirtualBox:~$ sudo ufw deny out from any to 151.101.
Rule added
sec-lab@seclab-VirtualBox:~$ sudo ufw deny out from any to 151.101.
Rule added
sec-lab@seclab-VirtualBox:~$ sudo ufw deny out from any to 151.101.
Rule added
sec-lab@seclab-VirtualBox:~$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To                         Action      From
--                         ----       ---
22/tcp                      DENY IN   Anywhere
80/tcp                      ALLOW IN  Anywhere
22/tcp (v6)                 DENY IN   Anywhere (v6)
80/tcp (v6)                 ALLOW IN  Anywhere (v6)

199.232.69.67               DENY OUT   Anywhere
151.101.1.67                DENY OUT   Anywhere
151.101.49.67               DENY OUT   Anywhere
199.232.33.67               DENY OUT   Anywhere
151.101.193.67              DENY OUT   Anywhere
151.101.129.67              DENY OUT   Anywhere
151.101.65.67               DENY OUT   Anywhere

sec-lab@seclab-VirtualBox:~$
```

12. Delete the last rule using the following command:

```
sudo ufw delete deny out from any to 151.101.1.67
```

Note: If you have added more than one rule for restricting access to www.cnn.com, then delete all of the rules related to accessing this website.

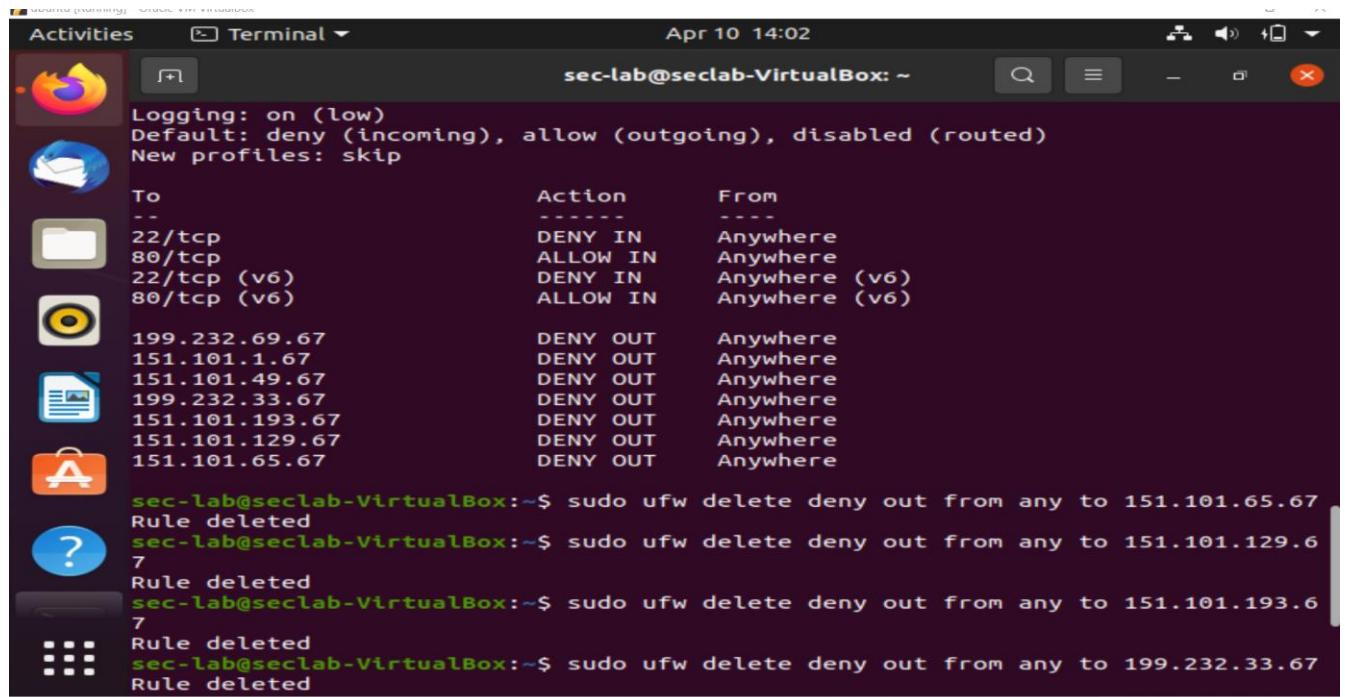
Confirm that the access to the www.cnn.com page has been restored.

Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab

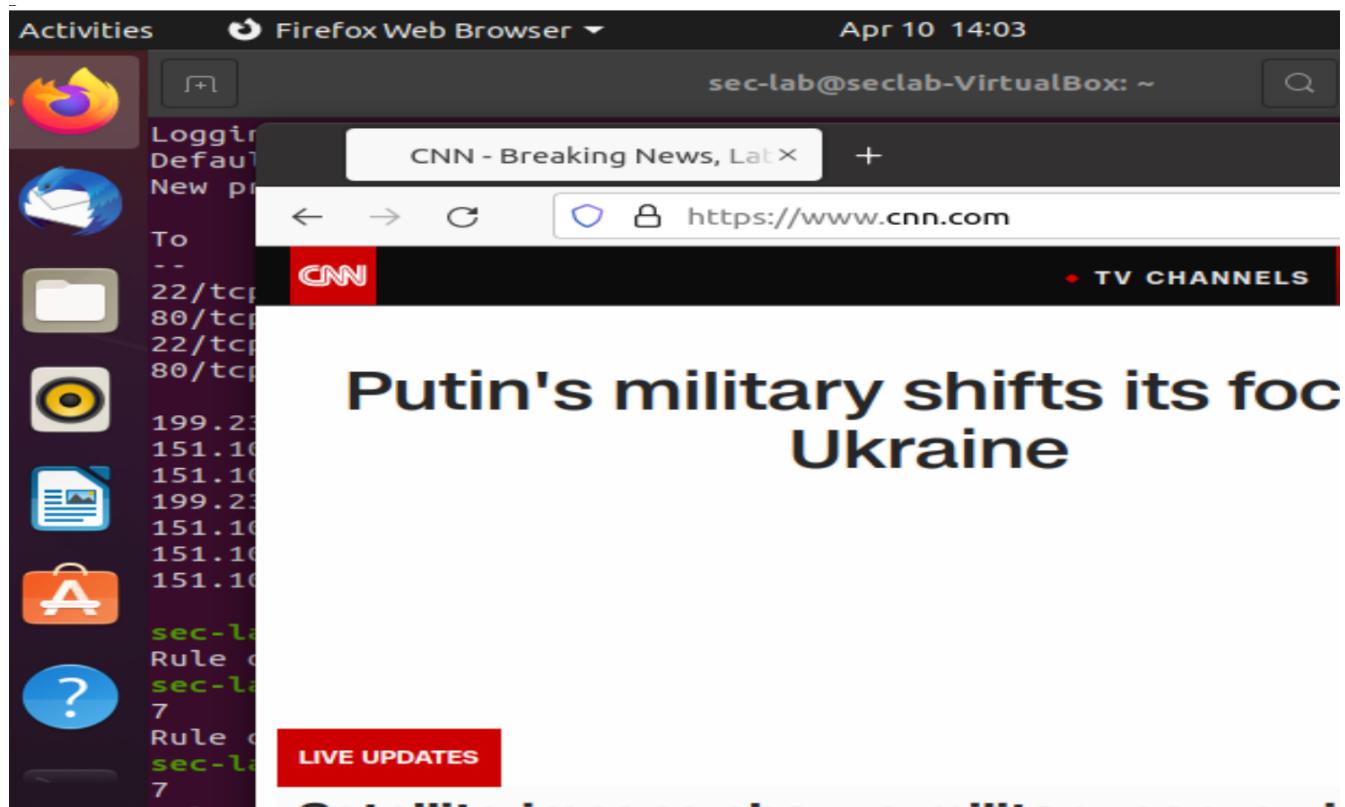


```
Activities Terminal ▾ sec-lab@seclab-VirtualBox: ~ Apr 10 14:02
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To Action From
-- -----
22/tcp DENY IN Anywhere
80/tcp ALLOW IN Anywhere
22/tcp (v6) DENY IN Anywhere (v6)
80/tcp (v6) ALLOW IN Anywhere (v6)

199.232.69.67 DENY OUT Anywhere
151.101.1.67 DENY OUT Anywhere
151.101.49.67 DENY OUT Anywhere
199.232.33.67 DENY OUT Anywhere
151.101.193.67 DENY OUT Anywhere
151.101.129.67 DENY OUT Anywhere
151.101.65.67 DENY OUT Anywhere

sec-lab@seclab-VirtualBox:~$ sudo ufw delete deny out from any to 151.101.65.67
Rule deleted
sec-lab@seclab-VirtualBox:~$ sudo ufw delete deny out from any to 151.101.129.67
7
Rule deleted
sec-lab@seclab-VirtualBox:~$ sudo ufw delete deny out from any to 151.101.193.67
7
Rule deleted
sec-lab@seclab-VirtualBox:~$ sudo ufw delete deny out from any to 199.232.33.67
Rule deleted
```



Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab

13. Now, we will block the access to the cse01.cse.unt.edu machine on the port 22 (SSH service).

The command for adding this rule can be typed in the Ubuntu VM terminal as follows (you need to fill in the IP address and port number):

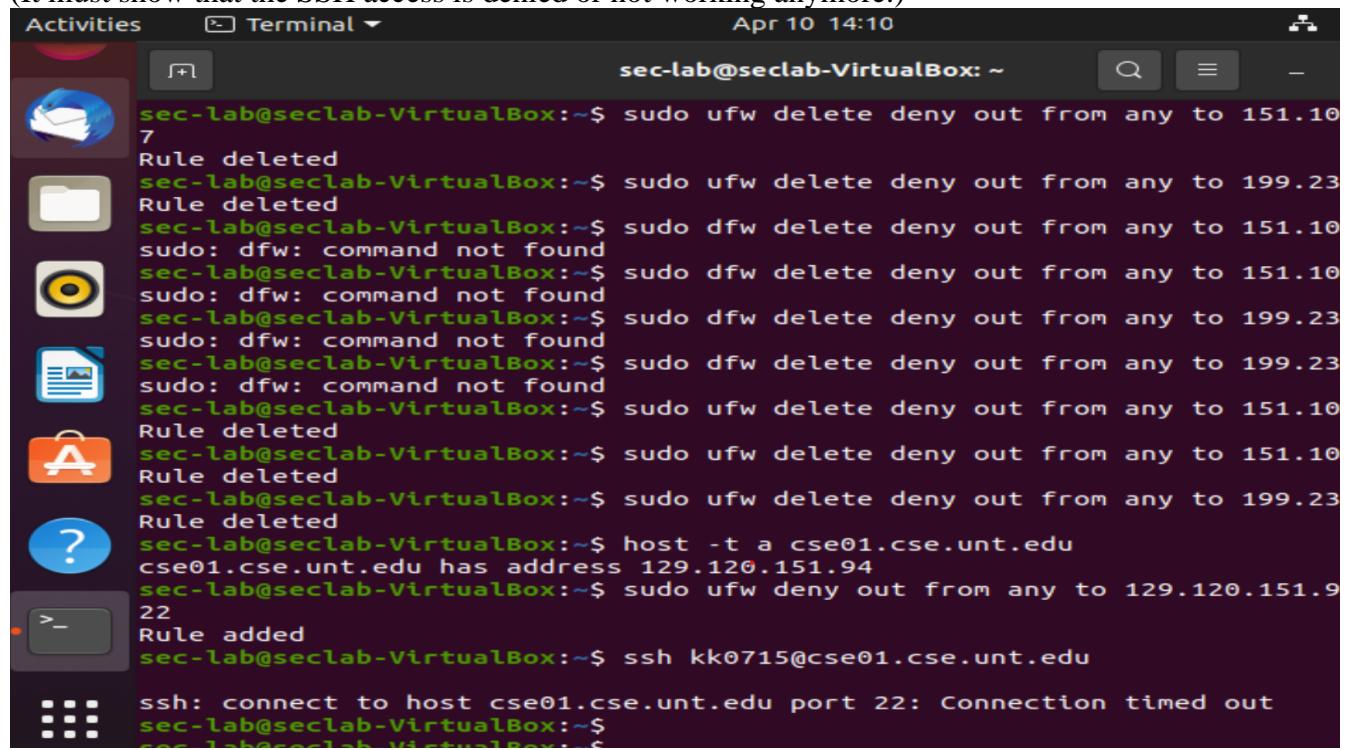
```
sudo ufw deny out from any to _ipaddress_ port _port-number_
```

To test whether the rule is working, in a terminal, try to establish a connection to cse01.cse.unt.edu server at port 22 (default SSH port). Remember that you need to use your EUID as a username for this connection.

The corresponding command is: “ssh euid@cse01.cse.unt.edu”.

Q11: Attach a screenshot of the terminal showing that you cannot create a SSH connection.

(It must show that the SSH access is denied or not working anymore.)



The screenshot shows a terminal window with the following session:

```
sec-lab@seclab-VirtualBox:~$ sudo ufw delete deny out from any to 151.10
7
Rule deleted
sec-lab@seclab-VirtualBox:~$ sudo ufw delete deny out from any to 199.23
Rule deleted
sec-lab@seclab-VirtualBox:~$ sudo dfw delete deny out from any to 151.10
sudo: dfw: command not found
sec-lab@seclab-VirtualBox:~$ sudo dfw delete deny out from any to 151.10
sudo: dfw: command not found
sec-lab@seclab-VirtualBox:~$ sudo dfw delete deny out from any to 199.23
sudo: dfw: command not found
sec-lab@seclab-VirtualBox:~$ sudo dfw delete deny out from any to 199.23
sudo: dfw: command not found
sec-lab@seclab-VirtualBox:~$ sudo ufw delete deny out from any to 151.10
Rule deleted
sec-lab@seclab-VirtualBox:~$ sudo ufw delete deny out from any to 151.10
Rule deleted
sec-lab@seclab-VirtualBox:~$ sudo ufw delete deny out from any to 199.23
Rule deleted
sec-lab@seclab-VirtualBox:~$ host -t a cse01.cse.unt.edu
cse01.cse.unt.edu has address 129.120.151.94
sec-lab@seclab-VirtualBox:~$ sudo ufw deny out from any to 129.120.151.9
22
Rule added
sec-lab@seclab-VirtualBox:~$ ssh kk0715@cse01.cse.unt.edu
ssh: connect to host cse01.cse.unt.edu port 22: Connection timed out
sec-lab@seclab-VirtualBox:~$
```

14. Delete the above rule by typing:

```
sudo ufw delete deny out from any to _ipaddress_ _sshport_
```

15. Now, try to establish an SSH connection to cse01.cse.unt.edu again.

Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab

Q12: Were you able to establish a connection? Attach a screenshot of the result.

Yes, we were able to establish a connection correctly.

Activities Terminal Apr 10 14:14
sec-lab@seclab-VirtualBox:~\$ sudo ufw delete deny out from any to 129.120.151.9
4 port 22
Rule deleted
sec-lab@seclab-VirtualBox:~\$ ssh kk0715@cse01.cse.unt.edu
The authenticity of host 'cse01.cse.unt.edu (129.120.151.94)' can't be established.
RSA key fingerprint is SHA256:zolu6BHYudz7Ca/MJlLaYE2KLJhn8TubDE4EGzpNExo.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'cse01.cse.unt.edu,129.120.151.94' (RSA) to the list
of known hosts.
WARNING
This system is the property of the University of North Texas
and your use of this resource constitutes an explicit binding
agreement to abide by relevant federal and state laws and UNT
policies (see UNT Policies 3.10, 3.6, and 3.11). Unauthorized
use of this system is prohibited. Violations can result in
severe penalties and possible criminal prosecution. There is
no reasonable expectation of privacy and you consent to
monitoring, review and disclosure of information by using
this system.
kk0715@cse01.cse.unt.edu's password:
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 4.15.0-167-generic i686)
The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the

Activities Terminal Apr 10 14:15
kk0715@cse01:~\$
kk0715@cse01.cse.unt.edu's password:
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 4.15.0-167-generic i686)
The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

UNT Department of Computer Science and Engineering

Administration of this system is provided by UNT College of Engineering.
For assistance or to resolve any server problems send an email
to cengsupport@unt.edu, call (940) 369-7250, or swing by room
B189 at Discovery Park.

Although we try to maintain regular backups of student home
directories, it is the responsibility of the student to ensure
he/she has taken appropriate precautions to safeguard his/her data.
Anything you place into your home directory on the CSP machines
should also be kept locally on your own computer to ensure that
you do not lose any data.

kk0715@cse01:~\$

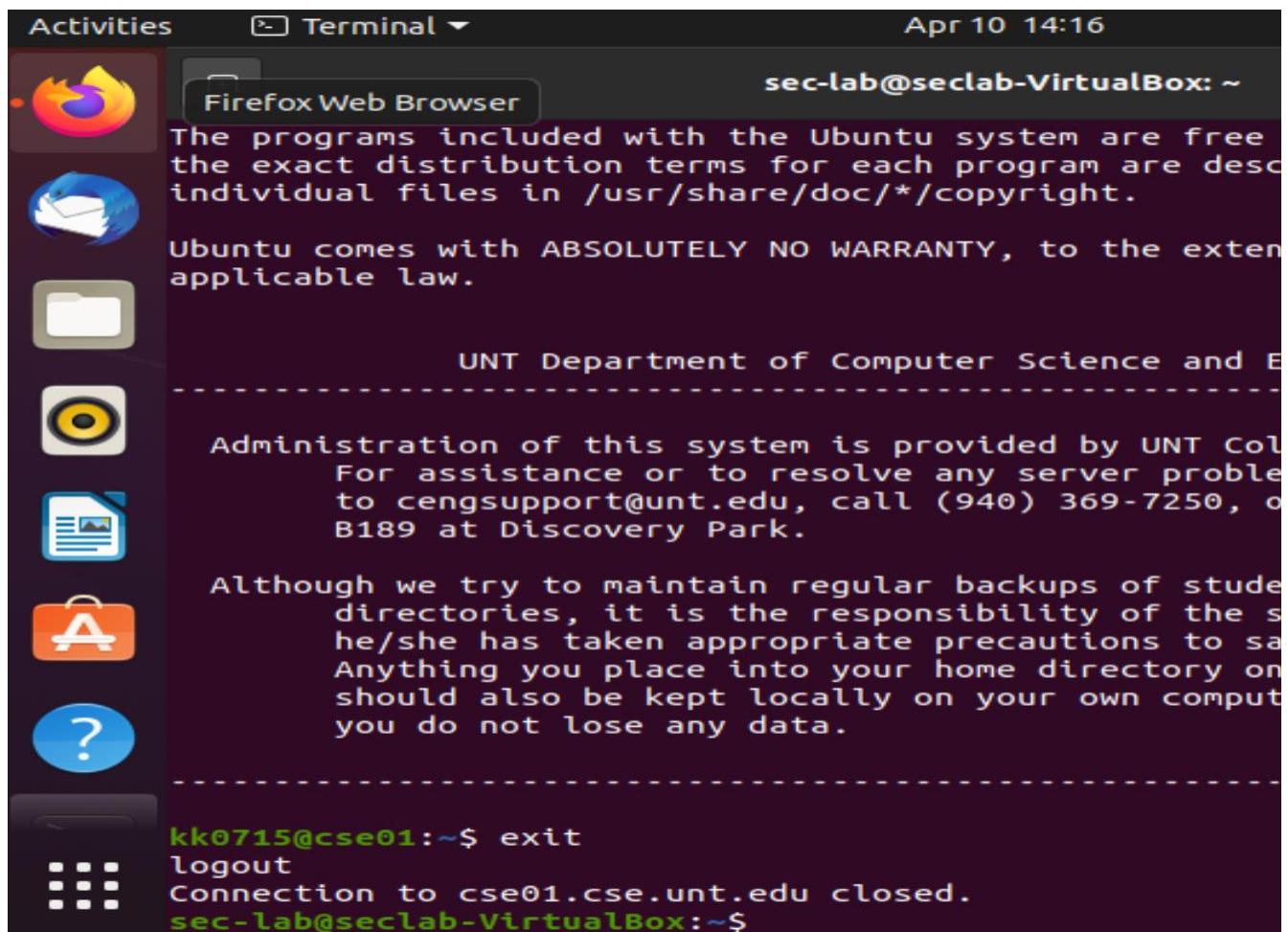
Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab

16. Terminate your session on CSE machine using the command “exit”, in case if you are logged in.



The screenshot shows a Linux desktop environment with a terminal window open. The terminal window title is "Terminal" and the date and time are "Apr 10 14:16". The terminal window displays the following text:

```
sec-lab@seclab-VirtualBox: ~
The programs included with the Ubuntu system are free
the exact distribution terms for each program are desc
individual files in /usr/share/doc/*/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the exten
applicable law.

-----[REDACTED]-----
Administration of this system is provided by UNT Col
For assistance or to resolve any server proble
to cengsupport@unt.edu, call (940) 369-7250, o
B189 at Discovery Park.

-----[REDACTED]-----
Although we try to maintain regular backups of stude
directories, it is the responsibility of the s
he/she has taken appropriate precautions to sa
Anything you place into your home directory on
should also be kept locally on your own comput
you do not lose any data.

-----[REDACTED]-----
kk0715@cse01:~$ exit
logout
Connection to cse01.cse.unt.edu closed.
sec-lab@seclab-VirtualBox:~$
```

17. Reset UFW on your Ubuntu20 VM and disable it:

```
sudo ufw reset
```

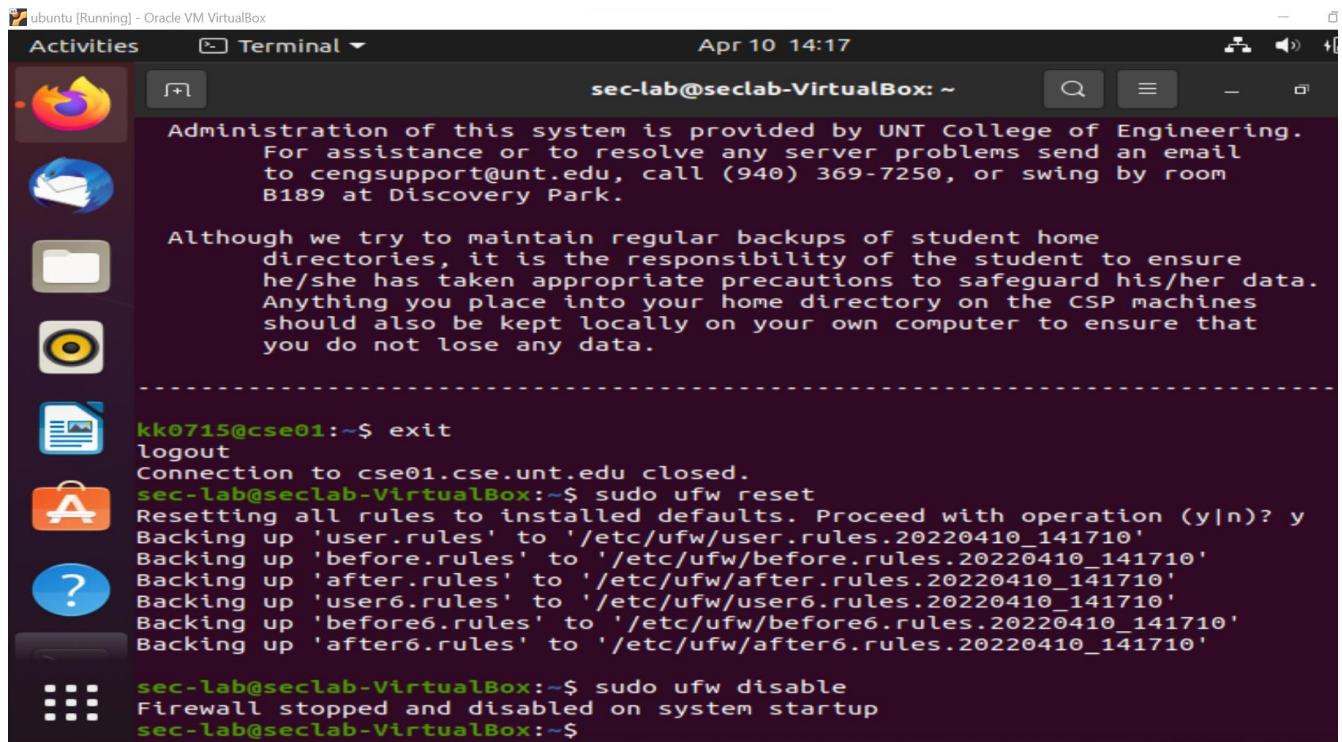
```
sudo ufw disable
```

Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab



The screenshot shows a terminal window titled "Terminal" with the command "sec-lab@seclab-VirtualBox: ~". The terminal displays several messages:

- A message about system administration by UNT College of Engineering.
- A note about backup responsibilities for student home directories.
- Logout information.
- Execution of "sudo ufw reset" followed by a confirmation prompt "y".
- Backing up of various UFW rule files.
- Execution of "sudo ufw disable" followed by a confirmation message.

```
Administration of this system is provided by UNT College of Engineering.  
For assistance or to resolve any server problems send an email  
to cengsupport@unt.edu, call (940) 369-7250, or swing by room  
B189 at Discovery Park.  
  
Although we try to maintain regular backups of student home  
directories, it is the responsibility of the student to ensure  
he/she has taken appropriate precautions to safeguard his/her data.  
Anything you place into your home directory on the CSP machines  
should also be kept locally on your own computer to ensure that  
you do not lose any data.  
  
kk0715@cse01:~$ exit  
logout  
Connection to cse01.cse.unt.edu closed.  
sec-lab@seclab-VirtualBox:~$ sudo ufw reset  
Resetting all rules to installed defaults. Proceed with operation (y|n)? y  
Backing up 'user.rules' to '/etc/ufw/user.rules.20220410_141710'  
Backing up 'before.rules' to '/etc/ufw(before.rules.20220410_141710'  
Backing up 'after.rules' to '/etc/ufw(after.rules.20220410_141710'  
Backing up 'user6.rules' to '/etc/ufw/user6.rules.20220410_141710'  
Backing up 'before6.rules' to '/etc/ufw(before6.rules.20220410_141710'  
Backing up 'after6.rules' to '/etc/ufw(after6.rules.20220410_141710'  
  
sec-lab@seclab-VirtualBox:~$ sudo ufw disable  
Firewall stopped and disabled on system startup  
sec-lab@seclab-VirtualBox:~$
```

18. Power off the Ubuntu VM.

Section 2: Denial of Service (DoS) Attack – TCP SYN Flood

In this section, we will run the TCP SYN flooding attack against the Ubuntu VM (target) from the Kali VM (attacker). We will use the hping3 utility (www.hping.org), which a TCP/IP packet assembler/analyzer – it is commonly used for network testing and port scanning.

Note: The Kali 20 VM will be the same as in the previous labs.

First, let us configure the VMs so that they can communicate with each other. Note that the VMs must be powered off during this configuration process.

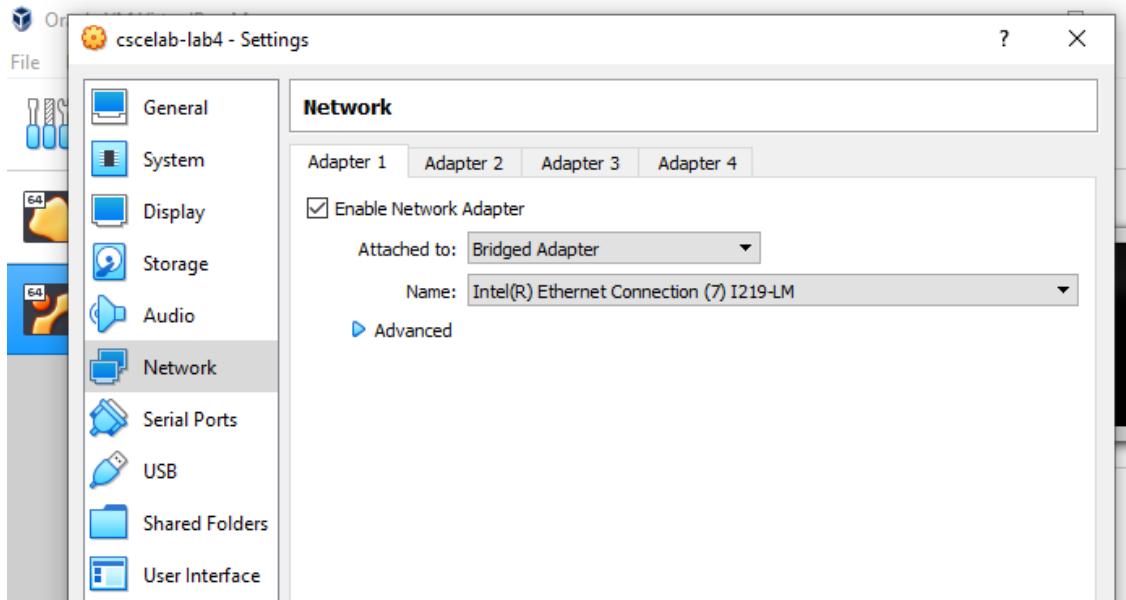
1. Use the following Bridged Adapter settings for the Ubuntu VM:

Student Name:

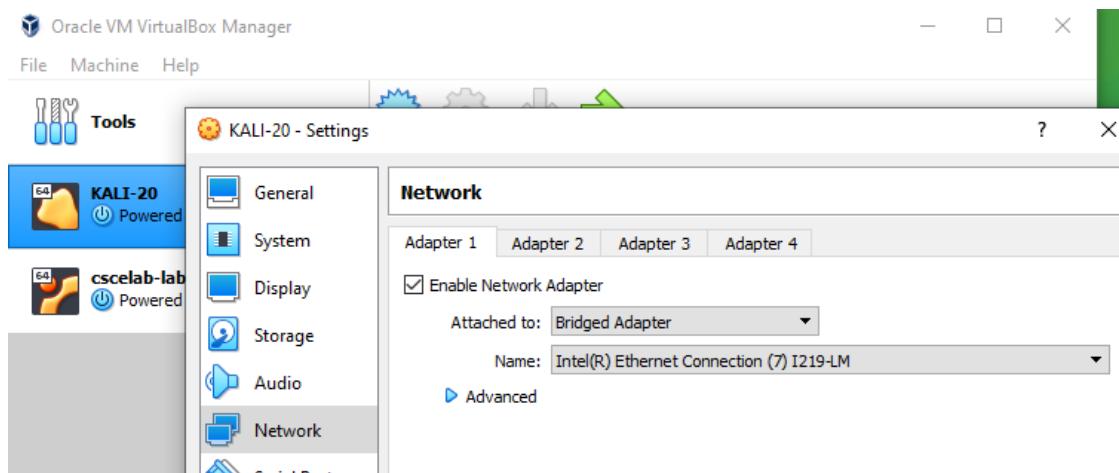
Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab



2. We recommend to “refresh” your Kali VM, i.e., to import it again from the respective OVA file. (Note that your previous work on that will be lost.)
3. Apply the following network adapter settings for the Kali VM:



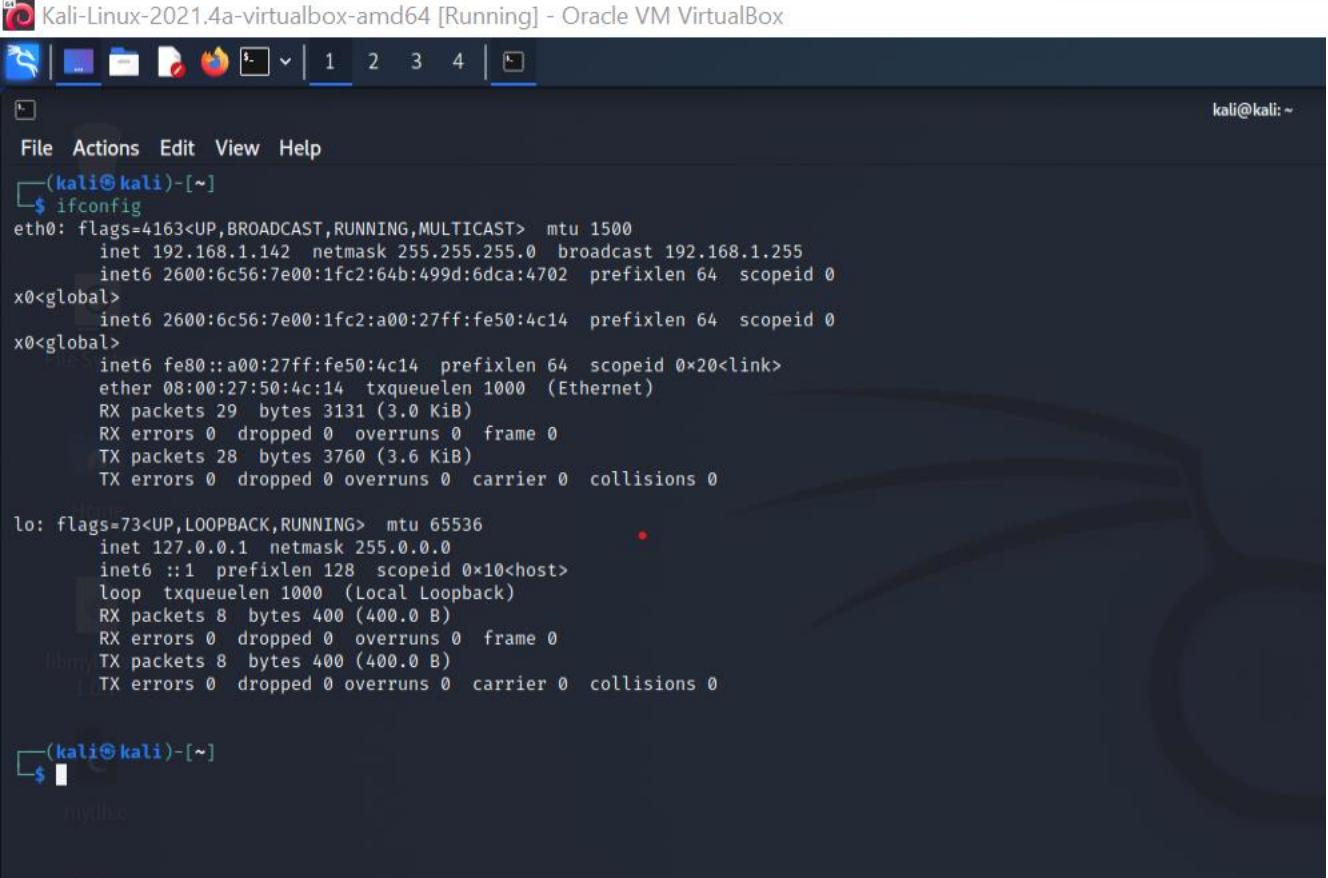
4. Start both the VMs and login with the respective credentials (Kali: osboxes/osboxes.org, Ubuntu: sec-lab/untccdc).
5. On Kali VM, start the Terminal and type “ifconfig”. Notice the IP address and interface name (It may be different in your scenario):

Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab



Kali-Linux-2021.4a-virtualbox-amd64 [Running] - Oracle VM VirtualBox

```
(kali㉿kali)-[~]
└─$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.142 netmask 255.255.255.0 broadcast 192.168.1.255
        inet6 2600:6c56:7e00:1fc2:64b:499d:6dca:4702 prefixlen 64 scopeid 0
    x0<global>
        inet6 2600:6c56:7e00:1fc2:a00:27ff:fe50:4c14 prefixlen 64 scopeid 0
    x0<global>
        inet6 fe80::a00:27ff:fe50:4c14 prefixlen 64 scopeid 0x20<link>
            ether 08:00:27:50:4c:14 txqueuelen 1000 (Ethernet)
            RX packets 29 bytes 3131 (3.0 KiB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 28 bytes 3760 (3.6 KiB)
            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 8 bytes 400 (400.0 B)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 8 bytes 400 (400.0 B)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

(kali㉿kali)-[~]
└─$
```

The screenshot shows a terminal window titled "Kali-Linux-2021.4a-virtualbox-amd64 [Running] - Oracle VM VirtualBox". The terminal is running on a Kali Linux system with the command "ifconfig" entered. The output shows two network interfaces: "eth0" and "lo". The "eth0" interface has an IP address of 192.168.1.142 and a MAC address of 08:00:27:50:4c:14. The "lo" interface is the loopback interface with an IP address of 127.0.0.1. The terminal prompt "(kali㉿kali)-[~]" is visible at the bottom.

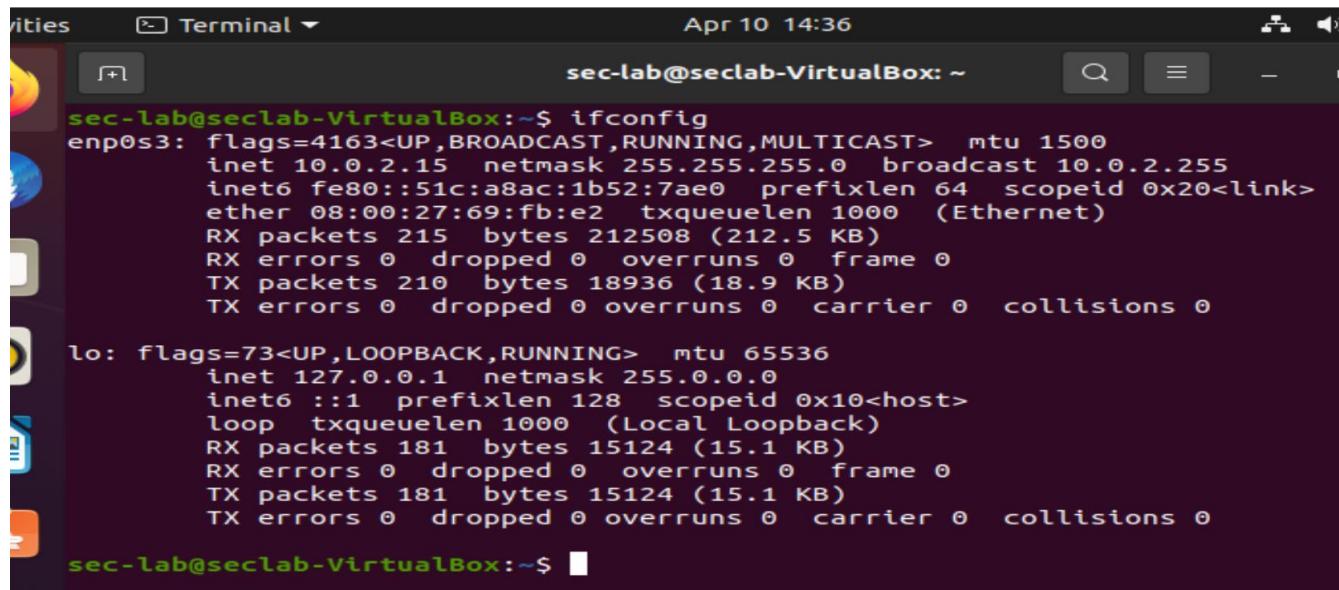
6. On Ubuntu VM, open the Terminal and type “ifconfig”. Notice the IP address and interface name (it may be different in your experiment):

Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab



```
sec-lab@seclab-VirtualBox:~$ 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::fe80:1b52:7ae0 prefixlen 64 scopeid 0x20<link>
                ether 08:00:27:69:fb:e2 txqueuelen 1000 (Ethernet)
                  RX packets 215 bytes 212508 (212.5 KB)
                  RX errors 0 dropped 0 overruns 0 frame 0
                  TX packets 210 bytes 18936 (18.9 KB)
                  TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

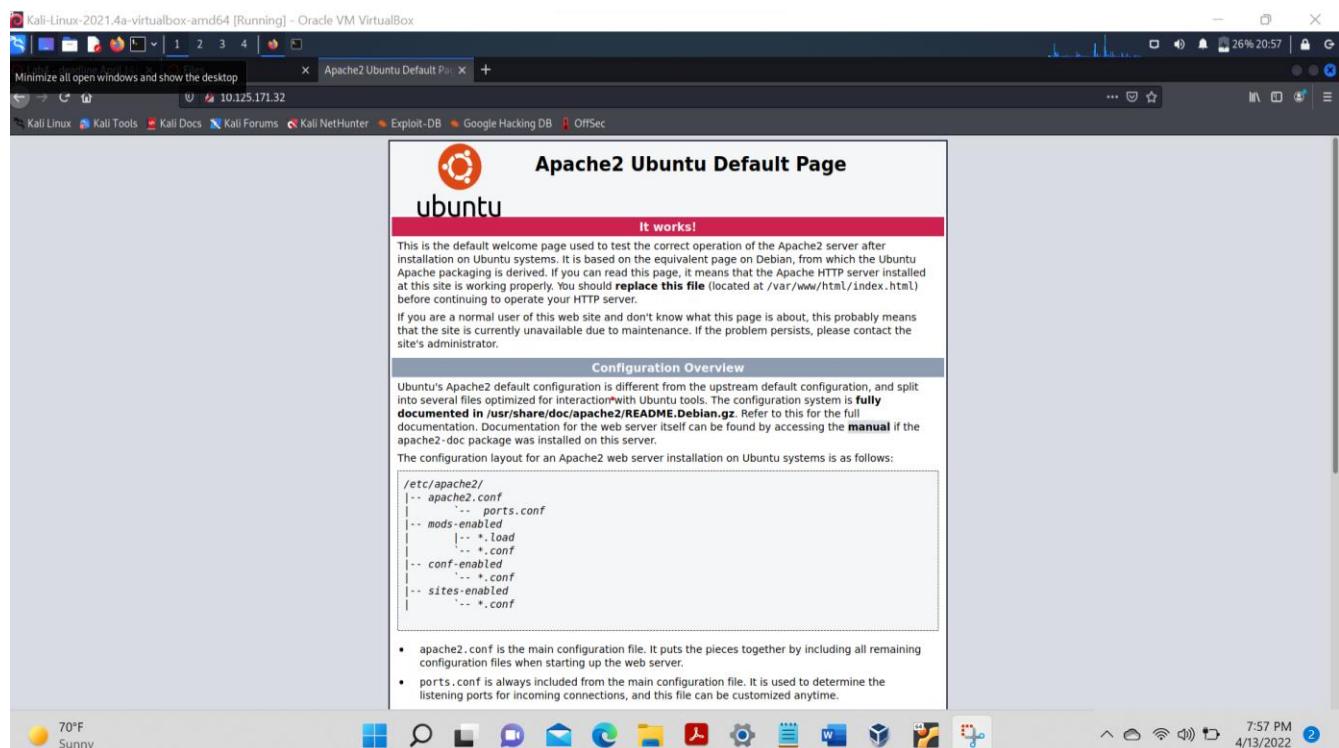
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
      inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
          loop txqueuelen 1000 (Local Loopback)
            RX packets 181 bytes 15124 (15.1 KB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 181 bytes 15124 (15.1 KB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

sec-lab@seclab-VirtualBox:~$
```

7. Next, we will access the Ubuntu VM from Kali VM via HTTP.

Open the Firefox browser in Kali and type `http://<ip address>` as shown below.

You may expect to see the page as shown below.



Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab

8. From Kali VM, let us run an attack against the web server as follows:

Open terminal and type below command

```
sudo hping3 <web server ip address> -p 80 -S --flood --rand-source
```

Let us briefly discuss the parameters of the above command.

- -p 80 : since we would like to attack the web server on the Ubuntu VM, we set the HTTP port 80.
- -S : specifies SYN packets.
- --flood : the packets are send as fast as possible and replies are ignored.
- --rand-source : a random IP source address is set.

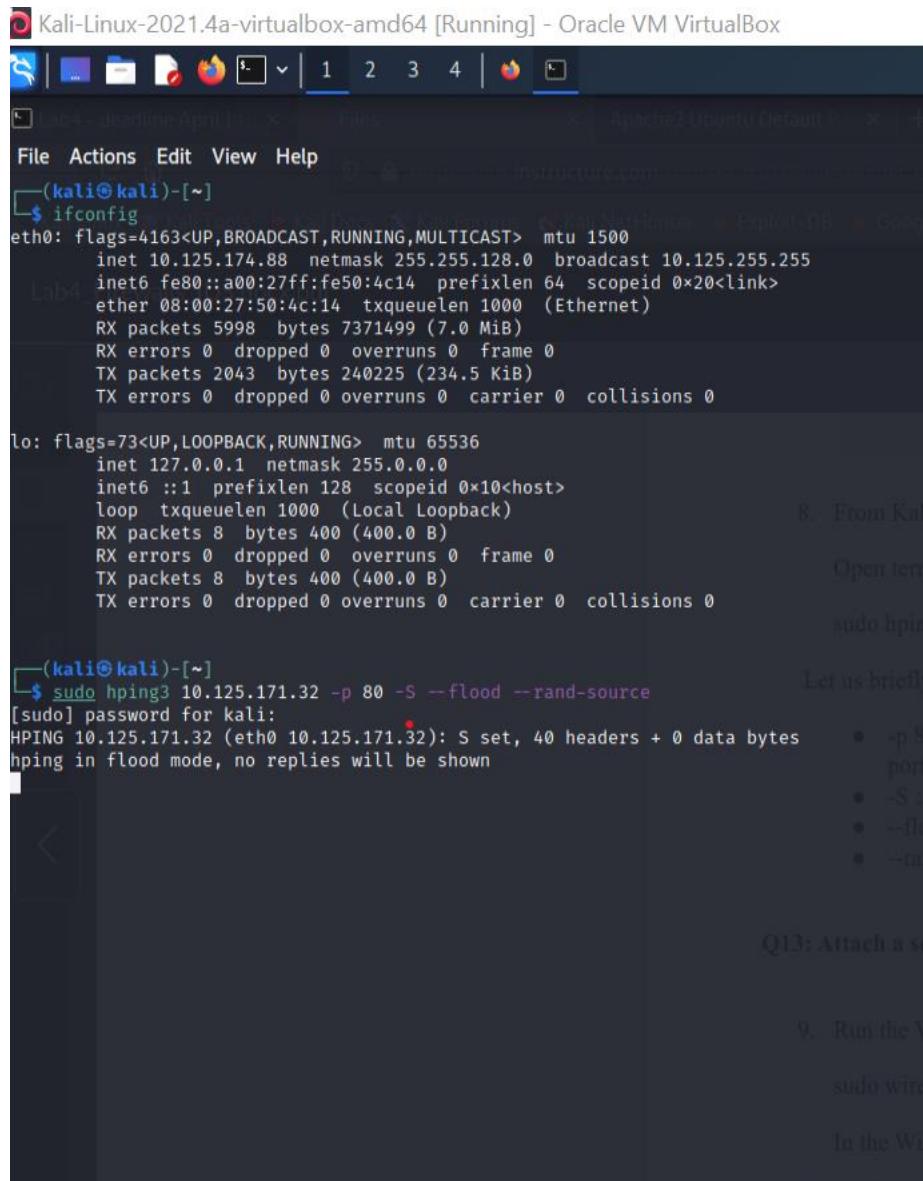
Q13: Attach a screenshot of the result.

Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab



Kali-Linux-2021.4a-virtualbox-amd64 [Running] - Oracle VM VirtualBox

```
(kali㉿kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
      inet 10.125.174.88  netmask 255.255.128.0  broadcast 10.125.255.255
          inet6 fe80::a00:27ff:fe50:4c14  prefixlen 64  scopeid 0x20<link>
            ether 08:00:27:50:4c:14  txqueuelen 1000  (Ethernet)
              RX packets 5998  bytes 7371499 (7.0 MiB)
              RX errors 0  dropped 0  overruns 0  frame 0
              TX packets 2043  bytes 240225 (234.5 KiB)
              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 8  bytes 400 (400.0 B)
              RX errors 0  dropped 0  overruns 0  frame 0
              TX packets 8  bytes 400 (400.0 B)
              TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

(kali㉿kali)-[~]
$ sudo hping3 10.125.171.32 -p 80 -S --flood --rand-source
[sudo] password for kali:
HPING 10.125.171.32 (eth0 10.125.171.32): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
```

9. Run the Wireshark on the Ubuntu VM:

sudo wireshark

In the Wireshark, start capturing packets.

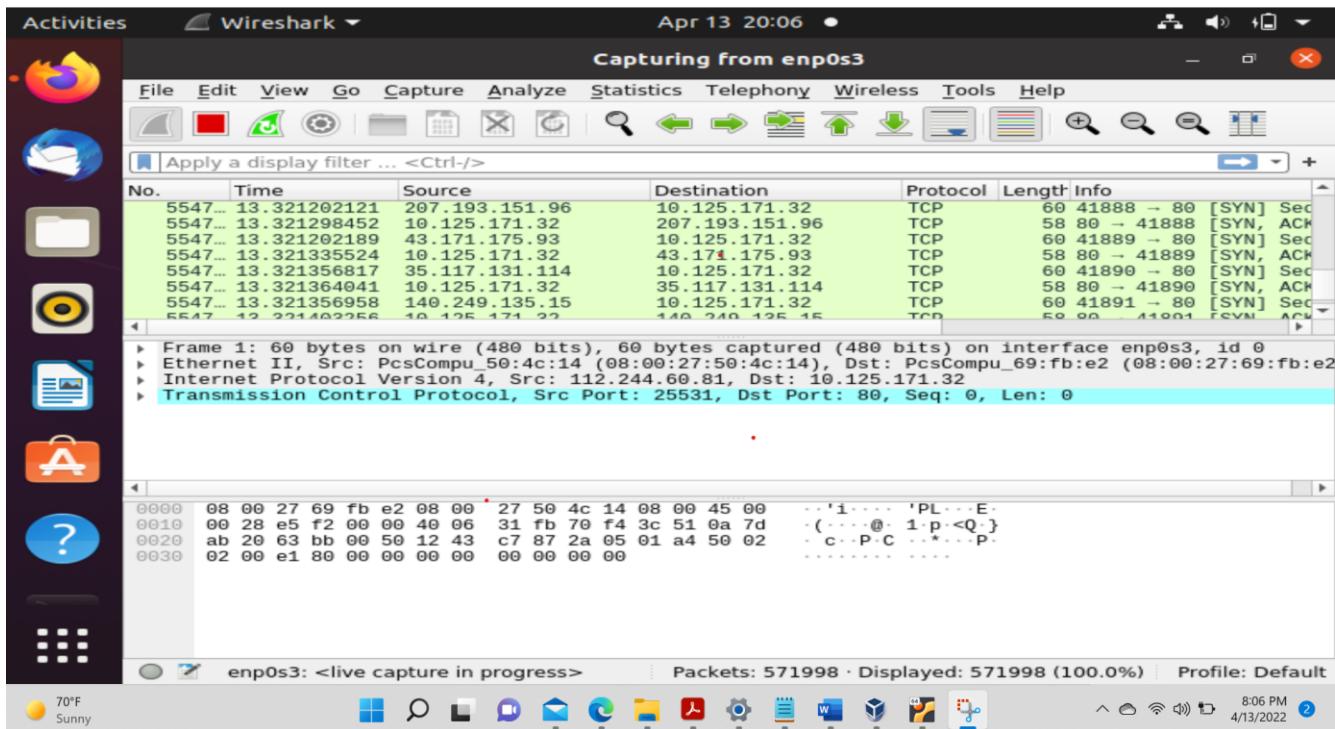
Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab

Q14: Attach a screenshot of the result.



10. Access the Ubuntu VM via browser with address (<http://<Ubuntu VM IP address>>) from your Kali VM – use a new private window in the browser. Then, refresh your browser window several times. Probably, you can still access the Ubuntu VM, which shows that a single instance of hping have not yet overloaded the web server. If this is the case, then run another instance of the hping utility (use a new terminal window) – for this, repeat the steps described in Item 8.
11. If you are still able to access the web server, keep adding the hping instances until the access to the Ubuntu VM via browser becomes very slow (or stops completely). Now, this would indicate that the VM is overloaded by your DoS attack.

Note: Keep adding the hping instances until you have 10 of them running simultaneously. After that, proceed to the next step.

Q15: Report your observations from the above steps. In particular, how many *hping* instances did you have to run? Did it slow down the access to Ubuntu VM via browser?

Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab

We had run 10 hping instances at the same time. Yes, it did slow down the access to Ubuntu VM via browser, taking 4 to 5 seconds to load the page. The Ubuntu VM and the virtual box-hosting system is also not working efficiently. I unable to access wireshark or make any changes, not even on the screen.

12. Stop all the hping instances on Kali VM (press Ctrl+C for exiting an hping process).
Make sure that you stop/kill all the hping instances. At this moment, your VM might be still overloaded, and you may not have access to it. If this is the case, rebooting it should solve the issue. Close Wireshark on Ubuntu.

Section 3: Intrusion Detection Systems (IDS) – Snort

IDS is software/hardware system which monitors network traffic for attacks or policy violations. The policies are defined as sets of rules. When their violation is detected, IDS typically alert administrators and/or automatically deploy certain pre-set countermeasures. IDS are typically used inside the local network as the next line of defense after firewalls.

In this section, we will use Snort IDS (<https://www.snort.org>) to detect the TCP SYN flood attack. Again, the attacker will use the Kali VM and the Ubuntu VM will be a target.

13. Let us first install and configure Snort on the Ubuntu VM. (Note that Snort comes with some default rulesets.) Run the following command:

```
sudo apt-get install snort
```

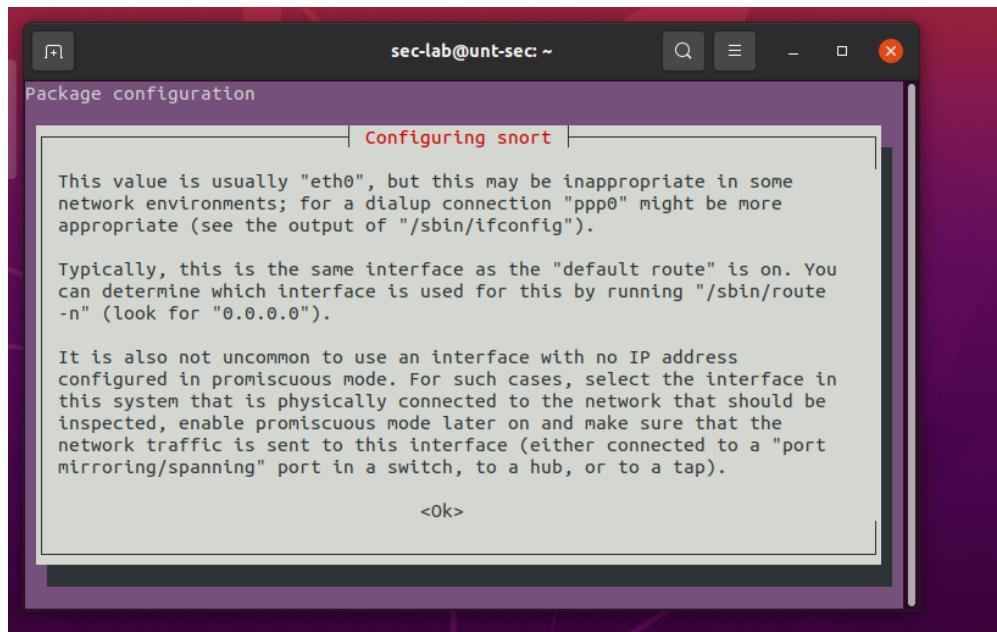
You may expect to see notification similar that shown below.

Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab



14. Press Tab on the keyboard to move your selection to <Ok> and press Enter.

During the installation, you will need to specify your network information. Input your interface name as shown below. In our case, it was “enp0s3”. You can get your network information using the command “ifconfig”, as shown earlier.

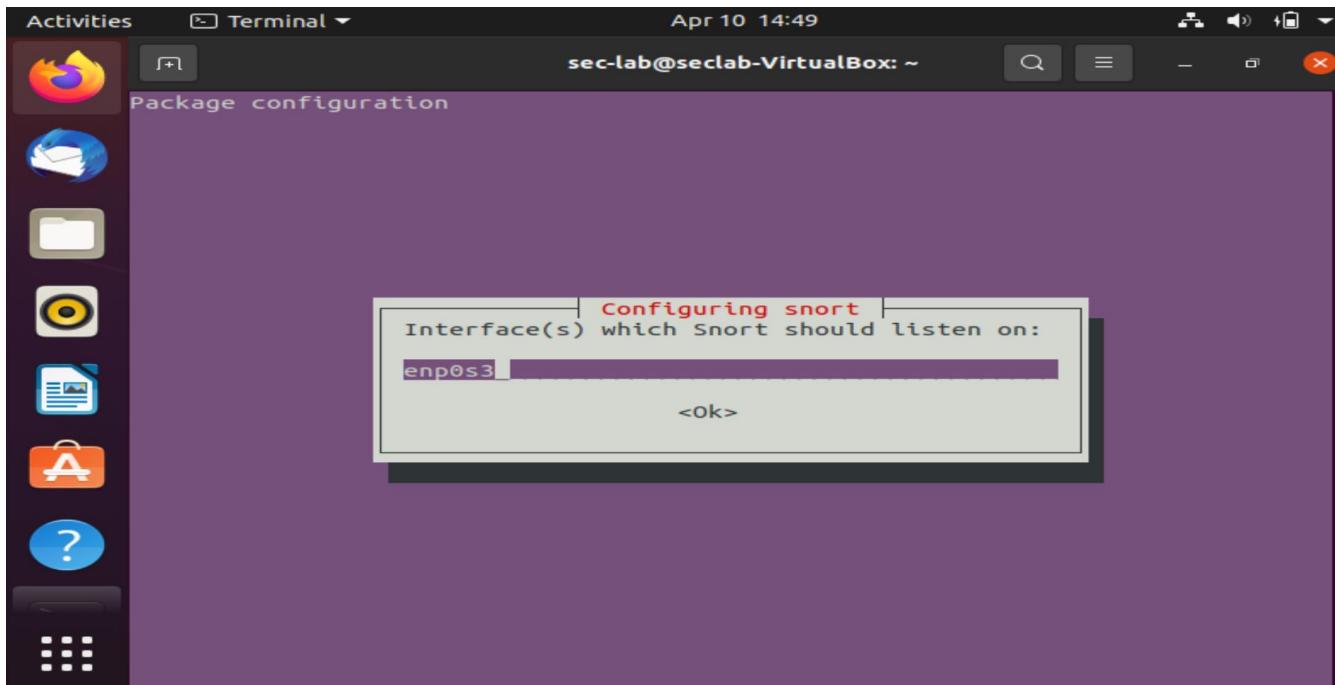
Note: If the terminal shows that ifconfig command is missing, then install the package *net-tools* using the command “sudo apt-get install net-tools”.

Student Name:

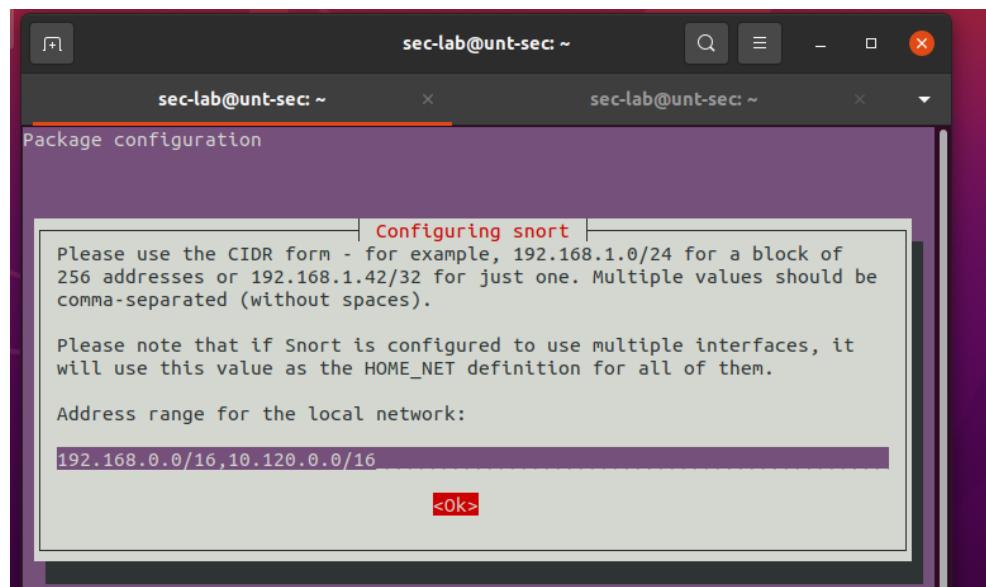
Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab



15. Add network 10.120.0.0/16 to the next page with a comma as shown below.



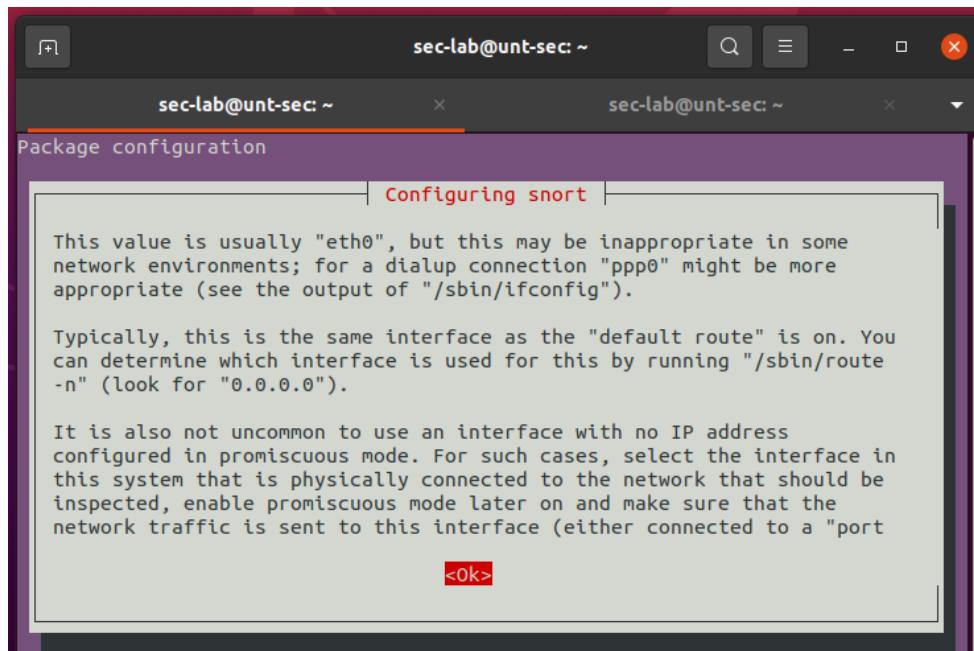
Press Tab and then press Enter in the next screen.

Student Name:

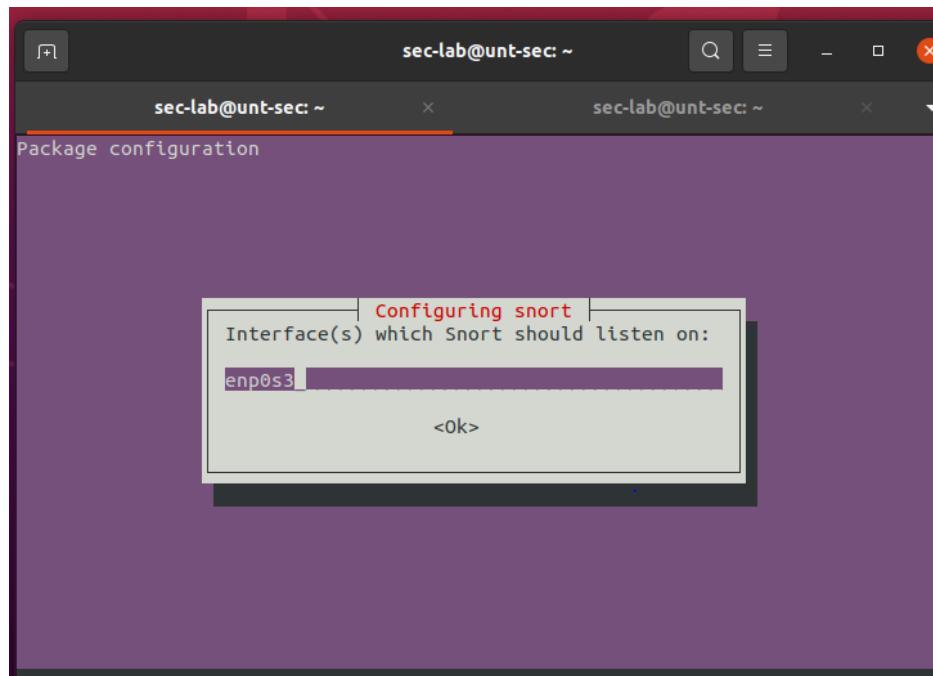
Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab



16. On the next page, select <Ok> by pressing Tab and then Enter.



After that, your installation will be complete.

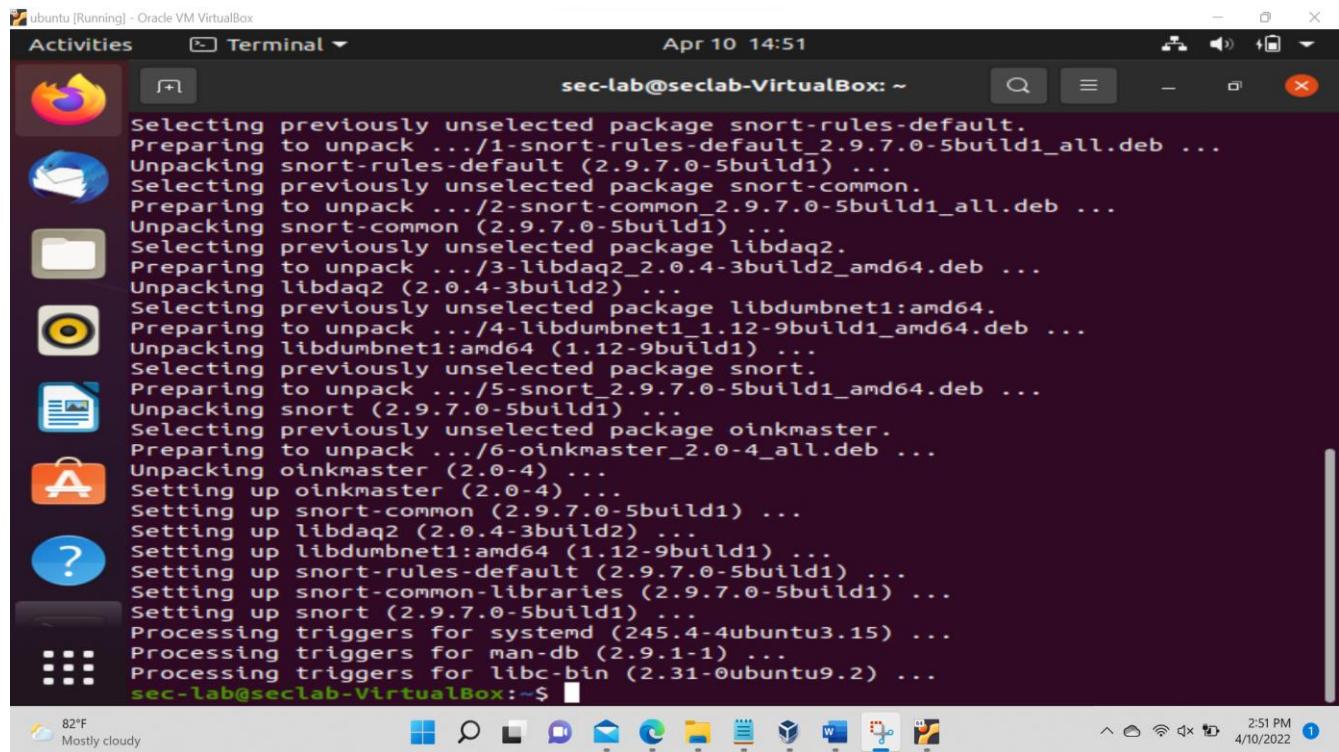
Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab

Q16: Attach a screenshot of the result. (Only one screenshot showing the terminal after the installation has been completed is enough.)



The screenshot shows a terminal window titled "Terminal" with the command "sudo apt-get install snort" run by user "sec-lab". The output lists the installation of several Snort-related packages, including snort-rules-default, snort-common, libdaq2, libdumbnet1:amd64, oinkmaster, and snort. The terminal window is part of a desktop environment with a dock at the bottom containing icons for various applications like a browser, file manager, and system tools.

```
ubuntu [Running] - Oracle VM VirtualBox
Activities Terminal Apr 10 14:51
sec-lab@seclab-VirtualBox: ~
Selecting previously unselected package snort-rules-default.
Preparing to unpack .../1-snort-rules-default_2.9.7.0-5build1_all.deb ...
Unpacking snort-rules-default (2.9.7.0-5build1) ...
Selecting previously unselected package snort-common.
Preparing to unpack .../2-snort-common_2.9.7.0-5build1_all.deb ...
Unpacking snort-common (2.9.7.0-5build1) ...
Selecting previously unselected package libdaq2.
Preparing to unpack .../3-libdaq2_2.0.4-3build2_amd64.deb ...
Unpacking libdaq2 (2.0.4-3build2) ...
Selecting previously unselected package libdumbnet1:amd64.
Preparing to unpack .../4-libdumbnet1_1.12-9build1_amd64.deb ...
Unpacking libdumbnet1:amd64 (1.12-9build1) ...
Selecting previously unselected package snort.
Preparing to unpack .../5-snort_2.9.7.0-5build1_amd64.deb ...
Unpacking snort (2.9.7.0-5build1) ...
Selecting previously unselected package oinkmaster.
Preparing to unpack .../6-oinkmaster_2.0-4_all.deb ...
Unpacking oinkmaster (2.0-4) ...
Setting up oinkmaster (2.0-4) ...
Setting up snort-common (2.9.7.0-5build1) ...
Setting up libdaq2 (2.0.4-3build2) ...
Setting up libdumbnet1:amd64 (1.12-9build1) ...
Setting up snort-rules-default (2.9.7.0-5build1) ...
Setting up snort-common-libraries (2.9.7.0-5build1) ...
Setting up snort (2.9.7.0-5build1) ...
Processing triggers for systemd (245.4-4ubuntu3.15) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for libc-bin (2.31-0ubuntu9.2) ...
sec-lab@seclab-VirtualBox:~$
```

17. Next, we will slightly change the default output format of Snort to make it convenient for the purpose of this lab.

Edit the file `/etc/snort/snort.debian.conf` with the root privilege (use “`sudo`”) and update the line starting with `DEBIAN_SNORT_OPTIONS` with the following:

```
DEBIAN_SNORT_OPTIONS="-A full"
```

18. Now, let us add a custom rule for the TCP SYN flood detection. For that, we will add the following rule in the file `/etc/snort/rules/dos.rules` (as a new line at the end) using the root privilege:

```
alert tcp any any -> $HOME_NET 80 (flags: S; msg:"Possible TCP SYN flood";
detection_filter: track by_dst, count 50, seconds 10; sid:1000001;)
```

- The first six parameters of the rule require to generate an alert for a TCP connection from any source IP address and port to any IP address in the protected network, port 80 (http).

Student Name:

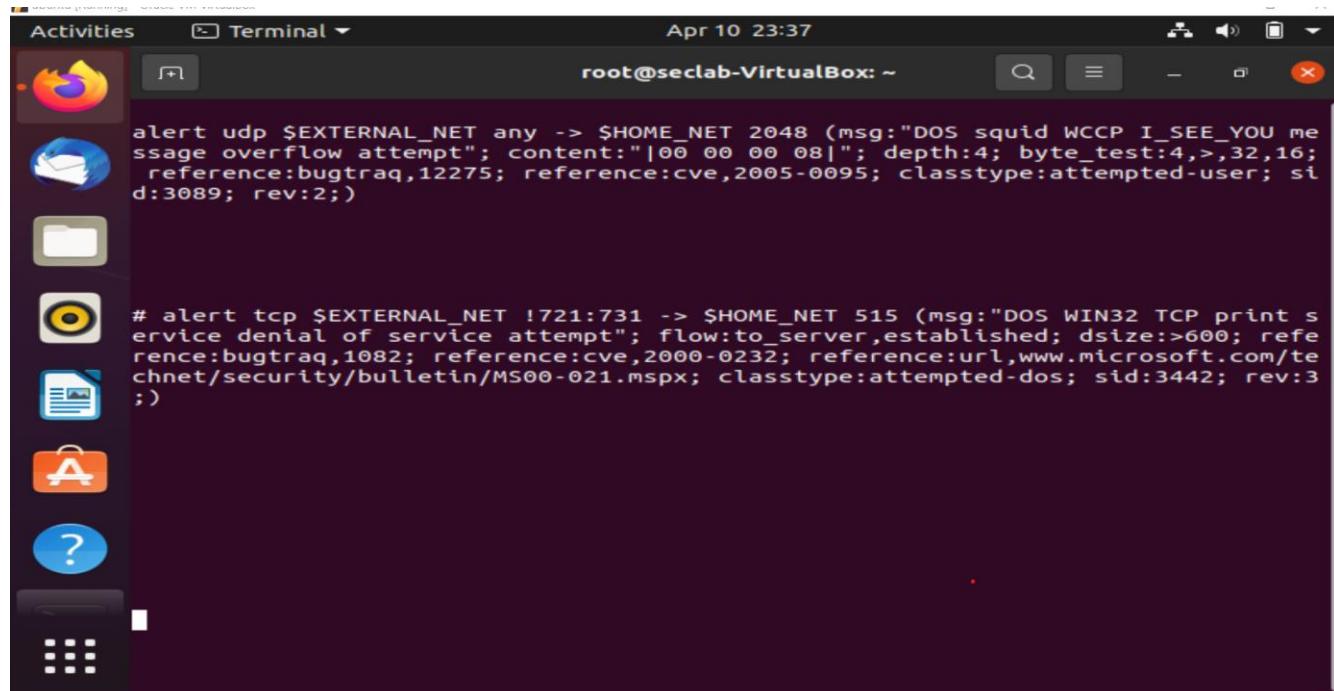
Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab

- The following conditions, which are specified in the brackets, need to be met: the packet must be SYN, and the following rate must be exceeded (counted by the destination IP): within each 10-second frame, after 50 packets which satisfy the conditions, each next packet will trigger an alert. Note that these condition define what frequency of packets will be considered as attack – this frequency must be substantially higher compared to what you normally expect for your network.
- The last parameter is a Snort identifier of your new rule. (We use this number because the SID's before 1000000 are reserved by Snort.)

Q17: Attach a screenshot of the updated *dos.rules* file.



The screenshot shows a terminal window titled "Terminal" running as root on a system named "seclab-VirtualBox". The terminal displays two Snort rules. The first rule is for UDP traffic from \$EXTERNAL_NET to \$HOME_NET port 2048, filtering for a specific message payload and depth, and setting a reference to CVE-2005-0095. The second rule is for TCP traffic from \$EXTERNAL_NET to \$HOME_NET port 515, filtering for a service denial of service attempt, and setting a reference to CVE-2000-0232. Both rules have a class type of "attempted-user" and a sid of 3089 and 3442 respectively.

```
root@seclab-VirtualBox: ~
alert udp $EXTERNAL_NET any -> $HOME_NET 2048 (msg:"DOS squid WCCP I_SEE_YOU me
ssage overflow attempt"; content:"|00 00 00 08|"; depth:4; byte_test:4,>,32,16;
reference:bugtraq,12275; reference:cve,2005-0095; classtype:attempted-user; si
d:3089; rev:2;)

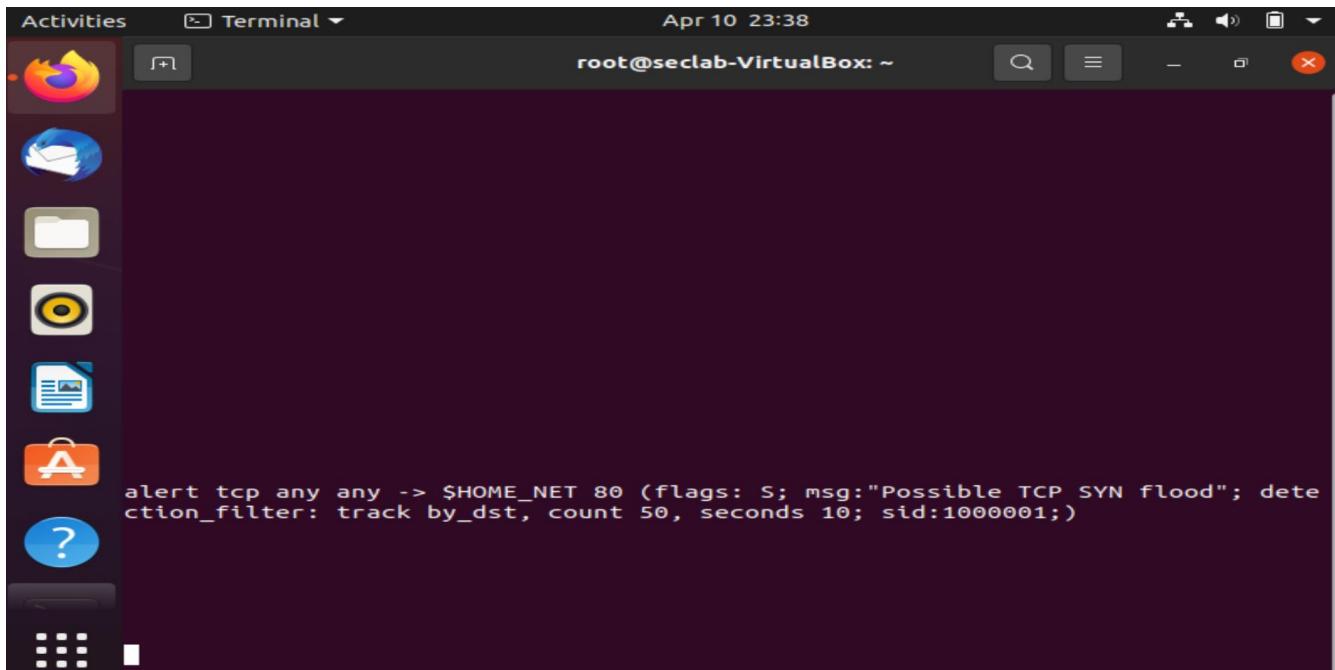
# alert tcp $EXTERNAL_NET !721:731 -> $HOME_NET 515 (msg:"DOS WIN32 TCP print s
ervice denial of service attempt"; flow:to_server,established; dsiz
e:>600; refe
rence:bugtraq,1082; reference:cve,2000-0232; reference:url,www.microsoft.com/te
chnet/security/bulletin/MS00-021.mspx; classtype:attempted-dos; sid:3442; rev:3
;)
```

Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab



19. After you updated the *dos.rules* file, restart Snort by typing:

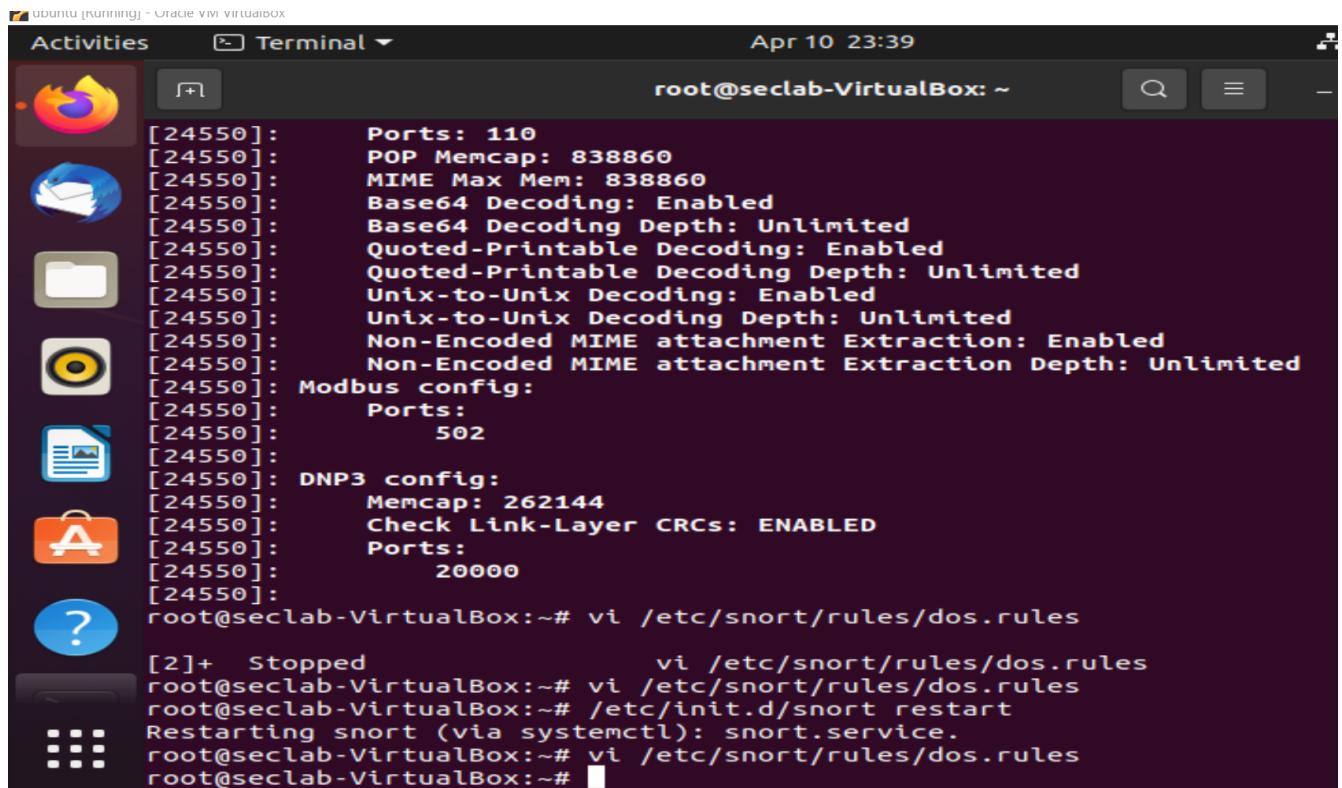
```
/etc/init.d/snort restart
```

Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab



The screenshot shows a Linux desktop environment with a terminal window open. The terminal window title is "root@seclab-VirtualBox: ~". The terminal content displays the configuration of the Snort intrusion detection system (IDS). It includes settings for various protocols and ports, such as POP, MIME, Base64, Quoted-Printable, Unix-to-Unix, Non-Encoded MIME attachment extraction, Modbus, DNP3, and Link-Layer CRCs. The terminal also shows the user editing the file "/etc/snort/rules/dos.rules" with the vi editor, stopping and restarting the snort service, and then returning to the dos.rules file.

```
[24550]: Ports: 110
[24550]: POP Memcap: 838860
[24550]: MIME Max Mem: 838860
[24550]: Base64 Decoding: Enabled
[24550]: Base64 Decoding Depth: Unlimited
[24550]: Quoted-Printable Decoding: Enabled
[24550]: Quoted-Printable Decoding Depth: Unlimited
[24550]: Unix-to-Unix Decoding: Enabled
[24550]: Unix-to-Unix Decoding Depth: Unlimited
[24550]: Non-Encoded MIME attachment Extraction: Enabled
[24550]: Non-Encoded MIME attachment Extraction Depth: Unlimited
[24550]: Modbus config:
[24550]:   Ports:
[24550]:     502
[24550]:
[24550]: DNP3 config:
[24550]:   Memcap: 262144
[24550]:   Check Link-Layer CRCs: ENABLED
[24550]:   Ports:
[24550]:     20000
[24550]:
root@seclab-VirtualBox:~# vi /etc/snort/rules/dos.rules
[2]+  Stopped                  vi /etc/snort/rules/dos.rules
root@seclab-VirtualBox:~# vi /etc/snort/rules/dos.rules
root@seclab-VirtualBox:~# /etc/init.d/snort restart
Restarting snort (via systemctl): snort.service.
root@seclab-VirtualBox:~# vi /etc/snort/rules/dos.rules
root@seclab-VirtualBox:~#
```

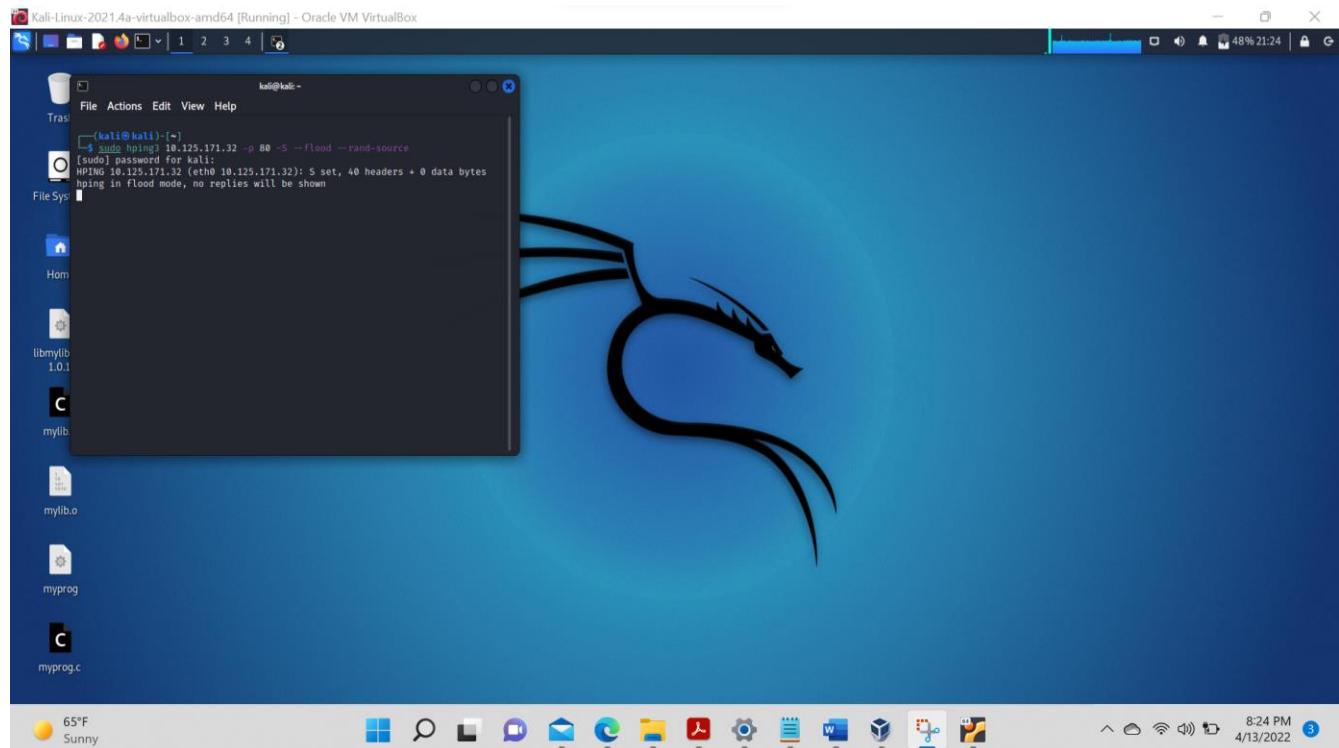
20. Next, run the attack described in Section 2 again from Kali VM (a single hping instance is sufficient).

Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab



21. Monitor the Snort log file, using the following command:

```
tail -f /var/log/snort/alert
```

You should be able to see the output with your own rule's sid (1000001).

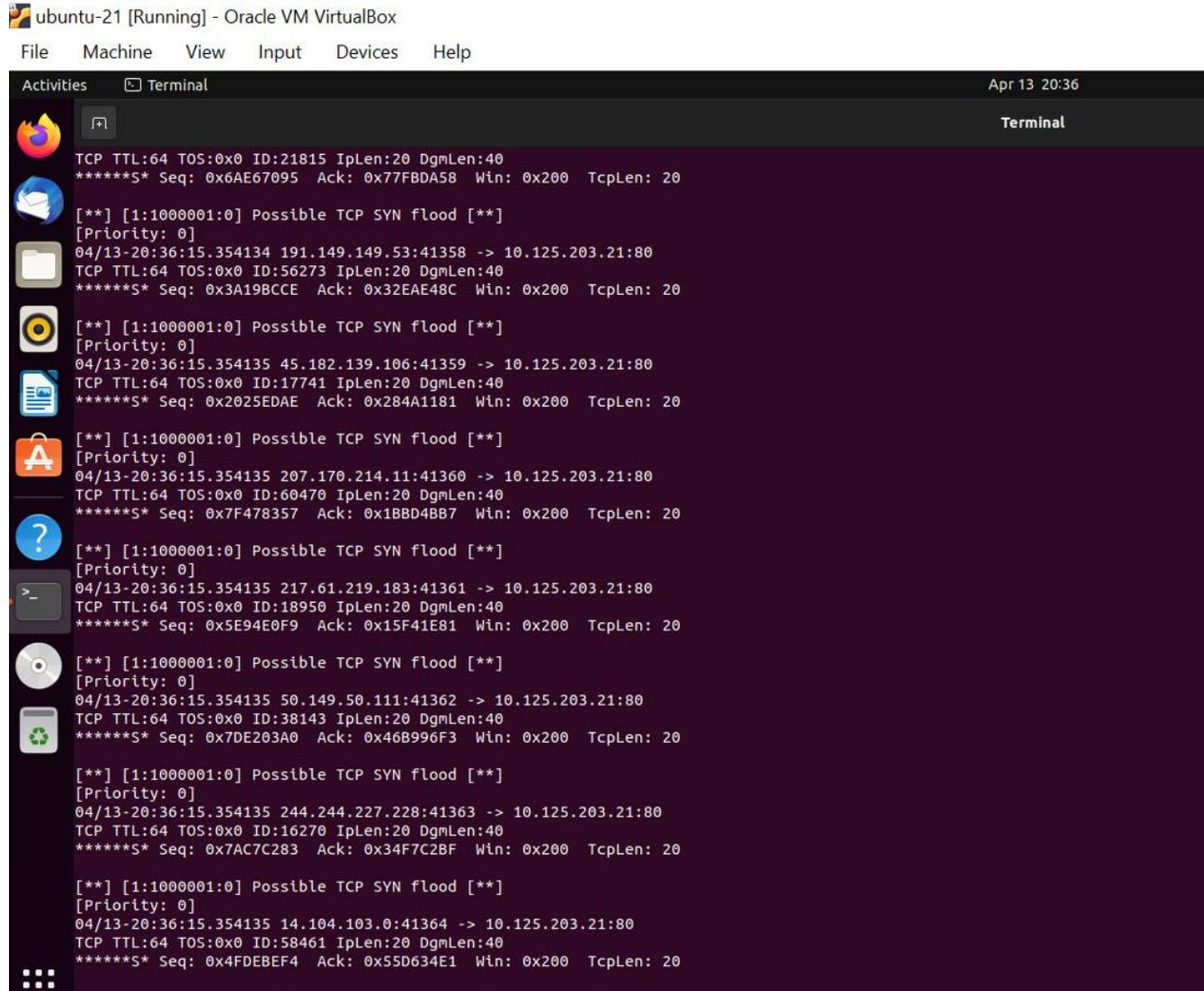
Q18: Attach a screenshot of the result. Explain what you see.

Student Name:

Course: CSCE 5550

Semester: Fall 2021

Firewalls and IDS Lab



The screenshot shows a Kali Linux desktop environment with a terminal window open. The terminal window title is "Terminal" and the date and time are "Apr 13 20:36". The terminal displays a series of Snort alerts, primarily TCP SYN flood detections, from various IP addresses to the local host (10.125.203.21:80). The alerts include details such as sequence numbers, acknowledgment numbers, and window sizes. The desktop background shows standard Kali Linux icons.

```
TCP TTL:64 TOS:0x0 ID:21815 IpLen:20 DgmLen:40
*****S* Seq: 0x6AE67095 Ack: 0x77FBDA58 Win: 0x200 TcpLen: 20
[**] [1:1000001:0] Possible TCP SYN flood [**]
[Priority: 0]
04/13-20:36:15.354134 191.149.149.53:41358 -> 10.125.203.21:80
TCP TTL:64 TOS:0x0 ID:56273 IpLen:20 DgmLen:40
*****S* Seq: 0x3A19BCCE Ack: 0x32EAE48C Win: 0x200 TcpLen: 20
[**] [1:1000001:0] Possible TCP SYN flood [**]
[Priority: 0]
04/13-20:36:15.354135 45.182.139.106:41359 -> 10.125.203.21:80
TCP TTL:64 TOS:0x0 ID:17741 IpLen:20 DgmLen:40
*****S* Seq: 0x2025EDAE Ack: 0x284A1181 Win: 0x200 TcpLen: 20
[**] [1:1000001:0] Possible TCP SYN flood [**]
[Priority: 0]
04/13-20:36:15.354135 207.170.214.11:41360 -> 10.125.203.21:80
TCP TTL:64 TOS:0x0 ID:60470 IpLen:20 DgmLen:40
*****S* Seq: 0x7F478357 Ack: 0xBBBD4BB7 Win: 0x200 TcpLen: 20
[**] [1:1000001:0] Possible TCP SYN flood [**]
[Priority: 0]
04/13-20:36:15.354135 217.61.219.183:41361 -> 10.125.203.21:80
TCP TTL:64 TOS:0x0 ID:18950 IpLen:20 DgmLen:40
*****S* Seq: 0x5E94E0F9 Ack: 0x15F41E81 Win: 0x200 TcpLen: 20
[**] [1:1000001:0] Possible TCP SYN flood [**]
[Priority: 0]
04/13-20:36:15.354135 50.149.50.111:41362 -> 10.125.203.21:80
TCP TTL:64 TOS:0x0 ID:38143 IpLen:20 DgmLen:40
*****S* Seq: 0x7DE203A0 Ack: 0x46B996F3 Win: 0x200 TcpLen: 20
[**] [1:1000001:0] Possible TCP SYN flood [**]
[Priority: 0]
04/13-20:36:15.354135 244.244.227.228:41363 -> 10.125.203.21:80
TCP TTL:64 TOS:0x0 ID:16270 IpLen:20 DgmLen:40
*****S* Seq: 0x7AC7C283 Ack: 0x34F7C2BF Win: 0x200 TcpLen: 20
[**] [1:1000001:0] Possible TCP SYN flood [**]
[Priority: 0]
04/13-20:36:15.354135 14.104.103.0:41364 -> 10.125.203.21:80
TCP TTL:64 TOS:0x0 ID:58461 IpLen:20 DgmLen:40
*****S* Seq: 0x4FDEBEF4 Ack: 0x55D634E1 Win: 0x200 TcpLen: 20
:::
```

A rule was set to allow snort to detect flooding requests from any server. Hence, we can see TCP flooding requests in the snort alerts file of Kali VM.