

CSE 406 Final Report

TCP Reset Attack On Video Streaming

Course: CSE 406

Lab Group: B1

Submitted By:

Rishov Paul

(1605084)

Submitted To:

Dr. Md. Shohrab Hossain

Abu Wasif

Md. Toufikuzzaman

Steps of Attack:

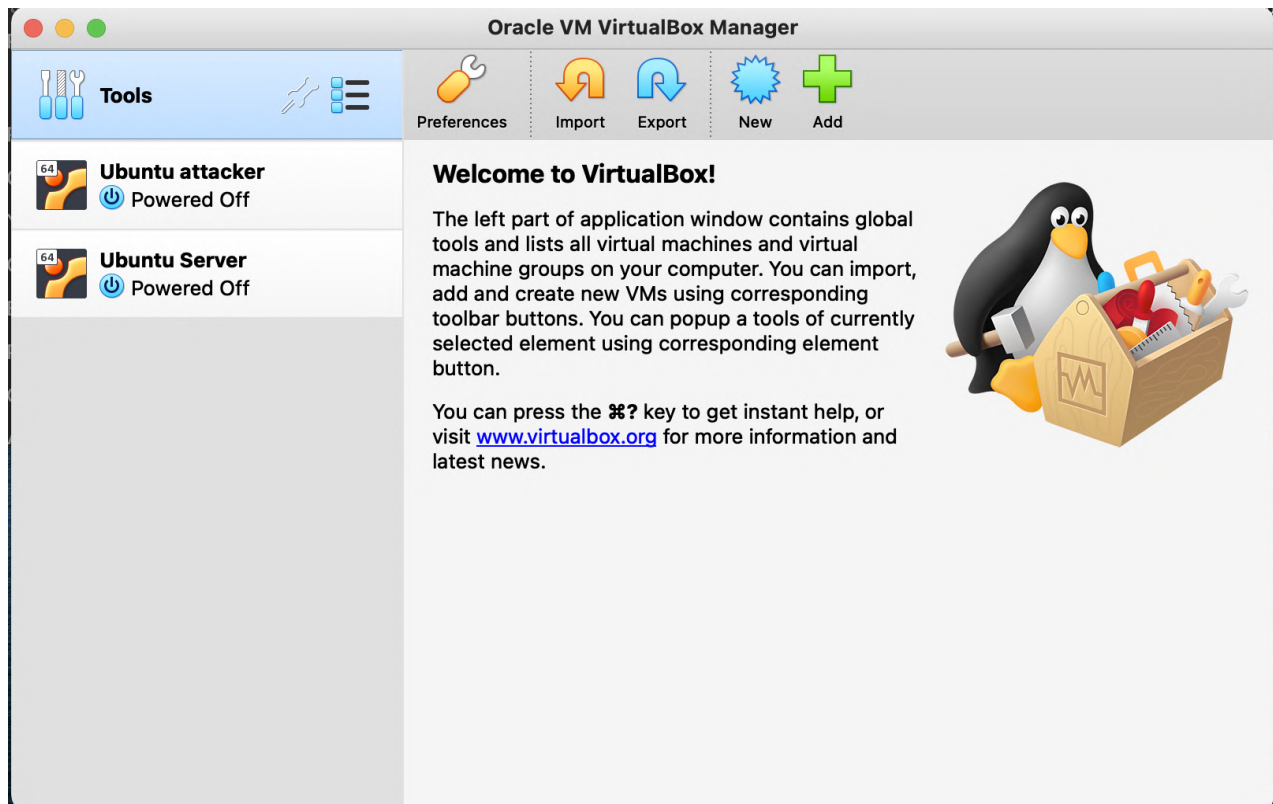
1.Installation:

- Wireshark software
- Pcap library using the following command
 - **sudo apt-get update -y**
 - **sudo apt-get install -y libpcap-dev**

2.Environment Setup:

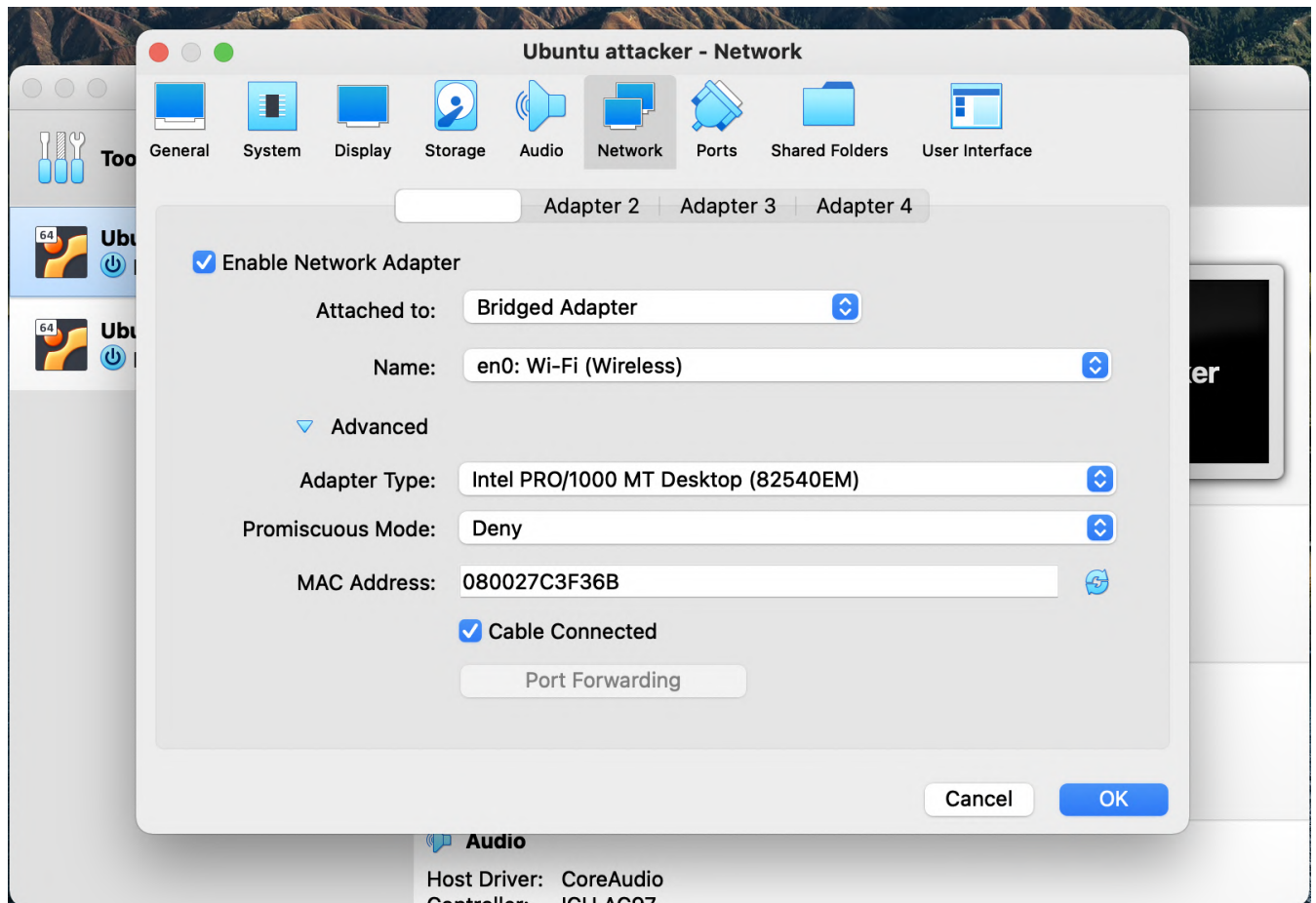
Step-1:

Two virtual machines and one host machine were used to implement the attack. I have created two virtual machines named “Ubuntu Server” which streams the video to the victim host machine and “Ubuntu attacker” which is the attacker. VLC media player was used to stream the video on wireless LAN.



Step-2:

Here to get all the machines in the same network we have connected the virtual machines via Bridged Adapter from the Network Settings of the Virtual Box and as I used the attacker machine to clone the Server machine, I need to refresh the Mac Address so that every machine gets a unique mac address. This network settings must be done on both the attacker and server machine.

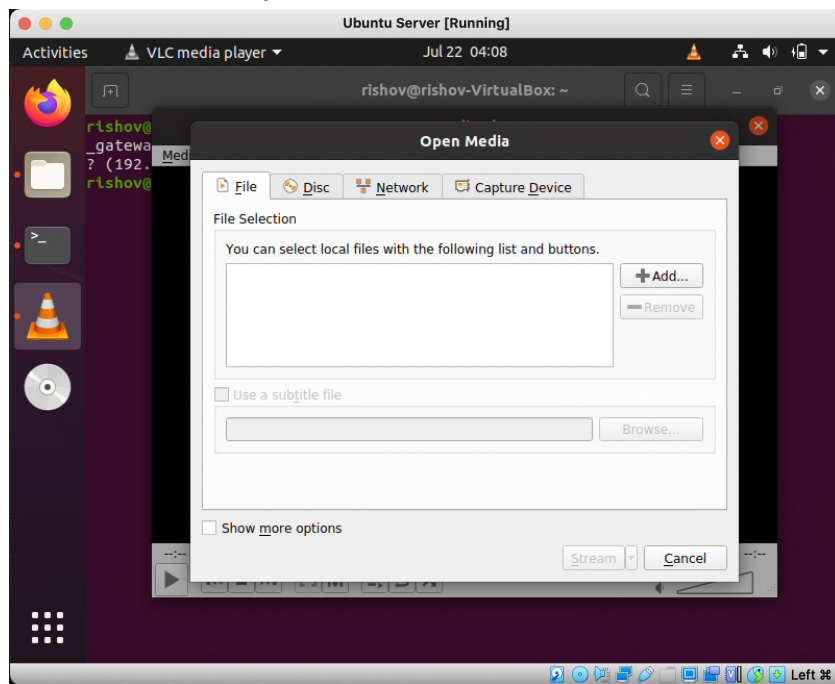


3.Stream Setup:

For Streaming I used VLC media player. So I installed VLC on the Server and the Victim host machine.

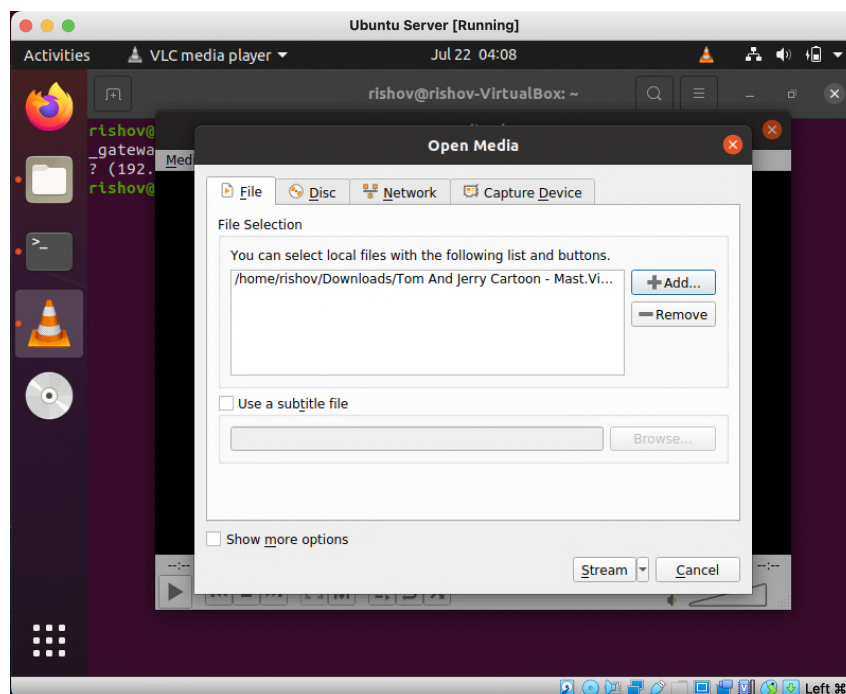
Step-1:

Open VLC media player from the Server machine. Go to Media → Stream and the following window will come up.



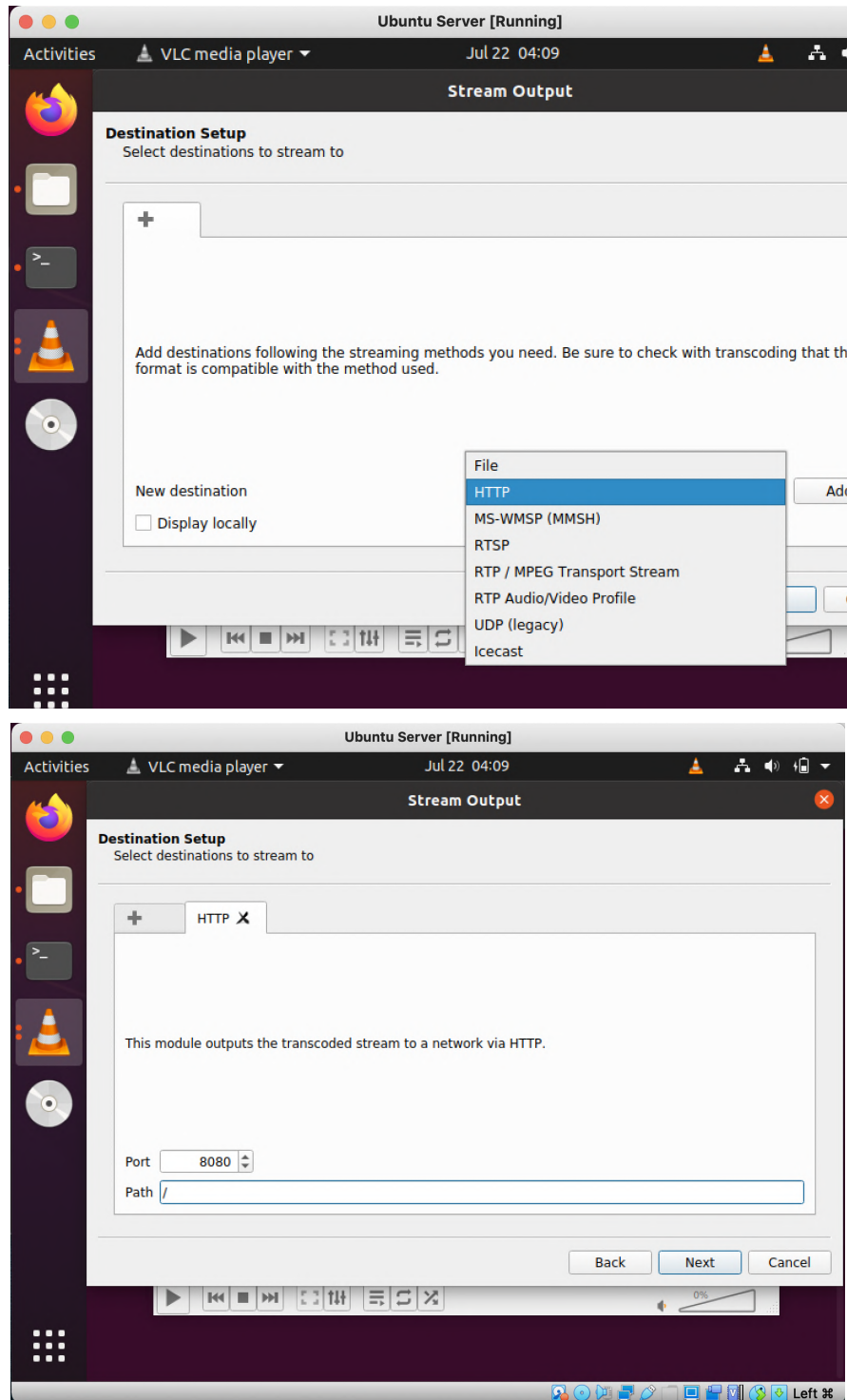
Step-2:

Click the **+Add** button, select the preferred video to stream and then hit the **Stream** button.



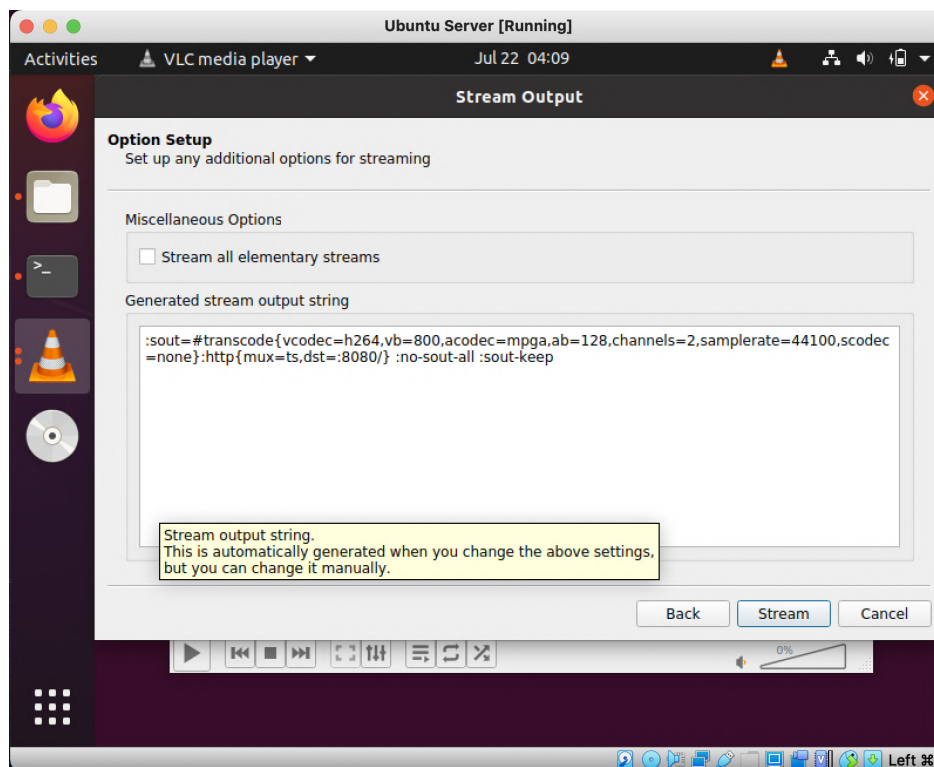
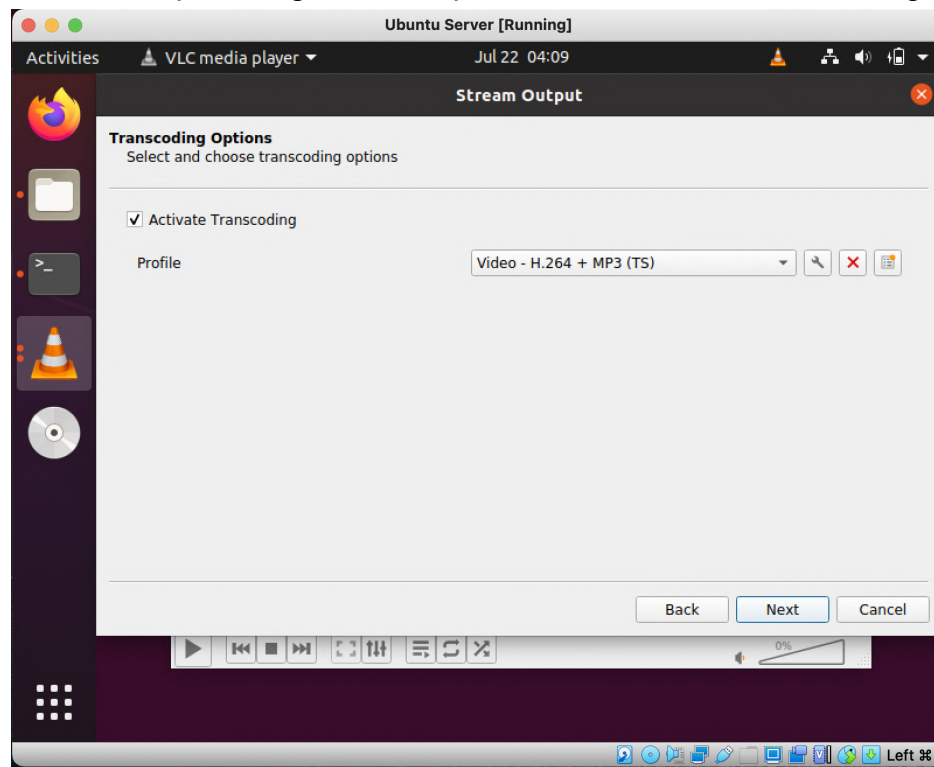
Step-3:

For Stream Output we need to select HTTP and click Add button to define a port. In my case the port was 8080.



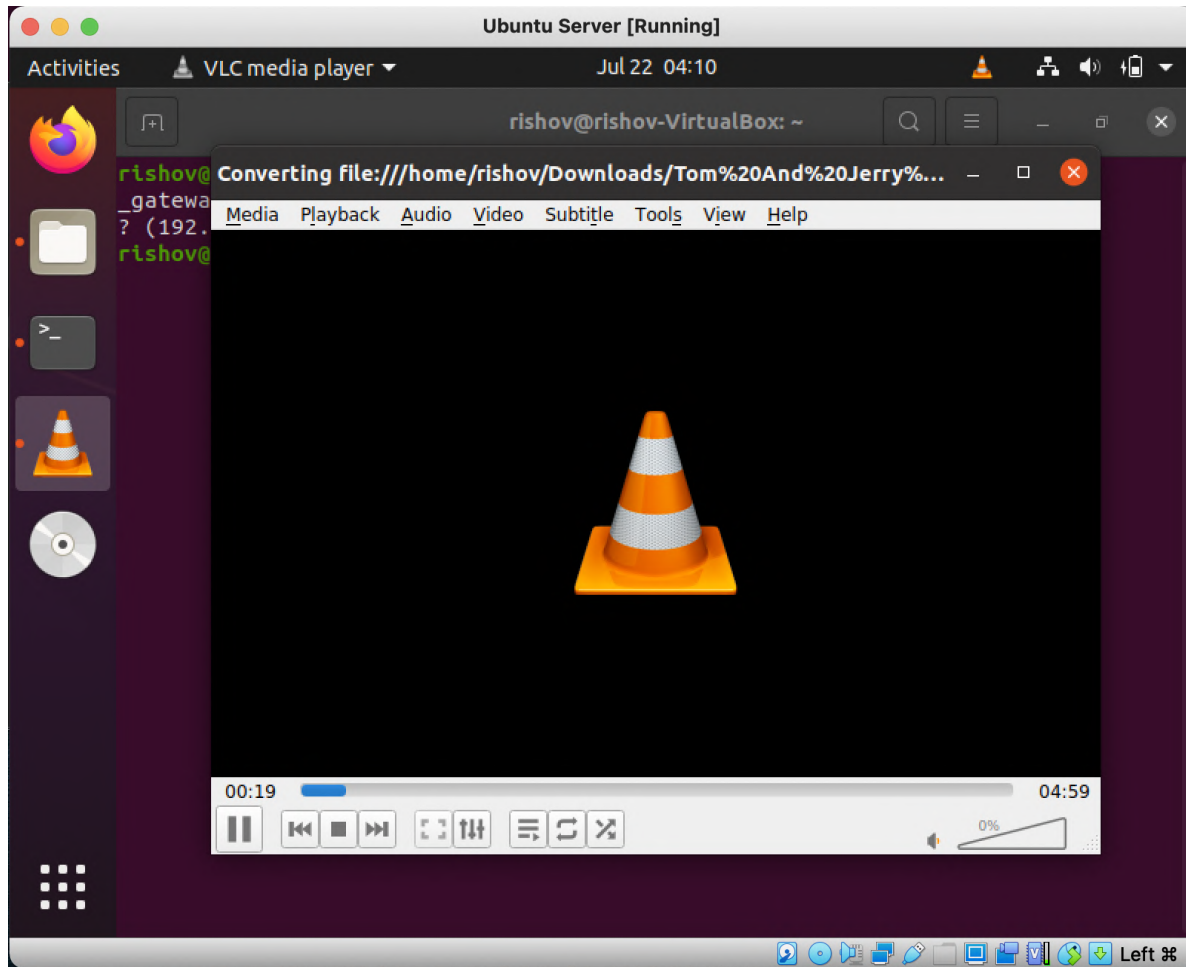
Step-4:

From Profile options, we need to select Video - H.264 + MP3 (TS) and click next to generate stream output string and if we press stream, the stream will begin.



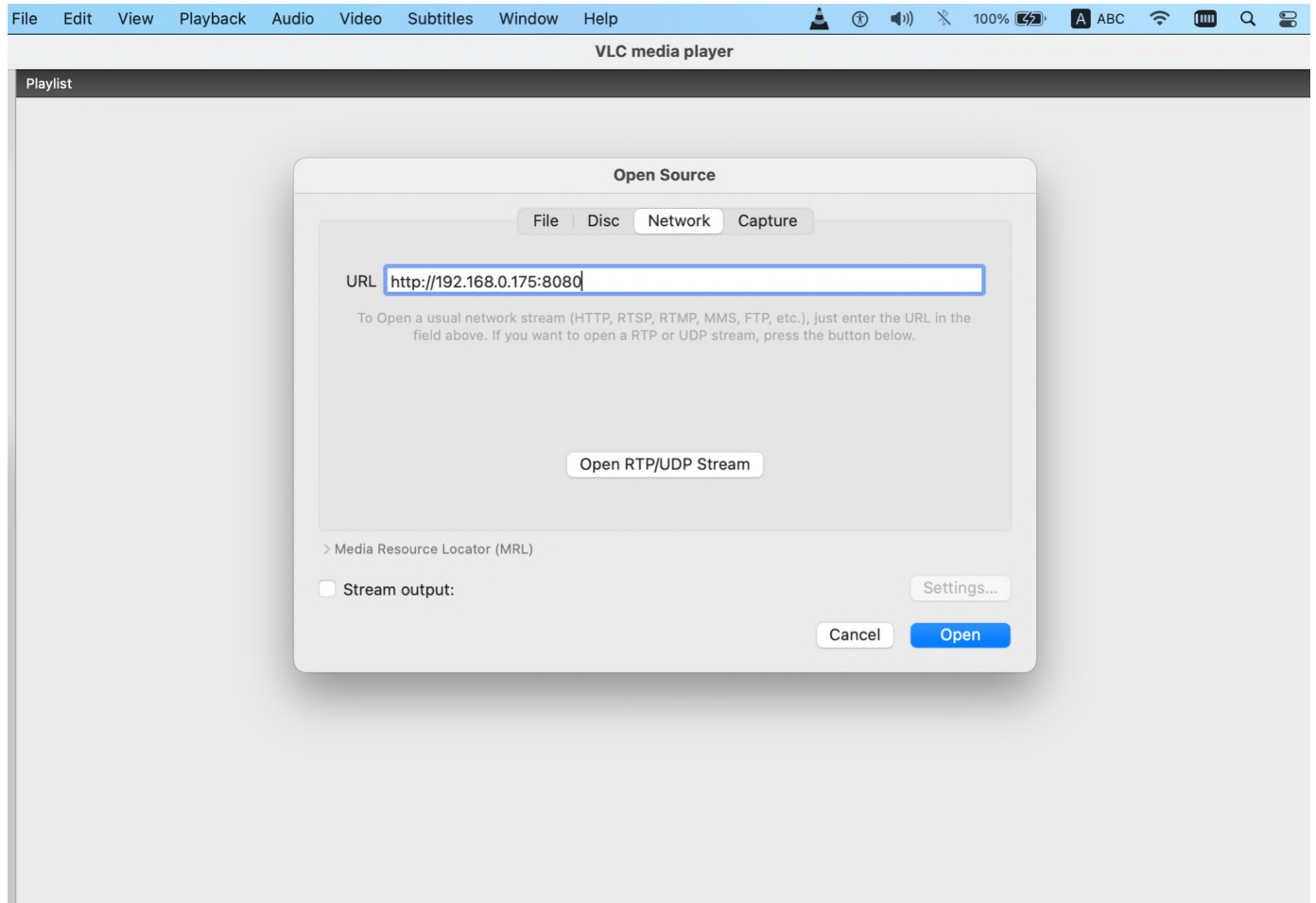
Step-5:

In the server machine, we can see the blue bar is increasing which indicates the server has started streaming.



Step-6:

From the host victim machine, open VLC media player and go to Files → Open Network Option.. and a window like below will come up and we need to input in the network URL field. The syntax of the URL is “[http://server IP address:Port number](http://server_IP_address:Port_number)”. Here in my case, the URL is “<http://192.168.0.175:8080>”. And finally click the “Open” button.



If all things go well, we can see the streamed video on the victim host machine. Here we can see the network URL on top of the VLC media player from where the video is streaming and thus the streaming video is successful.

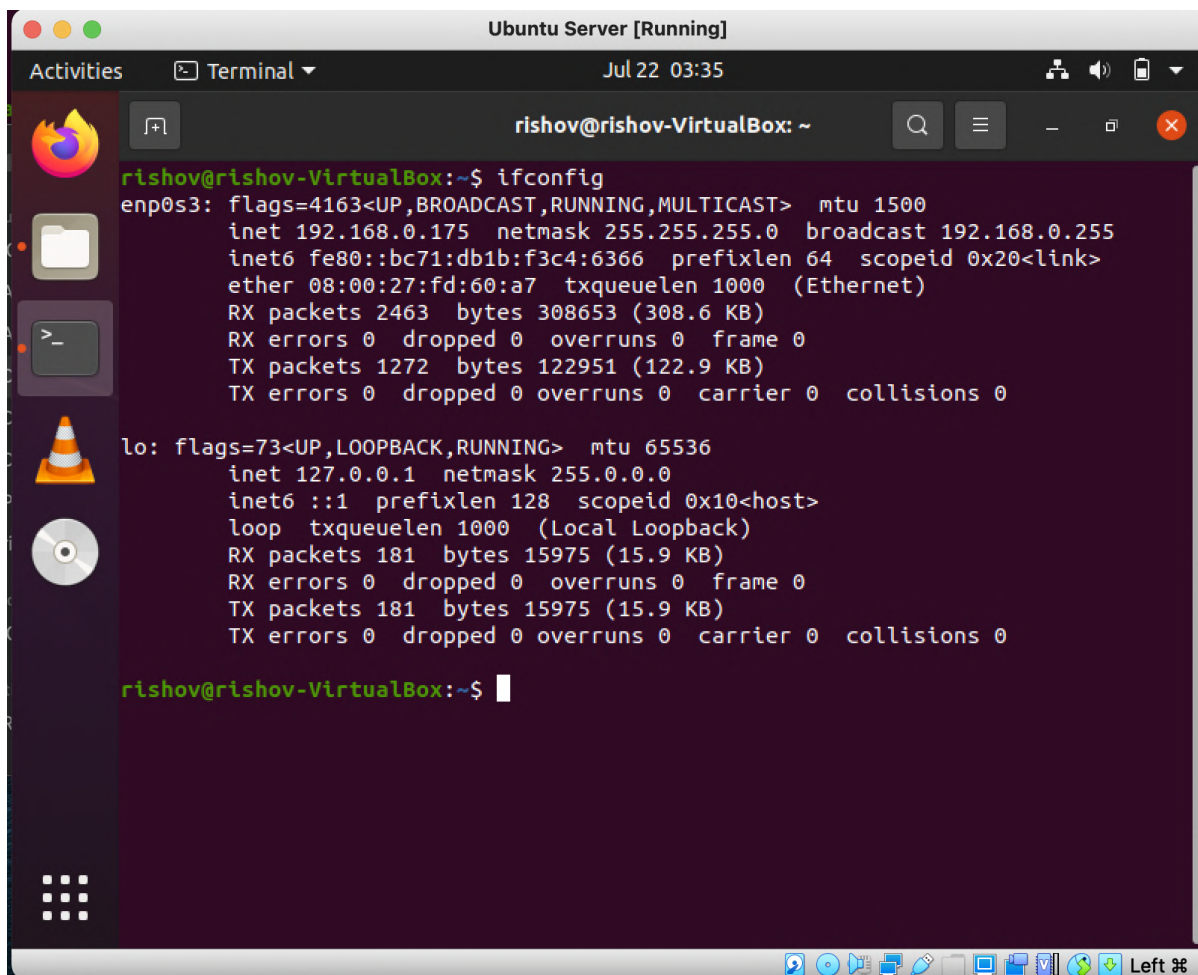


4.ARP Spoofing Man In The Middle Attack:

ARP spoofing attack will update the mac table of both the server and the victim. Both the server and the victim will send their packets to the attacker. Previously as the server was streaming video to the victim host machine, it was sending packets to the victim machine. So the attacker needs to turn on IP forwarding. IP forwarding ensures that when the server and the victim send packets to the attacker, the attacker will redirect the packets to their originally intended destination. ARP Spoofing helps to capture the packets between the two parties, i.e., to sniff them in order to extract meaningful information from the packets.

Step-1:

Check the Server machine's IP address using ifconfig command on the terminal and from that interface "enp0s3" holds the inet address which is the IPv4 address of the machine and ether holds the mac address.

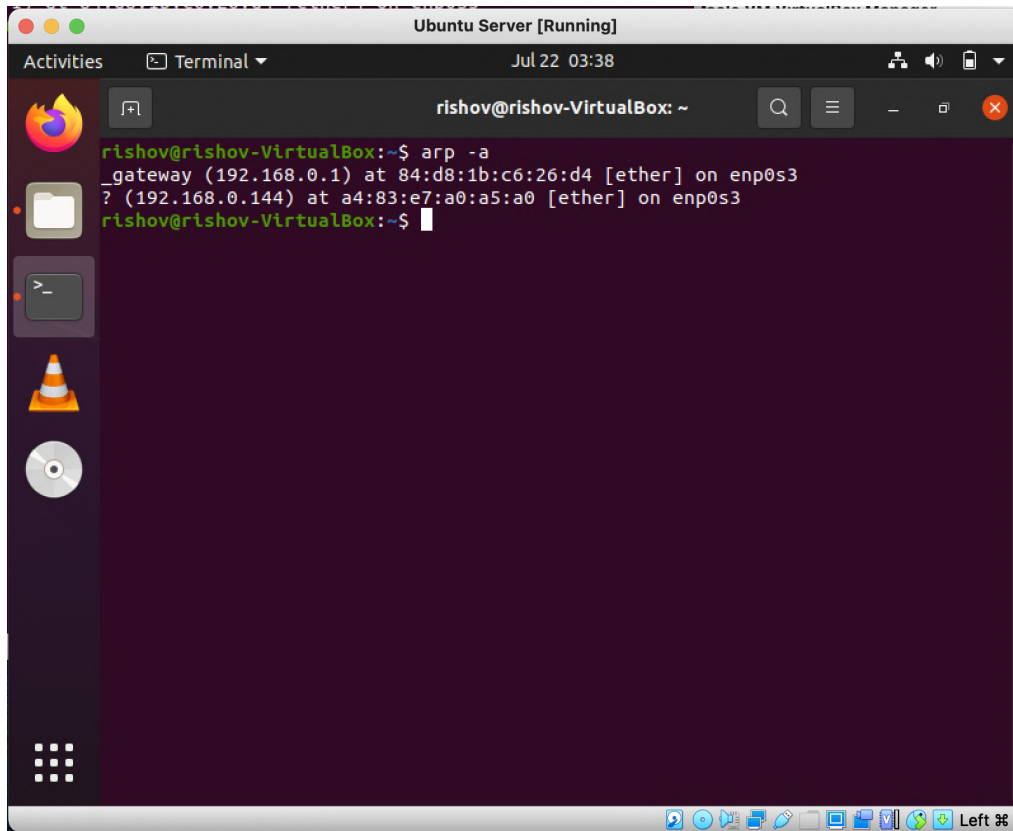


```
Ubuntu Server [Running]
Activities Terminal Jul 22 03:35
rishov@rishov-VirtualBox: ~
rishov@rishov-VirtualBox:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.175 netmask 255.255.255.0 broadcast 192.168.0.255
    inet6 fe80::bc71:db1b:f3c4:6366 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:fd:60:a7 txqueuelen 1000 (Ethernet)
    RX packets 2463 bytes 308653 (308.6 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1272 bytes 122951 (122.9 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 15975 (15.9 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 181 bytes 15975 (15.9 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

rishov@rishov-VirtualBox:~$
```

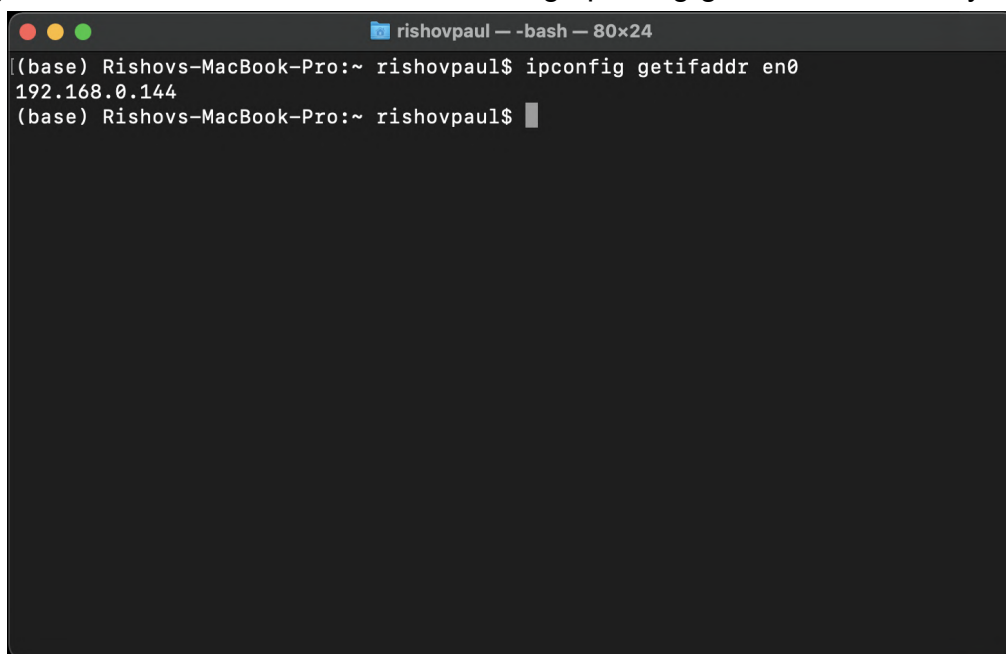
Check the arp table before running the man in the middle attack using the command “arp -a”.



A screenshot of a terminal window titled "Ubuntu Server [Running]". The window shows the command "arp -a" being executed. The output lists two entries in the ARP table: the gateway at 192.168.0.1 with MAC address 84:d8:1b:c6:26:d4 on interface enp0s3, and a host at 192.168.0.144 with MAC address a4:83:e7:a0:a5:a0 on the same interface. The terminal prompt is "rishov@rishov-VirtualBox: ~\$".

```
rishov@rishov-VirtualBox:~$ arp -a
_gateway (192.168.0.1) at 84:d8:1b:c6:26:d4 [ether] on enp0s3
? (192.168.0.144) at a4:83:e7:a0:a5:a0 [ether] on enp0s3
rishov@rishov-VirtualBox:~$
```

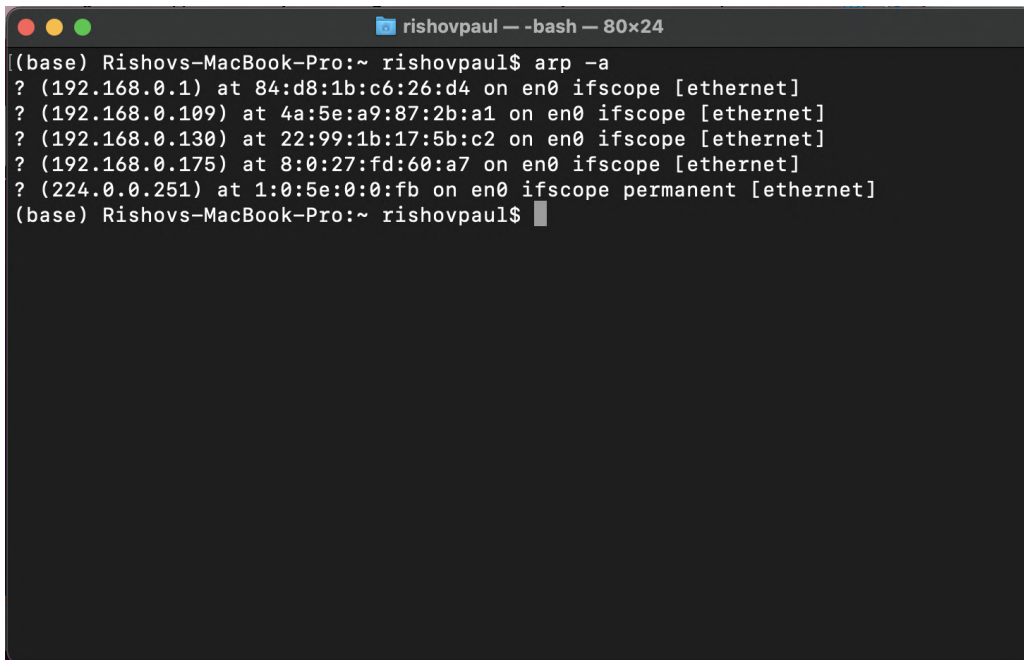
Similarly check the IP address of the host using “ipconfig getifaddr en0” in my case.



A screenshot of a terminal window titled "rishovpaul - -bash - 80x24". The window shows the command "ipconfig getifaddr en0" being executed. The output is the IP address "192.168.0.144". The terminal prompt is "(base) Rishovs-MacBook-Pro:~ rishovpaul\$".

```
(base) Rishovs-MacBook-Pro:~ rishovpaul$ ipconfig getifaddr en0
192.168.0.144
(base) Rishovs-MacBook-Pro:~ rishovpaul$
```

Check the arp table like the server machine.

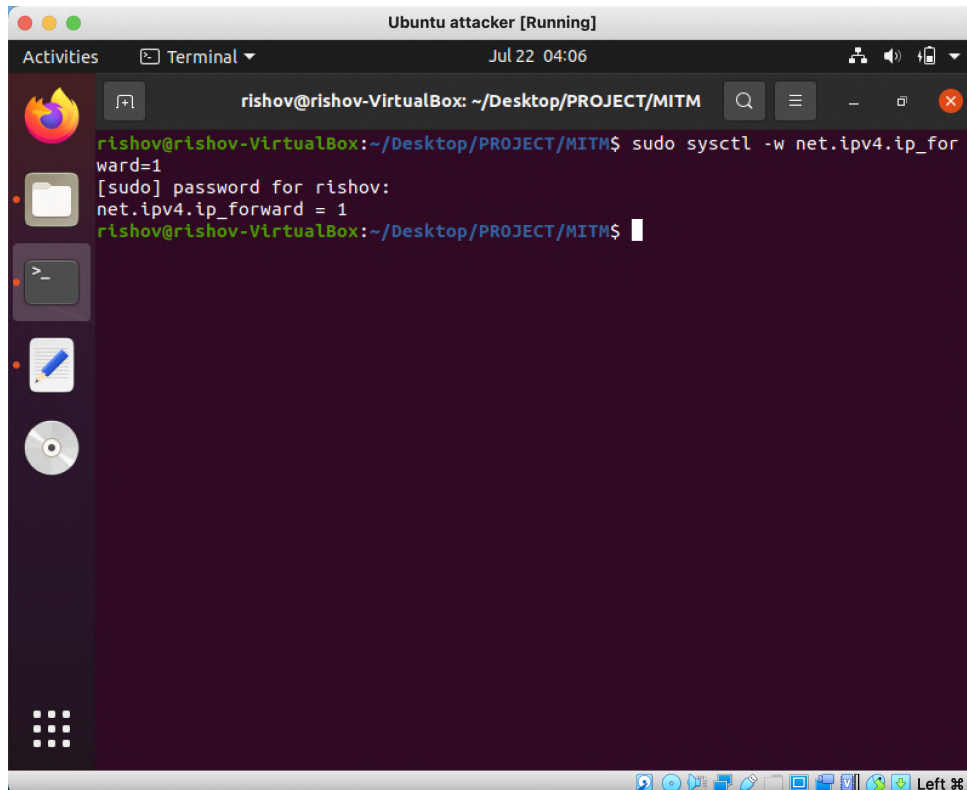


```
(base) Rishovs-MacBook-Pro:~ rishovpaul$ arp -a
? (192.168.0.1) at 84:d8:1b:c6:26:d4 on en0 ifscope [ethernet]
? (192.168.0.109) at 4a:5e:a9:87:2b:a1 on en0 ifscope [ethernet]
? (192.168.0.130) at 22:99:1b:17:5b:c2 on en0 ifscope [ethernet]
? (192.168.0.175) at 8:0:27:fd:60:a7 on en0 ifscope [ethernet]
? (224.0.0.251) at 1:0:5e:0:0:fb on en0 ifscope permanent [ethernet]
(base) Rishovs-MacBook-Pro:~ rishovpaul$
```

Step-2:

We need to turn on the IP forwarding on the attacker machine. On a linux based system, the command is as follows:

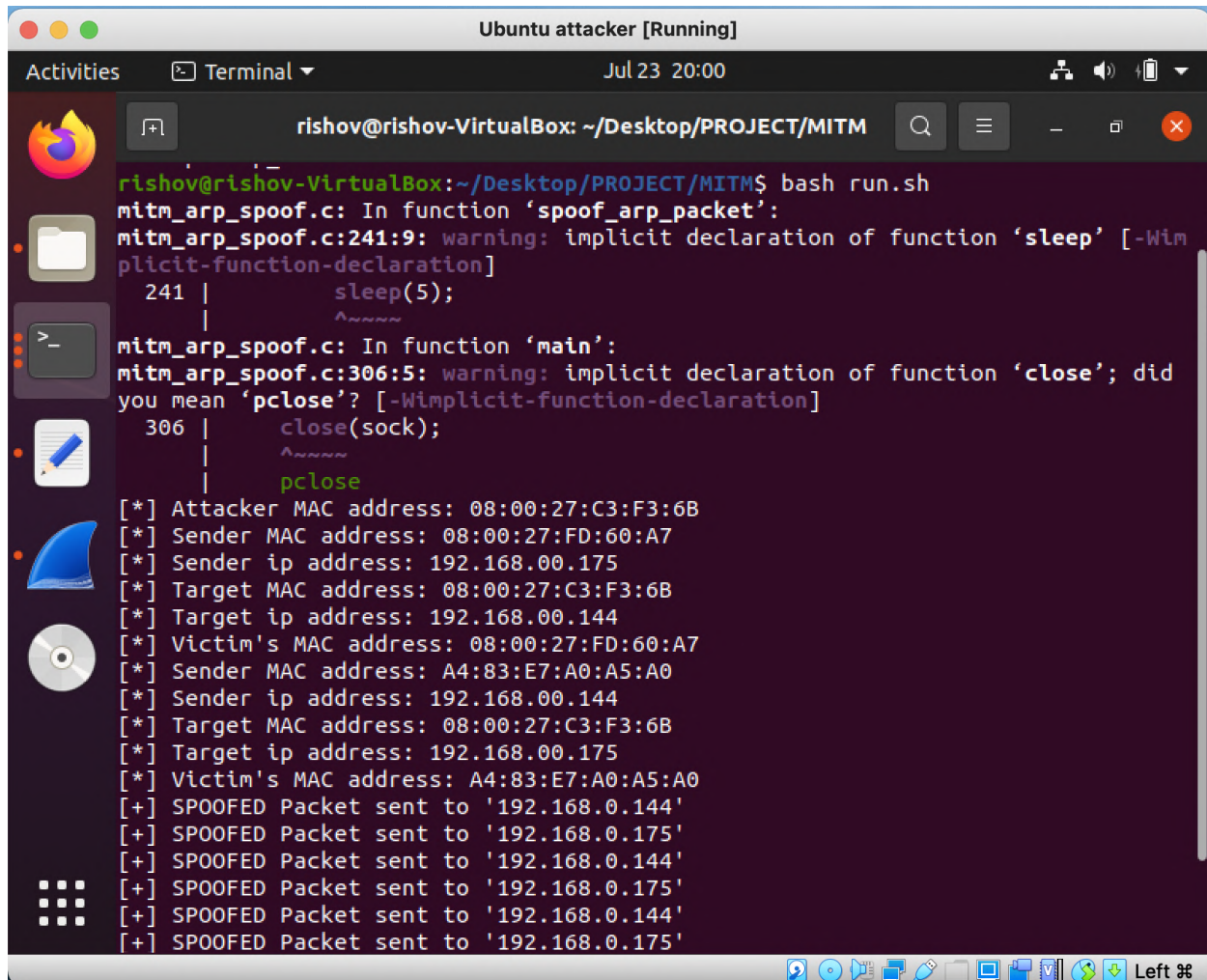
sudo sysctl -w net.ipv4.ip_forward=1



```
Ubuntu attacker [Running]
Activities Terminal Jul 22 04:06
rishov@rishov-VirtualBox: ~/Desktop/PROJECT/MITM
rishov@rishov-VirtualBox:~/Desktop/PROJECT/MITM$ sudo sysctl -w net.ipv4.ip_for
ward=1
[sudo] password for rishov:
net.ipv4.ip_forward = 1
rishov@rishov-VirtualBox:~/Desktop/PROJECT/MITM$
```

Step-3:

Then man in the middle attack is performed on the attacker machine using the run.sh file situated in the MITM folder.



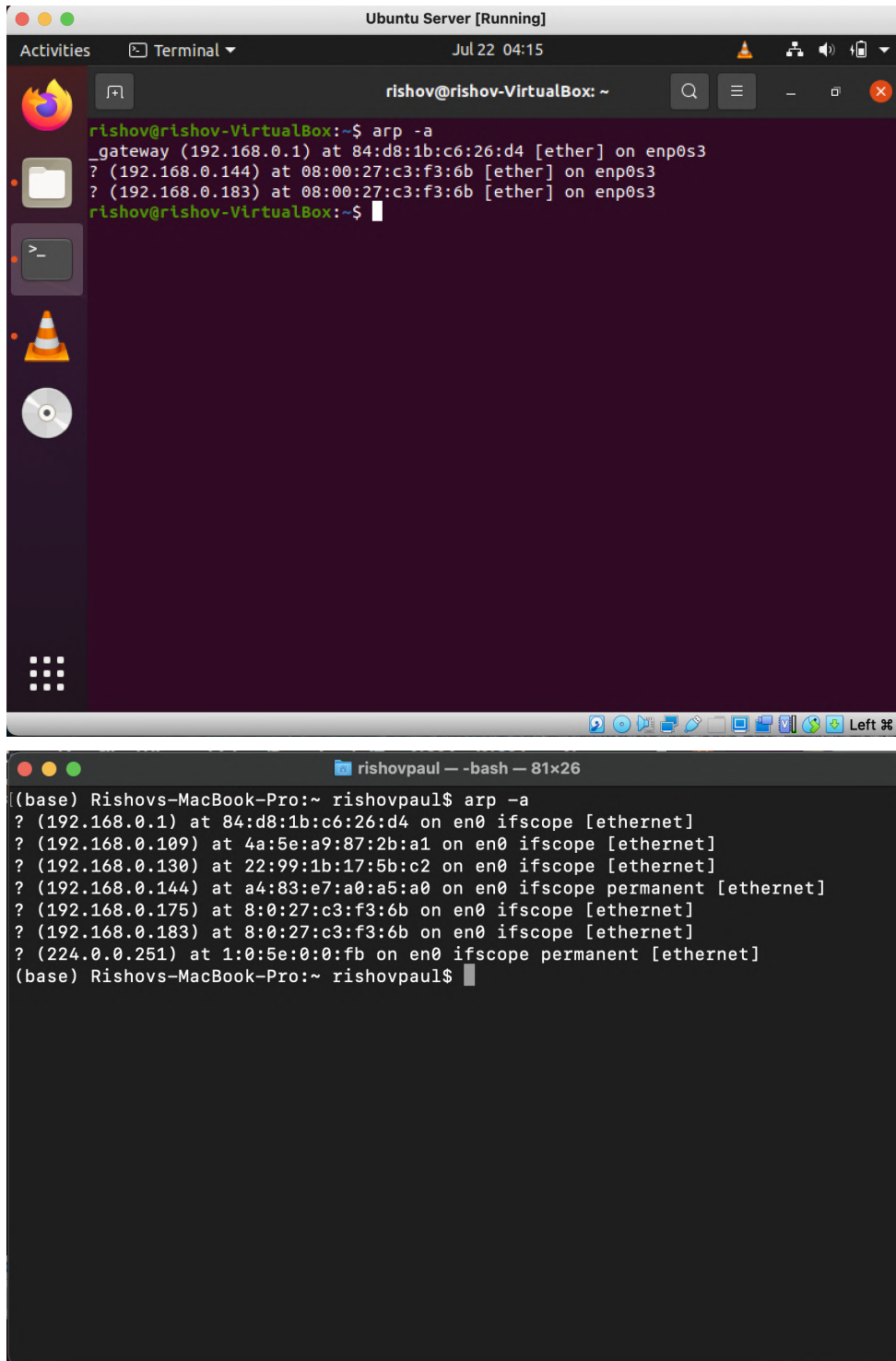
```

rishov@rishov-VirtualBox: ~/Desktop/PROJECT/MITM
rishov@rishov-VirtualBox:~/Desktop/PROJECT/MITM$ bash run.sh
mitm_arp_spoof.c: In function 'spoof_arp_packet':
mitm_arp_spoof.c:241:9: warning: implicit declaration of function 'sleep' [-Wimplicit-function-declaration]
  241 |         sleep(5);
      |         ^~~~~~
mitm_arp_spoof.c: In function 'main':
mitm_arp_spoof.c:306:5: warning: implicit declaration of function 'close'; did you mean 'pclose'? [-Wimplicit-function-declaration]
  306 |         close(sock);
      |         ^~~~~~
      |         pclose
[*] Attacker MAC address: 08:00:27:C3:F3:6B
[*] Sender MAC address: 08:00:27:FD:60:A7
[*] Sender ip address: 192.168.00.175
[*] Target MAC address: 08:00:27:C3:F3:6B
[*] Target ip address: 192.168.00.144
[*] Victim's MAC address: 08:00:27:FD:60:A7
[*] Sender MAC address: A4:83:E7:A0:A5:A0
[*] Sender ip address: 192.168.00.144
[*] Target MAC address: 08:00:27:C3:F3:6B
[*] Target ip address: 192.168.00.175
[*] Victim's MAC address: A4:83:E7:A0:A5:A0
[+] SPOOFED Packet sent to '192.168.0.144'
[+] SPOOFED Packet sent to '192.168.0.175'
[+] SPOOFED Packet sent to '192.168.0.144'
[+] SPOOFED Packet sent to '192.168.0.175'
[+] SPOOFED Packet sent to '192.168.0.144'
[+] SPOOFED Packet sent to '192.168.0.175'
```

Here we can see the details of the attacker, sender and victim host's mac addresses and IP addresses and see that spoofing has occurred. To see the result of the spoofing we need to see the arp table of both the server and victim host after the attack.

Step-4:

Here we see the arp table after the attack and notice that the corresponding mac address has been replaced by the attacker mac address in the arp table. So the man in the middle attack was successful.



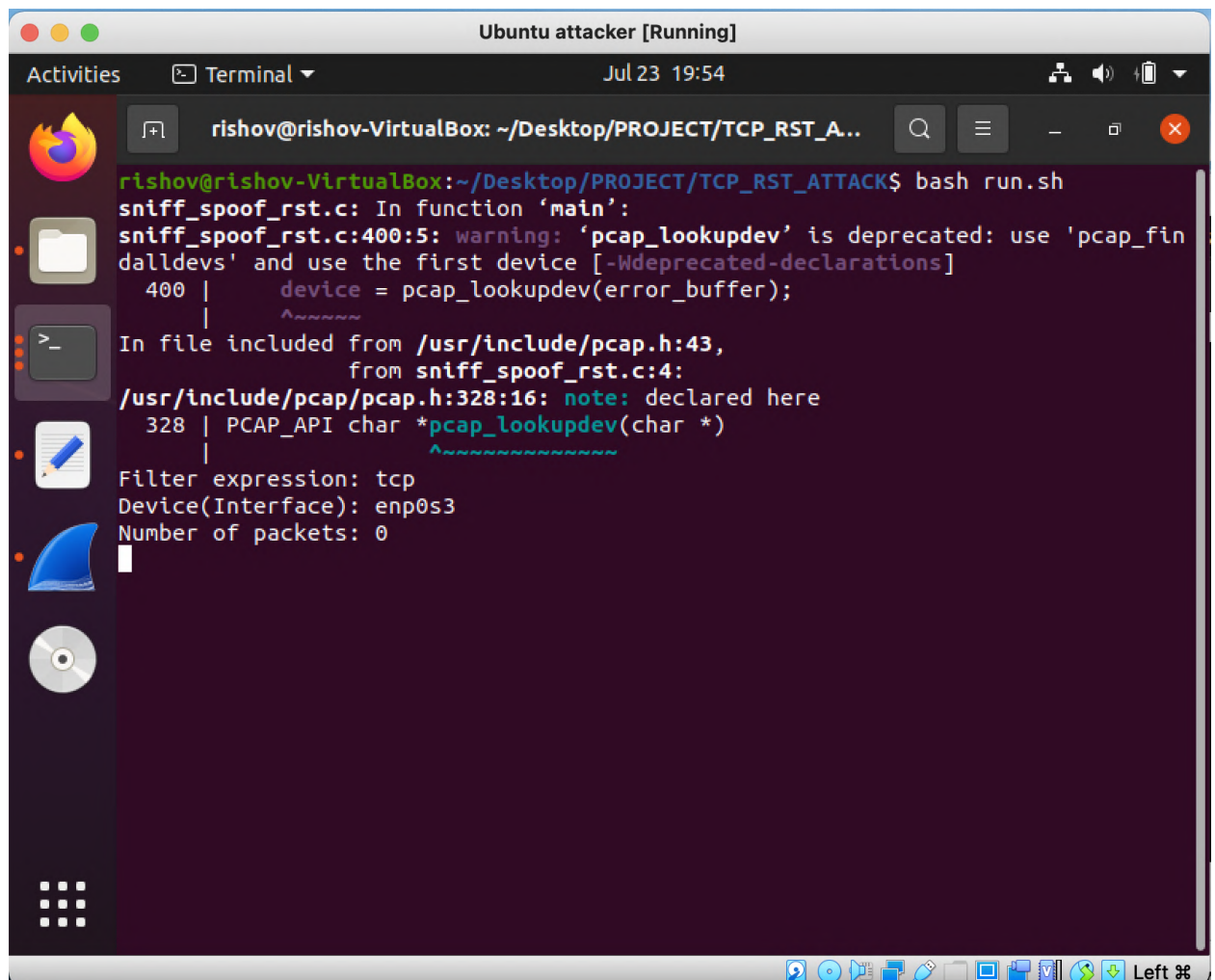
```
rishov@rishov-VirtualBox:~$ arp -a
_gateway (192.168.0.1) at 84:d8:1b:c6:26:d4 [ether] on enp0s3
? (192.168.0.144) at 08:00:27:c3:f3:6b [ether] on enp0s3
? (192.168.0.183) at 08:00:27:c3:f3:6b [ether] on enp0s3
rishov@rishov-VirtualBox:~$
```

```
(base) Rishovs-MacBook-Pro:~ rishovpaul$ arp -a
? (192.168.0.1) at 84:d8:1b:c6:26:d4 on en0 ifscope [ethernet]
? (192.168.0.109) at 4a:5e:a9:87:2b:a1 on en0 ifscope [ethernet]
? (192.168.0.130) at 22:99:1b:17:5b:c2 on en0 ifscope [ethernet]
? (192.168.0.144) at a4:83:e7:a0:a5:a0 on en0 ifscope permanent [ethernet]
? (192.168.0.175) at 8:0:27:c3:f3:6b on en0 ifscope [ethernet]
? (192.168.0.183) at 8:0:27:c3:f3:6b on en0 ifscope [ethernet]
? (224.0.0.251) at 1:0:5e:0:0:fb on en0 ifscope permanent [ethernet]
(base) Rishovs-MacBook-Pro:~ rishovpaul$
```

Fig: Poisoned Server and Victim arp table

5. Sniffing and RST Packet Spoofing:

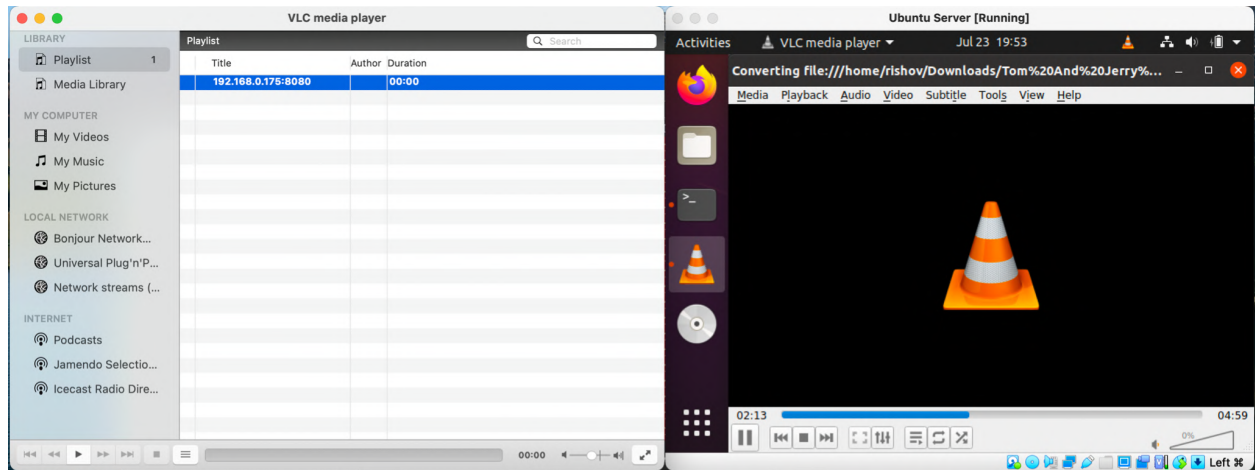
Now we are in the main phase of the attack. At this phase, the server is streaming uninterruptedly to the victim. So we need to achieve an interruption. So the sniffing_spoofing_rst.c file was run via run.sh file in the TCP Reset Attack folder in the project.



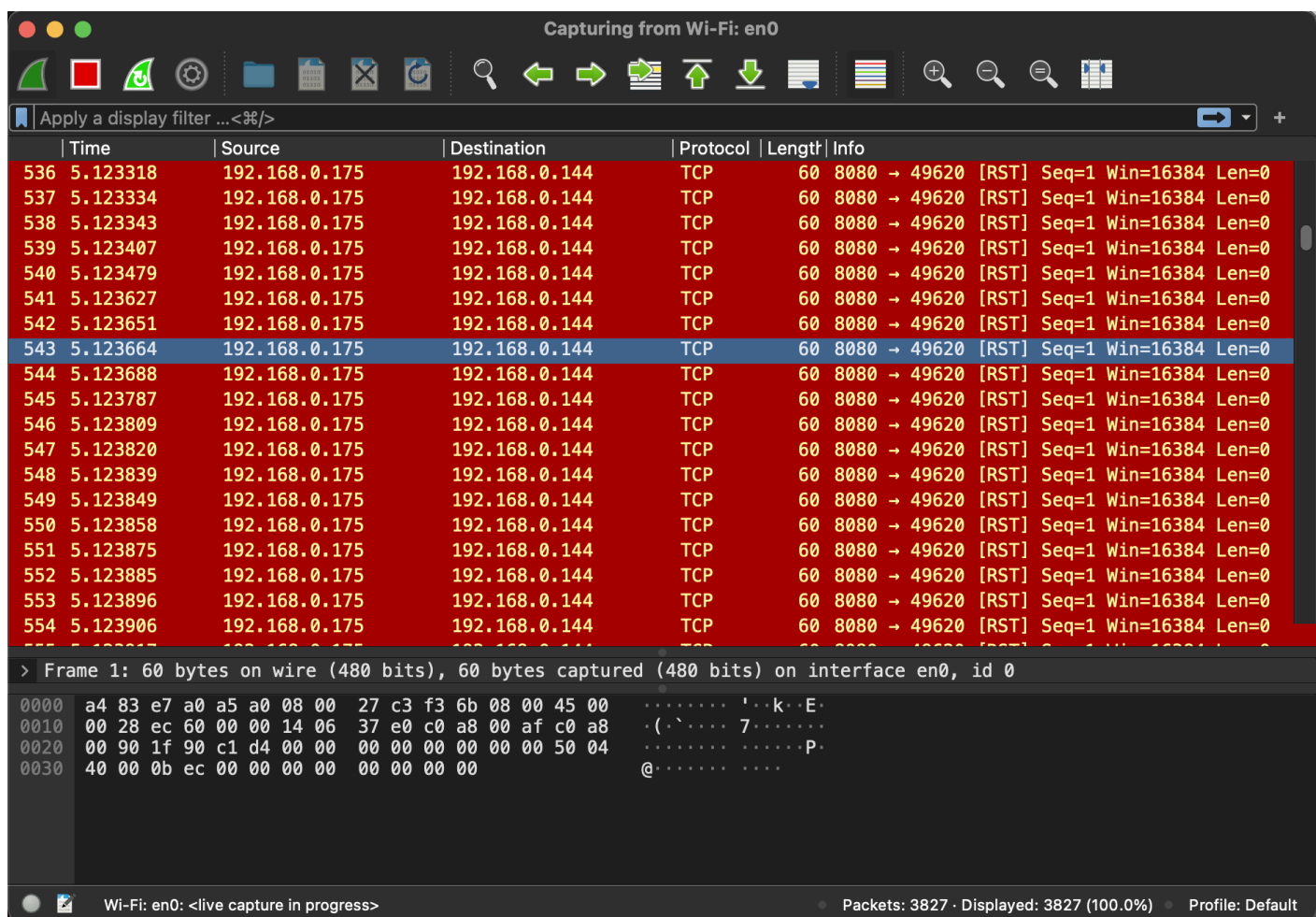
```
Ubuntu attacker [Running]
Activities Terminal Jul 23 19:54
rishov@rishov-VirtualBox: ~/Desktop/PROJECT/TCP_RST_A...
rishov@rishov-VirtualBox:~/Desktop/PROJECT/TCP_RST_ATTACK$ bash run.sh
sniff_spoof_rst.c: In function 'main':
sniff_spoof_rst.c:400:5: warning: 'pcap_lookupdev' is deprecated: use 'pcap_findalldevs' and use the first device [-Wdeprecated-declarations]
400 |     device = pcap_lookupdev(error_buffer);
    |
In file included from /usr/include/pcap.h:43,
    from sniff_spoof_rst.c:4:
/usr/include/pcap/pcap.h:328:16: note: declared here
328 | PCAP_API char *pcap_lookupdev(char *)
    |
Filter expression: tcp
Device(Interface): enp0s3
Number of packets: 0
```

Here the device “enp0s3” indicates the interface and filter expression indicates only tcp packets are captured. And the number of packets is 0 which indicates that the program will run and sniff and spoof packets until any error or ending condition is provided. Number of packets is provided in the pcap_loop() function of the code to infinitely capture packets.

After some time, it was seen that the stream had stopped on the victim host machine as every player has some buffer but the server was still running.



To verify this, I opened Wireshark and saw that RST packets were successfully sent to close the connection between the server and the victim machine.



Conclusion:

To stop a TCP connection, a RST packet is needed with a valid sequence number. I captured all packets in my network and sent an infinite number of packets with the same source and destination addresses (maintaining the Ethernet, IP and TCP headers) after manipulating their TCP RST flag and sequence number. I ensured that the sequence number is valid by setting it to the packet's acknowledgement number and sent the spoofed packet to the victim. Again, I sent infinite packets because video streaming is a fast and spontaneous connection and any packet can drop in transmission or the sequence number can increase in between of transmission. Thus, I ensured the termination of video streaming through TCP Reset Attack.