**TCP/IP Attack Lab**

Attacker Victim Observer

IP: 10.0.2.15 IP: 10.0.2.5 IP: 10.0.2.4

MAC: 08:00:27:0f:cd:b4 MAC: 08:00:27:ab:17:4f MAC: 08:00:27:87:d6:04

| | |

|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_| Fake machine

| LAN or Virtual Network | IP : 10.0.2.8

| Gateway 10.0.2.0 | MAC: aa:aa:aa:aa:aa:aa

|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_|

|

Internet tool

**Task 1: SYN Flooding Attack**

This is a form of Denial of service attack in which the attacker machine sends a lot of spoofed SYN request packets to the victim machine to which the victim will try allocating its resources to those packets and thus, any future legitimate request will be discarded.

Issuing the packets using Wireshark

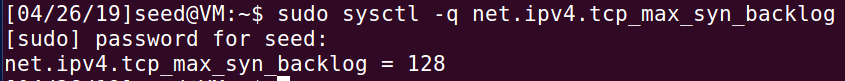
Using the netwox 76 the attacker can create a spoofed TCP SYN request packet and send it to the victim. The syntax of the command is,

netwox 76 --dst-ip <victim IP address> --dst-port <victim port number> --spoofip “<packet type>”

Protection from Ubuntu against SYN Flooding attacks

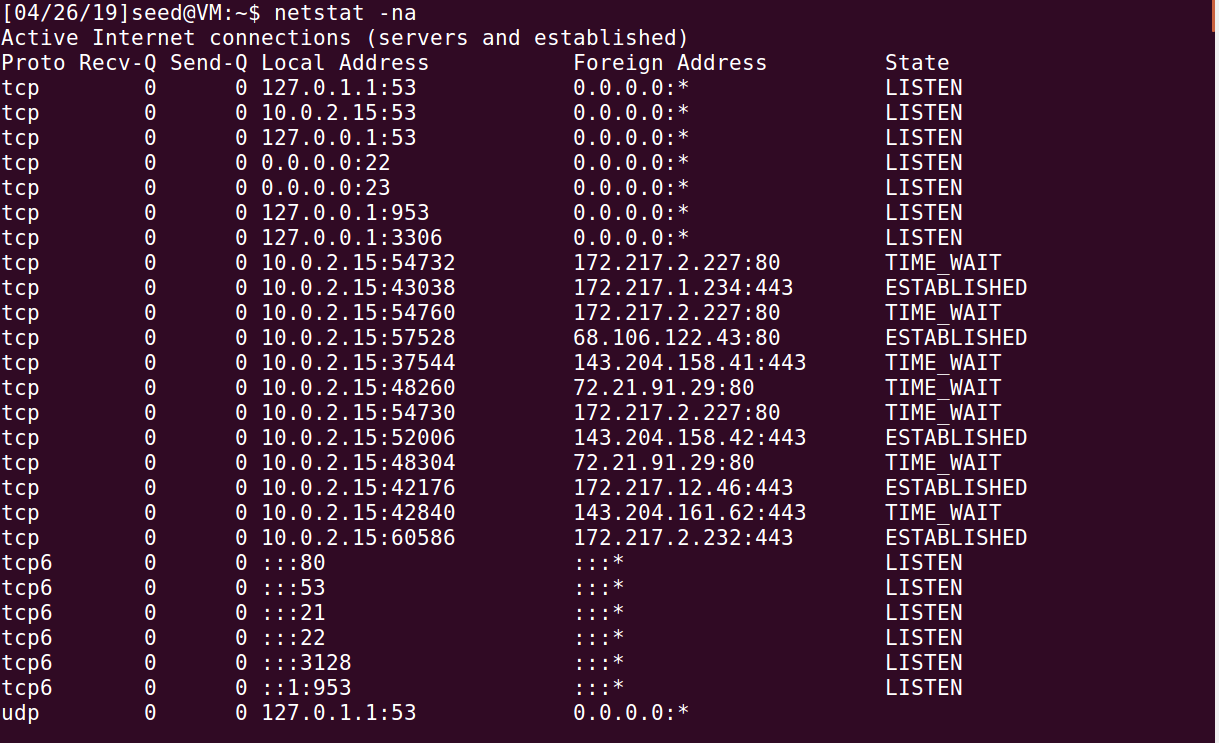
Ubuntu provides SYN cookies that can avoid dropping SYN request packets when the SYN queue is full. Instead they make the queue as if it is enlarged and send back SYN-ACK response and discards further SYN requests. If it receives any ACK response it will update the queue using the TCP sequence number. By default, the SYN cookie is disabled. But it can be changed by modifying the content in etc/sysctl.conf.

Size of the Queue



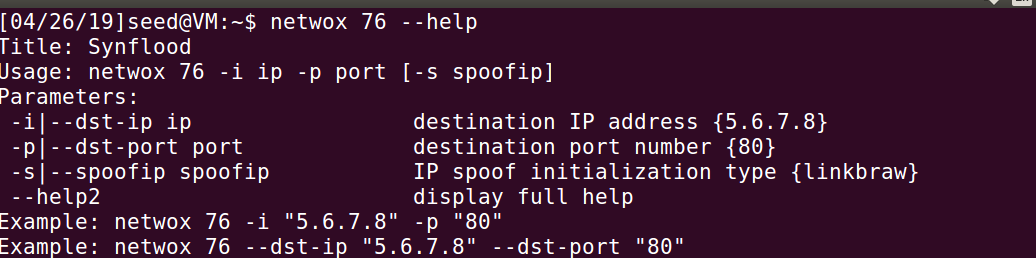
**Observation:** By default, the maximum size of syn\_backlog queue is 128. It can be increased or decreased by adding the entry into the /etc/sysctl.conf file.

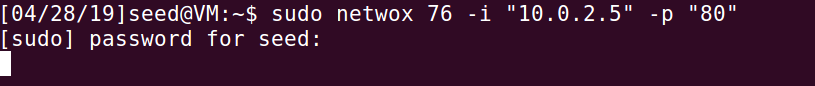
Using “netsat -na” to see the “SYN\_RECV” typed connection on the victim machine



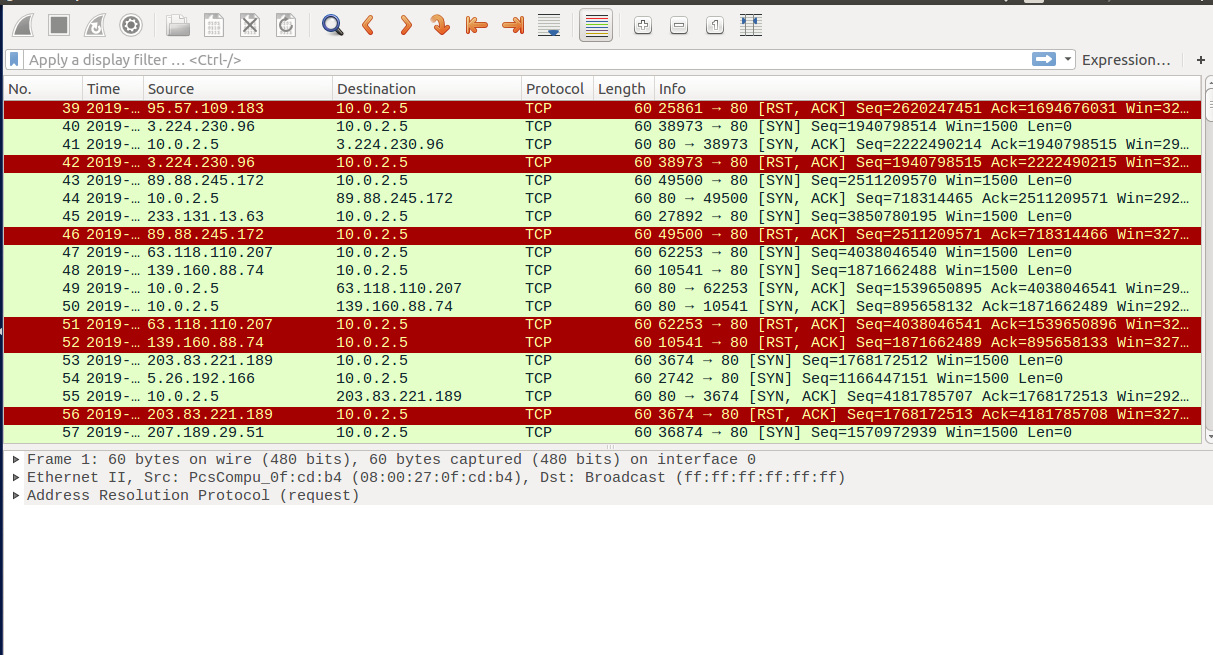
**Observation:** Initially in the victim machine all ports are open and listening for any incoming packets.

Using netwox -76 help command:



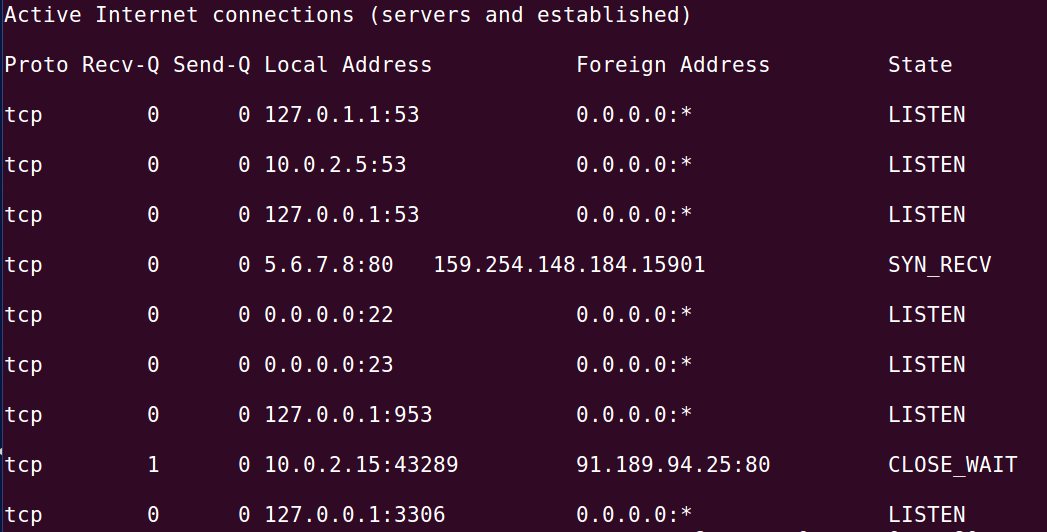


**Observation:** The attacker used netwox 76 to send spoofed SYN request to the victim under port 80. The packet type must be raw or else the attacker would send an ARP request for the MAC address of the spoofed source IPs (best).



**Observation:** The observer can see the spoofed TCP SYN packets with different IP address being sent to the victim.

Using netstat -na we can observe the following:



**Observation:** As the victim’s syn queue gets filled up it sends out the SYN-ACK packet and waits for the ACK which can be seen as SYN\_RECV which indicates that the connection is in half opened state.

As the attacker floods the syn queue of the victim, the victim machine becomes slow to response and it tries to process all the packets with overload. However, the attacker cannot provide denial of service on the victim.

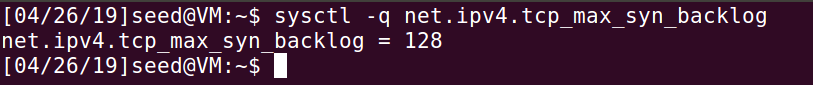
The attack can be more effective if the following parameters are changed in victim machine.

* By reducing the syn queue size of the victim so that it soon runs out of spaces to place the incoming SYN packets.

net.ipv4.tcp\_max\_syn\_backlog = 64

The above statement should be placed in the /etc/sysctl.conf file.

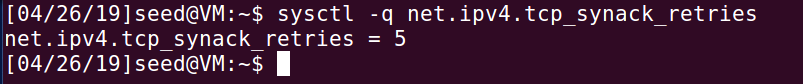
The default size is



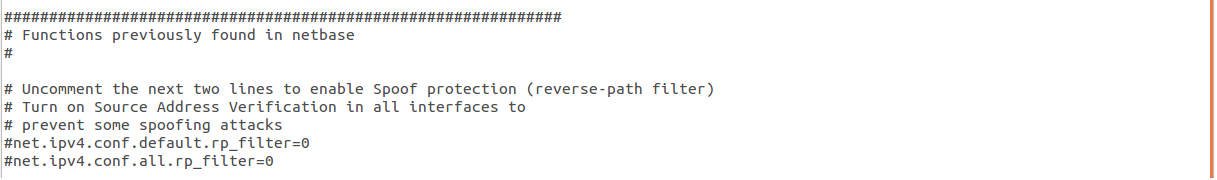
* By increasing the value of synack\_retries the closing time of SYN\_RECV is increased so the packet stay longer in the queue.

net.ipv4.tcp\_synack\_retries = 10.

the default value is

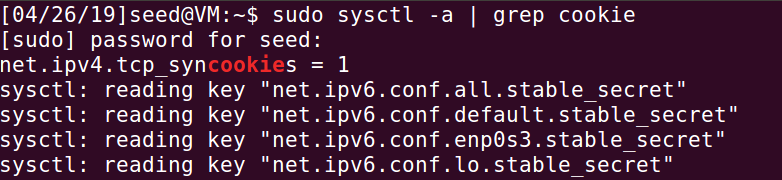


* By disabling the IP spoofing filter which prevents any spoofed packets.



SYN Cookie Countermeasure:

Display the SYN Cookie Flag



Turn off the SYN Cookie



Turn on the SYN Cookie

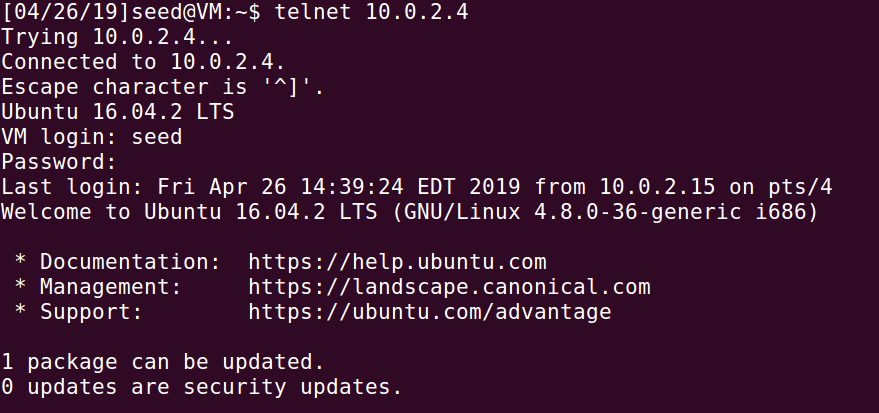


**Task 2: TCP RST Attacks on telnet and ssh Connections**

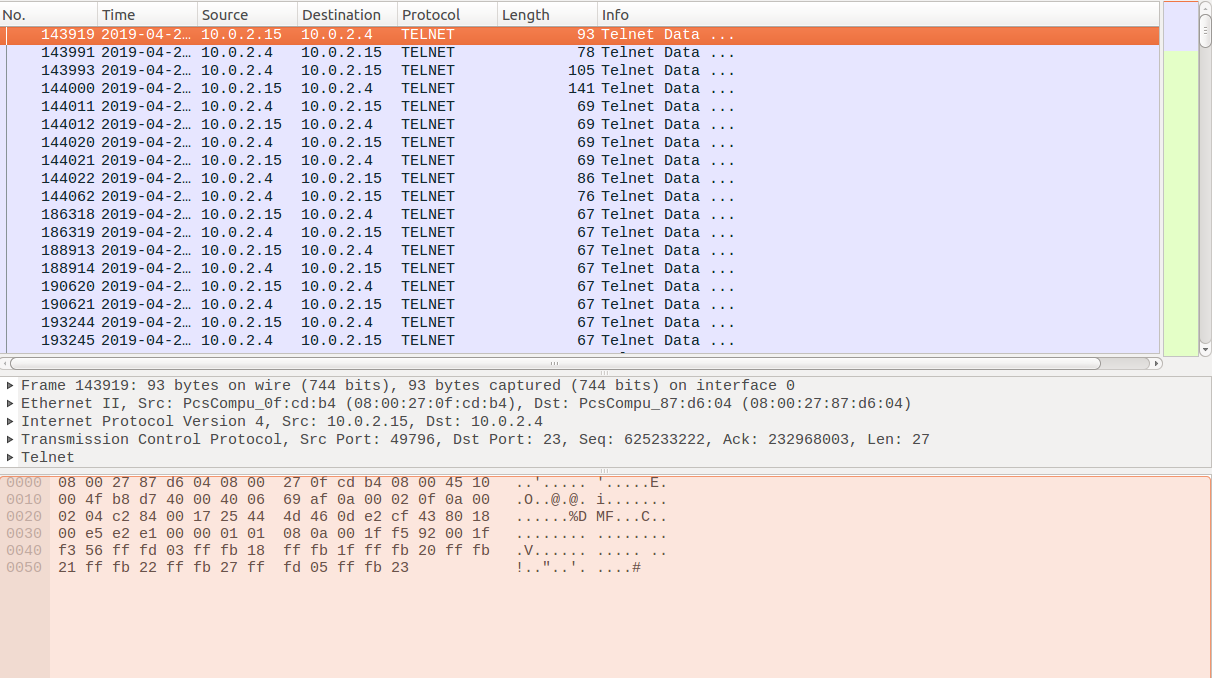
TCP reset is form of terminating a TCP connection between two computers. When one computer wants to terminate a TCP connection it sends out a TCP packet with the RST flag set to 1 to the other computer. However, such TCP reset packets can be forged by an attacker machine in the same LAN. The forged packet’s header must contain the IP address of one of the connected machines.

**Telnet**

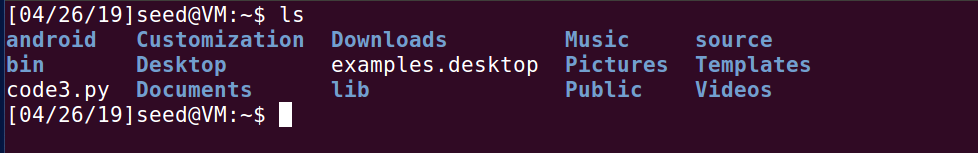
TCP RST packets can be generated and sent by the attacker through the netwox 78 tool. The syntax of the command is Netwox 78 --device <”device”> --filter <”filter parameter”> --spoofip <”packet type”> --ips <”destination ip”>.



**Observation:** When there is no TCP RST attack the victim can make a telnet connection to the observer.



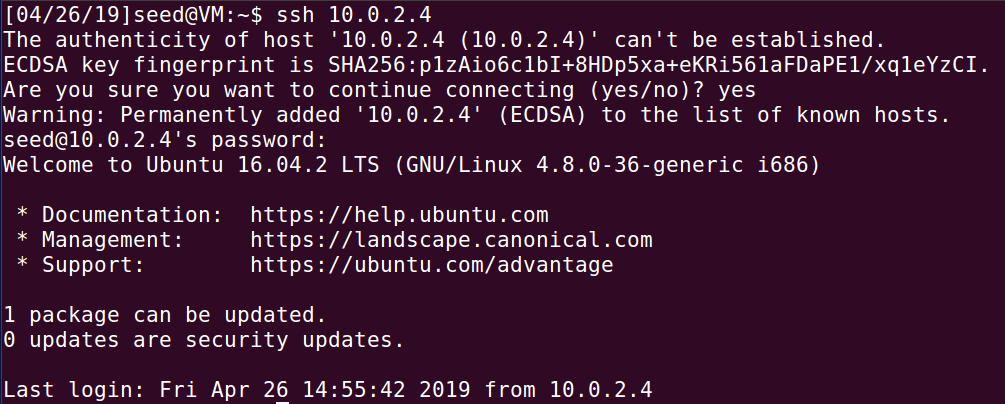
**Observation:** The attacker watches the packet being sent between the victim and observer.



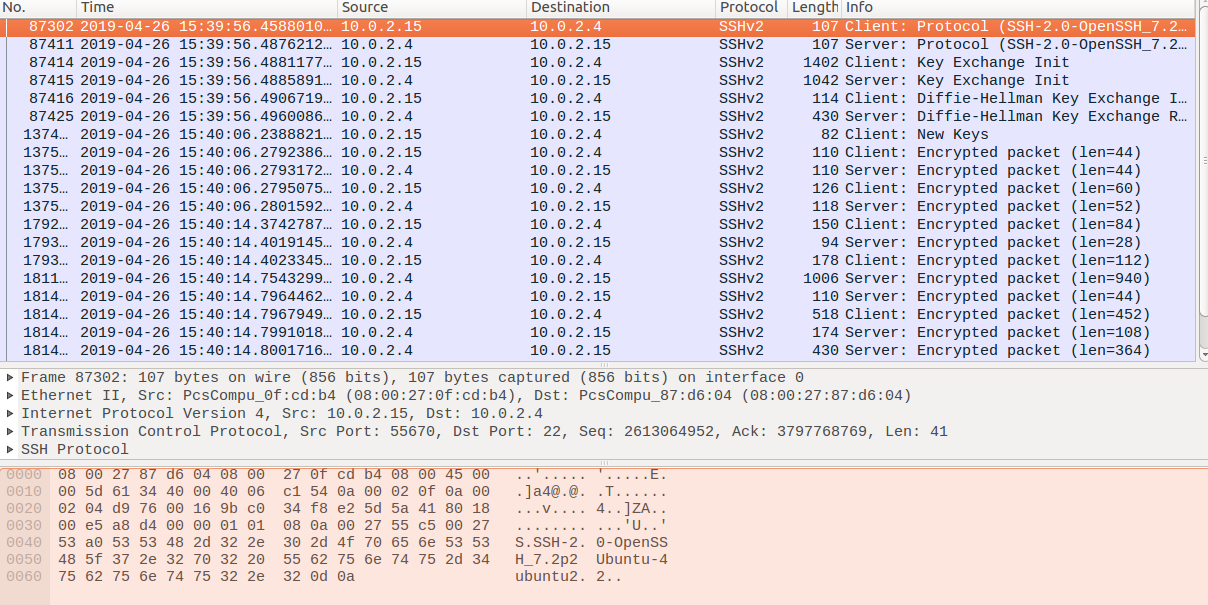
**Observation:** After the connection is established the victim can view the files of the observer.

**SSH Connection:**

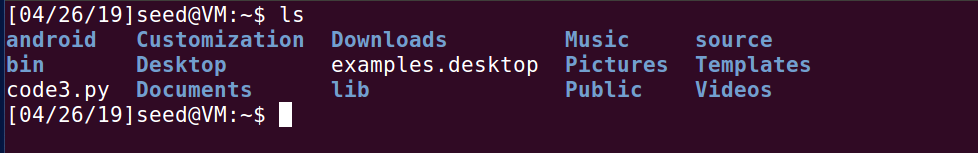
TCP RST packets can be generated and sent by the attacker through the netwox 78 tool. The syntax of the command is Netwox 78 --device <”device”> --filter <”filter parameter”> --spoofip <”packet type”> --ips <”destination ip”>



**Observation:** Before the attack the victim can make ssh connection with observer.



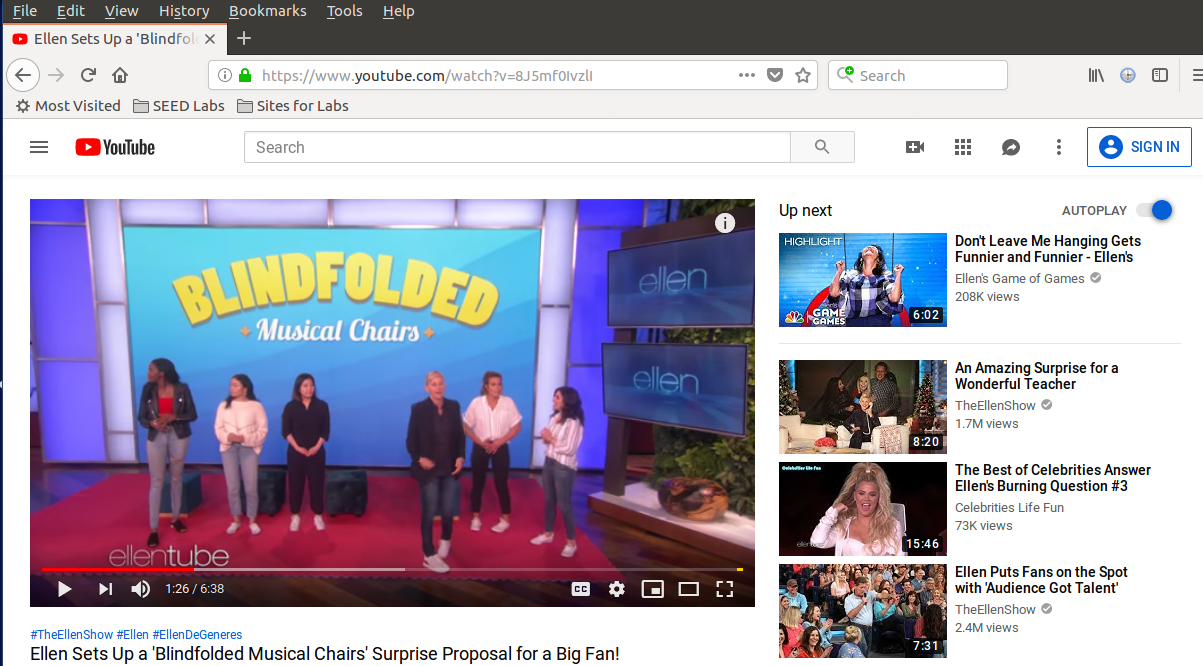
**Observation:** The attacker watches the ssh packets sent between victim and the observer.



**Observation:** After the connection is established the victim can view the files of the observer.

**Task 3: TCP RST Attacks on Video Streaming Applications**

TCP RST attack can be done on any TCP packet. One such attack is to block the online streaming video by sending a forged TCP reset packet to the victim machine. TCP RST packets can be generated and sent by the attacker through the netwox 78 tool. The syntax of the command is Netwox 78 --device <”device”> --filter <”filter parameter”> --spoofip <”packet type”> --ips <”destination ip”>.



**Observation:** The victim machine watches streaming video online. As there is no reset packet the video will keep on streaming.

**Within the same machine:**



**Observation:** The TCP RST packet is sent by the victim machine. The destination address is the 10.0.2.15 through which the packets are sent and received from NAT.



**Observation:** After the connection is terminated through forged TCP RST the video plays as long it was streamed before the attack.



**Observation:** Once the stream ends the video stops as if the entire video session has ended.

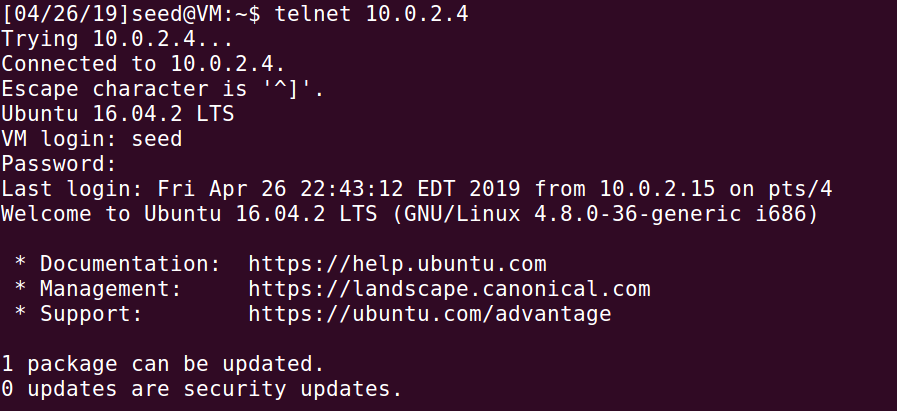
**With two virtual machines:**



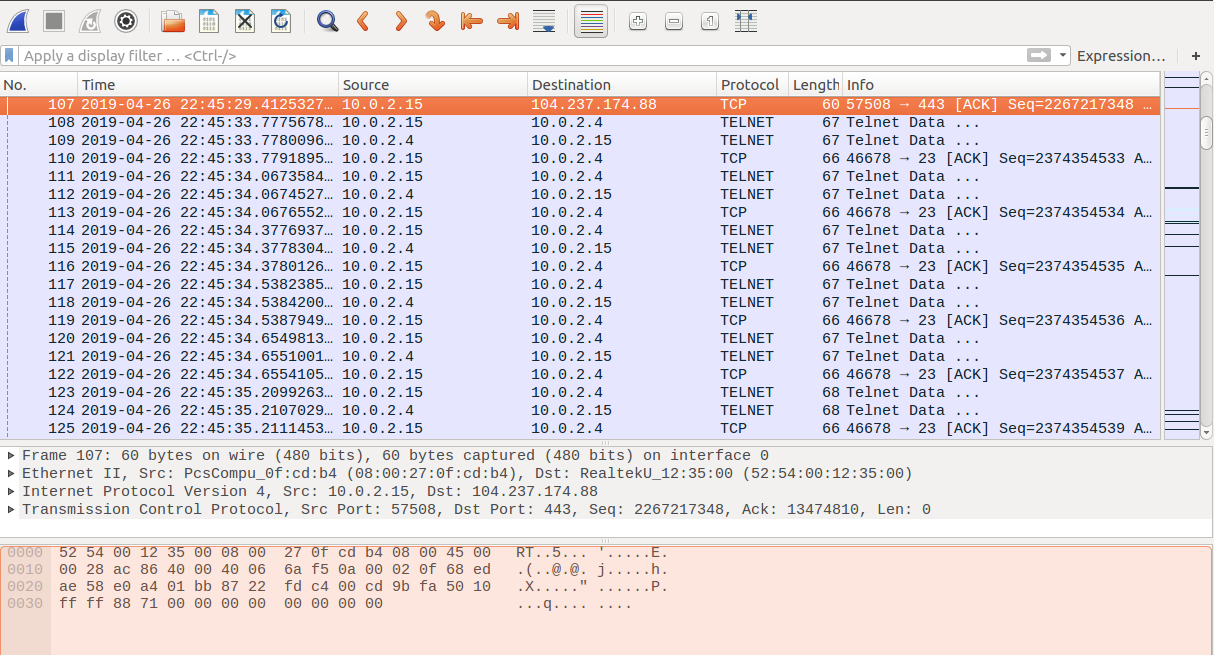
**Observation:** When another machine tries to send the forged TCP packet to the victim the reset packet will not delivered as the packets cannot be captured in a NAT network. This is because the promisc mode in NAT is denied by default.

**Task 4: TCP Session Hijacking**

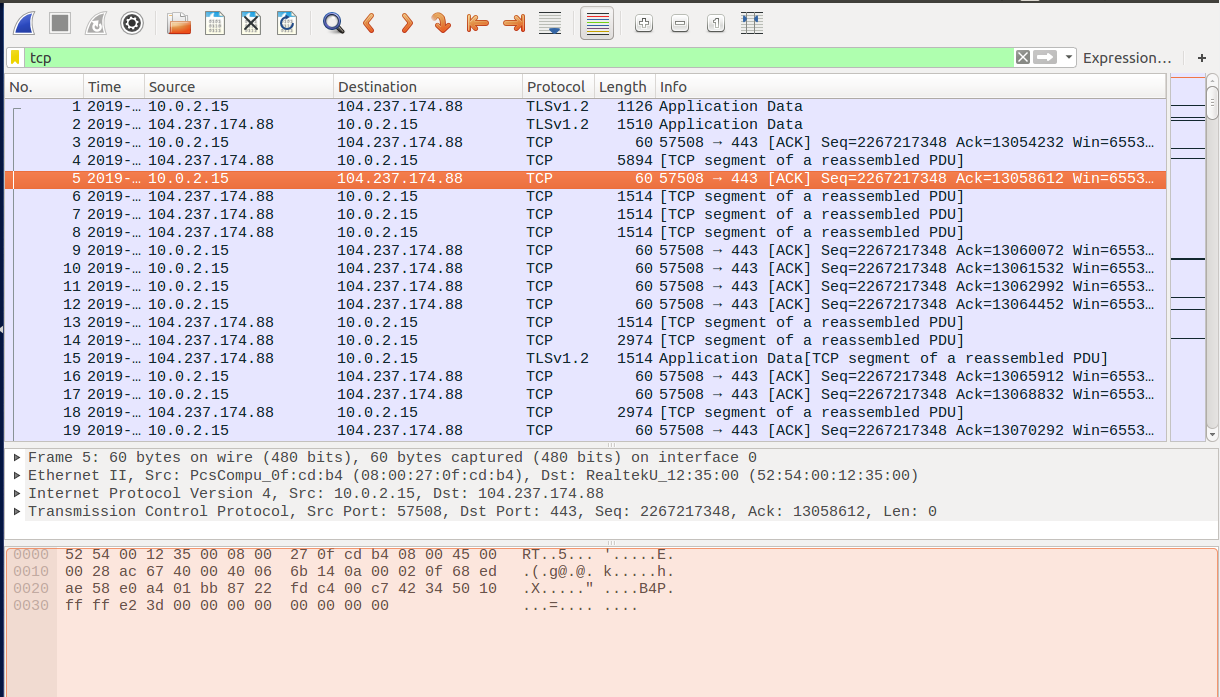
* TCP session hijacking is a process in which an attacker can intercept a TCP session between two machines. Since the authentication check is performed only during session initialization the attacker can perform the attack after some duration. The attacker gets the current value of the absolute sequence and acknowledgement number of the TCP session and forges a TCP packet with the next sequence and acknowledgement number and sends it to one of the two machines.
* The attacker uses the netwox 40 tool to send a fake packet on the network. The syntax of the command is, netwox 40 --ipv4-tos <DSCP value> --ipv4-id <the next id number> --ipv4-<flag> --ipv4-<frag offset> <bit>--ipv4-ttl <value> --ipv4—protocol <value> --ipv4-src <source IP > --ipv4-dst <destination IP> --tcp-src <port no.> --tcp-dst <port no.> --tcp-seqnum <next seq num> --tcp-acknum <next ack num> --tcp-<flag> --tcp-window <window size> --tcp-data <data> --spoofip <”type of packet”> --ipv4-totlen <length> --ipv4-checksum <value>.



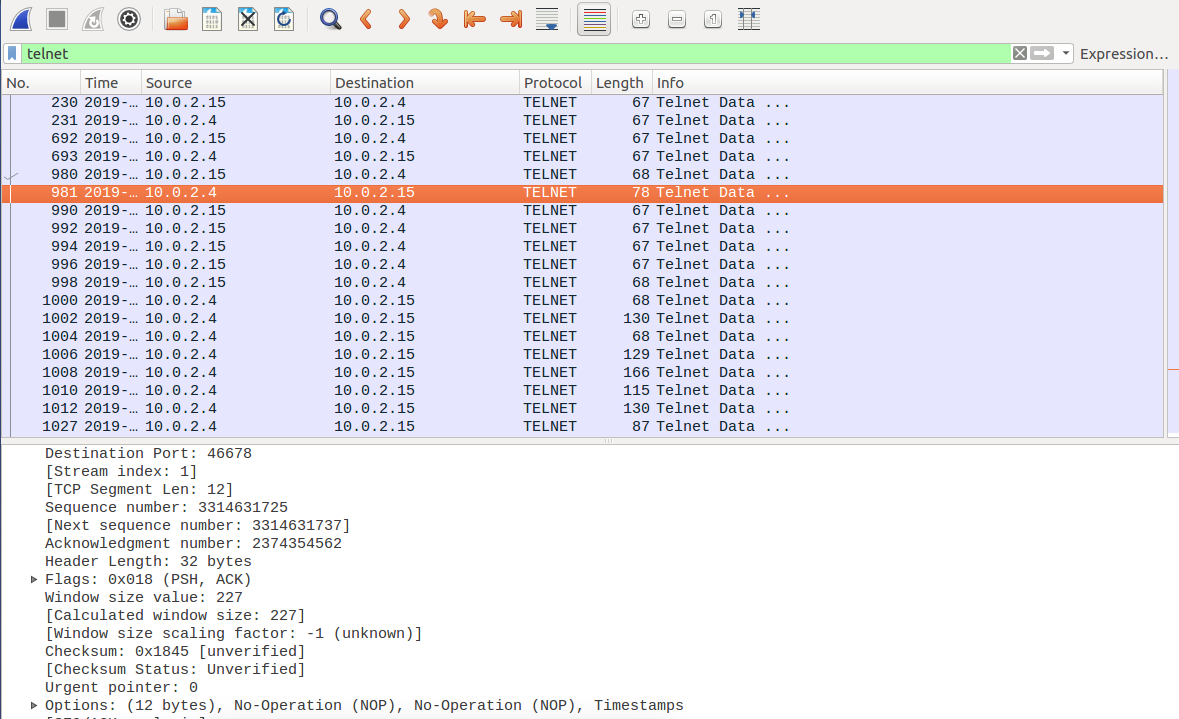
**Observation:** The victim 10.0.2.4 makes a telnet connection to the observer 10.0.2.5.



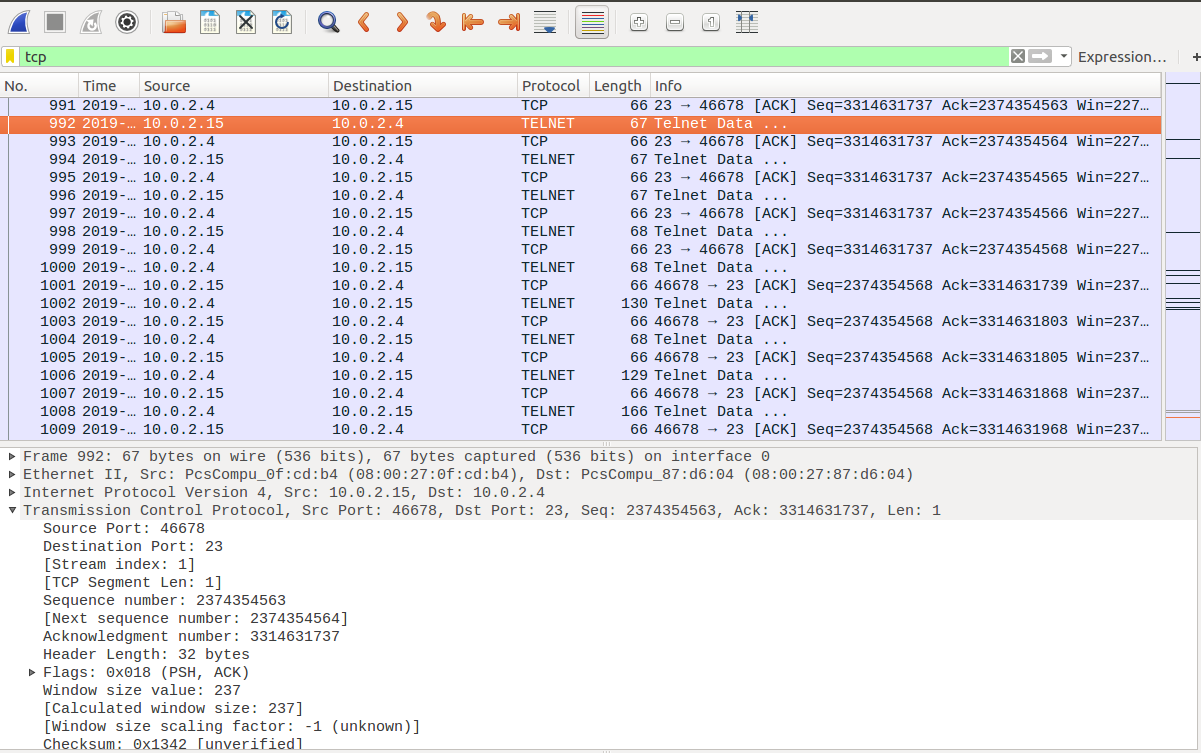
**Observation:** The victim uses the available port 57508 and provides an initial sequence number 2267217384 to the observer.



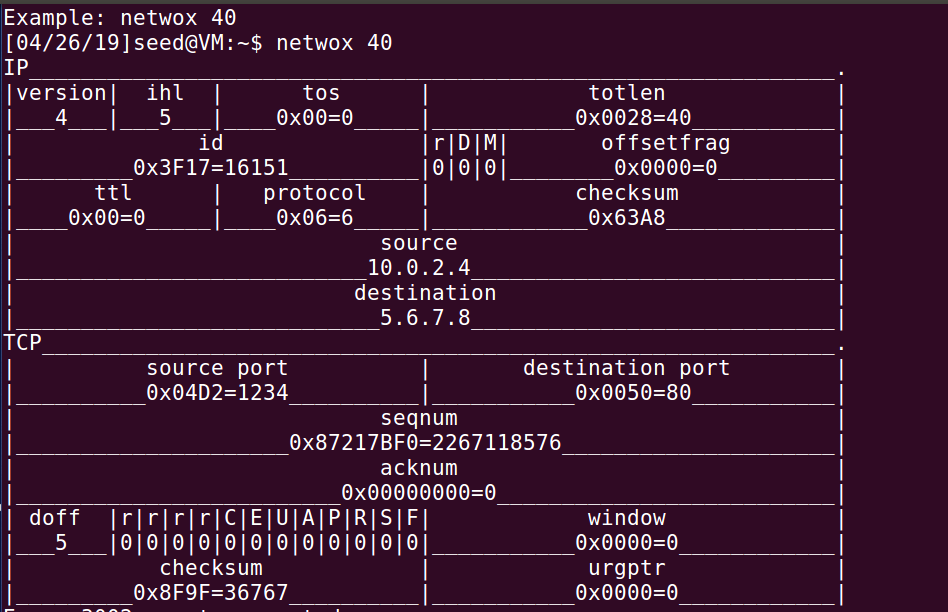
**Observation:** The observer acknowledges the victim’s TCP packet by sending the ack packet with the sequence number : 2267217348 and acknowledgement number as 13058612.



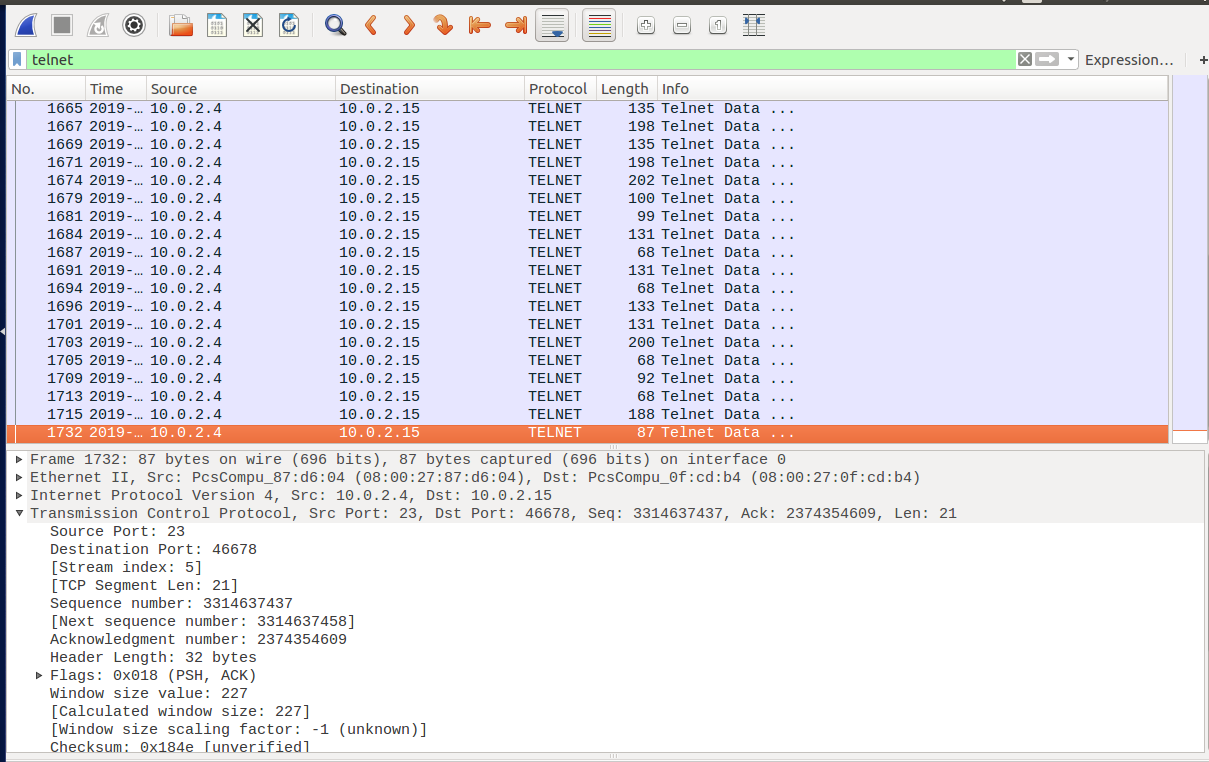
**Observation:** The last sequence number used by the victim is 3314631725, the next sequence number 33146354562 and acknowledgement number 2374354562.



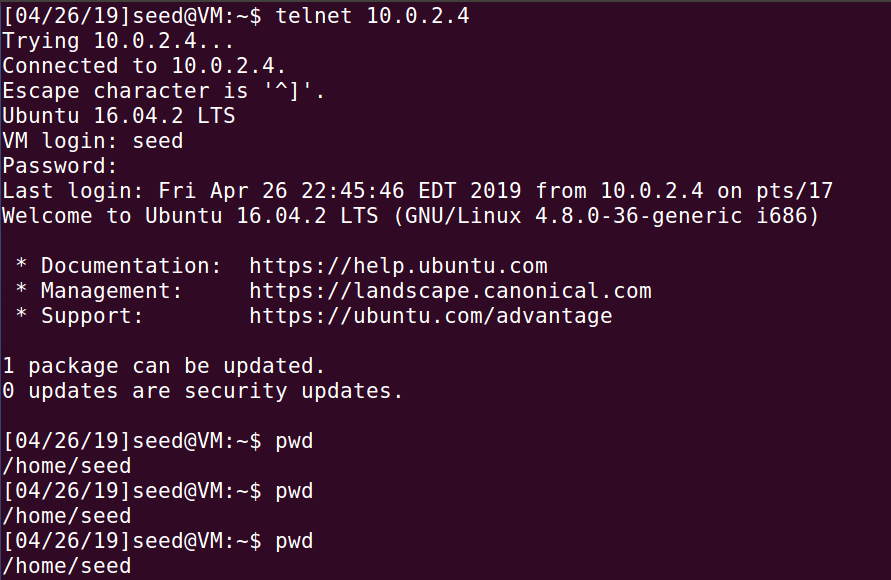
**Observation:** The victim acknowledges with the acknowledgement number: 2374354563.



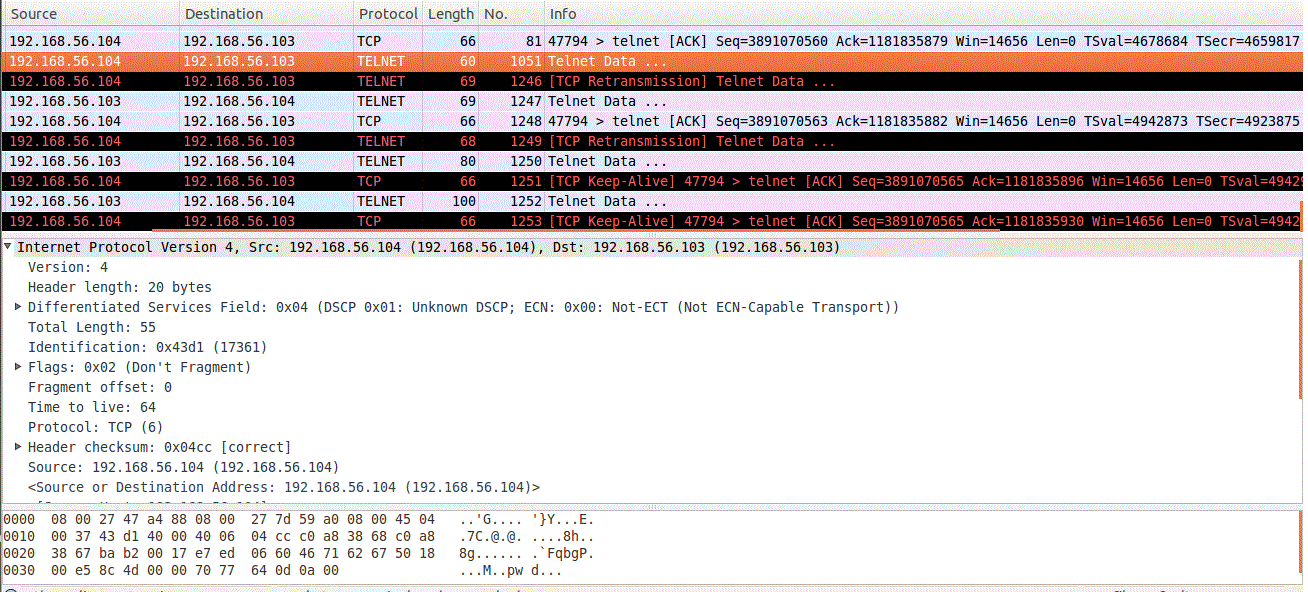
**Observation:** The attacker has successfully forged and sent a telnet packet to the observer with the next identification number: 16151.



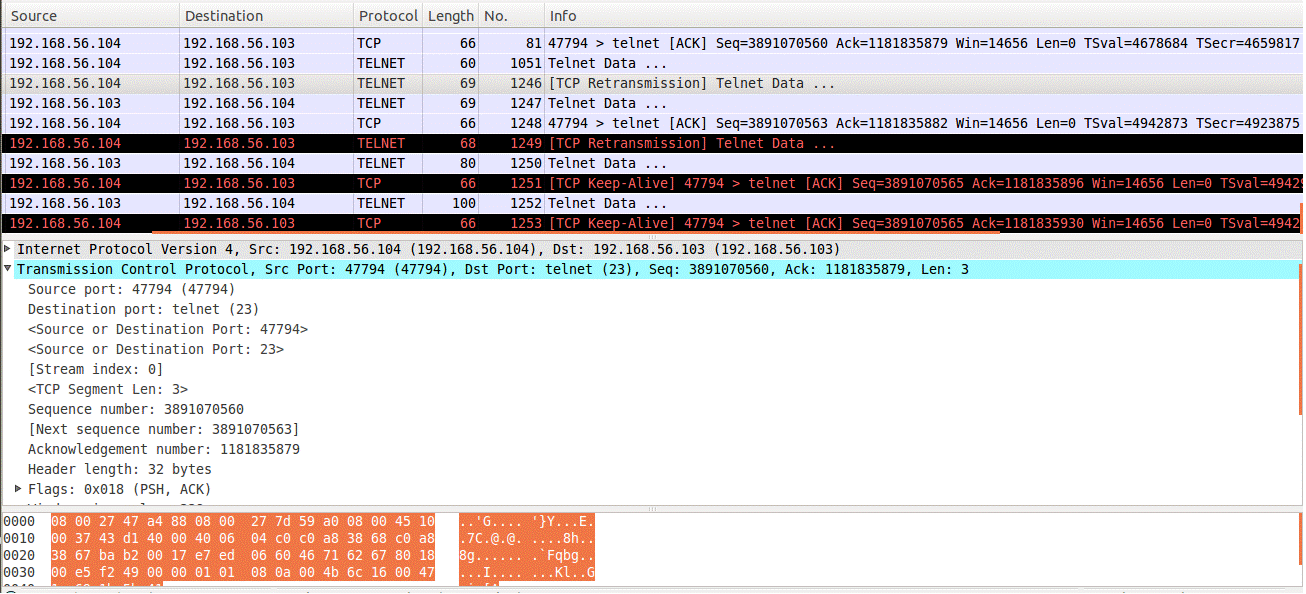
**Observation:** The forged packet can be seen in the victim’s wireshark (packet no.87)



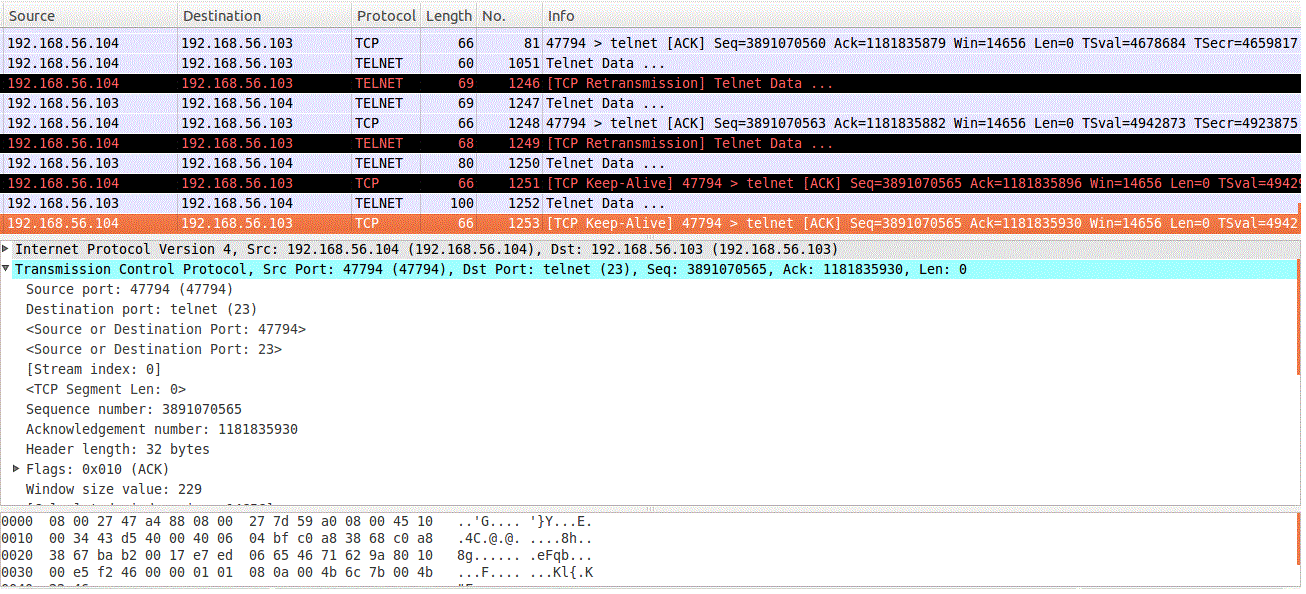
**Observation:** Unknown of the attack the victim sends out a telnet packet to get the current working directory of the observer.



**Observation:** The victim sends out a telnet packet with the ID number: 17361.



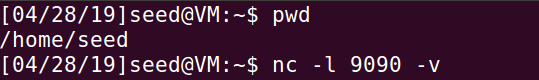
**Observation:** The victim uses the sequence number: 3891070560 acknowledgement number: 1181835879 which was used by the attacker, so the packet is taken as a retransmission.

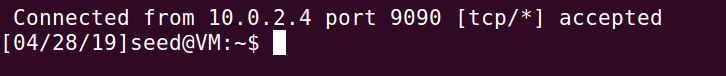


**Observation:** The victim acknowledges with the sequence number: 3891070563 and acknowledgement number: 1181835930. This can again be used by the attacker to forge further packets.

**Task 5: Creating Reverse Shell using TCP Session Hijacking**

A TCP session hijacking attack is a hijacking of a TCP session connection established by both parties to the communication, impersonating one of the parties to communicate further with the other party. Usually, some network services establish an application layer identity authentication after a TCP session. After the client authenticates , the client can request resources from the server through a TCP session connection. There is no need to authenticate again during the period. The TCP session hijacking provides an attacker with a technical way to bypass the application layer identity authentication, so it is favored by higher level attackers.







**Observation:** Using netcat to listen to the victim’s connection and thus, the victim’s connection is accepted.





**Observation:** Running the reverse shell