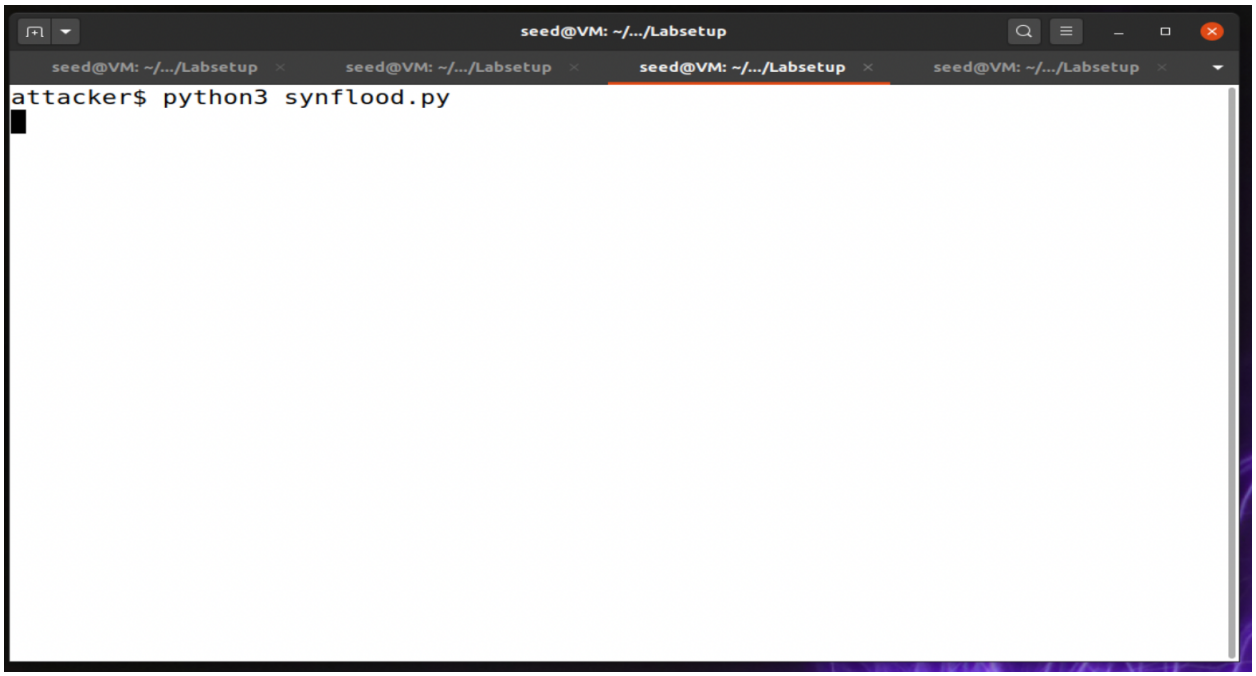
Tcp Attacks Lab
February 18, 2022
Will Sherrer
Dr. Long Cheng

Task 1.1 - TCP flooding python

I finished the python code and ran it, after a few minutes I was still able to successfully telnet into the victim server from one of the other user servers.




```
user1# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
4713b478f85a 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.
Last login: Sat Feb  5 19:43:20 UTC 2022 from user1-10.9.0.6.net-10.9.0.0 on pts
/2
seed@4713b478f85a:~$
```

Retransmission issue:

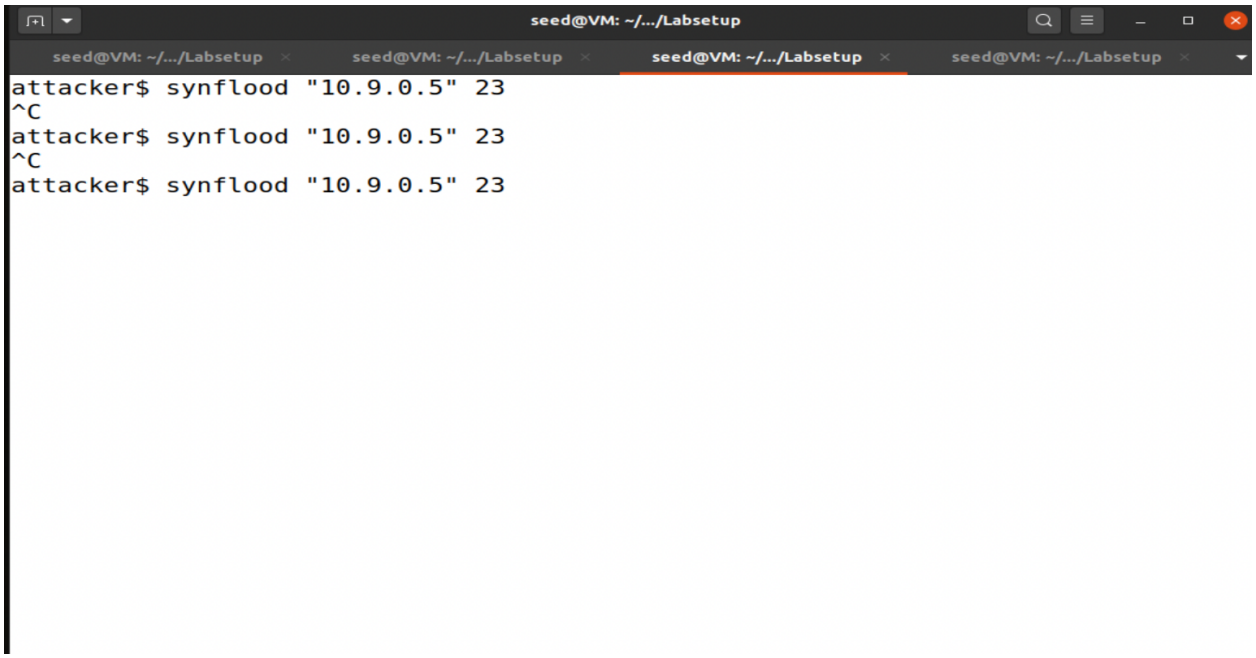
After running the program in 16 parallel instances I was able to successfully make the attack.

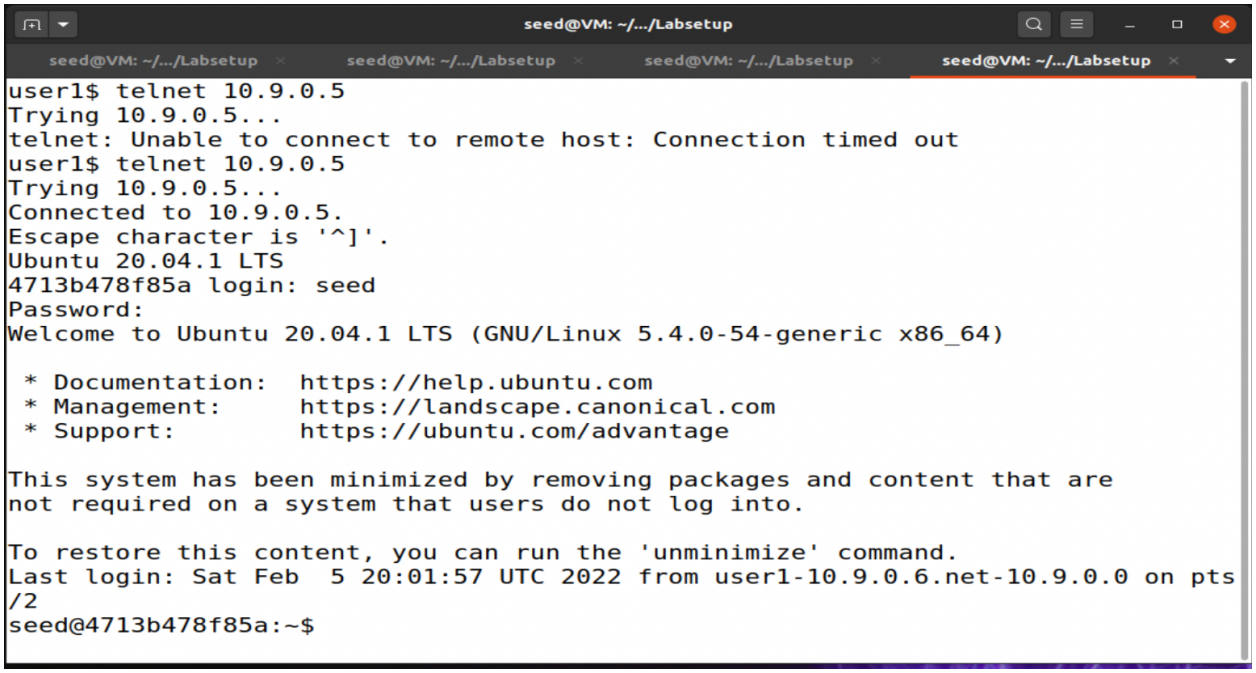
Size of the queue:

I decreased the size of the queue using sysctl -w net.ipv4.tcp_max_syn_backlog=80 which significantly increased the success rate. I was able to successfully time out the connection using only a few instances. While testing with the C program included, I discovered that it could packets significantly faster than the python script and didn't need to queue size to decrease.

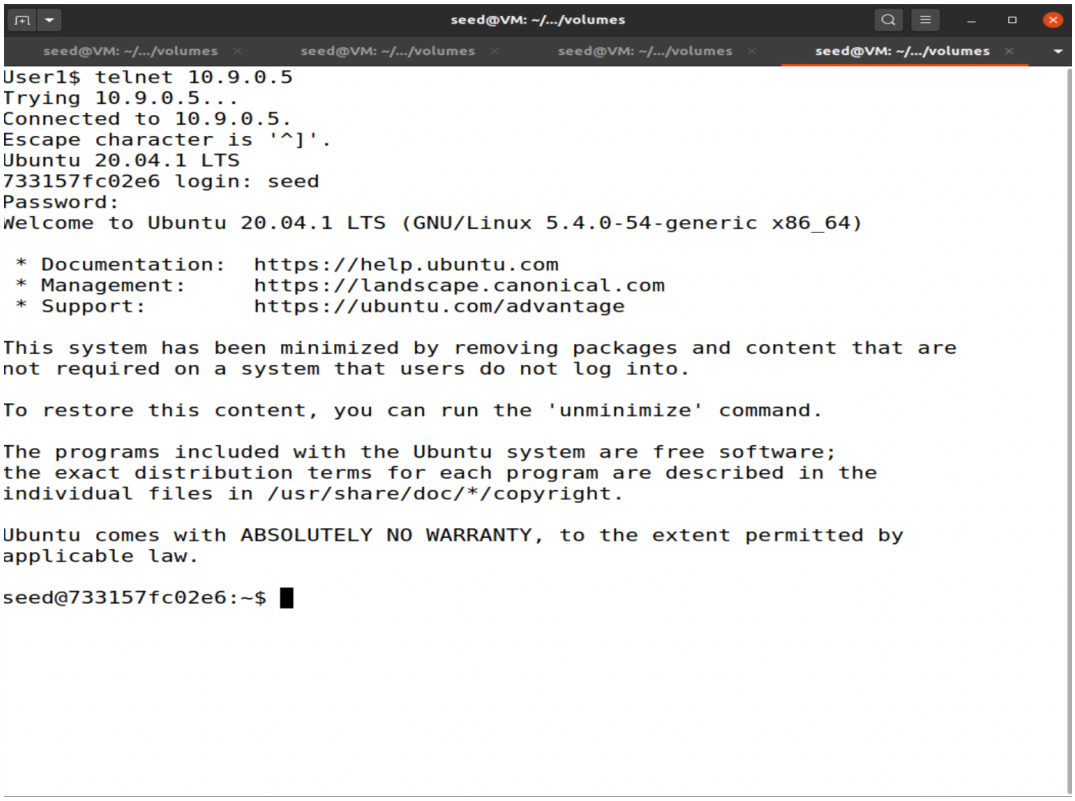
Task 1.3 - tcp flooding with cookies on

When I turned the syncookies on I ran multiple instances of the synflood file at once to attempt and overload the victim's server. However when logged into the user I was able to immediately telnet into the victim's server. Sidenote - when I turned the cookies on and ran the functions my computer fans kicked in as the processor worked harder to turn away the attacks. As soon as I disabled the cookies, my computer settled back to normal. It seems it had to work hard to throw away the extra packets.

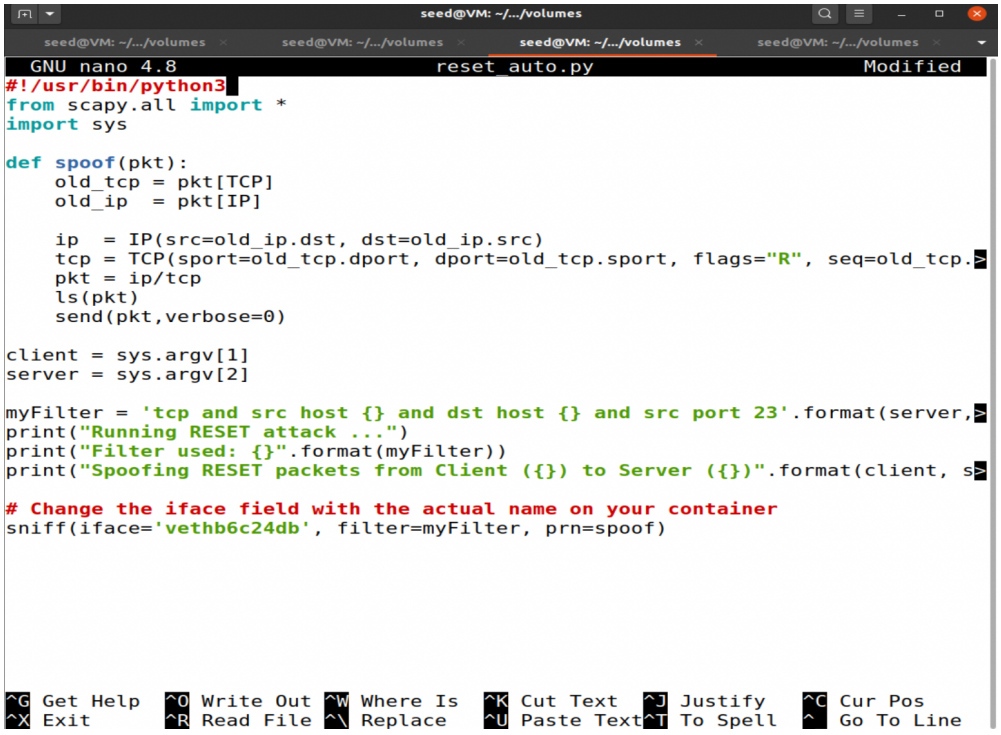




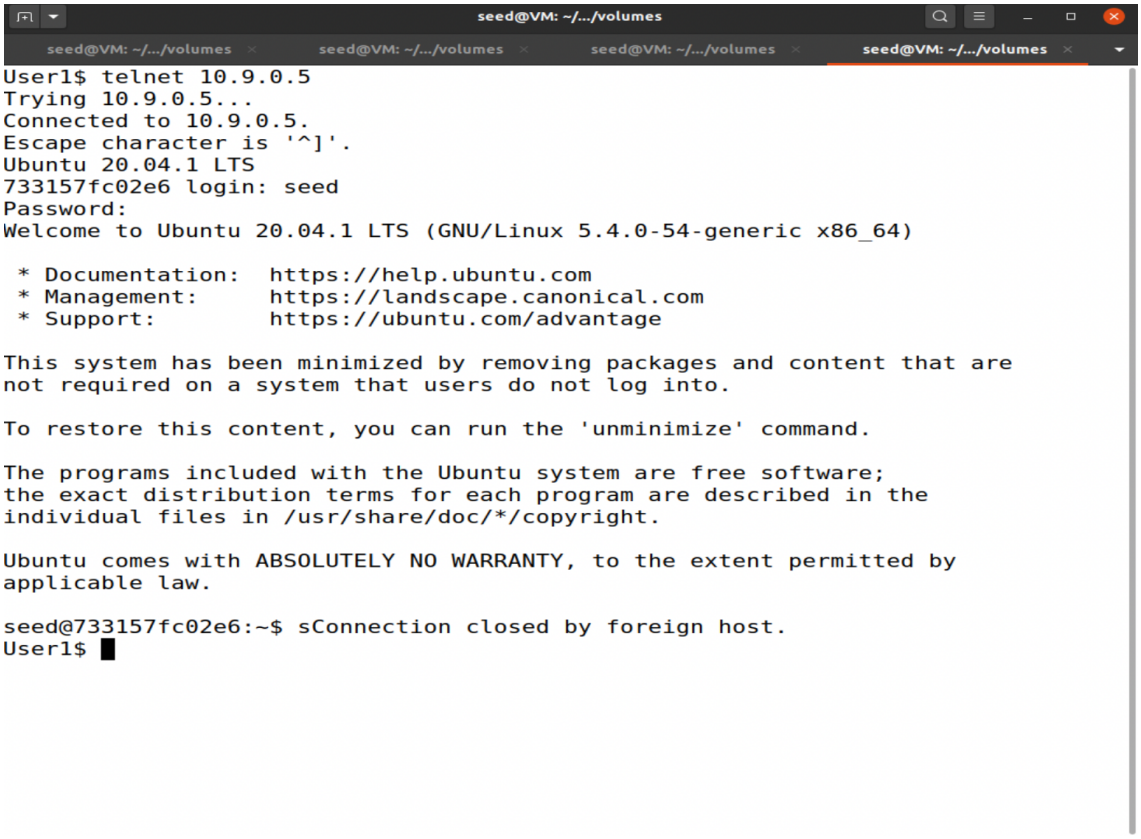
Task 2 RST Attack:

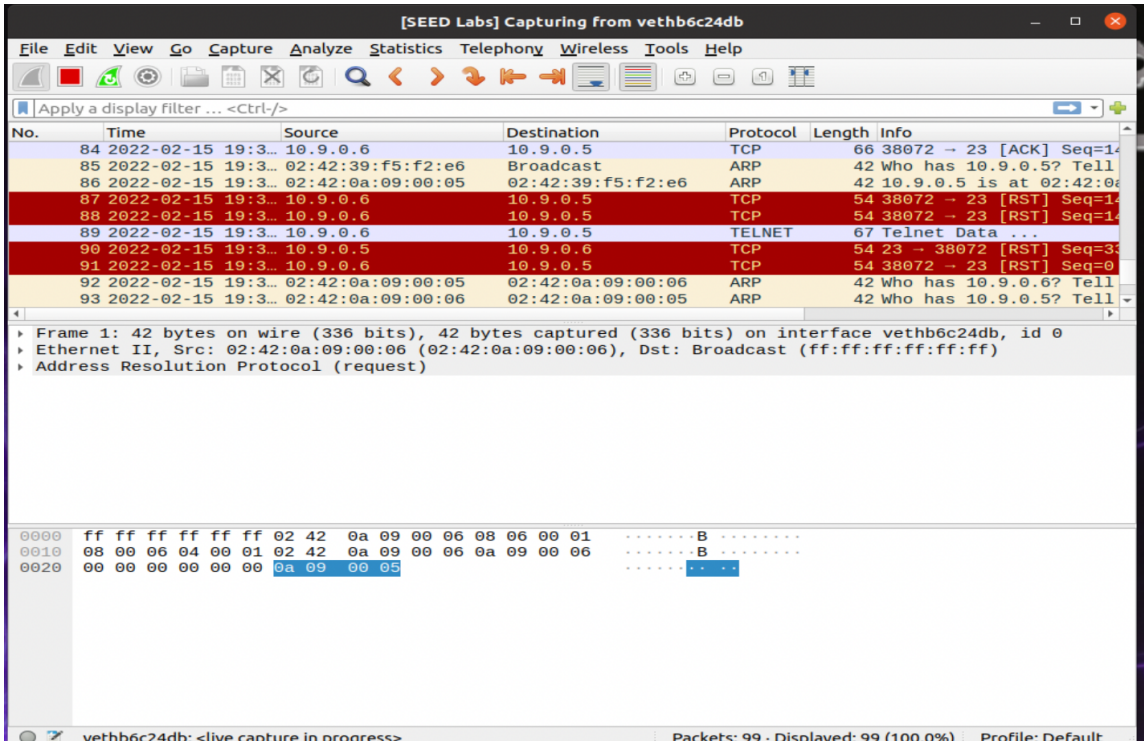


I establish a connection with the victim server from the user1

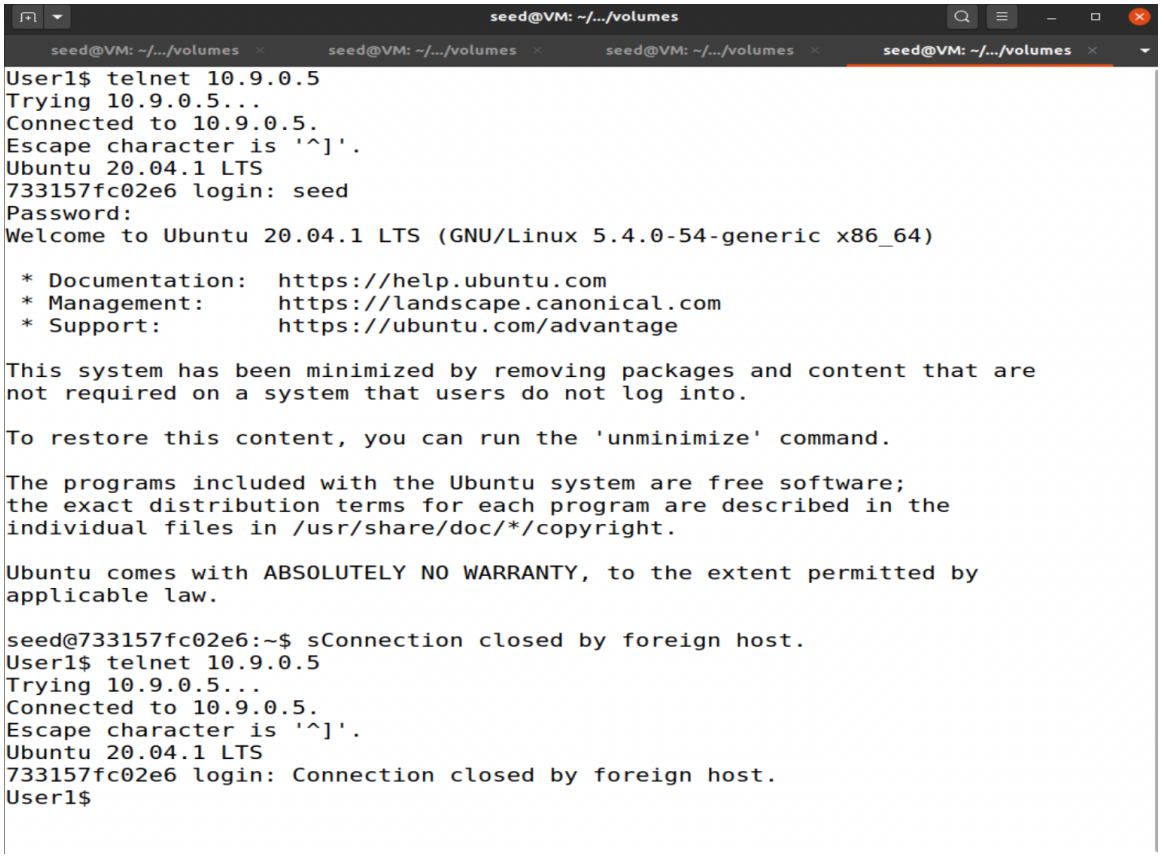


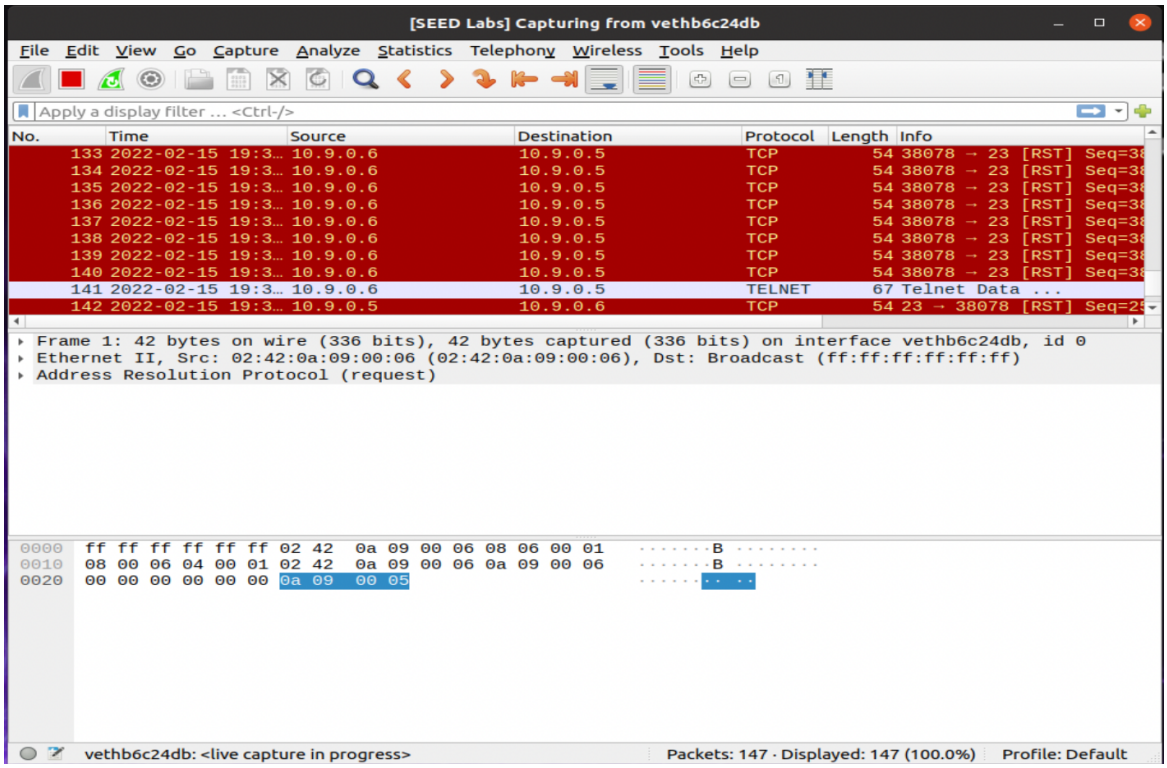
Here's the code from the attacker. I then run this code from the attacker





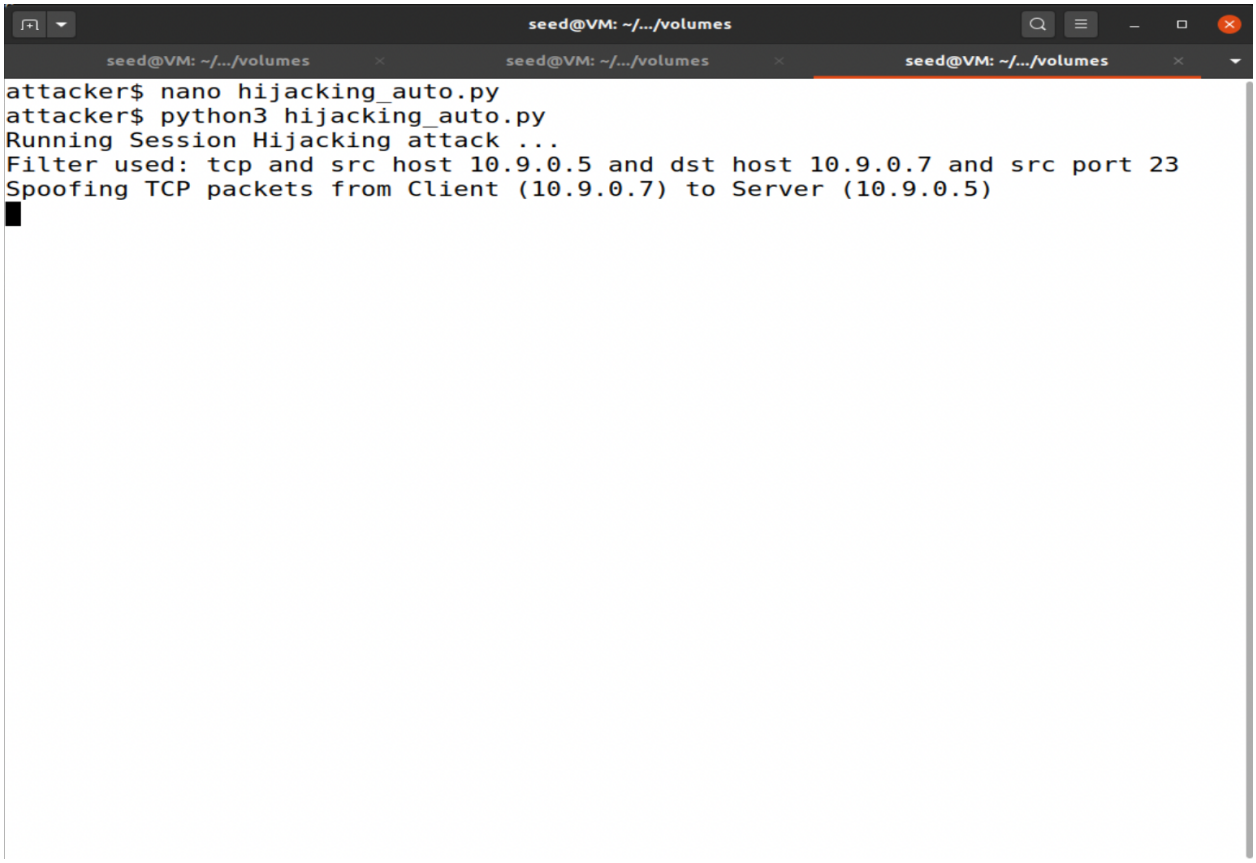
As soon as I type something in the window the RST packet is spoofed and the connection is closed



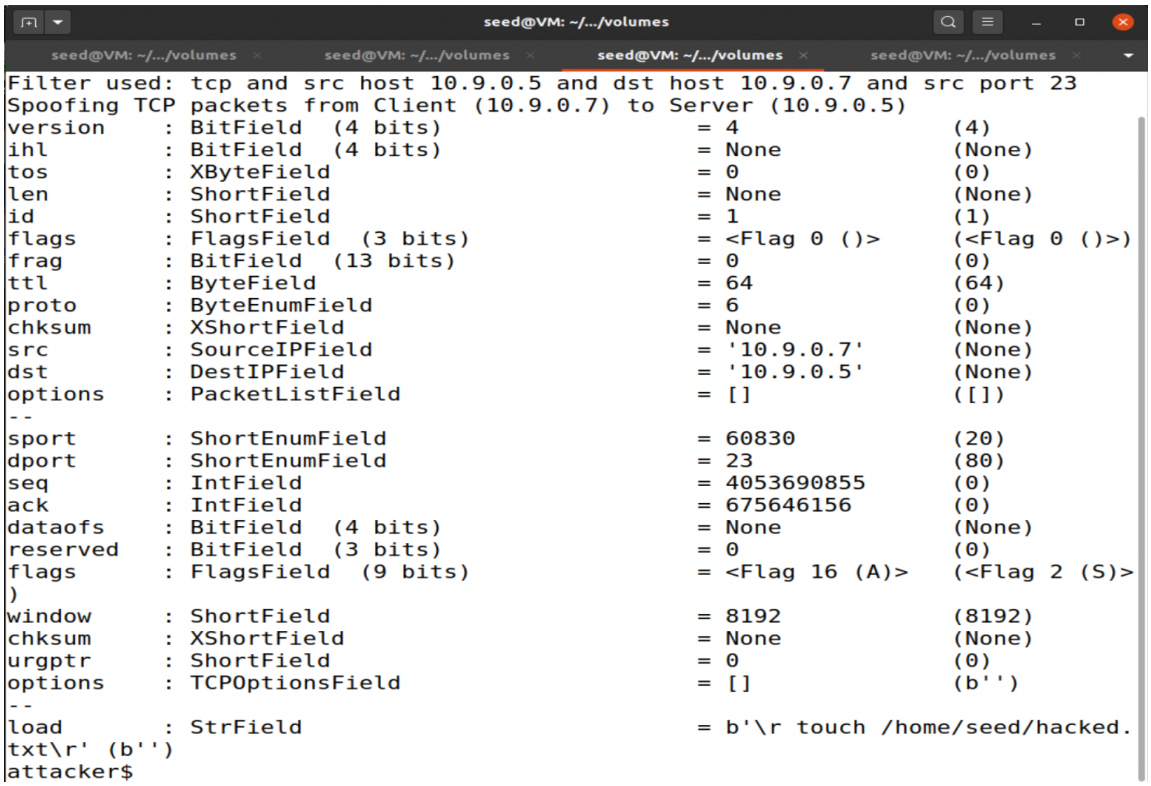


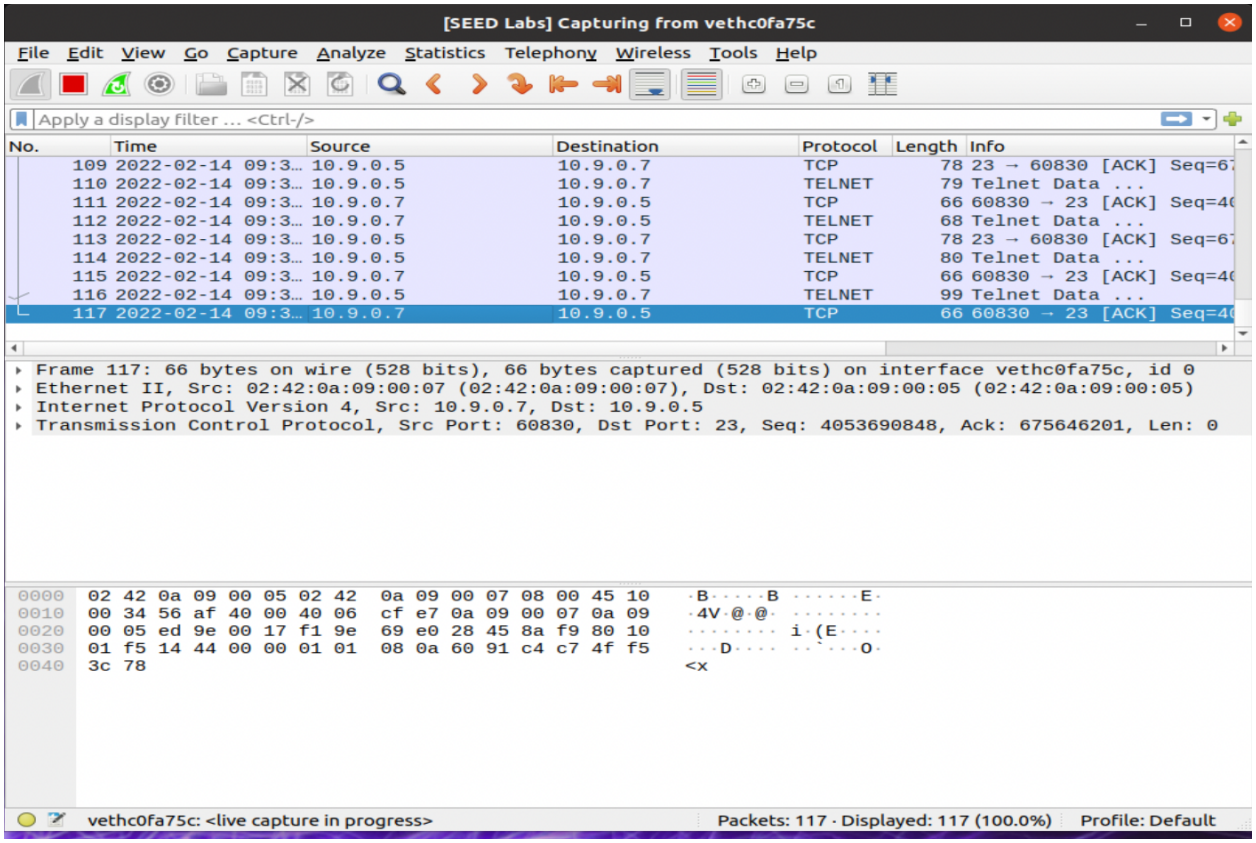
If I leave the attacker's code running the RST packet will always be sent and the user will not be able to open a connection with the victim server. It is like a DOS attack.

Task 3: Hijacking TCP connection

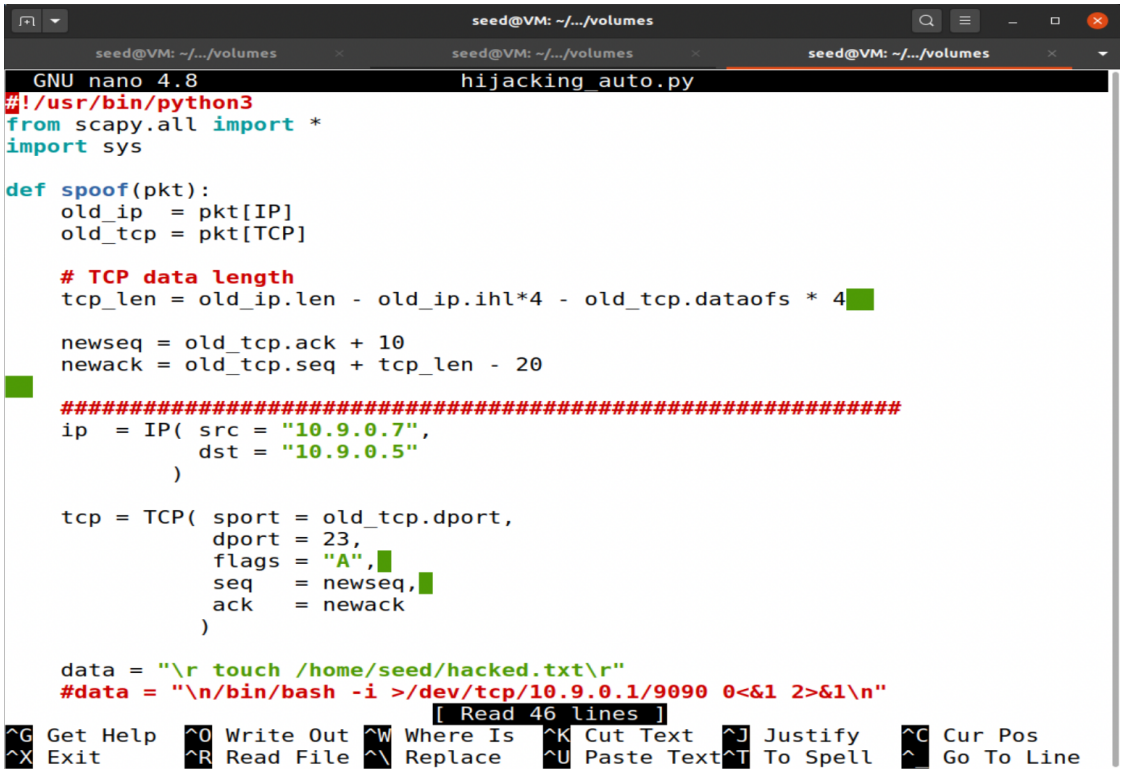


After I create a telnet connection between user 2 and the victim server. I run the auto hijacking attack program

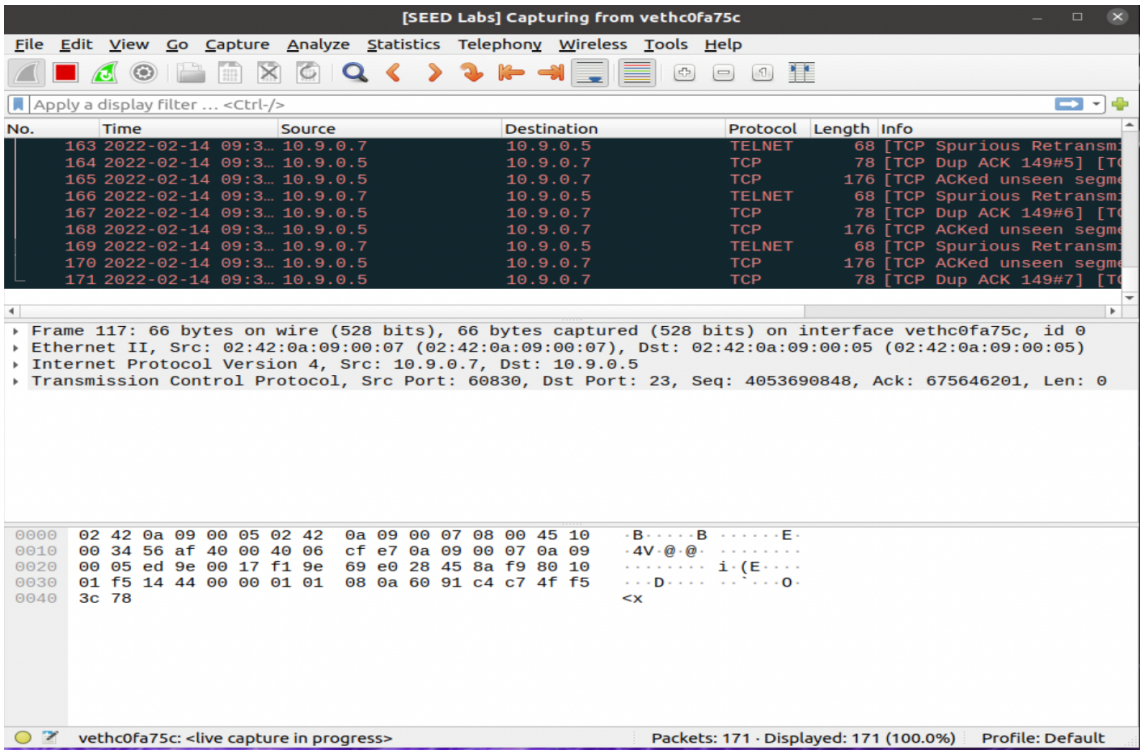




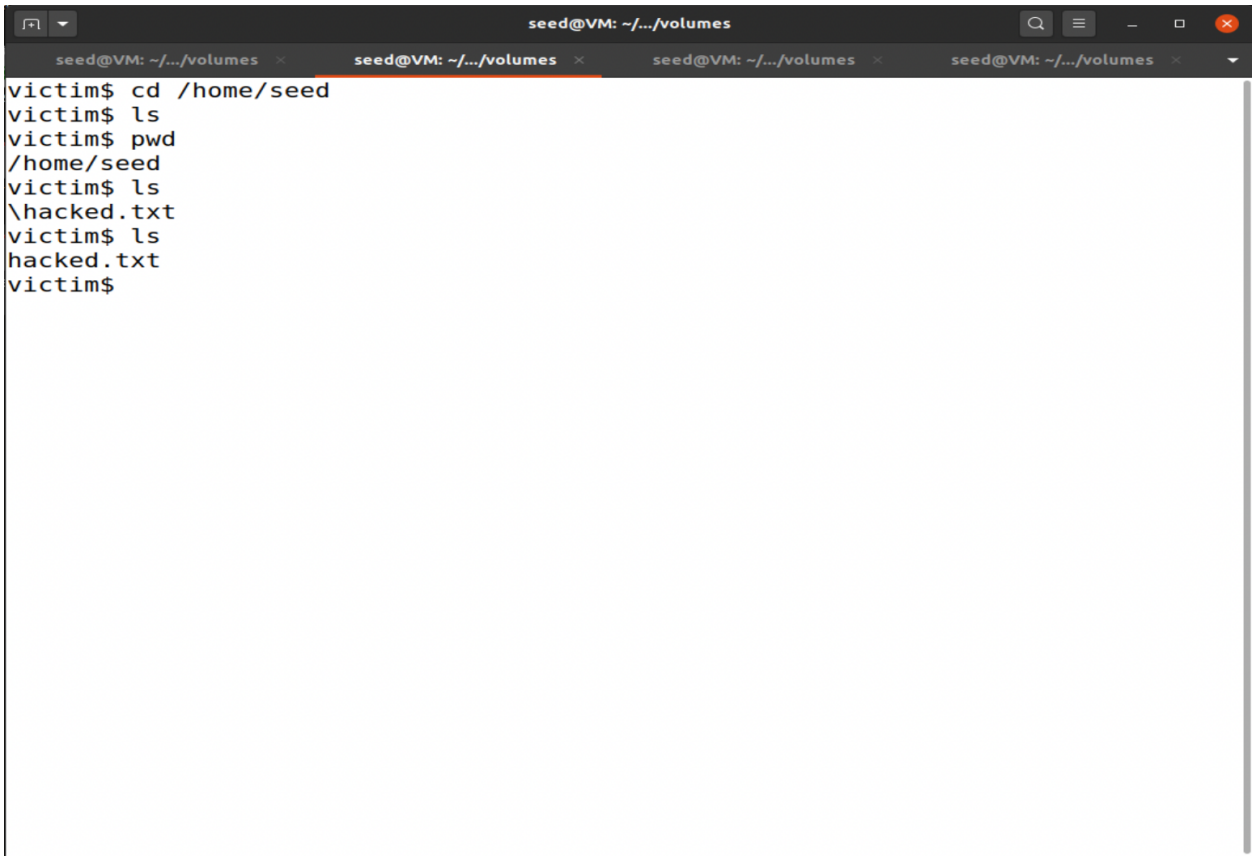
After collecting a packet, the program creates a malicious packet with data for a sequence number 10 ahead of the current packet. This will then be placed when the user types and reaches that sequence number.



This is the code that will be inserted into the telnet connection.

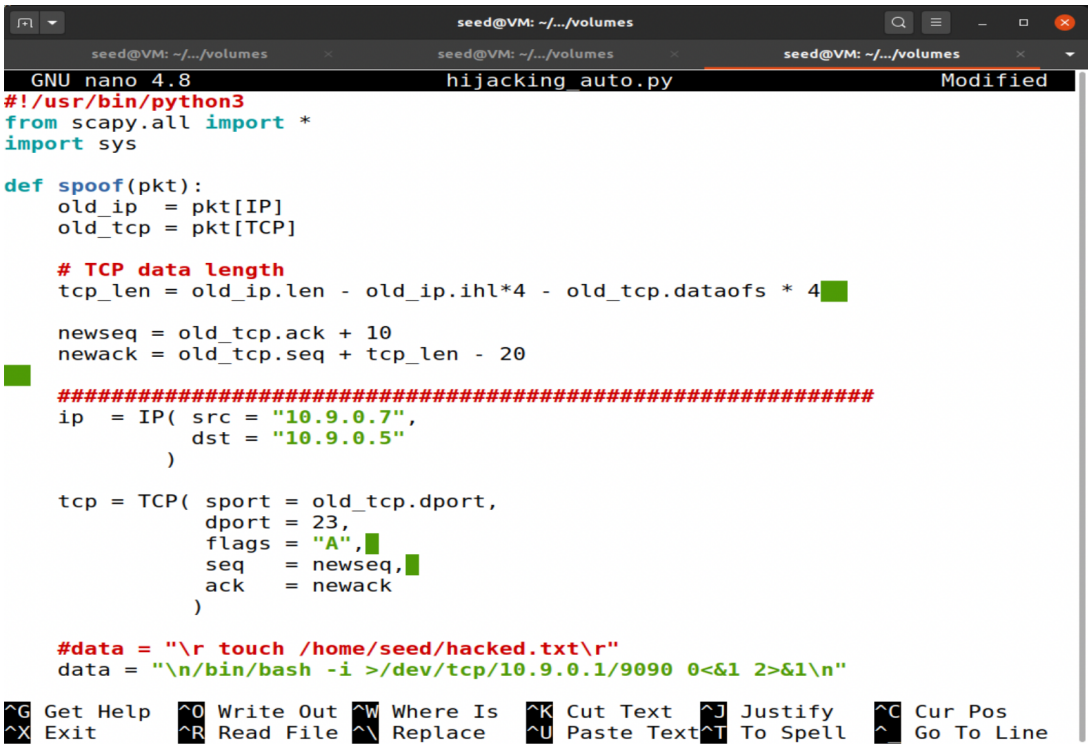


Wireshark after the sequence has reached the spoofed packet

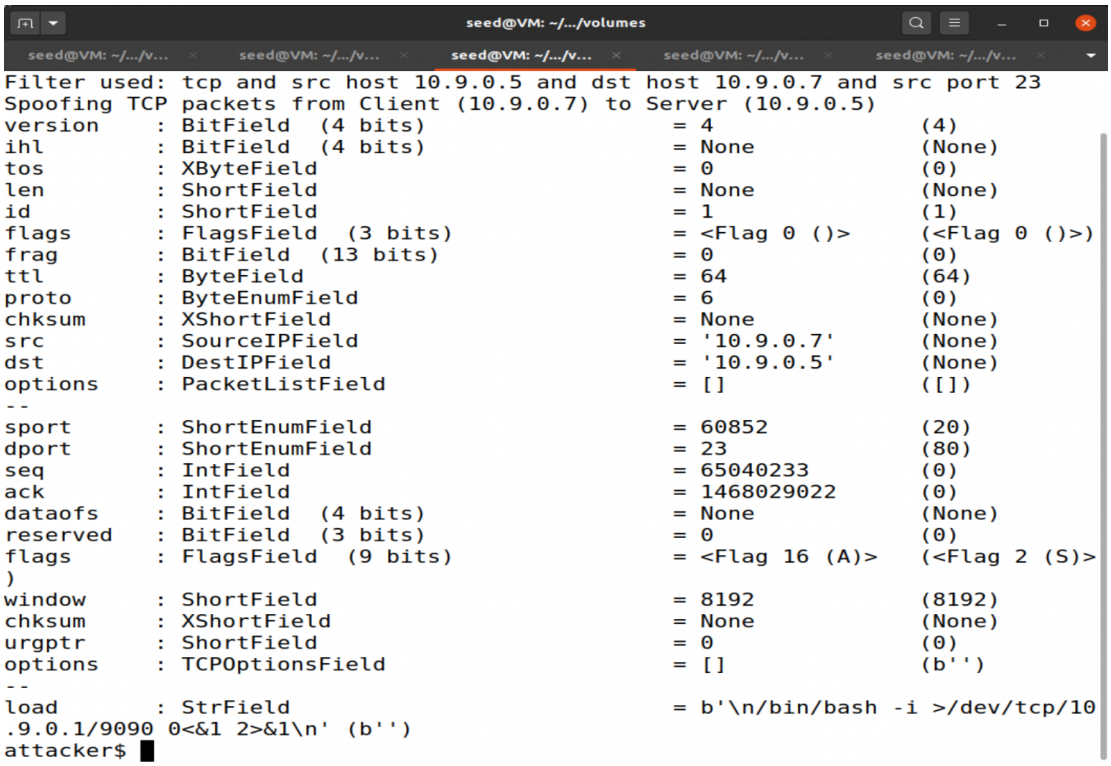


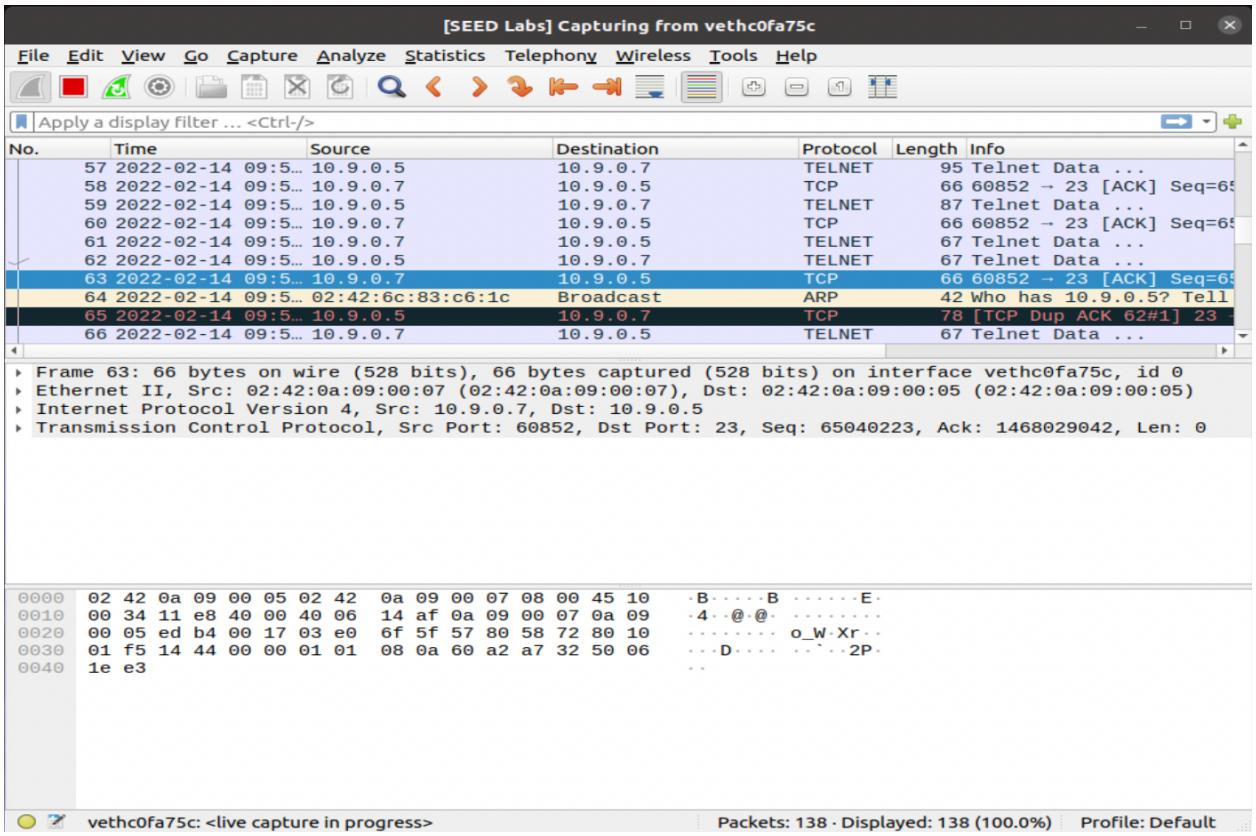
New File that appeared in the victim's server

Task 4: Reverse Shell

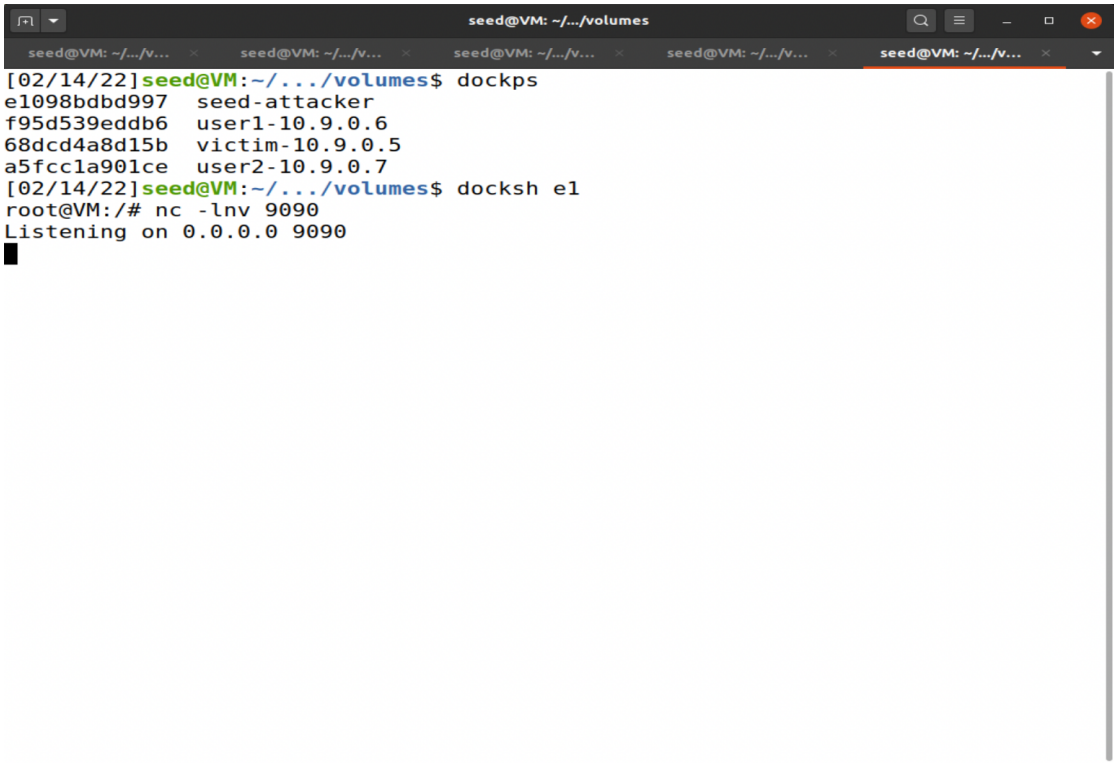


The hijacking code is the same from the previous except the command is set to send a shell to the attacker

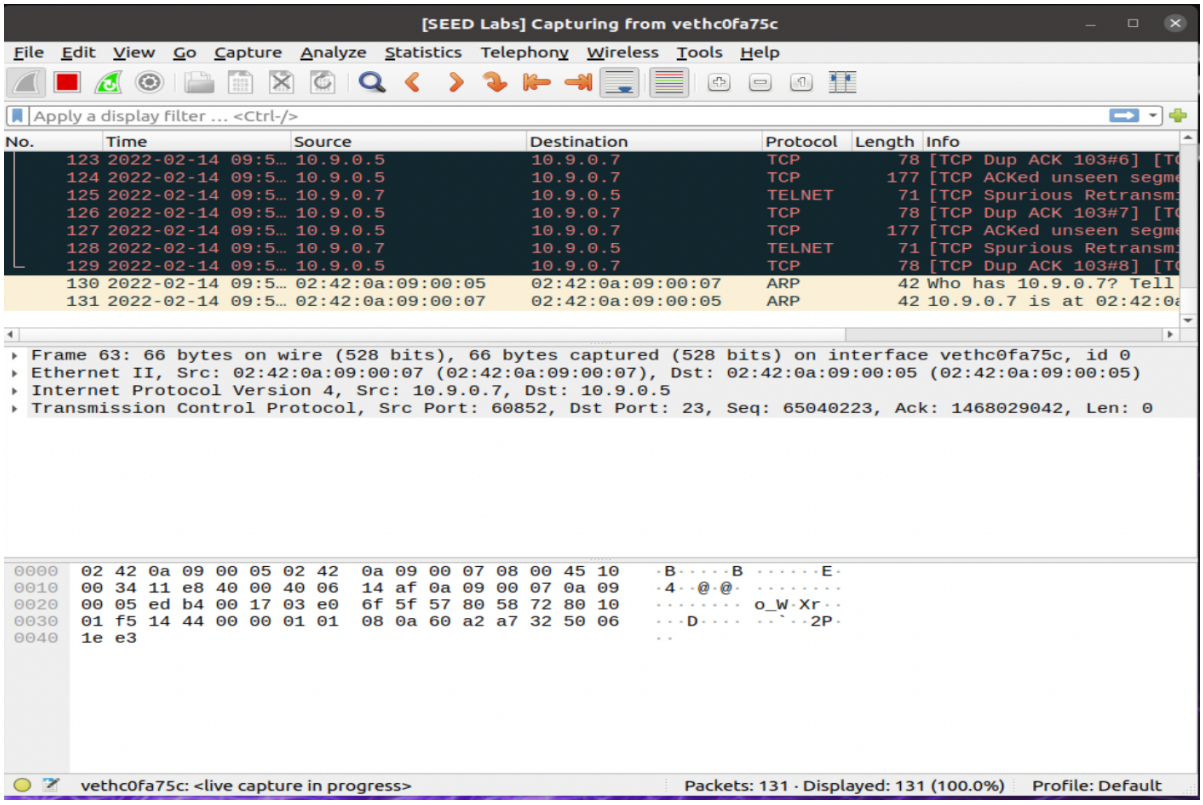




The code inserts the malicious packet with the code, the victim responds with a duplicate packet warning.



Attacker has another shell listening on port 9090



Packet gets spoofed

Attacker receives revershell, and can run commands from the victim's server