Computer Network Security

UE23CS343AB6

5th Semester, Academic Year 2023

Date: 22/08/2025

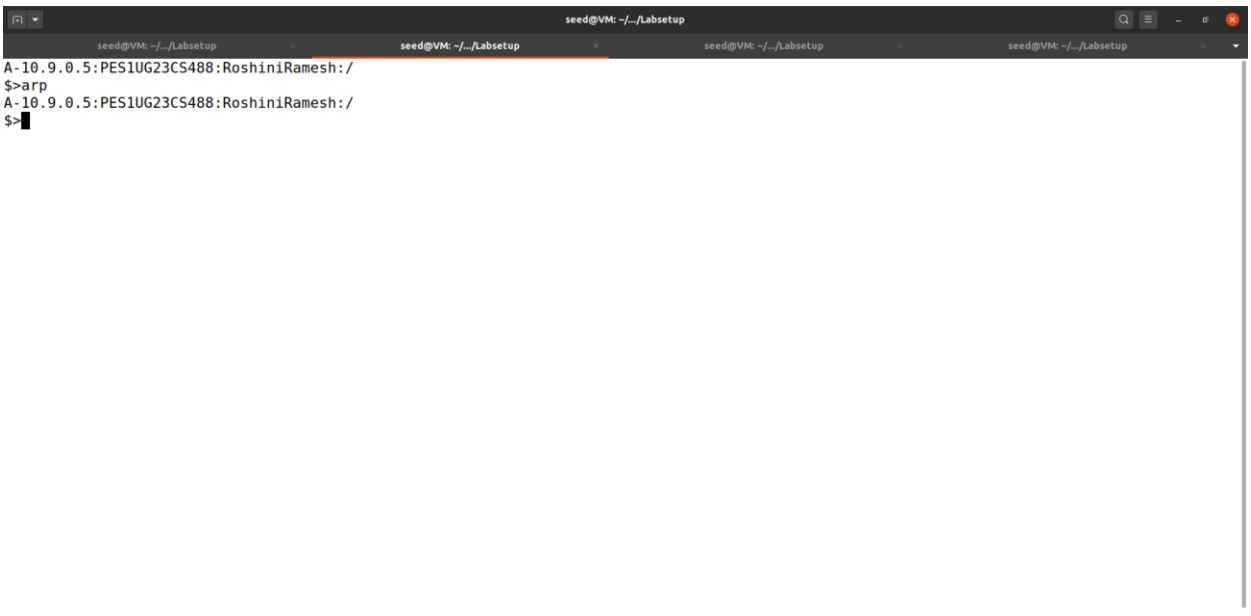
Name: Roshini Ramesh	SRN: PES1UG23CS488	Section: H

TASK 1: USING ARP REQUEST

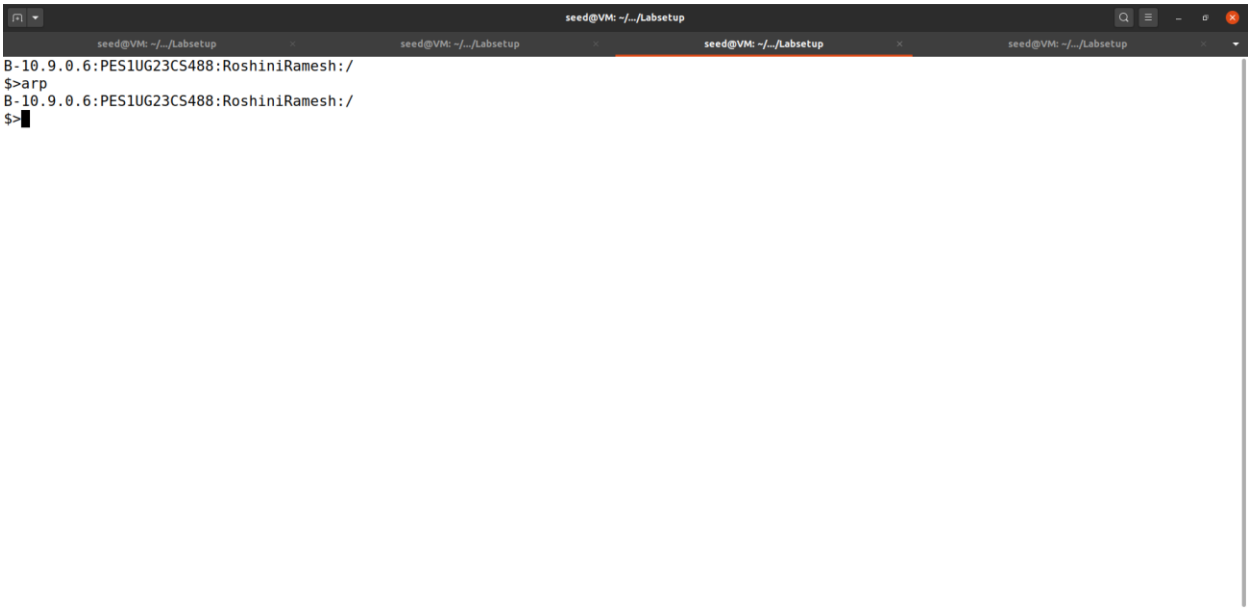
WITHOUT PASSING PARAMETERS TO ETHER()

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

A-10.9.0.5 Terminal Output:



B-10.9.0.6 Terminal Output:



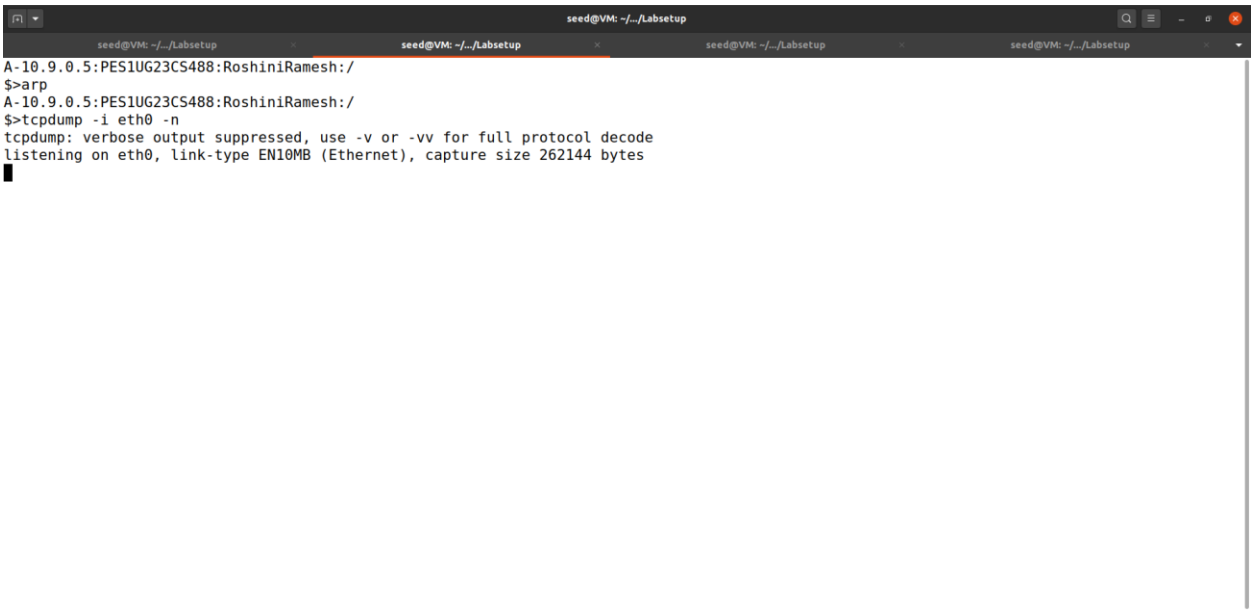
M-10.9.0.105 Terminal Output:

Explanation:

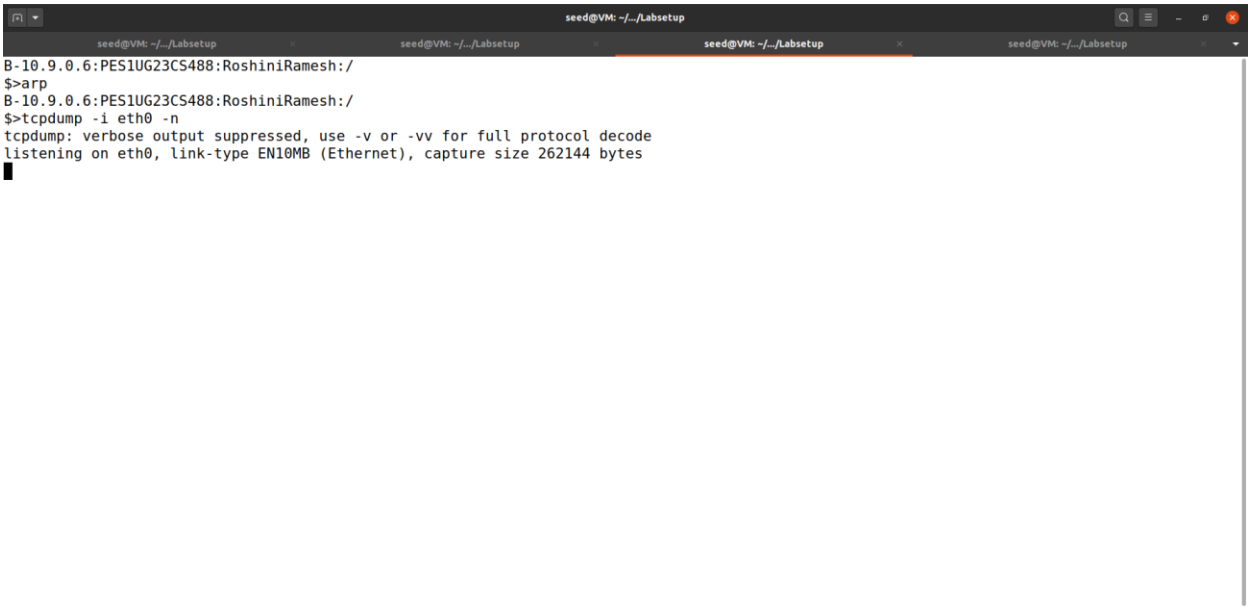
We are checking the status of the arp table on all 3 interfaces initially and it is all empty.

RUN TCPDUMP ON BOTH THE HOSTS A AND B.

A-10.9.0.5 Terminal Output:



B-10.9.0.6 Terminal Output:

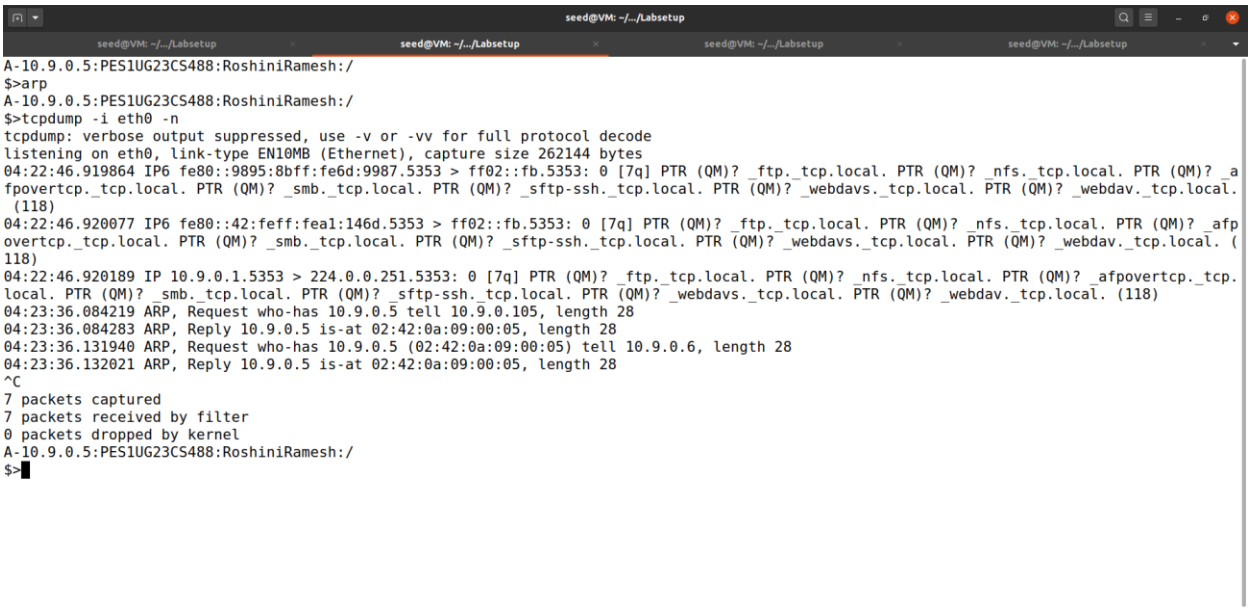


Explanation:

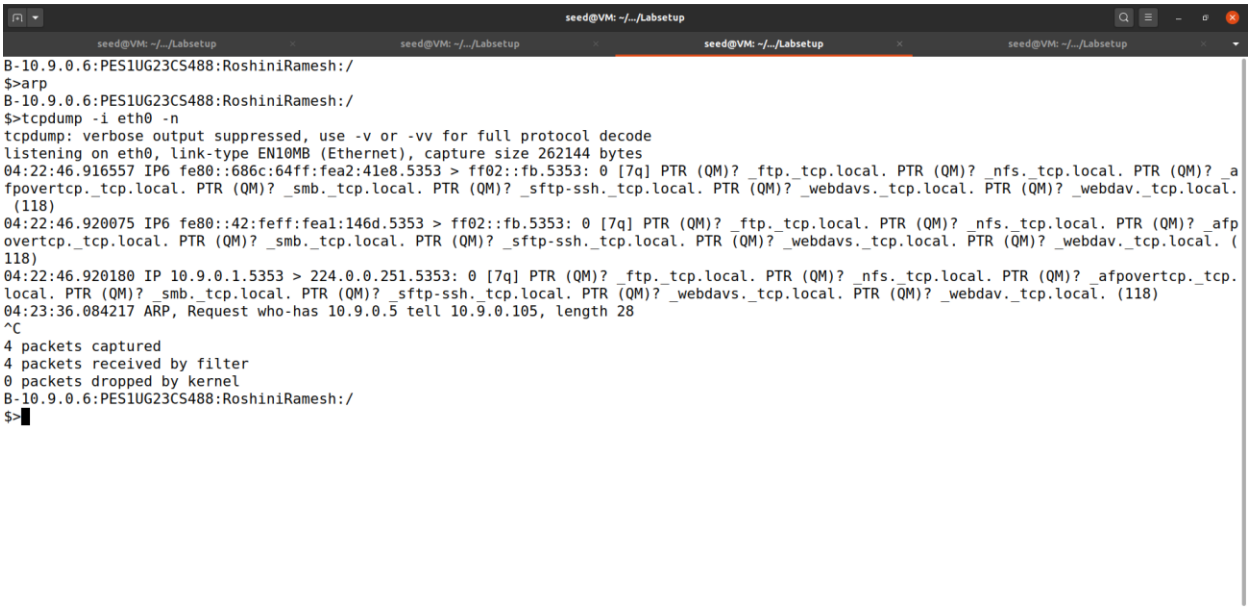
Here, we are using tcpdump to capture all packets on network interface eth0 on both A and B.

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

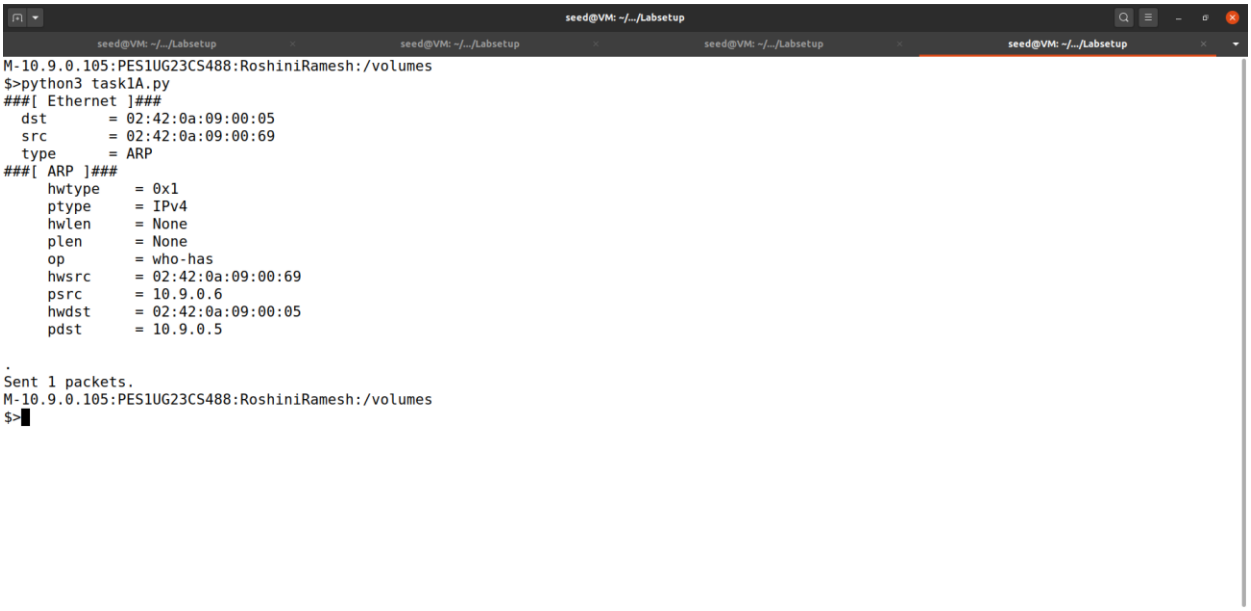
A-10.9.0.5 Terminal Output:



B-10.9.0.6 Terminal Output:



M-10.9.0.105 Terminal Output:

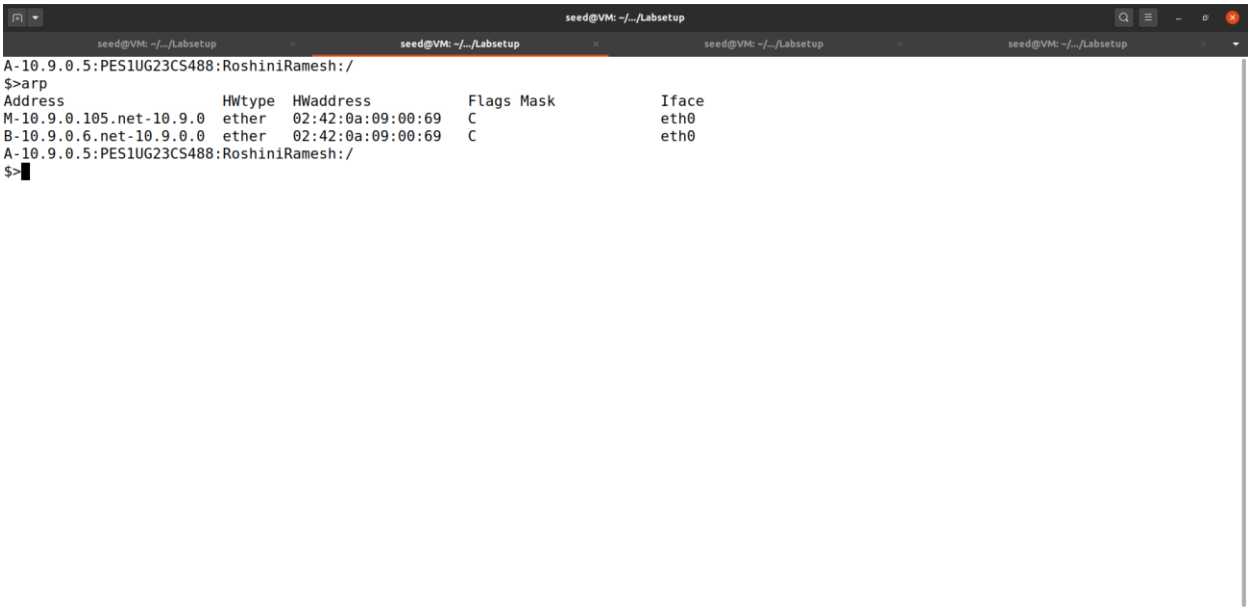


Explanation:

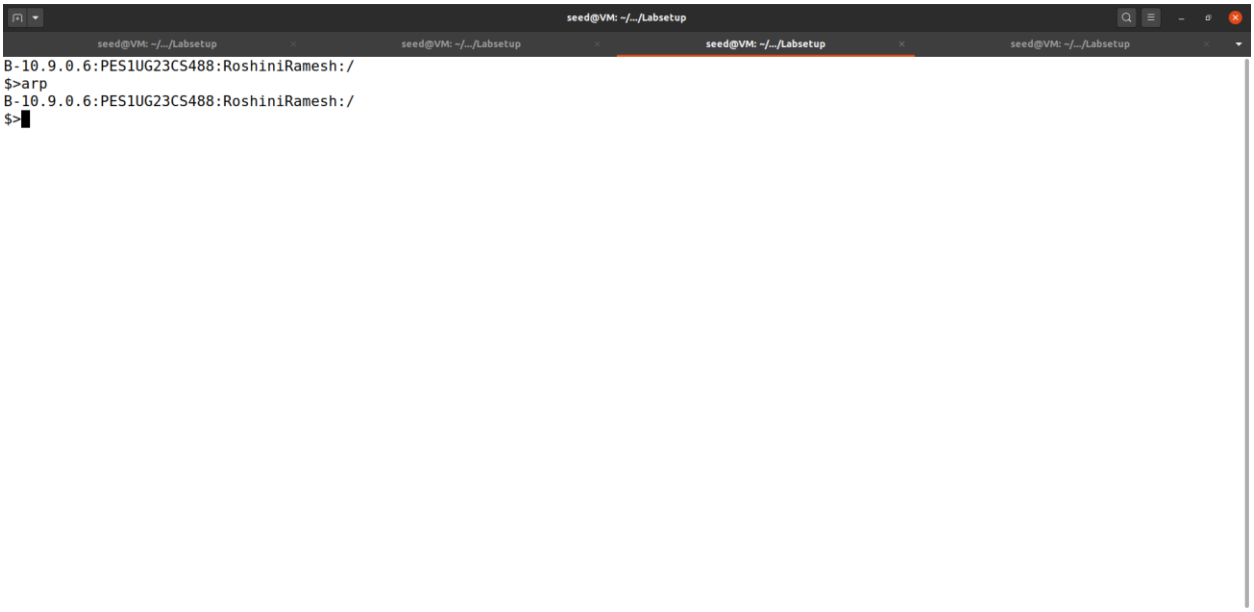
Machine M sends a forged ARP packet, on behalf of B by using B's IP address and updates machine A's ARP table. Scapy is used to make and send the packet. First an Ethernet frame is built using Ether(), then an ARP packet with spoofed details is created and then sent using sendp().

ARP OUTPUT AFTER THE ATTACK

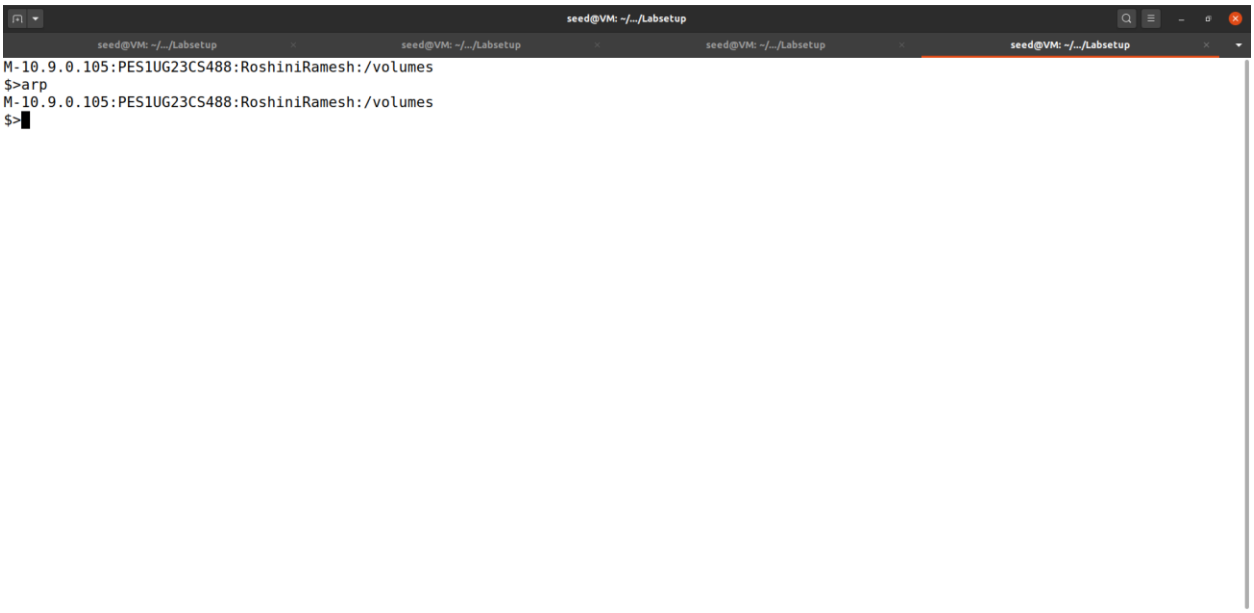
A-10.9.0.5 Terminal Output:



B-10.9.0.6 Terminal Output:



M-10.9.0.105 Terminal Output:

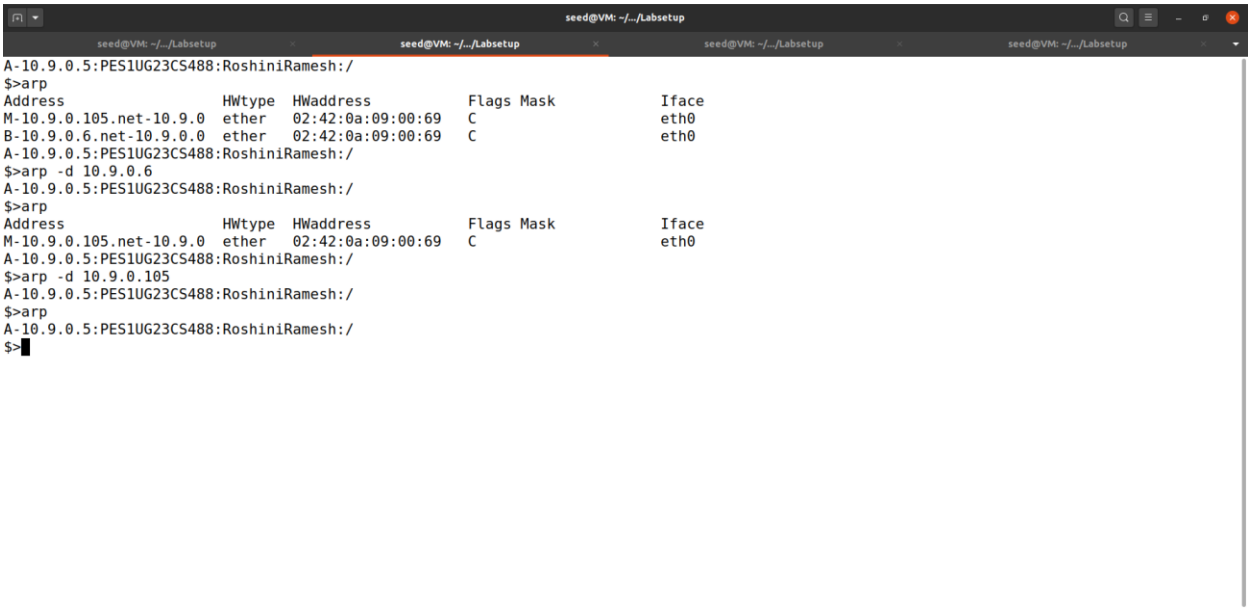


Explanation:

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:



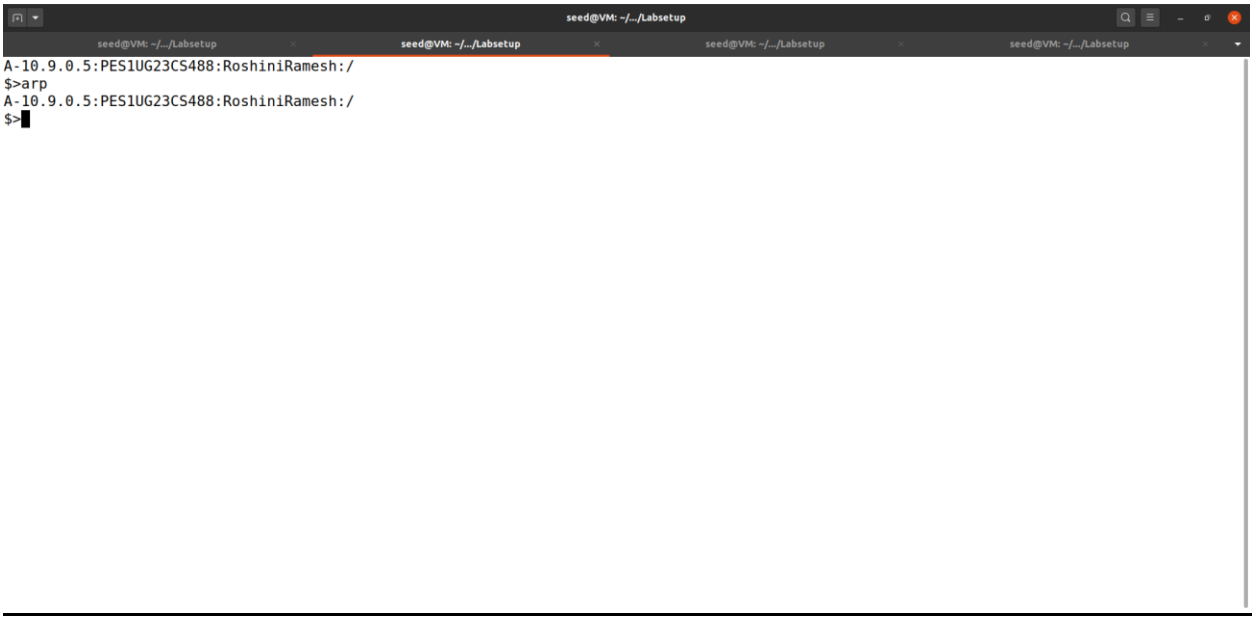
Explanation:

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

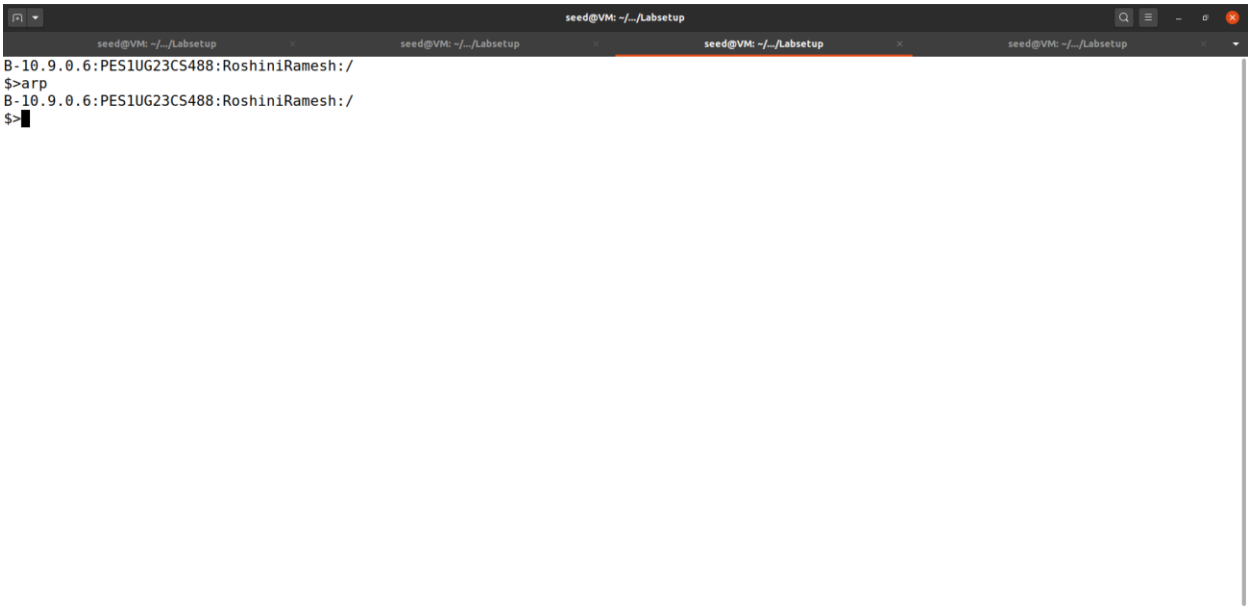
ETHER() WITH PARAMETERS

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

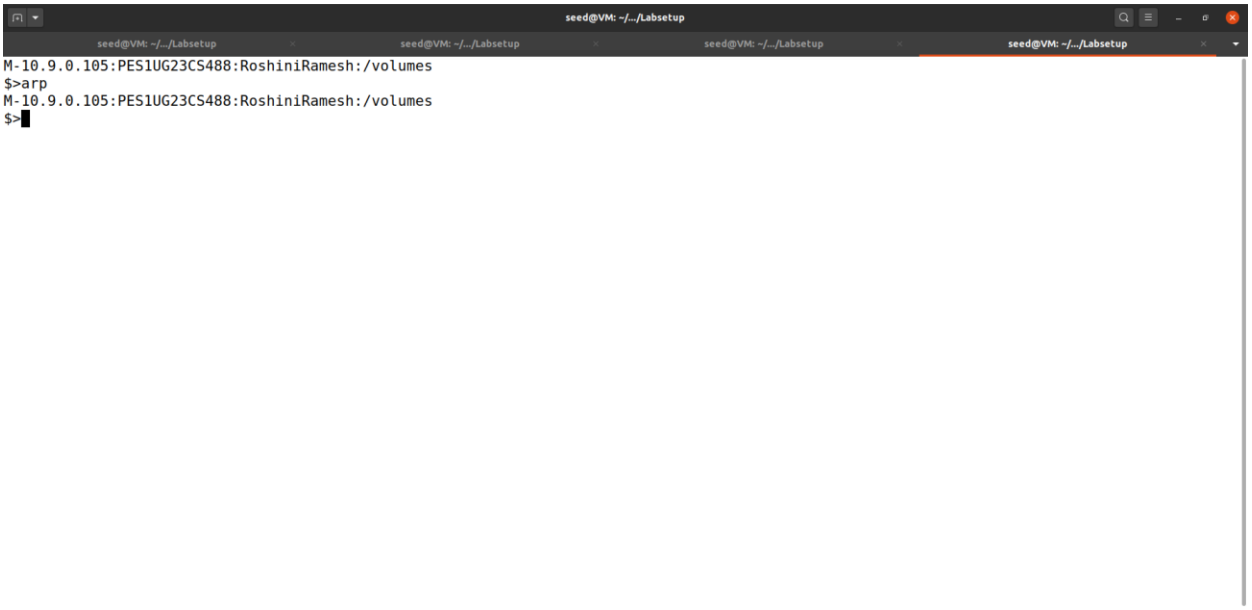
A-10.9.0.5 Terminal Output:



B-10.9.0.6 Terminal Output:



M-10.9.0.105 Terminal Output:

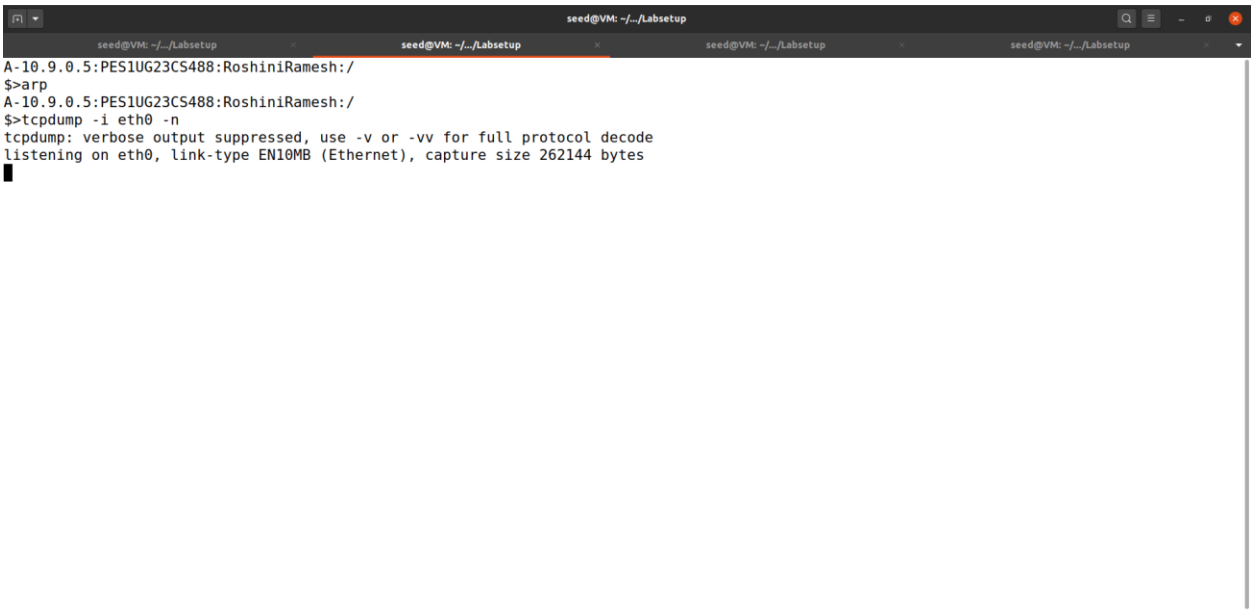


Explanation:

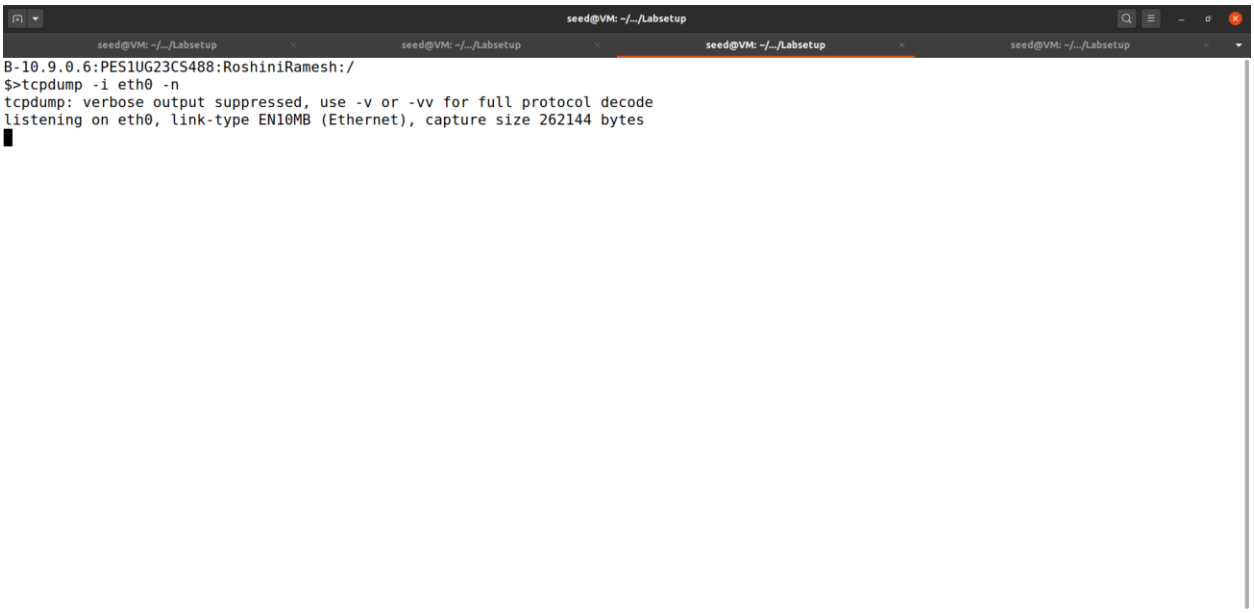
The ARP Table on all the machines is empty.

RUN TCPDUMP ON BOTH THE HOSTS A AND B.

A-10.9.0.5 Terminal Output:



B-10.9.0.6 Terminal Output:



Explanation:

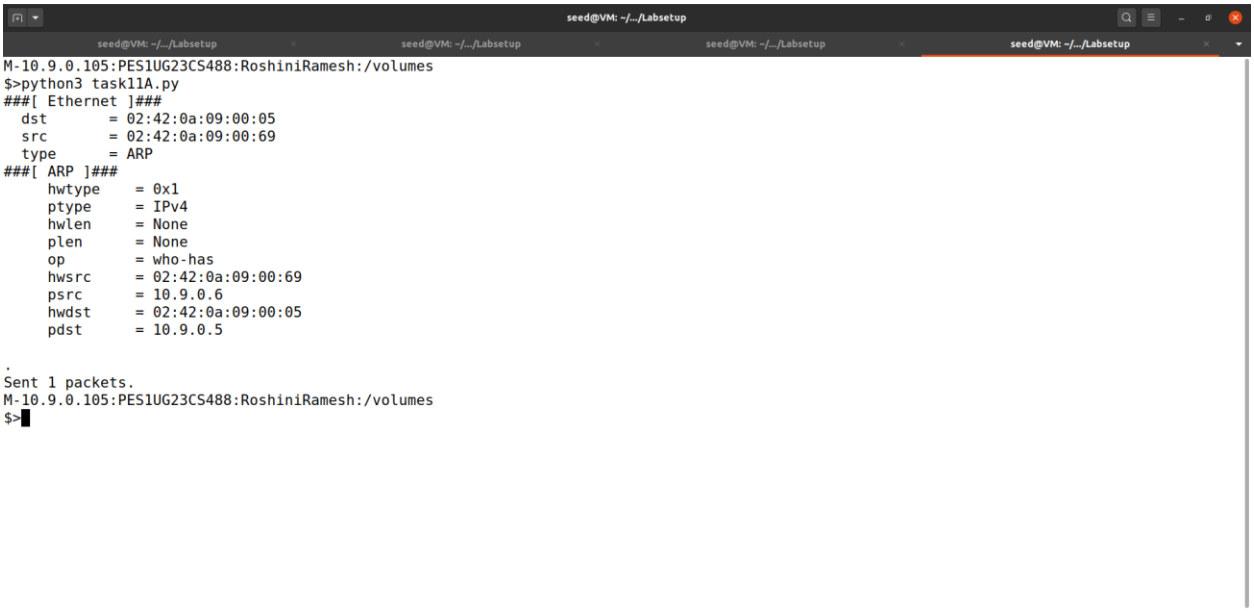
Here, we are using tcpdump to capture all packets on network interface eth0 on both A and B.

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

A-10.9.0.5 Terminal Output:

B-10.9.0.6 Terminal Output:

M-10.9.0.105 Terminal Output:



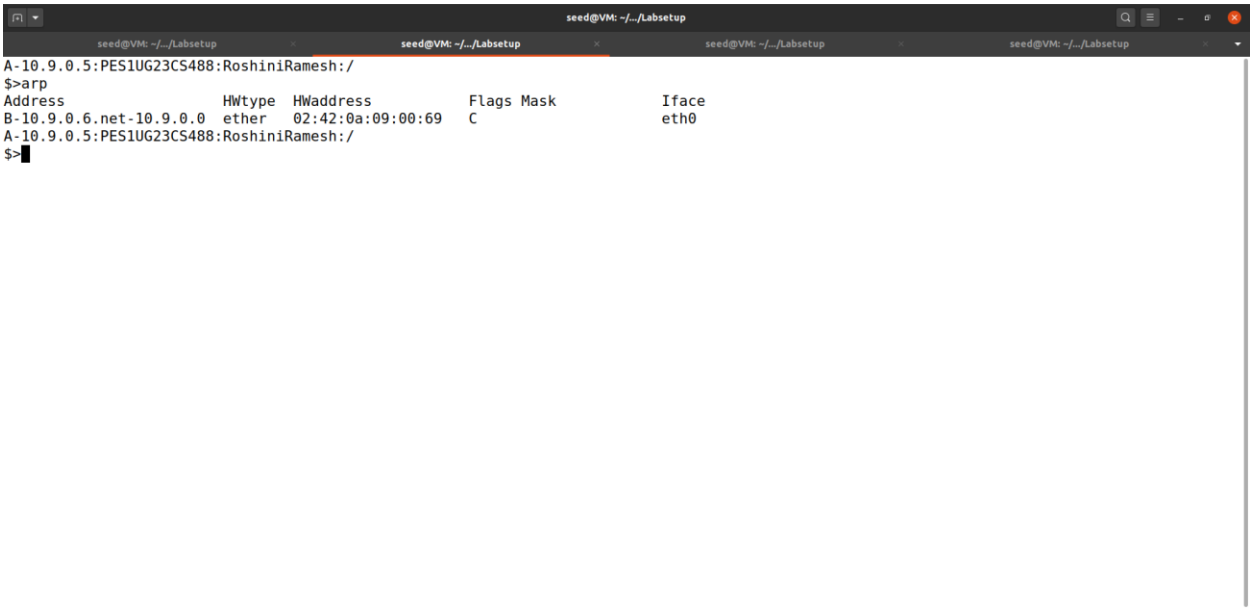
Explanation:

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,

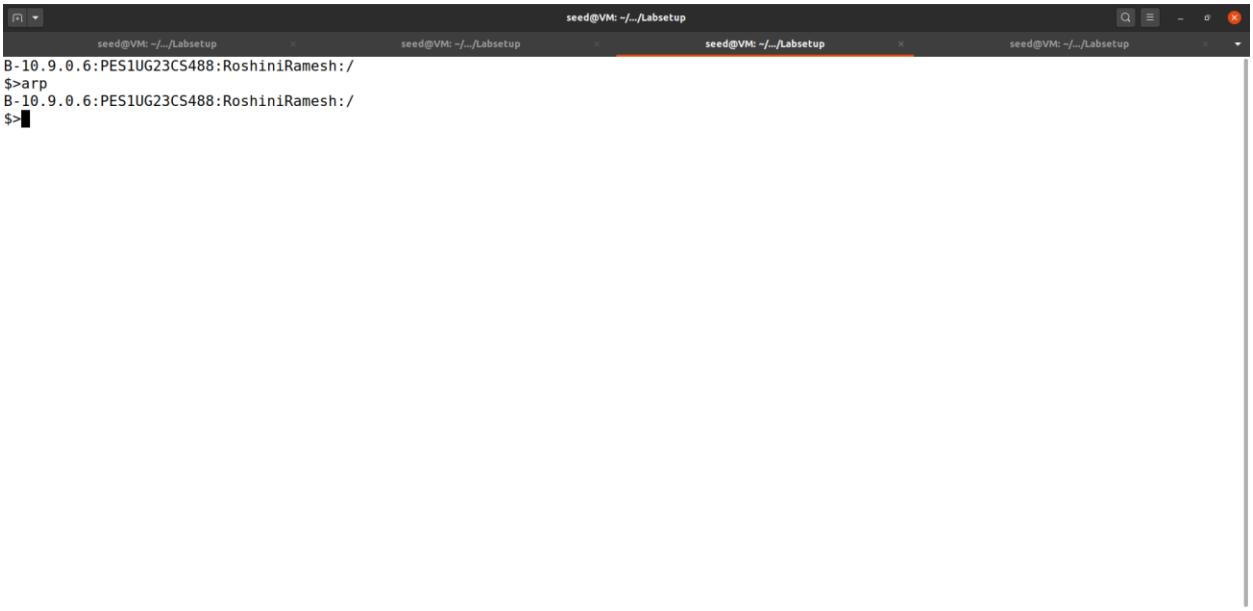
we are making a targeted attack. Hence, no broadcast occurs and hence the attacker M can’t be traced on the ARP table.

DISPLAY ARP TABLE ON ALL THE MACHINES

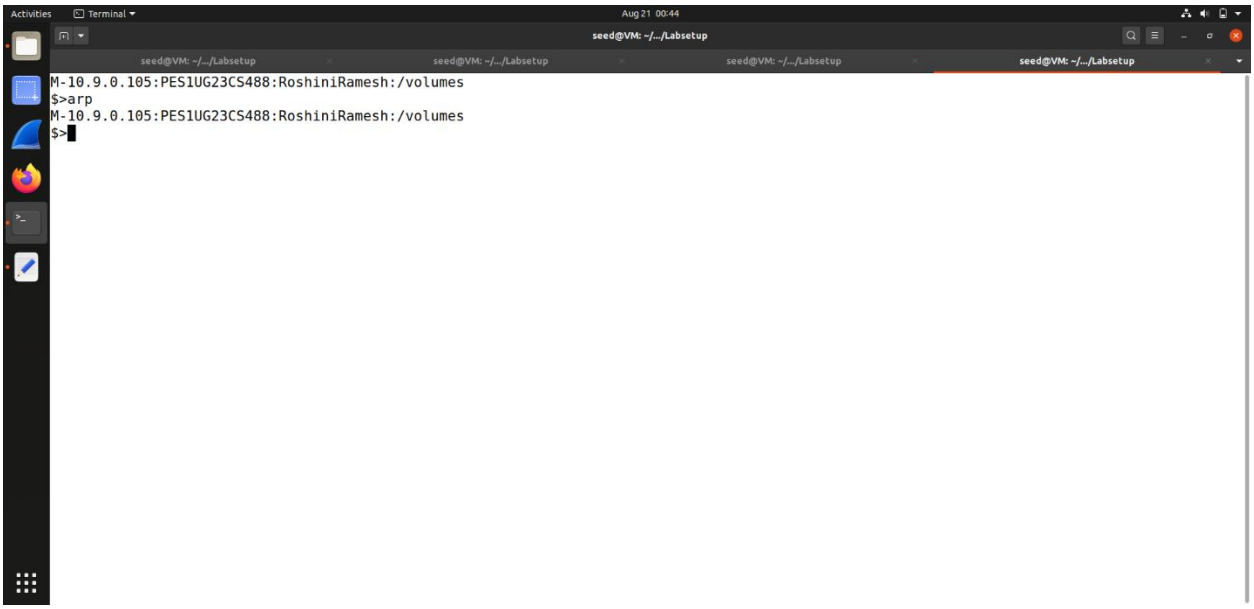
A-10.9.0.5 Terminal Output:



B-10.9.0.6 Terminal Output:



M-10.9.0.105 Terminal Output:

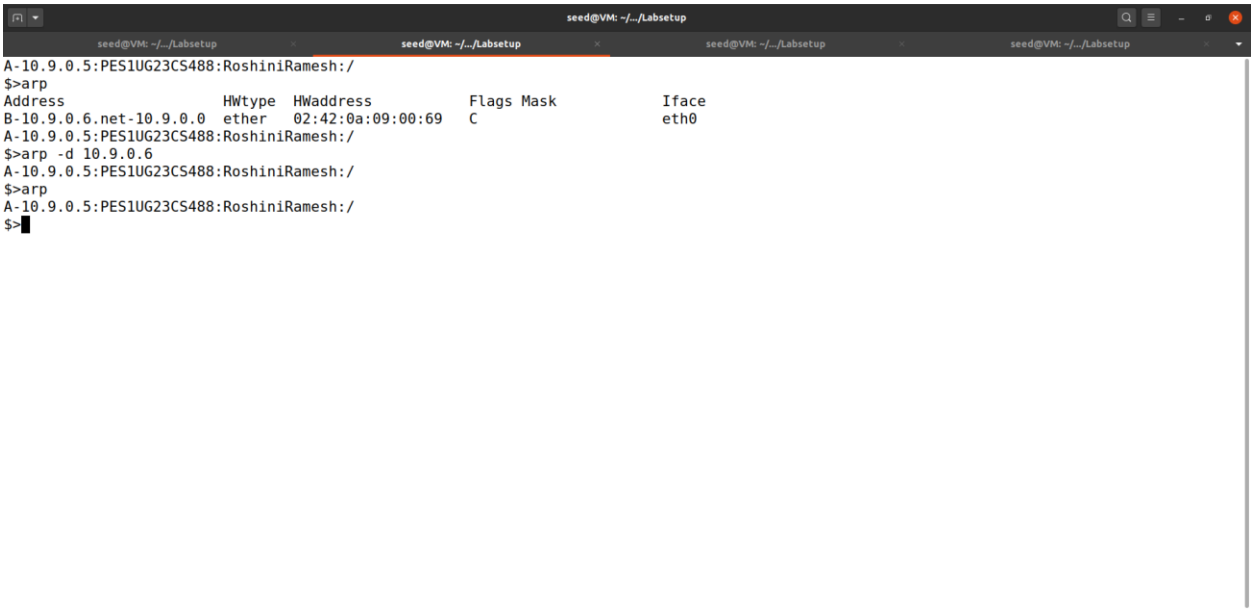


Explanation:

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:



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?

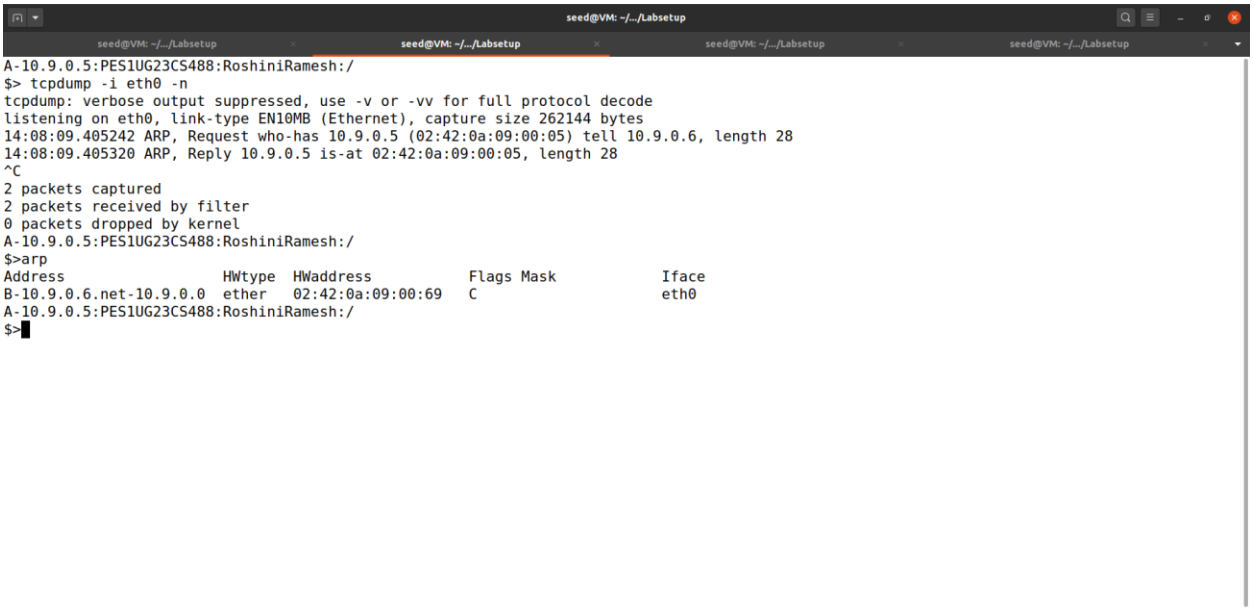
In method 1, the ARP packets are broadcasted and hence all the hosts on the network can see the forged ARP packet.

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

TASK 2: USING ARP REPLY

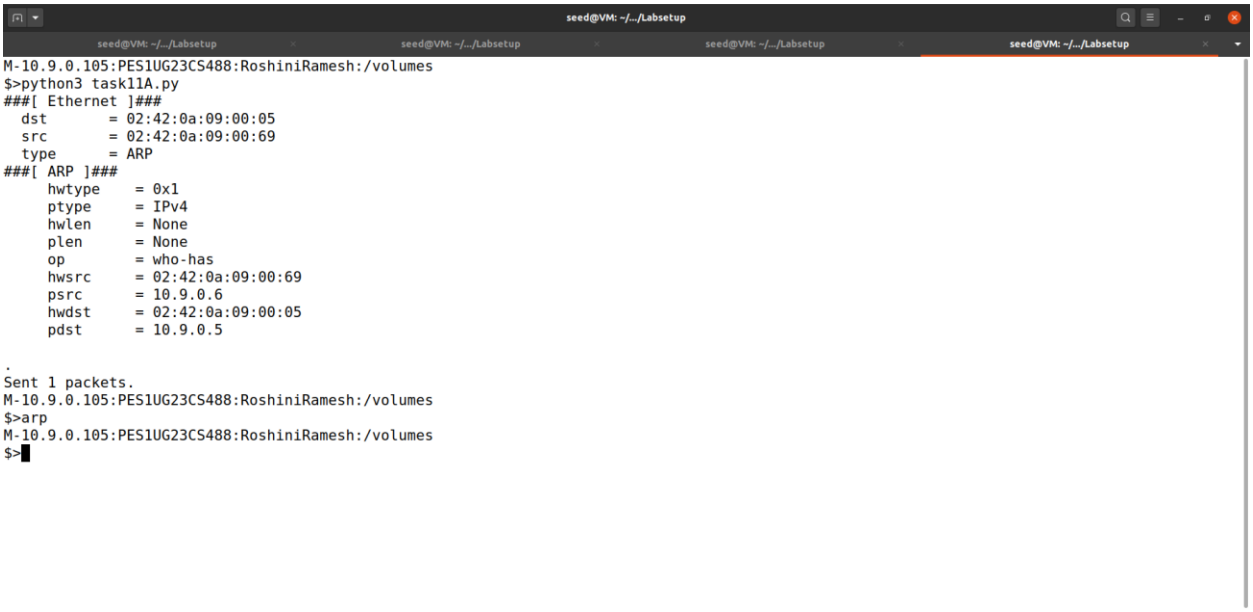
PUTTING B'S IP IN A'S ARP CACHE

A-10.9.0.5 Terminal Output:



B-10.9.0.6 Terminal Output:

M-10.9.0.105 Terminal Output:

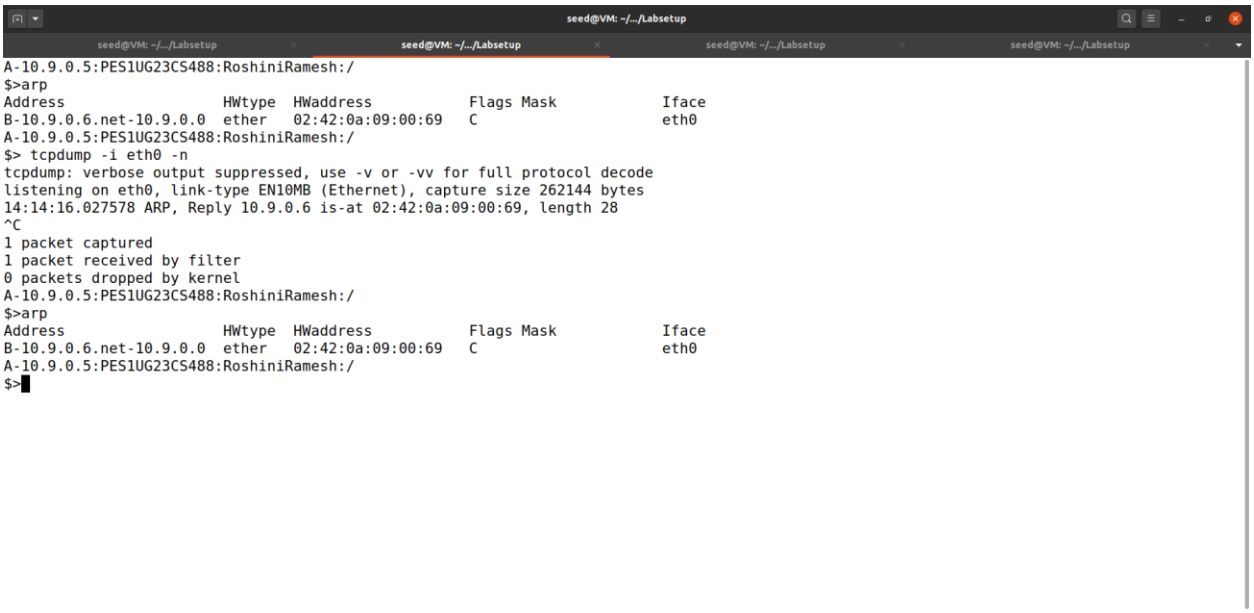


Explanation:

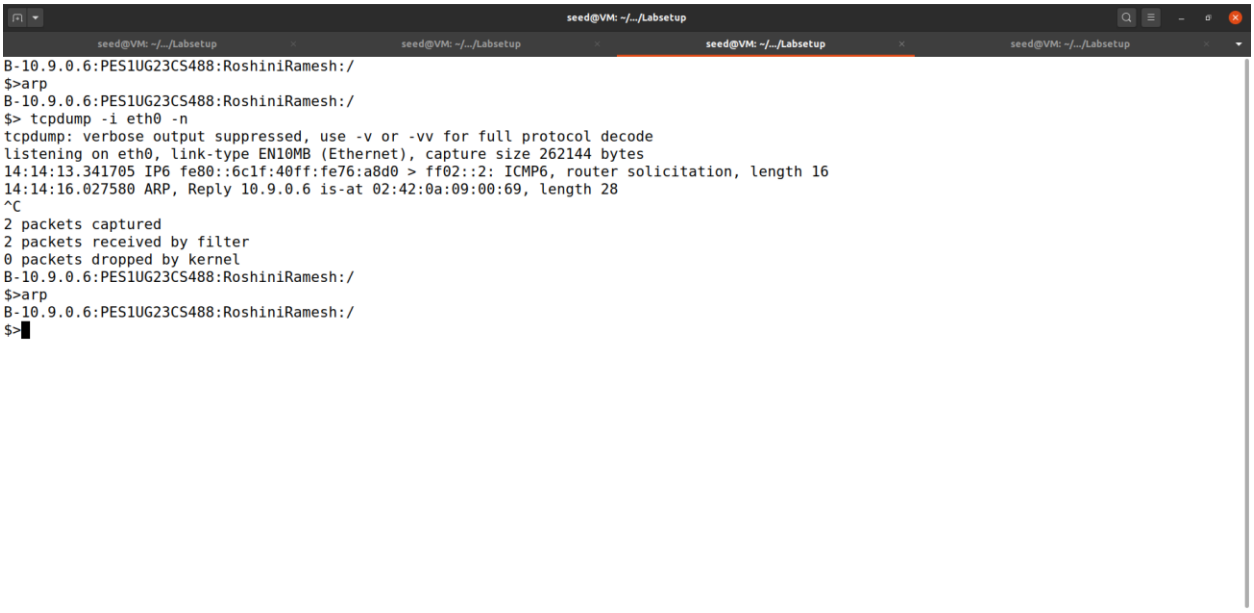
Here, we are just executing task11A.py like we did the previous time.

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

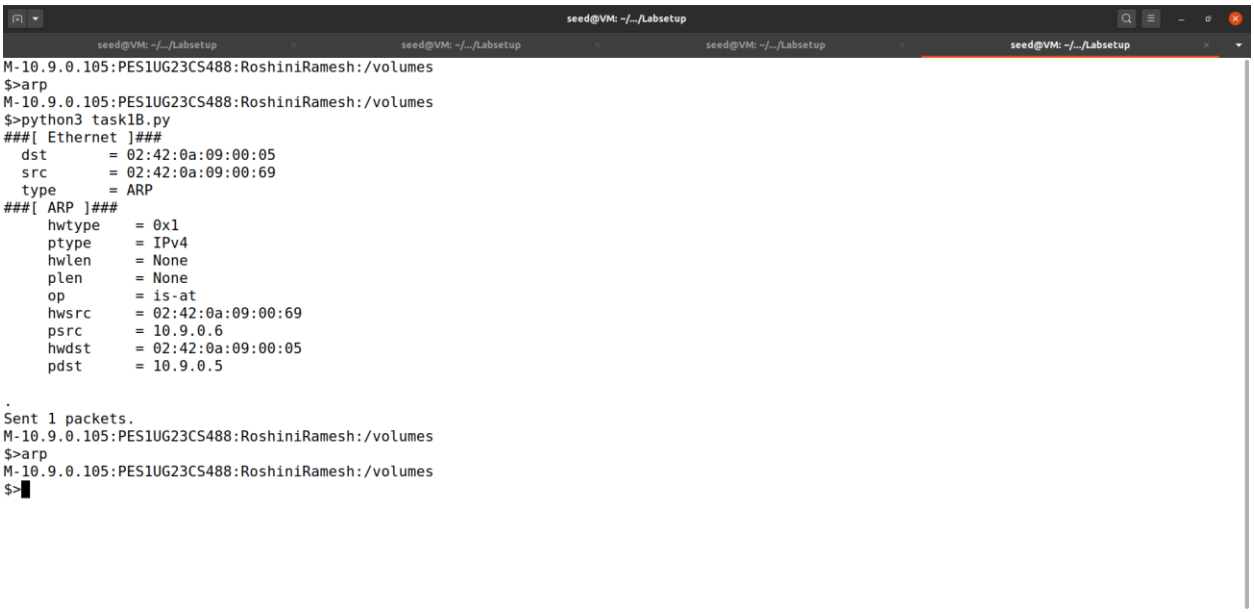
A-10.9.0.5 Terminal Output:



B-10.9.0.6 Terminal Output:



M-10.9.0.105 Terminal Output:

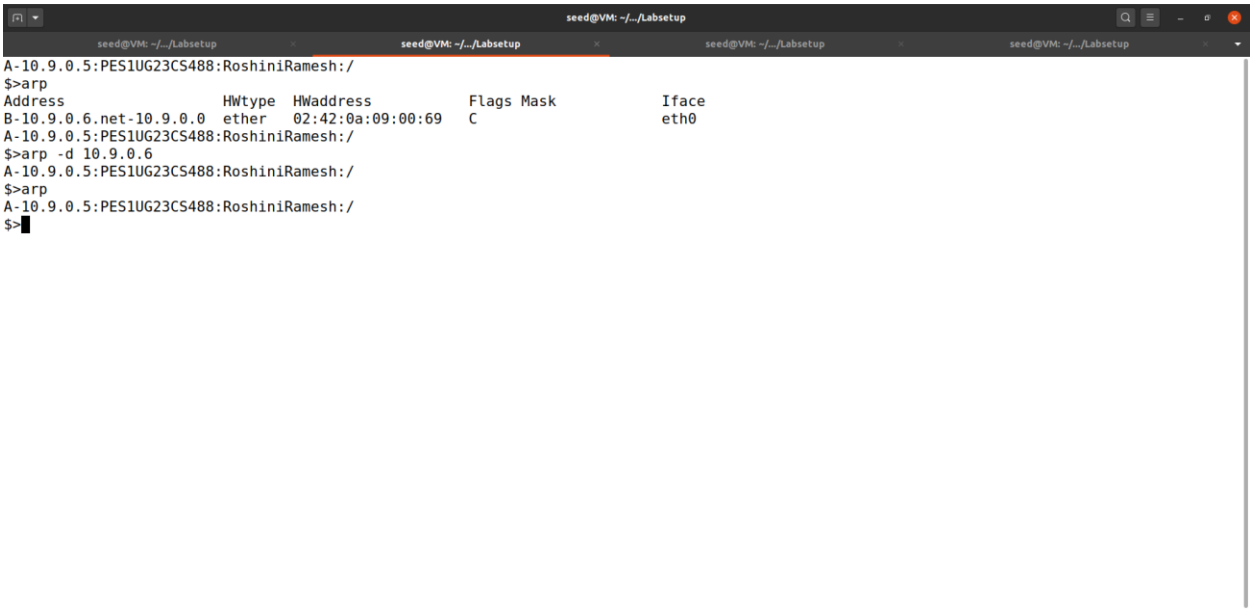


Explanation:

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

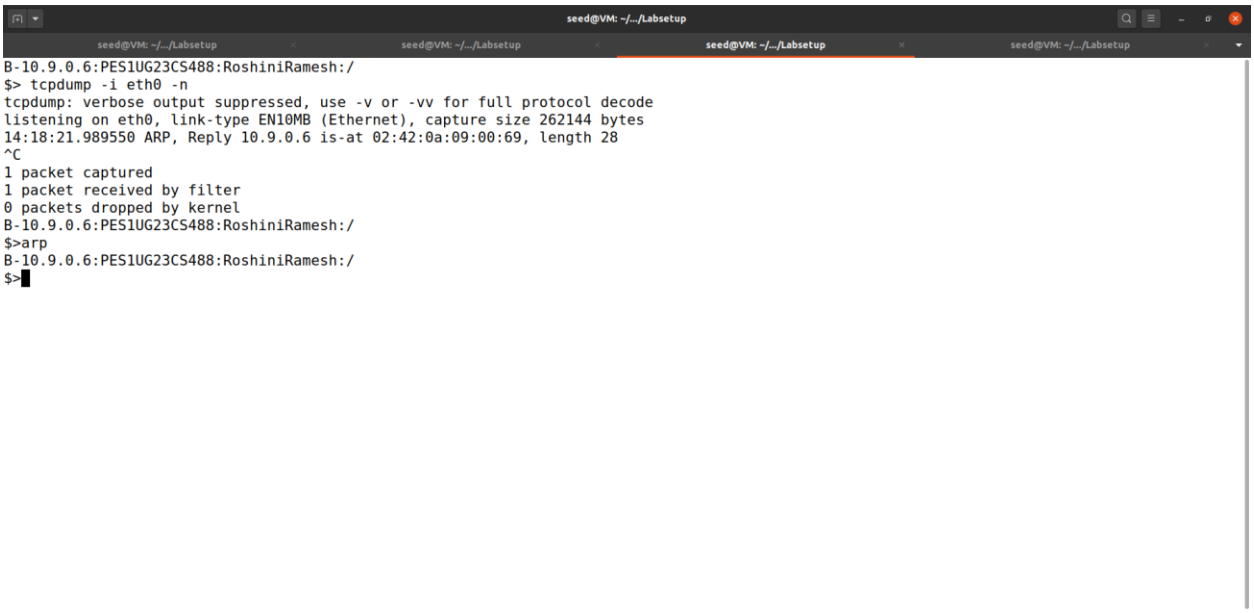
A-10.9.0.5 Terminal Output:



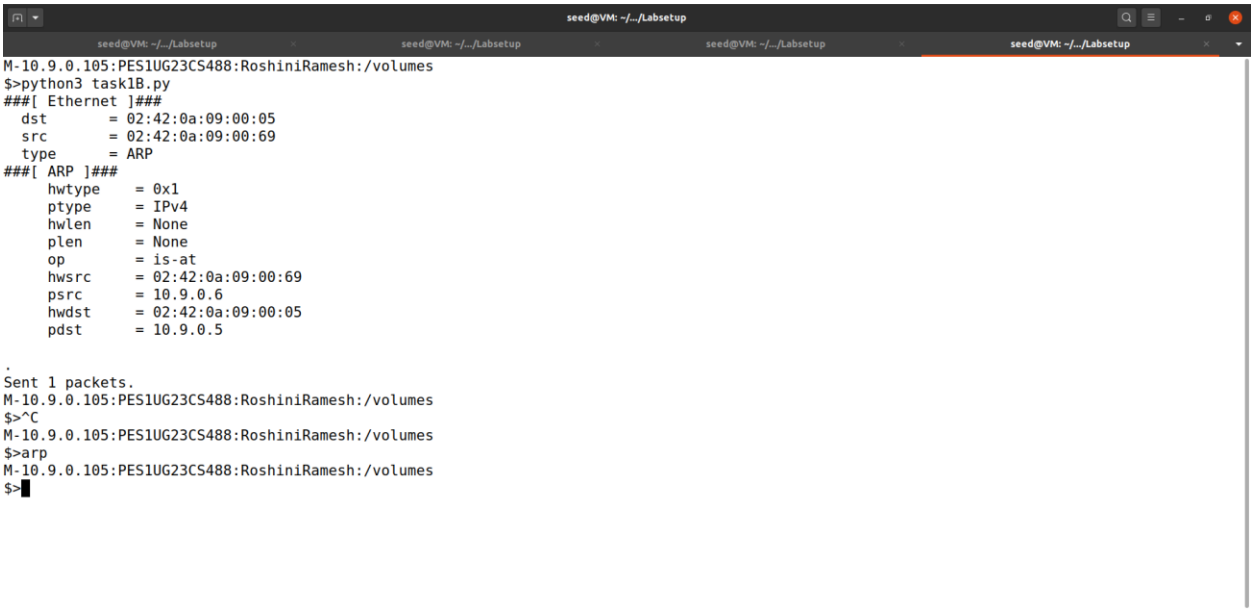
IF B'S IP IS NOT IN THE ARP TABLE

A-10.9.0.5 Terminal Output:

B-10.9.0.6 Terminal Output:



M-10.9.0.105 Terminal Output:



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.

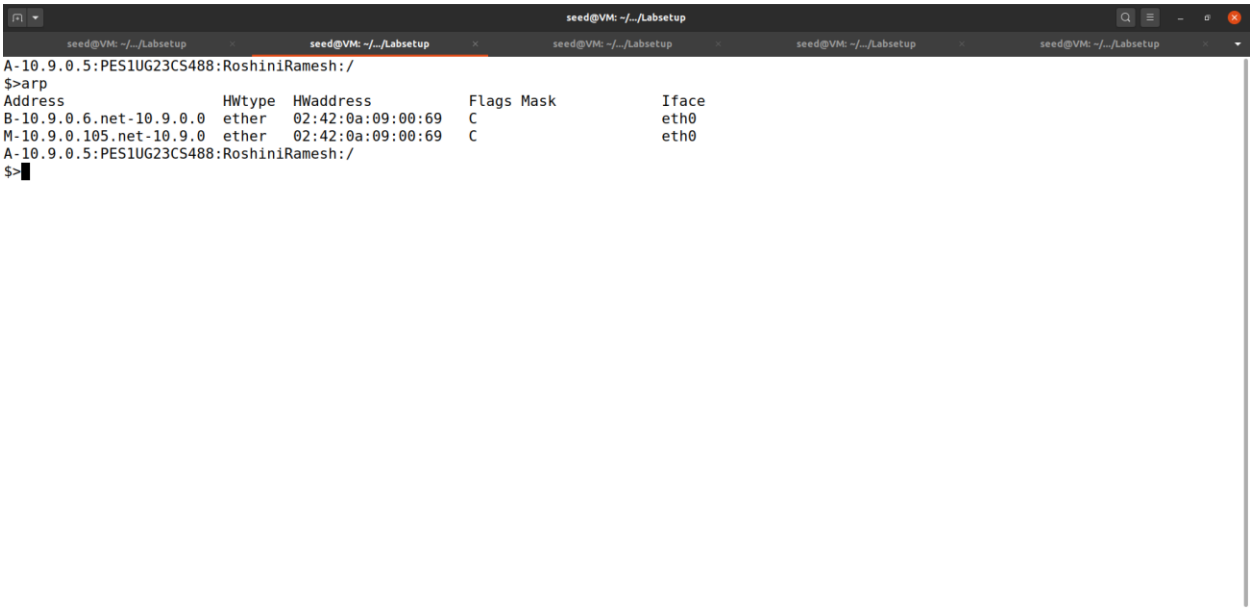
QUESTION 1: WHAT DOES OP=2 MEAN?

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

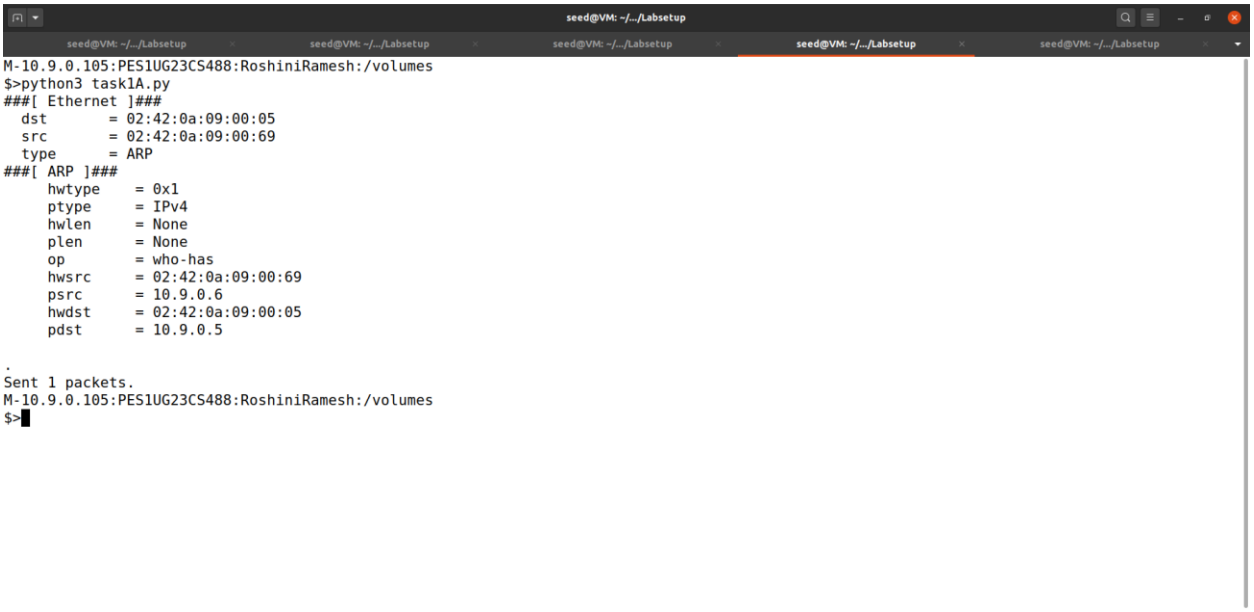
TASK 3: USING ARP GRATUITOUS MESSAGE

ADDING B’S IP TO A’S CACHE

A-10.9.0.5 Terminal Output:



M-10.9.0.105 Terminal Output:



Explanation:

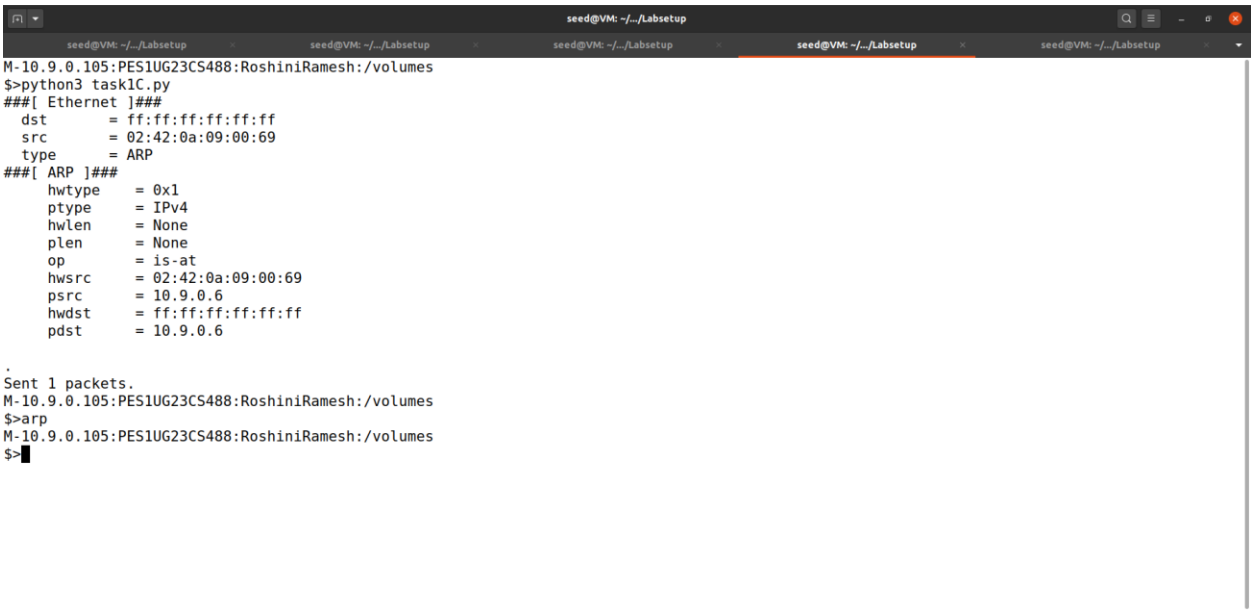
We are just populating the ARP table of A.

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

A-10.9.0.5 Terminal Output:

B-10.9.0.6 Terminal Output:

M-10.9.0.105 Terminal Output:



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.

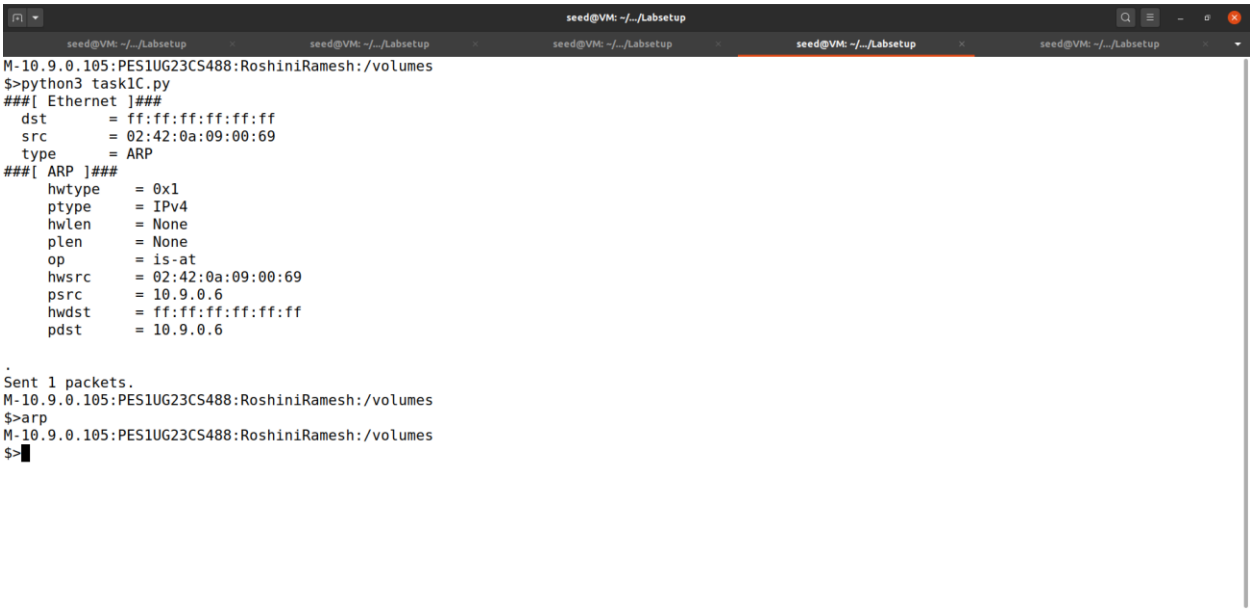
IF B'S IP ADDRESS DOESN'T EXIST IN A'S ARP TABLE

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

A-10.9.0.5 Terminal Output:

B-10.9.0.6 Terminal Output:

M-10.9.0.105 Terminal Output:



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?

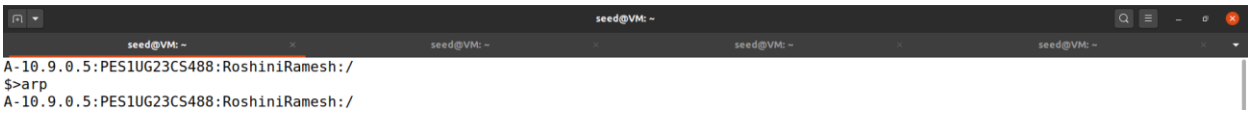
Host B's ARP Cache is unchanged because the machine won't change the ARP entry of its own IP and hence discards it. Other machines like Host A are tricked into poisoning their cache.

TASK 4: MITM ATTACK ON TELNET USING ARP CACHE POISONING

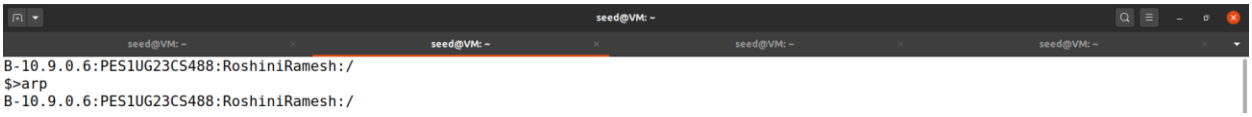
STEP 1 - LAUNCH THE ARP CACHE POISONING ATTACK

CHECK THE ARP CACHES OF HOST A AND HOST B

A-10.9.0.5 Terminal Output:



B-10.9.0.6 Terminal Output:

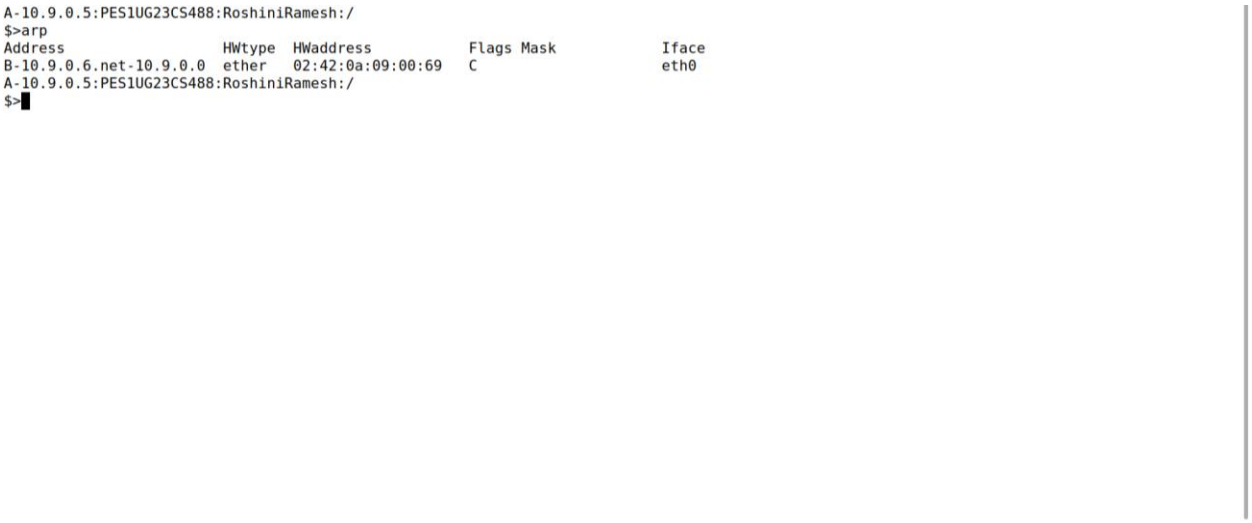


Explanation:

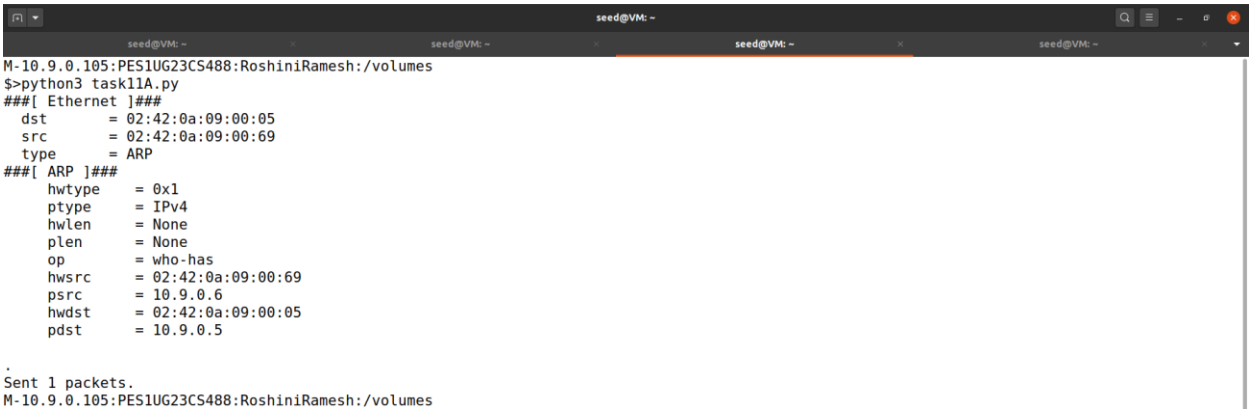
The ARP cache of both host A and host B has been cleared and it is empty.

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:



M-10.9.0.105 Terminal Output:

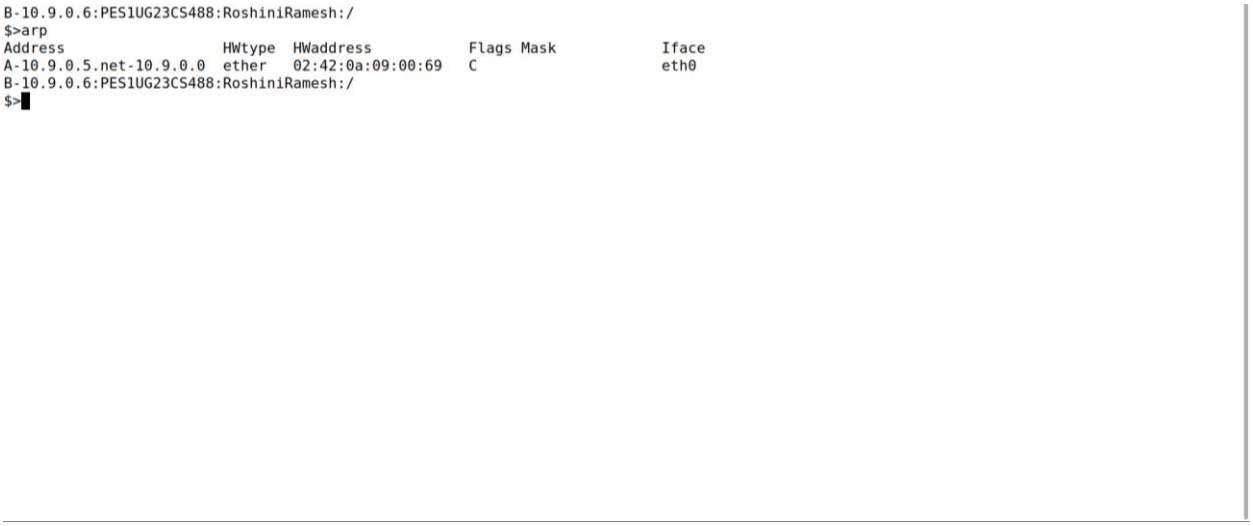


Explanation:

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:



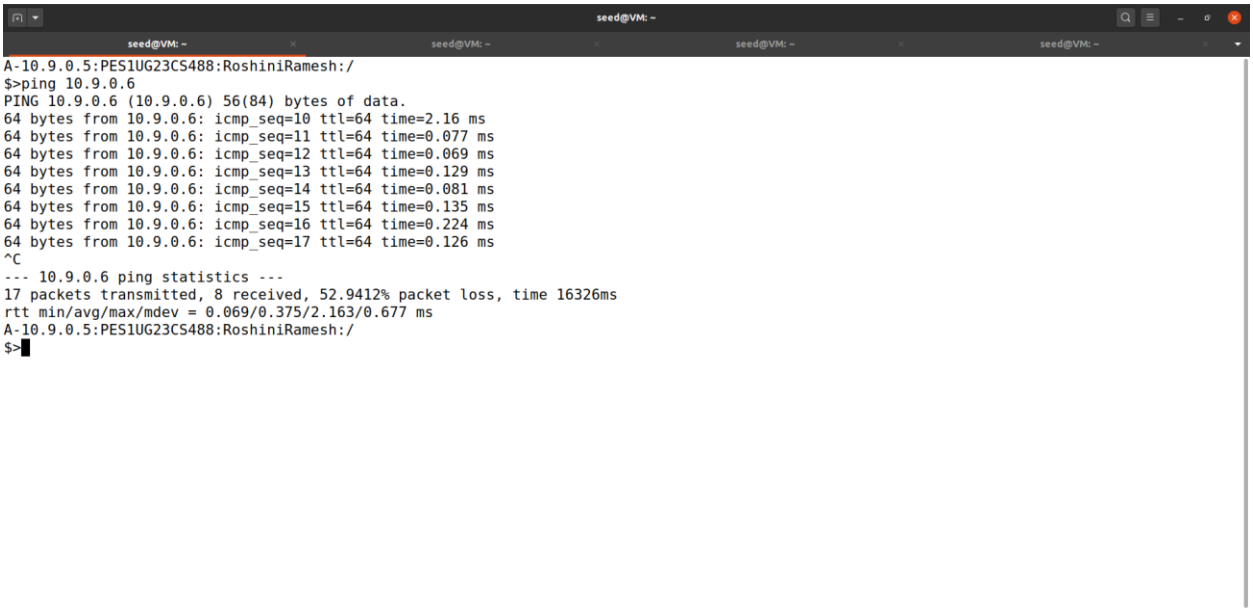
M-10.9.0.105 Terminal Output:

Explanation:

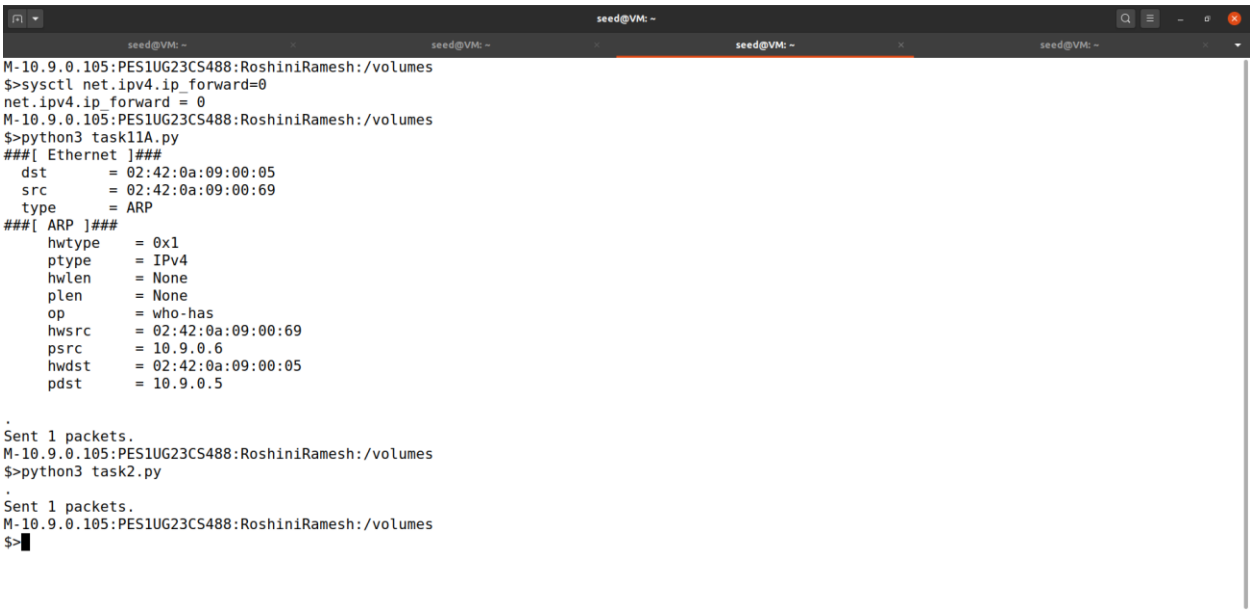
ARP cache of host B has been poisoned. In the ARP table, we can see that IP Address of host A is pointing to MAC address of attacker M.

STEP 2 – TESTING

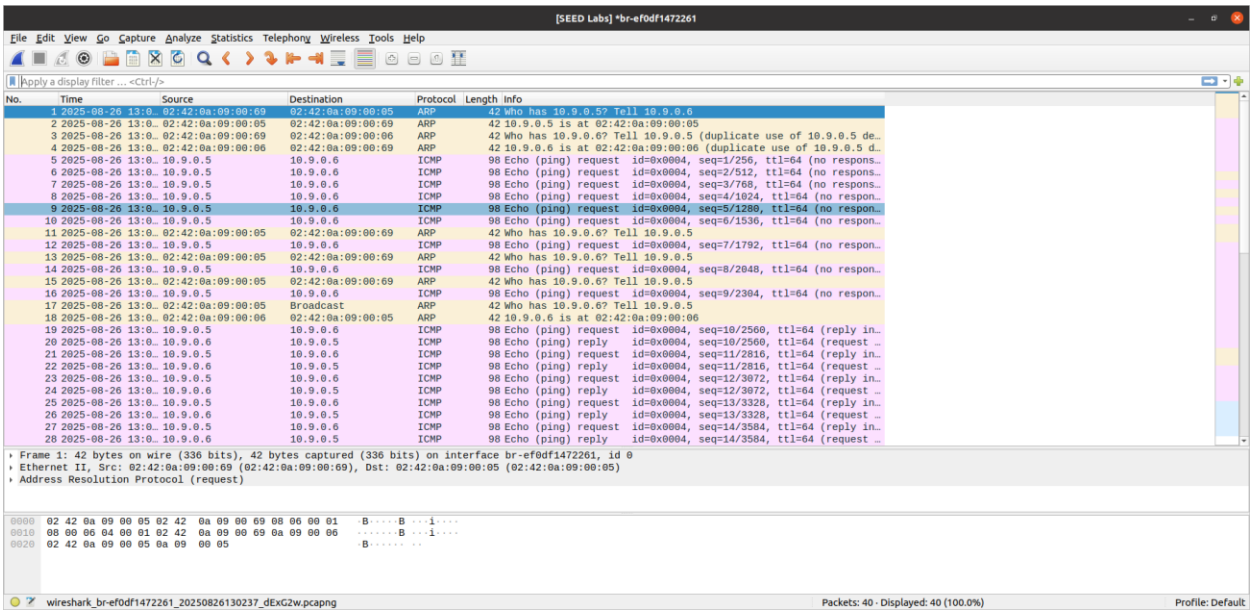
A-10.9.0.5 Terminal Output:



M-10.9.0.105 Terminal Output:



Wireshark Output:



Explanation:

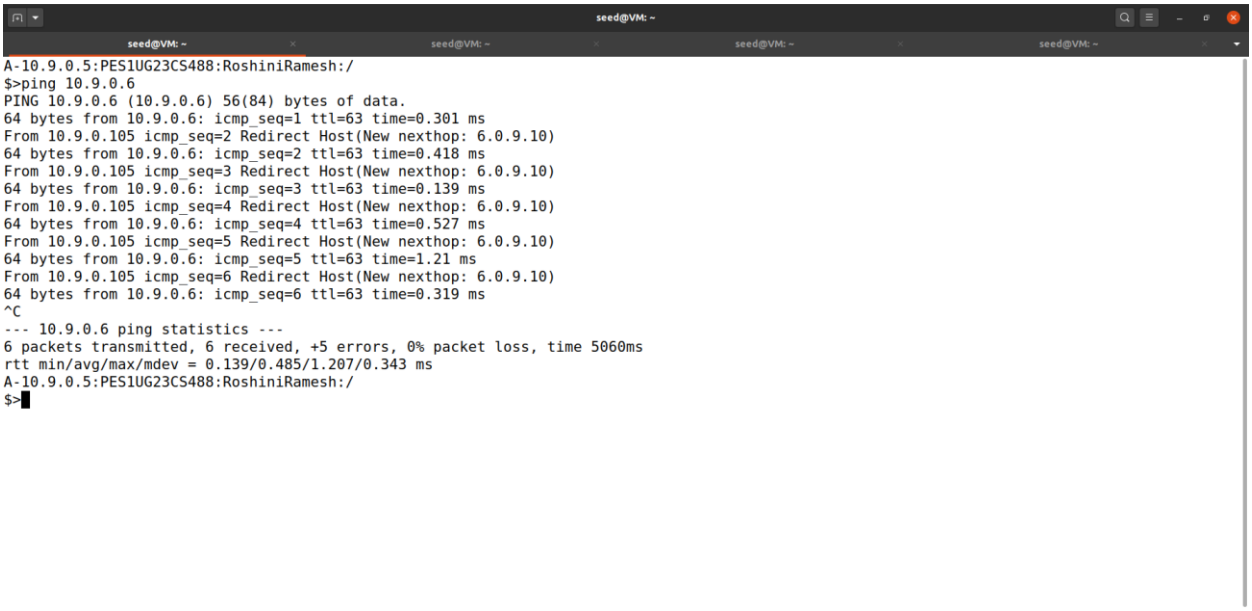
As the ARP table of A has been poisoned, B's IP address points to M's MAC address. As we have turned off IP forwarding on M, the packets are dropped. Eventually, due to multiple failed packet transfers, eventually, A sends a broadcast message asking who has the IP address 10.9.0.6 and corrects its ARP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

QUESTION 1: WHAT DO YOU OBSERVE? EXPLAIN

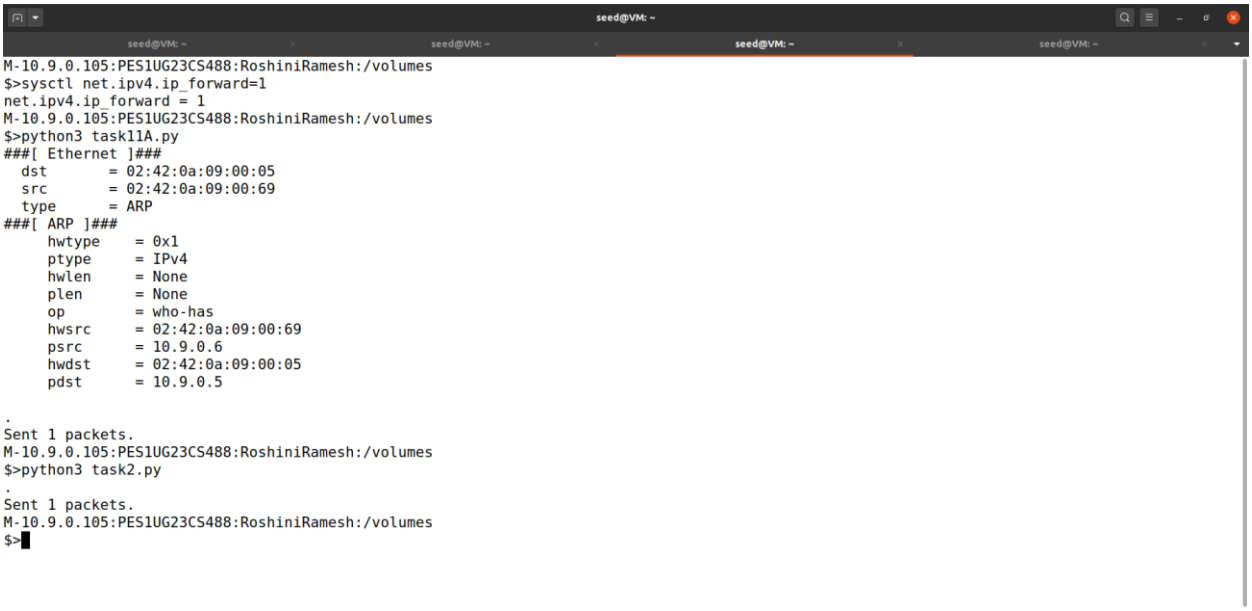
As the ARP table of A has been poisoned, B's IP address points to M's MAC address. As we have turned off IP forwarding on M, the packets are dropped. Eventually, due to multiple failed packet transfers, eventually, A sends a broadcast message asking who has the IP address 10.9.0.6 and corrects its ARP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

STEP 3 – TURN ON IP FORWARDING

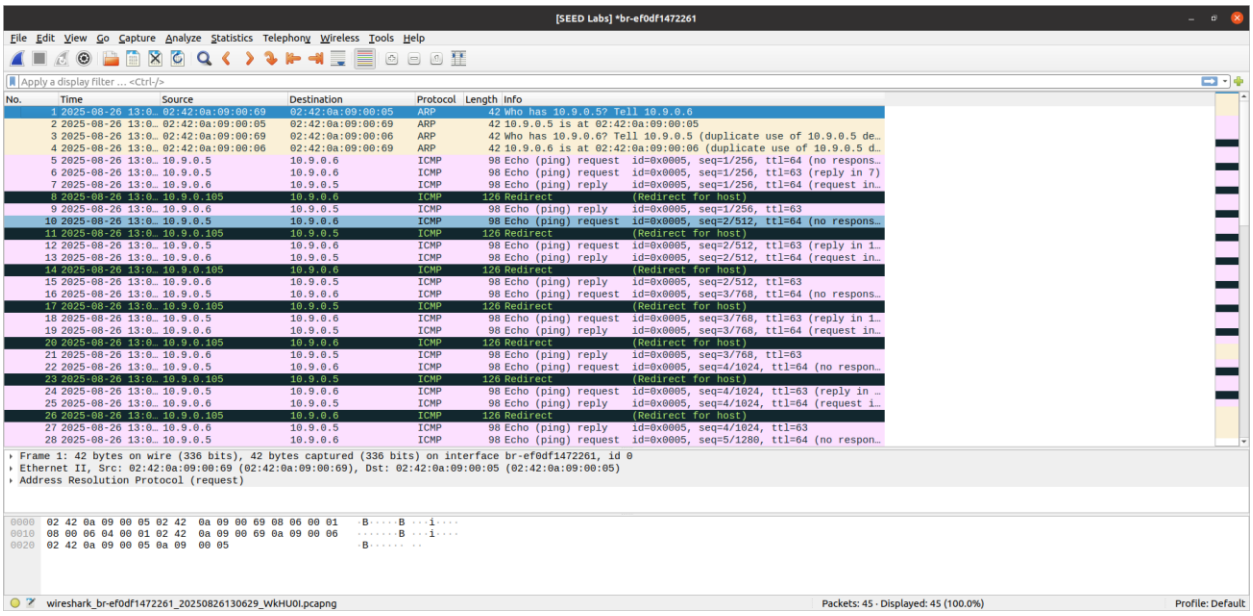
A-10.9.0.5 Terminal Output:



M-10.9.0.105 Terminal Output:



Wireshark Output:

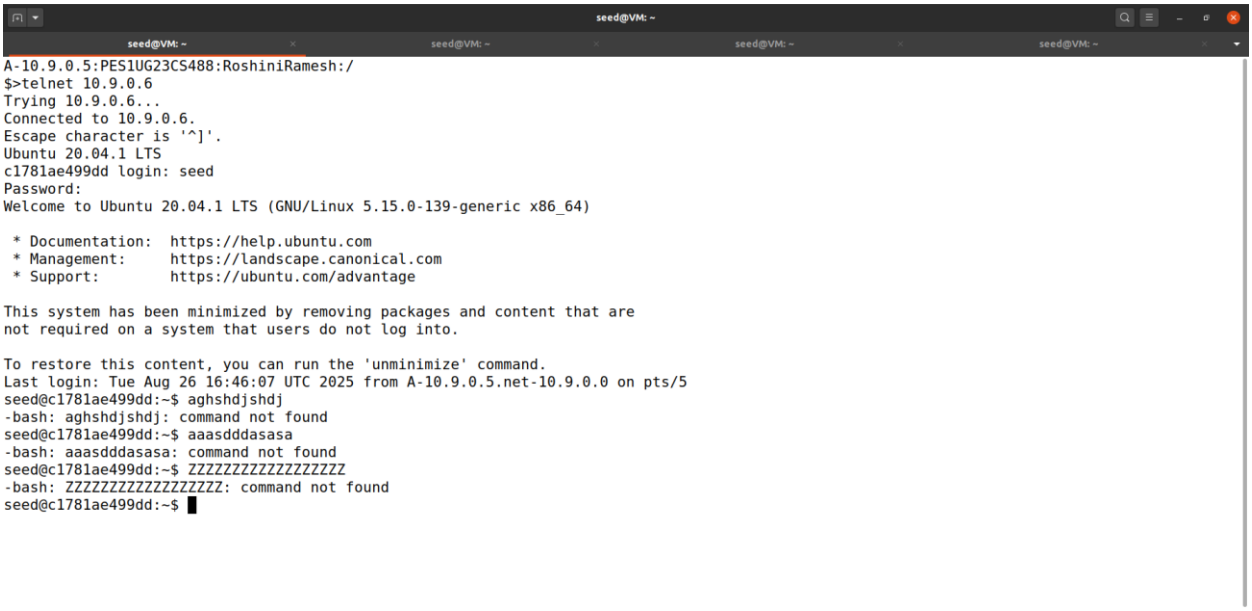


Explanation:

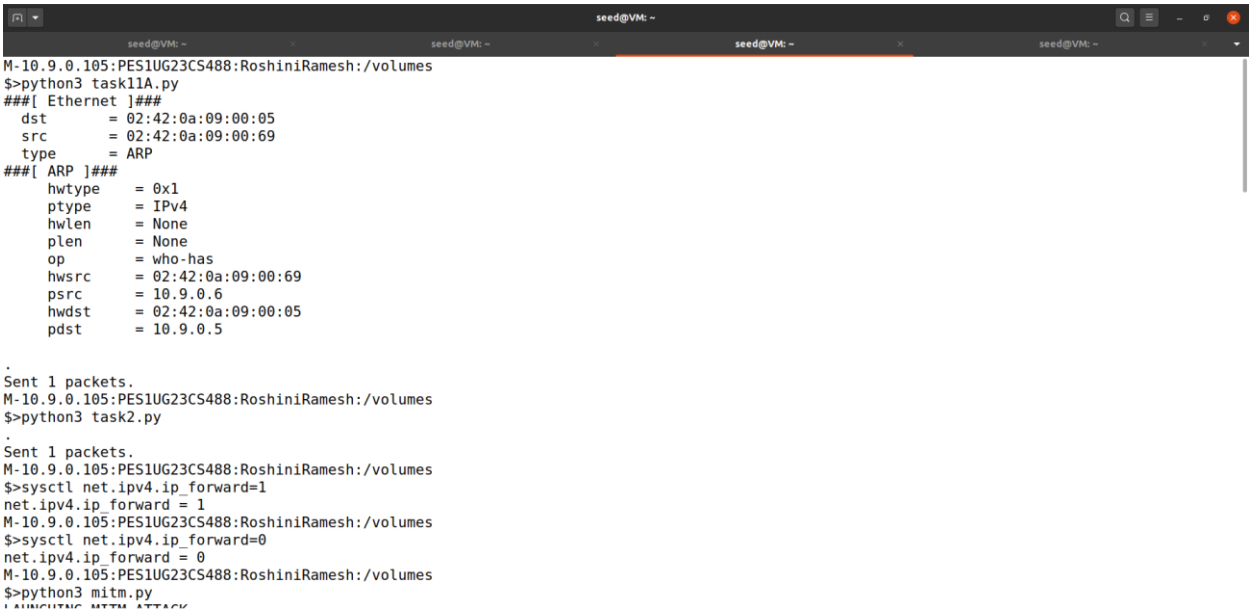
As the ARP table of A has been poisoned, B's IP address points to M's MAC address. As we have turned on IP forwarding on M, the packets are now forwarded to M. This can be seen in both the Wireshark output as well as the terminal output of A. There are redirect lines in both which proves this.

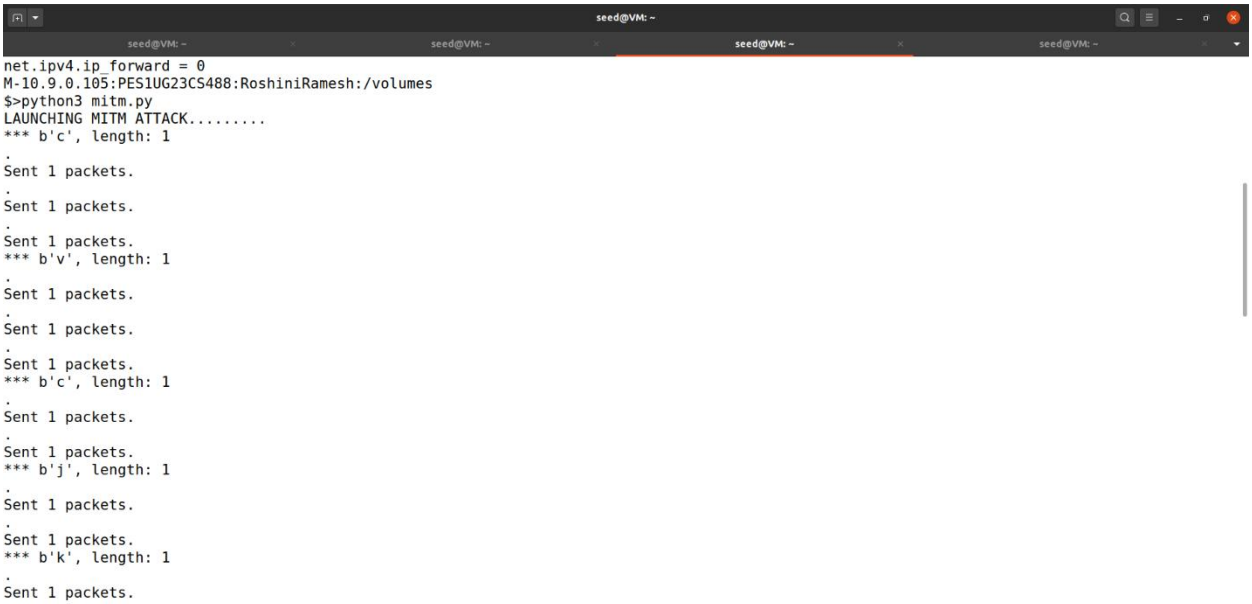
STEP 4 – LAUNCH THE MITM ATTACK

A-10.9.0.5 Terminal Output:

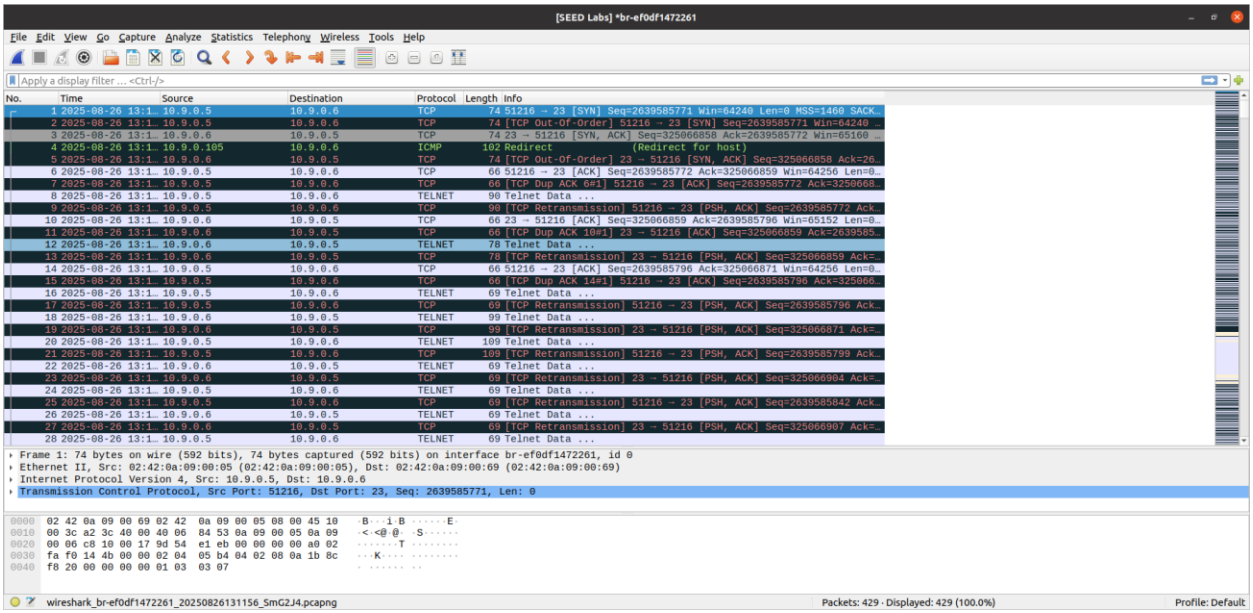


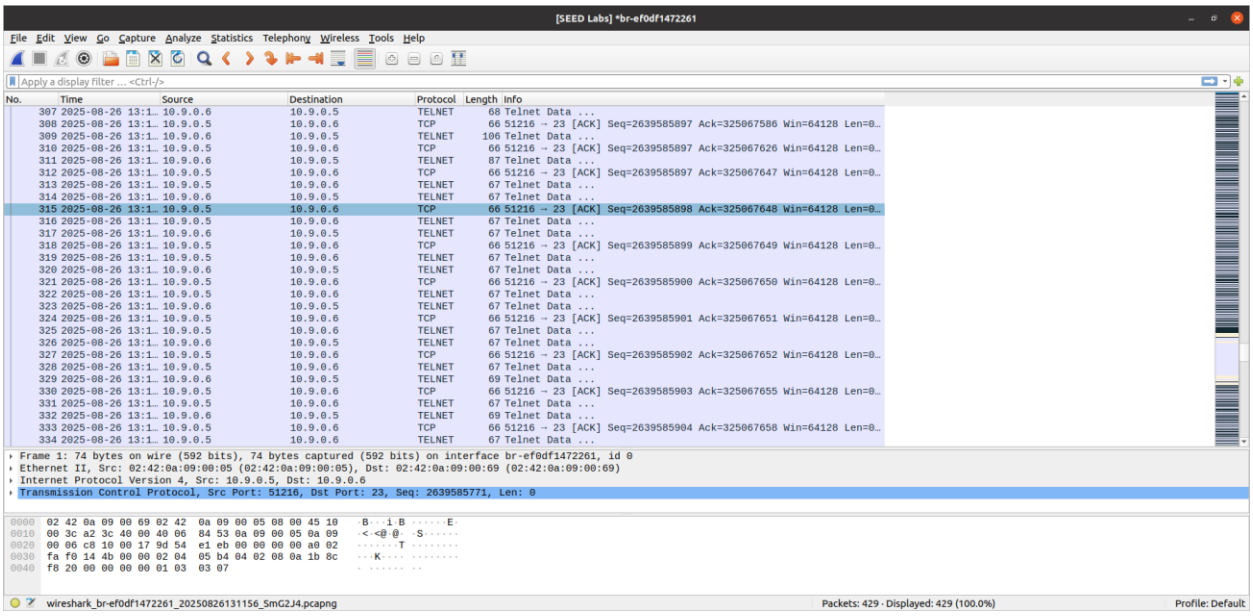
M-10.9.0.105 Terminal Output:

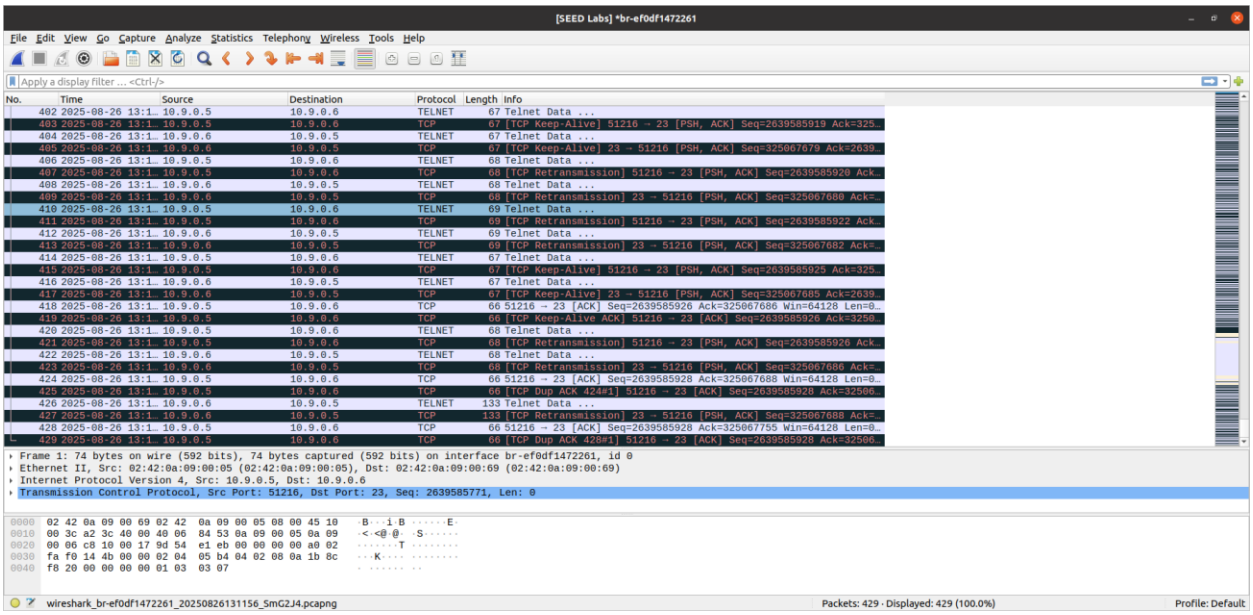




Wireshark Output:





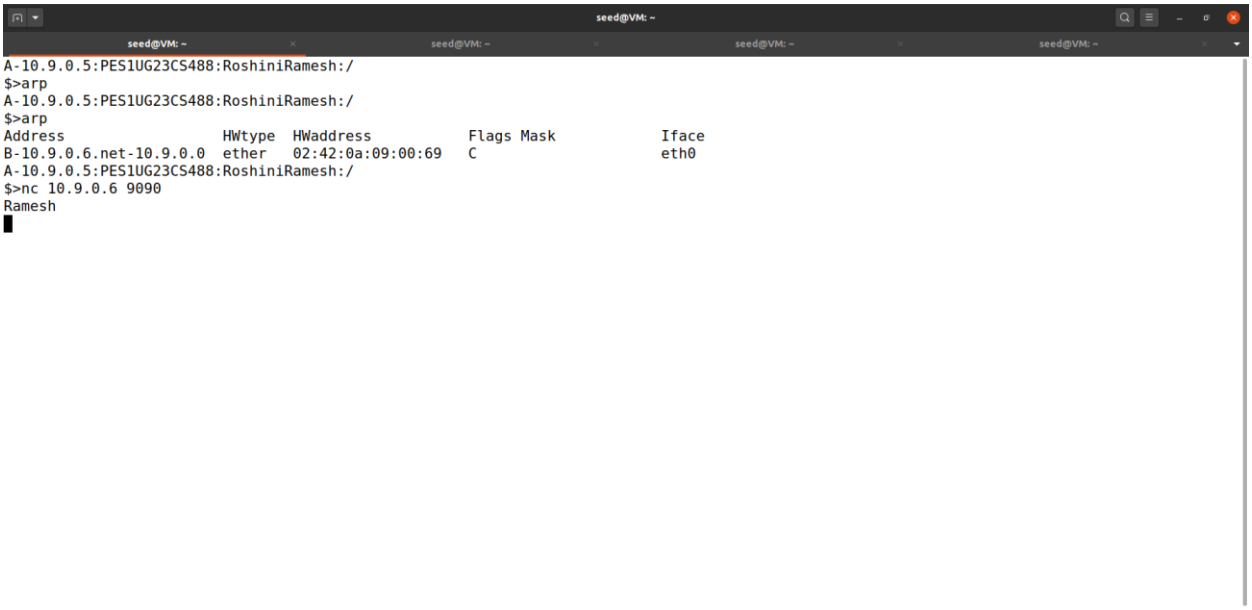


Explanation:

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.

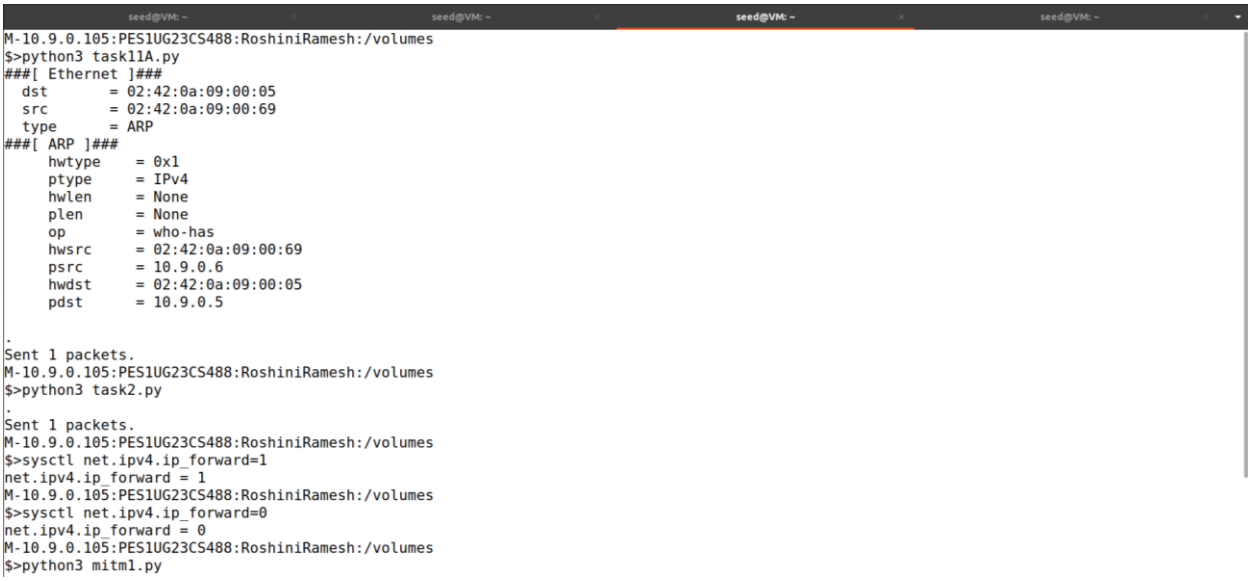
TASK 5: MITM ATTACK ON NETCAT USING ARP CACHE POISONING

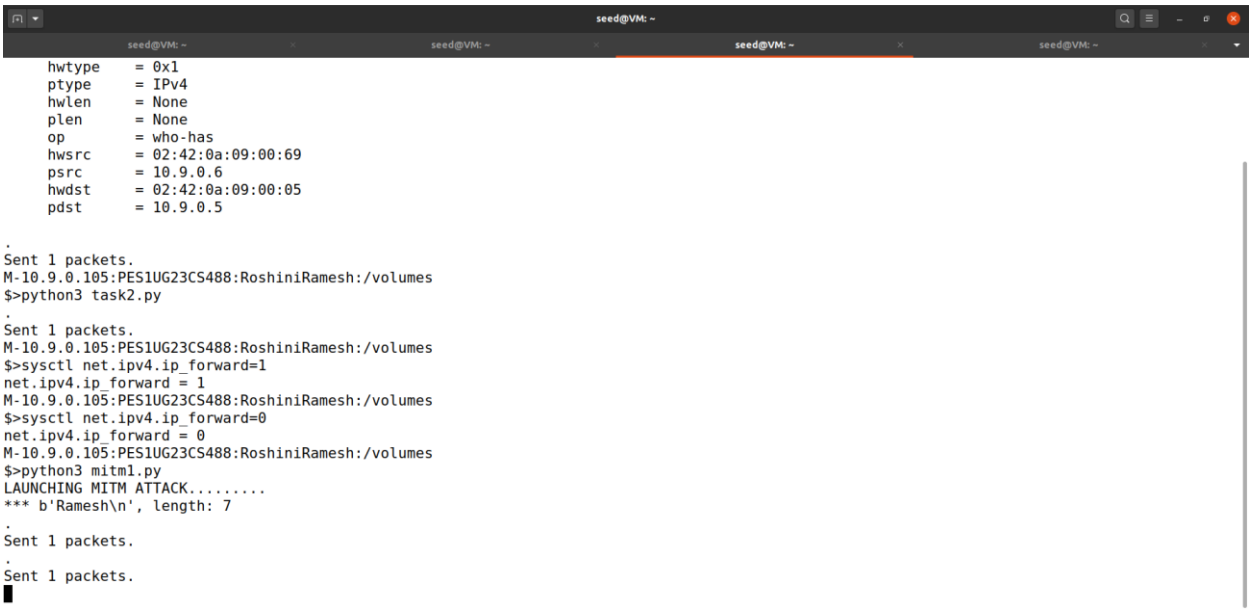
A-10.9.0.5 Terminal Output:



B-10.9.0.6 Terminal Output:

M-10.9.0.105 Terminal Output:





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.