**Comp434 – Project 4**

**ARP Cache Poisoning and MITM**

Ertan Can Güner

60610

**Pledge of Honor**



“I hereby declare that I have completed this individually, without support from anyone else. I hereby accept that only the below listed sources are approved to be used:

1. Course textbook,
2. All material that is made available to me via Blackboard for this course,
3. Notes taken by me during lectures.



I have not used, accessed or taken any unpermitted information from any other source. Hence, all effort belongs to me.”

60610, Ertan Can Güner, 18.05.2023



|  |  |  |
| --- | --- | --- |
|  | **IP** | **MAC** |
| **M–68cbaf3b8f9b** | 10.9.0.105 | 02:42:0a:09:00:69 |
| **A-b2316eb539a5** | 10.9.0.5 | 02:42:0a:09:00:05 |
| **B-1d95d55fefe1** | 10.9.0.6 | 02:42:0a:09:00:06 |
| **VM** | 10.9.0.1 | 02:42:66:d1:bc:69 |

**\*Following table is the IP-MAC mapping of our virtual environment**

**A picture containing text, software, screenshot, web page

Description automatically generatedTask 1.A**

Source Code 1 : ARP Request

Our first task is to use ARP Request to poison A’s ARP table to map the Victim B’s IP address to M’s MAC address. Normally, ARP Requests are broadcasted but we only want to poison A’s table, so we send it directly to A. We achieve this by setting the Ethernet Header source and destination as M’s MAC and A’s MAC address respectively. Then we fill ARP headers, the destination MAC and IP from A’s MAC and IP addresses but for source MAC and IP, we fill it with M’s MAC address and B’s IP address. We set the op field as 1 indicating this is a ARP Request.

**A screenshot of a computer

Description automatically generated with medium confidence**

Demo 1 : ARP Poisoning w/ ARP Request

First, we check the ARP tables in A and B, as it can be seen above, there are no entries in both tables. After we send the forged ARP Request, we check the tables in A and B, and as expected we see an entry in A’s table mapping M’s MAC address to B’s IP.

**Task 1.B**

**A picture containing text, software, screenshot, web page

Description automatically generated**

Source Code 2 : ARP Reply

Our second task is to use ARP Reply to poison A’s ARP table to map the Victim B’s IP address to M’s MAC address. Source code is same as in Task1.A, only difference is we change the ‘op’ field in ARP header to 2 indicating that it’s an ARP Reply.

A screenshot of a computer

Description automatically generated with medium confidence

Demo 2 : Scenario 1- ARP Poisoning w/ ARP Reply

A screenshot of a computer

Description automatically generated with medium confidence

Demo 3 : Scenario 2 - ARP Poisoning w/ ARP Reply

We try to poison the A’s ARP table in 2 scenarios. First one is: Poisoning the table if there is entry for B. Second one is: Poisoning the table if there is no entry for B. For the first scenario, we check the ARP Tables in A and B. Then, we send our ARP Reply packet to A and check the ARP Table again. We see that the table is poisoned with M’s MAC address mapped to B’s IP. For the second scenario, we clear the ARP Table in A and send our ARP Reply packet to A and check the ARP Table again. We observe that there is no entry, so our poison attack has failed.

**Task 1.C**

**A screenshot of a computer

Description automatically generated with medium confidence**

Source Code 3 : ARP Gratuitous

Our third task is to use ARP Gratuitous to poison A’s ARP table to map the Victim B’s IP address to M’s MAC address. Source code is same as Task 1.A, only difference is we remove the field ‘op’, and change the destination MAC address in Ethernet and ARP header to broadcast, which is “ff:ff:ff:ff:ff:ff”, and change the IP address’ in ARP header to our victims IP.

**A screenshot of a computer

Description automatically generated with medium confidence**

Demo 4 : Scenario 1 - ARP Poisoning w/ ARP Gratuitous

**A screenshot of a computer

Description automatically generated with medium confidence**

Demo 5 : Scenario 2 - ARP Poisoning w/ ARP Gratuitous

We try to poison the A’s ARP table in 2 scenarios, same scenarios in Task 1.B. For the first scenario, we check the ARP Tables in A and B. Then, we send our ARP Reply packet to A and check the ARP Table again. We see that the table is poisoned with M’s MAC address mapped to B’s IP. For the second scenario, we clear the ARP Table in A and send our ARP Reply packet to A and check the ARP Table again. We observe that there is no entry, so our poison attack has failed.

**A screenshot of a computer

Description automatically generatedTask 2**

Source Code 4 : ARP Poisoning for MITM

In this task, we need to poison A and B’s ARP Tables to launch a Man in the Middle Attack. From previous tasks, we constructed two ARP Requests destined for A and B to redirect their traffic to machine M. We send the poison again every 2 seconds so that A and B doesn’t update their ARP tables on their own.

**A screenshot of a computer

Description automatically generated**

Demo 6 : MITM Attack w/o IP Forwarding

We test the attack after we turn off the IP Forwarding on machine M. We stopped the ARP Poisoning after icmp\_seq 30, so after that machine A communicates with machine B. We analyze the packets sent from machine A and we see that ping requests are destined for machine M, at icmp\_seq 10 and 20 our ARP poisoning probably missed the machines and A and B sent the ping request/reply to each other. Machine A doesn’t receive a reply from machine M even though it sends the ping requests to M. That’s because when M receives the ping request, the kernel compares the destination IP from the IP header to its own IP and if they are not equal, the kernel drops the packet. But this can be solved by our sniff-spoof methods to forge a ping reply.

A screenshot of a computer

Description automatically generated

Demo 7 : MITM Attack w/ IP Forwarding

We turn on IP Forwarding and run the ping command from A to ping B. We see that now A receives ping replies from B and their communication is flowing from M. After M receives ping request or reply from any of the machines it sends ICMP redirect packet to each host stating that their requests are forwarded to their destination.

A screenshot of a computer

Description automatically generated with medium confidence

Demo 8 : MITM Attack on Telnet

We first run the code in the figure *‘Source Code 4’* using *‘task2.py > temp.log &’* in the background to poison the ARP tables in 2 second intervals. Then we open the Telnet connection from A with IP Forwarding set to 1 on machine M. We observe that keys that we enter are displayed on the terminal but after we turn off the IP Forwarding the keys, we press are not displayed that’s because Telnet displays keys after it receives the response from the destination, B in our case. But we observe that even though our packets are not forwarded to B, there is TCP Keep-Alive packets sent to our attacker machine M.

A screenshot of a computer

Description automatically generated with medium confidence

Source Code 5 : MITM Attack Sniff-Spoof

We use the skeleton code that was provided in the pdf. Only changes in code are:

* Added an if statement to check if the packet is sent out from the attacker machine, if it is then we don’t do anything.
* Replaced data (key presses from A) with ‘Z’, but this can be changed to anything, or this data can be accumulated to gather credential information of the user.
* Changed the filter to only accept tcp packets and packets coming from A and B.

A screenshot of a computer

Description automatically generated with medium confidence

Demo 9 : MITM Attack

First, we turn on the IP Forwarding on machine M and create the telnet connection from A to B. After we established the connection, we test it by pressing ‘a’ key two times, and we see it, so the data flows through M without any problems. Then we turn off the IP Forwarding and run our MITM program. Then we press ‘a’ key again and we see that A receives ‘Z’ on its terminal. Our MITM attack has succeded.