Arp Spoofing and Detection with Python

Mahmud Jamil
ID: 2017-1-60-021
Computer Science & Engineering
East West University
Dhaka, Bangladesh
Email: 2017-2-60-147@std.ewubd

Farhat Jebin
ID: 2017-2-60-059
Computer Science & Engineering
East West University
Dhaka, Bangladesh
Email: 2017-2-60-059@std.ewubd

Md. Fahimul Islam
ID: 2017-1-62-022
Computer Science & Engineering
East West University
Dhaka, Bangladesh
Email: 2017-1-62-022@std.ewubd

Sakib Khan
ID: 2017-2-60-039
Computer Science & Engineering
East West University
Dhaka, Bangladesh
Email: 2017-2-60-039@std.ewubd

Anisur Rahman
Associate Professor
Computer Science & Engineering
East West University
Dhaka, Bangladesh
Email: anis@ewubd.edu

Abstract— ***Address Resolution Protocol (ARP) is a protocol that resolves IP addresses to Media Access Control (MAC) addresses for transmitting data. ARP spoofing is used to link an attacker's MAC to a legitimate network IP address so the attacker can receive data meant for the owner associated with that IP address. ARP spoofing is commonly used to steal or modify or we can say knowing data but can also be used in denial-of-service and man-in-the-middle attacks. After ARP spoofing attacker gets the MAC address and gets the real time ARP as well. In this paper, we have tried to show how we can do the ARP spoofing perfectly between two different pcs. Then we showed how we can detect the ARP spoofing as well with our developing python tools.***

Keywords—Feature, Target IP, gateway IP, Real Time ARP, Implementation, Result, Prevention & Detection, Result Analysis

I. INTRODUCTION

Spoofing is a very exciting and interesting area in computer networking. There are different types of spoofing. ARP spoofing is one of them. In our modern arena, most organizations implement LAN for their communication and networking needs. In LANs, the identifier used for communication is MAC address. Thus the transfer of packets requires resolving IP address to MAC address for communication within a LAN. This resolution is done by the Address Resolution Protocol (ARP). ARP which stands for Address Resolution Protocol is a protocol used on networks to establish a device's MAC address, and their IP address. ARP Spoofing is the technique of redirecting the network traffic to the hacker by hacking the IP address.

ARP which stands for Address Resolution Protocol is a protocol used on networks to establish a device's MAC address, and their IP address. ARP Spoofing is the technique of redirecting the network traffic to the hacker by hacking the IP address. If we want to make it simple then we can say that ARP spoofing is a method of positioning ourselves between a target and the gateway on local network traffic. From there we can do naughty things like snooping on traffic, modifying packets as they pass by, or outright performing a denial-of-service attack. We draw a diagram to make it simpler to understand ARP spoofing.

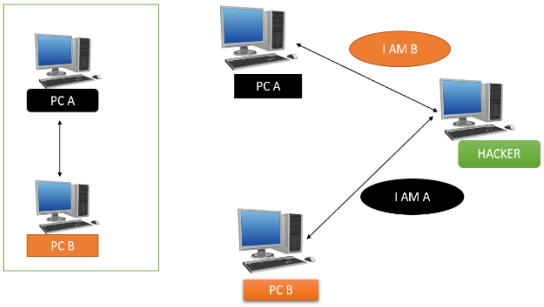


Figure: ARP Spoofing

In this diagram, we can see how ARP spoofing works. Here, the hacker acts on PC A that which is PC B, and to PC B hacker act as PC A. Using this technique, the hacker sent a malicious ARP then ping with that target PC's IP and gets a MAC address as well. Though it's happening regularly in daily life by hackers It has prevention or we can say detection method as well. Let's discuss the detection process of ARP spoofing. There are several approaches to preventing ARP Poisoning attacks. Like, static ARP Tables, switch Security, physical Security, network isolation, encryption. Now we discuss these processes.

In Static ARP Tables: It's possible to statically map all the MAC addresses in a network to their rightful IP addresses. Any change to the network will require manual updates of the ARP tables across all hosts, making static ARP tables unfeasible for most larger organizations. Still, in situations where security is crucial, where static ARP tables are used can help to protect critical information.

Physical Security: ARP messages are not routed beyond the boundaries of the local network, so would-be attackers must be in physical proximity to the victim network or already have control of a machine on the network.

Network Isolation: As stated previously, ARP messages don't travel beyond the local subnet. This means that a well-segmented network may be less susceptible to ARP cache poisoning overall, as an attack in one subnet cannot impact devices in another. Concentrating important resources in a dedicated network segment where enhanced security is present can greatly diminish the potential impact of an ARP Poisoning attack.

Avoid trust relationships: Organizations should develop protocols that rely on trust relationships as little as possible. Trust relationships rely only on IP addresses for authentication, making it significantly easier for attackers to run ARP spoofing attacks when they are in place.

Encryption: While encryption won't prevent an ARP attack from occurring, it can moderate the potential damage. A popular use of Man in the Middle attacks was to capture login that was once commonly transmitted in plain text. With the use of TLS encryption on the web, this type of attack has become more difficult. The threat actor can still intercept the traffic, but can't do anything with it in its encrypted form.

II. PROBLEM STATEMENT

From a computer Networking point of view, we need to work on a different way and implement the various technique to complete the spoofing because there are lots of barriers we have faced to reach the goal and get the desired result. Though the process is unethical day by day its working process grows rapidly. We can see the recent advancement in spoofing by using several tools or software in modern technologies, which have introduced several new interesting aspects to the area. So, this is an exciting research area where we can emerge the technique of our developing tools for spoofing and get the real time ARP. Though spoofing is

happened by hackers easily at the same time, it has prevention techniques as well like, static ARP Tables, switch Security, physical Security, network isolation, encryption is the key approach for prevention. The main goal of applying our developing python tools in this criteria is to computerize for the ARP spoofing and its detection process.

III. METHODOLOGY

A. Modeling Process

For building up a standard model tool, we need a gateway and targeted IP from two different PC's Using this gateway and targeted if we have to find target PC's resolving MAC address and real time ARP instantly. Here's the flow chart of the process.

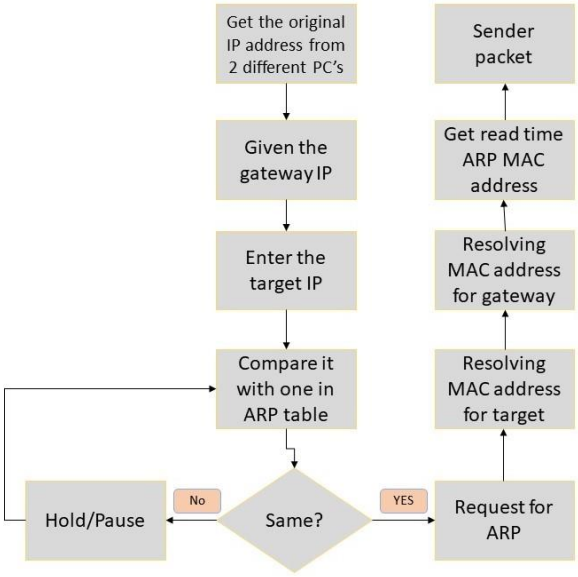


Figure: flow chart for ARP spoofing process.

For spoofing detection, we also consider a flow chart to show the whole process of detection.

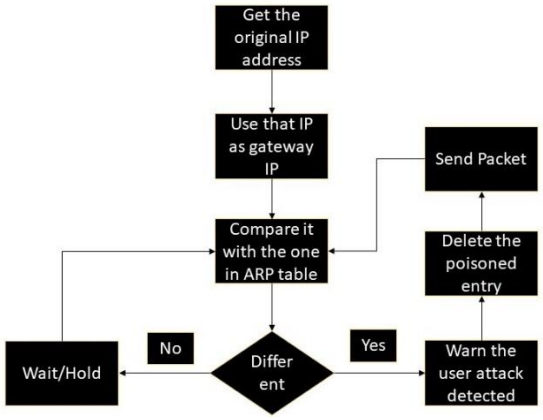


Figure: flow chart for ARP spoofing detection process.

B.Tools

- One PC's run with Windows 10 operating system for the target IP/gateway IP.
- One PC's run with Kali Linux for the gateway IP/target IP.
- PyCharm Ide for running our developing python code.
- Wireshark for seeing the real time ARP.
- Ettercap for sniffing the IP.
- We have used 'time' to handle time-related tasks.
- 'Scapy' is a library made in Python, with its command-line interpreter (CLI), which allows creating, modifying, sending, and capturing network packets.
- 'Termcolor' Color formatting for output in the terminal.
- 'sys' module in Python provides access to some variables used or maintained by the interpreter and some functions that interact strongly with the interpreter.

IV. APPLYING ALGORITHMS FOR SPOOFING

A. Using gateway Ip

The default gateway IP address is the private IP address refers to a device on a network that sends local network traffic to other networks.

```
gateway_ip = "192.168.189.134"
```

B. Using targeted IP

Target IP analysis filter to monitor a specific IP address that we suspect is the target of attacks. The IP address can be either internal or external. That means the target IP is which PC's IP we want to spoof to get our desired targeted MAC address.

```
┌──(root💀jamil)-[/home/jamil/PycharmProjects/spof]
└─# python3 spof.py
Enter the targetted ip: 192.168.189.129
```

C. MAC Address

MAC (Media Access Control address) is a unique number that is used to track a device in a network. MAC address provides a secure way to find senders or receivers in the network and helps prevent unwanted network access. MAC addresses are useful for network diagnosis because they never change, as opposed to a dynamic IP address, which can change from time to time.

```
def get_mac_from_ip(ip_address: str):
    # dst="ff:ff:ff:ff:ff:ff" broadcasts the request to the whole network
    ans = scapy.srp1(scapy.Ether(dst="ff:ff:ff:ff:ff:ff")/
                     scapy.ARP(pdst=ip_address, hwdst="ff:ff:ff:ff:ff:ff"),timeout=2,verbose=0,)
    if ans:
        return ans.hwsrc
    else:
        return None
def resolve_ip(name: str, ip_address: str):
    print(f"Resolving MAC address for {name} {target_ip}")
    # Resolve the target's MAC address
    mac = get_mac_from_ip(target_ip)
    if mac == None:
        print(f"Unable to resolve IP address. Exiting!")
        sys.exit(0)
    print(f"Resolved to {mac}")
    return mac
# Resolve the MAC addresses
target_mac = resolve_ip("target", target_ip)
gateway_mac = resolve_ip("gateway", gateway_ip)
```

V. ARP SPOOFING OUTPUT ANALYSIS

We used two PC IP addresses in our developing python code for ARP spoofing. One is windows another one is Kali Linux operating system. Firstly we used a Linux IP address for the gateway IP which is (192.168.189.134) and next we used the windows IP address for the target IP which is (192.168.189.129). In our code, we used two functions as well one is to get MAC to address from IP () and another one is resolved_ip().

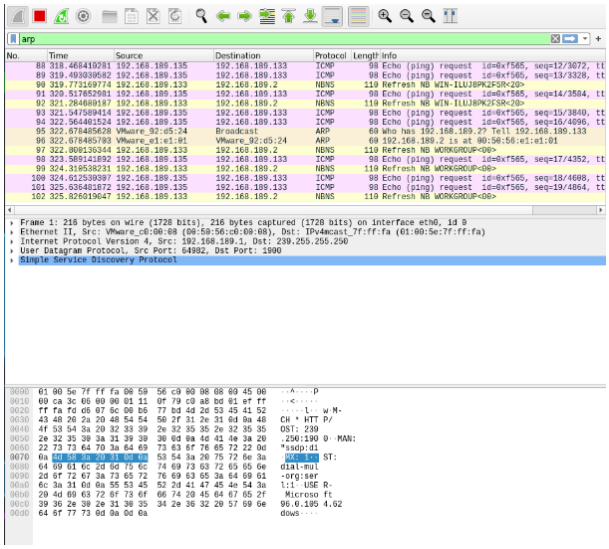
These two functions are used to get the target MAC address from our target pc and resolved the MAC address as well.

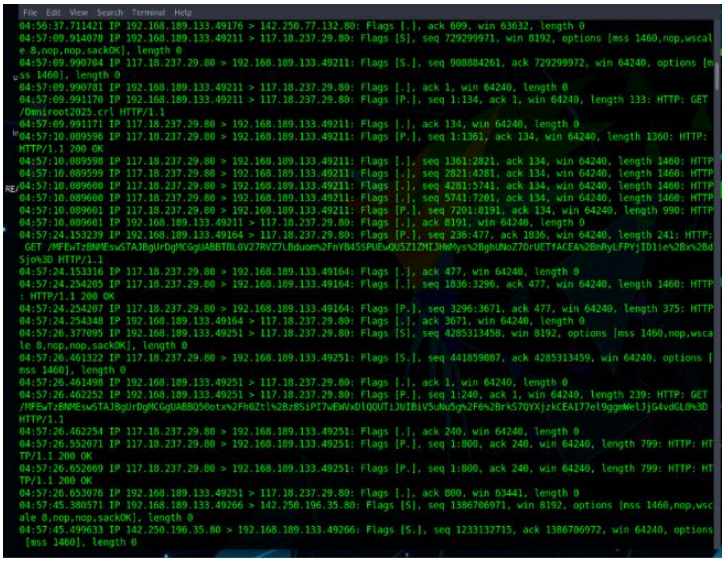
```
┌──(root💀jamil)-[/home/jamil/PycharmProjects/spof]
└─# sudo apt update
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.

┌──(root💀jamil)-[/home/jamil/PycharmProjects/spof]
└─# python3 spof.py
Enter the targetted ip: 192.168.189.129
Resolving MAC address for target 192.168.189.129
Resolved to 00:0c:29:92:d5:24
Resolving MAC address for gateway 192.168.189.129
Resolved to 00:0c:29:92:d5:24
net.ipv4.ip_forward = 1
Sending packets.......
```

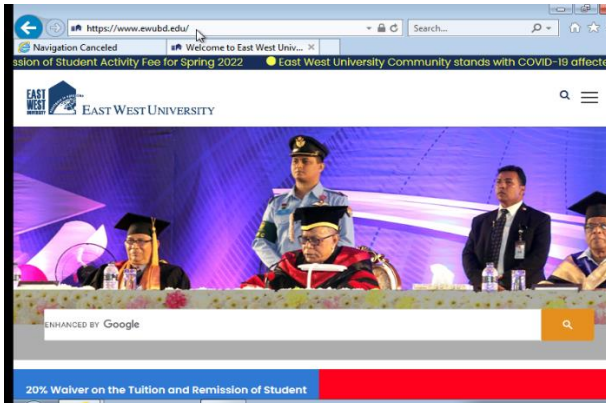
When we run the code and put the target IP which is 192.168.189.129. Our program returns the MAC address of this machine which is 00:0c:29:92:d5:24. Then we send packets to this victim machine.

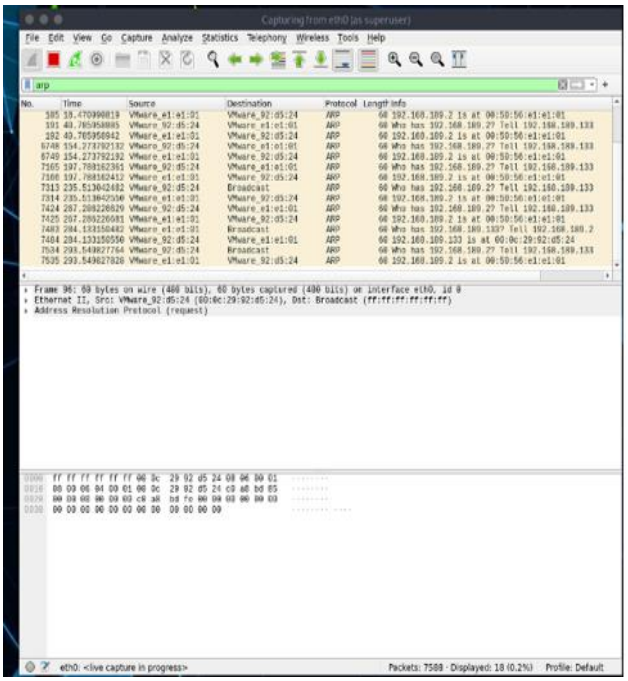
If we see Wireshark then we can see the ARP request and broadcast the real time as well.





If we want to see the real-time ARP and MAC address. We can consider an example like when the target (192.168.189.129) IP PC's used for browsing. For example, if the target PC's user browsed east west university and their service or other things. We got the real-time ARP request, MAC address, and IP address from that target PC.





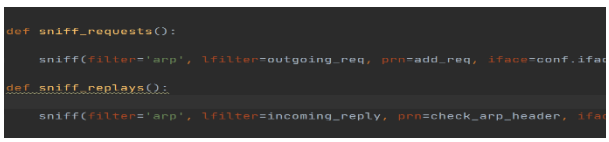
So, we got the real time ARP request, IP address, MAC address, and other information from the target (192.168.189.129) IP PC's.

So, ARP spoofing is done successfully between my PC's IP (192.168.189.134) and target (192.168.189.129) PC's IP.

VI. APPLYING ALGORITHMS FOR SPOOFING DETECTION AND ANALYSIS

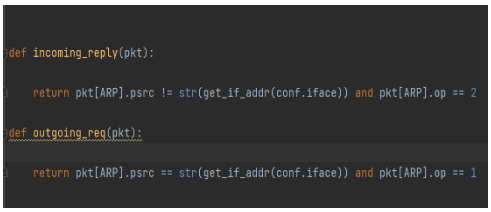
After successfully ARP spoofing now we are showing how to detect or we can say prevent ARP spoofing. The methods of detection use a passive approach, monitoring the ARP and looking for inconsistencies in the Ethernet to IP address mapping. The main concept of the passive approach is the time lag between learning and detecting spoofing. This sometimes leads to the attack being discovered long after it has been orchestrated. In this paper, we present an active technique to detect ARP spoofing. In ARP spoofing we saw that when we run the code and put the target IP. Our program returns the MAC address and real time ARP then sends packets to this victim machine. But in this detection system, we developed a module that when someone wants or tries to spoof a particular target PC's. If someone uses this tool or runs this module to check that pc is under attack or not. They can detect it and also got an alert notification when their pc is under attack. This technique is faster, more intelligent, and more reliable in detecting attacks than the passive methods.

A. Using sniff_request() and sniff_replays()



sniff_request() function is used for all ARP requests the pc made on the network and the sniff_replays() function is used to ARP replays the machine received from the network.

B. Using incoming_reply() and outgoing_req()



incoming_reply() is used to check if the packet is an incoming ARP reply and outgoing_req() is checked if the packet is an outgoing ARP request and check if that message is true or false.

C. Using spoof_detection()

If the reply is an answer for an ARP request message in full cycle, check if the source is genuine by sending a TCP SYN. If we don't receive a TCP ACK, we raise an alarm message. If we receive a TCP ACK, we add the IP and MAC pair to our IP_MAC_PAIRS table. If the message is an ARP reply without an ARP request message like a half cycle, send an ARP request for the IP of the source, thus causing the real owner of the IP on the network to respond with an ARP reply so we can treat it as a full cycle.

D. Using Winsound to get a sound alert.

```
ESIS_JAMIL > ARPDETECT > ARPDetect.py > ...
import subprocess
import time
import winsound

while True:

    choice = input("Would you like to test[T/t] or initialize[I/i] ARPDetect? ")

    if choice == "T" or choice == "t":
        print("ALERT! ARP POISONING DETECTED!")
        winsound.PlaySound("jamil", winsound.SND_FILENAME)

    elif choice == "I" or choice == "i":

        while True:
            subprocess.call("arp -a 192.168.43.1")
            get_output = subprocess.getoutput("arp -a 192.168.43.1")
            output_log = open("Logs.txt", "w")
            output_log.write(get_output)
            output_log.close()
            log = open("Logs.txt", "r")
            if log.mode == "r":
                contents = log.read()
                if "F0-D5-BF-DE-37-07" in contents:
                    print("ARP POISONING: FALSE")
                elif "F0-D5-BF-DE-37-07" != contents:
                    print("ARP POISONING DETECTED!")
                    winsound.PlaySound("jamil", winsound.SND_FILENAME)
                    break
            time.sleep(10)
```

Here, we used the target PC's IP as gateway IP (192.168.43.1). And when victim pc run this code they got the alert message and check their pc is under attack or not. After getting the notification an alert alarm rang instantly.

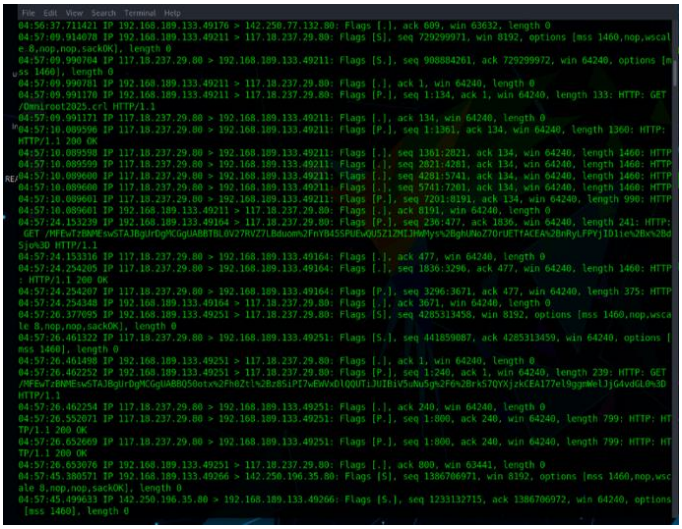
VII. Result Analysis and Discussion

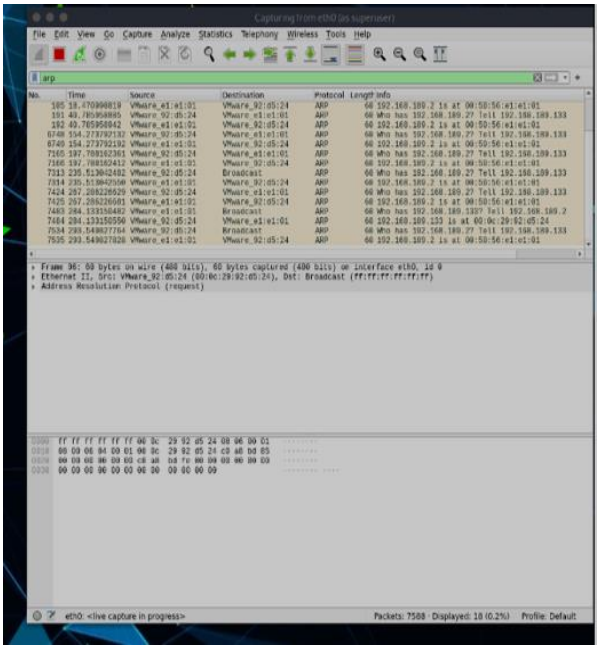
A. ARP spoofing

An ARP poisoning. When we run the code and put the target IP which is 192.168.189.129. Our program returns the MAC address from the targeted victim machine which is 00:0c:29:92:d5:24. Then sending packets from this victim machine.

```
┌──(root💀jamil)-[/home/jamil/PycharmProjects/spof]
└─# sudo apt update
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.

┌──(root💀jamil)-[/home/jamil/PycharmProjects/spof]
└─# python3 spof.py
Enter the targetted ip: 192.168.189.129
Resolving MAC address for target 192.168.189.129
Resolved to 00:0c:29:92:d5:24
Resolving MAC address for gateway 192.168.189.129
Resolved to 00:0c:29:92:d5:24
net.ipv4.ip_forward = 1
Sending packets.......
```

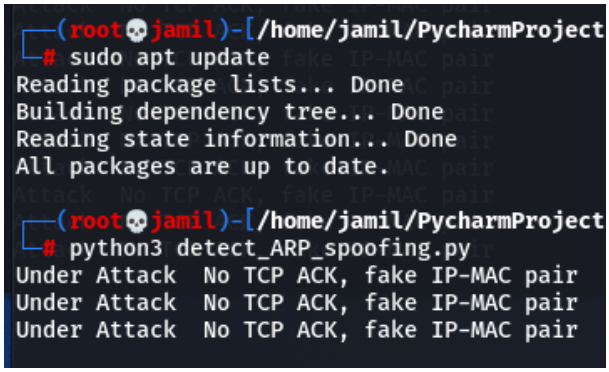


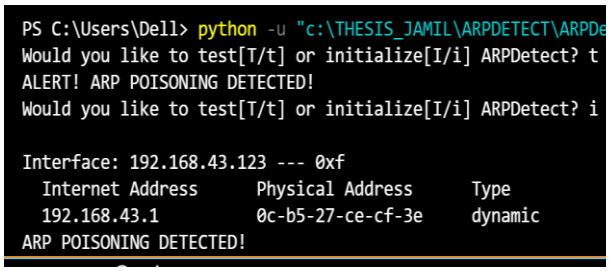


So, we got the real time ARP request, IP address, MAC address, and other information from a target(192.168.189.129) IP pc's.

So, ARP spoofing was done successfully between my PC's IP (192.168.189.134) and target(192.168.189.129) PC's IP.

B. ARP Spoofing Detection





When we input the gateway IP (192.168.43.123) in the victim machine and when the ARP spoofing detector program is successfully running then we got the detection message which pc is under attack and give us feedback with under attack, No TCP ACK, and fake pair message on the console. Besides this when ARP spoofing or poisoning is detected, the user gets a sound alert message also.

So, our developed model works properly for spoofing and its detection as well.

VIII. Difference features between other tools and our developing tools.

Now we are going to discuss the tools feature. And provide a table to show, why our tool is different from other tools. First, we discuss ARP spoofing part.

Tools Features	Other Tools	Our Tool
i)Reading Packet List	No	Yes
ii)Building Dependency Tree	No	Yes
iii)Reading State Information	No	Yes
iv)Get MAC address without using ether cap to sniff packets	No	Yes
v)Resolving MAC address for target	No	Yes
vi)Resolving MAC address for gateway IP	No	Yes
vii)Sending packets to target PC without using other software	No	Yes

We got the seven major differences between other tools and our developing tool in ARP spoofing. Now we discuss the difference between our developing tools and other tools in ARP the spoofing detection part.

Tools Features	Other Tools	Our Tool
i) Reading Packet List	**No**	**Yes**
ii)Building Dependency Tree	**No**	**Yes**
iii)Reading State Information	**No**	**Yes**
iv)Detect with an internet address	**No**	**Yes**
v)Detect with a physical address	**No**	**Yes**
vi)Detect with the type (like, static or dynamic)	**No**	**Yes**
vii)Detect with an alert message and get the sound alarm	**No**	**Yes**
viii)Prevent the spoofing and detect with an alert at the same time	**No**	**Yes**

We got eight major differences between other tools and our developing tool for the ARP spoofing detection process. So, we can clearly come to a conclusion that our developing tool has more features and is pretty much reliable than other tools.

CONCLUSION

We have trained our developed algorithm for spoofing and spoofing detection between two different PCs. The reason behind using two different PC's under different networking connections is to show the difference between them and find out the best for create a predictive model for ARP spoofing and its detection. This kind of research plays a very important role in the world of computer networking and its privacy as well. So, more and more involvement in this type of experiment is very necessary. Concluding, we realized that this project is made us more interested to work with more PC's IP addresses, real time ARP, and MAC addresses under different networking connections. We are eagerly waiting to contribute to more complex and bigger projects in the future.

ACKNOWLEDGMENT

The project has been done with the optimum contribution of four group members, Mahmud Jamil, Farhat Jebin, Md. Fahimul Islam and Sakib Khan students of East West University, Dhaka.