Communications and Network, 2016, 8, 118-130
Published Online August 2016 in SciRes. http://www.scirp.org/journal/cn
http://dx.doi.org/10.4236/cn.2016.83012

Address Resolution Protocol (ARP): Spoofing Attack and Proposed Defense

Ghazi Al Sukkar[1], Ramzi Saifan[2], Sufian Khwaldeh[3], Mahmoud Maqableh[4], Iyad Jafar[2]

[1]Electrical Engineering Department, The University of Jordan, Amman, Jordan
[2]Computer Engineering Department, The University of Jordan, Amman, Jordan
[3]Business Information Technology Department, The University of Jordan, Amman, Jordan
[4]Management Information Systems Department, The University of Jordan, Amman, Jordan
Email: ghazi.alsukkar@ju.edu.jo, r.saifan@ju.edu.jo, Sf.khwaldeh@ju.edu.jo, mmaqableh@gmail.com, iyad.jafar@ju.edu.jo

Received 7 May 2016; accepted 11 July 2016; published 14 July 2016

Abstract

Networks have become an integral part of today's world. The ease of deployment, low-cost and high data rates have contributed significantly to their popularity. There are many protocols that are tailored to ease the process of establishing these networks. Nevertheless, security-wise precautions were not taken in some of them. In this paper, we expose some of the vulnerability that exists in a commonly and widely used network protocol, the Address Resolution Protocol (ARP) protocol. Effectively, we will implement a user friendly and an easy-to-use tool that exploits the weaknesses of this protocol to deceive a victim's machine and a router through creating a sort of Man-in-the-Middle (MITM) attack. In MITM, all of the data going out or to the victim machine will pass first through the attacker's machine. This enables the attacker to inspect victim's data packets, extract valuable data (like passwords) that belong to the victim and manipulate these data packets. We suggest and implement a defense mechanism and tool that counters this attack, warns the user, and exposes some information about the attacker to isolate him. GNU/Linux is chosen as an operating system to implement both the attack and the defense tools. The results show the success of the defense mechanism in detecting the ARP related attacks in a very simple and efficient way.

Keywords

Address Resolution Protocol, ARP Spoofing, Security Attack and Defense, Man in the Middle Attack

1. Introduction

Communication is becoming more and more important in today's life, whether it is work-related, or to get connected with family and friends. Computer networks play a major part in this process. In recent days, it is hard to find a computer or a smart phone that is not connected to a network in some sort, whether it is directly connected to the Internet service provider (ISP) or through a local area network (LAN) alongside other devices.

To facilitate this process, network engineers came up with a set of rules called protocols for message exchange between computers. These protocols maintain the connection between the connected devices within a network and allow them to work efficiently and trouble free. Engineers, who tailored some of these protocols, did not take into account the bad intentions of some users who may exploit the vulnerabilities of these protocols. Some network attacks may give access to attackers (hackers) or allow them to alter the behavior of the network by deceiving these vulnerable protocols with false information. By doing so, they might capture valuable information of an institute or an organization or sensitive private data which may harm individuals and infringe their privacy.

In this paper, some of these vulnerabilities are illustrated, mainly in the Address Resolution Protocol (ARP) protocol. An application that exploits these vulnerabilities to perform a "Man-in-the-Middle" attack is implemented. In addition, an application that works as a defense against these types of attacks is designed and implemented.

Man-in-the-Middle (MITM) attack is an active eavesdropping attack, where in a communication session between two devices A and B, the attacker deceives A by pretending to be B. This means whenever A wants to send a message to B, it actually sends it to the attacker who will read/modify the message then forward it to B in order to keep the continuity of the communication. The attacker will be able to read and modify all the contents of the communication before forwarding it to B.

ARP is a protocol used by the data link layer to map IP address to MAC address [1]. Before encapsulating the network layer packet in a data link layer frame, the host sending the packet needs to know the recipient's MAC address. Given the IP address of a host, to find its MAC address, the source node broadcasts an ARP request packet which asks about the MAC address of the owner of the IP address. This request is received by all nodes inside the LAN. The node that owns this IP address replies with its MAC address. It is worth mentioning that the ARP reply is unicast while the ARP request is broadcast.

When the host receives the ARP reply, usually the IP/MAC addresses mapping are saved as entries in a table called the "ARP table". This table is used as a cache where the node will send an ARP request only if the ARP table does not contain the IP/MAC mapping.

ARP's weakness lies in the fact that it is a stateless protocol, *i.e.*, it accepts ARP replies without having to send an ARP request. ARP spoofing attack exploits this vulnerability by sending ARP reply messages that contain the IP address of a network resource, such as the default gateway or a DNS server, to a victim machine. The attacker replaces the MAC address of the corresponding network resource with his machine's MAC address. The victim's machine that receives the spoofed ARP replies cannot distinguish them from legitimate ones. Moreover, the ARP tables usually use the result of the last ARP reply only.

The attacker then takes the role of man in the middle; any traffic directed to the legitimate resource is sent through the attacking system. The attacker reads the packet looking for sensitive data; it may modify that data, and then passes it to the designated destination. As this attack occurs on the lower levels of the TCP/IP protocol stack, the end-user is unaware to the attack occurrence.

Furthermore, ARP spoofing is also capable of performing denial of service (DoS) attacks if the attacker drops the packets instead of passing them. This causes the victim's machine to be denied from service from network resources. There exist some tools which simplify generating ARP spoofing attack, like Ettercap which is developed by Ornaghi and Valleri [2] and another tool developed by Wagner [3].

In this paper, both types of ARP spoofing attacks; MITM and DoS attacks are implemented. And most importantly, a defense against these attacks is suggested and designed by developing a simple tool that can be deployed on clients' machines to warn them in case of an attack. Moreover, it exposes the IP and MAC addresses of the attacker.

The remaining of this paper is organized as follows. Section 2 presents some related works. In Section 3 an ARP spoofing scenario with packet inspection is explained. A description of the man in the middle attack in and its defenses techniques are introduced in Section 4. The suggested defense mechanism is illustrated in Section 5. Section 6 shows the implementation details of the attack and the defense tools. The results and the GUI interfaces for the attack and the defenses are shown in Section 7. Finally, the conclusion is drawn in Section 8.

2. Related Works

ARP spoofing attack is a well-known attack [4]. Some of the existing defenses and solutions exist in literature. Few of them are presented in this section. The authors in [5] assume that each host has a public/private key pair certified by a local trusted party on the LAN, which acts as a Certification Authority. Messages are digitally signed by the sender, thus preventing the injection of spoofed information. This can be a good solution for upper network layers. However, Data link layer authentication is not straight forward, where the solution should modify all existing network card drivers, which is not a simple issue.

In [6], the authors assume that there is a database deployed in the LAN which can be used to resolve the MAC address. However, this approach does not succeed with dynamic networks in which many nodes join and leave the network daily, where each node should be registered in the database before it can work. Also, the attacker can still generate the attack on the database.

The authors in [7] proposed a unicast ARP request instead of the broadcast ARP request, with the help of a DHCP. In their approach, they assume that the DHCP will resolve the IP/MAC translation without the need for broadcast. However, the DHCP is at the application layer. Also, the DHCP may work only for dynamic IP addresses; it does not succeed for static IP addressing.

A Ticket-based Address Resolution Protocol (TARP) was proposed in [8]. TARP implements security by distributing centrally issued secure IP/MAC ticket with the help of the DHCP. These tickets are given to clients as they join the network and are subsequently distributed through existing ARP messages. These tickets include asymmetric digital signature, which has a considerable overhead in generating the public/private key pairs, and consumes more time. It is also not suitable for the dynamic networks where many new computers join the network frequently.

The authors in [9] mentioned several resolutions and solutions to solve the ARP spoof problem and grouped them into cryptographic approaches, kernel-based patch, host-based approaches, port security on switch, manually configuration of static ARP entries, ARP spoof detection & protection software, server-based approaches and ASA (anti-ARP spoofing agent) software.

The authors of [10] introduced a system that consists of a MAC-Agent and a Client-Agent. The MAC-Agent makes a reliable ARP table and sends the data to the Client-Agent, which prevents the host from using ARP. Instead, the Client-Agent receives the reliable ARP table information from the MAC-Agent and updates the ARP table information as static type. However, this solution requires cryptographic authentication techniques which are not available at the level of the data link layer. Additionally, this approach requires modifying the existing ARP protocol which is unpractical. Similarly, the authors of [11] suggested a protocol that requires modifying the ARP protocol to solve uncertified ARP table renewal from an ARP reply.

The authors of [12] suggested search methods. However, search methods do not solve the ARP spoofing attack if the attacker is intelligent and customized its protocol stack, and this approach requires extra administration efforts. A technique that depends on installing a new DHCP server for use as a MAC-IP database center was proposed by [13]. However, DHCP is widely used and cannot be modified which makes this solution unrealistic.

The authors of [14] introduced a technique that includes collecting and analyzing the ARP packets, and then injecting ICMP echo request packets to probe for malicious host according to its response packets. However, this approach can be detected by the attacker and can be easily fooled.

In this paper, a distributed approach with no need for any central entity like the DHCP server is adopted. Our adopted approach is based on monitoring the hacking behavior. If an entity tries to inject some spoofed ARP replies, it will be detected and revealed. It is deployed on the clients and does not need any kind of cryptographic authentication. Once the attacker is detected, the proposed tool can recover from the attack and the information of the attacker is exposed.

3. ARP Spoofing Scenario with Packet Inspection

ARP is a protocol that works inside the LAN when a node has an IP address of a local destination and wants to know the MAC address of that destination. If the destination is outside the LAN, then there will be an intermediate node called the default gateway (router) which forwards the data to outside the LAN. In this case, the ARP is used to map the IP address of the gateway to its related MAC address. The gateway IP address is either statically stored inside the nodes or dynamically collected from the DHCP server. In the MITM attack, the attacker modifies the ARP tables to either pretend being the gateway or the target device.

For example, in a communication between two computers A and C with IPs 10.0.1.2 and 10.0.2.2 respectively, as shown in **Figure 1**. Computer A sends a packet to computer C which is outside the LAN with the source IP of A and destination IP of C. Then, computer A tries to find out whether the destination IP is in the same network or outside it. A will discover that 10.0.2.2 is outside the network. Therefore, A should forward the packet to the gateway which is Router 1 with IP address 10.0.1.1.

The IP packet is sent to the lower layer of the OSI model (data link layer). The source MAC address is set to the MAC address of computer A, and the destination MAC address is the one that matches Router 1 interface which has IP address 10.0.1.1. If A's ARP table does not contain the MAC address of the 10.0.1.1 interface, then A broadcasts an ARP request packet to figure out the MAC address of the gateway.

The attacker receives the broadcasted ARP request, and replies by a spoofed ARP reply packet to the victim's machine (*i.e.*, machine A). Even if the gateway sends a legitimate ARP reply, the attacker may send multiple spoofed ARP reply packets to overwrite the legitimate ARP reply packet sent by the gateway. In ARP protocol, the newest reply overwrites the oldest ones. Therefore, the ARP cache on the victim machine will save the spoofed IP/MAC mapping.

Now, all traffic going out of the victim's machine to the gateway will be sent to the attacker's machine. In order to manipulate the gateway's ARP table to force all the packets destined to the victim from outside the LAN, to be received by the attacker, the same above procedure will be repeated by sending spoofed ARP reply packets to the gateway. Additionally, to maintain the victim's connection to the gateway running, such that the attack will be un-detectable, the attacker must keep forwarding the victim's traffic to the gateway and vice versa.

Figure 2 shows a network topology with one attacker, one victim machine where an MITM attack will be applied against it, and one router which plays the role of a gateway. **Table 1** shows the contents of the spoofed ARP reply packets that should be sent by the attacker. Both packets should hold a proper op-code denoting them as ARP reply packets. **Table 2** shows the IP/MAC cache entries of the victim's machine and the router; before and after the attack.

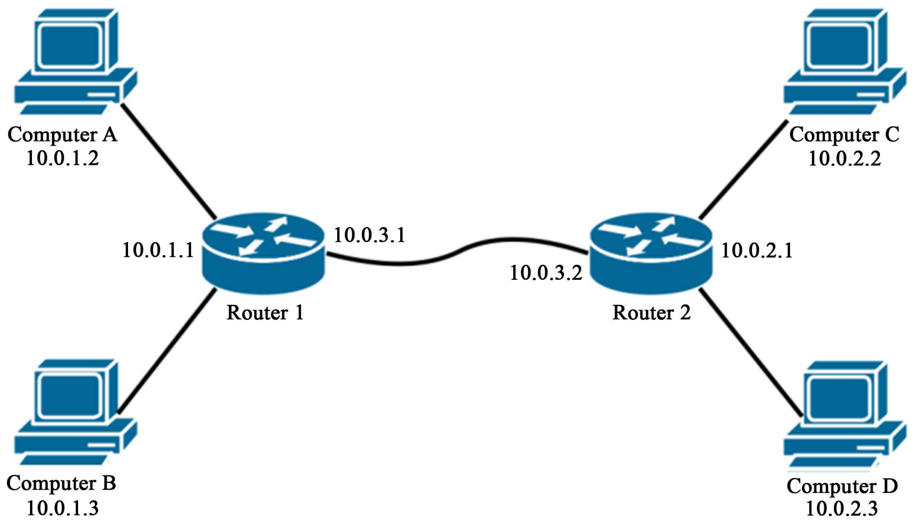


Figure 1. Example of network topology.

Table 1. Required ARP Messages fields to be sent.

ARP Message fields	To victim	To router
Sender MAC address	CD:EF:12:34:56:78	CD:EF:12:34:56:78
Sender IP address	192.168.1.1	192.168.1.10
Target MAC address	AB:CD:EF:12:34:56	12:34:56:78:9A:BC
Target IP address	192.168.1.10	192.168.1.1

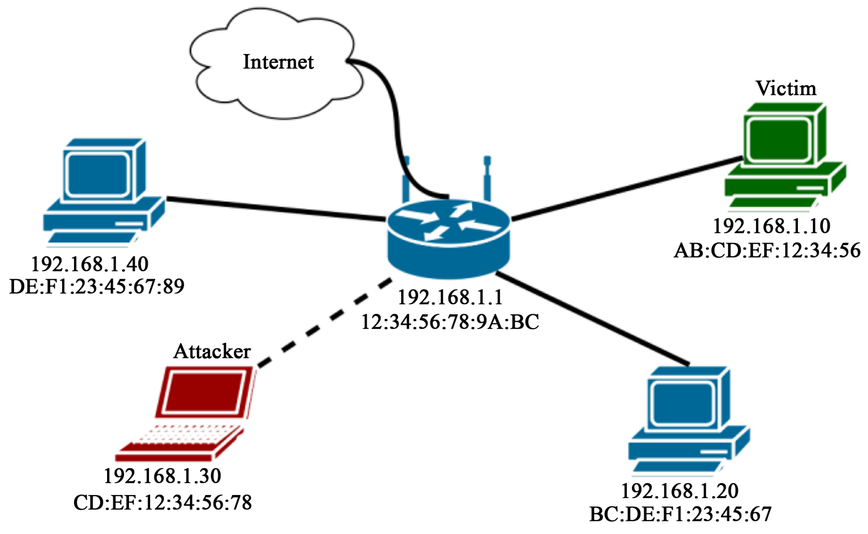


Figure 2. A victim and an attacker in a LAN.

Table 2. ARP cache entries before and after ARP spoofing attack.

	Victim's machine		Router	
	IP	MAC	IP	MAC
Before	192.168.1.1	12:34:56:78:9A:BC	192.168.1.10	AB:CD:EF:12:34:56
After	192.168.1.1	CD:EF:12:34:56:78	192.168.1.10	CD:EF:12:34:56:78

Subsequently, any traffic from the victim to outside the LAN will pass through the attacker machine. For example, suppose that the victim opens a web page on an HTTP server with some external IP address (A.B.C.D) that is outside the LAN. The operating system running on the victim's machine will choose the router as the next hop. To encapsulate the network layer packet inside a data link frame, the victim should know the MAC address of the router. It will check the ARP cash table first. The ARP table has the poisoned mapping of router IP to the attacker's MAC address, which is the result of the ARP spoofed reply packets. Hence, that packet will be sent to the attacker's machine instead of the real HTTP server. If the attacker drops the packets, then this will results in a DoS attack where the victim is precluded from browsing the Internet. On the other hand, if the attacker forwards the traffic to its real destination, the attack is classified as a MITM attack.

Some settings should be performed at the attacker machine in order to allow it to forward the packet even if the IP is not destined to it. This is because the network layer at the attacker machine will drop the packet if it is not destined to it. To change that, a feature called the forwarding feature in the attacking machine's operating system must be enabled.

When forwarding is enabled on a machine, it reads the destination IP address in the packet, and forwards it to the machine with MAC address bounded to that IP address based on the machine's ARP table (which contains the legitimate information). Packet forwarding is implemented in most operating systems; including Microsoft Windows, Linux and Mac OS. However, packet forwarding is disabled by default. It can be enabled easily as will see later. If packet forwarding is enabled, the victim connections will keep working without any suspensions. The attacker can still inspect the packets and extract any desired information.

***Packet Inspection procedure*:** After succeeding in attracting the traffic to the attacker's machine, the attacker needs to filter it and extract the information from the packets. Since there are millions of packets going in or out the network, it is impossible to manually check every single packet for the needed information. Therefore the attacker will filter the packets to extract the ones of interest.

The most convenient way of filtering the packets is by determining which packets are valuable for the attacker. First thing that comes to mind are the ones that contain sensitive information like passwords. Moreover, the websites that the victim browses might be of interest for the attacker. Each of these types of packets has different fingerprints.

The packet inspection process is done by disassembling each packet, searching for the protocols that are of interest and the fingerprint of these packets to extract the information that are needed. **Figure 3** shows a packet dump as an example.

Parsing the first 14 bytes, yields a relation to layer 2, the information that can be extracted from this layer are the MAC addresses of the source and the destination. MAC addresses are 48 bit or 6 bytes long. The first 6 bytes represent the destination address (c8:3a:35:1f:1b:c8) while the second 6 bytes represent the source address (a4:17:31:0c:9a:65). Bytes offset 14 through 33 belongs to layer 3 which contains IP addresses and the network protocol.

Table 3 shows some decoded parts of the packet and the information extracted from these parts. It is clear from the table that the packet is a standard DNS query, issued by a machine with IP address of 192.168.1.107 asking a DNS server at IP address 208.67.222.222 about the domain www.ju.edu.jo.

4. Defense against MITM Attack

The man in the middle attack has very dangerous consequences on the victim. Sensitive data could be extracted without any notice from the victim. A way to face this kind of attack should be implemented. In the world of networks security, there are two types of countermeasures; prevention systems and detection systems. Prevention systems try to prevent the occurrence of the attack, while the detection systems trigger some sort of alarm when an attack is detected. For best security measures, both prevention and detection systems should be implemented in the network.

```
offset 0  1  2  3  4  5  6  7  8  9  10 11 12 13 14 15
0      c8 3a 35 1f 1b c8 a4 17 31 0c 9a 65 08 00 45 00
16     00 46 88 01 00 00 40 11 81 70 c0 a8 01 6b d0 43
32     de de df 81 00 35 00 32 8d 31 d6 ee 01 20 00 01
48     00 00 00 00 00 01 03 77 77 77 02 6a 75 03 65 64
64     75 02 6a 6f 00 00 01 00 01 00 00 29 10 00 00 00
80     00 00 00 00
```

Figure 3. An example of packet dump in hexadecimal.

Table 3. Inspection of the packet shown in Figure 3in Hexadecimal.

Byte offset	Hex data	Decoded data	Description
23	11	17	UDP protocol
26:29	C0 A8 01 6B	192.168.1.107	Source IP address
30:33	D0 43 DE	208.67.222.222	Destination IP address
34:35	DF 81	57217	Source port
36:37	00 35	53	Destination port
44:45	01 20	0120	DNS parameters
54:68	03 77…6F	www.ju.edu.jo	DNS query domain

4.1. Prevention Systems

Some of the network configuration techniques are by de facto considered countermeasures for the MITM attack. Nevertheless, each method has its own disadvantages. The first method is Static ARP table: This type of table cannot be modified by an ARP reply packet. If an attacker starts an MITM attack, spoofed ARP replies cannot affect a static ARP table implemented at the victim's machine. Although this solution is efficient, the overhead in deploying these tables by network administrators in each machine on the network, and keeping them up to date is very hard and tedious in large networks.

The second method is ARP filtering; in which each ARP packet is monitored and logged. If that ARP reply packet was not requested, it gets dropped. This type of solutions is less efficient than static tables; because an expert attacker will wait for an ARP request from the victim's machine, and replies with a spoofed ARP reply packet. The attacker can read the ARP request because it is a broadcast packet.

The third method is Authentication: It is helpful in isolating malicious users entirely and denies them from accessing the network. Authentication systems are efficient in small and closed organizational environments; where users' selection process can be applied based on some authentication policies. But in a more open and public environments, number of users will increase and change rapidly, causing difficulties in keeping the circle of authenticated users free of unwanted malicious users. Moreover, ARP is at data link layer, and it is not straight forward to add authentication at data link layer.

4.2. Detection Systems

Detection systems are meant to be installed on end users devices to monitor the ARP table of the user's machine and detect any change in the gateway entry. Then, alert the user that he is under attack. Contrariwise to prevention systems, detection systems should be implemented and designed to be easily installed and used by end users without the need of skilled system and network administrators. In general, these systems can be applied to any type of network.

5. The Suggested Defense System

Our suggested defense system is an MITM attack detection program that can be installed on end users' machines. It alerts the victim once he/she falls as a victim of MITM attack. In addition, the program tries to heal the ARP table by reconfiguring it with the correct values. The same program can be installed on the network administrators' machines; revealing the IP and MAC addresses of the attacker to help in identifying and isolating the attacker's machine from the network.

The assumption is that the network is initially safe with no attackers. Once the user connects to a network, the program will try to acquire legitimate network parameters. Then, it monitors the ARP table for any non-legitimate changes that might be caused by an MITM attack and warns the user about it. The attacker MAC address can be found in the non-legitimate spoofed ARP replies since in the MITM attack the attacker uses his machine's MAC address bound to the gateway IP address.

The IP address of the attacker can be obtained by resolving his machine's MAC address using the Reverse Address Resolution Protocol (RARP). After an attack is detected, the program starts the healing process. The First part of the healing process is reversing the MITM attack. This is achieved by deleting non-legitimate ARP entries from the ARP table. Then, an ICMP ping packet is sent to the gateway router to force the operating system to re-acquire the gateway's MAC address again. Theoretically, a race condition will arise between the MITM attack initiated by the attacker, and the reversing the process performed by the victim. In fact, the victim will be the winner of this race, since the ARP table, which both the victim and the attacker is trying to control, is saved on the victim's machine that has the defense program running on it.

Second part of the healing process, is to send ARP reply packets with legitimate parameters to the gateway router. A balanced race condition will arise here; since both the attacker and the victim have the same control over the gateway router. The defense program continues sending the legitimate ARP replies while the attacker is sending the spoofed ones.

The suggested defense mechanism is summarized by the following points. This situation assumes a victim is being attacked and the defense program is running on the machine:

1. Initially, the router is presumed to be empoisoned.

2. Upon starting the defense program on the victim machine, it will send ARP request packet to the router, which in turn will reply with correct IP/MAC mapping.

3. The program saves that mapping. Then it keeps monitoring the router MAC address by periodically sending ARP request packets and compares the replies with the previously saved one.

4. The attacker starts initiating an MITM attack by sending poisoned ARP replies to the victim and the router.

5. The router updates its ARP table according to the received poisoned replies.

6. The victim's machine will also updates its ARP table according to the poisoned replies. Simultaneously, the defense program compares these replies with the previously saved one and detects a change in the router MAC address.

7. The program warns the victim about the attack, deletes the poisoned entry from the ARP table, sends healing packet to the router carrying the correct IP/MAC mapping and exposes the IP and MAC addresses of the attacker.

8. Both the defense and the attack processes will keep trying to alter the router ARP table yielding a race condition.

9. Finally, the victim can use the exposed information to report the attack incident.

The outcomes of an MITM attack scenario against a victim running the defense program is expected to be as following:

• The victim's machine remains safe, where the attacker is not able to read the traffic flowing from the victim's machine to the router.

• The router may face a balanced race condition, which will result in a semi-failure. But, since the victim detected the attack, it can tell the router about the attack and the attacker gets isolated.

6. Tools Implementation

This section explains the implementation details of the MITM attacking and defense tools. The goal is to create two efficient and easy to use tools; one is an attacking tool to perform a complete MITM attack scenarios, and the other is a defense tool that can be used by end users to protect themselves against MITM attacks.

Linux based operating systems are chosen as the development platform due to the availability of network related commands and compatibility with network programming libraries. Python language is used as the main programming language of implementation. Version 2.7.6 is used in our tests and it is guaranteed to work trouble-free, earlier and more recent versions of Python 2 may also work, however newer implementation of the Python interpreter (version 3.x) does not work due to compatibility issues with Scapylibrary [15]. The Scapy library contains classes and methods to sniff, manipulate and send data packets of a wide range of network protocols.

6.1. Attack Implementation

The ARP class from Scapy library is used to forge ARP reply packets. Considering the network topology shown in **Figure 2**, the following code block illustrates how to create a packet object of the ARP packet that should be sent to the victim's machine.

```
fake_vic = ARP(op=2,
hwsrc="CD:EF:12:34:56:78",
psrc="192.168.1.1",
hwdst="AB:CD:EF:12:34:56",
pdst="192.168.1.10")
```

The first parameter is the operation code of the ARP packet. The value 2 denotes that it is a reply packet. The second parameter is the MAC address of the source machine. The third parameter defines the IP address of the source machine. The IP address of the router is passed to this parameter to deceive the victim's machine. The fourth parameter is the destination IP address which is of the victim's machine. And the fifth parameter is the MAC address of the destination which is the victim's machine MAC address.

To send a packet, a function called send() provided by Scapy library that is capable of sending a packet object passed to it. The following code is a call to send() function to send the previously created ARP packet.

```
send(fake_vic)
```

After applying this command, the ARP table will have poisoned entries. Now, any packet sent from the victim's machine and directed at the router will be sent to the attacking machine. Another packet object should be created to deceive the router, with same parameter but with different values that correspond to the attacking situation shown in **Figure 2**. The following code block shows the creation of this packet.

```
fake_gw = ARP(op=2,
            hwsrc="CD:EF:12:34:56:78",
            psrc="192.168.1.10",
            hwdst="12:34:56:78:9A:BC",
            pdst="192.168.1.1")
```

Most routers feature a web based interface to manage their settings and configurations, which can be used to show the ARP table entries of the router to verify the success of this step.

A problem that could arise now is that after awhile the router may send an ARP reply telling its correct MAC address. This means the victim's machine will not remain deceived. A solution to this problem is to keep sniffing ARP replies sent by the router and resend the defined forged ARP replies once an ARP reply is detected. Scapy library provides the sniff() function to sniff specific data packets matching a defined filter. The ARP packet intended to be detected is either directed at or issued from the router or the victim's machine. Therefore, any ARP packet containing the IP address of the router or the victim's machine should be followed by forged ARP packets; to keep the attack functioning. The following call to sniff() function blocks the loop till any ARP packet matching the filter is detected.

```
while True:
sniff(filter="arp and host 192.168.1.1 or host
192.168.1.10", count=1)
send(fake_vic)
send(fake_gw)
```

The remaining last step to accomplish a complete MITM attack is to forward the IP packets to its real destination based on the attacker's machine ARP table. Linux systems provide this capability at the kernel level out of the box. The system flag "net.ipv4.ip_forward" holds the state of IP forwarding which is 0 by default in most Linux distributions. Once it is set to 1, the kernel starts the IP forwarding process. Sysctl is a user space interface to change system kernel parameters at run-time. The following command is to enable IP forwarding:

```
$ sudosysctl -w net.ipv4.ip_forward=1
```

Now, the attacker is in position of man in the middle. The victim's traffic should pass through the attacker's machine and then forwarded to its proper destination. It is clear from the details above that the attacker now is able to filter packets, modify the exchanged packets, and send forged packets. This is the source to many other attacks like DNS spoofing attack and phishing attacks. Due to space limitations, the implementation details of packet inspection and those attacks will not be shown here.

6.2. Defense Implementation

Implementation of the suggested defense system begins by monitoring the ARP table which is done by periodically calling the arp command, parsing its output and comparing that output with the previously saved correct IP/MAC entry. Once an attack is detected, a warning message containing the new MAC address assigned to the router will be sent to the GUI process described later. Finally, a healing ARP packet is to be created with Scapy and sent to the router to update its ARP table with the correct IP/MAC mapping of the machine. The following

Python function takes the router's IP address and makes the basic defense process.

Functions get_mac() and get_ip() are used to get the original MAC address of the router and the attacker's machine IP address respectively. Send_warning() function sends a warning message and information about the attacker to the GUI process.

```
defdefence(router_ip):
original_mac = get_mac(router_ip)
while True:
current_mac = get_mac(router_ip)
ifcurrent_mac != router_mac:
attacker_ip = get_ip(current_mac)
send_warning()
heal_router()
delete_mac(router_ip)
ping(router_ip)
else:
sleep(1)
```

Healing process consists of three steps as follows:

1. heal_router() function sends an ARP reply packet containing the correct IP/MAC mapping of the defended machine. The same method shown in Subsection 6.1 can be followed to create and send this healing packet.

2. delete_mac() function deletes the poisoned IP/MAC mapping from the ARP table by calling the command with proper arguments. The following command deletes the ARP entry corresponding to 192.168.1.1 IP address.

```
$ sudoarp --delete 192.168.1.1
```

3. ping() function sends an ICMP packet to the router in order to force the operating system to update its ARP table.

Figure 4 shows the flow of the defense process. As long as no attacks are detected, the process will pause for one second between the main loop iterations using the sleep() function. This procedure is optional but recommended to release the load on the system resources. But if an attack is detected, the instructions inside the "if clause" will get executed without any pauses; to fight back the attacking machine effectively.

To know the IP address of the victim's machine and the router, map library which contains a Port Scanner class can be used. Due to space limitation, it will not be shown here.

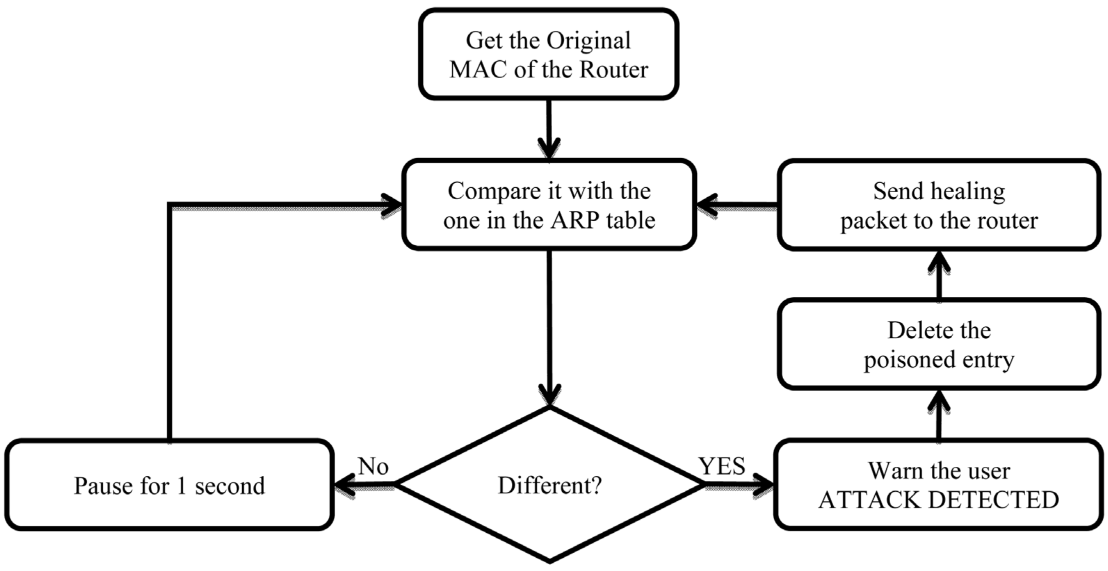

Get the Original MAC of the Router
Compare it with the one in the ARP table
Send healing packet to the router
Delete the poisoned entry
Pause for 1 second
No
Different?
YES
Warn the user ATTACK DETECTED

Figure 4. Defense process flowchart.

7. Results and GUI Tools

In this section, we will show the resulted tools that we have developed. They have easy to use interface to generate the attack and to apply the defense.

7.1. Attack Tool

A GUI is created to work as a front-end for the attacks we have already implemented. The tool is designed with simplicity in mind using minimal amount of widgets. **Figure 5** shows screen shot of the attacking tool.

After selecting the network interface, the application will scan the network for the devices connected to it. A list of all connected devices (except the user own machine) will be in the "Victim IP" list. Note that the scanning process needs a little time to be completed, so the user has to wait until the status bar says "Ready". After that, a "Spoofing mode" should be selected, either a denial of service attack; to block the victim's connection, or a MITM attack; to intercept the victim's packets. After having all the settings done correctly, the "Start" button should be pressed, if the attack is successful, the status bar will show the message "Victim under attack".

Now, all the URLs visited by the victim will appear in the "URLs" section, also if the victim logs in to a service, his credentials will be listed in the credentials section. To browse one of the listed URLs, it should be selected, so it will get copied to the Clipboard, and then it can be pasted in a web browser address bar. **Figure 5** shows a scenario, in which the victim's visited websites are appearing in the "URLs" section, and the DNS spoofing is performed against Facebook website and the credentials are hijacked.

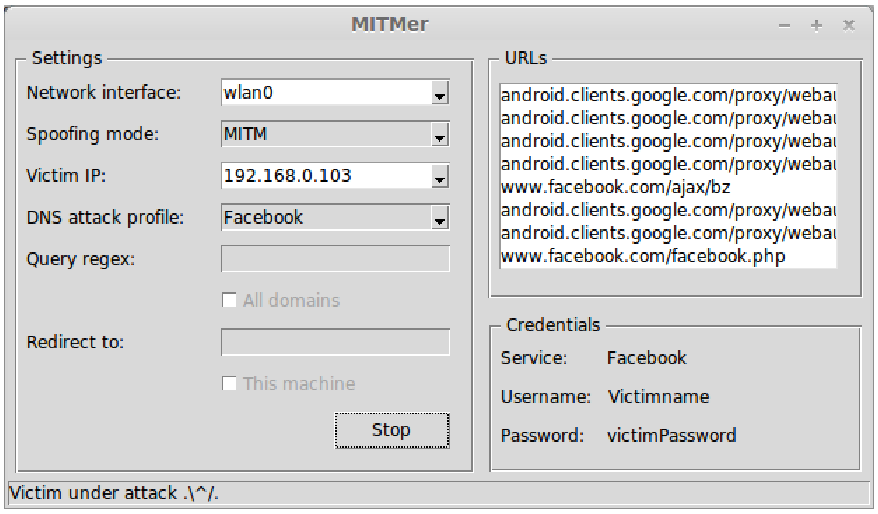
To stop the attack, the "Stop" button must be clicked on. If one of the settings fields is changed while the attack is running, the new settings will not take effect, unless current the attack is stopped and then start again.

7.2. Defense Tool

The main screen window of the defense tool is shown in **Figure 6**. The network interface that is connected to the network should be selected first. Then, the gateway IP address should be entered, or the "Auto detect IP" tick should be selected to automatically detect it. Then the "Start" button should be selected to start the defense process. If an attack incident is detected, the IP and MAC addresses of the attacker will appear in the "Status" section. The status section will show the IP address and MAC address of the attacker. If the "Reset ARP table if changed" check button is ticked; the table will be reset with the correct values. With this feature enabled, the tool will fix the ARP table of the machine running it with valid IP/MAC mapping once an attack is detected. Also, it will send the valid matching to the gateway.

8. Conclusions

The fact that the ARP protocol is a stateless protocol has very dangerous consequences. An attacker could ma-



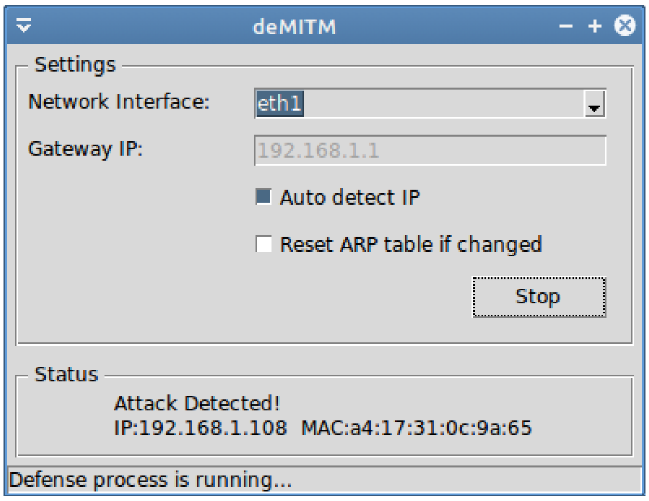
Network interface: wlan0
Spoofing mode: MITM
Victim IP: 192.168.0.103
DNS attack profile: Facebook

```
android.clients.google.com/proxy/webau
android.clients.google.com/proxy/webau
android.clients.google.com/proxy/webau
android.clients.google.com/proxy/webau
www.facebook.com/ajax/bz
android.clients.google.com/proxy/webau
android.clients.google.com/proxy/webau
www.facebook.com/facebook.php
```

Service: Facebook
Username: Victimname
Password: victimPassword

Victim under attack .\^/.

Figure 5. GUI of the attack tool.



Network Interface: eth1

Gateway IP: 192.168.1.1

■ Auto detect IP

☐ Reset ARP table if changed

Attack Detected!
IP:192.168.1.108 MAC:a4:17:31:0c:9a:65

Defense process is running...

Figure 6. GUI of the defense tool.

nipulate the connection of the users, steal their data, and even redirect their traffic to different websites than the ones they requested. Network administrators should become more aware of these attacks and take countermeasures against them. Users should also become more aware of them and use security solutions to prevent getting their data stolen.

In this paper, after explaining the vulnerabilities of the ARP protocol and their different types and categories, we have designed and implemented an application that performs ARP spoofing and MITM attack. After that, we have designed and implemented a simple and distributed defense mechanism and tool that works on end user devices to keep him protected from these types of attacks. This mechanism helps in protecting the user from such attacks as shown by applying and running the final product on the user machine.

References

[1] Plummer, D.C. (1982) An Ethernet Address Resolution Protocol. RFC 826.

[2] Ornaghi, A. and Valleri, M. (2004) A Multipurpose Sniffer for Switched LANs. http://ettercap.sf.net

[3] Wagner, R. (2001) Address Resolution Protocol Spoofing and Man in the Middle Attacks. SANS Institute. https://www.sans.org/reading-room/whitepapers/threats/address-resolution-protocol-spoofing-man-in-the-middle-attacks-474

[4] Bellovin, S.M. (2004) A Look Back at ''Security Problems in the TCP/IP Protocol Suite''. *Proceedings of the 20th Annual Computer Security Application Conference* (*ACSAC*), Tucson, 6-10 December 2004, 229-249. http://dx.doi.org/10.1109/CSAC.2004.3

[5] Bruschi, D., Ornaghi, A. and Rosti, E. (2003) S-ARP: A Secure Address Resolution Protocol. *Proceedings of the* 19th *Annual Computer Security Applications Conference*, Las Vegas, 8-12 December 2003, 66-74. http://dx.doi.org/10.1109/csac.2003.1254311

[6] Gouda, M.G. and Huang, C-T. (2003) A Secure Address Resolution Protocol. *Computer Networks*, **41**, 57-71. http://dx.doi.org/10.1016/S1389-1286(02)00326-2

[7] Issac, B. (2009) Secure ARP and Secure DHCP Protocols to Mitigate Security Attacks. *International Journal of Network Security*, **8**, 107-118.

[8] Lootah, W., Enck, W. and McDaniel, P. (2007) TARP: Ticket-Based Address Resolution Protocol. *Computer Networks*, **51**, 4322-4337. http://dx.doi.org/10.1016/j.comnet.2007.05.007

[9] Venkatramulu, S. and Guru Rao, C.V. (2013) Various Solutions for Address Resolution Protocol Spoofing Attacks. *International Journal of Scientific and Research Publications*, **3**, 2250-3153.

[10] Hong, S., Oh, M. and Lee, S. (2013) Design and Implementation of an Efficient Defense Mechanism against ARP Spoofing Attacks Using AES and RSA. *Mathematical and Computer Modelling*, **58**, 254-260. http://dx.doi.org/10.1016/j.mcm.2012.08.008

[11] Gouda, M.G. and Huang, C.T. (2003) A Secure Address Resolution Protocol. *Computer Networks*, **41**, 57-71. http://dx.doi.org/10.1016/S1389-1286(02)00326-2

[12] Ramachandran, V. and Nandi, S. (2005) Detecting ARP Spoofing: An Active Technique. In: Jajodia, S. and Mazumdar, C., Eds., *Information Systems Security*, Springer, Berlin, Heidelberg, 239-250. http://dx.doi.org/10.1007/11593980_18

[13] Pansa, D. and Chomsiri, T. (2008) Architecture and Protocols for Secure Land by Using a Software-Level Certificate and Cancellation of ARP Protocol. *ICCIT*'08 3*rd International Conference on Convergence and Hybrid Information Technology*, **2**, 21-26.

[14] Jinhua, G. and Kejian, X. (2013) ARP Spoofing Detection Algorithm Using ICMP Protocol. *International Conference on Computer Communication and Informatics*, Coimbatore, 4-6 January 2013, 1-6. http://dx.doi.org/10.1109/iccci.2013.6466290

[15] Biondi, P. (2007) Scapy Website. Retrieved on May 2015. http://www.secdev.org/projects/scapy/