

Detection and prevention of ARP cache poisoning

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering
in
Computer Science and Engineering

Submitted By
Inderjeet Kaur
(801132012)

Under the supervision of:

Mr. Sumit Miglani
Asstt. Professor



COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

THAPAR UNIVERSITY


PATIALA – 147004

June 2013

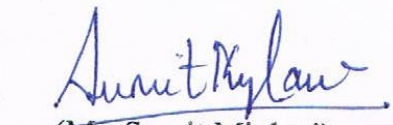
CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, "*Detection and Prevention of ARP cache poisoning*", in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Computer Science and Engineering* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of Mr. *Sumit Miglani* and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.


(Inderjeet Kaur)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.


(Mr. Sumit Miglani)
Asstt. Professor, CSED

Countersigned by


(Dr. Maninder Singh)

Head

Computer Science and Engineering Department

Thapar University

Patiala


(Dr. S. K. Mohapatra)

Dean (Academic Affairs)

Thapar University

Patiala

Acknowledgements

It is a great pleasure to have an opportunity to thank valuable beings for their continuous support and inspiration throughout the thesis work.

I would like to extend my gratitude towards Dr. Maninder Singh, HOD of Computer Science and Engineering Department, Thapar University for all the guidance and great knowledge he shared during our course. The abundance of knowledge he has always satisfied our queries at every point.

Thanks to Mr. Sumit Miglani, My guide for his contribution for timely reviews and suggestions in completing the thesis. Every time he provided the needed support and guidance.

At last but not the least, a heartiest thanks to all my family and friends for being there every time I needed them.

Abstract

Address Resolution Protocol (ARP) is a protocol having simple architecture and have been in use since the advent of Open System Interconnection (OSI) network architecture. Its been working at network layer for the important dynamic conversion of network address i.e. Internet Protocol (IP) address to physical address or Media Access Control (MAC) address.

Earlier it was sufficiently providing its services but in today's complex and more sophisticated unreliable network, security being one major issue, standard ARP protocol is vulnerable to many different kinds of attacks. These attacks lead to devastating loss of important information. With certain loopholes it has become easy to be attacked and with not so reliable security mechanism, confidentiality of data is being compromised.

Therefore, a strong need is felt to harden the security system. Since, LAN is used in maximum organizations to get the different computer connected. So, an attempt has been made to enhance the working of ARP protocol to work in a more secure way. Any kind of attempts to poison the ARP cache (it maintains the corresponding IP and MAC address associations in the LAN network) for redirecting the data to unreliable host, are prevented beforehand.

New modified techniques are proposed which could efficiently guard our ARP from attacker and protect critical data from being sniffed both internally and externally. Efficiency of these methods has been shown mathematically without any major impact on the performance of network. Main idea behind how these methods actually work and proceed to achieve its task has been explained with the help of flow chart and pseudo codes. With the help of different tools ARP cache is being monitored regularly and if any malicious activity is encountered, it is intimated to the administrator immediately. So, in this thesis underlying problem has been analyzed and a solution to detect and prevent ARP poisoning efficiently has been provided.

If any how malicious user by passes the prevention mechanism, and cache is modified according to his wish. It should be detected without further delays so that necessary actions could be taken. The implementation on how to prevent and detect attacks on ARP has been done with the help of virtualization tools. It is tested with the platforms of windows and linux in the backtrack. Linux is taken as the basis to act as the attacking machine.

But, the architecture of ARP is such that all the efforts made till date have not been proven the ultimate solution. The conflicting goals that we desire to get pull each other in different directions. There are many factors that decide the feasibility and reliability of proposed solutions. Hence, none solution is enough to provide complete security. But still we can minimize the attack attempts by prevention and detection of cache poisoning. The results have also shown that the defensive methodology have been proved efficient enough to provide required security.

TABLE OF CONTENTS

S. no.	Particulars	Page no.
	List of figures.....	8
CHAPTER 1	Introduction.....	9
	1.1 Background.....	10
	1.2 Aims and Objectives.....	11
	1.3 Thesis layout.....	12
CHAPTER 2	Literature Review.....	14
	2.1 Network security in general.....	15
	2.2 Major security threats.....	16
	2.2.1 Intrusion or hacking.....	16
	2.2.2 Viruses and Worms.....	17
	2.2.3 Trojan Horse.....	18
	2.2.4 Spoofing.....	18
	2.2.4.1 IP spoofing.....	18
	2.2.4.2 DNS spoofing.....	19
	2.2.4.3 ARP spoofing.....	19
	2.2.5 Sniffing.....	19
	2.2.6 Denial of service.....	20
CHAPTER 3	Address Resolution Protocol.....	21
	3.1 Introduction.....	21
	3.2 Working.....	22
	3.3 ARP poisoning.....	24
	3.4 Existing solutions	26
	3.5 Research Question	29
CHAPTER 4	Attacks.....	30
	4.1 Different type of attacks.....	30
	4.1.1 Sniffing	30
	4.1.2 Man-in-the- Middle attack	31

4.1.3 Denial of service	33
4.1.4 Cloning	33
4.1.5 Smart IP-spoofing	33
4.1.6 Connection hijacking	34
4.1.7 MIMT attacks on encrypted connections.....	34
4.1.8 IP Conflict Attack.....	34
4.1.8 Bombing Packet Attack.....	35
CHAPTER 5 Implementation.....	36
5.1 ARP exploits tools	36
5.2 Working environment	38
5.3 Performing attacks.....	39
5.3.1 Sending spoofed packets manually.....	39
5.3.2 Using Ettercap tool.....	41
5.3.3 Using Cain & Abel.....	45
CHAPTER 6 ARP prevention and detection techniques.....	49
6.1 Prevention techniques.....	51
6.2 Detection technique.....	58
CHAPTER 7 Results and Performance.....	62
7.1 For Prevention mechanism.....	62
7.2 For Detection mechanism.....	63
CHAPTER 8 Conclusion and future scope.....	66
References	
Publication	

LIST OF FIGURES

Figure 3.2.1: The client using ARP to acquire MAC address of other machine using its IP address.....	22
Figure 3.2.2: ARP packet transmitted as payload in Ethernet frame.....	23
Figure 3.2.3: ARP cache, IP and associated Mac addresses.....	23
Figure 3.3.1: ARP Poisoning, reply packet sent to victim and gateway.....	25
Figure 4.1.1: Sniffing of ARP data using wireshark.....	30
Figure 5.3.1: Capturing ARP packet through wireshark.....	39
Figure 5.3.2: Hex editor for creating spoofed ARP reply.....	40
Figure 5.3.3: Poisoned cache of victim machine.....	41
Figure 5.3.2.1 Scanning of hosts in the subnet.....	41
Figure 5.3.2.2 Five hosts discovered in the subnet.....	42
Figure 5.3.2.3 IP and MAC addresses of hosts.....	42
Figure 5.3.2.4 Target hosts are added.....	43
Figure 5.3.2.5 Choosing ARP Poisoning for MITM attack.....	44
Figure 5.3.2.6 Selecting sniffing of remote connections.....	44
Figure 5.3.2.7 Connection details viewed by attacker.....	45
Figure 5.3.3.1 Scanning of the MAC addresses in the network.....	45
Figure 5.3.3.2 Selecting IP range in the network.....	46
Figure 5.3.3.3 MAC addresses existing in the network.....	46
Figure 5.3.3.4 Selecting IP addresses that are to be poisoned.....	47
Figure 5.3.3.5 Poisoning the selected host.....	47
Figure 5.3.3.6 Capturing FTP user name and password.....	48
Figure 6.1: Protection mechanism.....	50
Figure 6.1.1 Flow chart of procedure.....	57
Figure 6.2.1: Trap ICMP ping packet.....	59
Figure 7.1.1: Efficiency in terms of transactions.....	64
Figure 7.2.1: Number of packet injected to number of hosts.....	64
Figure 7.2.2: Measurement of RTT to IP packet routing enabled and disabled.....	65

CHAPTER 1

INTRODUCTION

With the explosive growth rate of the IT (Information Technology) industry, digitized communication has taken over all the fields be it business, medical, education, banking, share market etc. From small to large organizations own their own private internal network to work in collaboration and share the information. They also join the outer network to take the competition of the global market head on. But when you are connected with outer world, you need to ensure that the information sharing is done in a secure way. Here comes the major role network security mechanism.

Creating an environment of secure networking is not an easy task. Network connectivity has diminished the distance barrier between two communicating terminals. Network size has also increased in size and complexity. Many people with wrong intention are also part of the same network pretending to be the genuine ones. There are many loopholes and vulnerabilities in the different protocols and applications that we use. These people make use of such discrepancies in network for fulfilling their illegal activities. Many network users are unaware of such attacks and easily fell as prey for such hackers.

It has become a tedious task for all the system administrators and network security professionals to be able enough to predict, prevent and detect different kind of attacks that the hackers use, to secure the information. Every day hackers pop up with new techniques to bypass the security mechanism. But the job of administrator is to act with more efficient counter measures.

In the thesis its been tried to explore all the aspects of one of the major security attack made on ARP cache. The attacker poison the ARP cache of the host system thereby leading the redirection of data travelling to its own system. Then it captures all the data and retrieves the valuable information such as usernames and passwords for further damage to the important accounts to get more confidential information or destroying the accounts. This thesis tries to implement certain techniques by which act of poisoning cache could be prevented and detected before major loss of data. It has really become very important to secure the data travelling on internet.

ARP spoofing is one of several vulnerabilities which exist in modern networking protocols, which allow a knowledgeable individual free reign over a network. These attacks are relatively simple to employ, as there are a wide variety of automated tools available, while any type of defense against them is minimum.

Because of the rapid advancements made in IT sector, wireless networks are also spreading its usage. ARP spoofing is not limited to LAN network now, it is being practiced over wireless networks also. Airtight Networks, the leading provider of wireless intrusion prevention systems and services, published its research on Wi-Fi Vulnerability in WPA2 security protocol on July 22, 2010. And on the company's website, it published a free software that serves the purpose of demonstrating how an attacker can utilize the software to access the network after getting the GTK (Group Temporal Key); with ARP Spoofing method, he/she can then mislead other users the attacking PC is the AP (Access Point) of the wireless network [21].

A research on the earlier proposed ideas and architectures has been made to formulate the basis this thesis. An analysis is also done to check where these were not enough to provide complete security. And results have been derived on testing basis that how far we can go in order to secure our ARP cache.

1.1 BACKGROUND

The most commonly employed network frame architecture is based on OSI model. It having different layers with their defined services that they provide to their upper layers. Each layers service depends on the varying protocols working at layers. But working of these protocols to ensure complete security to the applications running has really become a challenging task for the developers and security professionals. There have been different efforts made in the field of network security to provide and invent effective and reliable defensive mechanisms to control attacks being made of these protocols due to their lack of ability to work and prevent themselves in complex and insecure network. One such protocol is ARP working on network layer. Due to its architecture it cannot prevent itself from being attacked over LAN. Local Area Network (LAN) are generally the privately owned networks in the organizations, it may be small on large depending on the size of organization. It is prone to many

different kinds of attacks described in next chapters. But the loss of information it does in more devastating.

There could many different reasons for which some insiders also attack system in their own organization LAN network. Study and results have also proven that 70 to 80 percent of the attacks are done on the internal networks only [5]. Administrator and security analysts are going to hard phase to prevent these internal attacks, since protecting oneself from outside is easy as such attacks are predictable and obvious to happen. So, they keep themselves shielded and updated to these attacks all the time. But internal attacks are more difficult to protect against. Since poisoning of ARP cache is done due to flaws of the ARP only and allows the attackers to capture and misuse the information, and therefore attention of administrators to protect and enhance ARP is drawn. Due to many unsuccessful and ineffective tools to detect and prevent ARP cache poisoning I have been motivated to work on this area.

But the major question is how to prevent and detect attacks attempted to cache in the local network.

“How effective is the detection and prevention of such attacks?”

This thesis tries to give explanation to this question in the most convincing way. The derived results and analysis of working of each proposed method and tool will serve the basis for answering the above question. The results will also give clear look into the facts that how much these tools are able to provide security in varying environment with their own feasibility issues. These results have derived in the environment having system like linux and windows.

1.2 Aims and Objectives

The aim of this thesis is to provide with an effective approach that will be able to prevent and detect ARP cache poisoning. This is achieved by deploying certain third party tools to intimidate any wrong or unauthorized attempt to change the contexts of the cache with invalid associations. This prevent the unwanted redirection of the data thorough the attacker's system and valuable data would not be sniffed.

The following are the main objectives of the thesis:

1. Demonstrating a real time cache poisoning on ARP cache.
For doing this we will be using following tools in a virtually created LAN network with hosts working on different operating systems.
 - a) VMware workstation (for creating virtual environment hosting both windows and linux operating system)
 - b) Back Track 4 (Open source Hijacking tools)
 - c) Wireshark or Ethereal (Packet capturing tool)
 - d) Ettercap, Cain & Abel (Spoofing tools)
2. To demonstrate how to prevent and detect cache poisoning if attackers is in the same network. Tools used to implement prevention approach are:
 - a) ARP watch
 - b) Kernel patches
3. At user end side to implement and test the effectiveness of employed technique to achieve the task of being secure from ARP cache poisoning.

1.2 Thesis Layout

The main outline of the thesis how it is being described throughout is as following:

- **Study and analysis of exiting protocol architecture – its working scenario.**
- **What are the main loopholes in protocol.**
- **How the protocol is attacked – ARP cache poisoning.**
- **Understanding the underlying issues in prevention and detection of cache poisoning.**
- **Defining the approach to evaluate the problem.**
- **Research on the required tools and techniques to be used for implementation.**
- **Creating the virtual environment and testing in real environment.**
- **Implementation of the proposed approach.**

- **Carrying and analyzing the obtained results.**
- **Comparing the results with the existing ones.**

The above described plan is just the outline of the thesis; there were many other researches that were conducted in order to carry out the satisfactory implementation of the proposed detection and prevention plans. The inputs not only include the deep study of the all the earlier research carried related to this but also the analysis of practical implication of the mechanisms.

CHAPTER 2

LITERATURE REVIEW

As to lay down the foundation of this thesis many research papers and articles were thoroughly reviewed and studied. The key observation and their main characteristics and research implementations have been used to create the base framework of this project. One of the biggest threats to the computer network is pretending of any rogue system to be the trusted one. Once any such system has successfully impersonated another host, he can do a lot of harmful things [17]. For example he can intercept and log traffic destined for the real host, or makes the authorized client to wait in queue and may start sending confidential information to the rogue system.

Spoofing a host specially has severe consequences in IP networks, because it opens many other avenues of attacks. One of the known techniques to spoof the host is by spoofing the ARP cache, but it works only in the local segments. ARP spoofing in order to poison the ARP cache allows any computer in the LAN to have one of the most harmful and dangerous attack postures in the security. This kind of attack called Man-in-the-Middle attack would be able to capture, modify and filter all the data meant to be travelling between two trusted hosts of the network. And there would be nothing to prevent it from filling all the hosts' cache with its own IP and MAC association, thus enabling it to become effectively the master hub for all the information moving in the network.

Many efforts have been made and different methods have also been applied to prevent such attacks at ARP, but none has been able to give satisfactory results. Different tools and architectures have been proposed but each have their own feasibility issues. Today's techniques can't give you complete protection against ARP attacks, but we can guard our self with IDS and specialized ARP manipulation sensors, explained in upcoming chapters, to detect most manipulation attempts. Ignoring the issue is not a convincing option unless we can genuinely trust every user with access to our LAN. ARP spoofing is one of several vulnerabilities which exist in modern networking protocols, which allow a knowledgeable individual free reign over a network. As we have seen these attacks are relatively easy to implement, as there is a large variety of automated tools available, while any type of defense against them would not be enough.

2.1 Network security in general

Any communication happening over network requires having security. So, it needs to happen in the two links only in complete isolation from other systems though at the same are also part of the same network. This is what that formed the basis for securing the communication. For this securing the computer is done to secure the information and valuable assets in the computer. This could also be called the hardening of the system.

So, the whole concept of network security is based to have information security. The major goals of this information security are:

- **Confidentiality:** It means the protecting the information that is present in a system from unauthorized people. Such as information regarding customer credit card, information of patients in hospitals or information related to employees of an organization. If information of such level of confidentiality is not secured, the company or the organization involved will probably lose its reputation and goodwill in the market.
- **Integrity:** It indicates that information available in an organization should be complete and whole. It should not be changed by any unauthorized person. Any kind of intentional or unintentional alterations of the information will lead to damaging and making of information unreliable. A good example of such a case would be account information in a Bank. If anything is done with banking information, it is devastating and the Bank will eventually lose its customers and business. In fact, there is also a chance of facing a lawsuit also.
- **Authenticity:** It is the identification and assurance of the origin the data. It ensures that the access to the data should be to trusted people only. Valid usernames and passwords should be given and these should be kept confidential.
- **Availability:** It says the information requested or required by the authorized users should be available, always. For example, suppose a company met with a natural calamity and it has lost its computers and all the important data. In such cases, the affected company should be able to install new computers and recover its data from backups. Suppose, the company had not had any proper backup plans, they would definitely be not able to recover their data and resume its operations.

2.2 Major security threats

The main categories of the security threats for information security are as following [18]:

2.2.1 Intrusion or Hacking

It means getting access to a computer system without any prior knowledge to its owner. The people who do such kinds of unlawful activities are known as hackers. If anyhow they get into the system they can do whatever they want. They can make the system result in unpredictable behavior by changing the data of the crucial system files. They can easily change the log files without letting know the owner that any intruder has done something malicious to their system. They can also leave some port open providing a backdoor for them to enter the system again whenever they want. The intruder can also get access to all the personal information regarding the bank accounts, credit card information and other crucial data. Hackers generally target eCommerce websites, machines operating individually, banking sites that provide online services. The type choice of systems that hackers attack depends on their personal types. Some people involve in hacking activities just for curiosity or to showcase their intelligence. But, also is not an easy task. For this the hacker has to get to the target computer and gather every kind of possible information regarding the systems strengths, weaknesses, operating systems that are used, application running, opened ports and then analyzing how any weakness could be exploited to get the crucial information. Once a way has been found, they will enter the through that, and try to exploit the systems. Some of the techniques or loop holes that the Hackers generally use for hacking purposes are as follows:

1. Poor security mechanism
2. Hidden fields in the html forms
3. Client-side validation scripts
4. Direct SQL attack
5. Session Hijacking
6. Buffer Overflow Forms

7. Port Scanning

In order to prevent systems from such attacks, many of the eCommerce websites and individual users have started employing the use firewall system that work with higher layers and inspect the data travelling more deeply. And whenever there is any attack, the firewall systems immediately intimidate the administrator and sometimes it also helps in tracking the attackers. But hackers always remain find a way to bypass the firewall system by some new way and hence it is always better to have get done a vulnerability test before releasing any system for operation.

2.2.2 Viruses and Worms

Viruses and Worms are small but dangerous computer programs that are capable of making computer systems not to work properly. There is a minor difference between Virus and Worm. Though, both can replicate themselves while traveling on the network but virus needs a carrier file. It cannot travel on its own in the network where as worms don't need any carrier.

Viruses and Worms really create problem in smooth running of the systems. The main aim for which these viruses and worms are created is to lead the system to malfunction and many times worms come as handy to hackers to sniff in and steal private data of the system owner. According to Trendmicro, there are as many as 60,000 viruses that have been identified and nearly 400 new viruses are getting created every month. Viruses spread through Internet, which serves as gateway for such malicious programs. These can spread very quickly and easily affect all the systems in an organization within a few minutes and can cause loss of thousands of rupees.

The good way to avoid viruses is get the anti-virus softwares installed on all systems. Since, new viruses may even bypass these softwares. Therefore, it is very important to keep the virus-signature-database up to date. In addition to anti-virus software, users should be also remain careful while downloading files from the internet, because generally virus preferable is attached to such files for which user is obviously prompted to download. If these files are from unreliable source, it is best to delete them right away.

2.2.3 Trojan Horse :

Trojan horse programs were initially created to be used for system administration purposes. System administrators used these programs to control their work-stations remotely. These programs generally have two components, one runs as a server and other as client. The server is installed at the work station site and the client on the administrator machines. But it also gets used by hackers for illegal purposes, like sniffing of the data at the user side, controlling their machines etc. This can prove quite dangerous and lead to many threatening issues like integrity attack, stealing of private information, storing key strokes making visible to hackers, making screen shots available. It is also easy to get Trojan horse installed on user machine by sending it attaching it to an e-mail.

2.2.4 Spoofing

Literal meaning of spoofing is deceiving people by playing a trick. Here also, spoofing in network language is making the computer users think that the information flow to their system is from legitimate user, in actual it is not. Three main methods of spoofing are as follows:

1. IP Spoofing
2. DNS Spoofing
3. ARP(Address Resolution Protocol) Spoofing

2.2.4.1 IP Spoofing:

This method exposes the commonly used filtering devices based on IP addresses, such as firewalls. In this case, the attacker's goal is to spoof the client's IP address to bypass the security controls in the server side. Here attacker's box can be located in any intermediate network between the client and the server. To impersonate the client at the IP level the attacker needs to use Source NAT, a method that allows setting the source IP address in the IP packets to the client IP.

2.2.4.2 DNS Spoofing:

It directs the users to an incorrect location. Or we can DNS spoofing is done to direct the users to a different website and making the user to fill false web forms to collect personal information. DNS Spoofing is very dangerous since, DNS is responsible for managing the domain names and create corresponding IP addresses. Suppose, there is a domain with the name www.abc.com and DNS redirect this site to the IP of any hacker's website, then he is able to collect data of user easily.

2.2.4.3 ARP Spoofing:

ARP spoofing involves construction of forged ARP replies. As ARP is a stateless protocol, most operating systems will update their cache if a reply is received, regardless of whether they have sent out an actual request or not. By sending forged ARP replies, a target computer could be convinced to send frames destined for computer A to instead go to computer B. When done properly, computer A will have no idea that any such redirection in sending data has taken place. *The process of updating a target computer's ARP cache with a forged entry is referred to as "poisoning"*. The result of ARP cache poisoning is that the IP traffic intended for one host is diverted to a different host. This is further discussed in detail in upcoming sections since this is topic of our research.

2.2.5 Sniffing

Sniffing is the term used to describe the reading of all packets on a network segment. Network cards can enter a state called "promiscuous mode" where they are allowed to examine frames that are destined for MAC addresses other than their own. This way it helps in locating any network problems. But if hacker gets to sniff the packets, he can easily get used ID's and passwords. There are many tools available to sniff the data. The main targets for sniffing attacks are the system with UNIX operating system. So, to avoid sniffing it's better to send data in encrypted form before sending it on the wires.

2.2.6 Denial of Service

This DoS attack is made to bring down the targeted network and making it to provide its services to the legitimate users. For this attack, a simple ping command can also work. Or updating ARP caches with non-existent MAC addresses will cause frames to be dropped. These could be sent out in a sweeping fashion to all clients on the network in order to cause a Denial of Service attack. Another possibility is redirecting all the traffic to the attacker's box to silently discard it without forwarding the packets.

CHAPTER 3

ADDRESS RESOLUTION PROTOCOL

3.1 Introduction

When any two computers need to communicate in the network, they need to identify each other uniquely. There are two types of addresses that are used to uniquely identify a host:

- MAC Address

This address is also known as hardware address, LAN address, physical address, or Network Interface Card (NIC) address. Each computer's network interface card is assigned a globally unique six-byte address by the factory that manufactured the card. When a host sends out an IP packet, it uses this source address and it receives all packets that match its own hardware address. This Ethernet address, typically a 48-bit address, is a link layer address and depends on the network interface card used.

- IP Address

Internet Protocol operates at the network layer and is independent of the hardware address. The IP address of a host is a 32-bit address assigned to a host and is either static or dynamically assigned by Dynamic Host Configuration Protocol (DHCP).

The individual networks that make internet are connected by routers. These routers need to know the IP addresses to send the packets in the right direction. Since routing rely on the IP addresses, but LAN use only MAC addresses. So, it would be quite not as convenient for each program to know both IP and MAC address. This is where the role of ARP comes into existence, for providing this so crucial <IP, MAC> association. So, administrators do not need to set these pairs but at the same time automation of these is complex issue.

The Address Resolution Protocol (ARP) [12] is used by computers to map network addresses (IP) to physical addresses (MAC). When any host that wants to know the MAC (Media Access Control) address of another host in the LAN network, it broadcasts an ARP request in the network asking for the MAC address of a host with particular IP. The host with the given IP replies back in a unicast ARP replies along with its corresponding

MAC address. The host that issued the request store that <IP, MAC> association in a local ARP cache so that it could use that pairing in the near future if required.

3.2 Working

When an Ethernet frame is broadcasted from one machine on a LAN to another, the 48-bit MAC address is used to determine the interface for which the frame is destined. *Address resolution refers to the process of dynamically finding a MAC address of a computer on a network.* The protocol provides a dynamic mapping between the two different types of addresses that are IP address and MAC address which are used by data link layer. The process is dynamic since it happens automatically and is normally not a concern of either the application user or the system administrator. In a shared Ethernet where hosts use the TCP/IP suite for communication, IP packets need to be encapsulated in Ethernet frames before they can be transmitted on to the wire.

There is a one-to-one mapping between the set of IP addresses and the set of Ethernet addresses. Before the packet can be encapsulated in an Ethernet frame, the host sending the packet needs the recipient's MAC address. Therefore, ARP is used to find the destination MAC address using the IP address.

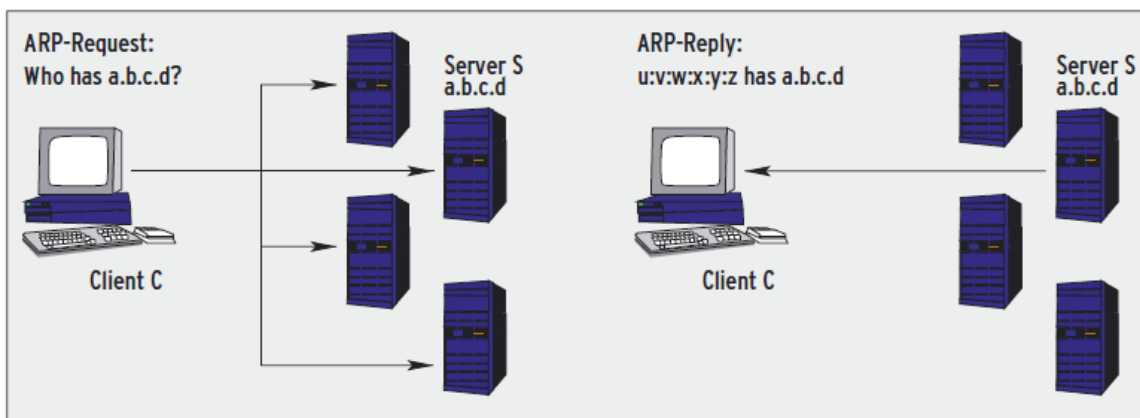


Figure 3.2.1: The client using ARP to acquire MAC address of other machine using its Internet Protocol address [1]

In the above figure, if client C needs to send a packet to server S, it needs to know the MAC address of S if both machines are on the same sub net. Even if S resides in a different network, C still needs the MAC address, in this case, the address of the next

router that will forward the packet. The router takes care of everything else. To ascertain the MAC address, C broadcasts an ARP request to all the machines on the local network, asking “Who has the IP address a.b.c.d?” The computer with the matching number replies and tells the client its MAC address.

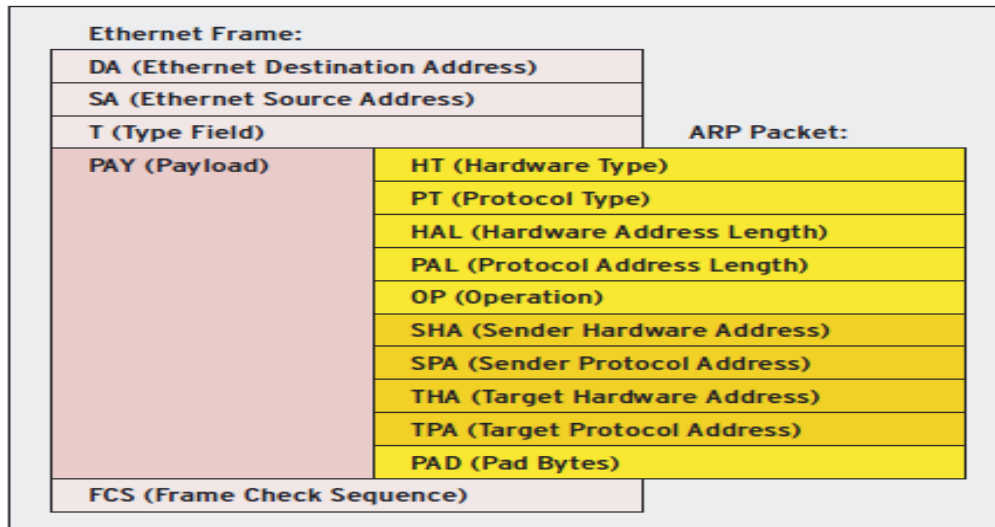


Figure 3.2.2: ARP packet transmitted as payload in Ethernet frame [1]

As shown in Figure 3.2.2, an ARP packet is carried as the payload in an Ethernet frame. As it would be too expensive to broadcast an ARP request and wait for the response before sending data, each IP stack has an ARP table, also known as an ARP cache.

```
C:\WINDOWS\system32\cmd.exe

C:\>ping 192.168.227.131

Pinging 192.168.227.131 with 32 bytes of data:

Reply from 192.168.227.131: bytes=32 time<1ms TTL=64
Reply from 192.168.227.131: bytes=32 time<1ms TTL=64
Reply from 192.168.227.131: bytes=32 time<1ms TTL=64
Reply from 192.168.227.131: bytes=32 time<1ms TTL=64

Ping statistics for 192.168.227.131:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>

C:\>arp -a

Interface: 192.168.227.1 --- 0x2
    Internet Address      Physical Address        Type
    192.168.227.131      00-0c-29-fe-21-cc      dynamic

C:\>_
```

Figure 3.2.3: ARP cache, IP and associated Mac addresses

The cache contains a table with IP addresses and corresponding MAC addresses. The table can hold static entries that are generated by the user and dynamic entries that are learned from the ARP protocol. Dynamic entries are often valid for a short period only, typically just a few minutes. And it is often refreshed automatically by system.

3.3 ARP Poisoning

ARP has been proved to work satisfactorily under regular circumstances, but it was not designed in order to cope with malicious hosts. With ARP cache poisoning or ARP spoofing attacks, an intruder can easily impersonate another host and can get access to sensitive information. Furthermore, these attacks can be easily performed by using widely available and easy to handle tools specially designed for attacking purposes only.

There are many security issues arise with the use of ARP in a LAN. It may create vulnerabilities and threat to the confidentiality of data, several schemes to mitigate, detect and prevent ARP attacks have been proposed. But each has its limitations. In this thesis we tried to work in a best possible way to identify a most reliable solution to the problem, analyzing each of the schemes to look into their advantages and weaknesses.

In order to minimize the number of ARP requests that are being broadcast, operating systems maintain a cache of ARP replies from different hosts. When a host receives any ARP reply, it will normally update its ARP cache with the new IP/MAC association entry. Note that the <IP, MAC> mapping received in the ARP reply should be used to update the ARP cache, only if that sender's IP address is already in the table [12]. ARP does not maintain the states of its own and hence does not check whether the upcoming ARP reply was actually requested or not, before updating the corresponding pairing in the ARP cache of the system. So, the attacker sends the bogus replies to the communicating systems, thereby making the changes favourable to attacker, in the pairing of IP and MAC addresses. By doing this the information starts going through the attacker's machine, without coming into notice of actual hosts. As a performance improvement, some operating systems (e.g., Linux and Windows) cache replies received from hosts whose IP addresses were not previously in the ARP table [5].

ARP spoofing is mainly construction of forged ARP replies. When a forged ARP reply is sent, a target computer could be easily pursued to send frames meant for Host A

to instead go to Host B. If done properly, Host A will have no idea that any such redirecting of data has taken place. *The process of updating a target computer's ARP cache with a forged entry is referred to as "poisoning".* The result of ARP cache poisoning is that the IP traffic intended for one host is diverted to a different host.

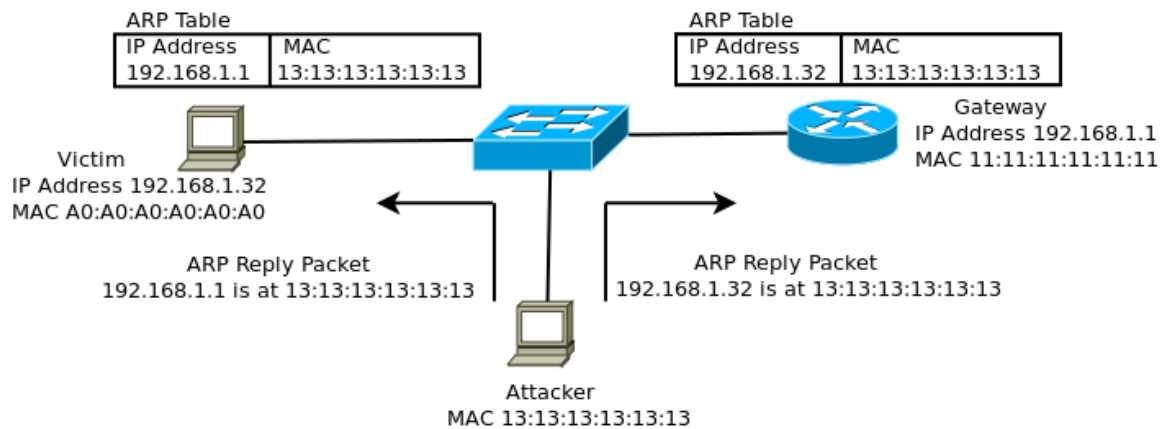


Figure 3.3.1: ARP Poisoning, reply packet sent to victim and gateway [23]

Following are the possible ways in which an ARP cache can be poisoned [3].

- **Unsolicited Response:**

A response which is not associated with any ARP request will be honored by such an ARP implementation attack. A rogue host only needs to send a response or reply ARP packet on the LAN with a spurious mapping in order to poison the ARP cache of victim host. Such a response can be broadcasted to poison cache of every host on the local network.

- **Request:**

ARP caches the replies based on the requests it made. That means, if any host X sends out a broadcast of ARP request to know MAC address of host Y, host Z may store in its cache the mapping information regarding host X based on the request host X sends. An attacker only needs to pretend as if it is the one to send the legitimate request in order to poison the ARP cache of a victim host.

- **Response to a request:**

Instead of sending an unsolicited response, or an unauthenticated request, a malicious user may choose to wait until a victim broadcast a request and sends a

response to that request. If legitimate host responds to the request, there is a race condition which the malicious host can win.

- Request and response:

A malicious user may also send out both a untrue request and a response corresponding to that request. And this could be used to poison the victim's ARP cache in a case the victim remembers a request, either its own, or from another host and only caches a response to a request.

3.4 Existing Solutions

The problem of ARP spoofing was discovered in 1989 [23]. Many efforts have been made and different methods have also been applied to prevent such attacks at ARP, but none has been able to give satisfactory results. Each has its own feasibility issues considering different factors like backward compatibility, cost, efficiency, ease of implementation, size of network, reliability, manageability etc. of these respectively. Different available techniques could be broadly classified on basis of approaching levels. They are:

1. System level

- a) Static IP: Here IP and Mac association is entered in cache by the administrator itself. Therefore forged replies are not able to manipulate the cache. But it also increases the workload of administrator. But for a small network, one is able to protect its gateway effectively.
- b) Operating system: In linux, its kernel 2.4 does not respond to the unrequested replies, but updates on requests. And it could be made to respond using tools like ettercap. Solaris[1] also updates its entries after some predefined time bounds. This also does not prevent attacker, since it can manage to reply before the legitimate user working fast enough to meet time limits.
- c) Firewalls: These also apply the act of detecting only the modified log entries. If found anything suspicious, intimidate the administrator.

- d) Etable: It is the utility available in linux for programming the switches. It could be used to avoid ARP poisoning also but much of the task is left on the shoulders of administrator, that one could easily made mistakes while programming. Also these rules for ARP prevention are not widely available.
2. Hardware level
- a) Sniffer: Efforts were made by M.M. Dessouky [4] to built a hardware easy to be installed acting as a sniffer for detecting the attacks. But major question is to prevent the attacks because once the attack has been made it gives enough time to the attacker to do malicious task in network.
 - b) Port security: What it mainly does, binds a specific MAC address to the port. Performance level is also maintained, requires certain rules to be configured. It provides security from only certain type ARP attacks not all.
 - c) Dynamic ARP Inspection[9]: These Cisco's high end switches that update IP/MAC binding after analyzing DHCP IP releases. And effectively drops the invalid replies. But these are quite expensive.
3. Middleware level
- a) ArpWatch: Is easily available tool that act as a monitoring agent for the arp related activities. It reads the previous and updated data and alarms the administrator if anything does not match. But it gives a lot of false notifications while in an environment where DHCP is used.
 - b) ARP-Guard: It works within architecture employing sensors. A better approach than arpsniff but not good enough.
 - c) Snort: It's a kind of intrusion detection system. It constantly observes the network for malicious activity regarding ARP and timely send the information to the administrator. But it is mainly deployed at network borders, and is not worth of deploying within the internal network. This whole approach goes in vain when many IDS system does not consider working with DHCP and not much backward compatible with general ARP.

- d) Anticap: Is a kernel patch that rejects the invalid combination of IP and MAC addresses learning from earlier entries. Does go well with DHCP. Being kernel patch requires kernel space thereby slowing down the performance.
4. Cryptographic level
- a) S-ARP: Secure ARP[5] involves cryptography to get the ARP replies be digitally signed before they are considered valid for updating the ARP cache. It is backward compatible also but requires a signing authority server that will keep track of all the public and private keys of all the participating hosts in the network. It adds to the complexity of the whole network and also slows down the performance as it requires time for validating the digital signatures. In case the authority server fails, it leads to the failure of the whole network.
 - b) IP-sec: Secure ARP provides authentication at link layer only where as IP-sec can provide more protection with almost the similar overheads. But it puts a lot of load on CPU thereby effecting the performance whereas S-ARP leads to less load on CPU.
5. Architectures proposed:
- a) A simple approach is to divide the large network into small networks so they are easily maintained by the administrator. In smaller number of hosts any malicious host is also easily identifiable.
 - b) Middleware approach given by Tripunitara[3] is not practically implemented. Its is a kind of asynchronous prevention and detection scheme. It requires a lot of changes to be made on all the hosts in the network. Though it is backward compatible but no widely acceptable implementations are available.
 - c) Gouda, [11] proposed an architecture using a secure server. The communication with server is done using two protocols. But it is not practical, as it requires new protocols to be installed at every host and failure of secure server will collapse the whole network.

3.5 Research Question

Communication over network has undoubtedly changed world in a lot better way. But at the same carrying out that transfer of data in a secure way is very important. Therefore preventing and detecting attacks that are launched over network is newsworthy for the security engineers. Because of certain vulnerabilities in many of the protocols and applications, data is being compromised on the grounds of integrity, availability and confidentiality.

One such vulnerable protocol is ARP. There are certain factors due to which no robust solution has been provided to prevent is poisoning. Lacking features to check the validity of its ARP packets, ARP will update a device's ARP Cache with every solicited or unsolicited "ARP Reply" it receives regardless who sent it. This is the vulnerability of ARP and any one with the right programs can exploit it [19]. There are many conflicting issues that lead such a situation where one or the other issue gets to compromise. For ideal solution following is required [2]:

- The solution should not suggest making changes on every host in the network.
- Cryptographic techniques should be avoided in order to not compromise performance as it generally takes processing time.
- Prevention must be preferred over detection since detection becomes overhead for administrators.
- It should be easy and widely available.
- Hardware expenses should be less.
- Backward compatibility with the existing ARP.
- ARP requests and replies should not slow down.
- Every kind of attack should be blocked.
- It should be reliable and manageable.

Design of ARP is such that we cannot look over its vulnerabilities. Being stateless it does not check whether the upcoming reply was actually asked or not. There are more number of exploits available to attack, that too free of cost than solutions to protect it. And all above factors lead us to choose this topic for research why till date ARP still being attacked and is not protected completely.

CHAPTER 4

ATTACKS

4.1 Different types of attacks

ARP poisoning attack that has been discussed in earlier chapter can also be used as a part of some other serious attacks. By using the proper tool which are generally easily available and open source, any novice or script kiddy attacker can perform these kinds of attacks [20]. ARP does not make any robust efforts to protect itself and hence is susceptible to get compromised. Following are the major attacks that could be attempted once the ARP cache gets compromised.

4.1.1 Sniffing:

Sniffing is the term used to describe the reading of all packets on a network segment. Network cards can enter a state called “promiscuous mode” where they are allowed to examine frames that are destined for MAC addresses other than their own.

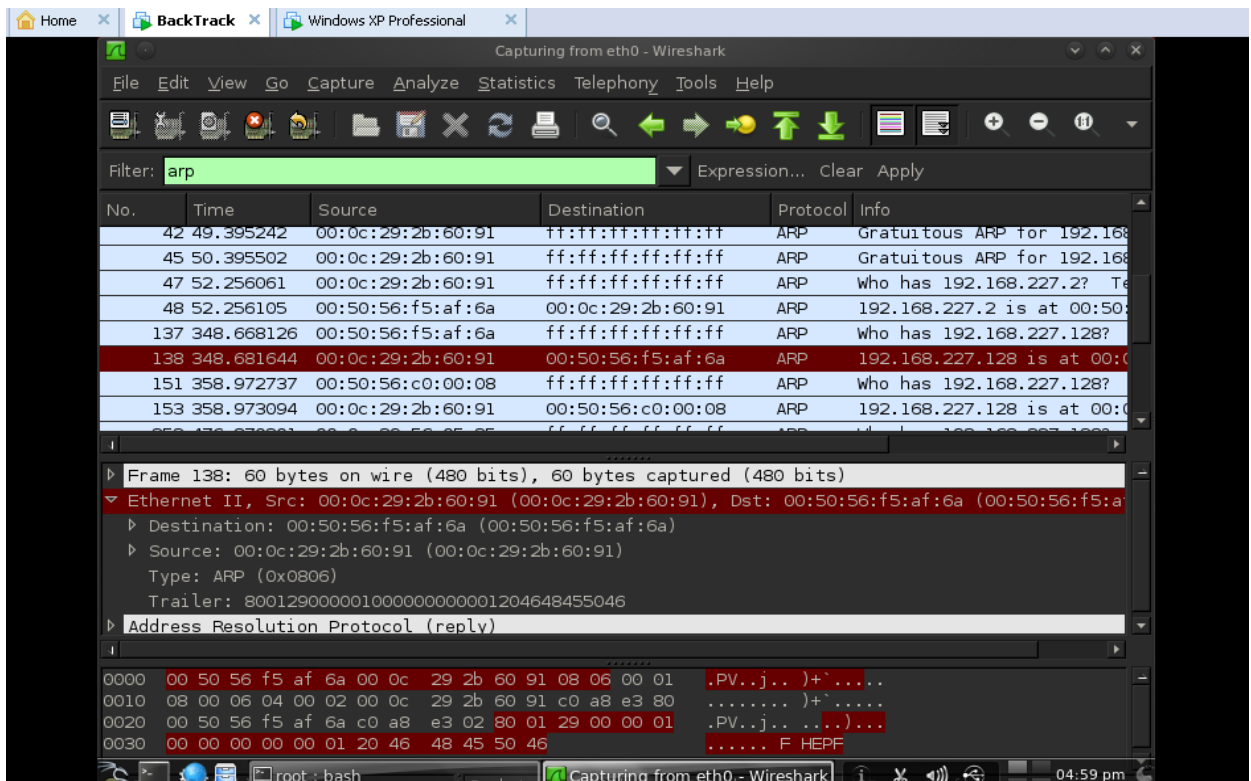


Figure 4.1.1: Sniffing of ARP data using wireshark

On switched networks this is not the scenario, because the switch routes frames based on the table maintained in cache. This prevents sniffing of other people's frames. However, using ARP spoofing, there are two methods to sniff traffic in a switched environment.

The first is in which the attacker sends multiple spoofed ARP replies to the switch. The switch will process these replies and will update the table. If this is done at a rapid rate the switch's table will overflow and the switch will default to broadcasting all traffic to all ports. Then the attacker would be able to sniff the traffic. The second method involves a 'man in the middle' attack described further in detail.

Sniffing can also be of two types, one is external network sniffing and other is internal network sniffing [26]. In external one the attacker is not the part of local network. For sniffing the data in this case it needs to pass the router and modifies its TTL of IP grouping to make sure that its response package reaches the destination. On the other hand in the internal sniffing attacker is itself a host in the network and fulfill its purpose of sniffing using MITM attack.

4.1.2 Man-in-the-Middle (MITM) attack:

It is one of the primary kind of attack used for hacking. It invokes attacks such as denial of service, DNS spoofing and port stealing. It can eventually lead to stealing of online account userid, passwords, local ftp id, ssh or telnet session too [30].

By spoofing any two communicating hosts in the network, an attacker can silently sit in between them and he cannot be detected. And all the data traveling between those two host move first to the one sitting between them, thereby listening to all the traffic. This attack can be performed being the part of the same network or from outside also as attacker can act between the host and the default gateway. With this attack one can access the sensitive information and can modify it thereby compromising its integrity. This attack is explained as following. Suppose there are 2 hosts that are Host A and Host B and an attacker

machine. Now, we have assumed arbitrary IP and MAC addresses for 3 machines as:

HOST	IP ADDRESSES	MAC ADDRESSES
Host A	192.168.10.2	00:00:00:00:00:02
Host B	192.168.10.3	00:00:00:00:00:03
Attacker	192.168.10.4	00:00:00:00:00:04

Firstly the Attacker will poison A's ARP cache with a spoofed ARP Reply. The ARP reply will tell A that the IP address of B now has a MAC address of 00:00:00:00:00:04. Once A has processed the ARP Reply its ARP cache will look like this:

HOST	IP ADDRESSES	MAC ADDRESSES
Host A	192.168.10.2	00:00:00:00:00:02
Host B	192.168.10.3	00:00:00:00:00:04
Attacker	192.168.10.4	00:00:00:00:00:04

Secondly the Attacker will poison B's ARP cache with a spoofed ARP Reply. The ARP reply will tell B that the IP address of A now has a MAC address of 00:00:00:00:00:04. Once B has processed the ARP Reply its ARP cache will look like this:

HOST	IP ADDRESSES	MAC ADDRESSES
Host A	192.168.10.2	00:00:00:00:00:04
Host B	192.168.10.3	00:00:00:00:00:03
Attacker	192.168.10.4	00:00:00:00:00:04

Now whenever A sends B an ethernet frame the switch will route it to the attacker's port, this will also be the case whenever B sends A an ethernet frame. The attacker may now sniff the traffic forwarding it on to its originally desired host.

4.1.3 Denial of Service (DoS) attack:

By poisoning the cache of host, attacker can make every data packet meant for real host to come to attacker's machine thereby blocking the communication to the host. Updating ARP caches with non-existent MAC addresses will also cause frames to be dropped. These could be sent out in a sweeping fashion to all clients on the network in order to cause a Denial of Service attack.

Another possibility is redirecting all the traffic to the attacker's box to silently discard it without forwarding the packets. Linux command that could work here:

```
# echo 0 > /proc/sys/net/ipv4/ip_forward
```

The Linux "iptables" module can also be used for this purpose:

```
# iptables -A FORWARD -s client -d server -j DROP
```

4.1.4 Cloning:

MAC addresses were supposed to be unique identifiers for each network interface produced. They were to be burned into the ROM of each interface and are not intended to be changed. But today, MAC addresses could be easily changed. Linux users can even change their MAC without spoofing software, using a single parameter to "ifconfig", the interface configuration program for the OS.

An attacker could DoS(denial of service) a target computer, then assign themselves the IP and MAC of the target computer, receiving all frames intended for the target.

4.1.5 Smart IP spoofing

This method exposes the commonly used filtering devices based on IP addresses, such as firewalls. In this case, the attacker's goal is to spoof the client's IP address to bypass the security controls in the server side. Here attacker's box can be located in any intermediate network between the client and the server. To impersonate the client at the IP level the attacker needs to use Source NAT, a method that allows setting the source IP address in the IP packets to the client IP.

Again, using the “iptables” Linux module, the attacker could implement Source NAT.

The Linux SNAT module will differentiate attacker’s connections from client’s ones

```
# iptables -t nat -A POSTROUTING -o eth0 -d server -j SNAT --to client
```

4.1.6 Connection hijacking

Connection hijacking allows an attacker to take control of a connection between two computers, using methods similar to the man-in-the-middle attack. This type of attack is used against connection oriented protocols such as TCP/IP. This transfer of control can result in any type of session being transferred. For example, an attacker could take control of a telnet session after a target computer has logged in to a remote computer as administrator.

4.1.7 MIMT attacks on encrypted connections:

The attacker can even attack in a secure connection (e.g., SSL or SSH). The application like web browser will generally give a warning to the user that the certificate that has been provided is not the valid one, but most users do not bother about such warnings. Furthermore, a bug in some versions of Internet Explorer enables attackers to hijack SSL sessions without the browser displaying a warning [1].

4.1.8 IP Conflict Attack

It is the case an attacker send modified ARP packet with the association of IP address of the destination with the spoofed MAC address. When there are two MAC addresses with a single IP in the network, IP conflict arises prompting to delete one the entry. Then the network services will be disconnected.

4.1.9 Bombing Packet Attack

The attacker would continually keep sending poisonous <IP, MAC> pairings to the victim host, thereby causing its cache to fill up quickly. So the victim host will spend more of its resources to maintain its cache, which may lead to buffer overflow. And real mapping would never be entered in the cache. In case such is victim is a switch, information flood will occur.

CHAPTER 5

IMPLEMENTATION

In this section actual attack has been performed in the virtual environment to show how ARP cache poisoning happens in the network without the knowledge of the communicating hosts. Here it has been shown how different tools work to send bogus replies and made the cache to update the <IP, MAC> associations.

5.1 ARP exploit tools

There are many tools which are easily available and free of cost. Many a time administrators use some of the following tools to check vulnerabilities in their network. We cannot say that ARP security problem is due to the existence of these tools but rather the protocols is inherently insecure [1].

- **ARP-SK:** The programmers describe this tool as Swiss Army Knife for ARP. It is available in both Unix and Windows versions. The program is made to change ARP tables on different devices.
- **Arpoc and WCI:** This program is used for performing a man in the middle attack on a LAN on both Unix and Windows. It replies to each ARP request that reaches the machine with a spoofed ARP replay and forwards any packets for non-local delivery to the appropriate router.
- **Arpoison:** It is a command line tool that is used for creating spoof ARP packets. The user can specify the <IP, MAC> pairing for both the source and target.
- **Brian:** It is a very simple tool which has been created using only a single C file that disable switching on a LAN by poisoning the ARP.

This easily allows any attacker to sniff all the network traffic.

- **Cain & Abel:** It is a password recovery tool for Microsoft OS. It can sniff the network traffic and uses a variety of techniques to recover different kinds of encrypted and obfuscated passwords. It uses dictionary and brute force techniques for decoding the scrambled passwords. Version 2.5 of this tool has ARP poisoning and spoofing capabilities making it an efficient hacking and auditing tool. It can also attack SSH and HTTPS connections.
- **Dsniff:** It consist of different individual programs for different purposes. Dsniff, Filesnarf, Mailsnarf, Msgsnarf, Urlnarf and Webspay sniff the network data and collect interesting information such as passwords, email, files etc. Arpspoof, Dnsspoof, and Macof helps the administrators getting access to the data protected by switch. Sshmitm and Webmitm perform MIMT attacks on SSH and HTTPS.
- **Ettercap:** Ettercap is generally used for performing man in the middle attacks. It can sniff data from live connections, content filtering on the fly, password collection automatically, manipulation of data within the connection etc. It supports active and passive dissection of many protocols and includes many features for network and host analysis. Can perform attacks on SSHv1 and SSL connections.
- **Hunt:** It has capability to crash the connections, sniffing of data, and hijacking sessions. Uses ARP spoofing and other techniques to perform the tasks.
- **Juggernaut [1]:** In 1997, Phrack Magazine published Juggernaut, a predecessor to many of today's sniffers with ARP cache poisoning capabilities.

- **Parasite:** It sniffs the LAN and sends spoofed ARP replies to the ARP requests. The tool helps the attacker to get a man in the middle position allowing it to get all the traffic of two communicating devices.

5.2 Working environment

We have performed the task of implementing the attack, preventing and detection on a system with following specifications.

Operating system	Microsoft Windows XP professional
Processor	Intel(R) Core(TM) 2 Duo Processor @ 1.83 Ghz
RAM	2 GB
Hard Disk	150 GB

For creating a virtual environment for visualizing local area network, we have used VMware approach to virtualization. It offers a robust virtualization platform that can scale over large number of interconnected physical computers and storage devices to form an entire virtual infrastructure. We used:

- VMware workstation 7.0.0
- Virtual machines – MS win-XP, Backtrack (powerful hacking and auditing tool)
- Wireshark - LAN sniffer
- Ettercap
- Hex-editor to manipulate data (creating spoofed packets)
- Cain & Abel

5.3 Performing attacks

Here, at first we have discussed and showed how the ARP spoofed packets are created manually and then sent to the victim host to poison its cache and how it updates its IP and MAC pairing without checking whether the actual request had been sent or not. Then we have performed it using tools like Ettercap and Cain & Abel.

5.3.1 Sending spoofed packets manually:

We are working using a virtual environment. We capture an ARP packet with the help of Wireshark so as to use it as a template that we can edit in the Hex editor to specify the needs of our attack.

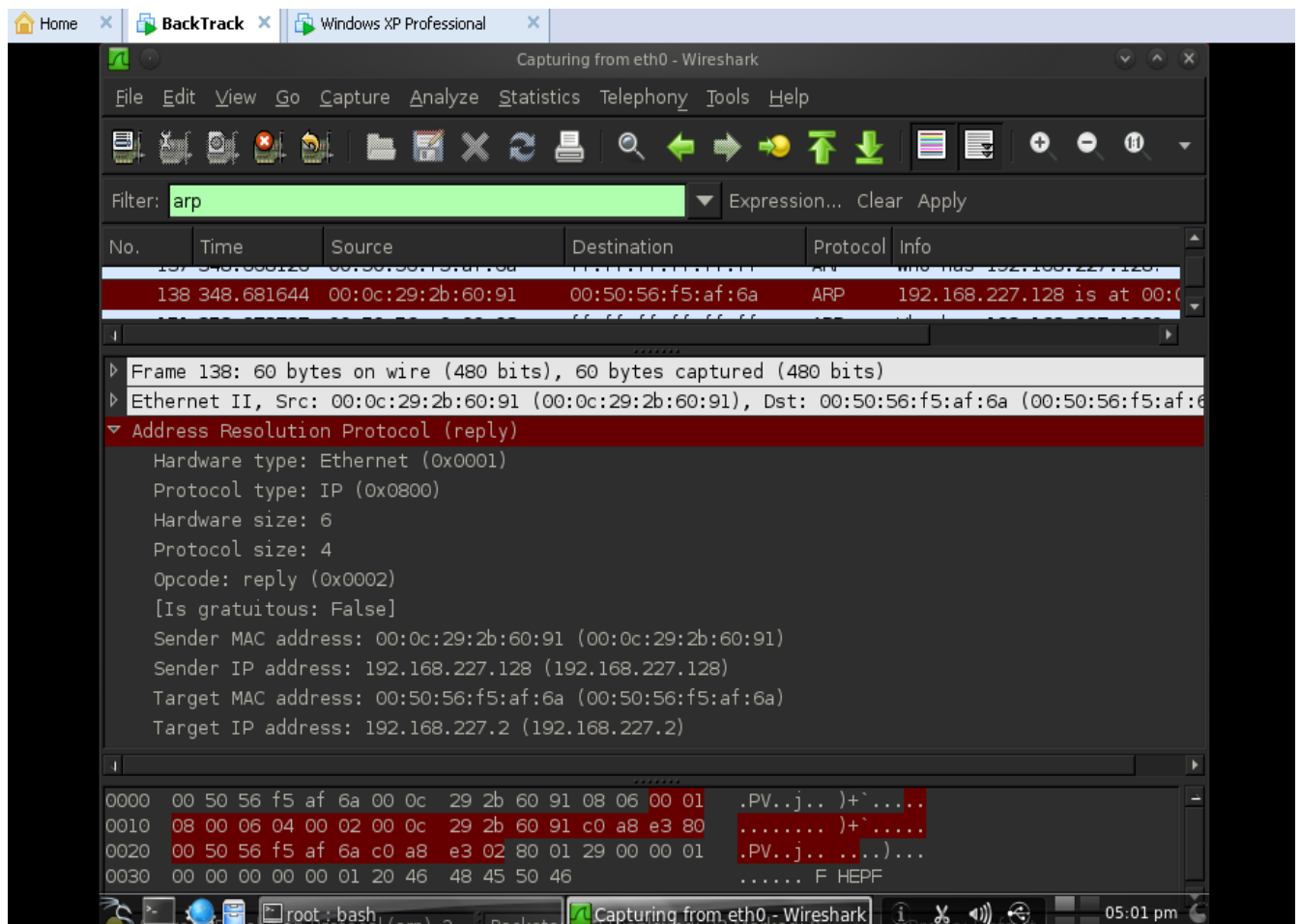


Figure 5.3.1: Capturing ARP packet through Wireshark

Above captured ARP packet is the modified as below:

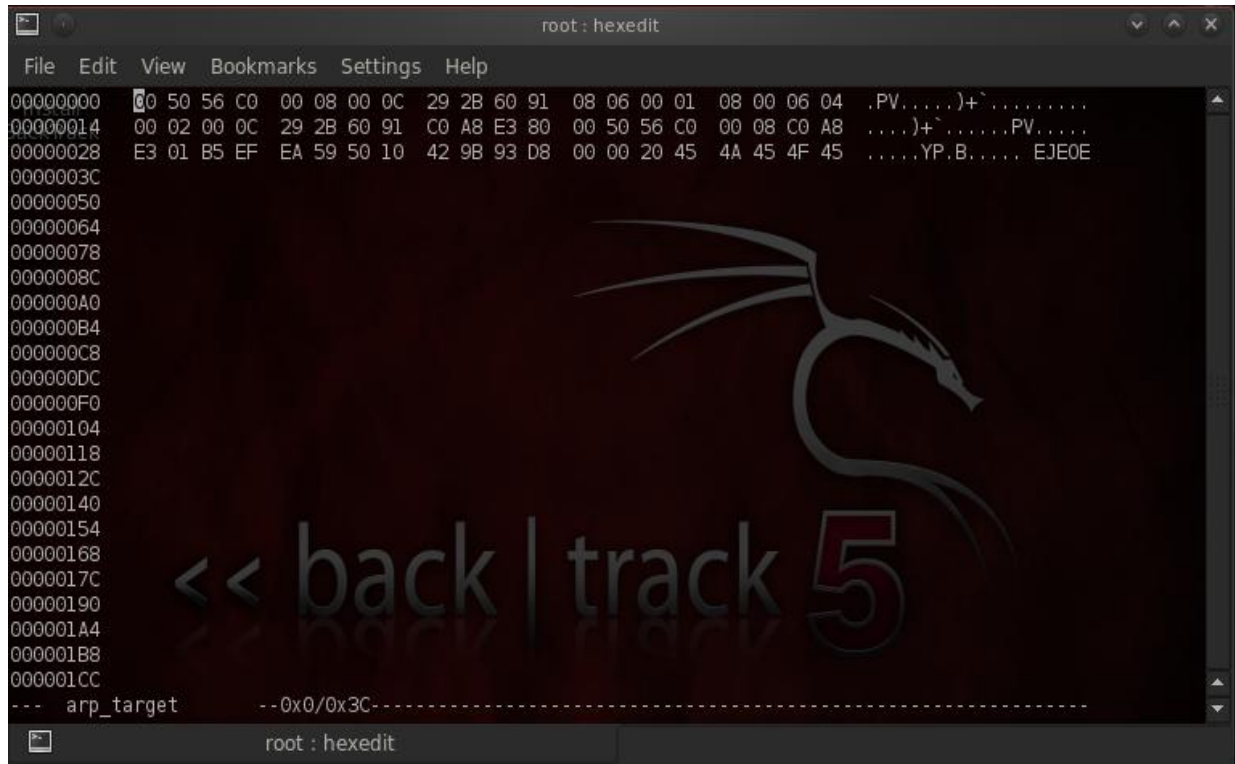


Figure 5.3.2: Hex editor for creating spoofed ARP reply

Attacker makes changes as following:

- Replace destination MAC address with Target host's MAC address
- Source field with Attacking computer's MAC

Changes to be made in reply packet:

- Replace sender's MAC address to Attacking computer's MAC
- Target MAC address to target host's MAC address
- Target IP address to target host's IP address

Now, with these changes we save our packet and sent it to the victim host using following command at attacker's terminal

```
File2cable -i eth0 -f arp_mod_packet
```

-i = network card interface

-f = file of packet (here arp_mod_packet)

Once the file is sent to victim it will update its cache with MAC of attacker corresponding to IP of gateway as below:

```
Interface: 192.168.1.7 --- 0x10003
Internet Address      Physical Address      Type
192.168.1.1          00-c0-9f-8f-3e-c1    dynamic
192.168.1.10         00-c0-9f-8f-3e-c1    dynamic
```

Figure 5.3.3: Poisoned cache of victim machine

Similarly, gateway is also sent a same kind of spoofed ARP reply packet. Its cache is also get poisoned. So the attacker acts as a system in between the communicating victim host and gateway. But the work is not yet completed since the attacker needs to act as a router also to forward the data coming from gateway to victim or vice versa. Otherwise victim may come to know that there is some problem if the data do not get to the gateway. In this way the attacker now can sniff all the data.

5.3.2 Using Ettercap tool:

Open Ettercap in your backtrack. Select the sniffing mode as Unified sniffing and give your network interface to be sniffed. Then scan for the host in your subnet.

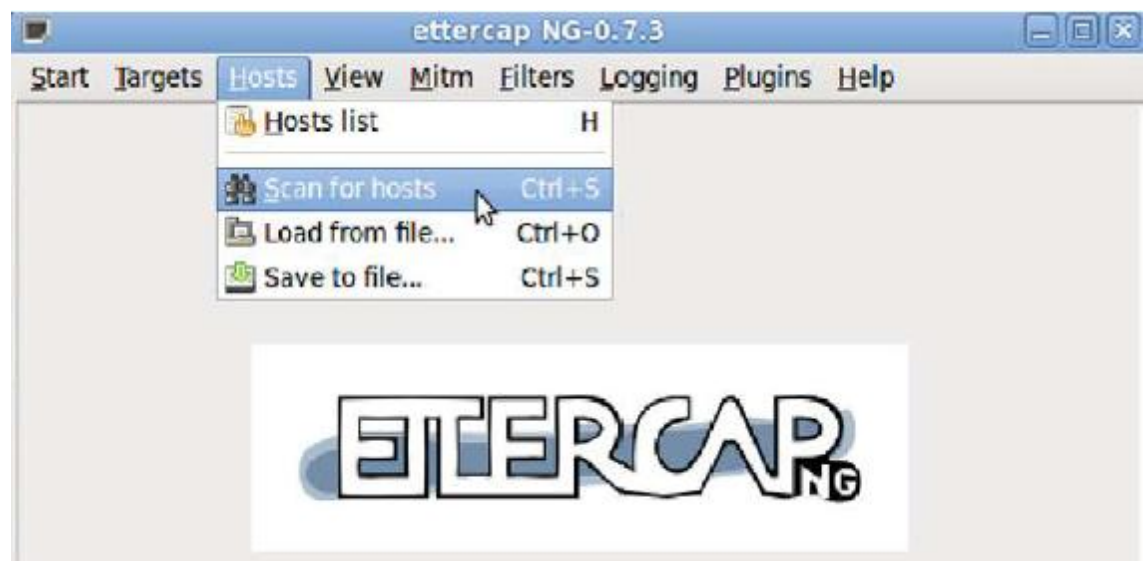


Figure 5.3.2.1 Scanning of hosts in the subnet

The discovered hosts by ettercap are shown as:

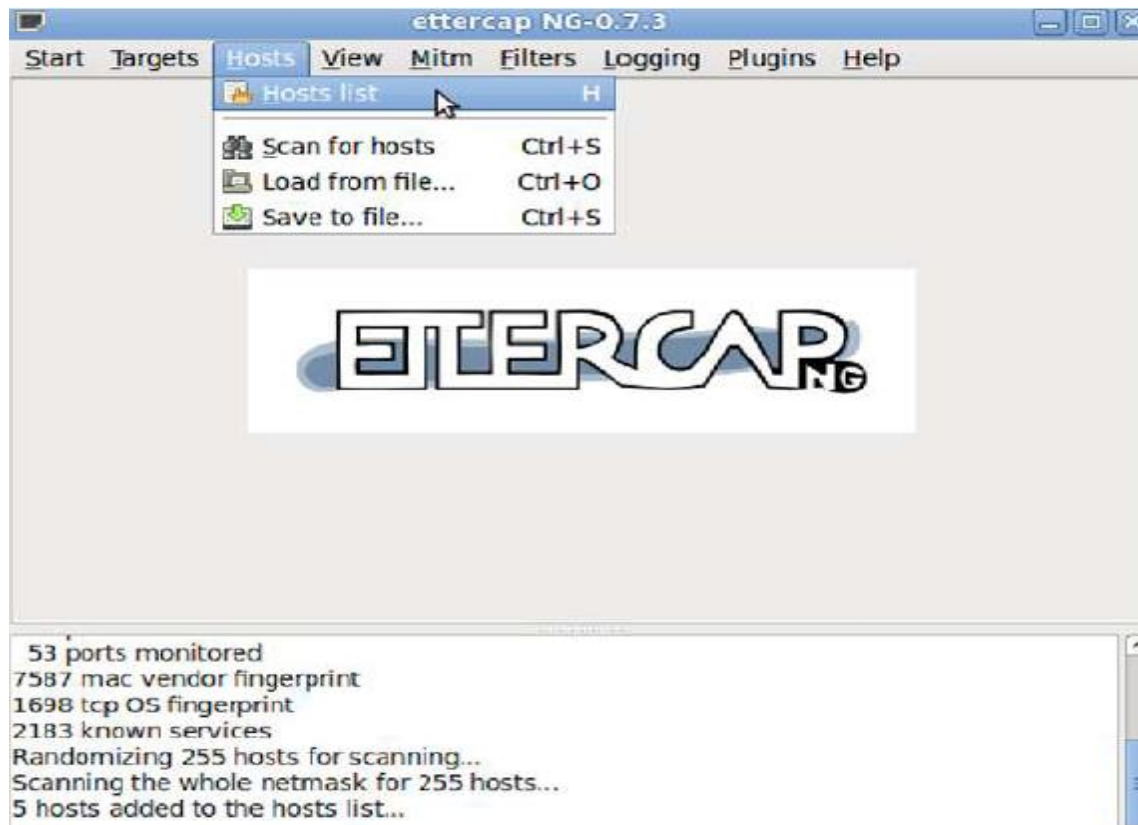


Figure 5.3.2.2 Five hosts discovered in the subnet

Corresponding hosts with their respective IP and MAC pairing is shown as:

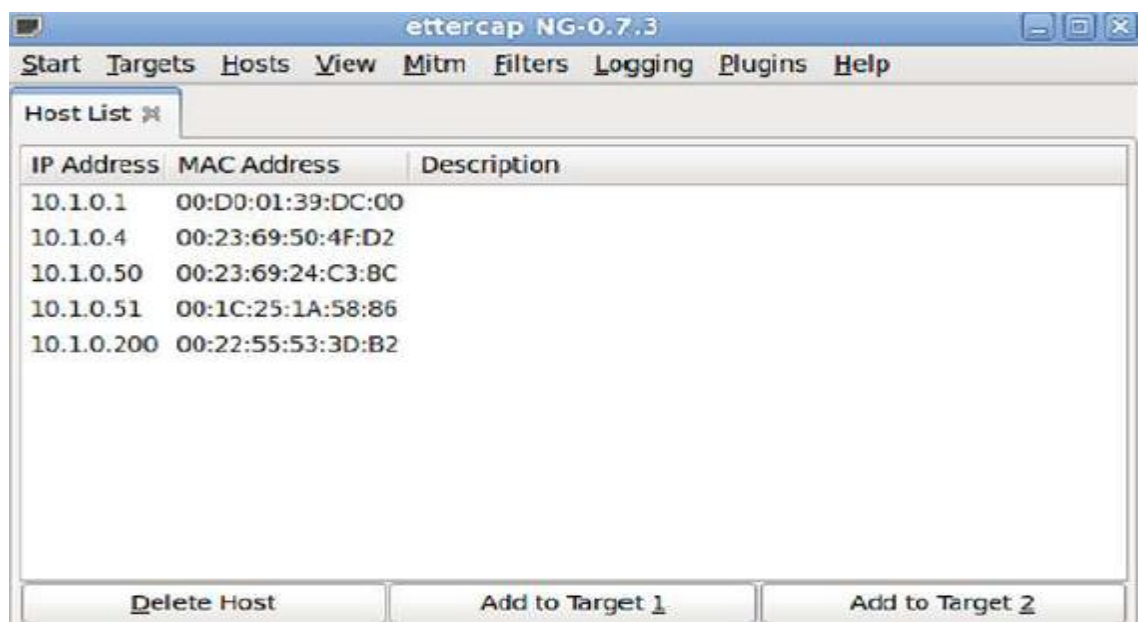


Figure 5.3.2.3 IP and MAC addresses of hosts

Now two target hosts are selected and added as Target 1 and Target 2. Victim with 10.1.0.1 IP is added to Target 1 and Victim with 10.1.0.51 to Target 2. For target 1 it is shown as below and for target 2, it can be done in similar way.

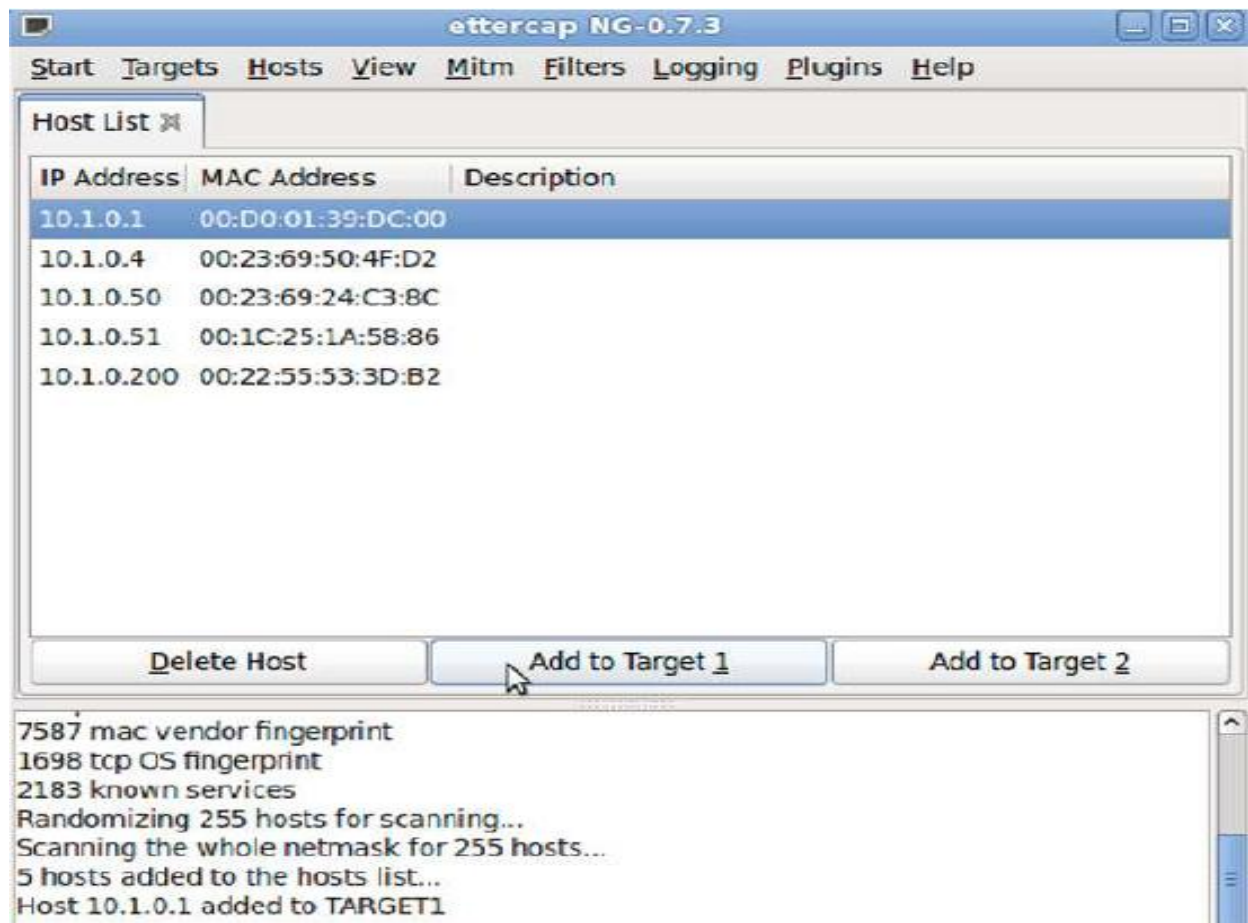


Figure 5.3.2.4 Target hosts are added

Now, after the targets have been added, sniffing of the data can be started. The 'start' drop menu is chosen before selecting 'Start sniffing'. The MITM attack is performed using ARP poisoning attack. So, choose 'MITM' drop down menu and select 'ARP Poisoning'. In optional parameters to select, choose sniffing of remote connections. In addition it can be seen while choosing ARP poisoning, we can also perform ICMP redirect, port stealing and DHCP spoofing also in the similar manner.

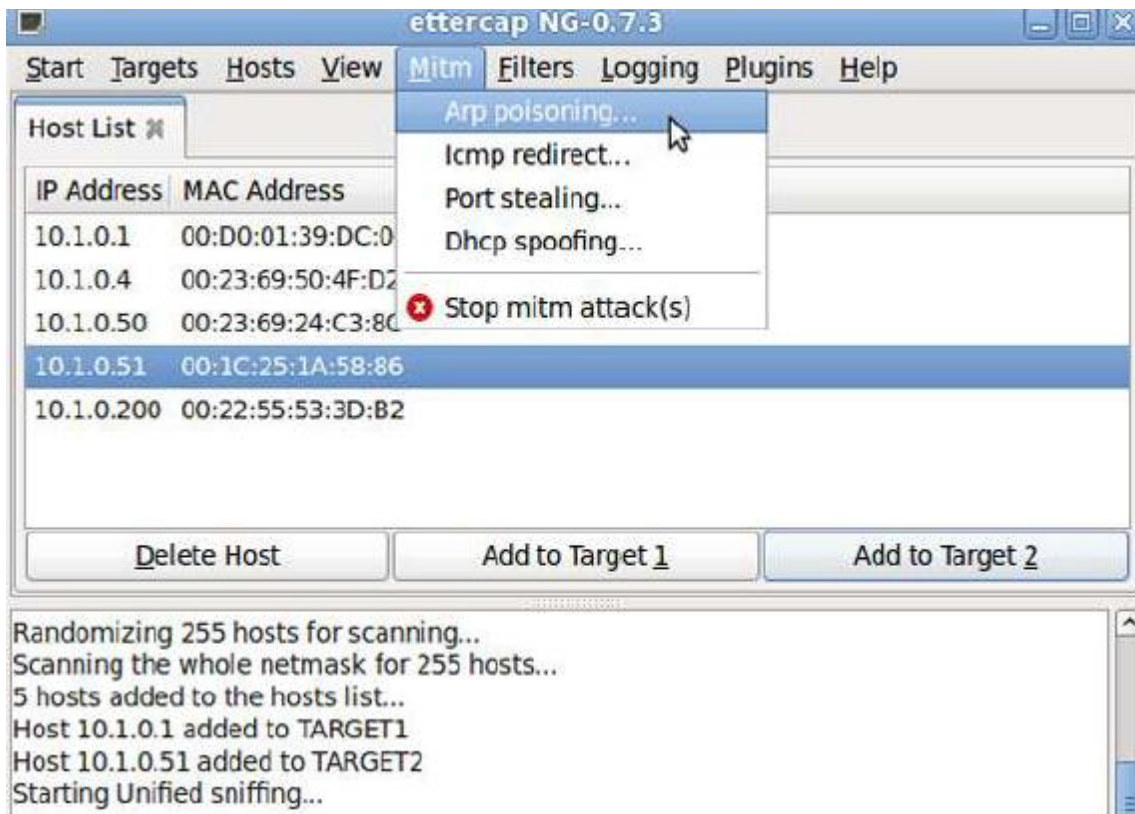


Figure 5.3.2.5 Choosing ARP Poisoning for MITM attack

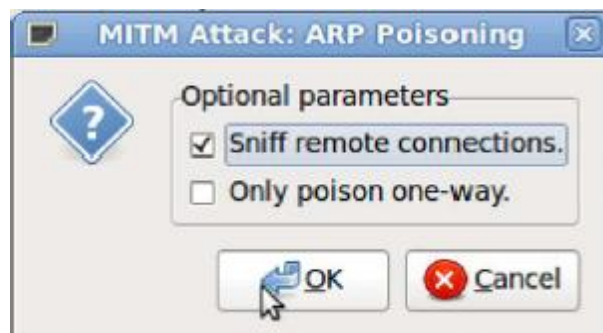


Figure 5.3.2.6 Selecting sniffing of remote connections

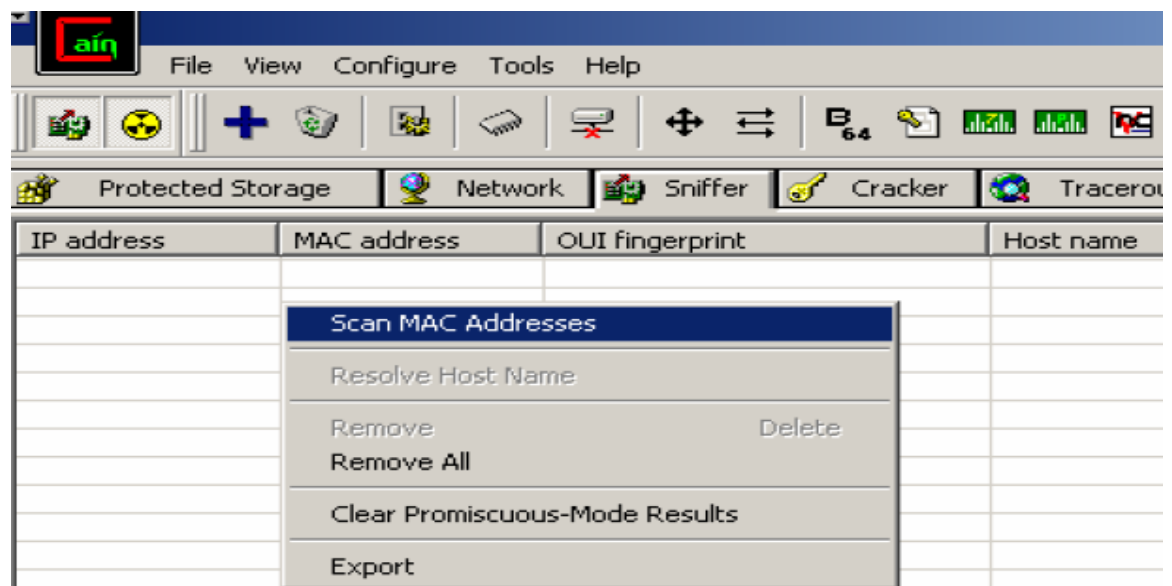
Then, a few more parameters are selected. They are 'repoison_arp' in the managing of plugins in the plugin menu. Next, 'remote_browser' parameter is chosen. Later, when we sniff the HTTP session of victim. This will enable the web browser of attacker to display the web pages of the victim that it will visit automatically. At the 'view' drop down menu choose connections. Once when the victim make connection, connection details could be seen. Here, victim tried a telnet session, and attacker can view details as shown below.



5.3.3 Using Cain & Abel:

To perform ARP spoofing using Cain & Abel, it is necessary to make sure WinPcap packet capture driver is properly installed and there must be 3 hosts present in the network to perform attack.

First we activated Sniffing and the ARP daemon. WE scan the MAC addresses in the network in the sniffer tab as:



Now, select the IP range according to your local area network subnet settings.

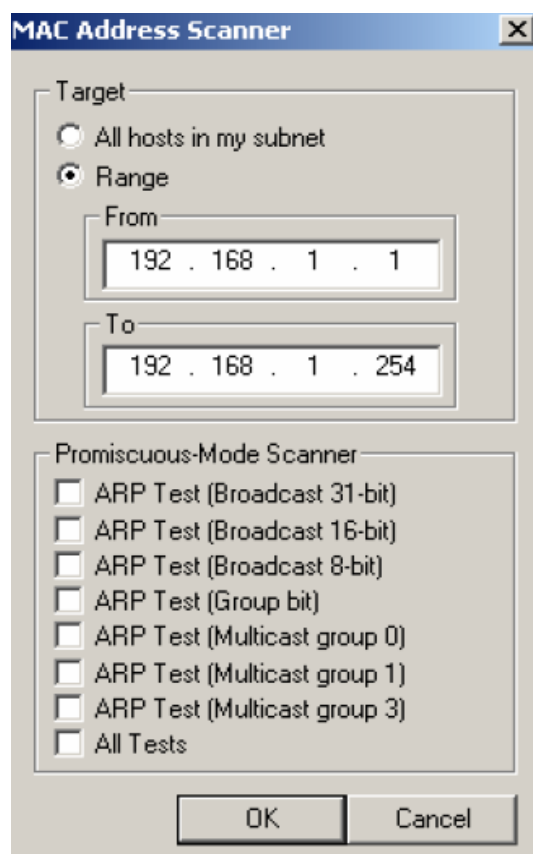


Figure 5.3.3.2 Selecting IP range in the network

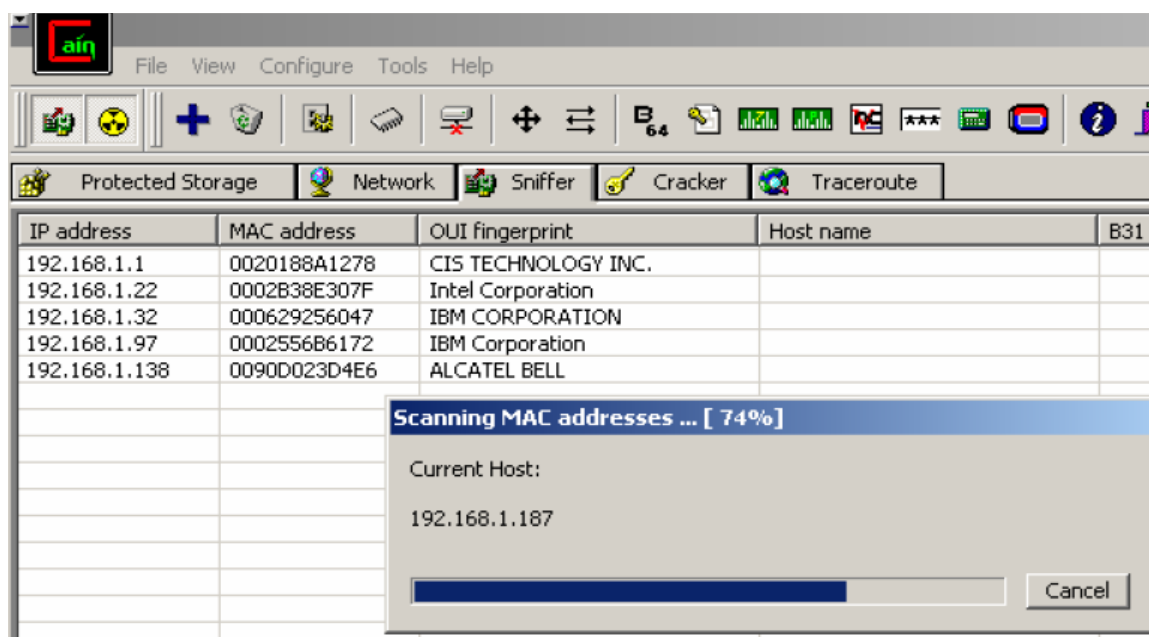


Figure 5.3.3.3 MAC addresses existing in the network

After scanning all the hosts in the network, add the victim in the ARP tab to poison. And a warning dialogue box appears as below:

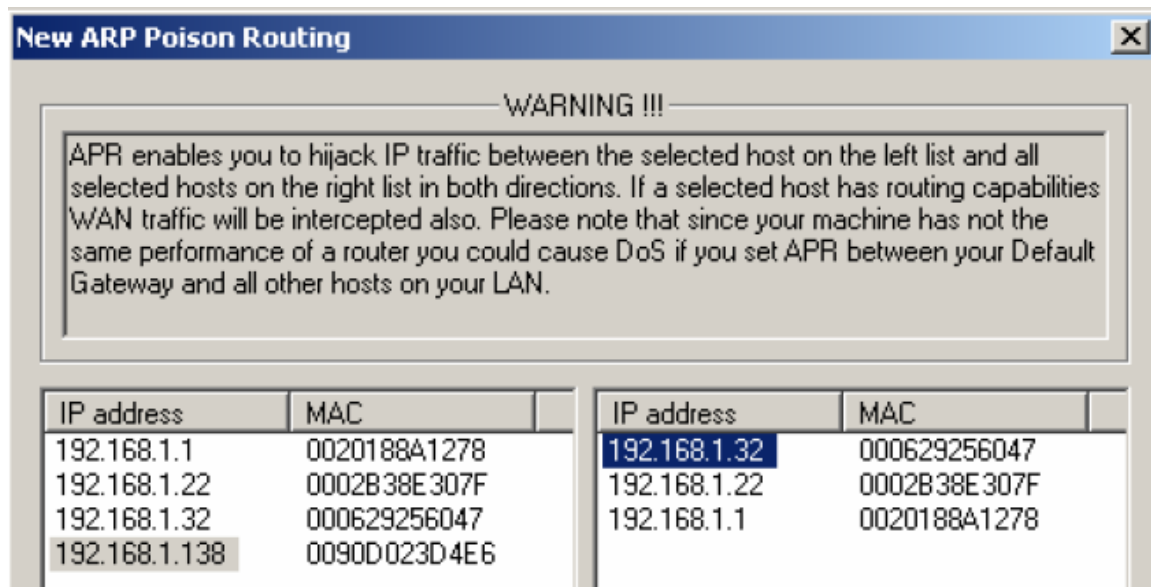


Figure 5.3.3.4 Selecting IP addresses that are to be poisoned

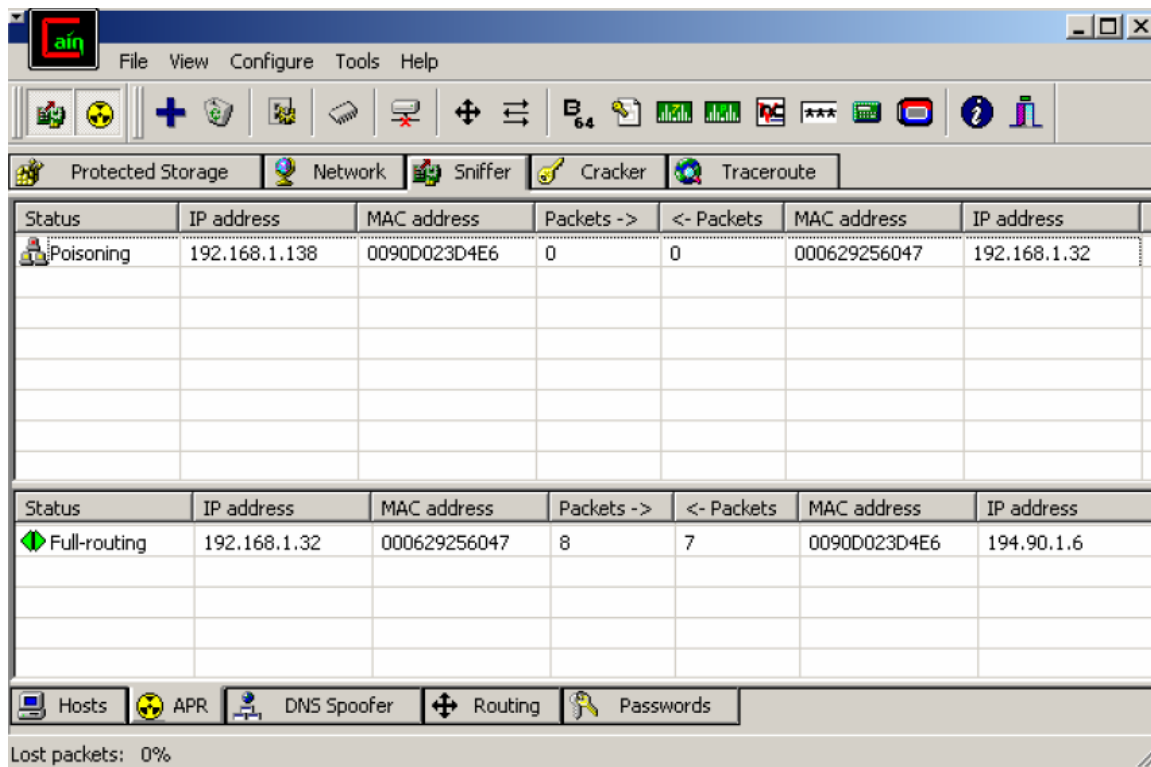


Figure 5.3.3.5 Poisoning the selected host

Now, once the hosts are poisoned, when ever victim enter any data using services like FTP, HTTP, POP3, IMAP etc. Here we have shown Capturing of data during a FTP session:

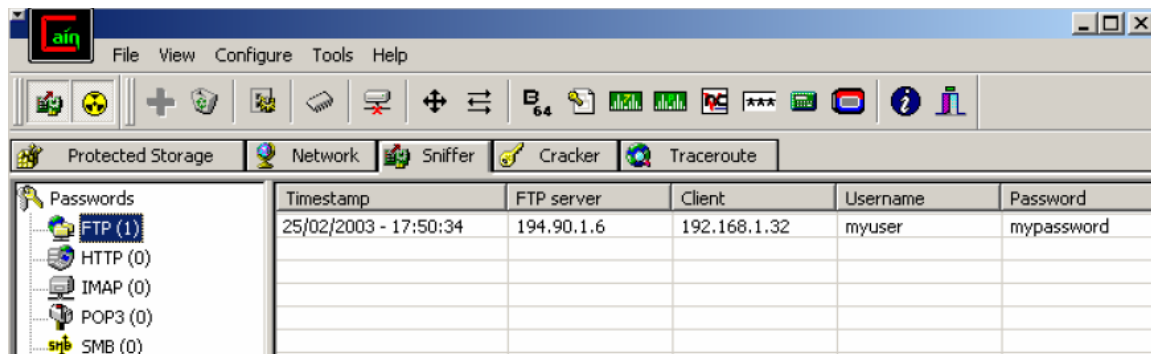


Figure 5.3.3.6 Capturing FTP user name and password

So, observing the attacks that we have described are quite easy to perform with the use of right tools and these are very easily available free of cost. Therefore, ARP poisoning poses a great threat to the crucial data endangering its integrity and confidentiality. So, it is very important to do something in order check ARP poisoning attacks on its cache. In the coming chapters, we have made an effort to provide some easy to use and implement methods to prevent poisoning in the first hand. And in case attacker somehow bypasses the preventions, it can be detected that which host in the network is the malicious one.

CHAPTER 6

ARP PREVENTION AND DETECTION TECHNIQUES

Here we have provided some easy to implement techniques in order to prevent ARP attacks of every kind. We have laid down the basic idea what is required by a system to protect itself from spoofing attack in the local area network. So, our mechanism needs to fulfill following things:

- It should be enough in itself to provide complete protection from all types of attacks on ARP
- It should have monitoring properties to keep in check the any malicious acts.
- Providing management function for the administrator for keeping the security of network intact.

ARP is very trusting and can really go on with the smooth running of its working. That is, to get MAC of a particular machine in the network with its IP. Request for MAC address is broadcasted and reply is received. Now the problem arises whether the reply received is with a valid <IP, MAC> pairing or not. Is it OK to update cache with new entry or not. All the secure and reliable working depends on this. So, all the efforts are done to ensure the reply received is genuine one.

A. Protection:

The main working of protection general protection system is explained here. The two principles for working of protection mechanism are as; the first is that we trust the existing old ARP cache entries. The existing pairings in the ARP cache are considered to be correct and valid. The second is that we expect the mapping from the ARP reply. The IP-MAC address pairing which is received must be verified before updating it in the cache. If after checking it is found correct, and then updations can be made. If the checking results in invalid association, it cannot be trusted and should be deleted from the ARP cache. This checking feature is composed of two steps. The first step is by comparing the IPs. It is done by comparing the IP in the packet which is received with the IP present in ARP cache, we can easily identify whether there is any IP conflict. Because this comparison is done in the local system

and its speed is also fast. So it can be proved to be an effective protection against IP conflict. The new pairing could be added to the cache when the first step gets finished. The second step is validating the IP-MAC mapping. It can be accomplished by sending the ARP request packets to the new MAC updated which we just added to the cache, and then looking for whether the reply packet is received or not. By this we can identify DoS attacks and many of the spoofing attacks. Since, this step is done with ARP cache, it could take time. Flow chart for the explained working is:

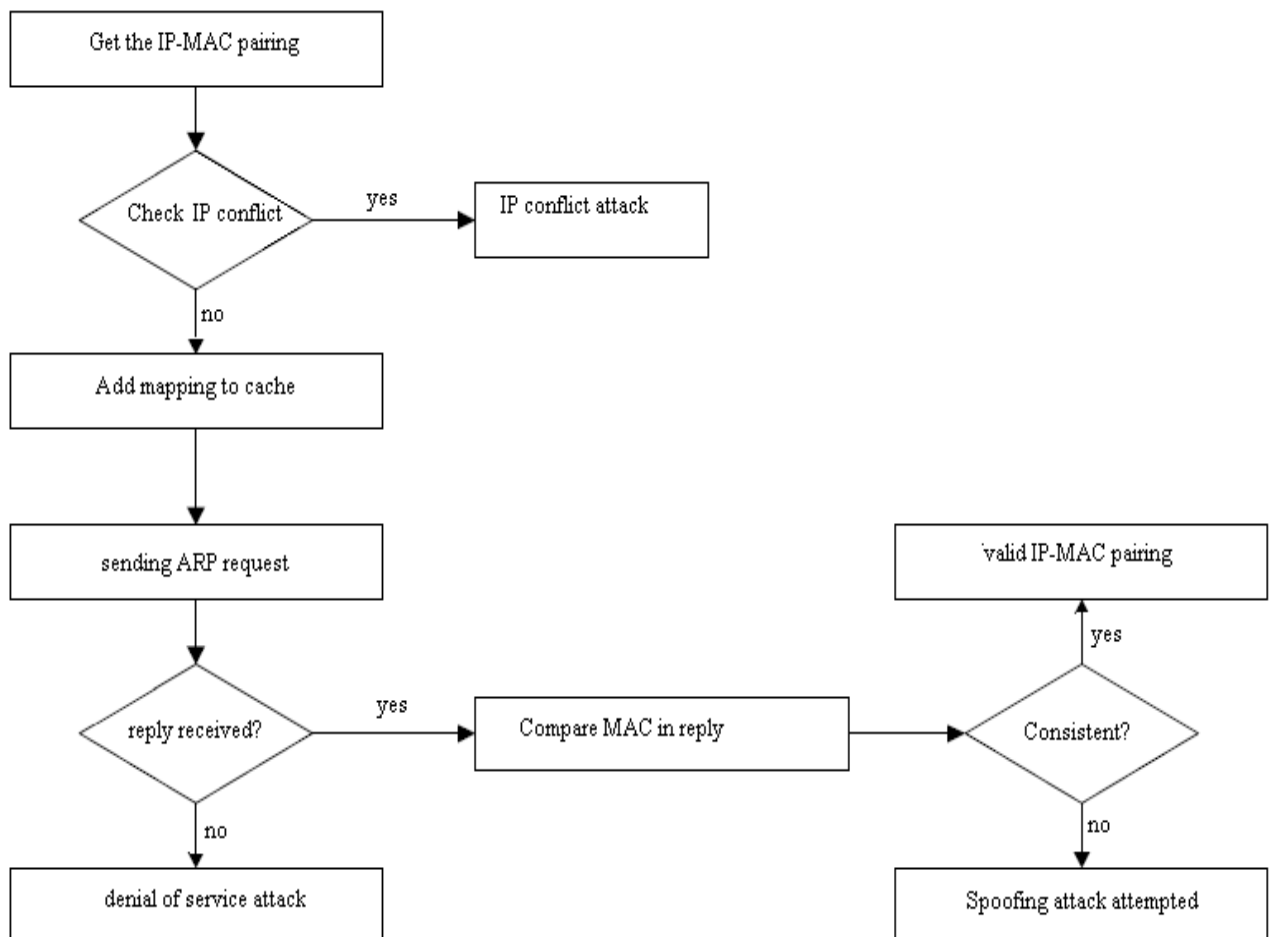


Figure 6.1: Protection mechanism

B. Monitoring for attacks:

This function is mainly adopted to keep constant eye on the malicious activities of the hackers in the local network. This could check the the bombing of data packets and provide warning against such attacks and alert the admin to take necessary steps. Here for packets we could set up a threshold value for the maximum number of packets that could exists in a time period. If at any point it is exceeded, a warning is issues to the administrator.

C. Management functions:

With this utility administrators can perform the management functions like that of manual management of ARP caches. For the system that are serving as important servers in the network, their corresponding entries could be made static, so that any attempt to change these pairing could not be made, other than the administrator himself. If any attempt made by unprivileged ones, it could possibly an attack. And timely actions can be taken.

6.1 Prevention Techniques:

- Using ARP packet request to the gateway:

It makes use of shell script by the user, who is aware enough to prevent themselves from the various attacks of modifying the cache according to their wish. Users keep these scripts running in the background. This provides satisfactory protection from attacks also. Shell script is as:

```
#!/bin/sh
While[1]
Do
    Arping -f 192.168.10.1
    Sleep 5
Done
```

Above shell script periodically keeps sending arping request to the gateway in order to keep the MAC address of the gateway system always in the

ARP cache. Arping is a kind of Broadcast request send to every user in the network. But here in the shell script we have restricted it to the gateway only and gateway would rely with its own MAC address. Since this is periodically running and hence hacker would not be able to poison the cache. If anyhow ARP gets poisoned, the script running would restore the original <IP, MAC> pairing.

- Monitoring arp –a table database:

A linux could make use of these shell script to monitor arp entry or arp table of the system. Any system always check its arp cache before sending message for delivering the data into the MAC layer. If in case there is a wrong MAC-IP association in the arp cache then, there is possibility of delivering data to a wrong user. And the system reels is under arp poisoning attack. Below is the script making use of awk command which controls the arp cache enteries of the user. It regularly checks the content of arp table at periodic intervals of time. In the script MAC address of gateway is being checked for possibility arp spoofing and it will alert the user if it occurs. On getting the alert, the user can take needed action by using different powerful linux command. The required shell scripts is as:

```
#!/bin/sh
while [ 1 ]
do
    a = `arp -a | awk '{print $4}' | grep 00:1b:d4:74:62:bf`
    if [ $a -gt 1 ]
    then
        notify - send -t 0 System is ARP poisoned
    fi
    sleep 9.5
done
```

This solution prevents from sending arp broadcast messages to the gateway. Rather than it checks the system's arp cache at periodic interval.

It generally checks that how many IP addresses get associated with the MAC of gateway. If number of such associations is more than 2, it alerts the user that ARP is being poisoned. But this solution also needs the user to know the gateway MAC beforehand. In case there is any change made by the network administrator in gateway, then user needs to change the entry. Once any poisoning is done, the user can use 'arp - d' command for deleting the arp poisoned entry.

- Using IDS Snort:

By changing the Snort configuration file that is snort.conf, snort can detect the ARP spoofing attacks, specifically as follows:

open the snort.conf, locate the following two lines:

```
#preprocessor arpspoof
```

```
#preprocessor arpspoof_detect_host 192.168.40.1  
f0:0f:00:f0:0f:00
```

At first, remove the front "#", then modify as follows:

```
preprocessor arpspoof
```

```
preprocessor arpspoof_detect_host: host_ip host_mac
```

```
preprocessor arpspoof_detect_host: gateway_ip gateway_mac
```

By snort we get to know spoofing attack being performed, and by using our detection technique we can find the malicious hosts.

- Storing the IP-MAC during Broadcast:

From the earlier discussion on ARP it has been observed that the main problem with ARP is that it is a trusting protocol. Here in this solution we made not only the request to be broadcasted but the reply also to be broadcasted. Here, we are trying to make our protocol work as a statefull protocol, by storing the information from the request I the ARP cache. In this when the request and reply are broadcasted, the systems in the network except the host will store pairing corresponding to

host in the ARP cache during both ARP request and ARP reply. Procedure of this working is as in the following steps:

1. Suppose a Machine A wants to communicate to D, but A is only aware of IP address of D.
2. So, A broadcasts a ARP Request with IP address of D.
3. All the systems on the LAN receive the ARP Request and update their ARP cache with MAC address of A.
4. Now, D recognizes its IP and replies with its MAC address by broadcasting ARP Reply.
5. All systems add the MAC of D to their respective ARP caches.
6. Now A can talk to D and can send data packets.

With this change of sending ARP reply in broadcast fashion, working ARP has proved to be much more secure and efficient. According to this, whenever any ARP reply is received by any host, it will check in its local ARP cache whether the entry corresponding to destination host entry is present or not. If the entry is already present, then source host will accept the reply, otherwise it will simply be discarded. It will also check for the valid combination of IP and MAC addresses of the destination in the ARP Reply frame before making the Reply frame go for broadcast. Also in this working updating of ARP caches will take place two times. First is when ARP request is broadcasted (during this IP and MAC of source host will be stored) and the second time when ARP reply broadcasted (during this IP and MAC of destination host will be stored).

ALGORITHMS AND FLOWCHART

The following pseudo code show how the communication takes place between the source host and destination host that is by procedure of broadcasting both the ARP request and ARP reply.

Procedure: (Source \rightarrow Destination)

BEGIN:

if (ARP cache contains MAC address) *then*

 Data packet will be delivered directly to the destination host

else

 Broadcast ARP Request Frame in the network

if (the source network has destination host) *then*

 Broadcast the ARP Reply Frame

else

if (not source itself)

 Update the ARP cache

else

 Destination host is in other network

if (routing table has entry) *then*

 Broadcast the ARP Request in destination network

else

 Update the Routing Table using ARP Frame and recheck

if (the destination network has destination host) *then*

if (IP and MAC of destination host in ARP Reply is valid) *then*

 Broadcast the ARP Reply Frame

else

 Invalid combination

else

if (not source itself)

 Update the ARP cache

else

 Router will forward the frame to another network

if (the Routing Table contains correct source network address) *then*

 ARP Reply Frame will be broadcasted in the source Network.

if (the MAC address in the frame is same as destination host MAC) *then*

Deliver the message to the destination host
else
Host not found
if (the source host match found) *then*
 Acknowledge delivered successfully
END: //end of procedure

The flow chart

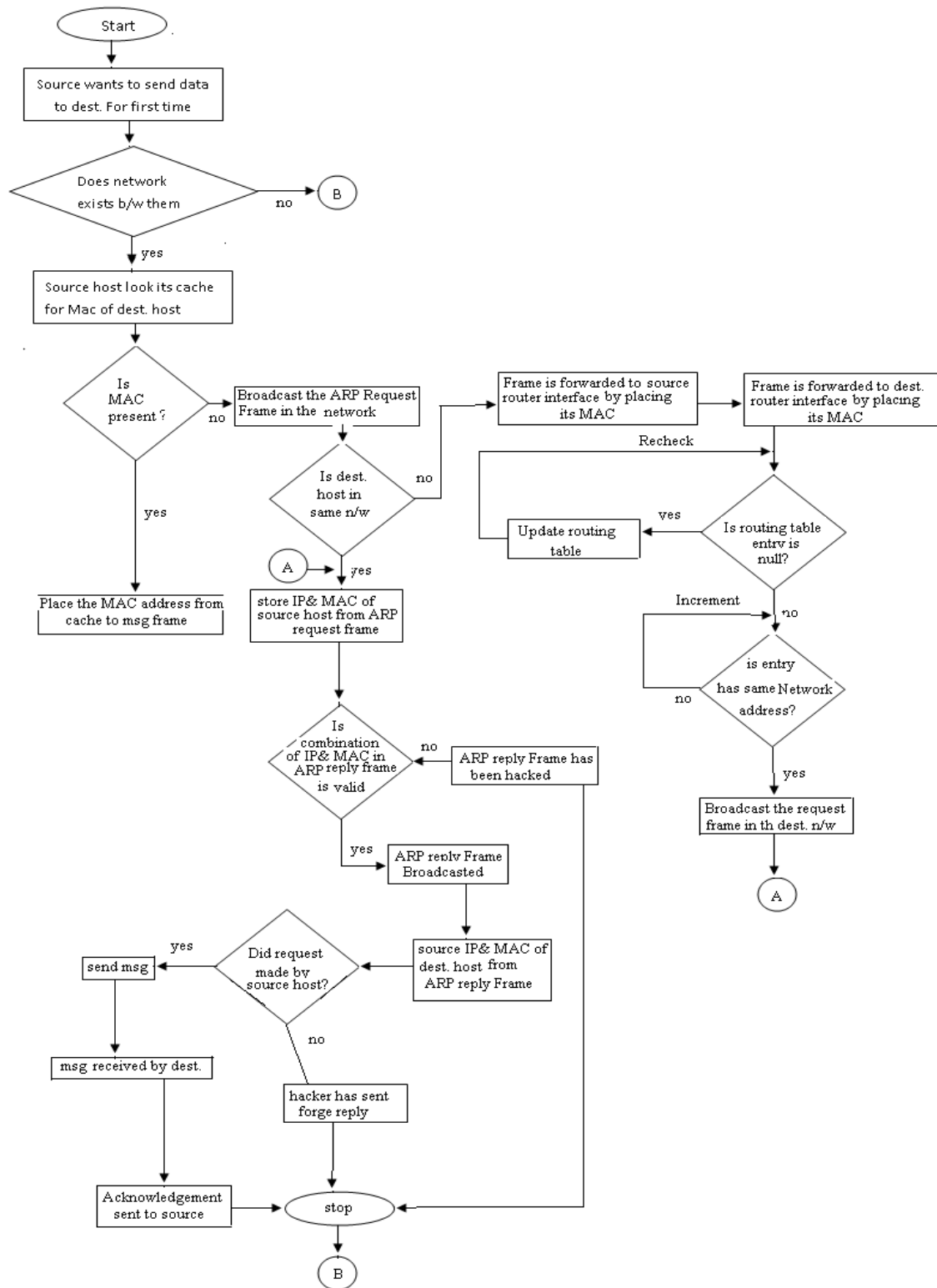


Figure 6.1.1 Flow chart of procedure

6.2 Detection Technique

Since most of the attacks on ARP cache is done by using MITM attack, the attacker being in the same network. Here we are going to show how such an attacker can be detected timely and we can cause denial of service to such system and take necessary action against the hacker.

Generally, the normal hosts on a LAN segment do not have IP packet routing enabled on their systems. Since the data packets they send and receive are routed using a router device. Therefore, if any host is detected with IP packet routing enabled, then that host can be taken as a suspicious host. However, we still need to prove that, that particular suspicious host is the one who has performed ARP cache poisoning attacks on the LAN hosts considering any measures are taken against that host. In the coming two sections, we discuss technique for detecting such hosts in the network who all have IP packet routing enabled on their systems. And, then identifying the actual malicious hosts who have performed ARP cache poisoning attacks.

A. Detection of hosts with enabled IP packet routing

The following is the technique that has been used to detect any host with enabled IP packet routing in a network. This technique mainly consists of two phases and assumption is made that there is one host in the network that is the Test host, which used to do all the tests during these phases.

Phase 1: Generation of ICMP echo request packets for trap.

In this phase, our Test host in the network will send a trap ICMP Ping packet to a target host. Generally, if any host A, with IP address IP_A and MAC address MAC_A wants to Ping a host B, with IP IP_B and a MAC address MAC_B, then host A is required to send an ICMP echo request packet to host B. A Ping packet is meant to check the connectivity of a particular host to the network. So, Test host tries to Ping itself and all other hosts in the network, by using a trap ICMP echo request packet. Therefore, “Destination MAC address” field in the Ethernet header for ICMP Ping packet (trap) is set to be the MAC address of the target host. Whereas, the “Destination IP address” field

in the IP header of the packet is the IP address of the Test host. Other values of the fields of the trap ICMP Ping packet are as in the below figure.

Ethernet header	
Source MAC address	MAC Test host
Destination MAC address	MAC Target host
Ethernet Type	0x0800 (IP message)
IP header	
Source IP address	IP Test host
Destination IP address	IP Test host
ICMP header	
Type	8 (echo request)
Code	0

Figure 6.2.1: Trap ICMP ping packet[29]

Every target host on the network will get a trap ICMP Ping packet that is sent to them by Test host. Since, “Destination MAC address” field of trap packet matches the MAC of the target host, so the packet will be accepted and sent to the IP layer by the target host for processing. But, “Destination IP address” field does not equate to the IP address of the target host, therefore, the target host will either discard the packet or forward that packet to the host whose IP address would be specified in the “Destination IP address” field. If the target host the one who has its IP packet routing enabled, the packet will be forwarded to Test host, since “Destination IP address” has IP of test host, otherwise, the packet will be discarded.

We observed that the target host who all has Packet routing enabled will forward the received trap ICMP Ping packet. When the target host will forward the packet, what it will do, it will forward the received original packet with the same IP and ICMP headers, but will change the Ethernet header, as a router is supposed to do. The destination MAC address field will have MAC of Test host, and the source MAC address will be the MAC of target host.

Phase 2: Detection of hosts with enabled IP packet routing.

All those hosts who will be acting as router will for sure route the trap ICMP packets to the Test host. Consequently, they will be considered as suspicious hosts. For

capturing the forwarded packet re-routed by suspicious hosts, the Test host will use a Sniffer like Wireshark to capture ICMP Ping packets whose “Destination IP address” field will match the IP address of the Test host.

B. Detection of ARP cache poisoning attack

Here, we discuss a technique for ensuring who among the suspicious hosts, have actually performed ARP cache poisoning attack on other hosts on the network. If any host who has enabled IP packet routing and also has performed ARP cache poisoning attacks on other hosts on the network, then they must also be the ones sniffing data that will be travelling in the network. This technique will act as, it will first corrupt the ARP cache of a suspicious host so as to make it to forward the data packets that it is capturing from the victims to the Test host. Then, Test host will analyze the forwarded data to identify whether the suspicious host has performed the attack or not.

To illustrate this mechanism, we assumed that A, B, C, D and E are 5 hosts on the LAN. Hosts A and B are communicating and sharing data packets. Host D is the suspicious one and has its IP packet routing enabled. Also, host D is the one who has poisoned the ARP caches of hosts A and B, for sniffing their data traffic. However, ARP cache of host C is not poisoned; being it not targeted by the malicious host D. Let Host E be our Test host. In earlier discussion it has been described how we can detect that Host D is the suspicious host.

The initial <IP, MAC> pairings in the respective ARP caches of hosts A, B, C and D, before Test Host corrupts the cache of Host D are as follows:

ARP cache of host A (Host A has poisoned cache) i.e. <IP_B, MAC_D>

ARP cache of host B (Host B has poisoned cache) i.e. <IP_A, MAC_D>

ARP cache of host C (Host C is not poisoned):

<IP_A, MAC_A> and <IP_B, MAC_B>

ARP cache of host D:

<IP_A, MAC_A> and <IP_B, MAC_B>

For every suspicious host, we will first poison its cache, using the ARP cache poisoning attack. For that, Test host E sends spoofed packets to the suspicious host D so

that all the cache entries of host D will point the MAC of Test Host. After poisoning of cache of Host D, its ARP cache has:

<IP_A ,MAC_E>, <IP_B, MAC_E> and <IP_C, MAC_E>.

Consequently, all the data packets that the host A sends to host B will go first to the host D, since the ARP caches of both hosts A and B have poisoned by Host D. But, then host D will also forward the received data packet to the Test host E, since its own cache has also been poisoned by the Test host E. Finally, Host E forward that data to the host it is meant for, that is Host B. Now, by analyzing data packets that we get from Host D, we can reveal that the source IP in the IP header of the packet is that of host A, but the source MAC in the Ethernet header is that of the suspicious host D whereas it should have been equal to the MAC of host A. By doing this, we have proved that the suspicious host D in the network is the one who has poisoned the cache of host A for sniffing of the data traffic. If the Test host receives packets from the host D, that is the packet with source IP, D's IP and the destination IP is that of E's IP, then we could say that the suspicious host D has only enabled IP packet routing but it is not sniffing any data traffic. This whole process above described process is repeated for all other remaining identified suspicious hosts.

CHAPTER 7

RESULTS AND PERFORMANCE

7.1 FOR PREVENTION MECHANISM

The procedure has been proved to be quite secure and efficient.

Mathematically:

Suppose there are N number of hosts in a LAN, the required number of transactions for updating of ARP cache of all hosts is given by,

If N is even, No. of transactions = $N/2$.

If N is odd, No. of transactions = $(N+1)/2$.

In ARP that is existing,

No. of transaction = $N(N-1)/2$.

Method of broadcasting the ARP reply helps in proving security against ARP cache poisoning also. As if any attacker tries to send spoofed ARP reply packets to any host in the network, then this reply would also be received by the targeted host whose IP address is getting used to map with MAC of attacker machine.

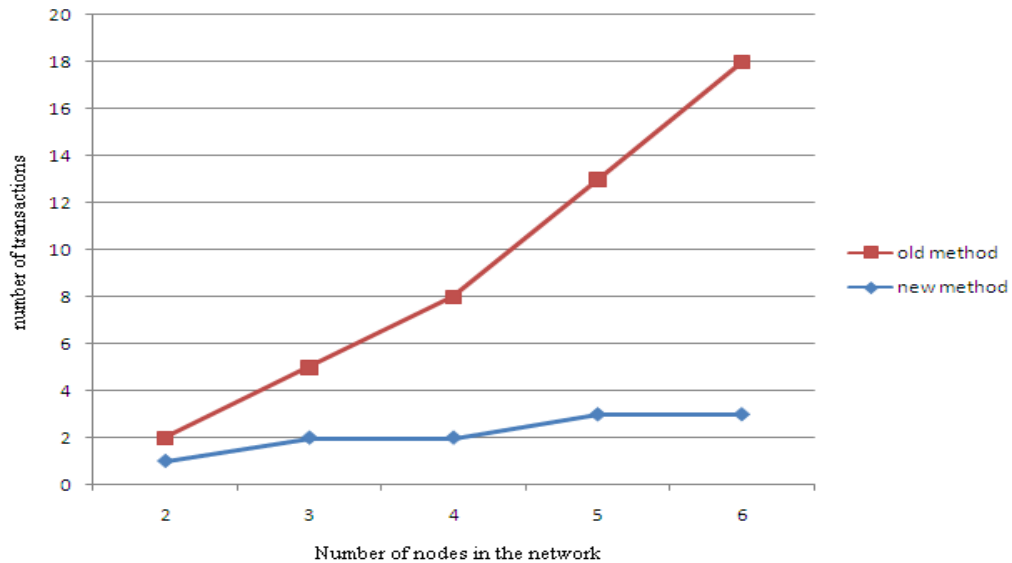


Figure 7.1.1: Efficiency in terms of transactions

So, this host gets to know that this ARP reply is spoofed one. And the system corresponding to this Mac is the attacker. **Hence, we can say that the idea of broadcasting the ARP reply is more secure and efficient also.**

It is feasible also since it does not require any additional installation of new system in the network. No cryptography is used, so no degradation in the performance.

7.2 FOR DETECTION MECHANISM

In this section we are analyzing the effect of the mechanism used in the detection mechanism on the overall performance of the network. Our detection system works by sending first trap ICMP packets and then the spoofed packets to the suspicious hosts and then also collects the response packets from them. Therefore, for calculating the efficiency of the mechanism, it is necessary to compute the approximate number of packets that we inject in the network. If the network gets flooded with these packets only, then it may negatively affect the performance of network.

Suppose there are n hosts in our network including Test host that was used to perform all the tests. The mechanism first detects the malicious host among the n hosts that has enabled their IP packet routing. Let us assume that there are m hosts which are suspicious hosts. So, the Test host will send $n-1$ trap ICMP packets to the hosts of network except itself. Then, only m hosts will send the received packet back since they will have enabled IP packet routing. Therefore, in total $((n-1)+m)$ packets get injected into the network by the Test Host, during the process of detecting the hosts who all have enabled IP packet routing.

Then, the mechanism tries to find who have actually poisoned the hosts, among those m suspicious hosts. For the Test host send spoofed ARP request packets to these hosts to corrupt their ARP cache, to force them to send the data they are sniffing back to the Test Host. We will have at maximum of $(n-1) \times \langle \text{IP}, \text{MAC} \rangle$ pairings in any ARP. Therefore, the Test host will have to generate $((n-1) \times m)$ spoofed packets. Also ARP cache gets itself refreshed after a periodic time, so to

keep the ARP cache corrupted, our Test Host is required to send the packets regularly. So, as long as the Test host keeps sending these packets regularly, the suspicious hosts would not broadcast any new ARP requests, since it will have its cache entries within the timeout threshold. If we take that Test host sends spoofed packets after every 10 seconds, then in 1 minute, it will eventually inject $((n-1)*m)*6$ packets into the network.

We have observed in general that the number of suspicious hosts in the LAN does not exceed 5; otherwise the network would not be reliable if more hosts get suspicious. The following Figure 7.2.1 shows the corresponding number of packets injected into the network by the Test Host with varying number of the hosts. It also depicts that the number of injected packets in the network increases in direct proportion to the number of hosts in the network. It also shows that as long as the number of suspicious hosts does not exceed 5, the mechanism of detecting the ARP spoofing in the network would not affect the performance.

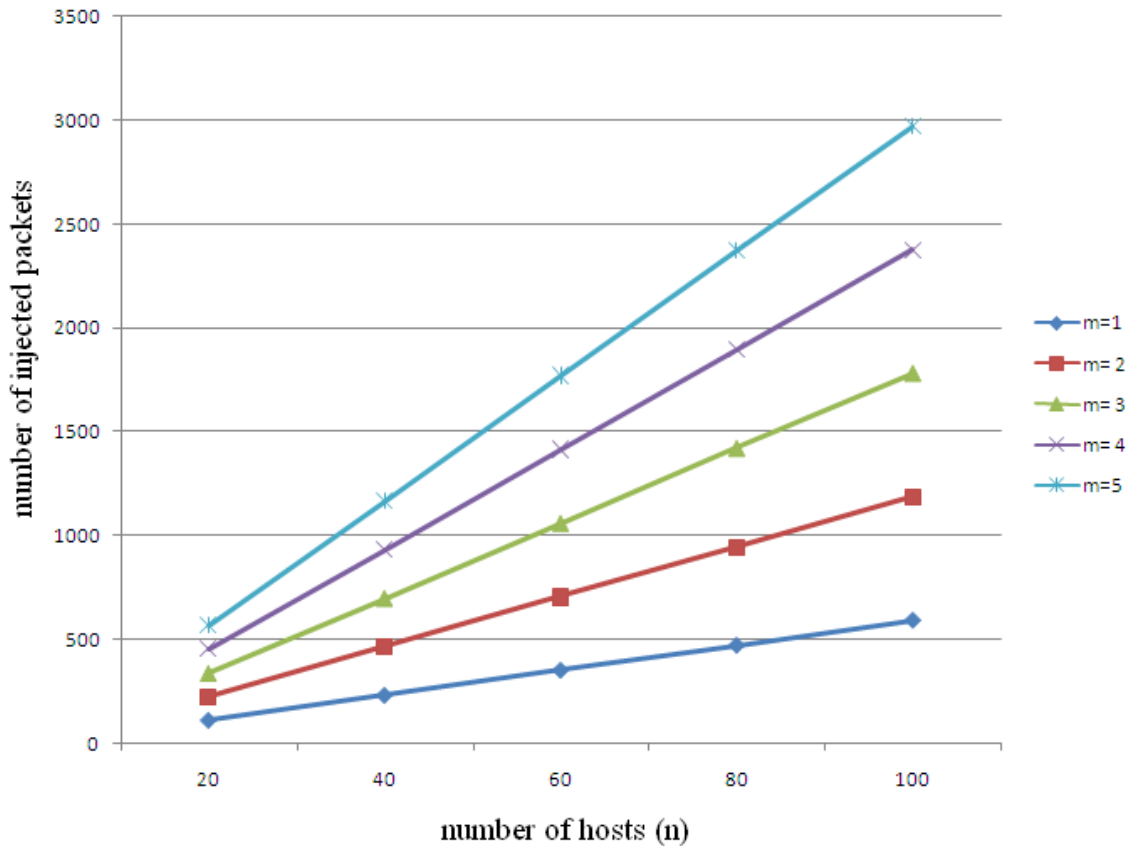


Figure 7.2.1: Number of packet injected to number of hosts

We have looked into another parameter to measure performance of mechanism that is the Round Trip Time (RTT). It is the response time any host takes to respond to the ICMP ping packets per second. Figure 7.2.2. Shows the different behavior of the hosts when the IP packet routing is enabled and disabled. However, even though in our network more packets are being injected while implementing the mechanism, but at the same time it will not degrade response time. Since, packets are no meant to be sent to only one host but rather sent to different hosts.

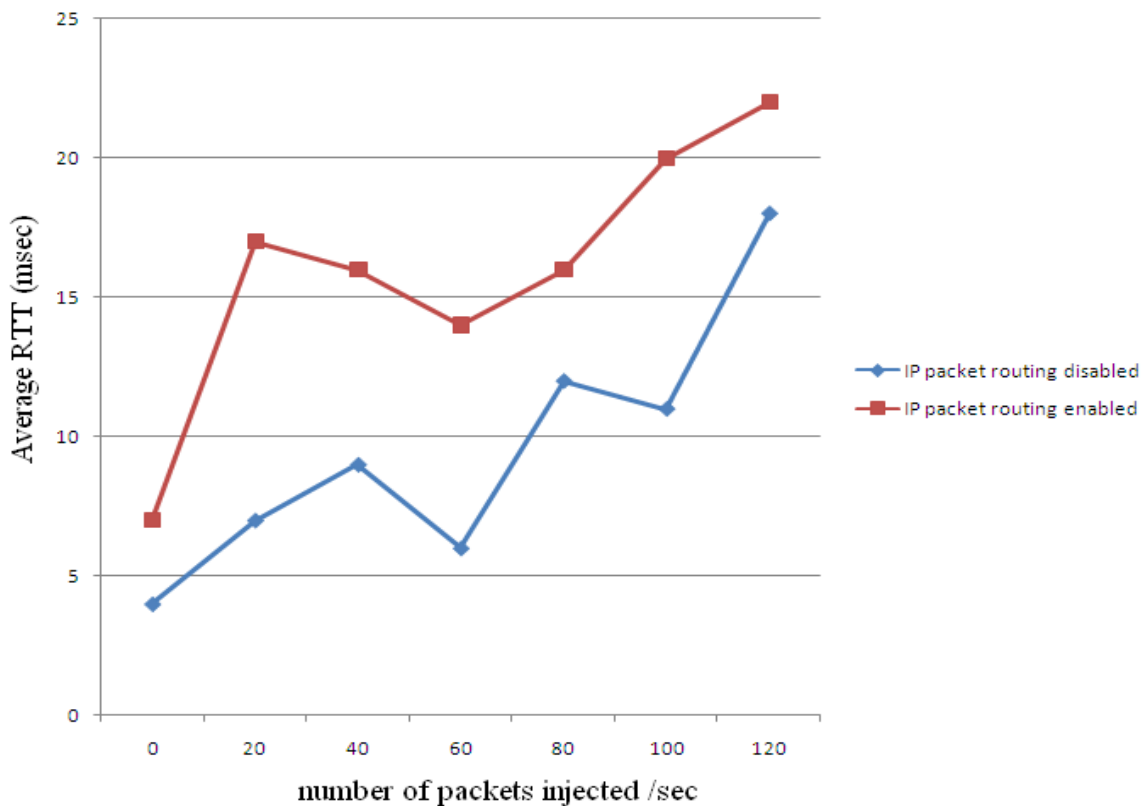


Figure 7.2.2: Measurement of RTT to IP packet routing enabled and disabled

Consequently, the use of described mechanism does not put any negative effect on the overall performance on the network as it does not flood the network with heavy traffic.

CHAPTER 8

CONCLUSION AND FUTURE SCOPE

From all the discussion we have seen that the matter of ARP spoofing is serious and it can lead to devastating results. With our mechanism that we have discussed, it has been shown that it is quite simple and easy to use. It could effectively prevent and detect the spoofing attacks being attempted in the Local Area Network.

Their effect on the working of normal network has also been analyzed and it does not have any performance degrading effects. Though, security do come at some cost, that we our self are injecting packets into the network. But its better than being under attack of getting our crucial data invaded. Different graphs are providing easy to analyze performance measures. These mechanisms are easy to implement also since these are not using any kind of cryptographic algorithms and does not take much execution time. These are feasible also and do not require any hardware implementation. And most importantly these could effectively detect and prevent spoofing attacks on the ARP cache.

Still there is a lot to be researched in this area. We analyze the working of mechanism with different operating systems. More work could be done on improving false positive/negative ratios, when different hosts in the network are using personal firewalls in order to filter the data for both inbound and outbound. And to develop a solution with the use of integrated technology that could minimize the risk posed by the ARP spoofing.

References

- [1]. Thomas Demuth, Achim Leitner, "ARP spoofing and poisoning", Linux magazine, issue 56, pp. 26-31, July 2005.
- [2]. Cristina L. Abad, Rafael I. Bonilla, "An Analysis on the Schemes for Detecting and Preventing ARP Cache Poisoning Attacks", 27th International Conference on Distributed Computing Systems Workshops, 2007.
- [3]. M. Tripunitara and P. Dutta. A middleware approach to asynchronous and backward compatible detection and prevention of ARP cache poisoning. In Proceedings of the 15th Annual Computer Security Applications Conference (ACSAC '99), Dec. 1999.
- [4]. M.M. Dessouky, W. Elkilany, N. Alfishawy, "A Hardware approach for detecting the ARP attack".
- [5]. D. Bruschi, A. Ornaghi, E. Rosti, "S-ARP : a Secure Address Resolution Protocol" in the proceedings of the 19th Annual Computer Security Application Conference, 2003.
- [6]. Somnuk Puangpronpitag, Narongrit Masusai, "An Efficient and Feasible Solution to ARP Spoof Problem". IEEE 2009.
- [7]. Thawatchai Chomsiri, "Sniffing packets on LAN without ARP spoofing". International Conference on Convergence and Hybrid Information Technology, 2008.
- [8]. Bhirud, S.G., Vijay Katkar, "Light weight approach for IP-ARP spoofing detection and prevention".
- [9]. Cisco Systems. Configuring Dynamic ARP Inspection, chapter 39, pages 1-22.
- [10]. Raul Siles, "Real world ARP spoofing", GIAC Certified Incident Handler (GCIH) Practical, Version 2.1a, august 2003.
- [11]. M. Gouda and C.-T. Huang. "A secure address resolution protocol", Computer Networks, January 2003.
- [12]. D. C. Plummer. "An ethernet address resolution protocol", IETF RFC 826, November 1982.
- [13]. Teerapat Sanguankotchakorn, Thanatorn Dechasawatwong, "Automatic attack detection and correction system development".

- [14]. Seung Yeob Narn, Dongwon Kim, Jeongeun Kim, “Enhanced ARP: Preventing ARP poisoning- based Man-in-the-Middle Attacks”, IEEE communications letters, vol. 14, no. 2 , February 2010.
- [15]. Haider Salim, Zhitang Li, Hao Tu, Zhengbiao Guo,” Preventing ARP Spoofing Attacks through Gratuitous Decision Packet”, 11th International Symposium on Distributed Computing and Applications to Business, Engineering & Science, 2012.
- [16]. Andrew S. Tanenbaum, “Computer Networks”, fourth edition.
- [17]. Andrew Lochart, “Network security hacks”, page 184.
- [18]. John Peter Jason, “Information security”, Magazine Ubiquity, vol. 2006 issue January, article no. 3.
- [19]. Thomas Baxley, Jinsheng Xu, Huiming Yu, Jinghua Zhang, Xiaohong Yuan, Joseph Brickhouse, “LAN Attacker: A Visual Education Tool”.
- [20]. Faheem Fayyaz, Hamza Rasheed, “Using JPCAP to prevent man-in-the-middle attacks in a local area network environment”, IEEE potentials, July/August 2012, page 35-37.
- [21]. Jin Cherng Lin, Men-Jue Koo, Cheng-Sheng Wang, “A Proposal for a schema for ARP Spoofing Protection”, ICET, 2012.
- [22]. Cisco Content Services Switch Routing and Bridging Configuration Guide, “Configuring the Address Resolution Protocol”, Chapter-4.
- [23]. Sumit Kumar, Shashikala Tapaswi, “A Centralized Detection and Prevention Technique against ARP Poisoning”.
- [24]. Zouheir Trabelsi, “Hands-on Lab Exercises Implementation of DoS and MiM Attacks using ARP Cache Poisoning”, Information Security Curriculum Development Conference 2011 October 7-9, 2011, Kennesaw, GA, USA.
- [25]. Libing Wu, Tianshui Yu, Dan Wu, Jing Cheng, “The Research and Implementation of ARP Monitoring and Protection”, IEEE 2011.
- [26]. Yang Liu, Kaikun Dong, Lan Dong, Bin Li, “Research of the ARP Spoofing Principle and a Defensive Algorithm”, International Journal of Communications, Issue 4, Vol. 1 2007.

- [27]. Md. Ataulah, Naveen Chauhan, “ES-ARP: an Efficient and Secure Address Resolution Protocol”, IEEE Students’ Conference on Electrical, Electronics and Computer Science, 2012.
- [28]. Haider Salim, Zhitang Li, Hao Tu, Zhengbiao Guo, “Preventing ARP Spoofing Attacks through Gratuitous Decision Packet”, 11th International Symposium on Distributed Computing and Applications to Business, Engineering & Science, 2012.
- [29]. Zouheir Trabelsi, Khaled Shuaib, “Man in the Middle Intrusion Detection”, IEEE Communications Society, IEEE GLOBECOM 2006 proceedings.
- [30]. Gopi Nath Nayak, Shefalika Ghosh Samaddar, “Different Flavours of Man-In-The-Middle Attack, Consequences and Feasible Solutions”, IEEE 2010.

PUBLICATIONS

1. Mr. Sumit Miglani and Inderjeet kaur, “Feasibility analysis of different methods for prevention against ARP spoofing” in International Journal of Scientific and Research publications, 3rd edition, 7 volume, 2013