ARP Spoofing And Mitigations

Mayada Amr El-Roumy 1722
Nourane Hanie Abdel-Moneim 1748
Hind Ahmed Tarek Ashour 1759

Compu-communication department
>>>>mayada.elroumy@gmail.com<<<<<
>>>>nouranehanie1992@gmail.com<<<<<
>>>>hindashour2011.ha@gmail.com<<<<
**Abstract:**

ARP (Address Resolution Protocol) is a fundamental protocol that’s responsible for link-layer message transmission. It’s a trusting protocol that was not designed to cope with malicious hosts but unfortunately these malicious hosts exist. Due to the trusting naïve behaviour of the ARP, hackers introduced ARP spoofing or ARP poisoning. This spoofing attack clears the road for other attacks like DoS and Hijacking. Many solutions were introduced, in the struggle to prevent ARP spoofing attacks, but unfortunately none of them gave ideal protection against attacks.

Some related work to the ARP spoofing is DNS spoofing, where DNS stands for Domain Name System. We’re going to discuss it from a UDP point of view.

In this paper, we’re going to give a brief explanation of ARP, what it does and how it works. Then we’re going to point out the weakness point that made life easier for hackers and list some attacks and how they work. After that we’re going to list some introduced solutions, illustrate their methods and point out why they didn’t work. Then give a limited comparison between these solutions based on a certain criteria.

**Keywords:**

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARP</td>
<td>(Address Resolution Protocol)</td>
</tr>
<tr>
<td>DNS</td>
<td>(Domain Name System)</td>
</tr>
<tr>
<td>MAC</td>
<td>(media access control)</td>
</tr>
<tr>
<td>LAN</td>
<td>(local area network)</td>
</tr>
<tr>
<td>IP</td>
<td>(Internet Protocol)</td>
</tr>
<tr>
<td>DoS</td>
<td>(Denial of Service)</td>
</tr>
<tr>
<td>MITM</td>
<td>(Man-in-The-Middle)</td>
</tr>
<tr>
<td>NIC</td>
<td>(Network Interface Card)</td>
</tr>
<tr>
<td>SSL</td>
<td>(Secure Sockets Layer)</td>
</tr>
<tr>
<td>SSH</td>
<td>(Secure Shell)</td>
</tr>
<tr>
<td>ICMP</td>
<td>(Internet Control Message Protocol)</td>
</tr>
<tr>
<td>DDoS</td>
<td>(Distributed denial of service attack)</td>
</tr>
<tr>
<td>TCP</td>
<td>(Transmission Control Protocol)</td>
</tr>
<tr>
<td>UDP</td>
<td>(User Datagram Protocol)</td>
</tr>
<tr>
<td>SARP</td>
<td>(Secure Address Resolution Protocol)</td>
</tr>
<tr>
<td>HMAC</td>
<td>(hash message authentication code)</td>
</tr>
<tr>
<td>DHCP</td>
<td>(Dynamic Host Configuration Protocol)</td>
</tr>
<tr>
<td>NIDPS</td>
<td>(Network Intrusion Detection and Prevention System)</td>
</tr>
<tr>
<td>IA</td>
<td>(Information Agent)</td>
</tr>
</tbody>
</table>
1 Introduction

The project gives detailed illustration to the ARP spoofing and how it is done and gives a few solutions to avoid the attack. We are selecting this topic because security attacks activity has grown in such a scary way that makes a safe connection without taking precautions almost impossible.

The main goal and reason why we are writing this topic is to give ourselves the proper knowledge that can help us protect our data and help others protect theirs. Our topic is about ARP spoofing and from what we learned about it, it is obvious that it can happen to anybody whose only mistake is connecting his device to a public network. Data can be stolen easier than you can imagine and without an alarm and still your connection with the sender/receiver will proceed like nothing was stolen!

The ARP (Address Resolution Protocol) was defined in RFC 826, by David Plummer in 1982, it’s a very old protocol and naturally it was created without any regards for security. ARP operates by sending out “ARP request” packets. An ARP request asks the question “Is your IP address x.x.x.x? If so, send your MAC back to me.” These packets are broadcast to all computers on the LAN, even on a switched network. The ARP reply specifies the 48 bit Ethernet address to use for that IP address.

First we are going to give you a brief background about ARP spoofing: What ARP stands for and what is the main role of the protocol. Then, we are going to explain the history of ARP, how it started and who started it. After that, we are going to explain how the ARP works to accomplish its mission.
2 Specification of the Project

2.1 Introduction to ARP:

ARP (Address Resolution Protocol) is a protocol that provides a mechanism to translate IP addresses to link-layer addresses (MAC addresses). Link-layer address is usually called a “LAN” address, a “physical” address or a “MAC” address.

It’s not hosts and routers that have MAC addresses but it’s their adapters (network interfaces). Therefore a host/router with multiple MAC interfaces will have multiple MAC addresses.

The MAC address is 6-bytes long (48 bits) giving $2^{48}$ possible MAC addresses. Although MAC addresses were designed to be permanent, it became possible to change the adapter’s MAC address via software.

Now let’s focus on the mechanism of ARP. Data propagates in ARP protocol in the form of frames and every frame that leaves a host must contain a destination MAC address. Therefore we need both IP and MAC addresses in order to send a frame from one host to another. This is when we use the ARP to obtain the host’s MAC so that the frame can be delivered on the network.

ARP follows the following steps in order to send a frame from a host to another:

1) The host needs to know the MAC address of another host, so it broadcasts an ARP request to the network “Whoever who has the IP x.x.x.x, tell your MAC!”

2) All other hosts receive that request and also all hosts discard it EXCEPT the host with the host with the mentioned IP address which replies with its MAC address.

3) The host that requested the MAC address receives the reply and caches the $\langle$IP,MAC$\rangle$ pairing in a local ARP cache or table to avoid making the same request in the near future.

4) To accommodate for hosts that come and go and for dynamic IP assignments, the ARP cache entries expire every 20 minutes and some operating systems reset the time every time they use an entry.

What makes the ARP a trusting protocol is that it’s a stateless protocol. This results in, when an ARP reply is received, the host updating its ARP cache even if the host hadn’t requested that MAC address. Therefore it can simply update its ARP cache by adding a new host or editing the IP of another just because it received some anonymous reply from some host. This resulted in a serious problem called ARP spoofing or ARP poisoning that we are going to discuss straight away.
### 2.2 Introduction to ARP Spoofing:

**ARP spoofing** (Also known as ARP cache poisoning or ARP poison routing (APR)) is a technique whereby an attacker sends fake (“spoofed”) Address Resolution Protocol (ARP) messages onto a Local Area Network.

Generally, the aim is to associate the attacker’s MAC address with the IP address of another host (such as the default gateway), causing any traffic meant for that IP address to be sent to the attacker instead.

ARP spoofing may allow an attacker to intercept data frames on a LAN, modify the traffic, or stop the traffic altogether.

Often the attack is used as an opening for other attacks, such as denial of service, man in the middle, or session hijacking attacks.

**The idea behind ARP spoofing** is to trick a target computer’s ARP cache causing it to send all the traffic through an attacking machine before returning back to the target computer.

### Background:

ARP is a trusting protocol and was not designed to cope with malicious hosts. There are several ways in which a malicious host can make an unsuspecting host modify its ARP cache to add/update an entry with an <IP, MAC> mapping to enable the attacker to impersonate another host, perform man-in-the-middle attacks, gain access to sensitive information and perform Denial of Service (DoS) attacks.

When a host adds an incorrect <IP, MAC> mapping to its ARP cache, this is known as **ARP cache poisoning (or simply ARP poisoning)** or **ARP spoofing**. The last terminology refers to the fact that an attacker uses fake or “spoofed” ARP packets to poison an ARP cache.

In an ARP cache poisoning attack, the attacker sends ARP replies with fake <IP, MAC> mappings, in an attempt to poison the ARP cache of other host(s) on the network.

For example, if the attacker wants to impersonate host X so that host Y sends data destined to X to the attacker instead, the attacker can send an ARP reply indicating that the host with the IP x.x.x.x has the MAC mm:nn:mm:mm:mm:mm (the MAC address of the attacker).

Since **ARP is an stateless protocol**, the receiver will gladly update its ARP cache with the <IP, MAC> pairing received. Furthermore, some operating systems may even update static cache entries with information received from unsolicited ARP replies.

**Even if ARP is configured to be stateful**, an attacker can still perform an ARP spoofing attack by sending a fake ICMP (Internet Control Message Protocol) echo request to Y indicating it comes from X, but using the MAC address of the attacker.
Depending on the implementation, the operating system can either use the <IP, MAC> pairing inferred from the received Ethernet frame and ICMP packet, or it can issue an ARP request to learn the mapping (before sending the ICMP echo reply).

The effects of ARP spoofing attacks can have serious implications for enterprises. In their most basic application ARP spoofing attacks are used to steal sensitive information. Beyond this, ARP spoofing attacks are often used to facilitate other attacks such as:

2.2.1- Man in The Middle Attack - ARP Spoofing:

A Man-In-The-Middle (MITM) attack is achieved when an attacker poisons the ARP cache of two devices with the (48-bit) MAC address of their Ethernet NIC (Network Interface Card).
Once the ARP cache has been successfully poisoned, each of the victim devices send all their packets to the attacker when communicating to the other device.
This puts the attacker in the middle of the communications path between the two victim devices; hence the name Man-In-The-Middle (MITM) attack.

- By spoofing two hosts in the network at the same time, an attacker can silently sit in between the two hosts so that they think they are communicating with each other.

This attacker is then able to listen to all the traffic sent in both directions.
This attack can also be performed between any host in the LAN and an outside host, as the attacker can perform the attack between the host and the default gateway.

- The attacker can even sit in between a secure connection (e.g., SSL or SSH).

The application (e.g., the Web browser) will likely warn the user that the certificate provided is not valid, but many users tend to ignore this kind of warnings.
Furthermore, a bug in some versions of Internet Explorer enable attackers to hijack SSL sessions without the browser displaying a warning

- With a MITM attack, the attacker can gain access to sensitive information (e.g., passwords) or he/she can even modify the data being sent, compromising the data’s integrity.

The objective of this MITM attack is to take over a session. The intent is to intercept and view the information being passed between the two victim devices.
Example:

As we see in figure 1 the network consists of:
1. Wireless Router
2. Victim (which is communicating with the router to get webpages, read emails, etc......)
3. The attacker (who is going to use arp spoofing (MITM attack) to intercept the network between the victim and the wireless router gateway)
   - So the first thing this attacker needs to do is **to sniff traffic to discover the IP address on the network**, so once the attacker figures out the victim's IP address, then the attacker will **try to impersonate the router by assuming the router MAC address**.
   - There are two addresses involved here:
     1. IP address -- how we contact hosts on the network
     2. MAC address -- when we actually deliver packets, we deliver them through the MAC address "which is layer 2 address".

**Note:** MAC address is not a verified or authenticated protocol.
The attacker could just spoof or impersonate the MAC address at layer 2, so the victim thinks he is sending that packet to the gateway but in fact he is sending it to the attacker.
And then likewise we are going to also impersonate the Mac address of the victim computer so when the router send information back to the host it's actually sending it to our MAC address instead.
As we see in figure 1-2: Communication from the router to a host is a MAC address to mac address.

if we can convince these 2 devices that we are the mac address that they are tying to talk to then we will have a scenario like the one if **FIGURE 2-1 below**:

As we see:
1- The victim sends information to the **man in the middle**.
2- The **man in the middle** forwarded on the router.
3- Router sends it back to the **man in the middle** and then on to the victim. **hence name of the MITM attack** and we used ARP spoofing to impersonate the mac address of the victim and the gateway.
2.2.2- DOS Attack - ARP Spoofing:

Introduction:

An attacker can poison an ARP table of a host so that every packet that the host sends is sent to the attacker instead of its real destination. In this way, the attacker blocks the communication from the host being attacked.

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 (DoS).

This could also be a post-MITM attacks: target computers will continue to send frames to the attacker’s MAC address even after they remove themselves from the communication path.

In order to perform a clean MitM attack, the hacker will restore the ARP entries.

DOS attack could be:

1- SYN FLOOD:

• Normally when a client attempts to start a TCP connection to a server, the client and server exchange a series of messages which normally runs like this:

  1. The client requests a connection by sending a SYN (synchronize) message to the server.
  2. The server acknowledges this request by sending SYN-ACK back to the client.
  3. The client responds with an ACK, and the connection is established.

This is called the TCP three-way handshake, and is the foundation for every connection established using the TCP protocol.

• A SYN flood attack works by not responding to the server with the expected ACK code. The malicious client can either:

  A. Simply not send the expected ACK.

  B. Spoof the source IP address in the SYN, causing the server to send the SYN-ACK to a falsified IP address - which will not send an ACK because it "knows" that it never sent a SYN.
As we see in the above figure:
The attacker sends several packets but does not send the "ACK" back to the server. The connections are hence half-opened and consuming server resources. User, tries to connect but the server refuses to open a connection resulting in a denial of service.
2- Smurf Attack:

The Smurf Attack is a distributed denial-of-service attack in which large numbers of Internet Control Message Protocol (ICMP) packets with the intended victim’s spoofed source IP are broadcast to a computer network using an IP Broadcast address.

Most devices on a network will, by default, respond to this by sending a reply to the source IP address.

If the number of machines on the network that receive and respond to these packets is very large, the victim’s computer will be flooded with traffic.

This can slow down the victim’s computer to the point where it becomes impossible to work on.

Smurf attack (aka directed broadcast attacks) consumes network bandwidth of Victim.

![Diagram showing a Smurf attack](image)

**Fig 9.5** A Smurf attack results in a flood of the victim
3- DDOS Attack (Distributed denial of service attack):

→ Introduction:

A denial-of-service (DoS) or distributed denial-of-service (DDoS) attack is an attempt to make a machine or network resource unavailable to its intended users.

As clarification:
1- Distributed denial-of-service attacks are sent by one person.
2- Denial-of-service attacks are sent by one person or system.

As of 2014, the frequency of recognized DDoS attacks had reached an average rate of 28 per hour.

Perpetrators of DoS attacks typically target sites or services hosted on high-profile web servers such as banks, credit card payment gateways, and even root name servers. Denial-of-service threats are also common in business, and are sometimes responsible for website attacks.

DDOS Attack is:

- More powerful than Smurf attacks.
- No limitation on number of machines used to launch attack.
- No limitation on bandwidth that can be consumed.
- Used against Amazon, eBay, Etrade, and Zdnet in Feb 2000.
- Before performing a DDOS flood, attack must take over a large number of victim machines (zombies) and install zombie software.
- Attacker communicates with client machines which in turn send commands to zombies.
2.2.3- Session Hijacking:

Session hijacking is a serious threat to Networks and Web applications. Session hijacking attacks can use ARP spoofing to steal session IDs, granting attackers access to private systems and data. Once the user’s session ID has been accessed (through session prediction), the attacker can masquerade as that user and do anything the user is authorized to do on the network.

1-TCP Session hijacking:

TCP hijacks are meant to:
1- Intercept the already established TCP sessions between any two communicating parties.
2- Pretend to be one of them (ARP spoofing).
3- Redirect the TCP traffic to it by injecting spoofed IP packets so that your commands are processed on behalf of the authenticated host of the session.

TCP session hijacks can be implemented in two different ways: Middle Man Attack and the blind attack.

A. Man in the middle attack using Packet Sniffers:

- This technique involves using a packet sniffer to intercept the communication between client and the server.

Packet sniffer comes in two categories:

1- Passive sniffers: monitors and sniffs packet from a network having same collision domain, i.e. network with a hub, as all packets are broadcasted on each port of hub.

2- Active sniffers: works with Switched LAN network by ARP spoofing.

- Once the hijacker reads the TCP header, he can know the sequence number expected by the server, the acknowledgement number, the ports and the protocol numbers; so that hijacker can forge the packet and send it to the server before the client does so.

- Another way of doing so is to change the default gateway of the client’s machine so that it will route its packets via the hijacker’s machine. This can be done by ARP spoofing (i.e. by sending malicious ARP packets mapping its MAC address to the default gateways address so as to update the ARP cache on the client, to redirect the traffic to hijacker).
B. **The Blind Attack:**

- If you are not able to sniff the packets and guess the correct sequence number expected by server, you have to implement “Blind Session Hijacking”.

- You have to brute force 4 billion combinations of sequence number which will be an unreliable task.

**Here’s a simple example to Clarify TCP Session Hijacking:**

![TCP Session Hijacking Diagram]

**As We see in The Above figure:**
1- The attacker sends a single Z character to the server with sequence number x+2.
2- The server accepts it and sends the real client an ACK packet with acknowledgment number x+3 to confirm that it has received the Z character.
3- When the client receives the ACK packet, it will be confused, either because it did not send any data or because the next expected sequence is incorrect.
4- This confusion can cause a TCP ACK storm, which can disrupt a network.

**In any case, the attacker has now successfully hijacked this session.**

**Note:** The attacker needs a way to prevent the client from sending into the session new data that would shift sequence numbers forward.

**To do this, the attacker could :**
1- Send the data to inject and hope it is received before the real client can send new data, as shown in this example.
2- Perform a denial of service (DoS) attack on the client, or perhaps some tricks that use address resolution protocol (ARP) spoofing.
3- **UDP hijacking:**

- Since UDP does not use packet sequencing and synchronizing; it is easier than TCP to hijack UDP session.

- UDP attackers do not have to worry about the overhead of managing sequence numbers and other TCP mechanisms.

- The hijacker has simply to forge a server reply to a client UDP request before the server can respond. If sniffing is used than it will be easier to control the traffic generating from the side of the server and thus restricting server’s reply to the client in the first place.

- Since UDP is connectionless, injecting data into a session without being detected is extremely easy. The below Figure shows how an attacker could do this:
Solutions to ARP spoofing:

Many solutions were introduced to detect or prevent ARP spoofing. Some solutions aim at mitigating the problem and others aim at preventing it. Unfortunately no existing solution is ideal or solves the ARP spoofing attack perfectly.

A) Cryptographic approaches:

The first solution we’re going to talk about is a solution proposed by Bruschi. He proposed a Secure Address Resolution Protocol (SARP) which uses key cryptography to authenticate the hosts in LAN (Local Area Network). In SARP, each host uses an invite–accept protocol to periodically register its IP–MAC pairs in a secure server. IP-MAC pairs are then hashed. It adds an additional header at the end of the frame to carry the authentication information.

This solution may sound perfect and promising but the ugly truth is that it’s not. This solution requires medication of the ARP protocol as the sender needs to sign each ARP message with its private key, and the receiver needs to verify the signature with the sender’s public key. That’s quite of a complication and an overhead to the connection. Plus the hosts who have implemented the Secure ARP cannot accept the standard ARP, accepting only S-ARP frames.

Limmaneewichid and Lilakiatsakun proposed an ARP authentication scheme based on ARP authentication trailer, named P-ARP, which consists of a magic number, nonce and the authentication data produced by the HMAC hash function. This happens in order to hide the target IP address in an APR request message. Unfortunately additional operation such as hash function must be performed to create nonce and HMAC values. In addition, this approach is ineffective against ARP DoS attacks. There goes another ineffective approach.
B) **Static ARP configuration:**

The easiest way to protect the network to an ARP poisoning attack is the manual setting of a static ARP cache, adding static entry in the Operating System. For example in Windows XP is used the command:

```bash
arp –s IP MAC
```

where the IP and the MAC address are those of the device that the OS wants to reach. The static ARP can be created in all operating system with a similar command. This operation is very simple but it must be repeated in all computers in the network. It is compatible with all OS and maintain high performance but it is not manageable because all computers require this configuration and if a computer changes its network card interface (NIC) or its IP address, it is necessary reconfigure all computers in the network. Furthermore, in this kind of network, it is not possible to use some service as dynamic IP address assignment to hosts (DHCP)

C) **Network Intrusion Detection and Prevention System (NIDPS)**

Another mechanism of defense against ARP attacks is based on the inspection of ARP packets. The NIDPS is divided in two parts: an Information Agent (IA) and a NIDPS firewall. Each computer in the network has the IA installed on it. Every time that the computer starts up, it sends an ARP Request or receives an ARP reply, the Information Agents sends a notification to NIDPS. It also takes notes about how many ARP request and reply are received in a determinate time interval. All this information is send to the NIDPS firewall that analyzes the data, trying to detect an attack. An experiment shows that it is very powerful against external attacks and where the source of the attack is itself but it is not able to detect attacks in an internal network.

It maintains the standard ARP protocol and it has a very light weight. Nevertheless this system has some disadvantages like the cost of this type of hardware or a dedicate computer to use as NIDPS firewall and the ineffectiveness of this method for internal attacks. Furthermore this solution is not advantageous for a big company because would be very expensive.
D) **Kernel-based patch:**

A different manner to protect the network from an ARP poisoning attack is to install a kernel patch. There are a lot of this kind of program as XARP, ARPWatch, AntiARP, Anticap or Antidote.

For example, Anticap checks constantly the ARP cache and if it receives an ARP reply with a different MAC address for a specific IP before the expiration time, it does not update the ARP cache. When this happen, Anticap sends an alert to the user that someone is trying to poison the ARP cache.

More advanced than Anticap, is Antidote. When the computer receives a new ARP replies that wants to change IP and MAC address pairs, it tries to discover if the previous MAC address is still alive with an ARP request. If it replies to this request, it stores this in its ARP cache and rejects the other MAC address.

This method is compatible with the standard ARP, cheap because often this kind of program are free and open source and easily manageable but they require that all the computers in the network have this patch installed. On the other hand, these patches can be used only with some specific kernel and, sometimes, these are not compatible with the ARP mechanisms in other un-patched kernels.

E) **Server-based approaches:**

Gouda and Huang proposed an architecture in which a secure server is connected to the Ethernet and communications with the server take place using invite-accept and request-reply protocols. All ARP requests and replies occur between a host and the server, and replies are authenticated using shared pair keys.

Kwon proposed a similar approach. This approach uses an agent which retrieves genuine IP–MAC pairs from a host and forwards them to the manager to construct reliable IP–MAC mapping. The manager node monitors if IP addresses of licensed hosts are changed, and unauthorized hosts are disconnected as they are assumed to have suffered spoofing attacks. Unfortunately all these were attempts but no actual applied solution because they all failed to address how the server collects the correct IP–MAC mappings so that it may generate correct reply to the incoming ARP requests.
F) **ASA (Anti-ARP Spoofing Agent) software:**

Address resolution protocol (ARP) is widely used to maintain mapping between data link (e.g. MAC) and network (e.g. IP) layer addresses. Although most hosts rely on automated and dynamic management of ARP cache entries, current implementation is well-known to be vulnerable to spoofing or denial of service (DoS) attacks. There are many tools that exploit vulnerabilities of ARP protocols, and past proposals to address the weaknesses of the ‘original’ ARP design have been unsatisfactory. Suggestions that ARP protocol definition be modified would cause serious and unacceptable compatibility problems. Other proposals require customized hardware be installed to monitor malicious ARP traffic, and many organizations cannot afford such cost.

This study demonstrates that one can effectively eliminate most threats caused by the ARP vulnerabilities by installing anti-ARP spoofing agent (ASA) which intercepts unauthenticated exchange of ARP packets and blocks potentially insecure communications. The proposed approach requires neither medication of kernel ARP software nor installation of traffic monitors. Agent uses user datagram protocol (UDP) packets to enable networking among hosts in a transparent and secure manner. The authors implemented agent software on Windows XP and conducted an experiment. The results showed that ARP hacking tools could not penetrate hosts protected by ASA.

As we’ve seen in the previous solutions most solutions were ineffective and rather naïve. The best solution was the ASA software.

A comparison to see the advantages and points of power and weakness in defense against ARP spoofing is obligatory in this stage in order to understand all point that are whether with or against these software programs.

The following table (table-1) is a comparison between solutions introduced in our paper:

<table>
<thead>
<tr>
<th>Methods</th>
<th>Compatibility</th>
<th>Flexibility</th>
<th>Effectiveness</th>
<th>Performance</th>
<th>cheap</th>
<th>Manage-ability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cryptog.</td>
<td>In part</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Stat. ARP</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>NIDPS</td>
<td>Yes</td>
<td>In part</td>
<td>In part</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Kernel</td>
<td>Yes</td>
<td>In part</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Server-based</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>ASA</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

(Table-1)
3 Background/Related Work: (DNS Spoofing)

Introduction:

DNS or Domain Name Server translates an often humanly-meaningful, text-based identifier to a system-internal, often numeric identification or addressing component.

DNS spoofing (or DNS cache poisoning) is a computer hacking attack, whereby data is introduced into a Domain Name System (DNS) resolver's cache, causing the name server to return an incorrect IP address, diverting traffic to the attacker's computer (or any other computer). For simplification, DNS server accepts and uses incorrect information from a host that has no authority giving that information.

Spoofing attacks can cause serious security problems for DNS servers vulnerable to such attacks, for example causing users to be directed to wrong Internet sites or e-mail being routed to non-authorized mail servers.
Background:

- "The well known port 53" is the port which runs the Domain name Server (DNS) service, however, it is considered as the hackers first option to attack, this refers to the importance of the DNS service as it is the heart of the internet infrastructure.

- DNS translates domain names, which can be easily memorized by humans, to the numerical IP addresses needed for the purpose of computer services and devices worldwide.

  "Mapping between IP addresses and URLs is maintained as a service, DNS servers does this job of transforming between these two."

- **DNS runs on UDP, but since UDP is connectionless it makes it easy for spoofing.**

- To understand The **DNS Spoofing** we have to understand first **How the DNS server works?**

  1- Every host connecting to the internet or is a member of a big network that uses internal DNS, should be configured to use a particular DNS server or a group of redundant DNS servers which will do the resolution process for him.

  2- If the user for example is trying to access www.hotmail.com web site, queries will be sent to the DNS server to resolve the associated IP address for the required website.

  Host.company.com ----> www.hotmail.com ----> dns.company.com

  10.0.0.4 10.0.0.60

  3- The company DNS will send the question query to the top level root name servers or to the ns.internic.net which might be having the answer cached.

  dns.company.com ----> www.hotmail.com ----> a.root-servers.net

  1.0.0.60 192.5.6.30

  4- if it doesn’t have the answer, it will send back a list of the authoritative domain name servers for the com domain.

  5- dns.company.com will pick one of these authoritative domain names and send the question query again.
6- Now a.gtld-servers.net will send back the list of the authoritative domain name servers serving the hotmail.com domain.

7- dns.company.com will choose one of these name servers and resend the query again
dns.company.com ----> www.hotmail.com ---->ns1.hotmail.com
10.0.0.60 216.200.206.140
This query will send the list of ip addresses serving the hotmail webpage.

The DNS spoof attack takes place if the attacker could spoof the discussion between
dns.company.com and the authorized domain name servers.
Then he can send his responses with fault information back to the host before the real
answer is being sent by dns.company.com
This will cause a local cache poisoning for the host’s browser cache and each time he wants
to access the www.hotmail.com webpage, he will be redirected to the faulty page until the
cache file’s life time expires or the user deletes his browser cache.
However, if the attacker could send the faulty response to the DNS server itself,
This answer will be cached in the DNS cache records and all hosts connecting to
dns.company.com and trying to resolve www.hotmail.com will be redirected to the IP
address supplied by the attacker.
**Conclusion:**

DNS spoofing is a term used when a DNS server accepts and uses incorrect information from a host that has no authority giving that information.

DNS spoofing is in fact malicious cache poisoning where forged data is placed in the cache of the name servers.

Spoofing attacks can cause serious security problems for DNS servers vulnerable to such attacks, for example causing users to be directed to wrong Internet sites or e-mail being routed to non-authorized mail servers.

**Comparison between ARP Spoofing & DNS Spoofing:**

<table>
<thead>
<tr>
<th>ARP Spoofing</th>
<th>DNS Spoofing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Is when an attacker sends out fake replies to an ARP Request. This is done usually to impersonate a router so that an attacker can intercept traffic.</td>
<td>Is when an attacker replies to DNS Requests (sent to resolve the IP address of a hostname) with false IP information. This is typically used to redirect users to false websites.</td>
</tr>
</tbody>
</table>

**Note:** Basically both methods try the same (redirecting or forwarding traffic to malicious hosts), but they work at a different level.
4 Evaluation

Recent researches show that of all the attempts to solve the ARP security issues, unfortunately no method is ideal and a 100% protection just doesn’t exist. There are always defaults, misses or assumptions that couldn’t be put to action.

5 Conclusions

ARP is a very trusting protocol whose developers took no security regards into consideration but still made a very vital protocol that most connections can’t do without. Finding solutions to the security issues was a must but it seems like no perfectly successful solution was introduced even though they all seemed perfect and promising when learning about them. This shows how important and vital and obviously not easy the role of ethical hackers is. A single default in a protocol and many attacks come up out of the blue. Therefore the role of the ethical hacker should come before putting the protocol actually in action and witness spying and information leakage.

6 Future Work

This research simulated our interest in further learning in the ethical hacking field and made us understand how critical and serious this field is. It helped us know about the ethical hacking field and what an ethical hacker does. Most attempts to solve the ARP spoofing problem failed due to compatibility issue or inconvenience or other reasons which put the possible errors that can happen in the spot light in order to avoid similar errors in case of a new attempt. This research is a must for any ethical hacker that need to find a solution to ARP spoofing.
7 References


    -how-to-detect-and-protect-the-network-from-them/
