**Calin Irina, E2**

# Information Security - Homework 3 - ARP cache poisoning

## Prerequisites:

- Used virtualization software: VMware Workstation 16 PRO
- Ubuntu version: 20.04.1
- Remark: since the Ubuntu version is the one above, eth0 will be replaced with **ens33** and eth1 with **ens34**, respectively
- Other used applications:
  - Ettercap: for implementing ARP poisoning
  - Wireshark: for monitoring the packet exchange over the network

| Node | IP address | MAC address |
|---|---|---|
| Router ens34 (eth1) | 192.168.60.11 | 00:0C:29:B0:0A:EB |
| C1 (attacker) | 192.168.60.12 | 00:0C:29:44:26:E3 |
| C2 (victim) | 192.168.60.13 | 00:0C:29:ED:B1:34 |

**Table 1**

## Initial remarks

Hosts maintain an ARP cache - a mapping table between IP addresses and MAC addresses, and use it to connect to destinations on the network. If the host doesn't know the MAC address for a certain IP address, it sends out an ARP request packet, asking other machines on the network for the matching MAC address.[1]

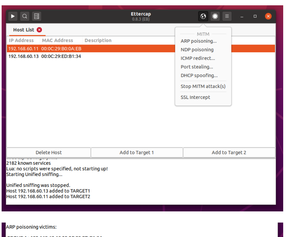
The ARP protocol was not designed for security, so it does not verify that a response to an ARP request really comes from an authorized party. It also lets hosts accept ARP responses even if they never sent out a request. This is a weak point in the ARP protocol, which opens the door to ARP spoofing attacks.[2]

An ARP spoofing, also known as ARP poisoning, is a **Man in the Middle** (MitM) attack that allows attackers to intercept communication between network devices. The attack works as described below.

## Observations during implementation:

- The attacker must have access to the network. They scan the network to determine the IP addresses of at least two devices—let's say these are a workstation and a router. [3]

After running Ettercap and selecting the targets, we can start sniffing by sending out forged ARP responses.

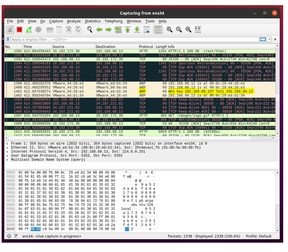


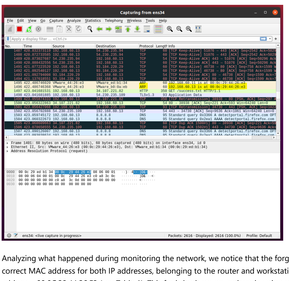

```
ARP poisoning victims:

GROUP 1 : 192.168.60.13 00:0C:29:ED:B1:34

GROUP 2 : 192.168.60.11 00:0C:29:B0:0A:EB
Starting Unified sniffing...
```

We choose the observer to be the router and from its point of view, the situation looks like this:




Analyzing what happened during monitoring the network, we notice that the forged responses advertise that the correct MAC address for both IP addresses, belonging to the router and workstation, is exactly the attacker's MAC address - 00:0C:29:44:26:E3 (see Table 1). This fools both router and workstation to connect to the attacker's machine, instead of to each other.

This is the point where the two devices update their ARP cache entries and from that point onwards, communicate with the attacker instead of directly with each other.

The attacker is now secretly in the middle of all communications. The attack succeeds almost instantly on a LAN.

Another conclusion we can take is that the victim can verify sniffing by running:

```
$ arp -a
```

and checking if are 2 identical MAC addresses for different IPs, which in this case will produce the following situation:


Once the attacker succeeds in an ARP spoofing attack, they can:

- Continue routing the communications as-is = > the attacker can sniff the packets and steal data, except if it is transferred over an encrypted channel like HTTPS.
- Perform session hijacking—if the attacker obtains a session ID, they can gain access to accounts the user is currently logged into.
- Alter communication—for example pushing a malicious file or website to the workstation.
- Distributed Denial of Service (DDoS)—the attackers can provide the MAC address of a server they wish to attack with DDoS, instead of their own machine. If they do this for a large number of IPs, the target server will be bombarded with traffic.[4]

## What can be done next:

We will demonstrate how the attacker can steal the victim's data simply when the victim logs into a website from its machine. This is purely demonstrative, which is why we are using a website made especially for testing security issues (using HTTP only).


Once the victim presses the login button, the attacker will receive the following notification:

```
HTTP: 18.192.172.30:80 -> USER: irina PASS: irina INFO: http://testphp.vulnweb.com/login.php
CONTENT: uname=irina&pass=irina
```

The attacker has just intercepted the victim's credentials.

## Proposed solutions against ARP poisoning

There are many proposed solutions for preventing ARP cache poisoning, from which we can mention some:

- One thing that could be noticed is that, by continuously monitoring the traffic, we could very easily identify that poisoning was taking place, since the MAC addresses of both the router and the workstation suffered changes.
  This means that one way to protect the network is by using software which monitors the ARP cache, sending notifications when mappings (as described above) suffer suspect changes.

- Another form of certification is the use of static, read-only entries for critical services in the ARP cache of a host. IP address-to-MAC address mappings in the local ARP cache may be statically entered. Hosts don't need to transmit ARP requests where such entries exist. (Remark: While static entries provide some security against spoofing, they result in maintenance efforts as address mappings for all systems in the network must be generated and distributed. This does not scale on a large network since the mapping has to be set for each pair of machines resulting in n^(2n) ARP entries that have to be configured when n machines are present; On each machine there must be an ARP entry for every other machine on the network; n-1 ARP entries on each of the n machines.) [5]

- One of the best solutions, however, would be traffic encryption. This could be done by using a VPN, which allows devices to connect to the Internet through an encrypted tunnel. This makes all communication secure, and worthless for an ARP spoofing attacker.

- Packet filtering solutions can identify poisoned ARP packets by seeing that they contain conflicting source information, and stop them before they reach devices on your network. [6]

Note: One thing to remember, however, is that ARP cache poisoning can only take place when the attacker is itself part of the network.