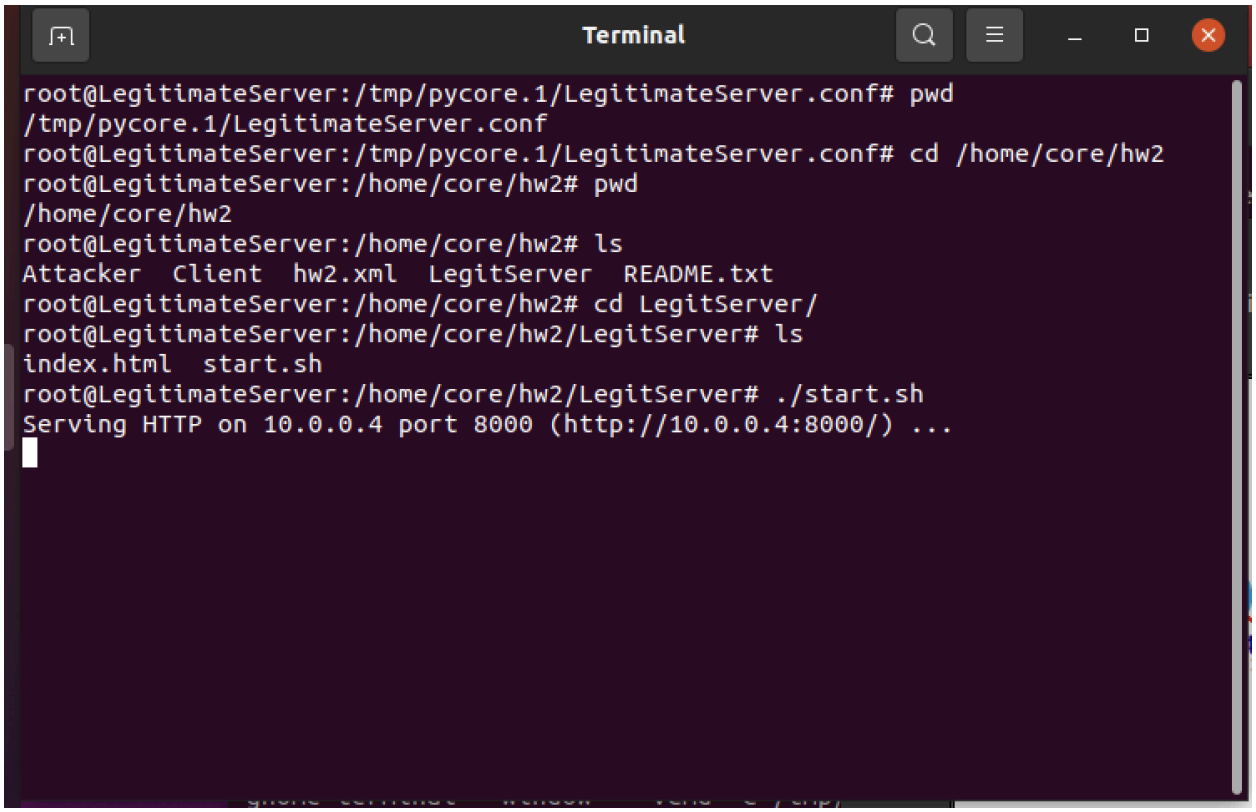
HW2 - Show the Effect of ARP Poisoning on a Network

1. Run an HTTP Server on the Server node

After powering up the core GUI and setting the environment using hw2.xml, I opened a terminal in the LegitServer node and ran the “start.sh” script using the command `./start.sh`.



2. Run a Script on the Client node

Now, I opened a terminal in the client node and ran the script “run_curl.sh” to request the front page of the server using the command `./run_curl.sh`.

```
Attacker  Client  hw2.xml  LegitServer  README.txt
root@Client:/home/core/hw2# cd Client
root@Client:/home/core/hw2/Client# ls
run_curl.sh
root@Client:/home/core/hw2/Client# ./run_curl.sh
<html>
  <b> Legit Server </b>
</html>

--Mon 09 Sep 2024 11:58:27 PM EDT--
<html>
  <b> Legit Server </b>
</html>

--Mon 09 Sep 2024 11:58:29 PM EDT--
<html>
  <b> Legit Server </b>
</html>

--Mon 09 Sep 2024 11:58:31 PM EDT--
<html>
  <b> Legit Server </b>
</html>
```

3. ARP Mapping

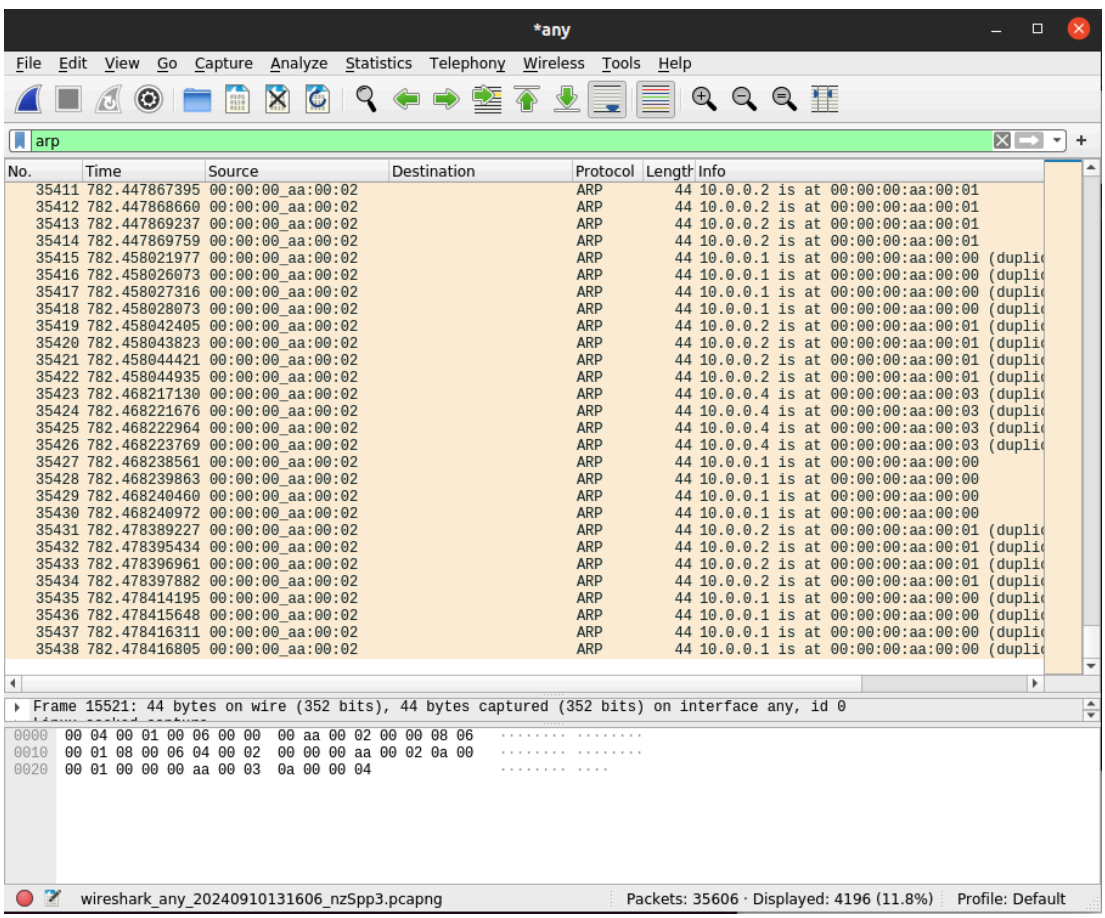
After running the script from both the client and server node, I used the command `arp -na` to see the content of the ARP mapping of the IP address to the MAC address. The output shows the ARP entry at the Client that maps the Server's IP address to its MAC address.

```
root@Client:/tmp/pycore.1/Client.conf# cd /home/core/hw2
root@Client:/home/core/hw2# arp -na
? (10.0.0.4) at 00:00:00:aa:00:03 [ether] on eth0
root@Client:/home/core/hw2#
```

4. Run the ARP Poison Attack

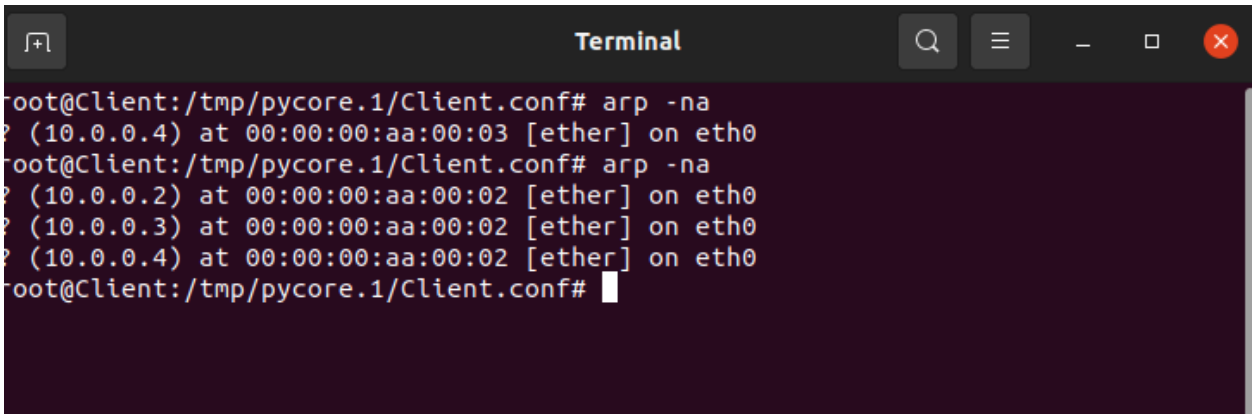
I opened a terminal in the Attacker node and ran the script run_arp_poison.sh using the command `./run_arp_poison.sh` which sent a lot of forged ARP packets mapping the IP address of the Server to the MAC address of the Attacker. In Wireshark, I can see the forged ARP packets sent by the Attacker.

```
root@Attacker:/home/core/hw2/Attacker# ls
config_firewall.sh  index.html  run_arp_poison.sh  start_server.sh
root@Attacker:/home/core/hw2/Attacker# ./run_arp_poison.sh

ettercap 0.8.3 copyright 2001-2019 Ettercap Development Team

Listening on:
  eth0 -> 00:00:00:AA:00:02
          10.0.0.3/255.255.255.0
          fe80::200:ff:feaa:2/64
          2001::3/64

SSL dissection needs a valid 'redir_command_on' script in the etter.conf file
Privileges dropped to EUID 65534 EGID 65534...

  34 plugins
  42 protocol dissectors
  57 ports monitored
24609 mac vendor fingerprint
1766 tcp OS fingerprint
```



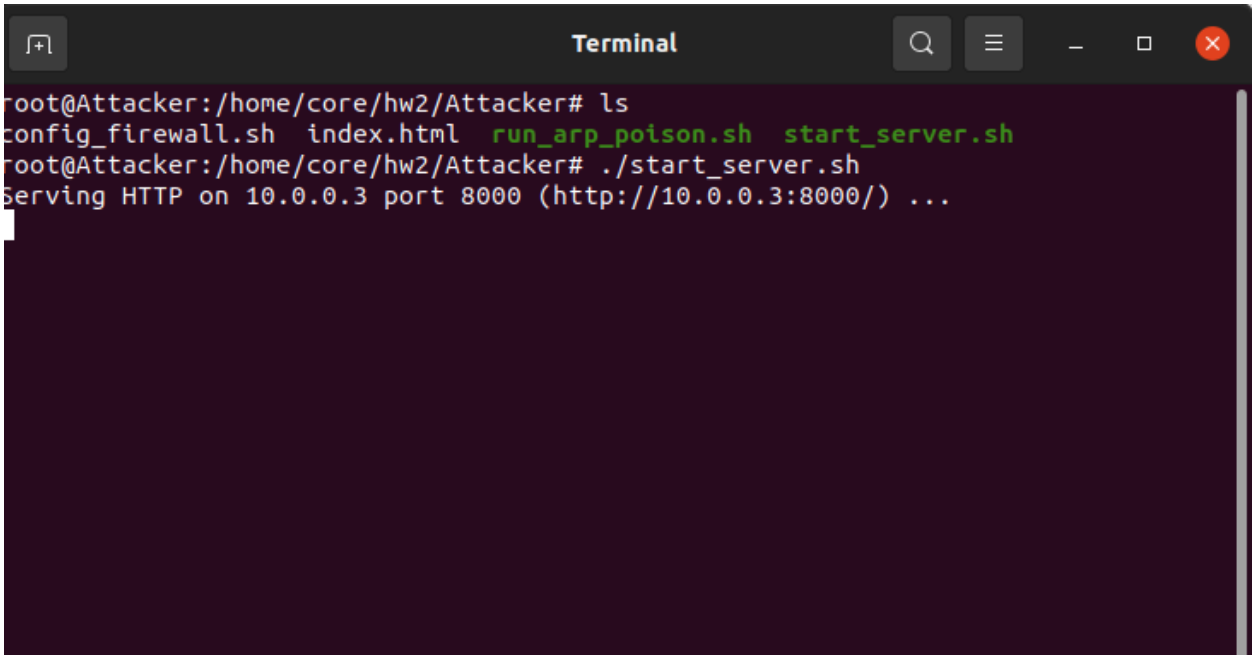
5. Notice the Mapping Changed

The ARP poisoning attack I ran was successful. The Client's ARP table now maps the server's IP address to the attacker's MAC address, meaning that all traffic from the client to the server is now routed through the attacker. I used the command `arp -na` to view the ARP table of the client.



6. Run an HTTP Server in the Attacker Node

Now, I ran the HTTP server using the script start_server.sh in the Attacker node using the command `./start_server.sh`.



7. Run config_firewall.sh in the Attacker's node

I ran the script config_firewall.sh using the command `./config_firewall.sh`. The script sets up the firewall to reroute the client's HTTP traffic to the attacker's HTTP server. After the ARP poisoning attack, it ensures that the attacker intercepts all client HTTP requests. Now, the attacker can serve different content from their own HTTP server running on port 8000.

```
Terminal
root@Attacker:/home/core/hw2/Attacker# ls
config_firewall.sh  index.html  run_arp_poison.sh  start_server.sh
root@Attacker:/home/core/hw2/Attacker# ./config_firewall.sh
root@Attacker:/home/core/hw2/Attacker# ./run_arp_poison.sh

ettercap 0.8.3 copyright 2001-2019 Ettercap Development Team

Listening on:
  eth0 -> 00:00:00:AA:00:02
          10.0.0.3/255.255.255.0
          fe80::200:ff:feaa:2/64
          2001::3/64

SSL dissection needs a valid 'redir_command_on' script in the etter.conf file
Privileges dropped to EUID 65534 EGID 65534...
```

```
Terminal
--Tue 10 Sep 2024 05:46:43 PM EDT--
<html>
  <b> Legit Server </b>
</html>

--Tue 10 Sep 2024 05:46:45 PM EDT--
<html>
  <b> Legit Server </b>
</html>

--Tue 10 Sep 2024 05:46:47 PM EDT--
<html>
  <b> Legit Server </b>
</html>

--Tue 10 Sep 2024 05:46:49 PM EDT--
<html>
  <b> Attacker Server </b>
</html>

--Tue 10 Sep 2024 05:46:51 PM EDT--
<html>
  <b> Attacker Server </b>
</html>
```

8. OS Fingerprinting Scan using Nmap

a) Install Nmap

I installed Nmap using the command `sudo apt-get -y install nmap` and checked its version using the command `nmap -version`.

```
Terminal
root@Attacker:/tmp/pycore.1/Attacker.conf# nmap --version
Nmap version 7.80 ( https://nmap.org )
Platform: x86_64-pc-linux-gnu
Compiled with: liblua-5.3.3 openssl-1.1.1f nmap-libssh2-1.8.2 libz-1.2.11 libpcr
e-8.39 libpcap-1.9.1 nmap-libdnet-1.12 ipv6
Compiled without:
Available nsock engines: epoll poll select
root@Attacker:/tmp/pycore.1/Attacker.conf#
```

b) OS Fingerprinting Scan

I ran the OS Fingerprint scan from an attacker on the server using the command `nmap -sV -O -v 10.0.0.4`.

```
Terminal
root@Attacker:/home/core/hw2# nmap -sV -O -v 10.0.0.4
Starting Nmap 7.80 ( https://nmap.org ) at 2024-09-10 18:41 EDT
NSE: Loaded 45 scripts for scanning.
Initiating ARP Ping Scan at 18:41
Scanning 10.0.0.4 [1 port]
Completed ARP Ping Scan at 18:41, 0.05s elapsed (1 total hosts)
mass_dns: warning: Unable to open /etc/resolv.conf. Try using --system-dns or sp
ecify valid servers with --dns-servers
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled.
 Try using --system-dns or specify valid servers with --dns-servers
Initiating SYN Stealth Scan at 18:41
Scanning 10.0.0.4 [1000 ports]
Discovered open port 8000/tcp on 10.0.0.4
Completed SYN Stealth Scan at 18:41, 0.06s elapsed (1000 total ports)
Initiating Service scan at 18:41
Scanning 1 service on 10.0.0.4
Completed Service scan at 18:41, 6.02s elapsed (1 service on 1 host)
Initiating OS detection (try #1) against 10.0.0.4
Retrying OS detection (try #2) against 10.0.0.4
Retrying OS detection (try #3) against 10.0.0.4
Retrying OS detection (try #4) against 10.0.0.4
Retrying OS detection (try #5) against 10.0.0.4
NSE: Script scanning 10.0.0.4.
Initiating NSE at 18:41
```

```
nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=7.80%E=4%D=9/10%OT=8000%CT=1%CU=38859%PV=Y%DS=1%DC=D%G=Y%M=000000
OS:%TM=66E0CB2B%P=x86_64-pc-linux-gnu)SEQ(SP=105%GCD=1%ISR=10A%TI=Z%CI=Z%II
OS:=I%TS=A)OPS(O1=M5B4ST11NW7%O2=M5B4ST11NW7%O3=M5B4NNT11NW7%O4=M5B4ST11NW7
OS:%O5=M5B4ST11NW7%O6=M5B4ST11)WIN(W1=FE88%W2=FE88%W3=FE88%W4=FE88%W5=FE88%
OS:W6=FE88)ECN(R=Y%DF=Y%T=40%W=FAF0%O=M5B4NNSNW7%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S
OS:=O%A=S+%F=AS%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%R
OS:D=0%Q=)T5(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=
OS:0%S=A%A=Z%F=R%O=%RD=0%Q=)T7(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U
OS:1(R=Y%DF=N%T=40%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DF
OS:I=N%T=40%CD=S)

Uptime guess: 13.350 days (since Wed Aug 28 10:17:49 2024)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=261 (Good luck!)
IP ID Sequence Generation: All zeros

Read data files from: /usr/bin/../share/nmap
OS and Service detection performed. Please report any incorrect results at https
://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 18.80 seconds
           Raw packets sent: 1111 (52.918KB) | Rcvd: 1071 (46.282KB)
root@Attacker:/home/core/hw2#
```

c) Was the scan able to detect the OS of the server?

No, the Nmap cannot be able to determine the OS of the server. You can see in the output:

```
Completed Service scan at 18:41, 6.02s elapsed (1 service on 1 host)
Initiating OS detection (try #1) against 10.0.0.4
Retrying OS detection (try #2) against 10.0.0.4
Retrying OS detection (try #3) against 10.0.0.4
Retrying OS detection (try #4) against 10.0.0.4
Retrying OS detection (try #5) against 10.0.0.4
NSE: Script scanning 10.0.0.4.
Initiating NSE at 18:41
Completed NSE at 18:41, 0.03s elapsed
Initiating NSE at 18:41
Completed NSE at 18:41, 0.00s elapsed
Nmap scan report for 10.0.0.4
Host is up (0.000065s latency).
Not shown: 999 closed ports
PORT     STATE SERVICE VERSION
8000/tcp open  http    SimpleHTTPServer 0.6 (Python 3.8.10)
MAC Address: 00:00:00:AA:00:03 (Xerox)
No exact OS matches for host (If you know what OS is running on it, see https://
nmap.org/submit/ ).
```

d) Uptime of the server and How it's useful for an attacker.

Nmap was able to guess the uptime of the server. You can see it in the output:

```
Uptime guess: 13.350 days (since Wed Aug 28 10:17:49 2024)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=261 (Good luck!)
IP ID Sequence Generation: All zeros
```

Uptime: Refers to how long the server has been running since the last reboot. In this case, it has been running for approx. 13.350 days since the last restart.

How it's useful for an attacker: Long uptime can tell that the server hasn't been rebooted for a while, which means security patches/updates haven't been made. Knowing the uptime might give an attacker leverage into the server's maintenance schedules or any vulnerabilities related to older versions of the server.

e) Any Open Ports and Services?

Nmap identified one open port i.e. `8000/tcp`, and its corresponding service running on this port is `SimpleHTTPServer` version 0.6, a basic web server used for development and testing purposes.

```
Nmap scan report for 10.0.0.4
Host is up (0.000065s latency).
Not shown: 999 closed ports
PORT     STATE SERVICE VERSION
8000/tcp open  http    SimpleHTTPServer 0.6 (Python 3.8.10)
MAC Address: 00:00:00:AA:00:03 (Xerox)
```