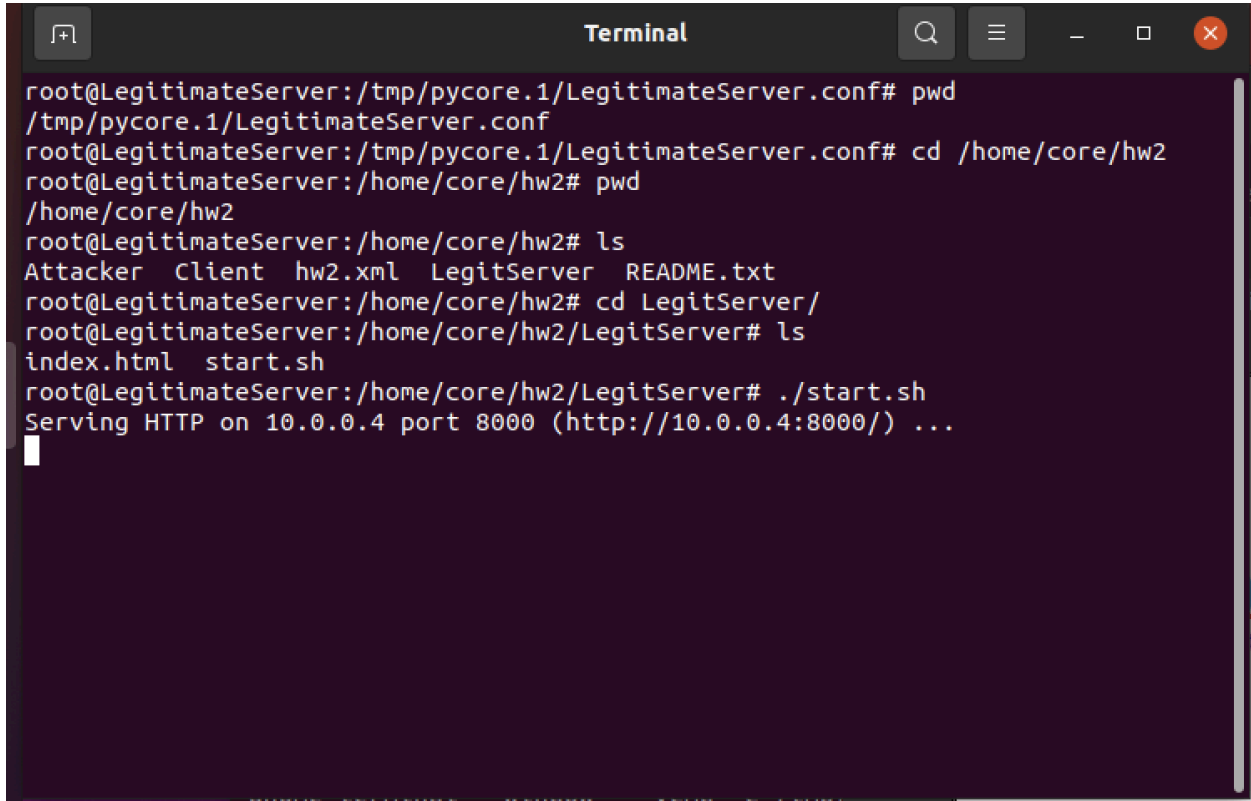


# HW2 - Show the Effect of ARP Poisoning on a Network

## 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`.



```
root@LegitimateServer:/tmp/pycore.1/LegitimateServer.conf# pwd
/tmp/pycore.1/LegitimateServer.conf
root@LegitimateServer:/tmp/pycore.1/LegitimateServer.conf# cd /home/core/hw2
root@LegitimateServer:/home/core/hw2# pwd
/home/core/hw2
root@LegitimateServer:/home/core/hw2# ls
Attacker  Client  hw2.xml  LegitServer  README.txt
root@LegitimateServer:/home/core/hw2# cd LegitServer/
root@LegitimateServer:/home/core/hw2/LegitServer# ls
index.html  start.sh
root@LegitimateServer:/home/core/hw2/LegitServer# ./start.sh
Serving HTTP on 10.0.0.4 port 8000 (http://10.0.0.4:8000/) ...
```

## 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
```

The image shows a Wireshark network traffic capture window. The top menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. The toolbar contains various icons for packet capture and analysis. The packet list pane shows a series of ARP packets (No. 35411 to 35438) captured on interface 'any'. Each packet is from source IP 10.0.0.2 to destination IP 10.0.0.1, with protocol ARP. The packet details pane shows the selected packet (No. 15521) with 44 bytes on wire (352 bits) and 44 bytes captured (352 bits) on interface any, id 0. The packet bytes pane shows the raw data in hexadecimal and ASCII format.

No.	Time	Source	Destination	Protocol	Length	Info
35411	782.447867395	00:00:00:aa:00:02	10.0.0.2	ARP	44	10.0.0.2 is at 00:00:00:aa:00:01
35412	782.447868660	00:00:00:aa:00:02	10.0.0.2	ARP	44	10.0.0.2 is at 00:00:00:aa:00:01
35413	782.447869237	00:00:00:aa:00:02	10.0.0.2	ARP	44	10.0.0.2 is at 00:00:00:aa:00:01
35414	782.447869759	00:00:00:aa:00:02	10.0.0.2	ARP	44	10.0.0.2 is at 00:00:00:aa:00:01
35415	782.458021977	00:00:00:aa:00:02	10.0.0.1	ARP	44	10.0.0.1 is at 00:00:00:aa:00:00 (duplicate)
35416	782.458026073	00:00:00:aa:00:02	10.0.0.1	ARP	44	10.0.0.1 is at 00:00:00:aa:00:00 (duplicate)
35417	782.458027316	00:00:00:aa:00:02	10.0.0.1	ARP	44	10.0.0.1 is at 00:00:00:aa:00:00 (duplicate)
35418	782.458028073	00:00:00:aa:00:02	10.0.0.1	ARP	44	10.0.0.1 is at 00:00:00:aa:00:00 (duplicate)
35419	782.458042405	00:00:00:aa:00:02	10.0.0.2	ARP	44	10.0.0.2 is at 00:00:00:aa:00:01 (duplicate)
35420	782.458043823	00:00:00:aa:00:02	10.0.0.2	ARP	44	10.0.0.2 is at 00:00:00:aa:00:01 (duplicate)
35421	782.458044421	00:00:00:aa:00:02	10.0.0.2	ARP	44	10.0.0.2 is at 00:00:00:aa:00:01 (duplicate)
35422	782.458044935	00:00:00:aa:00:02	10.0.0.2	ARP	44	10.0.0.2 is at 00:00:00:aa:00:01 (duplicate)
35423	782.468217130	00:00:00:aa:00:02	10.0.0.4	ARP	44	10.0.0.4 is at 00:00:00:aa:00:03 (duplicate)
35424	782.468221676	00:00:00:aa:00:02	10.0.0.4	ARP	44	10.0.0.4 is at 00:00:00:aa:00:03 (duplicate)
35425	782.468222964	00:00:00:aa:00:02	10.0.0.4	ARP	44	10.0.0.4 is at 00:00:00:aa:00:03 (duplicate)
35426	782.468223769	00:00:00:aa:00:02	10.0.0.4	ARP	44	10.0.0.4 is at 00:00:00:aa:00:03 (duplicate)
35427	782.468238561	00:00:00:aa:00:02	10.0.0.1	ARP	44	10.0.0.1 is at 00:00:00:aa:00:00 (duplicate)
35428	782.468239863	00:00:00:aa:00:02	10.0.0.1	ARP	44	10.0.0.1 is at 00:00:00:aa:00:00 (duplicate)
35429	782.468240460	00:00:00:aa:00:02	10.0.0.1	ARP	44	10.0.0.1 is at 00:00:00:aa:00:00 (duplicate)
35430	782.468240972	00:00:00:aa:00:02	10.0.0.1	ARP	44	10.0.0.1 is at 00:00:00:aa:00:00 (duplicate)
35431	782.478389227	00:00:00:aa:00:02	10.0.0.2	ARP	44	10.0.0.2 is at 00:00:00:aa:00:01 (duplicate)
35432	782.478395434	00:00:00:aa:00:02	10.0.0.2	ARP	44	10.0.0.2 is at 00:00:00:aa:00:01 (duplicate)
35433	782.478396961	00:00:00:aa:00:02	10.0.0.2	ARP	44	10.0.0.2 is at 00:00:00:aa:00:01 (duplicate)
35434	782.478397882	00:00:00:aa:00:02	10.0.0.2	ARP	44	10.0.0.2 is at 00:00:00:aa:00:01 (duplicate)
35435	782.478414195	00:00:00:aa:00:02	10.0.0.1	ARP	44	10.0.0.1 is at 00:00:00:aa:00:00 (duplicate)
35436	782.478415648	00:00:00:aa:00:02	10.0.0.1	ARP	44	10.0.0.1 is at 00:00:00:aa:00:00 (duplicate)
35437	782.478416311	00:00:00:aa:00:02	10.0.0.1	ARP	44	10.0.0.1 is at 00:00:00:aa:00:00 (duplicate)
35438	782.478416805	00:00:00:aa:00:02	10.0.0.1	ARP	44	10.0.0.1 is at 00:00:00:aa:00:00 (duplicate)

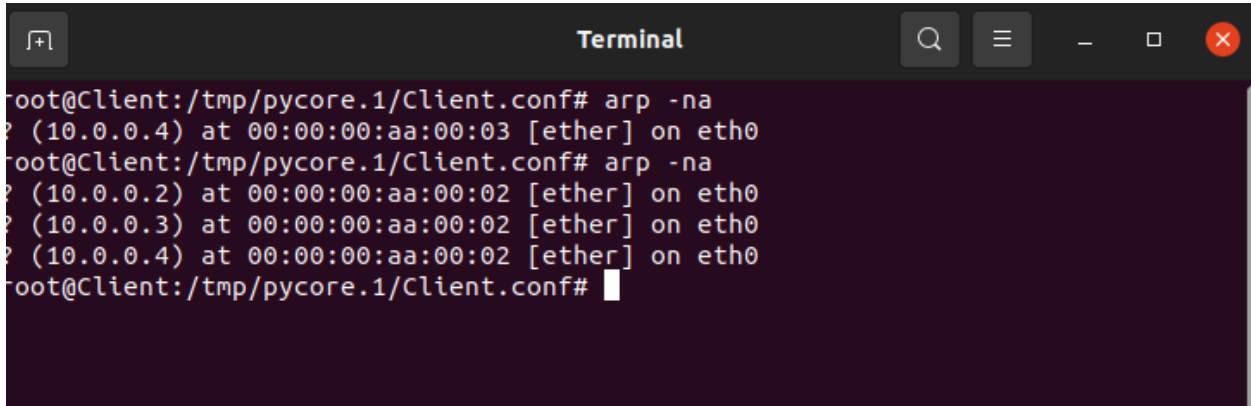
Frame 15521: 44 bytes on wire (352 bits), 44 bytes captured (352 bits) on interface any, id 0

0000 00 04 00 01 00 06 00 00 00 aa 00 02 00 00 08 06 .....  
0010 00 01 08 00 06 04 00 02 00 00 00 aa 00 02 0a 00 .....  
0020 00 01 00 00 00 aa 00 03 0a 00 00 04 .....  
.....

wireshark\_any\_20240910131606\_nzSpp3.pcapng Packets: 35606 · Displayed: 4196 (11.8%) Profile: Default

## 5. Notice the Mapping Changed

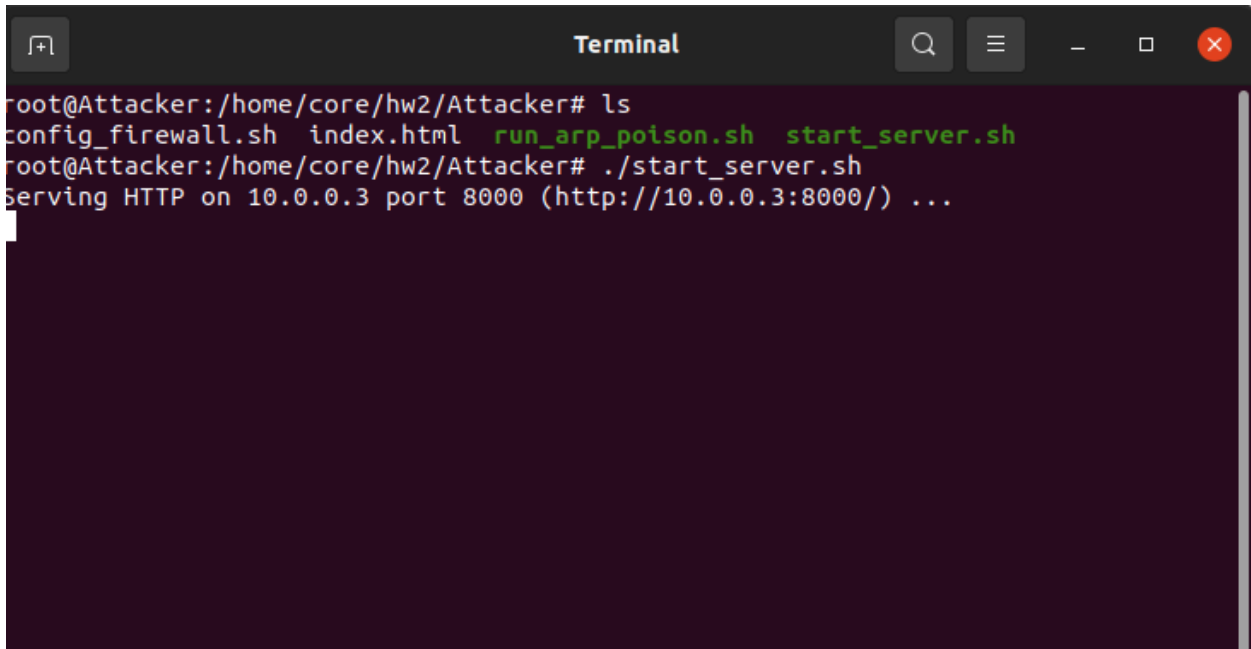
The ARP poisoning attack I ran was successful. The Client's ARP table now maps the server's IP address to the attacker's MAC address, meaning that all traffic from the client to the server is now routed through the attacker. I used the command `arp -na` to view the ARP table of the client.

A terminal window titled "Terminal" with a dark background. The prompt is "root@Client:/tmp/pycore.1/Client.conf#". The user enters "arp -na" twice. The output shows four entries in the ARP table, all with the same MAC address "00:00:00:aa:00:02" for different IP addresses. The first entry is for IP 10.0.0.4, and the others are for 10.0.0.2, 10.0.0.3, and 10.0.0.4.

```
root@Client:/tmp/pycore.1/Client.conf# arp -na
? (10.0.0.4) at 00:00:00:aa:00:03 [ether] on eth0
root@Client:/tmp/pycore.1/Client.conf# arp -na
? (10.0.0.2) at 00:00:00:aa:00:02 [ether] on eth0
? (10.0.0.3) at 00:00:00:aa:00:02 [ether] on eth0
? (10.0.0.4) at 00:00:00:aa:00:02 [ether] on eth0
root@Client:/tmp/pycore.1/Client.conf#
```

## 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`.

A terminal window titled "Terminal" with a dark background. The prompt is "root@Attacker:/home/core/hw2/Attacker#". The user enters "ls", showing a list of files including "config\_firewall.sh", "index.html", "run\_arp\_poison.sh", and "start\_server.sh". Then, the user enters "start\_server.sh", and the output shows "Serving HTTP on 10.0.0.3 port 8000 (http://10.0.0.3:8000/) ...".

```
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# ./start_server.sh
Serving HTTP on 10.0.0.3 port 8000 (http://10.0.0.3:8000/) ...
```

## 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>

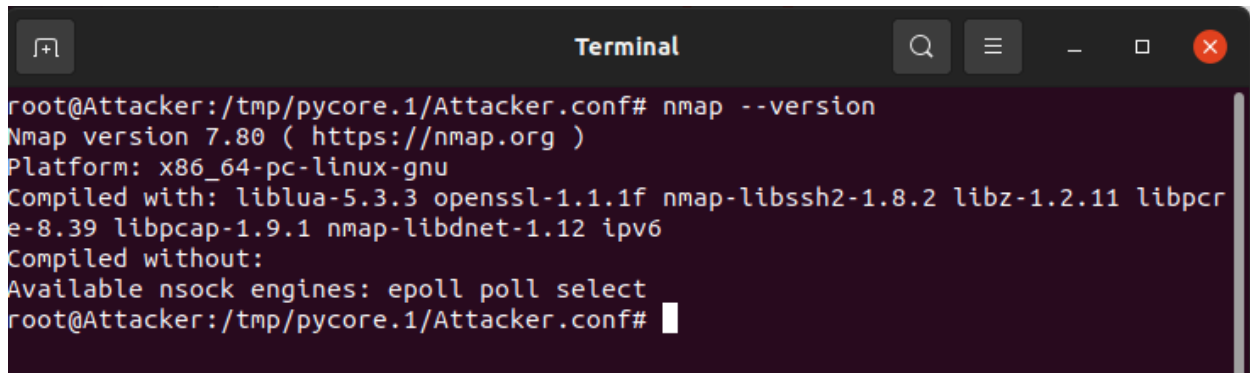
--Tue 10 Sep 2024 05:46:49 PM EDT--
<html>
  <b> Attacker Server </b>
</html>

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

## 8. OS Fingerprinting Scan using Nmap

### a) Install Nmap

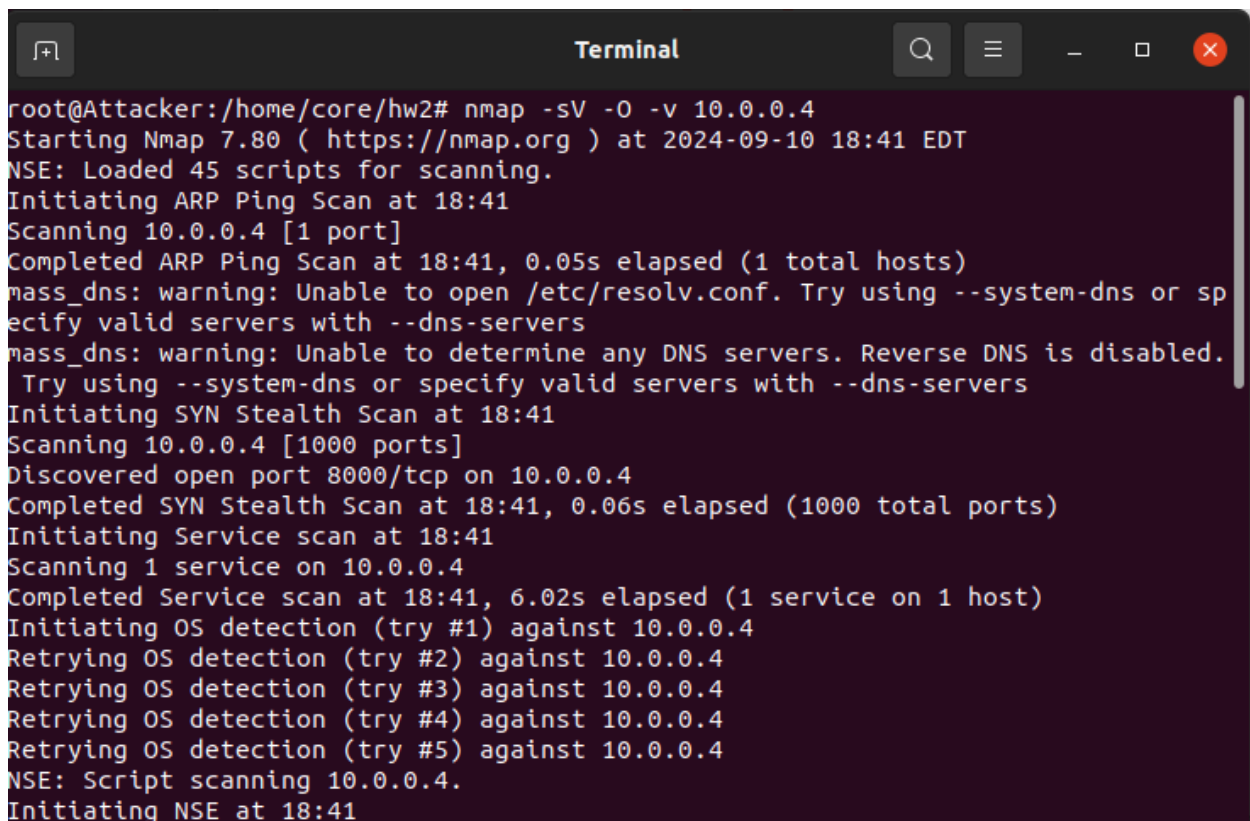
I installed Nmap using the command `sudo apt-get -y install nmap` and checked its version using the command `nmap -version`.

A terminal window titled "Terminal" with a dark background. The prompt is "root@Attacker:/tmp/pycore.1/Attacker.conf#". The command "nmap --version" has been executed, displaying the following output: "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# 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`.

A terminal window titled "Terminal" with a dark background. The prompt is "root@Attacker:/home/core/hw2#". The command "nmap -sV -O -v 10.0.0.4" has been executed, displaying the following output: "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".

```
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
```

```
Terminal
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:

```
Terminal
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
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)
```