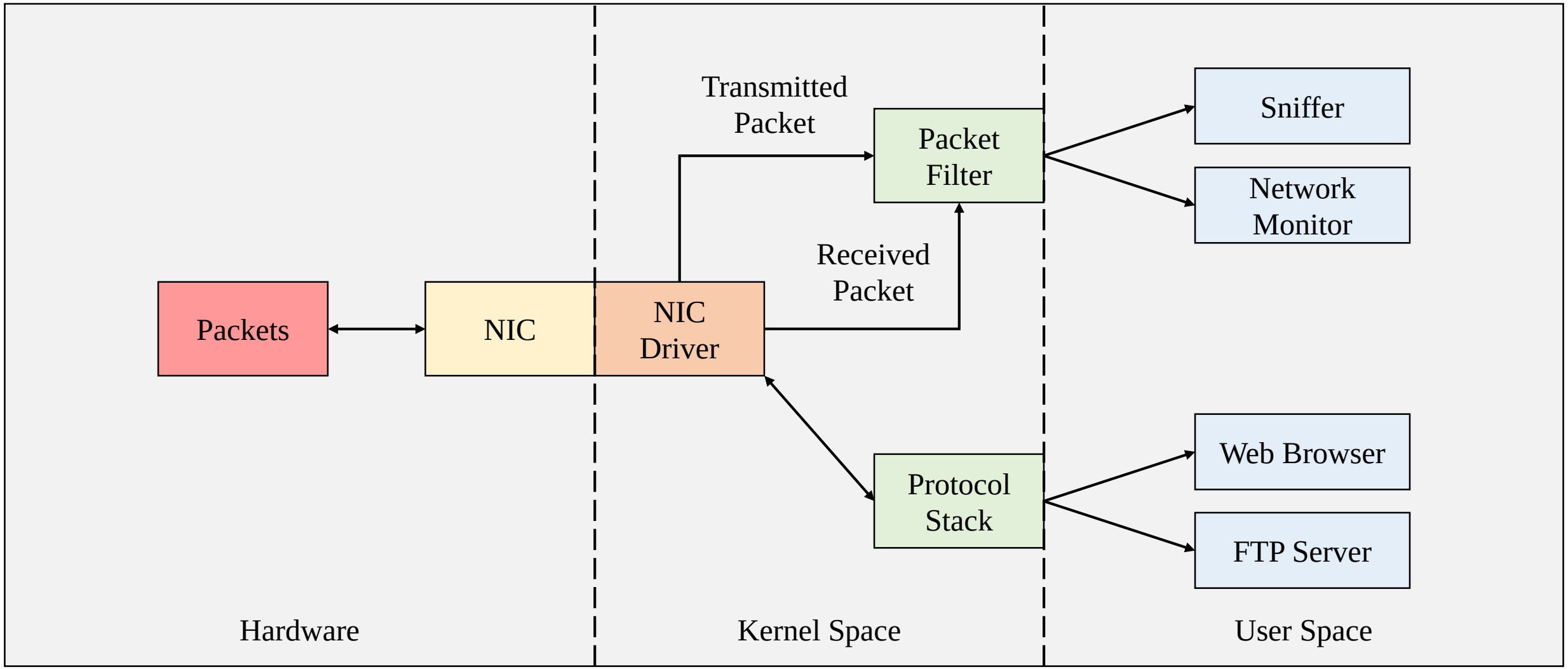
Packet Sniffing with RAW Socket

유명성

1. IP Packet Parsing

1. IP Packet Parsing

1.1 packet capture in kernel



Struct module

- Encoding된 이진 데이터를 python bytes 데이터로 해석해주는 모듈.
- https://docs.python.org/3/library/struct.html
- Python Bytes로 표현된 C-구조체와 Python 값 사이의 변환을 수행.
- 네트워크 등 다른 소스에 저장된 Binary 데이터를 처리할 때 사용
- 주요 메소드
  - struct.pack(format, v1, v2, …) : format에 맞게 Python 값 v1, v2, …를 연결한 Bytes를 리턴
  - struct.unpack(format, buf) : buf를 foamat에 맞는 Python 값으로 쪼갠 Tuple을 리턴

1. IP Packet Parsing

1.2 Struct module

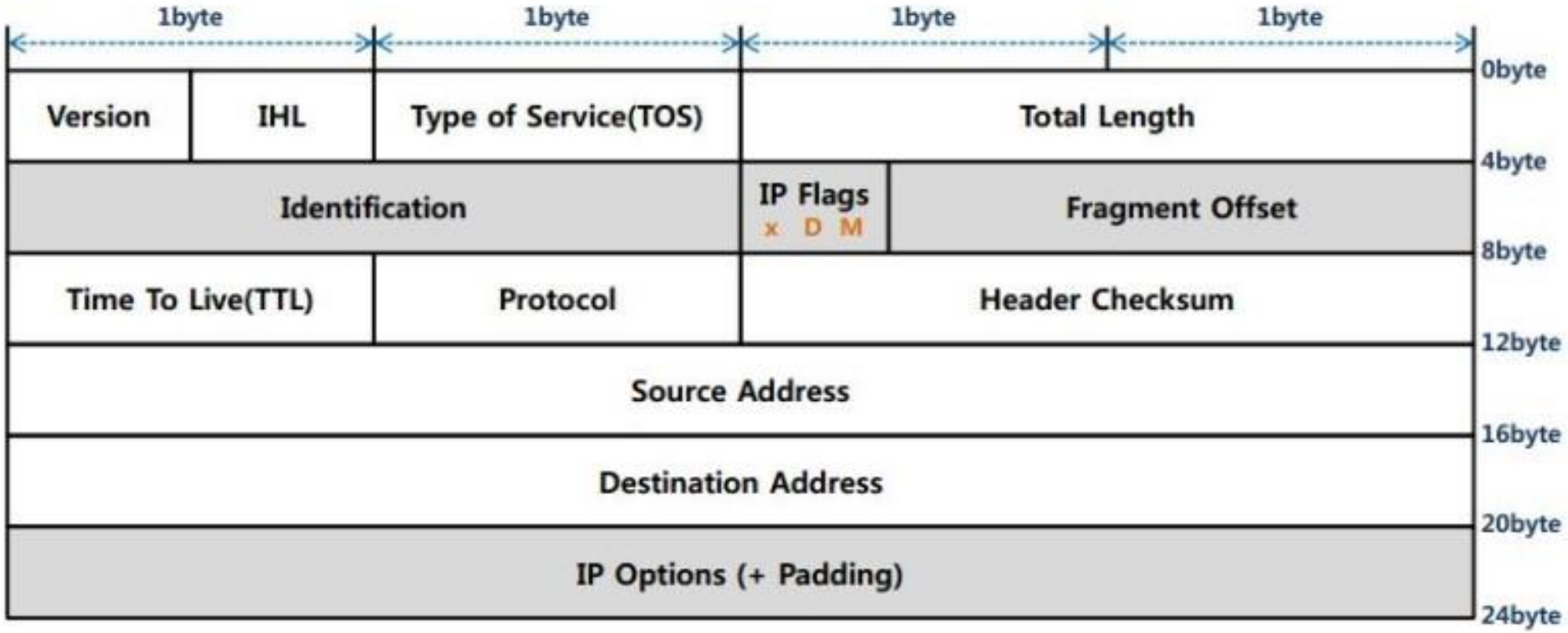
여러 Python Value를 엔디안, 구조체 정렬 등을
고려해 Bytes로 바꿀 때

pack(‘format’ , *arg) ↔ unpack(‘format’ , buf)

네트워크 등에서 수신한 Bytes를
엔디안, 구조체 정렬 등을 고려해 Python Value로 바꿀 때

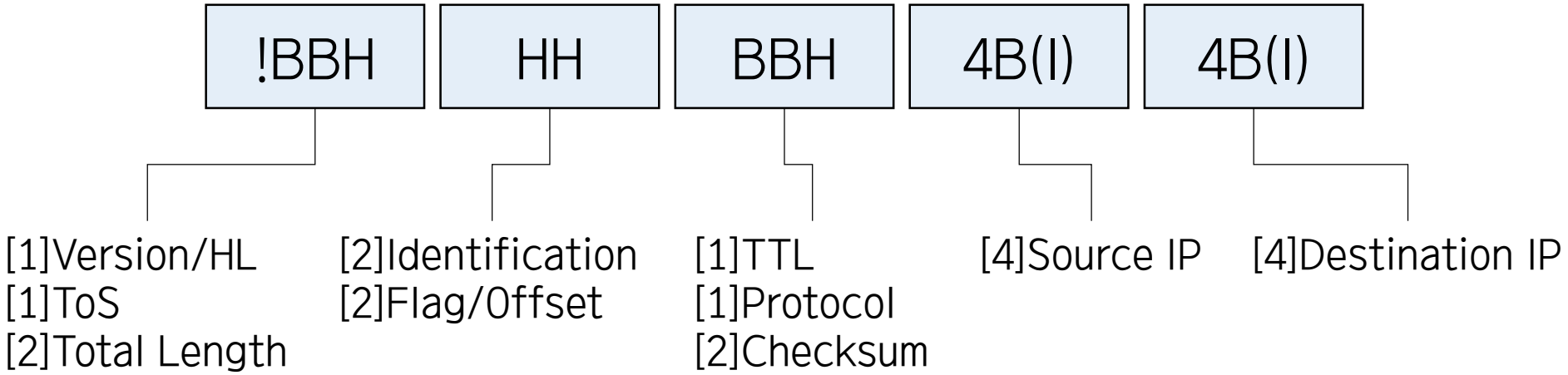
1. IP Packet Parsing

1.3 IPv4 header



1. IP Packet Parsing

1.4 IPv4 header parsing



1. IP Packet Parsing

3.3 IPv4 header parsing

1. while문을 통해 여러 번 동작하도록 작성
2. Ethertype를 확인해 IP(0x0800)일 경우에만 동작하도록 작성
3. IP 헤더 파싱 전에 headerlength 부분을 먼저 읽어 헤더 길이 만큼 IP 패킷 잘라보기
4. IP 헤더는 옵션은 고려하지 않고 20Byte 부분만 파싱해 출력

```python
    try:
        with socket.socket(socket.AF_PACKET, socket.SOCK_RAW, socket.ntohs(ETH_P_ALL)) as sniffe_sock:
            sniffe_sock.bind((args.i, 0))

            while True:
                data, _ = sniffe_sock.recvfrom(MTU)
                parse_packet(data)

    except KeyboardInterrupt:
        print('EXIT...')

    except socket.error as e:
        print(e)
```

```python
def parse_packet(data):
    eth_header = make_ethernet_header(data[:ETH_SIZE])

    if eth_header['ether_type'] != ETH_P_IP:
        return

    ip_header = make_ip_header(data[ETH_SIZE:])
    print_header(eth_header, ip_header)
    print('\nRaw Data')
    dumpcode(data)

    print('\n')
```

```python
43 def make_ip_header(raw_data):
44     ip_ver_and_hlen = struct.unpack('!B', raw_data[:1])[0]
45
46     ip_hlen = ip_ver_and_hlen & 0x0F
47     raw_data = raw_data[:ip_hlen*4]
48
49     ip = struct.unpack('!BHHHBBH', raw_data[1:12])
50     return {'version':ip_ver_and_hlen >> 4,
51     'header_length':ip_hlen,
52     'tos':ip[0],
53     'total_length':ip[1],
54     'id':ip[2],
55     'flag':ip[3] >> 13,
56     'offset':ip[3] & 0x1FFF,
57     'ttl':ip[4],
58     'protocol':ip[5],
59     'checksum':ip[6],
60     'src':socket.inet_ntoa(raw_data[12:16]),
61     'dst':socket.inet_ntoa(raw_data[16:20])
62     }
```

1. IP Packet Parsing

1.4 IPv4 header parsing

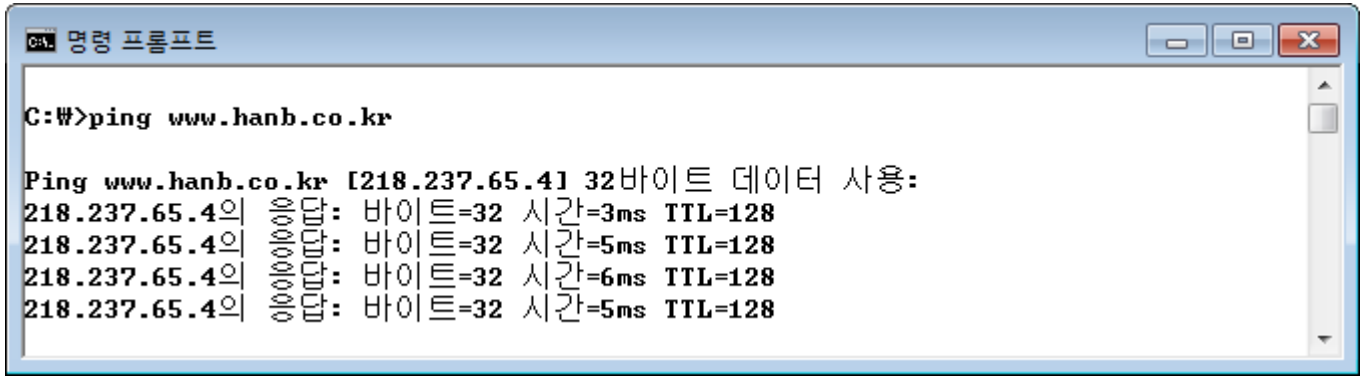
```
[2] IP_PACKET-------------------------------------------

Ethernet Header
[dst] 00:0c:29:44:5e:1b
[src] 00:50:56:fd:07:5c
[ether_type] 2048

IP HEADER
[version] 4
[header_length] 5
[tos] 0
[total_length] 84
[id] 20301
[flag] 0
[offset] 0
[ttl] 128
[protocol] 1
[checksum] 21019
[src] 8.8.8.8
[dst] 192.168.200.136

Raw Data
offset 00 01 02 03 04 05 06 07 - 08 09 0a 0b 0c 0d 0e 0f
0x0000 00 0c 29 44 5e 1b 00 50 - 56 fd 07 5c 08 00 45 00
0x0010 00 54 4f 4d 00 00 80 01 - 52 1b 08 08 08 08 c0 a8
0x0020 c8 88 00 00 61 cd 21 e0 - 00 01 fb 44 c5 5c 00 00
0x0030 00 00 f6 dc 06 00 00 00 - 00 00 10 11 12 13 14 15
0x0040 16 17 18 19 1a 1b 1c 1d - 1e 1f 20 21 22 23 24 25
0x0050 26 27 28 29 2a 2b 2c 2d - 2e 2f 30 31 32 33 34 35
0x0060 36 37
```

2. Traceroute

2. Traceroute

2.1 ICMP

- **Ping 응용 프로그램**
  - 호스트나 라우터의 작동 여부를 확인할 때 사용
  - ICMP 프로토콜을 이용하여 구현



ICMP

- 인터넷에 연결된 호스트나 라우터 간에 유용한 정보(오류 발생, 라우팅 정보 등)를 알리는 목적으로 사용
- 항상 IP 패킷에 포함된 형태로 전송되며 TCP/IP 프로토콜 동작에 필수 역할을 함

IP 헤더	ICMP 메시지

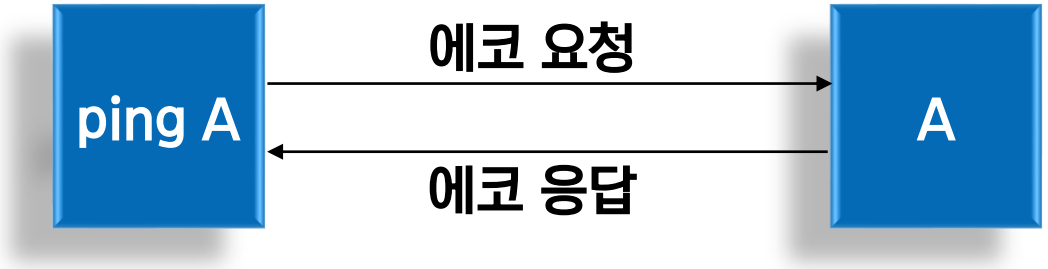
ICMP 메시지 구조

0 – 7	8 – 15	16 – 31
Type	Code	Checksum
Type과 Code에 따라 달라지는 부분(가변 길이)		

ICMP 메시지 정의

```
typedef struct _ICMPMESSAGE
{
    u_char  icmp_type;        // type of message
    u_char  icmp_code;        // type sub code
    u_short icmp_cksum;// checksum
    …
} ICMPMESSAGE;
```

Ping 응용 프로그램 동작 원리

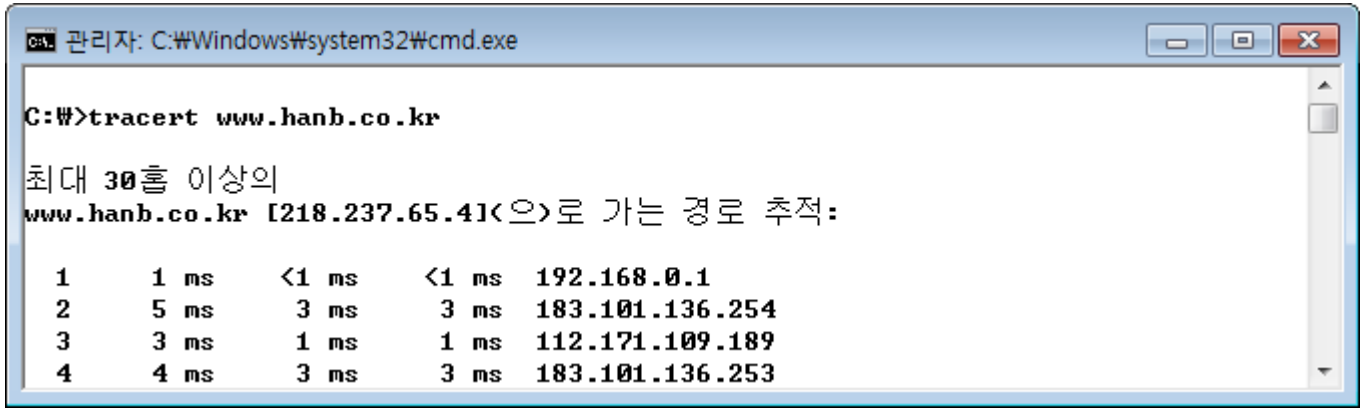


에코 요청, 에코 응답 ICMP 메시지

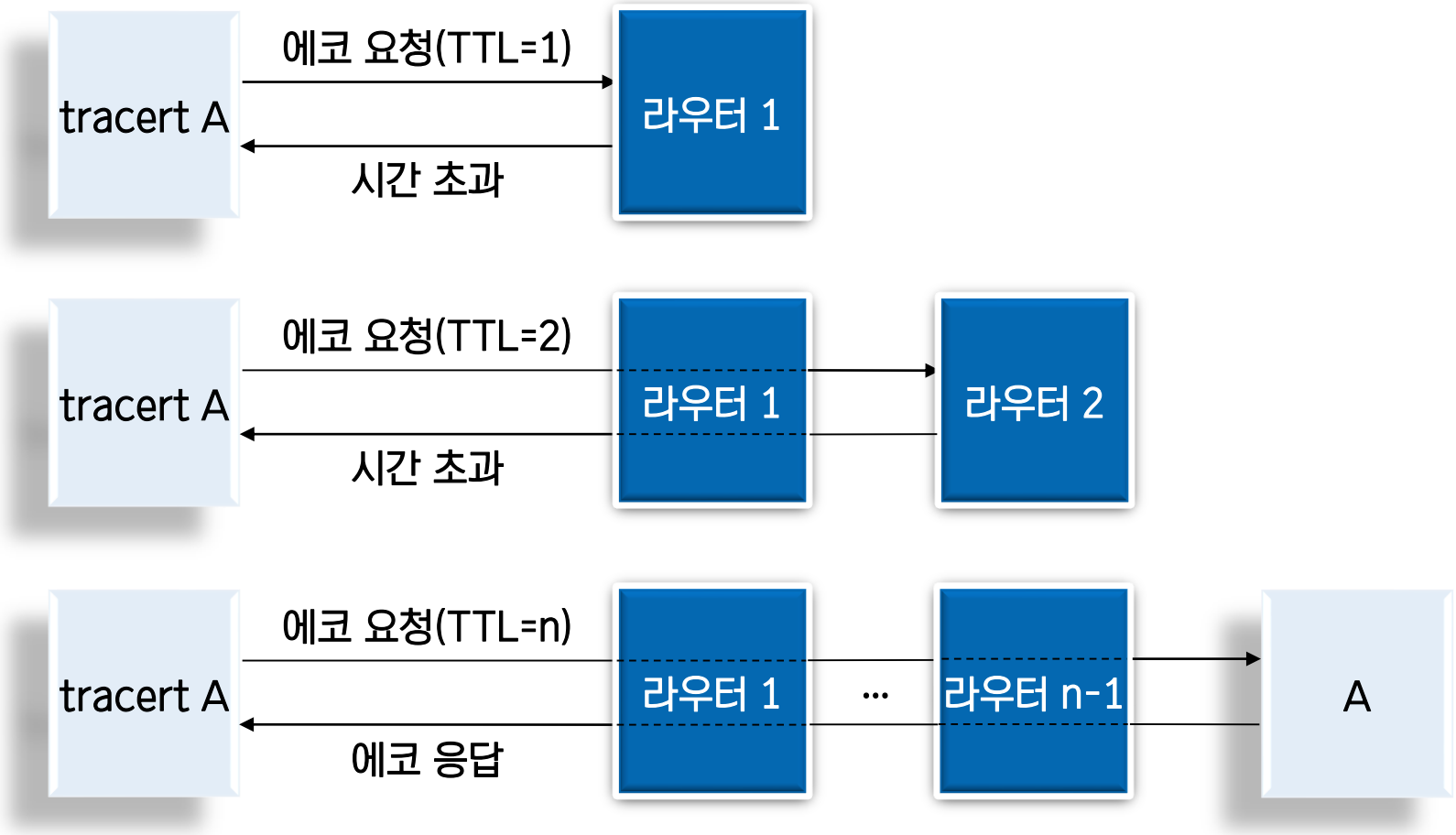
0 ~ 7	8 ~ 15	16 ~ 31
Type(8 또는 0)	Code(0)	Checksum
Identifier		Sequence Number
옵션 데이터(가변 길이)		

Traceroute 응용 프로그램

- 호스트나 라우터까지의 IP 패킷 전달 경로를 확인
- ICMP, UDP, TCP 등의 프로토콜을 이용하여 구현



Traceroute 동작 원리



2. Traceroute

2.2 Traceroute

root@kali: ~ | root@kali: ~ | root@kali: ~

홉수

RTT

```
root@kali:~# traceroute google.com -I
traceroute to google.com (172.217.25.14), 30 hops max, 60 byte packets
 1  _gateway (192.168.200.2)  0.347 ms  0.201 ms  0.255 ms
 2  * * *
 3  * * *
 4  * * *
 5  * * *
 6  * * *
 7  * * *
 8  * * *
 9  * * *
10  * * *
11  * * *
12  * * *
13  * * *
14  * * *
15  * * *
16  hkg07s24-in-f14.1e100.net (172.217.25.14)  44.597 ms  44.547 ms  44.443 ms
root@kali:~#
```

2. Traceroute

2.2 Traceroute

icmp

No.	Time	Source	Destination	Protocol	Length	Info
5	0.014305480	192.168.200.140	172.217.25.14	ICMP	74	Echo (ping) request id=0x1f69, seq=1/256, ttl=1 (no response found!)
6	0.014466729	192.168.200.140	172.217.25.14	ICMP	74	Echo (ping) request id=0x1f69, seq=2/512, ttl=1 (no response found!)
7	0.014623902	192.168.200.2	192.168.200.140	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
8	0.014651277	192.168.200.2	192.168.200.140	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
9	0.014753511	192.168.200.140	172.217.25.14	ICMP	74	Echo (ping) request id=0x1f69, seq=3/768, ttl=1 (no response found!)
10	0.014882355	192.168.200.140	172.217.25.14	ICMP	74	Echo (ping) request id=0x1f69, seq=4/1024, ttl=2 (no response found!)
11	0.014987665	192.168.200.2	192.168.200.140	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
12	0.015025884	192.168.200.140	172.217.25.14	ICMP	74	Echo (ping) request id=0x1f69, seq=5/1280, ttl=2 (no response found!)
13	0.015136207	192.168.200.140	172.217.25.14	ICMP	74	Echo (ping) request id=0x1f69, seq=6/1536, ttl=2 (no response found!)
14	0.015248595	192.168.200.140	172.217.25.14	ICMP	74	Echo (ping) request id=0x1f69, seq=7/1792, ttl=3 (no response found!)
15	0.015355581	192.168.200.140	172.217.25.14	ICMP	74	Echo (ping) request id=0x1f69, seq=8/2048, ttl=3 (no response found!)
16	0.015461136	192.168.200.140	172.217.25.14	ICMP	74	Echo (ping) request id=0x1f69, seq=9/2304, ttl=3 (no response found!)

```
▸ Frame 5: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
▸ Ethernet II, Src: Vmware_a1:0a:b4 (00:0c:29:a1:0a:b4), Dst: Vmware_fd:07:5c (00:50:56:fd:07:5c)
▸ Internet Protocol Version 4, Src: 192.168.200.140, Dst: 172.217.25.14
▸ Internet Control Message Protocol
```

2. Traceroute

2.2 Traceroute

11	0.014987665	192.168.200.2	192.168.200.140	ICMP	102 Time-to-live exceeded (Time to live exceeded in transit)
12	0.015025884	192.168.200.140	172.217.25.14	ICMP	74 Echo (ping) request id=0x1f69, seq=5/1280, ttl=2 (no response found!)
13	0.015136207	192.168.200.140	172.217.25.14	ICMP	74 Echo (ping) request id=0x1f69, seq=6/1536, ttl=2 (no response found!)
14	0.015248595	192.168.200.140	172.217.25.14	ICMP	74 Echo (ping) request id=0x1f69, seq=7/1792, ttl=3 (no response found!)
15	0.015355581	192.168.200.140	172.217.25.14	ICMP	74 Echo (ping) request id=0x1f69, seq=8/2048, ttl=3 (no response found!)
16	0.015461136	192.168.200.140	172.217.25.14	ICMP	74 Echo (ping) request id=0x1f69, seq=9/2304, ttl=3 (no response found!)

```
▸ Frame 11: 102 bytes on wire (816 bits), 102 bytes captured (816 bits) on interface 0
▸ Ethernet II, Src: Vmware_fd:07:5c (00:50:56:fd:07:5c), Dst: Vmware_a1:0a:b4 (00:0c:29:a1:0a:b4)
▸ Internet Protocol Version 4, Src: 192.168.200.2, Dst: 192.168.200.140
▾ Internet Control Message Protocol
    Type: 11 (Time-to-live exceeded)
    Code: 0 (Time to live exceeded in transit)
    Checksum: 0xf4ff [correct]
    [Checksum Status: Good]
  ▸ Internet Protocol Version 4, Src: 192.168.200.140, Dst: 172.217.25.14
  ▸ Internet Control Message Protocol
```

```
0000  00 0c 29 a1 0a b4 00 50  56 fd 07 5c 08 00 45 00   ··)····P V··\··E·
0010  00 58 0a 9e 00 00 80 01  1e 27 c0 a8 c8 02 c0 a8   ·X····· ·'......
0020  c8 8c 0b 00 f4 ff 00 00  00 00 45 00 00 3c 8c 76   ····· ··E··<·v
0030  00 00 01 01 de 2e c0 a8  c8 8c ac d9 19 0e 08 00   ····· ·· ·····
0040  63 0e 1f 69 00 03 48 49  4a 4b 4c 4d 4e 4f 50 51   c··i··HI JKLMNOPQ
0050  52 53 54 55 56 57 58 59  5a 5b 5c 5d 5e 5f 60 61   RSTUVWXY Z[\]^_`a
0060  62 63 64 65 66 67                                  bcdefg
```

2. Traceroute

2.2 Traceroute

```
root@kali: ~        root@kali: ~        root@kali: ~        root@kali: ~

root@kali:~# traceroute google.com -U
traceroute to google.com (172.217.161.142), 30 hops max, 60 byte packets
 1  _gateway (192.168.200.2)  0.148 ms  0.064 ms  0.146 ms
 2  * * *
 3  * * *
 4  * * *
 5  * * *
 6  * * *
 7  * * *
 8  * * *
 9  * * *
10  * * *
11  * * *
12  * * *
13  * * *
14  * * *
```

-U : UDP 사용, -T : TCP 사용

2. Traceroute

2.2 Traceroute

7	0.033270190	192.168.200.140	172.217.161.142	DNS	74	Unknown operation (8) 0x4041[Malformed Packet]
8	0.033354684	192.168.200.140	172.217.161.142	DNS	74	Unknown operation (8) 0x4041[Malformed Packet]
9	0.033407607	192.168.200.2	192.168.200.140	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
10	0.033414147	192.168.200.2	192.168.200.140	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
11	0.033441787	192.168.200.140	172.217.161.142	DNS	74	Unknown operation (8) 0x4041[Malformed Packet]
12	0.033543595	192.168.200.140	172.217.161.142	DNS	74	Unknown operation (8) 0x4041[Malformed Packet]
13	0.033583564	192.168.200.2	192.168.200.140	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
14	0.033609882	192.168.200.140	172.217.161.142	DNS	74	Unknown operation (8) 0x4041[Malformed Packet]

```
▸ Frame 7: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
▸ Ethernet II, Src: Vmware_a1:0a:b4 (00:0c:29:a1:0a:b4), Dst: Vmware_fd:07:5c (00:50:56:fd:07:5c)
▾ Internet Protocol Version 4, Src: 192.168.200.140, Dst: 172.217.161.142
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 60
    Identification: 0xa4f5 (42229)
  ▸ Flags: 0x0000
  ▸ Time to live: 1
    Protocol: UDP (17)
    Header checksum: 0x3d1f [validation disabled]
    [Header checksum status: Unverified]
    Source: 192.168.200.140
```

```
0000  00 50 56 fd 07 5c 00 0c  29 a1 0a b4 08 00 45 00   ·PV··\·· )·····E·
0010  00 3c a4 f5 00 00 01 11  3d 1f c0 a8 c8 8c ac d9   ·<····· =·******
0020  a1 8e 86 8b 00 35 00 28  d7 d6 40 41 42 43 44 45   ·····5·( ··@ABCDE
0030  46 47 48 49 4a 4b 4c 4d  4e 4f 50 51 52 53 54 55   FGHIJKLM NOPQRSTU
0040  56 57 58 59 5a 5b 5c 5d  5e 5f                     VWXYZ[\] ^_
```

2. Traceroute

2.2 Traceroute

5	0.027879076	192.168.200.140	216.58.220.206	TCP	74	44559 → 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=2540447981 TSecr=0 WS=4
6	0.028096700	192.168.200.2	192.168.200.140	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
7	0.028137618	192.168.200.140	216.58.220.206	TCP	74	34663 → 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=2540447981 TSecr=0 WS=4
8	0.028267636	192.168.200.2	192.168.200.140	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
9	0.028322206	192.168.200.140	216.58.220.206	TCP	74	39579 → 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=2540447981 TSecr=0 WS=4
10	0.028377795	192.168.200.140	216.58.220.206	TCP	74	33805 → 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=2540447981 TSecr=0 WS=4
11	0.028418686	192.168.200.2	192.168.200.140	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
12	0.028423203	192.168.200.140	216.58.220.206	TCP	74	54443 → 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=2540447981 TSecr=0 WS=4

```
.... 0101 = Header Length: 20 bytes (5)
Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 60
Identification: 0x6d95 (28053)
Flags: 0x0000
Time to live: 1
Protocol: TCP (6)
Header checksum: 0x0de9 [validation disabled]
[Header checksum status: Unverified]
Source: 192.168.200.140
Destination: 216.58.220.206
Transmission Control Protocol, Src Port: 44559, Dst Port: 80, Seq: 0, Len: 0
Source Port: 44559
Destination Port: 80
```

Traceroute 구현하기

1. -I : ICMP, -U : UDP
2. -d : 목적지 ip 혹은 도메인 네임
3. -h : 최대 홉수
4. -t : timeout

Term Project #1

- ICMP Echo Request 패킷을 작성해서 전송하는 프로그램 작성
  - 소켓은 : socket(socket.AF_INET, socket.SOCK_RAW, socket.IPPROTO_RAW)사용
  - struct 모듈을 사용해 직접 IP, ICMP의 내용 작성
  - -d : 목적지 ip 주소 혹은 도메인 네임
  - 프로그램 실행 뒤 google.com에 PING을 1번 보낸 결과를 wireshark로 캡쳐해 첨부
- 팀 대표가 barcel@naver.com으로 제출 (5.21일까지)
  - Title : [컴퓨터네트워크][학번][이름][과제_N]
  - Content : github repo url

    팀명 : 길동이네

    팀원 : 홍길동(학번), 고길동(학번)

2. Traceroute

2.3 Term project

```
root@ubuntu: /home/famous/Desktop/TA/socket/assign...   ×   root@ubuntu: /home/famou
root@ubuntu:~# python3 icmp.py -d google.com
root@ubuntu:~#
```

icmp

No.	Time	Source	Destination	Protocol	Length	Info
10	15.406273554	192.168.200.136	172.217.31.142	ICMP	44	Echo (ping) request id=0x0000, seq=0/0, ttl=255 (reply in 11)
11	15.483151740	172.217.31.142	192.168.200.136	ICMP	60	Echo (ping) reply id=0x0000, seq=0/0, ttl=128 (request in 10)

ICMP Checksum 계산

1. 헤더의 checksum 필드를 0x0000으로 채운다.
2. 헤더를 2byte 단위로 끊어서 더한다. 만약 홀수라면 0x00을 더한다.
3. 더한 값이 4byte 이상이라면 올림수를 값에 다시 더한다.
4. 3에서 계산한 값에 1의 보수를 취한다.

2. Traceroute

2.3 Term project

```
▾ Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0xf79b [correct]
    [Checksum Status: Good]
    Identifier (BE): 0 (0x0000)
    Identifier (LE): 0 (0x0000)
    Sequence number (BE): 0 (0x0000)
    Sequence number (LE): 0 (0x0000)

0000  00 50 56 fd 07 5c 00 0c  29 44 5e 1b 08 00 45 00   ·PV··\·· )D^···E·
0010  00 1e d4 31 00 00 ff 01  92 14 c0 a8 c8 88 ac d9   ···1···· ........
0020  1f 8e 08 00 f7 9b 00 00  00 00 00 64               ······· ···d
```

0x0800 + 0x0000 + 0x0000 + 0x0000 + 0x0064 = 0x864

~0x864 = 0xf79b