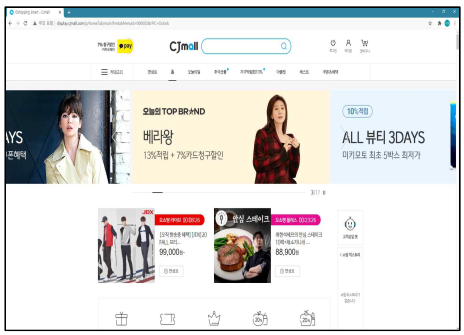
Computer Network Project 1

2015147531_서기원

1. Wireshark

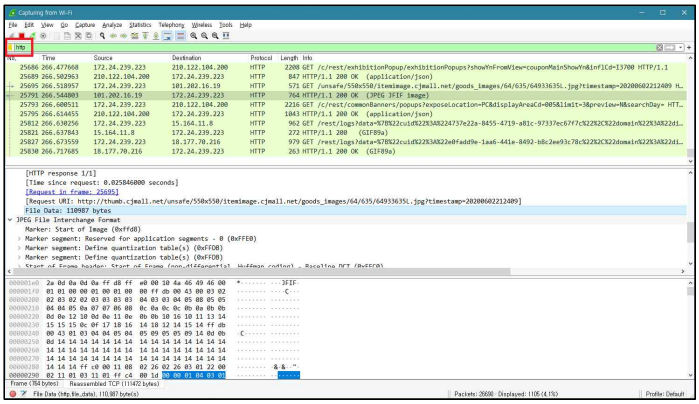
1-1. 대상 : CJ몰 홈페이지 (http://display.cjmall.com)



1-2. 과정

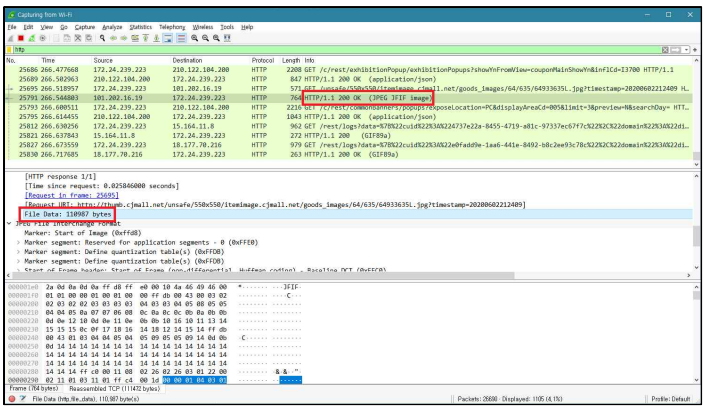
Step 1.

Display filter를 “http”로 설정하여 프로토콜이 http인 항목들만 리스트에 기재되도록 합니다.



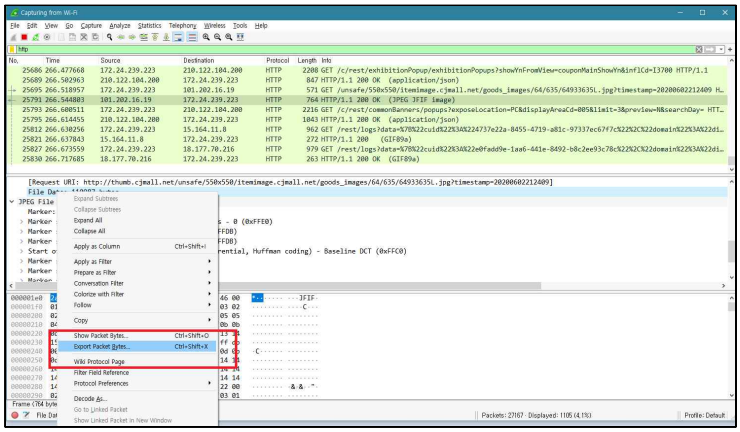
Step 2.

해당 사이트에 접속하였을 때, 올라오는 리스트들의 info 중에서 정상적으로 request를 받았다는 신호인 “HTTP/1.1 200 OK” 항목 중에서 수신한 패킷이 “JPEG JFIF image”이거나 “PNG”인 항목들을 선택합니다.



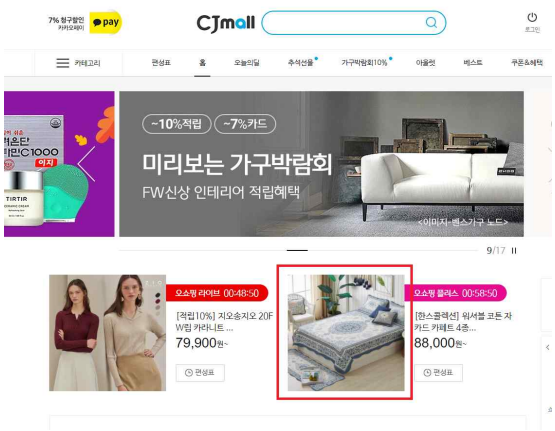
Step 3.

“Hypertext Transfer Protocol > File Data: ~~” 부분을 찾아내어 “Export Packet Bytes”를 이용하여 해당하는 이미지 파일의 확장자로 이미지를 저장합니다.

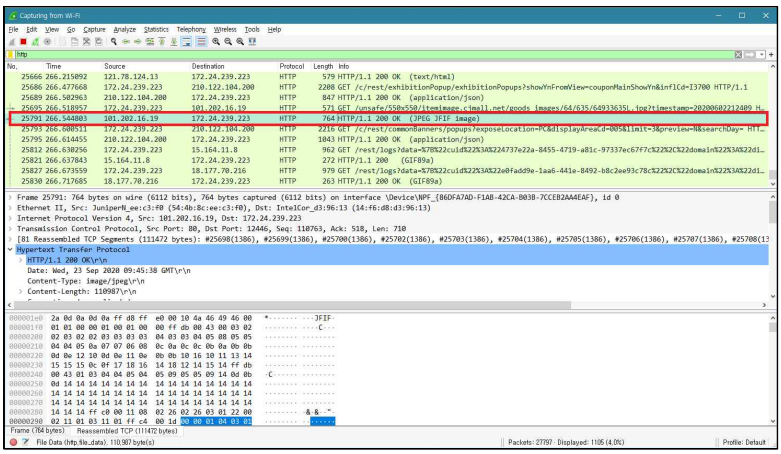


Step 4.

저장한 이미지 파일이 원래의 사이트에서 개시된 이미지 파일인지 확인합니다.



1-3. 패킷 정보.



2. iPerf3

2-1. 버전 정보

```
giwon@giwon-virtual-machine:~$ iperf3 --version
iperf 3.7 (cJSON 1.5.2)
Linux giwon-virtual-machine 5.4.0-47-generic #51-Ubuntu SMP Fri Sep 4 19:50:52 UTC 2020 x86_64
Optional features available: CPU affinity setting, IPv6 flow label, SCTP, TCP congestion algorithm settin
g, sendfile / zerocopy, socket pacing, authentication
```

2-2. Screenshot of iPerf3 execution on localhost.

[Server]

```
giwon@giwon-virtual-machine:~$ iperf3 -s
-----------------------------------------------------------
Server listening on 5201
-----------------------------------------------------------
Accepted connection from 172.24.234.94, port 36248
[  5] local 172.24.234.94 port 5201 connected to 172.24.234.94 port 36250
[ ID] Interval           Transfer     Bitrate
[  5]   0.00-1.00   sec  2.27 GBytes  19.5 Gbits/sec
[  5]   1.00-2.00   sec  2.46 GBytes  21.2 Gbits/sec
[  5]   2.00-3.00   sec  2.57 GBytes  22.1 Gbits/sec
[  5]   3.00-4.00   sec  2.54 GBytes  21.8 Gbits/sec
[  5]   4.00-5.00   sec  2.20 GBytes  18.9 Gbits/sec
[  5]   5.00-6.00   sec  2.33 GBytes  20.0 Gbits/sec
[  5]   6.00-7.00   sec  2.20 GBytes  18.9 Gbits/sec
[  5]   7.00-8.00   sec  2.35 GBytes  20.2 Gbits/sec
[  5]   8.00-9.00   sec  2.24 GBytes  19.3 Gbits/sec
[  5]   9.00-10.00  sec  2.43 GBytes  20.9 Gbits/sec
[  5]  10.00-10.00  sec   384 KBytes  11.7 Gbits/sec
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bitrate
[  5]   0.00-10.00  sec  23.6 GBytes  20.3 Gbits/sec                  receiver
-----------------------------------------------------------
Server listening on 5201
-----------------------------------------------------------
```

[Client]

```
giwon@giwon-virtual-machine:~$ iperf3 -c 172.24.234.94
Connecting to host 172.24.234.94, port 5201
[  5] local 172.24.234.94 port 36250 connected to 172.24.234.94 port 5201
[ ID] Interval           Transfer     Bitrate         Retr  Cwnd
[  5]   0.00-1.00   sec  2.27 GBytes  19.5 Gbits/sec    0   2.19 MBytes
[  5]   1.00-2.00   sec  2.46 GBytes  21.2 Gbits/sec    0   2.19 MBytes
[  5]   2.00-3.00   sec  2.57 GBytes  22.1 Gbits/sec    0   2.31 MBytes
[  5]   3.00-4.00   sec  2.54 GBytes  21.8 Gbits/sec    0   2.31 MBytes
[  5]   4.00-5.00   sec  2.20 GBytes  18.9 Gbits/sec    0   2.31 MBytes
[  5]   5.00-6.00   sec  2.33 GBytes  20.0 Gbits/sec    0   2.31 MBytes
[  5]   6.00-7.00   sec  2.20 GBytes  18.9 Gbits/sec    0   3.12 MBytes
[  5]   7.00-8.00   sec  2.35 GBytes  20.2 Gbits/sec    0   3.12 MBytes
[  5]   8.00-9.00   sec  2.24 GBytes  19.3 Gbits/sec    0   3.12 MBytes
[  5]   9.00-10.00  sec  2.44 GBytes  20.9 Gbits/sec    0   3.12 MBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bitrate         Retr
[  5]   0.00-10.00  sec  23.6 GBytes  20.3 Gbits/sec    0             sender
[  5]   0.00-10.00  sec  23.6 GBytes  20.3 Gbits/sec                  receiver

iperf Done.
giwon@giwon-virtual-machine:~$
```

2-2. Explanation

transfer : 해당하는 시간 간격 동안 client에서 server로 송수신된 데이터의 크기를 의미합니다.

bitrate : 전송속도를 나타내는 수단 중의 하나이며 어떤 네트워크 또는 시스템의 한 지점에서 다른 지점으로 데이터가 이동하는 속도를 특정한 시간 단위마다 처리하는 비트의 수를 나타낸 것을 의미합니다.

cwnd : “Congestion Window”의 줄임말로 acknowledge character를 받기 이전에 서버로 TCP가 서버로 전송할 데이터의 크기를 제한하는 TCP 상태 변수입니다. TCP data flow를 통제하고, 막힘을 최소화시키는 기능을 가졌으며, 최종적으로 네트워크 성능을 증가시킵니다.

```
.24.234.94 -u

ed to 172.24.234.94 port 5201
e         Total Datagrams
bits/sec  4
bits/sec  4
bits/sec  4
bits/sec  4
bits/sec  4
bits/sec  4
bits/sec  4
bits/sec  4
bits/sec  4
bits/sec  4
- - -
e         Jitter    Lost/Total Datagrams
bits/sec  0.000 ms  0/40 (0%)  sender
bits/sec  0.011 ms  0/40 (0%)  receiver
```

(UDP를 이용한 경우)

```
24.234.94

d to 172.24.234.94 port 5201
         Retr  Cwnd
its/sec    0   1.44 MBytes
its/sec    0   1.56 MBytes
its/sec    0   1.94 MBytes
its/sec    0   2.00 MBytes
its/sec    0   2.00 MBytes
its/sec    0   2.00 MBytes
its/sec    0   2.31 MBytes
its/sec    0   2.31 MBytes
its/sec    0   2.31 MBytes
its/sec    0   2.31 MBytes
- -
         Retr
its/sec    0             sender
its/sec                  receiver
```

(TCP를 이용한 경우)

2-3. Command description

“iperf -h” 명령어를 활용하여 해당 명령어들의 기능을 알아보았으며, 보충 설명은 iperf 홈페이지에서 참고하였습니다.

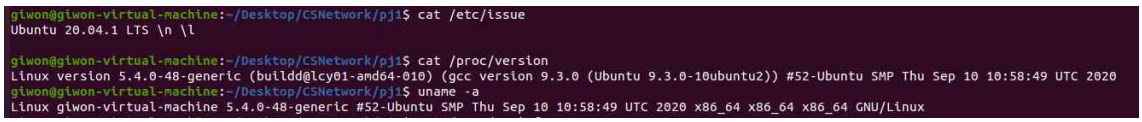
```
Client specific:
  -c, --client    <host>    run in client mode, connecting to <host>
  --sctp                    use SCTP rather than TCP
  -X, --xbind <name>        bind SCTP association to links
  --nstreams      #         number of SCTP streams
  -u, --udp                 use UDP rather than TCP
  --connect-timeout #       timeout for control connection setup (ms)
  -b, --bitrate #[KMG][/#]  target bitrate in bits/sec (0 for unlimited)
                            (default 1 Mbit/sec for UDP, unlimited for TCP)
                            (optional slash and packet count for burst mode)
  --pacing-timer #[KMG]     set the timing for pacing, in microseconds (default 1000)
  --fq-rate #[KMG]          enable fair-queuing based socket pacing in
                            bits/sec (Linux only)
  -t, --time      #         time in seconds to transmit for (default 10 secs)
  -n, --bytes     #[KMG]    number of bytes to transmit (instead of -t)
  -k, --blockcount #[KMG]   number of blocks (packets) to transmit (instead of -t or -n)
  -l, --length    #[KMG]    length of buffer to read or write
                            (default 128 KB for TCP, dynamic or 1460 for UDP)
  --cport         <port>    bind to a specific client port (TCP and UDP, default: ephemeral port
)
  -P, --parallel  #         number of parallel client streams to run
  -R, --reverse             run in reverse mode (server sends, client receives)
  --bidir                   run in bidirectional mode.
                            Client and server send and receive data.
  -w, --window    #[KMG]    set window size / socket buffer size
  -C, --congestion <algo>   set TCP congestion control algorithm (Linux and FreeBSD only)
  -M, --set-mss   #         set TCP/SCTP maximum segment size (MTU - 40 bytes)
  -N, --no-delay            set TCP/SCTP no delay, disabling Nagle's Algorithm
  -4, --version4            only use IPv4
  -6, --version6            only use IPv6
  -S, --tos N               set the IP type of service, 0-255.
```

-b : '--bitrate'로도 사용 가능하며 전송속도를 n bits/sec로 설정합니다. UDP를 사용할 경우 초기값이 1 Mbits/sec이고 TCP의 경우는 unlimited입니다. 만약 "-P" 명령어를 사용하여 여러 개의 스트림이 존재한다면, 각각의 스트림에 전송속도 제한 작업을 실행합니다.
-n : '--bytes'로도 사용 가능하며 전송할 바이트를 설정합니다.
-w : '--window'로도 사용 가능하며 소켓 버퍼의 크기를 설정합니다. TCP의 경우 TCP window 크기를 설정하여 서버로 보낸 다음 서버에서도 같은 작업을 실행합니다.
-l : '--length'로도 사용 가능하며 읽고 쓰는 버퍼의 크기를 설정합니다. TCP의 경우 기본값이 128KB이며 UDP의 경우 8KB로 설정되어 있습니다.

3. Packet Capturing coding

3-1. Introduction

- Code Language : Python 3.8.2
- OS : Ubuntu
- Library : python-pcapy
- 개발환경:



- 4코어 CPU를 사용하였습니다.

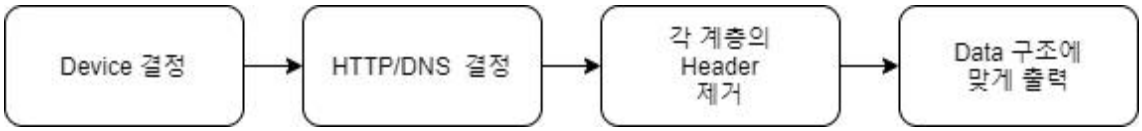
3-2. Block(Function/Class) Description

※ 공간의 이유로 각 함수의 parameter들은 추가하지 않았습니다.

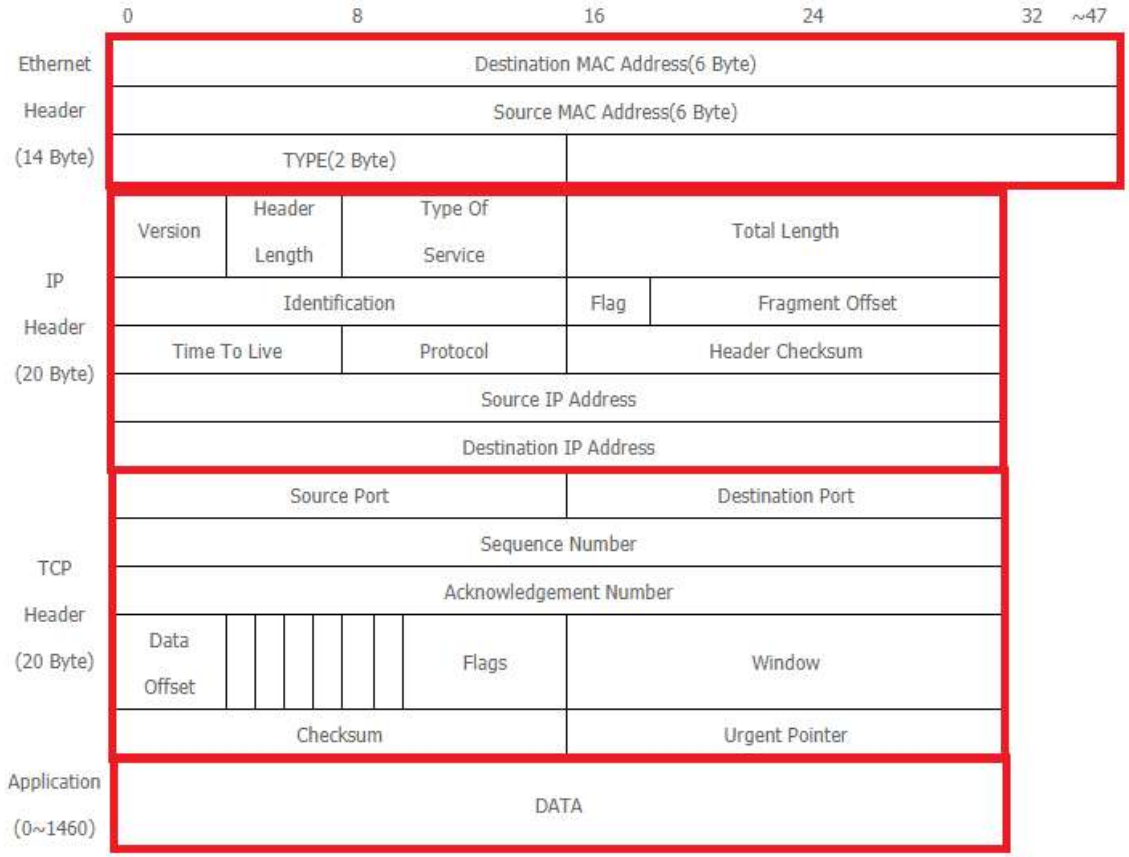
함수 이름	설명
finalldevs()	pcapy 라이브러리의 함수로 현재 장치에서 접근 가능한 네트워크 장비들을 list의 형태로 반환합니다.
open_live()	pcapy 라이브러리의 함수로 네트워크의 패킷들을 보기 위한 packet capture descriptor를 제공합니다. Reader Object를 반환합니다.
next()	pcapy 라이브러리의 함수로 다음 패킷을 읽기 위해 사용합니다.
int.from_bytes()	Python의 내장함수로 바이트 코드를 int 자료형으로 변환합니다.
int_to_IP()	integer를 IP 주소의 형태로 변환합니다.

3-3. Flow Chart

- pcapy 라이브러리를 통해 전달받은 네트워크의 패킷들은 위와 같은 형태를 이루고 있습니다.
- 데이터의 정보를 얻기 위해 4개의 헤더를 분리하는 작업이 필요합니다.



- 각각의 헤더들은 다음의 그림과 같은 형태를 띠고 있습니다.



3-4. Code Screenshot step by step

Step 1. Global Header 제거

```
header, payload = cap.next()
```

- cap.next()를 통해 다음 패킷의 header와 payload를 분리해서 저장합니다.

Step 2. Ethernet Frame 제거

```
# Structure of Frame header
# destination addr(6 byte), source addr(6 byte), type(2 byte)
fhdr = payload[:14]
eth_type = int.from_bytes(fhdr[-2:], byteorder='big')
```

- “Step 1”에서 구한 payload의 앞 14byte에 해당합니다.

```
# 0x0800 -> IP type, Use only IP
if eth_type != 0x0800:
    continue
```

- Ethernet Type이 IP인 것만을 고려합니다.

Step 3. IP Header 제거

```
iphdr_len = payload[14]%16 * 4                    # 4byte per 1
# Protocol types
# {1:ICMP, 2:IGMP, 3:IP, 6:TCP, 8:EGP, 17:UDP, 41:IPv6, 46:RSVP, 89:OSPF}
protocol = payload[13 + 10]
srcIP = int.from_bytes(payload[26:30], byteorder='big')
dstIP = int.from_bytes(payload[30:34], byteorder='big')
srcIP = int_to_IP(srcIP)
dstIP = int_to_IP(dstIP)
# slice the IP header
packet = payload[14 + iphdr_len:]
```

- 위 구조 그림에는 나타나 있지 않지만, IP Header는 뒤에 Option Code가 뒤따라올 수 있습니다. 따라서, IP Header의 정확한 길이를 알기 위해 첫 번째 바이트에서 “Header Length”를 구합니다.
- 이어지는 패킷의 프로토콜을 이번 프로젝트의 경우 “TCP”와 “UDP”로 나누기 위해 해당하는 헤더 정보로 확인합니다.
- 해당 계층의 핵심인 출발지와 도착지를 알 수 있는 부분을 IP 주소의 형태로 저장합니다.
- 이후 계산의 편리를 위해 payload를 IP Header까지 잘라내어 다음 단계로 넘어갑니다.

Step 4. TCP/UDP Header 제거

```
# ~2byte: Source Port / ~4byte: Destination Port
# HTTP : port 80, DNS : port 53
srcPort = int.from_bytes(packet[:2], byteorder='big')
dstPort = int.from_bytes(packet[2:4], byteorder='big')
```

- TCP와 UDP 헤더 모두 시작하는 4 byte의 정보는 Port 정보를 담고 있습니다.

```
# Structure of TCP Header
# 12~12.5byte: Data offset(TCP header length)
if protocol == 6:
    tcphr_len = packet[12]//16 * 4              # 4byte per 1
    data_packet = packet[tcphr_len:]
```

```
# Structure of UDP Header
# srcPort, dstPort // Length, Checksum
elif protocol == 17:
    udp_len = int.from_bytes(packet[4:6], byteorder='big')          # no use for this project
    data_packet = packet[8:]                                        # length of udp header is 8 byte
```

- TCP 헤더의 경우 IP header와 마찬가지로 Option code를 가지고 있을 수 있으므로 헤더의 길이를 특정하기 위해 길이 정보를 가지고 있는 패킷을 해석하여 Data가 시작하는 부분을 정확히 판별합니다. UDP 헤더의 경우 8 byte로 크기가 고정되어 있습니다.

Step 5. HTTP/DNS 판별

```
if srcPort == 80 or dstPort == 80:
```

```
if srcPort == 53 or dstPort == 53:
```

- 정보가 출발하는 포트와 도착하는 포트의 번호로 HTTP/DNS를 구분합니다.

```
if 'HTTP' in data_packet_str:
```

- port 80의 경우 HTTP가 아닌 데이터도 송수신되는 때도 있어 예외처리를 해주었습니다.

Step 6. Data 처리

“HTTP”

```
GET /emnet/log.php?aidx=9308&url=http%3A%2F%2Fwww.interpark.com%2Fmalls%2Findex.html%3Fsmid1%3Dheader%26smid2%3Dlogo&ref=http%3A%2F%2Fwww
Host: adlog.adinsight.co.kr\r\n
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:80.0) Gecko/20100101 Firefox/80.0\r\n
Accept: image/webp,*/*\r\n
Accept-Language: ko-KR,ko;q=0.8,en-US;q=0.5,en;q=0.3\r\n
Accept-Encoding: gzip, deflate\r\n
Connection: keep-alive\r\n
Referer: http://www.interpark.com/malls/index.html?smid1=header&smid2=logo\r\n\r\n'
```

```
HTTP/1.1 200 OK\r\n
Date: Sat, 26 Sep 2020 08:41:39 GMT\r\n
Server: Apache\r\n
P3P: CP="NON NID PSAa PSDa OUR IND UNI COM NAV STA",policyref="/w3c/p3p.xml"\r\n
Access-Control-Allow-Origin: *\r\n
HN: DE3\r\n
Pragma: no-cache\r\n
Expires: Fri, 30 Oct 1998 14:19:41 GMT\r\n
Cache-Control: no-cache,no-store,private\r\n
Content-Length: 371\r\n
Connection: close\r\n
Content-Type: application/x-javascript\r\n\r\n
document.write (\'<a href="http://addb.interpark.com/RealMedia/ads/click_lx.ads/www.interpar
```

- 정보를 직접 출력하였더니 위와 같이 각각의 항목은 ‘₩r₩n’으로 구분되고 다음 패킷으로 넘어갈 때 ‘₩r₩n’이 다시 등장하는 형태의 자료였습니다. (편의를 위해 ₩r₩n을 기준으로 개행)
- Request/Status Line을 문자열에서 처음으로 등장하는 ‘₩r₩n’을 기준으로 구분하였습니다.

“DNS”

```
b'L\xb4\x81\x80\x00\x01\x00\x01\x00\x01\x00\x01\x05siape\x04veta\x05naver\x03com\x00\x00\x1c
\x00\x01\xc0\x0c\x00\x05\x00\x01\x00\x00\x01\x1d\x00\x1d\x05siape\x04veta\x05naver\x03com
\x05nheos\xc0\x1d\xc0G\x00\x06\x00\x01\x00\x00\x00x\x00(\x04gns1\xc0G\nhostmaster\xc0Gxh)*
\x00\x00*0\x00\x00\x0e\x10\x00\t:\x80\x00\x00\x00\xb4\x00\x00)\x10\x00\x00\x00\x00\x00\x00\x00'
```

- 정보가 위와 같이 바이트 코드로 이루어져 있습니다.

- ID / flags / QDCOUNT / ANCOUNT / NSCOUNT / ARCOUNT 순서대로 각각의 크기에 맞게 잘라 내어 hex 형태로 표현하였습니다.

3-5. Example Cases

- Selection Interface

```
giwon@giwon-virtual-machine:~/Desktop/CSNetwork/pj1$ ./run.sh
1.ens32
2.lo
3.any
4.bluetooth-monitor
5.nflog
6.nfqueue
Enter the device number: 1
Which header do you want to sniff? Enter the number (HTTP = 1, DNS = 2): 1
```

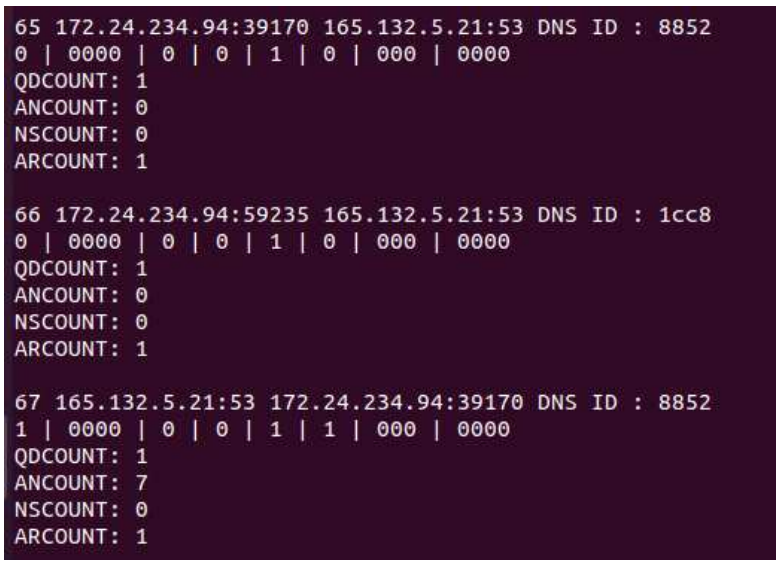
case 1. www.interpark.com

```
1 172.24.239.223:9736 211.233.74.23:80 HTTP Response
GET /malls/index.html HTTP/1.1
Host: www.interpark.com
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/85.0.4183.121 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchan
Referer: http://www.interpark.com/
Accept-Encoding: gzip, deflate
Accept-Language: ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7
Cookie: tourGUID=c1202552-6525-4b11-a61d-2816c508795e; pcid=159902239234989827; _ga=GA1.2.256233484.1599022393; _gac_UA-72644849-6
c_UA-72644849-9=1.1599022393.EAIaIQobChMI4oibxtbJ6wIVQ1RgCh0nJw0dEAAYASAAEgKBSfD_BwE; _gac_UA-86011220-3=1.1599022393.EAIaIQobChMI
93.EAIaIQobChMI4oibxtbJ6wIVQ1RgCh0nJw0dEAAYASAAEgKBSfD_BwE; _gac_UA-93889457-1=1.1599022393.EAIaIQobChMI4oibxtbJ6wIVQ1RgCh0nJw0dEA
5722716; _BS_GUUID=FlyNczVGP84LJIbCXwY6jCaK73z8doLTJjGVSCCH; _TRK_AUIDA_13778=ba00451872efed3a2ac380260cb4efdf:1; _gcl_aw=GCL.1599
318_t_uid=51511313017020792.1599022395328; TR10125305318_t_pa1=3.460.693.0.336d5ebc5436534e61d16e63ddfca327.c9ef9140a577539202456f

2 172.24.239.223:9739 222.231.50.18:80 HTTP Response
GET /static/styles/components/header.css?1601182804723 HTTP/1.1
Host: shopping.interpark.com
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/85.0.4183.121 Safari/537.36
Accept: text/css,*/*;q=0.1
Referer: http://www.interpark.com/
Accept-Encoding: gzip, deflate
Accept-Language: ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7
Cookie: tourGUID=c1202552-6525-4b11-a61d-2816c508795e; pcid=159902239234989827; _ga=GA1.2.256233484.1599022393; _gac_UA-72644849-6
c_UA-72644849-9=1.1599022393.EAIaIQobChMI4oibxtbJ6wIVQ1RgCh0nJw0dEAAYASAAEgKBSfD_BwE; _gac_UA-86011220-3=1.1599022393.EAIaIQobChMI
93.EAIaIQobChMI4oibxtbJ6wIVQ1RgCh0nJw0dEAAYASAAEgKBSfD_BwE; _gac_UA-93889457-1=1.1599022393.EAIaIQobChMI4oibxtbJ6wIVQ1RgCh0nJw0dEA
5722716; _BS_GUUID=FlyNczVGP84LJIbCXwY6jCaK73z8doLTJjGVSCCH; _TRK_AUIDA_13778=ba00451872efed3a2ac380260cb4efdf:1; _gcl_aw=GCL.1599
318_t_uid=51511313017020792.1599022395328; TR10125305318_t_pa1=3.460.693.0.336d5ebc5436534e61d16e63ddfca327.c9ef9140a577539202456f
IT15DAlTs; captcha_text=ESEr2EDRdp96J2WjAq5QKg%3D%3D; SAVEID=N; imfs_pcid=01569622850468748 6; __utma=41825914.256233484.1599022393
tmccn=(direct)|utmcmd=(none); sptc=46; interparkID=%2B717%2BcVVhxy7tK%2BafVVoeA%3D%3D; wcs_bt=s_22aec0678481:1601097422

3 172.24.239.223:9741 222.231.50.18:80 HTTP Response
GET /static/styles/pages/powerLink.css?v=2.1 HTTP/1.1
Host: shopping.interpark.com
Connection: keep-alive
```

case 2. www.youtube.com

```
30 172.217.24.35:80 172.24.234.94:60054 HTTP Request
HTTP/1.1 200 OK
Content-Type: application/ocsp-response
Date: Sun, 27 Sep 2020 05:45:07 GMT
Cache-Control: public, max-age=86400
Server: ocsp_responder
Content-Length: 472
X-XSS-Protection: 0
X-Frame-Options: SAMEORIGIN

31 172.217.24.35:80 172.24.234.94:60082 HTTP Request
HTTP/1.1 200 OK
Content-Type: application/ocsp-response
Date: Sun, 27 Sep 2020 05:45:07 GMT
Cache-Control: public, max-age=86400
Server: ocsp_responder
Content-Length: 472
X-XSS-Protection: 0
X-Frame-Options: SAMEORIGIN
```

case 3. www.naver.com



4. Reference

iPerf3 - https://iperf.fr/iperf-doc.php#3doc
cwnd/rwnd - https://blog.stackpath.com/glossary-cwnd-and-rwnd/
Pcapy library - https://rawgit.com/CoreSecurity/pcapy/master/pcapy.html#idp1073147198592
Network Packet Structure - https://karfn84.tistory.com/entry/network-%ED%8C%A8%ED%82%B7-%EA%B5%AC%EC%A1%B0-ether-tcp-ip-header
https://hack-cracker.tistory.com/112
https://kskang.tistory.com/476
HTTP protocol - https://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html
DNS packet - https://darksoulstory.tistory.com/62?category=511291
TCP/IP 5 layer Model - https://zion830.tistory.com/104