

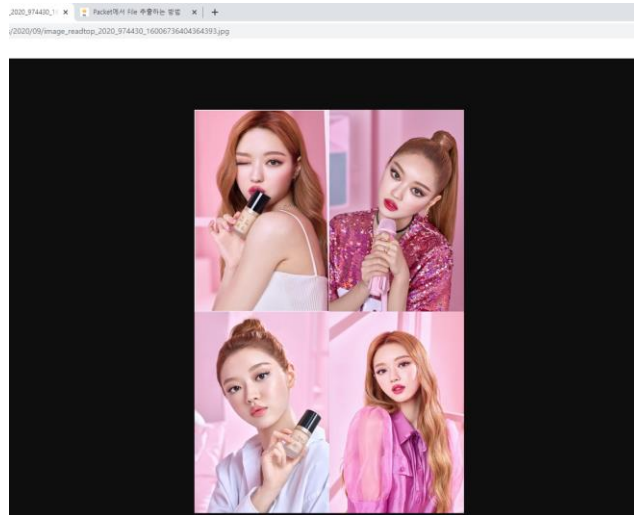
Assignment 1

2016147030 송민석

1-1.

다음 Web page의 이미지를 패킷 분석을 통해 Extract하고자 한다.

URL : http://file.mk.co.kr/meet/neds/2020/09/image_readtop_2020_974430_16006736404364393.jpg



WiFi 네트워크를 이용하고 있었으므로, Wireshark 를 실행한 후 WiFi 네트워크의 패킷 캡처를 시작한다. 이후 브라우저를 통해 해당 이미지의 URL 에 접속하여 Request / Response 를 Wireshark 가 기록하도록 한다.

Export Objects - HTTP 를 통해 해당 이미지의 HTTP Request/Response 의 Packet No. 를 확인한다.

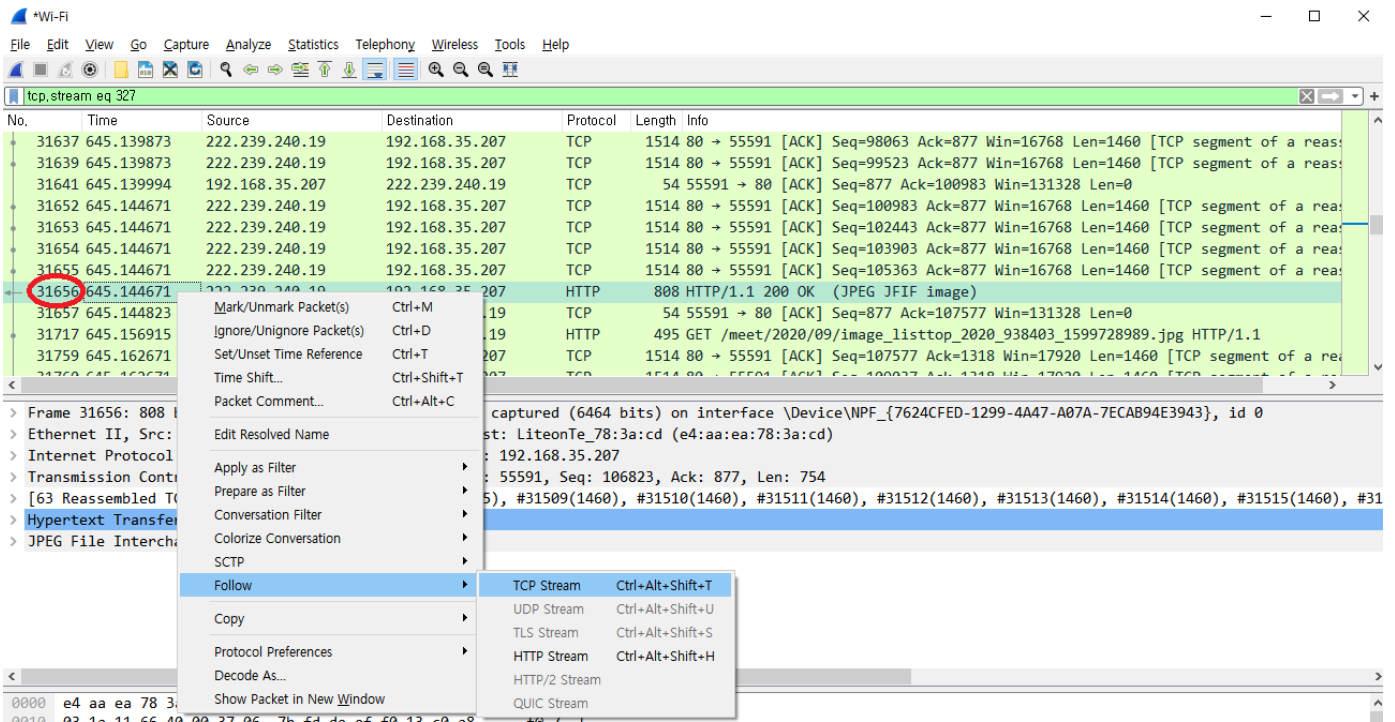
Wireshark - Export - HTTP object list

Packet	Hostname	Content Type	Size	Filename
33227	api.dable.io	text/html	24 kB	45462140.1591861066113?from=http%3A%2F%2Fmksp...
31656	file.mk.co.kr	image/jpeg	88 kB	image_readtop_2020_974430_16006736404364393.jpg
30956	mkspports.co.kr	text/html	72 kB	974430
32471	www.mk.co.kr	text/html	275 bytes	inc_news_cnt.php?oc=MKSP900317&year=2020&no=97...

Text Filter: 974430

Buttons: Save, Save All, Close, Help

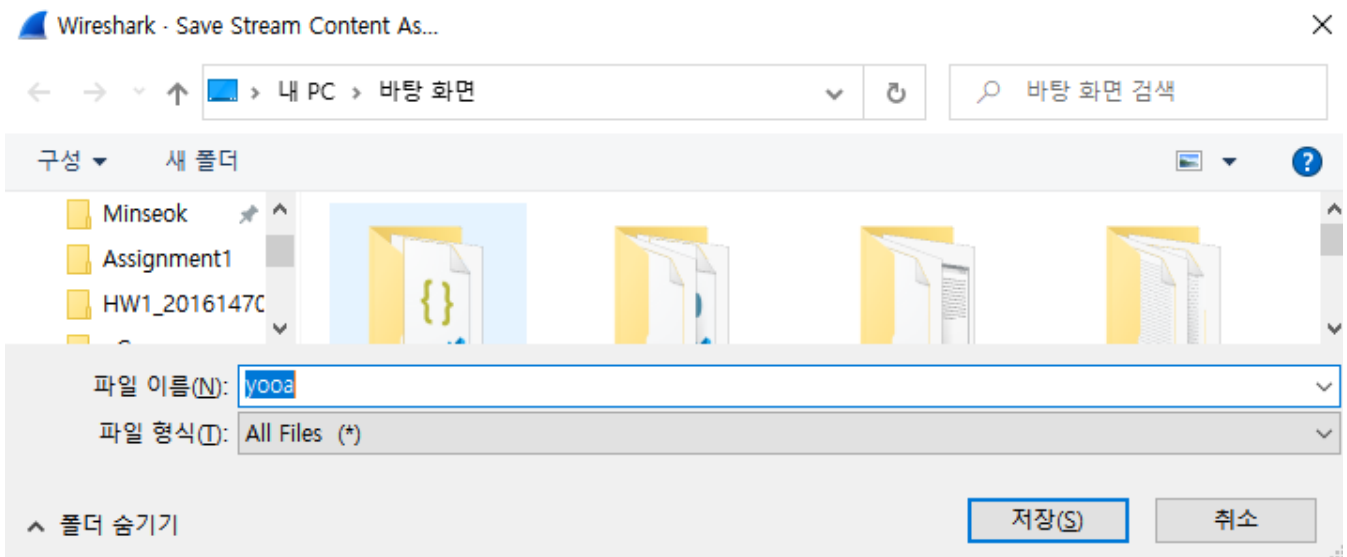
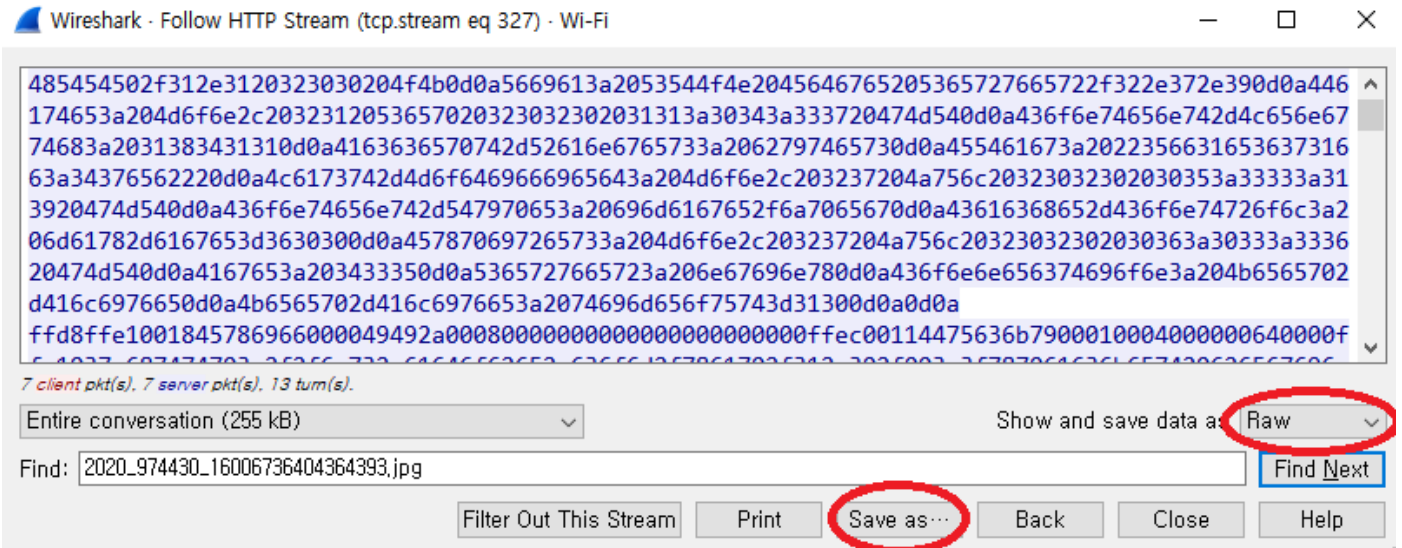
31656 번 패킷을 찾아 Response 의 내용을 추적한다.



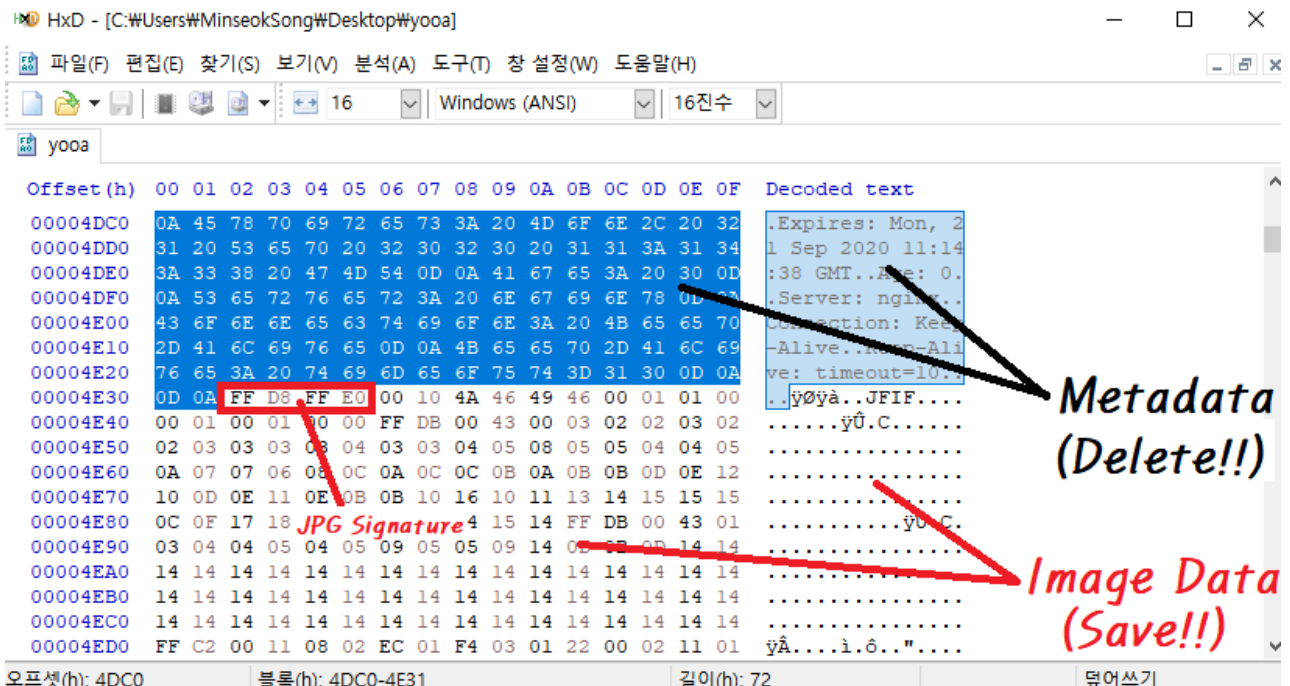
패킷 내부에서 해당 이미지의 Request / Response 를 찾아낸다.



해당 내용이 ASCII Code 형태로 되어 있어 확인하기 힘드므로, 이를 Raw format(16 진수 형태)로 저장하였다.

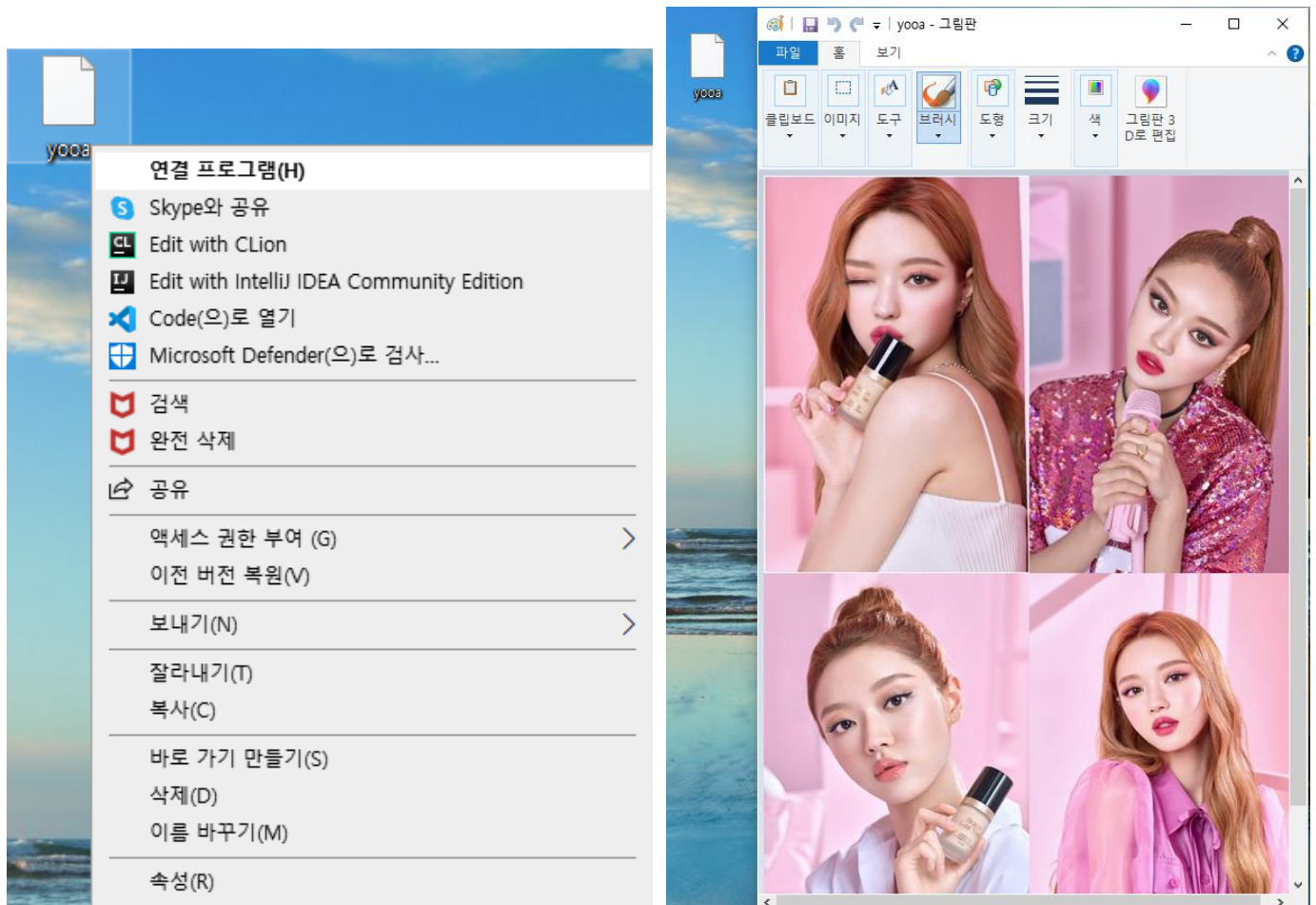


이후, 16 진수 파일을 decode 할 수 있는 Tool 인 HxD 를 이용하여 전체 Response 에서 이미지 데이터만 저장한다. JPG 파일의 Signature 는 16 진수 FF D8 FF E0 이므로 이를 기준으로 위 부분, 즉 실제 이미지 데이터가 아닌 부분(Status Code, Date 등)을 삭제하여 이미지 데이터만 남겨 저장한다.





이후 저장된 파일을 그림판, 사진 등의 jpg 호환 프로그램으로 열면 해당 이미지가 저장되었음을 확인할 수 있다.



1-2.

Iperf3은 두 컴퓨터(서버 컴퓨터, 클라이언트 컴퓨터)사이의 네트워크 속도를 측정하는 tool이다. 기본으로 TCP 프로토콜을 사용하며, -u 옵션을 사용하여 UDP 프로토콜을 이용하여 측정할 수 있다.

Iperf3을 내려 받은 경로에서 실행한다. 먼저 -s 옵션을 이용하여 server mode부터 실행한다.

```
명령 프롬프트 - iperf3 -s -V
C:\Users\MinseokSong\Desktop\Minseok\Study\School\3-2\컴퓨터네트워크\Assignment1\iperf-3.1.3-win64>iperf3 -s -V
iperf 3.1.3
CYGWIN_NT-10.0 LAPTOP-P1JRT9BJ 2.5.1(0.297/5/3) 2016-04-21 22:14 x86_64
-----
Server listening on 5201
-----
Time: Sun, 27 Sep 2020 10:57:19 GMT
Accepted connection from ::1, port 51271
Cookie: LAPTOP-P1JRT9BJ.1601204239.261751.79
TCP MSS: 0 (default)
[ 5] local ::1 port 5201 connected to ::1 port 51272
Starting Test: protocol: TCP, 1 streams, 131072 byte blocks, omitting 0 seconds, 10 second test
[ ID] Interval      Transfer    Bandwidth
[ 5] 0.00-1.00    sec  698 MBytes  5.85 Gbits/sec
[ 5] 1.00-2.00    sec  1.29 GBytes 11.1 Gbits/sec
[ 5] 2.00-3.00    sec  1.11 GBytes 9.56 Gbits/sec
[ 5] 3.00-4.00    sec  702 MBytes  5.89 Gbits/sec
[ 5] 4.00-5.00    sec  1.23 GBytes 10.5 Gbits/sec
[ 5] 5.00-6.00    sec  1.51 GBytes 13.0 Gbits/sec
[ 5] 6.00-7.00    sec  943 MBytes  7.91 Gbits/sec
[ 5] 7.00-8.00    sec  1.28 GBytes 11.0 Gbits/sec
[ 5] 8.00-9.00    sec  1.15 GBytes 9.89 Gbits/sec
[ 5] 9.00-10.00   sec  309 MBytes  2.59 Gbits/sec
[ 5] 10.00-10.00  sec   0.00 Bytes  0.00 bits/sec
-----
Test Complete. Summary Results:
[ ID] Interval      Transfer    Bandwidth
[ 5] 0.00-10.00   sec  10.2 GBytes  8.72 Gbits/sec
[ 5] 0.00-10.00   sec   0.00 Bytes  0.00 bits/sec
CPU Utilization: local/receiver 37.6% (7.3%/30.3%), remote/sender 0.0% (0.0%/0.0%)
iperf 3.1.3
CYGWIN_NT-10.0 LAPTOP-P1JRT9BJ 2.5.1(0.297/5/3) 2016-04-21 22:14 x86_64
-----
Server listening on 5201
-----
```

이후 새로운 콘솔에서 -c 옵션을 이용하여 client mode도 실행한다. 연결할 서버의 ip주소 또한 입력한다. 현재 서버 호스트가 localhost이므로 localhost를 입력한다.

```
명령 프롬프트
C:\Users\MinseokSong\Desktop\Minseok\Study\School\3-2\컴퓨터네트워크\Assignment1\iperf-3.1.3-win64>iperf3 -c localhost -V
iperf 3.1.3
CYGWIN_NT-10.0 LAPTOP-P1JRT9BJ 2.5.1(0.297/5/3) 2016-04-21 22:14 x86_64
Time: Sun, 27 Sep 2020 10:57:19 GMT
Connecting to host localhost, port 5201
Cookie: LAPTOP-P1JRT9BJ.1601204239.261751.79
TCP MSS: 0 (default)
[ 4] local ::1 port 51272 connected to ::1 port 5201
Starting Test: protocol: TCP, 1 streams, 131072 byte blocks, omitting 0 seconds, 10 second test
[ ID] Interval      Transfer    Bandwidth
[ 4] 0.00-1.00    sec  699 MBytes  5.86 Gbits/sec
[ 4] 1.00-2.00    sec  1.29 GBytes 11.1 Gbits/sec
[ 4] 2.00-3.00    sec  1.11 GBytes 9.56 Gbits/sec
[ 4] 3.00-4.00    sec  701 MBytes  5.88 Gbits/sec
[ 4] 4.00-5.00    sec  1.23 GBytes 10.5 Gbits/sec
[ 4] 5.00-6.00    sec  1.51 GBytes 13.0 Gbits/sec
[ 4] 6.00-7.00    sec  942 MBytes  7.91 Gbits/sec
[ 4] 7.00-8.02    sec  1.28 GBytes 10.8 Gbits/sec
[ 4] 8.02-9.01    sec  1.15 GBytes 10.0 Gbits/sec
[ 4] 9.01-10.00   sec  309 MBytes  2.60 Gbits/sec
-----
Test Complete. Summary Results:
[ ID] Interval      Transfer    Bandwidth
[ 4] 0.00-10.00   sec  10.2 GBytes  8.72 Gbits/sec
[ 4] 0.00-10.00   sec  10.2 GBytes  8.72 Gbits/sec
CPU Utilization: local/sender 56.5% (11.7%/44.9%), remote/receiver 37.6% (7.3%/30.3%)
iperf Done.
C:\Users\MinseokSong\Desktop\Minseok\Study\School\3-2\컴퓨터네트워크\Assignment1\iperf-3.1.3-win64>
```

두 명령어 모두 -V 옵션을 추가로 이용하여 보다 상세한 정보를 출력하도록 하였다.

현재 server 와 client 사이 네트워크의 Trasfer, Bandwidth 정보를 10 초간 1 초 간격 Interval 에 따라 출력한다.

Transfer : 해당 Interval 동안 전송한 데이터 양

Bitrate : 초당 비트 전송률 → Bandwidth : 네트워크의 대역폭(bitrate 단위로 표시)

Cwnd(Congestion Window) : sender 측에서 1 RTT 당 전송되는 packet 의 크기. default 값은 OS 마다 다르다.

또한 -b, -n, -w, -l 4 가지 옵션의 각 기능은 다음과 같다.

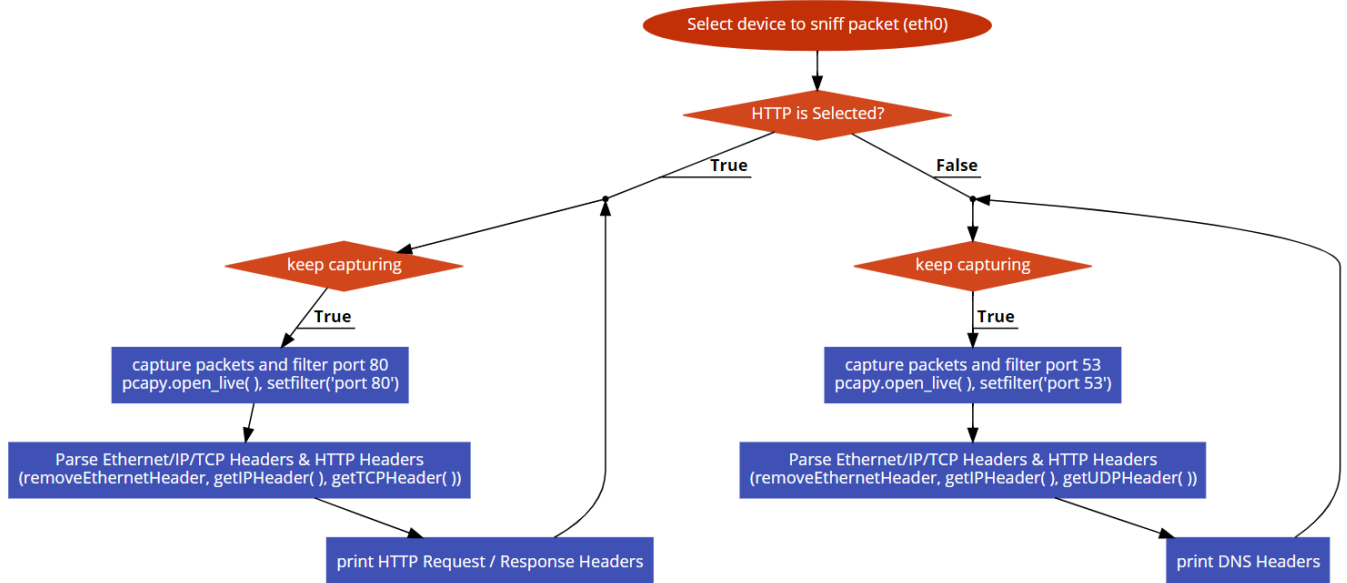
-b (n)	대역폭을 n bits/sec 으로 설정한다. UDP 프로토콜을 사용할 경우 기본값은 1Mbit/sec 이고, TCP 프로토콜을 사용할 경우 제한되어있지 않다. Iperf3 은 기본값으로 TCP 프로토콜을 사용하여 네트워크 속도를 측정한다.
-w (n)	window size / socket buffer size 를 n 으로 설정한다. TCP 프로토콜을 사용할 경우 TCP window size 로 설정된다. 설정된 n 은 server 에서도 적용된다.
-l (n)	Read / Write 할 buffer 의 size 를 n 으로 설정한다. Iperf 는 이 n 개 byte 의 배열을 여러 번 사용함으로써 동작한다. TCP 프로토콜을 사용할 경우 기본값은 128KB 이고, UDP 의 경우 8KB 이다.
-n (m)	Buffer 를 m bytes 만큼 전송하도록 설정한다. 기본적으로 Iperf 는 10 초간 전송하는데 이 m 이 10 초를 넘어가도록 설정되면 10 초를 무시하고 m bytes 를 전송한다.

1-3.

과제에 사용된 언어 및 버전, OS, 라이브러리는 다음과 같다.

Code Language : Python | Version : 3.6.9 | OS : Ubuntu 18.04 | Library : pcapy

project.py 의 Flowchart 는 다음과 같다.



먼저 `findalldevs()`를 이용하여 접근 가능한 네트워크 인터페이스 카드 목록을 출력하고, 사용자가 선택하도록 한다. 이후 사용자는 HTTP 패킷을 sniff 할 것인지 DNS 패킷을 sniff 할 것인지 선택하게 된다. 선택된 값에 따라 프로그램이 종료될 때까지 패킷을 캡처하여 Ethernet Header, IP Header, TCP/UDP Header 에서 필요한 값을 parsing 한 후 남은 부분을 형식에 맞게 print 한다.

각 단계에 사용된 코드는 다음과 같다.

```
project.py X
project.py > ...
1  # import Libraries(Only pcapy used)
2  import pcapy
3
4  devices = pcapy.findalldevs()
5
6  # Ask user to choose the device name to sniff
7  print ("Available Network Interface Cards are\n ", devices)
8  dev = int(input("\nSelect Network Device (1 ~ {}) : ".format(len(devices))))-1
9  dev = devices[dev]
10 print ("\n", dev, "has Selected")
11
12 # Ask user to choose HTTP mode / DNS mode
13 tp = input("\nSelect HTTP/DNS (1: HTTP / 2: DNS) : ")
14 print()
```

라이브러리를 import 하고, 사용자에게 device 목록을 보여준 뒤 device 와 HTTP/DNS 를 선택하도록 한다.

```
project.py X
project.py > ...

158 # if User selects 1, call 'capture_HTTP' / if user selects 2, call 'capture_DNS'
159 if (tp == '1'):
160     capture_HTTP(dev)
161 elif (tp == '2'):
162     capture_DNS(dev)
```

사용자가 선택한 값에 따라 HTTP 헤더를 출력하는 함수, DNS 헤더를 출력하는 함수를 실행한다.

```
project.py X
project.py > capture_HTTP

67 # print HTTP Request/Response Header values using 'getXXXHeader' functions
68 def capture_HTTP(device_type):
69
70     # Capture Packets Live & Filter HTTP
71     cap = pcap.open_live(device_type, 65536, 1, 0)
72     cap.setfilter('port 80')
73
74     counter = 0
75
76     while(1):
77         # Capture next packet
78         (header, packet) = cap.next()
79
80         # Get IP Header Informations
81         IPHeader = getIPHeader(packet)
82         s_ip_addr, d_ip_addr= IPHeader['S_IP_Addr'], IPHeader['D_IP_Addr']
83
84         # Get TCP Header Informations
85         TCPHeader_startIndex = 14 + IPHeader['IP_HL']
86         TCPHeader = getTCPHeader(packet, TCPHeader_startIndex)
87         s_port, d_port= TCPHeader['S_Port'], TCPHeader['D_Port']
88
89         # Decode Remaining Packets
90         header_exitIndex = TCPHeader_startIndex + TCPHeader['Offset']
91         packet_without_header = packet[header_exitIndex:]
92         packet_decoded = packet_without_header.decode('ISO-8859-1') # 'ISO-8859-1'
93
94         # Check if keyword 'HTTP' contained
95         if ('HTTP' in packet_decoded):
96
97             # Check if packet includes Request Header, and print if included
98             if ('GET' in packet_decoded) | ('POST' in packet_decoded) | ('PUT' in packet_decoded) | ('DELETE' in packet_decoded) | ('HEAD' in packet_decoded):
99                 counter+=1
100                 print("{} {}: {} HTTP Request".format(counter, s_ip_addr, s_port, d_ip_addr, d_port))
101                 index = packet_decoded.index('\r\n\r\n')
102                 print(packet_decoded[:index] + '\n')
103
104             # Check if packet includes Response Header, and print if included
105             else:
106                 counter+=1
107                 print("{} {}: {} HTTP Response".format(counter, s_ip_addr, s_port, d_ip_addr, d_port))
108                 index = packet_decoded.index('\r\n\r\n')
109                 print(packet_decoded[:index] + '\n')
```

HTTP 헤더를 출력하는 경우 위 함수가 실행된다. 먼저 while 문과 pcap.open_live()를 이용해 프로그램이 실행되는 동안 계속 패킷을 캡처하도록 하고, setfilter 를 이용하여 80 번 포트, 즉 HTTP 프로토콜만 필터링하도록 한다. 각 출력마다 번호를 매기기 위해 counter 변수에 선언하고 초기값을 0 으로 할당한다. 이후 next()를 이용해 패킷에 접근하고, Ethernet Header, IP Header, TCP Header 를 순서대로 parsing 하여 필요한 정보(S_IP, D_IP 등)을 추출한다. 각 parsing 을 위해 별도의 함수를 정의하였으며 이는 뒤에 서술하였다. Ethernet Header 의 경우 IP Header 를 Parsing 하는 과정에서 같이 parse 된다. 각 단계별 Header 를 추출한 후, 패킷의 남은 부분을 decode 하여 HTTP Request / Response 를 구분하고 헤더 부분을 양식에 맞게 출력한다.


```
project.py X
project.py > getUDPHeader

38 # Parsing UDP Header's informations from given packet
39 def getUDPHeader(packet, start_index):
40     UDPHeader = {}
41     UDPHeader['S_Port'] = packet[start_index]*256+packet[start_index+1]
42     UDPHeader['D_Port'] = packet[start_index+2]*256+packet[start_index+3]
43     return UDPHeader
44
```

UDP Header 를 parsing 하는 getUDPHeader. 역시 IP Header 의 길이를 넘겨받아 parsing 한다. DNS 의 경우에만 작동한다.

```
project.py X
project.py > getUDPHeader

46
47 # Parsing DNS Header's informations from given packet
48 def getDNSHeader(packet_without_header):
49     DNSHeader = {}
50     DNSHeader['ID'] = hex(packet_without_header[0]*256+packet_without_header[1])[2:]
51     DNSHeader['QR'] = bin(packet_without_header[2])[2:].zfill(8)[0]
52     DNSHeader['Opcode'] = bin(packet_without_header[2])[2:].zfill(8)[1:5]
53     DNSHeader['AA'] = bin(packet_without_header[2])[2:].zfill(8)[5]
54     DNSHeader['TC'] = bin(packet_without_header[2])[2:].zfill(8)[6]
55     DNSHeader['RD'] = bin(packet_without_header[2])[2:].zfill(8)[7]
56     DNSHeader['RA'] = bin(packet_without_header[3])[2:].zfill(8)[0]
57     DNSHeader['Z'] = bin(packet_without_header[3])[2:].zfill(8)[1]
58     DNSHeader['Rcode'] = bin(packet_without_header[3])[2:].zfill(8)[4:]
59     DNSHeader['QDCOUNT'] = packet_without_header[4]*256+packet_without_header[5]
60     DNSHeader['ANCOUNT'] = packet_without_header[6]*256+packet_without_header[7]
61     DNSHeader['NSCOUNT'] = packet_without_header[8]*256+packet_without_header[9]
62     DNSHeader['ARCOUNT'] = packet_without_header[10]*256+packet_without_header[11]
63     return DNSHeader
64
```

DNS Header 를 parsing 하는 getDNSHeader. Pcapy 를 통해 캡처된 Packet 은 byte 단위로 구성된 데에 반해 각 정보는 bit 단위로 담겨있어 각각 bin, hex 를 이용하여 변환해주었다.

run.sh 와 setup.sh 의 내용은 다음과 같다.

```
miniseok@LAPTOP-P1JRT9BJ: /1 x + v - □ x
#!/bin/bash
sudo python3 project.py
~
~
~
~
~
"run.sh" 2L, 37C 2,23 All |
```

run.sh 는 해당 project.py 파일을 실행한다.

```
miniseok@LAPTOP-P1JRT9BJ: /1 x + v - □ x
#!/bin/bash
sudo apt-get install python-pcap
|
~
~
~
~
"setup.sh" 5L, 50C 5,0-1 All |
```

setup.sh 는 실행에 필요한 pcap 라이브러리를 설치한다.

./setup.sh && ./run.sh 명령을 수행한 후, eth0, HTTP 를 선택한 뒤 3 가지 url 에 wget 명령어로 접근함으로써 수행한 결과는 다음과 같다.

```
miniseok@LAPTOP-P1JRT9BJ: /1 x + v - □ x miniseok@LAPTOP-P1JRT9BJ: /1 x + v - □ x
miniseok@LAPTOP-P1JRT9BJ: /mnt/c/Users/MiniseokSong/Desktop/Linux$ ./setup.sh && ./run.sh miniseok@LAPTOP-P1JRT9BJ: /mnt/c/Users/MiniseokSong/Desktop/Linux$ wget http://gen.lib.rus.ec
Building dependency tree Done
Reading package lists... Done
Building dependency tree Done
Reading state information... Done
python3-pcap is already the newest version (0.38.0-1build1).
0 upgraded, 0 newly installed, 0 to remove and 129 not upgraded.
Available Network Interface Cards are
['enx8086031b0000', 'lo', 'veth', 'veth-peer']
Select Network Device (1 - 5) : 1
eth0 has Selected
Select HTTP/HTTPS (1: HTTP / 2: HTTPS) : 1
1 172.18.245.14:36962 185.222.202.19:80 HTTP Request
GET / HTTP/1.1
User-Agent: wget/1.19.4 (linux-gnu)
Accept: */*
Accept-Encoding: identity
Host: gen.lib.rus.ec
Connection: Keep-Alive
2 185.222.202.19:80 172.18.245.14:36962 HTTP Response
HTTP/1.1 200 OK
Server: nginx
Date: Sun, 27 Sep 2020 13:33:32 GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive
Set-Cookie: lg_topic=lgigen; expires=Thu, 01-Oct-2020 17:33:32 GMT; Max-Age=360000
3 172.18.245.14:46638 211.110.63.99:80 HTTP Request
GET /meet/needs/2020/09/image_readtop_2020_974430_16086736404364393.jpg HTTP/1.1
User-Agent: wget/1.19.4 (linux-gnu)
Accept: */*
Accept-Encoding: identity
Host: file.mk.co.kr
Connection: Keep-Alive
4 211.110.63.99:80 172.18.245.14:46638 HTTP Response
HTTP/1.1 200 OK
Via: STON Edge Server/7.9
Date: Sun, 27 Sep 2020 13:33:35 GMT
Content-Length: 80434
Accept-Ranges: bytes
ETag: "56d6d8e1c1972"
Last-Modified: Mon, 21 Sep 2020 08:35:10 GMT
Content-Type: image/jpeg
Cache-Control: max-age=600
Expires: Thu, 24 Sep 2020 10:16:24 GMT
Age: 6
Server: nginx
Connection: Keep-Alive
Keep-Alive: timeout=60
5 172.18.245.14:39960 165.132.16.123:80 HTTP Request
GET / HTTP/1.1
User-Agent: wget/1.19.4 (linux-gnu)
Accept: */*
Accept-Encoding: identity
Host: ysecc.yonsei.ac.kr
Connection: Keep-Alive
6 165.132.16.123:80 172.18.245.14:39960 HTTP Response
HTTP/1.1 200 Found
Date: Sun, 27 Sep 2020 13:33:41 GMT
Server: Apache
Location: https://ysecc.yonsei.ac.kr/
Content-Length: 211
Keep-Alive: timeout=60, max=2048
Connection: Keep-Alive
Content-Type: text/html; charset=iso-8859-1
miniseok@LAPTOP-P1JRT9BJ: /mnt/c/Users/MiniseokSong/Desktop/Linux$ wget http://gen.lib.rus.ec
--2020-09-27 22:33:30-- http://gen.lib.rus.ec/
Resolving gen.lib.rus.ec (gen.lib.rus.ec)... 185.222.202.19, 198.167.223.167, 216.218.130.2, ...
Connecting to gen.lib.rus.ec (gen.lib.rus.ec):185.222.202.19:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [text/html]
Saving to: 'index.html.63'
index.html.63 [====>] 0 --KB/s
index.html.63 [====>] 45.04K 70.2Kb/s in 0.6s
2020-09-27 22:33:32 (70.2 KB/s) - 'index.html.63' saved [46123]
miniseok@LAPTOP-P1JRT9BJ: /mnt/c/Users/MiniseokSong/Desktop/Linux$ wget http://file.mk.co.kr/meet/needs/2020/09/image_readtop_2020_974430_16086736404364393.jpg
--2020-09-27 22:33:34-- http://file.mk.co.kr/meet/needs/2020/09/image_readtop_2020_974430_16086736404364393.jpg
Resolving file.mk.co.kr (file.mk.co.kr)... 211.110.63.99, 222.239.240.19, 211.110.211.253, ...
Connecting to file.mk.co.kr (file.mk.co.kr):211.110.63.99:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 80434 (80K) [image/jpeg]
Saving to: 'image_readtop_2020_974430_16086736404364393.jpg.18'
image_readtop_2020_974430_16086736404364393.jp 100%[====>] 86.36K --KB/s in 0.06s
2020-09-27 22:33:34 (1.47 MB/s) - 'image_readtop_2020_974430_16086736404364393.jpg.18' saved [80434/80434]
miniseok@LAPTOP-P1JRT9BJ: /mnt/c/Users/MiniseokSong/Desktop/Linux$ wget http://ysecc.yonsei.ac.kr/
--2020-09-27 22:33:39-- http://ysecc.yonsei.ac.kr/
Resolving ysecc.yonsei.ac.kr (ysecc.yonsei.ac.kr)... 165.132.16.123, 165.132.10.21, 165.132.5.21, ...
Connecting to ysecc.yonsei.ac.kr (ysecc.yonsei.ac.kr):165.132.16.123:80... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://ysecc.yonsei.ac.kr/[following]
--2020-09-27 22:33:39-- https://ysecc.yonsei.ac.kr/
Connecting to ysecc.yonsei.ac.kr (ysecc.yonsei.ac.kr):165.132.16.123:443... connected.
HTTP request sent, awaiting response... 303 See Other
Location: https://ysecc.yonsei.ac.kr/login/index.php [following]
--2020-09-27 22:33:40-- https://ysecc.yonsei.ac.kr/login/index.php
Reusing existing connection to ysecc.yonsei.ac.kr:443.
HTTP request sent, awaiting response... 200 OK
Length: 836 [text/html]
Saving to: 'index.html.64'
index.html.64 100%[====>] 836 --KB/s in 0s
2020-09-27 22:33:40 (13.6 MB/s) - 'index.html.64' saved [836/836]
miniseok@LAPTOP-P1JRT9BJ: /mnt/c/Users/MiniseokSong/Desktop/Linux$
```

같은 방법으로 실행한 후 DNS를 선택하여 실행한 과정은 다음과 같다.

```
miniseok@LAPTOP-P1JRT9B3: / $
miniseok@LAPTOP-P1JRT9B3: /mnt/c/Users/MinseokSong/Desktop/Linux$ ./setup.sh 66 ./run.sh
Reading package lists... Done
Building dependency tree
Reading state information... Done
python3-pyca is already the newest version (0.10.0-1build1).
0 upgraded, 0 newly installed, 0 to remove and 139 not upgraded.
Available Network Interface Cards are
['eth0', 'any', 'lo', 'mlog', 'mqueue']

Select Network Device (1 - 5) : 1

eth0 has Selected

Select HTTP/DNS (1: HTTP / 2: DNS) : 2

0 172.18.245.14:51676 172.18.240.1:53 DNS ID : 2697
0 | 0000 | 0 | 0 | 1 | 0 | 0 | 0000
QCOUNT:1
ANSICOUNT:0
NSICOUNT:0
ARICOUNT:0

1 172.18.245.14:51676 172.18.240.1:53 DNS ID : 76a1
0 | 0000 | 0 | 0 | 1 | 0 | 0 | 0000
QCOUNT:1
ANSICOUNT:0
NSICOUNT:0
ARICOUNT:0

3 172.18.240.1:53 172.18.240.14:51676 DNS ID : 2697
1 | 0000 | 0 | 0 | 1 | 0 | 0 | 0000
QCOUNT:1
ANSICOUNT:13
NSICOUNT:0
ARICOUNT:0

3 172.18.240.1:53 172.18.240.14:51676 DNS ID : 76a1
1 | 0000 | 0 | 0 | 1 | 0 | 0 | 0000
QCOUNT:1
ANSICOUNT:1
NSICOUNT:0
ARICOUNT:0

4 172.18.245.14:59983 172.18.240.1:53 DNS ID : 12aa
0 | 0000 | 0 | 0 | 1 | 0 | 0 | 0000
QCOUNT:1
ANSICOUNT:0
NSICOUNT:0
ARICOUNT:0

5 172.18.245.14:59983 172.18.240.1:53 DNS ID : 340b
0 | 0000 | 0 | 0 | 1 | 0 | 0 | 0000
QCOUNT:1
ANSICOUNT:0
NSICOUNT:0
ARICOUNT:0

6 172.18.240.1:53 172.18.240.14:59983 DNS ID : 12aa
1 | 0000 | 0 | 0 | 1 | 0 | 0 | 0000
QCOUNT:1
ANSICOUNT:16
NSICOUNT:0
ARICOUNT:0

7 172.18.240.1:53 172.18.240.14:59983 DNS ID : 340b
1 | 0000 | 0 | 0 | 1 | 0 | 0 | 0000
QCOUNT:1
ANSICOUNT:1
NSICOUNT:0
ARICOUNT:0

miniseok@LAPTOP-P1JRT9B3: /mnt/c/Users/MinseokSong/Desktop/Linux$ wget http://gen.lib.rus.ec
--2020-09-27 22:40:21-- http://gen.lib.rus.ec/
Resolving gen.lib.rus.ec (gen.lib.rus.ec)... 198.167.223.167, 185.222.202.19, 216.218.130.2, ...
Connecting to gen.lib.rus.ec (gen.lib.rus.ec):[198.167.223.167]:80... connected.
HTTP request sent, awaiting response... [4290 OK
Length: unspecified [text/html]
Saving to: 'index.html.67'

index.html.67 [ <=> 11.81K 45.1kB/s
index.html.67 [ 45.04K 19.4kB/s in 2.3s ^

2020-09-27 22:40:26 (19.4 kB/s) - 'index.html.67' saved [46123]

miniseok@LAPTOP-P1JRT9B3: /mnt/c/Users/MinseokSong/Desktop/Linux$ wget http://file.mk.co.kr/meet/nds/2020/09/image_readtop_2020_974430_16086736404364393.jpg
--2020-09-27 22:40:28-- http://file.mk.co.kr/meet/nds/2020/09/image_readtop_2020_974430_16086736404364393.jpg
Resolving file.mk.co.kr (file.mk.co.kr)... 58.228.245.67, 222.239.240.19, 211.110.211.253, ...
Connecting to file.mk.co.kr (file.mk.co.kr):[58.228.245.67]:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 88434 (86K) [image/jpeg]
Saving to: 'image_readtop_2020_974430_16086736404364393.jpg.20'

image_readtop_2020_974430_16086736404364393.jpg 100%[=====] 86.36K --.-kB/s in 0.04s

2020-09-27 22:40:28 (2.08 MB/s) - 'image_readtop_2020_974430_16086736404364393.jpg.20' saved [88434/88434]

miniseok@LAPTOP-P1JRT9B3: /mnt/c/Users/MinseokSong/Desktop/Linux$
```

```
minseok@LAPTOP-P1JRT9BJ: / ~$
8 172.18.245.14:57238 172.18.240.1:53 DNS ID : 59fb
0 | 0000 | 0 | 0 | 1 | 0 | 0 | 0000
QDCOUNT:1
ANCOUNT:0
NSCOUNT:0
ARCOUNT:0

9 172.18.245.14:57238 172.18.240.1:53 DNS ID : 130b
0 | 0000 | 0 | 0 | 1 | 0 | 0 | 0000
QDCOUNT:1
ANCOUNT:0
NSCOUNT:0
ARCOUNT:0

10 172.18.240.1:53 172.18.245.14:57238 DNS ID : 59fb
1 | 0000 | 0 | 0 | 1 | 0 | 0 | 0000
QDCOUNT:1
ANCOUNT:5
NSCOUNT:0
ARCOUNT:0

11 172.18.240.1:53 172.18.245.14:57238 DNS ID : 130b
1 | 0000 | 0 | 0 | 1 | 1 | 0 | 0000
QDCOUNT:1
ANCOUNT:0
NSCOUNT:1
ARCOUNT:0

minseok@LAPTOP-P1JRT9BJ: / ~$
minseok@LAPTOP-P1JRT9BJ: /mnt/c/Users/MinseokSong/Desktop/Linux$ wget http://yscec.yonsei.ac.kr/
--2020-09-27 22:40:53-- http://yscec.yonsei.ac.kr/
Resolving yscec.yonsei.ac.kr (yscec.yonsei.ac.kr)... 165.132.16.123, 165.132.10.21, 165.132.5.21, ...
Connecting to yscec.yonsei.ac.kr (yscec.yonsei.ac.kr)[165.132.16.123]:80... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://yscec.yonsei.ac.kr/ [following]
--2020-09-27 22:40:53-- https://yscec.yonsei.ac.kr/
Connecting to yscec.yonsei.ac.kr (yscec.yonsei.ac.kr)[165.132.16.123]:443... connected.
HTTP request sent, awaiting response... 303 See Other
Location: https://yscec.yonsei.ac.kr/login/index.php [following]
--2020-09-27 22:40:53-- https://yscec.yonsei.ac.kr/login/index.php
Reusing existing connection to yscec.yonsei.ac.kr:443.
HTTP request sent, awaiting response... 200 OK
Length: 836 [text/html]
Saving to: 'index.html.68'

index.html.68                                     100%[=====]

2020-09-27 22:40:54 (13.5 MB/s) - 'index.html.68' saved [836/836]

minseok@LAPTOP-P1JRT9BJ: /mnt/c/Users/MinseokSong/Desktop/Linux$
minseok@LAPTOP-P1JRT9BJ: /mnt/c/Users/MinseokSong/Desktop/Linux$
minseok@LAPTOP-P1JRT9BJ: /mnt/c/Users/MinseokSong/Desktop/Linux$
minseok@LAPTOP-P1JRT9BJ: /mnt/c/Users/MinseokSong/Desktop/Linux$
```

<http://gen.lib.rus.ec/>,
http://file.mk.co.kr/meet/neds/2020/09/image_readtop_2020_974430_16006736404364393.jpg,
<http://yscec.yonsei.ac.kr/>

사용된 3 가지 URL 은 다음과 같다.