WireShark

데이터 네트워크 연구실
이현호
lee075@cs-cnu.org

Previously

- 제출률 : 90%예상 -> 80% 이하
- 아직도 클래스룸에 안들어온 사람 손?

목표

- WireShark란 무엇인지 안다.
- WireShark를 설치한다.
- WireShark의 사용법을 안다.
- WireShark를 이용하여 패킷 캡쳐를 한다.

What is WireShark?

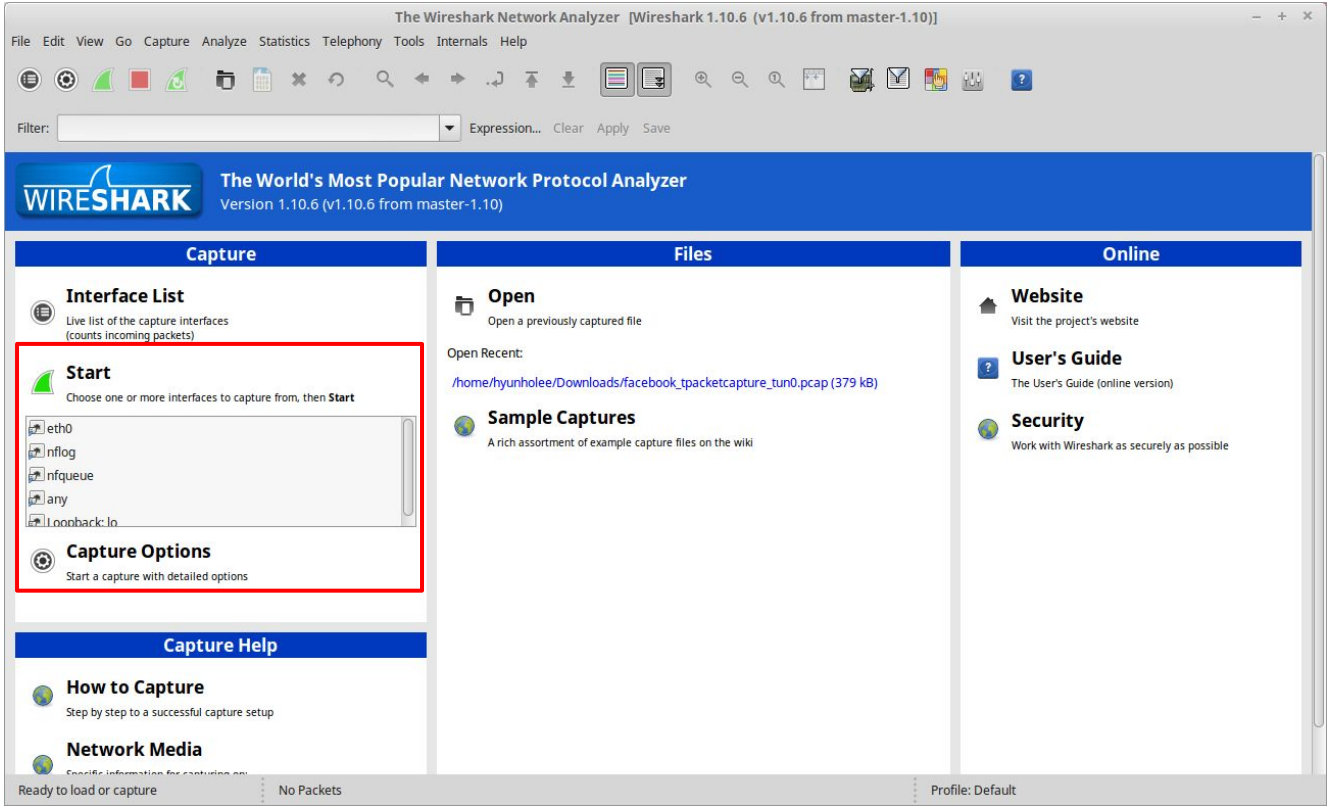
- https://www.wireshark.org/
- 널리 사용되는 네트워크 분석 프로그램
- Open-Source (GPL v2)
- Multi-Platform (Windows, Linux, Mac, ..)
- 이더넷, 토큰링, ATM 등의 네트워크 하드웨어로부터 패킷 캡쳐 가능
- Live Capture 및 Offline 분석 가능
- 암호화된 패킷 분석 가능
- 필터링 기능 - 원하는 패킷만 캡쳐 가능

DownLoad WireShark

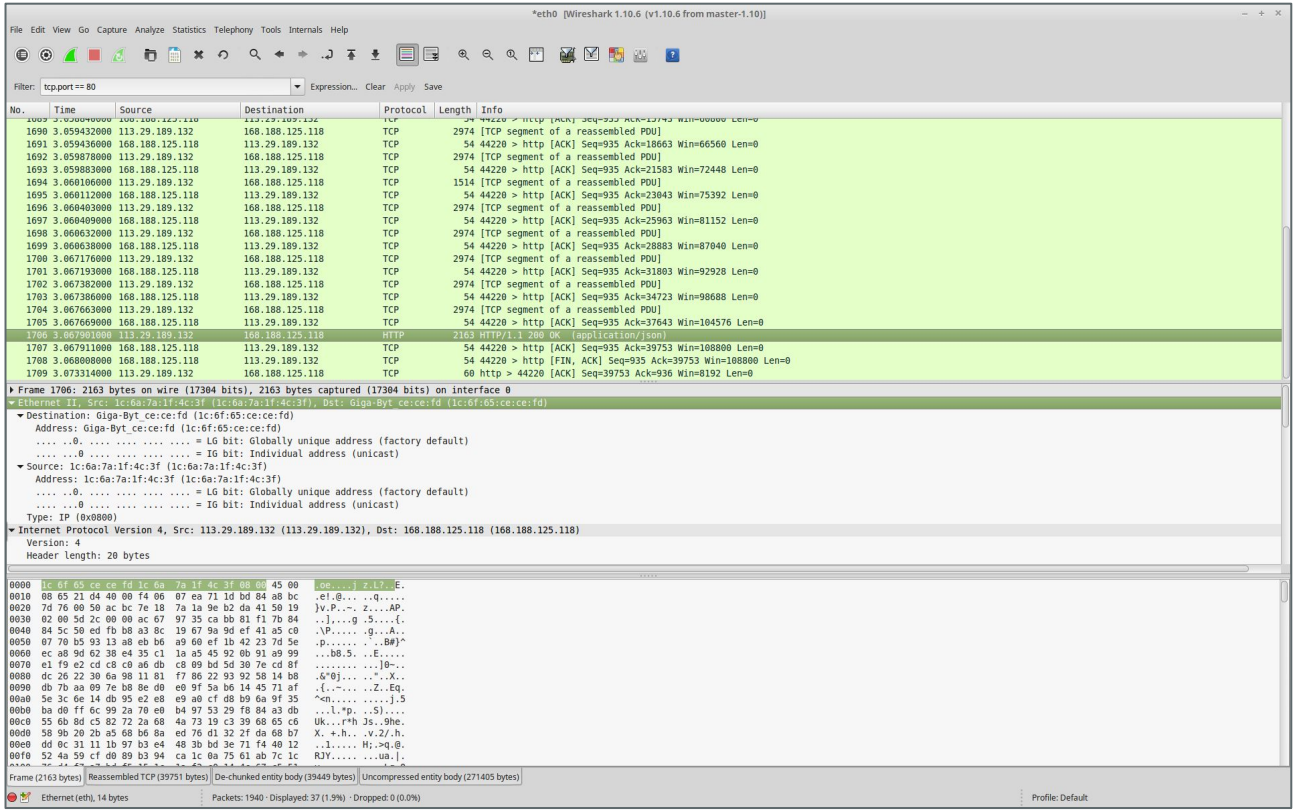
- https://www.wireshark.org/download.html
- 설치는 자유롭게
- /sudo wireshark

```
hyunholee@DNLAB ~ $ sudo apt-get install wireshark
Reading package lists... Done
Building dependency tree
Reading state information... Done
wireshark is already the newest version.
The following packages were automatically installed and are no longer required:
  ax25-node libax25 libtcl8.5 libtk8.5 openbsd-inetd tcl8.5 tk8.5
Use 'apt-get autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 418 not upgraded.
```

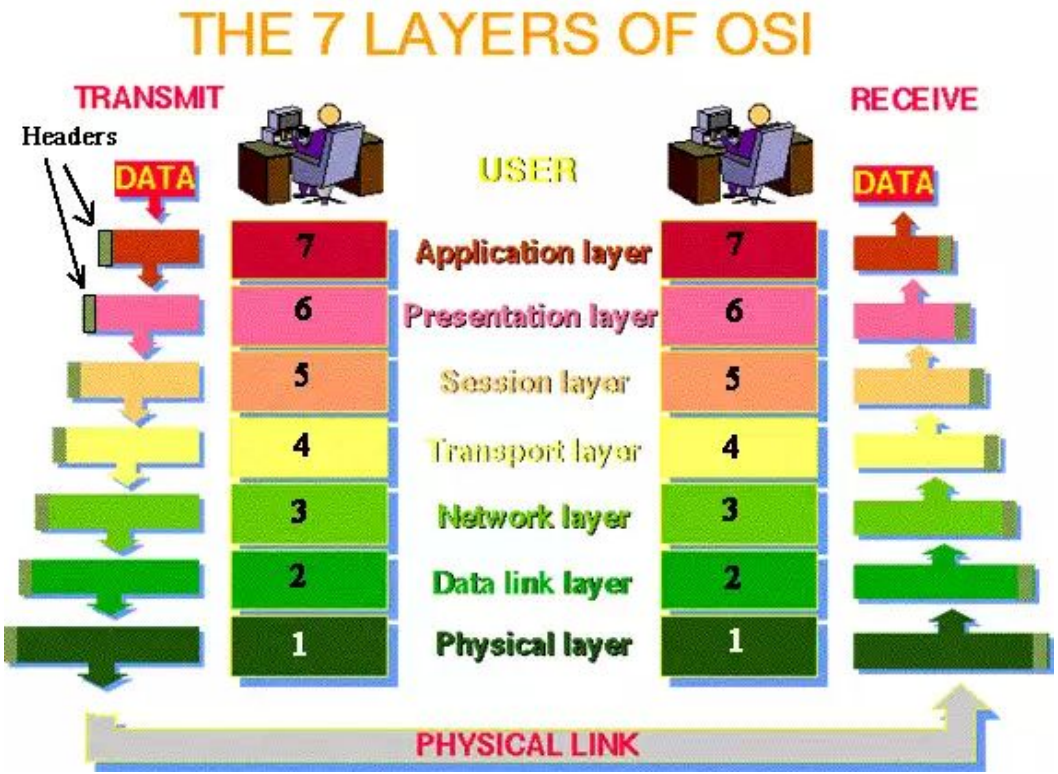
Main page


The Wireshark Network Analyzer [Wireshark 1.10.6 (v1.10.6 from master-1.10)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
Expression... Clear Apply Save
WIRESHARK
The World's Most Popular Network Protocol Analyzer
Version 1.10.6 (v1.10.6 from master-1.10)
Capture
Interface List
Live list of the capture interfaces
(counts incoming packets)
Start
Choose one or more interfaces to capture from, then Start
eth0
nflog
nfqueue
any
Loopback: lo
Capture Options
Start a capture with detailed options
Capture Help
How to Capture
Step by step to a successful capture setup
Network Media
Files
Open
Open a previously captured file
Open Recent:
/home/hyunholee/Downloads/facebook_tpacketcapture_tun0.pcap (379 kB)
Sample Captures
A rich assortment of example capture files on the wiki
Online
Website
Visit the project's website
User's Guide
The User's Guide (online version)
Security
Work with Wireshark as securely as possible
Ready to load or capture
No Packets
Profile: Default

START

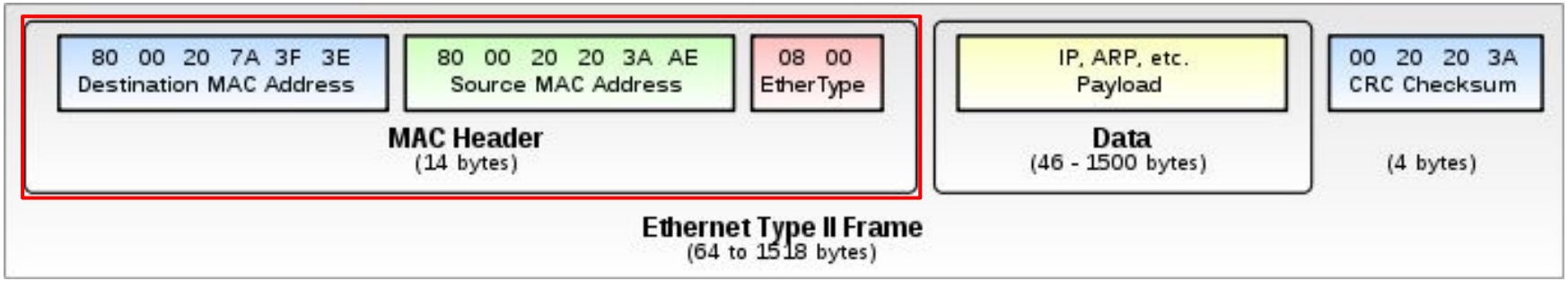

*eth0 [Wireshark 1.10.6 (v1.10.6 from master-1.10)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: tcp.port == 80
Expression... Clear Apply Save
No. Time Source Destination Protocol Length Info
1690 3.059432000 113.29.189.132 168.188.125.118 TCP 2974 [TCP segment of a reassembled PDU]
1691 3.059436000 168.188.125.118 113.29.189.132 TCP 54 44220 > http [ACK] Seq=935 Ack=18663 Win=66560 Len=0
1692 3.059870000 113.29.189.132 168.188.125.118 TCP 2974 [TCP segment of a reassembled PDU]
1693 3.059883000 168.188.125.118 113.29.189.132 TCP 54 44220 > http [ACK] Seq=935 Ack=21583 Win=72448 Len=0
1694 3.060106000 113.29.189.132 168.188.125.118 TCP 1514 [TCP segment of a reassembled PDU]
1695 3.060112000 168.188.125.118 113.29.189.132 TCP 54 44220 > http [ACK] Seq=935 Ack=23043 Win=75392 Len=0
1696 3.060403000 113.29.189.132 168.188.125.118 TCP 2974 [TCP segment of a reassembled PDU]
1697 3.060409000 168.188.125.118 113.29.189.132 TCP 54 44220 > http [ACK] Seq=935 Ack=25963 Win=81152 Len=0
1698 3.060632000 113.29.189.132 168.188.125.118 TCP 2974 [TCP segment of a reassembled PDU]
1699 3.060638000 168.188.125.118 113.29.189.132 TCP 54 44220 > http [ACK] Seq=935 Ack=28883 Win=87040 Len=0
1700 3.067176000 113.29.189.132 168.188.125.118 TCP 2974 [TCP segment of a reassembled PDU]
1701 3.067193000 168.188.125.118 113.29.189.132 TCP 54 44220 > http [ACK] Seq=935 Ack=31803 Win=92928 Len=0
1702 3.067382000 113.29.189.132 168.188.125.118 TCP 2974 [TCP segment of a reassembled PDU]
1703 3.067386000 168.188.125.118 113.29.189.132 TCP 54 44220 > http [ACK] Seq=935 Ack=34723 Win=98688 Len=0
1704 3.067663000 113.29.189.132 168.188.125.118 TCP 2974 [TCP segment of a reassembled PDU]
1705 3.067669000 168.188.125.118 113.29.189.132 TCP 54 44220 > http [ACK] Seq=935 Ack=37643 Win=104576 Len=0
1706 3.067901000 113.29.189.132 168.188.125.118 HTTP 2163 HTTP/1.1 200 OK (application/json)
1707 3.067911000 168.188.125.118 113.29.189.132 TCP 54 44220 > http [ACK] Seq=935 Ack=39753 Win=108800 Len=0
1708 3.068008000 168.188.125.118 113.29.189.132 TCP 54 44220 > http [FIN, ACK] Seq=935 Ack=39753 Win=108800 Len=0
1709 3.073314000 113.29.189.132 168.188.125.118 TCP 60 http > 44220 [ACK] Seq=39753 Ack=936 Win=8192 Len=0
Frame 1706: 2163 bytes on wire (17304 bits), 2163 bytes captured (17304 bits) on interface 0
Ethernet II, Src: 1c:6a:7a:1f:4c:3f (1c:6a:7a:1f:4c:3f), Dst: Giga-Byt_ce:ce:fd (1c:6f:65:ce:ce:fd)
Destination: Giga-Byt_ce:ce:fd (1c:6f:65:ce:ce:fd)
Address: Giga-Byt_ce:ce:fd (1c:6f:65:ce:ce:fd)
.... ..0. .... = LG bit: Globally unique address (factory default)
....0 .... = IG bit: Individual address (unicast)
Source: 1c:6a:7a:1f:4c:3f (1c:6a:7a:1f:4c:3f)
Address: 1c:6a:7a:1f:4c:3f (1c:6a:7a:1f:4c:3f)
.... ..0. .... = LG bit: Globally unique address (factory default)
....0 .... = IG bit: Individual address (unicast)
Type: IP (0x0800)
Internet Protocol Version 4, Src: 113.29.189.132 (113.29.189.132), Dst: 168.188.125.118 (168.188.125.118)
Version: 4
Header length: 20 bytes
0000 1c 6f 65 ce ce fd 1c 6a 7a 1f 4c 3f 08 00 45 00 .oe....j z.L?..E.
0010 08 65 21 d4 40 00 f4 06 07 ea 71 1d bd 84 a8 bc .e!.@... ..q.....
0020 7d 76 00 50 ac bc 7e 18 7a 1a 9e b2 da 41 50 19 }v.P..~. z....AP.
0030 02 00 5d 2c 00 00 ac 67 97 35 ca bb 81 f1 7b 84 ..],...g .5....{.
0040 84 5c 50 ed fb b8 a3 8c 19 67 9a 9d ef 41 a5 c0 .\P..... .g...A..
0050 07 70 b5 93 13 a8 eb b6 a9 60 ef 1b 42 23 7d 5e .p...... .`..B#}^
0060 ec a8 9d 62 38 e4 35 c1 1a a5 45 92 0b 91 a9 99 ...b8.5. ..E.....
0070 e1 f9 e2 cd c8 c0 a6 db c8 09 bd 5d 30 7e cd 8f ....]0~..
0080 dc 26 22 30 6a 98 11 81 f7 86 22 93 92 58 14 b8 .&"0j... ..".X..
0090 db 7b aa 09 7e b8 8e d0 e0 9f 5a b6 14 45 71 af .{..~... ..Z..Eq.
00a0 5e 3c 6e 14 db 95 e2 e8 e9 a0 cf d8 b9 6a 9f 35 ^<n..... .....j.5
00b0 ba d0 ff 6c 99 2a 70 e0 b4 97 53 29 f8 84 a3 db ...l.*p. ..S)....
00c0 55 6b 8d c5 82 72 2a 68 4a 73 19 c3 39 68 65 c6 Uk...r*h Js..9he.
00d0 58 9b 20 2b a5 68 b6 8a ed 76 d1 32 2f da 68 b7 X. +.h.. .v.2/.h.
00e0 dd 8c 31 11 1b 97 b3 e4 48 3b bd 3e 71 f4 40 12 ..1..... H;.>q.@.
00f0 52 4a 59 cf d0 89 b3 94 ca 1c 0a 75 61 ab 7c 1c RJY..... ...ua.|.
Frame (2163 bytes) Reassembled TCP (39751 bytes) De-chunked entity body (39449 bytes) Uncompressed entity body (271405 bytes)
Ethernet (eth), 14 bytes
Packets: 1940 · Displayed: 37 (1.9%) · Dropped: 0 (0.0%)
Profile: Default

Headers

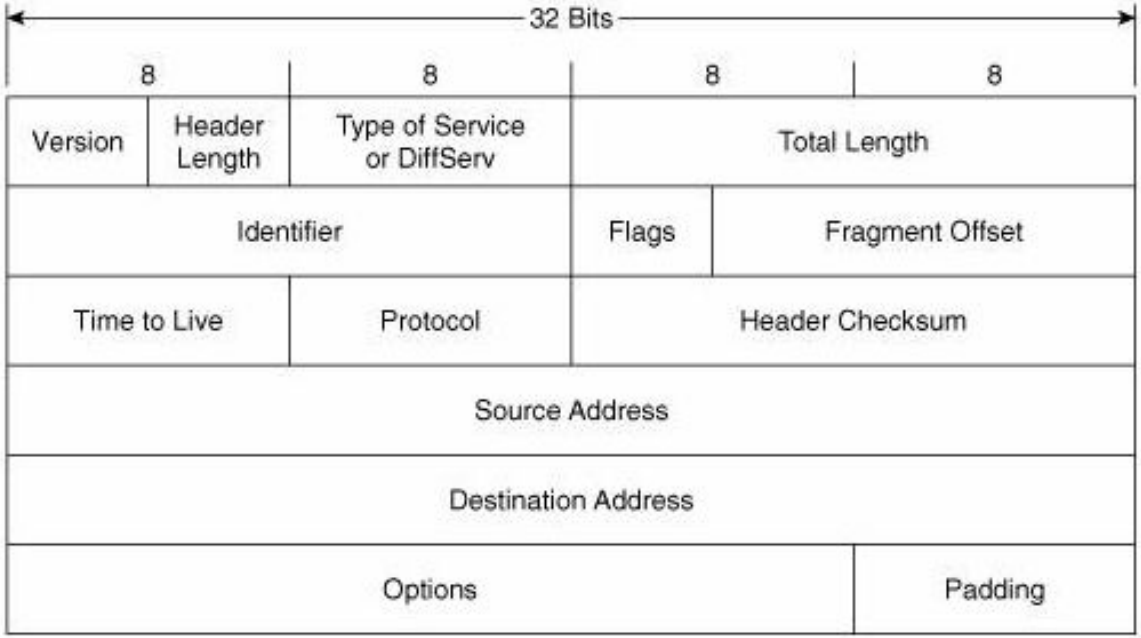

THE 7 LAYERS OF OSI
TRANSMIT
RECEIVE
Headers
DATA
USER
DATA
7
Application layer
7
6
Presentation layer
6
5
Session layer
5
4
Transport layer
4
3
Network layer
3
2
Data link layer
2
1
Physical layer
1
PHYSICAL LINK

Ethernet frame

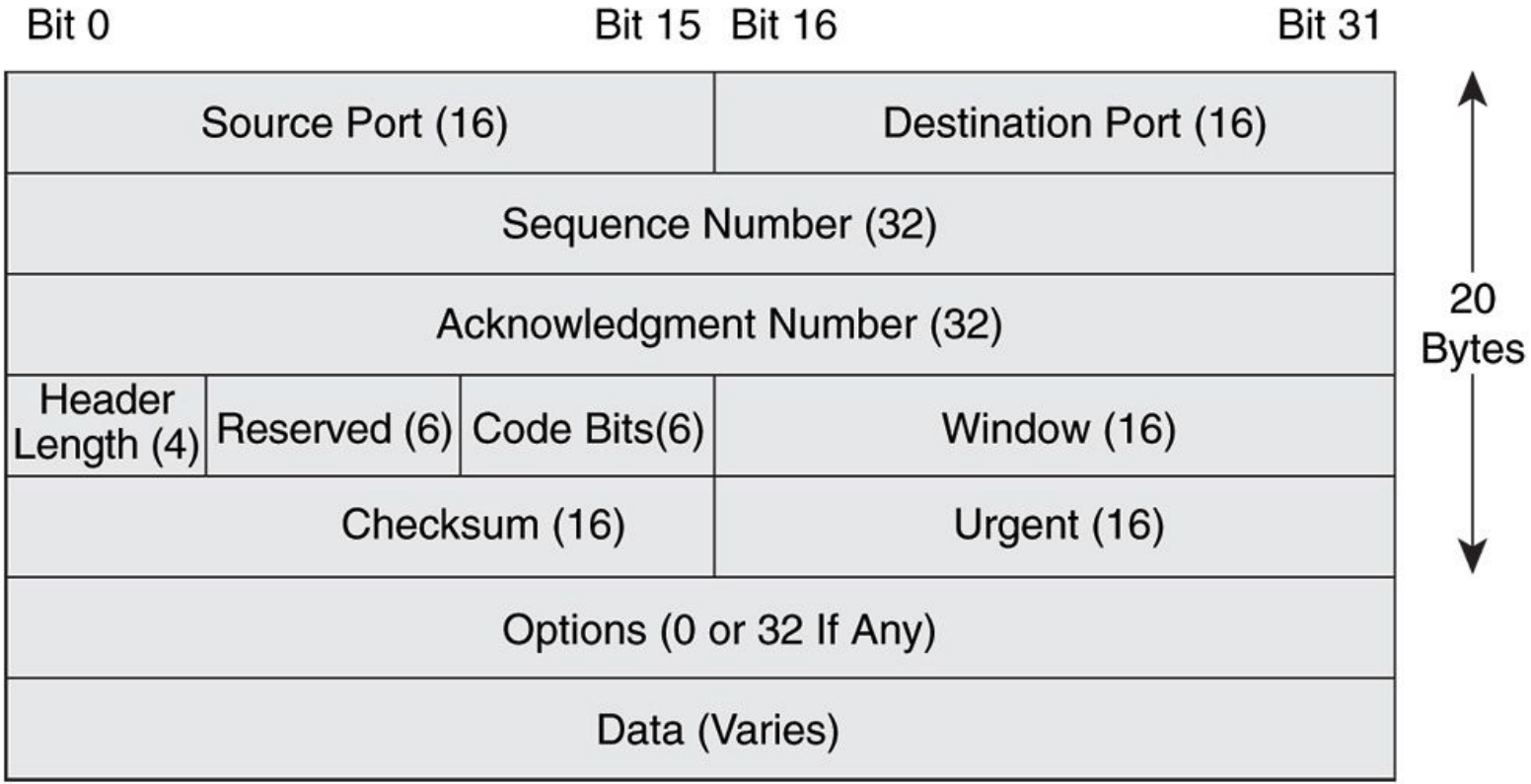
- 일단은 헤더의 구조만 익혀두자



IPv4 Header


32 Bits
8
8
8
8
Version
Header Length
Type of Service or DiffServ
Total Length
Identifier
Flags
Fragment Offset
Time to Live
Protocol
Header Checksum
Source Address
Destination Address
Options
Padding

TCP Header

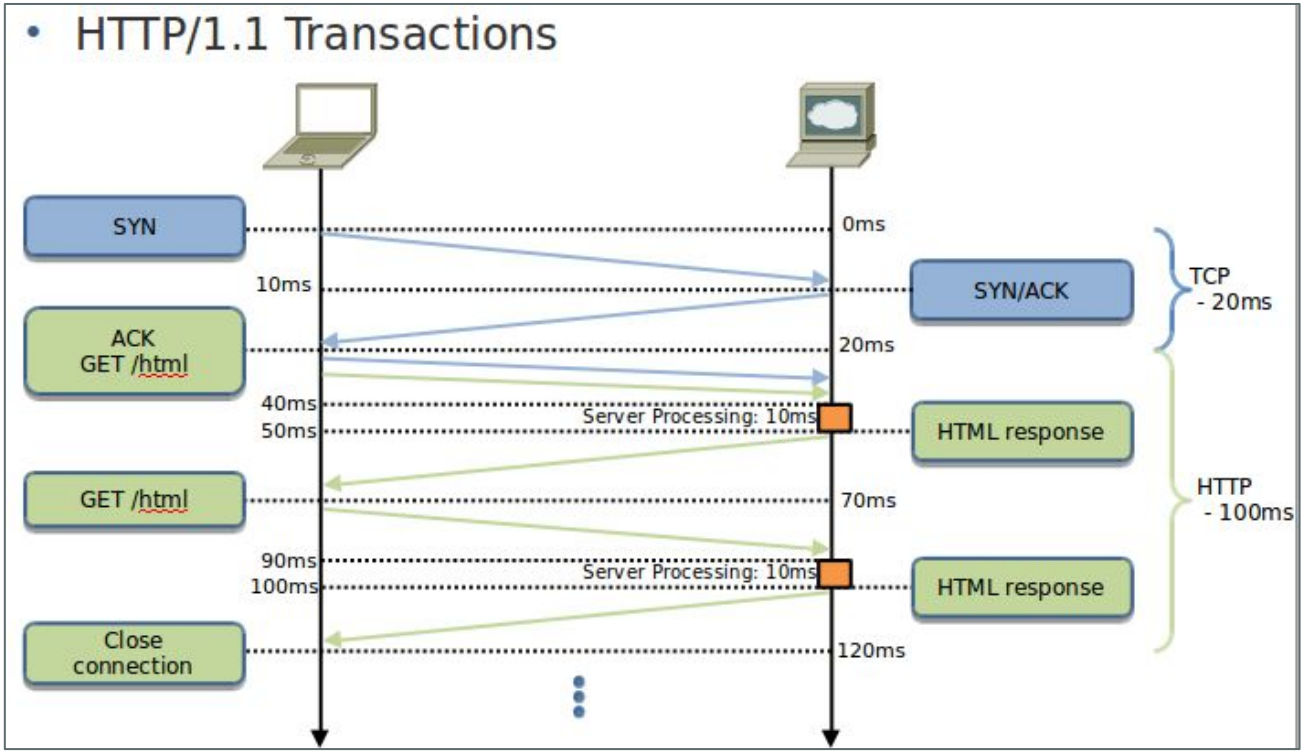

Bit 0
Bit 15
Bit 16
Bit 31
Source Port (16)
Destination Port (16)
Sequence Number (32)
Acknowledgment Number (32)
Header Length (4)
Reserved (6)
Code Bits(6)
Window (16)
Checksum (16)
Urgent (16)
Options (0 or 32 If Any)
Data (Varies)
20 Bytes

HTTP Header

- GET /test/test.htm HTTP /1.1
- HTTP /1.1 200 OK

GET	지정된 리소스(URI)를 요청
POST	서버가 클라이언트의 폼 입력 필드 데이터의 수락을 요청. 클라이언트는 서버로 HTTP Body 에 Data 를 전송한다
HAED	문서의 헤더 정보만 요청. 응답데이터(body) 를 받지 않는다
PUT	클라이언트가 전송한 데이터를 지정한 uri 로 대체 한다 ftp 의 put 와 동일. 역시 클라이언트는 서버로 HTTP Body 에 Data 를 전송한다
DELETE	클라이언트가 지정한 URI 를 서버에서 삭제
TRACE	클라이언트가 요청한 자원에 도달하기 까지의 경로를 기록하는 루프백(loop back) 검사용. 클라이언트가 요청 자원에 도달하기 까지 거쳐가는 프록시나 게이트웨이의 중간 경로부터 최종 수진 서버까지의 경로를 알아낼 때 사용.

HTTP의 전송 과정


HTTP/1.1 Transactions
SYN
0ms
10ms
SYN/ACK
TCP
- 20ms
ACK
GET /html
20ms
40ms
Server Processing: 10ms
50ms
HTML response
HTTP
- 100ms
GET /html
70ms
90ms
Server Processing: 10ms
100ms
HTML response
Close
connection
120ms

Filter

- Protocol:
- 사용 가능한 값: ether, fddi, ip, arp, rarp, decnet, lat, sca, moprc, mopdl, tcp and udp.
  - 프로토콜을 지정하지 않으면 모든 프로토콜을 사용합니다
- Direction:
- 사용 가능한 값: src, dst, src and dst, src or dst
- Logical Operations:
- 사용 가능한 값: not, and, or.

Filter

- eth.addr == 00:30:f9:00:00:01 출발지나 목적지 MAC 주소로 검색
- eth.src == 00:30:f9:00:00:01 출발지 MAC주소 검색
- eth.dst == 00:30:f9:00:00:01 목적지 MAC주소 검색
- ip.addr == 10.1.0.1 출발지나 목적지 IP주소로 검색
- ip.src == 10.1.0.1 출발지 IP주소로 검색
- tcp.port == 1470 TCP 출발지나 목적지 포트 번호로 검색
- tcp.dstport == 1470 TCP 포트 목적지 포트 번호로 검색
- tcp.srcport == 1470 TCP 포트 출발지 포트 번호로 검색
- udp.port == 2000 UDP 출발지나 포트 목적지 포트 번호로 검색
- udp.dstport == 2000 UDP 포트 목적지 포트 번호로 검색
- udp.srcport == 2000 UDP 포트 출발지 포트 번호로 검색

Practice

- 명령어를 실행 해 보고 그 의미를 파악해보자
- man netstat

```
slave@slave01 ~ $ netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         168.188.125.1   0.0.0.0         UG        0 0          0 eth0
168.188.125.0   0.0.0.0         255.255.255.0   U         0 0          0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U         0 0          0 eth0
```

Practice

- 명령어를 실행 해 보고 그 의미를 파악해보자
- man nslookup

```
slave@slave01 ~ $ nslookup www.naver.com
Server:         168.188.1.1
Address:        168.188.1.1#53

Non-authoritative answer:
www.naver.com   canonical name = www.naver.com.nheos.com.
Name:   www.naver.com.nheos.com
Address: 125.209.222.142
Name:   www.naver.com.nheos.com
Address: 125.209.222.141
```

Practice

- 명령어를 실행 해 보고 그 의미를 파악해보자
- man traceroute

```
slave@slave01 ~ $ traceroute daum.net
traceroute to daum.net (110.45.215.23), 30 hops max, 60 byte packets
 1  168.188.129.1 (168.188.129.1)  0.418 ms  0.486 ms  0.565 ms
 2  172.20.1.1 (172.20.1.1)  0.439 ms  0.486 ms  0.564 ms
 3  168.188.200.101 (168.188.200.101)  3.514 ms  8.744 ms  3.485 ms
 4  168.188.200.102 (168.188.200.102)  1.261 ms  1.486 ms  1.485 ms
 5  168.188.200.116 (168.188.200.116)  8.509 ms  8.479 ms  8.473 ms
 6  * * *
```

Assignment

- Q1. Linux에서 WireShark 설치 과정 스크린샷 첨부
- Q2. 다음 사이트에서 웹 사이트 5개를 선택하여 아래의 질문에 대한 답을 할 수 있는 패킷을 선택하여 답하고 pdf 보고서로 작성
    - http://www.alexa.com/topsites/countries/KR

Question

1. Is the frame an outgoing or an incoming frame?
2. What is the source IP address of the network-layer header in the frame?
3. What is the destination IP address of the network-layer header in the frame?
4. What is the total number of bytes in the whole frame?
5. What is the number of bytes in the Ethernet (data-link layer) header?
6. What is the number of bytes in the IP header?
7. What is the number of bytes in the TCP header?
8. What is the total bytes in the message (at the application layer)?

Answer

Question	Answer
Outgoing or incoming	
Source IP address	
Destination IP address	
Total number of bytes	
Number of bytes in the Ethernet header	
Number of bytes in the IP header	
Number of bytes in the TCP header	
total bytes in the message	

과제 제출

- 과제 제출 기한:
  - 실습 하루 전 18시
- Google Classroom에 제출
  - E-mail이 아닌 Classroom
- 보고서 제목 : DC_학번_이름_실습번호.pdf
- 추가 첨부파일 : DC_학번_이름_실습번호.zip

제출 파일 내용

- DC_학번_이름_실습번호.zip
  - 각종 소스코드
  - 그 외 파일
  - 보고서는 .pdf (**DC_학번_이름_실습번호.pdf**)
  - .hwp/.doc 등 채점 안 함
- 파일 이름 준수!
  - 파일 이름이 다를 경우 채점 안 함

보고서

- 과제를 해결한 방법
  - 주요 소스코드 포함 및 주석
- 과제를 해결하기 위해 알아야 하는 것
- 결과 화면 캡쳐와 설명
- 기본적으로 보고서는 자신이 직접 과제를 해결했다는 것을 증명하기 위함