# 프로토콜 씹고 뜯고 맛보고 즐기기

## 프로토콜 분석 & 설계

구팔 (김석환)

BBConf 2022 summer

# 프로토콜

# 프로토콜

# 프로토콜

# 통신규약

# 프로토콜

# 통신규약

# 서로간의 약속

# 너무나도 복잡한 약속

Related Topics

HTTP

가이드:

- ▶ 리소스와 URIs
- ▶ HTTP 가이드
- ▶ HTTP 보안
- HTTP 접근 제어(CORS)
- HTTP 인증
- HTTP 캐싱
- HTTP 압축
- HTTP 조건부 요청
- HTTP 컨텐츠 협상
- HTTP 쿠키
- HTTP range 요청
- HTTP 리다이렉트
- HTTP 명세
- Feature policy

레퍼런스:

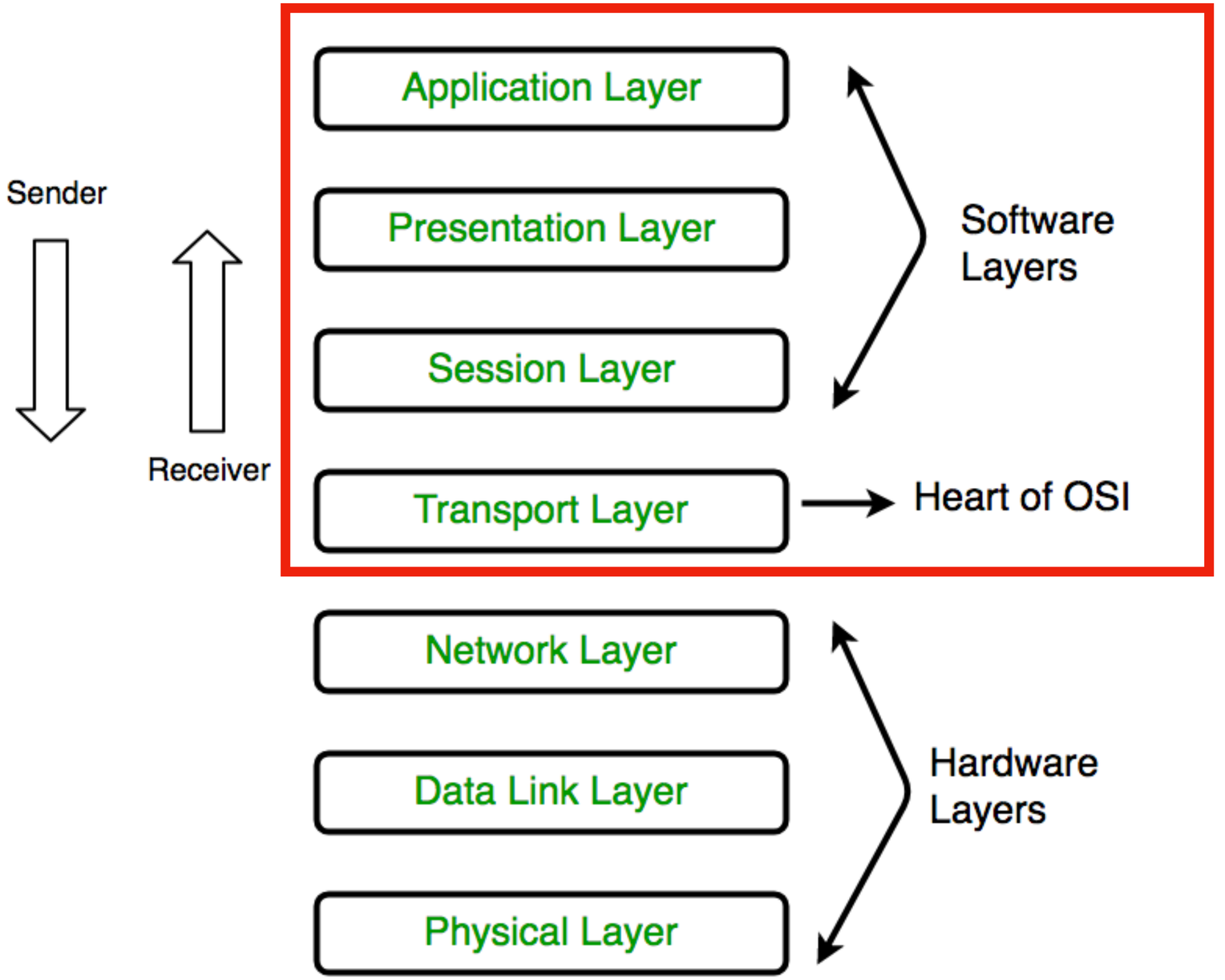
- ▶ HTTP 헤더
- ▶ HTTP 요청 메소드
- ▶ HTTP 응답 상태 코드

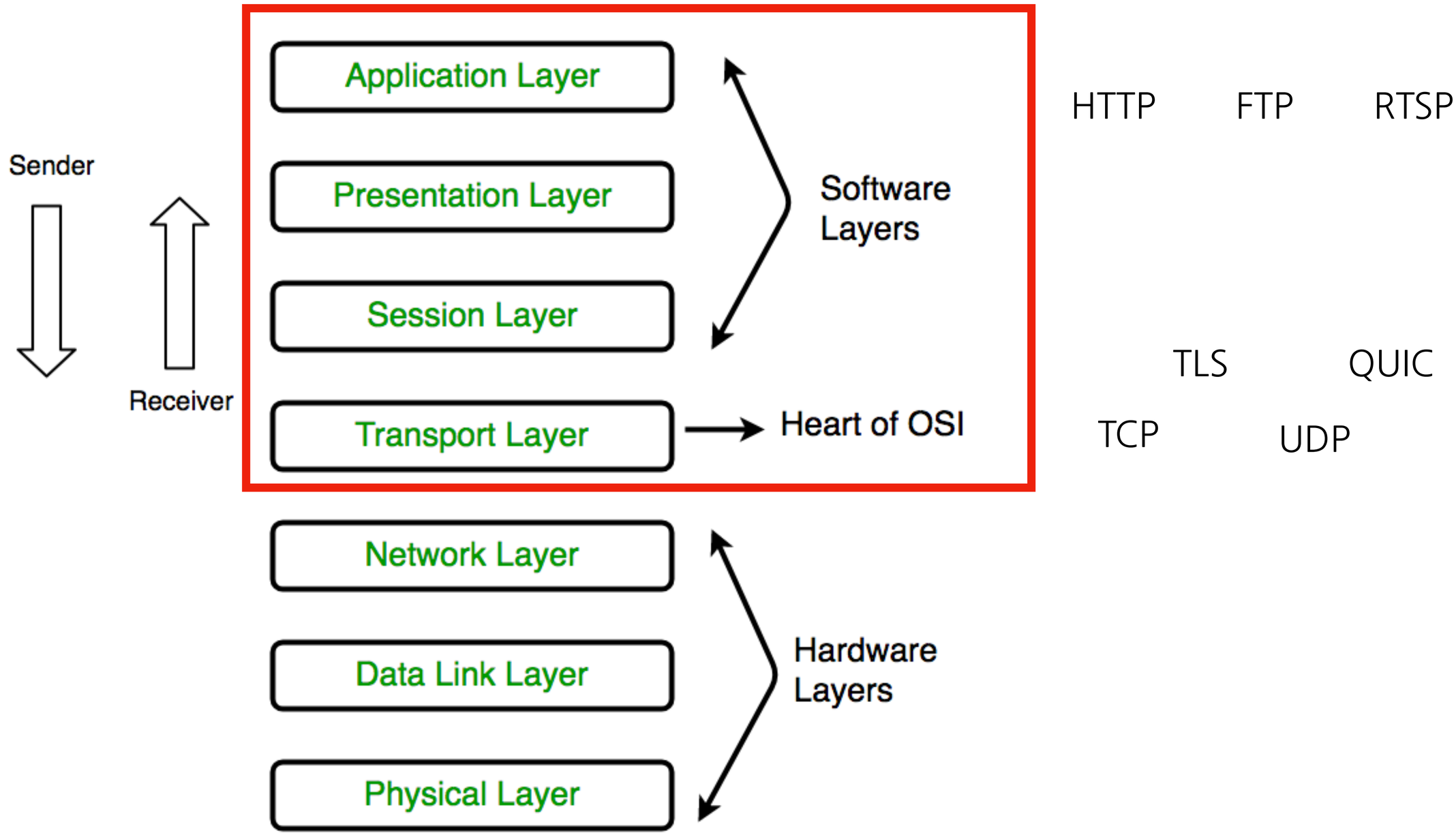
# HTTP 리소스와 명세

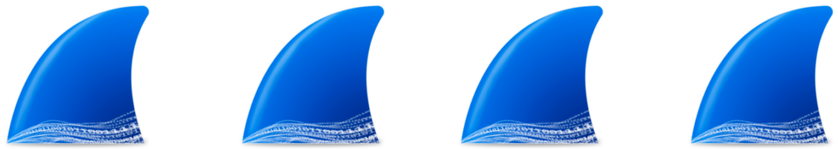
HTTP는 1990년 초에 처음으로 명세되었습니다. 확장성을 염두에 두고 설계하였고, 해를 거듭하면서 더 많은 추가 사항들이 세상에 나왔습니다; 이런 일로 많은 명세서를 통해 세상에 뿌려진 명세들이 나오게 되었습니다(실험되다가 폐기된 확장들 가운데에서도). 이 페이지에서는 HTTP와 관련해 의미가 있는 리소스들을 나열하고 있습니다.

| 명세 | 제목 | 상태 |
| --- | --- | --- |
| RFC 7230 | Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing | Proposed Standard |
| RFC 7231 | Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content | Proposed Standard |
| RFC 7232 | Hypertext Transfer Protocol (HTTP/1.1): Conditional Requests | Proposed Standard |
| RFC 7233 | Hypertext Transfer Protocol (HTTP/1.1): Range Requests | Proposed Standard |
| RFC 7234 | Hypertext Transfer Protocol (HTTP/1.1): Caching | Proposed Standard |
| RFC 5861 | HTTP Cache-Control Extensions for Stale Content | Informational |
| RFC 7235 | Hypertext Transfer Protocol (HTTP/1.1): Authentication | Proposed Standard |
| RFC 6265 | HTTP State Management Mechanism<br>*Defines Cookies* | Proposed Standard |
| Draft spec | Cookie Prefixes | IETF Draft |
| Draft spec | Same-Site Cookies | IETF Draft |
| RFC 2145 | Use and Interpretation of HTTP Version Numbers | Informational |
| RFC 6585 | Additional HTTP Status Codes | Proposed Standard |

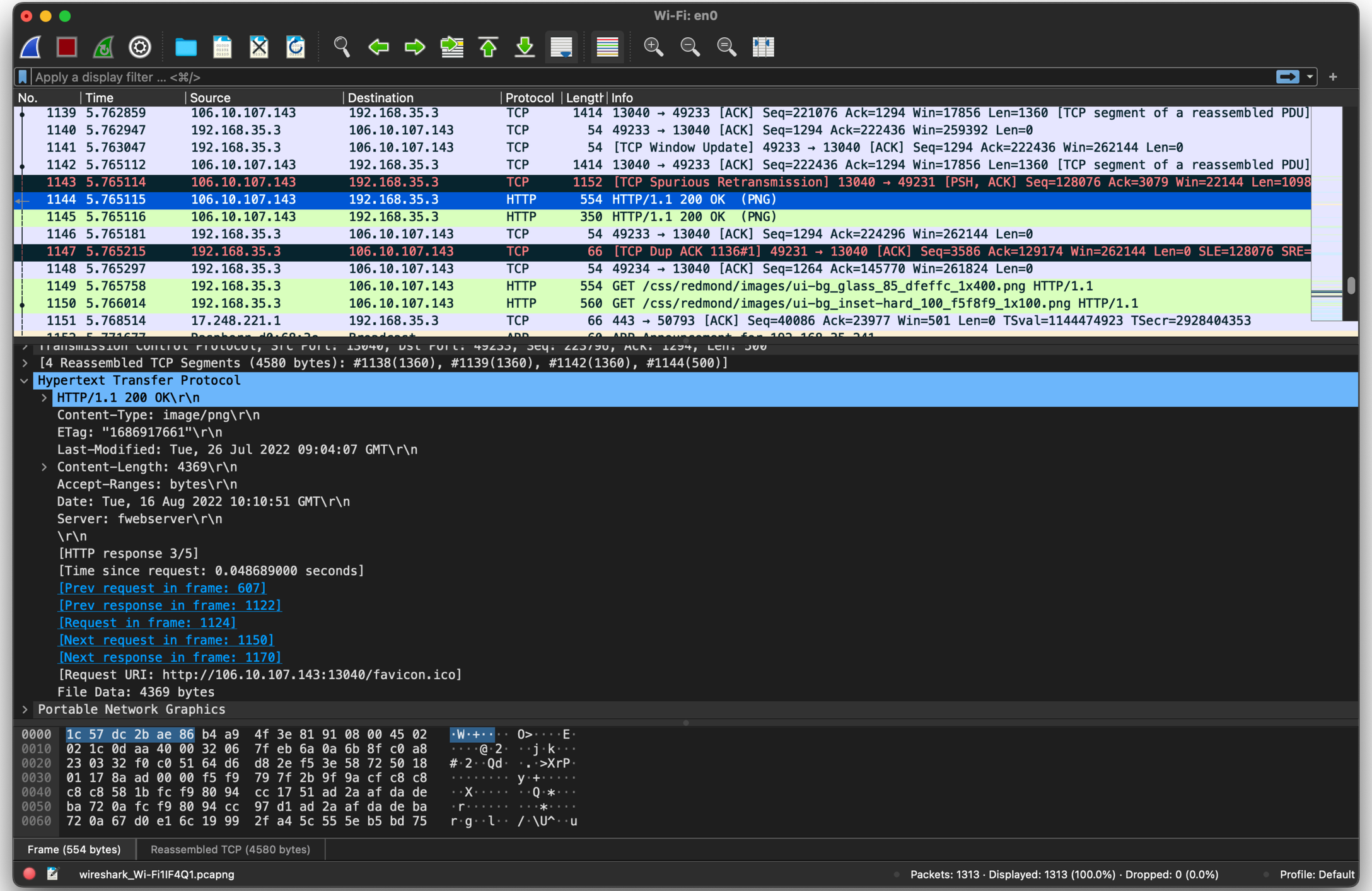
# 이 일들을 쉽게 하는방법

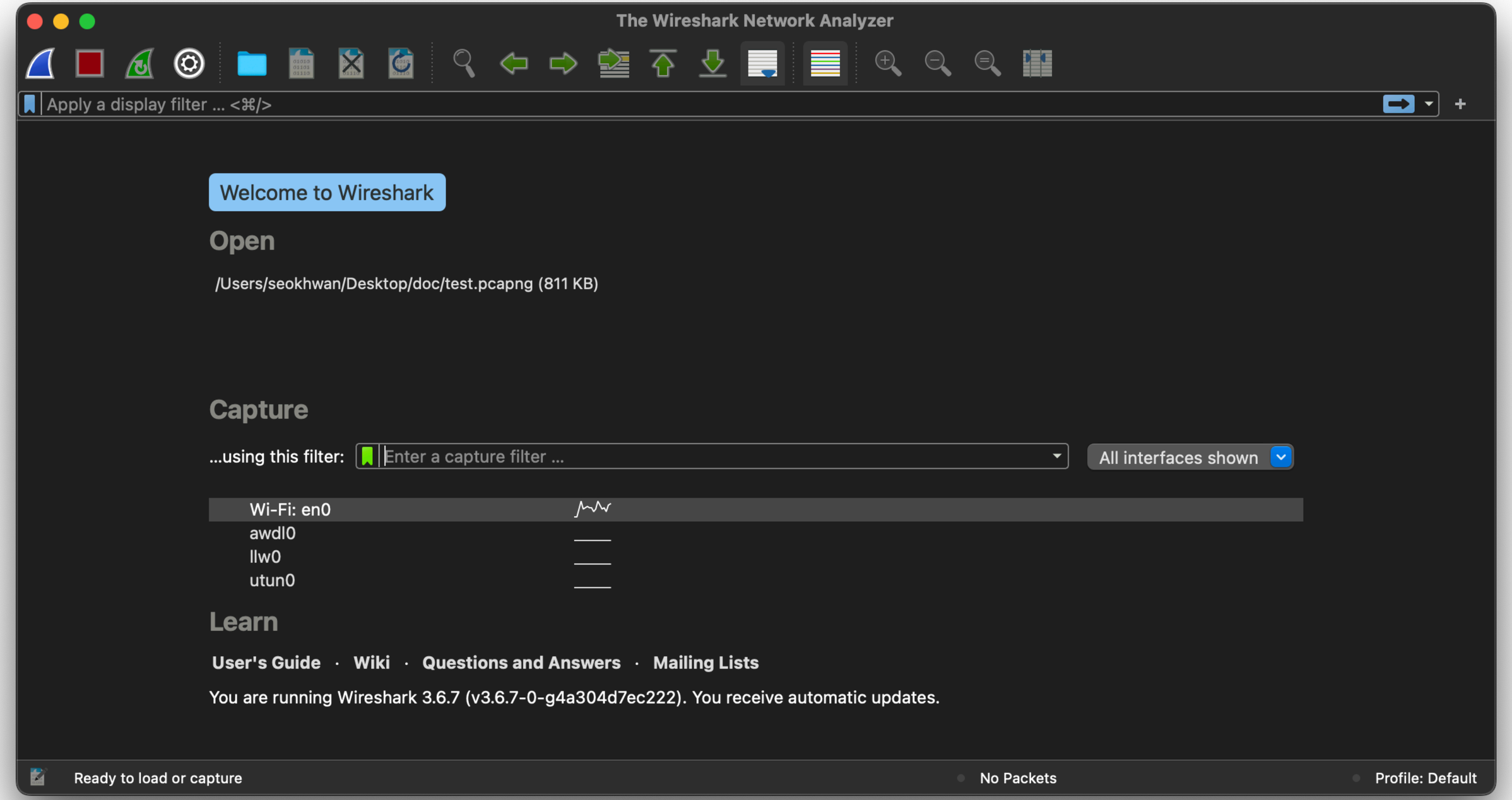

Sender
Receiver
Application Layer
Presentation Layer
Session Layer
Transport Layer
Software Layers
Heart of OSI
Network Layer
Data Link Layer
Physical Layer
Hardware Layers

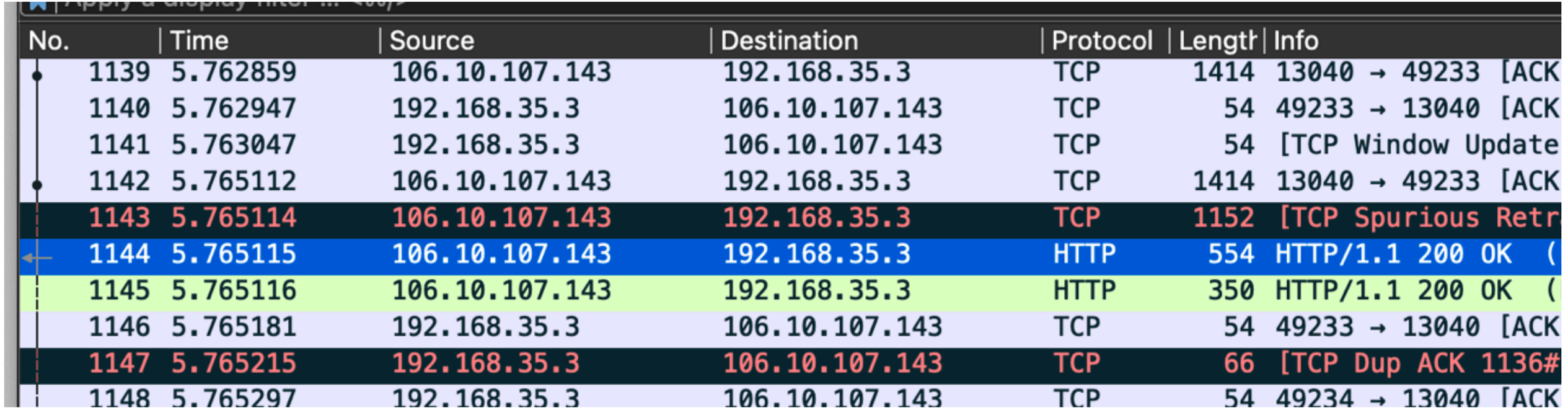

Application Layer
Presentation Layer
Session Layer
Transport Layer
Network Layer
Data Link Layer
Physical Layer
Software Layers
Hardware Layers
Heart of OSI
Sender
Receiver
HTTP
FTP
RTSP
TLS
QUIC
TCP
UDP



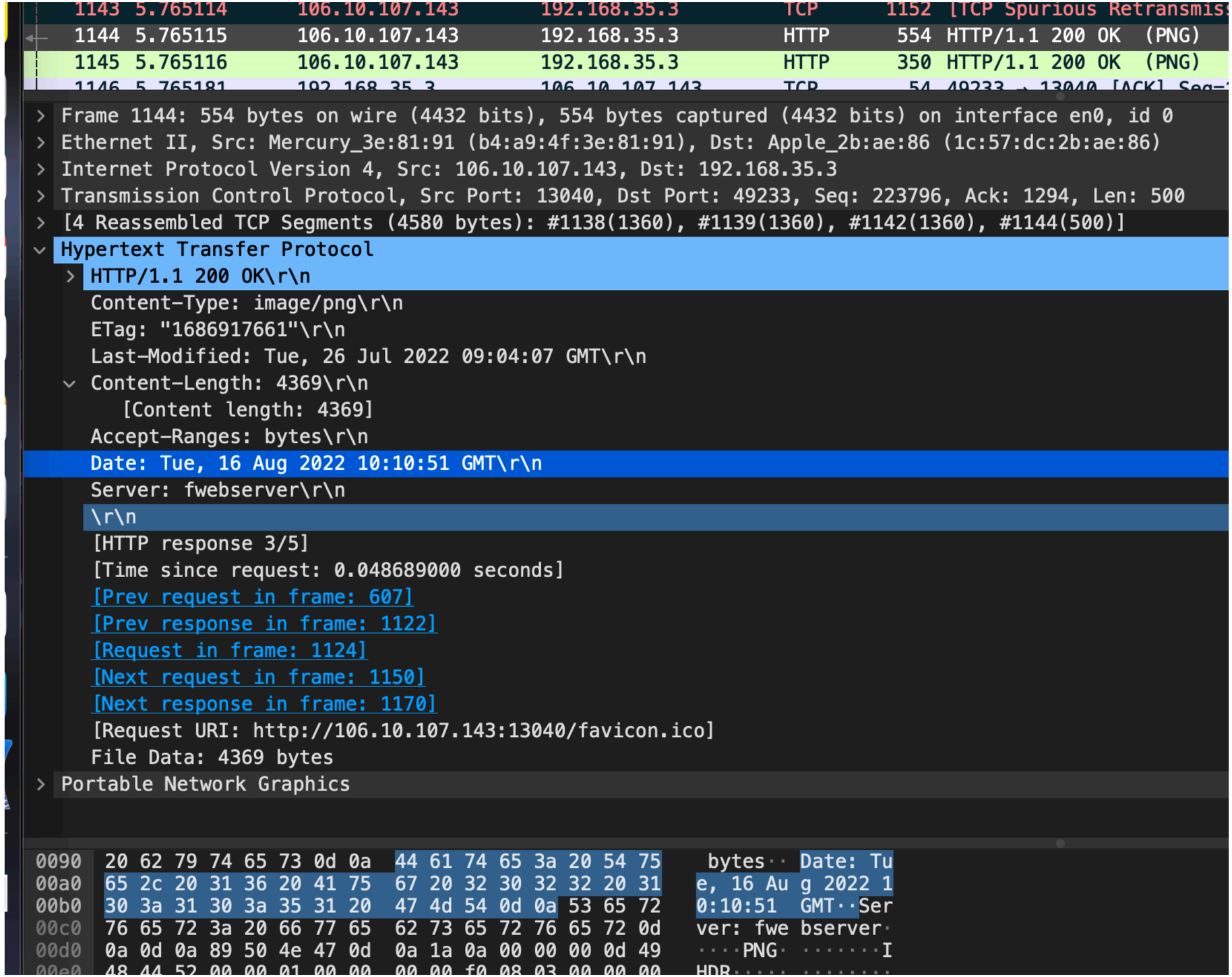
# 프로토콜 분석하기

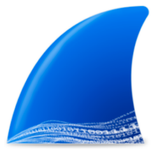

Wi-Fi: en0
Apply a display filter ... <⌘/>
No. Time Source Destination Protocol Length Info
1139 5.762859 106.10.107.143 192.168.35.3 TCP 1414 13040 → 49233 [ACK] Seq=221076 Ack=1294 Win=17856 Len=1360 [TCP segment of a reassembled PDU]
1140 5.762947 192.168.35.3 106.10.107.143 TCP 54 49233 → 13040 [ACK] Seq=1294 Ack=222436 Win=259392 Len=0
1141 5.763047 192.168.35.3 106.10.107.143 TCP 54 [TCP Window Update] 49233 → 13040 [ACK] Seq=1294 Ack=222436 Win=262144 Len=0
1142 5.765112 106.10.107.143 192.168.35.3 TCP 1414 13040 → 49233 [ACK] Seq=222436 Ack=1294 Win=17856 Len=1360 [TCP segment of a reassembled PDU]
1143 5.765114 106.10.107.143 192.168.35.3 TCP 1152 [TCP Spurious Retransmission] 13040 → 49231 [PSH, ACK] Seq=128076 Ack=3079 Win=22144 Len=1098
1144 5.765115 106.10.107.143 192.168.35.3 HTTP 554 HTTP/1.1 200 OK (PNG)
1145 5.765116 106.10.107.143 192.168.35.3 HTTP 350 HTTP/1.1 200 OK (PNG)
1146 5.765181 192.168.35.3 106.10.107.143 TCP 54 49233 → 13040 [ACK] Seq=1294 Ack=224296 Win=262144 Len=0
1147 5.765215 192.168.35.3 106.10.107.143 TCP 66 [TCP Dup ACK 1136#1] 49231 → 13040 [ACK] Seq=3586 Ack=129174 Win=262144 Len=0 SLE=128076 SRE=
1148 5.765297 192.168.35.3 106.10.107.143 TCP 54 49234 → 13040 [ACK] Seq=1264 Ack=145770 Win=261824 Len=0
1149 5.765758 192.168.35.3 106.10.107.143 HTTP 554 GET /css/redmond/images/ui-bg_glass_85_dfeffc_1x400.png HTTP/1.1
1150 5.766014 192.168.35.3 106.10.107.143 HTTP 560 GET /css/redmond/images/ui-bg_inset-hard_100_f5f8f9_1x100.png HTTP/1.1
1151 5.768514 17.248.221.1 192.168.35.3 TCP 66 443 → 50793 [ACK] Seq=40086 Ack=23977 Win=501 Len=0 TSval=1144474923 TSecr=2928404353
[4 Reassembled TCP Segments (4580 bytes): #1138(1360), #1139(1360), #1142(1360), #1144(500)]
Hypertext Transfer Protocol
HTTP/1.1 200 OK\r\n
Content-Type: image/png\r\n
ETag: "1686917661"\r\n
Last-Modified: Tue, 26 Jul 2022 09:04:07 GMT\r\n
Content-Length: 4369\r\n
Accept-Ranges: bytes\r\n
Date: Tue, 16 Aug 2022 10:10:51 GMT\r\n
Server: fwebserver\r\n
\r\n
[HTTP response 3/5]
[Time since request: 0.048689000 seconds]
[Prev request in frame: 607]
[Prev response in frame: 1122]
[Request in frame: 1124]
[Next request in frame: 1150]
[Next response in frame: 1170]
[Request URI: http://106.10.107.143:13040/favicon.ico]
File Data: 4369 bytes
Portable Network Graphics
0000 1c 57 dc 2b ae 86 b4 a9 4f 3e 81 91 08 00 45 02
0010 02 1c 0d aa 40 00 32 06 7f eb 6a 0a 6b 8f c0 a8
0020 23 03 32 f0 c0 51 64 d6 d8 2e f5 3e 58 72 50 18
0030 01 17 8a ad 00 00 f5 f9 79 7f 2b 9f 9a cf c8 c8
0040 c8 c8 58 1b fc f9 80 94 cc 17 51 ad 2a af da de
0050 ba 72 0a fc f9 80 94 cc 97 d1 ad 2a af da de ba
0060 72 0a 67 d0 e1 6c 19 99 2f a4 5c 55 5e b5 bd 75
Frame (554 bytes)
Reassembled TCP (4580 bytes)
wireshark_Wi-Fi1IF4Q1.pcapng
Packets: 1313 · Displayed: 1313 (100.0%) · Dropped: 0 (0.0%)
Profile: Default


The Wireshark Network Analyzer
Apply a display filter ... <⌘/>
Welcome to Wireshark
Open
/Users/seokhwan/Desktop/doc/test.pcapng (811 KB)
Capture
...using this filter:
Enter a capture filter ...
All interfaces shown
Wi-Fi: en0
awdl0
llw0
utun0
Learn
User's Guide · Wiki · Questions and Answers · Mailing Lists
You are running Wireshark 3.6.7 (v3.6.7-0-g4a304d7ec222). You receive automatic updates.
Ready to load or capture
No Packets
Profile: Default



| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 1139 | 5.762859 | 106.10.107.143 | 192.168.35.3 | TCP | 1414 | 13040 → 49233 [ACK |
| 1140 | 5.762947 | 192.168.35.3 | 106.10.107.143 | TCP | 54 | 49233 → 13040 [ACK |
| 1141 | 5.763047 | 192.168.35.3 | 106.10.107.143 | TCP | 54 | [TCP Window Update |
| 1142 | 5.765112 | 106.10.107.143 | 192.168.35.3 | TCP | 1414 | 13040 → 49233 [ACK |
| 1143 | 5.765114 | 106.10.107.143 | 192.168.35.3 | TCP | 1152 | [TCP Spurious Retr |
| 1144 | 5.765115 | 106.10.107.143 | 192.168.35.3 | HTTP | 554 | HTTP/1.1 200 OK ( |
| 1145 | 5.765116 | 106.10.107.143 | 192.168.35.3 | HTTP | 350 | HTTP/1.1 200 OK ( |
| 1146 | 5.765181 | 192.168.35.3 | 106.10.107.143 | TCP | 54 | 49233 → 13040 [ACK |
| 1147 | 5.765215 | 192.168.35.3 | 106.10.107.143 | TCP | 66 | [TCP Dup ACK 1136# |
| 1148 | 5.765297 | 192.168.35.3 | 106.10.107.143 | TCP | 54 | 49234 → 13040 [ACK |


1143 5.765114 106.10.107.143 192.168.35.3 TCP 1152 [TCP Spurious Retransmis
1144 5.765115 106.10.107.143 192.168.35.3 HTTP 554 HTTP/1.1 200 OK (PNG)
1145 5.765116 106.10.107.143 192.168.35.3 HTTP 350 HTTP/1.1 200 OK (PNG)
Frame 1144: 554 bytes on wire (4432 bits), 554 bytes captured (4432 bits) on interface en0, id 0
Ethernet II, Src: Mercury_3e:81:91 (b4:a9:4f:3e:81:91), Dst: Apple_2b:ae:86 (1c:57:dc:2b:ae:86)
Internet Protocol Version 4, Src: 106.10.107.143, Dst: 192.168.35.3
Transmission Control Protocol, Src Port: 13040, Dst Port: 49233, Seq: 223796, Ack: 1294, Len: 500
[4 Reassembled TCP Segments (4580 bytes): #1138(1360), #1139(1360), #1142(1360), #1144(500)]
Hypertext Transfer Protocol
HTTP/1.1 200 OK\r\n
Content-Type: image/png\r\n
ETag: "1686917661"\r\n
Last-Modified: Tue, 26 Jul 2022 09:04:07 GMT\r\n
Content-Length: 4369\r\n
[Content length: 4369]
Accept-Ranges: bytes\r\n
Date: Tue, 16 Aug 2022 10:10:51 GMT\r\n
Server: fwebserver\r\n
\r\n
[HTTP response 3/5]
[Time since request: 0.048689000 seconds]
[Prev request in frame: 607]
[Prev response in frame: 1122]
[Request in frame: 1124]
[Next request in frame: 1150]
[Next response in frame: 1170]
[Request URI: http://106.10.107.143:13040/favicon.ico]
File Data: 4369 bytes
Portable Network Graphics
0090 20 62 79 74 65 73 0d 0a 44 61 74 65 3a 20 54 75 bytes·· Date: Tu
00a0 65 2c 20 31 36 20 41 75 67 20 32 30 32 32 20 31 e, 16 Au g 2022 1
00b0 30 3a 31 30 3a 35 31 20 47 4d 54 0d 0a 53 65 72 0:10:51 GMT··Ser
00c0 76 65 72 3a 20 66 77 65 62 73 65 72 76 65 72 0d ver: fwe bserver·
00d0 0a 0d 0a 89 50 4e 47 0d 0a 1a 0a 00 00 00 0d 49 ····PNG· ·····I



# 필터

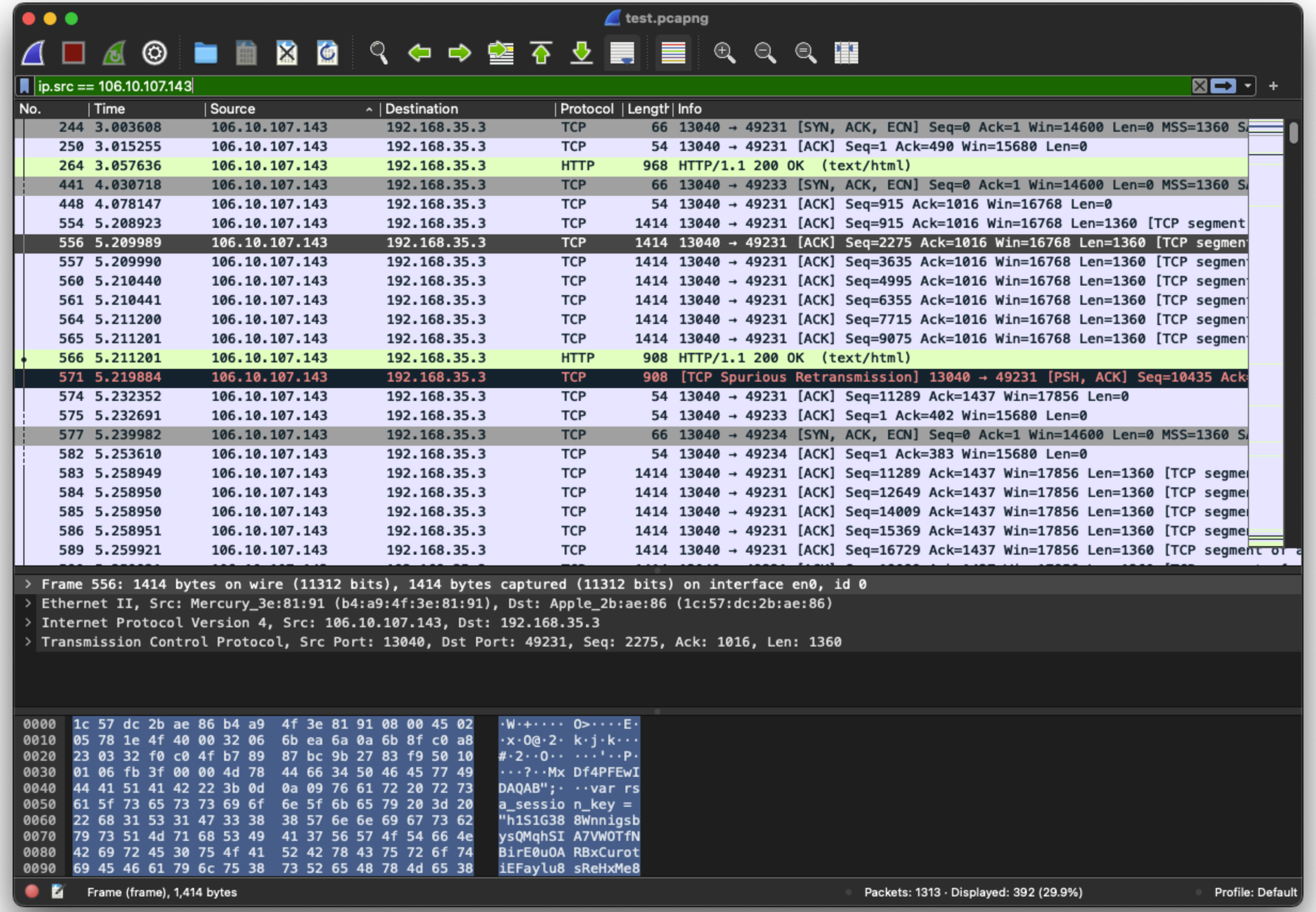
IP TCP 기타

# IP

IP.addr: IP address

IP.src: Source IP address

IP.dst: Destination IP address

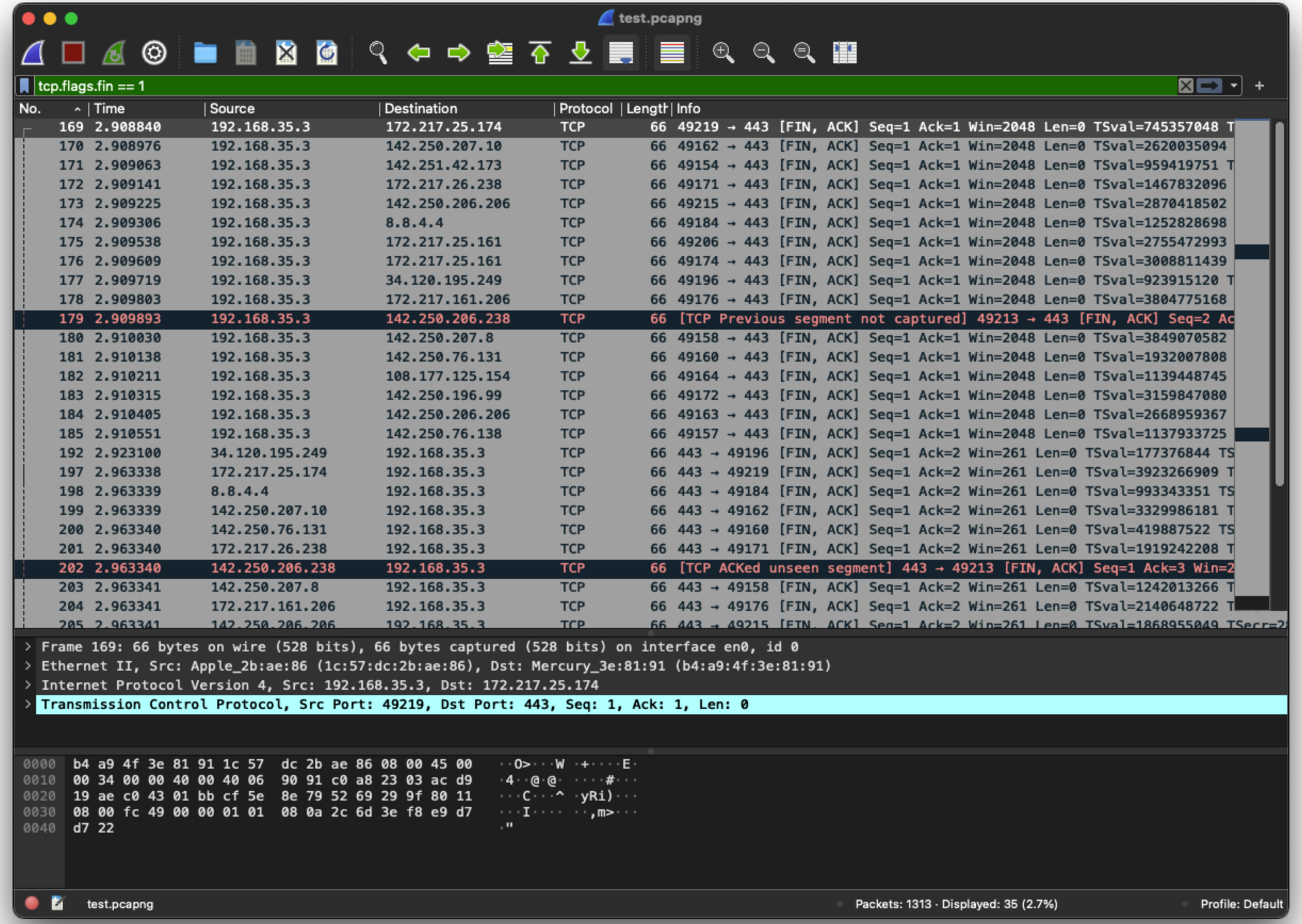

test.pcapng
ip.src == 106.10.107.143
No. Time Source Destination Protocol Length Info
244 3.003608 106.10.107.143 192.168.35.3 TCP 66 13040 → 49231 [SYN, ACK, ECN] Seq=0 Ack=1 Win=14600 Len=0 MSS=1360
250 3.015255 106.10.107.143 192.168.35.3 TCP 54 13040 → 49231 [ACK] Seq=1 Ack=490 Win=15680 Len=0
264 3.057636 106.10.107.143 192.168.35.3 HTTP 968 HTTP/1.1 200 OK (text/html)
441 4.030718 106.10.107.143 192.168.35.3 TCP 66 13040 → 49233 [SYN, ACK, ECN] Seq=0 Ack=1 Win=14600 Len=0 MSS=1360
448 4.078147 106.10.107.143 192.168.35.3 TCP 54 13040 → 49231 [ACK] Seq=915 Ack=1016 Win=16768 Len=0
554 5.208923 106.10.107.143 192.168.35.3 TCP 1414 13040 → 49231 [ACK] Seq=915 Ack=1016 Win=16768 Len=1360 [TCP segment
556 5.209989 106.10.107.143 192.168.35.3 TCP 1414 13040 → 49231 [ACK] Seq=2275 Ack=1016 Win=16768 Len=1360 [TCP segmen
557 5.209990 106.10.107.143 192.168.35.3 TCP 1414 13040 → 49231 [ACK] Seq=3635 Ack=1016 Win=16768 Len=1360 [TCP segmen
560 5.210440 106.10.107.143 192.168.35.3 TCP 1414 13040 → 49231 [ACK] Seq=4995 Ack=1016 Win=16768 Len=1360 [TCP segmen
561 5.210441 106.10.107.143 192.168.35.3 TCP 1414 13040 → 49231 [ACK] Seq=6355 Ack=1016 Win=16768 Len=1360 [TCP segmen
564 5.211200 106.10.107.143 192.168.35.3 TCP 1414 13040 → 49231 [ACK] Seq=7715 Ack=1016 Win=16768 Len=1360 [TCP segmen
565 5.211201 106.10.107.143 192.168.35.3 TCP 1414 13040 → 49231 [ACK] Seq=9075 Ack=1016 Win=16768 Len=1360 [TCP segmen
566 5.211201 106.10.107.143 192.168.35.3 HTTP 908 HTTP/1.1 200 OK (text/html)
571 5.219884 106.10.107.143 192.168.35.3 TCP 908 [TCP Spurious Retransmission] 13040 → 49231 [PSH, ACK] Seq=10435 Ack
574 5.232352 106.10.107.143 192.168.35.3 TCP 54 13040 → 49231 [ACK] Seq=11289 Ack=1437 Win=17856 Len=0
575 5.232691 106.10.107.143 192.168.35.3 TCP 54 13040 → 49233 [ACK] Seq=1 Ack=402 Win=15680 Len=0
577 5.239982 106.10.107.143 192.168.35.3 TCP 66 13040 → 49234 [SYN, ACK, ECN] Seq=0 Ack=1 Win=14600 Len=0 MSS=1360
582 5.253610 106.10.107.143 192.168.35.3 TCP 54 13040 → 49234 [ACK] Seq=1 Ack=383 Win=15680 Len=0
583 5.258949 106.10.107.143 192.168.35.3 TCP 1414 13040 → 49231 [ACK] Seq=11289 Ack=1437 Win=17856 Len=1360 [TCP segme
584 5.258950 106.10.107.143 192.168.35.3 TCP 1414 13040 → 49231 [ACK] Seq=12649 Ack=1437 Win=17856 Len=1360 [TCP segme
585 5.258950 106.10.107.143 192.168.35.3 TCP 1414 13040 → 49231 [ACK] Seq=14009 Ack=1437 Win=17856 Len=1360 [TCP segme
586 5.258951 106.10.107.143 192.168.35.3 TCP 1414 13040 → 49231 [ACK] Seq=15369 Ack=1437 Win=17856 Len=1360 [TCP segme
589 5.259921 106.10.107.143 192.168.35.3 TCP 1414 13040 → 49231 [ACK] Seq=16729 Ack=1437 Win=17856 Len=1360 [TCP segment
Frame 556: 1414 bytes on wire (11312 bits), 1414 bytes captured (11312 bits) on interface en0, id 0
Ethernet II, Src: Mercury_3e:81:91 (b4:a9:4f:3e:81:91), Dst: Apple_2b:ae:86 (1c:57:dc:2b:ae:86)
Internet Protocol Version 4, Src: 106.10.107.143, Dst: 192.168.35.3
Transmission Control Protocol, Src Port: 13040, Dst Port: 49231, Seq: 2275, Ack: 1016, Len: 1360
0000 1c 57 dc 2b ae 86 b4 a9 4f 3e 81 91 08 00 45 02
0010 05 78 1e 4f 40 00 32 06 6b ea 6a 0a 6b 8f c0 a8
0020 23 03 32 f0 c0 4f b7 89 87 bc 9b 27 83 f9 50 10
0030 01 06 fb 3f 00 00 4d 78 44 66 34 50 46 45 77 49
0040 44 41 51 41 42 22 3b 0d 0a 09 76 61 72 20 72 73
0050 61 5f 73 65 73 73 69 6f 6e 5f 6b 65 79 20 3d 20
0060 22 68 31 53 31 47 33 38 38 57 6e 6e 69 67 73 62
0070 79 73 51 4d 71 68 53 49 41 37 56 57 4f 54 66 4e
0080 42 69 72 45 30 75 4f 41 52 42 78 43 75 72 6f 74
0090 69 45 46 61 79 6c 75 38 73 52 65 48 78 4d 65 38
Frame (frame), 1,414 bytes
Packets: 1313 · Displayed: 392 (29.9%)
Profile: Default

# TCP

TCP

TCP.port, srcport, dstport

TCP.flag


test.pcapng
tcp.flags.fin == 1
No. | Time | Source | Destination | Protocol | Length | Info
169 2.908840 192.168.35.3 172.217.25.174 TCP 66 49219 → 443 [FIN, ACK] Seq=1 Ack=1 Win=2048 Len=0 TSval=745357048 T
170 2.908976 192.168.35.3 142.250.207.10 TCP 66 49162 → 443 [FIN, ACK] Seq=1 Ack=1 Win=2048 Len=0 TSval=2620035094
171 2.909063 192.168.35.3 142.251.42.173 TCP 66 49154 → 443 [FIN, ACK] Seq=1 Ack=1 Win=2048 Len=0 TSval=959419751 T
172 2.909141 192.168.35.3 172.217.26.238 TCP 66 49171 → 443 [FIN, ACK] Seq=1 Ack=1 Win=2048 Len=0 TSval=1467832096
173 2.909225 192.168.35.3 142.250.206.206 TCP 66 49215 → 443 [FIN, ACK] Seq=1 Ack=1 Win=2048 Len=0 TSval=2870418502
174 2.909306 192.168.35.3 8.8.4.4 TCP 66 49184 → 443 [FIN, ACK] Seq=1 Ack=1 Win=2048 Len=0 TSval=1252828698
175 2.909538 192.168.35.3 172.217.25.161 TCP 66 49206 → 443 [FIN, ACK] Seq=1 Ack=1 Win=2048 Len=0 TSval=2755472993
176 2.909609 192.168.35.3 172.217.25.161 TCP 66 49174 → 443 [FIN, ACK] Seq=1 Ack=1 Win=2048 Len=0 TSval=3008811439
177 2.909719 192.168.35.3 34.120.195.249 TCP 66 49196 → 443 [FIN, ACK] Seq=1 Ack=1 Win=2048 Len=0 TSval=923915120 T
178 2.909803 192.168.35.3 172.217.161.206 TCP 66 49176 → 443 [FIN, ACK] Seq=1 Ack=1 Win=2048 Len=0 TSval=3804775168
179 2.909893 192.168.35.3 142.250.206.238 TCP 66 [TCP Previous segment not captured] 49213 → 443 [FIN, ACK] Seq=2 Ac
180 2.910030 192.168.35.3 142.250.207.8 TCP 66 49158 → 443 [FIN, ACK] Seq=1 Ack=1 Win=2048 Len=0 TSval=3849070582
181 2.910138 192.168.35.3 142.250.76.131 TCP 66 49160 → 443 [FIN, ACK] Seq=1 Ack=1 Win=2048 Len=0 TSval=1932007808
182 2.910211 192.168.35.3 108.177.125.154 TCP 66 49164 → 443 [FIN, ACK] Seq=1 Ack=1 Win=2048 Len=0 TSval=1139448745
183 2.910315 192.168.35.3 142.250.196.99 TCP 66 49172 → 443 [FIN, ACK] Seq=1 Ack=1 Win=2048 Len=0 TSval=3159847080
184 2.910405 192.168.35.3 142.250.206.206 TCP 66 49163 → 443 [FIN, ACK] Seq=1 Ack=1 Win=2048 Len=0 TSval=2668959367
185 2.910551 192.168.35.3 142.250.76.138 TCP 66 49157 → 443 [FIN, ACK] Seq=1 Ack=1 Win=2048 Len=0 TSval=1137933725
192 2.923100 34.120.195.249 192.168.35.3 TCP 66 443 → 49196 [FIN, ACK] Seq=1 Ack=2 Win=261 Len=0 TSval=177376844 TS
197 2.963338 172.217.25.174 192.168.35.3 TCP 66 443 → 49219 [FIN, ACK] Seq=1 Ack=2 Win=261 Len=0 TSval=3923266909 T
198 2.963339 8.8.4.4 192.168.35.3 TCP 66 443 → 49184 [FIN, ACK] Seq=1 Ack=2 Win=261 Len=0 TSval=993343351 TS
199 2.963339 142.250.207.10 192.168.35.3 TCP 66 443 → 49162 [FIN, ACK] Seq=1 Ack=2 Win=261 Len=0 TSval=3329986181 T
200 2.963340 142.250.76.131 192.168.35.3 TCP 66 443 → 49160 [FIN, ACK] Seq=1 Ack=2 Win=261 Len=0 TSval=419887522 TS
201 2.963340 172.217.26.238 192.168.35.3 TCP 66 443 → 49171 [FIN, ACK] Seq=1 Ack=2 Win=261 Len=0 TSval=1919242208 T
202 2.963340 142.250.206.238 192.168.35.3 TCP 66 [TCP ACKed unseen segment] 443 → 49213 [FIN, ACK] Seq=1 Ack=3 Win=2
203 2.963341 142.250.207.8 192.168.35.3 TCP 66 443 → 49158 [FIN, ACK] Seq=1 Ack=2 Win=261 Len=0 TSval=1242013266 T
204 2.963341 172.217.161.206 192.168.35.3 TCP 66 443 → 49176 [FIN, ACK] Seq=1 Ack=2 Win=261 Len=0 TSval=2140648722 T
Frame 169: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface en0, id 0
Ethernet II, Src: Apple_2b:ae:86 (1c:57:dc:2b:ae:86), Dst: Mercury_3e:81:91 (b4:a9:4f:3e:81:91)
Internet Protocol Version 4, Src: 192.168.35.3, Dst: 172.217.25.174
Transmission Control Protocol, Src Port: 49219, Dst Port: 443, Seq: 1, Ack: 1, Len: 0
0000 b4 a9 4f 3e 81 91 1c 57 dc 2b ae 86 08 00 45 00
0010 00 34 00 00 40 00 40 06 90 91 c0 a8 23 03 ac d9
0020 19 ae c0 43 01 bb cf 5e 8e 79 52 69 29 9f 80 11
0030 08 00 fc 49 00 00 01 01 08 0a 2c 6d 3e f8 e9 d7
0040 d7 22
test.pcapng
Packets: 1313 · Displayed: 35 (2.7%)
Profile: Default

# 기타

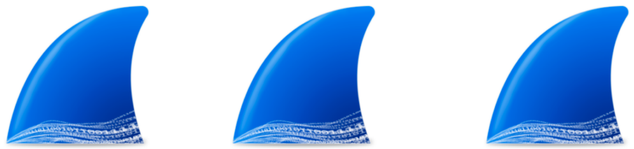
HTTP

QUIC

TLS

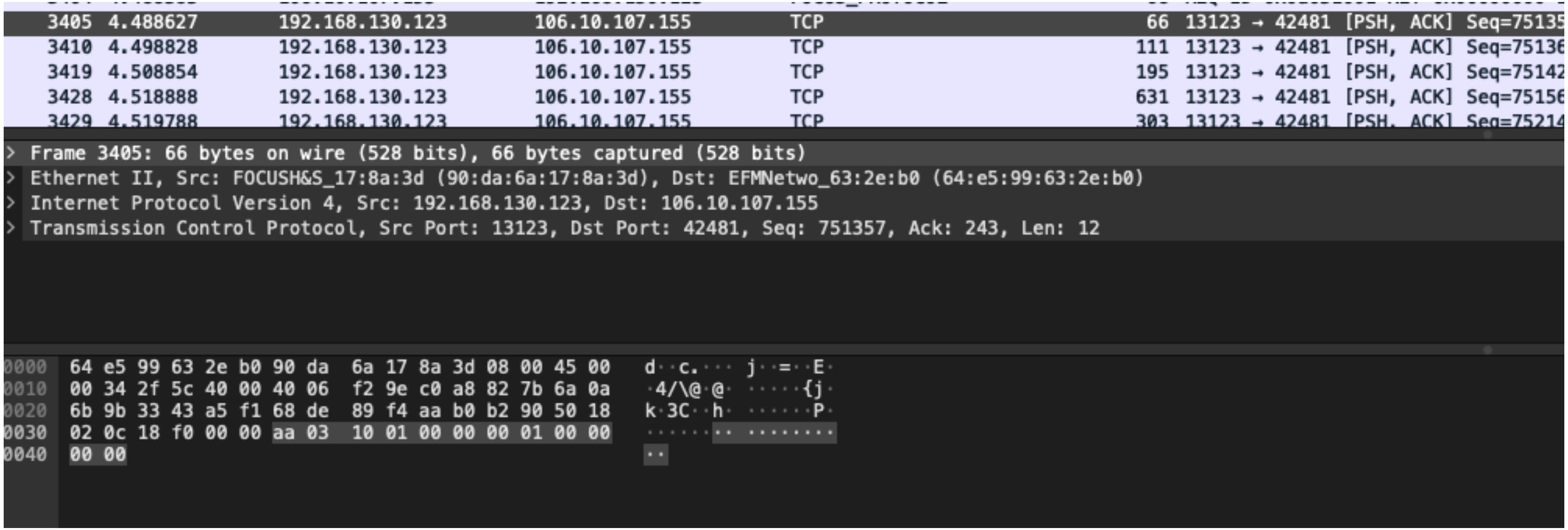
UDP

RTSP



# 커스텀 분석기

언제나 표준과 일 하지 않을 수 있음


3405 4.488627 192.168.130.123 106.10.107.155 TCP 66 13123 → 42481 [PSH, ACK] Seq=75135
3410 4.498828 192.168.130.123 106.10.107.155 TCP 111 13123 → 42481 [PSH, ACK] Seq=75136
3419 4.508854 192.168.130.123 106.10.107.155 TCP 195 13123 → 42481 [PSH, ACK] Seq=75142
3428 4.518888 192.168.130.123 106.10.107.155 TCP 631 13123 → 42481 [PSH, ACK] Seq=75156
3429 4.519788 192.168.130.123 106.10.107.155 TCP 303 13123 → 42481 [PSH, ACK] Seq=75214
> Frame 3405: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
> Ethernet II, Src: FOCUSH&S_17:8a:3d (90:da:6a:17:8a:3d), Dst: EFMNetwo_63:2e:b0 (64:e5:99:63:2e:b0)
> Internet Protocol Version 4, Src: 192.168.130.123, Dst: 106.10.107.155
> Transmission Control Protocol, Src Port: 13123, Dst Port: 42481, Seq: 751357, Ack: 243, Len: 12
0000 64 e5 99 63 2e b0 90 da 6a 17 8a 3d 08 00 45 00 d··c.··· j··=··E·
0010 00 34 2f 5c 40 00 40 06 f2 9e c0 a8 82 7b 6a 0a ·4/\@·@· ·····{j·
0020 6b 9b 33 43 a5 f1 68 de 89 f4 aa b0 b2 90 50 18 k·3C··h· ·····P·
0030 02 0c 18 f0 00 00 aa 03 10 01 00 00 00 01 00 00 ······· ·····
0040 00 00 ··

# Chapter 11. Wireshark's Lua API Reference Manual

**Table of Contents**

# 프로토콜 상세정보 필드 정의

```
-- Create FOCUS Streaming protocol
--  "focus_protocol" : 프로토콜 이름. Filter 창 등에서 사용
--  "FOCUSSTREAM" : Packet Detail, List의 Protocol 컬럼에 표시될 프로토콜 Description
p_focusprotocol = Proto ("focus_protocol", "FOCUSSTREAM")

local f = p_focusprotocol.fields

f.protocol_id = ProtoField.uint32("tpcstream.protocol_id", "PROTOCOL_ID", base.HEX)
f.protocol_return = ProtoField.uint32("tpcstream.protocol_return", "PROTOCOL_RETURN", base.HEX)
f.protocol_lenght = ProtoField.uint32("tpcstream.protocol_lenght", "PROTOCOL_LENGTH", base.DEC)
f.protocol_body = ProtoField.bytes("tpcstream.protocol_body", "PROTOCOL_BODY")
```

# 패킷 분석함수 작성 1

```
-- focus_protocol dissector function
function p_focusprotocol.dissector (buffer, pinfo, tree)
    -- validate packet length is adequate, otherwise quit
    if buffer:len() == 0 then return end

    local info_str = "";

    ----------------------------------------------------------
    -- 패킷 목록 표시창 info 컬럼에 표시될 정보
    ----------------------------------------------------------
    -- Protocol 컬럼에 표시될 프로토콜 이름 지정
    pinfo.cols.protocol = p_focusprotocol.name

    ----------------------------------------------------------
    -- 패킷 상세정보 창에 SubTree 추가하기
    ----------------------------------------------------------
    local reamaining = buffer:reported_length_remaining()
    local protocol_count = 0
    local pos = 0
    while reamaining > 0 do
```

# 패킷 분석함수 작성 2

```
-------------------------------------------------------
-- 패킷 상세정보 창에 SubTree 추가하기
-------------------------------------------
local reamaining = buffer:reported_length_remaining()
local protocol_count = 0
local pos = 0
while reamaining > 0 do

  local protocol_id = buffer(pos, 4)
  local protocol_flag = buffer(pos, 1):uint()

  local is_request = bitand(protocol_flag, 0xa0) == 0

  local protocol_return = buffer(pos + 4, 4)
  local protocol_length = buffer(pos + 8, 4):uint()

  local readable_data_length = math.min(reamaining, protocol_length)

  local subtree = tree:add(p_focusprotocol, buffer:range(pos, readable_data_length), "FocusPacket")

  readable_data_length = math.min(reamaining-12, protocol_length)

  subtree:add(f.protocol_id, protocol_id)
  subtree:add(f.protocol_return, protocol_return)
  subtree:add(f.protocol_lenght, protocol_length)
  subtree:add(f.protocol_body, buffer(pos + 12, readable_data_length))
```

# 패킷 분석함수 작성 3

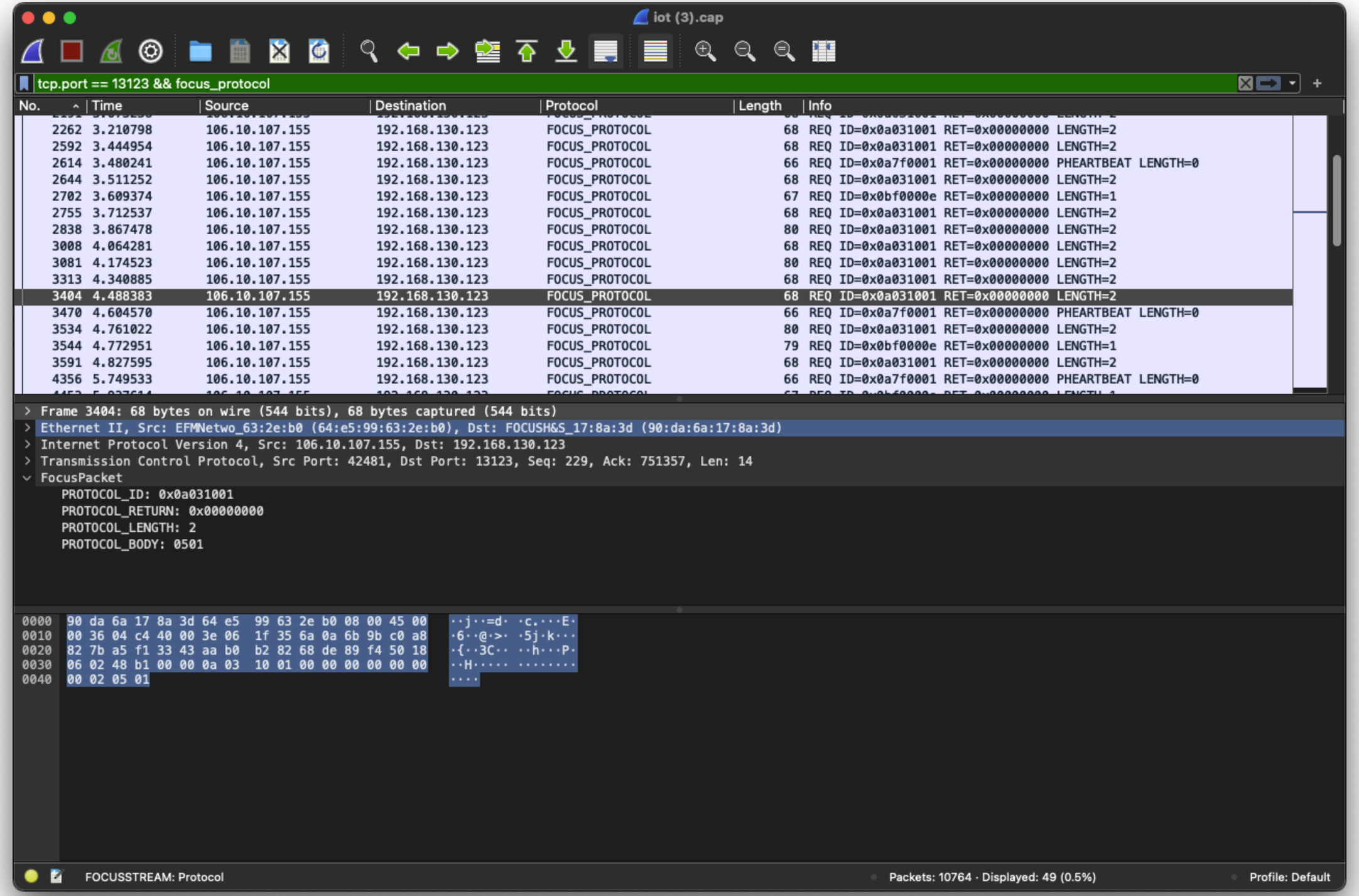
```
    subtree:add(f.protocol_body, buffer(pos + 12, readable_data_length))

    pos = pos + 12 + protocol_length
    reamaining = reamaining - (12 + protocol_length)

    -- 프로토콜 이름을 찾는 기준은 요청 ID로 통일한다
    if is_request then
      info_str = info_str.."REQ "
    else
      info_str = info_str.."RES "
    end
    info_str = info_str.."ID=0x"..protocol_id.." "
    info_str = info_str.."RET=0x"..protocol_return.." "
    local protocol_name_key = string.format("%08x", bitand(protocol_id:uint(), 0x0fffffff))
    info_str = info_str..get_protocol_name(protocol_name_key)
    info_str = info_str.."LENGTH="..protocol_length.." "
    protocol_count = protocol_count + 1
  end

  pinfo.cols.info = info_str

  if reamaining < 0 then
    pinfo.desegment_len = math.abs(reamaining)
    pinfo.desegment_offset = 0
    info_str = info_str .. "SEGMENT ".. (pinfo.desegment_len - pinfo.desegment_offset)
    pinfo.cols.info = info_str
    return
  end
end
```

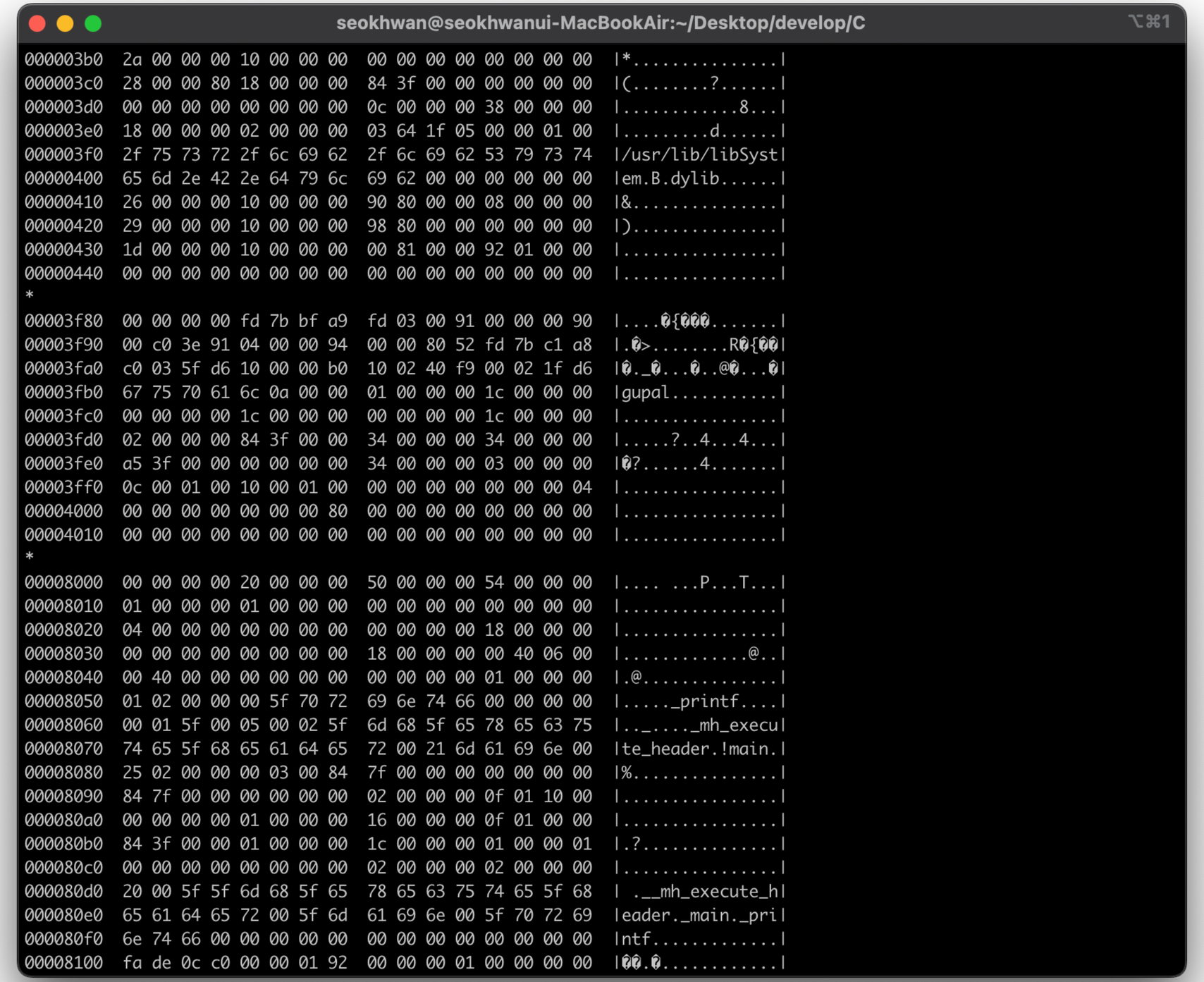

iot (3).cap
tcp.port == 13123 && focus_protocol
No. Time Source Destination Protocol Length Info
2262 3.210798 106.10.107.155 192.168.130.123 FOCUS_PROTOCOL 68 REQ ID=0x0a031001 RET=0x00000000 LENGTH=2
2592 3.444954 106.10.107.155 192.168.130.123 FOCUS_PROTOCOL 68 REQ ID=0x0a031001 RET=0x00000000 LENGTH=2
2614 3.480241 106.10.107.155 192.168.130.123 FOCUS_PROTOCOL 66 REQ ID=0x0a7f0001 RET=0x00000000 PHEARTBEAT LENGTH=0
2644 3.511252 106.10.107.155 192.168.130.123 FOCUS_PROTOCOL 68 REQ ID=0x0a031001 RET=0x00000000 LENGTH=2
2702 3.609374 106.10.107.155 192.168.130.123 FOCUS_PROTOCOL 67 REQ ID=0x0bf0000e RET=0x00000000 LENGTH=1
2755 3.712537 106.10.107.155 192.168.130.123 FOCUS_PROTOCOL 68 REQ ID=0x0a031001 RET=0x00000000 LENGTH=2
2838 3.867478 106.10.107.155 192.168.130.123 FOCUS_PROTOCOL 80 REQ ID=0x0a031001 RET=0x00000000 LENGTH=2
3008 4.064281 106.10.107.155 192.168.130.123 FOCUS_PROTOCOL 68 REQ ID=0x0a031001 RET=0x00000000 LENGTH=2
3081 4.174523 106.10.107.155 192.168.130.123 FOCUS_PROTOCOL 80 REQ ID=0x0a031001 RET=0x00000000 LENGTH=2
3313 4.340885 106.10.107.155 192.168.130.123 FOCUS_PROTOCOL 68 REQ ID=0x0a031001 RET=0x00000000 LENGTH=2
3404 4.488383 106.10.107.155 192.168.130.123 FOCUS_PROTOCOL 68 REQ ID=0x0a031001 RET=0x00000000 LENGTH=2
3470 4.604570 106.10.107.155 192.168.130.123 FOCUS_PROTOCOL 66 REQ ID=0x0a7f0001 RET=0x00000000 PHEARTBEAT LENGTH=0
3534 4.761022 106.10.107.155 192.168.130.123 FOCUS_PROTOCOL 80 REQ ID=0x0a031001 RET=0x00000000 LENGTH=2
3544 4.772951 106.10.107.155 192.168.130.123 FOCUS_PROTOCOL 79 REQ ID=0x0bf0000e RET=0x00000000 LENGTH=1
3591 4.827595 106.10.107.155 192.168.130.123 FOCUS_PROTOCOL 68 REQ ID=0x0a031001 RET=0x00000000 LENGTH=2
4356 5.749533 106.10.107.155 192.168.130.123 FOCUS_PROTOCOL 66 REQ ID=0x0a7f0001 RET=0x00000000 PHEARTBEAT LENGTH=0
Frame 3404: 68 bytes on wire (544 bits), 68 bytes captured (544 bits)
Ethernet II, Src: EFMNetwo_63:2e:b0 (64:e5:99:63:2e:b0), Dst: FOCUSH&S_17:8a:3d (90:da:6a:17:8a:3d)
Internet Protocol Version 4, Src: 106.10.107.155, Dst: 192.168.130.123
Transmission Control Protocol, Src Port: 42481, Dst Port: 13123, Seq: 229, Ack: 751357, Len: 14
FocusPacket
PROTOCOL_ID: 0x0a031001
PROTOCOL_RETURN: 0x00000000
PROTOCOL_LENGTH: 2
PROTOCOL_BODY: 0501
0000 90 da 6a 17 8a 3d 64 e5 99 63 2e b0 08 00 45 00
0010 00 36 04 c4 40 00 3e 06 1f 35 6a 0a 6b 9b c0 a8
0020 82 7b a5 f1 33 43 aa b0 b2 82 68 de 89 f4 50 18
0030 06 02 48 b1 00 00 0a 03 10 01 00 00 00 00 00 00
0040 00 02 05 01
FOCUSSTREAM: Protocol
Packets: 10764 · Displayed: 49 (0.5%)
Profile: Default

# 프로토콜 만들기

# 데이터의 형태 정하기

바이너리?

텍스트?


seokhwan@seokhwanui-MacBookAir:~/Desktop/develop/C
⌥⌘1
000003b0  2a 00 00 00 10 00 00 00  00 00 00 00 00 00 00 00  |*...............|
000003c0  28 00 00 80 18 00 00 00  84 3f 00 00 00 00 00 00  |(........?......|
000003d0  00 00 00 00 00 00 00 00  0c 00 00 00 38 00 00 00  |............8...|
000003e0  18 00 00 00 02 00 00 00  03 64 1f 05 00 00 01 00  |.........d......|
000003f0  2f 75 73 72 2f 6c 69 62  2f 6c 69 62 53 79 73 74  |/usr/lib/libSyst|
00000400  65 6d 2e 42 2e 64 79 6c  69 62 00 00 00 00 00 00  |em.B.dylib......|
00000410  26 00 00 00 10 00 00 00  90 80 00 00 08 00 00 00  |&...............|
00000420  29 00 00 00 10 00 00 00  98 80 00 00 00 00 00 00  |)...............|
00000430  1d 00 00 00 10 00 00 00  00 81 00 00 92 01 00 00  |................|
00000440  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00003f80  00 00 00 00 fd 7b bf a9  fd 03 00 91 00 00 00 90  |....{...........|
00003f90  00 c0 3e 91 04 00 00 94  00 00 80 52 fd 7b c1 a8  |..>........R.{..|
00003fa0  c0 03 5f d6 10 00 00 b0  10 02 40 f9 00 02 1f d6  |.._.......@.....|
00003fb0  67 75 70 61 6c 0a 00 00  01 00 00 00 1c 00 00 00  |gupal...........|
00003fc0  00 00 00 00 1c 00 00 00  00 00 00 00 1c 00 00 00  |................|
00003fd0  02 00 00 00 84 3f 00 00  34 00 00 00 34 00 00 00  |.....?..4...4...|
00003fe0  a5 3f 00 00 00 00 00 00  34 00 00 00 03 00 00 00  |.?......4.......|
00003ff0  0c 00 01 00 10 00 01 00  00 00 00 00 00 00 00 04  |................|
00004000  00 00 00 00 00 00 00 80  00 00 00 00 00 00 00 00  |................|
00004010  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00008000  00 00 00 00 20 00 00 00  50 00 00 00 54 00 00 00  |.... ...P...T...|
00008010  01 00 00 00 01 00 00 00  00 00 00 00 00 00 00 00  |................|
00008020  04 00 00 00 00 00 00 00  00 00 00 00 18 00 00 00  |................|
00008030  00 00 00 00 00 00 00 00  18 00 00 00 00 40 06 00  |.............@..|
00008040  00 40 00 00 00 00 00 00  00 00 00 00 01 00 00 00  |.@..............|
00008050  01 02 00 00 00 5f 70 72  69 6e 74 66 00 00 00 00  |....._printf....|
00008060  00 01 5f 00 05 00 02 5f  6d 68 5f 65 78 65 63 75  |.._...._mh_execu|
00008070  74 65 5f 68 65 61 64 65  72 00 21 6d 61 69 6e 00  |te_header.!main.|
00008080  25 02 00 00 00 03 00 84  7f 00 00 00 00 00 00 00  |%...............|
00008090  84 7f 00 00 00 00 00 00  02 00 00 00 0f 01 10 00  |................|
000080a0  00 00 00 00 01 00 00 00  16 00 00 00 0f 01 00 00  |................|
000080b0  84 3f 00 00 01 00 00 00  1c 00 00 00 01 00 00 01  |.?..............|
000080c0  00 00 00 00 00 00 00 00  02 00 00 00 02 00 00 00  |................|
000080d0  20 00 5f 5f 6d 68 5f 65  78 65 63 75 74 65 5f 68  | .__mh_execute_h|
000080e0  65 61 64 65 72 00 5f 6d  61 69 6e 00 5f 70 72 69  |eader._main._pri|
000080f0  6e 74 66 00 00 00 00 00  00 00 00 00 00 00 00 00  |ntf.............|
00008100  fa de 0c c0 00 00 01 92  00 00 00 01 00 00 00 00  |................|

| | 바이너리 | 텍스트 |
|---|---|---|
| 읽기쉬운가 | X | O |
| 데이터를 절약하기 좋은가 | O | X |
| 파싱하기 쉬운가 | X | ? |
| 바이트 오더를 신경써야 하는가 | O | X |

# 요구사항

# 요구사항

1. 누가 봐도 알만큼 단순해야함

# 요구사항

1. 누가 봐도 알만큼 단순해야함
2. 요청/응답 구조가 아니여야함

# 요구사항

1. 누가 봐도 알만큼 단순해야함
2. 요청/응답 구조가 아니여야함
3. 연결 지향형

# 요구사항

1. 누가 봐도 알만큼 단순해야함
2. 요청/응답 구조가 아니여야함
3. 연결 지향형
4. 형식 제한이 없어야함

# 구조

HTTP와 유사하게 Header영역과 Body영역으로 구성되어있습니다.

# Header

Header의 데이터는 아래와 같이 표현됩니다.
\r\n을 한번 더 붙여 헤더의 끝을 표현합니다.

{key1}={value1}\r\n
{key2}={value2}\r\n
\r\n

아래 3가지는 필수 키 입니다.
version: 버전
method: 어떤 동작을 원하는지에 대한 값입니다.
bodyLength: Body의 사이즈입니다. (Byte)

예시

```
version=1
method=command
bodyLength=28
```

# Body & Example

Body의 데이터는 헤더에 명시한대로 보내집니다.

```
version=1
method=command
bodyLength=28
```

Header의 Body 예시

```
{"command": "getJpegImage"}
```

예시 2
Header

```
version=1
method=JpegImage
bodyLength=286720
```

Body

```
이미지 데이터
```

# 감사합니다.