우산, 파란색, 남자, 보드이(가) 표시된 사진

자동 생성된 설명

WireShark Project

Junseop Lim

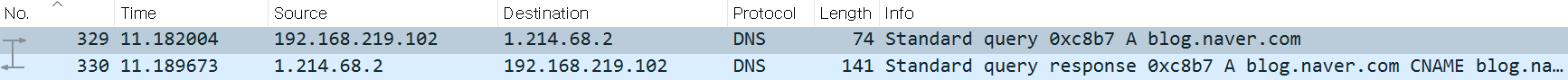
***1. Summarize the content of DNS request response message.***

***a) Analyze the transport layer segment to determine which transport layer to user and the port number***

> DNS Requset 패킷

**스크린샷이(가) 표시된 사진

자동 생성된 설명**



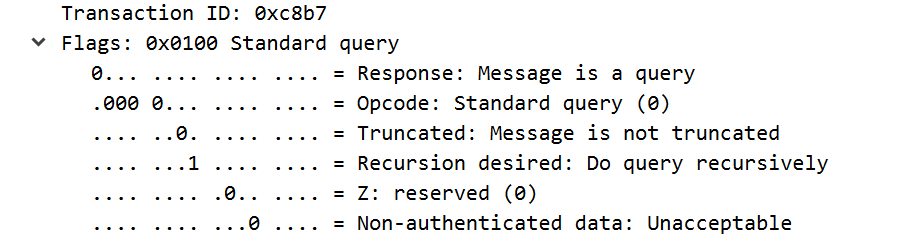


DNS 요청 패킷을 분석해보면 IPv4를 사용하며 **클라이언트(192.168.219.102)**가 DNS(1.214.68.2)로 쿼리를 보낸 것을 확인할 수 있다.



UDP를 사용했으며, 이때 클라이언트는 임의의 포트 (49922)를 사용하고, 목적지엔 DNS의 **포트인 53**을 사용한 것을 확인할 수 있다.

> DNS 헤더



DNS 헤더의 Flags 필드는 쿼리 특성을 정의하는 수많은 필드로 구성되어 있다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

위 그림의 ID 밑의 8가지가 Flags로 OR, Opcode, AA, TC, RD, RA, Z, RCODE로 구성되어 있다. 위의 DNS 요청 패킷을 참고해서 순서대로 분석하면

**- OR(Query / Response)**

OR은 패킷이 요청하는 패킷(0)인지, 응답하는 패킷(1)인지 표시하는 비트이다. 캡쳐에서 메시지가 쿼리이므로 response 값에 0이 표시된 것을 확인할 수 있다.

**- Opcode**

Opcode는 쿼리의 유형을 지정하는데, 보통은 0000을 포함하고 있다.

**- Truncation**

응답이 길어서 잘렸는지에 대해 알려주는 비트인데, 보통은 잘리는 경우가 없어 거의 0비트로 표시되고 캡쳐에서 또한 0으로 표시되어 있다.

**- Recurison Desired**

재귀를 사용하는지 안사용하는지를 알려주는 비트로 대부분의 DNS는 재귀 쿼리를 사용하기에 1로 표시되어 있다.

**- Z**

나중을 위해 예약된 필드로 항상 0으로 설정되어 있다.

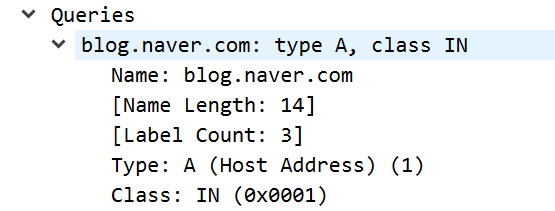
**- Authoritative Answer**

도메인 이름에 대해 믿을 수 있는 서버로부터의 응답임을 표시한다.

오렌지, 검은색, 사람들, 하얀색이(가) 표시된 사진

자동 생성된 설명

Flags 밑에는 Questions, Answer RRs, Authority RRs, Additional RRs 항목이 존재하는데 Questions는 몇개의 질문인지 질문의 개수를 표시한다. 보통 한 패킷당 한개의 질문을 가지기 때문에 캡쳐에서도 1로 표기되어 있다. Answer RRs, Authority RRs, Additional RRs는 각 세션의 개수를 표시한다.



Question Section은 가변 길이 필드로 DNS Query의 이름을 정의하며 위에서부터 [ Name – Type – Class] 순의 구조로 이루어져 있다. Name에는 DNS에 요청한 도메인의 네임, 호스트 네임이 들어가고, blog.naver.com이 request(요청)한 도메인의 네임인 것을 알 수 있다.

Type은 쿼리의 유형을 나타내는데 A일 경우 호스트의 주소를 나타낸다.

Class는 네트워크의 클래스 타입을 표시하며, IN은 Internet을 의미한다.

> DNS Response 패킷



쿼리 패킷과는 반대로 이번에는 목적지였던 서버가(1.214.68.2) 클라이언트를 목적지로(192.168.219.102) 요청에 대한 응답을 보낸다. UDP를 사용한다는 것을 알 수 있으며 DNS의 port 넘버인 53번이 출발지 포트인 것을 확인할 수 있다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

응답 패킷의 Flags 구조는 요청 패킷의 것과는 구조가 다른데, 추가된 필드를 확인해보면

**- Authoritative**

도메인 이름에 대해 믿을 수 있는 서버로부터의 응답인지를 표시한다.

**- Recursion Available**

응답에서 정의된 재귀가 사용가능한지를 표시한다.

**- Reply Code or Response Code**

응답에서 오류가 존재하는지 표시한다. 이 비트에는 0에서 5까지의 숫자가 오는데, 0번은 오류가 없다는 것을 뜻한다.

***2. Check TCP connection setup process before HTTP.***



HTTP는 TCP를 이용하고, TCP는 connection oriented 방식이기 때문에 TCP connection establishment(3-way handshake) 과정을 HTTP message를 주고받기 전에 거쳐야 한다. 총 3단계로 이루어지며 처음에 client가 DNS로부터 얻은 IP address(210.97.143.41)로 SYN(0) flag가 활성화된 TCP message를 보내고, 웹 서버는 SYN(0)과 ACK(1) flag가 활성화된 TCP message를 보내며 마지막으로 클라이언트는 다시 ACK(1) flag가 활성화된 message를 보내며 TCP connection이 establish되고 이후 HTTP message를 주고받게 된다. TCP는 IP, 802.3 Ethernet를 거쳐 전송이 되며 포트 넘버는 HTTP를 위한 것이므로 80이다.

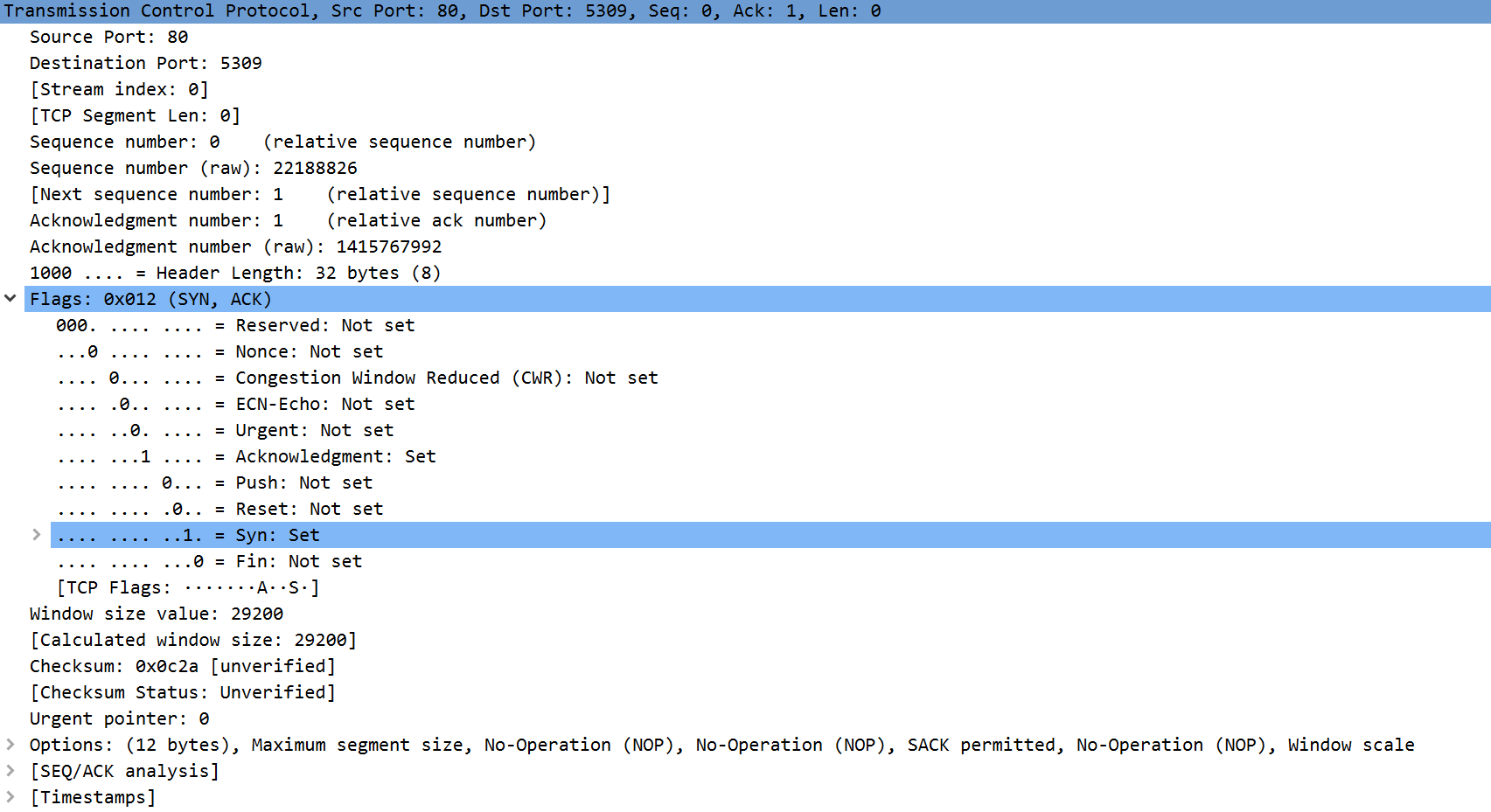
**(1) SYN(0) message**

스크린샷이(가) 표시된 사진

자동 생성된 설명

source(192.168.219.102)의 5467 port에서 Destination(103.24.8.9)의 80 port(http로 고정)로 [SYN] flag가 활성화된 segment를 보낸다. SYN은 client가 server와 connection을 맺고 싶을 때 사용하며 이때 SYN은 1로 set 되어있는 상태이다. Sequence number는 0이며 ACK number도 0이다. Window size도 명시되어 있는데 64240였으며, Option에 MSS도 1460으로 명시되어 있었다. 이는 MTU 1500에서 TCP/IP header length 40을 빼낸 값이다.

**(2) SYN(0),ACK(1) message**



destination(103.24.8.9)의 80 port에서 source(192.168.219.102)의 5309 port 로 SYN과 ACK flag가 활성화된 segment를 보낸다. 이때 ACK flag는 client 측의 SYN에 대한 응답으로 1로 set 되어 있어 다음에 client가 이 segment를 받을 때 알맞은 연결이 되었음을 알려주고, SYN flag는 server또한 client와 연결을 맺기 위해 보내는 것으로 1로 set되어 있다. ACK num은 client가 보낸 seq num에 1을 더한 1로 되어있음을 확인할 수 있다.

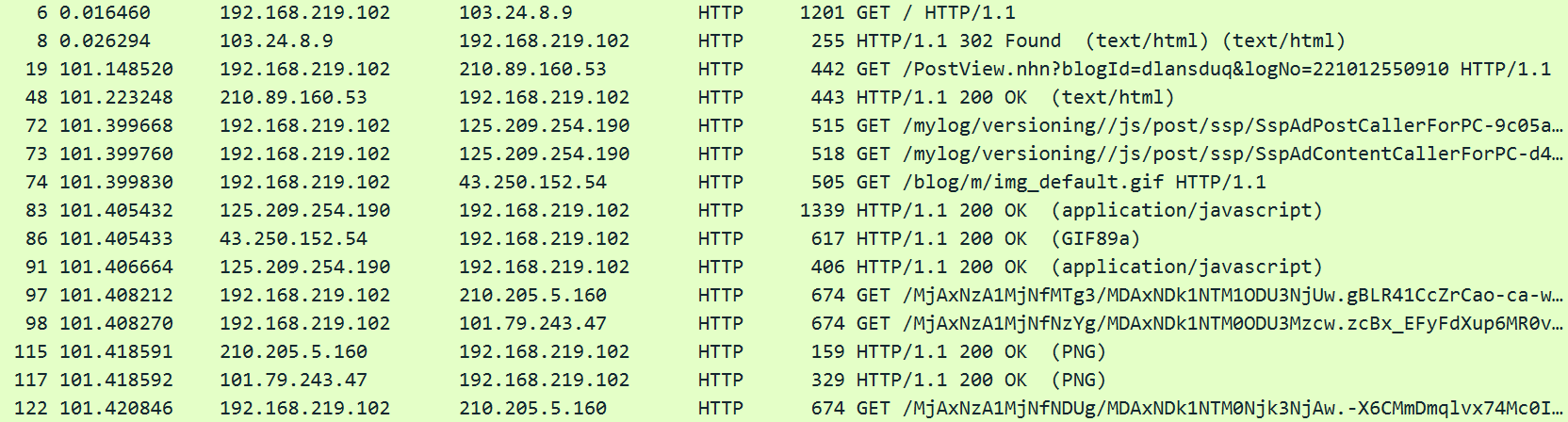
**(3) ACK(1) message**

스크린샷이(가) 표시된 사진

자동 생성된 설명

source(192.168.219.102)의 50334 port에서 Destination(103.24.8.9)의 80 port로 ACK flag가 활성화된 segment를 보낸다. 이때 ACK는 client 측의 SYN에 대한 응답으로 1로 set 되어 있어 다음에 server가 이 segment를 받을 때 알맞은 연결이 되었음을 알려주며, seq num은 서버로부터 받은 ACK num이 그대로 1로 되어있으며 ACK num은 서버측 seq num에 1을 더한 1로 되어있다. 이러한 3-way handshake 과정을 통해 TCP connection setup 과정이 마무리된다.

***3. Summerize the contents and meaning for the HTTP request and response message for the first time.***



HTTP는 웹을 위한 protocol이며 TCP, IP, 802.3 Ethernet을 거쳐 전송이 되며 포트 넘버는 80으로 고정이다. TCP를 통해 connection이 보장되어 있으므로 중간에 계속 연결을 하는 것이 아닌 연속적으로 계속 정보를 주고받는다. Client는 GET 함수를 통해 query를 하며 웹 서버는 이를 응답한다. 맨 처음에 전체 html파일을 요청하며 그 이후엔 각 URL들을 요청하게 된다. HTTP response message를 분석해 보면 connection이 항상 keep-alive 상태인데, 이는 server가 request를 보내더라도 계속 connection을 open 상태로 두는 것을 의미하며, persistent HTTP임을 나타낸다. 그러므로 GET query가 연속적으로 나타나는 것을 확인할 수 있다. 정상적으로 데이터 교환이 이루어 질 경우 response message에서 status code가 200 OK임을 확인할 수 있었다.

**(1) request message**

스크린샷이(가) 표시된 사진

자동 생성된 설명

GET : specified resource를 요청하는 함수

blog.naver.com\r\n: URL

HTTP/1.1 : version

\r\n : carriage return, line feed

Host: blog.naver.com\r\n 서버의 도메인 네임

Connection : 현재 연결에 대한 control options와 hop-by-hop request fields – keep alive 이므로 persistent HTTP

User-Agent : user agent 정보 (브라우저) - chrome

Accept : content types

Referer : HTTP 요청을 시도한 페이지의 URL이다. 어떤 페이지를 통해 현재 페이지로 들어왔는지에 대한 정보를 제공한다.

Accept-Encoding : acceptable한 encodings 정보

Accept-Language : accept하고싶은 human language(한국-한국어)

마지막 \r\n : end of header lines

<entity body> empty with the GET method but is used with the POST method.

(2) response message

스크린샷이(가) 표시된 사진

자동 생성된 설명

Version : HTTP/1.1

Status Code : 200 OK (request succeeded, requested object later in this message)

이 외에 301 Moved Permanently, 400 Bad Request, 404 Not Found, 505 HTTP Version Not Supported 등이 있다.

- 응답 헤더

Date : 웹서버의 현재 시간을 나타낸다. 실제 시간과 다를 수 있다.

﻿﻿﻿Server : 응답을 담당하는 웹서버의 종류를 나타낸다.

﻿﻿﻿Cache-Control : 브라우저 측의 캐싱 여부를 결정한다.

﻿﻿﻿Set-Cookie : 사용자의 브라우저에 쿠키값을 설정한다.

Content-Type : 응답의 mime-type을 나타낸다.

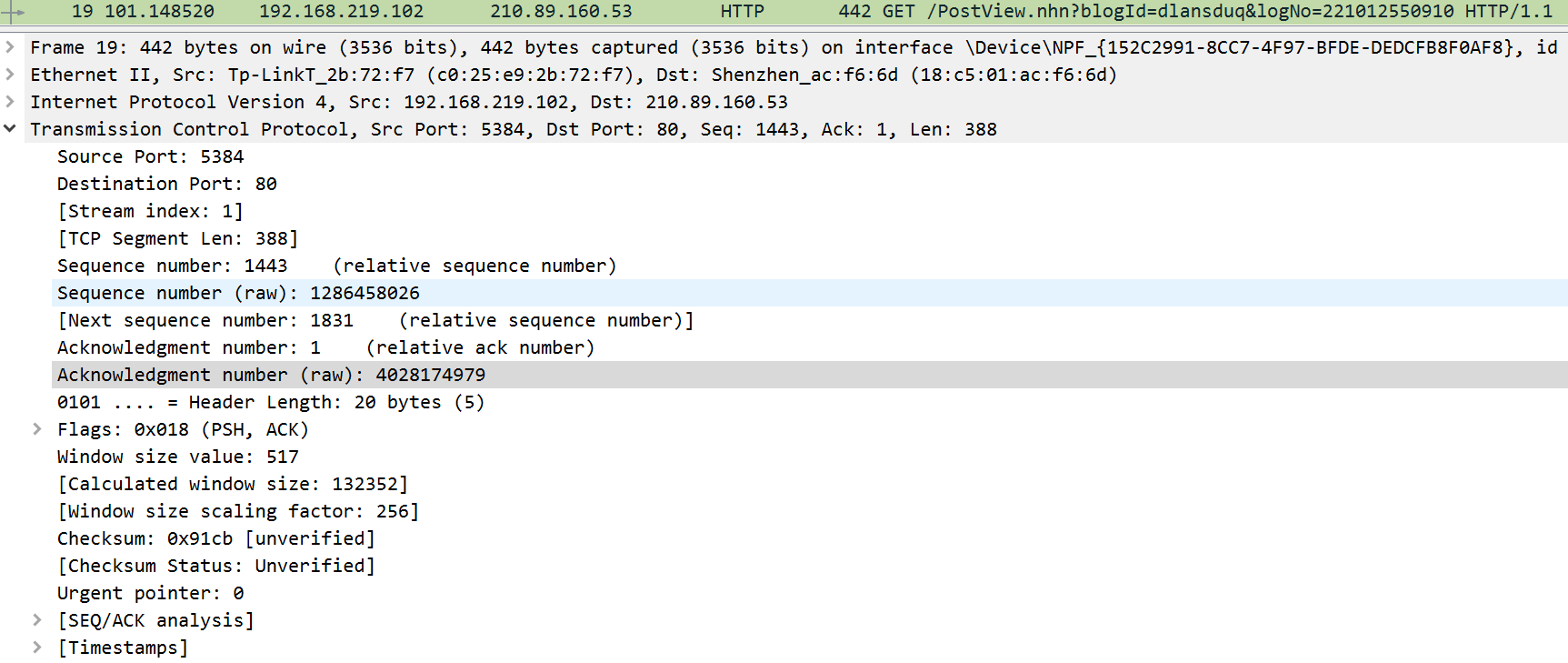
Connection : 소켓 연결을 유지할지 끊을지를 결정한다.

Location : 302 응답이 올 경우, 리다이렉션 할 페이지의 URL을 나타냅니다

***4. Data transfer process***

***a) Sequence, ACK, Number***

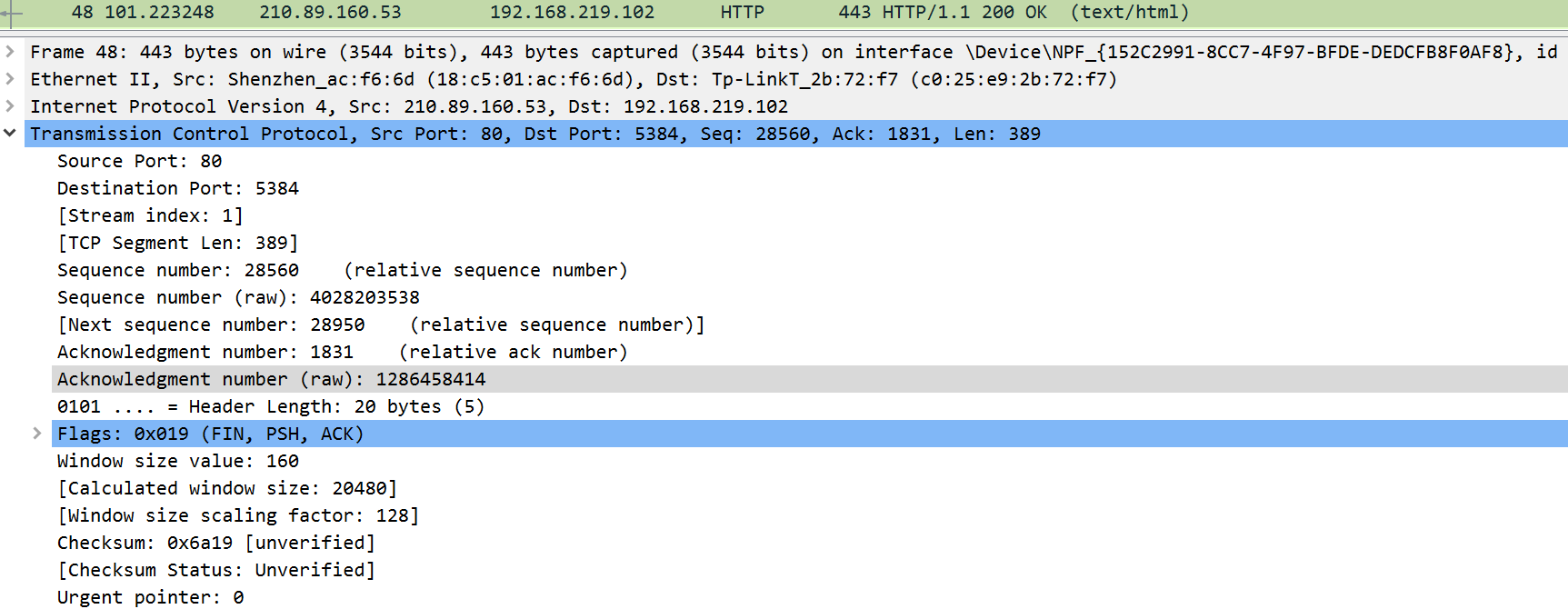
(1) Client -> Server 에게 보내는 데이터



- seq : 1443 , Ack :1, Len : 388

- 지금 보내는 데이터는 seq가 1443 이고 길이는 388이다. 서버는 1번부터 보내 달라고 ack 1 전송.

(2) Server -> Client, (1)에 대한 응답 데이터

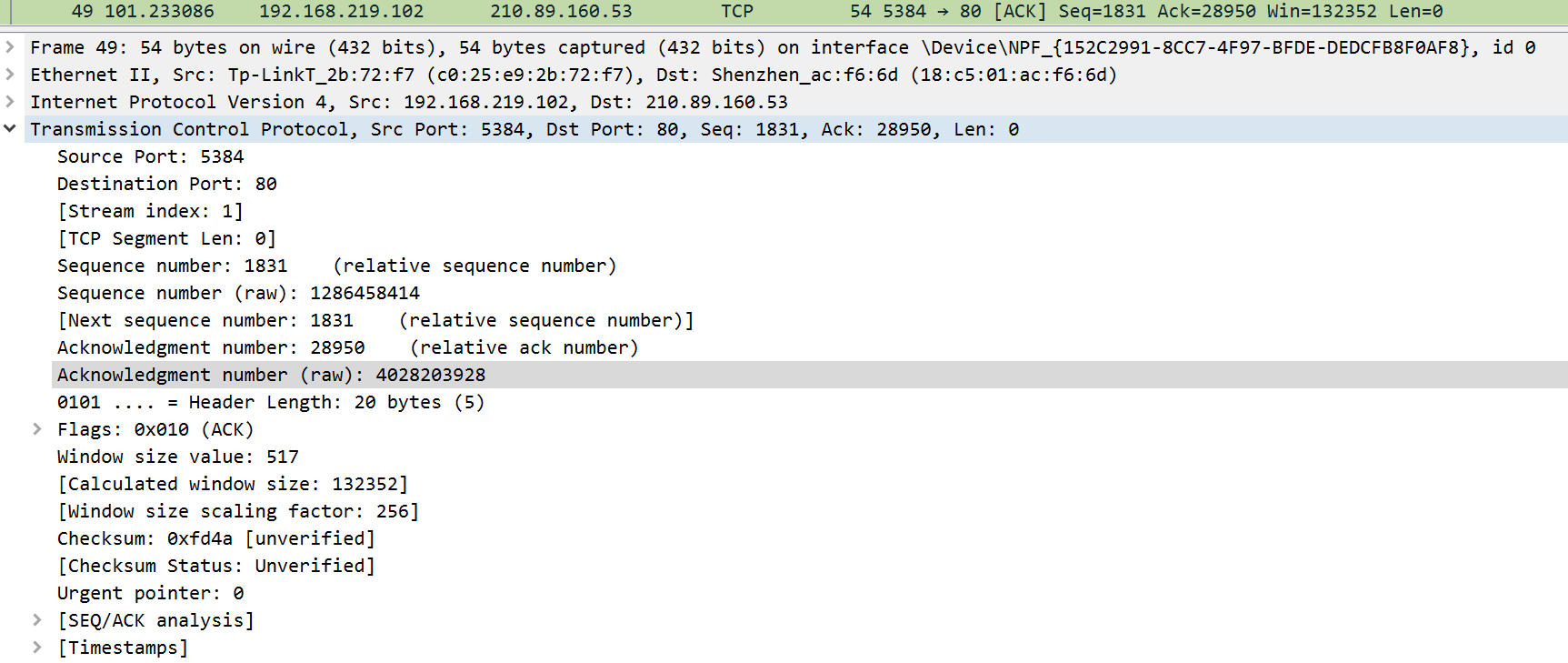


- seq : 28560, Ack : 1831, Len : 389으로 Server 가 Client 에게 응답을 보냈다.

- 서버는 클라이언트에게 지금 보내는 데이터는 28560 번 데이터라고 알려준다. (seq: 28560) 길이는 389.

- 다음에 클라이언트가 보낼 데이터 번호는 Ack(1831)이다. [(1)에서 온 데이터길이 388 + (1) Seq값 3 (1443 +388) 이다. 이 말은 즉, 클라이언트로부터 온 388 byte의 데이터를 서버가 모두 잘 받았다는 것을 의미한다.]

(3) Client -> Server 로 (2)에 대한 응답데이터



- seq : 1831, Ack: 28950, Len : 0

- 클라이언트는 서버에게 1831부터 보내고 데이터 길이는 0이라고 신호를 준다.

- 다음에 서버로부터 받을 Number은 28950부터이다. ( Ack : 28950, (2)에서 받은 데이터길이(389) + #2의 Seq(28560))

- 서버가 보낸 데이터길이 389를 모두 받았으니 서버는 클라이언트에게 28950부터 보내면 된다.

(4) 통신종료

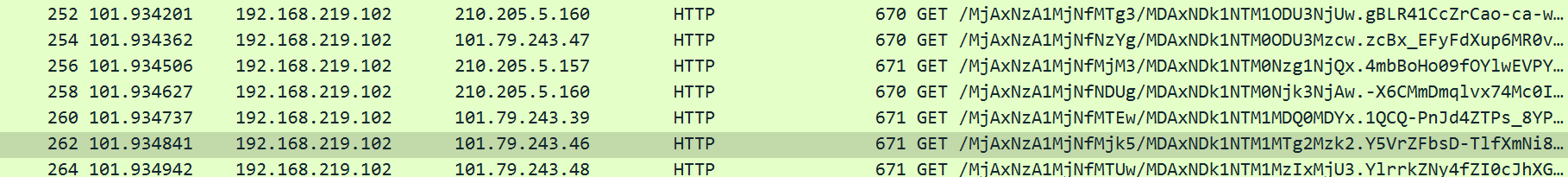


통신이 완료가 되면 [FIN]을 주고 받으며 통신을 종료하게 된다.

***5. Analyze whether persistent or non-persistent connection.***

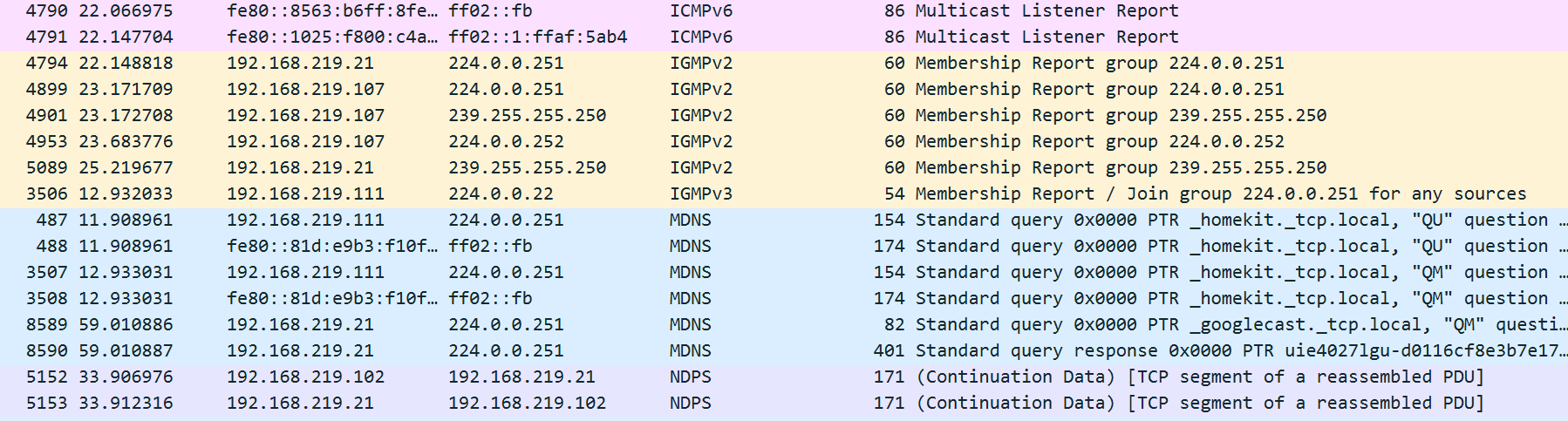


HTTP response message를 분석해보면 connection이 항상 keep-alive 상태인데, 이는 server가 request를 보내더라도 계속 connection을 open상태로 두는 것을 의미하며, 이는 persistent HTTP임을 나타낸다.



또 다른 방법으로 persistent HTTP임을 알 수 있는데, #252~264에서 보이듯이 client가 multiple HTTP request를 parallel하게 보내고 있음을 볼 수 있으며 이는 persistent HTTP의 특징이다.

***6. Write what protocols other than HTTP and DNS appear.***



(1) ICMP

ICMP는 TCP/IP에서 IP 패킷을 처리할 때 발생되는 문제를 알려주는 프로토콜이다. IP에는 오로지 패킷을 목적지에 도달시키기 위한 내용들로만 구성되어 있다. 따라서 정상적으로 목적지 호스트에 도달하는 경우에는 IP에서 통신이 성공하고 종료되므로 아무런 문제가 없다

(2) IGMP

인터넷 그룹 관리 프로토콜(Internet Group Management Protocol, IGMP). 인터넷 그룹 관리 프로토콜은 호스트 컴퓨터와 인접 라우터가 멀티캐스트 그룹 멤버십을 구성하는 데 사용하는 통신 프로토콜이다. 특히 IPTV와 같은 곳에서 호스트가 특정 그룹에 가입하거나 탈퇴하는데 사용하는 프로토콜을 가리킨다

(3) MDNS

Multicast Domain Name System(mDNS)는 로컬 네트워크 영역에서 설정없이(zero configuration) 호스트 이름을 찾기 위해서 사용하는 서비스이다. Unicast Domain Name System(DNS)와 유사한 프로그래밍 인터페이스 그리고 패킷을 사용한다. 소형 네트워크 환경에서 별도의 네임서버를 사용하지 않고 호스트를 찾을 수 있다.