**Computer Network Project 2**



학과 전자공학과

학번 12181457

이름 김범수

제출일 23.06.14

**주제 및 연결 환경**

이번 프로젝트의 주제는 네트워크 분석 툴인 Wireshark를 이용하여 컴퓨터 네트워크에서 배운 다양한 종류의 패킷 확인 및 분석이었다.

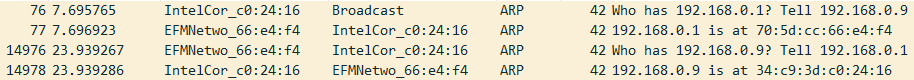
연결한 인터넷 환경 정보는 아래와 같다.



IP 주소는 192.168.0.9로 DHCP로 자동으로 할당되어 있으며 MAC주소는 34-C9-3D-C0-24-16으로 할당되어 있는 것을 확인할 수 있다.

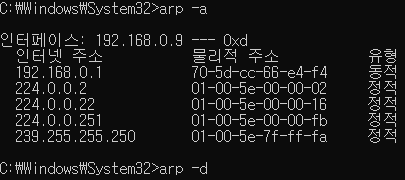
**프로토콜 분석**

1. **ARP**



Cmd에서 arp -a 로 ARP Table을 조회하고 arp -d로 Tabel을 초기화 하는 과정을 Wireshark로 Capture한 것이다.

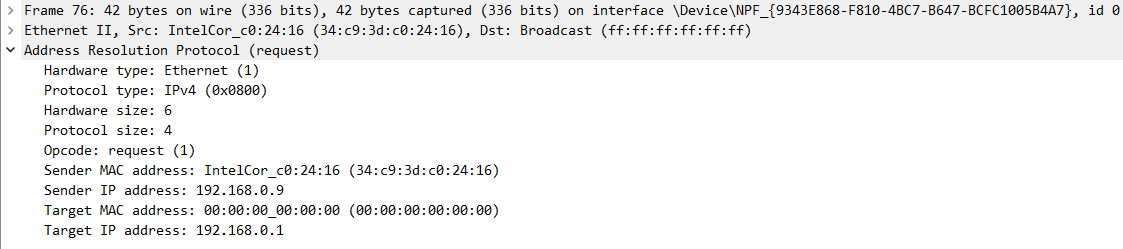
먼저 MAC주소를 알아내기 위해 Broadcast를 진행하고 그에 반응하여 Reply하면서 Table을 채우는 모습을 확인할 수 있었다.



위 그림은 cmd 창에 나타난 ARP Table이다.

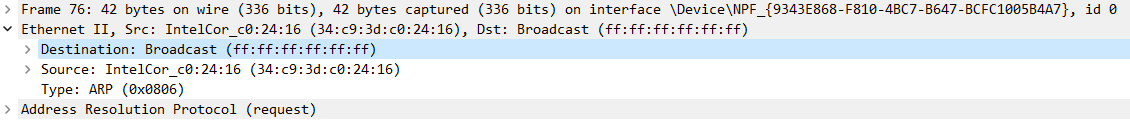
내 IP주소는 동적으로 설정되어 있고 나머지 Table은 정적으로 설정되어 있는 것을 확인할 수 있다.

* 1. ARP Request



위 그림을 보면 Address Resolution Protocol (request)로 나타나 있는 것을 확인할 수 있다.

Opcode 또한 request(1)로 해당 Packet이 request를 진행하고 있다는 것을 보여준다.

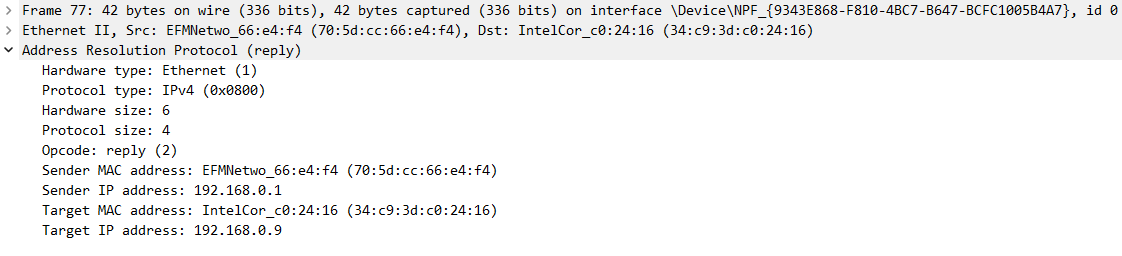


현재는 Destination의 MAC Address를 알 수 없다.

따라서 Target MAC address가 모두 1로(f로) 초기화 되어있는 것을 확인할 수 있다.

ARP request는 Destination의 주소가 Frame의 Data 부분에 넣고 ARP Query를 Broadcast하게 되고 이 Request Packet을 받은 Node들은 Packet의 IP Address를 보고 자신에게 요청된 Request인지 판단하고 Reply하게 된다.

* 1. ARP Reply



위 그림은 ARP Request에 대한 Reply를 보여준다.

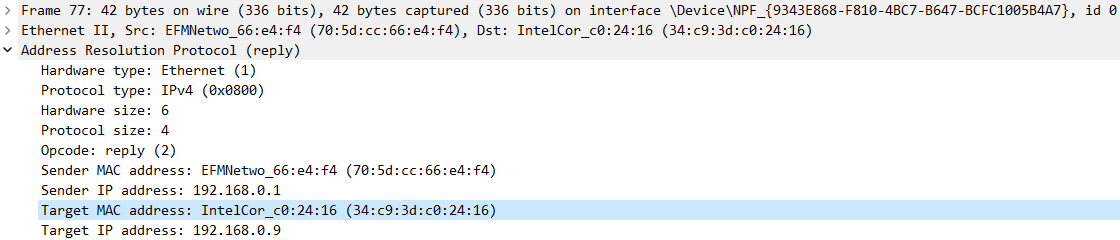
Address Resolution Protocol (reply)를 통해 reply임을 알 수 있다.

또한 Opcode도 reply(2)로 나타나 있는 것을 볼 수 있다.

ARP Request를 받은 Node가 본인에게 요청된 Request임을 확인하고 Reply하는 것으로 본인의 MAC Address가 담긴 ARP reply Packet을 생성하고 Request를 요청한 Source Node에게 보내게 된다.

이때 Reply를 진행하는 Node는 Request를 통해 Reply를 보내야하는 Destination의 MAC Address를 이미 알고 있기 때문에 Broadcast가 아닌 Point to Point로 전송하게 된다.

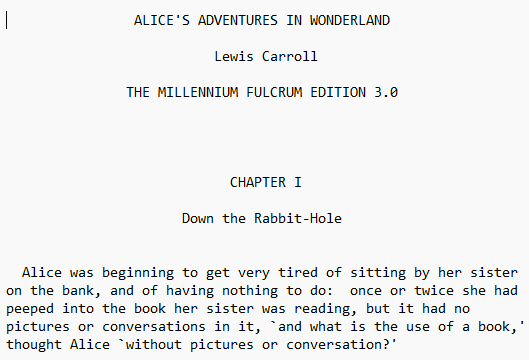
아래 그림을 통해 위 동작이 진행되고 있음을 확인할 수 있다.



Request에서 Source였던 IntelCor\_c0:24:16의 MAC Address와 IP가 Reply에서는 Destination으로 사용되고 있는 것을 확인할 수 있다.

1. **TCP**

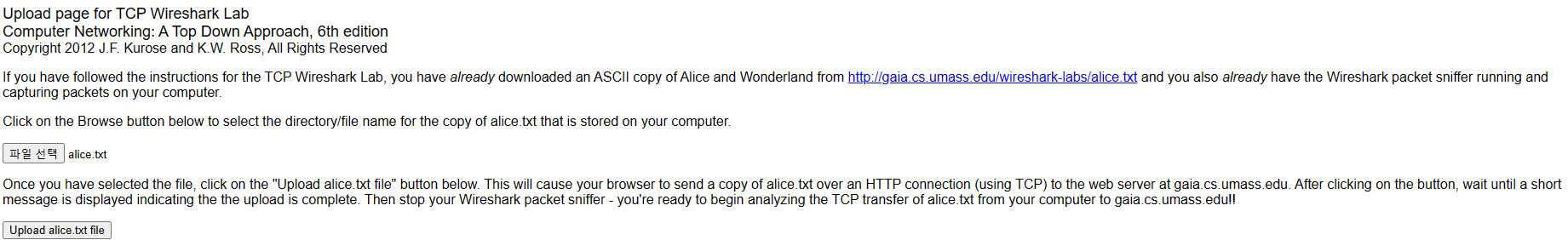
2-1) Alice 작성



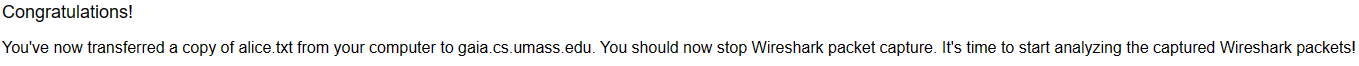
튜토리얼에 따라 Alice를 위와 같이 다운받아 준비하였다.

2-2) <http://gaia.cs.umass.edu/wireshark-labs/TCP-wireshark-file1.html> 사용

* 전송 전



* 전송 후



이후 TCP Packet을 관찰하기 위해 위의 지정된 사이트에 aclice.txt를 보내는 과정을 Wireshark로 Capture하였다.





위 그림은 웹 서버와 주고받은 TCP Packet을 Capture한 것이다.

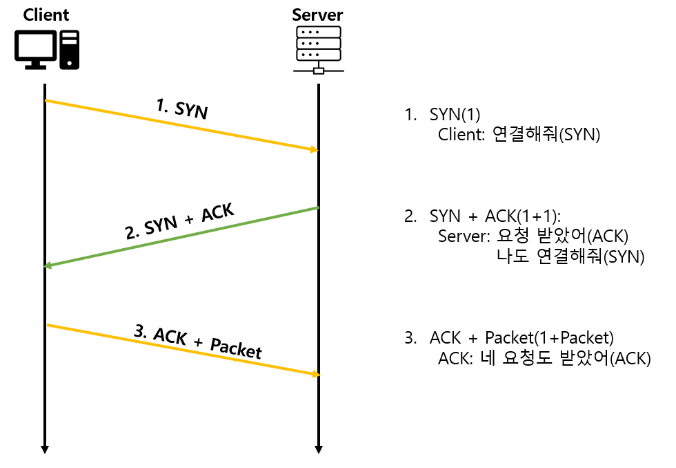
모든 Packet은 12895 <-> 80 Port에서 교환되고 있다.

80번 포트는 웹 서버가 사용중인 Port이므로 내 Port는 12895인 것을 알 수 있다.

맨 왼쪽 index에서 6번과 10번 11번을 통해 3way hand shake 과정을 확인할 수 있었다.

먼저 Client가 연결을 위해 SYN를 보내고 Server가 이에 대한 SYN+ACK을 보낸다.

Client는 ACK을 보내면서 3way hand shake를 완성한다.





위 그림은 서로 전송된 Packet의 일부를 가져온 것이다.

Sequence 값이 이전 Sequence 값에 Length를 더해진 것만큼 변화하고 ACK은 Sequence 값에 맞춰 전송하는 것을 확인할 수 있었다.

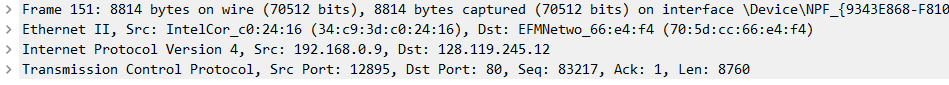


이렇게 전송된 Data를 홈페이지에 HTTP 프로토콜을 사용해서 POST하는 것을 확인할 수 있었다.



FIN, ACK을 보내 연결을 종료하는 것을 확인할 수 있다.

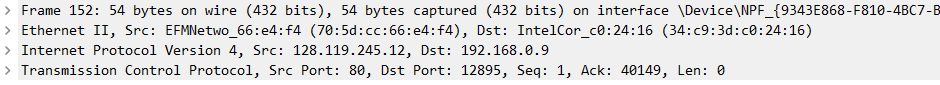
2-3) User -> Server Packet



User가 보내는 Packet은 매우 큰 크기를 가지고 있는 것을 확인할 수 있다.

이는 데이터를 전송하고 요청하는 내용들이 들어있기 때문에 상대적으로 큰 크기를 가질 수 밖에 없다.

2-4) Server -> User Packet

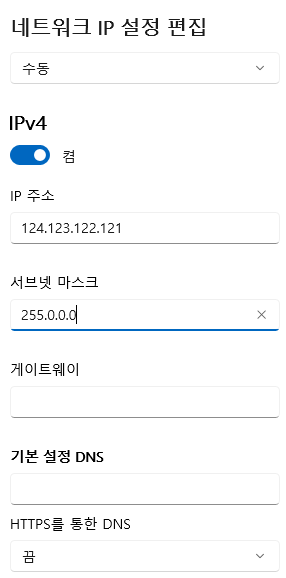


반면 Server가 Uset에게 보내는 Packet은 매우 작은 크기를 가지는 것을 확인할 수 있다.

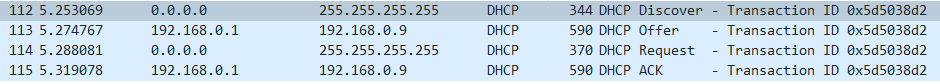
이는 Server는 ACK을 보내는 역할을 하면서 굉장히 간단한 역할을 하기 때문에 이런 결과가 나온 것이다.

1. **DHCP**

DHCP 프로토콜를 확인하기 위하여 IP를 Static으로 Hard Matching 하였다가 다시 Dynamic으로 변경하였다.

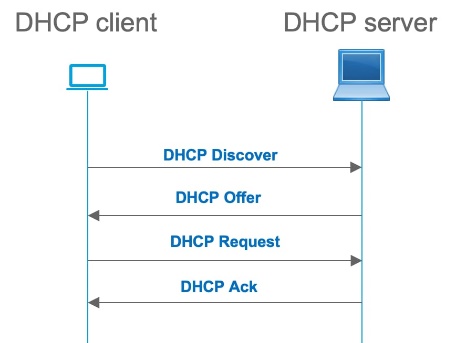
->

위 그림처럼 먼저 Static으로 임의의 IP 주소를 할당하였다가 자동(DHCP)으로 할당하여 DHCP가 어떻게 동작하는지 확인해보았다.



위 그림은 DHCP로 변경하였을 때 생성된 Packet이다.

Discover->Offer->Request->ACK 순서로 발생한 것을 확인할 수 있다.

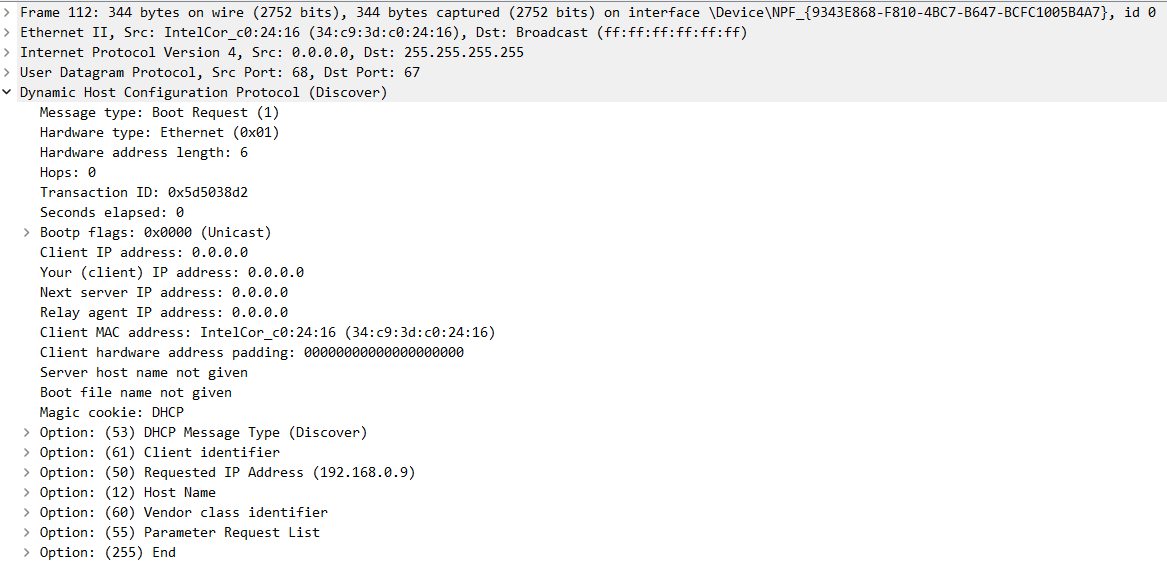


또한 모든 DHCP 프로토콜로 생성된 Packet은 UDP를 사용해 전송되는 것을 확인할 수 있었다.

모든 Packet은 아래와 같이 동일하게 UDP로 전송되고 있는 것을 확인할 수 있다.



3-1) Discover



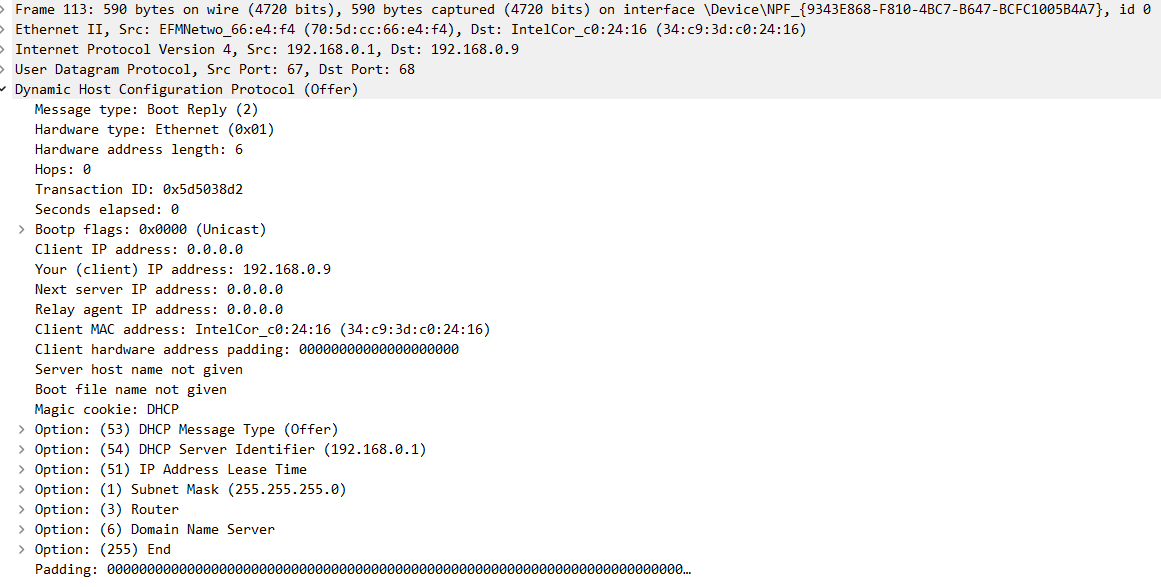
위 그림은 Client가 IP 주소를 할당 받기 위해 DHCP Server에 요청한 Discover 메시지다.

현재 IP가 없는 상태이므로 Source IP는 0.0.0.0으로 설정되어 있고 68번 Port를 사용하는 것을 확인할 수 있다.

Destination은 IP가 255.255.255.255, 66번 Port로 설정한 것을 확인할 수 있다.

이는 Broadcast를 하고 66번 Port를 사용하는 건 DHCP로 규정되어 있기 때문에 DHCP가 이 요청을 받고 응답할 수 있게 만들기 위해서이다.

3-2) Offer



Offer는 DHCP Server가 Client에게 보내는 Packet으로 사용가능한 IP 주소를 제안하는 Packet이다.

Discover에 대한 응답으로 Transaction ID가 동일한 것을 확인할 수 있다.

이 Transaction ID는 Client와 Server간에 메시지를 식별하고 요청과 응답을 매칭하는 역할을 한다.

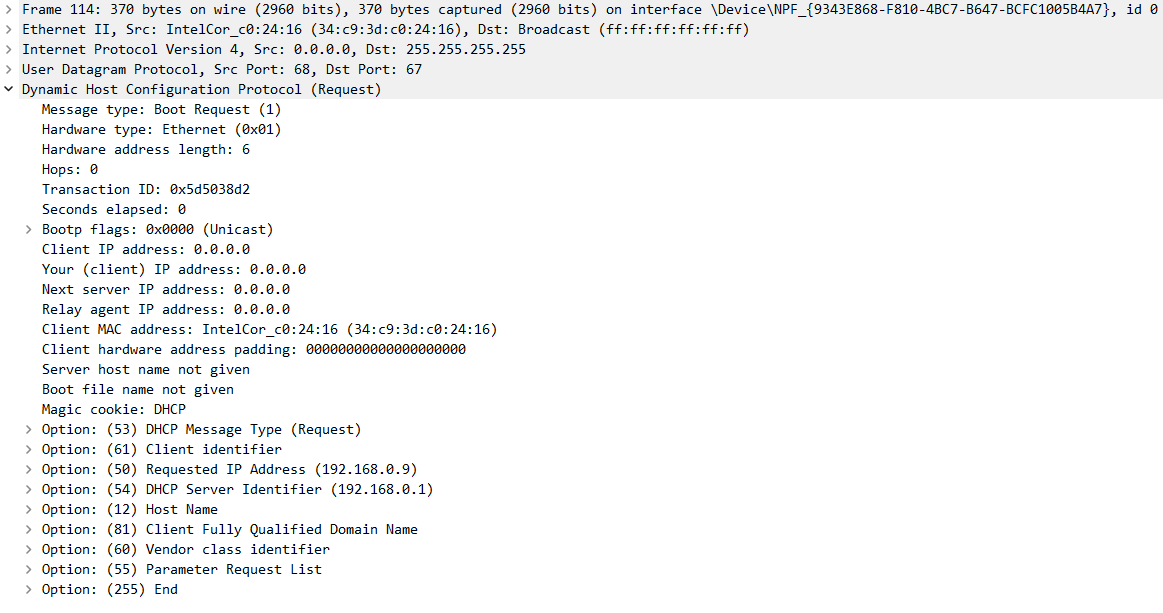
위 Packet의 내용을 보면 Source의 IP 주소가 192.168.0.1이고 Destination의 주소가 192.168.0.9인 것을 확인할 수 있다.

이를 통해 DHCP Server의 IP 주소는 192.168.0.1이고 제안한 주소는 192.168.0.9이다.

이는 이론적인 동작과 약간의 차이를 보였다.

원래 DHCP Offer 단계에서는 Destination IP를 255.255.255.255로 하고 Port를 68로 하지만 위와 같은 결과로 차이를 보였다.

3-3) Request



Request는 제안받은 IP를 Client가 승낙하고 사용하겠다는 의미를 가지고 있다.

위 그림을 통해 Option: (50) Requested IP Address (192.168.0.9)를 보면 Client가 제안받은 IP 주소인 192.168.0.9를 사용하겠다는 의미를 가지고 있는 것을 확인할 수 있다.

하지만 이 Packet의 Source와 Destination 주소는 여전히 각각 0.0.0.0과 255.255.255.255로 설정되어 있는 것을 확인할 수 있다.

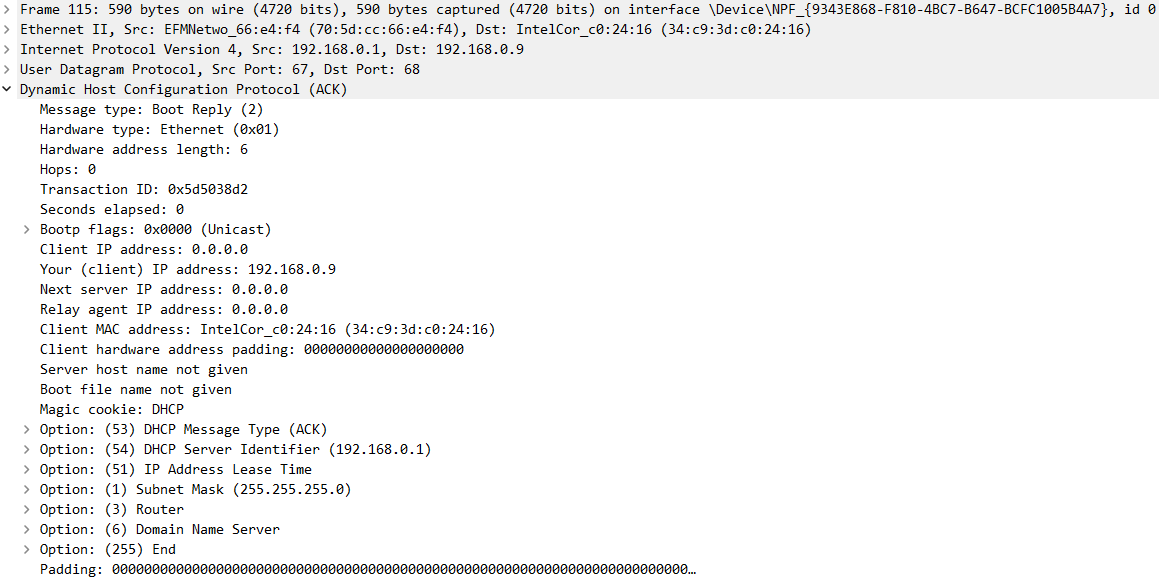
이는 아직 IP를 제안 받은 상황으로 따른 DHCP Server에게 어떤 IP를 사용할지 알리지 못했기 때문에 IP를 설정할 수 없기 때문이며 이에 따라 Broadcast를 진행하기 위해 255.255.255.255를 목적지로 하여 Packet을 날리는 것이다.

또한 Transaction ID가 동일했다.

이는 이론적인 내용과 약간의 차이를 보였다.

이론적으로 Request의 Transaction ID와 달라야 하지만 동일한 결과를 보였다.

3-4) ACK



이는 Request에 대한 ACK으로 동일한 Transaction ID를 가지는 것을 확인할 수 있다.

1. **HTTP**

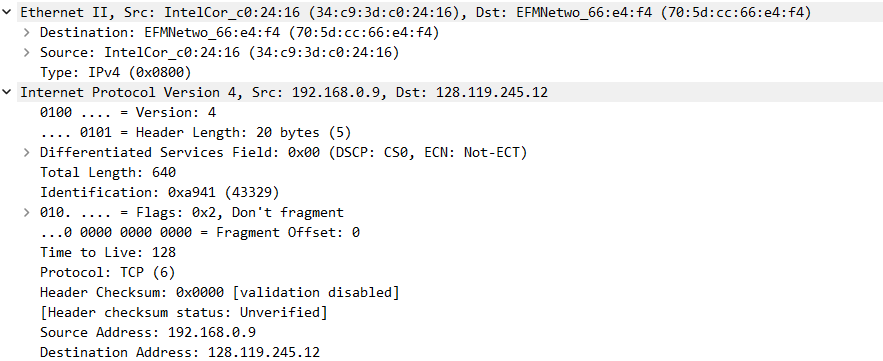
먼저 HTTP의 Packet을 분석하기 위해 튜토리얼의 [gaia.cs.umass.edu/wireshark-labs/INTRO-wireshark-file1.html](http://gaia.cs.umass.edu/wireshark-labs/INTRO-wireshark-file1.html) 사이트를 방문해 Packet을 캡쳐했다.





GET에 대한 Packet을 분석해보았다.

frame의 크기와 Packet의 전송 시간 등에 대한 정보를 얻을 수 있었다.

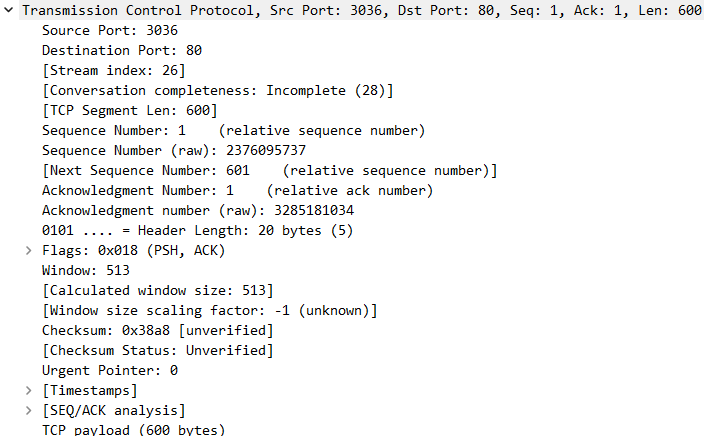


다음으로는 ID 값과 IPv4로 동작한다는 사실과 HTTP는 TCP를 기반으로 동작한다는 사실을 확인할 수 있었다.

ID 값은 Packet을 쪼개서 보내야 할 때 하나의 Packet이었다는 사실을 나타내주는 값이다.

또한 TTL (Time to live) 값이 128로 설정되어 있다는 것을 확인할 수 있었고 Source Address와 Destination Address를 확인할 수 있었다.

또한 Header에 대한 Checksum값이 존재함을 확인할 수 있었다.



위 내용에서는 Port값을 무엇을 사용했는지 확인할 수 있었고 TCP Segment의 길이에 대해 알 수 있었다.

또한 Sequence 값과 Ack 값, 어떤 packet인지 알려주는 Flag값, checksum 등에 대한 정보를 알 수 있었다.



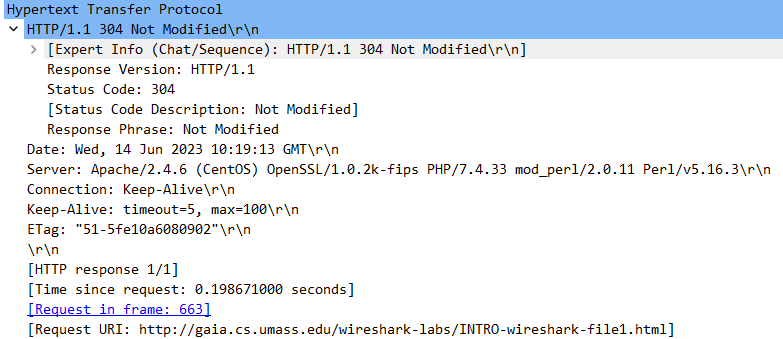
맨 윗줄은 HTTP Command, 버전 url 등을 나타내고 있다.

HTTP Command는 GET으로 데이터를 가져오는 Request를 보냈다는 것을 알 수 있다.

또한 사용중인 HTTP 버전은 1.1 버전임을 알 수 있었다.

Host의 url과 Encoding 방식, Language 등에 대한 정보를 알 수 있었다.

다음으로는 response에 대해 분석해보았다.



HTTP 버전과 Status Code라는 결과를 상태를 알려주는 세자리 코드가 들어있는 것을 확인할 수 있었다.

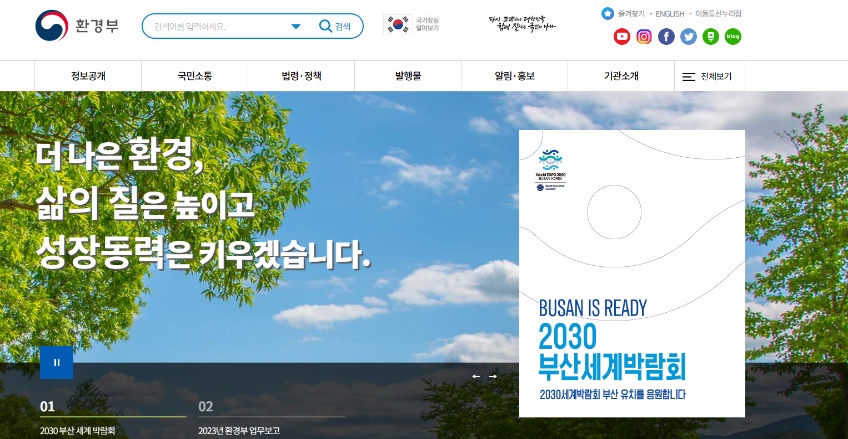
변경되지 않았다는 의미로 이미 접속한 적이 있어 캐시에서 정보를 가져왔다는 뜻이다.

다음으로 timeout 시간과 server 정도 등에 대해 알 수 있었다.

4-1) text with image

먼저 HTTP를 분석하기 위해 HTTPs 프로토콜을 사용하지 않는 사이트를 검색해야 했다.

HTTPs를 사용하는 사이트에서는 보안이 향상되어 Wireshark에 packet이 capture되지 않았기 때문이다.

(<http://me.go.kr/home/web/main.do>)

이번 프로젝트에서는 환경부 사이트를 사용하였다.

텍스트, 스크린샷, 폰트, 메뉴이(가) 표시된 사진

자동 생성된 설명

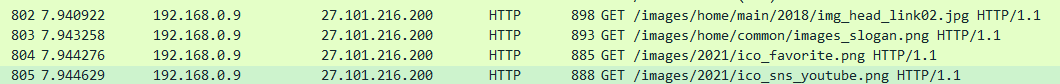
위 그림은 먼저 환경부 사이트에 접속한 그림이다.

링크를 통해 접속하여 Packet을 분석해 보았다.



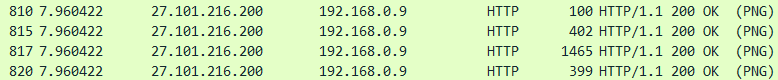
위는 text를 전송받을 때 사용된 packet이다.

HTTP 1.1 버전을 사용하여 전송되었으며 파일을 가져오는데 성공했다는 200 OK라는 값이 들어있었다.



위는 image를 받을 때 사용된 Packet으로 Client가 Server에게 요청할 때 사용된 Packet이다.

마찬가지로 HTTP 1.1 버전을 사용하고 GET을 통해 image를 가져온 것을 확인할 수 있었다.



이 packet들은 GET에 대한 요청으로 마찬가지로 HTTP 1.1 버전이 사용되었고 200 OK라는 값으로 성공이라는 것을 알려주고 있다.



위 내용을 통해 Content-Type이 image/png인 것을 확인할 수 있었다.

4-2) multimedia

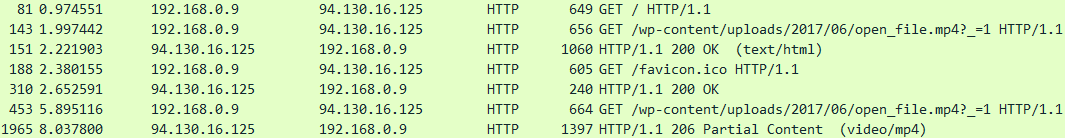
마찬가지의 이유로 https가 아닌 http를 사용하는 Packettrain이라는 사이트를 이용하였다.

텍스트, 스크린샷, 디자인이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 번호, 소프트웨어이(가) 표시된 사진

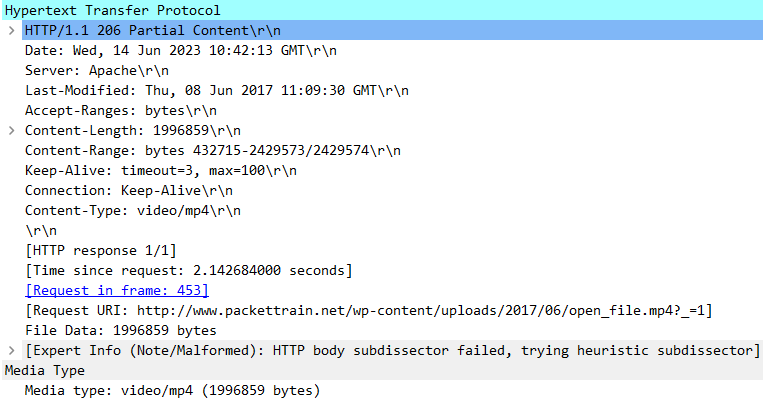
자동 생성된 설명

(http://www.packettrain.net)



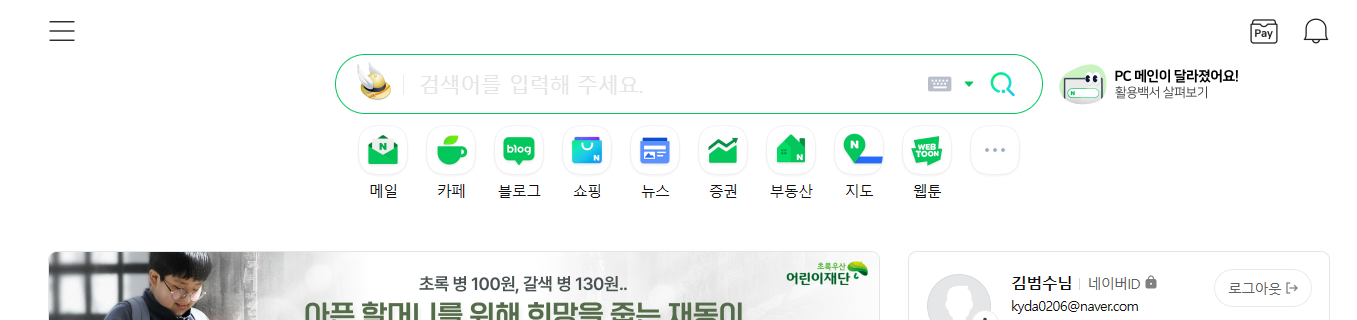
HTTP 1.1 버전을 사용하고 있음을 확인할 수 있었다.

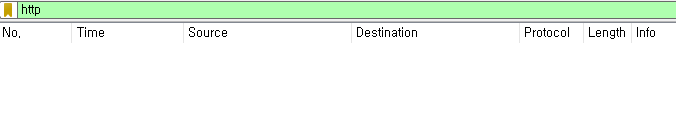
GET으로 mp4 파일을 요청하는 것을 확인할 수 있었고 이에 따라 video/mp4 파일을 받은 것을 확인할 수 있었다.



Content-Type은 video/mp4인 것을 확인할 수 있었고 Media type 또한 video/mp4인 것을 확인할 수 있었다.

4-3) famous portal homepage

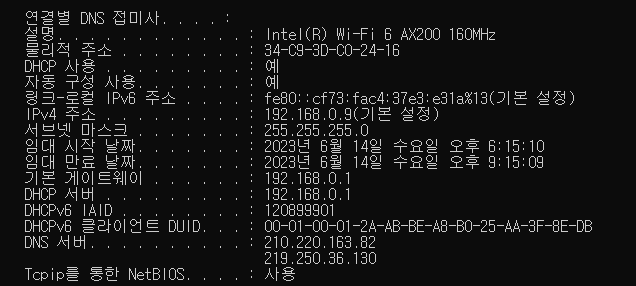




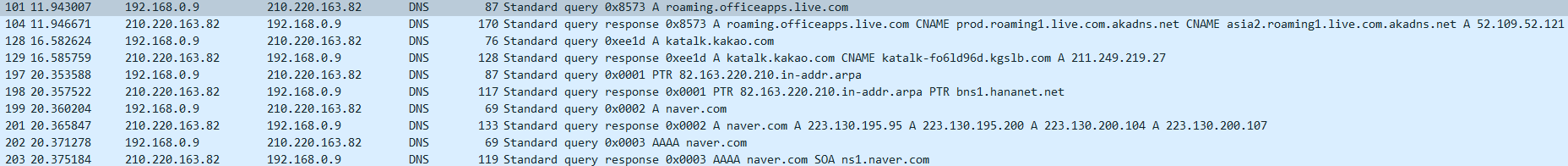
네이버와 같은 홈페이지는 https로 SSL이 추가되어 보안화 되어있기 때문에 아무 packet도 capture할 수 없게

1. **DNS**

Ipconfig -all을 이용하여 DNS 를 관찰해보았다.

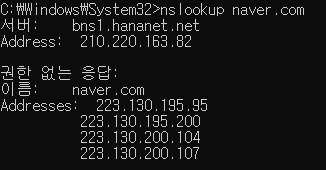


DNS 서버는 210.220.163.82와 219.250.36.130이 연결되어 있는 것을 확인할 수 있었다.



이후 nslookup naver.com을 통해 DNS를 요청하였다.

이는 DNS Server에서 domain name이나 IP 주소를 요청하는 명령어이다.



연결되어있는 DNS 서버에 naver.com에 대한 응답을 요청하였고 IP 주소를 3개 얻어왔다.

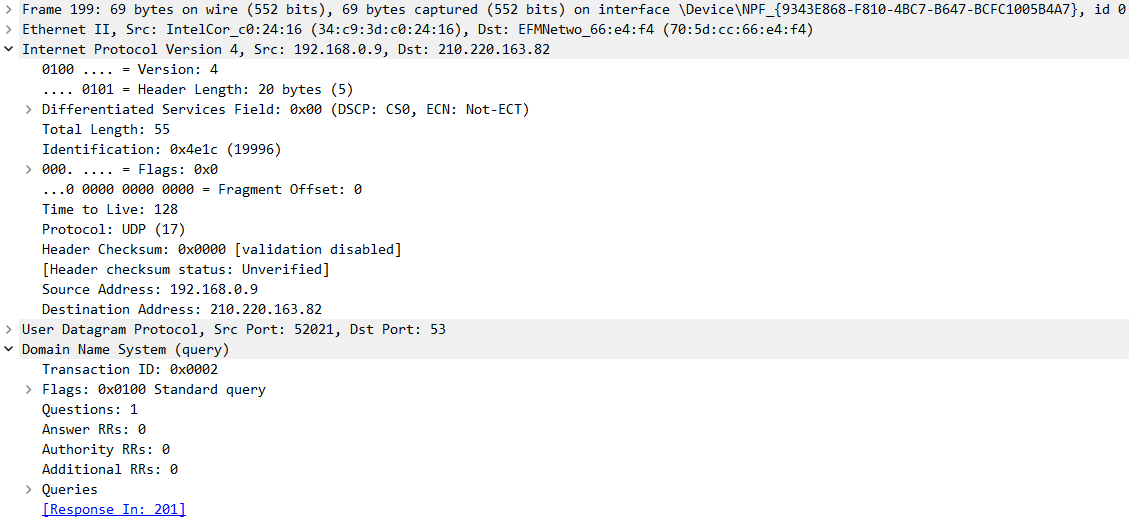
위 과정에서 발생하는 packet을 분석해보았다.

5-1) query

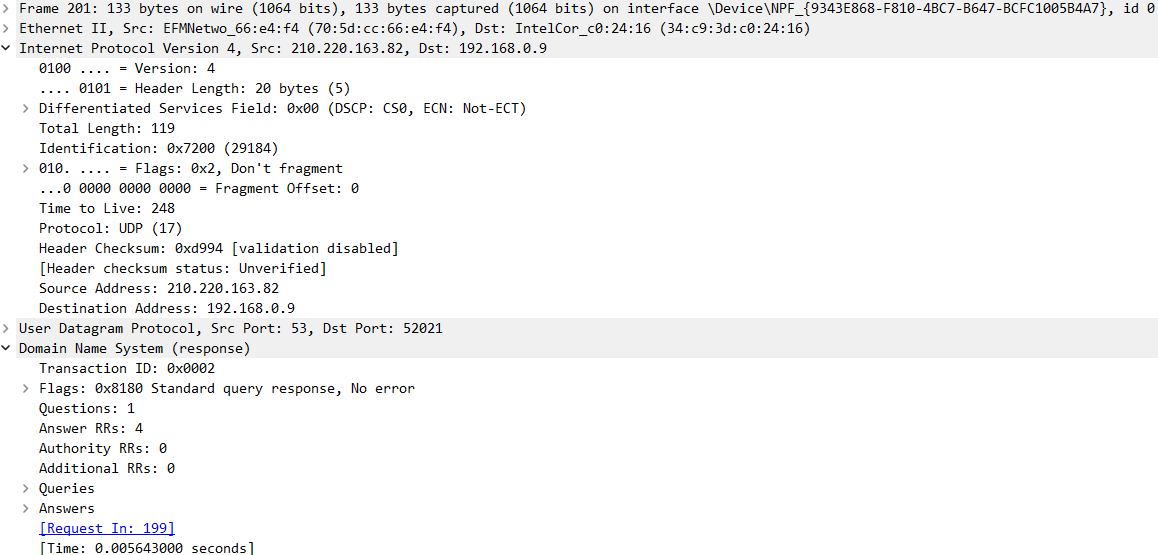
아래 그림을 통해 알 수 있듯 DNS 와의 통신은 UDP를 사용한다는 것을 확인할 수 있었다.

Source와 Destination IP는 각각 요청한 Host의 IP와 연결된 DNS Server의 IP가 들어있었다.

Transaction ID는 0x0002로 나타났으며 standard query임을 보여주었다.



5-2) reply

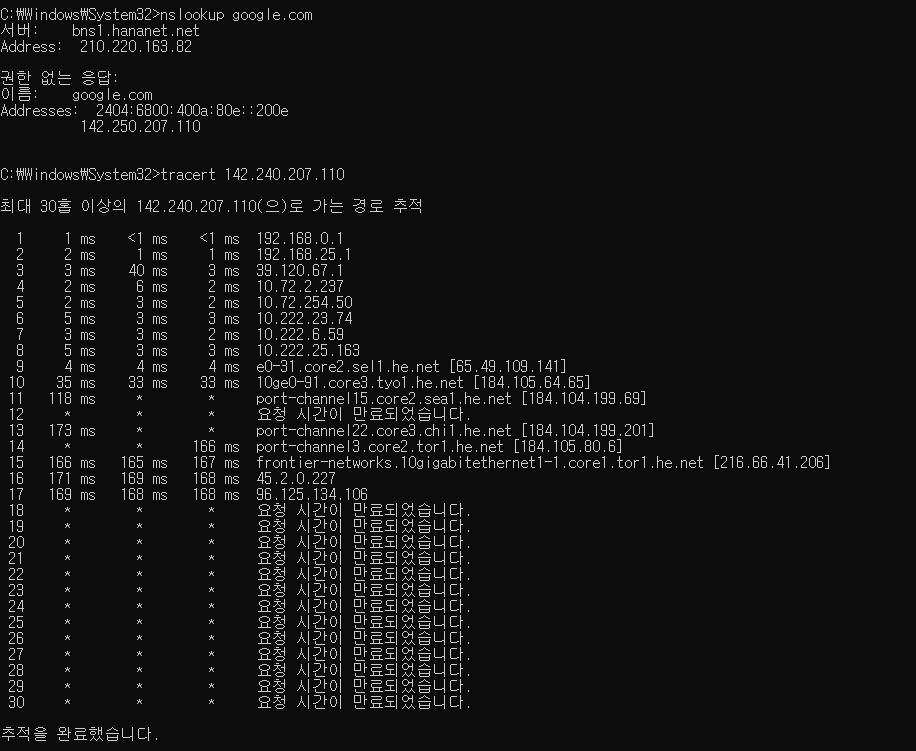


반대로 reply에서는 DNS Server의 IP가 Source로 사용되고 요청한 Host의 IP가 Destination으로 사용되었다.

마찬가지로 UDP를 사용하는 것을 확인할 수 있었고 Transaction ID는 동일하게 0x0002인 것을 확인할 수 있었다.

1. **ICMP**

Nslookup을 통해 google.com의 IP를 알아내고 알아낸 IP를 통해 tracert를 진행하여 ICMP packet을 생성해 전송해보았다.

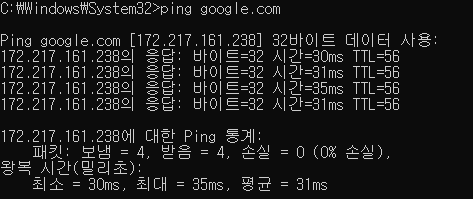


위와 같은 결과를 얻을 수 있었다.

이와 같은 결과를 얻을 수 있는 이유는 TTL을 이용해서 TTL 값이 0이 되는 지점에서 packet을 폐기하고 ICMP packet을 얻어오는 방식이다.

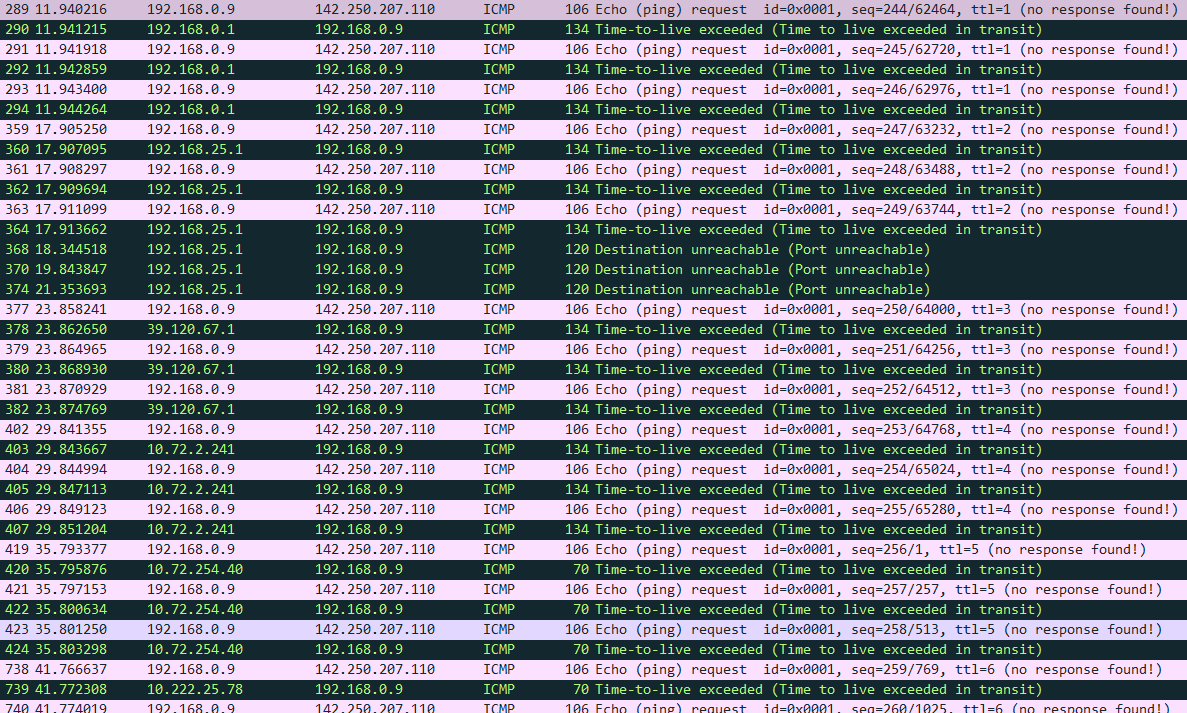
모든 router가 ICMP를 보내는 것은 아니라는 것을 확인할 수 있었다.

먼저 ping을 이용해서 google까지 잘 연결되어 있는지 확인하였다.



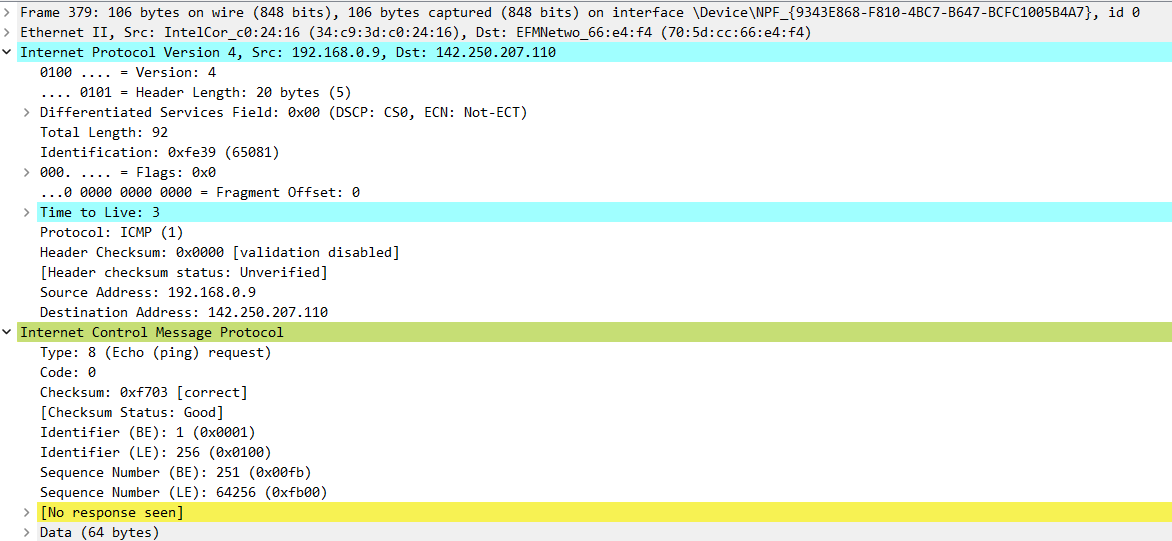
정상적으로 연결되어 있는 것을 확인할 수 있었다.

다음으로는 wireshark에서 ICMP를 분석해보았다.



위 결과를 통해 Tracert의 동작 원리를 확인할 수 있었다.

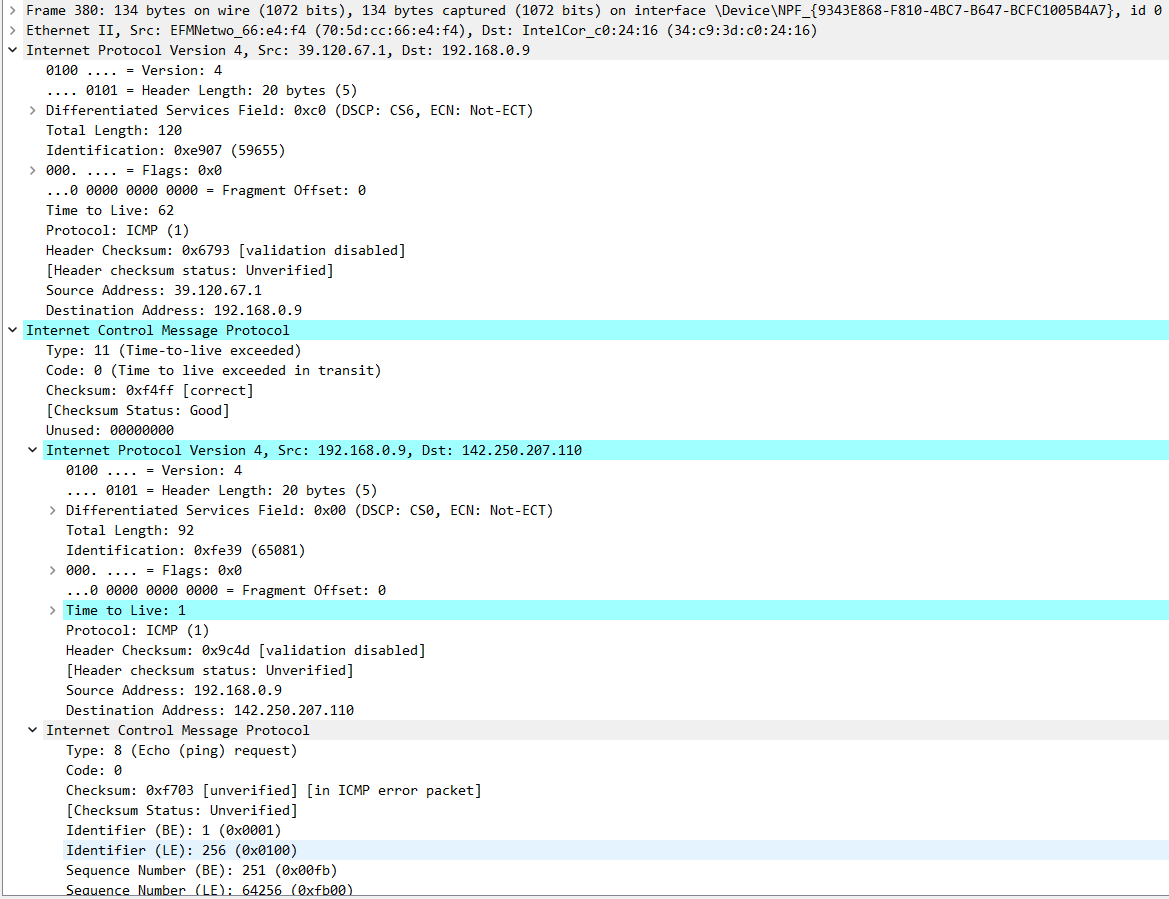
TTL을 1씩 증가하면서 packet을 3개씩 보내고 결과를 확인하는 방식으로 구현되는 것을 확인할 수 있었다.



위 그림은 TTL을 설정하고 packet을 보낼 때 packet을 분석한 것이다.

Source는 내 PC로 되어있는 것을 확인할 수 있고 Destination은 google.com으로 되어있는 것을 확인할 수 있었다.

TTL은 3으로 설정되어 있는 packet임을 확인할 수 있었고 type은 8 (Echo (ping) request)로 설정되어 있는 것을 확인할 수 있었다.



위 사진은 ICMP로 중간 router가 내 PC에게 보낸 packet이다.

이에 맞춰 source와 destination IP가 설정되어 있는 것을 확인할 수 있었다.

모든 ICMP packet은 TTL이 1로 설정되어 있는 것을 확인할 수 있었다.

이는 원래로부터 ICMP Time Exceeded 메시지를 생성하기 위해 라우터에서 폐기되었음을 나타낸다.

**출처**

[[네트워크] TCP/UDP와 3way handshake (velog.io)](https://velog.io/@rlacksals96/%EB%84%A4%ED%8A%B8%EC%9B%8C%ED%81%AC-TCPUDP%EC%99%80-3way-handshake)

[IOS XE EVPN/VXLAN에서 DHCP 구성 - Cisco](https://www.cisco.com/c/ko_kr/support/docs/switches/catalyst-9300-series-switches/217366-configure-dhcp-in-ios-xe-evpn-vxlan.html)

<http://me.go.kr/home/web/main.do>

http://www.packettrain.net