Packet Analysis: PING, HTTP, DNS

2020년 3월

경북대학교 사물인터넷표준연구실

김소용 (thdyd324@gmail.com)

요 약

인터넷에서 사용하는 가장 기본적인 프로토콜인 ICMP, HTTP, DNS 패킷을 직접 캡처하여 프로토콜들의 구조를 자세히 살펴보고 교환되는 메시지 순서를 관찰하면서, 네트워크 프로토콜이 동작하는 방식과 구조에 대해 깊은 이해를 하는 것을 목표로 한다.

목 차

1. 서론

네트워크 프로토콜에 대해 깊게 이해를 하기 위해서는 직접 패킷을 분석할 필요가 있다. 패킷 분석을 한다면 같은 프로토콜을 사용하는 개체간에 교환되는 메시지 순서를 관찰할 수 있고, 프로토콜 동작에 대한 세부 사항들을 자세히 살펴볼 수 있으며, 프로토콜에 특정한 조치를 취하고 거기에 관한 결과를 확인할 수 있기 때문에 네트워크에 대한 심도 있는 이해뿐만 아니라 더 나아가 자신만의 프로토콜을 개발하고 이를 테스트할 수 있을 것이다.

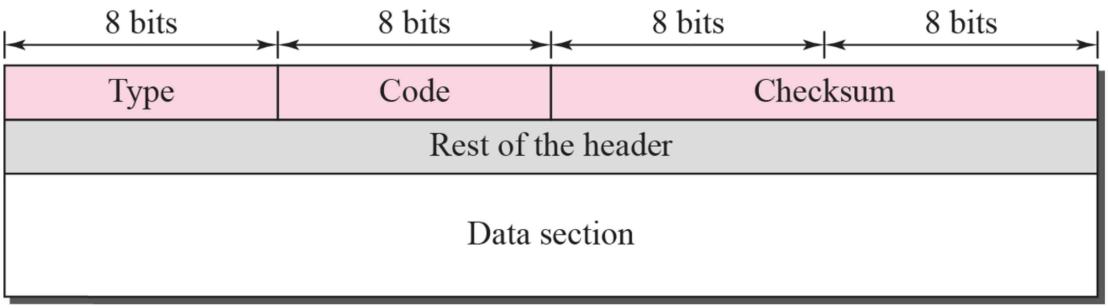
본 고에서는 패킷 분석 도구 중 하나인 Wireshark를 이용하여 패킷 분석을 할 것이며, Window 환경에서 실험을 진행하였다. Wireshark에 대한 자세한 사용 방법은 링크를 참조하길 바란다. [1] 또한, 본 고에서 분석할 패킷은 ICMP, HTTP, DNS로 인터넷에서 가장 기본으로 사용되는 패킷들을 직접 분석하여 프로토콜에 대해 깊이 이해하는 것을 목표로 한다. 2장에서는 PING 기능과 ICMP 프로토콜에 대한 소개를 하고 실제로 PING을 이용하여 ICMP 패킷을 캡처하여 분석한다. 3장에서는 HTTP 프로토콜에 대해 소개를 하고 직접 웹사이트에 방문하면서 생성되는 패킷들을 분석한다. 4장에서는 DNS와 nslookup 도구에 대한 소개 및 DNS 패킷을 분석하고, 5장에서 결론을 맺는다.

2. PING

2.1 PING 소개

PING(Packet Internet Groper)은 컴퓨터 네트워크 상태를 점검, 진단하는 기능이다. IP 기반의 네트워크에 연결된 호스트 사이의 접근을 확인하고 응답이 돌아올 때까지의 반환 시간과, 상대방 도달 경로의 혼잡 상황도 알아 낼 수 있다.

PING은 ICMP를 기반으로 동작한다. ICMP(Internet Control Message Protocol)는 IP에서 제공하지 않는 Error-Reporting 및 Error-Correcting을 제공해주기 위해 개발되었고 패킷 구조는 그림 1과 같다.

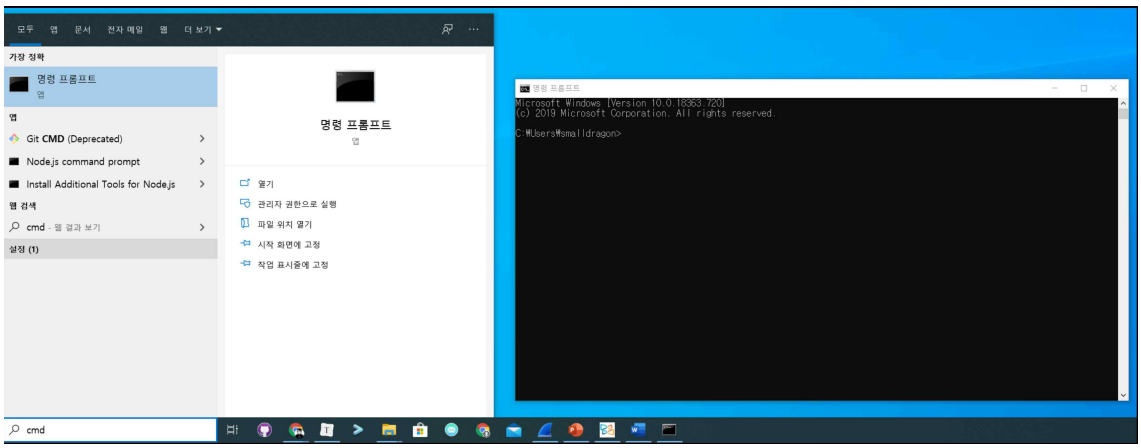


<그림 > ICMP 패킷 구조

Type은 ICMP의 여러 기능에 따른 타입을 나타내기 위해 사용하며, Code는 타입에 따라 자세한 정보를 표현할 때 사용한다. Checksum은 패킷의 오류를 체크하기 위해 사용하며, Rest of the header는 타입 별로 다른 필드를 나타낸다. 여기서 PING은 Type이 각각 8과 0인 Echo-Request, Echo-Reply를 사용한다. [2]

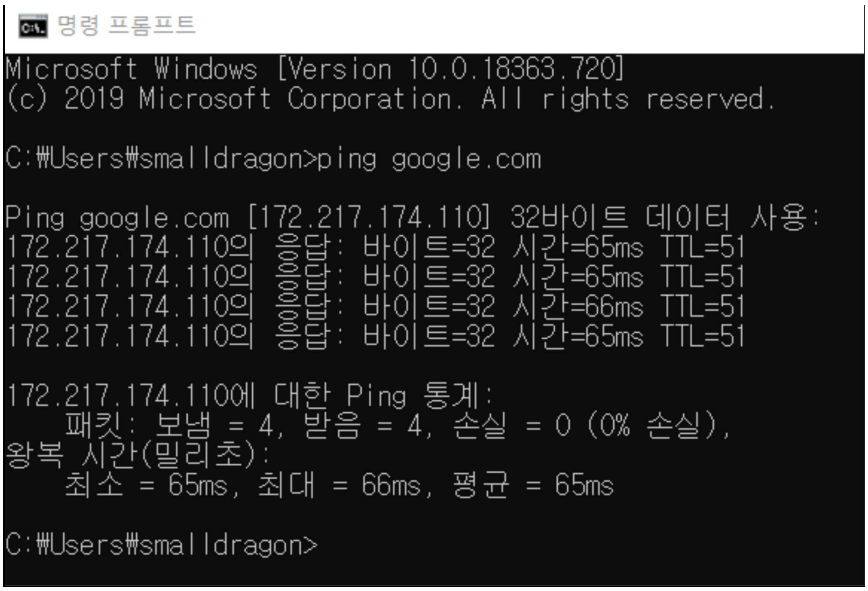
2.2 PING 사용법

Window 환경에서 PING을 사용하기 위해 명령 프롬프트를 활용한다. 먼저 윈도우 검색창에 cmd를 입력하고 명령 프롬프트를 클릭하면 그림 2와 같이 명령 프롬프트가 실행된다.



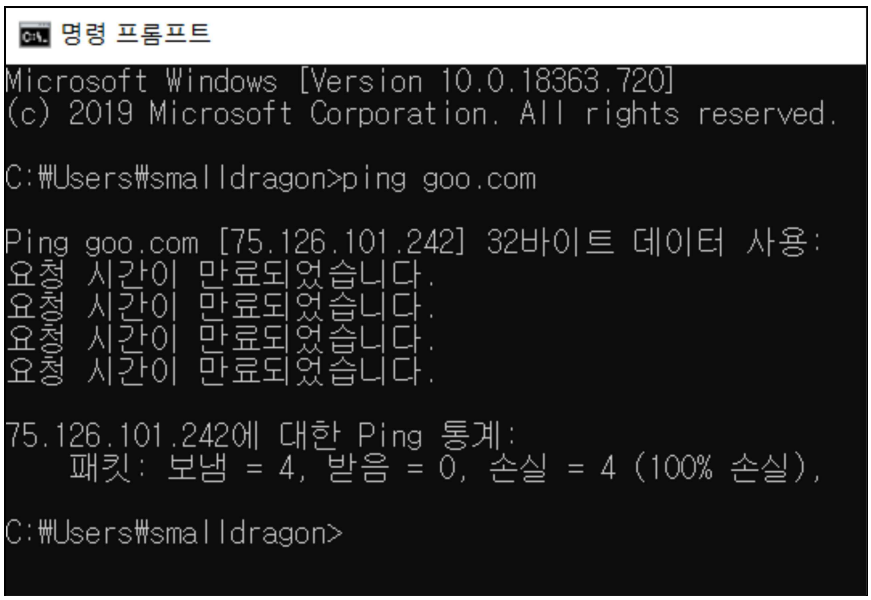
<그림 > Window의 명령 프롬프트 실행

PING을 이용하여 구글 웹 사이트에 접근할 수 있는지 확인하기 위해 그림 3과 같이 입력한다. 도달할 수 있다면 반환 시간과 함께 출력될 것이다.



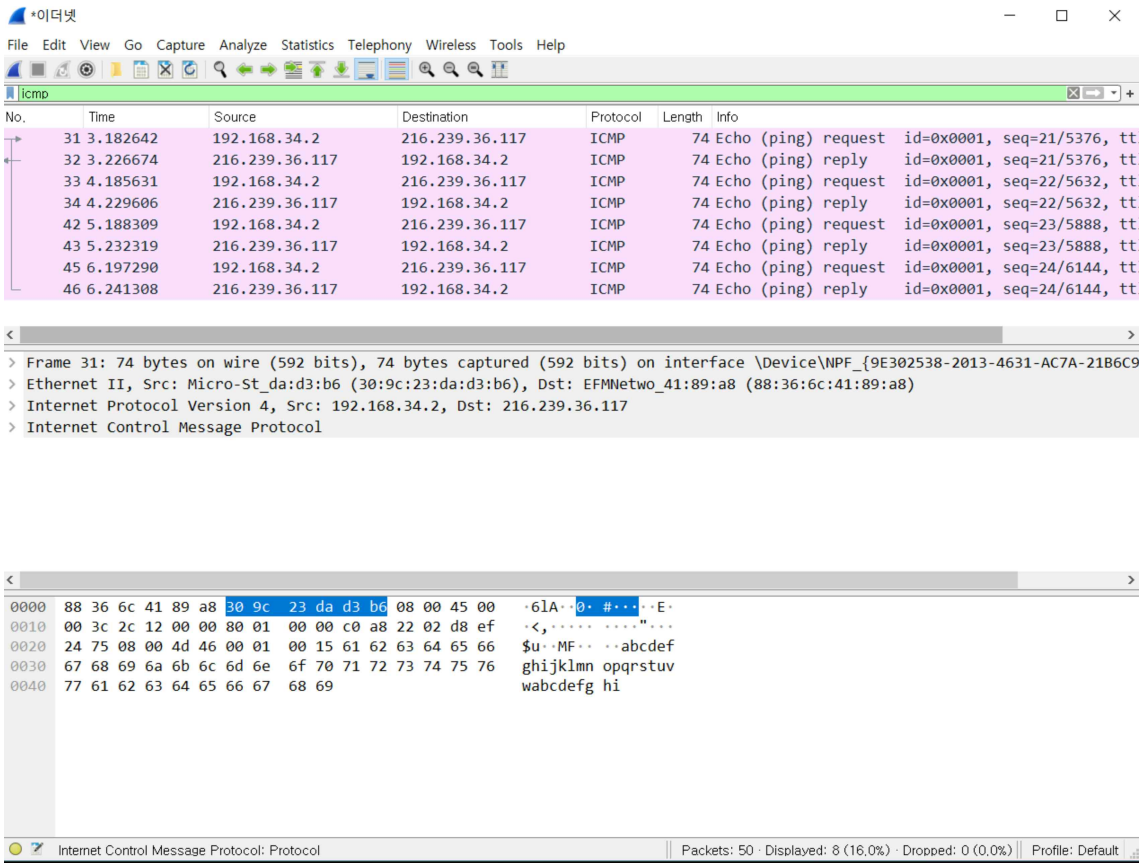
<그림 > PING을 통한 구글 웹 사이트 접근 확인

만약 특정 웹 사이트에서 PING 요청 자체를 막아 두었거나, 도달할 수 없었다면 그림 4와 같이 출력된다.



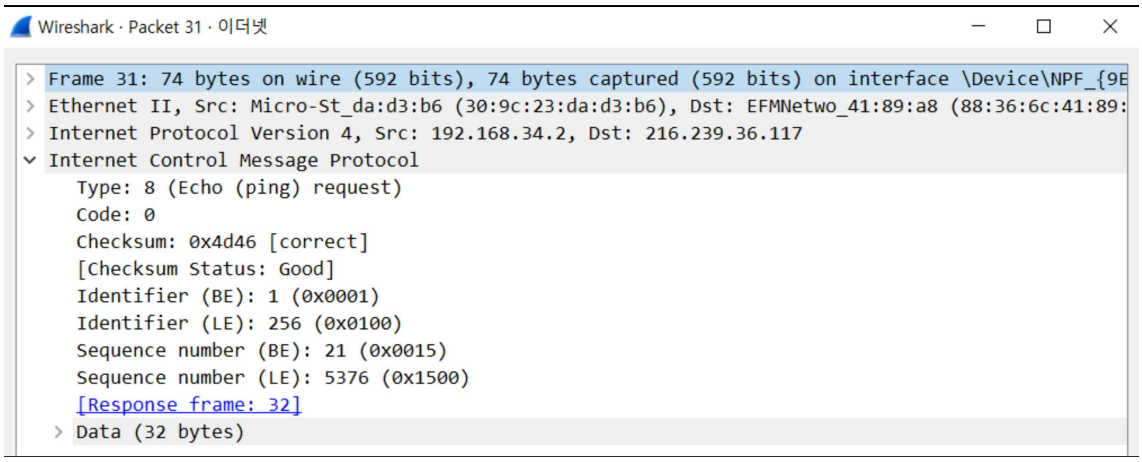
<그림 > PING으로 도달할 수 없는 경우

위 과정에서 발생한 패킷을 Wireshark로 캡처한다면 그림 5와 같이 ICMP 패킷을 확인할 수 있다.

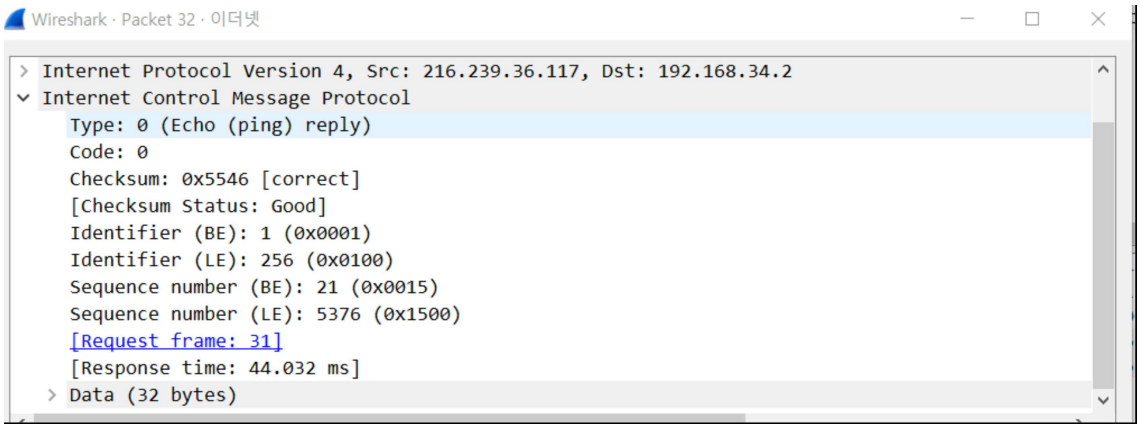


<그림 > PING을 통한 ICMP 패킷 캡쳐 화면

그림 6은 캡처한 Echo-Request 패킷의 상세한 속성을 나타내며, ICMP 패킷 구조와 비교하였을 때 서로 일치하는 것을 볼 수 있다.



<그림 > Echo-Request 패킷 캡쳐 화면



<그림 > Echo-Reply 패킷 캡쳐 화면

Type 8: Echo request
Type 0: Echo reply

Type: 8 or 0	Code: 0	Checksum
Identifier		Sequence number
Optional data Sent by the request message; repeated by the reply message		

<그림 > Echo-Request, Reply 패킷 구조

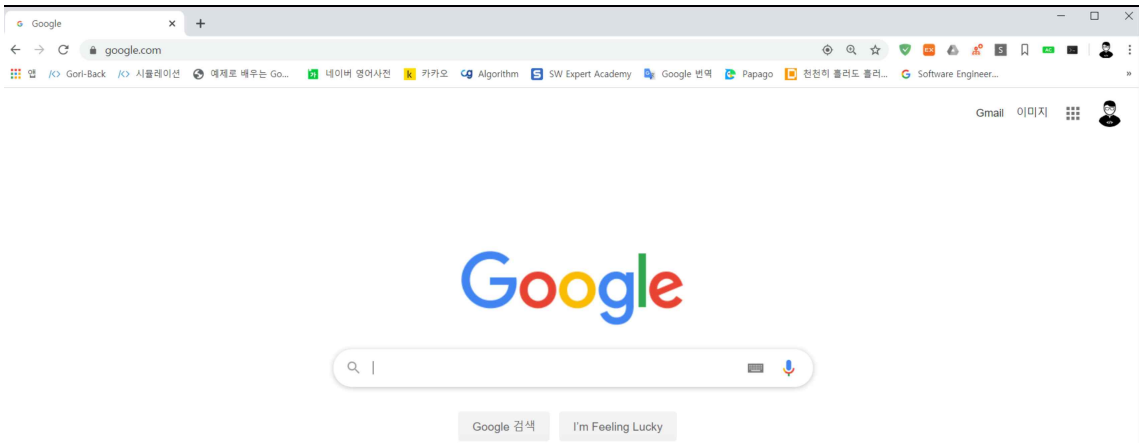
각 패킷들의 타입을 살펴보면 8과 0으로 Echo-Request와 Echo-Reply 패킷이라는 것을 알 수 있다. 또한 Checksum 필드를 통해 오류가 생기지 않은 것을 확인할 수 있고, Echo-Request, Echo-Reply 패킷의 Identifier과 Sequence number 필드를 가지고 있는 것을 확인할 수 있다.

3. HTTP

3.1 HTTP 소개

WWW(World Wide Web)은 브라우저를 사용하는 클라이언트가 서버가 제공하는 서비스에 접근하는 서버-클라이언트 서비스를 나타낸다. 서버는 웹 사이트라는 형태로 많은 곳에 분산되어 있으며 각 사이트들은 하나 이상의 웹 페이지를 참조하는 문서를 가지고 있다.

HTTP(Hyper Text Transfer Protocol)는 이러한 WWW 환경에서 사용하는 프로토콜로 클라이언트가 서버에게 웹 페이지를 요청하거나, 서버가 응답을 할 때 주로 사용한다. 주변에서 가장 흔하게 찾아볼 수 있는 예시는 웹 브라우저를 열어서 특정 웹 사이트의 주소를 입력하고 웹 페이지를 보는 것이다. [3]



<그림 > 웹 사이트 방문

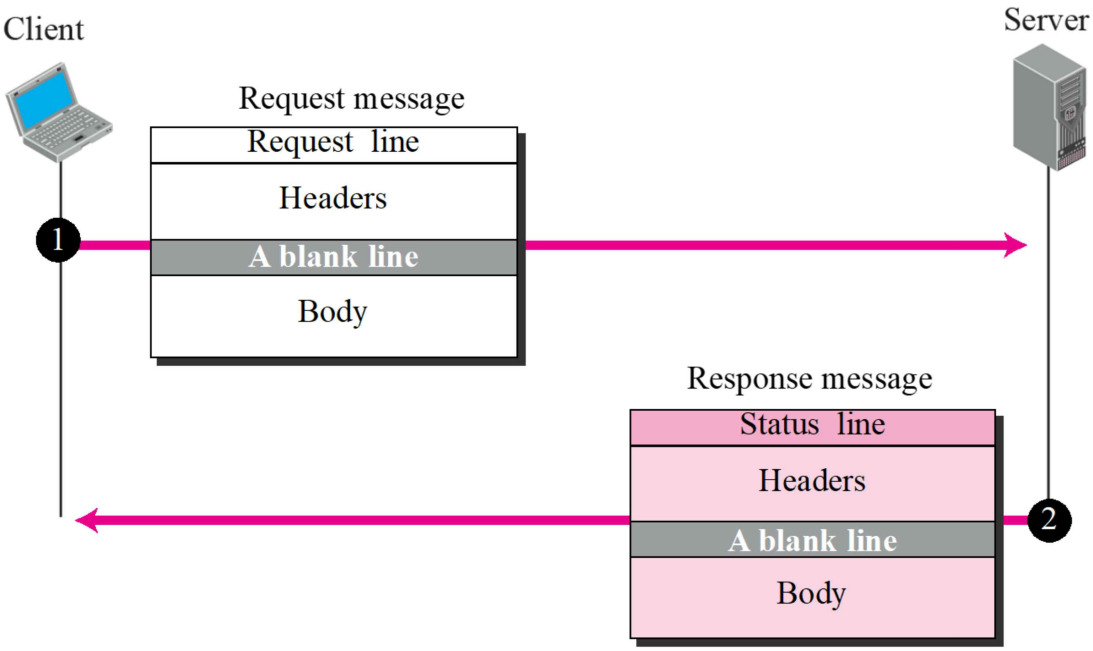
이때 URL을 사용하여 서버에게 특정 웹페이지를 요청하는데, URL(Uniform Resource Locater)은 네트워크 상에서 리소스의 위치를 알려주기 위한 규약으로, 프로토콜의 종류와 목적지의 주소 그리고 서버에서 제공하는 여러 웹페이지 중에 특정 웹페이지를 식별할 수 있는 형식을 갖추고 있다. 이러한 URL의 구조는 그림 10과 같이 이루어져 있다. [4]

<그림 > URL 구조

구글의 웹페이지를 검색할 때 나타내는 URL은 http://google.com으로 프로토콜은 HTTP를 사용하고 구글의 웹 주소는 google.com임을 알 수 있다. 여기에 포트와 경로가 생략되어 있는데, 웹 서버로 사용하는 포트는 80번으로 사전에 약속을 하였기 때문에 생략을 할 수 있다 이렇게 특정 용도에 따라 미리 정해 둔 포트를 Well-Known 포트라고 한다. 따라서 주소창에 http://google.com:80을 입력하더라도 웹페이지를 성공적으로 받아올 수 있다. [5]

또한 경로는 아무것도 입력하지 않으면 루트 페이지를 요구하게 되며, 그림 9의 경우 구글의 루트 페이지는 검색 페이지라는 것을 알 수 있다.

HTTP의 패킷 구조는 그림 11과 같으며, Request 메시지와 Response 메시지로 구분할 수 있다.



<그림 > HTTP 구조

Request 메시지는 Request Line을 통해 클라이언트의 요구사항을 명시하며 표 1과 같이 서버에게 요청할 수 있다. Response 메시지는 클라이언트의 요청에 대한 응답을 Status Line으로 표현하는데, 특정 응답 코드를 사용하여 클라이언트가 응답을 식별하도록 한다. 사용하는 응답 코드는 표2와 같다. [6]

Method	*Action*
GET	Requests a document from the server
HEAD	Requests information about a document but not the document itself
POST	Sends some information from the client to the server
PUT	Sends a document from the server to the client
TRACE	Echoes the incoming request
CONNECT	Reserved
DELETE	Remove the Web page
OPTIONs	Enquires about available options

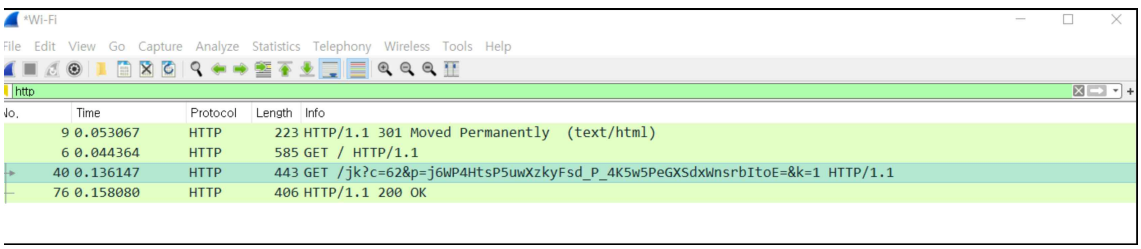
<표 > Request Line 목록

Status Code	*Status Phrase*	*Description*
	Informational	
100	Continue	The initial part of the request received, continue.
101	Switching	The server is complying to switch protocols.
	Success	
200	OK	The request is successful.
201	Created	A new URL is created.
202	Accepted	The request is accepted, but it is not immediately acted upon.
204	No content	There is no content in the body.
	Redirection	
301	Moved permanently	The requested URL is no longer used by the server.
302	Moved temporarily	The requested URL has moved temporarily.
304	Not modified	The document has not modified.
	Client Error	
400	Bad request	There is a syntax error in the request.
401	Unauthorized	The request lacks proper authorization.
403	Forbidden	Service is denied.
404	Not found	The document is not found.
405	Method not allowed	The method is not supported in this URL.
406	Not acceptable	The format requested is not acceptable.
	Server Error	
500	Internal server error	There is an error, such as a crash, at the server site.
501	Not implemented	The action requested cannot be performed.
503	Service unavailable	The service is temporarily unavailable.

<표 > Status Line의 응답 코드 목록

3.2 웹 페이지 방문

그림 12는 본 연구실 웹 사이트를 방문했을 때, Wireshark로 패킷을 캡처한 모습이다.

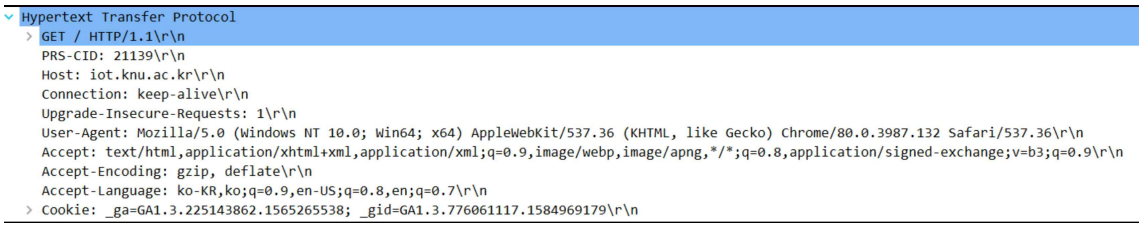


<그림 > HTTP 캡처 화면

HTTP의 Request 메시지의 Request Line에서 GET을 사용하여 웹 페이지를 요청하는 것을 볼 수 있으며, Response 메시지에서 Status Line에 200을 사용하여 성공적으로 웹 페이지

를 전송한 것을 알 수 있다.

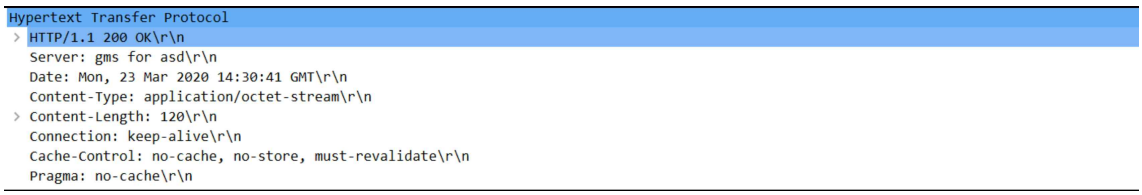
그림 13은 Request 메시지를 상세히 나타낸 것이다.



<그림 > HTTP Request 메시지

Request Line에 표시된 GET을 확인할 수 있고, Host를 통해 검색했던 주소를 확인할 수 있다 또한, Accept 필드를 통해 지원하는 문서 타입과, 인코딩 방식, 지원 언어를 확인할 수 있다.

그림 14는 Response 메시지를 상세히 나타낸 것이다.



<그림 > HTTP Response 메시지

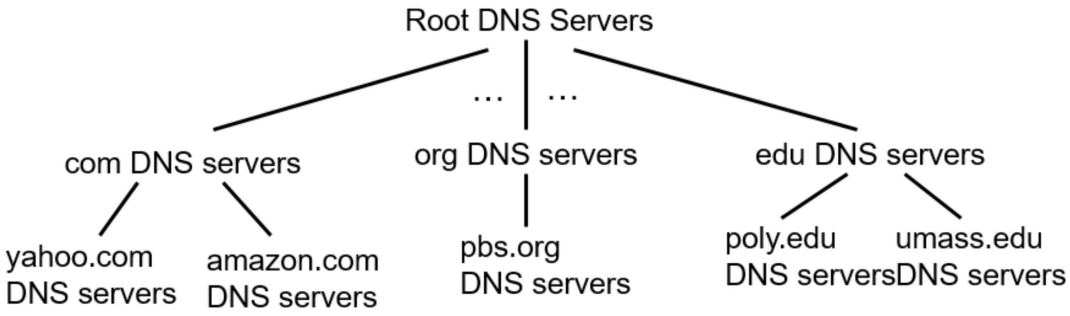
GET 요청에 대한 응답 메시지로 Status Line에 200번 코드를 통해 성공적으로 응답하였다는 것을 확인할 수 있다. 또한 Content 필드를 통해 해당 URL에서 제공하는 내용의 타입과 길이 등을 알 수 있다.

4. DNS

4.1 DNS 소개

DNS(Domain Name System)는 십진수 혹은 십육진수로 이루어진 IP 주소를 사람이 이해하기 쉬운 도메인 이름으로 변환시켜주는 시스템이다. 인터넷에 연결된 모든 개체들은 자신만의 IP 주소를 가지고 있으며, 이를 통해 서로를 식별한다. 그러나 IP 주소는 사람이 사용하기에는 알아보기 힘들고, 혼란을 줄 수 있으므로 google.com과 같이 쉽게 인지할 수 있는 도메인 이름으로 변환하여 사용자에게 편의성을 제공해준다. [7]

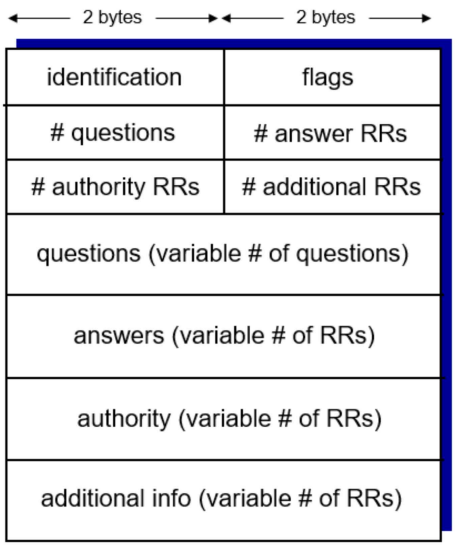
DNS 서버는 그림 15와같이 계층적 구조로 구성된다.



<그림 > DNS의 구조

최상단에 위치하는 Root DNS 서버는 클라이언트에게 가장 먼저 도메인 이름과 연결된 IP 주소를 요청하는 쿼리 메시지를 받게 된다. Root DNS 서버는 도메인 이름을 확인하고 com, org 등의 TLD(Top-Level Domain) 서버에게 해당 쿼리 메시지를 전달한다. TLD 서버는 도메인 이름을 확인하여 yahoo.com이나 amazon.com과 같이 도메인 이름을 직접 등록한 authoritative DNS 서버에게 쿼리 메시지를 전달하고, authoritative DNS 서버에서 IP 주소를 클라이언트에게 알려준다. 이 후 사용자는 자신이 등록한 로컬 네임 서버에 도메인 이름과 IP 주소 쌍을 캐싱하여, 재 접근 시 빠르게 해당 웹 서버로 접근할 수 있도록 한다.

DNS 요청 및 응답에 사용하는 패킷의 구조는 그림 16과 같다.



<그림 > DNS 패킷 형식

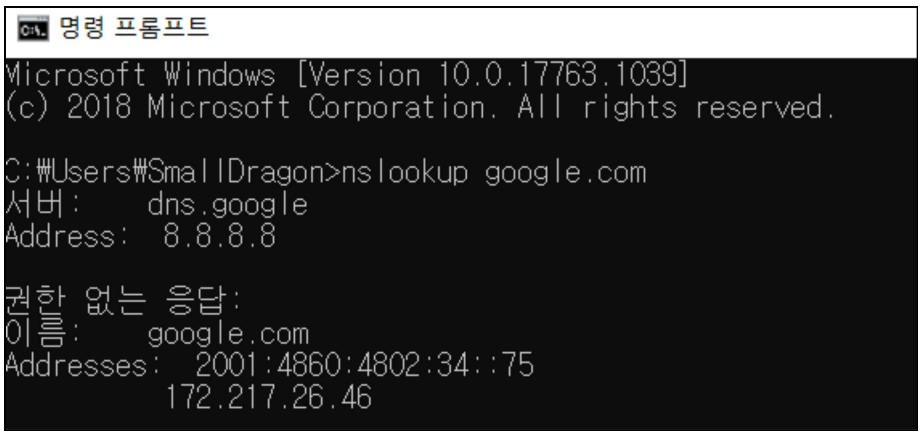
identification은 Transaction ID로 하나의 쿼리 메시지와 그에 대한 응답 메시지를 한 쌍으로 식별하기 위해 사용하며, flags는 쿼리 메시지와 응답 메시지를 구분한다. questions는 쿼리 메시지에서 도메인 이름과 타입을 나타내는데 이렇게 이름, 타입, TTL 등을 나타내는 데이터 형식을 RR(Resource Record)이라고 한다. answers는 쿼리 메시지에 대한 응답 메시지를

RR로 나타낸다. authority는 클라이언트가 authority DNS 서버의 주소를 알고 싶을 때 사용하며, additional info는 서버가 클라이언트에게 추가적으로 도움이 되는 정보를 넣기 위해 사용한다. [8]

4.2 Nslookup 사용법

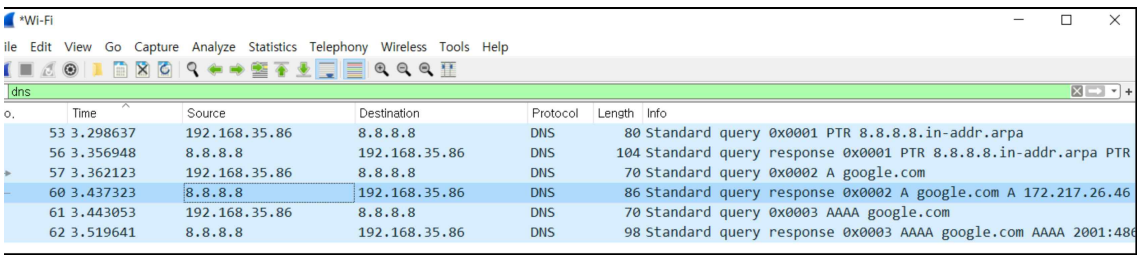
Nslookup은 DNS 패킷을 이용하여 특정 도메인 주소의 실제 IP 주소를 알아내거나, authority DNS 서버의 IP 주소를 알아 낼 수 있는 도구이다. 본 고에서는 이 도구를 이용하여 구글의 IP 주소를 알아보고 여기서 발생한 패킷을 캡처하고 분석할 것이다.

Window 명령 프롬프트에서 그림 17과 같이 nslookup google.com입력을 하면 구글의 IP 주소를 알 수 있다.



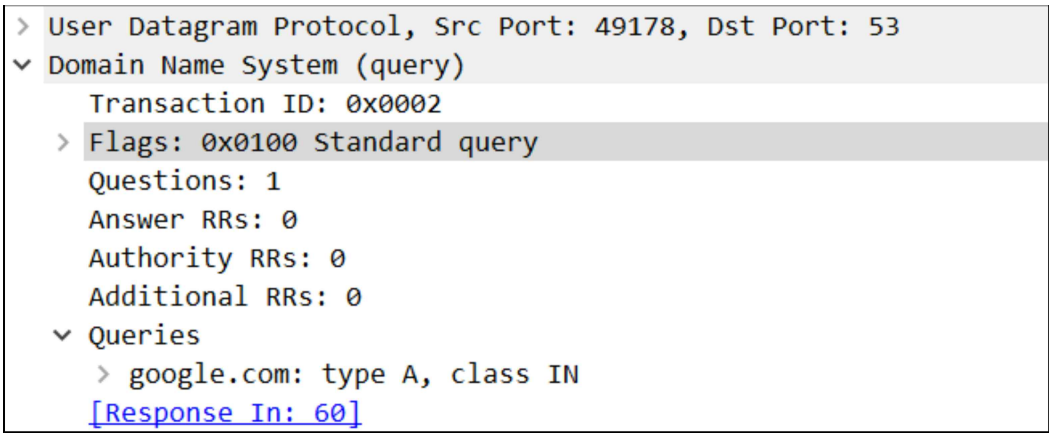
<그림 > nslookup 사용법

위 과정을 Wireshark로 캡처한다면 그림 18과 같은 결과를 볼 수 있다.



<그림 > DNS 패킷 캡쳐

그림 19는 캡처한 DNS 쿼리 메시지의 자세한 정보를 나타낸 것이다.



<그림 > DNS 쿼리 메시지

Transaction ID를 통해 2번째 쿼리 메시지라는 것을 알 수 있으며, flags 필드를 확인하면 해당 메시지가 쿼리 메시지임을 알 수 있다. 또한 Question 필드를 통해 쿼리에 사용되는 RR이 1개임을 알 수 있으며, 그에 해당하는 내용은 Queries 필드에서 확인할 수 있다.

5. 결론

지금까지 본 고에서는 ICMP 프로토콜에 관해 설명하고, PING 기능을 통해 ICMP 패킷을 직접 캡처하여 분석하였으며, HTTP 프로토콜에 관한 정의와 Request 및 Response 메시지에 대해 자세히 알아보고, 직접 웹 사이트를 방문하면서 발생한 HTTP 패킷들을 캡처하여 분석하였다. 또한, DNS에 정의와 동작 구조에 대해 살펴보고, nslookup 도구를 활용하여 DNS 쿼리 메시지와 응답 메시지를 확인하였다. 이를 통해 인터넷에서 사용되는 기본적인 프로토콜에 대한 깊은 이해에 도움이 되었으면 하며, 추후 QUIC과 같이 새롭게 등장하고 있는 프로토콜을 분석하거나, 자신만의 프로토콜을 개발하고 분석할 때 도움이 되길 바란다.

참고 문헌

[1] Packet Analyzer: Wireshark 설치 가이드, https://iot.knu.ac.kr/tech/CPL-TR-08-07-wireshark.pdf

[2] Postel J, “Internet Control Message Protocol”, IETF RFC 777, April 1981

[3] R. Fielding, et al., “Hypertext transfer Protocol”, IETF RFC 2616, June 1999

[4] T. Berners-Lee, et al., “Uniform Resource Locators (URL)”, IETF RFC 1738,

December 1994

[5] J. Reynolds, et al., “ASSIGNED NUMBERS”, IETF RFC 1700, October 1994

[6] Behrouz A. Forouzan, “World Wide Web and HTTP”, TCP-IP Protocol Suite, July 1999

[7] P. Mockapetris, “DOMAIN NAMES – CONCEPTS AND FACILITES”, IETF RFC 1034, November 1987

[8] Computer Networking: A Top-Down Approach 6th ed. Chapter 2: The Application Layer, http://www-net.cs.umass.edu/kurose-ross-ppt-6e/Chapter_2_V6.3.ppt