사물 인터넷 통신 프로토콜

경북대학교 컴퓨터학부

고석주

<목 차>

1. 사물인터넷(IoT) 기술 개요

(1) 사물인터넷(Internet-of-Things) 개념

- 사물인터넷(IoT, Internet-of-Things)이란 "사람, 사물, 데이터 등 모든 것을 **인터넷으로 연결하여 정보의 생성·수집·처리·활용을 통해 부가가치를 생성**"하는 기술을 의미한다[1].

- 예전에는 주로 각종 센서를 연결해주는 무선 근거리 통신에서 IoT 기술이 사용되었지만, 최근에는 클라우드 플랫폼을 포함하여 유무선 원거리 통신에도 광범위하게 적용되고 있다.

<모든 것을 연결해주는 사물인터넷(IoT) 개념도>

1) IoT와 유사한 개념으로서 IoE (Internet of Everything), IoST (Internet of Small Things), WoT (Web of Things) 등 다양한 용어가 사용된다.

(2) IoT 서비스/기술 구성 요소

- IoT 서비스는 **디바이스(Device)/센서(Sensor), 네트워크(Network), 플랫폼(Platform), 클라이언트(Client)/서비스(Service)**로 구성된다[2].

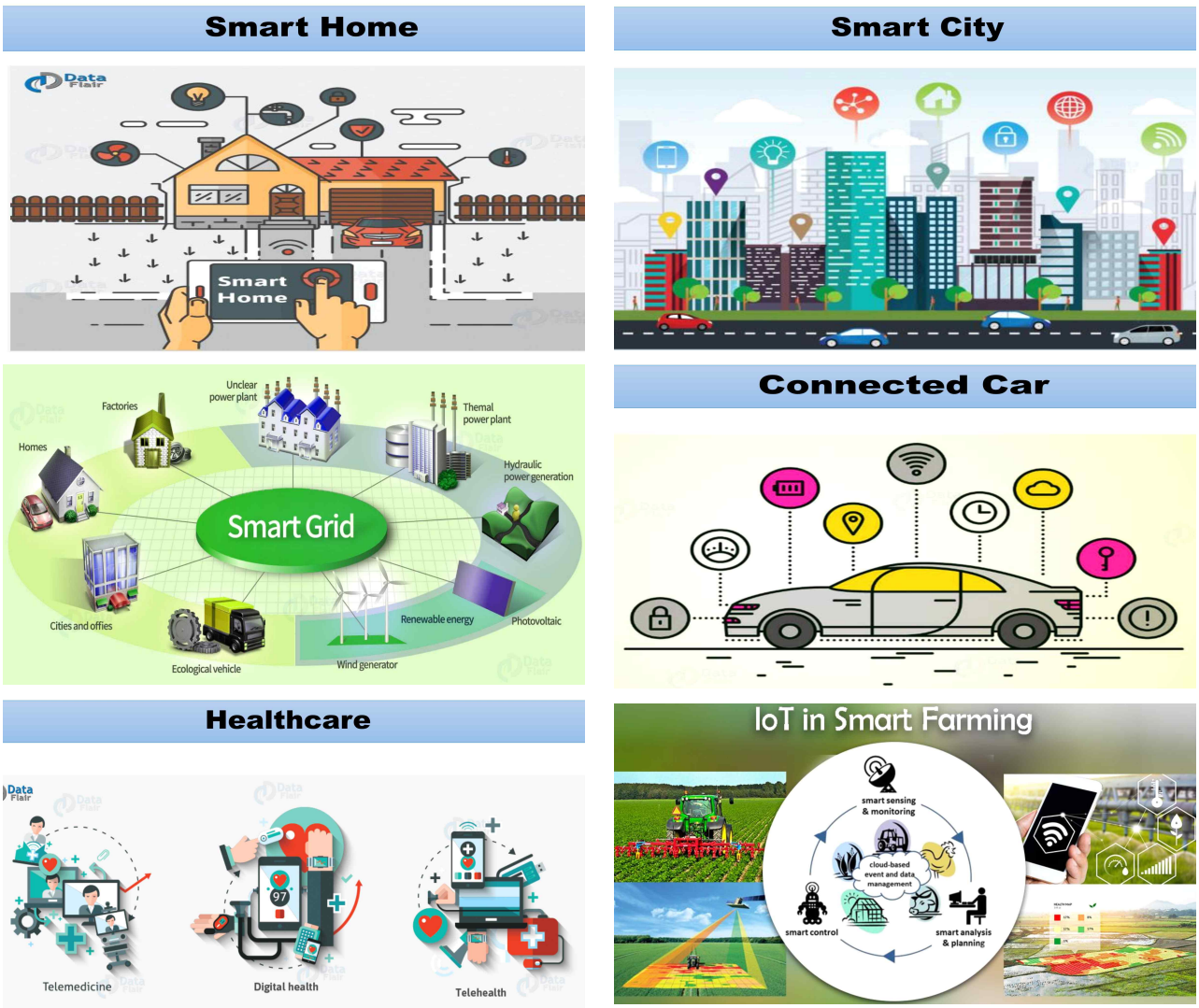
- 사물인터넷 서비스는 각종 **센서(디바이스)**에서 시작된다. 센서를 통해 측정된 다양한 데이터는 Wi-Fi 및 이동통신 등의 **네트워크** 통신 기술을 통해 **플랫폼(미들웨어)** 서버에 전달한다. 플랫폼 서버는 센서를 통해 수집한 데이터를 취합, 분류, 분석하여 의미있는 정보를 생성하고[3] 이를 **고객(Client)** 혹은 응용 **서비스(Service)**에 제공한다.

<IoT 서비스/기술 구성 요소 (D-N-P-C)>

2) IoT 구성요소 4가지를 줄여서 D-N-P-C(S)로 표현한다. 이는 IoT 뿐만 아니라, 인공지능(AI) 및 빅데이터 관련 서비스에도 유사하게 적용된다.
3) 플랫폼에서는 다양한 데이터를 취합, 저장, 관리하기 위해서 '빅데이터' 기술을 사용하고, 빅 데이터 기반의 기계학습, 추론 및 유용한 정보 생성 등을 위해 '인공지능' 기술을 사용한다.

(3) IoT 응용 서비스

- IoT를 통해 제공할 수 있는 서비스는 매우 다양하다. 스마트 홈, 스마트 시티, 스마트 그리드(전력), 자율주행차, 헬스케어, 스마트 팜 등 모든 산업 분야에 IoT 기술이 적용될 수 있다[4].



<다양한 IoT 응용/서비스>

4) 통계자료에 의하면 전체 IoT 서비스 시장에서 '스마트 홈' 서비스가 30% 정도를 차지하고 있다.

▪ 조선·해양 관련 **'자율운항 선박'에도 IoT 기술이 적용**되고 있다. IoT 기술을 활용하여 선박 및 주변 환경에 대한 데이터를 수집하고(상황 인식·탐지), 플랫폼(관제 센터)에 전달하면, 플랫폼에서 데이터 처리·가공 및 인공지능 기술을 적용하여 선박의 자율운항에 활용한다.

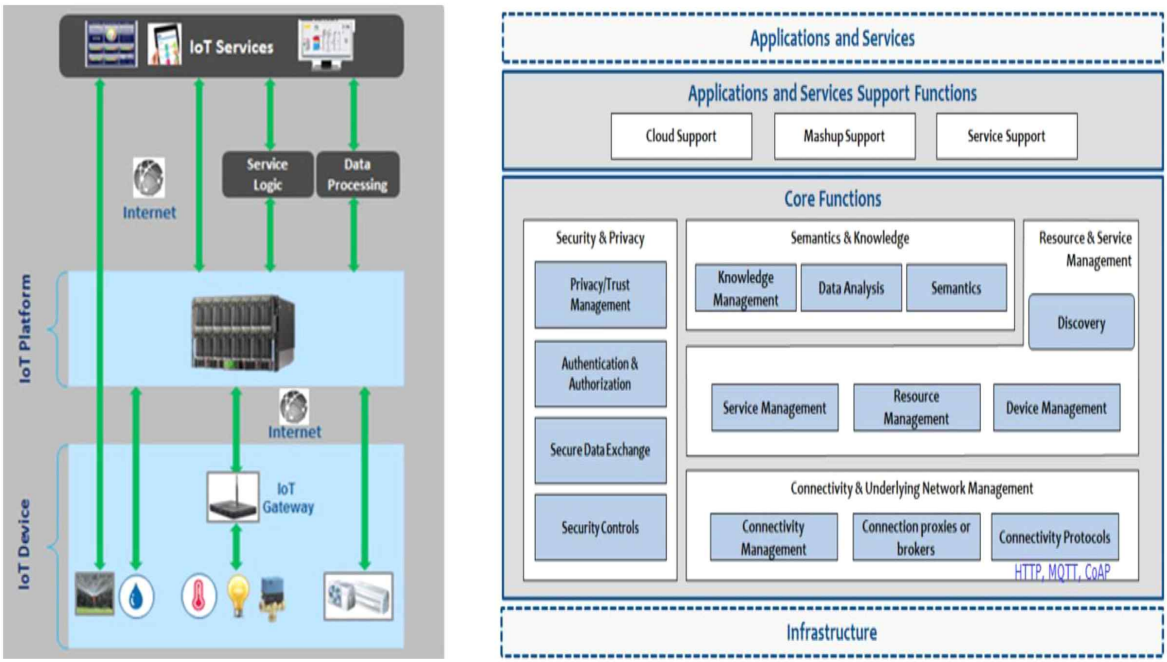
<IoT 기반 자율운항 선박 시스템>

(4) IoT 플랫폼

A. 플랫폼(서버) 기능 구성

- IoT 플랫폼은 다양한 기능을 제공한다. 디바이스가 전송하는 데이터를 취합하고 이를 토대로 빅데이터 및 인공지능 기법을 적용하여 사용자 혹은 응용 서비스를 대상으로 다양한 정보를 제공한다.

- 아래 그림에서 알 수 있듯이, 플랫폼에서는 다양한 디바이스 관리, 네트워크 통신 관리, 데이터 분석 및 인공지능 학습·추론 기능 등을 수행한다.



<IoT 플랫폼 기능 구성도>

- 최근에는 국내외의 다양한 기업에서 '개방형' 플랫폼을(SDK, Open API 등) 개발하여 제공하고 있으며, 응용 서비스 개발자들은 IoT 플랫폼을 기반으로 다양한 IoT 응용 서비스를 쉽게 개발할 수 있다.

B. 국내외 IoT 플랫폼 개발 동향

- **(국내 동향)** 삼성, LG, KT, SKT 등의 여러 대기업에서 자체 IoT 플랫폼을 개발하고 지원하고 있으며, KETI(한국전자기술연구원)에서는 개방형 IoT 플랫폼을 개발하여 일반인 대상으로 무료로 제공하고 있다.

- **(국외 동향)** Google, Amazon, Microsoft 등의 기업들이 IoT 플랫폼을 개발하여 자사에서 개발한 다양한 IoT 서비스와 연계하고 있다.

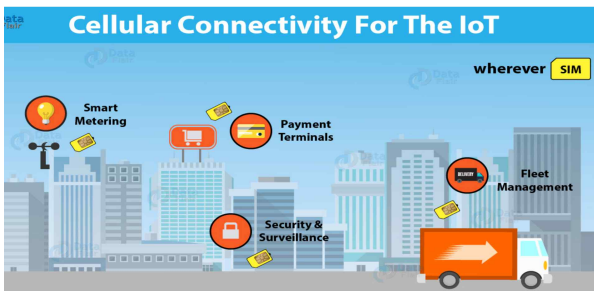
<국내외 IoT 플랫폼 개발 동향>

구분	플랫폼명	기관명	특 징
국내	SmartThings	삼성전자	- 빅스비를 연계하여 각종 IoT 디바이스 제어 - 음성 인식을 통해 디바이스 제어
	IoTMakers	KT	- KT Cloud를 활용한 서비스를 지원할 수 있음 - B2B 사업을 통해 기업용 IoT 관리 솔루션 제공
	ThingPlug	SKT	- SKT 인프라/데이터를 기반으로 IoT 서비스 지원 - LoRA 디바이스와 연동 기능 지원
	Mobius	KETI	- 일반 IoT 응용 개발자를 위해 소스 코드 제공 - http://www.iotocean.org/
국외	Cloud IoT	Google	- Google 지도를 활용한 위치 기반 IoT 솔루션 제공 - TensorFlow를 활용한 머신러닝 서비스 제공
	AWS IoT	Amazon	- Amazon CloudWatch, Amazon DynamoDB 등의 클라우드/DB와 연계 서비스 제공
	Azure IoT	Microsoft	- IoT 업계의 유일한 엔드투엔드 보안 솔루션을 사용하여 더 안전한 애플리케이션 빌드 가능
	IoTivity	OCF	- 인텔, 삼성전자 등 글로벌 기업들이 공동으로 설립 - 사물인터넷 국제 플랫폼 표준 개발

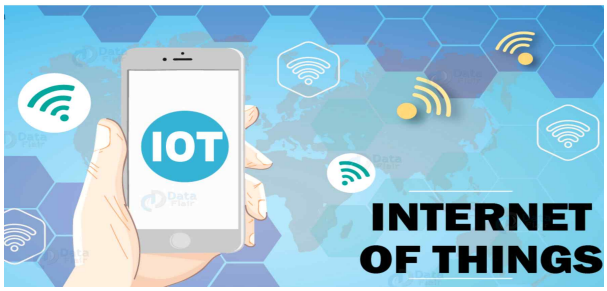
(5) 네트워크 전송 기술

- 각종 IoT 디바이스에서 수집된 정보는 네트워크 전송 기술을 활용하여 IoT 플랫폼으로 전달된다. 이를 위해 사용되는 네트워킹 기술로는 이동통신(3G/LTE/5G), Wi-Fi, Bluetooth, ZeeBee, LoraWAN 등 다양하다.

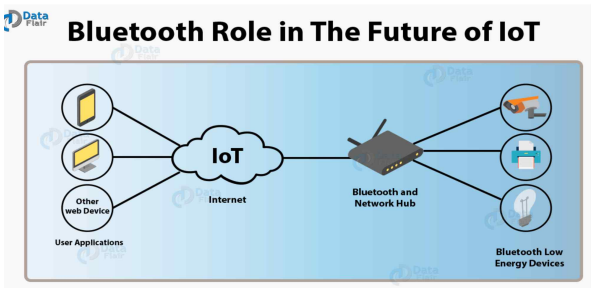
Cellular/LTE/5G (3GPP)



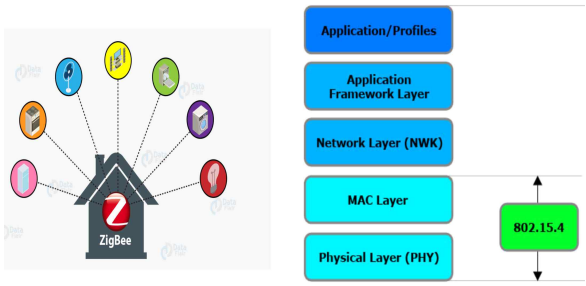
Wi-Fi (IEEE 802.11)



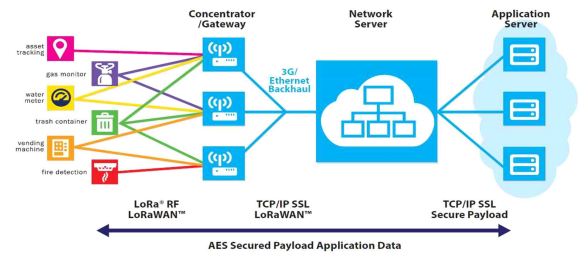
Bluetooth (IEEE 802.15.1)



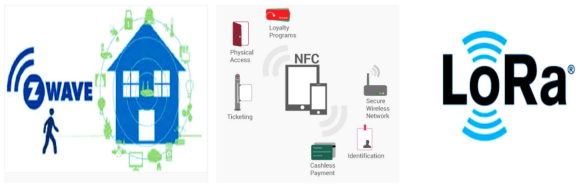
ZigBee (IEEE 802.15.4)



LPWA (LoRaWAN)

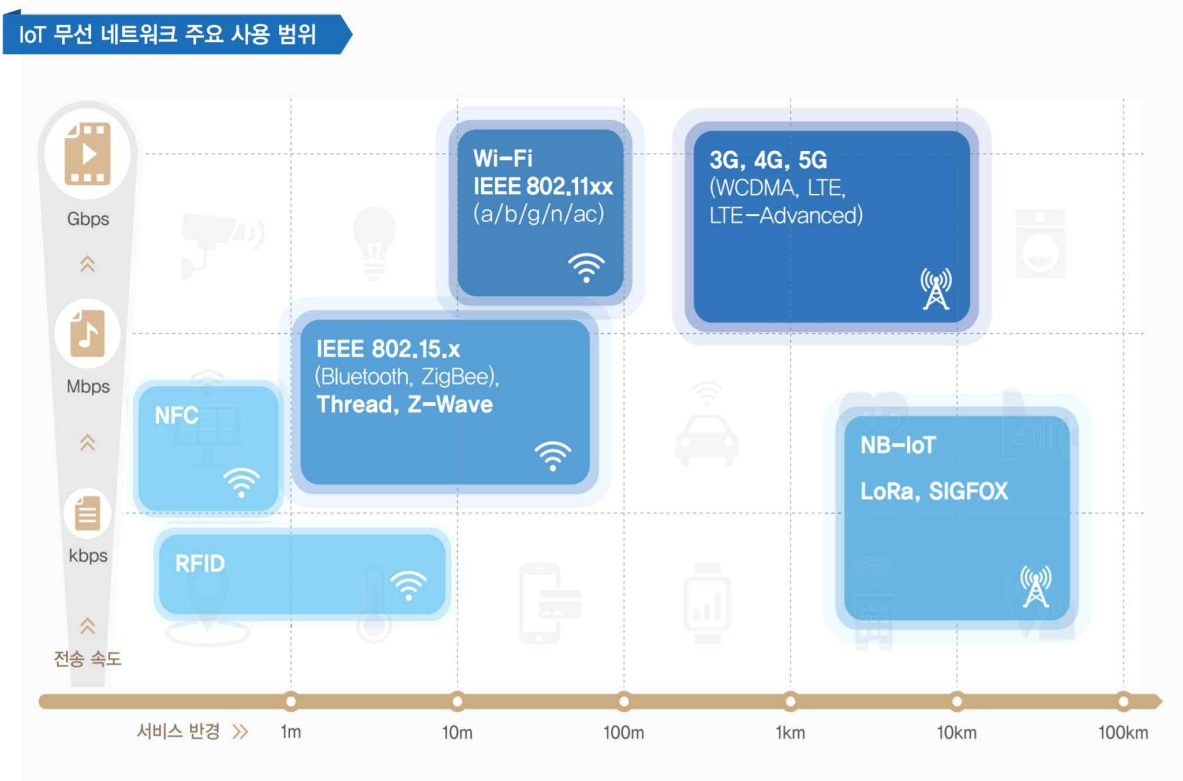


Z-Wave/NFC/LoRaWAN



<다양한 IoT 네트워킹 기술 (디바이스 ↔ 플랫폼)>

- 각 네트워킹 기술들은 데이터 전송속도 및 전송거리(coverage) 측면에서 특징이 있다. 이동통신 기술은 원거리 통신에 적합하고, Wi-Fi는 중거리, 블루투스 및 ZeeBee는 근거리 고속 통신에 적합하다[5].



<IoT 네트워킹 기술의 특징 비교(전송속도 vs. 커버리지)>

- 한편, 이러한 네트워크를 토대로 디바이스와 플랫폼간에 데이터를 전송하기 위해서는 IoT 메시지 전송 프로토콜이 사용된다. 예전에는 웹(Web) 전송을 위한 HTTP(HyperText Transfer Protocol) 프로토콜이 사용되었으나, 최근에는 저전력 소형 센서용으로 MQTT(Message Queue Telemetry Transport) 및 CoAP(Constrained Application Protocol) 프로토콜이 개발되어 사용되고 있다[6].

---

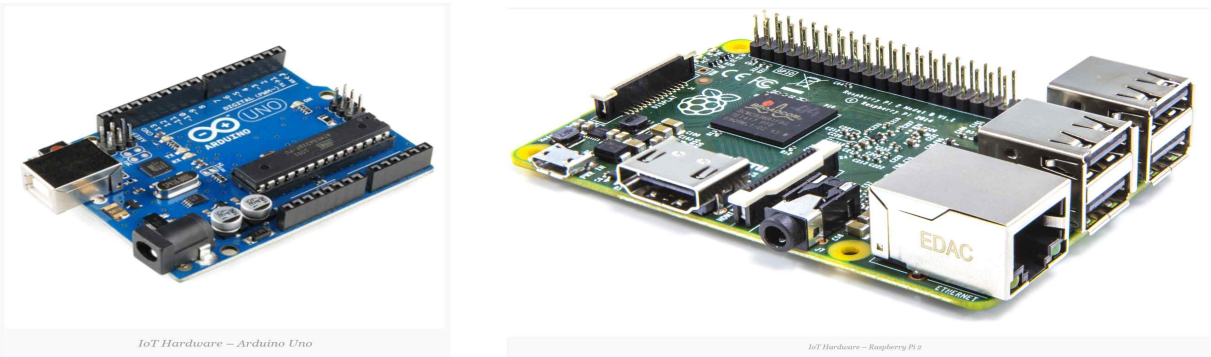
5) LoRaWAN 및 NB-IoT 통신 기술은 저렴한 비용으로 원거리 저속 통신 기능이 가능하다.
6) IoT 메시지 전송 프로토콜 관련 내용은 이 장의 후반부에서 설명한다.

(6) IoT 디바이스/센서

- IoT 서비스에 따라 다양한 종류의 센서 혹은 디바이스가 사용된다. 예를 들어 스마트홈을 위한 CCTV, 스마트팜용 온도 센서, 헬스케어를 위한 체온계 등, 서비스 특성 및 목적에 따라 디바이스 종류가 다양하다.

<다양한 IoT 디바이스/센서>

- 각 IoT 디바이스는 주변 정보를 측정·수집하여 플랫폼 서버에 전달하는데, 이를 위해 TCP/IP 및 MQTT/CoAP 등의 프로토콜 기능이 디바이스에 탑재되어야 한다. 이러한 IoT 디바이스 개발을 위해 **아두이노 혹은 라즈베리 파이** 등의 Open Source 하드웨어 장비가 널리 활용되고 있다.



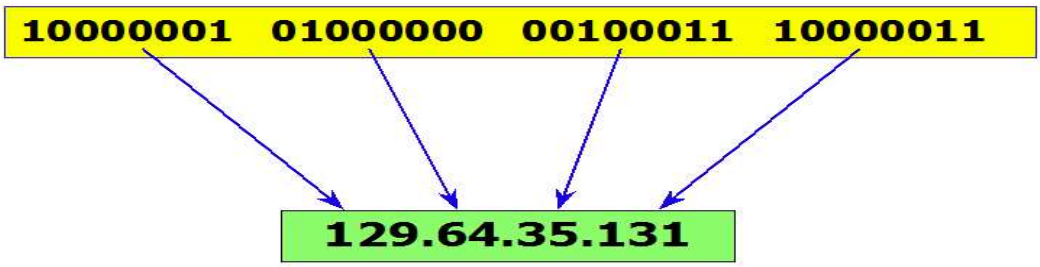
<IoT 디바이스 개발용 하드웨어(아두이노, 라즈베리파이)>

2. TCP/IP 통신 프로토콜

(1) IP주소(address)

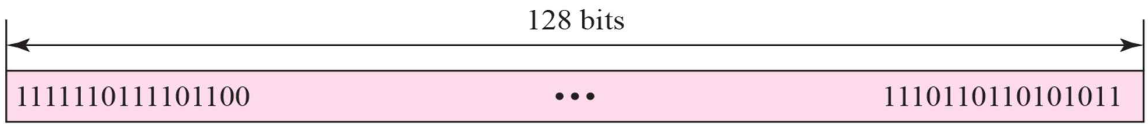
- **IP(Internet Protocol) 주소**란 "인터넷 상에서 내 컴퓨터를 식별하기 위한 주소"이다. 즉, IP 주소를 통해 상대방의 컴퓨터에 인터넷 데이터를 보낼 수 있으며, 또한 상대방이 내 컴퓨터에 데이터를 전달할 수 있다.

- IP는 버전(version)에 따라 IPv4와 IPv6로 구분되는데, IPv4 주소는 "129.64.35.131"처럼 4개의 십진수와 "."를 사용하여 표현한다.



<IPv4 주소의 표현(dotted-decimal notation)>

- IoT 서비스를 위해서는 수 많은 센서에 IP 주소가 필요하게 되어 IPv4 (32비트) 주소가 부족하게 되었다. 이러한 주소 부족 문제를 해결하기 위해 IPv6 (128비트) 주소가 개발되었다.



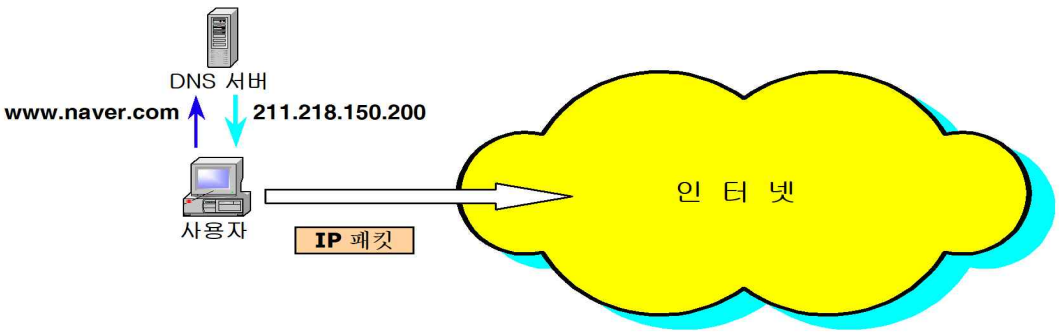
FDEC : BA98 : 7654 : 3210 : ADBF : BBFF : 2922 : FFFF

<IPv6 주소의 표현(colon-hexadecimal notation)>

(2) 도메인 이름(domain name) 및 웹주소(URL)

- IP주소는 사람이 기억하기 어렵기 때문에 기억하기 쉬운 **도메인 네임(domain name)**이 필요하다. 도메인 네임은 인터넷에서 컴퓨터를 식별하기 위해 사용하는 컴퓨터의 고유 이름이다. 예를 들어, 네이버 홈페이지 서버는 "www.naver.com"라는 도메인 네임을 사용한다.

- 인터넷 통신을 위해서는 DNS(Domain Name System) 서버를 사용하여 도메인 이름에 해당하는 IP 주소를 얻어오는 과정이 필요하다.



<DNS 서버를 활용한 IP 주소 획득>

- **웹주소**란 **URL(Uniform Resource Locator)**이라고도 하며, 인터넷에서 특정 '웹문서의 위치'를 나타내는 주소이다. 웹주소는 다음과 같이 "웹서버 이름"과 "경로명"으로 구성된다[7].

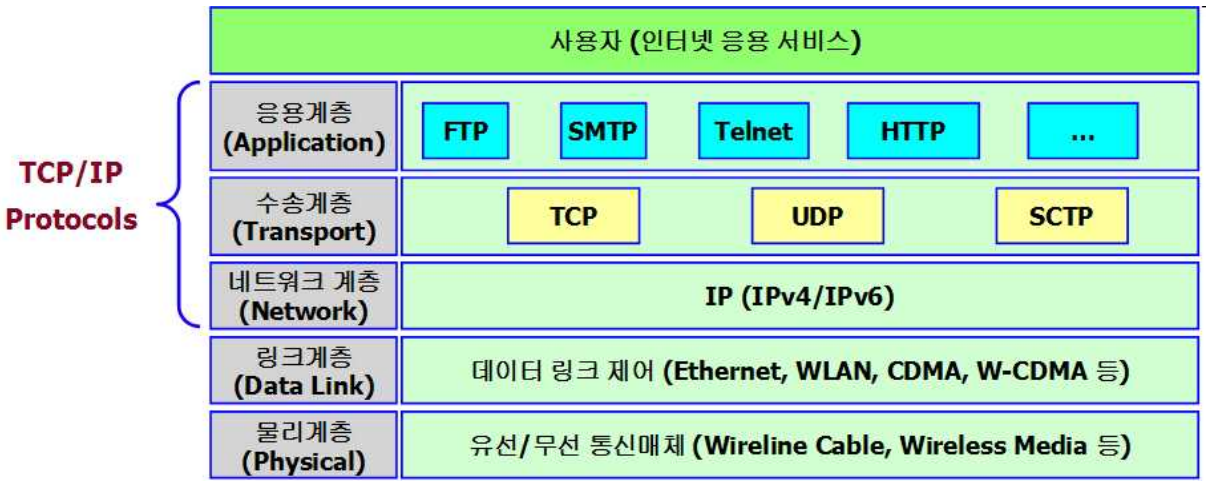
http://	웹서버 이름	/	경로명(파일 이름)

<웹주소(URL) 예시>

7) 예를 들어, "www.naver.com/index.html"은 네이버 웹서버(www.naver.com)에서 "index.html" 이란 문서 경로를 나타내는 URL이다

(3) TCP/IP 프로토콜 스택(Protocol Stack)

- 인터넷 통신은 다음 그림처럼 'TCP/IP 프로토콜 스택'에 해당하는 여러 가지 프로토콜들을 혼합 사용하여 이루어진다[8].



<TCP/IP 프로토콜 스택>

- TCP/IP 프로토콜 스택은 5개의 계층(layers)으로 구분되며, 계층별로 역할이 다르다. 이 중 **상위 3계층(네트워크, 수송, 응용)을 'TCP/IP 프로토콜 스택'**이라 하며 인터넷에서 컴퓨터간 데이터 전송 기능을 수행한다.

<메 모>

- 통신 프로토콜이란 "통신규칙"을 의미한다. 즉, 두 대의 컴퓨터가 서로 통신을 하기 위해서는 모든 인터넷 장비들이 프로토콜에 따라 동작을 해야 한다.
- 표준화 기구에서 통신 프로토콜 표준을 제정하며, TCP/IP 프로토콜의 표준화는 IETF 표준화 기구에서 진행되고 있다. (http://www.ietf.org/)

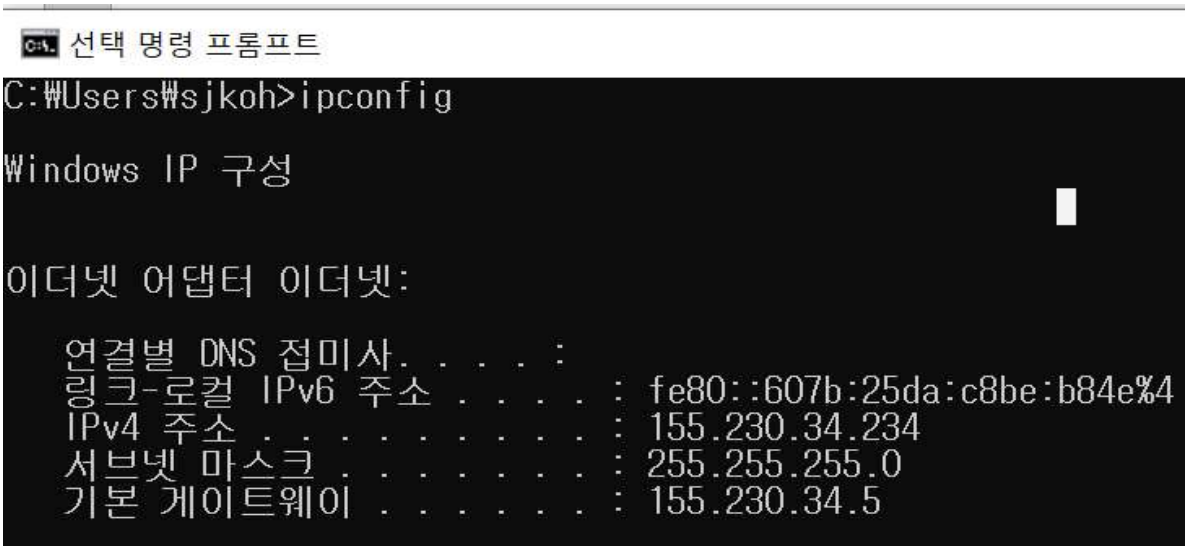
8) 주요 프로토콜 약어: IP(Internet Protocol), TCP(Transmission Control Protocol), UDP(User Datagram Protocol), HTTP(HyperText Transfer Protocol), FTP(File Transfer Protocol)

(4) Internet Protocol (IP)

A. IP 주소 확인

- IP 프로토콜은 인터넷 네트워크(망)을 통해 IP 패킷을 목적지(수신) 컴퓨터까지 전달하기 위해 사용된다. 패킷전달 과정에서 인터넷에 있는 여러개의 라우터(router)를 경유하게 되는데, 각 라우터에서는 패킷에 포함되어 있는 '수신 IP주소'를 참고하여 목적지 컴퓨터까지 IP 패킷을 전달하는 기능을 수행한다.

- 먼저, 내 컴퓨터의 IP주소가 무엇인지 알아보자[9]. 아래 그림에서 보여지듯이, 내 컴퓨터의 IP주소 및 서브넷 마스크, 게이트웨이[10], IPv6 주소 등의 정보를 파악할 수 있다.



<컴퓨터에서 IP주소 확인하기>

9) Windows를 사용하는 경우 왼쪽 하단의 검색창에 'cmd'라고 입력하면 위 그림처럼 '명령 프롬프트' 창이 열린다. 이 때 'ipconfig' 명령어를 입력하면 내 컴퓨터의 IP주소를 확인할 수 있다.
10) 여기에서 게이트웨이(gateway)는 내 컴퓨터가 연결되어 있는 라우터를 의미함.

B. IP 패킷 전달 과정

- TCP/IP 프로토콜에 의한 인터넷 패킷 전달과정을 정리하면 다음과 같다.

IP 패킷 전달 과정

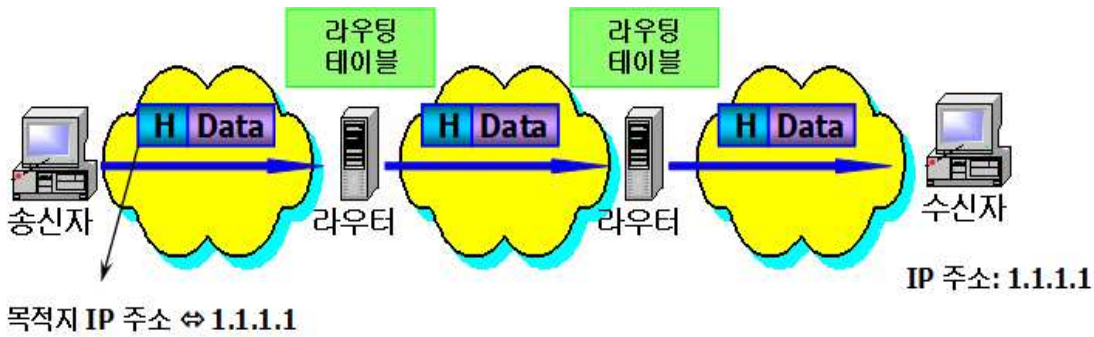
① 송신컴퓨터는 IP 패킷을 구성하고, 수신 컴퓨터의 IP주소를 파악한다.

② IP 패킷 헤더에 수신자 IP 주소를 첨가한 후, 패킷을 인터넷으로 전송한다.

③ 인터넷에 있는 각 라우터들은 라우팅 절차에 따라 수신 컴퓨터에게 패킷을 전달한다.

④ 수신 컴퓨터는 도착한 패킷에서 데이터를 추출하여 필요한 작업을 수행한다.

- 다음 그림은 TCP/IP 프로토콜을 이용한 IP 패킷 전달 과정이다.



<인터넷 망에서 패킷 전달 과정>

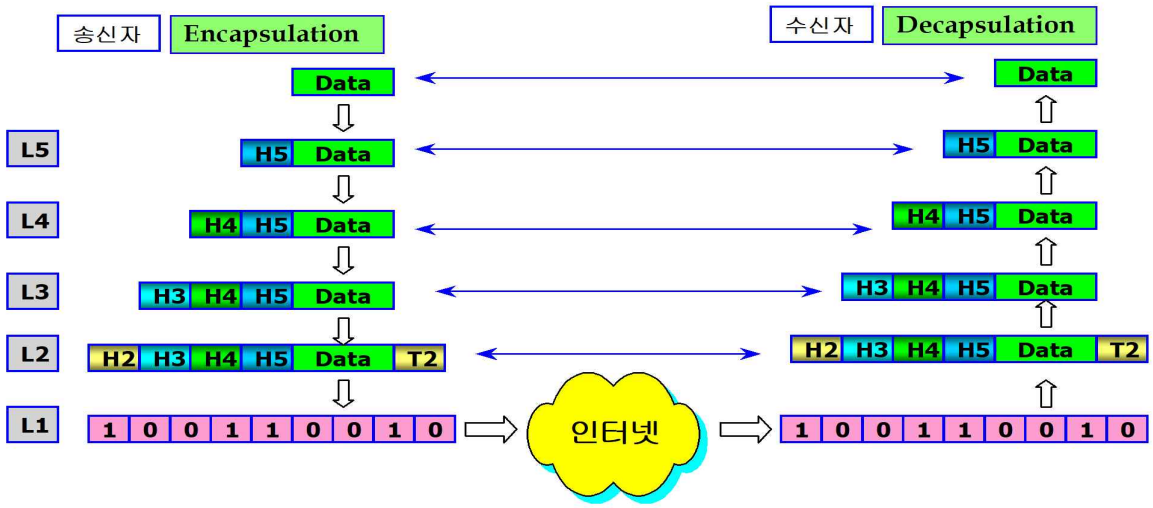
- IP 패킷은 헤더(header)와 데이터(data) 부분으로 구성되는데, 헤더 부분에 송신자와 수신자의 IP 주소가 기록된다. IP 패킷전달은 인터넷 상의 라우터(router)에 의해서 이루어진다. 각 라우터는 '라우팅 테이블(routing table)'을 참조하여, 특정 패킷을 어디로 전달(forwarding)해야 할지를 결정한다. 이처럼 여러 라우터의 패킷 전달과정을 통해 최종적으로 IP 패킷이 수신컴퓨터에 도착하게 된다.

- 다음은 **ping** 프로그램을 사용한 IP 패킷전달 확인과정을 보여준다.

```
명령 프롬프트

C:₩Users₩sjkoh>ping www.google.com

Ping www.google.com [142.250.207.68] 32바이트 데이터 사용:
142.250.207.68의 응답: 바이트=32 시간=41ms TTL=110
142.250.207.68의 응답: 바이트=32 시간=41ms TTL=110
142.250.207.68의 응답: 바이트=32 시간=41ms TTL=110
142.250.207.68의 응답: 바이트=32 시간=41ms TTL=110

142.250.207.68에 대한 Ping 통계:
    패킷: 보냄 = 4, 받음 = 4, 손실 = 0 (0% 손실),
왕복 시간(밀리초):
    최소 = 41ms, 최대 = 41ms, 평균 = 41ms
```

<ping 활용 예제>

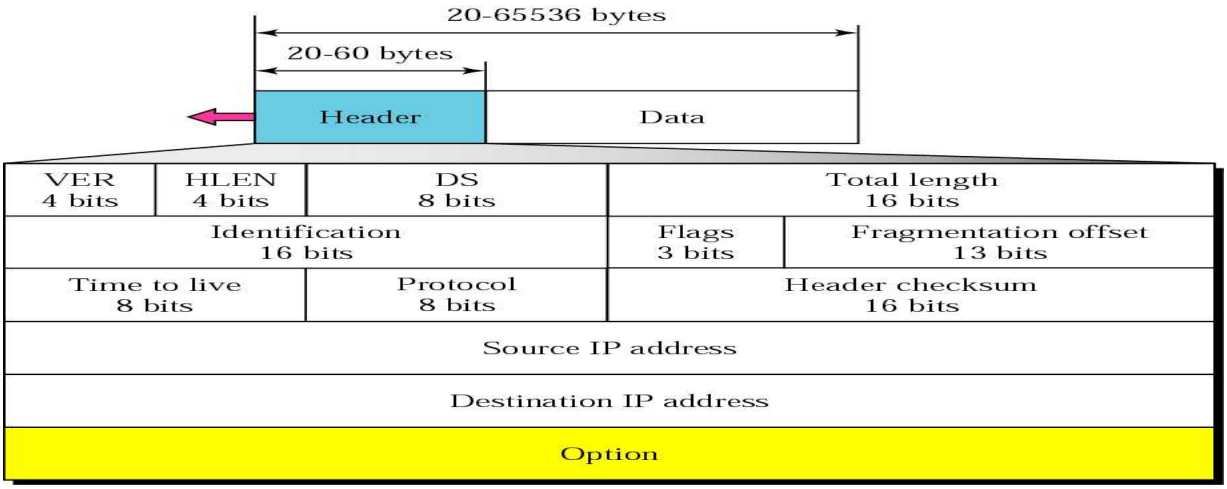
- 패킷 전달과정에서 중요한 점은 **캡슐화(encapsulation)**이다. 사용자의 데이터는 캡슐화 과정을 통해 인터넷 패킷으로 구성되고, 송신 컴퓨터를 떠나 인터넷을 향해한다. 인터넷을 도착한 패킷이 수신 컴퓨터에 도달했을 때, 데이터는 **역캡슐화(decapsulation)**를 통해 수신자에게 전달된다.



<인터넷 패킷 캡슐화(encapsulation) 과정>

C. IP 패킷 헤더(header) 포맷

- IP 주소는 IP 패킷 헤더 부분에 기록되어 라우터의 패킷 전달과정에 참조된다. 그 밖에 IP 헤더는 여러 가지 다양한 정보를 포함한다. IPv4 헤더는 **총 20바이트로 구성**되며, 필요에 따라 옵션(option)이 추가될 수 있다.



<IP 패킷 헤더>

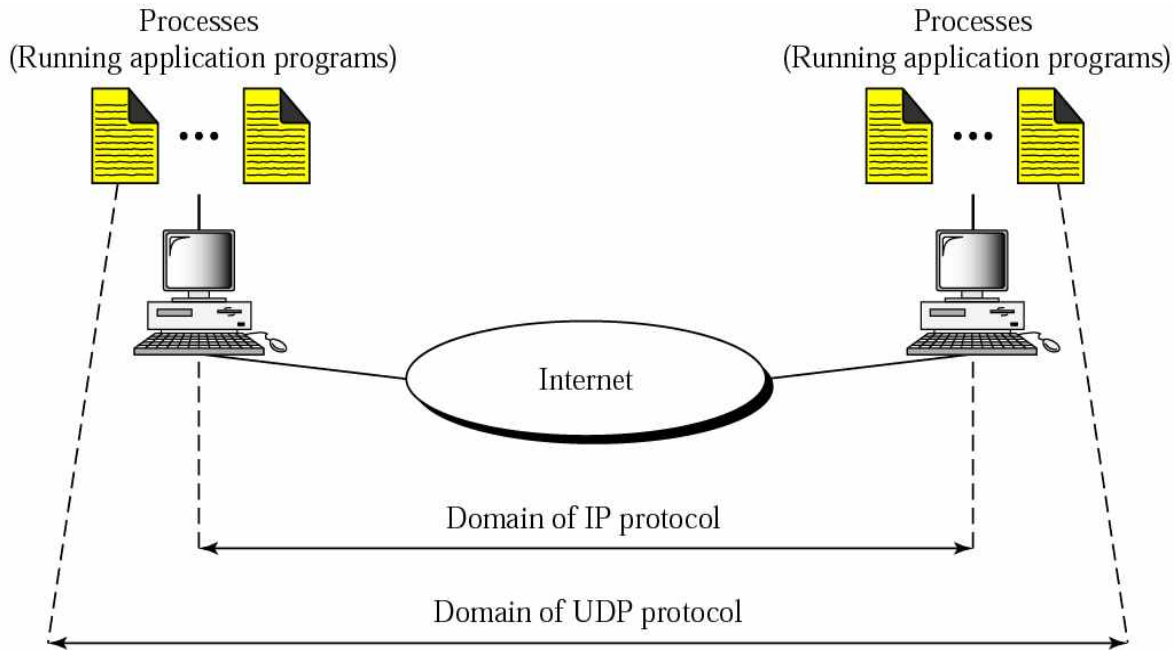
IP 패킷 헤더에 포함되는 필드 내용

- VER: IPv4 혹은 IPv6 등의 IP 버전을 나타냄
- HLEN: 옵션을 포함한 IPv4 헤더 길이(20~60바이트; 4바이트 단위)
- DS: Differentiated Service 프로토콜이 사용되는 경우 해당 정보를 표시함
- Total Length: 헤더와 데이터를 포함한 IP 패킷의 전체 길이를 표시함
- 2번째 라인: 패킷이 전송 도중에 fragmentation 되었을 경우에 관련 정보 표시
- Time To Live (TTL): IP 패킷이 몇 개의 라우터를 경유하였는지를 표시함
- Protocol: 패킷의 데이터가 포함하는 프로토콜 정보 (예: TCP, UDP 등)
- Header Checksum: 헤더가 전송 도중에 오류를 경험했는지 검사하기 위해 사용됨
- Source 및 Destination IP 주소: 송수신 컴퓨터의 IP 주소를 기록함

(5) User Datagram Protocol (UDP)

A. 프로세스간 통신

- IP 프로토콜은 네트워크에서 패킷을 전달하기 위해 사용되는 반면에, UDP 및 TCP 프로토콜은 송수신 컴퓨터의 프로세스(프로그램)간 통신을 위해 사용된다. 이를 '**프로세스간(process-to-process) 통신**'이라 부른다.



<프로세스간 통신 (process-to-process communication)>

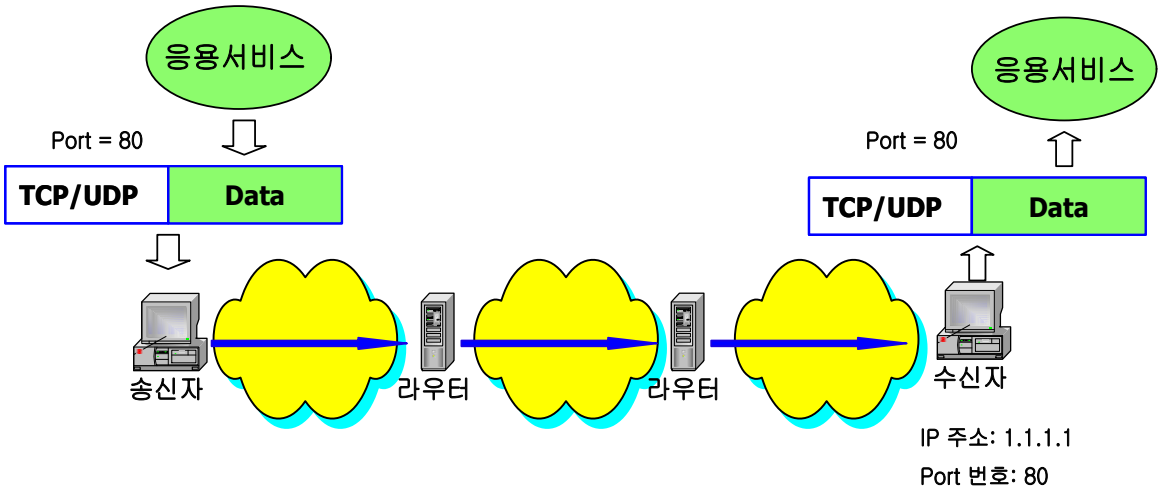
- UDP의 주요 기능(사실상 유일한 기능)은 송수신 컴퓨터에서 동작하는 응용 프로그램(혹은 프로세스)간 통신을 위해 포트(port) 번호를 제공하는 것이다. UDP 패킷 헤더(8바이트)에는 2바이트 크기의 송신 포트번호와 수신 포트번호를 포함한다[11].

11) 다음 절에서 설명할 TCP 패킷 헤더에도 2바이트 크기의 송수신 포트번호가 포함된다.

B. 포트 번호 (port number)

- 한편, UDP (TCP 포함) 등의 수송계층 패킷의 헤더에는 2 바이트 크기의 포트번호가 포함되어 있으며, 이러한 포트번호는 컴퓨터에서 동작 중인 특정 응용서비스를 식별하는 데에 사용된다.

- 즉, 컴퓨터에는 웹서비스, 메일서비스 등의 다양한 응용서비스가 동시에 실행될 수 있는데, 그 중에 어떤 응용서비스에 대한 정보를 패킷이 포함하고 있는 지를 나타내기 위해서 포트번호가 TCP, UDP 헤더에 포함되는 것이다.



<포트번호를 활용한 응용서비스 식별>

- 위 그림에서 송신자는 IP 주소 1.1.1.1을 가지는 수신자(웹서버)에게 데이터를 전달하기 위해, 포트번호를 80번으로 설정하여 패킷을 전송하고 있다. 해당 패킷이 라우팅 과정을 통해 수신컴퓨터에 도착하였을 때에, 수신컴퓨터는 포트번호가 80번임을 확인하고 상위 웹 응용 프로그램에 데이터 내용을 전달한다[12].

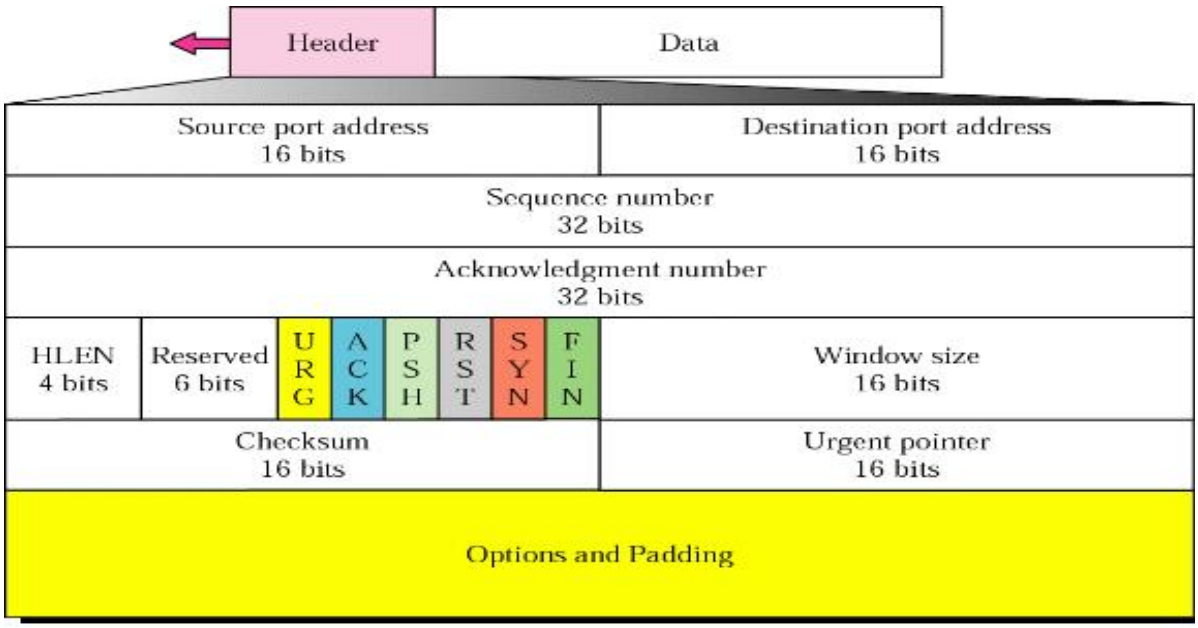
12) 주요 서비스별로 포트번호가 정해져 있는데 이를 'well-known 포트번호'라 한다. (웹서비스: 80번, 메일서비스: 25번, 파일전송 서비스: 20번 혹은 21번 등)

(6) Transmission Control Protocol (TCP)

A. TCP 특징 및 헤더 구조

- TCP는 가장 널리 사용되는 수송계층 프로토콜이다. UDP는 프로세스간 통신을 위해 포트번호 정보만 제공하는 반면에, **TCP는 연결(connection) 관리, 오류(error) 제어 및 흐름(flow) 제어기능, 혼잡(congestion) 제어 기능 등의 다양하고 복잡한 기능을 제공**한다.

- UDP는 '메시지(message)' 단위의 데이터 전송을 수행하는 반면에, TCP는 일련의 대용량 스트림(stream) 데이터 전송을 수행한다[13]. 이처럼 다양한 기능 제공을 위해 TCP 헤더는 다양한 필드로 구성된다[14].



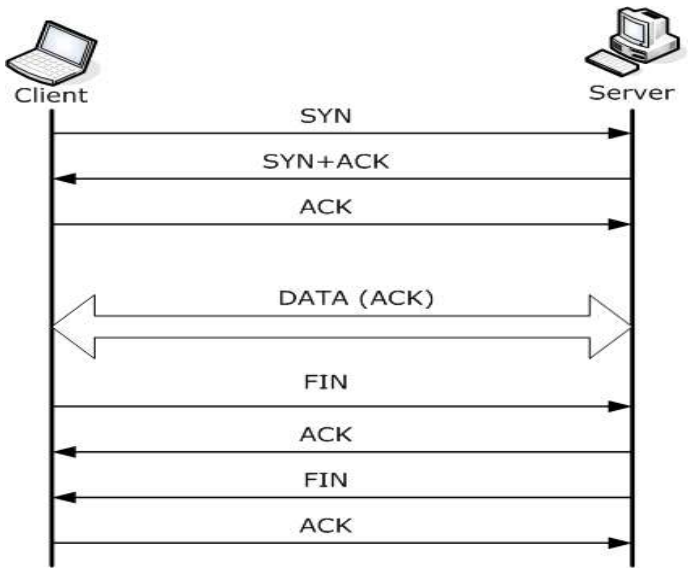
<TCP 패킷 헤더 (20 바이트)>

13) UDP는 주로 적은 데이터를 실시간(real-time) 전송에 활용되고, TCP는 대용량 데이터의 신뢰 전송(reliable transport) 서비스에 활용된다.

14) TCP 헤더는 20바이트 기본 필드와 옵션으로 구성되며, 맨 위 4바이트 필드에 송수신 포트번호를 포함한다. 그 밖에 오류제어 및 흐름제어와 관련된 필드를 포함한다 (세부 내용에 대한 설명은 생략함)

B. TCP 연결 관리(connection management)

- UDP와 달리 TCP은 연결관리 기능을 제공하며, 이를 통해 데이터 송수신 상태를 지속적으로 관리한다. 아래 그림은 TCP 연결관리 과정을 보여준다.



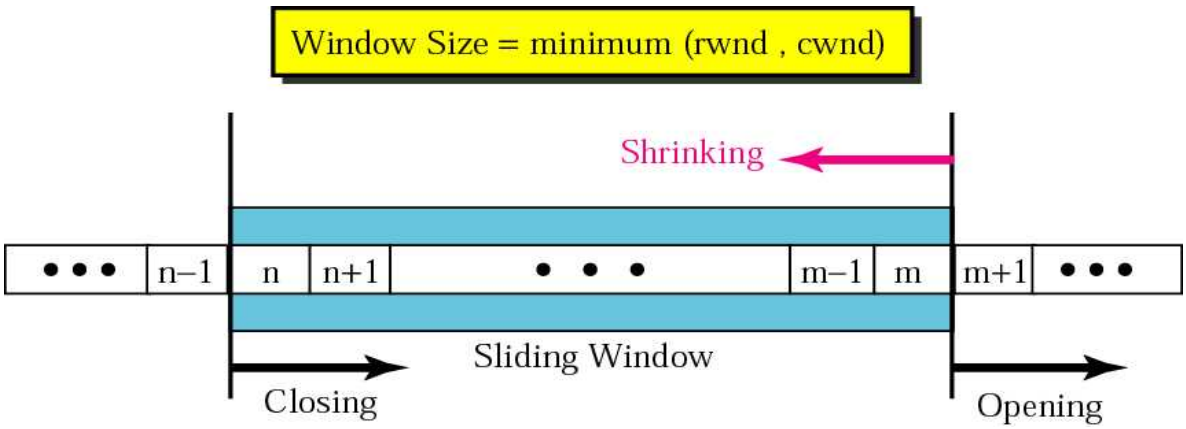
<TCP 연결 설정 및 해제 과정>

TCP 연결관리 과정

- **(연결 설정)** client가 먼저 SYN(synchronization) 패킷을 전송하면(1), server는 이에 대한 ACK(acknowledgment) 정보와 자신의 SYN 정보를 담은 패킷을 전송한다(2). 이어서 client는 서버의 SYN에 대한 ACK를 전송함으로써 연결 설정을 완료한다(3). 이러한 TCP 과정을 3-way handshake 과정이라 부른다.
- **(데이터 송수신)** 연결 설정 후에 client와 server는 데이터 패킷을 교환하고 상응하는 ACK 패킷을 통해 데이터의 성공적은 수신 여부를 상대방에 통보한다.
- **(연결 종료)** 연결 종료를 위해 client는 FIN 패킷을 전송하고(1), server는 ACK로 응답한다(2). 이어서, server가 FIN 패킷을 전송하면 (3) client가 ACK로 응답하여 연결을 종료한다(4). 이러한 과정을 4-way handshake 과정이라 부른다.

C. TCP 오류 제어 및 흐름 제어

- TCP는 UDP와 달리 송신한 패킷이 손실(loss)되었을 경우, 재전송(retransmission)을 통해 오류를 복구한다. 이러한 오류제어(error control) 기능과 함께 수신자의 버퍼 용량을 고려한 데이터 전송을 수행하는데, 이를 "sliding window 기반 흐름 제어(flow control)"라 한다[15].



<TCP 흐름제어(Sliding Window 기반)>

TCP 흐름제어(Flow Control)

- 그림에서 n-1 번째 바이트까지의 데이터는 이미 전송이 완료된 상태이고, 송신자는 추가적으로 "n ~ m 번째" 데이터를 보낼 수 있으며, 이를 Window 크기라 한다. 이러한 Window 크기는 수신자의 버퍼용량에 의해 결정된다.
- 송신된 데이터에 대하여 수신자로부터 해당 ACK 패킷이 (TCP 헤더의 ACK number를 이용) 도착하면, 전송이 완료된 것으로 간주하고 Window를 전체적으로 오른쪽으로 이동시킨다(이를 'sliding'이라 함).
- 만약 특정 시간 내에 ACK 패킷이 오지 않는 경우, 송신자는 해당 패킷을 '재전송'함으로써 오류 복구기능을 수행한다.

15) TCP는 오류제어 및 흐름제어 외에도 혼잡제어(congestion control) 기능도 제공한다.

3. IoT 메시지 전송 프로토콜

- IoT 센서는 측정한 데이터를 플랫폼으로 전송하기 위해 TCP/IP의 응용 계층 프로토콜로서 HTTP, MQTT 혹은 CoAP을 사용한다.

- 초기에는 TCP 기반의 HTTP 및 MQTT가 널리 사용되었으나, 최근에는 소형 센서를 고려하여 가벼운 UDP 기반의 CoAP[16] 프로토콜이 널리 사용되고 있다. 3가지 프로토콜의 특징을 정리하면 다음과 같다.

<HTTP, MQTT 및 CoAP의 특성 비교>

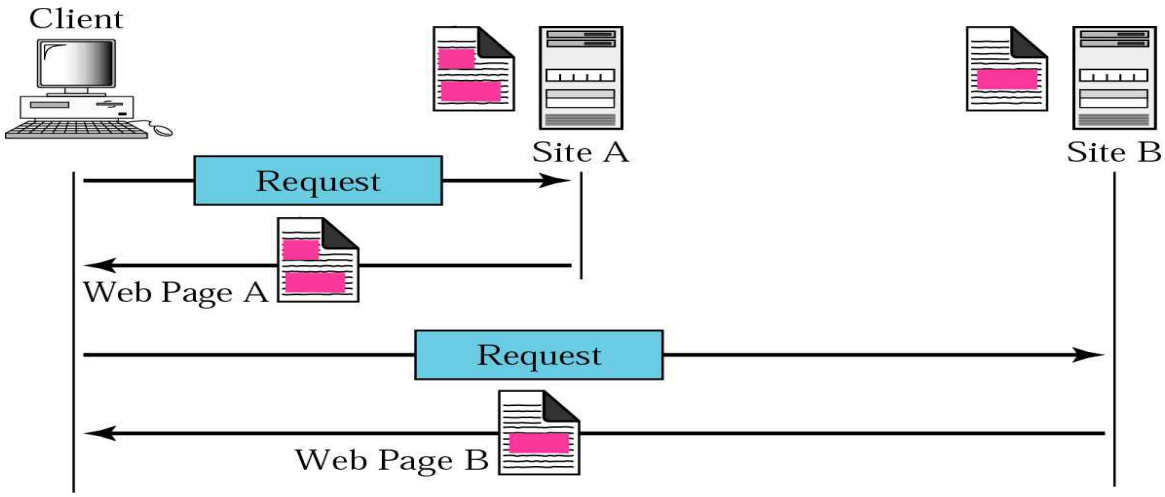
프로토콜	표준기구	전송 모델	하위 프로토콜	보안 기능
HTTP	IETF	GET/POST	TCP	TLS
MQTT	OASIS	Publish-Subscriber	TCP	TLS
CoAP	IETF	GET/POST	UDP	DTLS

A. HTTP (HyperText Transfer Protocol)

- HTTP는 인터넷 상에 WWW(World Wide Web) 혹은 웹문서를 전달하기 위해 개발된 프로토콜로서 Client-Server 구조를 따른다[17].

- 웹 Client는 웹서버에게 WWW 웹문서를 요청(Request)하면, 웹서버는 해당(URL) 웹문서로 응답(Response)한다. 웹 클라이언트는 여러 웹서버와 동시에 통신을 수행할 수 있다.

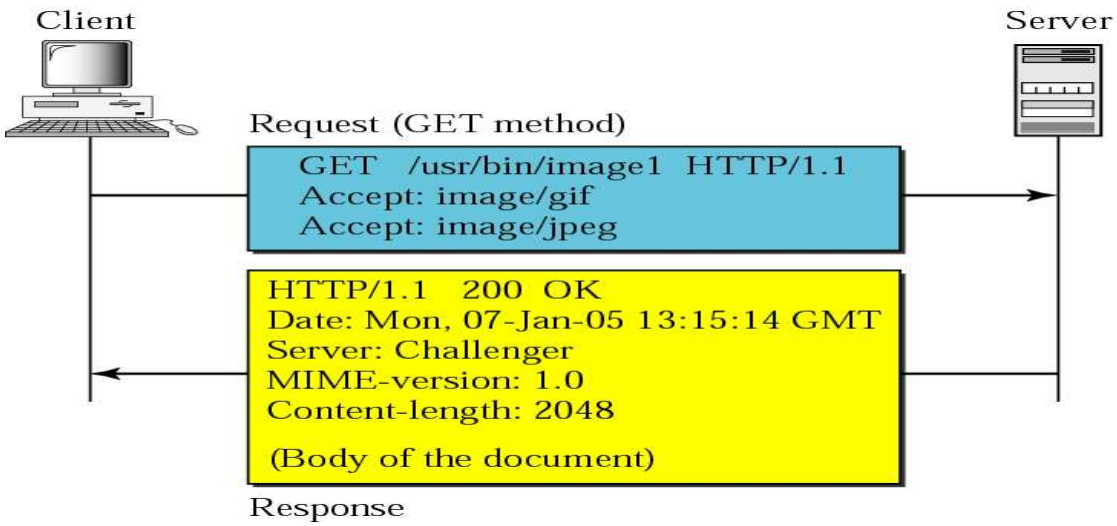
16) CoAP 프로토콜은 하위 전송계층 프로토콜로서 UDP를 사용하나 TCP를 사용하기도 한다.
17) HTTP는 하위 전송 프로토콜로서 TCP를 사용하기 때문에 신뢰전송을 보장한다.



<WWW 서비스 구조>

- 웹문서는 Client와 서버사이에 HTTP 프로토콜을 사용하여 전달된다. Client는 Request(요청) 메시지를 통해 웹문서를 요청하고, 서버는 Response(응답) 메시지를 통해 해당 웹문서를 전달한다[18].

- HTTP 요청메시지와 응답메시지의 구성 예시는 다음과 같다.



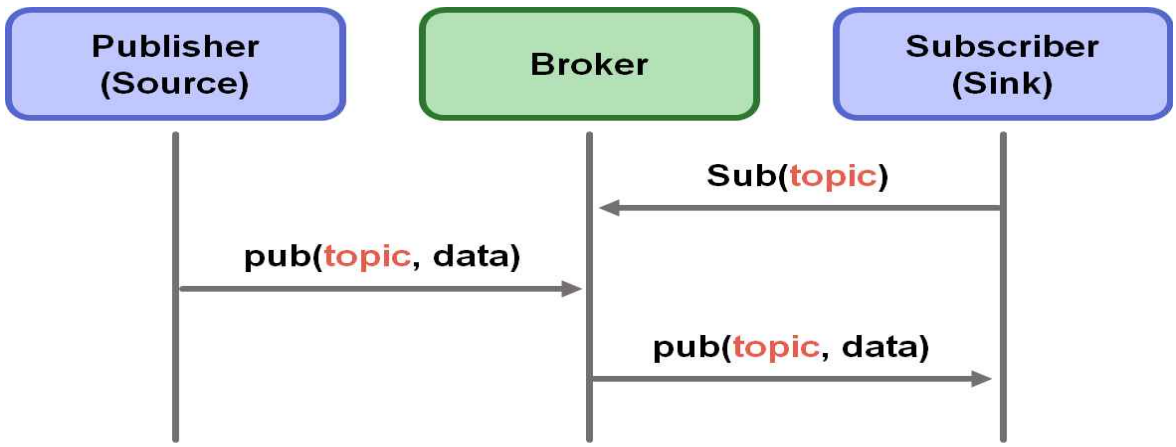
<HTTP 요청(Request) 및 응답(Response) 메시지>

18) HTTP 요청-응답 메시지 교환 절차를 '트랜잭션(transaction)'이라고 한다.

B. MQTT (Message Queue Telemetry Transport)

- MQTT 프로토콜은 사물인터넷 등에서 사용하기 위해 개발한 **<Publish(발행/게시)-Subscribe(가입/구독)> 기반의 경량화 프로토콜**이다. HTTP에 비해 적은 오버헤드로 패킷을 처리할 수 있으므로 낮은 대역폭 환경에서 저전력 디바이스에 사용될 수 있다.

- MQTT 통신을 위해 **브로커(Broker)가 중재 기능**을 수행한다. 특정 토픽(Topic)[19]에 관심이 있는 Subscriber는 브로커에 해당 토픽의 구독 메시지를 브로커에 전달한다. 이후 Publisher가 해당 토픽으로 데이터 메시지를 전달하면 브로커는 토픽을 구독하고 있는 모든 Subscriber에게 토픽 데이터를 전달한다.



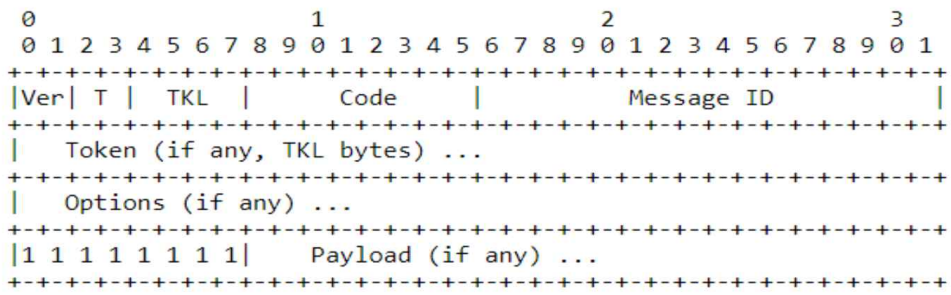
<Publish-Subscribe 기반의 MQTT 전송 모델>

- MQTT 통신을 위해서 Publisher/Subscriber는 브로커와 연결을 유지해야 한다는 단점이 있지만, 여러 토픽을 동시에 관리하는 경우 좋은 성능을 기대할 수 있다. 이러한 특성은 사물인터넷 서비스에 적합한 구조이다.

19) 토픽은 IoT 서비스 항목에 해당한다. 예를 들면, 온도, 습도 등이 이에 해당한다.

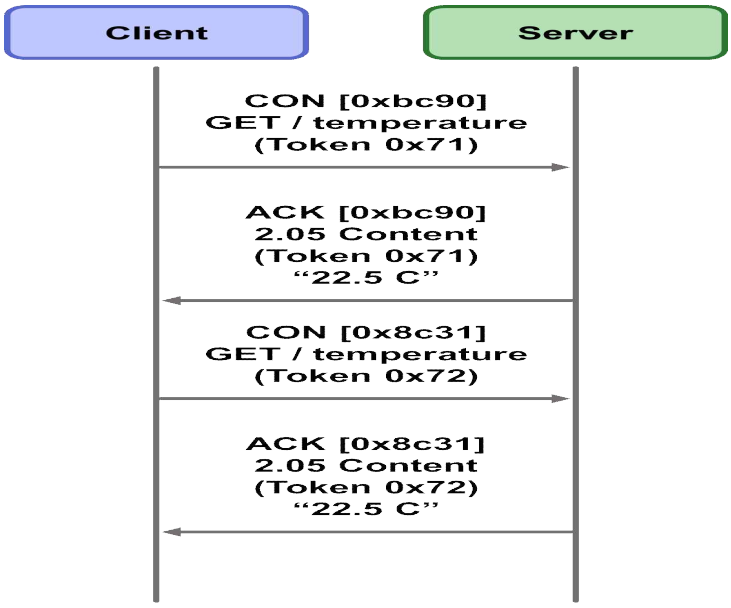
C. CoAP (Constrained Application Protocol)

- IETF에서 개발한 CoAP은 저전력 센서로 구성되는 IoT 네트워크에서 메시지 전송에 적합하다. CoAP은 HTTP와 유사한 메시지 구조를 가진다[20].



<CoAP 메시지 구조>

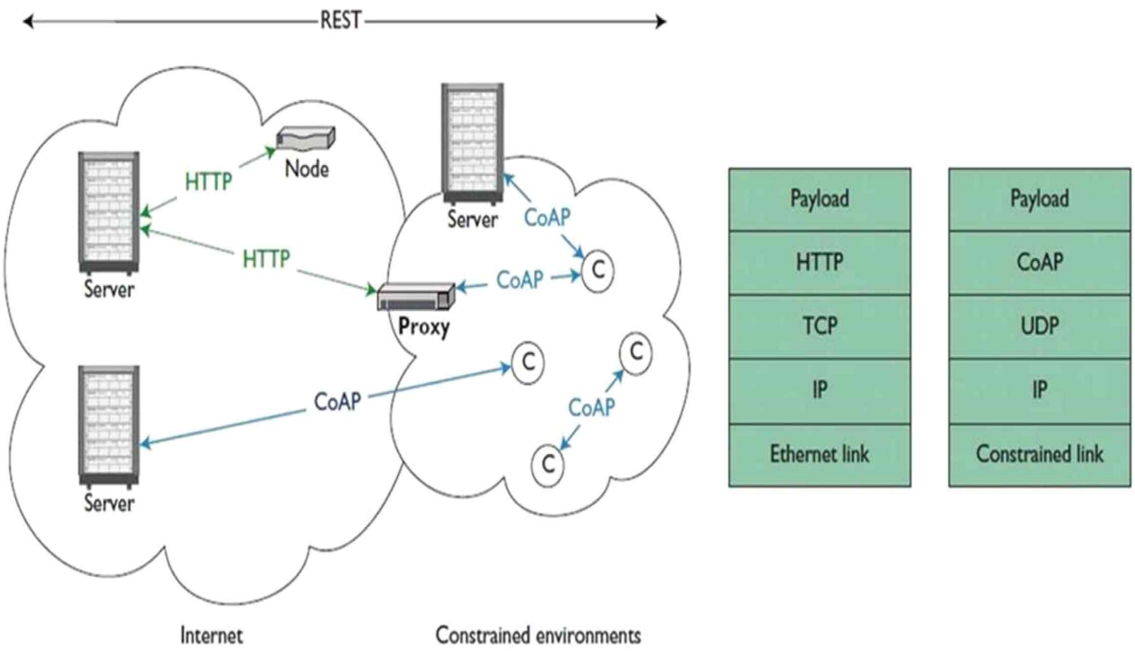
- CoAP은 UDP 기반으로 설계되었으며, 다음 그림처럼 Client-Server 모델에 따라 Client가 정보를 요청하면 Server가 응답하는 구조이다.



<CoAP 메시지 전송 모델>

20) CoAP 헤더는 4바이트의 기본 헤더와 필요에 따라 Token, Option 정보가 추가된다.

- CoAP과 HTTP는 모두 REST[21] 구조를 따르며 동시에 사용될 수 있다. 즉, 인터넷 망에서는 HTTP를 사용하고 IoT 센서로 구성되는 제한된(constrained)은 접속망에서는 CoAP를 사용한다.



<CoAP-HTTP 통신 모델>

- 위 그림에서 보여지듯이 IoT 네트워크의 센서 혹은 클라이언트(C)는 측정된 정보를 CoAP을 사용하여 서버(server)에 전송하거나, 혹은 프락시(Proxy) 서버를 경유하여 서버에 전달할 수도 있다. 이때 프락시 서버에서는 CoAP-HTTP 변환 기능을 수행한다.

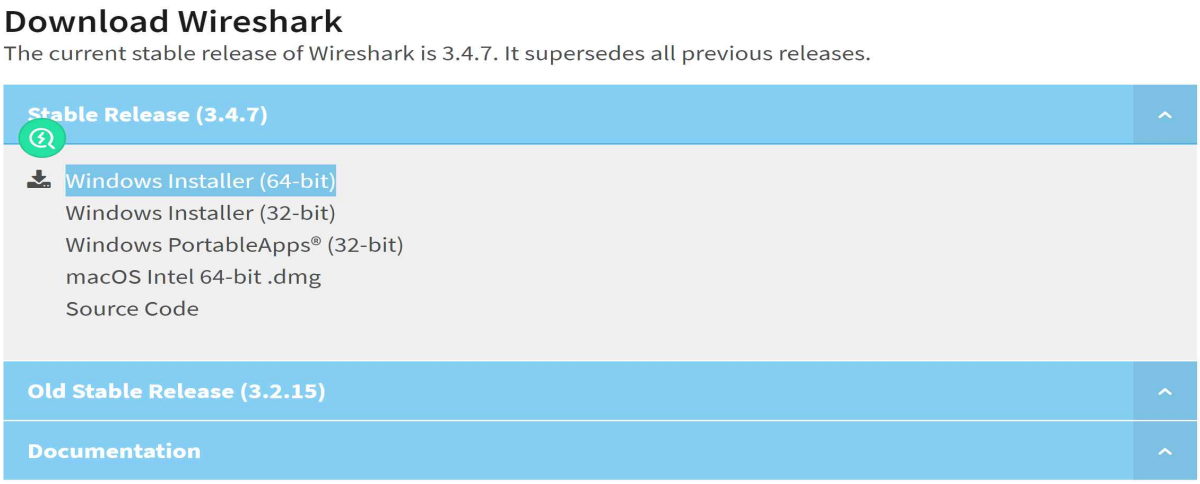
- CoAP은 일반적으로 UDP 위에서 동작하며, HTTP는 TCP 위에서 동작한다. 즉, CoAP은 센서 등의 저전력 IoT 디바이스에서 주로 사용된다.

21) REST(REpresentational State Transfer)란 디바이스 상태정보를 표현하고 전송하는 기법을 의미하여, 주로 Web 기반으로 간단한 디바이스 상태 정보를 교환할 때 사용함

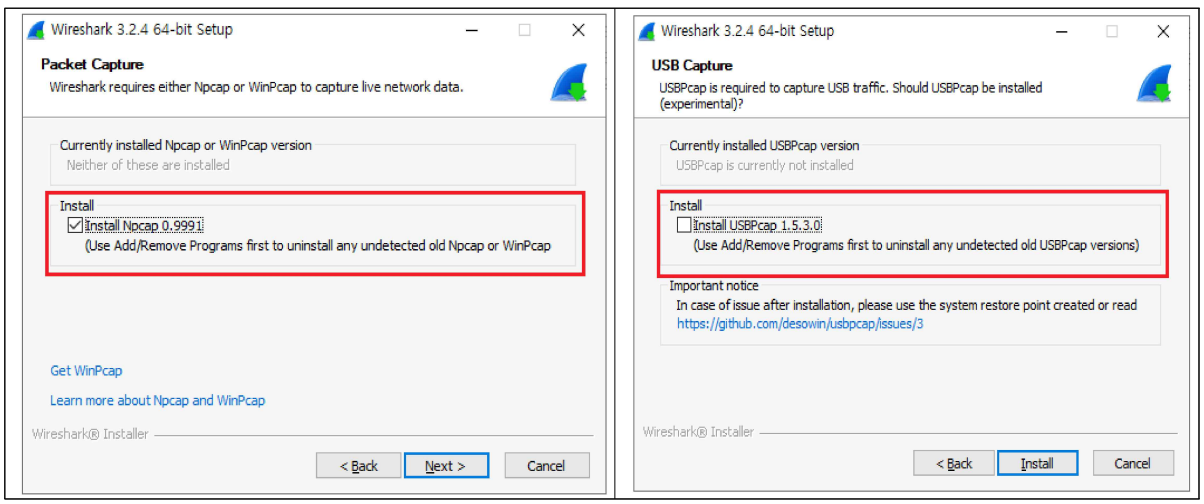
4. (실습) 인터넷 패킷 분석(Wireshark 활용)

(1) 윈도우즈 환경에서 Wireshark 설치[22)]

- https://www.wireshark.org/download.html
- Windows Installer (64-bit) 선택

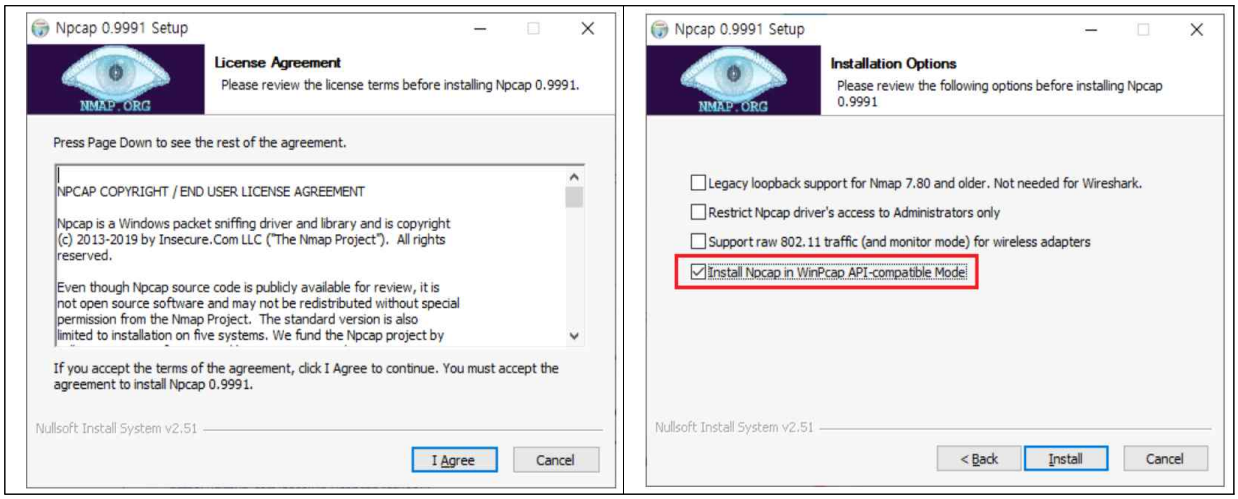


- 설치 중 Npcap 체크(필요), USBPcap은 체크 해제(불필요)

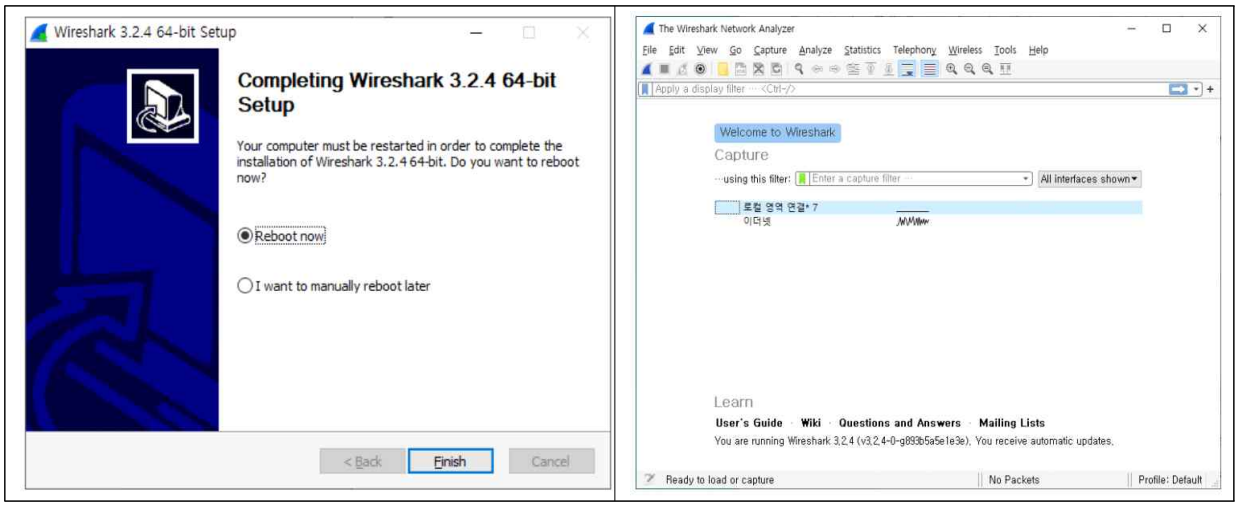


22) Wireshark(와이어샤크)는 네트워크 패킷을 캡쳐하고 분석하는 오픈소스 도구이다. 네트워크에서 패킷의 흐름을 파악하고 분석하기 위한 도구이며 통신 소프트웨어 개발 시에 널리 사용된다.

- Npcap 설치 시 Install Npcap in WinPcap API-compatible Mode 체크

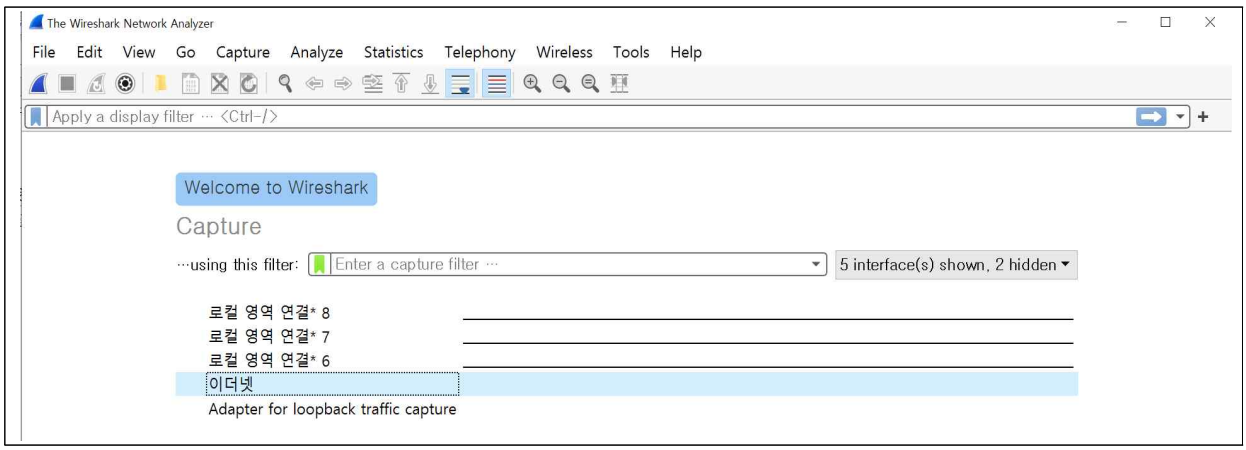


- 설치 완료 후 재부팅 및 실행

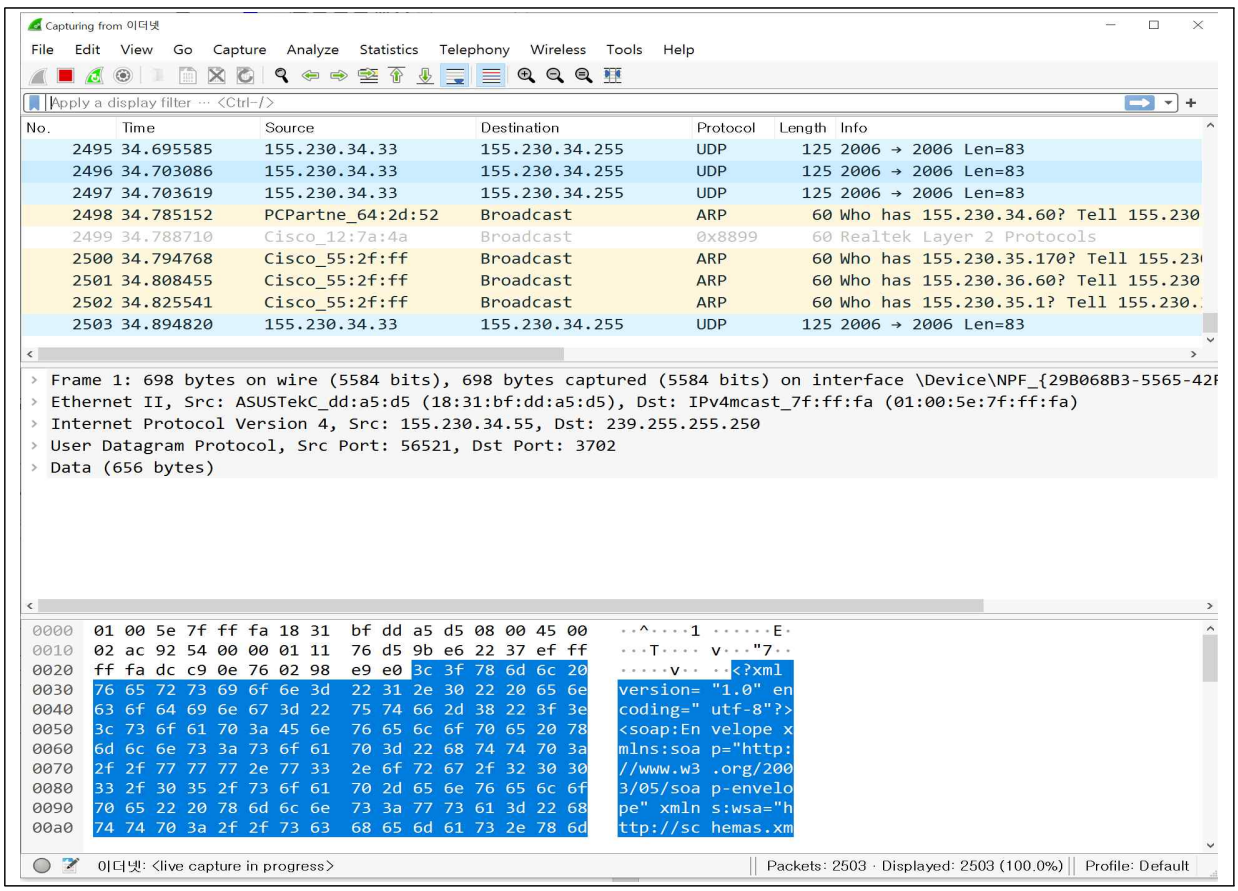


(2) 패킷 캡쳐 및 분석

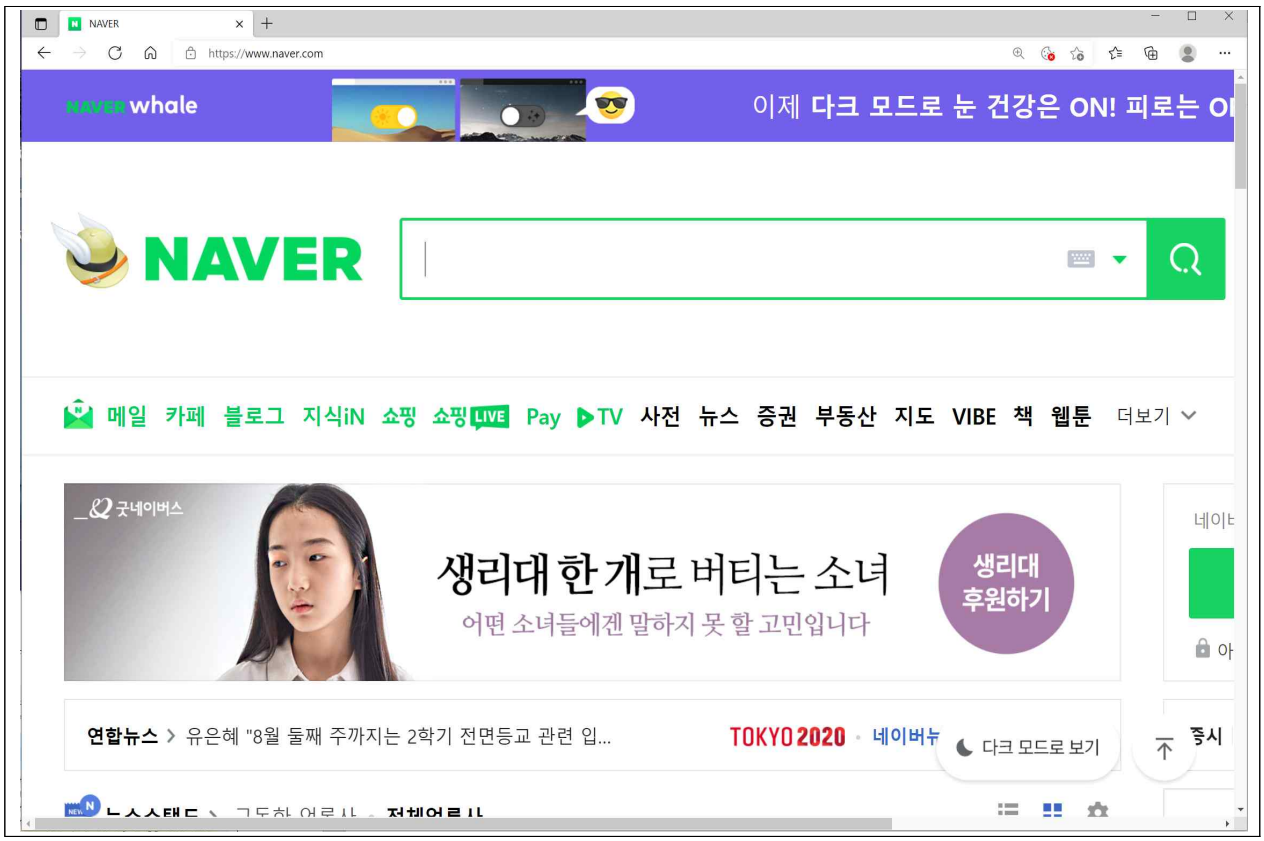
- Wireshark 실행 후 이더넷 클릭



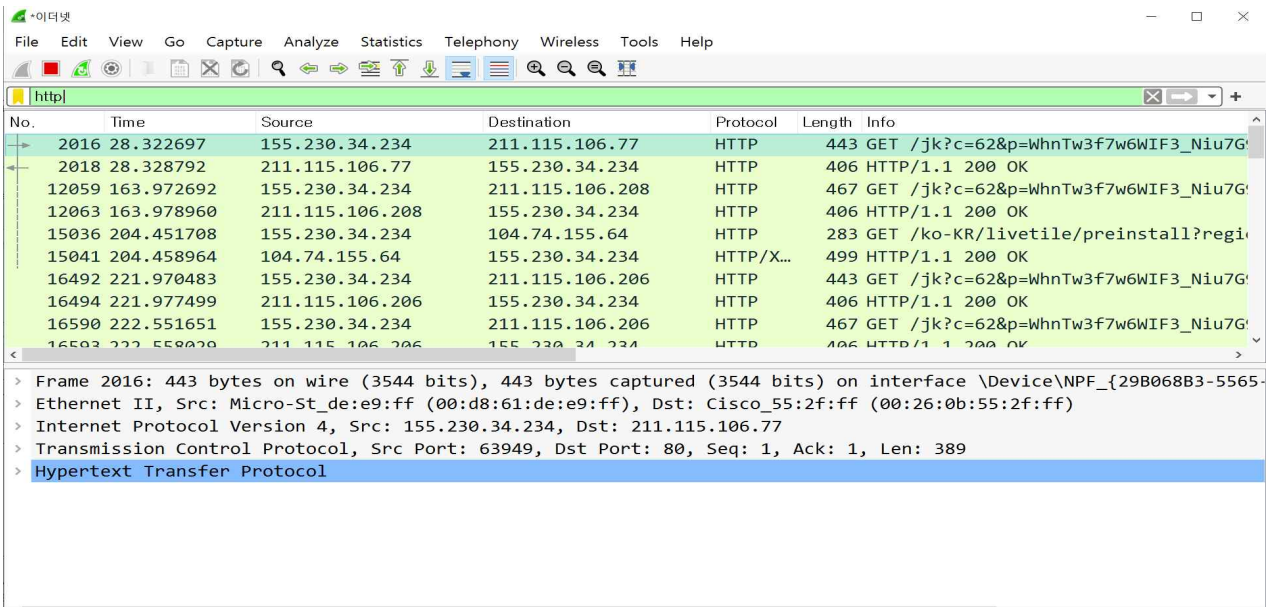
- 현재 내 컴퓨터에서 송수신 중인 패킷 확인



- 웹 브라우저를 열고 홈페이지(www.naver.com) 접속

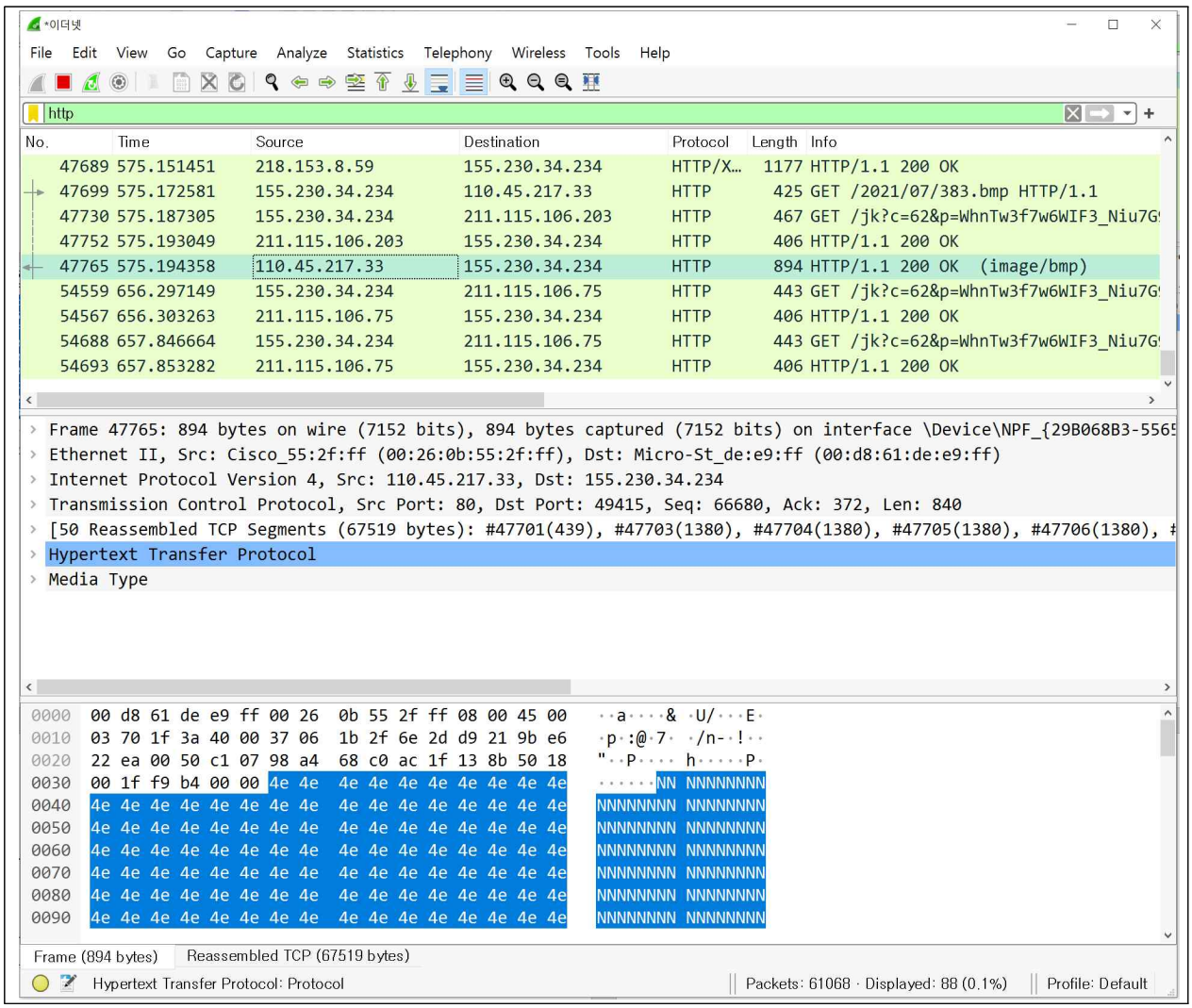


- Wireshark에서 패킷 필터(filter) 설정하기 (HTTP)



- 특정 패킷의 세부 내용 확인하기 (해당 패킷 라인 클릭)

  - 아래 화면 하단에서 패킷의 세부 내용을(16진수 형태) 확인할 수 있음[23)]



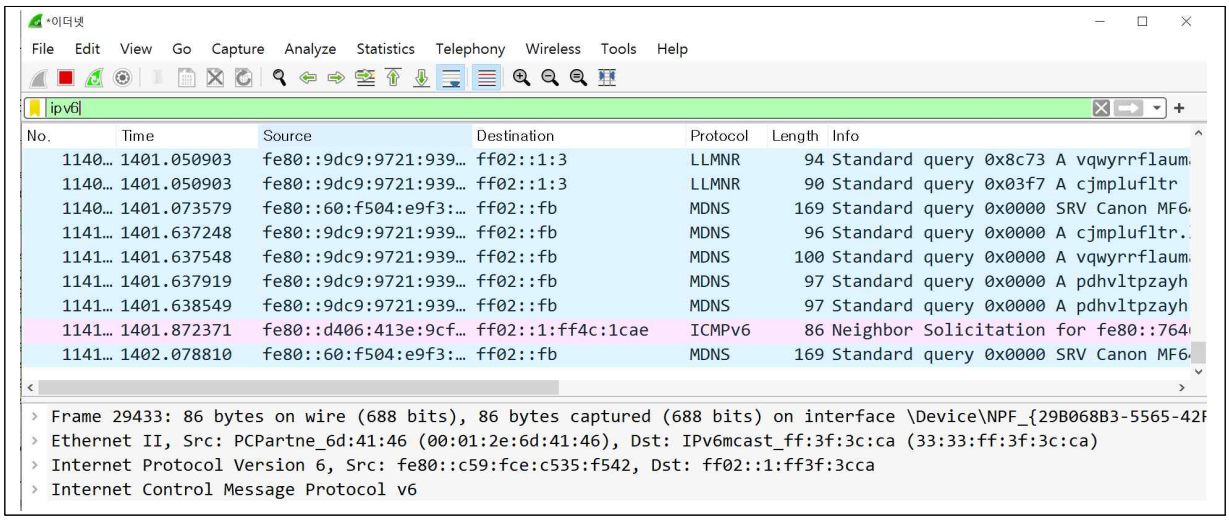
- 다른 프로토콜 동작 확인하기

  - HTTP 외에도 다양한 다른 프로토콜 패킷들도 확인할 수 있다.

  - 패킷 필터에 해당 프로토콜을 입력하고 wireshark를 확인해 보자 (예: UDP, TCP, DNS, IPv6, ICMP 등)

23) 패킷의 세부 내용 파악을 위해서는 패킷 구조(포맷)를 알아야 한다.

- IPv6 프로토콜 패킷 확인하기



- ‘ping’ 프로그램을 실행하고 ICMP 프로토콜 패킷 확인하기