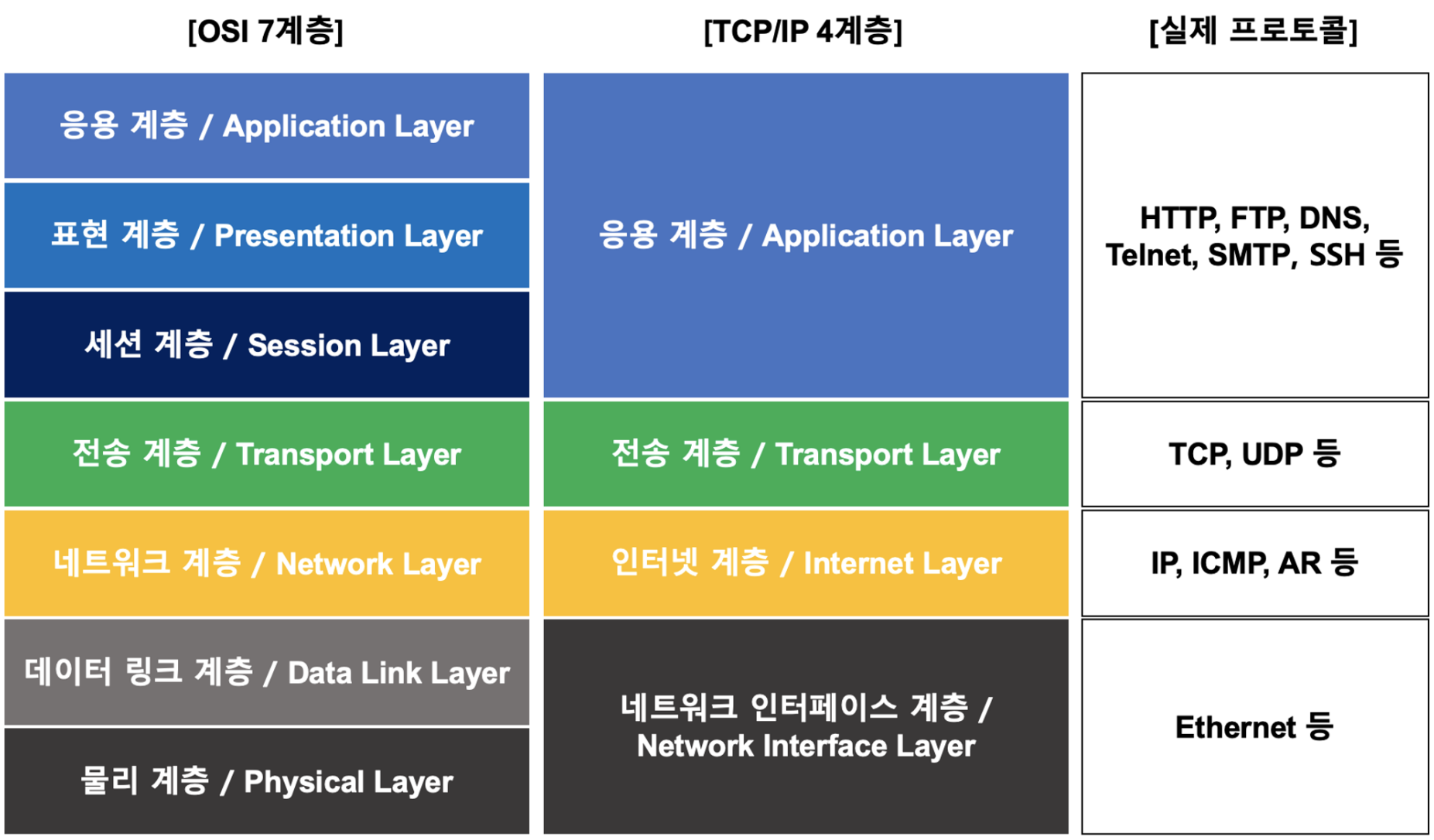
TCP / IP

OSI 7계층



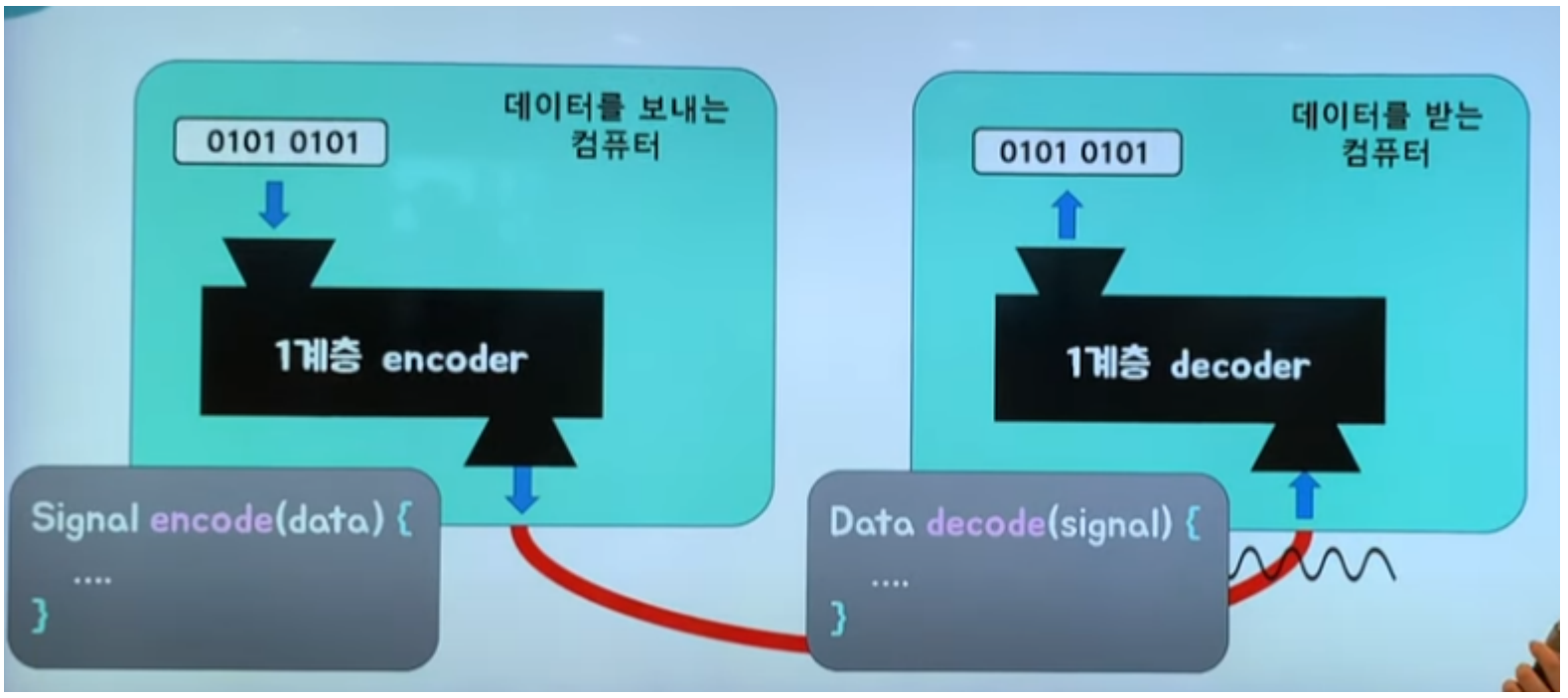
1. 물리 계층

- **[데이터 단위 bit | 프로토콜 DSL, ISDN 등]**
- 7계층 중 최하위 계층.
- 물리계층은 장치 간 **전기적 신호를 전달**하는 계층이며, 데이터 프레임 내부의 각 **bit를 한 노드에서 다음 노드로 실제로 이동**시키는 계층이다
- **데이터는 0과 1의 비트열**로 ON, OFF의 **전기적 신호 상태**로 이루어져 있다.
- **단지 데이터 전달의 역할**을 할 뿐이라 알고리즘, 오류제어 기능이 없음
- 장비로는 케이블, 리피터, 허브가 있음

▼ 추가

- **물리적으로 연결된 두 대의 컴퓨터**가 데이터를 송수신할 수 있게 해주는 모듈.
- 물리적 주소인 **MAC 주소**를 사용한다.

- 데이터를 전송하기 위해서는 0과 1을 전송할 수만 있으면 된다.
- 근데 그냥 전송하면 손실이 나는 주파수가 있기 때문에 이를 아날로그 신호로 바꿔서(인코딩) 보내고, 아날로그 신호를 받으면 이를 다시 원본 데이터로 변환(디코딩) 하는 것이 물리 계층에서 하는 일

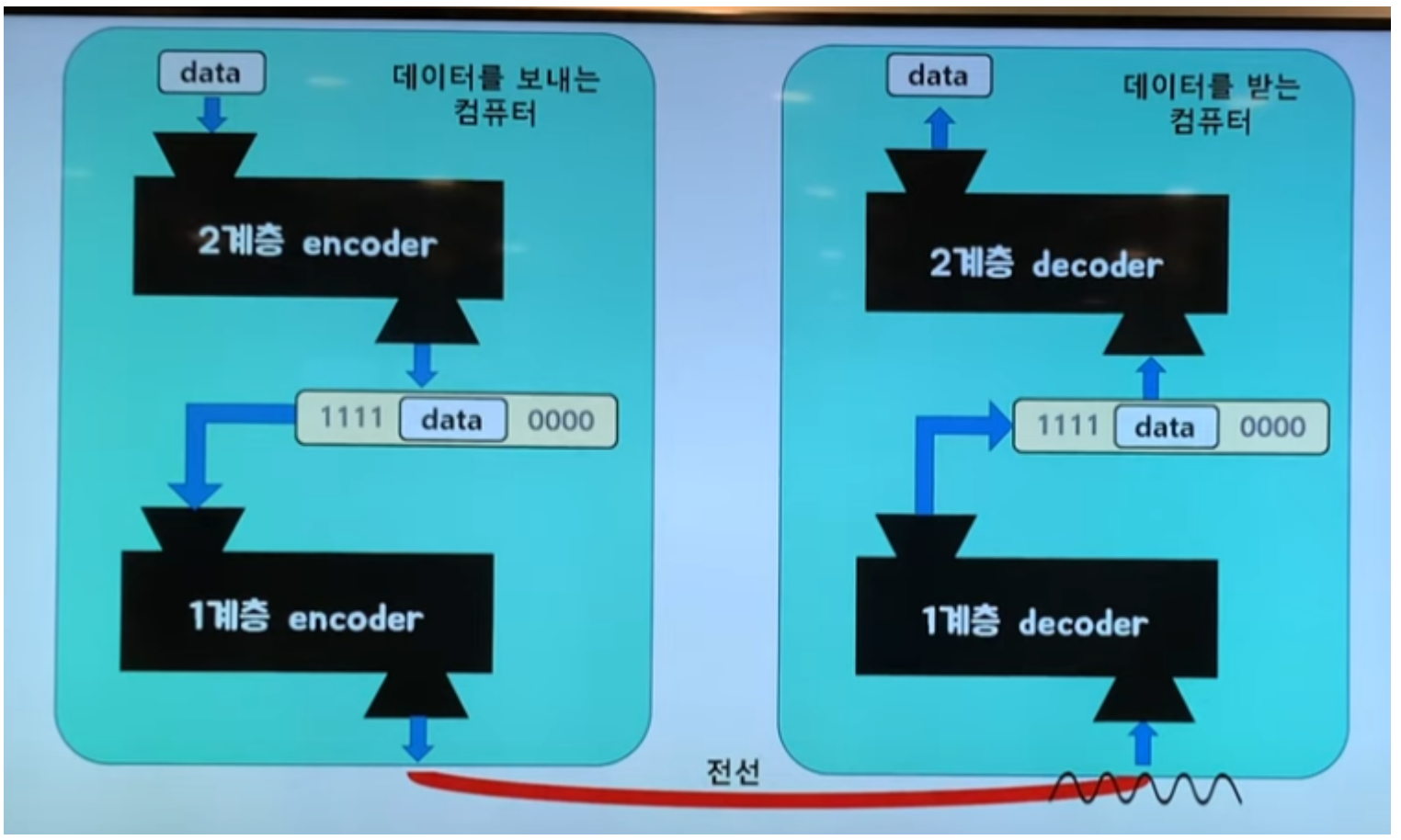


2. 데이터 링크 계층

- [**데이터 단위 frame | 프로토콜 PPP, Ethernet, Token ring, IEE 802.11(Wifi) 등]**
- 데이터링크 계층은 **물리적인 네트워크**를 통해 **데이터**를 **전송**하는 **수단을 제공**
- 1홉 통신을 담당한다고도 말한다. 홉(hop)은 컴퓨터 네트워크에서 노드에서 다음 노드로 가는 경로를 말한다. 1홉 통신이면 한 라우터에서 그다음 라우터까지의 경로를 말한다. 주목적은 물리적인 장치를 식별하는 데 사용할 수 있는 **주소 지정 체계를 제공**하는 것이다.
- 데이터 링크 계층은 **포인트 투 포인트** 간의 **신뢰성 있는 전송**을 보장하기 위한 계층으로 **CRC 기반의 오류 제어와 흐름 제어**가 필요.
- MAC 주소를 통해서 통신, 주소 값은 물리적으로 할당 받는데, 이는 **네트워크 카드가 만들어질 때부터 맥 주소(MAC address)가 정해져 있다**는 뜻이다.
- 장비로는 브리지, **스위치**가 있음

▼ 추가

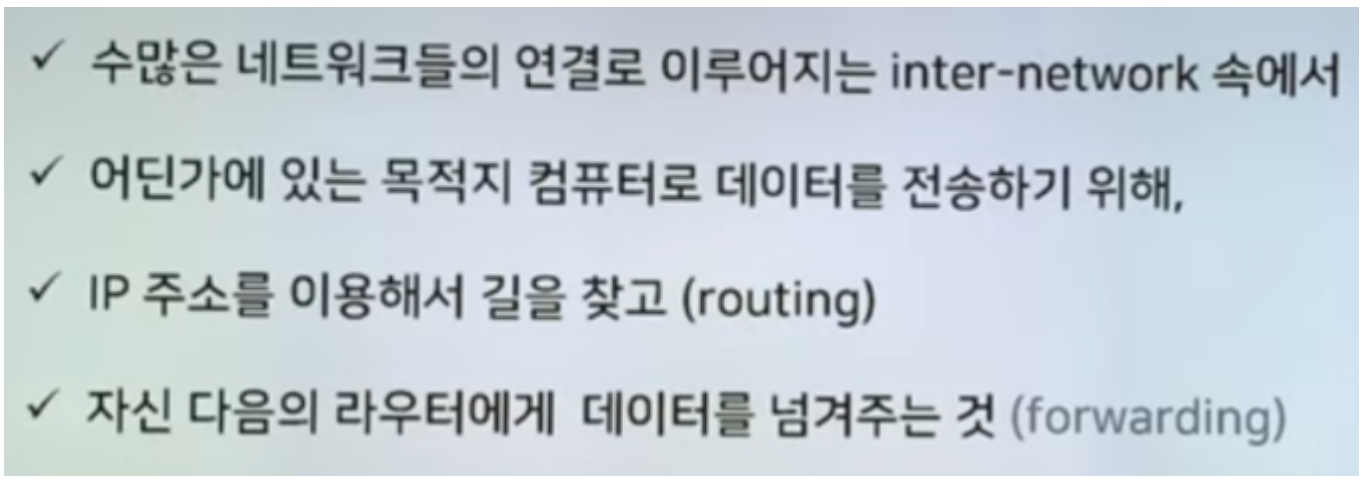
- **같은 네트워크에 있는 여러 대의 컴퓨터**들이 데이터를 주고받기 위해 필요한 모듈
- a,b,c가 같은 네트워크에 연결되어 있을 때(= 같은 라우터에 연결되어 있을 때), a가 c한테 데이터를 보내고 싶은데 같은 네트워크에 있는 b도 이를 볼 수 있는 문제가 생김. 그래서 **데이터에 목적지를 같이 보내면 스위치라는 장치가 이를 확인**해서 목적지에 데이터를 보내줌
- 여러 대의 컴퓨터가 a라는 목적지를 갖고 같은 스위치로 데이터를 보내면 모든 데이터가 쭉 연결된 상태로 전달되기 때문에 데이터를 잘 끊어 읽는 것이 중요함. 그래서 데이터의 **앞,뒤에 특정한 비트열**을 붙임. 1111 data 0000 이면 아래 그림처럼 데이터를 뽑을 수 있음



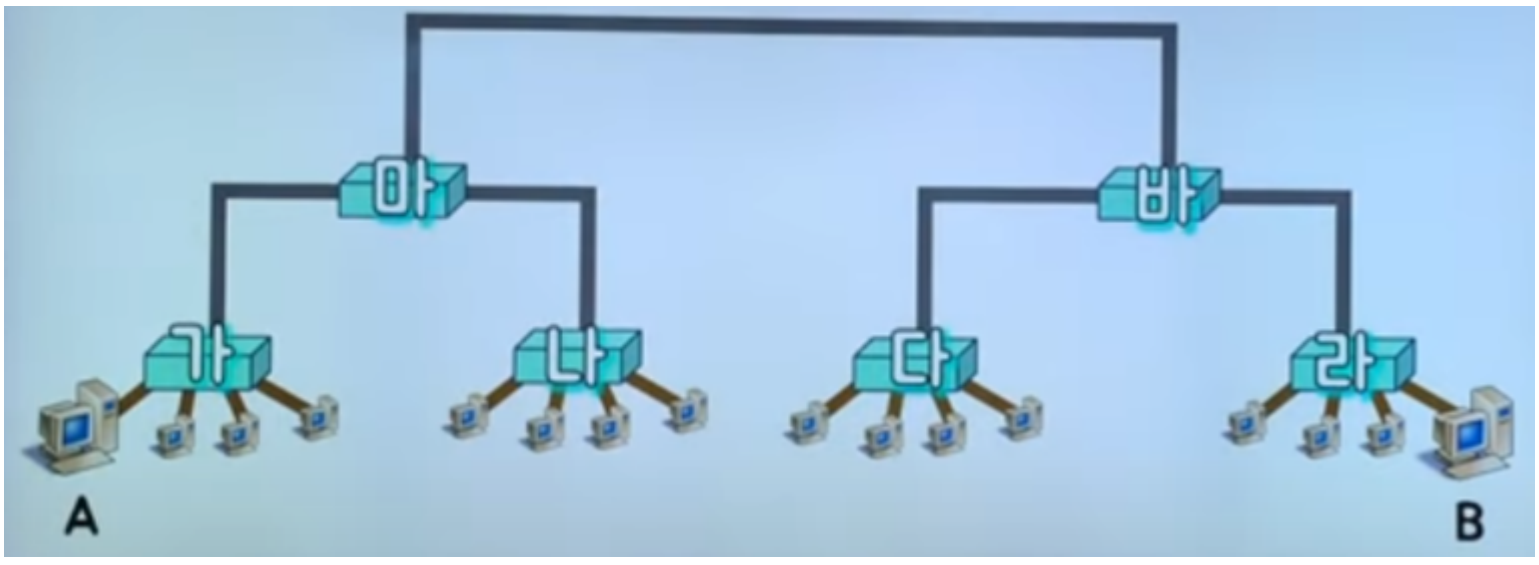
3. 네트워크 계층

- [**데이터 단위 datagram, packet | 프로토콜 IP, ICMP, ARP, RIP, BGP 등**]
- 네트워크 계층은 여러개의 노드를 거칠때마다 **경로를 찾아주는 역할**을 하는 계층으로 목적지까지 **가장 안전하고 빠르게 데이터를 보내는 기능**을 가지고 있음(**최적의 경로를 설정가능=라우팅**)
- 다양한 길이의 **데이터를 네트워크들을 통해 전달**하고, 그 과정에서 전송 계층이 요구하는 **서비스 품질(QoS)**을 제공하기 위한 **기능적, 절차적 수단을 제공**한다.
- 네트워크 계층은 **라우팅, 흐름 제어, 세그멘테이션(segmentation/desegmentation), 오류 제어, 인터네트워킹(Internetworking)** 등을 수행한다.
- 컴퓨터에게 **데이터를 전송할지 주소**를 갖고 있어서 **통신**가능(=우리가 자주 듣는 IP 주소가 바로 네트워크 계층 헤더에 속함)
- 장비로는 라우터, L3 스위치가 있음

▼ 추가

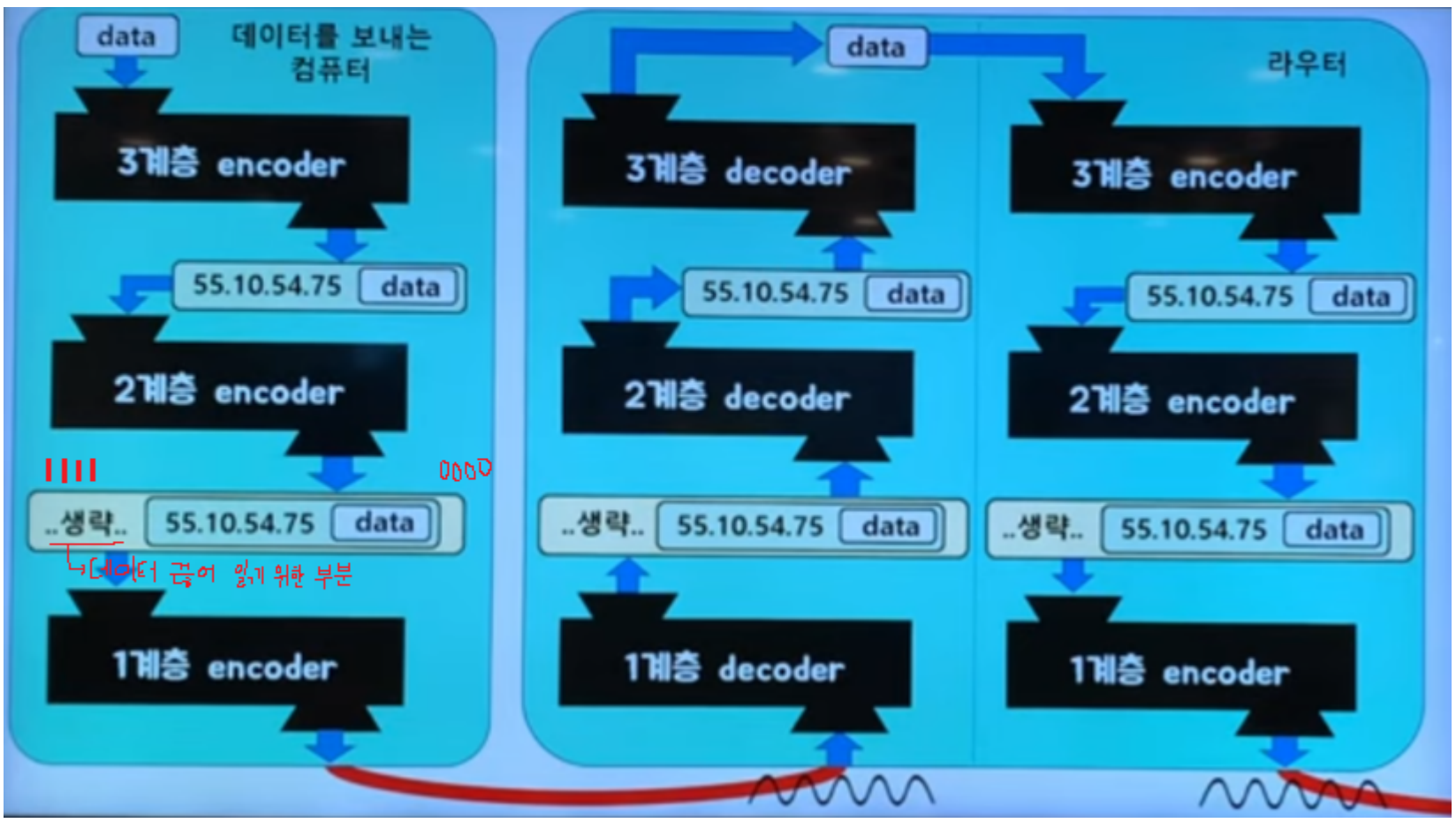


⇒ **IP 주소를 이용**해 **서로 다른 네트워크에 속한 컴퓨터끼리** 데이터를 주고 받게 해주는 것



- A가 B에게 데이터를 보내려고 함
- 데이터 앞에 목적지 B의 IP 주소를 붙임 (`11.22.33.44 data`) → 상대의 IP 주소를 알아야 보낼 수 있다.
    - IP 주소를 어떻게 알까? B가 네이버 서버에 있는 컴퓨터라고 생각을 해보자. 그럼 www.naver.com이라고 입력하면 DNS를 통해 IP 주소로 변환돼서 사용된다. 결론적으로 www.naver.com을 알고 있다면 즉, 도메인을 알고 있다면 IP 주소를 아는 것과 같다.
    - 패킷 = 목적지 IP 주소 + data (`11.22.33.44 data`)

1. **A가 라우터 ‘가’에게 패킷을 전달**합니다.
2. 패킷을 받은 **라우터 ‘가’는 패킷을 열어 목적지 IP 주소를 확인**합니다.
3. 라우터 ‘가’는 **자신과 연결된 컴퓨터 중에 목적지 IP 주소와 같은 IP 주소를 가진 컴퓨터가 있는지 확인**합니다. 없는 걸 확인하고 데이터를 **다시 포장한 다음, 자신과 연결된 유일한 라우터인 ‘마’에게 전달**합니다.
4. 패킷을 받은 **라우터 ‘마’는 패킷을 열어 목적지 IP 주소를 확인**합니다.
   라우터 ‘마’는
   라우팅(네트워크에서 경로를 선택하는 프로세스)을 통해 패킷이 B에 도착하려면 어느 전선으로 패킷을 내보내야 하는지 알아냅니다.
5. **라우터 ‘마’는 데이터를 다시 패킷으로 포장해서 라우터 ‘바’에게 넘겨줍니다.**
6. 같은 방법을 통해 **라우터 ‘바’는 라우터 ’라’에게 패킷을 보냅니다.**
7. 라우터 ‘라’는 **자신과 연결된 컴퓨터 중에 목적지 IP 주소와 같은 IP 주소를 가진 컴퓨터가 있는지 확인**하고 **B에게 패킷을 전달**합니다.



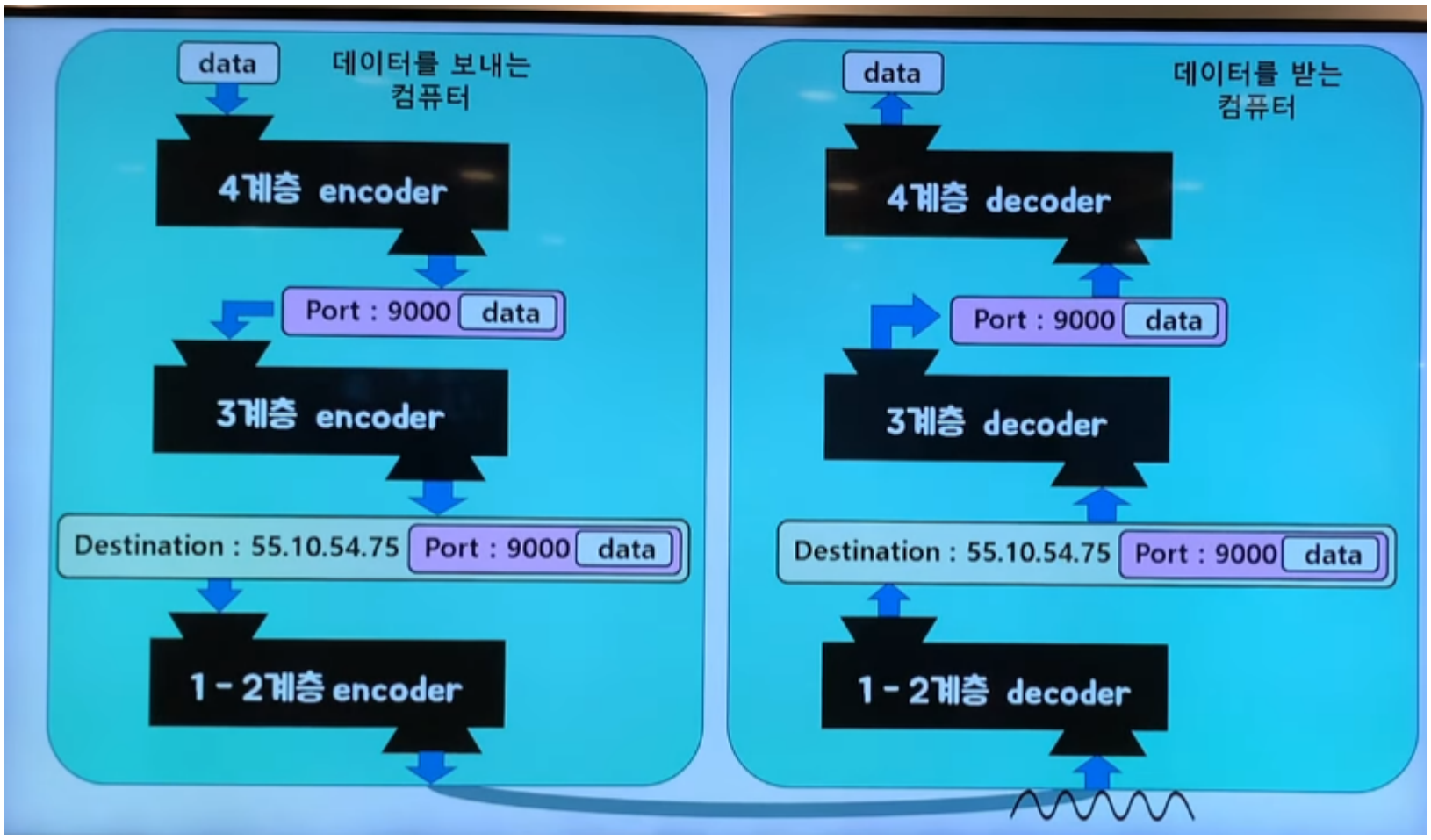
4. 전송 계층

- **데이터 단위 segment | 프로토콜 TCP, UDP, SCTP 등]**
- 전송계층은 **양 끝단(End to end)**의 사용자들이 **신뢰성있는 데이터**를 주고 받을 수 있도록 해 주어, **상위 계층**들이 **데이터 전달의 유효성이나 효율성을 생각하지 않도록** 해준다.

- **시퀀스 넘버 기반**의 오류 제어 방식을 사용하여 **메시지의 오류를 제어**
- **메시지**가 클 경우 이를 **나눠서(Segmentation)** 네트워크 계층으로 전달한다. 그리고 받은 **패킷을 재조립**해서 상위 계층으로 전달한다.
- 데이터 전송을 위해서 **Port 번호를 사용**함.(포트는 전송할 대상이 누구인지 파악하는 것)
- 대표적으로 TCP, UDP 프로토콜이 있다. **TCP**는 **연결 지향형 통신**을, **UDP**는 **비연결형 통신**을 제공한다.
- 전송 계층이 **패킷들의 전송이 유효한지 확인**하고 **전송 실패**한 패킷들을 **다시 전송**한다

▼ 추가

- **Port 번호를 사용**하여 도착지 컴퓨터의 최종 도착지인 **프로세스**에까지 데이터가 도달하게 하는 모듈



5. 세션 계층

- **[데이터 단위 message | 프로토콜 NetBIOS, TLS 등]**
- 세션 계층에서는 **두 컴퓨터 간**의 대화나 **세션을 관리**하며, **포트(Port)연결**이라고도 한다.
- **세션 계층의 프로토콜은** 연결이 손실되는 경우 **연결 복구를 시도**한다. 오랜 시간 연결이 되지 않으면 세션 계층의 프로토콜이 연결을 닫고 다시 연결을 재개한다.
- **데이터**를 상대방이 보내고 있을 때 동시에 보낼지에 대한 **전이중(동시에 보냄**, 전화기), **반이중(동시에 보내지 않음**, 무전기) **통신을 결정**할 수 있다.
- **TCP/IP 세션을 만들고 없애고** 모든 통신 장치 간에 연결을 설정하고 관리 및 종료하고 체크 포인팅과 유휴, 재시작 과정 등을 수행
- 호스트가 갑자기 중지되지 않고 **정상적으로 호스트를 연결하는 데 책임**이 있다.

6. 표현 계층

- [**데이터 단위 message | 프로토콜 ASCII, MPEG 등**]
- 표현계층은 **데이터를 어떻게 표현할지** 정하는 역할을 하는 계층
- **코드 간의 번역**을 담당하여 사용자 시스템에서 데이터의 형식상 차이를 다루는 부담을 응용 계층으로부터 덜어 준다.
- 송신자에서 온 데이터를 해석하기 위한 **응용계층 데이터 부호화, 변화**
- 수신자에서 데이터의 압축을 풀수 있는 방식으로 된 **데이터 압축**
- **데이터의 암호화와 복호화**

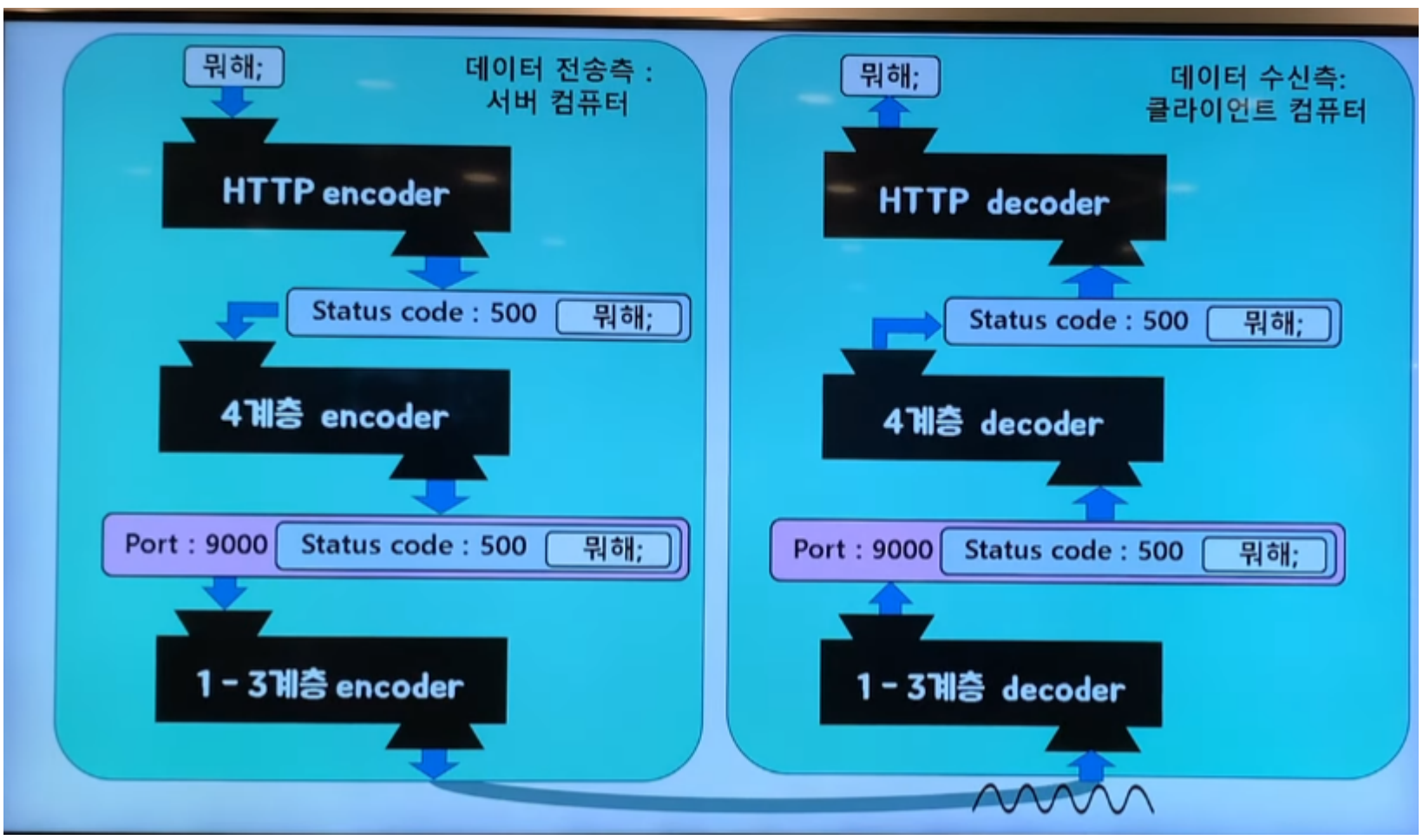
  (MIME 인코딩이나 암호화 등의 동작이 표현계층에서 이루어짐. EBCDIC로 인코딩된 파일을 ASCII 로 인코딩된 파일로 바꿔주는 것이 한 가지 예임)
- **응용 프로그램 ⇔ 네트워크 간 정해진 형식대로 데이터를 변환**, 즉 표현한다.

7. 응용 계층

- [**데이터 단위 message | 프로토콜 HTTP, SMTP, FTP, SIP 등]**
- 응용 계층에서는 **최상위 계층**으로 **사용자**가 **네트워크 자원에 접근하는 방법**을 **제공**
- **사용자**에게 **가장 가까운** 계층이며 **사용자가 볼 수 있는** 유일한 계층으로, 웹 브라우저, 응용 프로그램을 통해 **사용자와 직접적으로 상호작용**
- **응용 프로세스 간의 정보 교환**을 담당
- 많은 프로토콜이 존재하는 계층으로, **새로운 프로토콜 추가**도 굉장히 **쉽다**.
- 예) 웹 브라우저 : Chrome, Firefox 등/ 응용 프로그램 : Skype, Outlook, Office 등

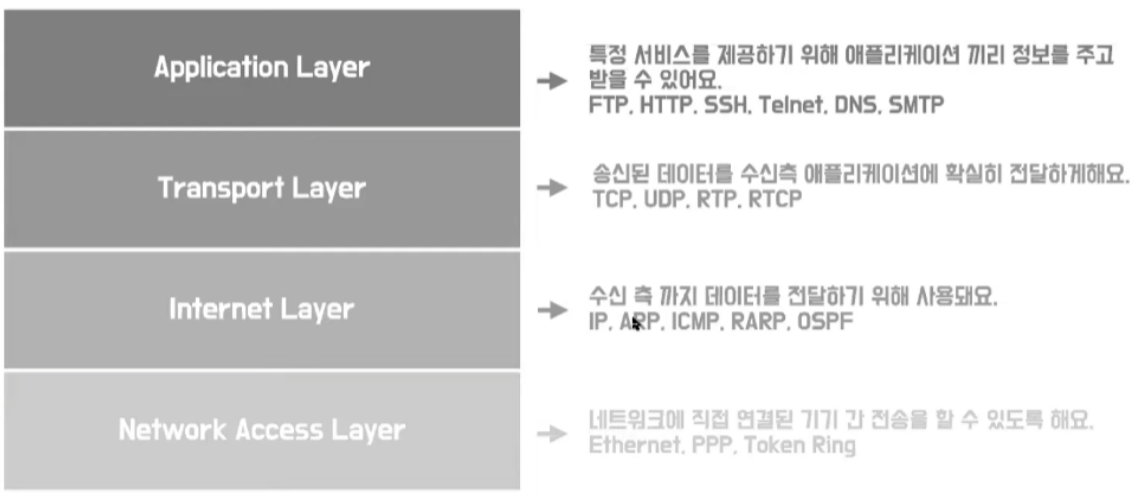
▼ 추가

- 네트워크를 통해 송수신된 이진 데이터를 인코딩, 디코딩 하는 방법(메타 정보)을 넘겨주는 것
- TCP / IP 소켓 프로그래밍
  - 운영체제의 Transport layer에서 제공하는 API를 활용해서 통신 가능한 프로그램을 만드는 것
  - 소켓 프로그래밍 만으로도 클라이어트, 서버 프로그램을 따로따로 만들어서 동작 시킬 수 있다.
  - 소켓 프로그래밍을 통해 누구나 자신만의 Application Layer 인코더와 디코더를 만들 수 있다. ⇒ 누구든 자신만의 Application Layer 프로토콜을 만들어서 사용할 수 있다.
- HTTP를 이용한 인코더와 디코더



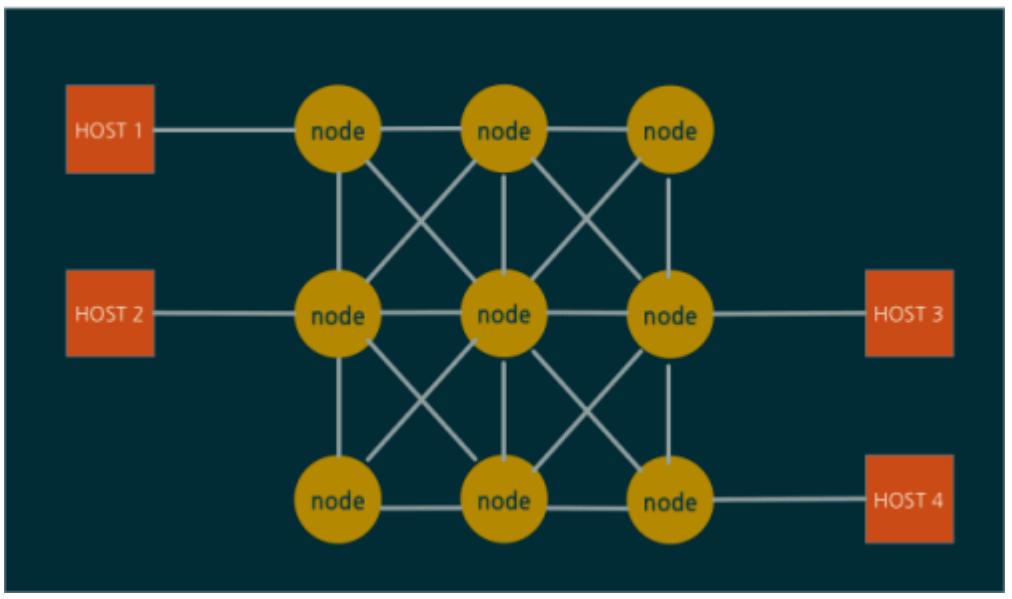
TCP/IP

[OSI 7계층]	[TCP/IP 4계층]	[실제 프로토콜]
응용 계층 / Application Layer	응용 계층 / Application Layer	HTTP, FTP, DNS, Telnet, SMTP, SSH 등
표현 계층 / Presentation Layer		
세션 계층 / Session Layer		
전송 계층 / Transport Layer	전송 계층 / Transport Layer	TCP, UDP 등
네트워크 계층 / Network Layer	인터넷 계층 / Internet Layer	IP, ICMP, AR 등
데이터 링크 계층 / Data Link Layer	네트워크 인터페이스 계층 / Network Interface Layer	Ethernet 등
물리 계층 / Physical Layer		



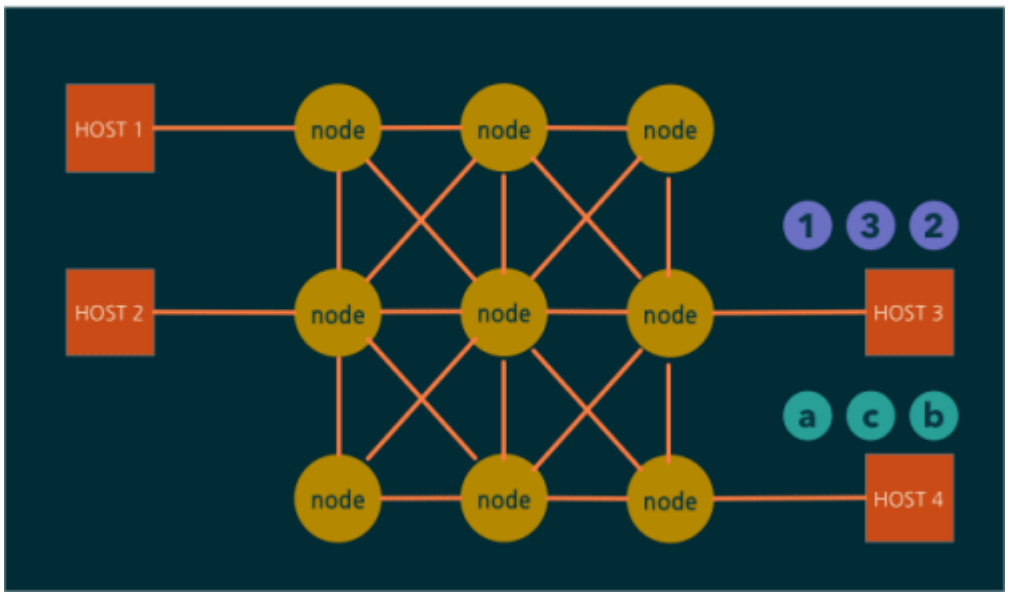
- TCP/IP를 사용하겠다는 것은 **IP주소 체계**를 따르고 IP Routing을 이용해 목적지에 도달하며, **TCP의 특성을 활용해 신뢰성 있는 통신**을 한다는 것을 말한다.
- TCP/IP는 특정 컴퓨터를 인터넷에 연결하는 방법과 컴퓨터 간에 데이터를 전송할 수 있는 방법을 결정하는 데 도움이 된다.
- HTTP, FTP, SMTP 등 TCP를 기반으로 한 많은 수의 애플리케이션 프로토콜들이 IP 위에서 동작하기 때문에, 묶어서 TCP/IP로 많이 불린다.

- **TCP**
  - Transmission Control Protocol의 약자
  - 목적지에 도착한 패킷의 순서가 뒤바뀌어있다면 다시 순서를 맞추고, 빠진 조각을 다시 요청하는 **신뢰성 있는 통신**을 한다.
  - 전송 계층의 프로토콜
  - **IP보다는 느리다.**



목적지에 도착한 패킷들은 뒤섞이거나 누락될 수 있다.

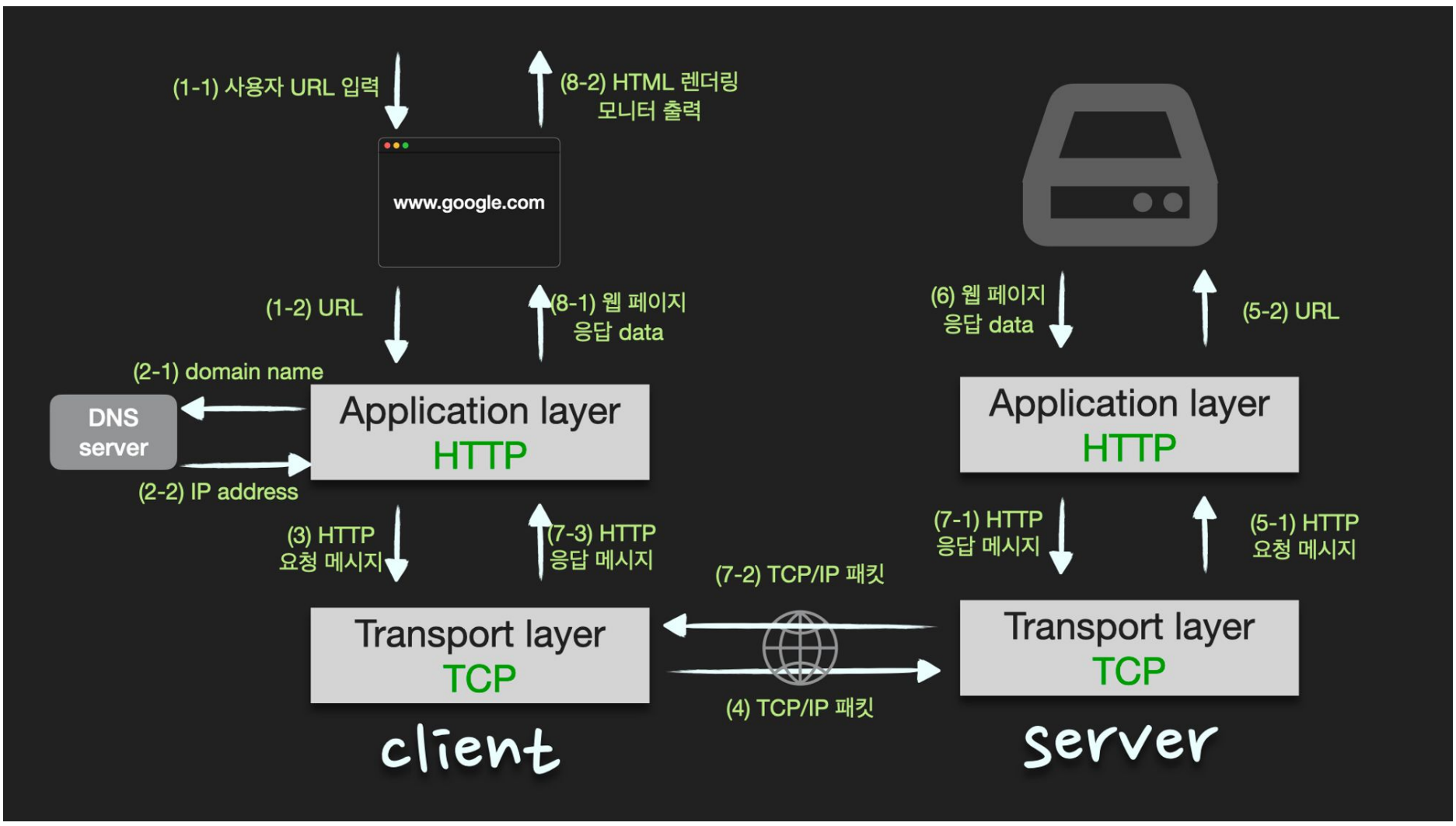
이때 TCP는 빠진 패킷을 다시 요청하고, 순서가 바뀐 패킷은 재조립한다. (아래 GIF 참고)



- **IP**
  - Internet Protocol의 약자
  - 네트워크 계층의 프로토콜
  - 패킷을 **최대한 빠르게** 목적지로 보낸다.
  - 패킷의 순서가 뒤바뀌거나, 패킷이 누락되어도 신경쓰지 않고 **보내는 데에만 집중**한다.

www.google.com을 주소창에 치면?

1. 사용자가 브라우저에 URL 입력
2. 브라우저는 DNS를 통해 서버의 IP 주소를 찾는다
3. client에서 HTTP request 메시지 ⇒ TCP/IP 패킷 생성 ⇒ server로 전송
4. server에서 HTTP requset에 대한 HTTP response 메시지 ⇒ TCP/IP 패킷 생성 ⇒ client로 전송
5. 도착한 HTTP responsse message는 웹 브라우저에 의해 출력(렌더링)



▼ 네트워크 관점에서의 대답

1. 유저가 브라우저에서 www.google.com(URL)을 입력을 하면 HTTP request message를 생성합니다.
2. IP주소를 알아야 전송을 할 수 있으므로, DNS lookup을 통해 해당 domain의 server IP주소를 알아냅니다.
3. 반환된 IP주소(구글의 server IP)로 HTTP 요청 메시지(request message) 전송 요청을 합니다.
    a. 생성된 HTTP 요청 메시지를 TCP/IP층에 전달합니다.
    b. HTTP 요청 메시지에 헤더를 추가해서 TCP/IP 패킷을 생성합니다.
4. 해당 패킷은 전기신호로 랜선을 통해 네트워크로 전송되고, 목적지 IP에 도달합니다.
5. 구글 server에 도착한 패킷은 unpacking을 통해 message를 복원하고 server의 process로 보냅니다.
6. server의 process는 HTTP 요청 메시지에 대한 response data를 가지고 HTTP 응답 메시지(response message)를 생성 합니다.
7. HTTP 응답 메시지를 전달 받은 방식 그대로 client IP로 전송을 합니다.
8. HTTP response 메시지에 담긴 데이터를 토대로 웹브라우저에서 HTML 렌더링을 하여 모니터에 검색창이 보여집니다.

TCP UDP

- TCP와 UDP는 모두 전송계층의 프로토콜이다. 전송계층은 신뢰성 있는 데이터 전송을 담당하는 계층이다.
    - 신뢰성 : 데이터를 순차적, 안정적으로 전달
    - 전송 : 포트 번호에 해당하는 프로세스에 데이터를 전달

TCP

- **인터넷상에서 데이터를 메세지의 형태로 보내기 위해 IP와 함께 사용하는 프로토콜**
- TCP 특징
    - 연결 지향 방식으로 패킷 교환 방식을 사용한다(가상 회선 방식이 아님)
        - 패킷을 전송하기 위한 논리적 경로를 배정한다
    - 3-way handshaking과정을 통해 연결을 설정하고 4-way handshaking을 통해 해제한다.
    - 흐름 제어 및 혼잡 제어.
    - 높은 신뢰성을 보장한다.
    - UDP보다 속도가 느리다.

    - 전이중(Full-Duplex), 점대점(Point to Point) 방식
- TCP 서버의 특징
    - T서버소켓은 연결만을 담당한다.
    - 연결과정에서 반환된 클라이언트 소켓은 데이터의 송수신에 사용된다
    - 서버와 클라이언트는 1대1로 연결된다.
    - 스트림 전송으로 전송 데이터의 크기가 무제한이다.
    - 패킷에 대한 응답을 해야하기 때문에(시간 지연, CPU 소모) 성능이 낮다.
    - Streaming 서비스에 불리하다.(손실된 경우 재전송 요청을 하므로)

UDP

- 데이터를 데이터그램 단위로 처리하는 프로토콜
    - 데이터그램이란 독립적인 관계를 지니는 패킷
- UDP 특징
    - 비연결형 서비스로 데이터그램 방식을 제공한다
    - 정보를 주고 받을 때 정보를 보내거나 받는다는 신호절차를 거치지 않는다.
    - UDP헤더의 CheckSum 필드를 통해 최소한의 오류만 검출한다.
    - 신뢰성이 낮다
    - TCP보다 속도가 빠르다
- UDP 서버의 특징
    - UDP에는 연결 자체가 없어서(connect 함수 불필요) 서버 소켓과 클라이언트 소켓의 구분이 없다.
    - 소켓 대신 IP를 기반으로 데이터를 전송한다.
    - 서버와 클라이언트는 1대1, 1대N, N대M 등으로 연결될 수 있다.
    - 데이터그램(메세지) 단위로 전송되며 그 크기는 65535바이트로, 크기가 초과하면 잘라서 보낸다.
    - 흐름제어(flow control)가 없어서 패킷이 제대로 전송되었는지, 오류가 없는지 확인할 수 없다.
    - 파일 전송과 같은 신뢰성이 필요한 서비스보다 성능이 중요시 되는 경우에 사용된다.

프로토콜 종류	TCP	UDP
연결 방식	연결형 서비스 (패킷 교환 방식)	비연결형 서비스 (데이터그램 방식)
전송 순서	전송 순서 보장	전송 순서가 바뀔 수 있음
수신 여부 확인	수신 여부를 확인함	수신 여부를 확인하지 않음
통신 방식	1:1 통신	1:1 OR 1:N or N:N 통신
신뢰성	높다	낮다
속도	느리다	빠르다

3-way handshake

▼ TCP 헤더

TCP segment header

Offsets	Octet	0								1								2								3							
Octet	Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
0	0	Source port																Destination port															
4	32	Sequence number																															
8	64	Acknowledgment number (if ACK set)																															
12	96	Data offset				Reserved 0 0 0				NS	CWR	ECE	URG	ACK	PSH	RST	SYN	FIN	Window Size														
16	128	Checksum																	Urgent pointer (if URG set)													
20	160	Options (if *data offset* > 5. Padded at the end with "0" bits if necessary.)																															
⋮	⋮																																
60	480																																

▼ 상세설명

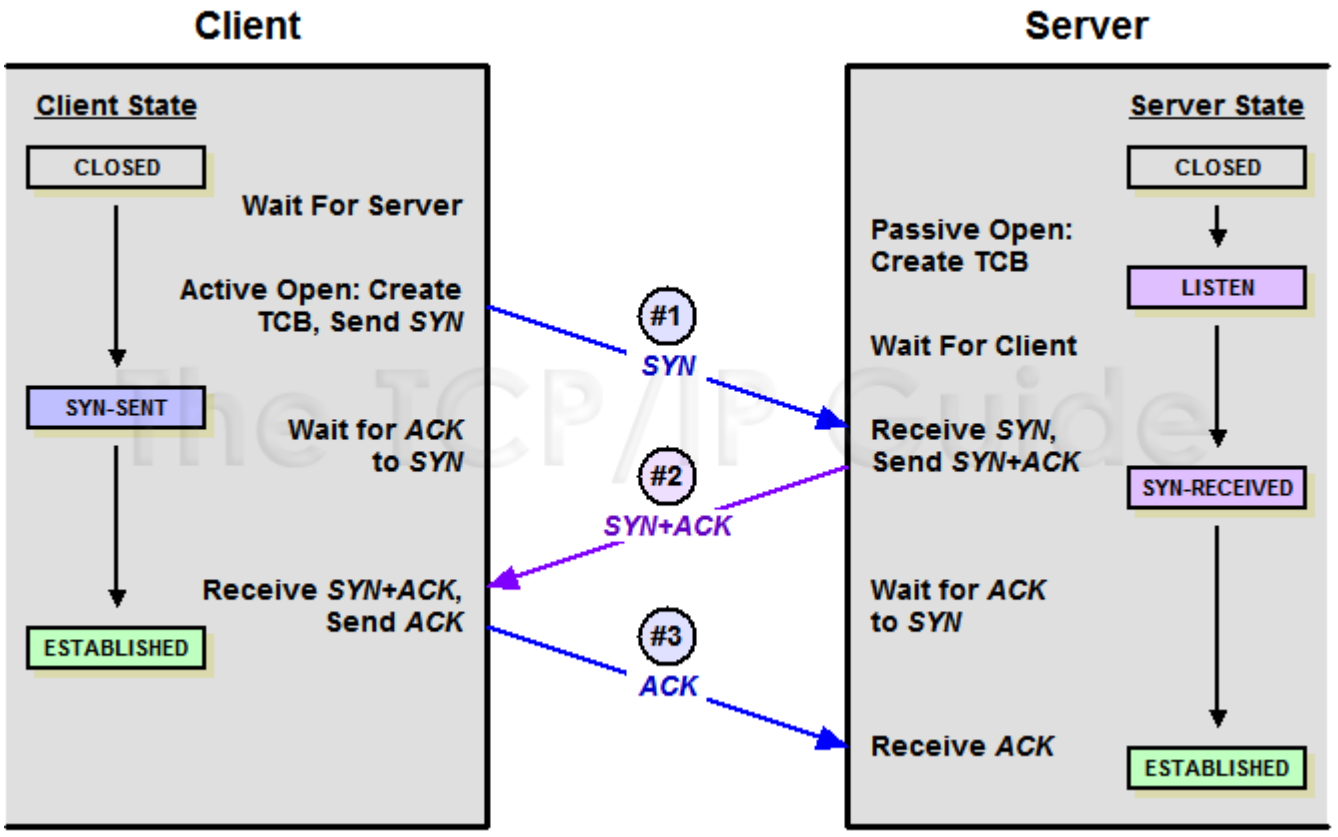
1. **Source/Destination Port** (2 bytes): Source와 Destination에 해당하는 16-bit port number
2. **Sequence Number** (4 bytes): 송신하는 데이터의 순서. 이를 바탕으로 수신 측에서 데이터를 순서대로 재조립 가능
3. **Acknowledgement Number** (4 bytes): 수신 측에서 다음으로 받을 거라 생각하는 Sequence Number
   - Handshaking할 때: 상대방이 보낸 Sequence Number + 1
   - 데이터 주고 받을 때: 상대방이 보낸 Sequence Number + 자신이 받은 데이터의 bytes
4. **Data Offset** (4 bits): 전체 세그먼트 중 헤더를 제외한 Data가 시작하는 위치. 아래 Option 필드가 가변적이기 때문에 필요
5. **Reserved** (3 bits): 추후에 사용하기 위해 비워둔 비트
6. **Flags** (9 bits)
   - URG: Urgent Pointer 필드에 값이 있음을 표시. 해당 필드가 가리키는 데이터가 우선적으로 처리되도록 함
   - ACK: Acknowledgement Number 필드에 값이 있음을 표시
   - PSH: Push 플래그. 수신 측에 이 데이터를 최대한 빠르게 응용 프로그램에 전달해달라는 플래그. 이 플래그가 0이면, 수신 측은 자신의 버퍼가 다 채워질 때까지 기다린 후 데이터를 전송함
   - RST: Reset 플래그. 연결을 이미 생성한 상대방에게 연결을 강제로 리셋해달라는 요청을 할 때 사용함
   - SYN: Synchronize 플래그. 상대방과 연결을 생성할 때, Sequence Number의 동기화를 맞추기 위한 세그먼트임을 표시함
   - FIN: Finish 플래그. 상대방과의 연결을 종료하고 싶다는 요청인 세그먼트임을 표시함

   [혼잡 상태를 알려주기 위한 플래그]

   ▼ [참고] ECN (Explicit Congestion Notification)
   - 지연 감지 시, 라우터 측에서 수신자에게 IP 헤더의 CE(Congestion Experienced) 플래그를 세팅한 후 알림. 수신자는 ECE(ECN-Echo) 플래그를 설정해 송신자에게 전송하면, 송신자는 혼잡 제어를 위한 처리를 한 후, CWR 플래그 송신
   - NS: ECN의 은폐되는 경우 방지하기 위해 사용하는 플래그
   - CWR: Congestion Window Reduced 플래그. ECE 플래그를 가진 TCP 세그먼트를 받아 혼잡 제어 처리를 했다는 표시
   - ECE: ECN-Echo 플래그
     - SYN Flag가 1일 때: ECN을 사용할 수 있다는 것을 알림
     - SYN Flag가 0일 때: 혼잡 상태가 발생했다는 것을 알림
7. **Window Size** (2 bytes): 세그먼트를 보내는 측에서 받을 수 있는 윈도우 크기
8. **Checksum** (2 bytes): TCP 헤더의 오류를 검출하기 위한 값
9. **Urgent Pointer** (2 bytes): URG 플래그가 1일 때, 수신 측은 이 포인터가 가리키고 있는 데이터 우선 처리
10. Options: 추가로 사용할 수 있는 옵션
11. Padding: TCP 헤더가 4 byte 단위로 나눠떨어지도록 하는 역할. 0으로만 이루어져 있음

- ACK: Acknowledgement Number 필드에 값이 있음을 표시
- SYN: Synchronize 플래그. 상대방과 연결을 생성할 때, Sequence Number의 동기화를 맞추기 위한 세그먼트임을 표시함
- FIN: Finish 플래그. 상대방과의 연결을 종료하고 싶다는 요청인 세그먼트임을 표시함

- 3-way handshake란 TCP 네트워크에서 통신을 하는 장치가 서로 연결이 잘 되었는지 확인하는 방법이다. 송신자와 수신자는 총 3번에 걸쳐 데이터를 주고 받으며 통신이 가능한 상태임을 확인한다.
- 편의상 클라이언트와 서버라고 하지만, 그보다는 연결 생성을 요청한 쪽와 연결 생성 요청을 받은 쪽이라고 생각하면 된다. 둘 다 연결 생성 요청을 보낼 수 있다.
- 클라이언트가 서버에 연결을 시도하기 전, 서버는 먼저 연결을 위해서 포트를 열어놓아야 한다 (passive open). 그 후, 클라이언트는 3WH를 통해 서버와의 연결을 설정한다. 이 과정이 끝나면, Connection ESTABLISHED 상태가 된다.



Client (**연결 생성 요청한 쪽**)	Server (**연결 생성 요청 받은 쪽**)
[CLOSED] 클라이언트는 서버가 연결을 위해 포트를 열어주지 않는 이상 아무것도 할 수 없음	**[CLOSED]** 아직 포트를 열지 않은 상태. 서버는 passive open을 수행함. 이 때, TCB (Transmission Control BLock, TCP 연결 상태에 대한 정보 갖고 있는 자료구조)를 생성하는 등, 클라이언트의 연결 요청(SYN)에 대한 준비를 함 ⇒ **[LISTEN]**
1️⃣ 서버에 SYN 패킷 전송 ⇒ **[SYN-SENT]**	**[LISTEN]** 클라이언트의 연결 요청 대기
[SYN-SENT] 자신이 보낸 SYN 패킷에 대한 서버의 SYN+ACK 패킷 대기	2️⃣ 클라이언트가 보낸 SYN 패킷을 수신한 후, 클라이언트의 SYN에 대한 ACK과, 자신의 SYN 전송 ⇒ **[SYN-RECEIVED]**
3️⃣ 서버가 보낸 SYN+ACK 패킷을 수신한 후, 서버의 SYN에 대한 ACK을 전송 ⇒ **[ESTABLISHED]**	자신이 보낸 SYN 패킷에 대한 클라이언트의 ACK 패킷 대기
서버의 연결 설정 완료 대기	클라이언트가 보낸 ACK 수신 ⇒ **[ESTABLISHED]**
[ESTABLISHED] 연결 생성 완료	**[ESTABLISHED]** 연결 생성 완료