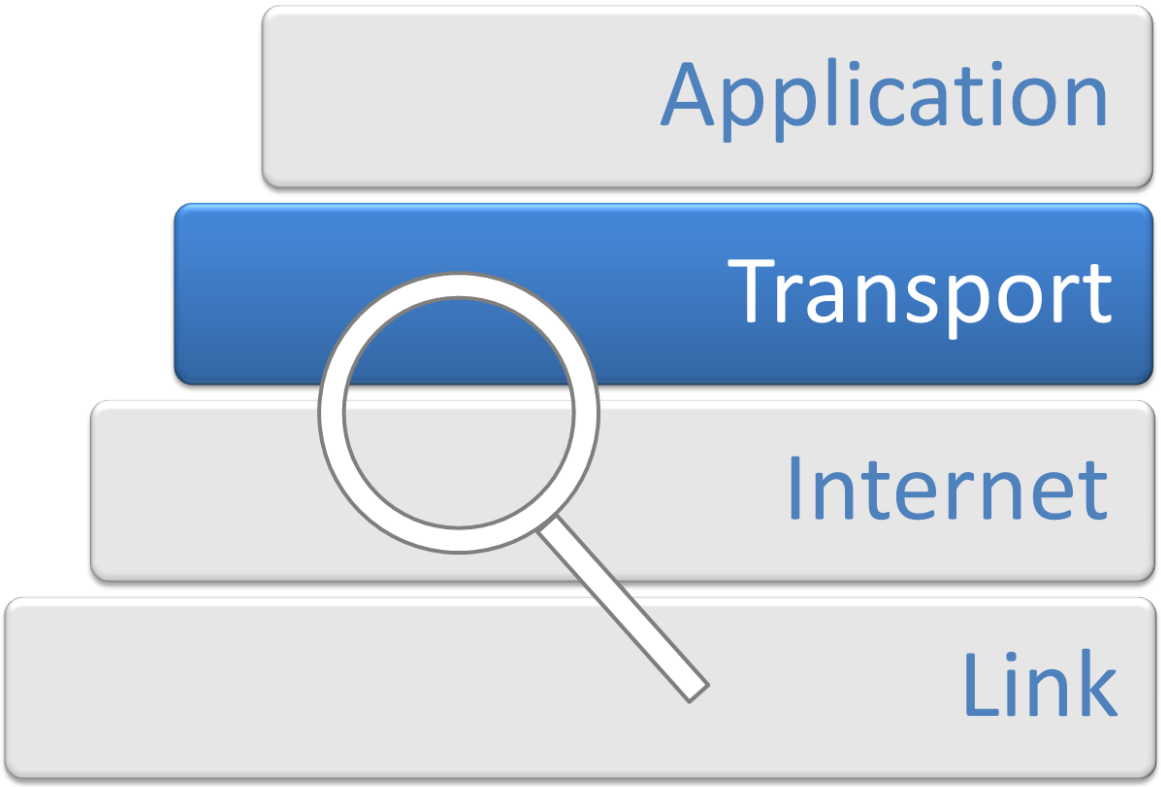
TCP 3 & 4 way handshake

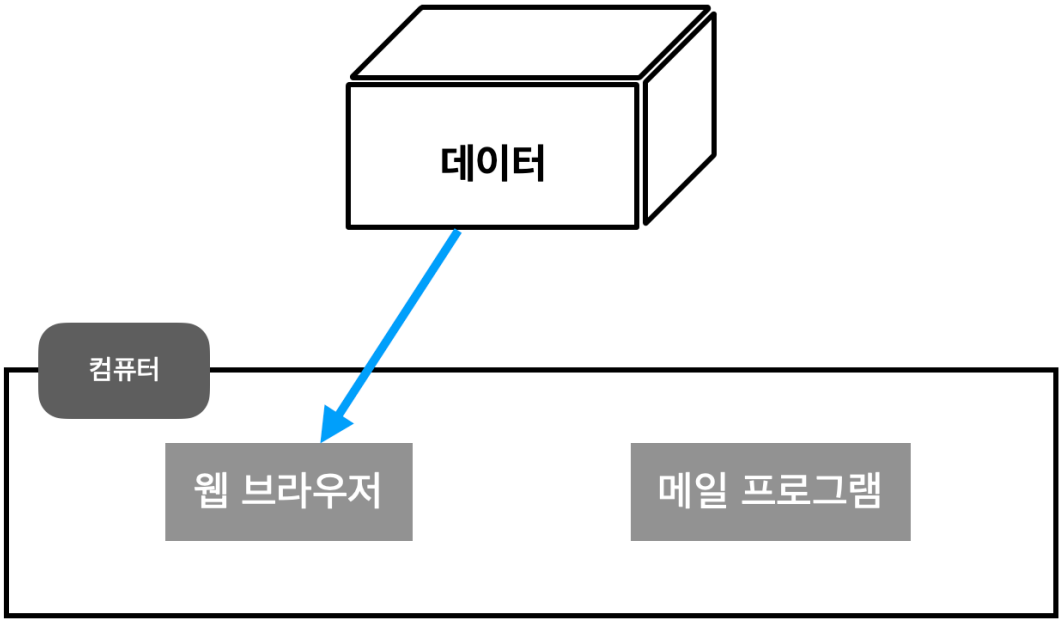
전송 계층

- 물리 계층, 데이터 링크 계층, 네트워크 계층의 3계층만 있더라도 **목적지에 데이터를 보낼 수는 있다.**
    - 그러나, **수신 컴퓨터가 존재하더라도 패킷을 수신할 준비가 되었는지, 전송 과정에서 패킷이 손상되거나 유실되었는지와 같은 문제들은 전혀 신경쓰지 않는다.**
    - 즉, IP 프로토콜은 통신 호스트 간 패킷 전달을 위해 노력하지만 **패킷 전송 순서나 완전성을 보장하지 않아 비신뢰형 서비스**라고도 부른다.

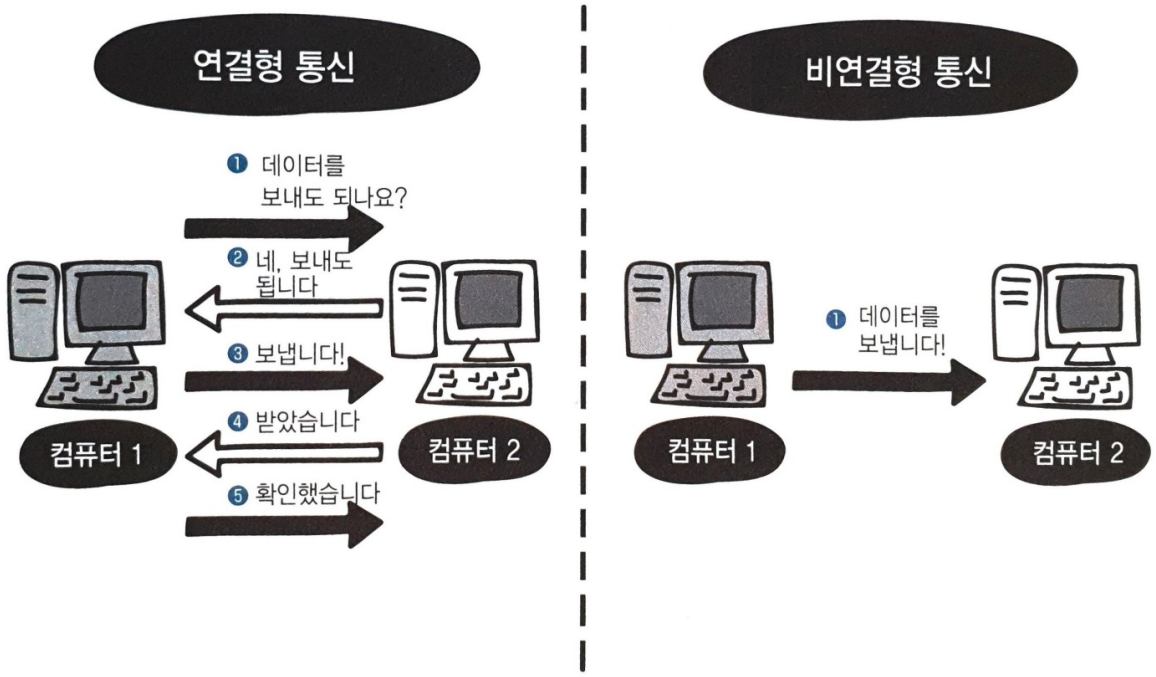


- **패킷 전송 과정에서 문제없이 수신 컴퓨터에 도착할 수 있도록 패킷 전송을 제어하는 역할은 전송 계층이 담당한다.**

- 네트워크 혼잡 상황에 따라 패킷 전송량을 조절하여 패킷의 흐름을 제어한다.
- 패킷 전송 중 오류가 없었는지 점검하고, 수신 컴퓨터까지 패킷이 제대로 도착했는지 확인한다.
- **즉, 수신 컴퓨터까지 신뢰할 수 있는 데이터를 전송하기 위해 필요한 계층이다.**



- 또한, **전송 계층**은 **전송된 데이터의 목적지가 어떤 애플리케이션인지 식별하는 기능**도 존재한다.
  - 다양한 애플리케이션이 동작하는 컴퓨터 내에서 **어떤 애플리케이션이 사용하는 데이터인지 식별**해 수신 컴퓨터에 도착한 데이터를 **수신 컴퓨터 내의 애플리케이션에 배분하는 역할**을 한다.



- 전송 계층은 간단히 **신뢰성/정확성**과 **효율성**으로 특징을 구분할 수 있다.

- **신뢰성/정확성은 데이터를 목적지에 문제없이 전달하게 하는 것**이다.
  - **신뢰할 수 있고 정확한 데이터를 전달하는 통신을 연결형 통신**이라고 한다.
  - **연결형 통신 프로토콜**로 **TCP**가 사용된다.

- **효율성은 데이터를 효율적으로, 빠르게 전달하는 것**이다.
  - **효율적이고 빠르게 데이터를 전달하는 통신**을 **비연결형 통신**이라고 한다.
  - 동영상과 같은 데이터를 전송할 때 주로 사용한다.
  - **비연결형 통신 프로토콜**로 **UDP**가 사용된다.

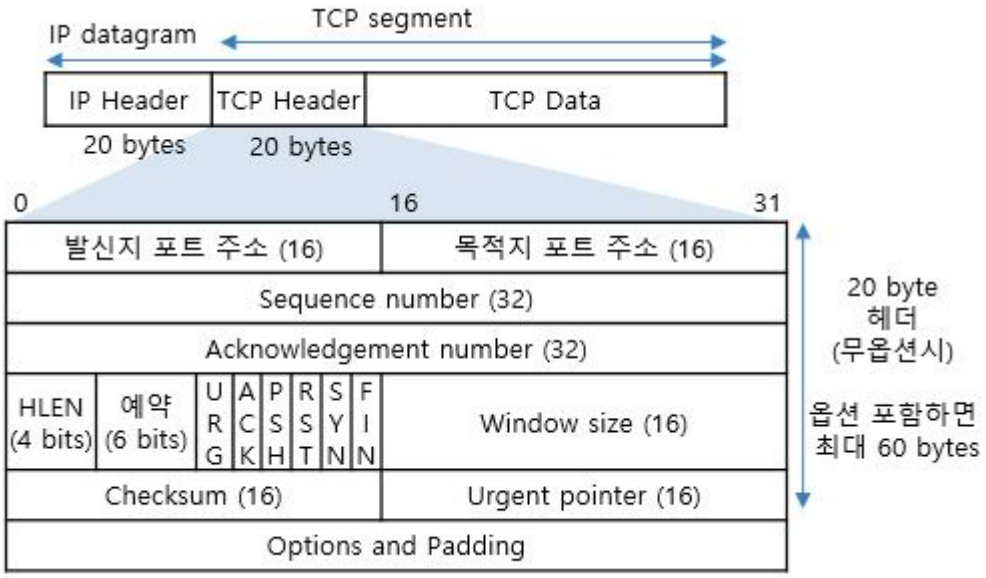
TCP(Transmissioon Control Protocol)

- **신뢰성과 정확성을 우선으로 하는 연결형 통신 프로토콜**을 의미한다.

특징

- **연결형 서비스**로 **연결이 성공해야 통신이 가능**하다.
- 데이터 경계를 구분하지 않는다.
- **데이터 전송 순서를 보장**한다.
  - 데이터 순서 유지를 위해 각 바이트마다 번호를 부여한다.
- **신뢰성 있는 데이터**를 전송한다.
- **점대점, 전이중 통신**을 사용한다.
- `3-Way Handshaking` 을 통해 연결하고, `4-Way Handshaking` 을 통해 연결을 종료한다.

TCP 헤더

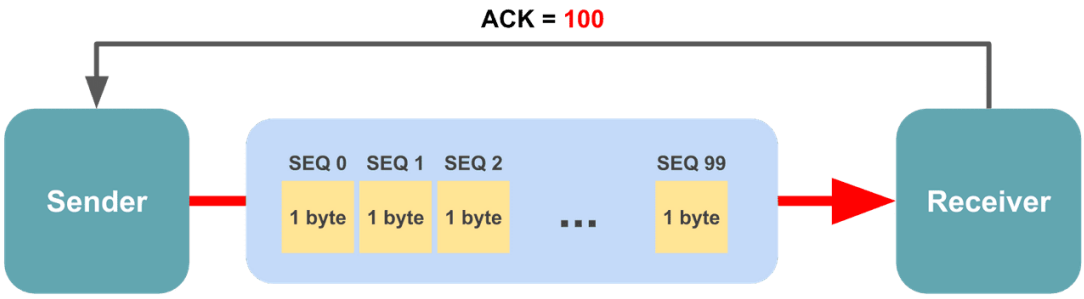


- 응용 계층의 데이터 단위인 메시지를 받아 **TCP 프로토콜에 따라 분할**하고, **분할된 데이터에 TCP 헤더가 붙는다.**
  - **TCP 헤더가 붙은 데이터를 세그먼트**라고 한다.

- **일련 번호(Sequence number)**

- 전송하는 데이터의 순서를 의미한다.
- 수신자는 일련 번호를 통해 쪼개진 세그먼트의 순서를 파악해 올바른 순서로 데이터를 재조립할 수 있다.

- **확인 응답 번호(Acknowledgement number)**
  - 데이터를 받은 수신자가 예상하는 다음 일련 번호를 의미한다.
  - 핸드쉐이크 과정에서는 상대방이 보낸 `일련 번호 + 1` 로 번호를 만들어낸다.



  - 실제 데이터를 주고받을 때는 상대방이 보낸 `일련 번호 + 수신 받은 데이터의 바이트` 로 번호를 만들어낸다.

- **코드 비트(Flags)**
  - 6개의 비트 플래그로, **현재 세그먼트의 속성을 나타낸다.**
  - **URG**
    - 긴급 포인터에 값이 채워져있음을 알리는 플래그
    - 이 포인터가 가리키는 긴급 데이터는 높게 처리되어 먼저 처리된다.
  - **ACK**
    - 승인 번호 필드에 값이 채워져있음을 알리는 플래그
    - `0` 이라면 필드 자체가 무시된다.
  - **PSH**
    - 수신 측에게 이 데이터를 최대한 빠르게 애플리케이션으로 전달해달라는 플래그
    - `0` 이라면 수신 측은 자신의 버퍼가 다 채워질 때까지 대기한다.
    - `1` 이라면 이 세그먼트 이후 더 연결된 세그먼트가 없음을 의미한다.

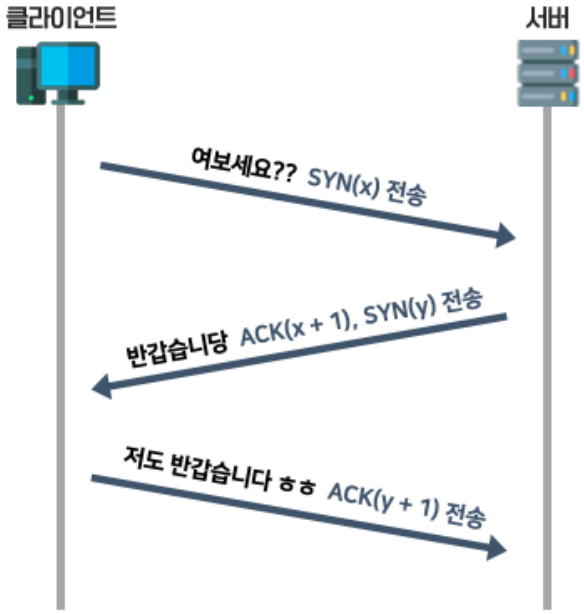
- **RST**
  - 상대방과 연결이 되어있는 상태에서 어떤 문제 등이 발생하여 연결 상태를 리셋해달라는 요청을 보내는 플래그
- **SYN**
  - 상대방과 연결을 생성할 때 일련 번호의 동기화를 맞추기 위한 플래그
- **FIN**
  - 상대방과 연결을 종료하고 싶다는 요청을 보내는 플래그

TCP의 연결 생성 및 종료

- TCP 프로토콜에서 데이터를 전송하려면 먼저 `연결`이라는 **가상의 독점 통신로를 확보**해야 한다.
  - TCP는 **신뢰성 있는 연결**을 추구하므로 `연결`**을 생성하고 종료하는 순간에도 신뢰성 확보를 위해 핸드쉐이크 과정**을 거친다.

연결을 위한 3-Way Handshake

- **연결을 생성할 때 거치는 핸드쉐이크 과정**을 `3-Way Handshake` 라고 한다.
  - `SYN`, `ACK` **비트**를 사용해 통신한다.

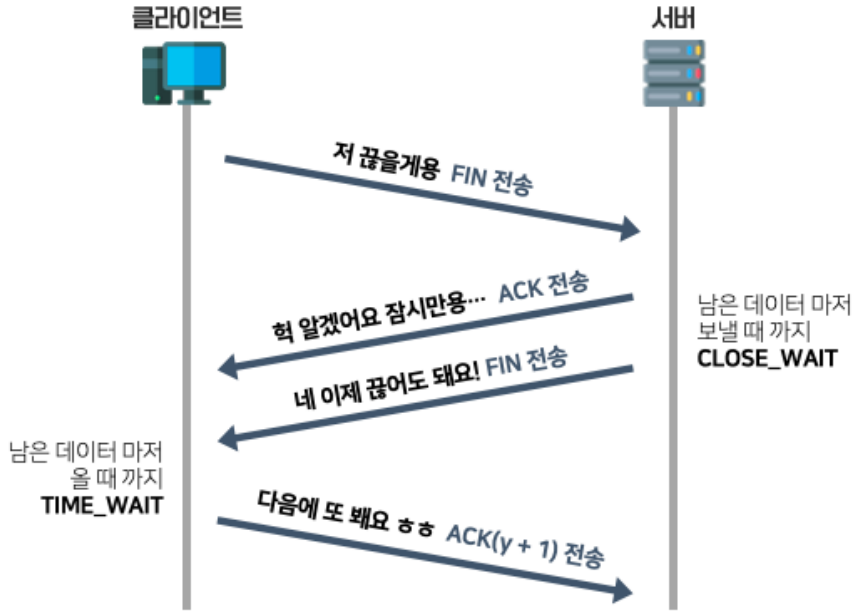


1. 전송자가 수신자에게 **연결 확립 허가를 받기 위한 요청(`SYN`)을 전송**한다.
2. 수신자는 요청을 받은 후 **허가한다는 응답을 회신하기 위해 연결 확립 응답(`ACK`)를 전송**한다. 이와 동시에, 수신자도 전송자에게 **데이터 전송 허가를 받기 위해 연결 확립 요청(`SYN`)을 전송**한다.
3. 수신자의 요청을 받은 전송자는 **허가한다는 응답으로 연결 확립 응답(`ACK`)를 전송**한다.

- **3번의 통신을 마치면, 종단 간 연결이 성립된다.**

연결 종료를 위한 4-Way Handshake

- **연결을 종료하기 위해 거치는 핸드쉐이크 과정**을 `4-Way Handshake` 라고 한다.
  - `FIN`, `ACK` **비트**를 사용해 통신한다.



1. 전송자가 수신자에게 **연결 종료 요청(`FIN`)을 전송**한다.
2. 수신자가 전송자에게 **연결 종료 응답(`ACK`)를 전송**한다.
    - 이 때, 남은 데이터를 모두 보내기 위해 `CLOSE_WAIT` 상태로 전환된다.
3. 남은 데이터를 모두 보냈다면 수신자가 전송자에게 **연결 종료 요청(`FIN`)을 전송**한다.
4. 전송자는 연결 종료 요청 수신 후 확인했다는 의미로 **연결 종료 응답(`ACK`)를 전송**한다.
    - 데이터를 모두 받지 못했을 경우 대기하기 위해 `TIME_WAIT` 상태로 전환된다.
5. 위 과정을 통해 수신자가 전송자로부터 `ACK`를 수신했다면 연결을 완전히 종료한다.
6. `TIME_WAIT` 시간이 끝나면 전송자 역시 연결을 완전히 종료한다.

- 위 두 개의 핸드쉐이크 과정을 통해 **TCP가 단순하게 연결을 생성하고 종료하는 과정**임에도 **신뢰성을 확보**하기 위해 얼마나 많은 작업을 하는지 알 수 있다.

References

- https://evan-moon.github.io/2019/11/10/header-of-tcp/
- https://velog.io/@haero_kim/TCP