질문	답변
tcp/udp 필요한 이유	**네트워크를 계층화해 놓은 이유**를 생각해보면 된다. 하나의 계층에서 오류나 처리를 하는 것보다 **각 계층을 의미와 기능적으로 분류**하고 각각의 기능을 수행함으로써 **오류 대응에 용이**하다. 장치로의 이동은 ip인터넷 프로토콜로 해결이 가능하지만, ip에서 오류가 발생하면 icmp에서 알려주고 대응은 tcp/udp에서 하는 것임.
TCP (Transmission Control Protocol)	TCP 는 **연결형 서비스로 가상 회선 방식을 제공**합니다. 데이터의 **손실을 최소화**하고 데이터의 **순서를 보장**하는 등 데이터를 **신뢰성** 있게 전송하는 역할을 합니다. 따라서, 데이터의 **완전한 전달과 순서가 중요한 상황**에서 사용됩니다. 단점은 전화 통화처럼 시작하고 끝나는 과정이 추가로 필요하기 때문에 **약간 느릴** 수 있습니다. TCP는 **멀티캐스팅**이나 **브로드캐스팅**과 같은 한 번에 여러 대의 컴퓨터에 데이터를 보내는 기능은 지원하지 않습니다. 또한, 통신 중에 데이터가 누락되거나 유실되는 상황에 대비하여 데이터의 흐름을 조절하고 혼잡을 제어하는 기능도 갖추고 있습니다.
가상 회선(Virtual Circuit)	**컴퓨터 네트워크에서 사용되는 통신 방식** 중 하나입니다. 가상 회선은 연결형 서비스를 제공하며, 데이터 패킷이 전송되기 전에 논리적인 경로를 설정하는 특징을 가지고 있습니다.
흐름 제어와 혼잡 제어	1. 흐름 제어 (Flow Control): 흐름 제어는 수신자와 송신자 사이의 데이터 **전송 속도**를 조절하는 메커니즘입니다. 슬라이딩 윈도우 (Sliding Window): 흐름 제어에서 사용되는 주요 개념 중 하나로, 수신자는 송신자에게 받을 수 있는 데이터의 양을 알려줍니다. 송신자는 이 정보를 바탕으로 데이터를 보냅니다. 이유는 수신자가 데이터를 처리하지 못하면 데이터가 손실되거나 낭비될 수 있기 때문입니다. 수신자의 알림 (Acknowledgment): 수신자는 성공적으로 받은 데이터를 확인하는 ACK 패킷을 송신자에게 보냅니다. 송신자는 ACK를 받으면 데이터를 계속 전송합니다. 흐름 제어는 네트워크의 병목 현상을 방지하고 송신자와 수신자 간의 데이터 흐름을 최적화하는 데 도움을 줍니다. 2. 혼잡 제어 (Congestion Control): 혼잡 제어는 네트워크 혼잡을 방지하고 **효율적인 데이터 전송**을 보장하는 메커니즘입니다. 혼잡 윈도우 (Congestion Window): 혼잡 제어에서 사용되는 주요 개념 중 하나로, 송신자는 현재 네트워크 혼잡 상태를 **감지**하고 데이터를 전송하는 **속도를 조절**합니다. 혼잡 윈도우 크기를 동적으로 조절하여 혼잡을 방지하고 효율적인 전송을 유지합니다. 패킷 손실 감지: 혼잡 제어는 네트워크 혼잡을 감지하기 위해 패킷 손실을 모니터링합니다. 패킷 손실이 발생하면 송신자는 전송 속도를 감소시키고 혼잡을 완화합니다. 혼잡 제어는 네트워크의 혼잡을 방지하여 데이터의 손실과 대기 시간을 줄이는 역할을 합니다. 이를 통해 네트워크 품질을 향상시키고 효율적인 데이터 전송을 가능하게 합니다. 이 두 가지 메커니즘은 TCP의 핵심 기능 중 하나로, 안정적이고 효율적인 데이터 통신을 보장하는 데 기여합니다.
UDP (User Datagram Protocol)	UDP는 편리하고 **빠른** 방법으로 데이터를 보낼 수 있지만 데이터의 정확성과 신뢰성을 **보장하지 않습니다**. 따라서, **연속적인** 데이터 전송이 중요하고 조금의 데이터 **손실을 감수**할 수 있는 경우에 주로 사용됩니다. UDP는 실시간 영상 스트리밍, 음성 통화 등과 같이 연속적인 데이터 전송이 중요한 상황에서 유용합니다.
TCP와 UDP의 주요 차이점은 무엇인가요?	TCP (Transmission Control Protocol)는 연결 지향적이며 신뢰성이 있는 프로토콜로 데이터 전송을 보장합니다. 반면, UDP (User Datagram Protocol)는 연결이 없는 비신뢰성 프로토콜로 데이터를 전송하지만 보장하지 않습니다.

3-Handshaking의 과정을 설명해주세요	클라이언트가 서버에게 연결 요청 (SYN): 클라이언트는 서버에게 TCP 연결을 설정하려는 의도를 가지고 SYN (Synchronize) 패킷을 보냅니다. 클라이언트의 초기 시퀀스 번호 (ISN)가 이 패킷에 포함되어 있으며, 이 값은 클라이언트에서 생성되는 데이터 스트림의 시작을 나타냅니다. 서버가 응답 (SYN-ACK): 서버는 클라이언트의 SYN을 받고, 클라이언트에게 연결을 수락한다는 신호로 SYN-ACK 패킷을 반환합니다. 서버는 또한 자체적으로 초기 시퀀스 번호 (ISN)를 생성하여 이를 패킷에 포함시킵니다. 클라이언트가 응답에 대한 확인 (ACK): 클라이언트는 서버로부터의 SYN-ACK를 받으면 이에 대한 확인을 보내는 ACK (Acknowledgment) 패킷을 생성하고 서버로 보냅니다. 이 패킷은 서버에게 클라이언트의 연결 요청이 성공적으로 처리되었음을 알려줍니다. 이제 양쪽 모두가 연결 설정을 완료하고, 데이터 전송이 시작될 수 있습니다.
4-way 핸드셰이크 (TCP 연결 해제)	클라이언트가 연결 종료 요청 (FIN): 클라이언트가 데이터 전송을 모두 마치고 연결을 종료하려고 할 때, 클라이언트는 서버에게 FIN (Finish) 패킷을 보냅니다. 이것은 클라이언트가 더 이상 데이터를 보내지 않을 것임을 나타냅니다. 서버가 확인 및 데이터 전송 (ACK): 서버는 FIN을 받으면, 이것에 대한 확인으로 ACK 패킷을 반환합니다. 그러나 아직 서버가 보내지 않은 데이터가 있을 수 있으므로, 이 패킷에는 FIN을 포함하지 않습니다. 서버는 데이터 전송을 완료한 후에 FIN을 보내기 위해 대기합니다. 서버가 연결 종료 요청 (FIN): 서버가 데이터 전송을 완료하고 연결을 종료하려고 할 때, 서버는 클라이언트에게 FIN 패킷을 보냅니다. 이것은 서버가 더 이상 데이터를 보내지 않을 것임을 나타냅니다. 클라이언트가 확인 (ACK): 클라이언트는 서버로부터의 FIN을 받으면 이에 대한 확인으로 ACK 패킷을 반환합니다. 이로써 클라이언트는 연결 해제 요청에 동의하고 연결이 종료됩니다.
TCP의 혼잡 제어(congestion control)란 무엇이며 왜 중요한가요?	혼잡 제어는 네트워크 혼잡을 방지하고 효율적인 데이터 전송을 위한 메커니즘입니다. 혼잡 제어가 없으면 네트워크 혼잡으로 인해 패킷 유실과 대기시간이 증가할 수 있습니다.
어떻게 혼잡 제어가 동작하나요?	혼잡 제어는 주로 TCP (Transmission Control Protocol)에서 사용됩니다. TCP는 혼잡 제어 알고리즘을 통해 데이터 전송 속도를 동적으로 조절합니다. 혼잡 제어 알고리즘은 네트워크 혼잡을 감지하고 패킷 손실을 모니터링합니다. 패킷 손실이 감지되면 송신측은 데이터 전송 속도를 감소시키고 네트워크 혼잡을 완화하기 위한 조치를 취합니다.
DNS(Domain Name System)의 기능은 무엇인가요?	DNS 도메인 이름을 IP 주소로 변환하고, IP 주소를 도메인 이름으로 역으로 변환하는 역할을 합니다. 도메인 이름 해석 (Name Resolution): DNS는 사용자가 읽기 쉬운 도메인 이름(예: www.example.com)을 컴퓨터가 이해할 수 있는 숫자 형태인 IP 주소(예: 192.0.2.1)로 변환합니다. 도메인 이름의 계층 구조 관리: DNS는 도메인 이름을 계층 구조로 관리합니다. 최상위 도메인(Top-Level Domain, TLD)에서부터 하위 도메인으로 이어지는 계층 구조를 통해 도메인 이름을 관리하고 식별합니다. 도메인 이름 등록 및 관리: DNS는 도메인 이름의 등록 및 관리를 위한 인프라를 제공합니다. 도메인 이름 등록기관은 고유한 도메인 이름을 부여하고, 이를 DNS에 등록하여 사용자가 해당 도메인 이름을 조회할 수 있도록 합니다.
DNS의 동작 방식	도메인 이름 쿼리: 클라이언트(예: 웹 브라우저)가 도메인 이름을 사용하여 웹 사이트에 액세스하려고 할 때, 클라이언트는 DNS 서버에 도메인 이름 쿼리를 보냅니다. DNS 서버 조회: DNS 서버는 도메인 이름을 검색하고, 로컬 캐시에서 해당 도메인 이름과 관련된 IP 주소를 찾습니다. 로컬 캐시에 없는 경우, 상위 DNS 서버에 쿼리를 보내 IP 주소를 찾습니다. 계층적인 조회: DNS 서버는 도메인 이름을 계층적으로 조회하며, 최상위 도메인(TLD) 서버부터 시작하여 하위 도메인 서버로 이동합니다. 이렇게 계층 구조를 따라 도메인 이름을 해석합니다. IP 주소 반환: DNS 서버는 도메인 이름에 대한 IP 주소를 찾으면 이를 클라이언트에게 반환합니다. 로컬 캐싱: DNS 서버는 조회한 도메인 이름과 해당 IP 주소를 일정 기간 동안 로컬 캐시에 저장하여 다음에 동일한 도메인 이름에 대한 조회 시에 빠르게 응답할 수 있도록 합니다.