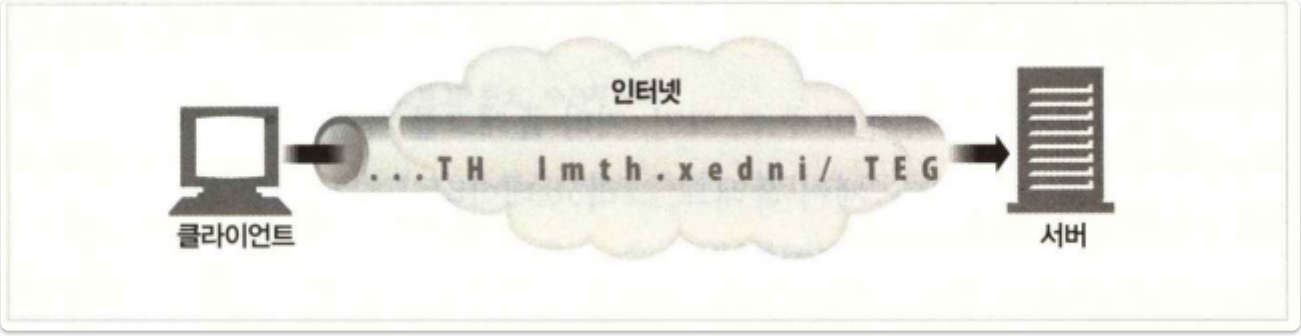
4장. 커넥션 관리

[# 웹 프로그래머를 위한 HTTP 완벽 가이드 읽는 법]

1. 소개

- TCP (Transmission Control Protocol)는 인터넷 프로토콜 스위트의 중요한 프로토콜 중 하나입니다.
- 네트워크 통신을 위해 사용되며, 신뢰성 있는 데이터 전송을 보장합니다.
- TCP는 연결 지향형 프로토콜로, 두 시스템 간에 가상의 통신 선로를 설정하고 유지합니다.



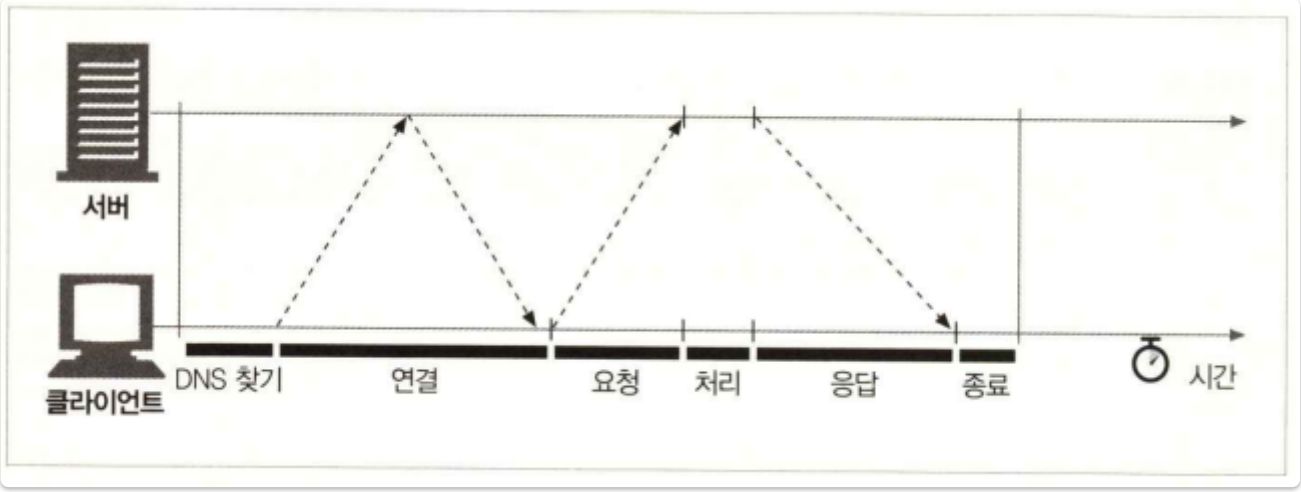
2. TCP 커넥션의 상태

- **CLOSED:** 초기 상태, 커넥션이 설정되지 않은 상태입니다.
- **LISTEN:** 서버가 클라이언트의 연결 요청을 기다리는 상태입니다.
- **SYN_SENT:** 클라이언트가 서버에게 커넥션을 요청하는 중입니다.
- **SYN_RECEIVED:** 서버가 클라이언트의 요청을 받았고, 응답을 준비하고 있는 상태입니다.
- **ESTABLISHED:** 커넥션이 성공적으로 설정되어 데이터가 주고받아지는 상태입니다.
- **FIN_WAIT_1, FIN_WAIT_2:** 커넥션 종료 과정 중에 클라이언트가 있는 상태입니다.
- **CLOSE_WAIT:** 커넥션 종료 과정 중에 서버가 있는 상태입니다.
- **LAST_ACK:** 서버가 클라이언트의 종료 응답을 기다리는 상태입니다.
- **TIME_WAIT:** 커넥션 종료 후 일정 시간 동안 대기하는 상태로, 잔류 패킷을 처리하기 위한 것입니다.

3. TCP 성능 문제

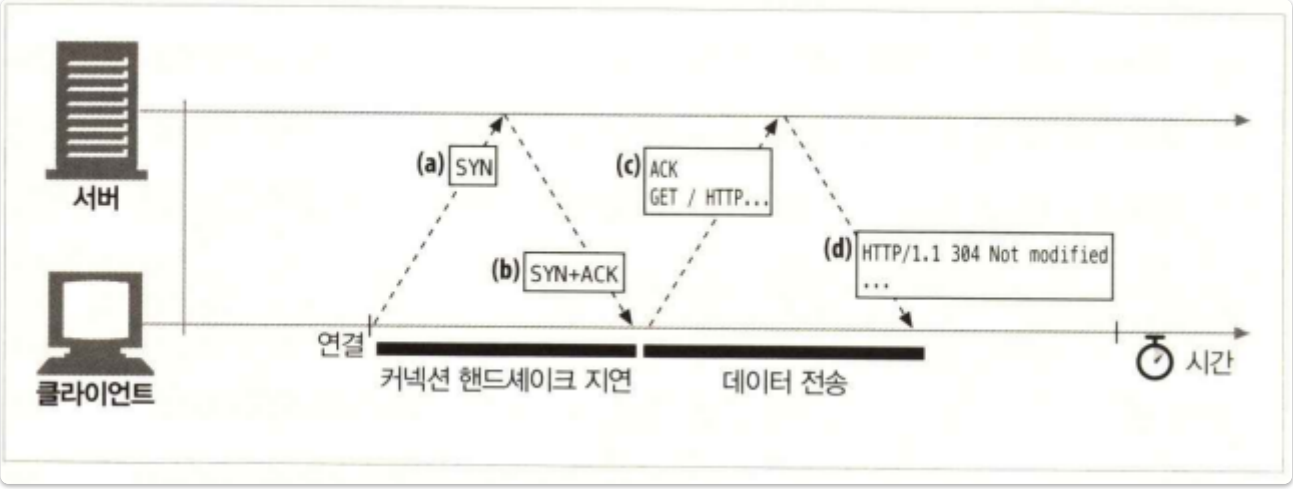
3.1 트랜잭션 지연 (Transaction Delay)

- TCP는 데이터 전송을 보장하기 위해 패킷의 송수신을 확인하고 응답해야 한다.
- 이 과정에서 데이터 전송이 지연될 수 있으며, 이를 트랜잭션 지연이라고 한다.
- 작은 데이터를 전송하는 경우에도 확인 및 응답 과정이 추가되어 지연이 발생할 수 있다.



3.2 핸드셰이크 지연 (Handshake Delay)

- TCP 연결을 설정할 때 세션을 확립하기 위해 "순방향 핸드셰이크"와 "역방향 핸드셰이크"가 필요하다.
- 이 핸드셰이크 단계에서도 지연이 발생할 수 있는데, 특히 원격지 서버의 응답이 늦어지면 연결 설정 자체가 늦어질 수 있다.



3.3 확인응답 지연 (Acknowledgment Delay)

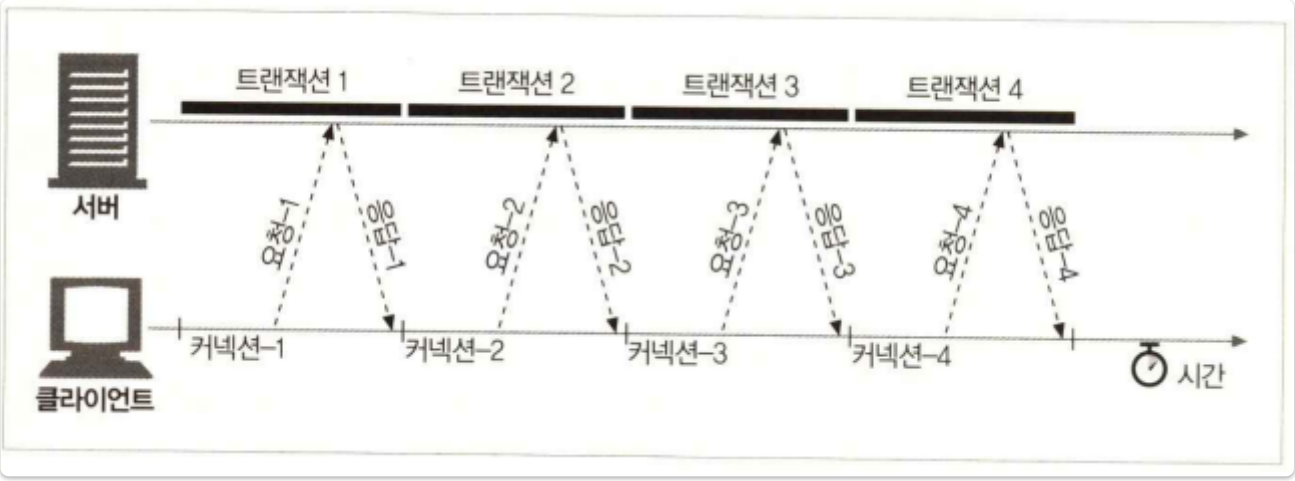
- TCP는 데이터 패킷을 받은 후 이를 확인응답(ACK)으로 응답한다.
- 데이터를 받은 후 ACK를 기다리는 동안 대기하므로, ACK 지연이 데이터 전송의 속도를 제한할 수 있다.

3.4 느린 시작 (Slow Start)

- TCP의 혼잡 제어 알고리즘 중 하나인 느린 시작은 네트워크 혼잡을 피하기 위해 처음에는 전송 속도를 느리게 증가시키는 방법이다.
- 초기에 전송할 수 있는 윈도우 크기를 작게 설정하고, 데이터가 성공적으로 전송되면 점진적으로 크기를 증가시킨다.
- 이로 인해 초기 데이터 전송 속도가 느릴 수 있다.

3.5 순차적인 트랜잭션 처리에 의한 지연

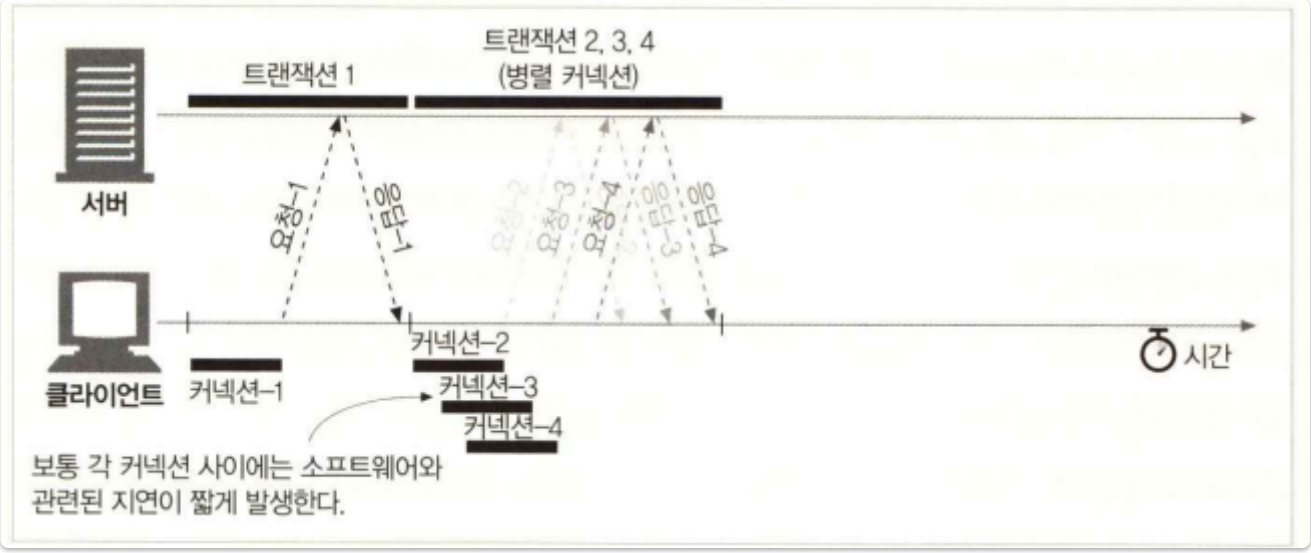
- TCP는 데이터의 신뢰성을 보장하기 위해 순차적인 트랜잭션 처리를 수행한다.
- 하지만 이로 인해 여러 개의 트랜잭션이 동시에 처리되지 못하고 대기해야 하는 상황이 발생할 수 있다.



4. 순차적인 트랜잭션 처리 지연 완화 방법

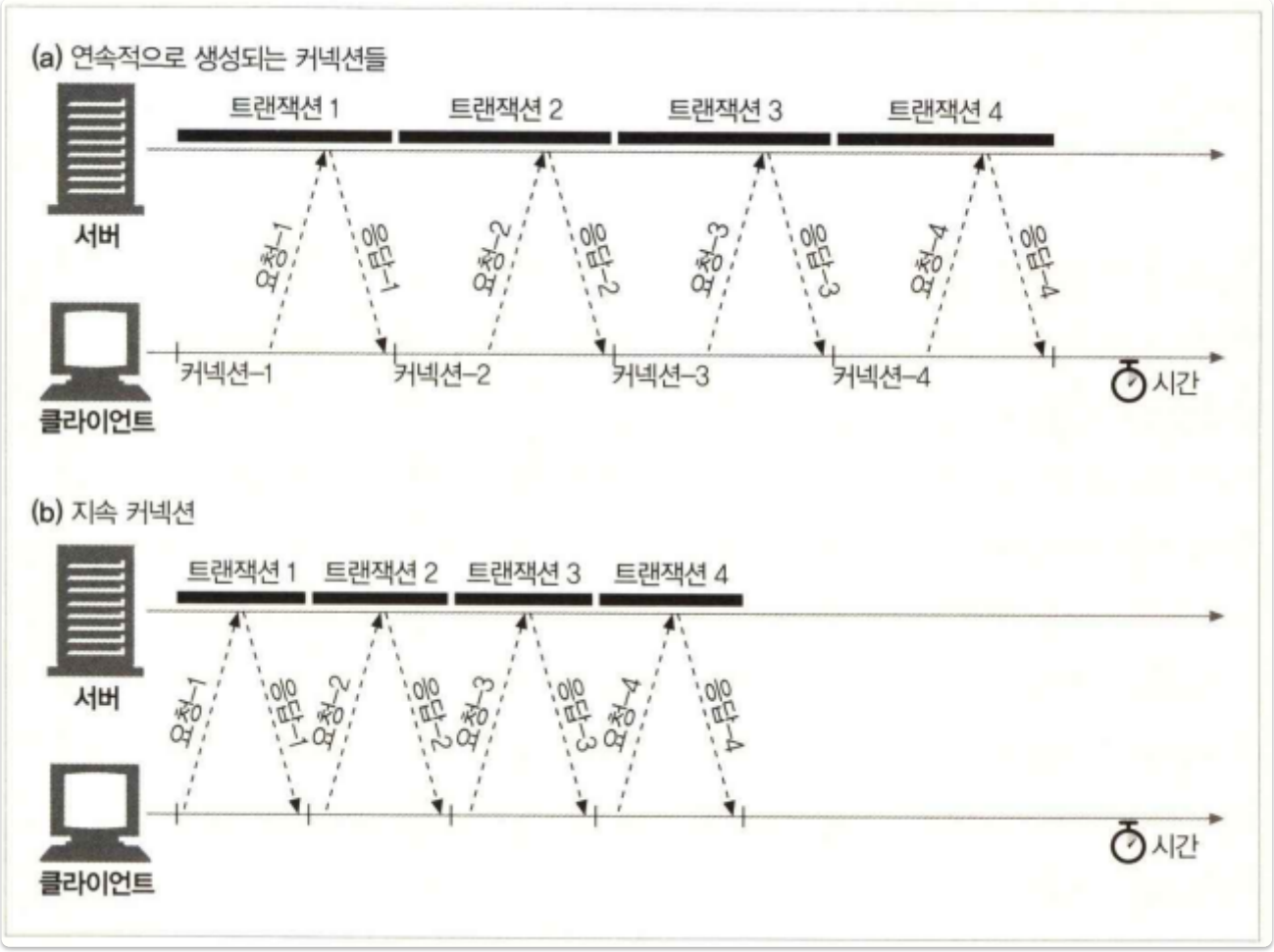
4.1 병렬 트랜잭션 처리

- 여러 개의 병렬 TCP 연결을 활용하여 동시에 여러 개의 트랜잭션을 처리할 수 있다.
- 이를 통해 하나의 트랜잭션이 다른 트랜잭션의 처리를 기다리는 지연을 줄일 수 있다.



4.2 Keep-Alive 커넥션 활용

- Keep-Alive 커넥션은 하나의 TCP 연결을 여러 번의 트랜잭션에 재사용하는 방식이다.
- 기본적으로 TCP 연결은 하나의 트랜잭션을 마치면 종료되지만, Keep-Alive 커넥션을 통해 연결을 유지하고 다수의 트랜잭션을 처리할 수 있다.
- 이를 통해 핸드셰이크 지연을 줄이고 성능을 향상시킬 수 있다.



4.3 Keep-Alive 활성화 방법

1. **HTTP 헤더를 통한 Keep-Alive 설정:** 웹 서버와 클라이언트 간의 통신에서 HTTP 헤더에 `Connection: keep-alive`를 포함시켜 Keep-Alive 커넥션을 활성화할 수 있다.
2. **서버/클라이언트 설정:** 서버나 클라이언트에서 Keep-Alive 기능을 활성화하는 설정을 변경할 수 있다.

4.4 Keep-Alive의 장단점

- 장점:
  - 핸드셰이크 오버헤드를 줄여 트랜잭션 처리 속도를 향상시킬 수 있다.
  - TCP 연결을 여러 번 재사용하여 시스템 리소스를 절약할 수 있다.
- 단점:
  - 지속적으로 연결을 유지하기 때문에 일부 시스템 리소스를 소비할 수 있다.
  - 일부 상황에서 연결이 잘못 유지될 수 있어 보안 취약점이 될 수도 있다.