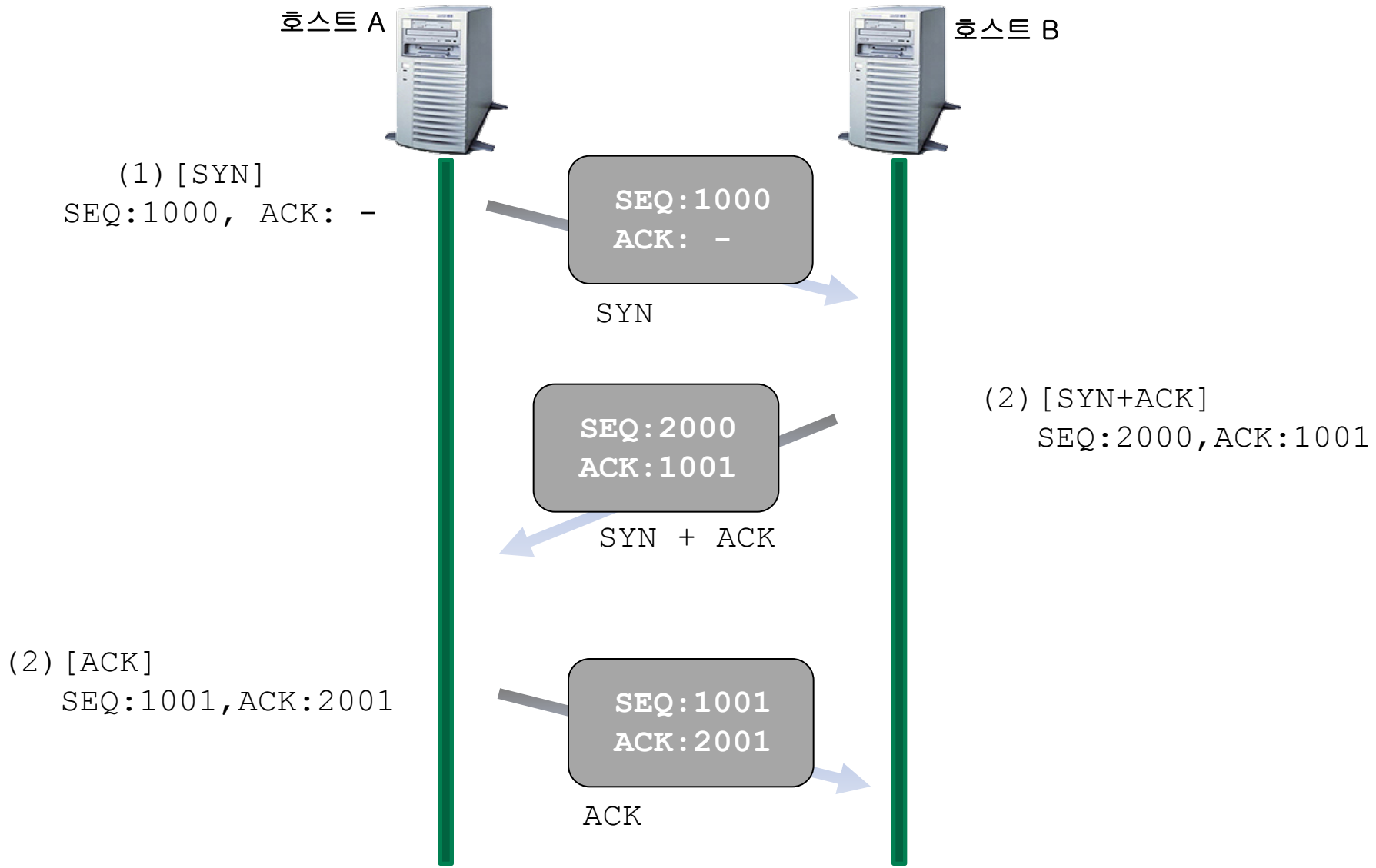
목차

6강
TCP 소켓 프로그래밍 V

1. TCP 내부동작 원리

1.TCP 연결 과정



Sunnyvale

1.TCP 연결 과정

- **TCP** 소켓의 연결설정 과정에서 총 세 번의 대화를 주고 받는다 **(3-Way Handshaking)**
- 데이터 송수신을 위한 준비를 하는 과정이다.
- 과정

(1) [SYN] SEQ:1000, ACK:-

- 연결 요청에 사용되는 메시지
- 동기화 메시지 **(SYN)**
- "이 패킷에 **1000**번이라는 번호를 부여하니, 잘 받으면 다음에 **1001**번을 보내달라고 요청해라"

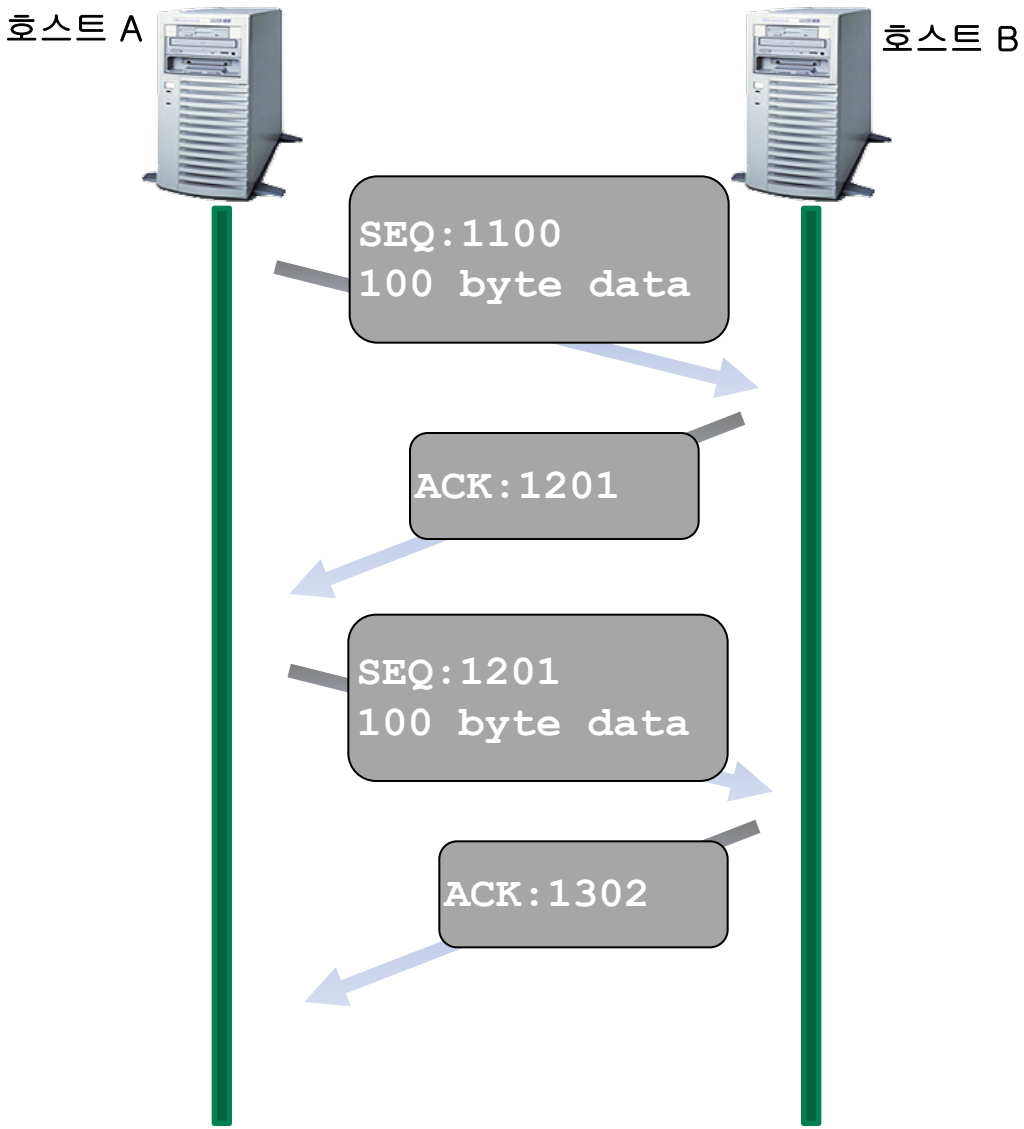
(2) [SYN + ACK] SEQ:2000, ACK:1001

- 동기화 메시지**(SYN)** + 응답 메시지**(ACK)**
- "이 패킷에 **2000**번이라는 번호를 부여하니, 잘 받으면 다음에 **2001**번을 보내 달라고 요청해라"
- "요청한 **1000**번 패킷은 잘 받았으니 다음에는 **SEQ**가 **1001**인 패킷을 보내라"

(3) [ACK] SEQ:1001, ACK:2001

- 응답 메시지**(ACK)**
- "요청한 **2000**번 패킷은 잘 받았으니 다음에는 **SEQ**가 **2001**인 패킷을 보내라"

2.TCP 데이터 송수신



Sunnyvale

2.TCP 데이터 송수신

❑ **200byte 데이터를 호스트 A가 호스트B에게 두 번에 나누어서 보내는 과정**

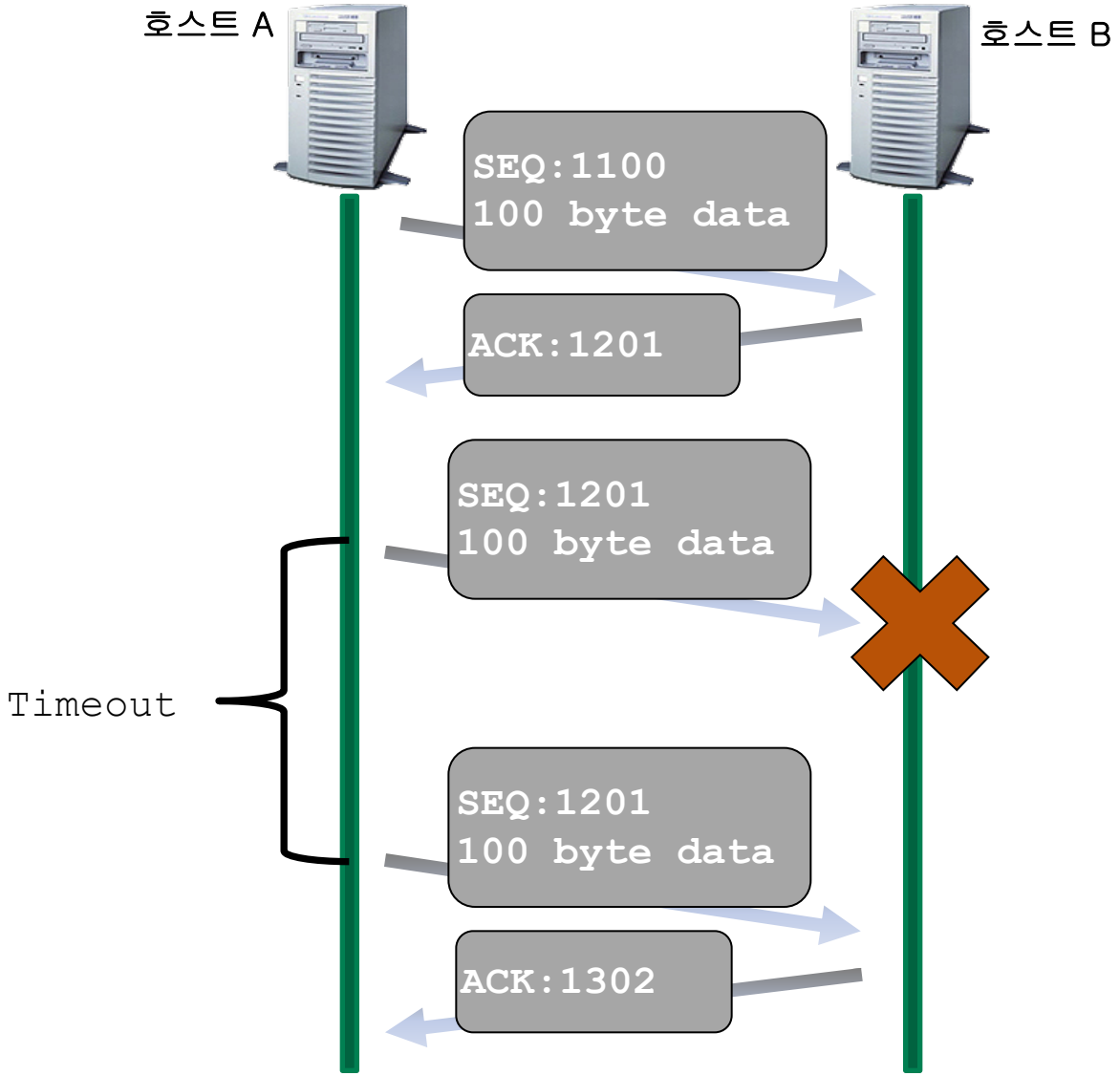
(1) 첫 패킷을 **SEQ 1100**으로 부여하고 **100byte**의 데이터를 보낸다.

(2) 첫 패킷을 받으면 호스트B는 제대로 수신되었음을 알려야 한다.
- **ACK 1201**인 이유는 **ACK**번호를 전송된 크기만큼 추가로 증가시켰기 때문이다.
- 이는 중간에 패킷에 담긴 데이터의 유실을 고려한 설계다.

ACK 번호 = SEQ 번호 + 전송된 바이트 크기 + 1

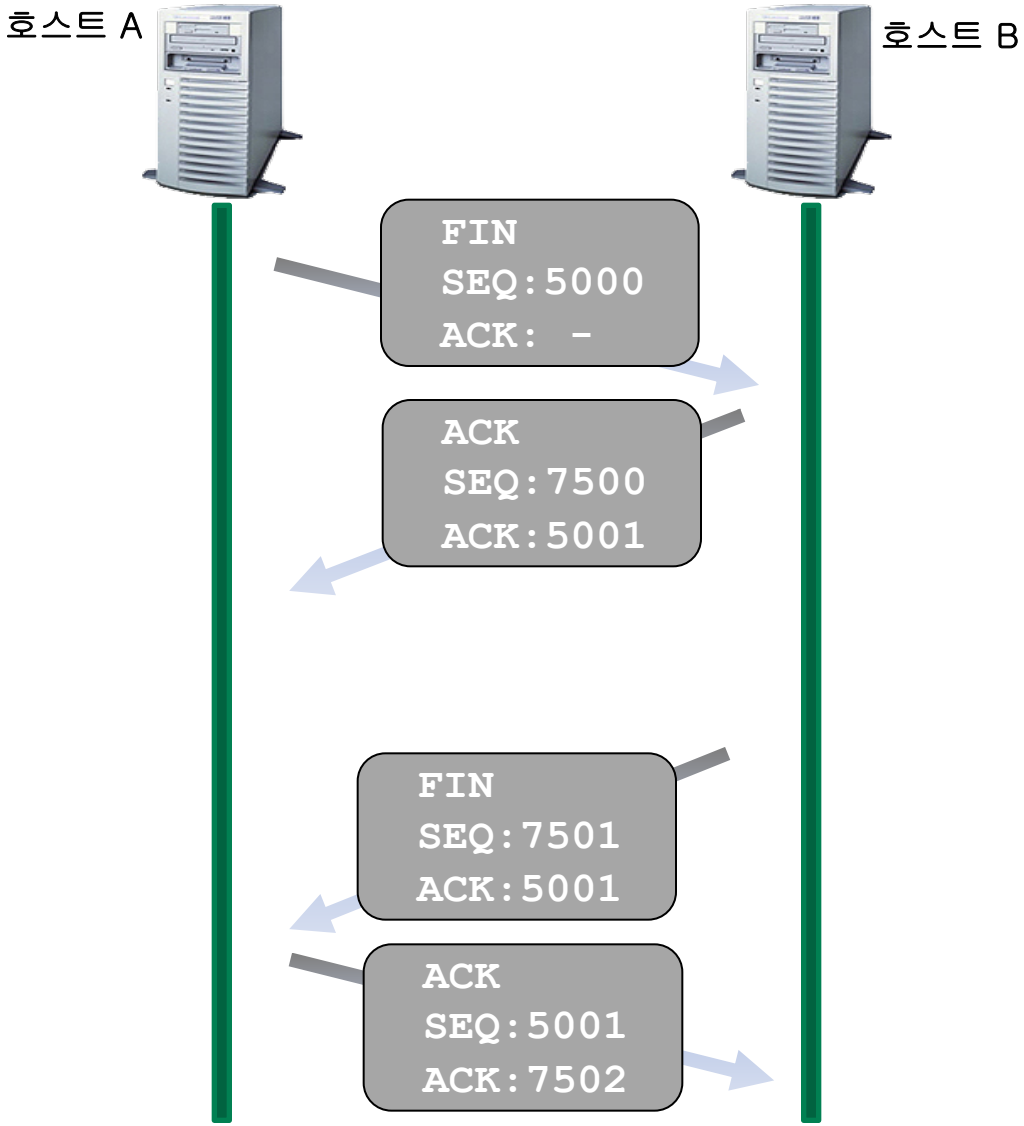
(3) 마지막에 **1**을 더한 이유는 다음 번에 전달 될 **SEQ** 번호를 요청하기 위해서이다.

2.TCP 데이터 송수신



Sunnyvale

3.TCP 연결종료



Sunnyvale

3.TCP 연결종료

- 연결종료에서는 바로 뚝! 끊지 않는다. (우아한 종료)
- 상대방이 전송할 데이터가 남아 있을 지 모르니 상호간에 연결종료에 대한 합의과정을 거친다.
- 패킷 안에 **FIN**은 종료를 알리는 메시지를 뜻한다.
- 상호간에 **FIN** 메시지를 한 번씩 주고 받는다 **(4-Way Handshaking)**

목차

6강
TCP 소켓 프로그래밍 V

1. TCP Socket Option

1. SO_SNDBUF 와 SO_RCVBUF

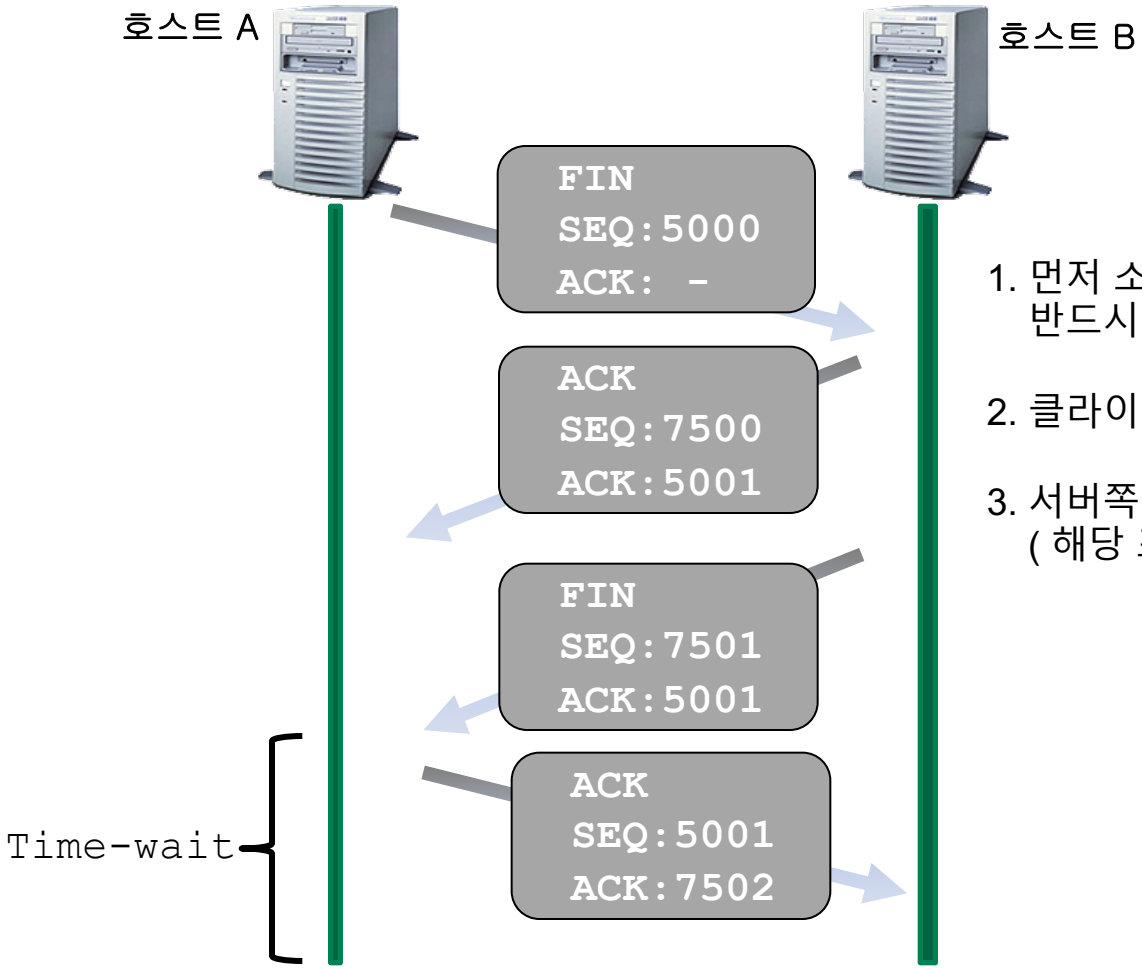
- 소켓의 입출력 버퍼의 크기대한 옵션이다.
- 입출력 버퍼의 크기를 참조할 수 있고 변경할 수 있다.
- 변경은 주의 깊게 다루어야 하는 부분이고 요구한 크기로 변경되지 않을 수도 있다.

```
//Socket Buffer Size 참조
int receiveBufferSize = socket.getReceiveBufferSize()
int sendBufferSize = socket.getSendBufferSize()

//Socket Buffer Size 변경
socket.setReceiveBufferSize( 1024*10 );
socket.setSendBufferSize( 1024*10 );
```

2. SO_REUSEADDR

- **Time-wait 상태의 소켓에 Port 번호 재할당을 가능하게 하는 것이다.**
- **Time-wait 상태란?**



1. 먼저 소켓을 끊는 쪽(FIN 메시지를 전송한 호스트) 에서는 반드시 거치는 과정이다.

2. 클라이언트의 Time-wait은 별 문제가 없다.

3. 서버쪽에서는 문제가 발생할 여지가 있다.
   (해당 포트를 다시 사용할 수 없다.)

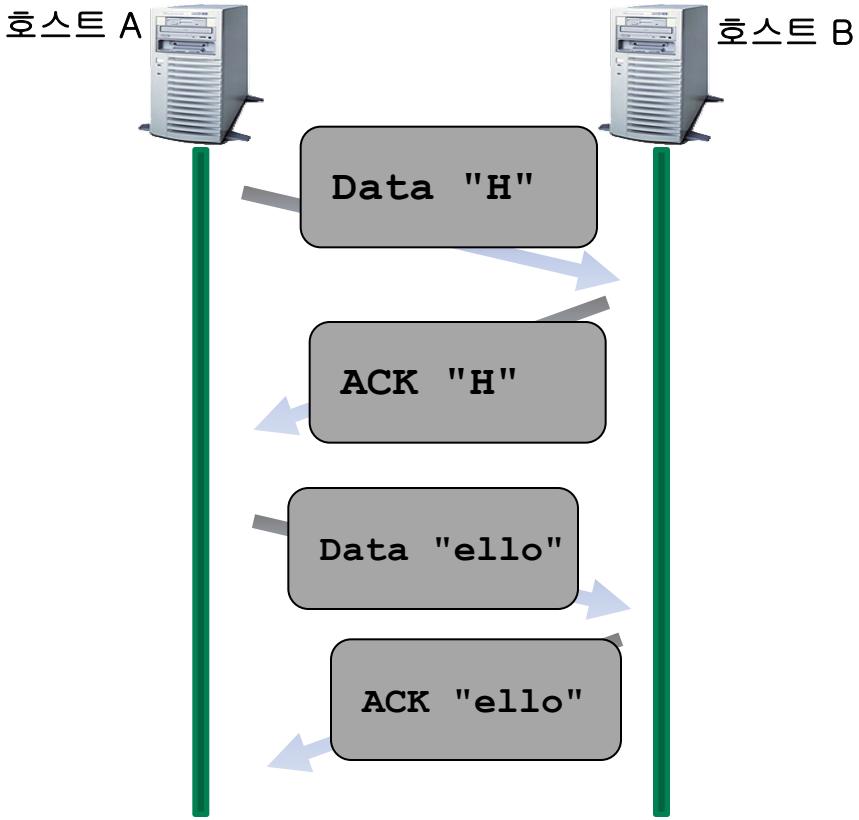
2. SO_REUSEADDR

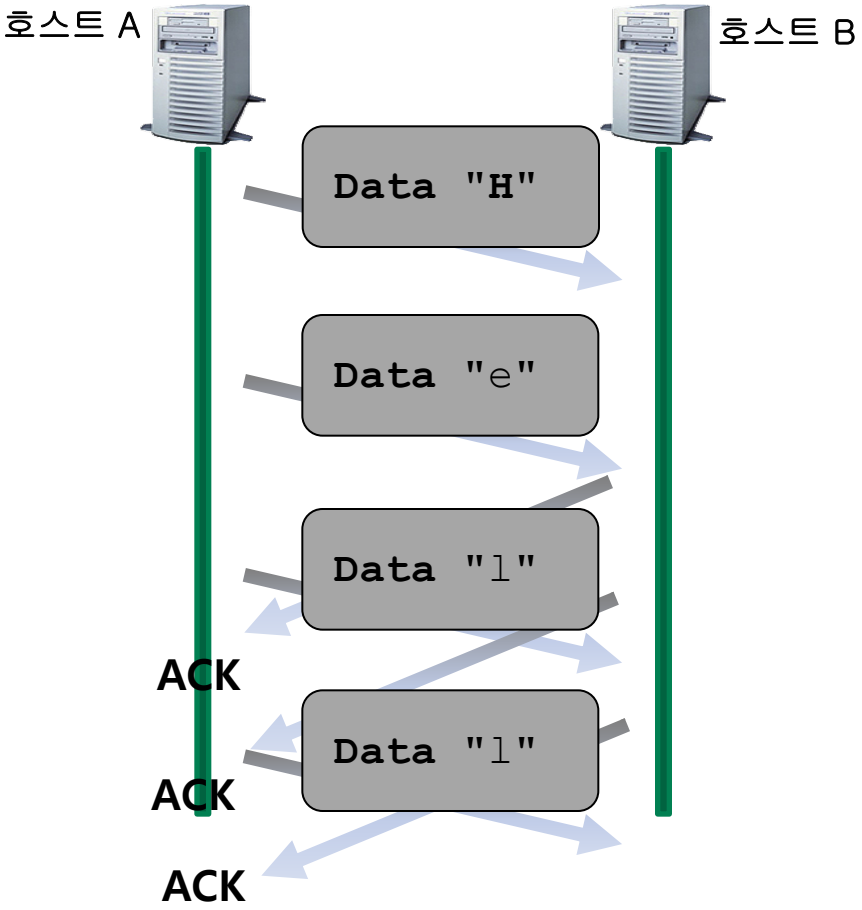
- **SO_REUSEADDR** 옵션은 서버쪽의 **Time-wait** 상태에서의 문제점을 해결한다.
- 디폴트 값은 **0(false)**이고 이는 **Time-wait** 상태에 있는 소켓의 같은 **Port**번호로 재 할당이 불가능함을 의미한다.
- **1(True)**로 변경하면 재할당이 가능해진다.

```
//Time-wait 상태에서 서버 재실행이 가능하게 끔 함
serverSocket.setReuseAddress( true );
```

3. TCP_NODELAY

- Nagle 알고리즘 적용하지 않는 옵션이다.
- Nagle 알고리즘이란?





Sunnyvale

4. SO_TIMEOUT

- 소켓에 데이터를 쓰고/읽는 타임아웃을 정한다.
- **Blocking** 되는 소켓 메소드에만 적용 될 수 있다. **(read)**
- 만일 시간이 경과하게 되면 **SocketTimeoutException** 를 받게 된다.

```
//데이터 읽기에 타임아웃 설정
socket.setSoTimeout( 3000 );
```