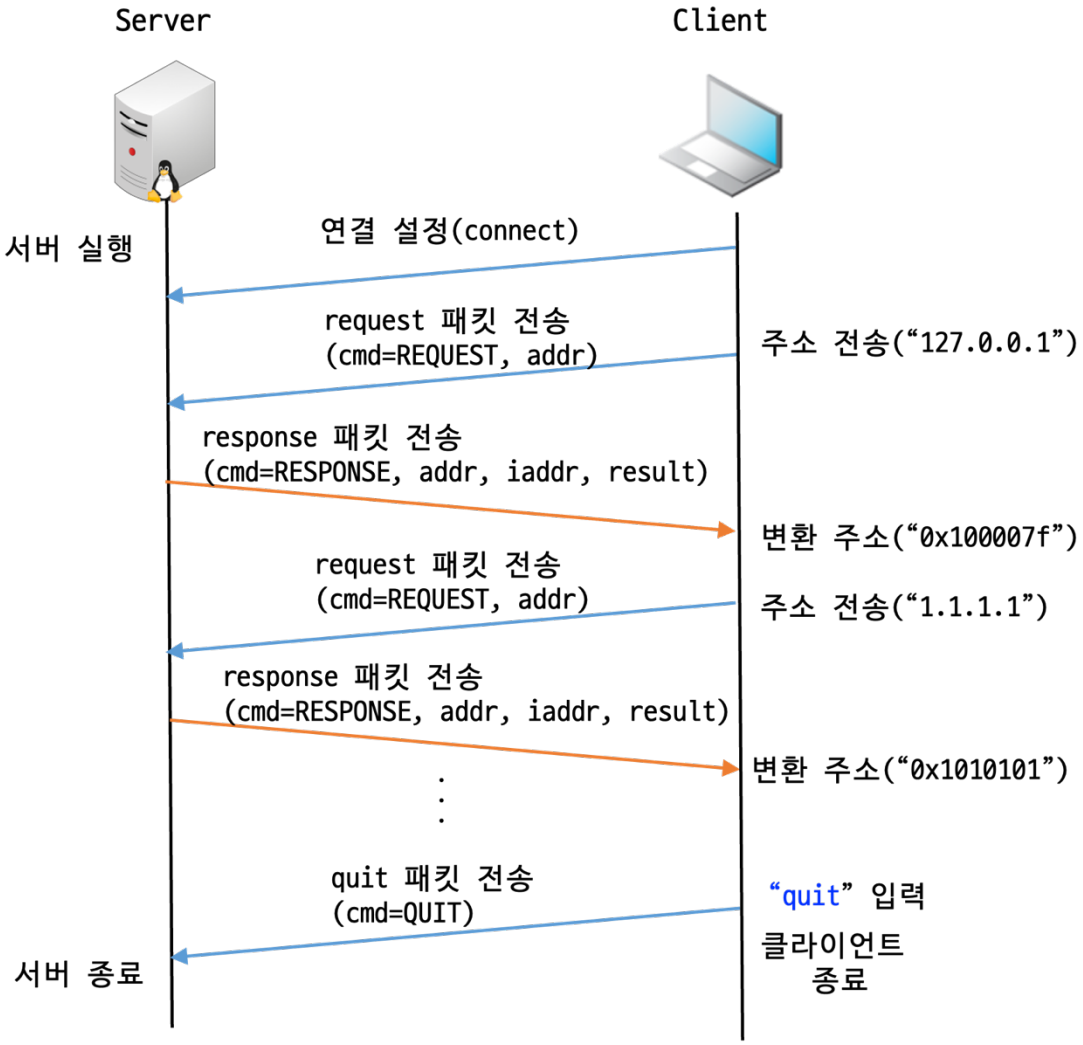
네트워크 프로그래밍 과제 #01

1. 소켓 통신을 이용한 주소 변환 프로그램 (20점)

제출 파일: hw1_server.c, hw1_client.c

- 소스 파일에 반드시 주석 (학번, 이름) 추가하세요(한글 또는 영문).

■ 서버와 클라이언트 동작 과정 및 메시지 흐름



■ 공동 구조체 타입 (데이터 송수신시 클라이언트, 서버 모두 사용)

```
#define ERROR 0
#define SUCCESS 1

#define BUF_SIZE 20

#define REQUEST 0
#define RESPONSE 1
#define QUIT 2

typedef struct {
    int cmd;                // 0: request, 1: response, 2: quit
    char addr[BUF_SIZE];    // dotted-decimal address
    struct in_addr iaddr;   // inet_aton() result
    int result;             // 0: Error, 1: Success
}PACKET;
```

✓ PACKET 구조체를 사용하지 않고 구현한 경우, 0점 처리함

■ 클라이언트 기능 (9점)

✓ 클라이언트는 dotted-decimal 형태의 주소("192.168.0.1")를 화면에서 입력 받고 PACKET 구조체의 addr 배열에 담아서 서버로 전송 (서버로 전송시 cmd 값은 0(request)으로 설정) (5점)

✓ 서버가 전송한 PACKET 데이터에서 result값이 정상(1)인 경우, 주소 변환 결과 (iaddr 변수의 값)를 화면에 출력함 (3점)

✓ result 변수의 값이 0인 경우, "Address conversion fail!" 을 화면에 출력 (1점)

■ 서버 기능 (9점, 각 3점)

✓ 클라이언트가 보낸 dotted-decimal 주소를 inet_aton() 함수를 호출하여 변환하고, 그 결과가 정상인 경우에 서버쪽 화면에 출력 (실행 결과 참고)

✓ 주소가 정상인 경우: 서버는 PACKET 구조체 변수에 변환된 주소 및 그 결과 (cmd=RESPONSE , result=SUCCESS, iaddr=변환된 주소)를 저장하여 클라이언트에게 다시 전송하며, "Address conversion (전송주소) -> 변환된 주소"를 화면에 출력

✓ 잘못된 주소인 경우: "Address conversion fail(클라이언트가 전송한 주소)" 를 출력하고, 그 결과(cmd=RESPONSE, result=ERROR)를 클라이언트에게 전송함

■ 클라이언트와 서버는 무한 반복하며 클라이언트가 "quit" 메시지(cmd=2)를 보내면 아래와 같이 메시지를 출력하고 클라이언트와 서버 프로그램을 모두 종료함 (2점, 각 1점)

■ 문자열 처리 과정(화면 출력 등)에서 쓰레기 값이 출력되는 경우 -5점

[실행 결과]

서버 실행 #1: bind() 에러가 발생할 경우, 포트 번호를 변경해서 테스트 하면 됨

```
$ ./hw1_server 9190
---------------------------
 Address Conversion Server
---------------------------
[Rx] Received Dotted-Decimal Address: 1.1.1.1
inet_aton(1.1.1.1) -> 0x1010101
[Tx] cmd: 1, iaddr: 0x1010101, result: 1

[Rx] Received Dotted-Decimal Address: 127.0.0.1
inet_aton(127.0.0.1) -> 0x100007f
[Tx] cmd: 1, iaddr: 0x100007f, result: 1

[Rx] Received Dotted-Decimal Address: 155.230.256.256
[Tx] Address conversion fail:(155.230.256.256)

[Rx] QUIT message received
Server socket close and exit.
```

서버 실행 #2: 명령행 인자의 수가 다른 경우, 에러 메시지 출력 후 종료

```
$ ./hw1_server
Usage : ./hw2_server <port>
```

클라이언트 실행 #1

```
$ ./hw1_client 127.0.0.1 9190
Input dotted-decimal address: 1.1.1.1
[Tx] cmd: 0, addr: 1.1.1.1
[Rx] cmd: 1, Address conversion: 0x1010101 (result: 1)

Input dotted-decimal address: 127.0.0.1
[Tx] cmd: 0, addr: 127.0.0.1
[Rx] cmd: 1, Address conversion: 0x100007f (result: 1)

Input dotted-decimal address: 155.230.256.256 // 잘못된 주소 입력
[Tx] cmd: 0, addr: 155.230.256.256
[Rx] cmd: 1, Address conversion fail! (result: 0)

Input dotted-decimal address: quit
[Tx] cmd: 2(QUIT)
Client socket close and exit
```

클라이언트 실행 #2: 명령행 인자의 수가 다른 경우

```
$ ./hw1_client 9190
Usage : ./hw2_client <IP> <port>
```

■ 서버 구현 예시

```
while(1) {
    rx_len = read(clnt_sock, &recv_packet, sizeof(PACKET));
    if(rx_len == 0)
        break;

    if(recv_packet.cmd == REQUEST) {
        printf("[Rx] Received Dotted-Decimal Address: %s\n", recv_packet.addr);

        if(inet_aton(recv_packet.addr, &iaddr) == 0) {
            // 주소 변환 실패
            write(. . .);
        }
        else {
            // 주소 변환 성공
             write(. . .);
        }
    }
    else if(recv_packet.cmd == QUIT) {
            . . .
        break;
    }
    else {
        printf("[Rx] Invalid command: %d\n", recv_packet.cmd);
        break;
    }
}
```

수신된 패킷의 cmd를 반드시 확인

예외 처리