

# TCP, UDP 소켓통신 프로그램(클라이언트, 서버 총 4개)

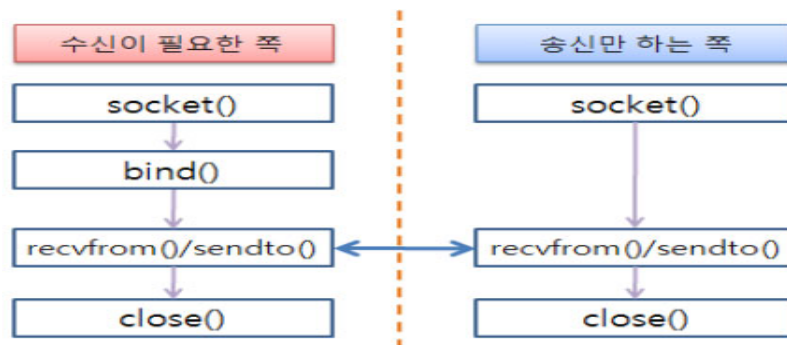
## 1. 프로그램 목록

- 1) TCP 클라이언트 1개
- 2) TCP 서버 1개
- 3) UDP 클라이언트 1개
- 4) UDP 서버 1개

## 2. 개념 정리

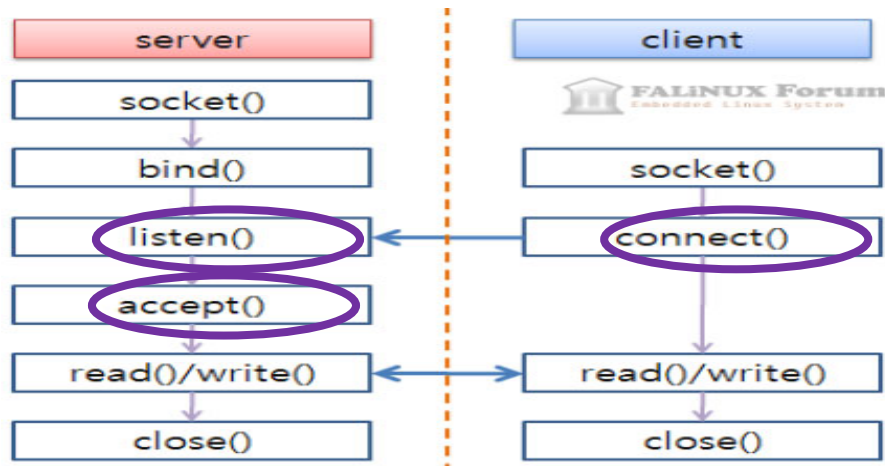
UDP(User Datagram Protocol)란?

: 인터넷에서 정보를 주고받을 때, 서로 주고받는 형식이 아닌 한쪽에서 일방적으로 보내는 방식의 통신 프로토콜. 비 연결지향 방식



TCP(Transmission Control Protocol)란?

: 서로 다른 운영체제를 쓰는 컴퓨터 간에도 데이터를 전송할 수 있어 인터넷에서 정보전송을 위한 표준 프로토콜로 사용됨. 연결지향 방식



### 3. 함수 설명

1) accept() 함수 : 소켓으로부터 연결을 받아들이는 함수로 아직 처리되지 않은 연결들이 대기하고 있는 큐에서 제일 처음 연결된 연결을 가져와서 새로운 소켓을 만든다. 연결에 성공하면 0보다 큰 파일 지정번호를 반환하고 실패(에러)시 -1을 반환한다.

int accept(int s, struct sockaddr \*addr, socklen\_t \*addrlen); 형식으로 사용함

2) memset() 함수 : 메모리의 내용을 원하는 크기만큼 특정한 값으로 설정할 수 있으며 memory set의 줄임말이다. memset(포인터, 설정할 값, 크기); 형식으로 사용함

3) write() 함수 : txt파일의 내용을 입력하는 함수. 리눅스에서는 파일과 소켓을 동일하게 취급하므로 소켓을 통해서 다른 호스트에게 데이터를 전송할 때도 사용함

4) read() 함수 : txt파일의 내용을 읽어오는 함수. 리눅스에서는 파일과 소켓을 동일하게 취급하므로 소켓을 통해서 다른 호스트에게 데이터를 수신할 때도 사용함

5) bzero()함수 : 메모리 공간을 size 바이트만큼 0으로 채우는 함수

6) socket() 함수 : 소켓을 생성하는 함수. socket(domain, type, protocol) 형식으로 사용함

7) strlen() 함수 : char\*가 가리키는 주소에서부터 시작해서 '\0' 문자가 나올 때까지의 문자들의 개수를 카운팅하여 최종 길이를 반환하는 함수

8) strcpy() 함수 : 문자열을 복사하는 함수. strcpy(대상 문자열, 원본 문자열) 형식으로 사용함

9) bind() 함수 : 소켓에 IP주소와 포트 번호를 지정해주는 함수로 소켓 통신 준비를 함  
bind(int sockfd, struct \*myaddr, socklen\_t addrlen) 형식으로 사용

10) socket(int domain, int type, int protocol) 함수 : 소켓을 만드는 함수

- Domain : 프로토콜 체계 ex) PF\_INET

- Type : 데이터 전송 방법 ex) SOCK\_STREAM(TCP), SOCK\_DGRAM(UDP)

- Protocol : 호스트와 호스트 사이에 사용할 규칙 ex) IPPROTO\_TCP, IPPROTO\_UDP

11) listen() 함수 : 클라이언트가 ServerSocket에 부여한 IP와 PORT로 접속했는지를 감시  
int listen(int socket, int backlog) 형식으로 사용함

12) send() 함수 : 클라이언트로 데이터를 전송하는 함수

int send(int socket, const char FAR\* buf, int len, int flags) 형식으로 사용함

13) recv() 함수 : send 함수를 이용해서 서버로 전달한 데이터를 읽어 들이는 함수  
int recv(int socket, char FAR\* addr, int len, int flags) 형식으로 사용함

14) send to() 함수 : 지정된 주소로 데이터를 보내는 기능의 함수  
int sendto(int socket, const void \*msg, int len, unsigned flags, const struct  
sockaddr \* addr, int addrlen) 형식으로 사용함

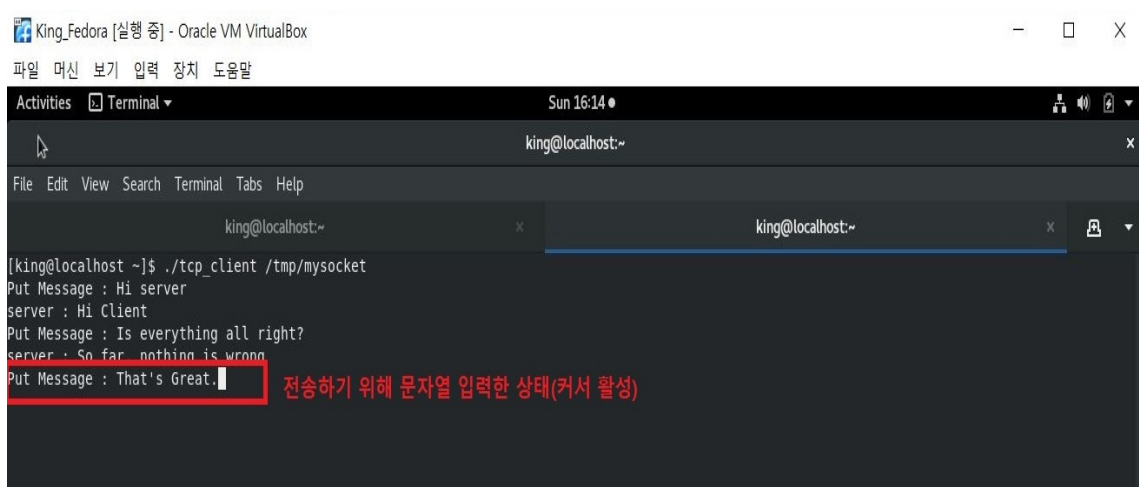
15) recvfrom() 함수 : sendto 함수를 이용해서 전달한 데이터를 읽어 들이는 함수  
int recvfrom(int socket, const void \*msg, int len, unsigned flags, const struct  
sockaddr \* addr, int addrlen) 형식으로 사용

## 4. 실행 결과 스크린샷

### 1) TCP 프로그램 2개(tcp\_server, tcp\_client)

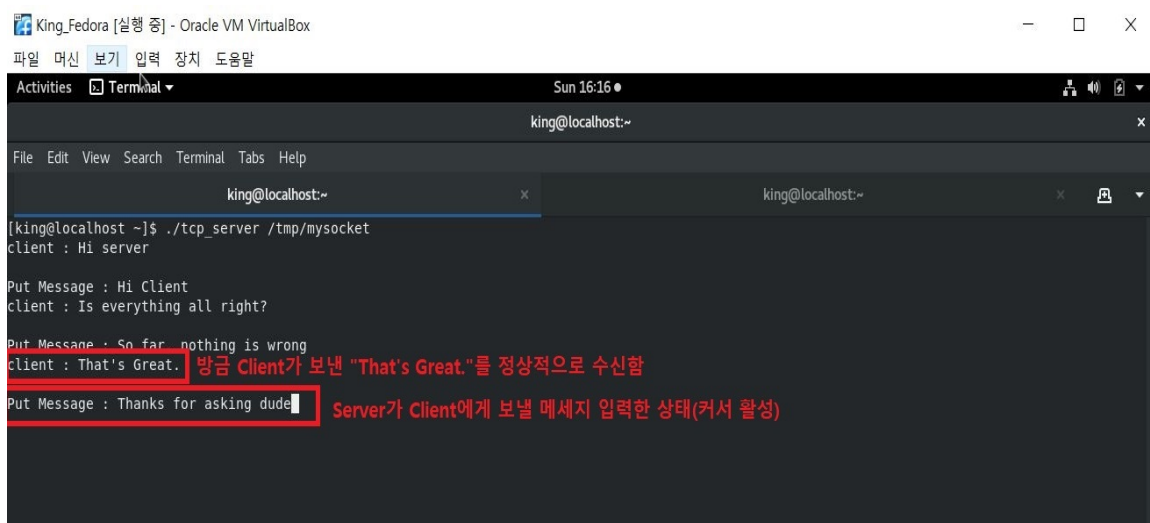
- i) 서버가 연결을 기다림
  - ii) 클라이언트가 연결을 시도
  - iii) 연결되면 클라이언트에서 메시지 송신 활성화 및 메시지 입력 후 송신
  - iv) 서버는 클라이언트가 보낸 메시지 수신 후 송신 활성화
  - v) 서버가 클라이언트로 메시지 송신
  - vi) 클라이언트는 서버가 보낸 메시지 수신 및 다시 송신 활성화
- ※ 서버와 클라이언트가 양방향으로 메시지를 주고받는 프로그램

### 1) tcp\_client 대화 전송 전 커서 활성화 장면



```
King_Fedora [실행 중] - Oracle VM VirtualBox
파일  머신  보기  입력  장치  도움말
Activities  Terminal  Sun 16:14
king@localhost:~
File Edit View Search Terminal Tabs Help
king@localhost:~
[king@localhost ~]$ ./tcp_client /tmp/mysocket
Put Message : Hi server
server : Hi Client
Put Message : Is everything all right?
server : So far nothing is wrong
Put Message : That's Great.  전송하기 위해 문자열 입력한 상태(커서 활성화)
```

## 2) tcp\_server가 메시지 수신 후 대화 전송 전 커서 활성화 장면

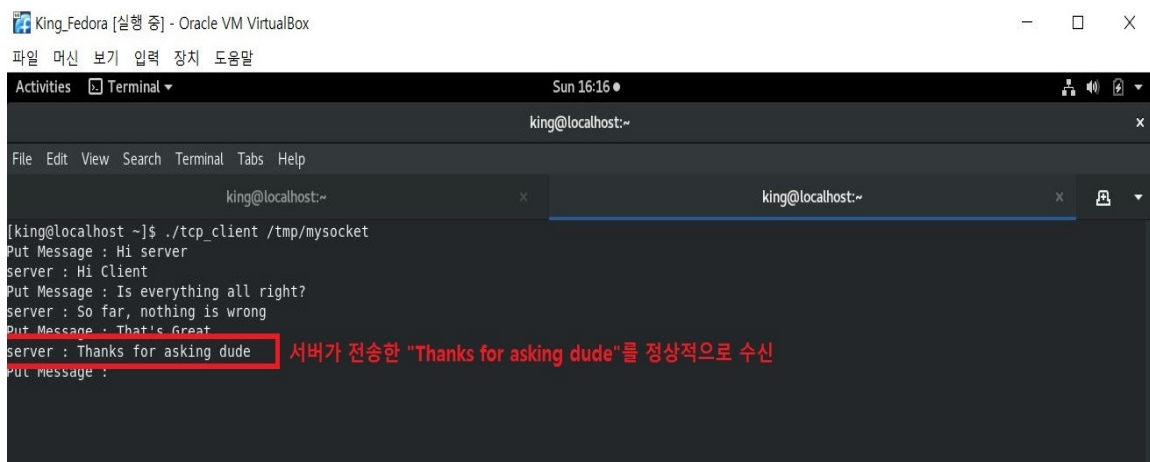


```
King_Fedora [실행 중] - Oracle VM VirtualBox
파일  머신  보기  입력  장치  도움말
Activities  Terminal
Sun 16:16
king@localhost:~
File Edit View Search Terminal Tabs Help
king@localhost:~
[king@localhost ~]$ ./tcp_server /tmp/mysocket
client : Hi server
Put Message : Hi Client
client : Is everything all right?
Put Message : So far, nothing is wrong
client : That's Great.
Put Message : Thanks for asking dude
```

방금 Client가 보낸 "That's Great."를 정상적으로 수신함

Server가 Client에게 보낼 메시지 입력한 상태(커서 활성화)

## 3) tcp\_server가 보낸 메시지를 tcp\_client가 수신 후에 다시 전송 활성화



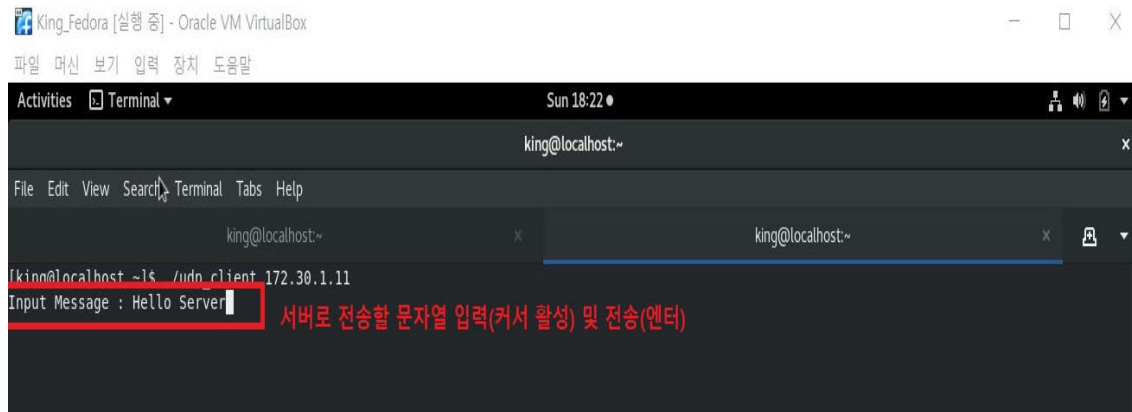
```
King_Fedora [실행 중] - Oracle VM VirtualBox
파일  머신  보기  입력  장치  도움말
Activities  Terminal
Sun 16:16
king@localhost:~
File Edit View Search Terminal Tabs Help
king@localhost:~
[king@localhost ~]$ ./tcp_client /tmp/mysocket
Put Message : Hi server
server : Hi Client
Put Message : Is everything all right?
server : So far, nothing is wrong
Put Message : That's Great
server : Thanks for asking dude
```

서버가 전송한 "Thanks for asking dude"를 정상적으로 수신

## 2) UDP 프로그램 2개(udp\_server, udp\_client)

- i) 소켓을 생성하여 IP주소와 포트 지정
  - ii) 클라이언트가 IP주소와 포트로 데이터 전송
  - iii) 서버는 클라이언트가 보낸 메시지 수신 후 송신 활성화
  - iv) 서버가 클라이언트로 메시지 송신(같은 방법 - sendto, recvfrom)
  - v) 클라이언트가 서버가 보낸 메시지 수신 후 다시 송신 활성화
- ※ 서버와 클라이언트가 양방향으로 메시지를 주고받는 프로그램

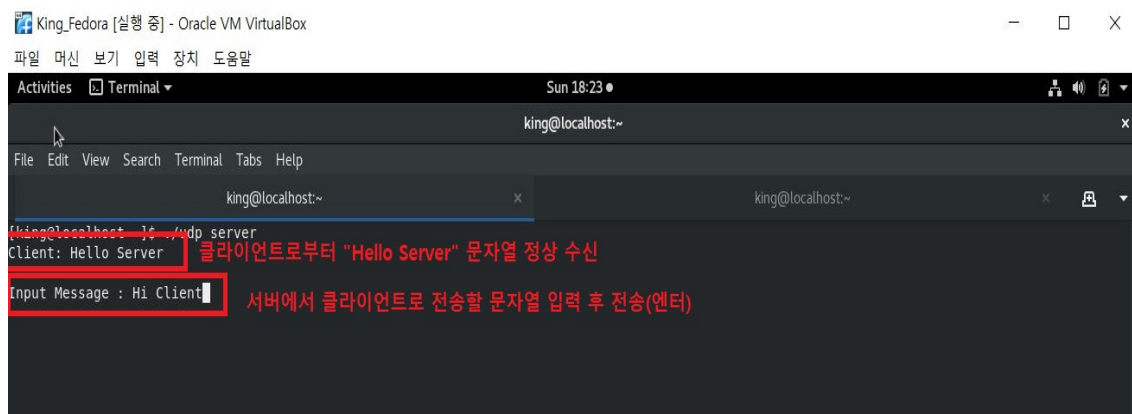
### 1) udp\_client에서 server로 메시지 전송 전 커서 활성화 단계



```
King_Fedora [실행 중] - Oracle VM VirtualBox
파일  머신  보기  입력  장치  도움말
Activities  Terminal
Sun 18:22
king@localhost:~
File Edit View Search Terminal Tabs Help
king@localhost:~
king@localhost:~
[king@localhost ~]$ ./udp_client 172.30.1.11
Input Message : Hello Server
```

서버로 전송할 문자열 입력(커서 활성화) 및 전송(엔터)

### 2) udp\_server 수신 후 대화 전송 전 커서 활성화 단계

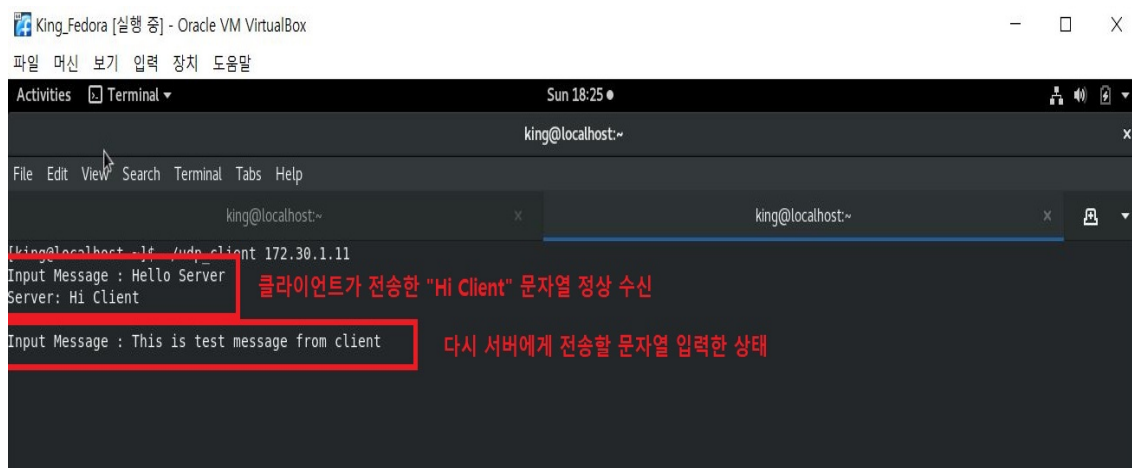


```
King_Fedora [실행 중] - Oracle VM VirtualBox
파일  머신  보기  입력  장치  도움말
Activities  Terminal
Sun 18:23
king@localhost:~
File Edit View Search Terminal Tabs Help
king@localhost:~
king@localhost:~
[king@localhost ~]$ ./udp_server
Client: Hello Server
Input Message : Hi Client
```

클라이언트로부터 "Hello Server" 문자열 정상 수신

서버에서 클라이언트로 전송할 문자열 입력 후 전송(엔터)

### 3) udp\_client에서 서버가 전송한 대화 수신



```
King_Fedora [실행 중] - Oracle VM VirtualBox
파일  머신  보기  입력  장치  도움말
Activities  Terminal
Sun 18:25
king@localhost:~
File Edit View Search Terminal Tabs Help
king@localhost:~
king@localhost:~
[king@localhost ~]$ ./udp_client 172.30.1.11
Input Message : Hello Server
Server: Hi Client
Input Message : This is test message from client
```

클라이언트가 전송한 "Hi Client" 문자열 정상 수신

다시 서버에게 전송할 문자열 입력한 상태