TCP 소켓 프로그래밍 2

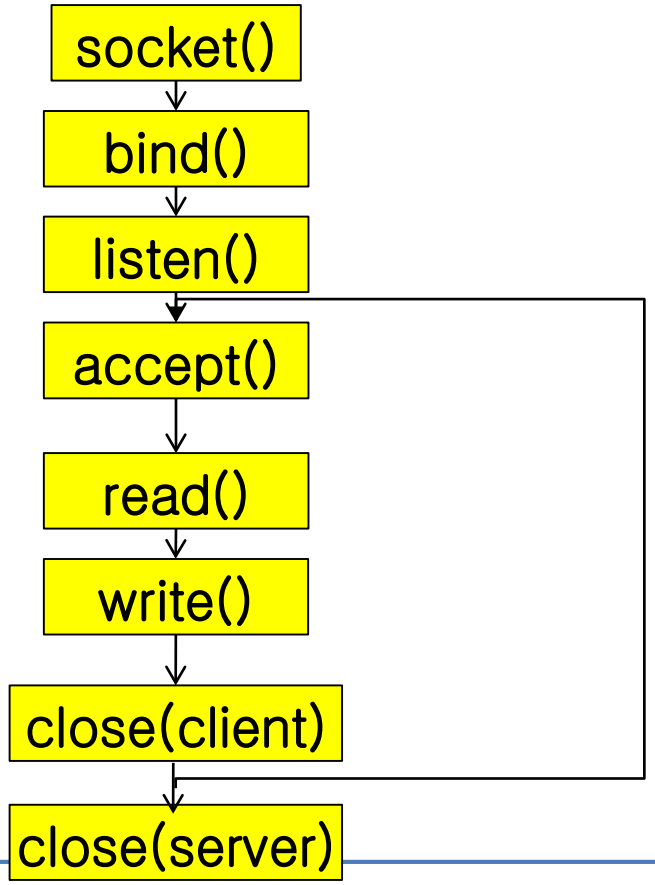
컴퓨터 통신 실습

홍석준

1. TCP 소켓 프로그램 작성

TCP 소켓 프로그램 작성(서버 프로그램)

- 아래 그림과 같이 여러 클라이언트를 수락하는 TCP소켓 echo 서버 프로그램 작성
- tcp_server2.c 파일로 작성(gedit tcp_server2.c명령 사용)



1. TCP 소켓 프로그램 작성

TCP 소켓 프로그램 작성(서버 프로그램)

❑ **Hint : tcp_server1.c의 아래 소스 코드 부분이 반복될 수 있도록 수정**

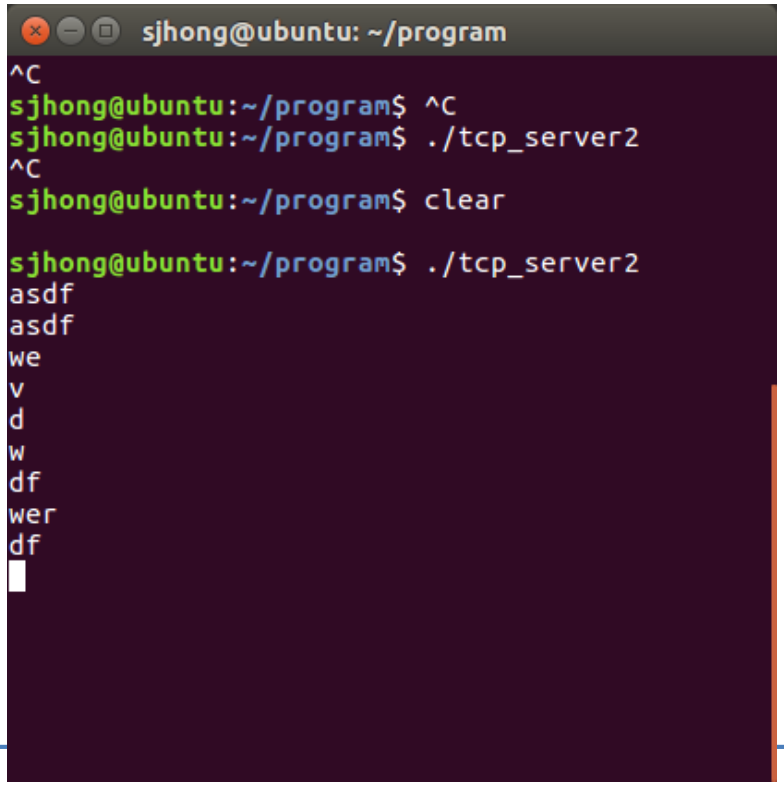
```
        memset(&serv_addr, 0, sizeof(serv_addr));
        serv_addr.sin_family=AF_INET;
        serv_addr.sin_addr.s_addr=htonl(INADDR_ANY);
        serv_addr.sin_port=htons(4000);

        if(bind(serv_sock, (struct sockaddr*) &serv_addr, sizeof(serv_addr)) == -1)
                error_handling("bind() error");

        if(listen(serv_sock, 5) == -1)
                error_handling("listen() error");

        clnt_addr_size = sizeof(clnt_addr);
        clnt_sock=accept(serv_sock, (struct sockaddr*) &clnt_addr, &clnt_addr_size);

        if(clnt_sock == -1)
                error_handling("accept() error");

        while((str_len = read(clnt_sock, message, BUFSIZE)) != 0) {
                write(clnt_sock, message, str_len);
                write(1, message, str_len);
        }
        close(clnt_sock);

        return 0;
}

void error_handling(char * message)
{
        fputs(message, stderr);
        fputc('\n', stderr);
        exit(1);
}
```

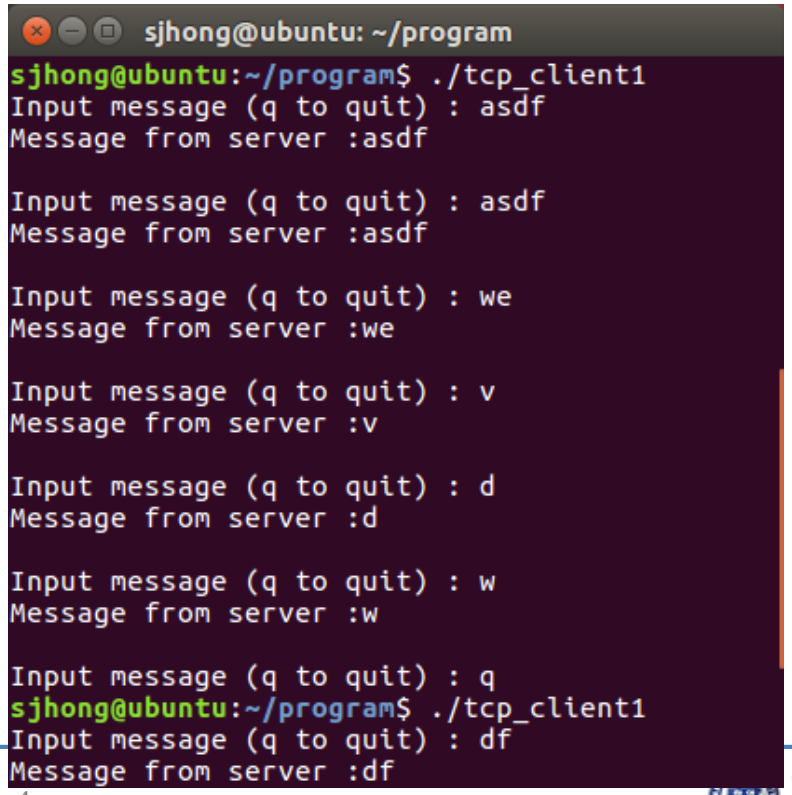
(끝)

2. TCP 소켓 프로그램 컴파일 및 실행

TCP 소켓 프로그램 컴파일 및 실행

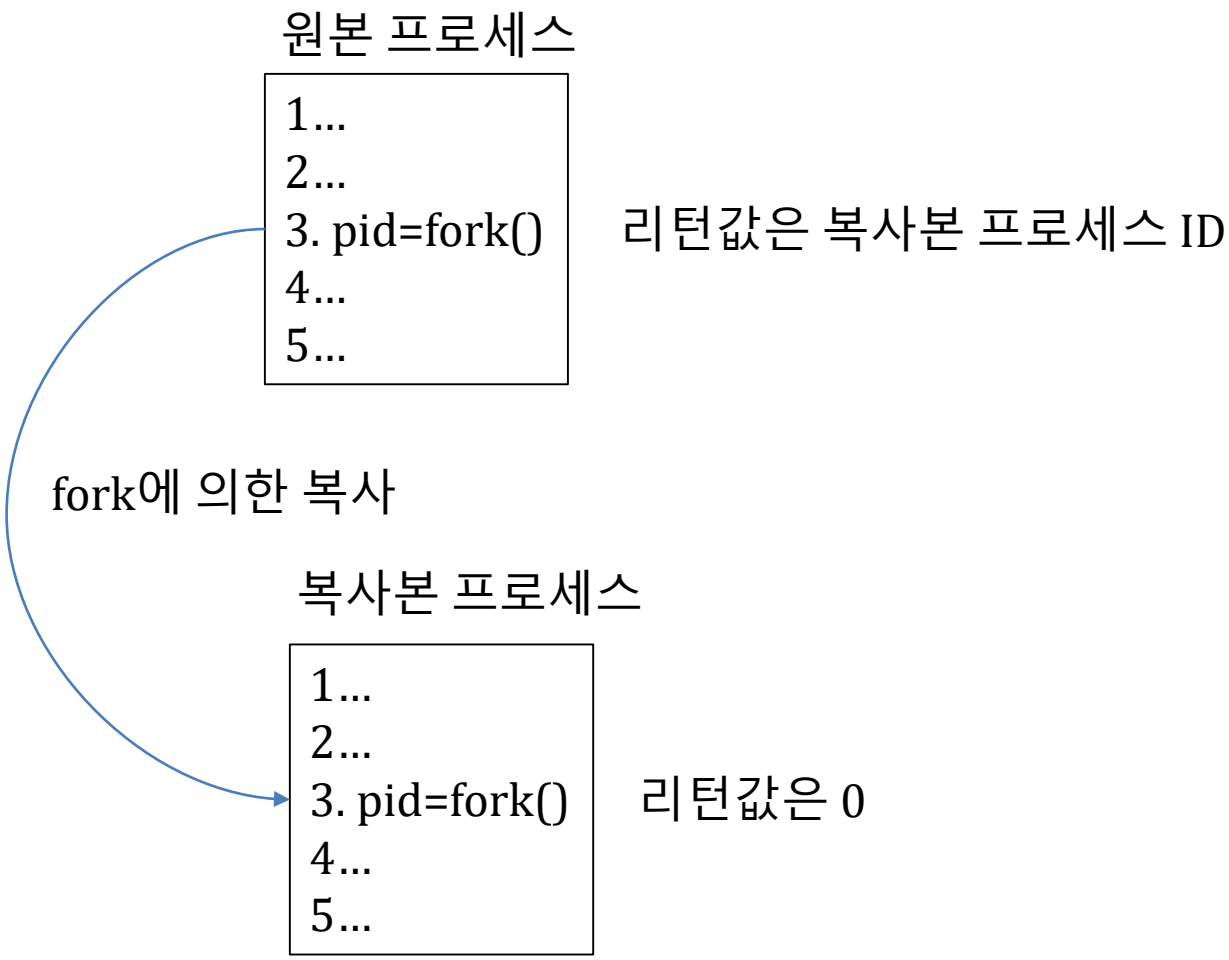
- gcc –o tcp_server2 tcp_server2.c 컴파일 및 실행 확인
- 아래 그림처럼 실행되는지 확인(클라이언트는 기존 tcp_client1사용)
  - 클라이언트 종료 (q)후에도 서버가 종료되지 않는지를 확인
  - 다른 클라이언트 혹은 재접속을 하는 경우에도 계속 접속 가능한지 확인





프로세스의 생성

fork 함수 호출을 통한 프로세스 생성



프로세스의 생성

fork 함수 호출을 통한 프로세스 생성

- 아래 파일을 fork.c로 작성
- gcc –o fork fork.c로 컴파일 및 실행

fork.c (~/program) - gedit

Open Save

```c
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>

int main()
{
        pid_t pid;
        int data=10;

        pid=fork();
        if(pid==-1)
                printf("fork fail, process id : %d\n", pid);

        printf("fork success, process id : %d \n", pid);

        if(pid==0)
                data +=10;
        else
                data -= 10;

        printf("data : %d \n", data);

        return 0;
}
```

C Tab Width: 8 Ln 1, Col 1 INS

프로세스의 생성

fork 함수 호출을 통한 프로세스 생성

❑ fork 프로그램 실행 예

```
sjhong@ubuntu: ~/program
^CSignal : 2
5 : wait
^CSignal : 2
sjhong@ubuntu:~/program$
sjhong@ubuntu:~/program$
sjhong@ubuntu:~/program$
sjhong@ubuntu:~/program$
sjhong@ubuntu:~/program$
sjhong@ubuntu:~/program$ gedit fork.c
sjhong@ubuntu:~/program$ gedit fork.c
^C
sjhong@ubuntu:~/program$ gcc -o fork fork.c
sjhong@ubuntu:~/program$ ./fork
fork success, process id : 23069
data : 0
sjhong@ubuntu:~/program$ fork success, process id : 0
data : 20

sjhong@ubuntu:~/program$ ./fork
fork success, process id : 23073
data : 0
sjhong@ubuntu:~/program$ fork success, process id : 0
data : 20
```

프로세스의 생성

좀비 프로세스가 생성될 수 있음

- 아래와 같이 zombie.c프로그램을 작성
- gcc –o zombie zombie.c로 컴파일

```c
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>

int main()
{
        pid_t pid;
        int data=10;

        pid=fork();
        if(pid==-1)
                printf("fork fail, process id : %d\n", pid);

        printf("fork success, process id : %d \n", pid);

        if(pid==0)
                data +=10;
        else
        {
                data -= 10;
                sleep(20);
        }

        printf("data : %d \n", data);

        return 0;
}
```

프로세스의 생성

좀비 프로세스가 생성될 수 있음

- 아래와 같이 zombie 프로그램을 실행하여 좀비 프로세스를 확인
  - ./zombie &으로 실행(&은 백그라운드로 실행을 의미)
  - ps -u를 통해 현재 좀비 프로세스가 존재하는지 확인

```
sjhong@ubuntu: ~/program
sjhong@ubuntu:~/program$ gedit zombie.c
sjhong@ubuntu:~/program$
sjhong@ubuntu:~/program$
sjhong@ubuntu:~/program$ clear

sjhong@ubuntu:~/program$ gcc -o zombie zombie.c
sjhong@ubuntu:~/program$ ./zombie &
[1] 23207
sjhong@ubuntu:~/program$ fork success, process id : 23208
fork success, process id : 0
data : 20

sjhong@ubuntu:~/program$ ps -u
USER        PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
sjhong    22223  0.0  0.2  29608  5380 pts/4    Ss   17:19   0:00 bash
sjhong    23207  0.0  0.0   4352   636 pts/4    S    18:35   0:00 ./zombie
sjhong    23208  0.0  0.0      0     0 pts/4    Z    18:35   0:00 [zombie] <defunct>
sjhong    23209  0.0  0.1  44432  3292 pts/4    R+   18:35   0:00 ps -u
sjhong@ubuntu:~/program$ data : 0
```

프로세스의 생성

좀비 프로세스 생성을 막기

❑ 아래와 같이 wait 프로그램을 작성

– Wait함수로 리턴값을 받으면 좀비가 생성되지 않음

```c
#include <stdio.h>
#include <unistd.h>
#include <sys/wait.h>
#include <sys/types.h>

int main()
{
        pid_t pid, child;
        int data=10;
        int state;

        pid=fork();
        if(pid < 0)
                printf("fork fail, process id : %d\n", pid);

        printf("fork success, process id : %d \n", pid);

        if(pid==0)
                data +=10;
        else
        {
                data -= 10;
                child = wait(&state);
                printf("child process id = %d\n", child);
                printf("Return value = %d\n", WEXITSTATUS(state));
                sleep(20);
        }

        printf("data : %d \n", data);

        return 0;
```

프로세스의 생성

좀비 프로세스 생성을 막기

❑ 아래와 같이 wait 프로그램을 실행 후 zombie프로세스 생성확인

```
sjhong@ubuntu: ~/program
sjhong    22223  0.0  0.2  29608  5380 pts/4    Ss   17:19   0:00 bash
sjhong    23207  0.0  0.0   4352   636 pts/4    S    18:35   0:00 ./zombie
sjhong    23208  0.0  0.0      0     0 pts/4    Z    18:35   0:00 [zombie] <defunct>
sjhong    23209  0.0  0.1  44432  3292 pts/4    R+   18:35   0:00 ps -u
sjhong@ubuntu:~/program$ data : 0
^C
[1]+  Done                    ./zombie
sjhong@ubuntu:~/program$
sjhong@ubuntu:~/program$
sjhong@ubuntu:~/program$ gcc -o wait wait.c
sjhong@ubuntu:~/program$ ./wait &
[1] 23304
sjhong@ubuntu:~/program$ fork success, process id : 23305
fork success, process id : 0
data : 20
child process id = 23305
Return value = 0

sjhong@ubuntu:~/program$ ps -u
USER        PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
sjhong    22223  0.0  0.2  29608  5380 pts/4    Ss   17:19   0:00 bash
sjhong    23304  0.0  0.0   4352   632 pts/4    S    18:41   0:00 ./wait
sjhong    23306  0.0  0.1  44432  3400 pts/4    R+   18:41   0:00 ps -u
sjhong@ubuntu:~/program$
```


Thank you for your attention !!