4. Homework

12201922 이규민

0) Open two Cygwin/terminal windows and type cli.c and serv.c at each window (the code is in Section 1).

[cli.c]

텍스트, 전자제품, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

[serv.c]

텍스트, 전자제품, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명 텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

1) Adjust port and IP address for both cli.c and serv.c. SERV\_ADDR in cli.c and serv.c should be the IP address of the linux server you are using. Use "ifconfig"(in ubuntu) or "ipconfig"(in cygwin) to find out the IP address of your virtual machine or PC. cli.c will use this SERV\_ADDR to access the server while serv.c will use this SERV\_ADDR to set its own IP address. Pick a port number in the range of [10000..65535]. You need two terminals: one for the server and the other for the client. Compile both and run the server first and then the client.

(\* Sometimes, you have binding failure when running the server. It happens because the server port number is blocked temporarily. Wait for 10 seconds and retry or use a different port number.)

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 전자제품, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명텍스트, 전자제품, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

cygwing에서 ipconfig 명령어를 통해 나의 ip를 확인한 후 cli.c와 serv.c의 ip를 수정해줬다. 나의 pc가 보내고, 나의 pc가 받아야하므로 두 ip를 동일하게 입력해줬다. port번호의 경우 10000 이하는 시스템의 포트이므로 10000~60000 중 임의로 숫자를 입력했다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

서버가 수신을 하고 있을 때 클라이언트가 정보를 보내야하므로 serv를 먼저 실행한 후 cli를 실행했다. cli에서 hello라는 메시지를 입력했고, serv 창을 보니 hello 메시지가 정상적으로 수신되었다. 그리고 hi라는 메시지를 입력하니 cli에서도 정상적으로 입력을 받았다.

1-1) Modify cli.c and serv.c such that they can talk in sentence (not just in word as in the current implementation).

[cli.c]

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

[serv.c]

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

위 두 코드에서 사용자의 입력을 받는 함수인 scanf를 제거하고, fgets를 이용하여 입력을 받았다. Scanf는 단어 단위로 입력을 받으므로 문장을 입력 받지 못한다. 그러나 fgets의 경우 사용자가 엔터를 누를 때까지 입력을 받으므로 문장 전체를 입력 받을 수 있다. Fget는 마지막에 개행문자까지 입력되므로 buf의 마지막 문자를 ‘\0’으로 수정을해줬다.

[결과]

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

Serv를 실행한 후 cli를 실행했다. 그리고 cli에서 hello world를 입력하니 serv에서 정상적으로 수신했다는 메시지가 나타난다. 그리고 my name is kyumin을 입력하니 cli에서 정상적으로 수신했다는 메시지가 나타난다.

2) Modify cli.c and serv.c such that they can keep talking until the client sends "bye". **Use a finite loop.**

[cli.c]

텍스트, 스크린샷, 폰트, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

[serv 결과]

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

if(strcmp(buf, "bye") == 0) break;코드를 추가하였다. Bye가 입력되면 cli는 for문을 탈출하고, x를 close 한다. 이 때 serv를 보면 화면에 공백이 출력된다.

2-1) Try to talk with the other student. Note that one of you would be the client and the other the server. SERV\_ADDR and SERV\_TCP\_PORT in cli.c should match to those in serv.c of the other student. (If you prefer to work alone, run the client in ubuntu and run the server in cygwin terminal. Make sure the client specify the IP and port number set in the cygwin server.)

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

Putty를 실행하고 ip와 port를 serv.c와 동일하게 입력해줬고, Connection type은 Telnet으로 설정 후 실행했다. 그리고 위 사진과 같이 서로 입력한 내용이 출력 되는 것을 볼 수 있다.

2-2) Modify cli.c such that it connects to Inha web server and read the web page. Inha web server domain name is www.inha.ac.kr and port number is 80. You can find the IP address for www.inha.ac.kr with ping command. To receive the web page from a web server, use GET command (your code should send below string automatically using write() function – do not type it by hand):

GET / HTTP/1.1\r\nHOST:www.inha.ac.kr\r\n\r\n

where \ is a backslash character.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명 텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

GET 명령어를 write 함수를 통해서 보내고, 받은 내용을 출력하는 과정을 10번 반복하였다. 그 결과는 위와 같다.

3) Modify the code such that the client and the server can talk in any order. **Use a finite loop to avoid infinite number of processes.**

[cli.c]

텍스트, 스크린샷, 소프트웨어, 디스플레이이(가) 표시된 사진

자동 생성된 설명

기존 코드는 클라이언트와 서버가 한 번씩 말할 수 있었다. 응답을 받아야 다시 메시지를 보낼 수 있었다. Socket의 메시지를 쓰고 읽는 과정을 유한 루프로 만들기 위해서 for문을 10번 반복했는데 for 문에 진입하기 전에 fork함수를 사용하여 프로세스를 복제했다.

그리고 for문에 진입하면 부모 프로세스는 쓰기만 반복하고, 자식 프로세스는 읽기만 반복한다.

[serv.c]

텍스트, 스크린샷, 소프트웨어, 디스플레이이(가) 표시된 사진

자동 생성된 설명

Serv.c 코드도 동일한 방법으로 수정했다. Socket의 메시지를 쓰고 읽는 과정을 유한 루프로 만들기 위해서 for문을 10번 반복했는데 for 문에 진입하기 전에 fork함수를 사용하여 프로세스를 복제했다.

그리고 for문에 진입하면 자식 프로세스는 쓰기만 반복하고, 부모 프로세스는 읽기만 반복한다.

[결과]

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

위 사진이 cli.c 코드를 실행한 화면이고, 아래 사진이 serv.c를 실행한 화면이다.

Cli를 실행했을 때 hello와 world를 입력했고, serv의 화면을 보면 정상적으로 두 단어가 출력된 것을 볼 수 있다.

serv에서도 hi와 system programming이라는 단어를 입력했고, cli 화면에서 정상적으로 두 메시지가 출력되었다.

Client와 server가 한 번씩만 번갈아 가면서 말하지 않고, 한쪽에서 두 번 이상 말할 수 있게 되었다.

**Week 12 homework**

3-1) Write a program that reads “login.txt” and answers whether a given name is in this list or not For example, if “login.txt” has

kim

hong

steve

ted

Then you program(ex31) should answer

$ ex31

enter name: ted

ted is legal user

enter name: kim2

kim2 is not legal user

………….

[ex31.c]

텍스트, 스크린샷, 소프트웨어, 폰트이(가) 표시된 사진

자동 생성된 설명

우선 사용자로부터 이름을 입력 받고, name 변수에 저장한다.

그리고 fopen 함수를 사용하여 login.txt 파일을 읽기 모드로 열었다. 파일의 내용을 한 줄씩 읽고 buf에 저장하기 위해서 fgets를 사용했다. 무한 루프인 while 내부에서 fgets를 사용하고, 읽은 내용이 없을 경우 일치하는 이름이 없다는 메시지를 출력하고 탈출한다.

여기서 scanf는 줄바꿈이 들어가지 않지만, fgets는 줄바꿈이 들어가므로 buf의 마지막 문자를 ‘\0’으로 바꿔주었다.

만약 동일한 이름이 존재한다면 메시지를 출력하고 바로 while문을 탈출한다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

왼쪽 사진은 login.txt 내용이고, 오른쪽은 이름을 검색해본 결과다.

3-2) Write a server that speaks only with registered user in "login.txt" file. The server should open login.txt file before chatting and remember the names of legal users.

cli -> serv: hello

serv -> cli: name?

cli -> serv: kim

serv -> cli: ok. now we can talk

.... cli and serv chats ....

cli -> serv: hello

serv -> cli: name?

cli -> serv: hong

serv -> cli: not legal user. disconnecting.

[serv.c]

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 폰트, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

텍스트, 폰트, 스크린샷, 그래픽이(가) 표시된 사진

자동 생성된 설명

Listen 함수를 이용해서 client와 연결이 된다면 우선 사용자 확인이 필요하다.

우선 처음에 open 함수를 사용해서 “login.txt” 파일을 열고, 그 내용을 캐릭터 name 배열에 저장한다. 이 때 name[y - 1] = '\0';라는 코드가 있는데 이는 파일의 내용을 read할 때 마지막에 개행문자가 들어있으므로 “kyumin\n”이라는 메시지가 name에 저장된다. 그러나 client로부터 입력되는 메시지에는 개행문자가 없으므로 마지막 문자를 종료로 바꿔주었다.

그리고 client로부터 socket을 받는다면, 이름을 묻는 메시지 “name?”을 보낸다.

client로부터 메시지(이름)을 받는다면 name 배열에 저장되어있던 메시지와 동일한 문자열인지 비교를 한다. 만약 다르다면 모든 socket을 닫고 연결을 종료한다. 올바른 이름이 입력되었다면 for문을 실행하여 통신을 시작한다.

[로그인에 성공한 결과]

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

Login.txt 파일에는 “kyumin”이라는 메시지가 저장되어있다. Cli에서 kyumin을 입력하니 정상적으로 통신이 되는 것을 볼 수 있다.

[로그인에 실패한 경우]

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 폰트, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

다른 이름이 입력된 경우 통신이 끊긴다. Cli의 경우 통신이 끊겼을 때 메시지를 주고 받는 for문을 빠져나오는 코드가 없어 위 사진에서는 “serv -> cli :” 메시지가 for문 반복 횟수만큼 출력된다. 이 문제는 통신이 끊겼을 때 따로 처리 방법을 만든다면 해결할 수 있다.

4) Implement simple ftp server and client. If the client doesn't follow the protocol, the server should stop the communication. This is not chatting program. **Do not type "hello", "what file do you want?", etc.** The server and client should automatically send or receive the ftp protocol messages. The user will only provide the file name to download. **Do not use the code from Problem 3, which will make the coding very hard.** Modify the code from Problem 1. Make sure your ftp server and client can handle any file size.

Simple ftp protocol

client => server: hello

server => client: what file do you want?

client => server: file name

server => client: file contents

[cli.c]

텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명 텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

간단히 구현한 코드이기 때문에 1번 문제의 코드와 거의 비슷하다. 파일 이름만 사용자로부터 입력을 받아서 서버로 전송하면 되고, 이 때 파일 이름은 띄어쓰기가 없으므로 scanf 함수를 사용해도 문제가 없다.

[serv.c]

텍스트, 전자제품, 스크린샷, 디스플레이이(가) 표시된 사진

자동 생성된 설명 텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

Serv.c 에서는 client로부터 입력받은 파일명을 오픈해서 내용을 전송해야한다. Client로부터 파일명을 전송 받으면 buf에 저장이 되고, open함수를 사용해서 buf에 저장된 문자열의 파일 이름을 오픈한다. 이 때 파일 내용은 수정이 필요없으므로 O\_RDONLY 옵션을 사용하여 읽기모드로 열었고, 파일이 없다면 빈 내용을 buf에 저장하기 위해서 O\_CREAT 옵션을 사용했다.

그리고 파일의 모든 내용을 읽기 위해서 무한 for문을 만들고, read 함수를 사용해서 파일 내용을 읽는다. 읽은 내용은 client로 바로 전송한다. 만약 파일의 모든 내용을 다 읽고, 읽은 내용이 없다면 break 함수를 사용해서 for문을 탈출한다.

그러나 cli.c 코드에서는 문제가 있다. 파일의 내용이 길다면 serv.c의 for문이 여러 번 돌면 write를 여러 번 한다. 그러나 cli.c에서 파일의 내용을 전송받아서 읽는 read 코드는 한 번밖에 없으므로 코드가 종료될 것이다. 이 문제를 해결하기 위해선 cli.c의 read 함수를 여러 번 반복할 수 있도록 for문 안에서 read를 하면 된다.

4-1) Modify your ftp client so that the client saves the file content under the same file name. For this purpose, run the client in a different directory if you run the server and client in the same machine.

[cli.c]

텍스트, 스크린샷, 소프트웨어, 디스플레이이(가) 표시된 사진

자동 생성된 설명

이전 문제에서 파일의 크기가 크면 메시지를 모두 받지 못한다는 문제가 있었지만 for문을 추가하여 해결하였다. 이 때 for문은 무한 루프로 만들어도 되지만 안전을 위해 10번에 제한을 두었다.

동일한 파일명을 생성하기 위해서 파일 디스크립터의 변수 이름을 fd라고 지정하였다. 그리고 내가 입력했던 파일명을 오픈하기 위해서 open 함수를 사용했고, 옵션은 O\_RDWR | O\_CREAT | O\_TRUNC을 사용했다.

For문 안에서는 server로부터 입력받은 내용을 for문을 돌면서 fd에 저장을한다. 만약 서버에서 모든 내용을 다 보내면 client도 읽을 내용이 없으므로 y는 0이 되면서 for문을 탈출할 수 있게 된다.

[serv.c]

텍스트, 스크린샷, 소프트웨어, 폰트이(가) 표시된 사진

자동 생성된 설명

이전 문제와 비슷한 코드이지만 이전 문제에서 오류가 있었다. Write함수 사용 시 보낼 문자의 길이를 buf에 크기로 지정했었다. 그러나 파일의 마지막 부분을 읽을 때 즉 y가 50보다 작을 때 보내지 말아야하는 메시지가 보내지는 문제가 있다. 그래서 길이를 y로 수정해줬다. 이외에는 모두 동일한 내용이다.

[실행 결과]

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

이 문제는 server에 있는 파일을 client가 동일한 파일명과 동일한 내용을 가진 파일을 생성하는 것이다. 그러나 나는 동일한 pc로 진행하는데 원본 파일이 훼손되는 문제가 있을 수 있으므로 d1이라는 디렉토리 안에서 ftpcli를 실행해줬다.

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

왼쪽 사진은 원본 파일이고, 오른쪽 사진은 복제한 파일이다. 파일 내용이 훼손되지 않고 정상적으로 client에 전달되었음을 볼 수 있다.

5) Modify your ftp server such that it can handle multiple clients at the same time. Make sure the client can download an executable file also. Run the client in another directory so that you don’t overwrite an existing file and for an executable file, confirm that the downloaded executable runs ok.

텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명 텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명 텍스트, 폰트, 스크린샷, 그래픽이(가) 표시된 사진

자동 생성된 설명 텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

5-1) Modify your ftp server in Problem 5 such that it can handle three commands: ls, get, put. For “ls” command, the server will show the file list in the current directory. For “get f1”, the server will send file f1. For “put f1” command, the server will store the file content sent by the client under the same file name f1. The server keeps handling a client’s command until it says “bye”.

cli 🡺 server: hello

server 🡺 cli: ok

cli 🡺 serv: ls

serv 🡺 cli: (the file list)

cli 🡺 serv: get ex1.exe

server 🡺 cli: the content of ex1.exe

cli 🡺 serv: put f1

server 🡺 cli: send f1

cli 🡺 server: the content of f1 (the server will store them under file name f1)

……

cli 🡺 serv: bye

serv 🡺 cli: bye

6) Write a client in your PC as follows and let it talk to the server program in the lab server. To compile the client program:

- Make an empty c++ project and copy the code given below.

- Adjust the server IP and port number

- Select

“project->project property->manifest tools->input and output->include manifest”

and set “No”

- add ws2\_32.lib in project>project property>link>input>additional dependencies

- Select build->Solution Build

- You should see “Success 1” at the bottom of the compile window.

- Sometimes, the platform of the program (the one shown in the main interface screen:release or debug) is different from the platform in properties configuration. They should match.)

- If you have an error related with "pre-compiled header", go to

“project->project property->c/c++>pre-compiled header > pre-compiled header>”

and set “No”

(\* If you use MAC, download cli.c from the lab server and use it as your local client program. You may have to include "unistd.h" to avoid compile errors.)

#define \_CRT\_SECURE\_NO\_WARNINGS

#include "winsock2.h"

#include "ws2tcpip.h"

#include "stdio.h"

#define SERVER\_PORT 9924 // server port number

#define BUF\_SIZE 4096 // block transfer size

#define QUEUE\_SIZE 10

#define IPAddress "165.246.38.152" // server IP address

int main()

{

WORD wVersionRequested;

WSADATA wsaData;

SOCKADDR\_IN target; //Socket address information

SOCKET s;

int err;

int bytesSent;

char buf[100];

//--- INITIALIZATION -----------------------------------

wVersionRequested = MAKEWORD( 1, 1 );

err = WSAStartup( wVersionRequested, &wsaData );

if ( err != 0 ) {

printf("WSAStartup error %ld", WSAGetLastError() );

WSACleanup();

return false;

}

//------------------------------------------------------

//---- Build address structure to bind to socket.--------

target.sin\_family = AF\_INET; // address family Internet

target.sin\_port = htons (SERVER\_PORT); //Port to connect on

inet\_pton(AF\_INET, IPAddress, &(target.sin\_addr.s\_addr)); // target IP

//--------------------------------------------------------

// ---- create SOCKET--------------------------------------

s = socket (AF\_INET, SOCK\_STREAM, IPPROTO\_TCP); //Create socket

if (s == INVALID\_SOCKET)

{

printf("socket error %ld" , WSAGetLastError() );

WSACleanup();

return false; //Couldn't create the socket

}

//---------------------------------------------------------

//---- try CONNECT -----------------------------------------

if (connect(s, (SOCKADDR \*)&target, sizeof(target)) == SOCKET\_ERROR)

{

printf("connect error %ld", WSAGetLastError() );

WSACleanup();

return false; //Couldn't connect

}

//-------------------------------------------------------

//---- SEND bytes -------------------------------------------

printf("enter a string to send to server\n");

gets\_s(buf, 99);

bytesSent = send( s, buf, strlen(buf), 0 ); // use "send" in windows

printf( "Bytes Sent: %ld \n", bytesSent );

// now receive

int n;

n=recv(s, buf, 50, 0); // read max 50 bytes

buf[n]=0; // make a string

printf("received: %s\n", buf);

//--------------------------------------------------------

closesocket( s );

WSACleanup();

return 0;

}

7) Write an ftp client in your PC and let it talk to the ftp server you made in problem 5). Use this client to download a file from the lab server and save under the same file name.