시스템 프로그래밍 실습

[Assignmen3-1]

Class : D 반(실습 2 금 56)

Professor : 최상호 교수님

Student ID : 2022202104

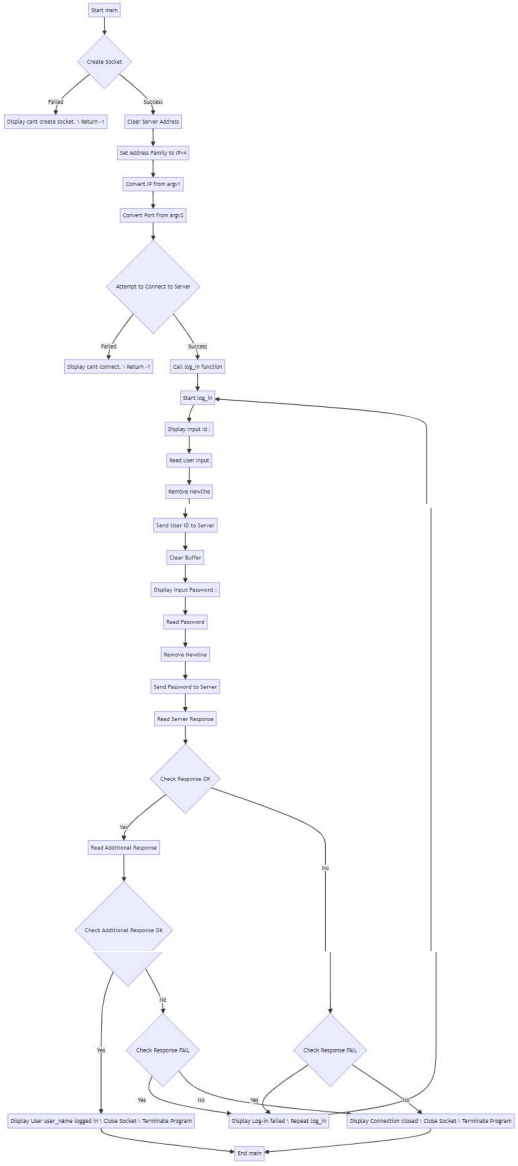
Name : 김유찬

Introduction

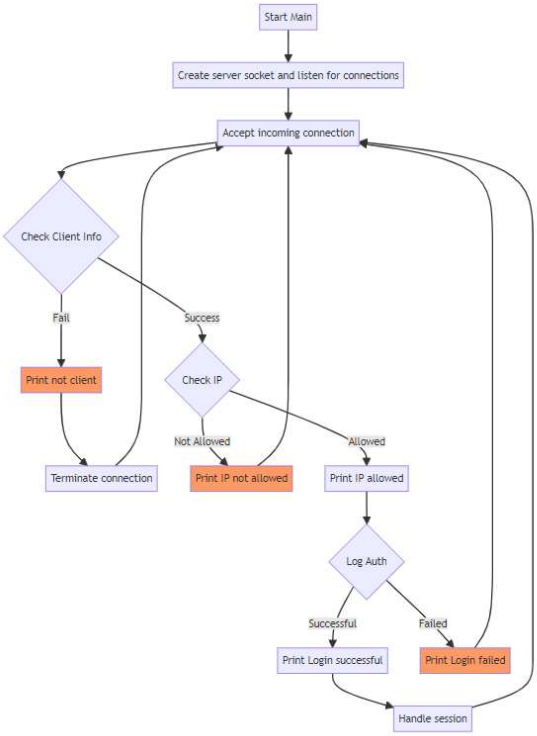
지금까지 어떤 client 의 IP 라고 해도 받을 수 있게 server 에 접속하게 만들었다. 원하는 IP 만 허용되게 만들고 싶을 수 있다. 또한 client 쪽에서 id 와 password 을 입력해서 server 에 있는 passwd 에 저장된 아이디와 비번과 비교해서 있으면 로그인 되고 안 되면 다시 client 한테 id 와 password 을 입력을 받을 수 있게 한다. 즉 client 와 server 간의 Authorization 을 만들어 보는 시간을 갖을 것이다.

Flow chart

1) client



2) server



Pseudo code

1) client

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#define MAX_BUF 1024
#define CONT_PORT 20001

void log_in(int sockfd) {
    repeat {
        clear buffer

        display "Input Id : "
        read user input into buffer
        remove newline character from buffer if present
        send buffer to server

        clear buffer

        display "Input Password : "
        read password into buffer
        remove newline character from buffer if present
        send buffer to server

        read server response into buffer
        if (buffer equals "OK") {
            read additional server response into buffer
            if (buffer equals "OK") {
                display "** User '[user_name]' logged in **"
                close socket
                terminate program
            } else if (buffer equals "FAIL") {
                display "** Log-in failed **"
            } else {
```

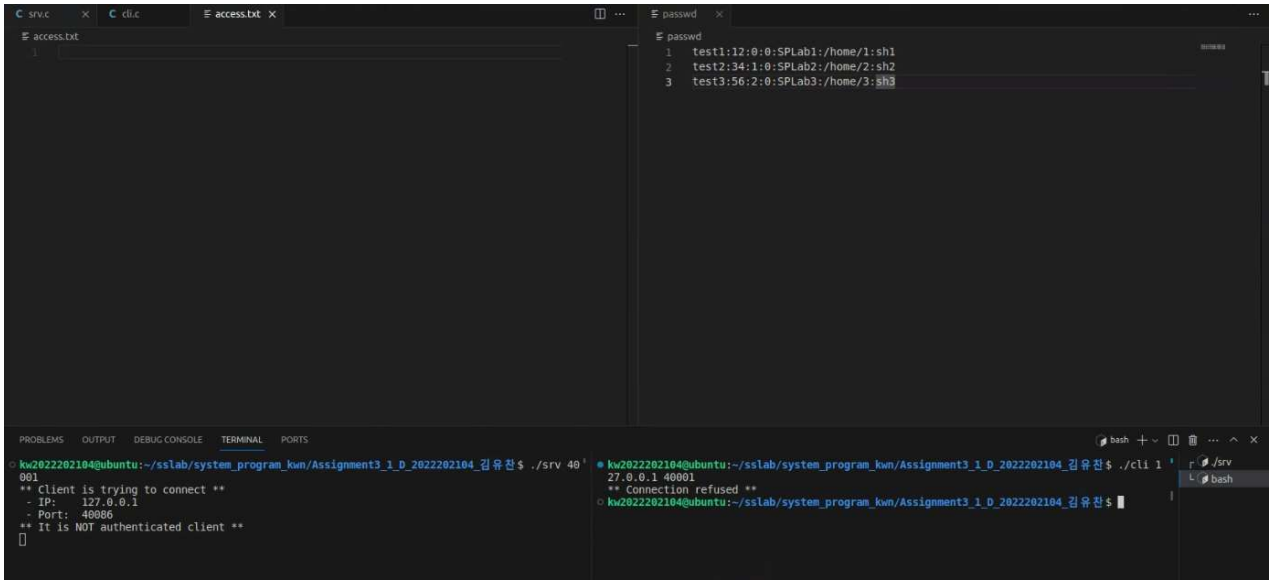
```
                display "** Connection closed **"
                close socket
                terminate program
            }
        }
    } until false
}

int main() {
    sockfd = create socket(ipv4, stream)
    if (sockfd < 0) {
        display "can't create socket."
        return -1
    }

    clear servaddr

    servaddr.family = ipv4
    servaddr.ip = convert ip argv[1]
    servaddr.port = convert port argv[2]

    if (connect socket to server fails) {
        display "can't connect."
        return -1
    }

    call log_in(sockfd)

    close sockfd
    return 0
}
```

2) server

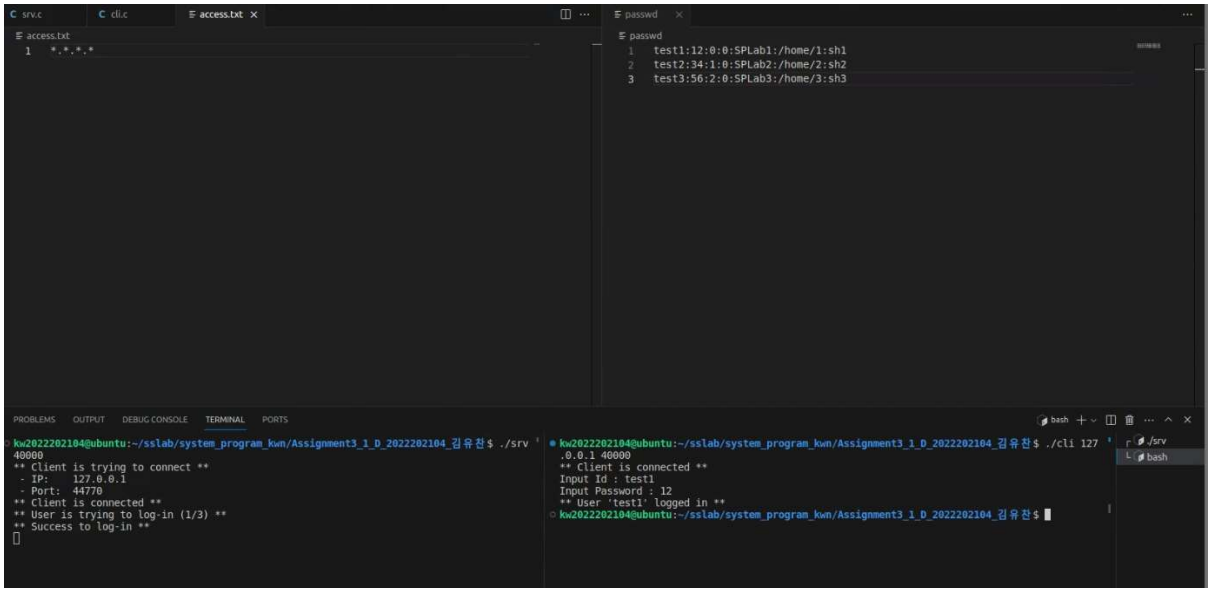
```
#include <stdio.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <stdlib.h>
#include <dirent.h>
#include <pwd.h>
#include <grp.h>
#include <unistd.h>
#include <sys/stat.h>
#include <time.h>

int main() {
    create server socket and listen for connections
    for each incoming connection {
        if (client_info fails) {
            print "not client"
            terminate connection
        }

        if (not allowed IP) {
            print "IP not allowed"
            continue to next iteration
        } else {
            print "IP allowed"
        }

        if (log_auth successful) {
            print "Login successful"
        } else {
            print "Login failed"
            continue to next iteration
```

```
        }
    }
}

int client_info(pointer to client address) {
    print "client trying to connect"
    if (family is not IPv4) {
        return failure
    }
    print IP and port
    return success
}

int user_match(username, password) {
    open and read password file
    if (file not open) {
        return failure
    }
    for each entry in password file {
        if (username and password match) {
            return success
        }
    }
    return failure
}

int log_auth(connection file descriptor) {
    for up to 3 attempts {
        read username and password from connection
        if (user_match is successful) {
            send "OK"
            return success
        } else {
            send "FAIL"
            if (3 attempts exceeded) {
                send "FAILFAIL"
                return failure
            }
        }
    }
    return success
}
```

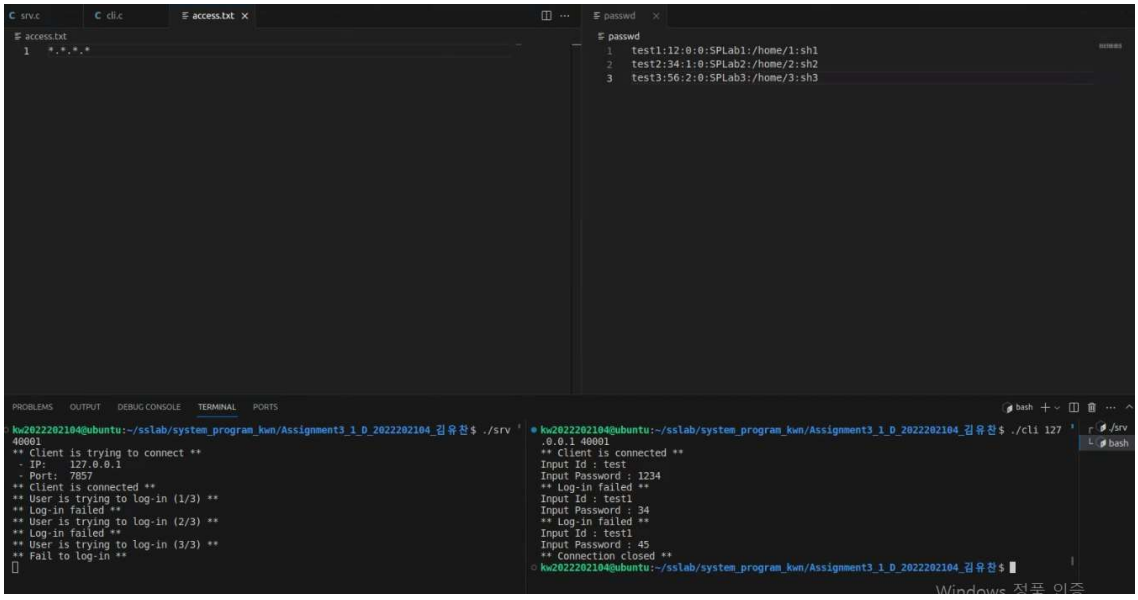
결과화면



- access.txt 에 어떤 ip 도 허용되지 않기 때문에 client 가 server 에 접근하지 못하고 있다.



- access.txt 에 어떤 ip 도 허용하고 passwd 에 있는 회원정보와 일치한 아이디, 비밀번호를 적었기 때문에 로그인 성공 출력이 나왔다.



- access.txt 에 어떤 ip 도 허용했지만 passwd 에 있는 회원정보와 일치하지 않는 아이디, 비밀번호를 3 번이나 입력해서 client 쪽에서 server 와 끊겼다.

고찰

Client 쪽에서 server 에 보낼 때 쓰는 buffer 을 항상 초기화 해야 한다는 것을 다시 한번 깨닫게 되었다. 허용된 IP 주소만 server 에 허용하게 만들려고 했을 때 와일드 카드 구현에서 시간이 걸렸다. 하지만 strtok 를 적절히 잘 이용하면 쉽게 해결하는 있는

부분이었다. 이번 시간을 통해 리눅스 passwd 파일을 어떻게 c 언어로 읽어서 Authorization 이 되는 알 수 있는 시간이 되었다.

Reference

시스템프로그래밍 실습 강의자료