

گزارش پروژه اول درس شبکه‌های کامپیوتری (FTP Server)

کیمیا محمدطاهری (۸۱۰۱۹۸۵۳۵)

دانشور امراللهی (۸۱۰۱۹۷۶۸۵)

• امراز هویت:

○ دریافت نام کاربری:

```
int CommandHandler::handleUser(std::string username)
{
    if (logged_in)
        throw Exception();

    bool found = false;
    for (User* user: users) {
        if (user->username == username)
        {
            found = true;
            logging_in_user = user;
            logging_in = true;
            is_admin = user->admin;
        }
    }

    if (!found)
        throw InvalidUsernameOrPassword();
    return USERNAME_FOUND;
}
```

تابع `handleUser` از کلاس `CommandHandler` مسئولیت دریافت نام کاربری را دارد. اگر کاربری با `username` داده شده پیدا شود، آن کاربر را برای دستور بعدی (`pass`) در متغیر `logging_in_user` ذخیره می‌کند. در صورت پیدا نشدن کاربر یا `logged in` نبودن کاربری، خطاهای متناسب را `raise` می‌کند. در صورت موفقیت‌آمیز بودن پیدا کردن کاربر، کد مورد نظر (`USERNAME_FOUND`) که همان ۳۳۱ است را برمی‌گرداند تا پیغام متناسب با آن در کنسول نوشته شود. متغیر `logging_in` نیز به `true` مقداردهی می‌شود که هنگام دریافت دستور `pass` اطمینان پیدا کنیم دنباله دستورات مورد نظر درست است.

○ دریافت رمز کاربر:

```
int CommandHandler::handlePass(std::string password) {
    if (!logging_in)
        throw BadSequenceOfCommands();

    if (logging_in_user->password == password)
    {
        logging_in = false;
        logged_in = true;
        current_user = logging_in_user;
        if (logging_in_user->admin)
            is_admin = true;
        return USER_LOGGED_IN;
    }
    throw InvalidUsernameOrPassword();
}
```

تابع `handlePass` رمز دریافت شده را، با رمزی که انتظار می‌رفته دریافت کند (از مرحله قبل کاربر متناظر در `logging_in_user` ذخیره شده) تطبیق می‌دهد و در صورت موفقیت‌آمیز بودن، کد `USER_LOGGED_IN` که ۲۳۰ می‌باشد را برای چاپ پیغام مناسب باز می‌گرداند. همچنین متغیر بولین `logged_in` نیز `true` می‌شود تا برای دستورات بعدی بدانیم کاربر `logged_in` شده است. در صورت درست نبودن رمز عبور، خطای `InvalidUserNameOrPassword` را `raise` می‌کند.

```
daneshvar@daneshvar-ZenBook:~/Desktop/CN/FTP-Server/client$ ./Client.out
user abcde
430: Invalid username or password
user Ali
331: User name okay, need password.
pass 12345
430: Invalid username or password
pass 1234
230: User logged in, proceed. Logged out if appropriate.
```

شکل(۱): نمونه ورودی/خروجی برای امراز هویت

● دستورات:

برای اجرای کامندهای دریافت شده در ترمینال و دریافت فرومی آنها از تابع `execShellCommand` استفاده می‌کنیم که منبع آن لینک زیر می‌باشد:

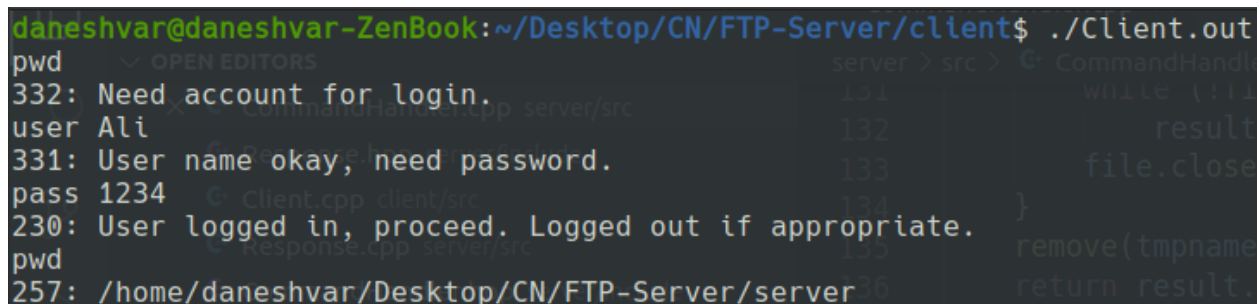
<https://stackoverflow.com/q/478898/3250120>

به ازای هر دستور ورودی کاربر، آن را به دستور استاندارد `linux shell script` کرده و آن را به تابع `execShellCommand` برای اجرا پاس می‌دهیم.

○ دایرکتوری فعلی (`pwd`):

دایرکتوری فعلی برنامه در مشخصه `current_directory` نگه داشته شده است که با انجام دستور `cwd` آپدیت می‌شود.

```
std::string CommandHandler::handlePwd(std::vector<std::string> args) {
//args will be empty
    if (!logged_in)
        throw NotLoggedIn();
    try {
        return current_directory;
    } catch (...) {
        throw Exception();
    }
}
```



```
daneshvar@daneshvar-ZenBook:~/Desktop/CN/FTP-Server/client$ ./Client.out
pwd
332: Need account for login.
user Ali
331: User name okay, need password.
pass 1234
230: User logged in, proceed. Logged out if appropriate.
pwd
257: /home/daneshvar/Desktop/CN/FTP-Server/server
```

شکل(۲): نمونه اجرای دستور `pwd`

○ سافتن دایرکتوری جدید (mkd)

```
std::string CommandHandler::handleMkd(std::vector<std::string> args) {
    if (!logged_in)
        throw NotLoggedIn();
    try {
        execShellCommand(("cd " + current_directory + " && mkdir").c_str(),
args);
    } catch (...) {
        throw Exception();
    }
    return args[0];
}
```

معادل دستور mkd در linux shell script می‌باشد که یک آرگومان دریافت می‌کند که نام دایرکتوری جدید است. همچنین نام دایرکتوری جدید برای چاپ پیغام متناسب بازگردانده می‌شود.

```
daneshvar@daneshvar-ZenBook:~/Desktop/CN/FTP-Server/client$ ./Client.out
mkd hello
332: Need account for login.
user Ali
331: User name okay; need password.
pass 1234
230: User logged in, proceed. Logged out if appropriate.
mkd hello
257: hello created.
```

شکل (۳): نمونه اجرای دستور mkd

○ پاک کردن فایل یا دایرکتوری

برای پاک کردن فایل در shell لینوکس کافی است از rm و برای پاک کردن دایرکتوری کافی

است از rm -r استفاده کنیم.

```
std::string CommandHandler::handleDele(std::vector<std::string> args) {
    if (!logged_in)
        throw NotLoggedIn();
    std::string file_name = args[1];
    if (isPrivateFile(file_name) && !current_user->admin)
        throw FileUnavailable();
    if (args.size() < 2)
        throw SyntaxErrorInParamsOrArgs();

    if (args[0] == "-f")
    {
        args.erase(args.begin()); //remove -f/-d
        try {
            execShellCommand(("cd " + current_directory + " &&
rm").c_str(), args);
        }
        catch (...){
            throw Exception();
        }
    }

    if (args[0] == "-d")
    {
        args.erase(args.begin()); //remove -f/-d
        try {
            execShellCommand(("cd " + current_directory + " && rm -
r").c_str(), args);
        }
        catch (...){
            throw Exception();
        }
    }

    return args[0];
}
```

```
throw SyntaxErrorInParamsOrArgs();  
}
```

بنابراین اگر دستور `delete -f filename` را دریافت کنیم، کافی است دستور `rm filename` را اجرا کنیم و اگر دستور `delete -d` `dirname` را دریافت کنیم، کافی است دستور `rm -r dirname` را اجرا کنیم که با صدا زدن تابع `execShellCommand` پیش‌تر توضیح داده شد انجام می‌شوند. به عنوان آرگومان این دستورات نیز کافی است نام فایل/دایرکتوری را پاس دهیم. همچنین در حالات خاصی که لاگین نشده باشد، فایل محافظت شده باشد (تنها توسط ادمین قابل دسترسی است)، نام فایل پاس داده نشده باشد خطاهای متناسب را `raise` می‌کند.

```
daneshvar@daneshvar-ZenBook:~/Desktop/CN/FTP-Server/client$ ./Client.out  
user Mohsen  
331: User name okay, need password.  
pass 1234  
230: User logged in, proceed. Logged out if appropriate.  
delete -f config.json  
550: File unavailable.  
delete -f not_private.txt  
250: not_private.txt deleted.
```

شکل (۱۴) - نمونه اجرای دستور **delete -f** در این حالت Mohsen کاربر غیر admin است. بنابراین اجازه پاک کردن فایل `config.json` که از فایل‌های محافظت‌شده توسط admin هست را ندارد. اما فایل `not_private.txt` که یک فایل معمولی است را می‌تواند پاک کند.

```

daneshvar@daneshvar-ZenBook:~/Desktop/CN/FTP-Server/client$ ./Client.out
user Ali
331: User name okay, need password.
pass 1234
230: User logged in, proceed. Logged out if appropriate.
ls
226: List transfer done.
build
config.json
hello
include
log.txt
Makefile
Server.out
src
test_dir
test.json
dele -d test_dir
250: test_dir deleted.
ls
226: List transfer done.
build
config.json
hello
include
log.txt
Makefile
Server.out
src
test.json

```

شکل (۵) - نمونه اجرای دستور **dele -d**: با اجرای دستور **dele -d test_dir**، دایرکتوری **test_dir** پاک شده که درستی آن را با مقایسه فرومی‌های دستور قبل و پس از اجرای این دستور می‌توان سنجید.

○ لیست فایل‌های موجود در دایرکتوری

```
std::string CommandHandler::handleLs(std::vector<std::string> args) {  
    if (!logged_in)  
        throw NotLoggedIn();  
    try {  
        return execShellCommand(("cd " + current_directory + " && ls  
").c_str(), args);  
    } catch(...) {  
        throw Exception();  
    }  
}
```

انجام این دستور سراسر است و نکته فاصی ندارد.

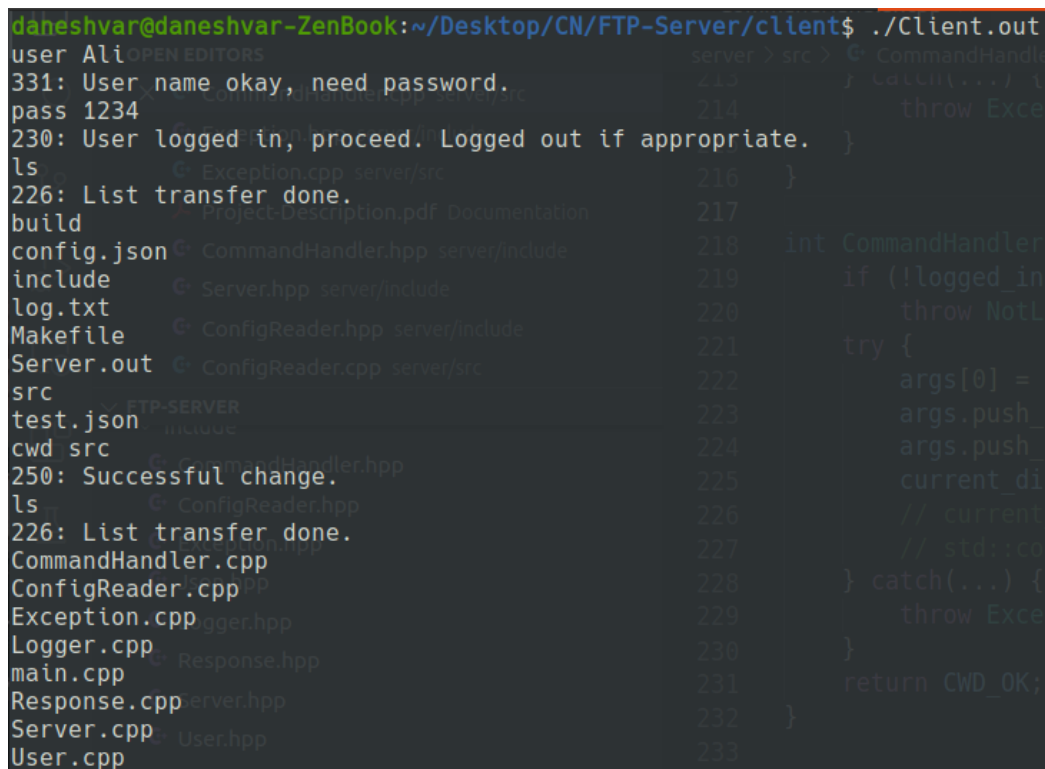
برای مشاهده خروجی نمونه آن می‌توانید به خروجی نمونه دستور `ls -f` مراجعه کنید که در آن از دستور `ls` استفاده شده بود.

○ عوض کردن دایرکتوری (cwd)

ابتدا دستور cd را صدا می‌کنیم و نهایتاً با دستور pwd، مقدار مشخصه current_directory را آپدیت می‌کنیم.

```
int CommandHandler::handleCwd(std::vector<std::string> args) {
    if (!logged_in)
        throw NotLoggedIn();
    try {
        args[0] = current_directory + "/" + args[0];
        args.push_back("&&");
        args.push_back("pwd");
        current_directory = execShellCommand("cd", args);

    } catch(...) {
        throw Exception();
    }
    return CWD_OK;
}
```



```
daneshvar@daneshvar-ZenBook: ~/Desktop/CN/FTP-Server/clients$ ./Client.out
user Ali
331: User name okay, need password.
pass 1234
230: User logged in, proceed. Logged out if appropriate.
ls
226: List transfer done.
build
config.json
include
log.txt
Makefile
Server.out
src
test.json
cwd src
250: Successful change.
ls
226: List transfer done.
CommandHandler.cpp
ConfigReader.cpp
Exception.cpp
Logger.cpp
main.cpp
Response.cpp
Server.cpp
User.cpp
```

شکل (۶) - نمونه اجرای دستور **cwd**: با اجرای src cwd به فولدر SRC در دایرکتوری فعلی می‌رود.

○ عوض کردن نام فایل

معادل دستور rename from to در linux shell script همان mv from to است. بنابراین پیاده‌سازی این دستور نیز سراسر است:

```
int CommandHandler::handleRename(std::vector<std::string> args) {  
    if (!logged_in)  
        throw NotLoggedIn();  
    try {  
        execShellCommand("mv", args);  
    } catch(...) {  
        throw Exception();  
    }  
    return RENAME_OK;  
}
```

در صورت اجرای موفقیت‌آمیز، مقدار ثابت RENAME_OK یا همان ۲۵۰ برگردانده می‌شود تا پیغام متناسب با آن در کنسول چاپ شود.

```

daneshvar@daneshvar-ZenBook:~/Desktop/CN/FTP-Server/client$ ./Client.out
user Ali
331: User name okay, need password.
pass 1234
230: User logged in, proceed. Logged out if appropriate.
ls
226: List transfer done.
build
config.json
hello
include
log.txt
Makefile
old_name.txt
Server.out
src
test.json
rename old_name.txt new_name.txt
250: Successful change
ls
226: List transfer done.
build
config.json
hello
include
log.txt
Makefile
new_name.txt
Server.out
src
test.json

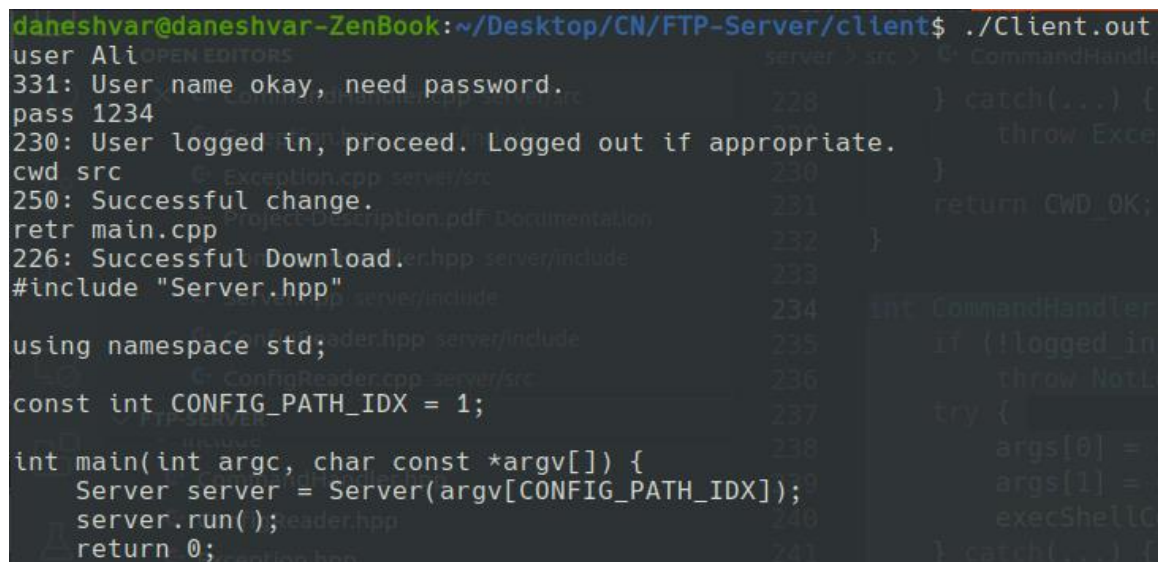
```

شکل (۷): نمونه اجرای دستور **rename** نام فایل `old_name.txt` به `new_name.txt` تغییر یافته است.

○ دانلود فایل

برای دریافت محتوای یک فایل از دستور `cat` استفاده می‌کنیم. همچنین اگر حجم فایل بیشتر از حجم باقی‌مانده برای کاربری که لاگین شده باشد، فضای متناسب با آن `raise` می‌شود.

```
std::string CommandHandler::handleRetr(std::vector<std::string> args) {
    if (!logged_in)
        throw NotLoggedIn();
    try {
        std::string file_name = args[0];
        if (isPrivateFile(file_name) && !current_user->admin)
            throw FileUnavailable();
        int file_size = getFileSize(file_name);
        if (file_size > current_user->download_capacity)
            throw NotEnoughDownloadCapacity();
        current_user->download_capacity = current_user->download_capacity -
file_size;
        return execShellCommand(("cd " + current_directory + " && cat
").c_str(), args);
    } catch(...) {
        throw Exception();
    }
}
```



```
daneshvar@daneshvar-ZenBook:~/Desktop/CN/FTP-Server/client$ ./Client.out
user Ali
331: User name okay, need password.
pass 1234
230: User logged in, proceed. Logged out if appropriate.
cwd src
250: Successful change.
retr main.cpp
226: Successful Download.
#include "Server.hpp"

using namespace std;
const int CONFIG_PATH_IDX = 1;

int main(int argc, char const *argv[]) {
    Server server = Server(argv[CONFIG_PATH_IDX]);
    server.run();
    return 0;
}
```

شکل (۸) - نمونه اجرای دستور `retr`: کاربر متقاضی دانلود فایل `main.cpp` است و محتوای آن را در کانال داده دریافت می‌کند.

○ راهنما

```

daneshvar@daneshvar-ZenBook:~/Desktop/CN/FTP-Server/client$ ./Client.out
help
USER [name], Its argument is used to specify the users string. It is used for users authentication
PASS [password], Its argument is used to specify the users string. It is used for users authentication and must be used after USER.
PWD, returns current directory
MKD [directory path], it is used to create a new directory in directory path
DELE -f [filename] it is used to delete a file
DELE -d, it is used to delete a directory
LS, it is used to get the list of the contents of the current directory
CWD [path], it is used to move from the current direcotry to the directory in input path
RENAME [from] [to],changes the name of the folder. First argument indicates the current name of the file, second argument indicates the new name.
RETR [name], used to download the file with the name given as argument.
HELP, used to list the commands and their usage
QUIT, used to quit the server

```

شکل(۹): راهنما

○ خارج شدن از سرور

```

daneshvar@daneshvar-ZenBook:~/Desktop/CN/FTP-Server/client$ ./Client.out
user Ali
331: User name okay, need password.
pass 1234
230: User logged in, proceed. Logged out if appropriate.
quit
221: Successful Quit.
user Mohsen
331: User name okay, need password.
pass 1234
230: User logged in, proceed. Logged out if appropriate.

```

شکل(۱۰): نمونه اجرای دستور quit