

# BÁO CÁO ĐỒ ÁN



**Môn học: Mạng máy tính**  
**Đồ án: FTP Client**

*Thành viên:*

*Trần Thanh Tịnh – 1612704*

*Lâm Anh Toàn – 1612712*

*Nguyễn Minh Triết - 1612736*

**GVHD: Lê Giang Thanh**

## A. Phân công:

Họ tên	MSSV	Công việc
Trần Thanh Tịnh	1612704	- cd - lcd - delete - mdelete - mkdir - active
Lâm Anh Toàn	1612712	- Login - ls/dir - get - put - mget - mput
Nguyễn Minh Triết	1612736	- rmdir - pwd - quit - passive

## B. Cấu trúc dữ liệu và chức năng hàm:

### I. Cấu trúc dữ liệu:

Bài làm sử dụng struct `FtpClientContext`, trong đó bao gồm:

- `CSocket` để điều khiển kết nối, gửi lệnh cho server. (tạm gọi là Control)
- `SOCKET` dùng cho việc truyền dữ liệu. (Data)
- Biến bool để xác định server đang sử dụng cơ chế gì, passive hay active. (P)
- Buffer dùng để nhận tin từ server/lệnh gửi server/dữ liệu gửi server/...

Trong struct không có lưu trữ IP server, đó là do tại đây, chúng ta luôn sử dụng localhost, có IP mặc định là 127.0.0.1.

### II. Chức năng hàm:

#### 1. Đăng nhập (Login)

Sau khi kết nối thành công đến server, để có thể sử dụng các dịch vụ của server thì người dùng cần phải đăng nhập.

```
bool Login(FtpClientContext* cInt);
```

- + Đầu vào là struct `FtpClientContext`.
- + Đầu ra: Hàm sẽ trả về 1 nếu đăng nhập thành công, trả về 0 nếu đăng nhập thất bại (lỗi server/sai mật khẩu/...).
- + Chức năng: đăng nhập vào server

Trong thân hàm, lần lượt yêu cầu người dùng nhập tài khoản và mật khẩu rồi gửi lên server với các lệnh `USER <tài khoản>` và `PASS <mật khẩu>` bằng Control. Tất nhiên, chúng ta không được cho hiển thị mật khẩu, che nó đi bằng cách lưu ngay kí tự vừa nhập vào biến phụ, sau đó tùy vào kí tự đó mà ta dùng `push_back`, `pop_back` lên string chứa mật khẩu hoặc kết thúc nhập. Cuối cùng là kiểm tra xem server có báo lỗi gì không, nếu có: trả về 0, nếu không: trả về 1.

Về UTF8, chế độ UTF8 luôn luôn bật trên server, nên gửi cũng không sao, không gửi cũng không sao. Hình ảnh minh họa đăng nhập thành công:

```
C:\WINDOWS\system32\cmd.exe
Successfully connected to server
220-Unknown 33.32768 final
220-You're on the verge of a new truth
220 I hope these will assist you on your journey
Username: User1
331 Password required for user1
Password:
230 Logged on
ftp> _
```

```
(000309)5/29/2018 15:17:51 PM - (not logged in) (127.0.0.1)> Connected on port 21, sending welcome message..
(000309)5/29/2018 15:17:51 PM - (not logged in) (127.0.0.1)> 220-Unknown 33.32768 final
(000309)5/29/2018 15:17:51 PM - (not logged in) (127.0.0.1)> 220-You're on the verge of a new truth
(000309)5/29/2018 15:17:51 PM - (not logged in) (127.0.0.1)> 220 I hope these will assist you on your journey
(000309)5/29/2018 15:17:56 PM - (not logged in) (127.0.0.1)> USER User1
(000309)5/29/2018 15:17:56 PM - (not logged in) (127.0.0.1)> 331 Password required for user1
(000309)5/29/2018 15:17:57 PM - (not logged in) (127.0.0.1)> PASS
(000309)5/29/2018 15:17:57 PM - user1 (127.0.0.1)> 230 Logged on
```

## 2. Passive

`bool Passive(CSocket& cInt, SOCKET& data);`

- + Đầu vào: CSocket control, SOCKET data.
- + Đầu ra: 0: kết nối thất bại, 1: kết nối thành công.
- + Chức năng: kết nối passive

Dùng control gửi lệnh PASV (sử dụng IPv4) cho server, server sẽ gửi lại gói tin có dạng như sau: 227 Entering Passive Mode (h1, h2, h3, h4, p1, p2). Trong đó h1.h2.h3.h4 là địa chỉ IP server, ta lấy  $p1 * 256 + p2$  được port mà server đề nghị kết nối kênh truyền dữ liệu đến. Tạo SOCKET data với port vừa có được, kết nối đến server, thành công thì trả về 1, không thì trả về 0.

## 3. Active

### a) Active

`int Active(FtpClientContext* cInt, int port);`

- + Đầu vào: FtpClientContext, port client hiện tại (được random tại bên ngoài).
- + Đầu ra: mã lệnh server trả lời.
- + Chức năng: mở kết nối active (gửi port cho server kết nối đến).

Gửi lệnh PORT h1, h2, h3, h4, p1, p2 cho server, đề nghị server kết nối kênh truyền dữ liệu đến port  $p1 * 256 + p2$  này. Trong đó, h1, h2, h3, h4 là IP server. Để có được p1, ta lấy tham số port thực hiện phép & với 0xFF, còn p2 thì dịch phải 8 bit rồi & với 0xFF.

### b) Create listen

`SOCKET CreateListen(int port);`

- + Đầu vào: port client hiện tại.
- + Đầu ra: NULL: tạo thất bại, SOCKET: tạo thành công.

- + Chức năng: Tạo socket listen cho cơ chế active.

#### 4. Liệt kê danh sách thư mục, tập tin trên server (dir/ls)

##### a) List

```
string List(FtpClientContext* cInt, string type, string loc, int& port);
```

- + Đầu vào: struct FtpClientContext, kiểu liệt kê (ls/dir), đường dẫn thư mục, port client hiện tại.
- + Đầu ra: chuỗi string chứa tất cả thông tin liệt kê nhận được: chuỗi rỗng nếu thất bại (không kết nối được, không nhận được, ...).
- + Chức năng: Liệt kê danh sách thư mục, tập tin trên server.

Trong thân hàm:

- Nếu P = 0 (dùng active), chúng ta gọi hàm Active để gửi lệnh PORT lên server, sau đó tạo listen socket bằng hàm hỗ trợ CreateListen. Tại đây, chúng tôi dùng Data là listen socket, bạn có thể tạo SOCKET khác để lắng nghe nếu muốn. Nếu SOCKET đó hợp lệ, gọi hàm SendLIST\_NLSTCommand, không thì bỏ qua. Lưu ý là client và server vẫn chưa kết nối kênh truyền dữ liệu, xem hàm dưới sẽ thấy.
- Nếu P = 1 (dùng passive), chúng ta gọi hàm Passive để gửi lệnh PASV/EPSV lên server. Nếu kết nối thành công thì gọi SendLIST\_NLSTCommand, không thì bỏ qua.

Tăng giá trị port lên 1 để về sau không bị trùng, đóng các socket khi không còn sử dụng chúng. Trả về string từ hàm con đã gọi. Sau đó xuất ra màn hình hay sử dụng tùy ý.

##### b) SendLIST\_NLSTCommand

Đây là hàm con của hàm List:

```
string SendLIST_NLSTCommand(FtpClientContext* cInt, string type, string loc);
```

- + Đầu vào: struct FtpClientContext, kiểu liệt kê (ls/dir), đường dẫn thư mục.
- + Đầu ra: string chứa tất cả thông tin liệt kê nhận được: chuỗi rỗng nếu thất bại.
- + Chức năng: thực hiện gửi lệnh lên server và nhận dữ liệu về

Nhiệm vụ của hàm khá đơn giản:

- Kiểm tra xem biến type là ls thì Control gửi lệnh NLST + loc, dir thì Control gửi LIST + loc. Lưu ý khi gửi lệnh, ở cuối chuỗi phải có dấu ENTER.

- Sau đó nhận phản hồi từ server, nếu mã gửi về là 150 thì tiếp tục, khác thì thoát hàm, trả về rỗng.
- Nếu tiếp tục, ta phải xem server hiện tại đang sử dụng cơ chế active hay passive, nếu là passive thì không có gì, nếu như là active thì phải accept listen socket đã tạo ở hàm List. Tại đây, do chỉ cần truyền dữ liệu một lần, chúng tôi tạo Data là ListenSocket, sau đó cho Data tự accept chính nó. Bạn cũng có thể làm một biến SOCKET khác tương tự nếu không muốn như thế.
- Dùng SOCKET accept đó để nhận dữ liệu từ server, chép vào chuỗi string S cho đến khi không còn dữ liệu.
- Nhận tin từ server, mã 226 tức đã truyền dữ liệu thành công, còn 425 thì là thất bại.
- Trả về S.

Hình ảnh minh họa liệt kê thư mục thành công:

```
ftp> ls
150 Opening data channel for directory listing of "/"
226 Successfully transferred "/"
B C.txt
DoAnWireshark.pdf
eryn.ico
Files
Ha lu
luxubu.txt
Mo ly
No way.txt
put
tada.png
Uhhhhh.txt
Ultimate Showdown.txt
UsersShare
```

```
-user1 (127.0.0.1)> PORT 127.0.0.1,12,106
-user1 (127.0.0.1)> 200 Port command successful
-user1 (127.0.0.1)> NLST
-user1 (127.0.0.1)> 150 Opening data channel for directory listing of "/"
-user1 (127.0.0.1)> 226 Successfully transferred "/"
```

```
ftp> dir
150 Opening data channel for directory listing of "/"
226 Successfully transferred "/"
-rw-r--r-- 1 ftp ftp          52 May 25 08:36 B C.txt
-rw-r--r-- 1 ftp ftp    351352 May 19 14:30 DoAnWireshark.pdf
-rw-r--r-- 1 ftp ftp    92672 May 24 20:29 eryn.ico
drwxr-xr-x 1 ftp ftp           0 May 19 19:36 Files
-rw-r--r-- 1 ftp ftp           0 May 25 11:00 Ha lu
-rw-r--r-- 1 ftp ftp     512 May 25 11:33 luxubu.txt
drwxr-xr-x 1 ftp ftp           0 May 24 20:30 Mo ly
-rw-r--r-- 1 ftp ftp    2048 May 25 11:33 No way.txt
-rw-r--r-- 1 ftp ftp           0 May 10 16:24 put
-rw-r--r-- 1 ftp ftp    47291 May 25 10:08 tada.png
-rw-r--r-- 1 ftp ftp    2048 May 22 20:37 Uhhhhh.txt
-rw-r--r-- 1 ftp ftp    1740 May 19 19:16 Ultimate Showdown.txt
drwxr-xr-x 1 ftp ftp           0 May 22 19:14 UsersShare
```

```
user1 (127.0.0.1)> PORT 127.0.0.1,12,107
user1 (127.0.0.1)> 200 Port command successful
user1 (127.0.0.1)> LIST
user1 (127.0.0.1)> 150 Opening data channel for directory listing of "/"
user1 (127.0.0.1)> 226 Successfully transferred "/"
```

## 5. Upload một file đến server (put)

### a) Put

```
void Put(FtpClientContext* clnt, string file, int& port);
```

- + Đầu vào: struct FtpClientContext, tên file (bao hàm đường dẫn) trong máy client, port client hiện tại.
- + Đầu ra: không có
- + Chức năng: upload một file lên server

Nếu như tên file là một đường dẫn đầy đủ (ví dụ: E:\\<folder>\\file) thì không cần qua các thao tác kiểm tra, gắn ghép đường dẫn, tiến hành lấy tên file ra khỏi đường dẫn, xóa dấu ngoặc kép (nếu có), kiểm tra P và thực hiện các công việc như hàm List đã đề cập, chỉ khác hàm con lần này là StorFileToServer.

Tạm gọi tên file trong máy client là locfname, tên file này trên server (tức up file này lên server với một tên khác) là serfname.

Nếu như locfname không thỏa yêu cầu trên, chúng ta sẽ phải:

- Kiểm tra locfname có rỗng không, nếu rỗng, đề nghị nhập Local file và Remote file (locfname và serfname).
- Kiểm tra locfname có đường dẫn đầy đủ không (có dấu hai chấm là đủ). Nếu không, gắn locfname với đường dẫn mặc định (đường dẫn mặc định được định nghĩa trong header)
- Xóa hết các dấu ngoặc kép trong locfname, serfname (trong code, chúng tôi có sử dụng hàm DeleteAllDoubleQuotationMarks, hàm này vô cùng đơn giản, chỉ sử dụng vòng for lặp đến cuối chuỗi, thấy dấu ngoặc kép thì xóa).

Sau khi hoàn tất các thao tác trên, chúng ta mới kiểm tra P và gọi StorFileToServer.

Lưu ý tăng giá trị port lên 1, đóng các socket không còn sử dụng. Sau khi đóng socket sử dụng cho kênh truyền dữ liệu thì server mới biết được quá trình upload đã hoàn tất. Server gửi tin về có mã 226.

### b) StorFileToServer

```
bool StorFileToServer(FtpClientContext* clnt, string file, string serfname);
```

- + Đầu vào: struct FtpClientContext, tên file trong máy client (locfname), tên file này trên server (serfname)
- + Đầu ra: 0: upload thất bại, 1: upload thành công
- + Chức năng: thực hiện gửi lệnh up và quá trình truyền dữ liệu

Đầu tiên, chúng ta phải xác định đây là file chứa chữ (VD: .doc) hay file hình ảnh (VD: .png). Do đuôi file rất đa dạng, vì vậy, chúng tôi quyết định cho người dùng



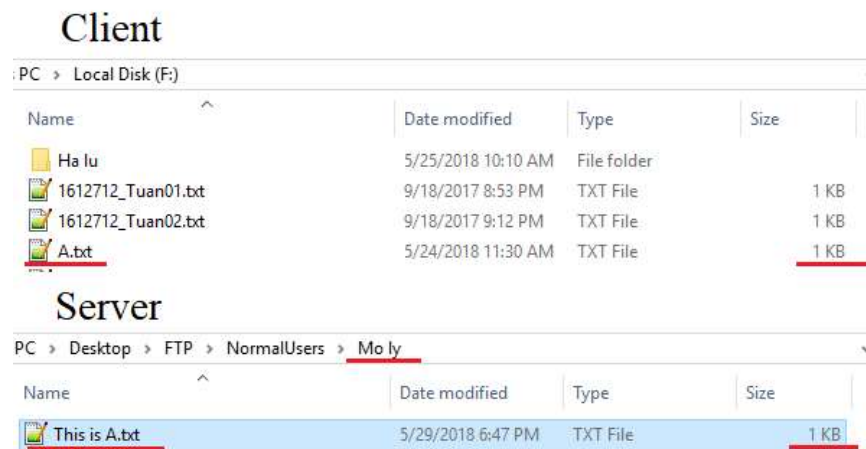
tự xác định cho nhanh và chính xác, sai thì do người dùng. Chữ A (ASCII) dùng cho file chữ, I (IMAGE) dùng cho file ảnh. Rồi dùng Control gửi lệnh TYPE A hoặc I cho server.

Tiếp theo, để bắt đầu quá trình truyền dữ liệu lên server, ta cho Control gửi lệnh STOR + serfname. Nếu mã nhận về là 150 tức đã thành công mở cổng truyền dữ liệu. Hãy nhớ accept nếu đang sử dụng active. Đến đây thì mọi chuyện đơn giản rồi, chúng ta mở locfname dưới dạng nhị phân và chỉ đọc, lần lượt đọc vào buffer và dùng Data gửi lên server khi chưa đi hết tập tin, đóng tập tin lại, trả về 1. Ngược lại, trả về 0.

Hình ảnh minh họa:

```
ftp> put
Local file: A.txt
Remote file: "/Mo ly"/"This is A.txt"
200 Port command successful
Please enter the data type of the file (A/I): A
200 Type set to A
150 Opening data channel for file upload to server of "/Mo ly/This is A.txt"
226 Successfully transferred "/Mo ly/This is A.txt"

user1 (127.0.0.1)> PORT 127,0,0,1,12,109
user1 (127.0.0.1)> 200 Port command successful
user1 (127.0.0.1)> TYPE A
user1 (127.0.0.1)> 200 Type set to A
user1 (127.0.0.1)> STOR Mo ly/This is A.txt
user1 (127.0.0.1)> 150 Opening data channel for file upload to server of "/Mo ly/This is A.txt"
user1 (127.0.0.1)> 226 Successfully transferred "/Mo ly/This is A.txt"
```



## 6. Download một file từ server (get)

### a) Get

`void Get(FtpClientContext* clnt, string serfname, int& port)`

+ Đầu vào: struct FtpClientContext, tên file trên server (serfname), port client hiện tại.

- + Đầu ra: không có.
- + Chức năng: download một file từ server

Nếu như serfname truyền vào khác rỗng, ta lấy locfname bằng cách ghép đường dẫn mặc định của client với tên của file đó trên server. Xóa dấu nháy kép, kiểm tra P (active hay passive) rồi gọi hàm DownloadFile.

Nếu như serfname truyền vào là rỗng, lúc này chúng ta sẽ yêu cầu người dùng nhập vào Remote file (serfname) và Local file (locfname). Kiểm tra locfname, nếu không có dấu hai chấm thì ghép với đường dẫn mặc định. Xóa dấu nháy kép của hai tên. Kiểm tra P và gọi hàm DownloadFile.

Tăng giá trị port lên 1, đóng các socket không còn sử dụng. Sau khi đóng socket kênh truyền dữ liệu thì quá trình truyền dữ liệu mới kết thúc.

### *b) DownloadFile*

```
bool DownloadFile(FtpClientContext* clnt, string serfname, string locfname);
```

- + Đầu vào: struct FtpClientContext, tên file trên server (serfname), tên của file trong client sau khi download (locfname)
- + Đầu ra: 0: download thất bại, 1: download hoàn tất
- + Chức năng: thực hiện gửi lệnh tải và quá trình truyền dữ liệu

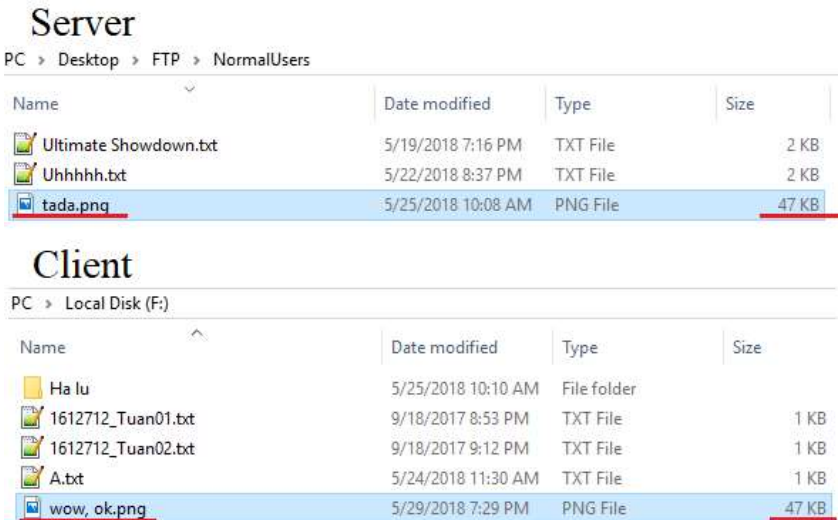
Dùng Control gửi lệnh RETR + serfname cho server, nếu mã server trả lời là 150 thì bắt đầu. Nếu là active thì nhớ accept. Tạo một file trong máy client theo locfname, mở dưới dạng ghi và nhị phân, nếu file đó đã tồn tại thì xóa hết dữ liệu trong đó. Lần lượt nhận dữ liệu từ server và ghi vào file cho đến khi không còn dữ liệu nữa, đóng file lại, trả về 1. Mã khác 150 thì trả về 0.



Hình ảnh minh họa:

```
ftp> get
Remote file: tada.png
Local file: "wow, ok.png"
150 Opening data channel for file download from server of "/tada.png"
226 Successfully transferred "/tada.png"

user1 (127.0.0.1)> PORT 127.0.0.1,12,110
user1 (127.0.0.1)> 200 Port command successful
user1 (127.0.0.1)> RETR tada.png
user1 (127.0.0.1)> 150 Opening data channel for file download from server of "/tada.png"
user1 (127.0.0.1)> 226 Successfully transferred "/tada.png"
```



**Server**

PC > Desktop > FTP > NormalUsers

Name	Date modified	Type	Size
Ultimate Showdown.txt	5/19/2018 7:16 PM	TXT File	2 KB
Uhhhhh.txt	5/22/2018 8:37 PM	TXT File	2 KB
tada.png	5/25/2018 10:08 AM	PNG File	47 KB

**Client**

PC > Local Disk (F:)

Name	Date modified	Type	Size
Ha lu	5/25/2018 10:10 AM	File folder	
1612712_Tuan01.txt	9/18/2017 8:53 PM	TXT File	1 KB
1612712_Tuan02.txt	9/18/2017 9:12 PM	TXT File	1 KB
A.txt	5/24/2018 11:30 AM	TXT File	1 KB
wow, ok.png	5/29/2018 7:29 PM	PNG File	47 KB

## 7. Upload nhiều file đến server (mput)

### a) *MPut*

```
void MPut(FtpClientContext* clnt, string files, int& port);
```

- + Đầu vào: struct FtpClientContext, locfname, port client hiện tại.
- + Đầu ra: không có
- + Chức năng: upload nhiều file đến server

Nếu locfname trống thì đơn giản là cho người dùng nhập lại, ngăn cách giữa các tên file là dấu khoảng trắng. Tại đây, chúng tôi dùng hàm `SeperateFilePaths` để tách các tên file ra. Những tên không có dấu hai chấm thì ghép với đường dẫn mặc định. Khi locfname không rỗng:

- Trường hợp 1: có tên file rõ ràng

Trong trường hợp này chỉ có tập tin, chúng ta chỉ việc lấy ra tên các tập tin này và lần lượt gọi hàm put với từng tên lấy được. Nhớ xóa dấu nhảy kếp.

- Trường hợp 2: Có folder

Lúc này, ta cần biết được trong folder có những file gì, chúng tôi sử dụng thư viện dirent và hàm LisFilesInClientFolder để làm điều đó, nhưng lại liệt kê luôn cả những thứ không phải là file hoặc folder mà có thể upload được lên server nên cần có bộ lọc.

➤ Trường hợp 3: Tên file dạng \*.<đuôi file> (VD: \*.jpg)

Cũng tương tự như trường hợp 2, nhưng cụ thể hơn chút là cần up những file có đuôi là gì, điều này giúp chúng ta phân biệt dễ dàng hơn. Ta dùng một biến để lưu lại định dạng đuôi và một biến để lưu lại đường dẫn, nếu không có đường dẫn thì dùng đường dẫn mặc định.

Ta gọi hàm LisFilesInClientFolder với biến đường dẫn vừa có được để lấy hết tất cả các tên file, folder, những thứ khác ở trong thư mục đó vào string str1, những tên này được ngăn cách với nhau bởi dấu xuống dòng (do LisFilesInClientFolder quy định). Ta lần lượt lấy các file (theo cơ bản thì có ít nhất một dấu chấm) trong str1, cho một biến int chạy từ dấu chấm đến cuối dòng lấy được đuôi file, đem đuôi file đi so sánh với đuôi file đã nhập trước đó, khớp thì đẩy vào string str2.

Sau khi đã có các tên cần thiết cho việc upload, ta lại lần lượt lấy các tên trong chuỗi lưu trữ đó ra, truyền vào hàm put và hỏi xem người dùng có muốn up file ấy không. Port client sẽ được tăng trong hàm Put nên chúng ta không cần quan tâm.

### b) LisFilesInClientFolder

```
bool ListFileInClientFolder(string dirpath, string& allnames);
```

- + Đầu vào: đường dẫn folder cần liệt kê danh sách thư mục, tập tin bên trong, string để lưu các tên liệt kê được.
- + Đầu ra: 0: không có lỗi, liệt kê hoàn thành; 1: không mở được đường dẫn.
- + Chức năng: liệt kê các file/folder trong thư mục của client

Nếu mở được đường dẫn thì chép các tên vào string allnames, đến khi hết thì đóng đường dẫn thư mục lại. Xem code để thấy rõ hơn.

### c) SeparateFilePaths

```
string SeparateFilePaths(string &files);
```

- + Đầu vào: một chuỗi gồm rất nhiều file/folder ((không) có đường dẫn) ngăn cách nhau bởi dấu khoảng trắng.
- + Đầu ra: file/folder đầu tiên trong rất nhiều file đó.
- + Chức năng: lấy đầy đủ tên file/folder thứ nhất, xóa tên đó đi trong danh sách.

Lấy tên file/folder đầu tiên ra trong một chuỗi gồm nhiều tên ấy. Tạo vòng lặp chạy từ đầu đến cuối. Trong đó, nếu gặp dấu nháy kép thì flag tăng 1. Khi flag là số lẻ tức là đang trong một tên có khoảng trắng, vì vậy khi gặp khoảng trắng thì bỏ qua; khi flag là số chẵn tức là chỉ số đã đi qua cái tên có khoảng trắng, nếu gặp

khoảng trống nữa thì đây là kết thúc của một tên, lấy tên này ra, xóa phần tên này trong chuỗi truyền vào, kết thúc vòng lặp, trả về tên lấy được.

Hình ảnh minh họa (up tất cả .txt trong folder Ha lu và 1 .pdf khác):

```
ftp> mput "Ha lu"/*.txt DoAnWireshark.pdf
mput F:\Ha lu\luxubu.txt? yes
200 Port command successful
Please enter the data type of the file (A/I): A
200 Type set to A
150 Opening data channel for file upload to server of "/luxubu.txt"
226 Successfully transferred "/luxubu.txt"
mput F:\Ha lu\No way.txt? yes
200 Port command successful
Please enter the data type of the file (A/I): A
200 Type set to A
150 Opening data channel for file upload to server of "/No way.txt"
226 Successfully transferred "/No way.txt"
mput F:\Ha lu\W A 00.txt? yes
200 Port command successful
Please enter the data type of the file (A/I): A
200 Type set to A
150 Opening data channel for file upload to server of "/W A 00.txt"
226 Successfully transferred "/W A 00.txt"
mput F:\DoAnWireshark.pdf? yes
200 Port command successful
Please enter the data type of the file (A/I): A
200 Type set to A
150 Opening data channel for file upload to server of "/DoAnWireshark.pdf"
226 Successfully transferred "/DoAnWireshark.pdf"

user1 (127.0.0.1)> PORT 127,0,0,1,12,114
user1 (127.0.0.1)> 200 Port command successful
user1 (127.0.0.1)> TYPE A
user1 (127.0.0.1)> 200 Type set to A
user1 (127.0.0.1)> STOR luxubu.txt
user1 (127.0.0.1)> 150 Opening data channel for file upload to server of "/luxubu.txt"
user1 (127.0.0.1)> 226 Successfully transferred "/luxubu.txt"
user1 (127.0.0.1)> PORT 127,0,0,1,12,115
user1 (127.0.0.1)> 200 Port command successful
user1 (127.0.0.1)> TYPE A
user1 (127.0.0.1)> 200 Type set to A
user1 (127.0.0.1)> STOR No way.txt
user1 (127.0.0.1)> 150 Opening data channel for file upload to server of "/No way.txt"
user1 (127.0.0.1)> 226 Successfully transferred "/No way.txt"
user1 (127.0.0.1)> PORT 127,0,0,1,12,116
user1 (127.0.0.1)> 200 Port command successful
user1 (127.0.0.1)> TYPE A
user1 (127.0.0.1)> 200 Type set to A
user1 (127.0.0.1)> STOR W A 00.txt
user1 (127.0.0.1)> 150 Opening data channel for file upload to server of "/W A 00.txt"
user1 (127.0.0.1)> 226 Successfully transferred "/W A 00.txt"
user1 (127.0.0.1)> PORT 127,0,0,1,12,117
user1 (127.0.0.1)> 200 Port command successful
user1 (127.0.0.1)> TYPE A
user1 (127.0.0.1)> 200 Type set to A
user1 (127.0.0.1)> STOR DoAnWireshark.pdf
user1 (127.0.0.1)> 150 Opening data channel for file upload to server of "/DoAnWireshark.pdf"
user1 (127.0.0.1)> 226 Successfully transferred "/DoAnWireshark.pdf"
```

## Client

Name	Date modified	Type	Size
Ha lu	5/29/2018 9:04 PM	File folder	
1612712_Tuan01.txt	9/18/2017 8:53 PM	TXT File	1 KB
1612712_Tuan02.txt	9/18/2017 9:12 PM	TXT File	1 KB
A.txt	5/24/2018 11:30 AM	TXT File	1 KB
DoAnWireshark.pdf	5/19/2018 2:30 PM	Adobe Acrobat D...	344 KB
wow, ok.png	5/29/2018 7:29 PM	PNG File	47 KB

s PC > Local Disk (F:) > Ha lu

Name	Date modified	Type	Size
luxubu.txt	5/22/2018 5:07 PM	TXT File	1 KB
No way.txt	5/24/2018 8:28 PM	TXT File	2 KB
W A OO.txt	5/20/2018 10:20 AM	TXT File	2 KB

## Server

PC > Desktop > FTP > NormalUsers >

Name	Date modified	Type	Size
DoAnWireshark.pdf	5/29/2018 9:06 PM	Adobe Acrobat D...	344 KB
W A OO.txt	5/29/2018 9:06 PM	TXT File	2 KB
No way.txt	5/29/2018 9:06 PM	TXT File	2 KB
luxubu.txt	5/29/2018 9:06 PM	TXT File	1 KB

## 8. Download nhiều file từ server (mget)

```
void MGet(FtpClientContext* clnt, string list, int& port);
```

- + Đầu vào: FtpClientContext, chuỗi gồm các file/folder cần download (list), port client hiện tại.
- + Đầu ra: không có.
- + Chức năng: download nhiều file từ server

Nếu như list rỗng thì yêu cầu người dùng nhập lại.

Tạo một vòng lặp A lần lượt lấy từng tên file/folder ra, xóa tên đó trong list. Gọi hàm List với kiểu là "ls" và đường dẫn là tên file/folder vừa lấy ra đó, dùng string str1 lưu lại, lúc này, trong str1 sẽ có rất nhiều file/folder và được ngăn cách nhau bởi "\r\n". Ta lại có một vòng lặp B lần lượt lấy các tên trong str1 ra, nếu đó là file thì đẩy vào str2, nếu nó là folder (không có dấu chấm nào) thì push lại vào list, kết thúc vòng lặp B. Mục đích của việc để folder vào lại list là để sau một lúc chạy các vòng lặp, trong str2 chỉ toàn là file, trong list toàn là folder, và list trong vòng lặp



A lại liệt kê ra file rồi để vào str2. Vòng A kết thúc khi list rỗng, cuối cùng, str2 sẽ chứa tên của tất cả các file cần download.

Sau đó đơn giản chỉ việc lấy lần lượt các tên trong str2 ra, hỏi người dùng có download file này không, có thì gọi hàm Get.

Hàm List có thể áp dụng cho cả file kiểu \*.<đuôi> nên không cần lo lắng. Còn về dạng <đường dẫn>/\* thì trong vòng A khi lấy được tên phải xóa “/\*” đi, do không thể ls được /\*.

Hình ảnh minh họa (down tất cả file trong folder Files, trong thư mục Files có B C.txt và thư mục Alo, trong Alo có A.txt):

```
ftp> mget Files/*
150 Opening data channel for directory listing of "/Files"
226 Successfully transferred "/Files"
150 Opening data channel for directory listing of "/Files/Alo"
226 Successfully transferred "/Files/Alo"
mget Files/B C.txt? yes
150 Opening data channel for file download from server of "/Files/B C.txt"
226 Successfully transferred "/Files/B C.txt"
mget Files/Alo/A.txt? yes
150 Opening data channel for file download from server of "/Files/Alo/A.txt"
226 Successfully transferred "/Files/Alo/A.txt"
ftp>
```

```
user1 (127.0.0.1)> TYPE A
user1 (127.0.0.1)> 200 Type set to A
user1 (127.0.0.1)> PORT 127,0,0,1,91,48
user1 (127.0.0.1)> 200 Port command successful
user1 (127.0.0.1)> NLST Files
user1 (127.0.0.1)> 150 Opening data channel for directory listing of "/Files"
user1 (127.0.0.1)> 226 Successfully transferred "/Files"
user1 (127.0.0.1)> PORT 127,0,0,1,91,49
user1 (127.0.0.1)> 200 Port command successful
user1 (127.0.0.1)> NLST Files/Alo
user1 (127.0.0.1)> 150 Opening data channel for directory listing of "/Files/Alo"
user1 (127.0.0.1)> 226 Successfully transferred "/Files/Alo"
user1 (127.0.0.1)> PORT 127,0,0,1,91,50
user1 (127.0.0.1)> 200 Port command successful
user1 (127.0.0.1)> RETR Files\B C.txt
user1 (127.0.0.1)> 150 Opening data channel for file download from server of "/Files/B C.txt"
user1 (127.0.0.1)> 226 Successfully transferred "/Files/B C.txt"
user1 (127.0.0.1)> PORT 127,0,0,1,91,51
user1 (127.0.0.1)> 200 Port command successful
user1 (127.0.0.1)> RETR Files\Alo\A.txt
user1 (127.0.0.1)> 150 Opening data channel for file download from server of "/Files/Alo/A.txt"
user1 (127.0.0.1)> 226 Successfully transferred "/Files/Alo/A.txt"
```

## Server

PC > Desktop > FTP > NormalUsers > Files

Name	Date modified	Type	Size
Alo	5/19/2018 4:03 PM	File folder	
B C.txt	5/2/2018 10:48 AM	TXT File	1 KB

PC > Desktop > FTP > NormalUsers > Files > Alo

Name	Date modified	Type	Size
A.txt	5/2/2018 10:46 AM	TXT File	1 KB

## Client

A.txt	5/29/2018 10:35 PM	TXT File	1 KB
DoAnWireshark.pdf	5/19/2018 2:30 PM	Adobe Acrobat D...	344 KB
wow, ok.png	5/29/2018 7:29 PM	PNG File	47 KB
B C.txt	5/29/2018 10:35 PM	TXT File	1 KB

## 9. Thay đổi đường dẫn trên server (cd)

`void ChangeDirector(FtpClientContext *context, string pathName);`

- + Đầu vào: FtpClientContext, pathName (tên đường dẫn).
- + Đầu ra: không có.
- + Chức năng: thay đổi đường dẫn trên server.
- + Cách hoạt động:
  - Nếu tên đường dẫn trống: cho người dùng nhập tên đường dẫn.
  - Gửi lệnh lên server trên kênh ftpControl, định dạng: 'CWD pathName\r\n'
  - Nhận thông báo kết quả từ server và xuất ra màn hình.

Hình ảnh minh họa:

- Truy cập vào một thư mục trên server:

```
ftp> cd
Remote directory tinh
250 CWD successful. "/tinh" is current directory.
```

- Trở về thư mục trước:

```
ftp> cd ..
250 CWD successful. "/" is current directory.
```



## 10. Thay đổi đường dẫn dưới Client (lcd)

`void ChangeLocalDirector(string pathName);`

- + Đầu vào: pathName (tên đường dẫn).
- + Đầu ra: không có.
- + Chức năng: thay đổi đường dẫn dưới Client.
- + Cách hoạt động:
  - Dùng 1 string directorLocal lưu lại đường dẫn hiện tại dưới Client cho phiên hoạt động sau. (mặc định là 'C:\')
  - Nếu pathName rỗng thì cho người dùng nhập.
  - Dùng hàm system có sẵn trong C/C++ để thay đổi đường dẫn. Kiểm tra đường dẫn có đúng không bằng kết quả trả về của lệnh: `system('cd ' + pathName + ' > nul 2> nul')` trong đó `> nul 2> nul` nhằm không xuất kết quả ra màn hình mà chỉ có mục đích kiểm tra.
  - Nếu đường dẫn tồn tại thì dùng lệnh: `system('cd ' + pathName + ' && cd')` trong đó `&& cd` xuất ra màn hình. Vì mỗi lệnh system là một phiên hoạt động khác nhau trên cmd terminal nên phải làm chung trên 1 lần gọi lệnh.
  - Cập nhật lại directorLocal.

Hình ảnh minh họa:

- Truy cập vào một thư mục dưới Client:

```
ftp> lcd C:\
Local directory now C:\
ftp> lcd Users
Local directory now C:\Users
```

- Trở về thư mục trước:

```
ftp> lcd ..
Local directory now C:\
```

## 11. Xóa một file trên server (delete)

`void DeleteFileOnServer(FtpClientContext *context, string fileName);`

- + Đầu vào: FtpClientContext, fileName (tên file cần xóa).
- + Đầu ra: không có.
- + Chức năng: xóa một file trên server.
- + Cách hoạt động:
  - Gửi lệnh theo định dạng: `'DELE ' + fileName + '\r\n'` trên kênh ftpControl
  - Nhận thông báo kết quả từ server và xuất ra.

Hình ảnh minh họa :

- Xóa 1 file:

```
ftp> delete b.txt
250 File deleted successfully
```

- Xóa 1 file không tồn tại :

```
ftp> delete a.txt
550 File not found
```

## 12. Xóa cùng lúc nhiều file trên server (mdelete)

`void DeleteMultipleFileOnServer(FtpClientContext *context, string pathName, int& port);`

- + Đầu vào: ftpClientContext, fileName (tên file cần xóa)
- + Đầu ra: không có
- + Chức năng: xóa cùng lúc nhiều file trên server
- + Cách hoạt động:
  - Nhập tên file nếu pathName trống.
  - Chọn chế độ truyền tin và mở mode Active hoặc Passive tùy thuộc vào trường passiveMode trong FtpClientContext truyền vào.
  - Nếu là xóa nhiều file theo tên file lần lượt trong pathName thì kiểm tra từng file có tồn tại hay không bằng cách gửi lệnh 'NLST ' + pathName + '\r\n'. và nhận lại kết quả từ Server. Nếu có 1 file không đúng thì dừng lại, ngược lại xóa từng file bằng lệnh 'DELE ' + pathName + '\r\n '.
  - Nếu là xóa tất cả các file (hoặc tất cả file theo định dạng), giống như trên chọn kiểu truyền file và kiểm tra bằng cách gửi lệnh 'NLST ' + pathName + '\r\n'. Nếu thành công thì gửi lệnh NLST tới Server và nhận danh sách tên file trên SOCKET ftpDataControl đã tạo ở trên. Dựa theo danh sách tên và tùy vào xóa tất cả hay xóa theo định dạng mà ta xóa bằng lệnh 'DELE ' + '[tên file cần xóa]' + '\r\n'.

Hình ảnh minh họa:

- Xóa theo danh sách tên:

```
ftp> mdelete a.txt b.txt
200 Type set to A
mdelete a.txt?
226 Successfully transferred "/b.txt"
mdelete b.txt?
250 File deleted successfully
ftp>
```

- Xóa tất cả các file:

```
ftp> mdelete *
200 Type set to A
mdelete a.txt?
250 File deleted successfully
mdelete anh?
550 File not found
mdelete b.txt?
250 File deleted successfully
mdelete c.bmp?
250 File deleted successfully
mdelete tinh?
550 File not found
ftp>
```

- Xóa tất cả các file theo định dạng (trong đường dẫn hiện tại có các định dạng file khác) :

```
ftp> mdelete *.txt
200 Type set to A
mdelete a.txt?
250 File deleted successfully
mdelete b.txt?
250 File deleted successfully
ftp>
```

### 13. Tạo thư mục trên server (mkdir).

`void MakeDir(FtpClientContext *context, string pathName);`

- + Đầu vào : FtpClientContext, pathName (tên thư mục cần tạo).
- + Đầu ra : không có.
- + Chức năng : tạo thư mục trên server.
- + Cách hoạt động :
  - Nếu pathName trống thì cho người dùng nhập vào
  - Gửi lệnh 'MKD' + pathName ++ '\r\n' lên server để tạo thư mục.
  - Nhận kết quả trả về từ server và xuất ra.

Hình ảnh minh họa

- Tạo thư mục:

```
ftp> mkdir 6muis
257 "/6muis" created successfully
ftp>
```

> This PC > Local Disk (C:) > ftp > user

Name	Date modified	Type
6muis	29/05/2018 11:37 ...	File folder
anh	15/05/2018 2:22 CH	File folder
tinh	16/05/2018 12:03 SA	File folder
c	29/05/2018 11:35 ...	BMP File

## 14. Xóa thư mục rỗng trên server (rmdir)

`void DeleteEmptyFolder(FtpClientContext *context, string folder);`

- + Đầu vào: FtpClientContext, chuỗi tên folder cần xóa
- + Đầu ra: không có
- + Chức năng: xóa thư mục rỗng trên server
- + Cách hoạt động:

Khi đưa các tham số vào hàm, hàm sẽ kiểm tra xem chuỗi tên folder có rỗng hay không, nếu rỗng thì yêu cầu người dùng nhập vào. Sau đó tạo 1 chuỗi string strSend để gửi lên server, gửi chuỗi strSend lên server bằng hàm Send đã được giải thích ở trên. Cuối cùng là nhận thông báo từ server, in ra màn hình và kết thúc.

Hình ảnh minh họa:

```
ftp> rmdir
Directory name a b
250 Directory deleted successfully
```

## 15. Hiện thị đường dẫn hiện tại trên server (pwd)

`void DisplayCurrentWorkingDirectory(FtpClientContext *client);`

- + Đầu vào: FtpClientContext
- + Đầu ra: không có
- + Chức năng: hiện thị đường dẫn hiện tại trên server
- + Cách hoạt động:

Tạo ra 1 chuỗi gửi đi strSend với nội dung “XPWD\r\n”, gửi chuỗi strSend lên server bằng hàm Send đã được giải thích ở trên. Cuối cùng là nhận thông báo từ server, in ra màn hình và kết thúc.

Hình ảnh minh họa:

```
230 Logged on
ftp> pwd
257 "/" is current directory.
ftp> cd Files
250 CWD successful. "/Files" is current directory.
ftp> pwd
257 "/Files" is current directory.
```

## 16. Thoát khỏi server (quit)

`void Quit(FtpClientContext *client);`

- + Đầu vào: FtpClientContext
- + Đầu ra: không có

- + Chức năng: thoát khỏi server
- + Cách hoạt động:

Tạo ra 1 chuỗi gửi đi strSend với nội dung “QUIT\r\n”, gửi chuỗi strSend lên server bằng hàm Send đã được giải thích ở trên. Cuối cùng là nhận thông báo từ server, in ra màn hình và kết thúc.

Hình ảnh minh họa:

```
230 Logged on
ftp> quit
221 Goodbye
```

## C. Đánh giá đồ án:

❖ Chức năng làm được:

Tên	Login	ls/dir	put	get	mput	mget	cd	lcd
Mức độ hoàn thành	100%	100%	100%	100%	100%	100%	100%	100%
Tên	delete	mdelete	mkdir	rmdir	pwd	passive	active	quit
Mức độ hoàn thành	100%	100%	100%	100%	100%	100%	100%	100%

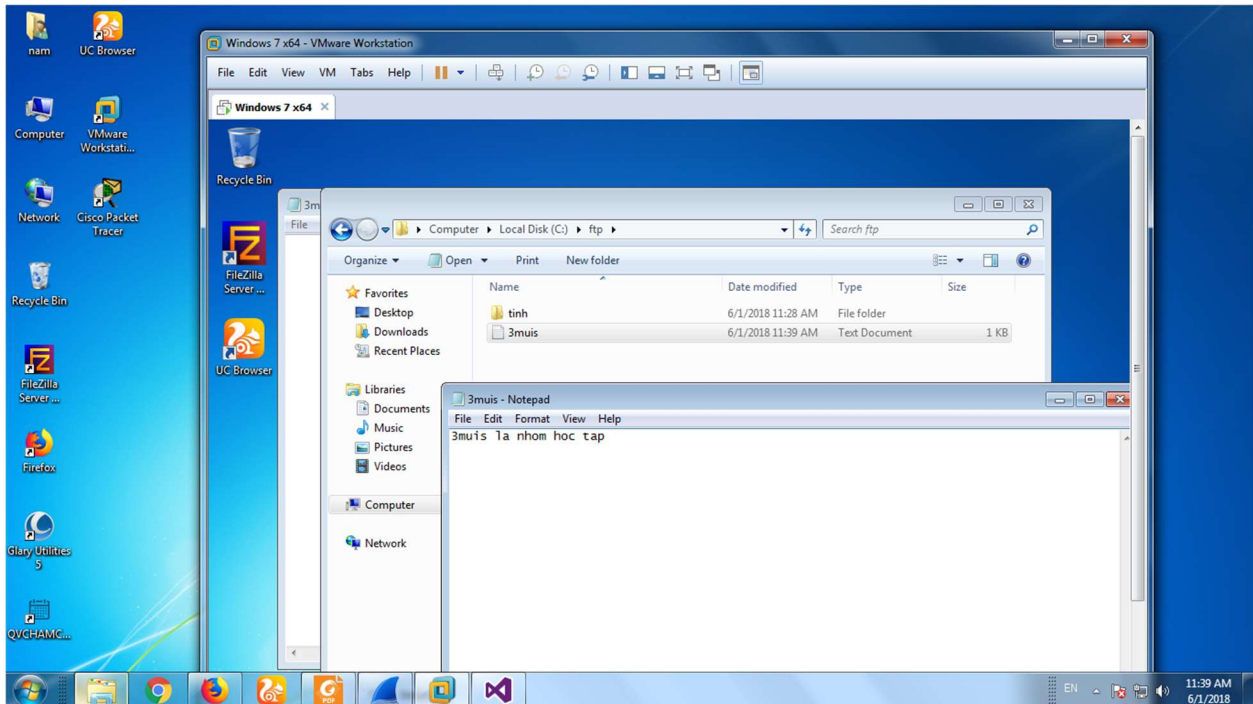
❖ Chức năng không làm được: Không có

## D. Bắt gói tin bằng WireShark:

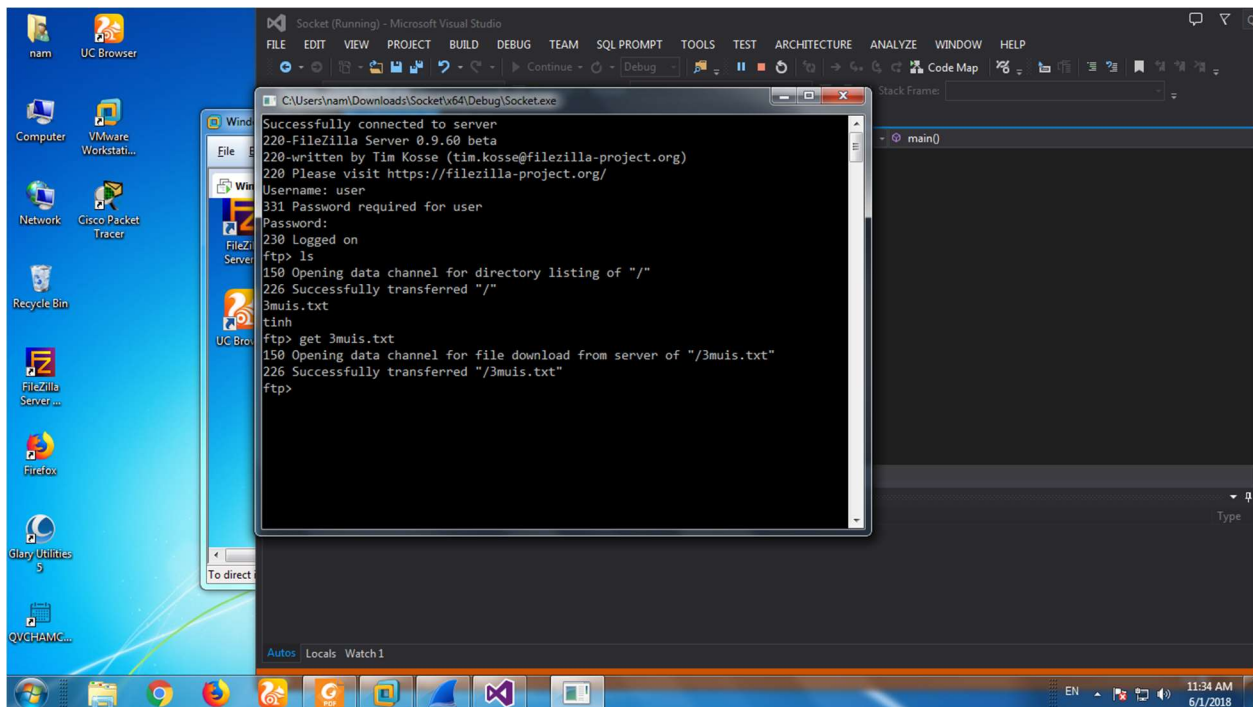
Trên server trong máy ảo, tại thư mục ftp có một tập tin 3muis.txt. Sử dụng chương trình đã viết để download tập tin ấy mà không thay đổi nội dung.

Hình ảnh minh chứng:

- Ảnh thể hiện file 3muis.txt trên server:

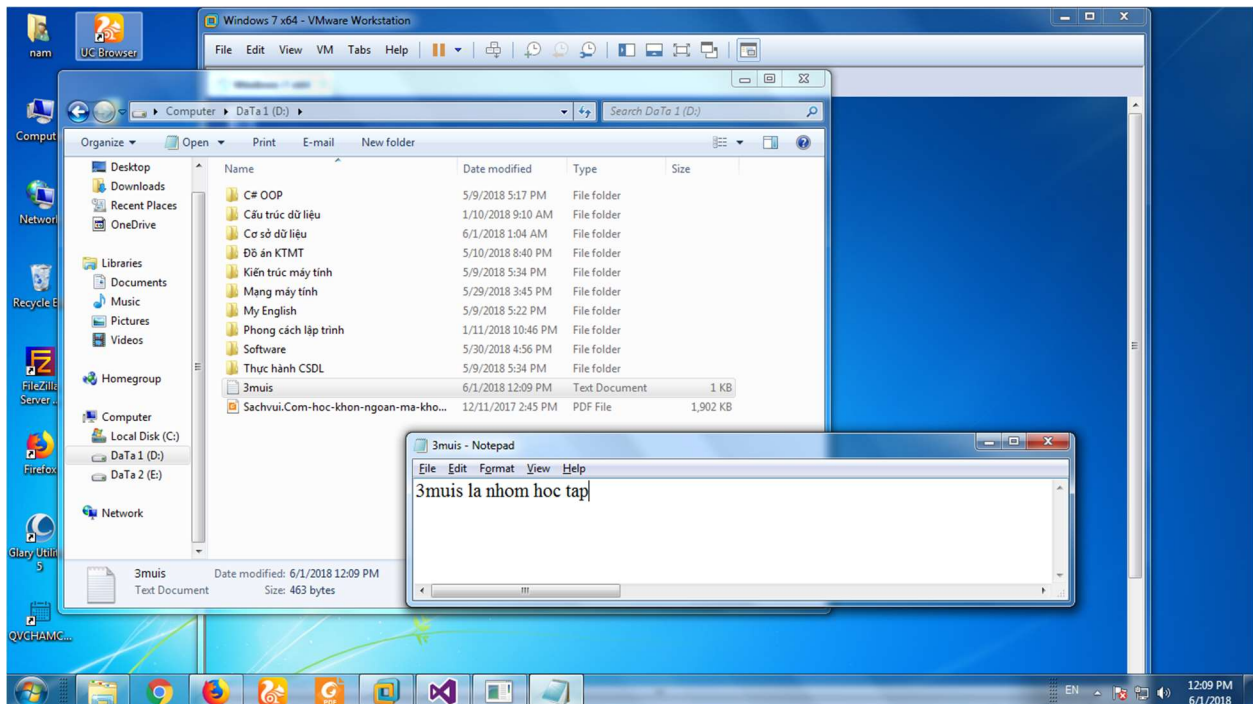


- Ảnh chạy hàm get (tại đây sử dụng active và có đường dẫn mặc định là ổ đĩa D trong máy client):

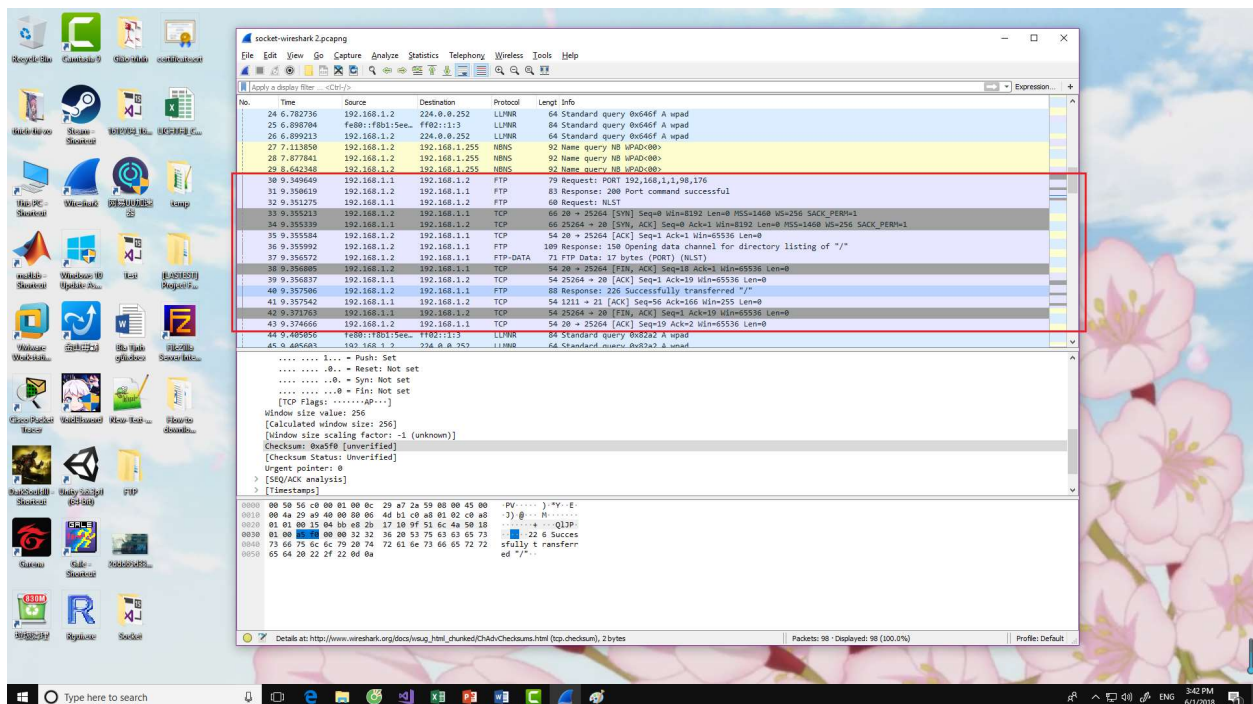




- Kết quả trong máy client:



- Ảnh Wireshark bắt gói tin (toàn màn hình):



- Mô tả (control port của client là 1211):

+ Khi client gửi lệnh PORT 192,168,1,1,78,183, client nói cho server biết rằng server hãy dùng active mode để truyền file, đồng thời bảo là nó sẽ mở port 20407 ( $78 \times 256 + 183$ ) để server kết nối đến.

192.168.1.1	192.168.1.2	FTP	79 Request: PORT 192,168,1,1,98,177
192.168.1.2	192.168.1.1	FTP	83 Response: 200 Port command successful

+ Server nhận được tin, liền tiến hành bắt tay 3 bước với client, nó gửi SYN cho client (20 → 20407) rồi nhận lại SYN,ACK và gửi lại ACK để hoàn tất quá trình bắt tay. Trong lúc đó, client gửi lệnh RETR 3muis.txt yêu cầu server cho download file này.

192.168.1.1	192.168.1.2	FTP	70 Request: RETR 3muis.txt
192.168.1.2	192.168.1.1	TCP	66 20 → 25265 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
192.168.1.1	192.168.1.2	TCP	66 25265 → 20 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
192.168.1.2	192.168.1.1	TCP	54 20 → 25265 [ACK] Seq=1 Ack=1 Win=65536 Len=0

+ Server trả lời đồng ý và bắt đầu gửi data (150), trong gói FTP-DATA chính là nội dung của file 3muis.txt.

192.168.1.2	192.168.1.1	FTP	126 Response: 150 Opening data channel for file download from server of "/3muis.txt"
192.168.1.2	192.168.1.1	FTP-DATA	75 FTP Data: 21 bytes (PORT) (RETR 3muis.txt)

```
FTP Data (21 bytes data)
[Setup frame: 71]
[Setup method: PORT]
[Command: RETR 3muis.txt]
Command frame: 73
[Current working directory: ]
Line-based text data (1 lines)
3muis la nhom hoc tap
```

+ Sau đó server gửi gói FIN,ACK (20 → 20407) báo rằng kết nối hoàn tất (việc truyền dữ liệu hoàn tất) và client trả lời ACK. Tiếp theo, server gửi tin báo quá trình truyền dữ liệu thành công và chính thức kết thúc (226), client trả lời ACK (1211 → 21) đã nhận được gói tin. Cuối cùng, client báo server sẽ đóng port 20407 (20407 → 20) này vào server trả lời ACK.

192.168.1.2	192.168.1.1	TCP	54 20 → 25264 [FIN, ACK] Seq=18 Ack=1 Win=65536 Len=0
192.168.1.1	192.168.1.2	TCP	54 25264 → 20 [ACK] Seq=1 Ack=19 Win=65536 Len=0
192.168.1.2	192.168.1.1	FTP	88 Response: 226 Successfully transferred "/"
192.168.1.1	192.168.1.2	TCP	54 1211 → 21 [ACK] Seq=56 Ack=166 Win=255 Len=0
192.168.1.1	192.168.1.2	TCP	54 25264 → 20 [FIN, ACK] Seq=1 Ack=19 Win=65536 Len=0
192.168.1.2	192.168.1.1	TCP	54 20 → 25264 [ACK] Seq=19 Ack=2 Win=65536 Len=0