

**BÁO CÁO ĐỒ ÁN SOCKET**

**XÂY DỰNG CHƯƠNG TRÌNH FTP CLIENT**

GIÁO VIÊN:

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN TP. HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**



SINH VIÊN – MSSV:

ĐỖ MINH HIẾU – 1612199

LÊ QUỐC DŨNG – 1612126

HOÀNG HÙNG ĐỨC - 1612117

TP. HỒ CHÍ MINH 01/06/2018

Mục lục

[**I.** **Phân công công việc:** 3](#_Toc515368997)

[1. Đỗ Minh Hiếu: 3](#_Toc515368998)

[2. Lê Quốc Dũng: 3](#_Toc515368999)

[3. Hoàng Hùng Đức: 3](#_Toc515369000)

[**II.** **Cấu trúc chương trình – Các hàm (phương thức) chính:** 3](#_Toc515369001)

[1. Cấu trúc chương trình: 3](#_Toc515369002)

[2. Các phương thức quan trọng: 4](#_Toc515369003)

[**III.** **Hướng dẫn sử dụng chương trình:** 10](#_Toc515369004)

[1. Kết nối và đăng nhập vào server: 10](#_Toc515369005)

[2. Liệt kê danh sách các thư mục, tệp tin dưới dạng đơn giản (ls), chi tiết (dir): 11](#_Toc515369006)

[3. Hiển thị đường dẫn hiện tại trên server (pwd): 12](#_Toc515369007)

[4. Thay đổi đường dẫn trên server (cd): 12](#_Toc515369008)

[5. Thay đổi đường dẫn dưới Client (lcd): 13](#_Toc515369009)

[6. Tạo thư mục trên server (mkdir): 13](#_Toc515369010)

[7. Xóa thư mục rỗng trên server (rmdir): 14](#_Toc515369011)

[8. Xóa một file trên server (delete): 14](#_Toc515369012)

[9. Xóa nhiều file trên server (mdelete): 15](#_Toc515369013)

[10. Chuyển sang chế độ active (active): 16](#_Toc515369014)

[11. Chuyển sang chế độ passive (passive): 16](#_Toc515369015)

[12. Tải lên một file đến server (put): 16](#_Toc515369016)

[13. Tải lên nhiều file đến server (mput): 17](#_Toc515369017)

[14. Tải xuống một file từ server (get): 17](#_Toc515369018)

[15. Tải xuống nhiều file từ server (mget): 17](#_Toc515369019)

[16. Thoát khỏi Server (quit): 18](#_Toc515369020)

[17. Hiện các lệnh hỗ trợ (help): 18](#_Toc515369021)

[**IV.** **Đánh giá mức độ hoàn thành:** 18](#_Toc515369022)

[**V.** **Mô tả quá trình gửi nhận dữ liệu khi download một file bằng WireShark:** 19](#_Toc515369023)

# **Phân công công việc:**

## Đỗ Minh Hiếu:

* Thiết lập socket truyền dữ liệu với 2 mode active và passive.
* Viết chức năng liệt kê chi tiết (dir) và liệt kê đơn giản (ls).
* Viết chức năng xóa 1 file (delete), nhiều file (mdelete).

## Lê Quốc Dũng:

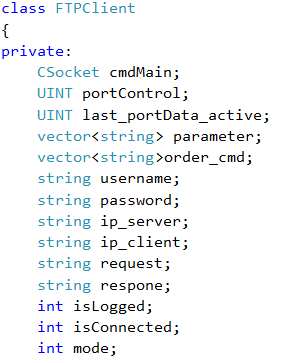
* Viết download 1 file (get), nhiều file từ server (mget).
* Viết upload 1 file (put), nhiều file (mput) từ local đến server.

## Hoàng Hùng Đức:

* Khởi tạo project, viết những thuộc tính, phương thức ban đầu của class.
* Viết kết nối socket điều khiển với server.
* Viết phương thức đăng nhập.

# **Cấu trúc chương trình – Các hàm (phương thức) chính:**

## Cấu trúc chương trình:

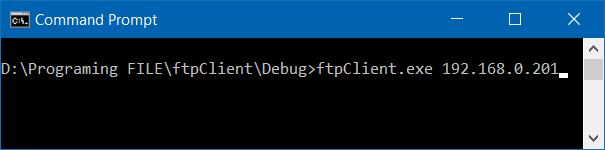
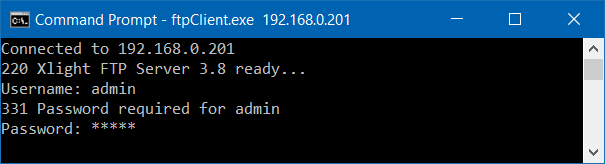
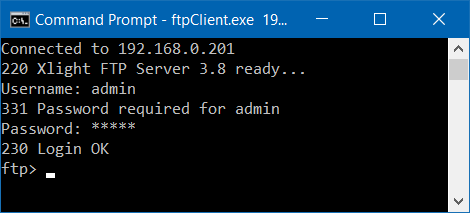
* Chương trình có 1 class là FTPClient chứa toàn bộ các chức năng cũng như giao diện(Interface) của chương trình ftp.
* Thuộc tính lớp FTPClient được tổ chức bao gồm:
* cmdMain: socket điều khiển.
* portControl: port của socket điều khiển.
* last\_portData\_active: port được socket Data sử dụng gần đây nhất, lúc khởi tạo là bằng portControl + 2.
* parameter: chứa tham số của các lệnh nhập vào.
* order\_cmd: lệnh trong console.
* username, password: chứa chuỗi tài khoản, mật khẩu nhập vào của người dùng.
* ip\_server, ip\_client: địa chỉ ip của server ftp và client.
* request, response: chuỗi chứa thông báo gửi đi (từ client) và nhận về (từ server).
* isLoged, isConnect: kiểm tra tình trạng đăng nhập, kết nối.
* mode: chế độ active(1) hoặc passive(0).

## Các phương thức quan trọng:

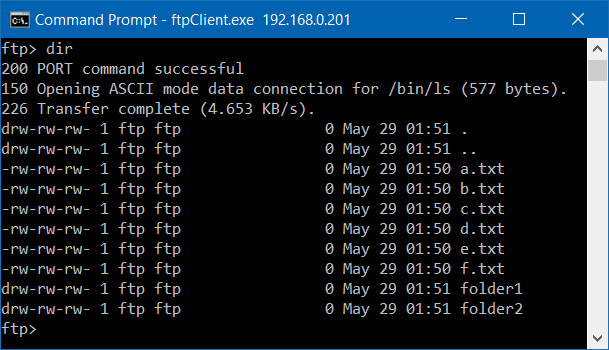
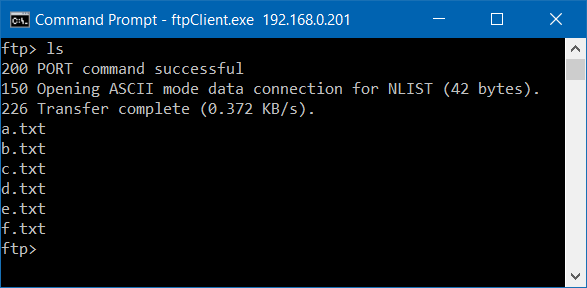
* Phương thức: int Login();
* Tham số truyền vào: không.
* Chức năng: nhập username, password ,gửi lệnh đăng nhập tới server và nhận trả lời từ server.
* Kết quả: thuộc tính username, password sẽ chứa tên tài khoản, mật khẩu nhập vào. Nếu đăng nhập thành công thì isLoged = true ngược lại bằng false. Xuất ra console thông báo đăng nhập thành công “230”.
* Trả về: isLoged.
* Phương thức: int Connect();
* Tham số truyền vào: không.
* Chức năng: Kết nối tới server ftp qua công 21 bằng địa chỉ IP của server.
* Kết quả: Nếu kết nối thành công, các thuộc tính ip\_client, portControl, last\_portData\_active được khởi tạo từ socket điều kiển cmdMain. Nếu kết nối thành công thì isConnected bằng true ngược lại bằng false. Xuất ra console thông báo kết nối thành công “220”.
* Trả về: isConnected.
* Phương thức: Csocket\*Act\_Mode();
* Tham số truyền vào: không.
* Chức năng: Khởi tạo socket truyền dữ liệu theo chế độ active.
* Kết quả: Nếu lệnh trả về khác 200 thì socket không được tạo thành công. Sau khi gọi phương thức thì sẽ đặt kênh truyền và last\_portData\_active cộng thêm 1. Dữ liệu sẽ được truyền trên socket này. Xuất ra console thông báo mở port thành công “200”.
* Trả về: Con trỏ CSocket\* đến địa chỉ của socket truyền dữ liệu.
* Phương thức: CSocket\*Pass\_Mode();
* Tham số truyền vào: không.
* Chức năng: Khởi tạo socket truyền dữ liệu theo chế độ passive.
* Kết quả: Sau khi gọi hàm, con trỏ hứng kết quả trả về sẽ được dùng để nhận dữ liệu. Xuất ra console thông báo mở được passive mode và port server “227”.
* Trả về: Con trỏ CSocket\* đến địa chỉ của socket truyền dữ liệu.
* Phương thức: void splitOrderString(string);
* Tham số truyền vào: chuỗi string – chứa chuỗi lệnh và tham số người dùng nhập vào trên console.
* Chức năng: Tách các tham số nhập vào thành các chuỗi và lưu vào vector<string> parameter.
* Kết quả: Mỗi tham số nhập vào đều lưu vào 1 string trong vector.
* Trả về: Không.
* Phương thức: int find\_pos\_order(string);
* Tham số truyền vào: chuỗi string chỉ chứa lệnh của người dùng nhập vào trên console.
* Chức năng: Tìm vị trí của chuỗi lệnh trong vector<string> order\_cmd dùng trong menu switch case.
* Kết quả: Có được vị trí của lệnh.
* Trả về: vị trí của lệnh trong vector các chuỗi lệnh. Nếu không tìm được trả về -1.
* Phương thức: void cmd\_dir();
* Tham số truyền vào: không.
* Chức năng: Dùng socket điều khiển gửi lệnh LIST và nhận về danh sách tệp dưới dạng chuỗi thông qua socket truyền dữ liệu.
* Kết quả: Hiển thị trên console danh sách các tệp, thư mục gồm các thông tin: tên, ngày sửa đổi, giờ sửa đổi, dung lượng. Xuất ra thông báo mở kênh truyền “150” và truyền thành công “226”.
* Trả về: không.
* Phương thức: void cmd\_ls();
* Tham số truyền vào: không.
* Chức năng: Dùng socket điều khiển gửi lệnh NLST và nhận về danh sách tệp dưới dạng chuỗi thông qua socket truyền dữ liệu.
* Kết quả: Hiển thị trên console tên các tệp tin. Xuất ra thông báo mở kênh truyền “150” và truyền thành công “226”.
* Trả về: không.
* Phương thức: void cmd\_del(string);
* Tham số truyền vào: chuỗi string là tên file cần xóa.
* Chức năng: Gửi lệnh DELE cùng tên file tới server để xóa file ở server.
* Kết quả: file được chọn bị xóa khỏi server. Xuất ra console thông báo xóa thành công file “250” ngược lại xuất ra không tìm thấy “530”.
* Trả về: không.
* Phương thức: void cmd\_mdele();
* Tham số truyền vào: không.
* Chức năng: Xóa các file ở trên server. Tên các file được lưu trong parameter. Mỗi lần xóa sẽ hỏi có muốn xóa file không.
* Kết quả: Các file được chọn và đồng ý xóa sẽ bị xóa khỏi server. Xuất ra console thông báo xóa thành công file “250” ngược lại xuất ra không tìm thấy “530”.
* Trả về: không.
* Phương thức: void cmd\_pwd();
* Tham số truyền vào: không.
* Chức năng: Lấy đường dẫn hiện tại trên server bằng lệnh PWD.
* Kết quả: Xuất ra console đường dẫn đang đứng trên server với thông báo “257”.
* Trả về: không.
* Phương thức: void cmd\_mkdir(string);
* Tham số truyền vào: chuỗi string là tên của thư mục hoặc đường dẫn của thư mục cần tạo.
* Chức năng: Tạo một thư mục rỗng trên server.
* Kết quả: Trên server có 1 thư mục rỗng mới với tên được truyền vào ở trên. Xuất ra thông báo “257” tạo thư mục thành công.
* Trả về: không.
* Phương thức: void cmd\_rmdir(string);
* Tham số truyền vào: chuỗi string là tên thư mục cần xóa.
* Chức năng: xóa thư mục rỗng.
* Kết quả: Xóa thư mục rỗng trên server. Xuất ra thông báo “250” xóa thành công ngược lại “530” thư mục không rỗng hoặc không tồn tại.
* Trả về: không.
* Phương thức: void cmd\_cd(string);
* Tham số truyền vào: chuỗi string là đường dẫn mới.
* Chức năng: Thay đổi đường dẫn trên server để thực hiện các thao tác với thư mục file ở đó.
* Kết quả: Xuất thông báo thay đổi thành công “250” ngược lại không tìm thấy “550”.
* Trả về: không.
* Phương thức: void cmd\_get\_once\_File(string);
* Tham số truyền vào: chuỗi string là tên file cần download.
* Chức năng: Download một file từ server bằng parameter truyền vào.
* Kết quả: Xuất thông báo mở kênh truyền “150”, download thành công “226”. Ngược lại xuất ra lỗi hoặc không tồn tại file.
* Trả về: không.
* Phương thức: void cmd\_get();
* Là phương thức download một file – phương thức bao bọc bên ngoài của cmd\_get\_once\_File.
* Phương thức: void cmd\_mget();
* Tham số truyền vào: không.
* Chức năng: Liệt kê các tên các file nhập vào lưu trong parameter và xác thực đồng ý xóa chúng.
* Kết quả: Xuất thông báo mở kênh truyền “150”, download thành công “226”. Ngược lại xuất ra lỗi hoặc không tồn tại file.
* Trả về: không.
* Phương thức: void cmd\_lcd();
* Tham số truyền vào: không.
* Chức năng: Thay đổi đường dẫn làm việc dưới client.
* Kết quả: Thông báo đường dẫn mới “Local directory now …” hoặc thay đổi thất bại “Set current directory false.”
* Trả về: không.
* Phương thức: bool cmd\_put\_once\_File(string);
* Tham số truyền vào: chuỗi string là tên file cần upload lên server.
* Chức năng: Upload một file lên server.
* Kết quả: Thông báo mở kênh truyền “150” và upload thành công “226”. Ngược lại xuất ra lỗi hoặc không tồn tại file.
* Trả về: true (upload thành công) hoặc false (upload không thành công).
* Phương thức: void cmd\_put();
* Là phương thức upload 1 file – phương thức bao bọc bên ngoài của phương thức cmd\_put\_once\_File.
* Phương thức: void cmd\_mput();
* Tham số truyền vào: không
* Chức năng: Xác thực có muốn upload từng file không và upload nó lên server.
* Kết quả: Thông báo mở kênh truyền “150” và upload thành công “226”. Ngược lại xuất ra lỗi hoặc không tồn tại file.
* Trả về: không.
* Phương thức: void cmd\_quit();
* Tham số truyền vào: không
* Chức năng: Thoát hẳn khỏi chương trình FTP Client.
* Kết quả: Xuất thông báo thoát “221”
* Trả về: không.
* Phương thức: void cmd\_open();
* Tham số truyền vào: không.
* Chức năng: tạo lại kết nối mới với địa chỉ IP server và username, password mới được nhập vào.
* Kết quả: Xuất ra các thông báo của phương thức Login và Connect.
* Trả về: không.

# **Hướng dẫn sử dụng chương trình:**

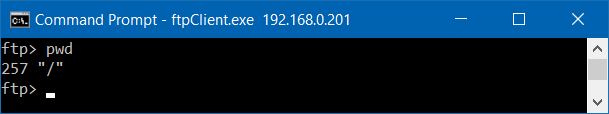
## Kết nối và đăng nhập vào server:

* Mở command line và chuyển đến vị trí chưa file ftpClient.exe bằng lệnh cd.
* Cú pháp: ftpClient.exe www.xxx.yyy.zzz
* Trong đó: ftpClient.exe: là tên file thực thi chương trình, www.xxx.yyy.zzz: địa chỉ IP của server.
* Nhấn enter để thực thi.
* Màn hình kết nối thành công với server và đăng nhập xuất hiện. Nhập tài khoản, mật khẩu khi được yêu cầu và nhấn enter để thực thi.
* Tại đây, khi kết nối và đăng nhập thành công, ta có thể thực hiện các lệnh của ftp.

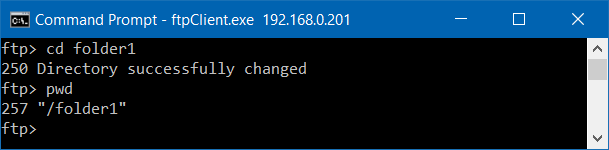
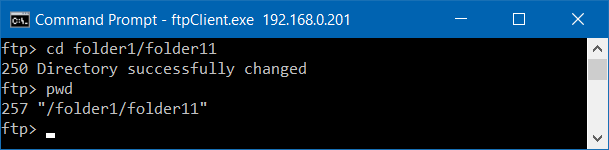
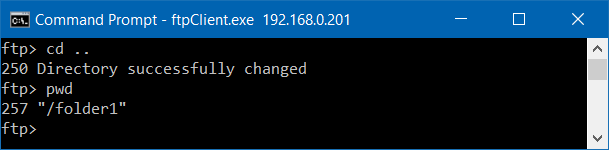
## Liệt kê danh sách các thư mục, tệp tin dưới dạng đơn giản (ls), chi tiết (dir):

* Liệt kê đơn giản: Cú pháp: ls
* Nhấn enter.
* Tương tự như lệnh ls, lệnh dir liệt kê chi tiết các file và thư mục.

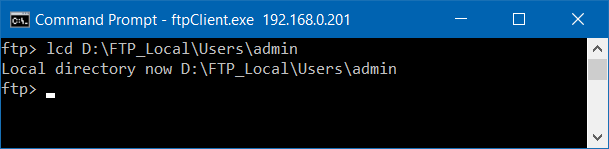
## Hiển thị đường dẫn hiện tại trên server (pwd):

* Cú pháp: pwd và nhấn enter.
* Như ta thấy, đường dẫn hiện tại là “/” tức là root ở server.

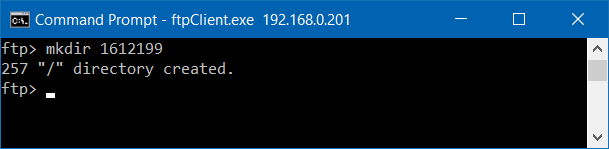
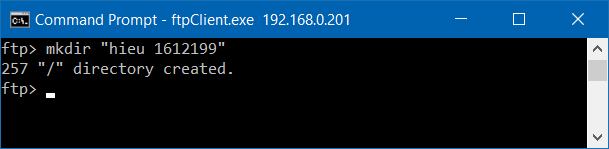
## Thay đổi đường dẫn trên server (cd):

* Cú pháp: cd <đường dẫn mới>
* Nhấn enter.
* Nếu từ thư mục cha, muốn vào thư mục con của nó thì chỉ việc gõ tên thư mục con. Ví dụ ở đây là root, muốn vào thư mục folder1 thì chỉ cần dùng lệnh cd folder1.
* Nếu muốn từ thư mục root vào thư mục con của thư mục folder1 thì làm như sau:
* Nếu muốn trở về thư mục cha của thư mục hiện tại thì làm:

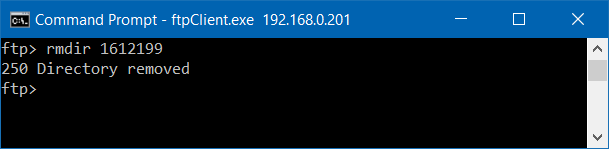
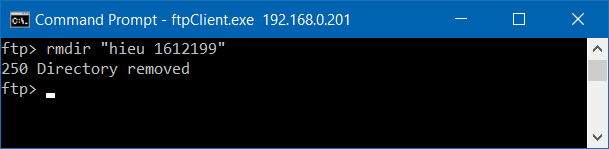
## Thay đổi đường dẫn dưới Client (lcd):

* Cú pháp: lcd <đường dẫn ở máy client>
* Ví dụ: Muốn thay đổi đường dẫn ở Client thành D:\FTP\_Local\Users\admin

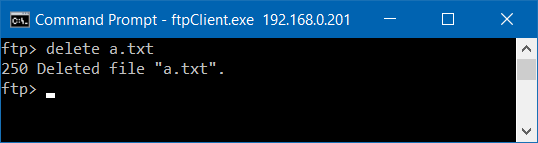
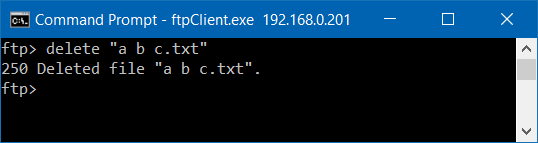
## Tạo thư mục trên server (mkdir):

* Cú pháp: mkdir <tên thư mục>
* Ví dụ tạo thư mục tên 1612199
* Lưu ý: Nếu tên thư mục muốn tạo có khoảng trắng thì phải đặt trong dấu nháy kép (block quote). Ví dụ tạo thư mục “hieu 1612199”.

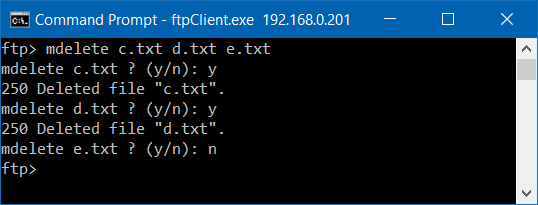
## Xóa thư mục rỗng trên server (rmdir):

* Cú pháp: rmdir <tên thư mục>
* Ví dụ xóa thư mục tên 1612199
* Lưu ý: Tương tự như tạo thư mục có khoảng trắng, khi xóa thư mục có khoảng trắng thì phải đặt tên thư mục trong dấu nháy kép.

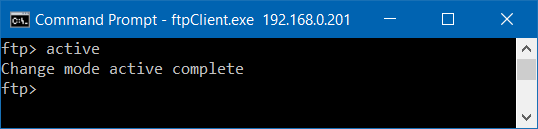
## Xóa một file trên server (delete):

* Cú pháp: delete <tên file cần xóa>
* Ví dụ xóa file a.txt
* Lưu ý: khi xóa file mà tên file có khoảng trắng thì phải đặt trong dấu nháy kép. Ví dụ xóa file “a b c.txt”

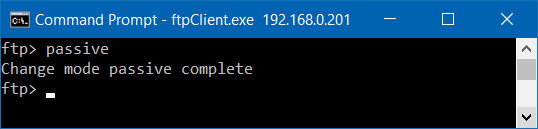
## Xóa nhiều file trên server (mdelete):

* Cú pháp: mdelete <tên file 1> <tên file 2> <tên file n>
* Sau đó xuất hiện thông báo muốn xóa hay không 1 file. Nhấn y để xóa hoặc n để không xóa.
* File xóa thành công sẽ xuất hiện thông báo xóa thành công mã “250”.
* Ví dụ xóa các file: c.txt, d.txt, e.txt
* Lưu ý: tương tự như xóa 1 file, nếu tên file có khoảng trắng thì phải đặt trong dấu nháy kép. Ví dụ xóa 2 file “a a a.txt” và “b b b.txt”.

## Chuyển sang chế độ active (active):

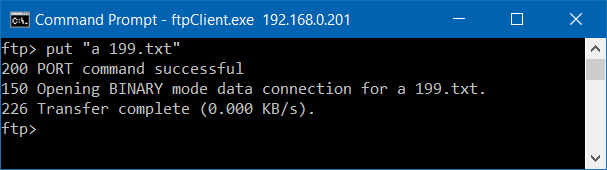
* Cú pháp: active

## Chuyển sang chế độ passive (passive):

* Cú pháp: passive

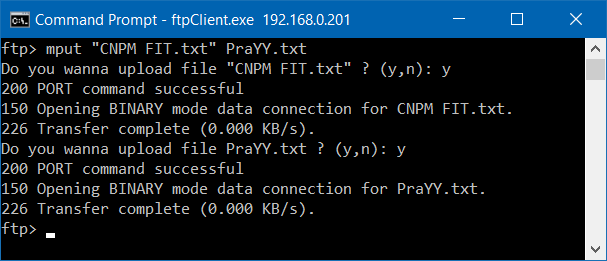
## Tải lên một file đến server (put):

* Cú pháp: put <tên file tải lên>
* Nếu tải lên thành công sẽ xuất ra thông báo mã “226”.
* Lưu ý: nếu tên file có khoảng trắng thì phải đặt trong dấu nháy kép.

Ví dụ tải lên file: “a 199.txt”

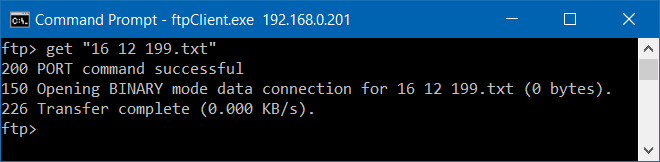
## Tải lên nhiều file đến server (mput):

* Cú pháp: mput <tên file 1> <tên file 2> <tên file n>
* Lưu ý: giống như tải lên 1 file, tên file nếu có khoảng trắng thì phải đặt trong dấu nháy kép.

Ví dụ tải lên các file sau: “CNPM FIT.txt”, PraYY.txt

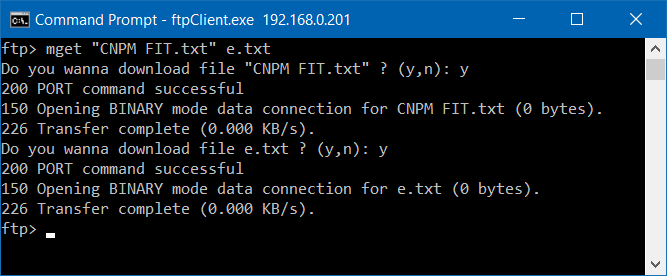
## Tải xuống một file từ server (get):

* Cú pháp: get <tên file cần tải>
* Nếu tải thành công sẽ có thông báo mã “226”.
* Lưu ý: nếu tên file có khoảng trắng thì phải đặt trong dấu nháy kép.

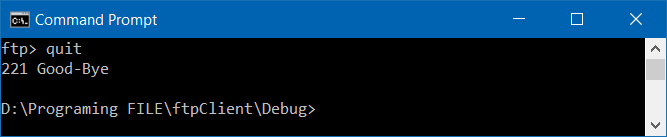
Ví dụ tải xuống file: “16 12 199.txt”

## Tải xuống nhiều file từ server (mget):

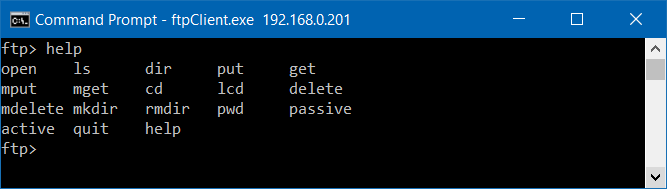
* Cú pháp: mget <tên file 1> <tên file 2> <tên file n>
* Lưu ý: nếu tên file có khoảng trắng thì đặt trong dấu nháy kép.

Ví dụ tải xuống các file: “CNPM FIT.txt”, e.txt

## Thoát khỏi Server (quit):

* Cú pháp: quit

## Hiện các lệnh hỗ trợ (help):

* Cú pháp: help

# **Đánh giá mức độ hoàn thành:**

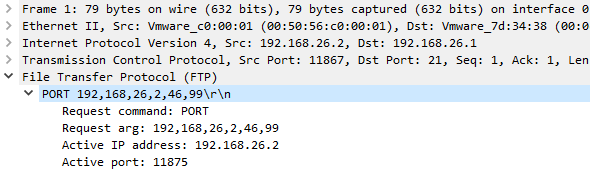
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Chức năng | login | ls | dir | put | get | mput | mget | cd | lcd |
| Mức độ hoàn thành | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Chức năng | delete | mdelete | mkdir | rmdir | pwd | passive | active | quit |
| Mức độ hoàn thành | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% |

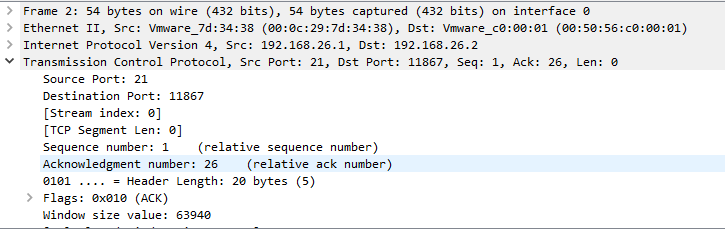
# **Mô tả quá trình gửi nhận dữ liệu khi download một file bằng WireShark:**

**Đối với active-mode:**

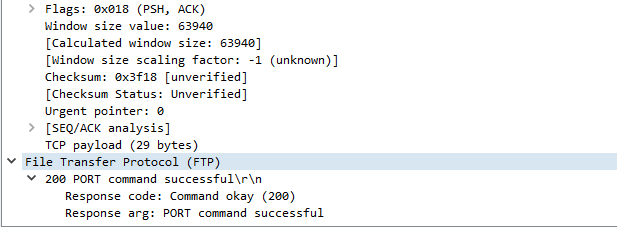
-Client gửi lệnh PORT kèm theo cổng được mở để nhận kết nối.



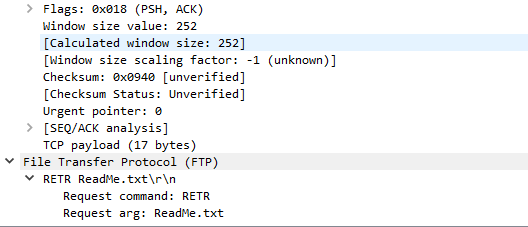
-Server gửi gói tin có cờ ACK để xác nhận rằng đã nhận được request PORT từ Client



Đồng thời Server cũng gửi respone về Client dưới dạng Messeage thông báo đã chấp nhận port

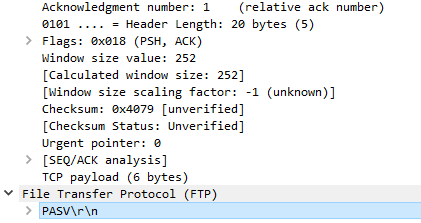


-Client sau khi nhận được respone với mã 200 thì gửi đi request:

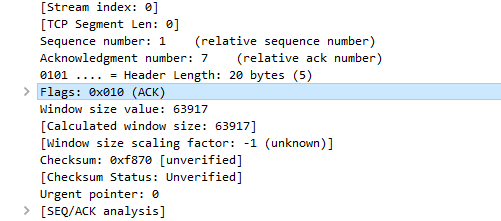


**Đối với chế độ Passive-Mode**

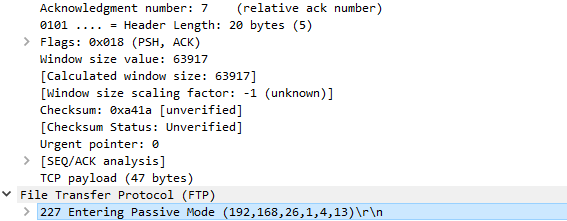
-Client gửi request với message PASV



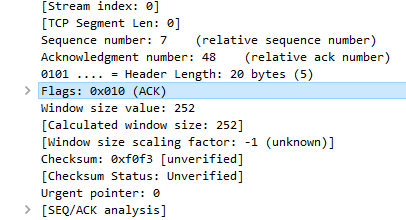
-Server gửi packet với cờ ACK để xác nhận



-Server gửi message respone về client với port được mở sẵn để chờ kết nối



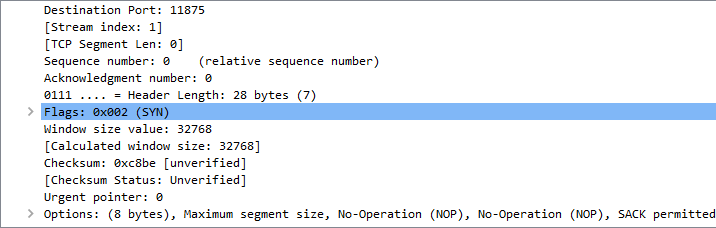
-Client gửi packet với cờ ACK để xác nhận



**Quá trình mở connection và transfer-data**

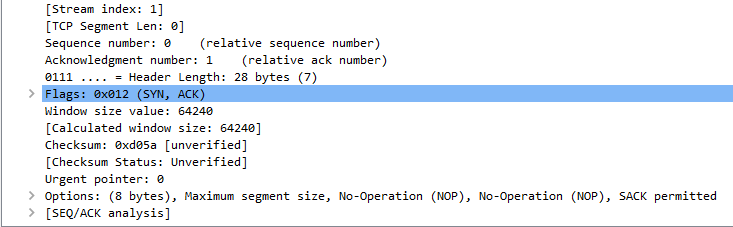
-Server yêu cầu mở Connection với Client – gói tin có cờ SYN

(đối với passive-mode thì client yêu cầu mở Connection với server)

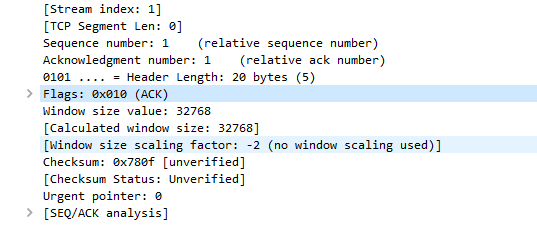


-Client nhận được yêu cầu mở Connection, Client sẽ gửi lại packet chấp nhận với 2 bit cờ là SYN và ACK

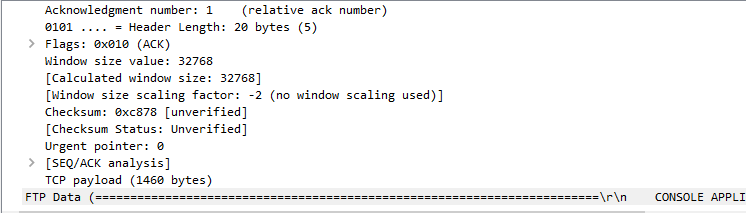
SYN/ACK packet được gửi ngược lại bằng cách đổi hai IP của server và client, server IP sẽ thành IP đích và client IP sẽ thành IP bắt đầu. Tương tự như vậy, cổng cũng sẽ thay đổi, client nhận được packet ở cổng nào thì cũng sẽ dùng cổng đó để gửi lại packet vào cổng mà server đã gửi.  
Client gửi lại packet này để thông báo là client đã nhận được tín hiệu và chấp nhận connection, trong trường hợp client không chấp nhận connection, thay vì SYN/ACK bits được bật, client sẽ bật bit RST/ACK (Reset Acknowledgement) và gởi ngược lại RST/ACK packet.  
Client bắt buộc phải gửi thông báo lại bởi vì TCP là chuẩn tin cậy nên nếu server không nhận được thông báo thì sẽ nghĩ rằng packet đã bị lạc và gửi lại thông báo mới.

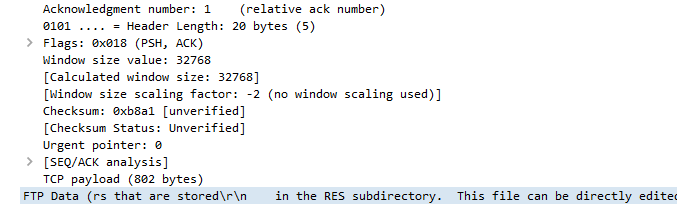


-Khi server nhận được SYN/ACK packet thì sẽ trả lời bằng ACK packet. Packet này được gởi với mục đích duy báo cho client biết rằng server đã nhận được SYN/ACK packet và lúc này connection đã được thiết lập và dữ liệu sẽ bắt đầu lưu thông tự do.  
Đây là tiến trình bắt buộc phải thực hiện khi client muốn trao đổi dữ liệu với server thông qua giao thức TCP.

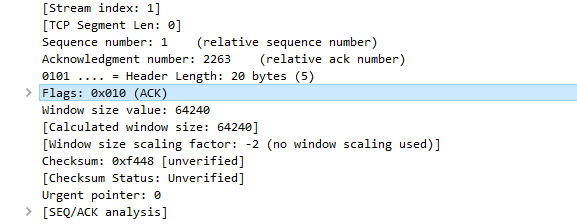


-Tiếp theo là quá trình truyền dữ liệu từ server về client

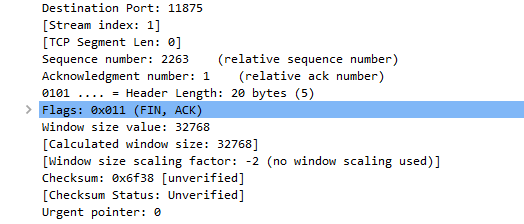




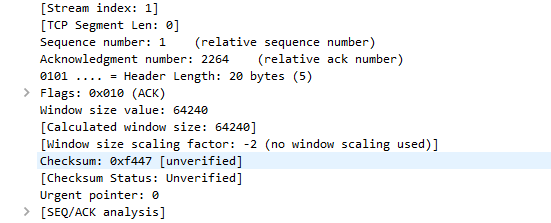
-Client gửi packet có cờ ACK để xác nhận đã nhận thành công data

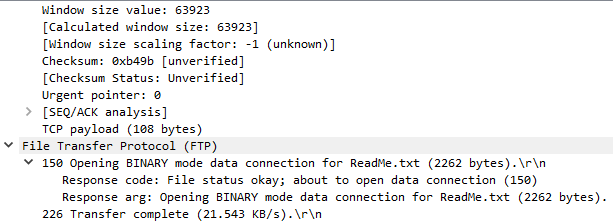


-Server gửi packet có 2 bit cờ là FIN và ACK kết thúc quá trình truyền data, đóng connection

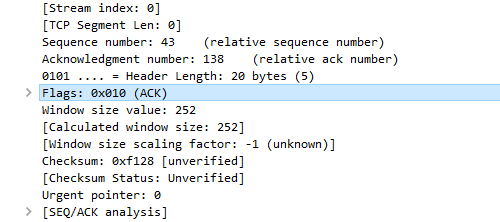


-Client gửi packet có bit cờ ACK để xác nhận

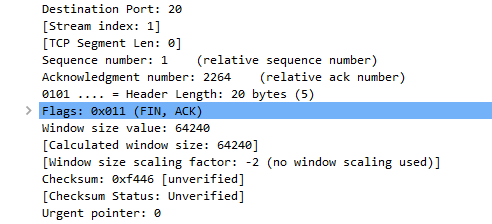


-Server gửi respone trả về mã 150 mở dưới dạng nhị phân 

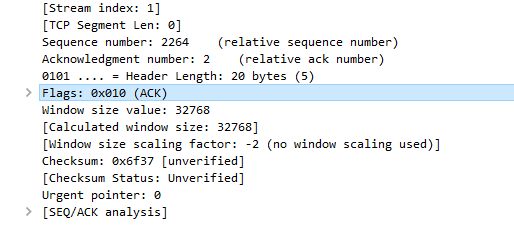
-Client gửi packet xác nhận



-Client gửi packet có 2 bit cờ FIN và ACK để yêu cầu đóng Connection



-Server gửi packet có cờ ACK để xác nhận



Quá trình kết thúc là 2 quá trình kết thúc 1 chiều độc lập nhau. Khi một bên muốn kết thúc nó sẽ gửi 1 cờ FIN và bên còn lại sẽ gửi lại tin báo nhận bằng cờ ACK. Khi đó kết nối tồn tại dạng "nửa mở" một bên đã kết thúc việc gửi dữ liệu nên chỉ nhận trong khi bên khi bên kia vẫn tiếp tục gửi. Để kết nối đóng hoàn toàn, bên còn lại cũng phải làm tương tự tức là gửi 1 cờ FIN cho bên kia và chờ nhận 1 cờ ACK.