**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**



**BÁO CÁO BÀI TẬP LỚN MẠNG MÁY TÍNH**

*Đề tài:*

**Tìm hiểu hoạt động và cài đặt dịch vụ FTP**

Sinh viên thực hiện: Nguyễn Huy Định

Nguyễn Vĩnh Thụy

Nguyễn Ngọc Đức

Giảng viên hướng dẫn: Th.s Bùi Trọng Tùng

*Hà Nội - 2018*

**MỤC LỤC**

[**Phần 1. Giao thức FTP** 3](#_Toc514967074)

[**I.** **Tổng quan** 3](#_Toc514967075)

[**II.** **Hoạt động của giao thức FTP** 4](#_Toc514967076)

[1. Mô hình: 4](#_Toc514967077)

[2. Hoạt động của giao thức: 5](#_Toc514967078)

[**III.** **Các kiểu dữ liệu khi truyền dữ liệu** 7](#_Toc514967079)

[1. Kiểu ASCII 7](#_Toc514967080)

[2. Kiểu EBCDIC 8](#_Toc514967081)

[3. Kiểu Image (nhị phân) 8](#_Toc514967082)

[4. Kiểu Local 8](#_Toc514967083)

[**IV.** **Cấu trúc dữ liệu** 8](#_Toc514967084)

[**V.** **Các phương thức truyền dữ liệu** 9](#_Toc514967085)

[1. Stream mode 9](#_Toc514967086)

[2. Block mode 10](#_Toc514967087)

[3. Compressed mode 11](#_Toc514967088)

[**VI.** **Lệnh và mã hồi đáp trong FTP** 12](#_Toc514967089)

[1. Các lệnh FTP (FTP commands) 12](#_Toc514967090)

[2. Mã hồi đáp (Reply code) 16](#_Toc514967091)

[**Phần 2. Cài đặt FTP Server và viết chương trình FTP Client** 19](#_Toc514967092)

[**I.** **Cài đặt FTP Server** 19](#_Toc514967093)

[**II.** **Chương trình FTP Client** 22](#_Toc514967094)

[***Danh mục tài liệu tham khảo:*** 25](#_Toc514967095)

# **Phần 1. Giao thức FTP**

# **Tổng quan**

FTP: File Transfer Protocol dịch ra là giao thức truyền tập tin. FTP cũng giống như các giao thức khác của tầng ứng dụng, nó hoạt động trên kiến trúc TCP/IP và chỉ chạy trên nền của TCP. Sự xuất hiện của FTP đã làm cho việc tải dữ liệu trở nên tin cậy và có hiệu quả cao, nó che đậy sự khác biệt về hệ thống lưu trữ tập tin giữa các máy chủ giải quyết được vấn đề khác nhau giữa các hệ thống, do đó người sử dụng không cần quan tâm đến những sự khác biệt này.

FTP sử dụng mô hình client-server, thuật ngữ client có thể được hiểu như user (người sử dụng) do phía client, người sử dụng là đối tượng sử dụng trực tiếp. Máy chủ FTP có vai trò cung cấp dịch vụ, lắng nghe và chấp nhận các yêu cầu kết nối, yêu cầu về điều khiển từ các clients. Mỗi bên đều có vùng lưu trữ file riêng, phía client có thể thao tác với các tệp tin trên server: tải tệp tin, xóa tệp tin,... Khác với giao thức tầng ứng dụng khác, FTP sử dụng 2 cổng kết nối TCP khác nhau, một kết nối để truyền các lệnh điều khiển, kết nối còn lại dùng để truyền dữ liệu. Cổng sử dụng thường là 20 và 21 ứng với 2 kết nối trên, tuy nhiên kết nối truyền dữ liệu có thể được sử dụng cổng khác bất kì thay vì 20.

* Lịch sử phát triển

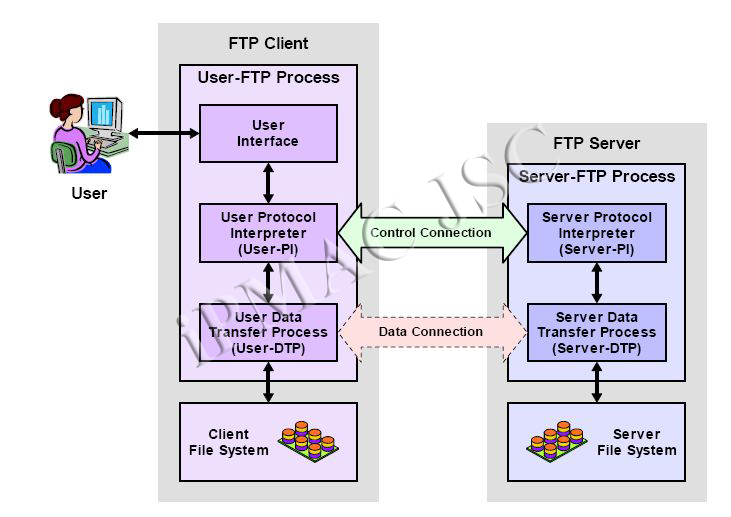
FTP có sự cải tiến lâu dài trong nhiều năm, các cải tiến được đặc tả trong các RFC. FTP xuất hiện đầu tiên vào năm 1971 được chạy thử nghiệm trên hệ thống máy tính của hãng MIT (RFC 114). RFC 172 cung cấp một giao thức định hướng mức độ người dùng cho việc truyền tập tin giữa các máy. RFC 265 khôi phục lại FTP để xem xét trong việc bổ sung, trong khi RFC 281 đề xuất thêm các thay đổi. Việc sử dụng “Set Data Type” được đề xuất trong RFC 294 vào tháng 1 năm 1982.

Sự xuất hiện của ARPANET thời kì đầu 1971-1974 đã thay đổi định nghĩa FTP, FTP giờ đây được định nghĩa như là một giao thức cho việc truyền file giữa các HOSTs trên mạng ARPANET, với chức năng chính của FTP là truyền tệp tin một cách hiệu quả và đáng tin cậy giữa các máy; cho phép sử dụng thuận tiện các tệp tin lưu trữ từ xa. RFC 385 bình luận nhiều hơn về các lỗi, những điểm quan trọng và các bổ sung cho giao thức, trong khi RFC 414 cung cấp báo cáo trạng thái của máy chủ khi hoạt động và phía người dùng. RFC 430 bổ sung thêm cho giao thức. Cuối cùng tài liệu FTP chính thức được xuất bản thành RFC 454.

Tháng 7/1973, có sự thay đổi đáng kể từ các bản trước của FTP nhưng cấu trúc tổng thể thì vẫn giữ nguyên. RFC 542 được xuất bản, đặc tả về những thay đổi này. Các bản RFC tiếp theo: 607, 614, 624, 686, 691 tiếp tục bình luận về FTP. RFC 765 là bản đặc tả nói về việc sử dụng FTP trên nền giao thức TCP. RFC 959 khắc phục một vài lỗi nhỏ về tài liệu, giải thích một số tính năng và thêm một số lệnh tùy chọn mới. RFC 959 là bản đặc tả cuối cùng nói về giao thức FTP, được xuất bản vào năm 1985.

# **Hoạt động của giao thức FTP**

## Mô hình:



*Mô hình giao thức FTP*

FTP sử dụng mô hình client-server, mỗi bên đều có thành phần quản lý 2 kênh truyền tương ứng. FTP sử dụng 2 kết nối TCP tương ứng với 2 kênh điều khiển và kênh dữ liệu. Kênh điều khiển chịu trách nhiệm truyền các tín hiệu điều khiển phía client tới server và hồi đáp từ server với các yêu cầu từ client, quá trình kết nối tới server cũng được truyền đi trên kết nối TCP này. Kênh dữ liệu chịu trách nhiệm truyền dữ liệu tệp tin khi có yêu cầu truyền dữ liệu qua các lệnh trên kênh điều khiển. Điều này làm cho FTP khác các giao thức khác, nó không truyền trực tiếp dữ liệu file và các lệnh FTP trên cùng một kết nối, chỉ khi nào có yêu cầu truyền dữ liệu file hoặc một số dữ liệu khác (lệnh LIST yêu cầu danh sách thuộc tính của các file) thì kết nối dữ liệu mới được mở.

Bên phía client, User-PI chịu trách nhiệm gửi các lệnh mà người dùng yêu cầu tới Server-PI qua kết nối điều khiển, khi một phiên làm việc FTP bắt đầu, User-PI sẽ gửi yêu cầu thiết lập kết nối TCP tới Server qua cổng 21. User-PI cũng quản lý User-DTP trong quá trình truyền dữ liệu. Tiến trình Server-PI sẽ lắng nghe các kết nối này và phản hồi lại các mã hồi đáp cho clients. Các lệnh FTP được gửi và hồi đáp trong suốt quá trình kết nối giữa client và server.

Tiến trình User-DTP bên phía client và Server-DTP bên phía server chịu trách nhiệm quản lý kết nối dữ liệu. Các tiến trình này tương ứng sẽ gửi hoặc nhận dữ liệu file trong hệ thống file system mỗi bên. Mặc định User-DTP sẽ lắng nghe kết nối dữ liệu. Mỗi khi dữ liệu được truyền xong, kết nối này sẽ bị đóng và được mở lại khi có yêu cầu trao đổi dữ liệu file. Tiến trình User Interface cung cấp giao diện cho người dùng thao tác và yêu cầu, các yêu cầu này sẽ được gửi đi bằng lệnh FTP thông qua User-PI.

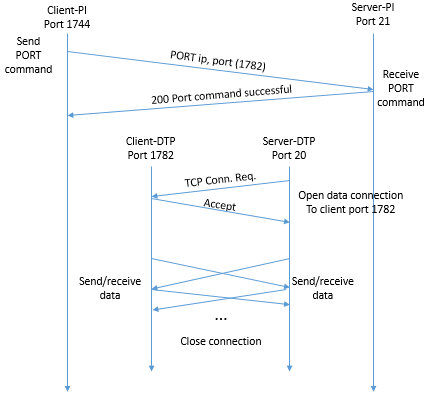
## Hoạt động của giao thức:

Khi một phiên làm việc của FTP bắt đầu, phía client sẽ gửi yêu cầu thiết lập kết nối TCP (số hiệu cổng bất kì) tới server, server lắng nghe trên cổng 21. Khi yêu cầu kết nối được server chấp nhận, người dùng sẽ gửi thông tin username và password. Server sẽ kiểm tra và hồi đáp, nếu thông tin về người dùng là đúng thì người dùng có thể gửi các yêu cầu thông qua các lệnh FTP. Khi có một yêu cầu trao đổi dữ liệu, kết nối dữ liệu sẽ được mở, mặc định server sẽ gửi yêu cầu kết nối dữ liệu, sử dụng cổng 20. User-DTP sẽ phải lắng nghe kết nối này ở cổng đã được xác định trước trong lệnh FTP. Kết nối này sẽ được đóng lại khi dữ liệu được truyền xong, trong phiên làm việc đó nếu có yêu cầu thì kết nối này lại được khởi tạo, còn kết nối điều khiển sẽ tồn tại cho đến khi kết thúc phiên làm việc.

Giao thức FTP hỗ trợ 2 kiểu kết nối kênh dữ liệu: chủ động (active) và bị động (passive). Sự khác biệt của 2 kiểu kết nối này là bên nào sẽ lắng nghe yêu cầu kết nối trên kết nối dữ liệu.

1. *Kết nối dữ liệu kiểu chủ động*

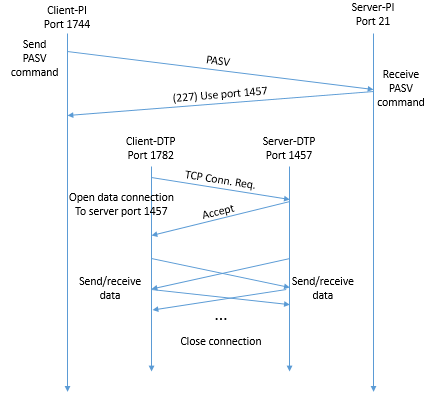
Trong kết nối kiểu chủ động, phía client sẽ lắng nghe kết nối, server sẽ là bên yêu cầu kết nối. Sau khi yêu cầu trao đổi dữ liệu file, client gửi lệnh PORT mang thông tin địa chỉ ip của server và số hiệu cổng mà Client-DTP sẽ lắng nghe, ở hình dưới là 1782. Số hiệu cổng này khác với số hiệu cổng của Client-PI. Server-DTP sẽ mở một kết nối TCP sử dụng cổng 20 kết nối tới cổng 1782 trên Client-DTP. Khi kết nối được được chấp nhận, quá trình trao đổi dữ liệu bắt đầu. Kết thúc quá trình trao đổi dữ liệu, kết nối sẽ bị đóng lại. Lệnh PORT có thể không được gửi đi, khi đó mặc định server sẽ khởi tạo kết nối tới Client-DTP với số hiệu cổng trùng với Client-PI (tức là số hiệu cổng của Client-DTP cũng là 1744). Trong kết nối kiểu chủ động, cổng 20 của server là mặc định.



*Kết nối dữ liệu kiểu chủ động*

1. *Kết nối dữ liệu kiểu bị động*

Trong kiểu kết nối này, phía client sẽ gửi yêu cầu kết nối, server là bên lắng nghe kết nối. Client gửi lệnh PASV, lệnh này để báo cho server là sử dụng kiểu bị động, server sẽ hồi đáp thành công bằng thông điệp trả lời chứa địa chỉ ip mà nó lắng nghe và số hiệu cổng (ở hình trên là 1457). Client-DTP sẽ khởi tạo kết nối sử dụng số hiệu cổng bất kì (ở hình dưới là cổng 1782) kết nối tới Server-DTP. Sau khi kết nối thành công, quá trình truyền dữ liệu bắt đầu và đóng kết nối khi kết thúc truyền dữ liệu. Ở kết nối kiểu bị động, số hiệu cổng của server không phải là số hiệu cổng mặc định. FTP hỗ trợ kết hợp cả 2 lệnh PASV và PORT để sử dụng các cổng không mặc định: lệnh PORT thay đổi cổng mặc định (trùng với client-PI) của client, lệnh PASV thay đổi cổng mặc định (cổng 20) của server.



*Kết nối dữ liệu kiểu bị động*

# **Các kiểu dữ liệu khi truyền dữ liệu**

Dữ liệu được truyền từ thiết bị lưu trữ trên máy chủ đến thiết bị lưu trữ trên máy khách. Cần có các biến đổi trên dữ liệu vì các cách biểu diễn dữ liệu trên hai hệ thống là khác nhau. Tệp tin sẽ được chuyển đổi thành các kiểu dữ liệu trung gian để truyền đi và được chuyển đổi về kiểu dữ liệu cục bộ tại máy nhận. Client sẽ thông báo cho server biết kiểu dữ liệu được sử dụng qua lệnh TYPE, mặc định sẽ là kiểu ASCII.

## Kiểu ASCII

Đây là kiểu mặc định và được sử dụng khi truyền các tệp tin văn bản. Mỗi kí tự đều được gửi đi dưới dạng mã ASCII. Máy gửi sẽ chuyển đổi dữ liệu sang dạng ASCII, máy nhận sẽ chuyển đổi từ mã ASCII sang dạng thích hợp. Các chuyển đổi này bao gồm việc thay thế mã kết thúc dòng và kết thúc tệp tin. Các tệp tin dạng ASCII luôn chứa CR và LF cuối mỗi dòng. Ví dụ khi gửi một tệp tin kiểu ASCII từ máy chủ đến máy khách chạy hệ điều hành Unix thì máy khách sẽ loại bỏ kí tự LF cuối mỗi dòng, chỉ để lại CR.

## Kiểu EBCDIC

EBCDIC là một bảng mã tương tự ASCII do công ty IBM phát triển, mỗi ký tự được biểu diễn với 8 bit. Kiểu EBCDIC được thiết kế để phù hợp với các máy sử dụng EBCDIC để biểu diễn ký tự. Dữ liệu được biểu diễn dưới dạng các ký tự EBCDIC 8 bit. Cuối mỗi dòng sẽ chứa ký tự NL.

## Kiểu Image (nhị phân)

Dữ liệu được truyền dưới dạng các bit nhị phân liền kề nhau, được đóng gói thành các byte 8 bit để truyền. Phía nhận cũng lưu dữ liệu dưới dạng các bit liền kề nhau. Kiểu Image được thiết kế để lưu trữ và truy xuất hiệu quả các tệp và truyền dữ liệu nhị phân.

## Kiểu Local

Dữ liệu được truyền dưới dạng các bit nhị phân, truyền theo các byte logic có kích thước được chỉ định sẵn bởi tham số của lệnh TYPE (tham số byte size). Kích thước này không nhất thiết phải giống với kích thước byte khi truyền qua kết nối dữ liệu (luôn là 8 bit). Khi dữ liệu đến máy nhận, nó sẽ chuyển đổi theo kích thước byte logic.

Ví dụ: người dùng gửi các số dấu phẩy động 36 bit tới một máy chủ có kích thước từ nhớ 32 bit có thể gửi dữ liệu dạng Local với kích thước byte logic là 36 bit. Máy chủ nhận sẽ lưu các byte logic 36 bit vào từ nhớ kép 64 bit (double word).

# **Cấu trúc dữ liệu**

Ngoài các kiểu biểu diễn khác nhau, tùy thuộc vào hệ thống tệp tin trên mỗi máy, cấu trúc của tệp tin trên các máy là khác nhau. FTP định nghĩa 3 cấu trúc của một tệp tin:

* *File-structure:* không có cấu trúc bên trong, tệp tin được coi là một chuỗi dữ liệu liên tục.
* *Record-structure:* tệp tin được tạo thành bởi dãy các bản ghi tuần tự.
* *Page-structure:* tệp tin được tạo thành bởi các trang được đánh chỉ mục độc lập.

File-structure là cấu trúc mặc định, client có thể thay đổi cấu trúc bằng cách gửi lệnh STRU. File-structure và record-structure phải được chấp nhận cho các tệp tin kiểu ASCII và EBCDIC bởi tất cả các cài đặt của giao thức FTP. Cấu trúc ban đầu của một tệp tin sẽ phụ thuộc vào máy chủ nơi lưu trữ tệp tin đó. Khi tệp tin được truyền đến để lưu trữ trên một máy khác, thì máy nhận phải có sự chuyển đổi giữa các kiểu cấu trúc sao cho phù hợp với máy đó.

Page-structure được sử dụng khi truyền các tệp tin không liên tục, các phần của tệp được chia thành các trang, các tệp được kèm theo một số thông tin mô tả và điểu khiển truy nhập trang. Cấu trúc chung của một trang bao gồm các trường: tiêu đề và nội dung trang. Tiêu đề trang gồm các thông tin:

* *Header length:* Độ dài của trường tiêu đề, độ dài là khác nhau với từng loại trang, tối thiểu là 4.
* *Page Index:* chỉ mục của trang, được sử dụng để xác định các trang trong một tệp.
* *Data Length:* Độ dài phần dữ liệu của trang.
* *Page Type:* Loại trang, có các loại trang được định nghĩa:
  + *0 = Last Page:* cho biết kết thúc của một tệp tin có cấu trúc phân trang. Trang này có Data Length = 0 và Header length = 4.
  + *1 = Simple Page:* Trang tập tin đơn giản, chứa dữ liệu của tệp tin, không kết hợp các thông tin điều khiển. Trang này có Header Length = 4.
  + *2 = Descriptor Page:* Trang mô tả cho tệp được truyền.
  + *3 = Access Controlled Page:* Chứa các thông tin điều khiển truy nhập trang. Header Length = 5.
* *Optional Fields:* cung cấp cho mỗi trang chứa thông tin điều khiển.

Tất cả các trường đều dài 1 byte logic. Kích thước byte logic được xác định bởi lệnh TYPE.

# **Các phương thức truyền dữ liệu**

FTP có 3 phương thức truyền dữ liệu, cách mà dữ liệu được truyền đi trên kết nối dữ liệu khi đã được thiết lập: Stream, Block và Compressed. Các tệp tin truyền đi phải chứa EOF hoặc thay vào đó là đóng kết nối dữ liệu để thông báo rằng dữ liệu đã được truyền xong. Đối với các tệp có cấu trúc bản ghi, tất cả các bản ghi đều chứa EOR. Còn đối với tệp tin sử dụng cấu trúc trang, có một trang thuộc loại Last Page báo hiệu kết thúc tệp tin. Việc lựa chọn phương thức truyền nằm ở phía client, client sẽ gửi lệnh MODE kèm tham số báo hiệu phương thức truyền dữ liệu. Nếu lệnh MODE không được gửi đi, mặc định Stream mode sẽ được sử dụng.

## Stream mode

Trong phương thức này, dữ liệu được truyền đi dưới dạng các luồng byte, được gửi qua kết nối TCP tới phía nhận. Dữ liệu được truyền đi là dữ liệu của tệp tin, không có các trường tiêu đề được thêm vào. Phương thức này chủ yếu dựa vào tính tin cậy trong quá trình truyền dữ liệu của TCP. Việc báo kết thúc file sẽ dựa vào EOR và EOF, nếu tệp tin sử dụng cấu trúc file-structure, các byte được gửi đi đều là các byte dữ liệu, EOF sẽ được thêm vào cuối khi bên gửi đóng kết nối dữ liệu.

## Block mode

Trong phương thức này, dữ liệu được chia thành các khối nhỏ và mỗi khối có 3 byte tiêu đề. Dưới đây là phần tiêu đề của mỗi block (Block Header):

|  |  |  |
| --- | --- | --- |
| Descriptor  8 bits | Byte Count  16 bits | Phần dữ liệu |

Trường Byte Count cho biết tổng chiều dài của phần dữ liệu tính theo đơn vị byte. Trường Descriptor chứa mã mô tả gồm các mã sau:

|  |  |
| --- | --- |
| Mã mô tả | Ý nghĩa |
| 128 | Kết thúc của khối dữ liệu là EOR |
| 64 | Kết thúc của khối dữ liệu là EOF |
| 32 | Nghi ngờ lỗi trong khối dữ liệu |
| 16 | Đánh dấu khởi động lại |

Các mã mô tả được hiển thị bằng các cờ bit tương ứng trong byte mô tả: 128 là bit cao nhất bằng 1, các bit còn lại bằng 0 (10000000), 64 = 01000000, 32 = 00100000, 16 = 00010000. Khi mã mô tả là 64 hoặc 128 thì tương ứng khối này là cuối cùng của tệp tin hoặc cuối cùng của bản ghi trong cấu trúc bản ghi. Khi mã mô tả là 32, khối dữ liệu đang truyền bị nghi ngờ là có lỗi và không đáng tin cậy. Khi mã mô tả là 16, khối này được đánh dấu khởi động lại, các marker được nhúng vào luồng dữ liệu dưới dạng số nguyên 8 bits thể hiện các ký tự. Các cờ trong mã mô tả có thể được sử dụng đồng thời, ví dụ với giá trị sau của mã mô tả: 10100000 = 160, mang ý nghĩa là kết thúc của bản ghi và nghi ngờ lỗi.

Ví dụ, để truyền một marker gồm 6 ký tự, thông tin sau sẽ được gửi đi:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Descriptor = 16 | Byte count = 6 | Marker  8 bits | Marker  8 bits | Marker  8 bits | Marker  8 bits | Marker  8 bits | Marker  8 bits |

Các marker mang các thông tin đặc biệt để từ thông tin này hệ thống có thể xác định được vị trí của dữ liệu. Marker được sử dụng để khởi động lại khi có lỗi hệ thống xảy ra. Khi một lỗi hệ thống xảy ra, một lệnh khởi động lại được gửi đi kèm theo mã đánh dấu và ngay sau đó là lệnh đang được thực hiện sẽ được gửi lại (ví dụ như yêu cầu tải 1 file), dữ liệu sẽ được truyền lại từ vị trí được đánh dấu. Cơ chế này chỉ được sử dụng khi dữ liệu được truyền đi theo block mode hoặc compressed mode và chỉ kiểm soát trong cấp độ hệ thống. Việc kiểm soát lỗi trong khi truyền được xử lý bởi TCP (ví dụ như bit bị mất khi truyền).

## Compressed mode

Trong phương thức truyền dữ liệu này, chuỗi các byte giống nhau sẽ được nén lại để giảm kích thước của dữ liệu khi gửi đi. Kỹ thuật nén này sẽ phát hiện và xử lý các đoạn bị lặp trong dữ liệu. Trong phương thức nén, dữ liệu thông thường được gửi đi sẽ được bắt đầu bằng một bit 0 và 7 bit sau đó là kích thước byte của phần dữ liệu. Do đó chuỗi các byte này sẽ có kích thước tối đa là 127 bytes. Hình sau minh họa cho cách nén này: d(i) là các byte dữ liệu, n là kích thước của phần dữ liệu tính theo bytes.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | n  7 bits | d(1)  8 bits | d(2)  8 bits | .... | d(n)  8 bits |

*Byte string*

Trường hợp dữ liệu được truyền đi xuất hiện các chuỗi bản sao của một byte dữ liệu d nào đó, FTP sẽ nén tất cả các byte giống nhau này thành một byte và thêm các thông tin biểu diễn cho cách nén này, bao gồm số các bản sao và 2 bit đầu tiên mang giá trị 0 và 1. Tổng kích thước trong trường hợp này chỉ là 2 byte. Cụ thể trong hình dưới, n là số các bản sao trong chuỗi các byte trùng nhau.

|  |  |  |
| --- | --- | --- |
| 0 1 | n  6 bits | d (8 bits)  byte dữ liệu |

*Replicated Byte*

Một trường hợp đặc biệt của cách nén này là khi dữ liệu xuất hiện chuỗi các byte tạo thành các ký tự đặc biệt, cụ thể là khoảng trắng khi kiểu dữ liệu là ASCII hoặc EBCDIC (mã ASCII 32, mã EBCDIC 64) và là byte 0 nếu kiểu là Image hoặc Local. Trường byte dữ liệu sẽ bị lược bỏ, FTP sẽ mặc định đây là các byte đặc biệt trên, được gọi là các filler byte. 2 bit đầu tiên sẽ được bật (1 1) để báo hiệu và 6 bits sau là kích thước của chuỗi các byte này. Do vậy toàn bộ chuỗi sẽ được nén lại chỉ còn 1 byte duy nhất.

|  |  |
| --- | --- |
| 1 1 | n  6 bits |

*Filler String*

Sau khi dữ liệu được nén nó sẽ được xử lý giống như block mode, cũng có một trường tiêu đề được thêm vào, dữ liệu bây giờ sẽ được hiểu là dữ liệu đã được nén nghĩa là các cách nén ở trên được mô tả trong phần dữ liệu tương tự trường dữ liệu của block mode.

# **Lệnh và mã hồi đáp trong FTP**

Lệnh được truyền đi trên kết nối điều khiển từ client tới server, mã trả lời được truyền từ server tới client tương ứng với mỗi lệnh.

## Các lệnh FTP (FTP commands)

Mỗi câu lệnh chứa 4 ký tự ASCII in hoa, nếu thiếu sẽ được thêm cuối bởi dấu cách. Cách lệnh lại được chia thành 3 loại là nhóm các lệnh điều khiển truy nhập, nhóm các lệnh tham số cho quá trình truyền và nhóm các lệnh dịch vụ.

1. *Nhóm lệnh điều khiển truy nhập (access control commands)*

|  |  |
| --- | --- |
| Cú pháp | Ý nghĩa |
| USER <SP> <username> <CRLF>  <SP> ::= dấu cách  <CRLF>::= kết thúc một dòng  <username>::= <string> // xâu kí tự | Sử dụng để gửi thông tin định danh người dùng cho server, lệnh này thường sẽ là lệnh đầu tiên được truyền bởi người dùng sau khi kết nối điều khiển được thiết lập |
| PASS <SP> <password> <CRLF>  <password>::= <string> | Sử dụng để xác minh mật khẩu với user tương ứng vừa được gửi qua lệnh USER. Sau khi xác minh chính xác, một phiên làm việc FTP sẽ được bắt đầu. |
| CWD <SP> <path-name> <CRLF>  <path-name> ::= <string> | Thay đổi thư mục đang làm việc, tham số pathname có thể là đường dẫn tương đối hoặc tuyệt đối. |
| ACCT <SP> <account-information> <CRLF>  <account-information>::= <string> | Thông tin tài khoản trong phiên làm việc. |
| CDUP <CRLF> | Là trường hợp đặc biệt của lệnh CWD, lệnh này sẽ đặt thư mục làm việc là thư mục cha của thư mục hiện tại (quay lại làm việc trên thư mục cha). |
| SMNT <SP> <pathname> <CRLF>  <pathname> ::= <string> | Gắn kết tài khoản người sử dụng với một hệ thống file khác. Nghĩa là trao quyền truy cập tới một thư mục trong tham số pathname. |
| REIN <CRLF> | Khởi tạo lại kết nối điều khiển, người dùng sẽ bị đăng xuất. Một phiên làm việc mới sẽ được thiết lập với một người dùng mới |
| QUIT <CRLF> | Máy chủ sẽ đóng kết nối điều khiển. |

1. *Nhóm lệnh tham số cho quá trình truyền (transfer parameter commands)*

|  |  |
| --- | --- |
| Cú pháp | Ý nghĩa |
| PORT <SP> <host-port> <CRLF>  <host-port>::=<host num>, <port num>  <host number> = n1, n2, n3, n4  <port number> = n5, n6 | Báo cho server biết số hiệu cổng của Client-DTP, thông thường lệnh này sẽ được gửi đi mỗi khi kết nối kênh truyền dữ liệu theo dạng chủ động. Cú pháp của tham số host number bao gồm 4 số cách nhau bởi dấu phẩy tương ứng với địa chỉ ip của server. Port number bao gồm 2 số cách nhau bởi dấu phẩy tương ứng 8 bit cao và 8 bit thấp của số hiệu cổng. Giả sử client-DTP sử dụng cổng 1780 = 6\*256 + 244.  Vậy thông số cần truyền đi là port number = 6, 244. |
| PASV <CRLF> | Báo cho server để khởi tạo kết nối dữ liệu theo kiểu bị động. Sau lệnh này, server sẽ gửi mã trả lời, nếu thành công, một cổng được gửi cho client giống như dạng ở trên. |
| TYPE <SP> <type-code> <CRLF>  <type-code>::= A | E | I | L  // ASCII | EBCDIC | Image | Local | Chuyển đổi kiểu dữ liệu được truyền đi. Tham số type code tương ứng với các kiểu dữ liệu mà FTP quy định đã trình bày ở mục trên. |
| STRU <SP><structure-code> <CRLF>  <structure-code>::= F | R | P  // File | Record | Page | Xác định cấu trúc tệp tin được sử dụng. Tham số structure-code tương ứng với các cấu trúc file, cấu trúc bản ghi, cấu trúc trang. |
| MODE <SP> <mode-code> <CRLF>  <mode-code>::= S | B | C | Xác định phương thức truyền dữ liệu gồm có 3 phương thức tương ứng với tham số mode-code là Stream, Block và Compressed. |

1. *Nhóm các lệnh dịch vụ (service commands)*

|  |  |
| --- | --- |
| Cú pháp | Ý nghĩa |
| RETR <SP> <pathname> <CRLF> | Yêu cầu máy chủ truyền một bản sao của tệp trong đường dẫn pathname tới client (tải tệp tin từ máy chủ). |
| STOR <SP> <pathname> <CRLF> | Yêu cầu máy chủ lưu dữ liệu được truyền qua kết nối dữ liệu tại đường dẫn path name, dữ liệu được gửi từ phía client (upload tệp tin lên máy chủ). Tên của tệp lưu trên máy chủ cũng nằm trong tham số pathname. |
| STOU <CRLF> | Tương tự STOR nhưng loạt bỏ tham số pathname, lúc này tệp sẽ được lưu tại thư mục đang làm việc và tên sẽ được máy chủ đặt tự động, tên này sẽ nằm trong mã hồi đáp của lệnh này. |
| APPE <SP> <pathname> <CRLF> | Lệnh này làm yêu cầu máy chủ lưu dữ liệu được truyền qua kết nối dữ liệu vào cuối của một tệp tin trong tham số pathname, nếu tệp chưa có sẽ được tạo mới. |
| ALLO <SP> <decimal-int> [<SP> R <SP> <decimal-int>] <CRLF> | Yêu cầu máy chủ bố trí không gian để lưu trữ tệp trên đĩa. Tham số decimal-int đại diện cho số byte được dành cho tệp. Các tham số tiếp theo có thể được sử dụng khi tệp được gửi với cấu trúc bản ghi (R), với tham số kích thước bản ghi (bytes). Lệnh này nếu được sử dụng sẽ luôn theo sau lệnh STOR hoặc APPE. |
| REST <SP> <marker> <CRLF> | Khởi động lại từ một điểm đánh dấu trong tham số marker. |
| RNFR <SP> <pathname> <CRLF> | Yêu cầu đổi tên của tệp tin trong pathname |
| RNTO <SP> <pathname> <CRLF> | Tên mới cho tệp tin được yêu cầu đổi tên, lệnh này và lệnh trên luôn được đi cùng nhau để đổi tên một tệp tin. |
| ABOR <CRLF> | Yêu cầu hủy bỏ lệnh dịch vụ FTP trước đó và đóng kết nối dữ liệu. |
| DELE <SP> <pathname> <CRLF> | Yêu cầu xóa tệp tin trong đường dẫn. |
| RMD <SP> <pathname> <CRLF> | Yêu cầu loại bỏ thư mục trong đường dẫn. Thư mục bị loại bỏ phải trống (không chứa tệp tin nào). |
| MKD <SP> <pathname> <CRLF> | Yêu cầu tạo thư mục với tên và vị trí trong tham số pathname. |
| PWD <CRLF> | Yêu cầu trả về thư mục đang làm việc. |
| LIST [<SP> <pathname>] <CRLF> | Yêu cầu trả về danh sách thuộc tính (tên, kích thước, ngày tạo,..) các tệp tin và thư mục con của một thư mục trong đường dẫn. Nếu tham số này trống, thư mục đang làm việc sẽ được trả về. Dữ liệu truyền đi sẽ coi như là dữ liệu tệp tin và được truyền trên kết nối dữ liệu. |
| NLST [<SP> <pathname>] <CRLF> | Giống như LIST nhưng lệnh này chỉ yêu cầu trả về tên của tệp tin và thư mục con. |
| SITE <SP> <string> <CRLF> | Gửi các lệnh cung cấp các dịch vụ cụ thể, không nằm trong danh sách các lệnh dịch vụ của FTP. |
| SYST <CRLF> | Yêu cầu trả về thông tin hệ điều hành của máy chủ. |
| STAT [<SP> <pathname>] <CRLF> | Yêu cầu phản hồi trạng thái của máy chủ. Lệnh này được gửi trong khi truyền tệp tin. Khi tham số được sử dụng, lệnh này cho biết thông tin trạng thái về một tệp tin trong đường dẫn, thông tin này được truyền qua kết nối điều khiển (gần giống với LIST). |
| HELP [<SP> <string>] <CRLF> | Yêu cầu hướng dẫn về các lệnh cụ thể hoặc toàn bộ lệnh. |
| NOOP <CRLF> | Không ảnh hưởng đến các lệnh khác. Không yêu cầu gì cả. Máy chủ sẽ hồi đáp OK. |

## Mã hồi đáp (Reply code)

Sau mỗi câu lệnh FTP được gửi đi, sẽ có một phản hồi từ server, phản hồi này bao gồm mã hồi đáp và thông tin mô tả cụ thể mã hồi đáp này. Mã hồi đáp bao gồm 3 chữ số ký hiệu là xyz. Tùy thuộc vào giá trị của các chữ số này và vị trí của chúng mà có thể chia ra các loại hồi đáp khác nhau, cụ thể như sau.

Về chữ số đầu tiên:

* 1yz, 3yz: Lệnh đã được chấp nhận nhưng bị hoãn lại và chờ thêm thông tin để hoàn thành.
* 2yz: Yêu cầu đã được hoàn thành, có thể bắt đầu một lệnh mới.
* 4yz, 5yz: Lệnh không được chấp nhận.

Về chữ số thứ hai:

* x0z: Đề cập tới lỗi về cú pháp.
* x1z: Trả lời các yêu cầu về thông tin (ví dụ: help hoặc trạng thái của máy chủ).
* x2z: Trả lời đề cập đến kết nối dữ liệu và kết nối điều khiển.
* x3z: Trả lời đề cập về xác thực (đăng nhập).
* x4z: Không sử dụng
* x5z: Trả lời liên quan đến hệ thống tệp tin của máy chủ.

Chữ số thứ ba sẽ chỉ rõ từng mã hồi đáp cụ thể và mục đích của chúng. Mỗi một lệnh FTP sẽ có nhiều mã hồi đáp cho lệnh đó, ví dụ như những mã lỗi cú pháp, mã về kết nối. Tuy nhiên, có chỉ có một mã báo lệnh đó được chấp nhận thành công, tương ứng một lệnh sẽ có một mã như vậy. Ví dụ, khi lệnh PASV được chấp nhận, mã hồi đáp 227 sẽ báo cho client biết yêu cầu đã được hoàn thành và có thông tin kèm theo sau mã hồi đáp đó.

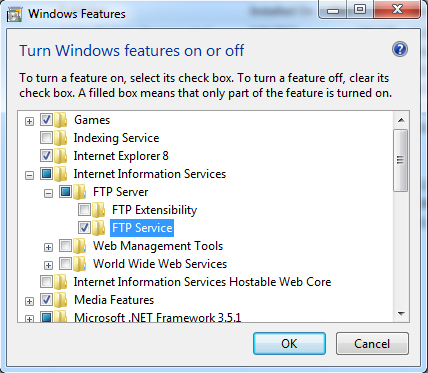
Dưới đây là danh sách các mã hồi đáp của FTP và thông tin phản hồi mô tả chung. Tùy vào tình huống cụ thể mà thông tin đi kèm có thể bổ sung thêm để mô tả rõ phải hồi của server.

* 110 Restart marker reply.
* 120 Service ready in nnn minutes.
* 125 Data connection already open; transfer starting.
* 150 File status okay; about to open data connection.
* 200 Command okay.
* 202 Command not implemented, superfluous at this site.
* 211 System status, or system help reply.
* 212 Directory status.
* 213 File status.
* 214 Help message.
* 215 NAME system type. *// NAME là hệ thống chính thức*
* 220 Service ready for new user.
* 221 Service closing control connection.
* 225 Data connection open; no transfer in progress.
* 226 Closing data connection.
* 227 Entering Passive Mode (h1,h2,h3,h4,p1,p2).
* 230 User logged in, proceed.
* 250 Requested file action okay, completed.
* 257 "PATHNAME" created.
* 331 User name okay, need password.
* 332 Need account for login.
* 350 Requested file action pending further information.
* 421 Service not available, closing control connection.
* 425 Can't open data connection.
* 426 Connection closed; transfer aborted.
* 450 Requested file action not taken.
* 451 Requested action aborted: local error in processing.
* 452 Requested action not taken.
* 500 Syntax error, command unrecognized.
* 501 Syntax error in parameters or arguments.
* 502 Command not implemented.
* 503 Bad sequence of commands.
* 504 Command not implemented for that parameter.
* 530 Not logged in.
* 532 Need account for storing files.
* 550 Requested action not taken.
* 551 Requested action aborted: page type unknown.
* 552 Requested file action aborted.
* 553 Requested action not taken. File name not allowed.

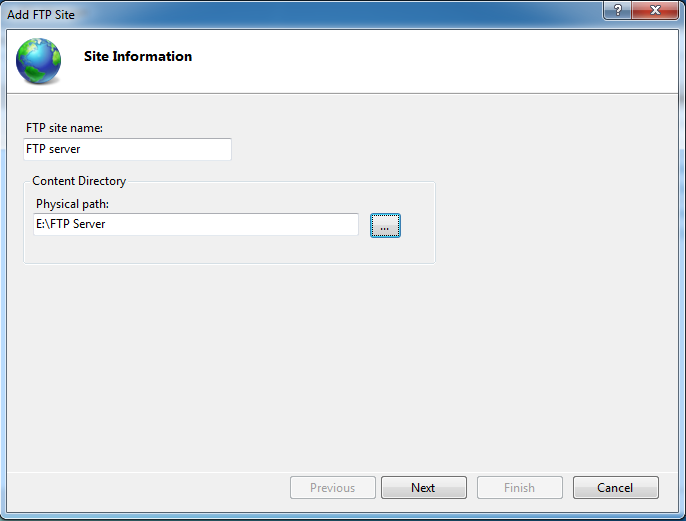
# **Phần 2. Cài đặt FTP Server và viết chương trình FTP Client**

# **Cài đặt FTP Server**

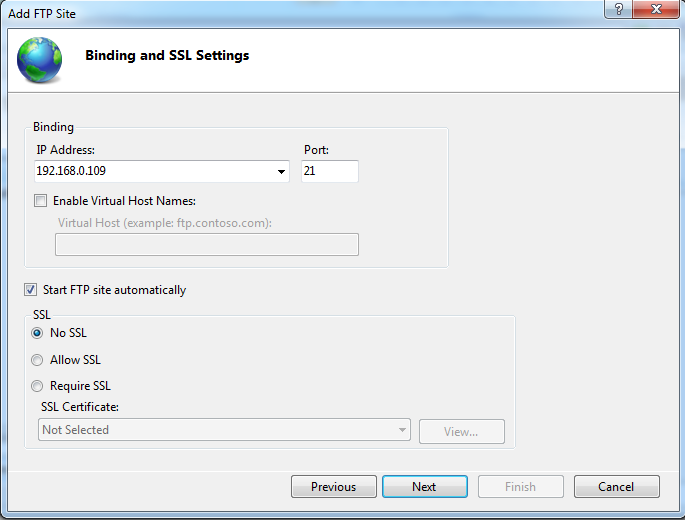
* Cài đặt trên Windows.
* Kích hoạt dịch vụ FTP được Windows hỗ trợ.



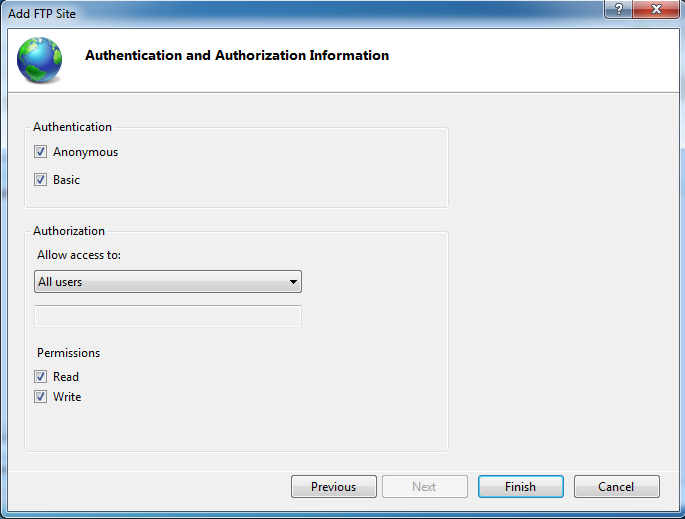
* Tạo đường dẫn thư mục chia sẻ trên máy chủ.



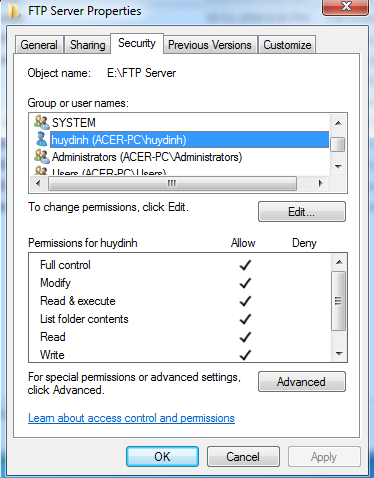
* Đặt địa chỉ ip lắng nghe kết nối và số hiệu cổng (mặc định của FTP là 21).



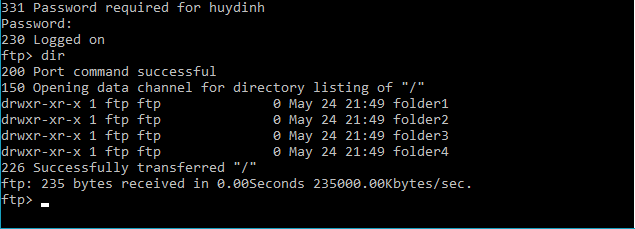
* Phân quyền các Users chung với các tệp tin chia sẻ



* Tạo thêm User mới và phân quyền đối với user này



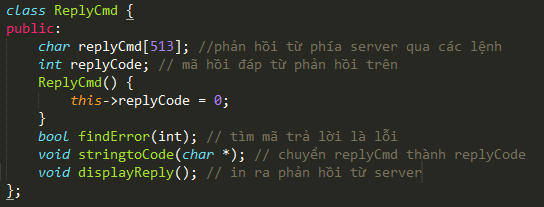
* Kiểm tra bằng command line trên Windows.



# **Chương trình FTP Client**

Chương trình sử dụng ngôn ngữ C/C++, thư viện Winsock. Đáp ứng được một số yêu cầu từ người dùng như: tải file, upload file, hiển thị danh sách file trong một thư mục, xóa file, đổi tên thư mục/file, tạo thư mục.

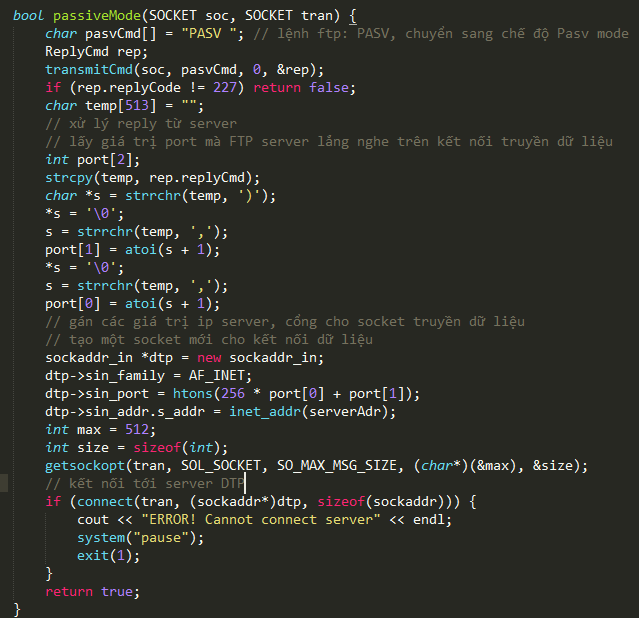
* Sử dụng chế độ truyền mặc định Stream mode.
* Sử dụng kiểu mặc định ACSII type.
* Sử dụng cấu trúc mặc định: file-structure.
* Kết nối dữ liệu kiểu bị động.
* Cấu trúc các hồi đáp từ server:



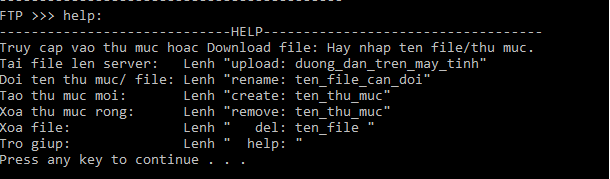
* Hàm gửi các câu lệnh FTP:



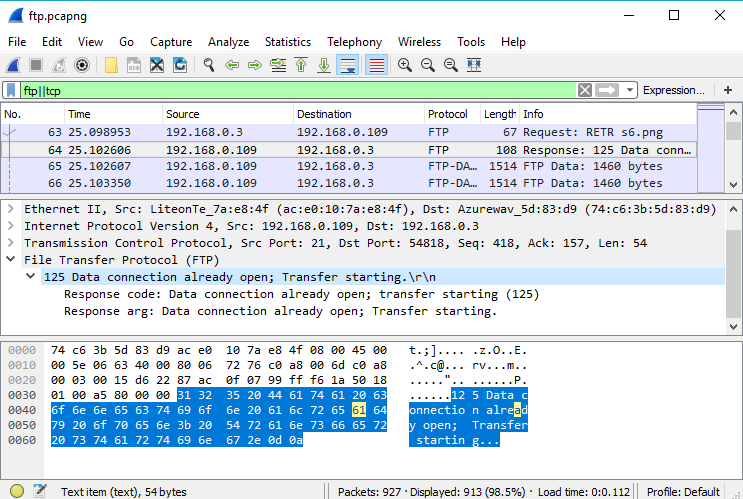
* Hàm kết nối truyền dữ liệu bằng kiểu bị động:

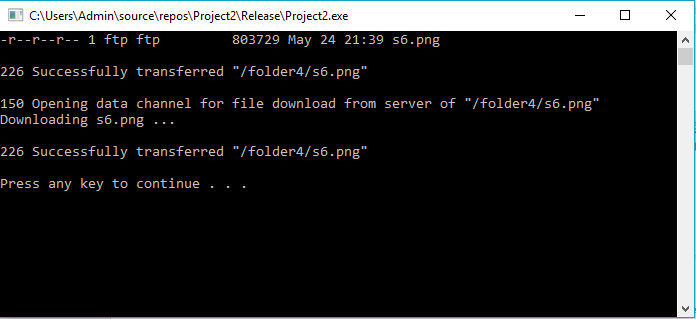


* Giao diện và các thao tác mà chương trình hỗ trợ



* Sử dụng Wireshark để bắt gói tin truyền lệnh và phản hồi giữa hai máy tính trong cùng mạng LAN. Sử dụng lệnh RERT để tải file s6.png





# ***Danh mục tài liệu tham khảo:***

* RFC 959 File Transfer Protocol, J. Postel & J. Reynolds, 10/1985

***Phân công công việc trong nhóm:***

* Nguyễn Vĩnh Thụy: Trình bày slide và viết báo cáo các phần:
  + *I. Tổng quan*
  + *II.* *Hoạt động của giao thức FTP*
* Nguyễn Ngọc Đức: Trình bày slide và viết báo cáo các phần:
  + *III. Các kiểu dữ liệu khi truyền dữ liệu*
  + *IV. Cấu trúc dữ liệu*
  + *V. Các phương thức truyền dữ liệu*
* Nguyễn Huy Định: Trình bày slide, cài đặt, viết chương trình và viết báo cáo các phần:
  + *VI. Các lệnh FTP*
  + *Cài đặt FTP Server và viết chương trình FTP Client*