# LỜI NÓI ĐẦU

*Thế kỷ 21 được mệnh danh là thế kỷ của công nghệ thông tin, với sự bùng nổ mạnh mẻ về khoa học công nghệ. Đây là kỷ nguyên của nền văn minh dựa trên cơ sở công nghiệp trí tuệ . Ngày nay, tin học đã trở thành một môn khoa học quan trọng trên thế giới.*

*Sự phát triển mạnh mẽ nhƣ vậy thì công việc lập trình các ứng dụng nhằm phục vụ nhu cầu, lợi ích của con ngƣời trở nên cấp thiết. Máy tính đã trở thành công cụ đắc lực và không thể thiếu của con người. Các tổ chức, công ty hay các cơ quan cần phải* [*xây*](http://kilobooks.com/forums/125-X%C3%A2y-D%E1%BB%B1ng)[*dựngluận văn - báo cáo - tiểu luận chuyên ngành xây dựng*](http://kilobooks.com/forums/125-X%C3%A2y-D%E1%BB%B1ng) *hệ thống mạng máy tính cho riêng mình để trao đổi dữ liệu giữa các bộ phận. Dữ liệu được truyền đi trên mạng phải đảm bảo: dữ liệu đƣợc chuyển tới đích nhanh chóng và đúng đắn. Hầu hết dữ liệu đƣợc truyền qua mạng là truyền dưới dạng file.*

*Nhằm tìm hiểu thấu đáo một trong số các phương pháp truyền file qua mạng em chọn đề tài "Sử Dụng Kỹ Thuật Lập Trình Socket Xây Dựng Chương Trình Truyền file Qua Mạng Lan”.Với lập trình socket sẽ bắt buộc các máy đó phải được nối mạng với nhau. Ta đã thấy các máy muốn trao đổi dữ liệu qua mạng, chúng sẽ tạo ra ở mỗi phía một socket và trao đổi dữ liệu bằng cách đọc/ghi từ socket. Khi một chương trình tạo ra một socket, một định danh dạng số (định danh dạng số này còn được gọi là số hiệu cổng) sẽ được gán cho socket. Việc gán số hiệu cổng này cho socket có thể được thực hiện bởi chương trình hoặc hệ điều hành. Trong mỗi gói tin mà socket gửi đi có chứa hai thông tin để xác định đích đến của gói tin:*

* *Một địa chỉ mạng để xác định hệ thống sẽ nhận gói tin*
* *Một số định danh cổng để nói cho hệ thống đích biết socket nào trên nó sẽ nhận dữ liệu*

*Mục tiêu của đồ án là tìm hiểu về mạng máy tính và kĩ thật lập trình socket với ngôn ngữ lập trình là C#, trên cơ sở đó xây dựng ứng dụng truyền file trên mạng cục bộ. Đồ án sẽ xây dựng thử nghiệm một hệ thống cho phép trao đổi dữ liệu dƣới dạng truyền*

*file, tương tác điểm – điểm trên mạng LAN. Đề tài được trình bày gồm các chương với bố cục như sau :*

*CHƯƠNG I: Tổng Quan Về Lập Trình Mạng CHƯƠNG II: Lập Trình Socket Với C#*

*CHƯƠNG III: Xây Dựng Và Thiết Kế Chương Trình TÀI LIỆU THAM KHẢO*

*Do đây là sản phẩm đầu tay nên chúng em chưa có nhiều kinh nghiệm,vậy nên sẽ có nhiều hạn chế và sai sót trong qúa trình thực hiện, mong thầy cô và các bạn đóng góp ý kiến để đề tài của nhóm em được hoàn chỉnh hơn. Từ đó làm cơ sở để nhóm có thể củng cố và phát triển thêm .*

*Cuối cùng, nhóm em xin chân thành cảm ơn thầy Nguyễn Đức Toàn và các bạn trong lớp đã góp ý giúp đỡ nhóm em hoàn thành xong đề tài này. Chúng em xin chúc thầy và các bạn thành công và vui vẻ trong cuộc sống.*

*Hà Nội, 30/09/2018*

# MỤC LỤC

[LỜI NÓI ĐẦU 1](#_bookmark0)

[MỤC LỤC 3](#_bookmark1)

[DANH MỤC HÌNH VẼ 5](#_bookmark2)

[CHƯƠNG I 6](#_bookmark3)

[TỔNG QUAN VỀ LẬP TRÌNH MẠNG 6](#_bookmark4)

* 1. [MÔ HÌNH THAM CHIẾU OSI 6](#_bookmark5)
  2. [CHỒNG GIAO THỨC TCP/IP 10](#_bookmark6)
  3. [MÔ HÌNH TRUYỀN THÔNG TRONG CẤU TRÚC MẠNG 12](#_bookmark7)
     1. [Nguyên tắc truyền thông 12](#_bookmark8)
     2. [Nguyên tắc của phương pháp phân tầng 12](#_bookmark9)
  4. [ĐỊA CHỈ IP – CÁC ĐỊA CHỈ IP DÀNH RIÊNG 15](#_bookmark10)
  5. [GIAO THỨC TRUYỀN FILE FTP 15](#_bookmark11)
     1. [Khái niệm về giao thức FTP 15](#_bookmark12)
     2. [Đặc điểm và nguyên lý hoạt động 16](#_bookmark13)
        1. Mô hình hoạt động của FTP 16
        2. Thiết lập kênh điều khiển và chứng thực người dùng của FTP 19
        3. Quản lí dữ liệu của FTP 21
        4. Các phương thức truyền dữ liễu trong FTP 25

[CHƯƠNG II 27](#_bookmark14)

[LẬP TRÌNH MẠNG VỚI SOCKET 27](#_bookmark15)

* 1. [TỔNG QUAN VỀ C# 27](#_bookmark16)
     1. [Cơ Bản Về NET Framework 27](#_bookmark17)
     2. [Các thành phần của .NET Framework 27](#_bookmark18)
     3. [Các tính năng cơ bản của ngôn ngữ lập trình C# 28](#_bookmark19)
     4. [Các ứng dụng của C# 28](#_bookmark20)
     5. [Các lợi ích của C# 28](#_bookmark21)
     6. [Môi trường phát triển tích hợp Visual Studio .NET 2005 29](#_bookmark22)
  2. [GIỚI THIỆU VỀ LẬP TRÌNH SOCKET 29](#_bookmark23)
  3. [LẬP TRÌNH MẠNG VỚI TCPSOCKET 30](#_bookmark24)
     1. [Mô hình giao thức 30](#_bookmark25)
     2. [Thiết lập kết nối 31](#_bookmark26)
     3. [Truyền nhận dữ liệu 34](#_bookmark27)
     4. [Đóng liên kết 34](#_bookmark28)
  4. [LẬP TRÌNH MẠNG VỚI UDP SOCKET 35](#_bookmark29)
     1. [Giao thức UDP 35](#_bookmark30)
     2. [Một số thuật ngữ UDP 36](#_bookmark31)
     3. [Hoạt động của giao thức UDP 38](#_bookmark32)
     4. [Các nhược điểm của giao thức UDP 38](#_bookmark33)
     5. [Các ưu điểm của UDP 38](#_bookmark34)
     6. [Khi nào thì nên sử dụng UDP 39](#_bookmark35)
  5. [MÔ HÌNH CLIENT/SERVER 39](#_bookmark36)
     1. [Tổng quan 39](#_bookmark37)
     2. [Giao thức cho ứng dụng Client/Server 47](#_bookmark38)

[CHƯƠNG III 51](#_bookmark39)

[PHÂN TÍCH THIẾT KẾ CHƯƠNG TRÌNH 51](#_bookmark40)

* 1. [MỤC ĐÍCH CỦA ĐỀ TÀI 51](#_bookmark41)
  2. [GIỚI THIỆU VỀ CHƯƠNG TRÌNH 51](#_bookmark42)
  3. [GIỚI THIỆU THÀNH VIÊN VÀ PHÂN CÔNG NHIỆM VỤ 52](#_bookmark43)
     1. [Giới thiệu thành viên: 52](#_bookmark44)
     2. [Phân công nhiệm vụ các thành viên: 52](#_bookmark45)
        1. [Kế hoạch đề ra 52](#_bookmark46)

[3.3.2.1. Phân công nhiệm vụ 52](#_bookmark47)

* 1. [GIAO DIỆN VÀ SỬ DỤNG CHƯƠNG TRÌNH: 53](#_bookmark48)
     1. [Giao diện: 53](#_bookmark49)
        1. [Chương trình server 53](#_bookmark50)
        2. [Chương trình client 54](#_bookmark51)
     2. [Sử dụng chương trình: 54](#_bookmark52)

[KẾT LUẬN 55](#_bookmark53)

[TÀI LIỆU THAM KHẢO 56](#_bookmark54)

# DANH MỤC HÌNH VẼ

*Hình I - 1. Mô hình OSI 6*

*Hình I - 2. Mô hình TCP/IP 10*

*Hình I - 3. Mô hình truyền thông đơn giản 3 tầng 12*

*Hình I - 4. Trao đổi giữa các tầng 14*

*Hình I - 5. Các tiến trình trong mô hình FTP 17*

*Hình I - 6. Tiến trình làm việc dạng chủ động trong FTP 21*

*Hình I - 7. Tiến trình làm việc dạng bị động trong FTP 22*

*Hình II - 1. Các giao thức trong lập trình Socket. 29*

*Hình II - 2. Kết nối TCP 29*

*Hình II - 3. Dạng thức của gói tin UDP 33*

*Hình III - 1. Quá trình kết nối 48*

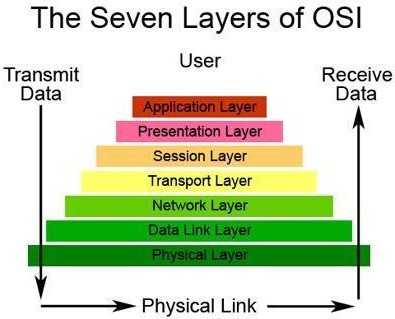
*Hình III - 2. Giao diện chương trình Server 50*

*Hình III - 3. Giao diện chương trình Client 55*

# CHƯƠNG I

# TỔNG QUAN VỀ LẬP TRÌNH MẠNG

### MÔ HÌNH THAM CHIẾU OSI



*Hình I - 1. Mô hình OSI*

* + - Mô hình tham chiếu Open System Interconnect (OSI) là một mô hình được phát triển bởi Open System Interconnect (OSI), đây là mô hình mô tả cách dữ liệu từ một ứng dụng trên máy tính được truyền đến một ứng dụng trên máy tính khác như thế nào. Mô hình tham chiếu OSI gồm có 7 lớp, mỗi lớp giữ các chức năng mạng khác nhau. Mỗi một chức năng của một mạng có thể được gán với một hoặc có thể một cặp lớp liền kề, của 7 lớp này và có quan hệ độc lập với các lớp khác.
    - Sự tách lớp của mô hình này mang lại những lợi ích sau:

o Chia hoạt động thông tin mạng thành những phần nhỏ hơn, đơn giản hơn giúp chúng ta dễ khảo sát. o Chuẩn hóa các thành phần mạng để cho phép phát triển mạng từ nhiều nhà cung cấp sản phẩm.

* + - Ngăn chặn được tình trạng sự thay đổi của một lớp làm ảnh hưởng đến các lớp khác, như vậy giúp mỗi lớp có thể phát triển độc lập và nhanh chóng hơn.

#### Mô hình tham chiếu OSI định nghĩa các qui tắc :

* + - * Cách thức các thiết bị giao tiếp và truyền thông được với nhau.
      * Các phương pháp để các thiết bị trên mạng khi nào thì được truyền dữ liệu, khi nào thì không được.
      * Các phương pháp để đảm bảo truyền đúng dữ liệu và đúng bên nhận.
      * Cách thức vận tải, truyền, sắp xếp và kết nối với nhau.
      * Cách thức đảm bảo các thiết bị mạng duy trì tốc độ truyền dữ liệu thích hợp.
      * Cách biểu diễn một bit thiết bị truyền dẫn.

#### Mô hình tham chiếu OSI với 7 lớp với các chức năng :

* + - * *Lớp ứng dụng (Application Layer):* là giao diện giữa các chương trình ứng dụng của người dùng và mạng. Lớp Application xử lý truy nhập mạng chung, kiểm soát luồng và phục hồi lỗi. Lớp này không cung cấp các dịch vụ cho lớp nào mà nó cung cấp dịch vụ cho các ứng dụng như: truyền file, gởi nhận E-mail, Telnet, HTTP, FTP, SMTP…
      * *Lớp trình diễn (Presentation Layer):* lớp này chịu trách nhiệm thương lượng và xác lập dạng thức dữ liệu được trao đổi. Nó đảm bảo thông tin mà lớp ứng dụng của một hệ thống đầu cuối gởi đi, lớp ứng dụng của hệ thống khác có thể đọc được. Lớp trình bày thông dịch giữa nhiều dạng dữ liệu khác nhau thông qua một dạng chung, đồng thời nó cũng nén và giải nén dữ liệu.
      * *Lớp phiên (Session Layer*): lớp này có chức năng thiết lập, quản lý, và kết thúc các phiên thông tin giữa hai thiết bị truyền nhận. Lớp phiên cung cấp các dịch vụ cho lớp trình bày. Lớp Session cung cấp sự đồng bộ hóa giữa các tác vụ người dùng bằng cách đặt những điểm kiểm tra vào luồng dữ liệu. Bằng cách này, nếu

mạng không hoạt động thì chỉ có dữ liệu truyền sau điểm kiểm tra cuối cùng mới phải truyền lại. Lớp này cũng thi hành kiểm soát hội thoại giữa các quá trình giao tiếp, điều chỉnh bên nào truyền, khi nào, trong bao lâu. Ví dụ như: RPC, NFS,... Lớp này kết nối theo ba cách: Haft duplex,Simplex, Fullduplex.

* + - * *Lớp vận chuyển (Transport Layer)****:*** lớp vận chuyển phân đoạn dữ liệu từ hệ thống máy truyền và tái thiết lập dữ liệu vào một luồng dữ liệu tại hệ thống máy nhận đảm bảo rằng việc bàn giao các thông điệp giữa các thiết bị đáng tin cậy.

Dữ liệu tại lớp này gọi là segment.

Lớp này thiết lập, duy trì và kết thúc các mạch ảo đảm bảo cung cấp các dịch vụ sau:

* + - * + Xếp thứ tự các phân đoạn: khi một thông điệp lớn được tách thành nhiều phân đoạn nhỏ để bàn giao, lớp vận chuyển sẽ sắp xếp thứ tự các phân đoạn trước khi ráp nối các phân đoạn thành thông điệp ban đầu. o Kiểm soát lỗi: khi có phân đoạn bị thất bại, sai hoặc trùng lắp, lớp vận chuyển sẽ yêu cầu truyền lại.
        + Kiểm soát luồng: lớp vận chuyển dùng các tín hiệu báo nhận để xác nhận. Bên gửi sẽ không truyền đi phân đoạn dữ liệu kế tiếp nếu bên nhận chưa gởi tín hiệu xác nhận rằng đã nhận được phân đoạn dữ liệu trước đó đầy đủ.
      * *Lớp mạng (Network Layer):* lớp mạng chịu trách nhiệm lập địa chỉ các thông điệp, diễn dịch địa chỉ và tên logic thành địa chỉ vật lý đồng thời nó cũng chịu trách nhiệm gởi packet từ mạng nguồn đến mạng đích. Lớp này quyết định đường đi từ máy tính nguồn đến máy tính đích. Nó quyết định dữ liệu sẽ truyền trên đường nào dựa vào tình trạng, ưu tiên dịch vụ và các yếu tố khác. Nó cũng quản lý lưu lượng trên mạng chẳng hạn như chuyển đổi gói, định tuyến, và kiểm soát sự tắc nghẽn dữ liệu.Dữ liệu ở lớp này gọi packet hoặc datagram.
      * *Lớp liên kết dữ liệu (Data link Layer):* cung cấp khả năng chuyển dữ liệu tin cậy xuyên qua một liên kết vật lý. Lớp này liên quan đến: o Địa chỉ vật lý o Mô hình mạng o Cơ chế truy cập đường truyền o Thông báo lỗi o Thứ tự phân phối frame
        + Điều khiển dòng.

Tại lớp data link, các bít đến từ lớp vật lý được chuyển thành các frame dữ liệu bằng cách dùng một số nghi thức tại lớp này. Lớp data link được chia thành hai lớp con: o Lớp con LLC (logical link control). o Lớp con MAC (media access control)

Lớp con LLC là phần trên so với các giao thức truy cập đường truyền khác, nó cung cấp sự mềm dẻo về giao tiếp. Bởi vì lớp con LLC hoạt động độc lập với các giao thức truy cập đường truyền, cho nên các giao thức lớp trên hơn (ví dụ như IP ở lớp mạng) có thể hoạt động mà không phụ thuộc vào loại phương tiện LAN. Lớp con LLC có thể lệ thuộc vào các lớp thấp hơn trong việc cung cấp truy cập đường truyền. Lớp con MAC cung cấp tính thứ tự truy cập vào môi trường LAN. Khi nhiều trạm cùng truy cập chia sẻ môi trường truyền, để định danh mỗi trạm, lớp cho MAC định nghĩa một trường địa chỉ phần cứng, gọi là địa chỉ MAC address.

Địa chỉ MAC là một con số đơn nhất đối với mỗi giao tiếp LAN (card mạng). Lớp vật lý (Physical Layer): định nghĩa các qui cách về điện, cơ, thủ tục và các đặc tả chức năng để kích hoạt, duy trì và dừng một liên kết vật lý giữa các hệ thống đầu cuối.

Một số các đặc điểm trong lớp vật lý này bao gồm:

Mức điện thế.

Khoảng thời gian thay đổi điện thế.

Tốc độ dữ liệu vật lý.

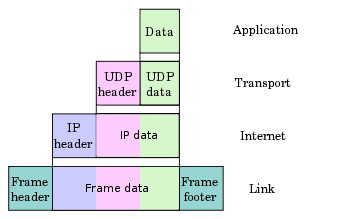
Khoảng đường truyền tối đa.

Các đầu nối vật lý.

* + - * *Lớp vật lý(physical):* Lớp vật lý bao gồm việc truyền tải các tín hiệu trong môi trường từ máy tính này đến máy tính khác. Lớp này gồm có các chi tiết kỹ thuật về các đặc tính điện và cơ như: mức điện áp, định thời tín hiệu, tốc độ dữ liệu, độ dài truyền tải lớn nhất và các kết nối vật lý của thiết bị mạng. Để một thiết bị hoạt động chỉ trong lớp vật lý, nó sẽ không có bất kỳ kiến thức nào về dữ liệu mà nó

truyền tải. Một thiết bị lớp vật lý chỉ truyền tải hoặc nhận dữ liệu một cách đơn giản*.*

### CHỒNG GIAO THỨC TCP/IP



*Hình I - 2. Mô hình TCP/IP*

* + - Sự ra đời của họ giao thức TCP/IP gắn liền với sự ra đời của Internet mà tiền thân là mạng ARPAnet (Advanced Research Projects Agency) do Bộ Quốc phòng Mỹ tạo ra. Đây là bộ giao thức được dùng rộng rãi nhất vì tính mở của nó. Hai giao thức được dùng chủ yếu ở đây là TCP (Transmission Control Protocol) và IP (Internet Protocol). Chúng đã nhanh chóng được đón nhận và phát triển bởi nhiều nhà nghiên cứu và các hãng công nghiệp máy tính với mục đích xây dựng và phát triển một mạng truyền thông mở rộng khắp thế giới mà ngày nay chúng ta gọi là Internet.
    - TCP/IP có cấu trúc tương tự như mô hình OSI, tuy nhiên để đảm bảo tính tương thích giữa các mạng và sự tin cậy của việc truyền thông tin trên mạng, bộ giao thức TCP/IP được chia thành 2 phần riêng biệt: giao thức IP sử dụng cho việc kết nối mạng và giao thức TCP để đảm bảo việc truyền dữ liệu một cách tin cậy.
    - *Lớp ứng dụng:* Tại mức cao nhất này, người sử dụng thực hiện các chương trình ứng dụng truy xuất đến các dịch vụ hiện hữu trên TCP/IP Internet. Một ứng dụng tương tác với một trong những protocol ở mức giao vận (transport) để gửi hoặc nhận dữ liệu. Mỗi chương trình ứng dụng chọn một kiểu giao vận mà nó cần, có thể là một dãy tuần tự từng thông điệp hoặc một chuỗi các byte liên tục. Chương trình ứng dụng sẽ gửi dữ liệu đi dưới dạng nào đó mà nó yêu cầu đến lớp giao vận.
    - *Lớp giao vận:* Nhiệm vụ cơ bản của lớp giao vận là cung cấp phưng tiện liên lạc từ một chương trình ứng dụng này đến một chưng trình ứng dụng khác. Việc thông tin liên lạc đó thường được gọi là end-to-end. Mức chuyên trở có thể điều khiển luông thông tin. Nó cũng có thể cung cấp sự giao vận có độ tin cậy, bảo đảm dữ liệu đến nơi mà không có lỗi và theo đúng thứ tự. Để làm được điều đó, phần mềm protocol lớp giao vận cung cấp giao thức TCP, trong quá trình trao đổi thông tin nơi nhận sẽ gửi ngược trở lại một xác nhận (ACK) và nơi gửi sẽ truyền lại những gói dữ liệu bị mất. Tuy nhiên trong những môi trường truyền dẫn tốt như cáp quang chẳng hạn thì việc xy ra lỗi là rất nhỏ. Lớp giao vận có cung cấp một giao thức khác đó là UDP.
    - *Lớp Internet:* Nhiệm vụ cơ bản của lớp này là xử lý việc liên lạc của các thiết bị trên mạng. Nó nhận được một yêu cầu để gửi gói dữ liệu từ lớp cùng với một định danh của máy mà gói dữ liệu phi được gửi đến. Nó đóng segment vào trong một packet, điền vào phần đầu của packet, sau đó sử dụng các giao thức định tuyến để chuyển gói tin đến được đích của nó hoặc trạm kế tiếp. Khi đó tại nơi nhận sẽ kiểm tra tính hợp lệ của chúng, và sử dụng tiếp các giao thức định tuyến để xử lý gói tin. Đối với những packet được xác định thuộc cùng mạng cục bộ, phần mềm Internet sẽ cắt bỏ phần đầu của packet, và chọn một trong các giao thức lớp chuyên trở thích hợp để xử lý chúng. Cuối cùng, lớp Internet gửi và nhận các thông điệp kiểm soát và sử lý lỗi ICMP.
    - *Lớp giao tiếp mạng:* Lớp thấp nhất của mô hình TCP/IP chính là lớp giao tiếp mạng, có trách nhiệm nhận các IP datagram và truyền chúng trên một mạng nhất định. Người ta lại chia lớp giao tiếp mạng thành 2 lớp con là:
      * Lớp vật lý: Lớp vật lý làm việc với các thiết bị vật lý, truyền tới dòng bit 0, 1 từ ni gửi đến nơi nhận.
      * Lớp liên kết dữ liệu: Tại đây dữ liệu được tổ chức thành các khung (frame). Phần đầu khung chứa địa chỉ và thông tin điều khiển, phần cuối khung dành cho viêc phát hiện lỗi.

### MÔ HÌNH TRUYỀN THÔNG TRONG CẤU TRÚC MẠNG

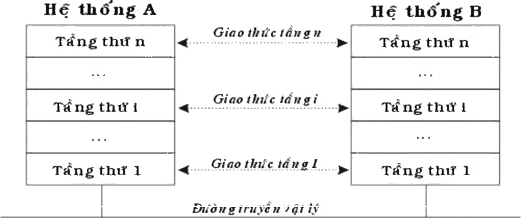
### Nguyên tắc truyền thông

 *Để một mạng máy tính trở một môi trƣờng truyền dữ liệu thì nó cần phải có những yếu tố sau:*

* + - * Các hệ thống được liên kết với nhau theo một cấu trúc kết nối (topology) nào đó
      * Việc chuyển dữ liệu từ máy tính này đến máy tính khác do mạng thực hiện thông qua những quy định thống nhất gọi là giao thức của mạng.
      * Phân chia hoạt động truyền thông của hệ thống thành nhiều lớp theo các nguyên tắc nhất định
      * Việc xét các module một cách độc lập với nhau cho phép giảm độ phức tạp cho việc thiết kế và cài đặt. Phương pháp này được sử dụng rộng rãi trong việc xây dựng mạng và các chương trình truyền thông và được gọi là phương pháp phân tầng (layer).

### Nguyên tắc của phƣơng pháp phân tầng

* + - * Mỗi hệ thống thành phần trong mạng được xây dựng như một cấu trúc nhiều tầng và đều có cấu trúc giống nhau như: số lượng tầng và chức năng của mỗi tầng.
      * Các tầng nằm chồng lên nhau, dữ liệu được chỉ trao đổi trực tiếp giữa hai tầng kề nhau từ tầng trên xuống tầng dưới và ngược lại.
      * Cùng với việc xác định chức năng của mỗi tầng chúng ta phải xác định mối quan hệ giữa hai tầng kề nhau. Dữ liệu được truyền đi từ tầng cao nhất của hệ thống truyền lần lượt đến tầng thấp nhất sau đó truyền qua đường nối vật lý dưới dạng các bit tới tầng thấp nhất của hệ thống nhận, sau đó dữ liệu được truyền ngược lên lần lượt đến tầng cao nhất của hệ thống nhận.
      * Chỉ có hai tầng thấp nhất có liên kết vật lý với nhau còn các tầng trên cùng thứ tư chỉ có các liên kết logic với nhau. Liên kết logic của một tầng được thực hiện thông qua các tầng dưới và phải tuân theo những quy định chặt chẽ, các quy định đó được gọi giao thức của tầng.



*Hình I - 3. Mô hình truyền thông đơn giản 3 tầng*

* + - * Trong kiến trúc phân tầng, một số mô hình được phát triển o Mô hình tham chiếu kết nối các hệ thống mở OSI

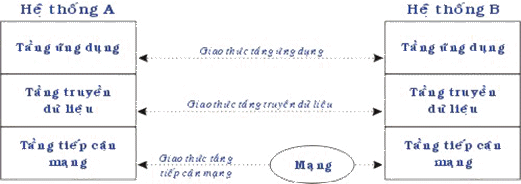
o Mô hình Internet : TCP/IP

* + - * Xét trên phương diện lập trình với máy tính, ta xét mô hình phân tầng thu gọn: Nói chung trong truyền thông có sự tham gia của các thành phần: các chương trình ứng dụng, các chương trình truyền thông, các máy tính và các mạng. Các chương trình ứng dụng là các chương trình của người sử dụng được thực hiện trên máy tính và có thể tham gia vào quá trình trao đổi thông tin giữa hai máy tính. Trên một máy tính với hệ điều hành đa nhiệm (như Windows, UNIX) thường được thực hiện đồng thời nhiều ứng dụng trong đó có những ứng dụng liên quan đến mạng và các ứng dụng khác. Các máy tính được nối với mạng và các dữ liệu được trao đổi thông qua mạng từ máy tính này đến máy tính khác.
      * Việc gửi dữ liệu được thực hiện giữa một ứng dụng với một ứng dụng khác trên hai máy tính khác nhau thông qua mạng được thực hiện như sau: Ứng dụng gửi chuyển dữ liệu cho chương trình truyền thông trên máy tính của nó, chương trình truyền thông sẽ gửi chúng tới máy tính nhận. Chương trình truyền thông trên máy nhận sẽ tiếp nhận dữ liệu, kiểm tra nó trước khi chuyển giao cho ứng dụng đang chờ dữ liệu.
      * Với mô hình truyền thông đơn giản người ta chia chương trình truyền thông thành ba tầng không phụ thuộc vào nhau là: tầng ứng dụng, tầng giao vận và tầng tiếp cận mạng.
      * *Tầng tiếp cận mạng* liên quan tới việc trao đổi dữ liệu giữa máy tính và mạng mà nó được nối vào. Để dữ liệu đến được đích máy tính gửi cần phải chuyển địa chỉ của máy tính nhận cho mạng và qua đó mạng sẽ chuyển các thông tin tới đích. Ngoài ra máy gửi có thể sử dụng một số phục vụ khác nhau mà mạng cung cấp như gửi ưu tiên, tốc độ cao. Trong tầng này có thể có nhiều phần mềm
      * *Tầng giao vận:* thực hiện quá trình truyền thông end-to-end giữa 2 ứng dụng không liên quan tới mạng và nằm ở trên tầng tiếp cận mạng. Tầng truyền dữ liệu không quan tâm tới bản chất các ứng dụng đang trao đổi dữ liệu mà quan tâm tới làm sao cho các dữ liệu được trao đổi một cách an toàn. Tầng truyền dữ liệu đảm bảo các dữ liệu đến được đích và đến theo đúng thứ tự mà chúng được xử lý.

Trong tầng truyền dữ liệu người ta phải có những cơ chế nhằm đảm bảo

sự chính xác đó và rõ ràng các cơ chế này không phụ thuộc vào bản chất của từng ứng dụng và chúng sẽ phục vụ cho tất cả các ứng dụng.

* + - * *Tầng ứng dụng* sẽ chứa các module phục vụ cho tất cả những ứng dụng của người sử dụng. Với các loại ứng dụng khác nhau (như là truyền file, truyền thư mục) cần các module khác nhau.



*Hình I - 4. Trao đổi giữa các tầng*

### ĐỊA CHỈ IP – CÁC ĐỊA CHỈ IP DÀNH RIÊNG

* + - Mỗi địa chỉ IP có độ dài 32 bits (đối với IP4) được tách thành 4 vùng (mỗi vùng 1 byte), có thể được biểu thị dưới dạng thập phân, bát phân, thập lục phân hoặc nhị phân. Cách viết phổ biến nhất là dùng ký pháp thập phân có dấu chấm để tách giữa các vùng. Địa chỉ IP là để định danh duy nhất cho một host bất kỳ trên liên mạng. Khuôn dạng địa chỉ IP: mỗi host trên mạng TCP/IP được định danh duy nhất bởi một địa chỉ có khuôn dạng <Network Number, Host number>
    - Do tổ chức và độ lớn của các mạng con của liên mạng có thể khác nhau, người ta chia các địa chỉ IP thành 5 lớp ký hiệu A,B,C, D, E. Các bit đầu tiên của byte đầu tiên được dùng để định danh lớp địa chỉ (0-lớp A; 10 lớp B; 110 lớp C; 1110 lớp D; 11110 lớp E).Subneting
    - Trong nhiều trường hợp, một mạng có thể được chia thành nhiều mạng con(subnet), lúc đó có thể đưa thêm các vùng subnetid để định danh các mạng con. Vùng subnetid được lấy từ vùng hostid, cụ thể đối với 3 lớp A, B, C .

### GIAO THỨC TRUYỀN FILE FTP

* + 1. **Khái niệm về giao thức FTP**

- FTP (viết tắt của tiếng Anh File Transfer Protocol, "Giao thức truyền tập tin") thường được dùng để trao đổi tập tin qua mạng lưới truyền thông dùng giao thức TCP/IP (chẳng hạn như Internet - mạng ngoại bộ - hoặc intranet - mạng nội bộ). Hoạt động của FTP cần có hai máy tính, một máy chủ và một máy khách). Máy chủ FTP, dùng chạy phần mềm cung cấp dịch vụ FTP, gọi là trình chủ, lắng nghe yêu cầu về dịch vụ của các máy tính khác trên mạng lưới. Máy khách chạy phần mềm FTP dành cho người sử dụng dịch vụ, gọi là trình khách, thì khởi đầu một liên kết với máy chủ. Một khi hai máy đã liên kết với nhau, máy khách có thể xử lý một số thao tác về tập tin, như tải tập tin lên máy chủ, tải tập tin từ máy chủ xuống máy của mình, đổi tên của tập tin, hoặc xóa tập tin ở máy chủ v.v. Vì giao thức FTP là một giao thức chuẩn công khai, cho nên bất cứ một công ty phần mềm nào, hay một lập trình viên nào cũng có thể viết trình chủ FTP hoặc trình khách FTP. Hầu như bất cứ một nền tảng

hệ điều hành máy tính nào cũng hỗ trợ giao thức FTP. Điều này cho phép tất cả các máy tính kết nối với một mạng lưới có nền TCP/IP, xử lý tập tin trên một máy tính khác trên cùng một mạng lưới với mình, bất kể máy tính ấy dùng hệ điều hành nào (nếu các máy tính ấy đều cho phép sự truy cập của các máy tính khác, dùng giao thức FTP). Hiện nay trên thị trường có rất nhiều các trình khách và trình chủ FTP, và phần đông các trình ứng dụng này cho phép người dùng được lấy tự do. - FTP thường chạy trên hai cổng, 20 và 21, và chỉ chạy riêng trên nền của TCP. Trình chủ FTP lắng nghe các yêu cầu dịch vụ từ những kết nối vào máy của các trình khách FTP, trên cổng 21. Đường kết nối trên cổng 21 này tạo nên một dòng truyền điều khiển, cho phép các dòng lệnh được chuyển qua trình chủ FTP. Để truyền tải tập tin qua lại giữa hai máy, chúng ta cần phải có một kết nối khác. Tùy thuộc vào chế độ truyền tải được sử dụng, trình khách (ở chế độ năng động - active mode) hoặc trình chủ (ở chế độ bị động - passive mode) đều có thể lắng nghe yêu cầu kết nối đến từ đầu kia của mình. Trong trường hợp kết nối ở chế độ năng động, (trình chủ kết nối với trình khách để truyền tải dữ liệu) , trình chủ phải trước tiên đóng kết vào cổng 20, trước khi liên lạc và kết nối với trình khách. Trong chế độ bị động, hạn chế này được giải tỏa, và việc đóng kết trước là một việc không cần phải làm.

* + 1. **Đặc điểm và nguyên lý hoạt động**

#### Mô hình hoạt động của FTP

* + - * + Giao thức FTP được mô tả một cách đơn giản thông qua mô hình hoạt động của FTP. Mô hình này chỉ ra các nguyên tắc mà một thiết bị phải tuân theo khi tham gia vào quá trình trao đổi file, cũng như về hai kênh thông tin cần phải thiết lập giữa các thiết bị đó. Nócũng mô tả các thành phần của FTP được dùng để quản lý các kênh này ở cả hai phía – truyền và nhận.
        + Do đó, mô hình này tạo cho ta một khởi điểm lý tưởng để xem xét hoạt động của FTP ở mức khái quát

 *Tiến trình Server-FTP và User-FTP*

FTP là một giao thức dạng client/server truyền thống, tuy nhiên thuật ngữ client thông thường được thay thế bằng thuật ngữ user – người dùng – do thực tế

là người sử dụng mới là đối tượng trực tiếp thao tác các lệnh FTP trên máy clients. Bộ phần mềm FTP được cài đặt trên một thiết bị được gọi là một tiến trình. Phần mềm FTP được cài đặt trên máy Server được gọi là tiến trình Server-FTP, và phần trên máy client được gọi là tiến trình User-FTP.

 *Kênh điều khiển và kênh dữ liệu trong FTP*

Một khái niệm cốt yếu mà ta cần phải nắm về FTP là: mặc dù giao thức này sử dụng kết nối TCP, nhưng nó không chỉ dùng một kênh TCP như phần lớn các giao thức truyền thông khác.

Mô hình FTP chia quá trình truyền thông giữa bộ phận Server với bộ phận client ra làmhai kênh logic: o Kênh điều khiển: đây là kênh logic TCP được dùng để khởi tạo một phiên kết nối FTP. Nó được duy trì xuyên suốt phiên kết nối FTP và được sử dụng chỉ để truyền các thông tin điều khiển, như các lệnh và các hồi đáp trong

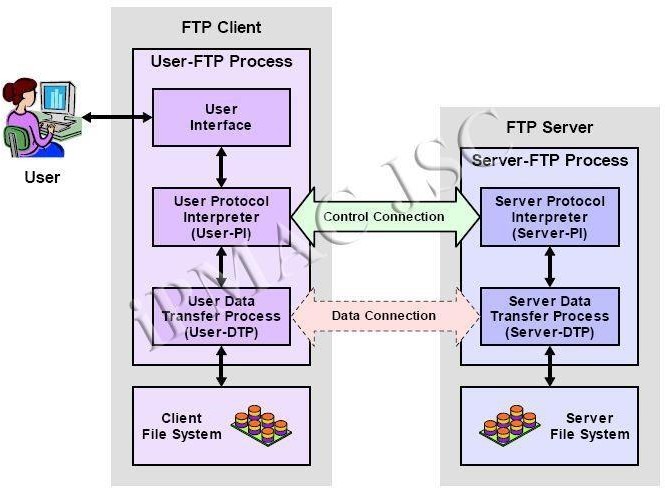
FTP. Nó không được dùng để truyền file o Kênh dữ liệu: Mỗi khi dữ liệu được truyền từ server tới client, một kênh kết nối TCP nhất định lại được khởi tạo giữa chúng. Dữ liệu được truyền đi qua kênh kết nối này – do đó nó được gọi là kênh dữ liệu. Khi file được truyền xong, kênh này được ngắt. Việc sử dụng các kênh riêng lẻ như vậy tạo ra sự linh hoạt trong việc truyền truyền dữ liệu – mà ta sẽ thấy trong các phần tiếp theo. Tuy nhiên, nó cũng tạo cho FTP độ phức tạp nhất định.

 *Các tiến trình và thuật ngữ trong FTP*

Do các chức năng điều khiển và dữ liệu sử dụng các kênh khác nhau, nên mô hình hoạt động của FTP cũng chia phần mềm trên mỗi thiết bị ra làm hai thành phần logic tương ứng với mỗi kênh. Thành phần Protocol Interpreter (PI) là thành phần quản lý kênh điều khiển, với chức năng phát và nhận lệnh. Thành phần Data Transfer Process (DTP) có chức năng gửi và nhận dữ liệu giữa phía client với server.

Ngoài ra, cung cấp cho tiến trình bên phía người dùng còn có thêm thành phần thứ ba là giao diện người dùng FTPthành phần này không có ở phía server. Do

đó, có hai tiến trình xảy ra ở phía server, và ba tiến trình ở phía client. Các tiến trình này được gắn với mô hình FTP để mô tả chi tiết hoạt động của giao thức FTP. Dưới đây là hình đối chiếu các tiến trình vào trong mô hình FTP:



*Hình I - 5. Các tiến trình trong mô hình FTP*

*Các tiến trình phía server:*

Các tiến trình phía server bao gồm hai giao thức: o Server Protocol Interpreter (Server-PI): chịu trách nhiệm quản lý kênh điều khiển trên server. Nó lắng nghe yêu cầu kết nối hướng tới từ users trên cổng dành riêng. Khi kết nối đã được thiết lập, nó sẽ nhận lệnh từ phía User-PI, trả lời lại, và quản lý tiến trình truyền dữ liệu trên server.

Server DataTransfer Process (Server-DTP): làm nhiệm vụ gửi hoặc nhận file từ bộ phận User-DTP. Server-DTP vừa làm nhiệm thiết lập kết nối kênh dữ liệu và lắng nghe một kết nối kênh dữ liệu từ user. Nó tương tác với server file trên hệ thống cục bộ để đọc và chép file.

*Các tiến trình phía client:*

User Protocol Interpreter (User-PI): chịu trách nhiệm quản lý kênh điều khiển phía client. Nó khởi tạo phiên kết nối FTP bằng việc phát ra yêu cầu tới phía Server-PI. Khi kết nối đã được thiết lập, nó xử lý các lệnh nhận

được trên giao diện người dùng, gửi chúng tới Server-PI, và nhận phản hồi trở lại. Nó cũng quản lý tiến trình User-DTP. o User Data Transfer Process (User-DTP): là bộ phận DTP nằm ở phía người dùng, làm nhiệm vụ gửi hoặc nhận dữ liệu từ Server-DTP. User-

DTP có thể thiết lập hoặc lắng nghe yêu cầu kết nối kênh dữ liệu trên server. Nó tương tác với thiết bị lưu trữ file phía client. o User Interface: cung cấp giao diện xử lý cho người dùng. Nó cho phép sử dụng các lệnh đơn giản hướng người dùng, và cho phép người điều khiển phiên FTP theo d i được các thông tin và kết quả xảy ra trong tiến trình.

#### Thiết lập kênh điều khiển và chứng thực người dung trong FTP:

* + - * + Mô hình hoạt động của FTP mô tả rõ các kênh dữ liệu và điều khiển được thiết lập giữa FTP client và FTP server. Trước khi kết nối được sử dụng để thực sự truyền file, kênh điều khiển cần phải được thiết lập. Một tiến trình chỉ định sau đó được dùng để tạo kết nối và tạo ra phiên FTP lâu bền giữa các thiết bị để truyền files.
        + Như trong các giao thức client/server khác, FTP server tuân theo một luật passive trong kênh điều khiển. Bộ phận Server Protocol Interpreter (Server-PI) sẽ lắng nghe cổng TCP dành riêng cho kết nối FTP là cổng 21. Phía User-PI sẽ tạo kết nối bằng việc mở một kết nối TCP từ thiết bị người dùng tới server trên cổng đó. Nó sử dụng một cổng bất kỳ làm cổng nguồn trong phiên kết nối TCP.
        + Khi TCP đã được cài đặt xong, kênh điều khiển giữa các thiết bị sẽ được thiết lập, cho phép các lệnh được truyền từ User-PI tới Server-PI, và Server-PI sẽ đáp trả kết quả là các mã thông báo. Bước đầu tiên sau khi kênh đã đi vào hoạt động là bước đăng nhập của người dùng (login sequence). Bước này có hai mục đích:

o Access Control - Điều khiển truy cập: quá trình chứng thực cho phép hạn chế truy cập tới server với những người dùng nhất định. Nó cũng cho phép server điều khiển loại truy cập như thế nào đối với từng người dùng. o Resource Selection - Chọn nguồn cung cấp: Bằng việc nhận dạng người dùng tạo kết nối,

FTP server có thể đưa ra quyết định sẽ cung cấp những nguồn nào cho người dùng đã được nhận dạng đó.

 *Trình tự truy cập và chứng thực FTP*

Quy luật chứng thực trong FTP khá đơn giản, chỉ là cung cấp username/password.Trình tự của việc chứng thực như sau:

* Người dùng gửi một username từ User-PI tới Server-PI bằng lệnh USER. Sau đó password của người dùng được gửi đi bằng lệnh PASS.
* Server kiểm tra tên người dùng và password trong database người dùng của nó. Nếu người dùng hợp lệ, server sẽ gửi trả một thông báo tới người dùng rằng phiên kết nối đã được mở. Nếu người dùng không hợp lệ, server yêu cầu người dùng thực hiện lại việc chứng thực. Sau một số lần chứng thực sai nhất định, server sẽ ngắt kết nối.
* Giả sử quá trình chứng thực đã thành công, server sau đó sẽ thiết lập kết nối để cho phép từng loại truy cập đối với người dùng được cấp quyền. Một số người dùng chỉ có thể truy cập vào một số file nhất định, hoặc vào một số loại file nhất định. Một số server có thể cấp quyền cho một số người dùng đọc và viết lên server, trong khi chỉ cho phép đọc đối với những người dùng khác. Người quản trị mạng có thể nhờ đó mà đáp ứng đúng các nhu cầu truy cập FTP.
* Một khi kết nối đã được thiết lập, server có thể thực hiện các lựa chọn tài nguyên dựa vào nhận diện người dùng. Ví dụ: trên một hệ thống nhiều người dùng, người quản trị có thể thiết lập FTP để khi có bất cứ người dùng nào kết nối tới, anh ta sẽ tự động được đưa tới "home directory" của chính anh ta. Lệnh tùy chọn ACCT (account) cũng cho phép người dùng chọn một tài khoản cá nhân nào đó nếu như anh ta có nhiều hơn một tài khoản.

 *Mở rộng về bảo mật FTP*

Giống như phần lớn các giao thức cũ, phương pháp đăng nhập đơn giản của FTP là một sự kế thừa từ những giao thức ở thời kỳ đầu của Internet. Ngày nay, nó không còn bảo đảm tính an toàn cần thiết trên môi trường Internet toàn cầu

vì username và password được gửi qua kênh kết nối điều khiển dưới dạng clear text. Điều này làm cho các thông tin đăng nhập có thể bị nghe lén. Chuẩn RFC 2228 về các phần mở rộng cho bảo mật FTP đã định ra thêm nhiều tùy chọn chứng thực và mã hóa phức tạp cho những ai muốn tăng thêm mức độ an toàn vào trong phần mềm FTP của họ.

#### Quản lí kênh dữ liệu của FTP:

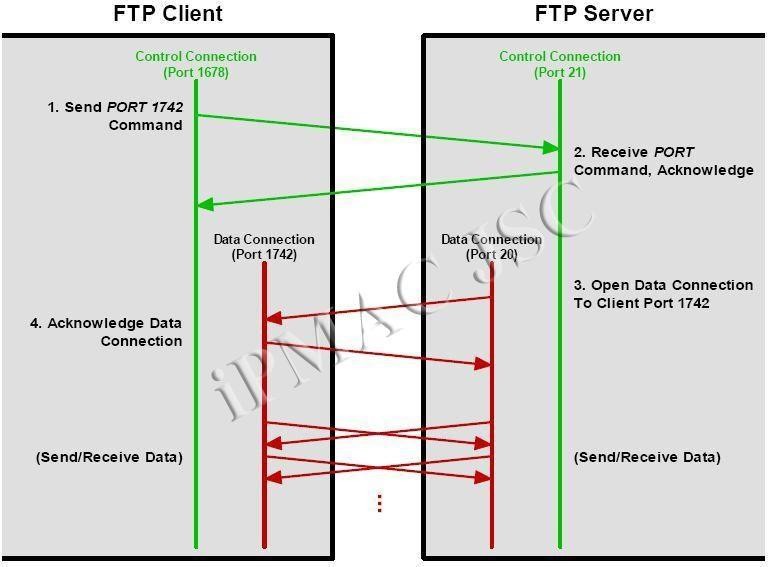
* + - * + - Kênh điều khiển được tạo ra giữa Server-PI và User-PI sử dụng quá trình thiết lập kết nối và chứng thực được duy trì trong suốt phiên kết nối FTP. Các lệnh và các hồi đáp được trao đổi giữa bộ phận PI (Protocol Interpreter) qua kênh điều khiển, nhưng dữ liệu thì không.
        + Mỗi khi cần phải truyền dữ liệu giữa server và client, một kênh dữ liệu cần phải được tạo ra. Kênh dữ liệu kết nối bộ phận User-DTP với Server-DTP. Kết nối này cần thiết cho cả hoạt động chuyển file trực tiếp (gửi hoặc nhận một file) cũng như đối với việc truyền dữ liệu ngầm, như là yêu cầu một danh sách sách file trong thư mục nào đó trên server.
        + Chuẩn FTP chỉ định hai phương thức khác nhau để tạo ra kênh dữ liệu. Khác biệt chính của hai phương thức đó là ở mặt thiết bị: phía client hay phía server là phía đã đưa ra yêu cầu khởi tạo kết nối. Điều này nghe qua có vẻ khá đơn giản, nhưng kỳ thực nó lại khá quan trọng.

 *Kết nối kênh dữ liệu dạng chủ động*

Phương thức đầu tiên đôi khi còn được gọi là kết nối kênh dữ liệu dạng thông thường (vì nó là phương pháp mặc định) và đôi khi được gọi là kết nối dạng chủ động (để đối chiếu với dạng kết nối bị động mà ta sẽ xét ở phần sau). Trong dạng kết nối này, phía Server-DTP khởi tạo kệnh dữ liệu bằng việc mở một cổng TCP cho phía User-DTP. Phía server sử dụng cổng được dành riêng, là cổng 20 cho kênh dữ liệu. Trên máy client, một giá trị cổng được chọn theo mặc định chính là cổng được sử dụng đối với kênh điều khiển, tuy nhiên phía client sẽ luôn chọn hai cổng riêng biệt cho hai kênh này.

Giả sử phía User-PI thiết lập một kết nối điều khiển từ cổng bất kỳ của nó là 1678 tới cổng điều khiển trên server là cổng 21. Khi đó, để tạo một kênh dữ liệu cho việc truyền dữ liệu, phía Server-PI sẽ báo cho phía Server-DTP khởi tạo một kênh kết nối TCP từ cổng 20 tới cổng 1678 của phía client. Sau khi phía client chấp nhận kênh được khởi tạo, dữ liệu sẽ được truyền đi.

Thực tế, việc sử dụng cùng một cổng cho cả kênh dữ liệu và kênh điều khiển không phải là một ý hay, nó làm cho hoạt động của FTP trở nên phức tạp. Do đó, phía client nên chỉ định sử dụng một cổng khác bằng việc sử dụng lệnh PORT trước khi truyền dữ liệu. Ví dụ: giả sử phía client chỉ định cổng 1742 với lệnh PORT. Phía Server-DTP sau đó sẽ tạo ra một kết nối từ cổng 20 của nó tới cổng 1742 phía client thay vì cổng 1678 như mặc định. Quá trình này được mô tả trong hình dưới đây.



*Hình I - 6. Tiến trình làm việc dạng chủ động trong FTP*

Thông thường, đối với kênh dữ liệu FTP, phía server sẽ khởi tạo việc truyền dữ liệu bằng cách mở kết nối dữ liệu tới client.

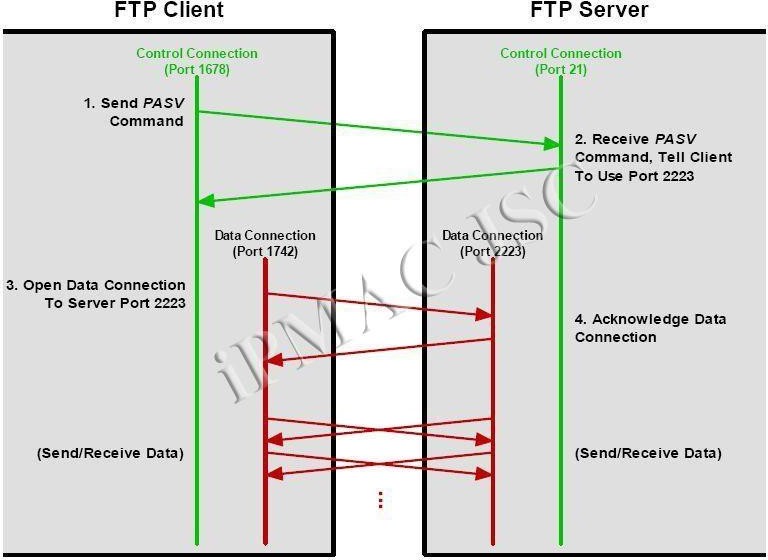
Trong trường hợp trên, phía client trước tiên sẽ đưa ra lệnh PORT để yêu cầu server sử dụng cổng 1742. Sau đó, server sẽ mở kết nối kênh dữ liệu từ cổng 20

mặc định của nó tới cổng 1742 phía client. Dữ liệu sau đó sẽ được truyền giữa các thiết bị qua các cổng này.

 *Kết nối kênh dữ liệu dạng bị động*

Phương pháp kế tiếp được gọi là kết nối dữ liệu dạng bị động. Phía client sẽ nhận server là phía bị động, làm nhiệm vụ chấp nhận một yêu cầu kết nối kênh Dữ liệu được khởi tạo từ phía client. Server trả lời lại phía client với địa chỉ IP cũng như địa chỉ cổng mà nó sẽ sử dụng. Phía Server-DTP sau đó sẽ lắng

nghe một kết nối TCP từ phía User-DTP trên cổng này.

Mặc định, phía client sử dụng cùng một cổng đối với cả hai kênh điều khiển và dữ liệu như trong trường hợp kết nối chủ động ở trên. Tuy nhiên, ở đây, một lần nữa phía client có thể chọn sử dụng một giá trị cổng khác cho kênh dữ liệu. Ta sẽ xét lại ví dụ ở trên một lần nữa, với cổng điều khiển phía client là 1678 tới cổng 21 phía server. Nhưng lần này truyền dữ liệu theo phương thức kết nối bị động, như mô tả trong hình dưới đây:

*Hình I - 7. Tiến trình làm việc dạng bị động trong FTP*

Phía client sẽ sử dụng lệnh PASV để yêu cầu server rằng nó muốn dùng phương thức điều khiển dữ liệu bị động. Phía Server-PI sẽ trả lời lại phía client

với một giá trị cổng mà client sẽ sử dụng, từ cổng 2223 trên nó. Sau đó phía Server PI sẽ hướng cho phía Server-DTP lắng nghe trên cổng 2223. Phía UserPI cũng sẽ hướng cho phía User-DTP tạo một phiên kết nối từ cổng 1742 phía client tới cổng 2223 phía server. Sau khi Server chấp nhận kết nối này, dữ liệu bắt đầu được truyền đi.

 *Các vấn đề về tính hiệu quả và tính bảo mật trong việc chọn một phƣơng thức kết nối*

Vấn đề phía nào là phía khởi tạo kết nối kênh dữ liệu đưa ra một câu hỏi: sự khác nhau giữa hai phương thức là gì Điều này cũng giống như việc hỏi ai đã thực hiện một cuộc điện thoại nội bộ. Câu trả lời là sự bảo mật. Việc FTP sử dụng nhiều hơn một kết nối TCP có thể giải quyết các vấn đề về phần mềm cũng như về phần cứng mà người dùng cần phải có để đảm bảo sự an toàn cho hệ thống của họ.Khi xem xét việc gì sẽ xảy ra trong trường hợp kênh dữ liệu chủ động như trong *ví dụ* phía trên:

Đối với phía client, có một kênh kết nối điều khiển được thiết lập từ cổng 1678 client tới cổng 21 server. Nhưng kênh dữ liệu lại được khởi tạo từ phía server. Do đó, client sẽ nhận được một yêu cầu kết nối tới cổng 1678 (hoặc cổng nào khác). Một số client sẽ nghi ngờ về việc nhận được những kết nối tới như vậy, vì trong tình huống thông thường, client mới là phía khởi tạo kết nối chứ không phải đáp trả kết nối. Do các kênh kết nối TCP hướng tới có thể mang theo những mối đe dọa nhất định, một số client có thể sẽ ngăn chặn các luồng kết nối hướng tới bằng việc sử dụng tường lửa.

Tại sao người ta lại không làm cho phía client luôn chấp nhận kết nối từ một chỉ số port được dùng trong kênh điều khiển ? Vấn đề ở đây là vì client thường dùng các cổng khác nhau cho mỗi phiên kết nối bằng việc sử dụng câu lệnh PORT. Và tại sao điều này lại được thực hiện ? Vì theo luật TCP: sau khi một kết nối được đóng lại , có một khoảng thời gian trống trước khi cổng đó có thể được sử dụng lại – điều này để ngăn ngừa tình trạng các phiên kết nối liên tiếp bị lẫn với nhau. Điều này sẽ tạo ra độ trễ khi gửi nhiều file – do đó phía client thường dùng các giá trị cổng khác nhau cho mỗi kết nối. Điều này rất hiệu quả

nhưng cũng dẫn tới việc firewall của client sẽ hỏi có chấp nhận phiên kết nối tới với nhiều giá trị cổng không ổn định hay không.

Việc dùng kết nối kiểu kênh gián tiếp sẽ giảm thiểu vấn đề này một cách hiệu quả. Phần lớn các tường lửa có nhiều vấn đề liên quan tới kết nối hướng về với các giá trị cổng bất kỳ, hơn là gặp vấn đề với các kết nối hướng đi. Ta có thể xem chi tiết hơn về vấn đề này trong chuẩn RFC 1579. Chuẩn này khuyến nghị rằng phía client nên sử dụng kết nối kiểu bị động làm dạng mặc định thay vì sử dụng kiểu kết nối dạng chủ động cùng với lệnh PORT, để ngăn chặn tình trạng block theo cổng.

Tất nhiên, phương thức kết nối kiểu bị động không hoàn toàn giải quyết được vấn đề, chúng chỉ đẩy vấn đề về phía server mà thôi. Phía server, giờ đây phải đối mặt với việc có nhiều kênh kết nối hướng về trên hàng loạt các cổng khác nhau. Tuy nhiên việc xử lý các vấn đề bảo mật trên một nhóm nhỏ server vẫn dễ hơn nhiều so với việc phải đối mặt với một lượng lớn các vấn đề từ nhiều client. FTP server phải được cấu hình chấp nhận phương thức truyền bị động từ client, do đó cách thông thường để thiết lập trên server là thiết lập chấp nhận một số cổng kết nối hướng về trên server trong khi vẫn khóa các yêu cầu kết nối hướng về trên các cổng khác.

#### Các phương thức truyền dữ liệu trong FTP

* + - * + Khi kênh dữ liệu đã được thiết lập xong giữa Server-DTP với User-DTP, dữ liệu sẽ được truyền trực tiếp từ phía client tới phía server, hoặc ngược lại, dựa theo các lệnh được sử dụng. Do thông tin điều khiển được gửi đi trên kênh điều khiển, nên toàn bộ kênh dữ liệu có thể được sử dụng để truyền dữ liệu. (Tất nhiên, hai kênh logic này được kết hợp với nhau ở lớp dưới cùng với tất cả các kết nối TCP/UDP khác giữa hai thiết bị, do đó điều này không hẳn đã cải thiện tốc độ truyền dữ liệu so với khi truyền trên chỉ một kênh – nó chỉ làm cho hai việc truyền dữ liệu và điều khiển trở nên độc lập với nhau mà thôi)
        + FTP có ba phương thức truyền dữ liệu, nêu lên cách mà dữ liệu được truyền từ một thiết bị tới thiết bị khác trên một kênh dữ liệu đã được khởi tạo, đó là: stream mode, block mode, và compressed mode Stream mode
        + Trong phương thức này, dữ liệu được truyền đi dưới dạng các byte không cấu trúc liên tiếp. Thiết bị gửi chỉ đơn thuần đầy luồng dữ liệu qua kết nối TCP tới phía nhận. Không có một trường tiêu đề nhất định được sử dụng trong phương thức này làm cho nó khá khác so với nhiều giao thức gửi dữ liệu rời rạc khác. Phương thức này chủ yếu dựa vào tính tin cậy trong truyền dữ liệu của TCP. Do nó không có cầu trúc dạng header, nên việc báo hiệu kết thúc file sẽ đơn giản được thực hiện việc phía thiết bị gửi ngắt kênh kết nối dữ liệu khi đã truyền xong.
        + Trong số ba phương thưc, stream mode là phương thức được sử dụng nhiều nhất trong triển khai FTP thực tế. Có một số lý do giải thích điều đó. Trước hết, nó là phương thức mặc định và đơn giản nhất, do đó việc triển khai nó là dễ dàng nhất. Thứ hai, nó là phương pháp phổ biến nhất, vì nó xử lý với các file đều đơn thuần như là xử lý dòng byte, mà không để ý tới nội dung của các file. Thứ ba, nó là phương thức hiệu quả nhất vì nó không tốn một lượng byte overload để thông báo header.

 *Block mode*

Đây là phương thức truyền dữ liệu mang tính quy chuẩn hơn, với việc dữ liệu được chia thành nhiều khối nhỏ và được đóng gói thành các FTP blocks. Mỗi block này có một trường header 3 byte báo hiệu độ dài, và chứa thông tin về các khối dữ liệu đang được gửi.

Một thuật toán đặc biệt được sử dụng để kiểm tra các dữ liệu đã được truyền đi và để phát hiện, khởi tạo lại đối với một phiên truyền dữ liệu đã bị ngắt.

 *Compressed mode*

Đây là một phương thức truyền sử dụng một kỹ thuật nén khá đơn giản, là run-length encoding – có tác dụng phát hiện và xử lý các đoạn lặp trong dữ liệu được truyền đi để giảm chiều dài của toàn bộ thông điệp. Thông tin khi đã được nén, sẽ được xử lý như trong block mode, với trường header. Trong thực tế, việc nến dữ liệu thường được sử dụng ở những chỗ khác, làm cho phương thức truyền kiểu compressed mode trở nên không cần thiết nữa. Ví dụ: nếu bạn đang truyền

đi một file qua internet với modem tương tự, modem của bạn thông thường sẽ thực hiện việc nén ở lớp 1; các file lớn trên FTP server cũng thường được nén s n với một số định dạng như IP, làm cho việc nén tiếp tục khi truyền dữ liệu trở nên không cần thiết.

# CHƯƠNG II

# LẬP TRÌNH MẠNG VỚI SOCKET

### TỔNG QUAN VỀ C#

### Cơ Bản Về NET Framework

 NET Framework là một thành phần cơ bản của Windows cho việc xây dựng và chạy các ứng dụng viết bởi các ngôn ngữ lập trình mới (ứng dụng thế hệ kế tiếp). NET Framework được thiết kế để:

* + - * Cung cấp một môi trường nhất quán cho lập trình hướng đối tượng.
      * Tối ưu hóa việc phát triển phần mềm và sự xung đột phiên bản bằng việc cung cấp một môi trường thực hiện code.
      * Cung cấp môi trường thực thi code an toàn hơn.
      * Cung cấp trải nghiệm (experience) nhất quán cho những người phát triển trong việc tạo ra các kiểu ứng dụng khác nhau từ các ứng dụng trên nền tảng Windows, các ứng dụng trên nền tảng Web cho đến các ứng dụng trên nền

tảng thiết bị di động, các ứng dụng nhúng…

### Các thành phần của .NET Framework

 NET Framework bao gồm 2 thành phần chính:

* CLR (Common Language Runtime – Môi trường quản lý ngôn ngữ chung): đây là thành phần cốt lỗi (xương sống – backbone) của NET Framework thực hiện các chức năng sau: o Quản lý bộ nhớ. o Thực hiện code.
  + Xử lý lỗi.
  + Xác nhận sự an toàn của code.
  + Thu gom rác.
* Framework Class Library (FCL): là một tập hợp các kiểu dữ liệu có khả năng sử dụng lại (tập hợp các lớp) và hướng đối tượng hoàn toàn, được sử dụng để phát triển các ứng dụng từ những ứng dụng dòng lệnh truyền thống cho đến những ứng dụng với giao diện đồ họa.

### Các tính năng cơ bản của ngôn ngữ lập trình C#

 C# là một ngôn ngữ lập trình hướng đối tượng thuần túy (pure object oriented programming). Kiểm tra an toàn kiểu.

* Thu gom rác tự động: giảm bớt gánh nặng cho người lập trình viên trong việc phải viết các đoạn code thực hiện cấp phát và giải phóng bộ nhớ.
* Hỗ trợ các chuẩn hóa được ra bởi tổ chức ECMA (European Computer Manufactures Association).
* Hỗ trợ các phương thức và các kiểu phổ quát (chung).

### Các ứng dụng của C#

 C# có thể sử dụng để viết các kiểu ứng dụng khác nhau:

* Các ứng game.
* Các ứng dụng cho doanh nghiệp.
* Các ứng dụng cho thiết bị di động: PC Pocket, PDA , cell phone.
* Các ứng dụng quản lý đơn giản: ứng dụng quản lý thư viện, quản lý thông tin cá nhân…
* Các ứng dụng phân tán phức tạp trải rộng qua nhiều thành phố, đất nước.

### Các lợi ích của C#

* Cross Language Support: hỗ trợ khả năng chuyển đổi dễ dàng giữa các ngôn ngữ.
* Hỗ trợ các giao thức Internet chung.
* Triển khai đơn giản.
* Hỗ trợ tài liệu XML: các chú thích XML có thể được thêm vào các đoạn code và sau đó có thể được chiết xuất để làm tài liệu cho các đoạn code để cho phép các lập trình viên khi sử dụng biết được ý nghĩa của các đoạn code đã viết.

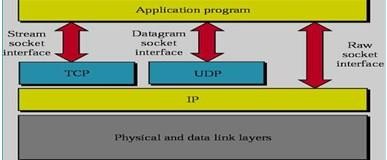
### Môi trường phát triển tích hợp Visual Studio .NET 2017

 VS 2017 là một tập hợp các công cụ phát triển cho việc xây dựng các ứng dụng desktop với hiệu năng cao, các ứng dụng cho thiết bị di động, các dịch vụ Web, các ứng dụng Web. Ngoài ra VS 2017 cũng được sử dụng để làm đơn giản hóa quá trình phát triển nhóm, triển khai cài đặt các ứng dụng enterprise.VS 2017 cung cấp các lợi ích mở rộng cho việc phát triển các ứng dụng:

* Nâng cao tính sản phẩm.
* Phát triển các ứng dụng cho NET Framework 4.6.1
* Phát triển các ứng dụng cho các thiết bị cầm tay với .NET Framework Compact 4.6.

### GIỚI THIỆU VỀ LẬP TRÌNH SOCKET

* + - Sockets cung cấp một interface để lập trình mạng tại tầng Transport. Một socket là một end-point của một liên kết giữa hai ứng dụng. Ngày nay, Socket được hỗ trợ trong hầu hết các hệ điều hành như MS Windows (WinSock), Linux và được sử dụng trong nhiều ngôn ngữ lập trình khác nhau: như C, C++, Java, Visual Basic, C#, . . .
    - Windows Socket Application Programming Interface (Winsock API) là một thư viện các hàm socket. Winsock hỗ trợ các lập trình viên xây dựng các ứng dụng mạng trên nền TCP/IP.
    - Là giao diện lập trình ứng dụng (API), giao diện giữa chương trình ứng dụng với lớp mạng trong hệ thống mạng TCP/IP.Thông qua giao diện này chúng ta có thể lập trình điều khiển việc truyền thông giữa hai máy sử dụng các giao thức mức thấp làTCP,UDP…
    - Thiết lập các lập kênh giao tiếp với mỗi đầu kênh được đánh dấu bằng một cổng. Dữ liệu có thể đi vào và ra khỏi kênh giao tiếp thông qua cổng này.
      * Cơ chế giao tiếp:
        + Một trong hai quá trình phải công bố số hiệu cổng của socket mà mình sử dụng để nhận và gởi dữ liệu.
        + Các quá trình khác có thể giao tiếp với quá trình đã công bố cổng cũng bằng cách tạo ra một socket.
      * Các loại socket:
        + Socket hướng kết nối (TCP Socket)
        + Socket không hướng kết nối (UDP Socket)
        + Raw Socket

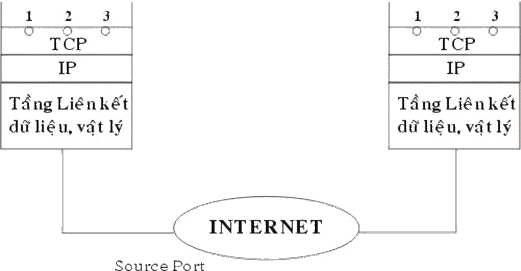


*Hình II - 1. Các giao thức trong lập trình Socket*

### LẬP TRÌNH MẠNG VỚI TCPSOCKET

### Mô hình giao thức

* + - * TCP là một giao thức "có liên kết" (connection - oriented), nghĩa là cần phải thiết lập liên kết giữa hai thực thể TCP trước khi chúng trao đổi dữ liệu với nhau. Một tiến trình ứng dụng trong một máy tính truy nhập vào các dịch vụ của giao thức TCP thông qua một cổng (port) của TCP. Số hiệu cổng TCP được thể hiện bởi 2 bytes.



*Hình II - 2. Kết nối TCP*

* + - * Một cổng TCP kết hợp với địa chỉ IP tạo thành một đầu nối TCP/IP (socket) duy nhất trong liên mạng. Dịch vụ TCP được cung cấp nhờ một liên kết logic giữa một cặp đầu nối TCP/IP. Một đầu nối TCP/IP có thể tham gia nhiều liên kết với các đầu nối TCP/IP ở xa khác nhau. Trước khi truyền dữ liệu giữa 2 trạm cần phải thiết lập một liên kết TCP giữa chúng và khi không còn nhu cầu truyền dữ liệu thì liên kết đó sẽ được giải phóng.
      * Các thực thể của tầng trên sử dụng giao thức TCP thông qua các hàm gọi (function calls) trong đó có các hàm yêu cầu để yêu cầu, để trả lời. Trong mỗi hàm còn có các tham số dành cho việc trao đổi dữ liệu.

### Thiết lập kết nối

* + - * Các bước thực hiện để thiết lập một liên kết TCP/IP: Thiết lập một liên kết mới có thể được mở theo một trong 2 phương thức: chủ động (active) hoặc bị động (passive).
        + Phương thức bị động, người sử dụng yêu cầu TCP chờ đợi một yêu cầu liên kết gửi đến từ xa thông qua một đầu nối TCP/IP (tại chỗ). Người sử dụng dùng hàm passive Open có khai báo cổng TCP và các thông số khác (mức ưu tiên, mức an toàn)
        + Với phương thức chủ động, người sử dụng yêu cầu TCP mở một liên kết với một một đầu nối TCP/IP ở xa. Liên kết sẽ được xác lập nếu có một hàm

Passive Open tương ứng đã được thực hiện tại đầu nối TCP/IP ở xa đó.

*Bảng liệt kê một vài cổng TCP phổ biến.*

|  |  |
| --- | --- |
| Số hiệu cổng | Mô tả |
| 0 | Reserved |
| 5 | Remote job entry |
| 7 | Echo |
| 9 | Discard |
| 11 | Systat |
| 13 | Daytime |

|  |  |
| --- | --- |
| 15 | Nestat |

|  |  |  |  |
| --- | --- | --- | --- |
|  | 17 | quote (quote odd day |  |
| 20 | ftp-data |
| 21 | ftp (control) |
| 23 | Telnet |
| 25 | SMTP |
| 37 | Time |
| 53 | Name Server |
| 102 | ISO – TSAP |
|  |  |
| 104 | X.400 Sending |
| 111 | Sun RPC |
| 139 | Net BIOS Session source |
| 160 – 223 | Reserved |

* + - * Khi người sử dụng gửi đi một yêu cầu mở liên kết sẽ được nhận hai thông số trả lời từ TCP. o Thông số Open ID được TCP trả lời ngay lập tức để gán cho

một liên kết cục bộ (local connection name) cho liên kết được yêu cầu. Thông số này về sau được dùng để tham chiếu tới liên kết đó. (Trong trường hợp nếu TCP không thể thiết lập được liên kết yêu cầu thì nó phải gửi tham số Open

Failure để thông báo.) o Khi TCP thiết lập được liên kết yêu cầu nó gửi tham số Open Success được dùng để thông báo liên kết đã được thiết lập thành công. Thông báo này dược chuyển đến trong cả hai trường hợp bị động và chủ động. Sau khi một liên kết được mở, việc truyền dữ liệu trên liên kết có thể được thực hiện.

### Truyền nhận dữ liệu

* + - * Các bước thực hiện khi truyền và nhận dữ liệu: Sau khi xác lập được liên kết người sử dụng gửi và nhận dữ liệu. Việc gửi và nhận dữ liệu thông qua các hàm Send và receive.
        + *Hàm Send:* Dữ liệu được gửi xuống TCP theo các khối (block). Khi nhận được một khối dữ liệu, TCP sẽ lưu trữ trong bộ đệm (buffer). Nếu cờ PUSH được dựng thì toàn bộ dữ liệu trong bộ đệm được gửi, kể cả khối dữ liệu mới đến sẽ được gửi đi. Ngược lại cờ PUSH không được dựng thì dữ liệu được giữ lại trong bộ đệm và sẽ gửi đi khi có cơ hội thích hợp (chẳng hạn chờ thêm dữ liệu nữa để gửi đi với hiệu quả hơn). o *Hàm receive:* Ở trạm đích dữ liệu sẽ được TCP lưu trong bộ đệm gắn với mỗi liên kết. Nếu dữ liệu được đánh dấu với một cờ PUSH thì toàn bộ dữ liệu trong bộ đệm (kể cả các dữ liệu được lưu từ trước) sẽ được chuyển lên cho người sử dụng. Còn nếu dữ liệu đến không được đánh dấu với cờ PUSH thì TCP chờ tới khi thích hợp mới chuyển dữ liệu với mục tiêu tăng hiệu quả hệ thống.
      * Nói chung việc nhận và giao dữ liệu cho người sử dụng đích của TCP phụ thuộc vào việc cài đặt cụ thể. Trường hợp cần chuyển gấp dữ liệu cho người sử dụng thì có thể dùng cờ URGENT và đánh dấu dữ liệu bằng bit URG để báo cho người sử dụng cần phải sử lý khẩn cấp dữ liệu đó.

### Đóng liên kết

* + - * Các bước thực hiện khi đóng một liên kết: Việc đóng một liên kết khi không cần thiết được thực hiên theo một trong hai cách: dùng hàm Close hoặc dùng hàm Abort. o *Hàm Close*: yêu cầu đóng liên kết một cách bình thường. Có nghĩa là

việc truyền dữ liệu trên liên kết đó đã hoàn tất. Khi nhận được một hàm Close TCP sẽ truyền đi tất cả dữ liệu còn trong bộ đệm thông báo rằng nó đóng liên kết. Lưu ý rằng khi một người sử dụng đã gửi đi một hàm Close thì nó vẫn phải tiếp tục nhận dữ liệu đến trên liên kết đó cho đến khi TCP đã báo

cho phía bên kia biết về việc đóng liên kết và chuyển giao hết tất cả dữ liệu cho người sử dụng của mình. o *Hàm Abort:* Người sử dụng có thể đóng một liên kết bất và sẽ không chấp nhận dữ liệu qua liên kết đó nữa. Do vậy dữ liệu có thể bị mất đi khi đang được truyền đi. TCP báo cho TCP ở xa biết rằng liên kết đã được hủy bỏ và TCP ở xa sẽ thông báo cho người sử dụng của mình.

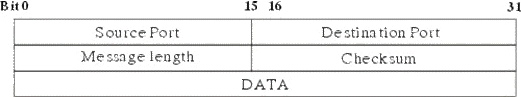
*Một số hàm khác của TCP:*

* *Hàm Status:* cho phép người sử dụng yêu cầu cho biết trạng thái của một liên kết cụ thể, khi đó TCP cung cấp thông tin cho người sử dụng.
* *Hàm Error:* thông báo cho người sử dụng TCP về các yêu cầu dịch vụ bất hợp lệ liên quan đến một liên kết có tên cho trước hoặc về các lỗi liên quan đến môi trường.

### LẬP TRÌNH MẠNG VỚI UDP SOCKET

### Giao thức UDP

* + - * UDP (User Datagram Protocol) là giao thức theo phương thức không liên kết được sử dụng thay thế cho TCP ở trên IP theo yêu cầu của từng ứng dụng. Khác với TCP, UDP không có các chức năng thiết lập và kết thúc liên kết. Tương tự như IP, nó cũng không cung cấp cơ chế báo nhận (acknowledgment), không sắp xếp tuần tự các gói tin (datagram) đến và có thể dẫn đến tình trạng mất hoặc trùng dữ liệu mà không có cơ chế thông báo lỗi cho người gửi. Qua đó ta thấy UDP cung cấp các dịch vụ vận chuyển không tin cậy như trong TCP.
      * Khuôn dạng UDP datagram được mô tả với các vùng tham số đơn giản hơn nhiều so với TCP segment.



*Hình II - 3. Dạng thức của gói tin UDP*

* + - * UDP cũng cung cấp cơ chế gán và quản lý các số hiệu cổng (port number) để định danh duy nhất cho các ứng dụng chạy trên một trạm của mạng. Do ít chức năng phức tạp nên UDP thường có xu thế hoạt động nhanh hơn so với TCP. Nó thường được dùng cho các ứng không đòi hỏi độ tin cậy cao trong giao vận.

### Một số thuật ngữ UDP

* + - * Trước khi kiểm tra xem giao thức UDP hoạt động như thế nào, chúng ta cần làm quen với một số thuật ngữ. Trong phần dưới đây, chúng ta sẽ định nghĩa một số thuật ngữ cơ bản có liên quan đến giao thức UDP. o *Packet:* Trong truyền số liệu, một packet là một dãy các số nhị phân, biểu diễn dữ liệu và các tín hiệu điều khiển, các gói tin này được chuyển đi và chuyển tới tới host. Trong gói tin,thông tin được sắp xếp theo một khuôn dạng cụ thể.
        + *Datagram*: Một datagram là một gói tin độc lập, tự chứa, mang đầy đủ dữ liệu để định tuyến từ nguồn tới đích mà không cần thông tin thêm.
        + *MTU:* là viết tắt của Maximum Transmission Unit. MTU là một đặc trưng của tầng liên kết mô tả số byte dữ liệu tối đa có thể truyền trong một gói tin. Mặt khác, MTU là gói dữ liệu lớn nhất mà môi trường mạng cho trước có thể truyền. Ví dụ, Ethernet có MTU cố định là 1500 byte. Trong UDP, nếu kích thước của một datagram lớn hơn MTU, IP sẽ thực hiện phân đoạn, chia datagram thành các phần nhỏ hơn (các đoạn), vì vậy mỗi đoạn nhỏ có kích thước nhỏ hơn MTU.
        + *Port:* UDP sử dụng các cổng để ánh xạ dữ liệu đến vào một tiến trình cụ thể đang chạy trên một máy tính. UDP định đường đi cho packet tại vị trí xác định bằng cách sử dụng số hiệu cổng được xác định trong header của datagram. Các cổng được biểu diễn bởi các số 16-bit, vì thế các cổng nằm

trong dải từ 0 đến 65535. Các cổng cũng được xem như là các điểm cuối của các liên kết logic, và được chia thành ba loại sau:

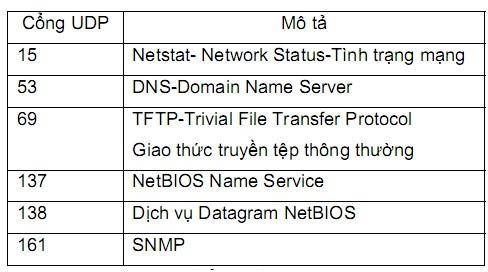
Các cổng phổ biến: Từ 0 đến 1023

Các cổng đã đăng ký: 1024 đến 49151

Các cổng động/dành riêng 49152 đến 65535

* + - * Chú ý rằng các cổng UDP có thể nhận nhiều hơn một thông điệp ở một thời điểm. Trong một số trường hợp, các dịch vụ TCP và UDP có thể sử dụng cùng một số hiệu cổng, như 7 (Echo) hoặc trên cổng 23 (Telnet).

*UDP có các cổng thông dụng sau:*



* + - * + *TTL (Time To Live)* Giá trị TTL cho phép chúng ta thiết lập một giới hạn trên của các router mà một datagram có thể đi qua. Giá trị TTL ngăn ngừa các gói tin khỏi bị kẹt trong các vòng lặp định tuyến vô hạn. TTL được khởi tạo bởi phía gửi và giá trị được giảm đi bởi mỗi router quản lý datagram. Khi TTL bằng 0, datagram bị loại bỏ.
        + Multicasting: là phương pháp dựa trên chuẩn có tính chất mở để phân phối các thông tin giống nhau đến nhiều người dùng. Multicasting là một đặc trưng chính của giao thức UDP. Multicasting cho phép chúng ta truyền tin theo kiểu một nhiều, ví dụ gửi tin hoặc thư điện tử tới nhiều người nhận, đài phát thanh trên Internet, hoặc các chương trình demo trực tuyến.

### Hoạt động của giao thức UDP

Khi một ứng dụng dựa trên giao thức UDP gửi dữ liệu tới một host khác trên mạng, UDP thêm vào một header có độ dài 8 byte chứa các số hiệu cổng nguồn và đích, cùng với tổng chiều dài dữ liệu và thông tin checksum. IP thêm vào header của riêng nó vào đâu mỗi datagram UDP để tạo lên một datagram IP:

### Các nhược điểm của giao thức UDP

 So với giao thức TCP, UDP có những nhược điểm sau: o Thiếu các tín hiệu bắt tay. Trước khi gửi một đoạn, UDP không gửi các tín hiệu bắt tay giữa bên gửi và bên nhận. Vì thế phía gửi không có cách nào để biết datagram đã đến đích hay chưa. Do vậy, UDP không đảm bảo việc dữ liệu đã đến đích hay chưa.

* Sử dụng các phiên. Để TCP là hướng liên kết, các phiên được duy trì giữa các host. o TCP sử dụng các chỉ số phiên (session ID) để duy trì các liên kết giữa hai host. UDP không hỗ trợ bất kỳ phiên nào do bản chất phi liên kết của nó.
* Độ tin cậy. UDP không đảm bảo rằng chỉ có một bản sao dữ liệu tới đích. Để gửi dữ liệu tới các hệ thống cuối, UDP phân chia dữ liệu thành các đoạn nhỏ. UDP không đảm bảo rằng các đoạn này sẽ đến đích đúng thứ tự như chúng đã được tạo ra ở nguồn. Ngược lại, TCP sử dụng các số thứ tự cùng với số hiệu cổng và các gói tin xác thực thường xuyên, điều này đảm bảo rằng các gói tin đến đích đúng thứ tự mà nó đã được tạo ra. o Bảo mật. TCP có tính bảo mật cao hơn UDP. Trong nhiều tổ chức, firewall và router cấm các gói tin UDP, điều này là vì các hacker thường sử dụng các cổng UDP. o Kiểm soát luồng. UDP không có kiểm soát luồng; kết quả là, một ứng dụng UDP được thiết kế tồi có thể làm giảm băng thông của mạng.

### Các ưu điểm của UDP

* + - * Không cần thiết lập liên kết. UDP là giao thức phi liên kết, vì thế không cần phải thiết lập liên kết. Vì UDP không sử dụng các tín hiệu handshaking, nên có thể tránh được thời gian trễ. Đó chính là lý do tại sao DNS thường sử dụng giao thức UDP hơn là TCP-DNS sẽ chậm hơn rất nhiều khi dùng TCP. o Tốc độ. UDP

nhanh hơn so với TCP. Bởi vì điều này, nhiều ứng dụng thường được cài đặt trên giao thức UDP hơn so với giao thức TCP.

* + - * + Hỗ trợ hình trạng (Topology). UDP hỗ trợ các liên kết 1-1, 1-n, ngược lại TCP chỉ hỗ trợ liên kết 1-1. o Kích thước header. UDP chỉ có 8 byte header cho mỗi đoạn, ngược lại TCP cần các header 20 byte, vì vậy sử dụng băng thông ít hơn.

#### Sự khác nhau giữa UDP và TCP



### Khi nào thì nên sử dụng UDP

* + - * Rất nhiều ứng dụng trên Internet sử dụng UDP. Dựa trên các ưu và nhược điểm của UDP chúng ta có thể kết luận UDP có ích khi: o Sử dụng cho các phương thức truyền broadcasting và multicasting khi chúng ta muốn truyền tin với nhiều host. o Kích thước datagram nhỏ và trình tự đoạn là không quan trọng o Không cần thiết lập liên kết
        + Ứng dụng không gửi các dữ liệu quan trọng o

Không cần truyền lại các gói tin o Băng thông của mạng đóng vai trò quan trọng

### MÔ HÌNH CLIENT/SERVER

### Tổng quan

* + - * Trong nhiều năm gần đây, thuật ngữ client/server đã trở nên phổ biến trong lĩnh vực công nghệ thông tin. Các công ty xem client/server như là một mô hình công

nghệ mang tính cách mạng để trợ giúp cho việc quản lý và điều hành các hoạt động của công ty.

* + - * Nhiều năm trước, khi các máy tính lớn mainframe của IBM còn thống trị trong các lĩnh vực điện toán, một cách tiếp cận mới công nghệ mạng máy tính với tên gọi client/server đã xuất hiện vào đầu thập niên 80 và nhanh chóng phổ biến. Đến cuối thập niên 80, các mô hình client/server thật sự được chấp nhận và đã xuất hiện các ứng dụng đầu tiên. Ngày nay, client/server đã trở nên thông dụng và phổ biến đến mức hầu hết các ứng dụng mạng đều dựa trên nguyên tắc client/server. Mô hình này đã được chứng minh là phương pháp hiệu quả về chi phí khi xây dựng nhiều loại mạng và dịch vụ mạng, đặc biệt đối với mạng LAN ( PCS) chạy các ứng dụng end-user database.

 ***Khái niệm***

* + - * + Một mô hình kiến trúc tính toán (computational architecture) gồm các tiến trình khách (client) yêu cầu dịch vụ từ các tiến trình phục vụ( server )

#### Mô hình kiến trúc và ứng dụng

* + - * + Client/server là dạng phổ biến của mô hình ứng dụng phân tán trong đó phần mềm được chia thành các tiến trình client và các tiến trình server. Một client sẽ gửi yêu cầu cho một server theo một giao thức định s n (để server có thể hiểu được yêu cầu của client), để lấy thông tin hoặc chỉ thị server thực hiện một tác vụ nào đó. Sau khi thực hiện xong yêu cầu của client, server sẽ trả lời.
        + *Ví dụ :* Mối quan hệ tương tác qua lại tương tự như người khách hàng gửi một đơn đặt hàng theo một mẫu đến một nhà cung cấp, sau đó sẽ cung cấp hàng và hóa đơn. Mấu đơn hàng và hóa đơn là một phần của giao thức.Client/server thường được dịch qua tiếng Việt là khách/chủ. Tuy nhiên ý nghĩa của nó không hoàn toàn giống như vậy mà gần với quan hệ: khách hàng/người cung cấp dịch vụ. Tuy nhiên, đó không phải là mô tả chung cho hoạt động của client/server. Bản thân thuật ngữ client/server, cũng giống như các thuật ngữ tin học khác, không thể suy

trực tiếp từ nghĩa của client và server mà đã trở thành một khái niệm độc lập.

* + - * + Nhìn chung, trong client/server duy trì một sự phân biệt rõ rệt giữa các tiến trình và các thiết bị mạng. Thường thì máy tính client và máy server là hai thiết bị mạng riêng rẽ và có kiến trúc, cấu hình theo mục đích đã được thiết kế. Ví dụ, một máy chủ Web thường có cấu hình mạnh với năng lực xử lí cao (CPU) và bộ nhớ dung lượng lớn trong khi các Web client thường gồm các tính năng hỗ trợ giao diện đồ họa của trình duyệt như chất lượng hiển thị của card đồ họa, độ phân giải, kích thước màn hình Tuy nhiên, client/server tập trung chủ yếu vào các ứng dụng hơn là hệ thống phần cứng. Cùng một thiết bị có thể vừa là client vừa là server. Ví dụ : khi chạy trình duyệt trên máy chủ Web, máy chủ đó lại trở thành client. Tương tự, một thiết bị tại thời điểm này là server nhưng khi cần có thể đảo ngược vai trò để trở thành client đối với một server khác

#### Các ứng dụng phổ thông ví dụ trong môi trường internet trên client/server

* + - * + Email
        + FTP
        + Web

#### Đặc điểm của mô hình

* + - * + Trong kiến trúc Client/Server, client, chương trình đại diện cho người dùng đang cần các dịch vụ tài nguyên tính toán, và server, chương trình cung cấp các dịch vụ, là các đối tượng logic riêng rẽ tách biệt trao đổi, tương tác với nhau qua một một mạng máy tính truyền thông để cùng nhau thực hiện một tác vụ nào đó. Client tạo một yêu cầu dịch vụ và nhận được kết quả trả lời cho dịch vụ đó. Server sẽ nhận và xử lí các yêu cầu từ client, sau đó gửi trả lại. Mô hình có những đặc trưng sau: o Giao thức bất đối xứng: thể hiện ở quan hệ một-nhiều

giữa các client và một server. Client luôn bắt đầu phiên hội thoại bằng cách yêu cầu dịch vụ. Server luôn s n sàng chờ những yêu cầu từ client.

Đóng gói dịch vụ (Encapsulation of service): Server như một chuyên gia, luôn biết làm thế nào để hoàn thành tác vụ đáp ứng lại các yêu cầu từ client. Server có thể được nâng cấp mà không ảnh hưởng đến client (tất nhiên là thông điệp trao đổi giữa hai bên không đổi và cả hai cùng hiểu nhau)

Tính toàn vẹn : Mã và dữ liệu đối với một server được bảo trì tập trung để giảm chi phí bảo trì và bảo vệ sự toàn vẹn của các dữ liệu chung (được chia sẻ giữa các server). Trong khi đó, client duy trì tính chất cá nhân và độc lập.

Trong suốt định vị : Server là một tiến trình chạy trên cùng một máy với client hoặc trên một máy khác trong hệ thống mạng. Các phần mềm client/server thường ẩn vị trí của server đối với client bằng cách chuyển hướng các yêu cầu dịch vụ. Một chương trình có thể là server hoặc đồng thời cả hai

Trao đổi dựa trên thông điệp (message-based): client yêu cầu dịch vụ và server trả lời đều qua các message

* + - * + Tính modun, và khả năng điều chỉnh thích hợp: Một ứng dụng Client/Server được thiết kế gồm nhiều modun. Mục đích là để :

Chia để trị : một ứng dụng lớn được chia thành các modun nhỏ hơn làm cho việc thiết kế, phát triển và bảo trì trở nên đơn giản hơn o Khả năng chịu lỗi : lỗi xảy ra ở một modun nào đó không làm sập toàn bộ hệ thống.

Cơ chế chia tải (workload sharing) và dự phòng (redundancy) cũng làm tăng độ tin cậy.

Khả năng điều chỉnh : ứng dụng có thể đáp ứng tự động đối với sự tăng hay giảm của tải hệ thống bằng cách thêm vào hoặc tắt bớt các server và dịch vụ trong hệ thống.

* + - * + Tính không phụ thuộc nền tảng hệ thống: một ứng dụng client/server lí tưởng là phải có khả năng không phụ thuộc nền tảng phần cứng và hệ điều hành, cho phép sử dụng nhiều platform với client/server
        + Tính co giãn (mở rộng hoặc thu hẹp): Hệ thống client/server có thể co giãn theo chiều ngang và chiều dọc. Co giãn theo chiều ngang nghĩa là có thể thêm hay bớt các máy trạm client với ảnh hưởng vê hiệu năng tương đối nhỏ. Co giãn theo chiều dọc nghĩa là chuyển dịch vụ sang server mạnh hơn hoặc bổ sung thêm máy server.
        + Tách biệt chức năng giữa client và server: hai bên đảm nhận chức năng riêng
        + Chia sẻ tài nguyên : Một server có thể cung cấp dịch vụ cho nhiều client tại một thời điểm. Server cũng điều phối truy nhập của client đến các các tài nguyên dùng chung

#### Ưu điểm của mô hình

* + - * + Client/server được phát triển với mục đích cho phép nhiều người dùng cùng chia sẻ truy nhập đến các dịch vụ hay ứng dụng database. So sánh với mainframe, client/server đem lại khả năng mở rộng tốt hơn vì có thể tạo thêm các kết nối theo nhu cầu sử dụng mà không cần phải sử dụng thêm hard-wired. Mô hình client/server cũng hỗ trợ các ứng dụng được modun hóa. Trong từng trường hợp, một phần mềm ứng dụng được chia thành nhiều modun, mỗi modun được cài đặt trên các hệ thống phần cứng khác nhau được chuyên biệt hóa cho hệ thống modun đó, được gọi là mô hình client/server hai lớp hoặc ba lớp
        + Vì Server có thể tập trung hoặc phân tán trên nhiều máy, cho phép client và server có thể đặt tách biệt trên nhiều node trong mạng, mỗi node là một hệ thống máy tính độc lập, có cấu hình phần cứng, hệ điều hành , phần mềm khác nhau để phù hợp với tính năng và nhiệm vụ của nó.
        + Tóm lại, mục đích của client/server là để nâng cao tính dễ sử dụng

(usability), tính thích nghi linh hoạt (flexibility), tính trao đổi tương tác (interoperability) và tính co giãn (scalability) của một hệ thống tính toán

#### Nhược điểm của mô hình

* + - * + Một vấn đề được quan tâm đặc biệt trong mô hình client/server là quản trị hệ thống. Khi các ứng dụng được phân bố trên toàn hệ thống mạng, sẽ rất khó khăn để duy trì thông tin cấu hình luôn cập nhật và nhất quán giữa tất cả các thiết bị. Tương tự, nâng cấp một phiên bản mới của ứng dụng client/server rất khó đồng bộ. Cuối cùng, một hệ thống client/server phụ thuộc rất nhiều và độ tin cậy của mạng, nâng cao tính dự phòng và các tính năng chịu lỗi có thể làm tăng chi phí triển khai hệ thống.
        + Chi phí đầu tư cho việc thiết kế, cài đặt, quản trị và bảo trì là rất lớn. Ngoài ra, những kĩ sư hệ thống và người quản trị phải đối mặt với các vấn đề hóc búa như sự tương tác trong hệ thống, tính tương thích của các thành phần cũng như việc cấu hình hệ thống.

#### Kiến trúc 2 lớp client/server

* + - * + Trong ứng dụng client/server 2 lớp, khối nghiệp vụ được đặt bên trong lớp giao diện người dùng tại client hoặc được đặt bên trong lớp cơ sở dữ liệu dưới dạng các stored procedure. Khối nghiệp vụ cũng có thể được chia ra đặt tại cả client và server. File server và database server với stored procedure là ví dụ kiến trúc 2 lớp

#### Kiến trúc 3 lớp client/server

* + - * + Mô hình kiến trúc 3 lớp bắt đầu phát triển vào thập niên 90’s, khi nhu cầu đối với những hệ thống lớn hơn và ổn định hơn mà mô hình 2 lớp chưa đáp ứng được.
        + Mô hình kiến trúc 3 lớp vẫn dựa trên ý tưởng phân lớp các khối chức năng của hệ thống và được mở rộng từ mô hình 2 lớp.
        + Mô hình kiến trúc 3 lớp nâng cao hiệu năng (performance), tính linh hoạt

(flexibility), khả năng bảo trì (maintainability), khả năng dùng lại (reusability) và tính co giãn (scalability) trong khi ẩn đi sự phức tạp của quá trình xử lí phân tán đối với người dùng. Những đặc tính trên đã khiến cho kiến trúc 3

lớp trở thành sự lựa chọn cho nhiều ứng dụng Internet và các hệ thống thông tin mạng.

* + - * + *Kĩ thuật :*

Trong kiến trúc 3 lớp, hệ thống giao diện nằm tại lớp ngoài cùng, nơi các đặt các dịch vụ dành cho người dùng ( ví dụ như session, text, dialog, và display management).

Chức năng quản trị cơ sở dữ liệu gồm dịch vụ file và data đặt tại lớp trong cùng. Thành phần quản trị dữ liệu đảm bảo rằng sự nhất quán của dữ liệu trong môi trường phân tán thông qua các tính năng như : data lock, consistency và replication. Kết nối giữa các lớp có thể được thay đổi phụ thuộc vào yêu cầu của người dùng đối với dịch vụ và dữ liệu.

Lớp thứ 3 nằm giữa lớp giao diện và quản trị dữ liệu, vì vậy còn được gọi là lớp trung gian (middle tier). Tại lớp này đặt khối nghiệp vụ bao gồm các tiến trình xử lí nghiệp vụ được tách biệt khỏi dữ liệu và giao diện người dùng. Khối này có thể cung cấp dịch vụ quản lí các tiến trình nghiệp vụ cho nhiều ứng dụng khác nhau Theo cách đó, các tiến trình xử lí nghiệp vụ được triển khai và quản lý tách biệt với dữ liệu và giao diện người dùng. Như vậy, hệ thống 3 lớp có thể tích hợp dữ liệu từ nhiều nguồn khác nhau. Lớp middle tier giúp hệ thống nâng cao các khả năng (như đã nói trên) nhờ việc tập trung các tiến trình nghiệp vụ vào 1 lớp riêng, sử dụng server riêng, làm cho việc giám sát, thay đổi, quản trị được dễ dàng hơn. Những thay đổi, nâng cấp có thể thực hiện tại lớp middle tier chỉ một lần và có tác dụng đối với toàn bộ hệ thống. Đó là ưu điểm của kiến trúc 3 lớp so với kiến trúc 2 lớp.

Hơn thế nữa, lớp middle tier sẽ điều khiển các giao dịch và các hàng đợi yêu cầu (asynchronous queuing) để đảm bảo giao dịch được thực hiện trọn vẹn và chính xác. Truy nhập đến các tài nguyên theo tên (không theo vị trí) , các thành hệ thống được cài đặt thêm hoặc gỡ bỏ một cách đơn giản, không ảnh hưởng đến hoạt động của toàn bộ hệ thống.

* + - * + *Sử dụng:*

Kiến trúc 3 lớp được sử dụng trong môi trường client/server phân tán, trong các ứng dụng thương mại (commerce) và quân sự, với nhiều hệ cơ sở dữ liệu và các nghiệp vụ khác nhau

Kiến trúc này có thể hỗ trợ hàng trăm người dùng, quy mô lớn hơn nhiều so với kiến trúc 2 lớp

Kiến trúc giúp đơn giản công việc phát triển phần mềm vì mỗi lớp có thể được xây dựng và chạy trên các nền tảng hệ thống khác nhau. Hơn nữa, kiến trúc này cho phép các lớp khác nhau có thể phát triển trên các ngôn ngữ khác nhau. Ví dụ như : Client sử dụng ngôn ngữ HTML; lớp trung gian, làm nhiệm vụ sắp xếp nội dung hiển thị (layout) cho client và liên kết với lớp nghiệp vụ, sử dụng các ngôn ngữ script như Perl, PHP, ASP, JSP...;lớp middle tier được phát triển bằng C++, Java, SmallTalk, Delphi; ngôn ngữ SQL và một số ngôn ngữ khác cho từng DBMS riêng sử dụng cho lớp data.

Di chuyển một hệ thống ứng dụng cũ sang kiến trúc 3 lớp có thể thực hiện với ít rủi ro và chi phí thấp theo cách : duy trì cơ sở dữ liệu cũ, khối tiến trình xử lí nghiệp vụ để hệ thống cũ và mới cùng song hành hoạt động cho đến khi mỗi thành phần ứng dụng hoặc dữ liệu được chuyển sang thiết kế mới

* + - * + *Hạn chế*

Xây dựng một ứng dụng theo kiến trúc 3 lớp là một công việc phức tạp.Các công cụ thiết kế và lập trình không cung cấp tất cả các dịch vụ cần thiết để hỗ trợ môi trường tính toán phân tán.

Một vấn đề quan trọng khác trong thiết kế là việc phân tách 3 lớp không phải lúc nào cũng rõ ràng. Nó đòi hỏi phải có những phân tích tốt và khả

năng linh hoạt và cân đối trong thiết kế Hiệu năng : được cân đối giữa tải xử lí và tải đườngtruyền.

Ví dụ như xử lí toàn bộ trên trên một lớp chỉ trả về kết quả cuối cùng hay chia tải xử lí cho nhiều thành phần trên các lớp khác nhau. Rõ ràng khi đó thì tải đường truyền tăng.

#### Ví dụ ứng dụng các kiến trúc ứng dụng trong các môi trường khác nhau:

* + - * + Cửa hàng nhỏ hay ứng dụng cá nhân : cả ba khối chức năng nằm trên cùng một máy tính (ứng dụng desktop)
        + Công ty nhỏ hoặc một chi nhánh, một phòng ban: Cần một ứng dụng với một server dựa trên mạng LAN. Trong ứng dụng này, nhiều client sẽ kết nối với một server, việc quản trị là khá đơn giản, bảo mật được triển khai ở cấp độ từng hệ thống máy tính, các lỗi xảy ra dễ được phát hiện
        + Công ty lớn, tập đoàn : nhiều server cung cấp các dịch vụ và chức năng khác nhau, được đặt trong mạng LAN, mạng Intranet của công ty hay mạng Internet với khả năng co giãn. Các server có thể được phân vùng theo chức năng, tài nguyên, cơ sở dữ liệu và có thể lặp để tăng tính năng chịu lỗi và hiệu năng hệ thống. Mô hình này có tính linh hoạt cao và khả năng xử lí mạnhcũng như là các kinh nghiệm thực tế. Việc sắp đặt một chức năng nhất định vào một lớp nên dựa trên các tiêu chí sau :

Khả năng dễ dàng trong phát triển và test (usability) o Dễ dàng trong quản trị o Khả năng co giãn của các server (scalability): đặt chức năng không hợp lí có thể làm giảm tính co giãn của hệ thống.

### Giao thức cho ứng dụng Client/Server

* + - * Giao thức ở đây là bộ các khuôn dạng bản tin, tập trạng thái và các quy tắc, quy ước trong truyền thông giữa client và server.
      * Khi tạo ra một ứng dụng client/server, việc đầu tiên là phải thiết kế giao thức. May mắn cho chúng ta là các giao thức phổ biến như FTP, HTTP, SMTP, RPC... đều đã đươc IETF (nằm trong Internet Architecture Board-IAB) chuẩn hóa thành

các giao thức chuẩn. Các chuẩn này được được đưa trong khuyến nghị của IAB gọi là RFC (Request for Comment). Mỗi một chuẩn hoặc một giao thức cho Internet đều được đánh số trong RFC.

* + - * Khi lập trình database theo mô hình client/server, giao thức trao đổi các câu truy vấn và tập các bản ghi đã được quy định và hỗ trợ bởi hệ quản trị Database DBMS và các thư viện lập trình. Do đó, ta cũng không phải quan tâm thiết kế giao thức truyền thông trong ứng dụng database. Khi lập trình sẽ có s n các giao diện giúp ta thực thi việc tương tác giữa Client và Database server. Trong môi trường 3 lớp, thì lớp Middleware ở giữa cũng sẽ cung cấp giao diện để thực thi việc gọi hàm từ xa (Remote Procedure Call-RPC). Đây là một kĩ thuật cho phép các ứng dụng client server tương tác với nhau bằng phương pháp gọi hàm thông thường. Nó cung cấp giao diện gọi hàm mà che đi công tác truyền thông client server và việc truyền thông là trong suốt đối với người lập trình. Kĩ thuật này sẽ được trình bày trong phần sau.
      * Nói chung là ta nên theo các giao thức chuẩn có s n cho mỗi ứng dụng phổ biến. Tuy nhiên, khi xây dựng một ứng dụng riêng chưa có giao thức chuẩn thì ta phải thiết kế giao thức.

#### Phân loại các giao thức :

Có thể chia giao thức thành 2 dạng cơ bản :

* + - * Giao thức đồng bộ (Synchronous Protocol) o Trong dạng giao thức này, quá trình truyền thông giữa client và server diễn ra theo hai chiều nhưng không đồng thời mà được thực hiện lần lượt. Mỗi bên sau khi truyền dữ liệu hoặc thông báo cho bên kia sẽ ngừng lại để chờ bên kia gửi sang

o *Ví dụ* về các giao thức kiểu này là : HTTP, POP, SMTP...

* + - * Giao thức không đồng bộ (Asynchronous Protocol) o Đối với giao thức dạng không đồng bộ, client và server có thể đồng thời gửi thông tin cho nhau mà không cần phải chờ đợi phản hồi của bên kia

o Các giao thức này thường là các giao thức stateless như TELNET, RLOGIN...

* + - * Ngoài ra có thể có dạng giao thức hybrid là sự kết hợp của hai dạng giao thức đồng bộ và không đồng bộ

#### Các yêu cầu đối với giao thức

Việc thiết kế được một giao thức tốt là rất khó khăn và phải đáp ứng các yêu cầu cơ bản sau:

* + - * + Rõ ràng (well-defined) o Trong mỗi dạng bản tin được truyền đi, mọi trường dữ liệu phải được định nghĩa trong giao thức - rõ ràng về kích thước, ý nghĩa, các trường hợp liên quan đến từng bit.

Giao thức chính là ngôn ngữ, do đó đòi hỏi phải chính xác, không có sự nhập nhằng

Sử dụng các phương pháp diễn tả mang tính hình thức (common formal decscription) như dạng BNF (trong automata)

Nếu sử dụng các ngôn ngữ mô tả thì dạng ngôn ngữ đó cũng phải được trình bày trong tài liệu giao thức (để người đọc có thể hiểu được) o Nên sử dụng các ví dụ minh họa cho giao thức

* + - * + Đầy đủ (bao hết mọi khả năng có thể), là các vấn đề sau:

Dữ liệu hỏng, không hợp lệ (khuôn dạng, kích thước, tránh tràn bộ đệm) Các phiên bản cũ (tương thích). Ví dụ, một client phiên bản proto là 1.0 kết nối với server phiên bản 1.2

Các yêu cầu không hợp lệ. Lệnh sai hoặc tham số sai hoặc trình tự yêu cầu không đúng quy tắc, hoặc lệnh không được phép...

Các điều kiện giới hạn. Kiểm tra một số điều kiện nào đó trước khi tiến hành bước tiếp theo. Ví dụ, kiểm tra tính xác thực, kiểm tra điều kiện cho phép...

Có thể phân tích cú pháp trong các bản tin.

* + - * + Một số vấn đề khác khi thiết kế và triển khai giao thức o Khả năng mở rộng và tương thích giữa các version o Thứ tự byte o Giao thức là hướng kí tự hay hướng bi
* Tính hiệu quả : Đồng bộ và không đồng bộ Thông tin điều khiển, header

Thời gian trễ round trip o Trạng thái :Ai đọc, ai ghi (ai truyền, ai nhận)

* Timeout : Đây là vấn đề không thể thiếu đối với ứng dụng mạng do tính biến động và không biết trước của trạng thái mạng.

# CHƯƠNG III

**PHÂN TÍCH THIẾT KẾ CHƯƠNG TRÌNH**

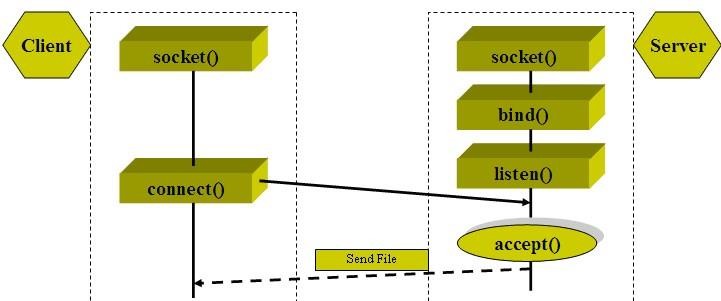
### MỤC ĐÍCH CỦA ĐỀ TÀI

Xây dựng chương trình trao đổi file giữa các máy trong mạng Lan với nhau

thông qua mô hình Cient/Server. Chức năng cơ bản của chương trình là Server gửi file cho Client.

### GIỚI THIỆU VỀ CHƯƠNG TRÌNH

* + - Dựa trên yêu cầu muốn trao đổi file giữa các máy trong mạng Lan với nhau thông qua mô hình Cient/Server. Dựa trên kiến thức đã học trong môn Kỹ thuật Phần mềm ứng dụng nhóm đã xây dựng chương trình Truyền File Mạng Lan được viết trên ngôn ngữ C#.
    - Chương trình có giao diện đơn giản, trực quan giúp người sử dụng có thể dễ dàng thao tác với các điều khiển.

Quy trình hoạt động của chương trình :

*Hình III - 1. Quá trình kết nối*

* ***socket()***: Client yêu cầu tạo một socket để có thể sử dụng các dịch vụ của tầng vận chuyển.
* ***connect()***: Client gởi yêu cầu nối kết đến Server có địa chỉ IP và port xác định.
* ***accept()***: Server chấp nhận nối kết của Client, kênh giao tiếp ảo được hình thành,Client và Server có thể trao đổi thông tin với nhau.
* Sau khi Server chấp nhận kết nối từ Client thì Server cũng sẽ truyền file cho Client. Client tiếp nhận và lưu file.

### GIỚI THIỆU THÀNH VIÊN VÀ PHÂN CÔNG NHIỆM VỤ:

### Giới thiệu thành viên:

Nhóm gồm 4 thành viên bao gồm:

|  |  |  |
| --- | --- | --- |
| 1) Đỗ Văn Nhu | INPG-15 | MSSV 20158292 |
| 2) Lê Văn Mạnh | INPG-15 | MSSV 20158248 |
| 3) Lê Minh Hiếu | INPG-15 | MSSV 20168680 |
| 4) Trần Quang Chiến | INPG-15 | MSSV 20168631 |

### Phân công nhiệm vụ các thành viên:

#### Kế hoạch đề ra :

* + - * + **Tuần 7**:
        + **Tuần 8**:

+ Tìm hiểu về mạng máy tính, giao thức FTP, Socket.

+ Lựa chọn ngôn ngữ lập trình, cụ thể là C#

+ Cài đặt Visual Studio và ôn lại các kỹ năng lập trình C#.

+Tập code ứng dụng vào bài tập. 

**Tuần 9,10,11,12,13**:

+ Thiết kế chương trình theo yêu cầu cho s n 

**Tuần 14**: Viết và nộp báo cáo.

#### Phân công thực hiện :

* + - * + **Tuần 7**:

*+ Cả nhóm* họp và thống nhất chọn ngôn ngữ lập trình C#

+ *Nhu* : Tìm hiểu giao thức trao đổi file FTP

+Hiếu*, Chiến,Mạnh*: Tìm hiểu Socket

+ *Cả nhóm*: Tìm hiểu tổng quan mạng máy tính

* + - * + **Tuần 8**:

+ Cả nhóm cùng cài đặt Visual C#, ôn lại kỹ năng lập trình. Tập code cho thuật toán.

* + - * + **Tuần 9,10,11,12,13:**

+ *Nhu*: Thiết kế giao diện

+ *Nhu*: Thiết kế lập trình giao diện, lập trình giao thức FTP

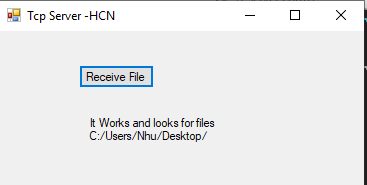
+ *Mạnh,Hiếu,Chiến:* lập trình Socket.

* + - * + **Tuần 14**: *Cả nhóm* cùng hoàn thiện báo cáo.

### GIAO DIỆN VÀ SỬ DỤNG CHƯƠNG TRÌNH:

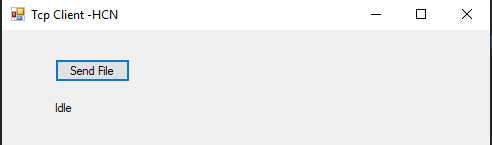
### Giao diện:

#### Chương trình server :



*Hình III - 2. Giao diện chƣơng trình Server*

#### Chương trình bên client

******

*Hình III - 3. Giao diện chương trình Client*

### Sử dụng chương trình:

 ***Phía server :*** Chương trình tại Server sẽ được kích hoạt trước.

* + - * *IP* : Chương trình sẽ lấy địa chỉ ip của Card mạng tại Server và gán vào phần địa chỉ IP
      * *Port* : Tại đây Server sẽ sử dụng 1 port để truyền file, số hiệu port này sẽ được gán từ người sử dụng tại Server.
      * Sau khi đã điền thông tin về địa chỉ và số hiệu port của Server thì kích vào điều khiển *Connect* để kích hoạt.
      * Tiếp theo chọn file cần truyền cho Client bằng cách kích vào điều khiển *Browse*

và chọn file đến file cần truyền. Tại ô File sẽ xuất hiện đường dẫn của file đã chọn.

* + - * Kích chọn điều khiển *Start* để lắng nghe, và khi một máy Client kết nối đến thì File sẽ được truyền đi. Trạng thái lắng nghe và báo hiệu khi máy Client đã nhận được File sẽ được biểu diễn ở *Status.*

 ***Phía Client:*** Sau khi chương trình Server đã được kích hoạt thì khi đó tại Client cần phải thực hiện các bước sau để nhận File.

* + - * *IP :* Tại địa chỉ ip người sử dụng Client sẽ điền địa chỉ của máy Server cần kết nối đến.
      * *Port :* Số hiệu port này sẽ dùng số hiệu port mà bên Server đã sử dụng để truyền.
      * Sau khi bên Server đã hoàn thành xong các bước để bắt đầu lắng nghe kết nối từ Client thì tại Client sẽ kết nối bằng điều khiển *Start.*
      * Khi máy Client đã kết nối thì File bên Server đã được truyền đến và tên File sẽ được hiển thị tại *Thông Báo Server gửi file.*
      * Tiếp theo Client muốn nhận được File thì kích chọn điều khiển *Save* để lưu File vào máy.

# KẾT LUẬN

*Qua đề tài nhóm chúng em có củng cố lại kiến thức đã được học. Và dưới sự hướng dẫn nhiệt tình của Thầy Nguyễn Đức Toàn đã giúp đỡ nhóm chúng em hoàn thiện chương trình “Truyền File Qua Mạng Lan” một cách hoàn chỉnh.*

*Chương trình của nhóm còn nhiều hạn chế mong Thầy và các bạn có những nhận xét đóng góp ý kiến để nhóm có thể hoàn thiện và phát triển chương hơn nhằm làm cho chương trình có thể được ứng dụng dễ dàng nhưng mang lại hiệu quả khi*

*truyền file . Thay mặt nhóm em xin chúc thầy và các bạn thành công và hạnh phúc trong cuộc sống .*

# TÀI LIỆU THAM KHẢO

## [1]. Giáo Trình Lập Trình Mạng [2]. Các trang Web

* [**http://google.com.vn**](http://google.com.vn/)
* **http://www.wikipedia**
* [**http://www.tailieu.vn**](http://www.tailieu.vn/)
* **http://www.codeproject**
* [**http://kilobook.com.vn**](http://kilobook.com.vn/)

**[3]. Và tham khảo thêm từ một số tài liệu và trang Web khác như**

1. [**https://www.linkedin.com**](https://www.linkedin.com/)
2. [**https://daynhauhoc.com**](https://daynhauhoc.com)
3. **https://www.howkteam.vn**