BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC DÂN LẬP HẢI PHÒNG

---------o0o---------

XÂY DỰNG HỆ THÔNG QUẢN LÝ

CẤU HÌNH MÁY TÍNH TRONG MẠNG LAN

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY

NGÀNH CÔNG NGHỆ THÔNG TIN

Sinh viên thực hiện: Hoàng Thị Ngoãn

Giáo viên hướng dẫn: Ths Lê Thụy

Mã số sinh viên: 111375

LỜI CẢM ƠN

Lời đầu tiên em xin được bày tỏ lòng biết ơn chân thành tới thầy giáo Ths Lê Thụy- giảng viên khoa CNTT trường ĐHDL Hải Phòng, người thầy đã trực tiếp giảng dạy và tận tình giúp đỡ em, chỉ bảo em trong suốt thời gian qua.Cảm ơn thầy đã luôn động viên, hướng dẫn, định hướng và truyền thụ cho em những kiến thức vô cùng quý báu để em có thể hoàn thành luận án tốt nghiệp này.

Em xin chân thành cảm ơn các thầy giáo, cô giáo trong trường ĐHDL Hải Phòng và đặc biệt là các thầy cô trong bộ môn tin học, những người đã không ngừng truyền đạt cho chúng em những kiến thức quý báu trong học tập cũng như trong cuộc sống suốt bốn năm học vừa qua.

Và cuối cùng, hơn hết em muốn được bày tỏ lòng biết ơn sâu sắc tới gia đình, bố mẹ, anh chị em cũng như tất cả bạn bè em, những người luôn ở bên động viên, cổ vũ và giúp đỡ em trong học tập cũng như trong cuộc sống.

Dưới đây là những gì em đã tìm hiểu và nghiên cứu được trong thời gian qua.Do tính thực tế và kiến thức còn hạn chế, vì vậy em rất mong nhận được sự chỉ bảo của các thầy cô giáo và sự tham gia đóng góp ý kiến của các bạn để em có thể hoàn thành tốt đề tài của mình.

Một lần nữa em xin chân thành cảm ơn!

Hải Phòng, ngày 06 tháng 07 năm 2011

Sinh viên

Hoàng Thị Ngoãn

MỤC LỤC

DANH MỤC HÌNH VẼ

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG BIỂU

LỜI MỞ ĐẦU

Trong những năm gần đây, mạng máy tính ngày càng trở nên phổ biến. Việc liên kết các máy tính trên môi trường mạng cũng như liên kết các mạng lại với nhau đem lại cho chúng ta nhiều lợi ích trong công việc cũng như trong học tập nghiên cứu, giải trí.Chúng ta có thể sử dụng các tài nguyên sẵn có được chia sẻ như file server, printer, máy fax,..môi trường mạng còn là một môi trường thông tin nhanh chóng và tiện lợi nhờ vào các cơ chế truyền thông trên mạng như: e-mail,www…

Cùng với sự phát triển của mạng máy tính là sự bùng nổ về số lượng máy tính được sử dụng trong các tổ chức, doanh nghiệp, trường học,… với cấu hình cao, các kỹ thuật hiện đại và việc trao đổi thông tin giữa các máy tính trong mạng trở lên dễ dàng hơn. Tuy nhiên, việc quản lý cũng gặp không ít khó khăn đặc biệt là quản lý về cấu hình của máy tính trong các phòng ban, trường học...

Em đã chọn đề tài "Xây dựng hệ thống quản lý cấu hình máy tính trong mạng LAN" làm đồ án tốt nghiệp. Mục tiêu chính của đồ án là giúp em hệ thống lại các kiến thức về mạng căn bản, tập trung vào nghiên cứu phương pháp lập trình socket và cơ chế vận hành của mô hình Client-Server nhằm giúp các máy tính trong mạng LAN có thể trao đổi dữ liệu với nhau.Sau là việc xây dựng hệ thống quản lý cấu hình máy tính mạng LAN giúp người quản lý nắm bắt được thông tin về cấu hình máy, sự thay đổi do khách quan hay chủ quan của cấu hình máy… và cụ thể là quản lý cấu hình máy tính trong phòng máy của trường ĐHDL Hải Phòng. Đồ án được trình bày theo 3 chương với bố cục như sau:

Chương 1: Mạng máy tính.

Chương 2: Phương pháp lập trình socket.

Chương 3: Phân tích và thiết kế hệ thống.

Việc nghiên cứu lý thuyết một cách hệ thống và xây dựng chương trình phần mềm đòi hỏi phải đầu tư rất nhiều thời gian.Với thời gian có hạn cho nên bài luận văn này của em không tránh khỏi những thiếu sót, em rất mong được sự chỉ dẫn thêm của thầy cô và các bạn.

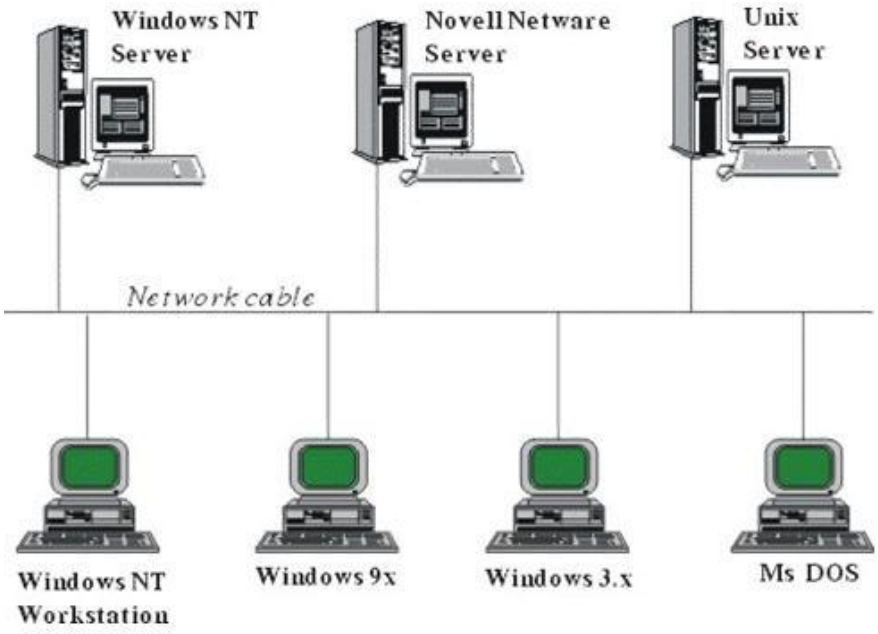
Em xin chân thành cảm ơn!

CHƯƠNG 1: MẠNG MÁY TÍNH

1.1 Định nghĩa mạng máy tính

Với sự phát triển của khoa học và kỹ thuật, hiện nay các mạng máy tính đã phát triển một cách nhanh chóng và đa dạng cả về quy mô, hệ điều hành và ứng dụng. Do vậy việc nghiên cứu chúng ngày càng trở nên phức tạp. Tuy nhiên các mạng máy tính cũng có cùng các điểm chung thông qua đó chúng ta có thể đánh giá và phân loại chúng.

Mạng máy tính là một tập hợp các máy tính được nối với nhau bởi đường truyền theo một cấu trúc nào đó và thông qua đó các máy tính trao đổi thông tin qua lại cho nhau.



Hình 1.1 Mô hình liên kết các máy tính trong mạng LAN

1.2 Nhu cầu phát triển máy tính

Ngày nay với một lượng lớn về thông tin, nhu cầu xử lý thông tin ngày càng cao. Mạng máy tính hiện nay trở nên quá quen thuộc đối với chúng ta, trong mọi lĩnh vực như khoa học, quân sự, quốc phòng, thương mại, dịch vụ, giáo dục... Hiện nay ở nhiều nơi mạng đã trở thành một nhu cầu không thể thiếu được. Người ta thấy được việc kết nối các máy tính thành mạng cho chúng ta những khả năng mới to lớn như:

Sử dụng chung tài nguyên: Những tài nguyên của mạng (như thiết bị, chương trình, dữ liệu) khi được trở thành các tài nguyên chung thì mọi thành viên của mạng đều có thể tiếp cận được mà không quan tâm tới những tài nguyên đó ở đâu.

Tăng độ tin cậy của hệ thống: Người ta có thể dễ dàng bảo trì máy móc và lưu trữ (backup) các dữ liệu chung và khi có trục trặc trong hệ thống thì chúng có

thể được khôi phục nhanh chóng. Trong trường hợp có trục trặc trên một trạm làm việc thì người ta cũng có thể sử dụng những trạm khác thay thế.

Nâng cao chất lượng và hiệu quả khai thác thông tin: Khi thông tin có thể được dùng chung thì nó mang lại cho người sử dụng khả năng tổ chức lại các công việc với những thay đổi về chất như:

- Đáp ứng những nhu cầu của hệ thống ứng dụng kinh doanh hiện đại.
- Cung cấp sự thống nhất giữa các dữ liệu.
- Tăng cường năng lực xử lý nhờ kết hợp các bộ phận phân tán.
- Tăng cường truy nhập tới các dịch vụ mạng khác nhau đang được cung cấp trên thế giới.

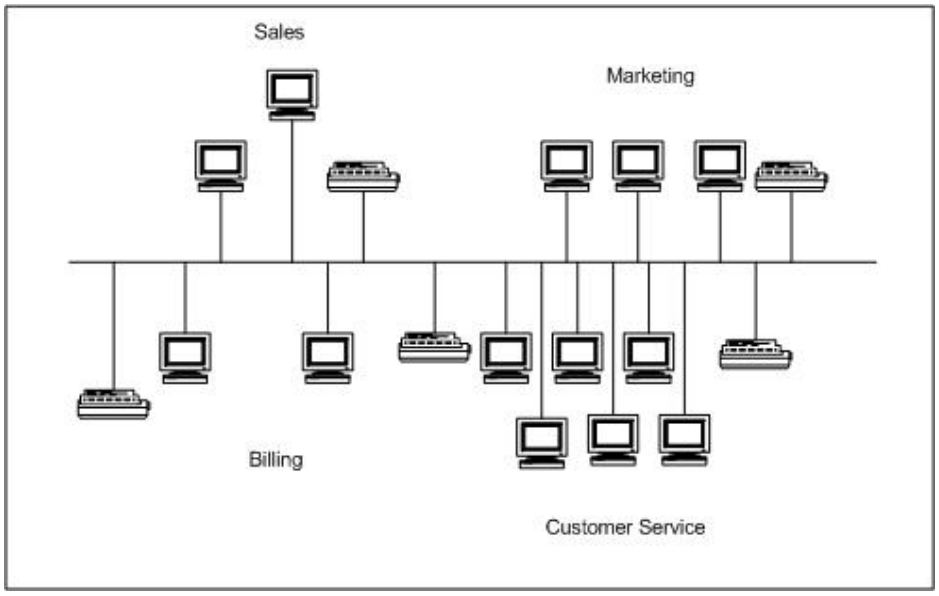
1.3 Phân loại mạng máy tính

1.3.1 Mạng cục bộ LAN (Local Area Network)

Mạng LAN là một nhóm các máy tính và các thiết bị truyền thông mạng được nối kết với nhau trong một khu vực nhỏ như một toà nhà cao ốc, khuôn viên trường đại học, khu giải trí…

Các mạng LAN thường có các đặc điểm sau đây:

- Băng thông lớn có khả năng chạy các ứng dụng trực tuyến như xem phim, hội thảo qua mạng.
- Kích thước mạng bị giới hạn bởi các thiết bị.
- Chi phí các thiết bị mạng LAN tương đối rẻ.
- Quản trị đơn giản.



Hình 1.2 Mô hình mạng cục bộ LAN

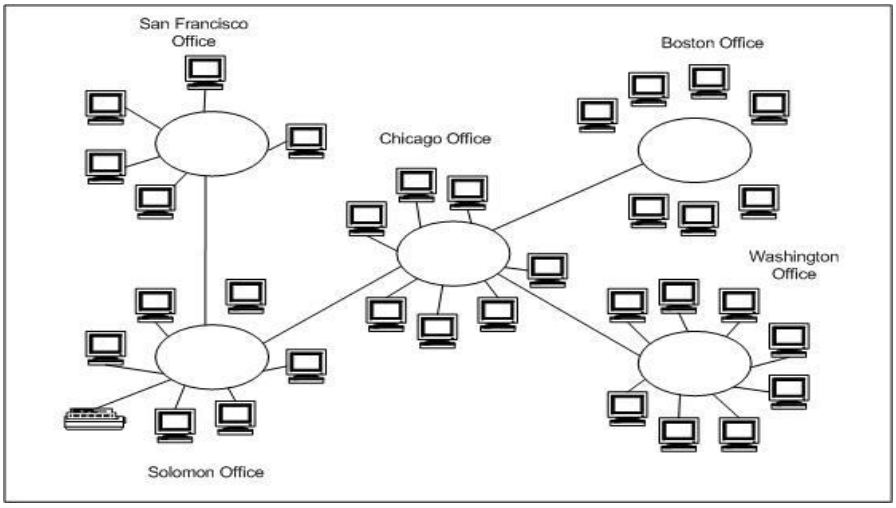
.

1.3.2 Mạng diện rộng WAN (Wide Area Network)

Mạng WAN bao phủ vùng địa lý rộng lớn có thể là một quốc gia, một lục địa hay toàn cầu. Mạng WAN thường là mạng của các công ty đa quốc gia hay toàn cầu điển hình là mạng Internet. Do phạm vi rộng lớn của mạng WAN nên thông thường mạng WAN là tập hợp các mạng LAN, MAN nối lại với nhau bằng các phương tiện như: vệ tinh (satellites), sóng viba (microwave), cáp quang, cáp điện thoại.

Đặc điểm của mạng WAN:

- Băng thông thấp, dễ mất kết nối thường chỉ phù hợp với các ứng dụng online như e-mail, web, ftp…
- Phạm vi hoạt động rộng lớn không giới hạn.
- Do kết nối của nhiều LAN, MAN lại với nhau nên mạng rất phức tạp và có tính toàn cầu nên thường là các tổ chức quốc tế đứng ra qui định và quản lý.
- Chi phí cho các thiết bị và các công nghệ mạng WAN rất đắt tiền.



Hình 1.3 Mô hình mạng diện rộng (WAN)

Mạng Internet:

Mạng Internet là trường hợp đặc biệt của mạng WAN, nó chứa các dịch vụ toàn cầu như Mail, Web, Chat, FTP và phục vụ miễn phí cho mọi người.

1.4 Phân loại mô hình xử lý mạng

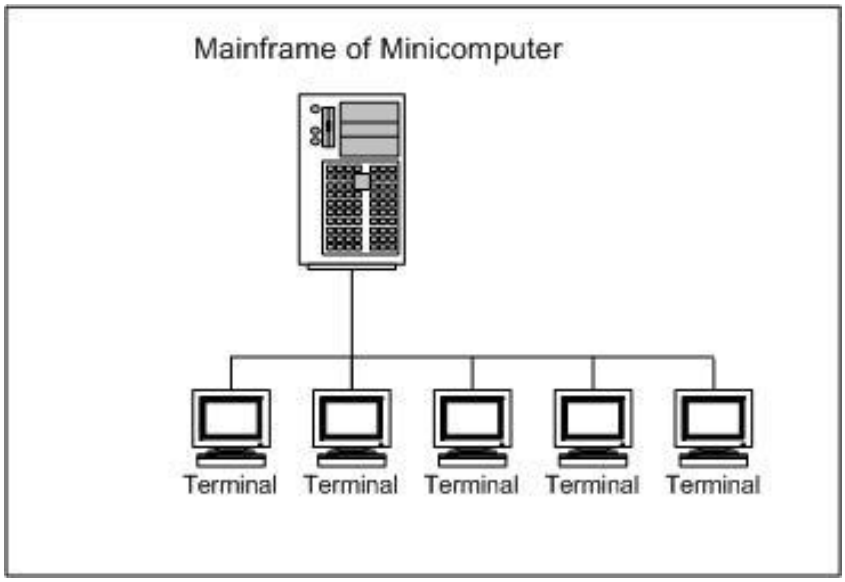
Cơ bản có 3 loại mô hình xử lý mạng bao gồm:

- Mô hình xử lý mạng tập trung.
- Mô hình xử lý mạng phân phối.
- Mô hình xử lý mạng cộng tác.

1.4.1 Mô hình xử lý mạng tập trung

Toàn bộ các tiến trình xử lý diễn ra tại máy tính trung tâm. Các máy trạm cuối (Terminals) được nối mạng với máy tính trung tâm và chỉ hoạt động như những thiết bị nhập xuất dữ liệu cho phép người dùng xem trên màn hình và nhập liệu bàn phím. Các máy trạm đầu cuối không lưu trữ và xử lý dữ liệu . Mô hình xử lý mạng trên có thể triển khai trên hệ thống phần cứng hoặc phần mềm được cài đặt trên Server.**Ưu điểm:** dữ liệu được bảo mật an toàn, dễ backup và diệt virus. Chi phí các thiết bị thấp.

Khuyết điểm: khó đáp ứng được các yêu cầu của nhiều ứng dụng khác nhau, tốc độ truy xuất chậm.



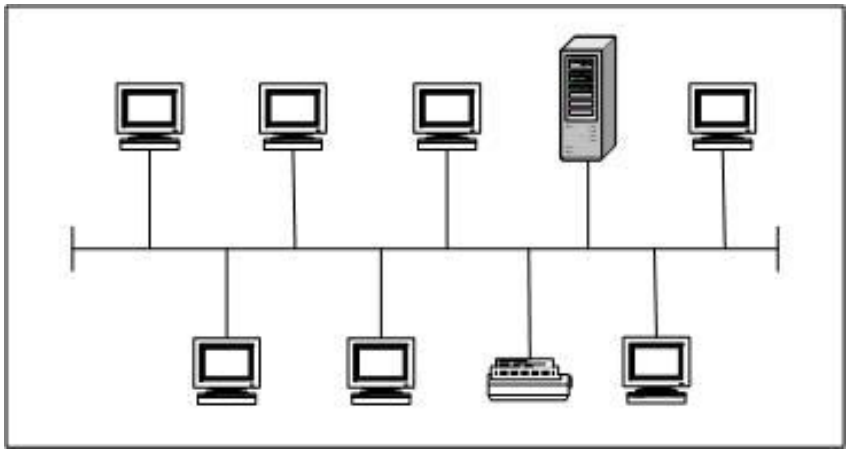
Hình 1.4 Mô hình xử lý mạng tập trung

1.4.2 Mô hình xử lý mạng phân phối

Các máy tính có khả năng hoạt động độc lập, các công việc được tách nhỏ và giao cho nhiều máy tính khác nhau thay vì tập trung xử lý trên máy trung tâm. Tuy dữ liệu được xử lý và lưu trữ tại máy cục bộ nhưng các máy tính này được nối mạng với nhau nên chúng có thể trao đổi dữ liệu và dịch vụ.

Ưu điểm: truy xuất nhanh, phần lớn không giới hạn các ứng dụng.

Khuyết điểm: dữ liệu lưu trữ rời rạc khó đồng bộ, backup và rất dễ nhiễm virus.



Hình 1.5 Mô hình xử lý mạng phân phối

1.4.3 Mô hình xử lý mạng cộng tác

Mô hình xử lý mạng cộng tác bao gồm nhiều máy tính có thể hợp tác để thực hiện một công việc. Một máy tính có thể mượn năng lực xử lý bằng cách chạy các chương trình trên các máy nằm trong mạng.

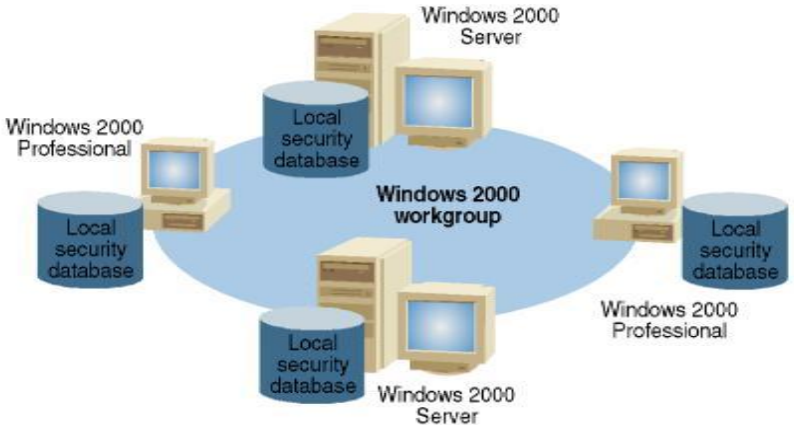
Ưu điểm: rất nhanh và mạnh, có thể dùng để chạy các ứng dụng có các phép toán lớn

Khuyết điểm: các dữ liệu được lưu trữ trên các vị trí khác nhau nên rất khó đồng bộ và backup, khả năng nhiễm virus rất cao.

1.5 Các mô hình quản lý mạng

1.5.1 Workgroup

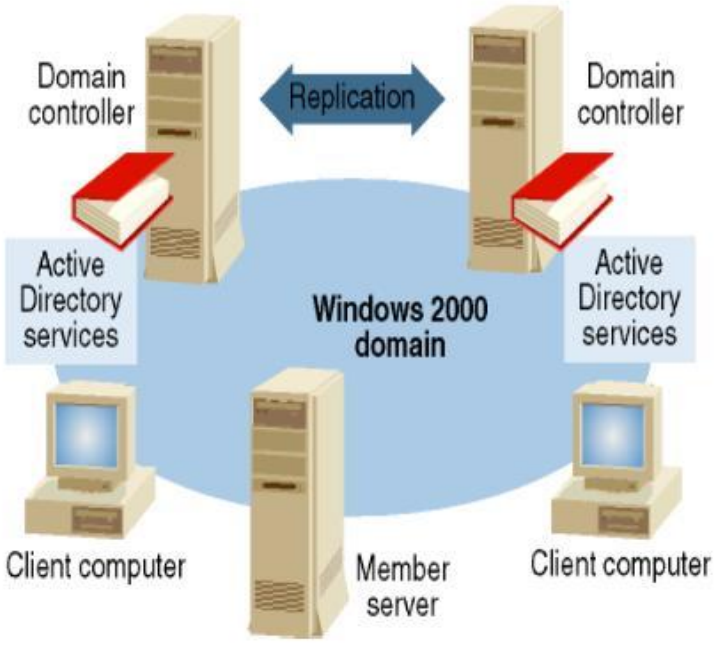
Trong mô hình này các máy tính có quyền hạng ngang nhau và không có các máy tính chuyên dụng làm nghiệp vụ cung cấp dịch vụ hay quản lý. Các máy tính tự bảo mật và quản lý tài nguyên của riêng mình. Đồng thời các máy tính cục bộ này cũng tự chứng thực cho người dùng cục bộ.



Hình 1.6 Mô hình quản lý mạng Workgroup

1.5.2 Domain

Ngược lại với mô hình Workgroup, mô hình Domain thì việc quản lý và chứng thực người dùng mạng tập trung tại máy tính Primary Domain Controller. Các tài nguyên mạng cũng được quản lý tập trung và cấp quyền hạn cho từng người dùng. Lúc đó trong hệ thống có các máy tính chuyên dụng làm nhiệm vụ cung cấp các dịch vụ và quản lý các máy trạm.



Hình 1.7 Mô hình quản lý mạng Workgroup

1.6 Các mô hình ứng dụng mạng

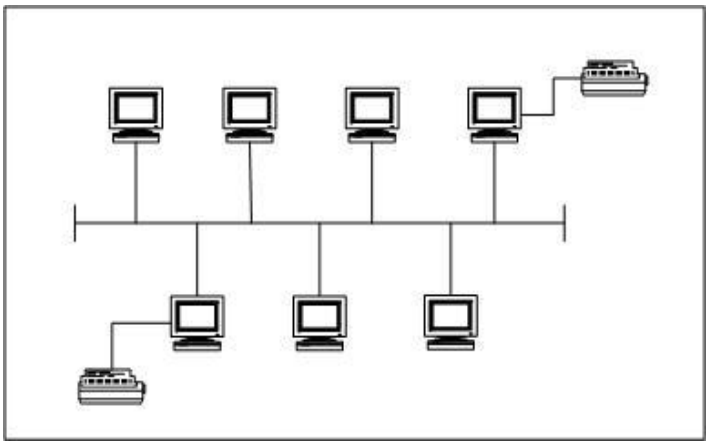
1.6.1 Mạng ngang hàng (Peer to peer)

Mạng ngang hàng cung cấp việc kết nối cơ bản giữa các máy tính nhưng không có bất kỳ một máy tính nào đóng vai trò phục vụ. Một máy tính trên mạng có thể vừa là Client vừa là Server. Trong môi trường này người dùng trên từng máy tính chịu trách nhiệm điều hành và chia sẻ tài nguyên của máy tính mình. Mô hình này chỉ phù hợp với tổ chức nhỏ, số người giới hạn (thông thường nhỏ hơn 10 người) và không quan tâm đến vấn đề bảo mật.

Mạng ngang hàng thường dùng các hệ điều hành sau: Win95, Windows for Workgroup, WinNT Workstation, Win2000 Proffessional, OS/2…

Ưu điểm: Do mô hình mạng ngang hàng đơn giản nên dễ cài đặt, tổ chức và quản trị, chi phí thiết bị cho mô hình này thấp.

Khuyết điểm: Không cho phép quản lý tập trung nên dữ liệu phân tán, khả năng bảo mật thấp rất dễ bị xâm nhập. Các tài nguyên không được sắp xếp nên rất khó định vị và tìm kiếm.



Hình 1.8 Mạng ngang hàng (peer to peer)

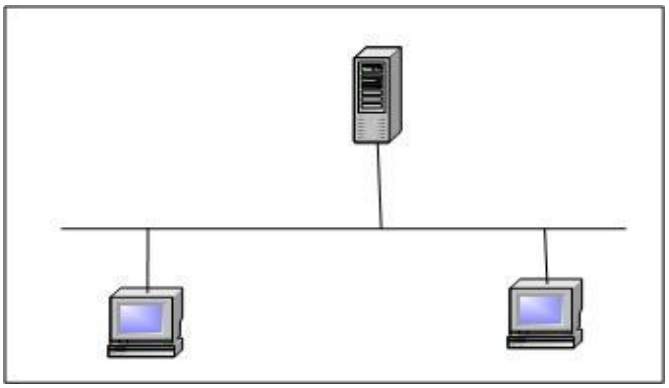
1.6.2 Mạng khách chủ (Client – Server)

Trong mô hình mạng khách chủ có một hệ thống máy tính cung cấp các tài nguyên và dịch vụ cho cả hệ thống mạng sử dụng gọi là các máy chủ (Server). Một hệ thống máy tính sử dụng các tài nguyên và dịch vụ này được gọi là máy khách (Client). Các Server thường có cấu hình mạnh (tốc độ xử lý nhanh, kích thước lưu trữ lớn) hoặc là các máy chuyên dụng.

Hệ điều hành mạng dùng trong mô hình Client - Server là WinNT, Novell Netware, Unix,Win2K…

Ưu điểm: Do các dữ liệu được lưu trữ tập trung nên dễ bảo mật, backup và đồng bộ với nhau. Tài nguyên và dịch vụ được tập trung nên dễ chia sẻ và quản lý và có thể phục vụ cho nhiều người dùng.

Khuyết điểm: Các Server chuyên dụng rất đắt tiền, phải có nhà quản trị cho hệ thống.



Hình 1.9 Mạng khách chủ (Client – Server)

1.7 Kiến trúc mạng cục bộ

1.7.1 Hình trạng mạng (Network Topology)

Topology mạng: Cách kết nối các máy tính với nhau về mặt hình học mà ta gọi là tô pô của mạng .

Có 2 kiểu nối mạng chủ yếu đó là:

- Nối kiểu điểm – điểm (point – to – point)
- Nối kiểu điểm – nhiều điểm (point – to – multipoint hay broadcast)

Point to Point: Các đường truyền nối từng cặp nút với nhau và mỗi nút đều có trách nhiệm lưu trữ tạm thời sao đó chuyển tiếp dữ liệu đi cho tới đích. Do cách làm việc như vậy nên mạng kiểu này còn được gọi là mạng "lưu và chuyển tiếp" (store and forward).

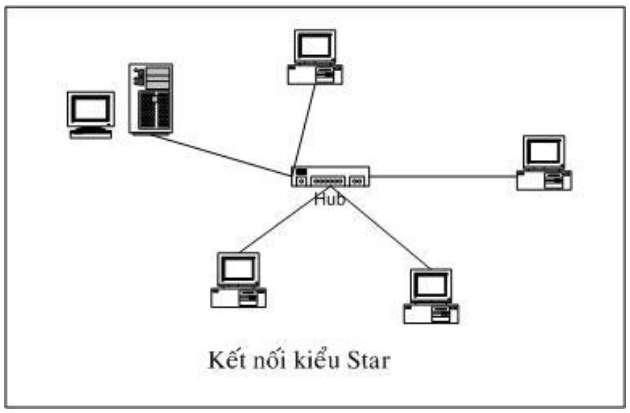
Point to multipoint: Tất cả các nút phân chia nhau một đường truyền vật lý chung. Dữ liệu gửi đi từ một nút nào đó sẽ được tiếp nhận bởi tất cả các nút còn lại trên mạng, bởi vậy chỉ cần chỉ ra địa chỉ đích của dữ liệu để căn cứ vào đó các nút tra xem dữ liệu đó có phải gửi cho mình không.

1.7.2 Mạng hình sao (Star)

Mạng hình sao có tất cả các trạm được kết nối với một thiết bị trung tâm có nhiệm vụ nhận tín hiệu từ các trạm và chuyển đến trạm đích. Tuỳ theo yêu cầu truyền thông trên mạng mà thiết bị trung tâm có thể là Switch, router, hub hay máy chủ trung tâm. Vai trò của thiết bị trung tâm là thiết lập các liên kết Point to Point.

Ưu điểm: Thiết lập mạng đơn giản, dễ dàng cấu hình lại mạng (thêm, bớt các trạm), dễ dàng kiểm soát và khắc phục sự cố, tận dụng được tối đa tốc độ truyền của đường truyền vật lý.

Khuyết điểm: Độ dài đường truyền nối một trạm với thiết bị trung tâm bị hạn chế (trong vòng 100m, với công nghệ hiện nay).



Hình 1.10 Mạng hình sao

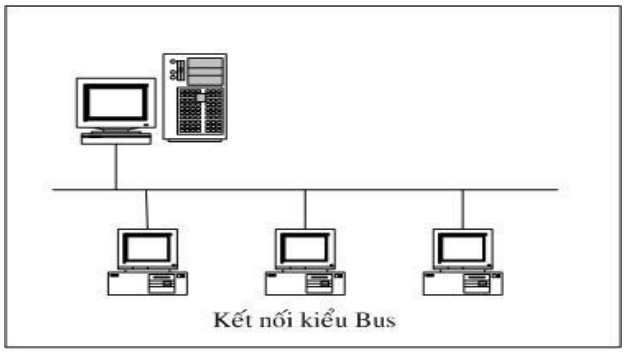
1.7.3 Mạng trục tuyến tính (Bus)

Tất cả các trạm phân chia một đường truyền chung (bus). Đường truyền chính được giới hạn hai đầu bằng hai đầu nối đặc biệt gọi là terminator. Mỗi trạm được nối với trục chính qua một đầu nối chữ T (T-connector) hoặc một thiết bị thu phát (transceiver).

Mô hình mạng Bus hoạt động theo các liên kết Point to Multipoint hay Broadcast.

Ưu điểm: Dễ thiết kế, chi phí thấp.

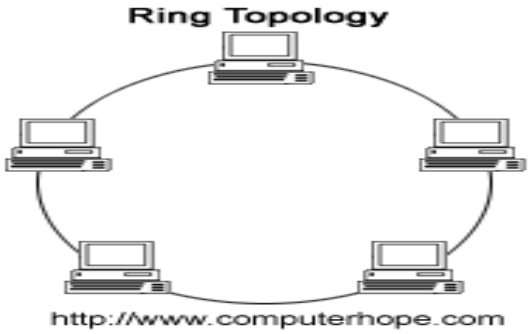
Khuyết điểm: Tính ổn định kém, chỉ một nút mạng hỏng là toàn bộ mạng bị ngừng hoạt động.



Hình 1.11 Mạng bus

1.7.4 Mạng hình vòng (Ring)

Trên mạng hình vòng tín hiệu được truyền đi trên vòng theo một chiều duy nhất. Mỗi trạm của mạng được nối với nhau qua một bộ chuyển tiếp (repeater) có nhiệm vụ nhận tín hiệu rồi chuyển tiếp đến trạm kế tiếp trên vòng. Như vậy tín hiệu được lưu chuyển trên vòng theo một chuỗi liên tiếp các liên kết Point to Point giữa các repeater.



Hình 1.12 Mạng hình vòng

Mạng hình vòng có ưu, nhược điểm tương tự như mạng hình sao, tuy nhiên mạng hình vòng đòi hỏi giao thức truy nhập mạng phức tạp hơn mạng hình sao.

Ngoài ra còn có các kết nối hỗn hợp giữa các kiến trúc mạng trên như: **Star Bus, Star Ring**

1.8 Giao thức mạng

1.8.1 Giao thức IP(Internet Protocol)

1.8.1.1 Tổng quát

Nhiệm vụ chính của giao thức IP là cung cấp khả năng kết nối các mạng con thành liên kết mạng để truyền dữ liệu, vai trò của IP là vai trò của giao thức tầng mạng trong mô hình OSI. Giao thức IP là một giao thức kiểu không liên kết

(connectionless) có nghĩa là không cần có giai đoạn thiết lập liên kết trước khi truyền dữ liệu.

Tổng quan về địa chỉ IP

- Là địa chỉ có cấu trúc, được chia làm hai hoặc ba phần là: Network_id & Host_id hoặc Network_id & Subnet_id & Host_id.
- Là một con số có kích thước 32 bit. Khi trình bày, người ta chia con số 32 bit này thành bốn phần, mỗi phần có kích thước 8 bit, gọi là octet hoặc byte.
- Có các cách trình bày sau:
  - Ký pháp thập phân có dấu chấm (dotted-decimal notation). Ví dụ: 172.16.30.56.
  - Ký pháp nhị phân. Ví dụ: 10101100 00010000 00011110 00111000.
  - Ký pháp thập lục phân. Ví dụ: AC 10 1E 38.
- Không gian địa chỉ IP (gồm 232 địa chỉ) được chia thành nhiều lớp (class) để dễ quản lý. Đó là các lớp: A, B, C, D và E; trong đó các lớp A, B và C được triển khai để đặt cho các host trên mạng Internet; lớp D dùng cho các nhóm multicast; còn lớp E phục vụ cho mục đích nghiên cứu.
- Địa chỉ IP còn được gọi là địa chỉ logical, trong khi địa chỉ MAC còn gọi là địa chỉ vật lý (hay địa chỉ physical).

Một số khái niệm và thuật ngữ liên quan

Địa chỉ host là địa chỉ IP có thể dùng để đặt cho các interface của các host. Hai host nằm cùng một mạng sẽ có network_id giống nhau và host_id khác nhau.

- **Địa chỉ mạng (network address):** là địa chỉ IP dùng để đặt cho các mạng. Phần host_id của địa chỉ chỉ chứa các bit 0. Địa chỉ này không thể dùng để đặt cho một Interface. Ví dụ 172.29.0.0

- **Địa chỉ Broadcast:** là địa chỉ IP được dùng để đại diện cho tất cả các host trong mạng. Phần host id chỉ chứa các bit 1. Địa chỉ này cũng không thể dùng để đặt cho một host được. Ví dụ 172.29.255.255

Bảng 1.1 Các phép toán làm việc trên bit

Phép AND			Phép OR		
A	B	A and B	A	B	A or B
1	1	1	1	1	1
1	0	0	1	0	1
0	1	0	0	1	1
0	0	0	0	0	0

Ví dụ sau minh hoạ phép AND giữa địa chỉ 172.29.14.10 và mask 255.255.0.0

172.29.14.10 = 10101100 00011101 00001110 00001010 AND

255.255.0.0 = 11111111 11111111 00000000 00000000

172.29.0.0 = 10101100 00011101 00000000 00000000

172.29.1.0

Mặt nạ mạng (Network Mask): là một con số dài 32 bits, là phương tiện giúp máy xác định được địa chỉ mạng của một địa chỉ IP (bằng cách AND giữa địa chỉ IP với mặt nạ mạng) để phục vụ cho công việc routing. Mặt nạ mạng cũng cho biết số bit nằm trong phần host_id. Được xây dựng bằng cách bật các bit tương ứng vớp phần network_ id và tắt các bit tương ứng với phần host_id.

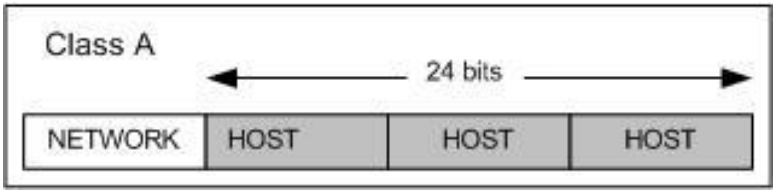
Bảng 1.2 Mặt nạ mặc định của các lớp không chia mạng con

Lớp A	255.0.0
Lớp B	255.255.0.0
Lớp C	255.255.255.0

Giới thiệu các lớp địa chỉ:

❖ **Lớp A**

Dành một byte cho phần network_id và ba byte cho phần host_id.



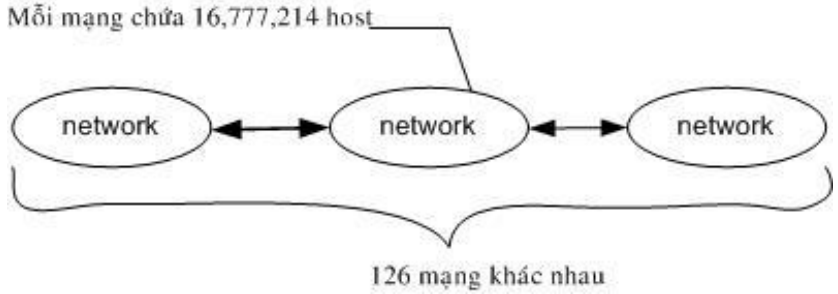
Hình 1.13 Địa chỉ lớp A

Để nhận biết lớp A, bit đầu tiên của byte đầu tiên phải là bit 0. Dưới dạnh nhị phân, byte này có dạng 0XXXXXXX . Vì vậy, những địa chỉ IP có byte đầu tiên nằm trong khoảng từ 0 (00000000) đến 127 (01111111) sẽ thuộc lớp A. Ví dụ : 50.14.32.8

Byte đầu tiên này cũng chính là network_id, trừ đi bit đầu tiên làm ID nhận dạng lớp A, còn lại 7 bits để đánh thứ tự các mạng, ta được 128 (27) mạng lớp A

khác nhau. Bỏ đi hai trường hợp đặc biệt là 0 và 127. Kết quả là lớp A chỉ còn 126 địa chỉ mạng, 1.0.0.0 đến 126.0.0.0

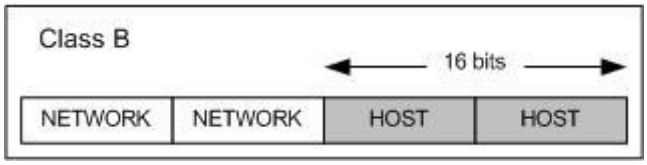
Phần host_id chiếm 24 bits, tức có thể đặt địa chỉ cho 16,777,216 host khác nhau trong mỗi mạng. Bỏ đi địa chỉ mạng (phần host_id chứa toàn các bit 0) và một địa chỉ Broadcast (phần host_id chứa toàn các bit 1) như vậy có tất cả 16,777,214 host khác nhau trong mỗi mạng lớp A. Ví dụ đối với mạng 10.0.0.0 thì những giá trị host hợp lệ là 10.0.0.1 đến 10.255.255.254



Hình 1.14 Số host trong một mạng lớp B

❖ **Lớp B**

Dành 2 bytes cho mỗi phần network_id và host_id.

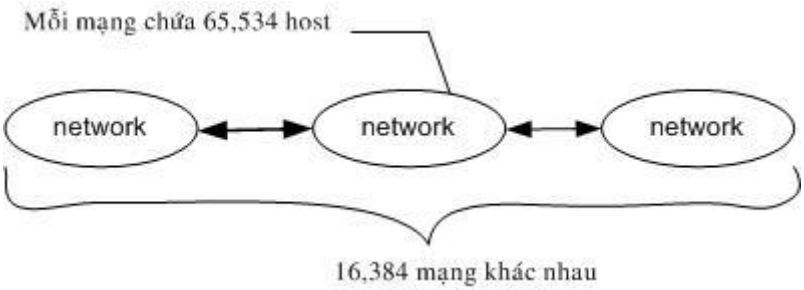


Hình 1.15 Địa chỉ lớp B

Dấu hiệu để nhận dạng địa chỉ lớp B là byte đầu tiên luôn bắt đầu bằng hai bit 10. Dưới dạng nhị phân, octet có dạng 10XXXXXX. Vì vậy những địa chỉ nằm trong khoảng từ 128 (10000000) đến 191 (10111111) sẽ thuộc về lớp B. Ví dụ 172.29.10.1 là một địa chỉ lớp B .

Phần network_id chiếm 16 bits bỏ đi 2 bits làm ID cho lớp, còn lại 14 bits cho phép ta đánh thứ tự 16,384 (214) mạng khác nhau (128.0.0.0 d8ến 191.255.0.0).

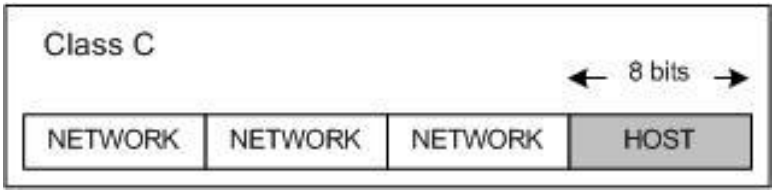
Phần host_id dài 16 bits hay có 65536 (216) giá trị khác nhau. Trừ đi 2 trường hợp đặc biệt còn lại 65534 host trong một mạng lớp B. Ví dụ đối với mạng 172.29.0.0 thì các địa chỉ host hợp lệ là từ 172.29.0.1 đến 172.29.255.254



Hình 1.16 Số host trong một mạng lớp B

❖ **Lớp C**

Dành 3 bytes cho phần network_id và một byte cho phần host_id

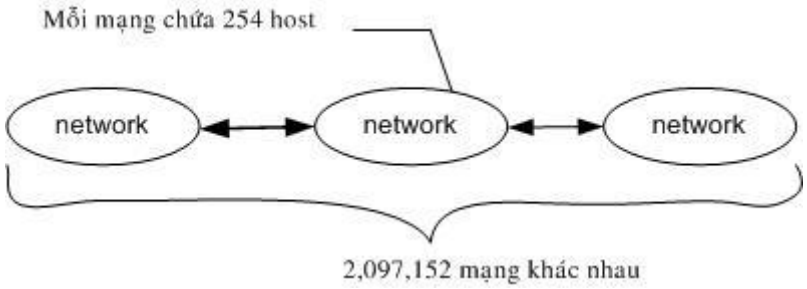


Hình 1.17 Địa chỉ lớp C

Byte đầu tiên luôn bắt đầu bằng 3 bits 110 và dạng nhị phân của octet này là 110XXXXX. Như vậy những địa chỉ nằm trong khoảng từ 192 (11000000) đến 223 (11011111) sẽ thuộc về lớp C. Ví dụ: 203.162.41.235

Phần network_id dùng 3 byte hay 24 bit, trừ đi 3 bit làm ID của lớp, còn lại 21 bit hay 2,097,152 (2 21) địa chỉ mạng (từ 192.0.0.0 đến 223.255.255.0).

Phần host_id dài 1 byte cho 256 (28) giá trị khác nhau. Trừ đi hai trường hợp đặc biệt ta còn 254 host khác nhau trong một mạng lớp C. Ví dụ, đối với mạng 203.162.41.0, các địa chỉ host hợp lệ là từ 203.162.41.1 đến 203.162.41.254



Hình 1.18 Số host trong một mạng lớp B

❖ **Lớp D và E**

Các địa chỉ có byte đầu tiên nằm trong khoảng 224 đến 256 là các địa chỉ thuộc lớp D hoặc E. Do các lớp này không phục vụ cho việc đánh địa chỉ các host nên không trình bày ở đây.

1.8.1.2 Các giao thức trong mạng IP

Để mạng với giao thức IP hoạt động được tốt người ta cần một số giao thức bổ sung, các giao thức này đều không phải là bộ phận của giao thức IP và giao thức IP sẽ dùng đến chúng khi cần.

❖ **Giao thức ARP (Address Resolution Protocol)**

Ở đây cần lưu ý rằng các địa chỉ IP được dùng để định danh các host và mạng ở tầng mạng của mô hình OSI, và chúng không phải là các địa chỉ vật lý (hay địa chỉ MAC) của các trạm trên đó một mạng cục bộ (Ethernet, Token Ring). Trên một mạng cục bộ hai trạm chỉ có thể liên lạc với nhau nếu chúng biết địa chỉ vật lý của nhau. Như vậy vấn đề đặt ra là phải tìm được ánh xạ giữa địa chỉ IP (32 bits) và

địa chỉ vật lý của một trạm. Giao thức ARP đã được xây dựng để tìm địa chỉ vật lý từ địa chỉ IP khi cần thiết.

❖ **Giao thức RARP (Reverse Address Resolution Protocol)**

Là giao thức ngược với giao thức ARP. Giao thức RARP được dùng để tìm địa chỉ IP từ địa chỉ vật lý.

❖ **Giao thức ICMP (Internet Control Message Protocol)**

Giao thức này thực hiện truyền các thông báo điều khiển (báo cáo về các tình trạng các lỗi trên mạng) giữa các gateway hoặc một nút của liên mạng. Tình trạng lỗi có thể là: một gói tin IP không thể tới đích của nó, hoặc một router không đủ bộ nhớ đệm để lưu và chuyển một gói tin IP, một thông báo ICMP được tạo và chuyển cho IP. IP sẽ "bọc" (encapsulate) thông báo đó với một IP header và truyền đến cho router hoặc trạm đích.

1.8.1.3 Các bước hoạt động của giao thức IP

Khi giao thức IP được khởi động nó trở thành một thực thể tồn tại trong máy tính và bắt đầu thực hiện những chức năng của mình, lúc đó thực thể IP là cấu thành của tầng mạng, nhận yêu cầu từ các tầng trên nó và gửi yêu cầu xuống các tầng dưới nó.

Đối với thực thể IP ở máy nguồn, khi nhận được một yêu cầu gửi từ tầng trên, nó thực hiện các bước sau đây:

1. Tạo một IP datagram dựa trên tham số nhận được.
2. Tính checksum và ghép vào header của gói tin.
3. Ra quyết định chọn đường: hoặc là trạm đích nằm trên cùng mạng hoặc một gateway sẽ được chọn cho chặng tiếp theo.
4. Chuyển gói tin xuống tầng dưới để truyền qua mạng.

Đối với router, khi nhận được một gói tin đi qua, nó thực hiện các động tác sau:

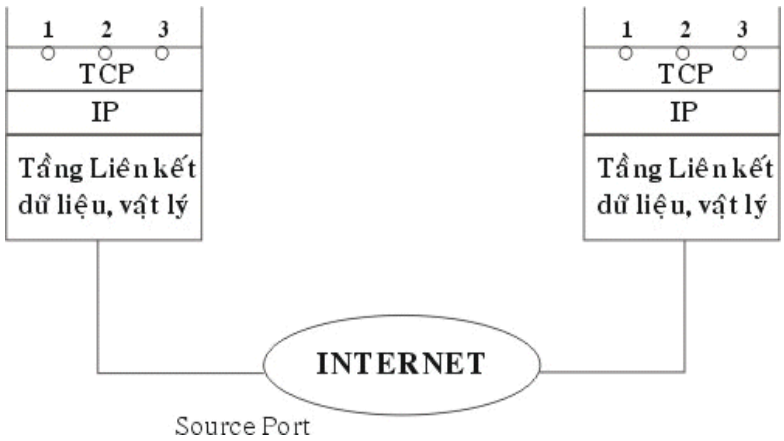
1. Tính chesksum, nếu sai thì loại bỏ gói tin.
2. Giảm giá trị tham số Time - to Live. Nếu thời gian đã hết thì loại bỏ gói tin.
3. Ra quyết định chọn đường.
4. Phân đoạn gói tin, nếu cần.
5. Kiến tạo lại IP header, bao gồm giá trị mới của các vùng Time - to -Live, Fragmentation và Checksum.
6. Chuyển datagram xuống tầng dưới để chuyển qua mạng.

Cuối cùng khi một datagram nhận bởi một thực thể IP ở trạm đích, nó sẽ thực hiện bởi các công việc sau:

1. Tính checksum. Nếu sai thì loại bỏ gói tin.
2. Tập hợp các đoạn của gói tin (nếu có phân đoạn).
3. Chuyển dữ liệu và các tham số điều khiển lên tầng trên.

1.8.2 Giao thức TCP (Transmission Control Protocol)

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 1.19 Cổng truy nhập dịch vụ 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.

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 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 1.3 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	Quotd (quote odd day)
20	ftp-data
21	ftp (control)
23	Telnet
25	SMTP
37	Time
53	Name Server
102	ISO - TSAP
103	X.400
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.

- 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).
- Khi TCP thiết lập được liên kết yêu cầu nó gửi tham số Open Sucsess được dùng để thông báo liên kết đã được thiết lập thành công. Thông báo này được 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.

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).

❖ **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 xử lý khẩn cấp dữ liệu đó.

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.

❖ **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.

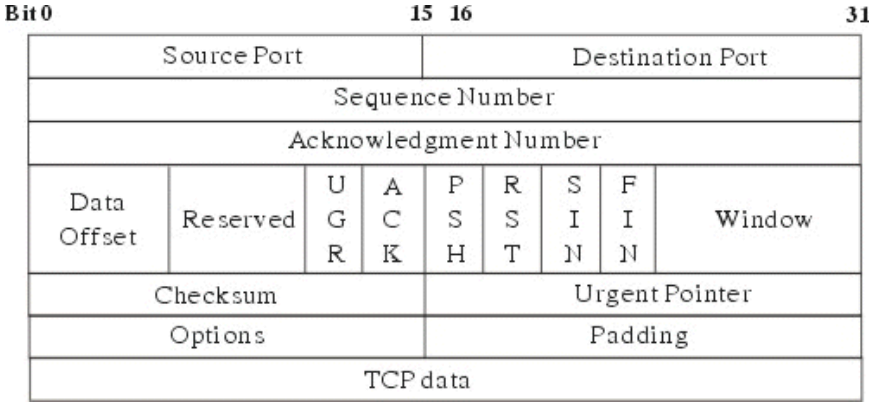
❖ **Hàm Abort:** Người sử dụng có thể đóng một liên kết bất kỳ và sẽ không chấp nhận dữ liệu qua liên kết đó nữa. Do vậy dữ liệu có thể bị mất đi khi đang được truyền đi. TCP báo cho TCP ở xa biết rằng liên kết đã được hủy bỏ và TCP ở xa sẽ thông báo cho người sử dụng của mình.

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.

Đơn vị dữ liệu sử dụng trong TCP được gọi là segment (đoạn dữ liệu), có các tham số với ý nghĩa như sau:



Hình 1.20 Dạng thức của segment TCP

- **Source Port** (16 bits): Số hiệu cổng TCP của trạm nguồn.
- **Destination Port** (16 bits): Số hiệu cổng TCP của trạm đích.
- **Sequence Number** (32 bits): số hiệu của byte đầu tiên của segment trừ khi bit SYN được thiết lập. Nếu bit SYN được thiết lập thì Sequence Number là số hiệu tuần tự khởi đầu (ISN) và byte dữ liệu đầu tiên là ISN+1.
- **Acknowledgment Number** (32 bits): số hiệu của segment tiếp theo mà trạm nguồn đang chờ để nhận. Ngầm ý báo nhận tốt (các) segment mà trạm đích đã gửi cho trạm nguồn.
- **Data offset** (4 bits): số lượng bội của 32 bit (32 bits words) trong TCP header (tham số này chỉ ra vị trí bắt đầu của nguồn dữ liệu).
- **Reserved** (6 bits): dành để dùng trong tương lai.
- **Control bit** (các bit điều khiển):

1. URG: Vùng con trỏ khẩn (Urgent Poiter) có hiệu lực.
2. ACK: Vùng báo nhận (ACK number) có hiệu lực.
3. PSH: Chức năng PUSH.
4. RST: Khởi động lại (reset) liên kết.
5. SYN: Đồng bộ hóa số hiệu tuần tự (sequence number).
6. FIN: Không còn dữ liệu từ trạm nguồn.

- **Window** (16 bits): cấp phát credit để kiểm soát nguồn dữ liệu (cơ chế cửa sổ). Đây chính là số lượng các byte dữ liệu, bắt đầu từ byte được chỉ ra trong vùng ACK number, mà trạm nguồn đã sẵn sàng để nhận.

- **Checksum** (16 bits): mã kiểm soát lỗi cho toàn bộ segment (header + data).

- **Urgemt Poiter** (16 bits): con trỏ này trỏ tới số hiệu tuần tự của byte đi theo sau dữ liệu khẩn. Vùng này chỉ có hiệu lực khi bit URG được thiết lập.

- **Options** (độ dài thay đổi): khai báo các option của TCP, trong đó có độ dài tối đa của vùng TCP data trong một segment.

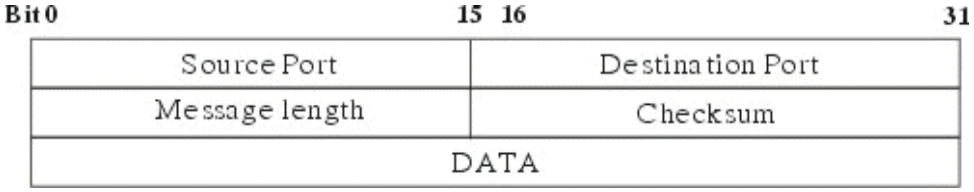
- **Paddinh** (độ dài thay đổi): phần chèn thêm vào header để đảm bảo phần header luôn kết thúc ở một mốc 32 bits. Phần thêm này gồm toàn số 0.

- **TCP data** (độ dài thay đổi): chứa dữ liệu của tầng trên, có độ dài tối đa ngầm định là 536 bytes. Giá trị này có thể điều chỉnh bằng cách khai báo trong vùng options.

1.8.3 Giao thức UDP (User Datagram Protocol)

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 1.21 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.

1.9 Các giao thức truy cập đường truyền trên mạng LAN

Để truyền được dữ liệu trên mạng người ta phải có các thủ tục nhằm hướng dẫn các máy tính của mạng làm thế nào và lúc nào có thể thâm nhập vào đường dây cáp để gửi các gói dữ kiện. Ví dụ như đối với các dạng bus và ring thì chỉ có một đường truyền duy nhất nối các trạm với nhau, cho nên cần phải có các quy tắc chung cho tất cả các trạm nối vào mạng để đảm bảo rằng đường truyền được truy nhập và sử dụng một cách hợp lý.

Có nhiều giao thức khác nhau để truy nhập đường truyền vật lý nhưng phân thành hai loại: các giao thức truy nhập ngẫu nhiên và các giao thức truy nhập có điều khiển.

1.9.1 Giao thức chuyển mạch (yêu cầu và chấp nhận)

Giao thức chuyển mạch là loại giao thức hoạt động theo cách thức sau: một máy tính của mạng khi cần có thể phát tín hiệu thâm nhập vào mạng, nếu vào lúc này đường cáp không bận thì mạch điều khiển sẽ cho trạm này thâm nhập vào đường cáp còn nếu đường cáp đang bận, nghĩa là đang có giao lưu giữa các trạm khác, thì việc thâm nhập sẽ bị từ chối.

1.9.2 Giao thức đường dây đa truy cập với cảm nhận va chạm

(Carrier Sense Multiple Access with Collision Detection hay CSMA/CD)

Giao thức đường dây đa truy cập cho phép nhiều trạm thâm nhập cùng một lúc vào mạng, giao thức này thường dùng trong sơ đồ mạng dạng đường thẳng. Mọi trạm đều có thể được truy nhập vào đường dây chung một cách ngẫu nhiên và do vậy có thể dẫn đến xung đột (hai hoặc nhiều trạm đồng thời cùng truyền dữ liệu). Các trạm phải kiểm tra đường truyền gói dữ liệu đi qua có phải của nó hay không. Khi một trạm muốn truyền dữ liệu nó phải kiểm tra đường truyền xem có rảnh hay không để gửi gói dữ liệu của, nếu đường truyền đang bận trạm phải chờ đợi chỉ được truyền khi thấy đường truyền rảnh. Nếu cùng một lúc có hai trạm cùng sử dụng đường truyền thì giao thức phải phát hiện điều này và các trạm phải ngưng thâm nhập, chờ đợi lần sau các thời gian ngẫu nhiên khác nhau.

Khi đường cáp đang bận trạm phải chờ đợi theo một trong ba phương thức sau:

- Trạm tạm chờ đợi một thời gian ngẫu nhiên nào đó rồi lại bắt đầu kiểm tra đường truyền.
- Trạm tiếp tục kiểm tra đường truyền đến khi đường truyền rảnh thì truyền dữ liệu đi.
- Trạm tiếp tục kiểm tra đường truyềnđến khi đường truyền rảnh thì truyền dữ liệu đi với xác suất p xác định trước ($0 < p < 1$).

Tại đây phương thức 1 có hiệu quả trong việc tránh xung đột vì hai trạm cần truyền khi thấy đường truyền bận sẽ cùng rút lui và chờ đợi trong các thời gian ngẫu nhiên khác nhau. Ngược lại phương thức 2 cố gắng giảm thời gian trống của đường truyền bằng các cho phép trạm có thể truyền ngay sau khi một cuộc truyền kết thúc song nếu lúc đó có thêm một trạm khác đang đợi thì khả năng xẩy ra xung đột là rất cao. Phương thức 3 với giá trị p phải lựa chọn hợp lý có thể tối thiểu hóa được khả năng xung đột lẫn thời gian trống của đường truyền.

Khi lưu lượng các gói dữ liệu cần di chuyển trên mạng quá cao, thì việc đụng độ có thể xẩy ra với số lượng lớn có gây tắc nghẽn đường truyền dẫn đến làm chậm tốc độ truyền tin của hệ thốnso s¸nh

1.9.3 Giao thức dùng thẻ bài vòng (Token ring)

Đây là giao thức truy nhập có điều khiển chủ yếu dùng kỹ thuật chuyển thẻ bài (token) để cấp phát quyền truy nhập đường truyền tức là quyền được truyền dữ liệu đi. Thẻ bài ở đay là một đơn vị dữ liệu đặc biệt, có kích thuóc và nội dung (gồm các thông tin điều khiển) được quy định riêng cho mỗi giao thức. Theo giao thức dùng thẻ bài vòng trong đường cáp liên tục có một thẻ bài chạy quanh trong mạng Thẻ bài là một đơn vị dữ liệu đặc biệt trong đó có một bit biểu diễn trạng thái sử dụng của nó (bận hoặc rỗi). Một trạm muốn truyền dữ liệu thì phải đợi đến khi nhận được một thẻ bài rảnh. Khi đó trạm sẽ đổi bit trạng thái của thẻ bài thành bận, nép gói dữ liệu có kèm theo địa chỉ nơi nhận vào thẻ bài và truyền đi theo chiều của vòng.

Vì thẻ bài chạy vòng quang trong mạng kín và chỉ có một thẻ nên việc đụng độ dữ liệu không thể xẩy ra, do vậy hiệu suất truyền dữ liệu của mạng không thay đổi.

Trong các giao thức này cần giải quyết hai vấn đề có thể dẫn đến phá vỡ hệ thống. Một là việc mất thẻ bài làm cho trên vòng không còn thẻ bài lưu chuyển nữa. Hai là một thẻ bài bận lưu chuyển không dừng trên vòng.

1.9.4 Giao thức dùng thẻ bài cho dạng đường thẳng (Token bus)

Đây là giao thức truy nhập có điều khiển trong để cấp phát quyền truy nhập đường truyền cho các trạm đang có nhu cầu truyền dữ liệu, một thẻ bài được lưu chuyển trên một vòng logic thiết lập bởi các trạm đó. Khi một trạm có thẻ bài thì nó có quyền sử dụng đường truyền trong một thời gian xác định trước. Khi đã hết dữ liệu hoặc hết thời đoạn cho phép, trạm chuyển thẻ bài đến trạm tiếp theo trong vòng logic.

Như vậy trong mạng phải thiết lập được vòng logic (hay còn gọi là vòng ảo) bao gồm các trạm đang hoạt động nối trong mạng được xác định vị trí theo một chuỗi thứ tự mà trạm cuối cùng của chuỗi sẽ tiếp liền sau bởi trạm đầu tiên. Mỗi trạm được biết địa chỉ của các trạm kề trước và sau nó trong đó thứ tự của các trạm trên vòng logic có thể độc lập với thứ tự vật lý. Cùng với việc thiết lập vòng thì giao thức phải luôn luôn theo dõi sự thay đổi theo trạng thái thực tế của mạng.

CHƯƠNG 2: PHƯƠNG PHÁP LẬP TRÌNH SOCKET

2.1 Socket

Socket là một giao diện lập trình ứng dụng(API –Application Programming Interface).Nó được giới thiệu lần đầu tiên trong ấn bản UNIX-BSD 4.2 dưới dạng các hàm hệ thống theo cú pháp ngôn ngữ C (socket(), bind(), connect(), send(), receive(), read(), write(), close(),…). 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, Visual C++,…Sau đây chúng ta sẽ đưa ra định nghĩa cụ thể về socket.

2.1.1 Định nghĩa

- Có nhiều định nghĩa khác nhau về socket tùy theo cách nhìn của người sử dụng.

- Một cách tổng quát nhất có thể định nghĩa : Một socket là một điểm cuối trong một kết nối giữa hai chương trình đang chạy trên mạng.

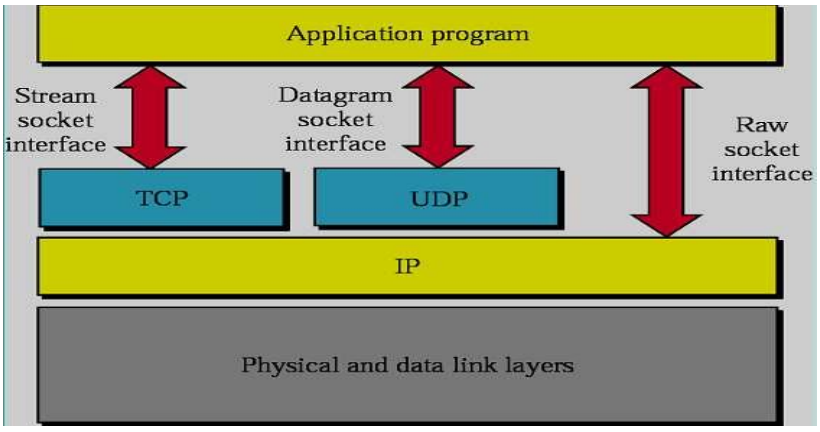
- Nhìn trên quan điểm của người phát triển ứng dụng người ta có thể định nghĩa Socket là một phương pháp để thiết lập kết nối truyền thông giữa một chương trình yêu cầu dịch vụ (được gán nhãn là Client) và một chương trình cung cấp dịch vụ (được gán nhãn là Server) trên mạng hoặc trên cùng một máy tính.

- Đối với người lập trình, họ nhìn nhận Socket như một giao diện nằm giữa tầng ứng dụng và tầng khác trong mô hình mạng OSI có nhiệm vụ thực hiện việc giao tiếp giữa chương trình ứng dụng với các tầng bên dưới mạng.

2.1.2 Phân loại

Có 3 loại Socket:

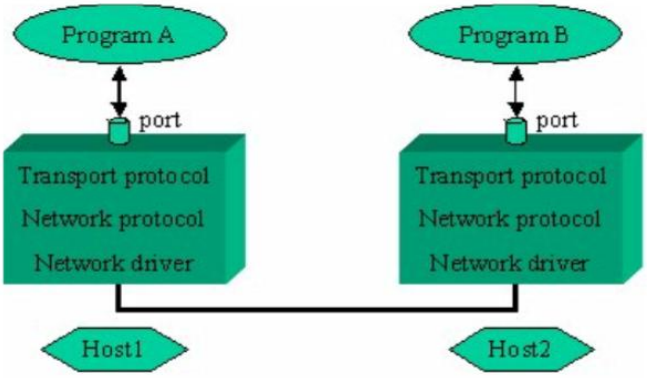
- Socket hướng kết nối (TCP Socket)
- Socket không hướng kết nối (UDP Socket)
- Raw Socket



Hình 2.1 Phân loại Socket

2.1.3 Chức năng

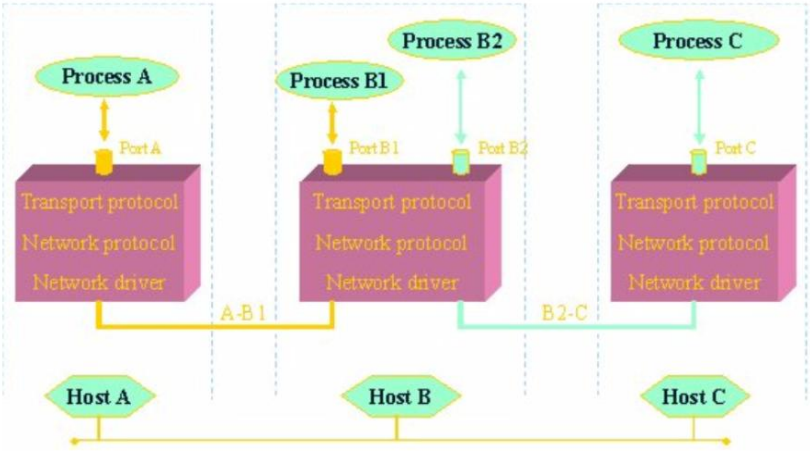
Socket cho phép thiết lập các kênh giao tiếp mà hai đầu kênh được xác định bởi hai cổng (port). Thông qua các cổng này một tiến trình có thể nhận và gửi dữ liệu với các tiến trình khác.



Hình 2.2 Mô hình Socket

Số hiệu cổng của Socket

- Để có thể thực hiện các cuộc giao tiếp, một trong hai quá trình phải công bố số hiệu cổng của socket mà mình sử dụng.
- Mỗi cổng giao tiếp thể hiện một địa chỉ xác định trong hệ thống. Khi quá trình được gán một số hiệu cổng, nó có thể nhận dữ liệu gởi đến cổng này từ các quá trình khác.
- Quá trình còn lại cũng yêu cầu tạo ra một socket.



Hình 2.3 Cổng trong Socket

Địa chỉ IP

Ngoài số hiệu cổng, hai bên giao tiếp còn phải biết địa chỉ IP của nhau. Địa chỉ IP giúp phân biệt máy tính này với máy tính kia trên mạng TCP/IP. Trong khi số hiệu cổng dùng để phân biệt các quá trình khác nhau trên cùng một máy tính.

Trong hình 2.2 địa chỉ của quá trình B1 được xác định bằng hai thông tin (Host B, Port B):

Địa chỉ máy tính có thể là địa chỉ IP dạng 203.162.88.162 hay là địa chỉ cho dạng trên miền như www.hpu.edu.vn

Số hiệu cổng gán cho Socket phải duy nhất trên phạm vi máy tính đó, có giá trị trong khoảng từ 0 đến 65535 (16 bit). Trong thực tế thì các số hiệu cổng từ 0 đến 1023 (gồm có cổng 1024) đã dành cho các dịch vụ nổi tiếng như http:80, telnet:21, ftp:23,… Nếu chúng ta không phải là người quản trị thì nên dùng từ cổng 1024 trở lên.

2.1.4 Nguyên lý hoạt động

Chúng ta có thể khái quát quá trình trao đổi dữ liệu thông qua các socket như sau:

- Chương trình phía Server tạo ra một socket, socket này được chương trình gắn với một cổng trên Server. Sau khi được tạo ra socket này(ta gọi socket phía server) sẽ chờ nghe yêu cầu từ phía client.
- Khi chương trình phía client cần kết nối với một server, nó cũng tạo ra một socket, socket này cũng được hệ điều hành gắn với một cổng. Chương trình client sẽ cung cấp cho socket của nó(ta gọi là socket phía client) địa chỉ mạng và cổng của socket phía server và yêu cầu thực hiện kết nối (nếu chương trình định sử dụng giao thức hướng kết nối) hoặc truyền dữ liệu(nếu chương trình sử dụng giao thức không hướng kết nối)
- Chương trình phía server và chương trình phía client trao đổi dữ liệu với nhau bằng cách đọc từ socket hoặc ghi vào socket của mình.Các socket ở hai phía nhận dữ liệu từ ứng dụng và đóng gói để gửi đi hoặc nhận các dữ liệu được gửi đến và chuyển cho chương trình ứng dụng bởi socket ở cả 2 phía đều được biết địa chỉ mạng và địa chỉ cổng của nhau.

Ở bước thứ 2 chúng ta thấy chương trình ứng dụng phải lựa chọn giao thức mà nó định sử dụng để trao đổi dữ liệu. Tùy theo việc chúng ta sử dụng giao thức nào (TCP hay UDP) mà cách thức xử lý yêu cầu trước yêu cầu của client có thể khác.

Sau đây chúng ta sẽ xem xét chi tiết cách thức trao đổi dữ liệu của socket với từng loại giao thức.

Socket hỗ trợ TCP

Ở phía Server : Khi một ứng dụng trên server hoạt động nó sẽ tạo ra một socket và đăng ký với server một cổng ứng dụng và chờ đợi yêu cầu kết nối từ phía client qua cổng này.

Ở phía Client: Nó biết địa chỉ của máy trên đó Server đang chạy vào cổng và Server đang chờ nghe yêu cầu. Do đó khi muốn kết nối đến Server, nó cũng tạo một socket chứa địa chỉ máy Client và cổng của ứng dụng trên máy Client đồng thời Client sẽ cung cấp cho socket của nó địa chỉ và cổng của Server mà nó cần kết nối và yêu cầu socket thực hiện kết nối.

Khi Server nhận được yêu cầu kết nối từ Client, nếu nó chấp nhận thì Server sẽ sinh ra một socket mới được gắn với một cổng khác với cổng mà nó đang nghe yêu cầu.Sở dĩ Server làm như vậy bởi nó cần cổng cũ để tiếp tục nghe yêu cầu từ phía Client trong khi vẫn cần một kết nối với Client.

Sau đó chương trình ứng dụng phía Server sẽ gửi thông báo chấp nhận kết nối cho Client cùng thông tin về địa chỉ cổng mới của socket mà nó dành cho Client.

Quay lại phía Client, nếu kết nối được chấp nhận nghĩa là socket của nó đã tạo ra thành công và nó có thể sử dụng socket để giao tiếp với Server bằng cách viết và ghi tới socket theo cách giao tiếp với một tài nguyên trên máy tính thông thường.

Socket hỗ trợ UDP

Ở phía Server: Khi một ứng dụng trên Server hoạt động nó sẽ tạo ra một socket và đăng ký với Server một cổng ứng dụng và chờ đợi yêu cầu kết nối từ phía Client qua cổng này.

Ở phía Client: Nó biết địa chỉ của máy trên đó Server đang chạy vào cổng và Server đang chờ nghe yêu cầu. Do đó khi muốn giao tiếp với Server, nó cũng tạo ra một socket chứa địa chỉ máy Client và cổng của ứng dụng trên máy Client đồng thời Client sẽ cung cấp cho socket của nó địa chỉ và cổng của Server mà nó cần kết nối. Khi Client muốn gửi tin đến Server nó sẽ chuyển dữ liệu cho socket của mình, socket này sẽ chuyển thẳng gói tin mà Client muốn gửi tới Server dưới dạng một datagram có chứa địa chỉ máy Server và cổng mà Server đang chờ nghe yêu cầu. Như vậy không hề có một kết nối nào được thực hiện giữa Client và Server, Server cũng không cần tạo ra một socket khác để kết nối với Client thay vào đó Server dùng ngay cổng ban đầu để trao đổi dữ liệu.

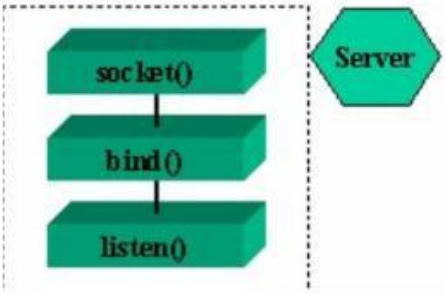
2.1.5 Cơ chế vận hành của mô hình Client-Server

Socket là phương tiện hiệu quả để xây dựng các ứng dụng theo kiến trúc Client-Server. Các ứng dụng trên mạng Internet như Web, mail, FTP là các ví dụ điển hình.

Phần này trình bày các bước cơ bản trong việc xây dựng các ứng dụng Client- Server sử dụng Socket làm phương tiện giao tiếp theo chế độ hướng kết nối.

Mô hình Client-Server sử dụng Socket ở chế độ có kết nối (TCP)

Giai đoạn 1: Server tạo Socket, gán số hiệu cổng và lắng nghe yêu cầu kết nối.



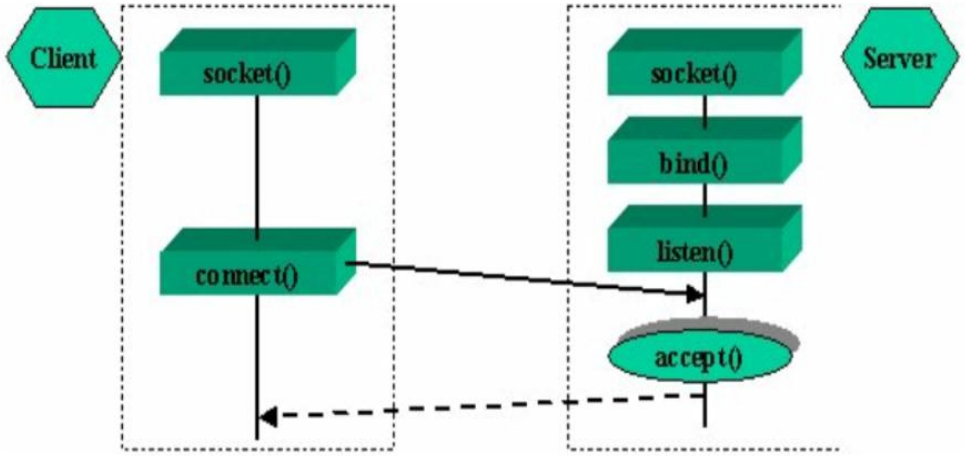
- Socket(): Server yêu cầu tạo một socket để có thể sử dụng các dịch vụ của

tầng vận chuyển.

- Bind(): Server yêu cầu gán số hiệu port cho socket.
- Listen(): Server lắng nghe các yêu cầu kết nối từ các client trên cổng đã được gán.

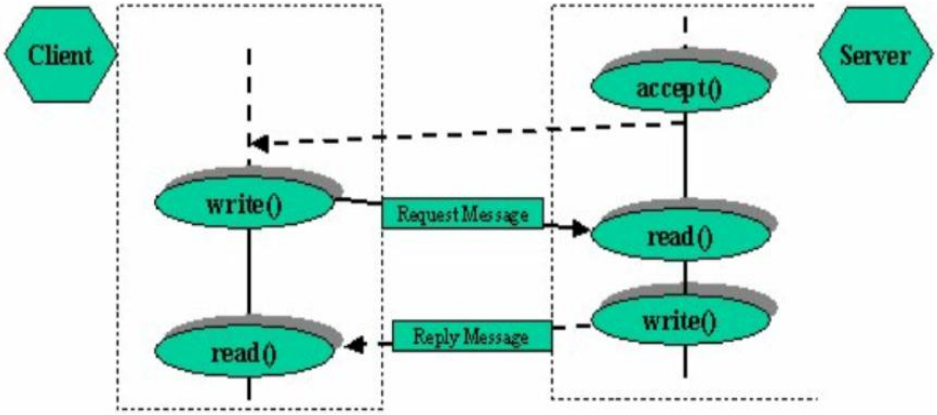
→Server sẵn sàng phục vụ Client.

Giai đoạn 2: Client tạo Socket, yêu cầu thiết lập một nối kết với Server.



- Socket(): Client yêu cầu tạo một socket để có thể sử dụng các dịch vụ của tầng vận chuyển, thông thường hệ thống tự động gán một số hiệu cổng còn trống cho socket của Client
- Connect(): Client gửi yêu cầu nối kết đến Server có địa chỉ IP và Port xác định.
- Accept(): Server chấp nhận nối kết của client, khi đó một kênh giao tiếp ảo được hình thành, Client và Server có thể trao đổi thông tin với nhau.

Giai đoạn 3: Trao đổi thông tin giữa Client và Server



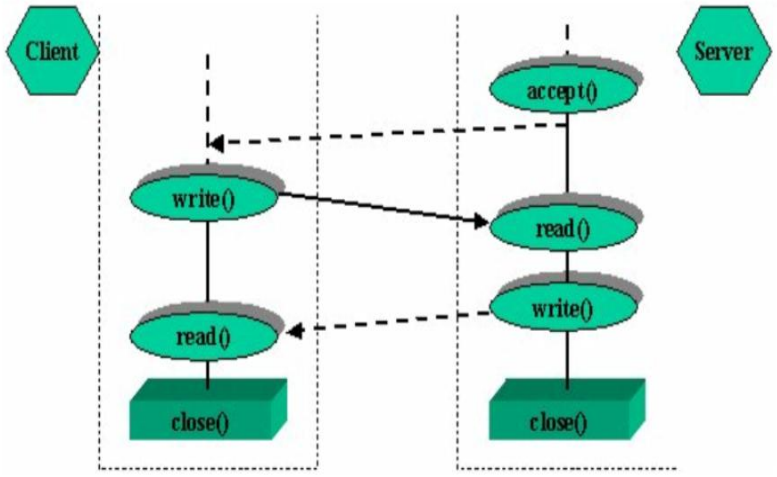
Sau khi chấp nhận yêu cầu nối kết, thông thường server thực hiện lệnh read() để đợi cho đến khi có thông điệp yêu cầu (Request Message) từ client gửi đến.

Server phân tích và thực thi yêu cầu.Kết quả sẽ được gửi về client bằng write().

Sau khi gửi yêu cầu bằng lệnh write(), client chờ nhận thông điệp kết quả (ReplyMessage) từ Server bằng lệnh read().

Trong giai đoạn này, việc trao đổi thông tin giữa Client và Server phải tuân thủ giao thức của ứng dụng (dạng thức và ý nghĩa của các thông điệp, quy tắc bắt tay, đồng bộ hóa,…) Thông thường Client sẽ là người gửi yêu cầu đến Server trước.

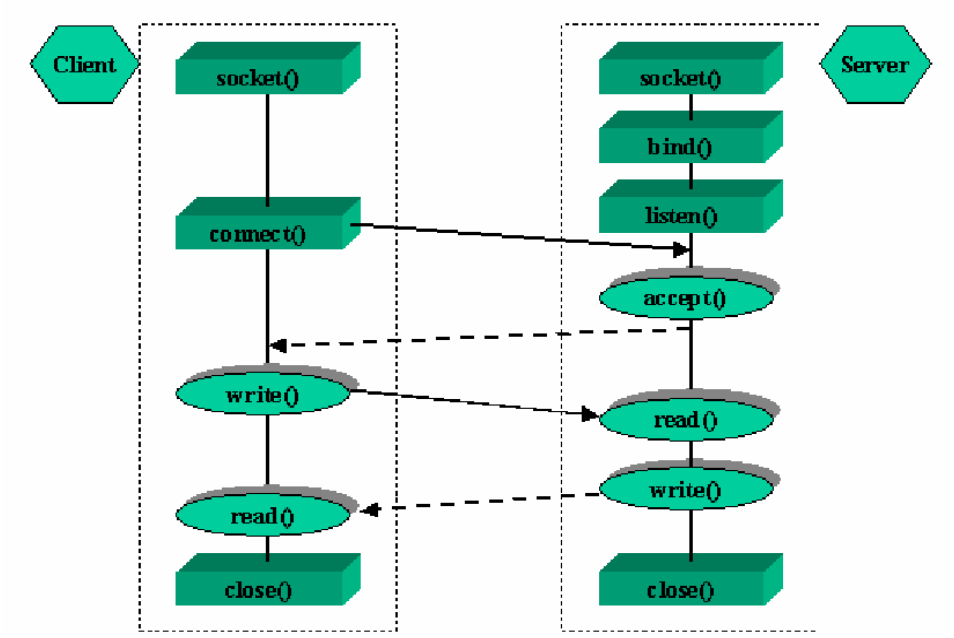
Giai đoạn 4: Kết thúc phiên làm việc



Các câu lệnh read(), write() có thể được thực hiện nhiều lần (kí hiệu bằng hình elipse).

Kênh ảo sẽ bị xóa khi Server hoặc Client đóng socket bằng lệnh close().

Như vậy toàn bộ tiến trình diễn ra như sau:



Hình 2.4 Mô hình Client-Server sử dụng socket ở chế độ có kết nối (TCP)

2.2 Lập trình Socket

2.2.1 Giới thiệu về NameSpace System.Net và System.Net.Sockets

- Cung cấp một giao diện lập trình đơn giản cho rất nhiều các giao thức mạng.

- Có rất nhiều lớp để lập trình: IPAdress, IPEndPoint, DNS, …

Lớp IPAdress

Một số Field cần chú ý:

- Any: Cung cấp một địa chỉ IP để chỉ ra rằng Server phải lắng nghe trên tất cả các Card mạng
- Broadcast: Cung cấp một địa chỉ IP quảng bá
- Loopback: Trả về một địa chỉ IP lặp
- AddressFamily: Trả về họ địa chỉ của IP hiện hành

Một số phương thức cần chú ý:

- Phương thức khởi tạo

  IPAddress(Byte[]).

  IPAddress(Int64).

- IsLoopback: Cho biết địa chỉ có phải địa chỉ lặp không
- Parse: Chuyển IP dạng xâu về IP chuẩn
- ToString: Trả địa chỉ IP về dạng xâu
- TryParse: Kiểm tra IP ở dạng xâu có hợp lệ không?

Lớp IPEndPoint

Một số phương thức cần chú ý:

- Phương thức khởi tạo
- Create: Tạo một EndPoint từ một địa chỉ Socket

- ToString : Trả về địa chỉ IP và số hiệu cổng theo khuôn dạng địa chỉ:cổng
Ví dụ: 192.168.1.1:8080

Lớp DNS

Một số thành phần của lớp:

- HostName: Cho biết tên của máy được phân giải
- GetHostAddress: Trả về tất cả IP của một trạm
- GetHostEntry: Giải đáp tên hoặc địa chỉ truyền vào và trả về đối tượng IPHostEntry.
- GetHostName: Lấy về tên của máy tính cục bộ

NameSpace System.Net.Sockets

Một số lớp hay dùng: TcpClient, UdpClient, TcpListener, Socket, NetworkStream, …

Để tạo ra Socket

Bảng 2.1 Socket (AddressFamily af, SocketType st, ProtocolType pt)

SocketType	Protocoltype	Description
Dgram	Udp	Connectionless communication
Stream	Tcp	Connection-oriented communication
Raw	Icmp	Internet Control Message
Raw	Raw	Plain IP packet communication

Ví dụ: Viết chương trình cho phía máy chủ

IPEndPoint ipep = new IPEndPoint(IPAddress.Any, 9050); Socket newsock = Socket(AddressFamily.InterNetwork,

SocketType.Stream, ProtocolType.Tcp);

newsock.Bind(ipep);

newsock.Listen(10);

Socket client = newsock.Accept();

//Gửi nhận dữ liệu theo giao thức đã thiết kế

……. newsock.Close();

2.2.2 Sử dụng các lớp hỗ trợ được xây dựng từ lớp Socket.

Mục đích của lớp UDPClient dùng cho lập trình với giao thức UDP, với giao thức này thì hai bên không cần phải thiết lập kết nối trước khi gửi do vậy mức độ tin cậy không cao. Để đảm bảo độ tin cậy trong các ứng dụng mạng, người ta còn dùng một giao thức khác, gọi là giao thức có kết nối : TCP (Transport Control Protocol). Trên Internet chủ yếu là dùng loại giao thức này, ví dụ như Telnet, HTTP, SMTP, POP3… để lập trình theo giao thức TCP, MS.NET cung cấp hai lớp có tên là TCPClient và TCPListener.

Lớp TCPClient

Các thành phần của lớp TcpClient

Bảng 2.2 Phương thức khởi tạo của lớp TcpClient

Constructor Method	
Name	Description
TcpClient()	Tạo một đối tượng TcpClient. Chưa đặt thông số gì.
TcpClient (IPEndPoint)	Tạo một TcpClient và gắn cho nó một EndPoint cục bộ. (Gán địa chỉ máy cục bộ và số hiệu cổng để sử dụng trao đổi thông tin về sau)
TcpClient (RemoteHost:String, Int32)	Tạo một đối tượng TcpClient và kết nối đến một máy có địa chỉ và số hiệu cổng được truyền

Bảng 2.3 Một số thuộc tính lớp TcpClient

Ký hiệu	Name	Description
	Available	Cho biết số byte đã nhận về từ mạng và có sẵn để đọc.
	Client	Trả về Socket ứng với TCPClient hiện hành.
	Connected	Trạng thái cho biết đã kết nối được đến Server hay chưa ?

Bảng 2.4 Một số phương thức khác lớp TcpClient

Ký hiệu	Name	Description
	Close	Giải phóng đối tượng TcpClient nhưng không đóng kết nối.
	Connect (RemoteHost, Port)	Kết nối đến một máy TCP khác có Tên và số hiệu cổng.
	GetStream	Trả về NetworkStream để từ đó giúp ta gửi hay nhận dữ liệu(Thường làm tham số khi tạo StreamReader và StreamWrite) Khi đã gắn vào StreamReader và StreamWriter rồi thì ta có thể gửi và nhận dữ liệu thông qua các phương thức Readln,writeline tương ứng của các lớp này.

Lớp TCPListener

TCPListerner là một lớp cho phép người lập trình có thể xây dựng các ứng dụng Server (Ví dụ như SMTP Server, FTP Server, DNS Server, POP3 Server hay server tự định nghĩa ….). Ứng dụng server khác với ứng dụng Client ở chỗ nó luôn luôn thực hiện lắng nghe và chấp nhận các kết nối đến từ Client.

Các thành phần của lớp TcpListener:

Bảng 2.5 Phương thức khởi tạo của lớp TcpListener

Constructor method	
Name	Description
TcpListener (Port: Int32)	Tạo một TcpListener và lắng nghe tại cổng chỉ định.
TcpListener	Tạo một TcpListener với giá trị Endpoint truyền vào.
TcpListener (IPAddress,Int32)	Tạo một TcpListener và lắng nghe các kết nối đến tại địa chỉ IP và cổng chỉ định.

Bảng 2.6 Các phương thức khác của lớp TcpListener

Ký hiệu	Name	Description
	AcceptSocket	Chấp nhận một yêu cầu kết nối đang chờ.
	AcceptTcpClient	Chấp nhận một yêu cầu kết nối đang chờ. (Ứng dụng sẽ dừng tại lệnh này cho đến khi nào có một kết nối đến)
	Pending	Cho biết liệu có kết nối nào đang chờ đợi không ? (True= có).
	Start	Bắt đầu lắng nghe các yêu cầu kết nối.
	Stop	Dừng việc nghe.

Lớp UDPClient

Giao thức UDP(User Datagram Protocol hay User Define Protocol) là một giao thức phi kết nối(Connectionless) có nghĩa là một bên có thể gửi dữ liệu cho bên kia mà không cần biết là bên đó đã sẵn sàng hay chưa? (Nói cách khác là không cần thiết lập kết nối giữa hai bên khi tiến hành trao đổi thông tin).Giao thức này không tin cậy bằng giao thức ICP nhưng tốc độ lại nhanh và dễ cài đặt. Ngoài ra, với giao thức UDP ta còn có thể gửi các gói tin quảng bá (broadcast) cho đồng thời nhiều máy.

Trong .NET, lớp UDPClient (nằm trong System.Net.Socket) đóng gói các chức năng của giao thức UDP.

Bảng 2.7 Phương thức khởi tạo của lớp UdpClient

UdpClient()	Tạo một đối tượng (thể hiện) mới của lớp UdpClient.
UdpClient (AddressFamily)	Tạo một đối tượng (thể hiện) mới của lớp UdpClient.Thuộc một dòng địa chỉ (AddressFamily) được chỉ định.
UdbClient(Int32)	Tạo một UdpClient và gắn (bind) một cổng cho nó.
UdpClient(IPEndPoint)	Tạo một UdpClient và gắn (bind) một IPEndPoint(gán địa chỉ IP và cổng)cho nó.
UdpClient(Int32,AddressFamily)	Tạo một UdpClient và gán số hiệu cổng, AddressFamily
UdpClient(String,Int32)	Tạo một UdpClient và thiết lập với một trạm từ xa mặc định.

Bảng 2.8 Các phương thức khác của lớp UdpClient

Methosd Public		
Ký hiệu	**Name**	**Description**
	BeginReceive	Nhận dữ liệu không đồng bộ từ máy ở xa.
	BeginSend	Gửi không đồng bộ tới máy ở xa
	Close	Đóng kết nối
	Connect	Thiết lập một Default remote host
	EndReceive	Kết thúc nhận dữ liệu không đồng bộ ở trên.
	Receive	Nhận dữ liệu(đồng bộ) do máy ở xa gửi. (Đồng bộ có nghĩa là các lệnh ngay sau lệnh Receive chỉ được thực thi nếu Receive đã nhận được dữ liệu về.Còn nếu nó chưa nhận được dù chỉ một chút thì nó vẫn cứ chờ (blocking)).
	Send	Gửi dữ liệu đồng bộ cho máy ở xa.

2.2.3 Sử dụng Thread trong các ứng dụng mạng

Một số khái niệm

- **Đa nhiệm (Multitasking):** Là khả năng hệ điêu hành làm nhiều công việc tại một thời điểm

- **Tiến trình (Process):** Khi chạy một ứng dụng, hệ điều hành sẽ cấp phát riêng cho ứng dụng đó bộ nhớ và các tài nguyên khác. Bộ nhớ và tài nguyên vật lý riêng biệt này được gọi là một tiến trình. Các tài nguyên và bộ nhớ của một tiến trình thì chỉ tiến trình đó được phép truy cập.

- **Tuyến (Thread):** Trong hệ thống, một tiến trình có thể có một hoặc nhiều chuỗi thực hiện tách biệt nhau và có thể chạy đồng thời. Mỗi chuỗi thực hiện này được gọi là một tuyến (Thread). Trong một ứng dụng, Thread khởi tạo đầu tiên gọi là Thread sơ cấp hay Thread chính.

Sử dụng Thread trong chương trình .Net

Để sử dụng Thread trong .Net ta sử dụng NameSpace System.Threading

Bảng 2.9 Một số phương thức thường dùng trong Thread

Public Method	Mô tả
Abort()	Kết thúc Thread
Join()	Buộc chương trình phải chờ cho thread kết thúc (Block) thì mới thực hiện tiếp (các câu lệnh đứng sau Join).
Resume()	Tiếp tục chạy thread đã bị tạm ngưng - suspended.
Sleep()	Static method : Tạm dừng thread trong một khoảng thời gian
Start()	Bắt đầu chạy (khởi động) một thread. Sau khi gọi phương thức này, trạng thái của thread chuyển từ trạng thái hiện hành sang Running.

Bảng 2.10 Một số thuộc tính thường dùng trong Thread

Public Property	Mô tả
CurrentThread	This static property: Trả về thread hiện hành đang chạy
IsAlive	Trả về giá trị cho biết trạng thái thực thi của thread hiện hành.
IsBackground	Sets or gets giá trị cho biết là thread là background hay foreground thread.
IsThreadPoolThread	Gets a value indicating whether a thread is part of a thread pool.
Priority	Sets or gets giá trị để chỉ định độ ưu tiên (dành nhiều hay ít CPU cho thread). Cao nhất là 4, thấp nhất là 0.
ThreadState	Lấy về trạng thái của Thread (đang dừng hay đang chạy)…

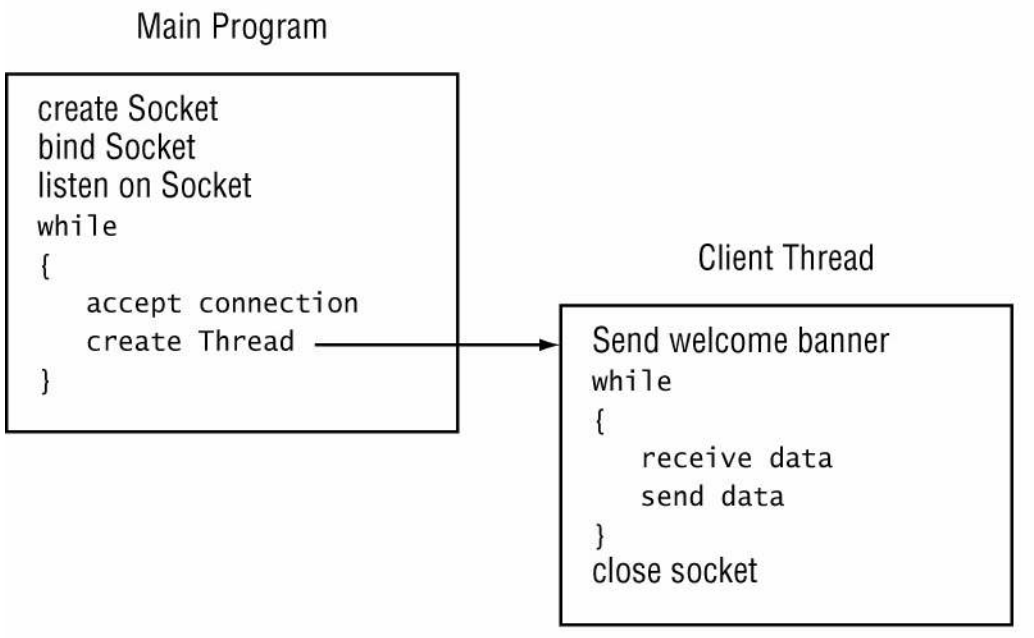
Tạo một tuyến trong C#

```
……
Thread newThread=newThread(new ThreadStart(newMethod));
…….
}
void newMethod() {
. . .
}
```

Sử dụng Thread trong các chương trình Server

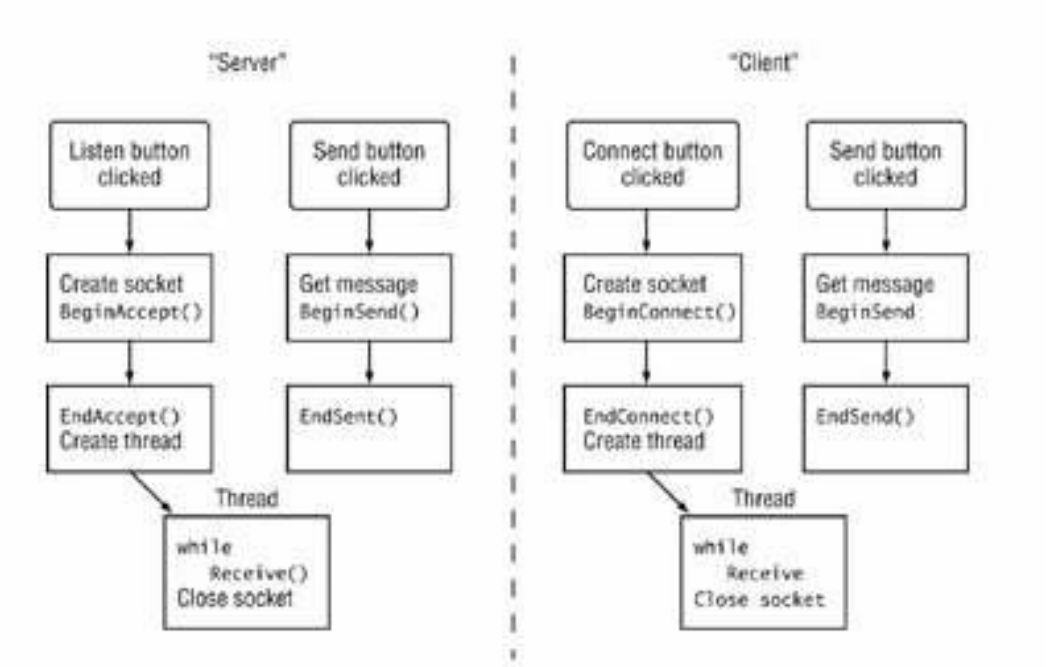
- Đa tuyến hay được ứng dụng trong các chương trình Server, các chương trình đòi hỏi tại một thời điểm chấp nhận nhiều kết nối đến từ các Client.

- Để các chương trình Server có thể xử lý nhiều Client tại một thời điểm ta có mô hình ứng dụng đa tuyến như sau:



Hình 2.5 Mô hình ứng dụng đa tuyến

Sử dụng Thread để gửi/nhận dữ liệu



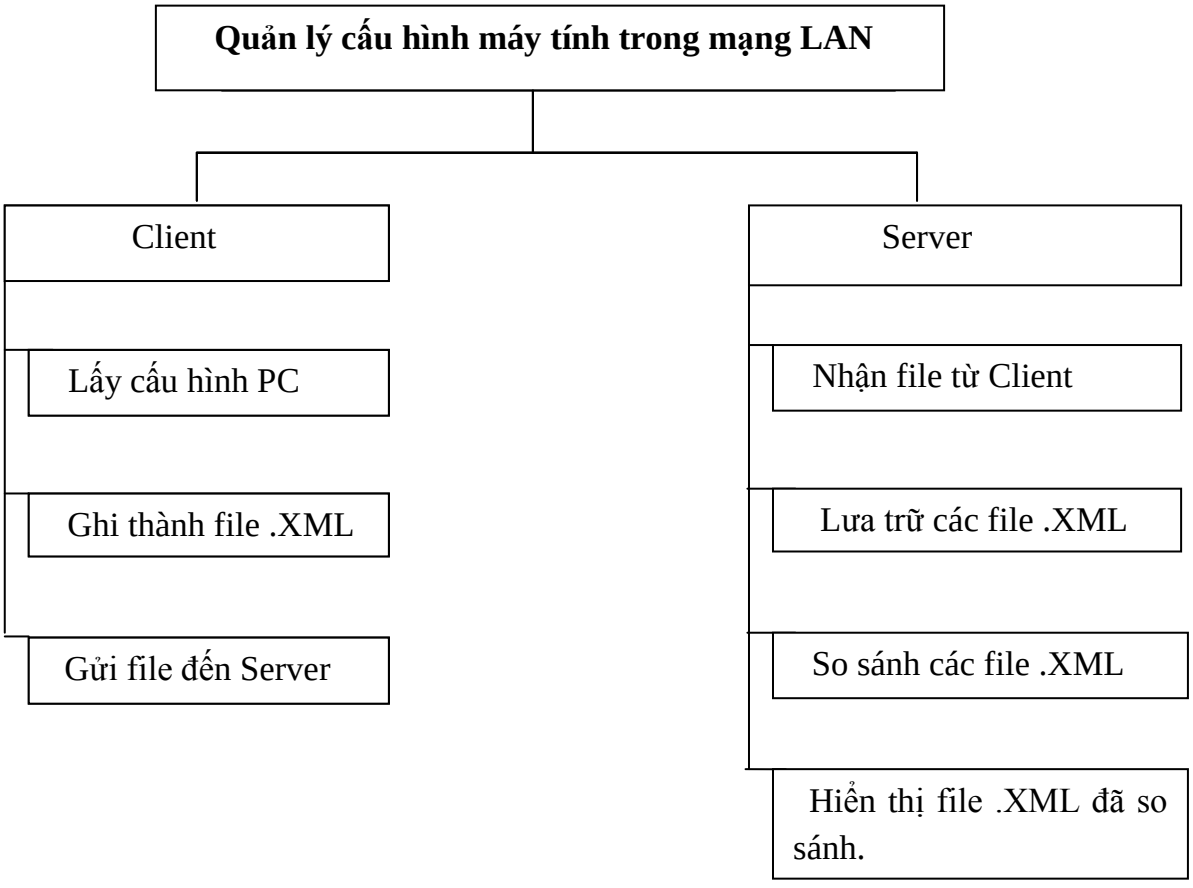
Hình 2.5 Mô hình sử dụng Thread để gửi nhận dữ liệu.

CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

3.1 Mô tả bài toán

Mô hình Client-Server là một mô hình tổ chức trao đổi thông tin trong đó mô tả cách mà các máy tính có thể giao tiếp với nhau theo một phương thức nhất định. Trong đó có một máy tính phục vụ các yêu cầu cho việc xử lý và lưu trữ cho tất cả các máy tính khác trong mạng. Mô hình mạng LAN được thiết lập từ nhiều máy tính khác nhau, trong đó có một máy tính gọi là máy chủ (Server). Một chương trình từ Client chạy từ bất kỳ máy nào trong mạng cũng có thể truyền file .xml đến Server, khi Server nhận được file .xml từ Client nó sẽ thực hiện lưu trữ file ra một thư mục. Nếu file .xml đó đã tồn tại thì Server thực hiện chức năng update file mới đồng thời so sánh file mới nhận được đó với file cũ để tìm kiếm sự khác biệt.

Chương trình xây dựng hệ thống quản lý cấu hình máy tính trong mạng LAN được chia làm 2 phần chính Server và Client với các chức năng sau:



Hình 3.1 Mô hình chức năng quản lý cấu hình máy tính trong mạng LAN

3.2 Phân tích và thiết kế hệ thống

3.2.1 Phân tích và thiết kế các chức năng chương trình

3.2.1.1 Chức năng Server

a. Nhận file từ Client.

Để nhận được file từ Client gửi đến Server cần thực hiện các chức năng sau:

Tạo một socket, liên kết với một IPEndPoint cục bộ.

- Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.IP):

Server yêu cầu tạo một socket để có thể sử dụng các dịch vụ của tầng vận chuyển.

- Bind(): Server yêu cầu gán số hiệu port cho socket.

Lắng nghe và chấp nhận kết nối đến Client.

Để Server có thể lắng nghe kết nối trên cổng đã được gán và chấp nhận nối kết của client để tạo ra kênh giao tiếp ảo giúp Client và Server có thể trao đổi thông tin với nhau, chương trình cần sử dụng 2 phương thức:

- Listen() : Server lắng nghe các yêu cầu kết nối từ các client trên cổng đã được gán.

- Accept(): Server chấp nhận nối kết của client, khi đó một kênh giao tiếp ảo được hình thành, Client và Server có thể trao đổi thông tin với nhau.

Nhận file

Khi một kết nối đã dược mở Client sẽ truyền file cho Server, Server sẽ nhận file đó bằng cách sử dụng phương thức:

- Receive(): Server nhận dữ liệu từ Client thông qua một kênh giao tiếp ảo.

Đóng kết nối sau khi đã hoàn thành và trở lại trạng thái lắng nghe chờ kết nối mới.

Sau khi nhận file dữ liệu từ Client, Server sẽ đóng kết nối và trở lại trạng thái lắng nghe chờ kết nối mới thông qua các phương thức sau:

- Close(): Kênh ảo sẽ bị xóa khi Server đóng socket.

b. Lưu trữ file mới nhận được từ Client.

Server tiến hành lưu trữ các file nhận được từ Client vào thư mục Debug.

c. So sánh file mới nhận được với file cũ.

Khi duyệt tất cả các file .xml trong mục Debug gặp một file .xml chương trình sẽ tìm kiếm trong thư mục D:\OldDoc file đó (tên file=tên máy) , nếu có sẽ Load 2 file để thực hiện việc so sánh.Sau khi so sánh file mới và file cũ thông tin sẽ được hiển thị lên màn hình dạng TreeView.

d. Hiển thị file .xml đã so sánh.

Server sau tiến hành so sánh các file mới trong mục Debug và file cũ trong mục OldDoc đồng thời sẽ hiển thị lên màn hình thông tin cấu hình PC dưới dạng cây.Bằng cách sử dụng lớp XmlDocument, bằng việc tạo một thể hiện của lớp này rồi gọi phương thức Load cùng với một tên file, một Stream, một TextReader, hay một XmlReader (bạn cũng có thể cung cấp một URL chỉ đến một tài liệu XML). Thể hiện XmlDocument sẽ chứa tất cả các nút (dạng cây) có trong tài liệu nguồn.

Khi làm việc với XmlNode hay một lớp dẫn xuất từ đó (như XmlElement hay XmlAttribute), ta có thể sử dụng các thuộc tính cơ bản sau đây:

- ChildNodes là tập hợp các nút lồng bên trong ở mức đầu tiên.
- Name là tên của nút.
- NodeType là một thành viên thuộc kiểu liệt kê System.Xml.XmlNodeType, cho biết kiểu

của nút (phần tử, đặc tính, text...).

- Value là nội dung của nút, nếu đó là nút text hay nút CDATA.
- Attributes là tập hợp các nút mô tả các đặc tính được áp dụng cho phần tử.
- InnerText là chuỗi chứa giá trị (text) của nút hiện hành và tất cả các nút lồng bên trong,…

3.2.1.2 Chức năng Client

a. Lấy cấu hình PC

Việc lấy cấu hình máy tính bao gồm : Địa chỉ IP, tên máy, thông số máy.

- Để lấy địa chỉ IP và tên máy của máy tính hiện hành ta sử dụng phương thức GetHostName() và GetHostByName() của lớp System.Net.

  - GetHostName(): Lấy tên host của máy tính hiện hành.
  - GetHostByName(): Lấy địa chỉ IP trùng khớp đầu tiên.

- Để lấy thông số máy tính hiện hành ta sử dụng ManagementObjectSearcher () để truy xuất systemInfo từ Win32.

b. Ghi cấu hình PC thành file .XML

Sau khi truy xuất được systeminfo hiển thị trong màn hình Dos ta ghi lại thông tin đó thành file .XML bằng cách sử dụng phương thức trong lớp System.XML như sau:

- XmlWriterSettings(): Khởi tạo một thể hiện mới của lớp XmlWriterSettings.
- WriteStartElement(): Một thẻ mở (*opening tag*) cho phần tử bạn chỉ định. Kế đó, bạn có thể thêm nhiều phần tử lồng bên trong phần tử này.
- WriteEndElement(): Để ghi thẻ đóng (*closing tag*).

- WriteAttributeString(): Ghi một đặc tính cho phần tử đang mở gần nhất, cùng với tên và giá trị.

- WriteString(): Viết nội dung văn bản nhất định.

- ToString():Trả về một chuỗi đại diện cho đối tượng hiện hành.(Kế thừa từ Object).

c. Truyền file đến Server.

Tạo socket kết nối đến Server.

Tạo socket phía Client cũng tương tự như phía Server để thực hiện được ta sử dụng các phương thức sau:

- Socket(): Client yêu cầu tạo một socket để có thể sử dụng các dịch vụ của tầng vận chuyển, thông thường hệ thống tự động gán một số hiệu cổng còn trống cho socket của Client.
- Connect(): Client gửi yêu cầu nối kết đến Server có địa chỉ IP và Port xác định.

Truyền file

Để thực hiện chức năng truyền file đến Server, Client gọi phương thức:

- Send():Phương pháp gửi dữ liệu từ bộ đệm đến một Socket kết nối. Khi bạn gọi phương thức send() nó trả về số byte đã được "gửi". Nhưng nó không có nghĩa là các byte đã được nhận bởi phía bên kia, nó chỉ có nghĩa là dữ liệu được lưu trữ trong một bộ đệm socket và socket sẽ phải cố gắng để gửi chúng.

Đóng socket.

Sau khi đã thực hiện truyền file lên Server, Client đóng socket:

- Close(): Kênh ảo sẽ bị xóa khi Client đóng socket.

3.2.2 Thiết kế các lớp

3.2.2.1 Thiết kế lớp Server

Để thực hiện các chức năng đã nêu, phía Server sẽ sử dụng các lớp với một số thuộc tính và phương thức sau:

class Form1
Properties: private System.Windows.Forms.TreeView treeXml; private System.Windows.Forms.NotifyIcon notifyIcon1; private System.Windows.Forms.Timer timer1; private System.Windows.Forms.Label label1; private System.ComponentModel.BackgroundWorker backgroundWorker1; private System.Windows.Forms.Label label2; private System.Windows.Forms.Label label3; FTServerCode fs = new FTServerCode(); XmlDocument document; TreeNode rootnode; XmlNodeList nodes; int sopc = 0;…

Methosd:
Form1();Main();
backgroundWorker1_DoWork(object sender, DoWorkEventArgs e);
Form1_FormClosing(object sender, FormClosingEventArgs e);
Form1_Load_1(object sender, EventArgs e);
Load_PcInfo(string filename, ref Test3.PcInfo.Processor Pro, ref Test3.PcInfo.Main Main, ref Test3.PcInfo.PhysicalMemory Phy, ref Test3.PcInfo.DiskDrive Disk,ref Test3.PcInfo.host Host);
loadTree(string filename);
timer1_Tick(object sender, EventArgs e);

3.2.2.2 Thiết kế lớp Client

Để thực hiện cách chức năng nêu trên phía Client sẽ sử dụng 2 lớp với một số phương thức và thuộc tính sau:

class GetHardwareInfo
Properties: string hostName = Dns.GetHostName(); string ghivao = hostName+".xml"; string ipAddress = Dns.GetHostByName(hostName).AddressList[0].ToString(); ManagementObjectSearcher cpu = new ManagementObjectSearcher("select * from Win32_Processor"); XmlWriterSettings set= new XmlWriterSettings();….

Methosd:
cpu.Get();Main();
w.WriteComment("Thong so PC");
w.WriteStartElement("PC");
w.WriteStartElement("Name");
w.WriteString(hostName);
w.WriteString(ipAddress);
w.WriteEndElement();
p.Name.ToString();p.Value.ToString()
w.Flush();w.Close(),…

class FTClientCode

Properties:

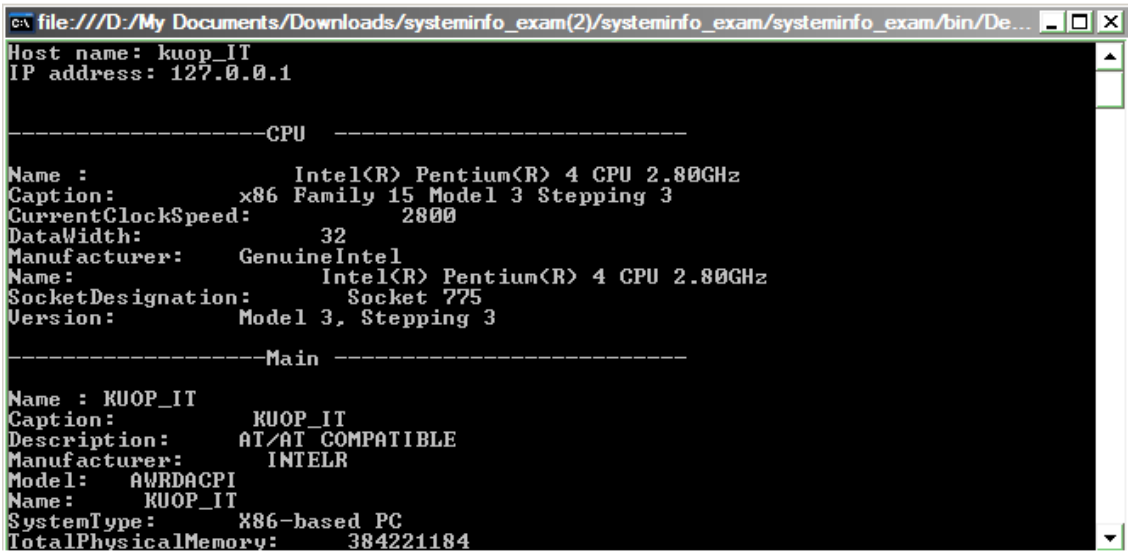
```
public static string curMsg = "Idle";
public static void SendFile(string fileName);
IPAddress[] ipAddress = Dns.GetHostAddresses("127.0.0.1");
IPEndPoint ipEnd = new IPEndPoint(ipAddress[0], 5656);
Socket clientSock =
new  Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.IP);
string filePath = "";
byte[] fileNameByte = Encoding.ASCII.GetBytes(fileName);
byte[] clientData = new byte[4 + fileNameByte.Length + fileData.Length];…
```

Methosd:

```
SendFile(string fileName);
clientSock.Connect(ipEnd);
CopyTo(clientData, 0);
clientSock.Send(clientData);
clientSock.Close();…
```

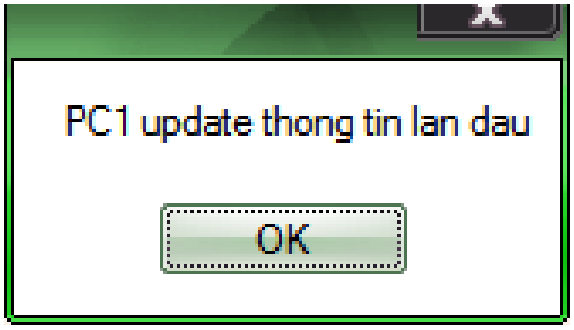
3.3 Một số giao diện chương trình

3.3.1 Giao diện phía Client



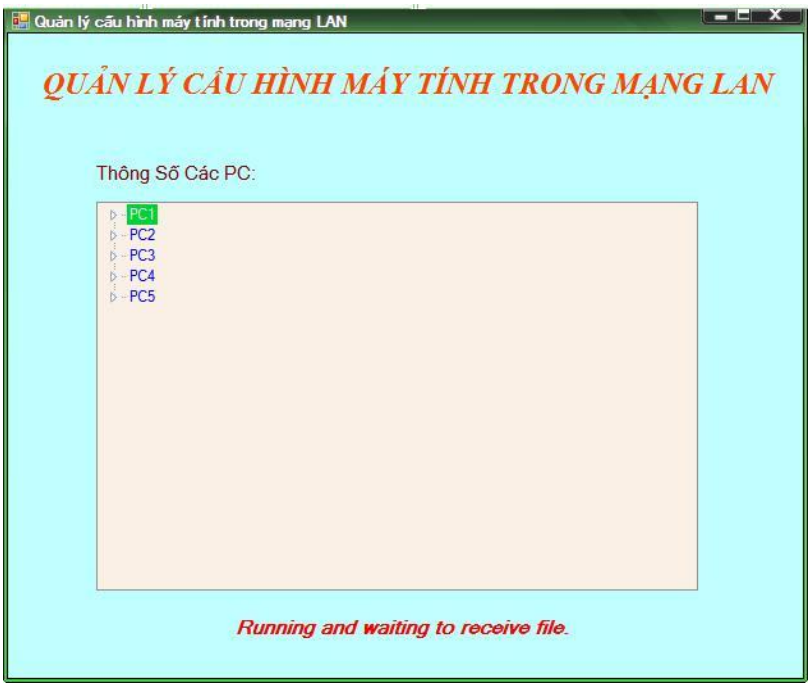
Hình 3.2 Lấy cấu hình PC

3.3.2 Giao diện phía Server

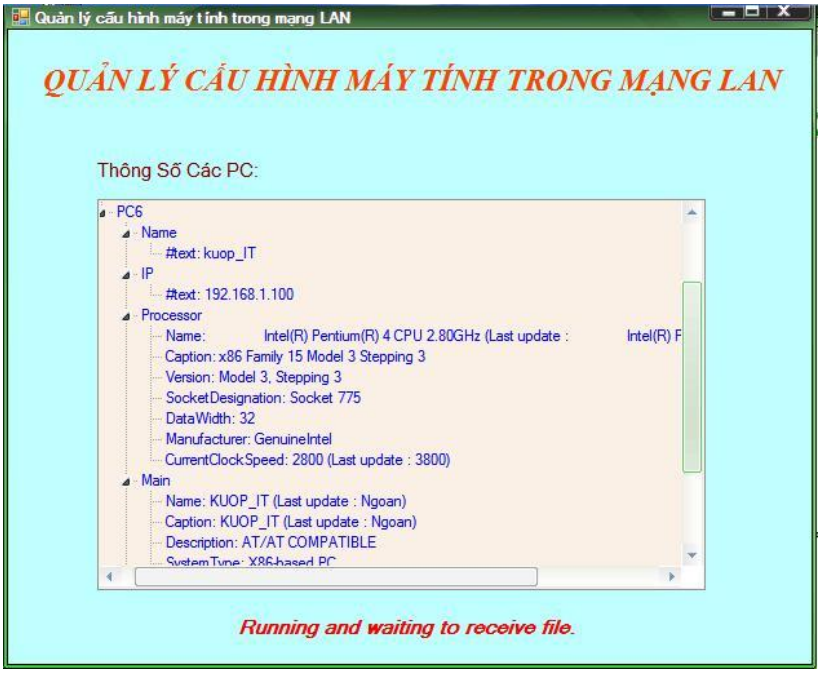


Hình 3.3 PC1 đã cập nhật thông tin lần đầu vào

Hình 3.4 Server đã hiển thị cấu hình PC1 lên TreeView.



Hình 3.5 Server đang chạy và đợi kết nối từ Client



Hình 3.6 File mới nhận được so sánh với file cũ trong thư mục OldDoc.

3.4 Hướng dẫn sử dụng

Chương trình hệ thống quản lý cấu hình máy tính trong mạng LAN được thực hiện khi cho chạy chương trình phía Server trước sau đó tiến hành chạy chương trình bên Client để truyền file tới Server.

- Chạy chương trình bên Client :
  - Lấy cấu hình PC.
  - Ghi thành file .xml tự động lưu trong thư mục Debug đồng thời gửi một yêu cầu Server và gửi file.
- Chạy chương trình bên Server:
  - Hiển thị màn hình đợi kết nối đến từ Client.
  - Nếu nhận được yêu cầu kết nối truyền file từ Client, Server chấp nhận và tiến hành nhận file.
    - Nếu server không tồn tại file cấu hình PC nào từ Client đã gửi tới thì Server sẽ update thông tin file đó lần đầu tiên.
    - Nếu server đã tồn tại các file cấu hình PC do Client gửi đến thì Server tiến hành so sánh các file mới nhận được trong thư mục Debug và các file cũ trong mục OldDoc rồi hiện thông tin đó lên màn hình.

KẾT LUẬN

Đồ án đã sử dụng các hàm có sẵn trong môi trường MS Visual Studio 2005 từ đó xây dựng được ứng dụng truyền file giữa hai máy tính (Client-Server) theo giao thức TCP/IP và thực hiện lưu trữ, quản lý thông tin các cấu hình máy tính trong mạng LAN. Đồ án đã đạt được các yêu cầu đề ra như sau:

Về mặt lý thuyết:

- ✓ Hệ thống lại các kiến thức về mạng căn bản.
- ✓ Tìm hiểu phương pháp lập trình socket trên mạng LAN.

Về mặt thực nghiệm:

- ✓ Xây dựng được hệ thống quản lý cấu hình máy tính trong mạng LAN.

Những vấn đề còn tồn tại cần được giải quyết : Phần thực nghiệm mới chỉ dừng lại ở việc xây dựng một ứng dụng truyền file giữa các máy tính trong mạng LAN theo giao thức TCP/IP và file truyền được là các file có định dạng .xml với kích thước nhỏ, tốc độ truyền chưa cao…

Hướng phát triển trong tương lai: Qua thực nghiệm cho thấy hướng nghiên cứu có thể tiếp tục được nâng cấp để đưa vào sử dụng, trước tiên là với các mô hình nhỏ dùng cho mạng cục bộ trong các phòng ban.Từ đó có thể phát triển và đưa vào sử dụng trên diện rộng nhằm tăng cường tốc độ truyền, khả năng lưu trữ và quản lý một cách khoa học.

TÀI LIỆU THAM KHẢO

Tài liệu Tiếng Việt

[1]. Ban biên dịch Vn-Guide, *Mạng căn bản*, Nhà xuất bản Thống Kê.

[2]. Nguyễn Ngọc Bình Phương, Thái Thanh Phong, *Các giải pháp lập trình C#*, Nhà xuất bản Giao thông vận tải.

[3]. Phạm Hồng Thư (2008-2010), *Đồ án Tốt Nghiệp*, Trường ĐHDL Hải Phòng

Tài liệu Tiếng Anh

[4]. Brian Brown(1996-2000). Networking Fundamentals About This Courseware.