TRƯỜNG CAO ĐẲNG NGHỀ CÔNG NGHIỆP HÀ NỘI
Chủ biên: Bùi Quang Ngọc
Đồng tác giả: Lê Văn Úy

GIÁO TRÌNH
LẬP TRÌNH MẠNG
(Lưu hành nội bộ)

Hà Nội năm 2011

Tuyên bố bản quyền

Giáo trình này sử dụng làm tài liệu giảng dạy nội bộ trong trường cao đẳng nghề Công nghiệp Hà Nội

Trường Cao đẳng nghề Công nghiệp Hà Nội không sử dụng và không cho phép bất kỳ cá nhân hay tổ chức nào sử dụng giáo trình này với mục đích kinh doanh.

Mọi trích dẫn, sử dụng giáo trình này với mục đích khác hay ở nơi khác đều phải được sự đồng ý bằng văn bản của trường Cao đẳng nghề Công nghiệp Hà Nội

MỤC LỤC

BÀI 1 : TỔNG QUAN VỀ LẬP TRÌNH MẠNG.....7
1. Lịch sử phát triển của Lập trình mạng.....7
2. Lý do lập trình mạng trên nền tảng .NET.....7
3. Phạm vi.....8
4. Địa chỉ IP.....9
5. Network stack.....11
6. Port.....12
7. Internet standards.....13
8 .NET framework.....13
9. Visual Studio .NET.....17
BÀI 2 : VẤN ĐỀ I/O TRONG .NET.....26
1. Giới thiệu về không gian tên IO.....26
2. Streams.....26
BÀI 3 : LÀM VIỆC VỚI SOCKETS.....41
1. Giới thiệu về socket trong lập trình mạng.....41
2. Tạo ứng dụng đơn giản “hello world”.....44
3. Dùng giao thức TCP/IP để chuyển files.....46
4. Gỡ rối trong lập trình mạng.....49
5. Mức Socket trong .NET.....50
1. Giới thiệu về HTTP 61
2. HTTP.....62
3. Máy chủ Web (Web servers).....72
4. Làm việc với lớp System.Net.HttpWebListener.....75
5. Trình duyệt Web di động (Mobile Web browsers).....75
BÀI 5 : TRUYỀN THÔNG VỚI EMAIL SERVERS.....77
1. Phương thức gởi và nhận Email 77
2. SMTP 79
3. POP3.....84
4. Làm việc với lớp System.Web.Mail 85
5. Xây dựng ứng dụng Mail.....85
BÀI 6 : TRUYỀN THÔNG VỚI FILE SERVER.....88
1. Tổng quan về File server và truyền File.....88
2. Truyền File.....90
.....111
1. Tổng quan về bảo vệ mạng.....113
2. Tunneling trong mạng doanh nghiệp.....116

3. Tránh những cạm bẫy mạng.....118

MÔ ĐUN LẬP TRÌNH MẠNG

Mã mô đun : MĐ35

Vị trí, tính chất, ý nghĩa và vai trò của Môđun

Vị trí: Mô đun được bố trí vào năm thứ 3 học kì II của khóa học.

Tính chất: Mô đun chuyên môn nghề tự chọn.

Ý nghĩa : Đây là mô đun tự chọn trong chuyên môn nghề, cung cấp cho sinh viên các kỹ năng cơ bản nhất về lập trình mạng, xây dựng các sản phẩm phần mềm để phục vụ công việc quản trị mạng.

Mục tiêu của môđun

Trình bày nguyên lý lập trình mạng, cơ chế hoạt động của chương trình thông qua các Giao thức, hàm truy xuất.

Mô tả mô hình mạng, Giao thức truy cập thông qua các chương trình được cài đặt.

Sử dụng thành thạo các công cụ lập trình Windows hoặc Java để lập trình.

Xây dựng được các ứng dụng mạng : dịch vụ, hệ thống, dữ liệu để bảo vệ hệ thống, giám sát hệ thống, truy vấn dữ liệu….

Bố trí làm việc khoa học đảm bảo an toàn cho người và phương tiện học tập.

Nội dung của môn học

Số TT	Tên các bài trong mô đun	Thời gian			
		Tổng số	Lý thuyết	Thực hành	Kiểm Tra*
1	Tổng quan lập trình mạng	10	5	5	
2	Vấn đề I/O trong .NET	10	4	5	1
3	Làm việc với Sockets	17	5	12	
4	Truyền thông với Web Servers	13	4	8	1
5	Truyền thông với Mail Servers	16	6	10	
6	Truyền thông với File server	13	4	8	1
7	Bảo mật mạng :Firewalls, Proxy Servers, and Routers	11	2	8	1
	Cộng	90	30	56	4

BÀI 1 : TỔNG QUAN VỀ LẬP TRÌNH MẠNG

Mã bài : MĐ35.01

Mục tiêu của bài :

- Trình bày các vấn đề về điều hành mạng và lập trình mạng: Vấn đề truyền thông tin, địa chỉ IP, Giao thức, các tầng liên lạc và tính phân cấp của các giao thức, thông điệp.
- Trình bày được các thành phần của môi trường .NET Framwork.
- Thực hiện các câu lệnh cơ bản của Visual Studio .NET.
- Thực hiện các thao tác an toàn với máy tính.

1. Lịch sử phát triển của Lập trình mạng

Cuốn sách này sẽ giúp bạn phát triển các ứng dụng mạng với NET, bằng cách sử dụng C (phát âm là C-sharp) hoặc ngôn ngữ lập trình VB.NET. Nó được chia thành ba phần riêng biệt: vấn đề cơ bản kết nối mạng, thiết kế ứng dụng phân tán, và các chủ đề mạng chuyên ngành. Sáu chương đầu tiên của cuốn sách bao gồm các công nghệ Internet thành lập, chẳng hạn như email và World Wide Web. Tận dụng công nghệ thành lập như thế này cho phép truy cập công cộng nói chung lớn hơn cho dịch vụ phần mềm của bạn bởi vì hầu hết người dùng đã có một trình duyệt web hoặc ứng dụng email trên máy tính của họ. Năm chương tiếp theo thảo luận về thiết kế ứng dụng mạng. Điều này bao gồm bảo mật ứng dụng, hiệu suất, và khả năng mở rộng. Chứa trong các chương này là thực tế, thực hành lời khuyên để giúp nâng cao chất lượng tổng thể của phần mềm của bạn. Với bảo mật khó khăn hơn, các ứng dụng của bạn sẽ ít nhạy cảm với hành vi trộm cắp sở hữu trí tuệ và các thông tin đặc quyền. Cải tiến hiệu suất và khả năng mở rộng được mô tả trong phần này sẽ đảm bảo rằng ứng dụng của bạn vẫn đáp ứng ngay cả dưới tải cực đoan nhất. Các mạng phần chuyên đề cung cấp vô số thông tin về cả hai thích hợp và các công nghệ Internet tiên tiến. Chúng bao gồm các chương về điện thoại, chụp gói, hàng đợi tin nhắn, IPv6, và dịch vụ mới nhất của Microsoft trong lĩnh vực phát triển ứng dụng phân tán: dịch vụ Web và truy cập từ xa.

2. Lý do lập trình mạng trên nền tảng .NET

Một trong những quyết định kỹ thuật đầu tiên được thực hiện bất cứ khi nào một dự án mới được thực hiện là ngôn ngữ để sử dụng. NET là một nền tảng có khả năng để phát triển hầu như bất kỳ giải pháp, và nó cung cấp hỗ trợ đáng kể cho lập trình mạng. Trong thực tế,. NET có hỗ trợ nội tại cho mạng hơn so với bất kỳ nền tảng khác được phát triển bởi Microsoft. Cuốn sách này giả định rằng bạn đã quyết định để phát triển với NET, và ngôn ngữ

bên ngoài nền tảng NET sẽ không được thảo luận trong bất kỳ chi tiết tuyệt vời, ngoại trừ cho mục đích so sánh. Điều này không phải là để nói rằng NET là được-tất cả và cuối cùng tất cả các ứng dụng lập trình mạng. Nếu ứng dụng của bạn chạy trên một cơ sở hạ tầng UNIX chỉ giao tiếp thông qua Java gọi phương thức từ xa (RMI), sau đó. NET không phải là con đường để đi. Trong hầu hết các trường hợp, tuy nhiên, bạn sẽ tìm thấy điều đó. NET là nhiều hơn khả năng xử lý bất cứ điều gì bạn ném vào nó.

3. Phạm vi

Một chương trình mạng là bất kỳ ứng dụng mà sử dụng một mạng máy tính để chuyển thông tin đến và đi từ các ứng dụng khác. Ví dụ từ trình duyệt web phổ biến như Internet Explorer, hoặc chương trình mà bạn sử dụng để nhận email của bạn, phần mềm điều khiển tàu vũ trụ tại NASA. Tất cả các thành phần này chia sẻ phần mềm khả năng giao tiếp với các máy tính khác, và khi làm như vậy, trở nên hữu ích hơn cho người sử dụng cuối. Trong trường hợp của một trình duyệt, tất cả các trang web bạn truy cập được các tập tin được lưu trữ trên một máy tính ở một nơi khác trên Internet. Với chương trình email của bạn, bạn đang giao tiếp với một máy tính tại nhà cung cấp dịch vụ Internet (ISP) của bạn hoặc trao đổi email của công ty được tổ chức email của bạn cho bạn Cuốn sách này là chủ yếu quan tâm đến việc tạo ra các chương trình mạng, không Web các trang web. Mặc dù khả năng của những trang web và các chương trình mạng một cách nhanh chóng hội tụ, nó là quan trọng để hiểu các đối số và đối với mỗi hệ thống. Một dịch vụ truy cập thông qua một trang web có thể truy cập ngay lập tức để người sử dụng trên nhiều nền tảng khác nhau, và toàn bộ kiến trúc mạng đã sẵn sàng xây dựng cho bạn, tuy nhiên, có một điểm mà tại đó tính năng này chỉ đơn giản là không khả thi để thực hiện bằng cách sử dụng các trang web và mà tại đó bạn có chuyển sang mạng các ứng dụng.

Người dùng thường tin tưởng các ứng dụng mạng, do đó, các chương trình này có quyền kiểm soát lớn hơn đối với các máy tính mà họ đang chạy hơn một trang web trên máy tính xem nó. Điều này làm cho nó có thể cho một ứng dụng mạng để quản lý các tập tin trên máy tính địa phương, trong khi một trang web, cho tất cả các mục đích thực tế, không thể làm điều này. Quan trọng hơn, từ góc độ kết nối mạng, một ứng dụng có quyền kiểm soát lớn hơn đối với làm thế nào nó có thể giao tiếp với các máy tính khác trên Internet..

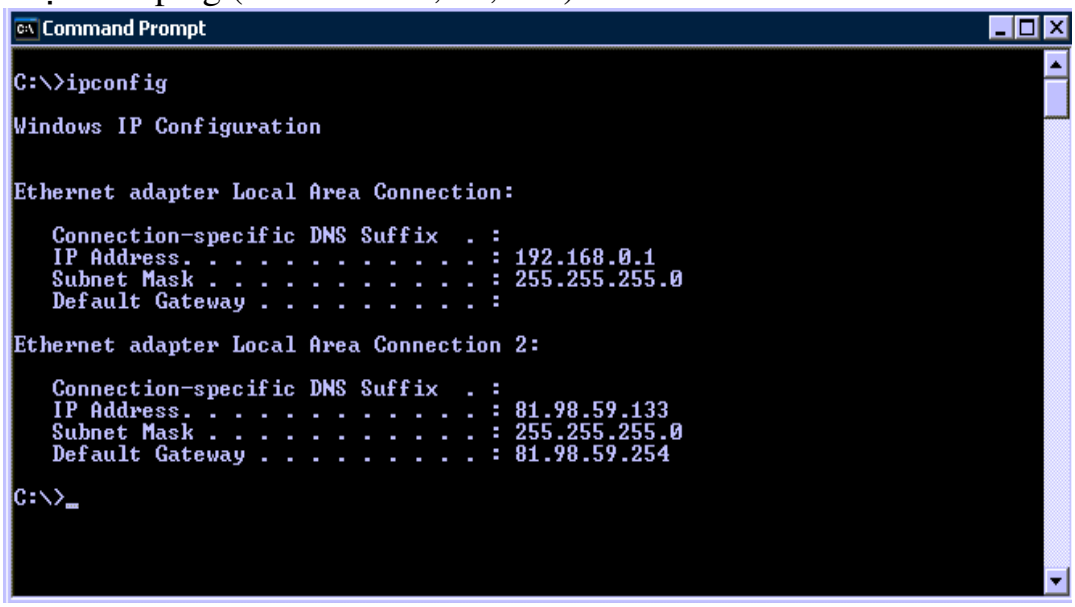
Có một ngoại lệ cho quy tắc này, khi thực thi nội dung (chẳng hạn như một điều khiển ActiveX) được bao gồm trong một trang. Trong trường hợp này, trang này là khả năng của tất cả mọi thứ có thể làm một chương trình mạng, nhưng hầu hết các trình duyệt và phần mềm chống virus sẽ cảnh báo chống lại hoặc phủ nhận nội dung thực thi như vậy. Vì vậy, kịch bản này thường được chấp nhận là không khả thi vì mất lòng tin công cộng. Để đưa ra một ví dụ đơn giản, một trang web không có thể làm cho máy tính được

xem nó mở một kết nối mạng liên tục cho các máy tính khác (ngoại trừ các máy tính mà từ đó các trang web đã được phục vụ). Điều này áp dụng ngay cả khi các trang web có chứa nội dung được nhúng như một applet Java hoặc phim Flash. Có một ngoại lệ cho quy tắc này, khi thực thi nội dung (chẳng hạn như một điều khiển ActiveX) được bao gồm trong một trang. Trong trường hợp này, trang này là khả năng của tất cả mọi thứ có thể làm một chương trình mạng, nhưng hầu hết các trình duyệt và phần mềm chống virus sẽ cảnh báo chống lại hoặc phủ nhận nội dung thực thi như vậy. Vì vậy, kịch bản này thường được chấp nhận là không khả thi vì mất lòng tin công cộng.

4. Địa chỉ IP

Mỗi máy tính kết nối trực tiếp với Internet phải có một địa chỉ duy nhất trên toàn cầu IP. Một địa chỉ IP là một số có bốn byte, mà thường được viết là bốn thập phân, số thời gian cách nhau, chẳng hạn như 192.168.0.1. Máy tính kết nối gián tiếp với Internet, chẳng hạn như thông qua mạng công ty của họ, cũng có địa chỉ IP, nhưng chúng không cần phải được trên toàn cầu duy nhất, chỉ có duy nhất trong cùng một mạng.

Để tìm ra những địa chỉ IP của máy tính của bạn, mở một cửa sổ giao diện điều khiển hệ điều hành DOS và loại ipconfig (Windows NT, 2000, and XP) hoặc winipcfg (Windows 95, 98, ME)



Trong hình, 1.1 máy tính. có hai địa chỉ IP: 192.618.0.1 và 81.98.59.133. Điều này là không bình thường bởi vì máy tính này đặc biệt có chứa hai card mạng và được kết nối với hai mạng khác nhau. Chỉ có một trong những địa chỉ IP truy cập công khai.

Nếu bạn nhận được địa chỉ IP 127.0.0.1, máy tính của bạn không kết nối với bất kỳ mạng nào. Địa chỉ IP này luôn luôn đề cập đến các máy tính địa phương và được sử dụng trong các ví dụ sau. Trong cùng một cách mà bạn có thể nói cho dù một số điện thoại là địa phương hoặc quốc tế bằng cách nhìn vào tiền tố, bạn có thể cho biết liệu máy tính với địa chỉ IP trên

mạng cùng một khu vực địa phương hoặc một nơi nào khác trên Internet bằng cách nhìn chặt chẽ tại một địa chỉ IP. Trong trường hợp các địa chỉ IP, họ luôn luôn cùng độ dài, nhưng tiền tố nhất định (192,168 phổ biến nhất) chỉ ra rằng máy tính trong một mạng lưới khu vực địa phương, hoặc mạng nội bộ, và không thể truy cập vào thế giới bên ngoài. Nếu bạn chia sẻ kết nối Internet của bạn với các máy tính khác trên mạng của bạn, bạn có thể có một địa chỉ IP riêng. Đây có thể được công nhận là trong phạm vi địa chỉ IP được liệt kê trong Bảng 1.1.

Dãy các địa chỉ	**Số lượng các địa chỉ**
10.0.0.0 đến 10.255.255.255	Tên 16 tỉ máy tính (lớp A)
172.16.0.0 đến 172.31.255.255	900.000 máy tính (lớp B)
192.168.0.0 đến 192.168.255.255	65.000 máy tính (lớp C)

Bảng 1.1 : Bảng liệt kê địa chỉ máy tính

Cùng một địa chỉ IP riêng có thể tồn tại trên hai máy tính trong các mạng khu vực khác nhau cục bộ (LAN). Điều này không gây ra một vấn đề bởi vì không phải máy tính có thể trực tiếp liên lạc với nhau. Trong khi đó, một máy tính giải quyết tư nhân có thể bắt đầu một yêu cầu thông tin từ một máy tính nước ngoài, không có máy tính nước ngoài có thể bắt đầu một yêu cầu thông tin từ một máy tính cá nhân giải quyết. Các trường hợp ngoại lệ cho quy tắc này sẽ là nơi network address translation (NAT) hoặc cổng chuyển tiếp được thiết lập trên router nằm ở thượng nguồn của máy tính tư nhân giải quyết. Đây là nơi mà các yêu cầu từ máy móc nước ngoài dành cho các địa chỉ IP của router được chuyển tiếp đến một com-puter định phía sau router. Câu trả lời từ máy tính này được chuyển tiếp từ phía sau router máy nước ngoài bắt đầu yêu cầu. Những lợi ích của kiến trúc an ninh và khả năng cân bằng tải, được mô tả chi tiết hơn trong chương sau. Tất cả các máy tính có địa chỉ IP riêng phải được kết nối với ít nhất một máy tính hoặc router mạng với một địa chỉ IP công cộng. truy cập Internet.

Để đảm bảo rằng không có hai máy tính trên Internet có địa chỉ cùng một IP, có một cơ quan quản lý trung ương được gọi là Internet Assigned Numbers Authority (IANA), và gần đây Tổng công ty Internet cho tên miền và số (ICANN). Cơ thể này hoạt động thông qua nhà cung cấp dịch vụ Internet để gán địa chỉ IP công cộng cho các tổ chức và cá nhân Mặc dù có thể được phân bổ một địa chỉ IP tại một thời điểm, nó là nhiều hơn phổ biến được phân bổ địa chỉ IP trong khối tiếp giáp. Tiếp giáp khối có ba lớp học: A, B, và C. Class A địa chỉ đều là các khối địa chỉ IP với byte đầu tiên chỉ. Class A là hơn 16 triệu địa chỉ IP trong kích thước. Địa chỉ lớp B là khối địa chỉ IP với byte đầu tiên và thứ hai. Lớp B giữ 65.024 địa chỉ IP công cộng. 216 phạm vi byte đầy đủ là không có bởi vì byte cuối cùng của một địa chỉ IP không thể là 0 hoặc 255 bởi vì chúng được dành riêng để sử dụng trong tương lai. Địa chỉ lớp C là những khối địa chỉ IP với byte đầu tiên, thứ hai, và thứ ba. Class C nắm giữ 254 địa chỉ công cộng, và địa chỉ lớp C

được thường xuyên giao cho các công ty. Một máy tính có thể không phải lúc nào cũng có cùng một địa chỉ IP. Nó có thể có được địa chỉ của nó IP từ máy chủ ISP năng động điều khiển máy chủ của bạn (DHCP) giao thức. Điều này có nghĩa là địa chỉ IP của bạn có thể thay đổi mỗi khi bạn đi trực tuyến. Một địa chỉ IP như vậy được gọi là một địa chỉ IP động. Nếu bạn đang trên một mạng nội bộ, bạn có thể kiểm tra xem nếu địa chỉ IP của bạn là chịu trách nhiệm thay đổi bằng cách kiểm tra "có được địa chỉ IP tự động" nút radio trong thuộc tính TCP / IP, theo mạng trong bảng điều khiển. Mục đích của DHCP là nếu có một số lượng hạn chế của IP địa chỉ có sẵn cho các ISP, nó sẽ phân bổ các thuê bao của mình với IP địa chỉ từ một hồ bơi trên cơ sở ai đến trước được phục vụ trước. Địa chỉ IP là số 32-bit, với một giá trị tối đa khoảng 4 tỷ đồng, và số lượng các máy tính trên thế giới đang nhanh chóng tiếp cận con số đó. IPv6 là một giải pháp cho vấn đề đó và được thảo luận trong chương sau.

Có một định danh được xây dựng vào tất cả các card mạng mà là thực sự
duy nhất và không thể thay đổi. Điều này được gọi là phần cứng, hoặc truy cập địa chỉ kiểm soát các phương tiện truyền thông (MAC). Một địa chỉ mẫu MAC là 00-02-E3-15-59-6C này được sử dụng trên mạng nội bộ để xác định các máy tính khi họ đăng nhập vào mạng. Một hệ thống được gọi là giao thức phân giải địa chỉ (ARP) được sử dụng để kết hợp địa chỉ MAC với địa chỉ IP.

5. Network stack

Các tín hiệu kỹ thuật số mà đi dò đường giữa các máy tính trên Internet vô cùng phức tạp. Nếu không có các khái niệm về đóng gói, các lập trình sẽ nhanh chóng trở thành sa lầy với các chi tiết không đáng kể. Kỹ thuật này được sử dụng trong cuộc sống hàng ngày, nơi bạn có thể yêu cầu một tài xế taxi
để đưa bạn đến trung tâm thành phố. Đó là trách nhiệm của lái xe taxi để tìm ra con đường nhanh nhất và để vận hành xe. Ở mức thấp hơn một lần nữa, nó là chiếc xe của nhà sản xuất chịu trách nhiệm đảm bảo rằng xăng sẽ có mặt trong các động cơ piston trong khi máy gia tốc là chán nản. Đóng gói là các chi tiết phức tạp của một nhiệm vụ ẩn,và lập trình viên chỉ cần tập trung vào những gì đang xảy ra ở một mức độ cao hơn. Các kết nối hệ thống mở (OSI) mô hình mạng ngăn xếp có bảy lớp đóng gói, như thể hiện trong Bảng 1.2.

Trong lập trình hiện đại, tuy nhiên, mạng ngăn xếp trông giống như Bảng 1.3 lớp quan trọng nhất cho bất kỳ lập trình viên là lớp cao nhất vì điều này sẽ đủ khả năng dễ dàng sử dụng và sẽ phù hợp với hầu hết các ứng dụng. Khi bạn đi xuống ngăn xếp, thực hiện trở nên khó khăn hơn, mặc dù linh hoạt hơn.

Bảng 1.2 : các lớp truyền thống

Lớp	Tên lớp	Giao thức

Level 7	Application layer	FTP
Level 6	Presentation layer	XNS
Level 5	Session layer	RPC
Level 4	Transport layer	TCP
Level 3	Network layer	IP
Level 2	Data-Link layer	Ethernet Frames
Level 1	Physical layer	Voltages

Bảng 1.3: Các lớp hiện đại

Lớp	Tên lớp	Giao thức
Level 4	Structured Information layer	SOAP
Level 3	Messaging layer	HTTP
Level 2	Stream layer	TCP
Level 1	Packet layer	IP

Cuốn sách này bao gồm các lớp ứng dụng chủ yếu, nhưng đảm bảo được đưa ra cho tất cả các lớp khác nhau, không bao gồm lớp vật lý, mà sẽ chỉ áp dụng cho các kỹ sư điện tử.

Trong lập trình mạng, bạn thường không cần phải quan tâm mình với cách thức thông tin truyền giữa hai máy tính, chỉ với những gì bạn muốn gửi. Chi tiết tốt hơn được đặt ở các cấp thấp hơn và được kiểm soát bởi hoạt động của máy tính hệ thống.

6. Port

Nếu bạn muốn trình duyệt Web và nhận email cùng một lúc, máy tính của bạn cần phải quyết định bit lưu lượng truy cập mạng email và các trang web. Để biết sự khác biệt, tất cả các mảnh dữ liệu trên mạng được gắn thẻ với một cổng số: 80 cho các trang Web, 110 cho các email gửi đến. Thông tin này được chứa trong một trong hai giao thức điều khiển ransmission (TCP) hoặc User Datagram Protocol (UDP) tiêu đề đó ngay lập tức sau header IP. Bảng 1.4 liệt kê các giao thức phổ biến và liên kết số cổng.

Số hiệu cổng	Quá trình hệ thống
7	Dịch vụ Echo
21	Dịch vụ FTP
23	Dịch vụ Telnet
25	Dịch vụ E-mail (SMTP)
80	Dịch vụ Web (HTTP)
110	Dịch vụ E-mail (POP)

Bảng 1.4 liệt kê các giao thức phổ biến và liên kết số cổng

Qui định :

- Không bao giờ có hai ứng dụng lại dùng cùng 1 Port
- Các Port từ 0 -> 1023 : Dùng cho các ứng dụng quan trọng trên hệ điều hành.
- Các Port từ 1024 -> 49151 : Dành cho người lập trình.
- Các Port từ 49152 -> 65535 :Dự trữ

7. Internet standards

Chỉ có các tập đoàn lớn có thể trở thành thành viên của W3C. W3C là chịu trách nhiệm về ngôn ngữ đánh dấu siêu văn bản (HTML), cascading style sheets (CSS), ngôn ngữ đánh dấu mở rộng (XML). Khi phát triển một ứng dụng mạng, điều quan trọng là không để tái tạo lại bánh xe hoặc nếu không tạo ra một ứng dụng đó là không cần thiết không tương thích với các ứng dụng khác cùng thể loại. Cuốn sách này thường đề cập đến các tài liệu tiêu chuẩn, do đó, nó là đáng giá biết nơi để tìm thấy chúng. Một tấm gương sáng là năng động HTML, được thực hiện khác nhau trên Internet Explorer và Netscape Navigator. Điều này có nghĩa rằng hầu hết các trang web sử dụng HTML năng động sẽ không hoạt động đúng trên tất cả các trình duyệt. Vì vậy, các nhà phát triển Web tránh nó và di chuyển về phía công nghệ qua trình duyệt, chẳng hạn như Macromedia Flash và Java Applet. Lý do cho sự sụp đổ này là thiếu tiêu chuẩn hóa. Hai tổ chức chịu trách nhiệm chính để điều tiết Internet tiêu chuẩn: Internet Engineering Task Force (IETF) và World Wide Web Consortium (W3C). IETF là một tổ chức phi lợi nhuận, trong đó quy định các giao thức cơ bản nhất trên Internet. Bất cứ ai cũng có thể gửi một giao thức để họ và nó sẽ được công bố công khai như là một yêu cầu cho ý kiến (RFC) trên trang web của họ tại www.ietf.org / rfc.html. Bảng 1.5 liệt kê một số tài liệu quan trọng RFC. W3C (www.w3c.org) được thiết kế để tạo thuận lợi cho khả năng tương tác tiêu chuẩn trong số các nhà cung cấp. Chỉ có các tập đoàn lớn có thể trở thành thành viên của W3C. W3C là chịu trách nhiệm về ngôn ngữ đánh dấu siêu văn bản (HTML), cascading style sheets (CSS), ngôn ngữ đánh dấu mở rộng (XML).

8 .NET framework

.NET Framework là hạ tầng cơ bản được chuẩn hoá, độc lập ngôn ngữ lập trình, cho phép người lập trình xây dựng, tích hợp, biên dịch, triển khai, chạy các dịch vụ Web, XML, tiện ích hay thực thi chương trình đa cấu trúc (phát triển bằng các ngôn ngữ lập trình hỗ trợ .NET) trên hệ điều hành có cài đặt .NET Framework.

8.1. Thành phần .NET Framework

.NET Framework bao gồm 2 phần chính là Common Language Runtime (CLR) và .NET Framework Class Library (FCL).

CLR là thành phần chính của .NET Framework, quản lý mã (code) có thể thực thi của chương trình, quản lý các tiến trình, quản lý tiểu trình (Threading), quản lý bộ nhớ, cung cấp dịch vụ để biên dịch, tích hợp và tác vụ truy cập từ xa (Remoting).

FCL bao gồm tất cả các dịch vụ như giao tiếp người sử dụng, điều khiển, truy cập dữ liệu, XML, Threading, bảo mật.

Tóm lại, CLR được xem như máy ảo .NET (.NET Virtual Machine), nó có thể kiểm soát, nạp và thực thi chương trình .NET.

Trong khi đó, FCL cung cấp các lớp, giao tiếp và các kiểu giá trị, phương thức truy cập và chức năng chính của hệ thống như: Microsoft.Csharp, Microsoft.Jscript, Microsoft.VisualBasic, Microsoft.Vsa, Microsoft.Win32, System (cùng với các không gian tên con của không gian tên System).

Microsoft.Csharp : cung cấp các lớp hỗ trợ biên dịch và phát sinh mã khi sử dụng ngôn ngữ lập trình C#.

Microsoft.Jscript : cung cấp các lớp hỗ trợ biên dịch và phát sinh mã khi sử dụng ngôn ngữ lập trình J#.

Microsoft.VisualBasic : cung cấp các lớp hỗ trợ biên dịch và phát sinh mã khi sử dụng ngôn ngữ lập trình VisualBasic.

Microsoft.Vsa : cung cấp các gia tiếp cho phép tích hợp với các kịch bản của .NET Framework vào ứng dụng khi biên dịch hay thực thi.

Microsoft.Win32: cung cấp hai lớp giao tiếp trực tiếp với tài nguyên của hệ điều hành và System Registry.

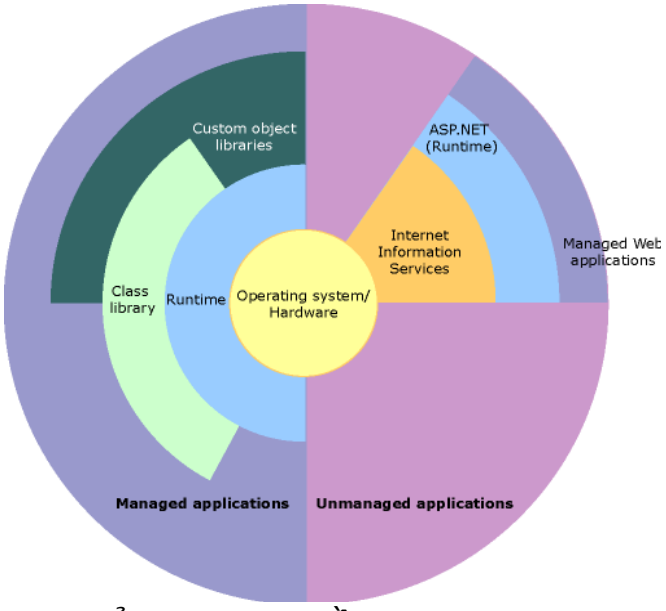
System: bao gồm các lớp cơ sở dùng để định nghĩa giá trị, tham chiếu, biên cố, giao tiếp, thuộc tính và kiểm soát ngoại lệ. Ngoài ra, một số lớp khác cung cấp các dịch vụ chuyển đổi kiểu dữ liệu, tham số, tính toán, xử lý và truy cập từ xa.

Trong đó, Code bao gồm hai loại :

Manage Code: bao gồm những chương trình được tạo ra từ các ngôn ngữ lập trình có hỗ trợ .NET, chẳng hạn, sử dụng ngôn ngữ lập trình C# để phát triển chương trình ứng dụng A, sau đó biên dịch chúng ra tập tin thi hành (.EXE), tập tin .EXE này đươc gọi là Manage Code trong môi trường .NET.

Unmanage Code : là những chương trình được tạo ra từ các ngôn ngữ lập trình ngoài .NET. Ví dụ, sử dụng ngôn ngữ lập trình Visual Basic 6.0 để khai báo lớp (Class) có tên là B, rồi biên dịch chúng ra tập tin thư viên (.DLL), tập tin .DLL được gọi là Unmanage Code khi tham chiếu chúng trong môi trường .NET

Như vậy, .NET Framework còn gọi là môi trường tương tác với hệ điều hành cho các ứng dụng và được minh hoạ như hình sau :



Hình 1.1: Mô tả các thành phần trong .NET Framework

8.2. Những đặc điểm chính của .NET Framework

.NET Framework bao gồm các đặc điểm chính như : CRL, FCL, Cômmn Type System (kiểu dư liệu thông dụng, Metadata and Selff Descring Component phần chính Siêu dữ liệu và tự đặc tả thành phần). Cross-Language Interopenrability (trao đổi và sử dụng), Assemblies (đơn vị phân phối), Application Domains (miền ứng dụng) và Runtime Host (trung tâm thi hành)

CLR : CLR là môi trường thi hành, nơi cung cấp dịch vụ để thực thi, quản lý bộ nhớ, tiểu trình cho các ứng dụng hỗ trợ bởi .NET

- Quản lý quá trình thực thi: để quản lý quá trình thực thi của trình, CLR thực hiện qua các bước sau: chọn chương trình biên dịch tương ứng với ngôn ngữ lập trình, biên dịch ứng dụng sang tập tin MSIL (trình bày chi tiết trong phần biên dịch và thực thi ứng dụng), biên dịch từ mã định dạng MSIL sang mã máy bằng trình JIT (Just-In-Time) rối sau đó CLR cung cấp cơ sở hạ tầng để thi hành chương trình.

- Quản lý bộ nhớ: tự quản lý bộ nhớ là một trong những dịch vụ mà CLR cung cấp trong quá trình thực thi chương trình. Trình thu gom (Garbage Collector) quản lý bộ nhớ đã cấp cho một tiến trình rồi sau đó tự động thu lại khi chương trình kết thúc (chúng ta sẽ tìm hiểu chi tiết về Garbage Collector trong cuốn sách " lập trình hướng đối tượng" sắp phát hành).

FCL: Bao gồm các thư viện lớp cơ sở cho phép bạn sử dụng để thực hiện mọi tác vụ liên quan đến giao diện, Internet, cơ sở dữ liệu, hệ điều hành,…

Common Type System (CTS): CTS đưa ra các quy tắc cho phép bạn khai báo, sử dụng và quản lý kiểu dữ liệu trong quá trình thi hành. Ngoài ra, CTS còn cung cấp các tiêu chuẩn cho phép phát hành tương tác giữa các ngôn ngữ lập trình với nhau. Tóm lại, CTS thực hiện các chức năng chính sau:

- Thiết lập khung cho phép tương tác giữa các ngôn ngữ, mã an toàn(safe code), tối ưu hóa xử lý.
- Cung cấp mô hình hướng đối tượng nhằm hỗ trợ quá trình cài đặt đa ngôn ngữ trong ứng dụng.
- Định nghĩa các quy tắc mà ngôn ngữ lập trình phải tuân theo và hỗ trợ tính chuyển đổi và bảo đảm đối tượng được tạo ra từ ngôn ngữ này có thể tương tác với ngôn ngữ khác.

Metadata and Self-Descrinbing Components (MSDC): trong những phiên bản trước đây, ứng dụng được tạo ra bởi một ngôn ngữ lập trình nào đó được biên dịch ra tập tin .EXE hay .DLL và khó khăn khi sử dụng chúng với một ứng dụng được viết trong một ngôn ngữ lập trình khác. Ví dụ COM là một điển hình. Tuy nhiên, .NET framework cung cấp giải pháp chuyển đổi cho phép khai báo thông tin cho mọi module và Assembly (có thể là .EXE hay .DLL). Những thông tin này được gọi là

siêu dữ liệu và sự mô tả.

Cross Language Interoperability (CLI): CLR là hỗ trợ tiến trình trao đổi và sử dụng giữa các ngôn ngữ với nhau.Tuy nhiên, hỗ trợ này không bảo đảm mã do bạn viết có thể dùng được bởi lập trình viên sử dụng ngôn ngữ lập trình khác.

Assemblies: là tập hợp các kiểu dữ liệu và tài nguyên được đóng gói dạng từng đơn vị chức năng. Ngoài ra, assemblies chính là các đơn vị chủ yếu dùng để triển khai, điều khiển phiên bản, thành phần sử dụng lại, chẳng hạn như các tập tin .EXE hay .DLL.

Applicatin Domains: miền ứng dụng cho CLR quản lý nhằm cách ly nhiều ứng dụng đang thi hành trên cùng một máy tính cụ thể:

- Mỗi ứng dụng sẽ được nạp vào tiến trình(Process) tách biệt mà không ảnh hưởng đến ứng dụng khác. Với kỹ thuật kiểu mã an toàn Application Domains bảo đảm đoạn mã đang chạy trong miền ứng dụng độc lập với tiến trình của ứng dụng khác trên cùng một máy.

- Khi tạm dừng từng thành phần thì sẽ không dừng toàn bộ tiến trình. Đối với trường hợp này Application Domains cho phép bạn loại bỏ đoạn mã đang chạy trong ứng dụng đơn.
- Application Domains cho phép bạn cấu hình, định vị, cấp quyền hay hạn chế quyền sử dụng tài nguyên đang thi hành.
- Ngoài ra, sự cách ly này cho phép CLR ngăn cấm truy cậptruwcj tiếp giữa các đối tượng của những ứng dụng khác nhau.

Runtime Hosts: là trung tâm thi hành cho phép nạp ứng dụng vào tiến trình, CLR hỗ trợ cho phép nhiều loại ứng dụng khác nhau cùng chạy trong một tiến trình.

- Mỗi loại ứng dụng thì cần đoạn mã để khởi động được gọi là Runtime hosts.
- Runtime Hosts nạp kênh thi hành vào tiến trình và tạo ra Application Domains ròi thi hành ứng dụng vào trong miền ứng dụng đó.

9. Visual Studio .NET

Microsoft Visual Studio là tập công cụ hoàn chỉnh dùng để xây dựng ứng dụng Web (ASP.NET Web Applications), dịch vụ XML, ứng dụng để bàn (Desktop application), ứng dụng màn hình với bàn phím (Console Applications) và ứng dụng trên điện thoại di động (Mobile Applications).

Các ngôn ngữ lập trình dùng Microsoft Visual studio để phát triển ứng dụng là Visual basic, Visual C++, Visual C# và Visual J#. Cả 4 ngôn ngữ lập trình chính trên đều sử dụng chung một IDE (Integrated Development Environment), nơi cho phép chúng ta chia sẻ các tiện ích và công cụ nhằm tạo nên giải pháp tích hợp.

Nếu đã làm việc với phiên bản Visual Studio 6.0, mỗi ngôn ngữ lập trình (C++, Visual Basic, J++, Fox Pro) sẽ có riêng một IDE tương ứng. Ngoài ra, để phát triển ứng dụng ASP, ta phải sử dụng Visual Studio InterDev.

9.1. Phiên bản Visual Studio .NET 2008

Visual Studio .NET 2008 có 5 phiên bản chính thức là: Express Products (Visual Studio Express Edition), Visual Studio Standard Edition, *Visual Studio Professional Edition*, Visual Studio Tools for Office và Visual Studio Team System.

9.1.1 Visual Studio Express Edition

Đây là phiên bản đơn giản, dễ học, dễ sử dụng dùng cho những người tự học, chưa có kinh nghiệm lập trình hoặc các bạn sinh viên bước đầu làm quen với *Visual Studio .NET 2008.*

Nếu sử dụng phiên bản này, bạn cần bộ nhớ khoảng 35MB đến &70MB, miễn phí 1 năm. Hơn thế nữa, sẽ có phiên bản *Microsoft SQL Server Express* miễn

phí hoàn toàn, cung cấp các chức năng chính dùng để làm việc với cơ sở dữ liệu *Microsoft SQL Server Express* 2008 từ cửa sổ *Visual Studio .NET 2008*.

Tương tự như vậy *Visual Studio Express Edition* cung cấp 4 phiên bản *Visual Basic 2008 Express Edition, Visual C# 2008 Express Edition, Visual C++ 2008 Express Edition* và *Visual J# 2008 Express Edition* ứng với 4 ngôn ngữ chính là: *Visual Basic, C#, C++* và *J#.*

Trong trường hợp phát triển ứng dụng Web, có thể sử dụng *Visual Web Developer 2008 Express* để nhanh chóng tạo ra các trang *ASP .NET* bằng các công cụ trực quan.

9.1.2 Visual Studio Standard Edition

Trong khi phiên bản *Visual Studio Express* có tính năng đơn giản, dễ sử dụng và miễn phí 1 năm thì phiên bản *Visual Studio Standard Edition* được thiết kế toàn diện hơn với chi phí vừa phải.

Được sử dụng cho các lập trình viên chuyên nghiệp làm việc đơn lẽ với các ứng dụng chạy nhanh, tối ưu, ứng dụng đa tầng trên nền *Windows,* các ứng dụng Web hay ứng dụng chạy trên thiết bị cầm tay.

Với những đặc điểm như vậy, phiên bản này là công cụ hỗ trợ cả bốn ngôn ngữ lập trình, thường dùng cho các nhà lập trình viên làm việc ngoài giờ hay công việc không thường xuyên.

Tương tự như các phiên bản khác của bộ *Visual Studio .NET 2008, Visual Studio Standard Edition* cung cấp giao diện trực quan cho phép bạn thiết lập giao diện cho các loại ứng dụng bằng việc kéo và thả (*drag and drop*), xây dựng và triển khai ứng dụng theo mô hình khách-chủ (client-server), các công cụ để thiết kế cơ sở dữ liệu.

9.1.3 Visual Studio Professional Edition

Nếu như *Visual Studio Standard Edition* dùng cho cá nhân phát triển ứng dụng thì *Visual Studio Professional Edition* bao gồm các công cụ giao diện trực quan cho phép bạn thiết lập giao diện cho các loại ứng dụng bằng việc kéo và thả *(drag and drop).*

Visual Studio Professional Edition có thể sử dụng cho cá nhân hay nhóm lập trình nhỏ khi xây dựng và triển khai ứng dụng theo mô hình khách chủ (*client-server*), thiết kế cơ sở dữ liệu, ứng dụng đa tầng trên nền Windows, ứng dụng Web hay ứng dụng chạy trên thiết bị cầm tay.

9.1.4 Visual Studio Team System

Đây là công cụ theo hướng mở rộng và tích hợp, dùng cho các công ty phát triển phần mềm hay những nhóm lập trình viên làm việc xuyên quốc gia.

Sử dụng phiên bản *Visual Studio Team System* cho phép nhóm lập trình có thể giảm độ phức tạp, tăng tính giao tiếp và hợp tác trong quá trình phát triển phần mềm.

Visual Studio Team System còn là bộ khung (Microsoft Solutions Framework) gọi là MSF. MSF cung cấp một tập được tối ưu hóa và tính uyển chuyển cùng các quy tắc đã được tích hợp áp dụng cho từng giai đoạn khi phát triển và triển khai một phần mềm.

Tùy vào từng công đoạn của quá trình xây dựng và triển khai phần mềm, có thể sử dụng 5 bộ công cụ thuộc *Visual Studio Team System như sau: Visual Studio 2008 Team Suite,Visual Studio 2008 Team Edition for Software Architects, Visual*

Studio 2008 Team Edition for Software Developers, Visual Studio 2008 Team Edition for Software Testers, Visual Studio 2008 Team Foundation Server và Visual Studio 2008 Team Test Load Agent.

Bộ Visual Studio 2008 Team Suite

Visual Studio 2008 Team Suite là bộ công cụ tích hợp, hiệu suất cao cho phép nhóm lập trình viên giao tiếp và kết hợp tốt trong quá trình phát triển phần mềm. Với tổ chức của *Visual Studio 2008 Team Suite,* bạn có thể dự đoán trước được chất lượng và tổ chức trong quá trình phát triển ứng dụng.

Bộ Visual Studio 2008 Team Edition for Software Architects:

Team Edition for Software Architects cung cấp các công cụ trực quan dùng để xây dựng một giải pháp dùng cho việc thiết kế ứng dụng hay triển khai chúng nhanh và hiệu quả hơn.

Bộ Visual Studio 2008 Team Edition for Software Developers:

Team Edition for Software Developers cung cấp bộ công cụ phát triển ứng dụng cho phép nhóm lập trình tương tác, phối hợp và cùng chia sẻ trong chu trình phát triển ứng dụng.

Bộ Visual Studio 2008 Team Edition for Software Testers:

Team Edition for Software Testers giới thiệu tập các công cụ dùng để kiểm tra, đánh giá sản phẩm phần mềm được tích hợp với môi trường Visual Studio. Bộ công cụ này cho phép những người kiểm tra chất lượng sản phẩm phần mềm thông báo đến tác giả hay nhà quản lý những công việc liên quan.

Bộ Visual Studio 2008 Team Foundation Server:

Team Foundation Server là những gì mạnh nhất của quá trình hợp tác trong *Visual Studio Team System* .Khi kết hợp với *Visual Studio Team System, Team Foundation Server* cho phép bạn quản lý và theo dõi quá trình thực hiện của dự án.

Bộ Visual Studio 2008 Team Test Load Agent:

Team Test Load Agent tạo ra tiến trình kiểm tra bổ sung được sử dụng với *Visual Studio 2008 Team Edition for Software Testers,* cho phép bạn tổ chức và mô phỏng một hay nhiều người sử dụng để kiểm tra chất lượng sử dụng của ứng dụng.

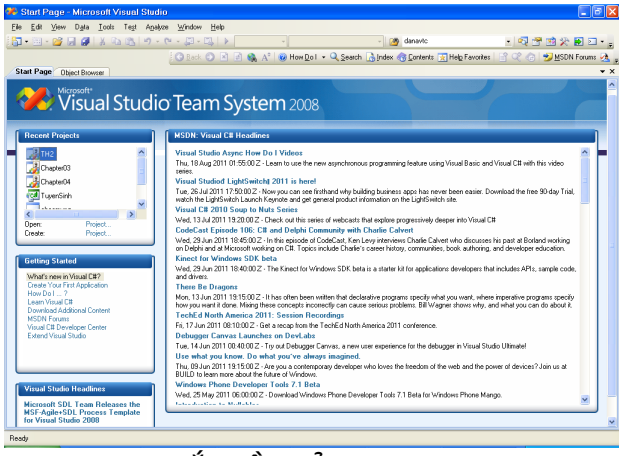
9.2 Làm việc với Visual Studio .NET 2008

Từ khi Visual studio .NET ra đời, nó là một IDE dùng chung duy nhất cho mọi ngôn ngữ lập trình và các loại ứng dụng được được tích hợp. Như vậy, ứng dụng Web Forms (ASP.NET) được xem như một phần của ngôn ngữ lập trình, bạn có thể sử dụng chung IDE với ứng dụng Windows Forms.

Chẳng hạn, bạn có thể mở dự án (Project) bằng ngôn ngữ lập trình Visual Basic.NET, rồi mở tiếp một *Project* bằng ngôn ngữ lập trình C# trong cùng một Solution.

Ngoài ra, Visual Studio.NET 2008 có sự thay đổi lớn so với Visual Studio.NET 2003 là môi trường lập trình, định dạng mã, cơ chế gỡ lỗi, xây dựng,

kiểm tra và triển khai ứng dụng, tự động hóa và trợ giúp người sử dụng. Ví dụ, trang bắt đầu của *Visual Studio.NET 2008 IDE* như hình 1-2

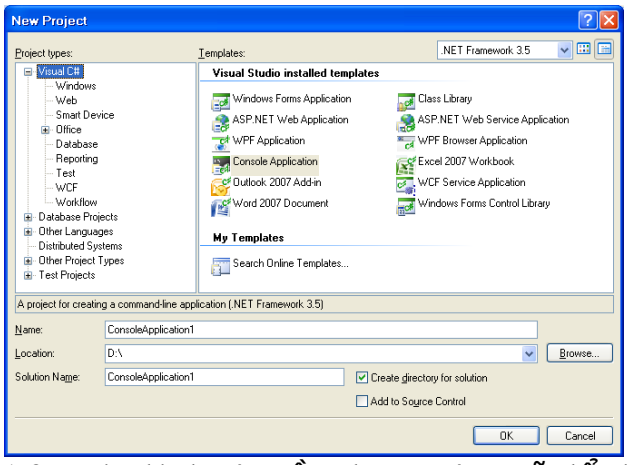


Hình 1.2 : Trang bắt đầu của *Visual Studio.NET 2008*

Sau khi cài đặt thành công Visual Studio.NET 2008, lần đầu tiên sử dụng Visual Studio.NET 2008 IDE, một cửa sổ xuất hiện yêu cầu chọn ngôn ngữ lập trình mặc định. Chẳng hạn, trong trường hợp này chúng ta chọn ngôn ngữ lập trình C# bằng cách di chuyển đến Visual C# Development Setting và nhấn mạnh Start Visual Studio.

Lưu ý, chúng ta sẽ tìm hiểu chi tiết về những công cụ, cửa sổ, cách cấu hình IDE để làm việc với ngôn ngữ lập trình C# trong những bài kế tiếp.

Sau khi chọn ngôn ngữ lập trình C# là ngôn ngữ mặc định, mỗi khi tạo mới *Project* hay *Solution, ngôn ngữ này nằm đầu tiên trong ngăn Project types như hình 1-3*, 3 ngôn ngữ lập trình còn lại là Visual basic, C++ và J# sẽ xuất hiện bên dưới phần *Other Language.*



Hình 1.3 : Màn hình yêu cầu chọn ngôn ngữ để cài đặt

Ngăn bên phải là danh sách các loại ứng dụng Windows ,bao gồm các loại như: Windows Application, Console Application, Class Library, Windows Service, Crystal Reports Application,…

Trong trường hợp muốn xây dựng ứng dụng ASP.NET, bạn có thể chọn vào trong phần tạo mới, khi đó cửa sổ sẽ xuất hiện như hình 1-5

Tương tự như trường hợp ứng dụng vWindows, ứng dụng vWebsite bao gồm các loại như: ASP.NET vWebsite, ASP.NET vWeb Service, Srystal Reports vWebsite.

Trên thực đơn (menu) của Visual studio .NET 2008, *menu* có tên là *Community* bao gồm các *menu* con như: *Ask a question, Check Question Status, Send Feedback* nhằm hỗ trợ cho bạn tìm kiếm, gởi và kiểm tra câu hỏi hay góp ý kiến về công ty *Microsoft.*

Ngoài ra, trên *menu* này còn có các *menu* con khác, chúng cho phép bạn trỏ đến địa chỉ *internet* chứa tài nguyên hay những thông tin cập nhật về Visual studio .NET 2008 nhằm hỗ trợ tối đa cho cộng đồng lập trình .NET.

Chẳng hạn, bạn chọn vào *menu* có tên *Developer center,* cửa sổ trình duyệt xuất hiện.

9.3. Các loại ứng dụng dùng C#

Microsoft Visual C# 2008 (C sharp) là ngôn ngữ lập trình thiết kế dùng để phát triển ứng dụng chạy trên *.NET Framework. C#* còn là ngôn ngữ lập trình đơn giản, mạnh, kiểu an toàn (type-safe) và hướng đối tượng (*object-oriented).*

Với nhiều đặc điểm mới,*C#* cho phép bạn xây dựng ứng dụng nhanh chóng nhưng vẫn giữ lại được sự diễn cảm và tao nhã của ngôn ngữ lập trình truyền thống C.

Mặc dù mọi ngôn ngữ lập trình trong bộ *.NET* đều sử dụng chung *.NET Framework,* nhưng mỗi ngôn ngữ vẫn có tính đặc thù riêng của nó. Sử dụng *C#* là một lựa chọn tối ưu khi bạn xây dựng loại ứng dụng như: quản lý, thương mại điện tử, ứng dụng tích hợp hệ thống,thư viện,ứng dụng dùng cho máy PDA hay điện thoại di động,…

9.3.1. Ứng dụng Windows Form

Khi xây dựng ứng dụng với giao diện người dùng chạy trên máy để bàn có cài đặt *.NET Framework,* bạn chọn *vWindows* trong phần *Project Types* rồi tiếp tục chọn vào *vWindows Application* trong phần *Templates*

9.3.2. Ứng dụng màn hình và bàn phím

Nếu ứng dụng với giao diện người dùng là bàn phím và màn hình chạy trên máy để bàn, bạn có thể chọn loại ứng dụng là Console Application trong phần Templates .Với ứng dụng loại này, người sử dụng thao tác bằng màn hình C onsole.

Tuy nhiên, bằng cách sử dụng các không gian tên của ứng dụng vWindows *Forms,* bạn cũng có thể tạo ra ứng dụng giao diện đồ họa bằng ứng dụng *Console Application.*

9.3.3. Dịch vụ hệ điều hành

Trong trường hợp ứng dụng chạy thường trú trong bộ nhớ,bạn có thể chọn loại ứng dụng là *vWindows Service* trong phần *Templates.*

Khi chọn ứng dụng này, bạn tạo ra tập tin .EXE và cài đặt chúng vàodịch vụ của hệ điều hành (Service), bạn có thể *Start, Stop hay Pause* và *Continue* như những dịch vụ của hệ điều hành đang tồn tại.Chú ý, ứng dụng dịch vụ hệ điều hành thì không cần giao diện, thay vào đó bạn sử dụng tiện ích *Service* của hệ điều hành.

9.3.4. Thư viện

Khi cần xây dựng thư viện dùng chung hay *COM+(*triển khai tầng *Business Logic),* bạn chọn vào *Class Library,* sau khi kết thúc khai báo,nếu biên dịch thành công thì ứng dụng này sẽ tạo ra tập tin .DLL.

Ví dụ, bạn muốn xây dựng thư viện bao gồm các lớp làm việc với cơ sở dữ liệu *SQL Server,* sau đó bạn sử dụng thư viện như những *Project* khác nhau, ứng với mục đích này bạn tạo mới *Project* loại *Class Library.*

9.3.5. Điều khiển do người sử dụng định nghĩa

Ngoài các điều khiển (*Control)*từ các lớp của .NET cung cấp, người sử dụng có thể kết hợp những điều khiển này thành một điều khiển tùy ý (*CustomControl)* phục vụ cho một yêu cầu cụ thể nào đó.

Đối với ứng dụng *vWindows Forms*, bạn có thể sử dụng loại *Project* là *vWindows Control Library.* Trong trường hợp làm việc với ứng dụng *ASP.NET* loại *Project* bạn dùng là *vWeb Control Library.*

Cả hai loại *Project* này đều biên dịch thành tập tin .DLL, bạn có thể thêm chúng vào công cụ (*Tool Box)* như những điều khiển của *.NET*

9.3.6. Ứng dụng báo cáo

Nếu có nhu cầu xây dựng ứng dụng báo cáo (*Report)* bằng *Crystal Report,* bạn chọn loại *Project* là *Crystal Report Applications.*Tuy nhiên, thông thường *Report* là một phần của ứng dụng nên bạn sử dụng *Crystal Report* như những đối tượng của *Project.*

9.3.7. Ứng dụng SQL Server

Để khai báo bảng dữ liệu (*Table),* bảng ảo (*View)*, thủ tục nội tại, (*Store Procedure),* hàm *(Funtion),…*bạn vào ngăn *Database* rồi chọn *Project* với loại *SQL Server Project.* Ứng dụng này cho phép bạn thiết kế cơ sở dữ liệu *SQL Server* từ *Visual Studio.NET 2008* thay vì từ trình *SQL Server Enterprise.*

Lưu ý, tương tự như trong ứng dụng *Report,* bạn có thể thêm cơ sở dữ liệu vào *Project* như một phần của ứng dụng thay vì tạo riêng*Project* về cơ sở dữ liệu.

9.3.8. Ứng dụng PDA và Mobile

Nếu bạn muốn xây dựng ứng dụng *.NET* cho thiết bị cầm tay như điện thoại di động (*Mobile)* hay máy kỹ thuật số hệ thống cá nhân (*PDA)* thì chọn vào *Smart Device.*

9.3.9. Ứng dụng đóng gói và triển khai

Sau khi kết thúc công đoạn xây dựng ứng dụng, bạn có thể đóng goi ứng dụng đó và triển khai trên máy khác. Để đóng gói ứng dụng, bạn vào ngăn *Other Project Types* rồi chọn loại *Project* là *Setup and Deployment.*

9.3.10. Tạo một Solution

Solution được xem như một *(Container)* dùng để quản lý nhiều *Project* trên *Visual Studio.NET 2008.*Khi tạo *Project* đầu tiên chưa tồn tại *Solution,* lập tức *Solution* được tạo ta mặc định. Trong trường hợp *Solution* đã tồn tại thì chọn *Solution* để thêm *Project* vào *Solution* đó.

Ngoài ra, bạn có thể tạo mới *Solution* trước khi thêm các *Project* khác bằng cách vào *Other Project Types* rồi chọn *Visual Studio Solutions.*

9.4. Cấu trúc chương trình C#

9.4.1. Cấu trúc chương trình

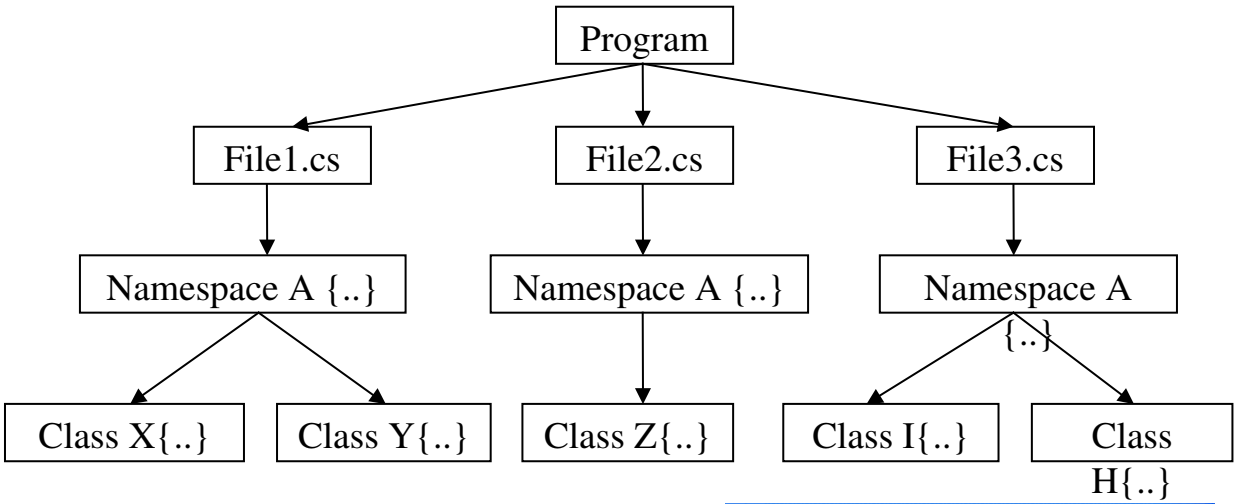
- ***Cấu trúc chương trình theo Windows Application Form***

```
//Vùng bắt đầu khai báo sử dụng không gian tên
using System;
using System.Collections.Generic;
using System.Linq;
using System.Windows.Forms;
//Vùng bắt đầu khai báo sử dụng không gian tên
//Khai báo không gian tên của ứng dụng
namespace TH2
{
        //Vùng bắt đầu khai báo tên các Class
   static class Program
   {
        //Vùng bắt đầu khai báo tên các phương thức trong lớp
      static void Main()
      {
                  //Vùng khai báo lệnh
      }
   }
}
```

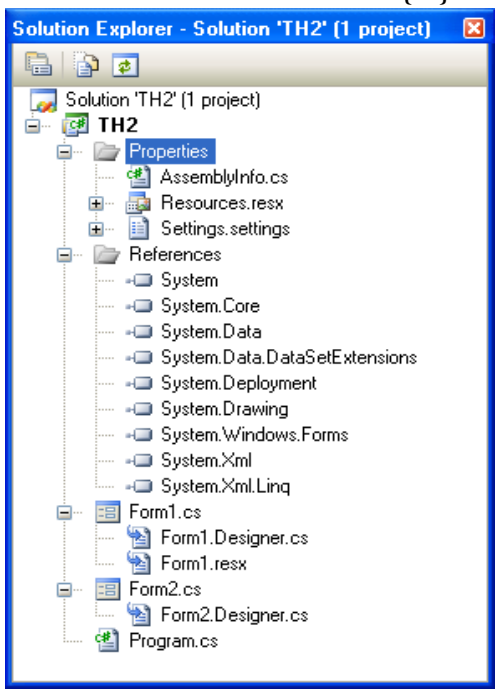
- ***Cấu trúc chương trình theo Console Command***

```
using System
Namespace MyNameSpace
{
 class HelloWorld
 {
  //Điểm bắt đầu của ứng dụng theo kiểu C
  static void Main(){
    Main(System.Environment.GetCommandLineArgs());
  }
  static void Main(string[] args){
    System.Console.WriteLine("Hello World")
  }
 }
}
```

- Cấu trúc của 1 chương trình C#



9.4.2 Tổ chức cây Project



9.4.2.1 Nút Properties

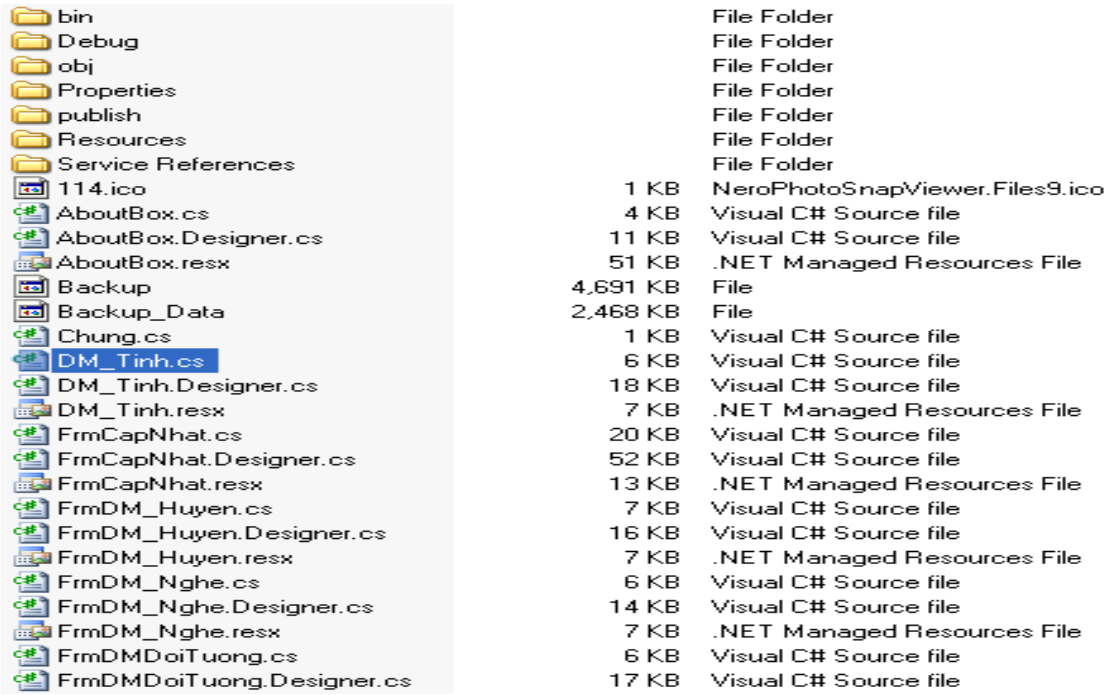
9.4.2.2 Nút References

9.4.2.3 Nút đối tượng có giao tiếp

9.4.2.4 Nút đối tượng không có giao tiếp

9.5. Cấu trúc thư mục của ứng dụng

- Các File của 1 chương trình C#



bin		File Folder
Debug		File Folder
obj		File Folder
Properties		File Folder
publish		File Folder
Resources		File Folder
Service References		File Folder
114.ico	1 KB	NeroPhotoSnapViewer.Files9.ico
AboutBox.cs	4 KB	Visual C# Source file
AboutBox.Designer.cs	11 KB	Visual C# Source file
AboutBox.resx	51 KB	.NET Managed Resources File
Backup	4,691 KB	File
Backup_Data	2,468 KB	File
Chung.cs	1 KB	Visual C# Source file
DM_Tinh.cs	6 KB	Visual C# Source file
DM_Tinh.Designer.cs	18 KB	Visual C# Source file
DM_Tinh.resx	7 KB	.NET Managed Resources File
FrmCapNhat.cs	20 KB	Visual C# Source file
FrmCapNhat.Designer.cs	52 KB	Visual C# Source file
FrmCapNhat.resx	13 KB	.NET Managed Resources File
FrmDM_Huyen.cs	7 KB	Visual C# Source file
FrmDM_Huyen.Designer.cs	16 KB	Visual C# Source file
FrmDM_Huyen.resx	7 KB	.NET Managed Resources File
FrmDM_Nghe.cs	6 KB	Visual C# Source file
FrmDM_Nghe.Designer.cs	14 KB	Visual C# Source file
FrmDM_Nghe.resx	7 KB	.NET Managed Resources File
FrmDMDoiTuong.cs	6 KB	Visual C# Source file
FrmDMDoiTuong.Designer.cs	17 KB	Visual C# Source file

BÀI THỰC HÀNH CỦA HỌC VIÊN

Kỹ năng 1 : Trình bày các ưu điểm và nhược điểm của ngôn ngữ lập trình C#.

Làm việc theo nhóm, tra cứu trên Internet, các phương tiện khác và trình bày báo cáo (tối đa khoảng 2 trang).

Kỹ năng 2 : Cài đặt Visual Studio 2008.

Cài đặt Visual Studio 2008 từ đĩa DVD

Kỹ năng 3 : Tìm các thông tin liên quan về C# : tính năng của phần mềm, các phiên bản

Làm việc theo nhóm.

Tìm hiểu các thông tin về C# và số lượng người dùng C# hiện nay.

Các tính năng vượt trội của C# so với các ngôn ngữ khác.

Các phiên bản C# hiện nay đã được Microsoft công bố.

Kể tên một số ứng dụng đã dùng ngôn ngữ lập trình C# mà các bạn biết.

Kỹ năng 4 : Tìm hiểu về các chương trình C# mẫu

Liệt kê và phân biệt được các thành phần trong thư mục của ứng dụng.

Cách tổ chức của Cây Project, khám phá và tìm hiểu các nút.

CÂU HỎI ÔN TẬP

Câu 1: Nêu các thành phần chính của .NET Framework.

Câu 2: Trình bày sơ đồ môi trường .NET Framework.

Câu 3: Liệt kê những đặc điểm chính của .NET Framework.

Câu 4: Liệt kê các ứng dụng dùng C#.

Câu 5: Trình bày cấu trúc chương trình C#.

BÀI 2 : VẤN ĐỀ I/O TRONG .NET

Mã bài MĐ35.1

Mục tiêu của bài:

- Trình bày được không gian tên I/O áp dụng cho các mạng truyền dữ liệu.
- Liệt kê các thành phần của không tên System.IO.Streams có sử dụng liên quan đến mạng.
- Mô tả được đối tượng Streams.
- Sử dụng không tên System.IO để ghi và đọc các dữ liệu lên các vùng lưu trữ.
- Sử dụng không tên System.IO để chuyển tải dữ liệu, truy vấn dữ liệu trên mạng.
- Thực hiện các thao tác an toàn với máy tính.

1. Giới thiệu về không gian tên IO

Chương này đặt nền tảng cho hầu như tất cả các ví dụ mạng chứa trong cuốn sách này. Nếu không có một kiến thức để làm việc .NET về xử lý I/O có thể rất khó khăn để thích ứng với các ví dụ mã trong cuốn sách này với nhu cầu riêng của bạn.

I/O áp dụng mạng dữ liệu chuyển giao, cũng như tiết kiệm và tải đĩa cứng của bạn máy tính của Sau đó chương sẽ mô tả làm thế nào để thực hiện chuyển mạng; tuy nhiên, chương này sẽ được quan tâm với các cơ bản I/O hoạt động được phổ biến đến cả hai loại chuyển. Nửa đầu của chương này sẽ chứng minh làm thế nào để đọc và ghi vào đĩa cứng, bằng cách sử dụng dòng NET. Phần thứ hai của chương này phát triển khái niệm dòng bằng cách chứng minh làm thế nào để chuyển đổi đối tượng phức tạp, chẳng hạn như các truy vấn cơ sở dữ liệu vào một định dạng mà có thể được ghi vào một dòng NET.

2. Streams

2.1. Mã hóa dữ liệu

Streams (Luồng dữ liệu) là sự trừu tượng hóa phương thức truyền dữ liệu (đọc, ghi) , đối với mỗi loại thiết bị khác nhau và trên các môi trường khác nhau thì sử dụng các loại stream khác nhau để đọc ghi dữ liệu.

Có hai loại Stream quan trọng: **NetworkStream** và **FileStream**

Network Stream: Được sử dụng để đọc dữ liệu trên mạng

FileStream: Được dùng để đọc dữ liệu cục bộ (ví dụ như tập tin trên đĩa)

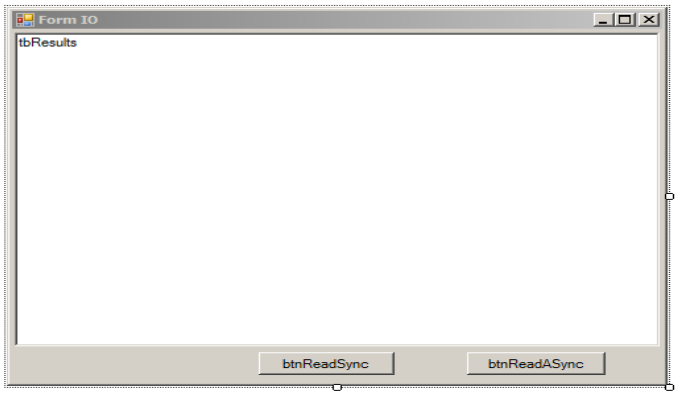
Có hai cách để sử dụng Stream là "dùng stream đồng bộ" và dùng "stream bất đồng bộ". Trong cách dùng đồng bộ thì các luồn (thread) tương ứng của

chương trình sẽ tạm ngưng đến khi ứng dụng hoàn thành việc đọc dữ liệu hoặc có lỗi xảy ra. Trong cách dùng không đồng bộ thì, các thread sẽ vẫn chạy song song với quá trình truyền dữ liệu, và khi quá trình truyền dữ liệu hoàn tất hay có lỗi xảy ra thì đều có trạng thái tương ứng được trả về.

Ví dụ sử dụng Stream để đọc tập tin:

Khởi tạo một đồ án .NET mới và thêm vào:

- Một Form đặt tên là **FormIO**
- Một File **Open Dialog Control** với tên là **openFileDialog**
- Một TextBox với tên là **tbResults**, chọn thuộc tính **Multiline = true**
- Hai Button với tên gọi là **btnReadAsync** và **btnReadSync** (**btnReadAsync** sẽ thực hiện chức năng đọc tập tin theo cơ chế không đồng bộ, **btnReadSync** sẽ thực hiện chức năng đọc tập tin theo cơ chế đồng bộ)



Khai báo thêm **Namespace** chứa các lớp **Stream**

```
using System.IO;
```

Khai báo các biến sử dụng trong chương trình:

```
FileStream fs; // Khai báo luồng đọc ghi tập tin
byte[] fileContents; // Mảng byte để lưu trữ nội dung tập tin khi đọc
AsyncCallback callback; //
```

Khai báo phương thức để cập nhật cho TextBox

```
// Sử dụng phương thức này để cập nhật cho các đối tượng trong trường
// hợp bị tranh chấp Thread ! (Lỗi : Cross-thread operation not valid !)
static public void CapNhatControl(Control control, MethodInvoker code)
{
    if (control.InvokeRequired)
    {
        control.BeginInvoke(code);
        return;
    }
    control.Invoke(code);
}
```

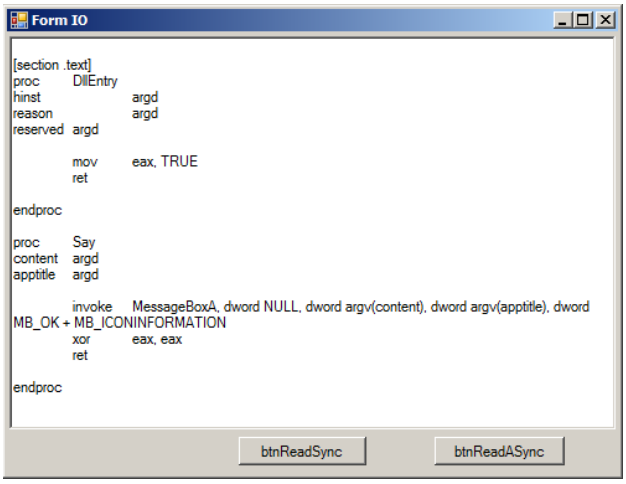
Viết code xử lý cho sự kiện click của Button **readAsync**

```
private void btnReadAsync_Click(object sender, EventArgs e)
{
    openFileDialog.ShowDialog(); // Gọi phương thức ShowDialog của đối
    // tượng OpenFileDialog để chọn tập tin cần mở
    // Tạo mới đối tượng AsyncCallBack để sử dụng cho việc đọc ghi bất
    // đồng bộ và truyền cho nó phương thức fs_StateChanged để
    // nó biết trạng thái khi đọc tập tin
    callback = new AsyncCallback(fs_StateChanged);
    // Tạo mới đối tượng fs của lớp FileStream để đọc tập tin
    fs = new FileStream(
    openFileDialog.FileName, // Tên tập tin được chọn
    FileMode.Open, // Chế độ mở tập tin
    FileAccess.Read, // Chế độ đọc
    FileShare.Read, // Chế độ đọc
    4096, // Kích thước khối dữ liệu mỗi lần đọc
    true); // Dùng cơ chế bất đồng bộ
    // Tạo mới mảng fileContents có kích thước bằng với kích thước của tập tin
    // cần đọc
    fileContents = new Byte[fs.Length];
    // Bắt đầu quá trình đọc tập tin
    fs.BeginRead(fileContents, 0, (int)fs.Length, callback, null);
} //
```

Viết code xử lý cho phương thức **fs_StateChanged**

```
// Phương thức này được gọi trong quá trình đọc tập tin
private void fs_StateChanged(IAsyncResult asyncResult)
{
    if (asyncResult.IsCompleted) // Nếu đã đọc hết dữ liệu
    {
        // Chuyển dãy byte dữ liệu đọc được sang dạng utf8 để hiển
        // thị tiếng Việt
        string s = Encoding.UTF8.GetString(fileContents);
        // Cập nhật vào tbResults bằng phương thức CapNhatControl
        CapNhatControl(tbResults, delegate {
            tbResults.Text = s;
        });
        // Đóng tập tin
        fs.Close();
    }
}
```

Kết quả :



Viết code xử lý cho nút lệnh **readSync** để đọc tập tin theo cơ chế đồng bộ:

Vì đọc dữ liệu đồng bộ nên nó sẽ làm dừng toàn bộ hoạt động của biểu mẫu trong khi đọc, vì vậy để khắc phục tính trạng này, có thể sử dụng Thread riêng để đọc. Khi sử dụng Thread thì phải khai báo thêm namespace Threading :

```
using System.Threading;
```

Phần code cho sự kiện click của btnReadSync

```
private void btnReadSync_Click(object sender, EventArgs e)
{
    openFileDialog.ShowDialog(); // Hiện hộp thoại mở tập tin
    // Tạo mới thead nhận tham số truyền vào là phương thức syncRead
    Thread thdSyncRead = new Thread(new ThreadStart(syncRead));
    // Cho chạy thread
    thdSyncRead.Start();
}
```

Phần code cho phương thức syncRead

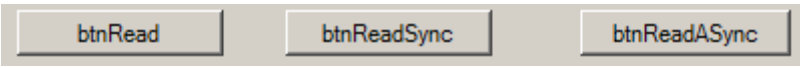
```
public void syncRead()
{
    try
    {
        fs = new FileStream(
            openFileDialog.FileName,
            FileMode.OpenOrCreate
        );
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
        return;
    }
    // Di chuyển đến đầu tập tin cần đọc
    fs.Seek(0, SeekOrigin.Begin);
    // Tạo mới mảng dữ liệu có kích thước bằng với kích thước của tập tin
    fileContents = new byte[fs.Length];
    // Đọc tập tin
    fs.Read(fileContents, 0, (int) fs.Length);
    string s = Encoding.UTF8.GetString(fileContents);
    CapNhatControl(tbResults, delegate {
        tbResults.Text = s;
    });
    fs.Close();
}
```

2.2 Sử dụng StreamReader để đọc tập tin Text

Tập tin được chia ra làm 2 loại, tập tin văn bản (TEXT) và tập tin nhị phân (Binary). Để đọc tập tin văn bản ta sử dụng lớp StreamReader:

Ví dụ:

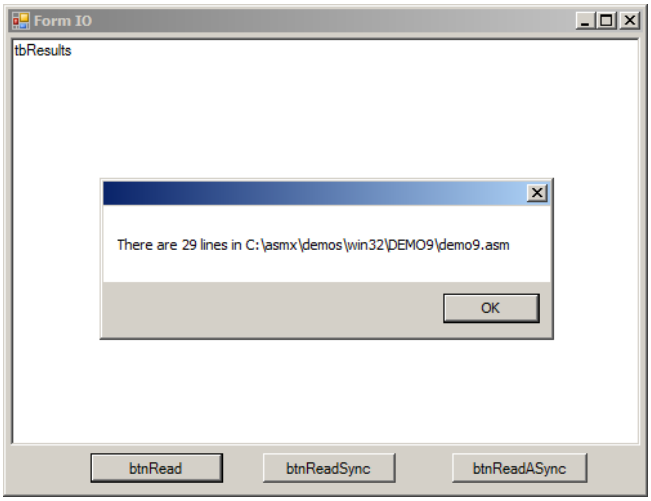
Thêm một Button vào FormIO đặt tên là **btnRead**



Viết code cho sự kiện click của btnRead:

```
private void btnRead_Click(object sender, EventArgs e)
{
    openFileDialog = new OpenFileDialog();
    openFileDialog.ShowDialog();

    fs = new FileStream(openFileDialog.FileName, FileMode.OpenOrCreate);
    // Khởi tạo đối tượng StreamReader để đọc tập tin Text
    StreamReader sr = new StreamReader(fs);
    int lineCount = 0;
    while (sr.ReadLine() != null) // Đọc từng dòng
    {
        lineCount++;
    }
    fs.Close();
    MessageBox.Show("There are " + lineCount +
        " lines in " + openFileDialog.FileName);
}
```

Kết quả:



Ví dụ về ghi tập tin sử dụng BinaryWriter để ghi tập tin nhị phân :

Thêm một Button và đặt tên là btnWrite :

Viết code sử lý sự kiện Click của btnWrite

```
private void btnWrite_Click(object sender, EventArgs e)
{
    // Tạo mới đối tượng Save Dialog
    SaveFileDialog sfd = new SaveFileDialog();
    sfd.ShowDialog(); // Hiển thị hộp thoại
    // Tạo mới đối tượng File Stream để mở tập tin theo chế độ tạo mới
    fs = new FileStream(sfd.FileName, FileMode.CreateNew);
    // Tạo mới đối tượng BinaryWriter để ghi tập tin nhị phân
    BinaryWriter binaryWriter = new BinaryWriter(fs);
    // Khai báo mảng 1000 phần tử
    int[] myArray = new int[1000];
    // Gán giá trị vào mảng này
    for (int i = 0; i < 1000; i++)
    {
        myArray[i] = i;
        binaryWriter.Write(myArray[i]); // ghi phần tử này vào tập tin
    }
    binaryWriter.Close();
}
```

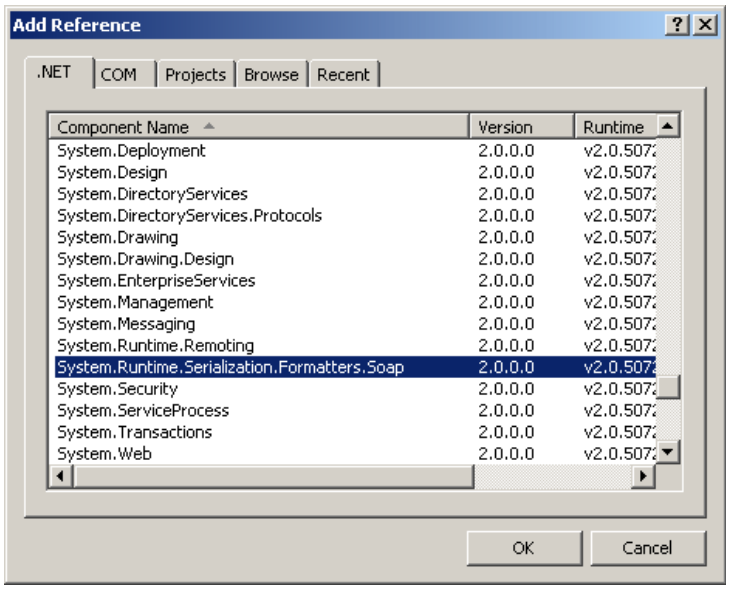
Serialization

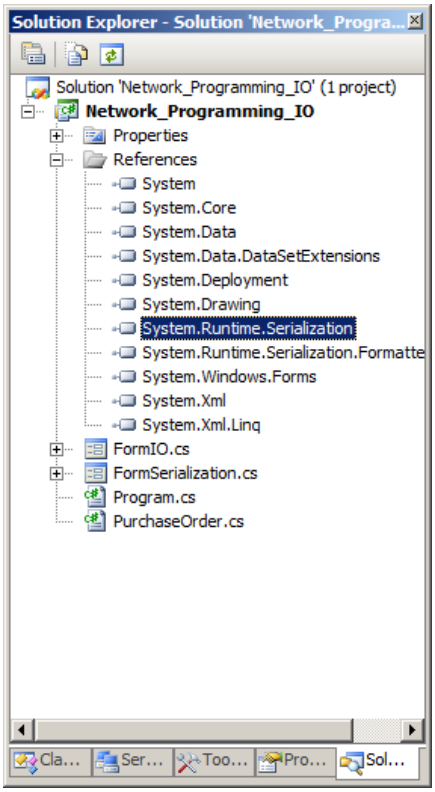
Quy trình Serialization là quy trình "đóng gói" các đối tượng thành dạng mà Stream trong .NET có thể hiểu được, nó giúp cho việc truyền tải dữ liệu sử dụng Stream được dễ dàng hơn, như việc truyền đối tượng qua mạng hoặc ghi xuống đĩa. Quy trình Deserialization là quy trình ngược lại, chuyển dữ liệu thành đối tượng.

Để sử dụng Serialization cần tham chiếu đến namespace:

```
using System.Runtime.Serialization.Formatters;
```

Và Add References cho nó :





Ví dụ:

Tạo mới một biểu mẫu và đặt tên là FormSerialization, thêm mới một Class và đặt tên là PurchaseOrder

Tạo mới cấu trúc dữ liệu mẫu sử dụng để Demo quá trình Serialization:

```
public enum PurchaseOrderStates
{
    ISSUED,
    DELIVERED,
    INVOICED,
    PAID
}

[Serializable()]
public class Company
{
    public string name;
    public string address;
    public string phone;
}

[Serializable()]
public class LineItem // Mặt hàng
```

```
{
    public string description; // Mô tả
    public int quantity; // Số lượng
    public double cost; // Giá
}

[Serializable()]
public class PurchaseOrder // Đặt hàng
{
    private PurchaseOrderStates _purchaseOrderStatus;
    private DateTime _issuanceDate;
    private DateTime _deliveryDate;
    private DateTime _invoiceDate;
    private DateTime _paymentDate;
    public Company buyer;
    public Company vendor;
    public string reference;
    public LineItem[] items;

    public PurchaseOrder()
    {
        _purchaseOrderStatus = PurchaseOrderStates.ISSUED;
        _issuanceDate = DateTime.Now;
    }

    public void recordDelivery()
    {
        if (_purchaseOrderStatus == PurchaseOrderStates.ISSUED)
        {
            _purchaseOrderStatus = PurchaseOrderStates.DELIVERED;
            _deliveryDate = DateTime.Now;
        }
    }
}
```

Trong biểu mẫu FormSerialization thêm một Button và đặt tên là btnSoap, thêm tham chiếu đến namespace.

```
using System.Runtime.Serialization.Formatters.Soap;
```

Viết code cho Button này như sau:

```
private void btnSoap_Click(object sender, EventArgs e)
{
    Company vendor = new Company();
    Company buyer = new Company();
    LineItem goods = new LineItem();
    PurchaseOrder po = new PurchaseOrder();

    vendor.name = "Acme Inc.";
    buyer.name = "Wiley E. Coyote";
    goods.description = "anti-RoadRunner cannon";
    goods.quantity = 1;
    goods.cost = 599.99;
    po.items = new LineItem[1];
    po.items[0] = goods;
    po.buyer = buyer;
    po.vendor = vendor;
    SoapFormatter sf = new SoapFormatter();
    FileStream fs = File.Create("..\\po.xml");
    sf.Serialize(fs, po);
    fs.Close();
}
```

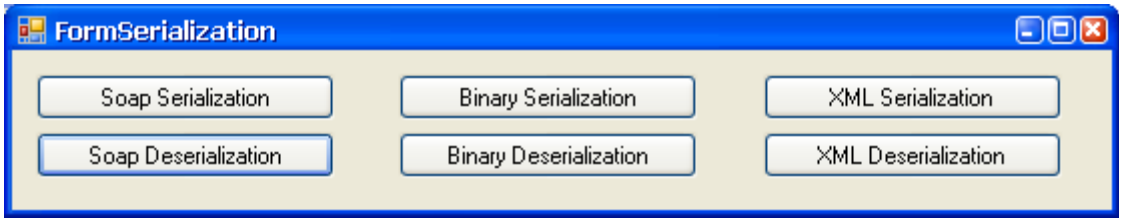
Đối với các dạng Serialization khác chỉ cần thay đối tượng Formater tương ứng, hoặc đối với Xml thì có thể sử dụng XmlSerializer để thực hiện thao tác Serialize và Deserialize, nhưng chú ý là XmlSerialize yêu cầu phải biết kiểu dữ liệu của đối tượng trước khi Serialize hoặc Deserialize, nên phải dùng phương thức GetType() của đối tượng để lấy kiểu dữ liệu trong lúc khởi tạo đối tượng XmlSerialize.

```
// Sử dụng XmlSerializer
XmlSerializer xmlSerializer = new XmlSerializer(po.GetType());
FileStream fs = File.Create("..\\po.xml");
xmlSerializer.Serialize(fs, po);
```

BÀI THỰC HÀNH CỦA HỌC VIÊN

Viết chương trình đọc và ghi các loại File

Bước 1 : Thiết kế giao diện



Bước 2 : Viết mã cho các nút lệnh

```
private void btnSoap_Click(object sender, EventArgs e)
    {
      Company vendor = new Company();
      Company buyer = new Company();
```

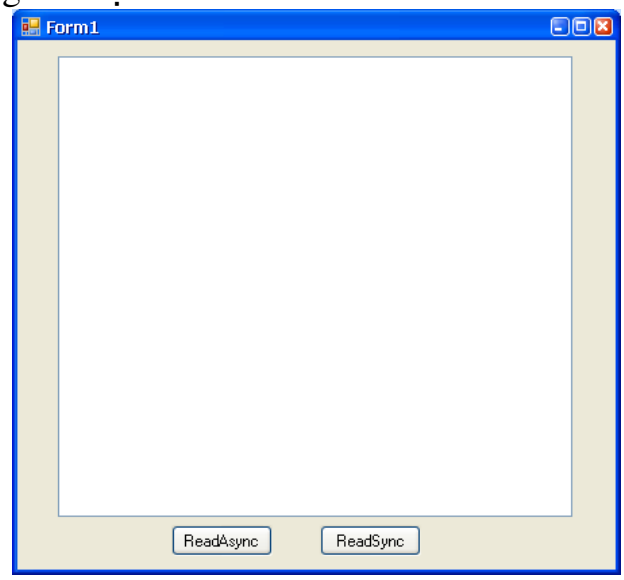
```
    LineItem goods = new LineItem();
    PurchaseOrder po = new PurchaseOrder();

    vendor.name = "Acme Inc.";
    buyer.name = "Wiley E. Coyote";
    goods.description = "anti-RoadRunner cannon";
    goods.quantity = 1;
    goods.cost = 599.99;
    po.items = new LineItem[1];
    po.items[0] = goods;
    po.buyer = buyer;
    po.vendor = vendor;
    // Sử dụng Soap Formatter
    SoapFormatter sf = new SoapFormatter();
    FileStream fs = File.Create("..\\po.xml");
    sf.Serialize(fs, po);
    fs.Close();
}

private void btnDeSoap_Click(object sender, EventArgs e)
{
    SoapFormatter sf = new SoapFormatter();
    FileStream fs = File.OpenRead("..\\po.xml");
    PurchaseOrder po = (PurchaseOrder)sf.Deserialize(fs);
    fs.Close();
    MessageBox.Show("Customer is " + po.buyer.name +
    "\nVendor is " + po.vendor.name + ", phone is " +
    po.vendor.phone +
    "\nItem is " + po.items[0].description +
    " has quantity " +
    po.items[0].quantity.ToString() + ", has cost " +
    po.items[0].cost.ToString(), "Soap Deserialization");
}

private void btnBinary_Click(object sender, EventArgs e)
{
    Company vendor = new Company();
    Company buyer = new Company();
    LineItem goods = new LineItem();
    PurchaseOrder po = new PurchaseOrder();

    vendor.name = "Acme Inc.";
    buyer.name = "Wiley E. Coyote";
    goods.description = "anti-RoadRunner cannon";
    goods.quantity = 1;
    goods.cost = 599.99;
    po.items = new LineItem[1];
    po.items[0] = goods;
    po.buyer = buyer;
    po.vendor = vendor;
```

```
    // Sử dụng Binary Formatter
    BinaryFormatter bf = new BinaryFormatter();
    FileStream fs = File.Create("..\\po.xml");
    bf.Serialize(fs, po);
    fs.Close();
}

private void btnDeBinary_Click(object sender, EventArgs e)
{
    BinaryFormatter bf = new BinaryFormatter();
    FileStream fs = File.OpenRead("..\\po.xml");
    PurchaseOrder po = (PurchaseOrder)bf.Deserialize(fs);
    fs.Close();
    MessageBox.Show("Customer is " + po.buyer.name +
    "\nVendor is " + po.vendor.name + ", phone is " +
    po.vendor.phone +
    "\nItem is " + po.items[0].description +
    " has quantity " +
    po.items[0].quantity.ToString() + ", has cost " +
    po.items[0].cost.ToString(), "Binary Deserialization");
}

private void btnXML_Click(object sender, EventArgs e)
{
    Company vendor = new Company();
    Company buyer = new Company();
    LineItem goods = new LineItem();
    PurchaseOrder po = new PurchaseOrder();

    vendor.name = "Acme Inc.";
    buyer.name = "Wiley E. Coyote";
    goods.description = "anti-RoadRunner cannon";
    goods.quantity = 1;
    goods.cost = 599.99;
    po.items = new LineItem[1];
    po.items[0] = goods;
    po.buyer = buyer;
    po.vendor = vendor;

    // Sử dụng XmlSerializer
    XmlSerializer xmlSerializer = new XmlSerializer(po.GetType());
    FileStream fs = File.Create("..\\po.xml");
    xmlSerializer.Serialize(fs, po);
    fs.Close();
}

private void btnDeXML_Click(object sender, EventArgs e)
{
    PurchaseOrder po = new PurchaseOrder();
```

```
        XmlSerializer xmlSerializer = new XmlSerializer(po.GetType());
        FileStream fs = File.OpenRead("..\\po.xml");
        po = (PurchaseOrder)xmlSerializer.Deserialize(fs);
        fs.Close();
        MessageBox.Show("Customer is " + po.buyer.name +
        "\nVendor is " + po.vendor.name + ", phone is " +
        po.vendor.phone +
        "\nItem is " + po.items[0].description +
        " has quantity " +
        po.items[0].quantity.ToString() + ", has cost " +
        po.items[0].cost.ToString(), "Xml Deserialization");
    }
```

Bài 2 : Xử lý File đồng bộ và không đồng bộ

Bước 1 : Thiết kế giao diện



Bước 2: Viết mã cho các nút lệnh

```
FileStream fs;
    byte[] fileContents;
    AsyncCallback callback;
    delegate void InfoMessageDel(String info);

    public Form1()
    {
        InitializeComponent();
    }

    private void btnReadAsync_Click(object sender, EventArgs e)
    {
        openFileDialog.ShowDialog();
        callback = new AsyncCallback(fs_StateChanged);
        fs = new FileStream(openFileDialog.FileName, FileMode.Open,
        FileAccess.Read, FileShare.Read, 4096, true);
        fileContents = new Byte[fs.Length];
        fs.BeginRead(fileContents, 0, (int)fs.Length, callback, null);
    }
```

```
private void fs_StateChanged(IAsyncResult asyncResult)
{
   if (asyncResult.IsCompleted)
   {
      string s = Encoding.UTF8.GetString(fileContents);

      InfoMessage(s);
      //tbResults.Text = Encoding.UTF8.GetString(fileContents);

      fs.Close();
   }
}

public void InfoMessage(String info)
{
   if (tbResults.InvokeRequired)
   {
      InfoMessageDel method = new InfoMessageDel(InfoMessage);
      tbResults.Invoke(method, new object[] { info });
      return;
   }

   tbResults.Text = info;
}

private void btnReadSync_Click(object sender, EventArgs e)
{
   openFileDialog.ShowDialog();
   Thread thdSyncRead = new Thread(new ThreadStart(syncRead));
   thdSyncRead.Start();
}

public void syncRead()
{
   //OpenFileDialog ofd = new OpenFileDialog();
   //ofd.ShowDialog();
   //openFileDialog.ShowDialog();
   FileStream fs;
   try
   {
      fs = new FileStream(openFileDialog.FileName, FileMode.OpenOrCreate);
   }
   catch (Exception ex)
   {
      MessageBox.Show(ex.Message);
      return;
   }
   fs.Seek(0, SeekOrigin.Begin);
   byte[] fileContents = new byte[fs.Length];
```

```
        fs.Read(fileContents, 0, (int)fs.Length);
        //tbResults.Text = Encoding.UTF8.GetString(fileContents);
        string s = Encoding.UTF8.GetString(fileContents);

        InfoMessage(s);
        fs.Close();
    }
```

CÂU HỎI ÔN TẬP

Câu 1 : Nêu khái niệm về Stream.

Câu 3 : Nêu phương thức truyền đồng bộ và bất đồng bộ của Stream

Câu 2 : Nêu phương thức canRead(); canSeek(); canwrite();

BÀI TẬP

Viết chương trình thao tác với File giống chương trình quản lý File của Norton Command (NC).

Viết chương trình soạn thảo File giống (Notepad) đối với File dạng Text.

BÀI 3 : LÀM VIỆC VỚI SOCKETS

Mã bài: MĐ35-03

Mục tiêu của bài:

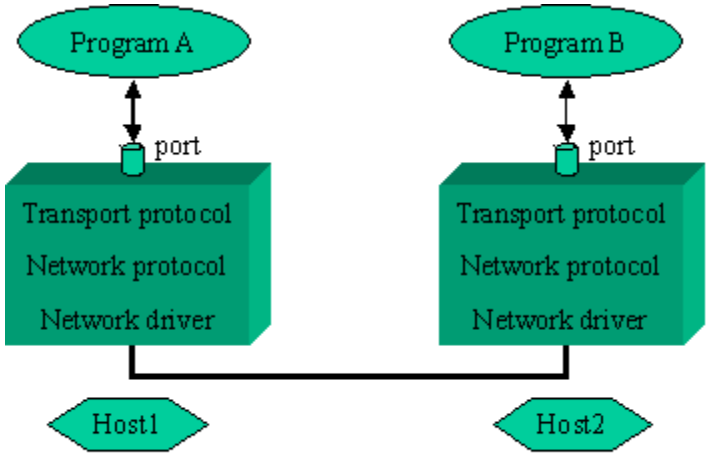
Mô tả mô hình client/server;
Mô tả lớp Socket;
Trình bảy chế độ làm việc của socket ở Clien và Server;
Viết các ứng dụng trên mạng dùng Socket.
Thực hiện các thao tác an toàn với máy tính.

1. Giới thiệu về socket trong lập trình mạng

1.1. Định nghĩa

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, Linux và được sử dụng trong nhiều ngôn ngữ lập trình khác nhau: như C, C++, Java, C# . . .

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



Hình 3.1. Mô hình Socket

Có hai kiểu socket:

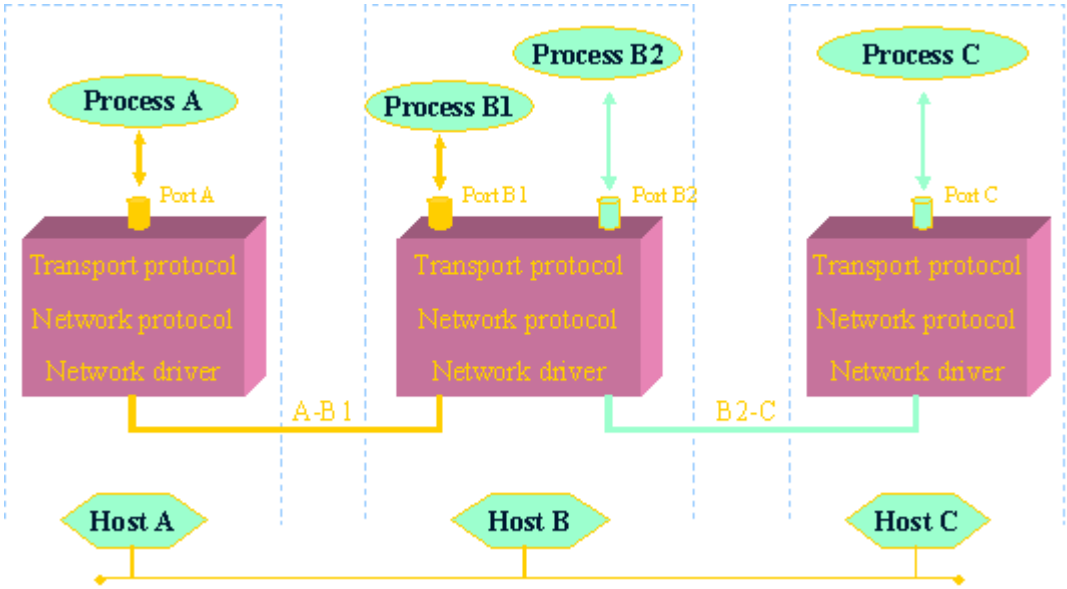
1. Socket kiểu AF_UNIX chỉ cho phép giao tiếp giữa các quá trình trong cùng một máy tính

2.Socket kiểu AF_INET cho phép giao tiếp giữa các quá trình trên những máy tính khác nhau trên mạng.

1.2. Số hiệu cổng (Port Number) 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 được yêu cầu tạo ra một socket.

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.



Hình 3.2. Cổng trong Socket

Trong hình trên, địa chỉ của quá trình B1 được xác định bằng 2 thông tin: (Host B, Port B1):

Địa chỉ máy tính có thể là địa chỉ IP dạng 203.162.36.149 hay là địa chỉ theo dạng tên miền như www.cit.ctu.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 bits). Trong đó, các cổng từ 1 đến 1023 được gọi là cổng hệ thống được dành riêng cho các quá trình của hệ thống.

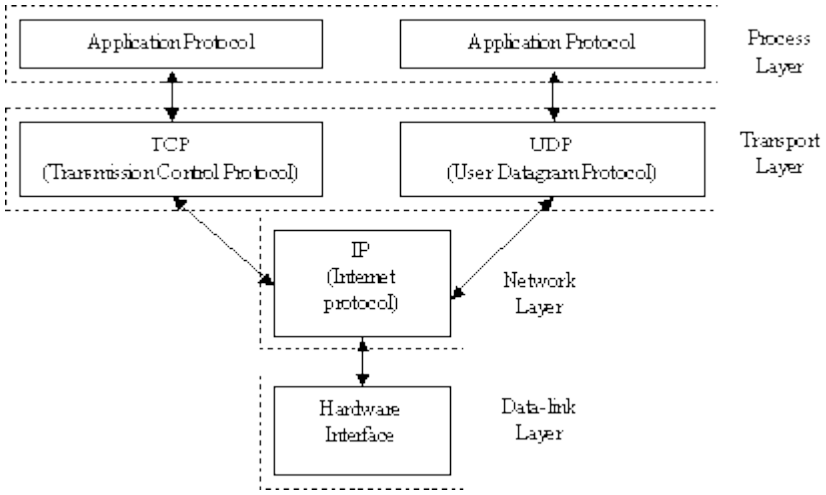
Các cổng mặc định của 1 số dịch vụ mạng thông dụng:

Số hiệu cổng	Quá trình hệ thống
7	Dịch vụ Echo
21	Dịch vụ FTP

23	Dịch vụ Telnet
25	Dịch vụ E-mail (SMTP)
80	Dịch vụ Web (HTTP)
110	Dịch vụ E-mail (POP)

1.3. Các chế độ giao tiếp

Xét kiến trúc của hệ thống mạng TCP/IP



Hình 3.3 Bộ giao thức TCP/IP

Tầng vận chuyển giúp chuyển tiếp các thông điệp giữa các chương trình ứng dụng với nhau. Nó có thể hoạt động theo hai chế độ:

- Giao tiếp có nối kết, nếu sử dụng giao thức TCP
- Hoặc giao tiếp không nối kết, nếu sử dụng giao thức UDP

Socket là giao diện giữa chương trình ứng dụng với tầng vận chuyển. Nó cho phép ta chọn giao thức sử dụng ở tầng vận chuyển là TCP hay UDP cho chương trình ứng dụng của mình.

Bảng sau so sánh sự khác biệt giữa hai chế độ giao tiếp có nối kết và không nối kết:

Chế độ có nối kết (TCP)	**Chế độ không nối kết (UDP)**

•Tồn tại kênh giao tiếp ảo giữa hai bên giao tiếp •Dữ liệu được gởi đi theo chế độ bảo đảm: có kiểm tra lỗi. truyền lại gói tin lỗi hay mất, bảo đảm thứ tự đến của các gói tin . . . •Dữ liệu chính xác, Tốc độ truyền chậm.	•Không tồn tại kênh giao tiếp ảo giữa hai bên giao tiếp •Dữ liệu được gởi đi theo chế độ không bảo đảm: Không kiểm tra lỗi, không phát hiện không truyền lại gói tin bị lỗi hay mất, không bảo đảm thứ tự đến của các gói tin . . . •Dữ liệu không chính xác, tốc độ truyền nhanh. •Thích hợp cho các ứng dụng cần tốc độ, không cần chính xác cao: truyền âm thanh, hình ảnh . . .=

2. Tạo ứng dụng đơn giản "hello world"

Mô hình Client-Server sử dụng Socket ở chế độ không nối kết (UDP)

2.1. Viết dưới dạng đơn giản UDP client

Yêu cầu: Viết chương trình phía client cho phép gởi một thông điệp đến máy chủ bất kỳ.

Hướng dẫn: Mở Visual Studio .NET, tạo một Project mới, chọn Visual C# projects, và sau đó chọn Windows Application. Đặt tên là "UDP Client" và nhấn OK.

Thiết kế form cho chương trình như hình sau. Đặt tên cho nút button là button1 và hộp văn bản textbox là tbHost.

Hình 3.4 Giao diện UDP Client

Kích chọn nút button và gõ đoạn lệnh sau:

```
private void button1_Click(object sender, System.EventArgs e)
{
UdpClient udpClient = new UdpClient();
  udpClient.Connect(tbHost.Text, 8080);
  Byte[] sendBytes = Encoding.ASCII.GetBytes("Hello
  World?");
    udpClient.Send(sendBytes, sendBytes.Length);
```

```
}
```

Cần bổ sung các namespace sau vào trong chương trình

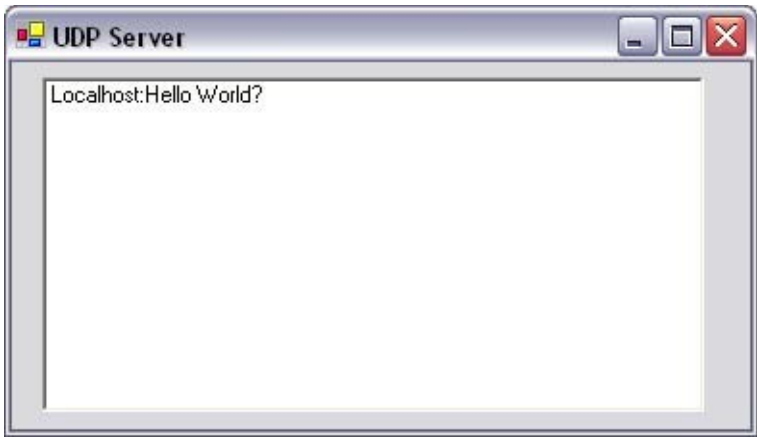
```
using System.Net;
using System.Net.Sockets;
using System.Text;
using System.IO;
```

Nhấn F5 để thực thi chương trình, ta có kết quả như hình 3.5

2.2. Viết dưới dạng đơn giản UDP server

Yêu cầu: Viết chương trình phía server để nhận và hiển thị thông điệp được gởi tới từ chương trình UDP client ở trên.

Hướng dẫn: Tạo một C# project với giao diện như sau



Hình 3.5 Giao diện UDP Server

Đặt tên cho list box là lbConnections.

Ở đây, chúng ta bắt đầu xây dựng hai luồng (thread) cho chương trình.

Đầu tiên là luồng quản lý dữ liệu được gởi đến, mã nguồn chương trình như sau:

```
public void serverThread()
{
 UdpClient udpClient = new UdpClient(8080);
 while(true)
 {
  IPEndPoint RemoteIpEndPoint = new
  IPEndPoint(IPAddress.Any,0);
  Byte[] receiveBytes = udpClient.Receive(ref
  RemoteIpEndPoint);
  string returnData = Encoding.ASCII.GetString(receiveBytes);
  lbConnections.Items.Add(RemoteIpEndPoint.Address.ToString
   () + ":" + returnData.ToString()
```

```
        );
      }
    }
```

Viết code cho sự kiện Form load

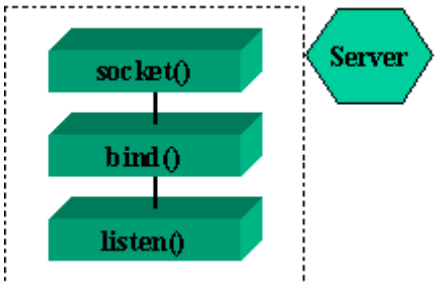
```
private void Form1_Load(object sender, System.EventArgs e){
Thread thdUDPServer = new
Thread(new
ThreadStart(serverThread));
thdUDPServer.Start();
}
```

Thực thi chương trình: Để kiểm tra kết quả của chương trình, cài đặt hai chương trình UDP client và UDP server trên hai máy khác nhau có kết nối mạng. Trên chương trình UDP client nhập vào địa chỉ IP của server. Sau khi bấm nút gởi, kết quả sẽ được hiển thị trên chương trình UDP server.

3. Dùng giao thức TCP/IP để chuyển files

Các giai đoạn xây dựng mô hình Client-Server sử dụng Socket ở chế độ có nối kết (TCP)

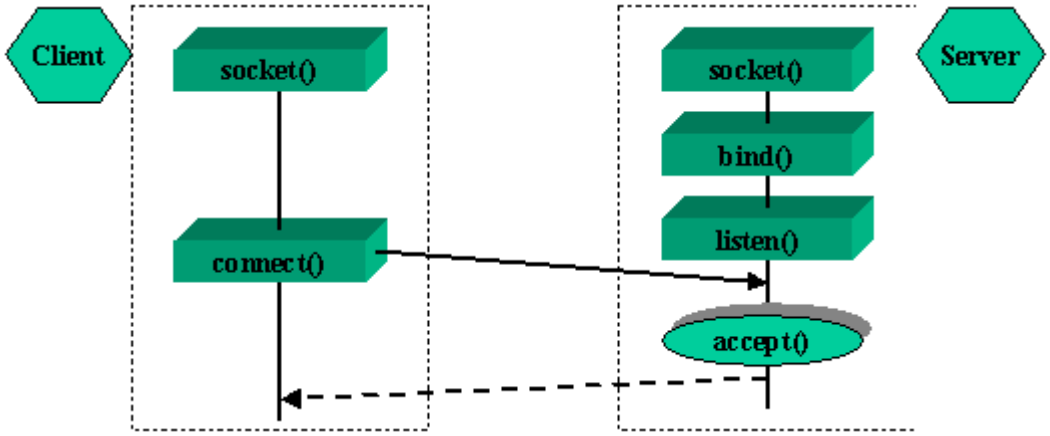
Giai đoạn 1: Server tạo Socket, gán số hiệu cổng và lắng nghe yêu cầu nối kết.



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

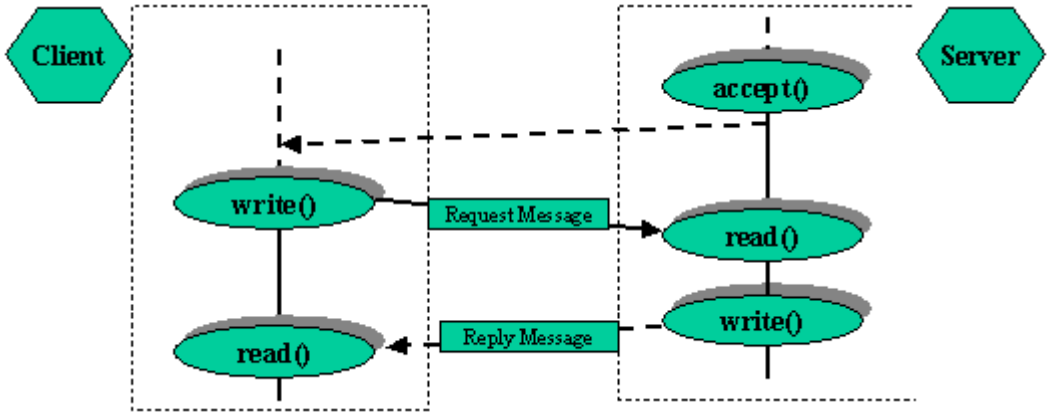
Server sẵn sàng phục vụ Client.

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 rảnh cho socket của Client.
- 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, khi đó một kênh giao tiếp ảo được hình thành, Client và server có thể trao đổi thông tin với nhau thông qua kênh ảo này.

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

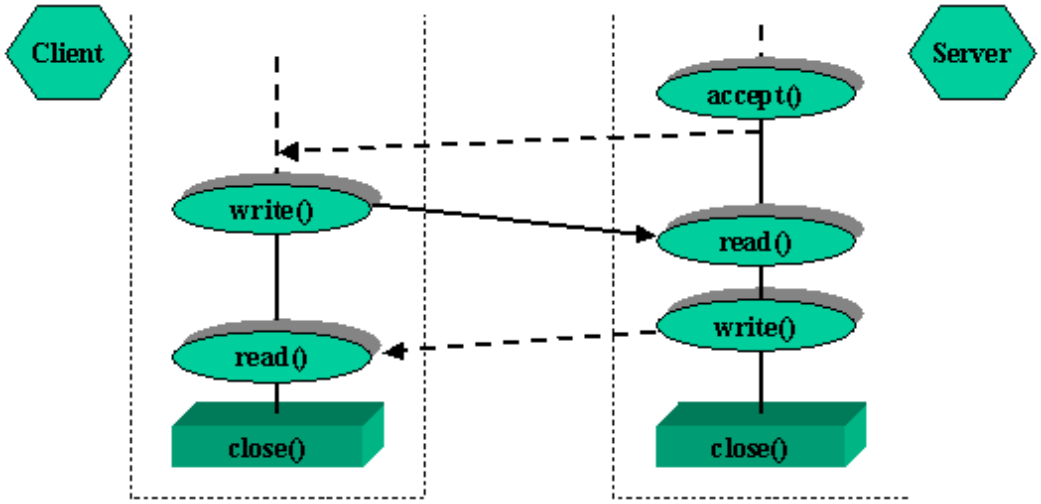


- Sau khi chấp nhận yêu cầu nối kết, thông thường server thực hiện lệnh read() và nghẽn cho đến khi có thông điệp yêu cầu (Request Message) từ client gởi đến.
- Server phân tích và thực thi yêu cầu. Kết quả sẽ được gởi về client bằng lệnh write().
- Sau khi gởi yêu cầu bằng lệnh write(), client chờ nhận 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, qui tắc bắt tay, đồng bộ hóa,...). Thông thường Client sẽ là người gởi yêu cầu đến Server trước.

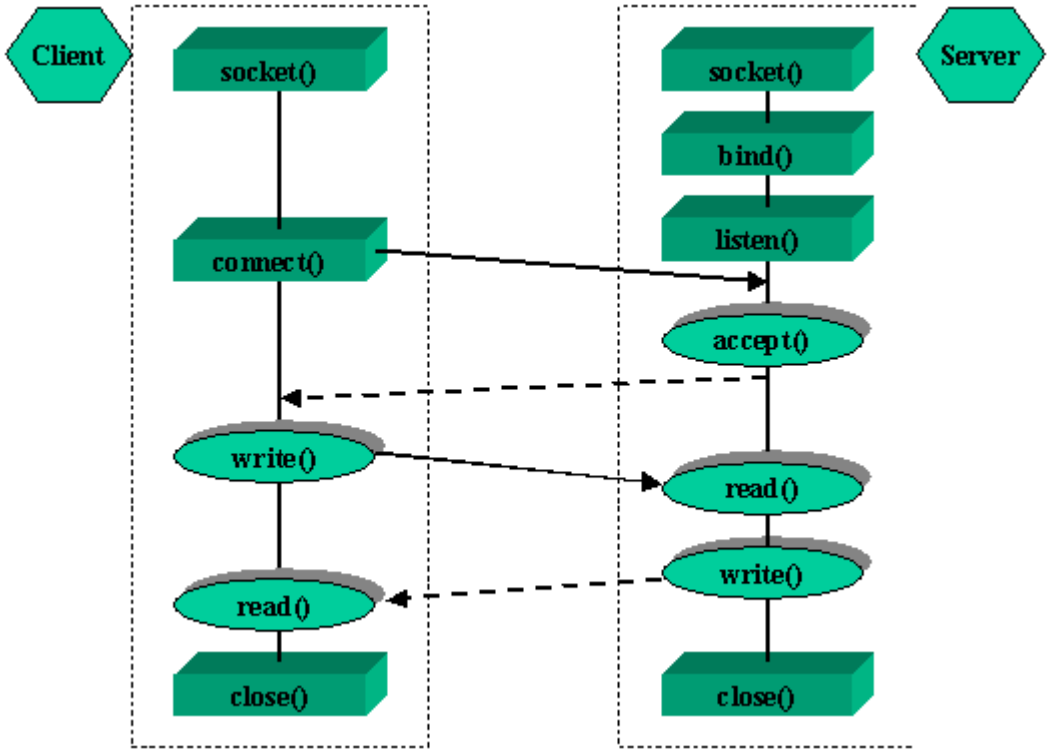
Nếu chúng ta phát triển ứng dụng theo các Protocol đã định nghĩa sẵn, chúng ta phải tham khảo và tuân thủ đúng những qui định của giao thức. Bạn có thể tìm đọc mô tả chi tiết của các Protocol đã được chuẩn hóa trong các tài liệu RFC (Request For Comments). Ngược lại, nếu chúng ta phát triển một ứng dụng Client-Server riêng của mình, thì công việc đầu tiên chúng ta phải thực hiện là đi xây dựng Protocol cho ứng dụng.

Giai đoạn 4: Kết thúc phiên làm việc.



- Các câu lệnh read(), write() có thể được thực hiện nhiều lần (ký hiệu bằng hình ellipse).
- Kênh ảo sẽ bị xóa khi Server hoặc Client đóng socket bằng lệnh close().

Như vậy toàn bộ tiến trình diễn ra như sau:



Chương trình ứng dụng truyền file qua mạng ứng dụng giao thức TCP/IP sẽ được trình bày trong phần bài tập.

4. Gỡ rối trong lập trình mạng

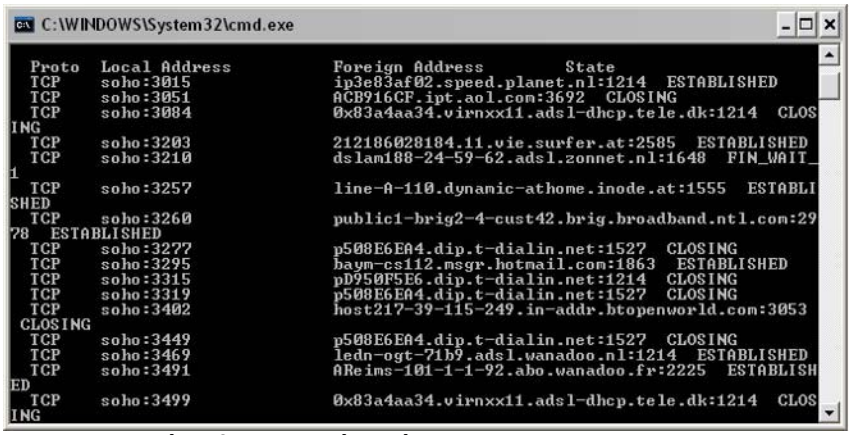
Kết nối mạng có thể bị ngắt do có ứng dụng khác đã chiếm dụng cổng mà chúng ta muốn sử dụng. Do đó phương thức Connect hoặc Listen không phải bao giờ cũng được thực hiện thành công. Do đó, sử dụng cấu trúc bắt lỗi try/catch trong lập trình mạng là cần thiết.

Ví dụ:

```
try
{
  serverSocket.Bind(ipepServer);
  serverSocket.Listen(-1);
}
catch(SocketException e)
{
  MessageBox.Show(e.Message);
}
catch(Exception e)
{MessageBox.Show(e.Message); Application.Exit();

}

Application.Exit()
End try
```

Vấn đề nữa hay gặp trong các ứng dụng mạng là khả năng mở rộng. Đây là khả năng đáp ứng của ứng dụng với số lượng kết nối lớn trong cùng một thời điểm. Để thử nghiệm, chúng ta có thể kích chuột nhiều lần vào nút Connect hoặc Send trên ứng dụng client để xem chương trình có thể bị đứng do hết bộ nhớ, ngắt giữa chừng, rớt kết nối hay vẫn hoạt động tốt. Để phát hiện ra lỗi của những ứng dụng mạng đa luồng, chúng ta sử dụng lớp System. Diagnostics.Trace, hoặc là câu lệnh Console.WriteLine ở phương thức vào hoặc thoát của chương trình.

Đối với các ứng dụng mạng có giao tiếp với một ứng dụng được phân phối bởi hãng thứ 3, việc kiểm tra lỗi giữa client và server trở nên khó khăn hơn, đặc biệt khi mà giao thức và cổng kết nối không được tiết lộ hoặc được mô tả đầy đủ. Trong trường hợp này, việc trước tiên chúng ta cần phải làm là kiểm tra xem những cổng nào đang hoạt động. Công cụ hữu ích để làm việc này là netstat. Để mở netstat, chúng ta gõ netstat vào trong dấu nhắc lệnh hoặc vào run. Xem hình sau



Netstat liệt kê tất cả các kết nối vào/ra trên máy tính tại thời điểm hiện tại cộng với số cổng tương ứng. Dùng phương pháp đối chiếu khi chạy và tắc chương trình để tìm ra cổng mà chương trình sử dụng. Việc xác định cổng của ứng dụng là bước đầu tiên để can thiệp vào trong giao thức, để phân tích từng bit và byte được gởi qua lại giữa hai ứng dụng chúng ta cần sử dụng các công cụ phân tích khác ví dụ Trace Plus (www.sstinc.com).

5. Mức Socket trong .NET

Lớp quan trọng nhất trong lập trình mạng với .NET là lớp socket. Lớp socket có thể được sử dụng cho giao thức TCP/IP hoặc UDP cho cả client và server.

Không gian tên của lớp Sockets: System.Net.Sockets

Khai báo lớp socket:

public class Socket : IDisposable

Ví dụ sau đây minh họa ứng dụng server sử dụng lớp socket.Ứng dụng server cho phép lắng nghe các kết nối gởi đến server qua cổng 8080.

Đoạn mã chương trình

```
private void btnListen_Click(object sender, System.EventArgs e)
{
int bytesReceived = 0; byte[] recv =
new byte[1]; Socket clientSocket;
Socket listenerSocket = new
Socket( AddressFamily.InterNetwork,
SocketType.Stream, ProtocolType.Tcp
);
IPHostEntry IPHost = Dns.GetHostByName(Dns.GetHostName());
IPEndPoint ipepServer = new
IPEndPoint(IPHost.AddressList[0],8080);
listenerSocket.Bind(ipepServer);
listenerSocket.Listen(-1);
```

```
clientSocket = listenerSocket.Accept();
if (clientSocket.Connected)
{
do
{
bytesReceived = clientSocket.Receive(recv);
tbStatus.Text += Encoding.ASCII.GetString(recv);
}

while (bytesReceived!=0);
}
}
```

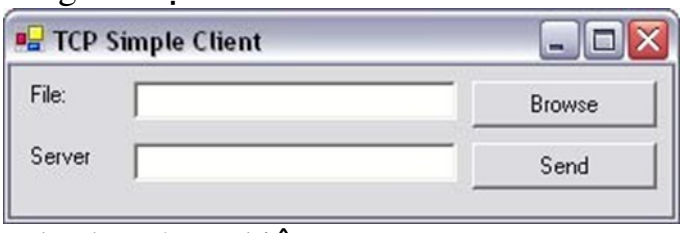
BÀI THỰC HÀNH CỦA HỌC VIÊN

Bài 1 : Viết chương trình cho phép gởi file qua mạng sử dụng giao thức TCP/IP.

- Thời gian: 04 giờ

HƯỚNG DẪN THỰC HIỆN BÀI TẬP ỨNG DỤNG

1. Viết ứng dụng gởi file trên máy client: TCP Simple Client
   1.1 Thiết kế giao diện như sau



   1.2 Viết code cho các sự kiện sau
   Khi người dùng kích button Browse chỉ đến đường dẫn file cần gởi
   Khi người dùng kích vào button Send, gởi file:
   + Nạp file cần gởi vào bộ đệm
   + Tạo một kết nối TCP/IP và gởi file
2. Viết ứng dụng gởi file trên máy client: TCP Simple Client
   2.1 Thiết kế giao diện như sau

2.2 Viết code

Lắng nghe kết nối từ client

Lưu file được gởi lên server tại địa chỉ c:\\my documents\ \upload.txt

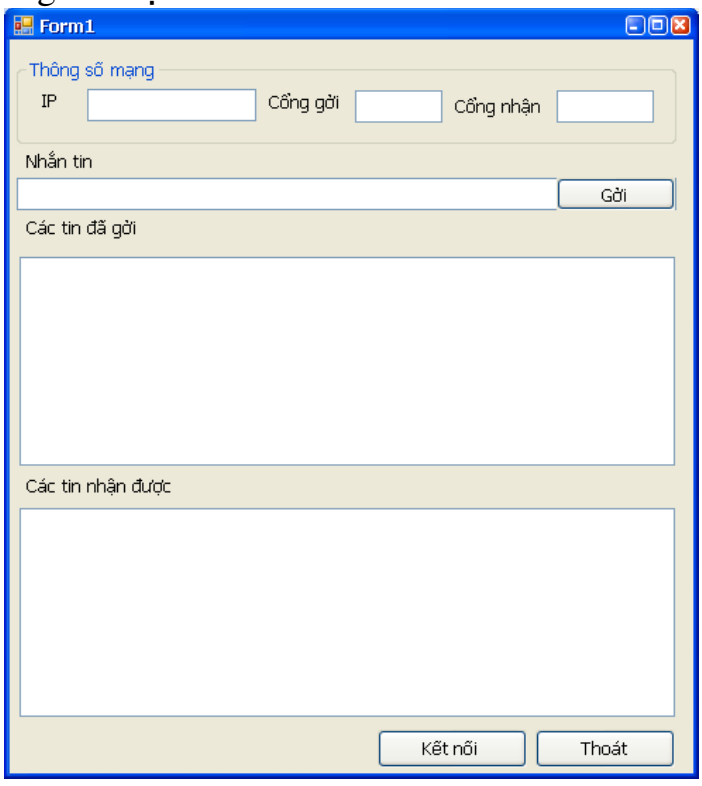
- Thời gian thực hiện bài tập vượt quá 5% thời gian cho phép sẽ không được đánh giá.

- Thí sinh phải tuyệt đối tuân thủ các qui định an toàn lao động, các qui định của xưởng thực tập, nếu vi phạm sẽ bị đình chỉ thi.

Bài 2 : Viết chương Chat giữa 2 máy tính dùng UDP và TCP

Cách 1 : Dùng UDP

Bước 1 : Thiết kế giao diện cho Client



Bước 2 : Viết mã

//Cho nút kết nối

```
private void btnKetNoi_Click(object sender, EventArgs e)
    {
```

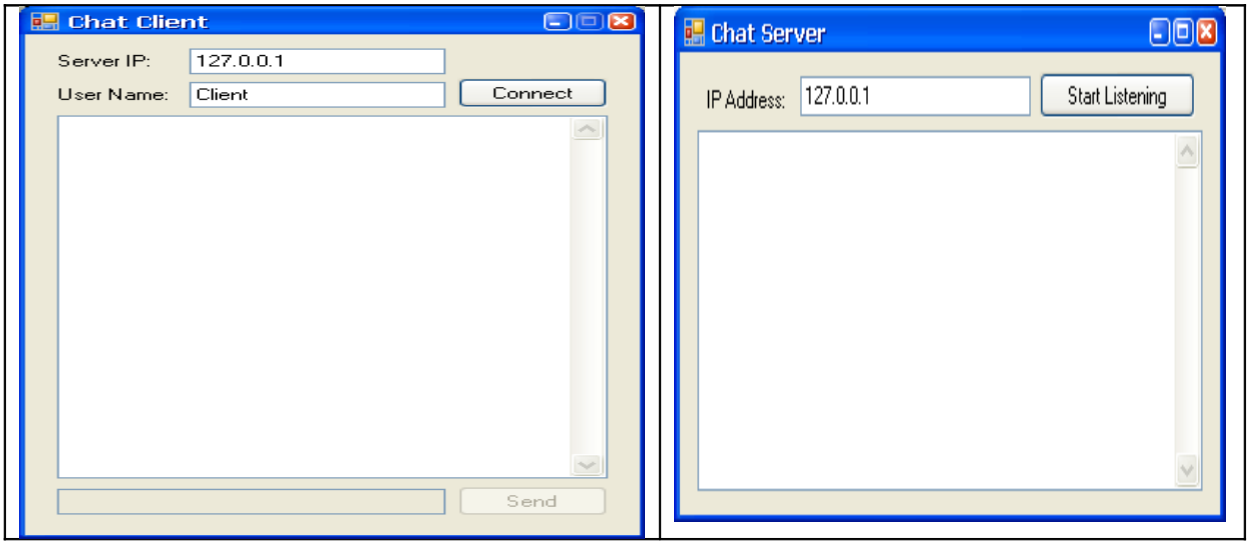
```
            _localPort = this.txtLocalPort.Text;
            _remotePort = this.txtPort.Text;
            _applications = new UdpClient(int.Parse(_localPort));
            _thread = new Thread(Explore);
            _thread.Start();
            this.btnGoi.Click += btnGoiClick;
            this.btnGoi.Enabled = true;
            this.btnKetNoi.Enabled = false;
            txtIP.ReadOnly = txtLocalPort.ReadOnly =txtPort.ReadOnly = true;
        }
//Cho nút gởi dữ liệu đi
public partial class Form1 : Form
    {
        string _localPort = "10";
        string _remotePort = "1000";
        UdpClient _applications = new UdpClient();
        Thread _thread;
        bool _exit = false;
        delegate void ClearCacheReceivedData(string Data, string RemoteHost);
        public Form1()
        {
            InitializeComponent();
        }

        //Khi ấn Enter vào trong o Chat
        private void txtChat_KeyDown(object sender, KeyEventArgs e)
        {
            if (e.KeyValue==13)
                {
                    lstSent.Items.Add(txtChat.Text);
                }
        }

        //Thủ tục gởi tin nhắn
        private void btnGoiClick(object sender, EventArgs e)
        {
            IPAddress ip;
            //Kiểm tra xem IP nhập để Chat có phù hợp hay không
            if (!IPAddress.TryParse(txtIP.Text, out ip))
                MessageBox.Show("Hãy nhập chính xác IP nhận",
                "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Error);
            else
                {
                    //Gởi dữ liệu đến máy Chat
                    SentData();
                    //Ghi dữ liệu vừa Chat vào mục tin nhắn đã gởi
                    lstSent.Items.Insert(0, txtChat.Text);
                    txtChat.Clear();
                }
        }
```

```
        private void SentData()
        {
            //Đổi chuỗi cần gởi qua mảng Byte
            byte[] msg;
            msg = System.Text.Encoding.UTF8.GetBytes(txtChat.Text);
            //Sử dụng phương thức Send của UDP để gởi dữ liệu
            _applications.Send(msg, msg.Length, txtIP.Text,int.Parse(_remotePort));

        }
        private void ReceivedData(string Data, string RemoteHost)
        {
            if (lstReceived.InvokeRequired)
                {
                    ClearCacheReceivedData clearCacheReceivedData = new
                    ClearCacheReceivedData(ReceivedData);
                    lstReceived.Invoke(clearCacheReceivedData, new object[] {Data,
RemoteHost });
                    return;
                }
            string msg = "";
            msg = "(Ngư_ời gửi: " + RemoteHost + ")" + Data;
            lstReceived.Items.Insert(0, msg);
        }

        private void Explore()
        {
            IPAddress ip;
            byte[] msg;
            string str = "";
            //lấy danh sách IP của ứng dụng ở Card đầu tiên
            ip = Dns.GetHostEntry(_remotePort).AddressList[0];

            IPEndPoint ep = new IPEndPoint(ip, Convert.ToInt16(_remotePort));
            while (_exit == false)
            {
                Application.DoEvents();
                //nếu UDP đang hoạt động
                if (_applications.Available > 0)
                {
                    msg = _applications.Receive(ref ep);
                    str = System.Text.Encoding.UTF8.GetString(msg);
                    ReceivedData(str, ep.Address.ToString());
                }
            }
        }
```

Cách 2 : Dùng TCP

Bước 1 : Thiết kế giao diện cho Client, Server

Client	Server



Bước 2 : Viết mã

Cho Client

//Khai báo thêm các không gian tên

```
using System.Net;
using System.Net.Sockets;
using System.IO;
using System.Threading;
 //Kết nối đến Server
private void btnConnect_Click(object sender, EventArgs e)
      {
        // nếu chưa kết nối ở thời điểm hiện tại thì đợi để kết nối
        if (Connected == false)
        {
          // Gọi thủ tục cài đặt kết nối
          InitializeConnection();
        }
        // nếu đã kết nối rồi thì ngắt kết nối
        else
        {
          //Gọi thủ tục đóng kết nối
          CloseConnection("Disconnected at user's request.");
        }
      }
{
      // Will hold the user name
      private string UserName = "Unknown";
      //Biến gởi dữ liệu
      private StreamWriter swSender;
      //Biến ghi dữ liệu
      private StreamReader srReceiver;
      //Đối tượng TcpClient
      private TcpClient tcpServer;
      // Needed to update the form with messages from another thread
      private delegate void UpdateLogCallback(string strMessage);
```

```
        // Needed to set the form to a "disconnected" state from another thread
        private delegate void CloseConnectionCallback(string strReason);
        private Thread thrMessaging;
        //Biến IP Address
        private IPAddress ipAddr;
        //Trạng thái kết nối
        private bool Connected;

        public Form1()
        {
            // On application exit, don't forget to disconnect first
            Application.ApplicationExit += new EventHandler(OnApplicationExit);
            InitializeComponent();
        }

        // The event handler for application exit
        public void OnApplicationExit(object sender, EventArgs e)
        {
            if (Connected == true)
            {
                // Closes the connections, streams, etc.
                Connected = false;
                swSender.Close();
                srReceiver.Close();
                tcpServer.Close();
            }
        }
    //Thủ tục kết nối
        private void InitializeConnection()
        {
            // Kiểm tra địa chỉ IP của hộp Textbox
            ipAddr = IPAddress.Parse(txtIp.Text);
            // Tạo  mới 1 TCP để kết nối với Chat server
            tcpServer = new TcpClient();
            tcpServer.Connect(ipAddr, 1986);
            // Ngắn trạng thái kết nối được rồi là true
            Connected = true;
            // Gắn tên người dùng
            UserName = txtUser.Text;

            // Đóng và kích hoạt các trường
            txtIp.Enabled = false;
            txtUser.Enabled = false;
            txtMessage.Enabled = true;
            btnSend.Enabled = true;
            btnConnect.Text = "Disconnect";

            // Gởi tên người dùng đến Server
            swSender = new StreamWriter(tcpServer.GetStream());
```

```
            swSender.WriteLine(txtUser.Text);
            swSender.Flush();

            // Bắt đầu tiến trình để nhận các tin nhắn và trạng thái nối kết
            thrMessaging = new Thread(new ThreadStart(ReceiveMessages));
            thrMessaging.Start();
        }

        //Thủ tục nhận tin nhắn
        private void ReceiveMessages()
        {
            // Nhận thông tin từ Server
            srReceiver = new StreamReader(tcpServer.GetStream());
            // Nếu kí tự đầu tiên của chuỗi nhận được là 1 thì kết nối thành công
            string ConResponse = srReceiver.ReadLine();
            // If the first character is a 1, connection was successful
            if (ConResponse[0] == '1')
            {
                // Update the form to tell it we are now connected
                this.Invoke(new UpdateLogCallback(this.UpdateLog), new object[] { "Connected
Successfully!" });
            }
            else // If the first character is not a 1 (probably a 0), the connection was unsuccessful
            {
                string Reason = "Not Connected: ";
                // Extract the reason out of the response message. The reason starts at the 3rd
character
                Reason += ConResponse.Substring(2, ConResponse.Length - 2);
                // Update the form with the reason why we couldn't connect
                this.Invoke(new CloseConnectionCallback(this.CloseConnection), new object[]
{ Reason });
                // Exit the method
                return;
            }
            // While we are successfully connected, read incoming lines from the server
            while (Connected)
            {
                // Viết các dữ liệu nhận được từ Server vào txtlog
                this.Invoke(new UpdateLogCallback(this.UpdateLog), new object[]
{ srReceiver.ReadLine() });
            }
        }

        // This method is called from a different thread in order to update the log TextBox
        private void UpdateLog(string strMessage)
        {
            // Append text also scrolls the TextBox to the bottom each time
            txtLog.AppendText(strMessage + "\r\n");
        }
```

```
// Thủ tục đóng kết nối hiện tại
private void CloseConnection(string Reason)
{
    // Xem lý do tại sao kết nối bị dừng
    txtLog.AppendText(Reason + "\r\n");
    // Kích hoạt hoặc đóng các điều khiển
    txtIp.Enabled = true;
    txtUser.Enabled = true;
    txtMessage.Enabled = false;
    btnSend.Enabled = false;
    btnConnect.Text = "Connect";

    // Đóng các đối tượng
    Connected = false;
    swSender.Close();
    srReceiver.Close();
    tcpServer.Close();
}

// Gởi tin nhắn đến server
private void SendMessage()
{
    if (txtMessage.Lines.Length >= 1)
    {
        swSender.WriteLine(txtMessage.Text);
        swSender.Flush();
        txtMessage.Lines = null;
    }
    txtMessage.Text = "";
}

// Khi muốn gởi tin nhắn thì ấn vào nút send để gởi
private void btnSend_Click(object sender, EventArgs e)
{
    SendMessage();
}

// But we also want to send the message once Enter is pressed
private void txtMessage_KeyPress(object sender, KeyPressEventArgs e)
{
    // If the key is Enter
    if (e.KeyChar == (char)13)
    {
        SendMessage();
    }
}
```

Code cho Server

```
//Lắng nghe kết nối
private void btnListen_Click(object sender, EventArgs e)
```

```
        {
            // Kiểm tra IP của server ở TextBox
            IPAddress ipAddr = IPAddress.Parse(txtIp.Text);
            // Tạo 1 đối tượng mới của ChatServer
            ChatServer mainServer = new ChatServer(ipAddr);
            // Hook the StatusChanged event handler to mainServer_StatusChanged
            ChatServer.StatusChanged += new
StatusChangedEventHandler(mainServer_StatusChanged);
            // lắng nghe kết nối
            mainServer.StartListening();
            // thể hiện lắng nghe các kết nối
            txtLog.AppendText("Monitoring for connections...\r\n");
        }

public void mainServer_StatusChanged(object sender, StatusChangedEventArgs e)
        {
            // Gọi phương thức cập nhật của Form
            this.Invoke(new UpdateStatusCallback(this.UpdateStatus), new object[]
{ e.EventMessage });
        }

        private void UpdateStatus(string strMessage)
        {
            // Cập nhật các tin nhắn vào txtLog
            txtLog.AppendText(strMessage + "\r\n");
        }
```

CÂU HỎI ÔN TẬP

Câu 1 : Nêu khái niệm về lập trình Socket ?

Câu 2 : Nêu khái niệm về IP và Port trong lập trình mạng

Câu 3 : Nêu một số ứng dụng và qui định đối với Port ?

Câu 4 : Nêu các đặc trưng cơ bản (Khái niệm, các thuộc tính và phương thức) của lớp IPAddress?

Câu 5 : Nêu các đặc trưng cơ bản (Khái niệm, các thuộc tính và phương thức) của lớp IPEndPoint?

Câu 6 : Nêu các đặc trưng cơ bản (Khái niệm, các thuộc tính và phương thức) của lớp IPHostEntry?

Câu 7 : Nêu các đặc trưng cơ bản (Khái niệm, các thuộc tính và phương thức) của lớp DNS?

Câu 8 : Nêu các đặc trưng cơ bản (Khái niệm, các thuộc tính và phương thức) của lớp UDPClient?

Câu 9 : Nêu trình tự kết nối của Lớp UDPClient? (Phía Server và Client)

Câu 10 : Nêu các đặc trưng cơ bản (Khái niệm, các thuộc tính và phương thức) của lớp TCPClient?

Câu 11 : Nêu trình tự kết nối của Lớp TCPClient? (Phía Server và Client)
Câu 12 : Nêu các đặc trưng cơ bản (Khái niệm, các thuộc tính và phương thức) của lớp TCPListener?
Câu 13 : Nêu các đặc trưng cơ bản (Khái niệm, các thuộc tính và phương thức) của lớp TCPListener?
Câu 14 : So sánh giữa hai giao thức TCP và UDP trong lập trình mạng.

BÀI TẬP
Bài 1: Viết chương trình Sever giải đáp tên miền. Nếu máy khách gữi tên máy thì sever sẽ gởi về địa chỉ IP (danh sách này tự tạo ra – khoản 3 cập để minh hoạ).
Bài 2: viết chương trình UDP (Ứng dụng A) đặt trên một máy thực hiện các công việc sau, khi một ứng dụng B gữi 1 kiểu chuỗi tiếng Anh thì ứng dụng A sẽ gửi trả lại nghĩa tiếng Việt tương ứng. Nếu từ này không có trong từ điển (chỉ có 3 từ Computer,Ram,HDD)thì ứng dụng A cho người dùng biết từ này không có trong từ điển.
Bài 3: Viết chương trình Client/Sever trong đó khi Client di chuyển chuột thì sever cũng di chuyển theo.
Bài 4: Viết Chương trình Client/Server khi Client gửi "Shutdown","Restast" thì Sever khởi động hoặc tắt máy tương ứng
Bài 5: Viết chương trình Chat giữa các máy tính ?

BÀI 4 : KẾT NỐI VỚI WEB SERVER

Mã bài : MĐ35.4

Mục tiêu của bài:

- Trình bày được cách lập trình sử dụng các Giao thức để truy cập với máy chủ Web (Web Server).
- Xây dựng các ứng dụng làm việc với máy chủ Web (WebServer).
- Thực hiện các thao tác an toàn với máy tính.

1. Giới thiệu về HTTP

Đây là chương hướng dẫn cách lấy dữ liệu từ WEB và sử dụng vào mục đích khác trong ứng dụng của riêng bạn. Như đã đề cập trong Chương 1, các trang web được lưu trữ trên máy tính chạy phần mềm máy chủ web như Microsoft Internet Information Services (IIS) hoặc Apache. Giao thức truyền siêu văn bản (HTTP) được sử dụng để giao tiếp với các ứng dụng và lấy các trang web.

Có nhiều lý do tại sao một ứng dụng có thể tương tác với một trang web Web, như sau:

+ Kiểm tra các bản cập nhật và tải về các bản vá lỗi và nâng cấp.
+ Lấy thông tin về dữ liệu mà thay đổi từ giờ này sang giờ khác (ví dụ như Chia sẻ các giá trị, tỷ lệ chuyển đổi tiền tệ, thời tiết)
+ Tự động truy vấn dữ liệu từ các dịch vụ do bên thứ ba (ví dụ như Zip code tra cứu, thư mục điện thoại, dịch vụ dịch thuật ngôn ngữ)
+ Xây dựng một công cụ tìm kiếm.
+ Cache các trang web để truy cập nhanh hơn hoặc hoạt động như một chủ proxy.

Nửa đầu của chương này mô tả làm thế nào để gửi và nhận dữ liệu đến các máy chủ web. Điều này bao gồm một ví dụ về làm thế nào để thao tác các dữ liệu HTML nhận được từ máy chủ web. Chương này được ký kết với một thực hiện một máy chủ web tùy chỉnh, mà có thể được sử dụng thay vì IIS.

Data mining

Khai thác dữ liệu là một ứng dụng tải một trang web và các chiết xuất thông tin cụ thể từ nó sẽ tự động. Nó thường đề cập đến việc thu hồi số lượng lớn dữ liệu từ các trang web mà không bao giờ được thiết kế để đọc tự động.

Một ứng dụng mẫu có thể là một hướng dẫn chương trình truyền hình mà có thể tải về thông tin lập kế hoạch từ các trang web Web TV và lưu trữ nó trong một cơ sở dữ liệu để tham khảo nhanh.

Lưu ý: Bạn nên luôn luôn kiểm tra với quản trị trang web cho dù họ cho phép khai thác dữ liệu trên các trang web của họ bởi vì nó có thể vi phạm quyền tác giả hoặc đặt tải quá nhiều trên các máy chủ của họ. Không được phép khai thác dữ liệu có thể dẫn đến một quản trị viên Web chặn địa chỉ IP của bạn hoặc tệ hơn!

Để trích xuất dữ liệu hữu ích từ HTML này, bạn sẽ cần phải được làm quen với ngôn ngữ và giỏi việc nhận các mẫu của HTML có chứa các dữ liệu cần thiết, tuy nhiên, một số sản phẩm tốt thương mại hỗ trợ các nhà phát triển với khai thác dữ liệu từ trang HTML, và giải pháp không phải luôn luôn là ý tưởng tốt nhất.

2. HTTP

HTTP hoạt động trên TCP / IP port 80 và được mô tả dứt khoát trong RFC 2616. Giao thức là khá đơn giản. Khách hàng sẽ mở cổng TCP 80 để một máy chủ, khách hàng sẽ gửi một yêu cầu HTTP, máy chủ sẽ gửi lại một phản ứng HTTP, và máy chủ đóng kết nối TCP.

2.1. Yêu cầu trong HTTP

Dạng đơn giản nhất như sau:

```
GET /
<enter><enter>
```

Với một số server cần phải xác nhận DNS Name trong lệnh GET

Yêu cầu này sẽ hướng dẫn các máy chủ để trả lại trang Web mặc định, tuy nhiên, các yêu cầu HTTP nói chung là phức tạp hơn, chẳng hạn như sau:

```
GET / HTTP/1.1
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg,
application/vnd.ms-powerpoint, application/vnd.ms-excel,
application/msword, */*
Accept-Language: en-gb
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT
5.1; .NET CLR 1.0.3705)
Host: 127.0.0.1:90
Connection: Keep-Alive
```

Này cho một vài điều về Client, chẳng hạn như loại của trình duyệt và những gì sắp xếp dữ liệu trình duyệt có thể làm cho các máy chủ.

Bảng 4.1 cho thấy một danh sách đầy đủ các tiêu chuẩn yêu cầu HTTP tiêu đề như sau:

HTTP header	Ý nghĩa
Accept	Được sử dụng để xác định các phương tiện truyền thông (MIME) các loại có thể chấp nhận được cho phản ứng. Các loại * / * cho tất cả các loại phương tiện truyền thông và type / * cho tất cả các phân nhóm của loại đó. Trong ví dụ trên, application / msword chỉ ra rằng trình duyệt có thể hiển thị các tài liệu Word.
Accept-Charset	Được sử dụng để xác định các bộ ký tự được chấp nhận

	trong phản ứng. Trong trường hợp một số vấn đề của khách hàng Accept-Charset: iso-8859-5, được servershould biết rằng khách hàng không có thể làm cho Nhật Bản (Unicode) ký tự.
Accept-Encoding	Được sử dụng để xác định nếu khách hàng có thể xử lý các dữ liệu nén. Trong ví dụ trên, trình duyệt có khả năng giải thích GZIP nén dữ liệu.
Accept-Language	Được sử dụng để chỉ ra tuỳ chọn ngôn ngữ của người sử dụng. Điều này có thể được sử dụng để ước tính vị trí địa lý của một khách hàng; en-gb trong ví dụ trên có thể cho thấy rằng khách hàng là từ Vương quốc Anh.
Authorization	Được sử dụng để cung cấp chứng thực giữa khách hàng và máy chủ. Tham khảo RFC 2617
Host	Máy chủ cho biết địa chỉ IP của máy chủ dự định gõ vào khách hàng. Điều này có thể khác với địa chỉ IP đích thực tế nếu yêu cầu phải đi qua một proxy. Địa chỉ host 127.0.0. 1:90 trong ví dụ trên cho thấy rằng các khách hàng trên cùng một máy tính như máy chủ, được chạy trên cổng 90.
If-Modified-Since	Chỉ ra rằng trang không được trả lại nếu nó đã không được thay đổi kể từ một ngày nhất định. Điều này cho phép một cơ chế bộ nhớ đệm để làm việc hiệu quả. Một ví dụ là Nếu-Modified-Since: Sat, 29 Oct 1994 19:43:31 GMT.
Proxy-Authorization	Điều này cung cấp để xác thực giữa khách hàng và các proxy. Tham khảo RFC 2617
Range	Cung cấp một cơ chế để lấy một phần của một trang web bằng cách xác định phạm vi các byte các máy chủ nên trở lại, điều này có thể không được thực hiện trên tất cả các máy chủ. Một ví dụ là byte = 500-600,601-999 .
Referer	Điều này cho thấy Client đã truy cập trang cuối cùng trước khi đi đến URL cụ thể này. Một ví dụ là Referer: http://www.w3.org/index.html. (Lỗi chính tả của "giới thiệu" không phải là một lỗi đánh máy).
TE	Chuyển mã hóa (TE) cho thấy nó có thể chấp nhận gia hạn chuyển giao mã hóa trong phản ứng và nếu nó có thể chấp nhận các trường trailer trong một mã hóa chuyển chunked.
User-Agent	Cho biết loại thiết bị Client đang chạy từ. Trong ví dụ trên, trình duyệt Internet Explorer 6.

Content-Type	Được sử dụng trong các yêu cầu POST. Nó chỉ ra kiểu MIME của dữ liệu được đăng, mà thường là ứng dụng / xwww-form-urlencoded.
Content-Length	Được sử dụng trong các yêu cầu POST. Nó cho biết chiều dài của dữ liệu ngay lập tức sau khi đường gấp đôi.

Lưu ý: thiết bị cụ thể tiêu đề HTTP yêu cầu được bắt đầu với "x"

GET và POST HTTP lệnh phổ biến nhất. Có những người khác, chẳng hạn như HEAD, OPTIONS, PUT, DELETE, và Trace, và bạn đọc quan tâm có thể tham khảo RFC 2616 để biết thông tin về các lệnh HTTP.

Nhà phát triển web có thể quen thuộc với GET và POST từ thẻ hình thức HTML, có dạng:

<form name="myForm" action="someDynamicPage" method="POST">

Sự khác biệt từ quan điểm của một người sử dụng xem là tham số hình thức không xuất hiện trong thanh URL của trình duyệt khi nộp mẫu đơn này. Các tham số được chứa trong khu vực ngay lập tức sau khi thức ăn doubleline. Một yêu cầu POST giống như sau:

POST / HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Content-Length: 17
myField=some+text

2.2. Đáp ứng trong HTTP

Khi máy chủ nhận được một yêu cầu HTTP, nó lấy trang được yêu cầu
và trả về nó cùng với một tiêu đề HTTP. Điều này được biết đến như là phản ứng HTTP.

Một ví dụ đáp ứng dữ liệu

HTTP/1.1 200 OK
Server: Microsoft-IIS/5.1
Date: Sun, 05 Jan 2003 20:59:47 GMT
Connection: Keep-Alive
Content-Length: 25
Content-Type: text/html
Set-Cookie:
ASPSESSIONIDQGGQQFCO=MEPLJPHDAGAEHENKAHIHGHGH;
path=/
Cache-control: private
This is a test html page!

HTTP request header	Ý nghĩa
ETag	Thẻ thực thể được sử dụng kết hợp với các yêu cầu HTTP Ifsuffixed. Các máy chủ hiếm khi trả lại

	nó.
Location	Nó được sử dụng trong chuyển hướng, trình duyệt được yêu cầu để tải một trang khác nhau. Được sử dụng kết hợp với các phản hồi HTTP 3xx.
Proxy-Authenticate	Điều này cung cấp để xác thực giữa khách hàng và các proxy. Tham khảo RFC 2617 Phần 14,33
Server	Chỉ phiên bản máy chủ và nhà cung cấp. Trong ví dụ trên, máy chủ IIS đang chạy trên Windows XP.
WWW-Authenticate	Điều này cung cấp để xác thực giữa khách hàng và các proxy. Tham khảo RFC 2617 Phần 14,47
Content-Type	Chỉ kiểu MIME của nội dung trả lại. Trong ví dụ trên, loại là HTML
Content-Length	Cho biết số lượng dữ liệu theo các nguồn cấp dữ liệu trực tuyến đôi. Các máy chủ sẽ đóng kết nối khi nó đã gửi tất cả dữ liệu, do đó, nó không phải là luôn luôn cần thiết để xử lý lệnh này.
Set-Cookie	Một cookie là một file nhỏ mà cư trú trên máy khách. Một cookie có một cái tên và giá trị. Trong ví dụ trên, tên cookie là ASPSESSIONIDQGGQQFCO

Trên màn hình của máy Client sẽ hiển thị thông báo "This is a test html page!" để đáp ứng với lệnh này.

HTTP response code range	**Ý nghĩa**
100–199	Thông tin: Yêu cầu nhận được, tiếp tục quá trình.
200–299	Thành công: hành động được thành công nhận được, hiểu, và được chấp nhận.
300–399	Điều hướng: thêm tác phải được thực hiện để hoàn thành theo yêu cầu
400–499	Điều hướng: thêm tác phải được thực hiện để hoàn thành theo yêu cầu
500-599	Lỗi máy chủ: Các máy chủ không thành công để hoàn thành một yêu cầu rõ ràng hợp lệ.

Tất cả các phản hồi HTTP có một mã phản hồi. Trong ví dụ trên, các mã phản ứng là 200. Con số này được theo sau bởi một số văn bản của con người có thể đọc được (tức là OK)

Mã phản hồi được chia thành năm loại chính thể hiện trong Bảng 4.3.

2.3. Kiểu MIME

Multipart Internet Mail Extensions (MIME) các loại là một phương tiện để mô tả các loại dữ liệu, như vậy mà một máy tính khác sẽ biết làm thế nào để xử lý các dữ liệu và làm thế nào để hiển thị nó có hiệu quả cho người sử

dụng
Để minh họa cho ví dụ., Nếu bạn thay đổi phần mở rộng của một hình ảnh JPEG (JPG). TXT, và nhấp vào nó, bạn sẽ thấy một mớ lộn xộn các ký tự lạ, không phải là hình ảnh. Điều này là bởi vì Windows có chứa một ánh xạ từ tập tin mở rộng để nộp loại, và JPG và TXT được ánh xạ tới các loại tập tin khác nhau: image / jpeg JPG và đồng bằng văn bản / TXT.

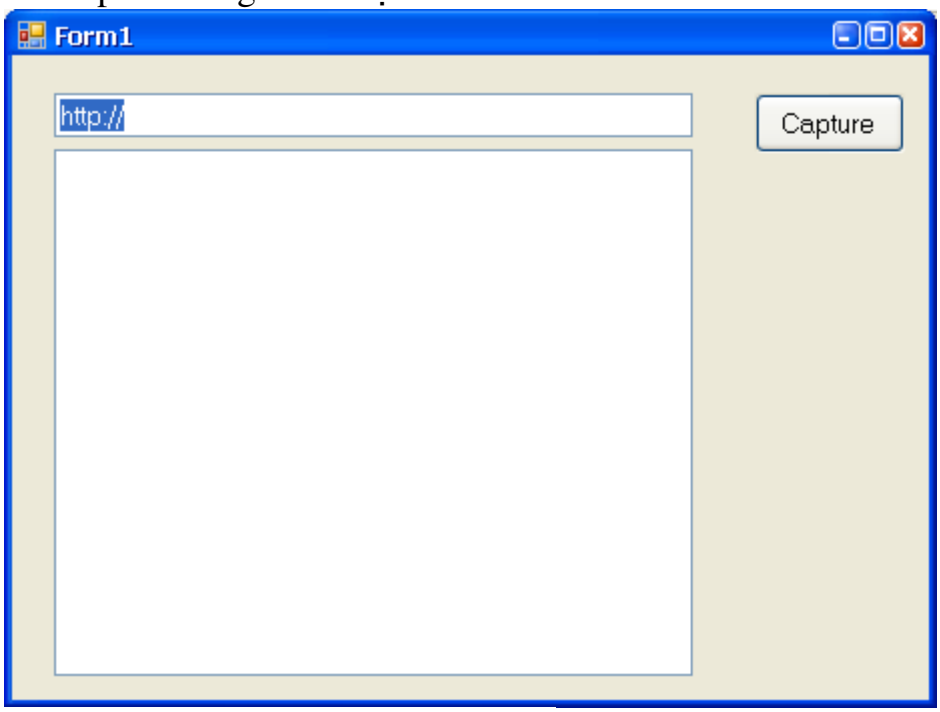
Để tìm một kiểu MIME cho một tập tin cụ thể, chẳng hạn như mp3, bạn có thể mở trình biên tập registry bằng cách vào Start> Run, sau đó gõ REGEDIT. Sau đó bấm vào HKEY_CLASSES_ROOT, di chuyển xuống. mp3, và các loại MIME được viết tiếp theo Nội dung Loại.

2.4. Không gian tên System.Web

Một trong những ứng dụng phổ biến nhất của HTTP trong các ứng dụng là khả năng để tải nội dung HTML của một trang thành một chuỗi. Ứng dụng sau đây chứng minh khái niệm này.

Đó chắc chắn là có thể thực hiện HTTP ở cấp ổ cắm, nhưng có là một sự giàu có của các đối tượng sẵn sàng cho sử dụng trong các ứng dụng của Client HTTP, và nó làm cho cảm giác ít để phát minh lại bánh xe. Các máy chủ HTTP trong phần tiếp theo được thực hiện bằng cách sử dụng HTTPWebReqest.

Bắt đầu một dự án mới trong Visual Studio. NET, và kéo trên hai textbox, tbResult và tbUrl. TbResults nên được thiết lập với multiline = true. Một nút, btnCapture cũng nên được thêm vào.



Nhấp vào nút Capture, và nhập vào sau code:

```
private void btnCapture_Click(object sender, EventArgs e)
    {
```

```
        tbResult.Text = getHTTP(tbUrl.Text);
    }
```

Xây dựng hàm getHTTP

```
private string getHTTP(string szURL)
    {
        HttpWebRequest httpRequest;
        HttpWebResponse httpResponse;
        string bodyText = "";
        Stream responseStream;
        Byte[] RecvBytes = new Byte[Byte.MaxValue];
        Int32 bytes;
        httpRequest = (HttpWebRequest)WebRequest.Create(szURL);
        httpResponse = (HttpWebResponse)httpRequest.GetResponse();
        responseStream = httpResponse.GetResponseStream();
        while (true)
        {
            bytes = responseStream.Read(RecvBytes, 0, RecvBytes.Length);
            if (bytes <= 0) break;
            bodyText += System.Text.Encoding.UTF8.GetString(RecvBytes, 0, bytes);
        }
        return bodyText;
    }
```

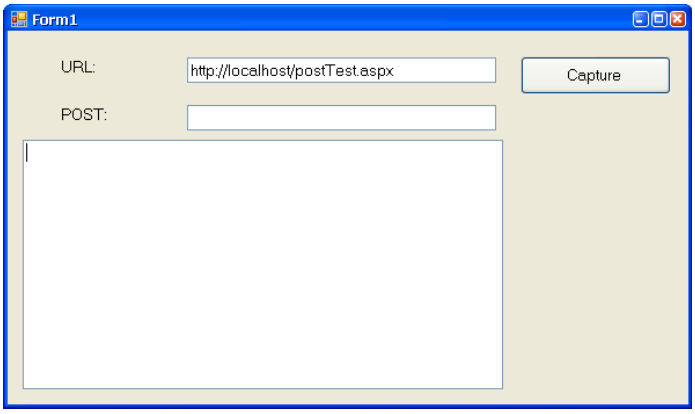
Để có một cái nhìn gần hơn vào mã này, nó cần được tương đối dễ dàng để xác định làm thế nào nó hoạt động. Hành động đầu tiên được thực hiện khi mã này được thực hiện là một phương pháp tĩnh trên lớp WebRequest được gọi và thông qua các szURL chuỗi như một tham số. Điều này tạo ra một đối tượng WebRequest có thể được đúc vào một đối tượng HttpWebRequest, sẽ xử lý các kết nối HTTP đi.

Một khi chúng ta có một đối tượng HttpWebRequest, sau đó chúng tôi có thể gửi yêu cầu HTTP đến máy chủ và bắt đầu nhận dữ liệu từ máy chủ bằng cách gọi phương thức GetResponse. Giá trị trả lại sau đó bỏ một đối tượng HttpWebResponse, mà sau đó được tổ chức tại biến HttpResponse.

Một phản ứng từ một máy chủ Web là không đồng bộ của tự nhiên, do đó, nó là tự nhiên để tạo ra một dòng từ dữ liệu này trở về và đọc nó trong khi nó trở nên có sẵn. Để làm điều này, chúng ta có thể tạo ra một dòng bằng cách gọi phương thức GetResponseStream. Một khi dòng thu được, chúng ta có thể đọc byte từ nó trong khối 256 byte (byte.Max). Đọc dữ liệu trong khối cải thiện hiệu suất. Kích thước đoạn có thể được tự ý lựa chọn, nhưng 256 là hiệu quả.

Mã này nằm trong một vòng lặp vô hạn cho đến khi tất cả các dữ liệu đến
nhận được. Trong một môi trường sản xuất, do đó, loại hành động này nên được chứa trong một chủ đề riêng biệt. Một khi chúng ta có một chuỗi chứa tất cả của HTML, chúng tôi chỉ đơn giản là có thể đổ nó vào màn hình.

Không có chế biến khác yêu cầu. Bạn cũng sẽ cần một số phụ không gian tên:




```
private void btnCapture_Click(object sender, EventArgs e)
        {
            tbPost.Text = HttpUtility.UrlEncode(tbPost.Text);
            tbResult.Text = getHTTP(tbUrl.Text, "tbPost=" + tbPost.Text);
        }
public string getHTTP(string szURL, string szPost)
        {
            HttpWebRequest httprequest;
            HttpWebResponse httpresponse;
            StreamReader bodyreader;
            string bodytext = "";
            Stream responsestream;
            Stream requestStream;
            httprequest = (HttpWebRequest)WebRequest.Create(szURL);
            httprequest.Method = "POST";
            httprequest.ContentType = "application/x-www-form-urlencoded";
            httprequest.ContentLength = szPost.Length;
            requestStream = httprequest.GetRequestStream();
            requestStream.Write(Encoding.ASCII.GetBytes(szPost), 0, szPost.Length);
            requestStream.Close();
            httpresponse = (HttpWebResponse)httprequest.GetResponse();
            responsestream = httpresponse.GetResponseStream();
            bodyreader = new StreamReader(responsestream);
            bodytext = bodyreader.ReadToEnd();
            return bodytext;
        }
```

2.5. Chuyển dữ liệu (Posting data)

Nhiều website động có chứa các hình thức cho các chi tiết đăng nhập, tiêu chí tìm kiếm, hoặc dữ liệu tương tự. Những hình thức thường được gửi thông qua phương thức POST. Này đặt ra một vấn đề, tuy nhiên, đối với bất kỳ ứng dụng mà cần phải truy vấn một trang mà nằm đằng sau hình thức đó vì bạn không thể xác định dữ liệu được đăng trong dòng URL

Đầu tiên, chuẩn bị một trang xử lý các yêu cầu POST. Trong trường hợp này, gõ dòng sau vào một tập tin được gọi là postTest.aspx trong c: \ inetpub \ wwwroot (gốc HTTP của bạn):

```
private void btnCapture_Click(object sender, System.EventArgs e)
                {
                        tbPost.Text = HttpUtility.UrlEncode(tbPost.Text);
                        tbResult.Text =getHTTP(tbUrl.Text,"tbPost="+tbPost.Text);
                }
public string getHTTP(string szURL,string szPost)
{
        HttpWebRequest httprequest;
        HttpWebResponse httpresponse;
        StreamReader bodyreader;
        string bodytext = "";
        Stream responsestream;
        Stream requestStream;
        httprequest = (HttpWebRequest) WebRequest.Create(szURL);
        httprequest.Method = "POST";
        httprequest.ContentType =
        "application/x-www-form-urlencoded";
        httprequest.ContentLength = szPost.Length;
        requestStream = httprequest.GetRequestStream();
        requestStream.Write(Encoding.ASCII.GetBytes(szPost),0,szPost.Length);
        requestStream.Close();
        httpresponse = (HttpWebResponse) httprequest.GetResponse();
        responsestream = httpresponse.GetResponseStream();
        bodyreader = new StreamReader(responsestream);
        bodytext = bodyreader.ReadToEnd();
        return bodytext;
}
```

2.6. Chú ý khi làm việc với cookies

HTTP không duy trì thông tin trạng thái. Do đó, khó khăn để phân biệt giữa hai người dùng truy cập vào một máy chủ hoặc một người sử dụng thực hiện hai yêu cầu. Từ quan điểm của máy chủ, có thể cho cả người dùng có cùng địa chỉ IP (ví dụ: , Nếu họ đều đi qua cùng một máy chủ proxy). Nếu dịch vụ này đang được truy cập chứa thông tin cá nhân người dùng mà này gắn liền dữ liệu được quyền hợp pháp để xem dữ liệu này, nhưng người dùng khác không nên được phép truy cập. Trong tình huống này, phía Client kết nối cần để phân biệt từ các Client khác. Điều này có thể được thực hiện theo nhiều cách, nhưng cho các trang web, cookies là giải pháp tốt nhất.

Cookies là các tập tin nhỏ được lưu trữ trong c: \ windows \ cookies (tùy thuộc vào cài đặt Windows của bạn). Chúng được đặt trong một trong hai cách:

đối tượng document.cookie JavaScript, hoặc bằng cách thiết lập các tiêu đề cookie trong yêu cầu HTTP. Những cookie này vẫn còn trên máy tính của

khách hàng trong một thời gian quy định và có thể được lấy trong JavaScript hoặc trong các đáp ứng HTTP cookie được hỗ trợ trong NET thông qua HttpWebResponse.Cookies và các đối tượng HttpWebRequest.CookieContainer

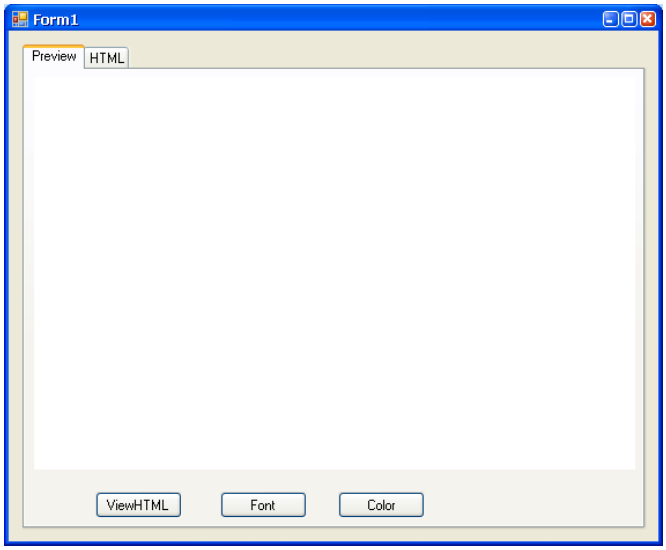
Cookie là những tên miền cụ thể;. Do đó, một cookie được lưu trữ trên www.library. com không có thể được lấy bởi www.bookshop.com. Trong trường hợp cả hai trang web được liên kết với nhau, hai trang web có thể cần phải chia sẻ thông tin trạng thái phiên. Trong ví dụ này, nó sẽ là thuận lợi cho các hiệu sách. com để biết sở thích của người dùng đọc, để nó có thể quảng cáo các relevanttitles nhất Các thủ thuật để sao chép các tập tin cookie trên các lĩnh vực. để chuyển đổi các tập tin cookie vào văn bản, thông qua văn bản giữa các máy chủ, và thông qua các tập tin cookie trở lại cho Client từ các máy chủ nước ngoài. NET cung cấp một cơ sở để serialize cookies, đó là lý tưởng cho mục đích này.

2.7. A WYSIWYG editor

WYSIWYG (những gì bạn thấy là những gì bạn nhận được) là một thuật ngữ dùng để mô tả Web và đồ họa biên tập cho phép bạn tự nhiên thao tác sản lượng đồ họa, mà không cần phải được quan tâm với các mã cơ bản. Tính năng này là một cách tiện dụng để cho phép người dùng có nhiều sáng tạo trong các loại hình tin nhắn văn bản, tài liệu mà họ tạo ra, mà không yêu cầu họ thực hiện một khóa học sụp đổ trong HTML. Internet Explorer có thể chạy trong một chế độ thiết kế đặc biệt, đó là chấp nhận được một trình soạn thảo WYSIWYG. Lừa để truy cập vào chế độ thiết kế trong Internet Explorer chỉ đơn giản là để thiết lập WebBrowser.Document.designMode tài sản để On. Người dùng có thể gõ trực tiếp vào cửa sổ Internet Explorer và sử dụng phím tắt wellknown định dạng văn bản (ví dụ như Ctrl + B, Bold, Ctrl + I, Italic, Ctrl + U, gạch dưới). Bằng cách kích chuột phải vào Internet Explorer trong chế độ thiết kế, người dùng có thể bao gồm hình ảnh, thêm các siêu liên kết, và chuyển sang chế độ trình duyệt. Khi một hình ảnh được bao gồm trong giao diện thiết kế, nó có thể được di chuyển và thu nhỏ bằng cách nhấp và kéo trên các cạnh của hình ảnh.

Những tính năng tiên tiến hơn có thể được truy cập thông qua chức năng của Internet Explorer execCommand. Chỉ FontName, FontSize, và ForeColor được sử dụng trong các chương trình mẫu sau đây, nhưng đây là một danh sách các lệnh được sử dụng bởi Internet Explorer.

Ví dụ




```
private void Form1_Load(object sender, EventArgs e)
    {
      //object any = null;
      string url = "about:blank";
      WebBrowser.Navigate(url, null);
      Application.DoEvents();
      ((HTMLDocument)WebBrowser.Document).designMode = "On";
      HtmlDocument oDoc = WebBrowser.Document;
      HTMLDocument oDocH = GetODocH(oDoc);
    }
private void btnViewHTML_Click(object sender, EventArgs e)
    {
      tbHTML.Text = ((HTMLDocument)WebBrowser.Document).body.innerHTML;
    }

 private void btnPreview_Click(object sender, EventArgs e)
    {
      ((HTMLDocument)WebBrowser.Document).body.innerHTML = tbHTML.Text;
    }
private void btnFont_Click(object sender, EventArgs e)
    {
      fontDialog.ShowDialog();
      HTMLDocument doc = (HTMLDocument)WebBrowser.Document;
      object selection = doc.selection.createRange();
      doc.execCommand("FontName", false,
      fontDialog.Font.FontFamily.Name);
      doc.execCommand("FontSize", false, fontDialog.Font.Size);
      ((IHTMLTxtRange)selection).select();
    }
private void btnColor_Click(object sender, EventArgs e)
    {
      colorDialog.ShowDialog();
      string colorCode = "#" +
        toHex(colorDialog.Color.R) +
        toHex(colorDialog.Color.G) +
        toHex(colorDialog.Color.B);
```

```
        HTMLDocument doc = (HTMLDocument)WebBrowser.Document;
        object selection = doc.selection.createRange();
        doc.execCommand("ForeColor", false, colorCode);
        ((IHTMLTxtRange)selection).select();
    }

private string toHex(int digit)
    {
        string hexDigit = digit.ToString("X");
        if (hexDigit.Length == 1)
        {
            hexDigit = "0" + hexDigit;
        }
        return hexDigit;
    }
```

3. Máy chủ Web (Web servers)

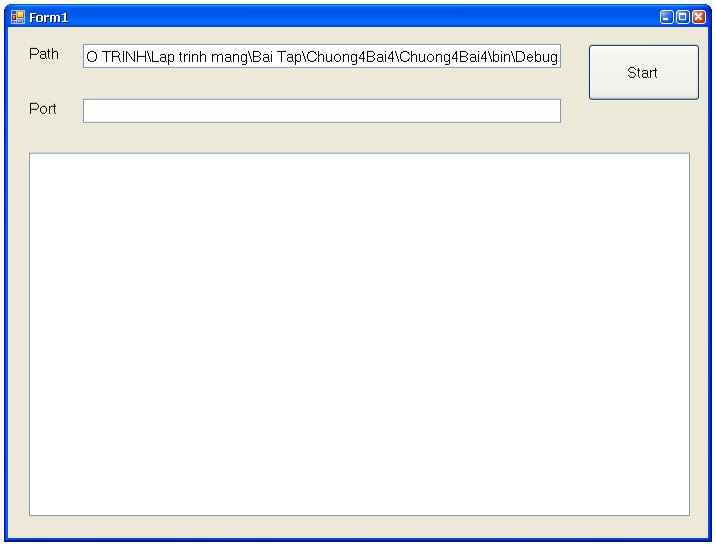
Ta có thể hỏi lý do tại sao bạn nên phát triển một máy chủ trong NET khi IIS là tự do có sẵn. Một máy chủ trong nhà phát triển có một số lợi thế, chẳng hạn như sau đây:

+ Máy chủ Web có thể được cài đặt như một phần của một ứng dụng, mà không đòi hỏi người sử dụng phải cài đặt IIS bằng tay từ đĩa CD cài đặt Windows.

+ IIS sẽ không cài đặt trên Windows XP Home Edition, chiếm một phần đáng kể người dùng Windows

3.1 Thực thi một máy chủ Web

Bắt đầu một mới Visual Studio. Dự án NET như bình thường. Vẽ hai textbox, tbPath và tbPort, vào biểu mẫu, theo sau bằng một nút, btnStart, và một hộp danh sách tên là lbConnections, trong đó có xem nó vào danh sách. Tại trung tâm của một máy chủ HTTP là một máy chủ TCP, và bạn có thể nhận thấy một trùng mã giữa các ví dụ này và máy chủ TCP trong chương trước. Các máy chủ đã được đa luồng, vì vậy bước đầu tiên là khai báo một danh sách mảng để chứa:




```
public partial class Form1 : Form
        {
                private ArrayList alSockets;
                ……..
        }
```

Thực thi-Path, mà trả về không chỉ con đường mà còn tên tập tin, và do đó chúng tôi có thể cắt bỏ tất cả các ký tự sau khi người cuối cùng dấu gạch chéo ngược Mỗi máy chủ HTTP có một gốc rễ HTTP, mà là một đường dẫn đến một thư mục trên đĩa cứng của bạn mà từ đó các máy chủ sẽ lấy các trang Web. IIS có một gốc HTTP mặc định C: \inetpub\wwwroot, trong trường hợp này, chúng ta sẽ sử dụng đường dẫn trong đó ứng dụng được lưu. Để có được đường dẫn ứng dụng, chúng ta có thể sử dụng ứng dụng. Thực thi-Path, mà trả về không chỉ con đường mà còn tên tập tin, và do đó chúng tôi có thể cắt bỏ tất cả các ký tự sau khi người cuối cùng dấu gạch chéo ngược.

```
        private void Form1_Load(object sender, EventArgs e)
    {
        tbPath.Text = Application.ExecutablePath;
        // trim off filename, to get the path
        tbPath.Text = tbPath.Text.Substring(0, tbPath.Text.LastIndexOf("\\"));
    }

    private void btnStart_Click(object sender, EventArgs e)
    {
        alSockets = new ArrayList();
        Thread thdListener = new Thread(new ThreadStart(listenerThread));
        thdListener.Start();
    }

    public void listenerThread()
    {
        int port = 0;
        port = Convert.ToInt16(tbPort.Text);
        TcpListener tcpListener = new TcpListener(port);
        tcpListener.Start();
```

```
    while (true)
    {
        Socket handlerSocket = tcpListener.AcceptSocket();
        if (handlerSocket.Connected)
        {
            lbConnections.Items.Add(handlerSocket.RemoteEndPoint.ToString() +
                " connected.");
            lock (this)
            {
                alSockets.Add(handlerSocket);
                ThreadStart thdstHandler = new
                ThreadStart(handlerThread);
                Thread thdHandler = new Thread(thdstHandler);
                thdHandler.Start();
            }
        }
    }
}

public void handlerThread()
{
    Socket handlerSocket = (Socket)alSockets[alSockets.Count - 1];
    String streamData = "";
    String filename = "";
    String[] verbs;
    StreamReader quickRead;
    NetworkStream networkStream = new NetworkStream(handlerSocket);
    quickRead = new StreamReader(networkStream);
    streamData = quickRead.ReadLine();
    verbs = streamData.Split(" ".ToCharArray());
    // Assume verbs[0]=GET
    filename = verbs[1].Replace("/", "\\");
    if (filename.IndexOf("?") != -1)
    {
        // Trim of anything after a question mark (Querystring)
        filename = filename.Substring(0, filename.IndexOf("?"));
    }

    if (filename.EndsWith("\\"))
    {
        // Add a default page if not specified
        filename += "index.htm";
    }
    filename = tbPath.Text + filename;
    FileStream fs = new FileStream(filename, FileMode.OpenOrCreate);
    fs.Seek(0, SeekOrigin.Begin);
    byte[] fileContents = new byte[fs.Length];
    fs.Read(fileContents, 0, (int)fs.Length);
    fs.Close();
    // optional: modify fileContents to include HTTP header.
```

```
        handlerSocket.Send(fileContents);
        lbConnections.Items.Add(filename);
        handlerSocket.Close();
    }
```

4. Làm việc với lớp System.Net.HttpWebListener

Trong NET 2. Whidbey, một giải pháp nhẹ nhàng hơn cho việc thực hiện các máy chủ Web tồn tại, cụ thể là lớp HttpWebListener. Lớp này thúc đẩy các Http.sys điều khiển (nếu có) để cung cấp hiệu suất chưa từng có, và tích hợp nhiều tính năng, chẳng hạn như mã hóa SSL và xác thực, đó sẽ là khó khăn để phát triển từ mặt đất lên.

Lớp HttpWebListener bao gồm các phương thức quan trọng và thuộc tính thể hiện trong Bảng 4.7.

Phương thức hoặc thuộc tính	Ý nghĩa
Abort / Close	Hủy hàng đợi yêu cầu
AddPrefix	Thêm một tiền tố để lắng nghe Web
BeginGetRequest	Đang chờ đợi một yêu cầu khách hàng không đồng bộ. Trả về IAsyncResult.
EndGetRequest	Xử lý yêu cầu khách hàng. Trả về ListenerWebRequest.
GetPrefixes	Lấy tất cả tiền tố xử lý. Trả về String []
GetRequest	Đang chờ đợi một yêu cầu khách hàng đồng bộ. Trả về ListenerWebRequest.
RemoveAll	Loại bỏ tất cả các tiền tố
RemovePrefix	Loại bỏ một tiền tố quy định
Start	Bắt đầu thực thi Web Server
Stop	Dừng Web Server
AuthenticationScheme	Thiết lập những phương tiện mà máy chủ xác thực khách hàng. Trả về AuthenticationScheme (tức là, Basic, Digest, NTLM).
IsListening	Xác định nếu máy chủ đang chạy. Trả về Boolean.
Realm string	Nếu hệ thống xác thực Basic hoặc Digest được lựa chọn, được chỉ thị lĩnh vực. Trả về String.

5. Trình duyệt Web di động (Mobile Web browsers)

Không phải tất cả các Client HTTP là máy tính. Nhiều người sử dụng điện thoại di động của họ truy cập Internet. Một số ứng dụng vô cùng hữu ích hơn khi có sẵn không dây. Mặc dù phà dữ liệu điện thoại di động trong một cách hoàn toàn khác nhau từ các mạng có dây, một ứng dụng giao thức không dây (WAP) điện thoại sẽ giao tiếp thông qua một cổng WAP, chuyển

đổi tín hiệu điện thoại di động vào giao thức TCP/IP và truy cập các máy chủ trong cách tương tự như các trình duyệt. WAP chạy trên HTTP và giao thức truyền không dây (WTP), với một vài thêm tiêu đề ném vào các yêu cầu HTTP. Sau đây là một mẫu yêu cầu HTTP được tạo ra bởi một WAP điện thoại:

```
GET / HTTP/1.1
Accept-Charset: ISO-8859-1
Accept-Language: en
Content-Type: application/x-www-form-urlencoded
x-up-subno: Fiach_hop
x-upfax-accepts: none
x-up-uplink: none
x-up-devcap-smartdialing: 1
x-up-devcap-screendepth: 1
x-up-devcap-iscolor: 0
x-up-devcap-immed-alert: 1
x-up-devcap-numsoftkeys: 3
x-up-devcap-screenchars: 15,4
Accept: application/x-hdmlc, application/x-up-alert,
application/x-up-cacheop, application/x-up-device,
application/x-up-digestentry, text/x-hdml;version=3.1, text/
x-hdml;version=3.0, text/x-hdml;version=2.0, text/x-wap.wml,
text/vnd.wap.wml, */*, image/bmp, text/html
User-Agent: UP.Browser/3.1-ALAV UP.Link/3.2
Host: 127.0.0.1:50
```

BÀI 5 : TRUYỀN THÔNG VỚI EMAIL SERVERS

Mã bài MĐ35.5

Mục tiêu của bài:

- Trình bày được cách sử dụng các lớp trong lập trình với Mail Server
- Xây dựng ứng dụng Mail.
- Thực hiện các thao tác an toàn với máy tính.

1. Phương thức gởi và nhận Email

Để nhận được thư điện tử bạn cần phải có một tài khoản (account) thư điện tử. Nghĩa là bạn phải có một địa chỉ để nhận thư. Một trong những thuận lợi hơn với thư thông thường là bạn có thể nhận thư điện tử từ bất cứ đâu. Bạn chỉ cần kết nối vào Server thư điện tử để lấy thư về máy tính của mình.

Để gửi được thư bạn cần phải có một kết nối vào internet và truy nhập vào máy chủ thư điện tử để chuyển thư đi. Thủ tục tiêu chuẩn được sử dụng để gửi thư là **SMTP** (Simple Mail Transfer Protocol). Nó được kết hợp với thủ tục **POP** (Post Office Protocol) và **IMAP** (Internet Message Access Protocol) để lấy thư.

Trên thực tế có rất nhiều hệ thống vi tính khác nhau và mỗi hệ thống lại có cấu trúc chuyển nhận thư điện tử khác nhau. Vì có sự khác biệt như vậy nên việc chuyển nhận thư điện tử giữa hai hệ thống khác nhau rất là khó khăn và bất tiện. Do vậy, người ta đã đặt ra một nghi thức chung cho thư điện tử. Có nghĩa là các hệ thống máy vi tính đều đồng ý với nhau về một nghi thức chung gọi là Simple Mail Transfer Protocol viết tắt là SMTP. Nhờ vào SMTP này mà sự chuyển vận thư từ điện tử trên Internet đã trở thành dễ dàng nhanh chóng cho tất cả các người sử dụng máy vi tính cho dù họ có sử dụng hệ thống máy vi tính khác nhau.

Khi gửi thư điện tử thì máy tính của bạn cần phải định hướng đến máy chủ SMTP. Máy chủ sẽ tìm kiếm địa chỉ thư điện tử (tương tự như địa chỉ điền trên phong bì) sau đó chuyển tới máy chủ của người nhận và nó được chứa ở đó cho đến khi được lấy về. Bạn có thể gửi thư điện tử đến bất cứ ai trên thế giới mà có một địa chỉ thư điện tử. Hầu hết các nhà cung cấp dịch vụ Internet đều cung cấp thư điện tử cho người dùng internet.

Chuyển thư (Send Mail)

Sau khi người sử dụng máy vi tính dùng chương trình thư để viết thư và đã ghi rõ địa chỉ của người nhận thì máy tính sẽ chuyển bức thư điện đến hộp thư người nhận. SMTP sử dụng nghi thức TCP (TCP protocol) để chuyển vận thư. Vì nghi thức TCP rất hữu hiệu và có phần kiểm soát thất lạc mất mát cho nên việc gửi thư điện có hiệu suất rất cao. Khi nhận được mệnh lệnh gửi đi của người sử dụng, máy vi tính sẽ dùng nghi thức

TCP liên lạc với máy vi tính của người nhận để chuyển thư. Đôi khi vì máy vi tính của người nhận đã bị tắt điện hoặc đường dây kết nối từ máy gửi tới máy nhận đã bị hư hỏng tạm thời tại một nơi nào đó (tranmission wire failure), hoặc là có thể là Máy Chuyển Tiếp (routers) trên tuyến đường liên lạc giữa hai máy tạm thời bị hư (out of order) thì máy gửi không cách nào liên lạc với máy nhận được. Gặp trường hợp như vậy thì máy gửi sẽ tạm thời giữ lá thư trong khu vực dự trữ tạm thời. Máy gửi sau đó sẽ tìm cách liên lạc với máy nhận để chuyển thư. Những việc này xảy ra trong máy vi tính và người sử dụng sẽ không hay biết gì. Nếu trong khoảng thời gian mà máy vi tính của nơi gửi vẫn không liên lạc được với máy nhận thì máy gửi sẽ gửi một thông báo cho người gửi nói rằng việc vận chuyển của lá thư điện đã không thành công.

Nhận Thư (Receive Mail)

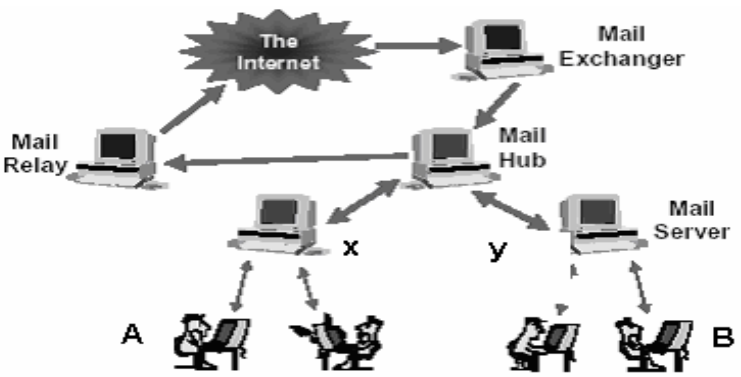
Nếu máy gửi có thể liên lạc được với máy nhận thì việc chuyển thư sẽ được tiến hành. Trước khi nhận lá thư thì máy nhận sẽ kiểm soát tên người nhận có hộp thư trên máy nhận hay không. Nếu tên người nhận thư có hộp thư trên máy nhận thì lá thư sẽ được nhận lấy và thư sẽ được bỏ vào hộp thư của người nhận. Trường hợp nếu máy nhận kiểm soát thấy rằng tên người nhận không có hộp thư thì máy nhận sẽ khước từ việc nhận lá thư. Trong trường hợp khước từ này thì máy gửi sẽ thông báo cho người gửi biết là người nhận không có hộp thư (user unknown).

Sau khi máy nhận đã nhận lá thư và đã bỏ vào hộp thư cho người nhận thì máy nhận sẽ thông báo cho người nhận biết là có thư mới . Người nhận sẽ dùng chương trình thư để xem lá thư. Sau khi xem thư xong thì người nhận có thể lưu trữ (save), hoặc xóa (delete), hoặc trả lời (reply) v.v... Trường hợp nếu người nhận muốn trả lời lại lá thư cho người gửi thì người nhận không cần phải ghi lại địa chỉ vì địa chỉ của người gửi đã có sẵn trong lá thư và chương trình thư sẽ bỏ địa chỉ đó vào trong bức thư trả lời.

Trạm Phục Vụ Thư (Mail Server)

Trên thực tế, trong những cơ quan và hãng xưởng lớn, máy vi tính của người gửi thư không gửi trực tiếp tới máy vi tính của người nhận mà thường qua các máy chủ thư điện tử (mail servers).

Ví dụ: quá trình gửi thư



Hình : Gửi thư từ A đến B

Như hình trên cho thấy, nếu như một người ở máy A gửi tới một người ở máy B một lá thư thì trước nhất máy A sẽ gửi đến máy chủ thư điện tử X. Khi trạm phục vụ thư X nhận được thư từ máy A thì X sẽ chuyển tiếp cho máy chủ thư điện tử Y. Khi trạm phục vụ thư Y nhận được thư từ X thì Y sẽ chuyển thư tới máy B là nơi người nhận. Trường hợp máy B bị trục trặc thì máy chủ thư Y sẽ giữ thư.

Thông thường thì máy chủ thư điện tử thường chuyển nhiều thư cùng một lúc cho một máy nhận. Như ví dụ ở trên trạm phục vụ thư Y có thể chuyển nhiều thư cùng một lúc cho máy B từ nhiều nơi gửi đến.

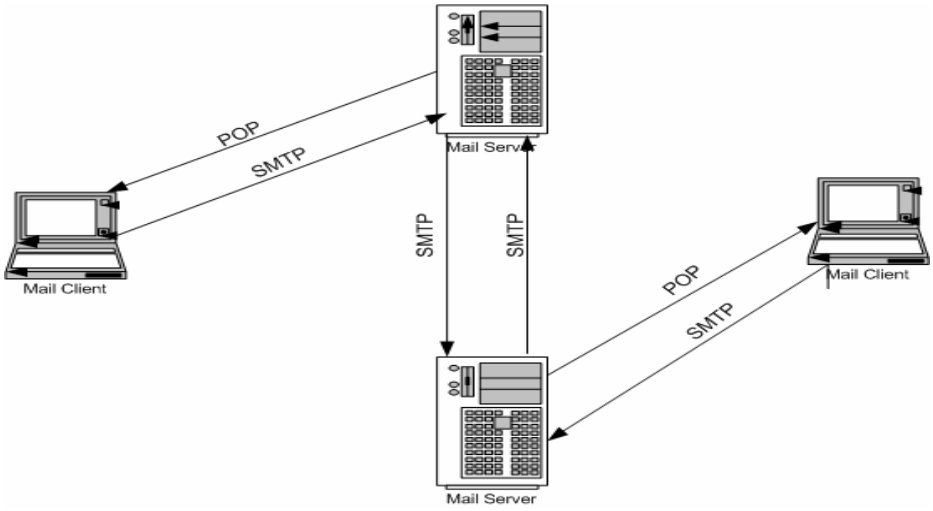
Một vài công dụng khác của máy chủ thư là khi người sử dụng có chuyện phải nghỉ một thời gian thì người sử dụng có thể yêu cầu máy chủ thư giữ giùm tất cả những thư từ trong thời gian người sử dụng vắng mặt hoặc có thể yêu cầu máy chủ thư chuyển tất cả thư từ tới một cái hộp thư khác.

Với những thông tin trên chúng ta đã có một cái nhìn khái quát về những chức năng và hoạt động của hệ thống thư điện tử.

2. SMTP

Công việc phát triển các hệ thống thư điện tử (Mail System) đòi hỏi phải hình thành các chuẩn chung về thư điện tử. Điều này giúp cho việc gửi, nhận các thông điệp được đảm bảo, làm cho những người ở các nơi khác nhau có thể trao đổi thông tin cho nhau.

Có 2 chuẩn về thư điện tử quan trọng nhất và được sử dụng nhiều nhất từ trước đến nay là X.400 và SMTP (Simple Mail Transfer Protocol). SMTP thường đi kèm với chuẩn POP3. Chuẩn SMTP miêu tả cách điều khiển các thông điệp trên mạng Internet. Điều quan trọng của chuẩn SMTP là giả định máy nhận phải dùng giao thức SMTP gửi thư điện tử cho một máy chủ luôn luôn hoạt động. Sau đó, người nhận sẽ đến lấy thư từ máy chủ khi nào họ muốn dùng giao thức POP (Post Office Protocol), ngày nay POP được cải tiến thành POP3 (Post Officce Protocol version 3).



Hình : Hoạt động của POP và SMTP

Thủ tục chuẩn trên Internet để nhận và gửi của thư điện tử là SMTP (Simple Mail Transport Protocol). SMTP là thủ tục phát triển ở mức ứng dụng trong mô hình 7 lớp OSI cho phép gửi các bức điện trên mạng TCP/IP. SMTP được phát triển vào năm 1982 bởi tổ chức IETF (Internet Engineering Task Force) và được chuẩn hoá theo tiêu chuẩn RFCs 821 và 822. SMTP sử dụng cổng 25 của TCP.

Mặc dù SMTP là thủ tục gửi và nhận thư điện tử phổ biến nhất nhưng nó vẫn còn thiếu một số đặc điểm quan trọng có trong thủ tục X400. Phần yếu nhất của SMTP là thiếu khả năng hỗ trợ cho các bức điện không phải dạng Text.

Ngoài ra SMTP cũng có kết hợp thêm hai thủ tục khác hỗ trợ cho việc lấy thư là POP3 và IMAP4.

MIME và SMTP

MIME (Multipurpose Internet Mail Extensions) cung cấp thêm khả năng cho SMTP và cho phép các file có dạng mã hoá đa phương tiện (multimedia) đi kèm với bức điện SMTP chuẩn. MIME sử dụng bảng mã Base64 để chuyển các file dạng phức tạp sang mã ASCII để chuyển đi.

MIME là một tiêu chuẩn mới như nó hiện đã được hỗ trợ bởi hầu hết các ứng dụng, và bạn phải thay đổi nếu chương trình thư điện tử của bạn không có hỗ trợ MIME. MIME được quy chuẩn trong các tiêu chuẩn RFC 2045-2049

Lệnh của SMTP

SMTP sử dụng một cách đơn giản các câu lệnh ngắn để điều khiển bức điện. Bảng ở dưới là danh sách các lệnh của SMTP

Các lệnh của SMTP được xác định trong tiêu chuẩn RFC 821

Lệnh	***Mô tả***

HELO	Hello. Sử dụng để xác định người gửi điện. Lệnh này này đi kèm với tên của host gửi điện. Trong ESTMP (extended protocol), thì lệnh này sẽ là EHLO.
MAIL	Khởi tạo một giao dịch gửi thư. Nó kết hợp "from" để xác định người gửi thư.
RCPT	Xác định người nhận thư.
DATA	Thông báo bắt đầu nội dung thực sự của bức điện (phần thân của thư). Dữ liệu được mã thành dạng mã 128-bit ASCII và nó được kết thúc với một dòng đơn chứa dấu chấm (.).
RSET	Huỷ bỏ giao dịch thư
VRFY	Sử dụng để xác thực người nhận thư.
NOOP	Nó là lệnh "no operation" xác định không thực hiện hành động gì
QUIT	Thoát khỏi tiến trình để kết thúc
SEND	Cho host nhận biết rằng thư còn phải gửi đến đầu cuối khác.
Sau đây là những lệnh khác nhưng không yêu cầu phải có. Xác định bởi RFC	
SOML	Send or mail. Báo với host nhận thư rằng thư phải gửi đến đầu cuối khác hoặc hộp thư.
SAML	Send and mail. Nói với host nhận rằng bức điện phải gửi tới người dùng đầu cuối và hộp thư.
EXPN	Sử dụng mở rộng cho một mailing list.
HELP	Yêu cầu thông tin giúp đỡ từ đầu nhận thư.
TURN	Yêu cầu để host nhận giữ vai trò là host gửi thư.

Các lệnh của SMTP rất đơn giản. Bạn có thể nhìn thấy điều đó ở ví dụ sau:

220 receivingdomain.com

Server ESMTP Sendmail 8.8.8+Sun/8.8.8; Fri, 30 Jul 1999 09:23:01

HELO host.sendingdomain.com

250 receivingdomain.com Hello host, pleased to meet you.

MAIL FROM:

250 Sender ok. RCPT

TO:

250 Recipient ok. DATA

354 Enter mail, end with a ì.î on a line by itself

Here goes the message.

.

250 Message accepted for delivery
QUIT
221 Goodbye host.sendingdomain.com

Và bức thư sẽ trông như sau:
From username@sendingdomain.com Fri Jul 30 09:23:39 1999
Date: Fri, 30 Jul 1999 09:23:15 -0400 (EDT)
From: username@sendingdomain.com Message-
Id:
Content-Length: 23
Here goes the message.

Mã trạng thái của SMTP

Khi một MTA gửi một lệnh SMTP tới MTA nhận thì MTA nhận sẽ trả lời với một mã trạng thái để cho người gửi biết đang có việc gì xảy ra tại đầu nhận. Và dưới đây là bảng mã trạng thái của SMTP theo tiêu chuẩn RFC 821. Mức độ của trạng thái được xác định bởi số đầu tiên của mã (5xx là lỗi nặng, 4xx là lỗi tạm thời, 1xx–3xx là hoạt động bình thường).

SMTP mở rộng (Extended SMTP)

SMTP thì được cải tiến để ngày càng đáp ứng nhu cầu cao của người dùng và là một thủ tục ngày càng có ích. Nhưng dù sao cũng cần có sự mở rộng tiêu chuẩn SMTP, và chuẩn RFC 1869 ra đời để bổ sung cho SMTP. Nó không chỉ mở rộng mà còn cung cấp thêm các tính năng cần thiết cho các lệnh có sẵn. Ví dụ: lệnh SIZE là lệnh mở rộng cho phép nhận giới hạn độ lớn của bức điện đến. Không có ESMTP thì sẽ không giới hạn được độ lớn của bức thư.
Khi hệ thống kết nối với một MTA, nó sẽ sử dụng khởi tạo thì ESMTP thay HELO bằng EHLO. Nếu MTA có hỗ trợ SMTP mở rộng (ESMTP) thì nó sẽ trả lời với một danh sách các lệnh mà nó sẽ hỗ trợ. Nếu không nó sẽ trả lời với mã lệnh sai (500 Command not recognized) và host gửi sẽ quay trở về sử dụng SMTP. Sau đây là một tiến trình ESMTP:

220 esmtpdomain.com
Server ESMTP Sendmail 8.8.8+Sun/8.8.8; Thu, 22 Jul 1999 09:43:01
EHLO host.sendingdomain.com
250-mail.esmtpdomain.com Hello host, pleased to meet you
250-EXPN
250-VERB
250-8BITMIME
250-SIZE
250-DSN

250-ONEX

250-ETRN

250-XUSR

250 HELP QUIT

221 Goodbye host.sendingdomain.com

Các lệnh cơ bản của ESMTP

Lệnh	Miêu tả
EHLO	Sử dụng ESMTP thay cho HELO của
8BITMIME	Sử dụng 8-bit MIME cho mã dữ liệu
SIZE	Sử dụng giới hạn độ lớn của bức điện

SMTP Headers

Có thể lấy được rất nhiều thông tin có ích bằng cách kiểm tra phần header của thư. Không chỉ xem được bức điện từ đâu đến, chủ đề của thư, ngày gửi và những người nhận. Bạn còn có thể xem được những điểm mà bức điện đã đi qua trước khi đến được hộp thư của bạn. Tiêu chuẩn RFC 822 quy định header chứa những gì. Tối thiểu có người gửi (from), ngày gửi và người nhận (TO, CC, hoặc BCC)

Header của thư khi nhận được cho phép bạn xem bức điện đã đi qua những đâu trước khi đến hộp thư của bạn. Nó là một dụng cụ rất tốt để kiểm tra và giải quyết lỗi. Sau đây là ví dụ:

From someone@mydomain.COM Sat Jul 31 11:33:00 1999

Received: from host1.mydomain.com by host2.mydomain.com

(8.8.8+Sun/8.8.8)

with ESMTP id LAA21968 for ;

Sat, 31 Jul 1999 11:33:00 -0400 (EDT)

Received: by host1.mydomain.com with Internet Mail Service(5.0.1460.8)

id ; Sat, 31 Jul 1999 11:34:39 -0400

Message-ID:

From: "Your Friend"

To: "'jamisonn@host2.mydomain.com'"

Subject: Hello There

Date: Sat, 31 Jul 1999 11:34:36 -0400

Trên ví dụ trên có thể thấy bức điện được gửi đi từ someone@mydomain.com. Từ mydomain.com, nó được chuyển đến host1.

Bức điện được gửi từ host2 tới host1 và chuyển tới người dùng. Mỗi chỗ bức điện dừng lại thì host nhận được yêu cầu điền thêm thông tin vào header nó bao gồm ngày giờ tạm dừng ở đó. Host2 thông báo rằng nó nhận được điện lúc 11:33:00. Host1 thông báo rằng nó nhận được bức điện vào lúc 11:34:36, Sự chênh lệch hơn một phút có khả năng là do sự không đồng bộ giữa đồng hồ của hai nơi.

Thuận lợi và bất lợi của SMTP

Như thủ tục X.400, SMTP có một số thuận lợi và bất lợi

Thuận lợi bao gồm:

- SMTP rất phổ biến.
- Nó được hỗ trợ bởi nhiều tổ chức.
- SMTP có giá thành quản trị và duy trì thấp.
- SMTP nó có cấu trúc địa chỉ đơn giản.

Bất lợi bao gồm:

- SMTP thiếu một số chức năng
- SMTP thiếu khả năng bảo mật như X.400.
- Nó chỉ giới hạn vào những tính năng đơn giản nhất.

3. POP3

Trong nhưng ngày tháng đầu tiên của thư điện tử, người dùng được yêu cầu truy nhập vào máy chủ thư điện tử và đọc các bức điện của họ ở đó. Các chương trình thư thường sử dụng dạng text và thiếu khả năng thân thiện với người dùng. Để giải quyết vấn đề đó một số thủ tục được phát triển để cho phép người dùng có thể lấy thư về máy của họ hoặc có các giao diện sử dụng thân thiện hơn với người dùng. Và chính điều đó đem đến sự phổ biến của thư điện tử.

Có hai thủ tục được sử dụng phổ biến nhất để lấy thư về hiện nay là POP (Post Office Protocol) và IMAP (Internet Mail Access Protocol).

Post Office Protocol (POP)

POP cho phép người dùng có account tại máy chủ thư điện tử kết nối vào MTA và lấy thư về máy tính của mình, ở đó có thể đọc và trả lời lại. POP được phát triển đầu tiên là vào năm 1984 và được nâng cấp từ bản POP2 lên POP3 vào năm 1988. Và hiện nay hầu hết người dùng sử dụng tiêu chuẩn POP3. POP3 kết nối trên nền TCP/IP để đến máy chủ thư điện tử (sử dụng cổng 110). Người dùng điền username và password. Sau khi xác thực đầu máy khách sẽ sử dụng các lệnh của POP3 để lấy hoặc xoá thư.

POP3 chỉ là thủ tục để lấy thư trên máy chủ thư điện tử. POP3 được quy định bởi tiêu chuẩn RFC 1939.

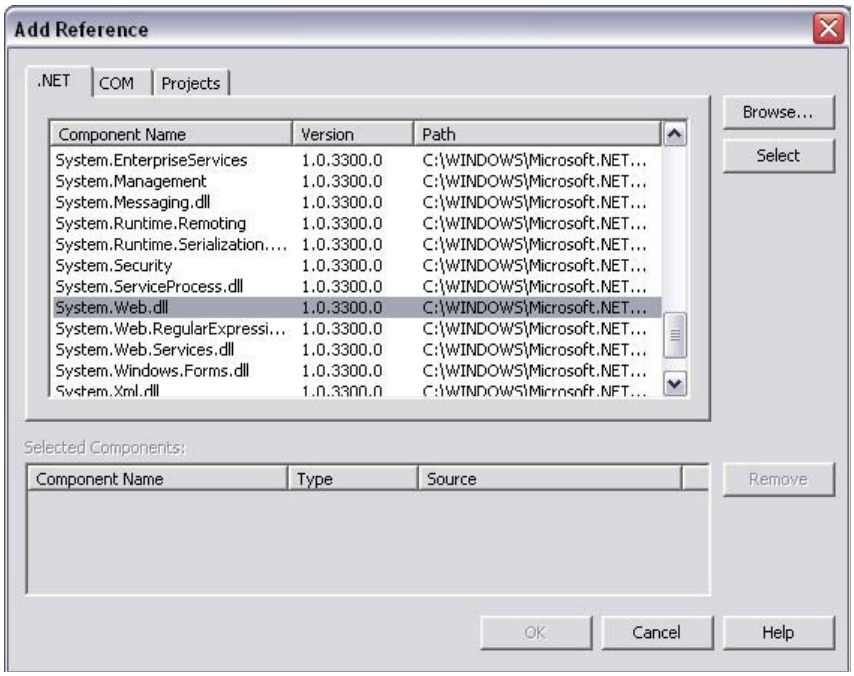
Lệnh của POP3

Lệnh	Miêu tả
USER	Xác định username

PASS	Xác định password
STAT	Yêu cầu về trạng thái của hộp thư như số
LIST	Hiện danh sách của thư
RETR	Nhận thư
DELE	Xoá một bức thư xác định
NOOP	Không làm gì cả
RSET	Khôi phục lại những thư đã xoá
QUIT	Thực hiện việc thay đổi và thoát ra

4. Làm việc với lớp System.Web.Mail

Thực hiện một tham chiếu đến System.Web.dll trước khi bạn có thể nhập không gian tên System.Web.Mail. DLL này là một thành phần của .NET, chứ không phải .COM. Để làm như vậy, Project→Add Reference, và sau đó click vào DLL



5. Xây dựng ứng dụng Mail

Viết chương trình cho phép gởi một email đơn giản từ địa chỉ source@here.com đến destination@there.com thong qua SMTP server smtp.ntl- world.com (có thể thay đổi SMTP server tuỳ ý).

HƯỚNG DẪN THỰC HIỆN BÀI TẬP ỨNG DỤNG

Viết ứng dụng gởi file trên máy client: TCP Simple Client

1. Thiết kế giao diện như sau

2. Viết code cho sự kiện gởi Send

```
private void btnSend_Click(object sender,
System.EventArgs e)
{
MailMessage email = new MailMessage();
email.From = tbFrom.Text;
email.To = tbTo.Text;
email.Subject = "email from .NET"; email.Body =
tbMessage.Text; SmtpMail.SmtpServer = tbServer.Text;
SmtpMail.Send(email);
}
```

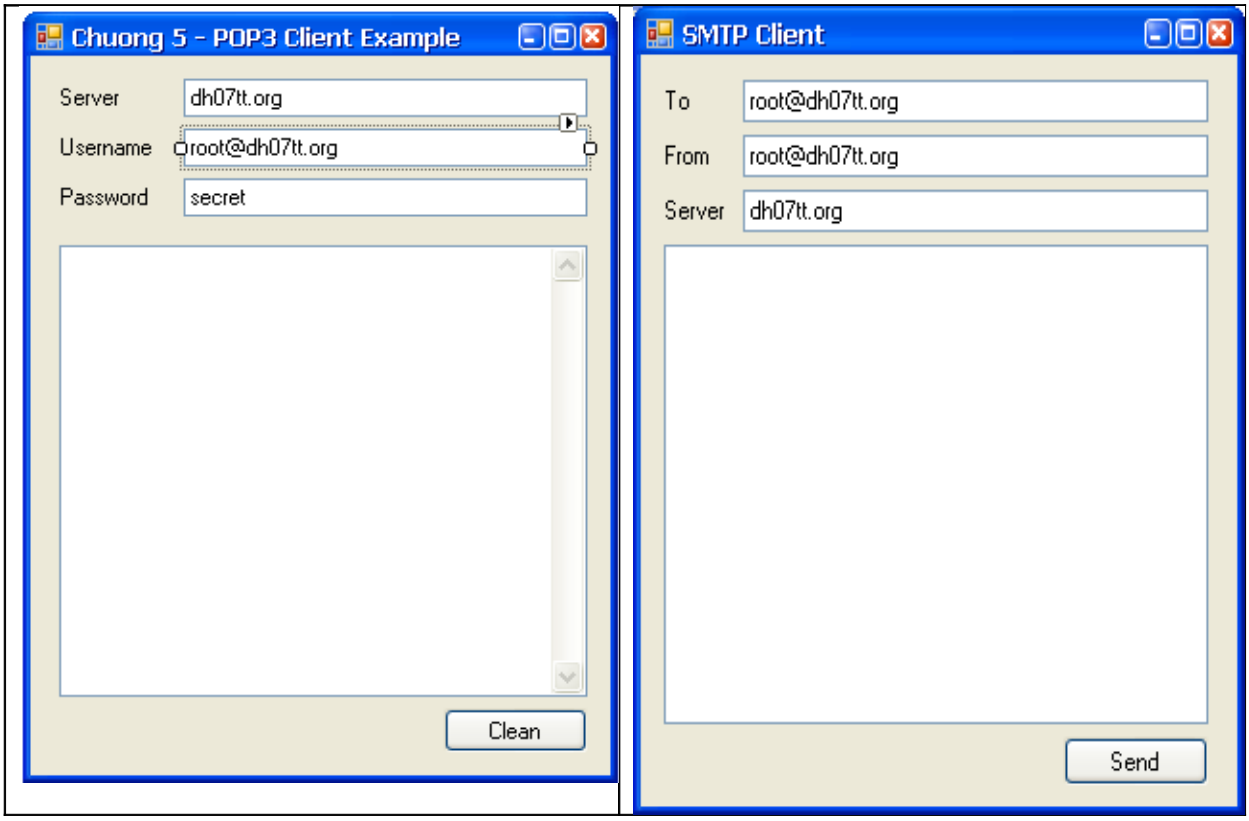
3. Mở rộng: Viết thêm chức năng đính kèm file cho chương trình

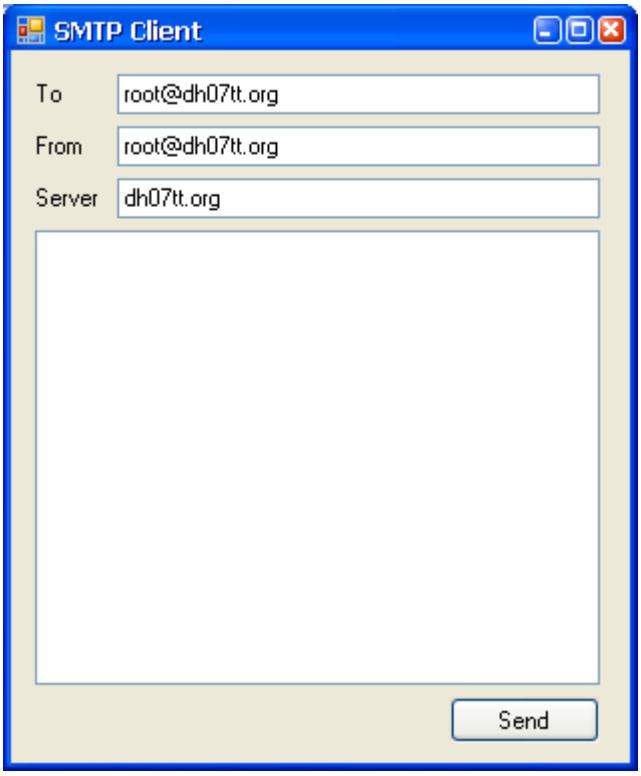
- Thời gian thực hiện bài tập vượt quá 5% thời gian cho phép sẽ không được đánh giá.

- Thí sinh phải tuyệt đối tuân thủ các qui định an toàn lao động, các qui định của xưởng thực tập, nếu vi phạm sẽ bị đình chỉ thi.

BÀI TẬP:

Thiết kế chương trình POP3, SMTP theo dạng sau


Chuong 5 - POP3 Client Example
Server
dh07tt.org
Username
root@dh07tt.org
Password
secret
Clean
SMTP Client
To
root@dh07tt.org
From
root@dh07tt.org
Server
dh07tt.org
Send


SMTP Client
To
root@dh07tt.org
From
root@dh07tt.org
Server
dh07tt.org
Send

BÀI 6 : TRUYỀN THÔNG VỚI FILE SERVER

Mã bài : MĐ35.6

Mục tiêu của bài:

- Trình bày được cách sử dụng các lớp khi sử dụng với FTP;
- Xây dựng ứng dụng trong việc truyền và nhận File.
- Thực hiện các thao tác an toàn với máy tính.

1. Tổng quan về File server và truyền File

Bất kỳ ai có kinh nghiệm trong thiết kế web biết rằng để đưa các trang web "sống", các tập tin trang Web cần phải được gửi đến một máy chủ Web được cung cấp bởi công ty lưu trữ của bạn hoặc ISP. Hầu hết mọi người không bao giờ có thể nhìn thấy máy vật lý rằng trang web của họ được lưu trữ trên, và liên lạc duy nhất của họ với nó là thông qua một giao thức truyền file, hoặc FTP, chương trình như CuteFTP hoặc SmartFTP.

FTP là nền tảng phổ biến nhất tập tin chuyển đổi cơ cấu giữa các máy tính trên Internet. FTP là phần mềm tự do có sẵn cho tất cả các hệ điều hành, bao gồm Windows, UNIX, và hệ điều hành Mac OS X. khả năng tương tác đa nền tảng này là rất quan trọng cho việc phát triển trang web

bởi vì hầu hết các công việc thiết kế Web trên Windows và hầu hết các máy chủ web chạy từ UNIX, Linux , và hệ điều hành Netware.

FTP như được định nghĩa trong RFC 1350 thay thế cho một giao thức cũ được gọi là tầm thường giao thức truyền tập tin (TFTP). Hệ thống này rất hiếm khi được sử dụng trên Internet, nhưng nó có thể được sử dụng cho các thủ tục chẳng hạn như khả năng khởi động không đĩa trên mạng. Nó không có cơ sở xác thực.

FTP là cơ chế chính để truyền File giữa các máy tính thông qua môi trường Internet

1.1. Chia sẻ File của Microsoft.

Một công nghệ mạnh mẽ của Microsoft là Internet Common File (CIF). Đây là giao thức chia sẻ tập tin tự nhiên của Windows 2000 và XP. Nó là một phần mở rộng của giao thức Server Messesge Block (SMB) được sử dụng trong các phiên bản trước của Windows. Nó được sử dụng để cung cấp tính năng ổ đĩa mạng và in chia sẻ. Nó là an toàn, nhan so với FTP, vì NTLM mã hóa, và nói chung là nhanh, tuy nhiên, không phải là Windows thực hiện không phải phổ biến, nhưng vẫn tồn tại cho VMS và UNIX. Giao thức là phần lớn độc quyền, thường là một cản trở cho không phải của Microsoft phát triển.

Windows chia sẻ file phổ biến nhất trong mạng văn phòng, nơi nhiều nhân viên chia sẻ một máy in hoặc một kho lưu trữ trung tâm cho các tập tin. Từ phương diện lập trình, nó là một công nghệ lý tưởng để sử

dụng như là một giải pháp một lần tại một công ty nơi mà tất cả các người sử dụng hệ thống sẽ được vào cùng một mạng nội bộ. Ví dụ, một công ty kiến trúc đã được tìm kiếm một kho lưu trữ trung tâm cho bản vẽ, chia sẻ mạng sẽ là lý tưởng bởi vì nó không yêu cầu phải lập trình. Hệ thống tương đương bằng cách sử dụng FTP sẽ chậm hơn, khó khăn, và kém an toàn, tuy nhiên, nếu công ty muốn chia sẻ bản vẽ với các công ty khác, sau đó FTP sẽ phù hợp hơn vì khả năng tương tác của nó và dễ triển khai trên Internet (chứ không phải là mạng nội bộ) môi trường. Các điều khoản NETBIOS và NetBEUI là tên chính xác hơn chia sẻ file và in cho Microsoft. Một hương vị của NETBIOS, NBT chạy trên IP, nhưng tất cả các hình thức khác không dựa trên địa chỉ IP, họ sử dụng tên máy NETBIOS Những tên máy chủ được giải quyết vào địa chỉ vật lý trong một trong bốn cách. Họ có thể phát sóng theo yêu cầu trên mạng (B-Node). Thay vào đó, họ có thể truy vấn một máy chủ WINS (P-Node). Sử dụng một sự kết hợp của các phương pháp này, bằng cách phát sóng trước khi truy vấn, hoạt động M-Node, và mặt sau là hoạt động H-Node.

1.2. Chia sẻ File của Netware

Xuất hiện cách đây hơn nhiều thập kỷ, đây là cơ chế chuyển tập tin nhanh nhất, . Đó là, tuy nhiên, một trong các giao thức chuyển giao nhanh nhất tập tin qua mạng nội bộ. Nó được xây dựng trên đầu trang của Internetworking Packet exchange / Sequenced Packet Exchange (IPX / SPX) do vậy giao thức không định tuyến được. Cần phải có bộ phiên dịch để chuyển đổi các gói tin sang TCP / IP, nên các yếu tố hiệu suất bị mất. Hệ thống Netware (cũng được gọi là IntranetWare) tập trung trên một máy chủ trung tâm Netware. Máy chủ này chạy hệ điều hành Novell, được bắt đầu từ một ứng dụng tải và khởi động hệ điều hành DOS. Các máy chủ lưu trữ Netware dịch vụ thư mục (NDS), được sử dụng để kiểm soát chứng thực và đặc quyền.

Novell máy chủ (3.x) sử dụng một chô đóng sách thay vì NDS. Sự khác biệt giữa hai hệ thống là NDS là một cơ sở dữ liệu quan hệ và có thể nhân rộng giữa các máy chủ khác, trong khi chô đóng sách không thể Novell khách hàng có sẵn cho hầu hết các nền tảng, từ hệ điều hành DOS và Windows Macintosh và UNIX. Các khách hàng xác định vị trí máy chủ bằng cách sử dụng giao thức Novell lõi (NCP). Khi một máy chủ tập tin từ xa được tìm thấy, nó ánh xạ một ổ đĩa cục bộ trên máy tính của khách hàng Có là không có hỗ trợ cho interoperating với Netware trong. NET, và là không có cam kết nhỏ để tích hợp một. NET với một mạng Novell. Nếu bạn có làm như vậy, nhìn vào giao diện dòng lệnh DOS vào mạng, hoặc không đó, hãy thử interfacing ở cấp IPX bằng cách sử dụng ổ cắm nguyên.

2. Truyền File

2.1.Cách thức dùng các cổng của FTP

Trong các giao thức email, phần dữ liệu có chiều dài biến (tức là, email) có thể được hậu tố <Enter> . Enter để đánh dấu sự kết thúc của dữ liệu. Nếu chuỗi ký tự này được phát hiện trong cơ thể của email, nó có thể được loại bỏ trước khi gửi mà không có bất kỳ suy thoái thực sự của mức độ dễ đọc của thư điện tử, tuy nhiên, trong FTP, một tập tin thực thi khá dễ dàng có thể có chuỗi ký tự được nhúng bên trong nó, và loại bỏ những người ký tự có thể gây ra các tập tin bị hỏng.
Để tránh vấn đề này, cổng 21 được sử dụng để gửi và nhận lệnh và phản ứng, mỗi chấm dứt bởi một . Khi dữ liệu chiều dài thay đổi được gửi giữa máy khách và máy chủ, chẳng hạn như các tập tin hoặc danh sách thư mục, một kết nối tạm thời được mở trên cổng 20, dữ liệu được chuyển giao, và cổng được đóng lại một lần nữa. Tuy nhiên, trong hầu hết các thực tế triển khai khách hàng FTP, FTP client có thể là một bức tường lửa, do đó, các máy chủ nên làm tất cả những việc phục vụ và khách hàng nên làm tất cả những yêu cầu
thụ động chế độ FTP là nơi mà khách hàng chỉ thị các máy chủ. lắng nghe trên một cổng khác hơn so với các cổng dữ liệu mặc định. Các khách hàng sau đó sẽ kết nối với cổng này và sử dụng nó để tải lên và tải về như bình thường.
Để đáp ứng với lệnh PASV sẽ luôn bao gồm một danh sách trong ngoặc vuông số sáu số cách nhau bằng dấu phẩy. Bốn nhóm chữ số đầu tiên epresent địa chỉ IP của máy chủ, và hai nhóm cuối cùng đại diện cho các cổng máy chủ lắng nghe cho các kết nối dữ liệu của nó. Trong ví dụ trước, bốn chữ số là 212,17,38,3,11,144 . Điều này có nghĩa là do máy chủ được đặt tại địa chỉ IP 212.17.38. 3 và lắng nghe trên cổng 2960 (11 × 256 + 144). Các máy chủ sẽ bắt đầu lắng nghe trên cổng ngay sau khi nhận được lệnh PASV. Nó sẽ trả về một tin nhắn 227 để cho biết rằng nó đã bắt đầu lắng nghe trên cổng này. Một khi khách hàng kết nối đến cổng này, máy chủ sẽ trả về một tin nhắn 150. Nếu khách hàng không kết nối với các cảng trong một cách kịp thời (một vài giây), máy chủ sẽ ra một thông điệp thời gian chờ 425. Máy chủ sẽ gửi các dữ liệu được yêu cầu trên cổng đó và đóng kết nối một khi tất cả các dữ liệu được gửi và sau đó ra thông báo 226. Quá trình tương tự xảy ra theo chiều ngược lại khi tải lên máy chủ. Trong trường hợp này , lệnh PASV được phát hành, và khách hàng kết nối vào cổng theo quy định của máy chủ. Khách hàng sau đó đặt nội dung của tập tin trên ổ cắm mới và đóng kết nối một khi tập tin được gửi.

2.2. Bắt tay truyền File

Tiến trình bắt tay làm việc của giao thức truyền file FTP

- Cơ chế chứng thực : FTP chấp nhận username/password dạng text thô, nên có thể nhìn thấy được với bất kỳ người nào dùng trình phân tích.
- FTP trên SSL được khuyến cáo khi website truyền những thông tin quan trọng.
- FTP Server cũng cho phép truy cập nặc danh (anonymous). Khi ấy username là anonymous và password là tùy ý. Đây là thiết lập mặc định của dịch vụ Microsoft FTP.
- Khi kết nối server ở port 21, server sẽ phản hồi lại như sau :
  220 <some message><enter>
- Sau đó diễn ra giống quá trình bắt tay của POP3, các lệnh USER và PASS được gởi lần lượt là :
  USER <username><enter>
- Server phản hồi
  331 <some message><enter>
- Client gởi :
  PASS <password><enter>
- Server phản hồi nếu đăng nhập thành công:
  230 <some message><enter>
- Server phản hồi nếu đăng nhập thất bại:
  530 <some message><enter>
- Sau thời điểm này, nếu đăng nhập thành công thì người dùng có thể truy cập vào FTP Server với quyền hạn tương ứng.
- Một số FTP server sẽ hủy kết nối với những người dùng không thao tác gì để tiết kiệm tài nguyên. Do đó, có thể dùng lệnh NOOP để báo cho server biết nhằm tránh hiện tượng trên. Diễn biến như sau :
  NOOP <enter>
  200 <some message><enter>
- Để đóng kết nối, client đơn giản chỉ cần đóng kết nối TCP bên dưới bằng cách phát lệnh QUIT, diễn biến như sau :
  QUIT <enter>

200 <some message><enter>

2.3. Truyền thông qua thư mục

Ở chế độ chủ động (active), máy khách FTP (FTP client) dùng 1 cổng ngẫu nhiên không dành riêng (cổng N > 1024) kết nối vào cổng 21 của FTP Server. Sau đó, máy khách lắng nghe trên cổng N+1 và gửi lệnh PORT N+1 đến FTP Server. Tiếp theo, từ cổng dữ liệu của mình, FTP Server sẽ kết nối ngược lại vào cổng dữ liệu của Client đã khai báo trước đó (tức là N+1) Ở khía cạnh firewall, để FTP Server hỗ trợ chế độ Active các kênh truyền sau phải mở:

- Cổng 21 phải được mở cho bất cứ nguồn gửi nào (để Client khởi tạo kết nối)

- FTP Server's port 21 to ports > 1024 (Server trả lời về cổng điều khiển của Client)

- Cho kết nối từ cổng 20 của FTP Server đến các cổng > 1024 (Server khởi tạo kết nối vào cổng dữ liệu của Client)

- Nhận kết nối hướng đến cổng 20 của FTP Server từ các cổng > 1024 (Client gửi xác nhận ACKs đến cổng data của Server)

- Bước 1: Client khởi tạo kết nối vào cổng 21 của Server và gửi lệnh PORT 1027.

- Bước 2: Server gửi xác nhận ACK về cổng lệnh của Client.

- Bước 3: Server khởi tạo kết nối từ cổng 20 của mình đến cổng dữ liệu mà Client đã khai báo trước đó.

- Bước 4: Client gửi ACK phản hồi cho Server.

Khi FTP Server hoạt động ở chế độ chủ động, Client không tạo kết nối thật sự vào cổng dữ liệu của FTP server, mà chỉ đơn giản là thông báo cho Server biết rằng nó đang lắng nghe trên cổng nào và Server phải kết nối ngược về Client vào cổng đó. Trên quan điểm firewall đối với máy Client điều này giống như 1 hệ thống bên ngoài khởi tạo kết nối vào hệ thống bên trong và điều này thường bị ngăn chặn trên hầu hết các hệ thống Firewall.

Ví dụ phiên làm việc active FTP:

Trong ví dụ này phiên làm việc FTP khởi tạo từ máy testbox1.slacksite.com (192.168.150.80), dùng chương trình FTP Client dạng dòng lệnh, đến máy chủ FTP testbox2.slacksite.com (192.168.150.90). Các dòng có dấu --> chỉ ra các lệnh FTP gửi đến Server và thông tin phản hồi từ các lệnh này. Các thông tin người dùng nhập vào dưới dạng chữ đậm. Lưu ý là khi lệnh PORT được phát ra trên Client được thể hiện ở 6 byte. 4 byte đầu là địa chỉ IP của máy Client còn 2 byte sau là số cổng. Giá trị cổng được tính bằng (byte_5*256) + byte_6, ví dụ ((14*256) + 178) là 3762.

Passive FTP.

Để giải quyết vấn đề là Server phải tạo kết nối đến Client, một phương thức kết nối FTP khác đã được phát triển. Phương thức này gọi là FTP thụ động (passive) hoặc PASV (là lệnh mà Client gửi cho Server để báo cho biết là nó đang ở chế độ passive).

Ở chế độ thụ động, FTP Client tạo kết nối đến Server, tránh vấn đề Firewall lọc kết nối đến cổng của máy bên trong từ Server. Khi kết nối FTP được mở, client sẽ mở 2 cổng không dành riêng N, N+1 (N > 1024). Cổng thứ nhất dùng để liên lạc với cổng 21 của Server, nhưng thay vì gửi lệnh PORT và sau đó là server kết nối ngược về Client, thì lệnh PASV được phát ra. Kết quả là Server sẽ mở 1 cổng khôngdành riêng bất kỳ P (P > 1024) và gửi lệnh PORT P ngược về cho Client.. Sau đó client sẽ khởi tạo kết nối từ cổng N+1

vào cổng P trên Server để truyền dữ liệu. Từ quan điểm Firewall trên Server FTP, để hỗ trợ FTP chế độ passive, các kênh truyền sau phải được mở:

- Cổng FTP 21 của Server nhận kết nối từ bất nguồn nào (cho Client khởi tạo kết nối)
- Cho phép trả lời từ cổng 21 FTP Server đến cổng bất kỳ trên 1024 (Server trả lời cho cổng control của Client)
- Nhận kết nối trên cổng FTP server > 1024 từ bất cứ nguồn nào (Client tạo kết nối để truyền dữ liệu)
- Cho phép trả lời từ cổng FTP Server > 1024 đến các cổng > 1024 (Server gửi xác nhận ACKs đến cổng dữ liệu của Client)
- Bước 1: Client kết nối vào cổng lệnh của Server và phát lệnh PASV.
- Bước 2: Server trả lời bằng lệnh PORT 2024, cho Client biết cổng 2024 đang mở để nhận kết nối dữ liệu.
- Bước 3: Client tạo kết nối truyền dữ liệu từ cổng dữ liệu của nó đến cổng dữ liệu 2024 của Server.
- Bước 4: Server trả lời bằng xác nhận ACK về cho cổng dữ liệu của Client.

Trong khi FTP ở chế độ thụ động giải quyết được vấn đề phía Client thì nó lại gây ra nhiều vấn đề khác ở phía Server. Thứ nhất là cho phép máy ở xa kết nối vào cổng bất kỳ > 1024 của Server. Điều này khá nguy hiểm trừ khi FTP cho phép mô tả dãy các cổng >= 1024 mà FTP Server sẽ dùng (ví dụ WU-FTP Daemon).

Vấn đề thứ hai là một số FTP Client lại không hổ trợ chế độ thụ động. Ví dụ tiện ích FTP Client mà Solaris cung cấp không hổ trợ FTP thụ động. Khi đó cần phải có thêm trình FTP Client. Một lưu ý là hầu hết các trình duyệt Web chỉ hổ trợ FTP thụ động khi truy cập FTP Server theo đường dẫn URL .

Ví dụ phiên làm việc passive FTP:

Trong ví dụ này phiên làm việc FTP khởi tạo từ máy testbox1.slacksite.com (192.168.150.80), dùng chương trình FTP Client dạng dòng lệnh, đến máy chủ FTP testbox2.slacksite.com (192.168.150.90), máy chủ Linux chạy ProFTPd 1.2.2RC2. Các dòng có dấu --> chỉ ra các lệnh FTP gửi đến Server và thông tin phản hồi từ các lệnh này. Các thông tin người nhập vào dưới dạng chữ đậm.

Lưu ý: đối với FTP thụ động, cổng mà lệnh PORT mô tả chính là cổng sẽ được mở trên Server. Còn đối với FTP chủ động cổng này sẽ được mở ở Client.

2.4.Tham khảo các lệnh của FTP

Sử dụng FTP để kết nối đến một remote machine qua Internet (trực tuyến hoặc thông qua các nhà cung cấp dịch vụ) hoặc qua mạng LAN, WAN

rất đơn giản. Để sử dụng FTP, khởi động phần mềm FTP của client và cung cấp tên của remote machine mà ta muốn kết nối đến. Ví dụ để kết nối đến một remote machine thông qua LAN hay Internet ta đánh vào dòng lệnh sau :

```
ftp chatton.com
```

Lệnh này chỉ dẫn cho phần mềm FTP cố gắng kết nối với máy có tên chatton.com và thiết lập một phiên giao dịch cho FTP.Khi kết nối hoàn tất hệ thống sẽ đòi hỏi userID. Nếu hệ thống yêu cầu FTP mặc định thì một thông báo sẽ được gửi đến cho user để biết chính xác điều đó. Truy xuất sau dùng trong Linux FTP với nơi lưu trữ là sunsite.unc.edu

Code:

```
ftp sunsite.unc.edu
331 Guest login ok, send your complete e-mail address as password.
Enter username (default: anonymous): anonymous
Enter password [tparker@tpci.com]:
|FTP| Open
230- WELCOME to UNC and SUN's anonymous ftp server
230- University of North Carolina
230- Office FOR Information Technology
230- SunSITE.unc.edu
230 Guest login ok, access restrictions apply.
FTP>
```

Sau khi kết nối xong ta sẽ thấy dấu nhắc FTP> cho biết remote system đang sẵn sàng nhận lệnh.

Khi log on vào một vài hệ thống, một thông báo ngắn có thể xuất hiện chứa các lệnh về download file, các giới hạn đối với các user sử dụng FTP mặc định hoặc thông tin về vị trí các file hữu dụng. Ví dụ :

Code:

```
To get a binary file,type: BINARY and then: GET "File.Name" newfilename

To get a text file, type: ASCII and then: GET "File.Name" newfilename

Names MUST match upper, lower case exactly. Use the "quotes" as shown.

To get a directory, type: DIR. To change directory, type: CD "Dir.Name"

To read a short text file, type: GET "File.Name" TT
```

```
For more, type HELP or see FAQ in gopher.

To quit, type EXIT or Control-Z.

230- If you email to info@sunsite.unc.edu you will be sent help information

230- about how to use the different services sunsite provides.

230- We use the Wuarchive experimental ftpd. if you "get" <directory>.tar.Z

230- or <file>.Z it will compress and/or tar it on the fly. Using ".gz" instead

230- of ".Z" will use the GNU zip (/pub/gnu/gzip*) instead, a superior
230- compression method.
```

Khi vào được remote machine ta có thể sử dụng các lệnh của Linux để hiển thị các files và di chuyển giữa các thư mục. Ví dụ muốn thể hiện các file có trong thư mục ta dùng lệnh ls, chuyển thư mục dùng lệnh cd, trở về thư mục cha dùng lệnh cd… Các lệnh nay cũng giống như các lệnh ta sử dụng trên máy đơn, ngoại trừ một điều hiện tại ta đang sử dụng trên remote system, để chuyển đổi thư mục trên local machine ta có thể dùng lệnh lcd.

FTP không có các phím tắt, khi cần thực hiện lệnh ta phải đánh vào đầy đủ tên file hay thư mục cần truy xuất. Khi đánh sai tên file hay thư mục thì thông báo lỗi sẽ xuất hiện và ta phải đánh lại.

• Truyền file :

Việc truyền nhận file là điểm chính của FTP, do đó ta cần biết làm thế nào để gọi một file từ remote system cũng như làm cách nào để ghi một file lên đó.

Khi muốn chuyển một file từ remote machine về máy mình ta dùng lệnh get và để tên file vào sau lệnh này. Ví dụ :

Code:

```
get “soundcard_driver”
```

Lệnh này sẽ chuyển file soundcard_driver từ remote machine về local machine. Khi sử dụng lệnh get remote system sẽ truyền dữ liệu về local machine và đưa ra một thông báo khi quá trình hoàn tất. Khi truyền các file có dung lượng lớn thì hệ thống sẽ không có thể hiện gì về quá trình truyền dữ liệu cho đến khi hoàn tất do đó hãy kiên nhẫn chờ đợi. Một số phiên bản FTP yêu cầu thể hiện các thông báo sau mỗi lần truyền xong 1024 bytes, việc này giúp chúng ta theo dõi tốt hơn quá trình truyền dữ liệu. Ví dụ :

Code:

```
FTP> get "file1.txt"

200 PORT command successful.

150 BINARY data connection for FILE1.TXT (27534 bytes)

226 BINARY Transfer complete.

27534 bytes received in 2.35 seconds (12 Kbytes/s).
```

Khi muốn ghi một file từ local machine lên remote machine ta dùng lệnh put. Câu lệnh sau dùng để ghi file comments từ local machine lên remote machine :
Code:

```
put "comments"
```

Dấu “” là không cần thiết trong các phiên bản của FTP, nó dùng để ngăn chặn cấu trúc mở rộng của ký tự. Trong hầu hết các file thì dấu “” là không cần thiết tuy nhiên sử dụng nó cũng là một thói quen tốt.

Một số phiên bản FTP cung cấp các khả năng mở rộng sử dụng các lệnh mget và mput. Các lệnh get và put chỉ dùng cho các file có tên đầy đủ, trong khi các lệnh mget và mput cho phép sử dụng phần mở rộng. Ví dụ để truyền tất cả các file có phần mở rộng .doc ta dùng lệnh sau :

Code:

```
mget *.doc
```

• Các định dạng file khác nhau :

FTP của Linux cung cấp hai dạng truyền file : ASCII và binary. Một số hệ thống tự động chuyển đổi giữa hai kiểu này khi nó nhận được file ở dạng binary. Chúng ta không nên tự chuyển đổi kiểu ngoại trừ đã thử trước và biết chắc là nó hoạt động tốt. Hầu hết các phiên bản FTP mặc nhiên khởi động với dạng ASCII.
Để thiết lập dạng truyền binary cho FTP ta dung lệnh binary, và ta cũng có thể trở về dạng ASCII với lệnh ascii. Tuy nhiên tốt nhất là nên truyền ở dạng binary, nếu ta truyền một file binary ở dạng ASCII thì nó không thể thực thi được trên hệ thống nhận. Việc truyền một file ASCII ở dạng binary se không làm ảnh hưởng gì đến nội dung ngoại trừ một số rất ít các thể hiện. Khi truyền file giữa hai hệ thống Linux ở dạng binary thì sẽ giữ được các thuộc tính của file, nhưng nếu truyền giữa một hệ thống Linux và một hệ thống không phải Linux thì có thể có vấn đề với một số kiểu file. Dạng ASCII chỉ thích hợp cho việc truyền trực tiếp các text file.

• Thoát khỏi FTP :

Để thoát khỏi FTP ta dùng lệnh quit hoặc exit, cả hai lệnh này sẽ đóng kết nối của chúng ta với remote machine sau đó chấm dứt FTP trên local machine của chúng ta. Các lệnh có thể sử dụng cho users trong các phiên bản FTP là :

ascii Chuyển sang dạng truyền mã ASCII
binary Chuyển sang dạng truyền mã binary
cd Đổi thư mục trên server
close Chấm dứt kết nối
del Xóa một file trên server
dir Xem nội dung thư mục trên server
get Lấy một file từ server
hash Chỉ ra một giá trị ký tự cho mỗi khối được truyền
help Giúp đỡ
lcd Thay đổi thư mục trên client
mget Tải nhiều file từ server
mput Gửi nhiều file đến server
open Kết nối đến một server
put Gửi một file đến server
pwd Xem thư mục hiện hành trên server
quote Cung cấp các lệnh FTP một cách trực tiếp
quit Kết thúc FTP

Trong hầu hết các phiên bản, các lệnh của FTP rất nhạy cảm, nếu ta đánh một lệnh ở dạng chữ hoa thì FTP sẽ báo lỗi, một số phiên bản sẽ chuyển đổi cho chúng ta nhưng nó không chỉ cho ta phải dùng dạng nào. Bởi vì Linux sử dụng chữ thường cho mọi thứ nên chúng ta cũng nên sử dũng kiểu chữ thường cho các phiên bản của FTP.

• FTP sử dụng TCP như thế nào ? :

FTP sử dụng 2 kênh truyền TCP : TCP cổng 20 dùng cho truyền dữ liệu, TCP cổng 21 dùng cho truyền các lệnh. Cả hai kênh này phải được cho phép tạo các hàm cho FTP trên hệ thống Linux cần sử dụng. Việc sử dụng cả hai kênh truyền này làm cho FTP khác với hầu hết các chương trình truyền file khác. Bằng cách sử dụng cả hai kênh truyền TCP cho phép truyền cùng lúc các lệnh và dữ liệu của FTP. FTP hoạt động mạnh và không sử dụng bộ đệm và hàng đợi.

FTP sử dụng một server deamon chạy một cách liên tục và phân chia các chương trình được thực thi trên client. Trong hệ thống Linux thì server deamon được gọi là ftpd và chương trình trên client được gọi là ftp. Trong suốt quá trình thiết lập kết nối giữa client và server và khi người sử dụng đánh vào một lệnh cho FTP thì cả hai máy sẽ truyền cho nhau một chuỗi các lệnh. Các lệnh này dành riêng cho FTP và được xem như các giao thức bên trong. Các lệnh của các giao thức bên trong của FTP bao gồm một

chuỗi 4 ký tự mã ASCII và được kết thúc bằng ký tự xuống dòng, một số lệnh đòi hỏi các tham số. Thuận lợi của việc dùng các ký tự ASCII cho các câu lệnh là người sử dụng dễ dàng quan sát và hiểu các lệnh, việc này rất hữu ích trong quá trình gỡ rối. Một người thông thạo có thể sử dụng các lệnh bằng mã ASCII một cách trực tiếp để liên lạc với các server của FTP mà không cần thông qua client (nói một cách khác, liên lạc với ftpd không cần ftp trên một local machine). Tuy nhiên việc này hiếm khi được sử dụng, ngoại trừ trong quá trình gỡ rối.
Sau khi dùng FTP để login vào remote machine, chúng ta không thật sự ở trên remote machine mà về mặt luận lý ta vẫn đang ở trên client, do đó tất cả các lệnh về truyền file và di chuyển thư mục phải phụ thuộc vào local machine chứ không phải remote machine.

Các quá trình theo sau FTP khi một kết nối được thiết lập như sau :

+ Login : kiểm tra user ID và password.
+ Define directory : nhận dạng thư mục bắt đầu .
+ Define file transfer mode : định nghĩa kiểu truyền file.
+ Start data trasfer : nhận các lệnh của người sử dụng .
+ Stop data transfer : đóng kết nối.

Việc chọn lựa gỡ rối là có sẵn trong câu lệnh của FTP bằng cách thêm vào tham số –d sau câu lệnh. Tùy chọn này cho biết kênh của các lệnh, các lệnh từ client được thể hiện bằng một mũi tên như là ký tự đầu tiên, các lệnh từ server có 3 chữ số đứng trước. Một PORT trong câu lệnh cho biết địa chỉ của kênh dữ liệu mà client đang chờ server trả lời, nếu không có PORT nào được nêu rõ thì mặc nhiên kênh 20 sẽ được sử dụng. Quá trình xử lý truyền dữ liệu không thể thực hiện trong chế độ gỡ rối. Sau đây là một ví dụ về tùy chọn gỡ rối :

Code:

```
$ ftp -d tpci_hpws4
Connected to tpci_hpws4.
220 tpci_hpws4 FTP server (Version 1.7.109.2 Tue Jul 28 23:32:34 GMT 1992) ready.
Name (tpci_hpws4:tparker):
---> USER tparker
331 Password required for tparker.
Password:
---> PASS qwerty5
230 User tparker logged in.
---> SYST
215 UNIX Type: L8
Remote system type is UNIX.
---> Type I
```

```
200 Type set to I.
Using binary mode to transfer files.
ftp> ls
---> PORT 47,80,10,28,4,175
200 PORT command successful.
---> TYPE A
200 Type set to A.
---> LIST
150 Opening ASCII mode data connection for /bin/ls.
total 4
-rw-r----- 1 tparker tpci 2803 Apr 29 10:46 file1
-rw-rw-r-- 1 tparker tpci 1286 Apr 14 10:46 file5_draft
-rwxr----- 2 tparker tpci 15635 Mar 14 23:23 test_comp_1
-rw-r----- 1 tparker tpci 52 Apr 22 12:19 xyzzy
Transfer complete.
---> TYPE I
200 Type set to I.
ftp> <Ctrl-d>
$
```

• Cấu hình FTP :

Khi cung cấp một ẩn danh FTP hoặc một hệ thống user-login chúng ta cần thực hiện một số cấu hình cơ bản của FTP để thiết lập thuộc tính cho các thư mục và file hệ thống để ngăn chặn việc truy xuất hoặc sửa đổi của các user. Quá trình có thể bắt đầu bằng việc chọn một định danh FTP, thật ra việc này không thật sự cần thiết mặc dù nó sẽ dễ dàng hơn cho các user khác muốn truy xuất đến. Định danh FTP có dạng sau :

Trong đó domain_name là tên chính (hoặc ẩn danh) của FTP server và domain_type là phần mở rộng của DNS. Ví dụ chúng ta có thể có một định danh như sau :

Việc này chỉ ra rằng đây là một FTP ẩn danh cho phép mọi user truy xuất đến địa chỉ tpci.com, không nên đưa ra một tên máy cụ thể như sau :

Điều này gây khó khăn cho việc thay đổi FTP server, thay vào đó sử dung một bí danh để chỉ đến máy server sẽ dễ dàng hơn khi cần thay đổi server. Tuy nhiên nếu một máy đơn kết nối Internet thông qua một nhà cung cấp dịch vu thì việc này không ảnh hưởng gì, nó thường cần thiết đối với các mạng lớn hơn. Nếu sử dụng DNS thì bí danh rất dễ thiết lập. Ví dụ :

Code:

```
ftp.tpci.com. IN CNAME merlin.tpci.com.
```

Lệnh này chỉ ra rằng bất kỳ một user nào truy xuất đến địa chỉ sẽ truy xuất đến máy có tên merlin.tpci.com, nếu vì lý do nào đó máy merlin không

đóng vai trò server nữa thì việc thay đổi tên máy trong dòng lệnh này sẽ chỉ đến một server mới. Khoảng trống sau domain name là rất quan trọng vì nó ngăn chặn việc mở rộng tên bao gồm cả việc lặp lại domain name.

• Cài đặt ftpd :

Ftpd (FTP daemon) phải được khởi động trên FTP server (một số phiên bản Linux sử dụng daemon wu.ftpd làm server). Daemon được quản lý bởi inetd thay cho các file khởi động rc, do đó ftpd chỉ hoạt động khi được yêu cầu, điều đó làm cho FTP trở nên rất nặng. Khi inetd khởi động ftpd thì nó sẽ kiểm tra cổng lệnh (kênh 21) cho các gói dữ liệu đến và tạo một ftpd. Phải chắc chắn rằng inetd có thể khởi động ftpd bằng cách kiểm tra các file cấu hình của inetd (thường nằm trong /etc/inetd.config) bằng dòng lệnh sau :

Code:

```
ftp stream tcp nowait root /usr/etc/ftpd ftpd –l
```

Nếu lệnh này không tồn tại thì thêm nó vào file, với hầu hết các hệ thống Linux thì dòng lệnh trên luôn có sẵn trong file mặc dù nó có thể được chỉ dẫn từ bên ngoài. FTP entry chỉ rõ inetd mà FTP đang hoạt động để sử dụng TCP và nó sẽ sinh ra ftpd mỗi khi một kết nối mới được tạo ra cho cổng FTP. Trong ví dụ trên ftpd được khởi động với tùy chọn –l sẽ cho phép logging, ta có thể bỏ qua tùy chọn này. Các tùy chọn thường được sử dụng :

-d This option adds debugging information to the syslog.

-l This option activates logging of sessions (only failed and successful logins, not debug information). If the -l option is specified twice, all commands are logged, too. If specified three times, the size of all get and put file transfers are added, as well.

-t This option sets the timeout period before ftpd terminates after a session is concluded (default is 15 minutes). The value is specified in seconds after the -t option.

-T This option sets the maximum timeout period (in seconds) that a client can request. The default is two hours. This enables a client to alter the normal default timeout for some reason.

-u This option sets the umask value for files uploaded to the local system. The default umask is 022. Clients can request a different umask value.

• FTP logins :

Mỗi user muốn truy xuất hệ thống phải có một login name và password hợp lý, do đó ta phải tạo một account trong file /etc/passwd cho mỗi user. Nếu không cho phép truy xuất với FTP ẩn danh thì không tạo một login chung cho tất cả mọi user.

Để cài đặt một server FTP ẩn danh, ta phải tạo một login cho user ID ẩn danh. Việc này cũng giống như việc thêm một user vào file etc/passwd. Login name sẽ được sử dụng khi truy xuất vào hệ thống chẳng hạn như “anonymous” hoặc “ftp”. Ta phải chọn một thư mục cho các user ẩn danh truy

xuất vào, các thư mục này có thể được bảo vệ khỏi các file hệ thống. Ví dụ :

Code:

```
ftp:*:400:51:Anonymous FTP access:/usr/ftp:/bin/false
```

Dấu * trong password ngăn chặn người khác truy cập vào account, số của user ID là duy nhất trong toàn bộ hệ thống. Để bảo mật tốt hơn, ta nên tạo các nhóm riêng chỉ dành cho các user ẩn danh và đặt các ftp user vào nhóm này.

• Thiết đặt các thư mục :

Ta cần tạo một mini-filesystem chỉ dành cho các truy xuất FTP ẩn danh, các file này chỉ giữ các tên thư mục thông thường và các file cơ bản cần cho việc truy nhập.

Việc thiết đặt các thư mục cho các FTP ẩn danh rất đơn giản, ta chỉ cần tạo một số thư mục và copy các file vào đó. Theo các bước sau :

– Tạo thư mục bin (ví dụ như /usr/ftp/bin) và copy các lệnh (ls, l v.v…) cần thiết để user có thể xem nội dung thư mục và file, có thể copy thêm các ứng dụng nếu cần.

– Tạo thư mục etc (ví dụ /usr/ftp/etc) và copy file passwd (/etc/passwd) và file group (etc/group) vào đó.

– Tạo thư mục lib (ví dụ /usr/ftp/lib) copy hai file /lib/rld và /lib/libc.so.1 vào đó, các file này được sử dụng bởi lệnh ls. Chỉ thực hiện bước này khi lệnh ls yêu cầu các file trên, không phải phiên bản Linux nào cũng có đòi hỏi này, sau đó kiểm tra lại các lệnh đó.

– Tạo thư mục pub (ví dụ /usr/ftp/pub) để giữ các file cho phép truy xuất .

– Tạo thư mục dev (ví dụ /usr/ftp/dev) và dùng lệnh mknod để copy file /dev/zero vào đó cần để số các thiết bị lớn và nhỏ giống như trong file/dev/zero, các driver này sẽ được sử dụng bởi lệnh rld. Thực hiện bước này khi ls yêu cầu các file trong thư mục /lib như trên.

Bản sao của các file /etc/passwd và /etc/group được copy vào thư mục ~ftp/etc để tránh việc truy xuất đến các file thật sự trong thư mục /etc. Việc soạn thảo lại các file này và thay thế password bởi các dấu "*" để tránh việc truy xuất đến các account đó bởi các FTP ẩn danh. Xóa tất cả các ngõ vào đối với /etc/passwd và /etc/group được sử dụng bởi tên và group (nói một cách khác, nó chỉ được sử dụng bởi user và group hợp lý trên hệ thống) cũng như các đường vào khác ngoại trừ các ngõ vào dành cho các FTP ẩn danh. Ta cũng có thể sử dụng cấu trúc thư mục ~ftp/pub để lưu trữ các file cho phép các user ẩn danh truy xuất, copy các file vào đây. Ta cũng có thể tạo các thư mục con cần thiết cho việc tổ chức. Cũng nên tạo một thư mục upload cho phép ghi trong cấu trúc thư mục ~ftp/pub mà các user chỉ có thể upload các file vào vùng này.

• Setting Permissions :

Để bảo vệ hệ thống ta có thể dùng lệnh chroot, lệnh này làm cho thư mục gốc không xuất hiện trên dấu nhắc. Ví dụ khi chroot được thiết đặt cho FTP ẩn danh thì khi các user ẩn danh dùng lệnh cd, giả sử cd /bin họ sẽ thực sự được chuyển đến thư mục /usr/ftp/bin nếu thư mục gốc được thiết đặt là /usr/ftp. Việc này giúp tránh việc truy xuất vào vùng file hệ thống trong cấu trúc thư mục FTP. Tuy nhiên việc thay đổi đó chỉ có hiệu lực đối với các user ID mà lệnh chroot đang hoạt động.

Nếu tạo một upload area thì chỉ nên cho phép write và execute, không cho phép read để tránh việc một user download các file mà user khác upload lên.

Để thiết đặt sự cho phép cho các file và thư mục được dùng bởi những user ẩn danh ta thực hiện các thủ tục sau, nếu các file và thư mục không tồn tại thì copy hoặc tạo ra nếu cần :

– Đặt thư mục ~ftp với khả năng truy xuất cho phép là 555

– Đặt thư mục ~ftp/bin với khả năng truy xuất cho phép là 555

– Đặt file ~ftp/bin/ls với khả năng truy xuất cho phép là 111

– Đặt thư mục ~ftp/etc với khả năng truy xuất cho phép là 555

– Đặt các file ~ftp/etc/passwd và ~ftp/etc/group với khả năng truy xuất cho phép là 444

– Nếu cần sử dụng, đặt thư mục ~ftp/lib với khả năng truy xuất cho phép là 555

– Nếu cần sử dụng, đặt các file ~ftp/lib/rldvà ~ftp/lib/libc.so.1 với khả năng truy xuất cho phép là 444

– Nếu cần sử dụng, đặt thư mục ~ftp/dev với khả năng truy xuất cho phép là 555

– Nếu cần, dùng lệnh mknod tạo ~ftp/dev/zero với số các nút lớn nhỏ giống như trong /dev/zero

Ta có thể thiết đặt quyền sở hữu các file và thư mục bằng lệnh chown như sau :

Code:

```
chown root ~ftp/dev
```

Lệnh này thiết lập quyền sở hữu thư mục ~ftp/dev cho root. Tất cả các thư mục trong cấu trúc thư mục ~ftp nên được thiết lập quyền truy xuất với lệnh chmod. Ví dụ :

chmod 555 dir_name

Lệnh này chỉ cho phép read và excute cho thư mục dir_name. Ngoại trừ thư mục dùng để upload cho phép ghi như đã đề cập ở phần trên.

• Kiểm tra hệ thống :

Trước khi cho phép một ai đó truy xuất vào hệ thống Linux FTP của mình, ta nên kiểm tra bằng cách tự truy xuất vào và thư truy xuất các file bị cấm, vào các thư mục bên ngoài cấu trúc ~ftp, ghi một file lên vùng không cho phép. Thử copy, read, write, move các file và thử log in bằng một tên user nào

đó. Ta phải chắc rằng hệ thống hoàn toàn ổn định, nếu không một user nào đó có thể tìm ra "lỗ hổng" và khai thác nó.

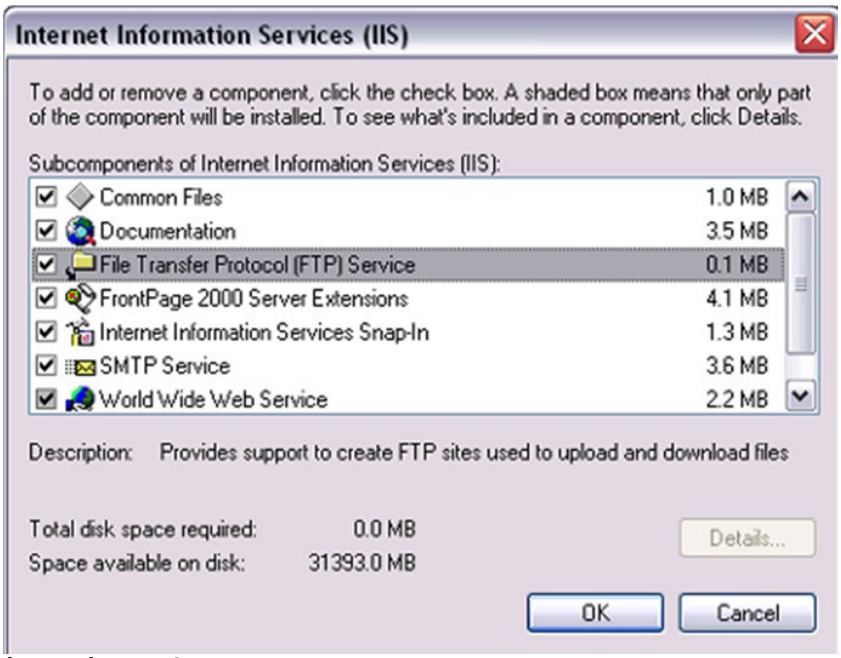
Tốt nhất nên tạo một mailbox để các user trên các hệ thống khác cần giúp đỡ hoặc cần thông tin gì có thể gửi mail cho chúng ta.

2.5. Công cụ FTP

Để truy cập vào một máy chủ FTP, cần phải biết địa chỉ IP của nó và có một tên người dùng và mật khẩu với nó. Hầu hết các ISP cung cấp cho bạn với một lượng nhỏ của không gian web trên máy chủ của họ khi bạn đăng ký, và bạn sẽ có thể để có được những chi tiết này nếu bạn gọi ISP của bạn. Để truy cập vào một máy chủ FTP, bạn cần phải biết địa chỉ IP của nó và có một tên người dùng và mật khẩu với nó. Hầu hết các ISP cung cấp cho bạn với một lượng nhỏ của không gian web trên máy chủ của họ khi bạn đăng ký, và bạn sẽ có thể để có được những chi tiết này nếu bạn gọi ISP của bạn.

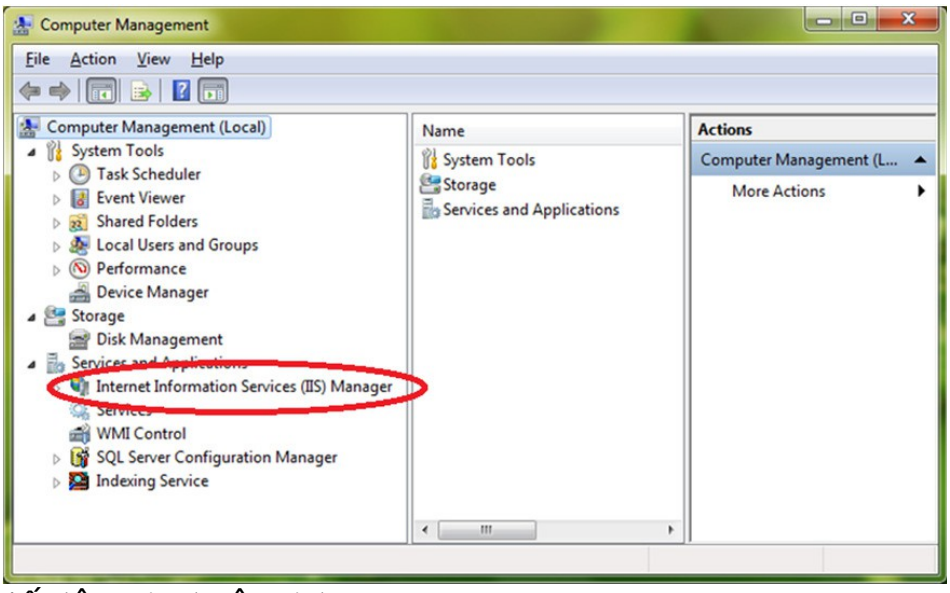
Cách tạo FTP Server trên các Version của Windows:

Click Control Panel→Add/Remove Programs→Add or Remove Windows Components→Internet Information Services→Details→FTP Service

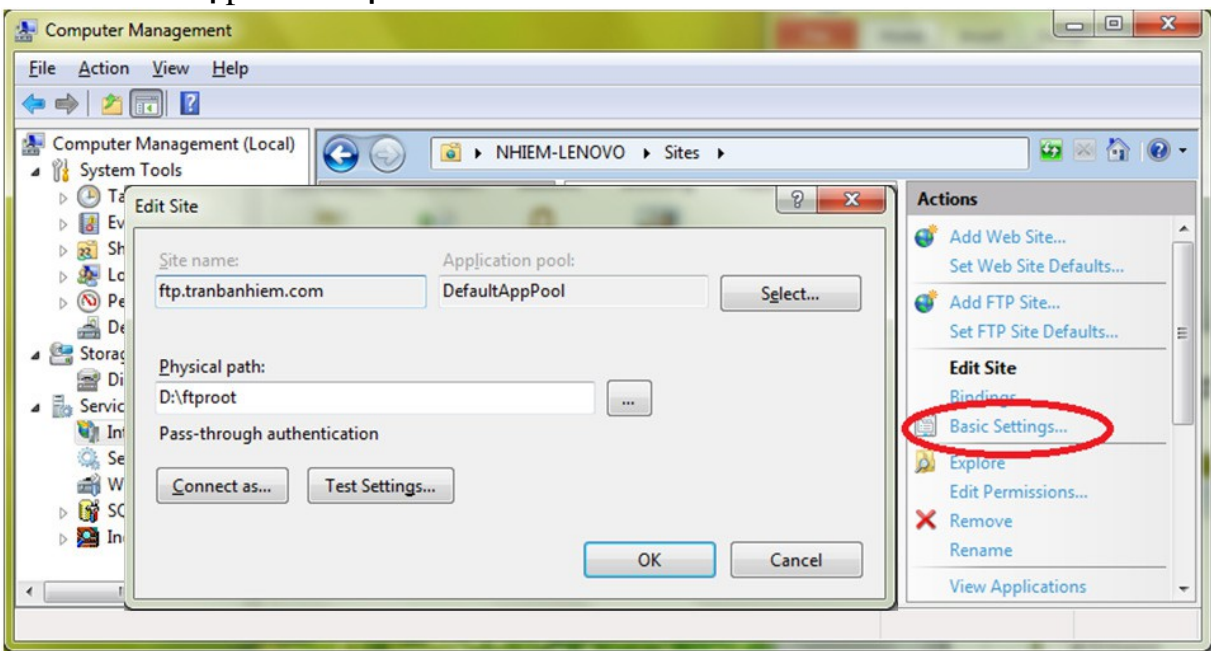


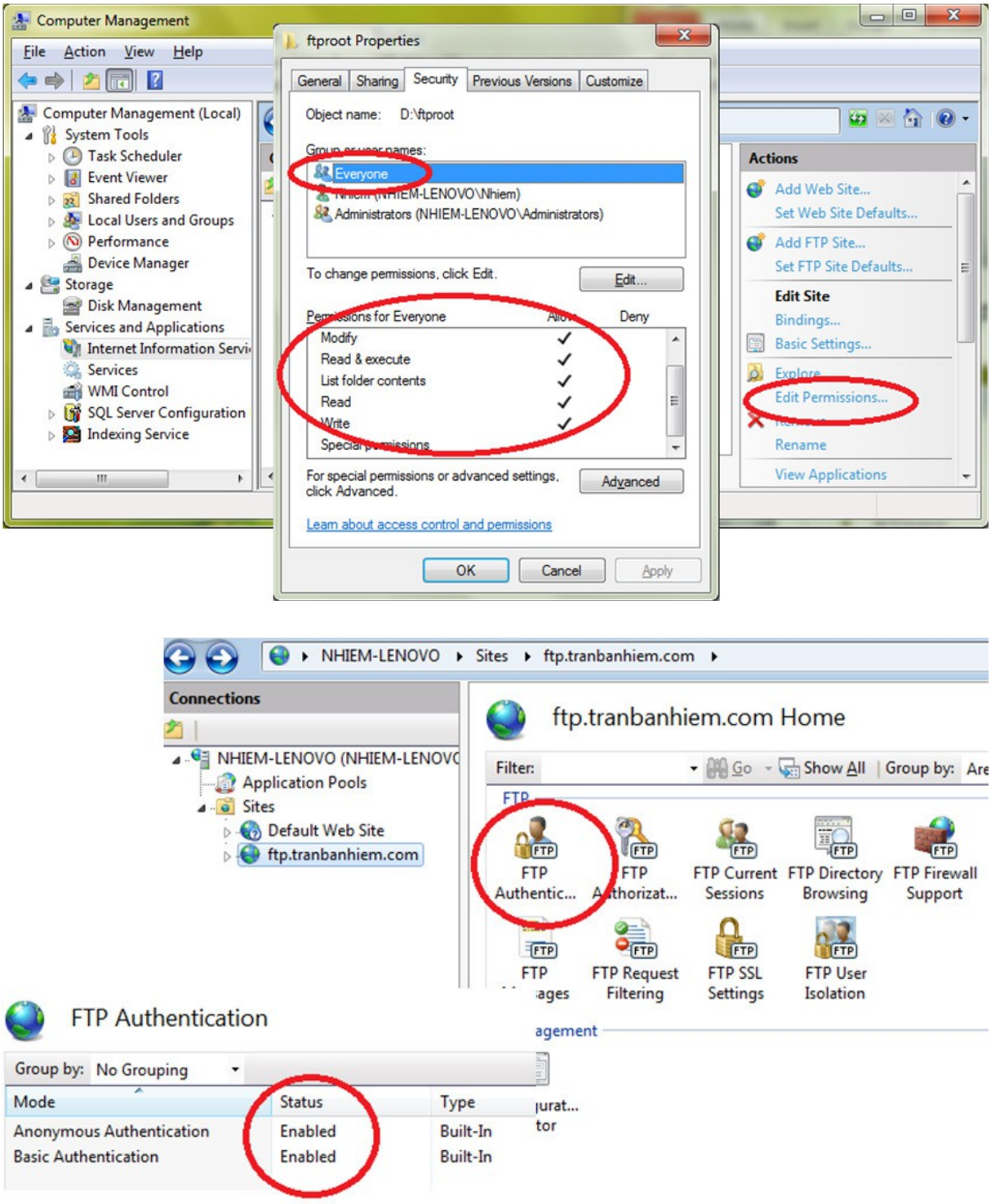
Để đến phần quản trị FTP server thực hiện như sau :

Click Control Panel→Administrative Tools→Internet Information Services→FTP Site→Default FTP site → right-click → chọn Properties.



Thiết lập các thuộc tính :



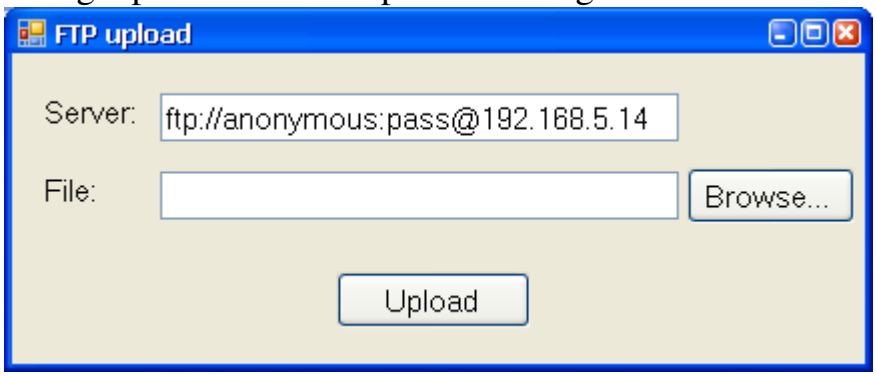


2.6. Công cụ FTP với điều khiển trên Internet

Thực thi đầy đủ của FTP là một cam kết. Nó có thể là đáng giá để xem xét kiểm soát Microsoft Chuyển Internet nếu bạn cần phải thực hiện nhiệm vụ này. Nó là một COM di sản kiểm soát (và do đó mang rất nhiều chi phí cho các ứng dụng NET.). Bản địa. Thành phần NET là có sẵn trên thị trường từ Dart và công trình * IP. Có nói rằng, đối với nhiều ứng dụng, bạn không cần phải hát tất cả,

thực hiện của FTP để có được công việc của bạn được thực hiện. Nếu bạn đang viết một tính năng để một ứng dụng để lên kế hoạch thực hiện một tải lên các tập tin đến một máy chủ, bạn có thể không muốn gây nhầm lẫn cho người sử dụng với các chi tiết của cấu trúc thư mục của máy tính từ xa. Tất cả những gì bạn có thể cần một vài dòng mã để chuyển các tập tin đến một vị trí được xác định trước.

Tạo một dự án ứng dụng Windows mới trong Visual Studio NET như bình thường, và rút ra hai hộp văn bản, một trong những tên tbServer và tbFile khác. Thêm hai nút, btnBrowse và btnUpload. Bạn cũng sẽ yêu cầu một kiểm soát Dialog Open File tên là OpenFileDialog




```
 private void btnBrowse_Click(object sender, EventArgs e)
      {
         openfileDialog.ShowDialog();
         tbFile.Text = openfileDialog.FileName;
      }
private void btnUpload_Click(object sender, EventArgs e)
      {
         if ((tbFile.Text.Trim() == ""))
         {
            MessageBox.Show("Please choose a file!", "Error", MessageBoxButtons.OK,
MessageBoxIcon.Error);
            return;
         }

         FileInfo thisFile = new FileInfo(tbFile.Text);
         Type ITC;
         object[] parameter = new object[2];
         object ITCObject;
         ITC = Type.GetTypeFromProgID("InetCtls.Inet.1");
         ITCObject = Activator.CreateInstance(ITC);
         parameter[0] = (string)tbServer.Text;
         parameter[1] = (string)"PUT " + thisFile.FullName + " /" + thisFile.Name;
         ITC.InvokeMember("execute", BindingFlags.InvokeMethod, null, ITCObject,
parameter);


      }
```

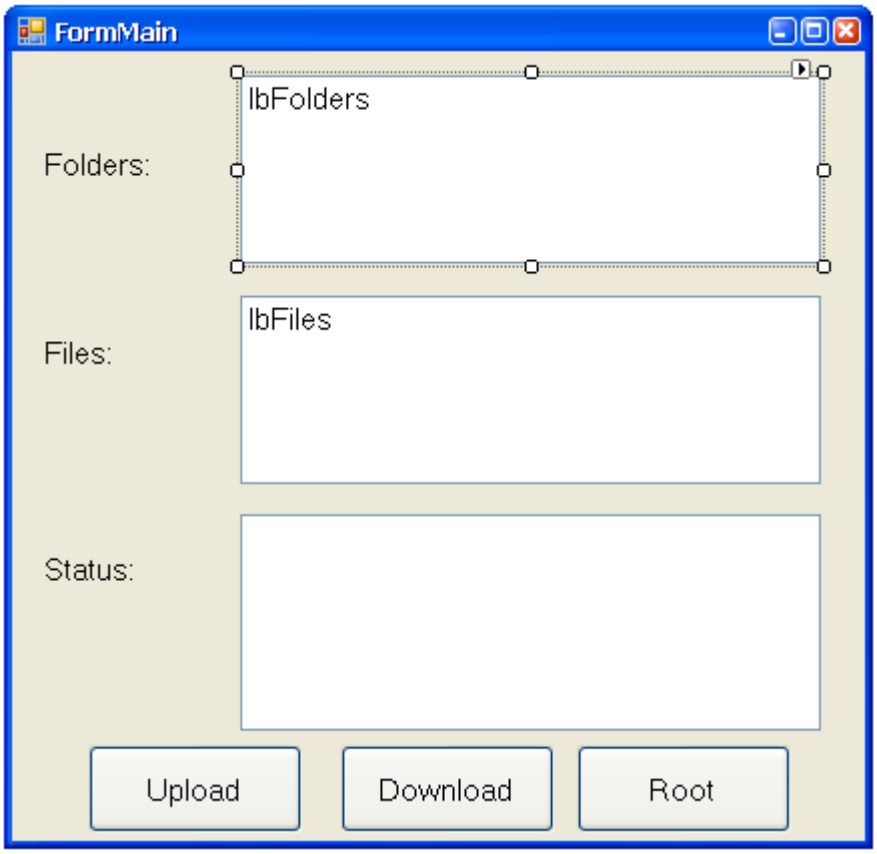
2.7. Một vài công cụ thực tế của FTP

ITC có một vài hạn chế, có khá một vài lỗi nổi tiếng, và là xa là một việc thực hiện hiệu suất cao. Hơn nữa, nó là không có nguồn gốc NET, và nhiều nhà phần mềm sẽ yêu cầu một NET dự án là mã quản lý 100%.

Bằng cách làm theo các mã trên các trang tiếp theo, bạn sẽ có đầy đủ FTP client, với khả năng để duyệt một hệ thống tập tin từ xa, tải lên, và tải về Bắt đầu một dự án mới trong Visual Studio. NET và thêm hai hình thức, frmLogon và frmMain. Về hình thức Logon, rút ra bốn hộp văn bản: tbServer, tbUsername, tbPassword, và tbStatus. Sau đó nên được thiết lập với = multiline đúng sự thật và màu xám ra ngoài một cách thích hợp. Một nút, btnLogon, cũng nên được thêm vào Trên các hình thức chính, vẽ hai danh sách hộp:. LbFiles và lbFolders. Thêm một hộp văn bản tên là tbStatus trong phong cách tương tự như trong các hình thức Logon. Thêm ba nút: btnUpload, btnDownload, và btnRoot. Ngoài ra thêm một hộp thoại File Open kiểm soát tên là OpenFileDialog và Save File Dialog kiểm soát đặt tên là SaveFileDialog.

Trong Form chính, thêm một vài biến toàn cục:

```
private const string STR_Constant = "\r\n";
public frmLogon LogonForm = new frmLogon();
public NetworkStream NetStrm;
public string RemotePath = "";
public string server = "";
```



```
    public string sendFTPcmd(string cmd)
{
    byte[] szData;
    string returnedData = "";
    StreamReader RdStrm = new StreamReader(NetStrm);
    szData = Encoding.ASCII.GetBytes(cmd.ToCharArray());
    NetStrm.Write(szData, 0, szData.Length);
    tbStatus.Text += "\r\nSent:" + cmd;
    returnedData = RdStrm.ReadLine();
    tbStatus.Text += "\r\nRcvd:" + returnedData;
    return returnedData;
}

public void getRemoteFolders()
{
    string[] filesAndFolders;
    string fileOrFolder;
    string folderList = "";
    int lastSpace = 0;
    folderList =
    Encoding.ASCII.GetString(sendPassiveFTPcmd("LIST\r\n"));
    lbFiles.Items.Clear();
    lbFolders.Items.Clear();
    filesAndFolders = folderList.Split("\n".ToCharArray());
    for (int i = 0; i < filesAndFolders.GetUpperBound(0); i++)
    {
        if (filesAndFolders[i].StartsWith("-") ||
        filesAndFolders[i].StartsWith("d"))
        {
            lastSpace = 59; // UNIX format
        }
        else
        {
            lastSpace = 39; // DOS format
        }
        fileOrFolder = filesAndFolders[i].Substring(lastSpace);
        if (fileOrFolder.IndexOf(".") != -1)
        {
            lbFiles.Items.Add(fileOrFolder.Trim());
        }
        else
        {
            lbFolders.Items.Add(fileOrFolder.Trim());
        }
    }
}

public byte[] sendPassiveFTPcmd(string cmd)
{
    byte[] szData;
```

```
    System.Collections.ArrayList al = new ArrayList();
    byte[] RecvBytes = new byte[Byte.MaxValue];
    Int32 bytes;
    Int32 totalLength = 0;
    szData =
    System.Text.Encoding.ASCII.GetBytes(cmd.ToCharArray());
    NetworkStream passiveConnection;
    passiveConnection = createPassiveConnection();
    tbStatus.Text += "\r\nSent:" + cmd;
    StreamReader commandStream = new StreamReader(NetStrm);
    NetStrm.Write(szData, 0, szData.Length);
    while (true)
    {
        bytes = passiveConnection.Read(RecvBytes,
        0, RecvBytes.Length);
        if (bytes <= 0) break;
        totalLength += bytes;
        al.AddRange(RecvBytes);
    }
    al = al.GetRange(0, totalLength);
    tbStatus.Text += "\r\nRcvd:" + commandStream.ReadLine(); // 125
    tbStatus.Text += "\r\nRcvd:" + commandStream.ReadLine(); // 226
    return (byte[])al.ToArray((new byte()).GetType());
}

private NetworkStream createPassiveConnection()
{
    string[] commaSeperatedValues;
    int highByte = 0;
    int lowByte = 0;
    int passivePort = 0;
    string response = "";
    TcpClient clientSocket;
    NetworkStream pasvStrm = null;
    response = sendFTPcmd("PASV\r\n");
    // 227 Entering Passive Mode (127,0,0,1,4,147).
    commaSeperatedValues = response.Split(",".ToCharArray());
    highByte = Convert.ToInt16(commaSeperatedValues[4]) * 256;
    commaSeperatedValues[5] =
    commaSeperatedValues[5].Substring(0,
    commaSeperatedValues[5].IndexOf(")"));
    lowByte = Convert.ToInt16(commaSeperatedValues[5]);
    passivePort = lowByte + highByte;
    clientSocket = new TcpClient(server, passivePort);//TcpClient
    pasvStrm = clientSocket.GetStream();//NetworkStream
    return pasvStrm;
}

private void lbFolders_SelectedIndexChanged(object sender, EventArgs e)
{
```

```
    RemotePath += "/" + lbFolders.SelectedItem.ToString();
    sendFTPcmd("CWD /" + RemotePath + STR_Constant);
    getRemoteFolders();
}

private void btnRoot_Click(object sender, EventArgs e)
{
    RemotePath = "/";
    sendFTPcmd("CWD /\r\n");
    getRemoteFolders();
}

private void btnUpload_Click(object sender, EventArgs e)
{
    openFileDialog.ShowDialog();
    NetworkStream passiveConnection;
    FileInfo fileParse = new FileInfo(openFileDialog.FileName);
    FileStream fs = new
    FileStream(openFileDialog.FileName, FileMode.Open);
    byte[] fileData = new byte[fs.Length];
    fs.Read(fileData, 0, (int)fs.Length);
    passiveConnection = createPassiveConnection();
    string cmd = "STOR " + fileParse.Name + "\r\n";
    tbStatus.Text += "\r\nSent:" + cmd;
    string response = sendFTPcmd(cmd);
    tbStatus.Text += "\r\nRcvd:" + response;
    passiveConnection.Write(fileData, 0, (int)fs.Length);
    passiveConnection.Close();
    MessageBox.Show("Uploaded");
    tbStatus.Text += "\r\nRcvd:" + new
    StreamReader(NetStrm).ReadLine(); getRemoteFolders();
}

private void btnDownload_Click(object sender, EventArgs e)
{
    byte[] fileData;
    saveFileDialog.ShowDialog();
    fileData = sendPassiveFTPcmd(
          "RETR " + lbFiles.SelectedItem.ToString() + "\r\n");
    FileStream fs = new FileStream(
            saveFileDialog.FileName, FileMode.CreateNew);
    fs.Write(fileData, 0, fileData.Length);
    fs.Close();
    MessageBox.Show("Downloaded");
}
```

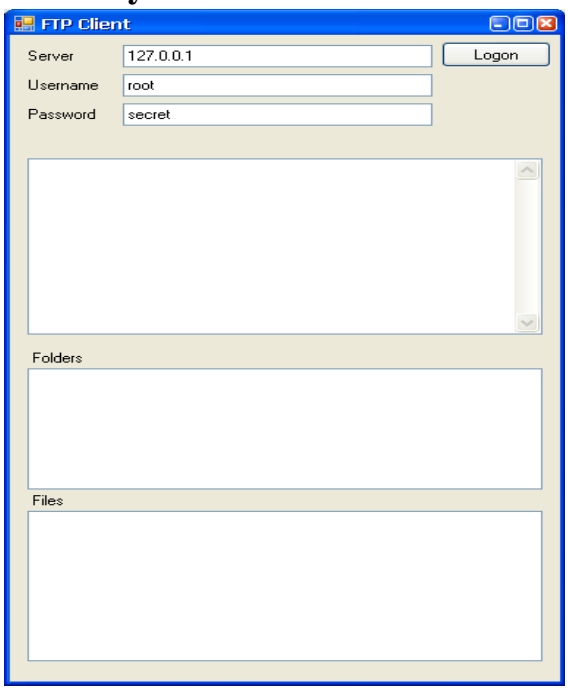
2.8. FTP hỗ trợ trong .NET 2.0

NET Framework phiên bản 2.0 (Whidbey), hỗ trợ FTP được bao gồm trong lớp WebClient, do đó phủ định sự cần thiết phải sử dụng một trong hai cấp lập trình hoặc các đối tượng COM. Các mã sau đây minh họa đơn giản số tính năng mới này:

```
public void downloadFile()
        {
                string filename = "ftp://ms.com/files/dotnetfx.exe";
                WebClient client = new WebClient();
                client.DownloadFile(filename,"dotnetfx.exe");
        }
```

BÀI THỰC HÀNH

Xây dựng chương trình truyền File từ Client lên server



Mã chương trình

```
private void Logs(string message)
   {
    FileStream file = new FileStream("log.txt", FileMode.Create, FileAccess.Write);
    byte[] data = Encoding.ASCII.GetBytes(message);
    file.Write(data, 0, data.Length);
    file.Flush();
    file.Close();
   }

   NetworkStream NetStrm;
   private void btnLogon_Click(object sender, EventArgs e)
   {
    tbStatus.Text = "";
    // Khởi tạo client kết nối đến cổng của dịch vụ FTP (21)
```

```
 TcpClient clientSocket = new TcpClient(tbServer.Text, 21);
 // Khởi tạo luồng xuất/nhập mạng và kết nối nó đến đối tượng client
 NetStrm = clientSocket.GetStream();
 // Lấy luồng nhập (đọc) từ luông xuất/nhập mạng
 StreamReader RdStrm = new StreamReader(NetStrm);
 string welcomeMessage = RdStrm.ReadLine() + "\r\n";
 tbStatus.Text += welcomeMessage;
 string returnMessage = "";
 // Tiến hành gởi thông tin đăng nhập lên POP3 Server
 returnMessage = sendFTPcmd("USER " + tbUsername.Text + "\r\n", NetStrm);
 returnMessage = sendFTPcmd("PASS " + tbPassword.Text + "\r\n", NetStrm);
 // 1. Lấy thông số hệ thống, lệnh SYST
 sendFTPcmd("SYST" + "\r\n", NetStrm);
 sendFTPcmd("QUIT" + "\r\n", NetStrm);
 clientSocket.Close();
 // Nhớ gọi hàm này
 Logs(tbStatus.Text);
}
public string sendFTPcmd(string cmd, NetworkStream NetStrm)
{
 byte[] szData;
 string returnedData = "";
 StreamReader RdStrm = new StreamReader(NetStrm);
 szData = Encoding.ASCII.GetBytes(cmd.ToCharArray());
 NetStrm.Write(szData, 0, szData.Length);
 tbStatus.Text += "\r\nSent:" + cmd;
 returnedData = RdStrm.ReadLine();
 tbStatus.Text += "\r\nRcvd:" + returnedData;
 return returnedData;
                                                    }
```

BÀI 7 : AN NINH MẠNG (FIREWALLS, PROXY SERVERS, AND ROUTERS)

Mã bài M35.06

Mục tiêu của bài:

- Mô tả cách xây dựng mạng lưới bảo vệ mạng;
- Trình bày được các cạm bẩy mạng;
- Thực hiện được xây dựng ứng dụng đơn giản bảo đảm an toàn hệ thống mạng trong doanh nghiệp.
- Thực hiện các thao tác an toàn với máy tính.

1. Tổng quan về bảo vệ mạng

1.1. Giới thiệu về An ninh mạng

Bài này đề cập tới các vấn đề thực tiễn của việc thiết lập một mạng và kiến trúc mạng. Việc nắm bắt được cấu trúc mạng giúp cho sinh viên biết làm thế nào để sửa chữa rất nhiều lỗi liên quan đến ứng dụng mạng. Hơn nữa, kiến thức cơ bản về thiết lập mạng là điều cần thiết.

Bài này gồm hai phần. Phần đầu tiên giải thích làm thế nào để tạo ra một mạng lưới từ máy độc lập. Ngay sau đó là thảo luận về các thiết bị phổ biến hình thành cửa ngõ giữa mạng của bạn và Internet. Các thiết bị cổng thường có thể tạo ra vấn đề cho phần mềm của bạn bằng cách áp đặt các hạn chế và các quy định riêng. Bằng cách có thể để phát hiện và làm việc xung quanh những vấn đề này, ứng dụng của bạn sẽ được ổn định hơn.

1.2. Xây dựng mạng lưới an ninh ngay từ đầu

Nếu bạn đang phát triển một hệ thống điểm bán hàng cho một siêu thị, mỗi đầu cuối sẽ cần phải giao tiếp với một máy chủ trung tâm. Đây không phải là dễ dàng đạt được mà không có mạng. Trong nhiều trường hợp, bạn có thể không chỉ cung cấp cho một người chủ tiệm một đĩa CD và hy vọng anh ta tìm ra cách để nối mạng được tất cả các máy tính của mình.

Lựa chọn một cấu trúc mạng

Nếu bạn chỉ có hai máy tính mà bạn muốn vào mạng, thì giải pháp kinh tế nhất là một cặp xoắn không được che chở (UTP) cáp chéo (cáp UTP bản vá). Điều này có thể được sử dụng để liên kết hai máy tính trực tiếp.

Có ba loại kết nối vật lý chính trong mạng hiện đạ là UTP, BNC, và không dây. Mạng không dây sử dụng sóng radio để truyền dữ liệu giữa các thiết bị đầu cuối, trong khi hai hệ thống khác sử dụng kết nối có dây.

Những lợi ích của một mạng không dây là khá rõ ràng. Người dùng có thể di chuyển trong vòng bán kính của máy phát và duy trì một kết nối Internet, tuy nhiên, mạng không dây là chậm hơn so với có dây của họ truy

cập các bộ phận. Ví dụ, một card mạng điển hình có thể hỗ trợ kết nối 100 Mbps, trong khi các card không dây tương đương sẽ hoạt động ở 11 hoặc 54 Mbps, và thông qua thực tế chỉ có thể là một phần nhỏ trong số đó. Một cáp mạng có thể dễ dàng kéo dài 100 mét, nhưng trung tâm không dây có bán kính nhỏ hơn. Các mạng không dây có nhiều tốn kém, nhưng khá tương đồng trong kiến trúc với một mạng UTP.

Sự khác biệt giữa UTP và BNC là rõ ràng nhất trong các loại cáp được sử dụng để kết nối các máy tính. UTP cáp giống như một đường dây điện thoại, chỉ mỏng hơn, trong khi BNC đồng trục, cáp truyền hình. Phích cắm BNC tròn, trong khi phích cắm UTP (RJ45 kết nối) có hình chữ nhật.

UTP được đặt ra trong một cấu trúc liên kết sao, nơi mà mỗi máy tính có một dòng dành riêng cho trung tâm gần nhất hoặc bộ định tuyến của nó. Trong các mạng nhỏ hơn, một trong các máy tính trên mạng sử dụng một modem (hoặc thiết bị khác) để kết nối với các ISP. Tất cả các máy khác trên mạng sau đó chia sẻ kết nối Internet. Trên các mạng lớn hơn, một bộ định tuyến kết nối trực tiếp với một đường dây cung cấp bởi nhà cung cấp dịch vụ. Sự sắp xếp này cung cấp hiệu suất tốt hơn bởi vì các bộ định tuyến sẽ giúp chỉ đạo các dữ liệu, cũng như được dành riêng cho nhiệm vụ cung cấp một kết nối mạng lưới làm việc, tuy nhiên, nó bổ sung thêm chi phí vào mạng.

BNC được đặt ra trong một cấu trúc liên kết bus. Đây là nơi mà tất cả các máy tính trên mạng chia sẻ một dòng duy nhất của truyền thông. Trong một mạng BNC, mỗi máy tính có kết nối hình chữ T gắn liền với card mạng của nó. Tại mỗi đầu của dây là terminator. BNC là hiếm hiện nay, và nó là com-Th 2 để sử dụng hoặc UTP hoặc không dây.

Các mạng khác, dựa trên Universal Serial Bus (USB) và các nối tiếp connec, có sẵn, nhưng họ cần phải tránh vì erability vấn đề có thể interop.

Thiết lập mạng máy tính

Khi xây dựng một mạng UTP, đảm bảo rằng mỗi máy tính được nối với một hub, và chắc chắn rằng trung tâm này được hỗ trợ. Trong một mạng BNC, mỗi máy tính được kết nối với hàng xóm của mình, và một kết thúc BNC sẽ được dán vào cuối của dây.

Người dùng sẽ mong đợi một cơ chế chia sẻ file trên mạng, vì vậy bạn nên cung cấp ngay từ đầu. Để cung cấp cơ chế này, bạn phải chọn một tên duy nhất cho mỗi máy tính trên mạng. Để đặt tên cho một máy tính trên Windows 2000, kích chuột phải vào My Computer→Properties→Network Identification, và chọn Properties. Đối với Windows XP, chọn My Computer→Properties→Computer Name→Change.

Thay đổi tên máy tính

Nhập vào tên máy tính, và nếu cần thiết, một nhóm làm việc. Sau đó nhấn
OK. Bạn có thể cần phải khởi động lại máy tính để các thay đổi có hiệu lực. Bạn sẽ cần phải ràng buộc một số giao thức và dịch vụ mới cho card mạng của bạn. Để làm điều này trong Windows 2000, kích chuột phải vào My Network Places→Properties→Local Area Connection→Properties. On Windows XP, click Control Panel→Network Connections→Local Area Connection. Trong hộp này, bạn cần phải nhìn thấy ba điều: Client for Microsoft Networks, File and Printer Sharing for Microsoft Networks, và Internet Protocol (TCP / IP). Nếu bất kỳ trong số này là mất tích, nhấn nút Install.

Nhiệm vụ tiếp theo là thiết lập các thiết lập TCP / IP cho máy tính. Để mở hộp thoại, đánh dấu Internet Protocol (TCP / IP) và kích Properties.

Nếu máy tính này là một phần của một mạng lưới lớn hơn, có thể là một máy chủ DHCP trên mạng, tự động gán địa chỉ IP. Trong trường hợp này, chọn tùy chọn "Có được một địa chỉ IP tự động" và "Xin địa chỉ máy chủ DNS tự động." Nếu không, thiết lập các lĩnh vực thủ công.

Bạn phải thiết lập địa chỉ IP là địa chỉ không công khai, và mỗi máy tính phải được chỉ định một địa chỉ IP duy nhất. Một loạt các địa chỉ IP có thể là 10.0.0.1, 10.0.0.2, 10.0.0.3,…. Subnet mask 255.255.255.0. Bấm OK để lưu các thiết lập.

Để chia sẻ một thư mục, nhấn chuột phải vào thư mục, chọn Properties → Sharing. Chia sẻ thư mục này (trên Windows XP, bạn sẽ cần bấm nhắn dis-claimer, "Nếu bạn hiểu được những rủi ro nhưng vẫn muốn chia sẻ con lều của thư mục này").

Để hạn chế người dùng truy cập từ xa cho các tập tin của bạn trên Windows 2000, Per-nhiệm vụ. Trên cửa sổ tiếp theo bạn có thể cấp, thu hồi

đọc, viết, và thay đổi quyền truy cập vào bất kỳ hoặc tất cả người dùng trên mạng. Trên Windows XP, điều này đã được đơn giản hóa một hộp kiểm "Cho phép người sử dụng mạng để thay đổi các tập tin của tôi."

Một tính năng hữu ích của mạng là khả năng điều khiển từ xa in tài liệu thông. Phần này giả định rằng bạn có một máy in kèm theo một máy tính trên mạng của bạn. Trên Windows 2000, nhấp vào Start → Cài đặt → Máy in. Trên Windows XP, nhấn Start → Control Panel → Printers and Faxes. Nhấp chuột phải vào một máy in mà bạn muốn chia sẻ, và chọn tùy chọn Sharing. Sau đó chọn chung Khi nhập một tên duy nhất, và một tên mô tả cho máy in. Bạn có thể thiết lập mức độ kiểm soát của người sử dụng sẽ có hơn máy in từ tab Security. Bấm OK để hoàn tất quá trình.

Làm thế nào để thiết lập một mạng riêng ảo

Một mạng riêng ảo (VPN) được sử dụng để cung cấp cho một khách hàng truy cập từ xa an toàn đến một mạng LAN. Các khách hàng từ xa sẽ có minh bạch (mặc dù, chậm hơn) truy cập vào mạng LAN và sẽ có thể chia sẻ các tập tin và sử dụng máy in từ xa.

VPN hoạt động trên các đường hầm giao thức điểm-điểm (PPTP) hay giao thức lớp 2 đường hầm (L2TP). Giao thông địa phương được xếp lớp trên đầu trang này để hỗ trợ minh bạch thực sự và hỗ trợ cho giao thức nonroutable chẳng hạn như IPX.

VPN có một số lợi thế hơn dial-in kết nối vào mạng. Đây là những bảo mật, nơi mà mỗi truyền được mã hóa, và minh bạch, bởi vì các khách hàng có thể giữ lại địa chỉ IP riêng của nó.

Để trở thành một khách hàng VPN, Windows 2000, nhấn Start → Settings → Net-công việc kết nối, và sau đó nhấp vào New Connection Wizard. Trên Windows XP, hãy nhấp vào Bắt đầu → Control Panel → Network Connections → Tạo một Con-nection → Tiếp theo.

Bấm vào "Kết nối với một mạng riêng thông qua Internet" Win-dows 2000 hoặc "Kết nối với mạng tại nơi làm việc của tôi", sau đó kết nối mạng ảo Pri-vate trên Windows XP.

Khi được nhắc, nhập vào địa chỉ IP của gateway VPN. Điều này nên được cung cấp bởi người quản trị của VPN. Nhấn Finish để hoàn tất việc cài đặt.

2. Tunneling trong mạng doanh nghiệp

Luôn luôn có hai cách để sửa chữa một vấn đề: giải quyết hoặc tránh nó. Cả hai phương pháp có giá trị ngang nhau và bình đẳng đối với các tình huống khác nhau. Giả sử tình huống là một ứng dụng hội thảo từ xa bị chặn bởi tường lửa. Bạn có thể di chuyển các máy chủ bên ngoài tường lửa, thiết lập cổng chuyển tiếp để đường hầm thông qua các bức tường lửa (hoặc router), hoặc trả lại dữ liệu ra một proxy máy chủ để tránh các bức tường

lửa. Có hai tuỳ chọn, một là sửa chữa trực tiếp (on-site) hoặc thuê một máy chủ chuyên dụng và việc một số chương trình để giải quyết vấn đề.

Proxy Tunneling

Nếu viết một ứng dụng cho thị trường, bạn phải nhớ rằng không phải tất cả người sử dụng phần mềm sẽ có một trong hai kết nối Internet trực tiếp, minh bạch. Trong một số trường hợp, người dùng có thể truy cập Internet thông qua một proxy. Thật không may, không có phương tiện hoàn hảo phát hiện nếu một proxy là sử dụng trên một mạng, nó ở đâu, hoặc những gì nó là loại.

Không giống như thiết bị định tuyến, proxy không minh bạch cho khách hàng. Bạn sẽ cần phải sửa đổi mã của bạn vào tài khoản cho một proxy. Nếu bạn đang sử dụng các HTTPWebRe nhiệm vụ và đang cố gắng để điều hướng một proxy ứng dụng, sau đó điều này là tương đối đơn giản:

```
WebProxy myProxy= new WebProxy("proxyserver",8080);
myProxy.BypassProxyOnLocal =
true; String url =
"http://www.yahoo.com";
HttpWebRequest request =
(HttpWebRequest)HttpWebRequest.Create(url);
request.Proxy = myProxy;
```

Firewall tunneling

Nếu một bức tường lửa ngăn chặn tất cả các cổng, sau đó bạn có thể làm thay đổi các bức tường lửa cho phép truy cập vào cổng yêu cầu của bạn. Các tường lửa được truy cập hoặc thông qua một giao diện Web (http://192.168.1.1 hoặc tương tự) hoặc thông qua một kết nối nối tiếp. Bạn sẽ cần phải có hướng dẫn sử dụng và mật khẩu trong tầm tay. Một số router cung cấp cổng chuyển tiếp để vượt qua tường lửa. Đây là nơi mà các dữ liệu trực tiếp tại địa chỉ IP của router trên một cổng được chỉ định là ngăn chặn, đến một địa chỉ IP nội bộ quy định. Quá trình này là minh bạch cho cả hai đầu của kết nối.

Cuối cùng, nếu bạn không có quyền truy cập vào các bức tường lửa, hoặc bạn muốn cung cấp một giải pháp thân thiện với người sử dụng, bạn có thể trả lại dữ liệu từ một proxy. Đây là nơi mà các máy phía sau tường lửa sẽ mở ra một TCP ổn định và kết nối đến một máy chủ proxy, mà là bên ngoài của các bức tường lửa và proxy cho phép khách hàng để kết nối với nó. Dữ liệu từ các khách hàng đến proxy được chuyển tiếp thông qua kết nối previ-

ously mở. Đây là kỹ thuật được sử dụng bởi các ứng dụng Instant Messenger. Một ví dụ mã của giải pháp này được cung cấp ở cuối chương này.

3. Tránh những cạm bẫy mạng

Phòng chống luôn luôn tốt hơn chữa bệnh.Nếu bạn đang phát hành một sản phẩm tự nhiên, nó là gần như chắc chắn rằng một số người sử dụng sẽ có một cấu hình mạng khác thường rằng phần mềm của bạn sẽ không làm việc. Đối với họ, mạng lưới của họ không phải là bất thường, và trong thực tế, một trăm người sử dụng khác ra khỏi đó có cùng một vấn-lem, nhưng họ không bận tâm để cho bạn biết rằng phần mềm của bạn không làm việc.

Cổng xung đột

Nếu phần mềm của bạn không thể bắt đầu trên cổng mặc định của nó, nó phải di chuyển đến cảng khác, hoặc ít nhất là nhắc nhở người dùng nhập vào một cảng mới. Nếu bạn không cung cấp chức năng này, bạn sẽ gặp phải một trong hai vấn đề: (1) người dùng sẽ chạy phần mềm sử dụng cùng một cổng như của bạn và rằng họ không muốn ngừng sử dụng, hoặc (2) bức tường lửa có thể đã được thiết lập để cho phép lưu lượng truy cập thông qua một số cảng, ngay cả khi khách hàng của bạn không sử dụng tường lửa, ISP của họ sức mạnh.

Các khách hàng đang chờ đợi để kết nối với phần mềm của bạn sẽ cần phải biết rằng nó đã di chuyển cảng. Bạn chỉ có thể hiển thị một hộp thông báo và yêu cầu người dùng nhập vào các cảng mới, hoặc bạn có thể sử dụng một yêu cầu DNS để nói với người sử dụng các cổng máy chủ lắng nghe và kết nối với nhau lần lượt. Nói chung, phương pháp này là quá mức cần thiết.

Một vấn đề thường xuyên gặp phải là địa chỉ IP động. Đây là nơi mà các địa chỉ IP của máy tính thay đổi mỗi khi nó đi trực tuyến. Nếu không được, nhiều ứng dụng sẽ lấy địa chỉ IP địa phương khi bắt đầu ứng dụng và giả định rằng sẽ vẫn tĩnh trong khoảng thời gian cuộc sống của ứng dụng. Khi người dùng kết nối quay số, họ có thể có được địa chỉ IP khác nhau trong không gian của một giờ theo sử dụng bình thường (đăng ký trên Internet). Tình trạng này đặt ra một vấn đề cho các ứng dụng máy chủ bởi vì không có cách nào một khách hàng có thể biết nó ở đâu nên kết nối. Điều này có thể được giải quyết trên cơ sở từng trường hợp cụ thể hoặc bởi máy chủ-ing một cơ chế theo dõi IP.
Phần mềm chẳng hạn như "IP-no" có thể được sử dụng để ánh xạ một địa chỉ IP động cho một tên DNS. Quá trình sử dụng phần mềm này là tương đối đơn giản, nhưng nó có thể không khả thi để yêu cầu người sử dụng phần mềm để sử dụng sản phẩm này để giải quyết các vấn đề địa chỉ IP động. Cách khác là có máy tính định kỳ gửi địa chỉ IP của nó vào một máy chủ, sau đó, máy chủ sẽ lưu trữ các địa chỉ IP, cùng với một dấu thời gian và nhận dạng một con người có thể đọc được. Khách hàng có thể tìm và kết nối địa chỉ IP động. Tem thời gian đảm bảo rằng các máy tính chưa có mặt trong diễn

đàn sẽ bị xóa khỏi danh sách.

Khi đăng một địa chỉ IP, phải thực hiện kiểm tra để đảm bảo rằng IP là hợp lệ cho Internet. Một mạng LAN IP như 192.168.0.1 là không tốt với một khách hàng ở phía bên kia của thế giới.

BỘ CÂU HỎI ÔN TẬP LẬP TRÌNH MẠNG

I. PHẦN LÝ THUYẾT

Câu 1 : Nêu khái niệm về lập trình Socket ?

Câu 2 : Nêu khái niệm về IP và Port trong lập trình mạng

Câu 3 : Nêu một số ứng dụng và qui định đối với Port ?

Câu 4 : Nêu các đặc trưng cơ bản (Khái niệm, các thuộc tính và phương thức) của lớp IPAddress?

Câu 5 : Nêu các đặc trưng cơ bản (Khái niệm, các thuộc tính và phương thức) của lớp IPEndPoint?

Câu 6 : Nêu các đặc trưng cơ bản (Khái niệm, các thuộc tính và phương thức) của lớp IPHostEntry?

Câu 7 : Nêu các đặc trưng cơ bản (Khái niệm, các thuộc tính và phương thức) của lớp DNS?

Câu 8 : Nêu các đặc trưng cơ bản (Khái niệm, các thuộc tính và phương thức) của lớp UDPClient?

Câu 9 : Nêu trình tự kết nối của Lớp UDPClient? (Phía Server và Client)

Câu 10 : Nêu các đặc trưng cơ bản (Khái niệm, các thuộc tính và phương thức) của lớp TCPClient?

Câu 11 : Nêu trình tự kết nối của Lớp TCPClient? (Phía Server và Client)

Câu 12 : Nêu các đặc trưng cơ bản (Khái niệm, các thuộc tính và phương thức) của lớp TCPListener?

Câu 13 : Nêu các đặc trưng cơ bản (Khái niệm, các thuộc tính và phương thức) của lớp TCPListener?

Câu 14 : So sánh giữa hai giao thức TCP và UDP trong lập trình mạng.

Câu 15 : Nêu phương thức hoạt động của HTTP khi thực hiện truyền thông với webserver.

Câu 16 : Nêu các đặc trưng cơ bản (Khái niệm, các thuộc tính và phương thức) của lớp HTTP request?

Câu 17 : Nêu các đặc trưng cơ bản (Khái niệm, các thuộc tính và phương thức) của lớp HTTP responese?

Câu 18 : Nêu cách thức để truyền thông với File Server

Câu 19 : Nêu các đặc trưng cơ bản (Khái niệm, các thuộc tính và phương thức) của lớp FTP?

Câu 20 : Nêu tiến trình bắt tay làm việc của giao thức truyền file FTP

Câu 21 : Viết chương trình thực hiện trên mạng LAN để giải quyết bài toán sau :

Client : gởi 2 số a, b cho server và yêu cầu server trả về số nào lớn nhất.

Server : so sánh hai số nhận được và trả về kết quả lớn nhất cho Client.

Yêu cầu : chỉ viết các hàm chính ở Client và Server, dùng TCP hoặc UDP.

Câu 22 : Viết chương trình thực hiện trên mạng LAN để giải quyết bài toán sau :

Client : gởi 2 số a, b cho server và yêu cầu server trả về kết quả a có chia hết cho b hay không?

Server : Nhận 2 số a, b và thực hiện phép toán chia, trả về kết quả cho Client.

Yêu cầu : chỉ viết các hàm chính ở Client và Server, dùng TCP hoặc UDP.

Câu 23 : Viết chương trình thực hiện trên mạng LAN để giải quyết bài toán sau :

Client : gởi số a cho server và yêu cầu server trả về kết quả a có chia hết cho 3 hay không?

Server : Nhận số a và thực hiện phép toán chia, trả về kết quả cho Client.

Yêu cầu : chỉ viết các hàm chính ở Client và Server, dùng TCP hoặc UDP.

Câu 24 : Viết chương trình thực hiện trên mạng LAN để giải quyết bài toán sau :

Client : gởi số a cho server và yêu cầu server trả về kết quả a là số lẻ hay chẵn?

Server : Nhận số a và thực hiện phép toán chia, trả về kết quả cho Client.

Yêu cầu : chỉ viết các hàm chính ở Client và Server, dùng TCP hoặc UDP.

Câu 25 : Viết chương trình thực hiện trên mạng LAN để giải quyết bài toán sau :

Client : gởi 3 số a, b cho server và yêu cầu server trả về số lớn nhất?

Server : Nhận 2 số a, b và thực hiện phép toán so sánh, trả về kết quả cho Client.

Yêu cầu : chỉ viết các hàm chính ở Client và Server, dùng TCP hoặc UDP.

II. PHẦN THỰC HÀNH

Bài 1: Viết chương trình Sever giải đáp tên miền. Nếu máy khách gữi tên máy thì sever sẽ gởi về địa chỉ IP (danh sách này tự tạo ra – khoản 3 cập để minh hoạ).

Bài 2: viết chương trình UDP (Ứng dụng A) đặt trên một máy thực hiện các công việc sau, khi một ứng dụng B gữi 1 kiểu chuỗi tiếng Anh thì ứng dụng

A sẽ gửi trả lại nghĩa tiếng Việt tương ứng. Nếu từ này không có trong từ điển (chỉ có 3 từ Computer,Ram,HDD)thì ứng dụng A cho người dùng biết từ này không có trong từ điển.

Bài 3: Viết chương trình Client/Sever trong đó khi Client di chuyển chuột thì sever cũng di chuyển theo.

Bài 4: Viết Chương trình Client/Server khi Client gửi "Shutdown","Restast" thì Sever khởi động hoặc tắt máy tương ứng

Bài 5: Viết chương trình Chat giữa các máy tính ?

Bài 6 : Viết chương trình FTP Client?

Bài 7 : Viết chương trình truyền nhận File văn bản đơn giản (dạng text) giữa 2 máy tính trong mạng LAN dùng cơ chế đa tiểu trình (multi-threading)

Bài 8: Viết chương trình điều khiển máy tính dùng công nghệ Bluetooth.

Bài 9: Remote Shell: điều khiển máy tính từ xa dùng - command line

Bài 10 : Viết Chương trình Mail Client (SMTP).

Bài 11: Sử dụng giao thức HTTP để viết chương trình thi trắc nghiệm qua mạng LAN.

Bài 12: Viết chương trình hỗ trợ thi (như đấu trường 100) qua mạng LAN

Bài 13: Viết chương trình tính toán N! qua mạng (dùng cơ chế đa tiểu trình (multi-threading)

Bài 14: Viết chương trình Telnet (có thể thay đổi tên máy,cổng…)