

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

LÊ TUẤN

Unix

Hệ điều hành

& MỘT SỐ VẤN ĐỀ

QUẢN TRỊ MẠNG

NHÀ XUẤT BẢN KHOA HỌC VÀ KỸ THUẬT



LÊ TUẤN

UNIX - Hệ điều hành và một số vấn đề quản trị mạng



**NHÀ XUẤT BẢN KHOA HỌC VÀ KỸ THUẬT
HÀ NỘI**

Lời giới thiệu

UNIX là hệ điều hành được sử dụng rất hiệu quả trong các môi trường ứng dụng phân tán, nhiều người dùng. Ngoài ra, do tính mở của nó nên UNIX ngay từ khi mới ra đời đã được tiếp nhận nồng nhiệt và được đưa vào giảng dạy, nghiên cứu rộng rãi trong các trường đại học ở Mỹ và các nước phát triển khác. Hiện nay, UNIX cũng đã bắt đầu được ứng dụng nhiều ở Việt Nam, và do vậy, nhiều trường đại học đã đưa UNIX vào chương trình giảng dạy chính thức cho sinh viên các chuyên ngành *Tin học, Công nghệ thông tin, Điện tử – Viễn thông, Tự động hóa*, v.v..

Tuy nhiên, có thể nói cho đến nay ở Việt Nam chưa có một tài liệu tham khảo hoặc giáo trình (*tiếng Việt*) nào về UNIX vừa có tính hệ thống, vừa có tính cập nhật, đáp ứng được cho nhiều đối tượng người đọc khác nhau : từ người dùng cuối (end-user) đến các sinh viên, giáo viên chuyên ngành. Trong bối cảnh đó, cuốn sách “**UNIX – Hệ điều hành và một số vấn đề quản trị mạng**” của tác giả **Lê Tuấn** là một công trình khoa học nghiêm túc đáng ghi nhận. Đây là kết quả của nhiều năm đi sâu nghiên cứu và thực hành về UNIX, là sự tập hợp có hệ thống các bài giảng mà tác giả - *Tiến sỹ Lê Tuấn* - đã thực hiện cho nhiều đối tượng khác nhau. Cuốn sách vì vậy vừa mang tính sư phạm vừa có tính hiện đại, bám sát tình hình phát triển hết sức nhanh chóng của *Công nghệ thông tin* nói chung và của lĩnh vực các *hệ điều hành máy tính và mạng* nói riêng.

Trong bối cảnh Công nghệ thông tin ngày càng đi vào xu hướng chuyên nghiệp hóa và mạng Internet trở thành phương tiện truyền thông mang tính xã hội cao, việc đào tạo và bồi dưỡng các kiến thức cơ bản về hệ điều hành UNIX là hết sức cấp thiết. Vì thế, cuốn sách của tác giả Lê Tuấn được xuất bản sẽ là một đóng góp khoa học hết sức kịp thời và có giá trị thực tiễn cao.

Xin trân trọng giới thiệu cuốn sách với bạn đọc gần xa.

GS. TS. Nguyễn Thúc Hải

Lời nói đầu

Hệ điều hành UNIX là hệ điều hành được gắn liền với môi trường mạng máy tính, và mang rất nhiều các ưu điểm nổi trội, luôn giành được sự tín nhiệm rất cao của người dùng chuyên nghiệp. Trải qua quãng thời gian hơn ba mươi năm - nếu lấy năm 1970 làm thời điểm bắt đầu - UNIX liên tục được cải tiến, hoàn thiện và phát triển, đã tự chứng tỏ là hệ điều hành đa nhiệm, nhiều người dùng ưu việt, rất ổn định, có tính mở và tính bảo mật cao và khả năng truyền thông tuyệt hảo. UNIX không công kênh như một số hệ điều hành khác, rất mạnh mẽ và linh hoạt, có thể được dùng từ việc chạy trên các máy tính cá nhân nhỏ bé cho đến điều khiển các mạng máy tính có quy mô toàn cầu, quản lý lượng tài nguyên khổng lồ.

Việc phổ cập và ứng dụng công nghệ thông tin vào thực tế sản xuất và đời sống ở Việt Nam ngày càng đóng một vai trò quan trọng. Đặc biệt, những năm gần đây, đối với các ngành kinh tế quan trọng và hiện đại như tài chính ngân hàng, hàng không, bưu điện, dầu khí..., có thể nói các hệ thống máy tính đang dần dần chiếm lĩnh vị trí then chốt trong sơ đồ công nghệ của các cơ quan đó. Đó là còn chưa kể đến hệ thống các viện nghiên cứu, các trường đại học đang có nhu cầu hết sức cấp thiết mở rộng các khả năng nghiên cứu, tính toán của mình. Do đó, rất cần phải xây dựng một đội ngũ những người làm tin học chuyên nghiệp và nhu cầu học tập, nghiên cứu và sử dụng UNIX đối với họ đã trở nên cấp thiết hơn bao giờ hết. Nhưng cho đến nay tại Việt Nam, các tài liệu về UNIX rất hiếm, tuyệt đại đa số được viết bằng tiếng Anh, có nguồn do đi kèm theo sản phẩm hay viện trợ, được lưu giữ tại các cơ sở có sử dụng UNIX, hoặc là tài liệu giảng dạy của các cơ sở đào tạo hợp tác với nước ngoài, không phải ai cũng có thể dễ dàng tham khảo được. Một số rất ít các tài liệu chuyên đề được các cá nhân tích cực soạn thảo cũng chỉ có phạm vi sử dụng rất nhỏ hẹp, không mang tính hệ thống, hoặc chỉ giới hạn trong các cơ sở đào tạo chuyên sâu.

Cuốn sách này được viết nhằm phục vụ cho nhiều đối tượng người đọc, với mục đích giúp người đọc tự đào tạo để trở thành người sử dụng UNIX chuyên nghiệp. Các kiến thức cơ sở tối thiểu đối với người dùng bình thường - “người dùng cuối” (dịch từ thuật ngữ tiếng Anh *end user*) được trình bày trong các chương đầu về hệ điều hành UNIX nói chung và các tiết đầu của các chương dành cho các vấn

để cụ thể. Khối lượng kiến thức này, kể cả phần thực hành, có thể được soạn tương đương với hai đơn vị học trình hiện nay của Trường Đại học Bách khoa Hà Nội cho sinh viên không thuộc chuyên ngành Công nghệ thông tin. Theo tác giả, khối lượng kiến thức ban đầu này tạm đủ để người đọc có thể làm việc với máy tính trong mạng có sử dụng UNIX. Các sinh viên chuyên ngành, các cán bộ kỹ thuật chuyên nghiệp, v.v., có thể tìm thấy thông tin cần thiết về một số vấn đề quản trị mạng UNIX, gồm các nội dung chuyên sâu hơn về UNIX và mạng máy tính, các trục trặc có thể hay gặp phải trong thực tế, các kinh nghiệm và kỹ năng để điều hành một mạng UNIX, được trình bày trong các chương tương ứng theo từng chủ đề. Tác giả cũng có tham vọng xây dựng phần này có tính chất cẩm nang, có ích để tham khảo trong vận hành thực tế một mạng UNIX. UNIX có nhiều bản, do các hãng phần mềm khác nhau phát hành. Tác giả cố gắng phản ánh các kiến thức chung trên cơ sở System V - tập lệnh cơ bản để các bản UNIX hiện đại liên thông với nhau. Các bản UNIX chủ yếu mà tác giả dùng để minh họa là Solaris của hãng Sun, được dùng phổ biến trong các tổ chức, doanh nghiệp Nhà nước, và SuSE Linux (hãng SuSE), một bản UNIX miễn phí phổ biến trong giới tin học Việt Nam. Các bản UNIX khác cũng được dùng để minh họa và có lưu ý người đọc trong từng trường hợp cụ thể. Một phần nội dung của cuốn sách này đã được dùng làm tài liệu giảng dạy cho sinh viên chuyên ngành của Trường Đại học Bách khoa Hà Nội, cho các khóa đào tạo dành cho các cán bộ điện toán của Ngân hàng Công thương Việt Nam liên tục từ năm 1995 đến nay, cho Ngân hàng Nông nghiệp và Phát triển nông thôn Việt Nam, cho Bưu điện, v.v..

Đối tượng sử dụng là các cán bộ tin học, các sinh viên và tất cả những người yêu thích UNIX, đã có các kiến thức cơ bản về tin học, và sẽ thuận lợi hơn, nếu họ đã được trang bị bằng các kỹ năng lập trình trên ngôn ngữ C/C++.

Trong khi học tập, nghiên cứu về UNIX, vai trò của thực hành là đặc biệt quan trọng. Các ví dụ trong cuốn sách này đã từng được người học thực hiện nhiều lần. Một số câu hỏi và bài tập được đưa vào chương cuối. Tuy nhiên, do hạn chế về thời gian nên tác giả chưa tập hợp riêng được các câu hỏi và bài tập theo nội dung từng chương để nâng cao hơn nữa tính sư phạm của tài liệu này.

Tác giả xin chân thành cảm ơn Ngân hàng Công thương Việt Nam (trước hết là Trung tâm Công nghệ thông tin của ngân hàng), Công ty điện tử tin học Phi Bình (FIBI Co.), Thạc sỹ Tin học Nguyễn Anh Dũng (Trường Đại học Bách khoa thành

phố Hồ Chí Minh), kỹ sư Lý Hiền Trung, cùng bạn bè gần xa đã tạo điều kiện, nhiệt tình giúp đỡ, cổ vũ để tác giả hoàn thành cuốn sách này.

Tác giả đặc biệt cảm ơn Giáo sư – Tiến sỹ Nguyễn Thúc Hải, Trưởng khoa Công nghệ thông tin Trường Đại học Bách khoa Hà Nội đã rất nhiệt tình đọc bản thảo, cho nhiều góp ý hết sức quý báu để hoàn thiện cuốn sách.

Tuy đã rất cố gắng, nhưng do nội dung kiến thức về UNIX quá rộng, không thể gói gọn chỉ trong một cuốn sách, và do trình độ bản thân tác giả và khả năng có hạn, trong lần đầu ra mắt này, chắc chắn cuốn sách còn có rất nhiều thiếu sót. Hy vọng rằng lần ra mắt tiếp theo của cuốn sách này sẽ xuất hiện trong tương lai gần, với các sửa chữa, bổ sung đáng kể và thông tin mới cập nhật.

Tác giả rất biết ơn và mong nhận được các ý kiến phê bình, xây dựng và hợp tác của bạn đọc. Mọi thư từ xin gửi về: TS Lê Tuấn, Viện Vật lý kỹ thuật, Trường Đại học Bách khoa Hà Nội, số 1, Đại Cồ Việt, Hà Nội, Việt Nam. Tel.: 8693350, E-mail: letuan_hn@hn.vnn.vn .

Tác giả

MỤC LỤC

I. Tổng quan về hệ điều hành UNIX	19
<i>I. 1. Các hệ điều hành UNIX và phiên bản mới nhất.....</i>	<i>19</i>
<i>I. 2. Các đặc điểm cơ bản</i>	<i>21</i>
I. 2. a. Nhiều người sử dụng	21
I. 2. b. Đa nhiệm	22
I. 2. c. Tổ chức file phân cấp	22
I. 2. d. Độc lập phần cứng	23
I. 2. e. Dùng chung thiết bị	23
I. 2. f. Bảo mật	23
I. 2. g. Kết nối mở	24
<i>I. 3. Các cấu hình phần cứng đáp ứng hệ điều hành UNIX.....</i>	<i>24</i>
<i>I. 4. Các thành phần chính của hệ điều hành UNIX</i>	<i>25</i>
I. 4. a. Kernel	25
I. 4. b. Bộ điều khiển thiết bị	26
I. 4. c. Lệnh và tiện ích cơ bản.....	26
I. 4. d. Shell.....	29
I. 4. e. Giao diện đồ họa cho người dùng	30
<i>I. 5. Các điểm tương quan giữa MS-DOS và UNIX.....</i>	<i>30</i>
II. Đăng nhập hệ thống.....	34
<i>II. 1. Bắt đầu và kết thúc phiên làm việc</i>	<i>34</i>
II. 1. a. Đăng nhập hệ thống	34
II. 1. b. Thoát khỏi hệ thống	35
<i>II. 2. Giao tiếp với hệ thống</i>	<i>35</i>
<i>II. 3. Quy tắc đặt mật khẩu người dùng.....</i>	<i>36</i>
III. Hệ thống file của UNIX.....	38
<i>III. 1. Cấu trúc hệ thống file</i>	<i>38</i>
III. 1. a. Hệ thống thư mục	38
III. 1. b. Các thành phần khác.....	41
III. 1. c. Phân loại hệ thống file UNIX.....	42
<i>III. 2. File, thư mục, nguyên tắc đặt tên và đường dẫn.....</i>	<i>43</i>
III. 2. a. File.....	43
III. 2. b. Thư mục	46

III. 3. Chủ quyền truy nhập file	47
III. 4. Các lệnh thao tác căn bản.....	51
III. 4. a. Các lệnh thao tác trên thư mục.....	51
III. 4. b. Các lệnh thao tác với file	54
III. 4. c. Các lệnh xử lý dữ liệu trong các file	57
III. 5. Các lệnh bảo vệ file và thư mục.....	69
III. 5. a. Khái niệm.....	69
III. 5. b. Các lệnh	70
III. 6. Soạn thảo file sử dụng vi	73
IV. Xử lý vào/ra	78
IV. 1. Đối tượng dữ liệu.....	78
IV. 2. Đường ống và lọc.....	79
IV. 3. Đối hướng vào/ra chuẩn.....	79
IV. 3. a. Khái niệm về đối hướng.....	79
IV. 3. b. Đa đối hướng	80
IV. 4. Tổ hợp các file với vào chuẩn.....	80
IV. 5. Các dạng khác của đối hướng.....	81
IV. 5. a. Đối hướng thêm vào >>.....	81
IV. 5. b. Đối hướng "File" <<	81
IV. 5. c. Các bộ mô tả file và đối hướng sai chuẩn 2>	82
IV. 5. d. Tổ hợp ra 2>&1	82
IV. 6. Lệnh tee.....	83
V. Sử dụng các dịch vụ Internet	84
V. 1. Lệnh mail.....	84
V. 2. Lệnh mailx.....	85
V. 3. Các lệnh làm việc khác với máy tính từ xa	87
V. 3. a. Đăng nhập từ xa.....	87
V. 3. b. Trao đổi với người khác trên mạng.....	88
V. 3. c. Sao chép file với máy tính khác từ xa	89
V. 3. d. Thực hiện lệnh trên máy tính từ xa có kiểm tra kết quả	92
V. 4. Dò tìm thông tin từ Internet, Intranet	93
V. 4. a. Vị trí của các thông tin cần tìm kiếm.....	93
V. 4. b. Các công cụ tìm kiếm thông tin	94
V. 5. Khái niệm chung về sendmail	101
V. 6. Các thư mục và file của sendmail	102
V. 7. Cấu hình sendmail cho hệ thống Solaris	107

VI. Quản trị hệ thống.....	115
VI. 1. Giới thiệu hệ điều hành UNIX Solaris	115
VI. 2. Quyền đăng nhập superuser	115
VI. 2. a. Làm thế nào để trở thành superuser.....	116
VI. 2. b. Thực hiện lệnh với quyền root hạn chế.....	117
VI. 3. Các file quản lý hệ thống của UNIX	119
VI. 4. Các mức chạy.....	124
VI. 4. a. Phân loại mức chạy.....	124
VI. 4. b. Quy định mức chạy cho người dùng.....	126
VI. 5. Sử dụng các tiện ích kiểu Midnight Commander (mc).....	128
VI. 5. a. Tải về mã nguồn của Midnight Commander	128
VI. 5. b. Khái niệm cơ bản	129
VI. 5. c. Chế độ liệt kê.....	129
VI. 5. d. Các chế độ khác của mc	130
VI. 5. e. Phân loại và lọc.....	130
VI. 5. f. Tìm file.....	131
VI. 5. g. Hỗ trợ hệ thống file ảo.....	131
VI. 5. h. Các điểm thuận tiện khác.....	133
VI. 6. Nhiệm vụ của người quản trị hệ thống.....	134
VII. Môi trường Shell và lập trình Shell.....	135
VII. 1. Các khái niệm cơ bản.....	135
VII. 1. a. Các lệnh trong shell.....	135
VII. 1. b. Sự thực hiện lệnh	139
VII. 1. c. Tiến trình nền	139
VII. 1. d. Gợi lại lệnh đã dùng	140
VII. 2. Xử lý tiến trình	141
VII. 2. a. Lệnh gọi hệ thống fork	142
VII. 2. b. Lệnh gọi hệ thống exec	143
VII. 2. c. Tiến trình login.....	143
VII. 2. d. Thực hiện lệnh đơn giản.....	143
VII. 2. e. Thực hiện chuỗi lệnh.....	143
VII. 2. f. Thực hiện nhóm lệnh	143
VII. 2. g. Thực hiện lệnh đường ống	144
VII. 2. h. Thực hiện lệnh nền.....	144
VII. 3. Lệnh nội tại của shell.....	144
VII. 4. Biến và tham số shell.....	145

VII. 4. a. Tham số vị trí	145
VII. 4. b. Tham số đặc biệt.....	146
VII. 4. c. Tham số được đặt tên	146
VII. 5. Cấu trúc lập trình shell	147
VII. 5. a. Vòng lặp for	147
VII. 5. b. Điều kiện while	149
VII. 5. c. Điều kiện until.....	149
VII. 5. d. Kiểm tra điều kiện if.....	150
VII. 5. e. Lệnh tình huống case	151
VII. 5. f. Hàm	152
VII. 6. Shell script.....	153
VII. 7. Bổ sung về shell.....	155
VII. 7. a. So sánh giữa Borne, Korn và C shell	155
VII. 7. b. Một vài lời khuyên khi sử dụng shell	156
VII. 7. c. Các kỹ thuật shell nâng cao	157
VIII. Các ngôn ngữ lập trình, các trình soạn thảo.....	165
VIII. 1. Ngôn ngữ awk.....	165
VIII. 2. Dự án GNU.....	169
VIII. 3. Ngôn ngữ lập trình PERL.....	169
VIII. 3. a. Bắt đầu một PERL Script.....	170
VIII. 3. b. Các tùy chọn.....	171
VIII. 3. c. Các loại dữ liệu và đối tượng.....	174
VIII. 3. d. Các lệnh ghép.....	179
VIII. 3. e. Các lệnh đơn giản	182
VIII. 3. f. Biểu thức	182
VIII. 3. g. Thông tin hệ thống	204
VIII. 3. h. Quyền ưu tiên	205
VIII. 3. i. Các chương trình con	205
VIII. 4. Chương trình soạn thảo sed	206
VIII. 4. a. Chèn các dòng hoặc văn bản.....	209
VIII. 4. b. Xóa các dòng văn bản	210
VIII. 4. c. Thay thế.....	211
VIII. 4. d. Dịch ký hiệu trong văn bản	212
VIII. 4. e. In các dòng văn bản	213
VIII. 5. Emacs.....	214

VIII. 5. a. Tìm hiểu cấu trúc Emacs.....	215
VIII. 5. b. Học cách dùng Emacs.....	216
VIII. 5. c. Các chế độ.....	221
VIII. 5. d. Các chế độ phụ.....	223
VIII. 5. e. Khởi động Emacs trong UNIX.....	225
VIII. 5. f. Macro.....	226
VIII. 6. LaTeX.....	227
VIII. 6. a. Có thể tải TeX/LaTeX về từ đâu.....	229
VIII. 6. b. Các đối tượng trong LaTeX.....	229
VIII. 6. c. Các lệnh LaTeX.....	231
VIII. 6. d. Bắt đầu chế bản.....	238
VIII. 6. e. Các lệnh trong chế độ văn bản.....	240
VIII. 6. f. Các lệnh trong chế độ toán học.....	250
VIII. 6. g. Làm đơn giản bớt các thao tác chế bản.....	263
VIII. 6. h. Làm việc với các file lớn.....	265
VIII. 6. i. Kiểm tra và in.....	266
VIII. 6. j. Đưa các hình ảnh vào tài liệu.....	268
VIII. 6. k. Bộ đếm.....	272
IX. Sử dụng X-Windows.....	274
IX. 1. Khái niệm chung.....	274
IX. 2. Thiết lập X terminals trên các máy tính với CPU Intel.....	275
IX. 2. a. X terminal.....	276
IX. 2. b. Thiết lập terminal.....	277
IX. 2. c. Để có chất lượng hình ảnh tốt.....	278
IX. 2. d. Chuột vi tính trong chế độ đồ họa.....	278
IX. 2. e. Đăng nhập vào phiên làm việc X.....	279
IX. 2. f. Cài đặt cho x86.....	280
IX. 2. g. Cải thiện X terminal và chế độ đa khởi động.....	281
IX. 2. h. Kết luận cho phần này.....	283
IX. 3. Môi trường KDE.....	283
IX. 3. a. Các đặc điểm của KDE.....	283
IX. 3. b. Ứng dụng K và KDE 2.0.....	285
IX. 3. c. Ví dụ một phiên làm việc của KDE.....	288
IX. 3. d. Cài đặt KDE.....	290
IX. 3. e. Biên dịch bản tiền phát hành KDE 1.1.2.....	291

IX. 3. f. Tích hợp KDE vào dtlogin của Solaris	292
IX. 3. g. Các vấn đề về bản quyền và GNOME	297
IX. 3. h. Các lỗi có thể gặp khi dùng KDE trong Solaris.....	298
IX. 3. i. Các thông tin tham khảo.....	298
IX. 4. StarOffice	298
IX. 4. a. Linh hoạt như con tắc kè hoa (Chameleon) đổi màu.....	299
IX. 4. b. Các lý do nên dùng StarOffice.....	302
IX. 4. c. Cài đặt StarOffice.....	302
X. Lưu trữ và khôi phục dữ liệu.....	306
<i>X. 1. Khái niệm và chiến lược lưu trữ dữ liệu.....</i>	<i>306</i>
<i>X. 2. Các lệnh UNIX thường dùng ở chế độ dòng lệnh.....</i>	<i>308</i>
X. 2. a. Đối với hệ điều hành Solaris	308
X. 2. b. Các lệnh và tiện ích trong các hệ điều hành UNIX khác	317
X. 2. c. Các file đặc biệt thiết bị băng từ	324
<i>X. 3. Ví dụ thực hiện các lệnh lưu trữ của Solaris</i>	<i>326</i>
<i>X. 4. Quản trị tự động lưu trữ - khôi phục dữ liệu bằng Solstice Backup</i>	<i>327</i>
<i>X. 5. Tiện ích băng từ MakeIndex.....</i>	<i>335</i>
XI. In và quản lý in	337
<i>XI. 1. Các file và tập lệnh in</i>	<i>337</i>
<i>XI. 2. Thêm máy in mới vào hệ thống</i>	<i>341</i>
XI. 2. a. Trình tự thiết lập và loại bỏ máy in tại chỗ	341
XI. 2. b. Thiết lập máy in từ xa tới máy chủ tại chỗ.....	344
XI. 2. c. Thiết lập máy in từ xa tới máy chủ từ xa (tạo các client).....	345
XII. Bảo mật.....	347
<i>XII. 1. Khái niệm về các cấp bảo mật.....</i>	<i>347</i>
<i>XII. 2. Nâng cấp bảo mật hệ thống tới C2.....</i>	<i>348</i>
XII. 2. a. Các file cần thiết cho C2	348
XII. 2. b. Chuyển đổi sang C2	349
<i>XII. 3. Quản lý tài khoản người dùng</i>	<i>350</i>
XII. 3. a. Thêm bớt các tài khoản và nhóm người dùng.....	350
XII. 3. b. Đặc điểm quản lý tài khoản người dùng trong hệ Solaris.....	353
XII. 3. c. Chống đăng nhập trái phép đến hệ thống và mạng Solaris....	356
XII. 3. d. Các bước tăng cường bảo mật máy chủ Solaris	361
XII. 4. FireWall	366
XII. 4. a. Khái niệm chung	366
XII. 4. b. Quy trình cài đặt FireWall	367

XII. 4. c. Ví dụ cài đặt và cấu hình FireWall trên hệ Solaris 2.6.....	373
XIII. Bảo trì phần cứng hệ thống.....	385
XIII. 1. Tổ chức bộ nhớ.....	385
XIII. 1. a. Thay đổi kích thước swap	385
XIII. 1. b. Thiết lập hệ thống file cache.....	387
XIII. 1. c. Tìm hiểu hệ thống file /proc	388
XIII. 2. Dùng đĩa mềm	391
XIII. 2. a. Dùng các file thiết bị đặc biệt cho đĩa mềm	391
XIII. 2. b. Dùng đĩa mềm MS-DOS trong hệ UNIX.....	392
XIII. 2. c. Dùng như thiết bị ký tự.....	394
XIII. 3. Thêm đĩa cứng	395
XIII. 3. a. Giao diện SCSI và RAID.....	395
XIII. 3. b. Thêm đĩa cứng mới.....	397
XIII. 4. Thêm ổ CD	402
XIII. 5. Thêm thiết bị cổng nối tiếp (modem)	403
XIII. 5. a. Cấu trúc của SAF	404
XIII. 5. b. Các lệnh cấu hình SAF và các bộ giám sát cổng.....	405
XIII. 6. Cách khắc phục một số lỗi phần cứng thường gặp.....	408
XIII. 6. a. Khi nào máy gặp sự cố.....	408
XIII. 6. b. Khi bị lỗi "System file full"	409
XIII. 6. c. Đối với các lỗi của đĩa cứng.....	409
XIII. 6. d. Đối với các thiết bị vào / ra (bàn phím, màn hình, chuột)	411
XIII. 6. e. Các lỗi với thiết bị đầu cuối	412
XIII. 6. f. Các lỗi với card mạng và mạng Ethernet.....	413
XIV. Khái niệm về mạng và quản trị mạng.....	414
XIV. 1. Nhiệm vụ của người quản trị mạng.....	414
XIV. 2. Khái niệm về mô hình mạng.....	414
XIV. 3. Khái niệm về cấu trúc mạng.....	416
XIV. 3. a. LAN.....	416
XIV. 3. b. WAN.....	421
XIV. 4. Phần cứng giao tiếp mạng.....	425
XIV. 4. a. Các loại cáp mạng và đầu nối.....	425
XIV. 4. b. Repeater	429
XIV. 4. c. Bridge	429
XIV. 4. d. Router.....	430
XIV. 4. e. Gateway (Application server hay Relay hay Forwarder)	431

XIV. 4. f. Terminal server	431
XV. Các giao thức và dịch vụ mạng	432
XV. 1. Giao thức TCP/IP	432
XV. 1. a. Giới thiệu.....	432
XV. 1. b. Các thành phần của TCP/IP.....	432
XV. 1. c. Các khái niệm cơ bản.....	433
XV. 1. d. Các ứng dụng phổ biến của TCP/IP	435
XV. 1. e. Các lệnh cơ bản trên (DOS, UNIX)	436
XV. 1. f. Mặt nạ mạng con và mặt nạ liên mạng.....	439
XV. 1. g. Anonymous ftp và Mirroring	442
XV. 1. h. Thiết lập máy UNIX để sử dụng Usenet.....	454
XV. 1. i. IPv6.....	472
XV. 2. uucp.....	489
XV. 2. a. Phần cứng của uucp	489
XV. 2. b. Các chương trình trong uucp.....	490
XV. 2. c. Khởi động uucp	494
XV. 2. d. Các file dữ liệu của uucp	494
XV. 2. e. Dùng TCP/IP trong uucp	497
XV. 3. Dịch vụ ppp	497
XV. 3. a. Các gói phần mềm cài đặt.....	498
XV. 3. b. Các thành phần chính của ppp	502
XV. 3. c. Tạo người dùng bình thường để khởi động và kết thúc daemon ppp	504
XV. 3. d. Thiết lập các máy client ppp để quay số ra ngoài	506
XV. 3. e. Thiết lập các máy client của ppp có cài tiến để quay số ra ngoài.....	509
XV. 3. f. Thiết lập máy chủ ppp trong mạng LAN, với các địa chỉ IP tĩnh.....	510
XV. 3. g. Thiết lập máy chủ ppp quay số vào với các địa chỉ IP động.....	513
XV. 3. h. Thiết lập nối kết ppp Mạng-tới-Mạng	516
XV. 3. i. Thiết lập client ppp quay số ra ngoài, với địa chỉ IP động	520
XV. 3. j. Dùng ppp quay số vào và quay số ra trên cùng một hệ thống	521
XV. 3. k. Cấu hình Windows 9x để quay số tới hệ thống Solaris	523
XV. 4. SLIP.....	524
XV. 4. a. Khái niệm về SLIP	524
XV. 4. b. So sánh giữa SLIP với ppp	525
XV. 4. c. Chạy SLIP trên hệ thống Linux	528
XV. 5. NFS	531

XV. 5. a. Dịch vụ NFS hoạt động như thế nào.....	531
XV. 5. b. Cài đặt và bảo trì NFS Server.....	532
XV. 5. c. Cài đặt và bảo trì NFS Client	533
XV. 6. Phân tích mạng.....	534
XV. 6. a. Các lệnh có ích.....	535
XV. 6. b. Kiểm tra phần mềm	542
XV. 7. Dịch vụ thông tin mạng (NIS và NIS +).....	543
XV. 7. a. Khái niệm	543
XV. 7. b. Các thành phần của NIS+	543
XV. 7. c. Các bước cài đặt NIS.....	545
XV. 8. Dịch vụ tên vùng (DNS).....	548
XV. 8. a. Các bản ghi nguồn	549
XV. 8. b. Các file cấu hình DNS	552
XV. 8. c. Thiết lập DNS.....	554
XV. 9. Server Apache Web và server proxy.....	554
XV. 9. a. Chạy server Web với người dùng bình thường	554
XV. 9. b. Cài đặt Server	555
XV. 9. c. Đặt quyền cho file và thư mục	556
XV. 9. d. Xóa các script CGI và các tài liệu HTML không cần thiết	560
XV. 9. e. Cấu hình Server.....	561
XV. 9. f. Cấu hình việc kiểm toán server.....	566
XV. 9. g. Cấu hình giám sát đăng nhập, kiểm tra xác nhận người dùng	566
XV. 9. h. Các thao tác từ xa có hạn chế.....	567
XV. 9. i. Tạo bảng chữ bảo mật.....	567
XV. 9. j. Lưu trữ thường kỳ và đặt lại log	568
XV. 9. k. Khởi động và kết thúc Server	569
XV. 9. l. Cấu hình proxy	570
XVI. Server PC Samba.....	577
XVI. 1. Tìm hiểu về Samba.....	577
XVI. 1. a. Tại sao lại là Samba	577
XVI. 1. b. Các thành phần của Samba.....	579
XVI. 2. Mạng SMB/CIFS.....	582
XVI. 2. a. NetBIOS và NetBUI.....	583
XVI. 2. b. Dịch vụ tên Internet của Windows (WINS).....	592
XVI. 2. c. Giao thức SMB	593

XVI. 3. Cài đặt Samba trên hệ thống UNIX	602
XVI. 3. a. Tạo file cấu hình cài đặt makefile của Samba	602
XVI. 3. b. Biên dịch và cài đặt Samba	609
XVI. 3. c. File cấu hình cơ bản của Samba	613
XVI. 3. d. Khởi động và kiểm tra các daemon Samba	619
XVI. 4. Cấu hình các client Windows	623
XVI. 4. a. Thiết lập các client Windows 9x	623
XVI. 4. b. Thiết lập các máy Windows NT	636
XVI. 5. Chia sẻ đĩa cứng	647
XVI. 5. a. Nghiên cứu về file cấu hình Samba	647
XVI. 5. b. Các phần đặc biệt của file cấu hình Samba	655
XVI. 5. c. Các tùy chọn của file cấu hình	658
XVI. 5. d. Cấu hình server Samba	661
XVI. 5. e. Cấu hình chia sẻ đĩa cứng	665
XVI. 5. f. Các tùy chọn về mạng của Samba	671
XVI. 5. g. Các server ảo	677
XVI. 6. In trong Samba	679
XVI. 6. a. Gửi công việc in tới Samba	679
XVI. 6. b. In trên các máy in của client Windows	694
XVII. Bài tập	708
XVII. 1. Các bài tập dành cho người dùng cuối	708
XVII. 1. a. Bài tập UNIX-1	708
XVII. 1. b. Bài tập UNIX-2	710
XVII. 1. c. Bài tập UNIX-3	712
XVII. 1. d. Bài tập UNIX-4	714
XVII. 2. Bài tập nâng cao	715
XVIII. Tài liệu tham khảo	718

I. Tổng quan về hệ điều hành UNIX

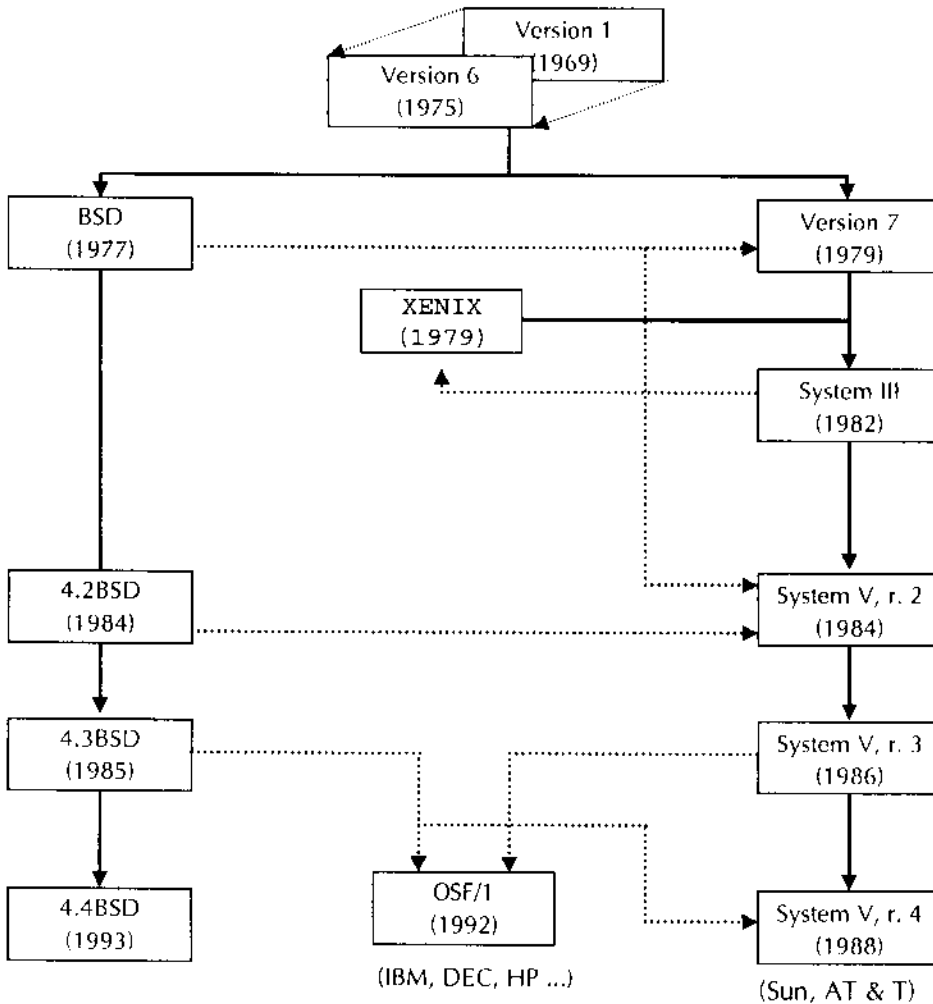
I.1. Các hệ điều hành UNIX và phiên bản mới nhất

Hầu hết các hãng máy tính đều phát triển một phiên bản UNIX riêng của mình. Dưới đây liệt kê các sản phẩm UNIX của các hãng, được sắp xếp một cách ngẫu nhiên và với phiên bản mới nhất:

- Microsoft: Xenix (không được nhiệt tình phát triển như DOS), v.2.3.4 và sau cũng được bán như SCO.
- Santa Cruz Operation: SCO UNIX, v. 5.04, là bản UNIX rất phổ biến trên thế giới và hỗ trợ nhiều phần cứng.
- Linux, là hệ điều hành miễn phí giống UNIX, được bắt đầu viết bởi Linus Torvalds nhằm hỗ trợ cho việc học UNIX, mã nguồn được cung cấp tự do và sau đó được phát triển bởi nhiều người. Chỉ cần chi phí cho môi trường chứa mã (đĩa mềm, CD,...) và phí bưu kiện. Các nhà phát hành chỉ được thu lợi từ việc bán các giá trị gia tăng và tài liệu. Hiện nay, Linux được phát hành bởi nhiều hãng với kernel mới nhất : 2.4.20. Ví dụ, tính đến đầu năm 2003, đã xuất hiện trên thị trường Việt Nam SuSE Linux 8.2, RedHat 9.0, Mandrack 8.1, v.v.. Các bản Linux được đánh giá tốt trong môi trường doanh nghiệp là SuSE Linux, Mandrack và cả RedHat.
- BSD: FreeBSD, của Trường Đại học Berkeley, dựa trên BSD3.2. Giống như Linux, là sản phẩm miễn phí dùng cho học tập và nghiên cứu phát triển.
- Novell Inc.: UNIXWare, v.2.0
- IBM: AIX (Advanced Interactive Executive), v. 5.
- Sun Microsystems: Solaris, chia làm hai loại chạy trên SPARC và cho x86. Các phiên bản hiện nay 2.7 (SPARC) và 7 (x86) và Solaris 8 được phát hành khoảng giữa năm 2000. Bản Solaris 9 được đưa ra thị trường vào khoảng đầu năm 2002, chỉ chạy cho các máy SPARC...
- Hewlett-Packard: HP-UX, v.9.03

- DEC: Ultrix
- NeXT Inc.: NeXTstep, v. 3.3., chạy trên máy Macintosh.
- ...

AT & T Bell Labs.

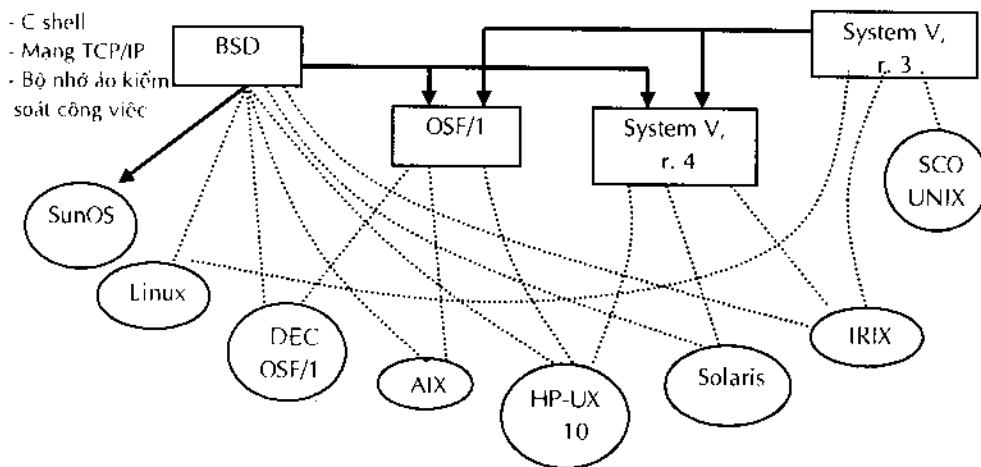


H. I. 1. Lịch sử phát triển và phân nhánh UNIX

Ngày nay, đã có hàng trăm ngàn hệ thống UNIX được cài đặt và sử dụng hiệu quả trên khắp thế giới. UNIX là hệ điều hành mạng chạy rất ổn định, bảo mật tốt, tin cậy. Nhưng người dùng gặp rất nhiều rắc rối do các bản UNIX của các hãng thường khác nhau. Hiện nay, để chuẩn hóa hệ điều hành UNIX, người ta quy ước tập lệnh chuẩn và gọi là UNIX System V. Trên máy PC phổ biến nhất là hai hệ điều hành SCO UNIX và Sun Solaris (cho x86), nếu không kể đến các bản Linux.

Sơ đồ trên H. 1. 1 cho thấy các mốc lịch sử phát triển, phân nhánh và quan hệ qua lại giữa các bản UNIX.

Ta có thể thấy ảnh hưởng qua lại của các bản UNIX trên H. 1. 2. Nên nhớ rằng một số tính năng ban đầu chỉ đặc trưng cho nhánh BSD (ví dụ, C shell), giờ đây đã có mặt trong tất cả các bản UNIX hiện đại.



H. 1. 2. Quan hệ giữa các bản UNIX

1. 2. Các đặc điểm cơ bản

Hệ điều hành UNIX có một số đặc điểm nổi bật sau:

1. 2. a. Nhiều người sử dụng

Nhiều người tại một thời điểm có thể sử dụng máy tính có cài UNIX. Ví dụ, trong server UNIX:

- User A: dùng Oracle, kết nối và cập nhật cơ sở dữ liệu
- User B: chương trình biên dịch
- User C: gửi thư điện tử

Hệ điều hành UNIX quản lý những người dùng theo cấu trúc phân cấp, người dùng có thể giao tiếp với nhau theo các nhóm. Mỗi người dùng có một mức đăng nhập hệ thống được quy định sẵn. Khi họ đồng thời cùng chạy một ứng dụng thì chỉ một bản sao chương trình được đưa vào bộ nhớ. Người dùng cao nhất (*superuser* hay *root*) có thể can thiệp đến các người dùng khác, thêm bớt danh sách người dùng và quyền của họ, nếu cần.

1. 2. b. Đa nhiệm

Tại một thời điểm một người dùng có thể thực hiện đồng thời nhiều tác vụ. Với hệ điều hành đơn nhiệm như MS-DOS, một lệnh thực hiện sẽ chiếm toàn bộ thời gian xử lý của CPU; bạn chỉ có thể thực hiện lệnh kế tiếp khi lệnh trước đó đã được thực hiện xong. Còn trong hệ điều hành UNIX bạn có thể đặt lệnh chạy ở chế độ nền, đồng thời khi đó có thể thực hiện các lệnh kế tiếp.

Tính đa nhiệm được cài trong nhân của UNIX. Một vùng trên đĩa được dành thực hiện thủ tục *swap*, cho phép tạm loại bỏ các tiến trình có mức độ ưu tiên thấp khỏi bộ nhớ, rồi lại đưa chúng vào để thực hiện sau khi đã chạy các tiến trình có ưu tiên cao hơn. Do thời gian làm việc của CPU dựa trên các xung nhịp, xét về thực chất, các tiến trình được thực hiện trên bề mặt, hay có mức ưu tiên cao hơn, là các tiến trình được phân chia nhiều thời gian CPU hơn.

Các tiến trình trong UNIX được phân cấp. Tiến trình gốc *init* chạy tự động khi khởi động hệ điều hành và có trách nhiệm tạo ra tất cả các tiến trình được thêm vào sau đó. Các tiến trình tuân theo quan hệ cha-con, và chúng còn có thể liên hệ với nhau bằng đường ống. Tiến trình con có thể thừa kế các biến môi trường và các file đã mở của tiến trình cha.

1. 2. c. Tổ chức file phân cấp

Các file của UNIX được tổ chức theo dạng cây có chung thư mục gốc được biểu diễn bởi ký tự *"/*". Bên trong một thư mục có thể là các thư mục con hay các file. Có 3 loại file của UNIX hay được nói tới nhiều nhất:

- File thường (*ordinary file*): là một file chứa các dữ liệu ASCII hay nhị phân.
- File thư mục (*directory file*): chứa danh sách các phần tử (thư mục, file, thiết bị) có thể truy xuất tới.
- File đặc biệt (*special file*): là các file liên quan đến các thiết bị phần cứng và truyền thông. Ví dụ:
 - bàn phím là một file nhập.
 - màn hình là một file xuất.

Đối với UNIX toàn bộ hệ thống file chỉ có một thư mục gốc (root). Có thể sử dụng lệnh **mount** để kết nối các ổ đĩa trong một hệ thống file duy nhất. Ví dụ:

Đĩa 1: /	lib	Đĩa 2: /	etc	Đĩa mount => /	lib
	bin		oracle		bin
	usr				usr
					etc
					oracle

I. 2. d. Độc lập phần cứng

Vì hệ điều hành UNIX được viết bằng ngôn ngữ cấp cao cho nên nó rất dễ cài đặt trên các cấu hình phần cứng khác nhau. Hơn nữa, với cách tổ chức các thiết bị là các file đặc biệt nên việc thêm vào hay loại bỏ các thiết bị rất nhẹ nhàng. Hiện nay, đặc điểm này hầu như chỉ còn mang tính lịch sử, vì các hệ điều hành khác như Windows, v.v.. cũng được viết trên ngôn ngữ bậc cao. Tuy nhiên, ưu điểm về xử lý các thiết bị như các file đặc biệt vẫn giữ nguyên giá trị của nó.

I. 2. e. Dùng chung thiết bị

Vì UNIX là môi trường nhiều người dùng, do đó các thiết bị ngoại vi như ổ đĩa mềm, máy in, modem, v.v.. có thể được sử dụng chung bởi nhiều người.

I. 2. f. Bảo mật

UNIX cung cấp rất nhiều cơ chế bảo mật khác nhau. Trong đó mỗi người dùng có một số quyền trên các file nhất định và chỉ được phép chạy một số chương trình nhất định. Ngoài ra cơ chế mã hóa và giải mã cũng là một phần hữu cơ của hệ điều hành.

I. 2. g. Kết nối mở

UNIX cung cấp các thiết bị mạng qua Ethernet, modem, X25, ... Với nhiều thủ tục truyền thông khác nhau UUCP (UNIX-to-UNIX Copy), TCP/IP, và các ứng dụng E-mail, FTP (File Transfer Protocol), NFS (Network File System), UNIX sẵn sàng tạo các kết nối mới, tự mở rộng hệ thống trên các thiết bị phần cứng khác nhau.

I. 3. Các cấu hình phần cứng đáp ứng hệ điều hành UNIX

Hệ điều hành UNIX được cài đặt trên các hệ thống máy tính có cấu hình rất khác nhau, từ PC (máy tính cá nhân), laptop (máy tính xách tay) cho tới các mainframe, các trung tâm máy tính lớn. Ví dụ, nếu xét về phương diện CPU của máy được cài đặt, hệ điều hành Solaris, từ bản v. 2.4 trở đi, có thể chạy với các loại họ Motorola 68000, SPARC, cũng như Intel và tương đương (x86). Ngay với họ Intel, máy có một CPU 386/33 MHz trở lên đã hỗ trợ Solaris .

Mặt khác, hệ điều hành UNIX trước hết là hệ điều hành mạng. Vì vậy, tồn tại một quan điểm chung đối với người quản trị mạng là chia các cấu hình phần cứng thành năm loại sau:

❖ File server

Mọi máy chủ chia sẻ dung lượng đĩa và các file - kể cả các file mà máy trạm có thể đăng nhập vào và chạy- đều được gọi là file server. Các file server được chia thành hai loại: *đồng nhất* - khi máy chủ hỗ trợ các máy trạm có cùng loại nhân hệ điều hành, *không đồng nhất* - trong trường hợp ngược lại. Các hệ điều hành đảm bảo hoạt động bình thường trên các máy chủ không đồng nhất là hệ điều hành "mở".

Hệ điều hành SunOS (Solaris) là một thí dụ về hệ điều hành mở. Các máy chủ Solaris là các Network File System (NFS) file server, và còn có khả năng cung cấp thêm dịch vụ Remote File Sharing (RFS) - chia sẻ file từ xa.

❖ Diskless client

Đó là máy tính không có đĩa cứng riêng của mình. Để hoạt động bình thường, máy tính này cần được kết nối với một mạng LAN nào đó với ít nhất một NFS file server. Diskless client dựa vào máy chủ để khởi động, hướng các dịch vụ mạng và chạy chương trình, cũng như sử dụng dung lượng đĩa cứng của máy chủ.

❖ Dataless client

Là máy tính có đĩa cứng của mình, trong đó bạn cấu hình các partition boot và swap tại chỗ. Dataless client lấy các thành phần khác từ NFS file server. Chính vì chỉ lấy /usr thiếu các file thực thi mà máy không thể hoàn toàn tự khởi động được và có tên gọi trên. Giống như diskless client, dataless client là máy chủ yếu dành cho một người dùng.

❖ Standalone system

Máy tính có cài đặt hệ điều hành UNIX trên đĩa của mình, hệ thống file có ít nhất các partition / , /swap và /usr, do đó tự khởi động và vận hành được, không cần tới sự trợ giúp của NFS file server. Bạn có thể dùng standalone system như một máy trạm hoạt động hoàn toàn độc lập với một người dùng, kết nối vào mạng, hay như một hệ thống chia sẻ thời gian (*time-sharing system*) cho vài người dùng. Nếu kết nối vào mạng, bạn có thêm các dịch vụ do máy chủ cung cấp, các thư mục, file bổ sung, thư điện tử, và như là:

❖ Time-sharing system

Hệ thống chia sẻ thời gian là một standalone system có các thiết bị đầu cuối nối vào các cổng nối tiếp của mình, hoặc hỗ trợ các thiết bị được nối qua đường điện thoại bằng modem. Thiết bị đầu cuối không có CPU của riêng mình, mà phải dựa vào các ổ đĩa cứng, đĩa mềm, file, hỗ trợ máy in, thời gian CPU của máy này. Hệ thống chia sẻ thời gian có thể là thành phần trong mạng hay không, nhưng ở các hệ thống chia sẻ thời gian lớn còn có thể kết nối trực tiếp với vài máy trạm.

Network domain là một cấu hình lớn, bao gồm thậm chí tất cả các loại đã kể trên. *Người quản trị mạng* trước hết phải là người quản trị hệ thống với các kỹ năng kỹ thuật cần thiết.

I. 4. Các thành phần chính của hệ điều hành UNIX

I. 4. a. *Kernel*

Là thành phần chủ yếu hay trái tim của hệ điều hành, chứa các dịch vụ tối thiểu đảm bảo hoạt động của hệ thống UNIX. Nó chiếm khoảng 500KB --> 2MB tùy theo tính chất phức tạp của hệ thống. Nhiệm vụ chính của kernel là:

- Quản lý thời gian: gọi thời gian hệ thống, xác định thời gian CPU, ngắt tiến trình nếu thời gian chờ tiến trình đó thực hiện xong là quá lâu.
- Quản lý tài nguyên: như bộ nhớ, đĩa cứng, v.v...
- Quản lý hệ thống file: có thể là các file, thư mục cục bộ hay từ xa.
- Quản lý các tiến trình thường trú (daemon).
- Quản lý bộ nhớ ảo: để thực thi nhiều tiến trình đồng thời trong khi số lượng bộ nhớ có hạn, UNIX tổ chức một vùng trên đĩa như là một vùng bộ nhớ (bộ nhớ ảo). Kernel phải “swap” các tiến trình giữa bộ nhớ và bộ nhớ ảo
- Quản lý tiến trình: như đã biết vì UNIX là một hệ điều hành đa nhiệm, do đó việc quản lý các tiến trình đồng thời rất phức tạp. Hệ điều hành phải quản lý việc khởi tạo và kết thúc các tiến trình cũng như các tranh chấp có thể xảy ra.
- Quản lý các bộ điều khiển thiết bị.
- Quản lý mạng: bao gồm nhiều thiết bị phần cứng khác và các thủ tục khác.
- Quản lý việc khởi động và dừng máy.

1. 4. b. Bộ điều khiển thiết bị

UNIX thể hiện các thiết bị vật lý như các file đặc biệt. Một file đặc biệt sẽ có một điểm vào trong thư mục và có một tên file. Do đó UNIX cho phép người dùng định nghĩa tên thiết bị.

Các thiết bị được chia làm hai loại: ký tự (*raw*) và khối (*block*).

- Thiết bị ký tự đọc và ghi dòng các ký tự một cách tuần tự (ví dụ, các thiết bị đầu cuối: modem, máy in).
- Thiết bị khối đọc và ghi dữ liệu trong các khối có kích thước cố định (ví dụ, ổ đĩa, băng từ, theo các khối 512 byte, 8 hay 16 KB).

Thiết bị có thể được đổi tên như đổi tên file. Thư mục chứa các bộ điều khiển thiết bị là `/dev`.

1. 4. c. Lệnh và tiện ích cơ bản

Các lệnh và tiện ích của UNIX rất đa dạng. Một lệnh UNIX có dạng:

```
$ command [option] [argument]
```

lệnh thường là chữ nhỏ. UNIX phân biệt các chữ lớn với các chữ nhỏ. Ví dụ:

```
$ ls -c /dev
```

Ta có thể chia lệnh thành các nhóm sau:

- *Các lệnh khởi tạo*

exit	Thoát khỏi hệ thống (Bourne-Shell hay Korn-Shell) (có thể dùng ^D hoặc exit để thoát)
logout	Thoát khỏi hệ thống chạy với C-Shell
Id	Định danh của người sử dụng
logname	Tên người dùng đang đăng nhập vào hệ thống
man	Trợ giúp
newgrp	Chuyển người dùng sang một nhóm mới
passwd	Thay đổi mật khẩu của người sử dụng
set	Xác định các biến môi trường
tty	Đặt các thông số terminal
uname	Tên của hệ thống (tên host)
who	Cho biết những ai đang đăng nhập hệ thống
whoami	Xem ai đang làm việc tại terminal

- *Trình tác màn hình*

echo	Hiển thị dòng ký tự hay biến lên màn hình
setcolor	Đặt màu nền và chữ của màn hình (chế độ dòng lệnh)

- *Tasklep*

bc	Tính biểu thức số học
cal	Xem lịch
date	Hiển thị và đặt ngày
mail	Gửi - nhận thư tín điện tử
mesg	Cấm/ cho phép hiển thị thông báo trên màn hình (bởi write/hello)

spell	Kiểm tra lỗi chính tả
vi	Soạn thảo văn bản
write/hello	Cho phép gửi dòng thông báo đến những người dùng trong hệ thống
xclock	Đồng hồ trong OpenWin

▪ *Thư mục*

cd	Chuyển tới thư mục mới
copy	Sao chép hai thư mục
mkdir	Tạo thư mục
rmdir	Xóa thư mục
pwd	Trình bày thư mục hiện thời

▪ *File*

more	Trình bày nội dung file
cp	Sao chép một hay nhiều file
find	Tìm vị trí của file
grep	Tìm vị trí của chuỗi ký tự trong file
ls, l, lf, lc	Trình bày tên và thuộc tính của các file trong thư mục
mv	chuyển/ đổi tên một file
sort	Sắp thứ tự nội dung file
wc	Đếm số từ trong file

▪ *Quản lý tiến trình*

kill	Hủy bỏ một tiến trình
ps	Liệt kê các tiến trình đang chạy (thường dùng với tùy chọn -ef)
sleep	Ngừng hoạt động một thời gian

▪ *Kiểm soát chủ quyền*

chgrp	Chuyển chủ quyền file, thư mục từ một nhóm sang một nhóm khác
--------------	---

chmod Thay đổi quyền sở hữu của file hay thư mục

chown Thay đổi người sở hữu file hay thư mục

- *.Kiểm soát in*

cancel Ngừng in

lp In tài liệu ra máy in

lpstat Trạng thái của hàng chờ in

1. 4. d. Shell

Là bộ xử lý lệnh của người sử dụng, cho phép người dùng tạo các lệnh rất phức tạp từ các lệnh đơn giản. Chúng ta có thể coi shell như một ngôn ngữ lập trình cấp cao. Các chức năng chính của shell là:

- Thực hiện lệnh: Lệnh có thể được thực hiện ở chế độ tương tác với người dùng hay chế độ nền. Một số shell còn cung cấp khả năng kiểm soát các công việc được đặt chỉ số.
- Thư viện lệnh nội tại: Các lệnh nội trú trong shell.
- Ngôn ngữ lập trình và môi trường: Cho phép tạo các file shell-script và các cấu trúc điều khiển như **do**, **while**, **until**, **if**, **case**.
- Tên file mở rộng: Cho phép biên dịch tên file ở dạng có các ký tự viết tắt ?
*.
- Các biến môi trường: Cho phép đặt các biến môi trường. Ví dụ:

PATH=/usr/bin

- Kiểm soát I/O (vào/ra) và đổi hướng: Shell định nghĩa các thiết bị vào/ra chuẩn và cho phép ta đổi hướng thiết bị vào/ra của các tiến trình.

Hiện nay người ta hay sử dụng các loại shell - tùy theo loại mà chúng có thể có cú pháp hơi khác nhau một chút - như sau:

- *Bourne-Shell*: là shell cơ bản, có sớm nhất, nhanh, hiệu quả, nhưng ít lệnh. Bourne-Shell được viết bởi Steven Bourne năm 1979 và công bố lần đầu trong UNIX Version 7 (AT&T).

- *C-Shell*: giống như Bourne-Shell nhưng cung cấp thêm các cấu trúc điều khiển, lịch sử lệnh, bí danh, v.v.. trên cơ sở ngôn ngữ C. Được viết bởi Bill Joy và xuất hiện trong thành phần của BSD UNIX từ năm 1981.
- *Korn-Shell*: Kết hợp cả Bourne-Shell và C-Shell.

Trong các bản Linux, shell cơ bản là *bash* (Bourne Again Shell), tương tự như Korn-Shell, nhưng được phổ biến miễn phí. *bash*, do Brian Fox tạo ra 10/1/1988, là shell chuẩn của dự án GNU, một dự án của Free Software Foundation (FSF) do Richard Stallman lãnh đạo, với mục đích viết lại và thay thế các bản UNIX thương phẩm giá đắt bằng bản miễn phí và có cung cấp mã nguồn để tạo và khuyến khích sự phát triển tự do.

I. 4. e. Giao diện đồ họa cho người dùng

Giao tiếp đồ họa và cửa sổ là một khả năng rất mạnh của hệ điều hành UNIX, cho phép hệ điều hành giao tiếp thân thiện hơn với người sử dụng. Hiện nay UNIX cài đặt X Windows (X11) là một môi trường quản lý đồ họa lý tưởng. Trong Sun Solaris còn sử dụng một dạng của X11 với tên gọi là OpenWindows. X Windows và các khía cạnh liên quan sẽ được trình bày chi tiết hơn trong chương *X-Windows* dành cho những người muốn tìm hiểu sâu hơn.

I. 5. Các điểm tương quan giữa MS-DOS và UNIX

- *Bộ biên dịch lệnh*

Cả **sh** (UNIX) và **command.com** (MS-DOS) đều tách khỏi nhân, vì vậy, cho phép chọn các loại bộ biên dịch khác nhau.

Tuy nhiên, trong UNIX cho phép cùng tồn tại cả shell mới và cũ như các tiến trình song song và hoàn toàn độc lập. Bạn cứ thử cùng nạp đồng thời trên máy của mình, ví dụ, cả MS-DOS 6.22 và PC-DOS 6.0 xem sao !

- *Tập lệnh shell*

shell script (UNIX) tương đương như batch file (MS-DOS).

▪ *Các biến môi trường*

Trong cả UNIX và MS-DOS biến môi trường đều là tập hợp các xâu ký tự dùng để chứa các thông số và tùy chọn cho chương trình chạy và có dạng: NAME=VALUE. Trong MS-DOS, giá trị của môi trường chỉ được tham chiếu đến trong phạm vi batch file hoặc chương trình "của nó" đang chạy, mà không thể làm như vậy từ dòng lệnh. Để thêm vào thư mục /new/dir vào đường dẫn có sẵn, trong UNIX chỉ cần gõ dòng lệnh:

```
path=$path:/new/dir
```

▪ *Vào, ra chuẩn và đổi hướng*

Cả UNIX và MS-DOS đều giữ trong bộ nhớ của mình một bảng chỉ số (*file descriptor* và *file handler*, tương ứng) để điều khiển các file mở.

Cả UNIX và MS-DOS đều dùng > : đổi hướng nhập, < : đổi hướng xuất. Đổi hướng nghĩa là thay đổi các thiết bị vào, ra chuẩn.

Cơ chế đường ống (pipeline) đều có ở MS-DOS và UNIX, với cùng cú pháp (|). Ví dụ, để in nội dung thư mục được lựa chọn:

```
MS-DOS:  dir|sort > prn      UNIX : ls|sort > /dev/lp1
```

▪ *Khởi động hệ thống*

Khi khởi động MS-DOS lấy các thông số thiết lập hệ thống và trình điều khiển thiết bị từ config.sys. Sau đó, chạy các lệnh trong autoexec.bat để khởi tạo môi trường hoạt động ban đầu của người dùng.

Đối với UNIX, các thông số cấu hình được lấy trong /etc/inittab và các lệnh trong /etc/rc được chạy. Tuy nhiên, các chức năng được thực hiện phức tạp hơn. /etc/inittab xác định nếu cấu hình phần cứng thay đổi khi khởi động máy thì nhân của hệ điều hành sẽ được xây dựng lại. Thêm nữa, /etc/inittab xác định chương trình nào sẽ chạy ở các giai đoạn khác nhau của chu kỳ hoạt động hệ thống UNIX (các mức chạy). Nhưng chức năng trước tiên của nó là xác định cổng

nào nối với thiết bị đầu cuối và thiết lập ban đầu của thiết bị đầu cuối đó. `/etc/rc` là một shell script, sẽ tự động khởi tạo hệ thống, và khởi tạo ban đầu môi trường làm việc theo quy định trước cho mỗi người dùng mỗi khi người dùng mới đăng nhập hệ thống.

- *Cấu trúc hệ thống file*

Đều là cấu trúc cây thư mục, nhưng giữa MS-DOS và UNIX có những khác nhau cơ bản. Thứ nhất, thư mục UNIX chỉ chứa tên file và con trỏ đến dữ liệu của file. Do đó, nếu thư mục MS-DOS phải giữ 2-3 bản sao của file thì ở UNIX chỉ cần giữ con trỏ tới file và file trong UNIX có thể có bí danh (tên khác). Thứ hai, nếu tên đường dẫn MS-DOS chỉ có thể liên hệ tới các thư mục và file, thì tên đường dẫn UNIX, ngoài ra, còn có thể liên hệ tới thiết bị hoặc đường ống. Cuối cùng, hệ thống file của MS-DOS có thể quản lý một số lượng có hạn (nhớ lại dòng lệnh `LASTDRIVE=Z` trong `config.sys`) các ổ đĩa vật lý và logic - mỗi ổ đĩa phải có tên và thư mục gốc (ví dụ. `C:\>`). Hệ thống file UNIX chỉ có duy nhất một thư mục gốc `/`. Để liên kết các thư mục trên các ổ đĩa khác nhau, thậm chí của các ổ đĩa ở xa, UNIX dùng lệnh `mount`.

- *Thuộc tính của file*

File trong MS-DOS có các thuộc tính *read only*, *archive*, *hidden*, *system*, *volume ID*. UNIX cũng có tất cả các thuộc tính của file, gọi là *file mode*, bao gồm 9 bit chia làm 3 nhóm cho 3 lớp người dùng: *owner* (người sở hữu file hoặc thư mục), *group* (nhóm mà người sở hữu đó tham gia), *other* (tất cả các người dùng hay nói cách khác, thế giới còn lại) và có thể thay đổi được bằng lệnh `chmod` (sẽ được trình bày kỹ hơn ở phần sau). Các thuộc tính của file trong UNIX là *(không)/đọc được*, *(không)/ghi được* (tức là sửa chữa được hay không) và *(không)/thực thi được* – đối với file, và tương ứng đối với thư mục là *(không)/truy nhập được*, *(không)/ghi được* file vào thư mục, và *(không)/thực hiện được lệnh* trong thư mục.

Sau đây là bảng các lệnh tương đương của MS-DOS và UNIX. Những điểm khác biệt, hay ví dụ minh họa, được chỉ ra trong cột cuối bên phải:

DOS	Chức năng	UNIX	Lưu ý
cd	đổi thư mục	cd	
md	tao thư mục	mkdir	
rd	xóa thư mục	rmdir	
copy	sao chép file	cp copy tar	sao chép file sao chép thư mục sao chép lên đĩa mềm
ren	đổi tên file	mv	
type	liệt kê nội dung file	cat more	
xcopy	sao chép thư mục	copy	
del	xóa file	rm	
dir	liệt kê danh sách file và thư mục	ls	
edit	soạn thảo văn bản	vi	
find	tìm một đoạn văn trong file	grep	
print	in file	lp, copy	lp <ten file> &

▪ *TSR và Daemon*

MS-DOS cung cấp loại tiến trình nền (terminate and stay resident - TSR) không phải là các tiến trình độc lập như trong UNIX. Trong UNIX có thể thực hiện đồng thời hai hay nhiều tiến trình. Để chạy tiến trình nền, cần chạy một lệnh theo sau bởi ký hiệu **&**, ngay sau đó shell trở lại dấu nhắc hệ thống và sẵn sàng chạy độc lập một lệnh tiếp. Nếu lệnh nền tách rời khỏi thiết bị đầu cuối, nó trở thành *daemon*.

II. Đăng nhập hệ thống

II. 1. Bắt đầu và kết thúc phiên làm việc

Khi bắt đầu làm việc trên hệ thống bạn phải đăng nhập vào hệ thống đó bằng thủ tục **login**. Việc đăng nhập báo cho hệ thống biết bạn là ai và các chú quyền làm việc của bạn, khi kết thúc bạn phải **logout**. Khi đó không có một ai khác có thể truy xuất file của bạn nếu không được phép. Trong một hệ thống có nhiều người sử dụng, mỗi người có một tên tài khoản người dùng và một mật khẩu duy nhất.

II. 1. a. Đăng nhập hệ thống

Khi hệ thống đã sẵn sàng ta sẽ thấy trên màn hình hiện lên các dòng thông báo các dịch vụ đã hoạt động và yêu cầu điền tiếp vào dòng lệnh:

```
login:
```

Bạn phải đưa vào tên của bạn và nhấn Enter. Kế đó là:

```
password:
```

Bạn hãy đưa vào mật khẩu của bạn. Mật khẩu phải dài ít nhất 6 ký tự và tối đa 13 (hoặc 8) ký tự, có thể thay đổi bởi lệnh **passwd**. Sau đó trên màn hình làm việc của thiết bị đầu cuối của bạn sẽ hiện lên dấu nhắc "\$", và dòng thông báo, ví dụ:

```
Last login: Fri Jul 24 21:53:54 EST 2000 from console..
```

lúc này bạn có thể thực hiện được các lệnh tiếp theo. Nếu mật khẩu sai, bạn được nhắc làm lại thao tác đăng nhập hệ thống.

Nếu bạn là quản trị hệ thống (superuser hay là root) dấu nhắc hệ thống sẽ là "#". Các ký hiệu dùng cho dấu nhắc hệ thống mới mô tả ở trên được áp dụng cho Bourne shell (**sh**), Korn shell (**ksh**). Dấu nhắc hệ thống dùng cho C shell (**cs**h) là

ký hiệu “%”, cho **bash** trong các bản Linux thường là ký hiệu “>”. Thông thường, trong hai trường hợp cuối, người ta hay kết hợp với cách viết giá trị của các biến **PS1**, **PWD** hoặc dùng sự thay thế lệnh (được trình bày trong chương *Môi trường Shell và lập trình Shell*) để hiển thị đồng thời tên tài khoản người dùng, thư mục làm việc hiện thời hay tên máy trên màn hình làm việc của bạn.

Nếu máy của bạn có cài đặt giao diện đồ họa X Windows, ví dụ như Common Desktop Environment (CDE), thao tác đăng nhập cũng được thực hiện tương tự, sau khi bạn chọn loại giao diện XWindows hay OpenWin (của Solaris). Bạn cũng có thể chọn chế độ dòng lệnh, nếu kích chuột vào ô *Command Line*.

Nếu cần đăng nhập vào hệ thống UNIX trên máy ở xa, trước hết, bạn hãy vào hệ thống trên máy của bạn. Sau đó, khởi động trên máy của bạn chương trình đòi hỏi kết nối với máy ở xa theo cú pháp dạng như sau:

```
$ program-name remote-hostname
```

chương trình có thể là **telnet**, **rlogin**, **cu** hoặc **tip** (kết nối bằng modem qua đường điện thoại), **remote-hostname** là tên máy ở xa bạn cần kết nối vào.

II. 1. b. Thoát khỏi hệ thống

Có thể kết thúc phiên làm việc bởi lệnh:

exit	(C-Shell)
logout	(Bourne-Shell, Korn_Shell) hoặc Ctrl-D

II. 2. Giao tiếp với hệ thống

Nếu bạn muốn tìm hiểu kỹ hơn về hệ thống mà bạn vừa đăng nhập, hãy dùng lệnh:

\$ uname	-n	(node name - tên của máy)
	-m	(chỉ ra phần cứng của máy)
	-v	(chỉ ra version - phiên bản của hệ điều hành)
	-s	(chỉ ra tên của hệ thống)

- r (chỉ ra phiên bản của phần mềm)
- a (= -nmvsr, chỉ ra tất cả các thông tin)

Nếu bạn muốn biết bạn là ai trong hệ thống vừa đăng nhập và thiết bị đầu cuối mà bạn vừa nối vào, hãy dùng lệnh (có thể viết liền nhau):

\$ who am i

Lệnh sau cho biết những thông tin trên, nhưng cho tất cả các người dùng đăng nhập hệ thống trong thời điểm đó:

\$ who

Tùy chọn -q không liệt kê thời gian hay thiết bị đầu cuối của tất cả các người dùng. Việc dùng tùy chọn này tương đương với lệnh:

\$ users

Là người dùng trong hệ thống, bạn có thể thực hiện các lệnh theo quyền hạn của mình. Chi tiết sẽ được trình bày lần lượt trong các chương sau.

II. 3. Quy tắc đặt mật khẩu người dùng

Sau đây là những điều nên tránh khi đặt mật khẩu cho người dùng:

- Gán mật khẩu bằng cách dùng phím Enter
- Dùng ngay tên người dùng làm mật khẩu (ví dụ: login: hoa; password: hoa)
- Dùng tên của bạn đã đảo thứ tự làm mật khẩu (ví dụ: login: h oa; password: aoh).
- Dùng tên con, vợ hoặc chồng, người yêu, ... làm mật khẩu.
- Lấy số điện thoại, địa chỉ, ngày sinh ... làm mật khẩu.
- Lấy tên thần tượng hoặc sở thích của mình làm mật khẩu.
- Dùng tên thành phố, địa phương bạn có nhiều liên hệ làm mật khẩu.
- Dùng bất kỳ từ nào sẵn có trong các từ điển thông dụng làm mật khẩu.

Còn các quy tắc đặt mật khẩu mà bạn được khuyên dùng là:

- Hãy thể hiện thật độc đáo.
- Hãy tự đặt mình vào vị trí "kẻ bẻ khóa" và kiểm tra mật khẩu của bạn có thật khó đoán hay không. Nếu không tin chắc vào mức độ an toàn của mật khẩu, hãy thay đổi nó.
- Thử nghiệm lại.
- Dùng lẫn chữ hoa và chữ thường, chữ số và chữ viết.
- Hãy dùng mật khẩu không nhiều hơn 8 ký tự. Mật khẩu khi đánh không hiện trên màn hình. Nếu mật khẩu dài, đặc biệt khi lâu không dùng đến, chắc gì bạn gõ đã đúng ngay, không sai sót?
- Không viết mật khẩu của bạn ra giấy, dù để trong ngăn tủ của bạn. Cố gắng liên hệ ý nghĩa của mật khẩu với cái gì đó đối với bạn thật khó quên.
- Thường xuyên thay đổi mật khẩu. Ba tháng là khung thời gian hợp lý, nhưng bất cứ khi nào bạn nghi bị lộ, nên đổi ngay mật khẩu.
- Hãy chọn xâu ký tự có nội dung mới làm mật khẩu mỗi khi bạn cần thay đổi. Không lấy lại mật khẩu cũ đã từng sử dụng trước đây.

III. Hệ thống file của UNIX

III. 1. Cấu trúc hệ thống file

III. 1. a. Hệ thống thư mục

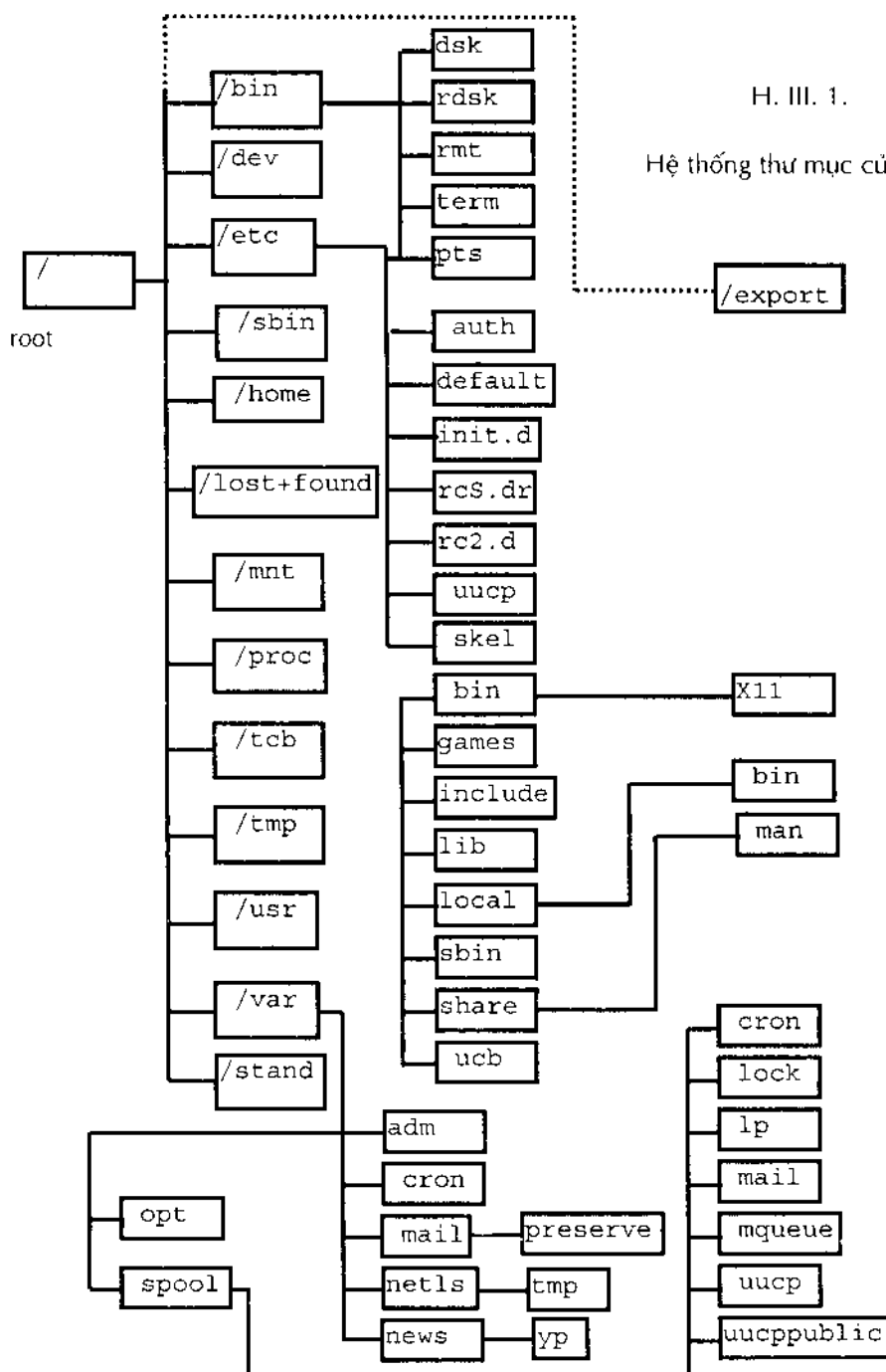
Bạn sẽ gặp trong thực tế hệ thống file có thể khác ít nhiều so với sơ đồ chung được trình bày dưới đây do các hệ thống UNIX không hoàn toàn giống nhau. Nhưng nhìn chung, hệ thống các thư mục trong UNIX như sau (H. III. 1):

/: Thư mục gốc, duy nhất cho cả hệ thống file trong UNIX. Các thư mục và file. không kể tại chỗ hay ở xa, về mặt logic đều được chứa trong thư mục gốc.

/root: Trong một số bản UNIX, đặc biệt là các bản Linux, thư mục này là thư mục home của superuser, được tách ra đứng riêng dưới thư mục gốc. Thư mục /root không rỗng vì còn chứa các thông số môi trường của root, ảnh hưởng tới toàn hệ thống.

/bin: Vị trí truyền thống để chứa các file thực thi (nhị phân) các lệnh và tiện ích của người dùng. Trong một số hệ UNIX, như Sun Solaris, một số file trong /bin liên kết hình thức với các file trong /usr/bin, như ufs.boot - chương trình nạp hệ thống. /bin đôi khi liên kết với /usr/bin. Các thư mục khác cũng chứa lệnh UNIX là /usr/bin, /usr/ucb, /usr/ccs.

/dev: Chứa các file đặc biệt - thiết bị trong UNIX được xem như file. Trong các bản UNIX System V thư mục này được chia thành các thư mục con cho từng loại thiết bị: disk và rdsk - cho đăng nhập đĩa ở các chế độ khối và ký tự, diskette và rdiskette - tương tự như vậy đối với đĩa mềm, mt và rmt - cho thiết bị băng từ, term - cho các thiết bị đầu cuối (các cổng nối tiếp), pts - cho các thiết bị đầu cuối giả định, lpt - cho máy in, v.v.. Trong BSD UNIX /dev không có các thư mục con. Solaris đưa vào thêm một cây thư mục, bắt đầu bằng /devices, và nhiều file dưới /dev được liên kết với các thư mục con của /devices.



Ví dụ: `/dev/dsk/c0t0d0s0` hoặc `/dev/dsk/c0t0d0p0` thường chỉ phần đĩa dành cho thư mục gốc trong Solaris (**c** - controller: các ổ đĩa SCSI đều có ID, ở đây ID là 0; **t** - target: đích, **d** - driver: ổ đĩa; **s** - slice: phân mảnh để phân phát dung lượng đĩa và **p** - partition đĩa bình thường).

`/etc` và `/sbin`: Các thư mục chứa các file dữ liệu hệ thống và thực thi, đặc biệt rất nhiều các file quản lý hệ thống và cấu hình. Các mã nhị phân thực thi của các lệnh quản lý hệ thống theo truyền thống được chứa tại `/etc`. Trong các bản UNIX gần đây, các file đó được chứa trong `/sbin` và `/usr/sbin`, nhưng các liên kết hình thức về `/etc` vẫn được tạo ra. Trong SunOS một số lệnh quản lý còn nằm trong `/usr/etc`. Thư mục con `/etc/default` lưu những giá trị ban đầu của các thông số cho nhiều lệnh khác nhau. Nên chú ý, cấu hình truyền thống trong Solaris được đặt ở `/etc` và `/etc/uucp`.

`/export`: Trong Solaris thư mục này cho phép các client (máy khách) có quyền đọc/ghi. Thường dùng lệnh **share** để chia sẻ tài nguyên cho các client.

`/home`: Được coi như điểm mount (dịch nghĩa: điểm lắp ghép, nối) cho các thư mục phân cấp cho người dùng. Thường rỗng.

`/lost+found`: Thư mục chứa các file bị mất. Chú ý *không* xóa nội dung thư mục này khi máy của bạn đang gặp sự cố. Lỗi ổ đĩa hoặc tắt máy không đúng (ngắt điện bất thành linh) có thể làm file *bị mất*: những địa chỉ trên đĩa được đánh dấu là đang được dùng trong cấu trúc dữ liệu của đĩa, nhưng không đọc được trong bất cứ thư mục nào. Khi khởi động trở lại, chương trình **fsck** sẽ tìm thấy chúng và dọn vào `/lost+found`. Mỗi phần đĩa (disk partition) hay thư mục đều có `/lost+found` của mình với chủ quyền thuộc về `root`.

`/mnt`: Điểm mount tạm thời. Thường được hiểu là thư mục rỗng, bạn có thể tạm thời nối vào đó một hệ thống file khác.

`/proc`: Thư mục tiến trình. Các file trong thư mục này ứng với các tiến trình được thực hiện bởi các lệnh gọi hệ thống (đầu vào trong bảng tiến trình của nhân hệ điều hành).

`/tc`: Cây thư mục chứa các file bảo mật cơ sở dữ liệu (tc - trusted computing base) với khả năng bảo mật nâng cao. Có trong các bản SCO UNIX, HP-UX 10, Digital UNIX. Các file cấu hình cho tc nằm trong `/etc/auth`.

`/usr`: Điểm mount hệ thống file `usr`, chứa các thư mục con cho các chương trình được tạo ra tại chỗ, các lệnh quản lý, các lệnh người dùng chạy được (dạng nhị phân), các thư viện chuẩn, trước hết là cho ngôn ngữ C/C++, chia sẻ dùng chung, từ điển tiếng Anh, các chương trình hệ thống và giao diện đồ họa, các chương trình ứng dụng... Đây là một thư mục quan trọng chứa các lệnh và tiện ích của người dùng, các lệnh của các chương trình ứng dụng được cài đặt vào hệ thống.

`/lib`: Liên kết hình thức với `/usr/lib`.

`/stand`: Chỗ chứa ảnh của nhân hệ điều hành trong HP-UX 10. Trong Solaris đó là thư mục `/kernel` nằm dưới `/` (phiên bản 2.4) hay trong thư mục `/platform` (từ phiên bản 2.5 trở về sau).

`/var`: Chứa các file tài khoản hệ thống (`/var/admin`), hàng chờ in hay E-mail (`/var/spool`), các file `*.log` ghi lại quá trình chạy (`/var/log`), một số các phần mềm tùy chọn (`/var`, `/var/opt`).

`/tmp` và `/temp`: Thư mục chứa các thư mục, file tạm thời, thường sẽ bị xóa khi kết thúc phiên làm việc hay khi tiện ích dọn dẹp hệ thống được thực hiện. Các thư mục này có thể được superuser hay người dùng khác tạo ra sau khi cài đặt hệ thống.

III. 1. b. Các thành phần khác

Các thành phần trong hệ thống file còn bao gồm:

- **superblock**: lưu giữ thông tin về hệ thống file và dữ liệu vật lý chứa trong nó. Các superblock trong hệ thống file UNIX, trừ loại hệ thống file S5, không tập trung tại phần đầu đĩa hoặc partition như bảng FAT của MS-DOS, mà nằm rải rác trong đĩa hoặc partition.

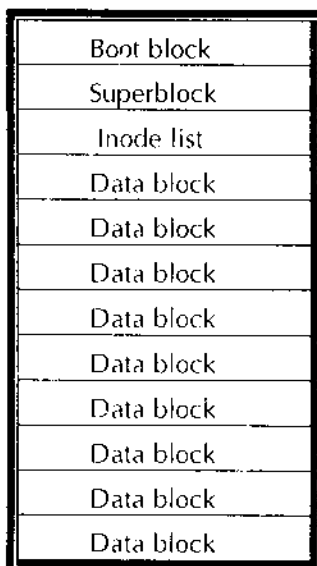
- **inode list**: danh sách các inode.

- **data blocks**: nơi lưu giữ dữ liệu.

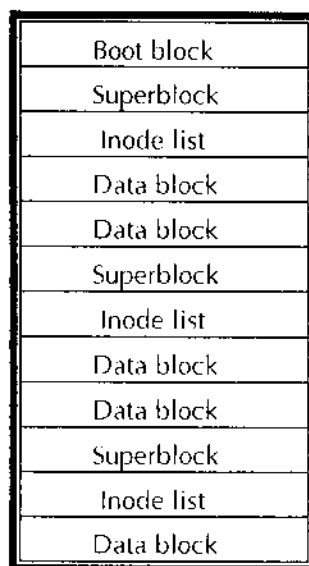
III. 1. c. Phân loại hệ thống file UNIX

Hệ thống file UNIX, theo lịch sử, được phân thành 2 loại chính từ 2 nhánh: AT&T - System V → **S5** và BSD - Fast File System → **FFS** và về sau - UNIX File System → **UFS**. Tuy nhiên, các bản UNIX hiện nay đều có khả năng hỗ trợ các loại hệ thống file khác nhau.

S5: Cấu trúc hệ thống file S5 được trình bày trên hình H. III. 2, chứa một *boot block*, một *superblock*, một *inode list* và còn lại là các *data block*. Boot block (logical block 0) chiếm các sector từ 0 tới 15, tùy theo hãng sản xuất UNIX, chứa các bán sao đầu và thứ hai của chương trình khởi động. Nếu bạn tạo một hệ thống file không tự khởi động thì không gian đĩa dành cho boot block vẫn tồn tại, nhưng không dùng hoặc truy nhập được. Superblock chiếm các sector từ 16 trở đi. Các data block có kích thước 512, 1024 hoặc 2048 byte.



H. III. 2. Hệ thống file S5



H. III. 3. Hệ thống file FFS/UFS

FFS/UFS: Cũng gồm các thành phần như trong S5. Tuy nhiên, superblock và inode list không chỉ xuất hiện một lần. Chúng được chia thành nhiều phần nhỏ và rải khắp, lẫn giữa các data block trong đĩa hoặc partition (xem hình H. III. 3). UFS hỗ trợ kích thước data block tới 8192 byte. UFS nhanh hơn và hiệu quả hơn, theo

dánh giá chung, khoảng 10 lần so với S5. Hiện nay, hầu như tất cả các bản UNIX thuộc System V cũng đều có thể sử dụng UFS.

Hệ thống file UNIX, nhìn từ góc độ mạng, còn chia thành các loại sau:

- Network File System (NFS): lần đầu tiên được áp dụng năm 1984 trong SunOS. Được thiết kế làm việc độc lập, không phụ thuộc phần cứng, NFS cho phép chia sẻ thư mục và file qua mạng có môi trường không đồng nhất. Ví dụ: trong mạng của Ngân hàng Công thương Việt Nam, các máy tính trong một thời gian khá dài đã làm việc với các hệ điều hành Solaris, MS-DOS, Novell NetWare, Windows for Workgroup và Windows9x.

- Cố gắng đưa ra điều gì đó tương tự, AT&T dùng Remote File Sharing (RFS). Không giống như NFS, RFS đòi hỏi môi trường thống nhất: tất cả các máy trong mạng đều phải chạy trên UNIX System V Release 3 trở lên.

Một vài loại hệ thống file UNIX khác:

- BFS (Boot File System): các tiện ích khởi động và đưa hệ thống vào hoạt động. Tạo lập một hệ thống file phẳng, chỉ có một thư mục. Dùng sau khi có sự cố.

- PCFS (PC File System): dùng được dạng DOS format.

- DTFS (Desk Top File System): của SCO, có thêm nén và tiện ích phục hồi sau xóa. Dùng cho thị trường đĩa dung lượng nhỏ.

- HTFS (High Throughput File System): của SCO, có thêm khả năng ghi nhật ký, bản ghi các thay đổi cần làm.

- VFS (Virtual File System): sản phẩm của Sun, như là giải pháp kế tiếp của NFS, cung cấp giao diện giữa nhân và inode của các tác vụ không phụ thuộc vào hệ thống file đang chạy.

III. 2. File, thư mục, nguyên tắc đặt tên và đường dẫn

III. 2. a. File

Hệ thống file UNIX bao gồm các file thường, theo nghĩa chúng ta vẫn hiểu tương tự như đối với MS-DOS, và các file đặc biệt. Các file đặc biệt sẽ thuộc một trong các loại sau:

- File thiết bị ký tự: Tất cả các loại thiết bị xử lý tín hiệu theo byte (máy in, scanner, máy đọc mã vạch, thiết bị đầu cuối, đường truyền thông, ...). Thường các file này là đơn hướng, nghĩa là nếu đọc được thì không ghi được, hoặc ngược lại.
- File thiết bị khối: Toàn bộ hay một partition của đĩa có thể được xem một cách đơn giản như là một file thường có kích thước lớn. File thiết bị khối truy nhập dữ liệu theo khối 512, 1024, 8192 byte hay lớn hơn. Tuy nhiên, không phải tất cả các lệnh cho file thường đều dùng được cho loại này. Ví dụ, các lệnh **append** và **rm** sẽ không có hiệu lực.
- Đường ống: Loại file này có điểm khác biệt là không cần đến các thiết bị ngoại vi.
- Liên kết cứng (*hard link*) và liên kết hình thức (*symbolic link*).

UNIX có hai dạng tổ chức file. Thứ nhất là dạng vật lý (file được tìm thấy trên đĩa cứng). Thứ hai là logic (file thuộc sở hữu của ai). Với tổ chức vật lý, UNIX sử dụng một tập hợp các bảng gọi là “*inode*” (information node - nút thông tin). Với tổ chức logic nó sử dụng hệ thống thư mục cây phân cấp (trong UNIX một thư mục là một file chứa danh sách của tất cả các file và thư mục con của thư mục đó). Các phần tử của thư mục có dạng:

inode	tên file
-------	----------

Mỗi inode chứa các thông tin liên quan đến một file, bao gồm:

- **Inode**: được đánh số thứ tự tăng dần. Số 0 và 1 dành cho các mục đích đặc biệt, số 2 dành cho thư mục gốc.
- **Loại**: gồm 2 ký tự chỉ thị loại file: 01 - đường ống đã đặt tên; 02 - file thiết bị ký tự; 04 - thư mục; 06 - file thiết bị khối; 10 - file thường; 12 - liên kết hình thức; 14 - socket.
- **Chủ quyền**: bao gồm 3 nhóm, mỗi nhóm 3 bit để kiểm soát quyền truy xuất và thực thi của người sở hữu, nhóm, chung.
- **Liên kết** (cứng): trong UNIX, một file vật lý (inode) có thể được liên kết với một hay nhiều file logic (tên file). Liên kết chỉ ra số file logic liên kết với file

vật lý, nói cách khác, số tên hoặc bí danh hiện có của file vật lý (cực đại khoảng 1000).

- **UID** (User IDentification): định danh của người sở hữu.
- **GID** (Group IDentification): định danh của nhóm sở hữu.
- **Thời gian**: ghi lại thời gian tạo, truy nhập và sửa đổi file.
- **Địa chỉ khối dữ liệu đầu, thứ cấp, cấp ba** (kích thước cực đại của file, tương ứng, là $1.7 \cdot 10^7$, $3.4 \cdot 10^{10}$, $7.0 \cdot 10^{13}$ bytes).

Inode	Loại	Chú quyền	Liên kết	UID	GID	Thời gian	Kích thước	Địa chỉ khối		
								Dữ liệu đầu	Dữ liệu thứ cấp	Dữ liệu cấp ba

Inode không chứa tên file. Tên file được xác định trong bảng thông tin thư mục.

Liên kết cứng là liên kết mặc định, được định nghĩa là dòng vào chuẩn trong liệt kê nội dung thư mục (nghĩa là một "file"), giống như khi một file đã được tạo ra. Liên kết cứng chỉ có thể được tạo ra với các file đang tồn tại, và không thể tồn tại giữa các hệ thống file (các phần đĩa cứng, hệ thống file được mount vào, ...). Nếu muốn xóa một file, ta phải xóa hết các liên kết cứng tới file đó, kể cả tên mà file đó mang lần đầu khi được tạo ra. Một khi liên kết cứng cuối cùng bị xóa, inode của file được giải phóng và file không còn tồn tại.

Ngược lại, *liên kết hình thức*, tuy cũng là một dòng vào đặc biệt của thư mục, nhưng có thể chỉ tới không chỉ một tên khác của file, mà còn có thể tới một thư mục, một hệ thống file. Hơn nữa, ta còn có thể tạo liên kết hình thức tới một file hiện thời không có mặt trong hệ thống file; việc xóa file được một liên kết hình thức chỉ tới không ảnh hưởng hay làm thay đổi liên kết đó. Trong một số trường hợp lệnh tác dụng lên liên kết hình thức tới một thư mục có thể đưa ra các kết quả khác nhau. Ví dụ, lệnh **ls** tác dụng lên một liên kết hình thức với một thư mục cho ta liệt kê các file trong thư mục đó; nhưng lệnh **ls -l** tác dụng lên liên kết hình thức lại hiển thị các thông tin về chính liên kết đó.

III. 2. b. Thư mục

UNIX tổ chức hệ thống file bao gồm chỉ một thư mục gốc (/) mà từ đó các thư mục con của nó được gắn vào một cách trực tiếp hay gián tiếp. Có một vài thư mục con chuẩn /bin, /usr, /etc, v.v... Mỗi thư mục này lại chứa các file hay thư mục con. Thư mục trong UNIX thực chất cũng là file. Số của inode trong trường hợp này đơn giản là số chỉ tới bảng các file của đĩa, nằm ở đầu mỗi cấu trúc cây hệ thống file. Các inode này có 2 trạng thái: còn trống hoặc được dùng. Số lượng các inode là xác định một khi cấu trúc cây hệ thống file được tạo ra, nhưng số file tích cực thay đổi theo thời gian.

Có hai file luôn luôn tồn tại trong mọi thư mục: . và .. File . tự nó là thư mục, là liên kết tới chính thư mục chứa nó. File .. cũng là thư mục, là liên kết tới thư mục ngay phía trên, chứa thư mục hiện thời (còn được gọi là thư mục cha).

Có thể sử dụng đường dẫn đầy đủ để xác định một file, ví dụ: /export/NVA/tailieu1. Bạn cũng có thể sử dụng chỉ tên file nếu file được chứa trong thư mục hiện thời. Thường khi bạn đăng nhập, thư mục hiện thời sẽ được đặt đến thư mục "home". Đây là thư mục được thiết lập bởi người quản trị hệ thống dành cho bạn, nó thường trùng với tên đăng nhập của bạn. Đường dẫn đầy đủ còn được gọi là đường dẫn *tuyệt đối*. Đường dẫn *tương đối* không bắt đầu bằng dấu "/", mà chỉ cho biết vị trí file ta cần truy nhập so với thư mục hiện thời. Dấu chấm "." di ghép với tên file, ví dụ, ./xin chào, là đường dẫn tới file xin chào ở thư mục hiện thời. Tương tự như vậy, đường dẫn chứa dấu ".." chỉ tới thư mục cha, chứa dấu "../.." - tới thư mục "ông", ... Thông thường, trong cách dùng đường dẫn tương đối, người ta dùng ký hiệu "~" để chỉ thư mục home của người dùng đang ở trong phiên làm việc của hệ thống UNIX. Điều này đặc biệt hay được thấy trong các bản Linux.

UNIX sử dụng lệnh **mount** để nhận hệ thống file. "*Mount*" là tiến trình thâm nhập một hệ thống file hay một thư mục phân cấp của một hệ thống file từ đĩa hay băng từ. Đặc biệt, đối với hệ điều hành UNIX (ví dụ, Sun Solaris) bạn còn có thể truy xuất hệ thống file hay thư mục của máy từ xa thông qua môi trường mạng.

Khi máy tính khởi động nó tự động chạy chương trình mount và gài những file cần thiết để hoạt động hoặc cục bộ hoặc từ xa. Bạn có thể sử dụng lệnh **mount** để

truy xuất thêm các file hay thư mục. Khi máy tính cài một thư mục phân cấp, nó cài thư mục vào một điểm mount. Điểm mount được cấp phát ở hệ thống file cục bộ, thường điểm mount là một thư mục rỗng trước khi được gán nội dung cụ thể bằng lệnh **mount**. Muốn gỡ bỏ nội dung trên, ta dùng lệnh **unmount**.

Dưới đây là ví dụ liệt kê các file được đọc (không cùng trong thư mục !!):

```
drwxr-xr-x 2 tuan   users 32    Jul  25 18:35 .
drwxr--r-- 1 tuan   users 48    Jul  25 18:35 ..
-rwxr-xr-x 2 tuan   staff 1024  Jul  20 10:25 /users/.profile
crw-rw-rw- 1 root   sys    7,0   Mar  12 19:08 /dev/lp0
                                     /export/output1
lrwxr-x--- 3 oracle dba     2195  Feb  25 09:28          →
                                     /var/kg1
brw-r----- 2 root   root   3,0   Jul  25 18:35 /dev/dsk/sd0
```

Tên file trong UNIX có thể dài 256 ký tự, ngoại trừ các ký tự đặc biệt sau: ! " ' ; / \$ < > () [] . { }. Ngoài ra ta cũng có thể sử dụng các ký tự sau:

- * đại diện cho một, nhiều hoặc không ký tự nào.
- ? đại diện cho một ký tự đơn
- [...] đại diện cho một dãy ký tự có thứ tự trong bảng chữ cái. Ví dụ: liệt kê tất cả các thư mục bắt đầu bằng chữ c, d, e: `lc [cde] *`

File có tên bắt đầu bằng dấu chấm "." là file ẩn.

III. 3. Chủ quyền truy nhập file

Những người dùng khi truy xuất các file trong UNIX phải bị kiểm soát bởi "chủ quyền" (được quy định trong trường chủ quyền của inode). UNIX đoán nhận 3 lớp chủ quyền, chủ quyền đọc (r), chủ quyền ghi (w), chủ quyền thực hiện (x). Nếu file là thư mục thì các chủ quyền trở thành chủ quyền trình bày nội dung của thư mục (r), thêm hay loại bỏ file khỏi thư mục (w), và tìm file hay thư mục con trong thư mục (x).

Ba chủ quyền đối với file và ba chủ quyền đối với người dùng tạo nên 9 chủ quyền đối với một file. Sử dụng lệnh **ls -l** sẽ hiển thị các chủ quyền này cho từng file. Ví dụ, ta chỉ để ý tới khu vực hiển thị các bit chủ quyền:

```
$ ls -l
-rwxr-xr--    ...
drwxr-xr--    ...
```

Ký tự đầu xác định loại file (d : thư mục, b : file thiết bị khối, c : file thiết bị ký tự, p : đường ống, l: liên kết, - : file thường), 3 ký tự kế tiếp xác định chủ quyền của người sở hữu (UID), 3 ký tự tiếp theo nữa xác định chủ quyền của nhóm (GID), 3 ký tự cuối là chủ quyền của một người dùng bất kỳ. Dấu “-” chỉ thị là không có chủ quyền.

Bạn cũng có thể sử dụng số bát phân để thể hiện chủ quyền.

Số	Ý nghĩa
04	chủ quyền đọc
02	chủ quyền ghi
01	chủ quyền thực hiện

Ví dụ: cho phép đọc ghi sẽ có số bát phân tương ứng là 06.

UNIX cũng đoán nhận 3 lớp người dùng trên dựa vào định danh (ID) liên hệ với mỗi file. Mỗi file có định danh người dùng và định danh nhóm người sử dụng. Người dùng thay đổi cách cho phép bản thân mình, nhóm của mình hoặc bất kỳ người dùng khác làm việc với file bằng lệnh **chmod**. Mặc định, người tạo file là người sở hữu và nhóm sở hữu là nhóm của người tạo file. Dĩ nhiên, ta có thể thay đổi các thiết lập đó bởi lệnh **chown** và **chgrp**.

Các chủ quyền mở rộng cho phép điều khiển uyển chuyển hơn. Từ trước tới bây giờ, chúng ta mới đề cập tới định danh thật sự của người và nhóm sở hữu (UID và GID). Đôi khi trong thời gian thực hiện lệnh, cần phải làm sao cho hệ thống nhận diện bạn như là một người dùng khác. Định danh của người dùng khác vừa nói đến chính là định danh hiệu dụng (SUID) của bạn. Hãy xét một ví dụ như sau. Thông tin về tài khoản người dùng của bạn nằm trong file `/etc/passwd` được bảo vệ bởi một loạt chủ quyền đối với file:

```
-rw-r--r--  1  root 1364 May 11 10:45  /etc/passwd
```

Bạn không phải là `superuser` hay `root`, nên không thể ghi, sửa chữa mật khẩu của người dùng ngang hàng khác được do chủ quyền lệnh `passwd` không thuộc về bạn trong trường hợp này:

```
-rwxr-xr-x 3 root 303104 Jul 19 07:55 /usr/bin/passwd
```

Nếu bạn thay được "s" vào vị trí của "x" trong chủ quyền người sở hữu (ở đây là `root`), hay giá trị bằng số là 4455:

```
-rwsr-xr-x 3 root 303104 Jul 19 07:55 /usr/bin/passwd
```

thì khi thay đổi mật khẩu, hệ thống xem bạn như là `root` - người sở hữu file `/usr/bin/passwd`. Đối với chủ quyền hiệu dụng với nhóm `SGID` cũng tương tự như vậy. Trong tiến trình thực hiện lệnh `mail`, bạn đóng vai trò của `root` ở trong nhóm `buudien`, nếu bạn thay đổi chủ quyền của file `/usr/bin/mail` như sau (giá trị bằng số là 4655):

```
-rwsr-sr-x 1 root buudien 303104 Jul 19 07:55  
/usr/bin/mail
```

thì bạn đóng vai trò của `root` và là thành viên của nhóm `buudien` khi thực hiện lệnh `mail`.

Chú ý: chủ quyền `SUID`, `SGID` chỉ dùng cho các file nhị phân hoặc đã biên dịch.

Một chủ quyền mở rộng khác, được gọi là *sticky bit* (bằng số: 1777) và được dùng như công cụ điều khiển bộ nhớ (nghĩa đen *sticky* là bám dính). Khi được dùng, *sticky bit* báo cho nhân hệ điều hành giữ lại một bản sao chương trình trong RAM, thậm chí cả khi chương trình ở thời điểm đó không chạy. Do sử dụng tài nguyên bộ nhớ, lệnh chạy chương trình đó được thực hiện rất nhanh mỗi khi ta gọi lại nó. Đối với thư mục, *sticky bit* có ý nghĩa khác hẳn. Nó không cho phép người dùng xóa những file không thuộc sở hữu khỏi các thư mục dùng chung. Ta thấy nếu đặt chủ quyền này (giá trị bằng số là 4177):

```
drwxrwxrwt 1 root root 512 Aug 14 21:15 /usr/lib/fs/ufs
```

mặc dù thư mục `/usr/lib/fs/ufs` bất cứ người dùng nào cũng có quyền đọc và ghi, nhưng bạn sẽ không xóa nổi file trong đó, nếu bạn không phải là người sở hữu.

Chú quyền đối với thư mục còn được quy định như sau đối với các ID hiệu dụng của người dùng và nhóm:

- Quyền đọc:

- 0444 : coi thư mục như file thường
- 0400 : nếu SUID tiến trình trùng UID của thư mục, tiến trình có thể đọc được thư mục.
- 0040 : nếu SGID tiến trình trùng GID của thư mục, tiến trình có thể đọc được thư mục.
- 0004 : bất kỳ tiến trình nào cũng đọc được thư mục.

- Quyền ghi:

- 0222 : không thể ghi được vào thư mục.
- 0200 : nếu SUID của tiến trình trùng UID của thư mục, có thể ghi được vào thư mục.
- 0020 : nếu SGID của tiến trình trùng GID của thư mục, có thể ghi được vào thư mục.
- 0002 : có thể ghi được vào thư mục.

- Quyền thực thi (trình bày nội dung thư mục, tìm, thêm, bớt file trong thư mục):

- 0111 : không thể được.
- 0100 : nếu SUID của tiến trình trùng UID của thư mục, có thể được.
- 0010 : nếu SGID của tiến trình trùng GID của thư mục, có thể được.
- 0001 : có thể được, không bị hạn chế.

III. 4. Các lệnh thao tác căn bản

III. 4. a. Các lệnh thao tác trên thư mục

- *Thay đổi thư mục làm việc hiện thời:* `cd [dir]`

Ví dụ (dấu nhắc hệ thống "\$" trong phiên làm việc của người dùng bình thường):

- chuyển đến thư mục `/usr/include`: `$ cd /usr/include`
- chuyển trở lại thư mục "home": `$ cd`
- chuyển đến một thư mục con của thư mục hiện thời: `$ cd ccs`
- chuyển đến thư mục cha: `$ cd ..`
- chuyển đến thư mục ông: `$ cd ../..`

- *Trình bày nội dung của thư mục:* `ls [option(s)] [dir]`

Ví dụ:

đanh sách các file trong thư mục hiện thời: `$ ls .`

```
passwd
hosts
```

đanh sách các file sắp theo thứ tự trong các lệnh: `$ ls -c`

```
hosts passwd
inittab
```

đanh sách file với loại: `$ ls -CF .`

```
bin/    chmod*
etc/    temp
```

(/ chỉ thị bin là thư mục, * chỉ thị chmod là file thực thi)

đanh sách các file ở thư mục khác: `$ ls /bin`

đanh sách một số file trong thư mục: `$ ls -x [pP] *`

```
prigfoot
Pasks
```

danhsách các file trong thư mục: `$ ls -l` (hoặc dùng `$ ll`)

- Khi dùng lệnh `ls -l` ta có thể phân biệt các kiểu file bằng cách xem ký tự đầu tiên của dòng hiển thị, nếu là:
 - `d` : file thư mục
 - `-` : file thường
 - `c` hoặc `b` : file đặc biệt - thiết bị ký tự và thiết bị khối
 - `p` : đường ống đã đặt tên

danhsách file với kích thước: `$ ls -s`

```
total 28
 4 passwd
 1 hosts
```

danhsách file với các thông tin chi tiết: `$ ls -ls`

```
total 28
drwxr-xr-x  2  NVA  adm   98   Sep 13  10:04   NVA
```

(chủ quyền-liên kết-tên người sở hữu-tên nhóm sở hữu-kích thước-thời gian-tên file)

Thông thường, nếu thư mục có nhiều file ta có thể sử dụng đường ống để có thể trình bày danhsách file theo từng trang màn hình. Ví dụ: `$ ls -C | more`

- *Tạo thư mục:* `mkdir [-mp] [dir]`

m

Đặt chủ quyền của thư mục theo số bát phân

p

Tạo các thư mục gián tiếp, nếu tên thư mục gián tiếp trong đường dẫn không tồn tại

Ví dụ, nếu người dùng bình thường (dấu nhắc hệ thống "\$") muốn:

- tạo thư mục con trong thư mục hiện thời:

```
$ mkdir tam1
```

- tạo thư mục con sử dụng đường dẫn tương đối:

```
$ mkdir tam1/chung
```

- tạo thư mục sử dụng đường dẫn đầy đủ:

```
$ mkdir /usr/tam1/tuan
```

- tạo các thư mục gián tiếp:

```
$ mkdir -p /usr/tam2/duc
```

(thư mục tam2 không tồn tại từ trước, do đó tạo cả thư mục tam2 và duc)

- *Trình bày lên (đường dẫn) của thư mục hiện thời: pwd*

```
$ pwd
/usr/tam1/tuan
```

- *Xóa thư mục: rmdir [-ps] dir*

P

Trình bày lần lượt các thư mục bị xóa

S

Xóa thông báo được tạo bởi -p

Ví dụ:

- xóa một thư mục rỗng:

```
$ rmdir tuan
```

- xóa một thư mục không rỗng (có chứa các thư mục con hoặc file):

```
$ rmdir tam2/*
```

hoặc

```
$ rm -ri <dir>
```


III. 4. b. Các lệnh thao tác với file

- Nhận biết một file thường: **file**

Lệnh **file** phân tích nội dung của một file và hiển thị tính chất của thông tin chứa trong file. Ví dụ:

```
$ file /etc/passwd => /etc/passwd: ascii text
```

- Liệt kê file: **ls** [option(s)] dir

Liệt kê danh sách các file: \$ **ls** <dir>

Hiển thị các file ẩn: \$ **ls** -a

Liệt kê thông tin về file: \$ **ls** -l <filename>

- Ghép nội dung một file và liên kết các file thành một file duy nhất: **cat** [option(s)] dir

Ta hãy xem ví dụ sau:

```
$ cat /etc/hosts
202.192.5.100    incombank
```

Tổ hợp hai file thành một: \$ **cat** part1 part2 > report

Ta cũng có thể sử dụng lệnh **cat** để tạo một file từ bàn phím. Ví dụ, ta tạo file có tên thu:

```
$ cat > thu          (tương tự copy con trên DOS)
                     (đưa vào nội dung file)
                     ^D (hoặc dấu chấm câu kết thúc "." - full stop)
```

- Ghép nội dung của file theo từng trang màn hình: **more** [file].
pg [file]

Ví dụ: \$ **more** thu

- *Đổi tên file:* `mv oldfile newfile` hay `mv filename dir`

Ví dụ:

Thay đổi tên của file ở thư mục hiện thời: `$ mv a.out test`

Chuyển file sang thư mục tam1: `$ mv test tam1`

(file mới sẽ là tam1/test)

Chuyển một vài file sang thư mục khác:

```
$ mv test1 test2 ./tam2
```

Chú ý: trong Linux và một số bản UNIX khác nếu ta gõ dòng lệnh trên, các file được chỉ định sẽ không bị di chuyển, thay vào đó có thể nhận được một thông báo lỗi. Người ta sử dụng cho mục đích này lệnh chuyển nhiều file (multiple move) với cú pháp tương tự `mmv`.

- *Liên thêm một tên mới (một liên kết) cho một file*

Cú pháp:

```
ln filename1 filename2 hay ln filename dirname
```

Ví dụ: - tạo tên thứ hai cho một file

```
$ ln test thu
```

```
$ ls -l *
```

```
3271 test
```

```
3271 thu -> cả hai file này có cùng một inode 3271
```

- tất cả các file trong thư mục /usr/nhct mà thích ứng với *.c sẽ được liên kết với thư mục hiện thời.

```
$ pwd
```

```
/usr/tuan
```

```
$ ln /usr/nhct/ *.c
```

- *Xóa file* `rm [-fri] file`

f

Xóa các file mà không hỏi, thậm chí cả khi chủ quyền ghi không cho phép

r

Chấp nhận thư mục như đối số. Toàn bộ nội dung của thư mục bị xóa (kể cả các thư mục con), sau đó bản thân thư mục cũng bị xóa.

i

Trước khi xóa file máy sẽ yêu cầu ta xác nhận lại.

Ví dụ:

xóa file thu: `$ rm thu`

xóa file có hỏi: `$ rm -i t*`
test: ? y
thu: ? n

xóa tất cả các thư mục con của thư mục tam và bản thân thư mục tam:

```
$ rm -r tam
```

- *Sao chép file* `cp source-file dest-file`

Ví dụ:

```
$ cp /etc/passwd /user/tuan/passwdold
```

- sao chép nhiều file vào một thư mục:

```
$ cp file1 .... filen dir
```

- sao chép tất cả trong thư mục hiện thời vào một thư mục:

```
$ cp * dir
```

- *Tìm kiếm một file:* **find** path -option[s] argument

options các tùy chọn

argument biểu thức tìm kiếm

Một vài dạng hay được dùng:

name filename --> tìm theo tên file

user username --> tìm theo tên người sử dụng

print --> nếu tìm không thấy thì trình bày đường dẫn

Ví dụ: Tìm file thu:

```
$ find / -name thu -print
/export/tam/thu
```

III. 4. c. Các lệnh xử lý dữ liệu trong các file

- *Xem phần đầu của nội dung file:* **head** [-num] file[s]

Lệnh **head** sẽ hiển thị phần đầu của nội dung file trên màn hình. Giá trị mặc định của số dòng nội dung file được hiển thị là 10, nếu không, có thể đặt lại theo tùy chọn -num. Lệnh này tỏ ra tiện dụng khi ta chỉ cần điểm nhanh lại nội dung các file dạng văn bản mà nội dung của nó ta chưa nắm chắc, hoặc cần trích rút phần đầu file được chỉ định rồi đối hướng sang một file khác.

Một số bản UNIX thuộc System V, kể cả dòng AT&T (trong số đó có Solaris các phiên bản 2.5.1 trở về trước) và UnixWare không có lệnh **head**. Thay vào đó, ta có thể dùng lệnh **sed** -10q để hiển thị mười dòng đầu của file. Sau đó điều khiển trở lại dấu nhắc hệ thống. Chi tiết hơn về trình soạn thảo này sẽ được trình bày trong chương X.

Đôi khi ta chỉ cần biết một số thông tin từ các lệnh được đưa vào mà không quan tâm tới toàn bộ thông tin được kết xuất. Một vài bản UNIX, như SCO và DEC Ultrix, có lệnh tên là **pstat** dùng để hiển thị thông tin về file và inode đang được sử dụng. Đầu tiên **pstat** cho biết bao nhiêu file đã được mở, sau đó là các thông tin

về file. Nhưng thông thường ta chỉ quan tâm đến số file được mở. Ví dụ sau cho thấy cách dùng các lệnh **head** và **pstat** để đạt được các thông tin mong muốn:

```
$ pstat -f | head -1
217/800 open files
$ ps -ef | grep tuan | head -3
tuan  1780   1      6    09:25:38    000  0:12   bin/sh
tuan  1792   1      2    10:15:08    000  0:12    vi
tuan  1801  1780   62    16:57:32    000  0:12    ps
$
```

Trong các bản UNIX cũ số dòng hiển thị được bằng lệnh **head** không vượt quá 99.

- *Đọc phần cuối của nội dung file bằng lệnh **tail***

Ngược lại, nếu ta quan tâm tới các dòng cuối của nội dung file được ghi ra ra chuẩn (hiển thị trên thiết bị ra chuẩn là màn hình) thì lệnh thích hợp là **tail** với cú pháp sau:

```
tail [±number [lbcrl]] [file]
tail [-lbcrl] [file]
tail [±number [lbcfl]] [file]
tail [-lbcfl] [file]
```

Số dòng mặc định được hiển thị cũng là 10. Lệnh này nằm trong số các lệnh được ưa dùng, kể cả trong Solaris. Nếu tên file bị bỏ qua thì thông tin được lấy từ nhập chuẩn (thiết bị nhập chuẩn là bàn phím). Lệnh **tail** đặc biệt tiện lợi khi người quản trị hoặc lập trình viên cần kiểm tra các file log đối với các thời điểm cuối cùng. Sau đây là liệt kê các tùy chọn của lệnh **tail**:

l

Số dòng.

b

Số khối.

c

Số ký tự.

r

Đưa ra các thông tin theo thứ tự ngược lại. Giá trị mặc định là hiển thị tất cả nội dung file theo thứ tự ngược lại.

f

Dùng cho các file có độ dài luôn tăng theo thời gian. Nếu những file đó không thuộc đường ống, lệnh **tail** sau khi hiển thị các thông tin cuối tại một thời điểm sẽ không kết thúc, mà sẽ tiếp tục hiển thị thông tin ở cuối file cho các thời điểm tiếp diễn sau này. Tùy chọn này thích hợp cho việc theo dõi một tiến trình.

-number

Chỉ số dòng hay khối hay ký tự kể từ cuối file sẽ được lệnh **tail** hiển thị.

+number

Chỉ số dòng hay khối hay ký tự kể từ vị trí thứ **number** của file trở đi sẽ được lệnh **tail** hiển thị. Tùy chọn này không có trong Solaris.

Kết hợp của hai lệnh **head** và **tail** cho phép ta chọn đúng dòng trong nội dung file để xem. Thực vậy, nếu ta chọn trong file **baocao** dòng thứ mười bằng lệnh:

```
$ head baocao
```

ta bắt buộc phải xem chín dòng đầu tiên một cách không cần thiết. Sẽ thoải mái hơn, nếu ta dùng:

```
$ head baocao | tail -1
```

Tương tự như vậy nếu dòng phải xem có vị trí tương ứng nằm tại nửa cuối file **baocao**:

```
$ tail baocao | head -1
```

- *Tách dữ liệu theo các trường dữ liệu*

Một lệnh khác cũng hay được dùng để xử lý nội dung file là lệnh **cut** có cú pháp:

```
cut -b list [-n] [file]
cut -c list [-n] [file]
cut -f list [-d delim] [-s] [file]
```

Lệnh **cut** cho phép nhặt và hiển thị theo cột các trường của nội dung file. Nếu tên file bị bỏ qua, lệnh sẽ lấy thông tin từ nhập chuẩn. Một trong các tùy chọn **b**, **c**, **f** là bắt buộc. Lệnh này rất linh hoạt và ta phải chú ý đến các tùy chọn thích hợp:

list

Danh sách bằng các số nguyên chỉ các trường, cách nhau bằng dấu phẩy “,”. theo thứ tự tăng dần. Dấu gạch ngang chỉ khoảng các trường. Ví dụ ; 1,4,7 ; 1-3,8 ; -5 (từ 1 tới 5) ; 3- (từ thứ ba trở đi).

-b list

Chỉ vị trí của byte. Ví dụ: -b1-32 - chọn 32 byte đầu tiên của mỗi dòng. Dòng chứa không quá 1023 byte.

-c list

Chỉ vị trí của ký tự.

-f list

Chỉ vị trí của trường, cách nhau bởi dấu phân chia trường.

-d delim

Quy định dấu phân chia trường. Cùng được dùng với tùy chọn -f. Dấu phân chia trường mặc định là Tab. Nếu dùng các dấu đặc biệt hay khoảng trống làm dấu phân chia trường, cần để trong ngoặc. Có thể dùng ký tự nhiều byte làm dấu phân chia trường.

-n

Không xé lẻ ký tự. Khi dùng cùng với tùy chọn -b list, danh sách list được chọn sao cho các ký tự nhiều byte không bị xé lẻ.

-s

Khi dùng cùng với tùy chọn `-f`: nén các dòng không có dấu phân chia trường. Các dòng đó được coi như trong một trường.

▪ *Tán nội dung theo các trường dữ liệu bằng lệnh **paste***

Nhằm mục đích gộp các dòng tương ứng của các file đầu vào. Lệnh này xử lý mỗi một file như một hoặc các cột của một bảng và sau đó dán nội dung của các cột tương ứng lại với nhau theo hàng ngang (gọi là gộp dòng song song). Nếu dòng nào đó của một file đầu vào chỉ gồm các khoảng trống, trong khi các dòng tương ứng của các file đầu vào khác chứa các ký tự bình thường thì lệnh **paste** vẫn giữ nguyên các khoảng trống. Cú pháp của lệnh **paste** là:

```
paste file1 file2 ...  
paste -d list file1 file2 ...  
paste -s [-d list] file1 file2 ...
```

Ý nghĩa các tùy chọn là:

-d list

Nếu không có tùy chọn này ký tự bắt đầu dòng mới đối với mỗi dòng, trừ dòng cuối cùng, được thay bằng ký tự Tab. Tùy chọn này cho phép thay ký tự Tab bằng một hay các ký tự khác đi liền ngay sau `-d`. list cũng có thể là dãy thoát đặc biệt: `\n` (dòng mới), `\t` (Tab), `\\` (dấu gạch xuống), `\0` (chuỗi trống - không phải ký tự 0). Trong trường hợp gộp dòng song song (không có tùy chọn `-s`), các dòng của file đầu vào cuối cùng đều kết thúc bằng dấu bắt đầu dòng mới, không phụ thuộc vào quy định trong list.

-s

Gộp các dòng tiếp theo sau chứ không phải mỗi dòng của mỗi file. Dùng Tab làm dấu kết thúc dòng, nếu không được quy định khác bởi tùy chọn `-d`. Cuối dòng cuối cùng bao giờ cũng là dấu bắt đầu dòng mới, không phụ thuộc vào list.

-

Có thể dùng thay bất cứ file đầu vào nào, để chỉ dòng cho thao tác **paste** lấy từ nhập chuẩn.

Ta hãy xét các ví dụ sau:

- dán đơn giản nội dung hai file theo kiểu gộp dòng song song:

```
$ who
    tuan      tty000   Jun 25   18:30
    kim       ws1      Jun 24   08:40
$ who | cut -c1-16 >/tmp/who.1
$ who | cut -c17- >/tmp/who.2
$ cat who.1
    tuan      tty000
    kim       ws1
$ cat who.2
    Jun 25   18:30
    Jun 24   08:40
$ paste /tmp/who.1 /tmp/who.2
    tuan      tty000   Jun 25   18:30
    kim       ws1      Jun 24   08:40
$ paste /tmp/who.2 /tmp/who.1
    Jun 25   18:30   tuan      tty000
    Jun 24   08:40   kim       ws1
```

- dán dòng có khoảng trống:

```
$ cat file1
    homnay
    homqua
$ cat file 2
    va
$ paste file1 file2
    homnay   va
    homqua
```

- dán liên tiếp:

```
$ cat quymot
```

```

Jan
Feb
Mar
$ paste -s quymot
Jan      Feb      Mar

```

- liệt kê một thư mục theo một cột:

```
$ ls | paste -d" " -
```

- liệt kê một thư mục theo bốn cột:

```
$ ls | paste - - - -
```

- gộp cú hai dòng của file thành một dòng:

```
$ paste -s -d"\t\n" file
```

▪ *Gộp hai dòng nội dung theo các trường dữ liệu từ hai file*

Nếu lệnh **paste** dán các dòng theo các tiêu chí kể trên thì lệnh **join** chỉ gộp các nội dung trong các trường tương ứng của các file đầu vào, theo các tiêu chí nhất định, gọi là các *key* - khóa hay *primary field* - trường sơ cấp. Điều kiện cần có là các file được xử lý phải có các key giống nhau, đều được xếp theo thứ tự vần chữ cái. Cú pháp của lệnh có dạng:

```
join [-optionargument[ -optionargument[...]]] file1 file2
```

Nếu thay tên *file1* bằng dấu gạch ngang "-", nhập chuẩn sẽ được sử dụng. Các trường đầu vào được ngăn cách bằng các dấu mặc định là khoảng trống, Tab hay ký tự bắt đầu dòng mới. Đối với các trường đầu ra (trong file kết quả của lệnh **join**), ký tự phân chia trường có giá trị mặc định là khoảng trống. Sau đây là các tùy chọn và đối số tương ứng của lệnh **join**:

-an

Ngoài kết quả gộp dòng bình thường còn tạo thêm dòng cho mỗi một dòng không được ghép đôi trong file n , với n là 1 hoặc 2.

-e s

Thay thế trường kết quả trống bằng chuỗi ký tự s .

-j n m

Thực hiện gộp trường thứ m của file n . Nếu bỏ qua n , thực hiện gộp trường thứ m của mỗi file. Thứ tự trường được đánh số từ 1.

-o $list$

Mỗi dòng kết quả bao gồm các trường được xác định trong $list$, với mỗi phần tử có dạng $n.m$ (n - số thứ tự của file, m - số thứ tự của trường). Trường chung của các file sẽ không được hiển thị nếu không yêu cầu.

-tc

Dùng ký tự c làm dấu ngăn cách trường cho cả các file đầu vào và đầu ra.

Trong ví dụ sau file `diadanh` có hai trường chỉ tên tỉnh và tên thị xã tương ứng, còn file `danso` gồm hai trường tên tỉnh và số dân:

```
$ cat diadanh
```

Hanoi	Hanoi
Hanam	Phuly
Nghean	Vinh
Quangninh	Halong

```
$ cat danso
```

Hanoi	3500000
Hanam	800000
Nghean	2000000
Quangninh	1000000

```
$ join diadanh danso > diadu
```

```
$ cat diadu
```

Hanoi	Hanoi	3500000
Hanam	Phuly	800000
Nghean	Vinh	2000000
Quangninh	Halong	1000000

- *Tìm chuỗi văn bản bên trong file:*

Cú pháp:

```
grep [-options]limited-regular-expression [filename ...]  
egrep [-options] [-e special-expression] [-f filename]  
full-regular-expression [filename ...]  
fgrep [-options] [-e special-string] [-f filename]  
strings [filename ...]
```

Lệnh **grep** là một trong các lệnh hay được ưa dùng nhất để tìm kiếm một chuỗi văn bản bên trong các file được chỉ định. Nếu tên file bị bỏ qua, nhập chuẩn sẽ được dùng thay thế. Nếu chuỗi văn bản dài hơn một ký tự thì phải để trong hai dấu nháy. Cần thận trọng khi dùng các ký tự đặc biệt \$, *, [, ^, |, (,), \, vì chúng mang ý nghĩa trong shell (xem trong chương *Môi trường Shell và lập trình Shell*). Các tùy chọn có nghĩa như sau:

-b

Đặt thứ tự số khối trước mỗi dòng được tìm thấy. Số thứ tự đầu tiên là 0.

-c

Chỉ hiển thị số dòng có chứa xâu ký tự cần tìm.

-h

Dùng trong trường hợp tìm nhiều file. Ngăn tên file có chứa xâu ký tự không hiển thị thêm vào dòng.

-i

Bỏ qua chữ thường, chữ hoa khi so sánh.

-l

Chỉ hiển thị tên file có chứa các dòng có xâu ký tự, cách nhau bởi ký tự bắt đầu dòng mới (giá trị của biến NEWLINE). Không lặp lại tên file nếu xâu ký tự được tìm thấy nhiều lần.

-n

Đặt số thứ tự của dòng trước mỗi dòng. Số đầu tiên là 1.

-s

Không hiển thị thông báo lỗi khi file không tồn tại hoặc không đọc được.

-v

Hiển thị tất cả các dòng không chứa chuỗi ký tự cần tìm.

-w

Thực hiện tìm kiếm đối với biểu thức như là đối với từ, nếu biểu thức đó được đặt trong "`<`" và "`>`".

Nếu **grep** được dùng khi chuỗi ký tự là các biểu thức hạn chế thì lệnh **egrep** lại được dùng cho các biểu thức đầy đủ, nghĩa là giá trị của các biểu thức có thể sử dụng toàn bộ bảng chữ cái và các ký tự đặc biệt. Do cũng cần thận trọng khi dùng các ký tự đặc biệt `$`, `*`, `[`, `^`, `|`, `(`, `)`, `\`, tốt nhất là ta ghi các biểu thức đầy đủ giữa hai dấu nháy đơn ngược "```" "```". Đối với các biểu thức đầy đủ ta có thêm các quy ước sau:

Dấu	Ý nghĩa
<code>+</code>	Nếu đứng sau biểu thức đầy đủ: tần suất tìm thấy biểu thức đầy đủ có thể lớn hơn 1.
<code>?</code>	Nếu đứng sau biểu thức đầy đủ: tần suất tìm thấy biểu thức đầy đủ là 0 hoặc 1.
<code> </code>	Ngăn cách giữa các biểu thức đầy đủ (hoặc bằng dấu bắt đầu dòng mới): tìm bất kỳ trong số các biểu thức đầy đủ đó.
<code>()</code>	Các biểu thức đầy đủ trong dấu ngoặc đơn được nhóm lại.

So với **grep**, lệnh **egrep** có thêm các tùy chọn sau:

-e *special-expression*

Tìm kiếm biểu thức đặc biệt (biểu thức đầy đủ bắt đầu bằng dấu "`-`")

-f *filename*

Lấy danh sách các biểu thức đầy đủ từ *filename*

Riêng với các chuỗi ký tự ta có thể dùng lệnh **fgrep**. Lệnh không dùng các ký tự đặc biệt `$`, `*`, `[`, `^`, `|`, `(`, `)`, `\`, vì chúng mang ý nghĩa trong shell. Tốt nhất nên đặt các chuỗi ký tự giữa hai dấu "```" "```". So với hai lệnh trên, ở đây ta chỉ có thêm một tùy chọn **-x** để hiển thị chỉ những dòng trùng hợp hoàn toàn với chuỗi ký tự.

Ví dụ: Tìm chuỗi ký tự "ngan hang" trong file *thu*:

```
$ grep "ngan hang" thu
```

ngan hàng công thương

- *Chia file thành các file nhỏ*

Lệnh **split** chia một file lớn thành n file nhỏ theo cú pháp:

```
split [-n] [filename [ name]]
```

Các file kết quả được chia ra có số dòng mặc định là 1000. Tên của chúng là tên name, thêm vào aa cho file đầu ra thứ nhất, ..., zz - cho file cuối cùng. Vậy, số file đầu ra cực đại là 676. Các file đầu ra có độ dài tên cực đại là 254 ký tự (ít hơn độ dài tên file bình thường 2 ký tự).

- *Chia file thành các file nhỏ theo các tiêu chí xác định bởi người dùng*

```
csplit [-s] [-k] [-f prefix] filename  
argument1 [ ...argumentn]
```

Lệnh này đọc file đầu vào filename và chia thành n+1 phần, dựa trên các đối số, với các tên mặc định là xx00 ... xxn, n không lớn hơn 99). File đầu vào filename vẫn được giữ nguyên, nếu không còn cần thiết nữa người dùng phải tự xóa. Nếu filename là dấu gạch ngang "-", các thông tin đầu vào cho lệnh **csplit** sẽ được lấy từ nhập chuẩn. Quy tắc chia thành các phần như sau:

- 00 : Từ đầu file filename cho tới (không bao gồm) dòng được tham chiếu bởi argument1
- 01 : Từ dòng được tham chiếu bởi argument1 cho tới (không bao gồm) dòng được tham chiếu bởi argument2
- v.v.. v.v..
- n : Từ dòng được tham chiếu bởi argumentn cho đến kết thúc file filename

Đối số argument1 ... argumentn của lệnh **csplit** có thể là kết hợp của:

- /rexp/** Tạo file cho phần từ dòng hiện thời đến dòng có chứa biểu thức *rexp*. Sau đó dòng hiện thời lại lấy giá trị là dòng có chứa biểu thức *rexp*. Có thể dùng để xác định đến dòng trước dòng có chứa biểu thức *number* dòng: *rexp-<number>* , hoặc sau dòng có chứa biểu thức *number* dòng: *rexp+<number>*
- %rexp%** Giống như với đối số trên, chỉ khác là không tạo file cho phần đó.
- lnno** Tạo file cho phần từ dòng hiện thời đến dòng thứ *lnno* (không chứa dòng đó). Sau đó, dòng hiện thời lại lấy giá trị là dòng *lnno*.
- {num}** Đối số lặp lại thao tác *num* lần.

Các tùy chọn như sau:

-s

Lệnh **csplit** bình thường hiển thị các số đếm ký tự cho mỗi file được tạo thành. Nếu dùng tùy chọn này sẽ loại việc hiển thị đó

-k

Lệnh **csplit** bình thường xóa các file tạo thành nếu có lỗi. Nếu dùng tùy chọn này các file tạo thành sẽ được giữ lại.

-f prefix

Thay tên mặc định *xx00 ... xxn* của các file được tạo thành bằng các tên *prefix00 ... prefixn*.

Giả sử ta có file *thucpham* sau:

```
$ cat thucpham
Thit
    Ga
    Bo
    Heo
    Vit
Hoaqua
    Dao
    Tao
```

Man
Cam
Dua
Rau
Cai
Muong
Ngot

Ta sẽ chia làm ba phần như sau:

```
$ egrep -n "Thit|Hoqua|Rau" thucpham
1: Thit
6: Hoqua
12: Rau
$ split thucpham 6 12
34
46
18
$ ls -l xx*
-rw-r--r-- 1 daubep cuahang 34 Jun 16 15:30 xx00
-rw-r--r-- 1 daubep cuahang 46 Jun 16 15:30 xx01
-rw-r--r-- 1 daubep cuahang 18 Jun 16 15:30 xx02
$ cat xx00
Thit
Ga
Bo
Heo
Vit
```

III. 5. Các lệnh bảo vệ file và thư mục

III. 5. a. Khái niệm

Một người dùng trong hệ thống UNIX được mô tả bằng các thông tin sau:

- Tên tài khoản người dùng
- Mật mã

- Số nhận dạng (UID: user identify number)
- Số của nhóm (GID: group identify number)
- Chú thích
- Thư mục xâm nhập (home)
- Tên chương trình (shell) cho chạy lúc bắt đầu làm việc

Các thông tin này được chứa trong file `/etc/passwd`.

Một nhóm người sử dụng: một nhóm người dùng là tập hợp của một số người dùng có thể dùng chung các file của nhau, được mô tả bằng những thông tin sau:

- Tên của nhóm
- Mật mã (có thể có hoặc không có)
- Số của nhóm (GID)
- Danh sách những người khách (có thể có hoặc không)

Các thông tin này được lưu trong file `/etc/group`.

III. 5. b. Các lệnh

- *Thay đổi nhóm tuy xuất của thư mục hoặc file:*

Cú pháp: các dạng sau là tương đương cho thư mục (hay cho file)

chgrp group dir	chgrp group file
chgrp GID dir	chgrp GID file

Chỉ có superuser hay người sở hữu mới được quyền thay đổi nhóm người sử dụng. Ví dụ:

```
$ ls -a test
-rwx--x--x 1 bin bin 13023 Jun 21 94 test
$ chgrp data test
$ ls -a test
-rwx--x--x 1 bin data 13023 Jun 21 94 test
```

- *Thay đổi người sở hữu thư mục hoặc file*

```
chown username dir[s]      hay chown UID dir[s]  
chown username file[s]     hay chown UID file[s]
```

Chỉ có superuser hay người sở hữu mới được quyền sử dụng lệnh này. Superuser có thể thao tác trong toàn bộ hệ thống file, còn người sở hữu – chỉ trong thư mục home của mình.

Ví dụ:

```
$ ls -a test  
-rwx--x--x 1 bin data 13023 Jun 21 94 test  
$ chown tuan test  
$ ls -a test  
-rwx--x--x 1 tuan data 13023 Jun 21 94 test
```

- *Đặt quyền truy xuất ngầm định: **umask** [X] XXX (X số bát phân)*

Sau khi đặt **umask** tất cả các file và thư mục tạo sẽ nhận chủ quyền. Giá trị thường dùng như sau:

```
$ umask 177  
$ cat test  
...  
^D  
$ ls -l test  
-rw----- 1 tuan adm 1 Mar 11 10:11 test
```

- *Thay đổi chủ quyền của thư mục hoặc file*

Nếu bạn là người sở hữu hay superuser, bạn có thể sử dụng lệnh **chmod** để thay đổi chủ quyền của file. Có hai cách dùng lệnh này như sau:

☞ Chủ quyền tuyệt đối:

```
chmod [X]XXX dir[s]    hoặc    chmod [X]XXX file[s]
```

Sử dụng các số bát phân, mỗi số đại diện cho chủ quyền của một lớp người sử dụng:

4	cho phép đọc
2	cho phép ghi
1	cho phép thực hiện

Ví dụ:

```
$ ls -l test
-rw-----  1 tuan adm 1 Mar 11 10:11 test
$ chmod 754 test
$ ls -l test
-rwxr-xr--  1 tuan adm 1 Mar 11 10:11 test
```

☞ Dùng ký hiệu tương trưng:

```
$ chmod type operator[permissions] dir[s]
$ chmod type operator[permissions] file[s]
```

Loại (type):

- u	người dùng
- g	nhóm
- o	chung
- a	tất cả

Toán tử (operator):

- +	thêm quyền
- -	bớt quyền
- =	gán giá trị khác

Quyền (permissions) - có thể đặt một hay nhiều quyền đồng thời:

- **r** đọc
- **w** ghi
- **x** thực hiện
- **s** đặt SUID hay SGID, thay vào vị trí quyền thực hiện x
- **t** đặt sticky bit, thay vào vị trí quyền thực hiện x

Ví dụ:

```
$ ls -l test
-rwxr-xr-- 1 tuan adm 1 Mar 11 10:11 test
$ chmod g +w test
$ ls -l test
-rwxrwxr-- 1 tuan adm 1 Mar 11 10:11 test
```

III. 6. Soạn thảo file sử dụng vi

vi là một trong những chương trình soạn thảo văn bản chuẩn trong UNIX. **vi** có hai chế độ:

- Chế độ lệnh: cho phép chèn, xóa, thay thế.
- Chế độ soạn thảo: cho phép soạn thảo văn bản.

Để vào trình soạn thảo **vi** ta đưa vào dòng lệnh: **vi** tên_tập_tin

Khởi đầu **vi** đặt ở chế độ lệnh; để vào chế độ soạn thảo đánh **vi**. Thoát khỏi chế độ này đánh **ESC**. Thoát khỏi **vi** nhấn : **x** hoặc **wq**.

Một số hàm lệnh của vi

```
vi file                      --> bắt đầu dòng 1
vi +n file                  --> bắt đầu ở dòng n
vi +/pattern                --> bắt đầu ở pattern
vi -r file                  --> phục hồi file sau khi hệ thống treo
```

Di chuyển con trỏ

- h** sang trái một khoảng trắng (có thể dùng phím → hoặc BackSpace)
- e** sang phải một khoảng trắng (có thể dùng phím ← hoặc Space bar)

w, W	sang phải một từ
b, B	sang trái một từ
e, E	về cuối từ hiện tại
k, ↑	lên một dòng
j	xuống một dòng
^	về đầu dòng
\$	về cuối dòng
<return>	cũng về cuối dòng
)	cuối câu (dấu kết thúc một câu là dấu ".", "!", " " hoặc "?")
(	đầu câu
}	đầu đoạn văn
{	cuối đoạn văn
^w	đến ký tự đầu tiên chèn vào
^u	cuốn lên 1/2 màn hình
^d	kéo xuống 1/2 màn hình
^z	kéo xuống một màn hình
^b	kéo lên một màn hình
H	về đầu màn hình (Home)
M	về giữa màn hình (Middle)
L	về cuối màn hình (Last)

Chèn văn bản

i	trước dấu con trỏ
I	trước ký tự đầu tiên trên dòng
a	sau dấu con trỏ
A	sau ký tự đầu tiên trên dòng
o	dưới dòng hiện tại
O	trên dòng hiện tại
r	thay thế một ký tự hiện thời
R	thay thế cho đến khi nhấn <ESC>

Nhấn <ESC> để kết thúc chế độ chèn

Lệnh xóa

D	xóa đến cuối dòng
x	xóa một ký tự tại vị trí con trỏ
nx	xóa n ký tự tại vị trí con trỏ
dw	xóa 1 từ tại con trỏ
d0	xóa đến đầu dòng
d\$	xóa cuối dòng
3dw	xóa 3 từ
dd	xóa dòng hiện thời
5dd	xóa 5 dòng kể từ dòng hiện thời
X	xóa 1 ký tự

Khi xóa (dùng **x**, **X** hoặc **d**) có thể phục hồi lại (với điều kiện là chưa xóa lần nữa) bằng cách dùng **p** hoặc **P**.

Lệnh thay thế

cw	thay thế một từ (cú pháp cw <text> <ESC>) bằng <text>. Từ được thay thế tính từ con trỏ đến ký tự \$
3cw	thay thế 3 từ
cc	dòng hiện thời
5cc	thay 5 dòng kể từ dòng hiện thời
R	thay thế số ký tự bằng <text> (cú pháp R <text> <ESC>)
S	thay thế dòng bằng <text> (cú pháp S <text> <ESC>)
rc	thay thế ký tự hiện tại bằng ký tự c

Lệnh tìm kiếm

*/and	từ kế tiếp của and
*?and	từ kết thúc là and
*/nThe	tìm dòng kế bắt đầu bằng The
n	lặp lại lần dò tìm sau cùng

Lệnh tìm kiếm và thay thế

s/text1/text2/g thay text1 bởi text2
l, \$s/file/dir thay file bằng dir từ hàng 1 đến cuối file
g/one/s//1/g thay thế one bằng 1

Sao chép & di chuyển văn bản

l,n co m sao chép các dòng từ thứ 1 đến n vào sau dòng m
l,n m m di chuyển các dòng từ thứ 1 đến n vào sau dòng m
y (yank) Lệnh cho phép sao chép văn bản vào vùng đệm.
p hay **P** Sao chép văn bản từ vùng đệm ra.

Ví dụ:

y3w sao chép 3 từ vào vùng đệm
 Y hoặc yy sao chép dòng hiện tại vào vùng đệm
 5yy sao chép 5 dòng kể từ vị trí con trỏ vào vùng đệm

Mỗi lần thực hiện một lệnh xóa (dùng **x** hoặc **d**) thì **vi** đều ghi lại phần văn bản bị xóa vào vùng đệm riêng cho đến lần xóa sau. Lệnh **p** và **P** cho phép lấy lại văn bản từ vùng đệm đó. Trước khi thực hiện lệnh này, con trỏ phải được đặt vào vị trí cùng kiểu với phần văn bản có trong vùng đệm:

- Ký tự
- Từ
- dòng
- cuối dòng

Sự khác nhau khi dùng các lệnh **p** và **P** chỉ ở chỗ:

P sao chép phần văn bản xóa lần cuối vào *sau* đối tượng cùng kiểu
p sao chép phần văn bản xóa lần cuối vào *trước* đối tượng cùng kiểu

Thoát khỏi chương trình vi

:w ghi file vào bộ nhớ

-
- | | |
|------------|---|
| :x | ghi và thoát (tương tự :wq , :Z) |
| :q | Thoát không ghi nội dung file |
| :q! | dù file có thay đổi thoát vẫn không ghi. |

Khi đang trong trình soạn thảo **vi**, muốn làm việc với shell ở chế độ lệnh, ta đánh: **! sh**. Sau đó muốn trở lại **vi**, đánh: **^D**.

Để hiển thị số thứ tự các dòng soạn thảo thì dùng: **set nu**. Để bỏ hiển thị này dùng: **set nonu**. Chuyển con trỏ đến dòng thứ n dùng: **n <Enter>**

Xem trạng thái văn bản đang soạn thảo dùng **^G** ; lệnh này sẽ hiển thị tên, trạng thái, số dòng vị trí con trỏ và phần trăm văn bản tính từ vị trí con trỏ đến cuối văn bản.

- | | |
|-----------------------|--|
| :! command | ngừng vi lệnh shell được xét để thực thi |
| :!! command vi | lệnh shell đã gọi trước lệnh vi được thực thi |
| :sh | ngừng vi bắt đầu một shell |
| :! csh | ngừng vi , bắt đầu một C-shell mới. |

IV. Xử lý vào/ra

IV. 1. Đối tượng dữ liệu

Các chương trình nhận dữ liệu tạo đầu vào và tạo đầu ra phải có các kênh liên lạc để chuyển các thông tin đó. Đôi khi công việc này được thực hiện tường minh bởi chương trình "mở" một file cụ thể. Ví dụ: giả sử bạn đánh lệnh: **sort tam**, việc này gây ra chương trình **sort** mở file **tam** - như vậy là **sort** có thể đọc file để vào dữ liệu.

UNIX tạo ra một giao diện lập trình như nhau đối với tất cả các loại file, biểu diễn file qua các số nguyên nhỏ, gọi là *file descriptor*, liên hệ với file mỗi khi file được mở và đại diện cho file trong hệ điều hành đối với các yêu cầu vào ra (**read**, **write**, **close**). Các file đều được xử lý như thể chúng là một dãy đơn giản các byte được truy nhập theo chỉ số được gọi là *file pointer*. File pointer có thể được tăng giảm theo từng đơn vị byte, tạo điều kiện cho chương trình chạy một cách độc lập với thiết bị. Tuy nhiên, file pointer có thể được thay đổi bởi các thao tác đọc (**read**) và ghi (**write**) và được đặt giá trị tùy ý bởi lệnh **lseek** đối với các file trên đĩa (không phải thiết bị, v.v..).

Trong thư viện chuẩn C của UNIX có một tập hợp các hàm được thiết kế để đảm bảo tối ưu việc truy nhập file tuần tự dưới dạng các dòng và ký tự, đặc biệt hay dùng với các lệnh **getchar** và **putchar**, được gọi là gói vào/ra chuẩn (standard I/O package). File **vào chuẩn** (**stdin**) bình thường được dùng để người dùng đọc dữ liệu vào ở thiết bị đầu cuối. Tương tự, file **ra chuẩn** (**stdout**) là nơi người dùng đọc được kết quả thực hiện chương trình, cũng như **sai chuẩn** (**stderr**) lưu các thông báo lỗi. Đa số các lệnh UNIX (**cat**, **grep**, **wc**, ...) tự động mở 3 file vào chuẩn, ra chuẩn và sai chuẩn. Trong UNIX thiết bị được xử lý như file, như vậy ba file chuẩn có thể là các thiết bị: vào chuẩn là bàn phím, ra chuẩn và sai chuẩn là màn hình. Dĩ nhiên, bạn có thể sử dụng "đổi hướng" để thay đổi các thiết bị ra vào chuẩn, ví dụ thành các file trên đĩa.

IV. 2. Đường ống và lọc

Đường ống của UNIX liên kết một lệnh với một lệnh khác. Đặc biệt hơn nó làm cho ra chuẩn của một lệnh thành vào chuẩn của một lệnh khác. Ký hiệu đường ống “|” được sử dụng để thiết lập đường ống. Ví dụ: **wc baocao* | sort -n** nhận ra của **wc** (trong trường hợp này là tổng số từ và ký tự của các file có tên bắt đầu là baocao) và gửi nó đến lệnh **sort** để sắp thứ tự số. Kết quả cuối cùng là các tổng số từ sắp theo thứ tự tăng dần hiển thị trên màn hình.

Đường ống có thể kết hợp với đối hướng. Ví dụ: **wc baocao* | sort -n > rep-count** kết quả sẽ đưa ra file **rep-count**.

Các chương trình đọc các dữ liệu ban đầu của nó từ chương trình khác, thực hiện các thao tác nào đó trên vào chuẩn và ghi kết quả của nó tới ra chuẩn được gọi là các **lọc**. Lọc thường được dùng để thay đổi đầu ra.

Hầu hết các lệnh UNIX đều có thể dùng để tạo đường ống. Một vài lệnh thông dụng như **grep**, **sort**, **pg** và **more** có thể là lọc, có thể vừa là bộ phận cấu thành của đường ống, đồng thời cũng hay được dùng như các lệnh riêng rẽ.

IV. 3. Đối hướng vào/ra chuẩn

IV. 3. a. Khái niệm về đối hướng

Mặc dù một vài lệnh như **sort** đã có sẵn chọn lựa để xác định file ra chuẩn và nhiều lệnh cho phép bạn xác định file vào chuẩn trên dòng lệnh, UNIX vẫn cung cấp một cơ cấu linh hoạt, tân tiến hơn và rất mạnh: “đối hướng” - cho phép gán lại các file chuẩn. Chương trình vẫn liên kết đến ba file vào/ra chuẩn nhưng định danh của các file trên có thể bị thay đổi. Ví dụ, bằng lệnh **cat** (sử dụng ra chuẩn) chúng ta có thể sử dụng “>” để tạo ra chuẩn là một file. Câu lệnh **cat tam1 tam2 > baocao** nối các file **tam1** và **tam2** rồi gửi kết quả đến ra chuẩn. Ký hiệu “> baocao” có nghĩa file ra chuẩn sẽ là file **baocao** thay vì màn hình.

Chú ý: đối hướng ra chuẩn sẽ tạo file ra thay thế file ra chuẩn đã tồn tại trước đó.

Đối hướng “vào” cũng giống như “ra”, ngoại trừ ký hiệu “<” được sử dụng để gán lại vào chuẩn. Ví dụ: **sort < tam**, lệnh **sort** sẽ lấy dữ liệu vào trong file **tam** thay vì lấy từ bàn phím.

Vậy sự khác biệt giữa các câu lệnh: **sort tam** và **sort < tam** thể hiện ở đâu ? Trong **sort tam**, lệnh **sort** mở thực sự file **tam**, như vậy nó có thể sử dụng **tam** thay cho vào chuẩn. Như vậy 2 file vào đều được mở: **tam** và vào chuẩn. Với đối hướng, dĩ nhiên file **tam** là vào chuẩn và chỉ một file vào chuẩn đó được mở mà thôi. Hơn nữa, với đối hướng, hệ điều hành mở file **tam** thay vì lệnh **sort** mở.

Đối hướng ra chuẩn không ảnh hưởng đến sai chuẩn, và thông báo sai vẫn gửi ra màn hình. Giả sử file **tam2** không tồn tại, ta đánh lệnh **cat** trong ví dụ sau và thấy trên màn hình:

```
$ cat tam tam2 > baocao
tam2: No such file or directory
```

Chỉ một file có thể sử dụng để đối hướng ra hay vào chuẩn, bởi vì mỗi lệnh chỉ nối với một ra chuẩn và một vào chuẩn.

IV. 3. b. Đa đối hướng

Ta có thể sử dụng cả đối hướng ra và vào chuẩn cho một lệnh. Ví dụ:

```
$ tr "[a-z]" "[A-Z]" < act1 > bigact1
```

lấy dữ liệu đầu vào từ file **act1**, gửi nó vào lệnh **tr** để xử lý và gửi đầu ra đến file **bigact1**. Cụ thể là lệnh **tr** đổi tất ký tự chữ in nhỏ trong file **act1** thành chữ in hoa và ghi vào file **bigact1**.

Thứ tự đưa vào chỉ thị đối hướng là không quan trọng. Ví dụ, lệnh được viết ở dưới đây hoàn toàn tương đương như câu lệnh trên:

```
$ tr "[a-z]" "[A-Z]" > bigact1 < act1
```

IV. 4. Tổ hợp các file với vào chuẩn

Nhiều lệnh UNIX có thể sử dụng vào chuẩn như một danh sách các file nhập. Ta hãy xem xét ba ví dụ sau:

\$ lp tam	lệnh mở file tam
\$ lp < tam	lệnh sử dụng vào chuẩn
\$ cat tam lp	lệnh sử dụng đường ống

Tất cả đều in nội dung file tam, nhưng hoạt động của các câu lệnh được xét trong các ví dụ trên hoàn toàn khác nhau. Lệnh đầu mở tam và bỏ qua vào chuẩn. Lệnh thứ hai gây ra shell mở tam và coi nó là vào chuẩn. Lệnh thứ ba coi ra chuẩn của **cat** là vào chuẩn của lệnh **lp**.

Một vài lệnh UNIX cho phép bạn sử dụng một danh sách các file vào bằng cách sử dụng ký hiệu “-” để thể hiện vào chuẩn như một danh sách các file. Ví dụ:

```
$ sort tam | lp tam1 - tam2
```

lp sẽ in đầu tiên là tam1, kế đó là kết quả sắp thứ tự của file tam, cuối cùng là file tam2.

IV. 5. Các dạng khác của đối hướng

IV. 5. a. Đối hướng thêm vào >>

Kiểu đối hướng này được sử dụng khi ta muốn thêm dữ liệu vào cuối một file đã tồn tại sẵn từ trước. Ví dụ:

```
$ cat baocao2 baocao3 >> baocao
```

thêm nội dung của baocao2 và baocao3 vào cuối của file baocao. Nếu file đích baocao không tồn tại thì nó sẽ được tạo ra.

IV. 5. b. Đối hướng “File” <<

Giả sử có thông báo rằng bạn có E-mail từ người dùng khác; bạn có thể thiết lập một shell script để đọc nội dung thư được gửi tới:

```
$ mail $* < message
```

message là tên file chứa thông báo chuẩn. \$* thể hiện các đối số của shell

Vấn đề với script này là lệnh và văn bản thông báo là hai file riêng rẽ. Dĩ nhiên bạn có thể tránh vấn đề này bởi sử dụng << lấy chỗ của văn bản trong cùng file lệnh. Văn bản như vậy được gọi là “Tài liệu ở đây”. Script mới có dạng:

```
$ mail $* << stop
Xin nho tra tien cuoc phi dien thoai va Internet hang
    thang cho dung han.
Buu dien Ha noi
stop
$ echo $* reminded on `date` >> $HOME/fi.log
```

IV. 5. c. Các bộ mô tả file và đổi hướng sai chuẩn 2>

Đôi khi chúng ta muốn đổi hướng thông báo sai. Ví dụ, khi ta chạy lệnh ở chế độ nền “&” để kiểm tra các lỗi chính tả trong file baocao.

```
$ spell baocao &
```

Lệnh **spell** chạy ở chế độ nền vẫn gửi các thông báo về các từ sai chính tả ra màn hình. Điều này gây rối cho các công việc khác cùng đang được thực hiện. Để tránh tình huống này, bạn phải sử dụng đổi hướng sai chuẩn với bộ mô tả file (0: vào chuẩn, 1: ra chuẩn, 2: sai chuẩn).

```
$ spell baocao > baocaodung 2> baocaosai &
```

Các từ sai sẽ được đưa ra file baocaosai.

IV. 5. d. Tổ hợp ra 2>&1

Đôi khi bạn muốn đổi hướng cả hai: ra chuẩn và sai chuẩn, đến cùng một file. Việc này có thể làm được với tổ hợp 2>&1.

```
$ spell baocao > baocaodung 2>&1 &
```

File baocaodung sẽ chứa cả ra chuẩn và các thông báo sai.

IV. 6. Lệnh tee

Hoạt động đối hướng và đường ống là đặc điểm của hệ điều hành UNIX. Tuy nhiên ta cũng có thể sử dụng một lệnh của UNIX để làm việc này. Đó là lệnh **tee**, nó sẽ giám sát các kết quả gián tiếp của chuỗi đường ống. Ví dụ:

```
$ sort baocao | tee baocaostt | lp
```

Đầu tiên, lệnh **tee** gửi vào chuẩn của nó đến ra chuẩn của nó, trong trường hợp này gửi ra của **sort** đến vào của **lp**. Thứ hai, **tee** lấy chỗ một bản sao của vào chuẩn vào tên file **baocaostt**.

V. Sử dụng các dịch vụ Internet

V.1. Lệnh mail

Nhu cầu trao đổi các bản tin giữa các người dùng là nhu cầu cơ bản mà hệ điều hành nào cũng phải đáp ứng. Lệnh **mail** được đưa vào sử dụng từ rất lâu trong UNIX và có cú pháp cực kỳ đơn giản:

```
$ mail
```

Thư điện tử cung cấp các khả năng cơ bản để gửi và nhận thư. **mail** sẽ hiển thị thư theo thứ tự vào trước ra sau. Sau khi hiển thị mỗi thư **mail** sẽ hiện lên dấu “?” để chờ lệnh kế tiếp của người sử dụng. Sau đây là các lệnh con hay được sử dụng:

newline	Hiển thị thư kế tiếp, nếu không còn thì thoát khỏi mail .
+	Giống như lệnh bắt đầu dòng mới newline
p	In thư
s [<i>file</i>]	Cất (ghi) thư vào file hoặc hộp thư mbox
w [<i>file</i>]	Giống như s nhưng không cất tiêu đề thư
d	Xóa thư
q	Thoát khỏi mail
x	Thoát khỏi mail mà không thay đổi thư
! <i>command</i>	Thực hiện lệnh của UNIX, được chỉ bằng tên lệnh ở vị trí của <i>command</i>

- *Gửi thư:* Đưa vào lệnh **mail** với địa chỉ của người nhận thư. Ví dụ:

```
$ mail tuan@hut.edu.vn
<nội dung thư>
^D (hoặc dấu chấm hết "." - full stop)
```

Thư sẽ được gửi cho người dùng có tên là tuan ở địa chỉ hut.edu.vn.

Có thể cùng một lúc gửi một thư cho nhiều người, lúc đó các địa chỉ phải được viết cách nhau một khoảng trống:

```
$ mail tuan@hut.edu.vn minh@icbv.com.vn
```

Cũng có thể gửi thư bằng cách dùng chuyển hướng đầu vào từ một file văn bản nội dung thư đã được soạn thảo từ trước. Cách này thường được dùng khi nội dung thư có kích thước lớn, hay là các báo cáo được sinh ra định kỳ (ví dụ, các file *.log) và gửi đi tự động.

```
$ mail -s daotao </export/thi/ketqua.log
```

● *Nhận thư*: Khi đăng nhập vào hệ thống, nếu có thư đã được gửi tới, hệ thống sẽ thông báo "You have mail". Khi đó, ta có thể đánh lệnh **mail** để xử lý thư điện tử bằng các lệnh con đã đề cập ở trên.

V.2. Lệnh **mailx**

Cú pháp:

```
$ mailx
```

mailx bao gồm các lệnh để truyền và nhận thư.

Lệnh có dạng cú pháp:

```
Lệnh [danh sách thư] [các đối số] [danh sách thư]
```

Danh sách thư bao gồm một trong số các ký hiệu sau:

n	Thư số n
\$	Thư cuối cùng
*	Tất cả thư
n-m	Từ thư n đến m
:d	Xóa tất cả thư

:n	Thư mới
:r	Thư chỉ đọc
:u	Thư chưa đọc
:o	Thư cũ

Lệnh:

help	Hiển thị cú pháp của lệnh
ignore	Bỏ qua tiêu đề file
print	Hiển thị thư
Print	In thư
top	Hiển thị một vài dòng đầu của thư
respond	Trả lời một thư
copy	Chép thư lên file
list	Hiển thị danh sách lệnh
save	Cất thư lên file
quit	Thoát khỏi mailx

– Thông tin bằng lệnh **write**

Lệnh **write** cho phép gửi thư tức thời tới những người khác đang làm việc trong hệ thống. Cú pháp:

```
$ write username
    < Câu thông báo cần gửi >
    ^D (hay chỉ một dấu chấm câu "." trong dòng
    cuối)
```

Lệnh **wall** cho phép gửi thông báo cho tất cả những người cùng làm việc trong hệ thống.

Lệnh **write** ghi thông báo trực tiếp lên màn hình nên có thể gây nhiễu cho người nhận nếu họ đang làm một công việc khác. Để tránh khỏi bị quấy rối vì những tin nhắn vô bổ, có thể dùng lệnh **mesg** với tùy chọn **-n**; ngược lại, với tùy chọn **-y**, ta sẽ khôi phục lại khả năng nhận và hiển thị các thông báo lên màn hình.

V. 3. Các lệnh làm việc khác với máy tính từ xa

Các giao thức được sử dụng là TCP/IP (Transmission Control Protocol/Internet Protocol) và **uucp** (Unix-to-Unix Copy). Do đó, các lệnh tương ứng trong mỗi phần sau sẽ được trình bày theo thứ tự với TCP/IP, sau đó với **uucp**.

V. 3. a. Đăng nhập từ xa

▪ . Về dựa trên giao thức TCP/IP

Lệnh **rlogin** được dùng để đăng nhập từ xa vào một máy hostname bởi người dùng có tên username theo cú pháp:

```
rlogin [-L] [-8] [-ec] [-l username] hostname
```

với các tùy chọn:

-L

Cho phép phiên làm việc **rlogin** thực hiện trong chế độ “litout”

-8

Dùng chế độ 8 bit dữ liệu thay vì chế độ 7 bit

-ec

Chỉ ra ký tự thoát khác (ở đây là ký tự c) để cắt đường truyền từ máy ở xa.

-l username

Dùng tên người dùng khác với tên người dùng đang đăng nhập hệ thống tại chỗ để đăng nhập vào máy ở xa.

Sau khi đăng nhập, máy tại chỗ sẽ thực hiện chức năng của một thiết bị đầu cuối cho máy từ xa. Tên của máy từ xa - hostname - phải có trong cơ sở dữ liệu các host, ví dụ như /etc/hosts, trong các ánh xạ hosts map của dịch vụ NIS+, trong máy chủ dịch vụ tên vùng Internet.

Trong mỗi một máy từ xa đều có file với tên /etc/hosts.equiv chứa danh sách các máy “được tin tưởng” cùng có chung các người dùng được đăng ký. Nếu cùng dùng một tên người dùng loại đó cho máy tại chỗ và máy từ xa, khi đăng nhập vào máy từ xa ta không cần phải nhập mật khẩu người dùng. Một người dùng có thể

đưa các dòng vào dạng `hostname username` vào trong file `.rhosts` trong thư mục `home` của mình để lập một danh sách tương tự.

▪ *.Lệnh dựa trên giao thức uucp*

Nghĩa đen của lệnh **cu** là “gọi một hệ thống UNIX khác” (call another UNIX system). Tuy nhiên, lệnh này không chỉ “gọi” một máy chủ UNIX, mà còn có thể đăng nhập vào một terminal, thậm chí một máy không chạy trên nền UNIX nữa. Lệnh **cu** thực hiện hai pha công việc. Trong pha thứ nhất, nó thực hiện việc đăng nhập vào máy từ xa. Pha thứ hai là pha đối thoại, với khả năng trao đổi file giữa hai máy và đồng thời thực hiện các lệnh trên máy tại chỗ. Lệnh này có cách viết và dùng tương đối phức tạp, nên trong phần này được biểu diễn một cách đơn giản hóa như sau:

```
cu [-C] telno|systemname [local-cmd]
```

Có thể xem chi tiết hơn các tùy chọn hay dùng trong chương XV “Các giao thức và dịch vụ mạng”, trong mục *Giao thức uucp*, hay tham khảo trang **man** tương ứng. Tùy chọn `-C` được dùng, nếu ta muốn lệnh **cu** thực hiện các lệnh `local-cmd` trên máy tại chỗ. Thay vì tên của máy từ xa `systemname`, có thể chỉ ra số điện thoại kết nối của nó `telno`.

V. 3. b. Trao đổi với người khác trên mạng

Lệnh **talk** cho phép các người dùng có thể trao đổi với nhau qua mạng UNIX, có hiển thị thông tin trên màn hình. Tuy nhiên, lệnh này chỉ có thể hoạt động giữa những máy có cùng cấu trúc hệ thống, có nghĩa là trong cùng một mạng LAN với máy chủ UNIX hay giữa các hệ thống UNIX giống nhau. Cú pháp của lệnh như sau:

```
talk username [ttyname]
```

với `username` - tên người dùng. Nếu nhiều người cùng đăng nhập hệ thống với cùng một tên người dùng (ví dụ, `userB`), mà ta (ví dụ, `userA`) muốn nói chuyện với một người cụ thể, cần phải chỉ ra `ttyname` - tên của terminal (thiết bị đầu cuối) thích hợp mà người đó đang dùng. Cuộc nói chuyện sau khi được thiết lập sẽ

diễn ra đồng thời, cả hai phía vừa có thể theo dõi thông báo của đối phương, vừa có thể gõ ngay câu trả lời.

Giả sử người dùng `userA` trong hệ thống với máy chủ `serverA` bắt đầu trao đổi với người dùng `userB` ngồi tại máy `tty15` của hệ thống `serverB` của cùng một mạng DNS `hut.edu.vn` lúc 5 giờ chiều thứ sáu ngày 30 tháng Sáu năm 2000. Lệnh `talk` thoát đầu được kích hoạt bởi `userA`:

```
talk userB@serverB.hut.edu.vn tty15
```

và gửi đi thông báo tới `userB`:

```
Message from TalkDaemon@serverB at 17:00 Fri 30 Jun  
2000
```

```
talk: connection requested by userA@serverA
```

```
talk: respond with: talk userB@serverB
```

Lúc đó, `userB` đánh vào lệnh sau để chấp nhận nói chuyện:

```
talk userA@serverA.hut.edu.vn
```

Ký tự `Ctrl-L` dùng để làm mới màn hình hiển thị khi người dùng cần xóa các dòng tin đang trao đổi trong hội thoại. Bằng cách dùng `Ctrl-D` có thể kết thúc cuộc nói chuyện, dấu nhắc sẽ trở về trạng thái chờ ở cuối màn hình.

Tất nhiên, có thể dùng lệnh `mesg -n` để ngăn chặn các lời mời nói chuyện vô bổ.

V. 3. c. Sao chép file với máy tính khác từ xa

Để sao chép các file của các máy tính từ xa, người ta có thể dùng các lệnh khác nhau, tùy thuộc vào các giao thức được sử dụng sẵn có trong họ giao thức TCP/IP. Tất nhiên, muốn thực hiện được các thao tác này, bạn phải thỏa mãn các điều kiện về mạng và bảo mật hệ thống như tên và địa chỉ các máy cần liên lạc và tài khoản người dùng đã được đăng ký hay được “tin tưởng” để có đủ các chủ quyền quy định, v.v..

- *. Vấn đề liên giao thức TCP/IP*

Các lệnh hay được dùng là **r**cp và **f**tp. Lệnh **f**tp còn được đề cập tới nhiều lần nên ở đây ta chỉ chú ý tới lệnh **r**cp, một lệnh cho phép ta sao chép file từ máy tại chỗ tới máy từ xa hay ngược lại. Cú pháp của lệnh này là:

```
rcp [option] source-hostname:dir/filename \
      dest-hostname:dir/filename
```

Nếu file nguồn hoặc đích nằm ở thư mục hiện thời của máy tại chỗ, ta chỉ cần gõ tên file là đủ.

Ví dụ:

sao chép file có tên **bangkengay** từ thư mục hiện thời của máy tại chỗ vào thư mục **/export/taphop** của máy **chinhanhtinh**:

```
rcp bangkengay chinhanhtinh:/export/taphop
```

sao chép một file có tên **doichieungay** từ thư mục **/export/lienhang** của máy **chinhanhtinh** vào thư mục hiện thời của máy tại chỗ:

```
rcp chinhanhtinh:/export/lienhang/doichieungay
```

Nếu dùng tùy chọn **-r**, ta có thể sao chép toàn bộ một thư mục, nghĩa là tất cả các thư mục con và file trong thư mục đó. Cần chú ý rằng do có thể sao các thư mục, file không chỉ vào ổ cứng mà còn vào cả băng từ, lệnh **r**cp với tùy chọn này cho phép ta thực hiện công việc sao lưu dữ liệu vào một máy từ xa. Thao tác này được thực hiện định kỳ, tự động một cách hết sức thuận lợi, nếu ta đưa dòng lệnh tương ứng vào trong file **crontab** và nếu đường truyền luôn ổn định.

- *. Vấn đề liên giao thức uucp*

Trong trường hợp này ta có thể sử dụng các lệnh **uucp** hay **uuto**. Các lệnh này cũng được mô tả chi tiết hơn ở chương *Các giao thức và dịch vụ mạng*, trong mục giao thức **uucp**. Cú pháp đơn giản của lệnh **uucp** có dạng:

```
uucp [option[[argument][ option[argument]...]]]
      source-file destination-file
```

với các **option** - tùy chọn và **argument** - thông số có thể bỏ qua được. Các **source-file** - file nguồn và **destination-file** - file đích được viết theo dạng **systemname!dir/filename**. Nếu vị trí nguồn hoặc đích là máy hiện thời, có thể chỉ cần viết tên file và đường dẫn. Nếu dùng C shell, dấu “!” phải đứng sau ký tự “\” (\!), hoặc để trong dấu ngoặc đơn (‘!’). Ta xét một vài ví dụ đơn giản khi muốn sao chép file **baocao** vào các vị trí khác nhau trong máy từ xa có tên **tonghop**:

vào thư mục **/data/solieu**:

```
uucp baocao tonghop\!/data/solieu
```

vào thư mục **home** của người dùng **NVA**:

```
uucp baocao tonghop\! ~ NVA
```

vào thư mục chia sẻ để dùng chung:

```
uucp baocao tonghop'!' ~ uucppublic
```

Lệnh **uuto** có cú pháp:

```
uuto [option[s]] source-files destination
```

với **source-files** - đường dẫn trên máy tại chỗ, **destination** - có dạng **system[!system]!user**, **system** - tên máy từ xa đã được đăng ký trong **uucp** của máy tại chỗ, **user** là tên người dùng đăng nhập vào máy từ xa.

Lệnh **uuto** khác lệnh **uucp** ở chỗ là chỉ được dùng để sao chép file từ máy tại chỗ tới một máy khác ở xa mà không có chiều ngược lại. Thêm nữa, tại máy ở xa, thư mục mặc định chứa các file được sao chép tới là **/var/spool/uucppublic**. Có thể thay đổi thư mục đích này bằng cách gán

giá trị khác cho biến PUBDIR trong mã nguồn của **uucp**. Ví dụ sau là rất đơn giản đối với người đọc:

```
uuto /export/oracle/ctl001.dbf hanoi\!oracle
```

V. 3. d. *Thực hiện lệnh trên máy tính từ xa có kiểm tra kết quả*

Cũng tùy thuộc vào việc sử dụng giao thức TCP/IP hay **uucp** mà ta có thể dùng các lệnh **rsh** hay **uux**. Nói chung, những lệnh này ít được người dùng bình thường sử dụng, mà chủ yếu là công cụ rất hiệu quả cho những người dùng với các nhiệm vụ đặc biệt hay người quản trị.

▪ *. Vấn đề liên giao thức TCP/IP*

Lệnh **rsh** cũng đòi hỏi tên máy từ xa hostname phải được đăng ký trong cơ sở dữ liệu các *host* (các file /etc/hosts.equiv, .rhosts, v.v..) và có cú pháp như sau:

```
rsh [-n] [-l username] hostname command
rsh hostname [-n] [-l username] command
```

Khi được thực hiện trong chế độ đồ họa, lệnh **rsh** mở trên màn hình của máy tại chỗ hai cửa sổ, một cửa sổ vẫn phản ánh các kết quả trả về của các lệnh thực hiện trên máy tại chỗ, cửa sổ thứ hai cho biết kết quả các lệnh thực hiện trên máy từ xa. Do tính đa nhiệm của mình, UNIX cho phép người dùng có thể thực hiện đồng thời các lệnh trên cả hai máy đó. Nếu dùng tùy chọn **-n**, ta có thể đổi hướng đầu vào của **rsh** tới /dev/null để tránh ảnh hưởng qua lại giữa **rsh** và shell dùng kích hoạt **rsh**. Nếu không dùng tùy chọn **-l username**, tên người dùng trên máy từ xa và trên máy tại chỗ là như nhau.

▪ *. Vấn đề liên giao thức uucp*

Với lệnh **uux** người dùng trên máy UNIX tại chỗ thực hiện được các lệnh trên một máy UNIX khác ở xa. Lệnh có cú pháp đơn giản hóa như sau:

```
uux [-] command-string
```

dấu “-” quy định vào chuẩn là chung cho cả lệnh **uux** và **command-string**, **command-string** là dòng lệnh phải thực hiện trên máy ở xa. Ví dụ, lệnh (không phải được dùng ở đây trong C shell):

```
uux  “!diff serverA!/home/ketqua/fileA
      serverB!/home/ketqua/tuan/fileB >
      !~/results/file.diff”
```

lấy các file **fileA** và **fileB** từ các máy **serverA** và **serverB**, thực hiện lệnh so sánh **diff** tại máy tại chỗ đối với các file đó, rồi gửi kết quả vào file **file.diff** trong thư mục công cộng **PUBDIR/results** của máy tại chỗ. Giá trị mặc định của biến **PUBDIR** là **/var/spool/uucppublic** và có thể thay đổi được.

V. 4. Dò tìm thông tin từ Internet, Intranet

V. 4. a. Vị trí của các thông tin cần tìm kiếm

Internet hiện nay đã trở thành một mạng toàn cầu, là một kho thông tin khổng lồ, đa dạng và luôn được cập nhật về tất cả các lĩnh vực. Xét về bản chất, các hệ thống UNIX và Internet có liên quan chặt chẽ nhất. Mạng ARPANET, khởi nguồn của Internet hiện nay, được xây dựng và có cốt lõi từ các máy tính với hệ điều hành UNIX. Cho đến nay, dù đã có ngày càng phổ biến các máy tính chạy trên MS Windows (Windows NT, Windows 2000), số máy chủ trên nền UNIX vẫn chiếm một tỷ lệ lớn và chắc chắn. Mạng Intranet là một mô hình thu nhỏ của Internet, chỉ hoạt động trong một phạm vi một tổ chức, một cơ quan, v.v.. Điểm chung dễ thấy của cả hai loại mạng là chúng đều được xây dựng theo các cách thức giống nhau, dùng các công cụ như nhau và thông tin được lưu giữ trong các hệ thống file của các máy tính gọi là các *hosts* hay *sites*, có tên gọi được đặt theo cấu trúc dịch vụ tên vùng. Các site này có các địa chỉ IP được quy định và được chia theo hai loại: i, *World Wide Web site* có các trang Web và tên được bắt đầu với các chữ đầu “www.”; ii, *File Transfer Protocol site* dạng *ftp.têncôngty.tên tổ chức* nhằm cung

cấp khả năng tìm kiếm và trao đổi file dựa trên dịch vụ ftp; *iii*, *Gopher site* chứa các cơ sở dữ liệu và tài liệu chỉ dẫn của toàn thế giới, cũng bắt đầu tên bằng chữ "gopher.". Hiện nay, các thao tác và dịch vụ truy cập thông tin chủ yếu được thực hiện với các Web site và ftp site.

Do mạng Internet là một tổ chức tự nguyên, với quy mô phát triển nhanh chóng và thành phần tham gia rất phong phú về nội dung, chủng loại, mức độ trang bị kỹ thuật v.v., nên lượng thông tin và địa chỉ trở nên lớn đến nỗi có cảm giác đôi khi không thể kiểm soát nổi. Công việc dò tìm địa chỉ và truy cập thông tin lúc đó trở nên khó khăn vất vả, thường xuyên phải lo đối phó với sự nghẽn mạch thông tin. Vì vậy, một mạng thông tin mới tốc độ cao, hiện đại với tên gọi ban đầu Internet2 đang được manh nha trong phạm vi các viện nghiên cứu, các trường đại học và các công ty tin học Mỹ và Tây Âu, Nhật.

Ngoài ra cũng còn phải kể đến một mạng khác - mạng USENET - được lập nên bởi rất nhiều người dùng và máy tính để phổ biến Netnews. Netnews truyền đi tất cả mọi loại thông tin "thượng vàng hạ cám" trên đời, mọi thảo luận, ý kiến, sự việc v.v.. trong hàng trăm nhóm thảo luận và nhóm tin tức khác nhau. Các nhóm tin tức đó có tên được bắt đầu bằng chữ "a.". Một số máy tính là một phần của Internet sử dụng USENET, số khác lại không, do đó, mạng USENET có thể có một phần cũng thuộc mạng Internet, phần khác lại được thực hiện trên các mạng riêng, ví dụ, dành cho giáo dục hay cho mục đích chuyên nghiệp nào đó. Họ giao thức TCP/IP bây giờ cũng được gộp vào trong giao thức mới có tên Network News Transfer Protocol (NNTP).

V. 4. b. Các công cụ tìm kiếm thông tin

▪ *Trình duyệt Internet*

Trình duyệt được dùng phổ biến hiện nay là Netscape, là cải tiến cơ bản của chương trình Mosaic. Thực ra Netscape là tên gọi tắt, tên gọi đầy đủ là Netscape Navigator, nằm trong cả bộ Netscape Communicator. Netscape hiện nay là sản phẩm dùng được cho nhiều hệ điều hành, miễn phí và có mã nguồn cung cấp tự do để phát triển, phiên bản mới nhất dùng cho UNIX cho đến cuối năm 2001 là 6.2.

Trang chủ mặc định là Netcenter - trang chủ của Netscape - cho đến khi chưa được thay bằng trang do mạng hoặc nhà cung cấp dịch vụ Internet tại chỗ xác định.

Ta có thể quay ngược lại các thao tác dò tìm bằng cách nhấn vào nút *Back*, hoặc *Forward* để đi tiếp theo các bước đã thực hiện. Nếu cần cập nhật lại trang, phải dùng nút *Reload*. Để xem được nhiều trang cùng một lúc, từ thực đơn *File*, chọn *New*, sau đó *Navigator Window*. Lúc đầu trang chủ sẽ được hiện lên, nhưng ta có thể sử dụng cửa sổ mới cho trang Web khác. Để chuyển tới một trang Web khác, ta thực hiện các bước sau:

- Trong Windows hoặc UNIX, mở thực đơn *File*, chọn nút *Open Page*. Sau đó nhấn nút *Navigator*.

Trong MacOS, mở thực đơn *File*, chọn nút *Open*. Sau đó nhấn nút *Location* trong *Navigator*.

- Gõ vào địa chỉ URL rồi nhấn *Open*. Trong trường hợp không biết rõ địa chỉ URL, có thể chỉ gõ vào một từ hay một phần của URL, *Navigator* sẽ tìm và hiện trang với lựa chọn các liên kết có liên quan tới từ đó.

Các trang Web cần thiết luôn được người dùng giữ lại bằng cách tạo một tên liên kết, hay còn gọi là *Bookmark*. Cách tạo *Bookmark* như sau:

- Chuyển tới trang Web định đánh dấu.
- Nhấn nút *Bookmarks*.
- Chọn nút *Add Bookmark*.

Các *Bookmark* này có thể được thêm (*Add*), xóa (*Delete*), sửa đổi (*Edit*) hoặc tổ chức chúng theo thư mục.

Các trang Web và các file tải xuống từ Internet có thể được lưu giữ lại, được in ra v.v.. theo cách thức thông thường như trong Windows. Sau đây là những điểm cần chú ý và các kỹ năng khác để sử dụng Netscape Navigator. Trước hết là việc xử lý các loại file khác nhau. Netscape có bộ trợ giúp, trong trường hợp bình thường giải phóng cho người dùng khỏi công việc định dạng và cấu hình trước cho các loại file. Tuy nhiên người dùng có kinh nghiệm có thể định dạng chi tiết hơn loại file theo trình tự sau:

- Từ thực đơn *Edit* chọn *Preferences*.

- Chọn *Navigator* rồi *Applications*.

- Để thêm loại file mới phải chọn *New*. Sau đó đưa vào mô tả cho loại file, ví dụ "movies" cho phim, "sound" cho âm thanh... Đánh vào phần đuôi của file, nhớ là phải điền hết các ký tự, ví dụ: file văn bản - txt, tài liệu siêu văn bản - htm hay html. Đối với loại file MIME, phần đuôi file gồm hai phần nhỏ, được ngăn cách bằng dấu "/", ví dụ loại file ảnh TIFF có đuôi là "image/tiff". Nếu không biết rõ loại file MIME, tốt nhất ta để trống, dành cho máy tự xác định. Sau đó nhấn vào *Applications*, rồi vào *Choose*, cuối cùng là *OK*.

Các file được tải về có thể được lưu trên đĩa cứng, hay được mở ra. Có thể chỉ định thư mục lưu giữ khác với thư mục tạm thời mặc định.

Nhiều tổ chức thực hiện các biện pháp ngăn chặn người ngoài truy cập trái phép mạng máy tính của mình, ví dụ như dùng phần mềm FireWall. Lúc đó, các kết nối từ trong ra ngoài mạng hoặc ngược lại đều phải thực hiện qua một máy chủ gọi là *proxy server*. Trước khi muốn dùng Netscape, trong trường hợp có proxy server, ta phải yêu cầu người quản trị mạng cho biết tên và địa chỉ cổng của proxy server, nếu trong máy của ta không có file cấu hình cho proxy server. Để đặt cho Netscape Communicator làm việc với proxy server, ta thực hiện các bước sau:

- Từ thực đơn *Edit*, chọn *Preferences*.

- Chọn *Advanced*, sau đó - *Proxies*.

- Chọn một trong ba khả năng sau: "*Direct Connection to the Internet*" - nếu không phải đi qua proxy server; "*Automatic proxy configuration*" - nếu đã có file cấu hình proxy server, lúc này cần đưa URL vào và nhấn nút *Reload*; "*Manual proxy configuration*" - để cấu hình bằng tay: nhấn nút *View*, sau đó đưa vào các thiết lập để dùng proxy server và các chi tiết loại trừ. Máy tính của bạn có thể được nối với nhiều máy chủ nên cần phải chỉ rõ máy chủ nào chạy phần mềm proxy server. Dưới mục "*Address of proxy server to use*" đưa vào tên hoặc địa chỉ IP của proxy server cho mỗi loại máy chủ HTTP, ftp hay Gopher. Dưới mục "*Port*" đưa vào địa chỉ cổng cho mỗi proxy server. Dưới mục "*Exceptions*" đưa vào tên của các domain (mạng

vùng) có thể kết nối trực tiếp, không qua proxy server. Nhớ rằng tên viết tắt “*” không dùng được ở đây.

▪ *Lệnh ftp và người dùng anonymous*

Việc truy cập vào các ftp server được thực hiện giống như khi bạn đăng nhập vào một máy chủ UNIX từ xa. Điểm khác biệt nhỏ giữa chúng là ftp server có thư mục công cộng chứa dữ liệu với nội dung và kiểu dạng rất phong phú, cho phép ta có thể tải về hoặc nạp lên một cách tự do. Để làm được việc đó, ta phải đăng nhập vào dưới tên người dùng vô danh **anonymous**. Thông thường các ftp server đòi hỏi mật khẩu cho anonymous chỉ với mục đích thống kê số khách hàng truy cập mà thôi, nên mật khẩu thường chính là địa chỉ E-mail của bạn. Các ftp server thường có các server giống hệt, gọi là *mirror server*, được phân bố trên toàn cầu theo vị trí địa lý để cải thiện khả năng và tốc độ truyền thông.

Ftp site hiện nay có thể cung cấp giao diện đồ họa thân thiện với người dùng. Tuy nhiên, hoạt động của ftp server vẫn chủ yếu dựa trên lệnh **ftp**, và trên nhiều site, bạn vẫn phải tự thực hiện lệnh này. Lệnh **ftp** có cú pháp như sau:

```
ftp [options] [hostname | IP Address]
```

Các tùy chọn có thể được dùng đồng thời:

-d

Có gỡ rối

-g

Không hiện tên file “globbing” (file có tên với tính chất toàn cầu)

-i

Tắt dấu nhắc giao tiếp khi trao đổi nhiều file

-n

Không thực hiện kết nối tự động “auto-login”. Với auto-login lệnh **ftp** kiểm tra file **.netrc** trong thư mục home của người dùng để lấy dòng vào mô tả tài khoản người dùng cho máy từ xa. Trong trường hợp dùng tùy chọn này, sẽ hiện dấu nhắc đưa vào tên và mật khẩu của người dùng để kết nối.

-t

Kích hoạt đầu theo các gói thông tin

-v

Hiển thị mọi trả lời từ máy chủ từ xa, cũng như thống kê truyền nhận file. Là mặc định nếu lệnh **ftp** thực hiện ở chế độ có giao tiếp.

Sau khi đã đăng nhập được vào ftp server, người dùng có thể thực hiện rất nhiều thao tác được quy định bởi các lệnh con của **ftp**. Các lệnh con này có thể tham khảo chi tiết trong trang man **ftp**, ở đây chỉ có các lệnh thường dùng nhất được liệt kê trong bảng sau:

! [command]	Thực hiện lệnh command trên máy tại chỗ. Nếu không chỉ rõ command, shell có giao tiếp sẽ được gọi.
\$ macro-name [args]	Chạy macro được xác định bằng lệnh macdef
account [passwd]	Cung cấp thêm mật khẩu để truy cập thông tin sau khi đã đăng nhập vào máy từ xa.
append local-file [remote-file]	Thêm file trên máy tại chỗ vào sau file trên máy từ xa
ascii	Mặc định. Dùng mã ASCII
bell	Tiếng chuông sau mỗi lệnh truyền file hoàn thành.
binary	Dùng mã nhị phân
bye close disconnect quit	Kết thúc phiên ftp và ra khỏi máy chủ từ xa.
cd remote-dir	Chuyển thư mục trên máy từ xa.
lcd !cd	Chuyển thư mục trên máy tại chỗ. Nếu không chỉ thư mục đích, sẽ về thư mục home của người dùng.
cdup	Chuyển thư mục làm việc trên máy từ xa lên thư mục "cha" của nó.
delete remote-file	Xóa file trên máy từ xa.
mdelete remote-files	Xóa nhiều file trên máy từ xa.

debug	Gỡ rối. Hiện mỗi lệnh được gửi tới máy từ xa, với dấu " ->" đứng trước.
dir [remote-dir] local-file	Hiển thị liệt kê nội dung thư mục trong thư mục remote-dir, và nếu chỉ ra, đưa kết quả ra. Nếu không có remote-dir, dùng thư mục làm việc hiện thời trên máy từ xa; nếu không có local-file - đưa ra terminal.
mdir remote-files local-file	Giống lệnh dir .
get remote-file [local-file] rcv remote-file [local-file]	Lấy file remote-file về với tên local-file.
mget remote-files	Lấy nhiều file remote-file về
help [command] ?	Trợ giúp về command. Nếu không chỉ rõ command, sẽ hiển thị danh sách tất cả các lệnh con
ls [remote-dir -al] [local-file]	Hiển thị liệt kê nội dung thư mục trên máy từ xa.
mls remote-files local-file	Giống như lệnh ls .
macrodef macro-name	Xác định macro. Giới hạn là 16 macro và 4096 ký tự cho tất cả các macro. Macro vẫn được xác định tiếp cho đến khi lệnh close được thực hiện.
mkdir dir-name	Tạo thư mục trên máy chủ từ xa.
rmdir dir-name	Xóa thư mục trên máy chủ từ xa.
put local-file [remote-file] send local-file [remote-file]	Nạp file local-file từ máy tại chỗ vào máy từ xa với tên remote-file, nếu được chỉ ra.
mput local-files	Nạp nhiều file local-file từ máy tại chỗ vào máy từ xa.
open host [port]	Thiết lập kết nối với ftp server

proxy ftp-command	Chạy lệnh con của ftp trên kết nối thứ cấp.
pwd	Hiển thị thư mục làm việc hiện thời của máy chủ từ xa.
rename from to	Đổi tên file trên máy chủ từ xa.
status	Hiển thị trạng thái của lệnh ftp.
type type-name	Mặc định là ascii , còn có binary , image , tenex .
user user-name [passwd] [account]	Tự định danh người dùng đối với máy chủ từ xa.
verbose	Chế độ giao tiếp, hiển thị tất cả các thông báo, thống kê.

▪ *Lệnh gopher*

Là chương trình để nhóm các thông tin trong các site ftp lại với nhau, giúp cho việc truy nhập các site trên một cách hợp lý. Hiện nay ít được dùng hơn các địa chỉ kiểu **http://** và **ftp://**.

▪ *Lệnh telnet*

Lệnh **telnet** dùng giao thức TELNET để biến máy tại chỗ thành một terminal của máy chủ từ xa theo cú pháp:

```
telnet [[host] { IP Address [port]]
```

Đây cũng là một lệnh rất thường dùng. Lệnh này biến máy tại chỗ của người dùng thành một thiết bị đầu cuối, để người dùng có thể đăng nhập vào máy từ xa theo tài khoản đã đăng ký. Chi tiết về lệnh này có thể tham khảo trong trang **man telnet**.

▪ *Lệnharchie*

Là lệnh rất hiệu quả cho phép xâm nhập vào các cơ sở dữ liệu chứa hàng trăm các anonymous ftp site trên thế giới.

- *Lệnh rn*

Lệnh **rn** hay **nn** hay **trn** dùng để đọc và đóng góp thông tin cho một nhóm tin trong mạng USENET. Lệnh này không có mặt trong Solaris, nhưng lại hay được dùng trong Linux.

- *Lệnh tin*

Giống như lệnh **rn**.

Trong phần trình bày về UNIX với mức độ cho người dùng cuối, chúng ta mới đề cập tới E-mail (Thư tin điện tử) từ khía cạnh người dùng, tức là cách sử dụng các lệnh **mail** và **mailx**. Việc tổ chức hệ thống thư tin điện tử - theo cách tiếp cận vấn đề của người quản trị hệ thống và quản trị mạng, là hết sức phức tạp. Do vậy, nội dung hết sức giới hạn sẽ được trình bày dưới đây dưới góc độ này.

V. 5. Khái niệm chung về **sendmail**

Chương trình **sendmail** không chỉ là giao diện với người dùng và cũng chẳng chỉ là bộ phận phát thư điện tử đơn thuần. Nó được thiết kế để gom các thư từ các chương trình như **mail**, sửa đổi thư theo yêu cầu của người gửi, phân đường, gọi những người nhận thích hợp để phân phát thư, hoặc xếp vào các hàng đợi thích hợp để chờ các cuộc chuyển phát qua mạng. **sendmail** tạo khả năng cho phép các nhóm người dùng có danh sách địa chỉ riêng của mình, cũng như từng người dùng có thể tự xử lý luồng thư đến. Do ở đầu nhận có thể là một máy chủ nên **sendmail** hoàn toàn có thể dùng như một cổng Internet thư điện tử.

sendmail dùng ba loại giao diện để giao tiếp với thế giới bên ngoài, kể cả việc gửi và nhận thư:

- Vectơ thông số/Trạng thái trả về: Vectơ thông số (tên lệnh và thông số) là phương pháp của UNIX chuẩn để giao tiếp với tiến trình. Danh sách người nhận được gửi đi trong vectơ thông số, còn văn bản của thông báo được gửi ra vào chuẩn. Những gì mà nơi nhận đọc được sẽ bị gửi trả về nếu có trục trặc. Trạng thái (giá trị) trả về được gom lại sau khi gửi thông báo đi, hay chẩn đoán được in ra nếu cần.

- SMTP (Simple Mail Transfer Protocol) qua đường ống: Giao thức SMTP
- SMTP qua kết nối TCP

Khi người gửi muốn gửi thư đi, yêu cầu theo một trong ba phương pháp trên được đưa tới **sendmail**. Các bước hoạt động của **sendmail** diễn ra như sau:

- *Xử lý thông số và tìm địa chỉ:* Các tên người nhận được duyệt qua lần lượt và tạo thành danh sách người nhận. Các bí danh, cũng như cú pháp, địa chỉ sẽ được kiểm tra, nhưng tên cụ thể các máy hãy còn được để lại.
- *Gom thư:* Các thư phải có tiêu đề ở trên đầu. Chưa thực hiện tiến trình định dạng nào, nhưng yêu cầu nội dung thư phải là các dòng văn bản. Nói cách khác, không chấp nhận dữ liệu nhị phân. Các tiêu đề được đưa vào bộ nhớ, còn nội dung thư được xếp vào file tạm thời. Để đơn giản hóa tiến trình, các thư có tên sai cũng được gom, nhưng về sau bị trả lại với thông báo lỗi.
- *Phân phát thư:* **sendmail** gọi một bộ gửi thư thích hợp cho mỗi một máy có tên trong danh sách người nhận. Bộ gửi thư này gửi yêu cầu nhận thư tới tất cả người dùng trong máy, nhưng nếu chỉ chấp nhận một người nhận trong một thời điểm mới được coi là hoạt động đúng. Thư được gửi theo một trong ba phương pháp trên, các bản sao được bảo quản với các tiêu đề đặt ra. Giá trị trả về của bộ gửi thư được bắt và kiểm tra, nếu mà được xác nhận bởi chuẩn của hệ thống thì tiến trình kết thúc thành công, bằng không, thông báo chung "Service unavailable" được đưa ra.
- *Hàng đợi chờ chuyển lại:* Nếu mà trả về cho thấy còn khá thì, **sendmail** sẽ cho thư vào hàng đợi và sẽ yêu cầu lặp lại tiến trình.

V. 6. Các thư mục và file của **sendmail**

❖ /usr/lib/sendmail

Đây là chương trình thực sự để phân đường cho thư điện tử, nhận các thư từ các chương trình giao diện người dùng và dùng thông tin trong những file cấu hình để gửi thư điện tử, là mã nhị phân của **sendmail**. Chủ quyền mặc định là SUID.

Lệnh liên quan có cú pháp như sau:

```
/usr/lib/sendmail [-ba] [-bd] [-bi] [-bm] [-bp] [-bs]
                  [-bt] [-bv] [-bz] [-C file] [-d X] [-F fullname]
```

```
[-f name] [-h N] [-M id] [-n] [-o xvalue] [-q [time]]  
[-r name] [-t] [-v] [address ...]
```

Với định nghĩa của vài thông số quan trọng như sau:

ba

Chế độ APARNET.

bd

Chạy như daemon, chờ tín hiệu từ kết nối SMTP.

bi

Khởi tạo cơ sở dữ liệu các bí danh.

bm

Phân phát thư theo mặc định.

bp

In tổng kết hàng chờ thư điện tử.

bt

Chạy trong chế độ kiểm tra địa chỉ. Dùng để gỡ rối các bảng cấu hình.

bv

Chỉ kiểm tra tên, không gom hay phân phát thư. Dùng để kiểm tra danh sách địa chỉ.

C *file*

Dùng file cấu hình khác.

d *X*

Đặt giá trị gỡ rối là X.

F *fullname*

Đặt tên đầy đủ của người gửi.

f *name*

Đặt tên của người gửi sau "from:". Chỉ dùng cho những người dùng có tên trong file cấu hình.

h *N*

Đặt giá trị tăng quãng thời gian lặp lại. Nếu đến giới hạn mail sẽ gửi thông báo lỗi.

M *id*

Phân phát thông báo có định danh ID trong hàng chờ.

n

Không dùng bí danh.

o *xvalue*

Đặt tùy chọn xử lý PROCESSING OPTION.

q [*time*]

Quãng thời gian lặp lại (s - giây; m - phút; h - giờ; d - ngày, w - tuần).

t

Đọc thư cho người nhận. Sẽ quét các trường **To:**, **Cc:**, **Bcc:**.

Tùy chọn xử lý PROCESSING OPTION:

A *file*

Dùng file bí danh khác.

c

Vào hàng chờ, chế độ tiết kiệm chi phí.

d *x*

Chế độ phân phát thư (x là i - hồi đáp, b - nền, q - vào hàng đợi lần sau).

D

Chạy lệnh **newaliases** để tự động dựng lại cơ sở dữ liệu bí danh.

e *x*

Đặt chế độ lỗi (x là m - gửi trả lại; w - viết lại thông báo lỗi; p - in thông báo lỗi tại thiết bị đầu cuối - mặc định; q - bỏ tất cả thông báo lỗi).

F *mode*

Dùng khi tạo file tạm thời.

g *N*

Định danh nhóm mặc định để sử dụng mỗi khi gọi các bộ gửi thư.

i

Không coi các dấu chấm ".", " như ký hiệu kết thúc thư.

m

Tự gửi thư cho mình.

Q *queuedir*

Chọn thư mục cho hàng chờ thư.

r *timeout*

Thời gian đọc. Nếu không đặt, **sendmail** sẽ chờ ở chế độ thường trực.

T[time] | [n=time[, u | l=time] ...]

Đặt thời gian tồn tại của thư trong hàng chờ. Không dùng với dòng lệnh mà chỉ đặt trong `/etc/mail/sendmail.cf`.

u

Định danh người dùng mặc định để dùng **mail**.

❖ **/usr/lib/sendmail.mx**

File **sendmail** nhị phân liên kết với bộ giải, dùng với dịch vụ tên vùng.

❖ **/etc/sendmail.cf**

Là cấu hình chính nhằm tạo môi trường và điều kiện hoạt động cho **sendmail**, được điều chỉnh cụ thể cho từng máy. Sau đây là ba nhánh nhỏ:

● **/usr/lib/sendmail.main.cf**

File cấu hình mặc định cho máy chính, ở dạng văn bản. Các dòng vào chứa những mã sau:

- Dw : Xác định tên máy tại chỗ
- DD : Xác định tên vùng tại chỗ
- Cw : Xác định các tên khác được dùng bởi máy tại chỗ
- Cd : Xác định các tên khác được dùng bởi vùng tại chỗ
- CS : Liệt kê các máy có trao đổi thư trực tiếp
- DF : Tên máy *forwarder* sẽ xử lý thư nếu hệ thống tại chỗ không biết phân phát thư thế nào
- CF : Các tên khác của máy *forwarder*
- DR : Tên máy *relay* sẽ nhận thư sau khi thư về tới vùng tại chỗ
- CV : Tên máy ngoài (xác định bởi máy *forwarder*)
- DW : Tên máy *uucp* của máy tại chỗ (nếu dùng **uucp**)
- DM : Phương pháp gửi thư điện tử (ví dụ: **uucp**, **tcp**, v.v..)
- OA : Vị trí các file định nghĩa bí danh (thường ở `/etc/aliases`)
- OQ : Vị trí thư mục hàng chờ thư (thường ở `/var/spool/mqueue`)

● **/usr/lib/sendmail.subsidiary.cf**

Là các file mặc định cho máy nhánh, ở dạng văn bản.

● **/usr/lib/sendmail.fc**

File cấu hình giữ cứng.

❖ /var/spool/mqueue

Thư mục này chứa các hàng chờ thư. Sở hữu và chủ quyền mặc định là root và 755, nếu không, ta phải đặt 777. Các thư điện tử được chia vào hai loại file: **qf*** - chứa thông tin điều khiển; **df*** - chứa nội dung thư. Các loại file khác: **lf*** - file khóa; **tf*** - file tạm thời của **qf***, dùng khi tạo lại hàng chờ; **nf*** - dùng khi tạo cùng định danh; **xf*** - mã truyền phiên hiện thời.

Thư nhận về được chứa trong các thư mục hàng chờ /var/spool/mqueue, /var/spool/mail/login_name.

❖ /etc/aliases*

❖ /etc/aliases

Hai loại file bí danh này đều là dạng văn bản.

❖ /etc/aliases.{pag,dir}

Dạng dmb(3).

❖ /usr/ucb/newaliases

Dùng để cập nhật cơ sở dữ liệu bí danh, tương tự như chạy **sendmail -bi**. Có thể để chạy tự động bằng cách liên kết với **/usr/lib/sendmail**.

❖ /etc/rc.local

Daemon **sendmail** cần được bắt đầu chạy khi hệ thống khởi động để thực hiện hai chức năng. Đó là soát lại kết nối qua socket SMTP để đón thư mới và xử lý định kỳ hàng chờ. Trong file /etc/rc.local, vì vậy cần có các dòng sau:

```
if [ -f /etc/sendmail -a -f usr/lib/sendmail.cf ]
then
    (cd /var/spool/mqueue; rm -f nf* lf*)
    /usr/lib/sendmail -bd -q<time> & echo -n 'sendmail'
    > /dev/console
fi
```

Chú ý: cờ **bd** : soát socket SMTP để đón thư mới; **q<time>** : chu kỳ xử lý hàng đợi, ví dụ: 1h - hàng giờ, 15m - cứ 15 phút một lần.

❖ /usr/lib/sendmail.hf

File trợ giúp dùng cho lệnh SMTP HELP, không phải cho người quản lý hệ thống.

❖ /var/spool/mail

Thư mục hộp thư điện tử mailbox, có thể mount được bằng NFS từ máy chủ.

V.7. Cấu hình sendmail cho hệ thống Solaris

Cấu hình sendmail chưa bao giờ được coi là một trong những công việc đơn giản, đặc biệt là trình tự cấu hình chuẩn như trước đây đã không còn đủ đối với tình trạng hỗn loạn hiện nay, khi mà mỗi một người dùng bằng đủ mọi cách đều cố hết sức nhận cho nhanh, cho nhiều thư của riêng mình. Hiện nay, sendmail mới được phát hành với phiên bản 8.9.1, nhưng trước hết, dưới đây ta hãy theo dõi trình tự cấu hình sendmail version 8.8.8 cho Solaris 2.6. Tất nhiên, các hướng dẫn này hoàn toàn có thể tổng quát hóa cho các phiên bản 8.8.x và muộn hơn, cũng như cho các phiên bản Solaris khác nhau. Ở đây, ta phải chú ý rằng dòng lệnh:

```
ln -s /etc/mail/sendmail sendmail
```

chỉ cần thiết khi cấu hình sendmail lần đầu và là không cần thiết khi ta cài nâng cấp phiên bản mới.

```
cd /usr/local/src
```

```
# get the tar.gz file from sendmail.org
```

```
# Dòng sau cho bạn dùng gnu tar có chấp nhận khóa z
```

```
tar xzf installed/sendmail.8.8.8.tar.gz
```

```
cd sendmail-8.8.8/
```

```
cd src/Makefiles
```

```
# File Makefiles mặc định không có quyền ghi, ta thêm quyền  
chmod +w *
```

```
vi Makefile.SunOS.5.6 (bỏ giá trị -l44bsd khỏi LIBS=)
```

```
cd ..
```

```
# Nếu không có đường dẫn tới groff sẽ không mở được man
sh makesendmail (bỏ qua cảnh báo)
cd obj.SunOS.5.6.sun4/
rm *.h *.c *.o

# thực hiện với quyền root
mkdir /usr/local/etc/sendmail-8.8.8
cp * /usr/local/etc/sendmail-8.8.8
cd /usr/local/etc/sendmail-8.8.8
chmod u+s sendmail
vi sendmail.cw

# (thêm tên vùng (domain names) của máy chủ)
cp /etc/aliases .

# không cần thực hiện với quyền root
cd /usr/local/src/sendmail-8.8.8/cf/cf
m4 ../m4/cf.m4 generic-solaris2.mc > sendmail.cf
vi sendmail.cf

# (thêm nội dung hai file dnscheck.txt và
# relayblock.txt ở cuối file sendmail.cf)
# lại thực hiện với quyền root
cp sendmail.cf /usr/local/etc/sendmail-8.8.8
touch /usr/local/etc/sendmail-8.8.8/LocalIP
touch /usr/local/etc/sendmail-8.8.8/relayto
cd /usr/local/etc
ln -s sendmail-8.8.8 sendmail
cd /etc
mv mail mail.sun
ln -s /usr/local/etc/sendmail mail
ln -s mail/sendmail.cf sendmail.cf
cd /usr/lib
mv sendmail sendmail.sun
ln -s /etc/mail/sendmail sendmail
/etc/init.d/sendmail stop
/etc/init.d/sendmail start
```

newaliases

Khi cấu hình file `sendmail.cf` có thể bạn muốn bỏ dấu chú thích (dấu “#”) ở dòng tùy chọn `MeToo` để cho phép mọi người có thể tự gửi một bản sao thư cho mình nhằm mục đích lưu trữ hồ sơ.

Nếu bạn chạy **majordomo** (chương trình lấy từ Linux sang) và muốn tránh các cảnh báo kiểm tra xác nhận người dùng trong X Windows, thì bạn phải thêm `majord` vào danh sách những người dùng được tin tưởng (trusted user - xem lại khái niệm này khi trình bày file `hosts.equiv`) trong `sendmail.cf`:

```
#####
#   Trusted users   #
#####

# this is equivalent to setting class "t"
# Ft/etc/sendmail.ct
Troot
Tdaemon
Tuucp
Tmajord
```

Sau đây là nội dung của file `dnscheck.txt`:

```
#
# Anti-bogus mail check
#

Scheck_mail

# check for valid domain name (incompatible with
DeliveryMode=defer)
RS*           $: $>3 $1    make domain canonical
R $* < @ $+ . >  $:         tag resolved names
```



```
R S* <@ $+ >      $#error $: 451 Domain must resolve
```

và file `relayblock.txt` được khuyến nghị đưa thêm vào trong file cấu hình `sendmail.cf`:

```
#
# Anti-relay system
#

F{LocalIP} /etc/mail/LocalIP
F{RelayTo} /etc/mail/relayto

Scheck_rcpt
# make sure you have TABs here, not BLANKs! sendmail
  will
# complain otherwise...
# first: get client addr
R$+      $: $(dequote " " ${client_addr} $) $| $1
R0 $| $* $@ ok          no client addr: directly
  invoked
R$={LocalIP}$* $| $* $@ ok    from here
# not local, check rcpt
R$* $| $*      $: $>3 $2
# remove local part, maybe repeatedly
R$+      $:$>remove_local $1
# still something left?
R$*$*      $#error $@ 5.7.1 $: 550 we do not relay

Sremove_local
# remove RelayTo part (maybe repeatedly)
R$*$*      $>3 $1 $4
R$*$*      $: $>remove_local $>3 $1 $3
R$*$*      $@ $1$3
```

```
# dequote local part
R$-      $: $>3 $(dequote $1 $)
R$*$*    $: $>remove_local $1$3
```

Thông tin về `dnscheck` có thể tìm thấy trên www.sendmail.org ở trang <http://www.sendmail.org/antispam.html> dưới đầu đề "Insist On Valid Host Name In MAIL FROM: Command". Phương pháp chặn relay trình bày ở đây rất phức tạp. Một ví dụ dùng trong file `relayblock` được lấy từ <http://www.informatik.uni-kiel.de/~ca/E-mail/check.html> dưới mục có đầu đề "`check_rcpt`" (chú ý, không được nhầm sang mục "`check_relay`"). File `/etc/mail/LocalIP` chứa danh sách các số mạng để chỉ các mạng con mà bạn muốn cho phép dùng relay. Ví dụ, để cho phép dùng relay từ mạng ICBV, file `/etc/mail/LocalIP` phải chứa 202.192 và ngoài ra không có gì nữa. Mỗi dòng vào, nếu có nhiều hơn một dòng vào, phải được viết riêng trên từng dòng, giống như trong các file khác của `sendmail`.

Nhằm nâng cao tính bảo mật khi dùng thư điện tử, người ta thường dùng mặt nạ che địa chỉ IP thực của máy chủ trong mạng. Một trong những đặc tính như vậy có trong phiên bản `sendmail` 8.9.1 có thể cài nâng cấp cho Solaris 2.6, hoặc `sendmail` của Solaris 2.7. Bạn có thể tải về `sendmail` 8.9.1 từ địa chỉ www.sendmail.org hoặc từ <http://sunfreeware.com> ở dạng nhị phân sẵn sàng cho việc cài đặt bằng lệnh **pkgadd**. Các file của `sendmail` được tải về ở dạng tar được nén bằng **gzip**. Sau khi tải về ta phải dần nén bằng lệnh **unzip**, sau đó mở bằng lệnh **tar** (xem **tar xvf**) vào trong thư mục `sendmail-8.9.1/src`. Bước tiếp theo là gõ các lệnh **./Build**, và **./Build install** với quyền của `root`. Tài liệu hướng dẫn đầy đủ ở dạng postscript có thể tìm thấy trong file `sendmail-8.9.1/doc/op/op.ps`.

Một lựa chọn khác so với phương án dùng lệnh **pkgadd** là biên dịch `sendmail` từ mã nguồn. Trước hết, ta phải cài đặt lên máy gói phần mềm mới Berkeley DB và bộ xử lý macro m4 UNIX. Các phần mềm này có thể tải về từ địa chỉ <http://sunfreeware.com> tương ứng trong các file `db-2.4.14-local.gz` và `4-1.4-local.gz`.

Sau khi đã có sendmail 8.9.1, ta cần tạo file `sendmail.cf` để cài thiện việc che mặt nạ cho cả máy chủ lẫn phong bì thư. Ta chuyển đến thư mục `cf/cf` của bộ cài đặt sendmail 8.9.1. Trong trường hợp sử dụng phương pháp cài từ bản nhị phân (dùng **pkgadd**), thư mục `cf` sẽ nằm trong `/opt/local/cf`. Sao file `generic-solaris2.6.mc` vào `your_host.mc` và sửa file cho giống với dạng đoạn mã dưới đây, thay `your_isp_domain` bằng tên vùng của ISP (Internet Service Provider - nhà cung cấp dịch vụ Internet). Ví dụ, đối với VDC (Công ty điện toán và truyền số liệu - Tổng cục Bưu chính viễn thông Việt Nam), tên vùng sẽ là `vnn.vn`. File `README` trong thư mục `cf` sẽ cho biết thông tin chi tiết về việc tạo ra file `sendmail.cf` theo yêu cầu.

File cấu hình sendmail m4 .mc :

```
1: VERSIONID(`@(#)solaris2.6.mc 8.9 (Berkeley)
    5/19/98')
2: OSTYPE(solaris2)dnl
3: DOMAIN(generic)dnl
4: MASQUERADE_AS(your_isp_domain)dnl
5: FEATURE(masquerade_envelope)dnl
6: MAILER(local)dnl
7: MAILER(smtp)dnl
8: define(`confDOMAIN_NAME', `not.a.real.machine')dnl
```

Các dòng 2 và 3 sẽ cấu hình cho hệ thống Solaris nói chung. Dòng 4 sẽ tạo mặt nạ cho các tiêu đề thư điện tử và dòng 5 - cho các phong bì. Các dòng 6 và 7 kích hoạt bộ phận phát thư cục bộ cho các địa chỉ cục bộ và giao thức SMTP (Simple Mail Transfer Protocol) cho các địa chỉ từ xa. Dòng 8 cho phép bạn nhặt tên vùng được nhìn thấy trên các tiêu đề thư để tin chắc rằng mọi người nhận thấy họ không thể trả lại thư về máy của bạn, có tên được xác định trong `/etc/hosts`. Nếu bạn muốn thư của bạn được chuyển tiếp thông qua ISP thì bạn thêm vào một dòng:

```
define(`SMART_HOST', your_isp_smtp_server)dnl
```

trong đoạn mã cấu hình relay sau đây. Ví dụ, đối với Trường Đại học Bách khoa Hà Nội, `your_isp_smtp_server` sẽ được đặt là `hut.edu.vn`.

File cấu hình relay sendmail `m4 .mc` được xây dựng như sau:

```
VERSIONID(`@(#)solaris2.6.mc          8.9 (Berkeley)
5/19/98')
OSTYPE(solaris2)dnl
DOMAIN(generic)dnl
MASQUERADE_AS(your_isp_domain)dnl
FEATURE(masquerade_envelope)dnl
define(`SMART_HOST', your_isp_smtp_server)dnl
MAILER(local)dnl
MAILER(smtp)dnl
define(`confDOMAIN_NAME', `not.a.real.machine')dnl
```

Sau đây là các ưu và nhược điểm khi chuyển tiếp thư qua máy chủ chuyển tiếp. Ưu điểm là thư điện tử sẽ được phân phối ngay lập tức đến ISP của bạn sau khi được chuyển tiếp. Nếu bạn không chuyển tiếp thư và nếu có sự cố gì đó ở bộ phận phát thư, thư sẽ xếp hàng chờ để phân phát sau trong máy của bạn, và nếu kết nối PPP bị ngắt, thư sẽ còn nằm mãi trong máy của bạn trước khi nối kết được lập lại. Ngược lại, nếu giao tiếp cục bộ trực tiếp được với máy đích, bạn có khả năng theo dõi và điều khiển tốt hơn. Lệnh `mailq` cho phép bạn theo dõi trạng thái của các thư đã được gửi đi, và file `/var/log/syslog` của bạn sẽ cho thấy thư được phân phát đúng địa chỉ. Để tạo file `sendmail.cf` sau khi sửa file `.mc`, ta thực hiện lệnh:

```
m4 ../m4/cf.m4 your_host.mc > sendmail.cf
```

trong thư mục `cf/cf`.

Ta sao file `sendmail.cf` vào thư mục `/etc`, và các mã nhị phân của `sendmail` vào `/usr/lib`, nếu bạn chưa thực hiện các thao tác này, và khởi động lại `sendmail` từ thư mục `/etc/init.d` bằng lệnh `./sendmail stop`, sau đó - `./sendmail start`. Giá trị khoảng thời gian chạy mặc định của

`sendmail` là một giờ, bạn có thể quy định khác đi bằng cách sửa script `/etc/init.d/sendmail`. Ví dụ, để `sendmail` cứ mười phút chạy một lần, bạn viết lệnh mới `/usr/lib/sendmail -bd -q10m` thay cho lệnh `/usr/lib/sendmail -bd -q1h`.

Ta phải kiểm tra để chắc chắn rằng đã thêm tên máy chính xác và đầy đủ vào trong file `/etc/hosts`, hoặc dùng một trong số các lệnh trong file `.mc` của mình

```
define(`confDOMAIN_NAME',`not.a.real.machine')dnl
```

Nếu bạn không làm như trên, **sendmail** sẽ bị treo 60 giây trước khi cố gắng chạy lại.

Nếu máy chủ dùng Solaris 7, bạn chỉ cần sửa lại file `etc/mail/sendmail.cf` như trong đoạn mã dưới đây, sau khi sao file `etc/mail/main.cf` vào `etc/mail/sendmail.cf`, rồi khởi động lại `sendmail`. Không nên sửa macro `$j` ở dòng 67 thành tên vùng ISP của bạn, nếu không, **sendmail** sẽ cố phân phát tất cả thư với địa chỉ của các người dùng khác trong vùng của ISP về máy tại chỗ của bạn. Các thay đổi trong `sendmail.cf` để dùng cho Solaris 7 như sau:

```
Line 65: # tên vùng chính thức
```

```
Line 67: Djnot.a.registered.domain
```

```
Line 72: # "Smart" relay host (may be null)
```

```
Line 72: DSyour_ismp_server
```

```
Line 112: # who I masquerade as (null for
```

```
    No masquerading) (see also &=M)
```

```
Line 113: DMyour_ism_domain
```

Có thể kiểm tra việc cài đặt **sendmail** bằng cách tự gửi thư cho mình ở địa chỉ cho bởi ISP, dùng lệnh **mailx -v your_mail_address**. Thực hiện lệnh **mailx** với tùy chọn `-v` sẽ cho thấy sự trao đổi SMTP giữa các daemon **sendmail** của bạn và của ISP.

VI. Quản trị hệ thống

VI. 1. Giới thiệu hệ điều hành UNIX Solaris

Lịch sử xuất hiện và phát triển, phân nhánh và chuẩn hóa của UNIX đã được nói tới trong chương đầu tiên *Tổng quan về hệ điều hành UNIX* của cuốn sách này. Solaris với các phiên bản từ 2.4 trở về sau - năm 2001 là 2.8 (đôi lúc trong các tài liệu kỹ thuật xuất bản trước còn có tên SunOS 5.x) - là một sản phẩm UNIX System V Release 4 hết sức phổ biến. Solaris kết hợp trong mình những yếu tố của cả dòng UNIX System (AT&T,...) lẫn của BSD (Berkeley). Có thể nói trên thế giới hiện nay, tuyệt đại đa số các mạng lớn chạy trên hệ điều hành UNIX, trong đó tỷ lệ Solaris là đáng kể). Solaris đại thể về hệ lệnh là như nhau, nhưng xét về mặt thiết bị được chia thành hai nhánh. Nhánh lớn hơn là **Solaris cho SPARC** - hệ điều hành chạy trên các máy SunSPARC và PowerPC có CPU kiểu xử lý song song (RISC). Nhánh thứ hai là **Solaris cho x86** - dùng trên các máy tính trên cơ sở CPU Intel và tương thích. Hiện ở Việt Nam sử dụng cả các hệ thống Solaris cho SPARC (một số cơ quan có thiết bị viện trợ - ở quy mô tương đối nhỏ, VDC,...) và Solaris cho x86 (trước hết - mạng WAN của Ngân hàng Công thương Việt Nam). Với nhu cầu và tình hình phát triển các hệ thống và ứng dụng mạng ngày càng nhanh hiện nay, chắc chắn các hệ điều hành nói chung và Solaris nói riêng, sẽ được sử dụng ngày càng rộng rãi ở nước ta.

VI. 2. Quyền đăng nhập superuser

Trong hệ thống UNIX *superuser* là tài khoản người dùng được ưu tiên cao nhất, gắn với tên người dùng tối cao *root*, có quyền đăng nhập không hạn chế tới bất cứ file và người dùng khác nào. Nhiều nhiệm vụ quản trị và các lệnh có liên quan đòi hỏi phải có quyền *superuser* mới thực hiện được. Ví dụ, thao tác tắt máy trong các bản UNIX thương phẩm phải do *root* thực hiện - hoạt động của các server trong doanh nghiệp hay server Web do người quản trị chịu trách nhiệm - không thể đổ thừa nguyên nhân tắt máy cho một người dùng bất kỳ nào khác.

VI. 2. a. Làm thế nào để trở thành superuser

Có hai cách đăng nhập quyền superuser. Cách thứ nhất, tất nhiên, bạn đăng nhập vào với tên và mật khẩu của root. Cách thứ hai, dùng lệnh **su** khi đã đăng nhập vào hệ thống với tên người dùng khác:

```
$ su [-]  
password: <mật khẩu của root>  
#
```

dấu nhắc hệ thống “#” cho thấy bạn đã trở thành superuser sau khi vào đúng mật khẩu của root. Trong trường hợp gõ vào sai mật khẩu, bạn sẽ nhận được thông báo: “Sorry ...” và sau vài lần cố gắng lại thấy mình ngồi trước màn hình với trạng thái cũ, trước khi hệ thống hoàn toàn cự tuyệt bạn. Khi muốn kết thúc công việc cần làm với vai trò root, bạn cần gõ lệnh **exit** hay **^D**. Bạn có thể chỉ dùng shell **su** và giấu nó xuống thành tiến trình nền bằng lệnh **suspend**, hoặc lệnh **bg**, hoặc lại lấy nó lên bề mặt bằng lệnh **fg**.

Khi bạn chạy lệnh **su**, shell mới thừa hưởng môi trường từ shell hiện thời của bạn. Dấu “-” trong tùy chọn của lệnh **su** giúp bạn mô phỏng phiên làm việc thực tế của root.

root bao giờ cũng phải có mật khẩu; mật khẩu này phải được thay đổi thường xuyên vì lý do bảo mật. Về các quy tắc và các điều nên tránh khi đặt mật khẩu, xin xem lại ở phần cuối của chương II *Đăng nhập hệ thống*. Để thay đổi mật khẩu của superuser, trước hết bạn phải trở thành superuser hay root, sau đó đánh lệnh: **# passwd [root]**. Hệ thống sẽ yêu cầu bạn gõ vào lại mật khẩu cũ của root, sau đó hai lần đưa vào mật khẩu mới. Đôi khi, ở những hệ UNIX khác nhau, sau khi **su** thành root, bạn phải chỉ rõ root khi dùng lệnh **passwd** để tránh hệ thống lại thay đổi mật khẩu chính tên người dùng của bạn. Ngược lại, nếu đã là root, để trở thành một người dùng khác trong cùng hệ thống, bạn chỉ cần thực hiện lệnh như sau: **# su <username>** mà không cần biết đến mật khẩu của người ấy.

Đối với ngay cả người quản trị, việc dùng lệnh **su** để trở thành root chỉ khi thật cần thiết là hết sức tiện lợi cho mục đích bảo mật.

Đặc biệt cần chú ý, chính vì superuser có quyền tối cao trong hệ thống UNIX, nên rất dễ thay đổi mọi thứ, thậm chí làm hỏng cả hệ thống file. Trong thực tế công việc, chính tác giả cũng đã gặp phải những trường hợp như người dùng, sau khi đăng nhập với quyền root, đã di chuyển một file có tên /vmunix sang thư mục khác và thay bằng một file khác tương tự. Sau khi tắt máy, hệ thống UNIX đã không thể khởi động trở lại cho đến khi lỗi được tìm ra và khắc phục xong. Vì vậy, nên tuân theo một quy tắc vàng sau: *những người biết mật khẩu của root, kể cả người quản trị hệ thống, không nên dùng quyền root để làm những công việc thường ngày, mà chỉ dùng quyền đó khi thật cần thiết.*

VI. 2. b. Thực hiện lệnh với quyền root hạn chế

▪ Thực hiện một lệnh đơn lẻ

Khi làm việc tại máy trạm và cần thực hiện một lệnh với quyền root, bạn nên dùng lệnh **su** với tùy chọn -c:

```
$ su root -c <["]command ["]>
```

command là lệnh bạn cần thực hiện, nếu chứa cả khoảng trống hoặc các ký tự đặc biệt của shell thì bạn phải để trong hai dấu ngoặc kép. Nếu bạn đưa vào đúng mật khẩu của superuser, lệnh bạn đưa vào ở dòng trên được thực hiện như là với root, lệnh kế tiếp được thực hiện từ shell nguyên thủy. Nếu lệnh kết thúc hoặc thực hiện bị lỗi, điều khiển sẽ quay trở lại shell có quyền hạn chế của người dùng.

Ví dụ: nếu cần mount ổ CD vào hệ thống UNIX (Solaris), ta thực hiện các lệnh và theo dõi các thông báo xuất hiện trên màn hình như sau:

```
$ su root -c "mount /dev/cdrom/Solaris_Updated_Drivers  
/mnt"  
passwd: <mật khẩu của root>  
$ df /mnt  
Filesystem    kbytes    used    avail   capacity Mounted on  
/dev/cd0      1348044  1348044    0    100%      /mnt
```


▪ *Tiện ích sudo*

UNIX chuẩn đảm bảo hoặc là quyền đăng nhập đầy đủ, hoặc tuyệt đối ngăn chặn quyền đăng nhập superuser. Đôi khi bạn lại rơi vào tình thế giữa dòng: cần thực hiện lệnh với quyền root, nhưng trong tay lại chỉ có tài khoản của một người dùng cụ thể. Tiện ích **sudo** được cung cấp rộng rãi, cho phép người dùng một khi đã đăng nhập hệ thống, có thể thực hiện các lệnh (không phải các shell script) trong thời gian mặc định 5 phút với quyền của root. Mật khẩu được yêu cầu để thực hiện lệnh là mật khẩu của người dùng cụ thể, chứ không phải của root.

Cú pháp của tiện ích này như sau:

\$ sudo command

<command là lệnh muốn được dùng với quyền root, có đường dẫn đầy đủ>

password: password

<mật khẩu của user dùng **sudo**>

sudo dùng file cấu hình `/etc/sudoers` do người quản trị hệ thống soạn để xác định từng người dùng và các lệnh mà từng người dùng trên được phép dùng với **sudo** theo cú pháp dạng:

user host=command[s]

với **host** - tên hoặc bí danh của máy. Nếu thế vào chỗ **host** và lệnh là **ALL**, có nghĩa là tất cả các máy và tất cả bộ lệnh tùy theo từng máy cụ thể. Để minh họa, ta hãy xét ví dụ `/etc/sudoers` sau:

```
# Host alias specification
Host_Alias KETOAN=hoa, ha, dung, binh, chuyen
Host_Alias TINDUNG=viet, son, thang, nga
#Cmd alias specification
Cmd_Alias BACKUP=/usr/sbin/bkdump, /usr/sbin/ufsrestore
Cmd_Alias SHUTDOWN=/usr/sbin/shutdown, /usr/sbin/tat
# User specification
```

```

root      ALL=ALL
hoa       TINDUNG=BACKUP:KETOAN=ALL
thang     ALL=SHUTDOWN:KETOAN=/usr/bin/uucp/uucheck

```

Trong ví dụ trên, những người dùng hoa, ha, dung, binh, chuyen được coi như “thuộc” về máy KETOAN, còn viet, son, thang, nga – máy TINDUNG. Các lệnh được phép dùng với quyền root là lệnh lưu dữ liệu ở dạng nén vào băng từ (**bkdump**) và khôi phục dữ liệu từ băng từ ra đĩa cứng (**ufsrestore**), được ký hiệu bởi bí danh BACKUP, các lệnh tắt máy của hệ thống UNIX (**shutdown**) và lệnh tắt máy do root tự viết để chấm dứt các ứng dụng và đóng cơ sở dữ liệu rồi mới tắt máy (**tat**) – dưới bí danh SHUTDOWN. File `/etc/sudoers` trên quy định root của máy KETOAN có quyền như root trên máy TINDUNG và ngược lại; người dùng hoa có quyền như root để thực hiện các công việc liên quan tới lưu trữ và phục hồi dữ liệu trên cả máy “của mình” lẫn máy TINDUNG, ngoài ra còn có thể tắt máy KETOAN; người dùng thang có quyền tắt tắt cả hai máy trên (quyền tắt máy trong hầu hết các bản UNIX thương phẩm chỉ thuộc về root), và còn có thể kiểm tra đường truyền thông trên máy KETOAN.

VI. 3. Các file quản lý hệ thống của UNIX

❖ crontab

File này được tạo bởi lệnh **crontab -e** và nằm tại `/var/spool/cron/`, được dùng để định lịch cho các lệnh, shell script và các chương trình khác với những khoảng thời gian đều đặn. Mỗi dòng trong **crontab** có cấu trúc gồm sáu trường cách nhau bởi dấu phẩy hay Tab, theo thứ tự: phút (0 - 59); giờ (0 - 23); ngày trong tháng (1 - 31); tháng (1 - 12); ngày trong tuần (0 - 6); dòng lệnh được thực hiện theo lịch đặt. Nếu một trong năm trường đầu để trống thì dùng dấu “*”. Ví dụ:

```

7      ,2 ,*,    *,1-6      /usr/lib/acct/runacct 2> \
      /usr/adm/acct/nite/fdzlog

```

Daemon tương ứng là `/etc/cron`. Để có thể soạn thảo được file **crontab**, nếu chưa có, phải đưa vào biến môi trường `EDITOR=/usr/bin/vi` (nếu bạn dùng chương trình soạn thảo **vi**) vào môi trường shell tương ứng. Tốt nhất là đưa các

giá trị cần thiết của các biến môi trường vào trong các file `.profile` tương ứng của từng người dùng (với root là `/.profile`, với người dùng có tên `username` - là `/home/username/.profile`):

```
EDITOR = /usr/bin/vi
export EDITOR
```

❖ fstab

Cho thấy các hệ thống file được mount vào máy khi máy khởi động, cũng như khi cấp thêm dung lượng cho swap. File được đọc bởi các lệnh `mount`, `umount` và các lệnh kiểm tra hệ thống file. Có thể xem xét cả file tương đương `/etc/vfstab` có riêng trong Solaris.

❖ passwd

Là file quan trọng, chứa tất cả tên người dùng, mật khẩu đã mã hóa, và các thông tin khác, nằm ở `/etc`. Mỗi dòng gồm bảy trường cách nhau bởi dấu ":" theo cú pháp:

```
login-id:passwd:user-id#:group-id#:User Info:home-dir
:shell
```

- `login-id` : 2 - 8 ký tự, chữ thường.
- `password` : mật khẩu đã mã hóa, 13 ký tự.
- `user-id#` : số UID, 0 đến 60000, giá trị cực đại được quy định trong file `/usr/include/sys/param.h` bởi biến `MAXUID`.
- `group-id#` : số GID, 0 - 60000.
- `User Info` : tên thật hay bất cứ thông tin khác của người dùng.
- `home-dir` : đường dẫn tới thư mục của người dùng.
- `shell` : shell ban đầu của người dùng, nếu không có, sẽ có dòng chỉ shell mặc định `/usr/bin/sh`

❖ shadow

Là biện pháp tăng cường bảo mật của Solaris, nằm ở /etc. Trong khi file `passwd` có thể đọc được bởi bất kỳ ai, `shadow` cũng chứa các mật khẩu đã mã hóa lại chỉ có thể đọc và sửa chữa bởi `root`. Mỗi dòng trong file `shadow` gồm chín trường sau:

```
login-id:password:lastchg:min:max:warn:inactive:
                                expire:flag
```

- `lastchg` : số ngày từ 1/1/1970 tới thời điểm thay đổi mật khẩu lần cuối.
- `Min` : số ngày tối thiểu giữa hai kỳ thay đổi mật khẩu.
- `Max` : số ngày tối đa giữa hai kỳ thay đổi mật khẩu.
- `Warn` : số ngày trước khi hết hạn của mật khẩu đã đăng ký và người dùng sẽ được cảnh báo.
- `Inactive` : số ngày được phép không hoạt động của người dùng.
- `Expire` : ngày hết hạn dùng mật khẩu
- `Flag` : chưa dùng đến

❖ group

Cũng nằm ở /etc, mỗi dòng chứa bốn trường cách nhau bởi dấu ":" và chứa tên nhóm và tên thành viên của mỗi nhóm:

```
group-name:password:GID:list
```

- `group-name` : tên nhóm
- `password` : nếu trống, khi thay đổi nhóm bạn sẽ không được nhắc đổi mật khẩu
- `GID` : định danh nhóm
- `list` : danh sách các người dùng thuộc nhóm, cách nhau bởi dấu phẩy ",",

Người quản trị hệ thống có thể tổ chức tất cả các người dùng và quản lý theo các mô hình phù hợp với yêu cầu từng thời kỳ hoặc dự án.

❖ hosts

Nằm ở `/etc`, mỗi dòng vào có dạng (trường `Aliases` để chỉ các tên khác nếu có, của máy, được viết cách nhau bằng dấu Tab, nếu có nhiều hơn một tên khác):

IP address	Hostname	Aliases
------------	----------	---------

Chương trình tìm trong file `/etc/hosts` địa chỉ của tất cả các máy, kể cả máy đang nối tới, cần liên hệ trong các dịch vụ truyền thông, mạng. Chú ý, dòng "127.0.0.1 localhost" luôn luôn cần có mặt trong file `hosts` để máy có khả năng "tự tìm thấy" mình. Nên chú ý, việc thay đổi tên máy (`hostname`) không đơn giản như đối với địa chỉ IP ở trong file `/etc/hosts`. Nó kéo theo sự thay đổi bắt buộc tương ứng của tên máy trong các file `/etc/net/ticlts/hosts`; `/etc/hostname.<network_card_name>`; `/etc/nodename`; `/etc/net/ticots/hosts`; `/etc/net/ticotsord/hosts`.

❖ hosts.equiv

File `/etc/hosts.equiv` là danh sách các máy mà người dùng của chúng được phép đăng nhập hệ thống file máy của bạn qua mạng, không cần tới mật khẩu. Nói nôm na là chúng có thể được "nối" với nhau, dùng chung đĩa cứng. Kinh nghiệm thực tế cho thấy nên để chính tên máy của mình lại trong file này. Cú pháp của dòng trong file có dạng:

```
hostname [-[@]] username (hay groupname)
```

Dấu "+" là mặc định, dấu "-" loại bỏ một cách phân biệt quyền này của người dùng hoặc nhóm người dùng có tên nêu ở trên.

❖ .rhosts

Tương tự như `/etc/hosts.equiv` nhưng cho mỗi người dùng được nói đến, và nằm tại thư mục riêng của họ, cho phép các người dùng khác từ các máy khác đăng nhập vào. Trong một vài trường hợp, tạo cho `root` một file `/.rhosts` cho phép tiến hành lưu trữ từ xa.

❖ aliases

Có hai loại file aliases: `/etc/aliases` - cơ sở dữ liệu được dùng bởi chương trình tìm đường thư tín điện tử `sendmail` và aliases cho dịch vụ tên vùng (NIS). Loại thứ hai ta sẽ xét tới sau. `/etc/aliases` dùng để tạo ra tên khác cho địa chỉ thư điện tử của người dùng hoặc tên chung cho một nhóm người dùng. Thay đổi nội dung các dòng :

`new_name : old_name (hay E-mail address)`

bằng lệnh **`newaliases`**.

❖ **printcap**

Cơ sở dữ liệu tại chỗ chứa tên các máy in nối vào máy của bạn. Mỗi dòng gồm tên máy in, các bí danh, vị trí (thư mục) hàng chờ, và các khả năng của máy in. Các trường cách nhau bởi dấu "|". Ký hiệu các bộ lọc dùng trong file này:

- `if` bộ lọc đầu vào các tác vụ thuần văn bản
- `of` bộ lọc đầu ra (tiêu đề)
- `rf` bộ lọc file văn bản kiểu FORTRAN
- `tr` bộ lọc dữ liệu không để lại vết
- `nf` bộ lọc dữ liệu không để lại vết và không phụ thuộc thiết bị
- `df` bộ lọc dữ liệu TeX (dạng DVI)
- `cf` bộ lọc dữ liệu `cifplot`
- `gf` bộ lọc dữ liệu đồ họa
- `vf` bộ lọc hình ảnh màn hình

❖ **dumpdates**

Chứa các bản ghi mỗi lần lưu trữ hệ thống file, nằm ở `/etc`. Mỗi dòng vào có bảy trường với nội dung như sau:

Tên thiết bị	Mức lưu trữ	Ngày (tuần)	Tháng	Ngày (tháng)
Giờ:Phút:Giây	Năm			

❖ **ttytab**

Cơ sở dữ liệu cho truyền thông, nằm ở `/etc`, chứa các bản ghi mô tả các thiết bị đầu cuối nối vào cấu hình máy. Mỗi dòng vào là một bản ghi năm trường có dạng:

name	getty	type	status	comments
------	-------	------	--------	----------

❖ install

Chứa các file thông tin về cấu hình hệ thống sau khi cài đặt xong.

❖ bootparams

Nằm ở /etc. Các máy client phải lấy thông tin từ máy chủ NFS để khởi động.

❖ network

Nằm ở /etc. Chứa tên và địa chỉ của các mạng tham gia liên mạng đang nói tới.

❖ ethers

Nằm ở /etc. Là cơ sở dữ liệu chứa các địa chỉ Ethernet của các máy trong mạng và tên tương ứng của chúng (thông số của các card mạng).

❖ netgroup

Nằm ở /etc. Có vai trò và chức năng giống như /etc/group, nhưng ở quy mô toàn mạng.

❖ exports

Nằm ở /etc. Chứa danh sách các hệ thống file mà máy chủ NFS đánh dấu là có thể chia sẻ cho các máy khác trong mạng.

VI. 4. Các mức chạy

VI. 4. a. Phân loại mức chạy

Hệ thống UNIX System V có tám mức chạy, ở mỗi mức có những đặc thù riêng tùy theo chế độ vận hành như sau (một số tài liệu về UNIX không phân biệt các mức s và mức 1):

- **s hoặc S:** Khi file /etc/inittab trống, hay khi ta chú ý vào chế độ này, tiến trình **init** sẽ đi vào mức chạy *một người dùng*, thường là root. Chỉ có các tiến trình cần thiết của kernel được chạy nếu máy tính bước vào mức chạy s từ thời điểm khởi động, nhưng nếu máy tính hạ mức chạy xuống mức

s từ các mức nhiều người dùng thì các hệ thống file của người dùng vẫn không bị gỡ bỏ. Mức chạy s dùng để thực hiện những chức năng bảo trì như cài đặt thêm các gói phần mềm, thay đổi vị trí các hệ thống file, thêm hoặc kiểm tra các thiết bị mới, kiểm tra và sửa chữa các hệ thống file, v.v..

Ngoài ra, còn bảy mức chạy, được gọi từ file `/etc/inittab` mà người quản trị hệ thống có thể cho chạy bằng lệnh **init**:

- **0:** Dùng để thực hiện lệnh *tắt toàn bộ hệ thống*. Lần khởi động tiếp theo phải thực hiện bằng tay. Tại mức này các người dùng được cảnh báo để thoát khỏi hệ thống, tất cả các hệ thống file của người dùng được unmount, các daemon bị dừng, hệ thống sẽ ở vào tình trạng an toàn để tắt điện. Có thể tùy theo thiết kế phần cứng các máy chủ còn tự ngắt điện.
- **1:** Đây cũng là mức chạy *một người dùng*, chỉ khác là được gọi từ `/etc/inittab`. Các hệ thống file vẫn được mount. Được thiết kế để chấm dứt các tiến trình nền chuẩn, gỡ bỏ các hệ thống file và hủy bỏ những gì đã xảy ra khi hệ thống chạy ở mức 2 trước đó, cũng như để cài đặt các gói phần mềm tiện ích tùy chọn.
- **2:** *Mức chạy nhiều người dùng*, thường được chạy ngay lập tức sau khi thủ tục khởi động hệ thống kết thúc. Các hệ thống file được mount, các daemon bắt đầu chạy, các dịch vụ như in, quản lý đĩa,... trong máy hiện thời được khởi động. Tiến trình getty được chạy cho mỗi thiết bị đầu cuối gắn vào máy.
- **3:** *Mức chạy nhiều người dùng trong môi trường mạng*, giống như mức 2, nhưng được hỗ trợ thêm các chức năng mạng và chia sẻ file từ xa.
- **4:** *Mức chạy nhiều người dùng trong môi trường mạng*, giống như mức 3, nhưng do người dùng xác định. Hiện chưa thấy được dùng.
- **5:** *Mức chạy phần cứng*. Dùng để dò lỗi phần cứng, ít dùng trong thực tế. Mức này không thể thực hiện được việc tắt hoàn toàn hệ thống.
- **6:** *Mức tắt máy và tự động khởi động trở lại*. Tên gọi lỏng là khởi động nóng, khác với các mức 0 và 5 là khởi động nguội, do phải tắt điện rồi bật lại máy.

Riêng trong Solaris còn có các mức phụ khác được thiết lập như sau:

- **a, b, c:** chỉ xử lý các dòng vào của `/etc/inittab` có các mức chạy a, b, và c được thiết lập. Đây là các *giá trạng thái*, cho phép xác định chỉ một số

lệnh nào đó được chạy mà thôi, nhưng không làm ảnh hưởng tới các mức chạy hiện thời.

- **Q** hoặc **q**: Kiểm tra lại `/etc/inittab`.

VI. 4. b. Quy định mức chạy cho người dùng

Theo mặc định, các hệ thống UNIX khi khởi động sẽ lần lượt chuyển mức chạy từ mức **s** sang mức chạy theo chế độ xác định. Nếu máy chạy ở chế độ đơn lẻ (Stand-alone), mức chạy được dùng là **1**, người dùng chính là **root**. Nếu máy dùng trong mạng, các mức chạy sẽ là **2** hoặc **3**. Mức **2** thường được dùng với các hệ thống làm việc ở chế độ dòng lệnh, còn mức **3** - ở môi trường đồ họa. Tuy nhiên, nếu ở chế độ dòng lệnh mà hệ thống có sử dụng các dịch vụ mạng phổ biến như DNS, Apache, v.v.. thì mức chạy cũng là **3**. Các mức chạy khác thường được dùng với các mục đích cụ thể xác định và do người quản trị quy định. Chỉ có người quản trị (**root**) mới có quyền chạy lệnh chuyển mức chạy bằng lệnh `/sbin/init` (hoặc `/etc/telinit`) theo cú pháp:

```
/sbin/init [0123456abcQqSs]
/etc/telinit [0123456abcQqSs]
```

Mỗi một mức chạy được xác định với môi trường và các phần mềm hệ thống và ứng dụng cụ thể trong các thư mục `con rc#.d` (**#** - chỉ ký hiệu của mức chạy) tương ứng nằm trong `/etc/init.d` (Solaris). Các script trong các thư mục con trên được bắt đầu với các chữ cái **S** (start - bắt đầu) và **K** (kill - kết thúc).

Đối với các bản Linux như SuSE Linux, RedHat, v.v.., mức chạy **3** là mức chạy trong môi trường mạng, nhiều người dùng, nhưng trong chế độ dòng lệnh. Mức chạy **5** lại là mức tương tự như mức **3**, được thực hiện trong môi trường đồ họa như KDE, GNOME, v.v..

Trạng thái của các mức chạy còn có thể được thay đổi bằng cách sửa các dòng vào trong file cấu hình hệ thống `/etc/inittab`. Các dòng vào đó có cú pháp như sau:

```
id:run-level:action:process
```

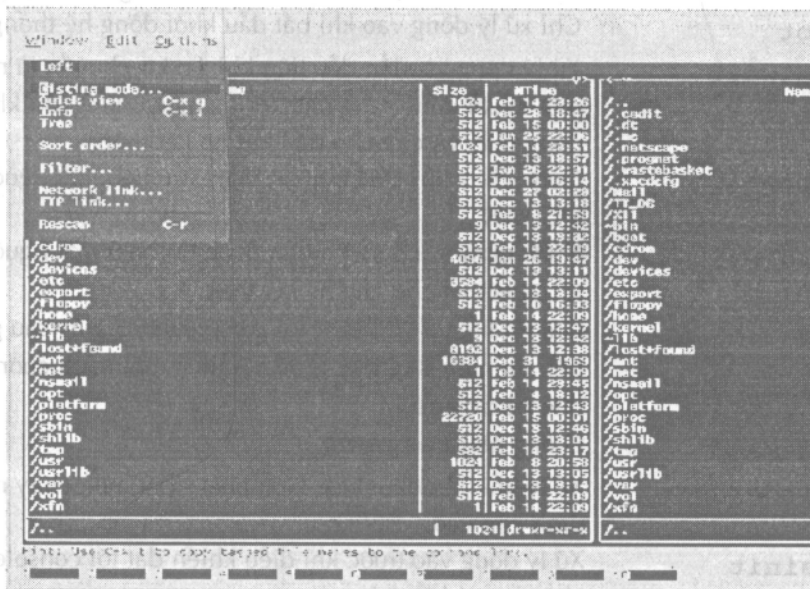
với:

id	Một hoặc hai ký tự định danh cho dòng vào
run-level	Mức chạy mà dòng vào này áp dụng tới. Nếu không chỉ rõ, sẽ áp dụng cho tất cả các mức từ 0 tới 6.
action	Chỉ ra cách thức mà process được tiến trình init xử lý. Các khóa hợp lệ là:
respawn	Khởi động process nếu chưa tồn tại, khởi động lại nếu process đã chết.
wait	Khởi động process và chờ cho nó kết thúc
once	Khởi động process khi vào mức chạy, nhưng không chờ nó kết thúc, và nếu nó chết - không khởi động lại.
boot	Chỉ xử lý dòng vào khi bắt đầu khởi động hệ thống.
bootwait	Xử lý dòng vào lần đầu tiên khi init chuyển từ mức một người dùng sang mức nhiều người dùng khi khởi động máy và chờ cho đến khi nó kết thúc.
powerfail	Được chạy khi init nhận được tín hiệu mất nguồn SIGPWR.
powerwait	Được chạy khi init nhận được tín hiệu mất nguồn SIGPWR và chờ cho nó kết thúc.
off	Gửi tín hiệu SIGTERM tới process, 5 giây sau gửi tiếp tín hiệu SIGKILL để kết thúc một cách cưỡng bức process.
ondemand	Giống như respawn .
initdefault	Xử lý khi lần đầu được kích hoạt - đặt mức chạy mặc định cần vào.
sysinit	Xử lý dòng vào trước khi điều khiển đạt tới console và chờ cho nó kết thúc.
process	Lệnh hay shell script cần chạy.

Tất nhiên, ta chỉ nên sửa đổi chế độ chạy cho các chương trình ứng dụng hoặc chương trình cho người dùng mà ta đã biết rõ để tránh mọi sự cố hệ thống có thể xảy ra. Nhất thiết không nên thay đổi các chi tiết của các dòng vào mà ta không biết hoặc không hiểu rõ nội dung của chúng, vì ở chế độ nền còn có thể có rất nhiều tiến trình (daemon) được quy định chạy ở các mức chạy khác nhau khi máy tính hoạt động với UNIX.

VI. 5. Sử dụng các tiện ích kiểu Midnight Commander (mc)

Đã từ lâu người dùng MS-DOS đã rất quen thuộc với tiện ích Norton Commander (NC), được viết và phổ biến bởi Peter Norton Computing. Trong UNIX, trước hết là trong Linux, một tiện ích tương tự, được gọi là Midnight Commander (**mc**) cũng đã xuất hiện và chiếm được nhiều cảm tình của người dùng. Màn hình làm việc của Midnight Commander bao gồm nhiều chức năng hơn NC. Ưu điểm nổi bật nữa của **mc** là có thể chạy được trên nhiều hệ điều hành khác nhau. Ta có thể thấy màn hình làm việc của **mc** được trình bày như sau.



H. VI. 1. Màn hình làm việc của Midnight Commander (mc).

VI. 5. a. Tải về mã nguồn của Midnight Commander

Nếu dùng trình duyệt bất kỳ để vào Web site www.gnome.org/mc, bạn sẽ có thể tải về mã nguồn, cũng như các file nhị phân của **mc**. Tuy nhiên, phải nói rằng tự biên dịch lấy phần mềm **mc** không quá dễ dàng. Nếu bạn dùng **gcc**, bạn chỉ cần chạy các script **Configure**, sau đó là **Make**. Vài phút sau, bạn cần phải gõ **make install** để định vị **mc** vào thư mục thường được quy ước là **/usr/local**. Có thể chỉ ra thư mục khác cho **mc**, nếu bạn thấy đó là thuận tiện hơn. Trong khi biên

dịch bạn có thể tham khảo tất cả các tùy chọn một cách trực tuyến từ Web site trên. Bạn cũng phải chú ý rằng có thể một vài phương án sử dụng **mc** ở chế độ đồ họa đang còn ở trong quá trình thử nghiệm (bản beta) nên không chạy được trên máy của bạn.

VI. 5. b. Khái niệm cơ bản

mc khởi động shell. Sau khi được khởi động, màn hình Midnight Commander được chia thành ba phần chính. Đó là hai bảng nằm kế nhau, giống như trong NC, dùng để chứa các thông tin thư mục, và dòng lệnh. Ngoài ra còn có thanh thực đơn nằm suốt bề ngang phía trên màn hình và danh sách các phím chức năng nằm phía dưới màn hình.

Mỗi bảng trên có thể hiển thị các thư mục giống hay khác nhau trong đĩa cứng của bạn. Các thao tác thường dùng nhất được đặt vào bảng thông qua thanh lựa chọn, trong đó, các thao tác như sao chép, di chuyển, đổi tên, ... lấy bảng còn lại làm đối tượng đích mặc định. Nếu cần dùng các thư mục khác làm đích, bạn cần đánh lệnh trực tiếp từ dòng lệnh.

Midnight Commander cũng sử dụng các phím nóng, là các chương trình dùng tương tự như các phím của trình soạn thảo GNU Emacs. Danh sách đầy đủ các phím nóng có thể tham khảo trang trợ giúp của Midnight Commander.

VI. 5. c. Chế độ liệt kê

Các chế độ liệt kê quy định phương cách mà các hệ thống file được hiển thị trong mỗi bảng. Để thay đổi chế độ liệt kê, bạn cần phải ấn phím F9 và chọn để thay đổi trong thực đơn sổ xuống cho bảng phía bên trái hoặc bên phải. Các chế độ được lựa chọn nằm trong số Full, Brief, Long, hay User. Chế độ *Full* cho biết tên file, kích thước file và thời gian cập nhật lần cuối. Chế độ *Brief* chỉ cho biết tên file, vì thế trong mỗi bảng sẽ sử dụng hai cột để hiển thị. *Long view* cho thấy chủ quyền, số liên kết, người sở hữu, nhóm làm việc của người sở hữu, kích thước, thời điểm sửa đổi lần cuối và tên file. Ở chế độ *Long* chỉ một bảng được hiển thị và chiếm hết chiều ngang màn hình.

Bạn có thể chọn chế độ *User*, nếu các chế độ khác không làm bạn vừa ý. Với chế độ này bạn có thể lựa chọn các thông tin cần hiển thị theo ý thích. Trước hết

bạn phải chọn kích thước bảng: toàn bộ (một bảng) hay một nửa màn hình (hai bảng). Sau đó, bạn có thể chỉ định chế độ hai cột như trong chế độ *Brief*, rồi chọn thêm vào tên các trường cần hiển thị. Các trường đó là kích thước file, loại file, thời điểm sửa đổi lần cuối, thời điểm đăng nhập cuối, thời điểm tạo file, các bit chủ quyền, chủ quyền ở dạng bát phân, số liên kết đến file, định danh nhóm (GID) bằng số của file, định danh người dùng (UID) bằng số của file, người sở hữu, nhóm làm việc của người sở hữu và inode của file.

Bạn cũng cần chỉ ra cách thức mà các trường trên được phân cách. Có thể dùng khoảng trống, dấu chấm, hoặc đường vạch đứng cho mục đích trên. Một lựa chọn khác là bắt chương trình hiển thị dấu sao "*" nếu file được lấy làm đích, hoặc khoảng trống, nếu file không được chọn.

VI. 5. d. Các chế độ khác của mc

Nếu bạn không muốn nhìn thấy hai thư mục nằm cạnh nhau, bạn có thể đặt để một trong hai bảng có một trong ba chế độ khác nhau. Khi bạn chọn một file trong một bảng, ở bảng kia sẽ hiển thị các thông tin về file tùy thuộc vào chế độ mà bạn lựa chọn sau:

Info - Thông tin, tương tự như được lấy bằng lệnh `ls -l`, về file được chọn, và nếu có thể, về cả hệ thống file hiện thời.

Tree - Dạng cây thư mục đơn giản của hệ thống file. Chỉ hiển thị tên các thư mục và thư mục con, còn các file có thể nhìn thấy trong bảng bên kia. Có thể duyệt các thư mục bằng cách chọn thư mục rồi ấn phím Enter.

Quick View - Khi chọn một file trong bảng khác, trong bảng mà ta đặt tại chế độ sẽ hiển thị nội dung của file nếu có thể. Chế độ này đặc biệt tiện lợi khi ta làm việc với các file văn bản, ví dụ, khi cần xem qua mã của một shell script đơn giản.

VI. 5. e. Phân loại và lọc

Các file có thể được phân loại theo nhiều tiêu chí khác nhau như: tên file (vân chữ cái), phần mở rộng, thời điểm sửa đổi, thời điểm đăng nhập, thời điểm sửa đổi inode của file, kích thước, inode hoặc không phân loại. Theo mặc định, **mc** bao giờ cũng đặt các thư mục lên đầu danh sách, sau đó mới đến các file. Bạn có thể thay

đổi bằng cách chỉ phân loại theo thứ tự tên văn chữ cái mà không phân biệt thư mục với file bằng cách ấn phím F9, sau đó chọn Options => Configuration => Mix All Files. Hay bạn cũng có thể sắp xếp các file và thư mục theo thứ tự ngược lại.

Giả sử bạn có một thư mục với số file lớn hơn 1000. Thật bất tiện nếu bạn cứ phải lục tung từng ấy file hay thư mục để tìm cái bạn cần. Lúc này việc đưa vào bộ lọc là cần thiết để giảm nhẹ khó khăn. Việc đưa bộ lọc vào khá đơn giản: bạn chỉ cần gõ các điều kiện lọc vào hộp thoại lọc và Midnight Commander sẽ tự làm phần việc còn lại.

Một mẹo vặt với bộ lọc có thể sẽ giúp ích cho bạn ít nhiều. Khi đặt lọc, chương trình sẽ hiển thị tổng dung lượng của các file được chọn. Bạn chỉ cần đặt bộ lọc với giá trị *All files* và sau đó đọc dung lượng đĩa được dùng ở phía dưới màn hình. Có vẻ như cách này dễ dùng hơn là thực hiện lệnh **du** ở chế độ dòng lệnh.

VI. 5. f. Tìm file

Để tìm file, ta ấn phím F9, sau đó chọn Command => Find File, nhập vào thư mục bắt đầu, tên file cần tìm, và theo tùy chọn, có thể cả xâu ký tự chứa trong nội dung file đó. Một khi tìm thấy file, bạn có thể làm cho chương trình chuyển đến thư mục chứa file đó, hoặc bạn có thể “bảng hóa” kết quả tìm kiếm. Trong trường hợp này, một trong hai bảng của **mc** sẽ được chuyển để hiển thị kết quả của lệnh Find. Trong bảng đó chỉ xuất hiện tên những file phù hợp với các tiêu chuẩn tìm kiếm.

Nếu muốn tìm file để xem hoặc sửa đổi nội dung file, bạn phải chú ý rằng **mc** có các tiện ích nội tại duyệt và soạn thảo riêng. Có thể đặt chế độ để sử dụng trình soạn thảo mặc định trong hệ thống UNIX của bạn, ví dụ như **vi**, để dùng trong **mc**. Nếu không nhớ chính xác tên thư mục bắt đầu, bạn nên dùng tùy chọn **Tree** từ thực đơn để dùng cấu trúc cây tìm thư mục bắt đầu.

VI. 5. g. Hỗ trợ hệ thống file ảo

Nếu dùng Midnight Commander trong hệ thống UNIX, bạn có thể truy nhập b hay bốn hệ thống file ảo. Đó là các hệ thống file được Midnight Commander hỗ : sau: Hệ thống file FTP ảo, Hệ thống file TAR ảo, Hệ thống file mạng ảo, và tr

Linux, nếu ta kích hoạt nó - Hệ thống file được phục hồi ảo. Sau đây ta sẽ xem xét chi tiết hơn lần lượt các hệ thống file ảo trên.

- *Hệ thống file FTP ảo*

Midnight Commander có thể được đặt để đăng nhập các ftp site giống như các trình duyệt Web. Bạn kết nối đến site từ xa đơn giản bằng cách chuyển thư mục đến địa chỉ ftp muốn truy nhập. Ví dụ:

```
cd ftp://ftp.netscape.com
```

sẽ kết nối bạn với ftp site của Netscape, từ đó bạn có thể lấy về phiên bản mới nhất của Netscape Communicator. Cú pháp đầy đủ của lệnh **ftp** là:

```
ftp://[!]user:password@remote host [:port] [directory]
```

Dấu "!", tên người dùng - user, mật khẩu - password, địa chỉ cổng kết nối - port và thư mục - directory, là các tùy chọn. Dấu "!" nhắc chương trình dùng qua máy proxy để cố gắng kết nối với máy từ xa thông qua **ftp**. Thay vì dấu "!", có thể mở file cấu hình mc.ini và đặt thêm vào ftpfs_always_use_proxy cho mọi phiên làm việc. Nếu quyết định dùng các thiết lập proxy, bạn cũng cần tạo file mc.no_proxy để xác định các máy tại chỗ không cần qua proxy. Bình thường các trình duyệt Web dùng cái gọi là chế độ thụ động để đăng nhập các ftp site, trong khi đó lệnh **ftp** chuẩn của UNIX lại dùng chế độ tích cực. Nếu bạn thêm vào file mc.ini dòng vào add ftpfs_use_passive_connections, chương trình sẽ truy cập ftp site như một nối kết thụ động.

Khi bạn đã kết nối được với ftp host, danh sách liệt kê thư mục và file của máy từ xa được hiển thị trong một bảng. Bạn có thể đặt bảng kia để hiển thị nội dung hoặc của hệ thống file của bạn, hoặc của một máy từ xa nào đấy khác nữa. Các thao tác bình thường với thư mục và file được thực hiện trong Midnight Commander giống y như trên máy tại chỗ của bạn. Để cắt kết nối với máy từ xa, bạn chỉ cần gõ lệnh **cd** tới thư mục trên máy tại chỗ.

- *Hệ thống file TAR ảo*

mc hỗ trợ hệ thống file TAR ảo chỉ đọc được (read-only). Người dùng đã được làm quen với lệnh **tar** của UNIX, dùng để nén và giải nén file, cũng như cho mục đích lưu trữ. Chi tiết hơn về lệnh **tar** xin tham khảo tại các trang trợ giúp, ví dụ như **man tar**, hay trong chương X “Lưu trữ và khôi phục dữ liệu”. Các lệnh khác như **gzip**, **gunzip**, **b2zip**, **rar**, v.v.. cũng thích hợp ở đây. Các loại file nén có thể dùng được trong Midnight Commander thường có phần đuôi **.tar**, **.tar.Z**, **.Z**, **.zip**, v.v.. Chú ý rằng trong UNIX tên file có thể chứa tới 255 ký tự và đôi khi bao gồm nhiều hơn một chu kỳ bắt đầu bằng dấu chấm “.”. Khi ta trở tới một file thuộc một trong các dạng đã nêu, sau đó ấn phím F3 hoặc Enter, chương trình sẽ cho ta thấy liệt kê thư mục của tất cả các file chứa trong file nén trên. Có thể thực hiện các lệnh trên ở chế độ dòng lệnh trong Midnight Commander nếu bạn muốn xác định chi tiết hơn các tùy chọn được dùng.

- *Hệ thống file mạng*

Midnight Commander có khả năng hỗ trợ hệ thống file mạng riêng của mình, một hệ thống file khác với NFS do Sun Microsystems phát triển và rất được phổ biến. Để sử dụng được hệ thống file mạng của **mc**, máy chủ từ xa (host) phải chạy một chương trình server đặc biệt có tên là **mcserver**. Sau đó bạn kết nối với máy chủ đó bằng cách chạy Midnight Commander trên máy client của bạn. Lúc này bạn có thể thao tác trên các bảng Midnight Commander của máy chủ như cách mà bạn dùng NFS.

- *Phục hồi hệ thống file ảo đã bị xóa*

Riêng trên các hệ thống Linux đã được kích hoạt hệ thống file ảo phục hồi, Midnight Commander có thể hoạt động như một tiện ích để phục hồi các file và thư mục đã bị xóa. Chi tiết hơn có thể tham khảo tài liệu về Midnight Commander.

VI. 5. h. Các điểm thuận tiện khác

Midnight Commander còn có nhiều tiện ích khác nữa như xử lý ngầm, danh sách nóng, thông báo về dung lượng. Bạn có thể thực hiện các lệnh Copy và Move trong thực đơn File ở chế độ nền, nếu muốn. Bạn cũng có thể chọn các tùy chọn để bất

đầu, dừng hoặc khởi động lại các tác vụ được thực hiện ngầm từ các mục trong thực đơn Background Job.

Danh sách nóng các thư mục là danh sách các thư mục mà bạn thông thường sẽ tới ngay lập tức khi dùng Midnight Commander. Chú ý rằng khi lập danh sách này, bạn có thể sử dụng được cả những liên kết tới các thư mục khác.

Việc *hiển thị dung lượng* của thư mục rất tiện lợi đối với người quản trị, cũng như đối với người dùng thông thường. Ta cần phải biết, ví dụ như liệu đĩa cứng trong máy có còn nhiều chỗ trống không, hay thư mục `home` của người dùng nào đó đã ngốn quá nhiều tài nguyên so với mức cho phép. Để thực hiện công việc cuối, ta chỉ việc dùng Midnight Commander để xem thư mục `home` đó, sau đó chọn *Show Directory Sizes* từ thực đơn. Có thể, để tính dung lượng của một hệ thống file lớn cần đòi hỏi đôi chút thời gian nên bạn sẽ phải chờ. Chương trình sẽ cho biết tất cả dung lượng được sử dụng cho mỗi thư mục và cho các thư mục con của chúng, sau đó hiển thị tên các thư mục và tổng số dung lượng đĩa đã được sử dụng.

VI. 6. Nhiệm vụ của người quản trị hệ thống

Sau khi đã làm quen kỹ hơn với hệ thống UNIX, bây giờ là thời điểm thích hợp để tổng kết lại các nhiệm vụ cơ bản của người quản trị hệ thống. Ở các chương cuối chúng ta sẽ xem xét chi tiết hơn một số nhiệm vụ cũng như kỹ năng cụ thể trong giới hạn hợp lý của cuốn sách.

1. Thêm, bớt và thay đổi các tài khoản người dùng.
2. Cài đặt phần mềm, cả phần mềm ứng dụng lẫn nâng cấp hệ điều hành.
3. Cài đặt thêm phần cứng: các bảng mạch bổ sung, máy in, modem, thiết bị đầu cuối, v.v..
4. Bảo trì bảo mật và tính toàn vẹn của hệ thống.
5. Chẩn đoán và khắc phục những sự cố về phần cứng và phần mềm, nếu chúng phát sinh.
6. Kiểm tra tình hình sử dụng hệ thống file, dung lượng đĩa.
7. Bảo trì máy in, modem, thiết bị đầu cuối từ xa.
8. Lưu trữ các hệ thống file.
9. Bảo trì các dịch vụ mạng, thư điện tử và các dịch vụ truyền thông khác.

VII. Môi trường Shell và lập trình Shell

VII. 1. Các khái niệm cơ bản

UNIX shell bao gồm bộ biên dịch lệnh và ngôn ngữ lập trình. Có ba loại shell trong các bản UNIX thương phẩm:

- Bourne shell của Steven Bourne đơn giản và hiệu quả, là shell mặc định trong đa số các hệ UNIX (hoặc có thể gọi bởi **sh**).
- C shell của Bill Joy ở trường đại học Berkeley, giống như Bourne shell, nhưng hoàn toàn tuân theo các quy tắc của ngôn ngữ lập trình C/C++ và được bổ sung thêm các đặc điểm như bí danh, lịch sử lệnh, v.v.. và được gọi bởi lệnh **csh**.
- Korn shell của David F. Korn kết hợp Bourne shell và C shell, nhưng bổ sung thêm các đặc điểm riêng. Lệnh gọi là **ksh**.

Phần này sẽ trình bày cả ba loại shell. Với Korn shell sẽ có chú thích riêng.

VII. 1. a. Các lệnh trong shell

▪ *Lệnh đơn giản*

Là một chuỗi từ được phân cách bởi khoảng trắng. Từ đầu là lệnh UNIX hoặc lệnh nội tại của shell. Các từ khác, hoặc là tên file cần được xử lý, hoặc là các đối số, làm thay đổi hành vi của lệnh.

Dạng cú pháp chung của lệnh shell cũng giống như của lệnh hệ thống UNIX. Đối với C shell, còn có các lệnh hầu như được lấy hoàn toàn từ trong ngôn ngữ lập trình C/C++.

Lệnh trong môi trường UNIX thường xuyên trả lại một giá trị số gọi là trạng thái thoát, hay giá trị trả về, khi shell kết thúc. Nếu lệnh hoàn thành, trạng thái thoát sẽ bằng không. Nếu trạng thái thoát khác không, thì điều đó có nghĩa là lệnh gặp điều

kiện không bình thường hoặc là kết thúc bất thường. Với phát biểu **if** giá trị 0 tương ứng với điều kiện đúng.

Khi lệnh kết thúc bất thường, nó trả về giá trị (128 + trạng thái), với trạng thái là giá trị gây ra kết thúc bất bình thường. Ví dụ nếu nhấn **Del** để kết thúc lệnh, giá trị trả về sẽ là 130 (tức là 128+02).

▪ Đường ống

Bao gồm một hay nhiều lệnh đơn giản được phân cách bởi ký hiệu "|". Đầu ra của mỗi lệnh được gửi đến đầu vào của lệnh kế tiếp. Mỗi lệnh được thực hiện như một tiến trình riêng rẽ. Trạng thái thoát của đường ống là trạng thái thoát của lệnh cuối.

▪ Chuỗi lệnh

Bao gồm một hoặc nhiều lệnh đơn giản (hay đường ống) được phân cách bởi ";", "&", "&&", "|". Trong một danh sách lệnh, các dấu phân cách được xử lý từ trái sang phải với độ ưu tiên theo các thứ tự như sau:

```
&&  ->  |
;    ->  &
```

☞ Dấu ";" gây ra thực hiện tuần tự các lệnh trong danh sách lệnh. Shell đợi mỗi lệnh hoàn thành công việc trước khi thực hiện lệnh kế tiếp.

☞ Dấu "&" gây ra thực hiện bất đồng bộ các lệnh trong danh sách lệnh. Shell thực hiện lệnh sau mà không đợi lệnh trước được hoàn thành.

☞ Dấu "&&" gây ra lệnh sau được thực hiện chỉ khi lệnh trước trả về giá trị 0.

Ví dụ:

```
$ mail -s && echo You have mail
```

sẽ trình bày thông báo "You have mail" chỉ khi lệnh **mail -s** trả về giá trị 0.

☞ Dấu "|" làm lệnh sau được thực hiện chỉ khi lệnh trước trả về giá trị khác 0.

Ví dụ:

```
$ mail -s || echo No mail
```

sẽ trình bày thông báo “No mail” chỉ khi lệnh **mail -s** trả về giá trị khác 0.

- *Nhóm lệnh*

Có thể nhóm một chuỗi lệnh trong dấu ngoặc đơn “()”. Vào chuẩn và ra chuẩn có thể đổi hướng cho toàn nhóm. Ví dụ:

```
$ ls; who; date > status      -> chỉ có ra chuẩn của lệnh date
                             được gửi tới file status.
$ (ls; who; date) > status    -> ra chuẩn của cả ba lệnh được gửi
                             tới file status.
$ (ls; who; date) > status & -> cả ba lệnh đều chạy ở chế độ nền.
```

- *Thay thế lệnh*

Khi shell gặp một lệnh nằm trong các dấu “` `”, lệnh đó sẽ được thực hiện, và kết quả ra của nó trở thành đầu vào của lệnh khác, hay được gán cho biến trong sự thay thế lệnh. Ví dụ:

```
$here=`pwd`
```

thực hiện lệnh **pwd** trước, sau đó gán cho biến **here**.

- *Chú thích*

Đưa chú thích, bình luận vào trong shell bằng cách sử dụng dấu thăng “#” ở đầu dòng. Các dòng được bắt đầu bằng ký hiệu này sẽ không được shell biên dịch. Cách viết này là thông dụng đối với nhiều ngôn ngữ lập trình quen thuộc.

- *Biên dịch khoảng trắng*

Sử dụng biến phân cách vùng IFS để thay đổi ký hiệu phân cách khoảng trắng. Ví dụ sau dùng dấu hai chấm “:” thay cho khoảng trắng để phân biệt các trường như đã được dùng trong file `/etc/passwd`:

```
$IFS=:
```

- *Dấu nháy đơn*

Shell không diễn dịch các dòng ký tự giữa hai dấu nháy đơn. Ví dụ:

```
$ echo '$TERM'
$TERM
```

- *Dấu nháy đôi*

Shell không diễn dịch các dòng ký tự giữa hai dấu nháy đôi, ngoại trừ các dấu "\$, \, ' ". Ví dụ:

```
$ name= "Ng Van A"
echo $name
Ng Van A
echo "$name"
Ng Van A
```

Dấu "\" ngăn ngừa shell diễn dịch ký tự đứng trực tiếp sau nó. Ta đã từng dùng cách này khi tìm hiểu lệnh **uucp** trong chương *Sử dụng các dịch vụ Internet*. Ta có thể lấy lại lệnh **echo** làm ví dụ như sau:

```
echo "Dua vào ten mat hang nhap kho :\nSoluong :"
```

```
Dua vào ten mat hang nhap kho :
Soluong :
```

Ký hiệu "\n" chỉ quy định bắt đầu dòng mới, hoàn toàn không có tác động lên kết quả xử lý của shell, thông qua lệnh **echo** của nó, như được thấy ở trên.

- *Tên file*

Có thể là tên đầy đủ hay tên viết tắt. Trước khi thực hiện lệnh, shell quét tên viết tắt. Có các viết tắt được quy định như sau:

Ký tự	Ý nghĩa của việc viết tắt
*	tương ứng với mọi ký tự
?	tương ứng với một ký tự
[xyz]	tương ứng với ba ký tự x,y,z
[a-z]	tương ứng với các ký tự từ a đến z
[!a-z]	tương ứng với các ký tự khác a đến z

▪ *Lưu nhắc*

Người ta sử dụng biến PS1 để thay đổi dấu nhắc ở dòng lệnh, biến PS2 - đối với dấu nhắc của lệnh con trong chương trình nào có dùng biến PS2.

VII. 1. b. Sự thực hiện lệnh

Shell thực hiện lệnh theo thứ tự ưu tiên như sau:

- Giải quyết các tên file, dấu nháy, sự thay thế lệnh, sự thay thế biến, và đổi hướng ra vào .
- Thực hiện các lệnh nội tại.
- Tên lệnh được so sánh với tên của các hàm đã được định nghĩa. Nếu tương ứng, các hàm sẽ được thực hiện
- Nếu lệnh không phải là lệnh nội tại hay hàm một tiến trình mới được tạo và shell sẽ thực hiện lệnh bởi gọi hàm **exec**.

Shell sử dụng biến PATH để tìm lệnh. Giá trị mặc định là thư mục hiện thời, kể đó là thư mục /bin, sau đó là /usr/bin.

VII. 1. c. Tiến trình nền

Shell có thể chạy ở chế độ nền hay tương tác. Khi lệnh chạy ở chế độ nền shell trở lại dấu nhắc ngay lập tức. Người dùng có thể đưa tiếp vào một lệnh khác.

Ví dụ:

```
$ sort large_file >file_out &
1222
```

Khi lệnh nền chạy định danh tiến trình (PID) của nó sẽ được trình bày. Cũng có thể sử dụng lệnh **ps** để trình bày PID và sử dụng lệnh **kill** để hủy bỏ tiến trình.

Ví dụ:

```
$ kill -2 1222
```

Nếu muốn lệnh chạy ở chế độ nền bỏ qua tín hiệu *hang up* thì ta có thể sử dụng lệnh **\$ nohupsort**

Ví dụ:

```
$ nohupsort database > db.out &  
1245
```

Có thể sử dụng lệnh **wait** để đợi tiến trình nền thực hiện xong và nhận giá trị trả về của nó. Ví dụ:

```
$ wait 1245
```

Ngoài ra, đối với **ksh** còn có cơ cấu kiểm soát công việc:

Ví dụ:

```
$ sort large > sort.out &  
[1] 1923
```

[1] là chỉ số công việc.

VII. 1. d. Gọi lại lệnh đã dùng

Trong thực tế, nhiều khi người dùng phải lặp lại các lệnh đã được gọi. **ksh** đưa các dòng lệnh vào file `/home/.ksh_history` với số lệnh cực đại mặc định là 128. Tuy nhiên có thể đổi biến `HISTFILE` để **ksh** dùng một file khác, `/home/lichsulenh` chẳng hạn. Số dòng lệnh cực đại có thể đưa vào file mới (ví dụ: `/home/lichsulenh`) được đặt bởi biến `HISTSIZE`.

Để xem lại các lệnh đã dùng, ta cần dùng lệnh **fc** theo cú pháp: **\$ fc -l [mmm [-nnn]]**, trong đó tùy chọn **l** nghĩa là list - liệt kê, **mmm, nnn** - số thứ tự

dòng lệnh. Chú ý: lệnh **fc** dùng để hiệu đính lại lệnh, dùng biến **FCEDIT** để chọn chương trình soạn thảo (**vi**, **sed**, ...).

Có thể dùng lệnh **\$ history** để có kết quả tương tự. Lệnh này hay được dùng cả trong C shell.

Muốn lặp lại các lệnh đã gọi trước trong một phiên làm việc, ta dùng lệnh **r** theo cú pháp: **\$ r [-] n**, trong đó, nếu không có tùy chọn - lặp lại lệnh cuối cùng; **n** - thứ tự của lệnh; có dấu “-” - số thứ tự ngược từ cuối trở về trước. Tùy chọn “-” có thể không hoạt động trong một số bản UNIX.

VII. 2. Xử lý tiến trình

Hệ thống UNIX dùng các tiến trình như là phương tiện để chạy các chương trình. Mỗi lệnh mà người dùng nhập vào tạo ra một tiến trình. Một tiến trình là một chương trình đang chạy, kết hợp với các dữ liệu liên quan. Ngoài ra, tiến trình còn kết hợp cả thông tin về người dùng và môi trường hệ thống. Mỗi tiến trình có một mã số nhận dạng (PID) riêng liên kết với nó và hệ điều hành sẽ dùng mã số này để phân biệt các tiến trình.

Tiến trình tạo ra một tiến trình khác được gọi là tiến trình cha. Các tiến trình được tạo ra bởi các tiến trình cha được gọi là các tiến trình con. Tiến trình con nhận biết cha nó qua PID của tiến trình cha.

Mỗi tiến trình cũng có một ID nhóm tiến trình liên kết với nó, hợp thành một vỏ đăng ký. Một nhóm tiến trình bao gồm tất cả các tiến trình có cùng ID nhóm tiến trình. Mỗi vỏ đăng ký ở mỗi thời điểm chỉ có thể có một ID nhóm tiến trình liên kết với nó. Nếu một tiến trình của người dùng không nằm trong ID của nhóm tiến trình liên kết với vỏ đăng ký của người dùng, thì tiến trình đó là một tiến trình nền.

Cùng được liên kết với mỗi tiến trình là ID của người dùng thực và hiệu dụng, cũng như ID của nhóm thực và hiệu dụng. Các ID của người dùng thực và tiến trình thực được thiết lập bởi tiến trình đăng ký và được tất cả các tiến trình mà người dùng tạo ra thừa hưởng. ID của người dùng hiệu dụng (SUID) thường thì giống với ID của người dùng thực nhưng có thể được thay đổi bởi các chương trình có thể thiết lập ID người dùng (đó là **su**). Cũng tương tự như vậy, các ID của nhóm thực và hiệu dụng thường cũng giống nhau, nhưng ID của nhóm hiệu dụng (SGID) có thể được

thay đổi bởi các chương trình có khả năng thiết lập ID nhóm (ví dụ, **mail**). Các ID của nhóm hiệu dụng và người dùng hiệu dụng của một tiến trình được dùng để xác định quyền đăng nhập file và thư mục của một tiến trình (đọc, ghi, thực hiện hay tìm kiếm).

Các tiến trình được tạo bởi lệnh gọi hệ thống **fork** hay **exec** (một lệnh gọi hệ thống là một lời gọi của chương trình gửi đến hệ điều hành để yêu cầu các dịch vụ hệ thống).

VII. 2. a. Lệnh gọi hệ thống fork

Tiến trình cha tạo ra một tiến trình con bằng một lệnh gọi hệ thống **fork**.

Lệnh **fork** tạo ra một bản sao của tiến trình gốc giống như tiến trình cha, ngoại trừ giá trị PID là mới cho tiến trình con. Tiến trình cha có thể tạm dừng sự thực hiện của nó cho đến khi tiến trình con kết thúc. Thông thường, shell thực thi lệnh bằng cách tạo tiến trình con, sau đó đợi tiến trình con hoàn thành trước khi tạo dấu nhắc kế tiếp. Tiến trình con thừa kế các thông tin môi trường sau từ tiến trình cha:

- PID của tiến trình cha
- ID nhóm mà tiến trình cha là thành viên
- Các file mở
- Thư mục làm việc
- Mặt nạ tạo file (umask)
- UID của người dùng đang chạy tiến trình
- GID của nhóm mà người dùng đang chạy tiến trình tham gia
- Giới hạn kích thước file được tạo ra
- Giới hạn bộ nhớ tối đa được dùng cho tiến trình
- Các tác động đặt tín hiệu (dạng dữ liệu lệnh đặc biệt có tác dụng điều khiển hệ thống)
- Các biến

Khi tiến trình con hoàn thành, kiểm soát hệ thống sẽ được đưa trở lại tiến trình cha. Các tài nguyên được cấp phát sẽ được giải phóng. Tiến trình cha tiếp tục được chạy cho đến khi kết thúc nếu hệ điều hành không gọi một lệnh **fork** nữa để tạo một tiến trình con khác.

VII. 2. b. Lệnh gọi hệ thống `exec`

Lệnh gọi hệ thống **exec** nạp chương trình vào vùng bộ nhớ của tiến trình hiện thời. Không giống như **fork**, **exec** không tạo tiến trình mới.

VII. 2. c. Tiến trình login

Các sự kiện xảy ra khi đăng nhập sẽ mô tả cách thức làm việc của **fork** và **exec**. Tiến trình `/etc/init` (PID 1) chạy khi hệ thống khởi động và sẽ tiếp tục chạy cho đến khi ta tắt điện dừng máy. Tiến trình `/etc/init` là một tiến trình **daemon**.

`/etc/init` xem xét file cấu hình hệ thống `/etc/inittab`. File này chứa các dòng lệnh phải thực hiện với các mức hệ thống khác nhau. Ví dụ như dòng vào phải đưa ra kết quả đăng nhập hoặc đặt chế độ cho thiết bị đầu cuối:

```
11:234:respawn:/etc/getty tty11 9600
```

respawn chỉ thị là mỗi khi **getty** chết nó sẽ được khởi động lại bởi `/etc/init`.

VII. 2. d. Thực hiện lệnh đơn giản

Khi đưa vào lệnh đơn giản, shell **fork** một shell khác và sau đó **exec** lệnh. Shell cha đợi cho đến khi tiến trình con hoàn thành

VII. 2. e. Thực hiện chuỗi lệnh

Khi đưa vào chuỗi lệnh, shell **fork** một shell con và sau đó gọi **exec** để thực hiện lệnh đầu tiên. Khi lệnh đầu hoàn thành, shell **fork** tiếp một shell khác và sau đó gọi lệnh **exec** để thực hiện lệnh thứ hai, v.v...

VII. 2. f. Thực hiện nhóm lệnh

Khi đưa vào nhóm lệnh, shell cũng **fork** một shell con để thực hiện các lệnh. Shell con này gọi **exec** để thực hiện lệnh đầu tiên. Khi lệnh đầu hoàn thành, shell **fork** tiếp một shell khác và sau đó gọi lệnh **exec** để thực hiện lệnh thứ hai, v.v..

VII. 2. g. Thực hiện lệnh đường ống

Khi đoán nhận đường ống “|” shell khởi động mỗi tiến trình trong dòng lệnh và thiết lập liên lạc giữa chúng thông qua một file đường ống đặc biệt. Các lệnh trong đường ống được thực hiện từ trái qua phải.

VII. 2. h. Thực hiện lệnh nền

Khi đưa vào lệnh nền, shell **fork** shell con, sau đó gọi lệnh **exec** để thực hiện lệnh. Kiểm soát được trả lại tiến trình cha ngay lập tức. Shell khi đó có thể thực hiện các lệnh khác.

VII. 3. Lệnh nội tại của shell

Các lệnh nội tại là một phần của shell. Có các lệnh sau:

. file	Shell hiện hành đọc và biên dịch lệnh trong file.
alias [name [=value]] (ksh)	Lệnh alias cho phép đổi tên các lệnh trong shell. Ví dụ: \$ alias dir="ls " \$ alias opt="-CF" \$ dir opt bin/ unit01.n unit02.n status *
bg %job_index	Đặt công việc ở chế độ nền
break	Thoát khỏi các cấu trúc for , while , ...
cd [dir]	Chuyển thư mục
continue	Lặp các cấu trúc
echo [args]	Hiển thị nội dung đối số
exec [command]	Gọi lệnh thực hiện
exit	Kết thúc shell
export	Đưa các biến môi trường thành các biến chung
fc	Hiển thị file lịch sử lệnh
fg %job_index	Đặt công việc ở chế độ tương tác
kill PID	Xóa đi một tiến trình có PID đã nêu

newgrp [<i>group</i>]	Đổi nhóm của người sử dụng	
read <i>variable</i>	Đọc dữ liệu từ bàn phím để gán cho biến	
pwd	Hiển thị thư mục hiện thời	
return	Trả giá trị về hàm gọi	
set	Đặt biến môi trường	
test [<i>expression</i>]	Kiểm tra biểu thức. Ví dụ:	
-r <i>file</i>	Đúng nếu file tồn tại và có thể đọc v.v..	
ulimit [<i>size</i>]	Định kích thước tối đa của file được tạo ra	
unalias <i>name</i>	Xóa alias (bí danh) của một tên	
umask [<i>mask</i>]	Định mặt nạ chủ quyền file	
Giá trị mặt nạ	Chủ quyền file	Chủ quyền thư mục
000	-rw-rw-rw-	drwxrwxrwx
002	-rw-rw-r--	drwxrwxr-x
007	-rw-rw----	drwxrwx---
022	-rw-r--r--	drwxr-x---
027	-rw-r-----	drwxr-x---
077	-rw-----	drwx-----
unset	Xóa các biến môi trường	
wait [<i>PID</i>]	Đợi tiến trình có PID hoàn thành, rồi báo cáo trạng thái kết thúc của nó.	

VII. 4. Biến và tham số shell

Các tham số shell được sử dụng chứa các giá trị của shell hiện hành hay các shell con để sử dụng sau đó. Có các loại tham số sau:

- Tham số vị trí
- Tham số đặc biệt
- Tham số được đặt tên (biến)

VII. 4. a. Tham số vị trí

Tham số vị trí là các giá trị truyền qua khi shell hay shell con được gọi. Có thể đặt các tham số vị trí như các đối số của shell script và các tham số này được đánh số 1,2, v.v.. Tên của lệnh hiện hành được tham chiếu bởi **\$0**. Các tham số khác sẽ là **\$1**, **\$2**, v.v..., ký hiệu chung là **\$i**. Các tham số chạy được ký hiệu là **\${i}**.

Trong C shell, tham số vị trí được viết dưới dạng `$argv[0]`, `$argv[0]`, v. v. , `$argv[i]`.

VII. 4. b. Tham số đặc biệt

Tham số đặc biệt được đặt tự động khi shell được gọi. Chúng có một ý nghĩa đặc biệt không thể thay đổi.

Biến	Ý nghĩa
<code>\$*</code>	Tất cả các tham số vị trí như " <code>\$1 \$2 \$3 \$4...</code> " có trên dòng lệnh. Trong C shell là <code>\$argv</code>
<code>\$@</code>	Tham số vị trí như " <code>\$1</code> ", " <code>\$2</code> ", " <code>\$3</code> ", " <code>\$4</code> " v.v.. theo thứ tự được chuyển đến thủ tục shell
<code>\$#</code>	Số các tham số vị trí trên dòng lệnh được truyền đến shell. Trong C shell là <code>\$#argv</code>
<code>\$?</code>	Giá trị trả về (Mã thoát) của lệnh được thực hiện cuối cùng trước đó
<code>\$\$</code>	Chỉ số tiến trình (PID) của shell hiện hành
<code>\$!</code>	Chỉ số của lệnh nền cuối cùng được kéo bởi shell hiện hành

VII. 4. c. Tham số được đặt tên

Tham số được đặt tên được gọi là biến shell và phải bắt đầu với một chữ, sau đó có thể là chữ hay số. Để đặt biến sử dụng dấu "=" (trước và sau dấu "=" không để khoảng trống). Ví dụ:

```
$ ten="Ng Van A"
$ echo $ten
    Ng Van A
```

Dấu "{}" phải được sử dụng với tên biến theo sau bởi chữ hay số không phải là một phần của tên biến. Ví dụ, nếu viết:

```
$ filename=chapt
$ echo $filename0
    Invalid argument or command not found
```

vì hệ thống sẽ không hiểu được. Ta phải viết như sau:

```
$ filename=chapt
$ echo ${filename}0
chapt0
```

Các biến được đặt bởi Korn shell (**ksh**)

<code>_</code> (dấu gạch dưới)	Biến tạm thời dùng trong Korn shell để chỉ giá trị của đối số cuối cùng trong câu lệnh ngay trước đó.
<code>ERRNO</code>	Giá trị sai của lần gọi hệ thống cuối
<code>LINENO</code>	Chỉ số dòng hiện thời trong shell script
<code>OLDPWD</code>	Giữ thư mục hiện thời cũ được đặt bởi lệnh <code>cd</code>
<code>PPID</code>	Chỉ số tiến trình của shell cha
<code>PWD</code>	Thư mục hiện thời

Các biến được sử dụng bởi Korn shell (**ksh**). Các biến này thường được đặt bởi người quản trị hệ thống trong file `/etc/profile` hay bởi người dùng trong file `.profile` trong thư mục home của mình.

<code>CDPATH</code>	Đường dẫn để tìm lệnh <code>cd</code>
<code>HOME</code>	Thư mục "home"
<code>PATH</code>	Đường dẫn để tìm lệnh
<code>PS1</code>	Dấu nhắc thứ nhất mặc định " <code>\$</code> ", dùng làm dấu nhắc tại đầu dòng lệnh
<code>PS2</code>	Dấu nhắc thứ hai mặc định " <code>></code> ", dùng làm dấu nhắc tại đầu dòng lệnh cấp dưới của các lệnh, tiện ích có thực đơn phân nhánh
<code>TERM</code>	Loại Terminal (thiết bị đầu cuối)

VII. 5. Cấu trúc lập trình shell

VII. 5. a. Vòng lặp `for`

Cú pháp:

```
for variable in from
do
    <danh sách lệnh>
done

for variable
do
    <danh sách lệnh>
done
```

Ví dụ, ta thử ngay từ bàn phím như sau với giả thiết đã có sẵn các file unit??.n:

```
$ ls unit??.n
unit01.n unit02.n unit04.n unit12.n
$ for i in unit??.n
>do
>spell $i > ${i}_sp
>done
$ls unit??*
unit01.n unit02.n unit04.n unit12.n
unit01.n_sp unit02.n_sp unit04.n_sp unit12.n_sp
```

Hay ta có thể thu thập các file ra chuẩn vào một file duy nhất:

```
for i in unit??.n
do
    echo "$i:"
    spell $i
    echo "\n"
done > spell.out
```

VII. 5. b. Điều kiện while

Cú pháp:

```
while command
do
    <danh sách lệnh>
done
```

Vòng lặp thực hiện danh sách lệnh chừng nào trạng thái thoát của lệnh command còn bằng 0. Nếu ngay lần đầu lệnh đã trả về giá trị khác 0 thì danh sách lệnh sẽ không được thực hiện. Ví dụ:

```
while who | grep NVA > /dev/null
do
    sleep 10
done
echo "NVA is log off " > /etc/w_NVA
```

VII. 5. c. Điều kiện until

Cú pháp:

```
until command
do
    <danh sách lệnh>
done
```

Thực hiện danh sách lệnh, chừng nào trạng thái thoát của lệnh command còn khác 0. Nếu ngay lần đầu lệnh đã trả về giá trị bằng 0 thì danh sách lệnh sẽ không được thực hiện. Ví dụ:

```
until who | grep NVA > /dev/null
do
```



```
    sleep 10
done
echo "NVA is log on"
```

VII. 5. d. Kiểm tra điều kiện if

Cú pháp:

```
if command
then
    <danh sách lệnh>
fi

if command
then
    <danh sách lệnh 1>
else
    <danh sách lệnh 2>
fi

if command1
then
    <danh sách lệnh a>
elif command2
then
    <danh sách lệnh b>
fi
```

Nếu lệnh trả lại giá trị 0, danh sách lệnh sau **then** sẽ được thực hiện.

Ví dụ:

```
if who | grep NVA > /dev/null
then
```

```
        echo "NVA is log on"
    fi

    if [[test]-f status ]
    then
        echo " status la file thuong"
    else
        echo " status khong phai la file thuong"
    fi

    if [ -f status ]
    then
        echo " status la file thuong"
    elif [ -d status ]
    then
        echo " status la thu muc"
    fi
```

VII. 5. e. Lệnh tình huống case

Cú pháp:

```
case $variable in
    pattern1)
        <đanh sách lệnh 1>
        ;;
    pattern2)
        <đanh sách lệnh 2>
        ;;
    ...
esac
```

Lệnh **case** sẽ thực hiện danh sách lệnh liên hệ với pattern đầu tiên thích ứng với từ. Mỗi pattern được kết thúc bởi dấu **;;**. Ví dụ:

```
echo " Enter command : \c"
read cmd
case $cmd in
  ls)
    /bin/ls ;;
  who)
    /bin/who;;
  date)
    /bin/date;;
  *)
    echo " Error command";;
esac
```

VII. 5.f. Hàm

Cú pháp:

```
name()
{
    < các lệnh >
}

function
{
    < các lệnh >
}
```

Các hàm được đọc vào shell hiện hành. Để gọi hàm chỉ cần gọi bằng tên - chú ý tên hàm phải khác với tên của các lệnh UNIX đã tồn tại. Shell thực hiện các lệnh trong dấu ngoặc nhọn "{ }" khi hàm được gọi. Hàm có thể truy xuất ra tập hợp biến của shell hiện hành.

Ví dụ:

```
dir()
{
    /bin/ls -C
}
```

VII. 6. Shell script

UNIX được thiết kế để tạo ra hàng loạt các lệnh nhỏ nhằm thực hiện những tác vụ nhất định. Để đơn giản và tự động hóa các thao tác của người dùng, người ta dùng những *shell script*. Shell script là một file chứa các lệnh nội tại, hàm và lệnh UNIX. Shell script thường sử dụng khi một chuỗi lệnh phải thực hiện nhiều hơn một lần. Để tạo shell script ta phải sử dụng một chương trình soạn thảo, ví dụ như **vi**, **sed**, v.v..., để đưa các lệnh muốn thực hiện vào nội dung file.

Để thực hiện shell script có thể làm như sau:

- Đưa nó vào trong thư mục **bin**, ví dụ: **/oracle/bin** chẳng hạn.
- Thay đổi chủ quyền file đến chế độ thực hiện được bởi lệnh **chmod +x script_name**. Sau đó chạy shell script như một lệnh bình thường.
- hoặc sử dụng lệnh **sh**: **\$ sh myprog**

Tên của shell script không nên trùng với các lệnh UNIX, lệnh nội tại, hàm. Vì shell script luôn được gọi thực hiện đầu tiên.

Có thể sử dụng lệnh chấm “.” để thực hiện chương trình trực tiếp shell script trong môi trường của nó. Ví dụ: **\$. .profile** (Đọc và thực hiện các lệnh trong file **.profile**).

• Viết vài ví dụ các shell script đơn giản:

- *Tạo script cho lệnh in*

Giả sử ta đặt tên cho lệnh này là **psp**. Dùng **vi** tạo file **psp** chứa dòng lệnh sau:

```
lp -o nobanner -T postcrip -d psdoc $1
```

lp - lệnh in; **nobanner** - không in trang banner (chứa thông tin về tên máy in và tên người dùng); **poscript** - loại máy in postscript; **psdoc** - tên máy in; **\$1** chỉ ra rằng lệnh này được thực hiện với tên file đầu tiên bạn đưa vào sau lệnh **psp**.

- *Tạo script để chép các file vào đĩa mềm*

Giả sử ta đặt tên là **toA**. Nội dung file này được soạn thảo bằng **vi** như sau:

```
find $* -print | cpio -ocv -0 /dev/rdisk /f0t
```

Lệnh **find** sẽ tìm tới một điểm bắt đầu trên hệ thống file và chép tất cả các file và thư mục dưới nó sang đĩa mềm đầu tiên, và nếu không đủ chỗ, sang các đĩa mềm tiếp theo. Ví dụ, nếu bạn cần lưu nội dung của file **/siba/giaodich** ra đĩa mềm, thì chỉ cần gõ lệnh:

```
toA /siba/giaodich
```

- *Tạo script để khôi phục các file từ đĩa mềm vừa chép*

Giả sử ta đặt tên là **fromA**. Nội dung file được soạn thảo bằng **vi** như sau:

```
cpio -icdv -1 /dev/rdisk /f0t
```

Khi thực hiện lệnh này chỉ cần đánh **fromA**, không cần tùy chọn nào. Sau khi đọc và chép hết đĩa thứ nhất, máy sẽ yêu cầu đưa vào các đĩa tiếp theo, nếu có.

- *Tạo shell cho biết ngày giờ hệ thống, bao nhiêu người dùng và cụ thể là ai đang đang nhập hệ thống, dung lượng đĩa còn trống hiện thời*

Giả sử ta đặt tên cho script này là **sys-info**:

```
# file sys-info cho biet:
# ngay, gio he thong
# so luong user tham gia he thong
# ten cac user tham gia he thong hien thoi
# dung luong dia trong trong thu muc /export
```

```

echo "Hom nay la ngay : `date`"
echo "Chuc ban mot ngay lam viec tot dep!"
NUM=`who | wc -l`
echo "Nhưng người dùng hôm nay vào mạng : $NUM"
echo "Người dùng      O đầu      Lúc nào"
who
echo "Dùng lương địa con trong thư mục /export:"
df -v /export
exit 0

```

VII. 7. Bổ sung về shell

VII. 7. a. So sánh giữa Bourne, Korn và C shell

Các mục so sánh	Bourne	C	Korn
Biến môi trường	Có	Có	Có
Biến chung	Có	Có	Có
Thay thế tên file	Có	Có	Có
Tên viết tắt (Wildcard)	Có	Có	Có
Hoàn thành lệnh	Không	Có	Không
Lịch sử lệnh	Không	Có	Có
Hiệu đính lệnh	Không	Có	Có
Điều khiển thực hiện công việc	Không	Có	Có
Thay thế lệnh	Có	Có	Có
Đường ống	Có	Có	Có
Đổi hướng vào/ra	Có	Có	Có
Hàm shell	Có	Không	Có
Bí danh (Aliases)	Không	Có	Có
Kiểm tra phát âm	Không	Có	Không
Bộ số học cố sẵn	Không	Có	Có
Mảng cố sẵn	Không	Không	Có

VII. 7. b. Một vài lời khuyên khi sử dụng shell

- Nên sử dụng trước hết các lệnh nội tại của shell.
- Nên dùng ký hiệu trên dòng đầu tiên của shell script nhằm chỉ rõ shell script được viết để chạy trên shell nào, ví dụ cho Bourne shell, ta viết như sau: `#!/bin/sh`.
- Tận dụng các ưu điểm của lệnh:

```
sort file          hoặc          cat file | sort
```

- Nên chọn các đối hướng ra vào hơn là dùng đường ống. Ví dụ:

```
$ lp > file  nhanh hơn $ cat file | lp
```

- Nên tính toán số byte cần xử lý khi sử dụng đường ống để tăng tốc độ thực hiện lệnh. Ví dụ:

```
$ sort file | grep pattern    : phải xử lý nhiều byte hơn
$ grep pattern file | sort    : số byte phải xử lý ít hơn
```

- Nên sử dụng các hàm hơn là dùng các shell con.
- Nên sử dụng lệnh "." để gọi các shell con có thay đổi biến.
- Nên sử dụng sự thay thế lệnh hơn là đường ống.
- Nếu có thể, nên chỉ đối hướng ra vào chuẩn một lần ở cuối vòng lặp. Ví dụ:

```
for i in 01020304
do
    sort unit${i} >> temp
done > temp
```

- Nên sử dụng đường ống thay vì tạo các file tạm thời.
- Nên chọn lựa các lệnh UNIX hiệu quả hơn trong từng hoàn cảnh. Ví dụ:

```
$ find -type d -print    : nhanh hơn
$ ls -lR | grep "rd"     : chậm hơn
```

VII. 7. c. Các kỹ thuật shell nâng cao

Sau đây ta xét một vài mẫu ứng dụng shell để tự động hóa việc quản trị hệ thống. Dòng chú thích đầu tiên cho biết shell script được viết cho loại shell nào, ví dụ, script ngay sau đây được viết với Bourne shell vì ta thấy `#!/bin/sh`. Thêm nữa, dấu `\` nếu được viết đơn lẻ ở vị trí tận cùng bên phải chỉ có nghĩa dòng lệnh vẫn chưa kết thúc và còn được viết tiếp. Xin đừng lẫn lộn với cách báo cho shell không biên dịch một ký tự `c` nào đó bằng cách viết `\c`.

▪ *Tạo shell kiểm tra file /etc/passwd*

Ví dụ, ta tạo một Bourne shell tên là `ckpwd` như sau để kiểm tra file `passwd` và nội dung của nó:

```
# ! /bin/sh
# ckpwd - check password file (run as root)
umask 077
cd /usr/local/bin/old
# move to dir . where stored passwd
echo ">>Password file check for `date`"; echo " "
echo "*** Accounts without passwords :"
grep '^[^:]*::' /etc/passwd
if [$? -eq 1]      # grep found no matches
then
    echo "None found"
fi
echo " "
echo "*** Non-root UID=0 or GID=0 accounts :"
grep ':00*:' /etc/passwd | awk -F: 'BEGIN {n=0}\
    $1!="root" {print $0 ; n=1} END {{if \
        (n=0) print "None found."}}'
echo " "
sort</etc/passwd > tmp1
sort < opg > tmp2      # opg is the saved copy
```

```

echo "*** Accounts added : "
comm -23 tmp[1-2]      #lines only in /etc/passwd
echo " "
echo "*** Accounts deleted : "
comm -13 tmp[1-2]      #lines only in ./opg
echo " "
rm -f tmp[1-2]
echo "*** Password file protection : "
echo "-rw-r--r--  1  rootsys >>> correct values"
ls -l /etc/passwd
echo " "; echo " >>> End of report."; echo " "
```

Có thể dùng lệnh **diff** để so sánh hai file `/etc/passwd` và `/usr/local/bin/old` thay cho hai lần dùng lệnh **comm**, nhưng ta sẽ mất đi khả năng theo dõi danh sách người dùng được thêm vào hay bị loại bỏ bớt.

▪ *Theo dõi mức độ dùng đĩa cứng*

Chắc hẳn bạn đã từng bị dầy vò vì câu hỏi, liệu ổ đĩa hay thư mục cụ thể nào đó trên máy trong hệ thống của bạn còn trống khoảng bao nhiêu, còn có đủ để ... Shell có tên `chkdisk` sau đây được thiết kế để so sánh tình trạng sử dụng các thư mục cụ thể `/export/oracle`, `/usr/lib`, `/export/ct100ppp` trên đĩa ngày hôm nay so với ngày hôm qua, và lại chuẩn bị cho lần so sánh tiếp vào ngày mai. (Khi khảo nghiệm lại shell script này, bạn có thể thay đổi số lượng và tên các thư mục cho phù hợp với máy tính và hệ điều hành UNIX của bạn).

```

#!/bin/sh
# chkdsk - for disk usage information in comparison
# with yesterday's size
du_it()
{
# $1 = cutoff in blocks for new directories
# $2 = cutoff as block change for old directories
# $3 = starting directory
```

```
# $4 = flags to du
du $4 $3 > du.tmp
cat du.tmp | xargs -n2 cmp_size $1 $2 du.sav
cat du.tmp >> du.log
rm du.tmp
}

umask 077
rm -f du.log du.tmp 2>&1 > /dev/null
if [! -s du.sav]
then
    echo " run_cmp: không tìm thấy file so lieu cu;    \
          chạy du_init."
    exit 1
fi
echo "Bao cao hang ngay su dung dia ngay `date`."
echo " "
df
echo " "
echo "Old\New"
echo "Size\tSize\tDirectory Name"
echo "-----"
du_it 40 250 /export/oracle
du_it 1 1 /usr/lib
du_it 1 1000 /export/ct100ppp/\* -s
echo "-----"
echo " "
mv -f du.log du.sav
exit 0
```

Shell trên dùng hàm `du_it` để thực hiện lệnh `du` và gửi kết quả ra `cmp_size`, nơi thực chất tiến hành so sánh số liệu của ngày hôm qua và hôm nay. Kết quả ra của `du_it` được lưu ở file tạm `du.tmp`, sau sẽ được nối vào file

du.log, tạo file có nội dung mới cho lần so sánh của ngày hôm sau. `cmp_size` là một shell con được gọi trong lệnh **xargs**:

```
xargs [flag] [command [initial-arguments]]
```

Lệnh **xargs** kết hợp các thông số ban đầu xác định bởi `initial-arguments` từ vào chuẩn để chạy lệnh `command` được chỉ định - cũng có thể là một shell, sử dụng biến môi trường `$PATH`. Nếu `command` để trống, lệnh mặc định `/usr/bin/echo` sẽ được thực hiện. Các tùy chọn của **xargs**: `-l number`: `command` sẽ chạy cho `number` dòng không trống các thông số từ vào chuẩn, mặc định là 1; `-i replstr`: chế độ chèn, `command` sẽ chạy cho mỗi dòng từ vào chuẩn, coi cả dòng như một thông số và chèn vào `initial-arguments` mỗi khi gặp `replstr`; `-n number`: `command` sẽ dừng cho đến cực đại `number` thông số; ...

Sau đây là nội dung của shell con `cmp_size`:

```
#!/bin/sh
# cmp_size - shell for comparison old and new directory
# sizes
# $1 (limit)= minimal size for new dirs to be included
#in report
# $2 (dlimit)= minimal size for old dirs to be included
#in report
# $3 (sfile)= pathname for file with yesterday's data
# $4 (csize)= current dirrectory size
# $5 (file)= pathname for directory
# osize=previous size (extracted from sfile)
# diff=size difference between yesterday and today
if [ $# -lt 5 ]
then
echo "Usage cmp_size newlim oldlim data_file size dir"
exit 1
fi
```

```
# save initial parameters
limit=$1
dlimit=$2
sfile=$3
file=$5
csize=$4
# get yesterday data
osize=`grep " $file\$" $sfile | awk '{print \$1}'`
if [ -z "$osize" ]
then
    # it's a new dir.
    if [ $csize -ge $limit ] # report if size >= limit.
    then
        echo "new\t$csize\t$file"
    fi
    exit 0
fi
# compute the size change from yesterday
if [ $osize -eq $csize ]
then
    exit 0
elif [ $osize -gt $csize ]
then
    diff=`expr $osize - $csize`
else
    diff=`expr $csize - $osize`
fi
# check if size change if big enough to report on
if [ $diff -ge $dlimit ]
then
    echo " $osize\t$csize\t$file"
fi
```

▪ *Tạo từ hệ thống file / và lấy thông tin nhanh về hệ thống*

Sau sự cố máy chủ, công việc cài đặt lại hệ điều hành và nhất là các phần mềm ứng dụng đòi hỏi rất nhiều thời gian và công sức. Đó là chưa kể đến việc có thể mất dữ liệu hay các bộ cài đặt phần mềm ứng dụng, có sẵn các thiết bị phần cứng tương ứng (ổ CD chẳng hạn) hay không. Vì vậy, việc lưu trữ hệ thống file root (/) là chú ý đầu tiên của người quản trị hệ thống ngay sau khi bạn cảm thấy hệ thống của bạn bắt đầu vận hành tốt. Nhưng bản thân việc lưu trữ này cũng chiếm không ít thời gian của bạn, vì bạn hoặc phải tốn nhiều dung lượng đĩa để lưu trữ *toàn bộ* hệ thống file, hoặc nếu có kinh nghiệm, bạn phải đi vào tất cả các thư mục con để tìm và lưu trữ các file cần thiết.

Shell `bkup_sys` sau viết bằng C shell giúp bạn rất nhiều bằng cách chỉ tìm các file hệ thống thực sự có thay đổi và lưu trữ như đối với các file thường:

```
# ! /bin/csh
# bkup_sys - lưu trữ các file có thay đổi trong các
# phân chia hệ thống
umask 077
# bảo vệ mọi thư mục tạo ra
set SAVE_DIR="/`hostname`/1/SYS_SAVE"
if (" $1 " != "") set SAVE_DIR="$1"
  set dir_list=`cat /etc/bkup_dir`
  foreach dir ($dir_list)
    echo " Đang làm việc ở $dir ..."
    if (! -d $SAVE_DIR/$dir)
      mkdir -p $SAVE_DIR/$dir
      set files='file $dir/{,.[a-zA-Z]}* | \
        egrep 'text|data' | awk -F: '{print $1}'
      if (" $files " != "") cp -p $file $SAVE_DIR/$dir:t
    end
  end
echo " Lưu trữ các file riêng biệt..."
foreach file ('cat /etc/bkup_files')
  if (" $file:h " == " $file:t ") continue
```

```
# không có đường dẫn đầy đủ
if (" $file:t" == "") continue
# không có tên file
if (! -d $SAVE_DIR/$file:h)
    mkdir -p $SAVE_DIR/$file:h
    cp -p $file:t $SAVE_DIR/$file:h
end
echo " Tất cả đã thực hiện xong."
```

Shell script này thực hiện việc lưu trữ trong hai giai đoạn. Giai đoạn đầu tất cả các file văn bản và nhị phân từ danh sách các thư mục được ghi vào một thư mục định trước; được phân loại bởi lệnh **file**, và lệnh **grep** chọn những file có vẻ là file cấu hình (nếu có lấy thừa file khác còn tốt hơn bỏ sót). Thư mục đích mặc định được đặt cho máy cụ thể của bạn, ví dụ: `/icbv_cn1/SYS_SAVE`. Danh sách thư mục được lấy từ file `/etc/bkup_dirs` mà bạn soạn thảo bằng **vi** chẳng hạn, trong đó chứa `/, /var/spool/cron/crontabs, /etc/default, /etc, v.v..`

Giai đoạn sau hình như không có gì khó hiểu đối với bạn. Các file có tên trong `/etc/bkup_files` được sao lại mà không làm ảnh hưởng tới hệ thống. Nên nhớ, khi sao lại các file đó bằng cách thông thường, xác suất bạn lỡ tay xóa nhầm hay di chuyển chúng đi chỗ khác khác không rất nhiều !! Chỉ cần chú ý, vì ta dùng C shell, **:h** (head) lấy phần đầu (phần thư mục) và **:t** (tail) – phần đuôi (phần tên file và mở rộng của chúng) từ tên file trong biến được chỉ ra (`$file`).

Còn lại vấn đề cuối cùng trong khi dùng shell này: làm sao bạn biết vị trí nguyên thủy của các file đã lưu trữ để khôi phục sau sự cố nghiêm trọng của máy? Trong script có dùng các lệnh **cp** và **mkdir** đều với tùy chọn **p**, nghĩa là giữ nguyên chủ quyền của các file. Thêm nữa, các file được sao lại đều có vị trí tương đối giống như các file nguyên thủy. Để tái tạo lại hệ thống, bạn cần biết các thông tin về bức tranh tổng thể cấu hình ban đầu. Trong Solaris, bạn có thể dùng lệnh `/usr/sbin/prtvtoc` để biết bảng nội dung các đĩa, thư mục cần quan tâm:

```
prtvto c [-fhs] [-t vfstab] [-m mnttab] device
```

với:

f

Báo cáo về dung lượng đĩa trống = (địa chỉ khối đầu, số khối, các partition chưa dùng).

h

Bỏ qua các file tiêu đề (headers - file *.h trong thư viện).

s

Bỏ qua các file tiêu đề (file *.h trong thư viện), trừ các tiêu đề cột (column header).

VIII. Các ngôn ngữ lập trình, các trình soạn thảo

VIII. 1. Ngôn ngữ **awk**

Là ngôn ngữ lập trình và xử lý dữ liệu, **awk** có thể tạo ra những chương trình ngắn để đọc, phân loại, xử lý dữ liệu, hiệu đính các file, v.v., và do Alfred Aho, Peter Weinberger và Brian Kernighan viết ra. Dạng cú pháp của lệnh như sau:

```
awk [-Fc] [-f prog] ['pattern'] [parameters] [filename]
```

Fc - chọn dấu phân cách trường bằng ký tự **c**; **f prog** - đọc file *prog* để lấy các lệnh con thực hiện chương trình; **pattern** - những lệnh con mà **awk** phải dùng để xử lý dữ liệu đầu vào, nếu không được chỉ ra trong *prog*. Các **pattern** phải được viết ngăn cách nhau bằng dấu ";", được đặt trong hai dấu nháy đơn để tránh bị shell biên dịch, và có dạng cú pháp:

pattern {action}

pattern có thể là các biểu thức logic, dùng các toán tử sau:

==	bằng
!=	khác
>	lớn hơn
<	nhỏ hơn
>=	lớn hơn hoặc bằng
<=	nhỏ hơn hoặc bằng
~	chứa
!~	không chứa
&&	và

`| |` hoặc
có thể là các biểu thức dùng các toán tử:

<code>+</code>	cộng
<code>-</code>	trừ
<code>*</code>	nhân
<code>/</code>	chia
<code>%</code>	lấy phần nguyên
<code>(sp)</code>	trùng nhau
<code>++n</code>	tăng thêm n
<code>n++</code>	dùng n sau đó tăng thêm một đơn vị
<code>--n</code>	giảm đi n
<code>n--</code>	dùng n sau đó bớt đi một đơn vị
<code>var+=value</code>	thêm vào biến giá trị bằng value
<code>var-=value</code>	bớt đi từ biến giá trị bằng value
<code>var*=value</code>	nhân thêm vào biến giá trị bằng value
<code>var/=value</code>	chia bớt đi từ biến giá trị bằng value
<code>var^=value</code>	lũy thừa giá trị biến với mũ bằng value

Các biến nội tại của **awk**:

FILENAME	Tên của file đầu vào hiện thời.
FS	Dấu ngăn cách các trường của file đầu vào. Mặc định là Tab hoặc khoảng trống.
NF	Số trường trong bản ghi hiện thời.
NR	Số thứ tự của bản ghi hiện thời.
OFMT	Định dạng kết quả ra cho các số. Mặc định là <code>%.6g</code> (sáu số thập phân).
OFS	Dấu ngăn cách các trường của kết quả đầu ra. Mặc định là khoảng trống.
ORS	Dấu ngăn cách các bản ghi của kết quả đầu ra. Mặc định là dấu bắt đầu dòng mới.
RS	Dấu ngăn cách các bản ghi của đầu vào. Mặc định là dấu bắt đầu dòng mới.

action là chuỗi lệnh con cần thực hiện. Các lệnh đó có thể là:

```
if (expression) statement [else statement]
while (expression) statement
do statement while (expression)
for (expression; expression; expression) statement
for (var in array) statement
break
continue
{[statement]...}
expression
    # thường là variable=expression
print [expression-list] [>expression]
printf format [ ,expression-list] [>expression]
next
    # bỏ qua các pattern còn lại trong dòng đầu vào
exit [expr]
    # bỏ qua phần còn lại của dòng đầu vào, trạng thái thoát là
    expr
```

Ngoài ra, còn có các hàm nội tại:

- **length** (exp)
- **split**(s,a[,d])
- **substr**(a,m[,n])
- **index**(s1,s2)
- **sprintf**(format[,arg ...])
- **getline**

Lệnh **awk** tỏ ra đặc biệt hữu hiệu khi được dùng như thành phần của đường ống, nhằm điều khiển một cách có chọn lựa kết quả ra của các lệnh khác. Trong khía cạnh này, việc dùng **awk** mang tính tổng quát hơn là dùng **grep**. Trước hết, lệnh **awk** rất hiệu quả trong việc lọc ra, và nếu muốn, còn sắp xếp lại các cột trong kết quả ra của một lệnh nào đó. Ta hãy xét ví dụ sau, giả sử giáo viên trong lớp muốn lập danh sách các người dùng trong mạng đang chơi trò chơi điện tử có tên là doom:

```
$ ps -ef | grep "doom" | awk '{print $1}'
```

Lệnh **awk** chỉ hiển thị trường đầu tiên của mỗi dòng ra của lệnh **ps**, được lọc và đi tới nó qua lệnh **grep**. Lệnh này tránh cho ta sự phức tạp và tối nghĩa khi phải dùng kiểu **grep -v grep** cho cùng một mục đích.

Một khi danh sách những người dùng đang chơi trò chơi doom đã được lập, giáo viên có thể xử lý tiếp theo, chẳng hạn:

```
$ (date;ps -ef | grep "doom" | awk '{print $1 "[" $7
    "]"')|sort | uniq) >> doomed.users
```

để phân loại từng người tham gia trò chơi trên theo tiêu chí nào đấy - ở đây là thời gian CPU - và đưa tất cả kết quả trên vào một file có tên `doomed.users`. Trong một mạng đã ổn định, còn có thể đưa thêm các tham số vị trí vào, ví dụ như để chỉ máy (terminal) mà từng người chơi cụ thể đã sử dụng. Với những câu lệnh như trên, người quản trị có thể theo dõi công việc của từng thành viên trong mạng theo các tập hợp tiêu chí xác định.

Lệnh **awk** còn được dùng để lấy tổng tất cả các cột chữ số. Ví dụ, để tìm hệ thống file cục bộ được sở hữu bởi người dùng `tuan` và tính dung lượng đã dùng, ta gõ:

```
# find / -user tuan -fstype ufs -name /dev/* -ls| awk      \
    '{sum+=$7}; END {print " User tuan total disk use = "  \
        sum}'
User tuan total disk use = 4198632
```

Thành phần lệnh **awk** sẽ lần lượt cộng tổng của cột thứ bảy là số byte trong mỗi file được tìm cho đến khi đưa ra con số cuối cùng khi đầu ra của lệnh **find** tìm các file loại `ufs`, có tên từ `/dev/`, và thuộc quyền sở hữu của người dùng `tuan`, được xử lý xong. Thay vì tính tổng số các byte, có thể tính số byte trung bình trên mỗi file được tìm ra bằng cách viết biểu thức `"sum/NR"` thay chỗ của `"sum"` trong mệnh đề **END**. Mẫu số `NR` là biến nội tại của **awk**, chỉ số thứ tự của dòng đầu vào; đối với dòng cuối cùng đồng thời có nghĩa là tổng số dòng đầu vào đã được đọc.

Lệnh **awk** còn được dùng tương tự như lệnh **date** để tạo ra file có tên dựa theo ngày tháng hiện thời. Ví dụ, dòng lệnh sau đặt kết quả ra của một shell script nào đó có tên `sys_doc` vào trong file có tên theo ngày tháng hiện thời và tên máy:

```
$ sys_doc > `date |awk '{print $3 $2 $6}'`.hostname`\
    .sysdoc
```

Ví dụ, nếu lệnh này được thực hiện vào ngày 8 tháng 8 năm 2000 trên máy chủ `icbvtw`, tên file được tạo ra sẽ là `8Aug2000.icbvtw.sysdoc`.

VIII. 2. Dự án GNU

GNU là viết tắt của “GNU is Not UNIX” theo kiểu chơi chữ mà hacker thường dùng, là tên sản phẩm của quỹ Free Software Foundation (FSF), một tổ chức được thành lập với mục đích phát triển và cung cấp - kể cả mã nguồn - hệ điều hành và các chương trình, tiện ích, công cụ phát triển giống như UNIX, nhưng miễn phí cho mọi đối tượng. Các sản phẩm GNU đặc biệt tìm được môi trường phát triển trong những hệ thống Linux. Cho đến ngày nay, hầu hết các nhà cung cấp đều lồng vào hệ điều hành UNIX các sản phẩm đi kèm của GNU. Các sản phẩm đó, ví dụ như **gcc** với các trang trợ giúp của nó, rất thông dụng đối với không chỉ những người phát triển hệ thống hay ứng dụng, mà còn đối với cả những người dùng bình thường. Trong tài liệu này chúng ta còn rất nhiều lần đề cập chi tiết đến các sản phẩm GNU.

VIII. 3. Ngôn ngữ lập trình PERL

PERL, viết tắt của *Practical Extraction and Report Language*, là một ngôn ngữ được biên dịch để quét các file văn bản bất kỳ, nhặt các thông tin từ chúng và in ra các báo cáo. Vì thế PERL có các chức năng xử lý văn bản rất mạnh, nhưng đồng thời cũng là ngôn ngữ lập trình rất tốt cho các mục đích quản trị hệ thống. PERL được cung cấp tự do và chạy được trên nhiều hệ điều hành UNIX, MVS, VMS, MS-DOS, Macintosh, OS/2, Amiga, v.v.. Tác giả đầu tiên của PERL, Larry Wall (tại Carnegie-Mellon University, địa chỉ E-mail: lwall@netlabs.com), khi xây dựng ngôn ngữ này đã đặt ưu tiên hàng đầu cho tính tiện dụng của nó trong thực tế - dễ dùng, hiệu quả và mang tính tổng thể. Tuy mới ra đời không lâu, nhưng theo đánh giá của

nhiều người, PERL kết hợp được các ưu điểm của nhiều ngôn ngữ như C, sed, awk và sh, v.v..

Bản PERL 5 (hiện nay là các phiên bản mới hơn, ví dụ, 5.5), được Larry Wall đưa ra từ tháng 10 năm 1994, mang đến các khả năng hướng đối tượng, có nhiều cấu trúc dữ liệu hơn và các giao diện mới cho các chức năng hệ thống và cơ sở dữ liệu, tiện lợi hơn. Phần mở rộng cho giao diện đồ họa và các thành phần mới khác hiện nay vẫn đang được tiếp tục phát triển.

Có hai đối tượng người dùng chính sử dụng PERL. Người quản trị mạng dùng ngôn ngữ này như công cụ kết dính giữa các hệ thống, cơ sở dữ liệu, tiến trình và người dùng và như công cụ tìm kiếm và báo cáo cho hệ thống. Đối với người phát triển, PERL là ngôn ngữ lập trình dễ dùng hơn C và được ưa thích để lập các bảng điện tử của World Wide Web khi xây dựng các UNIX Web Server, trước hết là cho các phép kiểm tra dữ liệu và các cơ sở dữ liệu đơn giản.

VIII. 3. a. Bắt đầu một PERL Script

PERL script chứa một dãy các khai báo và lệnh. Trong PERL chỉ có dạng báo cáo và các chương trình con là phải khai báo. Tất cả các đối tượng chưa khởi tạo do người dùng tạo ra đều được cho là bắt đầu với giá trị không (null) hoặc 0 cho đến khi chúng được xác định bằng các thao tác bổ trợ như phép gán chẳng hạn. Dãy các lệnh được thực hiện chỉ một lần, không giống như trong script của **sed** hay **awk**, khi dãy lệnh được thực hiện cho mỗi một dòng đầu vào.

Các khai báo có thể đặt ở mọi nơi mà lệnh có thể thấy, không ảnh hưởng tới việc chạy dãy lệnh chính. Chúng chỉ có hiệu lực khi biên dịch. Thông thường người ta đặt các khai báo ở đầu hoặc ở cuối script.

PERL là ngôn ngữ có dạng tự do. Câu chú thích được bắt đầu bằng ký tự "#". Nếu bạn cố dùng dạng chú thích của C: /* */, tùy theo ngữ cảnh chương trình sẽ hiểu hoặc như phép chia, hoặc như tìm kiếm chuỗi ký tự thích hợp.

Cú pháp của PERL như sau:

```
perl [options] filename args
```

Khi bắt đầu chạy, PERL tìm script ở một trong các vị trí sau:

- Lần lượt theo các dòng thông qua tùy chọn `-e` ở chế độ dòng lệnh.
- Trong file được chỉ ra bởi tên file đầu tiên trên dòng lệnh.
- Hoàn toàn thông qua vào chuẩn. Chế độ này chỉ hoạt động một khi không có tên file nào được chỉ ra.

Sau khi tìm được script, PERL sẽ biên dịch nó. Nếu script đúng cú pháp, chương trình sẽ chạy ngay.

Ví dụ:

```
perl -e 's/gopher/World Wide Web/gi' -p -i.bak *.html
```

Lệnh được thực hiện tại dấu nhắc hệ thống của UNIX sẽ chạy một chương trình PERL ngắn được chỉ ra sau `-e`. Chương trình chỉ gồm một lệnh trong ví dụ này sẽ thay thế từ "gopher" bằng "World Wide Web", kiểm tra các file có kết thúc ".html" trong thư mục hiện thời. Nếu có file bất kỳ "tochuc.html" cần có sự thay đổi, file nguyên thủy sẽ được lưu lại với tên "tochuc.html.bak".

VIII. 3. b. Các tùy chọn

Nội dung mục này có thể chưa thật cần thiết đối với người bắt đầu làm quen với PERL. Vì thế, liệt kê đầy đủ các tùy chọn có thể sẽ được trình bày trong phần Phụ lục trong lần tái bản gần đây nhất.

Các tùy chọn được mô tả chỉ bằng một ký tự có thể được viết cùng nhau. Đặc biệt, điều này sẽ rất tiện lợi khi script sử dụng cấu trúc `#!` là cấu trúc chỉ cho phép một đối số. Ví dụ:

```
#!/usr/bin/PERL -spi.bak # tuong tu -s -p -i.bak ...
```

Các tùy chọn gồm:

-0digits

Chỉ các dấu ngăn cách bản ghi "\$/" bằng số bát phân. Nếu không có số, sẽ dùng ký tự không. Có thể kết hợp được. Ví dụ, tìm và hiện tên file kết thúc bằng ký tự không:

```
find . -name '*.bak' -print0 | PERL -n0e unlink -a
```

mở chế độ chia nhỏ tự động khi dùng với `-n` hay `-p`. Lệnh ngầm `split` đối với mảng `@F` được thực hiện đầu tiên trong vòng lặp `while` tạo bởi `-n` hay `-p`:

```
PERL -ane 'print pop (@F), "\n";'
```

tương tự như:

```
while (<>) {
    @F = split (' ');
    print pop (@F), "\n";
}
```

-c

PERL kiểm tra cú pháp của script rồi kết thúc mà không chạy nó.

-d

Chạy script trong bộ gỡ rối của PERL.

-Dnumber

Đặt cờ gỡ rối. `-D14` - theo dõi script chạy thế nào, `-D1024`, liệt kê cây cú pháp được biên dịch, `-D512`, hiện các biểu thức được biên dịch.

-e *commandline*

Có thể dùng để đưa vào từng dòng của script - dùng nhiều lần - tạo script nhiều dòng. Nếu có `-e`, PERL sẽ không tìm kiếm tên file script trong danh sách đối số.

-i*extension*

Chỉ ra file được xử lý bởi cấu trúc `<>` cần được sửa chữa tại chỗ.

-I*directory*

Có thể được dùng chung với `-P` để chỉ chỗ cho bộ tiền xử lý C các file include.

Các thư mục mặc định `/usr/include` và `/usr/lib/perl`.

-l*octnum*

Kích hoạt xử lý tự động kết thúc dòng.

-n

Bắt PERL theo vòng lặp với script, giống như `sed -n` hoặc `awk`:

```
while (<>) {  
    ...      # nội dung của script  
}
```

-p

Bắt PERL theo vòng lặp với script, giống như **sed**:

```
while (<>) {  
    ... # nội dung của script  
} continue {  
    print;  
}
```

Các dòng được hiển thị tự động. Nếu muốn không hiện, phải dùng **-n**. Tùy chọn **-p** sẽ đè được **-n**.

-P

Bắt script phải chạy qua bộ tiền xử lý **C** trước khi được dịch bởi PERL.

-s

Xử lý sơ bộ các khóa trên dòng lệnh ở sau tên script file, nhưng trước tên file đối số (tức là trước a --).

-S

Bắt PERL dùng biến môi trường **PATH** để tìm script (trừ phi tên của script được bắt đầu bằng dấu “\”). Thường được dùng để mô phỏng việc khởi động **#!** trên các máy không hỗ trợ **#!**, theo cách sau:

```
#!/usr/bin/PERL  
eval "exec /usr/bin/PERL -S $0 $*"   
if $running_under_some_shell;
```

-u

Bắt PERL kết xuất thành core sau khi dịch script. core có thể lại được biến thành file chạy bằng cách dùng chương trình **undump** (không phải hệ UNIX nào cũng có lệnh này).

-U

Cho phép PERL thực hiện các thao tác “không an toàn”, như bỏ các liên kết khi đang chạy như `root`, hoặc chạy các chương trình như `setuid`.

-v

Hiện thông tin về phiên bản và mức cập nhật của bản PERL đang chạy.

-w

Hiển thị các cảnh báo về các định danh (identifier) được nêu lần đầu và về các biến vô hướng trước khi chúng được thiết lập. Cũng cảnh báo các chương trình con được xác định lại, tham chiếu đến các mô tả file (filehandle) không được xác định hay chỉ được mở “chỉ-đọc” mà bạn lại muốn ghi vào, hay cảnh báo nếu bạn dùng “==” cho các giá trị không phải là số, hoặc khi chương trình con lặp hơn 100 lần.

-xdirectory

Cho PERL biết script được nhúng vào trong thông báo.

VIII. 3. c. Các loại dữ liệu và đối tượng

PERL có ba loại dữ liệu: vô hướng, mảng vô hướng và các mảng liên hợp vô hướng. Mảng bình thường được đánh chỉ số bằng chữ số, còn mảng liên hợp vô hướng - bằng xâu ký tự. Đôi khi việc giải thích các thao tác và giá trị trong PERL lại phụ thuộc vào đòi hỏi của ngữ cảnh. Tồn tại ba ngữ cảnh chính: xâu ký tự, số và mảng. Một số thao tác trả về giá trị mảng đối với các ngữ cảnh đòi hỏi mảng, và các giá trị vô hướng trong trường hợp ngược lại. Các thao tác trả về các giá trị vô hướng không phụ thuộc vào ngữ cảnh đòi hỏi xâu ký tự hay số, nhưng các giá trị và biến vô hướng lại được diễn dịch như số hoặc xâu ký tự thích hợp với ngữ nghĩa. Vô hướng được hiểu như TRUE trong ngữ nghĩa nhị phân nếu như chúng không phải xâu ký tự không "" hoặc số 0. Các toán tử nhị phân trả về 1 nếu là TRUE, trả về xâu ký tự không "" hoặc 0 nếu là FALSE. Tuy nhiên xâu ký tự không lại được chia làm hai loại: không xác định và xác định. Xâu ký tự không không xác định được trả về nếu trên thực tế không có giá trị thực gì, ví dụ như cho trường hợp lỗi, hoặc ở cuối file, hoặc khi tham chiếu đến các biến hay phần tử của mảng chưa được khởi tạo.

Tham chiếu đến với các biến vô hướng, thậm chí nếu là thành phần trong mảng, luôn luôn được bắt đầu bằng ký tự “\$”. Sau đây là các ví dụ và giải nghĩa:

```
boolean $days
```

```

# biến vô hướng đơn giản
$days [28]

# phần tử thứ 29 của mảng @days
$days { 'Feb' }

# một giá trị từ mảng liên hợp
$#days

# chỉ số cuối cùng của mảng @days

```

nhưng toàn bộ mảng hay mặt cắt của mảng được ký hiệu bằng dấu "@":

```

@days          # ($days[0], $days[1], \ | ... $days[n])
@days[3,4,5]   # giống như @days[3.\|.5]
@days{'a','c'} # giống như $days{'a'}, $days{'c'})

```

và toàn bộ mảng liên hợp ký hiệu bởi dấu "%":

```
%days          # (key1, val1, key2, val2 ...)
```

Toàn bộ tám cấu trúc trên đều có thể dùng như một giá trị, có nghĩa có thể gán được đến chúng. Giống như trong **csh**, có thể tìm được chiều dài của mảng @days bằng cách viết "\$#days". Mảng nhiều chiều không được hỗ trợ trực tiếp, và thêm nữa, người ta thường viết một chương trình con để biến một script phức tạp thành một script đơn.

Mỗi một loại dữ liệu đều có khoảng đặt tên riêng, nên ta có thể sử dụng cùng một tên cho biến vô hướng, cho mảng, mảng liên hợp, cho filehandle, cho tên của chương trình con, cho nhân. Vì tham chiếu đến biến và mảng đều bắt đầu bằng "\$", "@", hay "%".

Các chữ số được viết theo dạng thông thường:

```

12345
12345.67
.23E-10
0xffff      # hex

```

```
0377      # octal
4_294_967_296
```

Xâu các ký tự được giới hạn bởi dấu nháy đơn hoặc kép. Chúng có chức năng giống như các dấu tương ứng của shell: xâu ký tự trong dấu nháy kép là đối tượng của dấu gạch xuống “\” và thay biến; còn xâu trong dấu nháy đơn lại không, trừ trường hợp cho “\’ ” và “\e”. Các quy tắc thông thường đối với dấu “\” cũng được áp dụng kèm với một vài dạng mới:

```
\t      tab
\n      newline
\r      return
\f      form feed
\b      backspace
\a      alarm (bell- chuông)
\e      escape
\033    octal char (ký tự bát phân)
\x1b    hex char (ký tự hexa)
\c[     control char (ký tự Ctrl)
\l      lowercase next char (ký tự tiếp theo in thường)
\u      uppercase next char (ký tự tiếp theo in hoa)
\L      lowercase till \E (ký tự in thường cho đến \E)
\U      uppercase till \E (ký tự in hoa cho đến \E)
\E      end case modification (kết thúc đổi dạng chữ)
```

Xâu ký tự trong dấu nháy đơn cần được ngăn cách với từ đứng trước nó bằng khoảng trống.

Hai ký tự đặc biệt là `__LINE__` và `__FILE__`, dùng để biểu diễn số thứ tự của dòng hiện thời và tên file tại vị trí đã xác định trong chương trình của bạn. Chúng chỉ được dùng như những dấu hiệu riêng mà không được thêm vào xâu ký tự. Dấu hiệu `__END__` có thể được dùng để chỉ kết thúc logic của script trước kết thúc thật sự của file. Mọi văn bản tiếp theo sẽ bị bỏ qua, tuy vẫn có thể được đọc thông qua filehandle `DATA`. (filehandle `DATA` có thể đọc dữ liệu chỉ từ script chính, mà

không phải từ bất cứ file hoặc xâu ký tự khác được yêu cầu). Hai ký tự điều khiển ^D và ^Z đồng nghĩa với __END__.

Các giá trị mảng được thêm vào các xâu ký tự trong dấu nháy kép bằng cách hợp tất cả các phần tử mảng với dấu ngăn cách được quy định trong biến \$", và mặc định là khoảng trống. Hai đoạn mã sau là tương đương:

```
$temp =join("$",@ARGV) ;  
system "echo $temp" ;
```

và:

```
system "echo @ARGV" ;
```

Dạng các dấu ngoặc hướng tới dòng được dựa trên cú pháp của shell. Sau dấu "<<" bạn chỉ ra xâu để kết thúc phần trong dấu nháy. Xâu kết thúc có thể hoặc là một identifier (từ), hoặc một văn bản trong dấu nháy. Nếu có văn bản trong dấu nháy, loại dấu nháy bạn dùng sẽ chỉ mang ý nghĩa thông thường. Từ (identifier) sẽ hoạt động như xâu ký tự trong dấu nháy kép. Không cho phép để khoảng trống giữa "<<" và identifier, nếu không, chương trình sẽ hiểu là bạn dùng identifier không.

Các ký tự mảng được đánh dấu như các giá trị riêng biệt, cách nhau bằng dấu phẩy và cả danh sách được đặt trong dấu ngoặc đơn:

(LIST)

Trong ngữ cảnh đòi hỏi các giá trị mảng, giá trị của ký tự mảng là giá trị của phần tử cuối như trong ngôn ngữ C. Ví dụ:

```
@foo = ('cc', '-E', $bar) ;
```

gán toàn bộ giá trị mảng tới mảng foo, nhưng

```
$foo = ('cc', '-E', $bar) ;
```

chỉ gán giá trị của biến bar tới biến foo. Chú ý rằng giá trị của một mảng thả trong ngữ cảnh vô hướng là kích thước của mảng; ví dụ sau gán giá trị 3 tới \$foo

```
@foo = ('cc', '-E', $bar);
$foo = @foo;      # $foo gets 3
```

Một ký tự của mảng liên hợp gồm một cặp được hiểu như khóa và giá trị:

```
# same as map assignment above
%map = ('red',0x00f,'blue',0x0f0,'green',0xf00);
```

Nếu <FILEHANDLE> được dùng trong ngữ cảnh tìm kiếm mảng, một mảng gồm tất cả các dòng đầu vào sẽ được trả về, mỗi dòng tương ứng với một phần tử mảng.

Filehandle không "<>" là một trường hợp đặc biệt, và có thể được dùng để mô phỏng **sed** hoặc **awk**. Đầu vào được lấy từ "<>" đến hoặc từ vào chuẩn, hoặc từ mỗi file được liệt kê ở dòng lệnh. Quá trình thực hiện diễn ra như sau: khi lần đầu tiên "<>" được gọi, mảng ARGV được kiểm tra, và nếu nó là mảng không, \$ARGV[0] được đặt là "-", tức là khi được mở sẽ cho dữ liệu từ vào chuẩn. Sau đó mảng ARGV được xử lý như là danh sách các file. Vòng lặp

```
while (<>) {
    ...                # mã nguồn cho mỗi dòng
}
```

tương đương với:

```
unshift(@ARGV, '-') if $#ARGV < $[;
while ($ARGV = shift) {
    open (ARGV, $ARGV);
    while (<ARGV>) {
        ...            # mã nguồn cho mỗi dòng
    }
}
```

VIII. 3. d. Các lệnh ghép

Trong PERL một dãy lệnh có thể được xử lý như một lệnh nếu chúng được đặt trong ngoặc tròn. Ngoặc tròn là bắt buộc. Tập hợp đó được gọi là BLOCK.

Các lệnh ghép sau đây được dùng để điều khiển luồng:

```
if (EXPR) BLOCK
if (EXPR) BLOCK else BLOCK
if (EXPR) BLOCK elsif (EXPR) BLOCK ... else BLOCK
LABEL while (EXPR) BLOCK
LABEL while (EXPR) BLOCK continue BLOCK
LABEL for (EXPR; EXPR; EXPR) BLOCK
LABEL foreach VAR (ARRAY) BLOCK
LABEL BLOCK continue BLOCK
```

Nếu dùng **unless** vào chỗ của **if**, ngữ nghĩa sẽ ngược lại.

Lệnh **while** thực hiện BLOCK với độ chính xác kép, nếu biểu thức là có giá trị TRUE. LABEL là tùy chọn, nếu được gọi, sẽ gồm identifier và dấu hai chấm tiếp theo. LABEL định danh vòng lặp cho các lệnh điều khiển vòng lặp **next**, **last** và **redo**. Nếu có **continue** BLOCK, nó sẽ được thực hiện ngay trước khi kiểm tra điều kiện về nó được gọi, giống như phần thứ ba trong vòng lặp **for** của ngôn ngữ C. Nó được dùng để tăng giá trị cho biến lặp, thậm chí khi vòng lặp đang được tiếp tục thông qua lệnh **next** (giống như lệnh **continue** của C). Nếu như **while** được thay bằng **until**, ngữ nghĩa sẽ ngược lại, nhưng thao tác kiểm tra vẫn được thực hiện trước lần lặp đầu tiên.

Trong cả câu lệnh với **if** hoặc **while**, có thể thay (EXPR) bằng BLOCK, và kiểm tra điều kiện sẽ là TRUE nếu giá trị của lệnh cuối cùng trong BLOCK là TRUE.

Vòng lặp **for** làm việc giống như vòng lặp **while** tương ứng:

```
for ($i = 1; $i < 10; $i++) {
    ...
}
```

tương tự như

```
$i = 1;
while ($i < 10) {
    ...
} continue {
    $i++;
}
```

Vòng lặp **foreach** thực hiện các bước lặp trên giá trị của mảng thông thường và đặt biến VAR lần lượt là mỗi phần tử mảng. Biến sẽ là cục bộ trong vòng lặp và lấy lại giá trị cũ của nó khi kết thúc lặp. Thực tế "**foreach**" tương tự như "**for**". Nếu VAR bị bỏ qua, \$_ được đặt cho mỗi giá trị. Nếu ARRAY là mảng thực sự (chứ không phải một biểu thức trả về giá trị mảng), ta có thể thay đổi mỗi phần tử mảng bằng cách thay đổi VAR bên trong vòng lặp.

Tự thân BLOCK, dù có nhân hay không, cũng tương đương với vòng lặp thực hiện một lần, nên có thể dùng các lệnh điều khiển vòng lặp để thoát khỏi hay khởi động lại BLOCK. Có thể tạo ra đoạn mã nguồn phân tình huống như sau:

```
foo: {
    if (/^abc/) { $abc = 1; last foo; }
    if (/^def/) { $def = 1; last foo; }
    if (/^xyz/) { $xyz = 1; last foo; }
    $nothing = 1;
}
```

PERL chính thức không có lệnh chuyển trường hợp (như **case** trong shell), vì ngoài cách trên còn có nhiều cách khác nữa để thực hiện mục đích trên:

```
foo: {
    $abc = 1, last foo if /^abc/;
    $def = 1, last foo if /^def/;
    $xyz = 1, last foo if /^xyz/;
```

```
        $nothing = 1;
    }
```

hoặc:

```
foo: {
    /^abc/ && do { $abc = 1; last foo; };
    /^def/ && do { $def = 1; last foo; };
    /^xyz/ && do { $xyz = 1; last foo; };
    $nothing = 1;
}
```

hoặc:

```
foo: {
    /^abc/ && ($abc = 1, last foo);
    /^def/ && ($def = 1, last foo);
    /^xyz/ && ($xyz = 1, last foo);
    $nothing = 1;
}
```

hoặc thậm chí:

```
if (/^abc/)
    { $abc = 1; }
elsif (/^def/)
    { $def = 1; }
elsif (/^xyz/)
    { $xyz = 1; }
else
    {$nothing = 1;}
```


VIII. 3. e. Các lệnh đơn giản

Mỗi lệnh đơn giản cần được kết thúc bằng dấu chấm phẩy “;”, cho đến lệnh cuối cùng trong BLOCK. Lệnh cuối cùng đó có tùy chọn kết thúc bằng dấu hai chấm “:”.

Mỗi lệnh đơn giản theo tùy chọn có thể được tiếp nối bằng từ bỏ nghĩa đơn, ngay trước dấu chấm phẩy “;” kết thúc. Các từ bỏ nghĩa có thể là:

```
if  EXPR
unless EXPR
while EXPR
until EXPR
```

VIII. 3. f. Biểu thức

▪ Các thao tác điều khiển dòng

do BLOCK

Trả về giá trị của lệnh cuối trong dãy lệnh được chỉ bởi BLOCK.

do SUBROUTINE (LIST)

Thực hiện SUBROUTINE được xác định qua khai báo sub, và trả về giá trị của biểu thức cuối trong SUBROUTINE. Sẽ báo lỗi nặng nếu không có chương trình con với tên đã cho.

SUBROUTINE có thể là biến vô hướng duy nhất, lúc đó tên của chương trình con được lấy là tên của biến.

do EXPR

Dùng giá trị của EXPR như tên file và thực hiện nội dung file như PERL script. Công dụng trước hết là để gọi chương trình con từ thư viện chương trình con của PERL. Ví dụ:

```
do 'stat.pl';
```

tương tự

```
eval ``cat stat.pl``;
```

Chú ý rằng hai thao tác sau hoàn toàn khác nhau:

```
do $foo;      # eval a file
do $foo();    # call a subroutine
```

Việc gọi các chương trình con của thư viện tốt nhất nên thực hiện với toán tử **require**.

goto LABEL

Tìm lệnh được gắn nhãn và thực hiện. Nói chung ít được dùng và đang được tác giả chuyển ngữ nghĩa.

last

last LABEL

last

Lệnh **last** tương tự như lệnh **break** trong C (khi dùng trong vòng lặp); để dừng lập tức vòng lặp.

next LABEL

next

Lệnh **next** tương tự như lệnh **continue** trong C (khi dùng trong vòng lặp); để bắt đầu bước lặp tiếp theo của vòng lặp:

```
line: while(<STDIN>) {
    next line if /^#/;    # discard comments
    ...
}
```

redo

redo LABEL

redo

Lệnh **redo** khởi động lại vòng lặp của BLOCK mà không gọi lại kiểm tra điều kiện.

▪ *Các toán tử*

Các biểu thức PERL hoạt động hầu như giống các biểu thức của C, chỉ trừ các khác biệt sau đây:

Toán tử hàm mũ.

****=**

Toán tử gán hàm mũ.

()

Danh sách không, dùng để khởi tạo mảng không.

.

Nối hai xâu ký tự.

. =

Toán tử gán nối hai xâu ký tự.

eq

Phép bằng cho xâu ký tự (cho số, ký hiệu được dùng là ==).

ne

Phép không bằng cho xâu ký tự (cho số, ký hiệu được dùng là !=).

lt

Xâu ký tự nhỏ hơn.

gt

Xâu ký tự lớn hơn.

le

Xâu ký tự nhỏ hơn hoặc bằng.

ge

Xâu ký tự lớn hơn hoặc bằng.

cmp

Số sánh xâu ký tự, trả về -1, 0, hay 1.

<=>

Số sánh xâu số, trả về -1, 0, hay 1.

=~

Mặc định là tìm hoặc sửa đổi xâu ký tự "\$_". Đối số về phải là dãy ký tự tìm kiếm, thay thế hoặc dịch. Đối số về trái là dãy ký tự được đề nghị tìm kiếm, thay thế hoặc dịch, được đưa vào thay thế giá trị mặc định "\$_".

! ~

Giống = ~ , chỉ trừ một điểm là giá trị trả về bị mang dấu âm.

x

Toán tử lặp. Trả về xâu chứa toán hạng bên trái, được lặp một số lần theo sự xác định do toán hạng bên phải. Ví dụ:

```
print '-' x 80; # in dòng các dấu gạch ngang
print '-' x80; # không hợp lệ, x80 là identifier
print "\t" x ($tab/8), ' ' x ($tab%8);# dấu Tab
@ones = (1) x 80; # mảng gồm 80 số 1
@ones = (5) x @ones; # đặt tất cả các phần tử bằng 5
```

x=

Toán tử gán lặp. Chỉ làm việc với dữ liệu vô hướng.

..

Toán tử khoảng, thực tế là hai toán tử hoàn toàn khác nhau tùy thuộc ngữ cảnh. Trong ngữ cảnh mảng, trả về mảng các số đếm giá trị từ trái sang phải. Trong ngữ cảnh vô hướng, toán tử .. trả về giá trị nhị phân. Giá trị là FALSE nếu toán hạng bên trái là FALSE. Nếu toán hạng bên trái là TRUE, toán tử .. sẽ giữ là TRUE cho đến khi toán hạng bên phải là TRUE, sau đó .. lại trở thành FALSE.

Các toán tử kiểm tra file.

r	File đọc được bởi các quyền SUID/SGID.
w	File ghi được bởi các quyền SUID/SGID.
x	File chạy được bởi các quyền SUID/SGID.
o	File được sở hữu bởi quyền SUID.
R	File đọc được bởi các quyền UID/GID thực sự.
W	File ghi được bởi các quyền UID/GID thực sự.
X	File chạy được bởi các quyền UID/GID thực sự.

o	File được sở hữu bởi quyền UID thực sự.
e	File tồn tại.
z	File có kích thước 0.
s	File có kích thước khác 0 (trả về kích thước file).
f	File là file thường.
d	File là thư mục.
l	File là liên kết hình thức.
p	File là đường ống có tên (FIFO).
S	File là socket.
b	File là file đặc biệt (thiết bị khối).
c	File là file đặc biệt (thiết bị ký tự).
u	File có đặt bit setuid.
g	File có đặt bit setgid.
k	File có đặt sticky bit.
t	Filehandle được mở tới tty.
T	File là file văn bản.
B	File file nhị phân (ngược với T).
M	Tuổi của file (ngày), kể từ khởi động script.
A	Tương tự cho thời gian truy nhập file.
C	Tương tự cho thời gian thay đổi inode của file.

▪ *Các hàm số học*

atan2 (Y, X)

Cho giá trị arctang của Y/X trong khoảng -PI đến PI.

cos (EXPR)

cos EXPR

Cho giá trị cos của EXPR (radians). Nếu EXPR bị bỏ qua, lấy giá trị cos của \$_.

exp (EXPR)

exp EXPR

Cho giá trị hàm mũ cơ số e của EXPR. Nếu EXPR bị bỏ qua, cho giá trị của

exp (\$_).

int (EXPR)

int EXPR

Lấy phần nguyên của **EXPR**. Nếu **EXPR** bị bỏ qua, dùng **\$_**.

log(EXPR)

log EXPR

Cho giá trị hàm lô ga rit cơ số e của **EXPR**. Nếu **EXPR** bị bỏ qua, cho log của **\$_**.

rand(EXPR)

rand EXPR

rand

Cho giá trị ngẫu nhiên trong khoảng từ 0 đến giá trị của **EXPR**. (**EXPR** phải dương.) Nếu **EXPR** bị bỏ qua, cho giá trị trong khoảng giữa 0 và 1. Xem cả

srand().

sin(EXPR)

sin EXPR

Cho giá trị sin của **EXPR** (radian). Nếu **EXPR** bị bỏ qua, lấy giá trị sin của **\$_**.

sqrt(EXPR)

sqrt EXPR

Cho giá trị căn bậc hai của **EXPR**. Nếu **EXPR** bị bỏ qua, lấy giá trị căn của **\$_**.

srand(EXPR)

srand EXPR

Đặt số ngẫu nhiên cho toán tử **rand** operator. Nếu **EXPR** bị bỏ qua, lấy **srand(time)**.

- *Các hàm tiến đời*

gmtime(EXPR)

gmtime EXPR

Chuyển giá trị thời gian trả về bởi lệnh **time** thành dạng mảng 9 phần tử cho thời gian múi giờ Greenwich (thời gian chuẩn quốc tế). Thường dùng như sau:

```
($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst)  
=gmtime(time);
```

hex(EXPR)

hex EXPR

Trả về giá trị thập phân của **EXPR** được đọc như xâu hexa. Nếu bỏ qua **EXPR**, dùng **\$_**.

localtime(**EXPR**)

localtime **EXPR**

Chuyển giá trị thời gian trả về bởi lệnh **time** thành dạng mảng 9 phần tử cho thời gian địa phương. Thường dùng như sau:

```
($sec, $min, $hour, $mday, $mon, $year, $wday, $yday, $isdst)
= localtime(time);
```

oct(**EXPR**)

oct **EXPR**

Trả về giá trị thập phân của **EXPR** được đọc như xâu bát phân. Nếu bỏ qua **EXPR**, dùng **\$_**.

ord(**EXPR**)

ord **EXPR**

Trả về giá trị số ASCII cho ký tự đầu tiên của **EXPR**. Nếu bỏ qua **EXP**, dùng **\$_**.

pack(**TEMPLATE**, **LIST**)

Lấy mảng hoặc danh sách các giá trị và đóng gói thành cấu trúc nhị phân, trả về xâu chứa cấu trúc. **TEMPLATE** là dãy các ký tự xác định thứ tự và loại của các giá trị như sau:

- A** Xâu ký tự ASCII, có đệm bằng các khoảng trống.
- a** Xâu ký tự ASCII, có đệm bằng ký tự null.
- c** Giá trị signed char (ký tự có dấu).
- C** Giá trị unsigned char (ký tự không dấu).
- s** Giá trị signed short (số độ chính xác đơn có dấu).
- S** Giá trị unsigned short (số độ chính xác đơn không dấu).
- i** Giá trị signed integer (số nguyên có dấu).
- I** Giá trị unsigned integer (số nguyên không dấu).
- l** Giá trị signed long (số độ chính xác kép có dấu).
- L** Giá trị unsigned long (số độ chính xác kép không dấu).
- n** Giá trị short trong thứ tự "mạng".
- N** Giá trị long trong thứ tự "mạng".

f	Dấu phẩy động độ chính xác đơn ở dạng nguyên thủy.
d	Dấu phẩy động độ chính xác kép ở dạng nguyên thủy
p	Con trỏ tới xâu ký tự.
v	Giá trị short trong thứ tự "VAX".
V	Giá trị long trong thứ tự "VAX".
x	Giá trị byte null.
X	Lưu trữ byte.
@	Điền ký tự không (null) tới vị trí tuyệt đối.
u	Xâu được mã hóa theo kiểu uuencode.
b	Xâu bit (thứ tự tăng bit, giống như vec ()).
B	Xâu bit (thứ tự giảm bit).
h	Xâu hexa (bậc thấp xếp trước).
H	Xâu hexa (bậc cao xếp trước).

unpack (TEMPLATE, EXPR)

Ngược lại với **pack**.

vec (EXPR, OFFSET, BITS)

Xử lý xâu như vectơ các số unsigned integers.

- *Các hàm xâu ký tự*

chop

chop (LIST)

chop (VARIABLE)

chop VARIABLE

chop

Cắt ký tự cuối cùng của xâu và trả về ký tự đó. Thường dùng để bỏ dấu xuống dòng khỏi đầu của bản ghi đầu vào. Nếu bỏ qua VARIABLE, sẽ cắt bỏ \$_.

crypt (PLAINTEXT, SALT)

Mã hóa xâu ký tự giống hệt như lệnh **crypt ()** trong thư viện C. Tiễn dùng để kiểm tra file mật khẩu. Quyền dùng hạn chế.

eval (EXPR)

eval EXPR

eval BLOCK

EXPR được đọc và chạy như một chương trình PERL nhỏ.

index(STR, SUBSTR, POSITION)

index(STR, SUBSTR)

Cho biết vị trí của lần gặp SUBSTR đầu tiên trong STR hoặc ở sau POSITION.

Nếu bỏ qua POSITION, bắt đầu tìm từ đầu xâu.

length(EXPR)

length EXPR

Trả về kích thước (theo số ký tự) của giá trị EXPR. Nếu bỏ qua EXPR, trả về kích thước của `$_`.

q/STRING/

qq/STRING/

qx/STRING/

Đây không phải là một hàm thực, mà là cách tránh khỏi phải viết nhiều dấu ngoặc. Toán tử `q` là ngoặc đơn, `qq` - ngoặc kép; `qx` - ngoặc ngược.

rindex(STR, SUBSTR, POSITION)

rindex(STR, SUBSTR)

Hoạt động giống **index** nhưng còn trả về vị trí lần gặp cuối cùng của SUBSTR trong STR.

substr(EXPR, OFFSET, LEN)

substr(EXPR, OFFSET)

Dẫn xâu con khỏi EXPR và trả nó về.

- *Các hàm mảng và danh sách*

delete \$ASSOC{KEY}

Xóa giá trị được xác định khỏi mảng liên hợp được xác định. Trả về giá trị bị xóa, hoặc giá trị không xác định nếu không xóa gì. Việc xóa từ `$ENV{ }` sẽ làm thay đổi môi trường.

each(ASSOC_ARRAY)

each ASSOC_ARRAY

Trả về hai phần tử mảng chứa khóa và giá trị đối với giá trị kế tiếp của mảng liên hợp.

grep(EXPR, LIST)

Tính `EXPR` cho mỗi phần tử của `LIST` và trả về giá trị mảng chứa các phần tử mà đối với chúng biểu thức là `TRUE`. Trong ngữ cảnh vô hướng sẽ trả về số lần biểu thức là `TRUE`.

join(`EXPR`,`LIST`)

join(`EXPR`,`ARRAY`)

Hợp hai xâu riêng rẽ của `LIST` hay `ARRAY` thành một xâu với các trường được ngăn cách bằng các giá trị của `EXPR`, rồi trả về xâu đó.

keys(`ASSOC_ARRAY`)

keys `ASSOC_ARRAY`

Trả về mảng chứa tất cả các khóa của mảng liên hợp được gọi tên.

pop(`ARRAY`)

pop `ARRAY`

Hiện và trả về giá trị cuối cùng của mảng. Nếu mảng không có phần tử, trả về giá trị không xác định.

push(`ARRAY`,`LIST`)

Xử lý `ARRAY` (@ là tùy chọn) như ngăn xếp, và đẩy các giá trị của `LIST` về cuối `ARRAY`. Kích thước `ARRAY` tăng một lượng bằng kích thước `LIST`.

reverse(`LIST`)

reverse `LIST`

Trong ngữ cảnh mảng, trả về giá trị mảng chứa các phần tử của `LIST` theo thứ tự ngược. Trong ngữ cảnh vô hướng, trả về giá trị xâu chứa các byte của phần tử `LIST` đầu tiên theo thứ tự ngược.

shift(`ARRAY`)

shift `ARRAY`

shift

Dịch chuyển giá trị đầu tiên của mảng và trả nó về, làm ngắn mảng đi một giá trị bằng 1. Nếu bỏ qua `ARRAY`, sẽ dịch mảng `@ARGV` trong chương trình chính, và mảng `@_` trong các chương trình con.

sort(`SUBROUTINE LIST`)

sort(`LIST`)

sort `SUBROUTINE LIST`

sort `BLOCK LIST`

sort `LIST`

Phân loại **LIST** và trả về giá trị mảng đã sắp xếp.

splice(**ARRAY**, **OFFSET**, **LENGTH**, **LIST**)

splice(**ARRAY**, **OFFSET**, **LENGTH**)

splice(**ARRAY**, **OFFSET**)

Loại các phần tử được chọn bởi **OFFSET** và **LENGTH** khỏi mảng, thay thế chúng bằng các phần tử của **LIST**, nếu có. Trả về các phần tử đã bị loại khỏi mảng. Nếu bỏ qua **LENGTH**, sẽ loại mọi thứ kể từ **OFFSET** trở đi.

split(**/PATTERN/**, **EXPR**, **LIMIT**)

split(**/PATTERN/**, **EXPR**)

split(**/PATTERN/**)

split

Chia xâu thành mảng các xâu và trả về.

unshift(**ARRAY**, **LIST**)

Có tác dụng ngược lại với **shift**.

values(**ASSOC_ARRAY**)

values **ASSOC_** **OFFSET**

Trả về mảng chứa tất cả các giá trị của mảng liên hợp được gọi.

- *Các thao tác với file*

chmod(**LIST**)

chmod **LIST**

Đổi chủ quyền của danh sách các file. Phần tử đầu tiên của danh sách phải ở trong chế độ số. Trả về số file đã được đổi thành công. Ví dụ:

```
$cnt = chmod 0755, 'foo', 'bar';
chmod 0755, @executables;
```

chown(**LIST**)

chown **LIST**

Đổi người sở hữu (và nhóm) của danh sách các file. Hai phần tử đầu tiên của danh sách phải là UID và GID ở trong chế độ số, theo thứ tự như trên. Trả về số file đã được đổi thành công. Ví dụ:

```
$cnt = chown $uid, $gid, 'foo', 'bar';
```

```
chown $uid, $gid, @filenames;
```

link(OLDFILE,NEWFILE)

Tạo tên file mới liên kết với tên cũ của file. Trả về 1 nếu thành công, 0 nếu thất bại.

lstat(FILEHANDLE)

lstat FILEHANDLE

lstat(EXPR)

lstat SCALARVARIABLE

Thực hiện chức năng như hàm **stat()**, nhưng xem trạng thái của liên kết hình thức thay vì file mà liên kết hình thức trỏ đến.

mkdir(FILENAME,MODE)

Tạo thư mục có tên **FILENAME**, với chủ quyền xác định bởi **MODE**. Trả về 1 nếu thành công, 0 nếu thất bại và đặt **\$! (errno)**.

readlink(EXPR)

readlink EXPR

Trả về giá trị của liên kết hình thức. Nếu có lỗi hệ thống trả về giá trị không xác định và đặt **\$! (errno)**. Nếu bỏ qua **EXPR**, dùng **\$_**.

rename(OLDNAME,NEWNAME)

Đổi tên file. Trả về 1 nếu thành công, 0 nếu thất bại.

rmdir(FILENAME)

rmdir FILENAME

Xóa thư mục có tên **FILENAME** nếu nó rỗng. Trả về 1 nếu thành công, 0 nếu thất bại. Nếu bỏ qua **FILENAME**, dùng **\$_**.

select(RBITS,WBITS,EBITS,TIMEOUT)

Chọn lệnh gọi hệ thống với mặt nạ bit đã chọn.

stat(FILEHANDLE)

stat FILEHANDLE

stat(EXPR)

stat SCALARVARIABLE

Trả về mảng 13 phần tử cho biết thông kê về file.

symlink(OLDNAME,NEWNAME)

Tạo tên file mới có liên kết hình thức với tên cũ.

truncate(FILEHANDLE, LENGTH)

truncate(EXPR, LENGTH)

Cắt file được mở trên FILEHANDLE, hoặc được chỉ tên bởi EXPR, đến độ dài xác định.

unlink(LIST)

unlink LIST

Xóa đi danh sách các file. Trả về số file đã bị xóa.

utime(LIST)

utime LIST

Thay đổi thời gian truy nhập và sửa đổi cho mỗi file trong danh sách file. Hai phần tử đầu tiên phải là các thời gian truy nhập và sửa đổi file bằng số, theo thứ tự trên. Trả về số file đã được thay đổi. Thời gian thay đổi inode của mỗi file được đặt bằng thời gian hiện thời.

▪ *Các chương trình con đọc thư mục*

closedir(DIRHANDLE)

closedir DIRHANDLE

Đóng thư mục được mở bằng **opendir()**.

opendir(DIRHANDLE, EXPR)

Mở thư mục có tên EXPR để xử lý bằng các lệnh **readdir()**, **telldir()**, **seekdir()**, **rewinddir()** và **closedir()**.

readdir(DIRHANDLE)

readdir DIRHANDLE

Trả về dòng vào tiếp theo của thư mục.

rewinddir(DIRHANDLE)

rewinddir DIRHANDLE

Đặt vị trí hiện thời đến đầu thư mục cho chương trình con **readdir()** chạy trên DIRHANDLE.

seekdir(DIRHANDLE, POS)

Đặt vị trí hiện thời đến đầu thư mục cho chương trình con **readdir()** chạy trên DIRHANDLE. POS cần phải là giá trị trả về bởi **telldir()**.

telldir(DIRHANDLE)

telldir DIRHANDLE

Đặt vị trí hiện thời đến đầu thư mục cho chương trình con **readdir()** chạy trên **DIRHANDLE**. Giá trị có thể được cho **seekdir()** để truy nhập vị trí cụ thể trong thư mục

▪ *Các thao tác vào / ra*

binmode(FILEHANDLE)

binmode FILEHANDLE

Sắp xếp file để được đọc trong chế độ “nhị phân” trong các hệ điều hành có phân biệt các file văn bản và file nhị phân.

close(FILEHANDLE)

close FILEHANDLE

Đóng file hay đường ống.

dbmclose(ASSOC_ARRAY)

dbmclose ASSOC_ARRAY

Hủy liên kết giữa file loại dbm và mảng liên hợp.

dbmopen(ASSOC,DBNAME,MODE)

Tạo liên kết giữa file loại dbm hay ndbm với mảng liên hợp có tên ASSOC. DBNAME là tên cơ sở dữ liệu (không có phần mở rộng .dir hay .pag). Nếu không tồn tại cơ sở dữ liệu thì tạo bảo vệ theo xác định bởi MODE.

eof(FILEHANDLE)

eof()

eof

Trả về 1 nếu lần đọc tiếp theo trên FILEHANDLE sẽ trả giá trị kết thúc file (EOF), hay nếu FILEHANDLE không mở. FILEHANDLE có thể là biểu thức mà giá trị của nó cho tên của filehandle thực sự.

fcntl(FILEHANDLE,FUNCTION,SCALAR)

Cải tiến của hàm **fcntl(2)**.

fileno(FILEHANDLE)

fileno FILEHANDLE

Trả về mô tả file (file descriptor) cho filehandle.

flock(FILEHANDLE,OPERATION)

Gọi **flock(2)** trên FILEHANDLE.

getc(FILEHANDLE)

getc FILEHANDLE

getc

Trả về ký tự tiếp theo từ file đầu vào được gắn với FILEHANDLE, hoặc xâu không ở EOF. Nếu bỏ qua FILEHANDLE, sẽ đọc từ STDIN.

ioctl (FILEHANDLE, FUNCTION, SCALAR)

Cải tiến của hàm **ioctl(2)**.

open (FILEHANDLE, EXPR)

open (FILEHANDLE)

open FILEHANDLE

Mở file có tên được cho bởi EXPR, và liên kết nó với FILEHANDLE. Nếu tên file được bắt đầu với "<" hoặc không có gì, file mở cho đầu vào, ">" - cho đầu ra, ">>" - file mở để thêm vào.

pipe (READHANDLE, WRITEHANDLE)

Mở một cặp đường ống nối với nhau theo lệnh gọi hệ thống tương ứng.

print (FILEHANDLE LIST)

print (LIST)

print FILEHANDLE LIST

print LIST

print

In xâu hoặc danh sách các xâu cách nhau bằng dấu phẩy ",", ".".

printf (FILEHANDLE LIST)

printf (LIST)

printf FILEHANDLE LIST

printf LIST

Tương đương với **print** FILEHANDLE, **sprintf** (LIST).

read (FILEHANDLE, SCALAR, LENGTH, OFFSET)

read (FILEHANDLE, SCALAR, LENGTH)

Đọc byte LENGTH của dữ liệu vào biến SCALAR từ FILEHANDLE đã cho.

seek (FILEHANDLE, POSITION, WHENCE)

Định vị ngẫu nhiên con trỏ file cho FILEHANDLE, tương tự như lệnh gọi **fseek()** của **stdio**.

select (FILEHANDLE)

select

Trả về filehandle hiện thời được chọn.

sprintf(FORMAT, LIST)

Trả về xâu được định dạng theo **printf**. Ký tự viết tắt “*” không được hỗ trợ.

sysread(FILEHANDLE, SCALAR, LENGTH, OFFSET)

sysread(FILEHANDLE, SCALAR, LENGTH)

Đọc byte LENGTH của dữ liệu vào biến SCALAR từ FILEHANDLE đã cho, dùng lệnh gọi hệ thống **read(2)**. Trả về số byte đọc được. Dùng OFFSET nếu vị trí đọc khác với đầu của xâu.

syswrite(FILEHANDLE, SCALAR, LENGTH, OFFSET)

syswrite(FILEHANDLE, SCALAR, LENGTH)

Ghi byte LENGTH của dữ liệu vào biến SCALAR từ FILEHANDLE đã cho, dùng lệnh gọi hệ thống **write(2)**. Trả về số byte ghi được. Dùng OFFSET nếu vị trí ghi khác với đầu của xâu.

tell(FILEHANDLE)

tell FILEHANDLE

tell

Trả về vị trí hiện thời của file cho FILEHANDLE.

write(FILEHANDLE)

write(EXPR)

write

Ghi bản ghi đã định dạng (có thể nhiều dòng) vào file đã cho.

- *Các thao tác tìm kiếm và sửa đổi*

m/PATTERN/gio

/PATTERN/gio

Tìm kiếm xâu có mẫu trùng hợp, trả về TRUE hay FALSE. Nếu không có xâu nào được xác định bằng các toán tử “=~” hay “!~”, xâu \$_ sẽ được tìm. Nếu dấu hiệu của thao tác tìm là “/” thì (m) là tùy chọn để có thể dùng đôi ký tự không phải chữ và số bất kỳ làm dấu hiệu của toán tử tìm kiếm.

?PATTERN?

Tương tự như toán tử **/pattern/** trên, nhưng chỉ thực hiện một lần giữa các lệnh gọi hệ thống đến toán tử **reset**.

s/PATTERN/REPLACEMENT/gio

Tìm kiếm xâu ký tự theo mẫu, và nếu tìm được, thay bằng văn bản thay thế và trả về giá trị số thay thế đã thực hiện. Tùy chọn (g) chỉ rằng tất cả mọi mẫu bắt gặp đều bị thay thế; tùy chọn (i) - có phân biệt chữ hoa và chữ thường; (e) - xâu thay thế được coi là biểu thức cần xem xét chữ không phải xâu nằm trong dấu ngoặc kép; (o) - mẫu chỉ được biên dịch một lần. Nếu không có xâu nào được xác định bằng các toán tử =~ hay !~, xâu \$_ sẽ được tìm và thay đổi.

study (SCALAR)

study SCALAR

study

Lấy thêm thời gian để xem xét SCALAR khi có nhiều mẫu trùng trên xâu.

tr/SEARCHLIST/REPLACEMENTLIST/cds

y/SEARCHLIST/REPLACEMENTLIST/cds

Dịch mọi lần xuất hiện của các ký tự trong danh sách tìm kiếm thành ký tự tương ứng trong danh sách thay thế. Nếu không có xâu nào được xác định bằng các toán tử =~ hay !~, xâu \$_ sẽ được dịch. Nếu dùng (c) bộ ký tự trong SEARCHLIST sẽ được bù; nếu dùng (d) - mọi ký tự được chỉ trong SEARCHLIST nhưng không tìm thấy trong REPLACEMENTLIST sẽ bị xóa; (s) - dãy ký tự được dịch thành cùng một ký tự sẽ bị nén thành một lần xuất hiện của ký tự đó.

▪ *Các chương trình con giúp hệ thống*

alarm (SECONDS)

alarm SECONDS

Sắp xếp để có SIGALRM được phân phối đến tiến trình đó sau một số giây nhất định.

chdir (EXPR)

chdir EXPR

Thay đổi thư mục làm việc hiện thời đến EXPR, nếu có thể. Nếu bỏ qua EXPR, đích sẽ là thư mục home.

chroot (FILENAME)

chroot FILENAME

Thực hiện tương tự như lệnh gọi hệ thống cùng tên.

die (LIST)

die LIST

Thoát ra ngoài, in giá trị của `LIST` tới `STDERR` và kết thúc với giá trị hiện thời của `$!` (`errno`).

exec(`LIST`)

exec `LIST`

Nếu có nhiều hơn một đối số trong `LIST`, hoặc nếu `LIST` là mảng với nhiều hơn một giá trị, gọi lệnh **execvp**() với đối số trong `LIST`.

exit(`EXPR`)

exit `EXPR`

Tính `EXPR` và thoát ngay với giá trị đó.

fork

Thực hiện lệnh gọi hệ thống **fork**(). Trả `PID` của tiến trình con về cho tiến trình cha và 0 cho tiến trình con.

getlogin

Trả về quá trình đăng nhập hiện thời từ `/etc/utmp`, nếu có thể.

getpgrp(`PID`)

getpgrp `PID`

Trả về nhóm tiến trình hiện thời đối với `PID` xác định, 0 cho tiến trình hiện thời.

Nếu bỏ qua `EXPR`, trả nhóm tiến trình về tiến trình hiện thời.

getppid

Trả về `PID` của tiến trình cha.

getpriority(`WHICH, WHO`)

Trả về mức độ ưu tiên hiện thời cho tiến trình, nhóm tiến trình, hoặc người dùng.

kill(`LIST`)

kill `LIST`

Gửi tín hiệu đến danh sách các tiến trình. Phần tử đầu tiên phải là tín hiệu cần gửi. Trả về số tiến trình nhận được tín hiệu.

Nếu tín hiệu âm sẽ thực hiện hủy nhóm tiến trình thay vì các tiến trình.

setpgrp(`PID, PGRP`)

Đặt nhóm tiến trình hiện thời `PID` xác định, 0 cho tiến trình hiện thời.

setpriority(`OPTNAME, WHO, PRIORITY`)

Đặt mức độ ưu tiên hiện thời cho tiến trình, nhóm tiến trình, hoặc người dùng.

sleep(`EXPR`)

sleep `EXPR`

sleep

Cho script dừng **EXPR** giây, hoặc dừng hoàn toàn nếu không chỉ ra **EXPR**.

syscall (OPTNAME)**syscall LIST**

Gọi lệnh gọi hệ thống được xác định như là phần tử đầu tiên của danh sách.

system (LIST)**system LIST**

Tương tự như **exec LIST**, nhưng lệnh **fork** được thực hiện đầu tiên, và tiến trình cha chờ cho đến khi tiến trình con kết thúc.

time

Trả về số giây kể từ thời điểm 00:00:00 UTC, January 1, 1970.

umask (EXPR)**umask EXPR****umask**

Đặt mặt nạ cho tiến trình và trả về giá trị cũ. Nếu bỏ qua **EXPR**, trả về mặt nạ hiện thời.

wait

Chờ tiến trình con kết thúc và trả về PID của tiến trình đã chết, hay -1 nếu không có tiến trình con.

waitpid (PID, FLAGS)

Chờ tiến trình con cụ thể kết thúc và trả về PID của tiến trình đã chết, hay -1 nếu không có tiến trình con.

warn (LIST)**warn LIST**

Tạo thông báo trên **STDERR** giống như "**die**", nhưng không kết thúc tiến trình.

- *Các thao tác mạng và giao tiếp giữa các tiến trình*

accept (NEWSOCKET, GENERICSOCKET)

Thực hiện tương tự như chấp nhận (**accept**) lệnh gọi hệ thống.

bind (SOCKET, NAME)

Thực hiện tương tự như nối kết (**bind**) lệnh gọi hệ thống.

connect (SOCKET, NAME)

Thực hiện tương tự như nối (**connect**) lệnh gọi hệ thống.

getpeername (SOCKET)

Trả về địa chỉ sockaddr được đóng gói hay đầu khác của kết nối SOCKET.

getsockname (SOCKET)

Trả về địa chỉ sockaddr được đóng gói của đầu này kết nối SOCKET.

getsockopt (SOCKET, LEVEL, OPTNAME)

Trả về tùy chọn được yêu cầu của socket, hoặc không xác định nếu có lỗi.

listen (SOCKET, QUEUESIZE)

Thực hiện tương tự như lệnh gọi hệ thống **listen** đã làm.

msgctl (ID, CMD, ARG)

Gọi hàm IPC **msgctl** của System V. Nếu CMD là &IPC_STAT, sau đó ARG phải là biến giữ cấu trúc msqid_ds trả về.

msgget (KEY, FLAGS)

Gọi hàm IPC **msgget** của System V. Trả về định danh của hàng chờ các thông báo, hoặc không xác định nếu có lỗi.

msgsnd (ID, MSG, FLAGS)

Gọi hàm IPC **msgsnd** của System V để gửi thông báo MSG tới ID hàng chờ thông báo. MSG phải được bắt đầu với loại thông báo long integer.

msgrcv (ID, VAR, SIZE, TYPE, FLAGS)

Gọi hàm IPC **msgrcv** của System V để nhận thông báo từ ID hàng chờ thông báo tới biến VAR với kích thước thông báo cực đại SIZE. Nếu một thông báo được nhận, loại thông báo phải chiếm chỗ đầu tiên trong VAR, và kích thước cực đại của VAR là SIZE cộng với kích thước của loại thông báo.

recv (SOCKET, SCALAR, LEN, FLAGS)

Nhận thông báo trên socket. Nhận các byte LENGTH của dữ liệu tới biến SCALAR từ filehandle SOCKET xác định.

semctl (ID, SEMNUM, CMD, ARG)

Gọi hàm IPC **semctl** của System V, nếu CMD là &IPC_STAT hay &GETALL, sau đó ARG phải là biến giữ giá trị cấu trúc semid_ds hay mảng giá trị semaphore trả về.

semget (KEY, NSEMS, SIZE, FLAGS)

Gọi hàm IPC **semget** của System V. Trả về ID của semaphore, hoặc không xác định nếu có lỗi.

semop (KEY, OPSTRING)

Gọi hàm IPC **semop** của System V để thực hiện các thao tác semaphore như gửi tín hiệu hay chờ đợi. **OPSTRING** phải là mảng đóng gói các cấu trúc **semop**. Số thao tác semaphore được coi là bằng kích thước của **OPSTRING**.

send(SOCKET, MSG, FLAGS, TO)

send(SOCKET, MSG, FLAGS)

Gửi thông báo trên socket.

setsockopt(SOCKET, LEVEL, OPTNAME, OPTVAL)

Đặt các tùy chọn socket được yêu cầu.

shmctl(ID, CMD, ARG)

Gọi hàm IPC **shmctl** của System V. Nếu **CMD** là **&IPC_STAT**, sau đó **ARG** phải là biến giữ cấu trúc **shmid_ds** trả về.

shmget(KEY, SIZE, FLAGS)

Gọi hàm IPC **shmget** của System V. Trả về ID của phần bộ nhớ được chia sẻ, hoặc không xác định nếu có lỗi.

shmread(ID, VAR, POS, SIZE)

shmwrite(ID, STRING, POS, SIZE)

Đọc hay ghi ID phần bộ nhớ được chia sẻ System V, bắt đầu từ vị trí **POS** cho kích thước **SIZE**. Khi đọc, **VAR** phải là biến giữ các dữ liệu đã đọc. Khi ghi, nếu **STRING** quá dài, chỉ có **SIZE** byte được dùng; nếu **STRING** quá ngắn, các ký tự không sẽ được điền đầy **SIZE** bytes.

shutdown(SOCKET, HOW)

Tắt kết nối socket theo cách mà **HOW** quy định, giống như thao tác của lệnh gọi hệ thống cùng tên.

socket(SOCKET, DOMAIN, TYPE, PROTOCOL)

Mở socket loại xác định và nối nó với filehandle **SOCKET**. **DOMAIN**, **TYPE** và **PROTOCOL** được xác định tương tự như trong lệnh gọi hệ thống cùng tên. Có thể phải chạy **h2ph** trên **sys/socket.h** để có giá trị thích hợp trong file thư viện **PERL**.

socketpair(SOCKET1, SOCKET2, DOMAIN, TYPE, PROTOCOL)

Tạo cặp sockets không tên trong **DOMAIN** xác định, với **TYPE** xác định. **DOMAIN**, **TYPE** và **PROTOCOL** được xác định tương tự như trong lệnh gọi hệ thống cùng tên.

▪ *Các thao tác khác*

caller(EXPR)

caller

Trả về ngữ cảnh của lệnh gọi chương trình con hiện thời:

```
($package,$filename,$line) = caller;
```

defined(EXPR)

defined EXPR

Trả về giá trị nhị phân xác định rằng EXPR có giá trị thực hay không.

dump LABEL

Tạo nên core ngay lập tức.

local(LIST)

Khai báo các biến được lập danh sách tới BLOCK, chương trình con cục bộ lệnh **eval** hoặc "do".

require(EXPR)

require EXPR

require

Tính đến các file thư viện được xác định bằng EXPR, hoặc bằng \$_, nếu không có EXPR.

reset(EXPR)

reset EXPR

reset

Thường được dùng trong **continue** BLOCK ở cuối vòng lặp để xóa biến và đặt lại các tìm kiếm cho thao tác tiếp theo.

scalar(EXPR)

Bắt buộc EXPR được giải nghĩa theo ngữ cảnh dữ liệu vô hướng và trả về giá trị của EXPR.

undef(EXPR)

undef EXPR

undef

Làm bất định giá trị của EXPR.

wantarray

Trả về TRUE nếu ngữ cảnh của chương trình con đang chạy hiện thời tìm kiếm giá trị mảng, hoặc FALSE - cho giá trị vô hướng.

VIII. 3. g. Thông tin hệ thống

getpwnam(NAME)
getgrnam(NAME)
gethostbyname(NAME)
getnetbyname(NAME)
getprotobyname(NAME)
getpwuid(UID)
getgrgid(GID)
getservbyname(NAME,PROTO)
gethostbyaddr(ADDR,ADDRTYPE)
getnetbyaddr(ADDR,ADDRTYPE)
getprotobynumber(NUMBER)
getservbyport(PORT,PROTO)
getpwent
getgrent
gethostent
getnetent
getprotoent
getservent
setpwent
setgrent
sethostent(STAYOPEN)
setnetent(STAYOPEN)
setprotoent(STAYOPEN)
setservent(STAYOPEN)
endpwent
endgrent
endhostent
endnetent
endprotoent

endservent

Các chương trình con này thực hiện các chức năng giống hệt như các hàm cùng tên trong thư viện hệ thống.

VIII. 3. h. Quyền ưu tiên

Các toán tử PERL có các liên hệ và ưu tiên sau đây:

```

nonassoc  print printf exec system sort reverse
           chmod chown kill unlink utime die return

left      ,

right     = += -= *= etc.
right     ?:

nonassoc  ..

left      ||

left      &&

left      | ^

left      &

nonassoc  == != <=> eq ne cmp

nonassoc  <; >; <= >= lt gt le ge

nonassoc  chdir exit eval reset sleep rand umask

nonassoc  -r -w -x etc.

left      << >>

left      + - .

left      * / % x

left      =~ !~

right     ! ~ -

right     **

nonassoc  ++ --

left

```

VIII. 3. i. Các chương trình con

Một chương trình con được khai báo như sau:

sub NAME BLOCK

Mọi đối số đến chương trình con được đưa tới như mảng `@_`, nghĩa là (`$_[0]`, `$_[1]`, ...). Mảng `@_` là mảng cục bộ, nhưng các giá trị của nó tham chiếu đến các thông số vô hướng thực. Giá trị trả về của chương trình con là giá trị của biểu thức được tính cuối cùng và có thể là giá trị mảng hay vô hướng. Có thể dùng lệnh **return** để quy định giá trị trả về và kết thúc chương trình con. Để tạo các biến cục bộ, xin bạn tham khảo toán tử **local**.

Gọi chương trình con bằng toán tử **do** hay **&**.

VIII. 4. Chương trình soạn thảo sed

Chương trình **sed** – khác với các trình soạn thảo thông thường trên nền MS Windows – soạn thảo văn bản theo luồng xử lý dữ liệu, tức là lệnh xử lý cùng được đưa vào với file văn bản. Kiểu xử lý dữ liệu văn bản cùng với các lệnh theo luồng như thế này còn được gặp lại trong các chương trình soạn thảo văn bản nổi tiếng khác. Cú pháp của **sed** có dạng:

```
$ sed 'commands' filename(s)
```

với command có dạng chung được chấp nhận như sau:

```
[address[,address]] function [arguments]
```

Các phần tử **address** được dùng để chỉ khoảng các dòng mà **function** sẽ xử lý, còn **function** lại là lệnh được áp dụng, với **argument** là dòng văn bản, ví dụ như phải tìm kiếm và thay thế. Nếu **address** là các số (**m,n**) thì nghĩa là **function** xử lý các dòng văn bản có số thứ tự từ **m** đến **n**. Tuy nhiên, số địa chỉ – tức là số giá trị trở đến các số thứ tự của các dòng văn bản tương ứng mà hiệu lực của lệnh áp dụng tới – bị giới hạn.

Một số lệnh hay dùng của **sed** được minh họa trên bảng sau:

Lệnh	Đối số	Số địa chỉ cực đại	Diễn giải
a	\text	1	Thêm vào cuối đoạn văn đã chọn.
b	label	2	Tới lệnh:command với nhãn đã chọn.
c	\text	2	Đổi nội dung văn bản bằng cách bỏ khoảng trống và chèn nội dung mới ở chỗ đã chỉ.
d		2	Xóa khoảng trống.
D		2	Xóa khoảng trống đầu tiên ở dòng thứ nhất.
i	\text	1	Chèn vào đoạn văn bản ở ra chuẩn.
p		2	In dòng từ file đầu vào.
q		1	Kết thúc sed .
r	file	2	Đọc nội dung file và đưa tới ra chuẩn trước khi đọc mới.
s	/search pattern /replacement /flag	2	Tìm và đổi pattern. flag có 4 giá trị: n=1-512 - chỉ cho lần thứ n; g - cho tất cả các lần gặp; p - in, nếu đã đổi pattern, w - ghi ra file, nếu đã đổi pattern.
y	str1/str2	2	Đổi nội dung str1 bằng str2 tương ứng ở tất cả các lần tìm thấy. Độ dài của str1 và str2 phải như nhau.
!func-tion		2	Dùng function chỉ cho các dòng không có địa chỉ.
:	label	0	Bỏ nhãn cho lệnh b , t
{		2	Thực hiện các lệnh cho tới khi gặp } , nếu pattern được tìm thấy.
#		0	Nếu đứng trước dòng sẽ biến toàn dòng thành chú thích.

Có ba tùy chọn hay được dùng:

-f file

Lấy các lệnh từ file được chỉ tên.

-e script

script được lấy làm lệnh soạn thảo cho **sed**

-n

Bỏ ra chuẩn

Bình thường các lệnh **sed** liệt kê ở bảng trên sẽ được đưa ra thực hiện trong chế độ dòng lệnh, nếu ta không dùng tùy chọn **-f**. Có thể chỉ ra đồng thời nhiều lệnh. Tuy nhiên, mỗi lệnh đều có cách dùng phức tạp nên dễ gây nhầm lẫn. Tốt nhất là ta nên thực hiện tuần tự các lệnh con đó. Ví dụ:

xóa khoảng trống đầu tiên ở dòng thứ hai của file baocao:

```
$ sed `2D` baocao
```

thêm một dòng ở sau dòng thứ ba:

```
3a\
```

chèn thêm văn bản ở vị trí trước dòng thứ năm:

```
5i\
```

```
<văn bản được chèn vào>
```

```
$
```

Dấu gạch xuống “\” được dùng ở đây có ý nghĩa kết thúc hiệu lực lệnh con của **sed** mỗi khi bắt gặp dấu xuống dòng đầu tiên. Nội dung của công việc vừa kể trên có thể đưa vào một file gọi là file lệnh để dùng với tùy chọn **-f**. Trong các mục nhỏ dưới đây ta sẽ xem xét chi tiết hơn cách dùng **sed** với các thao tác cơ bản qua các ví dụ. Nhằm mục đích đó, ta hãy lấy một file văn bản viđused sau làm đối tượng để xử lý trong các ví dụ sẽ được dẫn ra trong phần này:

Các lệnh **sed**

Đây là file văn bản sed được lấy làm đối tượng xử lý cho các lệnh sed.

sed là một trình soạn thảo rất thông dụng được dùng trong các hệ thống UNIX hiện nay.

Người dùng có thể sử dụng sed rất linh hoạt cho các mục đích của mình.

VIII. 4. a. Chèn các dòng hoặc văn bản

Trong file vidused trên, nếu muốn thêm một dòng có nội dung “Xin chào người dùng UNIX” vào đầu file, ta sẽ làm như sau:

```
$ sed '1a\  
> Xin chào người dùng UNIX  
> ' vidused
```

Xin chào người dùng UNIX
Các lệnh sed

Đây là file văn bản sed được lấy làm đối tượng xử lý cho các lệnh sed.

sed là một trình soạn thảo rất thông dụng được dùng trong các hệ thống UNIX hiện nay.

Người dùng có thể sử dụng sed rất linh hoạt cho các mục đích của mình.

\$

Nếu dòng đã cho giả sử cần được chèn vào sau dòng thứ 5, ta thay số 1 ở lệnh **sed** trên bằng số 5.

Như đã được chỉ ra trên bảng liệt kê các lệnh con của **sed**, lệnh con a chỉ được áp dụng cho một địa chỉ (một dòng có số thứ tự nào đó). Trong trường hợp cần thêm

hoặc chèn cả một đoạn văn bản, ta cần dùng lệnh `i`. Ví dụ sau chèn cả văn bản để trong hai dấu ngoặc móc vào *trước* dòng thứ sáu của file `vidused`:

```
$ sed '6i\  
> <van ban can chen vao>  
> 'vidused
```

Các lệnh `sed`

Đây là file văn bản `sed` được lấy làm đối tượng xử lý cho các lệnh `sed`.

`sed` là một trình soạn thảo rất thông dụng được dùng trong các hệ thống UNIX hiện nay.

<van ban can chen vao>

Người dùng có thể sử dụng `sed` rất linh hoạt cho các mục đích của mình.

\$

Bạn đọc có thể tự kết hợp cả hai thao tác trên trong một lần thực hiện lệnh từng bước.

VIII. 4. b. Xóa các dòng văn bản

Thao tác xóa được thực hiện một cách đơn giản như trong ví dụ xóa dòng trước dòng thứ năm như sau:

```
$ sed '5d'  
> 'vidused
```

Các lệnh `sed`

`sed` là một trình soạn thảo rất thông dụng được dùng trong các hệ thống UNIX hiện nay.

Người dùng có thể sử dụng sed rất linh hoạt cho các mục đích của mình.

\$

Ta cũng có thể dùng lệnh xóa những dòng có chứa chuỗi ký tự được chọn. Chuỗi ký tự được chọn của ta ở ví dụ này là "cho các lệnh sed.". Muốn vậy, ta phải gõ lệnh:

```
$ sed '/cho các lệnh sed/d' vidused
```

Đây là file văn bản sed được lấy làm đối tượng xử lý cho các lệnh sed.

\$

Khi sed tìm thấy chuỗi ký tự cần tìm, chuỗi ký tự đó được hiện lên màn hình trước khi lệnh đã được thực hiện.

VIII. 4. c. Thay thế

Thay thế là một thao tác cơ bản hay được dùng của bất cứ trình soạn thảo nào. Giả sử ta cần thay thế từ sed bằng "!sed!" trong file vidused:

```
$ sed 's/sed/"!sed!"/g' vidused
```

Các lệnh "!sed!"

Đây là file văn bản "!sed!" được lấy làm đối tượng xử lý cho các lệnh "!sed!".

"!sed!" là một trình soạn thảo rất thông dụng được dùng trong các hệ thống UNIX hiện nay.

Người dùng có thể sử dụng "!sed!" rất linh hoạt cho các mục đích của mình.

\$

Trong thao tác trên chúng ta đã sử dụng lệnh **s** theo cú pháp:

```
s/xau_ky_tu_tim_kiem/ xau_ky_tu_thay_the/flag
```

với cờ (flag) là **g** để thực hiện thay thế tất cả các trường hợp bắt gặp từ **sed**. Nếu không dùng cờ, việc thay thế chỉ được thực hiện cho lần bắt gặp đầu tiên từ **sed** trong câu mà thôi:

```
$ sed 's/sed/"!sed!"/' vidused
```

Các lệnh **"!sed!"**

Đây là file văn bản **sed** được lấy làm đối tượng xử lý cho các lệnh **sed**.

sed là một trình soạn thảo rất thông dụng được dùng trong các hệ thống UNIX hiện nay.

Người dùng có thể sử dụng **sed** rất linh hoạt cho các mục đích của mình.

\$

VIII. 4. d. Dịch ký hiệu trong văn bản

Nhiều khi xuất hiện nhu cầu đối với người dùng biến file văn bản được tổ chức trong dạng các trường với những dấu phân cách trường thành dạng bình thường hoặc ngược lại. **sed** có thể thực hiện công việc trên bằng hai cách. Lấy ví dụ file **/etc/passwd**, ta thấy các dòng vào có định dạng:

```
root:TtKCmLnvtHctEYa:0:0:Root:/:
daemon:NONE:1:1:Admin:/:
bin:NONE:2:2:Admin:/bin:
sys:NONE:3:3:Admin:/:
```

```
adm :NONE:4:4:Admin:/usr/adm:
ltuan:kjrFDkALJnRTmI:102:100:Le Tuan:/home/ltuan:/bin
```

với các dấu phân cách trường là các dấu hai chấm ":". Ta có thể áp dụng lệnh `s` thay thế các dấu hai chấm trên bằng khoảng trống như sau:

```
$ sed 's/:/ /g' /etc/passwd
root TtKcMlnvthctEYa 0 0 Root /
daemon NONE 1 1 Admin /
bin NONE 2 2 Admin /bin
sys NONE 3 3 Admin /
adm NONE 4 4 Admin /usr/adm
ltuan kjrFDkALJnRTmI 102 100 Le Tuan /home/ltuan /bin
```

Với cách thứ hai bằng lệnh `y`, ta có thể thay một lúc nhiều hơn một ký tự, ví dụ thay bất kỳ ký tự nào của tổ hợp "0:" bằng ký tự tương ứng trong tổ hợp khóa son và dấu chấm than "&!" đối với file `/etc/passwd`. Chú ý rằng tổ hợp "0:" chỉ xuất hiện ở dòng vào đầu tiên cho `root` và dòng thứ sáu cho người dùng `ltuan` trong ví dụ này. (Cũng để ý rằng câu lệnh trên nếu được thực hiện trên máy Solaris x86 sẽ không có khóa `"/g"`):

```
$ sed 'y/0:/&!/g' /etc/passwd
root!TtKcMlnvthctEYa!&!!Root!/!
daemon!NONE!1!1!Admin!/!
bin!NONE!2!2!Admin!/bin!
sys!NONE!3!3!Admin!/!
adm!NONE!4!4!Admin!/usr/adm!
ltuan!kjrFDkALJnRTmI!1&2!1&&!Le Tuan!/home/ltuan!/bin
```

VIII. 4. e. In các dòng văn bản

Trong các thao tác trước ta thường thấy `sed` tự động hiển thị tất cả các thông tin các dòng đầu vào, và cả những gì ta thêm vào, lên thiết bị ra chuẩn (thiết bị mặc định là màn hình). Có thể điều khiển việc in thực tế bằng qua hai lệnh `con p` và `w`.

Các lệnh này thường được kết hợp với tùy chọn `-n`. Nếu chỉ dùng tùy chọn này, `sed` sẽ không hiển thị gì trên màn hình khi thực hiện các thao tác. Nhưng nếu ta dùng với lệnh `con p`, lệnh này sẽ khiến hiển thị những dòng có xâu ký tự cần tìm kiếm đồng thời với việc in ra chúng. Ví dụ:

```
$ sed -n 's/van ban/VAN BAN/p' vidused
```

Đây là file `VAN BAN` `sed` được lấy làm đối tượng xử lý cho các lệnh `sed`.

```
$
```

Lệnh `y` có những hạn chế khi dùng, vì vậy người dùng thường tìm kiếm các phương án ít phức tạp, ví dụ như chỉ đơn giản in ra sáu dòng đầu của file `/etc/passwd`:

```
$ sed -n '1,6p' /etc/passwd
root:TtKCmLnvthctEYa:0:0:Root:/:
daemon:NONE:1:1:Admin:/:
bin:NONE:2:2:Admin:/bin:
sys:NONE:3:3:Admin:/:
adm :NONE:4:4:Admin:/usr/adm:
ltuan:kjrFDkALJnRTmI:102:100:Le Tuan:/home/ltuan: /bin
```

VIII. 5. Emacs

Tác giả chính của Emacs, Richard Stallman, định nghĩa "Emacs là trình soạn thảo mở rộng được, có thể tùy biến, và hiển thị tài liệu theo thời gian thực." Emacs là một trong những sản phẩm nổi bật nhất của dự án GNU. Emacs được biết đến như là một công cụ đa năng, có thể dùng soạn thảo văn bản, đọc E-mail, duyệt Internet, quản lý file, đọc tin trong nhóm người dùng Usenet, duyệt tài liệu trực tuyến và giao tiếp với các tiến trình khác. Emacs còn chứa cả một ngôn ngữ lập trình mạnh có tên là `Elisp`, nên người dùng thậm chí có thể lập trình trong Emacs.

Ở đây, chúng ta quan tâm đến một trong những chức năng cơ bản của Emacs là soạn thảo văn bản. Emacs cung cấp nhiều “chế độ” soạn thảo khác nhau để làm cho việc soạn thảo dễ dàng hơn, ví dụ, chế độ TeX, chế độ C, chế độ HTML và chế độ Lisp.

VIII. 5. a. Tìm hiểu cấu trúc Emacs

▪ Màn hình Emacs

Một cửa sổ Emacs được chia làm bốn phần:

Menubar

Là dòng trên cùng của cửa sổ, chứa danh sách thực đơn. Nếu ấn phím chuột vào một trong số đó, thực đơn tương ứng sẽ được xổ xuống.

Editing Window

Là vùng chính mà bạn dùng để soạn thảo.

Modeline

Là dòng áp chót, chứa các thông tin có ích từ trái sang phải: nếu file được sửa đổi; tên bộ đệm; chế độ soạn thảo chính; chế độ soạn thảo phụ; vị trí của bạn trong bộ đệm. Để tham khảo chi tiết hơn các thông tin trên Modeline, hãy dùng bộ duyệt thông tin Emacs: gõ C-h i m Emacs <Ret> m Mode Line <Ret>.

Minibuffer

Là dòng cuối cùng, nơi bạn gõ các lệnh.

Nếu bạn chạy Emacs trong X Windows bạn cũng sẽ có thanh trượt bên phải của sổ.

▪ Vùng soạn thảo (Editing Area)

Là vùng chính của màn hình, có thể được chia thành các phần, được gọi là các cửa sổ. Mỗi cửa sổ hiển thị một bộ đệm, thường là một file. Emacs có thể có vài cửa sổ được mở cùng một lúc, thậm chí cả khi chúng chẳng hiển thị gì cả. Ví dụ, có thể soạn thảo file LaTeX có tên `indexes.tex` và dùng Emacs chạy LaTeX với nó. Trong trường hợp đó vùng soạn thảo được chia đôi, một phần chứa bộ đệm `indexes.tex`, còn phần kia chứa bộ đệm gọi là `*LaTeX output*`, là kết quả

chạy LaTeX từ Emacs. Có thể mở thêm vài bộ đệm nữa và đổi chỗ giữa các bộ đệm bằng lệnh `C-x b buffername`.

▪ *Khung*

Nếu dùng Emacs trong X Windows thì bạn có thể có nhiều hơn một khung (*frame*) được mở. Khung tức là một cửa sổ Emacs đầy đủ khác. Tuy nhiên, khung sẽ chạy cùng một bản sao Emacs, nghĩa là có cùng các đối tượng đang được xử lý. Mở một khung mới bằng lệnh `C-x 5 2` và đóng bằng lệnh `C-x 5 0`.

Emacs được gọi bằng lệnh **emacs** khi đã được cài đặt trên máy. Trong nhiều hệ UNIX hiện nay Emacs được đưa vào trong bộ cài đặt như một thành phần của hệ điều hành. Phiên bản được dùng hiện nay là 19. Khi muốn chạy Emacs phiên bản 18, ta gõ lệnh **emacs18**, nhưng thường ta được khuyên dùng phiên bản mới, nếu không có nhu cầu cấp thiết phải dùng phiên bản cũ, vì nhiều gói phần mềm mới đòi hỏi phiên bản 19.

Để kết thúc Emacs, ta gõ `C-x C-c`, nghĩa là đồng thời phím `Ctrl` và phím `x`.

VIII. 5. b. *Học cách dùng Emacs*

▪ *Hướng dẫn Emacs*

Emacs có một bộ hướng dẫn nội tại và có giao tiếp với người dùng. Bạn có thể gọi bộ hướng dẫn này thông qua việc chạy Emacs và gọi lệnh `C-h t`). Bộ hướng dẫn này được đánh giá rất cao, và có chứa danh sách các lệnh cơ bản. Hầu như mọi thứ bạn muốn làm Emacs đều có thể xử lý giúp bạn, vì Emacs được coi là trình soạn thảo tận tụy nhất. Bạn chỉ cần tìm và dùng cho đúng lệnh cần thiết mà thôi.

▪ *Danh sách các lệnh của Emacs*

Các lệnh kết thúc, undo và redo

<code>C-g</code>	Kết thúc lệnh hiện thời
<code>C-x u</code>	Undo lệnh vừa được thực hiện
<code>C-u <number> <command></code>	Lặp lại lệnh <code>command</code> với <code>number</code> lần

Các lệnh trợ giúp

C-h	Bắt đầu chương trình trợ giúp
?	Hiện thực đơn Help
a command-apropos	Cho một xâu con và hiện danh sách các lệnh (có giao tiếp) chứa xâu con đó. Hãy xem cả lệnh apropos.
b describe-bindings	Hiện bảng các kết nối.
c describe-key- briefly	Gõ dãy khóa lệnh; sẽ hiển thị tên hàm mà dãy khóa lệnh đó chạy.
f describe-function	Gõ tên hàm và có tài liệu về hàm đó.
C-f Info-goto-emacs- command-node	Gõ tên hàm; bạn sẽ được chỉ đến nút Info cho lệnh đó.
i info	Bộ đọc tài liệu thông tin.
k describe-key	Gõ dãy khóa lệnh; sẽ hiển thị tài liệu đầy đủ.
C-k Info-goto-emacs- key-command-node	Gõ dãy khóa lệnh; bạn sẽ được chỉ đến nút Info cho lệnh được kết nối đến khóa đó.
l view-lossage	Cho thấy 100 ký tự bạn gõ.
m describe-mode	Hiện tài liệu về chế độ chính hiện thời, mô tả các lệnh riêng của nó.
n view-emacs-news	Cho thấy file tin tức về Emacs.
p finder-by-keyword	Tìm gói phần mềm phù hợp với từ khóa đã cho.
s describe-syntax	Hiện bảng cú pháp và chú giải.
t help-with-tutorial	Chọn bộ hướng dẫn Emacs qua thực hành.
v describe-variable	Gõ tên biến; sẽ hiển thị tài liệu về biến đó và giá trị của nó.
w where-is	Gõ tên lệnh; sẽ cho biết các phím nào kích hoạt lệnh đó.
C-c	Hiện bản quyền Emacs (General Public License).
C-d	Hiện thông tin đặt mua Emacs.
C-n	Hiện các tin tức về các thay đổi mới nhất của Emacs.
C-p	Hiện các thông tin về dự án GNU.

C-w	Hiện các thông tin về không chịu trách nhiệm về các hậu quả có liên quan tới GNU Emacs.
M-x info	Hệ thống tài liệu siêu văn bản
M-x apropos	Tìm trực tuyến từ khóa.

Các lệnh di chuyển con trỏ

C-v	Xuống một trang
M-v	Lên một trang
C-M-v	Lên một trang ở cửa sổ khác
M-f	Tiến lên một từ
M-b	Lùi về một từ
M-<	Con trỏ ở đầu văn bản
M->	Con trỏ ở cuối văn bản
C-a	Con trỏ ở đầu dòng
C-e	Con trỏ ở cuối dòng
M-a	Con trỏ ở đầu đoạn văn
M-e	Con trỏ ở cuối đoạn văn
M-m	Con trỏ ở ký tự đầu tiên không phải là khoảng trống
C-x C-x	Thay đổi con trỏ và đánh dấu
C-l	Đặt con trỏ giữa cửa sổ.

Các lệnh xóa và di chuyển

C-k	Xóa đến hết dòng.
C-y	Chèn đoạn văn bản đã bị xóa.
C-<Space bar>	Đánh dấu.
C-x h	Đánh dấu toàn bộ bộ đệm.
C-w	Xóa từ vị trí con trỏ đến vị trí được đánh dấu.
M-w	Xóa từ vị trí con trỏ đến vị trí được đánh dấu vào vòng tròn loại bỏ.
M-<space bar>	Loại bỏ các khoảng trống và dấu Tab, chỉ để lại một.
C-x C-o	Xóa các dòng trống, chỉ để lại một.

M-d	Xóa đến hết từ.
M-<End>	Xóa đến đầu từ.
M-k	Xóa đến cuối câu.
M-z <character>	Xóa đến ký tự <character>.

Các lệnh tìm kiếm và thay thế

C-s	Tìm về phía trước.
C-r	Tìm về phía sau.
C-x C-x	Trở về vị trí bắt đầu tìm kiếm.
M-x replace-string	Thay thế chuỗi ký tự.
M-x query-replace	Thay thế chuỗi ký tự theo yêu cầu.
M-%	Tìm một chuỗi ký tự và thay thế bằng chuỗi ký tự khác. Cả hai chuỗi trên đều được nhắc tới.

Các lệnh đối với file

C-x C-f	Nạp file.
C-x C-r	Nạp file chỉ đọc.
C-x C-s	Lưu giữ file.
C-x C-w	Lưu giữ file với tên khác.
C-x i	Chèn file vào văn bản.

Các lệnh với bộ đệm và cửa sổ

C-x b	Chuyển sang bộ đệm khác.
C-x C-b	Hiện danh sách của tất cả các bộ đệm.
C-x k	Loại bỏ bộ đệm hiện thời.
C-x C-q	Chuyển khóa <i>chỉ đọc</i> cho bộ đệm hiện thời.
C-x 2	Chia cửa sổ hiện thời theo chiều ngang.
C-x 3	Chia cửa sổ hiện thời theo chiều dọc.
C-x 1	Loại bỏ tất cả các cửa sổ, trừ cửa sổ hiện thời.
C-x 0	Loại bỏ cửa sổ hiện thời.

C-x o	Chuyển đến cửa sổ khác.
C-x >	Trượt theo thanh trượt ngang về phía bên phải.
C-x <	Trượt theo thanh trượt ngang về phía bên trái.
C-x ^	Mở rộng cửa sổ hiện thời theo chiều dọc.
C-x +	Điều chỉnh tất cả các cửa sổ.
C-x -	Giảm kích thước cửa sổ cho phù hợp với nội dung của nó.
C-x {	Làm nhỏ cửa sổ theo chiều ngang.
C-x }	Mở rộng cửa sổ theo chiều ngang.

Các lệnh chuyển đổi

M-l	Đổi từ tiếp theo thành chữ thường.
M-u	Đổi từ tiếp theo thành chữ hoa.
C-x C-l	Chuyển vùng được đánh dấu thành chữ thường.
C-x C-u	Chuyển vùng được đánh dấu thành chữ hoa.
M-c	Ký tự đầu tiên viết hoa, còn lại - bằng chữ thường.

Các lệnh khác

<Shift>-<F5>	Hiện đánh số các dòng.
C-t	Thay đổi các ký tự ở vị trí con trỏ.
C-x C-t	Thay đổi các dòng ở vị trí con trỏ.
C-q <character>	Chèn ký tự, có thể là bất phân.
C-x =	Thông tin về ký tự tại vị trí con trỏ.
C-x f	Tạo cột tại vị trí con trỏ khi gõ Enter.
C-x <Tab>	Làm thụt đầu dòng cho vùng được đánh dấu.
M-i	Điền đầy bằng khoảng trống cho đến dấu Tab tiếp theo.
M-!	Thực hiện các lệnh shell.
<Esc> 	Thực hiện các lệnh shell, lấy vùng được đánh dấu làm đầu vào.

Các lệnh macro bàn phím

C-x (	Bắt đầu tạo macro (Emacs bắt đầu ghi nhớ).
C-x)	Kết thúc tạo macro.
C-x e	Thực hiện macro được định nghĩa cuối cùng.
M-x name-last-kbd-macro	Đặt tên cho macro cuối được tạo ra.
M-x insert-kbd-macro	Chèn macro vào bộ đệm hiện thời

Ví dụ, nếu muốn xem toàn bộ danh sách lệnh của Emacs có các chế độ ("mode") trong tên của chúng, ta gõ **C-h a mode**. Để tìm xem hàm `compile` của Emacs làm gì, ta gõ **C-h f compile**, còn xem **C-x C-c** làm gì - phải gõ **C-h k C-x C-c**.

VIII. 5. c. Các chế độ

▪ *Giới thiệu về các chế độ*

Tại mọi thời điểm các bộ đệm trong Emacs đều ở trong một chế độ cụ thể nào đó. Các chế độ khác nhau xác định các tổ hợp phím nóng khác nhau cho thao tác hiện thời của Emacs. Ví dụ, `c-mode` đặt hàm thực đầu dòng C vào dây phím `ESC C-q`, nên bạn có thể gõ các phím đó để làm biểu thức C được viết thụt vào khi bạn ở chế độ `c-mode`. Tên của chế độ hiện thời nằm trong dấu ngoặc đơn ở "dòng chế độ", dòng áp chót trên màn hình Emacs.

Bình thường, Emacs tự động đưa bạn vào chế độ cần thiết khi bạn tạo hay đọc file, dựa trên phần mở rộng của file đó. Ví dụ, nếu phần mở rộng của file là `.c`, Emacs mở file trong chế độ `c-mode`. Nếu bạn mở file `.tex`, Emacs mở trong chế độ `TeX`. Hầu hết các ngôn ngữ lập trình (như C, C++, Lisp, Scheme, Elisp, Perl, awk) đều có chế độ tương ứng riêng của mình.

Bạn có thể bắt buộc Emacs dùng chế độ bất kỳ bằng dây lệnh **M-x**. Chế độ chỉ là hàm của Emacs, nên bạn có thể đặt bộ đệm vào đó, ví dụ như, `tar-mode` bằng lệnh **M-x tar-mode**.

Tài liệu về chế độ hiện thời luôn luôn có thể đọc được bằng lệnh **C-h m**.

▪ *Chế độ Shell*

Nhiều người dùng thích giao tiếp với shell của mình (shell được gọi khi người dùng đó đăng nhập hệ thống UNIX - xem phần *UNIX cho người dùng cuối*) từ trong

bộ đệm của Emacs. Để có thể khởi động shell, ta gõ lệnh **M-x shell**. Điểm lợi là có thể chuyển ngược từ màn hình shell không những về bộ đệm Emacs từ đó shell được gọi, mà còn tới các bộ đệm Emacs khác nữa. Một lợi thế khác là cả đầu vào và kết quả ra của shell đều có thể được sửa đổi bằng Emacs.

- *Chế độ Info*

Xem phần Các thông tin hỗ trợ.

- *Chế độ Dired*

Chế độ dired-mode dành để “soạn thảo” liệt kê nội dung thư mục và được gọi bằng lệnh **C-x d**. Sau khi gọi lệnh, Emacs nhắc ta đưa tên thư mục vào. Chú ý rằng phím Tab cho Emacs biết phải viết nốt tên thư mục mà bạn mới đánh các chữ đầu. Sau đó bạn có thể sao chép, di chuyển, nén, giải nén, sửa đổi, xóa và tìm kiếm các file một cách dễ dàng trong danh sách của thư mục đó.

Chế độ này cũng được dùng để sửa đổi file trong hệ thống file từ xa thông qua cải tiến ftp của Emacs có tên là ANGE-ftp. Để kích hoạt, bạn dùng **C-x C-f** và chỉ cần gõ đầy đủ các đặc tính của file, kể cả tên tài khoản và địa chỉ IP. Ví dụ:

```
C-x C-f /letuan@hut.edu.vn:~ltuan/latex/indexes.tex
```

Bạn phải gõ vào mật khẩu nếu được yêu cầu.

- *Chế độ TeX*

Emacs có các chế độ mặc định cơ bản để soạn thảo các file TeX và LaTeX: **tex-mode** và **LaTeX-mode**, được gọi tự động khi bạn đọc hay tạo file TeX hoặc LaTeX. Ngoài ra còn có các chế độ tốt hơn nhiều được gọi là **AUC-TeX tex-mode** và **LaTeX-mode**. Để chạy được các chế độ này, trước hết bạn phải nạp một file có tên **tex-site** bằng lệnh **M-x load-library tex-site** hay nạp nó trước lúc khởi động (xem phần *Khởi động Emacs trong UNIX*).

Các chế độ AUC-TeX rất mạnh, chứa tất cả các kiểu chữ cơ bản, các lệnh định dạng và môi trường trong các thực đơn xổ xuống để giúp người dùng viết văn bản. Khi đã sẵn sàng chạy LaTeX hay TeX bạn có thể dùng chế độ này để chạy nó bên

trong Emacs, với ưu thế là nếu bạn mắc lỗi, bạn có thể nhảy ngay đến vị trí có lỗi trong file. Ngoài ra bạn còn có một bộ kiểm tra phát âm biết bỏ qua hầu hết các lệnh TeX và LaTeX để chú ý vào nội dung văn bản, v.v..

- *Chế độ HTML*

Chế độ HTML được thiết kế nhằm trợ giúp người dùng viết các siêu văn bản. Bạn có thể dùng nó để đưa vào các ký tự đặc biệt "&", "<" và ">", mà trình duyệt bình thường có thể hiểu sai nghĩa. Tất cả các thao tác chính được thực hiện trong các thực đơn xổ xuống, và cuối cùng bạn có thể kiểm tra lại tài liệu của mình.

- *Chế độ RMAIL*

Emacs có một gói phần mềm đầy đủ dùng cho E-mail có tên là RMAIL. Chú ý: Các gói phần mềm E-mail khác nhau (ví dụ như Pine, mail, và RMAIL) cho ra các loại file khác nhau, đôi khi không tương thích với nhau.

Để bắt đầu RMAIL, gõ lệnh **M-x rmail**.

- *Chế độ Man*

Emacs còn có chế độ để đọc các trang **man** của UNIX. Chỉ cần gõ **M-x man**, và kèm theo tên của trang hướng dẫn bạn muốn tham khảo, và bạn có trang đó trong bộ đệm của Emacs. Ví dụ, **M-x man ps** cho bạn thấy trang **man** của UNIX đối với lệnh **ps**.

- *Chế độ w3*

Emacs có gói phần mềm được gọi là **w3** để duyệt World Wide Web. Trước khi có thể bắt đầu **w3**, bạn phải nạp thư viện **w3** bằng lệnh **M-x load-library w3** (xem phần *Khởi động Emacs trong UNIX*). Vào chế độ **w3** bằng lệnh **M-x w3**.

VIII. 5. d. Các chế độ phụ

Trong khi Emacs chỉ cho phép bạn dùng một chế độ chính trong một bộ đệm thì bạn lại có thể sử dụng nhiều chế độ phụ. Chế độ phụ là các chế độ có thể điều chỉnh (thường là không đáng kể) chế độ chính theo phương cách nào đấy.

Tất cả các chế độ phụ đều được kích hoạt bằng lệnh **M-x**, nhưng bình thường người ta chỉ định chế độ phụ nào sẽ được dùng với chế độ chính nào trong khi khởi động (xem phần *Khởi động Emacs trong UNIX*).

- *Auto Fill*

Đây là một chương trình đơn giản để ngắt dòng sau một tập hợp các ký tự. Emacs mặc định không ngắt dòng cho đến khi bạn ấn phím Enter. Chương trình này sẽ ngắt dòng ở dấu cách (khoảng trống) hay dấu Tab, sau khi độ dài đạt giá trị nhất định. Lệnh **M-x auto-fill-mode** sẽ bật, tắt chức năng này.

- *Line Number*

Chế độ này còn đơn giản hơn nữa. Nó hiển thị số thứ tự của dòng mà con trỏ đang ở đó tại dòng chế độ. Lệnh **M-x line-number-mode** sẽ bật, tắt chức năng này.

- *Chế độ toán học TeX/LaTeX*

Chế độ phụ này là mở rộng của chế độ chính AUC-TeX. Nó cho phép bạn gõ vào các ký hiệu toán học với chỉ một hay hai phím, hoặc dùng thanh thực đơn có hiện thực đơn toán học. Chế độ này được khởi động bằng lệnh **C-c ~** khi đang ở trong chế độ chính TeX hoặc LaTeX.

- *Khóa kiểu chữ*

Khóa kiểu chữ còn phức tạp hơn cả nhiều chế độ chính, nhưng vẫn chỉ là chế độ phụ vì không có đối tượng đích của riêng mình (nhớ lại, ví dụ **c-mode** dùng cho các file **.c**, v.v...). Được dùng để nêu bật các vùng khác nhau của văn bản với các mẫu và kiểu chữ khác nhau. Ví dụ, trong **direct-mode** dùng các mẫu khác nhau cho các thư mục, file văn bản, file thực thi, v.v..., còn trong chế độ TeX hoặc LaTeX bạn có thể có môi trường, chế độ toán học, chế độ văn bản, lệnh và chú thích được bôi mẫu khác nhau.

Điều rắc rối là nói chung, người dùng phải tự thiết lập lấy cho mình khóa kiểu chữ. Nếu bạn là người mới bắt đầu dùng Emacs, tốt hơn cả là bạn hãy sao chép của

những người đi trước. Cấu hình cho khóa kiểu chữ được thực hiện khi khởi động (xem phần Khởi động Emacs trong UNIX).

VIII. 5. e. Khởi động Emacs trong UNIX

Giống như nhiều chương trình UNIX khác, Emacs có file có dấu chấm “.” do người dùng xác định, được gọi tên là `.emacs`, và nếu nó tồn tại, thì sẽ nằm tại thư mục `home` của bạn. Emacs đọc các lệnh trong file này mỗi khi được khởi động, nên sẽ thực hiện theo các tùy biến mà bạn đưa vào.

File `.emacs` được viết bằng ngôn ngữ lập trình của Emacs có tên là `Elisp` (Hãy dùng lệnh **info** đối với `Elisp` để biết rõ hơn về cách viết mã nguồn trên `Elisp`).

Sau đây là ví dụ file `.emacs`:

```
;; Better mode line.
(require 'mode-line)

;; Show line number on modeline
(line-number-mode t)

;; Much better tex-mode (AUC-TeX)
(require 'tex-site)

;; Put goto-line function on C-c C-l
(global-set-key "\C-c\C-l" 'goto-line)

;; Make sure Emacs-on-a-dumb-terminal understands Meta.
(set-input-mode t nil t 7)

;; Set number of characters before auto fill activates
(setq default-fill-column 75)

;; Turn on auto fill in each mode
```

```
(add-hook 'text-mode-hook
  '(lambda () (auto-fill-mode 1)))
(add-hook 'LaTeX-mode-hook
  '(lambda () (auto-fill-mode 1)))

;; a funky auto-fill mode
(require 'filladapt)

;; Highlight marked region
(transient-mark-mode t)
```

VIII. 5. f. *Macro*

Emacs có hệ thống macro đơn giản để kích hoạt chuỗi lệnh. Ta hãy xét một ví dụ với một file sau:

```
Tuan      92
Hai       84
Thanh     66
Minh      93
Dung      88
...
```

Giả sử với một lý do nào đấy, bạn quyết định đổi các tên thành chữ hoa và thêm cho mỗi số dấu gạch phân “.00”. Để làm việc đó bạn phải đặt con trỏ ở đầu mỗi dòng và gõ lệnh **M-u C-e .00** (nghĩa là `upcase-word`, `end-of-line`, và thêm “.00”). Bạn có thể định nghĩa một macro chứa các lệnh trên như sau:

```
C-x ( M-u C-e .00 C-n C-a C-x )
```

Định nghĩa này được bắt đầu bằng **C-x (** (`start-kbd-macro`) và kết thúc với **C-x)** (`end-kbd-macro`). Phần lệnh **M-u .00** là nội dung chính của macro. Để có thể bắt macro làm việc lặp đi lặp lại, trong macro còn có các lệnh định vị lại con trỏ “chuyển đến dòng tiếp theo” và “chuyển đến đầu dòng”.

Macro được gọi bằng lệnh **C-x e** (nghĩa là gọi macro được xác định cuối cùng). Bạn có thể chạy macro mười lần với **M-10 C-x e**. Nói chung, lệnh **M-x n** sẽ lặp lại n lần số mà bạn gõ, ví dụ, **M-3 6** cho ta 666 lần.

VIII. 6. LaTeX

TeX (viết tắt từ chữ Hy Lạp " $\tau\epsilon\chi$ - Tau Epsilon Chi" và có âm gần giống với vần đầu của từ tiếng Anh *technology* - công nghệ, là một hệ thống miễn phí và được thiết kế cho chế bản điện tử, đặc biệt cho các tài liệu chứa nhiều công thức toán học và ký hiệu đặc biệt. Tác giả đầu tiên của TeX là Donald Knuth, tại Stanford University. TeX Users Group (TUG) là tổ chức phi lợi nhuận phát hành các bản cài đặt, cập nhật thường kỳ và tài liệu hướng dẫn kèm theo.

TeX từ lâu nay là chương trình soạn thảo chuẩn được sử dụng rộng rãi ở phương Tây. Ta hãy lấy một ví dụ đơn giản là hầu hết các tạp chí khoa học kỹ thuật đều được chế bản với TeX. Giống như các trình soạn thảo khác như **sed**, **awk**, ... TeX/LaTeX không làm việc ở chế độ tương tác mà xử lý file đầu vào chứa cả dữ liệu và các lệnh được biên dịch để chỉ ra cách thức mà văn bản cần được định dạng.

Ta hãy xem cách thức tạo ra văn bản TeX. Trước tiên ta phải gõ vào văn bản bằng các trình soạn thảo đơn giản đã biết, ví dụ **vi**, **emacs**, v.v., và có các file gọi là file đầu vào hay file TeX. Các file này sẽ được TeX đọc và chúng phải được cất giữ trong dạng ASCII hoặc không phải trong chế độ văn bản, nghĩa là không hề chứa ký tự điều khiển nào. Kết quả đọc của TeX cho ta file *.dvi (DeVice Independent - không phụ thuộc thiết bị) không phải là file đọc, mà chỉ là đầu vào cho một chương trình khác, có tên là trình điều khiển thiết bị, với chức năng tạo ra kết quả có thể đọc được. Sự nhiễu nhê này được biện minh bởi lý do file *.dvi có thể đọc được bằng nhiều loại thiết bị như máy in phun, máy in laser, máy chiếu và máy phóng ảnh, ... một cách rất trung thực với kết quả soạn thảo, nhất là khi có kèm hình ảnh.

Nhiều trung tâm máy tính hiện nay dùng những gói macro TeX khác nhau để làm cho việc viết và in các tài liệu được dễ dàng hơn. Trong số đó trước hết phải kể đến LaTeX, hay còn được gọi là *Generalized Mark-up Language macro package* cho TeX, được viết bởi Leslie Lamport. LaTeX là chương trình xử lý, hay đúng hơn là đánh máy văn bản, dùng chương trình TeX và các macro LaTeX, tạo ra nhiều loại

văn bản một cách nhanh chóng và với chất lượng rất tốt. Nếu bạn cần có một tài liệu đòi hỏi có cấu trúc chương mục, LaTeX sẽ đáp ứng yêu cầu của bạn. Ngược lại, nếu bạn chỉ cần một văn bản ngắn và đơn giản, không có các macro xác định trước của LaTeX, dùng LaTeX ở đây khác nào “đem dao trâu cắt tiết gà” - dùng TeX sẽ tiện lợi hơn.

Không giống như nhiều trình soạn thảo đặt chú ý vào việc định dạng dàn trang văn bản (thường gọi là WYSIWYG - what you see is what you get: bạn nhận được cái mà bạn thấy), LaTeX, chú ý tới “nội dung” của văn bản. Nghĩa là LaTeX xác định cấu trúc của văn bản: các phần, đầu đề, phụ lục, tên tác giả, v.v.. qua các file kiểu dạng văn bản (*.sty) có thể được thay đổi.

LaTeX cũng cho ở đầu ra các file kết quả loại *.dvi, trong đó chứa các lệnh có thể được biên dịch thành các lệnh thích hợp với các thiết bị đầu ra khác nhau. LaTeX còn cho ra các file log (*.log) hay bản ghi chứa thông tin tóm tắt, các thông báo chẩn đoán cho các lỗi phát hiện được trong file đầu vào; và các file phụ trợ (*.aux) chứa mã số tài liệu tham khảo về các bảng, chú thích, v.v.. Các file loại cuối đặc biệt cần thiết nếu ta có các thay đổi liên quan tới việc đánh số các đối tượng như phần, mục, bảng, hình vẽ. Các bản mới nhất hiện nay là LaTeX 2.09 và sau đó là bản được phổ biến LaTeX2e.

Để chạy TeX, trên các máy có cài đặt TeX, ta dùng lệnh sau:

tex filename

với filename là tên file đầu vào có phần mở rộng là .tex. Để dùng LaTeX, ta gõ:

latex filename

cũng với filename có phần mở rộng .tex. Tuy nhiên, hai loại file đầu vào cho TeX và cho LaTeX hoàn toàn khác nhau; chúng không thể hoán đổi cho nhau được.

Sau đây chúng ta chú ý hơn tới cách dùng LaTeX, vì nó hàm chứa nhiều lệnh, macro và tiện ích hơn của TeX nguyên thủy.

VIII. 6. a. Có thể tải TeX/LaTeX về từ đâu

TeX/LaTeX có thể tải về miễn phí bằng anonymous FTP từ kho dữ liệu của CTAN (Comprehensive TeX Archive Network) tại các địa chỉ:

- ftp.shsu.edu
- ftp.dante.de
- ftp.tex.ac.uk

kho dữ liệu này thường nằm ở thư mục /tex-archive.

Ngoài ra còn có nhiều site làm mirror (bản sao gương) mà ta có thể tìm bằng cách dùng lệnh **finger** tới địa chỉ ctan-us@ftp.shsu.edu .

VIII. 6. b. Các đối tượng trong LaTeX

Như đã trình bày ở trên, LaTeX khác TeX nguyên bản ở chỗ nó có rất nhiều các macro định trước. Đó chính là các đối tượng được trình bày trong mục này.

Accenting characters (umlauts, graves, etc.)	Các dấu trên chữ cái (dấu hai chấm phía trên, dấu huyền, ...)
Arrays (math mode)	Mảng (chế độ toán học)
Commands, defining and redefining	Các lệnh xác định và tái xác định
Comments in input file	Chú thích trong file đầu vào
Counters	Bộ đếm
Cross References	Mã số tài liệu tham khảo
Dashes	Gạch ngang
Declarations (changing how things look)	Khai báo
Defining New Commands	Xác định lệnh mới
Document Styles	Kiểu dạng văn bản
Environments (or see list below)	Môi trường
Footnotes	Chú thích

Fonts	Kiểu chữ
Fragile commands (and how to protect them)	Các lệnh nhạy cảm (và cách bảo vệ chúng)
Input to LaTeX	Đầu vào cho LaTeX
International characters and punctuation	Các ký tự quốc tế và dấu chấm câu
Lengths and how to change them	Độ dài và cách thay đổi
Letters (as in "Dear John")	Kiểu thư từ
Line and Page Breaking	Các ngắt dòng và ngắt trang
Lists	Danh sách
Making Paragraphs	Tạo đoạn văn bản
Math Formulas and Math Symbols	Công thức và ký hiệu toán học
Modes (paragraph, left-right, math)	Các chế độ (đoạn văn bản, trái - phải, toán học)
Non-English characters and punctuation	Các ký tự và dấu chấm câu không theo Anh - Mỹ
Page Styles	Kiểu trang
Picture environment	Môi trường cho hình vẽ
Quotation Marks	Dấu hỏi "?"
Sectioning	Phân đoạn
Spaces and Boxes	Khoảng trống và hộp
Special Characters:	Các ký tự đặc biệt:
# \$ % & ~ _ ^ \ { }	# \$ % & ~ _ ^ \ { }
Special Symbols	Các biểu tượng đặc biệt:
Accents on characters	Các dấu trên ký tự (dấu hai chấm ngang, dấu huyền, v.v..)
(umlauts, graves, etc.)	
Arrows (math mode)	Mũi tên (chế độ toán học)
Binary and relational	Các toán tử nhị phân và quan hệ (chế độ toán học)
operators (math mode)	
Delimiters (parenthesis-like objects; math mode)	Các dấu phân cách (giống như dấu ngoặc)
Ellipses	Hình ellip
Greek letters (math mode)	Ký tự Hy Lạp (chế độ toán

Miscellaneous symbols	học)
Math functions (sin, cos, log, ...; math mode)	Các ký hiệu khác
Variable size math symbols (sums, integrals, ...; math mode)	Các hàm toán học (sin, cos, log, ...; chế độ toán học)
Splitting the Input	Các ký hiệu toán học có kích thước thay đổi (dấu tích phân, tổng, ...; chế độ toán học)
Starting and Ending	Phân chia nhỏ đầu vào
Subscripts and Superscripts	Bắt đầu và kết thúc
Table, Tabular and Tabbing Environments	Chỉ số và số mũ
Table of Contents	Môi trường cho bảng, các từ và sự lặp bảng
Terminal Input and Output	Mục lục
Typefaces	Terminal vào và ra
	Kiểu chữ

VIII. 6. c. Các lệnh LaTeX

Lệnh LaTeX được bắt đầu bằng tên, gồm dấu “\” và sau đó là hoặc xâu các chữ cái, hoặc xâu các ký tự không phải là chữ cái. Trong câu lệnh có phân biệt chữ thường và chữ hoa. Lệnh được kết thúc bằng một ký tự không phải là chữ cái, nghĩa là bằng một khoảng trống, một số, một dấu chấm, hoặc bằng ký tự đặc biệt. Các đối số được chứa trong ngoặc vuông “[] ” là tùy chọn, nhưng các đối số trong ngoặc móc “{ }” là bắt buộc. Nói chung, các lệnh của LaTeX đều nên được viết bằng các chữ cái thường, nếu không cần thiết phải có các thay đổi khác. Sau đây là danh sách các lệnh:

<code>\<space></code> (chứa chỗ trống theo space)	<code>\@</code> (theo chu kỳ ngắt câu)
<code>\\</code> (xuống dòng)	<code>\,</code> (chứa chỗ trống nhỏ)
<code>\;</code> (chứa chỗ trống lớn, chế độ toán học)	<code>\:</code> (chứa chỗ trống trung bình, chế độ toán học)
<code>\!</code> (chứa chỗ trống âm nhỏ, chế độ toán học)	<code>\-</code> (tách từ; tạo bảng)

<code>\=</code> (đặt dấu Tab, xem: tạo bảng)	<code>\></code> (dấu Tab, xem: tạo bảng)
<code>\<</code> (đặt dấu Tab ngược, xem: tạo bảng)	<code>\+</code> (xem: tạo bảng)
<code>\'</code> (dấu hoặc tạo bảng)	<code>\`</code> (dấu hoặc tạo bảng)
<code>\ </code> (hai dấu gạch đứng, chế độ toán học)	<code>\(\varnothing \)</code> (xác định môi trường toán học)
<code>\[\varnothing \]</code> (xác định môi trường hiển thị toán học)	<code>\addcontentsline</code>
<code>\addtocontents</code>	<code>\addtocounter</code>
<code>\address</code>	<code>\addtolength</code>
<code>\addvspace</code>	<code>\alph</code>
<code>\appendix</code>	<code>\arabic</code>
<code>\author</code>	<code>\backslash</code>
<code>\baselineskip</code>	<code>\baselinestretch</code>
<code>\bf</code>	<code>\bibitem</code>
<code>\bigskip</code>	<code>\boldmath</code>
<code>\cal</code>	<code>\caption</code>
<code>\cdots</code>	<code>\centering</code>
<code>\circle</code>	<code>\cite</code>
<code>\cleardoublepage</code>	<code>\clearpage</code>
<code>\cline</code>	<code>\closing</code>
<code>\dashbox</code>	<code>\date</code>
<code>\ddots</code>	<code>\dotfill</code>
<code>\em</code>	<code>\ensuremath</code> (LaTeX2e)
<code>\fbox</code>	<code>\flushbottom</code>
<code>\fnsymbol</code>	<code>\footnote</code>
<code>\footnotemark</code>	<code>\footnotesize</code>
<code>\footnotetext</code>	<code>\frac</code>
<code>\frame</code>	<code>\framebox</code>
<code>\frenchspacing</code>	<code>\hfill</code>
<code>\hline</code>	<code>\hrulefill</code>
<code>\hspace</code>	<code>\huge</code>
<code>\Huge</code> (chữ hoa đầu tiên "H")	<code>\hyphenation</code>

<code>\include</code>	<code>\includeonly</code>
<code>\indent</code>	<code>\input</code>
<code>\it</code>	<code>\item</code>
<code>\kill</code>	<code>\label</code>
<code>\large</code>	<code>\Large</code> (chữ hoa đầu "L")
<code>\LARGE</code> (tất cả chữ viết hoa)	<code>\ldots</code>
<code>\left</code>	<code>\lefteqn</code>
<code>\line</code>	<code>\linebreak</code>
<code>\linethickness</code>	<code>\linewidth</code>
<code>\location</code>	<code>\makebox</code>
<code>\maketitle</code>	<code>\markboth</code> <code>\markright</code>
<code>\mathcal</code>	<code>\mathop</code>
<code>\mbox</code>	<code>\medskip</code>
<code>\multicolumn</code>	<code>\multiput</code>
<code>\newcommand</code>	<code>\newcounter</code>
<code>\newenvironment</code>	<code>\newfont</code>
<code>\newlength</code>	<code>\newline</code>
<code>\newpage</code>	<code>\newsavebox</code>
<code>\newtheorem</code>	<code>\nocite</code>
<code>\noindent</code>	<code>\nolinebreak</code>
<code>\normalsize</code> (mặc định)	<code>\nopagebreak</code>
<code>\not</code>	<code>\onecolumn</code>
<code>\opening</code>	<code>\oval</code>
<code>\overbrace</code>	<code>\overline</code>
<code>\pagebreak</code>	<code>\pagenumbering</code>
<code>\pageref</code>	<code>\pagestyle</code>
<code>\par</code>	<code>\parbox</code>
<code>\parindent</code>	<code>\parskip</code>
<code>\protect</code>	<code>\providecommand</code> (của LaTeX2c)
<code>\put</code>	<code>\raggedbottom</code>
<code>\raggedleft</code>	<code>\raggedright</code>
<code>\raisebox</code>	<code>\ref</code>
<code>\renewcommand</code>	<code>\right</code>

<code>\rm</code>	<code>\roman</code>
<code>\rule</code>	<code>\savebox</code>
<code>\sbox</code>	<code>\sc</code>
<code>\scriptsize</code>	<code>\setcounter</code>
<code>\setlength</code>	<code>\settowidth</code>
<code>\sf</code>	<code>\shortstack</code>
<code>\signature</code>	<code>\sl</code>
<code>\small</code>	<code>\smallskip</code>
<code>\sqrt</code>	<code>\stackrel</code>
<code>\tableofcontents</code>	<code>\telephone</code>
<code>\textwidth</code>	<code>\textheight</code>
<code>\thanks</code>	<code>\thispagestyle</code>
<code>\tiny</code>	<code>\title</code>
<code>\today</code>	<code>\tt</code>
<code>\twocolumn</code>	<code>\typeout</code>
<code>\typein</code>	<code>\underbrace</code>
<code>\underline</code>	<code>\unitlength</code>
<code>\usebox</code>	<code>\usecounter</code>
<code>\value</code>	<code>\vdots</code>
<code>\vector</code>	<code>\verb</code>
<code>\vfill</code>	<code>\vline</code>
<code>\vphantom</code>	<code>\vspace</code>

Sau đây, ta sẽ xem xét kỹ hơn một vài nhóm lệnh đáng chú ý.

- *Thêm lệnh đầu tiên* `\newcommand`, `\renewcommand`

Dùng để xác định hoặc tái xác định nội dung của một lệnh và có cú pháp:

```
\newcommand{cmd} [args] [opt] {def}
\renewcommand{cmd} [args] [opt] {def}
\providecommand{cmd} [args] [opt] {def} -- LaTeX2e
```

trong đó, các thành phần có ý nghĩa như sau:

cmd

Tên của lệnh mới hoặc lệnh được tái xác định, gồm dấu “\”, tiếp theo sau là một xâu có thể toàn chữ cái thường, có thể toàn là chữ hoa hoặc hỗn hợp, và kết thúc bởi một ký tự đơn không phải là chữ cái. Đối với lệnh `\newcommand` tên phải là chưa được xác định trước hoặc không được bắt đầu với `\end`, với `\renewcommand` tên lại phải là đã được xác định.

Lệnh `\providecommand` tương tự như lệnh `\newcommand` nếu lệnh với tên trên chưa tồn tại. Nếu tên lệnh trên đã có, `\providecommand` sẽ không làm gì cả và định nghĩa cũ của lệnh vẫn còn hiệu lực.

args

Số nguyên từ 1 tới 9 quy định số đối số mà dùng cho lệnh sẽ được xác định. Giá trị mặc định là không có đối số.

opt

(Chỉ có ở LaTeX2e) Nếu tồn tại, đối số đầu tiên được xác định bởi `args` sẽ là tùy chọn với giá trị mặc định `opt`; nếu không có mặt, tất cả các đối số là bắt buộc.

def

Đoạn văn bản sẽ được thay thế mỗi khi bắt gặp `cmd`; thông số với dạng form `#n` trong `cmd` sẽ bị thay thế bởi đoạn văn bản của đối số thứ `n` khi sự thay thế được thực hiện.

Ví dụ, lệnh:

```
\newcommand{\water}{H$_2$O}
```

cho phép ta viết:

```
Cong thuc hoa hoc cua nuoc la \water.
```

hoặc:

```
water\ la cong thuc hoa hoc cua nuoc.
```

Chú ý rằng trong cách viết thứ hai dấu “\” phải được đi kèm khoảng trống ngay sau nó để đảm bảo có quãng cách trước “H₂O”.

Ví dụ thứ hai:

```
\newcommand{\hypotenuse}{\$a^{2}+b^{2}\$}
```

sẽ tạo ra công thức a^2+b^2 trong chế độ văn bản do có cặp dấu “\$...\$” trong định nghĩa. Trong chế độ toán học, dấu “\$” đầu tiên lại làm LaTeX thoát khỏi chế độ toán học, và gây ra lỗi. Trong LaTeX 2.09 cách chuyển sang chế độ toán học là dùng với `\mbox`, *viz.*,

```
\newcommand{\hypotenuse}{\mbox{\$a^{2}+b^{2}\$}}
```

Trong LaTeX2e lệnh `\ensuremath` được cung cấp để giải quyết vấn đề trên. Đối số của lệnh `\ensuremath` luôn luôn được xử lý trong chế độ toán học, bất kể đang ở chế độ hiện thời nào:

```
\newcommand{\hypotenuse}{\ensuremath{a^{2}+y^{2}}}
```

▪ *Tên và các định hoặc tái xác định môi trường*

Cú pháp của lệnh:

```
\newenvironment{nam}[args]{begdef}{enddef}
\renewenvironment{nam}[args]{begdef}{enddef}
```

với:

nam

Tên của thông số môi trường, phải là cho đến hiện thời chưa được xác định đối với `\newenvironment`, hoặc đối với lệnh `\nam`. Đối với lệnh `\renewenvironment` tên “nam” phải là đã được xác định. nam có thể chứa chữ cái, chữ số, dấu sao “*”, nhưng không được bắt đầu với “end...”.
args

Số nguyên từ 1 tới 9 xác định số đối số của thông số môi trường mới được định nghĩa. Giá trị mặc định là không có đối số.

begdef

Đoạn văn bản thay thế cho mỗi lần xuất hiện của `\begin{nam}`; thông số có dạng `#n` trong `begdef` được thay thế bởi đoạn văn bản của đối số thứ `n` khi thực hiện thay thế.

enddef

Đoạn văn bản thay thế cho mỗi lần xuất hiện của `\end{nam}`. Không chứa các thông số của đối số.

Trong LaTeX2e có một thay đổi: đối số `opt` được đưa vào cú pháp của lệnh. Nếu tồn tại, đối số đầu tiên được xác định bởi `args` sẽ là tùy chọn với giá trị mặc định `opt`; nếu không có mặt, tất cả các đối số là bắt buộc.

```
\newenvironment{nam} [args] [opt] {begdef} {enddef}
\renewenvironment{nam} [args] [opt] {begdef} {enddef}
```

• *Tệnh xác định môi trường dạng định lý*

Trong khi trình bày các tài liệu khoa học, giáo trình, có nhiều khi ta cần nhấn mạnh một kết luận, một luận điểm, một định lý, v.v.. TeX/LaTeX có khả năng tạo ra môi trường đáp ứng nhu cầu trên thông qua lệnh có cú pháp như sau:

```
\newtheorem{env_name} {caption} [within]
\newtheorem{env_name} [numbered_like] {caption}
```

với các thành phần được giải nghĩa dưới đây:

env_name

Tên của môi trường, gồm một xâu các chữ cái, không trùng với môi trường hay bộ đếm đã tồn tại.

caption

Văn bản được in ở đầu môi trường, ngay trước chữ số.

within

Tên của bộ đếm đã có, thường là đơn vị theo các phần tài liệu. Cung cấp phương tiện đặt mới lại việc đánh số các định lý.

numbered_like

Tên của môi trường dạng định lý đã tồn tại.

Lệnh `\newtheorem` có tối đa là một đối số tùy chọn.

- *Lệnh xác định kiểu chữ mới*

Cú pháp:

```
\newfont{cmd}{font_name}
```

Xác định tên lệnh `cmd`, hiện thời chưa được định nghĩa, để khai báo rằng kiểu chữ có tên `font_name` sẽ trở thành kiểu chữ hiện thời.

VIII. 6. d. Bắt đầu chế bản

Các file LaTeX được bắt đầu trong LaTeX với lệnh `\documentclass` để xác định loại văn bản cần soạn thảo. Ví dụ, nếu muốn dùng định dạng *article* theo tiêu chuẩn AMS cho các tạp chí, và đưa kèm vào một số ký hiệu toán học và các biểu đồ giao hoán, bạn phải gõ:

```
\documentclass[11pt]{article}
\usepackage{amstex, amssymb, amscd}
...
```

Ngoài `article`, các thể loại khác là `book`, `report` dùng tốt cho việc trình bày luận án, viết sách và `letter` cho thư từ.

(Trong LaTeX v.2.09 còn có thể chọn loại văn bản trong số `article`, `book`, `letter` và `report`, với các tùy chọn mặc định, bằng lệnh tương tự:

```
\documentstyle[options]{style}
```

Các tùy chọn mặc định cụ thể cho từng loại là:

1. `article`: 11pt, 12pt, twoside, twocolumn, draft, fleqn, leqno, acm
2. `report`: 11pt, 12pt, twoside, twocolumn, draft, fleqn, leqno, acm
3. `letter`: 11pt, 12pt, fleqn, leqno, acm
4. `book`: 11pt, 12pt, twoside, twocolumn, draft, fleqn, leqno

Nếu bạn muốn quy định thêm một tùy chọn, thì chúng phải được viết cách nhau bằng dấu phẩy.)

Gói phần mềm `amstex` sẽ nạp phần lớn các đặc điểm của AMS-**L^AT_EX**, trong khi `amssymb` nạp một số ký hiệu toán học bổ sung, và `amscd` nạp các lệnh đặc biệt để vẽ các biểu đồ giao hoán.

Một lựa chọn khác của lệnh `\documentclass` cho kết quả gần với dạng chuẩn AMS cho tạp chí là:

```
\documentclass{amsart}
```

Lớp văn bản này tự động nạp gói phần mềm `amstex` nên không còn cần thiết phải gõ lệnh kế tiếp `\usepackage{amstex}`.

Cỡ chữ mặc định trong `article` là 10 (đơn vị tính bằng point, tức 1/72 inch), nếu ta cảm thấy cỡ đó là quá nhỏ, có thể đặt lại cỡ 11 hoặc 12.

Nội dung văn bản mà bạn gõ trong **L^AT_EX** phải nằm giữa hai dòng `\begin{document}` và `\end{document}`, nên cấu trúc cơ bản của tài liệu được soạn thảo có dạng:

```
\documentclass[11pt]{article}
\usepackage{amstex}

\begin{document}
```

Đây là bài viết rất quan trọng và thu hút vì về đề tài

(Phần nội dung còn lại của tài liệu viết dưới dạng bài báo)

```
\end{document}
```

VIII. 6. e. Các lệnh trong chế độ văn bản

▪ Các kiểu và cỡ chữ

Theo mặc định, tất cả các chữ bạn gõ được dùng phông chữ roman. Ta có thể chọn các kiểu khác nhau, ví dụ để nhấn mạnh, để làm đầu đề bảng biểu, v.v.. như sau:

<code>\textbf{ho hoan kiem}</code>	bold face: ho hoan kiem
<code>\textit{ho hoan kiem}</code>	italic: <i>ho hoan kiem</i>
<code>\textsf{ho hoan kiem}</code>	sans serif: ho hoan kiem
<code>\textsl{ho hoan kiem}</code>	slanted: <i>ho hoan kiem</i>
<code>\texttt{ho hoan kiem}</code>	typewriter: ho hoan kiem
<code>\emph{ho hoan kiem}</code>	emphasized: kết hợp roman và italic
<code>\sc{ho hoan kiem}</code>	small caps: HO HOAN KIEM

Để thay đổi cho một vài chữ trong câu, ví dụ trong:

```
...Thus, we define a \emph{projective representation}
to be...
```

ta có:

```
...Thus, we define a projective representation to be...
```

Nếu lựa chọn văn bản lớn hơn (ví dụ: lớn hơn một đoạn văn bản được giới hạn bằng dấu xuống dòng), ta phải đặt nó vào trong khai báo `\textem...{}`. Đó chính là sơ đồ chọn kiểu chữ NFSS2 (New Font Selection Scheme) mà LaTeX2 mở rộng TeX. Bảng sau tổng kết các lệnh khác nhau để chuyển giữa các phông chữ mặc định sẵn có trong LaTeX2:

Hiệu ứng	Dạng cục bộ	Dạng chung
Họ phông chữ roman serif	<code>\textrm{...}</code>	<code>{\rmfamily...}</code>
Họ phông chữ sans serif	<code>\textsf{...}</code>	<code>{\sffamily...}</code>
Họ phông chữ đánh máy (typewriter)	<code>\texttt{...}</code>	<code>{\ttfamily...}</code>
Bộ phông chữ in đậm (bold face)	<code>\textbf{...}</code>	<code>{\bfseries...}</code>
Bộ phông chữ in đậm vừa (medium face)	<code>\textmd{...}</code>	<code>{\mdseries...}</code>
Dạng phông chữ có phía phải nhô cao hơn (upright)	<code>\textup{...}</code>	<code>{\upshape...}</code>
Dạng in nghiêng (italic)	<code>\textit{...}</code>	<code>{\itshape...}</code>
Dạng chú ý đánh dấu (slant)	<code>\textsl{...}</code>	<code>{\slshape...}</code>
Dạng chữ viết hoa cỡ nhỏ (SMALL CAPS)	<code>\textsc{...}</code>	<code>{\scshape...}</code>
Phông chữ cho văn bản	<code>\textnormal{...}</code>	<code>{\normalfont...}</code>

Cỡ chữ có thể thay đổi bằng lệnh “dạng cục bộ” `\cochu{...}` và “dạng chung” `{\cochu...}`, với `cochu` có thể là bất kỳ trong:

<code>\tiny</code>	<code>cochu</code>	<code>\normalsize</code>	<code>cochu</code>
<code>\scriptsize</code>	<code>cochu</code>	<code>\large</code>	<code>cochu</code>
<code>\footnotesize</code>	<code>cochu</code>	<code>\Large</code>	<code>cochu</code>
<code>\small</code>	<code>cochu</code>	<code>\LARGE</code>	<code>cochu</code>

Trong các trường hợp đặc biệt, bạn thậm chí có thể tăng cỡ chữ đến:

`\huge`

cochu

`\Huge`

cochu

Chú ý rằng không có `\HUGE` !

▪ *Tế môi trường*

Trong LaTeX phương pháp chủ yếu để điều khiển dạng của tài liệu là dùng cặp lệnh tương hợp `\begin{environment}` và `\end{environment}`, hay còn gọi là môi trường (*environment*). Môi trường có thể ẩn. Các giá trị của *environment* có ở trong danh sách sau:

<code>abstract</code>	<code>array</code>
<code>center</code>	<code>description</code>
<code>displaymath</code>	<code>enumerate</code>
<code>eqnarray</code>	<code>equation</code>
<code>figure</code>	<code>flushleft</code>
<code>flushright</code>	<code>itemize</code>
<code>list</code>	<code>math</code>
<code>minipage</code>	<code>picture</code>
<code>quotation</code>	<code>quote</code>
<code>tabbing</code>	<code>table</code>
<code>tabular</code>	<code>thebibliography</code>
<code>theorem</code>	<code>titlepage</code>
<code>trivlist</code>	<code>verbatim</code>
<code>verse</code>	

Các quy tắc bỏ dấu phụ thuộc vào chế độ được sử dụng: chế độ văn bản, chế độ toán học hay môi trường lập bảng. Dưới đây ta sẽ xem xét cụ thể các quy tắc đó.

Chế độ văn bản

Ta có các dấu sau có thể đặt lên chữ cái . Để đỡ rối ta thống nhất lấy "a" làm chữ cái để đặt ký hiệu dấu, nhưng về nguyên tắc, có thể đặt lên bất cứ ký tự nào. Thậm chí dấu có thể đặt được lên các chữ cái "bị bỏ qua", ví dụ `\~{ }` sẽ viết dấu ngã "~" lên trên khoảng trống. Các lệnh sau chỉ dùng được trong đoạn văn hoặc trong chế độ LR:

<code>\`{a}</code>	dấu huyền	: à
<code>\' {a}</code>	dấu sắc	: á
<code>\^{a}</code>	dấu mũ	: â
<code>\" {a}</code>	dấu hai chấm ngang	: ä
<code>\H{A}</code>	dấu hai chấm ngang kiểu Hungary	: Ä
<code>\~{a}</code>	dấu ngã	: ã
<code>\c{a}</code>	dấu ngoặc dưới (chữ có chân)	: ạ
<code>\={A}</code>	dấu gạch ngang phía trên	: Ą
<code>\b{a}</code>	dấu gạch chân	: <u>a</u>
<code>\. {a}</code>	dấu chấm trên chữ cái	: ă
<code>\d{a}</code>	dấu chấm dưới chữ cái	: ȃ
<code>\u{o}</code>	dấu ngân (nhạc) trên chữ cái	: ơ
<code>\v{a}</code>	dấu mũ trên chữ cái	: â
<code>\t{aa}</code>	nối hai chữ cái bằng chữ u ngược	: ȳ aa

Riêng đối với các chữ cái "i" và "j" dấu sắc cần phải bỏ một cách thận trọng vì thường người ta cũng hay thay dấu chấm bằng dấu sắc.

Chế độ toán học

Vài loại dấu ở trên và các loại tương tự khác cũng có thể dùng được trong chế độ toán học. Sau đây là các loại dấu chỉ dùng được trong chế độ toán học:

<code>\hat{o}</code>	dấu mũ (xem <code>\^</code>)	: ô
<code>\widehat{oo}</code>	dấu mũ liền cho vài chữ cái	: ồ
<code>\check{o}</code>	kiểm tra (xem <code>\v</code>)	: ố
<code>\tilde{o}</code>	dấu ngã (xem <code>\~</code>)	: ơ
<code>\widetilde{oo}</code>	dấu ngã liền cho vài chữ cái	: ồ
<code>\acute{o}</code>	dấu sắc (xem <code>\'</code>)	: ó
<code>\grave{o}</code>	dấu huyền (xem <code>\`</code>)	: ò
<code>\dot{e}</code>	dấu chấm trên chữ cái (xem <code>\.</code>)	: é
<code>\ddot{o}</code>	dấu hai chấm ngang trên chữ cái	: ö
<code>\breve{a}</code>	dấu ngân (nhạc) trên chữ cái (xem <code>\u</code>)	: ă

<code>\bar{o}</code>	dấu macron trên chữ cái (xem <code>\=</code>)	: $\bar{o}$
<code>\vec{o}</code>	dấu vectơ trên chữ cái	: $\vec{o}$

Trong bảng biểu

Một vài dấu trong chế độ văn bản có ý nghĩa khác trong môi trường lập bảng như sau:

<code>\a'</code>	dấu sắc	: á
<code>\a`</code>	dấu huyền	: à
<code>\a=</code>	dấu gạch ngang phía trên	: ā

▪ Định lý, tiên đề và giả thiết

LaTeX cung cấp biện pháp đơn giản để viết các định lý, tiên đề và giả thiết, v.v.. Đối với mỗi loại trên ta phải tạo môi trường bằng các dùng lệnh `\newtheorem`. Ví dụ:

```
\newtheorem{prop}{Proposition}
```

tạo lập môi trường có tên `prop` cho kiểu xếp chữ Đề xuất - Proposition. Sau khi lệnh đó được gọi, bạn có thể gõ:

```
\begin{prop}
    This is the text of my proposition.
\end{prop}
```

để được:

Proposition 1. *This is the text of my proposition.*

Chú ý rằng “đề xuất” trên được tự động đánh số và xếp chữ in nghiêng.

Lệnh `\newtheorem` có thể nằm bất kỳ ở đâu trong file sau khi môi trường được sử dụng lần thứ nhất, nhưng tiện lợi hơn là để phía đầu văn bản, giữa hai dòng lệnh `\documentstyle` và `\begin{document}`. (Văn bản không được phép

viết trước `\begin{document}`}, nhưng các lệnh không tạo đầu ra như `\newtheorem` thì được phép viết).

Có thể đặt tên cho định lý bằng cách viết lệnh `\begin` với đối số trong ngoặc vuông. Ví dụ,

```
\newtheorem{thm}{Theorem}
\begin{thm}[Main Theorem]
This is the main theorem.
\end{thm}
```

cho:

Theorem 1 (Main Theorem). *This is the main theorem.*

▪ *Danh sách*

Hãy tưởng tượng rằng bạn cần lập danh sách các điều kiện của định lý. Ta phải dùng môi trường `enumerate`. Mỗi một điều kiện được đưa vào bằng lệnh `\item`. Ví dụ:

```
\begin{enumerate}
\item  $M$  is complete
\item  $f$  is proper
\end{enumerate}
```

cho bạn:

1. M is complete.
2. f is proper.

▪ *Căn lề chính giữa và gạch chân*

Chỉ cần gõ lệnh:

```
\begin{center}
```



```
This will be centered!
\end{center}
```

LaTeX sẽ cho:

```
This will be centered!
```

Lệnh gạch chân cũng được thực hiện tương tự, nhưng với môi trường có tên `underline`.

▪ *Tạo khoảng trống*

Thông thường LaTeX thận trọng khi làm việc với những khoảng trống để trang văn bản được trình bày đúng. Tuy nhiên, có khi bạn thấy cần phải tạo khoảng trống, ví dụ như để sẵn chỗ trống cho các hình vẽ, công thức, v.v., sẽ điền vào sau. Để làm được việc này cần dùng tới lệnh `\vskip`, với đối số viết liền ngay sau chữ `"p"` (không cách một khoảng trắng). Ví dụ,

```
...the following diagram.
\vskip100pt
```

để tạo khoảng cách 100 points giữa dòng kết thúc bằng từ `"diagram"` và khối văn bản tiếp theo. Đối số cũng có thể được đặt với đơn vị là inch (1 inch $\approx$ 2,54 cm), như `\vskip.25in`.

Nếu bạn cố tạo khoảng trống nhiều hơn khoảng không gian còn lại cho đến cuối trang thì lệnh chỉ cho nhảy đến cuối trang mà thôi. Để thêm khoảng trống ở đầu trang thì phải dùng lệnh `\vglue` thay vì `\vskip`. Sau đây là vài lệnh khác có liên quan tới khoảng trống. Xin nhắc lại, trong trường hợp bình thường, tự LaTeX chăm lo tới việc tạo các khoảng trống một cách chính xác.

<code>\newpage</code>	tạo khoảng trống và nhảy sang trang mới.
<code>\newline</code>	nhảy sang dòng tiếp theo.
<code>\linebreak</code>	Tốt nhất là giải thích qua ví dụ:

```
...This is the end of the line.
```

```
\linebreak
```

```
This is where I want to start the next line.
```

sẽ tạo văn bản:

```
...This is the end of the line.
```

```
This is where I want to start the next line.
```

▪ *Tạo bảng*

Để tạo bảng cần dùng môi trường tabular. Ví dụ:

```
\begin{center}
\begin{tabular}{c|ccccccc}
 $\lambda$  & (5) & (4,1) & (3,2) & (3,1,1) & (2,2,1) & (2,1,1,1) & (1,1,1,1,1) \\
\hline
 $d_\lambda$  & 1 & 4 & 5 & 6 & 5 & 4 & 1
\end{tabular}
\end{center}
```

cho bảng:

λ	(5)	(4,1)	(3,2)	(3,1,1)	(2,2,1)	(2,1,1,1)	(1,1,1,1,1)
d_λ	1	4	5	6	5	4	1

Xâu `{c|ccccccc}` mô tả bố trí của bảng. Tám chữ `c` chỉ rằng bảng có tám cột, và dữ liệu ghi ở mỗi cột đều được căn chính giữa. Bạn cũng có thể dùng các chữ cái `l` và `r` để chỉ cách căn lề trái và căn lề phải. Dấu gạch đứng “|” chỉ rằng có nét gạch đứng giữa các cột thứ nhất và thứ hai, và sau cột thứ tám. Bảng nằm trong môi trường `center` có nghĩa là được đặt chính giữa trang.

Nội dung văn bản của bảng tự thân xuất hiện sau xâu chỉ sự sắp xếp của bảng mà ta vừa đề cập tới ở trên. Các dữ liệu của bảng được phân cách bởi các dấu khóa son “&”, và mỗi dòng được kết thúc bởi lệnh xuống dòng (carriage return) “\\”.

Lệnh `\hline` tạo một đường kẻ ngang suốt chiều rộng bảng giữa hai hàng. Nếu muốn tạo thêm những đường kẻ ngang, ta phải đặt thêm lệnh `\hline` vào những vị trí tương ứng.

▪ *Phân chia phần văn bản và mã số tài liệu tham khảo*

LaTeX được thiết kế để làm việc với các văn bản lớn, do đó nó cung cấp các lệnh để tạo ra tiêu đề cho các phần, để đánh số tự động các phần.

Để bắt đầu một phần văn bản mới, ta dùng lệnh `\section{ }` như trong ví dụ sau:

```
\section{Introductory Remarks}
```

Ta còn có thể dùng các lệnh `\subsection{ }` và `\subsubsection{ }` để chia phần văn bản thành từng đoạn nhỏ hơn. Trong các kiểu dạng văn bản `report` và `book`, còn có lệnh `\chapter{ }` để bắt đầu một chương, mỗi một trong số đó lại có thể có các lệnh `\section`, `\subsection`, v.v..

LaTeX tự gánh vác việc đánh số chương, mục, hình vẽ, bảng biểu...Thêm nữa, nó cung cấp cơ chế tạo các mã số tài liệu tham khảo có thể cập nhật tự động khi thay đổi việc đánh số thông qua các lệnh `\label{ }` và `\ref{ }`. Lệnh `\label{ }` liên kết các số thứ tự của phần, phần nhỏ, phần nhỏ nữa, v.v.. Số thứ tự có thể được gọi lại từ bất kỳ chỗ nào trong văn bản bằng cách gõ lệnh `\ref{ }` với khóa xác định trong các ngoặc móc. Thậm chí khi số thứ tự của phần văn bản bị thay đổi, các tài liệu tham khảo vẫn được liệt kê đúng.

Một vài điểm cần được chú ý khi dùng các lệnh `\label{ }` và `\ref{ }`:

1. Khóa có thể chứa chữ thường và chữ hoa, chữ số và dấu chấm, nhưng không được chứa khoảng trống
2. Có phân biệt chữ thường và chữ hoa. Ví dụ `\ref{Sectioning}` khác với `\ref{sectioning}`.
3. Có thể dùng lệnh `\ref{ }` để tham chiếu đến khóa trước khi nó được xác định trong file của bạn. Khi lần đầu LaTeX đọc file của bạn, chương

trình sẽ hiểu rằng nó chưa biết khóa tham chiếu đến cái gì, nhưng sau khi chương trình đã nhớ đích tham chiếu đến lần cuối của khóa, nó sẽ dùng giá trị đó. Vậy, nếu dùng `\twice` trên file đầu vào, bạn có thể tin tưởng rằng tất cả mã số tài liệu tham khảo đều sẽ được cập nhật.

4. Nếu bạn dùng lệnh `\label{ }` trong phạm vi môi trường `enumerate`, hoặc trong môi trường được tạo bằng lệnh `\newtheorem`, thì sau đó khóa sẽ được liên kết với các mục được đánh số, hay với số thứ tự của định lý (tiên đề v.v.).

▪ *Thực lục sách tham khảo*

Tốt nhất là ta xét ví dụ minh họa sau:

```
\begin{thebibliography}{[AHU]}
\bibitem[AHU]{ahu} Aho, A.,\ Hopcroft, J.,\ and Ullman,
J.\ (1976). {\em{The Design and Analysis of
Computer Algorithms.}} Addison Wesley, Reading,
Mass.
\bibitem[AT]{AT} Auslander, L. and Tolmieri, R.(1979).
Is Computing with the Fast Fourier Transform Pure
or Applied Mathematics? {\em{Bulletin (New Series)
of the AMS Vol. 1}}847-897.
\end{thebibliography}
```

Các dòng mã của ví dụ trên cho ta:

Bibliography

- [AHU] Aho, A., Hopcroft, J., and Ullman, J. (1976). *The Design and Analysis of Computer Algorithms*. Addison Wesley, Reading, Mass.
- [AT] Auslander, L. and Tolmieri, R. (1979). Is Computing with the Fast Fourier Transform Pure or Applied Mathematics? *Bulletin (New Series) of the AMS Vol. 1* 847-897.


```
...
...
\end{array}
```

Môi trường `array`, tương tự như môi trường `tabular`, tạo ra hộp chứa dãy các dòng được giống theo các cột dọc, và chỉ được dùng trong phạm vi môi trường `math`.

Chỉ có một đối số bắt buộc để mô tả số cột và cách căn lề trong mỗi một cột đó. Mỗi chữ `v` trong đối số dùng để xác định cột và một trong các chữ `l`, `c`, hoặc `r` chỉ rằng cột phải được dàn tương ứng theo bên trái, chính giữa hay bên phải. Có thể xem chi tiết hơn trong môi trường `tabular` cũng được tổ chức theo dạng trên.

Đối số không bắt buộc `pos` xác định cách bố trí của cả mảng trong môi trường bao quanh (nhắc lại rằng LaTeX xử lý các hộp như các đối tượng toàn thể, dù hộp có lớn đến đâu đi chăng nữa). Các tùy chọn cho đối số này là:

`t` - căn trên dòng trên cùng

`b` - căn trên dòng dưới cùng

Giá trị mặc định là căn chính giữa trang.

Các cột phải được phân cách bằng dấu “&”. Các phần tử của cột có thể chứa các lệnh LaTeX khác, thậm chí cả các mảng khác. Mỗi hàng của mảng phải được kết thúc bằng dấu “\\”. Chú ý rằng không có dấu “&” sau phần tử cuối cùng của hàng, cũng như không có dấu “\\” sau dòng cuối cùng của mảng. (Một ngoại lệ là khi `\hline` được dùng để đặt dòng ở dưới, thì nó đứng sau “\\”).

Mỗi một phần tử trong mảng được xử lý như nó đứng trong dấu ngoặc. Vì vậy, một khai báo trong phạm vi phần tử nào đó của mảng thì là cục bộ đối với phần tử ấy, nghĩa là phạm vi hiệu lực của nó được giới hạn cho đến dấu “&” và “\\”.

Các đường nằm ngang và đường thẳng đứng được tạo bằng các lệnh tương ứng:

```
\cline
```

```
\hline
```

```
\vline
```

▪ *Các phương trình có đánh số thứ tự*

Cần nhắc lại rằng việc gọi chế độ toán học trong dòng được thực hiện bằng cách dùng các dấu "\$" đơn như sau: \$... \$, và hiện các phương trình - thông qua các dấu kép "\$\$" như \$\$... \$\$. Để viết các phương trình có đánh số thứ tự, ta dùng môi trường `equation`, ví dụ,

```
\begin{equation}
\chi(M_g) = 2-2g,
\end{equation}
```

sẽ cho:

$$(1) \qquad \chi(M_g) = 2 - 2g.$$

Chú ý rằng tự môi trường `equation` sẽ đưa bạn vào chế độ toán học mà không cần phải dùng dấu "\$\$... \$\$" ở xung quanh `\begin{equation}` ... `\end{equation}`.

Nếu bạn muốn tham chiếu đến phương trình có đánh số thứ tự, bạn cần phải đặt lệnh `\label{ }` vào giữa `\begin{equation}` và `\end{equation}`. Sau đó khóa sẽ được liên kết với số thứ tự của phương trình, và bạn có thể tham chiếu đến phương trình trên bằng cách gõ:

...as we see from (`\ref{eqnkey}`).

▪ *Chỉ số và số mũ*

Chỉ số được viết nhờ chỉ thị bằng dấu gạch dưới "_". Ví dụ:

<code>\$\alpha_j\$</code>	cho	α_j
<code>\$\alpha^j\$</code>	cho	α^j
<code>\$\alpha_j^i\$</code>	cho	α_j^i
<code>\$\alpha_{\{j^i\}}\$</code>	cho	$\alpha_{\{j^i\}}$

Ta cần chú ý khi dùng ngoặc móc “{ }” để viết chỉ số và số mũ, vì trong chế độ toán học có nhiều cách nhóm chúng lại với nhau. Nhiều lệnh của chế độ toán học được định nghĩa để hoạt động trên đối tượng xuất hiện tiếp theo trong file: hoặc là chữ cái đơn, hoặc là một từ lệnh, hay một nhóm được giới hạn trong ngoặc. Ví dụ, toán tử viết chỉ số “_” có thể được dùng viết chỉ số cho một chữ cái đơn ($\$a_b\$$) hay cho một lệnh đơn ($\$a_beta\$$). Nhưng nếu bạn muốn viết chỉ số một cách phức tạp hơn, ví dụ $a_{k+\beta}$, bạn phải dùng dấu ngoặc: $\$a_{\{b+\beta\}}\$$.

Một ví dụ khác, ngoặc móc phải được dùng nếu bạn muốn chỉ số được viết thấp hơn và nhỏ hơn. Nếu bạn viết:

$$\alpha_{k_j}.$$

thì toán tử được áp dụng hai lần và bạn có chỉ số bậc hai: a_{k_j} .

Biểu thức α_{k_j} (không có dấu ngoặc móc) sẽ gây lỗi.

▪ Văn bản trong chế độ toán học

Trong chế độ toán học, LaTeX sẽ bỏ qua mọi khoảng trống. Tồn tại riêng các quy tắc phức tạp bên trong mà LaTeX dùng để xác định cách mà phương trình được bố trí các quãng cách cần thiết, ví dụ, có khoảng cách xung quanh các dấu “+” và “=”. Tuy nhiên, đôi khi bạn lại muốn đặt vài từ vào giữa biểu thức phương trình. Ví dụ, nếu muốn viết:

$$\{\psi \mid \psi \text{ is an additive character}\}.$$

Ta phải dùng lệnh `\text{ }` để báo cho LaTeX lập tức quay trở lại chế độ văn bản. Trong ví dụ này, ta phải gõ lệnh:

```
$$ \{\psi \mid \psi \text{ is an additive character}\}
$$.
```

Lúc đó, LaTeX sẽ đặt “is an additive character” với khoảng trống ở phía trước trong chế độ văn bản. Chú ý rằng nếu ta bỏ đi khoảng trống trước chữ “is”, ta sẽ nhận được kết quả với lỗi chế bản thường gặp:

$\{\psi \mid \psi \text{ is an additive character}\}.$

Phải thận trọng để sử dụng chế độ toán học cho đúng. Trong chế độ toán học LaTeX cho rằng mọi xâu các chữ cái tiếp theo một xâu khác đều là *kết quả*, và do đó sẽ bố trí xâu được nói tới một cách tương ứng. Thực vậy, nếu đó là một từ thì ta không nhận thấy thật rõ. Nhưng nếu đối với các ký hiệu chỉ là một phần của các từ, như là *Stab*, *Spec*, *Proj*, v.v., bạn phải dùng lệnh `\operatorname{ }` để báo LaTeX sắp xếp các chữ cái trong các ký hiệu đó như văn bản, chứ không phải là kết quả. Ví dụ, nếu gõ:

```
$$ X := \operatorname{Spec} A $$
```

ta có:

$$X := \operatorname{Spec} A$$

Bạn có thể viết chỉ số và số mũ trong các ký hiệu được tạo ra thông qua lệnh `\operatorname{ }`, ví dụ, `$\operatorname{Stab}_G(H)$` cho ta $\operatorname{Stab}_G(H)$

▪ Các ký hiệu đặc biệt

Chúng ta đã dùng khá nhiều ví dụ trong LaTeX với các ký hiệu sẵn có trong TeX - `\alpha`, `\psi`, v.v. Một số ký hiệu khác được các gói phần mềm `amstex` và `amssymb` đưa vào, trong đó gồm có cả các chữ hoa viết bằng (**C**, **R**, **Z**), các chữ kiểu *caligraphic* (**A**, **B**, **C**) và kiểu *Fraktur* (**ℒ**, **℔**, **℔**). Các lệnh dùng ở đây là `\mathbb`, `\mathcal` và `\mathfrak`, là những lệnh chỉ có hiệu lực trong chế độ toán học và chỉ tác dụng lên chữ cái hoặc nhóm chữ cái đi liền ngay sau. Ví dụ,

$$\mathfrak{S} = \mathbb{R}^2/G$$

được tạo bằng lệnh `$\mathfrak{S} = \mathbb{R}^2/G$`.

Để tạo ra bảng đầy đủ sau gồm các ký hiệu toán học (chữ cái Hy Lạp và các biểu tượng khác), sẵn sàng cho việc sử dụng trong file văn bản của bạn, phải tính đến việc gọi lệnh nạp các gói phần mềm tương ứng sau ngay sau lệnh `\documentclass`:

```
\usepackage{amstex, amssymb}
```

α	<code>\alpha</code>	β	<code>\beta</code>
γ	<code>\gamma</code>	δ	<code>\delta</code>
ϵ	<code>\epsilon</code>	ε	<code>\varepsilon</code>
ζ	<code>\zeta</code>	η	<code>\eta</code>
θ	<code>\theta</code>	ϑ	<code>\vartheta</code>
ι	<code>\iota</code>	κ	<code>\kappa</code>
λ	<code>\lambda</code>	μ	<code>\mu</code>
ν	<code>\nu</code>	ξ	<code>\xi</code>
ϕ	<code>\phi</code>	π	<code>\pi</code>
φ	<code>\varphi</code>	ρ	<code>\rho</code>
ϱ	<code>\varrho</code>	σ	<code>\sigma</code>
ς	<code>\varsigma</code>	τ	<code>\tau</code>
υ	<code>\upsilon</code>	ϕ	<code>\phi</code>
φ	<code>\varphi</code>	χ	<code>\chi</code>
ψ	<code>\psi</code>	ω	<code>\omega</code>
Γ	<code>\Gamma</code>	Δ	<code>\Delta</code>
Θ	<code>\Theta</code>	Λ	<code>\Lambda</code>
Ξ	<code>\Xi</code>	Π	<code>\Pi</code>
Σ	<code>\Sigma</code>	Υ	<code>\Upsilon</code>
Φ	<code>\Phi</code>	Ψ	<code>\Psi</code>
Ω	<code>\Omega</code>	$\leq$	<code>\leq</code>
$\geq$	<code>\geq</code>	$\equiv$	<code>\equiv</code>
$\models$	<code>\models</code>	$\prec$	<code>\prec</code>
$\perp$	<code>\perp</code>	$\sim$	<code>\sim</code>
$\lrcorner$	<code>\lrcorner</code>	$\preceq$	<code>\preceq</code>
$\simeq$	<code>\simeq</code>	$\simeq$	<code>\simeq</code>
$\mid$	<code>\mid</code>	$\nmid$	<code>\nmid</code>
$\gg$	<code>\gg</code>	$\asymp$	<code>\asymp</code>
$\parallel$	<code>\parallel</code>	$\subset$	<code>\subset</code>
$\supset$	<code>\supset</code>	$\approx$	<code>\approx</code>
$\bowtie$	<code>\bowtie</code>	$\subseteq$	<code>\subseteq</code>
$\supseteq$	<code>\supseteq</code>	$\cong$	<code>\cong</code>
$\sqsubset$	<code>\sqsubset</code>	$\sqsupset$	<code>\sqsupset</code>
$\neq$	<code>\neq</code>	$\smile$	<code>\smile</code>
$\sqsubseteq$	<code>\sqsubseteq</code>	$\sqsupseteq$	<code>\sqsupseteq</code>
$\doteq$	<code>\doteq</code>	$\frown$	<code>\frown</code>
$\in$	<code>\in</code>	$\ni$	<code>\ni</code>
$\propto$	<code>\propto</code>	$\vdash$	<code>\vdash</code>
$\dashv$	<code>\dashv</code>	$\leftarrow$	<code>\leftarrow</code>
$\longleftarrow$	<code>\longleftarrow</code>	$\uparrow$	<code>\uparrow</code>
$\Leftarrow$	<code>\Leftarrow</code>	$\Longleftarrow$	<code>\Longleftarrow</code>
$\Uparrow$	<code>\Uparrow</code>	$\rightarrow$	<code>\rightarrow</code>
$\longrightarrow$	<code>\longrightarrow</code>	$\downarrow$	<code>\downarrow</code>
$\Rightarrow$	<code>\Rightarrow</code>	$\Longrightarrow$	<code>\Longrightarrow</code>
$\Downarrow$	<code>\Downarrow</code>	$\leftrightarrow$	<code>\leftrightarrow</code>

$\lesssim$	<code>\lessapprox</code>	$\leqslant$	<code>\leqslantless</code>
$\leqslantgtr$	<code>\leqslantgtr</code>	$\curlyeqprec$	<code>\curlyeqprec</code>
$\curlyeqsucc$	<code>\curlyeqsucc</code>	$\preccurlyeq$	<code>\preccurlyeq</code>
$\leqq$	<code>\leqq</code>	$\leqslant$	<code>\leqslant</code>
$\lessgtr$	<code>\lessgtr</code>	$\backprime$	<code>\backprime</code>
$\risingdotseq$	<code>\risingdotseq</code>	$\fallingdotseq$	<code>\fallingdotseq</code>
$\succcurlyeq$	<code>\succcurlyeq</code>	$\geqq$	<code>\geqq</code>
$\geqslant$	<code>\geqslant</code>	$\gtrless$	<code>\gtrless</code>
$\vartriangleright$	<code>\vartriangleright</code>	$\vartriangleleft$	<code>\vartriangleleft</code>
$\trianglerighteq$	<code>\trianglerighteq</code>	$\trianglelefteq$	<code>\trianglelefteq</code>
$\bigstar$	<code>\bigstar</code>	$\between$	<code>\between</code>
$\blacktriangledown$	<code>\blacktriangledown</code>	$\blacktriangleright$	<code>\blacktriangleright</code>
$\blacktriangleleft$	<code>\blacktriangleleft</code>	$\vartriangle$	<code>\vartriangle</code>
$\blacktriangle$	<code>\blacktriangle</code>	$\triangledown$	<code>\triangledown</code>
$\eqcirc$	<code>\eqcirc</code>	$\lesseqgtr$	<code>\lesseqgtr</code>
$\gtreqless$	<code>\gtreqless</code>	$\lesseqqgtr$	<code>\lesseqqgtr</code>
$\gtreqqless$	<code>\gtreqqless</code>	$\Rrightarrow$	<code>\Rrightarrow</code>
$\Lleftarrow$	<code>\Lleftarrow</code>	$\veebar$	<code>\veebar</code>
$\barwedge$	<code>\barwedge</code>	$\doublebarwedge$	<code>\doublebarwedge</code>
$\measuredangle$	<code>\measuredangle</code>	$\sphericalangle$	<code>\sphericalangle</code>
$\varpropto$	<code>\varpropto</code>	$\smile$	<code>\smile</code>
$\smallfrown$	<code>\smallfrown</code>	$\subset$	<code>\subset</code>
$\supset$	<code>\supset</code>	$\cup$	<code>\cup</code>
$\Cap$	<code>\Cap</code>	$\curlywedge$	<code>\curlywedge</code>
$\curlyvee$	<code>\curlyvee</code>	$\leftthreetimes$	<code>\leftthreetimes</code>
$\rightthreetimes$	<code>\rightthreetimes</code>	$\subseteq$	<code>\subseteq</code>
$\supseteq$	<code>\supseteq</code>	$\bumpeq$	<code>\bumpeq</code>
$\Bumpeq$	<code>\Bumpeq</code>	$\lll$	<code>\lll</code>
$\ggg$	<code>\ggg</code>	$\circledS$	<code>\circledS</code>
$\pitchfork$	<code>\pitchfork</code>	$\dotplus$	<code>\dotplus</code>
$\backsim$	<code>\backsim</code>	$\backsimeq$	<code>\backsimeq</code>
$\complement$	<code>\complement</code>	$\intercal$	<code>\intercal</code>
$\circledcirc$	<code>\circledcirc</code>	$\circledast$	<code>\circledast</code>
$\circledddash$	<code>\circledddash</code>	$\lvertneqq$	<code>\lvertneqq</code>
$\gvertneqq$	<code>\gvertneqq</code>	$\lneq$	<code>\lneq</code>
$\ngeq$	<code>\ngeq</code>	$\lnless$	<code>\lnless</code>
$\ngtr$	<code>\ngtr</code>	$\lnprec$	<code>\lnprec</code>
$\nsucc$	<code>\nsucc</code>	$\lneqq$	<code>\lneqq</code>
$\gneqq$	<code>\gneqq</code>	$\lneqslant$	<code>\lneqslant</code>
$\ngeqslant$	<code>\ngeqslant</code>	$\lneq$	<code>\lneq</code>
$\gneq$	<code>\gneq</code>	$\lnpreceq$	<code>\lnpreceq</code>
$\nsucceq$	<code>\nsucceq</code>	$\precnsim$	<code>\precnsim</code>
$\succnsim$	<code>\succnsim</code>	$\lnsim$	<code>\lnsim</code>
$\gnsim$	<code>\gnsim</code>	$\lnleqq$	<code>\lnleqq</code>

$\nlessapprox$	<code>\ngeqq</code>	$\nlessapprox$	<code>\precneqq</code>
$\nsuccapprox$	<code>\succneqq</code>	$\nsuccapprox$	<code>\precnapprox</code>
$\nlessapprox$	<code>\succnapprox</code>	$\lnapprox$	<code>\lnapprox</code>
$\nlessapprox$	<code>\gnapprox</code>	$\nsim$	<code>\nsim</code>
$\ncong$	<code>\ncong</code>	$\diagup$	<code>\diagup</code>
$\diagdown$	<code>\diagdown</code>	$\varsubsetneq$	<code>\varsubsetneq</code>
$\varsupsetneq$	<code>\varsupsetneq</code>	$\subsetneqq$	<code>\subsetneqq</code>
$\nsupseteqq$	<code>\nsupseteqq</code>	$\subsetneqq$	<code>\subsetneqq</code>
$\supsetneqq$	<code>\supsetneqq</code>	$\varsubsetneqq$	<code>\varsubsetneqq</code>
$\varsupsetneqq$	<code>\varsupsetneqq</code>	$\subsetneq$	<code>\subsetneq</code>
$\supsetneq$	<code>\supsetneq</code>	$\nsupseteq$	<code>\nsupseteq</code>
$\nsupseteq$	<code>\nsupseteq</code>	$\nparallel$	<code>\nparallel</code>
$\nmid$	<code>\nmid</code>	$\nshortmid$	<code>\nshortmid</code>
$\nshortparallel$	<code>\nshortparallel</code>	$\nvdash$	<code>\nvdash</code>
$\nVdash$	<code>\nVdash</code>	$\nVDash$	<code>\nVDash</code>
$\nVDash$	<code>\nVDash</code>	$\ntrianglerighteq$	<code>\ntrianglerighteq</code>
$\ntrianglelefteq$	<code>\ntrianglelefteq</code>	$\ntriangleleft$	<code>\ntriangleleft</code>
$\ntriangleright$	<code>\ntriangleright</code>	$\nleftarrow$	<code>\nleftarrow</code>
$\nrightarrow$	<code>\nrightarrow</code>	$\nLeftarrow$	<code>\nLeftarrow</code>
$\nRrightarrow$	<code>\nRrightarrow</code>	$\nLeftrightarrow$	<code>\nLeftrightarrow</code>
$\nleftrightarrow$	<code>\nleftrightarrow</code>	$\divideontimes$	<code>\divideontimes</code>
$\varnothing$	<code>\varnothing</code>	$\nexists$	<code>\nexists</code>
$\Finv$	<code>\Finv</code>	$\Game$	<code>\Game</code>
$\eth$	<code>\eth</code>	$\eqsim$	<code>\eqsim</code>
$\beth$	<code>\beth</code>	$\gimel$	<code>\gimel</code>
$\daleth$	<code>\daleth</code>	$\lessdot$	<code>\lessdot</code>
$\gtrdot$	<code>\gtrdot</code>	$\ltimes$	<code>\ltimes</code>
$\rtimes$	<code>\rtimes</code>	$\shortmid$	<code>\shortmid</code>
$\shortparallel$	<code>\shortparallel</code>	$\smallsetminus$	<code>\smallsetminus</code>
$\thicksim$	<code>\thicksim</code>	$\thickapprox$	<code>\thickapprox</code>
$\approx$	<code>\approx</code>	$\succapprox$	<code>\succapprox</code>
$\precapprox$	<code>\precapprox</code>	$\curvearrowleft$	<code>\curvearrowleft</code>
$\curvearrowright$	<code>\curvearrowright</code>	$\digamma$	<code>\digamma</code>
$\varkappa$	<code>\varkappa</code>	$\Bbbk$	<code>\Bbbk</code>
$\hslash$	<code>\hslash</code>	$\backepsilon$	<code>\backepsilon</code>

▪ Lấy các phương trình

Để viết được các phương trình xếp hàng đẹp mắt, ta dùng lệnh `\eqnarray*` như trong ví dụ sau:

```
\begin{eqnarray*}
```

```

i_w\cdot\psi_v\cdot N_w&=&\psi_w\cdot i_w\cdot N_w\\
&=&\psi_w\cdot i_w\cdot\sum_{\{\sigma\in G_w\}}\sigma\\
&=&\psi_w\cdot\sum_{\{\sigma\in G_w\}}\sigma\\
&-&i_w\cdot N_w\cdot\psi_w.
\end{eqnarray*}

```

để được dãy phương trình:

$$\begin{aligned}
 i_w \cdot \psi_v \cdot N_w &= \psi_w \cdot i_w \cdot N_w \\
 &= \psi_w \cdot i_w \cdot \sum_{v \in G_w} \sigma \\
 &= \psi_w \cdot \sum_{v \in G_w} \sigma \\
 &= \sum_{v \in G_w} \sigma \cdot \psi_w \\
 &= i_w \cdot N_w \cdot \psi_w
 \end{aligned}$$

Trong khi sử dụng kiểu viết này ta cần chú ý mấy điểm sau:

- Các dấu “&” bao quanh ký hiệu mà dựa vào đó các phương trình được xếp hàng - trong ví dụ trên ký hiệu đó là dấu bằng “=”.
- Các dòng được kết thúc bằng “\\”, hay còn gọi là dấu xuống dòng.

▪ *Tiết phân số*

Nếu bạn cần viết biểu thức dạng phân số, hãy dùng lệnh **\frac**. Lệnh **\frac** tác dụng lên mọi nội dung chứa trong hai dấu ngoặc móc, hoặc lên toàn bộ phương trình nếu không có đôi ngoặc móc đó. Trong ví dụ sau là biểu thức định nghĩa đạo hàm bậc nhất của hàm $f(x)$ nào đó:

$$\lim_{\{\delta \rightarrow 0\}} \frac{f(x+\delta) - f(x)}{\delta}$$

cho ta kết quả được trình bày dưới đây:

$$\lim_{\delta \rightarrow 0} \frac{f(x+\delta) - f(x)}{\delta}$$

▪ *Phương dùng dấu ngoặc.*

Trong trường hợp phải viết một biểu thức phức tạp, để kết quả được trình bày đẹp mắt và đúng theo quy ước, các dấu ngoặc phía ngoài phải có kích thước lớn hơn các dấu phía trong. Ta có thể làm như vậy đối với bất kỳ cặp dấu ngoặc tròn, ngoặc vuông và ngoặc móc nào như trong ví dụ sau:

```
$$ \Biggl( \biggl( \Bigl( \bigl( ( ) \bigr) \Bigr) \biggr) \Biggr)
    \biggr) \Biggr) $$
```

và được:

$$\left(\left(\left(\left(\right)\right)\right)\right)$$

Câu lệnh tương tự được dùng cho ngoặc vuông “[” và ngoặc móc “{”. Chú ý rằng đối với dấu ngoặc móc phải có thêm dấu “\” đi liền phía trước.

Bạn cũng có thể viết cặp dấu ngoặc tròn đủ to để ôm vừa được biểu thức chứa bên trong. Một ví dụ dùng \left(và \right) được diễn tả dưới đây như sau:

$$$$ \left(\sum a_i^2\right)^{1/2} $$$$

sẽ cho: $\left(\sum a_i^2\right)^{1/2}$.

Tất nhiên, hiệu ứng trên cũng có đối với các lệnh cho ngoặc vuông và ngoặc móc \left[, \right], \left\{ và \right\}.

■ *Ma trận*

Có thể tạo ra nhiều kiểu ma trận trong LaTeX. Đoạn mã sau:

```


$$\begin{matrix}
a_{11} & 0 & \ldots & a_{1n} \\
0 & a_{22} & \ldots & a_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \ldots & a_{nn}
\end{matrix}$$


```

sẽ tạo ra ma trận:

$$\begin{matrix}
a_{11} & 0 & \ldots & a_{1n} \\
0 & a_{22} & \ldots & a_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \ldots & a_{nn}
\end{matrix}$$

Cũng phải ghi nhớ rằng mỗi một phần tử ma trận được ngăn cách bằng dấu "&", mỗi một dòng ma trận được kết thúc bằng dấu xuống dòng "\\ ", và các quy tắc sau:

<code>\ldots</code>	Dấu ba chấm ngang "..."
<code>\vdots</code>	Dấu ba chấm dọc "..."
<code>\ddots</code>	Dấu ba chấm theo đường chéo "..."

và mỗi dấu trên được xem như một phần tử ma trận.

Để viết đường gạch thẳng đứng, các dấu ngoặc, v.v., xung quanh phần tử ma trận, ta dùng các môi trường `vmatrix`, `pmatrix`, hay `bmatrix` tương ứng.

Cuối cùng, để viết ma trận với các chú thích bên rìa, LaTeX có lệnh `\bordermatrix {}`, như được minh họa trong ví dụ tiếp theo:

```


$$\begin{matrix} & \text{corner} & c_1 & c_2 & \dots & c_n \\ r_1 & a_{11} & 0 & \dots & a_{1n} \\ r_2 & 0 & a_{22} & \dots & a_{2n} \\ r_3 & \vdots & \vdots & \ddots & \vdots \\ r_4 & 0 & 0 & \dots & a_{nn} \end{matrix}$$


```

cho:

$$\begin{matrix} & c_1 & c_2 & \dots & c_n \\ r_1 & a_{11} & 0 & \dots & a_{1n} \\ r_2 & 0 & a_{22} & \dots & a_{2n} \\ r_3 & \vdots & \vdots & \ddots & \vdots \\ r_4 & 0 & 0 & \dots & a_{nn} \end{matrix}$$

Chú ý rằng nếu thế vào chỗ phần tử "corner" xuất hiện ở góc trái trên cùng bạn muốn bỏ trống hoặc viết một từ khác thì phải bỏ trống hay viết từ đó vào chỗ tương ứng trong đoạn mã trên.

▪ *Tình chia trường hợp tùy theo tình huống*

Để có một biểu thức có nhiều trường hợp tình huống như trong ví dụ:

$$x_{\lambda} = \begin{cases} x & \text{if } \lambda \text{ is an eigenvalue;} \\ -x & \text{if } -\lambda \text{ is an eigenvalue;} \\ 0 & \text{otherwise.} \end{cases}$$

ta viết:

```


$$x_{\lambda} = \begin{cases} x & \text{if } \lambda \text{ is an eigenvalue;} \\ -x & \text{if } -\lambda \text{ is an eigenvalue;} \\ 0 & \text{otherwise.} \end{cases}$$


```



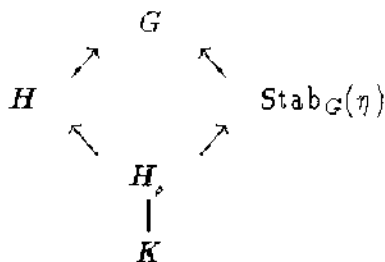
```
\end{cases}
$$
```

▪ Biểu đồ

Việc tạo các biểu đồ không khó khăn lắm trong LaTeX, nếu ta quan niệm biểu đồ tương tự như một mảng hai chiều mà mỗi một phần tử của nó hoặc được diễn một ký hiệu, hoặc để trống. Khái niệm “mảng” đã được trình bày trong mục riêng. Xét ví dụ sau:

```
$$\begin{array}{rcccl}
\; & \& \; & G & \& \; & & \& \; & \\
\; & \& \; & \nearrow & \& \; & \nwarrow & \& \; & \\
H & \& \; & \; & \& \; & \text{Stab}_G(\eta) & \& \; & \\
\; & \& \; & \nwarrow & \& \; & \nearrow & \& \; & \\
\; & \& \; & H_{\rho} & \& \; & & \& \; & \\
\; & \& \; & \mid & \& \; & & \& \; & \\
\; & \& \; & K & \& \; & & \& \; & \\
\end{array}$$
```

Đôi số {rcccl} hoạt động tương tự như đôi số đối với môi trường tabular đã trình bày. Trong ví dụ này {rcccl} xác định có năm cột, cột đầu tiên căn lề phải, các cột giữa căn lề chính giữa, cột cuối cùng căn lề trái. Giống như trong môi trường tabular, mỗi dòng được kết thúc bằng dấu “\\”, và dấu khóa son “&” ngăn cách các phần tử trong dòng đã nói trên. Ví ta dựng biểu đồ nên một số phần tử là mũi tên, các ký hiệu và rất nhiều khoảng trống. Lệnh được viết ở trên sẽ tạo ra biểu đồ sau:



LaTeX bao gồm cả bộ tạo tự động các biểu đồ giao hoán hình chữ nhật được chứa trong AMS-**TEX** khi bạn nạp gói phần mềm `amscd`.

Nếu đã nạp gói phần mềm trên, bạn có thể gõ vào máy tính những dòng lệnh của ví dụ:

```
\begin{CD}
H^2(K_{nr}/K) @>{\operatorname{Res}}>> H^2(L_{nr}/L) \\
@V{\operatorname{inv}}_KV V @VV{\operatorname{inv}}_LV \\
{\mathbb Q}/{\mathbb Z} @>n>> {\mathbb Q}/{\mathbb Z}
\end{CD}
```

để tạo biểu đồ sau:

$$\begin{array}{ccc} H^2(K_{nr}/K) & \xrightarrow{\operatorname{Res}} & H^2(L_{nr}/L) \\ \operatorname{inv}_K \downarrow & & \downarrow \operatorname{inv}_L \\ \mathbb{Q}/\mathbb{Z} & \xrightarrow{n} & \mathbb{Q}/\mathbb{Z} \end{array}$$

Bạn nên đọc phần tiếp ngay sau đây để biết cách tránh phải gõ thường xuyên từ bàn phím những lệnh như `\operatorname`, ...

VIII. 6. g. *Làm đơn giản bớt các thao tác chế bản*

LaTeX cung cấp khả năng xếp chữ để tạo ra hầu như mọi loại biểu thức toán học mà bạn đã từng biết đến trong các tài liệu, tạp chí, v.v.. Tuy nhiên, việc lặp lại các động tác gõ nhiều lần các ký hiệu lệnh để viết các biểu thức toán học phức tạp và dài chẳng hề làm cho ai dễ chịu cả. Chỉ đơn cử một ví dụ: những người hay làm việc liên quan tới lý thuyết biểu diễn hay phải gõ các mã lệnh đại loại như:

```
$$\widehat{f}(\rho\uparrow_{_G}^{\phantom{H}})$$
```

để viết:

$$\widehat{f}(\rho \uparrow_{\sigma}^H)$$

Thật là tốt biết bao nếu ta có thể gõ lệnh ngắn và đơn giản hơn. Việc đó không khó nếu ta tự định nghĩa và đưa vào dùng một “macro” của riêng mình như sau:

```
\newcommand{\fhatrhoinG}{\widehat{f}(\rho\uparrow_{\scriptstyle G}^H)}
```

Bây giờ, mỗi khi trong chế độ toán học bạn gõ dòng lệnh:

```
\fhatrhoinG
```

- bạn sẽ có được kết quả hệt như ở trên. Giống như lệnh `\newtheorem`, dòng mã trên có thể đặt ở vị trí bất kỳ trong file đầu vào của bạn trước lần dùng đầu tiên, nhưng tiện nhất là đặt nó giữa hai dòng lệnh `\documentstyle` và `\begin{document}`.

Bạn cũng có thể định nghĩa các lệnh có thông số. Lấy luôn ví dụ vừa nêu, nếu H và G là các đại lượng có thể biến đổi, thì macro có thể được định nghĩa như:

```
\newcommand{\fhatrhoin}[2]{\widehat{f}(\rho\uparrow_{\scriptstyle \{#1\}}^{\{#2\}})}
```

Số [2] có nghĩa là có hai đối số, và chúng sẽ được chèn vào trong đoạn văn bản của macro mỗi khi xuất hiện `\{#1\}` và `\{#2\}`. Vậy, ví dụ nếu gõ lệnh:

```
$$\fhatrhoin{K}{\operatorname{Stab}(K)}$$
```

ta sẽ thu được

$$\widehat{f}(\rho \uparrow_K^{\operatorname{Stab}(K)})$$

Khi đang thực hiện thao tác trên, bạn cũng có thể định nghĩa:

```
\newcommand{\Stab}{\operatorname{Stab}}
```

nếu bạn có bộ ổn định trong văn bản đang soạn thảo của bạn.

VIII. 6. h. *Làm việc với các file lớn*

LaTeX rất thích hợp cho việc tạo ra các file văn bản lớn như luận văn, sách, v.v.. Ví dụ, lệnh `\ref{ }` làm nhẹ nhàng đi rất nhiều công việc cập nhật mã số các tài liệu tham khảo - một công việc mà bản thân tác giả cũng đã tiêu tốn rất nhiều thời gian. Tuy nhiên, ngay cả LaTeX cũng có thể khiến bạn bức dọc khi phải chờ đợi chương trình lần theo tài liệu hàng trăm, hàng ngàn trang của bạn, trước khi biểu đồ cần tìm lọt vào tầm nhìn của bạn. May sao, ta có cách để tránh sự chậm chạp đó.

Giả sử bạn phải viết luận văn. Trước hết, hãy chia luận văn của bạn thành từng phần nhỏ, ví dụ trong mỗi phần đó chứa các chương chẳng hạn. Giả sử sau khi chia, thay vì một file lớn, bạn có các file nhỏ hơn: `chapter1.tex`, `chapter2.tex`, v.v...

Sau đó, bạn tạo lại file chính, giả sử có tên là `thesis.tex`, theo cách sau:

```
\documentclass[12pt]{report}
\usepackage{amstex}
...
\newtheorem{thm}{Theorem}

\begin{document}
...
\end{document}
```

Hiển nhiên, bạn cần dùng lệnh `\documentclass` nào thích hợp, và đặt lệnh `\newtheorem` nào bạn dùng trước lệnh `\begin{document}`.

Bây giờ, khi bạn dùng LaTeX trên file `thesis.tex`, LaTeX sẽ đọc các chương khác nhau từ những file con mà bạn chọn. Giả sử bạn đang sửa chương 5 và không muốn động vào các chương khác. Bạn hãy đặt lệnh:

```
\includeonly{chapter5}
```

giữa lệnh `\documentclass` và lệnh `\begin{document}`, và bạn có thể bắt LaTeX xử lý lại riêng chương 5 mà bỏ qua các lệnh `\include` cho các chương khác. Tuy nhiên, LaTeX sẽ nhớ các lệnh `\label{ }` từ các chương khác, nên mọi lệnh `\ref` trong file `chapter5` đều được thực hiện tốt. Thêm nữa, LaTeX sẽ nhớ rằng lệnh `\chapter{ }`, thường xuất hiện ở đầu file, sẽ bắt đầu từ chương 5, chứ không phải từ chương 1.

VIII. 6. i. Kiểm tra và in

Bây giờ giả sử bạn đã chế bản xong tài liệu (`.tex`) và đang muốn xem lại thành quả lao động của mình. Điều đầu tiên bạn cần làm là phải tạo bản `.dvi` cho file `.tex`.

Lại giả sử tiếp bạn đã lưu giữ file văn bản của mình với tên `mythesis.tex` và đang gõ lệnh trở về C shell (có nghĩa là bạn đang ở trong hệ điều hành UNIX và có dấu nhắc hệ thống `"%"`.) Bạn có thể tạo file `.dvi` từ file `.tex` của bạn bằng cách gõ:

```
latex mythesis
```

Tiếp đó chương trình sẽ dịch các lệnh LaTeX2 trong file `mythesis.tex`.

Bây giờ, một trong hai khả năng sau sẽ xảy ra:

- Bạn không bị mắc lỗi nào. Lúc này LaTeX sẽ hỏi bạn muốn tạo ra bao nhiêu trang và tạo ra các file và `mythesis.log` trong số nhiều file khác. Ta chỉ quan tâm đến file `mythesis.dvi`, vì đó là kết quả mà ta mong đợi được xem hoặc in ra.
- Bạn mắc lỗi. Trong trường hợp này LaTeX sẽ thực hiện đến vị trí nào đó trong file `mythesis.tex` rồi dừng lại, thông báo cho ta một cách rõ ràng rằng ở đó có cái gì đó không hiểu được. Thông báo lỗi sẽ hiện ra tùy theo mức độ; lỗi hay gặp nhất thường là bỏ quên mất các dấu `"$"` hay ngoặc móc. LaTeX sẽ thông báo cho bạn dòng có thể có chứa lỗi, nên hợp lý hơn là bạn bắt đầu mò tìm lỗi từ vị trí đó hoặc gần đó.

Sau khi các lỗi đã được phát hiện, bạn có thể muốn quay trở lại file `mythesis.tex` để sửa. Hãy kết thúc tiến trình **latex** bằng cách gõ `""`. Sau khi đã sửa xong, bạn lưu giữ file `mythesis.tex` lại lần nữa, rồi lại dùng lệnh **latex** để tiếp tục.

Giả sử rằng bây giờ bạn đã có một file `mythesis.dvi` tốt, không hề bị dính lỗi và nóng lòng muốn có ngay kết quả. Đầu tiên, bạn hãy nên xem lại thành quả lao động của mình trên màn hình, để có các nhận xét, thậm chí để có các sửa đổi cần thiết, trước khi in ra. Các bước thao tác sẽ được giải thích dưới đây.

■ *Xem lại văn bản được soạn thảo*

Hệ điều hành của Sun (Solaris) - và một số bản UNIX khác, đặc biệt là các bản Linux phát triển rất nhanh và nhạy bén với thị trường tin học - có khả năng hiển thị các văn bản ở dạng `.dvi` mà không cần in ra giấy. Nên nhớ rằng ngoài yếu tố thời gian, chi phí cho một trang in trên máy in laser hiện nay vẫn chưa phải là nhỏ, nhất là đối với các tài liệu lớn. Khả năng đó được gọi là *previewing* - xem trước khi in.

Bộ xem trước khi in chuẩn có tên là **xdvi** - để chạy trong chế độ đồ họa - và có thể được gọi bằng lệnh, theo ví dụ trên:

```
xdvi mythesis
```

để xem trước file `mythesis.dvi`. Trang đầu của văn bản `mythesis` sẽ được hiện lên. Thao tác xem và di chuyển giữa các trang văn bản được thực hiện theo cách thông thường bằng bàn phím hay bằng chuột vi tính, tương tự như trong MS Office hay Adobe Reader. Sự tương tự này càng rõ nét hơn khi so sánh với Adobe Reader, vì ta cũng có thể phóng to hay thu nhỏ phần cần xem bằng cách nhấn nút chuột tại đó.

Các lệnh bàn phím được tổng kết trong trang **man** `xdvi`. Ta nhắc lại một cách sơ lược : **p** cho bạn di ngược trở về trước, phím Space bar (tạo khoảng trống) làm bạn tiến lên phía trước, còn **q** - để thoát khỏi **xdvi**. Bạn có thể gõ một chữ số trước khi thực hiện lệnh tiến lên (Space bar) hoặc lùi lại (**p**) để chỉ số trang cần di chuyển. Một lệnh rất có ích là **R** sẽ làm cho file `.dvi` được đọc lại. Vì thế bạn có thể vừa sửa đổi tài liệu trong một cửa sổ màn hình, vừa chạy LaTeX trong cửa sổ khác để

định dạng nó, sau đó chuyển đến cửa sổ thứ ba được mở đang chạy lệnh **xdvi** và gõ **R** để theo dõi các thay đổi đó.

- *In văn bản*

Thao tác in được thực hiện với lệnh **laser** ; trong ví dụ đã nêu ta thực hiện như sau:

```
laser mythesis
```

và **mythesis** sẽ được in ra trên máy in laser. Chú ý rằng lệnh **laser** chấp nhận ưu tiên. Ví dụ, nếu bạn muốn in ra năm trang đầu tài liệu của bạn ở máy in công cộng có tên **lw8** (tên máy in HP được giá sử là **laserwriter8**), bạn có thể gõ lệnh:

```
laser -Plw8 -page1-5 mythesis
```

VIII. 6. j. *Đưa các hình ảnh vào tài liệu*

- *D/fig*

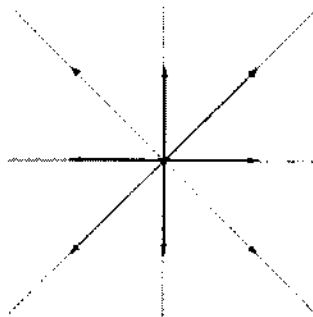
xfig là một chương trình dựa trên cơ sở X-Windows, được dùng để tạo ra các hình ảnh đủ loại, có thể đưa vào trong văn bản TeX hay in riêng. Trong tài liệu này ta không đi sâu vào các chi tiết tạo hình cụ thể, nếu cần, bạn đọc có thể tham khảo các thông tin trực tuyến đối với lệnh **fig**. Thay vào đó, ta cố gắng tìm hiểu cách biến kết quả ra của lệnh **fig** thành điều gì đó có lợi. Có ba khả năng cơ bản, với các ưu và nhược điểm của riêng từng trường hợp. Khả năng đầu tiên là tạo đầu ra tương thích với môi trường **picture** của LaTeX, khả năng thứ hai là đầu ra tương thích với môi trường **picture** mở rộng, còn gọi là môi trường **eepic**, và khả năng cuối là tạo file PostScript file để có thể in trực tiếp trên máy in laser hay đưa vào file của LaTeX bằng lệnh **\psfig**.

▪ *.lời tường picture*

LaTeX có môi trường thiết kế sẵn để mô tả hình ảnh sẽ được đưa vào file một cách tự động nhờ lệnh **fig**. Hãy tưởng tượng rằng bạn muốn đưa vào file văn bản một biểu đồ, ví dụ như cái gọi là *tọa độ gốc của $SO(5)$* . Đầu tiên hãy dùng **fig** để tạo ra biểu đồ và lưu giữ với tên `rootso5.fig`, nhớ rằng phải chọn các tùy chọn của LaTeX trên màn hình của **fig** sao cho **fig** không cho phép vẽ các đường mà LaTeX không thể tạo ra được.

Bạn sẽ phải chọn đúng định dạng file từ thực đơn *Export*, để tạo ra file có tên `rootso5.tex` với biểu đồ được mã hóa theo cách mà LaTeX có thể hiểu được. Do file trên không thể nạp lại được vào **xfig**, nên bạn cũng phải lưu cả một bản trong dạng cho **xfig**. Ở vị trí căn chèn hình ảnh trong file văn bản của bạn, hãy thêm các dòng lệnh sau:

```
\begin{figure}
\begin{center}
\end{center}
\caption{The Root System of  $SO(5)$ }
\end{figure}
```



H. VIII. 1. Ảnh hệ tọa độ gốc của $SO(5)$

Ta sẽ thu được hình ảnh được căn chính giữa trang và có tiêu đề, tại vị trí thích hợp trong tài liệu của bạn. Tuy công việc được thực hiện tự động và nhanh, nhưng vẫn tồn tại một vài vướng mắc. Vấn đề người ta gặp phải sớm nhất là chương trình

xfig tỏ ra khá buồng bỉnh khi chỉ in tốt các tài liệu với phông chữ Roman chuẩn. Nếu bạn muốn có được các tiêu đề hợp lý, ví dụ như với các chỉ số và số mũ kiểu a và β , thì việc đầu tiên phải làm là viết các ký tự tương ứng bằng phông chữ Roman trong hình vẽ nguyên thủy bằng **xfig**. Sau đó bạn phải chạy lệnh **fig2dev** để sửa đổi văn bản TeX và thay các chữ cái Hy Lạp cần thiết vào chỗ các chữ cái La tinh tương ứng.

Vấn đề nghiêm trọng hơn là bản thân LaTeX không được vũ trang để xử lý các hình vẽ cực kỳ phức tạp, ngoài các đường tròn có bán kính xác định và các đường thẳng, các vectơ với độ dốc nhất định. Để làm được các hình phức tạp hơn, ví dụ như các đường cong, bạn phải truy xuất từ **xfig** tới eepic hoặc PostScript.

■ *.lời tường eepic picture*

Giả sử bạn muốn đưa vào tài liệu của mình hình ảnh mặt Riemann surface loại 2. Rõ ràng với môi trường `picture` không thể thực hiện nổi mà ta cần tới phần mở rộng `eepic` cho LaTeX.

Đầu tiên ta phải sử dụng **xfig** để vẽ hình, trước hết là với công cụ Spline của nó. Và sẽ rất có ích nếu bạn *bật chế độ lưới* (*grid*) và *tắt chế độ nam châm* (*magnet*). Sau đó lưu giữ file với tên, ví dụ là `genus2.fig`. Việc còn lại là truy xuất file đó đến eepic.

Thao tác trên sẽ tạo ra file được gọi là `genus2.tex` chứa mô tả về đường gờ xoắn của bạn trong môi trường `eepic` mở rộng cho `picture`. Để đưa vào trong văn bản của bạn, bạn phải nạp môi trường hình ảnh mở rộng bằng cách đặt `epic` và *tiếp theo sau là eepic* trong lệnh `\usepackage` của bạn, ví dụ:

```
\usepackage{epic,eepic}
```

Sau đó dùng các lệnh giống hệt như ta đã dùng cho việc in ra hệ tọa độ gốc của $SO(5)$ như trên:

```
\begin{figure}
\begin{center}
\end{center}
\end{figure}
```

```
\caption{Riemann Surface of Genus 2}
\end{figure}
```



H. VIII. 2. Mặt Riemann loại 2

▪ *Đưa kèm các mã PostScript vào tài liệu soạn thảo*

Các gói phần mềm `psfig`/**TEX** and `psbox` có thể được dùng để đưa các đoạn PostScript thẳng vào văn bản. Chi tiết về các gói phần mềm này có thể tham khảo trong các trang **man** tương ứng, hoặc trong các tài liệu hướng dẫn được cập nhật thường xuyên về TeX/LaTeX.

Các chương trình xem trước khi in bình thường của TeX (**xdvi**, v.v..) sẽ không đưa các kết quả của lệnh `\psfig` vào trong các hình ảnh của chúng, thay vào đó trên màn hình ta sẽ chỉ thấy các chỗ trống. Để có thể xem được các file `.dvi` và các ảnh PostScript, bạn phải dùng **dvi2ps mythesis.dvi** để có được bản PostScript của file (gọi là **mythesis.ps**) có thể xem trực tiếp trên màn hình bằng lệnh **ghostview mythesis.ps**.

Để dùng `psfig`, phải thay đổi lệnh `\usepackage` của bạn nhằm đưa `psfig` vào; ví dụ:

```
\usepackage{epsfig}
```

Sau đó gõ:

```
\psfig{figure=filename.ps}
```

trong văn bản của bạn để chèn vào file PostScript có tên `filename.ps`. Toàn bộ dãy lệnh để chèn hình ảnh vào tài liệu như sau:

```

\begin{figure}
\centerline{\psfig{figure=eye.ps}}
\caption{Eye.}
\end{figure}

```



H. VIII. 3. Ảnh con mắt (Tiêu đề ảnh là "Eye.").

VIII. 6. k. Bộ đếm

Mọi số thứ tự trong LaTeX được dành cho bạn đều được liên hệ với các bộ đếm. Tên của bộ đếm trùng với tên của môi trường hay tên lệnh đã tạo ra số thứ tự, ngoại trừ với trường hợp không có dấu "\". Sau đây là danh sách của các bộ đếm được dùng trong kiểu dạng chuẩn tài liệu LaTeX (v. 2.09) để điều khiển việc đánh số thứ tự:

Part paragraph	Subsubsection	figure	enumi
Chapter	subparagraph	table	enumii
Section	page	footnote	enumiii
Subsection	equation	mpfootnote	enumiv

▪ Các cách điều khiển việc in ra của các bộ đếm

Bộ đếm được in trong LaTeX bằng cách dùng lệnh **\the** tiếp liền theo là tên của bộ đếm, Ví dụ, nếu muốn đánh số thứ tự các phương trình trong văn bản được in ra, ta viết lệnh **\theequation**, đánh số các phần tài liệu - bằng **\thesection**.

Nếu bạn muốn thay đổi kiểu các số thứ tự được in ra, ví dụ, từ kiểu chữ số Ả rập (1, 2, 3, ...) thành kiểu La mã (I, II, III, ...), bạn phải định nghĩa lại lệnh tương ứng bằng lệnh **\renewcommand** (xem lại phần *Các lệnh LaTeX*).

Ví dụ, để đánh số các phương trình trong phần phụ lục bằng các ký hiệu A-1, A-2, ... , ta dùng:

```
\renewcommand{\theequation}{A-\arabic{equation}}
```

Đối số thứ hai ngụ ý lệnh `\theequation` sẽ tạo các số được bắt đầu bằng chữ "A-" và tiếp theo là kết quả của lệnh `\arabic{equation}`, cho ta lần lượt các giá trị của bộ đếm cho phương trình được in với các chữ số Ả rập.

Dạng của tiêu đề các đề mục trong *List Environments* được cho bởi lệnh `\labelitemi` và `\labelitemii` (đối với hai mức đầu của *Itemize Environment* – môi trường theo đề mục), và `\labelenumi` (đối với *Enumerate Environment* – môi trường theo số thứ tự); các lệnh này có thể được định nghĩa lại.

Bạn có thể thay đổi cường bức các số được tạo bởi bất kỳ bộ đếm nào. Ví dụ, nếu bạn muốn đánh số trang chỉ cho trang thứ 13, bạn hãy dùng:

```
\setcounter{page}{13}
```

▪ Các lệnh để sử dụng bộ đếm

Các lệnh sử dụng bộ đếm:

```
\addtocounter
\newcounter
\setcounter
\usecounter
\value
\renewcommand
```

Các lệnh in các giá trị của bộ đếm trong các dạng khác nhau:

```
\arabic
\alph      \Alph
\roman     \Roman
\fnsymbol
```

IX. Sử dụng X-Windows

IX. 1. Khái niệm chung

Môi trường làm việc với giao diện GUI (*Graphic User Interface* - giao diện đồ họa người dùng) của UNIX giúp thực hiện các thao tác dễ dàng hơn. Môi trường X-Windows cung cấp cho người dùng một số công cụ với giao diện đồ họa để dùng với thiết bị chuột vì tính, ví dụ:

- **Administration tool**: quản lý người & nhóm người sử dụng, cấu hình máy in, quản lý các host name, ...
- **File manager**: quản lý hệ thống file và thư mục
- **Mail tool**: quản lý hệ thống E-mail của từng người sử dụng
- **Text editor**: trình soạn thảo văn bản
- **Backup tool**: tiện ích lưu trữ và khôi phục
- **Command console và Terminal**: cho phép thực hiện lệnh ở chế độ dòng lệnh.
- v.v..

X-Windows trải qua rất nhiều lần cải tiến từ năm 1984, bắt nguồn từ sản phẩm quy mô xí nghiệp của hãng DEC (Data Equipment Corporation) và dự án Athena của MIT (Viện Công nghệ Massachusetts, Mỹ), và đến nay mang tên X11. Phiên bản đầu tiên được dùng là X11R1 ra mắt từ 1987. Kể từ phiên bản thứ sáu, X11R6, việc phát triển và phát hành X Windows System do công ty X Consortium đảm nhiệm. Riêng cho các máy PC, một dự án bắt đầu năm 1992 và chính thức từ 1994 có tên The XFree86 Project đưa ra một cải tiến phát hành miễn phí của X-Windows là Xfree86TM. Sản phẩm này nhanh chóng có mặt trong nhiều bản Linux và hiện giờ cũng là thành phần miễn phí trong các bản UNIX thương mại khác. X-Windows hiện nay không đơn giản chỉ là giao diện đồ họa cho người dùng. Nó còn gồm các thư viện đồ họa nội tại (Xt - X Toolkit Intrinsics) và mở rộng (Xaw - Athene widget set), được gọi chung là X Toolkit, giúp cho người dùng có thể tự tạo riêng cho mình

giao diện đồ họa theo ý muốn. Các giao diện X-Windows hiện nay được dùng trong UNIX nói chung được phát triển rất đa dạng: CDE (Common Desktop Environment - Môi trường màn hình làm việc chung), KDE (K Desktop Environment), GNOME, các mô phỏng màn hình làm việc của Mac (Macintosh), của MS-Windows (3.x, 9x) trong Linux, OpenLook (Hewlett-Packard) v.v.. X-Windows được xây dựng theo mô hình Client/Server và được chia thành hai phần tương ứng Server và Client. Mô hình này có những ưu điểm sau:

- Có thể cài đặt riêng rẽ các phần trên các máy khác nhau.
- Có thể đăng nhập tại một thời điểm vào nhiều hơn một máy.
- Có thể mở rộng cửa sổ ra trên nhiều hơn một màn hình.
- Có thể gửi cửa sổ chứa ứng dụng tới nhiều hơn một màn hình.
- Có thể dễ dàng xuất ứng dụng của X client tới hệ thống khác.

Tương ứng, trong Solaris X-Windows mang tên **OpenWin**, như một phiên bản riêng của hãng Sun, bên cạnh CDE. Trong thực tế hoạt động, phần lớn sự chú ý của người dùng mới tập trung vào phần cài đặt trên máy chủ. Để chạy được trên máy trạm, cần cài đặt thêm phần mềm mô phỏng X-Terminal, ví dụ như eXceed 4,5,6, thậm chí các bản dạng shareware (phát triển tự do, trả tiền tùy tâm cho tác giả) có rất nhiều trên Internet. Người đọc có thể tham khảo và tải về các bản này, ví dụ từ các website như: <http://www.simtel.net/win9x>, <http://www.zdnet.com/download>, <http://www.cnet.com>, <http://www.tucows.com>, <ftp://ftp.download.com>, ...

IX. 2. Thiết lập X terminals trên các máy tính với CPU Intel

Solaris x86 hiện nay được hỗ trợ phần cứng rộng rãi cho các máy tính trên cơ sở Intel. Có thể tạo nên các X terminal trên các máy PC để đăng nhập và chạy các ứng dụng thương mại, vốn rất chuyên nghiệp và phong phú, được thiết kế để hoạt động trong hệ Solaris SPARC.

X terminal hiệu quả cao, dễ điều khiển có thể được thiết lập trên các máy PC từ loại rất rẻ tiền đến các mẫu máy hiện đại nhất. Để làm được việc đó, ta chỉ cần tối thiểu dung lượng đĩa cứng khoảng hơn 200 MB một chút, 32 MB RAM và CPU loại Pentium 90 MHz.

IX. 2. a. *X terminal*

Về mặt lịch sử, người dùng PC đã làm việc với các X server dưới môi trường Windows, cho phép đăng nhập từ xa ở chế độ đồ họa vào các ứng dụng X Windows chạy trên các server tính toán với hệ điều hành UNIX. Thông thường, các X server dưới môi trường Windows làm việc tốt, với điểm lợi là cho phép giao tiếp giữa Windows và UNIX. Nhưng cũng tồn tại không ít các điều bất tiện, trong số đó có thể kể đến các vấn đề so sánh màu sắc hay thiếu phần mở rộng để hiển thị phông chữ kiểu postscript cho các máy Sun. Quan trọng hơn, trong các tình huống không bắt buộc phải dùng môi trường Windows, chi phí cho việc mua và cài đặt, bảo trì môi trường kết hợp UNIX-Windows và cho việc đào tạo nhân viên hoàn toàn lãng phí và vô ích.

Một lựa chọn khác là mua những thiết bị đầu cuối với giao diện đồ họa cho người dùng (X terminal), hay còn gọi là các máy tính mạng chuyên dụng (network computer - NC). Đó là các *thin clients* kiểu như NCD's Explora họ 400 và 700, dùng nối với các X server có kiến trúc tính toán độc lập (*Independence Computing Architecture* - ICA) hỗ trợ cho hệ điều hành nhiều người dùng Windows NT với vai trò hệ điều hành tại chỗ, và máy tính ảo Java. Các server ứng dụng Windows dựa trên cơ sở ICA cho phép đăng nhập từ xa tới các chương trình Windows theo cách giống hệt như đối với X Windows.

Ta còn có thể dùng các bản miễn phí của hệ điều hành Linux trên PC để biến PC thành một X terminal, có khả năng giao tiếp với các X server. Tuy nhiên, cần để ý rằng tự thân Linux đã là một hệ điều hành UNIX, nhưng cũng có các tiện ích và khác biệt riêng, có thể không thật cần thiết đối với nhu cầu công việc cụ thể của đa số người dùng. Nếu chấp thuận giải pháp này, các cơ quan phải chi phí thêm để đào tạo bổ sung cho người dùng với hiệu quả không cao.

Lựa chọn cuối cùng là dùng PC như là X terminal bằng cách chạy Solaris x86 ở chế độ X terminal thực sự. Bằng cách này chúng ta có được X terminal Sun Solaris hoàn chỉnh với chức năng quản lý từ xa tuyệt vời, với các khả năng cấu hình liên quan tới môi trường Solaris, kể cả **rsh**, **telnet**, in từ xa nhờ daemon **lpd** và chế độ cài đặt JumpStart.

Chạy Solaris ở chế độ X terminal cho phép ta bớt được nhu cầu hỗ trợ nhiều hệ điều hành, các vấn đề về chạy file, tên máy, liên quan tới cấu hình Client/Server như NFS, NIS+.

IX. 2. b. Thiết lập terminal

Bước đầu tiên để thiết lập X terminal Solaris x86 là kiểm tra xem máy PC x86 và các linh kiện thành phần của nó có tương thích phần cứng với Solaris hay không, qua bản danh sách tương thích phần cứng do hãng Sun cung cấp và cập nhật thường kỳ. Danh sách này bao gồm các máy chủ, máy đơn nguyên chiếc, được cung cấp bởi các hãng khác nhau, các loại bo mạch chủ, các bộ điều khiển SCSI, các CPU, các loại card mạng, card đồ họa, màn hình, card âm thanh, các thiết bị lưu trữ trên băng từ, v.v.. được coi là thích hợp, dùng được trong Solaris. Nếu máy PC được cấu hình bởi các nhà cung cấp, thông báo xác nhận tính tương thích phần cứng với Sun Solaris x86 của họ sẽ giúp ta cài đặt lần đầu tiên một cách suôn sẻ hơn.

Có nhiều nguyên nhân khiến cho việc cài đặt Solaris x86 không thành công. Nguyên nhân thường thấy nhất là nơi chứa môi trường cài đặt không được xác định, do ổ đĩa EIDE CD-ROM không hoàn toàn tuân theo chuẩn ATAPI (một tiêu chuẩn phổ biến cho các đầu đọc đĩa CD-ROM hiện nay) hay ổ đĩa SCSI CD-ROM không có tên trong bản danh sách tương thích phần cứng của Solarisx86. Các trục trặc khác khi cài đặt Solaris x86 có thể xuất hiện do card điều khiển SCSI, do card mạng... Các lỗi không khởi động được giao diện đồ họa thường liên quan tới card đồ họa và màn hình. Khi gặp phải các khó khăn trên, ta cần phải xác định các thiết bị phần cứng có vấn đề phải xem xét, tìm kiếm các trình điều khiển mới nhất cho chúng, hoặc trong trường hợp bất khả kháng, thay chúng bằng các thiết bị tương thích Solaris x86 có tính năng tương đương hoặc tốt hơn. Các cập nhật trình điều khiển thiết bị thường xuyên được hãng Sun bổ sung và cung cấp trên các bộ đĩa DU (Driver Updates) cho khách hàng chính thức, hay có thể tải về từ Internet, ví dụ từ website <http://access1.sun.com/drivers> hay của các hãng thứ ba.

Nếu máy tính trên nền Intel được thiết kế chỉ để dùng trong chế độ X terminal, ta thậm chí không cần cài đặt ổ CD-ROM. Thay vào đó, ta có thể cài đặt Solaris x86 qua mạng từ máy chủ Solaris SPARC hay Solaris x86 bằng chương trình cài đặt sẵn JumpStart. Để giảm chi phí, ta cũng có thể chỉ lắp ổ đĩa CD-ROM hoặc di

trong thời gian cài đặt mà thôi. Một X terminal đã được hoàn tất chỉ đòi hỏi cấu hình gồm bàn phím, chuột, màn hình, vỏ máy, bo mạch chủ, CPU, bộ nhớ, đĩa cứng, các card mạng và đồ họa.

IX. 2. c. Để có chất lượng hình ảnh tốt

Đối với người dùng đã làm việc với các màn hình công nghệ cao Sony Trinitron đi kèm với các trạm làm việc Sun SPARC thì phải có một thời gian nào đó để làm quen với các màn hình 17 inches rẻ tiền, độ phân giải 1158x768 của các PC thông thường (chứ chưa nói đến các màn hình 14 inch rẻ tiền phổ biến hiện nay). Việc có ý nghĩa quan trọng hơn là chọn lựa card đồ họa thích hợp để máy có thể hoạt động tốt, không gây ra các trục trặc và hình ảnh không nét. Cần kiểm tra các ứng dụng cá trong môi trường OpenWindows lẫn trong CDE (với Linux - trong các môi trường đồ họa khác nhau: KDE, CDE, GNOME,...) để phát hiện những lỗi thường gặp như mép các ảnh không nét (các hình tròn trong chương trình **xfig** có thể là phương tiện kiểm tra tốt), tốc độ làm tươi màn hình không đúng, vết di chuyển của chuột, các mục của thực đơn không được cập nhật đúng, hay cuối cùng là các vật thể không được tạo dáng, đánh bóng hay tô mầu hoàn toàn.

Thường là việc thay thế bằng một card đồ họa khác sẽ khắc phục hầu hết - nếu không phải là tất cả - các lỗi trên. Lệnh **kdmconfig** (trong SuSE Linux là một trong các lệnh **Sax**, **iSax**, **Sax2**, **XF86Setup**, **XF86Config**), cho phép ta thay đổi các trình điều khiển thiết bị đồ họa mà không cần phải gỡ bỏ, cài đặt lại hoàn toàn các card đó. Do đó, việc kiểm tra và cấu hình các card khác nhau có thể được thực hiện nhanh chóng hơn rất nhiều so với các thao tác cài đặt - tháo - lắp - cài đặt lại cố điển vẫn được dùng. Cần phải chú ý rằng Solaris chỉ hỗ trợ các chế độ 8 bit mầu (256 mầu) và 24 bit mầu (16 triệu mầu), trong khi đó Linux hỗ trợ nhiều chế độ mầu với các độ sâu khác nhau, giống hệt như đối với Microsoft Windows.

IX. 2. d. Chuột vi tính trong chế độ đồ họa

Vấn đề thường gặp phải, khi ta muốn kết hợp sử dụng cả hệ điều hành UNIX và Windows, là do lịch sử để lại các hệ thống UNIX thường sử dụng loại chuột vi tính có ba phím, còn Microsoft Windows lại được thiết kế cho các loại chuột hai phím.

Phần mềm cho X server trên PC vì vậy hay mô phỏng phím giữa của chuột bằng việc bấm đồng thời cả hai phím chuột phải và chuột trái.

Các X server cho Solaris lại không làm như vậy, mặc dù trong một vài chương trình ứng dụng như `xterm` cũng có các mô phỏng phím giữa chuột. Và dù rằng thậm chí để có sự tương thích với các phiên bản Windows, hầu hết các chương trình UNIX hiện đại không dùng phím giữa của chuột, vẫn còn có các chương trình UNIX thực tế sẽ trở nên vô dụng nếu thiếu phím chuột đó. Vì vậy, hãy kiểm tra các ứng dụng định khai thác và nếu cảm thấy cần phím giữa, ta nên thay chuột vì tính hai phím bằng chuột ba phím. Và nhớ rằng, phải đảm bảo không một trình điều khiển nào trong Windows được sử dụng trong chế độ chuột ba phím đối với chuột đó.

IX. 2. e. Đăng nhập vào phiên làm việc X

X terminal cần có sự hỗ trợ của máy chủ để điều khiển tiến trình đăng nhập hệ thống và để đóng vai trò máy chủ phiên làm việc. Máy chủ điều khiển đăng nhập sẽ cung cấp bộ lựa chọn để hiển thị danh sách các máy chủ phiên làm việc mà người dùng tại X terminal muốn chỉ định để điều khiển phiên làm việc của mình. Nói cách khác, người dùng có thể đăng nhập một trong các hệ thống đã đăng ký từ trước trong chế độ đồ họa. Trong trường hợp đăng nhập hệ thống trực tiếp, nghĩa là bản thân người dùng chính là thành viên của máy chủ nguyên thủy, ví dụ máy PC của Nguyễn Văn A đã đăng ký trong mạng tại chỗ với máy chủ đơn vị AA của mình, và đăng nhập hệ thống AA, chứ không vào hệ thống BB của đơn vị bên cạnh, ..., phiên làm việc sẽ được khởi tạo ngay trên máy chủ AA, bỏ qua khâu lựa chọn kể trên. Nếu máy chủ là Solaris SPARC hay Solaris x86 có dùng lệnh **dtlogin** để điều khiển đăng nhập hệ thống tại chỗ, thì cấu hình mặc định đã cho phép hỗ trợ truy nhập của X terminal vào máy chủ thông qua Giao thức kiểm tra điều khiển hiển thị X (X Display Manager Control Protocol - XDMCP). đương nhiên, máy chủ phiên làm việc Solaris phải đã được cài đặt CDE. Tương tự như vậy, trong Linux hay các bản UNIX khác, X server phải được kích hoạt. Nếu tiến trình **dtlogin** không chạy, ta có thể dùng lệnh **dtlogin -e** trên máy chủ để khiến nó hoạt động ngay khi X terminal được khởi động. Lệnh này là cái tiến đối với lệnh **xdm** trong CDE của hãng Sun. File `/usr/dt/config/Xaccess` chứa điều khiển việc đăng nhập hệ thống của X terminal đối với các tiến trình **dtlogin**.

Nếu ứng dụng đòi hỏi sử dụng các phông chữ phi chuẩn, ta phải cài đặt chúng trên máy chủ phiên làm việc và thêm chức năng cho nó thành máy chủ phông chữ. Để bổ sung các phông chữ vào máy chủ phiên làm việc, ta cần sửa file cấu hình `/usr/openwin/lib/X11/fontserver.cfg` của máy chủ phông chữ `fs` và gửi tín hiệu `SIGUSR1` tới máy chủ phông chữ khiến nó phải đọc lại file cấu hình. Trong các bản Solaris hiện thời `fs` được khởi động tự động bởi daemon `inetd`.

Ta có thể thêm máy chủ phông chữ vào đường dẫn phông chữ của người dùng bằng nhiều phương pháp. Phương pháp đầu tiên là sửa lại file `/usr/dt/config/Xaccess/Xsetup` trên máy chủ phiên làm việc sử dụng tiến trình `dtlogin` sao cho có thể đặt được các đường dẫn phông chữ mặc định, đặt lệnh `xset` vào file `~/.xinitrc` của người dùng trong chế độ OpenWindows, hoặc trong file `~/.dtprofile` trong CDE. Lệnh thêm máy chủ phông chữ vào trong đường dẫn phông chữ của X terminal là `xset fp+tcp/session_hostname:7100`.

IX. 2. f. Cài đặt cho x86

Cài đặt hệ điều hành đáp ứng được các yêu cầu chuyên nghiệp không bao giờ là công việc đơn giản và dễ dàng, thậm chí trên các X terminal. Sau khi đã cài đặt xong bản Solaris x86 (từ bản 2.6 trở đi đã bao gồm cả OpenWindows và CDE, còn từ v. 2.5/2.5.1 trở về trước - phải cài đặt thêm CDE từ bộ đĩa CD cài đặt đi kèm) ở mức độ phù hợp cho người dùng cuối, ta phải thực hiện kiểm tra cấu hình. Thông thường, chỉ cần có các bộ phông chữ của CDE là đủ. Nếu trên máy Solaris x86 (dùng làm X terminal) tiến trình `dtlogin` đã chạy, ta phải tắt nó đi bằng lệnh `dtconfig -e`. Sau đó đăng nhập hệ thống với quyền `root` và thực hiện lệnh `/usr/openwin/bin/Xsun -once -query session_host` ở chế độ dòng lệnh. Ta sẽ phải nhận thấy X server `Xsun` bắt đầu chạy và xuất hiện từ máy chủ phiên làm việc. Bây giờ ta phải đăng nhập hệ thống với tên người dùng cuối từ X terminal để kiểm tra cấu hình xác định cho người dùng đó. Khi ta ra khỏi hệ thống, X server sẽ ngừng hoạt động và máy tính sẽ trở về chế độ dòng lệnh với dấu nhắc hệ thống trên màn hình.

Sau khi tiến trình `dtlogin` đã hoạt động tốt, ta có thể thay đổi nội dung dòng vào `xs` theo yêu cầu trong file cấu hình hệ thống `/etc/inittab`. Việc khởi động

X server trên Solaris x86 từ `/etc/inittab` trước khi chạy sẽ tránh được các xung đột các tín hiệu bàn phím được đưa vào giữa `ttymon` và `Xsun`. `ttymon` lúc đó sẽ không khởi động cho đến khi X server kết thúc, cho phép ta có thể tắt `Xsun` một cách suôn sẻ bằng cách gửi lệnh `kill -TERM` tới định danh tiến trình (PID) của `Xsun`. Chỉ từ thời điểm đó `ttymon` mới khởi động và thiết bị đầu cuối sẵn sàng cho việc đăng nhập vào hệ thống trong chế độ dòng lệnh bình thường. Ta cũng có thể khởi động `Xsun` từ shell script trong `/etc/rc3.d`, nếu trước đó đã loại bỏ dòng vào liên quan với `ttymon` trong file `/etc/inittab`. Tuy nhiên biện pháp này không tỏ ra linh hoạt lắm, và từ bản Solaris 2.6 trở đi xung đột giữa X server và `ttymon` đã được giải quyết xong. Lệnh `Xsun` hỗ trợ kết nối trực tiếp tới một máy chủ phiên làm việc cụ thể bằng tùy chọn `-query`, kết nối có lựa chọn (`-indirect`) với máy chủ đóng vai trò máy chủ đăng nhập, hay gửi thông báo tìm kiếm máy chủ đầu tiên bắt gặp nào để điều khiển phiên làm việc (`-broadcast`). Trong số các tùy chọn rất tiện lợi khác của `Xsun` còn cho phép chỉ chạy X server có một lần và X server sẽ tắt khi phiên làm việc kết thúc bằng cách dùng khóa `-once`. Từ phiên bản Solaris 2.6 trở đi có thể chạy không những chỉ trên máy chủ phiên làm việc, mà còn trên cả các máy tại chỗ, cho phép chọn các phiên làm việc từ xa và đưa các thủ tục đăng nhập rườm rà, phức tạp ra khỏi tầm mắt của những người dùng cuối. Có thể tìm hiểu chi tiết hơn về các lệnh `Xsun`, `dtconfig`, và `dtlogin` trong các trang **man**.

IX. 2. g. Cải thiện X terminal và chế độ đa khởi động

Rất khó cải tiến gì thêm các khả năng tuyệt vời của Solaris x86 trong chế độ X terminal thực thụ. Tuy nhiên, ta có thể tiết kiệm không gian đĩa cứng và tăng cường tính bảo mật bằng cách loại bớt các gói chương trình, dùng các daemon không cần thiết. Ví dụ, một X terminal bình thường không cần có khả năng trao đổi thư điện tử, vì thế, việc tắt và loại bỏ `sendmail` khi khởi động là một biện pháp an toàn.

Lệnh `pkginfo` sẽ liệt kê các gói chương trình, nhờ đó ta có thể xem xét để loại bỏ, ví dụ, gói `SUNWdoc` có chứa chương trình hỗ trợ `nroff` để định khuôn dạng các trang trợ giúp. Nhưng với khối lượng cài đặt chuẩn cho người dùng cuối trong phạm vi 200 MB dung lượng đĩa cứng, lại còn chứa cả OpenWindows lẫn CDE, 32

MB cho swap, tiết kiệm bằng biện pháp cắt giảm các gói chương trình không thực sự cần thiết là hoàn toàn hợp lý.

Một vài bản UNIX có khả năng hỗ trợ các ứng dụng Windows, như SuSE Linux 6.4 phát hành tháng 5 năm 2000 có **VMware** cho phép chạy hầu như tất cả các ứng dụng Windows 9x. Tuy nhiên, nếu không đòi hỏi phải truy nhập đồng thời các ứng dụng của cả UNIX lẫn Windows, cấu hình tốt nhất là cấu hình cho phép đa khởi động Windows và Solaris x86. Điều này cũng đúng cho cả trường hợp cài đặt Windows NT và Solaris x86 trên cùng một máy. Ta chỉ việc cài đặt Windows NT trước, trong tiến trình cài đặt, khi thực hiện lệnh **fdisk** phải để dành lại dung lượng đĩa cứng cho Solaris. Sau đó, cài đặt Solaris vào phần đĩa cứng đó. Thông thường ta không cần phải chú ý vì Solaris sẽ chọn phần đĩa cứng chưa được dùng đến sau khi hoàn thành lệnh **fdisk**. Điều khiển khởi động của Solaris, cũng như một chương trình đa khởi động phổ biến khác - System Commander - cho phép ta khởi động máy theo Windows NT hay Solaris. Số hệ điều hành cực đại có thể cài đặt theo cách này để có được chế độ đa khởi động là bốn.

Ta cũng có thể thiết lập máy chủ các ứng dụng Windows dựa trên cơ sở một trong số vô vàn sản phẩm có khả năng cho phép đăng nhập đa hệ điều hành từ xa các ứng dụng Windows trên máy chủ Windows. Một số sản phẩm đó là Microsoft's Hydra Windows Terminal Server, các chương trình của Citrix, Tektronix, NCD, Exodus NEnterprise Windows application server software, EZ-Win software của Tera Technologies, v.v.. Phải chú ý trong đó có nhiều sản phẩm được viết cho Windows 3.51 chứ không phải cho 4.0 hoặc bản mới hơn.

Trước khi quyết định sử dụng các sản phẩm trên, phải chạy thử chúng cẩn thận với các ứng dụng định khai thác để kiểm tra cả tính tương thích lẫn hiệu năng. Phần mềm Samba, một sản phẩm theo định ước GNU, được cài đặt thêm vào máy chủ Solaris - cũng như các máy UNIX khác - cho phép chia sẻ các file và máy in với máy chủ (và máy trạm) Windows thông qua các giao thức *Khối thông báo của máy chủ* (Server Message Block - SMB) đặc trưng của Windows, cho phép thực hiện các dịch vụ file và in ấn trên Windows. Samba sẽ được mô tả chi tiết về cách cấu hình và sử dụng trong một chương riêng của cuốn sách này. Về thực chất, giao thức này không khác biệt nhiều lắm so với một giao thức phổ biến khác - NetWare Core - hay được dùng với mạng LAN. Mới đây hãng Microsoft mở rộng giao thức SMB để

các hệ điều hành khác có thể hỗ trợ được và đổi tên thành Hệ thống file Internet chung (Common Internet File System - CIFS) với tham vọng thiết lập một chuẩn Internet chung.

IX. 2. h. Kết luận cho phần này

Các kiến thức được trình bày trong phần này có dựa theo bài báo của Richard Auletta (6/1998), người chuyên làm việc với đồ họa và với các máy Sun SPARC, nhằm cung cấp cho bạn đọc một cách nhìn toàn diện và đầy đủ hơn về vấn đề ít khi gặp phải trong thực tế sử dụng ở Việt Nam. Hầu hết mọi máy tính trên cơ sở bộ vi xử lý (CPU) Intel hiện đại đều có thể làm việc tốt như một X terminal khi chạy trên Solaris x86. Solaris x86 X terminal với phần cứng Intel là giải pháp tốt cho những người dùng cần có truy nhập rẻ tiền vào các máy Sun SPARC với các khả năng quản lý từ xa tuyệt vời. Những người dùng trong hệ thống của bạn thậm chí còn có thể hỗ trợ cho các ứng dụng Windows bằng đa khởi động (tức là khởi động có chọn lựa theo thực đơn - hoặc UNIX, hoặc một trong vài phiên bản Windows, nếu cần) hay sử dụng các sản phẩm phong phú trên thị trường cho phép đăng nhập từ xa vào các ứng dụng Windows.

IX. 3. Môi trường KDE

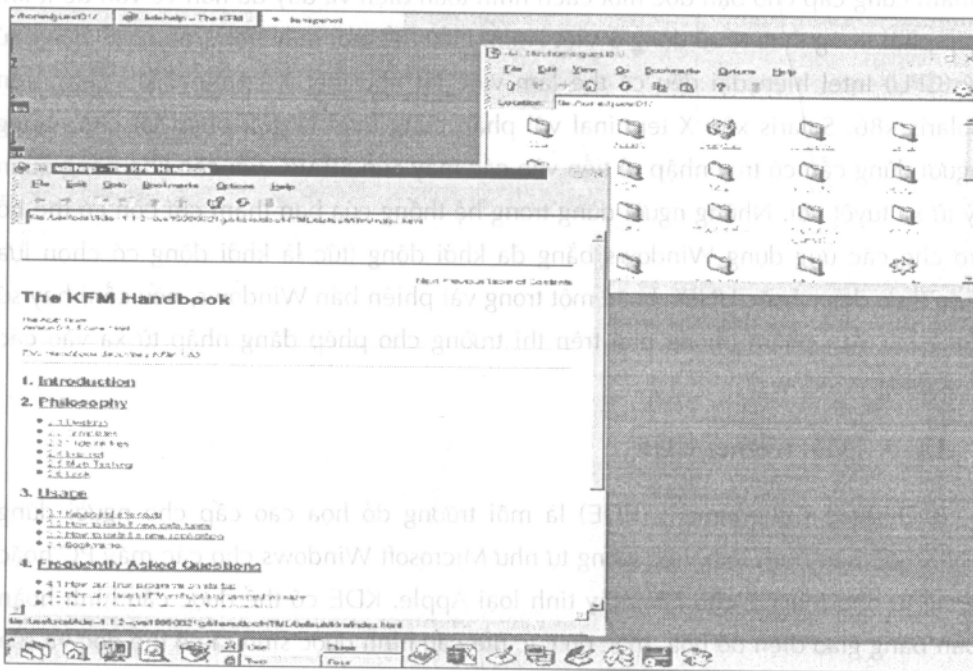
K Desktop Environment (KDE) là môi trường đồ họa cao cấp cho người dùng UNIX với màn hình làm việc tương tự như Microsoft Windows cho các máy PC hoặc tương tự như MacOS cho các máy tính loại Apple. KDE có thể được cấu hình hoàn toàn bằng giao diện đồ họa, mặc dù các file cấu hình được sinh ra có thể đọc và sửa đổi bằng các trình soạn thảo đơn giản. KDE trước đây rất phổ biến trong các bản Linux. Gần đây, nhiều hệ điều hành thương mại, nhất là Solaris, đã đưa KDE vào thành phần của mình. Với KDE, người dùng không có nhiều kinh nghiệm với chế độ dòng lệnh có thể sử dụng UNIX được dễ dàng hơn.

IX. 3. a. Các đặc điểm của KDE

KDE cung cấp nhiều đặc tính hiện đại như đăng nhập mạng một cách trong suốt, rất tiện lợi không những cho người dùng mới mà còn cho cả các chuyên gia. Thêm nữa, ta thấy có những người dùng Solaris ở nơi làm việc và Linux ở nhà; do KDE là

màn hình làm việc mặc định của nhiều bản Linux, những người đó có giao diện đồ họa nhất quán và quen thuộc ở mọi nơi.

KDE, hiện nay được biết tới như là một dự án mã nguồn mở lớn nhất, đã được bắt đầu bởi một nhóm nhỏ các nhà phát triển từ tháng 10 năm 1996. GNU Public License (GPL) phát hành KDE và đòi hỏi phải có thư viện QT từ hãng Troll Tech, có thể tải về - cả mã nguồn - một cách miễn phí theo điều kiện của QT Public License (QPL).



H. IX. 1. Màn hình làm việc chính của KDE đang hiển thị cảm nang.

Màn hình làm việc được trình bày trên H. IX. 1, có thể được phân thành các màn hình ảo và bao gồm ba thành phần chính:

- Thanh khởi động nhanh chứa nút **K** để bắt đầu chương trình, tương tự như nút *Start* của MS Windows. Sự xuất hiện của thanh khởi động và các đặc tính như tự động giấu thanh, là có thể cấu hình được bằng giao diện đồ họa của người dùng. Có tất cả bốn màn hình ảo. Ta có thể chuyển từ màn hình ảo

này sang màn hình ảo kia và ngược lại chỉ bằng thao tác bấm nút đơn giản trên thanh.

- *Thanh công cụ* cho thấy tất cả các ứng dụng đang chạy và cho phép bạn chuyển giữa các ứng dụng đó, thậm chí khi chúng đang được chạy trên các màn hình ảo khác nhau.
- Các biểu tượng cũng xuất hiện trên *màn hình làm việc chính*. Trong số các biểu tượng có cả biểu tượng của thùng rác (Trash) và khởi động tự động (Autostart). Đặc biệt, nhất là đối với các bản Linux, khi các lệnh **mount** và **share** mặc định chỉ có **root** mới có chủ quyền thực hiện, các biểu tượng CD-Rom và Floppy cho phép người dùng bình thường có thể **mount** các đĩa CD hay đĩa mềm tương ứng để sử dụng hệ thống file trong chúng.

KDE hỗ trợ quản lý theo phiên làm việc; do đó, khi bạn tắt KDE, tất cả các ứng dụng của KDE sẽ được khởi động lại và ở trạng thái gần với trạng thái trước khi tắt KDE. Ví dụ, trong trường hợp của chúng ta, văn file văn bản trước được nạp, nhưng bây giờ con trỏ ở đầu của file. Đặc tính này cần phải được hỗ trợ bởi một ứng dụng theo kiểu “kéo-rời-thả” được gọi là *ứng dụng K*.

IX. 3. b. Ứng dụng K và KDE 2.0

KDE có thể chạy mọi ứng dụng của X Windows. Bộ cài đặt KDE chuẩn gồm các công cụ được liệt kê tại bảng sau.

Bảng IX. 1. Các công cụ của bộ cài đặt KDE chuẩn

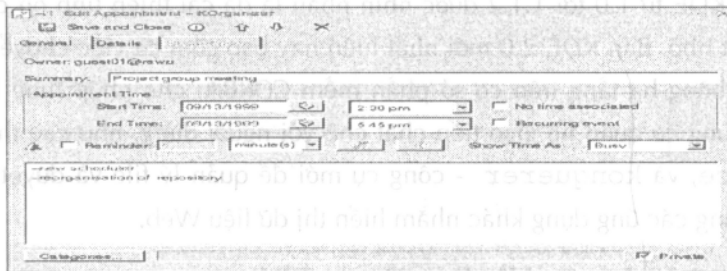
Công cụ	Mô tả
kfm	File manager giống như Microsoft Windows Explorer.
kmail	Chương trình thư điện tử đầy đủ chức năng.
korganizer	Công cụ xếp lịch tương tự như Microsoft Outlook . Định dạng file .vcs.
knews	Chương trình đọc tin tức.
kedit	Trình soạn thảo, tương tự MS Notepad.
kview	Chương trình hiển thị đồ họa.

document previewers for postscript	Xem trước khi in đối với tài liệu có các định dạng .dvi và fax.
kpaint	Chương trình vẽ đồ thị đơn giản.
network utilities	Các công cụ mạng, chi tiết trên H. XI. 4.
command line terminal	Cửa sổ để người dùng đưa lệnh giao tiếp với máy.

Ngoài các chương trình nằm trong thành phần của bộ cài đặt chuẩn KDE còn có rất nhiều các phần mềm ứng dụng K khác, hoặc đã hoàn chỉnh, đầy đủ các chức năng để trở thành phần mềm độc lập, như StarOffice, hiện nay là phiên bản 5.1a (tương tự như MS Office) mà ta sẽ đề cập đến ở ngay phần sau, hoặc đang trong quá trình phát triển. Ta có thể tham khảo thông tin về vài loại phần mềm trên trong bảng sau.

Bảng IX. 2. Vài ứng dụng K khác được hoàn thiện và đã được đưa vào sử dụng trong phiên bản KDE 2 (Xfree86 v. 4.x, từ giữa năm 2001).

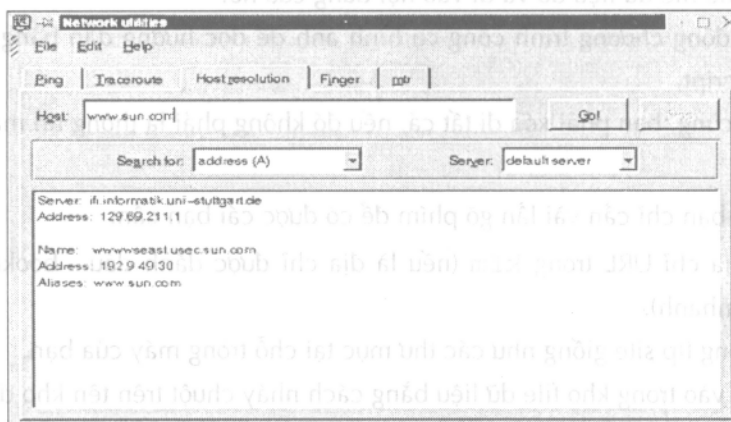
Công cụ	Mô tả
koffice	Chương trình văn phòng đầy đủ cho KDE, gồm bộ xử lý văn bản, chương trình bảng tính và chương trình trình bày.
kdevel	Môi trường phát triển tích hợp, hiện hỗ trợ các công cụ phát triển của GNU.
StarOffice	Ứng dụng chứa một phần hỗ trợ KDE, như giao thức kéo-và-thả. Hiện là phần mềm văn phòng rất được ưa chuộng trong UNIX. Hãng Sun đã đưa ra bộ StarOffice miễn phí từ phiên bản v.5.1a. Phiên bản StarOffice 6.2 được phát hành năm 2002. StarOffice có thể chạy được trong cả môi trường MS Windows và quan trọng hơn – có thể xử lý được các file của MS Office.



H. IX. 2. KDE Organizer đang soạn thảo giấy mời.



H. IX. 3. Có thể đọc được các file postscript bằng bộ đọc tài liệu.



H. IX. 4. Trình bày đồ họa của các công cụ mạng.

Các bản KDE từ 1.0 tới 1.1.2 được nhìn nhận là đã cải thiện tính ổn định và có các cập nhật nhỏ. Bản KDE 2.0 mới nhất hiện nay bao gồm cả việc chuyển tiếp cấu trúc truyền thông hạ tầng trên cơ sở phần mềm CORBA, cho phép thực hiện được nhiều ứng dụng có quan hệ giao tiếp chặt chẽ với người dùng, như các thành phần của *koffice*, và *konquerer* - công cụ mới để quản lý file và duyệt Web của KDE, có những các ứng dụng khác nhằm hiển thị dữ liệu Web.

IX. 3. c. Ví dụ một phiên làm việc của KDE

Giả sử rằng bạn đang tìm kiếm một công cụ để xem trước các hình ảnh khi in, và có ai đó mách bảo cho bạn rằng công cụ mà bạn cần tìm đó chính là **ghostview**. Bạn có thể tìm thấy chương trình được cung cấp miễn phí này theo địa chỉ: ftp://ftp.debian.de/debian/dists/stable/main/source/text/ghostview_1.5.orig.tar.gz.

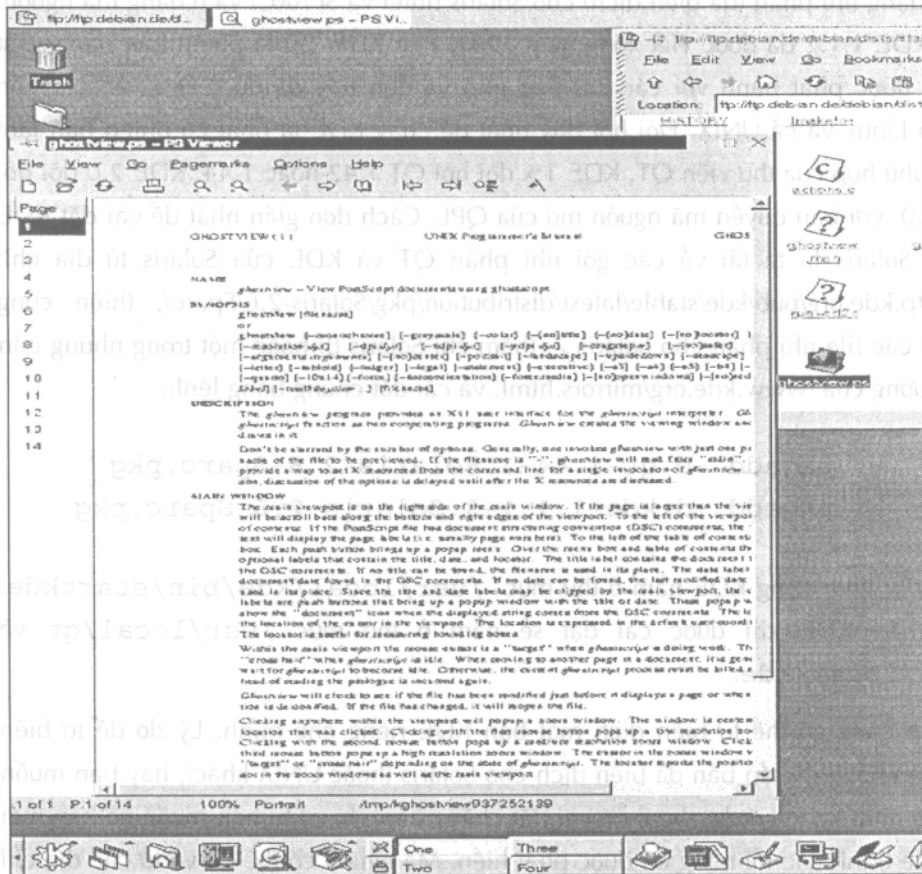
Theo lệ thường, bạn sẽ phải thực hiện các bước sau:

- **ftp** tới site [ftp.debian.de](ftp://ftp.debian.de), và đăng nhập hệ thống file tại đó như người dùng *anonymous*.
- Thực hiện lệnh **cd** qua các thư mục.
- Tải về kho file dữ liệu.
- Kiểm tra mục lục của kho file dữ liệu (văn bản) đã được tải về.
- Giải nén kho file dữ liệu, ví dụ, bằng cách dùng lệnh **tar**.
- Mở kho file dữ liệu đó và đi vào nội dung của nó.
- Khởi động chương trình công cụ hình ảnh để đọc hướng dẫn bằng kiểu chữ *postscript*.
- Cuối cùng, bạn phải xóa đi tất cả, nếu đó không phải là thông tin mà bạn tìm kiếm.

Với KDE, bạn chỉ cần vài lần gõ phím để có được cái bạn cần:

- Mở địa chỉ URL trong *kfm* (nếu là địa chỉ được đánh dấu - *bookmark*- thì càng nhanh).
- Đi trong ftp site giống như các thư mục tại chỗ trong máy của bạn.
- Nháy vào trong kho file dữ liệu bằng cách nháy chuột trên tên kho đó.
- Cuối cùng, nháy chuột trên file *ps*, bộ xem *postscript* trước khi in sẽ hiển thị file lên cho bạn.

Nếu đó không phải thông tin bạn muốn có, chỉ cần đóng bộ xem trước postscript. Còn nếu đó chính là thông tin bạn cần, bạn chỉ việc rê chuột kéo thả lên màn hình làm việc của mình là xong. Hình vẽ H. IX. 5 cho thấy màn hình làm việc với kfm và bộ xem postscript trước khi in để minh họa công việc vừa được mô tả ở trên. Chú ý rằng địa chỉ sau không có giá trị thực, và được viết một cách giả tưởng nhằm minh họa cho ví dụ:



H. IX. 5. Kiểm tra trang thông tin ghostview với bộ xem postscript trước.

ftp://ftp.debian.de/debian/dists/stable/main/source/text/ghostview_1.5.orig.tar.gz#tar:/ghostview-1.5.orig/ghostview.ps

trong kfm, chỉ cho bạn cách nạp file `ghostview.ps` nằm trong thư mục `con hostview-1.5.orig` trong thư mục `con ghostview` của kho dữ liệu `ghostview_1.5.orig.tar.gz`.

IX. 3. d. Cài đặt KDE

Cách đây không lâu, phiên bản KDE chính thức được dùng là 1.1.1, được đưa ra dưới dạng nhị phân (đã biên dịch) cho Solaris (Intel và SPARC) và ở dạng mã nguồn. Bản KDE 1.1.2 đã được viết xong năm 2001. Bản KDE 2.0 là phiên bản đang phát triển, được phát hành với các cải tiến nhỏ và đến nay đã được đưa vào các sản phẩm Linux và cả UNIX. Điều kiện duy nhất để chạy KDE là phải có phiên bản làm việc phù hợp của thư viện QT. KDE 1.x đòi hỏi QT 1.42 hoặc 1.44. KDE 2.0 đòi hỏi QT 2.0, với bản quyền mã nguồn mở của QPL. Cách đơn giản nhất để cài đặt KDE trong Solaris là tải về các gói nhị phân QT và KDE của Solaris từ địa chỉ: `ftp://ftp.kde.org/pub/kde/stable/latest/distribution/pkg/Solaris-2.6/Sparc/`, (hiện cũng đã có các file nhị phân cho Solaris 7, thậm chí Solaris 8) hoặc một trong những bản sao gương của `www.kde.org/mirrors.html`, và cài đặt chúng bằng lệnh:

```
pkgadd -d qt-1.44-1.Solaris-2.6-Sparc.pkg
pkgadd -d kde-1.1.1-3.Solaris-2.6-Sparc.pkg
```

Script khởi động KDE nằm ở thư mục `/usr/local/kde/bin/startkde`. Bản thân KDE khi được cài đặt sẽ nằm ở thư mục `/usr/local/qt` và `/usr/local/kde`.

Bạn cũng có thể tự biên dịch một bản KDE riêng cho mình. Lý do để tự biên dịch là không có sẵn bản đã biên dịch cho Solaris (hoặc UNIX khác), hay bạn muốn có cấu hình khác (ví dụ, thư mục cài đặt chẳng hạn), hay khi bạn muốn có bản mới nhất để tránh các lỗi nặng đã được phát hiện. Mã nguồn có thể tải về từ các địa chỉ `ftp://ftp.de.kde.org/pub/kde/stable/latest/distribution/tar/generic/source/` đối với KDE và `www.troll.no/` đối với thư viện QT. KDE 1.1.1 gồm `kdeadmin`, `kdebase`, `kdegames`, `kdegraphics`, `kdelibs`, `kdemultimedia`, `kdenetwork`, `kdesupport`, `kdetoys`, `kdeutils`, và `korganizer`. `kdesupport` và `kdelibs` cần thiết để chạy các ứng dụng K.

IX. 3. e. Biên dịch bản tiền phát hành KDE 1.1.2

Vì KDE liên tục được cải tiến nên mỗi khi cài đặt KDE phiên bản nào, bạn nên tham khảo thật kỹ các tài liệu hướng dẫn đi kèm theo các gói mã nguồn KDE mà bạn tải về được. Hiện nay, người ta dùng bộ biên dịch GNU C++ mới nhất (gcc, version 2.95.1). Do hạn chế của ngôn ngữ máy (hợp ngữ – assembler) trong file `/usr/ccs/bin/as` của Solaris 2.6 (xem <http://egcs.cygnum.com/ml/gcc-bugs/1999-04/msg00618.html>), ta cần phải dùng các tiện ích GNU **binutils** để biên dịch. Cũng cần phải có phần mềm GNU **tar** phiên bản 1.12 hoặc muộn hơn trên máy để thực hiện các thao tác giải nén. Thêm nữa, mọi người dùng đều phải để chương trình **tar** của GNU vào trong thư mục mà biến đường dẫn quy định, chứ chưa chắc đã nằm trong thư mục quy định cho **tar** của Sun là `/usr/bin`, hoặc sau khi giải nén xong mã nguồn từ kho dữ liệu `.tar.bz2` bằng lệnh:

```
bunzip2 -c file.tar.bz2 | tar xf
```

Bạn phải thay lệnh **tar** trong file `kdebase/kfm/kioslave/tar.cpp` với tên đường dẫn đầy đủ của nó (gỡ bỏ dòng có dấu "<" và thay bằng các dòng có dấu ">" tại các vị trí xxxcxxx) nhờ vào lệnh **diff**:

```
$ diff tar.cpp~ tar.cpp
77c77
< QString Command( "tar-%sOxf - \"" );
---
> QString Command( "/usr/local/bin/tar -%sOxf - \"
\" );
217c217
< QString Command( "tar -%stvf -");
---
> QString Command( "/usr/local/bin/tar -%stvf -");
376c376
< QString cmdTempl( "tar -%srf %s \"%s\"" );
---
> QString cmdTempl( "/usr/local/bin/tar
```

```
-%srf %s \"%s\" ) ;
```

Sau đó, ta vào từng thư mục con và thực hiện (dấu “\” tại vị trí tận cùng bên phải thêm vào mã nguồn để ký hiệu rằng dòng lệnh còn được tiếp tục viết):

```
CFLAGS="-O -DNeedVarargsPrototypes"
CXXFLAGS="-O -fpermissive" LDFLAGS="-s"
./configure --prefix=/usr/local/kde-1.1.2 --enable- \
    shared --with-quota --with-qt-dir=/usr/local/qt-1.44\
--with-extra-libs=/usr/openwin/lib --with-pam
make
make install
```

Chương trình kdesupport cần phải được biên dịch và cài đặt trước tiên, sau đó là kdelibs. File `resource.c` trong `kdebase/kdm` chỉ biên dịch được khi không có cờ `-DNeedVarargsPrototypes`. Trong bước tiếp theo bạn cần điều chỉnh các giá trị mặc định của các biến chung. Ví dụ, nếu bạn muốn dùng Emacs để khởi động theo các giá trị mặc định với kiểu chữ *fixed font*, bạn phải sửa:

```
/usr/local/kde-1.1.2/share/apps/kdisplay/app-defaults \
    /Emacs.ad.
```

IX. 3. f. Tích hợp KDE vào dtlogin của Solaris

Nếu bạn dùng `dtlogin` làm chương trình điều khiển đăng nhập hệ thống của bạn, nghĩa là bạn đăng nhập hệ thống và bước luôn vào chế độ đồ họa, bạn cần phải thực hiện ba bước sau để tích hợp KDE vào trong phần lựa chọn kiểu môi trường đồ họa trong Solaris (vốn mặc định chỉ bao gồm CDE và OpenWindows).

Bước thứ nhất là sửa file `Xresources.kde`, bước thứ hai là cài đặt `Xsession.kde` và `Xsession.kde2` vào thư mục có chứa các file nhị phân của KDE. Bước cuối cùng là chạy script `Install`. Sau khi khởi động lại, bạn sẽ thấy lựa chọn mới vào hệ thống bằng KDE đã nằm trong `dtlogin`. Các bản mã nguồn từ A đến D cho bạn thấy nội dung của các file trên.

Listing A: The Xresources.kde file

```

!!#####
!!
!!   Xresources.kde
!!
!!   Configuration file for the KDE alternate desktop
!!   resources
!!
!!#####
!!
!!   This file used by the Solaris Desktop Login
!!   manager
Dtlogin*altDtsIncrement:      True
Dtlogin*altDtName:           KDE Desktop
Dtlogin*altDtKey:             /usr/local/kde-1.1.2/bin/kwm
Dtlogin*altDtStart:           /usr/local/kde-1.1.2/dtlogin
                               /Xsession.kde
Dtlogin*altDtLogo:           KDElogo

```

Listing B: The Installscript for KDE

```

#!/bin/sh
#
#   These files are used to install KDE as an startup
#   option in the desktop #   window manager.
#
#   Copy Xresources.kde to /usr/dt/config/C/
#   Xresources.d/Xresources.kde
#
#   Copy KDElogo.bm and KDElogo.pm to /usr/dt/
#   appconfig/icons/C/
#
#   The file startkde belongs in the KDE binary

```

```

#   directory /   css/apps/kde/bin
#
echo "Copying Xresources file to /usr/dt/config/C/   \
      Xresources.d"
cp Xresources.kde /usr/dt/config/C/Xresources.d/   \
      Xresources.kde
#
echo "Copying KDElogo bitmaps to /usr/dt/           \
      appconfig/icons"
cp KDElogo.*m /usr/dt/appconfig/icons/C
#
# Edit to point to where your kde binaries are.
#
echo "Copying Xsession.kde and Xsession.kde2       \
      to /css/apps/kde/bin"
if [ ! -x /usr/local/kde-1.1.2/dtlogin ]; then
    mkdir /usr/local/kde-1.1.2/dtlogin
fi
cp Xsession.kde* /usr/local/kde-1.1.2/dtlogin
chmod a+x /usr/local/kde-1.1.2/dtlogin

      /Xsession.kde*

```

Listing C: The Xsession.kde file

```

#!/bin/ksh
# DTLogin variables
DTDSPMSG=/usr/dt/bin/dtdspmsg

if [ -z "$SESSIONTYPE" ]then
    export SESSIONTYPE="altDt"
fi

if [ -z "$DTSTARTIMS" ] then

```

```
    export DTSTARTIMS="False"
fi

if [ -z "$SDT_ALT_SESSION" ]
then
    export SDT_ALT_SESSION="/usr/local/kde-1.1.2/      \
        dtlogin/Xsession.kde2"
fi

if [ -z "$SDT_ALT_HELLO" ]
then
    if [ -x $DTDSPMSG ]; then
        export SDT_ALT_HELLO="/usr/dt/bin/          \
            dthello -string ' ` $DTDSPMSG -s 37      \
                /usr/dt/lib/nls/msg/$LANG/dthello.cat 1 \
                'Starting the K Desktop Environment'``' &"
    else
        export SDT_ALT_HELLO="/usr/dt/bin/dthello    \
            -string 'Starting the K Desktop Environment' &"
    fi
fi

export SDT_NO_DSDM=""

/usr/dt/bin/Xsession
```

Listing D: The Xsession.kde2 file.

```
#!/bin/ksh
# First a little namespace cleanup of vars associated
# with this
# (and Xsession.kde) scripts.

unset SDT_ALT_SESSION
```

```
unset SDT_ALT_HELLO
unset SDT_NO_DSDM

#KDE variables

KDEDIR=/usr/local/kde-1.1.2
export KDEDIR
# Paths of the necessary dynamiclibraries
LD_LIBRARY_PATH=/usr/local/lib:/usr/local/qt-1.44-2 \
    /lib:$KDEDIR/lib:/usr/local/gcc-2.95.1-binutils \
    /lib:/usr/dt/lib:$LD_LIBRARY_PATH
export LD_LIBRARY_PATH
PATH=$PATH:$KDEDIR/bin
export PATH
MANPATH=$MANPATH:$KDEDIR/man
export MANPATH
XAPPLRESDIR=$XAPPLRESDIR:$KDEDIR/lib/
    app-defaults
export XAPPLRESDIR
TERM=dtterm
export TERM
MAIL=/var/mail/$USER
export MAIL

# initialize the configuration first.

kcontrol -init

# Start the common desktop tools in the background.
# The sleeps reduce disk usage during startup.
# kaudioserver will put itself in the background
#     automatically
```

```
sleep 1 ; kaudioserver
(sleep 1 && exec kwmsound) &

# Add -ncols 96 after kfm if using a 8-bit display
(sleep 1 && exec kfm) &

(sleep 2 && exec krootwm) &
(sleep 1 && exec kpanel) &
(sleep 3 && exec kbgndwm) &

# finally, give the session control to the window
#      manager

sleep 2 ; kwm
```

IX. 3. g. Các vấn đề về bản quyền và GNOME

Việc phát hành các bản KDE nhị phân gặp phải khó khăn về mặt bản quyền. KDE được phát hành với các điều kiện về bản quyền của GPL. Mặc dù có thể tái QT về được và thư viện chạy là miễn phí, nhưng hãng Troll Tech quy định bản quyền cho QT-1.x, không cho phép các nhà phát triển thay đổi các bản QT đã được sao chép về. Thêm nữa, đối với Solaris, do QT không nằm trong thành phần Solaris, một cách hình thức, bản quyền GPL không cho phép phát hành các bản nhị phân cho Solaris. Trong thực tế, dự án KDE vẫn được thực hiện và việc phát hành mã nguồn KDE cho Solaris là hoàn toàn hợp lệ.

Trong năm 2001, Troll Tech đưa ra bản quyền bớt chặt chẽ hơn đối với QT 2.0. Do đó, KDE 2.0 được phát hành với mọi vấn đề về bản quyền đã được giải quyết.

Do có vấn đề bản quyền trên của KDE, dự án GNOME (www.gnome.org) được bắt đầu ngay sau dự án KDE. GNOME cũng đưa ra một môi trường đồ họa xây dựng trên thư viện GTK+ (www.gtk.org/), được phát hành dưới danh nghĩa thư viện GPL (LGPL), tương thích với GPL. GNOME hiện đã được phát hành trong các bản Linux.

Vào thời điểm này, cả hai dự án KDE và GNOME có tác dụng kích thích lẫn nhau và có vẻ như đang hợp tác với nhau ở mức cùng đưa ra các giao thức chung.

IX. 3. h. Các lỗi có thể gặp khi dùng KDE trong Solaris

Tuy Sun chưa hỗ trợ KDE cho tới các phiên bản 7 (x86), 2.7 (SPARC), nhưng KDE chạy khá ổn định trong Solaris. Có một vài yếu tố của KDE không được hỗ trợ trong Solaris, vì mục đích chính của KDE là được viết để các nhà phát triển chạy trong Linux. Ví dụ, việc hiển thị các thống kê bộ nhớ không làm việc trên các máy UltraSPARC trong bảng thông tin *K Control Center*. Trong khi cài đặt, KDE sẽ quét hệ thống để tìm các ứng dụng không phải của KDE, để tích hợp chúng vào *K-Panel*. Điều trở ngại thực sự là các tiện ích hệ thống của Solaris, như *audiotool* chẳng hạn, chưa được thử nghiệm.

Tiếp theo, đối với KDE 1.1.1, chế độ màn hình tiết kiệm năng lượng không làm việc tốt, và việc mở khóa màn hình không thực hiện được trong môi trường NIS+ với Solaris 2.6. Các lỗi trên đã được khắc phục trong bản tiền phát hành KDE 1.1.2.

IX. 3. i. Các thông tin tham khảo

Do dự án KDE rất lớn, được nhiều nhà phát triển tham gia, nên nếu gặp phải vấn đề gì vướng mắc, ta có thể tiết kiệm thời gian và công sức bằng cách tham khảo kho dữ liệu các trao đổi thư tín về KDE trên <http://lists.kde.org>. Các lỗi được phát hiện và mô tả quá trình khắc phục chúng được thuật lại ở <http://bugs.kde.org>. Nếu muốn có cơ sở mã nguồn mới nhất, bạn nên tham khảo các hướng dẫn về CVS tại www.kde.org/cvsup.html. Để biên dịch các công cụ từ CVS, bạn sẽ cần đến các chương trình GNU **autoconf** và GNU **automake**. Chúng có thể được tải về từ site <ftp://ftp.uni-stuttgart.de/pub/gnu/>.

IX. 4. StarOffice

Từ tháng 11 năm 1997, Sun Microsystems đưa StarOffice của StarDivision (một chi nhánh của Sun Microsystems tại Đức) vào trong mỗi một bộ Ultra Workstation được bán ra. StarOffice 4.0 bao gồm các thành phần:

- bộ xử lý văn bản
- bảng tính

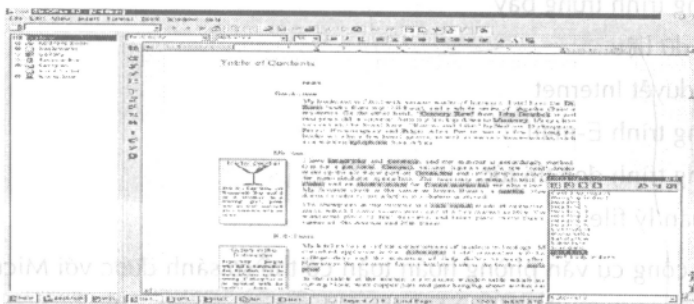
- chương trình trưng bày
- cơ sở dữ liệu
- trình duyệt Internet
- chương trình E-mail
- chương trình đọc tin tức
- bộ quản lý file

và là một bộ công cụ văn phòng hoàn toàn có thể so sánh được với Microsoft Office. Có thể thấy sự tương tự không chỉ theo chức năng, mà còn theo cả cách trình bày trên màn hình (các thực đơn và biểu tượng). Và thêm nữa, trong StarOffice, chức năng tích hợp mạng là tuyệt hảo.

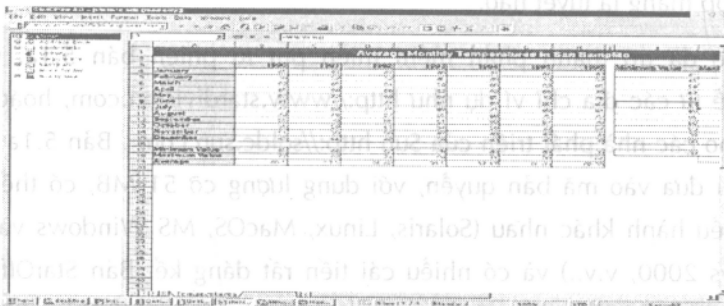
StarOffice đã trở thành phần mềm miễn phí từ phiên bản 5.1a. Có thể tải StarOffice về từ các địa chỉ ví dụ như <http://www.stardivision.com>, hoặc vào trong site dành cho các nhà phát triển của Sun <http://sldc.sun.com>. Bản 5.1a không còn đòi hỏi phải đưa vào mã bán quyền, với dung lượng cỡ 51 MB, có thể chạy trên nhiều hệ điều hành khác nhau (Solaris, Linux, MacOS, MS Windows và Windows NT/Windows 2000, v.v..) và có nhiều cải tiến rất đáng kể. Bản StarOffice 5.2 đã xuất hiện khá lâu ở Việt Nam, có tính năng tương đương như MS Office 97 về mặt xử lý văn bản và tốt hơn về mặt các ứng dụng Web (trình duyệt, thư điện tử, ...). Vào thời điểm đầu năm 2002, người ta hay nhắc tới bản StarOffice 6.x.

IX. 4. a. Linh hoạt như con tắc kè hoa (Chameleon) đổi màu

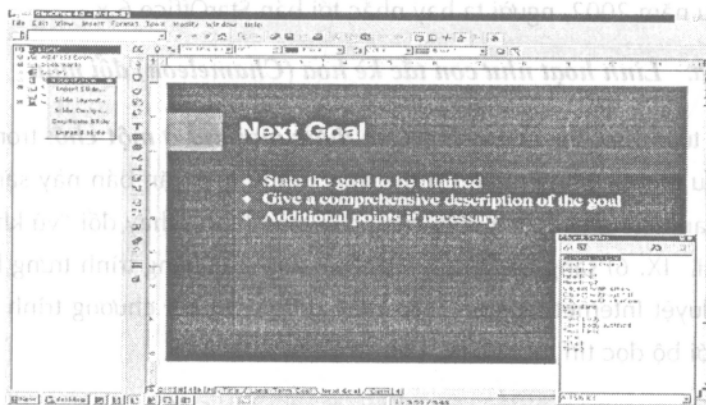
StarOffice tuân theo phương châm ***Làm được mọi thứ ở một chỗ***: trong một ứng dụng, mọi yếu tố đều thống nhất. Có thể thay đổi từ một văn bản này sang văn bản khác bằng thanh công cụ (dòng dưới cùng của màn hình), thay đổi “vũ khí” từ bộ xử lý văn bản (H. IX. 6) sang bảng tính (H. IX. 7), sang chương trình trưng bày (H. IX. 8), tới trình duyệt Internet và soạn thảo Web (H. IX. 9), tới chương trình E-mail (H. IX. 10), hay tới bộ đọc tin tức (H. IX. 11).



H. IX. 6. StarOffice sử dụng giao diện thống nhất.



H. IX. 7. Dạng bảng tính trong StarOffice.



H. IX. 8. Chương trình trưng bày trong StarOffice.

H. IX. 9. Sử dụng bộ trình duyệt và soạn thảo Web của StarOffice.



H. IX. 10. Chương trình E-mail của StarOffice.



H. IX. 11. Bộ đọc tin tức NNTP của StarOffice.

IX. 4. b. Các lý do nên dùng StarOffice

StarOffice chạy được trên nhiều hệ điều hành phổ biến hiện nay, trong đó có Solaris, Windows 9x/NT/2000/XP, OS/2, Linux, và MacOS, do đó nó là giải pháp tốt cho môi trường phần cứng không đồng nhất. Trên tất cả các hệ điều hành StarOffice đều có các chức năng giống nhau nên việc trao đổi văn bản giữa các máy tính chạy trên các hệ điều hành khác nhau được thực hiện một cách dễ dàng, không đòi hỏi các thao tác chuyển đổi gì. Đây là một ưu thế lớn, vì không những tiết kiệm được đầu tư, mà còn giúp tăng năng suất đáng kể cho các nhân viên, nhất là cho những người dùng Windows 95/MS Office có kinh nghiệm, ví dụ như thư ký văn phòng, nay chuyển ngay sang các trạm làm việc UNIX mà chưa được đào tạo các kiến thức cơ bản của UNIX.

Việc tích hợp Internet trong StarOffice được thực hiện một cách tuyệt vời. Bạn có thể gửi văn bản bằng E-mail cho đồng nghiệp, thậm chí đang làm việc trên máy có hệ điều hành khác, rất nhanh chóng. Tất cả các ứng dụng đều được hỗ trợ để truy xuất ra dạng HTML. Bạn cũng có thể lưu giữ các file một cách trong suốt từ các site FTP ở xa hay các server Web.

Bản StarOffice 5.1a có thể coi như tương đương với MS Office 97 về khả năng xử lý các loại văn bản. Có thể dùng StarOffice để đọc và viết các tài liệu theo định dạng của MS Office.

IX. 4. c. Cài đặt StarOffice

Phần này mô tả quá trình cài đặt cho StarOffice 4.0 với Service Pack 2 (SP 2) và một số khuyến nghị về SP 3. (Số của Service Pack được giải nghĩa như các lần phát hành tiếp theo, nhưng các cải tiến chưa khác về chất lớn đến nỗi sản phẩm được coi là phiên bản sau). Từ phiên bản StarOffice 5.1a trở đi, việc cài đặt dễ dàng hơn vì nhiều thao tác được thực hiện tự động.

Ta coi như có hai quy trình cài đặt: cho người dùng đơn lẻ và cài đặt mạng (từ SP 2). Quy trình cài đặt cho người dùng đơn lẻ cho phép từng người dùng cài đặt hệ thống đầy đủ trong thư mục home của mình, không cần đến quyền của root. Quy trình cài đặt mạng cài đặt StarOffice trong một thư mục chung, và mỗi người dùng có thể thay đổi các file trong thư mục home của mình. Vì quy trình thứ hai này có

nhiều điều cần quan tâm, nhất là rất thích hợp trong bối cảnh sử dụng mạng UNIX nói chung, ta sẽ xem xét quy trình này kỹ hơn.

Đĩa CD-ROM cài đặt của Solaris chứa các thư mục cho các ngôn ngữ khác nhau (SP 2 cho phép có các hiển thị bằng tiếng Anh, Đức, Pháp, Hà Lan, Nga và Thụy Điển). Các thư mục này chỉ khác nhau ở ngôn ngữ trình bày thực đơn, trợ giúp trực tuyến và các từ điển cho kiểm tra phát âm. Ngôn ngữ trong thực đơn bị biên dịch cứng trong mã nhị phân và tất nhiên là bằng tiếng Anh. Tất cả các thư mục chứa các từ điển tiếng Anh và tiếng bản địa tương ứng.

Sau khi đã **mount** đĩa CD-ROM (giả sử vào thư mục /cdrom và giả sử ta dùng tiếng Anh), gõ lệnh sau để bắt đầu quá trình cài đặt:

```
/cdrom/english/prod_sols/setup /net
```

Câu lệnh trên gọi tiện ích bằng đồ họa để cài đặt và tháo dỡ. Nếu người quản trị muốn cài đặt từ điển tiếng Anh (thêm vào thực đơn từ điển các thứ tiếng) thì phải chọn phương án cài đặt *Custom Installation*, nếu không - chỉ cần dùng *Standard Installation* là đủ. Sau khi chỉ định thư mục cài đặt, thường trên phần đĩa có dung lượng còn trống không dưới 120MB (giả sử là /opt/Office40), quá trình cài đặt bắt đầu lần lượt theo từng bước.

Sau khi rời khỏi tiện ích cài đặt, người quản trị hệ thống cần làm cho hàng chờ in được StarOffice nhận biết. Ta sẽ thực hiện điều đó qua việc sửa đổi file /opt/Office40/xp3/Xpdefaults. Để thêm vào hàng chờ in Postscript (ở đây hp3 ký hiệu là in trên máy in HP III Postscript) và thiết bị xem trước khi in, ta phải thêm các dòng sau vào các phần [devices] và [ports]:

```
[devices]
```

```
HP LaserJet III PostScript=HPIII522  
PostScript, hp5a
```

```
PreViewDevice=GENERIC PostScript,  
PreViewPort
```

```
[ports]
```

```
PreViewPort=/usr/bin/X11/gv - &
```

```
hp5a=lp -d hp5a
```

Trong ví dụ trên, các trang *HP LaserJet III PostScript* và *PreViewDevice* sẽ xuất hiện trên hộp thoại in của StarOffice. *Hp5a* và *PreViewPort* là các tên tượng trưng liên kết với lệnh shell tới dòng vào của hộp thoại in. *HPIII522 PostScript* và *GENERIC PostScript* là các định nghĩa máy in. Các định nghĩa tương ứng có thể tìm thấy trong nội dung của các file `/opt/Office40/xp3/ppds/GENERIC.ps` và `/opt/Office40/xp3/ppds/HPIII552.ps`. Danh sách đầy đủ các máy in hiện có được liệt kê trong phần `[other-devices]`. Ta có thể điều chỉnh được các lề trái, phải, trên, dưới, kích thước trang giấy, tỷ lệ và số bản in trong phần `[GENERIC,PostScript,PreViewPort]` và trong phần `[HPII I552 PostScript, hp5a]`. Máy in mặc định trong phần section có cùng định dạng như đối với các máy in lựa chọn khác ở trong phần `devices`. Trong Service Pack 3, việc cấu hình các máy in còn có thể được thực hiện một cách khác thông qua tiện ích thiết lập máy in bằng đồ họa `/opt/Office40/bin/psetup`.

Sau những bước trên, mỗi người dùng phải cá nhân hóa StarOffice cho riêng mình, thông qua chương trình có tên `/opt/Office40/bin/setup`. Ở bước này, có một điểm quan trọng cần chú ý: các người dùng trong mạng khi chạy chương trình trên **không được chọn** *Standard Installation* hoặc *Custom Installation*, mà **phải chọn** *Installation from Net or CD* ! Nếu không, toàn bộ phần mềm sẽ được sao về thư mục `home` của người dùng, chứ không phải chỉ các file có thể sửa đổi. Hãy tưởng tượng tình huống, nếu mỗi người dùng bình thường trong mạng, chưa kể tới các tài nguyên khác, còn chiếm 120 MB dung lượng đĩa cứng cho bản StarOffice không được cá nhân hóa. Chưa nói đến việc sử dụng StarOffice, liệu khả năng phần cứng có đáp ứng nổi yêu cầu về tài nguyên máy chủ nếu trong mạng có một số lượng lớn những người dùng như vậy. Sau khi cài đặt StarOffice trong mạng, mỗi người dùng phải đưa vào các dữ liệu cá nhân của họ, ví dụ như họ và tên, cơ quan, số điện thoại, fax, địa chỉ E-mail, là những thông tin mà chương trình *auto-pilot* sẽ tự động đặt vào tài liệu được tạo ra. Các dữ liệu có thể được thay đổi hoặc hoàn chỉnh

trong StarOffice bằng dãy lệnh thực đơn: **Tools->Options->General->User Data**). Mỗi người dùng đòi hỏi một tài nguyên khoảng 10MB dữ liệu trong thư mục home của mình (và chỉ khoảng 2MB trong SP3).

Sau công việc cá nhân hóa, StarOffice có thể được người dùng gọi ra chạy bằng lệnh **soffice**, nếu giá trị **/opt/Office40/bin** đã được thêm vào cho biến môi trường **PATH**. Ngoài ra, còn phải chú ý đến một đặc điểm: đa số người dùng đều muốn thực hiện công việc đầu tiên là thay đổi thư mục làm việc về thư mục home của mình. Điều đó được thực hiện trong mục **Tools->Options->General->Path->Working Directory**.

X. Lưu trữ và khôi phục dữ liệu

X. 1. Khái niệm và chiến lược lưu trữ dữ liệu

Các file của những hệ thống thông tin ngày càng lớn, sự cần thiết và mức độ quan trọng của các file này vô cùng lớn, đặc biệt là đối với những hệ thống thông tin của các tổ chức như ngân hàng, hàng không, dầu khí, cơ quan chính phủ, v.v..

Dữ liệu của các hệ thống thông tin xử lý nghiệp vụ tức thời đòi hỏi khi có bất cứ sự cố nào xuất hiện làm hỏng dữ liệu của hệ thống, thì ngay sau đó dữ liệu phải được khôi phục lại hoàn toàn.

Thông thường, những hệ thống thông tin, dù lớn hay nhỏ, đều xây dựng cho mình một hệ thống lưu trữ dữ liệu, sao cho khi có sự cố xảy ra (hỏng đĩa cứng, hỏng máy, ..) thì vẫn còn có dữ liệu đã được lưu trữ để phục hồi, không làm ảnh hưởng đến sự hoạt động của hệ thống.

Tùy theo các nhà cung cấp phần mềm và phần cứng khác nhau mà hệ thống lưu trữ có những chức năng và tiện ích khác nhau, nhưng đều có chung một mục đích là lưu lại dữ liệu. Ví dụ: có hệ thống cho phép lưu tức thời các dữ liệu đang hoạt động, người dùng không cần thiết phải ngừng hoạt động của mình; có hệ thống lưu cùng lúc trên nhiều băng từ khác nhau cho nhiều máy khác nhau; v.v..

Để lưu trữ được dữ liệu ta cần phải có ổ đọc thiết bị lưu trữ, phương tiện lưu trữ (băng từ, đĩa quang, ...) và phần mềm tương ứng để điều khiển tiến trình lưu trữ.

Các mức lưu trữ được quy định như sau:

- **0:** còn gọi là *master backup*, hay *full backup*, là mức lưu trữ đầy đủ. Đầy đủ đối với người quản trị nghĩa là toàn bộ các hệ thống file trong hệ thống UNIX (kể cả cấu trúc hệ thống file); còn đối với người dùng bình thường là toàn bộ các thư mục, các file của riêng họ, đều được lưu trữ. Thường đòi hỏi dung lượng rất lớn của đĩa, băng từ, ... Thời gian dành cho lưu trữ ở mức 0 cũng rất đáng kể, nhất là đối với các hệ thống chỉ dùng các thiết bị đơn như ổ băng

từ, v.v.. Các hệ thống UNIX lớn đều dùng các thư viện băng từ, cho phép vừa nâng cao dung lượng có ích, vừa có thể tiến hành lưu trữ song song các hệ thống file.

- **1 - 9:** các mức lưu trữ gia tăng, *incremental*, lưu trữ lại tất cả các file bị thay đổi kể từ lần lưu trữ với mức thấp hơn, do đó, cần ít dung lượng hơn. Theo định nghĩa, mức 1 lưu trữ tất cả những thay đổi kể từ lần lưu trữ trước ở mức 0, mức 2 - đối với lần lưu trữ trước ở mức 1, v.v.. Mức lưu trữ 9 thường là giá trị mặc định cho nhiều lệnh và tiện ích lưu trữ.

Đôi khi trong tài liệu kỹ thuật người ta còn gặp thêm một chế độ lưu trữ nữa được gọi là *differential*, tức là chỉ lưu trữ những dữ liệu có khác biệt so với trong lần lưu trữ trước đó, mà không chú ý tới các mức lưu trữ kể trên. Chế độ lưu trữ này chỉ được áp dụng nhiều trong các tiện ích lưu trữ chạy trên nền Microsoft Windows NT/2000. Trong UNIX, ta thấy quy định các mức lưu trữ 0-9 là đầy đủ và đáp ứng được các đòi hỏi an toàn dữ liệu.

Một trong những nhiệm vụ của người quản trị mạng là tổ chức và lưu trữ dữ liệu. Tùy theo quy mô mạng và dữ liệu, theo yêu cầu chức năng của cơ quan mà người quản trị phải lên lịch lưu trữ cho hợp lý. Một trong những sơ đồ đơn giản nhất được nêu lên sau đây:

Ngày trong tuần	Mức lưu trữ
Thứ hai	0 (đầy đủ)
Thứ ba	1 (gia tăng)
Thứ tư	1 (gia tăng)
Thứ năm	1 (gia tăng)
Thứ sáu	1 (gia tăng)

Lịch này có thuận lợi là chỉ cần bảo quản hai bộ băng từ - một cho mức 0 (tối thiểu là 2 băng) và một cho mức 1 (tối thiểu là 8 băng). Điểm bất lợi là thời gian lưu trữ lớn.

Một trong những lịch lưu trữ được áp dụng rộng rãi cho các hệ thống hoạt động rất tích cực, cập nhật dữ liệu thường xuyên trong các tổ chức như ngân hàng, có nội dung như sau:

Ngày trong tuần	Mức lưu trữ
Thứ hai đầu tiên trong tháng	0 (đầy đủ)
Các thứ hai còn lại	1 (gia tăng hàng tuần so với mức 0)
Thứ ba	2 (gia tăng hàng ngày so với mức 1)
Thứ tư	2 (gia tăng hàng ngày so với mức 1)
Thứ năm	2 (gia tăng hàng ngày so với mức 1)
Thứ sáu	2 (gia tăng hàng ngày so với mức 1)

Lịch thứ hai này đòi hỏi phải có ba bộ băng từ: bộ đầu tiên cho mức 0 (tối thiểu là 1 băng), bộ thứ hai cho mức 1 (tối thiểu là 3 băng) và cuối cùng là cho mức 2 (tối thiểu là 12 băng).

Ngoài ra, người quản trị còn phải thực hiện một nhiệm vụ quan trọng là xác định nội dung cần lưu trữ. Ở đây cần phân thành hai loại mục đích lưu trữ. Người quản trị hệ thống chỉ chú ý đảm bảo cho hệ thống làm việc an toàn, do vậy, thư mục khởi động, các file cấu hình, v.v.. được cất giữ trước tiên. Loại thứ hai là các dữ liệu liên quan tới một hay một nhóm ứng dụng chạy trên mạng UNIX, thường được quan tâm bảo vệ hơn. Nếu không vì lý do đặc biệt nào, các thư mục kiểu như /tmp, lost+found, các file .log và wtmp, core, v.v.. không nằm trong phạm vi dữ liệu cần cất giữ. Chặt chẽ hơn, chỉ có các cơ sở dữ liệu và file liên quan tới một ứng dụng cụ thể mới được lưu trữ.

Nói chung, việc lưu trữ và khôi phục dữ liệu cần phải được tiến hành ở mức chạy 1 hay s (S). Trong thực tế, nhiều khi yêu cầu này không được tuân thủ chặt chẽ, nhưng ít nhất người quản trị phải báo cho các người dùng khác ra khỏi hệ thống hay không được mở các file cần lưu trữ.

X. 2. Các lệnh UNIX thường dùng ở chế độ dòng lệnh

X. 2. a. Đối với hệ điều hành Solaris

Dưới đây là một số lệnh thường dùng ở chế độ dòng lệnh để lưu trữ dữ liệu. Các lệnh này đã có sẵn trên hệ điều hành, chúng cung cấp một số tiện ích đơn giản cho người sử dụng. Ta dừng lại ở trường hợp cụ thể đối với hệ điều hành Solaris:

❖ /usr/sbin/ufsdump

Sao chép dữ liệu từ đĩa cứng lên thiết bị lưu trữ theo từng mức.

Thường được dùng để lưu trữ (với mức 0) toàn bộ thư mục được chỉ định. Nhất thiết phải để hệ thống làm việc ở chế độ một người dùng (mức chạy 1) khi thực hiện **ufsdump** và **ufsrestore**.

Cú pháp:

```
[/usr/sbin/] ufsdump [option] [arguments] files_to_dump
```

Ví dụ:

```
ufsdump 0vf /dev/rmt/0n /dev/rdisk/c0t3d0s0
```

Giá trị mặc định là 9uf , /dev/rmt/0 - file đích là băng từ.

Các tùy chọn:

0 - 9

Mức lưu trữ

a *archive_file*

File hồ sơ

b *factor*

Thông số khối, mặc định là 20 block một lần ghi cho băng từ dưới 6250 BPI, 64 cho băng 6250 BPI trở lên (BPI - byte per inch).

c

Băng cartridge.

d *bpi*

Mật độ băng tính theo số byte trên mỗi độ dài của băng từ là 1 inch.

D

Lưu trữ vào đĩa mềm.

f *dump_file*

File lưu trữ. Mặc định là /dev/rmt/0, nếu là dấu "-" : ra ra chuẩn.

l

Autoload - tự động nạp băng.

o

Offline.

S

Đánh giá dung lượng.

u

Cập nhật bản lưu trữ.

v

Kiểm tra lại sau khi lưu trữ.

Lưu trữ sẽ không báo lỗi nếu số lỗi nhỏ hơn 32.

❖ **/usr/lib/fs/ufs/ufsrestore**

Khôi phục dữ liệu đã được lưu trữ bằng lệnh **ufsdump**. Trước khi thực hiện lệnh, thư mục làm việc hiện thời phải là thư mục cần khôi phục dữ liệu để các file và thư mục con về lại đúng vị trí cũ. Lệnh này nhất thiết phải được thực hiện bằng quyền root.

Cú pháp:

```
[/usr/lib/fs/ufs/]ufsrestore [option] [arguments]  
[filename ...]
```

Ví dụ:

```
ufsrestore xfs /dev/rmt/0n /export/oracle
```

Một trong các tùy chọn sau là bắt buộc:

i

Chế độ hỏi đáp

r

Chế độ đệ quy. Thường dùng tùy chọn này để khôi phục đầy đủ bản lưu trữ ở mức 0.

R

Cho phép khôi phục tiếp tục sau khi việc khôi phục đầy đủ bản lưu trữ ở mức 0 bị tạm dừng.

t

Bảng nội dung kê danh sách từng file. Loại trừ lẫn nhau với các tùy chọn **x**, **r**.

x

Giãn nén. Nếu để trống - đối tượng là thư mục /.

Các tùy chọn sau là không bắt buộc:

a *archive_file*

Đọc từ file hồ sơ thay vì từ môi trường lưu giữ (băng từ)

b *factor*

Thông số khối, mặc định là 20 block một lần đọc cho băng từ dưới 6250 BPI, 64 cho băng 6250 BPI trở lên.

c

Đổi nội dung từ chuẩn 4.1BSD sang chuẩn hệ thống file *ufs* mới.

d

Gỡ rối.

f *dump_file*

File lưu trữ. Mặc định là */dev/rmt/0*, nếu là dấu “-” : đọc từ vào chuẩn.

s *n*

Bỏ qua (n-1) file và khôi phục từ file thứ n.

v

Hiển thị tên và inode của từng file được phục hồi

❖ **/usr/sbin/tar**

Gom và thêm file và thư mục và lưu trên thiết bị lưu trữ, bung ra khi cần thiết. Cú pháp của lệnh có các dạng như sau:

```
/usr/sbin/tar c {bBefFhilvwX [ 0-7]} [device] [block]
    [exclude-filename ...] [-I include-filename ]
    filename... [-C directory filename].....
/usr/sbin/tar r{bBefFhilvw [ 0-7]} [device] [block]
    [-I include-filename]filename ... [-C directory
    filename]... ..
/usr/sbin/tar t[BefFhilvX [ 0-7]} [device] [exclude-
    filename ...] [-I include-filename ]filename .....
```

```
/usr/sbin/tar u [bBefFhilvw [ 0-7]] [device] [block]
      filename ...
/usr/sbin/tar x [BefFhilmopvwX [ 0-7]] [device]
      [exclude-filename .....] filename .....
```

Một trong các tùy chọn sau là bắt buộc:

c

Tạo file lưu trữ, bắt đầu ghi từ đầu file lưu trữ. Loại trừ lẫn nhau với **r**.

r

Thay thế. Loại trừ lẫn nhau với **c** và **u**.

t

Bảng nội dung kê danh sách từng file. Giống như lệnh **ls -l**.

u

Cập nhật. Thêm file vào nếu chưa có hoặc bị sửa đổi từ lần lưu trữ trước.

x

Giãn nén hoặc khôi phục.

Các tùy chọn sau là không bắt buộc:

b

Thông số khối, mặc định là 1, cực đại là 20 block. Xác định bởi dung lượng bộ nhớ sẵn có khi chạy lệnh **tar**.

B

Block. Bắt **tar** đọc nhiều lần cho đến khi điền đủ block. Cần khi chạy lệnh **tar** qua mạng Ethernet.

e

Lỗi. Nếu có lỗi sẽ làm dừng ngay lệnh **tar**.

f

File. Bắt **tar** dùng thiết bị chỉ bởi thông số **device**. Nếu để trống, **tar** sẽ dùng thiết bị được chỉ bởi biến môi trường **TAPE**, nếu là dấu **"-"**: đọc từ vào chuẩn và ghi ra ra chuẩn.

F

Loại khỏi danh sách tất cả các thư mục được đặt tên **SCCS**. Nếu có hai thông số này, loại thêm các file **.o**, các file có tên **errs**, **core**, **a.out**.

h

Tuân theo các liên kết hình thức.

i

Bỏ qua kiểm tra lỗi checksum của thư mục.

l

Liên kết. Sẽ thông báo nếu không giải quyết hết được mọi liên kết tới file đã lưu trữ.

m

Sửa đổi. Với tùy chọn *x*. Không khôi phục thời gian sửa đổi của file mà lấy luôn thời gian khôi phục.

o

Sở hữu. Gắn luôn định danh người dùng và nhóm đang chạy lệnh **tar**. Dùng với tùy chọn *x*.

p

Khôi phục nguyên trạng chủ quyền (mode) của file. Dùng với tùy chọn *x*.

v

Hiển thị tên của từng file được xử lý.

w

In từng thao tác sẽ thực hiện, kèm theo tên file, chờ người dùng xác nhận.

x

Exclude. Loại các file có tên trong các *exclude-filename*.

0 - 7

Chọn các ổ băng từ khác với ổ mặc định */etc/default/tar*.

❖ **cpio**

Sao lưu vào và ra các file.

Cú pháp:

```
cpio -i [bBcdfkmrsStuvV6] [-C bufsize] [-E filename] [-H
header] [-I filename [-M message]] [-R id]
[pattern]...
```

```
cpio -o [aAÂBcLvV] [-C bufsize] [-H header] [-O filename
[-M message]]...
```

cpio -p[adlLmuvV] [-R *id*] *directory*...

Các tùy chọn:

i

(Copy in) giải nén file từ vào chuẩn.

o

(Copy out) đọc từ vào chuẩn để lấy danh sách đường dẫn rồi sao những file đó ra ra chuẩn.

p

(Pass) đọc từ vào chuẩn để lấy danh sách đường dẫn.

a

Đặt lại thời gian truy nhập file sau khi chúng được sao lưu.

A

Thêm file vào cuối bản lưu trữ. Đi kèm với tùy chọn O.

b

Thứ tự ngược của các byte trong mỗi từ. Đi kèm với tùy chọn i.

B

Kích thước của block vào/ra để ghi. Buffer mặc định 512 byte.

c

Đọc và ghi thông tin header bằng ký tự ASCII. Hay dùng, nếu máy nguồn và máy đích khác loại.

C

Mặc định 512 byte.

d

Tạo thư mục nếu cần.

E *filename*

Chỉ ra file đầu vào chứa danh sách các file sẽ được giải nén..

f

Sao lưu tất cả các file, trừ những file có tên trong *pattern*.

H *header*

Tiêu đề.

k

Bỏ qua các file bị hỏng.

l

Tạo liên kết hình thức, được dùng nếu có thể thay cho việc sao chép.

R *id*

Gán lại các thông tin chủ quyền và nhóm cho từng file (bằng ID của root).

t

Chỉ in ra nội dung các tên file mà không sao lưu.

u

Sao lưu không điều kiện.

v

Sao lưu và in ra nội dung các tên file.

❖ **dd**

Chuyển đổi hay sao chép toàn bộ dữ liệu của băng từ, đĩa cứng, đĩa mềm. Cú pháp có dạng:

```
dd [option=value]
```

Các tùy chọn:

if=filename

Thiết bị nguồn.

of=filename

Thiết bị đích.

ibs=n

Kích thước một khối vào. Mặc định 512 byte, max: 4098.

obs=n

Kích thước một khối xuất. Mặc định 512 byte, max: 4098.

bs=n

Kích thước của cả khối vào và ra.

skip=n

Nhảy qua n khối rồi mới bắt đầu sao lưu.

count=n

Chỉ sao lưu n khối.

❖ **mt**

Trong thực tế sử dụng, lệnh điều khiển ổ băng từ **mt** tuy không có tác dụng lưu trữ dữ liệu, nhưng rất hay được dùng tới. Cú pháp của nó như sau:

```
mt [-f tapename] command ... [count]
```

Nếu *tapename*, liên quan tới thiết bị ký tự băng từ, không được chỉ ra, lệnh **mt** sẽ dùng giá trị của biến môi trường **TAPE**. Nếu biến môi trường cũng không có, thiết bị mặc định **/dev/rmt/0** sẽ được dùng. Theo mặc định, lệnh thực hiện thao tác một lần, số lần thao tác lặp lại được đặt bởi *count*.

Các lệnh con của **mt**:

eof, weof	Đặt các dấu EOF trên vị trí hiện thời của băng
fsf	Cuốn băng về trước qua <i>count</i> dấu EOF
fsr	Cuốn băng về trước qua <i>count</i> bán ghi
bsf	Cuốn ngược băng qua <i>count</i> dấu EOF
bsr	Cuốn ngược băng qua <i>count</i> bán ghi
nbsf	Cuốn ngược băng qua <i>count</i> file, tới khối đầu của file. Tương đương tổ hợp lệnh count+1 bsf và fsf .
asf	Dùng lượng tuyệt đối cho tới số <i>count</i> file. Tương đương tổ hợp lệnh rewin và fsf .

Các lệnh sau không cần *count*:

eom	Cuối băng
rewin	Cuốn về đầu băng
offline, rewoffl	Cuốn về đầu băng, nếu có thể, đưa ổ băng về trạng thái nghỉ
status	Hiển thị trạng thái
retention	Tua băng xuôi ngược làm giảm độ căng cơ học
erase	Xóa toàn bộ băng

Tên các thiết bị băng được dùng trong các hệ thống UNIX, trước hết là Solaris (dấu * là các số thứ tự, bắt đầu từ 0):

```

/dev/rmt/*, /dev/rmt/*b, /dev/rmt/*bn, /dev/rmt/*c,
/dev/rmt/*cb, /dev/rmt/*cbn, /dev/rmt/*cn,
/dev/rmt/*h, /dev/rmt/*hb, /dev/rmt/*hbn,
/dev/rmt/*hn, /dev/rmt/*l, /dev/rmt/*lb,
/dev/rmt/*lbn, /dev/rmt/*ln, /dev/rmt/*m,
/dev/rmt/*mb, /dev/rmt/*mbn, /dev/rmt/*mn,
/dev/rmt/*u, /dev/rmt/*ubn

```

X. 2. b. Các lệnh và tiện ích trong các hệ điều hành UNIX khác

Các lệnh **cpio**, **tar**, **dd**, **mt** không bị thay đổi quá nhiều khi chuyển từ hệ điều hành UNIX này sang hệ UNIX khác. Có chăng là các tùy chọn của chúng bị thêm bớt hoặc sửa đổi nội dung. Nhưng điều đó không đảm bảo một lệnh được viết trong một hệ UNIX, ví dụ như SCO UNIX có thể chạy suôn sẻ trên các hệ thống khác như AIX chẳng hạn. Ta chỉ có thể dựa vào sự tương tự của các lệnh để không phải cảm thấy quá bất ngờ khi đột nhiên phải làm việc với một hệ thống hoàn toàn mới lạ. Nhận xét trên tỏ ra không đúng đối với cặp lệnh **ufsdump**, **ufsrestore** của Solaris. Bảng sau cho ta thấy tên gọi các lệnh tương ứng với lệnh **ufsdump** của Solaris, cũng như các file thiết bị mà chúng sử dụng trong vài hệ UNIX:

Tên hệ UNIX	Tên lệnh	Thiết bị mặc định
Digital UNIX	dump	/dev/rmt0b
HP-UX	dump	/dev/rmt/0m
IRIX	dump	/dev/tape
SunOS	dump	/dev/rmt8, /dev/rst0
AIX	backup	/dev/rfd0
Solaris	ufsdump	/dev/rmt/0
Linux	-	-

■ Tạo file mục lục

Trong thực tế sử dụng đôi khi ta rất muốn biết danh sách các file và thư mục được lưu trữ hay khôi phục lại, nhất là khi phải làm việc với các hệ thống thư mục

nhiều mức, hay với thao tác lưu trữ gia tăng. Thông tin như vậy được chứa trong file được gọi là *file mục lục*.

Có thể dùng lệnh **tar** hay **cpio** với tùy chọn **-v** để tạo ra file mục lục. Ta hãy xét các ví dụ sau:

```
$ today=`date +%d%b%y`  
$ tar -cv /home > /backup/home_full_$today.TOC  
$ tar -cv /home | tee /backup/home_full_$today.TOC
```

Cả hai lệnh **tar** đều nén và lưu nội dung của thư mục **home**, tạo ra một file kiểu giống như liệt kê thư mục và có tên dài liên quan tới ngày tháng hiện thời, ví dụ **/backup/home_full_25dec99.TOC**.

Lệnh **cpio** gửi danh sách file tới sai chuẩn, nên thông tin đó có thể nắm bắt được theo cách hơi khác một chút:

```
$ toc=`date +/backup/home_full_5d%b%y.TOC`  
$ find /home -print | cpio -ov > /dev/rmt/0 2> $toc
```

Nếu dùng C shell, câu lệnh sẽ biến dạng nhẹ thành:

```
$ set toc=`date +/backup/home_full_5d%b%y.TOC`  
$ (find /home -print | cpio -ov > /dev/rmt/0) >& $toc
```

Kết quả ra với cách dùng lệnh **cpio** như ở trên chỉ gồm có mỗi tên đường dẫn của các file. Để có được kết quả đầy đủ hơn, ta phải sử dụng thêm một lệnh **cpio** nữa trong đường ống phức tạp sau:

```
$ cpio -itv < /dev/rmt/0 > $toc  
$ find /home -exec ls -ld {} \; | tee $toc | awk  
    '{print $NF}' | cpio -o > /dev/rmt/0
```

Đường ống trên được thực hiện như sau. Lệnh **cpio** đầu lập danh sách các file trong thư mục. Lệnh **cpio** thứ hai để tránh phải đọc lại băng từ đã dùng lệnh **find**

tạo ra một liệt kê có định dạng dài (và cách nhau bằng dấu chấm phẩy ";"). Kết quả trên được lưu vào file mục lục và được đưa qua lệnh **awk**, lệnh này lại nhặt lấy các trường cần thiết và chuyển chúng đến lệnh **cpio** thứ hai nhằm lưu lại file mục lục.

Đối với các lệnh **dump** và **restore** thao tác có phức tạp hơn. Ta xét một shell script chạy trên C shell với các lệnh **dump** và **restore** chạy trên các hệ điều hành không phải Solaris sau:

```
#!/bin.csh
# bkup+toc - thực hiện nên lưu trữ
# và kiểm tra bằng tu/ tạo file TOC
# $1 = filesystem
# $2 = dump level (default = 0)
#
if ($#argv < 1) then
    echo "do_backup : filesystem [dump-level]"
    exit 1
endif

set lev=0
if (" $2 " != " ") set lev=$2
dump ${lev}uf /dev/rmt1 $1
if ($status) then
    echo "do_backup : dump failed"
    exit 1
endif
restore tvf /dev/rmt1 > /backup/~date +$1:t_%m-%d- \
    %y.$lev`
```

Kết quả ra, ví dụ là `quyettoan_12-31-99.2`, có nghĩa là file mục lục của việc lưu trữ và khôi phục ở mức 2 (gia tăng) đối với thư mục `quyettoan` vào ngày 31 tháng 12 năm 1999 trong ví dụ trên.

▪ *Tiện ích **mksysb** của AIX*

Một số hệ điều hành UNIX có sẵn các tiện ích đặc biệt nhằm tạo ra các băng từ khởi động được từ các hệ thống đang vận hành và có thể tự động khôi phục hệ thống khi sự cố xảy ra. AIX cho ta tiện ích **mksysb** nhằm mục đích trên, lưu giữ tất cả các hệ thống file trong nhóm root, thường là /, /usr, /var, /home (nếu không bị bỏ bớt đi) và cả /tmp cùng một số thư mục và file khác. Tiện ích này được kích hoạt như sau lần lượt cho các phiên bản AIX 3.2.5 và AIX 4.1:

```
# mkszfile && mksysb /dev/rmt0      # AIX 3.2.5
# mksysb -i /dev/rmt0              # AIX 4.1
```

Tất cả các thông tin về cấu hình hệ thống được **mkszfile** (AIX 3.2.5) hoặc **mksysb** với tùy chọn **-i** ghi lên băng từ. Tùy chọn **-m** được dùng nếu ta muốn khôi phục lại nội dung và vị trí của các hệ thống file trên vào chỗ cũ trên hệ thống đang hoạt động.

Sau khi khắc phục xong sự cố phần cứng, máy chủ sẽ được khởi động từ băng từ và xuất hiện các dòng thông báo tương ứng theo các phiên bản AIX trên. Quá trình khôi phục thông tin cấu hình hệ thống từ băng từ được **mksysb** thực hiện tự động.

Tất nhiên, ta có thể sao chép kỹ thuật này và phát triển sang các hệ điều hành khác. Ta phải viết một shell script để quy định nội dung công việc giống như của **mksysb**. Điều kiện cần là máy chủ với cấu hình phần cứng cụ thể phải hỗ trợ cho việc khởi động từ băng từ. Trong trường hợp ngược lại, có thể dùng đĩa CD, MO, hay đĩa cứng thay vào vị trí của băng từ.

Tiện ích **mksysb** cũng có thể dùng không phải như phương tiện cấp cứu, mà như tiện ích lưu trữ và khôi phục dữ liệu thông thường. Các băng từ **mksysb** được chia làm bốn phần và phần cuối cùng dành cho các thư mục và file riêng lẻ mà người quản trị chọn ra để lưu trữ và khôi phục lại, nếu cần. Để khôi phục lại, ví dụ, thư mục /etc/mf và file /usr/bin/csh, trong hệ AIX 3.2.5 ta làm như sau:

```
# mt -f /dev/rmt0 rewind          # quay lại đầu băng
# mt -f /dev/rmt0.1 fsf 3         # dinh vi đầu tu
# tar -xf /dev/rmt0 ./bin/csh ./etc/mf # gian nen file
```

Trong hệ AIX 4.1, thao tác tương tự được thực hiện bằng lệnh:

```
# restore -s 4 -x -q -f /dev/rmt0 ./bin/csh ./etc/mf
```

▪ *Tạo băng từ khởi động trong IRIX*

Để có thể khôi phục được hệ thống file root từ băng từ cấp cứu trong hệ điều hành IRIX, trước hết ta phải hoàn thành hai thao tác chuẩn bị. Trước hết, ta phải tạo băng từ **sash** bằng cách đưa đĩa CD-ROM cài đặt của IRIX vào máy (máy sẽ tự mount đĩa CD), rồi gõ lệnh:

```
# distcp /CDROM/dist/sa /dev/tape
```

Nếu không dùng ổ băng từ mặc định, ta phải thay tên ổ băng từ vào chỗ /dev/tape.

Sau đó, phải tạo bản lưu trữ đầy đủ của hệ thống file root bằng tiện ích **Backup** - cải tiến của tiện ích **bru**, cho phép tạo băng từ khởi động. Trước khi lưu giữ các file và thư mục, **Backup** gọi lệnh **/etc/dvhtool** để đặt tiêu đề cho đĩa cứng, giúp cho việc khôi phục khối khởi động của đĩa sau này.↵

Sau đây là quy trình khôi phục lại hệ thống file root:

- Bật điện máy tính, nhưng ngắt quá trình khởi động máy bằng phím Esc.
- Chọn *Recover the System* từ thực đơn xuất hiện trên màn hình, sau đó đưa băng từ **sash** vào ổ băng. Chương trình cài đặt được nạp.
- Chọn tên ổ băng từ dùng để khôi phục. Trước hết xác định là ổ băng tại chỗ hay từ xa, tên thiết bị mặc định /dev/tape hay tên khác.
- Khi được nhắc trên màn hình, đưa băng từ **Backup** vào và chờ đến khi hệ thống được khôi phục xong.

Tiện ích **bru**, cho phép thực hiện các thao tác lưu trữ và khôi phục các thư mục và file theo lựa chọn của người dùng.

▪ *Tạo đĩa mềm khởi động cho 'Linux và SCO 4.1.2'*

Đĩa mềm khởi động không những là phương tiện để khôi phục hệ thống UNIX mà đối với một số hệ UNIX còn là một biện pháp bảo mật. Trong trường hợp thứ hai, đĩa mềm do người quản trị bảo quản và sử dụng. Ở đây, ta quan tâm tới việc khởi động một hệ thống UNIX gặp sự cố bằng đĩa mềm, sau đó nhờ vào các băng từ lưu trữ từ trước để khôi phục dữ liệu. Đối với Linux, việc tạo đĩa khởi động rất đơn giản:

```
$ dd if=/zImage of=/dev/fd0
```

Bạn hãy thử làm tương tự với Solaris, thay zImage bằng .vmunix, /dev/fd0 bằng /dev/diskette0. Nhớ rằng ảnh zImage của kernel cho Linux khá nhỏ, cỡ 700 kB, nhưng đối với các hệ thống Solaris phức tạp, kích thước của .vmunix có thể vượt quá dung lượng 1,44 MB của đĩa mềm thông dụng hiện nay. Đối với các hệ UNIX, còn có một giải pháp tổng quát hơn là khởi động máy bằng môi trường cài đặt (đĩa CD). Nhưng vì lý do bảo mật hay an toàn, không phải ai cũng có quyền dùng môi trường cài đặt để sửa đổi lại cấu hình hệ thống hay thay đổi cấu trúc hệ thống file trên máy.

Đối với cả hệ thống file root và hệ thống file người dùng, các thao tác lưu trữ và khôi phục dữ liệu được thực hiện rất tốt nhờ lệnh **tar** của GNU.

Trong SCO UNIX, lệnh tạo đĩa khởi động là:

```
# mkdir fd
```

Sau khi đưa tên file thiết bị của đĩa mềm, một thực đơn sau sẽ xuất hiện trên màn hình:

```
Choices for content of floppy filesystem
```

1. Filesystem
2. Bootable only (96ds15 and 135ds18 only)
3. Root filesystem only (96ds15 and 135ds18 only)

Enter the option or q to exit:

Bạn cần chọn mục 2, sau đó là mục 3 để tạo ra hai bộ đĩa mềm. Do đĩa mềm có dung lượng nhỏ, đôi khi trong đĩa mềm lưu hệ thống file root, bạn chỉ cần ghi lại vài lệnh tối cần thiết cho việc khôi phục hệ thống như sau:

```
# mount /dev/fd0 /mnt
# cp /bin/mkdir /mnt/bin
# cp /bin/tar /mnt/bin
# cp /bin/file /mnt/bin
# cp /usr/bin/mkdir /mnt/bin
```

Trong trường hợp khẩn cấp, ta cũng nên tham khảo cách lấy file từ môi trường cài đặt của SCO UNIX để sửa chữa trong hệ thống bị gặp sự cố. Đối với SCO UNIX, lệnh thường dùng cho các hệ UNIX khác như sau không thực hiện được:

```
# mount /dev/cd0 /mnt
```

do đĩa CD không thuộc định dạng chuẩn ISO 9660 (High Sierra). Để có thể khôi phục file nào đó, bạn cần phải biết file đó nằm ở đĩa CD nào. Các file hệ thống thường nằm ở đĩa CD 1 cho hầu hết các hệ UNIX. Giả sử trong trường hợp này, file `/bin/csh` bị xóa mất hay bị hỏng và cần được khôi phục lại. Ta phải tìm tên file đó trong danh sách bằng cách lần theo mục lục của đĩa:

```
# tar tvf /dev/nrcdt0 | more
```

Khi tìm thấy file, tốt nhất bạn nên ghi lại các thuộc tính của file vì các thông tin đó về sau sẽ còn được dùng tới. Sau đó, ta trở về đầu của kho dữ liệu chứa file được tìm thấy và lại dùng lệnh `tar`, nhưng lần này để giãn nén nó vào vị trí vốn có trong hệ thống UNIX:

```
# tar xvpf /dev/nrcdt0 ./bin/csh
```

Tùy chọn `p` sẽ đặt đúng chủ quyền, sở hữu và các thuộc tính khác cho file.

Tiện ích **fsphoto** là cải tiến của **cpio**, cho phép đặt lịch để tự động lưu giữ các thư mục và file cần thiết. Tiện ích này được gọi bởi lệnh:

```
# /usr/lib/sysadmin/fsphoto
```

- *Các tiện ích trong HP-UX*

HP-UX cung cấp hai tiện ích **fbackup** và **frecover** cho các mục đích lưu trữ và khôi phục dữ liệu. Ta hãy xét cách dùng chúng trong các trường hợp sau (mỗi dòng lệnh tương ứng với một tình huống giả định):

```
# fbackup -0u -f /dev/rmt/1m -i /physics
# fbackup -lu -i /chem -i /bio -e /bio/med
# fbackup -0u -g /backup/math -I /backup/math.TOC
# frecover -x -i /chem/fullerenes
# frecover -r -f /dev/rmt/1m
```

Trong ví dụ đầu, thư mục `/physics` được lưu trữ đầy đủ, cập nhật nội dung so với lần thao tác trước trong ổ băng từ số 1. Ở ví dụ thứ hai, ta thực hiện lưu trữ mức 1 các thư mục `/chem` và `/bio`, nhưng trừ thư mục con `/bio/med`. Ví dụ thứ ba lưu trữ đầy đủ thư mục `/backup/math` và ghi chỉ số lưu trữ vào file có tên `/backup/math.TOC`. Ví dụ thứ tư có câu lệnh chỉ khôi phục thư mục con `/chem/fullerenes`, nhưng trong ví dụ cuối cùng ta khôi phục tất cả các file có trong băng từ của ổ băng từ 1.

- *Cải tiến lệnh tar của SunOS và Solaris*

Cải tiến của lệnh **tar** trong SunOS và Solaris giúp cho lệnh này có thể thực hiện được ở mức hệ thống.

X. 2. c. Các file đặc biệt thiết bị băng từ

Dạng chung của tên gọi các file thiết bị băng từ đối với các hệ điều hành UNIX là `/dev/rmn` hay `/dev/rmt/n`, với `n` là chữ số chỉ số thứ tự của ổ băng từ. Nói chung, các ổ băng từ đều được truy nhập tới nhờ vào các file thiết bị ký tự (xem

chương III *Hệ thống file của UNIX*), tên gọi của chúng trong các hệ UNIX thường được thêm các ký hiệu phía trước và/hoặc phía sau. Ví dụ, trong Solaris, tên các file thiết bị băng từ có thể viết là

```
/dev/rmt/0l
/dev/rmt/0ln
/dev/rmt/0m
/dev/rmt/0lm, v.v.,
```

chữ số (ở đây là 0) chỉ số thứ tự của ổ băng từ, l, m, h, u để chỉ mật độ ghi dữ liệu thấp, vừa, cao và rất cao, tương ứng; b - để chỉ rằng dữ liệu được ghi theo chuẩn BSD UNIX; n - không cuộn lại đến đầu băng, dùng trong trường hợp muốn ghi tiếp sau vị trí trên băng từ đã ghi dữ liệu lần trước.

Trong một vài hệ điều hành các ký tự chỉ mật độ dữ liệu lại được dùng để giải mã chế độ truy nhập, như file thiết bị băng từ của Digital UNIX `/dev/rmt0!` quy định ghi không nén dữ liệu khi dùng với loại băng từ 4 mm (mặc định là ghi có nén). Nếu ký tự chỉ mật độ bị bỏ qua, các băng từ chỉ được dùng với mật độ đơn. AIX có chế độ ký hiệu khác hẳn:

<code>/dev/rmt0</code>	Cuộn lại khi ghi xong, không giảm lực căng trước khi ghi, mật độ #1 (cao, có nén dữ liệu)
<code>/dev/rmt0.1</code>	Không cuộn lại khi ghi xong, không giảm lực căng trước khi ghi, mật độ #1
<code>/dev/rmt0.2</code>	Cuộn lại khi ghi xong, có giảm lực căng trước khi ghi, mật độ #1
<code>/dev/rmt0.3</code>	Không cuộn lại khi ghi xong, có giảm lực căng trước khi ghi, mật độ #1
<code>/dev/rmt0.4</code>	Cuộn lại khi ghi xong, không giảm lực căng trước khi ghi, mật độ #2 (thấp, không nén dữ liệu)
<code>/dev/rmt0.5</code>	Không cuộn lại khi ghi xong, không giảm lực căng trước khi ghi, mật độ #2

<code>/dev/rmt0.6</code>	Cuốn lại khi ghi xong, có giảm lực căng trước khi ghi, mật độ #2
<code>/dev/rmt0.7</code>	Không cuộn lại khi ghi xong, có giảm lực căng trước khi ghi, mật độ #1

Giả sử ID của thiết bị SCSI là 4 (xem chương XIII “Bảo trì phần cứng hệ thống”), ta có thể tổng kết cách gọi tên của ổ băng từ thứ nhất trong các hệ UNIX khác nhau như sau:

Hệ UNIX	Chế độ có cuộn băng lại	Chế độ không cuộn băng lại
SunOS	<code>/dev/rst0</code>	<code>/dev/nrst0</code>
Solaris	<code>/dev/rmt/0</code>	<code>/dev/rmt/0n</code>
Digital UNIX	<code>/dev/rmt0</code>	<code>/dev/nrmt0</code>
AIX	<code>/dev/rmt0</code>	<code>/dev/rmt0.1</code>
IRIX	<code>/dev/rmt/tps0d4</code> <code>/dev/tape</code>	<code>/dev/rmt/tps0d4nr</code> <code>/dev/nrtape</code>
HP-UX 9	<code>/dev/rmt/c201d4m</code>	<code>/dev/rmt/c201d4mn</code>
HP-UX 10	<code>/dev/rmt/clt4d0BESTb0mb</code>	<code>/dev/rmt/clt4d0BESTb0mn b</code>
Linux	<code>/dev/st0</code>	<code>/dev/nst0</code>
SCO UNIX	<code>/dev/rStp0</code>	<code>/dev/nrStp0</code>

X. 3. Ví dụ thực hiện các lệnh lưu trữ của Solaris

- *lệnh tar*

- Lưu mới thư mục `/export` lên băng từ:

```
# tar cvf /dev/rmt/0 /export
```

- Cập nhật sự thay đổi lên bảng từ:

```
# tar ufs /dev/rmt/0 /export
```

- Thêm vào thư mục /tools:

```
# tar rvf /dev/rmt/0 /tools
```

- Xem nội dung bảng từ:

```
# tar tvf /dev/rmt/0
```

- Giãn nén thư mục /export:

```
# tar xvf /dev/rmt/0 /export
```

- *lệnh ufsdump*

- Lưu đầy đủ thư mục /export :

```
# ufsdump ovf /dev/rmt/0 /export
```

- Lưu gia tăng ở mức 5 thư mục /export:

```
# ufsdump 5vf /dev/rmt/0 /export
```

- *lệnh ufsrestore*

- Khôi phục thư mục /export:

```
# ufsrestore xvf /dev/rmt/0 /export
```

X. 4. Quản trị tự động lưu trữ - khôi phục dữ liệu bằng Solstice Backup

Solstice Backup (hay còn gọi là **Networker**) là một phần mềm dùng để lưu trữ các file và thư mục trên mạng diện rộng không đồng bộ của các hệ thống máy tính, là sản phẩm của Legato Inc., được hãng Sun mua lại bản quyền. Networker được

thiết kế cho nhiều môi trường Client/Server khác nhau, nên thích hợp với bất kỳ môi trường mạng và hệ điều hành nào.

Hàng ngày, Networker tự động lưu trữ dữ liệu của các máy tính trên mạng, bảo vệ các file của hệ thống khỏi bị hỏng hóc. Với Networker, việc quản lý các thiết bị lưu trữ, các thông báo khi xuất hiện các sự cố, các thao tác quản trị diễn ra một cách dễ dàng, đơn giản qua giao diện X-Windows (GUI).

Người dùng dễ dàng khôi phục các file bị hỏng hóc thông qua GUI để chọn lựa. Networker quản lý tiến trình lưu trữ và khôi phục dữ liệu thông qua 2 loại cơ sở dữ liệu thông tin: *chỉ số file (file Index)* và *chỉ số thiết bị lưu trữ (Media Index)*. Chỉ số file lưu một đầu vào cho mỗi file được lưu trữ; chỉ số thiết bị quản lý các khối dữ liệu lưu trữ của các file được cất giữ. Các khối dữ liệu lưu trữ được hiểu là những thiết bị lưu trữ như băng từ.

Bản Networker hay Solstice Backup 4.2 là phiên bản hội đủ các tính năng đặc trưng của dòng sản phẩm này. (Hiện nay, bản 5.1 đã được phát hành). Ví dụ, đối với Sun SPARC, Solstice Backup 4.2 có các tính năng sau đây:

- Hierarchical Storage Management (HSM) - quản lý lưu trữ theo kiến trúc - đối với Solaris (từ 2.3, 2.4, 2.5) và SunOS (từ 4.1.3, 4.1.4).
- Tăng khả năng chạy được máy nhiều CPU từ 16 lên 32.
- Với Backup Server Edition hoặc TurboPak số thiết bị lưu trữ tăng từ 10 lên 16.
- SmartMedia (TM) hỗ trợ các thiết bị đơn lẻ.
- Hỗ trợ các lệnh tiền/hậu xử lý.
- Ghi lại file log các thao tác lưu trữ.
- Cải tiến hiệu năng: giảm thời gian CPU; kiểm tra và sửa chữa chỉ số.

Trong các hệ thống nhỏ, không nhất thiết phải lưu trữ dữ liệu bằng tiện ích tương tự Solstice Backup, người đọc có thể bỏ qua mục này .

▪ *Cài đặt và loại bỏ Solstice Backup*

 Solstice Backup thường được cài đặt từ môi trường đĩa CD với quyền của root. Ta phải tuần tự thực hiện các bước sau:

- Cài đặt ổ băng từ. Thường thao tác này được thực hiện ngay từ khâu cài đặt hệ thống.
- Đưa đĩa CD-ROM cài đặt Solstice Backup vào ổ CD. Đĩa CR-ROM sẽ được hệ thống tự động mount và ta chuyển dấu nhắc hệ thống đến đó, ví dụ:

```
# cd /cdrom/networker_4_1_2
```

- Gõ lệnh:

```
# swmtool &
```

- Đặt thuộc tính
- Chọn phần mềm để cài đặt. Đối với các máy chủ UNIX hoạt động như các client trong thao tác lưu trữ – khôi phục dữ liệu ta chỉ cần chọn phần mềm cho client, nhưng đối với các máy có vai trò *backup server*, ta phải chọn cả các gói phần mềm cho cả client và server (khoảng 21 MB trong tổng số 32 MB). Sau khi cài đặt, các file sẽ được ghi vào các thư mục */usr/lib/nsr*, */usr/bin*, *usr/sbin*, */usr/man*.

Sau khi bắt đầu cài đặt, ta phải trả lời một số câu hỏi, kể cả các phần có liên quan tới các thuộc tính đã xác định ở bước trước. Các câu hỏi đó có nội dung như phải cất giữ thông tin về máy chủ trong thư mục nào; nếu thư mục đó chưa có thì có tạo mới không, tên và loại của thiết bị đọc băng từ (ổ băng từ), ví dụ *"/dev/rmt/0hbn"* và *"qic"*. Nếu máy chủ UNIX của ta có nhiều thiết bị băng từ, ta phải ấn phím Enter để tiếp tục khai báo. Sau khi được hỏi hai lần về việc khởi động các daemon của Solstice Backup, ta trả lời *"Yes"*. Thông thường, quá trình cài đặt chiếm khoảng 9 phút.

 Nếu muốn loại bỏ Solstice Backup ra khỏi hệ thống UNIX, ta chọn chế độ Remove trong cửa sổ *swmtool* và trả lời khi được yêu cầu khẳng định:

- xóa các bó phần mềm
- xóa các file cấu hình và cơ sở dữ liệu của Solstice Backup.
- xóa các file cấu hình và cơ sở dữ liệu của backup server.
- xóa cấu hình jukebox.
- xóa thư mục RAP.

- xóa thư mục `login`.
- xóa các thông tin về chỉ số (index) và tên các băng từ (volume)
- xóa thư mục rỗng `./nsr`.

▪ *Sử dụng giao diện*

Các giao diện của trình duyệt trên Solstice Backup (Networker) có những đặc trưng chung được mô tả trong phần này như sau:

Xem nội dung thư mục bằng lệnh thực đơn *View*. Thư mục được hiển thị theo cấu trúc dạng cây phân cấp. Ta có thể mở rộng để xem thư mục theo kiểu mở rộng chi tiết thêm một mức (lệnh *Expand one level*) hay hiển thị tất cả các mức trong thư mục con của thư mục ta quan tâm bằng lệnh mở rộng một nhánh – *Expand Branch*. Đối với các nhánh, thư mục lớn, thao tác mở rộng có thể chiếm đến một vài phút. Ngược lại, ta cũng có thể thu gọn lại như trước khi mở bằng cách chọn lệnh thực đơn *Collapsing Branch*. Các thao tác mở rộng hay thu gọn có thể được thực hiện một cách đơn giản là nhấn liên tiếp hai lần phím chuột trái vào thư mục được chọn.

Một thao tác bắt buộc phải thực hiện là đánh dấu và bỏ đánh dấu các file và thư mục cần lưu trữ hay phục hồi. Ta có thể đánh dấu hay hủy bỏ dấu đã đánh tại vị trí các hộp kiểm nằm bên trái mỗi biểu tượng của một file hay thư mục. Một cách khác nữa là đưa vệt sáng đánh dấu tới hàng chứa biểu tượng của file hay thư mục đã chọn rồi dùng các lệnh *Mark* và *Unmark*, hoặc nhấn phím Space bar. Chú ý rằng nếu ta đánh dấu hay bỏ đánh dấu một thư mục thì điều đó có nghĩa là tất cả các thư mục con và các file của thư mục đã chọn đều đã chịu tác động tương ứng. Nếu hộp kiểm của một thư mục nào đó có mẫu xám, một phần hay tất cả nội dung bên trong thư mục đó đã được đánh dấu chuẩn bị cho các thao tác lưu trữ hay khôi phục dữ liệu.

Để quan sát chi tiết một file, ta có thể sử dụng hai lệnh *File Name* và *File Detail*. Lệnh đầu là lệnh được gọi một cách mặc định và cho ta biết tên các file sắp xếp theo thứ tự vần chữ cái. Lệnh sau hiển thị các thông tin như sau cho mỗi file: tên file, dung lượng tính theo block, chủ quyền truy cập file, số các liên kết, đích trỏ đến (nếu file là liên kết hình thức), người và nhóm sở hữu, kích thước theo byte, ngày và giờ thay đổi file lần cuối. Riêng đối với giao diện Networker Recover,

lệnh *File Detail* còn hiển thị thêm một bit thông tin về số lần lưu trữ file. Thông tin này giúp ta xác định các bản lưu trữ khác nhau của một file.

Khi cần tìm một file hay một thư mục trong hệ thống file hay hệ thống các chỉ số, người ta dùng lệnh *Search*. Dấu hiệu ta phải đưa vào để tìm kiếm là một phần tên của file và vị trí thư mục mà dưới nó công việc tìm kiếm sẽ được thực hiện.

Ta có thể chuyển sang một thư mục khác bằng cách chỉ đường dẫn của nó trong trường *Selection*. Nếu muốn thoát hẳn khỏi chương trình Solstice Backup, ta chọn lệnh *Exit* trong thực đơn *File*.

▪ *Thực hiện thao tác lưu trữ dữ liệu*

Nếu bản Solstice Backup chạy trên hệ thống của bạn là bản có bản quyền đầy đủ, có chức năng *multi server*, nghĩa là hoạt động được trên nhiều server Networker, thao tác lưu trữ và khôi phục dữ liệu có thể thực hiện trên toàn mạng WAN, tại chỗ cũng như từ xa. Do chiến lược lưu trữ dữ liệu được hoạch định bởi người quản trị, nên ta có thể tổ chức lưu trữ từ trung tâm hay tạo nhiều bản lưu trữ tại các địa điểm khác nhau trên toàn mạng. Khi chạy chương trình Solstice Backup, server UNIX mặc nhiên trở thành server Networker. Tuy nhiên, hệ thống UNIX của bạn cũng có thể trở thành client để cho một server Networker khác thực hiện lưu trữ từ xa. Việc chuyển đổi server Networker được thực hiện bằng cách chọn lệnh hay kích phím chuột vào *Change Server*. Sau đó, điền tên *hostname* của server muốn chọn và nhấn *OK* để chấp nhận. Nếu tên của server chưa có trong danh sách, ta cần thực hiện lệnh *Update Server List* trước khi chọn. Tất nhiên, chương trình sẽ báo lỗi nếu server mà bạn chọn trong thực tế không phải là một server Networker như đã định.

Ta có thể thực hiện lưu trữ ngay mà không cần chờ lưu trữ tự động theo lịch bằng cách nhấn nút công cụ *Start* hay chọn lệnh *Start Backup* từ thực đơn. Tất nhiên, ngoài việc chọn server vừa trình bày ở trên, trước đó ta đã phải đánh dấu để chọn các thư mục và file cần lưu trữ bằng lệnh *Mark*. Ngoài ra, ta phải đảm bảo đã thực hiện *mount* và đặt tên (giá trị của *Volume*) cho băng từ.

Trước khi lưu trữ dữ liệu được thực sự thực hiện, hộp thoại *Backup Option* sẽ hiện ra. Việc nén file trong lưu trữ sẽ giúp giảm thiểu lưu lượng thông tin truyền trên

mạng. Vì vậy, ta nên chọn câu trả lời mặc định cho câu hỏi có nên dữ liệu hay không là “Yes”. Sau khi trả lời xong, ta chọn OK và chuyển tới phần xác định kiểu lưu trữ. Trong phần này, bạn sẽ xác định các xâu ký tự nào trong mục *Pattern* để loại khỏi bản lưu trữ. Nếu không muốn thay đổi gì, bạn hãy chọn phương án trả lời mặc định “No” và OK để tiếp tục.

Cửa sổ trạng thái lưu trữ sẽ xuất hiện ngay sau đó, cho biết thời gian bắt đầu và thời gian hoàn thành lưu trữ, tên của các file và thư mục đã lưu trữ, v.v.. Khi thực hiện xong công việc lưu trữ, trong cửa sổ đó xuất hiện dòng thông báo, ví dụ như “BACKUP COMPLETION TIME : MON DEC 25 09:30:24 2000”. Để hủy bỏ giữa chừng việc lưu trữ dữ liệu, ta chọn nút lệnh *Cancel*.

■ *Phục hồi dữ liệu*

Ưu điểm thuận tiện nhất của Networker Recover – cũng như của đa số các phần mềm phục hồi dữ liệu có giao diện đồ họa khác – là có thể xem và lựa chọn dễ dàng các thư mục và file, và cả những phiên bản lưu trữ khác nhau của chúng, cần khôi phục từ trong nội dung của băng từ lưu trữ. Để bắt đầu, ta chọn nút công cụ *Start* hay lệnh thực đơn *Start Recover*.

Thông thường, thao tác khôi phục dữ liệu được thực hiện bởi người quản trị hệ thống và trên hệ thống tại chỗ. Trong trường hợp đó có thể bỏ qua một trong hai, hay cả hai bước chọn server và client được trình bày ngay sau đây.

Nếu dữ liệu trong máy tại chỗ được lưu trữ bởi một Networker server từ xa khác, thì phải sử dụng server đó để khôi phục dữ liệu. Cách chọn Networker server bằng lệnh *Change Server* tương tự như trong phần lưu trữ dữ liệu. Còn nếu cần phục hồi dữ liệu từ một máy client khác trong mạng, ta cũng chọn một cách tương tự bằng lệnh *Change Client*. Chú ý rằng một khi ta đã chọn xong client, ta không có quyền phục hồi các dữ liệu của các client khác.

Cửa sổ *Version* trong thực đơn *View* hiển thị tất cả các thông tin về các thư mục, file được lưu trữ bởi Networker server. Tuy nhiên, ta chỉ có thể đánh dấu vào các hộp kiểm tương ứng với các file mà thôi. Thông tin được cung cấp về file ngoài những thông số như kết quả ra của lệnh *File Detail* còn có cả thời gian lưu trữ và định vị. Định vị bao gồm tên volume của băng từ lưu trữ, tên thiết bị, nơi mà

băng từ đó đã được mount. Nếu không có tên thiết bị được chỉ ra, có nghĩa là trước khi định phục hồi dữ liệu từ băng từ tương ứng, ta phải **mount** nó lên trước, đặt tên volume cho nó (có thể đặt tên ngay trong giao diện của Solstice Backup). Ta phải chọn phiên bản cần phục hồi bằng việc đánh dấu vào hộp kiểm của phiên bản đó rồi chọn *OK*. Có thể sử dụng vùng *Browse Time* để hiển thị thời gian đọc các file hoặc thay đổi mốc thời gian hiển thị các file được đọc.

Thao tác quan trọng tiếp theo là xác định nguồn dữ liệu để khôi phục. Ta có thể liệt kê các file đã được đánh dấu bằng lệnh *Show Marked* trong thực đơn *Mark*. Cửa sổ *Marked for Recover* cho thấy thời gian lưu trữ, đường dẫn của các file đã đánh dấu cho việc khôi phục dữ liệu. Nếu không nhìn thấy các file và thư mục cần khôi phục lại, ta phải xác định lại băng từ cần thiết thông qua lệnh *Load Volume*, rồi chọn qua lệnh *Volume* trong cửa sổ *View*. Nếu volume đã được chọn, ta sẽ nhìn thấy dòng thông báo "ALL ONLINE".

Trước khi bắt đầu khôi phục, ta phải dùng lệnh *Relocate* trong thực đơn *Option* để xác định vị trí của file phục hồi trong thư mục, nơi mà trước đây file đã được lưu trữ. Lệnh *Relocate* cũng sẽ tạo ra thư mục mới nếu trước đây thư mục không tồn tại (giả sử trong trường hợp toàn bộ thư mục được tạo lại mới sau sự cố).

Khi bắt đầu quá trình khôi phục, Networker hiển thị hộp thoại *Conflict Resolution* để giúp ta xác định hướng giải quyết các xung đột có thể xảy ra. Ta có thể bỏ qua các câu hỏi nhắc nhở đối với mỗi trường hợp xung đột hay lựa chọn phương án cho từng trường hợp trong hộp thoại *Filename Conflict*. Có ba phương án giải quyết được đưa ra như sau: i, *Rename the Recover file* - đổi tên file phục hồi bằng cách thêm phần đuôi mở rộng *.R* (nếu file có phần mở rộng *.R* đã tồn tại, Networker sẽ thêm một phần mở rộng *.R* khác); ii, *Discard the Recover file* - bỏ qua file phục hồi: Networker đơn giản xóa đi file phục hồi và file đang tồn tại vẫn được giữ nguyên; iii, *Overwrite the Existing file*: Networker sẽ ghi đè file phục hồi lên file cùng tên đang tồn tại. Sau khi hoàn thành việc khôi phục dữ liệu, một thông báo được đưa ra có kèm theo ghi nhận thời điểm kết thúc thao tác.

X. 5. Tiện ích băng từ **MakeIndex**

Khi đã có vài ba bộ băng từ lưu trữ thì vấn đề hay nảy sinh không phải trong khâu lưu trữ, khôi phục dữ liệu, mà trong cách quản lý và sử dụng số băng từ đó. Ta sẽ làm quen với tiện ích **MakeIndex** do Arthur Haigh viết trên Bourne shell nhằm tạo chỉ số cho các file được lưu trữ bằng nhiều lần dùng lệnh **ufsdump** trên một băng từ. Nội dung script đó như sau:

```
1  #!/bin/sh
2  #-----
3  #  MakeIndex:
4  #  Creates an index for a tape in ufsdump format
5  #
6  #-----
7  TAPE=/dev/rmt/0n
8  ERROR=0
9
10 while [ $ERROR -eq 0 ]
11 do
12     FILENUM=`mt -f $TAPE status | grep file | awk \
13             '{print $3 }' `
14     echo "\nFile: $FILENUM"
15     echo "quit\c"|ufsrestore -ivf $TAPE 2> \
16             /dev/null|grep "ump"
17     # echo "quit\c"|ufsrestore -ivf $TAPE \
18             2>/dev/null|grep "ump "
19     ERROR=$?
20 done
21 echo "EOM"
22 mt $TAPE rewofl &
23 exit
```

Thông thường băng từ lưu trữ chứa một loạt các file nén. Vòng lặp **while** làm việc theo số các file. Nhằm mục đích đơn giản, hiệu quả và linh hoạt, shell script này không xác định số file, tức là số vòng lặp, trước khi tiến hành lặp. Thay vào đó, vòng lặp **while** sẽ tiếp tục chạy khi mà biến **ERROR** vẫn có giá trị 0. Có thể thấy trong dòng lệnh thứ 12 rằng chỉ số file của bản ghi thứ nhất được xác định bằng cách giảm chỉ số file từ kết quả ra của lệnh trạng thái **mt**. Xuất chuẩn từ lệnh **mt** được đưa vào đường ống tới **grep** file và sau đó bị giảm bởi đường ống với lệnh **awk**. Cuối cùng, số chỉ vị trí file hiện thời trong băng từ được gán tới biến **FILENUM**. Dòng 13 hiện chỉ số file ra ra chuẩn. Dòng 14 cũng là dòng chính, giống như dòng 12, với phần tử quan trọng nhất là lệnh:

```
ufsrestore -ivf $TAPE
```

Các tùy chọn **-ivf** được sử dụng, trong đó **-f** có nghĩa rằng ta muốn đọc từ file được xác định bởi biến **TAPE**. Lệnh **ufsrestore** hướng điều khiển ra sai chuẩn, theo mặc định được ký hiệu bằng chỉ số 2. Sai chuẩn trong dòng 14 được đổi hướng ra thiết bị không **/dev/null** để không hiện ra màn hình.

```
ufsrestore -ivf $TAPE 2>/dev/null
```

Ta nhặt lấy chỉ các thông tin cần thiết bằng cách đổi hướng sai chuẩn tới lệnh **grep**. Thông tin có ích đối với ta chỉ có trong những dòng có xâu "dump" bên trong. Đó là **Dump**, **Dumped**, và **dump**. Vì thế ta dùng **grep** với xâu "ump" và cuối cùng là:

```
ufsrestore -ivf $TAPE 2>/dev/=>null | grep "ump"
```

Vì dùng tùy chọn **-i** cho chế độ giao tiếp, ta cần đưa vào lệnh **quit** tới vào chuẩn của lệnh **ufsrestore**. Ta lại có thể dùng đường ống để đổi hướng kết quả ra của lệnh **echo** vào vào chuẩn của **ufsrestore**:

```
echo "quit\c" | ufsrestore -ivf $TAPE 2>/=>dev/null | \
    grep "ump"
```

XI. In và quản lý in

XI. 1. Các file và tập lệnh in

❖ /etc/printcap

Là cơ sở dữ liệu chính, mô tả các máy in nối trực tiếp và nối qua mạng vào hệ thống UNIX, đã được đề cập tới trong phần *Các file quản lý hệ thống của UNIX*. Việc tạo file `printcap` đúng có ý nghĩa quyết định tới khả năng in của hệ thống, vì vậy, nên thận trọng nếu bạn muốn thay đổi nó. Trong Solaris 2.x trở đi, các thông tin chứa trong `printcap` được chuyển vào nội dung file cơ sở dữ liệu của các thiết bị đầu cuối nối chung `/etc/termcap`.

❖ /etc/lp

Chứa các tập tin dữ liệu liên quan tới máy in: `system`, `postscript`, ..., mô tả các chức năng và loại máy in được hỗ trợ.

❖ /usr/lib/lpd

Daemon chính quản lý việc in. Các lệnh liên quan:

```
/usr/ucb/lp [-c] [-m] [-p] [-s] [-w] [-d dest] [-f form-name  
  [-d any]] [-H special-handling] [-n number]  
  [-o option] [-P page-list] [-q priority-level] [-S  
  character-set [-d any]] [-t title] [-T content-type  
  [-r]] [-y mode-list] [file ...]  
cancel -u login-ID-list [printer ...]
```

Ý nghĩa của các tùy chọn cho lệnh `lp`:

`-c`

Sao lại file trước khi in.

`-d dest`

Đích. Xác định mặc định bởi biến môi trường LPDEST.

-m

Gửi mail sau khi file được in xong.

-n *number*

Số bản in, mặc định là 1.

-o *option*

Tùy chọn có thể là một hay nhiều hơn một trong số: nobanner, nofilebreak, length=XXX (ký hiệu XXX chỉ số với cơ số 10) (i - inch, c - centimeter), width=XXX, lpi= XXX (số dòng trên inch), cpi= XXX (số ký tự trên inch).

-P *page-list*

In các trang được chỉ định trong *page-list*.

-q *priority-level*

Thứ tự ưu tiên, cao nhất: 0, thấp nhất: 39.

-t

In tiêu đề trên trang đầu.

-w

Thông báo cho người dùng ở thiết bị đầu cuối khi in xong.

❖ /etc/lpadmin

Tập tin quản lý, thêm bớt máy in. Các lệnh liên quan:

```
lpadmin -p printer options
```

```
lpadmin -x dest          (bớt máy in)
```

```
lpadmin -d [dest]
```

```
lpadmin -S printer-wheel -A alert-type [-W minutes]
                                   [-Q request]
```

```
lpadmin -M -f form-name [-a[ -o filebreak] [-t tray-
                                   number] ]
```

❖ /etc/lpc

Tập tin quản lý in và hàng chờ: cấm hoặc cho phép in, sắp xếp thứ tự trong hàng chờ, thông báo trạng thái máy in.

Lệnh liên quan `/usr/ucb/lpc [command [parameter ...]]`, trong đó `command` là một trong các lệnh con sau:

help [command]	Trợ giúp
abort [all printer ...]	Kết thúc một công việc in trong hàng chờ.
clean [all printer ...]	Xóa tất cả các file tạo bởi daemon hàng chờ trong hàng chờ máy in.
disable [all printer ...]	Tắt hàng chờ của máy in.
down [all printer ...] [message]	Tắt hẳn việc in và tạo thông báo trạng thái máy in.
enable [all printer ...]	Mở hàng chờ để tiếp nhận công việc mới.
restart [all printer ...]	Khởi động lại daemon in mới. Có lợi khi tiếp tục việc in bị bỏ dở.

...

Những điều được trình bày sau đây chủ yếu thuộc về UNIX System V, trường hợp Solaris. Với Solaris, có thể thiết lập máy in và quản lý in tiện lợi hơn nhiều khi dùng giao diện đồ họa OpenWin, công cụ **AdminTool => Print Manager**. Khi áp dụng sang các hệ UNIX khác, xin hãy thận trọng và tra cứu lại các trang trợ giúp. Hệ thống in của UNIX System V gồm các thành phần sau:

- *Các lệnh cho người dùng*

lp

cancel

lpstat [-d] [-r] [-R] [-s] [-t] [-a [list]] [- [list]]

[-f[list]] [-o[list]] [-p[list]] [-D] [-u] ...

để hiển thị trạng thái dịch vụ in. Một số thông số: **c** - class; **a** - các yêu cầu đã tiếp nhận; **o** - trạng thái các yêu cầu xuất; **p** - trạng thái các máy in; **P** - loại giấy in; **r** - trạng thái lịch in; **R** - thứ tự công việc in trong hàng chờ; **s** - trạng thái tổng tiến độ; **t** - như **s**, nhưng thêm trạng thái bận, rồi của tất cả máy in; **u** - trạng thái yêu cầu ra của người dùng (`login-ID; system_name!login-ID; system_name!all, all`).

- *Taemon hàng chờ*

lp sched (thay cho **lpd**)

- *Các lệnh quản lý in*

Các lệnh này nằm trong `/usr/lib`.

accept destination

reject [-r reason] destination

enable printer

disable [-c | -W] [-r [reason]] printer ...

lpadmin

lpmove request dest, **lpmove** dest1 dest2

(chuyển yêu cầu in sang máy khác, đổi máy in)

lpshut, **lprm**

lpssystem [-t type] [-T timeout] [-R retry] [-y
"comment"] system-name [system-name ...]

lpssystem -l [system-name ...]

lpssystem -r system-name [system-name ...]

lpssystem -A

l - liệt kê các thông số; r - loại tất cả các dòng đầu vào liên quan tới system-name

lpusers -d priority-level

lpusers -q priority-level

lpusers -l

lpusers -q priority-level -u login-ID-list

- *Các thư mục hàng chờ*

Nằm ở dưới `/var/spool/lp/request`, tương ứng mỗi máy in có một thư mục hàng chờ.

XI. 2. Thêm máy in mới vào hệ thống

Trong các file cơ sở dữ liệu máy in như `printcap`, `terminfo`, mỗi máy in đều được đưa vào với một bộ dữ liệu vào mô tả chức năng và các thông số, như ta thấy trong ví dụ sau cho máy in dòng và máy in laser (dấu “\” lại được dùng để chỉ việc viết tiếp dòng):

```
# Line printer
lp | May in dong:                                \
:sd=/var/spool/lp:if=/usr/lib/lpf                \
:lf=/var/adm/lpd-errs:lo=lp0LOCK                 \
:lp=/dev/lp:pl=66:pw=132:af=/var/adm/lpacct
#
# Laser printer
ps | May in Laser tang hai:                      \
:sd=/var/spool/ps:br#9600:rw:sh:lo=lock:mx#10000: \
:af=/var/adm/ps_acct/lp3:lf=/var/spool/ps/log:    \
:lp=/dev/null:cf=/usr/lib/pscif:df=/usr/lib/psdvi:\
:gf=/usr/lib/psgraph:if=/usr/lib/pspf:           \
:nf=/usr/lib/ips:rf=/usr/lib/psfort:              \
:tf=/usr/lib/pscat:vf=/usr/lib/ps:
```

Các ký hiệu được dùng trong file cơ sở dữ liệu máy in: `af` - thư mục file tài khoản in; `br` - baud rate (tốc độ); `lf` - file log ghi lỗi; `lo` - tên file khóa; `lp` - tên file thiết bị (mặc định: `/dev/lp`, cổng nối tiếp: `/dev/ttyx`, ...); `mx` - kích thước cực đại của file (0 - không giới hạn); `pl` - chiều dài của trang in; `pw` - chiều rộng trang in; `sd` - thư mục hàng chờ; `sb` - nén trang tiêu đề.

XI. 2. a. Trình tự thiết lập và loại bỏ máy in tại chỗ

- *Thiết lập máy in tại chỗ*
- Nối các cáp điện và tín hiệu vào máy. Nếu nối qua cổng nối tiếp, trước hết phải xác định tốc độ và chế độ truyền dữ liệu. Cáp của modem là chọn lựa tốt để nối với máy tính trong trường hợp này.

- Đổi chủ quyền và mode của file thiết bị (ví dụ, /dev/lp) tới các người dùng lp và 600.
- Đối với máy nối qua cổng nối tiếp, đặt *off* cho trường mức chạy trong dòng đầu vào của cổng nối tiếp tương ứng tại /etc/inittab.
- Nếu là máy in đầu tiên trong hệ thống, kiểm tra xem đã có liên kết từ /etc/init.d/lp tới các file S và K trong các thư mục khởi động con (thường là /etc/rc2.d và /etc/rc0.d). Nếu chưa có, ta phải thêm vào, ví dụ:

```
# ln /etc/init.d /etc/rc0.d/K25lp
# ln /etc/init.d /etc/rc2.d/K85lp
# ln /etc/init.d /etc/rc2.d/K85lp
```

- Tắt dịch vụ in bằng lệnh **lpshut**. Thêm máy in mới vào hệ thống bằng lệnh **lpadmin**. Nếu là máy in đầu tiên, phải **su** tới người dùng lp trước khi chạy lệnh **lpadmin**. Ví dụ, đối với máy in cổng song song, ta phải thực hiện lệnh:

```
# lpshut
# lpadmin -p printer_name -v /dev/ppx
# lpadmin -p printer_name -I content_type
# lpadmin -p printer_name -t printer_type
```

- Chú ý: đối với máy in PostScript ta phải thực hiện hai thay đổi. Trước hết, trong lệnh **lpadmin** ta phải đặt các giá trị của `content_name` và `printer_type` là PS. Sau đó, đăng ký các lọc PostScript:

```
# cd /etc/lp/fd
# lpfilter -f <filter> -F <filter.fd>
```

Một vài tên của các bộ lọc Postscript như sau: download, dpost, postdaisy, postdmd, postio, postior, postplot, postprint, postreverse, posttek.

- Bắt đầu chạy máy in và hàng chờ của nó. Ví dụ:

```
# accept printer_name
# enable printer_name
```

- In kiểm tra lại.

- *Loại bỏ máy in tại chỗ*

Để loại bỏ máy in tại chỗ khỏi dịch vụ in, ta thực hiện các bước sau:

- Làm treo hàng chờ để khỏi nhận thêm các yêu cầu in mới:

```
# reject -r "printer printer_name is down"
printer_name
```

với r chỉ nguyên nhân loại bỏ máy in. Nguyên nhân này sẽ được hiển thị lại sau này bằng lệnh **lpstat**.

- Ngừng in do tắt máy in bằng lệnh:

```
# disable -W -r "printer printer_name is down"
printer_name
```

với một vài tùy chọn của lệnh **disable** là:

- W**
Chờ cho đến khi yêu cầu in hiện thời được thực hiện xong.
- c**
Ngừng ngay yêu cầu in hiện thời.
- r**
Nguyên nhân loại bỏ máy in
- Loại máy in ra khỏi dịch vụ in:

```
# lpadmin -x printer_name
```

XI. 2. b. Thiết lập máy in từ xa tới máy chủ tại chỗ

▪ Từ client BSD tới máy chủ Solaris 2.x

- Đăng ký dịch vụ:

```
# lpsystem -t bsd print_server_name
```

Lệnh **lpsystem** thêm dòng đầu vào trong file `/etc/lp/Systems.s5 - System V, bsd - BSD`.

- Tạo phiên điều khiển cổng **listen** để điều khiển mạng đối với các yêu cầu in:

```
# sacadm -a -p tcp -t listen -c "/usr/lib/saf/listen
tcp" -v 'nlsadmin -V'
```

- Lấy địa chỉ tổng hợp của máy chủ in ở dạng hexa (4 chữ số đầu - họ giao thức internet; 4 chữ số tiếp - hoặc 0203 hoặc 0ACE, cổng in BSD (515 -> 203₁₆) hay cổng listen của System V (2766 -> 0ACE₁₆); 8 chữ số tiếp - địa chỉ IP dạng hexa của máy chủ - thay bằng các số 0 ở các bản Solaris sau; cuối cùng là 16 chữ số 0). Ví dụ, ta lấy địa chỉ sau 00202038092740400000000000000000.

```
# lpsystem -A 00202038092740400000000000000000
```

- Cấu hình dịch vụ listenBSD:

```
# pmadm -a -p tcp -s lpd -i root -v 'nlsadmin -V' -m
'nlsadmin -o /var/spool/lp/fifos/listenBSD -A
"\00202038092740400000000000000000"'
```

▪ Từ client System V (Solaris) tới máy chủ Solaris 2.x

- Đăng ký dịch vụ:

```
# lpsystem -t s5 print_server_name
```

- ```
lpadmin -p printer -s print server name
```

- Xác định loại máy in:

```
lpadmin -p printer -T unknown -I any
```

- Khởi động máy in:

```
accept printer_name
```

```
enable printer_name
```

- Xác định tên máy in mặc định (tùy chọn):

```
lpadmin -d printer_name
```

#### ▪ *Tìm Client từ máy chủ S5*

- Xác định máy chủ in S5:

```
lpsystem -t s5 print_server_name
```

- Xác định tên máy in:

```
lpadmin -p printer -s print_server_name
```

- Xác định loại máy in:

```
lpadmin -p printer -T unknown -I any
```

- Khởi động máy in:

```
accept printer_name
```

```
enable printer_name
```

- Xác định tên máy in mặc định (tùy chọn):

```
lpadmin -d printer_name
```

## XII. Bảo mật

Trong các chương trước chúng ta đã làm quen với các cách đăng nhập hệ thống, cách trở thành superuser, cũng như sử dụng, sửa đổi các file `/etc/passwd`, `/etc/group`, `etc/shadow`, `/etc/.rhosts`, v.v.. và xử lý chủ quyền, sở hữu và mode của file và thư mục. Trong chương này, chúng ta sẽ làm quen với việc quản lý bảo mật ở quy mô toàn hệ thống UNIX.

### XII. 1. Khái niệm về các cấp bảo mật

Trung tâm quốc gia về An toàn máy tính (NCSC) của Mỹ đã công bố bộ các tiêu chuẩn an toàn cho máy tính và thiết bị xử lý thông tin. Các tiêu chuẩn này được chia thành bốn bảng với các cấp độ nhỏ hơn, đã trở thành phổ biến trên thế giới và có nội dung được định nghĩa như sau:

- **D** : Cấp thấp nhất, không có biện pháp bảo mật nào đặc biệt. Ví dụ: MS-DOS.
- **C1** : (*Discretionary Security Protection*). Điều khiển đăng nhập gián đoạn. Hệ thống yêu cầu thực hiện thủ tục login/password và cung cấp các quyền đăng nhập thông qua người dùng, nhóm, v.v.. Ví dụ: các hệ thống UNIX bình thường chúng ta vẫn sử dụng.
- **C2** : (*Controlled Access Protection*). Bảo mật theo kiểm tra và thẩm quyền. Những sự kiện có liên quan tới bảo mật được kiểm tra, thủ tục login/password cung cấp thẩm quyền và các mật khẩu đã mã hóa được cất vào nơi người dùng bình thường không thể truy nhập được. Ví dụ: các hệ thống UNIX gần đây có cấp C1 được cải tiến tới gần nhưng vẫn chưa đạt C2 (SunOS 4.x), nếu không được cài đặt với tùy chọn Security, hoặc không chạy lệnh **C2conv**. Windows NT/2000 cũng được quảng cáo đạt cấp C2.
- **B1** : (*Labeled Security Protection*). Bảo mật điều khiển quyền truy nhập theo giấy phép và nhân hóa đầu xuất. Là mức bảo mật đầu tiên hỗ trợ các

mức con - *secret* và *top secret*. Người sở hữu file không thể thay đổi chủ quyền file nếu nó nằm dưới điều khiển của giấy phép truy nhập.

- **B2** : (*Structured Protection*). Bảo mật cấu trúc hóa. Mọi đối tượng đều được nhân hóa. Các thiết bị đều được gán nhiều mức bảo mật. Các chức năng quản lý, bảo mật, vận hành đều riêng biệt.
- **B3** : (*Security Domain level*). Mức bảo mật vùng có danh sách điều khiển truy nhập. Việc truy nhập file không những dựa trên cơ sở nhân, mà còn vào danh sách các người dùng có hay không có quyền truy nhập.
- **A1** : (*Verified Design level*). Mức bảo mật kiểm tra thiết kế. Đòi hỏi kiểm tra thiết kế về mặt toán học, phân tích chi tiết các kênh, kiểm tra cá dấu kiểm tra trên phần cứng và phần mềm trong vận chuyển khi mua. Hiện nay, theo công bố công khai, chưa hệ thống nào trên thế giới đạt được mức này.

## XII. 2. Nâng cấp bảo mật hệ thống tới C2

Việc nâng cấp gồm hai bước chính: chép các file cần thiết vào `/usr/etc` và chuyển đổi sang C2.

### XII. 2. a. Các file cần thiết cho C2

| File         | Chức năng                                                                                                                                                                                  |
|--------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| C2conv       | Shell script. Chuyển các file chuẩn passwd, group thành dạng bảo mật, thiết lập thông tin tài khoản.                                                                                       |
| C2unconv     | Shell script. Chuyển ngược lại C2 → chuẩn.                                                                                                                                                 |
| c2convert    | Tiện ích hỗ trợ cho C2conv.                                                                                                                                                                |
| audit        | Giao diện người dùng tới daemon, cho phép thay đổi kiểm tra người dùng, chọn nơi giữ vết tài khoản                                                                                         |
| praudit      | Lấy vết kiểm tra như đầu vào và biến đổi thành đầu ra có thể đọc được.                                                                                                                     |
| rpc.pwdauthd | Daemon kiểm tra thẩm quyền. Cung cấp các thẩm quyền người dùng và mật mã mà không để lộ mã đã khóa. Được kích hoạt trong <code>/etc/rc.local</code> nếu file mật mã bị chia ra bởi C2conv. |

|             |                                                                                             |
|-------------|---------------------------------------------------------------------------------------------|
| *audit_warn | Shell script. Cảnh báo cho người quản trị khi hệ thống file kiểm tra gần đầy hoặc có sự cố. |
| *auditd     | Daemon tài khoản. Chịu trách nhiệm viết các file vết tài khoản.                             |
| grpadj.h    | Mô tả dạng của file group.adjunct.                                                          |
| pwdadj.h    | Mô tả dạng của file passwd.adjunct.                                                         |
| aevents.h   | Mô tả các sự kiện được kiểm tra hóa.                                                        |
| sys/audit.h | Mô tả dạng của bản ghi tài khoản.                                                           |
| sys/label.h | Mô tả dạng của nhãn bảo mật.                                                                |

### ***XII. 2. b. Chuyển đổi sang C2***

#### ***▪ Trước khi chuyển đổi***

- Phải chép các file /etc/passwd và /etc/group thành /etc/passwd.bak và /etc/group.bak.
- Hoạch định chính sách kiểm tra đối với người dùng.

#### ***▪ Chuyển đổi***

Bằng cách chạy **C2conv**, và **c2convert**. Các bước như sau:

- Chuyển về chế độ một người dùng.
- Chuyển các mật khẩu đã mã hóa từ /etc/passwd thành /etc/security/passwd.adjunct và /etc/group thành /etc/security/group.adjunct.
- Thiết lập các thư mục cho các file vết kiểm tra (cho máy tại chỗ và từ xa), nếu cần, tạo và mount các hệ thống file này (file audit\_control trong thư mục /etc/security/audit).
- Xây dựng danh sách các sự kiện cần kiểm tra hóa. Các cờ là: dr - đọc dữ liệu; dw : ghi dữ liệu; dc : tạo dữ liệu; da : thay đổi truy cập dữ liệu; lo - login-logout; ad : quản trị; p0 : ưu tiên thấp; p1 : ưu tiên cao.
- Đặt giá trị tối thiểu không gian trống cho cảnh báo của hệ thống file kiểm tra (bằng biến minfree). Xác định rằng ai được thông báo khi đạt tới giá trị minfree, tức là khi một trong những hệ thống file kiểm tra bị đầy.



- Đặt mật khẩu cho người dùng `audit`.
- Sau khi chạy xong **C2conv**, khởi động lại máy ở chế độ nhiều người dùng, kiểm tra xem daemon đã chạy chưa:

```
<hostname># ps aux | grep auditd
```

## XII. 3. Quản lý tài khoản người dùng

### XII. 3. a. Thêm bớt các tài khoản và nhóm người dùng

Việc quản lý danh sách người dùng trong hệ thống là một trong những nhiệm vụ thường xuyên của người quản trị. Để tạo thêm một tài khoản người dùng mới trong hệ thống, người quản trị có thể dùng vài cách sau. Trước hết, ngay trong chế độ dòng lệnh, người quản trị dùng lệnh **useradd** theo cú pháp sau:

```
useradd [-c comment] [-d dir] [-e expire] [-f inactive]
 [-g group] [-G group[, group...]] [-m [-k skel_dir]]
 [-u uid [-o]] [-s shell] login
useradd -D [-b base_dir] [-e expire] [-f inactive] [-g
 group]
```

Lệnh **useradd** thêm dòng vào về người dùng vào các file `/etc/passwd` và `/etc/shadow`. Tuy nhiên, tài khoản người dùng mới chỉ thực sự có hiệu lực khi người dùng đó (hoặc `root`) đặt mật khẩu bằng lệnh **passwd**. `login` là xâu ký tự chỉ tên tài khoản người dùng, được dùng như tên truy nhập hệ thống và có độ dài cho đến tám ký tự, không chứa các dấu hai chấm ":" và xuống dòng "\n". Các tùy chọn của lệnh **useradd** có ý nghĩa như sau:

**-c** *comment*

Xâu ký tự bất kỳ nằm trong trường thứ năm của file `/etc/passwd`. Thường là mô tả ngắn về người dùng, hay họ tên đầy đủ của người dùng.

**-d** *dir*

Thư mục `home` của người dùng.

**-e** *expire*

Thời hạn hiệu lực của tài khoản người dùng, tính đến ngày được xác định bằng giá trị `expire`. Ngày giờ có thể được đưa vào ở dạng ngày tháng hợp lệ bất kỳ, nhưng nếu có khoảng trống thì dấu chấm phẩy chỉ ngày tháng phải được để trong dấu ngoặc kép. Sau ngày đó, tài khoản người dùng sẽ bị mất hiệu lực. Nếu `expire` có giá trị `null` (""), việc đặt thời hạn hiệu lực của người dùng bị xóa bỏ.

**-f** *inactive*

Số ngày cực đại cho phép giữa các lần dùng tên tài khoản người dùng trước khi tên tài khoản người dùng đó bị coi là không hợp lệ. Thường là số nguyên dương.

**-g** *group*

Là dấu ký tự chỉ tên nhóm hay số nguyên định danh nhóm. Nếu không có tùy chọn `-D`, sẽ xác định nhóm sơ cấp của người dùng và là nhóm mặc định. Có thể đặt lại nhóm mặc định bằng cách gọi lệnh **useradd** `-D -g group`.

**-G** *group*

Là dấu ký tự chỉ tên nhóm hay số nguyên định danh nhóm. Dùng để xác định sự tham gia của người dùng vào trong nhóm thứ cấp mới. Tên nhóm bị chỉ trùng giữa các tùy chọn `-g` và `-G` bị bỏ qua. Số nhóm thứ cấp cực đại không được vượt quá giá trị của biến `NTROUUPS_MAX`.

**-k** *skel\_dir*

Thư mục đã tồn tại trong hệ thống và chứa các thông tin cốt lõi, ví dụ file `.profile`, có thể được sao chép vào thư mục home của người dùng. Hệ thống cung cấp thư mục `/etc/skel` cho mục đích này.

**-m**

Tạo thư mục home mới cho người dùng. Nếu thư mục đã tồn tại, nó phải có các quyền `r, w, x` của nhóm sơ cấp của người dùng.

**-s** *shell*

Tên đường dẫn đầy đủ của chương trình thực thi được dùng như shell chạy khi người dùng đăng nhập hệ thống. Nếu bỏ trống, shell mặc định sẽ là `/sbin/sh`.

**-u** *uid*

UID của người dùng mới. Là số nguyên theo cơ số mười, nhỏ hơn giá trị `MAXUID` xác định trong `<sys/param.h>`. Trong trường hợp không chỉ rõ, giá trị mặc định sẽ tăng dần theo một đơn vị từ số UID cao nhất đã có trong hệ thống. Các số từ 0 đến 99 được Sun dành cho hệ thống (0 - UID của `root`).

**-o**

Tùy chọn này cho phép tài khoản người dùng có thể được sao chép (nhân bản).

**-D**

Hiển thị các giá trị mặc định của `group`, `base_dir`, `skel_dir`, `shell`, `inactive`, `expire`. Nếu dùng cùng với các tùy chọn `-g`, `-b`, `-f`, `-e`, tùy chọn `-D` sẽ đặt lại các giá trị mặc định tương ứng.

**-b base\_dir**

Thư mục cơ sở, nếu không chỉ ra `-d dir`.

Cách thứ hai cho phép ta thực hiện các thao tác quản lý danh sách các tài khoản người dùng và nhóm người dùng thông qua tiện ích **Admintool** trong giao diện đồ họa X Windows. Trong cửa sổ của tiện ích **Admintool**, ta có thể chọn các trang **User**, **Group**, hoặc **Ports** để thêm bớt, sửa đổi tài khoản người dùng, nhóm người dùng và cấu hình các cổng nối tiếp của máy tính một cách tương ứng. Các thông số được nêu ra trong các dòng vào, sau khi được chấp nhận sẽ được ghi vào các trường của các file `/etc/passwd` và `/etc/shadow`. Tất nhiên, cả hai cách trên đều phải được thực hiện với quyền của `root` và chỉ tạo được các người dùng bình thường có UID khác 0.

Cách thứ ba được dùng là ta sửa đổi thẳng vào các file `/etc/passwd` và `/etc/shadow` để thêm các dòng vào cho các tài khoản người dùng mới hoặc sửa lại các thông số quy định cho các người dùng đã có đăng ký tài khoản trong hệ thống.

Nếu muốn loại bỏ một tài khoản người dùng nào đó ra khỏi hệ thống, người quản trị có thể thực hiện theo các cách trên. Riêng trong chế độ dòng lệnh, lệnh **userdel** sẽ được dùng với cú pháp sau:

```
userdel [-r] login
```

với tùy chọn `-r` để xóa bỏ luôn thư mục home của người dùng đó. Kết quả trả về sẽ như sau: 0 - thành công; 2 - sai cú pháp, dòng nhắc cú pháp lệnh sẽ được hiển thị; 6 - tên người dùng `login` không tồn tại; 8 - tên người dùng `login` đang được sử dụng; 10 - không thể cập nhật file `/etc/group`, nhưng tên người dùng bị xóa khỏi file `/etc/passwd`; 12 - không thể xóa hay sửa đổi thư mục home của người dùng.

Ngoài ra, người quản trị có thể quy định lại chế độ và quyền sử dụng hệ thống của người dùng đã đăng ký tài khoản trong hệ thống bằng lệnh **usermod** có cú pháp:

```
usermod [-u uid [-o]] [-g group] [-G group [,group...]]
 [-d dir [-m]] [-s shell] [-c comment] [-l new_logname]
 [-f inactive] [-e expire] login
```

Các tùy chọn có ý nghĩa giống như trong lệnh **useradd**, chỉ có **new\_logname** dùng để tạo ra tên đăng nhập mới cho người dùng.

Solaris thường quy định người dùng thuộc về nhóm người dùng chính duy nhất. Để thay đổi nhóm người dùng chính cho một người dùng nào đó, ta dùng lệnh chuyển nhóm sau:

```
newgroup [-] [group]
```

Việc thực hiện lệnh sẽ thay thế shell hiện tại của người dùng bằng shell mới, người dùng vẫn ở trong mạng và thư mục làm việc hiện thời không bị thay đổi. Nếu lệnh **newgroup** được gọi không có thông số, định danh nhóm thực (GID) và hiệu dụng (SGID) của người dùng được lấy trở lại bằng các giá trị quy định trong **/etc/passwd**. Nếu có dấu “-” trong dòng lệnh **newgroup**, môi trường người dùng sẽ bị thay đổi đến các giá trị của nhóm mới mà người dùng sẽ thuộc về khi đăng nhập lại vào hệ thống.

Việc tham gia đồng thời của người dùng vào các nhóm người dùng được quy định khác nhau trong các bản UNIX. Một số bản UNIX – như được trình bày về Solaris ở trên chẳng hạn – cho phép người dùng trở thành thành viên chính thức của một nhóm người dùng duy nhất, các nhóm người dùng khác mà người dùng đó tham gia đồng thời chỉ là các nhóm thứ cấp. Nhưng trong BSD UNIX, người dùng có thể tham gia đồng thời cho tới 128 nhóm người dùng ngang hàng.

### ***XII. 3. b. Đặc điểm quản lý tài khoản người dùng trong hệ Solaris***

Gói phần mềm quản lý tài khoản người dùng (hay còn gọi là **audit**) được sử dụng trong Solaris chính là thành phần của gói phần mềm BSM (Basic Security

Module - Môđun Bảo mật cơ bản). Daemon **auditd** là tiến trình thực hiện nhiệm vụ kiểm toán đó trong các hệ điều hành Solaris và được khởi động một cách mặc định, với điều kiện file `/etc/security/audit_startup` có tồn tại. Danh mục các hoạt động bị kiểm toán được xác định trong file `/etc/security/audit_event`, và người ta có thể thay đổi nội dung các dòng vào của file trên, nhưng đây là một việc đòi hỏi một trình độ hiểu biết tốt về hệ thống. **answerbook** trong Solaris mô tả rất tốt toàn bộ tiến trình và các file có liên quan, do đó, người mới bắt đầu quản trị hệ thống trước hết nên tìm đọc (nhắc lại, **answerbook** là tiện ích trợ giúp có sẵn, hoặc được cài đặt bổ sung trong môi trường đồ họa của Solaris).

Các cờ kiểm toán chỉ cho ta biết những loại sự kiện nào được kiểm toán. Những sự kiện mặc định có tính chất toàn hệ thống được liệt kê trong file cơ sở cực kỳ quan trọng `/etc/security/audit_control`, tương tự như việc thiết lập "User Manager->Policies->Audit" trong Windows NT/2000. Nếu thiết lập không đúng, ta sẽ có hoặc quá ít, hoặc quá nhiều thông tin. Có thể tìm hiểu về các loại thông tin để chọn lựa qua trang **man** về `audit_classes` (4). Thoạt tiên, ta có thể tham khảo file ví dụ `audit_control` với nội dung như sau:

```
dir:/etc/security/audit
flags:lo,ad,-fm,-fc,-nt
naflags:lo,ad
minfree:10
```

Ta hãy tìm hiểu nội dung trên có ý nghĩa gì. Dòng đầu tiên xác định thư mục định vị các file kiểm toán, cụ thể là `/etc/security/audit`. Phải đảm bảo cho thư mục này được dành **đủ** dung lượng đĩa, nếu không, hệ thống sẽ bị khóa đối với tất cả các người dùng, kể cả root. Có thể chỉ ra nhiều hơn một vị trí bằng cách dùng thêm các cờ `dir` đối với các mục đích đặc biệt. Dòng thứ hai dùng liệt kê các sự kiện thuộc về người dùng, và xác định cho chúng ta rằng:

```
lo -> tất cả các sự kiện login và logout
ad -> tất cả các hoạt động quản trị
-fm -> tất cả các thay đổi thuộc tính của đối tượng (chmod, flock,
```

v.v..), nhưng không thành công.

- fc -> tất cả các việc tạo đối tượng không thành công.
- fd -> tất cả các thao tác xóa file.
- nt -> tất cả các sự kiện không thành công về mạng.
- pc -> tất cả các thao tác đối với tiến trình: **fork**, **exec**, **exit**,  
v.v..

Người đọc nên xem nội dung của file `audit_event` để biết danh sách các sự kiện được phân thành các lớp ra sao.

Dòng thứ ba dành mô tả các sự kiện không thuộc về một người dùng cụ thể nào cả. Dòng thứ tư xác định dung lượng trống tối thiểu còn lại dành cho cờ `dir`, khi hệ thống kiểm toán sẽ phát ra cảnh báo. Lệnh `audit -s` sẽ bắt daemon `auditd` đọc lại file `audit_control` sau khi được sửa đổi.

Một file khác cần quan tâm tới là `/etc/security/audit_user`, cho phép thực hiện kiểm toán đặc biệt đối với từng người dùng riêng rẽ. Nếu file này có dòng vào không trống, các cờ được xác định sẽ kết hợp với các cờ chung trong file `audit_control` để cung cấp các khả năng kiểm toán với các mức độ phân biệt khác nhau.

Các bước cụ thể để thực hiện cấu hình kiểm toán cho các hệ điều hành UNIX về cơ bản tương tự như nhau. Ta xét một trình tự cấu hình việc kiểm toán trong ví dụ sau:

1. Kích hoạt Basic Security Module (BSM):

```
/etc/security/bsmconv
```

2. Cấu hình các lớp sự kiện trong file `/etc/security/audit_control`:

```
dir:/var/audit
flags:lo,ad,pc,fc,fd,fm
naflags:lo,ad
minfree:10
```

3. Tạo shell script `/etc/security/newauditlog.sh`:

---

```
#!/sbin/sh
newauditlog.sh
- Bắt đầu file audit mới và kết thúc file cũ
#
AUDIT_EXPIRE=30
AUDIT_DIR="/var/audit"

/usr/sbin/audit -n

cd $AUDIT_DIR
trong trường hợp là liên kết
/usr/bin/find . $AUDIT_DIR -type f -mtime \
 +$AUDIT_EXPIRE -exec rm {} > /dev/null 2>&1 \;
```

4. Chạy shell script bạn đặt từ **cron**:

```
/usr/bin/crontab -e root
0 0 * * * /etc/security/newauditlog.sh
```

Các file audit không thể đọc được, ta phải dùng lệnh **praudit** (xem trang trợ giúp **man**) để biến đổi dữ liệu của chúng sang dạng ASCII.

### ***XII. 3. c. Chống đăng nhập trái phép đến hệ thống và mạng Solaris***

Như đã nói ở trên, các hệ UNIX nói chung, và Solaris nói riêng đều có khả năng ghi nhận lại tất cả các hoạt động của người dùng, hệ thống và các tiến trình trong các file log. Do đó, mọi cố gắng đăng nhập trái phép hệ thống và mạng UNIX đều có thể bị phát hiện thông qua các file log trên. Với tư cách người quản trị mạng, bạn hãy kiểm tra mỗi loại file log vào thời điểm được xác định trước đối với chúng. Tốt nhất là nên kiểm tra chúng hàng ngày. Hãy ghi lại mọi dòng vào bất thường mà bạn phát hiện được, trước hết phải kiểm tra việc đăng nhập hệ thống của các người dùng bình thường và các người dùng được ưu tiên.

Bản cài đặt chung của Solaris 2.x trở đi có chứa file **syslog** làm nhiệm vụ gửi các thông tin log quan trọng nhất tới file **/var/adm/messages**; chỉ có các thông tin **mail.debug** được gửi tới file **/var/adm/syslog**. Cấu hình các

thông tin đó được quy định trong file `syslog.conf`. Dưới đây ta sẽ xem xét các loại file log và cách xử lý với chúng.

- *Hạn chế truy nhập đến các file log của kiểm toán và các file log khác*

Phải kiểm tra để tin chắc rằng chỉ có `root` mới có quyền truy nhập ghi vào các file kiểm toán sau :

❖ `/var/log/authlog`

❖ `var/log/syslog`

Theo mặc định, file `syslog` chỉ chứa các thông báo về thư điện tử, như được xác định bởi file `/etc/syslog.conf`. Mọi thông tin khác loại, nếu không được cấu hình trước, đều coi là bất thường.

❖ `/var/adm/messages*`

Đây là bản ghi log của kết quả ra trên console hệ thống và các thông báo `syslog` (có thể có nhiều file với tên bắt đầu như trên). Trong trường hợp có sự đe dọa xâm phạm hệ thống máy tính ta phải tìm các dấu hiệu đáng nghi ngờ. Trước hết, tìm những lần treo máy bất thường với dòng vào có dạng:

```
Mar 31 12:48:41 <hostname> unix: halted by <userid>
```

Tìm những lần khởi động lại bất thường:

```
Jun 23 10:45:11 <host.domain.com> unix: rebooting...
```

Tìm những lần thực hiện lệnh `su` và lệnh `login` không thành công:

```
Mar 30 09:14:00 <hostname> login: 4 LOGIN FAILURES ON
0, <userid>
```

```
Mar 31 12:37:43 <hostname> su: 'su root' failed for
<userid> on /dev/pts/??
```

Tìm những lần thực hiện lệnh `su` thành công:



```
Mar 28 14:31:11 <hostname> su: 'su root' succeeded for
 <userid> on /dev/console
```

❖ /var/log/sulog, /var/adm/sulog

Là nơi chứa các bản ghi log của lệnh **su**. Ta phải tìm những dòng vào khi một người dùng không được phép nào đó đã lạm dụng lệnh **su**:

```
SU 03/31 12:52 - pts/0 <userid>-root
```

Dòng vào hợp lệ ghi lần thực hiện thành công (ký hiệu bằng dấu "+") lệnh **su** của người dùng **userid** thành **root** có dạng sau:

```
SU 03/31 12:52 + pts/0 <userid>-root
```

Tìm dòng vào thực hiện lệnh **su** vào thời điểm bất thường như sau giờ làm việc, quá sớm, v.v...:

```
SU 03/31 02:12 + pts/0 <userid>-root
```

❖ /var/log/utmp, /var/adm/utmp;  
/var/log/utmpx, /var/adm/utmpx

Các file này chứa vết lưu lại của người dùng hiện thời đăng nhập hệ thống. Ta dùng lệnh **who**, để kiểm tra ngày giờ vào hệ thống của các người dùng đăng nhập hệ thống ở thời điểm hiện thời:

```
root <userid> pts/1 Mar 31 08:40 (origination hostname)
```

Hãy tìm những lần đăng nhập hệ thống bất thường, như của người đang đi nghỉ phép, hay vào thời điểm bất thường, hoặc từ những vị trí địa lý bất thường.

❖ /var/log/wtmp, /var/adm/wtmp;  
/var/log/wtmpx, /var/adm/wtmpx

Các file log trên giữ vết login và logout (vào và ra khỏi hệ thống). Dùng lệnh **last**, bạn hãy thực hiện những thao tác sau đây:

Tìm những lần đăng nhập hệ thống vào thời điểm bất thường, theo các dòng vào như sau:

```
<userid> pts/4 <hostname> Sat Mar 22 03:14 - 06:02
(02:47)
```

Tìm những lần đăng nhập hệ thống từ những vị trí bất thường (vị trí địa lý, địa chỉ hoặc thiết bị):

```
<userid> pts/12 <strange hostname> Fri Mar 21 08:59 -
13:30 (04:31)
```

Tìm các lần khởi động lại máy một cách bất thường:

```
reboot system boot Sun Mar 23 05:36
```

#### ❖ /var/adm/pacct

Đây là file log ghi lại các lệnh được thực hiện bởi tất cả các người dùng, với điều kiện việc thống kê các tiến trình phải được mở trước khi ghi file log này. Bạn có thể dùng lệnh **lastcomm** để kiểm lại các lệnh đã được chạy bởi người dùng cụ thể nào đó trong khoảng thời gian xác định theo các dòng vào như sau:

```
compile <userid> ttyp1 0.35 secs Mon Mar 31 12:59
```

#### ❖ /var/adm/aculog

File log giữ vết của các lần quay số của các modem. Hãy tìm các bản ghi các cuộc quay số mâu thuẫn với quy định sử dụng các modem, cũng như việc lạm dụng các modem khi không được phép.

Chỉ có người sở hữu của tiến trình **httpd** mới có quyền ghi vào file log truy nhập tới **httpd**.

- *Ghi lại Log tất cả các hoạt động su*

Các thông tin su login có thể được ghi lại vào file log tại chỗ, hay vào trong máy chủ đặc trách cho ghi log (loghost). Chú ý: Phải dùng phím Tab để đưa vào ký hiệu khoảng trống trong file syslog.conf.

Thao tác cấu hình ghi vào file log tại chỗ: Ta sửa file /etc/syslog.conf và thêm vào dòng vào sau:

```
auth.notice /var/log/authlog
```

Thao tác cấu hình ghi vào file log tại máy chủ ghi log: Ta sửa file /etc/syslog.conf và thêm vào dòng vào sau:

```
auth.warning ifdef (`LOGHOST`, /var/log/authlog,
@loghost)
```

Thực hiện các lệnh sau để ghi log lại các lần cố gắng vào hệ thống không thành công của người dùng vào file log tại chỗ:

```
touch /var/adm/loginlog
chown root /var/adm/loginlog
chgrp sys /var/adm/loginlog
chmod 600 /var/adm/loginlog
```

- *Ghi log các kết nối đến của các dịch vụ TCP*

Nếu không dùng tcp-wrapper (gói phần mềm dùng theo dõi và lọc các yêu cầu đến đối với các lệnh **sysstat**, **finger**, **ftp**, **telnet**, **rlogin**, **rsh**, **exec**, **tftp**, **talk** và các dịch vụ mạng khác), bạn có thể cấu hình hệ thống để ghi log lại các hoạt động TCP (bao gồm ngày giờ kết nối, địa chỉ IP của máy muốn kết nối và tên dịch vụ TCP) bằng cách đưa vào các thay đổi như sau. Chú ý: Phải dùng phím Tab để đưa vào ký hiệu khoảng trống trong file syslog.conf. Ta sửa file /etc/syslog.conf và thêm vào dòng vào sau:

```
daemon.notice /var/log/syslog
```

Thay đổi dòng vào sau trong file `/etc/rc2.d/S72inetdsvc:`

```
/usr/sbin/inetd -s
```

để đọc:

```
/usr/sbin//inetd -s -t
```

### ***XII. 3. d. Các bước tăng cường bảo mật máy chủ Solaris***

Dưới đây là mô tả cụ thể các bước tổng quát để tăng cường bảo mật cho máy tính UNIX. Mặc dù ví dụ được đưa ra cho hệ Solaris 2.5, nhưng quy trình này có thể áp dụng cho hầu hết các hệ UNIX khác. Đối với các hệ UNIX khác với Solaris, trong khi thao tác cần chú ý tới đặc thù của hệ thống cụ thể.

1. Áp dụng tất cả các khuyến nghị và các bổ sung bảo mật được tải về từ trang Web của SunSolve (<http://sunsolve.sun.com>).

2. Tắt hết các dịch vụ không cần thiết trong `/etc/inetd.conf`. Một quy tắc có ích là nếu bạn không biết dòng vào cụ thể nào dùng làm gì thì hãy tắt nó đi bằng cách biến nó thành chú thích. Như vậy, bạn sẽ chỉ mở những dịch vụ mà bạn hiểu rõ chức năng của chúng.

3. Thêm khóa `-l` vào dòng vào `in.ftpd` trong file `/etc/inetd.conf` để ghi log cho các lệnh **ftp**.

4. Tạo file bằng lệnh **touch**:

```
touch /etc/notrouter
```

5. Tạo file:

```
touch /etc/defaultrouter
```

6. Đặt thành chú thích (thêm dấu “#” ở đầu dòng) mọi dòng vào có tham chiếu đến “action\_filemgr.so” trong file /etc/mmount.conf.

7. Thực hiện lệnh:

```
chmod o-rx /bin/su /sbin/su.static
```

8. Thực hiện lệnh:

```
chgrp sysadmin /bin/su /sbin/su.static
```

9. Thực hiện lệnh:

```
chmod 644 /etc/hostname.*
```

10. Thực hiện lệnh:

```
find /var/statmon -exec chmod o-w {} \;
```

11. Phân tích các file được định vị bằng lệnh:

```
find / -user root \(-type d -o -type f \) -perm -2 -ls
```

12. Thực hiện lệnh:

```
find /opt/SUNWdxlib -exec chmod go-w {} \;
```

13. Thực hiện lệnh:

```
chmod 755 /etc /usr
```

14. Thực hiện lệnh:

```
chmod 700 /etc/security
```

15. Thực hiện lệnh:

---

```
chmod o-rw /dev/rmt/*
```

16. Thêm các dòng vào sau vào file /etc/init.d/inetinit:

```
#####
Turn off all IP forwarding
#####
nnd -set /dev/ip ip_forwarding 0
nnd -set /dev/ip ip_forward_src_routed 0
nnd -set /dev/ip ip_forward_directed_broadcasts 0
nnd -set /dev/ip ip_respond_to_timestamp 0
nnd -set /dev/ip ip_respond_to_timestamp_broadcast 0
nnd -set /dev/ip ip_ignore_redirect 1
#####
Protect against SYN flood
#####
nnd -set /dev/tcp tcp_ip_abort_cinterval 10000
echo "tcp_param_arr+14/W 0t10240" |adb -kw /dev/ksyms \
 /dev/mem

Solaris 2.5 and 2.5.1 w/o patch 103582-12
nnd -set /dev/tcp tcp_conn_req_max 8192
Solaris 2.5.1 w/ patch 103582-12 and 2.6
nnd -set /dev/tcp tcp_conn_req_max_q 1024
nnd -set /dev/tcp tcp_conn_req_max_q0 8192
#####
Protect against "Ping Of Death"
#####
nnd -set /dev/ip ip_respond_to_echo_broadcast 0
#####
Tweak I/O performance, especially 100Base-T
#####
nnd -set /dev/tcp tcp_xmit_hiwat 65536
nnd -set /dev/tcp tcp_recv_hiwat 65536
```

```
ndd -set /dev/udp udp_xmit_hiwat 65536
ndd -set /dev/udp udp_recv_hiwat 65536
ndd -set /dev/tcp tcp_cwnd_max 65535
```

17. Sửa file `/etc/init.d/rpc` và chỉ để lại dòng vào `rpcbind`:

```
set ` /usr/bin/who -r `
if [$9 = "S" -o $9 = "1" -o $9 = "?"]
then
 if [-x /usr/sbin/rpcbind]
 then
 #/sbin/sh /etc/init.d/rpc rpcstart
 echo "starting rpc services:\c"
 /usr/sbin/rpcbind > /dev/console 2>&1
 echo " rpcbind"
 else
 exit # nothing else can do anything
 fi
fi
;;
```

18. Sửa file `/etc/init.d/inetsvc`. Thay `"/usr/sbin/inetd -s"` bằng `"/usr/sbin/inetd -st"` để kích hoạt việc để lại vết của `inetd`.

19. Tạo file `/etc/ftpusers` với các dòng vào sau:

```
root
daemon
bin
sys
adm
lp
smtp
nobody
```

```
nobody4
uucp
nuucp
listen
```

20. Làm thủ động hóa tất cả các dòng vào không cần thiết trong các thư mục `/etc/rc*.d`, như:

```
S47asppp S73nfs.client S76nscd S80lp
S88sendmail S89bdconfig S15nfs.server
```

Đối với Solaris 2.6 trở đi, thêm cả:

```
S74xntpd S80spc S85power S76snmpdx
S77dmi
```

21. Kiểm tra tất cả các kết nối listen đang được mở bằng lệnh:

```
netstat -a | grep LISTEN
```

22. Đảm bảo rằng tất cả các hệ thống file được đưa tới đầu ra đều là chỉ-đọc được với lệnh **share**. Dùng tùy chọn `"-o ro"` trong file `/etc/dfs/dfstab`.

23. Mount tất cả các hệ thống file khác với các thư mục hệ thống cơ sở máy của bạn với tùy chọn `nosuid` trong file `/etc/vfstab`.

24. Loại bỏ tất cả các file `.rhost`.

25. Luôn luôn dùng **xhost -**. Đặt lệnh đó như là lệnh hệ thống (thứ trong file `/usr/dt/config` hoặc `/usr/openwin/lib/xdm`).

26. Không chạy NIS, hay ít nhất phải đảm bảo đã tăng cường bảo mật cho máy chủ, và bắt các client chạy với lệnh **ypbind -ypsetme**. Nếu bạn phải chạy dịch vụ thông tin mạng, hãy dùng NIS+ có kích hoạt bảo mật DES.

27. Cài đặt các tiện ích bảo mật khác như COPS và Tripwire. Chúng có thể tải về được từ site COAST (<http://www.cs.purdue.edu/coast/>) tại Purdue University.



## XII. 4. FireWall

### *XII. 4. a. Khái niệm chung*

FireWall - Bức tường lửa, là một hệ thống bảo mật linh hoạt, được thiết kế để ngăn chặn các truy nhập trái phép từ Internet và có thể hoạt động được trên nền các hệ điều hành khác nhau. Chính sách sử dụng FireWall được hoạch định trên cơ sở các đối tượng mạng, các dịch vụ, tài khoản người dùng và các quy tắc quan hệ giữa chúng. Các bản FireWall do nhiều hãng cung cấp, ví dụ như Raptor, Checkpoint (Firewall-1), SunScreen, v.v.. đối với Solaris. Ta hãy xét kiến trúc của FireWall-1 sau đây như ví dụ minh họa.

- *Kiến trúc của FireWall*

Nói chung, FireWall được xây dựng trên bốn thành phần chính:

#### Thành phần quản lý

Người quản trị giao tiếp với thành phần quản lý (*Management Component*) qua giao diện đồ họa của nó để điều khiển cơ sở dữ liệu FireWall: Quy tắc cơ bản, đối tượng mạng, các dịch vụ, tài khoản người dùng, v.v..

#### Mô đun kiểm tra

Mô đun kiểm tra FireWall (*FireWall Inspection Module*) gồm hai daemon là snmpd và fwd, có thể được cài đặt trên cổng (gateway) cũng như trên server. FireWall được nạp vào kernel hệ thống, nó bắt được các gói thông tin trước khi chúng được gửi đi nên không cần thiết phải dừng việc phân đường, các liên lạc không bị ngắt đoạn.

Mô đun kiểm tra FireWall là thành tố quan trọng nhất. Khi được cài trên gateway, là cấu hình phổ biến nhất, mô đun này giám sát việc lưu chuyển thông tin giữa các mạng. Khi nói mô đun kiểm tra được nạp vào kernel hệ thống, nghĩa là nó nằm giữa lớp Liên kết dữ liệu (đối tượng vật lý là card mạng) và lớp Mạng (lớp giao thức đầu tiên) (xem Mô hình mạng OSI-ISO 7 lớp trong phần "*Giao thức TCP/IP*") và bắt tất cả các gói đi và đến gateway, xử lý chúng theo các quy tắc được định sẵn.

### Mô đun phân đường

Mô đun phân đường cho phép điều khiển danh sách điều khiển truy nhập của router thông qua giao diện đồ họa.

### Thành phần mã hóa

Thành phần mã hóa thiết lập kênh truyền thông hoàn toàn an toàn trong Internet cho mạng ảo riêng.

Các thành phần này có thể được cài đặt trên cùng một máy hay trên các máy riêng biệt.

#### ▪ *Các yêu cầu về phần cứng và phần mềm đối với việc cài đặt FireWall*

- Máy tính Sun SPARC hay trên nền Intel
- Có cài đặt SunOS™ 4.1.3, U1 hay Solaris 2.3 trở đi
- OpenWindows 3 trên máy định cài Thành phần quản lý và màn hình (các máy Sun SPARC có thể chạy không cần tới màn hình và bàn phím, chuột).
- Địa hệ thống tại chỗ có dung lượng trống không nhỏ hơn 6 MB
- 16 MB RAM cho Thành phần quản lý
- Ổ đĩa CD
- Các card mạng tương thích phần cứng với Solaris.
- Router Cisco IOS 9.x, 10.x hay Bay Network Wellfleet 8.0 trở lên cho Mô đun phân đường.

### **XII. 4. b. Quy trình cài đặt FireWall**

#### ▪ *Chuẩn bị*

Nếu có nhiều hơn một người quản lý FireWall thì phải tạo nhóm UNIX trong file /etc/group hoặc /etc/netgroup, gồm chỉ những người dùng quản trị. Nhóm này phải được tạo trước khi tiến hành cài đặt FireWall.

Nhớ rằng ta nên chạy C shell.

▪ *Việc đưa CD-ROM là cần đặt*

1. Trở thành superuser:

```
hostname% su
Password: <đưa vào mật khẩu của root>
hostname%
```

2. Đưa đĩa CD-ROM vào ổ CD. Giả sử ổ CD của ta theo chuẩn SCSI, có ID là 6 (Xem lại chương *Bảo trì phần cứng hệ thống*).
3. Mount CD-ROM vào thư mục tại chỗ:
  - Nếu daemon vold (Volume Manager) chạy thì đĩa CD-ROM tự động được mount dưới thư mục /cdrom/firewall1\_x\_0, x chỉ phiên bản, trong quy trình này để xác định, ta lấy x = 2. Phiên bản hiện nay là 4.
  - Nếu vold không chạy, ta tạo lấy thư mục đó và mount bằng tay:

```
hostname% mkdir /cdrom/firewall1_2_0
hostname% mount -r /dev/dsk/c0t6d0s0
 /cdrom/firewall1_2_0
```

**Chú ý:** phải chọn tên file thiết bị cho phù hợp với hệ thống cụ thể của bạn.

▪ *Các thao tác cài đặt đầu tiên*

- Đối với **Solaris x86**

1. Thực hiện lệnh **pkgadd** như sau:

```
hostname% /usr/sbin/pkgadd -d /cdrom/
 firewall1_2_0/i386-2.x
```

2. Trên máy sẽ hiện ra màn hình với các dòng sau:

The following packages are available:

```
1 SUNWfw Solstice FireWall-1 (i386) 2.0
```

Select package(s) you wish to process (or "all" to process all packages). (default: all) (?,?,q):

3. Ấn phím *Enter* để bắt đầu cài đặt.
4. Trả lời các câu hỏi xuất hiện trên dòng lệnh.
- Đối với **Solaris SPARC 2.x**
1. Thực hiện lệnh **pkgadd** như sau:

```
hostname% /usr/sbin/pkgadd -d /cdrom/firewall1_2 \
 _0/sparc-2.x
```

2. Trên máy sẽ hiện ra màn hình với các dòng sau:

The following packages are available:

```
1 SUNWfw Solstice FireWall-1 (sparc) 2.0
```

Select package(s) you wish to process (or "all" to  
process all packages). (default: all) (?,??,q):

3. Ấn phím *Enter* để bắt đầu cài đặt.
4. Trả lời các câu hỏi xuất hiện trên dòng lệnh.
- Đối với **Solaris SPARC 1.x**
1. Thực hiện các lệnh như sau:

```
hostname% cd /cdrom/firewall1_2_0/sparc-1.x
hostname% ./fwinstall
```

2. Trên máy sẽ hiện ra màn hình với các dòng sau:

Please, choose a directory for instalation of the  
Solstice FireWall-1 package. You may enter an  
alternate path, if desired.

Enter the instalation directory (/usr/local/fw):

The directory /usr/local/fw does not exist.  
Should it be created now (y/n) [y]

Installation directory /usr/local/fw created.

3. Ấn phím *Enter* để bắt đầu cài đặt.

4. Trả lời các câu hỏi xuất hiện trên dòng lệnh.

▪ *Các bước cài đặt tiếp theo được áp dụng chung cho các hệ Solaris*

5. Hệ thống hỏi kiểm tra xem có tồn tại sẵn phiên bản trước của FireWall trên máy hay không.

6. Ấn phím *Enter* để chấp nhận giá trị mặc định (n), hay gõ **y** trong trường hợp ngược lại.

7. Ấn phím *Enter* để tiếp tục, hay gõ Ctrl-C để hủy bỏ việc cài đặt.

8. Gõ số chỉ phương án cài đặt mà bạn lựa chọn trên màn hình có thông báo sau:

Solstice FireWall-1 instalation/configuration options

-----  
Choose one of the following options for this  
instalation:

- (1) Solstice FireWall-1      Evaluation Pack
- (2) Solstice FireWall-1      Light Security Center
- (3) Solstice FireWall-1      Medium Security Center
- (4) Solstice FireWall-1      Internet Security Center
- (5) Solstice FireWall-1      Network Security Center
- (6) Solstice FireWall-1      Inspection Module
- (7) Solstice FireWall-1      Authentication Inspection  
                                         Module

Enter your selection (1-7) :

9. Điền vào tên của nhóm quản lý UNIX. Nếu chỉ muốn có root là người quản lý FireWall, chỉ cần ấn phím *Enter*.

10. Gõ **y** để kích hoạt Authentication (Kiểm định), hay *Enter* để bỏ qua tùy chọn này.

11. Gõ **y** để kích hoạt loại Authentication muốn sử dụng, hay **n** khi không dùng tùy chọn này.

12. Khẳng định sự lựa chọn bằng cách gõ **y**, hoặc gõ **n** nếu muốn thay đổi lại.

13. Đưa vào tất cả tên các host mà máy được cài đặt có thể điều khiển bằng FireWall hoặc ngược lại; mỗi tên host viết ở một dòng.
14. Khi đã điền hết tên các host, gõ Ctrl-D (hoặc ký tự EOF trên máy của bạn, nếu có quy định riêng).
15. Khi được yêu cầu khẳng định lại danh sách các host đã được lập, nếu đúng, ta ấn phím Enter, nếu sai, gõ “n” để sửa lại danh sách đó.
16. Đưa vào (hai lần) mật khẩu muốn sử dụng. Trên các máy nằm trong tầm quản lý của máy đang được cài đặt FireWall cũng đưa vào mật khẩu giống như trong máy cài đặt.
17. Trên màn hình sẽ hiện lên câu hỏi về vị trí đặt script khởi động FireWall. Đối với Solaris 2.x trở đi, thư mục mặc định là /etc/rc3.d. Gõ “y” để chấp nhận giá trị mặc định, gõ “n” nếu muốn định vị script ở thư mục khác. Trong trường hợp sau, mỗi khi muốn khởi động FireWall, ta phải thực hiện lệnh /etc/fw/bin/fwstart.
18. Trên màn hình sẽ hiện ra vài hướng dẫn cần phải làm theo để tạo lập môi trường chạy FireWall với C shell, đồng thời có cả chỉ dẫn về cách đăng ký bản quyền cho bản FireWall vừa được cài đặt. Các hướng dẫn đó tương tự như sau:

DON'T FORGET TO:

1. add the line:

```
setenv FWDIR /opt/SUNWfw to .cshrc
or FWDIR=/opt/SUNWfw
export FWDIR to .profile
```

2. add \$FWDIR/bin to path

3. add \$FWDIR/man to MANPATH enviroment

-----

Note: This product requires separate instalation of a license before use. Please consult your Solstice FireWall Administrator's Guide for information how to obtain and install a license for Solstice FireWall-1.

-----

19. Đối với Solaris 2.x trở đi, sau khi cài đặt xong, lệnh **pkgadd** giúp đưa ta trở lại màn hình giống như ở mục 2 phần này. Ta chọn “q” để kết thúc **pkgadd** sau khi cài đặt xong. Tháo đĩa CD-ROM cài đặt khỏi ổ đĩa CD bằng lệnh:

```
eject cdrom
```

khi daemon **vold** đang chạy.

20. Nếu **vold** không chạy, ta phải thực hiện thao tác trên bằng tay như sau:

```
umount /cdrom/firewall1_2_0
eject cdrom
rmdir /cdrom/firewall1_2_0
```

▪ *Cài đặt mật khẩu bản quyền*

21. Ngừng chạy FireWall trước khi cài mã bản quyền. Bản quyền được License Distribution Center (Trung tâm Phân phối Bản quyền Phần mềm) của Sun gửi về qua đường E-mail sau khi có yêu cầu cùng xuất trình các thông tin về việc mua phần mềm và về cấu hình phần cứng của máy định sử dụng cho FireWall và hostID của máy (phụ thuộc vào đĩa cứng). Lệnh được dùng như sau:

```
hostname% /etc/fw/bin/fwstop
```

22. Tại dấu nhắc dòng lệnh của root, đưa vào lệnh sau:

```
hostname% /etc/fw/bin/fw putlic <hostid> <license-
string> <keyword1 [keyword2] [...]>
```

23. Khởi động lại FireWall:

```
hostname% /etc/fw/bin/fwstart
```

#### ***XII. 4. c. Ví dụ cài đặt và cấu hình FireWall trên hệ Solaris 2.6***

Sau đây là ví dụ cụ thể cài đặt FireWall trên máy chạy Solaris 2.6. Tuy nhiên, ví dụ này có nhiều điểm chung với các hệ UNIX khác. Một trong các phương án tốt nhất là cài đặt FireWall ngay sau khi mới cài đặt hệ điều hành.

##### **1) Biên dịch các thông tin của host**

- 1.1 hostname= \_\_\_\_\_
- 1.2 IP addresses = \_\_\_\_\_
- 1.3 defaultrouter = \_\_\_\_\_ (nếu có)
- 1.4 timeserver = \_\_\_\_\_ (nếu có)
- 1.5 date = \_\_\_\_\_
- 1.6 root password = \_\_\_\_\_
- 1.7 part of a subnet = \_\_\_\_\_
- 1.8 timezone = \_\_\_\_\_ (ví dụ, US/Pacific)
- 1.9 naming service = \_\_\_\_\_ (ví dụ, DNS = other)

##### **2) Cài đặt Solaris 2. 6 từ CD**

2.1 Chọn phương án phần mềm "core system".

2.2 Tùy biến bằng cách thêm vào các gói phần mềm sau:

- a) "automated security enhancement tools"
- b) "interprocess communication"
- c) "network time protocol"
- d) "source Compatibility" (cho người dùng)
- e) "static utilities"
- f) "sunOS 4.x binary compatibility"
- g) "system accounting"
- c) "system & network administration"
- h) "volume management" (cho root và người dùng)
- i) "xcu4 utilities"

2.3 Phân chia ổ đĩa cứng bằng tay, sao cho dung lượng tối thiểu là:

root = 1.9GB+



swap = 500MB+

/var partition (nếu có), sẽ dùng phần đĩa còn lại

Chú ý:

\* kích thước của swap phụ thuộc ứng dụng.

\* các máy tính thế hệ cũ không hỗ trợ root partitions lớn hơn 2GB.

## 2.4 Chọn "install & reboot"

## 2.5 Cài đặt thêm các phần bổ sung được khuyến nghị dùng từ site:

<http://sunsolve.sun.com/pub-cgi/show.pl?target=patches/patch-access>

# 3) Cấu hình hệ thống

## 3.1 Sửa các file cấu hình:

vi /etc/hosts (xác định loghosts, mailhost, timehost)

vi /etc/hostname.\* (thay các địa chỉ IP cho phù hợp với các tên host)

vi /etc/vfstab (biến thành chú thích - tức là thêm dấu "#" ở đầu dòng vào - nếu không cần đến /tmp swapfs)

vi /etc/resolv.conf (tên vùng và 2 server tên vùng)

vi /etc/nsswitch.conf (cú pháp: hosts: files dns)

vi /etc/power.conf (biến thành chú thích autoshtutdown)

vi /etc/defaultrouter (đưa vào địa chỉ IP của router mặc định, tắt RIP)

vi /etc/ntp.conf (trong cấu trúc if dùng xntp để đồng bộ thời gian)

vi /etc/default/{su, login} (để cấm đăng nhập hệ thống bằng root từ mạng, phải sửa CONSOLE=/dev/console)

vi /etc/mail/sendmail.cf (sửa theo yêu cầu cụ thể các giá trị cho các thông số PrivacyOptions, MaxDaemonChildren, Mprog, MaxMessageSize, SuperSafe, ...)

## 3.2 Tạo các file cho tài khoản người dùng:

```
vi /.cshrc /.profile /.kshrc /.bashrc /.exrc
more /etc/skel/cshrc
```

3.3 Cài đặt các tiện ích quản trị hệ thống, ví dụ, lấy từ site <http://sunfreeware.com/>:

```
RCS: mkdir ~admin/RCS; ln -s ~admin/RCS /etc
 cd /etc; ci -l inetd.conf host* ftpusers, ...
 bash, gzip, less, lsof, rsync, sudo, tar, tcsh, top, zip, ...
```

3.4 Tắt các script khởi động không cần thiết trong các thư mục quy định mức chạy:

```
cd /etc/rcS.d
 mv S50devfsadm s50devfsadm
cd /etc/rc2.d
 mv S92volmgt s92volmgt
 #mv S88sendmail s88sendmail
 mv S74autofs s74autofs
 mv S73nfs.client s73nfs.client
 mv S99dtlogin s99dtlogin
 mv S85power s85power
 mv S47asppp s47asppp
 mv S70uucp s70uucp
 mv S80lp s80lp
 ...
cd /etc/rc3.d
 mv S15nfs.server s15nfs.server
 mv S76snmpdx s76snmpdx
 mv S77dmi s77dmi
 ...
```

3.5 Cấu hình việc ghi lại các file log:

vi /etc/syslog.conf (đổi khoảng trống thành Tab)

```
syslog,auth,local7,local5.debug;daemon.notice;local
6.info;user.none /dev/console
```

```
*.alert;auth.debug;kern.err;daemon.err;user.none
operator
```

```
kern.notice /var/log/kern.messages
daemon.debug /var/log/daemon.messages
user.debug /var/log/user.messages
cron.emerg /var/log/cron.messages
auth.info /var/log/auth.messages
news.debug /var/log/news.messages
mail.info /var/log/mail.messages
uucp.notice /var/log/uucp.messages
lpr.debug /var/log/lpr.messages
local0.info /var/log/local0.messages
local1.debug /var/log/local1.messages
local2.debug /var/log/local2.messages
local3.debug /var/log/local3.messages
local4.debug /var/log/local4.messages
local5.debug /var/log/local5.messages
local6.debug /var/log/local6.messages
local7.debug /var/log/local7.messages
debug @loghost2
```

### 3.6 Tạo ra các file log trong /var/log (hàng lệnh touch):

```
kill -1 <syslog pid>
touch /var/adm/loginlog
chmod o-w /var/{log,adm}/* /var/log/sysidconfig.log
```

### 3.7 Kích hoạt ASET:

```
/usr/aset/aset -p -l med
```

### 3.8 Đặt lịch định kỳ cho các tác vụ quản trị:

```
crontab -e
22 * * * * /var/admin/cron/hourly
22 22 * * * /var/admin/cron/daily
2 22 * * 0 /var/admin/cron/weekly
2 2 1 * * /var/admin/cron/monthly
```

### 3.9 Xóa sạch các file crontab không phải của root:

```
cd /var/spool/cron/crontabs
rm -f uucp sys adm lp
chmod go-rwx . *
ls -l
sed 's/:.*$//' /etc/passwd | grep -v root >
 /etc/cron.d/at.deny (hoặc rm at.deny)
sed 's/:.*$//' /etc/passwd | grep -v root >
 /etc/cron.d/cron.deny (hoặc rm cron.deny)
```

### 3.10 Thiết lập bí danh E-mail:

gửi chuyển root & postmaster tới người dùng thực  
gửi chuyển các tài khoản người dùng hệ thống tới /dev/null  
dùng lại danh sách các bí danh E-mail: /usr/lib/sendmail -bi

### 3.11 Xóa các file thừa:

```
rm -f /usr/sbin/{eeprom} /dev/openprom \
 /devices/pseudo/openeep@0:openprom
ls -l /usr/sbin/{eeprom} /dev/openprom \
 /devices/pseudo/openeep@0:openprom
rm -rf /var/spool/uucp* /etc/uucp
ls -l /var/spool/uucp* /etc/uucp
```

---

```
rm -rf /tftp* /.dt* /.Xauthority /.wastebasket /xfr \
 /net /export /mnt /core
ls -l /tftp* /.dt* /.Xauthority /.wastebasket /xfr \
 /net /export /mnt /core
rm -f /var/adm/spellhist
ln -s /dev/null /var/adm/spellhist
ln -s /dev/null /core (?)
```

### 3.12 Tắt các kết nối ftp đến:

```
sed 's/:.*$//' /etc/passwd > /etc/ftpusers
```

3.13 Tắt việc ghi log cho tác vụ “cron”, nhằm tránh cho đĩa cứng bị đầy sau này:

```
vi /etc/default/cron (đặt CRONLOG=NO)
```

### 3.14 Tạo hàng chữ trên màn hình đăng nhập hệ thống:

```
echo "----- Truy nhập trái phép bị cam -----"
 > /etc/issue
echo "" >> /etc/motd
echo "----- Truy nhập trái phép bị cam -----"
 >> /etc/motd
echo "" >> /etc/motd
```

### 3.15 Tắt việc phân đường bằng kernel:

```
echo "" >> /etc/rc2.d/S69inet
echo "ndd -set /dev/ip ip_forwarding 0" >> \
 /etc/rc2.d/S69inet
echo "ndd -set /dev/ip ip_forward_src_routed 0" >> \
 /etc/rc2.d/S69inet
```

---

```
echo "ndd -set /dev/ip ip_forward_directed_broadcasts\
 0" >> /etc/rc2.d/S69inet
vi /etc/default/inetinit (TCP_STRONGS_ISS=2)
touch /etc/notrouter
touch /etc/defaultrouter
```

### 3.16 Hạn chế các dịch vụ chuẩn và rpc:

- `vi /etc/inetd.conf`
- biến tất cả các dòng vào thành chú thích:
 

```
(:1,$s/^/#/)
```
- bỏ dấu chú thích đối với các dòng vào cần thiết:

```
kill -1 <inetd_pid>
echo "rpcbind : ALL" > /etc/hosts.deny (với rpcbind đã
sửa)
```

### 3.17 Kiểm tra file password:

`vipw`

- thêm shell `/bin/false` vào cho những người dùng hệ thống:
   
(đổi shell `root` tới `/bin/csh` nếu `/usr` nằm dưới `/`)
- thêm người dùng "admin":
 

```
(admin:x:0:1:Super-User:/var/admin:/bin/csh)
```
- thêm người dùng "operator" và "suoperator", nếu cần:

`pwck`

- kiểm tra file `/etc/passwd` để phát hiện lỗi:

`awk -F: '{print $2}' /etc/shadow`

- kiểm tra file `/etc/shadow` để phát hiện lỗi.

### 3.18 Kiểm tra file /etc/system:

```
grep -v '^*' /etc/system | grep -v '^$'
```

3.19 Thất chặt chủ quyền và các lỗ hổng bảo mật nếu có (tải phần mềm bảo mật từ các site của ISS và CERT):

```
find /opt/SUNWrtvc -perm 777 -exec chmod go-w {} \; -ls
chmod 660 /devices/sbus@1f,0/espdma@e,8400000/esp@e, \
 8800000/st@4*
chmod 550 /usr/lib/fs/ufs/ufsrestore
chmod 555 /usr/openwin/bin/ff.core
#chmod 550 /bin/su
chmod 000 /etc/default/sys-suspend \
 /usr/openwin/bin/sys-suspend
vi /usr/openwin/lib/speckeyds.map
chgrp sysadmin `which admintool`
chmod o-rwx `which admintool`
chmod -R ugo+r /usr/openwin
chmod 400 /usr/vmsys/bin/chkperm
```

## 4) Mã hóa và kiểm tra xác nhận

### 4.1 Cài đặt shell có tăng cường bảo mật (secured shell) **ssh**:

```
ftp://ftp.cs.hut.fi/pub/ssh/
ftp://sunsite.unc.edu/pub/packages/security/ssh/
ftp://ftp.kyoto.wide.ad.jp/pub/security/ssh/
vi /etc/services (ssh 22/tcp)
vi sshd_config (với nội dung sau)
 kill -1 `ps auxww|grep inetd|grep -v grep|awk \
 '{print $2}'`
 telnet localhost 22
```

Nếu không có **inetd** : dùng script khởi động (/etc/rc2.d/S81sshd)  
hay thêm vào **inetd.conf** .

## 5) Kiểm tra và kiểm tra lại

### 5.1 Tại chỗ:

```
ps auxww (ps -ef) ; netstat -a ; showrev -p ; prtconf ;
...
```

### 5.2 Từ xa:

```
nmap ; portscan ; rpcinfo -p ; ...
```

## 6) Các phần mềm phụ trợ:

### 6.1 Biên dịch bảo mật **rpcbind**, nếu bạn phải dùng **rpc**:

Tải mã nguồn của **rpcbind** về từ site [ftp://ftp.win.tue.nl/pub/security/rpcbind\\_2.tar.gz](ftp://ftp.win.tue.nl/pub/security/rpcbind_2.tar.gz), sau đó, thực hiện các thao tác sau:

```
make; cp /usr/sbin/rpcbind /usr/sbin/rpcbind.ori;
cp rpcbind /usr/sbin/rpcbind; reboot
vi /etc/hosts.deny (và /etc/hosts.allow) để kiểm tra việc đăng
nhập hệ thống
```

Chú ý: Thông thường **KHÔNG** dùng các chương trình do hệ điều hành có sẵn (**bind**, **dhcp**, v.v...) hay **sendmail** để biên dịch. Hãy biên dịch bằng công cụ có sẵn với mã nguồn.

### 6.2 Cách tạo lập các cron script thường kỳ (/var/admin/cron/ hoặc /etc/periodic/):

hàng giờ :

**ping** các host từ xa

kiểm tra/đặt chủ quyền cho hộp thư điện tử.



kiểm tra các daemon (`syslog`, `sendmail`, `named`, `fwm`, `ssh`)

làm trống mọi file log khi kích thước của chúng vượt quá 1.5MB

hàng ngày:

kiểm tra mức độ sử dụng đĩa cứng

kiểm tra hàng chờ `sendmail`

Đồng bộ thời gian (lệnh `ntpdate`, `rdate`)

hàng tuần (có báo cáo qua E-mail):

kiểm tra mức độ sử dụng đĩa cứng

kiểm tra/báo cáo (file `password`, hàng chờ SYN, `hosts.{allow,deny}`),

dăng nhập ftp, hàng chờ `sendmail`, kích thước của `wtmp/wtmpx`)

báo cáo về ASET

các patch (miếng vá) cài đặt sửa lỗi của Solaris

chạy riêng rẽ các báo cáo về các chương trình `tripwire` và `mail`

hàng tháng (có báo cáo qua E-mail):

làm tươi và lập hồ sơ lưu trữ các file log

### 6.3 Cấu hình `ssh`:

File `/etc/sshd*config`:

|                                      |                     |
|--------------------------------------|---------------------|
| <code>PermitRootLogin</code>         | <code>no</code>     |
| <code>IgnoreRhosts</code>            | <code>yes</code>    |
| <code>SyslogFacility</code>          | <code>LOCAL3</code> |
| <code>RhostsAuthentication</code>    | <code>no</code>     |
| <code>RhostsRSAAuthentication</code> | <code>no</code>     |
| <code>RSAAuthentication</code>       | <code>no</code>     |
| <code>PasswordAuthentication</code>  | <code>yes</code>    |
| <code>PermitEmptyPasswords</code>    | <code>no</code>     |
| <code>UseLogin</code>                | <code>no</code>     |
| <code>AllowHosts</code>              | <code>...</code>    |

#### 6.4 Tải xuống, biên dịch và cài đặt **skey**:

- Tải về từ site <ftp://ftp.win.tue.nl/pub/security/logdaemon...tar.gz>
- Biên dịch **skeysh**
- Để cho phép đăng nhập bằng tài khoản người dùng của **root**, phải biến thành chú thích phần "ROOT LOGIN REFUSED " trong file **skeysh/skeysh.c**.
- (tạo tài khoản **skey** , có hay không có mật khẩu cho **ssh**)

#### 6.5 Firewall-1:

- Cài đặt dưới **/opt/SUNWfw**  
Chọn tất cả các patch.
- Thêm những dòng sau vào file **/.cshrc**

```
if (-d /opt/SUNWfw) then
 setenv FWDIR /opt/SUNWfw
 setenv PATH ${PATH}:${FWDIR}/bin
 setenv MANPATH ${MANPATH}:${FWDIR}/man
endif
```

- Dùng lệnh soạn thảo để sửa: **vi /etc/rc3.d/S95firewall1** .  
Thêm vào **\$FWDIR/bin/fwm**
- Tương tự, **vi /opt/SUNWfw/bin/fwstart**  
Nếu không cần thiết, biến thành chú thích dòng vào để khởi động **snmp**
- Thêm vào các người dùng **mgmt** (quản lý)

```
fwm -a
```

- Thêm vào các host **mgmt** (không phải là **\$FWDIR/conf/masters**):

```
vi $FWDIR/conf/gui-clients
```

- Xóa sạch file bản quyền, nếu được yêu cầu:

```
cd $FWDIR/conf
mv fw.license fw.license.ori
mv fw.LICENSE fw.LICENSE.ori
../bin/fw putlic ...
```

- Tắt SNMP; Security Policy, Properties, ...
- Khởi động lại và kiểm tra file `/etc/rcS.d/S30rootuser.sh`

6.6 Tạo các mật khẩu dùng một lần, xác định xem có phải chỉ để cho `root` hay không:

```
skey/logdaemon
/usr/sbin/skeysh, skey account; keyinit+key
```

6.7 Kiểm tra thử tính toàn vẹn và khả năng chịu tải của hệ thống file bằng các tiện ích sau:

```
iss s3, (axent), tripwire, ...
```

6.8 Nếu dịch vụ `inetd` được kích hoạt, chạy:

```
tcp_wrappers (để đăng nhập hệ thống và kiểm tra xác nhận tài khoản)
tcpdchk
```

6.9 Kiểm tra **sendmail** bằng:

```
spam filters, RBL, DUL, ORBS
```

## XIII. Bảo trì phần cứng hệ thống

### XIII. 1. Tổ chức bộ nhớ

#### *XIII. 1. a. Thay đổi kích thước swap*

Có thể thêm swap vào máy như là một phần logic của ổ đĩa cứng (partition) hay như một file. Swap trên partition – thường được thực hiện bằng cách dành một phần đĩa khi phân chia đĩa cứng thành các phần logic để cài đặt các hệ thống file – hiệu quả hơn swap bằng file. Tuy nhiên trong các phiên bản cuối của UNIX sự khác biệt này hầu như bị xóa bỏ. Vì vậy, ở đây ta chủ yếu xem xét trường hợp các file swap, là cách thức mà ta thường áp dụng **sau khi** đã cài đặt xong toàn bộ hệ thống. Đối với máy tại chỗ, việc thêm kích thước swap được thực hiện với lệnh **swap**. Đối với các hệ thống file NFS khi thêm swap bằng lệnh thực hiện từ máy trạm, máy chủ còn cần xuất nó ra.

Các bước thực hiện như sau:

- Nếu sử dụng hẳn một partition đĩa cứng mới được phân chia cho swap thì ta bỏ qua bước này. Nếu dùng swap trên file, đầu tiên ta phải tạo ra một hay nhiều file riêng cho swap bằng lệnh có cú pháp như sau:

```
mkfile [-nv] size[k|b|m] filename ...
```

với kích thước file được đo bằng byte theo mặc định, hoặc nếu có chỉ rõ đơn vị k (KB), b (block), m (MB). Tùy chọn **-n** khiến hệ thống ghi nhận việc tạo ra file, nhưng thực tế các khối dữ liệu chỉ phân bố trên đĩa nếu có dữ liệu được ghi vào. Tùy chọn này không áp dụng được cho các điểm mount tại chỗ cho UFS. Tùy chọn **-v** thông báo tên và kích thước của file được tạo trên màn hình.

- Thêm vào trong file `/etc/dfs/dfstab` dòng vào sau:

---

```
share -F nfs -o rw=clientname, root=clientname path- \
to-swap-file
```

- Chạy lệnh:

```
shareall
```

- Trên máy client thêm các dòng sau vào file /etc/vfstab:

```
server:path-to-swap-file -local-path-to-swap-file
nfs --- local-path-to-swap-file -- swap ---
```

- Trên máy client, chạy lệnh:

```
mount local-path-to-swap-file
```

- Client chạy lệnh để thêm swap:

```
swap -a local-path-to-swap-file
```

Nhắc lại, đối với máy tại chỗ chỉ cần thực hiện bước 3 và 5. Để bớt swap, ta dùng lệnh trên với tùy chọn d.

Cú pháp của lệnh **swap** như sau:

```
usr/sbin/swap -a swapname [swaplow] [swaplen]
usr/sbin/swap -d swapname [swaplow]
usr/sbin/swap -l
usr/sbin/swap -s
```

với:

**a**

Thêm

**d**

Bớt

**l**

Liệt kê trạng thái swap theo các trường path, dev, swaplo, blocks, free.

**s**

Tất cả thông tin về swap theo các trường allocated, reserved, used, available.

**swapname**

Tên file swap, ví dụ: /dev/dsk/c0t0d0s1 hoặc file thường, được tạo bằng lệnh **mkfile**.

**swaplow**

Địa chỉ thấp nhất được dùng cho vùng swap.

**swaplen**

Độ dài file swap theo đơn vị block 512 byte. Giá trị tối thiểu là 1 trang bộ nhớ, tức là 8 block hay 4 KB.

### ***XIII. 1. b. Thiết lập hệ thống file cache***

Hệ thống file cache giúp làm giảm thời gian truy cập vào NFS hay các môi trường hệ thống file có tốc độ thấp bằng cách lưu sẵn trên đĩa tại chỗ sau lần truy cập đầu. Nói cách khác, cache điều khiển việc làm chìm xuống hay nổi lên các chương trình trong giao diện với người dùng. Để thiết lập hệ thống file cache, ta làm như sau:

- Tạo thư mục cache /local/cache và các thông số ban đầu của nó:

```
cfsadmin -c -o maxblocks=80,minblocks=30,
 threshblock=60 /local/cache
```

- Sửa đổi các thông số cache theo yêu cầu:

```
cfsadmin -u -o parameter1=value1,parameter2= value2
/local/cache
```

- Mount hệ thống file "chìm" UFS:

```
mount -F cachefs -o backfstype=nfs, cachedir=
/local/cache server:/export/home
```

- Mount hệ thống file "chìm" CDROM:

```
mount -F cachefs -o backfstype=hsfs,ro,backpath=
 /cdrom/cd_name /cdrom/cd_name/mount/point
```

- Để mount hệ thống file từ /etc/vfstab ta dùng các dòng vào có dạng như sau:

| device<br>to mount      | device<br>to fsck | mount<br>point | FS<br>type | fsck<br>pass | mount<br>at boot | mount<br>options      |
|-------------------------|-------------------|----------------|------------|--------------|------------------|-----------------------|
| server:<br>/export/home | /local<br>/cache  | /home          | cachefs    | 2            | yes              | rw,<br>backfstype=nfs |

- Xem trạng thái cache và các thư mục "chìm":

```
cfsadmin -l /local/cache
```

Để loại bỏ cache, trước hết phải **umount** nó, sau đó dùng lệnh **cfsadmin** với tùy chọn d.

### ***XIII. 1. c. Tìm hiểu hệ thống file /proc***

Phần này được biên soạn dựa trên bài báo của Robert Owen Thomas trong tạp chí *Inside Solaris*<sup>1</sup>. Trong sơ đồ hệ thống thư mục được trình bày ở đầu chương *Hệ thống file UNIX*, có một thư mục /proc luôn tồn tại, dù rằng hệ UNIX được cài đặt với phương án tối thiểu đi chăng nữa. Vai trò của thư mục /proc là gì, và tại sao trong đó có nhiều file lớn đến thế? Trước đây, việc lấy các thông tin (ví dụ bằng lệnh **ps**) về các tiến trình đang chạy là một công việc rất không đơn giản, đòi hỏi truy nhập trực tiếp đến bộ nhớ của kernel và tất nhiên, cả quyền root nữa. Từ UNIX SVR4, Sun đã giải quyết khó khăn này và cho phép ta sử dụng /proc để lấy thông tin một cách dễ dàng hơn.

Thư mục /proc không chỉ đơn giản là hệ thống file theo nghĩa thông thường, mà còn là giao diện tối không gian địa chỉ của các tiến trình đang chạy, tức là tối địa chỉ các vùng của bộ nhớ máy tính (RAM, swap và cả cache) được huy động cho các tiến trình đang được nói tới. Tài nguyên bộ nhớ được dùng cho tiến trình đang chạy bao hàm không những bản sao chương trình tương ứng (xem lại cách dùng các

<sup>1</sup> Robert O. Thomas, *Inside Solaris*, 2001, <http://www.elmentkjournals.com/sun/>.

lệnh gọi hệ thống **fork**, **exec**, và về cách xử lý tiến trình trong UNIX), mà còn cả địa chỉ dùng cho các file được tiến trình đó mở, v.v... Với `/proc` ta có thể dùng các lệnh gọi hệ thống chuẩn của UNIX, ví dụ, **open()**, **read()**, **write()**, và **ioctl()** để hỏi hay điều khiển không gian địa chỉ của các tiến trình. Thực vậy, lệnh **ps** của Solaris dùng `/proc` để xác định trạng thái của các tiến trình được liệt kê trong bảng tiến trình.

Các file lớn được nhìn thấy trong `/proc` là các không gian địa chỉ của các tiến trình đang chạy - không phải là các file chuẩn của UNIX. Mỗi một tên file trong thư mục đó thực chất là PID của tiến trình đang chạy. Người sở hữu và nhóm sở hữu của mỗi file cũng là các định danh thực của người dùng (UID) và nhóm đầu tiên của người sở hữu tiến trình (Nhắc lại là Solaris coi như mỗi người dùng thuộc về một nhóm, hẳn hữu lắm mới thuộc về các nhóm thứ cấp khác, trong khi đó, số nhóm trong các hệ UNIX khác (BSD UNIX) lớn nhất có thể là 128). Các bit chủ quyền cũng giống như đối với các file UNIX khác. Phần khác biệt nhất là kích thước của file - đó thực chất là tổng số bộ nhớ (kích thước ảnh) mà tiến trình sử dụng, chứ không phải là dung lượng các byte thực chiếm chỗ trên đĩa cứng. Tuy nhiên, không nên xóa các file này đi. Nội dung thư mục `/proc` có thể được trình bày tương tự như trong ví dụ sau:

```
$ ls -l /proc
total 43384
-rw----- 1 root root 0 Apr 2 20:07 00000
-rw----- 1 root root 393216 Apr 2 20:07 00001
-rw----- 1 root root 0 Apr 2 20:07 00002
-rw----- 1 root root 0 Apr 2 20:07 00003
-rw----- 1 root root 1695744 Apr 2 20:07 00081
-rw----- 1 root root 1597440 Apr 2 20:07 00083
-rw----- 1 root root 1777664 Apr 2 20:08 00096
-rw----- 1 root root 1683456 Apr 2 20:08 00099
-rw----- 1 root root 1589248 Apr 2 20:08 00101
-rw----- 1 root root 1445888 Apr 2 20:08 00116
-rw----- 1 root root 1404928 Apr 2 20:08 00126
```



---

```

-rw----- 1 root root 798720 Apr 2 20:08 00135
-rw----- 1 root root 1368064 Apr 2 20:08 00195
-rw----- 1 root root 1585152 Apr 2 20:08 00197
-rw----- 1 root root 1368064 Apr 2 20:08 00200
-rw----- 1 root other 225280 Apr 2 20:08 00201
-rw----- 1 root root 1454080 Apr 2 20:08 00203
-rw----- 1 root root 1519616 Apr 2 20:14 00243
-rw----- 1 hien users 1499136 Apr 2 20:14 00245
-rw----- 1 hien users 806912 Apr 2 20:16 00261

```

Thao tác trên các file trong `/proc` cũng tương tự như đối với các file UNIX chuẩn, mọi lệnh gọi hệ thống đều có tác dụng, kể cả `ioctl()`.

Các tiện ích được Solaris cung cấp, dùng `/proc` và nằm trong thư mục `/usr/proc/bin`, có thể cho rất nhiều thông tin quan trọng trong số dữ liệu về tiến trình đã cho. Ví dụ, nếu muốn biết bao nhiêu file đã được một tiến trình nào đấy mở, ta có thể dùng lệnh **crash**, nhưng ta lại không có quyền root. Giải pháp luôn có sẵn - ta có thể dùng lệnh `/usr/proc/bin/pfiles` như sau:

```

$ ps
 PID TTY TIME CMD
 245 pts/0 0:00 ksh
$ pfiles 245
245: -ksh
Current rlimit: 64 file descriptors
0: S_IFCHR mode:0620 dev:32,24 ino:197800 uid:101
 gid:7 rdev:24,0
 O_RDWR
1: S_IFCHR mode:0620 dev:32,24 ino:197800 uid:101
 gid:7 rdev:24,0
 O_RDWR
2: S_IFCHR mode:0620 dev:32,24 ino:197800 uid:101
 gid:7 rdev:24,0
 O_RDWR

```

```
63: S_IFREG mode:0600 dev:32,27 ino:2881 uid:101
gid:666 size:282
O_RDWR|O_APPEND close-on-exec
```

Tuy nhiên, phải nhớ để ý tới chủ quyền. Bạn sẽ không thể truy nhập được tới dữ liệu của một tiến trình với PID đã cho, nếu không có các chủ quyền tương ứng.

Có thể sử dụng hai cấu trúc **prstatus** và **prpsinfo**, đều được xác định trong file `/usr/include/sys/procfs.h`, để tạo ra các script nhằm lấy ra các thông tin về tiến trình. Ví dụ, cần xem tổng kích thước của file ảnh, của ngăn xếp bộ nhớ (stack), v.v.. đối với một tiến trình nào đó. Chúng ta có thể đơn giản hóa công việc trên bằng cách dùng hệ thống file `/proc`. Tốt nhất là bạn đọc trong phần thực hành nên cố gắng viết một shell script bằng C-shell tên là **memlook** để hiển thị một cách thống kê bộ nhớ dùng cho một tiến trình có định danh PID nào đó sau mỗi khoảng thời gian xác định. Ví dụ, kết quả hiển thị của **memlook** đối với tiến trình có định danh 245 như sau:

```
$ memlook 245
PID IMAGE RSS HEAP STACK
245 1499136 1044480 24581 8192
```

## XIII. 2. Dùng đĩa mềm

Đĩa mềm trong hệ thống UNIX – cũng như mọi thiết bị phần cứng khác như đĩa cứng, ổ CD/DVD, modem, v.v.. - được xem như các file đặc biệt, được xử lý một cách nhất quán bởi hệ điều hành, chứ không đơn giản như trong các hệ điều hành Windows của Microsoft như chỉ là môi trường lưu giữ dữ liệu. Các hệ thống file trên đĩa mềm (1.44 MB) cũng có thể được UNIX tạo ra hay truy cập tới theo một trong những phương pháp sau:

### *XIII. 2. a. Dùng các file thiết bị đặc biệt cho đĩa mềm*

Các file thiết bị đặc biệt dùng cho đĩa mềm (thông số mặc định 3.5 inch, 1.44 MB) trong các hệ UNIX gần đây được liệt kê trong bảng sau:

| UNIX    | Tên file thiết bị                        | Mô tả                                                                                                                        |
|---------|------------------------------------------|------------------------------------------------------------------------------------------------------------------------------|
| AIX     | /dev/rfd0                                | Thêm l =< 1.44, h = 2.88 MB                                                                                                  |
| HP-UX   | /dev/rdisk/c*t*d0<br>/dev/rfloppy/c*t*d0 |                                                                                                                              |
| Digital | /dev/rfd0                                |                                                                                                                              |
| SCO     | /dev/rdisk/f03ht<br>/dev/rfd0            | Khuôn dạng: /dev/rf0sdu -<br>s=size; u=usage (t=all;<br>u=trừ cyl 0, rãnh 0; null=trừ cyl 0);<br>d=density (h=high;d=double) |
| Linux   | /dev/fd0                                 |                                                                                                                              |
| IRIX    | /dev/rdisk/fds0d2.3.5                    |                                                                                                                              |
| Solaris | /dev/diskette<br>/dev/rdiskette          | Tên thiết bị cũ /dev/fd0                                                                                                     |

Hãy theo dõi tiến trình dùng một đĩa mềm đã định dạng trong hệ thống SunOS như sau:

```
mkdir /floppy
mount -F pcfs /dev/rdiskette0 /floppy
ls -l /floppy
cd / export/home/tuan/bienban
cp -p /floppy/tailieu2 .
umount /floppy
eject
```

### ***XIII. 2. b. Dùng đĩa mềm MS-DOS trong hệ UNIX***

#### **▪ *Daemon vold***

Trong hệ thống Solaris các đĩa mềm được kiểm soát bằng cơ chế quản lý đĩa và daemon **vold** của nó. Lệnh **volcheck** mặc định được dùng cho đĩa mềm và có tác dụng kiểm tra có đĩa trong ổ hay không (ký hiệu sec chỉ đơn vị thời gian là giây):

---

```
volcheck [-v] [-i secs] [-t secs] pathname
```

Trong ví dụ sau, một đĩa mềm được định dạng, trao đổi các file với hệ thống và lấy ra khỏi máy:

```
[/etc/init.d/]volmgt stop
[/etc/init.d/]volmgt start
volcheck
fdformat -t dos -b diamem1
cp /export/home/baihoc/bai*.txt /floppy/diamem1
cp /floppy/diamem1/bai2.txt /opt/luu
eject
```

#### ▪ *Định dạng đĩa mềm*

Một đĩa mềm có thể được định dạng sao cho có thể dùng được trong cả UNIX và DOS bằng lệnh có cú pháp sau (trường hợp Solaris):

```
fdformat [-dDeEfHlLmMUqvx] [-b label] [-B filename] [-t
dostype] [devname]
```

Các tùy chọn:

**d**

Định dạng theo chuẩn DOS. Nên dùng tùy chọn tương đương `t dos`.

**D**

Định dạng đĩa 720 KB (3.5") hoặc 360 KB (5.25") như các tùy chọn `l`, `L`.

**e**

Nhả đĩa ra khỏi ổ sau khi định dạng.

**E**

Định dạng đĩa 2.88 MB.

**f**

Bắt đầu định dạng không hỏi lại người thao tác trên máy.

**H**

Định dạng 1.44 MB (3.5") hay 1.2 MB (5.25"). Mặc định.

**U**

umount mọi hệ thống file rồi định dạng đĩa.

**q**

Không thông báo trạng thái.

**v**

Kiểm tra từng block đĩa.

**x**

Không định dạng, chỉ ghi nhãn SunOS lên đĩa.

**b** *label*

Đặt nhãn đĩa với độ dài của *label* cho đến 8 ký tự.

**B** *filename*

Định dạng đĩa DOS tự khởi động bởi bộ nạp trong *filename*. Chỉ có nghĩa với tùy chọn *d* hay *t dos*.

### ***XIII. 2. c. Dừng như thiết bị ký tự***

Thiết bị ký tự được xử lý bằng các lệnh như **cpio**, **tar**, v.v.. Ví dụ:

```
[/etc/init.d/]volmgt stop
tar cvf /dev/rdiskette0 /data/pwflag
```

Có thể thực hiện chép file trên nhiều đĩa mềm. Máy sẽ dừng chờ thay đĩa tiếp nếu kích thước các file trao đổi lớn hơn một đĩa:

```
ls | cpio -oacB -H odc -O /dev/rdiskette0 -M filename
```

hay:

```
find pathname -depth -print | cpio -pdlmv -o >
 /dev/rdiskette0
```

### XIII. 3. Thêm đĩa cứng

#### XIII. 3. a. Giao diện SCSI và RAID

Small Computer System Interface (SCSI) hiện nay đã trở thành chuẩn công nghiệp thông dụng để nối các thiết bị ngoại vi như đĩa cứng, ổ băng từ, CD, ổ quang từ, v.v., vào máy tính, nhằm làm cho máy tính không phụ thuộc vào các thiết bị cụ thể. Hiện nay người ta phân ra các loại sau: SCSI chuẩn - còn gọi là SCSI-1, Fast SCSI hay SCSI-2 và Wide SCSI. Các chỉ tiêu kỹ thuật chủ yếu của ba loại trên được so sánh ở bảng sau:

| Các chỉ tiêu kỹ thuật chính                                               | SCSI-1              | Fast SCSI                          | Wide SCSI                      |
|---------------------------------------------------------------------------|---------------------|------------------------------------|--------------------------------|
| Số thiết bị nối vào (ID cuối dành cho điều khiển và nối với bus máy tính) | 0 - 7               | 0 - 15                             | 0 - 15                         |
| Tốc độ trao đổi dữ liệu đồng bộ cực đại (MBytes/s)                        | 5                   | 10                                 | 40                             |
| Độ rộng bus (bits)                                                        | 8                   | 8                                  | 16 hoặc 32                     |
| Độ dài cáp cực đại (m)                                                    | 6                   | 6                                  | 25                             |
| Số đầu dây trong cáp                                                      | 50                  | 50                                 | 68                             |
| Khả năng lắp lẫn trên một bus                                             | Được                | Được                               | Phải có bộ chuyển tiếp         |
| Loại terminator                                                           | Thụ động (diện trở) | Chủ động (điều khiển bằng điện áp) | Thụ động (khác với của SCSI-1) |

Ngày nay, trong các máy chủ có hai bus SCSI, thường gọi là Bus 0 và Bus 1 với điều khiển *onboard* - ghép sẵn trên board mạch chính, dựa trên các sản phẩm Adaptec AHA (154x, 274x, 284x, 294x), BusLogic (BT-946C, 956C), Compaq (53C710, 53C8xx), IBM Micro Channel,..., để cắm thẳng các thiết bị vào máy. Tuy nhiên, để nâng cao tính năng của máy, người ta còn dùng các loại card điều khiển SCSI ngoài như Compaq SMART-2, DPT (PM-202x, 212x, 22xx, 322x), Mylex 960,

v.v.. Có thể lắp card điều khiển SCSI (và các thiết bị mà nó quản lý) vào một bus máy, còn bus kia nối với các thiết bị khác như ổ CD hoặc băng từ.

Do có đến 7 hoặc 15 thiết bị được nối vào máy nên phải định danh chúng bằng các ID. Số ID càng cao, mức độ ưu tiên càng lớn. Đối với SunOS và Solaris (for SPARC), thiết bị khởi động được định danh mặc định là ID3. Phân chia SCSI ID trong SunOS như sau:

| Bộ điều khiển | SCSI ID | Tên file thiết bị |
|---------------|---------|-------------------|
| 0             | 3       | sd0 ( /dev/sd0a ) |
| 0             | 1       | sd1               |
| 0             | 2       | sd2               |
| 0             | 0       | sd3               |
| 1             | 3       | sd4               |
| 1             | 1       | sd5               |
| 1             | 2       | sd6               |
| 1             | 0       | sd7               |

Dữ liệu trong UNIX, nhất là đối với những file lớn, được xử lý bằng kỹ thuật *stripping* (phân mảnh) trong khi hệ thống thực hiện các thao tác trao đổi dữ liệu. Một cách vắn tắt, kỹ thuật này cho phép kết hợp nhiều đĩa vật lý hoặc partition thành một đĩa logic, một file theo cách hiểu truyền thống của UNIX. Do đó, sự trao đổi dữ liệu vào/ra lẽ ra phải thực hiện một cách tuần tự đối với từng file, trên từng đĩa vật lý hay partition, thì nay được chia thành những phần nhỏ để thực hiện song song trên tất cả các đĩa hoặc partition ấy, như là thao tác trao đổi dữ liệu cho một đĩa (tức là file trong UNIX) duy nhất. Điều đó giúp tăng đáng kể tốc độ xử lý dữ liệu so với đĩa cứng đơn và cách truy xuất dữ liệu cổ điển.

Để tăng khả năng đảm bảo an toàn dữ liệu người ta dùng công nghệ RAID (**R**edundant **A**rray of **I**ndependent **D**isks) nhờ cách tổ chức phân chia dữ liệu trên đĩa cứng. RAID hiện nay được dùng phổ biến không những cho các máy tính có kiến trúc phần cứng (bo mạch chủ) theo kiểu server, mà còn trong cả các máy tính để bàn (desktop). RAID được chia làm sáu mức. Bảng so sánh các mức RAID như sau:

| Mức | Mô tả                                                                                | Đặc điểm                                                                                                                                                                |
|-----|--------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 0   | Stripping.                                                                           | Tốc độ cao nhất (đặc biệt với file lớn). Dung lượng đĩa có ích lớn nhất. Mức độ an toàn thấp nhất.                                                                      |
| 1   | Sao chép gương từng đĩa.                                                             | 100% dữ liệu được sao chép. Mức độ an toàn cao nhưng tốn đĩa. Số đĩa: chẵn.                                                                                             |
| 2   | Stripping theo sector trên nhóm đĩa.                                                 | Hiện không dùng trong thực tế.                                                                                                                                          |
| 3   | Stripping theo sector trên nhóm đĩa, nhưng dành riêng một đĩa cho thông tin chẩn lẻ. | Tốn đĩa. Tốc độ xử lý tốt nhất cho các thao tác ra vào lớn. Khi một đĩa bị thay thế, dữ liệu được phục hồi theo phép logic XOR. Môi trường một người dùng.              |
| 4   | Stripping theo đĩa. Dành riêng một đĩa cho thông tin chẩn lẻ.                        | Hiện không dùng trong thực tế.                                                                                                                                          |
| 5   | Stripping theo đĩa. Thông tin chẩn lẻ được chia trên tất cả các đĩa.                 | Tỷ số tốc độ xử lý dung lượng đĩa tốt nhất. Môi trường nhiều người dùng. Thao tác đọc đồng thời trên các đĩa, ghi trên từng đĩa kèm thông tin chẩn lẻ. Tối thiểu 3 đĩa. |

Trong thực tế, tốt nhất ta nên dùng RAID-5, tiếp đến là RAID-10 (tổ hợp hai mức 1 và 0) cho máy chủ.

### **XIII. 3. b. Thêm đĩa cứng mới**

Khi lắp thêm đĩa cứng vào máy, bật máy lên, trên màn hình của hệ thống SunOS hay Solaris ta thấy hiện mấy dòng thông báo sau:

```
yorick UNIX : c0t2d0 at esp0 target2 lun0
yorick UNIX : corrupt label - wrong magic number
yorick UNIX : Vendor 'Quantum', product 'CTS160S',
 333936 512 byte blocks
```



(đối với SunOS tên file và partition đĩa tương ứng: vmUNIX và sd2), đó là hệ thống tìm thấy đĩa và cố đặt tên hệ thống cho nó. Tiếp theo, ta phải dùng tiện ích định dạng **format**. (Nếu dùng các điều khiển SCSI ngoài, có thể dùng luôn các tiện ích phần mềm kèm theo, ví dụ, định dạng trong DPT).

Tiện ích **format** được thiết kế dưới dạng menu cấu trúc cây. Theo ví dụ trên, trước hết, ta xác định loại đĩa:

```
format c0t0d2
Searching for disks ... done
FORMAT MENU:
 disk - select a disk
 type - select (define) a disk type
 partition - select (define) a partition table
 current - describe the current disk
 format - format and analyze the disk
 repair - repair the defective sector
 show - translate a disk address
 label - write label to the disk
 analyze - surface analyze
 defect - defect list management
 backup - search for backup label
 quit
format> type
AVAILABLE DRIVE TYPES:
 0. Quantum ProDrive 80S
 1. Other
Specify disk type (enter its number) :1
Enter number of data cylinders : 1302
Enter number of alternative cylinders [2]: <Enter>
Enter number of physical cylinders [1304]: <Enter>
Enter number of head : 4
Enter number of data sectors/track: 64
```

```
Enter rpm of drive [3600]: <Enter>
Enter disk name type (remember quotes): "Quantum
CTS160S"
Selecting sd2 < Quantum CTS160S >
```

Nếu có nhiều hơn một đĩa mới thêm vào, phải nhắc lại thao tác cho từng đĩa. Thường là các đĩa đã được định dạng sẵn (**format** trong UNIX chỉ là định dạng - tương đương định dạng thấp trong MS-DOS). Nếu cần, có thể định dạng lại đĩa:

```
format> format
```

Các lỗi trên đĩa cũng có thể được khắc phục phần nào nhờ lệnh con **repair**. Tuy nhiên, nên thận trọng, nếu đĩa của bạn chứa các thông tin quý giá chưa được lưu trữ hoặc chứa chính hệ điều hành.

Sau đó, ta phải tạo partition bằng lệnh con **partition** trong menu chính. Giả sử ta tạo ba partition trên đĩa:

```
format> partition
PARTITION MENU:
0 - change `0' partition
1 - change `1' partition
2 - change `2' partition
3 - change `3' partition
4 - change `4' partition
5 - change `5' partition
6 - change `6' partition
7 - change `7' partition
8 - change `8' partition
9 - change `9' partition
select - select a predefined table
name - name the current table
print - display the current table
```

```
label - write partition map and label to the
disk
quit
partition> print
Curent partition table (unnamed) :
partition 0 - starting cyl0, # blocks 0 (0/0/0)
partition 1 - starting cyl0, # blocks 0 (0/0/0)
partition 2 - starting cyl0, # blocks 0 (0/0/0)
partition 3 - starting cyl0, # blocks 0 (0/0/0)
partition 4 - starting cyl0, # blocks 0 (0/0/0)
partition 5 - starting cyl0, # blocks 0 (0/0/0)
partition 6 - starting cyl0, # blocks 0 (0/0/0)
partition 7 - starting cyl0, # blocks 0 (0/0/0)
partition 8 - starting cyl0, # blocks 0 (0/0/0)
partition 9 - starting cyl0, # blocks 0 (0/0/0)
partition> 3
partition - starting cyl 0, # blocks 0 (0/0/0)
Enter new startng cyl [0]: <Enter>
Enter new # blocks [0, 0/0/0]: 605/0/0
partition> 4
partition - starting cyl 0, # blocks 0 (0/0/0)
Enter new startng cyl [0]: 605
Enter new # blocks [0, 0/0/0]: 300/0/0
partition> 5
partition - starting cyl 0, # blocks 0 (0/0/0)
Enter new startng cyl [0]: <805>
Enter new # blocks [0, 0/0/0]: 297/0/0
```

Bước tiếp theo là đặt nhãn cho đĩa:

```
partition> label
Ready to label disk, continue? Y
```

Menu con partition còn có lệnh **name** cho phép đặt tên cho bảng partition của người dùng.

Trong ví dụ ta đã dùng lệnh con **print** để xem thông tin các partition. Có thể dùng lệnh UNIX ở ngoài dấu nhắc dòng lệnh để đạt kết quả tương tự. Đó là lệnh **prtvtoc** (Solaris) ta đã làm quen hoặc **dkinfo** (SunOS).

Nếu kiểu định dạng của đĩa loại Quantum này chẳng hạn, được dùng thường xuyên, ta có thể cất giữ vào trong file dữ liệu định dạng `/etc/format.dat`.

Bây giờ ta phải tạo hệ thống file trên đĩa vừa định dạng xong. Lệnh được dùng cho mục đích này là:

```
newfs [-Nv] [mkfs-options] raw-device
```

với:

**N**

Chỉ liệt kê các thông tin của hệ thống file mà không làm lại hệ thống file đó.

**v**

Vừa tạo hệ thống file mới, vừa in thông báo trên màn hình.

**mkfs-options**

một số tùy chọn hay dùng

**a** *apc*

Số block dự trữ trên mỗi sector để thay thế các block hỏng. Mặc định là 0.

**b** *size*

Kích thước của block (4096, 8192). Mặc định 8192.

**f** *fragsize*

Kích thước nhỏ nhất của đơn vị lưu giữ thông tin. Mặc định 1024. Nếu block có kích thước 8192 byte, fragsize có thể là 512, 1024, 2448 và 4096 byte.

**s** *size*

Kích thước hệ thống file theo đơn vị sector. Mặc định: toàn partition.

**t** *ntrack*

Số rãnh trên một cylinder. Mặc định lấy từ nhãn của đĩa.

Chú ý quan trọng: Nên chép lại các thông số mà **newfs** -v thông báo như dung lượng, số sector, cylinder, track và nhất là danh sách các superblock để khôi

phục với lệnh **fsck** -b sẽ được trình bày ở phần sau, nếu chẳng may sau này ta gặp phải sự cố với đĩa cứng.

Trong ví dụ trên, ta phải lần lượt thực hiện tạo các hệ thống file, kiểm tra chúng và mount vào hệ thống UNIX (có thể thay lệnh **mount** bằng những sửa đổi tương ứng trong `/etc/mnttab`):

```
newfs /dev/rdisk/c0t2d0s3
fsck -y /dev/rdisk/c0t2d0s3
newfs /dev/rdisk/c0t2d0s4
fsck -y /dev/rdisk/c0t2d0s4
newfs /dev/rdisk/c0t2d0s5
fsck -y /dev/rdisk/c0t2d0s5
mkdir /export /data /tool
mount -F ufs /dev/rdisk/c0t2d0s3 /export
mount -F ufs /dev/rdisk/c0t2d0s4 /data
mount -F ufs /dev/rdisk/c0t2d0s5 /tool
```

Trong Solaris, lệnh **mkfs** cũng tương tự như **newfs** của System V nói chung.

Một thao tác quan trọng nữa là tạo file thiết bị cho phần cứng bằng lệnh:

```
/usr/sbin/mknod name [b | c major minor] [p]
```

### XIII. 4. Thêm ổ CD

Các ổ đĩa CD (thường là ổ chỉ đọc CD-ROM) trong Solaris thường có ID6 và được điều khiển bởi daemon **vold** giống như đối với ổ đĩa mềm. Các đĩa CD-ROM và đĩa mềm, nếu không có nhãn, đều được gán tên *unnamed\_cdrom* hoặc *unnamed\_floppy*. Cấu hình ổ đĩa CD trong hệ thống UNIX (Solaris) được quy định bởi dòng vào sau trong file cấu hình `/etc/vold.conf`:

```
use cdrom drive /dev/rdisk/c0t6 dev_cdrom.so cdrom%d
```

Có thể thêm ổ CD bằng tay, nếu việc sử dụng ổ CD chỉ là nhất thời (ví dụ, chỉ dùng để cài đặt phần mềm):

```
mount -F hsfs /dev/rdisk/c0t6d0s0 /cdrom (hay /mnt)
```

#### ⇒ Nghe nhạc từ CD trên máy chủ UNIX:

Trong thực tế triển khai mạng UNIX ở nhiều nơi, không ít lần tác giả đã gặp những thắc mắc đại loại như: "Trong mạng chỉ có mỗi máy chủ có ổ CD, mà hầu như bỏ không. Làm sao tôi có thể dùng để nghe nhạc được đây?". Có rất nhiều các tiện ích dạng shareware cho phép làm được việc này, trong đó tốt nhất là các phần mềm *WorkMan* và *xcdplayer*. Solaris cung cấp cách kết hợp WorkMan vào trong daemon **bold**. Để hệ thống tự động chơi nhạc mỗi khi tìm thấy đĩa CD nhạc được đưa vào ổ, bạn thêm dòng sau vào file cấu hình */etc/rmmountconf*:

```
action cdrom action_workman.so path-to-workman- \
 executable
```

Solaris cung cấp công cụ **audiotool** [-p] [-d device] [audio-file ...] trong OpenWin để xử lý các dữ liệu âm thanh. Có thể dùng bảng điều khiển âm thanh nhờ lệnh **audiocontrol** [-d device] cũng trong OpenWin. SunOS có tiện ích **cdplayer** như là một tùy chọn trong gói chương trình trò chơi.

Linux, trong chế độ giao diện đồ họa, có rất nhiều các chương trình dùng để ghi CD, DVD, để truyền nhận, xử lý và đọc các loại đĩa âm thanh và phim, v.v..

Trước khi định tìm kiếm một cái gì đó đại loại như WorkMan hay nghe nhạc trong chế độ OpenWin, ít nhất bạn hãy thử tìm hiểu xem, máy chủ của bạn có hỗ trợ phần cứng (chẳng hạn, có sound card) cho thú vui ấy không?

### XIII. 5. Thêm thiết bị cổng nối tiếp (modem)

Các thiết bị cổng nối tiếp (terminal) thường hay dùng là các modem, router, v.v.. Việc thiết lập chế độ làm việc hợp lý cho chúng ảnh hưởng quyết định không những

tới hoạt động của máy mà còn thay đổi hẳn chất lượng của mạng mà chúng tham gia vào. Cấu hình của chúng thường được giữ ở các file:

❖ /etc/inittab

Cấu hình khởi động của hệ thống.

❖ /etc/gettydefs

Xác định đường truyền terminal.

❖ /usr/share/lib/terminfo/?/\*

Cơ sở dữ liệu các loại cấu hình terminal.

Đây là công việc khá phức tạp của người quản trị và được thực hiện theo các cách thức khác nhau, tùy vào hệ điều hành bạn đang dùng là thuộc bản UNIX nào. Dưới đây là các điểm cơ bản của *Phương tiện dịch vụ truy cập* (Service Access Facility - SAF), tiến trình chính chịu trách nhiệm điều khiển các thiết bị cổng nối tiếp trong Solaris.

### *XIII. 5. a. Cấu trúc của SAF*

▪ *• Hoạt động của RAC*

Daemon **sac** được khởi động bằng dòng lệnh tương tự như sau trong file `/etc/inittab`, với thông số `t` chỉ khoảng thời gian tính theo giây nó quét qua các cổng:

```
sc:234:respawn:/usr/lib/saf/sac -t 300
```

▪ *Port Monitors*

Daemon **sac** điều khiển các bộ giám sát cổng (*Port Monitor*) khác nhau. Bộ giám sát cổng là các tiến trình chịu trách nhiệm theo dõi một hay nhiều cổng và gán các yêu cầu nhận được từ các cổng tới các tiến trình tương ứng của hệ thống. **sac** khởi động tất cả các bộ giám sát cổng được liệt kê trong danh sách ở file cấu hình `/etc/saf/_sactab`.

Hiện nay các bộ giám sát cổng được chia làm hai loại: **ftt** - chịu trách nhiệm đối với các đường truyền terminal, và **listen** - về điều khiển các dịch vụ mạng chung.

▪ *Kết nối lại các dịch vụ tại chỗ*

Các bộ giám sát cổng nối các yêu cầu tới các dịch vụ hệ thống tại chỗ. Do có khả năng giám sát nhiều hơn một cổng cùng loại, ta có thể gọi các phiên - *instant* - của một bộ giám sát cổng ở cùng thời điểm cho mỗi một cổng.

### **XIII. 5. b. Các lệnh cấu hình SAF và các bộ giám sát cổng**

❖ **sacadm**

Dùng để cấu hình SAF ở mức cao, nhằm các mục đích: thêm, bớt, bắt đầu, kết thúc, mở, đóng các bộ giám sát cổng. Cú pháp được quy định trong các dạng dưới đây:

```
sacadm -a -p pmtag -t type -c cmd -v ver [-f dx] [-n
count] [-y comment] [-z script]
sacadm -r -p pmtag
sacadm -s -p pmtag
sacadm -e -p pmtag
sacadm -d -p pmtag
sacadm -l [-p pmtag | -t type]
sacadm -L [-p pmtag | -t type]
sacadm -G [-p pmtag | -z script]
sacadm -x [-p pmtag]
```

Các tùy chọn phổ biến hay được dùng:

**a**

Thêm bộ giám sát cổng.

**c** *cmd*

Chạy lệnh *cmd* khi khởi động bộ giám sát cổng.

**d/e**

Đóng/mở bộ giám sát cổng.



**p** *pmtag*

Xác định mục tiêu cụ thể của bộ giám sát cổng (PMTAG).

**v**

Chỉ phiên bản (trả về giá trị **ttysadm** -V)

**y**

Mô tả cho dòng vào của */etc/\_pmtab*.

**t** *type*

Loại bộ giám sát cổng (ttymon, listen)

**n** *count*

Số lần khởi động lại bộ giám sát cổng nếu nó bị chết.

❖ **pmadm**

Dùng để cấu hình các dịch vụ của các bộ giám sát cổng cho từng cổng. Cú pháp:

```
pmadm -a -p pmtag -t type -s svctag -i id -m pmspecific
 -V ver [-f xu] [-y comment] [-z script]
pmadm -r -p pmtag -s svctag
pmadm -e -p pmtag -s svctag
pmadm -d -p pmtag -s svctag
pmadm -l [-t type | -p pmtag] [-s svctag]
pmadm -L [-t type | -p pmtag] [-s svctag]
pmadm -g -p pmtag -s svctag [-z script]
pmadm -a -s svctag -t type -z script
```

Các tùy chọn:

**a**

Thêm dịch vụ.

**d/e**

Đóng/mở dịch vụ.

**p** *pmtag*

Xác định PMTAG, ví dụ: zsmo.

**i** *id*

Chạy dịch vụ cho người dùng có id.

**f u**

Tạo dòng vào trong file `utmp` cho cổng (lệnh `login` yêu cầu)

**v ~ttyadm ...`**

Chỉ phiên bản (xác định và trả giá trị `tttyadm -V`)

**m pmspecific**

Mô tả cho dòng vào xác định đặc trưng dữ liệu.

### ❖ tttyadm

Trợ giúp các dạng vào cho lệnh `pmaadm` đối với cổng nối tiếp. Xuất các thông tin của bộ giám sát cổng. Cú pháp của lệnh như sau:

```
tttyadm [-b] [-c] [-h] [-I] [-r count] [-t timeout] -s svctag
 [-i mesg] [-m modules] [-p prompt] [-S y|n] [-T
 termtypel] -d device -l ttylabel -s service
tttyadm -V
```

Các tùy chọn:

**b**

Hai chiều.

**h**

Hang up

**I**

Chỉ bắt đầu.

**d device**

Đường dẫn đầy đủ của file thiết bị cho cổng TTY.

**p pmtag**

Xác định PMTAG, ví dụ: `zsmon`.

**i message**

Thông báo sẽ hiện khi ở trạng thái đóng.

**l ttylabel**

Chỉ `ttylabel` trong `/etc/ttydefs`.

**p prompt**

Chỉ nội dung đầu nhắc, ví dụ: `"login :"`

**r** *count*

*count=0/count: ttmon* chờ đến khi nhận được bất kỳ ký tự nào/ nhận được *count* dòng.

#### ❖ sttydefs

Tạo các dòng vào trong */etc/ttydefs* để mô tả các đặc trưng của các đường truyền terminal. Cú pháp:

**sttydefs** [-a] [-l] [-r]

Các tùy chọn:

**a**

Thêm dòng vào. Có các tùy chọn thứ cấp: **b** - tự động xác định tốc độ (autobaud); **i** - cờ bắt đầu; **n** - nhấn tiếp; **f** - cờ kết thúc.

**l**

Liệt kê các dòng vào.

**r**

Bớt dòng vào.

#### ❖ nlsadmin

Quản lý các dịch vụ **listen** mạng. Trợ giúp các dạng vào cho lệnh **pmadm** đối với dịch vụ in từ xa. Cú pháp (một trong nhiều dạng của lệnh này):

**sttydefs** [option] -N *port\_monitor\_tag*

## XIII. 6. Cách khắc phục một số lỗi phần cứng thường gặp

### *XIII. 6. a. Khi nào máy gặp sự cố*

Các nguyên nhân gây lỗi rất nhiều, muôn hình muôn vẻ. Có thể liệt kê các nguyên nhân chung nhất khiến máy bị trục trặc như sau:

- Lỗi thiết bị phần cứng: điều khiển các thiết bị SCSI (DPT card,...), board mạch chính, RAM, nguồn điện, đĩa cứng,...
- Các lỗi không thể khắc phục được, thỉnh thoảng lại xuất hiện (CMOS, các sector lỗi trên đĩa cứng, RAM,...)

- Các vấn đề về môi trường xung quanh (độ ẩm cao, hóa chất, nhiệt độ thay đổi bất thường, nguồn điện không ổn định, chuột và côn trùng cắn, làm tổ, thải chất bẩn, ...). Những sự cố kiểu này đặc biệt hay gặp phải ở nước ta.
- Các vấn đề vào, ra; kể cả các điều kiện khiến chức năng của thiết bị giảm sút.
- Các vấn đề về phần mềm, nhất là các phần mềm ứng dụng hay của các hãng thứ ba.
- Các lỗi về khả năng chịu tải (system full, swap bị dùng hết...)
- Các lỗi cấu hình. Thường hay gặp lúc mới cài đặt, thay đổi thiết bị, ...

### ***XIII. 6. b. Khi bị lỗi "System file full"***

Khi gặp lỗi này, người quản trị phải:

- Dùng lệnh **kill** xóa tiến trình hiện hành.
- Tìm file có tên **core** và xóa nó đi.
- Vào **/var/adm**, tìm và xóa file có tên **wtmp**.
- Vẫn trong **/var/adm**, tìm file **messages**.
- Thực hiện lệnh làm rỗng file **messages**:

```
cp /dev/null /var/adm/messages
```

- Khởi động lại máy.

### ***XIII. 6. c. Đối với các lỗi của đĩa cứng***

- *“Khi có thông báo lỗi "BAD SUPERBLOCK #####"*

Lỗi này xảy ra khi superblock có số ##### của đĩa cứng bị hỏng vật lý hoặc logic. Để sửa chữa, bạn phải có quyền **root**. Có thể sửa chữa theo một trong hai cách với các bước như sau đối với Solaris:

#### ***Cách thứ nhất:***

- Khởi động máy ở chế độ một người dùng: khi bắt đầu khởi động, gõ **b -s** hoặc tắt máy bằng lệnh **shutdown -y -g0 -is**. Sau đó đăng nhập vào **root** (dấu nhắc "#").

- Ghi lại thông tin về bảng partition bằng lệnh **# format**, lệnh con: **partition => print**.
- Nhận biết tên file thiết bị **/dev/dsk/device-name** gắn với partition đĩa bị hỏng:

```
df -lt
```

- Xem và ghi lại các bản lưu superblock của hệ thống file, dùng lệnh:

```
newfs -N /dev/dsk/device-name
```

- Thay thế các superblock bị hỏng bằng các backup superblock NNNN ở vị trí gần nhất:

```
fsck -o b=NNNN /dev/dsk/device-name
```

### Cách thức hai:

- Nhận biết địa chỉ của các superblock bị hỏng (ghi lại):

```
fsck -y
```

- Dùng lệnh

```
format
```

- Dùng lệnh con **repair**, nếu cần, định dạng lại bằng lệnh con định dạng rồi **repair**.
- Có thể kiểm tra lại danh sách các vị trí bị lỗi trên đĩa bằng lệnh con **defect**.

- *Khi có thông báo lỗi "COMMAND NOT FOUND"*

- Nên kiểm tra lại xem có đường dẫn tới thư mục đặt lệnh chưa? Sau mỗi khi thay đổi, sắp xếp lại vị trí các thư mục, nếu không cẩn thận, lỗi này thường hay xảy ra với các chương trình ứng dụng. Dùng lệnh **# env** hoặc **\$ env** (tùy theo bạn đang là root hay người dùng bình thường) để xem lại các biến môi trường.

- Nếu chưa có đường dẫn, bạn phải đặt. Tốt nhất là đặt đường dẫn trong file `.profile` tương ứng theo cách thức đã quen thuộc. Trong Solaris, đường dẫn tới các thư mục đặt lệnh và thư viện trước hết là: `/usr/bin;`  
`/usr/sbin;` `/usr/openwin/bin;` `/opt/SUNWpcnfs/bin;`  
`/usr/openwin/lib;` `/etc/init.d, ...`

### ***XIII. 6. d. Đối với các thiết bị vào / ra (bàn phím, màn hình, chuột)***

Các lỗi hay gặp nhất thường xảy ra với chế độ của điều khiển màn hình như sai loại, sai tần số quét,... làm cho bạn không thể thấy được gì trong chế độ đồ họa OpenWin, mặc dù trong chế độ dòng lệnh máy vẫn hoạt động bình thường. Bạn nên:

- Khởi động lại máy. Đăng nhập với quyền root.
- Gõ những dòng lệnh sau:

```
kdmconfig -u
kdmconfig -c
```

Tùy chọn `u` gỡ bỏ các thông số cấu hình không đúng, và khi dùng tùy chọn `c` bạn đã sửa các dòng vào trong các file `/etc/defaultkb` và `/etc/openwin/server/etc/OWconfig` và cả trong `bootparams`.

Bạn sẽ gặp phải vấn đề tương tự khi máy tính hay chính bạn nhầm, lẫn lộn giữa các loại chuột vi tính PS/2 và Serial (cổng nối tiếp). Các lệnh và file liên quan đến việc xác định lại thông số cho chuột, bàn phím, vẫn là **kdmconfig**, `/etc/openwin/server/etc/OWconfig`, `/etc/defaultkb`. Đặc biệt, nếu dùng chuột Serial, bạn phải mở thêm cổng cho kết nối mạng qua modem bằng cách sửa dòng vào liên quan tới cổng đó (`/dev/tty00` hoặc `/dev/tty01`) trong file `/platform/kernel/x86/driver/OWconfig`. Bạn nên dùng lệnh **admintool** trong OpenWin để đưa cổng nối tiếp nối với chuột về các giá trị mặc định. Các bản hệ điều hành mới nhất, được viết cho các cấu hình phần cứng mới nhất của máy tính, còn hỗ trợ các thiết bị ngoại vi, trước hết là chuột vi tính, nối qua các cổng USB.

Nhiều khi lệnh **touch/reconfigure** sau khi khởi động lại máy cũng giúp bạn đưa máy trở lại hoạt động bình thường.

### ***XIII. 6. e. Các lỗi với thiết bị đầu cuối***

- *Lỗi với các thiết bị mới*

- Kiểm tra xem mỗi nối vào ổ cắm RS-232 đã có tiếp xúc tốt chưa?
- Đường cáp có bị trục trặc gì không?
- Màn hình, điều chỉnh độ chói, độ tương phản làm việc tốt?
- Tiến trình **getty** chạy không? Kiểm tra bằng lệnh **ps | grep -c "stty"** xem **getty** có chạy đúng số lần không.
- Có dùng đúng loại cáp không? Nếu không, lệnh sau sẽ làm treo thiết bị:

```
cat file > /dev/ttyn
```

Dùng lệnh **kill** để xóa tiến trình bị treo.

- *Khi thiết bị bị treo*

- Không quên kiểm tra cáp và chất lượng nối, kể cả ở các ổ cắm RS-232.
- Nếu biết người dùng vừa làm động tác gì gây ra lỗi, người quản lý cố gắng **undo** thao tác đó.
- Người dùng ấn **^S**, hoặc phím nào làm dừng xuất ra. Sau đó ấn **^Q** và các phím khác để xem cái gì xuất hiện trên màn hình.
- Cố dùng lệnh **reset**. Nếu không được, vào các lệnh theo thứ tự: **^J → reset → ^J**.
- Tắt máy đi rồi khởi động lại.
- Sang thiết bị đầu cuối khác, xóa tiến trình gây lỗi bằng lệnh **kill** với các tùy chọn khác nhau. Có thể xem trong **/usr/include/sys/signal.h** để biết loại tín hiệu trong tiến trình cần xóa.
- Nếu vẫn không được, cố xóa shell login của người dùng đó. Rồi tắt điện đi, bật lại.

---

### ***XIII. 6. f. Các lỗi với card mạng và mạng Ethernet***

- "le0 no carier - transceiver cable problem ? ". Kiểm tra lại cáp mạng xem nối đã tốt chưa. Đặc biệt kiểm tra và thay thử các bộ chuyển đổi. Kiểm tra cả khả năng có những lỗi phần cứng khác như ở router, bridge... Đối với card mạng 3C509 file thiết bị tương ứng trong Solaris cho x86 là elx0.
- "le0 ethernet jammed". Kiểm tra xem các đầu cáp có *terminator* tốt chưa, nhất là đối với mạng theo cấu trúc kiểu bus (xem chương XIV).
- "unknown host". Không thể nào nhận ra địa chỉ IP của máy từ xa. Thử lại địa chỉ IP. Nếu không có kết quả, kiểm tra lại việc phân giải địa chỉ theo tên trên máy của bạn.
- "network unreachable". Dịch vụ phân đường ở máy bạn không làm việc. Dùng lệnh **netstat -rn** để xem lại bảng phân đường và nếu cần, đặt lại router mặc định.
- "no answer " hay "connection timed out " hay "cannot connect". Máy của bạn tìm được đường tới máy ở xa, nhưng không nhận được trả lời. Có thể mạng bị tắt, máy từ xa không tìm được đường tới máy của bạn, hay cả hai máy đều bị cấu hình không đúng. Một nguyên nhân hay được nhận thấy là máy đích từ xa mà bạn muốn kết nối tới thực ra đã kết thúc hoạt động, ở trạng thái tắt điện, mặc dù các thông số cấu hình vẫn bình thường. Kiểm tra lại cấu hình mạng bằng lệnh **ifconfig**.



## **XIV. Khái niệm về mạng và quản trị mạng**

### **XIV. 1. Nhiệm vụ của người quản trị mạng**

1. Thực hiện các nhiệm vụ của người quản trị hệ thống.
2. Kiểm tra trạng thái của mạng.
3. Thay đổi các file server, các trạm làm việc, tạo thêm hoặc bớt đi các thư mục.
4. Thay đổi danh sách và quyền của người dùng trong toàn mạng.
5. Phân đường dữ liệu.
6. Đảm bảo truyền thông và bảo mật khi giao tiếp với các mạng bên ngoài.

Dưới đây, chúng ta sẽ xem xét cụ thể các kiến thức liên quan và một số công việc thuộc chức năng của người quản trị mạng.

### **XIV. 2. Khái niệm về mô hình mạng**

Theo cách hiểu thông thường, mạng máy tính là một tập hợp các máy được kết nối với nhau, có chia sẻ tài nguyên chung. Nếu các máy tính được bố trí tại một địa điểm không xa nhau lắm, chúng có mặt trong mạng cục bộ - mạng *Local Area Network* (LAN). Nếu phạm vi của mạng vào khoảng 10 km, nó được gọi tên là mạng diện cụm - *Metropolitan Area Network* (MAN) - một khái niệm tương đối ít phổ biến hơn. Tập hợp các mạng LAN ở các vị trí địa lý khác nhau có thể được tổ chức thành mạng diện rộng - *Wide Area Network* (WAN), ... Ta thấy có khá nhiều tên gọi và loại mạng, vậy cái gì là đặc trưng nhất cho các mạng máy tính? Cơ quan Tiêu chuẩn quốc tế về kết nối hệ thống mở đưa ra mô hình cấu trúc mạng gồm bảy lớp (ISO/OSI), hiện được thống nhất coi là mô hình chung cho mạng máy tính. Tuy nhiên, đối với UNIX còn tồn tại một mô hình mạng nữa. Bảng sau cho thấy so sánh nội dung giữa các lớp mạng của các mô hình thực tế được áp dụng cho UNIX và mô hình ISO/OSI:

| Thực tế    | ISO/OSI          | Chức năng                                                                                                                                       |
|------------|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| Ứng dụng   | Ứng dụng         | Cung cấp các dịch vụ mạng: <b>mail, ftp, telnet</b> , NFS, DNS, <b>www</b> (World Wide Web), <b>yp</b> .                                        |
|            | Trình bày        | Các dịch vụ chuyển đổi: nén văn bản, chuyển mã, ... để máy khác nhận biết                                                                       |
|            | Phiên làm việc   | Gọi thủ tục từ xa (Remote Procedure Call – <b>rpc</b> ); các chương trình tạo kết nối theo tên qua socket, phục hồi kết nối, ...                |
| Vận chuyển | Vận chuyển       | Theo các giao thức Transmission Control Protocol (TCP), User Datagram Protocol (UDP).                                                           |
| IP         | Mạng             | Giao thức Internet Protocol (IP), nối mạng con với Internet, chia/hợp dữ liệu, phân đường và đệm, bắt đầu và kết thúc các kết nối giữa các máy. |
| Vật lý     | Liên kết dữ liệu | Xác định khung dữ liệu, kiểm tra và sửa sai gói dữ liệu, xác định khả năng chia sẻ đường truyền.                                                |
|            | Vật lý           | Cung cấp các kết nối vật lý, xác định thủ tục bắt đầu và kết thúc phiên, chuyển gói.                                                            |

Dữ liệu trao đổi giữa các lớp theo các đơn vị được tổ chức theo khung dữ liệu, có đánh dấu đầu, cuối. Trong trình tự bình thường, tại mỗi lớp, thông tin địa chỉ từ lớp trước được kiểm tra và gỡ bỏ, dữ liệu được chuyển tiếp tới lớp sau. Từ lớp trên cùng, thông tin được chia thành các gói nhỏ và hợp thành các đơn vị chuyển tiếp cực đại (Maximum Transmission Unit - MTU) có kích thước cho đến 1500 byte tại đầu phát. Cấu trúc một khung dữ liệu tại lớp Vật lý (chuẩn 802.3) được trình bày ở hàng trên của bảng như sau:

|                                |                 |                  |                          |                                |                    |                    |             |
|--------------------------------|-----------------|------------------|--------------------------|--------------------------------|--------------------|--------------------|-------------|
| Pre-<br>amble                  | Địa chỉ<br>đích | Địa chỉ<br>nguồn | Kích<br>thước dữ<br>liệu | Điều khiển<br>kết nối<br>logic | Dữ liệu<br>cực đại | Điện dây<br>cho đủ | CRC         |
| 8 byte                         | 2-6 byte        | 2-6 byte         | 2 byte                   |                                | 1500 byte          | 1500<br>byte       | 4 byte      |
| Điều khiển truy cập môi trường |                 |                  |                          | Phần dữ liệu                   |                    |                    | Soát<br>lỗi |

Trong trường Preamble có dấu bắt đầu của khung dữ liệu, trường thứ tư cho biết kích thước của ba trường tiếp theo. Trường Điều khiển kết nối logic (LLC – xem phần

XIV. 3. a) xác định loại giao thức và loại mạng cụ thể (LAN, Ethernet, TCP/IP) được dùng.

Dữ liệu được truyền đi trong mạng với các khung dữ liệu như trên từ lớp Vật lý của máy nguồn tới lớp Vật lý của máy đích. Tại máy đích, dữ liệu lại được xử lý theo trình tự ngược lại khi lần lượt qua các lớp của mô hình mạng, được tái hợp và trình bày bằng phần mềm ở lớp Ứng dụng.

### **XIV. 3. Khái niệm về cấu trúc mạng**

#### **XIV. 3. a. LAN**

##### ▪ *Khái niệm về LAN*

LAN là thuật ngữ chỉ mạng cục bộ. Có nhiều định nghĩa về LAN, nhưng những điểm chung nhất bao gồm:

- Chiều dài nối kết lớn nhất giữa các thiết bị của người dùng thường nằm trong phạm vi từ vài trăm đến vài nghìn mét.
- Hầu hết các dữ liệu (có thể gồm cả âm thanh và hình ảnh) truyền nhận đều được thực hiện giữa máy tính thực hiện chức năng server và các trạm làm việc của người dùng.
- Khả năng truyền đạt thông tin của mạng LAN thường cao hơn so với mạng WAN. Tốc độ truyền đạt dữ liệu trong LAN thường từ 1 đến 20 Mbit/s, cao nhất hiện nay là 100 Mbit/s. (Đơn vị bit/s còn thường được viết là bps - bit per second).
- Mạng LAN thường được một tổ chức cụ thể sở hữu và quản lý.
- Khả năng mắc lỗi thường rất ít, cỡ 1 lỗi trên  $10^9$  bit được truyền nhận.

Theo quan điểm mô hình mạng OSI/ISO, mạng LAN sẽ chiếm hai lớp thấp nhất là lớp Vật lý và lớp Liên kết dữ liệu. Lớp Vật lý thực hiện các chức năng kết nối các thiết bị vật lý: card mạng, bộ lặp (repeater), v.v., xác định các thủ tục bắt đầu và kết thúc việc truyền nhận dữ liệu. Đối với mạng LAN, lớp Vật lý được chia thành các phần Giao diện đơn vị kết nối (*Attachment Unit Interface* - AUI), Bộ phận kết nối trung gian (*Medium Attachment Unit* - MAU), Kết nối vật lý trung gian (*Physical Medium Attachment* - PMA), Giao diện phụ thuộc trung gian (*Medium Dependent*

*Interface* - MDI). AUI cung cấp kết nối giữa lớp Vật lý của thiết bị người dùng với môi trường LAN, gồm các phần tử như đầu cắm RJ-45, ... MAU thực chất chứa cả PMA và MDI, để điều khiển kết nối của thiết bị đầu cuối cho dữ liệu với cáp mạng, cung cấp các dịch vụ truyền nhận dữ liệu, kiểm tra, phát hiện các sự cố và theo dõi chất lượng thông tin được truyền nhận. PMA chứa mạch điện hỗ trợ chức năng của MAU. MDI cung cấp các giao diện cơ học và điện giữa môi trường và PMA. Lớp Liên kết dữ liệu đối với mạng LAN bao gồm hai phần: Điều khiển truy nhập môi trường (*Media Access Control* - MAC) và Điều khiển kết nối logic (*Logical Link Control* - LLC). MAC chịu trách nhiệm điều khiển mức độ truyền dữ liệu của LAN: xác định xem ở thời điểm nào LAN rỗi hay bận, có phải truyền lại dữ liệu hay không, v.v.. LLC cung cấp giao diện giữa LAN và các lớp mạng của người dùng với các dịch vụ. Việc cấu hình LLC cho đúng quyết định tới tốc độ và chất lượng của việc trao đổi thông tin.

LAN là một mạng máy tính, nhưng bản thân LAN lại không có lớp Mạng theo mô hình OSI/ISO. Theo mô hình đó lớp Mạng được dùng để hỗ trợ việc phân đường và các thao tác giao diện. Nhưng hầu hết các mạng LAN đều thuộc dạng mạng *broadcast*, có nghĩa là dữ liệu được truyền đi và phân phát rộng rãi cho mọi đối tượng mạng, nên không dùng các kỹ thuật chuyển mạch, do đó, khái niệm phân đường, hay phân đường, không được áp dụng cho các mạng LAN. Thêm nữa, chức năng hỗ trợ các thao tác giao diện của lớp Mạng phải đảm bảo có thể thay đổi *chất lượng dịch vụ* của kết nối giữa người dùng và mạng, trong khi mạng LAN chỉ có các giao diện đơn giản định sẵn, không có tùy chọn. Vì những lý do trên, người ta nói, LAN không dùng lớp Mạng OSI/ISO, hay nếu có thì lớp Mạng đó hết sức đơn giản, chỉ được cải thiện bằng giao thức IP.

LAN bao gồm bốn thành phần chính sau để hỗ trợ việc truyền nhận dữ liệu giữa những người dùng cuối:

- *Kênh truyền dẫn*. Là môi trường truyền dẫn dữ liệu trong mạng. Có thể là cáp đồng trục béo hay gầy (xem *Phần cứng giao tiếp mạng*), cáp xoắn hoặc sợi thủy tinh quang học, gọi tắt là cáp quang.
- *Các giao diện vật lý* có thể là các anten sóng cực ngắn, các đi ốt phát quang hồng ngoại và đi ốt laser, các bộ lặp, bộ đệm cho luồng dữ liệu, ....

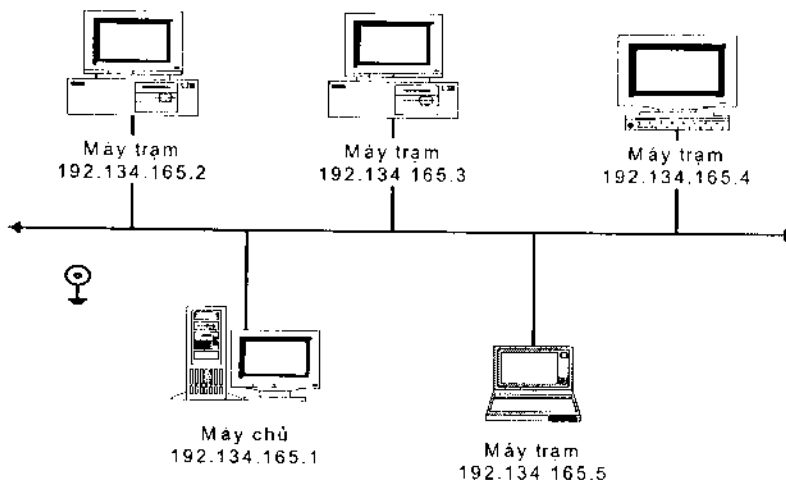


- *Giao thức mạng* điều khiển LAN và cung cấp khả năng vào mạng cho những người dùng cuối.
- *Các trạm làm việc của người dùng*, nói đơn giản là các máy tính hay thiết bị đầu cuối.
  - *Các loại mạng 'P.A.V'*

Theo hình thái, LAN được chia làm ba loại chính sau:

### 1. Bus

Các trạm làm việc được nối song song với nhau trên một trục chính được gọi là *backbone* (xương sống), như trên H. XIV. 1. Hai đầu của backbone phải được chặn bởi một dụng cụ đặc biệt, *terminator*, để giới hạn phạm vi dữ liệu có thể lan truyền trong mạng. Loại mạng này rất đơn giản, linh hoạt, chi phí lắp đặt thấp và dễ dàng mở rộng khi tăng số máy trong mạng. Backbone có thể rẽ nhánh để tạo thành mạng phức tạp hơn. Các mạng là các dây đồng trục có phối hợp trở kháng tốt và tính



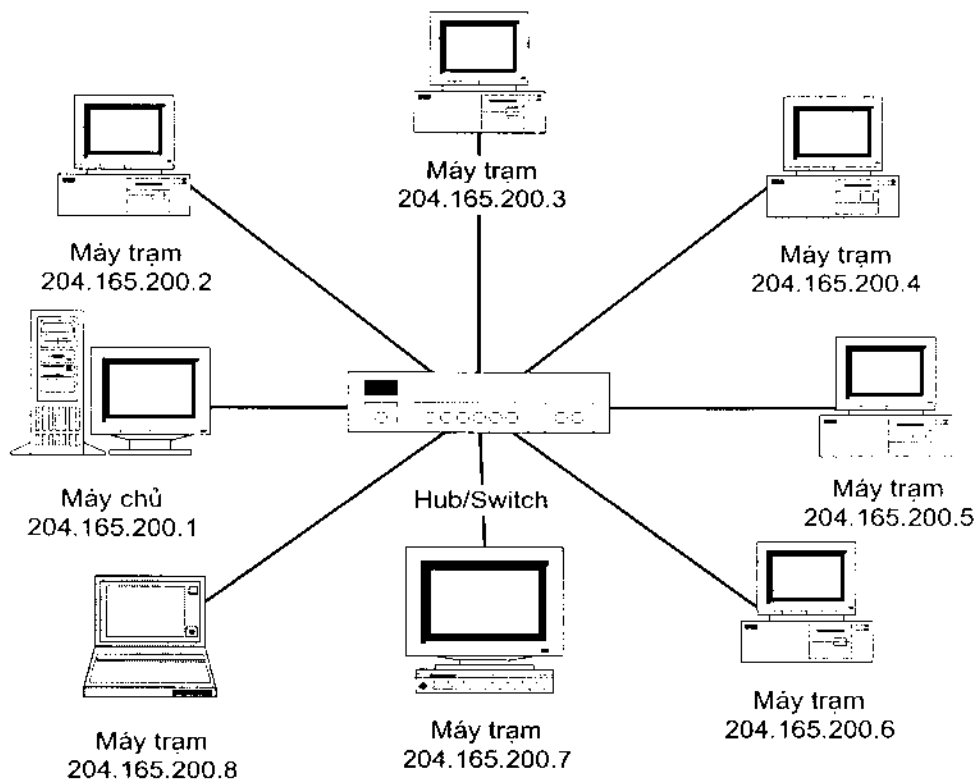
H. XIV. 1. Sơ đồ mạng LAN cấu trúc bus

chống nhiễu cao. Tuy nhiên, mạng bus làm việc không tốt trong điều kiện lưu lượng thông tin lớn và có nhược điểm lớn nhất dễ nhận thấy là độ tin cậy thấp - chỉ cần có một máy bị trục trặc ở điểm nối mạng là cả mạng ngừng hoạt động. Hơn nữa, một khi phát triển backbone đến kích thước tương đối lớn, việc đảm bảo có tiếp xúc tốt

của các mối nối, cũng như sự cách điện nối chung, trở thành các vấn đề lớn phải chú ý thường xuyên.

## 2. Star

Loại mạng LAN hình sao (star) hiện là loại mạng được dùng phổ biến tại các doanh nghiệp Việt Nam. Trong mạng loại này, các trạm làm việc đều được nối về trung tâm, tới một thiết bị chuyển mạch đặc biệt, có thể là loại thụ động - *hub*, hay tích cực - *switch* (H. XIV. 2). Dây nối thường thuộc loại cáp xoắn gồm bốn đôi dây, nhưng trong các mạng có yêu cầu kỹ thuật cao có thể có một số cổng dùng cáp quang.



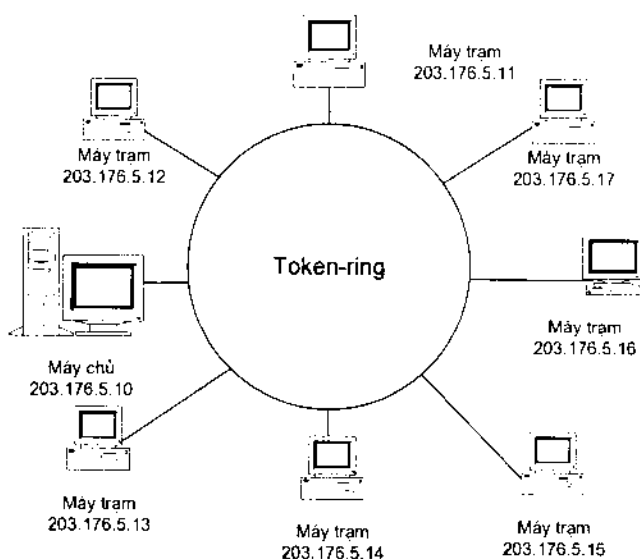
H. XIV. 2. Sơ đồ mạng LAN cấu trúc hình sao (star).

Mạng hình sao cung cấp một môi trường làm việc ổn định, có độ tin cậy cao, vì nếu có một trạm làm việc nào đó bị hỏng thì ảnh hưởng lên các trạm khác là cực

tiểu. Việc định vị trạm có trục trặc cũng rất dễ dàng. Nếu muốn phát triển thêm số lượng trạm làm việc, ta chỉ cần tạo thêm các kết nối điểm - điểm với hub hoặc switch. Có các loại hub và switch có thể đấu nối trực tiếp với nhau (gọi là *xếp chồng* - *stackable*) để mở rộng số cổng trong mạng. Tuy nhiên, việc mở rộng mạng gặp đôi chút khó khăn, vì thường phải đi với việc cải tạo lại mạng và đi thêm các dây cáp mạng theo tòa nhà làm việc. Đó là chưa kể đến khả năng xếp chồng bị giới hạn, cũng như chi phí tương đối cao để mua sắm hub, đặc biệt là switch.

### 3. Token ring

Trong mạng mạch vòng (Token ring) môi trường truyền dẫn được thực hiện dưới dạng hình tròn; mỗi trạm làm việc được nối với hai trạm khác gần nhất. Các gói thông tin sẽ quay theo vòng tròn trong mạng trước khi đến được máy đích (H. XIV.3). Kiểu mạng này ít được áp dụng trong thực tế tại Việt Nam và chỉ có công dụng rõ rệt trong các mạng có áp dụng các kỹ thuật đặc biệt để truy nhập mạng với các lưu lượng dữ liệu theo yêu cầu. Giống như trong trường hợp mạng bus, nếu điểm nối mạng của một trạm làm việc trong mạng bị trục trặc thì cả mạng cũng sẽ bị ngừng hoạt động.



H. XIV. 3. Sơ đồ mạng LAN cấu trúc Token ring.

Theo chức năng, một trong những loại mạng LAN dùng họ giao thức TCP/IP thông dụng nhất được gọi là mạng *Ethernet*. Phần lớn các kiến thức trình bày trong các phần về mạng sau này trong tài liệu này có đối tượng chính là mạng Ethernet.

Từ tháng 2 năm 1980 Viện Nghiên cứu kỹ thuật điện và điện tử (Institute of Electrical and Electronic Engineers - IEEE) Mỹ đưa ra các tiêu chuẩn IEEE áp dụng cho mạng LAN. Từ đó bắt đầu tên gọi các chuẩn IEEE với các số 802 như được liệt kê trong bảng sau.

| Chuẩn IEEE  | Định nghĩa                                                                                     |
|-------------|------------------------------------------------------------------------------------------------|
| IEEE 802.1  | Mạng LAN dùng cầu (bridge)                                                                     |
| IEEE 802.2  | Điều khiển kết nối logic (Logical Link Control - LLC)                                          |
| IEEE 802.3  | Chuẩn của công nghệ Ethernet. Bao gồm cả 100BASE-TX, 100BASE-TF và 100BASE-T4 (Ethernet nhanh) |
| IEEE 802.4  | Mạng Token Bus chuẩn                                                                           |
| IEEE 802.5  | Mạng Token Ring chuẩn                                                                          |
| IEEE 802.6  | Mạng diện cụm (Metropolitan Area Network - MAN)                                                |
| IEEE 802.7  | Mạng tư vấn kỹ thuật băng rộng (Broadband technical advisory)                                  |
| IEEE 802.8  | Mạng tư vấn kỹ thuật cáp quang                                                                 |
| IEEE 802.9  | Mạng tích hợp dữ liệu/âm thanh (Integrated Voice/Data - IVD)                                   |
| IEEE 802.10 | Mạng LAN bảo mật                                                                               |
| IEEE 802.11 | Mạng LAN không dây (ví dụ, dùng kỹ thuật hồng ngoại, siêu cao tần)                             |
| IEEE 802.12 | 100BASE-VG (100VG-AnyLAN)                                                                      |

#### XIV. 3. b. WAN

Mạng diện rộng (WAN) là loại mạng được người ta coi rằng có cấu trúc mạng tương ứng với ba lớp thấp nhất của mô hình OSI/ISO. Lớp thấp nhất - lớp Vật lý - thường gồm các modem cho đường truyền tương tự, và các Bộ phận dịch vụ dữ liệu (*Data Services Unit* - DSU) cho các đường truyền kỹ thuật số. Lớp Liên kết dữ liệu



dùng để phát hiện lỗi và truyền lại các dữ liệu bị hỏng. Gần đây, các nhà sản xuất còn cung cấp các giao thức điều khiển liên kết dữ liệu ở mức cao (ví dụ: *Link Access Protocol Balanced* - LAPB). Lớp Mạng dùng để chuyển mạch và phân đường cho luồng dữ liệu trong mạng. Đối với mạng WAN, ta chú ý hơn tới lớp Mạng. Có bốn kỹ thuật chuyển mạch được áp dụng: chuyển mạch bằng sơ đồ (*circuit switching*), chuyển mạch bản tin (*message switching*), chuyển mạch gói (*packet switching*) và tiếp mạch phần tử (*cell relay* hay *cell switching*).

- *Chuyển mạch bằng sơ đồ* cung cấp kết nối trực tiếp giữa hai phần tử của mạng dưới dạng một đường ống mở, cho phép hai người dùng sử dụng các tiện ích có sẵn trong mạng trong điều kiện dải thông và chủ quyền cho phép. Loại chuyển mạch này thường thấy nhiều trong các mạng điện thoại. Có ba dạng kiến trúc của chuyển mạch được dùng: 1) tập trung,  $N > M$ ; 2) mở rộng,  $N < M$ ; 3) nối kết,  $N = M$ , với  $N$  là số đường vào. Trong trường hợp đơn giản nhất, chuyển mạch bằng sơ đồ được biểu diễn như mảng  $N \times M$  các đường truyền tới điểm nối. Trong các mạng phức tạp,  $N$  là số đường vào tới phần tử mạng (máy tính, terminal, v.v...) được nối đến, còn  $M$  là số đường ra của bộ phận chuyển mạch khác. Chuyển mạch bằng sơ đồ chỉ đơn giản cung cấp đường nối cho các phiên làm việc để trao đổi dữ liệu trong mạng; còn lại, việc kiểm tra lỗi, thiết lập phiên làm việc, thiết lập và điều khiển khung cho luồng dữ liệu, chọn lựa các giao thức và cách mã hóa thuộc về phạm vi trách nhiệm của người dùng. Việc kiểm soát quá tải luồng dữ liệu thường được cải tiến, bổ sung bởi các nhà cung cấp.

- *Chuyển mạch bản tin* là phương tiện kỹ thuật nhằm chuyển mạch với mục đích tránh các tắc nghẽn mạch, được áp dụng phổ biến trong những năm từ 1960 đến 1970. Vai trò chuyển mạch bản tin thường là do một máy tính đặc biệt chịu trách nhiệm nhận biết các tắc nghẽn mạch từ các máy tính và terminal được nối tới qua đường dial-up (đường thoại quay số) hay đường thuê bao riêng. Máy tính trên kiểm tra địa chỉ trong phần đầu của bản tin và chuyển mạch nó tới trạm thu hoặc tới thiết bị chuyển mạch tiếp theo để phân đường. Khác với kỹ thuật chuyển mạch trước, nội dung cơ bản của kỹ thuật này là giữ lại bản tin một thời gian ngắn, rồi chuyển đi. Do đó, lưu lượng dữ liệu trong mạch không thể được coi là có giao tiếp tức thời với người thao tác, hay còn gọi là trong thời gian thực. Ta có thể quy định các mức độ ưu tiên, cũng như đảm bảo ít có khả năng xảy ra nghẽn mạch. Chuyển mạch bản tin có thể

làm việc theo quan hệ server/slave. Tuy nhiên, kỹ thuật này có ba điểm yếu sau đây cần được chú ý. Trước hết, do làm việc theo mô hình server/slave, nếu phần tử chuyển mạch gặp trục trặc, toàn bộ mạng sẽ bị tê liệt. Do đó, nhiều tổ chức phải thiết lập thêm thiết bị chuyển mạch dự trữ cho trường hợp sự cố. Thứ hai, do tất cả các thông tin phải đi qua thiết bị chuyển mạch, chính bản thân thiết bị đó có thể trở thành một nút cổ chai tiềm năng. Ta rất thường gặp hiện tượng nghẽn mạch kiểu này đối với các hub tốc độ thấp. Cuối cùng, do cơ chế giữ lại - gửi đi, đường truyền thông không được kỹ thuật này sử dụng hết hiệu năng.

- *Chuyển mạch gói* được bắt đầu đưa vào sử dụng từ những năm 1970, và đến nay đã trở thành kỹ thuật cơ bản trong các mạng chuyển mạch dữ liệu, trước hết là trong các hệ thống bưu điện. Chuyển mạch gói phân phối xác suất các rủi ro lên nhiều thiết bị chuyển mạch, do đó, làm giảm đi các khả năng dễ bị tổn thương của mạng và cải thiện hiệu năng sử dụng các đường truyền. Sở dĩ kỹ thuật này có tên trên là vì dữ liệu, ví dụ như một bản tin, được phân chia thành từng gói thông tin nhỏ, có phần đầu chứa các thông tin giao thức điều khiển và được phân phối trong mạng như những đơn vị độc lập. Mạng chuyển mạch gói có nhiều thiết bị chuyển mạch, và tương ứng, nhiều kênh truyền dẫn thông tin, nên các gói trên có thể được phân phát đến đích nhanh chóng và có độ tin cậy cao hơn. Mặt khác, ngay trong một kênh dẫn thông tin nào đó, kỹ thuật này điều hòa được thời gian tự do của kênh và có thể gửi các gói từ nhiều thiết bị chuyển mạch qua kênh đó. Cơ chế *multiplexing* (đa lộ) phức hợp còn cho phép thiết lập nhiều phiên làm việc đồng thời trên một cổng.

Trong chuyển mạch gói, vấn đề chọn đường đi cho gói thông tin được chú ý đặc biệt, dựa trên các tiêu chuẩn về dung lượng của nối kết, số gói chờ được truyền đi trên mỗi nối kết, mức độ tải và bảo mật qua mạng, các loại nghẽn mạch có thể xảy ra, số các bước nối kết trung gian và khả năng truy cập chúng. Việc truyền các gói thông tin trước hết phải thỏa mãn yêu cầu giá thành thấp nhất.

- *Tiếp mạch phần tử* được coi như kỹ thuật thay thế cho các mạng chuyển mạch gói trong tương lai gần. Khác với chuyển mạch gói, tiếp mạch phần tử sử dụng các đơn vị thông tin được điều khiển bằng giao thức (Protocol Data Unit - PDU) cố định, được gọi là *phần tử*. Phần tử thường chứa 48 byte dữ liệu (nếu được truyền đi bởi thuê bao thì phải trả tiền) và 5 byte cho phần đầu, và được dùng trong chế độ

truyền dị bộ (*Asynchronous Transfer Mode* - ATM) và theo chuẩn IEEE 802.6, nhất là trong các mạng MAN đặc biệt. Tuy nhiên, Dịch vụ chuyển mạch dữ liệu nhiều megabit (*Switched Multimegabit Data Services* - SMDS) dựa trên cơ sở chuẩn IEEE 802.6 cung cấp dịch vụ chuyển mạch nhanh, thích hợp cho cả các mạng WAN và LAN.

#### ▪ *Tiên mạng*

Trong thực tế sử dụng, nhiều mạng LAN, và cả WAN nữa, với các chức năng logic đầy đủ, gọi là *mạng con*, được nối với nhau qua các *router* hay *gateway* (forwarder), tạo thành một kiến trúc được gọi là *liên mạng*. Liên mạng thường có cấu trúc phức tạp, đòi hỏi người quản trị phải có trình độ cao và nhiều kinh nghiệm. Các khó khăn hay nảy sinh trong thực tế quản trị liên mạng được phân thành các nhóm cơ bản sau:

- Mỗi một mạng con dùng một kích thước riêng cho PDU. Khía cạnh này có liên quan tới việc nhận biết kích thước và khả năng phân chia dữ liệu của mạng hay gateway.
- Các giá trị *đếm thời gian*, *thời gian trễ tạm ngừng* (timeout) và *thời gian lập lại* thường khác nhau đối với các mạng con khác nhau. Mạng phải có khả năng chấp nhận tất cả các sự khác biệt đó. Giao thức TCP/IP hỗ trợ khả năng đó, nhưng các chức năng chỉ được thực hiện trên máy chủ mà không phải trong các mạng con cụ thể.
- Các mạng con có thể sử dụng các *cách biến đổi địa chỉ* khác nhau.
- Các mạng con có các *mức hiệu năng* khác nhau.
- Các mạng con sử dụng các *thuật toán phân đường* khác nhau.
- Các mạng con đòi hỏi các loại giao diện với người dùng khác nhau.
- Các mạng con có các mức bảo mật khác nhau.
- Việc *tìm lỗi*, *chẩn đoán*, và *báo tri mạng* có thể khác nhau, được sử dụng hoặc không sử dụng khi đi từ mạng con này sang mạng con khác.

## XIV. 4. Phần cứng giao tiếp mạng

### XIV. 4. a. Các loại cáp mạng và đầu nối

#### ▪ Cáp Ethernet béo

Là loại cáp đồng trục có trở kháng 50 Ohm, có vỏ cách điện màu vàng tươi hoặc vàng đậm, đường kính cỡ 1 cm (0,4 inch). Cáp này được coi là cáp chuẩn của mạng Ethernet, có ký hiệu 10BASE5. Số 10 có nghĩa tốc độ truyền dữ liệu 10 Mbit/s, số 5 chỉ rằng độ dài cực đại của một phân đoạn mạng là 500 m. Cáp Ethernet béo được nối với nhau, cũng như bị chặn ở hai đầu bằng các dụng cụ đặc biệt, gọi là đầu nối loại N và terminator loại N có dây nối đất. Một đầu cáp phải được nối đất, còn đầu kia không nối đất để tránh dòng điện kín với đất. Các trạm làm việc được nối với cáp qua các chạc ba, được gọi là bộ chuyển đổi loại N. Độ dài cực đại của mạng Ethernet với cáp béo là 2800 m, gồm 2500 m chiều dài các phân đoạn và 300 m chiều dài các đầu cáp nối vào máy. Bảng sau cho ta thấy các thông số của cáp Ethernet béo và quy tắc đi dây mạng.

| Thông số của cáp Ethernet béo                                                                                           | Giá trị             |
|-------------------------------------------------------------------------------------------------------------------------|---------------------|
| Tốc độ truyền dữ liệu cực đại                                                                                           | 10 Mbit/s           |
| Số các bộ lặp cực đại khi không có các IRL (đoạn nối giữa các phân đoạn mạng không chứa trạm làm việc mà chỉ có bộ lặp) | 2                   |
| Số các bộ lặp cực đại khi có các IRL                                                                                    | 4                   |
| Độ dài cực đại của một phân đoạn mạng                                                                                   | 500 m               |
| Độ dài cực đại của cáp nối với bộ chuyển đổi                                                                            | 50 m                |
| Số cực đại các liên kết giữa các phân đoạn mạng                                                                         | 2                   |
| Độ dài cực đại của phân đoạn mạng đã liên kết                                                                           | 1000 m              |
| Số lượng cực đại các trạm làm việc trên một phân đoạn mạng                                                              | 100                 |
| Số lượng cực đại các trạm làm việc                                                                                      | 1024                |
| Khoảng cách giữa các trạm làm việc                                                                                      | Số nguyên lần 2,5 m |

Cáp Ethernet béo có độ bền cao, nhưng trong thực tế sử dụng của Việt Nam ta ít khi bắt gặp, vì cáp và các thiết bị đi kèm (đầu nối, bộ chuyển đổi, terminator) hiếm và có giá thành cao. Chỉ có các mạng đặc biệt mới sử dụng loại cáp này.

#### ▪ *Cáp Ethernet mảnh*

Thay thế cho cáp Ethernet béo là *cáp Ethernet gậy* hay còn gọi là *mảnh*, vỏ ngoài thường có màu đen. Số dĩ có tên gọi như vậy là do đường kính của cáp chỉ 0,5 cm (0,2 inch). Ký hiệu của cáp là 10BASE2, hay RG-58 A/U 50-Ohm. Terminator cho cáp Ethernet mảnh cũng có trở kháng 50 Ohm. Thiết bị nối vào trạm làm việc có tên là đầu nối T, hai đầu của nó nối với hai đoạn cáp, đầu còn lại nối với đầu cắm BNC của giao diện mạng (card mạng). Quy tắc đi dây tương tự như đối với cáp Ethernet béo, chỉ có các thông số có giá trị khác:

| Thông số của cáp Ethernet mảnh                                                                                          | Giá trị   |
|-------------------------------------------------------------------------------------------------------------------------|-----------|
| Tốc độ truyền dữ liệu cực đại                                                                                           | 10 Mbit/s |
| Số các bộ lặp cực đại khi không có các IRL (đoạn nối giữa các phân đoạn mạng không chứa trạm làm việc mà chỉ có bộ lặp) | 2         |
| Số các bộ lặp cực đại khi có các IRL                                                                                    | 4         |
| Độ dài cực đại của một phân đoạn mạng                                                                                   | 185 m     |
| Số cực đại các liên kết giữa các phân đoạn mạng                                                                         | 2         |
| Số lượng cực đại các trạm làm việc trên một phân đoạn mạng                                                              | 30        |
| Số lượng cực đại các trạm làm việc                                                                                      | 1024      |
| Khoảng cách cực tiểu giữa các trạm làm việc                                                                             | 0,5 m     |

#### ▪ *Cáp xoắn UTP*

Sự quan tâm tới các loại cáp xoắn bắt đầu từ những năm 1990 do có giá thành thấp, dễ cấu hình và lắp đặt dây mạng. Hiện nay, hầu hết các loại card mạng đều có đầu cắm loại RJ-45 dùng cho cáp xoắn, hay có tên gọi chính thức là MDI (xem lớp Vật lý trong mô hình mạng OSI/ISO). Cáp xoắn có tên gọi theo phân loại IEEE là 10BASE-T, thường gồm bốn đôi dây xoắn vào nhau, có đường kính lõi từ 0,4 đến 0,6 mm, trở kháng 8 Ohm. Nhược điểm của các loại cáp xoắn là do không có vỏ bọc

kim như các cáp Ethernet béo và mảnh nên chúng có khả năng chống nhiễu chéo kém hơn, và tốc độ truyền dữ liệu cũng thấp hơn. Tuy nhiên, với các thiết bị mạng hiện nay, mạng máy tính với tất cả các loại cáp trên có thể đảm bảo tốc độ truyền dữ liệu qua mạng đến 100 Mbit/s được gọi là mạng *Ethernet nhanh* (Fast Ethernet). Loại cáp xoắn dùng cho mạng Ethernet nhanh được gọi là cáp UTP loại 5 (tên viết tắt tiếng Anh là UTP Cat 5). Độ dài cực đại của một cáp (trong mạng hình sao) là 250 m. Số trạm làm việc cực đại trong một phân đoạn mạng (mạng hình sao) là 512. Số phân đoạn cực đại được liên kết là 2.

Ta cần chú ý đặc biệt tới các đầu cắm RJ-45 và cách đấu dây tới chúng. Nếu nhìn đầu cắm “dục” giống như đầu cắm dây điện thoại, sao cho lấy nhựa nằm ở phía dưới thì các đầu dây được đánh số từ 1 đến 8 theo chiều từ phải sang trái. Tương ứng, ổ cắm “cái” từ phía MDI (của card mạng) có các dây được đánh số từ trái sang phải, nếu chỗ khuyết dành cho lấy nhựa cũng ở phía dưới. Chức năng và mẫu của các dây như sau:

| Đầu số | Tín hiệu MDI               | Mã mẫu theo chuẩn AT&T T568B       | Mã mẫu theo chuẩn TIA 2840 cho T568B-UTP |
|--------|----------------------------|------------------------------------|------------------------------------------|
| 1      | TD+                        | Trắng gạch da cam                  | Trắng - Xanh lục                         |
| 2      | TD-                        | Da cam gạch trắng                  | Xanh lục                                 |
| 3      | RD+                        | Trắng gạch xanh lục                | Trắng - Da cam                           |
| 4      | Không sử dụng bởi 10BASE-T | Xanh lục gạch trắng                | Xanh lam                                 |
| 5      | Không sử dụng bởi 10BASE-T | Trắng gạch xanh lam                | Trắng - Xanh lam                         |
| 6      | RD-                        | Xanh lam gạch trắng                | Da cam                                   |
| 7      | Không sử dụng bởi 10BASE-T | Trắng gạch nâu (hoặc da cam tuyền) | Trắng - nâu                              |
| 8      | Không sử dụng bởi 10BASE-T | Nâu gạch trắng (hoặc xám tuyền)    | Nâu                                      |

Có hai cách đấu dây mạng UTP được áp dụng trong cấu hình mạng như sau. Cách thứ nhất thực hiện đấu chéo ngoài. Nếu xét hai đầu tương ứng của cáp mạng (trong mạng hình sao) thì quy tắc đấu dây là 1-2 -> 3-6, 3-6 -> 1-2, nghĩa là đầu dây phát tín hiệu (TD+, TD-) ở một đầu cáp nối vào đầu thu (RD+, RD-) ở đầu cáp kia, và ngược lại. Cách đấu này thường dùng để nối trạm làm việc với hub hoặc switch. Cách thứ hai thực hiện đấu chéo phía trong, nên quy tắc đấu dây không xáo trộn trật tự: 1-2 -> 1-2, 3-6 -> 3-6. Các dây cáp được đấu theo quy tắc này áp dụng cho các MDI có đánh dấu "X", thường dùng trong trường hợp đấu xếp chồng các hub hay switch.

#### ▪ *Cáp quang*

Trong các mạng đòi hỏi tốc độ truyền nhận dữ liệu nhanh và độ tin cậy cao, chống nhiễu tốt, người ta dùng cáp quang, có tên gọi theo IEEE là 10BASEF. Dữ liệu trong mạng từ dạng các tín hiệu điện qua các bộ biến đổi (điốt laser) được biến thành các tín hiệu quang được điều biến và được truyền đi trong môi trường thủy tinh của cáp quang tới đầu thu, lại được giải mã thành các tín hiệu điện. Cáp quang hiện nay có nhiều loại, tùy theo cách chế tạo, dải thông (giới hạn bước sóng ánh sáng được truyền đạt) và khả năng truyền đạt (đơn mode, đa mode). Dải thông của cáp quang lớn hơn nhiều so với các loại cáp đồng trục và cáp xoắn. Nhưng cáp quang có giá thành rất cao, kèm theo các thiết bị phụ trợ (đầu phát, đầu thu, đầu nối, bộ khuếch đại, v.v...) tương đối đắt. Thêm nữa, việc lắp đặt, sửa chữa cũng đòi hỏi kỹ năng cao của những cán bộ kỹ thuật. Sau đây là một số thông số của cáp quang.

| Các thông số của cáp quang                        | Giá trị     |
|---------------------------------------------------|-------------|
| Dải thông                                         | 10 Mbytes/s |
| Chiều dài cực đại của phân đoạn mạng              | 1000 m      |
| Số bộ chuyển tiếp cực đại trên một phân đoạn mạng | 2           |
| Số bộ chuyển tiếp cực đại trên toàn mạng          | 1024        |
| Khoảng cách tối thiểu giữa hai bộ chuyển tiếp     | 2000 m      |
| Terminator                                        | -           |

|                                                              |                              |
|--------------------------------------------------------------|------------------------------|
| Chiều dài mạng                                               | 4 bộ lặp xếp chồng           |
| Chiều dài cực đại của đoạn cáp từ cáp mạng tới trạm làm việc | 50 m                         |
| Số cực đại các phân đoạn mạng                                | -                            |
| Loại cáp                                                     | multimode,<br>50/125; 62,5/1 |
| Đầu nối cáp                                                  | AUI                          |

#### ***XIV. 4. b. Repeater***

Bộ lặp (repeater) được dùng ở lớp Vật lý. Repeater dùng để nối hai mạng hoặc mạng con với nhau bằng cách sao chép từng bit một của các gói thông tin. Chiều dài dòng tín hiệu (tức là số bit có thể xử lý đồng thời) và số lượng của chúng là các đặc trưng phản ánh chất lượng repeater.

Do bản chất của mình, hiện nay repeater được dùng ở phạm vi rất hạn chế trong các mạng có dây cáp mạng bình thường. Các đòi hỏi của mạng hiện đại, ví dụ như Ethernet, bắt buộc thông tin chỉ được truyền qua mạng trong tổng lượng thời gian nào đó, vượt quá khả năng cho phép của repeater. Do khả năng đáp ứng của cáp quang lớn hơn nhiều, trong thực tế ta hay bắt gặp repeater trong các mạng cáp quang nhiều hơn.

#### ***XIV. 4. c. Bridge***

Hoạt động ở lớp Liên kết dữ liệu. Cầu (bridge) được thiết kế để kết nối mạng một cách trong suốt. Cho phép chia nhỏ mạng thành từng phần để tăng tốc độ thông qua.

Bridge sao chép dữ liệu - khác với repeater - theo từng gói. Vì vậy, mọi giao thức phía trên lớp Liên kết dữ liệu đều được hỗ trợ và đảm bảo hoạt động. Về mặt lý thuyết, kết nối như vậy được coi là "vô hình" hay "trong suốt" - không làm ảnh hưởng gì - đối với phần mềm trên tất cả các máy trong mạng.

Trong từng gói, như đã thấy ở bảng minh họa về khung dữ liệu (phần *Khái niệm về mô hình mạng*), có chứa kèm dữ liệu với các thông tin khác như địa chỉ đích,



nguồn,... Khác với repeater, bridge có thể sao chép các gói một cách có chọn lọc, dựa trên các thông tin địa chỉ đó. Điều đó giúp làm giảm đáng kể lưu lượng thông tin và sự ách tắc trong mỗi phần của mạng.

#### ***XIV. 4. d. Router***

Router có vị trí nằm ở lớp Mạng và được dùng để: hợp nhất hai mạng con thành một mạng lớn hơn, gửi các gói thông tin (chứa dữ liệu) với các giao thức đã định từ một mạng con này tới mạng con khác, chuyển các thông báo giữa các giao thức khác nhau. Router dùng để kết nối giữa hai mạng logic, có nghĩa là có thể kết nối các mạng với bản chất vật lý khác nhau, ví dụ, từ mạng Ethernet tới mạng ring.

Trong router luôn có một bảng phân đường thường xuyên được cập nhật thông qua thủ tục phân đường, giúp router gửi các gói thông tin tới những địa chỉ cần thiết. Router có thể là đa ngôn ngữ, nghĩa là đồng thời hỗ trợ trên một thiết bị nhiều giao thức mạng khác nhau.

Một máy chủ Sun Solaris có thể đồng thời làm router nếu có nhiều hơn một giao diện mạng. Daemon **routed** chịu trách nhiệm làm máy chủ chạy như một router. Để cấu hình router ta cần thực hiện các bước:

- Kiểm tra địa chỉ mạng (IP) của các mạng sẽ tham gia vào các kết nối có router.
- Gán cho router tên host và địa chỉ IP với từng mạng mà nó tham gia.
- Đăng nhập bằng quyền root vào từng máy trong các mạng.
- Đối với từng máy bổ sung thêm dòng vào cho router ở file `/etc/networks`.
- Thêm dòng vào cho router (tên và địa chỉ IP) ở file `/etc/hosts` cho từng mạng trong từng máy. Ví dụ, router tham gia vào mạng Kythuat với tên `rou1` và địa chỉ `202.192.17.100` và vào mạng Ketoan với tên `rou1b`, địa chỉ `202.192.23.102`:

```
Local Net 202.192.17 -- 10Mb/s Ethernet --Kythuat
#
202.192.17.100 rou1
202.192.17.1 anh
```

```
202.192.17.2 binh
202.192.17.3 dung
Local Net 202.192.23 -- 10Mb/s Ethernet --Ketoan
202.192.23.102 rou1b
202.192.23.10 hoa
202.192.23.11 chi
```

- Đăng nhập vào router với quyền supervisor và thêm dòng vào cho router (tên và địa chỉ IP) ở file `/etc/hosts` cho từng mạng. Theo ví dụ trên:

```
202.192.17.100 rou1
202.192.23.102 rou1b
```

- Trong trường hợp dùng một máy Solaris làm router, sửa file `/etc/hosts` để bổ sung thêm dòng vào cho router (tên và địa chỉ IP) đối với từng mạng. Tạo các file `/etc/hostname.xx#` và ghi tên, địa chỉ IP của máy (host) đó vào file. Dấu xx chỉ viết tắt của file thiết bị card mạng, dấu # chỉ thứ tự. Ví dụ, cho hai card mạng 3Com 3C509 trong máy chủ Solaris, phần mở rộng của các file đó là `e1x0` và `e1x1`.

Cách sử dụng router trong xây dựng mạng sẽ được trình bày chi tiết hơn trong các phần sau liên quan tới mạng WAN và cụ thể với các loại router Cisco. Do khuôn khổ của cuốn sách nên trong lần xuất bản này, phần trình bày chi tiết đã nói ở trên, trong một chương riêng về cấu hình và quản trị router trong mạng WAN, sẽ được đưa vào trong lần tái bản tiếp theo.

#### ***XIV. 4. e. Gateway (Application server hay Relay hay Forwarder)***

Dùng để hợp nhất hai mạng khác loại thành một hay để chuyển đổi giao thức này thành giao thức kia. Một máy chủ UNIX thường đóng vai trò là gateway luôn với cùng địa chỉ IP, nếu không có một máy riêng làm nhiệm vụ đó.

#### ***XIV. 4. f. Terminal server***

Nối hệ thống trong mạng LAN với các thiết bị cổng nối tiếp (thiết bị đầu cuối, máy in, ...). Có thể hỗ trợ các giao thức Telnet.

## XV. Các giao thức và dịch vụ mạng

### XV.1. Giao thức TCP/IP

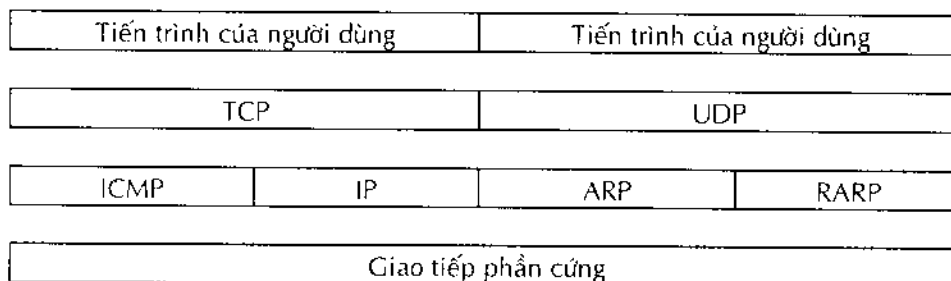
#### XV.1.a. Giới thiệu

Giao thức TCP/IP được thiết lập bởi tổ chức Internet. TCP/IP là một tập các giao thức (FTP, SMTP, Telnet, v.v...) hiện được coi là chuẩn cho các loại hình dịch vụ truyền thông khác nhau (truyền file, thư điện tử, mô phỏng terminal, v.v...). Hiện nay TCP/IP là giao thức được sử dụng phổ biến nhất trên tất cả các loại hệ điều hành và là cơ sở để xây dựng các họ giao thức mạng khác.

Các giao thức TCP/IP được cài đặt như các ứng dụng Client/Server. Một "client" được sử dụng để yêu cầu sự thực hiện một nhiệm vụ bởi máy "server" từ xa. Máy "server" có trách nhiệm xử lý yêu cầu của máy "client". Dữ liệu truyền qua LAN và WAN theo từng gói dữ liệu.

Một host mà có phần cứng cho cả LAN và WAN được gọi là "gateway", vì chúng có khả năng phân đường giữa LAN và WAN. Để cho các máy PC sử dụng các phần cứng LAN, WAN các bộ điều khiển thiết bị phải được cài đặt. Các bộ điều khiển cho LAN có thể là NDIS, ODI, ASI, PDS. Các bộ điều khiển cho WAN có thể là X.25, PPP và SLIP.

#### XV.1.b. Các thành phần của TCP/IP



**TCP** (*Transmission Control Protocol*) Giao thức hướng đến liên kết: tin cậy, hai chiều, kiểm soát sự truyền nhận các dòng dữ liệu. Đa số các ứng dụng sử dụng TCP.

**UDP** (*User Datagram Protocol*) Giao thức không liên kết: không tin cậy, dòng dữ liệu được truyền mà không cần kiểm tra kết quả của cuộc truyền nhận.

**ICMP** (*Internet Control Message Protocol*) Giao thức điều khiển sai và kiểm soát thông tin giữa “gateway” và các máy chủ.

**IP** (*Internet Protocol*) Giao thức cung cấp các dịch vụ chuyển gói cho TCP, UDP và ICMP.

**ARP** (*Address Resolution Protocol*) Giao thức phân giải địa chỉ, dùng để ánh xạ địa chỉ Internet (32 bit) vào địa chỉ phần cứng (địa chỉ MAC, 48 bit).

**RARP** (*Reverse Address Resolution Protocol*) Giao thức phân giải địa chỉ ngược, ánh xạ địa chỉ phần cứng vào địa chỉ Internet.

### **XV. 1. c. Các khái niệm cơ bản**

#### ▪ *Địa chỉ IP*

*Địa chỉ IP*, hay còn được gọi là địa chỉ Internet, là chỉ số duy nhất dùng để định danh cho mỗi máy trong mạng. Địa chỉ IP được biểu diễn bằng 4 byte, với 3 byte đầu tiên dành cho chỉ số mạng và mạng con, gọi là *địa chỉ mạng* và *địa chỉ mạng con*. Việc phân chia độ dài theo số byte của các địa chỉ mạng và mạng con phụ thuộc vào việc mạng được xét thuộc “lớp” nào. Byte cuối được dành cho địa chỉ máy. Mỗi byte cách nhau bởi dấu chấm “.”.

Địa chỉ IP có dạng sau cho mạng lớp C:

|        | Địa chỉ mạng | Địa chỉ mạng con | Địa chỉ máy |
|--------|--------------|------------------|-------------|
| Ví dụ: | 192.127.     | 50.              | 1           |

Địa chỉ IP không phải là được đặt tùy ý theo sở thích của từng người quản trị. Các lớp mạng được quy định như sau:

| Lớp | Khoảng    | Địa chỉ mạng | Địa chỉ máy |
|-----|-----------|--------------|-------------|
| A   | 0 - 127   | xxx          | xxx.xxx.xxx |
| B   | 128 - 191 | xxx.xxx      | xxx.xxx     |
| C   | 192 - 223 | xxx.xxx.xxx  | xxx         |
| D   | 224 - 239 |              |             |

Lớp mạng A có số lượng ít nhất (127), nhưng là mạng lớn nhất. Địa chỉ máy cuối cùng trong mạng lớp A là 255.255.254, nghĩa là mạng đó có đến 16 777 214 máy. Những mạng thuộc lớp B, số lượng 16 383, là mạng "tầm trung", nghĩa là có cỡ hơn 65 000 máy. Mạng loại C có số máy cực đại là 254. Số mạng loại C là ngoài 2 000 000. Ngoài ra còn có một số địa chỉ được gọi là mạng lớp D, dành cho mục đích thông báo. Trong các địa chỉ thuộc ba lớp A, B, C cũng có một số địa chỉ đặc biệt được dành riêng. Ở nước ta, chủ yếu mới có các mạng loại C hoặc riêng lẻ, hoặc được kết hợp lại, nhưng nhiều khi địa chỉ được đặt không thống nhất.

Các đặc điểm của địa chỉ IP:

- Địa chỉ mạng: Nhận từ Trung tâm thông tin mạng (Network Information Center - NIC), phải trả phí đăng ký và không thể thay đổi được.
- Địa chỉ mạng con: Tùy chọn, chỉ số mạng con mà máy nối vào.
- Địa chỉ máy: Xác định duy nhất cho mỗi máy trên mạng. Do người quản trị hệ thống đặt khi tạo mạng hay thêm máy, dựa trên địa chỉ mạng đã được phân phối.
- Mặt nạ mạng con (*subnet mask*): Số sử dụng bởi phần mềm để phân cách địa chỉ mạng con và phần còn lại của địa chỉ IP. Ví dụ: 255.255.255.0 ba byte đầu biểu diễn chỉ số mạng và mạng con.
- Địa chỉ chung (*Broadcast Address*): Được sử dụng để lan truyền thông báo đến tất cả các máy trong mạng. Ví dụ, địa chỉ 195.127.50.255 cho mạng lớp C 195.127.50. Khi sử dụng địa chỉ này để gửi thông báo thì tất cả các máy trong mạng 195.127.50 sẽ nhận được thông báo đó.
- Tên vùng (*domain name*): Tên dùng cho một nhóm máy được quản trị đồng thời. Tên vùng có tính phân cấp, các thành phần được phân cách bởi dấu chấm ". ". Mức vùng được tính từ phải sang trái. Mức vùng cao nhất có thể là: .com : các tổ chức thương mại; .edu : các tổ chức giáo dục; .org : các

cơ quan, tổ chức; .mil : các tổ chức quân sự, hay theo tên nước : .vn Việt Nam, .us – Hoa Kỳ, .uk – Anh, .ca – Ca na đa, .cn – Trung quốc, .ru – Nga, v.v..

Tên vùng đầy đủ phải duy nhất trên toàn thế giới. Ví dụ: hcm.com.vn.

- Dịch vụ tên vùng (*Domain Name Service*): dịch vụ tên của họ TCP/IP, cung cấp các thông tin về các máy cục bộ cũng như từ xa.
- Tên máy (*host name*): Có thể là địa chỉ IP hay một chuỗi ký tự như là tên hình thức của máy. Được định nghĩa trong file `/etc/hosts`. Ví dụ: trong file `/etc/hosts` có dòng "195.127.50.1 sunhost" thì sunhost là tên máy.

Tất cả nội dung được trình bày trên đây nằm trong tiêu chuẩn địa chỉ IP gồm bốn byte, hay còn gọi là **IPv4**. Hiện nay, do sự phát triển rất mạnh của công nghệ thông tin, số máy tính nối mạng đã tăng vọt. Để khắc phục sự thiếu hụt các địa chỉ IP cần cấp phát, người ta đã đưa ra một chuẩn địa chỉ mới, **IPv6**, gồm sáu byte địa chỉ. **IPv6** hiện nay bắt đầu được dùng rộng rãi và đã được nhiều phiên bản mới của các hệ điều hành (ví dụ, Solaris 8, các bản Linux từ v.6.4 của SuSE, RedHat 6.2, v.v..) hỗ trợ. Mô tả chi tiết về **IPv6** sẽ được trình bày với bạn đọc ở một mục riêng trong chương này.

#### ***XV. 1. d. Các ứng dụng phổ biến của TCP/IP***

TCP/IP hiện nay là giao thức mạng phổ biến rộng rãi nhất và được hỗ trợ bởi tất cả các hệ điều hành hiện có. Do đó, có rất nhiều ứng dụng khác nhau được xây dựng dựa trên TCP/IP và đã được trình bày chi tiết trong các phần trước, ví dụ như trong chương V "*Sử dụng các dịch vụ Internet*". Trong mục này ta chỉ ôn lại cách dùng cơ bản của các lệnh đã nêu:

- Telnet:  
Cho phép các terminal liên lạc với nhau theo từng ký tự. Nó cũng cho phép mô phỏng PC như một terminal của máy chủ.
- Ftp (File Transfer Protocol):  
Cho phép truyền file giữa các máy (mạng) cách xa nhau.
- Dịch vụ tên vùng (DNS):

Cung cấp các ánh xạ Tên -Địa chỉ.

- Ngoài ra còn có các lệnh khác như **rcp**, **rsh**, **ssh**, v.v..

Dùng để thực hiện từ xa hay sao chép các file ở các máy từ xa.

#### ***XV. 1. e. Các lệnh cơ bản trên DOS, UNIX***

Các lệnh này đã được trình bày dưới góc độ người dùng trong chương *Sử dụng các dịch vụ Internet*. Trong phần này, ta quan tâm trước hết tới phương diện tổ chức cấu hình và quản lý đối với chúng.

##### ▪ *Kiểm soát đường truyền*

```
ping <host name> [time out]
```

với timeout là thời gian cố gắng kết nối. Ví dụ:

```
$ ping sunhost
```

Lệnh **ping** gửi một ICMP tới hostname. Nếu máy đích sunhost được nêu trên có đáp ứng thì sẽ có thông báo trả về: sunhost is alive. Nếu không, ta sẽ nhận được trả lời: No answer from sunhost.

##### ▪ *Mô phỏng terminal*

Cho phép terminal liên kết với máy chủ từ xa. Có hai loại terminal thường xuyên có thể được mô phỏng là VT100 hay IBM3270.

Ví dụ:

```
$ rlogin sunhost
login:
password:
```

##### ▪ *Truyền file*

Cho phép truyền và nhận file với máy từ xa. Cú pháp:

```
ftp [-b] [-d] [-g] [-i] [hostname [port]]
```

Ví dụ:

```
$ ftp sunhost
login:
password:
>
```

Tại dấu nhắc này có thể đưa vào các lệnh hay sử dụng lệnh **help** để liệt kê các lệnh có thể sử dụng. Danh sách các lệnh con của **ftp** đã được liệt kê chi tiết trong chương)V “Sử dụng các dịch vụ Internet”.

- Sao chép file từ xa

Sao chép file giữa các máy. Cú pháp được quy định cho lệnh:

```
rcp [options] source destination
```

Các tùy chọn hay dùng:

**p**

Sao chép cả thời gian và chủ quyền của file nguồn

**r**

Sao chép cả các thư mục con.

nguồn và đích có dạng hostname:pathname, hostname có thể được bỏ qua ở máy cục bộ.

Ví dụ:

```
$ rcp test sunhost:/usr/doson
```

hoặc:

```
$ rcp sunhost:/usr/mai/test /usr/binh/quang
```



- *Thực hiện lệnh từ xa*

Liên kết với host và thực hiện lệnh. Cú pháp:

```
rsh host [options] [command]
```

Các tùy chọn:

**-l** *username*

Liên kết host với *username*.

**-N**

Hướng vào đến `/dev/null`.

Ví dụ:

```
$ rsh sunhost -l root cat /test > /thu
```

chép file `/test` ở máy `sunhost` sang máy cục bộ.

- *Chuyển đổi dạng văn bản*

Cú pháp:

```
dos2UNIX [-b|u|-l] [-f] [input-file [output-file]]
```

```
UNIX2dos [-b|u|-l] [-f] [-z] [input-file [output-file]]
```

Lệnh đầu chuyển văn bản dạng DOS sang dạng UNIX, lệnh sau - ngược lại. Nếu không dùng các lệnh chuyển đổi này, chúng ta có thể gặp các điều bất tiện như các ký hiệu kết thúc câu “^M” luôn luôn hiện ra mỗi khi xem nội dung file văn bản của UNIX bằng các chương trình chạy trên MS-DOS hay Windows (ví dụ như Notepad chẳng hạn).

- *In file*

In nội dung file ra máy in từ xa. Cú pháp:

```
lpr hostname filename
```

hostname là tên máy có gắn máy in.

Ví dụ:

```
$ lpr sunhost test
```

Ngoài ra trong PC/TCP còn có các chương trình chạy trong Windows mà cho phép sử dụng các giao thức TCP/IP ở chế độ đồ họa.

### ***XV. 1. f. Mặt nạ mạng con và mặt nạ liên mạng***

#### ***▪ Cấu hình mặt nạ mạng con***

Mạng con và mặt nạ mạng con được dùng nhằm giúp cho người quản trị mạng có thể phân chia một mạng thành các phần khác nhau, ví dụ như để nối với các cổng khác nhau của router, với mục đích sử dụng tách bạch và có hiệu quả hơn vùng địa chỉ IP đã đăng ký. Mặt nạ mạng con cho biết có bao nhiêu mạng con trong một mạng đã cho và phải là duy nhất cho tất cả các máy trong cùng một mạng con.

Nguyên tắc che mặt nạ mạng con là tìm cách viết bù thêm vào trường 32 bit của địa chỉ IP theo quy tắc: số 1 nhị phân xác định phần địa chỉ mạng và số 0 nhị phân xác định phần địa chỉ máy của địa chỉ IP. Các giá trị mặc định của mặt nạ mạng con được tự động xác định dựa vào giá trị *class bit* của mạng, nếu ta không đặt giá trị mặt nạ mạng con:

- Lớp A (class bit 0): 11111111 00000000 00000000 00000000 theo mã nhị phân hay 255.0.0.0 theo mã thập phân.
- Lớp B (class bit 10): 11111111 11111111 00000000 00000000 theo mã nhị phân hay 255.255.0.0 theo mã thập phân.
- Lớp C (class bit 110): 11111111 11111111 11111111 00000000 theo mã nhị phân hay 255.255.255.0 theo mã thập phân.

Ngoài cách trên ta có thể dùng các cách khác nữa để ký hiệu giá trị mặt nạ mạng con. Ví dụ, đối với địa chỉ IP 134.162.3.5 thuộc mạng lớp B, nếu ta đặt cho mặt nạ mạng giá trị 255.255.252.0, viết dưới dạng mã nhị phân là 11111111 11111111 11111100 00000000, ta thấy về phía bên trái sẽ có 22 chữ số 1. Vậy, một cách viết tương đương là:

---



---

subnetbits=22

hay viết đầy đủ cả địa chỉ IP và mặt nạ mạng con:

134.162.3.5/22

Ngoài ra, ta còn có thể viết dưới dạng *extension bit* (bit mở rộng) như sau. Để ý rằng giá trị mặc định của mặt nạ mạng lớp B là 255.255.0.0 hay theo mã nhị phân 11111111 11111111 00000000 00000000 gồm 16 chữ số 1 về phía bên trái. Hiệu số giữa giá trị mặt nạ mạng ta chọn và giá trị mặc định là  $22 - 16 = 6$  chữ số 1, nên ta có thể viết:

bits = 6

#### ▪ *Lựa chọn giá trị mặt nạ mạng con*

Dùng các giá trị mặt nạ mạng khác nhau, ta có thể điều khiển cho host thuộc về mạng con nào và mạng con đó gồm bao nhiêu máy. Ví dụ, ta có host với địa chỉ 131.130.45.17 hay 10000011 10000001 00101101 00010001 và ta chọn giá trị mặt nạ mạng con là 255.255.252.0 hay các bit cuối là 11111100 00000000. Thực chất, địa chỉ vật lý của host đó với lựa chọn của ta sẽ là: chỉ số mạng 131.130.44.0 hay 10000011 10000001 00101100 00010001 và host ID là 0.0.1.17 hay 00000000 00000000 000000 01 00010001. Như vậy, khi ta chọn giá trị mặt nạ mạng con khác tức là ta đã thay đổi cấu hình của cả hệ thống mạng. Do các mạng con cần phải nối với nhau qua router, nên số router cũng thay đổi theo tương ứng. Nếu ta chia thành nhiều mạng con sẽ thuận lợi hơn cho việc quản lý mạng, nhưng giá thành sẽ tăng lên rất đáng kể do số router tăng thêm. Ngược lại, nếu số mạng con ít thì số máy (host) lớn gây phức tạp nhiều hơn. Có hai điểm nên chú ý khi lựa chọn giá trị mặt nạ mạng con trong thực tế vận hành hệ thống máy tính. Thứ nhất, thông thường người ta chọn giá trị mặc định, trong ví dụ mạng lớp B của ta là 255.255.255.0, tức là có 254 mạng con (lớp C) với 254 máy trong mỗi mạng con, để dễ điều hành mạng. Tiếp theo là trong một mạng máy tính lớn nên tránh thay đổi các giá trị mặt nạ mạng.

Đối với mạng lớp B hay được thấy trong các tổ chức có quy mô cỡ trung bình, chúng ta có thể hệ thống lại các giá trị mặt nạ mạng con được dùng ở dạng số thập phân, các bit đáng quan tâm trong dạng nhị phân (chu kỳ thứ ba của địa chỉ IP), số mạng con và số host cực đại của mỗi mạng con trong bảng sau:

| Dạng thập phân | Dạng nhị phân    | Số mạng con | Số host |
|----------------|------------------|-------------|---------|
| 255.255.0.0    | ... 00000000 ... | 1           | 65534   |
| 255.255.128.0  | ... 10000000 ... | 2           | 32766   |
| 255.255.192.0  | ... 11000000 ... | 4           | 16382   |
| 255.255.224.0  | ... 11100000 ... | 6           | 8190    |
| 255.255.240.0  | ... 11110000 ... | 14          | 4090    |
| 255.255.248.0  | ... 11111000 ... | 30          | 2046    |
| 255.255.252.0  | ... 11111100 ... | 62          | 1022    |
| 255.255.254.0  | ... 11111110 ... | 126         | 510     |
| 255.255.255.0  | ... 11111111 ... | 254         | 254     |

Đối với các mạng lớp C, người ta cũng hay dùng hai giá trị mặt nạ mạng con sau để các mạng con, cũng như các máy trong mỗi mạng có số lượng vừa phải:

| Dạng thập phân  | Dạng nhị phân | Số mạng con | Số host |
|-----------------|---------------|-------------|---------|
| 255.255.255.224 | ... 11100000  | 6           | 30      |
| 255.255.255.240 | ... 11110000  | 14          | 14      |

#### ▪ *Tiền mạng*

Trong thực tế vận hành các mạng máy tính có thể một cơ quan nào đó được phân nhiều mạng lớp C, nhưng lại có nhu cầu liên hệ các máy tính với nhau. Thông thường, có bao nhiêu mạng lớp C với các giá trị mặt nạ mạng mặc định 255.255.255.0 cần liên hệ với nhau, ta phải cần có bấy nhiêu router. Ví dụ, nếu có ba mạng 202.192.37.0, 202.192.33.0, 202.192.35.0, ta cần dùng ba router. Người ta áp dụng một kỹ thuật có tên là *liên mạng* – đặt giá trị thích

hợp cho mặt nạ mạng của cả ba mạng trên, ví dụ 255.255.252.0 – biến ba mạng đó trở thành các mạng con của một liên mạng lớn hơn và có thể nối với nhau qua bridge. Kết quả là số router được giảm thiểu; trong trường hợp ví dụ này ta không cần dùng tới router.

Kỹ thuật liên mạng, về lý thuyết, tất nhiên có thể áp dụng cho các mạng lớp cao hơn. Tuy nhiên, xác suất vận dụng liên mạng đối với các mạng lớp C, đặc biệt là ở nước ta, là lớn nhất.

### ***XV. 1. g. Anonymous ftp và Mirroring***

Thông thường, khi thiết lập kết nối **ftp**, tên tài khoản người dùng và mật khẩu được kiểm định. Trong trường hợp dùng tài khoản người dùng vô danh *anonymous*, thủ tục kiểm định trên bị bỏ qua và khi nối kết được thiết lập, lệnh **ftp** thay đổi thư mục hiện thời đến thư mục mặc định, thường là thư mục ``pub''. Vì lý do bảo mật, việc truy nhập bằng *anonymous ftp* bao giờ cũng bị hạn chế; với tài khoản này ta chỉ có thể sử dụng được các tài nguyên công cộng trên máy chủ của site ftp.

Mirror (gương) là script được viết bằng ngôn ngữ Perl và sử dụng giao thức **ftp** để sao chép y nguyên cấu trúc các thư mục qua Internet. Script này so sánh kích thước và thời gian của file giữa các máy tại chỗ và từ xa, rồi thực hiện việc truyền file chỉ khi nào phát hiện ra có sự khác biệt các thông số của file. Thoạt đầu, gương được viết cho các nhân viên làm công tác bảo trì dữ liệu, nhưng rồi có thể được dùng cho bất cứ ai muốn chuyển một số lượng lớn các file qua **ftp**. Hiện nay, hầu hết các trung tâm **ftp** phục vụ cho rất nhiều các yêu cầu của người dùng đều có các site gương rải khắp các châu lục để giảm tải. Các site gương đó có thể cập nhật thông tin bằng cách chạy định kỳ script gương, thường là qua daemon định thời biểu **cron**, để đảm bảo có dữ liệu tương ứng với máy nguồn.

#### ▪ *Thiết lập anonymous ftp site*

Trong mục nhỏ thứ nhất của phần này chúng ta hãy từng bước tạo ra một ftp site để người dùng *anonymous* có thể truy cập được các thông tin mà ta muốn chia sẻ cho tất cả mọi người. Cùng lúc đó, ta cũng tạo ra một thư mục để cho phép người dùng lưu lại các file sao cho ta có thể sắp xếp theo các tiêu chí được xác định (loại ứng dụng, tên tác giả, tên công ty, v.v..) rồi đưa ra phục vụ công cộng. Đặc biệt, ta

cũng phải chú ý thao tác sao cho không để sót một lỗ hổng bảo mật nào khiến cho những kẻ xâm nhập có thể lợi dụng can thiệp vào hoạt động của ftp site. Trình tự thao tác cụ thể phải thực hiện như sau:

### Tạo tài khoản người dùng ftp

Ta có thể sử dụng shell script "add\_user" để thêm tài khoản người dùng vào trong máy theo kiểu cú pháp:

```
useradd -m -u uid -g gid -c fullname -d homedir
-s shell uname
```

và gõ (giả sử các thao tác được thực hiện trên host có tên letuan):

```
[letuan #] cd /usr/etc/install
[letuan #] add_user 100 20 "anonymous ftp"
/export/data/letuan/ftp /bin/false ftp
```

Câu lệnh cuối sẽ tạo ra, đồng thời với tài khoản người dùng ftp, thư mục /export/data/letuan/ftp được coi như thư mục gốc đối với các người dùng anonymous khi họ truy nhập vào hệ thống của ftp site. Nên chú ý rằng thư mục này được sở hữu bởi root, chứ không phải bởi người dùng ftp. Nếu bỏ qua chi tiết này, ta có thể gặp rắc rối sau này với các vấn đề bảo mật.

Tất nhiên, ta cũng có thể thực hiện bước này bằng tay, thông qua các lệnh hệ thống UNIX như **useradd**, **usermod**, v.v.. Bất luận bằng cách nào, ta cũng phải thực hiện các thao tác với quyền root.

### Tạo các thư mục cần thiết dưới thư mục ftp

Thao tác dưới đây làm cho tất cả các file dưới thư mục ftp có thể đọc và thực thi được bởi mọi người dùng, nghĩa là tất cả người sở hữu, nhóm người dùng cùng với người sở hữu và các người dùng khác.

```
[letuan #] chmod 555 /export/data/letuan/ftp
```

• Ta còn phải tạo các thư mục bin, dev, etc, pub, usr, và usr/lib dưới thư mục ftp:

```
[letuan #] cd /export/data/letuan/ftp
[letuan #] mkdir bin etc dev pub lib usr
```

• Ta phải kiểm tra bằng lệnh **ls** xem có đúng root là người sở hữu các thư mục đó hay không:

```
[letuan #] ls -lg
total 16
-rw----- 1 ftp other 2897 Jan 21 00:32 .cshrc
-rw----- 1 ftp other 2496 Jan 21 00:32 .login
-rw----- 1 ftp other 2750 Jan 21 00:32 .rootmenu
-rw----- 1 ftp other 478 Jan 21 00:32 .sunview
dr-xr-xr-x 2 root other 512 Jan 21 00:46 bin/
dr-xr-xr-x 2 root other 512 Jan 21 00:54 dev/
dr-xr-xr-x 2 root other 512 Jan 21 00:58 etc/
dr-xr-xr-x 2 root other 512 Jan 21 00:52 lib/
dr-xr-xr-x 4 root other 512 Jan 21 01:44 pub/
dr-xr-xr-x 3 root other 512 Jan 21 00:34 usr/
```

• Tạo thư mục "incoming" để cho phép mọi người nạp file lên ftp site.

Anonymous ftp site thường cho phép người dùng nạp lên các thông tin mà họ muốn chia sẻ với người khác vào các thư mục "incoming". Nhưng người quản trị sẽ sàng lọc các file đó một cách định kỳ để chọn ra các thông tin có giá trị thích hợp và phân loại sắp xếp chúng theo trật tự nhất định trong cấu trúc cây thư mục (trong tên thư mục ta bỏ bớt một chữ cái "m" để còn không quá 8 ký tự, nhằm thích hợp với cả chuẩn MS-DOS truyền thống 8.3).

Vì người dùng cần lưu các file của họ vào thư mục "incoming", ta cần mở cho họ quyền ghi vào thư mục đó. Tuy nhiên, để tránh không cho một số người dùng không đúng đắn gây lộn xộn (ví dụ như xóa hoặc thay đổi các file), ta lại phải bỏ đi

quyền đọc. Nên dùng các phần đĩa khác nhau cho các file được nạp lên, cách ly chúng, để tránh bị kẻ xấu tung dây rác rưởi và làm tê liệt hệ thống.

```
[letuan #159] cd /export/data/letuan/ftp
[letuan #161] mkdir incoming
[letuan #163] chmod 722 incoming
[letuan #164] ls -ld incoming
total 1
drwx-w--w- 2 root 512 Jan 21 01:52 incoming/
```

### Sao chép các lệnh và thư viện cần thiết

☞ Sao chép lệnh **ls** vào thư mục ftp/bin, và đổi chủ quyền của lệnh đó thành 555.

```
[letuan #] cd /export/data/letuan/ftp/bin
[letuan #] cp /bin/ls .
[letuan #] chmod 555 /export/data/letuan/ftp/bin/ls
[letuan #] ls -l
total 14
-r-xr-xr-x 1 root 13352 Jan 21 00:46 ls*
```

☞ Sao chép các thư viện vào thư mục ftp/lib, và đổi chủ quyền của chúng thành 555.

```
[letuan #] cd /export/data/letuan/ftp/usr/lib
[letuan #] cp /lib/ld.so .
[letuan #] cp /lib/libdl.so.* .
[letuan #] cp /lib/libc.so.* .
[letuan #] chmod 555 *
[letuan #] ls -l
total 1608
-r-xr-xr-x 1 root 40960 Jan 21 00:51 ld.so*
-r-xr-xr-x 1 root 516096 Jan 21 00:52 libc.so.1.9*
```



```
-r-xr-xr-x 1 root 524288 Jan 21 00:52 libc.so.1.9.1*
-r-xr-xr-x 1 root 24576 Jan 21 00:51 libdl.so.1.0*
```

☛ Chép các file /etc/passwd và file /etc/group vào thư mục ftp/etc.

```
[letuan #] cd /export/data/letuan/ftp/etc
[letuan #] cp /etc/passwd .
[letuan #] cp /etc/group .
```

### Thay đổi các quyền truy cập file

☛ Tạo file đặc biệt zero dưới thư mục ftp/dev. File đặc biệt zero là mã nguồn của bộ nhớ không có tên được diễn bởi các số 0. Thao tác đọc các file đặc biệt zero bao giờ cũng trả về bộ đệm chứa toàn giá trị 0. File này có chiều dài không xác định. Thao tác ghi vào file đặc biệt zero bao giờ cũng thực hiện được, nhưng dữ liệu ghi vào bị bỏ qua. Do đó, **không nên cố gắng** chép file zero mà phải tạo ra nó bằng lệnh **mknod**.

```
[letuan #] cd /export/data/letuan/ftp/dev
[letuan #] mknod zero c 3 12
[letuan #] chmod 555 zero
[letuan #] ls -l
total 0
crw-rw-r-- 1 root 3, 12 Jan 21 00:54 zero
```

☛ Thay đổi các file /etc/passwd và /etc/group.

Trong file passwd chỉ giữ lại tài khoản người dùng ftp, trong trường mật khẩu ta để dấu "\*". Trong file group chỉ giữ lại nhóm mà người dùng ftp tham gia, trong trường mật khẩu ta để dấu "\*".

Sau khi sửa xong, các file đó có nội dung như sau (chú ý rằng trong ví dụ ta lấy UID là 100, GID là 20 đối với người dùng ftp):

```
[letuan #] cat passwd
ftp:*:100:20:anonymous ftp:/export/data/letuan/ftp:
 /bin/false

[letuan #] cat group
other:*:20:
```

☞ Đổi quyền truy cập của các file /etc/passwd và /etc/group thành 444 để ai cũng có thể đọc được chúng.

```
[letuan #] chmod 444 *
[letuan #] ls -l
total 2
-r--r--r-- 1 root 196 Jan 21 00:58 group
-r--r--r-- 1 root 644 Jan 21 00:57 passwd
```

### Kiểm tra

Coi như bây giờ ta đã thiết lập xong một site anonymous ftp. Bước tiếp theo là phải kiểm tra xem hoạt động của nó có đúng như ta mong đợi không. Bạn có thể **ftp** vào máy letuan một cách vô danh từ một máy may-tu-xa nào đó. Khi được nhắc, ta gõ “ftp” hay “guest” rồi nhập địa chỉ E-mail của mình vào vị trí của mật khẩu để bắt đầu một phiên làm việc anonymous ftp thông thường.

```
[may-tu-xa:~:] ftp letuan
Connected to letuan.hut.edu.vn.
220 letuan FTP server (Version 4.98 Fri Feb 3
 10:44:29 PST 1995) ready.
Name (letuan:sunsol8-25): ftp
331 Guest login ok, send your E-mail address as
 password.

Password:
230- Guest login ok, access restrictions apply.
230- Local time is: Sun Jan 27 01:24:05 2001
230
```

```
ftp> ls
200 PORT command successful.
150 Opening ASCII mode data connection for ..
.
..
.cshrc
.login
.sunview
.rootmenu
bin
etc
usr
lib
dev
pub
incoming
226 Transfer complete.
71 bytes received in 0.022 seconds (3.2 Kbytes/s)
ftp> bye
221 Goodbye.
```

- *Tạo một mirror ftp site (ftp site gương)*

Trong mục nhỏ này chúng ta sẽ theo dõi các bước thiết lập một ftp site gương và luyện tập cách thực hiện các thao tác gương để sao chép cấu trúc cây thư mục. Trước hết, ngoài anonymous ftp site có tên letuan ta vừa thiết lập xong ở phần trước, ta còn phải thiết lập một anonymous ftp site khác, giả sử được lấy tên là may-tu-xa, theo cách thức tương tự. Sau đây là các thao tác cụ thể nhằm biến máy letuan thành ftp site gương của máy nguồn may-tu-xa:

**Thiết lập máy ftp nguồn. Tạo một site anonymous ftp khác**

- ☛ Thiết lập một site anonymous ftp khác trên may-tu-xa.

Ta lặp lại các thao tác trong mục trước với giả thiết rằng thư mục gốc mặc định để truy nhập anonymous ftp là /export/data/may-tu-xa/ftp. Ftp site trên may-tu-xa được chọn là nguồn, hay còn gọi là nguyên bản, còn ftp site gương (hay bản sao) được thực hiện trên máy letuan.

# Chuẩn bị kiểm tra cấu trúc cây thư mục trên may-tu-xa.

Công việc cần phải hoàn thành là sao chép cấu trúc cây thư mục vào thư mục /export/data/may-tu-xa/ftp/pub để cho phép các máy khác có thể thực hiện sao chép gương từ đó. Tiếp theo, phải thực hiện lệnh **du** để ghi chép lại hạn ngạch sử dụng đĩa dưới thư mục ftp/pub sao cho về sau ta có thể so sánh kết quả khi thực hiện xong sao chép gương.

```
[may-tu-xa #] cd /export/data/may-tu-xa/ftp/pub
[may-tu-xa #] cp /tmp/mirror-2.3.tar.gz .
[may-tu-xa #] cp -r /var/* var
[may-tu-xa #] chmod -R 555 *
[may-tu-xa #] du .
1 ./var/adm/acct/nite
1 ./var/adm/acct/fiscal
1 ./var/adm/acct/sum
4 ./var/adm/acct
902 ./var/adm
1 ./var/crash
2 ./var/log/httpd
11 ./var/log
1 ./var/preserve
1 ./var/spool/mqueue
1 ./var/spool/mail
2 ./var/spool/cron/crontabs
1 ./var/spool/cron/atjobs
6 ./var/spool/cron
9 ./var/spool
925 ./var
```

998

### Tải về mã nguồn của script gương

☞ Tạo thư mục ftp/pub/mirror

Thông thường, ta sẽ tạo thư mục gương ở dưới thư mục ftp/pub, sau đó thay đổi quyền truy cập file thành 555. Thư mục gương phải được đặt dưới quyền sở hữu của root, nếu không, ta phải thực hiện dòng lệnh sau để làm việc đó: **chown** root /export/data/letuan/ftp/pub/mirror.

```
[letuan #] cd /export/data/letuan/ftp/pub
[letuan #] mkdir mirror
[letuan #] chmod 555 mirror
[letuan #] ls -l
total 1
dr-xr-xr-x 2 root 512 Apr 21 02:00 mirror/
```

☞ tải mã nguồn về máy.

File nén mirror-2.3.tar.gz có thể được tải về bằng lệnh **ftp** hay các tiện ích sẵn có khác từ site <ftp://sunsite.unc.edu/pub/packages/ftp-archive/mirror/mirror-2.3.tar.gz>.

☞ Bung file nén mã nguồn ra thư mục /tmp/mirror

```
[letuan #] cd /tmp
[letuan #] mkdir mirror
[letuan #] gzcatt mirror-2.3.tar.gz | tar xvf -
[letuan #] cd mirror
```

### Thay đổi file cấu hình của gương

☞ Sửa đổi file mirror.pl

Ở dưới thư mục `/tmp/mirror`, ta thay đổi giá trị các biến `$extra_path` và `$big_temp` cho phù hợp. Chú ý rằng đường dẫn gọi chương trình PERL phải được xác định cho chính xác.

Thông thường, sau khi cài đặt, chương trình PERL được định vị tại thư mục `/usr/local/bin`, ta cần thực hiện các động tác cụ thể như sau:

thay:

```
#!/usr/bin/perl
```

bằng:

```
#!/usr/local/bin/perl
```

Chú ý: Các phiên bản trước của PERL thường gây ra lỗi cho script gương với thông báo không xác định được loại hệ thống file UNIX “fstype UNIX unknown”. Nếu gặp phải rắc rối tương tự, giải pháp tốt nhất là cập nhật phiên bản mới hơn của PERL.

✎ Sửa file `mirror.defaults`.

Ta thay:

```
package=defaults
hostname=src.doc.ic.ac.uk
local_dir=/public/
remote_password=ukuug-soft@doc.ic.ac.uk
```

bằng:

```
package=letuan_mirror
hostname=letuan.hut.edu.vn
local_dir=/export/data/letuan/ftp/pub/mirror
remote_password=baove@hut.edu.vn
```

và thêm:

```
site=may-tu-xa.hut.edu.vn
remote_dir=/pub/
```

và biến thành chú thích dòng:

```
compress_patt=.
```

Động tác sau cùng sẽ ngăn chặn việc nén file – file không nén sẽ làm ta dễ so sánh với kết quả hơn.

☛ Chép file socket.ph vào thư mục /tmp/mirror/sys

Trong trường hợp máy của bạn chạy trên nền Solaris 2.x (nghĩa là từ phiên bản 2.6 trở về trước cho x86), thay vì file socket.ph, ta chép file socket.ph-solaris.

```
[letuan #] mkdir sys
[letuan #] cp socket.ph ./sys
```

### Thử và kiểm tra

Ta sẽ thực hiện sao chép gương từ:

```
may-tu-xa(/export/data/may-tu-xa/ftp/pub/*a)
```

tới:

```
letuan(/export/data/letuan/ftp/pub/mirror).
```

☛ Thực hiện lệnh **mirror**

```
[letuan #] cd /tmp/mirror/
```

```
[letuan #] mirror
```

```
package=letuan_mirror may-tu-xa.hut.edu.vn:/pub/ ->
/export/data/letuan/ftp/pub/mirror/
```

```
Got mirror-2.3.tar.gz 73583
Got var/spool/cron/.proto 67
Got var/spool/cron/queuedefs 65
Got var/spool/cron/at.deny 0
```

---

|                                  |        |
|----------------------------------|--------|
| Got var/spool/cron/cron.deny     | 0      |
| Got var/spool/cron/FIFO          | 0      |
| Got var/spool/cron/crontabs/root | 188    |
| Got var/log/amdlog               | 1441   |
| Got var/log/syslog.0             | 458    |
| Got var/log/authlog              | 3943   |
| Got var/log/syslog               | 0      |
| Got var/log/syslog.1             | 950    |
| Got var/log/httpd/error_log      | 0      |
| Got var/log/httpd/access_log     | 0      |
| Got var/log/httpd/httpd.pid      | 4      |
| Got var/adm/messages.1           | 18148  |
| Got var/adm/messages             | 819    |
| Got var/adm/lpd-errs             | 0      |
| Got var/adm/usracct              | 0      |
| Got var/adm/lastlog              | 872088 |
| Got var/adm/msgbuf               | 0      |
| Got var/adm/messages.0           | 7961   |
| Got var/adm/wtmp                 | 5616   |
| Got var/adm/aculog               | 0      |

# So sánh kết quả

```
[letuan #] cd /export/data/letuan/ftp/pub/mirror
[letuan #] du .
```

```
1 ./var/spool/cron/atjobs
2 ./var/spool/cron/crontabs
6 ./var/spool/cron
1 ./var/spool/mail
1 ./var/spool/mqueue
9 ./var/spool
1 ./var/preserve
2 ./var/log/httpd
```



---

```

11 ./var/log
1 ./var/adm/acct/nite
1 ./var/adm/acct/fiscal
1 ./var/adm/acct/sum
4 ./var/adm/acct
902 ./var/adm
1 ./var/crash
925 ./var
998 .

```

Ta sẽ phải có cấu trúc y nguyên như hạn ngạch đĩa mà ta đã ghi chép lại từ trước cho phép.

☞ Thực hiện lại lệnh **mirror**

Nếu ta thực hiện lại lệnh **mirror** một lần nữa, sẽ không có file nào sẽ được truyền vì vết thời gian và kích thước của các file giữa máy nguồn may-tu-xa và máy gương letuan không được khác nhau một chút nào.

```

[letuan #] cd /tmp/mirror/
[letuan #] mirror
package=def may-tu-xa.hut.edu.vn:/pub/ -> /export/
 data/letuan/ftp/pub/mirror/
No files to transfer

```

### ***XV. 1. h. Thiết lập máy UNIX để sử dụng Usenet***

Usenet là hệ thống phân phát các thông báo, tin tức tương tự như các hệ thống thư điện tử (E-mail), chỉ khác mỗi một điểm là thư điện tử chỉ người nhận mới có quyền nhận và đọc, còn Usenet cho mọi người tự do trao đổi tin tức. Usenet sử dụng giao thức Network News Transfer Protocol (NNTP) để chuyển thông tin giữa rất nhiều các site trên thế giới. Giao thức này được coi như sự phát triển tiếp theo của họ TCP/IP. Do các điều kiện khác nhau, việc sử dụng Usenet ở Việt Nam chưa phát triển, nhưng trong tương lai, có lẽ sẽ có các hình thức thích hợp để giúp ta có thể khai thác các thông tin có ích qua mạng Usenet. Vì thế, có lẽ việc giới thiệu cách sử

dùng Usenet và thiết lập các máy có khả năng tham gia vào Usenet sớm hay muộn sẽ được bạn đọc quan tâm. Thông tin trên Usenet được phân loại vào các nhóm tin tức bố trí theo dạng cấu trúc cây. Một số hàng tin tức cốt lõi được xếp ở hàng trên cùng, sau đó được chia thành các nhóm con, rồi các nhóm này lại được chia thành các nhóm con nữa, v.v... Nói chung, không tồn tại một quy tắc chính thức nào cho việc đặt tên các nhóm tin tức, nhưng theo thói quen thông thường, người ta hay dùng dấu chấm “.” làm dấu phân cách để tạo các tên nhóm. Mỗi dấu chấm như vậy ký hiệu một mức trong cấu trúc cây của hệ thống các nhóm Usenet.

Có lẽ cách khai thác thông tin tối ưu là tìm kiếm thông qua các nhóm tin tức trên Internet. Rất nhiều người coi các nhóm tin tức Usenet là nguồn thông tin chủ yếu của mình. Hiện nay, trên toàn thế giới có khoảng hơn tám nghìn nhóm tin tức như vậy, phản ánh đủ mọi loại thông tin, từ “chối cùn, rế rách” cho tới những điều cao siêu, trừu tượng nhất. Một phần trong các nhóm đó cập nhật hàng ngày một khối lượng dữ liệu khổng lồ lên Internet. Bạn có thể đăng ký vào các nhóm mà bạn quan tâm để giảm thiểu thời gian nhận các bài báo từ những máy chủ chuyên trách được gọi là *news server*. Đối với một số nhóm tin tức, bạn có thể đăng tải trực tiếp các bài của mình, trong khi một số nhóm khác yêu cầu người dùng trước hết gửi bài tới những người phản biện để được kiểm duyệt. Có thể bạn sẽ nhận được trả lời sau khoảng vài ba ngày kể từ lúc gửi câu hỏi đi. Đôi khi, các vấn đề được đặt ra kéo theo một chuỗi các tranh luận nảy lửa, làm cho mọi người tham gia phải huy động hết cả nhiệt tình và kiến thức của mình. Nhưng bạn cũng nên tránh đặt lại những câu hỏi vừa mới được ai đó đưa ra không lâu, kéo lại nhận được những lời chê trách thói lười biếng của bạn vì không chịu theo dõi danh sách các tin tức đã đăng tải. Nhiều nhóm tin tức hoạt động tốt thường duy trì các file được gọi là “Những câu hỏi thường gặp” (Frequently Asked Questions - FAQ), gồm những sưu tập các vấn đề chung. Những file FAQ này rất bổ ích và thường là các gợi ý, chỉ dẫn – thậm chí không phải là ban đầu, mà là chi tiết, từng bước – giúp bạn khắc phục các khó khăn gặp phải trong công việc.

Trong phần này, trước tiên, chúng ta hãy làm quen với cách dùng giao thức NNTP, tức là tập đọc tin tức trên mạng một cách thành thạo mà không cần dùng trình duyệt Netscape. Sau đó, tự chúng ta sẽ thiết lập lấy hai news server để trao đổi tin tức trong cùng các nhóm tin.

- *Sử dụng Network News Transfer Protocol (NNTP), đọc tin tức trên Internet*

NNTP được đưa ra như là tiêu chuẩn cho việc truyền các tin tức theo luồng. Nó xác định giao thức để phân phát, truy vấn, nhận và gửi các bài, tin tức trong cộng đồng Internet. Bạn có thể **telnet** tới một news server để giao tiếp với NNTP, tuy nhiên, trước tiên bạn cần phải đăng ký tại server đó.

Nằm dưới thư mục `/etc`, file có tên `services` chứa một dòng vào cho mỗi một dịch vụ khả dĩ qua TCP/IP, trong đó có các dịch vụ mạng, các dịch vụ đặc biệt của máy (host), và các dịch vụ đặc biệt của UNIX. File này xác định tên dịch vụ, tên cổng dành cho dịch vụ và tên giao thức mà dịch vụ dựa vào đó để hoạt động, cũng như các bí danh mà dịch vụ có thể đòi hỏi. Trước khi bắt tay vào xử lý các dịch vụ mạng, bạn nên nghiên cứu kỹ trang **man** của dịch vụ tương ứng.

#### Tìm địa chỉ cổng được dùng cho NNTP

Giả sử ta muốn biết các tin tức được công bố từ máy có tên `bachkhoaHN`. Sau khi khảo sát lại nội dung của file `services`:

```
[bachkhoaHN #] more /etc/services
```

Bạn sẽ nhìn thấy một dòng vào tương tự như dòng sau:

```
nntp 119/tcp usenet # Network News Transfer
```

Dòng vào trên có nghĩa NNTP sử dụng cổng 119 và giao thức TCP và có thể được tham chiếu đến với bí danh "nntp".

#### Telnet tới news server

Bây giờ bạn đã có địa chỉ cổng cho NNTP (thông thường, đó là địa chỉ quen thuộc 119). Thao tác tiếp theo của bạn là **telnet** tới news server mà bạn đã đăng ký, trong ví dụ này là `hut.edu.vn`, với giả thiết bạn đang làm việc tại máy `daihocquocgia1`.

```
[daihocquocgia 1] telnet hut.edu.vn nntp
```

```
Trying 203.125.253.136 ...
Connected to hut.edu.vn.
Escape character is '^]'.
200 hut InterNetNews NNTP server INN 1.4 20-Mar-93
 ready (posting ok).
```

Sau khi đã kết nối thành công tới news server, bạn có thể gõ lệnh **help** để xem danh sách các lệnh con như sau:

```
help
100 Legal commands
 authinfo user Name|pass Password
 article [MessageID|Number]
 body [MessageID|Number]
 date
 group newsgroup
 head [MessageID|Number]
 help
 ihave
 last
 list [active|newsgroups|distributions|schema]
 listgroup newsgroup
 mode reader
 newgroups yymmdd hhmmss ["GMT"] [<distributions>]
 newnews newsgroups yymmdd hhmmss ["GMT"]
 [<distributions>]

 next
 post
 slave
 stat [MessageID|Number]
 xgtitle [group_pattern]
 xhdr header [range|MessageID]
 xover [range]
```

---

```

xpat header range|MessageID pat [morepat...]
xpath xpath MessageID
Report problems to usenet@hut.edu.vn

```

### Dùng các lệnh con được liệt kê ở trên để đọc tin

#### *Dùng lệnh list để lấy danh sách các nhóm tin tức*

```

list
215 Newsgroups in form "group high low flags".
hut.khoatoan.toancaocap 0000009245 0000008978 y
vatlykythuat.vatlydaicuong 0000035605 0000035181 y
vatlykythuat.vatlychatran 0000000159 0000000141 y
.
.
.
hut.thuvien.sachgiaokhoa 0000000011 0000000007 y
hut.hoisinhvien.class 0000000084 0000000002 y
hut.org.vanphongdoan 0000000006 0000000004 y
.
.
.
hut.congdoan 0000000006 0000000001 y
hut.hoidongkhoahoc 0000000623 0000000001 y
hut.hanhchinhhtonghop 0000000000 0000000001 y

```

Các trường thứ hai, thứ ba chỉ số thứ tự của các bài báo cũ nhất và mới nhất có trong từng nhóm tin tức, trường cuối cùng là cờ, giá trị y cho biết người dùng có thể gửi được bài lên nhóm tin đó.

#### *Dùng lệnh group để chọn nhóm tin tức sẽ xem*

```

group hut.thuvien.sachgiaokhoa
211 5 7 11 hut.thuvien.sachgiaokhoa

```

Câu lệnh trên cho thấy nhóm tin tức hut.thuvien.sachgiaokhoa có 5 bài báo được đánh số thứ tự từ 7 đến 11.

***Dùng lệnh listgroup để xem danh sách các bài báo trong nhóm tin tức***

```
listgroup hut.thuvien.sachgiaokhoa
211 Article list follows
7
8
9
10
11
```

Các bài báo trong nhóm tin trên được liệt kê theo số thứ tự.

***Dùng lệnh article để xem tin trong nhóm tin tức đã chọn***

```
article 7
220 7 <NguyenvanA@hut.edu.vn> article
Path: hut!usenet
From: sachgiaokhoa@thuvien.hut.edu.vn (HUT Book Store)
Newsgroups: hut.thuvien.sachgiaokhoa
Subject: Gio dong cua nghi Tet
Date: 21 Jan 2001 01:00:00 GMT
Organization: Cua hang sach giao khoa, HUT Thu vien
Lines: 17
Sender: sachgiaokhoa@bophanphucvu.hut.edu.vn
Message-ID: <NguyenvanA@hut.edu.vn>
NNTP-Posting-Host: bophanphucvu.hut.edu.vn
Mime-Version: 1.0
```

Cua hang sach giao khoa se mo cua gio hanh chinh phuc vu binh thuong trong dip nghi Tet, tru nhung ngay sau :

January 24, 2001 (dong cua)

January 25, 2001 (dong cua)

January 26, 2001 (chi mo cua tu 8:00AM toi 5:00pm)

Bat dau tu thang 2 nam 2001, ngoai nhung gio hanh chinh binh thuong, cua hang sach giao khoa se mo cua them vao nhung ngay chu nhat tu 02:00PM toi 5:00PM.

```

+=====+
| HUT Cua hang sach giao khoa
| E-mail: sachgiaokhoa@thuvien.hut.edu.vn
| Thu vien nha truong
| WWW: http://www-thuvien.hut.edu.vn
| Hanoi University of Technology
| Tel: 8692113 Fax: 8695213
+=====+

```

Như vậy, sau khi thử sử dụng ví dụ tương tự như trên, ta thấy làm việc với giao thức NNTP cũng không đến nỗi quá khó khăn. Công việc thực sự của người quản trị mạng chỉ bắt đầu ở phần tiếp theo, khi chúng ta phải tự thiết lập lấy các máy chủ UNIX để trao đổi tin tức.

- *Thiết lập hai news servers, tạo các nhóm tin tức mẫu trên mỗi server và bắt chúng cung cấp tin tức cho nhau*

Mỗi một bước thao tác đều phải được thực hiện trên cả hai máy chủ, bachkhoaHN và letuan. Tuy nhiên, để cho việc trình bày được sáng sủa, chúng ta sẽ chỉ thấy những thao tác dưới đây được thực hiện từ phía máy chủ bachkhoaHN. Ta ngầm hiểu rằng, với những sửa đổi thích hợp, chúng cũng được hoàn thành từ phía máy chủ letuan. Các bước cụ thể như sau:

### Login với quyền root

### Tải mã nguồn về máy

Bạn có thể lấy từ trang: <ftp.uu.net/networking/news/nntp/inn> file nén `inn.tar.Z` chứa mã nguồn bằng lệnh **ftp** hay bằng bất cứ tiện ích nào có trong tay. Theo kinh nghiệm của tác giả, nếu dùng **ftp**, tốt nhất bạn nên để ở chế độ nhị phân bằng lệnh con **b** (viết tắt của **binary**).

### Giải nén rồi bung mã nguồn ra

Bạn nên cài đặt chương trình INN vào thư mục `/export/inn`, thay vì vào thư mục `/usr` do hạn chế về dung lượng đĩa được phân bổ sẵn cho `/usr` từ khi cài đặt hệ thống.

```
[bachkhoaHN #] mkdir /export/inn
[bachkhoaHN #] cd /export/inn
[bachkhoaHN #] zcat inn.tar.Z | tar xvf -
```

### Tạo bản sao theo tùy biến của file config.dist

Ta chép file cấu hình `config.data` từ file nguyên bản `config.dist`, sau đó căn chỉnh nội dung của nó cho phù hợp với môi trường tại chỗ. Trước khi sửa đổi file cấu hình ta phải đổi chủ quyền truy nhập file.

```
[bachkhoaHN #] cd config
[bachkhoaHN #] cp config.dist config.data
[bachkhoaHN #] chmod o+w config.data
[bachkhoaHN #] vi config.data
```

Thường là ta phải thay đổi vị trí đặt file nếu không muốn dùng đường dẫn mặc định. Bạn cũng cần thay đổi cả tiện ích UNIX không thích hợp với chương trình INN bằng một tiện ích tương tự mà hệ thống của bạn hỗ trợ. Trong ví dụ này, ta làm như sau:

Thay dòng:



```
_PATH_EGREP /usr/local/bin/egnugrep
```

bằng dòng:

```
_PATH_EGREP /bin/egrep
```

Do trong ví dụ này ta cần cài đặt mọi thứ trong thư mục `/export`, nên các bước thực hiện tương ứng sẽ là:

Thay đường dẫn:

```
/usr/local/*
```

bằng:

```
/export/usr/local/*
```

Có tất cả khoảng 45 dòng vào cần được thay thế một cách tương tự. Và bạn còn cần tạo ra một số thư mục trống. Tất cả các phiên nhiều này sẽ tự nhiên mất đi, nếu bạn dùng đường dẫn mặc định, nhưng trong trường hợp đó, bạn mất khả năng tùy biến.

```
[bachkhoaHN #] mkdir /export/usr
[bachkhoaHN #] cd /export/usr
[bachkhoaHN #] mkdir local
[bachkhoaHN #] cd local
[bachkhoaHN #] mkdir news etc bin man lib
```

Ta còn phải biến thành chú thích câu lệnh sau trong file nguồn `icd.c`,

```
[bachkhoaHN #] cd /export/inn/innd
[bachkhoaHN #] chmod o+w icd.c
[bachkhoaHN #] vi icd.c
```

câu lệnh là:

```

if (NGfind("control") == NULL || NGfind("junk") ==
 NULL) {
 syslog(L_FATAL, "%s internal no control and/or
 junk group", LogName);
 exit(1);
}

```

### Hoàn chỉnh chương trình INN và cài đặt nó vào thư mục đích

```

[bachkhoaHN #] cd /export/inn
[bachkhoaHN #] make world install

```

Quy tắc: Không được quên cập nhật các dòng vào của file định thời lịch (lệnh **cron**). Ta đang dùng **csh** và nên đặt lịch sử lệnh (xem lại chương VII “Môi trường Shell và lập trình Shell”).

### Sửa đổi các file điều khiển

Trước khi bắt đầu khởi động daemon **innd**, ta cần tạo và sửa đổi nội dung một số file điều khiển. Bảng dưới đây liệt kê các file điều khiển chủ yếu nhất hay được dùng và tính năng của chúng:

|                    |                                                                 |
|--------------------|-----------------------------------------------------------------|
| <b>active</b>      | Tất cả các nhóm tin tức đang hoạt động và trạng thái của chúng. |
| <b>host.nnrp</b>   | Các host cung cấp tin cho news site tại chỗ.                    |
| <b>nnrp.access</b> | Chủ quyền truy vấn tin tức.                                     |
| <b>newsfeeds</b>   | Các bài báo cung cấp cho các site cụ thể.                       |
| <b>expirectl</b>   | Điều kiện về thời gian lưu lại tin.                             |

Một số ít các file điều khiển khác, ít quan trọng hơn, và ta có thể tham khảo chi tiết được về chúng, nằm trong thư mục **/export/inn/samples**.

#### ❖ active

File này liệt kê tất cả các nhóm tin tức đang hoạt động cùng với trạng thái của chúng theo cú pháp:

```
<newsgroup name> <oldest article number> <newest
 article number> <group posting status>
```

với trường newsgroup name chứa tên các nhóm tin tức, trường oldest article number chứa số tin đã được đọc, trường newest article number - số tin mới vừa được đăng tải, và trường cuối cùng cho biết điều kiện đăng tải tin trong nhóm tương ứng.

Ví dụ ta có file active như sau:

***Trên máy bachkhoaHN :***

|                          |           |           |   |
|--------------------------|-----------|-----------|---|
| hut.lab.networklab       | 000000000 | 000000000 | y |
| hut.vatlykythuat.lab     | 000000000 | 000000000 | y |
| hcmc.bachkhoa.tinhoc     | 000000000 | 000000000 | y |
| vn.hanoi.uybannhandan    | 000000000 | 000000000 | y |
| hut.khoatoan.toantin     | 000000000 | 000000000 | y |
| hut.khoahoa.hoadau       | 000000000 | 000000000 | y |
| hut.khoadien.hethongdien | 000000000 | 000000000 | y |

***Trên máy letuan :***

|                            |           |           |   |
|----------------------------|-----------|-----------|---|
| hut.lab.vatlybandan        | 000000000 | 000000000 | y |
| hut.lab.softwarebank       | 000000000 | 000000000 | y |
| hut.lab.database           | 000000000 | 000000000 | y |
| hcmc.daihocquocgia.tunhien | 000000000 | 000000000 | y |
| hut.daotao.sinhvien        | 000000000 | 000000000 | y |

Trường đầu tiên biểu diễn bảy nhóm tin tức mới được tạo trên máy bachkhoaHN. Đồng thời, trên máy letuan, trường đầu cũng chứa năm bản ghi biểu thị cho năm nhóm tin tức mới được tạo. Vì các nhóm trên mới được tạo ra, chưa có một tin nào tồn tại trong các nhóm đó, ta phải điền vào các vị trí của trường thứ hai và trường thứ ba các chữ số 0. Cuối cùng, giá trị y ở trường cuối cùng cho biết có

thể gửi đăng tin vào nhóm tin tức này. Các giá trị của trường cuối được hệ thống lại trong bảng dưới đây.

|        |                                                                                                 |
|--------|-------------------------------------------------------------------------------------------------|
| y      | Nhóm tin được chấp nhận và có thể gửi đăng tin được.                                            |
| n      | Nhóm tin tồn tại, nhưng không ở tại site này (không nhận/ không gửi đăng).                      |
| m      | Nhóm tin được biên tập (chấp nhận tin nhưng tin còn được gửi đi nơi khác - để kiểm duyệt - đã). |
| x      | Nhóm tin được chấp nhận, nhưng không cho phép gửi đăng tin.                                     |
| j      | Không giữ các bài, tin, mà chỉ xử lý cho qua.                                                   |
| =alias | Nhóm tin đã được đổi tên thành alias                                                            |

#### ❖ hosts.nntp

File này được đọc bởi daemon **innd** để lấy ra danh sách các host dùng giao thức NNTP cung cấp tin tức cho site tại chỗ của Usenet. Server đọc file đó khi khởi động hệ thống hoặc khi được daemon **ctlinnd** ra lệnh. Để có được sự giải thích tường tận hơn, bạn có thể tham khảo trang **man** của file **hosts.nntp**. Trong ví dụ cụ thể này, bạn cần phải biến dòng vào:

```
news.foo.com
```

thành chú thích và thêm vào hai dòng vào mới:

```
bachkhoaHN.hut.edu.vn
letuan.hut.edu.vn
```

cho hai máy **bachkhoaHN.hut.edu.vn** và **letuan.hut.edu.vn** là các host chạy chương trình INN.

#### ❖ nnrp.access

Nội dung của file này là liệt kê các máy được phép đọc tin tức từ máy chủ đang xét đến. Nhớ rằng ta chỉ xác định điều khiển truy nhập cho những site NNTP nào không được xử lý bởi daemon chính **innd** của InterNetNews. Daemon **nnrpd** của

server đọc file `nnrp.access`, trong khi bản thân nó lại được tái sinh bởi `innd`. Dòng vào của file phải gồm năm trường, cách nhau bởi dấu hai chấm ":", và tuân theo cú pháp như sau:

`host:perm:username:password:patterns`

Sau đây là tổng kết nội dung của file điều khiển `nnrp.access`.

|                       |                                                           |
|-----------------------|-----------------------------------------------------------|
| <code>host</code>     | Tên lỏng hoặc địa chỉ IP.                                 |
| <code>perm</code>     | R là đọc; P là gửi đăng tin.                              |
| <code>username</code> | Tên người dùng, dùng để kiểm định trước khi gửi đăng tin. |
| <code>password</code> | Mật khẩu, dùng để kiểm định trước khi gửi đăng tin.       |
| <code>patterns</code> | Các mẫu nhóm tin có thể đọc được hay không đọc được.      |

Trường đầu tiên xác định các tên hoặc địa chỉ Internet của tập hợp các `hosts`. Trường thứ hai là tập hợp các chữ cái dùng để chỉ các quyền được gán cho người dùng, các giá trị của `perms` có ý nghĩa như sau: R – người dùng có thể lấy được tin về, P cho biết người dùng có thể gửi đăng tin. Trường thứ ba và thứ tư xác định tên tài khoản người dùng và mật khẩu mà họ cần dùng cho việc kiểm tra tư cách thành viên của bản thân trước khi server nhận bất cứ tin bài nào. Trường cuối cùng là tập hợp các mẫu ký tự xác định các nhóm tin tức mà người dùng được phép truy nhập. Bộ chương trình cài đặt INN có cung cấp trang `man` cho file `nnrp.access` để tham khảo chi tiết.

Trong ví dụ của chúng ta, bạn muốn tất cả các `host` trong vùng `hut.edu.vn` đều có thể cả đọc lẫn gửi tin, vì thế, bạn cần làm như sau:

- Biến thành chú thích dòng vào mặc định:

`*.foo.com:Read Post:::*`

và thêm vào một dòng vào sau:

`*.hut.edu.vn:Read Post:::*`

❖ newsfeeds

File này xác định các tin đi đến sẽ được phân phát cho các site khác như thế nào. Dạng cú pháp của dòng vào được quy định như sau:

```
sitename[/exclude,exclude,...]\
:pattern,pattern...[/distrib,distrib...]\
:flag,flag...\
:param
```

Với:

**sitename**

Tên của site mà tin tức mới lấy làm đích để được gửi đến.

**patterns**

Xác định các nhóm tin nào sẽ gửi tin tới site đã cho và thực chất là tạo ra một “danh sách đăng ký” cho site đó.

**flag**

Xác định các thông số phụ khác.

**param**

Ý nghĩa của giá trị trường này phụ thuộc vào kiểu loại cung cấp tin.

Dấu “\” trong câu cú pháp của dòng vào trên chỉ có nghĩa nối dòng, để cho ta biết câu lệnh chưa kết thúc. Ý nghĩa của các thông số cụ thể trong nội dung của file newsfeeds được liệt kê trong bảng dưới. Cũng như đối với các file khác, file newsfeeds cũng được hỗ trợ qua trang **man**.

|           |                                                                                                                              |
|-----------|------------------------------------------------------------------------------------------------------------------------------|
| size      | Kích thước cực đại của bản tin.                                                                                              |
| A<flags>  | Các kiểm tra bản tin<br>d: phải có tiêu đề phân phát (distribution header).<br>p: không kiểm tra tên site trong Path header. |
| B<hi/low> | Đưa vào bộ đệm trước khi ghi ra đầu xuất.                                                                                    |
| H<count>  | Số đếm bước nhảy của bản tin (qua bao nhiêu host).                                                                           |
| I<size>   | Kích thước bộ đệm nội tại cho việc cung cấp file.                                                                            |
| Nm        | Các nhóm tin được biên tập có tên chứa patterns.                                                                             |

|        |                                                                                                                                                                                                                                                                                                                                                                                   |
|--------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Nu     | Các nhóm tin không được biên tập có tên chứa <b>patterns</b> .                                                                                                                                                                                                                                                                                                                    |
| Ssize  | Bắt đầu hàng chữ nếu số byte dồn lại lớn hơn size.                                                                                                                                                                                                                                                                                                                                |
| Ttype  | Các kiểu loại cung cấp tin<br>f: file<br>m: phễu; thông số đặt tên cho dòng vào thực sự<br>p: đường ống tới chương trình<br>c: gửi tới kênh nhập chuẩn của các tiến trình con cho param<br>x: giống như c, nhưng còn xử lý các lệnh trên nhập chuẩn                                                                                                                               |
| Witems | Ghi như thế nào<br>b: theo số byte của bản tin<br>f: theo đường dẫn đầy đủ<br>g: vào nhóm tin tức đầu tiên<br>m: Message-I<br>n: theo đường dẫn tương đối<br>s: site cung cấp bản tin<br>t: theo thời gian nhận tin<br>*: theo tên của phễu nhận tin vào<br>N: theo tiêu đề của các nhóm tin tức<br>H: theo tất cả các tiêu đề<br>O: tổng quan dữ liệu<br>R: sao chép lại dữ liệu |

Trong ví dụ mà ta vẫn theo dõi từ trước đến bây giờ:

**Trên máy bachkhoaHN, ta thêm vào dòng lệnh sau:**

```
letuan.hut.edu.vn\
:hut.*,hcmc.*\
:Tf,Wnm\
:letuan.hut.edu.vn
```

**Còn trên máy letuan, ta cũng thêm tương ứng:**

```
bachkhoaHN.hut.edu.vn\
:hut.*,hcmc.*,vn.*\
:letuan.hut.edu.vn
```

```
:Tf,Wnm\
:bachkhoaHN.hut.edu.vn
```

### ❖ expire.ctl

File này được sử dụng để quản lý các bản tin trong phạm vi của một site tin tức. Cú pháp đối với dòng vào của file có dạng tập hợp các trường với các định nghĩa như sau:

```
/remember/:<keep>
<patterns>:<modflag>:<keep>:<default>:<purge>
```

|                   |                                                                                                       |
|-------------------|-------------------------------------------------------------------------------------------------------|
| /remember/:<keep> | Số ngày giữ lại lịch sử định danh (ID) của bản tin.                                                   |
| pattern           | Các mẫu ký tự theo kiểu tự do cho nhóm tin. Phải chứa dòng mặc định "*:A:...".                        |
| modflags          | Một trong số M, U hoặc A, chỉ tương ứng các nhóm có được biên tập, không biên tập và tất cả các nhóm. |
| keep              | Số ngày tối thiểu lưu giữ bản tin.                                                                    |
| default           | Số ngày mặc định lưu giữ bản tin.                                                                     |
| purge             | Xóa bỏ bản tin sau số ngày được xác định ở đây.                                                       |

Đối với `keep`, `default` và `purge`, ta có thể dùng các số ngày có dấu phẩy động hoặc dùng từ `never`. Một file ví dụ có dạng:

```
/remember/:14
*:A:1:5:10
hut.*:A:never:never:never
```

Dòng thứ nhất chỉ ra rằng định danh của bản tin được lưu giữ hai tuần (14 ngày), dòng thứ hai cho biết – nếu không có thay đổi gì tiếp theo – nhóm tin và kiểu loại của nhóm tin (được biên tập hay không) sẽ được lưu lại trong khoảng thời gian ít nhất một ngày, bình thường là năm ngày và nhiều nhất là mười ngày. Dòng cuối cùng cho thấy nhóm `hut.*` sẽ tồn tại vô thời hạn. Ta cũng nên nhớ rằng định



danh của bản tin có thể giúp server từ chối tiếp nhận hay loại bỏ bản tin nếu các phía cung cấp lại mời chào một lần nữa.

### Tạo hai file cần thiết cho dbmaw

```
[bachkhoaHN #] cat /dev/null > history.dir
[bachkhoaHN #] cat /dev/null > history.pag
```

### Khởi động daemon tin tức

```
[bachkhoaHN #] cd /export/usr/local/etc
[bachkhoaHN #] innd
```

Kiểm tra lần nữa để tin chắc rằng daemon **innd** đã thực sự hoạt động.

```
[bachkhoaHN #] ps -aux|grep innd
root 4828 0.0 1.5 32 220 p1 S 03:43 0:00 grep innd
root 4823 0.0 0.0 632 0 ? IW 03:42 0:00 innd
```

### Kiểm tra các news server

Bạn có thể **telnet** tới các news server, giống như trong mục sử dụng NNTP đọc tin tức, để kiểm tra xem các nhóm tin mặc định mà bạn đã tạo ra có hoạt động hay không.

### Kiểm tra máy bachkhoaHN.hut.edu.vn

```
[bachkhoaHN #] telnet bachkhoaHN nntp
Trying 203.125.253.116 ...
Connected to bachkhoaHN.hut.edu.vn.
Escape character is '^]'.
200 bachkhoaHN.hut.edu.vn InterNetNews server INN
 1.4 22-Dec-93 ready
list
215
hut.lab.networklab 0000000000 0000000000 y
```

```

hut.vatlykythuat.lab 0000000000 0000000000 y
hcmc.bachkhoa.tinhoc 0000000000 0000000000 y
vn.hanoi.uybannhandan 0000000000 0000000000 y
hut.khoatoan.toantin 0000000000 0000000000 y
hut.khoahoa.hoadau 0000000000 0000000000 y
hut.khoadien.hethongdien 0000000000 0000000000 y
.
quit
205

```

### ***Kiểm tra máy letuan.hut.edu.vn***

```

[bachkhoaHN #] telnet letuan nntp
Trying 203.125.253.104 ...
Connected to letuan.hut.edu.vn.
Escape character is '^]'.
200 letuan.hut.edu.vn InterNetNews server INN 1.4
 22-Dec-93 ready
list
215
hut.lab.vatlybandan 0000000000 0000000000 y
hut.lab.softwarebank 0000000000 0000000000 y
hut.lab.database 0000000000 0000000000 y
hcmc.daihocquocgia.tunhien 0000000000 0000000000 y
hut.daotao.sinhvien 0000000000 0000000000 y
.
quit
205

```

Connection closed by foreign host.

Bạn có thể dùng trình duyệt Netscape hoặc StarOffice làm bộ đọc tin để duyệt tin tức. Trong danh sách duyệt các server bạn sẽ thấy hiện tên cả hai news server bachkhoaHN lẫn letuan, không phụ thuộc vào việc bạn đang kết nối với máy

nào. Bạn có thể kích nút chuột lên siêu liên kết tương ứng để đọc tin. Nếu chịu khó chờ một chút, bạn sẽ nhận thấy cả hai news server trên đều có các nhóm tin giống hệt, bởi vì các file điều khiển *newsfeeds* của chúng cho phép chúng có thể phân phát tin tức lẫn nhau.

### ***XV. 1. i. IPv6***

#### ***■ Nhu cầu mở rộng số địa chỉ Internet***

Trong tất cả các nội dung đã trình bày ở trên, chúng ta làm việc với các địa chỉ IP 32 bit, nghĩa là theo chuẩn IPv4. Về lý thuyết, với IPv4 chúng ta có thể duy trì tất cả  $2^{32}$ , tức là cỡ 4,2 tỉ địa chỉ. Số lượng địa chỉ này đến nay rõ ràng là không còn đủ thỏa mãn nhu cầu đang gia tăng một cách bùng nổ các ứng dụng Internet. Mặt khác, như đã thấy trong quy tắc thiết lập địa chỉ IPv4, một số lớn các địa chỉ được dành riêng cho các ứng dụng đặc biệt hay để dự trữ cho các quốc gia phát triển và công ty lớn, còn làm giảm đi đáng kể khoảng địa chỉ thực sự có thể sử dụng được. Các chính vụ hệ thống nhằm tiết kiệm thời gian tìm kiếm địa chỉ như BOOTP (Bootstrap Protocol), DHCP (Dynamic Host Configuration Protocol) chỉ có tác dụng giảm nhẹ sự căng thẳng trong việc xác định địa chỉ cho các máy tính mà thôi. Chính vì lẽ đó, không chỉ ở Việt Nam, mà cả ở phạm vi toàn thế giới, khoảng 66 % số công ty dùng các địa chỉ IP không chính thức. Thêm nữa, việc cấp phát các địa chỉ này đã diễn ra một cách bị động theo sự phát triển công nghệ thông tin, chứ không dựa trên nền tảng kế hoạch tổng thể, nên đã xảy ra tình trạng bất hợp lý nghiêm trọng. Phần lớn các mạng các lớp A, B đều đã được phân phối cho các quốc gia sớm phát triển Internet, nên các quốc gia chậm chân hơn đã bị thiếu địa chỉ cho nhu cầu phát triển của mình. Việt Nam cũng nằm trong tình cảnh trên, trong “xã hội tin học” nước nhà ta chỉ còn thấy hoạt động của các mạng phần lớn là lớp C, như 203.162.5.0 của Tổng công ty Bưu chính Viễn thông, 202.167.116.0 của VDC, 202.192.0.0 của Ngân hàng Công thương Việt Nam, 192.168.127.0 của Trường Đại học Bách khoa Hà nội, v.v...

Cùng với sự phát triển của quá trình số hóa trong khoa học kỹ thuật, sản xuất và đời sống, thời điểm hiện nay đánh dấu sự phổ biến kỹ thuật không dây, số các thiết bị có gắn chip thông minh tăng lên ngày càng nhanh chóng, càng đòi hỏi cấp thiết phải có thêm nhiều không gian địa chỉ mới. Thêm nữa, các địa chỉ mới còn phải

đáp ứng thêm nhiều yêu cầu khắt khe về bảo mật, về các chức năng mới được hỗ trợ mà trong thiết kế IPv4 người ta đã không dự tính hết được.

Tất cả các nguyên nhân nêu trên đã khiến nhân loại phải đầu tư nghiên cứu một chuẩn địa chỉ Internet mới. Kết quả đầu tiên theo hướng này được công bố năm 1994 trong tài liệu Request For Comments (RFC) 1752, nhưng phải một năm sau, các kiến nghị chính thức mới được đưa ra, và vào năm 1996, các nguyên tắc chính về IPv6 được chấp nhận. Các thông tin cơ bản về IPv6 được chứa đựng trong các RFC sau (dấu \* đánh dấu các RFC được Solaris hỗ trợ):

| Tên        | Mô tả                                                                                                                                 |
|------------|---------------------------------------------------------------------------------------------------------------------------------------|
| RFC 1883   | Giao thức cơ bản IPv6                                                                                                                 |
| RFC 1884   | Đặc điểm kỹ thuật của địa chỉ                                                                                                         |
| RFC 1885   | Mô tả giao thức kiểm tra ICMP (chung)                                                                                                 |
| RFC 1886 * | Xác định các vấn đề liên quan tới DNS mở rộng                                                                                         |
| RFC 1933 * | Cơ chế chuyển đổi IPv6 cho host và router                                                                                             |
| RFC 1981 * | Nhận biết MTU đường dẫn cho IPv6                                                                                                      |
| RFC 2019 * | Chuyển các gói IPv6 bằng Giao diện dữ liệu phân bố nhờ cáp quang (Fiber Distributed Data Interface - FDDI), hôm na là dùng cáp quang. |
| RFC 2080 * | Giao thức thông tin phân đường (Routing Information Protocol - RIP) cho IPv6                                                          |
| RFC 2292 * | API các socket cải tiến cho IPv6.                                                                                                     |
| RFC 2373 * | Kiến trúc địa chỉ IPv6.                                                                                                               |
| RFC 2428 * | Mở rộng FTP cho IPv6.                                                                                                                 |
| RFC 2452 * | Cơ sở thông tin quản lý IPv6 cho TCP.                                                                                                 |
| RFC 2454 * | Cơ sở thông tin quản lý IPv6 cho UDP.                                                                                                 |
| RFC 2460 * | Đặc điểm kỹ thuật IPv6.                                                                                                               |
| RFC 2461 * | Phát hiện láng giềng cho IPv6                                                                                                         |
| RFC 2462 * | Tự động cấu hình địa chỉ không chính thức IPv6.                                                                                       |
| RFC 2463 * | ICMP cho IPv6 (ICMPv6)                                                                                                                |
| RFC 2464 * | Một phương pháp truyền các gói IPv6 qua các mạng Ethernet.                                                                            |



địa chỉ viết sai kiểu BAED : : ABCD : : 45 : FEDC, ta sẽ không thể biết được nhóm hexa ABCD nằm ở đâu, hay các dãy chữ số 0 có chiều dài thế nào. Cuối cùng, để tránh các lỗi biến đổi khi chuyển từ số thập phân sang số hexa, với mục đích chuyển đổi từ các địa chỉ IPv4 cũ sang địa chỉ IPv6 mới mà không làm xáo trộn trật tự đã được thiết lập, IPv6 cho phép viết theo kiểu hỗn hợp: 96 bit đầu viết theo mã hexa, 32 bit cuối có thể viết bằng số thập phân. Ví dụ, vào một thời điểm truy nhập Internet qua nhà cung cấp dịch vụ VDC, máy tính của tác giả được gán địa chỉ IPv4 là 202.167.116.187, nếu chuyển sang IPv6 sẽ là 0000:0000:0000:0000:0000:0000:202.167.116.187, hay viết tắt là : : 202.167.116.187. Ta thấy, mặc dù không gian địa chỉ trong IPv6 được tăng lên gấp nhiều lần so với IPv4, cách viết này giúp cho những người quản trị mạng chỉ đơn giản thêm hai dấu hai chấm “:” vào trước địa chỉ IP đã tồn tại khi phải chuyển đổi địa chỉ sang IPv6.

Liên quan tới các địa chỉ mới này, người đọc còn cần tìm hiểu thêm về giao thức IPv6, với tên gọi ban đầu trong các RFC là *IPng* (Internet Protocol Next Generation), cấu trúc cùng với cách gửi các gói thông tin IPv6 trong mạng máy tính và sử dụng các phần đầu (header) của chúng khi các gói đó được chuyển qua các lớp của mô hình mạng. Những nội dung trên không được trình bày trong phạm vi của cuốn sách này.

#### ■ *Các bước chuyển địa chỉ từ IPv4 sang IPv6 đối với Linux*

Do Linux là các bản UNIX miễn phí, mã nguồn mở, nên mọi phát triển, mọi sản phẩm mới đều lấy Linux làm môi trường thử nghiệm. Mặt khác, đối với đa số người dùng, đó cũng là môi trường UNIX thông dụng nhất. Bởi vậy, cũng không có gì đáng ngạc nhiên nếu IPv6 được đưa vào Linux sớm nhất. Tuy vậy, việc chuyển sang dùng IPv6 không đơn thuần là viết lại địa chỉ Internet theo chuẩn mới. Trong mục này, chúng ta hãy theo dõi các thao tác nhằm chuyển Linux từ IPv4 sang IPv6. Tại thời điểm hiện nay, các bản Linux mới được phát hành tự thân chúng đã sẵn sàng hoạt động với IPv6 (SuSE Linux v.6.4, v.7.0; RedHat v.7.0, Debian GNU/Linux v.2.2, Kondara MNU/Linux (Jirai), TurboLinux v.6.0 hoặc muộn hơn, bản tiếng Nhật, Vine Linux), ta hầu như không phải làm gì nhiều. Đối với các bản khác hoặc phiên bản sớm hơn, điều kiện cần để chuyển sang dùng IPv6 là cập nhật kernel của Linux,

vmlinux, v.2.2.16 hoặc muộn hơn, có cài đặt glibc 2.1 và một số gói phần mềm khác. Giả sử ta đang có bản RedHat v.6.2 và cần cài đặt máy làm việc với IPv6. Các bước phải thực hiện như sau:

Trước hết, cài đặt RedHat v.6.2 vào máy với chế độ mạng IPv4. Bản RedHat v.6.2 đã sẵn có thư viện glibc 2.1, kernel vmlinux phiên bản 2.2.15. Cấu hình máy chọn theo nhu cầu: minimal system, có hay không in, có hay không kết nối Internet qua LAN, Xfree86, v.v..

**Cài đặt modutils mới cho Linux.** Nếu dùng kernel nguyên khối (liền, không dùng các mô đun như các bộ phận cấu thành của kernel), hoặc đã có modutils v.2.1.121 hay muộn hơn, ta có thể bỏ qua bước này để thực hiện bước sau. Nội dung của modutils bao gồm:

| Lệnh     | Đường dẫn      | Mô tả                                                          |
|----------|----------------|----------------------------------------------------------------|
| insmod   | /sbin/insmod   | Cài đặt mô đun nạp được.                                       |
| rmmod    | /sbin/insmod   | Gỡ bỏ mô đun. Đường dẫn liên kết.                              |
| lsmod    | /sbin/lsmod    | Liệt kê các mô đun.                                            |
| ksyms    | /sbin/ksyms    | Liệt kê các ký hiệu của kernel được xuất ra.                   |
| modprobe | /sbin/modprobe | Kiểm tra tự động các mô đun nạp được.                          |
| depmod   | /sbin/depmod   | Xử lý tự động các mô đun nạp được.                             |
| kernelld | /sbin/kernelld | Thực hiện các thao tác của kernel trong không gian người dùng. |
| genksyms | /sbin/genksyms | Bộ sinh phiên bản.                                             |
| modinfo  | /sbin/modinfo  | Hiển thị thông tin về các mô đun.                              |

Các lệnh cụ thể để giải nén, cấu hình và cài đặt như sau:

- Chuyển về thư mục sẽ chứa mã nguồn:

```
cd /usr/src
```

- Giải nén modutils mới của kernel:

---

```
tar xzf src-path/modutils-version -C /usr/src
```

- Tạo hay ghi đè liên kết hình thức. Liên kết này sẽ cần thiết về sau:

```
ln -sf modutils-version modutils
```

- Chuyển về thư mục modutils của kernel:

```
cd modutils
```

- Cấu hình modutils của kernel:

```
./configure
```

- Biên dịch modutils của kernel:

```
make clean;make
```

- Thực hiện lệnh cài đặt **install**:

```
make install
```

- Nếu có sử dụng các mô đun và kích hoạt kmod trong kernel, ta phải làm mất khả năng khởi động kerneld, thường nằm ở trong file /etc/init.d/rc.sysinit, bằng cách tương tự như sau:

```
Kerneld useful only for versions earlier 2.1.90,
after version 2.1.90 uses built-in kmod (faster)
if [-f /proc/sys/kernel/modprobe]; then
 echo "Don't start kerneld, using kernel built-in kmod"
else
 if [-x /sbin/kerneld -a -n "$KERNELMODULES"]; then
 /sbin/kerneld > /dev/nul 2>&1
 STARTKERNELD=yes
 fi
fi
```



- Khởi động lại máy để kích hoạt modutils mới:

```
reboot
```

Cài đặt kernel mới, sẵn sàng cho IPv6 . Ta hiểu kernel bao gồm:

| Tên file    | Đường dẫn                    | Mô tả             |
|-------------|------------------------------|-------------------|
| vmlinux-ip6 | /boot/vmlinux-ip6            | Kernel của Linux  |
| *.o         | /lib/modules/ <i>version</i> | Các mô đun kernel |

Các lệnh cụ thể để giải nén, cấu hình:

- ♦ Đối với RedHat 6.2

- Cài đặt gói mã nguồn của kernel:

```
rpm -ihv path-to-rpm-packages/kernel-source-version.rpm
```

- Chuyển tới thư mục chứa mã nguồn của kernel:

```
cd /usr/src/linux
```

- Chép file config nguyên bản (ta chọn một trong số các file nguyên bản được cung cấp sẵn trong mã nguồn, phụ thuộc vào loại CPU được dùng trên máy):

```
cp configs/kernel-version-cputype.config .config
```

- Cấu hình kernel cho IPv6 bằng lệnh:

```
make config
```

hay:

```
make menuconfig
```

rồi kích hoạt các yếu tố của tùy chọn cấu hình mạng như sau trước khi biên dịch:

| Tùy chọn kernel   |                                      | Giá trị lựa chọn                                          |
|-------------------|--------------------------------------|-----------------------------------------------------------|
| Các tùy chọn mạng | The IPv6 protocol                    | yes (hoặc module, nhưng nếu là module thì có thể gây lỗi) |
|                   | IPv6: enable EUI-64 token format     | yes                                                       |
|                   | IPv6: disable provider based address | yes                                                       |

- Cập nhật file Makefile cho đúng số của phiên bản, nếu không, sẽ có xung đột phiên bản của các mô đun. Thay đổi giá trị biến:

```
EXTRAVERSION = -5.0
```

thành:

```
EXTRAVERSION = -5.0-IPv6
```

- ♦ Đối với các bản Linux khác

- Chuyển tới thư mục chứa mã nguồn của kernel:

```
cd /usr/src/linux
```

- Loại bỏ liên kết hình thức tới mã nguồn của kernel cũ:

```
rm linux
```

- Giải nén mã nguồn của kernel mới:

```
tar xzf src-path/linux-version -C /usr/src
```

- Nếu mã nguồn kernel ta đang dùng cũ hơn phiên bản trên, ta phải cập nhật thêm các phần sửa lỗi:

```
gzip -cd src-path/patch-2.y.x.gz | patch -p0
```

- Đổi tên thư mục mã nguồn mới tạo ra để dễ phân biệt:

```
mv linux linux-version
```

- Tạo lại liên kết hình thức cần thiết:

```
ln -s linux-version linux
```

- Chuyển con trỏ vào trong thư mục mã nguồn của kernel:

```
cd linux
```

- Cấu hình kernel cho IPv6 bằng lệnh:

```
make config
```

hay:

```
make menuconfig
```

rồi kích hoạt trước khi biên dịch các tùy chọn cấu hình được liệt kê trong bảng sau. Cột đầu tiên là các tùy chọn được chia theo chức năng, với các trường hợp con tương ứng được nêu lên trong cột thứ hai. Chú ý: phải lựa chọn thêm các tùy chọn cấu hình phù hợp với hệ thống của bạn!

| Tùy chọn kernel                       |                                                                              | Giá trị lựa chọn                                                                                      |
|---------------------------------------|------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------|
| Các tùy chọn mức độ hoàn hảo mã nguồn | Hỏi khi phát triển / đang dở mã nguồn đối với các trình điều khiển (driver). | yes                                                                                                   |
| Hỗ trợ mô đun nạp được                | Có hỗ trợ mô đun nạp được                                                    | no                                                                                                    |
|                                       | Đặt thông tin phiên bản lên tất cả các biểu tượng cho mô đun                 | yes (phải chọn giá trị này để tránh nhầm mô đun, trong file Makefile xem dòng chứa biến EXTRAVERSION) |
|                                       | Bộ nạp mô đun cho kernel                                                     | yes (bình thường không còn sử dụng <b>kernel</b> )                                                    |

|                                                       |                                          |                                                                                                                                                 |
|-------------------------------------------------------|------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| Thiết lập chung                                       | Hỗ trợ chọn chế độ video                 | yes (để có thể dùng được chế độ 50 dòng ở console, để biết thêm chi tiết xem các tùy chọn trong file <code>lilo.conf: [vga = extended]</code> ) |
| Các tùy chọn mạng<br><br>Các tùy chọn mạng (tiếp tục) | Socket gói                               | yes (hoặc module), quan trọng đối với lệnh <b>tcpdump</b> !                                                                                     |
|                                                       | Socket vùng UNIX                         | yes (không làm việc như mô đun!)                                                                                                                |
|                                                       | Mạng TCP/IP                              | yes                                                                                                                                             |
|                                                       | Giao thức IPv6                           | yes (hoặc module, nhưng có thể gây lỗi)                                                                                                         |
|                                                       | IPv6: có định dạng token EUI-64          | yes                                                                                                                                             |
|                                                       | IPv6: tắt địa chỉ cơ sở của nhà cung cấp | yes                                                                                                                                             |
| Các hệ thống file                                     | Hỗ trợ hệ thống file/proc                | yes                                                                                                                                             |
| Kernel hacking                                        | Khóa Magic sysrq                         | yes (rất có ích khi bị treo máy, đọc file <code>sysrq.txt</code> để biết thêm chi tiết)                                                         |

#### ▣ Các lệnh cụ thể để biên dịch và cài đặt:

- Tạo bản sao lưu cấu hình kernel:

```
cp .config ../config-version-counter
```

- Bạn cũng có thể tự tạo ra phiên bản con mới của kernel bằng cách sửa đổi file `Makefile` , ví dụ thay hoặc thêm biến:

```
EXTRAVERSION = -IPv6
```

- Biên dịch kernel, nếu kernel được dựng quá lớn, ta dùng ảnh kernel bzImage thay vì zImage:

```
make dep; make clean; make {zImage | bzImage}
<; make modules>
```

Nếu gặp lỗi trong quá trình biên dịch, bạn hãy kiểm tra lại cấu hình kernel của mình và thử các tùy chọn khác. Đôi khi kernel mới có vẻ hơi kỳ lạ (do ta chưa quen?). Một kinh nghiệm nữa là bạn đầu tốt nhất nên dựng kernel mới không dùng các mô đun.

- Chép lại và đặt tên cho kernel:

```
cp arch/i386/boot/{zImage | bzImage} /boot/vmlinuz-
version
```

- Nếu dựng kernel với các mô đun tham gia như các thành phần, ta phải cài đặt chúng và đồng thời cũng dựng chúng vào đĩa RAM khởi tạo (phần được đọc trước để khởi động Linux):

```
make modules_install
mkinitrd /boot/initrd-2.2.14-5.0-IPv6.img
2.2.14-5.0-IPv6
```

- Sửa file /etc/lilo hay chạy lệnh `liloconf` để thêm dòng vào kernel mới /boot/vmlinuz-ip6, đồng thời, không được quên thêm hay sửa dòng vào trong file /etc/lilo.msg. ***Chú ý: Phải rất cẩn thận khi xóa kernel cũ hoặc chạy kernel mới. Đôi khi kernel IPv6 được dựng không tốt làm treo máy, dẫn đến các hỏng hóc nghiêm trọng, nếu ta không có kernel dự trữ thì không thể khởi động lại máy để sửa chữa.***
- Tạo mới hoặc ghi đè liên kết hình thức tới kernel mới:

```
ln -sf /boot/vmlinuz-version /boot/vmlinuz-ip6
```

- Chạy lệnh `lilo` để cập nhật các thay đổi của file `lilo.conf` vào bản ghi khởi động của đĩa cứng.

- Nếu tất cả các thao tác trước đó đã được thực hiện chính xác, ta khởi động lại máy bằng lệnh **reboot** và chọn kernel mới cho IPv6. Máy phải hoạt động bình thường.

■ **Cài đặt các công cụ quan trọng.** Việc cài đặt được thực hiện tương tự như các bước đã được trình bày với gói `modutils` hay với kernel mới. Ta có thể chia công cụ thành các nhóm như sau:

- *Các công cụ cấu hình IPv6*, nằm trong gói phần mềm `net-tools`, gồm các lệnh **hostname**, **netstat**, **arp**, **ifconfig**, **rarp**, **route**. Phiên bản mới nhất của `net-tools` bao giờ cũng có thể tải về miễn phí từ site <http://www.tazenda.demon.co.uk/phil/net-tools/>. Chú ý rằng khi lựa chọn cấu hình để biên dịch, ta đặt giá trị `yes` cho các tùy chọn GNU `gettext`, `Device hardware types` (`SIT` (`IPv6 -inIPv4`) `support`) và tất cả các tùy chọn về họ giao thức IPv6.
- *Các ứng dụng chung.* Trước hết, đó là siêu daemon **xinetd** (eXtended InterNET super Daemon), được lấy từ site <http://synack.net/xinetd/>. Khi cấu hình, ngoài các tùy chọn cấu hình khác thích hợp với từng hệ thống cụ thể, nhất thiết bạn phải có tùy chọn cho IPv6: **configure --with-inet6**. Trước khi cài đặt phải gỡ bỏ **xinetd** cũ. Sau đó phải sửa nội dung file `inetd.conf` cho phù hợp, chạy thử daemon để kiểm tra. Sau đó là bộ công cụ `netkit`, bao gồm các lệnh và daemon **finger[d]**, **inetd**, **ping/ping6**, **telnet[d]** và cho riêng IPv4 **ruptime**, **rwho[d]** (có thể tải từ site <ftp://ftp.suse.com/pub/people/kukuk/ipv6/> cho Linux). Ngoài ra, còn cần cài đặt hai tiện ích `ftp` cho server: daemon **ftpd-BSD**, và cho client: **lukemftp**, mã nguồn lấy từ các site tương ứng [http://www.eleves.ens.fr:8080/home/madore/programs/#prog\\_ftpd-BSD](http://www.eleves.ens.fr:8080/home/madore/programs/#prog_ftpd-BSD), và <ftp://ftp.netbsd.org/pub/NetBSD/misc/lukemftp/>.
- Các công cụ gỡ rối IPv6 bao gồm các tiện ích IP nhiều khi được cung cấp bởi các hãng phát hành Linux (**ping6**, **tracpath6**, **traceroute6**, ...), và **tcpdump** (cùng với `libpcap.a`), lấy từ site <http://www.tcpdump.org/>.

■ **Cài đặt ứng dụng mạng khác** có dùng IPv6 như Apache, **thttpd**, MoZila, **ssh** (Secure Shell), v.v..

### ▪ *IPv6 trong Solaris*

IPv6 chỉ có thể hoạt động trong Solaris 7 (x86), 2.7 (SPARC) trở đi, bản được cập nhật đầu năm 2001, là IPv6 Prototype Version 1.1 for Solaris 7. Yêu cầu về dung lượng đĩa trống dành cho cài đặt IPv6 có khác nhau đối với các bản cho SPARC và x86. Đối với SPARC, ta cần ít nhất 23 MB, nếu đĩa cứng không phân partition, hoặc 12 MB nếu /usr nằm ở partition riêng, 11 MB trong /. Đối với x86, các dung lượng cần thiết tương ứng là 11 MB, hoặc 8 và 3 MB. Ngoài ra, dung lượng đĩa cứng trống dành cho việc bung ra ban đầu gói phần mềm là 25 MB đối với SPARC, 11 MB cho x86. Gói phần mềm đó là patch số 107788-01 cho SPARC, 107916-01 cho x86. Bản Solaris phải chưa được cài đặt bất cứ patch nào khác trước khi chuyển sang IPv6.

Việc cài đặt được tiến hành theo cách thông thường, dùng lệnh **patchadd**. Để gỡ bỏ IPv6, trở lại dùng IPv4, ta loại patch tương ứng ra khỏi hệ thống bằng lệnh **patchrm**. Kết quả thử nghiệm và so sánh với các RFC về IPv6 đã được nhóm Kỹ thuật Internet của hãng Sun Microsystems công bố.

### ▪ *Các ứng dụng hiện nay với IPv6*

#### Mạng 6Bone

Như đã thấy, mặc dù việc nghiên cứu, đề xuất IPv6 đã được tiến hành từ khoảng hai chục năm trước, dù rằng nhu cầu đối với địa chỉ Internet ngày càng cấp thiết cùng với sự nhảy vọt về số lượng các dịch vụ Internet hiện đại, việc cải tạo lại hệ thống các mạng máy tính dùng IPv4 không hề đơn giản chút nào. Ngay như ở Việt Nam, Internet mới được khai thác từ năm 1997, nhưng sau khoảng 4-5 năm, số các tên vùng được đăng ký đã lên tới hơn 1400. Chính vì vậy, người ta lập ra một mạng trực quốc tế mới, dùng IPv6, và đặt tên là 6Bone. Cho tới đầu năm 2001, mạng 6Bone đã hoạt động trên 50 quốc gia và được các quốc gia bước vào Internet “chậm chân” hơn nhưng hiện nay phát triển công nghệ tin học rất mạnh như Ấn Độ, Trung Quốc, v.v.. hết sức nhiệt tình hưởng ứng. Kiến trúc của mạng 6Bone được thiết kế như sau. Ở lớp trên cùng là một số chưa nhiều (~ 500 vào đầu năm 2000) các máy chủ rất mạnh tham gia vào các site, được nối với nhau và được coi như những “nút” của mạng trực, với nhiệm vụ duy trì hoạt động cho toàn mạng. Một trong những

chức năng chính yếu của chúng là gán địa chỉ IPv6 cho các máy khác được nối vào, nhưng không nằm trên các nút chính của 6Bone, để thực hiện việc phân cấp địa chỉ. Đến lượt mình, các site gán các nút trực nhất được xem như thuộc lớp trung gian, có nhiệm vụ chuyển tiếp các dịch vụ từ các nút mạng trực tới các nút phía dưới nữa, theo thuật ngữ thường được gọi là các *nút lá*. Mạng 6Bone vào thời điểm hiện nay vẫn trong giai đoạn hình thành kiến trúc cơ bản, nhưng trong thời gian rất ngắn nữa – chỉ khoảng vài ba năm - chắc chắn sẽ là một trong những giải pháp quan trọng nhất để thay thế các mạng IPv4 làm việc cần mẫn, nhưng đang bị lỗi thời. Tuy vậy, mạng IPv4 sẽ không chấm dứt sự có mặt của mình và còn phục vụ chúng ta một thời gian khá dài nữa. Về phần mình, với việc tin học mới được phổ biến, phát triển trong tất cả các ngành kinh tế, khoa học kỹ thuật chính, Việt Nam hoàn toàn có thể hy vọng nhanh chóng áp dụng chuẩn IPv6, xây dựng một hạ tầng công nghệ thông tin mới với chi phí không cao và đáp ứng được các đòi hỏi của các dịch vụ hiện đại.

### *Các dịch vụ trên nền IPv6*

Trước hết, đó là *các chương trình, tiện ích “truyền thống” của UNIX*, nếu nhìn từ phía người quản trị mạng, như DNS, Apache Server Web và Proxy, **tthttpd** cho server của Web site, các trình duyệt Mozilla, ...; chương trình tự động cấu hình mạng **freenet6**, daemon có tên Router ADvertisement **radvd**; shell bảo mật **ssh**; các phần mềm đo lường truyền thông qua của mạng **ttcp**, **ptcp**; các chương trình thư điện tử **sendmail**, POP, IMAP, SMTP, v.v... Trong số trên có những chương trình chạy được trong môi trường hỗn hợp, nhưng số còn lại chỉ hoạt động với IPv6 và thay thế một số lệnh truyền thống với IPv4 (**ping**, **finger**, **ftp**, ...). Nói chung, các chương trình trên đều tận dụng các ưu điểm của IPv6 mà ta sẽ điểm qua hai trong số đó dưới đây.

Như đã biết, thông tin được chuyển trong mạng dưới dạng các gói dữ liệu. Khi qua mỗi một lớp trong mô hình mạng OSI/ISO, file dữ liệu lần lượt được gán thêm một số thông tin và tín hiệu điều khiển, được gọi là *header* – phần đầu. Từ phía máy gửi dữ liệu đi, ta sẽ có theo chiều từ lớp Ứng dụng trở xuống các header AH, PH, SH, TH, NH, và DH. Tại lớp Liên kết dữ liệu còn có thêm phần đuôi (*trailer*) – DT để soát lỗi. Tại lớp Vật lý dữ liệu được chuyển đi dưới dạng một chuỗi các bit trong



khung dữ liệu (*frame*) hay *datagram*. Tuy tại lớp Mạng mà ta quan tâm dữ liệu được chia thành các gói để chuyển đi, trong thực tế người ta vẫn dùng như nhau các tên gói hoặc *datagram*. Một trong những nhược điểm của IPv4 là header có cấu trúc rất phức tạp, gồm tất cả 10 trường, trong số đó có hai trường đầu địa chỉ 32 bit cho máy nguồn và máy đích, và trường các tùy chọn. Thậm chí nếu trường các tùy chọn để trống, header theo IPv4 của gói đã có độ dài 20 byte. Nếu tính rằng địa chỉ IPv6 dài gấp bốn lần, một phần đầu theo IPv6 ngắn nhất cũng phải dài 80 byte thì rõ ràng là quá cồng kềnh cho việc vận chuyển trong mạng. IPv6 header được đơn giản hóa đi bằng cách tạo chuỗi: nó chỉ còn chứa 6 trường để chỉ hai địa chỉ 128 bit (không có tùy chọn), xác định thông tin tối thiểu cần thiết cho gói IPv6: *version*, *priority*, *flow label*, *payload length* và *hop limit* và một trường mới được dùng để chỉ tới tiếp một header nữa, nằm ngay sau header đang xét, nhưng ở trước phần chứa dữ liệu, và có thể dùng chứa các thông tin điều khiển giống như trường tùy chọn trong header của IPv4.

Số lượng các header nằm phía sau trong chuỗi không bị giới hạn. Do trường để chỉ tới header phía sau là một số 8 bit, nên ta có tất cả 255 loại header khác nhau. Đến thời điểm hiện nay, người ta mới định nghĩa được cho sáu loại header sau:

- Hop-by-hop options (các tùy chọn về hop). *Hop* được hiểu nôm na như điểm dừng khi gói IP “nhảy” từ điểm dừng này tới điểm dừng khác trên đường lan truyền trên mạng.
- Routing header (chứa thông tin phân đường).
- Fragment header (chứa thông tin về phân mảnh dữ liệu).
- Authentication header (chứa thông tin về kiểm định).
- Payload header (chứa thông tin về khả năng tái).
- Destination options header (chứa thông tin tùy chọn về đích tới của gói IP).

Kết quả là header IPv6 chỉ còn dài 40 bytes – gấp hai lần header IPv4 không có tùy chọn – mặc dù bản thân hai địa chỉ tham gia vào đó đã lớn hơn bốn lần kích thước của header IPv4. Không những vậy, do cấu trúc gọn hơn, nhưng chứa đựng nhiều khả năng tiềm tàng hơn, header uyển chuyển hơn cho việc vận chuyển trong mạng.

Ưu điểm thứ hai liên quan tới lĩnh vực bảo mật trên mạng. Trong thiết kế IPv4 người ta đã không chú ý tới khả năng bảo mật. Việc lấy cắp các thông tin cá nhân, mật khẩu và truy cập trái phép, các hành động phá hoại các site và dữ liệu là các hiện tượng thường thấy trong các mạng IPv4 lỗi, không tăng cường bảo mật. IPv6 ngay từ đầu đã chú ý kết hợp bảo mật từ mức cơ bản nhất. Giao thức bảo mật có tên IP-Sec và các cải tiến của nó sử dụng các header tùy chọn để cung cấp thao tác kiểm định tư cách và một khả năng nữa có tên gọi là Encapsulating Security Payload (ESP – tạm dịch là Tải trọng bảo mật đóng mũ). Hai phần này trong các đặc điểm bảo mật có thể được nâng cấp một cách độc lập. Hơn thế nữa, header cho ESP còn có một trường xác định mức bảo mật và cách thức nâng cấp bảo mật. Hiện nay, luật pháp của Mỹ cấm xuất ra nước ngoài các kỹ thuật và thuật toán mã hóa lớn hơn 40 bit cho các mục đích thông dụng. Vì vậy, nhờ có khả năng này, ta vẫn có thể áp dụng các biện pháp bảo mật tương đối an toàn.

Các ứng dụng liên quan tới **thương mại điện tử**. Trong thời gian gần đây xu hướng toàn cầu hóa đang tăng lên mạnh mẽ. Cùng với việc mạng Internet đang phát triển ở hầu hết các quốc gia, các hoạt động quảng cáo, mua bán, giao dịch trực tuyến trên mạng được triển khai rộng rãi. IPv6 cung cấp không gian địa chỉ rộng rãi và an toàn cho đồng đảo các thành phần tham gia vào quá trình thương mại điện tử trên. Trong những năm sắp tới đây, ta có thể chờ đón cơ hội sử dụng các dịch vụ thương mại điện tử được cung cấp phổ biến tại Việt Nam với tính bảo mật cao trên nền IPv6.

Giao thức mới và địa chỉ IPv6 có một thị trường ứng dụng cực kỳ lớn mà thời điểm hiện nay đánh dấu sự bắt đầu giai đoạn phát triển bùng nổ của nó. Đó là thị trường các sản phẩm với **công nghệ không dây**, đầu tiên là các máy tính có tính di động, tiếp theo là điện thoại di động có kết nối Internet, bước sau nữa có thể là các sản phẩm tích hợp có kích thước nhỏ gọn, ... Do không còn phải gấn chúng tại vị trí cố định, có thể dự đoán rằng số lượng các sản phẩm loại này sẽ tăng lên rất nhanh trong thời gian tới, dẫn tới một số lượng địa chỉ khổng lồ mà chỉ IPv6 mới có thể đáp ứng nổi. Giao thức Internet mới không những phải đảm bảo gọn, tiện lợi, mà phải linh hoạt, tự động cấu hình được. Ta hãy tưởng tượng một người dùng, tại văn phòng anh ta tham gia vào mạng của cơ quan mình qua dây nối (cáp mạng, modem, v.v...),

nhưng trên đường đi, anh ta có thể dùng các phương cách khác để trao đổi thông tin với bất cứ mạng máy tính khả dĩ nào.

Công nghệ giải trí sẽ là một địa chỉ ứng dụng quan trọng vào bậc nhất của IPv6 trong tương lai gần. Hiện nay, các loại hình giải trí đang được thiết kế để gắn kết với nhau một cách thống nhất qua mạng. Ví dụ, ta sẽ dùng máy tính ở nhà với màn hình tinh thể lỏng kích thước lớn để thay thế tivi truyền thống và có thể xem bất cứ chương trình nào được phát trên thế giới và được truyền đến máy tính của ta. Hiện nay, trên thế giới đã có hàng trăm kênh truyền hình và video như vậy. Nếu xét về phương diện kỹ thuật, các dữ liệu đa phương tiện được dùng là ở dạng video, audio, ... có kích thước rất lớn (một ví dụ đơn giản, chỉ một file audio có đuôi .*dat* cho một bài hát nhạc nhẹ đã chiếm dung lượng khoảng 30 – 50 MB, file nén dạng .*mp3* chỉ giảm kích thước nhiều nhất được khoảng 10 lần. Các file video có đuôi .*dat* với thời gian xem 60 phút chiếm dung lượng khoảng 20 GB). Vì vậy, việc nghẽn mạch hoàn toàn có thể xảy ra. Các giao thức IPv4 đã có khả năng xử lý các nghẽn mạch tương tự, nhưng IPv6 đưa vào cả các cải tiến cũng như các yếu tố mới.

Thêm nữa, một vấn đề tồn tại của kỹ thuật phát thanh truyền hình qua mạng Internet là phải giải quyết việc các kênh có dung lượng khác nhau nối các máy thu khác nhau tới cùng một dữ liệu như thế nào cho hợp lý. Rõ ràng giải pháp sao chép lại buổi phát để sau này lựa cho phù hợp với khả năng dung lượng của máy thu chậm nhất hoàn toàn không phải là tối ưu. Ta hãy tưởng tượng tivi nối mạng của ta tự nhiên bị mất tín hiệu màu chỉ vì ở đâu đó dây phố bên, hay thành phố khác, ai đó cũng xem cùng chương trình nhưng lại bật máy thu đen trắng. Nguyên tắc chung là IPv6 chia riêng các dải băng thông của mạng cho dữ liệu đa phương tiện là dữ liệu thời gian thực, nghĩa là dữ liệu không thể điều khiển chống nghẽn mạch, và cho các dữ liệu chương trình máy tính “bình thường”, nghĩa là dữ liệu có thể điều khiển chống nghẽn mạch như của **mail**, **ftp**, NFS và gán giá trị cho trường có tên *Drop Priority* trong gói dữ liệu. Trong nhóm dữ liệu thứ hai, điều khiển chống nghẽn mạch có mức độ ưu tiên cao hơn mức độ ưu tiên tương tác của phiên làm việc **telnet**; về phần mình, **telnet** lại được ưu tiên hơn **mail**. Đối với nhóm dữ liệu thời gian thực, người thiết kế ứng dụng phải quy định mức độ ưu tiên, nhưng thông thường, các ứng dụng phát thanh truyền hình như dữ liệu video, audio băng thông hẹp có mức độ ưu tiên cao hơn so với các gói có chứa các thành phần độ phân giải cao của

dữ liệu audio, video. Như vậy, nếu mạch có bị nghẽn thì các tín hiệu độ phân giải thấp vẫn có thể đi qua được, còn thành phần với độ phân giải cao bị router bỏ lại. Người dùng thử xem chương trình chất lượng không cao còn hơn là không nhận được gì!

Các thiết bị điều khiển, tự động hóa chính là đối tượng ứng dụng của IPv6 mang lại hiệu quả nhiều nhất. Ta thấy bất cứ trong lĩnh vực sản xuất, đời sống nào cũng hiện diện các thiết bị, công cụ mà nếu biến chúng thành thông minh, thì chúng có thể chấp hành lệnh từ máy tính điều khiển và có khả năng tự thích nghi. Thực ra, hiện nay cũng có nhiều công cụ được điều khiển, như động cơ bước, thiết bị tạo vi khí hậu, chiếu sáng, máy móc thí nghiệm, v.v.. Tình trạng chung hiện nay là nếu điều khiển được thực hiện bằng kỹ thuật tương tự thì vừa cồng kềnh, tiêu hao nhiều năng lượng, vừa có độ chính xác và độ tin cậy không cao, nếu bằng kỹ thuật số thì chỉ có tính cục bộ, đơn lẻ. Với số lượng khổng lồ các địa chỉ IPv6 tiềm tàng, ta có thể dự tính tới các thiết kế hạ tầng kỹ thuật có điều khiển cho đủ mọi loại đối tượng, từ quy mô toàn xã hội tới từng gia đình, với yêu cầu phải có các giải pháp đơn giản, tiện lợi và giá thành rẻ.

## XV. 2. uucp

**uucp** (UNIX to UNIX Copy Protocol) ngay từ tên gọi đã cho thấy đó là giao thức cơ bản trong truyền thông giữa các hệ thống UNIX với nhau. Tuy hiện nay rất nhiều các giao tiếp mạng đã được chuyển sang cơ sở TCP/IP, **uucp** vẫn được ứng dụng hết sức phổ biến. Là người quản trị mạng UNIX, bạn cần phải biết rõ các công cụ quản lý, các cơ sở dữ liệu và các phương tiện ghi lại hoạt động (log file) được dùng trong **uucp**.

### XV. 2. a. Phần cứng của uucp

- Đường nối trực tiếp:

Có thể dùng cáp nối trực tiếp giữa các cổng nối tiếp của hai máy tính, nếu chúng thường xuyên liên hệ với nhau và được bố trí không cách xa nhau (dưới 15 m). Tốc độ truyền dữ liệu phụ thuộc vào các cổng, nhưng giá trị tới 19200 hoặc 38400 bps không phải là khó đạt được trong các hệ thống hiện nay.

- Đường điện thoại:

Máy tính của bạn có thể dùng modem qua đường điện thoại bình thường hay chuyên dụng để kết nối với các máy khác. Tốc độ phụ thuộc không những vào cổng nối tiếp của máy, mà còn vào tính năng của modem và nhất là vào chất lượng đường truyền.

– Mạng:

Nếu máy tính của bạn hiện diện như một nút trong mạng máy tính, nó có thể giao tiếp với các nút khác trong mạng.

### ***XV. 2. b. Các chương trình trong uucp***

**uucp** chia các chương trình thành hai nhóm cho người dùng và cho quản trị.

#### ***▪ Các chương trình cho người dùng***

Các chương trình này nằm trong `/usr/bin` và không bị hạn chế chủ quyền chạy.

#### ***❖ cu***

Nối máy tính của bạn với máy từ xa, bạn có thể **login** vào cả hai máy cùng một lúc, chạy hoặc sao chép các chương trình. Cú pháp của lệnh:

```
cu [-c device] [-l line] [-s speed] [-b bits] [-h] [-n] [-t] [-d] [-o] [-e] [-L] [-C] [-H] telno|systemname [local-cmd]
```

Các tùy chọn được thường xuyên sử dụng:

**c** *device*

Chỉ dùng giá trị trường Type trong `/etc/uucp/Devices`.

**l** *line*

Dùng giá trị trường Line trong `/etc/uucp/Devices`.

**s** *speed*

300, 1200, 2400, 9600, 19200, 38400 bit/s (hay bps). Mặc định: Any. Đối với các bản UNIX mới nhất, *speed* có thể có giá trị cao hơn.

**b** *bits*

7 hoặc 8 bit.

**h**

Đổi chế độ half-duplex (đơn công-song công).

**n**

Yêu cầu người dùng điền số điện thoại cần kết nối.

**d**

In chẩn đoán.

**L**

Buộc **cu** chuyển sang thủ tục đăng nhập theo quy định trong file */etc/uucp/Systems*.

**C**

Để **cu** thực hiện lệnh *local -cmd* trên máy tại chỗ.

### ❖ uucp

Cho phép người dùng sao chép file từ máy này sang máy khác. Cú pháp của lệnh được quy định như sau:

```
uucp [-c|-C] [-d|-f] [-g grad] [-j] [-m] [-n user] [-r] [-s
 file] [-x debug_level] source-file destination-file
```

Các tùy chọn hay được sử dụng lần lượt có ý nghĩa là:

**c**

Không chép các file tại chỗ vào thư mục hàng chờ để truyền đi (mặc định).

**C**

Ngược lại với **c**.

**d**

Tạo mọi thư mục để chép file (mặc định).

**m**

Gửi mail cho người yêu cầu khi chép xong file.

**n** *user*

Thông báo cho *user* ở máy từ xa là file đã được gửi.

**r**

Chưa chuyển file, chỉ xếp công việc vào hàng chờ.

**h**

0 - 9. Thông báo gõ lỗi trên màn hình, số càng cao, càng chi tiết.

❖ uuto

Sao chép file từ một máy sang thư mục hàng chờ chung của máy kia. Cú pháp:

```
uuto [options] source-file destination
```

❖ uupick

Lấy file đã nhận bằng **uuto** từ thư mục hàng chờ chung. Cú pháp:

```
uupick [-s system]
```

❖ uux

Tạo các file làm việc, dữ liệu và thực thi cần thiết để chạy lệnh trên máy từ xa. Cú pháp:

```
uux [-] [-a name] [-b] [-c] [-C] [-g grad] [-j] [-n] [-p] [-r]
 [-s file] [-x debug_level] [-z] command-string
```

Các tùy chọn:

-

Vào chuẩn của **uux** là vào chuẩn của *command-string*.

**c**

Không chép các file tại chỗ vào thư mục hàng chờ để truyền đi (mặc định)

**C**

Ngược lại với **c**.

**d**

Tạo mọi thư mục để chép file (mặc định).

**n**

Không thông báo cho người dùng khi lệnh không thực hiện được.

**r**

Chưa chuyển file, chỉ xếp công việc vào hàng chờ.

**z**

Thông báo thành công cho user ở máy từ xa.

❖ uustat

Hiển thị thông tin về các cuộc truyền file được yêu cầu.

- *Các chương trình quản trị*

❖ uulog

Hiển thị nội dung các file log ghi lại mỗi lần dùng **uucp**, **uuto** và **uux**. Cú pháp:

```
uulog [-s sys] [-f system] [-x] [-number] system
```

❖ uucleanup

Xóa thư mục hàng chờ. Thường cũng được gọi bằng script `uudemon.cleanup` bởi **cron**.

❖ Uutry

Kiểm tra khả năng xử lý cuộc gọi. Kích hoạt daemon **uucico** để kết nối với máy từ xa được chỉ định. Cú pháp:

```
/usr/lib/uucp/Uutry [-r] [-c type] [-x debug-level]
system-name
```

❖ uucheck

Kiểm tra xem các thư mục **uucp**, chương trình và một vài phần trong file Permissions để tìm lỗi cú pháp. Cú pháp:

```
/usr/lib/uucp/uucheck [-v] [-x debug-level]
```

- *Daemon*

❖ uucico

Chọn các thiết bị dùng để kết nối, tạo kết nối, thực hiện thủ tục đăng nhập và kiểm tra chủ quyền, truyền dữ liệu và chạy các file, ghi lại log.



❖ uuxqt

Dùng để chạy các lệnh theo yêu cầu từ xa.

❖ uusched

Đặt lịch cho hàng chờ. Thường được gọi chạy bởi `/etc/rc` khi khởi động máy, và sau đó bởi `uudemon`. hour nhờ **cron**.

**XV. 2. c. Khởi động uucp**

Để khởi động **uucp** bằng tay, phải đăng nhập vào với quyền `root`, sau đó gõ lệnh:

```
su uucp
crontab < /usr/lib/uucp/uudemon.crontab
```

**XV. 2. d. Các file dữ liệu của uucp**

Các file này nằm trong thư mục `/etc/uucp`.

❖ Devices

Chứa thông tin về vị trí bộ gọi tự động (Automatic Calling Unit - ACU), các thiết bị mạng. Các dòng vào có dạng:

| Type       | Line | Line2 | Class     | Dialer-Token-Pairs |
|------------|------|-------|-----------|--------------------|
| Ví dụ: ACU | cua0 | -     | Any hayes | \D                 |

với các trường có ý nghĩa sau: **Type** : loại **Direct** (kết nối trực tiếp) hay **ACU**; **Line** và **Line2** : tên cổng và cổng thứ hai, ở đây chỉ dùng cổng nối tiếp thứ nhất là `/etc/cua0`; **Class** : tốc độ đường truyền (phân biệt cho từng đường).

❖ Dialers

Quy định đối thoại ban đầu vào thời điểm thiết lập kênh truyền thông, trước khi có thể truyền dữ liệu. Dạng dòng vào như sau:

| Dialers | Substitutions | Expect - Send |
|---------|---------------|---------------|
|---------|---------------|---------------|

Ví dụ:

```
hayes =, -, " " \dA\pTE1V1X1Q0S0=1S2=255S12=255&W&YZ\r\c
 OK\r \EATDT\T\r\c CONNECT STTY=crtscts
```

Các ký hiệu được dùng để quy định chế độ kết nối có các ý nghĩa như sau:

- \p : dừng 1/4 giây
- \d : trễ 2 giây
- \D : số điện thoại không bị biến đổi bởi Dialcodes.
- \T : số điện thoại bị biến đổi bởi Dialcodes.
- \K : thêm tín hiệu BREAK
- \E : hiển thị các tín hiệu kiểm tra.
- \e : cấm các tín hiệu kiểm tra.
- \r : đầu dòng
- \c : không xuống dòng
- \n : gửi dòng mới (xuống dòng)

Ở ví dụ trên ATE1V1X1Q0S0=1S2=255S12=255&W&YZ là câu lệnh chuẩn đặt cấu hình cho modem loại Hayes, có nghĩa là quay số có hiển thị các tín hiệu kiểm tra và tín hiệu trả lời trên màn hình, dùng mã kết nối thích hợp mà không kiểm tra tín hiệu “dial tone”, đặt mã cho các thanh ghi rung chuông trả lời tự động, dãy ký tự thoát và thời gian dừng dãy ký tự thoát, có ghi và lưu lại câu lệnh.

Bạn đọc có thể tham khảo thêm về tập lệnh AT dùng cho modem trong bất cứ tài liệu kỹ thuật kèm theo nào của modem.

### ❖ Systems

Chứa các thông tin để **uucico** thiết lập các kết nối với máy từ xa. Dạng dòng vào như sau:

| System-Name | Time | Type | Class | Phone | Login |
|-------------|------|------|-------|-------|-------|
|-------------|------|------|-------|-------|-------|

Ví dụ:

```
ct100ppp Any ACUH 38400 8433295 "" P_ZERO
 "" \r hanoi2> ppp name: 202.3.1.5
```

Trường đầu tiên System-name chứa tên node của máy từ xa, có thể là một router hay một host đóng vai trò router, Time có dạng daytime [:retry] (Any: trao đổi hai chiều bất cứ thời điểm nào; Never: chỉ được nhận tín hiệu gọi đến; và ngày giờ, ví dụ: Wk 800-1500 - tất cả các ngày làm việc trong tuần, từ 8<sup>00</sup> đến 15<sup>00</sup>), Type: giống như trong file Devices, ở trong ví dụ này có nghĩa là bộ gọi tự động theo chuẩn Hayes, Class: tốc độ đường truyền, Phone: số điện thoại, Login: xác định tên, mật khẩu của người dùng kết nối vào máy từ xa.

#### ❖ Dialcodes

Chứa ký hiệu viết tắt của số điện thoại trong trường Phone của file Systems. Dòng vào có dạng:

Abb dial-seq

Ví dụ: HN 835 → số điện thoại HN2594 nếu trong trường Dialer-Token-Pairs của file Devices có ký hiệu \T.

Việc xác định nội dung bốn file trên rất cần cho việc thiết lập cấu hình truyền thông **uucp**. Trong nội dung các file đó, nhà cung cấp thường cho ta sẵn các chỉ dẫn và các câu lệnh mặc định đối với các tình huống và loại modem cụ thể.

#### ❖ Permissions

Chứa quyền đăng nhập quy định cho các máy từ xa. Dạng dòng vào:

NAME=VALUE

trong đó: LOGNAME: tên đăng nhập cho các máy từ xa truy nhập vào hệ thống; MACHINE: tên máy từ xa để máy tại chỗ truy nhập vào máy từ xa; REQUEST=YES/NO; SENDFILE=YES/NO; [NO]READ; [NO]WRITE ; COMMANDS: quy định các lệnh và cách thực hiện chúng trên máy tại chỗ bởi các máy từ xa., v.v..

❖ Poll

Xác định số trượt sẽ được quay đến, nếu số điện thoại chính đang bị bận.

❖ Sysfiles

Xác định những file khác dùng thay cho Systems, Devices, Dialers đối với **uucp** và **cu**.

**XV. 2. e. Dùng TCP/IP trong uucp**

Ta phải có những thay đổi sau:

- Thêm hoặc thay đổi các trường trong file Systems để dòng vào có dạng như sau:

| System-Name | Time | TCP | Port | Networkname | Login |
|-------------|------|-----|------|-------------|-------|
|-------------|------|-----|------|-------------|-------|

- Thêm dòng vào sau trong file Devices:

TCP --Any TCP -

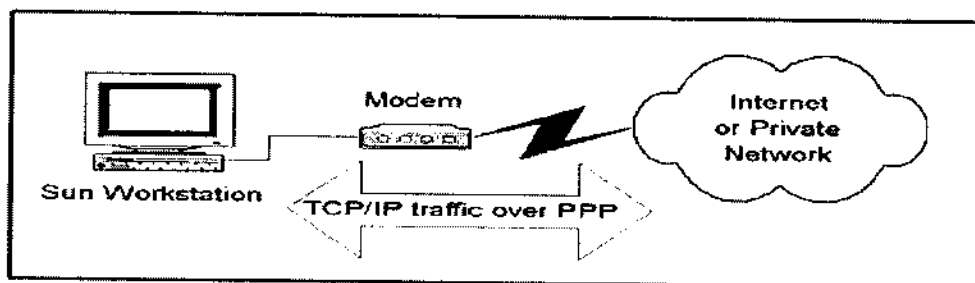
Nếu trong trường hợp TCP/IP vẫn không hoạt động tốt, ta phải sửa bằng tay thêm như sau:

- Thêm tên host vào file `/etc/nodename`.
- Thêm địa chỉ IP hoặc tên máy cho giao diện mạng (ví dụ `elx0` cho 3C509 thứ nhất trong máy) vào file `/etc/hostname.elx0`.
- Thêm địa chỉ IP hoặc tên máy cho gateway mặc định trong file `/etc/defaultrouter`.
- Đầu tiên sửa file `/etc/inet/hosts` để root có quyền ghi (`chmod u+w /etc/inet/hosts`), sau đó ít nhất phải thêm tên của máy tại chỗ và của gateway.
- Thêm tên (các) mạng tại chỗ và mặt nạ mạng của chúng vào file `/etc/inet/netmasks`.

**XV. 3. Dịch vụ ppp**

Kể từ đầu những năm 1970, khi Internet chính thức ra đời, cho đến nay rất nhiều tiến bộ kỹ thuật đã được áp dụng. Các phần mềm tìm kiếm mạnh đã thay cho lệnh

**archie**, lệnh **gopher** được thay bằng các trình duyệt Web, nhưng **ppp** (viết tắt của Point-to-Point protocol), một chuẩn công nghiệp để gửi các gói thông tin mạng qua nối kết điểm-tới-điểm, không có nhiều thay đổi căn bản. Trong bản cài đặt Solaris có **ppp** đi kèm, nhưng Sun còn đưa ra cải tiến lấy tên là Sunlink PPP. H.XV.1 cho ta thấy vai trò của **ppp** trong việc trao đổi thông tin qua mạng.



H. XV.1. Thiết lập phần cứng để dùng **ppp** .

Các đặc điểm của **ppp** bao gồm:

- Hỗ trợ các chuẩn Internet 1144, 1331, 1332, 1334, 1661, 1662.
- Cấu hình modem và các cổng dễ dàng.
- Các bản ghi các cuộc trao đổi qua mạng có thể được lưu và đọc lại.
- Khả năng gán địa chỉ động, đối với Solaris từ phiên bản 2.5 trở đi .
- Kiểm tra xác nhận người dùng.
- Hỗ trợ các đường truyền nối tiếp đồng bộ.
- Thiết lập kết nối bằng tay hoặc tự động.
- Có các tùy chọn lần theo vết và gỡ rối mạnh.

### **XV. 3. a. Các gói phần mềm cài đặt**

Nếu đã cấu hình hệ thống để quay số dùng **ppp** kết nối ra ngoài tới nhà cung cấp dịch vụ Internet nào đó, ta có thể bỏ qua bước kiểm tra này. Trong trường hợp ngược lại, ta dùng lệnh để kiểm tra xem đã có gói phần mềm nào của **ppp** đã được cài đặt:

```
pkginfo | grep ppp
```

Các gói phần mềm cần cho **ppp** là:

|           |                                                                   |
|-----------|-------------------------------------------------------------------|
| SUNWapppr | Các file cấu hình daemon PPP của PPP/IP không đồng bộ.            |
| SUNWapppu | Các dịch vụ truy nhập PPP và daemon PPP của PPP/IP không đồng bộ. |
| SUNWpppk  | Các trình điều khiển thiết bị quay số IP và PPP/IP.               |

Nếu có các nối kết quay số ra ngoài, còn cần có các gói sau:

|          |                                |
|----------|--------------------------------|
| SUNWbnur | Các tiện ích mạng UUCP (Root). |
| SUNWbnuu | Các tiện ích mạng UUCP (User). |

Nếu bạn đã có các bản Solaris 7 hoặc 8 được cài trên máy, bạn cần phải có thêm gói phần mềm sau:

|           |                                                              |
|-----------|--------------------------------------------------------------|
| SUNWpppkx | Các trình điều khiển thiết bị quay số IP và PPP/IP (64 bit). |
|-----------|--------------------------------------------------------------|

Nếu như các gói phần mềm này chưa được cài đặt hay bị hỏng, ta phải dùng lệnh **pkgadd** để cài đặt lại chúng theo trình tự sau:

### **Đối với Intel (x86) Solaris 8:**

Đưa đĩa CD có tên "Solaris 8 Intel Platform Edition Software CD 2 of 2" vào ổ CD và gõ (dấu "\ " ký hiệu viết tiếp dòng lệnh):

```
pkgadd -d /cdrom/sol_8_ia_2/Solaris_8/Product \
 SUNWpppk
pkgadd -d /cdrom/sol_8_ia_2/Solaris_8/Product \
 SUNWapppu
pkgadd -d /cdrom/sol_8_ia_2/Solaris_8/Product \
 SUNWapppr
```

Và thêm gói sau cho bản Solaris 8 64-bit:

```
pkgadd -d /cdrom/sol_8_ia_2/Solaris_8/Product \
SUNWpppkx
```

### Đối với SPARC Solaris 8:

Đưa đĩa CD có tên "Solaris 8 SPARC Platform Edition Software CD 2 of 2" rồi gõ các dòng lệnh:

```
pkgadd -d /cdrom/sol_8_sparc_2/Solaris_8/Product \
SUNWpppk
pkgadd -d /cdrom/sol_8_sparc_2/Solaris_8/Product \
SUNWapppu
pkgadd -d /cdrom/sol_8_sparc_2/Solaris_8/Product \
SUNWapppr
```

Và cũng thêm gói cho bản 64-bit của Solaris 8:

```
pkgadd -d /cdrom/sol_8_sparc_2/Solaris_8/Product \
SUNWpppkx
```

### Đối với Intel (x86) Solaris 7:

Đưa đĩa CD Intel Solaris 7 vào ổ đĩa CD và gõ:

```
pkgadd -d /cdrom/sol_7_x86/s2/Solaris_2.7/Product \
SUNWpppk
pkgadd -d /cdrom/sol_7_x86/s2/Solaris_2.7/Product \
SUNWapppu
pkgadd -d /cdrom/sol_7_x86/s2/Solaris_2.7/Product \
SUNWapppr
```

Và cho bản Solaris 7 64-bit ta phải thêm gói:

```
pkgadd -d /cdrom/sol_7_x86/s2/Solaris_2.7/Product \
SUNWpppkx
```

### Đối với SPARC Solaris 7

Đưa đĩa CD SPARC Solaris 7 vào ổ CD và gõ:

```
pkgadd -d /cdrom/sol_7_sparc/s0/Solaris_2.7/Product \
 SUNWpppk
pkgadd -d /cdrom/sol_7_sparc/s0/Solaris_2.7/Product \
 SUNWapppu
pkgadd -d /cdrom/sol_7_sparc/s0/Solaris_2.7/Product \
 SUNWapppr
```

Và thêm gói sau cho bản 64-bit Solaris 7:

```
pkgadd -d /cdrom/sol_7_sparc/s0/Solaris_2.7/Product \
 SUNWpppkx
```

Các bản Solaris từ phiên bản 7 trở về trước chỉ có một đĩa CD-ROM cài đặt các thành phần chính của hệ điều hành, nên ta có thể áp dụng cách làm với Solaris 7, chỉ cần thay số “7” bằng các số tương ứng chỉ phiên bản, ví dụ như “2.5.1”. Nếu đã cài đặt bản thương phẩm cho server Solaris 7 hoặc Solaris 8, ta không nên cài đặt bản tùy chọn Solstice PPP nữa, vì nó sẽ dừng aspppd của hệ thống.

Để kiểm tra xem các gói phần mềm có được cài đặt đúng cách không, ta dùng lệnh **pkgchk**:

```
pkgchk SUNWpppk SUNWapppu SUNWapppr
ERROR: /etc/asppp.cf
 group name <sys> expected <other> actual
 file size <360> expected <689> actual
 file cksum <27915> expected <57161> actual
#
```

Thông báo trên nói rằng file /etc/asppp.cf đã bị thay đổi từ khi được cài đặt hệ thống. Điều đó là bình thường vì file /etc/asppp.cf mặc định là file trống và luôn luôn được cập nhật mỗi khi cấu hình **ppp**.



Sau khi đã cài đặt đúng các gói phần mềm cần có, ta phải khởi động lại hệ thống với cờ `-r`:

```
reboot -- -r
```

cho Solaris x86, hoặc cho Solaris SPARC bằng:

```
ok boot -r
```

### *XV. 3. b. Các thành phần chính của **ppp***

Các file và chương trình của **ppp** bao gồm:

❖ **/usr/sbin/aspppd**

Daemon điều khiển nối kết **ppp**, được dùng để khởi động **ppp**. Nếu các gói thông tin đã xếp hàng chờ ở giao diện **ppp**, nhưng nó chưa được kết nối, daemon này sẽ cố gắng khởi tạo nối kết.

❖ **/etc/asppp.cf**

Là file cấu hình được **aspppd** đọc khi hệ thống khởi động. Nó xác định các giao diện được cấu hình và các đặc trưng xác định nối kết **ppp** (các dòng vào trong phần "path").

❖ **/etc/log/asppp.log**

Là file log của **ppp**, chứa các thông tin đi ra và đi đến của các kết nối. Khi **aspppd** được gọi với mức gỡ rối cao hơn, file log này sẽ chứa các thông tin quan trọng về các script gọi để quay số đến các hệ thống UNIX khác, cũng như các dòng đối thoại **ppp** khi nối kết được thiết lập. Nói riêng, bản thân tác giả cũng rất ưa dùng file log này để tìm nguyên nhân các sự cố về truyền thông. Nội dung của file log này và của file `/var/adm/log/asppp.log` thực chất là một.

❖ **/etc/init.d/asppp**

Đây là script khởi động khi hệ thống UNIX khởi động. Nó thực hiện tất cả các dòng vào "ifconfig" trong file cấu hình `/etc/asppp.cf`, sau đó khởi động daemon `/usr/sbin/aspppd`. Nhằm đảm bảo có thể gọi và kết thúc được dịch

vu **ppp** trong tất cả các mức chạy s (S), 1, 2, file này còn được biết đến với các tên gọi (tức là chúng được liên kết với nhau) `/etc/rc2.d/S47asppp`, `/etc/rc0.d/K47asppp` và `/etc/rc1.d/K47asppp`. File này có thể được dùng để kết thúc **ppp** với lệnh:

```
/etc/init.d/asppp stop
```

và khởi động lại **ppp** bằng lệnh:

```
/etc/init.d/asppp start
```

#### ❖ /tmp/.asppp.fifo

File này được **aspppd** tạo ra cho truyền thông với dịch vụ đăng nhập **aspppls**. Nếu không có file này, có thể bộ quản lý kết nối **aspppd** không hoạt động.

☛ Các thành phần sau, đồng thời là các thành phần của **uucp**, và chỉ được dùng trên những hệ thống *client* của **ppp**, nghĩa là các hệ thống chỉ quay số ra ngoài tới các hệ thống UNIX khác:

#### ❖ /etc/uucp/Systems

File này xác định tốc độ truyền, số điện thoại và dòng lệnh trao đổi để đăng nhập vào máy chủ từ hệ thống *client* (hệ thống tại chỗ). Khi bộ quản lý nối kết, **aspppd**, bắt đầu một nối kết, nó dùng tên `peer_system_name` được các định trong file `asppp.cf` để tìm tên trùng với tên hệ thống trong file này.

#### ❖ /etc/uucp/Devices

Tìm thông số trùng với nội dung trường thứ ba của file `Systems` và xác định dòng vào trong file quay số (*Dialers*) để sử dụng.

#### ❖ /etc/uucp/Dialers

Xác định script trao đổi để chạy modem, quay số điện thoại từ file `Systems` và kết nối với hệ thống *server*.

☛ Các thành phần sau chỉ được dùng trên những hệ thống server của **ppp**, nghĩa là các hệ thống được quay số tới từ các hệ thống UNIX khác:

❖ **/usr/sbin/aspppls**

Dịch vụ truy nhập **ppp**, được gọi tới như là shell của các kết nối **ppp** từ bên ngoài. Nó liên hệ với **aspppd** qua `/tmp/.asppp.fifo` để khởi tạo các kết nối **ppp** đến. (Người dùng **ppp** trong hệ thống phải có `/usr/sbin/aspppls` được cấu hình như là shell đăng nhập trong các hệ thống server khi người dùng **ppp** thực hiện động tác đăng nhập vào hệ thống.)

❖ **/etc/passwd**

Nội dung file này phải có tên đăng nhập xác định cho mỗi một người dùng **ppp** với shell đăng nhập là `/usr/sbin/aspppls`. Tên đăng nhập phải được kiểm tra xem có trùng với tên được xác định trong phần `"peer_system_name"` của file `/etc/asppp.cf` hay không. Theo đúng quy tắc đặt tên người dùng, chỉ có 8 ký tự đầu là có nghĩa khi đặt tên những người dùng đó.

Sau đây ta sẽ lần lượt theo dõi các thao tác cụ thể để cấu hình **ppp** trong các tình huống sử dụng khác nhau.

### ***XV. 3. c. Tạo người dùng bình thường để khởi động và kết thúc daemon ppp***

Ta đã biết daemon **aspppd** có chủ quyền sở hữu hạn chế đối với những người dùng bình thường. Nhưng trong thực tế sử dụng, nhiều khi những người dùng bình thường cần sử dụng dịch vụ **ppp** trong điều kiện không có người quản trị ở gần. Vì vậy, việc tạo khả năng cho họ có thể bắt đầu hoặc kết thúc sử dụng dịch vụ **ppp** là cần thiết. Để làm được việc đó, ta cần biên dịch hai chương trình ngắn sau:

❖ **pppstart.c**

```
#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>
```

```
main() {
```

```
 setuid(geteuid())
 setgid(getegid())
 system("/etc/init.d/asppp start")
}
```

❖ pppstop.c:

```
#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>

main() {
 setuid(geteuid())
 setgid(getegid())
 system("/etc/init.d/asppp stop")
}
```

Để biên dịch chúng, ta gõ các dòng lệnh sau. Chú ý rằng bộ biên dịch C - **cc** được kèm vào trong các phiên bản Solaris từ 2.5.1 trở về trước như một thành phần của hệ điều hành và ta có thể sử dụng trôi chảy, không gặp vướng mắc gì. Nhưng trong các bản Solaris sau này, bộ biên dịch **cc** cần có thêm mô đun hỗ trợ hiển thị bằng các ngôn ngữ bản địa mới có thể hoạt động được. Hay nói cách khác, **cc** không còn là miễn phí (có nghĩa là không mặc nhiên đi kèm với các gói phần mềm cài đặt). Có thể khắc phục trở ngại này bằng các giải pháp như sau. Hoặc ta phải mua bộ phần mềm ngôn ngữ lập trình phát triển C Workshop. Giải pháp này đòi hỏi chi phí lớn, thường chỉ những cơ sở lớn có bộ phận nghiên cứu phát triển mới áp dụng để phát triển hệ thống và các ứng dụng riêng. Hoặc ta có thể trông chờ vào các tiện ích miễn phí như **gcc** đã được đề cập tới trong giáo trình này. Để đơn giản, trong ví dụ này chỉ biểu diễn các dòng lệnh với bộ biên dịch **cc**:

```
cc -o pppstart pppstart.c
cc -o pppstop pppstop.c
chown root pppstart pppstop
chmod 4755 pppstart pppstop
```

### ***XV. 3. d. Thiết lập các máy client ppp để quay số ra ngoài***

Trong ví dụ ở mục này ta chỉ xét máy tính với một cách kết nối duy nhất với bên ngoài thông qua giao diện **ppp** mà không xét tới khía cạnh mạng Ethernet tại chỗ. Trường hợp máy tính còn nằm trong mạng Ethernet tại chỗ sẽ được mô tả trong phần *Thiết lập nối kết PPP mạng-tới-mạng*. Máy tính được xét tới ở đây – có tên ví dụ là sun - được nối đến nhà cung cấp dịch vụ Internet (ISP), nơi cung cấp hai địa chỉ IP tĩnh (cố định) cho nối kết PPP của nó và của gateway, cũng như địa chỉ IP của máy chủ dịch vụ tên vùng (*Domain Name Service* - DNS).

Chú ý: Trong các bản Solaris 2.3 và 2.4, **ppp** không hỗ trợ việc gán địa chỉ IP động cho các máy. Nếu cần dùng các địa chỉ IP động được chỉ ra trong file `asppp.cf`, ta phải cài đặt Solaris ít nhất là từ phiên bản 2.5 trở đi.

Thông thường ISP cung cấp cho ta các dữ liệu sau (các số liệu cụ thể được nêu ra cho ví dụ minh họa):

- Địa chỉ IP cho máy tại chỗ (trong ví dụ này ta dùng 192.202.200.1).
- Địa chỉ IP cho gateway (ta dùng 192.202.200.253).
- Địa chỉ IP cho DNS server (ta dùng 192.202.200.10).
- Tên vùng (ta dùng `mycoquan.com`)

Sau đây sẽ là các bước tiến hành để thiết lập đúng máy client PPP:

1) Cài đặt hệ điều hành Solaris vào máy, chọn tùy chọn "Networked". Đối với dịch vụ tên, chọn "OTHER". Không chọn "NIS+" hay "NIS".

2) Vì không dùng mạng Ethernet tại chỗ nên ta xóa bỏ file `/etc/hostname.le0` để tránh phải cấu hình giao diện mạng (card mạng) `le0` và kích hoạt nó khi hệ thống khởi động. (Lúc này, giao diện chính sẽ là giao diện PPP, nó sẽ được cấu hình khi `S47asppp` được thực hiện). Phải đảm bảo tên máy `hostname` trong file `/etc/nodename` là đúng và tương ứng với địa chỉ IP của nối kết PPP.

3) Tạo file `/etc/asppp.cf` với nội dung như sau:

```
ifconfig ipdptp0 plumb 192.202.200.1 192.202.200.253 up
```

```

path
 interface ipdptp0
 inactivity_timeout 300
 # hủy bỏ nối kết rồi sau 5 phút
 peer_system_name myprovider
trùng với "myprovider" trong file /etc/uucp/Systems
 ipcp_async_map 0
 # hiệu quả hơn
debug_level 5
 # bỏ dấu thăng để kích hoạt gỡ rối script hội thoại
debug_level 9
 # bỏ dấu thăng để kích hoạt gỡ rối trao đổi ppp

```

4) Tạo file /etc/defaultrouter có nội dung: 192.202.200.253

5) Tạo file /etc/resolv.conf:

```

domain mycoquan.com
nameserver 192.202.200.10

```

6) Thiết lập nsswitch.conf sao cho hệ thống được coi như thành viên trong dịch vụ tên vùng:

```

cp /etc/nsswitch.files /etc/nsswitch.conf
vi /etc/nsswitch.conf

```

và thêm "dns" vào trong dòng vào của file nsswitch.conf, sao cho dòng vào đó có dạng như sau:

```

hosts: dns files

```

7) Thiết lập cổng modem như là modem quay số ra ngoài (dial-out modem) hoặc là modem quay số hai chiều (bi-directional modem) bằng cách sử dụng phần quản lý cổng nối tiếp của tiện ích **Admintool**.

8) Tạo dòng vào trong file `/etc/uucp/Systems` cho "myprovider". Tên này trùng với tên của dòng `peer_system_name` trong file `asppp.cf`. Chú ý rằng tên tài khoản người dùng và mật khẩu đăng nhập trong ví dụ của ta tương ứng là `mylogin` và `mypaswd`.

```
cd /etc/uucp
grep myprovider Systems
myprovider Any ACUTEC 9600 9030874 "" P_ZERO ""
login:-\r\n\c-login:-\r\n\c-login:-EOT-login: mylogin
 ssword: mypaswd

grep ACUTEC Devices
ACUTEC cua/a - Any tb9600-ec

grep tb9600-ec Dialers
tb9600-ec =W-, ""
```

```
\dA\pA\pA\pT&D2E1V1X1Q0S2=255S12=255S50=6S58=2\r\c
OK\r \EATDT\T\r\c CONNECT\s9600 STTY=crtsets
```

Đến đây ta phải khởi động lại máy. Sau đó, thực hiện lệnh **ping** đến đầu kia của nối kết PPP (trong ví dụ này là 192.202.200.253) để bắt đầu kết nối.

Chú ý:

- `ipd` là viết tắt tên loại giao diện *IP/Dialup*, còn `ipdptp` - giao diện *IP/Dialup Point-to-Point*, chữ số tiếp theo (bắt đầu từ số 0) chỉ số thứ tự của giao diện cùng loại trong hệ thống.
- Ba dòng vào trong file `Systems` và hai dòng vào trong file `Dialers` là trùng nhau và thực chất là một dòng vào.

Nếu bạn chạy `/etc/init.d/asppp stop` hệ thống có thể bị treo hoặc tốc độ truyền nhận bị giảm nhiều do giao diện chính đã bị tắt. Tình huống này có thể gây ra lỗi trong một số ứng dụng. Vì vậy, để dùng **ppp** ta phải luôn nhớ gọi thực hiện lệnh `/etc/init.d/asppp start`.

Trong trường hợp này ta sẽ sử dụng việc phân đường cố định, tức là không gọi chạy script `in.routed`, vì file `/etc/defaultrouter` đã xác định đường truyền mặc định cho dữ liệu.

Nếu không sử dụng DNS, đơn giản ta chỉ cần bỏ qua các bước 5 và 6 và đảm bảo tên của các hệ thống mà bạn muốn kết nối đến đều được liệt kê trong file `/etc/hosts`.

Cũng cần chú ý rằng modem mà ta sử dụng trong ví dụ này ("`tb9600-ec`") là loại Telebit trailblazer tốc độ 9600 bps có chức năng sửa lỗi. Ta phải thay dòng vào tương ứng trong file `Dialers` cho phù hợp với loại modem được dùng trong thực tế.

Đặc biệt phải chú ý tới hai dãy ký tự gửi đi / chờ nhận về trong các file **uucp**. Dãy ký tự gửi đi/chờ nhận về " " `P_ZERO` trong file `/etc/uucp/Systems` đặt chế độ 8 bit/no parity khi đăng nhập vào hệ thống khác. Còn dãy ký tự gửi đi `STTY=crtsets` trong file `/etc/uucp/Dialers` thiết lập việc điều khiển luồng thông tin bằng phần cứng cho đường truyền modem đi ra ngoài.

Các chi tiết này đều rất quan trọng, nếu chúng bị bỏ qua ta có thể gặp các sự cố khác nhau khi sử dụng các dịch vụ truyền thông.

### ***XV. 3. e. Thiết lập các máy client của ppp có cải tiến để quay số ra ngoài***

Khâu khởi tạo nối kết truyền thông đóng một vai trò quan trọng và bao giờ người vận hành thiết bị cũng muốn có nhiều phương tiện kiểm tra và theo dõi. Trong trường hợp hệ thống được thiết lập như phần trước, nhưng trong **ppp** lại hay gặp phải khó khăn khi khởi tạo hoặc kết thúc nối kết, ta có thể tìm cách khắc phục bằng các thay đổi sau đây:

Không tạo file `/etc/defaultrouter` như trong bước 4 của phần trên, thay vào đó thêm dòng vào sau vào file `/etc/asppp.cf`:

```
default_route
```

Để bắt đầu nối kết, tất nhiên bạn phải gửi đi một gói thông tin tới đầu kia của nối kết PPP (trong ví dụ này là `192.202.200.253`). Khi nối kết đã được thành lập, đường đi mặc định được thêm vào nhờ đã có đầu kia của nối kết. Nối kết sẽ được



giữ hoạt động bình thường cho đến khi hết giá trị thời gian của `inactivity_timeout`, và khi nối kết đó bị tắt đi, đường đi mặc định của dữ liệu cũng bị loại bỏ. Khi tắt nối kết, tất cả các kênh truyền theo DNS cũng không còn vì không có đường tới máy chủ DNS. Tóm lại, khi ta kết thúc dùng nối kết PPP, nối kết này sẽ không được tự động bật lại và sẽ ở trạng thái tắt cho đến lần được gọi sau.

Chú ý thêm rằng bằng thay đổi trên ta đã cho chạy script `in.routed` ở chế độ kín, với giá thiết từ đầu rằng không tồn tại file `/etc/gateways` ngoài giao diện PPP duy nhất (tất nhiên không kể tới giao diện tự tìm loopback `lo0`).

### ***XV. 3. f. Thiết lập máy chủ ppp trong mạng LAN, với các địa chỉ IP tĩnh***

Trong ví dụ ở phần này, ta sẽ thiết lập một server có hai modem quay số vào và ba tài khoản người dùng PPP cho những người dùng có thể đăng nhập hệ thống từ nhà hay từ máy ở xa. Khi người dùng PPP quay số và đăng nhập vào hệ thống, một giao diện đang rồi mới được cấu hình để sử dụng, vì đây là cấu hình điểm-tới-diểm động (được chỉ ra bởi `ipdptp*`). Số giao diện mạng `ipdptp` cực đại trong Solaris 7 và 8 là 64, được đánh số thứ tự từ 0 đến 63. Trong ví dụ này mỗi người dùng ppp đều được gán địa chỉ IP tĩnh (từ phía máy của họ).

1) Kiểm tra bằng lệnh `ifconfig`, và ghi lại địa chỉ Ethernet để dùng cho script `S99proxyarp` sau này:

```
/usr/sbin/ifconfig le0
le0:
flags=863<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST
> mtu 1500 inet 192.202.200.1 netmask ffffffff0
broadcast 192.202.200.255 ether 8:0:20:1:2:3
```

2) Sửa đổi các file `hosts` và `asppp.cf`:

```
☞ /etc/hosts:
127.0.0.1 localhost
192.202.200.1 myserver loghost
```

```

192.202.200.100 Tuanhome
192.202.200.101 Kimhome
192.202.200.102 MinhPC

```

✎ `/etc/asppp.cf:`

```

ifconfig ipdptp0 plumb myserver down
modem thứ nhất
ifconfig ipdptp1 plumb myserver down
bỏ dấu chú thích nếu dùng cả modem thứ hai
ifconfig ipdptp2 plumb myserver down
bỏ dấu chú thích nếu dùng cả modem thứ ba

```

defaults

```

 interface ipdptp*
 inactivity_timeout 120
 ipcp_async_map 0
debug_level 9

```

# bỏ dấu chú thích để gỡ rối

path

```

 peer_system_name tuanppp
 peer_ip_address Tuanhome
 interface ipdptp0

```

path

```

 peer_system_name kimppp
 peer_ip_address Kimhome
 interface ipdptp1

```

path

```

 peer_system_name minhppp
 peer_ip_address MinhPC
 interface ipdptp2

```

3) Các bước cấu hình trên mới đảm bảo cho người dùng ppp quay số vào có thể thiết lập được nối kết PPP với server và thực hiện các thao tác dịch vụ TCP/IP như **telnet** để đăng nhập hệ thống, **ftp** để truyền nhận file hay **http** để nối với Server Web chạy trên máy chủ Solaris. Nhưng đến đây, người dùng ppp quay số vào chưa thể giao tiếp được với các máy dưới máy chủ Solaris, nghĩa là vào các máy khác trong mạng LAN. Để làm được việc đó ta phải kích hoạt việc chuyển tiếp IP của kernel Solaris, sao cho mọi gói thông tin bất kỳ từ người dùng ppp quay số vào định gửi tới máy khác trong mạng đều được nhận trên giao diện **ipdptp** quay số vào và được chuyển tiếp tới mạng LAN. Tương tự như vậy, theo chiều ngược lại, mọi gói thông tin từ một máy nào đó trong mạng LAN có thể qua server Solaris thông qua giao diện **ipdptp** của nó để tới địa chỉ của người dùng ppp quay số vào. (Ta có thể bỏ qua bước này nếu chỉ cần hệ thống Solaris được cấu hình như trong phần trước, nghĩa là tách biệt, không nằm trong mạng LAN).

Tạo file thực thi có tên **/etc/rc2.d/S99proxyarp**. File này sẽ có dòng vào arp cho mỗi một client ppp, trong đó có ghi địa chỉ Ethernet của server:

```
ndd -set /dev/ip ip_forwarding 1
arp -s Tuanhome 8:0:20:1:2:3 pub
arp -s Kimhome 8:0:20:1:2:3 pub
arp -s MinhPC 8:0:20:1:2:3 pub
```

Thêm quyền chạy cho file đó:

```
chmod +x /etc/rc2.d/S99proxyarp
```

4) Thêm các dòng vào sau vào file **/etc/passwd**:

3

```
tuanppp:x:1200:1045:tuan ppp login:/:usr/sbin/aspppls
kimppp:x:1201:1045:kim ppp login:/:usr/sbin/aspppls
minhppp:x:1203:1045:minh ppp login:/:usr/sbin/aspppls
```

Ta cũng có thể thay bước 4 bằng cách dùng tiện ích **Admintool** để tạo lần lượt các tài khoản người dùng trên. Lấy ví dụ cho người dùng **tuanppp**, chọn mục

*Users* trong thực đơn *Browse* , sau đó chọn *Add* từ thực đơn *Edit*. Điền tên người dùng, *tuanppp*, vào trong hộp thoại *User Name* và giữ nguyên không đổi các giá trị *User ID* và *Primary Group* nếu bạn quyết định để cho hệ thống tự chọn. Gõ vào dòng mô tả, thường là tên đầy đủ của người dùng, vào trong hộp *Comment*. Bên cạnh hộp thoại *Login Shell*, chọn *Other*, xóa xâu ký tự */bin/* và gõ vào:

```
/usr/sbin/aspppls
```

Xâu trên sẽ là shell đăng nhập của người dùng *tuanppp* và khi *tuanppp* đăng nhập hệ thống, shell đó sẽ gọi chạy dịch vụ **ppp**. Nháy phím chuột trái ở nút bấm cạnh *Create home dir* sao cho nút bấm sổ xuống và gõ */tmp* vào hộp thoại *Path*. Giữ nguyên các trường khác và nháy chuột vào nút *OK* để tạo tài khoản người dùng. **Admintool** sẽ hiện thông báo "Home Directory is owned by a different user". Ta lại nháy chuột vào nút *OK* (người dùng *ppp* quay số vào không có và không cần thư mục *home*) rồi kết thúc tiện ích **Admintool**.

5) Thêm các dòng vào cho file */etc/shadow*:

```
tuanppp:::::::::
kimppp:::::::::
minhppp:::::::::
```

Dùng lệnh **passwd** để đặt mật khẩu cho những người dùng trên.

#### Chú ý:

Bằng các động tác trên ta đã thiết lập proxy arp để cho phép dùng các địa chỉ với cùng một địa chỉ mạng của giao diện mạng, ở đây là giao diện *le0*.

Hệ thống này hoạt động như một router khi gửi các gói RIP (xem lại phần *Giao thức TCP/IP*) ra tất cả các giao diện cho tới khi nó được cấu hình để chạy các giao thức phân đường trong chế độ kín.

### ***XV. 3. g. Thiết lập máy chủ ppp quay số vào với các địa chỉ IP động***

Ưu điểm của hệ thống này nằm ở chỗ chỉ cần hai modem và gán hai địa chỉ IP trong ví dụ này là 192.202.200.100 và 192.202.200.101 (vì trong hầu hết các

trường hợp, hai người có thể đăng nhập hệ thống đồng thời). Nhiều tài khoản người dùng có thể được đăng ký và mỗi một trong số tài khoản đó sẽ được gán địa chỉ IP động hay địa chỉ phụ thuộc vào tình huống cụ thể khi đăng nhập vào hệ thống.

Nhược điểm của trình tự này với việc sử dụng địa chỉ IP động còn tồn tại khá rõ. Thường tồn tại một quan niệm sai rằng địa chỉ IP liên hệ với máy tính nên rất khó xác định các địa chỉ IP được sử dụng thế nào trong nối kết quay số. Nói chính xác hơn, mỗi địa chỉ IP liên quan tới một giao diện mạng, và một máy tính có thể có vài địa chỉ IP do có thể được lắp tới vài card mạng. Cho đến Solaris 2.5, khi nối kết PPP lần đầu được thiết lập, các địa chỉ IP động được sử dụng bình thường cho các client từ xa. Nhưng một khi tắt nối kết và thiết lập lại nó lần sau, các địa chỉ IP tự động nhảy về giá trị 255.255.255.255, và nối kết PPP bị hỏng hoàn toàn. Có thể đó là do lỗi phần mềm, nhưng cũng có thể do ta chưa đặt đúng địa chỉ IP cần dùng.

Bình thường các client của hệ thống Solaris cần địa chỉ IP tĩnh nên ví dụ này không áp dụng cho các bản Solaris 2.4 trở về trước. Tốt nhất là ta nên sử dụng với Solaris 7 hoặc 8 trở đi, nếu muốn sử dụng các địa chỉ IP động. Trình tự thiết lập máy chủ PPP quay số vào với địa chỉ IP động như sau:

1) Kiểm tra bằng lệnh **ifconfig**, và ghi lại địa chỉ Ethernet (của máy chọn làm máy chủ) để dùng trong file **S99proxyarp** ở các bước sau:

```
/usr/sbin/ifconfig le0
le0:
flags=863<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST>
mtu 1500 inet 192.202.200.1 netmask ffffffff00 broadcast
192.202.200.255 ether 8:0:20:1:2:3
```

2) Sửa đổi nội dung các file **hosts** và **asppp.cf**:

```
/etc/hosts:
127.0.0.1 localhost
192.202.200.1 myserver loghost
192.202.200.100 client1
192.202.200.101 client2
```

```

/etc/asppp.cf:
 ifconfig ipdptp0 plumb myserver client1 down
 ifconfig ipdptp1 plumb myserver client2 down

defaults
 interface ipdptp*
 inactivity_timeout 120
 ipcp_async_map 0
 negotiate_address on
debug_level 9
 # mức 9 : ghi log đầy đủ

path peer_system_name pppclient1
path peer_system_name pppclient2
path peer_system_name pppclient3
path peer_system_name pppclient4
path peer_system_name pppclient5
path peer_system_name pppclient6

```

3) Tạo file thực thi có tên `/etc/rc2.d/S99proxyarp`. Nội dung của file này phải gồm một dòng vào `arp` cho mỗi một client PPP và địa chỉ Ethernet của server:

```

ndd -set /dev/ip ip_forwarding 1
 arp -s client1 8:0:20:1:2:3 pub
 arp -s client2 8:0:20:1:2:3 pub

```

Ta thêm quyền thực hiện cho file đó:

```
chmod +x /etc/rc2.d/S99proxyarp
```

4) Thêm các dòng vào sau cho file `/etc/passwd`:

```
pppclient1:x:1200:1045:pc login:/:usr/sbin/aspppls
```

```
pppclient2:x:1201:1045:pc login:/:usr/sbin/aspppls
pppclient3:x:1202:1045:pc login:/:usr/sbin/aspppls
pppclient4:x:1203:1045:pc login:/:usr/sbin/aspppls
pppclient5:x:1204:1045:pc login:/:usr/sbin/aspppls
pppclient6:x:1205:1045:pc login:/:usr/sbin/aspppls
```

Và cập nhật file /etc/shadow, làm cho các người dùng trên có hiệu lực, tương tự như trong phần trước.

### XV. 3. h. Thiết lập nối kết ppp Mạng-tới-Mạng

Trong phần này ta hãy xem xét một ví dụ mô tả việc cấu hình mạng tới mạng, khi mà mỗi máy tham gia vào nối kết PPP đều có mạng riêng của mình. Địa chỉ IP của giao diện mạng le0 (card mạng) và của giao diện mạng ipdptp0 là như nhau đối với mỗi máy (trong trường hợp Solaris 2.3, 2.4 cần cài đặt thêm một phần sửa chữa lớn lần cuối cho kernel, được lấy, ví dụ từ <http://www.sunsolve.com>). Trong ví dụ này có thể thiết lập sao cho mỗi máy của một mạng đều có khả năng đi tới các máy của mạng kia, tức là vừa là máy quay số ra, vừa là máy quay số vào. Phải chú ý rằng mỗi máy chủ của một mạng đều có địa chỉ mạng khác với của máy kia, được phân biệt rõ trong file /etc/gateways.

☛ Đầu tiên, ta cần sửa đổi cho máy quay số ra ngoài DIAL-OUT MACHINE.

1) Tạo file aspppp.cf:

☛ /etc/aspppp.cf:

```
ifconfig ipdptp0 plumb 192.202.200.1 200.100.99.25 up

path
 interface ipdptp0
 peer_system_name server
 inactivity_timeout 120
 ipcp_async_map 0
debug_level 9
mức 9 : ghi log đầy đủ
```

2) Sửa nội dung file `/etc/gateways` nhằm:

- Ngăn các đường khỏi bị gửi đi như những gói RIP qua giao diện PPP ( sẽ được giữ ở trạng thái hoạt động sau khoảng thời gian `inactivity_timeout`).
- Thêm đường tới mạng kia sao cho các máy thuộc mạng này có thể tìm được đường trở về. Máy chủ này sẽ phổ biến đường này tới mạng tại chỗ thông qua việc sử dụng `in.routed`.

# `/etc/gateways:`

```
norip ipdptp0
net 200.100.99.0 gateway 200.100.99.25 metric 1 \
 passive
```

3) Tắt việc phân đường của router (sao cho nối kết không tự động bật lên khi máy khởi động và không được giữ ở trạng thái nối thông do các gói của `in.rdisc` được gửi cứ 10 phút một lần):

```
cd /usr/sbin
mv in.rdisc in.rdisc.save
```

4) Thiết lập cổng modem như là modem quay số ra ngoài hay modem hai chiều bằng cách dùng bộ quản lý cổng của tiện ích **Admintool**.

5) Tạo dòng vào trong file `/etc/uucp/Systems` cho "server". Tên "server" này phải trùng với tên được chỉ trong dòng `peer_system_name` của file `asppp.cf`.

```
cd /etc/uucp
grep myprovider Systems
server Any ACUTECH 9600 9030874 "" P_ZERO ""
 login:-\r\n\c-login:-\r\n\c-login:-EOT-
 login:mylogin
 ssword: mypasswd
```



```
grep ACUTEK Devices
ACUTEK cua/a - Any tb9600-ec
```

```
grep tb9600-ec Dialers
tb9600-ec =W-, ""
```

```
\dA\pA\pA\pT&D2E1V1X1Q0S2=255S12=255S50=6S58=2\r\c
OK\r \EATDT\T\r\c CONNECT\s9600 STTY=crtscts
```

● Tiếp theo, ta sửa cho máy quay số vào DIAL-IN MACHINE:

Bạn cần thiết lập máy như là máy chủ quay số vào khi nối kết được lập để đường đi tới mạng khác được thêm vào file /etc/gateways. Đôi khi ta gặp phải thông báo kiểu như "Call Failed: SYSTEM NOT IN Systems FILE " trong file log /etc/log/asppp.log; thông báo này có thể được ta bỏ qua mà không có hậu quả xấu nào.

Bạn cũng có thể biến server trong mạng tại chỗ thành máy chủ vừa quay số ra, vừa quay số vào bằng cách thêm dòng vào "mylogin" vào file /etc/uucp/Systems để có thể quay số ra ngoài. Các máy client phải đăng ký tài khoản đăng nhập vào máy chủ tại chỗ để người dùng ppp đó khởi động shell /usr/sbin/aspppls. Nhưng phải chú ý nếu máy chủ thực hiện cả hai chức năng quay số ra và vào, ta dễ lâm phải tình huống lờ đờ: cả hai hệ thống đồng thời cùng cố gắng quay số cho nhau. Trước hết, ta cấu hình máy chủ quay số vào, còn việc cấu hình để máy thực hiện cả hai chức năng quay số vào và ra sẽ được trình bày trong một mục riêng:

1) Tạo file /etc/asppp.cf cho máy chủ quay số vào:

```
/etc/asppp.cf:
```

```
ifconfig ipdptp0 plumb 200.100.99.25 192.202.200.1 up
```

```
path
```

```
 interface ipdptp0
 , peer_system_name mylogin
 inactivity_timeout 120
```

```

 ipcp_async_map 0
debug_level 9

```

2) Tạo tài khoản người dùng để thực hiện dịch vụ đăng nhập PPP bằng cách thêm dòng vào tới nội dung file `/etc/passwd`:

```
mylogin:x:1200:1045:tuan ppp login:/:usr/sbin/aspppls
```

và thêm dòng vào sau vào file `/etc/shadow`:

```
mylogin:::::::::
```

3) Sửa nội dung file `/etc/gateways` nhằm:

- Ngăn các đường khỏi bị gửi đi như những gói RIP qua giao diện PPP ( sẽ được giữ ở trạng thái hoạt động sau khoảng thời gian `inactivity_timeout`).
- Thêm đường tới mạng kia sao cho các máy thuộc mạng này có thể tìm được đường trở về. Máy chủ này sẽ phổ biến đường này tới mạng tại chỗ thông qua việc sử dụng `in.routed`.

```

/etc/gateways
norip ipdptp0
net 192.202.200.0 gateway 192.202.200.1 metric 1 \
 passive

```

4) Tắt việc phân đường của router (sao cho nối kết không tự động bật lên khi máy khởi động và không được giữ ở trạng thái nối thông do các gói của `in.rdisc` được gửi cứ 10 phút một lần):

```

cd /etc
mv in.rdisc in.rdisc.save

```

### *XV. 3. i. Thiết lập client ppp quay số ra ngoài, với địa chỉ IP động*

Việc nhận các địa chỉ IP động được máy chủ PPP gán cho máy tại chỗ đóng vai trò máy client chỉ có thể thực hiện được từ phiên bản Solaris 2.5. Máy chủ thực hiện việc gán địa chỉ động hay gặp nhất chính là ISP. Giả sử địa chỉ IP tĩnh của ISP (ta luôn được cung cấp địa chỉ này một cách dễ dàng) là 203.162.1.10, với tên là *isp*.

Ta hãy xét file cấu hình cho **ppp** như sau (các dòng cần chú ý được in đậm):

```
ifconfig ipdptp0 plumb 192.1.22.99 203.162.1.10 up

path
 interface ipdptp0
 peer_system_name isp
 inactivity_timeout 120
 ipcp_async_map 0
 # Use async control character map 0
 negotiate_address on
 default_route
debug_level 9
```

Trong file `/etc/asppp.cf` trên, địa chỉ IP thứ nhất (192.1.22.99) là địa chỉ IP giả, chỉ có tác dụng tồn tại để khởi tạo giao diện PPP, `ipdptp0`, khi bắt đầu thiết lập nối kết PPP. Nếu sử dụng việc gán địa chỉ IP động, địa chỉ IP ban đầu không có nghĩa gì vì một khi nối kết PPP được thiết lập xong, ISP ở đầu phía bên kia của nối kết lập tức gán cho máy của ta một địa chỉ IP khác.

Địa chỉ IP thứ hai (203.162.1.10) trong dòng `ifconfig` của file cấu hình trên là địa chỉ IP của ISP. Địa chỉ này là địa chỉ IP tĩnh - việc gán địa chỉ IP động chỉ thực hiện đối với các client mà không áp dụng đối với server.

Dòng lệnh thứ nhất được cố ý in đậm trong file trên là điểm mới được đưa vào từ Solaris phiên bản 2.5, có tác dụng bật (trong ví dụ này) hay tắt việc dùng địa chỉ IP động. Tuy trang **man** của **aspppd** nói rằng giá trị mặc định của thông số này là bật ("on"), nhưng để chắc chắn, bạn hãy viết lại câu lệnh và đặt nó với "on".

Dòng in đậm thứ hai cho phép đặt lại giá trị của `default_route` mỗi lần một nối kết PPP được thành lập và một địa chỉ IP mới được gán.

### ***XV. 3. j. Dùng ppp quay số vào và quay số ra trên cùng một hệ thống***

Mức độ phức tạp của việc cấu hình trong trường hợp này lớn hơn, vì ta phải xử lý file cấu hình `/etc/asppp.cf` cho cả hai chức năng quay số theo chiều ngược nhau. Nếu bạn muốn quay số bây giờ, rồi sau lại quay số tiếp từ hệ thống Solaris tới ISP, nhưng vẫn muốn để hệ thống được cấu hình quay số PPP vào trong các thời điểm khác, bạn phải bố trí sao cho phương án quay số vào của `/etc/asppp.cf` phải hoạt động khi hệ thống Solaris khởi động, rồi ngắt nối kết, kích hoạt phương án quay số ra của `/etc/asppp.cf` khi muốn kết nối với ISP, sau đó lại đưa phương án PPP quay số vào trở lại hoạt động ở trạng thái bình thường. Để có thể linh hoạt sử dụng các phương án cấu hình PPP trên trong cùng một máy chủ Solaris, cách đơn giản và hiệu quả hơn cả là ta thực hiện việc điều khiển thông qua các shell script `dialup` và `hangup`.

Shell script `dialup`, dùng để nối hệ thống của ta tới ISP, có thể có các dòng lệnh kiểu như sau:

```
cp -p /etc/resolv.conf.isp /etc/resolv.conf
cp -p /etc/asppp.cf.isp /etc/asppp.cf
cp -p /etc/nsswitch.conf.isp /etc/nsswitch.conf
/etc/init.d/asppp start
/usr/sbin/ping 203.162.1.10
```

Ta sửa lại script này để kết thúc daemon **aspppd** trước khi phương án PPP quay số ra của file `/etc/asppp.cf` được sao chép vào vị trí, nhằm mục đích tắt việc chuyển mạch IP và xóa dòng vào `arp` của các client PPP quay số vào. Sau khi sửa, script `dialup` sẽ có dạng như:

```
/etc/init.d/asppp stop
/usr/sbin/ndd -set /dev/ip ip_forwarding 0
/usr/sbin/arp -d pppclient1
... ..
```

---

```
cp -p /etc/resolv.conf.isp /etc/resolv.conf
cp -p /etc/asppp.cf.isp /etc/asppp.cf
cp -p /etc/nsswitch.conf.isp /etc/nsswitch.conf
/etc/init.d/asppp start
/usr/sbin/ping 203.162.1.10
```

Nếu có nhiều hơn một client PPP quay số vào, ở vị trí bỏ trống với các dấu "...", ta phải thêm các dòng lệnh xóa việc đặt proxy arp cho mỗi client đó:

```
/usr/sbin/arp -d pppclient1
/usr/sbin/arp -d pppclient2
...
```

Các thay đổi tương tự cũng được thực hiện cho script hangup. Trước khi sửa đổi, script hangup có thể có dạng:

```
/etc/init.d/asppp stop
rm /etc/resolv.conf
cp -p /etc/asppp.cf.original /etc/asppp.cf
cp -p /etc/nsswitch.conf.original /etc/nsswitch.conf
/usr/sbin/route -f
```

Ta phải sửa file hangup sao cho nó đưa trở lại chuyển mạch IP, xác định các dòng vào proxy arp cho mỗi client PPP quay số vào, cài đặt lại phương án PPP quay số vào của file cấu hình /etc/asppp.cf, cuối cùng là khởi động lại daemon **aspppd**. Nội dung của shell script hangup sau khi sửa đổi sẽ có dạng:

```
/etc/init.d/asppp stop
rm /etc/resolv.conf
cp -p /etc/asppp.cf.dialin /etc/asppp.cf
cp -p /etc/nsswitch.conf.original /etc/nsswitch.conf
/usr/sbin/route -f
/usr/sbin/ndd -set /dev/ip ip_forwarding 1
/usr/sbin/arp -s ppp_client1 8:0:20:1:2:3 pub
```

```
/usr/sbin/arp -s ppp_client2 8:0:20:1:2:3 pub
...
/etc/init.d/asppp start
```

### ***XV. 3. k. Cấu hình Windows 9x để quay số tới hệ thống Solaris***

Do Microsoft Windows và UNIX có những khác biệt rõ rệt trong các giao thức và cách tổ chức các dịch vụ truyền thông nên việc cấu hình mạng qua quay số điện thoại (*Dial Up Networking* - DUN) của các máy trạm cài đặt Windows 95/98 để tạo nối kết PPP với hệ thống Solaris có những chi tiết cần đặc biệt chú ý. Trong nối kết DUN của MS Windows, việc xác định tư cách người dùng được thực hiện và hoàn thành tại máy chủ (Windows NT/2000), máy trạm chỉ chờ dịch vụ truyền thông của máy chủ được phát triển đến. Nhưng trong dịch vụ PPP của hệ Solaris (và UNIX nói chung) quá trình được tiến hành theo cách kiểm tra trao đổi tên người dùng/mật khẩu bình thường giữa máy client và server, và chỉ sau khi hoàn thành công đoạn kiểm tra trên, dịch vụ PPP mới được thiết lập giữa hai đầu của nối kết. Vì vậy, ta phải cấu hình các máy Windows 95/98 sao cho DUN của Windows hiển thị hộp thoại đăng nhập. Dưới đây là một trong các cách cấu hình nhằm mục đích trên.

Ta nhấp đúp chuột trái tại biểu tượng *My Computer* trên màn hình làm việc của Windows, rồi nhấp đúp chuột trên biểu tượng *Dial-Up Networking*, sau đó tiếp tục lại gọi *Make New Connection*. Trong hộp thoại có tên *Type a name for the computer you are dialing*, ta gõ tên, ví dụ như *Solaris Connection*, để đặt tên cho nối kết ta muốn tạo ra, rồi tiếp tục nhấp chuột lên *Next*. Đưa các giá trị tương ứng vào các trường *Area code* và *Telephone number* của modem được nối với hệ thống Solaris quay số vào, nhấp chuột lên các nút *Next* rồi *Finish*.

Trên nối kết *Solaris Connection* vừa được tạo trong thư mục *Dial Up Networking* ta nhấp phím chuột phải, sau đó chọn *Properties*. Nhấp chuột trên bảng *Server Types* và bỏ hết các dấu chọn trong các ô kiểm tra cạnh các trường *Log on to network*, *Enable software compression*, *NetBEUI* và *IPX/SPX Compatible*, chỉ để lại có mỗi dấu chọn của *TCP/IP*. Bây giờ nhấp chuột trên nút *TCP/IP Settings...* và đảm bảo rằng *Server assigned IP address* và *Server assigned name server addresses* được chọn. Ta cũng cần kiểm tra *Use IP header compression* và *Use default*

*gateway on remote network* được đánh dấu đã chọn. Nháy chuột trên phím *OK* để trở về bảng *Server Types*. Nháy chuột trên bảng *General* và sau đó nháy chuột tiếp vào nút nằm dưới tên *Configure....* đối với modem. Trong hộp *Modem Properties* mới xuất hiện, nháy chuột để mở bảng *Options* rồi đánh dấu chọn vào trường *Bring up terminal window after dialing*. Cuối cùng, nhấn chuột vào nút *OK* để thoát khỏi hộp thoại *Modem Properties*, và *OK* lại một lần nữa để kết thúc các trao đổi về *DUN Properties*. Nối kết mới DUN đã sẵn sàng hoạt động.

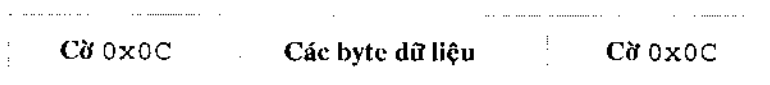
Để kết nối với hệ thống Solaris, ta nháy đúp chuột trái lên biểu tượng của nối kết mới tạo thành. Ta bỏ qua các hộp *User name* và *Password* và kích chuột lên phím *Connect*. Modem sẽ quay số ra và tạo nối kết với modem của hệ thống Solaris. Tại thời điểm này DUN hiển thị cửa sổ của terminal, nơi bạn sẽ thấy dấu nhắc *Login:* từ hệ thống Solaris. Hãy đưa vào tên người dùng của bạn, ví dụ như *tuannppp*, sau đó trả lời dấu nhắc *Password:* , và ấn phím *Enter*. Sau đó nháy chuột trên *Continue* và Windows sẽ bắt đầu **ppp**, hoàn thành các cuộc đối thoại và thiết lập nối kết PPP.

#### **XV. 4. SLIP**

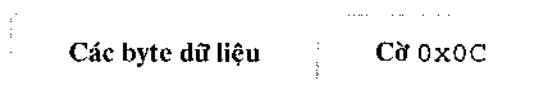
SLIP (Serial Line Internet Protocol), tên viết tắt tiếng Anh của giao thức Internet qua cổng nối tiếp, và **ppp** là hai giao thức truyền thông dùng cho việc truyền nhận dữ liệu tuần tự giữa hai thiết bị. SLIP và **ppp** cho phép máy tính có thể kết nối được với máy chủ qua đường nối tiếp với modem, sau đó có thể truy nhập vào lượng tài nguyên khổng lồ trên Internet. Cả hai giao thức SLIP và **ppp** hiện nay được sử dụng rất rộng rãi trong các mạng máy tính chuyên nghiệp hoạt động không chỉ trên nền hệ điều hành UNIX mà thôi.

##### ***XV. 4. a. Khái niệm về SLIP***

SLIP bắt đầu được mô tả và sử dụng từ năm 1988, tuy vậy nó lại không phải là một chuẩn của Internet. SLIP thực chất là một sơ đồ phân khung cực kỳ đơn giản, được dùng để gửi các gói IP lên đường truyền nối tiếp: một khung được thành lập nhờ một byte cờ đặc biệt (0x0C) ở phần đầu và một byte nữa ở phần đuôi như được chỉ ra trên hình H.XV. 2. Để đơn giản, cờ ban đầu thường bị bỏ qua, nên cờ kết thúc trở thành mốc phân giới giữa hai khung dữ liệu, như được biểu diễn trên H. XV. 3:



H. XV. 2. Dạng đầy đủ của gói SLIP



H. XV. 3. Dạng rút gọn của gói SLIP

Nếu ngẫu nhiên có một byte dữ liệu nào đó trùng với byte cờ (0x0C), byte dữ liệu đó sẽ được dịch thành cặp ký tự (0xDB 0xDC) trước khi được truyền đi và được tiếp nhận bởi đầu thu của nối kết. Do vậy, ta không bỏ sót byte dữ liệu nào với những lý do tương tự khi dùng SLIP.

Ưu điểm quan trọng của SLIP nằm ở tính đơn giản, do đó nó có độ tin cậy và còn có khả năng hoàn thiện.

SLIP vì rất đơn giản nên không có nhiều các đặc điểm như đối với ppp. SLIP không được cung cấp khả năng tìm lỗi – coi như các dịch vụ ở mức trên trong mô hình mạng phải gánh lấy trách nhiệm này. Vì vậy, nối kết SLIP hoặc là được mở và truyền đi trôi chảy các gói IP, hoặc là bị đóng.

#### ***XV. 4. b. So sánh giữa SLIP với ppp***

Ngược lại với SLIP, ppp là một chuẩn Internet, được đưa vào sử dụng ban đầu như là sản phẩm của một nhóm làm việc có tên là Internet Engineering Task Force's Point to Point (nhóm Điểm-tới-điểm thuộc Lực lượng tác nhiệm kỹ thuật Internet – IETF). ppp là giao thức mới hơn, nhưng cũng thực hiện chức năng truyền các gói IP như SLIP. Vì vậy, ppp hàm chứa nhiều đặc điểm như tìm lỗi trong mỗi khung IP hay trao đổi thỏa thuận về địa chỉ IP, v.v...

H. XV. 4 biểu diễn dạng đầy đủ của một gói PPP, (FCS - *Frame Check Sequence*: dãy ký tự kiểm tra khung). Đối với nhiều nối kết, phần đầu của khung không thực sự cần thiết và có thể được rút gọn. Đôi khi người ta dùng các kỹ thuật nén dữ liệu nên các gói IP đi qua các nối kết chậm trên thực tế lại có dạng giống với dạng được mô tả trên H. XV. 5.



| Cờ   | Địa chỉ | Điều khiển | Giao thức | Các byte dữ liệu | FCS            | Cờ   |
|------|---------|------------|-----------|------------------|----------------|------|
| 0x7E | 0xFF    | 0x03       | 0xFFFF    |                  | 16 hoặc 32 bit | 0x7E |

H. XV. 4. Dạng đầy đủ của gói PPP

| Giao thức | Các byte dữ liệu | FCS    | Cờ   |
|-----------|------------------|--------|------|
| 0xFFFF    |                  | 16 bit | 0x7E |

H. XV. 5. Dạng cực tiểu của gói PPP

Trong quá trình thiết lập nối kết PPP, tất cả các biến có giá trị chưa chắc chắn, có khả năng làm hỏng nối kết, đều là đối tượng để thỏa thuận, ví dụ như kích thước của khung được thông báo trong các bản ppp cải tiến.

Bảng sau so sánh các điểm khác nhau giữa SLIP và ppp. Hiện nay, cả Microsoft và nhiều công ty khác cũng đẩy mạnh việc ứng dụng ppp, ví dụ như trong Windows NT/2000. ppp tỏ ra có cơ sở để trở thành giao thức sẽ được dùng phổ biến trong tương lai.

| Đặc điểm                | SLIP                                                                                                                                                                                                                                                          | PPP                                                                                                                                              |
|-------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Nén dữ liệu</b>      | Mặc định là không có. Chỉ bản nén Compressed SLIP (CSLIP) có hỗ trợ cho cả host và client. Nhưng việc dùng CSLIP đòi hỏi phải can thiệp bằng tay và phải cấu hình cho cả host lẫn client.                                                                     | Có. Việc trao đổi thỏa thuận được tiến hành tự động như là một phần của quá trình kết nối.                                                       |
| <b>Cấu hình kết nối</b> | Phải cố gắng thực hiện thiết lập IP bằng tay, hoặc bằng các chương trình phụ trợ (shell script để đăng nhập bằng cách quay số vào, ...). Đây là khó khăn lớn nhất đối với các server sử dụng địa chỉ IP động vì đòi hỏi thiết lập IP lại mỗi khi kết nối vào. | Tự động. Cấu hình IP là một phần của quá trình kết nối và là trong suốt đối với người dùng. Phần mềm của chương trình ppp phức tạp hơn của SLIP. |

|                                |                                                                                                                                                                                                  |                                                                                                                                                                                          |
|--------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Tìm lỗi</b>                 | Không có. SLIP tỏ ra rất nhạy cảm đối với những đường truyền bị nhiễu nhiều. Thường những modem có tính năng sửa lỗi sẽ bù đắp một phần nhược điểm đó và làm cho người dùng không cảm nhận thấy. | Được cài đặt sẵn tính năng này.                                                                                                                                                          |
| <b>Hỗ trợ công nghiệp</b>      | Không thấy có.                                                                                                                                                                                   | Có rất nhiều hỗ trợ công nghiệp rộng rãi cho ppp. Riêng đối với Microsoft, hỗ trợ cho client đối với Windows95 và Windows NT, nhưng chỉ hỗ trợ cho phía server từ Windows NT 3.5 trở đi. |
| <b>Xử lý giao thức</b>         | Chỉ có IP.                                                                                                                                                                                       | Xử lý đa giao thức, trong số đó cả IP và IPX (cho mạng NetWare).                                                                                                                         |
| <b>Tốc độ</b>                  |                                                                                                                                                                                                  | Trong vài thử nghiệm, cả SLIP và ppp có tốc độ tương tự nhau. Về lý thuyết, khung ppp chứa thêm vài phần ở đầu khung như sửa lỗi, nên có làm giảm tốc độ đi chút ít.                     |
| <b>Chuẩn</b>                   | SLIP chỉ là “chuẩn” trong thực tế, không phải là chuẩn chính thức. Hiện nay vẫn được dùng nhiều.                                                                                                 | ppp là giao thức Internet chuẩn chính thức.                                                                                                                                              |
| <b>Phương pháp truyền nhận</b> | Dị bộ                                                                                                                                                                                            | Đồng bộ và dị bộ.                                                                                                                                                                        |

#### *XV. 4. c. Chạy SLIP trên hệ thống Linux*

Trong phần này ta hãy tìm hiểu cách thiết lập một máy có cài đặt Linux để sử dụng dịch vụ SLIP. Để có xác định cụ thể, ta giả sử rằng máy Linux có nối mạng và được cài đặt bản Linux của Slackware.

##### ■ *slip* .

*slip* là chương trình trên Linux dùng để quản lý việc thiết lập các nối kết cho SLIP. Lệnh này thường nằm ở thư mục `/sbin` - hoặc nếu không được tìm thấy trong máy của bạn, được tải về từ các nhà cung cấp, ví dụ như: <http://www.brunel.ac.uk/depts/cc/network/Linux/slip.html>. *slip* thực hiện cái gọi là “chat script” - script hội thoại, để thiết lập chế độ cho modem, quay số, và còn cho phép ghi lại file log vào ngân hàng modem của máy chủ thuộc địa chỉ trên. Sau đó, bạn sẽ chỉ ra kết nối được chấp nhận sẵn sàng cho nối kết bằng cách gõ tổ hợp phím **Ctrl-J**, chương trình *slip* sẽ bắt đầu thực hiện giao thức SLIP.

Script hội thoại có thể được thay đổi cho phù hợp với loại modem cụ thể mà bạn có trong hệ thống, nhưng của nó có nội dung cơ bản như sau:

```
#
slip.dip
#
setup line for SLIP
#
This version assumes a Penril Datalink 14.4k modem.
The config commands for other types will be different

print Clearing modem on port cua0
port cua0
speed 38400
reset
sleep 5
flush
```

---

```
print Configuring modem...

set low volume
send \ratl=0\r
wait OK 3
if $errlvl != 0 goto error

set hardware flow-control
send at\\q1,1\r
wait OK 3
if $errlvl != 0 goto error

print Dialling...

send atd048693350\r
wait CONNECT 30
if $errlvl != 0 goto error

sleep 1
send \r

print Da log in, nhưng KHONG go lenh slip voi!
print Khi thay dau nhac `port', hay go ^] (control-])

term

send slip\r
wait is 5
if $errlvl != 0 goto error
get $rmtip remote

wait is 2
if $errlvl != 0 goto error
```

```
get $locip remote
```

```
print Addresses $locip $rmtip
default
mode CSLIP
```

```
error:
```

```
print SLIP connection failed
```

Để thao tác dễ dàng hơn, bạn có thể đặt script hỏi thoại trên vào trong một shell script rồi đặt shell script đó vào một vị trí tiện lợi, ví dụ như `/usr/local/bin`. Shell script sau được đặt tên là `slip` và chứa script hỏi thoại được giả sử có tên là `/usr/local/etc/slip.dip`:

```
#!/bin/sh
#
slip - start or stop the SLIP connection

if test "$1" = start
then
 /sbin/dip /usr/local/etc/slip.dip
else
 if test "$1" = stop
 then
 /sbin/dip -k
 else
 echo 'Usage: slip start|stop' 1>&2
 exit 1
 fi
fi
```

Bây giờ ta thêm quyền thực thi cho shell script `slip` và có thể sử dụng cho dịch vụ SLIP như sau:

```
slip start
```

```
slip stop
```

## XV. 5. NFS

Network File System (NFS) là sản phẩm của Sun, cho phép chia sẻ tài nguyên trong toàn mạng. Khi kết hợp với lệnh **mount**, NFS cho phép người dùng sử dụng hệ thống file trên máy từ xa giống như trên máy của mình. NFS cho phép nhiều máy cùng lúc truy cập tới cùng một hay một số file, mọi máy trong mạng đều có thể truy cập được tới dữ liệu. Các chương trình lớn, tốn tiền cũng có thể được chia sẻ để dùng chung, giảm thiểu các chi phí cho dung lượng đĩa, ... Có thể quy định một cách tiếp cận thống nhất tới cơ sở dữ liệu, bảo đảm tính thống nhất, toàn vẹn và độ tin cậy của nó.

NFS được xây dựng theo mô hình Client/Server. Server là máy cho phép các máy khác mount vào ít nhất một hệ thống file của mình. Client là máy mount, tức là tham gia vào ít nhất một cấu trúc hệ thống file của máy khác. Mọi máy chủ Sun đều có thể vừa là server, vừa là client.

Trên máy PC Sun cung cấp một sản phẩm tương tự khác, được gọi là PC-NFS, dùng cho các máy PC trong mạng chạy ở chế độ MS-DOS và Windows 3.11, Windows 9x.

### *XV. 5. a. Dịch vụ NFS hoạt động như thế nào*

Trong hệ thống UNIX, dịch vụ NFS được xem xét dưới các góc độ máy chủ (NFS Server) và máy khách (NFS Client).

#### ▪ *Hoạt động của NFS Server*

Các trình thường trú (daemon) **rpc.mountd** và **nfsd** điều khiển hoạt động của dịch vụ NFS trong máy chủ cùng với chương trình hệ thống **share** (Solaris 2.x, 7, 8)

hay **exportfs** (SunOS 4.x). Khi khởi động NFS Server, file `/etc/rc2.d/K60nfs.server` sẽ làm chạy lệnh (dòng **shareall**), sau đó tìm đến file `/etc/dfs/sharetab`. Nội dung của file này được thông báo tới kernel của hệ điều hành. Server sẽ thông báo các file và thư mục mà nó cho các máy khác quyền truy xuất tới thông qua lệnh **share** hay **exportfs**.

#### ▪ Hoạt động của NFS Client

Các client truy xuất tới các thư mục và file được phép xuất ra trên máy chủ thông qua lệnh **mount -F nfs** hay qua cơ chế **automounter**.

Khi máy client mount thư mục của máy chủ, nó không sao chép thư mục ấy về mình. Ngược lại, tiến trình mount sử dụng các thủ tục gọi từ xa cho phép máy từ xa truy nhập vào thư mục trên đĩa cứng của máy chủ.

Máy client cũng có thể mount các thư mục của máy chủ ngay từ khi khởi động bằng **automounter**. Danh sách các thư mục và file được mount nằm trong file `/etc/ufstab`.

### XV. 5. b. Cài đặt và bảo trì NFS Server

Việc cài đặt NFS Server không phức tạp và có thể thực hiện bằng các bước sau. Trước hết, phải chọn máy xác định sẽ dùng làm máy chủ NFS. Sau đó, xác định các file và thư mục sẽ xuất ra cho các máy client. Bước này có thể thực hiện bằng lệnh **share**:

```
share [-F FSType] [-o specific_options]
 [-d description] [pathname]
```

với:

**F**

Loại hệ thống file, ở đây là **nfs**.

**o**

Tùy chọn:

```
ro[=client[:client]]...
```

Chỉ đọc [cho người dùng trong danh sách mà thôi]

```
rw[=client[:client]]...
```

Đọc và ghi được [cho người dùng trong danh sách mà thôi]  
anon=uid

Đặt SUID. Nếu là -1: cấm.

```
root=host[:host]...
```

Chỉ có các người dùng root ở các host mới có quyền root trên máy chủ NFS. Mặc định: không có người dùng nào.

hay bằng cách sửa trong file /etc/dfs/sharetab. Dòng vào trong /etc/dfs/sharetab có dạng sau:

```
directory -option[, option]
```

Nếu sửa file trên, nhất thiết ta phải khởi động lại máy để kernel được tự động cấu hình lại.

Việc bảo trì server NFS thực chất là thay đổi nội dung các dòng vào trong /etc/dfs/sharetab hay dùng lệnh để thay đổi danh sách các người dùng hoặc thư mục, file. Nếu cần thiết, có khi phải sửa trong cả các file /etc/passwd, /etc/group, /etc/shadow, /etc/hosts.

### ***XV. 5. c. Cài đặt và bảo trì NFS Client***

Để cài đặt NFS Client ta có ba cách làm:

- Sử dụng lệnh **mount**:

```
mount [-F nfs] [-o specific_options] hostname:pathname
/mountpoint
```

Khi cần xem trạng thái, dùng **share** không thông số, hay **showmount -e**.  
Khi bỏ dịch vụ NFS đi, dùng **unmount**.

- Sửa đổi file /etc/ufstab:

Các dòng vào của file /etc/ufstab có cú pháp:

```
filesystem directory type options freq pass
```



với:

**filesystem**

Có dạng `host:pathname`, với `host` - tên của server NFS, `pathname` - thư mục cần **mount**.

**directory**

Điểm mount trên máy client.

**type**

Ở đây là `nfs`.

**option**

`ro` - chỉ đọc; `rw` - đọc và ghi.

**freg**

Số ngày giữa hai lần lưu trữ thư mục được xuất ra trên máy chủ. Vì đây là máy trạm nên đặt giá trị 0.

**pass**

Số lần kiểm tra thư mục bằng **fsck**. Vì là máy trạm nên đặt bằng 0.

– Dùng lệnh **automounter**:

Ta phải sửa đổi các file map dùng cho **automounter**. Chỉ khi nào có lệnh truy xuất tới các thư mục và tập tin ở xa (trên NFS server) thì thư mục hay tập tin đó mới được mount trên máy trạm mà thôi. Về phương diện thực tế, ta sẽ được lợi về chi phí kết nối vì các kết nối, ví dụ, qua đường dial-up (quay số điện thoại), chỉ được thiết lập khi **automounter** được kích hoạt. Các file map đó là:

- `/etc/auto-master`: liệt kê các file map khác, các điểm mount, đường dẫn và các thông số khác.
- `/etc/auto-home`: tập tin thực sự chứa các điểm mount.

## XV. 6. Phân tích mạng

Nội dung được trình bày dưới đây sẽ giúp thêm cho bạn có kinh nghiệm theo dõi hoạt động của mạng và phân tích, tìm ra nguyên nhân gây ra những trục trặc trên mạng.

## XV. 6. a. Các lệnh có ích

### ▪ *Lệnh ping*

Lệnh **ping** sẽ gửi tín hiệu ICMP\_ECHO\_REQUEST tới máy đích. Nếu máy đó hoạt động và có thể nối kết được, tín hiệu ICMP\_ECHO\_RESPONSE được gửi trả về.

Cú pháp của lệnh có dạng:

```
[/usr/sbin/]ping [-s] hostname [timeout]
```

**hostname**: tên máy đích, có thể dùng địa chỉ IP hoặc bí danh của máy đích;  
**timeout**: khoảng thời gian cho phép để cố gắng kết nối với máy đích; **-s**: hiển thị thời gian chuyển gói đi về.

Nếu trong file `/.profile` bạn chưa đặt đường dẫn tới `/usr/sbin` thì bạn phải gõ đầy đủ `/usr/sbin/ping`. Trong các phần khác của tài liệu này, bạn còn gặp lại lệnh này nhiều lần nữa.

Trong trường hợp **ping** có kết quả, bạn sẽ nhận được thông báo:

```
hostname (hoặc địa chỉ IP) is alive.
```

hoặc một dãy các dòng hiển thị thời gian chuyển gói đi và về. Cần đặc biệt chú ý, lệnh **ping -s** liên tục chuyển và nhận các gói thông báo, nghĩa là liên tục giữ đường truyền. Mục đích dùng lệnh với tùy chọn này là kiểm tra số các gói thông tin được chuyển theo hai chiều có bị thất lạc không. Để ngắt lệnh này, cần gõ `^C`.

Trong trường hợp không kết nối được thông báo sẽ có nội dung:

```
hostname (hoặc địa chỉ IP) is unreachable.
```

Nguyên nhân có thể là máy đích không hoạt động, đường truyền (nhất là trong kết nối mạng WAN) trục trặc, hoặc có sự cố về mạng. Có thể còn có các dòng thông báo khác giúp tìm nguyên nhân dễ dàng hơn.

### ▪ *Lệnh ifconfig*

Lệnh hiển thị các thông tin về cấu hình của giao diện mạng (card mạng) đã chọn. Cú pháp:

```
[/usr/sbin/] ifconfig interface [address-family] \
 [address [dest_address]] [up] [down] [auto-revarp] \
 [netmask mask] [broadcast address] [metric n] \
 [mtu n] [trailers|-trailers] [private|-private] \
 [arp|-arp] [plumb] [unplumb]
```

interface là tên file thiết bị cho card mạng, ví dụ: card 3COM 3C5x9 trong Solaris for x86 là elxn, card 3COM 3C900 - elxln. Có ba tên đặc biệt: a - tất cả các card mạng trong hệ thống, ad - tất cả card đều ở trạng thái "down", au - tất cả, ở "up".

Các tùy chọn chủ yếu:

#### **[-] arp**

[ Không] Cho phép dùng giao thức ARP để ánh xạ các địa chỉ ở mức mạng và mức liên kết. Mặc định: giữa các địa chỉ TCP/IP và Ethernet 10 Mb/s.

#### **down/up**

Hệ thống không/có chuyển thông tin qua giao diện mạng chỉ định.

#### **broadcast address**

Xác định địa chỉ để gửi thông báo. Có tính đến thứ tự khi dùng cùng với **netmask**.

#### **ether address**

Đặt địa chỉ Ethernet (của card mạng)

#### **mtu n**

Đặt đơn vị truyền cực đại cho giao diện mạng.

#### **netmask mask**

Xác định số địa chỉ dành để chia mạng thành các mạng con.

Ví dụ, khi gõ lệnh # **ipconfig elx0**, bạn nhận được thông báo như:

```
elx0: flags=63<up, BROADCAST,NOTRAILERS, RUNNING>
 inet 129.144.50.28 netmask ffffffff00 broadcast
 129.144.50.0
```

### ▪ *Lệnh netstat*

Là một lệnh rất hữu ích, dùng để kiểm tra trạng thái của mạng. Thường hay được dùng với các tùy chọn a, s, r, i, I và rs.

#### **a**

Hiển thị tất cả các thông tin về mạng.

#### **s**

Hiển thị các thông tin theo từng giao thức.

#### **r**

Hiển thị bảng phân đường. Khi dùng chung với s sẽ cho thông tin thống kê phân đường.

#### **i**

Hiển thị thông tin về trạng thái của các card mạng.

#### **I interface**

Hiển thị thông tin về trạng thái của card mạng được chỉ định.

Tùy thuộc vào hệ điều hành UNIX cụ thể mà thông tin kết quả của lệnh **netstat** có thể được hiển thị hơi khác nhau chút ít, nhưng về đại thể ta vẫn có thể nắm bắt được chúng, nếu đã từng sử dụng lệnh này. Tiếp theo đây, do tầm quan trọng của các thông tin do lệnh **netstat** đưa lại, và để hiểu rõ hơn về cách dùng lệnh, ta hãy xét bốn ví dụ cụ thể sau với trường hợp SuSE Linux.

### *Hiển bảng phân đường*

Khi gọi lệnh **netstat** với tùy chọn **-r**, bảng phân đường của kernel hiện ra cho ta biết cách mà ta phải thực hiện khi làm việc với các đường truyền đó, như trong ví dụ:

```
netstat -nr
Kernel routing table
```

---



---

| Destination | Gateway    | Genmask         | Flags | Metric | Ref | Use | Iface |
|-------------|------------|-----------------|-------|--------|-----|-----|-------|
| 127.0.0.1   | *          | 255.255.255.255 | UH    | 1      | 0   | 50  | lo    |
| 191.72.1.0  | *          | 255.255.255.0   | U     | 1      | 0   | 478 | eth0  |
| 191.72.2.0  | 191.72.1.1 | 255.255.255.0   |       |        | UGN | 1   | 0     |
|             |            |                 |       |        | 250 |     | eth0  |

Tùy chọn **-n** bắt lệnh **netstat** hiện các địa chỉ dưới dạng địa chỉ IP bốn nhóm cách nhau bằng dấu chấm (xxx.xxx.xxx.xxx), chứ không hiện các tên mạng và tên hình thức của máy. Điều này đặc biệt tiện lợi khi bạn muốn tránh tìm địa chỉ rất lâu trên toàn bộ mạng, ví dụ như tới server DNS hay NIS).

Trường thứ hai trong kết quả ra của lệnh **netstat** cho thấy tên của gateway mà dòng vào phân đường chỉ đến. Nếu không dùng gateway, lệnh sẽ hiển thị một dấu sao "\*" trong trường đó. Cột thứ ba cho thấy "tính chất chung" của đường dẫn. Khi đang phân đường đi thích hợp cho một địa chỉ IP, kernel sẽ đi qua tất cả các dòng vào của bảng phân đường, lấy bit AND của địa chỉ và **genmask** trước khi so sánh chúng với đích của đường đi.

Trường thứ tư hiển thị các thông số khác nhau dùng để mô tả đường đi:

#### **G**

Đường đi sử dụng gateway.

#### **U**

Giao diện được đặt ở trạng thái up.

#### **H**

Chỉ có một máy có thể tìm được qua đường đi. Ví dụ, trong trường hợp dòng vào tự tìm lại 127.0.0.1.

#### **D**

Tham số này được đặt, nếu dòng vào của bảng được tạo bởi thông báo đổi hướng ICMP.

#### **M**

Tham số này được đặt, nếu dòng vào của bảng được thay đổi bởi thông báo đổi hướng ICMP.

Trường Ref của kết quả lệnh **netstat** cho biết số các tham chiếu đến đường, có nghĩa là số các đường khác (qua gateway chẳng hạn) dựa trên sự tồn tại của đường đã cho. Hai trường cuối cho biết số lần mà dòng vào cho đường đi cụ thể được sử dụng, và giao diện mạng được sử dụng.

### Hiển thị số liệu thống kê của giao diện mạng

Khi gọi lệnh với tùy chọn **-i**, **netstat** sẽ hiển thị các số liệu thống kê cho giao diện mạng hiện đang được cấu hình. Nếu tùy chọn **-a** còn được dùng thêm, thao tác trên được thực hiện với tất cả các giao diện mạng được quản lý trong kernel hệ thống:

```
$ netstat -i
Kernel Interface table
Iface MTU Met RX-OK RX-ERR RX-DRP RX-OVR TX-OK TX-ERR
 TX-DRP TX-OVR Flags
lo 0 0 3185 0 0 0 3185 0 0 0 BLRU
eth0 1500 0 972633 17 20 120 628711 217 0 0 BRU
```

Các trường MTU và Met cho thấy kích thước cực đại của đơn vị thông tin được truyền MTU hiện thời và giá trị *metric* đối với giao diện mạng.

Các cột RX và TX cho thấy có bao nhiêu gói thông tin được gửi hoặc truyền đi không bị lỗi (RX-OK/TX-OK), hoặc bị hỏng (RX-ERR/TX-ERR), bao nhiêu gói bị bó (RX-DRP/TX-DRP), và bao nhiêu gói bị mất do không kịp xử lý (RX-OVR/TX-OVR).

Cột cuối cùng cho biết thông số được đặt cho giao diện mạng dưới dạng một ký tự. Các thông số này được liệt kê dưới đây và có thể được xem ở dạng đầy đủ khi hiển thị cấu hình của giao diện mạng bằng lệnh **ifconfig**.

**B**

Đặt địa chỉ thông báo.

**M**

Nhận tất cả các gói thông tin.

**N**

Tránh việc lẫn theo vết.

O

Tất phân giải địa chỉ (ARP) đối với giao diện mạng này.

P

Chỉ ra đây là nối kết điểm-tới-điểm ppp.

R

Đang hoạt động.

U

Giao diện mạng được bật.

### Hiển thị nối kết

Lệnh **netstat** hỗ trợ một loạt các tùy chọn để hiển thị các *socket* tích cực và thụ động. Socket được hiểu là một nửa nối kết, là tập hợp *địa chỉ cổng truyền + tên máy + giao thức* hay *giao thức + tên máy + địa chỉ cổng nhận*. Các tùy chọn -t, -u, -w, và -x cho thấy lần lượt các nối kết socket TCP, UDP, RAW, hoặc UNIX tích cực. Nếu bạn còn thêm vào tùy chọn -a, các socket đang chờ được kết nối (*listening*) cũng được hiển thị. Bằng cách đó ta có thể liệt kê danh sách tất cả các server hiện đang hoạt động trong hệ thống.

```
$ netstat -ta
Active Internet connections
Proto Recv-Q Send-Q Local Address Foreign Address
(State)
tcp 0 0 *:domain *:* LISTEN
tcp 0 0 *:time *:* LISTEN
tcp 0 0 *:smtp *:* LISTEN
tcp 0 0 hatay:smtp hoabinh:1040 ESTABLISHED
tcp 0 0 *:telnet *:* LISTEN
tcp 0 0 localhost:1046 vbard-tw:telnet ESTABLISHED
tcp 0 0 *:chargen *:* LISTEN
tcp 0 0 *:daytime *:* LISTEN
tcp 0 0 *:discard *:* LISTEN
tcp 0 0 *:echo *:* LISTEN
```

```
tcp 0 0 *:shell *:* LISTEN
tcp 0 0 *:login *:* LISTEN
```

Ví dụ trên cho thấy hầu hết các server đang ở trong trạng thái chờ kết nối. Chỉ có ở dòng vào thứ tư cho biết có kết nối SMTP tới từ server hoabinh, và dòng thứ sáu cho thấy có kết nối **telnet** ra từ máy tại chỗ tới máy vbard-tw.

Việc dùng tùy chọn **-a** tự nó sẽ hiển thị tất cả các socket thuộc tất cả các loại.

#### ▪ *Kiểm tra bảng phân giải địa chỉ ARP*

Trong một số trường hợp việc xem, thậm chí cả sửa bảng ARP của kernel, tỏ ra hết sức có ích. ví dụ khi nghi ngờ có sự trùng địa chỉ Internet khiến xảy ra các vấn đề không nhất quán trong mạng. Lệnh **arp** được dùng cho các mục đích trên và có các cú pháp dòng lệnh như sau:

```
arp [-v] [-t hwtype] -a [hostname]
arp [-v] [-t hwtype] -shostname hwaddr
arp [-v] -d hostname [hostname : : :]
```

Tất cả các thông số của hostname có thể hoặc là tên host tượng trưng, hoặc là địa chỉ IP dạng xxx.xxx.xxx.xxx. Dòng lệnh đầu hiển thị dòng vào ARP cho địa chỉ IP hoặc tên host đã cho, hoặc tất cả các host được biết đến trong mạng nếu không có tên hostname nào được chỉ ra. Ví dụ, việc gọi lệnh **arp** trên máy hatay có thể cho:

```
arp -a
IP address HW type HW address
191.72.1.3 10Mbps Ethernet 00:00:C0:5A:42:C1
191.72.1.2 10Mbps Ethernet 00:00:C0:90:B3:42
191.72.2.4 10Mbps Ethernet 00:00:C0:04:69:AA
```

cho thấy địa chỉ Ethernet của hatay (00:00:C0:5A:42:C1), hoabinh (00:00:C0:90:B3:42) và thanhhoa ( 00:00:C0:04:69:AA) trong mạng.



Dùng tùy chọn `-t`, bạn có thể giới hạn chỉ hiển thị loại phần cứng được xác định trong số ether, ax25, orpronet, có nghĩa tương ứng là thiết bị Ethernet 10Mb/s, AMPR AX.25, và Token Ring theo chuẩn IEEE 802.5.

Tùy chọn `-s` được dùng để thêm địa chỉ Ethernet của `hostname` vào bảng ARP một cách cố định. Tham số `hwaddr` xác định địa chỉ phần cứng như là địa chỉ Ethernet gồm sáu byte hexa, cách nhau bởi dấu hai chấm `:`. Tuy nhiên, bạn có thể đặt địa chỉ phần cứng thành loại khác bằng cách dùng tùy chọn `-t`.

Việc gọi lệnh **arp** có dùng tùy chọn `-d` sẽ xóa hết mọi dòng vào của bảng ARP có liên quan tới máy chủ đã cho. Tùy chọn này có ích khi muốn bắt buộc cho giao diện mạng nhận địa chỉ Ethernet mới và thường được dùng để sửa lại hệ thống lúc trước bị cấu hình sai.

#### ▪ *Lệnh route*

Dùng để điều chỉnh bằng tay bảng phân đường. Cú pháp như sau:

```
route [-fn] add | delete [host | net] destination \
 [gateway [metric]]
```

Tùy chọn `f` - điền bảng phân đường cho tất cả các đầu vào của gateway; `n` - cấm in tượng trưng tên các máy và tên mạng.

### **XV. 6. b. Kiểm tra phần mềm**

Ngoài những lý do về phần cứng, những trục trặc mạng có thể do những nguyên nhân thuộc phần mềm. Bạn nên theo các bước kiểm tra sau đây:

- Kiểm tra lại nội dung `/etc/hosts` xem đã đúng và đủ chưa.
- Kiểm tra các địa chỉ Ethernet trong cơ sở dữ liệu `/etc/ethers`.
- Kiểm tra xem có đúng mỗi card mạng trong máy của bạn có file tương ứng `/etc/hostname.??#` và tên host có đúng không?
- Đối với mỗi card mạng cố gắng **telnet** (hay **rlogin**) vào máy từ xa mà bạn biết rõ và có đăng ký tài khoản người dùng. Rồi từ đó **telnet** (hay **rlogin**) ngược trở về máy của bạn. Thường ở bước này bạn đã xác định được lỗi trong hệ thống của bạn, nếu có.

- Kiểm tra xem daemon **inetd** có chạy không bằng cách gõ lệnh:

```
ps ax | grep inetd
```

- Kiểm tra file cấu hình của kernel xem đã có card mạng chưa.
- Dùng lệnh **netstat** để kiểm tra trạng thái mạng qua bảng phân đường.

## XV. 7. Dịch vụ thông tin mạng (NIS và NIS +)

### XV. 7. a. Khái niệm

*Network Information Services* (NIS) cho phép các máy mạng có giao diện chung, bất kể bạn đăng nhập vào hệ thống (**login**) từ máy nào. Nó cho phép người quản trị có thể phân phối cơ sở dữ liệu giữa các máy, cập nhật cơ sở dữ liệu tại trung tâm một cách tự động và đáng tin cậy.

NIS được cài cùng với hệ điều hành Solaris khi cài đặt hệ thống bằng lệnh **suninstall**, và về sau này là lệnh **/usr/etc/install/add\_services**, và được khởi động bởi lệnh **/usr/etc/ypserv** trên máy chủ và **/usr/etc/ypbind** trên các máy trạm. Một máy tính UNIX có thể vừa là máy chủ, vừa là máy trạm. Trong các hệ thống mạng thường có các máy chủ NIS chính (master) và phụ (slave hay còn gọi là *replica server*). Máy phụ có tác dụng nâng cao độ tin cậy và giảm tải cho máy chính. Trong Solaris 2.x (SunOS 5.x) trở đi, bản NIS có cải tiến được gọi là NIS+. NIS được khởi động bởi lệnh **nisinit**.

### XV. 7. b. Các thành phần của NIS+

#### ▪ *Domain*

Domain là một vùng, hay một nhóm truy xuất bởi một NIS map. Tên domainname phải được chỉ ra trong **/etc/rc.local**. Tên gồm những tên của từng cấp, còn được gọi là chu kỳ và được cách nhau bởi dấu “.”. Để biết tên máy NIS ta có thể dùng lệnh **ypwhich [hostname]**.

---

- *Map*

Là tên file lưu giữ thông tin. Map luôn được tạo trên máy chủ master. Đối với mỗi vùng (domain), file NIS map nằm trong thư mục `/var/yp/domainname`. Các NIS map cơ bản là:

|                                 |                               |                               |
|---------------------------------|-------------------------------|-------------------------------|
| <code>bootparams</code>         | <code>ether.byaddr</code>     | <code>group.byid</code>       |
| <code>group.byname</code>       | <code>hosts.byaddr</code>     | <code>hosts.byname</code>     |
| <code>netid.byname</code>       | <code>mail.aliases</code>     | <code>mail.byaddr</code>      |
| <code>netgroup.byhost</code>    | <code>netgroup.byuser</code>  | <code>netgroup</code>         |
| <code>netmask.byaddr</code>     | <code>network.byaddr</code>   | <code>network.byname</code>   |
| <code>passwd.byname</code>      | <code>passwd.byid</code>      | <code>protocols.byname</code> |
| <code>protocols.bynumber</code> | <code>publickey.byname</code> | <code>rpc.bynumber</code>     |
| <code>services.byname</code>    | <code>ypservers</code>        |                               |

- *Daemon*

### **ypserv**

Tiến trình của máy chủ, trả lời các yêu cầu từ phía các máy trạm.

### **ypbind**

Tìm kiếm và liên kết với **ypserv**.

### **ypxfrd**

Daemon chuyển thông tin map tốc độ cao

### **rpc.yupdated**

Server để thay đổi các dòng vào của map.

### **rpc.yppasswdd**

Server để thay đổi các file mật khẩu NIS.

### **in.named**

Daemon liên quan tới việc phân giải các địa chỉ.

- *Các liên kết*

### **ypcat**

Liệt kê dữ liệu trong NIS map.

**ypwich**

Liệt kê danh sách các NIS server và các map được phục vụ.

**ypmatch**

Số sánh khóa và giá trị của nó trong map.

**ypinit**

Xây dựng và cài đặt cơ sở dữ liệu NIS.

**ypoll**

Số thứ tự của map từ NIS server.

**ypset**

Gán một server làm NIS server.

**ypxfr**

Chuyển dữ liệu từ máy chủ NIS chính (master) sang máy phụ (slave)

**XV. 7. c. Các bước cài đặt NIS**

- *Cài đặt máy chủ - YP đầu tiên (root)*

- Chọn tên domain. Ví dụ: hut.edu.vn

```
domainname hut.edu.vn
domainname > /etc/defaultdomain
```

- Chọn phiên bản NIS+ :

```
cp /etc/nsswitch.nisplus /etc/nsswitch.conf
```

- Khởi động máy chủ NIS , với tùy chọn r - cho máy chủ root (đầu tiên trong mạng)

```
nisinit -r
```

- Bắt đầu daemon, r - cho máy root, S 0 - mức độ bảo mật 0 (không có hạn chế)

```
rpc.nisd -rS 0
```

- Thiết lập cấu trúc thư mục NIS+

```
/usr/lib/nis/nissetup hut.edu.vn
```

- Thêm dữ liệu từ `dulieu_map_file` vào bảng `dulieu_map`, tùy chọn a - chế độ thêm vào:

```
cat dulieu_map_file | nisaddent -a dulieu_map
```

- Kiểm tra lại các dòng vào:

```
niscat hosts.org_dir
```

#### ▪ *Thiết lập máy chủ của vùng con*

Ví dụ máy chủ của vùng con có tên là: `iep.hut.edu.vn`

- Trở thành máy trạm của vùng trên, trong dòng lệnh **nisinit** có tùy chọn c - client, H *hostname* - xác định những máy chủ "được tin cậy" - trusted server, ví dụ `tw.icbv.com.vn`:

```
domainname iep.hut.edu.vn
domainname > /etc/defaultdomain
cp /etc/nsswitch.nisplus /etc/nsswitch.conf
nisinit -c -H tw.icbv.com.vn
```

- Bắt đầu chạy daemon ở chế độ không bảo mật

```
rpc.nisd -S 0
```

- Tạo thư mục cho cơ sở dữ liệu. Tùy chọn m *hostname* - làm máy có tên trở thành máy chủ chính - master server. Trong ví dụ được trình bày ở đây, ta chọn luôn máy này làm master server:

```
nismkdir -m iep.hut.edu.vn iep.hut.edu.vn
```

- Đầu tiên, ta tìm tiến trình daemon NIS+ và PID của nó, xóa đi rồi khởi động lại:

```
ps -ef | grep rpc
kill <PID>
rpc.nisd -S 0
```

- Thiết lập bảng NIS+:

```
/usr/lib/nis/nissetup iep.hut.edu.vn
```

- Thêm dữ liệu từ *dulieu\_sub\_map\_file* vào bảng *dulieu\_sub\_map*, tùy chọn a - chế độ thêm vào:

```
cat dulieumap_sub_file | nisaddent -a dulieu_sub_map
```

#### ▪ *Thiết lập máy chủ phụ (replica server)*

Ví dụ, máy có tên *replica1.hut.edu.vn*:

- Trở thành máy trạm của vùng trên, trong dòng lệnh **nisinit** có tùy chọn c - client, H *hostname* - xác định những máy chủ trusted server, ví dụ *tw.icbv.com.vn*:

```
domainname iep.hut.edu.vn
domainname > /etc/defaultdomain
cp /etc/nsswitch.nisplus /etc/nsswitch.conf
nisinit -c -H tw.icbv.com.vn
```

- Bắt đầu chạy daemon ở chế độ không bảo mật

```
rpc.nisd -s 0
```

- Tạo thư mục cho cơ sở dữ liệu. Tùy chọn *s hostname* - làm máy có tên trở thành replica server. Trong ví dụ này, ta chọn máy làm replica server là máy `replica1.hut.edu.vn`:

```
nismkdir -s replica1.hut.edu.vn hut.edu.vn
```

- Sao lại vùng:

```
/usr/lib/nis/nisping hut.edu.vn
```

- *Thiết lập máy làm NIS+ của vùng con*

- Thiết lập vùng con:

```
domainname iep.hut.edu.vn
domainname > /etc/defaultdomain
```

- Chọn phiên bản NIS+:

```
cp /etc/nsswitch.nisplus /etc/nsswitch.conf
```

- Khởi động máy trạm:

```
nisinit -c -H tw.icbv.com.vn
```

## **XV. 8. Dịch vụ tên vùng (DNS)**

Dịch vụ tên vùng (Domain Name Services) có khởi nguồn gắn chặt chẽ với Internet. DNS khác với NIS / NIS+ ở *tính phân tán của cơ sở dữ liệu*. Nếu như cơ sở dữ liệu trong NIS hầu như tập trung tại máy chủ chính và được sao lại ở các máy replica, thì trong DNS mỗi máy chủ ở các cấp đều có thể có cơ sở dữ liệu của mình, có chủ quyền với chúng, và thường chịu trách nhiệm trong khu vực của mình. Mỗi khi xuất hiện một yêu cầu, bằng cơ chế phân giải theo tên, máy chủ ở cấp tương ứng

sẽ tìm thấy máy có thể đáp ứng được yêu cầu đó. Nếu không, nó tiếp tục gửi yêu cầu lên cấp vùng cao hơn. Tên được đặt theo quy tắc sau:

Máy . Bộ phận . Cơ quan . Cơ quan cấp cao nhất  
và địa chỉ IP có các địa chỉ mạng, mạng con tương ứng.

Cơ chế phân giải theo tên gồm hai thành phần chính: *name server* - chương trình lưu giữ tất cả các thông tin về vùng và trả lời các yêu cầu của người dùng; *resolver* - bộ phận giải, có thể là các chương trình độc lập như **dig**, **nslookup** hoặc được nhúng vào như **telnet**, **ftp**. Các thông tin được dùng chứa trong các bản ghi nguồn của các file cấu hình DNS. Daemon quan trọng nhất là **in.named**, gọi chạy script `/etc/init.d/inetsvc` có chức năng thiết lập hệ thống.

Mỗi vùng có một máy chủ chính (*primary master server*) quản lý tất cả các dữ liệu thuộc vùng, mà từ đó daemon DNS hoạt động. Nên có một máy dự bị khác để thay thế trong trường hợp có hỏng hóc, nếu bạn muốn cho vùng của bạn hoạt động liên tục và ổn định. Máy chủ chính thứ cấp (*secondary master server*) giữ bản sao dữ liệu của máy chủ sơ cấp và định kỳ soát lại để cập nhật.

#### XV. 8. a. Các bản ghi nguồn

- *Tổng cá phép tổng quát của các bản ghi*

```
{name} {ttl} class Record Type Record Specific data
```

với ý nghĩa của các trường như sau:

**name**

Tên vùng.

**ttl**

Viết tắt của “time to live” - thời gian (giây) lưu dữ liệu trong cơ sở dữ liệu.

**class**

IN (viết tắt của Internet)

**Record Type**

Một trong các loại SOA, NS, A, CNAME, HINFO, WKS, PTR, MX.

**Specific data**



Chứa dữ liệu tùy theo loại bản ghi

Các ký tự đặc biệt

- . Nếu đứng trong trường name chỉ vùng hiện thời
- @ Nếu đứng trong trường name chỉ nguồn hiện thời
- .. Nếu đứng trong trường name chỉ gốc vùng (root)
- \X Bỏ qua ký tự thay chỗ của X, nếu nó khác các số 0 - 9. Nếu thay chỗ X là DDD - các số bát phân.
- ( ) Nhóm các dữ liệu để chúng thuộc một dòng.
- ; Bắt đầu dòng chú thích
- \* Thay cho các ký tự bất kỳ

▪ *Bản ghi HINFO (HINFO - Bắt đầu xác lập chủ quyền)*

```
(name) (ttl) class SOA origin
person_in_charge (
 serial; Nếu máy thứ cấp thấy khác số, sẽ cập nhật lại cơ
 sở dữ liệu
 refresh ; Chu kỳ làm tươi (giây) của máy thứ cấp
 retry ; Khoảng thời gian (giây) lặp lại cố gắng làm tươi
 sau khi không thành công
 expire ;Giới hạn trên (giây)máy thứ cấp dùng dữ liệu
 minimum); Giá trị (giây) nếu chưa xác định ttl
```

▪ *Bản ghi NS (Name Server - Server phụ trách về các tên máy trong vùng)*

```
{name} {ttl} class NS Name-server name
Ví dụ: icbv.net IN NS tw.icbv.net
```

▪ *Bản ghi A (Address - Địa chỉ)*

```
{name} {ttl} class A address
```

```
{name} {ttl} class MG member name
```

### ***XV. 8. b. Các file cấu hình DNS***

#### **❖ /etc/named.boot**

Là file quan trọng nhất, điều khiển name server lấy ở đâu các dữ liệu ban đầu của nó để chạy, và được phân biệt cho primary và secondary server. Ví dụ, cho máy primary:

```
; File khai dong cho root server
;
; type domain source file or host
DIRECTORY /var/named
CACHE . cache
PRIMARY edu edu.zone
PRIMARY IN-ADDR.ARPA edu.rzone
PRIMARY 0.0.127. IN-ADDR.ARPA loop.back
```

#### **❖ root.cache**

Đưa ra quy tắc ban đầu để tìm máy chủ root ở gốc vùng. Ví dụ cho máy primary server của vùng .edu:

```
; file /var/named/cache cho domain server edu (.edu)
;
<nội dung của bản ghi NS>
<nội dung của bản ghi A>
```

#### **❖ /var/named/root.rzone**

Là file dữ liệu cấu hình địa chỉ ngược, dùng để ánh xạ địa chỉ Internet vào địa chỉ máy, tức là địa chỉ MAC của một trong các card mạng gắn trong máy. Dùng các bản ghi SOA, NS và PTR theo cú pháp:

```
$ORIGIN IN-ADDR.ARPA.
<nội dung của bản ghi SOA>
```

<nội dung của bản ghi NS>  
 <nội dung của bản ghi PTR>

#### ❖ /var/named/root.zone

Giữ thông tin về các máy trong vùng. Cú pháp:

```
$ORIGIN .
<nội dung của bản ghi SOA>
<nội dung của bản ghi NS>
<nội dung của bản ghi A >
```

#### ❖ loop.back

Để máy tự "tìm thấy" mình. Cú pháp:

```
bản ghi SOA giá trị trong trường name: 0.0.127.IN-ADDR.ARPA
bản ghi NS giá trị trường name: 0.0.127.IN-ADDR.ARPA
bản ghi PTR giá trị trường name: 1.0.0.127.IN-ADDR.ARPA,
 trong trường real name: localhost)
```

#### ❖ /etc/nsswitch.conf

Cú pháp:

```
passwd : files
group : files
hosts : dns files
```

#### ❖ /etc/resolv.conf

Là file dữ liệu DNS chính giúp máy client truy cập được cơ sở dữ liệu DNS. Cú pháp:

```
[DOMAIN domainname]; comment hay
[nameserver ipaddr]; comment
```

### ***XV. 8. c. Thiết lập DNS***

- Tạo và sửa các file cấu hình DNS cho từng loại máy tham gia DNS. Chọn loại tên vùng mặc định (ví dụ như .com)
- Đối với máy chủ cần gõ lệnh:

```
in.named
```

hoặc:

```
in.named -b /etc/myname
```

nếu file khởi động myname khác với mặc định.

- Đối với máy trạm chỉ cần sửa đổi các file `/etc/resolv.conf` và `/etc/nsswitch.conf`.
- Kiểm tra bằng lệnh **nslookup** với cú pháp sau:

```
nslookup [-opt] [host | -] [server]
```

## **XV. 9. Server Apache Web và server proxy**

Hơn một nửa số Web site hiện nay trên Internet sử dụng gói phần mềm miễn phí có tên là **Apache** như là server trang Web. Sau đây, ta sẽ nghiên cứu kỹ hơn việc sử dụng server Apache Web.

### ***XV. 9. a. Chạy server Web với người dùng bình thường***

Cách tốt nhất để tránh bị lộ trước những tin tặc đang rình cơ hội tấn công khi đang chạy server Web là tạo một nhóm và một người dùng bình thường duy nhất cho các ứng dụng server. Người dùng và nhóm *nobody* trong UNIX thường được sử dụng cho mục đích này, nhưng đứng dưới góc độ bảo mật, việc dùng một nhóm và người dùng bình thường duy nhất cho riêng server Web sẽ nâng cao hơn độ an toàn của mạng.

Theo mặc định, server Web sử dụng cổng **ưu tiên** (cổng 80), nhưng khi đã được cấu hình cho các thao tác được bảo mật, phải có quyền của `root` mới có thể mở

các file log của cổng đó và bắt đầu chạy daemon. Sau khi các hoạt động khởi động server đã hoàn thành, tất cả các lần xuất hiện tích cực của chương trình đều được chạy như do người dùng bình thường vận hành.

Bạn hãy dùng các dòng lệnh sau để tạo ra nhóm và người dùng httpd cho server Web . (Trong các bản Linux có cài đặt Apache ngay từ khi cài đặt hệ thống, thực ra nhóm và người dùng httpd đã được tạo ra một cách tự động). Hãy thay giá trị cụ thể của thư mục bạn muốn dùng như thư mục home của người dùng httpd vào vị trí của <server root> (trong Solaris 2.x, thường đó là một thư mục con dưới /opt, ví dụ như /opt/httpd).

```
groupadd httpd
useradd -d <server root> -g httpd -s /bin/false -c
 "Server Web " -m httpd
```

Để tin chắc người dùng này đã được tạo ra không thể được ai đó dùng để đăng nhập hệ thống, hãy kiểm tra dòng vào tương ứng trong file /etc/shadow. Trong trường mật khẩu phải chứa giá trị \*LK\* chỉ rằng người dùng này đã bị khóa và không thể được sử dụng để đăng nhập hệ thống.

```
httpd:*LK*:::::::::
```

### ***XV. 9. b. Cài đặt Server***

Với nhận xét trên về thư mục home của httpd ta thấy thư mục gốc của Server Web, thường được gọi là *ServerRoot*, theo thông lệ sẽ là một thư mục con của nơi cài đặt các ứng dụng có thể chia sẻ được của các hãng thứ ba - thư mục /opt. Nếu giữ đúng các quy tắc bảo mật, bạn sẽ phải giải nén và biên dịch phần mềm cho server Web trên các máy khác nhau. Khi giải nén file cài đặt, bạn sẽ nhìn thấy các thư mục sau:

```
cgi-bin
conf
htdocs
icons
```

logs  
**src**  
support

Nhằm đảm bảo cho server được an toàn hơn, người ta dùng các phần đĩa khác nhau cho nội dung của Web trong máy. Điểm mount duy nhất với tên là `htdocs` được tạo ra ngoài thư mục `ServerRoot` và ta phải cập nhật lại file `/etc/vfstab` để điểm mount đó có mặt trong thành phần hệ thống file khi server khởi động.

Không sử dụng thư mục `htdocs` có trong phần cài đặt như là thư mục `DocumentRoot` của bạn, bởi vì bạn sẽ không muốn đưa ra công khai trước tất cả mọi người các tài liệu trong `htdocs` có chứa các thông tin người dùng dễ bị kẻ xấu lợi dụng để tấn công hệ thống của bạn. Nên chuyển các file tài liệu đó về thư mục `support` để các người quản trị Web - *webmaster* - trên site của bạn có thể tham khảo khi cần.

Thư mục `src` (được in đậm để đánh dấu), không phải để cài đặt trên máy của bạn. Nội dung của nó chỉ được dùng khi bạn biên dịch các daemon thực thi mà thôi.

### ***XV. 9. c. Đặt quyền cho file và thư mục***

#### ***▪ Thông tin cơ bản***

Đối tượng xem xét của ta ở đây là trường hợp với hệ điều hành Solaris 2.x. Từ nay về sau, ta sẽ dùng `/opt/httpd` như là thư mục cơ sở của Web, tức `ServerRoot`. Thông thường, trong các hệ điều hành Linux, Apache được cài đặt tự động khi ta cài đặt hệ điều hành vào máy tính với tư cách là một tùy chọn mặc định. Đối với các hệ UNIX khác quy trình cài đặt Apache cũng tương tự như ví dụ ta sẽ khảo sát dưới đây.

Phần mềm của server Apache chứa các file được nhóm theo chức năng. Bảng sau cho thấy các thư mục con thường có trong thư mục `ServerRoot`:

| Các thư mục của ServerRoot |                                  |                                                      |
|----------------------------|----------------------------------|------------------------------------------------------|
| Tên                        | Chức năng                        | Nội dung                                             |
| conf                       | Cấu hình server                  | httpd.conf, srm.conf, access.conf                    |
| logs                       | File log cho server              | access_log, agent_log, error_log                     |
| cgi-bin                    | Các file thực thi trên cơ sở Web | Các script CGI                                       |
| icons                      | Các biểu tượng tĩnh              | Biểu tượng dùng để ký hiệu các thư mục được liệt kê. |
| support                    | Công cụ                          | Các tiện ích của Webmaster.                          |

Chú ý: Vì các file log của server tăng kích thước nhanh nên bạn có thể chuyển thư mục logs sang hệ thống file khác với dung lượng đủ lớn. Tất nhiên, việc chuyển vị trí của thư mục đó phải đi kèm với việc thay đổi cấu hình của server.

Để xác định các quyền ưu tiên đăng nhập server Web và các file liên quan, ta có thể chia các người dùng thành ba nhóm chức năng. Nếu tính đến việc ta muốn có một nhóm và người dùng bình thường riêng để chạy server thì đó sẽ là nhóm thứ tư. Ba nhóm chức năng được liệt kê trong bảng dưới đây:

| Vai trò của các nhóm chức năng |                                                        |
|--------------------------------|--------------------------------------------------------|
| Vai trò                        | Mô tả                                                  |
| Webmaster                      | Một nhóm nhỏ để bảo trì server.                        |
| Web developer                  | Những người viết chương trình (script CGI, Java, ...). |
| Web author                     | Những người viết các trang Web.                        |
| Web server                     | UID mà với nó server được chạy.                        |

Bảng liệt kê sau đây các chủ quyền cho từng thư mục, phân chia theo vai trò của từng nhóm người dùng trên, được Lincoln Steins, tác giả cuốn sách Web Security (An ninh mạng) khuyến cáo nên dùng. Chú ý rằng bạn có thể phải bổ sung thêm một vài



tên người dùng nào đó vào một loạt các nhóm để đảm bảo cho họ có đủ quyền truy nhập cần thiết.

| Các quyền được khuyến nghị để áp dụng |               |       |      |     |           |
|---------------------------------------|---------------|-------|------|-----|-----------|
| Nhóm                                  | Configuration | Tools | Logs | CGI | Documents |
| Webmaster                             | rwX           | r-X   | r-X  | rwX | rwX       |
| Web developer                         | -             | -     | -    | rwX | rwX       |
| Web author                            | -             | -     | -    | r-X | rwX       |
| Web server                            | -             | -     | -    | r-X | r-X       |

#### ▪ *Thực hiện các thay đổi*

Ta tiến hành các thay đổi theo từng bước sau:

- Tạo các nhóm của UNIX cho mỗi một chức năng được xác định ở trên:

```
groupadd webmastr
groupadd webauth
groupadd webdev
```

- Sửa file `/etc/group` để thêm danh sách người dùng vào các nhóm mới tạo trên. Thành viên của nhóm `webmastr` đồng thời phải là thành viên của các nhóm `webdev` và `webauth`, và một vài thành viên của nhóm `webdev` cần đồng thời là thành viên của nhóm `webauth`.
- Đảm bảo rằng `root` sở hữu tất cả các file và thư mục trong cây thư mục chứa tất cả các file liên quan tới `Web server`:

```
chown -R root /opt/httpd
```

- Thay đổi nhóm sở hữu đối với daemon của `Web server`:

```
chgrp bin /opt/httpd/httpd
```

- Đảm bảo chỉ root mới có quyền chạy daemon, tức là (---x-----):

```
chmod 100 /opt/httpd/httpd
```

- Thay đổi nhóm sở hữu đối với các thư mục con cụ thể (thay tên cụ thể của thư mục tài liệu người dùng vào chỗ của <Document Root>):

```
chgrp -R webmastr /opt/httpd/
chgrp -R webdev /opt/httpd/cgi-bin
chgrp -R webauth <Document Root>
```

- Thay đổi quyền đối với các file và thư mục cấu hình (-rw-rw---- và drwxrwx---):

```
chmod 770 /opt/httpd/conf
chmod 660 /opt/httpd/conf/*
```

- Thay đổi quyền đối với các file và thư mục công cụ (-rwxrwx--- và drwxrwx---):

```
chmod 770 /opt/httpd/support
chmod 770 /opt/httpd/support/*
```

- Thay đổi quyền đối với các file và thư mục ghi lại log (-rw-r----- và drwxr-x---):

```
chmod 750 /opt/httpd/logs
chmod 640 /opt/httpd/logs/*
```

- Thay đổi quyền đối với các file và thư mục cgi-bin (-rwxrwxr-x và drwxrws--x):

```
chmod 2771 /opt/httpd/cgi-bin
chmod 775 /opt/httpd/cgi-bin/*
```

Chú ý: Đối với các file trong /opt/httpd/cgi-bin, người ta còn khuyến nghị dùng quyền 775 đối với các script được diễn dịch và 771 đối với các script đã biên dịch.

- Thay đổi quyền đối với các file và thư mục htdocs (-rw-rw-r-- và drwxrws--x):

```
chmod 2771 <Document Root>
chmod -R 664 <Document Root>
```

- Thay đổi quyền đối với các file và thư mục biểu tượng (-rw-rw-r-- và drwxrwx--x):

```
chmod 750 /opt/httpd/icons
chmod 640 /opt/httpd/icons/*
```

Chú ý: Trong một số thư mục, Lincoln Stein, tác giả của cuốn sách *Web Security*, khuyên nên dùng chế độ 2xxx (tức là đặt bit quyền nhóm hiệu dụng SGID) để có thể dễ được chia sẻ hơn giữa các thành viên của các nhóm người dùng trong thành phần của Web server. Nếu áp dụng lời khuyên đó, ví dụ ở đây cho các thư mục cgi-bin và <Document Root>, ta phải thay đổi mặt nạ người dùng (bằng lệnh **umask**) cho trùng với nhóm sở hữu của thư mục một cách tương ứng.

#### ***XV. 9. d. Xóa các script CGI và các tài liệu HTML không cần thiết***

Chỉ nên cài đặt các script CGI nào mà bạn đã thử nghiệm và biết rằng chúng an toàn. Các script khác trong bộ cài đặt của Apache mà bạn không biết rõ hay không cần đến thì nên xóa đi. Nhưng ít nhất, trong số các script của Apache, script **pfh** cần được dùng để điều hòa giữa vài server Web.

Bạn chỉ có thể cho chạy tất cả các script nếu có được sự giám sát toàn diện đối với thư mục cgi-bin, và bạn cần không cho phép những người dùng bình thường có thể dùng được các shell script.

Nên xóa toàn bộ các file không cần thiết, được chỉ ra trong phần cấu hình cho file **srn.conf**, khỏi các thư mục dưới thư mục DocumentRoot.

### XV. 9. e. Cấu hình Server

Trước khi thực hiện cấu hình server, trong thư mục `/opt/httpd/conf` ta sao chép các file cấu hình mặc định như sau:

```
cd /opt/httpd/conf
cp httpd.conf-dist httpd.conf
cp srm.conf-dist srm.conf
cp access.conf-dist access.conf
```

Sau đó ta thay đổi lần lượt các file cấu hình sau đây để server có thể hoạt động tốt.

- `httpd.conf` (*Cấu hình cơ sở của Server*)

Trong bảng dưới đây liệt kê các chỉ dẫn và các giá trị được khuyên dùng trong file cấu hình cơ bản nhất khi dùng Apache. Các chỉ dẫn không được đề cập tới trong bảng vẫn giữ các giá trị mặc định.

| Các giá trị được khuyên dùng cho các chỉ dẫn trong file <code>httpd.conf</code> |                                                                                                                                                          |
|---------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| Chỉ dẫn                                                                         | Thiết lập giá trị                                                                                                                                        |
| <code>ServerType</code>                                                         | <code>standalone</code> (an toàn hơn việc khởi động với <code>inetd</code> ).                                                                            |
| <code>HostnameLookups</code>                                                    | <code>off</code> (DHCP, firewalls và proxy làm mất hiệu lực việc giải tên của client).                                                                   |
| <code>User</code>                                                               | <code>httpd</code>                                                                                                                                       |
| <code>Group</code>                                                              | <code>httpd</code>                                                                                                                                       |
| <code>ServerAdmin</code>                                                        | <code>root@&lt;hostname&gt;</code> (thay <code>hostname</code> bằng tên DNS cho máy của bạn).<br>Chú ý: không quan trọng lắm vì ta không dùng SMTP mail. |
| <code>ServerRoot</code>                                                         | <code>/opt/httpd</code>                                                                                                                                  |
| <code>ErrorLog</code>                                                           | <code>logs/error_log</code><br>(hoặc thay đổi đến thư mục có dung lượng tự do còn nhiều).                                                                |

|             |                                                                               |
|-------------|-------------------------------------------------------------------------------|
| TransferLog | logs/access_log<br>(hoặc thay đổi đến thư mục có dung lượng tự do còn nhiều). |
| ServerName  | Tên đầy đủ DNS cho server (ví dụ<br>www.<servername>.<domain>)                |

Những dòng vào sau đây có ảnh hưởng tới tải mà server Web đặt vào máy của bạn. Hai dòng vào `StartServers` và `MinSpareServers` có liên quan mật thiết với nhau - nếu bạn đổi dòng vào này, bạn sẽ phải đổi cả dòng lệnh kia. Giá trị mặc định của các thông số trên là 5. Tùy thuộc vào mức độ tải, bạn có thể điều chỉnh các giá trị sau theo chiều tăng hoặc giảm.

- o `StartServers`
- o `MinSpareServers`
- o `MaxSpareServers`

Sau đây là danh sách các dòng vào khác trong file cấu hình trên cũng có ảnh hưởng tới khả năng chịu tải của server. Bạn có thể yên tâm giữ nguyên các giá trị mặc định cho các thiết lập phụ trợ, nhưng trước hết nên tìm hiểu các dòng vào và ý nghĩa tác động của chúng lên server:

- `Timeout`
- `KeepAlive`
- `MaxKeepAliveRequests`
- `KeepAliveTimeout`
- `MaxClients`
- `MaxRequestsPerChild`
  - `srm.conf` (*Cấu hình thư mục và tài liệu*)

Các dòng vào, nếu không được chỉ ra trên bảng dưới đây, có thể giữ nguyên các giá trị mặc định.

| Giá trị được khuyên dùng cho các chỉ dẫn trong file <code>srm.conf</code> |                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Chỉ dẫn                                                                   | Thiết lập giá trị                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <code>DocumentRoot</code>                                                 | Thư mục chính cho nội dung Web - cần được để ở phần đĩa riêng                                                                                                                                                                                                                                                                                                                                                                                         |
| <code>UserDir</code>                                                      | Đặt dòng vào này thành chú thích. Dòng vào này cho phép dùng nội dung Web từ các thư mục khác ngoài thư mục <code>DocumentRoot</code> . Vì thế dòng vào này làm giảm khả năng ngăn chặn việc đăng nhập tới các hệ thống file khác trên server Web của bạn.                                                                                                                                                                                            |
| <code>DirectoryIndex</code>                                               | <code>index.html</code> (giá trị mặc định). Nếu một file được chỉ ra trong dòng vào này không tồn tại trong thư mục con của thư mục <code>Document Root</code> , server sẽ gửi đến trình duyệt danh sách của tất cả các file và thư mục trong <code>Document Root</code> . Client có thể truy nhập các file hoặc thư mục con bất kỳ, thậm chí khi không có siêu liên kết tới chúng từ các tài liệu khác. Cần xem thêm file <code>access.conf</code> . |
| <code>FancyIndexing</code>                                                | <code>off</code> -- Ngăn không cho server hiển thị biểu tượng làm lộ loại file khi server liệt kê danh sách các tên file trong thư mục không được xác định trong <code>DirectoryIndex</code> .                                                                                                                                                                                                                                                        |
| <code>Alias</code>                                                        | <code>/icons/</code> -- Biến thành chú thích, hoặc đặt <code>off</code> cho <code>FancyIndexing</code> cũng có tác dụng tương tự.                                                                                                                                                                                                                                                                                                                     |
| <code>ScriptAlias</code>                                                  | <code>/cgi-bin /opt/httpd/cgi-bin</code> . Chú ý: để sử dụng các script CGI, phải thay đổi cả ở trong file <code>access.conf</code> .                                                                                                                                                                                                                                                                                                                 |
| <code>AccessFileName</code>                                               | <code>.htaccess</code> -- được sử dụng với khả năng <i>Basic Security</i> của server. Có thể giữ giá trị mặc định, nhưng phải thận trọng với việc bảo vệ các trang Web và script trong các thư mục cần được bảo vệ.                                                                                                                                                                                                                                   |

Các dòng vào sau trong file cấu hình `srn.conf` cần được biến thành chú thích. Nếu muốn sử dụng dòng vào bất kỳ nào trong số này, bạn hãy tìm hiểu kỹ tác dụng của chúng.

- `AddHandler` (tất cả các loại)
- `MetaDir`
- `MetaSuffix`
- `access.conf` (*Cấu hình liên quan tới bảo mật truy nhập Web*)

File điều khiển vài yếu tố bảo mật quan trọng của server Web. Người quản trị mạng cần rất chú ý tới nội dung file `access.conf` để bảo đảm cho người dùng có thể truy cập được server Web, nhưng hệ thống vẫn giữ được độ an toàn cao. Nếu nhìn từ phía người sử dụng, quyền hạn thực tế của họ khi truy nhập vào server Web đang chạy bằng Apache do bạn quản trị, sẽ được quyết định bởi các giá trị thiết lập trong file này, ví dụ như được phép xem và tải về các file, được truy nhập tới các thư mục khác với thư mục được phép xem ban đầu, v.v..

Chú ý: Các chỉ dẫn trong file này được áp dụng tới các thư mục cụ thể. Ví dụ, nếu ta xác định thư mục `/htdocs` trong đoạn mã nguồn `<Directory ...> ... </Directory>`, ta có thể áp dụng giám sát truy nhập và các hạn chế khác tới toàn bộ cây thư mục đó bằng dòng vào có dạng như sau trong file cấu hình trên:

```
<Directory /htdocs>
 Options IncludesNoExec
 ...
 order allow,deny
 allow from all
 ...
</Directory>
```

| Giá trị được khuyên dùng cho các chỉ dẫn trong file <code>access.conf</code> |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Chỉ dẫn                                                                      | Thiết lập giá trị                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <pre>&lt;Directory /usr/local/etc /httpd/htdocs&gt;</pre>                    | Thay đổi đường dẫn cho phù hợp với dòng vào cho <code>DocumentRoot</code> được xác định trong file <code>srn.conf</code> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Options                                                                      | <p>Thay dòng mặc định, <code>IndexesFollowSymLinks</code>, bằng <code>IncludesNoExec</code>. Kết quả đó dẫn đến việc ngăn server của bạn không gửi danh sách các file trong thư mục nếu không có file nào có tên được xác định bởi <code>DirectoryIndex</code> trong file <code>srn.conf</code>. Nó cũng cấm dùng các liên kết hình thức là phương tiện cho phép truy nhập file ở ngoài thư mục <code>DocumentRoot</code>. Giá trị mới cho dùng <code>Server Side Includes (SSI)</code> nhưng ngăn chạy mã nguồn bằng SSI, cũng như hạn chế việc dùng các script CGI cho thư mục <code>ScriptAlias</code>.</p> <p><u>Chú ý:</u> Tùy chọn này được áp dụng cho toàn bộ server Web. Dùng <code>Options</code> trong phạm vi <code>&lt;Directory&gt;</code> và các khối <code>&lt;Limit&gt;</code> để giới hạn tác dụng cho các thư mục cụ thể.</p> |
| <pre>order allow,deny allow from all</pre>                                   | Dòng vào mặc định này cho phép truy nhập thế giới bên ngoài.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <pre>&lt;Directory /usr/local/etc/ httpd/cgi-bin&gt;</pre>                   | <p>Thay đổi đường dẫn cho phù hợp với dòng vào <code>ScriptAlias</code> trong file <code>srn.conf</code>.</p> <p><u>Chú ý:</u> Các script CGI trong thư mục <code>ScriptAlias</code> phải là thực thi. Nếu chúng không chạy được, bạn có thể phải thay đổi việc thiết lập <code>Options</code> cho thư mục này tới giá trị <code>ExecCGI</code>.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |



### ***XV. 9. f. Cấu hình việc kiểm toán server***

Theo mặc định, các file log được ghi vào thư mục `con logs` của thư mục `ServerRoot`. Bạn có thể thay đổi tên và vị trí của các file log bằng cách có các thay đổi cho các chỉ dẫn trong file cấu hình `httpd.conf`. Không tắt **việc ghi log**.

Bình thường, server tạo hai file log sau để ghi lại các diễn biến truy cập và các lỗi phát sinh:

- `access_log (TransferLog)`
- `error_log (ErrorLog)`

### ***XV. 9. g. Cấu hình giám sát đăng nhập, kiểm tra xác nhận người dùng***

Server Apache đưa ra nhiều cách hạn chế đăng nhập, từ việc cho phép kết nối từ các địa chỉ IP hay vùng cụ thể, đến việc kiểm tra chi tiết người dùng/mật khẩu cho từng file riêng lẻ. Chế độ giám sát đăng nhập đối với server được thiết lập trong file `access.conf`. Dòng vào `<Directory...>` chỉ đến toàn bộ thư mục `DocumentRoot` liên quan tới việc giám sát đăng nhập một cách căn bản. Trường `Options` theo mặc định cho phép server hiện các liệt kê mục lục và đi theo các liên kết hình thức để tới các trang Web. Người dùng được phép đăng nhập vào server Apache sẽ bị giám sát bởi hai trường `order` và `allow`. Theo giá trị mặc định thì tất cả các người dùng bất kỳ từ đâu có thể truy nhập bất cứ file nào. Để thay đổi quyền đăng nhập, ta phải thay đổi nội dung dòng vào của `allow` và thêm dòng vào cho `deny`, nếu cần.

Ví dụ, để cấp phép đăng nhập chỉ cho các client mà địa chỉ của chúng nằm trong không gian địa chỉ lớp B `162.45.xxx.xxx`, bạn phải thay đổi các thiết lập mặc định như sau:

```
order allow,deny
allow from all
```

thành:

```
order allow,deny
allow from 162.45
deny from all
```

Việc kiểm tra xác nhận người dùng và giám sát đăng nhập được tiến hành nhờ trường Options trong các chỉ dẫn <Directory> và <Limit>. Các chỉ dẫn này thường được xác định trong file `access.conf`. Việc sử dụng chúng còn có liên quan tới việc sử dụng các file `.htaccess` trong các thư mục con của thư mục chính DocumentRoot đối với server Apache, nhưng ta có thể thay đổi quan hệ đó bằng cách dùng chỉ dẫn AllowOverride.

#### *XV. 9. h. Các thao tác từ xa có hạn chế*

Quyền thêm file vào thư mục nhờ lệnh con **put** của **ftp** được kiểm tra bằng các chú quyền của file và thư mục. Cấu hình mặc định của Apache cho phép bất kỳ người dùng nào trong nhóm `webauth` đều có thể **ftp** các file đến cây thư mục DocumentRoot của bạn.

Lệnh **post** của HTTP được giám sát bởi chỉ dẫn Limit. Trước khi cấu hình server để cho phép các thao tác **post** của HTTP, bạn cần nghiên cứu công dụng của việc kiểm tra xác nhận người dùng và giám sát đăng nhập của Apache để hạn chế các hoạt động đang này đối với một số cá nhân không nằm trong diện ưu tiên sử dụng.

#### *XV. 9. i. Tạo băng chữ bảo mật*

Băng chữ bảo mật được dùng để cảnh báo các thao tác từ phía người truy nhập trang Web có nguy cơ xâm hại tới hệ thống, cũng như tới dữ liệu chứa trên đó. Người quản trị của Apache cũng có thể dùng băng chữ để truyền đạt các thông tin, khuyến cáo liên quan tới an ninh của mạng và server Web. Băng chữ này có thể được tạo ra bằng nhiều cách. Tất nhiên, ngôn ngữ hiển thị phổ biến là tiếng Anh. Khi các phông chữ Việt theo chuẩn Unicode chưa được dùng rộng rãi trong các hệ thống UNIX, ta đành chấp nhận viết chữ Việt không bỏ dấu như là một giải pháp tạm thời. Có thể dùng băng chữ dưới đây như một ví dụ:

#####

Hệ thống này chỉ được thiết lập cho những người dùng có thẩm quyền. Các thao tác thực hiện trên hệ thống này đều được theo dõi và ghi nhận lại. Nếu phát hiện thấy các hành động vượt qua thẩm quyền, các âm mưu xâm hại tới sự an toàn của hệ thống, hoặc các hoạt động có thể vi phạm pháp luật, người quản trị có quyền cung cấp chứng cứ cho các cơ quan pháp luật. Nếu bạn đã đọc những dòng này và tiếp tục thao tác trên hệ thống thì có nghĩa là bạn đã đồng ý tuân theo các quy định của chúng tôi.

#####

#### ***XV. 9.j. Lưu trữ thường kỳ và đặt lại log***

Apache thường giữ khoảng trống của các đoạn cuối các dòng vào mới nhất trong các file log, và cố gắng ghi các dòng vào tiếp theo vào chỗ của các khoảng trống đó. Vì vậy, để lưu trữ và làm tươi lại các log của Apache, ta phải chuyển nội dung của các file log đi, **rồi khởi động lại** Apache. Vì file `access.log` thường tăng thêm kích thước một khoảng cỡ 1 MB cho mỗi 10000 lần dùng (server Web có rất nhiều lượt được đăng nhập bởi các người dùng khác nhau), nên việc làm sạch này phải được thực hiện định kỳ.

Dưới đây là một shell script nhỏ mà bạn có thể chạy từ **cron** để di chuyển nội dung các file log, tạo các file log mới còn trống và lưu giữ các log cũ. Tiến trình này đòi hỏi có thư mục con `archive` trong thư mục `logs` của bạn. Theo shell script này, tốt nhất nên định kỳ hàng tuần cập nhật lại log. Tuy nhiên, nếu bạn muốn thực hiện các thao tác thống kê cho server Web của bạn, bạn có thể chạy script này ngay sau khi chương trình thống kê đã cập nhật số liệu.

```
#!/bin/sh
cd /opt/httpd/logs
mv *_log archive
touch access_log error_log
```

---

```
add other logs if you're using them
kill -1 `cat httpd.pid`
cd archive
DATE=`date -u +%d%m%Y`
tar -cvf httpd-logs.$DATE *_log
if [$? = 0]
then
 compress -f *.$DATE
 if [$? = 0]
 then
 rm *_log
 else
 echo "Compress failed"
 fi
else
 echo "Tar failed"
fi
```

### ***XV. 9. k. Khởi động và kết thúc Server***

Script sau được dùng để tự động hóa quá trình khởi động và ngừng server Web . Ta đặt tên cho script đó là httpd, và sao chép nó vào thư mục /etc/init.d, sau đó tạo các liên kết như được chỉ ra dưới đây để hoàn thành quá trình.

```
#!/bin/sh
#
/etc/init.d/httpd - Start/Stop httpd server
#
This is a startup file for the daemon at system init
time. By Solaris convention, it should be in
/etc/init.d with links to it from rc3.d and rc0.d.
#
```

```
killproc() {
kill the named process(es)
 pid=`cat /opt/httpd/logs/httpd.pid`
 ["$pid" != ""] && kill $pid
}

case $1 in 'start')
 if [-f /opt/httpd/httpd]; then
 /opt/httpd/httpd -d /opt/httpd
 fi

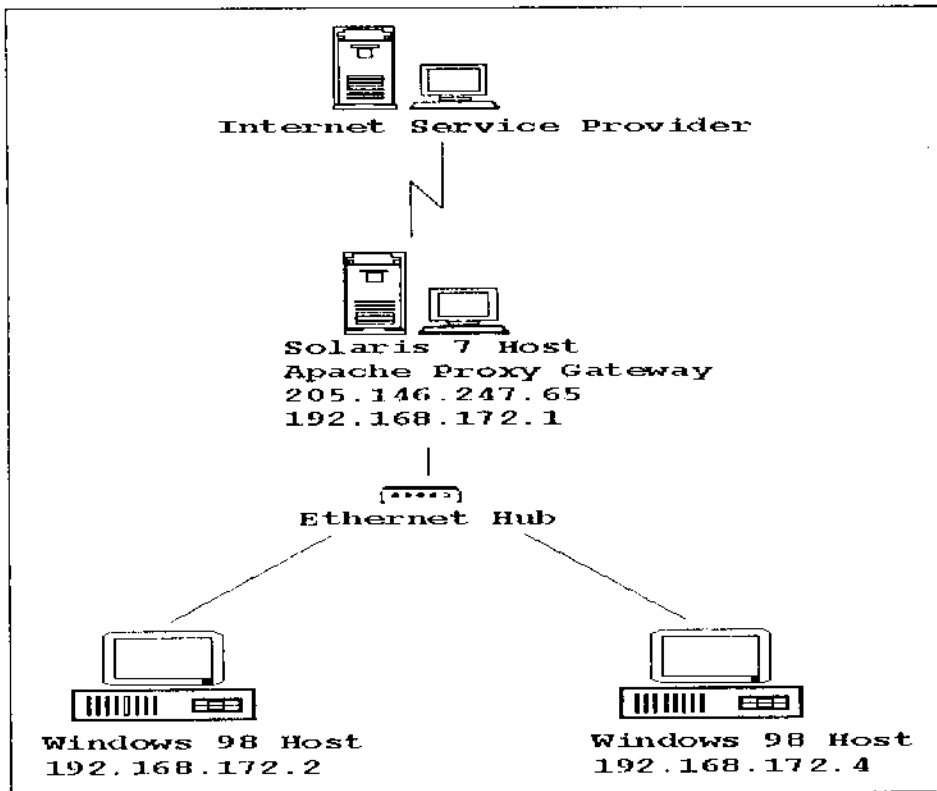
 ;;
 'stop')
 killproc httpd
 ;;
 *)
 echo "usage: /etc/init.d/httpd {start|stop}"
 ;;
esac
```

Tạo các liên kết hình thức sau cho script trên:

```
cd /etc/rc3.d
ln -s /etc/init.d/httpd S99httpd
cd /etc/rc0.d
ln -s /etc/init.d/httpd K99httpd
```

### ***XV. 9.1. Cấu hình proxy***

Trong phần này chúng ta sẽ tìm hiểu cách ứng dụng Apache như là server proxy cho ví dụ cụ thể một mạng có chứa hai máy client trên Windows 98 và một máy Solaris Apache proxy với các địa chỉ IP đã cho như ở hình H. XV. 6.



H. XV. 6. Mạng đơn giản có sử dụng proxy Apache

#### ▪ Chức năng của proxy Apache

Trước khi sử dụng proxy Apache, ta hãy tìm hiểu qua công dụng của nó. Trong kỹ thuật mạng nói chung, từ *proxy* mang nhiều ngữ nghĩa. Ở đây, ta coi proxy là tiến trình có thể gửi và nhận dữ liệu thông qua trình duyệt Web như Netscape hoặc Internet Explorer (IE). Apache chỉ thực hiện chức năng proxy cho các dịch vụ sau: **http**, **https**, **ftp**, **socks**, **gopher**, và **WAIS**. Đối với các dịch vụ như **telnet**, **DNS**, và **SMTP** ta phải dùng các phần mềm khác để thực hiện chức năng proxy.

Nhiều người sử dụng proxy Apache để cho phép một mạng riêng có thể chia sẻ một địa chỉ IP công cộng. Ta đã biết, trong môi trường tin học phổ thông, nhà cung cấp dịch vụ Internet (ISP) thường chỉ cung cấp cho ta một địa chỉ IP công cộng, có

nghĩa là trong mạng riêng tại chỗ có nối mạng Internet, chỉ có một máy tính (máy chủ) được gán địa chỉ IP công cộng đó, hay chỉ có người dùng trên máy đó mới có khả năng truy nhập Internet. Proxy Apache cho phép bạn chia sẻ một địa chỉ IP công cộng duy nhất bằng cách cung cấp đồng thời trang Web Internet cho tất cả các máy trong mạng riêng vừa nêu. Như vậy, nhiều người dùng đều có khả năng truy nhập Internet một lúc mà ta không phải đăng ký thêm tài khoản Internet với ISP.

Mặt khác, dù rằng có thể có nhiều tài khoản Internet được đăng ký trong một mạng riêng nào đó, người ta vẫn muốn bảo vệ mạng riêng khỏi nguy cơ phá hoại của tin tặc bằng cách cách ly mạng riêng khỏi Internet với sự giúp đỡ của proxy Apache. Điều này đặc biệt quan trọng đối với mạng có sử dụng các máy trạm làm việc trên Windows 9x, vì ta đã biết Windows 95 và 98 chẳng lấy gì làm đảm bảo tốt về phương diện bảo mật.

Giá thành thấp của phần mềm Apache, cộng với khả năng thích ứng rộng rãi của phần cứng đối với hệ điều hành đã được cải tiến của Sun đã mở ra một khả năng lớn cho các phần mềm của Sun trong thị trường các doanh nghiệp nhỏ là nơi rất nhạy cảm đối với các vấn đề giá thành.

#### ▪ *Mạng proxy*

Ta trở lại mô tả mạng đơn giản trên H. XV.6 có proxy Apache gồm hai máy client trên Windows 98 và một máy proxy Apache trên Solaris. Mạng của chúng ta được gán ba địa chỉ IP riêng (192.168.172.1, 192.168.172.2, và 192.168.172.4) cho các máy. Nếu không có proxy, các router Internet công cộng sẽ gửi các gói thông tin cho các địa chỉ IP trên, bắt đầu bằng địa chỉ mạng 192.168. Tuy nhiên, nếu dùng proxy, ta có thể tạo ra các đường truyền riêng trong nội bộ mạng của ta sao cho các gói thông tin từ ngoài gửi đến được đưa qua cùng các địa chỉ IP đến các máy dưới sự điều khiển trực tiếp của ta, tránh được việc có các đường truyền công cộng thẳng đến các máy Windows 98 vốn là điểm yếu dễ bị tin tặc tấn công, đồng thời, lại cho phép các gói thông tin từ những máy trạm đó qua địa chỉ IP công cộng có thể được gửi ra ngoài.

Máy chủ Solaris sử dụng một địa chỉ IP công cộng (205.146.247.65) và một địa chỉ IP riêng (192.168.172.1). Tất cả mọi nguy cơ trở thành đối tượng bị tấn công giờ

dây được máy chủ này gánh chịu và nhiệm vụ của người quản trị là tăng cường bảo mật cho máy chủ. Bằng cách đó, độ an toàn của mạng riêng được nâng cao.

- *Biên dịch và cài đặt mô đun proxy của Apache*

Trước khi sử dụng proxy server Apache bạn phải biên dịch mô đun proxy `mod_proxy`. Hầu hết các bản Apache nhị phân được biên dịch sẵn có trên Internet đều có cấu hình chuẩn, tắt đi chức năng biên dịch lại `mod_proxy`. Tốt hơn cả là bạn hãy tự tải về mã nguồn, cấu hình và biên dịch lại chương trình. Phiên bản cập nhật cuối cùng của Apache có thể tải về miễn phí từ [www.apache.org/dist/](http://www.apache.org/dist/). Trong phần này, ta sẽ xem xét cách biên dịch phiên bản 1.3.9. Để làm được việc này, ta cần có bộ biên dịch ANSI C từ trang Web của Sun hoặc GNU.

Sau khi đã giải nén mã nguồn (bằng **tar** hay **gzip** chẳng hạn) hãy chú ý đến script `INSTALL` ở trên đầu thư mục mã nguồn. `INSTALL` chứa phần hướng dẫn tóm tắt như sau:

1. Overview for the impatient

-----

```
$./configure --prefix=PREFIX
$ make
$ make install
$ PREFIX/bin/apachectl start
```

NOTE: PREFIX is not the string "PREFIX".  
Instead use the Unix filesystem path  
under which Apache should be  
installed. For instance use "/usr/  
local/apache" for PREFIX above.

Ta có thể theo chỉ dẫn trên để thực hiện việc cài đặt một cách chóng vánh và suôn sẻ. Việc cài đặt sẽ gây ra ít "rác" hơn cả đối với hệ thống file, nếu ta đặt tất cả



phần mềm này dưới thư mục PREFIX. Việc cài đặt cũng sẽ giữ nguyên tất cả các file cấu hình tồn tại dưới thư mục PREFIX/etc.

Có thể sử dụng lệnh cấu hình sau đây:

```
CC="cc" ./configure --prefix=/usr/local/apache \
--enable-module=proxy --enable-shared=proxy
```

(Dấu "\" trong câu lệnh trên có nghĩa là dòng lệnh được viết tiếp và không bị ngắt xuống dòng).

Trong câu lệnh cấu hình trên đã sử dụng phương án thông thường là định vị các file của Apache dưới thư mục /usr/local/apache. Nếu muốn, bạn có thể thay đổi vị trí của các file Apache trong phần câu lệnh

```
--prefix=/usr/local/apache
```

Bộ biên dịch được dùng trong ví dụ này là của Sun - **cc** là *c compiler*. Nếu bạn dùng bộ biên dịch của GNU (như trong trường hợp từ Solaris 2.6 trở đi mà không mua thêm C Workshop), ta bỏ **CC="cc"** khỏi phần đầu của câu lệnh và thay bằng **CC="gcc"**.

Sau khi cấu hình xong, ta gõ lệnh **make** tại dấu nhắc hệ thống. Tiếp theo, cần vào hệ thống bằng quyền **root** và gõ lệnh **make install** để cài đặt. Cuối cùng, ta gõ lệnh **/usr/local/apache/bin/apachectl start** để khởi động Apache. Bạn có thể dùng trình duyệt Netscape để kiểm tra việc cài đặt bằng cách mở tên máy được cài Apache như một URL. Netscape sẽ hiển thị trang Apache đầu với lời chào thay cho trang chủ ban đầu của server, nếu đã có trang đó từ trước.

Sau khi đã kiểm tra đúng là Apache đã được chạy như server Web riêng lẻ, bạn cần cấu hình mạng của bạn để dùng server Apache trên máy chủ của bạn như là proxy server.

Việc cấu hình proxy dẫn đến các thay đổi trong file cấu hình **httpd.conf** nằm trên máy chủ, trong thư mục /usr/local/apache/conf. Bạn còn cần điều chỉnh cấu hình proxy của Netscape trên mỗi máy client. Trong file cấu hình

---

httpd.conf bạn cần thay đổi nội dung hai dòng vào. Hãy dùng trình soạn thảo nào mà bạn quen thuộc và tìm dòng vào chứa:

```
LoadModule foo_module libexec/mod_foo.so
```

Dưới dòng đó bạn thêm một dòng vào mới để thêm thư viện cho proxy:

```
LoadModule proxy_module libexec/libproxy.so
```

Tiếp theo, bạn tìm và biến chúng thành chú thích đối với các dòng vào sau:

```
<IfModule mod_proxy.c>
ProxyRequests On
```

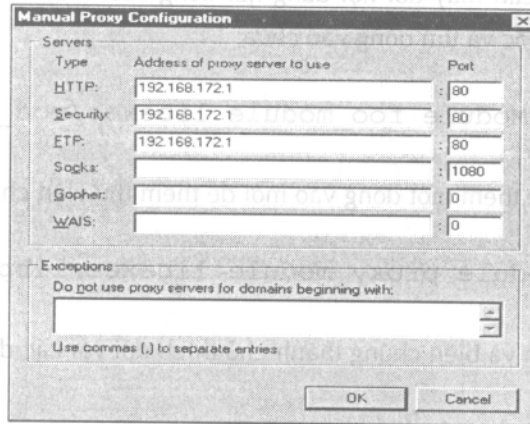
Cuối cùng, lưu lại file httpd.conf và bắt buộc Apache tiếp nhận cấu hình mới bằng cách dừng nó rồi khởi động lại thông qua các lệnh sau:

```
/usr/local/apache/bin/apachectl stop
/usr/local/apache/bin/apachectl start
```

Server Apache của bạn bây giờ đã có thể hoạt động như một proxy server.

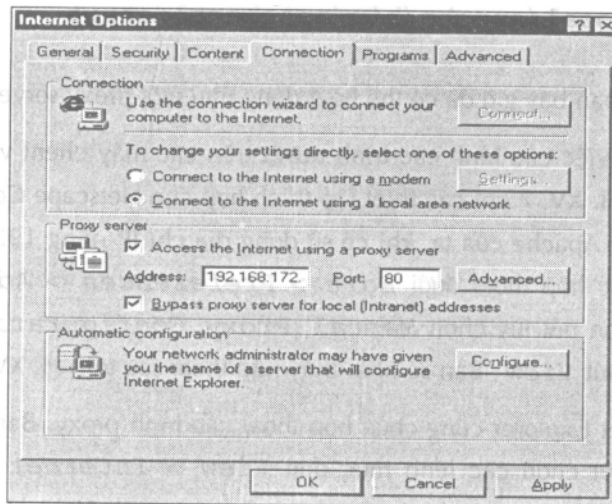
Bây giờ bạn chỉ việc cấu hình các trình duyệt trên các máy client và bắt đầu sử dụng proxy server. H. XV. 7 chỉ các thiết lập thích hợp cho Netscape Communicator đối với server proxy Apache của ta, khi có sử dụng địa chỉ IP riêng 192.168.172.1. Bạn sẽ tìm thấy các hộp thoại dưới *Edit -> Preferences -> Advanced -> Proxies*. Bạn chọn nút tùy chọn *Manual Proxy Configuration*, rồi sau đó nhấp chuột vào nút *View*. Bạn sẽ nhìn thấy hộp thoại sau trên H. XV. 7.

Microsoft Internet Explorer cũng chứa hộp thoại cấu hình proxy. Bạn sẽ vào hộp thoại đó khi lần lượt chọn các lệnh thực đơn *View -> Internet Options*. Nhấp chuột vào bảng *Connection*, hộp thoại sẽ hiện ra như trên H. XV. 8, dưới đây. Ta lại dùng địa chỉ 192.168.172.1 làm địa chỉ cho proxy server.



H. XV. 7. Hộp thoại *Manual Proxy Configuration* của Netscape

Sau khi đã đưa các thông tin về proxy server vào các hộp thoại tương ứng, nhấn phím **OK** để cập nhật. Từ thời điểm này bạn có thể tận dụng các tài nguyên Internet nhờ có Apache hoạt động như proxy server.



H. XV. 8. Hộp thoại *Proxy server* của Microsoft Internet Explorer

## XVI. Server PC Samba

### XVI. 1. Tìm hiểu về Samba

#### XVI. 1. a. Tại sao lại là Samba

Andrew Tridgell, người lãnh đạo một đội ngũ những nhà phát triển tại Canberra, Australia, là tác giả đầu tiên của Samba. Dự án này được hình thành từ 1991, khi Andrew Tridgell tạo ra một chương trình `fileserver` cho mạng LAN của mình nhằm hỗ trợ một giao thức của Digital Pathworks. Giao thức này về sau mang tên *Server Message Block* - SMB (giao thức khối thông báo của server), và Andrew Tridgell vài năm sau cũng viết lại sản phẩm của mình, tung ra thị trường Internet dưới tên *SMB Server*. Nhưng tên gọi này đã được dùng cho sản phẩm khác rồi, nên Andrew đã chọn tên khác bằng cách dùng lệnh UNIX để nhặt từ trong từ điển một từ có chứa ba chữ cái "S", "M", "B" như sau:

```
grep -i 's.*m.*b' /usr/dict/words
```

Và máy đã trả lời:

```
salmonberry samba sawtimber scramble
```

Thế là tên gọi "Samba" (tên một điệu nhảy Mỹ la <sup>gặt</sup>gặt) được chọn và dưới cái tên đó, một sản phẩm miễn phí vô cùng giá trị đã ra đời.

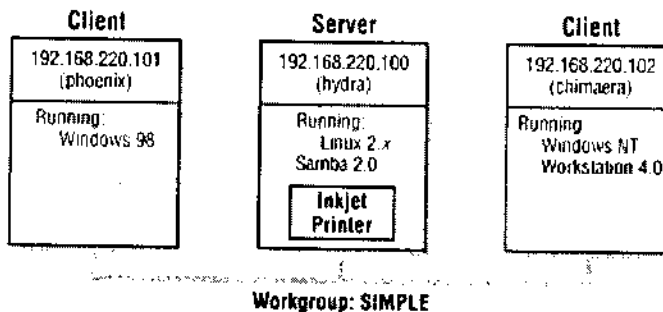
Vậy Samba giúp được gì cho ta? Samba là một sản phẩm UNIX làm việc với giao thức SMB. Nhiều hệ điều hành, trong đó có Windows và OS/2, sử dụng SMB để xây dựng mạng client-server. Bằng cách hỗ trợ giao thức này, Samba cho phép các server UNIX giao tiếp trên cùng một giao thức mạng với các sản phẩm Microsoft Windows, thực hiện các dịch vụ sau:

- Chia sẻ một hay nhiều hệ thống file.

- Chia sẻ các máy in được cài đặt ở cả hai phía server và client của nó.
- Hỗ trợ các client duyệt Network Neighborhood (*Mạng lân cận* của Microsoft).
- Kiểm tra xác nhận các client đăng nhập vào vùng của Windows.
- Cung cấp hoặc hỗ trợ việc phân giải địa chỉ của server bằng WINS (Windows Internet Name Service - Dịch vụ tên Internet của Windows).

Hiện nay Samba được duy trì và mở rộng nhờ một nhóm những người tình nguyện, có sự giám sát tích cực của Andrew Tridgell. Giống như hệ điều hành Linux, Samba gia nhập tổ chức mã nguồn mở OSS (Open Source Software) và được phát hành dưới bản quyền công cộng GPL (General Public License) của GNU. Ngoài ra, việc phát triển Samba còn được trường Đại học Quốc gia Australia, nơi Andrew Tridgell đã làm nghiên cứu sinh, tài trợ một phần. Các tổ chức tin học thương mại như Whistle và SGI (nhà phát hành bán IRIX UNIX) cũng tham gia tài trợ cho dự án Samba.

Để dễ hình dung hơn, ta lấy một ví dụ mạng đơn giản có dùng Samba. Giả sử ta có một cấu hình mạng cơ bản sau: một máy chủ UNIX có sử dụng Samba với tên được đặt là hydra, và hai client trên MS Windows, với các tên là phoenix và chimera, được nối với nhau trong một mạng LAN và tham gia vào nhóm làm việc có tên SIMPLE. Ta còn giả sử máy chủ hydra còn có một máy in phun tên là lp được nối tại chỗ, đồng thời việc chia sẻ đĩa cứng cho phép các máy "nhìn" thấy nhau. Mô tả mạng trên bằng hình vẽ được biểu diễn trên H. XVI. 1.



H. XVI. 1. Thiết lập mạng đơn giản với server Samba

Do mục đích chính dùng Samba là để làm cho máy (và mạng) UNIX có thể đăng nhập và thao tác như là file server đối với các máy Windows nên ta cần biết các vai trò Samba có thể đóng trong vùng của Windows NT Domain hay của nhóm làm việc Windows, cho dù nhiều giao thức mạng tên vùng của Windows NT là riêng biệt và tài liệu mô tả không được phát hành công khai. Bảng sau liệt kê các chức năng của Samba từ phiên bản 2.0.x trở đi đối với mạng Windows NT. Với sự phát triển của Samba, ta có thể chờ đợi Samba sẽ phủ kín phổ chức năng đã có trong mạng Windows NT.

**Các vai trò của Samba đối với mạng Windows NT (ví dụ cho phiên bản Samba 2.0.4b.**

Vai trò	Có thể thực hiện được ?
File Server	Có
Printer Server	Có
Primary Domain Controller	Có (Samba 2.1 hay muộn hơn)
Backup Domain Controller	Không
Windows 95/98 Authentication	Có
Local Master Browser	Có
Local Backup Browser	Không
Domain Master Browser	Có
Primary WINS Server	Có
Secondary WINS Server	Không

***XVI. 1. b. Các thành phần của Samba***

Samba thực chất chứa vài chương trình phục vụ cho những mục đích khác nhau nhưng có liên quan với nhau. Hạt nhân của Samba là hai daemon có những nhiệm vụ sau:

**smbd** Daemon **smbd** chịu trách nhiệm điều khiển các tài nguyên được chia sẻ giữa server Samba và các client của nó. Nó cung cấp các dịch vụ về file, in, và trình duyệt cho các client SMB thông qua một hay nhiều mạng. **smbd** xử lý tất cả các trao đổi giữa server Samba và

các client mạng của nó. Ngoài ra, daemon này còn chịu trách nhiệm kiểm tra xác nhận người dùng, khóa tài nguyên, và chia sẻ dữ liệu thông qua giao thức SMB.

**nmbd** Daemon **nmbd** là một server dịch vụ tên đơn giản bắt chước các chức năng máy chủ dịch vụ tên, chạy với các giao thức WINS và NetBIOS. Daemon này “lắng nghe” các yêu cầu của máy chủ dịch vụ tên và cung cấp các thông tin thích hợp khi được gọi tới. Nó còn cung cấp danh sách duyệt Network Neighborhood và tham gia vào lựa chọn các đối tượng mạng trong đó.

Bộ cài đặt Samba còn có một tập hợp nhỏ các công cụ dòng lệnh UNIX:

<b>smbclient</b>	Một client UNIX theo kiểu <b>ftp</b> có thể dùng tiện ích này để kết nối với tài nguyên được Samba chia sẻ.
<b>smbtar</b>	Chương trình để lưu trữ các tài nguyên được chia sẻ, tương tự như lệnh <b>tar</b> của UNIX.
<b>nmblookup</b>	Chương trình cung cấp NetBIOS thông qua việc tìm tên bằng TCP/IP.
<b>smbpasswd</b>	Chương trình cho phép người quản trị thay đổi mật khẩu đã mã hóa của Samba.
<b>smbstatus</b>	Chương trình báo cáo các nối kết mạng hiện thời tới các tài nguyên được chia sẻ trên Samba server.
<b>testparm</b>	Chương trình đơn giản để làm cho file cấu hình Samba có hiệu lực.
<b>testprns</b>	Chương trình kiểm tra liệu các máy in khác nhau có được daemon <b>smbd</b> nhận ra hay không.

Bản Samba ổn định mới nhất ở thời điểm cuối năm 2001 là 2.0.7, nhưng khi mô tả hoạt động của Samba, nhiều khi ta tham chiếu đến bản Samba 2.0.

Nếu nhìn từ phía UNIX, ta thấy Samba là một tập hợp các chương trình daemon. Bạn có thể xem chúng bằng các lệnh UNIX **ps** và **netstat**, có thể đọc mọi thông báo mà chúng sinh ra qua các file gỡ rối tùy biến hay qua **syslog** của UNIX, tùy theo cách thiết lập Samba, và cuối cùng - có thể cấu hình nó từ một file đặc tính duy

nhất của Samba là `smb.conf`. Thêm nữa, nếu bạn muốn xem từng daemon thực hiện những gì, Samba có chương trình với tên **smbstatus** sẽ đưa tất cả các thông tin đó lên màn hình như sau:

```
smbstatus
Samba version 2.0.4

Service uid gid pid machine

network ltuan users 7470 phoenix
 (192.168.220.101) Sun May 16
network ltuan users 7589 chimaera
 (192.168.220.102) Sun May 16
Locked files:

Pid DenyMode R/W Oplock Name

7589 DENY_NONE RDONLY EXCLUSIVE+BATC
 /home/samba/quicken/inet/common/system/help.bmp
 Sun May 16 21:23:40 1999
7470 DENY_WRITE RDONLY NONE
 /home/samba/word/office/findfast.exe Sun May 16
 20:51:08 1999
7589 DENY_WRITE RDONLY EXCLUSIVE+BATC
 /home/samba/quicken/lfbmp70n.dll Sun May 16
 21:23:39 1999
7589 DENY_WRITE RDWR EXCLUSIVE+BATC
 /home/samba/quicken/inet/qdata/runtime.dat Sun
 May 16 21:23:41 1999
7470 DENY_WRITE RDONLY EXCLUSIVE+BATC
 /home/samba/word/office/osa.exe Sun May 16
 20:51:09 1999
7589 DENY_WRITE RDONLY NONE
 /home/samba/quicken/qversion.dll Sun May 16
 21:20:33 1999
```



---

```
7470 DENY_WRITE RDONLY NONE
 /home/samba/quicken/qversion.dll Sun May 16
 20:51:11 1999
```

Share mode memory usage (bytes):

```
1043432(99%) free + 4312(0%) used + 832(0%) overhead
= 1048576(100%) total
```

Thông tin về trạng thái làm việc của Samba trong kết quả trên được chia thành ba tập hợp dữ liệu phân cách với nhau. Phần thứ nhất cho biết những hệ thống nào được nối kết với Samba server, xác định mỗi client bằng tên máy của nó (trong ví dụ này là phoenix và chimaera) và địa chỉ IP. Phần thứ hai báo cho ta biết tên và trạng thái của các file đang được dùng trên tài nguyên chia sẻ của server, bao gồm cả trạng thái đọc/ghi và mọi khóa đối với các file. Cuối cùng, Samba báo tổng số bộ nhớ hiện đang được huy động, bao gồm tổng số được tài nguyên chia sẻ dùng một cách tích cực và dung lượng bộ nhớ phụ trội. (Nhớ rằng tổng số bộ nhớ ở đây không giống như tổng số bộ nhớ mà các tiến trình **smbd** hoặc **nmbd** sử dụng.)

## XVI. 2. Mạng SMB/CIFS

Microsoft cũng mang nhiều công sức đóng góp vào việc xác định định nghĩa của SMB và Common Internet File System (CIFS - Hệ thống file Internet chung). Giao thức CIFS chính là tên mà Microsoft muốn đổi sang từ phiên bản tương lai của giao thức SMB, phiên bản sẽ được sử dụng trong các sản phẩm Windows, nên vì vậy ta coi hai tên gọi giao thức này là như nhau – giao thức SMB/CIFS.

Mạng SMB/CIFS khác biệt đáng kể so với nguyên tắc làm việc của mạng UNIX TCP/IP mà ta đã có dịp đề cập đến trong phần XV. 1. Vì vậy, để có thể nắm vững và sử dụng Samba - một sản phẩm UNIX có tác dụng kết nối với các mạng Windows - ta cần tìm hiểu kỹ giao thức SMB/CIFS và nguyên tắc hoạt động của mạng trên cơ sở giao thức đó.

## ***XVI. 2. a. NetBIOS và NetBUI***

### ***▪ Tìm hiểu về NetBIOS***

Ta hãy trở lại khoảng thời gian năm 1984, khi hãng IBM đưa ra một giao diện lập trình ứng dụng (Application Programming Interface - API) để tạo mạng cho các máy tính của họ. API đó có tên là Hệ cơ sở vào/ra mạng (Network Basic Input/Output System - NetBIOS) và cung cấp thiết kế bước đầu cho ứng dụng nối các máy tính và chia sẻ dữ liệu giữa chúng.

NetBIOS API được coi như phần mở rộng mạng của các lệnh gọi chuẩn BIOS API. Với BIOS, mỗi lệnh gọi mức thấp được xác định cho phần cứng của máy tại chỗ và không cần tới sự trợ giúp nào để tới được đích của nó. Với NetBIOS có thể thực hiện được việc trao đổi các lệnh, tuy nhiên, ban đầu chỉ giữa các máy tính qua các mạng IBM PC hoặc Token Ring. Vì vậy, cần phải có một giao thức vận chuyển mức thấp để chuyển các yêu cầu từ một máy tính tới máy tính khác.

Vào cuối năm 1985, hãng IBM đã đưa ra một giao thức như vậy, có kết hợp cả NetBIOS API và trở thành NetBIOS Extended User Interface - NetBEUI (Giao diện người dùng mở rộng NetBIOS). NetBEUI được thiết kế cho mạng LAN nhỏ (vào thời gian đó mạng có giới hạn không quá 255 nút), và cho phép mỗi máy tính có một tên dài cho đến 15 ký tự, quá đủ để dùng trong mạng.

Giao thức NetBEUI rất phổ biến trong các ứng dụng mạng, nhất là khi chạy với nhóm làm việc của Windows (Windows for Workgroups 3.11). Sau này, các cải tiến của NetBIOS đã được đưa vào giao thức mạng IPX của hãng Novell, làm cho IPX trở thành đối thủ cạnh tranh của NetBUI. Tuy nhiên, các giao thức mạng được chọn cho sự phát triển như vũ bão của Internet lại là TCP/IP và UDP/IP, nên sự cải tiến NetBIOS API nhanh chóng trở thành cấp thiết.

Nên nhớ lại rằng TCP/IP sử dụng các số để biểu diễn địa chỉ của máy tính, như 192.168.220.100 chẳng hạn, trong khi đó NetBIOS lại chỉ dùng các tên gọi. Đó là điểm khác biệt cơ bản giữa hai loại giao thức và người ta phải tìm cách kết hợp cả hai giao thức đó lại với nhau. Năm 1987, tổ chức IETF công bố một loạt các tài liệu về chuẩn hóa có tên là RFC 1001 và 1002, vạch ra cách thức mà NetBIOS có thể hoạt động qua mạng TCP/UDP. Các tài liệu ấy cho đến nay vẫn là tư tưởng chủ đạo

xuyên suốt các cải tiến giao thức cho đến nay, kể cả những gì mà Microsoft cung cấp cho các hệ điều hành Windows của họ, cũng như của Samba mà ta đang xét tới. Chuẩn mới được gọi là NetBIOS qua TCP/IP, hay gọi tắt là NBT. Chuẩn NBT (RFC 1001/1002) hiện nay là bộ ba, bao gồm dịch vụ mạng:

Name service

với hai dịch vụ truyền thông:

Datagrams

Sessions

Dịch vụ tên giải quyết vấn đề tên - địa chỉ mà ta đã nói đến ở trên, cho phép mỗi một máy tính có thể có tên trong mạng có thể chuyển sang địa chỉ IP có liên quan tới địa chỉ phần cứng, giống như DNS hiện nay trên Internet. Các dịch vụ datagram và session đều là các giao thức truyền thông thứ cấp được dùng để truyền dữ liệu đi và tới giữa các máy NetBIOS qua mạng.

#### ▪ *. Nhận tên máy*

Trên quan điểm NetBIOS, khi mỗi một máy trở thành trực tuyến trên mạng, máy đó cần tự gán tên cho mình - quá trình đó gọi là đăng ký tên. Tuy nhiên, hai máy trong cùng một nhóm làm việc không thể có cùng một tên để tránh sự bất định trong việc truyền dữ liệu. Có hai cách tiếp cận để loại trừ khả năng trùng tên như sau:

- Dùng *NetBIOS Name Server* - NBNS (máy chủ dịch vụ tên NetBIOS) để giữ đường đi của những máy đã đăng ký tên NetBIOS.
- Cho phép mỗi máy trong mạng tự bảo vệ tên của mình khi máy khác định dùng tên đó.

Hình H. XVI. 2 minh họa một cuộc đăng ký tên máy không thành công, trong các trường hợp có và không có NetBIOS Name Server.

Thêm nữa, cần phải có biện pháp phân giải một tên NetBIOS thành một địa chỉ IP cụ thể; biện pháp đó được gọi là *phan giải tên*. Có hai cách tiếp cận với NBT được biểu diễn trên hình H. XVI. 3:

- Bắt buộc mỗi máy báo lại địa chỉ IP của mình mỗi khi nó “nghe thấy” yêu cầu thông báo đối với tên NetBIOS.

– Dùng NBNS để giúp phân giải các tên NetBIOS thành các địa chỉ IP.

**Name Registration without a NetBIOS Name Server**

*I wish to register hydra...*  
(broadcast)



*No, I've already registered it.*

**Name Registration with a NetBIOS Name Server**

*I wish to register hydra...*



*No, someone has already registered it.*

**H. XVI. 2. Đăng ký tên không NBNS và có NBNS.**

**Name Resolution without a NetBIOS Name Server**

*Who is registered as phoenix?*  
(broadcast)



*I am. My address is 192.168.220.102*

**Name Resolution with a NetBIOS Name Server**

*Who is registered as phoenix?*



*That name currently belongs to 192.168.220.102.*

**H. XVI. 3. Phân giải tên không NBNS và có NBNS.**

Trong phương pháp không NBNS, khi máy client khởi động, nó sẽ thông báo muốn đăng ký cho mình một tên NetBIOS cụ thể. Nếu không có đối tượng nào trong

mạng sử dụng tên đó, sau nhiều lần đăng ký, máy client trên sẽ giữ được tên. Ngược lại, nếu đã có một máy nào đó trong mạng con đã dùng tên được chọn, nó gửi một thông báo ngược lại cho máy client để báo rằng tên đã được sử dụng. Cơ chế này được gọi là *sự bảo vệ tên*.

Với NBNS, tình huống tương tự cũng diễn ra, chỉ trừ một điều rằng giao tiếp truyền thông chỉ xảy ra giữa máy client có yêu cầu với server NBNS, mà không có thông báo từ máy client đó tới toàn bộ các máy trong mạng. Quá trình này được gọi là *truyền thông điểm-tới-điểm*, và thường được diễn ra trong mạng gồm nhiều hơn một mạng con.

Nguyên tắc tương tự cũng được áp dụng cho việc phân giải tên. Nếu không có NBNS, phân giải tên NetBIOS sẽ được tiến hành theo cơ chế thông báo rộng rãi. Tất cả các gói yêu cầu sẽ được gửi tới từng máy tính với hy vọng một máy nào đó bị ảnh hưởng sẽ trả lời trực tiếp cho máy yêu cầu. Từ đây ta cũng thấy rõ, nếu dùng NBNS, hay nói cách khác, truyền thông điểm-tới-điểm, sẽ tiết kiệm khả năng truyền dữ liệu hơn nhiều so với cơ chế thông báo rộng rãi.

#### ▪ *Các loại nút mạng*

Mỗi một máy trong mạng NBT sẽ có một trong số các giá trị nút mạng được gán như được trình bày trên bảng dưới đây.

Vai trò	Giá trị
b-node	Dùng đăng ký và phân giải theo cơ chế thông báo rộng rãi.
p-node	Dùng đăng ký và phân giải theo cơ chế điểm-tới-điểm.
m-node	Dùng thông báo rộng rãi để đăng ký. Nếu được sẽ báo cho NBNS server. Dùng thông báo rộng rãi để phân giải; dùng NBNS server nếu thông báo rộng rãi không có kết quả.
h-node (lai ghép)	Dùng NBNS server để đăng ký và phân giải; dùng thông báo rộng rãi, nếu NBNS server không trả lời hay không hoạt động.

Trong trường hợp các client Windows, bạn thường thấy chúng được liệt kê như là các nút lai (h-node). h-node được Microsoft đưa ra sau này như là đường truyền dữ liệu có khả năng chịu lỗi, chứ không phải đã có từ trước trong RFC 1001/1002.

Bạn có thể tìm ra loại nút mạng của bất cứ máy Windows nào bằng cách gõ lệnh **ipconfig /all** và tìm dòng vào nói về Node Type.

```
C:\>ipconfig /all Windows 98 IP Configuration
...
Node Type : Hybrid
...
```

#### ▪ “Tên” gồm những gì?

Các tên NetBIOS được dùng hoàn toàn khác với các tên máy DNS hostname mà ta đã xét đến trong phần TCP/IP (XV.1) hay phần về DNS (XV. 8). Trước hết, các tên NetBIOS tồn tại trong một không gian tên phẳng, không có thứ bậc, và mỗi một tên duy nhất cụ thể đại diện cho một máy. Thứ hai, các tên NetBIOS chỉ cho phép có đến 15 ký tự, không được bắt đầu bằng dấu sao “\*”, và chỉ gồm các ký tự chữ cái, chữ số chuẩn (a-z, A-Z, 0-9) và các ký tự sau:

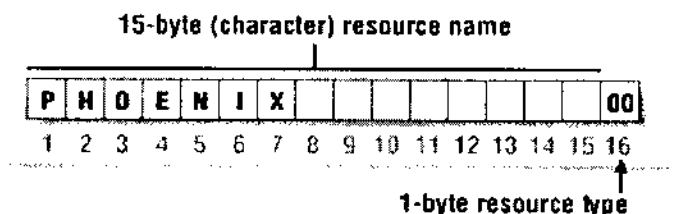
! @ # \$ % ^ & ( ) - ' { } . ~

Mặc dù được phép dùng dấu chấm “.” trong tên NetBIOS, nhưng tốt hơn cả là ta nên tránh vì dễ khỏi nhầm lẫn với kiểu cú pháp tên vùng, những tên NetBIOS như vậy không được đảm bảo hoạt động tốt trong các phiên bản NetBIOS về sau dùng thông qua TCP/IP.

Mọi tên vùng DNS hợp lệ không có nghĩa cũng là tên NetBIOS hợp lệ. Thực vậy, tên DNS cho server Samba thường được dùng lại như tên NetBIOS của nó. Ví dụ, nếu bạn có trong ví dụ xuyên suốt phần này một máy có tên là phoenix.ora.com, tên NetBIOS của nó sẽ là PHOENIX (và kèm theo 8 khoảng trống đi sau).

### Tài nguyên về tên và loại

Với NetBIOS máy tính không những chỉ báo về sự tồn tại của mình trong mạng, mà còn cho các máy khác biết những dịch vụ mà nó có thể đưa ra. Ví dụ, phoenix có thể chỉ ra nó không chỉ là trạm làm việc, mà còn là file server và có thể nhận các thông báo *WinPopup*. Điều đó được thực hiện bằng cách thêm byte thứ 16 vào cuối tên máy, hay tên tài nguyên, và được gọi là *loại tài nguyên* và bằng cách đăng ký tên nhiều lần. Cấu trúc tên NetBIOS được chỉ ra cho máy phoenix như trên H. XVI. 4.



H. XVI. 4. Cấu trúc của tên NetBIOS

Loại tài nguyên một byte chỉ ra loại dịch vụ duy nhất mà máy có tên được đặt có thể đề nghị. Ta sẽ dùng ký hiệu đôi dấu ngoặc nhọn "<>" đứng ngay sau tên NetBIOS của máy để chỉ loại tài nguyên, ví dụ như: PHOENIX<00>.

Ta có thể biết các tên đã được đăng ký cho một máy NBT cụ thể nào đó bằng tiện ích dòng lệnh **NBTSTAT**. Vì các dịch vụ là duy nhất, ta sẽ nhìn thấy dịch vụ đó được ghi trong kết quả ra là **UNIQUE**. Ví dụ, một phần kết quả ra cho máy chủ hydra có dạng:

```
D:\>NBTSTAT -a hydra
```

```
NetBIOS Remote Machine Name Table
```

Name	Type	Status
HYDRA	<00> UNIQUE	Registered
HYDRA	<03> UNIQUE	Registered
HYDRA	<20> UNIQUE	Registered
...		

Kết quả ra trên được giải nghĩa rằng máy chủ đăng ký tên NetBIOS hydra như là tên máy của trạm làm việc, là nơi nhận các thông báo WinPopup, và là file server. Một vài thuộc tính của tên NetBIOS được liệt kê trong bảng dưới đây.

### Các loại tài nguyên duy nhất NetBIOS

Tài nguyên được đặt tên	Giá trị byte hexa
Standard Workstation Service	00
Messenger Service (WinPopup)	03
RAS Server Service	06
Domain Master Browser Service (liên quan tới primary domain controller)	1B
Master Browser name	1D
NetDDE Service	1F
Fileserver (bao gồm cả printer server)	20
RAS Client Service	21
Network Monitor Agent	BE
Network Monitor Utility	BF

Chú ý rằng vì tên DNS không có loại tài nguyên, các nhà thiết kế đã đặt mặc định giá trị hexa 20 (khoảng trống ASCII) cho file server.

### Tên nhóm và loại

SMB cũng sử dụng ý tưởng về các nhóm mà các máy tính có thể tự đăng ký. Trong các khái niệm của Windows, nhóm làm việc của Windows và nhóm SMB thực chất là như nhau.

Tiếp tục xem xét ví dụ của ta về NBTSTAT, server Samba hydra đồng thời cũng là thành viên của nhóm làm việc SIMPLE (thuộc tính GROUP hexa 00), và được chọn như *browse master* - máy chính để duyệt (thuộc tính GROUP 1E ). Phần còn lại trong kết quả ra của tiện ích NBTSTAT như sau:

NetBIOS Remote Machine Name Table, continued



Name	Type	Status
-----		
SIMPLE	<00> GROUP	Registered
SIMPLE	<1E> GROUP	Registered
.._ _MSBROWSE_ _.	<01> GROUP	Registered

Các thuộc tính mà nhóm có thể có được liệt kê trong bảng sau:

### Các loại tài nguyên nhóm NetBIOS

Tài nguyên được đặt tên	Giá trị byte hexa
Standard Workstation group	00
Logon Server	1C
Master Browser name	1D
Normal Group name (dùng khi chọn bộ duyệt)	1E
Internet Group name (quản trị)	20
<01><02> MSBROWSE <02>	01

Dòng vào cuối, \_ \_ MSBROWSE \_ \_ , được dùng để công bố nhóm tới các máy duyệt bình khác. Các ký tự không in ra trong kết quả ra của NBTSTAT sẽ được biểu diễn bằng các dấu chấm.

#### ▪ *Datagrams và Sessions*

NBT có một trách nhiệm khác là cung cấp các dịch vụ kết nối giữa hai máy NetBIOS. Thực ra đó là hai dịch vụ được NetBIOS qua TCP/IP đưa ra: dịch vụ phiên làm việc (*session*) và dịch vụ datagram. Tuy việc tìm hiểu kỹ các dịch vụ này không liên quan nhiều lắm tới hoạt động của Samba, ta cũng dừng lại một chút để xem xét vì sẽ có ích cho việc tìm lỗi của Samba.

Dịch vụ datagram không có nối kết ổn định giữa các máy tính. Các gói dữ liệu chỉ được gửi hoặc phân phối rộng rãi một cách đơn giản từ máy tính này đến máy tính kia, không đếm xỉa đến việc chúng đến đích theo thứ tự nào, hay thậm chí cùng đến một lúc. Việc dùng datagram không gây ra cường độ luân chuyển dữ liệu cao trong mạng như dịch vụ *session*, mặc dù đôi lúc cũng có thể gây ra sự nghẽn

mạch. Vì vậy, datagram được dùng để gửi nhanh các khối dữ liệu đơn giản tới một hay nhiều máy tính. Dịch vụ datagram dùng các lệnh cơ bản như được chỉ trong bảng sau:

#### Các lệnh cơ bản về datagram

Lệnh	Mô tả
<b>Send Datagram</b>	Gửi các gói datagram tới máy tính hoặc nhóm các máy tính.
<b>Send Broadcast Datagram</b>	Phân phát rộng rãi datagram cho bất kỳ máy nào đang chờ Receive Broadcast Datagram.
<b>Receive Datagram</b>	Nhận datagram từ một máy tính.
<b>Receive Broadcast Datagram</b>	Chờ một datagram được phân phát rộng rãi.

Dịch vụ phiên làm việc hoạt động phức tạp hơn. Phiên làm việc là một phương thức truyền thông, về mặt lý thuyết, đưa ra khả năng phát hiện các nối kết có vấn đề hay không hoạt động giữa hai ứng dụng NetBIOS. Một kết nối song công đầy đủ được mở giữa máy tính gọi và máy tính được gọi và nó phải được giữ ở trạng thái mở trong suốt thời gian trao đổi của các máy tính trên. Mỗi một phía của nối kết nhận biết được đâu là máy gọi, đâu là máy được gọi và có thể trao đổi với nhau bằng các lệnh đơn giản được kê trong bảng sau:

#### Các lệnh cơ bản cho phiên làm việc

Lệnh	Mô tả
<b>Call</b>	Bắt đầu một phiên làm việc với một máy tính có tên cụ thể đang chờ trao đổi.
<b>Listen</b>	Chờ lệnh gọi từ máy gọi đã biết hay từ máy gọi bất kỳ.
<b>Hang-up</b>	Ngừng gọi.
<b>Send</b>	Gửi dữ liệu đến máy khác.
<b>Receive</b>	Nhận dữ liệu từ máy khác.
<b>Session status</b>	Lấy thông tin về trạng thái của phiên làm việc được yêu cầu.

Các phiên làm việc được coi như trục chính của các tài nguyên đang được chia sẻ trong mạng NBT. Chúng thường được sử dụng để tạo ra các nối kết ổn định từ các máy client tới đĩa cứng hay máy in được chia sẻ trong mạng. Máy client "gọi" máy server và bắt đầu các thông tin trao đổi kiểu như những file nào nó muốn mở, các dữ liệu gì muốn trao đổi, v.v... Những lệnh gọi trên có thể kéo dài, thậm chí hàng giờ, hàng ngày trong phạm vi một nối kết. Nếu xuất hiện lỗi, phần mềm phiên làm việc (với TCP) sẽ truyền lại cho đến bao giờ dữ liệu được nhận đúng mới thôi, chứ không như kiểu "đem con bỏ chợ" của dịch vụ datagram (với UDP). Mặt khác, nếu vì một lý do nào đó mà phiên làm việc bị đứt quãng, tất cả các thông tin dở dang đều bị mất hiệu lực, và ta chỉ còn cách duy nhất là để chính hai máy tính đó bắt đầu lại nối kết, nói cách khác, thiết lập lại phiên làm việc.

Tóm lại, ta cần nhớ so sánh giữa hai dịch vụ truyền thông như sau:

- ♣ Các phiên làm việc bao giờ cũng hoạt động giữa **hai** máy NetBIOS - không nhiều hơn mà cũng không ít hơn. Nếu dịch vụ phiên làm việc bị ngắt, máy client cần lưu giữ đủ các thông tin trạng thái để mở lại nối kết. Tuy nhiên, trong thực tế điều này rất hiếm khi xảy ra.

- ♣ Datagram có thể được phân phát rộng rãi cho nhiều máy tính, nhưng chúng không tin cậy. Nói cách khác, nguồn cung cấp không có cách nào để biết các dữ liệu được gửi đi có đến đích hay không.

### ***XVI. 2. b. Dịch vụ tên Internet của Windows (WINS)***

Dịch vụ tên Internet của Windows – *Windows Internet Name Service* (WINS) là cải tiến NetBIOS name server (NBNS) của Microsoft. WINS thừa kế nhiều đặc trưng của NetBIOS. Trước hết, WINS là phẳng. Thêm nữa, WINS là động: khi một máy client lần đầu vào mạng trực tuyến, nó được đòi hỏi phải thông báo tên máy, địa chỉ và tên nhóm làm việc của nó cho server WINS tại chỗ. Server WINS này sẽ giữ các thông tin đó cho đến khi máy client định kỳ làm tươi lại đăng ký WINS của nó để khẳng định nó vẫn được nối mạng. Chú ý rằng server WINS không phải máy chủ vùng hay nhóm làm việc đặc biệt nào mà có thể xuất hiện bất cứ ở đâu, phục vụ bất cứ máy nào. Nhiều server WINS có thể được đồng bộ hóa để phát triển phạm vi hiệu lực của dịch vụ WINS.

Trong họ các hệ điều hành Windows, chỉ có NT Workstation hoặc server NT mới có thể hoạt động như server WINS. Samba hiện nay có thể hoạt động như server WINS sơ cấp, nhưng chưa thể như server WINS thứ cấp.

### ***XVI. 2. c. Giao thức SMB***

SMB/CIFS là giao thức mà các máy Windows 95/98 và NT sử dụng để trao đổi với server Samba và với nhau. Ở mức cao, giao thức SMB là tương đối đơn giản, gồm các lệnh cho tất cả các thao tác với file và thao tác in, giống như khi bạn làm việc với đĩa cứng hoặc máy in tại chỗ, như:

- Mở và đóng file.
- Tạo và xóa file, thư mục.
- Đọc và ghi file.
- Tìm file.
- Xếp các file vào hàng chờ in và dỡ chúng khỏi hàng chờ in.

Mỗi một trong số các thao tác trên đều có thể được mã hóa vào trong các thông báo SMB và được truyền từ server hoặc đến server.

#### **▪ *Dạng của SMB***

SMB được định nghĩa như là giao thức "hỏi-trả lời", nghĩa là máy client gửi một yêu cầu SMB tới server, và server gửi ngược lại trả lời SMB cho máy client. Hiếm khi một server gửi đi một thông báo mà không phải là câu trả lời cho máy client.

Một thông báo SMB không đến nỗi quá phức tạp. Ta hãy xem xét cấu trúc bên trong của thông báo SMB. Có thể chia thông báo SMB thành hai phần: phần đầu có kích thước cố định và phần sau là dòng lệnh có kích thước thay đổi tùy thuộc vào nội dung của thông báo.

#### **Dạng của phần đầu SMB**

Trong bảng ở dưới đây là dạng của phần đầu SMB. Các lệnh của SMB không nhất thiết đòi hỏi phải sử dụng hết tất cả các trường trong phần đầu của SMB. Ví dụ, khi máy client lần đầu tiên cố gắng kết nối với server, nó không có giá trị *định danh cây* (Tree Identifier - TID) – giá trị này chỉ được gán sau khi đã kết nối thành công với server – nên giá trị null TID (0xFFFF) được đặt vào trong trường tương ứng trong

phần đầu SMB của máy client. Các trường khác cũng được điền bằng số 0 nếu không được dùng tới.

**Các trường trong phần đầu SMB**

Trường	Kích thước (bytes)	Mô tả
0xFF 'SMB'	1	Định danh giao thức.
COM	1	Mã lệnh, từ 0x00 tới 0xFF.
RCLS	1	Lớp lỗi.
REH	1	Dự trữ.
ERR	2	Mã lỗi.
REB	1	Dự trữ.
RES	14	Dự trữ.
TID	2	Định danh cây (TID); định danh duy nhất cho tài nguyên được client sử dụng.
PID	2	PID của máy gọi.
UID	2	UID.
MID	2	Định danh kép dẫn vào một kênh truyền thông, được dùng để phân đường cho các yêu cầu bên trong tiến trình.

### Dạng của râu dòng lệnh

Tiếp theo ngay sau phần đầu là một số lượng thay đổi được các byte để tạo nên một lệnh SMB hay câu trả lời. Mỗi một lệnh, ví dụ như Open File (COM field identifier: SMBopen) hoặc Get Print Queue (SMBsplretq), có các tập hợp tham số và dữ liệu riêng của mình. Tương tự như đối với các trường của phần đầu SMB, tùy theo lệnh cụ thể được gọi, ta không cần phải điền tất cả giá trị vào các trường của lệnh. Ví dụ, lệnh Get Server Attributes (SMBdskattr) đặt các số 0 vào các trường WCT và BCC. Các trường của phần lệnh được chỉ ra trong bảng dưới đây.

### Nội dung lệnh SMB

Trường	Kích thước (bytes)	Mô tả
WCT	1	Đếm từ.
VWV	Biến đổi	Các từ tham số (kích thước xác định bởi WCT)
BCC	2	Đếm byte tham số.
DATA	Biến đổi	Dữ liệu (kích thước xác định bởi BCC)

### Biến thể của SMB

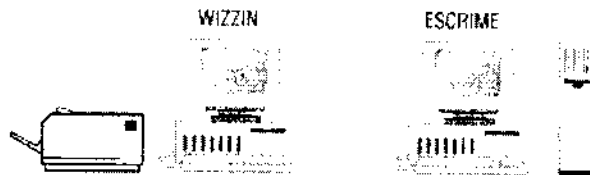
Giao thức SMB, từ khi xuất hiện, đã được mở rộng đáng kể vài lần với những lệnh mới. Trong bảng sau chỉ ra những điểm chính của các phiên bản SMB chính. Trong mỗi phiên bản chính đó lại có nhiều phiên bản “con” chứa nhiều lệnh cho các trường hợp cụ thể. Xâu ID string được máy client và server dùng để xác định mức của giao thức mà chúng sẽ dùng nói chuyện với nhau.

### Ngôn ngữ riêng của giao thức SMB

Tên giao thức	ID String	Được dùng bởi
Core	PC NETWORK PROGRAM 1.0	
Core Plus	MICROSOFT NETWORKS 1.03	
LAN Manager 1.0	LANMAN1.0	
LAN Manager 2.0	LM1.2X002	
LAN Manager 2.1	LANMAN2.1	
NT LAN Manager 1.0	NT LM 0.12	Windows NT 4.0
Samba's NT LM 0.12	Samba	Samba
Common Internet File System	CIFS 1.0	Windows 2000

### ■ *SMB Clients và Server*

Như đã trình bày ở trên, SMB là giao thức client/server, nghĩa là một máy client gửi một yêu cầu tới server, server đáp ứng yêu cầu đó và trả lời cho máy client. Tuy nhiên, vai trò client/server có thể đảo ngược, đôi khi ngay trong phạm vi một phiên làm việc SMB. Ví dụ, ta xét hai máy tính chạy trên Windows 95/98 trên H. XVI. 5.. Máy tính có tên WIZZIN chia sẻ máy in cho mạng, và máy tính ESCRIME chia sẻ thư mục đĩa cứng. WIZZIN đóng vai trò client khi truy nhập vào ổ đĩa mạng của ESCRIME, và trở thành server khi thực hiện công việc in cho ESCRIME.



H. XVI. 5. Hai máy tính đều có tài nguyên có thể chia sẻ

Ta rút ra được các điểm quan trọng sau trong thuật ngữ của Samba:

- Server là máy tính có tài nguyên được chia sẻ.
- Client là máy tính muốn sử dụng các tài nguyên đó.
- Một server có thể thành một client (của tài nguyên thuộc máy tính khác) tại mọi thời điểm đã cho.

Các sản phẩm Microsoft Windows có cài đặt sẵn cả SMB client và server vào hệ điều hành. Windows NT 4.0 sử dụng phiên bản mới hơn của giao thức SMB so với Windows for Workgroups, với các cải tiến về bảo mật mạng. Một số rất lớn các sản phẩm thương mại SMB server được nhiều hãng như Sun, Compaq, SCO, Hewlett-Packard, Syntax, và IBM phát hành. Tiếc rằng, về phía SMB client người ta không nhận thấy sự cung cấp phong phú như vậy: chỉ có sản phẩm Pathworks của Digital Equipment, và tất nhiên, Samba.

### ▪ . Với kết B (B) đơn giản

Ta hãy xét một nối kết SMB đơn giản. Để thực hiện kết nối tới tài nguyên, cả server và client phải thực hiện bốn bước sau:

- Thiết lập nối kết áo.
- Trao đổi về phương án giao thức định dùng.
- Đặt các thông số cho phiên làm việc.
- Tạo kết nối dạng cây tới tài nguyên.

Ta sẽ theo dõi từng bước trên thông qua một tiện ích rất hiệu quả. Đó là tiện ích **tcpdump** đã hoàn thiện mà bạn có thể tải về từ site của Samba <http://www.samba.org> , trong thư mục `samba/ftp/tcpdump-smb`. Phiên bản cuối cùng hiện nay của **tcpdump** là 3.4-5. Bạn có thể dùng tiện ích này như dùng ứng dụng **tcpdump** chuẩn, nhưng phải thêm vào khóa `-s 1500` để đảm bảo bạn đã lấy về toàn bộ gói phần mềm chứ không phải chỉ vài byte đầu.

### Thiết lập nối kết áo

Khi người dùng lần đầu tạo yêu cầu để truy nhập ổ đĩa mạng hoặc gửi tài liệu tới máy in ở máy in từ xa, NetBIOS đảm bảo tạo nối kết tại lớp phiên làm việc (xem mô hình mạng OSI/ISO). Kết quả là xuất hiện một kênh áo hai chiều giữa client và server. Trong thực tế, chỉ có hai thông báo mà theo đó client và server cần thiết lập nối kết đã nêu. Ta có thể theo dõi các yêu cầu và trả lời được bắt nhờ lệnh **tcpdump**:

```
>>> NBT Packet
NBT Session Request
Flags=0x81000044
Destination=ESCRIME
NameType=0x20 (Server)
Source=WIZZIN
NameType=0x00 (Workstation)
>>> NBT Packet
NBT Session Granted
Flags=0x82000000
```



### *Trao đổi về phương án giao thức định dùng*

Tại thời điểm này một kênh liên lạc đã được mở giữa client và server. Tiếp theo, client gửi thông báo tới server để trao đổi về giao thức SMB. Như đã trình bày ở trên, client đặt các số 0 vào trong trường định danh cây (TID) của nó, vì chưa biết TID nào sẽ được dùng.

Lệnh trong thông báo là SMBnegprot, một yêu cầu điều đình phương án giao thức SMB sẽ được sử dụng trong suốt thời gian phiên làm việc. Cần nhấn mạnh rằng chính client gửi cho server danh sách các phương án của giao thức SMB nó có thể dùng được, chứ không phải theo chiều ngược lại.

Server trả lời yêu cầu SMBnegprot với chỉ số vào trong danh sách mà client đề nghị, bắt đầu từ chỉ số 0, hoặc với giá trị 0xFF nếu không có phương án nào được chấp nhận. Tiếp tục theo dõi ví dụ, server trả lời với giá trị 5, có nghĩa NT LM 0.12 sẽ được dùng:

```
>>> NBT Packet
NBT Session Packet
Flags=0x0
Length=154 SMB PACKET: SMBnegprot (REQUEST)
SMB Command = 0x72
Error class = 0x0
Error code = 0
Flags1 = 0x0
Flags2 = 0x0
Tree ID = 0
Proc ID = 5371
UID = 0
MID = 385
Word Count = 0
Dialect=PC NETWORK PROGRAM 1.0
Dialect=MICROSOFT NETWORKS 3.0
Dialect=DOS LM1.2X002
Dialect=DOS LANMAN2.1
```

```
Dialect=Windows for Workgroups 3.1a
Dialect=NT LM 0.12 >>> NBT Packet
NBT Session Packet
Flags=0x0
Length=69 SMB PACKET: SMBnegprot (REPLY)
SMB Command = 0x72
Error class = 0x0
Error code = 0
Flags1 = 0x0
Flags2 = 0x1
Tree ID = 0
Proc ID = 5371
UID = 0
MID = 385
Word Count = 02
[000] 05 00
```

### Thiết lập phiên làm việc và các thông số truy nhập

Bước tiếp theo là phiên làm việc truyền nhận và các thông số truy nhập cho phiên làm việc đó, bao gồm cả tên tài khoản và mật khẩu, tên nhóm làm việc, kích thước cực đại của dữ liệu sẽ được truyền, và số các yêu cầu được xếp trong hàng chờ tại một thời điểm bất kỳ.

Trong ví dụ sau, lệnh `Session Setup` cho phép các lệnh phụ trợ SMB khác có thể được đi kèm. Chữ cái "X" ở cuối tên lệnh chỉ ra rằng mã hexa của lệnh thứ hai được xác định trong trường `Com2`, trong trường hợp ví dụ này là `0x75`, có nghĩa là lệnh `Tree Connect and X`. Thông báo `SMBtconX` tìm tên của tài nguyên trong bộ đệm `smb_buf`. Trong ví dụ được xét, bộ đệm `smb_buf` chứa xâu `\\ESCRIME\\PUBLIC`, là đường dẫn đầy đủ tới thư mục được chia sẻ trên nút mạng `ESCRIME`. Việc dùng lệnh "and X" đẩy nhanh tốc độ mỗi cuộc truyền, vì server không phải chờ client tạo thông báo thứ hai.

Chú ý rằng cho đến bây giờ TID vẫn là zero. Server sẽ cung cấp TID cho client một khi phiên làm việc được xác lập và nối kết đã được tạo tới tài nguyên được yêu

câu. Thêm nữa, chú ý rằng mật khẩu được gửi đi ở dạng mở. Sau này ta sẽ thay đổi bằng cách dùng mật khẩu được mã hóa:

```
>>> NBT Packet
NBT Session Packet
Flags=0x0
Length=139 SMB PACKET: SMBsesssetupX (REQUEST)
SMB Command = 0x73
Error class = 0x0
Error code = 0
Flags1 = 0x10
Flags2 = 0x0
Tree ID = 0
Proc ID = 5371
UID = 1
MID = 385
Word Count = 13
Com2=0x75
Res1=0x0
Off2=106
MaxBuffer=2920
MaxMpx=2
VcNumber=0
SessionKey=0x1FF2
CaseInsensitivePasswordLength=1
CaseSensitivePasswordLength=1
Res=0x0
Capabilities=0x1
Pass1&Pass2&Account&Domain&OS&LanMan=
 KRISTIN PARKSTR Windows 4.0 Windows 4.0
PassLen=2
Passwd&Path&Device=
smb_bcc=22
```

---

```
smb_buf [] = \\ESCRIME\\PUBLIC
```

### Tạo kết nối tới tài nguyên

Trong bước cuối cùng, server trả TID về cho client, biểu thị rằng người dùng đã được phép truy nhập và tài nguyên đã sẵn sàng để được dùng. Server còn đặt "A" vào trường ServiceType để chỉ đó là dịch vụ file. Các dịch vụ sẵn có là:

- "A" đối với đĩa cứng hoặc file.
- "LPT1" đối với kết quả ra được xếp hàng chờ.
- "COMM" đối với các modem hay máy in nối trực tiếp.
- "IPC" đối với đường ống được đặt tên.

Kết quả ra có dạng:

```
>>> NBT Packet
NBT Session Packet
Flags=0x0
Length=78 SMB PACKET: SMBsesssetupX (REPLY)
SMB Command = 0x73
Error class = 0x0
Error code = 0
Flags1 = 0x80
Flags2 = 0x1
Tree ID = 121
Proc ID = 5371
UID = 1
MID = 385
Word Count = 3
Com2=0x75
Off2=68
Action=0x1
[000] Unix Samba 1.9.1
[010] PARKSTR SMB PACKET: SMBtconX (REPLY) (CHAINED)
smbvbw[] =
```

```
Com2=0xFF
Off2=78
smbbuf [] =
ServiceType=A:
```

### XVI. 3. Cài đặt Samba trên hệ thống UNIX

#### XVI. 3. a. Tạo file cấu hình cài đặt *makefile* của Samba

##### ▪ *‘Mã nguồn hay bản nhị phân?’*

Các gói phần mềm biên dịch sẵn xuất hiện rất nhiều trong số lớn các bản UNIX, chứa mã nhị phân cho các file thực thi và tài liệu của Samba. Việc cài đặt bằng các file nhị phân giúp bạn tiết kiệm rất nhiều thời gian và công sức, tránh được nhiều lỗi, nhưng bị hạn chế hơn về tính tùy biến so với việc tự biên dịch lấy từ mã nguồn. Sau đây là các điểm cần chú ý khi bạn lựa chọn bản mã nguồn hay bản nhị phân:

- ✦ Các gói nhị phân có thể bị chậm so với các phiên bản mới nhất được phát hành, hoặc không có sẵn cho các hệ UNIX ít phổ biến.

- ✦ Khi dùng bản nhị phân biên dịch sẵn, bạn cần phải có đúng các thư viện cần cho các file thực thi. Trên một số hệ điều hành các file thực thi được liên kết tĩnh với các thư viện, nhưng trên các hệ UNIX hiện đại (như Linux, SGI IRIX, Solaris, HP-UX, v.v...), các thư viện thường được liên kết động. Điều đó có nghĩa là ta phải tìm cho đúng các phiên bản thư viện phù hợp với các phiên bản nhị phân.

Có một tiện ích trên nhiều máy có tên là **ldd** giúp nhận biết bản nhị phân cụ thể nào cần bản thư viện cụ thể nào và các thư viện nào có sẵn trong máy thỏa mãn điều kiện tìm kiếm trên. Ví dụ, kiểm tra chương trình **smbd** trên máy thử nghiệm đã cho các nhà nghiên cứu kết quả sau:

```
$ ldd smbdl libreadline.so.3 =>
 /usr/lib/libreadline.so.3
libdl.so.2 => /lib/libdl.so.2
libcrypt.so.1 => /lib/libcrypt.so.1
libc.so.6 => /lib/libc.so.6
```

---

```
libtermcap.so.2 => /lib/libtermcap.so.2
/lib/ld-linux.so.2 => /lib/ld-linux.so.2
```

♣ Nhớ rằng mỗi bộ cài đặt nhị phân thường mang theo các giá trị định trước về hệ điều hành đích như là các thư mục mặc định và các giá trị tùy chọn cấu hình. Vì thế, bạn hãy đọc thật kỹ tài liệu kèm theo và file `makefile` trong thư mục nguồn để tìm các chỉ dẫn và biến được dùng khi biên dịch mã nhị phân.

Một vài mục cấu hình có thể được đặt lại bằng các tùy chọn ở chế độ dòng lệnh thay vì trong thời gian biên dịch. Ví dụ, nếu bạn nhị phân của bạn cứ cố đặt mọi file log, file khóa hay file trạng thái vào chỗ “sai” (ví dụ, trong `/usr/local`), bạn có thể ghi đè lên mà không phải biên dịch lại.

Một điểm phải nhấn mạnh là mã nguồn Samba đòi hỏi bộ biên dịch ANSI C. Nếu bạn đang dùng hệ điều hành với bộ biên dịch khác với bộ ANSI, ví dụ như bộ biên dịch `cc` trên SunOS version 4, bạn có thể cài đặt một bộ biên dịch tương thích với ANSI như là `gcc` trước khi tiến hành mọi thứ. Nói chung, vì để cho có độ uyển chuyển và tương thích lớn nhất, người ta khuyên dùng cách biên dịch bản mới nhất mã nguồn của Samba.

#### ▪ *Cấu hình cài đặt Samba*

Bộ nguồn cài đặt của Samba phiên bản 2.0 trở đi ban đầu không có file `makefile`. Thay vào đó, ta phải sinh ra nó thông qua script cấu hình của GNU, nằm ở thư mục `samba-2.0.x /source/`. Script cấu hình này được chạy với quyền `root` và đảm bảo dựng Samba theo các đặc điểm của máy tính cụ thể. Tuy nhiên, có thể xuất hiện nhu cầu đối với vài tùy chọn chung và chúng được đặt bằng cách thông qua các tùy chọn trên dòng lệnh:

```
./configure --with-ssl
```

Ví dụ, dòng lệnh trên sẽ dựng file `makefile` của Samba có hỗ trợ cho giao thức mã hóa *Secure Sockets Layer* (SSL) để tăng cường bảo mật. Để xem toàn bộ danh sách các tùy chọn chung đó, ta dùng lệnh:

```
./configure -help
```

Mỗi một tùy chọn nói trên có thể bật lên hoặc tắt đi nhiều đặc điểm đặc trưng. Bình thường, ta bật lên một đặc điểm bằng cách gõ lệnh có tùy chọn theo cú pháp `--with-feature`, làm cho đặc điểm đó được biên dịch và cài đặt. Tương tự, nếu ta không muốn dùng một đặc điểm nào thì dùng tùy chọn theo cú pháp `--without-feature`. Đối với Samba 2.0.5, mỗi một đặc điểm tùy chọn sau có giá trị mặc định là tắt:

**`--with-smbwrapper`**

Bao gồm hỗ trợ trình bao học SMB, cho phép các chương trình thực thi từ phía máy UNIX có thể truy nhập được các hệ thống file SMB/CIFS như là các hệ thống file UNIX bình thường. Tùy chọn này được khuyên dùng. Tuy nhiên, cho đến thời điểm này vẫn còn một số chi tiết không tương thích giữa gói phần mềm trình bao học **smbwrapper** với thư viện GNU `libc` version 2.1, và không biên dịch được trên máy Linux bản Red Hat 6.0. Chú ý theo dõi các cập nhật trên trang chủ của Samba <http://www.samba.org>.

**`--with-afs`**

Bao gồm hỗ trợ hệ thống file Andrew FileSystem được đưa ra từ Trường Đại học Carnegie Mellon. Nếu bạn muốn dùng tới các file AFS thông qua Samba, nên biên dịch Samba lần thứ nhất không có tùy chọn này để tin chắc rằng Samba đã chạy tốt. Sau đó mới nên biên dịch lại với tùy chọn này cho các file AFS và so sánh các lỗi có thể có với lần thiết lập trước.

**`--with-dfs`**

Bao gồm hỗ trợ DFS, bản sau của AFS, được OSF/1 (Digital Unix) dùng (xem chương I *Tổng quan về hệ điều hành UNIX*). Chú ý rằng DFS ở đây không giống Microsoft DFS, là hệ thống file hoàn toàn khác. Cách biên dịch cũng nên thực hiện như đối với tùy chọn AFS.

**`--with-krb4=base-directory`**

Bao gồm hỗ trợ Kerberos phiên bản 4.0 như là thư mục chính của bộ cài đặt. Kerberos là giao thức bảo mật mạng được công bố từ Viện Công nghệ Massachusetts (MIT) và dùng các khóa mật mã riêng để cung cấp bảo mật rất mạnh giữa các nút mạng. Ngày nay, Microsoft công bố sẽ dùng Kerberos 5.0 làm cơ chế chuẩn kiểm tra xác nhận người dùng cho Microsoft Windows 2000 (NT

5.0). Tuy nhiên, cơ chế kiểm tra xác nhận người dùng của Kerberos 5.0 khác hẳn so với các cơ chế bảo mật của Kerberos 4.0. Nếu máy của bạn dùng Kerberos phiên bản 4, nên nâng cấp lên phiên bản 5 và dùng tùy chọn `--with-krb5` (xem tùy chọn tiếp theo). Có thể tham khảo thêm thông tin về Kerberos từ site <http://web.mit.edu/kerberos/www>.

**--with-krb5=base-directory**

Bao gồm hỗ trợ Kerberos phiên bản 5.0 như là thư mục chính của bộ cài đặt. Microsoft công bố sẽ dùng Kerberos 5.0 làm cơ chế chuẩn kiểm tra xác nhận người dùng cho Microsoft Windows 2000 (NT 5.0). Tuy nhiên, không có gì đảm bảo rằng Microsoft không mở rộng Kerberos cho những mục đích riêng của họ trong tương lai. Hiện nay, Kerberos của Samba chỉ hỗ trợ giao diện mật khẩu dạng plaintext mà không phải dạng được mã hóa. Có thể tham khảo thêm từ site <http://web.mit.edu/kerberos/www> thông tin về Kerberos.

**--with-automount**

Bao gồm hỗ trợ cho **automounter**, một đặc điểm hay được các site có hỗ trợ NFS ưa dùng.

**--with-smbmount**

Bao gồm hỗ trợ lệnh **smbmount** chỉ được dùng cho Linux. Đặc điểm này hiện nay không được bảo trì, vì thế nên coi là đặc điểm tùy chọn. Thay vào đó, ta nên dùng **smbwrapper**. Lệnh **smbwrapper** hoạt động trên nhiều hệ UNIX hơn là **smbmount**, nên ta thường hay dùng `--with-smbwrapper` thay cho tùy chọn này.

**--with-pam**

Bao gồm hỗ trợ Pluggable Authentication Modules - PAM, một đặc điểm chung trong các hệ điều hành Linux.

**--with-ldap**

Bao gồm hỗ trợ Lightweight Directory Access Protocol (LDAP). Phiên bản tương lai của LDAP được dùng trong Windows 2000 (NT 5.0); Samba đang thử nghiệm hỗ trợ. LDAP là giao thức thư mục client-server linh hoạt, có thể mang các thông tin như giấy chứng nhận (certificate) và tư cách thành viên của nhóm.

**--with-nis**

Bao gồm hỗ trợ việc lấy thông tin của file **password** từ NIS (network yellow pages - những trang vàng của mạng).



**--with-nisplus**

Bao gồm hỗ trợ việc lấy thông tin của file password từ NIS+, bản sau của NIS.

**--with-ssl**

Bao gồm hỗ trợ thử nghiệm đối với Secure Sockets Layer (SSL), dùng để cung cấp nối kết được mã hóa từ client tới server.

**--with-nisplus-home**

Bao gồm hỗ trợ cho việc định vị server có chứa thư mục home của người dùng cụ thể và báo cho người dùng kết nối đến vị trí đó. Tùy chọn này đòi hỏi `--with-nis`, và thường là với cả `--with-automounter`.

**--with-mmap**

Bao gồm hỗ trợ thử nghiệm mã ánh xạ bộ nhớ. Không cần thiết đối với việc khóa nhanh, vì khóa nhanh đã dùng mmap hoặc bộ nhớ được chia sẻ của System V.

**--with-syslog**

Bao gồm hỗ trợ để dùng tiện ích SYSLOG cho việc ghi lại các thông tin log được server Samba tạo ra. Đó là các tùy chọn cấu hình Samba (xem phần *Chia sẻ đĩa cứng của Samba*).

**--with-netatalk**

Bao gồm hỗ trợ thử nghiệm tương tác với Netatalk file server của máy Macintosh.

**--with-quotas**

Bao gồm hỗ trợ hạn ngạch đĩa cứng.

Có một số thông số phụ trợ nữa mà bạn có thể đưa thêm vào trong script cấu hình, nếu muốn cài đặt các phần của Samba ở các vị trí khác nhau, có thể là ở nhiều đĩa cứng hoặc phần đĩa cứng.

Cần nhắc lại là trước khi chạy script cấu hình, bạn phải có quyền `root` của hệ thống. Nếu không, bạn sẽ nhận được cảnh báo theo kiểu: "configure: warning: running as non-root will disable some tests".

Tất nhiên bạn không muốn bỏ đi một phép kiểm tra nào khi file `makefile` của Samba được tạo, vì trong trường hợp ngược lại, sẽ tiềm tàng khả năng mắc lỗi khi biên dịch hay khi chạy Samba trên máy của bạn.

Các thông số cấu hình phụ trợ được nhắc tới hiện diện trong bảng sau.

**Các tùy chọn cấu hình phụ trợ**

Tùy chọn	Ý nghĩa	Mặc định
--prefix= directory	Cài đặt các file không phụ thuộc vào kiến trúc máy tính vào thư mục cơ sở cụ thể.	/usr/local/samba
--eprefix= directory	Cài đặt các file phụ thuộc vào kiến trúc máy tính vào thư mục cơ sở cụ thể.	/usr/local/samba
--bindir= directory	Cài đặt các file thực thi của người dùng vào thư mục cơ sở cụ thể.	eprefix/bin
--sbindir= directory	Cài đặt các file thực thi của người quản trị vào thư mục cơ sở cụ thể.	eprefix/bin
--libexecdir= directory	Cài đặt các chương trình thực thi vào thư mục cơ sở cụ thể.	eprefix/libexec
--datadir= directory	Cài đặt các file chỉ đọc được, không phụ thuộc vào kiến trúc máy tính, vào thư mục cơ sở cụ thể.	prefix/share
--libdir= directory	Cài đặt các file thư viện trong thư mục cơ sở cụ thể.	eprefix/lib
--includedir= directory	Cài đặt các file của gói phần mềm include vào thư mục cơ sở cụ thể.	prefix/include
--infodir= directory	Cài đặt các file thông tin thêm vào thư mục cơ sở cụ thể.	prefix/info
--mandir= directory	Cài đặt các trang <b>man</b> vào thư mục cơ sở cụ thể.	prefix/man

Dưới đây là ví dụ chạy script cấu hình được tạo bởi file makefile của Samba 2.0.4 trên nền Linux. Chú ý rằng bạn phải chạy script cấu hình trong thư mục nguồn và trong đoạn trích dưới đây có bỏ qua vài dòng không mang nhiều thông tin liên quan tới việc cấu hình ở trong ví dụ này:

```
cd samba-2.0.4b/source/
./configure | tee mylog loading cache ./config.cache
checking for gcc... (cached) gcc
checking whether the C compiler (gcc -O) works... yes
checking whether the C compiler (gcc -O) is a cross-
 compiler... no
checking whether we are using GNU C... (cached) yes
checking whether gcc accepts -g... (cached) yes
checking for a BSD compatible install... (cached)
 /usr/bin/install -c ... (bỏ qua một số nội
 dung)...checking configure summary
configure OK
creating ./config.status
creating include/stamp-h
creating Makefile
creating include/config.h
```

Nói chung, mọi thông báo cấu hình không được bắt đầu từ chữ “checking” hoặc “creating” sẽ là lỗi; lệnh **tee** giúp ta đổi hướng kết quả ra của script cấu hình ra file để ta dễ tìm lỗi hơn. Nếu có lỗi trong quá trình cấu hình thì các thông tin cụ thể hơn có thể được tìm thấy trong file `config.log` được ghi vào thư mục tại chỗ bởi chính script cấu hình.

Nếu thao tác cấu hình được thực hiện suôn sẻ, bạn sẽ nhìn thấy thông báo tổng kết kiểm tra cấu hình gồm dòng `configure OK` và bốn hay năm thông báo tạo file. Bước tiếp theo sẽ là biên dịch Samba.

### XVI. 3. b. Biên dịch và cài đặt Samba

Quá trình biên dịch sẽ tạo nên các file thực thi của Samba và không phải là khó thực hiện. Trong thư mục nguồn, ta gõ lệnh **make** ở chế độ dòng lệnh. Tiện ích **make** sẽ sinh ra một loạt các thông báo giải thích và kết quả, được bắt đầu bằng câu:

```
Using FLAGS = -O -Iinclude ...
```

Kết quả sẽ bao gồm các bản biên dịch của cả **smbd** và **nmbd**, và được kết thúc bằng lệnh liên kết đối với **bin/make\_printerdef**. Ta lại xét tiếp ví dụ kết quả ra của lệnh **make** cho Samba phiên bản 2.0.4 trên server Linux:

```
make
```

```
Using FLAGS = -O -Iinclude -I./include -I./ubiqx -
 I./smbwrapper -
 DSMBLOGFILE="/usr/local/samba/var/log.smb" -
 DNMBLOGFILE="/usr/local/samba/var/log.nmb" -
 DCONFIGFILE="/usr/local/samba/lib/smb.conf" -
 DLMHOSTSFILE="/usr/local/samba/lib/lmhosts" -
 DSWATDIR="/usr/local/samba/swat" -
 DSBINDIR="/usr/local/samba/bin" -
 DLOCKDIR="/usr/local/samba/var/locks" -
 DSMBRUN="/usr/local/samba/bin/smbmun" -
 DCODEPAGEDIR="/usr/local/samba/lib/codepages" -
 DDRIVERFILE="/usr/local/samba/lib/printers.def" -
 DBINDIR="/usr/local/samba/bin" -DHAVE_INCLUDES_H -
 DPASSWD_PROGRAM="/bin/passwd" -
 DSMB_PASSWD_FILE="/usr/local/samba/private/smbpasswd"
Using FLAGS32 = -O -Iinclude -I./include -I./ubiqx -
 I./smbwrapper -
 DSMBLOGFILE="/usr/local/samba/var/log.smb" -
 DNMBLOGFILE="/usr/local/samba/var/log.nmb" -
```

```
DCONFIGFILE="/usr/local/samba/lib/smb.conf" -
DLMHOSTSFILE="/usr/local/samba/lib/lmhosts" -
DSWATDIR="/usr/local/samba/swat" -
DSBINDIR="/usr/local/samba/bin" -
DLOCKDIR="/usr/local/samba/var/locks" -
DSMBRUN="/usr/local/samba/bin/smbd" -
DCODEPAGEDIR="/usr/local/samba/lib/codepages" -
DDRIVERFILE="/usr/local/samba/lib/printers.def" -
DBINDIR="/usr/local/samba/bin" -DHAVE_INCLUDES_H -
DPASSWD_PROGRAM="/bin/passwd" -
DSMB_PASSWD_FILE="/usr/local/samba/private/smbpasswd"
```

```
Using LIBS = -lreadline -ldl -lcrypt -lpam
```

```
Compiling smbd/server.c
```

```
Compiling smbd/files.c
```

```
Compiling smbd/chgpasswd.c ... (nội dung được bỏ qua) ...
```

```
Compiling rpcclient/cmd_samr.c
```

```
Compiling rpcclient/cmd_reg.c
```

```
Compiling rpcclient/cmd_srvsvc.c
```

```
Compiling rpcclient/cmd_netlogon.c
```

```
Linking bin/rpcclient
```

```
Compiling utils/smbpasswd.c
```

```
Linking bin/smbpasswd
```

```
Compiling utils/make_smbcodepage.c
```

```
Linking bin/make_smbcodepage
```

```
Compiling utils/nmblookup.c
```

```
Linking bin/nmblookup
```

```
Compiling utils/make_printerdef.c
```

```
Linking bin/make_printerdef
```

Nếu thấy có lỗi trong khi biên dịch, bạn nên kiểm tra lại tài liệu của Samba xem lỗi đó đã được ghi nhận chưa. Một khả năng khác là tham gia vào danh sách gửi thư của Samba, được kê ở cuối appd-34717, và trong trang chủ Samba để trao đổi tin

tức với những người cùng dùng Samba. Hầu hết các thông báo về biên dịch được ghi nhận là phụ thuộc vào các đặc điểm của máy tính cụ thể và có thể giải quyết được dễ dàng.

Sau khi các file đã được biên dịch, bạn có thể cài đặt chúng vào các thư mục được bạn chỉ định bằng cách dùng lệnh:

```
make install
```

Nếu phải nâng cấp Samba, các file của bản Samba cũ sẽ được lưu lại với phần mở rộng `.old`, và bạn có thể quay lại phiên bản Samba cũ bằng lệnh **make revert**. Sau khi đã cài đặt bằng lệnh **make**, bạn cần sao các file `.old`, nếu có, vào một vị trí mới hoặc đổi sang những tên mới. Nếu không, lần cài đặt sau của Samba sẽ ghi đè các file `.old` nguyên thủy, làm bạn mất khả năng quay trở về phiên bản cũ theo nhiều bước. Nếu bạn cấu hình cho Samba dùng các vị trí mặc định cho các file, các file mới sẽ được cài vào các thư mục được chỉ trong bảng dưới đây. Nhớ rằng bạn phải thực hiện cài đặt bằng tài khoản người dùng có quyền ghi trên các thư mục đích; thường đó là tài khoản của `superuser` hay `root`.

### Các thư mục cài đặt của Samba

Thư mục	Mô tả
<code>/usr/local/samba</code>	Cây thư mục chính
<code>/usr/local/samba/bin</code>	Các file nhị phân
<code>/usr/local/samba/lib</code>	<code>smb.conf</code> , <code>lmhosts</code> , các file cấu hình, v.v...
<code>/usr/local/samba/man</code>	Tài liệu về Samba.
<code>/usr/local/samba/private</code>	File mật khẩu đã mã hóa của Samba.
<code>/usr/local/samba/swat</code>	Các file công cụ quản trị Web của Samba (Samba Web Administration Tool-SWAT).
<code>/usr/local/samba/var</code>	Các file log của Samba, file khóa, thông tin danh sách bộ duyệt, các file bộ nhớ chia sẻ, các file định danh tiến trình (PID).

Từ nay về sau, để cho gọn, ta ký hiệu cây thư mục chính của Samba /usr/local/samba bằng tên `samba_dir`. Trong trường hợp thư mục /usr trên máy của bạn chỉ đọc được (không có quyền ghi w), bạn phải nghĩ đến việc đặt các file log, khóa và mật khẩu vào vị trí khác.

Sau đây là ví dụ cài đặt Samba với thư mục cơ sở là /usr/local/samba, tức là `samba_dir`:

```
make install
Using FLAGS = -O -Iinclude -I./include -I./ubiqx -
I./smbwrapper -
DSMBLOGFILE="/usr/local/samba/var/log.smb" -
DNMBLOGFILE="/usr/local/samba/var/log.nmb" -
DCONFIGFILE="/usr/local/samba/lib/smb.conf" -
... (bỏ qua một số nội dung) ... The binaries are
installed. You may restore the old binaries (if
there were any) using the command "make revert".
You may uninstall the binaries using the command
"make uninstallbin" or "make uninstall" to
uninstall binaries, man pages and shell scripts.
... (bỏ qua một số nội dung) ...
=====
The SWAT files have been installed. Remember to read
the README for information on enabling and using
SWAT.
```

Nếu bạn đọc được thông báo cuối cùng là về SWAT, có nghĩa là bạn đã cài đặt thành công và Samba sẵn sàng được đưa vào hoạt động trên máy của bạn.

#### ▪ *Những bước cài đặt cuối cùng*

Ta chỉ phải còn thực hiện vài bước cài đặt cuối cùng. Đặc biệt là ta phải thêm công cụ quản trị Web của Samba (*Samba Web Administration Tool* - SWAT) vào các file cấu hình /etc/services và /etc/inetd.conf. SWAT hoạt động

như một daemon dưới `inetd` và cung cấp một trình soạn thảo trên cơ sở có định dạng trong bộ trình duyệt để tạo và thay đổi các file cấu hình SMB.

Để thêm SWAT, ta thêm dòng vào sau vào cuối file `/etc/services`:

```
swat 901/tcp
```

Thêm các dòng sau vào file `/etc/inetd.conf`. (Kiểm tra trang [man](#) về `inetd.conf` để xem dạng chính xác của file `inetd.conf` nếu nó khác với dạng được trình bày trong ví dụ trên.) Đừng quên thay đổi đường dẫn cho các file nhị phân của SWAT nếu bạn cài đặt chúng vào vị trí khác với thư mục mặc định `/usr/local/samba`.

```
swat stream tcp nowait.400 root
 /usr/local/samba/bin/swat swat
```

Đó là những gì liên quan tới việc tạo file cấu hình cài đặt và cài đặt Samba. Tuy nhiên, để có thể sử dụng được Samba, trước tiên, bạn phải tạo file cấu hình cho nó.

### ***XVI. 3. c. File cấu hình cơ bản của Samba***

Chìa khóa của việc cấu hình Samba là một file cấu hình đơn độc `smb.conf`, có thể rất đơn giản, nhưng cũng có thể cực kỳ phức tạp. Hầu như mọi tài liệu về Samba đều xoay quanh file cấu hình này và ta sẽ xem xét nó một cách chi tiết hơn nhiều trong một mục riêng. Tuy nhiên, tại thời điểm này, để cho đơn giản ta chỉ thiết lập một dịch vụ đơn đối với file, cho phép ta phát hiện thấy các daemon Samba và mình chứng được rằng mọi thứ đều hoạt động đúng như dự kiến của ta khi cài đặt.

Quá trình cài đặt không tự động tạo ra file cấu hình `smb.conf`, dù rằng một số file ví dụ mẫu `smb.conf` được gộp vào trong bộ cài đặt Samba. File cấu hình `smb.conf` thường nằm ở thư mục `/usr/local/samba/lib`, tuy nhiên, nếu bạn không tự biên dịch mà tải luôn bản mã nhị phân để cài đặt về dùng thì nên kiểm tra lại vị trí của `smb.conf`. Hoặc nếu máy UNIX đã cài đặt sẵn Samba, chắc chắn file cấu hình `smb.conf` phải nằm ở đâu đó trong máy. Ta hãy lấy ví dụ một file cấu hình Samba đơn giản `smb.conf` để kiểm tra phần mềm server như sau:



```
[global]
 workgroup = SIMPLE
[test]
 comment = For testing only, please
 path = /export/samba/test
 read only = no
 guest ok = yes
```

File cấu hình đơn giản này bắt server Samba đưa ra thư mục `/export/samba/test` nằm trên server như là một hệ thống file SMB/CIFS chia sẻ được gọi để kiểm tra. Server còn là thành viên của nhóm làm việc `SIMPLE`. Phần được chia sẻ `[test]`, sẽ được trình bày sau, dùng để thiết lập các clients trên Windows. Bây giờ bạn có thể hoàn thành việc thiết lập Samba bằng cách thực hiện các lệnh sau trên máy UNIX server với quyền `root`:

```
mkdir /export/samba/test
chmod 777 /export/samba/test
```

Sau này ta sẽ thấy file cấu hình trên là phương án tối nhất khi xét trên góc độ bảo mật. Tuy nhiên, vào thời điểm này ta chỉ quan tâm kiểm tra Samba có hoạt động tốt hay không mà thôi, nên vấn đề bảo mật tạm thời được xếp sang một bên.

Nếu bạn dùng Windows 98 hoặc Windows NT Service Pack 3 trở lên, bạn cần thêm dòng vào như sau vào trong phần `[global]` của file cấu hình Samba: `encrypt passwords = yes`. Thêm nữa, bạn cần dùng chương trình `smbpassword`, thường nằm trong thư mục `/usr/local/samba/bin/` để vào lại tổ hợp tên người dùng / mật khẩu cho những người dùng trên server UNIX có quyền truy nhập tới cơ sở dữ liệu Samba được mã hóa cho client. Ví dụ, nếu bạn muốn cho phép một người dùng UNIX có tên chung truy nhập vào tài nguyên được chia sẻ từ một máy client SMB, bạn phải gõ lệnh: **`smbpassword -a chung`**. Nếu người dùng lần đầu được thêm vào, chương trình sẽ đưa ra một thông báo lỗi nói rằng không tồn tại cơ sở dữ liệu mật khẩu đã mã hóa. Bạn đừng bối rối vì chương trình sẽ tự tạo cơ sở dữ liệu ấy. Chỉ cần quan tâm sao cho tổ hợp tên người dùng /

mật khẩu mà bạn thêm vào trong cơ sở dữ liệu mật khẩu đã mã hóa trùng với tên người dùng và mật khẩu mà bạn muốn dùng bên phía client Windows.

#### ■ *Tùng P.M.T*

Từ Samba 2.0 việc tạo file cấu hình thậm chí còn dễ hơn việc viết file đó bằng tay nhờ các trình soạn thảo. Bạn có thể sử dụng bộ trình duyệt của mình để nối vào Web site tại chỗ <http://localhost:901>, và đăng nhập hệ thống với quyền root như được biểu diễn trên H. XVI. 6.

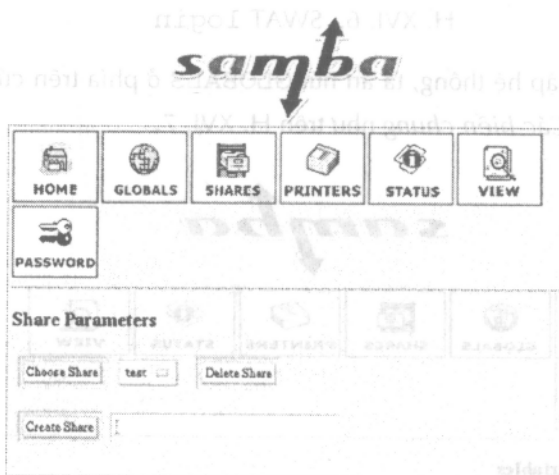
H. XVI. 6. SWAT login

Sau khi đăng nhập hệ thống, ta ấn nút GLOBALS ở phía trên của màn hình. Bạn sẽ nhìn thấy trang *Các biến chung* như trên H. XVI. 7.

H. XVI. 7. Trang *Các biến chung* của SWAT

Tiếp tục xét ví dụ nêu trên, ta đặt tên vào trường nhóm làm việc là `SIMPLE` và vào trường bảo mật là `USER`. Chỉ có một tùy chọn duy nhất khác mà bạn cần phải thay đổi trong thực đơn là xác định xem hệ thống nào là server WINS, có nghĩa là máy chịu trách nhiệm phân giải địa chỉ NetBIOS trong mạng LAN. Tại phần đáy của trang, đặt giá trị `Yes` vào trường hỗ trợ WINS, nếu bạn chưa có server WINS trong mạng của mình. Nếu đã có server WINS, thay vì phải thực hiện động tác trên, bạn điền địa chỉ IP vào trong trường `WINS server`. Sau đó, bạn quay trở lại đầu trang và nhấn nút `Commit Changes` để ghi các thay đổi vào file cấu hình của Samba `smb.conf`.

Tiếp theo, bạn ấn nút biểu tượng `SHARES`. Bạn sẽ thấy một trang tương tự như trên H. XVI. 8. Chọn nút `Test` trong trường nằm cạnh nút `Choose Share`. Bạn lại sẽ thấy một màn hình như trên H. XVI. 9. Ta thêm vào các chú thích để tự nhắc mình đó là sự kiểm tra việc chia sẻ tài nguyên được xác định trong file `smb.conf`, được SWAT sao chép ra đây.



H. XVI. 8. Màn hình tạo sự chia sẻ của SWAT

Nếu bạn nhấn nút `VIEW`, SWAT sẽ cho thấy file cấu hình Samba `smb.conf` như sau:

```
Samba config file created using SWAT
from localhost (127.0.0.1)
```

```
Date: 1998/11/27 15:42:40
```

```
Global parameters
```

```
workgroup = SIMPLE
```

```
[test]
```

```
comment = For testing only, please
```

```
path = /export/samba/test
```

```
read only = no
```

```
guest ok = yes
```

Một khi file cấu hình đã được hoàn thành, bạn có thể bỏ qua bước ngay sau đây, vì thường kết quả ra của SWAT đảm bảo cho ta về mặt cú pháp là file smb.conf được viết đúng.

#### Share Parameters

Choose Share	temp	Delete Share
Create Share	1	
Commit Changes	Reset Values	Advanced View
<b>Base Options</b>		
Help: comment	For testing only, please	Set Default
Help: path	/export/samba/test	Set Default
<b>Security Options</b>		
Help: guest account	nobody	Set Default
Help: read only	Yes <input type="checkbox"/>	Set Default
Help: guest ok	No <input type="checkbox"/>	Set Default

#### H. XVI. 9. Màn hình các thông số chia sẻ của SWAT

##### ■ Kiểm tra file cấu hình cơ bản

Nếu không dùng SWAT để tạo file cấu hình cho Samba, rất có thể bạn phải kiểm tra lại xem file smb.conf đã được viết đúng cú pháp chưa. Có vẻ như thật không đáng phải tốn công kiểm tra file cấu hình chỉ gồm có tám dòng như trên, nhưng thực

tế cho thấy, đối với các file cấu hình `smb.conf` thực thụ, trình tự kiểm tra sau đây là rất có ích.

Bộ phân tích kiểm tra **testparm** sẽ kiểm tra cú pháp của file cấu hình `smb.conf`, và nếu tìm ra lỗi, sẽ báo cho ta thông qua các dịch vụ sẵn có trong máy của bạn. Trong ví dụ sau đây, bạn sẽ thấy đoạn kết quả được đưa ra qua lệnh **tee** khi có lỗi gõ sai "workgroup" thành "workgrp":

```
Load smb config files from smb.conf
Unknown parameter encountered: "workgrp"
Ignoring unknown parameter "workgrp"
Processing section "[test]"
Loaded services file OK.
Press enter to see a dump of your service definitions
Global parameters
[global]
 workgroup = WORKGROUP
 netbios name =
 netbios aliases =
 server string = Samba 2.0.5a
 interfaces =
 bind interfaces only = No ... (bỏ qua một số nội
dung)...
[test]
 comment = For testing only, please
 path = /export/samba/test
 read only = No
 guest ok = Yes
```

Các đoạn thú vị trong kết quả ra nằm ở phần đầu và phần cuối. Phần đầu sẽ đánh dấu mọi lỗi cú pháp mà bạn có thể mắc phải, còn trong phần cuối server đưa ra danh sách các dịch vụ nó có thể đảm bảo. Hãy chọn lựa các yêu cầu của bạn sao cho server có thể đáp ứng được một cách tốt nhất.

Nếu mọi thứ đều được thực hiện suôn sẻ, bây giờ là lúc bạn cho chạy các daemon của Samba.

### ***XVI. 3. d. Khởi động và kiểm tra các daemon Samba***

Chỉ cần chạy hai tiến trình của Samba là **smbd** và **nmbd** là đủ để cho Samba hoạt động tốt. Ta có ba cách sau để khởi động chúng:

- Bằng tay.
- Khởi động như các daemon của máy riêng lẻ (stand-alone).
- Khởi động từ **inetd**.

#### ***▪ Khởi động bằng tay***

Phương pháp khởi động bằng tay thường được áp dụng khi bạn cần khởi động nhanh, ví dụ, khởi động lại sau khi tắt Samba. Để khởi động, với quyền của root, bạn chỉ cần gõ các lệnh sau:

```
/usr/local/samba/bin/smbd -D
/usr/local/samba/bin/nmbd -D
```

Sau đó, Samba sẽ được chạy trên hệ thống của bạn và sẵn sàng tiếp nhận các nối kết.

#### ***▪ Khởi động các daemon cho máy stand-alone***

Để chạy các tiến trình Samba như các daemon của máy stand-alone, bạn cần thêm những dòng lệnh đã nêu ở phần trên vào trong script khởi động UNIX. Các dạng lệnh phụ thuộc vào việc hệ thống UNIX của bạn thuộc loại BSD UNIX hay UNIX System V.

### **BSD Unix**

Với BSD UNIX, bạn cần thêm đoạn mã nguồn sau vào trong nội dung của file **rc.local**, thường nằm trong các thư mục **/etc** hoặc **/etc/rc.d**:

```
if [-x /usr/local/samba/bin/smbd];
then
```

```
 echo "Starting smbd..."
 /usr/local/samba/bin/smbd -D
 echo "Starting nmbd..."
 /usr/local/samba/bin/nmbd -D
fi
```

Đoạn mã nguồn này rất đơn giản, nó kiểm tra xem file `smbd` có quyền chạy không, và nếu có, nó khởi động các daemon của Samba trong thời gian khởi động của hệ thống.

### *System V Unix*

Đối với UNIX System V, các thao tác sẽ phức tạp hơn chút ít. Các bản UNIX System V thường dùng các scripts để khởi động và kết thúc các daemon trong hệ thống. Vì vậy, bạn cần phải chỉ cho Samba biết phải hoạt động thế nào khi nó khởi động và khi kết thúc. Bạn có thể thay đổi nội dung của thư mục `/etc/rc.local` và thêm vào một file có mã nguồn tương tự như file có tên `smb` sau đây:

```
#!/bin/sh
Contains the "killproc" function on Red Hat Linux
./etc/rc.d/init.d/functions
PATH="/usr/local/samba/bin:$PATH" case $1 in
'start')
 echo "Starting smbd..."
 smbd -D
 echo "Starting nmbd..."
 nmbd -D
 ;;
'stop')
 echo "Stopping smbd and nmbd..."
 killproc smbd
 killproc nmbd
 rm -f /usr/local/samba/var/locks/smbd.pid
 rm -f /usr/local/samba/var/locks/nmbd.pid
```

```
;;
*)
 echo "usage: smb {start|stop}"
;;
esac
```

Bằng script này, bạn có thể khởi động và kết thúc các dịch vụ SMB thông qua các lệnh như sau:

```
/etc/rc.local/smb start
Starting smbd...
Starting nmbd...
/etc/rc.local/smb stop
Stopping smbd and nmbd...
```

#### ▪ *Khởi động từ **inetd***

Daemon **inetd** là "super daemon" về Internet của một hệ thống UNIX. Nó "nghe" các cổng TCP được xác định trong file `/etc/services` và thực hiện các chương trình thích hợp cho từng cổng như quy định trong file `/etc/inetd.conf`. Ưu điểm của sơ đồ này nằm ở chỗ bạn có thể có một số lớn các daemon sẵn sàng đáp ứng các truy vấn, nhưng nếu chưa cần đến thì chúng chưa được chạy. Thay vào đó, daemon **inetd** sẽ làm nhiệm vụ trực thay chúng. Kết quả là ta có thể giảm tải cho máy, tức là làm giảm giá thành vận hành hệ thống, và cần phải sửa hai file thay vì một file để thiết lập một chức năng nào đó. Điều này sẽ rất tiện lợi, nếu bạn chỉ có một, hai người dùng trong hệ thống hoặc khi máy của bạn có quá nhiều daemon chạy. Và ta còn có thể thực hiện nâng cấp một cách dễ dàng mà không ảnh hưởng tới các nối kết đang tồn tại.

Nếu muốn khởi động Samba từ **inetd**, đầu tiên bạn hãy mở file `/etc/services` bằng trình soạn thảo bạn quen dùng. Nếu bạn chưa thấy có trong nội dung của file đó, hãy thêm vào hai dòng sau:

```
netbios-ssn 139/tcp
```



---

```
netbios-ns 137/udp
```

Sau đó, bạn phải sửa file `/etc/inetd.conf`. Hãy tìm xem đã có hai dòng lệnh sau đây hay chưa; nếu chưa có thì bạn thêm chúng vào. Nếu bạn đã có các dòng vào `smbd` và `nmbd` trong nội dung file, hãy sửa chúng để chỉ tới các daemon **smbd** và **nmbd** mới mà bạn vừa cài đặt. Cú pháp của các câu lệnh có thể khác nhau chút ít, phụ thuộc vào bản UNIX cụ thể mà bạn sử dụng, nhưng tốt nhất là bạn nên dựa trên các dòng vào đã sẵn có và trang **man** của `inetd.conf` để làm mẫu:

```
netbios-ssn stream tcp nowait root
 /usr/local/samba/bin/smbd smbd
netbios-ns dgram udp wait root
 /usr/local/samba/bin/nmbd nmbd
```

Cuối cùng, hãy diệt mọi tiến trình **smbd** hoặc **nmbd** và gửi tới tiến trình **inetd** tín hiệu gác máy (HUP). (Daemon **inetd** đọc lại file cấu hình của nó khi có tín hiệu HUP.) Ta đã biết cách thực hiện các thao tác trên: dùng lệnh **ps** để tìm định danh tiến trình PID của nó, sau đó gửi tín hiệu tới tiến trình bằng lệnh:

```
kill -HUP process_id
```

Sau đó Samba sẽ được bật lên lại và hoạt động.

#### ■ *Kiểm tra các daemon Samba*

Đến giai đoạn này hầu như bạn đã thiết lập xong server Samba trên hệ thống của bạn. Tất cả công việc còn lại chỉ là kiểm tra xem mọi thứ có hoạt động như yêu cầu được đề ra từ đầu hay không. Cách thích hợp nhất là dùng chương trình **smbclient** để kiểm tra xem server đưa ra các dịch vụ gì cho mạng. Nếu mọi thứ đều được thiết lập đúng cách, bạn sẽ có thể đọc được những thông tin như sau:

```
smbclient -U% -L localhost Added interface
 ip=192.168.220.100 bcast=192.168.220.255
 nmask=255.255.255.0
```

---



---

```
Domain=[SIMPLE] OS=[Unix] Server=[Samba 2.0.5a]
```

Sharename	Type	Comment
-----	----	-----
test	Disk	For testing only, please
IPC\$	IPC	IPC Service (Samba 2.0.5a)

Server	Comment
-----	-----
HYDRA	Samba 2.0.5a
Workgroup	Master
-----	-----
SIMPLE	HYDRA

Nếu không đọc được các dòng trên, hay nói cách khác là gặp phải rắc rối, bạn nên bình tĩnh và cố gắng khởi động các daemon bằng tay, rồi kiểm tra xem cụ thể vấn đề gì đã xảy ra trong kết quả ra hiện trên màn hình hay trong các file ghi rỗi tại `/usr/local/samba/var/log.smb`.

Cuối cùng, nếu mọi thứ đều ổn thỏa, bạn đã có server Samba hoạt động với đĩa cứng được chia sẻ. Bây giờ bạn phải thiết lập tiếp các client trên các máy Windows 9x và Windows NT. Sau đó, để hoàn thiện hệ thống, bạn sẽ phải thêm các dịch vụ khác nữa, như các thư mục home, máy in, bảo mật hoặc tích hợp server Samba vào mạng vùng lớn hơn của Windows.

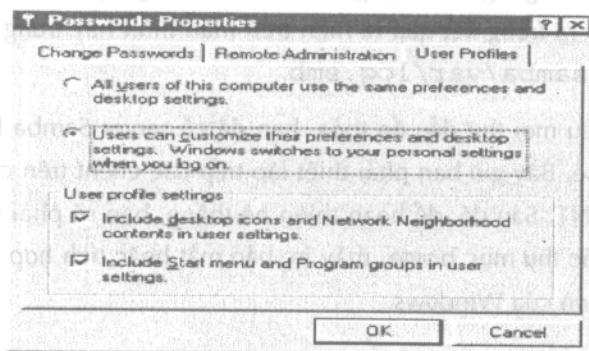
## **XVI. 4. Cấu hình các client Windows**

### ***XVI. 4. a. Thiết lập các client Windows 9x***

Ta đã biết Windows 95/98 không được thiết kế để cho PC có nhiều hơn một người dùng. Có chăng, Windows 95/98 chỉ giữ các bố trí màn hình làm việc và file mật khẩu cho từng người dùng khác nhau trong cái gọi là *profile* - điều này khác xa so với khái niệm bảo mật nhiều người dùng của UNIX. Nói cách khác, Windows 95/98 không thể để cho một người dùng sử dụng máy mà không làm ảnh hưởng tới công việc của người dùng khác đang thực hiện trên đĩa cứng tại chỗ.

### ▪ Tài khoản người dùng và mật khẩu

Việc đầu tiên mà ta phải thực hiện là tìm cách làm cho Windows giữ các profile của người dùng một cách riêng rẽ, và tập hợp các tên người dùng và mật khẩu để kiểm tra xác nhận người dùng nào đang tìm cách đăng nhập vào phần tài nguyên được chia sẻ trên Samba server. Ta sẽ thực hiện điều đó thông qua việc thiết lập *Password* trong cửa sổ *Control Panel*. Sau khi chọn biểu tượng *Passwords* trong *Control Panel*, ta nháy phím trái chuột vào bảng *User Profiles* và thấy hộp thoại như H. XVI. 10. Sau đó, nháy chuột vào nút phía dưới bên cạnh hàng chữ bắt đầu bằng "Users can customize their preferences ...". Như vậy, Windows sẽ giữ các profile riêng rẽ cho mỗi một người dùng với tên và mật khẩu đã được xác định và sẽ được sử dụng sau này khi kết nối với SMB/CIFS server. Cuối cùng, ta đánh dấu vào cả hai hộp kiểm tra để xác định các tùy chọn trong khung *User Profile Settings* như được chỉ trên hình H. XVI. 10.



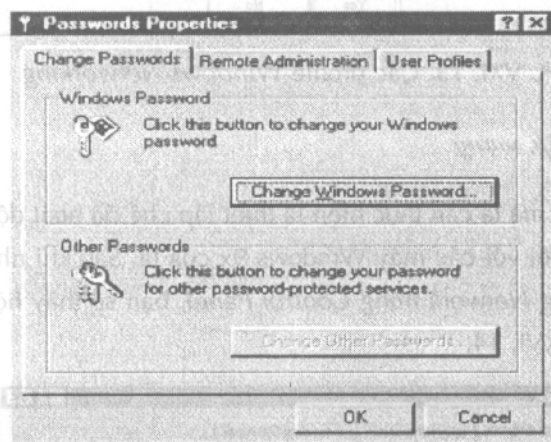
H. XVI. 10. Bảng *Properties*

Bước tiếp theo là chọn bảng *Change Passwords* ở phần bên trái của hộp thoại. Để Samba có thể cho phép bạn truy cập đến các tài nguyên được chia sẻ của nó, tên người dùng và mật khẩu mà bạn đưa vào phải trùng với tên và mật khẩu đăng ký trong Samba server. Nếu không nhìn thấy bảng *Change Password* trong hộp thoại, rất có thể bạn đã không tự đặt tên người dùng và mật khẩu cho mình trong Windows. (Bình thường, để tiện lợi, nhiều người dùng Windows hay bỏ qua động tác kiểm tra tên người dùng / mật khẩu cho đỡ tốn thời gian, và lại, xét về mặt bảo mật thì động tác trên không có ý nghĩa gì nhiều). Nếu thế, bạn chỉ cần nháy chuột trái

vào nút OK ở phía dưới và trả lời Yes khi Windows hỏi để tự khởi động lại. Sau đó, thực hiện các thao tác như trong mục nhỏ “Đăng nhập lần đầu tiên” được trình bày sau đây.

### Thay đổi mật khẩu Windows

Sau khi chọn bảng *Change Passwords*, hộp thoại như trên H. XVI. 11 sẽ xuất hiện. Ta hãy chọn nút *Change Windows Password*. Hộp thoại *Change Windows Password* xuất hiện trên màn hình như trong H. XVI. 12 và bạn có thể thay đổi mật khẩu của mình cho trùng với mật khẩu của tài khoản người dùng trong server Samba mà bạn muốn sử dụng để đăng nhập hệ thống.

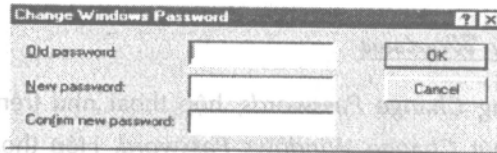


H. XVI. 11. Bảng *Change Passwords*

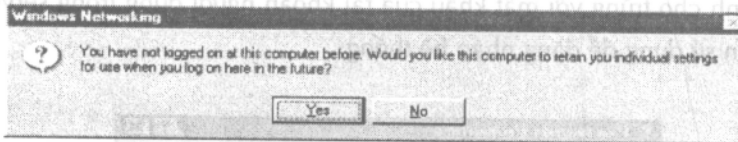
### Đăng nhập lần đầu tiên

Nếu bạn chưa thay đổi gì trong bảng *Change Passwords* của cửa sổ *Passwords Properties*, và sau khi Windows đã kết thúc quá trình khởi động lại, bạn sẽ được yêu cầu điền tên tài khoản người dùng và mật khẩu để đăng nhập vào. Bạn sẽ đưa vào tên của người dùng và mật khẩu đã đăng ký trong Samba server, và Windows sẽ hỏi bạn có muốn có riêng một *profile* không bằng hộp thoại như trên H. XVI. 13. Nếu bạn trả lời Yes, Windows sẽ tạo riêng một *profile* và file mật khẩu cho bạn. Windows gửi tới Samba mật khẩu của nó khi bạn kết nối với Samba để kiểm tra xác nhận người dùng cho mỗi một tài nguyên chia sẻ được sử dụng. Tất nhiên, cơ chế

bảo mật của Windows 9x là yếu và không thể so sánh được với Windows 2000 (NT 5.0).



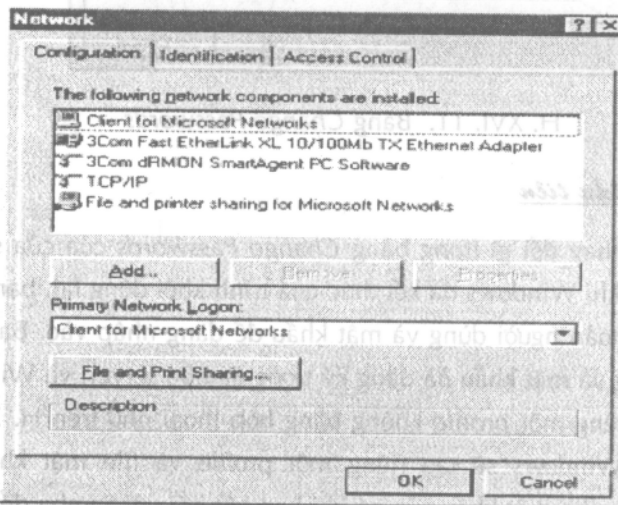
H. XVI. 12. Hộp thoại *Change Windows Password*



H. VXI. 13. Các profile *Windows Networking*

#### ▪ *Thiết lập mạng*

Động tác thứ hai mà ta cần thực hiện là thiết lập chế độ hoạt động đúng của giao thức mạng TCP/IP đối với các máy Windows 9x của ta. Sau khi nhấn phím trái chuột vào biểu tượng *Network* trong *Control Panel*, bạn sẽ thấy hộp thoại cấu hình mạng như trong H. XVI. 14.



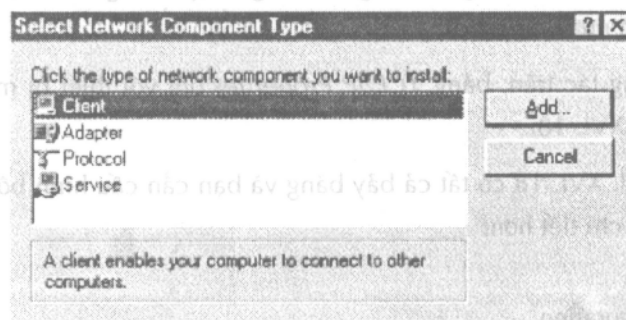
H. XVI. 14. Bảng *Network* của Windows 95/98

Mạng Microsoft làm việc theo kiểu nối kết các giao thức cụ thể như IPX hoặc TCP/IP tới các thiết bị phần cứng cụ thể như card Ethernet (card mạng thường dùng) hoặc nối kết quay số điện thoại. Bằng cách phân đường cho giao thức qua thiết bị phần cứng, máy tính có thể hoạt động như một client hoặc server cho một kiểu mạng cụ thể. Đối với Samba, chúng ta quan tâm tới việc nối kết giao thức TCP/IP qua thiết bị mạng để làm cho máy tính trở thành client trong mạng Microsoft. Trong hộp thoại, ít nhất bạn cũng sẽ thấy thành phần phần mềm *Client for Microsoft Networks* được cài đặt trên máy và các thiết bị mạng, chủ yếu là một card Ethernet, được nối đến giao thức TCP/IP.

Bạn cũng có thể nhìn thấy và nên chọn cả mục "*File and printer sharing for Microsoft Networks*" vì chắc chắn sẽ phải cần đến nó. Thêm nữa, bạn còn nhìn thấy cả các mạng *NetBEUI* hoặc *Novell Networking*, là các thành phần của bộ cài đặt Windows chuẩn, nhưng có thể không cần thiết khi chạy TCP/IP. Hãy loại bỏ NetBEUI nếu có thể, và nếu trong mạng LAN của bạn không có một server NetWare nào thì cũng nên xóa nốt cả Novell (giao thức IPX/SPX).

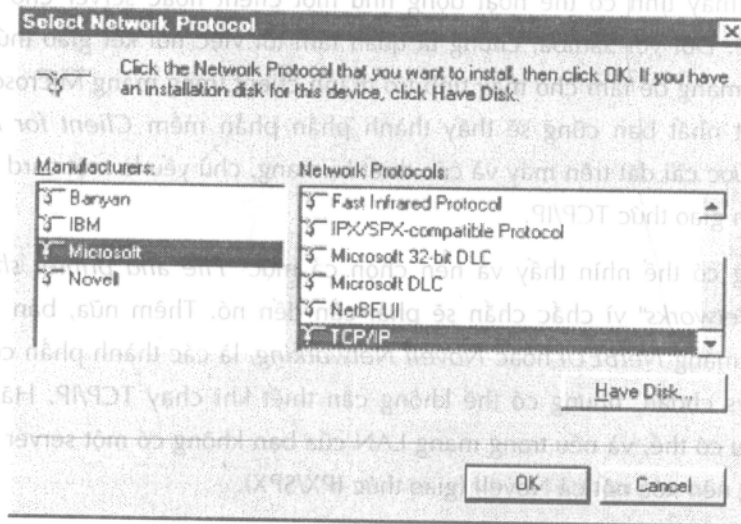
### Thêm giao thức TCP/IP

Nếu TCP/IP chưa được cài đặt, bạn có thể thực hiện những bước sau đây. Nếu đã có TCP/IP trên máy, bạn có thể bỏ qua mục này. TCP/IP đã có sẵn trong bộ cài đặt chuẩn của Windows và bạn có thể thêm giao thức này bằng cách nhấp chuột vào nút *Add* dưới cửa sổ các thành phần mạng trong H. XVI. 14. Bạn hãy chỉ định là cần thêm vào các giao thức bằng nút *Protocol*, sau đó là nút *Add...* trên hộp thoại H. XVI. 15.



H. XVI. 15. Chọn cài đặt giao thức

Sau đó, bạn chọn giao thức TCP/IP từ nhà cung cấp Microsoft, như trên H. XVI. 16, rồi ấn nút *OK*. Sau đó, bạn quay trở lại hộp thoại mạng H. XVI. 14, dùng nút *OK* đóng nó lại, để Windows cài đặt giao thức và khởi động lại máy.



H. XVI. 16. Chọn giao thức mạng TCP/IP

### Cấu hình TCP/IP

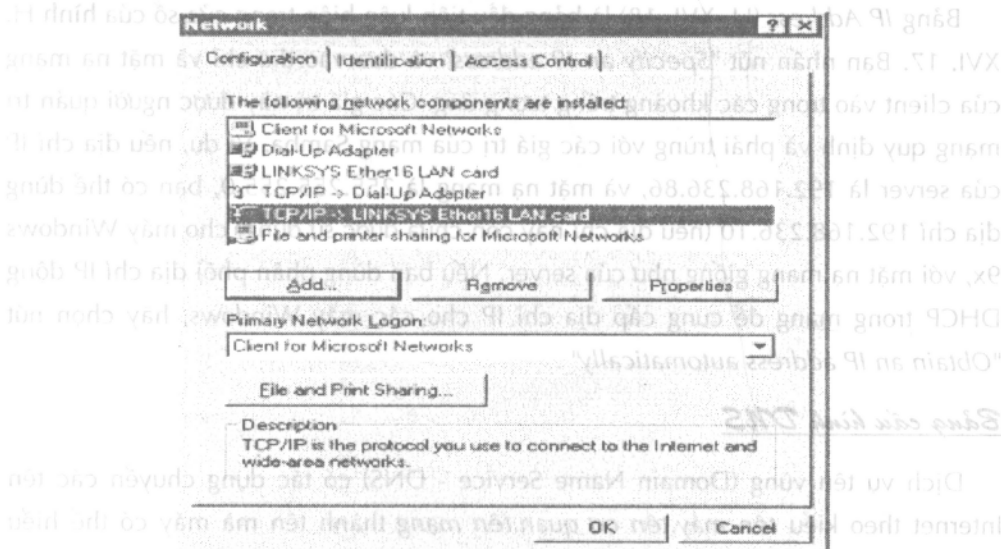
Nếu dùng nhiều hơn một thiết bị mạng (ví dụ cả card Ethernet và modem quay số điện thoại), mỗi một thiết bị phần cứng trên cần được “liên kết” với giao thức TCP/IP với ký hiệu một mũi tên như trong H. XVI. 17. Hãy chọn giao thức TCP/IP đã được liên kết với thiết bị mạng sẽ dùng để đăng nhập mạng Samba, và khi nó đã được đánh dấu, ấn nút *Properties*.

Sau những động tác trên, bảng *TCP/IP Properties* đối với thiết bị mạng đã chọn sẽ hiện ra trên H. XVI. 18.

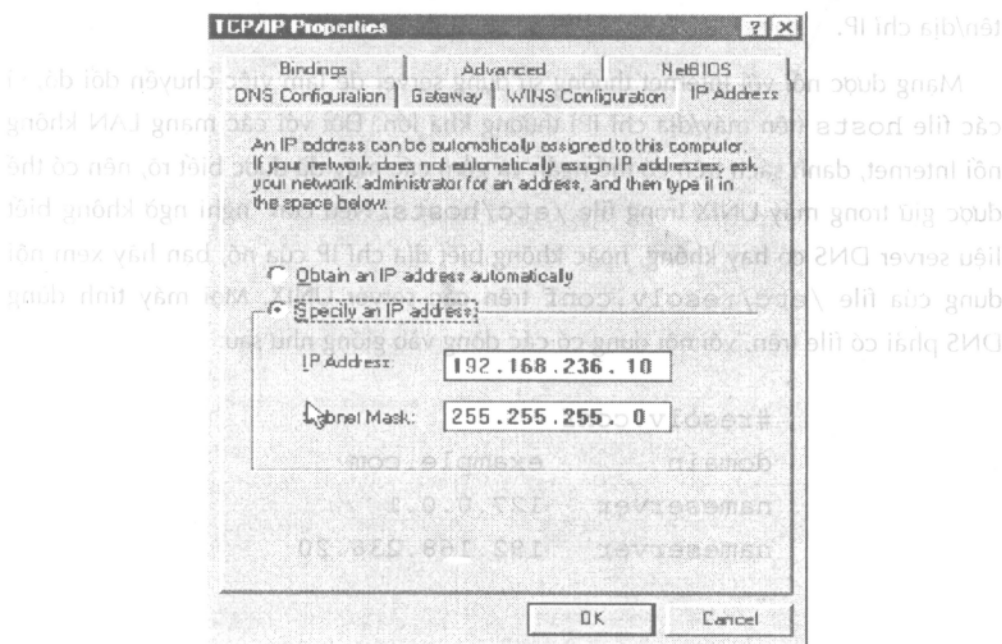
Trên hình vẽ H. XVI. 18 có tất cả bảy bảng và bạn cần cấu hình bốn bảng trong số đó mà ta sẽ xét chi tiết hơn:

- IP address
- DNS configuration
- WINS configuration

## – Bindings



H. XVI. 17. Chọn đúng giao thức TCP/IP



H. XVI. 18. Bảng TCP/IP Properties cho thiết bị mạng cụ thể



### **Bảng địa chỉ IP**

Bảng *IP Address* (H. XVI. 18) là bảng đầu tiên luôn hiện trong cửa sổ của hình H. XVI. 17. Bạn nhấn nút "*Specify an IP address*" và đưa vào địa chỉ và mặt nạ mạng của client vào trong các khoảng trống tương ứng. Các giá trị này được người quản trị mạng quy định và phải trùng với các giá trị của mạng Samba. Ví dụ, nếu địa chỉ IP của server là 192.168.236.86, và mặt nạ mạng là 255.255.255.0, bạn có thể dùng địa chỉ 192.168.236.10 (nếu địa chỉ này còn chưa được sử dụng) cho máy Windows 9x, với mặt nạ mạng giống như của server. Nếu bạn dùng phân phối địa chỉ IP động DHCP trong mạng để cung cấp địa chỉ IP cho các máy Windows, hãy chọn nút "*Obtain an IP address automatically*".

### **Bảng cấu hình DNS**

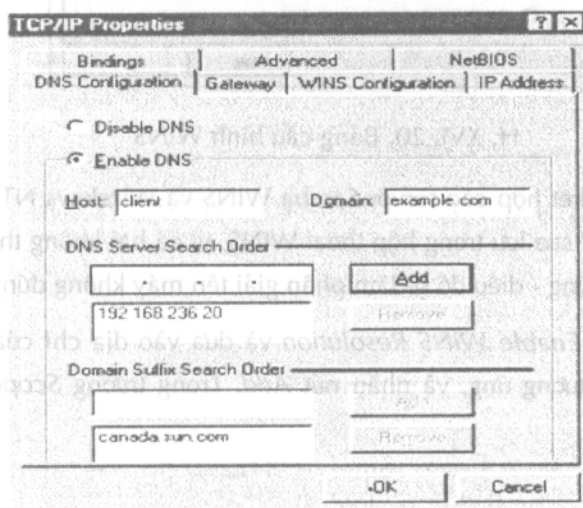
Dịch vụ tên vùng (Domain Name Service - DNS) có tác dụng chuyển các tên Internet theo kiểu *tên máy.tên cơ quan.tên mạng* thành tên mà máy có thể hiểu được là địa chỉ IP như 192.168.236.10. Ta có hai cách để thực hiện điều đó: hoặc để server chuyển tên cho các máy client, hoặc giữ trong các máy tại chỗ danh sách tên/địa chỉ IP.

Mạng được nối với Internet thường sử dụng server để làm việc chuyển đổi đó, vì các file *hosts* (tên máy/địa chỉ IP) thường khá lớn. Đối với các mạng LAN không nối Internet, danh sách trên có thể ngắn và gồm các máy đã được biết rõ, nên có thể được giữ trong máy UNIX trong file */etc/hosts*. Nếu còn nghi ngờ không biết liệu server DNS có hay không, hoặc không biết địa chỉ IP của nó, bạn hãy xem nội dung của file */etc/resolv.conf* trên các server UNIX. Mọi máy tính dùng DNS phải có file trên, với nội dung có các dòng vào giống như sau:

```
#resolv.conf
domain example.com
nameserver 127.0.0.1
nameserver 192.168.236.20
...
```

Trong ví dụ vừa nêu, dòng vào nameserver thứ hai trong danh sách chứa địa chỉ IP của một máy khác trong mạng tại chỗ: 192.168.236.20. Có thể đó chính là server DNS.

Bạn cần phải đưa vào đúng các địa chỉ IP của một hay nhiều server DNS (chú ý rằng không được dùng tên vùng kiểu *tên vùng.kiểu tên máy.tên cơ quan.tên mạng*, vì các máy sẽ không hiểu được) vào các trường tương ứng trong cửa sổ được biểu diễn trên H. XVI. 19. Tất nhiên, bạn không thể dùng địa chỉ IP 127.0.0.1, vì địa chỉ này được dùng cho các tiện ích đối với chính bản thân từng máy chủ. Mọi máy chủ được kể tên trong `/etc/resolv.conf` đều có thể dùng làm server DNS, nhưng bạn nên dùng máy chủ nào gần nhất thì tốt hơn. Giả sử tiếp trong ví dụ ta vẫn dùng là chỉ tồn tại một server DNS với địa chỉ IP là 192.168.236.20. Nhấn nút *Enable DNS* như trên H. XVI. 19, và thêm địa chỉ IP vào trường *DNS Server Search Order* ở phía trên (có dùng phím lệnh *Add*).

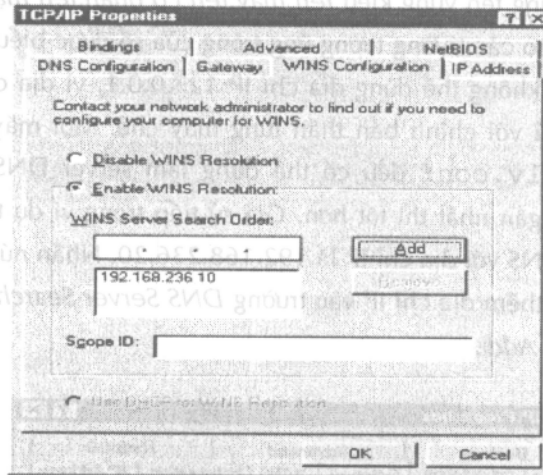


H. XVI. 19. Bảng cấu hình DNS

Thêm nữa, bạn nên cung cấp thêm tên của máy Windows 95/98 và tên vùng Internet mà bạn đang tham gia. Đối với Samba, có thể bỏ qua trường *Domain Suffix Search Order*.

### Bảng cấu hình WINS

Nếu WINS đã được kích hoạt trên Samba, bạn cần báo cho máy Windows địa chỉ của server Samba. Nếu các server WINS toàn là các máy Windows NT, ta phải thêm vào từng tên của chúng như trong hộp thoại cấu hình WINS của H. XVI. 20.



H. XVI. 20. Bảng cấu hình WINS

Chú ý: Không được kết hợp các server Samba WINS và Windows NT như là tổ hợp các máy chủ chính / sao lưu trong hộp thoại WINS, vì cả hai không thể sao lưu được cơ sở dữ liệu của chúng - điều đó sẽ làm phân giải tên máy không đúng.

Bây giờ ta chọn *Enable WINS Resolution* và đưa vào địa chỉ của server WINS trong khoảng trống tương ứng, và nhấn nút *Add*. Trong trường *Scope ID* ta không phải điền gì.

### Các file host

Nếu ta không dùng cả DNS lẫn WINS, và cũng không muốn dùng cả việc phân giải địa chỉ thông báo công cộng, ta có thể cung cấp một bảng các địa chỉ IP và tên máy trong dạng UNIX chuẩn `/etc/hosts`. Trên máy Windows, danh sách đó nằm trong `\WINDOWS\HOSTS`, ở thư mục cài đặt Windows, thường là `C:\`. Một file host mẫu có dạng như sau:

```
127.0.0.1
```

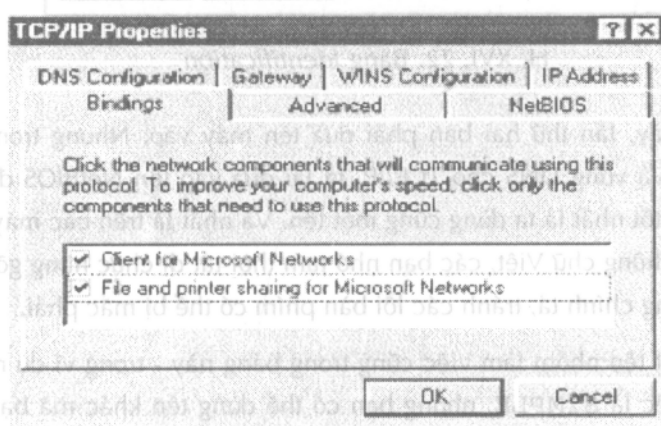
```
localhost
```

192.168.236.1	tuan.example.com	tuan
192.168.236.2	cuong.example.com	cuong
192.168.236.3	minh.example.com	minh
192.168.236.4	chi.example.com	chi
192.168.236.10	chuyen.example.com	chuyen

Bạn có thể sao file này thẳng từ file `/etc/hosts` bất kỳ trên máy UNIX; định dạng của chúng giống nhau. Tuy nhiên, bạn chỉ nên dùng các file hosts trong Windows như là biện pháp cuối cùng để phân giải các tên máy.

### Kiểm tra các nối kết

Bảng cuối cùng ta phải cấu hình có tên là *Bindings*, như trên hình H. XVI. 21.

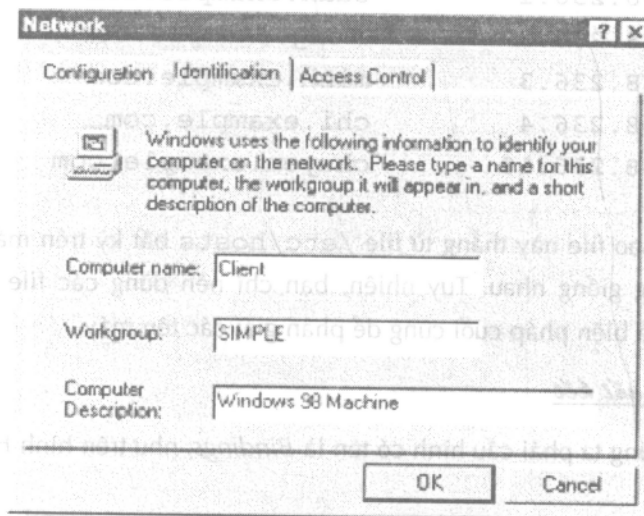


H. XVI. 21. Bảng *Bindings*

Bạn phải kiểm tra *Client for Microsoft Networks*, xem có dùng TCP/IP không. Nếu bạn có cả dòng "*File and printer sharing for Microsoft Networks*" trong hộp thoại, nó cũng cần phải được kiểm tra như ta thấy trong hình vẽ.

### ■ *Thiết lập tên và nhóm làm việc*

Cuối cùng, bạn nhấn nút *OK* trên bảng cấu hình TCP/IP, và bạn được đưa trở lại màn hình *Network Configuration*. Sau đó chọn bảng *Identification*, để có được hộp thoại như trên H. XVI. 22.

H. XVI. 22. Bảng *Identification*

Trong bảng này, lần thứ hai bạn phải đưa tên máy vào. Nhưng trong lần này, thay vì tên máy và vùng DNS cho TCP/IP, ta lại đưa vào tên NetBIOS để dùng với SMB. Tuy nhiên, tốt nhất là ta dùng cùng một tên. Và nhất là trên các máy Windows 9x có cài bộ gõ phonetic chữ Việt, các bạn nhớ tạm thời tắt đi chức năng gõ tiếng Việt để gõ tên cho đúng chính tả, tránh các lỗi bàn phím có thể bị mắc phải.

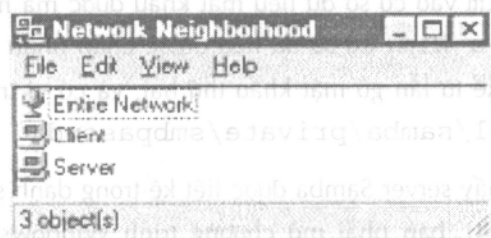
Bạn có thể đặt tên nhóm làm việc cũng trong bảng này - trong ví dụ mà ta dùng tên nhóm làm việc là **SIMPLE**, nhưng bạn có thể dùng tên khác mà bạn chỉ định khi tạo file cấu hình Samba để cài đặt Samba trên máy UNIX. Bạn nên tránh dùng tên **WORKGROUP** để khỏi sa vào cùng một nhóm làm việc với tất cả các máy Windows không được cấu hình hoặc bị cấu hình sai trên thế giới.

#### ■ *Tung cấp server Samba*

Sau khi đã thực hiện tất cả các bước trên, ta nhấn nút **OK** để hoàn thành việc cấu hình, sau đó, khởi động lại máy để Windows chấp nhận các thay đổi đã được đưa vào.

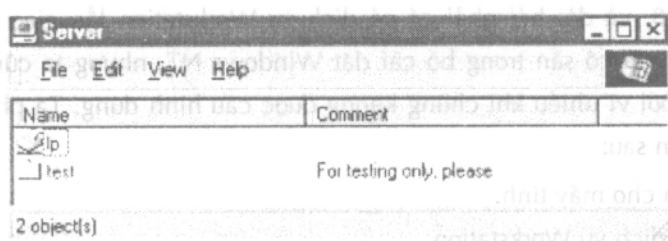
Server Samba của bạn đã chạy trên máy UNIX, bây giờ bạn cần thiết lập máy client Windows 9x để trao đổi với nó. Sau khi khởi động lại máy, bạn đăng nhập

vào máy Windows và nháy đúp chuột trái vào biểu tượng *Network Neighborhood* trên màn hình làm việc. Bạn sẽ nhìn thấy server Samba của bạn được kê trong danh sách là một thành viên của nhóm làm việc như được minh họa trên H. XVI. 23.



H. XVI. 23. Bảng *Network Neighborhood* của Windows

Nháy đúp chuột trái vào tên *Server* được chỉ trên H. XVI. 24, ta sẽ có danh sách các dịch vụ mà server Samba có thể đưa ra cho mạng. Trong ví dụ ta vẫn theo dõi, như trên H. XVI. 24 đã cho thấy, các tài nguyên server Samba được đưa ra dùng chung là máy in và thư mục test.



H. XVI. 24. Tài nguyên được chia sẻ trên Server

**Chú ý quan trọng:** Nếu bạn gặp phải hộp thoại đòi hỏi mật khẩu cho người dùng IPC\$, sau đó Samba không chấp nhận mật khẩu được gửi tới từ phía máy client, bạn phải kiểm tra lại xem tên người dùng và mật khẩu ở phía máy client Windows có giống như ở phía server Samba hay không. Nếu bạn dùng Windows 98 hoặc Windows NT Service Pack 3 trở đi, điều đó xảy ra có thể là do máy client đã gửi đi mật khẩu đã mã hóa thay vào chỗ của mật khẩu dạng văn bản thuần túy. Bạn có thể khắc phục tình trạng đó bằng hai bước làm trên server Samba. Thứ nhất, thêm dòng

vào sau vào phần [global] trong file cấu hình Samba: `encrypt password=yes`. Thứ hai, tìm chương trình **smbpasswd** trên server Samba (thư mục mặc định của nó là `/usr/local/samba/bin`) và dùng nó để thêm một dòng vào trong cơ sở dữ liệu mật khẩu được mã hóa của Samba. Ví dụ, để thêm người dùng có tên kim vào cơ sở dữ liệu mật khẩu được mã hóa của Samba, ta gõ lệnh **smbpasswd -a kim**. Cơ sở dữ liệu đó sẽ chấp nhận tên và mật khẩu mã hóa của người dùng kể từ lần gõ mật khẩu thứ hai, và có vị trí xác định theo mặc định tại `/usr/local/samba/private/smbpasswd`.

Nếu không nhìn thấy server Samba được liệt kê trong danh sách các thành phần của mạng (H. XVI. 24), bạn phải mở chương trình Windows Explorer (nhớ rằng không phải là Internet Explorer!) và chọn *Map Network Drive* từ thực đơn *Tools*. Bạn sẽ có một hộp thoại, trong đó, bạn phải đưa tên của server Samba và thư mục `test` được chia sẻ vào dưới dạng thường thấy trong Windows: `\\server\test` để cố kết nối với server Samba và các tài nguyên được tạm thời chia sẻ của nó.

#### ***XVI. 4. b. Thiết lập các máy Windows NT***

Cấu hình các máy Windows NT cho Samba hơi khác chút ít so với các máy Windows 95/98, và đòi hỏi phải có cả dịch vụ Workstation lẫn giao thức TCP/IP. Hai yếu tố này đã có sẵn trong bộ cài đặt Windows NT, nhưng ta cũng nên điem qua lại ở đây bởi vì nhiều khi chúng không được cấu hình đúng. Ta phải thực hiện sáu bước cơ bản sau:

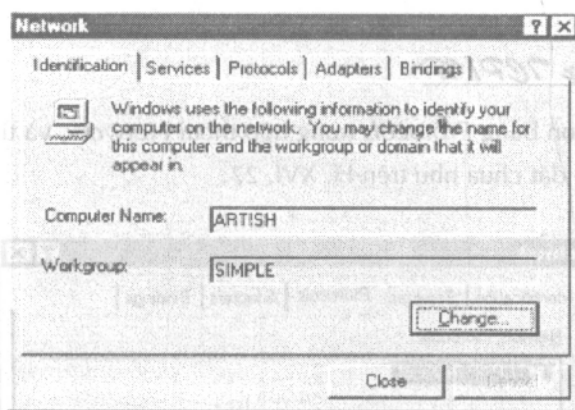
- Gán tên cho máy tính.
- Cài đặt dịch vụ Workstation.
- Cài đặt giao thức TCP/IP.
- Thiết lập tên và địa chỉ IP cho máy.
- Cấu hình các dịch vụ tên DNS và WINS.
- Liên kết các giao thức và dịch vụ với nhau.

##### **■ *Cấu hình cơ bản***

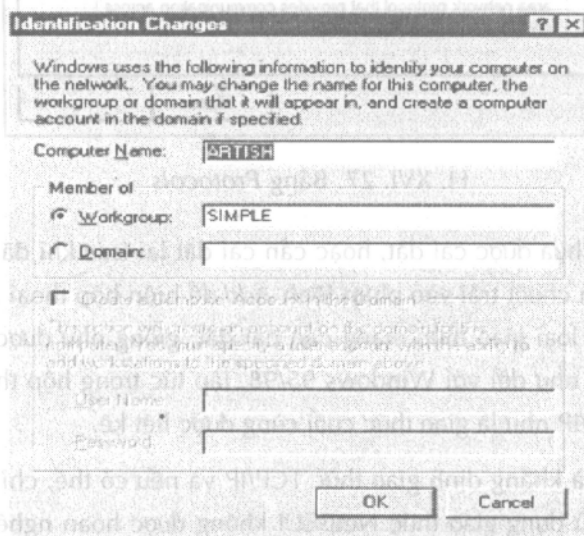
Phần này trình bày các bước tối cần thiết để làm cho Windows NT làm việc được với Samba. Các bước thao tác quản trị mạng TCP/IP được liệt kê dưới đây phải được thực hiện với quyền của người dùng "Administrator".

### Gán tên cho máy tính

Việc đầu tiên bạn phải làm là gán cho máy tính một tên NetBIOS. Từ *Control Panel* ta nhấp đúp chuột trái vào biểu tượng *Network* và có được hộp thoại *Network* với bảng đầu tiên là bảng *Identification*, như được minh họa trên H. XVI.25.



H. XVI.25. Bảng *Identification* của hộp thoại *Network*



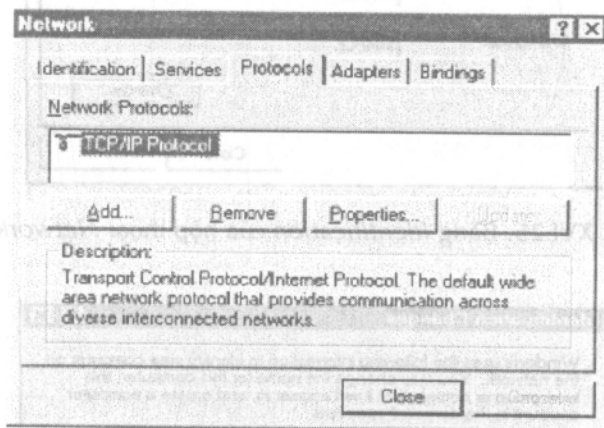
H. XVI. 26. Thay đổi định danh của máy tính



Trong bảng này bạn phải điền tên máy - trong ví dụ là *Artish* - và thay đổi tên nhóm làm việc mặc định bằng tên bạn đã xác định trong file *smb.conf* trên server Samba của bạn, ví dụ là *SIMPLE*. Tuy nhiên, trong phạm vi bảng này bạn chỉ có thể gõ các tên vào chứ không sửa được chúng như trong Windows 95/98. Muốn sửa, bạn phải dùng nút *Change* để vào hộp thoại *Identification Changes* như trên H. XVI. 26.

### Cài đặt giao thức TCP/IP

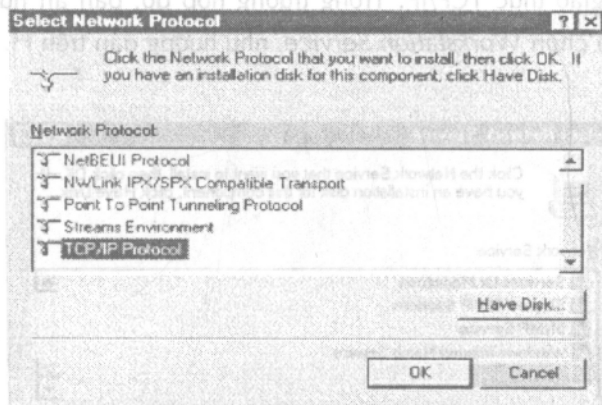
Tiếp theo, ta chọn bảng *Protocols* trong hộp thoại *Network*, và tìm xem giao thức TCP/IP đã được cài đặt chưa như trên H. XVI. 27.



H. XVI. 27. Bảng *Protocols*

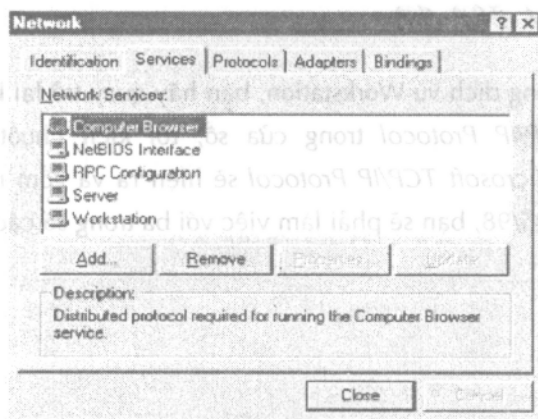
Nếu giao thức chưa được cài đặt, hoặc cần cài đặt lại (sau khi đã loại bỏ bản cài đặt trước), bạn kích chuột trái vào phím lệnh *Add* để hiện hộp thoại *Select Network Protocol* cho chọn loại giao thức mạng cần cài đặt, giống như được biểu diễn trên H. XVI. 28. Không như đối với Windows 95/98, lập tức trong hộp thoại trên bạn sẽ thấy giao thức TCP/IP như là giao thức cuối cùng được liệt kê.

Bạn hãy chọn và khẳng định giao thức TCP/IP và nếu có thể, chỉ dùng giao thức đó mà thôi. Việc sử dụng giao thức NetBEUI không được hoan nghênh cho lắm, vì hiệu năng của máy tính bị giảm đi do máy phải tìm kiếm và giám sát các dịch vụ được chạy dưới hai giao thức khác nhau, nhưng thực tế chỉ có một trong số đó là được sử dụng.

H. XVI. 28. Chọn hộp thoại *Select Network Protocol*

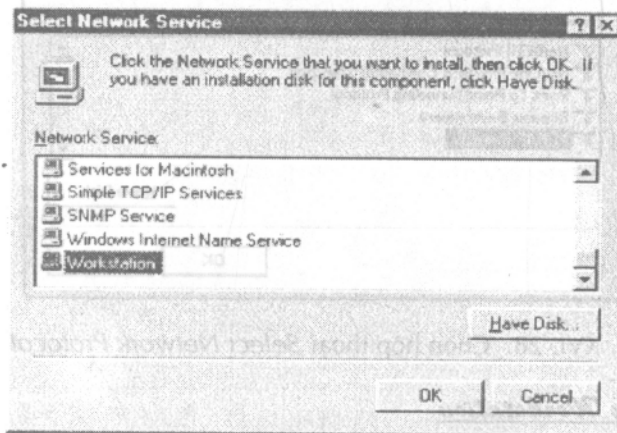
### Cài đặt dịch vụ Workstation

Sau khi đã cài đặt TCP/IP, ta nhấn nút chuột vào bảng *Services* trong khung *Network* để kiểm tra dịch vụ Workstation có được cài đặt hay chưa, tương tự như trên H. XVI. 29.

H. XVI. 29. Hộp thoại dịch vụ *Services* trong khung *Network*

Dịch vụ đó thực chất là Microsoft Networking Client, phần mềm cho phép các máy tính có thể truy cập các dịch vụ SMB. Dịch vụ Workstation là bắt buộc phải có và được cài đặt một cách mặc định trên cả Windows NT Workstation 4.0 và Server 4.0 trở đi. Nếu không thấy có hay phải xóa đi dịch vụ đó để cài đặt lại, bạn có thể

làm như đối với giao thức TCP/IP. Trong trường hợp đó, bạn ấn nút chuột trái vào phím lệnh *Add* và chọn *Workstation Service*, như hướng dẫn trên H. XVI. 30.



H. XVI. 30. Hộp thoại *Select Network Service*

### ▪ Cấu hình TCP/IP

Sau khi cài đặt xong dịch vụ *Workstation*, bạn hãy quay trở lại bảng *Protocols* và chọn dòng vào *TCP/IP Protocol* trong cửa sổ, rồi kích chuột vào phím lệnh *Properties*. Khung *Microsoft TCP/IP Protocol* sẽ hiện ra và gồm năm bảng. Giống như trong Windows 95/98, bạn sẽ phải làm việc với ba trong số các bảng đó:

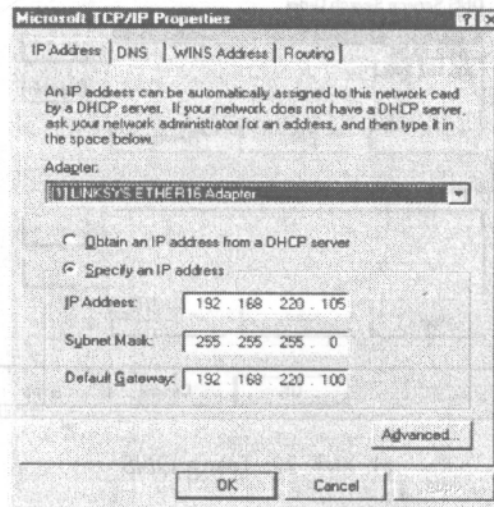
- IP address
- DNS
- WINS address

### Bảng địa chỉ IP

Bảng *IP Address* là bảng đầu tiên luôn hiện trên khung thuộc tính *TCP/IP* và được chụp trên H. XVI. 31 đối với ví dụ ta vẫn theo dõi từ trước đến nay.

Ta đánh dấu vào nút "*Specify an IP address*", sau đó điền vào địa chỉ IP của máy tính và mặt nạ mạng trong các khoảng trống tương ứng dành cho thiết bị mạng (card Ethernet). Người quản trị mạng phải chọn địa chỉ IP sao cho máy client nằm trong

cùng một mạng con (LAN) với server Samba. Ví dụ, nếu địa chỉ IP của server Samba ta dùng là 192.168.236.86 và mặt nạ mạng của nó là 255.255.255.0, bạn có thể chọn cho máy client là trạm làm việc Windows NT Workstation địa chỉ IP 192.168.236.10, nếu địa chỉ này còn trống, và kèm theo là mặt nạ mạng như trên. Nếu có dùng DHCP trong mạng, bạn hãy đánh dấu vào nút "Obtain an IP Address from a DHCP server".



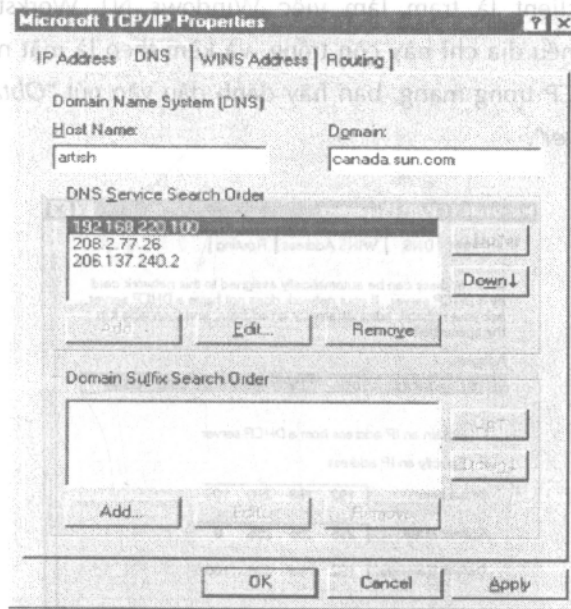
H. XVI. 31. Bảng Microsoft TCP/IP Properties của Windows

Nếu bạn không thấy còn dư giá trị địa chỉ IP trống, và bạn là người quản trị mạng, bạn hãy dùng giá trị địa chỉ IP, như trong ví dụ của ta là 192.168.x.x, nghĩa là giá trị nào đấy thuộc về mạng con là mạng LAN. Nếu bạn chỉ là người dùng bình thường, hãy yêu cầu người quản trị mạng cung cấp cho bạn địa chỉ IP còn trống để dùng cho máy của bạn.

Trường dành cho gateway thường để tham chiếu đến máy tính hoạt động như một router. Nếu bạn có các router nối liên mạng, bạn phải điền vào địa chỉ IP của một trong số router đó trong mạng con của bạn.

### Bảng DNS

Tiếp theo, ta xem xét đến bảng DNS được thể hiện cụ thể cho ví dụ trên H. XVI. 32.



H. XVI. 32. Trang DNS

Dịch vụ tên vùng (DNS) có tác dụng chuyển các tên Internet theo kiểu *atrish.example.com* thành tên mà máy có thể hiểu được là địa chỉ IP như 192.168.236.10. Ta có hai cách để thực hiện điều đó trên các máy Windows NT: hoặc để server DNS chuyển tên cho các máy client, hoặc giữ trong các máy trạm tại chỗ danh sách tên / địa chỉ IP.

Đối với các mạng LAN không nối Internet, danh sách các host thường không lớn và có thể được giữ trong file tại chỗ. Mạng được nối với Internet thường sử dụng dịch vụ DNS để làm việc chuyển đổi đó do không thể biết trước được địa chỉ nào và vào thời điểm nào bạn hay các người dùng trong mạng của bạn sẽ truy cập vào Internet. Nếu nghi ngờ không biết liệu có hay không server DNS, hoặc không biết địa chỉ IP của nó, bạn hãy xem nội dung của file `/etc/resolv.conf` trên server Samba. Mọi máy tính dùng DNS phải có file trên, với nội dung có các dòng vào giống như sau:

```
resolv.conf
domain example.com
nameserver 127.0.0.1
nameserver 192.168.236.20
```

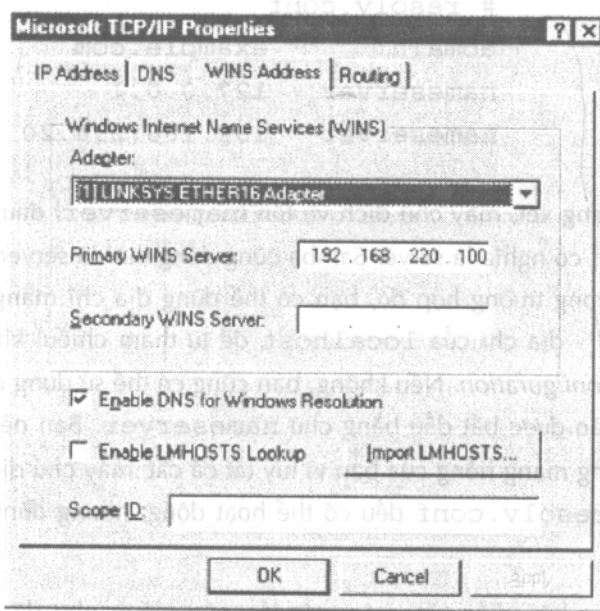
Trong ví dụ đang xét, máy chủ dịch vụ tên (nameserver) đầu tiên trong danh sách là 127.0.0.1, có nghĩa là server Samba cũng đồng thời là server DNS cho mạng LAN được xét. Trong trường hợp đó, bạn có thể dùng địa chỉ mạng của nó (không phải là 127.0.0.1 - địa chỉ của localhost để tự tham chiếu) khi phải điền vào hộp thoại *DNS Configuration*. Nếu không, bạn cũng có thể sử dụng các địa chỉ khác trong các dòng vào được bắt đầu bằng chữ nameserver. Bạn nên cố gắng chọn máy nào nằm trong mạng riêng của bạn vì tuy tất cả các máy chủ dịch vụ tên liệt kê trong file */etc/resolv.conf* đều có thể hoạt động, nhưng dùng máy gần nhất sẽ nhanh hơn.

Cuối cùng, bạn hãy điền tên máy một lần nữa giống như đã làm trong bảng *Identification* của hộp thoại khung *Network* (lần này cho TCP/IP). Và bạn nên xác định tên vùng DNS mà máy của bạn đang nằm trong đó. Ví dụ, nếu trạm làm việc của bạn có tên vùng là *example.com* thì hãy đưa tên đó vào. Các tùy chọn còn lại có thể bỏ qua.

### **Bảng địa chỉ WINS**

Nếu không dùng server DNS, bạn vẫn phải tìm cách chuyển các tên NetBIOS thành các địa chỉ IP và ngược lại. Người ta khuyên nên cấu hình cả DNS và WINS; Windows NT có vẻ ưa dùng WINS hơn và WINS có thể dùng DNS như là chỗ dựa nếu nó không thể phân giải được địa chỉ máy nào. Bảng *WINS Address* được chỉ ra trên H. XVI. 33.

Nếu có server WINS, bạn hãy điền địa chỉ của nó vào trong khoảng trống được dành cho *Primary WINS Server*. Nếu server Samba có cung cấp dịch vụ WINS (nói cách khác, bạn có dòng vào *wins service = yes* trong file cấu hình *smb.conf* của server Samba), bạn phải viết địa chỉ IP của server Samba vào đây. Nếu không, bạn phải đưa vào địa chỉ IP của một server WINS khác trong mạng của bạn.

H. XVI. 33. Trang *WINS Address*

Có thể bạn cũng nhận thấy rằng có một trường dành cho thiết bị mạng và bạn phải xác định card Ethernet mà bạn dùng giao thức TCP/IP trên nó để WINS có thể cung cấp đúng dịch vụ tên trên mạng. Nếu bạn có cả hai thiết bị mạng là thiết bị mạng LAN (card Ethernet) và thiết bị quay số điện thoại (modem chẳng hạn), bạn phải để tên của thiết bị mạng LAN vào trường đó.

Cuối cùng, bạn chọn hộp kiểm tra "*Enable DNS for Windows Resolution*" để WINS sẽ dùng DNS như chỗ dựa nếu nó không tìm được tên máy. Bạn có thể bỏ qua các tùy chọn khác.

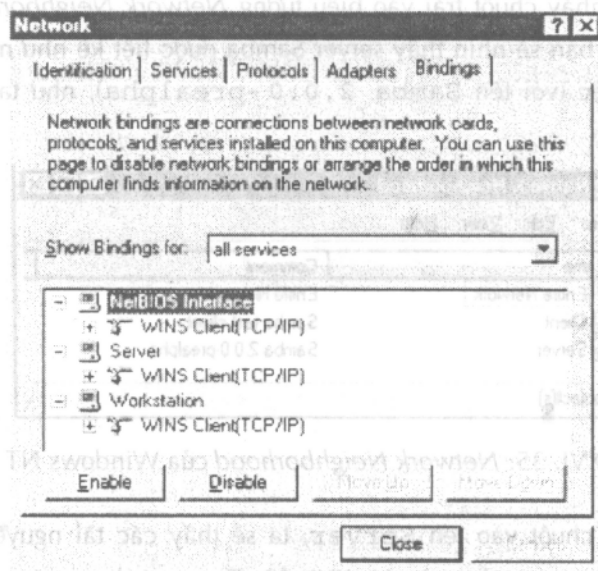
### Các file hosts

Nếu bạn không có cả DNS lẫn WINS, và nếu bạn không muốn dùng địa chỉ phân phát công cộng, bạn cần phải cung cấp cho máy một bản danh sách địa chỉ IP và tên máy cho các máy tính trong mạng theo khuôn dạng chuẩn của UNIX trong file `/etc/hosts`. Nói chung, cách dùng các file `hosts` không được khuyến khích vì việc bảo trì file `/etc/hosts` trên một mạng hoạt động mạnh là dễ bị

mắc lỗi. File `hosts` của Windows thường nằm trong thư mục `\WINDOWS\HOSTS` hoặc trên bất cứ ổ đĩa tại chỗ nào có cài đặt Windows. File mẫu có dạng như sau:

```
127.0.0.1 localhost
192.168.236.1 tuan tuan.example.com
192.168.236.2 cuong cuong.example.com
192.168.236.3 minh minh.example.com
192.168.236.4 chi chi.example.com
192.168.236.5 hien hien.example.com
192.168.236.6 chuyen chuyen.example.com
192.168.236.7 kim kim.example.com
```

Nếu muốn, bạn có thể chép thẳng nội dung file `/etc/hosts` của server Samba. Dạng cú pháp hoàn toàn như nhau. File này có chức năng giống hệt các file `hosts` trên server UNIX. Cũng phải nhắc lại một lần nữa là các file `hosts` của Windows chỉ được dùng như là biện pháp cuối cùng mà thôi.



H. XVI. 34. Các nối kết các dịch vụ



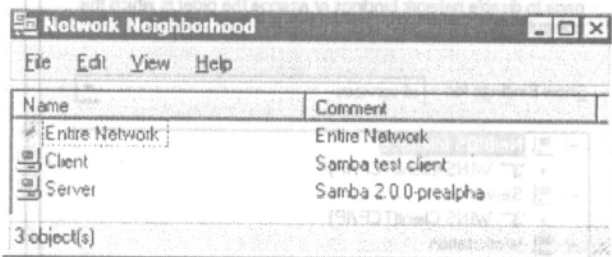
### Các nối kết

Thuật ngữ nối kết - *binding* - có nghĩa là "được kết nối với nhau trong thời gian cấu hình", hay, giao thức TCP/IP sẽ mở kênh thông qua card Ethernet một cách chắc chắn. Nếu bạn quay trở lại hộp thoại *Network* (ví dụ như H. XVI, 29), vào trang *Bindings* và đặt vào trường *Show* giá trị "all services", sau đó tách các nhánh bằng cách kích chuột vào dấu "+" trong cây, bạn sẽ nhìn thấy nội dung hiển thị tương tự như trên.

Hình vẽ H. XVI. 34 có nghĩa là các dịch vụ giao diện Workstation, Server, và NetBIOS được nối với máy client WINS khi đã kết nối được cho Microsoft TCP/IP.

#### ■ *Kết nối với server Samba*

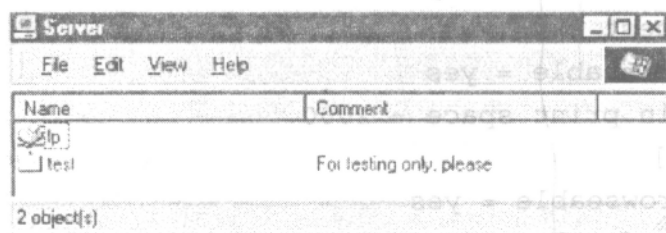
Bạn hoàn toàn có thể yên tâm sử dụng các giá trị mặc định cho các trường còn lại trong hộp thoại bảng *Network* và nhấn chuột trái vào nút OK để hoàn thành việc định cấu hình. Một khi các file đã được nạp đúng cách, bạn cần khởi động lại máy để nhận cấu hình mới. Sau khi máy khởi động lại xong, bạn hãy đăng nhập vào Windows NT rồi nhấn chuột trái vào biểu tượng *Network Neighborhood* trên màn hình làm việc, và bạn sẽ nhìn thấy server Samba được liệt kê như một thành phần của nhóm làm việc (với tên Samba 2.0.0 -prealpha), như ta thấy trong H. XVI. 35.



H. XVI. 35. *Network Neighborhood* của Windows NT

Hãy nhấn đúp chuột vào tên *Server*, ta sẽ thấy các tài nguyên được server Samba chia sẻ cho mạng như trên H. XVI. 36. Trong ví dụ ta xét, các tài nguyên được chia sẻ cho trạm làm việc Windows NT là thư mục *test* và máy in mặc định *lp*.

Chú ý: Xem phần tương tự về người dùng IPC\$ đối với các máy client Windows 9x.



H. XVI. 36. Các tài nguyên được chia sẻ của Server (Samba)

## XVI. 5. Chia sẻ đĩa cứng

Các daemon Samba **smbd** và **nmdb** được điều khiển thông qua một file ASCII duy nhất, **smb.conf**, có chứa tới hơn hai trăm các tùy chọn riêng. Các tùy chọn đó quy định cách Samba phải tác động trở lại tới mạng máy tính, bao gồm từ các quyền đơn giản tới các nối kết được mã hóa và các vùng của Windows NT. Trong phần này ta sẽ làm quen kỹ hơn với cấu trúc của file cấu hình Samba và cách dùng các tùy chọn của nó cho mục đích chia sẻ đĩa cứng, một tài nguyên quan trọng nhất, cho những người dùng làm việc trên các máy trạm Windows.

### XVI. 5. a. Nghiên cứu về file cấu hình Samba

Ta đã biết, để quy định chế độ làm việc ban đầu cho Windows, Microsoft đã dùng file **windows.ini**. Cấu trúc của file **smb.conf** cũng có thể được coi là rất giống như của file **windows.ini**. Ta hãy xét một ví dụ:

```
[global]
 log level = 1
 max log size = 1000
 socket options = TCP_NODELAY IPTOS_LOWDELAY
 guest ok = no
[homes]
 browseable = no
```

```
map archive = yes
[printers]
 path = /usr/tmp
 guest ok = yes
 printable = yes
 min print space = 2000
[test]
 browseable = yes
 read only = yes
 guest ok = yes
 path = /export/samba/test
```

Mặc dù có thể bây giờ bạn chưa biết hết ý nghĩa nội dung của nó, nhưng phương án này của file `smb.conf` là rất có ích khi bạn cần cấu hình nhanh Samba. Tóm lại, file cấu hình này thiết lập việc ghi lại log cho việc gỡ rối cơ bản trong một file log mặc định có kích thước không quá 1MB, tối ưu hóa các nối kết socket TCP/IP giữa server Samba và các client SMB bất kỳ, và cho phép Samba tạo đĩa cứng được chia sẻ cho mỗi một người dùng có đăng ký tài khoản người dùng UNIX chuẩn trong server. Thêm nữa, mỗi một chiếc trong số máy in được đăng ký trong server đều trở thành thiết bị sử dụng công cộng, do cùng có một sự chia sẻ chỉ đọc được thiết lập ánh xạ tới thư mục `/usr/tmp`. Phần cuối cùng của file cấu hình tương tự như phần xác định sự chia sẻ đĩa cứng (thư mục `/export/samba/test`) đã được dùng để kiểm tra kết quả cài đặt Samba trong phần XVI. 3 “*Cài đặt Samba trên hệ thống UNIX*”. Tại đây, tất cả người dùng đã đăng ký có quyền xem nội dung tài nguyên được chia sẻ, nhưng không thể sửa đổi được.

▪ *Cấu trúc của file cấu hình*

Nếu chỉ nhìn các thành phần chính trong file cấu hình, ta thấy như sau:

```
[global]
...
[homes]
...
```

```
[printers]
...
[test]
...
```

Những tên được bao trong các ngoặc vuông dùng để ký hiệu cho các phần của file cấu hình `smb.conf`, mô tả các chia sẻ hay dịch vụ mà Samba cung cấp. Ví dụ, các phần `[test]` và `[homes]` là các chia sẻ riêng rẽ đối với đĩa cứng; chúng chứa các tùy chọn được ánh xạ tới các thư mục cụ thể trên server Samba. Phần chia sẻ `[printers]` chứa các tùy chọn ánh xạ tới các máy in khác nhau của server. Tất cả các phần được xác định trong file `smb.conf`, trừ phần `[global]`, sẽ được coi như các chia sẻ đĩa cứng hoặc máy in cho những người dùng kết nối với server Samba.

Các dòng vào còn lại là các tùy chọn riêng được quy định cụ thể cho sự chia sẻ đã được đề cập tới. Các tùy chọn đó có tác dụng cho tới khi bắt đầu một phần mới được ký hiệu trong cặp ngoặc vuông, hoặc cho tới điểm cuối của file. Các tùy chọn trong file `smb.conf` được thiết lập bằng cách gán giá trị cho chúng. Mỗi một tùy chọn cấu hình đều có cú pháp đơn giản:

`option = value`

### *Khoảng trống, các dấu ngoặc và dấu phẩy*

Một điểm quan trọng mà ta luôn luôn phải nhớ tới trong các tùy chọn cấu hình là khoảng trống có giá trị có nghĩa. Ví dụ, ta xét tùy chọn sau: `volume = The Big Bad Hard Drive Number 3543`

Samba không coi trọng các khoảng trống sau ký tự “e” cuối cùng trong từ “volume” và trước ký tự “T” của từ “The”. Các khoảng trống còn lại đều có nghĩa và được Samba diễn giải khi đọc file. Trong tên của các tùy chọn, ví dụ như `guest ok`, các khoảng trống cũng không có nghĩa, nhưng ta nên tuân theo cách viết có khoảng trống một cách nhất quán để tạo sự trong sáng trong cách viết về sau này.

Nếu cảm thấy an toàn hơn khi dùng các dấu ngoặc ở đầu và cuối các giá trị của tùy chọn cấu hình, bạn có thể mạnh dạn dùng chúng vì Samba sẽ bỏ qua các dấu

ngoặc đó. Nhưng không bao giờ được dùng các dấu ngoặc cho chính các tùy chọn cấu hình vì Samba sẽ coi đó là lỗi.

Cuối cùng, bạn có thể dùng khoảng trống để ngăn cách chuỗi các giá trị trong danh sách, hoặc có thể dùng các dấu phẩy. Hai cách trên là tương đương, như được thấy trong các dòng ví dụ dưới đây:

```
netbios aliases = sales, accounting, payroll
netbios aliases = sales accounting payroll
```

Tuy nhiên, cũng vì lý do đảm bảo sự trong sáng trong cách viết, bạn nên dùng chỉ một cách (hoặc khoảng trống, hoặc dấu phẩy) trong một số trường hợp, còn cách kia - trong một số trường hợp khác.

### Chữ viết hoa

Chữ viết hoa không có ý nghĩa gì đối với file cấu hình Samba, ngoại trừ trong các vị trí mà hệ điều hành được chỉ tới không cho phép viết. Ví dụ, giả sử bạn đưa tùy chọn sau vào cho sự chia sẻ chỉ tới thư mục `/export/samba/simple`:

```
PATH = /EXPORT/SAMBA/SIMPLE
```

Đối với Samba sẽ không gây rắc rối gì nếu đường dẫn trên được viết hoàn toàn bằng chữ hoa, nhưng khi cố gắng kết nối với thư mục trên ta sẽ gặp thất bại, vì hệ điều hành UNIX phân biệt chữ viết thường và viết hoa. Trong hệ thống file UNIX thư mục đã cho được xác định với tên `/export/samba/simple`, nên tất nhiên đường dẫn với những ký tự viết hoa trên sẽ không bao giờ được tìm thấy và các máy client cũng không thể truy cập được tới tài nguyên chia sẻ đó.

### Viết tiếp dòng như thế nào

Trong trường hợp dòng vào quá dài không thể gói gọn trong khoảng không gian mà cửa sổ dòng lệnh cho phép, bạn có thể viết tiếp dòng trong file cấu hình Samba bằng cách dùng ký hiệu dấu gạch ngược `"\"`, tương tự như trong ví dụ sau:

---

```
comment = Su chia se dau tien la ban sao chinh cua s\
 an pham phan mem Teamworks moi.
```

Ta thấy, hai dòng trên được xử lý như một dòng duy nhất, chữ được viết khi nối dòng là “san”. Cách nối dòng bằng dấu gạch ngược “\” cũng là cách thường dùng khi ta viết mã nguồn cho shell script của UNIX.

### Chú thích

Các dòng chú thích có thể được đưa vào file cấu hình smb.conf bằng cách đặt trước chúng dấu thăng “#” hoặc dấu chấm phẩy “;”. Hai ký tự trên là tương đương. Ví dụ, ba dòng vào đầu trong ví dụ sau được coi như chú thích:

```
Đây là phần dành cho các máy in. Chúng ta dành
; khoảng không gian in tối thiểu là 2000 để tránh một
; số lỗi do tràn không gian hàng cho in.
[printers]
 public = yes
 min print space = 2000
```

Samba bỏ qua mọi dòng chú thích và số dòng chú thích không bị giới hạn cả về nội dung và vị trí đặt. Tuy nhiên, việc dùng dấu nối dòng “\” trong các dòng chú thích như trong ví dụ đã nêu không được khuyến khích nhiều. Thay vào đó, ta nên “chặt” các dòng chú thích thành các dòng chú thích nhỏ hơn được bắt đầu bằng dấu “#” hay dấu “;”.

### Thay đổi trong thời gian chạy

Có thể thay đổi file smb.conf và bất kỳ tùy chọn nào của nó vào thời điểm bất kỳ trong khi các daemon Samba đang chạy. Theo mặc định, Samba kiểm tra file cấu hình của mình cứ 60 giây một lần để tiếp nhận các thay đổi mới. Nếu không muốn chờ đợi lâu như vậy, bạn có thể bắt các daemon đó nạp lại bằng cách gửi tín hiệu SIGHUP tới chúng, hoặc chỉ đơn giản là tắt chúng đi rồi khởi động lại.

Ví dụ, nếu tiến trình **smbd** có PID là 893, ta có thể bắt nó đọc lại file cấu hình bằng lệnh sau đây:

```
kill -SIGHUP 893
```

Không phải tất cả các thay đổi đều được các máy client chấp nhận ngay. Ví dụ, các tài nguyên chia sẻ hiện thời đang được sử dụng sẽ không được đăng ký cho đến khi các máy client cắt các nối kết rồi kết nối lại tới các tài nguyên đó. Thêm nữa, các thông số đặc biệt của server như nhóm làm việc hoặc tên NetBIOS của server cũng sẽ không được đăng ký ngay lập tức. Điều đó giúp cho các máy client tích cực không bị ngắt nối kết một cách bất ngờ hoặc gặp phải các vấn đề không được chờ đợi về truy cập tài nguyên khi phiên làm việc vẫn đang được mở.

■ *Các biến*

Samba có một tập hợp đầy đủ các biến xác định các đặc trưng của server Samba và của các máy client nối với nó. Mỗi một biến được bắt đầu bằng dấu phần trăm "%", tiếp theo là một ký tự đơn viết hoa hoặc viết thường và chỉ có thể được dùng bên vế phải của dòng lệnh tùy chọn cấu hình theo cú pháp `option = variable` như trong câu lệnh ví dụ sau:

```
[pub]
```

```
path = /home/ftp/pub/%a
```

Ký hiệu biến `%a` có nghĩa đại diện cho kiến trúc của máy tính client, như `WinNT` để chỉ các máy tính chạy trên Windows NT, `Win95` - cho máy Windows 95 hoặc 98, hay như `WfWg` - cho Windows for Workgroups (Windows 3.11). Theo cách viết trên, Samba sẽ gán đường dẫn chung chỉ tới tài nguyên được chia sẻ trong phần `[pub]` cho các máy client chạy trên Windows NT, gán đường dẫn khác cho các máy Windows 9x, và một đường dẫn nữa cho các máy với Windows for Workgroups. Nói cách khác, các đường dẫn mà theo đó mỗi máy client nhận thấy tài nguyên chia sẻ sẽ khác nhau, tùy thuộc vào kiến trúc của client, hay câu lệnh trên tương đương với ba câu lệnh sau:

```
/home/ftp/pub/WinNT
```

```
/home/ftp/pub/Win95
```

```
/home/ftp/pub/WfWg
```

### Các biến của Samba

Biến	Định nghĩa
<b>Các biến của máy client</b>	
%a	Kiến trúc của các máy client (ví dụ, Samba, WfWg, WinNT, Win95, hoặc UNKNOWN).
%I	Địa chỉ IP của client (ví dụ, 192.168.220.100).
%m	Tên NetBIOS của client.
%M	Tên DNS của client.
<b>Các biến về người dùng</b>	
%g	Nhóm chính của %u.
%G	Nhóm chính của %U.
%H	Thư mục home hiện thời của %u.
%u	Tên người dùng hiện thời trong hệ thống UNIX.
%U	Tên người dùng được yêu cầu trên máy client (không phải lúc nào Samba cũng dùng tên này).
<b>Các biến về tài nguyên được chia sẻ</b>	
%p	Đường dẫn cho automounter tới thư mục gốc của tài nguyên được chia sẻ, nếu thư mục đó khác với %P.
%P	Thư mục gốc hiện thời của tài nguyên được chia sẻ.
%S	Tên hiện thời của tài nguyên được chia sẻ.
<b>Các biến của server</b>	
%d	Định danh tiến trình (PID) server hiện thời.
%h	Tên host DNS của server Samba.
%L	Tên host NetBIOS của server Samba.
%N	Thư mục home của server Samba, lấy từ file ảnh xạ (map) của <b>automount</b> .
%v	Phiên bản Samba.
<b>Các biến khác</b>	
%R	Mức của giao thức SMB được thỏa thuận thiết lập.
%T	Ngày giờ hiện thời.



Thông thường, người ta dùng biến theo cách được nêu trong ví dụ ở trên, khi muốn các người dùng khác nhau dùng các cấu hình tùy biến dựa trên cơ sở các điều kiện và đặc trưng riêng của các máy client mà họ sử dụng. Samba có tất cả mười chín biến được mô tả trong bảng *Các biến của Samba* vừa được trình bày trên đây.

Sau đây là một ví dụ dùng biến khác của Samba. Giả sử ta có năm máy client trong mạng, nhưng có một máy, với tên là *hai*, khi kết nối với server Samba đòi hỏi nạp một cấu hình phần `[homes]` hơi khác so với của các máy còn lại. Trong Samba, điều đó dễ dàng thực hiện được:

```
[homes]
...
include = /usr/local/samba/lib/smb.conf.%m
...
```

Tùy chọn `include` ở đây bắt Samba phải đọc thêm file cấu hình riêng rẽ cho mỗi một máy NetBIOS cụ thể “%m”, ngoài file cấu hình hiện thời. Nếu như tên máy của máy client là *hai*, và nếu file `smb.conf.hai` tồn tại trong thư mục `samba_dir/lib/` (hoặc bất cứ thư mục nào được chọn để lưu các file cấu hình), Samba sẽ chèn file cấu hình đó vào vị trí của file cấu hình mặc định. Nếu như bất kỳ một tùy chọn cấu hình nào được định nghĩa lại trong file `smb.conf.hai`, các giá trị đó sẽ ghi đè lên các giá trị cũ được xác định trước đó.

Một chú ý quan trọng là Samba không báo lỗi, nếu không có file cấu hình `include` - đơn giản là nó không làm gì cả trong trường hợp đó. Như vậy, Samba cho phép ta có thêm một file cấu hình cho máy client *hai*, thay vì chỉ có một file cho mỗi máy NetBIOS có trong mạng.

Các file cấu hình tùy thuộc vào đặc điểm cụ thể của máy có thể được dùng cho cả việc tùy biến các tài khoản người dùng và cả cho việc gỡ rối cho Samba một cách dễ dàng hơn. Thực vậy, nếu máy client gặp rắc rối, ta có thể dùng cách trên để có một file log riêng thay vì phải đọc và tìm lỗi trong file ở mức chung cho toàn mạng. Đối với mạng lớn, dùng cách này ta không làm ảnh hưởng tới tốc độ của tất cả các người dùng Samba còn lại khác, không làm máy chủ phải chịu thêm tải do những file log không cần thiết, và nhất là có thể tránh không khởi động lại Samba.

### *XVI. 5. b. Các phần đặc biệt của file cấu hình Samba*

Trong mục này ta sẽ xem xét cụ thể hơn các phần trong file cấu hình `smb.conf`.

- *Phần [globals]*

Phần `[globals]` xuất hiện hầu như trong mọi file cấu hình Samba, thậm chí khi trong đó không có dòng lệnh bắt buộc nào. Mọi tùy chọn được thiết lập trong phần này đều được áp dụng đối với tất cả các tài nguyên chia sẻ khác, vì nội dung của phần này sẽ được sao chép vào các phần khác. Nhưng các phần khác, nếu cũng có các tùy chọn giống như trong phần `[globals]`, thì trong các phần đó, các tùy chọn sẽ được xác định với các giá trị mới được ghi đè lên các giá trị cũ của `[globals]`.

Để minh họa, ta xét lại ví dụ đã nêu:

```
[global]
 log level = 1
 max log size = 1000
 socket options = TCP_NODELAY IPTOS_LOWDELAY
 guest ok = no

[homes]
 browseable = no
 map archive = yes

[printers]
 path = /usr/tmp
 guest ok = yes
 printable = yes
 min print space = 2000

[test]
 browseable = yes
```

```
read only = yes
guest ok = yes
path = /export/samba/test
```

Nếu ta kết nối máy client tới tài nguyên được chia sẻ [test], Samba trước tiên đọc phần [globals]. Tại đây, tùy chọn guest ok = no được xác định như giá trị mặc định chung cho tất cả các tài nguyên được chia sẻ trong suốt cả file cấu hình smb.conf này, bao gồm cả các phần tài nguyên chia sẻ [homes] và [printers]. Khi Samba đọc tới phần tài nguyên chia sẻ [test], nó sẽ nhận thấy tùy chọn trên đã được xác định lại và giá trị guest ok = yes sẽ được ghi đè lên giá trị của phần [globals].

Mọi tùy chọn xuất hiện trước phần được đánh dấu bằng ngoặc vuông “[ ]” đầu tiên, tức là bên ngoài phần [globals], cũng được coi là những tùy chọn chung.

#### ▪ Phần [homes]

Nếu một client nào đó cố gắng kết nối tới tài nguyên được chia sẻ không được nêu tên trong file cấu hình smb.conf, Samba sẽ tìm tài nguyên được chia sẻ [homes] trong file cấu hình. Nếu phần này tồn tại, tên của tài nguyên được chia sẻ không xác định kia sẽ được coi như tên người dùng của UNIX và được yêu cầu tìm trong cơ sở dữ liệu mật khẩu của server Samba. Nếu như có tên người dùng đó, Samba coi máy được nối tới ở trên là một người dùng UNIX đang cố kết nối tới thư mục home của mình trong server.

Ví dụ, giả sử một máy client kết nối với server Samba hydra lần đầu tiên, và cố truy cập tới tài nguyên được chia sẻ có tên là [dung]. Trong file smb.conf, không có tài nguyên được chia sẻ nào tên là [dung] được xác định, nhưng lại có phần [homes], vì thế Samba tìm file cơ sở dữ liệu mật khẩu và tìm xem có tài khoản người dùng dung trong hệ thống hay không. Sau đó Samba kiểm tra mật khẩu được client cung cấp và so sánh với mật khẩu của người dùng UNIX dung - hoặc trong file cơ sở dữ liệu mật khẩu nếu dùng mật khẩu không mã hóa, hoặc bằng file lệnh **smbpasswd** của Samba đối với các mật khẩu mã hóa. Nếu các mật khẩu đó trùng nhau, Samba nhận biết chắc là người dùng dung có quyền và đang muốn

kết nối tới thư mục `home` của mình trong máy UNIX. Sau đó Samba sẽ tự tạo tài nguyên được chia sẻ được gọi là `[dùng]` cho người dùng dùng.

Người ta cũng áp dụng phương pháp được thực hiện với phần `[homes]` để tạo tài khoản người dùng mới, kèm theo mật khẩu. Ta sẽ quay trở lại vấn đề này khi đề cập tới các nội dung về tài khoản người dùng, bảo mật và các vùng.

#### ▪ Phần `[printers]`

Phần đặc biệt thứ ba được gọi là `[printers]` tương tự như phần `[homes]`. Nếu một client cố kết nối tới tài nguyên được chia sẻ không có mặt trong file cấu hình  `smb.conf` file, và nếu tên của nó không thể tìm được trong file mật khẩu, Samba sẽ kiểm tra xem đó có phải sự chia sẻ máy in cho client đó. Samba thực hiện điều đó thông qua việc đọc file dữ liệu máy in (thường là `/etc/printcap` hay `/etc/terminfo`) để xem có tên của tài nguyên được chia sẻ đó hay không. Nếu có, Samba tạo ra tài nguyên được chia sẻ với tên liên quan tới việc chia sẻ máy in.

Giống như đối với phần `[homes]`, ta không cần phải bảo trì tài nguyên được chia sẻ cho mỗi một máy in của hệ thống trong file cấu hình  `smb.conf`. Thực vậy, Samba luôn dựa vào việc đăng ký máy in của UNIX nếu ta cần đến, và cung cấp các máy in đã đăng ký cho các client. Tuy nhiên, có một hạn chế nhỏ: nếu tài khoản người dùng và máy in đều có tên là hai, Samba bao giờ cũng tìm tài khoản người dùng trước tiên, bất kể là client thực ra là cần kết nối với máy in.

Các chi tiết về việc thiết lập tài nguyên được chia sẻ `[printers]` được trình bày trong phần liên quan tới việc in và phân giải tên.

#### ▪ Các tùy chọn cấu hình

Các tùy chọn trong file cấu hình Samba được chia làm hai loại: `global` (chung) và `share` (chia sẻ). Mỗi một loại quy định một tùy chọn sẽ được xuất hiện ở đâu trong file cấu hình.

#### **Global**

Các tùy chọn `global` phải có mặt chỉ trong phần `[global]` mà thôi. Đây là các tùy chọn thường chỉ áp dụng để xác định hoạt động của chính server Samba.

### **Share**

Các tùy chọn `share` có thể xuất hiện trong các tài nguyên được chia sẻ cụ thể, hoặc cả trong phần `[global]`. Nếu có mặt trong phần `[global]`, chúng sẽ xác định các giá trị mặc định cho tất cả các tài nguyên được chia sẻ, chừng nào chưa bị các tùy chọn cùng tên tại các phần tài nguyên được chia sẻ cụ thể ghi đè những giá trị mới.

Thêm nữa, các giá trị của các tùy chọn cấu hình còn được chia thành bốn loại theo các tiêu chí sau:

### **Boolean**

Đơn giản đó chỉ là các giá trị `yes` hoặc `no`, nhưng được biểu diễn qua một trong các chữ sau: `yes`, `no`, `true`, `false`, `0`, `1`. Không phân biệt chữ hoa và chữ thường.

### **Numerical**

Là số nguyên, số hexa, hoặc số bát phân. Các cú pháp chuẩn `0xnn` được áp dụng cho số hexa và `0nnn` cho số bát phân.

### **String**

Là một xâu ký tự có phân biệt chữ viết hoa và chữ viết thường, như tên file hoặc tên người dùng.

### **Enumerated list**

Là danh sách hữu hạn các giá trị được biết trước. Có thể coi loại `boolean` chính là loại `enumerated list` với chỉ hai giá trị.

## ***XVI. 5. c. Các tùy chọn của file cấu hình***

Samba có tất cả hơn 200 tùy chọn cấu hình. Trong mục này ta quan tâm tới một số tùy chọn cấu hình cho chính file cấu hình. Như ta đã thấy trong các mục trước, các file cấu hình là động, tức là luôn luôn biến đổi. Bạn có thể cho Samba thêm vào và thậm chí thay đổi các tùy chọn cấu hình khi Samba xử lý chúng. Các tùy chọn của riêng file cấu hình được kê trong bảng dưới đây và tiếp theo sẽ được xét cụ thể.

### Các tùy chọn của file cấu hình

Tùy chọn	Thông số	Chức năng	Mặc định	Quy mô
<code>config file</code>	<code>string</code> (tên xác định)	Đặt vị trí của file cấu hình được dùng thay cho file cấu hình hiện thời.	Không có	Global
<code>include</code>	<code>string</code> (tên xác định)	Xác định các phần thêm của các tùy chọn cấu hình sẽ được thêm vào vị trí đã cho trong file cấu hình.	Không có	Global
<code>copy</code>	<code>string</code> (tên tài nguyên được chia sẻ)	Cho phép nhân bản các tùy chọn cấu hình của phần khác vào trong phần hiện thời (phần chỉ tài nguyên được chia sẻ).	Không có	Share

#### ▪ *config file*

Tùy chọn chung `config file` xác định đối tượng thay thế cho file cấu hình mặc định. Nếu đối tượng đó - file đích - tồn tại, phần còn lại của file cấu hình trước khi Samba đọc tới tùy chọn này sẽ bị bỏ qua; Samba sẽ tự cấu hình với các giá trị mới được lấy từ file đích trên.

Ta lấy ví dụ, dòng lệnh cấu hình sau sẽ cho Samba biết phải dùng file cấu hình được xác định bởi tên NetBIOS của máy client đang kết nối, nếu như file đó tồn tại:

```
[global]
 config file = /usr/local/samba/lib/smb.conf.%m
```

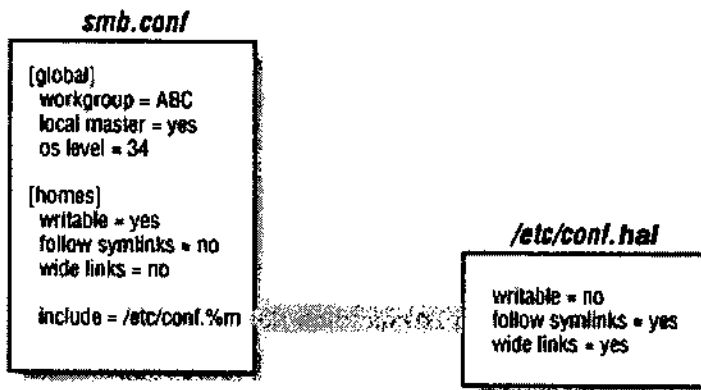
Nếu file đích không tồn tại, dòng lệnh trên không có hiệu lực và Samba tiếp tục dùng file cấu hình hiện thời.

### ▪ *include*

Tùy chọn này đã được giới thiệu khá kỹ từ trước và có tác dụng sao chép file đích vào trong file cấu hình hiện thời, như được chỉ trên H. XVI. 37. Tùy chọn này có thể được dùng như sau:

```
[global]
 include = /usr/local/samba/lib/smb.conf.%m
```

Nếu file cấu hình đích không được tìm thấy (trong ví dụ trên là file /etc/conf/hai), tùy chọn này sẽ bị bỏ qua.



H. XVI. 37. Tùy chọn *include* trong file cấu hình của Samba

Tùy chọn *include* không thể hiểu được các biến %u (người dùng), %p (thư mục gốc của tài nguyên được chia sẻ), hay %s (tài nguyên được chia sẻ hiện thời) bởi vì chúng chưa được xác lập trong thời gian file cấu hình của Samba đang được đọc.

### ▪ *copy*

Tùy chọn cấu hình *copy* cho phép nhân bản các tùy chọn cấu hình của tài nguyên được chia sẻ mà bạn đánh dấu trong phần đã cho. Tài nguyên được chia sẻ

phải được xuất hiện trong file cấu hình Samba trước thời điểm thực hiện việc nhân bản này. Ví dụ:

```
[template]
 writable = yes
 browsable = yes
 valid users = tuan, hai, minh [data]
 path = /usr/local/samba
 copy = template
```

Chú ý rằng mọi tùy chọn trong tài nguyên được chia sẻ được kích hoạt bởi lệnh `copy` sẽ bị ghi đè bằng giá trị được sao chép, không quan trọng là chúng có mặt trước hay sau lệnh `copy` đó.

#### ***XVI. 5. d. Cấu hình server Samba***

Bây giờ ta đã có thể cấu hình server Samba. Đầu tiên ta hãy chú ý đến ba tùy chọn cấu hình cơ bản xuất hiện trong phần `[global]` của file cấu hình `smb.conf` nằm trong thư mục `/usr/local/samba/lib`:

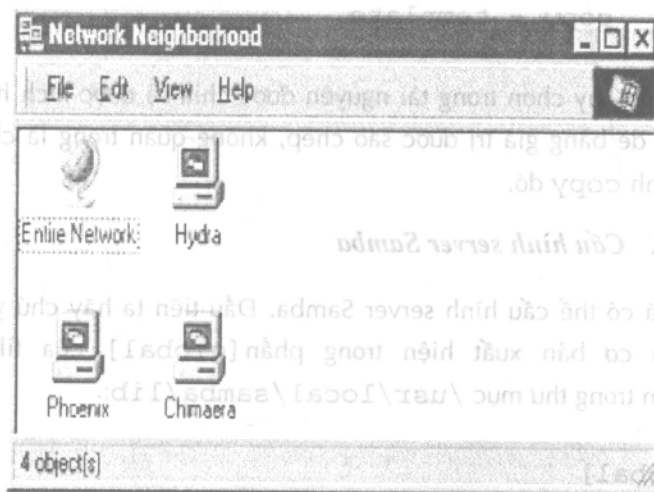
```
[global]
 # Server configuration parameters
 netbios name = HYDRA
 server string = Samba %v on (%L)
 workgroup = SIMPLE
```

Rõ ràng là file cấu hình này cực kỳ đơn giản, nó thông báo cho mạng NBT biết là có tồn tại server Samba với tên NetBIOS là `hydra`. Thêm nữa, máy tính này thuộc về nhóm làm việc `SIMPLE` và trong mô tả cho các máy client của mạng server Samba cho biết không những tên NetBIOS của nó mà còn cả phiên bản Samba nó được cài đặt.

Nếu như có dùng các mật khẩu đã mã hóa, ta phải thêm vào một dòng có nội dung `encrypt passwords=yes` vào file cấu hình trên.



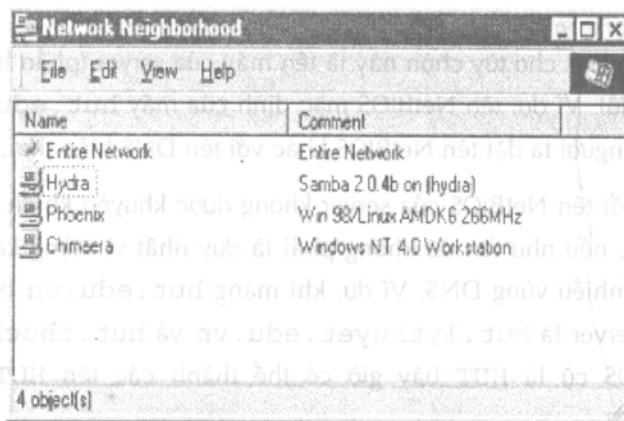
Sau khi đã soạn thảo nội dung như trên của file `smb.conf` và đặt nó vào đúng vị trí cần thiết, ta khởi động lại server Samba và dùng các máy client Windows để kiểm tra kết quả. Tất nhiên các máy client Windows đó cũng phải thuộc về nhóm SIMPLE - trong ví dụ ta vẫn dùng từ đầu chương - đó là các máy phoenix và chimaera. Sau khi nhấp phím chuột vào biểu tượng *Network Neighborhood* trên máy client Windows, một màn hình tương tự như H. XVI. 38 sẽ hiện ra.



H. XVI. 38. *Network Neighborhood* cho thấy server Samba

Ta có thể kiểm tra sâu kỹ tự chỉ server bằng cách liệt kê chi tiết của sổ *Network Neighborhood* (chọn mục *Details* dưới thực đơn *View*), như thấy trên H. XVI. 39 cho ví dụ của ta.

Nếu nhấp phím chuột vào biểu tượng *Hydra*, một cửa sổ với các dịch vụ được sẵn sàng cung cấp cho mạng sẽ xuất hiện. Nhưng bây giờ cửa sổ đó đang còn trống vì chưa hề có tài nguyên được chia sẻ nào được xác lập.

H. XVI. 39. Nhìn chi tiết cửa sổ *Network Neighborhood*

#### ■ Các tùy chọn cấu hình server

Bảng sau tổng kết các tùy chọn cấu hình server đã được trình bày.

#### Các tùy chọn cấu hình server

Tùy chọn	Thông số	Chức năng	Mặc định	Quy mô
netbios name	xâu ký tự	Đặt tên NetBIOS chính của server Samba.	Tên máy server DNS	Global
server string	xâu ký tự	Xác định xâu ký tự mô tả server Samba.	Samba %v	Global
workgroup	xâu ký tự	Đặt nhóm NetBIOS của các máy tính mà server Samba cũng thuộc về.	Xác định khi biên dịch	Global

#### netbios name

Tùy chọn netbios name cho phép đặt tên NetBIOS cho server. Ví dụ:

---

```
netbios name = DHBKHN1
```

Giá trị mặc định cho tùy chọn này là tên máy của server (phần bên trái cùng của tên DNS đầy đủ). Ví dụ, tên NetBIOS mặc định của máy `hut.edu.vn` sẽ là `HUT`. Thông thường, người ta đặt tên NetBIOS khác với tên DNS hiện thời.

Việc thay đổi tên NetBIOS của server không được khuyến khích nếu không có lý do chính đáng, nếu như tên đó không phải là duy nhất vì mạng LAN được chia ra thành hai hay nhiều vùng DNS. Ví dụ, khi mạng `hut.edu.vn` bị chia thành hai vùng với các server là `hut.lythuyet.edu.vn` và `hut.thuchanh.edu.vn` thì tên NetBIOS cũ là `HUT` bây giờ có thể thành các tên `HUTLYTHUYET` và `HUTTHUCHANH`.

Một ứng dụng khác của tùy chọn là định vị lại các dịch vụ SMB từ các máy tính đã thôi không hoạt động hoặc đã bị tháo bỏ. Ví dụ, `SALES` là server SMB cho một đơn vị kinh doanh nào đó đột nhiên bị hỏng phải thay bằng máy khác. Bạn chỉ cần xác định lại `netbios name = SALES` cho máy server Samba dự phòng và người dùng không cần phải thay đổi gì trong các máy client của họ.

### server string

Thông số của `server string` xác định nội dung dòng chú thích sẽ xuất hiện cạnh tên của server Samba trong cá cửa sổ *Network Neighborhood* (khi ở chế độ *Details*) lẫn cửa sổ quản lý in của Microsoft Windows. Bạn có thể dùng các biến chuẩn để cung cấp thông tin cho dòng mô tả đó, ví dụ ta đã dùng:

```
[global]
server string = Samba %v on (%h)
```

Giá trị mặc định của tùy chọn này chỉ cho biết phiên bản Samba hiện thời và tương đương với:

```
server string = Samba %v
```

*workgroup*

Thông số của tùy chọn `workgroup` thiết lập nhóm làm việc hiện thời, nơi mà server Samba tự thông báo cho các thành viên của mạng về mình. Các clients muốn truy cập được tài nguyên được chia sẻ trên server Samba phải cùng thuộc về một nhóm làm việc NetBIOS. Nên nhớ rằng các nhóm làm việc phải có các tên nhóm NetBIOS thực thụ, tuân theo quy tắc đặt tên NetBIOS chuẩn mà ta đã có dịp nói đến khi nghiên cứu về NetBIOS. Ví dụ:

```
[global]
 workgroup = SIMPLE
```

Giá trị mặc định của tùy chọn này được xác định trong thời gian biên dịch chương trình và nếu ta không thay đổi dòng vào tương ứng trong file `makefile`, nó sẽ vẫn giữ là `WORKGROUP`. Vì đó là tên nhóm làm việc dành cho các mạng NetBIOS không được cấu hình, ta nên thay tên nhóm làm việc trong file cấu hình Samba. Mặt khác, cũng rất không nên để tên nhóm làm việc trùng với tên của server.

***XVI. 5. e. Cấu hình chia sẻ đĩa cứng***

Trong ví dụ ở phần trước ta đã nhắc đến rằng do chưa có tài nguyên được chia sẻ nên cửa sổ chi tiết của server `hydra` đang còn trống. Bây giờ ta tiếp tục làm việc với file cấu hình Samba và tạo ra một đĩa cứng được chia sẻ còn trống có tên là `[data]`. Đây là các động tác cần thêm vào để đạt được kết quả vừa nêu:

```
[global]
 netbios name = HYDRA
 server string = Samba %v on (%L)
 workgroup = SIMPLE [data]
 path = /export/samba/data
 comment = Data Drive
 volume = Sample-Data-Drive
 writeable = yes
 guest ok = yes
```

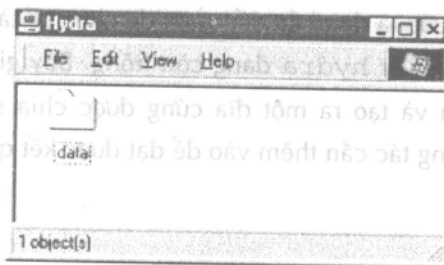
Tài nguyên được chia sẻ [data] thường là đĩa cứng được Samba chia sẻ và ánh xạ tới thư mục /export/samba/data trên server Samba. Ta đã cho thêm một dòng vào để có chú thích mô tả tài nguyên được chia sẻ đó là Data Drive, cũng như gán cho bản thân tài nguyên đó một cái tên Sample-Data-Drive.

Tài nguyên được chia sẻ được thiết lập có quyền ghi cho các người dùng. Giá trị mặc định của tùy chọn này là chỉ đọc được. Trong phương án không cần mức độ bảo mật chặt chẽ như ở đây, ta đặt giá trị yes cho tùy chọn guest ok, để cho bất kỳ ai cũng có thể kết nối được tới tài nguyên được chia sẻ.

Trên máy UNIX có cài đặt Samba ta tạo thư mục /export/samba/data với quyền root bằng các lệnh sau:

```
mkdir /export/samba/data
chmod 777 /export/samba/data
```

Bây giờ, nếu ta lại kết nối với server hydra (bằng cách kích phím chuột vào biểu tượng của server trong cửa sổ Network Neighborhood của Windows), một thư mục được chia sẻ với tên data đã xuất hiện, như được thấy trên H. XVI. 40, với các quyền đọc/ghi.



H. XVI. 40. Tài nguyên được chia sẻ ban đầu data trên server Samba

#### ■ Các tùy chọn cấu hình chia sẻ đĩa cứng

Các tùy chọn cơ bản của Samba dùng cho việc chia sẻ đĩa cứng mà ta đã đề cập đến từ trước được trình bày trong bảng sau.

### Các tùy chọn cơ bản chia sẻ đĩa cứng

Tùy chọn	Thông số	Chức năng	Mặc định	Quy mô
path (directory)	string (đường dẫn xác định)	Đặt thư mục UNIX dùng cho chia sẻ đĩa cứng hoặc cho việc xếp hàng chờ bởi máy in được chia sẻ.	/tmp	Share
guest ok (public)	nhị phân	Nếu đặt là yes, sẽ không cần kiểm tra xác nhận người dùng để truy cập tài nguyên được chia sẻ này.	no	Share
comment	string (xâu ký tự)	Đặt chú thích sẽ xuất hiện cùng tài nguyên được chia sẻ.	Không có	Share
volume	string	Đặt tên cho ổ đĩa, theo dạng của DOS.	Tên của tài nguyên được chia sẻ.	Share
read only	nhị phân	Nếu là yes, cho phép truy cập chỉ đọc tới tài nguyên được chia sẻ.	yes	Share
writable (write ok)	nhị phân	Nếu là no, cho phép truy cập chỉ đọc tới tài nguyên được chia sẻ.	no	Share

#### path

Tùy chọn này, đồng nghĩa với thư mục, cho biết đường dẫn tại gốc của tài nguyên các file hay công việc in được chia sẻ. Nếu đường dẫn cho việc chia sẻ in, tùy chọn sẽ chỉ đến một thư mục tạm thời để các file có thể được ghi trên server trước khi được xếp hàng chờ đến máy in đích (thường là /tmp và /var/spool). Nếu đường dẫn là đĩa cứng chia sẻ, nội dung của thư mục của tên tài nguyên được

chia sẻ hiển thị trên máy client sẽ trùng với nội dung trên server Samba. Ví dụ, nếu ta có đĩa cứng được chia sẻ như sau được chỉ ra trong file cấu hình:

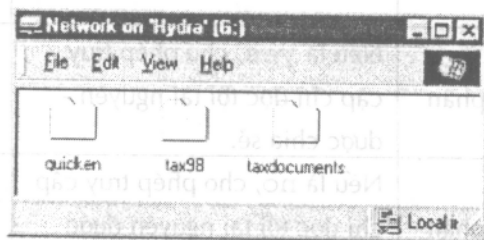
[network]

```
path = /export/samba/network
writable = yes
guest ok = yes
```

Và nội dung của thư mục /usr/local/network trên phía UNIX là:

```
$ ls -al /export/samba/network
drwxrwxrwx 9 root nobody 1024 Feb 16 17:17 .
drwxr-xr-x 9 nobody nobody 1024 Feb 16 17:17 ..
drwxr-xr-x 9 nobody nobody 1024 Feb 16 17:17 quicken
drwxr-xr-x 9 nobody nobody 1024 Feb 16 17:17 tax98
drwxr-xr-x 9 nobody nobody 1024 Feb 16 17:17 \
taxdocuments
```

Ta có thể thấy sự tương đương trên H. XVI. 41, nếu nhìn từ phía client của Samba.



H. XVI. 41. Hệ thống file mạng xác định bằng đường dẫn nhìn từ phía client Windows

### quest ok

Tùy chọn này (tên cũ là public) cho phép hoặc cấm việc truy cập của khách vào tài nguyên được chia sẻ. Giá trị mặc định là no. Nếu được đặt là yes, có nghĩa không cần phải có tên người dùng và mật khẩu đăng ký cũng có thể truy cập tài nguyên được chia sẻ. Khi người dùng kết nối đến, quyền được gán cho giống như

quyền của người dùng guest, mặc định trong Samba là nobody. Tuy nhiên, có thể quy định lại cả người dùng guest lẫn quyền hạn như trong ví dụ truy cập tài nguyên được chia sẻ [accounting] với quyền của tài khoản người dùng ftp:

```
[global]
 guest account = ftp
[accounting]
 path = /usr/local/account
 guest ok = yes
```

Chú ý rằng các người dùng vẫn kết nối tới tài nguyên được chia sẻ bằng cách dùng tổ hợp tên người dùng / mật khẩu hợp lệ. Nếu kết nối được, những người dùng đó vẫn có các quyền được quy định của mình. Chỉ khi kết nối hợp pháp trên không thể thực hiện được, lúc đó các quyền của người dùng guest mới được áp dụng. Có thể bắt buộc tất cả các người dùng nào kết nối tới tài nguyên được chia sẻ phải dùng tài khoản (và quyền) của guest bằng cách đặt tùy chọn `guest only = yes`.

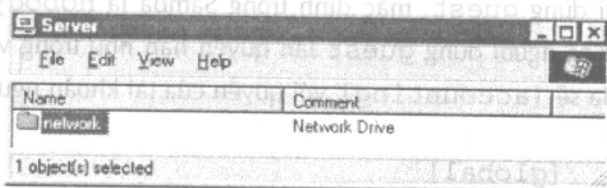
#### comment

Tùy chọn `comment` cho phép đưa vào dòng chú thích sẽ được gửi tới máy client khi nó duyệt tài nguyên được chia sẻ. Người dùng có thể đọc chú thích khi dùng lệnh *Details* đối với thư mục của tài nguyên được chia sẻ trong cửa sổ Windows *Network Neighborhood*, hoặc khi gõ lệnh `NET VIEW` trên dấu nhắc dòng lệnh của MS-DOS. Ví dụ, đây là cách để chèn chú thích đối với tài nguyên được chia sẻ [network]:

```
[network]
 comment = Network Drive
 path = /export/samba/network
```

Ta sẽ có thư mục tương tự như H. XVI. 42 trên máy client. Nên chú ý rằng chú thích chỉ hiện ra nếu ta thực hiện ánh xạ tài nguyên được chia sẻ như là ổ đĩa mạng của Windows.





#### H. XVI. 42. Chú thích về tài nguyên chia sẻ nhìn từ phía client Windows

Không nên bỏ qua tùy chọn này vì ta sẽ có thêm chú giải đối với tài nguyên được chia sẻ, cũng như tùy chọn server string đối với chính server Samba.

#### volume

Tùy chọn này cho phép đặt tên cho tài nguyên được chia sẻ để dùng như tên thư mục bởi SMB. Thông thường việc phân phối tên cho tài nguyên được chia sẻ được thực hiện trong file `smb.conf`. Tuy nhiên, nếu muốn đặt tên thêm ta có thể dùng tùy chọn này.

Ví dụ, một người cài đặt chương trình có thể kiểm tra tên của đĩa CD-ROM để tin chắc đĩa CD-ROM cần thiết đã nằm trong ổ đĩa. Nếu ta sao chép nội dung của đĩa CD-ROM vào tài nguyên mạng được chia sẻ, và muốn cài đặt từ đó, có thể dùng tùy chọn này như sau:

[network]

comment = Network Drive

volume = STAROFFICE-51A-ENGLISH

path = /home/samba/network

#### read only và writeable

Các tùy chọn `read only` và `writable` (hoặc `write ok`) là hai cách tiếp cận từ hai phía đến cùng một mục đích. Ví dụ, bạn có thể đặt tùy chọn nào cũng được trong số hai dòng sau trong phần `[global]` hoặc tại phần xác định cho các tài nguyên được chia sẻ cụ thể:

`read only = yes`

```
writeable = no
```

Nếu bất kỳ tùy chọn nào ở trên được dùng, dữ liệu chỉ được đọc từ tài nguyên được chia sẻ mà không thể ghi vào đó được.

#### ***XVI. 5. f. Các tùy chọn về mạng của Samba***

Nếu chạy Samba trên máy có nhiều thư mục home (nghĩa là máy với nhiều mạng con), hoặc thậm chí muốn tăng cường bảo mật trên mạng con của mình, ta phải cẩn nhìn vào các tùy chọn cấu hình mạng.

Giả sử server Samba của ta được nối với mạng có nhiều mạng con và có thể đăng nhập vào được cả mạng 192.168.220.\* lẫn 134.213.233.\*. Sau đây là các bổ sung các tùy chọn cấu hình mạng vào file cấu hình đang hoạt động đã được lấy làm ví dụ:

```
[global]
 netbios name = HYDRA
 server string = Samba %v on (%L)
 workgroup = SIMPLE
 # Networking configuration options
 hosts allow = 192.168.220. 134.213.233. localhost
 hosts deny = 192.168.220.102
 interfaces = 192.168.220.100/255.255.255.0 \
 134.213.233.110/255.255.255.0
 bind interfaces only = yes
[data]
 path = /home/samba/data
 guest ok = yes
 comment = Data Drive
 volume = Sample-Data-Drive
 writeable = yes
```

Để giải thích ý nghĩa các thiết lập trong đoạn cấu hình mạng của file `smb.conf`, trước hết ta hãy xét các tùy chọn `hosts allow` và `hosts deny`.

Chúng được coi như có tác dụng tương đương với các file `hosts.allow` và `hosts.deny`, nằm trong các thư mục `/etc` của nhiều hệ thống UNIX, dùng để cho phép hay cấm các nối kết dựa trên cơ sở địa chỉ IP của các máy. Vậy tại sao ta không dùng luôn chính các file `hosts.allow` và `hosts.deny`? Lý do là các file trên không chỉ phục vụ mỗi việc truy cập tài nguyên được chia sẻ bằng các dịch vụ Samba mà thôi.

Với tùy chọn `hosts allow`, ta đã xác định phần đầu của địa chỉ IP: 192.168.220, có nghĩa là "tất cả các máy trong mạng con 192.168.220". Tuy nhiên, bằng dòng vào `hosts deny` tiếp theo, ta lại cấm không cho máy 192.168.220.102 thuộc mạng con đó có quyền đăng nhập server Samba.

Hai tùy chọn khác được xem xét tới là các địa chỉ `interfaces` và `bind interface only`. Samba, theo tùy chọn, chỉ gửi dữ liệu từ giao diện mạng chính mà trong ví dụ của ta là mạng con 192.168.220.100. Nếu ta muốn Samba gửi dữ liệu tới nhiều hơn một giao diện, ta cần phải chỉ ra một danh sách đầy đủ các giao diện mạng đó bằng tùy chọn `interfaces`. Trong ví dụ trước, ta liên kết Samba tới giao diện với hai mạng con (192.168.220 và 134.213.233) mà máy hoạt động nhờ định rõ địa chỉ giao diện mạng khác: 134.213.233.100. Nếu bạn có nhiều hơn một giao diện mạng (card mạng) trong máy tính của bạn, bạn cần phải đặt tùy chọn này do không có gì đảm bảo Samba sẽ chọn đúng giao diện mạng.

Cuối cùng, tùy chọn `bind interfaces only` hướng dẫn cho tiến trình **nmbd** không chấp nhận một thông báo phân phát công cộng nào ngoài từ những mạng con đã được tùy chọn `interfaces` xác định. Chú ý rằng điều này khác với `hosts allow` và `hosts deny`, là các tùy chọn ngăn không cho các máy tạo nối kết tới các dịch vụ, chứ không phải là cấm nhận các thông báo phân phát công cộng. Dùng tùy chọn `bind interfaces only` ta có thể cắt các datagram từ các mạng con bên ngoài tới server Samba. Thêm nữa, tùy chọn còn cho tiến trình **smbd** chỉ kết nối tới danh sách các giao diện được xác định thông qua tùy chọn `interface`.

- *Các tùy chọn mạng*

Chức năng và giá trị của các tùy chọn mạng được trình bày đầy đủ hơn qua tổng kết trong bảng sau.

### Các tùy chọn cấu hình mạng

Tùy chọn	Thông số	Chức năng	Mặc định	Quy mô
<code>hosts allow</code> ( <code>allow hosts</code> )	string (danh sách tên máy)	Xác định các máy có thể kết nối với Samba.	Không có	Share
<code>hosts deny</code> ( <code>deny hosts</code> )	string (danh sách tên máy)	Xác định các máy không thể kết nối với Samba.	Không có	Share
<code>interfaces</code>	string (danh sách các tổ hợp IP/netmask)	Thiết lập các giao diện mạng mà Samba sẽ đáp ứng. Cho phép sửa đổi các giá trị mặc định.	Phụ thuộc hệ thống	Global
<code>bind interfaces only</code>	nhị phân	Nếu đặt là <code>yes</code> , Samba sẽ chỉ liên kết tới các giao diện xác định bởi tùy chọn <code>interfaces</code> .	<code>no</code>	Global
<code>socket address</code>	string (địa chỉ IP)	Đặt địa chỉ IP để “nghe”, dùng cho trường hợp có nhiều giao diện ảo trên server.	Không có	Global

#### hosts allow

Tùy chọn `hosts allow` (đôi khi được viết là `allow hosts`) xác định các máy có quyền truy cập các tài nguyên được chia sẻ trên server Samba, được viết như danh sách các máy hay địa chỉ IP của chúng, cách nhau bằng dấu phẩy hoặc khoảng trống. Ta có thể đặt được một chút ít mức độ bảo mật chỉ đơn giản bằng cách đặt địa chỉ mạng con LAN của mình vào chỗ giá trị của tùy chọn này. Ví dụ:

```
hosts allow = 192.168.220. localhost
```

Chú ý là ta đặt localhost (hoặc địa chỉ 127.0.0.1) ở vị trí sau địa chỉ của mạng con. Một trong số các lỗi thường thấy khi dùng tùy chọn `hosts allow` là cấm luôn server Samba liên hệ với chính nó. Chương trình **smbpasswd** sẽ cần được kết nối với server Samba như là một client để thay đổi mật khẩu mã hóa của người dùng. Thêm nữa, việc duyệt tại chỗ cũng đòi hỏi có được đăng nhập tại chỗ.

Sau đây là các quy tắc của Samba quy định cho việc dùng các tùy chọn `hosts allow` và `hosts deny`:

- Nếu không có tùy chọn `allow` hoặc `deny` nào được xác định trong file cấu hình `smb.conf`, Samba sẽ cho phép các nối kết từ bất kỳ máy nào mà hệ thống UNIX chấp nhận.
- Nếu có các tùy chọn `allow` hoặc `deny` được xác định trong phần `[global]` của file cấu hình `smb.conf`, chúng sẽ được áp dụng cho tất cả các tài nguyên được chia sẻ, thậm chí khi một tài nguyên nào đó có tùy chọn ghi đè lên được xác định.
- Nếu chỉ có tùy chọn `allow` được xác định cho một tài nguyên được chia sẻ, chỉ có các máy được liệt kê mới có quyền truy cập tài nguyên đó. Các máy khác đều bị cấm.
- Nếu chỉ có tùy chọn `deny` được xác định cho một tài nguyên được chia sẻ, mọi máy không có trong danh sách đều có quyền sử dụng tài nguyên đó.
- Nếu cả hai tùy chọn `allow` và `deny` được xác định, một máy đã xuất hiện trong danh sách được phép thì không có mặt trong danh sách bị cấm. Nếu không, máy đó sẽ bị cấm truy cập vào tài nguyên được chia sẻ.

Chú ý: Cẩn thận tránh trường hợp bạn cho phép một máy nào đó, nhưng sau đấy lại cấm cả mạng con mà máy đó tham gia.

Ta xét tiếp một ví dụ khác:

```
hosts allow = 111.222.
hosts deny = 111.222.333.
```

Trong trường hợp này, chỉ có các máy trong mạng con 111.222.\*.\* mới có quyền truy cập vào tài nguyên được chia sẻ của Samba. Tuy nhiên, các máy thuộc mạng

con cấp thấp hơn của nó 111.222.333.\* lại bị cấm, dù cũng thuộc phạm vi tác dụng của dòng cho phép ở phía trên.

Nói thêm về cách viết tên. Tên vùng được viết với các dấu chấm, ví dụ .edu.vn sẽ chỉ tất cả các máy trong vùng edu.vn. Tên nhóm được viết sau dấu “@”, ví dụ @hut. Mạng con được viết với dấu chấm ở cuối cùng, ví dụ, 203.162.9. có nghĩa là tất cả các máy có địa chỉ IP bắt đầu bằng 203.162.9. Khóa ALL cho phép tất cả các máy client có thể truy cập được. Khóa EXCEPT được viết tiếp bởi một hay nhiều tên máy, địa chỉ IP, tên vùng, tên nhóm hoặc tên mạng con. Ví dụ, nếu muốn Samba cho phép tất cả các máy, ngoại trừ những máy trong mạng con 192.168.110 ta viết `hosts allow = ALL EXCEPT 192.168.110`. (nhớ là phải có dấu chấm ở cuối).

Việc dùng khóa ALL không phải là một ý tưởng tốt, vì không có một biện pháp bảo mật nào được dùng để bảo vệ server.

### hosts deny

Tùy chọn `hosts deny` (cũng là `deny hosts`) xác định các máy không có quyền truy cập tài nguyên được chia sẻ, được viết như danh sách các tên máy hoặc địa chỉ IP của chúng, cách nhau bằng dấu phẩy hay khoảng trống, với cú pháp giống như đối với tùy chọn `hosts allow` ở trên. Ví dụ, để hạn chế truy cập tới server từ mọi máy, trừ từ vùng `example.com`, ta có thể viết:

```
hosts deny = ALL EXCEPT .example.com
```

Giống như `hosts allow`, không có giá trị mặc định cho tùy chọn `hosts deny`. Nếu muốn cho phép hay cấm truy cập tới tài nguyên được chia sẻ cụ thể ta phải bỏ qua cả hai tùy chọn `hosts allow` và `hosts deny` trong phần `[global]` hay nếu có dùng thì phải ghi đè giá trị mới trong phần cấu hình cho tài nguyên được chia sẻ đó.

### interfaces

Tùy chọn `interfaces` liệt kê các địa chỉ mạng mà ta muốn server Samba nhận biết và đáp ứng. Tùy chọn này rất tiện lợi nếu ta muốn đặt máy tính tham gia đồng thời nhiều mạng con. Nếu không được dùng, Samba tìm giao diện mạng chính

của server (thường là card Ethernet đầu tiên) khi khởi động và tự cấu hình để hoạt động chỉ trong mạng con có giao diện mạng đó. Ta phải dùng tùy chọn này để bắt buộc Samba phải phục vụ cả các mạng con khác nữa trong mạng của ta.

Giá trị của tùy chọn là một hay nhiều bộ gồm các đôi địa chỉ IP / mặt nạ mạng, giống như trong ví dụ sau:

```
interfaces = 192.168.220.100/255.255.255.0
 192.168.210.30/255.255.255.0
```

Có thể dùng định dạng mặt nạ bit CIDR như sau:

```
interfaces = 192.168.220.100/24 192.168.210.30/24
```

Số của mặt nạ bit chỉ số bit đầu tiên được bật trong mặt nạ mạng, ví dụ số 24 nghĩa là 24 bit đầu tiên (trong số tất cả 32 bit) sẽ được kích hoạt, hay đồng nghĩa với giá trị mặt nạ mạng 255.255.255.0. Tương tự như vậy, số 16 tương đương với mặt nạ 255.255.0.0, và 8 - với 255.0.0.0.

Tùy chọn này có thể hoạt động không đúng nếu ta dùng DHCP (phân phối địa chỉ IP động).

### *bind interfaces only*

Tùy chọn `bind interfaces only` có thể được dùng để bắt buộc các tiến trình **smbd** và **nmbd** phục vụ các yêu cầu SMB chỉ cho các địa chỉ được xác định bởi tùy chọn `interfaces` mà thôi. Tiến trình **nmbd** bình thường liên kết giao diện (0.0.0.0.) trên các cổng 137 và 138 tới tất cả các địa chỉ, cho phép chúng nhận các thông báo phân phối công cộng từ khắp mọi nơi. Tuy nhiên, nếu ta ghi đè lên giá trị đó bằng:

```
bind interfaces only = yes
```

thì chỉ các gói đi từ các địa chỉ nguồn xác định thông qua tùy chọn `interfaces` mới được chấp nhận. Với **smbd**, tùy chọn này cũng bắt Samba không phục vụ các yêu cầu về file của các mạng con ngoài danh sách của tùy chọn `interfaces`. Nếu muốn cho phép có các nối kết mạng tạm thời, như dùng SLIP hoặc **ppp**, ta

không được dùng tùy chọn này. Nói chung, tùy chọn này ít được dùng, và thường chỉ có những người quản trị dày kinh nghiệm mới để ý tới nó.

Nếu đặt giá trị cho `bind interfaces only` là `yes`, ta phải thêm địa chỉ của máy tại chỗ (127.0.0.1) vào danh sách của `interfaces`, nếu không, **smbpasswd** sẽ không thể hoạt động được.

### socket address

Tùy chọn `socket address` quy định địa chỉ nào trong số được xác định bởi `interfaces` sẽ “nghe”, tức là chờ các nối kết. Samba theo mặc định chấp nhận tất cả các nối kết với tất cả các địa chỉ. Khi được dùng trong file `smb.conf`, tùy chọn này hạn chế số địa chỉ mà Samba sẽ dùng để chờ các nối kết. Ví dụ:

```
interfaces = 192.168.220.100/24 192.168.210.30/24
socket address = 192.168.210.30
```

Tùy chọn này là công cụ lập trình. Bình thường, tùy chọn này không được khuyến dùng.

## **XVI. 5. g. Các server ảo**

Các server ảo là kỹ thuật tạo cảm giác có nhiều server NetBIOS trong mạng, trong khi thật ra chỉ tồn tại có mỗi một máy, bằng cách đăng ký cho nó nhiều tên NetBIOS với các địa chỉ IP tương ứng.

Ví dụ, phòng kế toán có một server kế toán, còn các máy client của nó chỉ có thể nhìn thấy đĩa cứng cho kế toán và các máy in. Phòng marketing cũng có thể có server của mình, với tất cả báo cáo, số liệu, v.v... Tuy nhiên, để phục vụ cho tất cả các nhu cầu đó chỉ cần có một trạm làm việc UNIX hạng trung thay vì có các server cỡ nhỏ, kèm theo mỗi máy một biên chế quản trị mạng, ở khắp tất cả các phòng.

Samba cho phép một server UNIX dùng nhiều hơn một tên NetBIOS bằng tùy chọn `netbios aliases`, được nêu chi tiết ở bảng sau.



### Tùy chọn cấu hình server ảo

Tùy chọn	Thông số	Chức năng	Mặc định	Quy mô
netbios aliases	Danh sách các tên NetBIOS	Các tên NetBIOS thêm để đáp ứng các yêu cầu của client, dùng cho nhiều server Samba ảo.	Không có	Global

#### ▪ *netbios aliases*

Tùy chọn `netbios aliases` được dùng để cho phép server Samba có hơn một tên NetBIOS, mà mỗi một trong số đó được ký hiệu bằng một biểu tượng riêng trong cửa sổ *Network Neighborhood* của máy duyệt mạng. Tùy chọn này rất tiện khi ta chuyển dữ liệu của nhiều đơn vị, phòng, ban về máy chủ UNIX có đĩa cứng dung lượng lớn. Tiếp tục ví dụ trên, các câu lệnh sau làm server Samba hoạt động như thể có riêng ba server cho ba phòng kinh doanh, kế toán và quản lý:

```
[global]
```

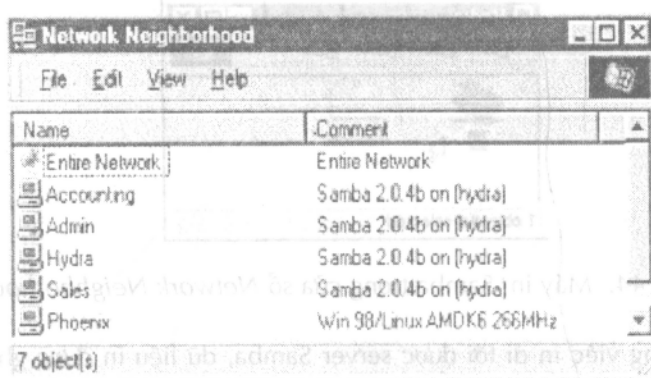
```
netbios aliases = sales accounting admin
include = /usr/local/samba/lib/smb.conf.%L
```

Trên H. XVI. 43 cho thấy máy client hiển thị các tên NetBIOS khác nhau của máy *hydra* và khi nó kết nối với Samba, ta phải chỉ ra tên của server mà nó muốn kết nối thông qua biến `%L`. Nếu server được yêu cầu là `sales`, file `/usr/local/samba/lib/smb.conf.sales` sẽ được server Samba đưa vào. File có thể chứa các khai báo chung cũng như khai báo chia sẻ chỉ để dùng cho các nhân viên của phòng kinh doanh trên như sau:

```
[global]
```

```
workgroup = SALES
hosts allow = 192.168.10.255 [sales1998]
path = /usr/local/samba/sales/sales1998/
```

```
...
```



H. XVI. 43. Dùng bí danh NetBIOS cho server Samba

## XVI. 6. In trong Samba

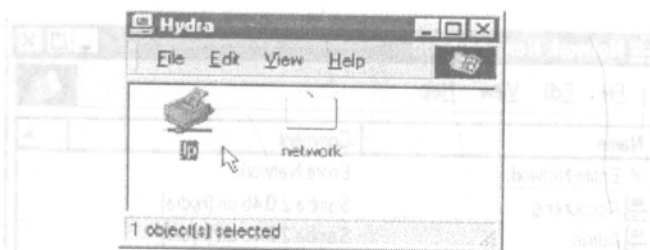
Trong phần này chúng ta sẽ tìm hiểu hai vấn đề chính là thiết lập các máy in để dùng với server Samba và cấu hình bản thân server Samba để dùng hoặc đóng vai trò server WINS.

### XVI. 6. a. Gửi công việc in tới Samba

Một máy in nối với server Samba được thể hiện trong danh sách của các tài nguyên được chia sẻ trong cửa sổ *Network Neighborhood* của máy client. Nếu máy in được đăng ký đúng cách với máy client và máy client đã có trình điều khiển cho máy in, máy client có thể dễ dàng gửi công việc in tới máy in nối vào server Samba. H. XVI. 44 cho thấy một máy in của Samba hiện trên cửa sổ *Network Neighborhood* của máy client Windows.

Để quản trị các máy in với Samba, ta cần phải hiểu rõ cơ chế in trên mạng. Việc gửi công việc in tới máy in của server Samba gồm bốn bước:

- Mở và kiểm định nối kết tới tài nguyên in được chia sẻ.
- Sao chép file cần in qua mạng.
- Đóng nối kết.
- In và xóa bản sao của file được in.



H. XVI. 44. Máy in Samba trong cửa sổ *Network Neighborhood*

Khi một công việc in đi tới được server Samba, dữ liệu in được ghi tạm thời lên đĩa cứng, vào thư mục được chỉ định riêng bởi tùy chọn path của tài nguyên in được chia sẻ. Sau đó Samba thực hiện lệnh in của UNIX để gửi file dữ liệu tới máy in, công việc in được thực hiện như là do một người dùng có quyền đối với tài nguyên in được chia sẻ này. Chú ý rằng người dùng guest (trong hệ thống UNIX là nobody) cũng có thể in được, tùy thuộc tài nguyên in được chia sẻ có được cấu hình cho người dùng đó sử dụng hay không.

#### ▪ Các lệnh in

Để có thể in được tài liệu, bạn phải cho Samba biết lệnh nào dùng để in và xóa file. Trong hệ Linux, lệnh cần dùng theo phong cách in của BSD UNIX là:

```
lpr -r -P printer file
```

Như vậy ta cho lệnh **lpr** biết phải sao tài liệu vào không gian hàng chờ, thường là /var/spool, lọc lấy tên của máy in trong file cấu hình hệ thống (/etc/printcap), và diễn giải các quy tắc tìm thấy được về cách xử lý dữ liệu và sau đó cần gửi đến thiết bị vật lý nào. Do có dùng tùy chọn **-r** file dữ liệu sau khi in xong sẽ bị xóa. Tất nhiên, ta chỉ xóa bản sao, còn nguyên bản vẫn được giữ an toàn tại máy client.

Đối với các bản UNIX System V (trong số đó có Solaris), quá trình cũng diễn ra tương tự, nhưng được thực hiện với lệnh ghép:

```
lp -d printer -s file; rm file
```

Trong System V, file `/etc/printcap` được thay bằng các tập hợp file cấu hình khác nhau nằm tại `/usr/spool/lp`, và không có tùy chọn nào để xóa file dữ liệu khi in xong. Vì thế, ta phải xóa file trên bằng tay nhờ lệnh `rm`.

#### ▪ *Các biến in*

Samba cung cấp bốn biến riêng cho việc cấu hình tác vụ in. Chúng được liệt kê trong bảng sau.

**Các biến in**

Biến	Định nghĩa
%s	Tên đường dẫn đầy đủ trên server Samba của file sẽ được in.
%f	Tên file (không đường dẫn) trên server Samba sẽ được in.
%p	Tên của máy in UNIX được dùng.
%j	Chỉ số của công việc in (dùng với các lệnh <code>lprm</code> , <code>lppause</code> , và <code>lpresume</code> )

#### ▪ *Thiết lập in tối thiểu*

Ta bắt đầu từ trường hợp đơn giản nhất. Giả sử rằng bạn đang làm việc trên hệ thống Linux và bạn có máy in tên là `lp` được liệt kê trong file định nghĩa các máy in. Các bổ sung sau đây được đưa vào file `smb.conf` sẽ làm cho máy in có thể truy nhập được qua mạng:

```
[printer1]
 printable = yes
 print command = /usr/bin/lpr -r %s
 printer = lp
 printing = BSD
 read only = yes
 guest ok = yes
```

Cấu hình này cho phép bất kỳ ai cũng có thể gửi dữ liệu tới máy in. Bây giờ ta chú ý tới biến `%s` trong tùy chọn của lệnh in. Biến này sẽ được thay bằng tên file

dầy đủ của file sẽ được in khi Samba xử lý dòng lệnh. Đối với các hệ thống UNIX cụ thể khác nhau, có các kiểu in và lệnh in cú pháp khác nhau, ta cần thay các biểu thức lệnh thích hợp vào vế phải của tùy chọn `print command` và kiểu in vào đích.

Đối với các lệnh in của các hệ thống UNIX System V, ta dùng thêm các biến như sau:

```
print command = lp -d%p -s %s; rm %s
```

Theo định nghĩa được trình bày từ trước, biến `%p` phân giải tên máy in, còn biến `%s` được dùng để ký hiệu tên file cần in. Để xác định ta đang sử dụng kiến trúc của System V, vế phải của tùy chọn `printing` sẽ được viết:

```
printing = SYSV
```

Nếu dùng bảo mật theo mức chia sẻ, ta phải chú ý đặc biệt tới tài khoản người dùng `guest` được Samba sử dụng. Tài khoản người dùng điển hình loại này của hệ thống UNIX, `nobody`, có thể không được hệ điều hành cho phép có quyền in. Trong trường hợp đó, bạn phải đặt tùy chọn `guest account` vào trong phần tài nguyên in được chia sẻ (hoặc thậm chí vào phần `[global]`) để người dùng đó có thể in được. Phương án được các tác giả của Samba chấp nhận là sử dụng tài khoản người dùng `ftp` (trong trường hợp bình thường trong hệ thống UNIX vì lý do bảo mật `ftp` hay được cấu hình sẵn là một trong số các người dùng `guest` không được tin tưởng). Ta có thể thiết lập để người dùng `ftp` có quyền in bằng lệnh sau:

```
guest account = ftp
```

Một vấn đề về in hay gặp nữa là các client có thể cần đòi hỏi được biết về trạng thái của các công việc in đã được gửi tới server Samba. Samba không dừng việc gửi tài liệu cần in tới tài nguyên in được chia sẻ đang bị bận. Vì thế, Samba cần có khả năng không chỉ giao tiếp với máy client về trạng thái công việc in hiện thời, mà còn với cả các tài liệu hiện đang chờ để được in trên máy in đó. Samba cũng cung cấp cho máy client khả năng dừng công việc in, tiếp tục lại và loại bỏ công việc in khỏi

hàng chờ in. Samba cung cấp các tùy chọn cho từng loại tác vụ trên, vay mượn các chức năng tương ứng từ các lệnh **lpq**, **lprm**, **lppause**, và **lpresume** của UNIX, và ta sẽ nghiên cứu chi tiết về chúng. Tuy nhiên, trong phần lớn các trường hợp, ta không cần thay đổi các giá trị mặc định của chúng.

Sau đây là một số điểm quan trọng, liên quan tới tài nguyên in được chia sẻ, cần phải nhớ:

- Ta cần phải đặt tùy chọn **printable = yes** vào tất cả các tài nguyên in được chia sẻ (thậm chí cả **[printers]**), để Samba biết đó là những tài nguyên in được chia sẻ. Nếu bạn quên, bỏ qua tùy chọn này, tài nguyên in được chia sẻ sẽ không thể dùng được và chỉ được coi như là tài nguyên đĩa cứng được chia sẻ.
- Nếu đặt tùy chọn **path** vào trong phần **[printer]**, mọi file gửi tới máy in đều được sao chép lại vào thư mục chỉ định, thay vì vào vị trí mặc định **/tmp**. Vì dung lượng đĩa dành cho **/tmp** trong một số hệ điều hành UNIX thường rất nhỏ, nhiều người quản trị mạng dùng thư mục **/var/spool** hoặc các thư mục khác để thay cho **/tmp**.
- Tùy chọn **read only** bị bỏ qua đối với tài nguyên in được chia sẻ.
- Nếu đặt **guest ok = yes** trong tài nguyên in được chia sẻ và Samba được cấu hình cho bảo mật theo mức chia sẻ, bất cứ ai cũng có thể gửi dữ liệu tới máy in với tư cách của người dùng **guest**.

Việc sử dụng một hay nhiều máy Samba như máy chủ in sẽ tạo khả năng mềm dẻo cho mạng LAN: ta có thể tách riêng các máy in, hạn chế thành viên nào đó truy cập tới một số hay tất cả các máy in, ... Thêm nữa, có thể hạn chế máy in cho một số người dùng hợp pháp bằng cách thêm tùy chọn **valid users** vào tài nguyên in được chia sẻ:

```
{deskjet}
 printable = yes
 path = /var/spool/samba/print
 valid users = minh chi
```

Tất cả các tùy chọn truy cập tài nguyên được chia sẻ được trình bày trong các phần trước đều có thể áp dụng cho các tài nguyên in được chia sẻ.

▪ *Tài nguyên được chia sẻ* [printers]

Trong phần *Chia sẻ đĩa cứng*, phần tài nguyên được chia sẻ [printers] được trình bày sơ lược như một tài nguyên chia sẻ đặc biệt được tạo ra một cách tự động cho những dịch vụ in. Nếu ta tạo phần tài nguyên chia sẻ có tên [printers] trong file cấu hình, Samba sẽ tự động đọc các file định nghĩa máy in và tạo các tài nguyên in được chia sẻ cho từng máy in có tên trong file đó. Ví dụ, nếu server Samba đọc thấy các máy in lp, pcl và ps trong file định nghĩa máy in, Samba sẽ cung cấp ba tài nguyên in được chia sẻ với các tên tương ứng, mỗi một trong số đó được cấu hình với các tùy chọn trong phần tài nguyên được chia sẻ [printers].

Dưới đây là ví dụ một tài nguyên in được chia sẻ [printers] cho hệ thống Linux (theo BSD UNIX). Tuy một số tùy chọn được dùng với các giá trị mặc định, để minh họa ta vẫn liệt kê ra đây cùng với các tùy chọn khác:

```
[global]
 printing = BSD
 print command = /usr/bin/lpr -P%p -r %s
 printcap file = /etc/printcap
 min print space = 2000
[printers]
 path = /usr/spool/public
 printable = true
 guest ok = true
 guest account = pcguest
```

Trong ví dụ trên ta đã quy định các tùy chọn chung để chỉ loại in theo kiểu BSD, lệnh in gửi dữ liệu tới máy in và sau đó xóa file dữ liệu tạm thời, định nghĩa máy in lấy từ file /etc/printcap và không gian in tối thiểu là 2 MB. Thêm nữa, tài nguyên in được chia sẻ [printers] được tạo ra cho mỗi máy in hệ thống. Thư mục hàng chờ tạm thời /usr/spool/public được chỉ ra bởi tùy chọn path. Một điểm quan trọng là ta phải chỉ ra các tài nguyên in được chia sẻ đó phải có thể in được - thậm chí ngay cả trong phần [printers].

---

- *Kiểm tra in*

Trong phần này ta tìm hiểu cách thức kiểm tra in trong server Samba. Giả sử ta có một trường hợp phức tạp và tài khoản người dùng guest. Đầu tiên, ta thực hiện lệnh Samba **testparm** như đã trình bày trong phần kiểm tra cài đặt server Samba trên hệ thống UNIX để kiểm tra cú pháp trong file cấu hình có chứa tài nguyên in được chia sẻ. Ví dụ, bạn có thể nhìn thấy kết quả của file cấu hình vừa nêu trên, nếu ta bỏ qua tùy chọn path:

```
testparm
Load smb config files from /usr/local/samba/lib/ \
 smb.conf
Processing configuration file "/usr/local/samba/ \
 lib/smb.conf"
Processing section "[global]"
Processing section "[homes]"
Processing section "[data]"
Processing section "[printers]"
No path in service printers - using /tmp
Loaded services file OK.
Press enter to see a dump of your service definitions
Global parameters:
load printers: Yes
printcap name: /etc/printcap
Default service parameters:
guest account: ftp
min print space: 0
print command: lpr -r -P%p %s
lpq command: lpq -P%p
lprm command: lprm -P%p %j
lppause command:
lpresume command:
Service parameters [printers]:
```



```
path: /tmp
print ok: Yes
read only: true
public: true
```

Sau đó, ta thử dùng lệnh **testprns** **printername**. Đây là một chương trình đơn giản để kiểm tra máy in xác định có tên trong file định nghĩa máy in (**printcap** chẳng hạn). Nếu file **printcap** không nằm ở vị trí thường có, với quyền root, ta phải chỉ ra vị trí của nó qua đường dẫn đầy đủ như là đối số thứ hai của lệnh **testprns**:

```
testprns lp /etc/printcap
Looking for printer lp in printcap file /etc/printcap
Printer name lp is valid.
```

Tiếp theo, truy nhập hệ thống với tư cách người dùng **guest**, chuyển đến thư mục hàng chờ, và đảm bảo rằng có thể in cũng bằng lệnh mà **testparm** quy định cho Samba sử dụng.

Cuối cùng, hãy in thử một cái gì đó thông qua server Samba bằng lệnh **smbclient**, và quan sát xem có các hiện tượng sau hay không:

- Tên công việc xuất hiện (sơ lược) trong thư mục hàng chờ của Samba, xác định bằng tùy chọn **path**.
- Công việc trên được chỉ ra trong danh sách các thiết bị in trong thư mục hàng chờ.
- Công việc in biến khỏi thư mục hàng chờ mà Samba đã dùng.

Nếu lệnh **smbclient** không thể in được, ta phải đặt lại giá trị của tùy chọn **print command** để thu thập các thông tin gỡ rối:

```
print command = /bin/cat %s >>/tmp/printlog; rm %s
```

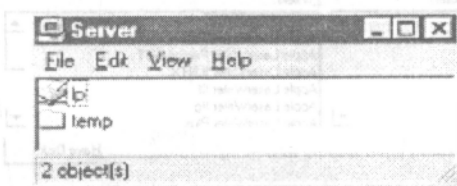
hoặc:

```
print command = echo "printed %s on %p" >> \
 /tmp/printlog
```

Lỗi chung hay bị mắc phải khi cấu hình việc in của Samba là ta hay quên dùng tên đường dẫn đầy đủ cho các lệnh. Một lỗi thường thấy nữa là không có đúng các quyền trên thư mục hàng chờ.

▪ *Thiết lập và kiểm tra client Windows*

Sau khi Samba đã cho dùng máy in, ta cần thiết lập các máy client Windows để dùng máy in được nối vào máy chủ UNIX. Ta tìm biểu tượng server Samba trong cửa sổ *Network Neighborhood*, trong đó có thể nhìn thấy máy in có tên lp được nối đến ở trên trong H. XVI. 45.



H. XVI. 45. Máy in lp trong cửa sổ *Network Neighborhood*

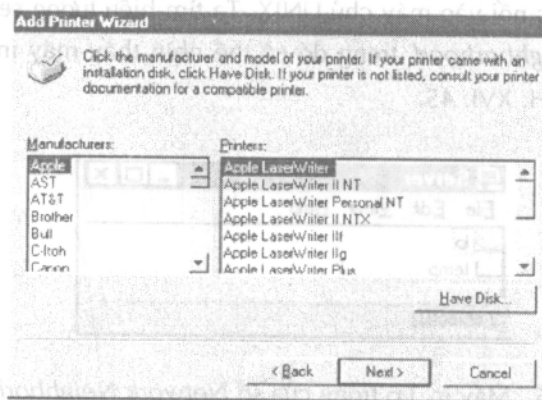
Tiếp theo, ta cần làm cho máy client Windows nhận biết được máy in, trước hết bằng động tác nhấp đúp chuột trái vào biểu tượng máy in. Nếu rơi vào tình huống chỉ định một máy in chưa được cài đặt như ta hiện giờ đang mắc phải, Windows sẽ hỏi để giúp ta cài đặt máy in đó trong hệ thống Windows. Hãy trả lời "Yes" và cửa sổ *Printer* sẽ được hiện ra.

Câu hỏi đầu tiên bạn phải trả lời là có muốn thực hiện các công việc in từ chế độ DOS hay không. Nếu máy tính PC chỉ để dùng làm client cho server Samba, ta chọn *No* và ấn tiếp nút *Next* để đi tới cửa sổ xác định kiểu máy in và tên nhà cung cấp, như trên H. XVI. 46.

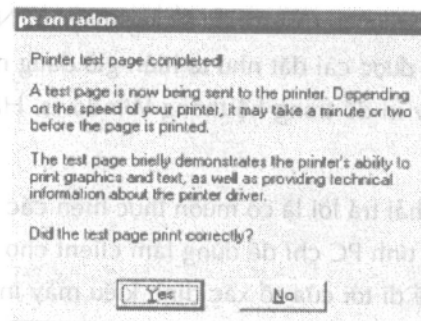
Trong hộp thoại này bạn sẽ nhìn thấy một danh sách lớn các hãng cung cấp và hầu như tất cả các kiểu máy in được đưa ra trên thị trường. Nếu bạn không tìm thấy tên máy in của mình trong danh sách đó, nhưng biết chắc chắn đó là máy in PostScript, bạn hãy chọn nhà cung cấp là *Apple* và kiểu máy in là *Apple LaserWriter*, vì với lựa chọn như vậy, bạn sẽ có được một thiết lập cơ bản cho máy

in Postscript với độ tin cậy cao. Nếu đã có máy in Postscript được nối với máy Windows, bạn có thể cập nhật hay đổi mới trình điều khiển máy in. Chú ý rằng để tránh những rủi ro có thể xảy ra khi không dùng đúng trình điều khiển máy in, ta nên duy trì trình điều khiển máy in nào đã làm việc tốt với máy tính và có sao lưu bản cũ lại trước khi tính chuyển thay đổi.

Tiếp theo, ta được yêu cầu đặt tên cho máy in. Hình H. XVI. 46 cho thấy trong ví



H. XVI. 46 Các nhà cung cấp và các loại máy in



H. XVI. 47 Hoàn thành việc in

dụ được xét, tên mặc định của máy in laser thứ hai là Apple Laserwriter. Ta đổi tên cho máy in đó thành "ps on server Samba" chẳng hạn.

Cuối cùng, máy yêu cầu in kiểm tra và ta chọn nút lệnh Yes để in và sẽ được nhận thông báo như trên H. XVI. 47.

Nếu việc in thử thất bại, ta chọn nút lệnh *No* trên H. XVI. 47 và *Printing Wizard* sẽ đưa ta trở lại bước gỡ lỗi nào đấy ở phía client. Còn nếu việc in thử diễn ra suôn sẻ, ta đã có thể sử dụng máy in này bằng các lệnh thực đơn *File* và *Print* thông thường.

- *Tự động thiết lập trình điều khiển máy in*

Trong phần trước ta đã theo dõi các bước cấu hình bằng tay trình điều khiển máy in cho hệ thống Windows. Tuy nhiên, nếu đứng từ góc độ người quản trị mạng, ta không thể luôn luôn đảm bảo rằng mọi người dùng đều thực hiện các bước cấu hình trên hoàn toàn đúng, không bị mắc lỗi nào. Rất may là ta có thể yêu cầu Samba tự động thiết lập các trình điều khiển máy in cho một máy in cụ thể nào đó.

Samba có ba tùy chọn có thể được dùng để tự động thiết lập các trình điều khiển máy in cho các máy client mới được kết nối lần đầu tiên. Đó là các tùy chọn `printer driver`, `printer driver file`, và `printer driver location`. Có thể tham khảo các thông tin chi tiết hơn về vấn đề này từ file `PRINTER_DRIVER.TXT` nằm trong bộ tài liệu của chương trình cài đặt Samba.

Có bốn bước chính như sau:

- Cài đặt các trình điều khiển cho máy in trên máy client Windows (máy in lúc này chưa được nối với máy tính).
- Tạo file định nghĩa máy in từ các thông tin trên máy Windows.
- Tạo tài nguyên được chia sẻ `PRINTER$`, nơi có thể đặt các file trình điều khiển kết quả.
- Thay đổi file cấu hình Samba một cách tương ứng.

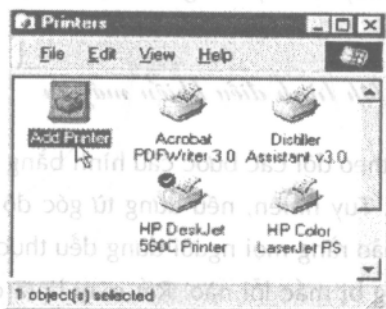
Ta sẽ lần lượt xét chi tiết hơn từng bước nói trên.

### *Thiết lập trình điều khiển trên máy client Windows*

Ta dùng các máy Windows 9x để minh họa bước này. Trước tiên, ta mở cửa sổ *Printers* của *My Computer* và nhấp đúp chuột vào biểu tượng *Add Printer* như được trình bày trên H. XVI. 48.

Từ đây, ta theo các chỉ dẫn của hộp thoại *Add Printer Wizard*. Nếu được hỏi có muốn in từ MS-DOS hay không, ta chọn *No*. Windows sẽ nạp trình điều khiển thích

hợp từ đĩa CD-ROM rồi yêu cầu thực hiện in thử. Ta lại trả lời *No* và đóng hộp thoại *Add Printer Wizard*.



H. XVI. 48 Cửa sổ *Printers*

### Tạo file định nghĩa máy in

Ta có thể tạo file định nghĩa máy in bằng cách sử dụng script `make_printerdef` trong thư mục `/usr/local/samba/bin`. Để dùng được script này, ta cần sao chép bốn file sau từ máy client Windows:<sup>2</sup>

```
C:\WINDOWS\INF\MSPRINT.INF
C:\WINDOWS\INF\MSPRINT2.INF
C:\WINDOWS\INF\MSPRINT3.INF
C:\WINDOWS\INF\MSPRINT4.INF
```

Nếu tên trình điều khiển máy in được bắt đầu từ các chữ cái A-K, ta dùng file `MSPRINT.INF` hoặc file `MSPRINT3.INF`, còn nếu được bắt đầu từ các chữ cái L-Z, hãy dùng file `MSPRINT2.INF` hoặc file `MSPRINT4.INF`. Nếu không biết tên cụ thể, có thể bạn phải mò trong cả bốn file trên để tìm cho ra tên trình điều khiển cần thiết. Đối với ví dụ của ta, file cần dùng là `MSPRINT3.INF` và file định nghĩa máy in `printers.def` được tạo ra cho máy in HP DeskJet 560C có nội dung như sau:

```
$grep "HP DeskJet 560C Printer" MSPRINT.INF
MSPRINT3.INF
```

<sup>2</sup> Các máy client chạy trên các bản Windows 95 cũ có thể chỉ có hai file đầu.

```
MSPRINT3.INF: "HP DeskJet 560C"
 Printer"=DESKJETC.DRV,HP_DeskJet_ ... "
$make_printerdef MSPRINT3.INF "HP DeskJet 560C \
 Printer" > printers.def
FOUND:DESKJETC.DRV
End of section found
CopyFiles: DESKJETC,COLOR_DESKJETC
Datasection: (null)
Datafile: DESKJETC.DRV
Driverfile: DESKJETC.DRV
Helpfile: HPVDJC.HLP
LanguageMonitor: (null) Copy the following files to \
 your printer
$ share location:
DESKJETC.DRV
HPVCM.HPM
HPVIOL.DLL
HPVMON.DLL
HPVRES.DLL
HPCOLOR.DLL
HPVUI.DLL
HPVDJCC.HLP
color\HPDESK.ICM
```

Nên ghi lại tên của các file mà script yêu cầu phải sao chép. Có thể ta còn cần đến chúng sau này.

### Tạo tài nguyên được chia sẻ PRINTER\$

Tạo tài nguyên chia sẻ có tên [PRINTER\$] trong file smb.conf và chỉ tới một thư mục trống trên server Samba. Sau đó, chép các file .INF mà script make\_printerdef đòi hỏi vào vị trí được xác định bằng tùy chọn path cho tài nguyên chia sẻ [PRINTER\$]. Ví dụ, bạn có thể đưa các dòng sau vào file cấu hình:

```
[PRINTER$]
```

```
path = /usr/local/samba/print
read only = yes
browsable = no
guest ok = yes
```

Các file mà script `make_printerdef` đòi hỏi thường nằm trong thư mục `C:\WINDOWS\SYSTEM`, nhưng bạn có thể dùng các lệnh sau đây để xác định chính xác vị trí của chúng:

```
cd C:\WINDOWS
dir filename /s
```

Trong ví dụ của ta, các file cần thiết đều được chép vào thư mục `/usr/local/samba/print` trên server Samba. Và ta còn phải chép file `printers.def` vừa tạo vào tài nguyên được chia sẻ đã nói tới.

### Sửa file cấu hình Samba

Bước cuối cùng là thêm ba tùy chọn `printer driver`, `printer driver file`, và `printer driver location` vào file cấu hình của Samba. Tùy chọn chung `printer driver file` chỉ đến file `printers.def` phải được đặt vào phần `[global]`. Các tùy chọn còn lại được đặt vào phần tài nguyên máy in được chia sẻ mà ta muốn cấu hình một cách tự động các trình điều khiển máy in. Giá trị cho `printer driver` phải trùng với xâu ký tự được hiện ra trong *Printer Wizard* trên hệ thống *Windows*. Giá trị của `printer driver location` là đường dẫn của tài nguyên `PRINTER$` mà ta thiết lập, chứ không phải đường dẫn UNIX trên server. Do đó, ta có thể dùng các dòng mã sau đây trong file cấu hình Samba (dấu `\` để nối dòng, như ta vẫn hay dùng trong các ví dụ đã dẫn):

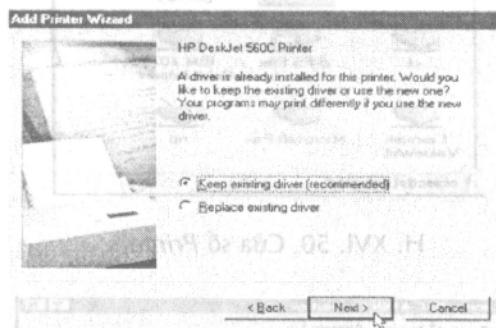
```
[global]
 printer driver file = \
 /usr/local/samba/print/printers.def
[hpdeskjet]
 path = /var/spool/samba/printers
```

```
printable=yes
printer driver = HP DeskJet 560C Printer
printer driver location = \\%L\PRINTER$
```

Bây giờ ta đã có thể tiến hành kiểm tra. Cần loại bỏ máy in Windows mà ta đã “thiết lập” trong bước đầu tiên khỏi danh sách các máy in trong cửa sổ *Printers* của *My Computer*. Nếu Samba yêu cầu xóa các file không cần thiết, ta hãy làm theo. Các file đó sẽ được thay thế rất nhanh sau đó trên máy client vì chúng đã tồn tại trên server Samba.

### Kiểm tra cấu hình

Ta khởi động lại các daemon Samba và quan sát tài nguyên chia sẻ [hpdeskjet] nằm dưới tên máy trong cửa sổ *Network Neighborhood*. Từ thời điểm này, nếu bạn kích chuột vào biểu tượng máy in, bạn sẽ bắt đầu quá trình thiết lập máy in và đi đến hộp thoại trên H. XVI. 49.



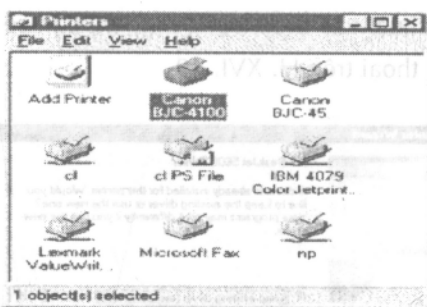
H. XVI. 49. Tự động cấu hình trình điều khiển máy in

Hộp thoại này khác hẳn hộp thoại mà bạn nhìn thấy từ trước, khi bạn thiết lập máy in. Hộp thoại sẽ hỏi bạn có giữ trình điều khiển “đã được cài đặt” - nói cách khác - được Samba đưa ra. Ta cần khẳng định bằng cách ấn phím lệnh *Next* để đi tiếp. Việc cuối cùng còn lại là in thử trước khi có thể sử dụng được máy in.

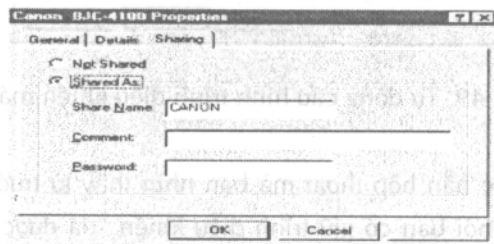


### XVI. 6. b. In trên các máy in của client Windows

Trong thực tế sử dụng, ta thường thấy các máy in được nối với các máy client chạy Windows 9x hay NT 4.0 và được truy cập từ phía Samba. Samba có một tiện ích tên là **smbprint** dùng để xếp hàng chờ cho các tác vụ in tới các máy in trên cơ sở Windows. Để có thể sử dụng được tiện ích đó, máy in nối với các máy client Windows phải được thiết lập như các tài nguyên chia sẻ được. Nếu chưa hoàn thành bước chuẩn bị đó, ta cần thiết lập lại trong cửa sổ *Printers*, đi từ phím *Start* quen thuộc của Windows, như được thấy trên H. XVI. 50. Ta chọn máy in nối với máy Windows tại chỗ (trong ví dụ là máy in Canon), dùng chuột phải để hiện thực đơn, rồi chọn *Sharing*. Bảng *Sharing* của khung *Printer Properties* sẽ hiện lên như trong H. XVI. 51. Nếu muốn mở máy in tự do cho tất cả các người dùng trong LAN, ta để trống ô mật khẩu.



H. XVI. 50 Cửa sổ *Printers*



H. XVI. 51 Bảng *Sharing* cho máy in

Sau khi máy in đã được chia sẻ, ta có thể thêm máy in đó vào danh sách các máy in chuẩn và Samba có thể làm cho nó truy cập được đối với tất cả các PC trong nhóm làm việc. Để giúp cho quá trình cài đặt được dễ dàng hơn, bộ cài đặt Samba

cung cấp hai scripts mẫu: `smbprint` và `smbprint.sysv`. Script đầu làm việc với các máy in kiểu BSD; script thứ hai - với các máy in kiểu System V. Chú ý rằng ta gọi máy in thuộc “kiểu” nào là do tùy thuộc vào bản UNIX được dùng trên server là loại BSD hay System V.

#### ▪ *Các máy in RPL*

Ta phải thực hiện hai bước để BSD UNIX nhận biết được các máy in từ xa. Trước hết, cần sửa file `/etc/printcap` và thêm dòng vào cho máy in từ xa cụ thể mà ta có. Nhớ rằng dòng vào lọc đầu vào (input filter - `if`) cần phải có mặt để chỉ tới chương trình **smbprint**, nếu máy client chạy trên Windows 9x. Bộ các dòng vào sau là thích hợp với máy chạy trên hệ thống Linux:

```
laserjet: \
 :sd=/var/spool/lpd/laser: \
 # spool directory
 :mx#0: \
 # maximum file size (none)
 :sh: \
 # surpress burst header (no)
 :if=/usr/local/samba/bin/smbprint:
 # text filter
```

Sau đó, ta cần tạo file cấu hình trong thư mục hàng chờ được chỉ định (ví dụ, trong `/var/spool`) bởi thông số `sd`. Tất nhiên là trong hệ thống UNIX ta đã phải tạo ra thư mục đó sẵn. File cấu hình này cần phải có tên với đuôi `.config` và chứa các thông tin sau:

- Tên NetBIOS của máy Windows mà máy in được nối với nó.
- Tên dịch vụ mà máy in đó mang.
- Mật khẩu truy cập.

Hai thông số cuối cùng đã được xác định trong hộp thoại *Sharing* trong máy Windows. Trong ví dụ này, file `.config` sẽ gồm ba dòng vào:

```
server = phoenix
```

---

```
service = CANON
password = ""
```

Sau khi thực hiện xong hai bước trên, ta khởi động lại máy có server Samba và thử in bằng lệnh in chuẩn nào đó của UNIX.

#### ▪ *Các máy in System V*

Việc gửi các công việc in từ hệ thống UNIX System V tới máy client để in đơn giản hơn so với trường hợp trước. Ta cần phải có script `smbprint.sysv` trong thư mục `/usr/local/samba/examples/printing` và thực hiện các bước như sau:

- Thay một cách tương ứng các thông số về tên server, dịch vụ, và mật khẩu trong script đó để trùng với tên NetBIOS của máy, tài nguyên máy in được chia sẻ, và mật khẩu cụ thể. Ví dụ, nếu vẫn giữ các điều kiện của ví dụ trước, nhưng áp dụng cho UNIX System V, ta sẽ có các dòng vào như sau:

```
server = phoenix
service = CANON
password = ""
```

- Ta thực hiện các lệnh sau để tạo ra phần tham chiếu cho máy in cụ thể của ta, tại ví dụ được xét là máy in có tên `canon_printer`, trong file định nghĩa máy in như là một máy in UNIX mới:

```
lpadmin -p canon_printer -v /dev/null -i \
./smbprint.sysv
enable canon_printer
accept canon_printer
```

- Sau đó, khởi động lại các daemon của Samba và in thử bằng cách dùng lệnh in chuẩn của UNIX.

#### ▪ *Các tùy chọn in của Samba*

Bảng sau tổng kết các tùy chọn cấu hình in của Samba.

### Các tùy chọn cấu hình in

Tùy chọn	Thông số	Chức năng	Mặc định	Quy mô
printing	bsd, sysv, hpux, aix, gnx, plp, softq, hay lprng	Đặt loại hệ thống in cho theo hệ thống UNIX.	Phụ thuộc hệ thống	Share
printable (print ok)	Boolean	Đánh dấu tài nguyên chia sẻ là tài nguyên in được chia sẻ.	no	Share
printer (printer name)	Xâu ký tự (tên máy in UNIX)	Đặt tên máy in để hiện ra cho các máy client.	Phụ thuộc hệ thống	Share
printer driver	Xâu ký tự (tên trình điều khiển máy in)	Nêu tên trình điều khiển mà máy client sẽ dùng để gửi dữ liệu tới máy in.	Không có	Share
printer driver file	Xâu ký tự (đường dẫn xác định)	Nêu tên file trình điều khiển máy in.	Không có	Global
printer driver location	Xâu ký tự (đường dẫn mạng)	Xác định đường dẫn của tài nguyên chia sẻ cho file trình điều khiển máy in.	Không có	Share
lpq cache time	số (giây)	Đặt tổng số thời gian mà Samba sẽ lưu lại trạng thái của lpq.	10	Global
postscript	Boolean	Xử lý tất cả các công việc in như là postscript bằng cách treo dấu %! ở đầu mỗi file.	no	Share

load printers	Boolean	Tự động nạp mỗi một máy in được định nghĩa trong file <code>printcap</code> như tài nguyên in được chia sẻ.	no	Global
print command	Xâu ký tự (lệnh shell)	Đặt lệnh in của UNIX để thực hiện việc in.	Xem tiếp ở dưới	Share
lpq command	Xâu ký tự (lệnh shell)	Đặt lệnh UNIX để trả về trạng thái của hàng chờ in.	Xem tiếp ở dưới	Share
lprm command	Xâu ký tự (lệnh shell)	Đặt lệnh UNIX để loại công việc in khỏi hàng chờ in.	Xem tiếp ở dưới	Share
lppause command	Xâu ký tự (lệnh shell)	Đặt lệnh UNIX để dừng công việc in trong hàng chờ in.	Xem tiếp ở dưới	Share
lpresume command	Xâu ký tự (lệnh shell)	Đặt lệnh UNIX để hủy bỏ việc dừng công việc in trong hàng chờ in.	Xem tiếp ở dưới	Share
printcap name ( <code>printcap</code> )	Xâu ký tự (đường dẫn xác định)	Đặt vị trí của file định nghĩa máy in.	Phụ thuộc hệ thống	Global
min print space	số (kB)	Đặt dung lượng tối thiểu của đĩa cứng cho các dịch vụ in.	0	Share
queuepause command	Xâu ký tự (lệnh shell)	Đặt lệnh UNIX để dừng hàng chờ in.	Xem tiếp ở dưới	Share
queueresume command	Xâu ký tự (lệnh shell)	Đặt lệnh UNIX để hủy bỏ việc dừng hàng chờ in.	Xem tiếp ở dưới	Share

printing

Tùy chọn cấu hình printing cho Samba biết về hệ thống in UNIX được dùng. Có vài họ lệnh UNIX khác nhau dùng để điều khiển việc in và thông báo về trạng thái in. Samba hỗ trợ bảy trong số đó, như được liệt kê trong bảng dưới đây.

**Các họ lệnh in UNIX**

<b>Biến</b>	<b>Định nghĩa</b>
BSD	Berkeley UNIX system
SYSV	System V
AIX	AIX Operating System (IBM)
HPUX	Hewlett-Packard UNIX
QNX	QNX Realtime Operating System (QNX)
LPRNG	LPR Next Generation (Powell)
SOFTQ	SOFTQ system
PLP	Portable Line Printer (Powell)

Giá trị mặc định của tùy chọn này là phụ thuộc vào hệ thống UNIX cụ thể và được cấu hình khi Samba lần đầu tiên được biên dịch. Đối với hầu hết các hệ thống UNIX, script cấu hình sẽ tự động phát hiện thiết bị in sẽ được dùng và cấu hình nó trong file `makefile` của Samba. Tuy nhiên, nếu hệ thống in là PLP, LPRNG, hoặc QNX, ta phải xác định cụ thể thêm trong file `makefile` hoặc tài nguyên in được chia sẻ.

Các loại hệ thống UNIX chung nhất là BSD và SYSV. Các máy in trên server BSD UNIX được mô tả trong file định nghĩa máy in - thường là `/etc/printcap`.

Việc cấu hình in tự động kéo theo cấu hình của ít nhất ba tùy chọn in khác nữa là `print command`, `lpq command`, và `lprm command`. Nếu bạn chạy Samba trên hệ thống UNIX nào không hỗ trợ các lệnh in trên, bạn sẽ phải tự thiết lập bằng tay.

### printable

Tùy chọn `printable` cần phải nhận giá trị để xác định tài nguyên chia sẻ ta đang muốn cấu hình là tài nguyên in được chia sẻ. Nếu tùy chọn này không được dùng, tài nguyên in được hiểu đơn giản chỉ là sự chia sẻ đĩa cứng mà thôi. Bạn có thể thiết lập như sau:

```
[printer1]
 printable = yes
```

### printer

Tùy chọn này, đôi khi còn được gọi là `printer name`, xác định tên của máy in trên server mà tài nguyên được chia sẻ chỉ đến. Tùy chọn không có giá trị mặc định nên ta phải xác định giá trị của nó trong file cấu hình Samba, dù bình thường các hệ thống UNIX đã nhận biết máy in với tên mặc định là `lp`. Ví dụ:

```
[deskjet]
 printer = hpdkjet1
```

### printer driver

Tùy chọn `printer driver` đặt xâu ký tự mà Samba dùng để chỉ cho Windows xác định máy in. Nếu tùy chọn này được đặt đúng thì tự *Windows Printer Wizard* sẽ nhận diện được máy in, các người dùng cuối ở các máy client Windows sẽ cài đặt máy in dễ dàng hơn, đỡ phải suy nghĩ để trả lời các câu hỏi. Vì thế, xâu ký tự (giá trị của tùy chọn) phải trùng với xâu ký tự hiện ra trên *Printer Wizard*, như được chỉ ra trên H. XVI. 52. Ví dụ, máy in Apple LaserWriter thường dùng xâu `Apple LaserWriter`; còn đối với máy in Hewlett Packard Deskjet 560C ta dùng `HP DeskJet 560C Printer`.

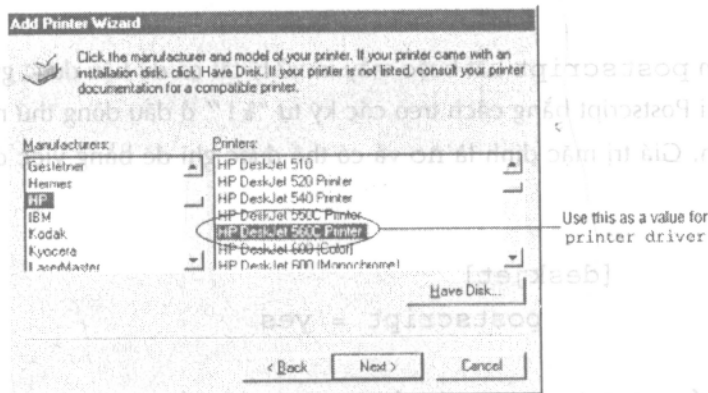
### printer driver file

Tùy chọn chung này cho biết vị trí của file định nghĩa trình điều khiển máy in cho Windows 9x, cần thiết để cung cấp cho các client dùng máy in của Samba. Giá

trị mặc định `/usr/local/samba/lib/printers.def` của tùy chọn có thể được ghi đè:

```
[deskjet]
```

```
printer driver file = /var/printers/printers.def
```



H. XVI. 52. Hộp thoại *Add Printer Wizard* của Windows 98

### printer driver location

Tùy chọn xác định tài nguyên chia sẻ cụ thể có chứa trình điều khiển máy in Windows 9x và file định nghĩa máy in và không có giá trị mặc định. Ta có thể chỉ vị trí bằng đường dẫn mạng, nhưng biện pháp thường được áp dụng là dùng luôn tài nguyên được chia sẻ ngay trên máy của bạn, như nội dung của các dòng sau:

```
[deskjet]
```

```
printer driver location = \\%L\PRINTERS
```

### lpq cache time

Tùy chọn chung `lpq cache time` cho phép đặt thời gian bằng số giây mà Samba sẽ nhớ trạng thái in hiện thời. Khi khoảng thời gian đó đã hết, Samba sẽ đưa ra lệnh `lpq` (hoặc lệnh nào được xác định bởi tùy chọn `lpq command`) để thu thập trạng thái được cập nhật. Giá trị mặc định là 10 giây, nhưng có thể được tăng



thêm nếu lệnh **lpq** chạy quá lâu hoặc có số lượng lớn các client. Ví dụ sau sẽ tăng thời lượng đó lên 30 giây:

```
[deskjet]
lpq cache time = 30
```

### postscript

Tùy chọn **postscript** bắt buộc máy in xử lý tất cả dữ liệu được gửi tới nó như là thuộc loại Postscript bằng cách treo các ký tự “%!” ở đầu dòng thứ nhất của mỗi công việc in. Giá trị mặc định là **no** và có thể được ghi đè bằng việc đặt lại giá trị như sau:

```
[deskjet]
postscript = yes
```

### Các tùy chọn print command, lpq command, lprm command, lppause command, lpresume command

Các tùy chọn này cho Samba biết các lệnh UNIX nào được dùng để giám sát và gửi dữ liệu tới máy in. Các lệnh UNIX đó là: **lpr** (gửi tới Line **P**rinter), **lpq** (List Printer **Q**ueue - đọc danh sách hàng chờ máy in), **lprm** (Line Printer **R**e**M**ove - loại bỏ máy in dòng), kèm theo các lệnh tùy chọn **lppause** và **lpresume**. Ví dụ, ta xét:

```
lpq command = /usr/ucb/lpq %p
```

dòng lệnh này đặt **lpq command** để dùng **/usr/ucb/lpq**. Tương tự như vậy:

```
lprm command = /usr/local/lprm -P%p %j
```

sẽ đặt lệnh loại máy in Samba để dùng **/usr/local/lprm**, và cung cấp chỉ số thứ tự công việc in bằng biến **%j**.

Các giá trị mặc định cho mỗi một tùy chọn trên đều phụ thuộc vào giá trị của tùy chọn **printing**. Trên bảng dưới đây là các lệnh mặc định cho mỗi một giá trị của

tùy chọn printing, tùy theo hệ UNIX nào được sử dụng. Ta chú ý nhiều đến hai cột đầu với các hệ tiêu biểu như Linux (BSD) và Solaris (SYSV), là các hệ điều hành phổ biến ở Việt Nam hiện nay.

### Các lệnh mặc định cho các giá trị khác nhau của các lệnh in

Tùy chọn (các lệnh)	BSD, AIX, PLP, LPRNG	SYSV, HPUX	QNX	SOFTQ
print	lpr -r -P%p %s	lp -c -d%p %s; rm %s	lp -r -P%p %s	lp -d%p -s %s; rm %s
lpq	lpq -P%p	lpstat -o%p	lpq -P%p	lpstat -o%p
lprm	lprm -P%p %j	cancel %p-%j	cancel %p-%j	cancel %p-%j
lppause	lp -i %p-%j -H hold (chỉ cho SYSV)	Không có	Không có	Không có
lpresume	lp -i %p-%j -H resume (chỉ cho SYSV)	Không có	Không có	qstat -s -j %j -r

Hệ thống in phổ biến nhất là BSD. Thông thường ta không phải thay đổi các tùy chọn trong Samba, ngoài việc thêm dòng lệnh cho những hệ thống UNIX không hỗ trợ tùy chọn -r (xóa sau khi in). Ví dụ, dòng lệnh đó là:

```
/usr/local/lpr -P%p %s; /bin/rm %s
```

Thậm chí, có thể dùng các tùy chọn trong smb.conf để gỡ rối như sau:

```
print command = cat %s >>/tmp/printlog;lpr -r -P%p %s
```

để kiểm tra xem có phải các file thực sự đã được chuyển đến server Samba hay không. Nếu đúng như vậy, nội dung của chúng phải được chỉ ra trong file tmp/printlog.

Hệ thống in phổ biến thứ hai là SYSV (ký hiệu của System V), và thêm một vài biến thể của nó như AIX của IBM và HP-UX của Hewlett-Packard. Các hệ thống này không có file `/etc/printcap`. Thay vào đó, tùy chọn `printcap` file có thể được đặt nhằm mục đích dùng lệnh tương ứng `lpstat` đối với hệ thống UNIX System V để lấy danh sách các máy in. Hoặc có thể dùng tùy chọn chung `printcap name` để đặt tên cho file `printcap` giả. Trong trường hợp cuối, file đó phải chứa chuỗi các dòng có dạng như sau:

```
lp|print1|My Printer 1
 print2|My Printer 2
 print3|My Printer 3
```

Mỗi dòng đó đặt tên máy in và cung cấp các bí danh cho nó. Trong ví dụ được nêu ở đây, máy in đầu tiên có tên là `lp`, `print1`, hoặc `My Printer 1`, tùy thuộc vào ý thích của người dùng. Tên thứ nhất sẽ được dùng vào vị trí giá trị của biến `%p` trong mọi lệnh Samba được gọi thực hiện cho máy in đó.

Hai loại máy in được Samba hỗ trợ thêm là: LPRNG (**LPR New Generation**) and PLP (**Public Line Printer**), là các hệ thống in cho vùng công cộng và mã nguồn mở, được rất nhiều site sử dụng để khắc phục các trở ngại liên quan tới phần mềm hỗ trợ của các nhà cung cấp. Ngoài ra ta còn phải kể đến hai loại hệ thống in thời gian thực SOFTQ và QNX cũng được Samba hỗ trợ.

### load printers

Tùy chọn `load printers` mách bảo Samba tạo các tài nguyên được chia sẻ cho tất cả các tên máy in đã biết và nạp các tài nguyên đó vào danh sách duyệt. Samba sẽ tạo và liệt kê tài nguyên in được chia sẻ cho mỗi một tên máy in trong file `/etc/printcap` (hoặc file tương đương trong các hệ thống không có file `printcap`, như `/etc/terminfo` hoặc `/etc/termcap` trong các bản Solaris từ phiên bản 2.5 trở về sau). Ví dụ, nếu file `printcap` có dạng:<sup>3</sup>

---

<sup>3</sup> Các chú thích được viết thêm để dễ theo dõi. Trong các file `printcap` thật sự thường không có các chú thích này.

```
lp:\
:sd=/var/spool/lpd/lp:\
 # thư mục spool
:mx#0:\
 # kích thước cực đại của file (không có)
:sh:\
 # bỏ tiêu đề (no)
:lp=/dev/lp1:\
 # tên thiết bị cho đầu ra
:if=/var/spool/lpd/lp/filter:
 # lọc văn bản cho máy in laser:\
:sd=/var/spool/lpd/laser:\
 # thư mục spool
:mx#0:\
 # kích thước cực đại của file (không có)
:sh:\
 # bỏ tiêu đề (no)
:lp=/dev/laser:\
 # tên thiết bị cho đầu ra
:if=/var/spool/lpd/lp/filter:
 # lọc văn bản
```

và ta có:

```
load printers = yes
```

thì các tài nguyên chia sẻ [lp] và [laser] sẽ được tự động tạo ra như những tài nguyên in được chia sẻ hợp lệ khi Samba khởi động. Cả hai tài nguyên này sẽ lấy các tùy chọn được xác định trong phần [printers] để tự cấu hình, và sẽ có mặt trong danh sách duyệt đối với server Samba.

### *printcap name*

Nếu tùy chọn `printcap name` (còn được gọi là `printcap`) xuất hiện trong tài nguyên in được chia sẻ, Samba sẽ dùng file được xác định như file định nghĩa

máy in. Giá trị bình thường của tùy chọn là `/etc/printcap`, tuy nhiên ta có thể thay đổi để dùng file nào chỉ chứa tên các máy in mà bạn muốn chia sẻ cho mạng. Giá trị của tùy chọn phải là tên được xác định của file định nghĩa máy in trên server:

```
[deskjet]
 printcap name = /usr/local/printcap
```

#### *min print space*

Tùy chọn `min print space` đặt tổng số dung lượng hàng chờ có được trong đĩa cứng trước khi công việc in được thực hiện. Đặt giá trị bằng 0 (giá trị mặc định) sẽ tắt đi việc kiểm tra; đặt bằng các số khác sẽ cho số dung lượng trống theo KB. Tùy chọn này giúp tránh bị các công việc in làm tràn không gian đĩa trên server, dẫn đến ảnh hưởng xấu tới các tiến trình khác:

```
[deskjet]
 min print space = 4000
```

#### *queuepause command*

Tùy chọn này xác định lệnh để Samba biết cách dừng hoàn toàn hàng chờ in, chứ không chỉ một công việc in cụ thể nào đó trong hàng chờ. Giá trị mặc định phụ thuộc vào hệ thống UNIX mà Samba được cài đặt. Bạn không cần phải thay đổi giá trị của tùy chọn này.

#### *queueresume command*

Tùy chọn này xác định lệnh để Samba biết cách tiếp tục hàng chờ in đã bị dừng. Giá trị mặc định phụ thuộc vào hệ thống UNIX mà Samba được cài đặt. Bạn không cần phải thay đổi giá trị của tùy chọn này.

## Thay lời kết luận

Trên đây, bạn đã theo dõi những kiến thức cơ bản về hệ điều hành UNIX. Tác giả đã cố gắng trình bày đầy đủ các khái niệm cơ động nhất của UNIX, sao cho các kiến thức cơ bản không bị lạc hậu và luôn luôn giúp ích cho người đọc, nhất là đối với những người mới làm quen với UNIX (và Linux, nói riêng). Mặt khác, tác giả cũng cập nhật các kiến thức mới trong lĩnh vực phát triển cực kỳ nhanh chóng này.

Tác giả cũng hy vọng rằng các vấn đề quản trị mạng, cũng như các kỹ năng, kinh nghiệm cụ thể được đưa ra trong cuốn sách này, là mang tính cấp thiết, đáp ứng được nhu cầu phổ biến và sử dụng các hệ thống UNIX ở Việt Nam và thiết thực giúp ích trong công việc chuyên môn của những người làm tin học.

Tuy nhiên, cũng phải nhìn nhận rằng, ngoài những khiếm khuyết có thể có về mặt chủ quan và khuôn khổ của cuốn sách, tác giả cũng chưa đủ điều kiện để đề cập tới những vấn đề quản trị mạng khác cũng rất quan trọng và cấp thiết mà bản thân tác giả cũng thấy rất hứng thú. Hy vọng rằng trong tương lai gần chúng ta có thể cùng nghiên cứu về những vấn đề mới đó.

Mặt khác, thực hành nhiều và thường xuyên là một yêu cầu bắt buộc để có thể nắm vững và sử dụng nhuần nhuyễn hệ thống và mạng UNIX. Thông thường, đối với người học, thực hành trên máy phải chiếm tối thiểu một nửa tổng lượng thời gian học. Các bài tập được đưa ra ở cuối sách nhằm gợi ý cho người đọc – chúng sẽ được bổ sung đáng kể và biên tập lại trong lần tái bản sau.

## XVII. Bài tập

### XVII. 1. Các bài tập dành cho người dùng cuối

#### *XVII. 1. a. Bài tập UNIX-1*

##### *1. Gia nhập hệ thống từ máy trạm*

Tại dấu nhắc MS-DOS hay bằng lệnh thực đơn *Run* trong Win 9x/Me/XP, gõ:

```
C:\> telnet
```

Màn hình sẽ bắt đầu hiện phiên làm việc với máy chủ bằng dòng thông báo:

```
login : <ten nguoi su dung>
```

Hãy điền vào tên người sử dụng, ấn phím “**Enter**” và đưa vào mật khẩu. Nếu gõ sai thì phải vào lại.

```
password: <mat khau cua nguoi su dung>
```

Nếu thao tác đúng, trên màn hình sẽ hiện ra dấu nhắc hệ thống \$.

##### *2. Ra khỏi hệ thống*

Có lệnh \$ **exit**, hoặc ^D, hoặc ^X

##### *3. Thử một số lệnh cơ bản theo thứ tự sau*

```
$ logname
```

```
$ id
```

---

```
$ date
$ uname
$ hostname
$ echo Hello! Happy working day
$ cal
$ cal 9 1969
$ clear
$ pwd
$ cd ..
$ cd /
$ cd
$ pwd
$ mkdir test1
$ ls
$ ls -l
$ ls -l /bin
$ passwd
$ who
$ who am i
$ groups
$ wall
$ write
$ talk
$ mesg [n,y]
$ ifconfig
$ find
$ grep
$ ufsdump
```

...

- \* Nên dùng các trang **man** để tìm hiểu về các lệnh và chủ quyền của chúng.
- \* Khi viết sai cú pháp của lệnh có thể dùng **^w** để xóa một từ, hoặc dùng **^u** để xóa cả dòng.



\* Có thể tăng tốc độ hiển thị của màn hình bằng lệnh `$ stty 38400`

#### *4. . Một số thao tác*

- Hiển thị dòng chữ sau lên màn hình:

Hom nay la ngay may man, chung ta se lam viec tot !!!

- Xóa sạch màn hình
- Cho biết ngày tháng và giờ của hệ thống
- Cho biết thư mục hiện thời
- Chuyển tới thư mục `/bin`
- Cho biết các file và thư mục trong `/bin`
- Tạo thư mục con `thuctap` trong thư mục làm việc
- Đổi mật khẩu, thoát ra khỏi hệ thống, sau đó, đăng nhập lại với mật khẩu mới
- Sử dụng lệnh **man** để xem cú pháp và ý nghĩa của các lệnh
- Sử dụng lệnh **wall** để gửi thông báo cho tất cả người dùng
- Sử dụng lệnh **write** để thông báo cho một người nào đó, cho terminal nào đó
- Dùng lệnh **talk** để thực hiện một cuộc nói chuyện với một người nào đó
- Sử dụng lệnh **mesg** để cho / không cho phép việc gửi các thông báo

#### ***XVII. 1. b. Bài tập UNIX-2***

1. Cho biết các thư mục ở root.
2. Xem kích thước các file trong thư mục `/bin`. Hiển thị danh sách các file theo từng trang màn hình.
3. Tạo ba thư mục `tm1`, `tm1/tm2`, `tm1/tm2/tm3` bằng một lệnh duy nhất, thực hiện một lần.
4. Xóa ba thư mục vừa được tạo ra cùng bằng một lệnh duy nhất.
5. Tạo một file có tên `HOI_DOAI` trong thư mục làm việc để ghi tỷ giá ngoại tệ (nội dung tùy ý).

6. Tạo một file có tên GIA trong thư mục làm việc để ghi giá một số mặt hàng thông dụng.
7. Nối hai file trên vào cuối một file có tên là THI\_TRUONG trong thư mục con tm của thư mục làm việc.
8. Kiểm tra lại nội dung của file THI\_TRUONG. Tạo file CAPNHAT ghi một số thông tin mới, rồi nối file này vào đầu của file THI\_TRUONG.
9. Tạo hai thư mục tt1 và tt2. Chép các file ở thư mục /etc vào tt1 (lưu ý: nếu dung lượng đĩa cứng nhỏ thì chỉ chọn chép một số file. Có cả những file không thể chép được vì bạn không có quyền trên chúng).
10. Trong tt1 tạo thư mục save\_pp để lưu thêm một bản sao các file p\* của tt1.
11. Chuyển các file có tên bắt đầu bằng f từ tt1 sang tt2.
12. Tìm tất cả các file có tên pass\* trên hệ thống.
13. Tìm tất cả các file có tên f\* trên thư mục làm việc.
14. Đổi tên tất cả các file có tên viết bằng chữ hoa trên màn hình sang các tên viết bằng chữ in thường. Xóa tất cả các thư mục con trong thư mục làm việc.
15. Trong thư mục làm việc tạo các thư mục con sau với các thuộc tính chủ quyền:

```
tt1: drwxr-x-w-
tt2: dr--rw-rw-
tt3: drwx-wx-wx
tt4: drwxrwxrwx
```

16. Trong thư mục làm việc tạo các file sau:

```
tin1, tin2, tin3 với các thuộc tính chủ quyền: -rwx-w-rw-
tin4, tin5, tin6 với các thuộc tính chủ quyền: -rw-r-x-w-
```

Nội dung của các file là tùy ý.

17. Đổi chủ quyền tương ứng sau cho các file:

```
tin1 -rwx-w-r-x
tin2 -rw-r-xrw-
tin3 -r-x-w-r-x
```

---

tin4	-r-x-w-r-x
tin5	-rw-rw-rw-
tin6	-rwxrwxrwx

18. Câu này chỉ có tính chất minh họa khả năng trao đổi thông tin bằng cách cấp chủ quyền xâm nhập thư mục làm việc và file dữ liệu. (Hạn chế dùng trong thực tế). Đổi chủ quyền thư mục làm việc về 777. Kiểm tra chủ quyền của các thư mục khác. Nếu có thể, thâm nhập và ghi thông tin từ file `tin1` của mình vào cuối file `tin3` của người khác. Kiểm tra lại xem nội dung file `tin3` đã có thông tin của mình chưa.

19. Dùng lệnh `$ find` để tìm trong `/etc`:

- Tìm theo tên `*pp*`
- Tìm theo số inode 2022
- Tìm theo tên người sở hữu `ct100ppp`
- Tìm theo thời gian file bị thay đổi sau 3 ngày so với thời điểm thực hiện lệnh
- Tìm file `asppp.cf` trong `/export`. Lập lại việc tìm kiếm trong thư mục `/etc`. Để tránh các thông báo lỗi làm nhiễu màn hình, ta hướng đầu ra của sai chuẩn tới thiết bị không (`/dev/null`).

20. Dùng lệnh `ln` để liên kết file `thuctap` với tên file `thuctap_link`. Dùng lệnh `ls -l`, ta có nhận xét gì về hai file `thuctap` và `thuctap_link`. Sau đó xóa file `thuctap`. Dùng lệnh `ls -l` rồi nhận xét.

### ***XVII. I. c. Bài tập UNIX-3***

1. Dùng `vi` để soạn file `btvi1` với nội dung và bố trí như sau:

```
There are times
when you make me laugh
There are moments
when you drive me mad
There are seconds
when I see the light
Though many times you made me cry
There's something you don't understand
I want to be your man
```

---

Nothing to lose  
You love to win  
hoping so bad  
that you'll let me in  
I'm at your feet  
waiting for you  
I've got time  
and nothing to lose

There are times  
when I believe in you  
These moments  
when I feel close to you  
There are times  
I think that I am yours  
Though many times I feel unsure  
There's something you don't understand  
I want to be your man

Nothing to lose  
You love to win  
hoping so bad  
that you'll let me in  
I'm at your feet  
waiting for you  
I've got time  
and nothing to lose  
I'll always be around you  
keep an eye on you  
Because my patience is strong  
and I won't let you run  
Because you are only one  
Nothing to lose.

2. Chép file trên vào file test1. Xóa dòng đầu tiên trong file test2. Chuyển về cuối văn bản, soạn thảo thêm:

Toi da dung lenh dd de xoa mot dong

To delete to the end of a word

1. Press <ESC> to make sure you are in command mode
2. Move the cursor to the line below marked ->
3. Move the cursor to the beginning of a word that need to be deleted
4. Type dw to make the word disappear

Note: The letter dw will appear on the last line of the screen as you type them. If you type some thing wrong, press <ESC> and start over

5. Repeat steps 3 and 4 until the sentence is correct

3. Kéo dòng thứ 3 nối tiếp vào dòng thứ 2. Chuyển về cuối văn bản, soạn thảo thêm dòng sau:

Toi da dung lenh 2yy de copy 2 dong cuoi cung xuong duoi dong dau tien

4. Soạn thảo lại dòng đầu tiên vào bộ nhớ, sau đó dùng lệnh **p** để đặt từ bộ nhớ vào dòng đầu tiên. So sánh với kết quả lệnh **P**. Ghi lại file test1 và thoát ra khỏi vi
5. Tìm các từ I am trong các file btvi1?. Kết quả ghi vào trong file danh sach
6. Đếm các dòng, các từ, ký tự trong các file btvi1?. Kết quả hay lỗi được ghi vào cuối file danh sach, được sắp xếp theo thứ tự từ lớn đến nhỏ.
7. Chia file btvi1 thành 2, 3, 4, 5 file nhỏ.
8. Trình bày nội dung file btvi1 dưới dạng 2, 3, 4 cột.

### ***XVII. 1. d. Bài tập UNIX-4***

1. Tạo một chương trình tên là thoigian có chứa lệnh sau:

```
banner `date | cut -c12-19`
```

2. Trong thư mục làm việc thêm những dòng sau vào cuối file `.profile`:

```
echo Chúc một ngày làm việc tốt lành
echo `date`
```

3. Sửa file `.profile` để chạy môi trường C-Shell, dùng lệnh `set history=64` và lệnh `alias` trong `.cshrc` để đặt lệnh `h` là tên rút gọn của lệnh `history` cho tiện dùng.
4. Sử dụng các cấu trúc `if`, `then`, `else`, `case`, v.v.. để kiểm tra sự tồn tại của file do người dùng lựa chọn.
5. Tương tự, kiểm tra xem file có phải là thư mục.
6. Viết shell tên là `gui` để gửi thông báo cho ba người dùng và thực hiện chạy shell đó.
7. Tương tự, gửi E-mail cho hai người dùng khác.
8. Thêm đường dẫn cho thư mục làm việc của mình (trong `.profile`).
9. Tìm file có kích thước lớn nhất trong thư mục `/var/opt/oracle`.
10. Đếm số từ trong file `/etc/uucp/Dialers`, so sánh với file `/usr/sbin/ping`.
11. Đếm xem thư mục `/etc` có bao nhiêu file.
12. Hiện thị lại tuần tự 12 lệnh trước, 3 lệnh đầu.

## XVII. 2. Bài tập nâng cao

1. Viết shell để sao chép một cây thư mục sang một vị trí khác.
2. Tìm tất cả các tập tin bị thay đổi trong 24 giờ và truyền chúng qua hệ thống khác hay lưu trữ lên vùng đĩa khác.
3. Viết shell để thực hiện một lệnh nào đó (như `fsck` chẳng hạn) tại một thời điểm nhất định.
4. Tạo và phân quyền cho hai người dùng với tên `sol1`, `sol2`. Thay đổi quyền của `sol1` và xóa `sol2`.
5. Viết chương trình `my-dir` để xem được thư mục hiện thời dưới dạng:

---

bin	<DIR>
dev	<DIR>
dead_letter	1120

6. Sửa lại chương trình để có thể đọc được thư mục bất kỳ.
7. Viết chương trình `my_del` để xóa các file liên kết với một file vật lý.
8. Viết chương trình kiểm tra xem người dùng đã được đăng ký và có đăng nhập hệ thống hay không. Nếu có, gửi họ lời chúc "Xin chúc bạn luôn vui vẻ!".
9. Viết chương trình in lại dạng kết quả ra của lệnh `cal` theo tuần, theo tháng và theo năm.
10. Thiết lập hệ thống **uucp** cho Ngân hàng Công thương Việt Nam từ phía chi nhánh và từ phía trung ương với giả định có một máy chủ tại trung ương và ba máy chủ tại các chi nhánh.
11. Viết chương trình truyền file theo cấu trúc cây thư mục.
12. Soạn một văn bản trên UNIX rồi chuyển đổi sang dạng DOS và ngược lại.
13. Thiết lập việc tạo thông báo giữa các máy PC chạy trên DOS trong mạng.
14. Sử dụng **telnet** và chương trình tìm thư và gửi cho các máy trong mạng.
15. Sử dụng **ftp** và **ftpd** để lấy dữ liệu giữa hai máy PC.
16. Sử dụng **ftp** trong Windows để lấy dữ liệu trên máy UNIX.
17. Tạo hai thư mục `/export/cs1` và `/export/cs2`. Làm cho các máy PC được quyền đọc ghi trên `/export/cs1`, nhưng chỉ được đọc `/export/cs2`, trừ máy `CLIENT2`.
18. Dùng lệnh **mt** để tạo ổ CD chung cho các máy PC.
19. Thiết lập kết nối **uucp** giữa một chi nhánh ngân hàng với trung tâm qua hai modem. Số điện thoại của trung tâm là 9 111 222, của chi nhánh là (097) 5 432 687 và (097) 5 432 639.

- 
20. Tổ chức một DNS với máy chủ gốc chính (root master) là `natbank.com`, máy chủ replica là `bank_replica.com`, các vùng con là `hanoi`, `haiphong`, `danang` và `hcm`.
  21. Viết shell **fdcopy** để thực hiện việc chép hai đĩa mềm.
  22. Mở thêm cổng nối tiếp cho kết nối **uucp** và cấu hình nó.
  23. Sao chép hai đĩa cứng.
  24. Viết một shell script bằng C-shell tên là **memlook** để hiển thị một cách thống kê bộ nhớ dùng cho một tiến trình có định danh nào đó sau mỗi khoảng thời gian xác định.
  25. Viết shell quản lý (sao lưu, nén, xóa, v.vv) các file log theo hàng tuần, hàng ngày.



## XVIII. Tài liệu tham khảo

1. *Solaris Enhanced User Skills*. Student Guide. SunSoft. February 1995. Part No. : TUSK-000-SUP-A.
2. *Solaris System Administration I*. Student Guide. SunSoft. February 1995. Part No. : SASK-100-SUP-A.
3. *Solaris System Administration I & II*. Student Guide. SunSoft. February 1995. Part No. : SASK-200-SUP-A.
4. *OpenWindows DekSet Productivity Tools*. Student Guide. SunSoft. February 1995. Pat No. : OWSK-000-SUP-A.
5. *XENIX System V*. Programmer's Reference. Microsoft and Santa Cruz Corp. Portions@ 1988
6. *Inside UNIX*. C. Hare, E. Dulaney, G.Eckel, S. Lee, L. Ray, W. Steen. 2<sup>nd</sup> Edition, Vol. 1, 2 and 3. New Riders Publishing. Indianapolis, USA. 1996. 1212 pp.
7. *System & Network Administration*. Sun Microsystems. 1990. 878 pp.
8. *Practical UNIX Programming*. K. A. Robbins & S. Robbins. Prentice-Hall, Inc. New Jersey. 1996. 658 pp.
9. *SunOS Reference Manual*. SunSoft. 1990.
10. *OpenWindows Reference Manual*. SunSoft. 1990.
11. *Learning the UNIX Operating System*. G. Todino, J. Strang, J. Peek. O'Reilly & Associates, Inc. 92 pp.
12. *UNIX Programmer's Reference*. J. Valley. QUE Corp. 1994.
13. *UNIX Network Programming*. R. Stevens. Prentice-Hall. New Jersey. 1990. 772 pp.
14. *Méthodologie des systèm d'exploitation (Unix)*. O. Camp, M. Hennis. Institute Francophone d'Informatique. 1998.
15. *Solstice. FireWall – 1 Administrator's Guide*. SunSoft. 1996. 287 pp.
16. *UNIX® System Administration*. F. G. Fiamingo. The Ohio State University. 1996. 300 pp. (Có thể tìm thấy tại [http://www-wks.acs.ohio-state.edu/sysadm\\_course/sysadm\\_book.ps](http://www-wks.acs.ohio-state.edu/sysadm_course/sysadm_book.ps).)

17. *SuSE Linux 6.4*. B. Bauer et al. SuSE GmbH. 2000. 586 pp.
18. *Essential System Administration*. Eelen Frisch. O'Reilly & Associates, Inc. 1996. 759 pp.
19. *Microsoft Windows NT Network Administration*. Microsoft Press. 1997. 519 pp.
20. *The Professional Reference NetWare*. K. Siyan. New Riders Publishing. 19. 1634 pp.
21. *Independence through knowledge. Solaris 2.x Network Administration*. Student Guide. SunService. Part No.: SA-360. 1993.
22. *TCP/IP & Related Protocols*. 2<sup>nd</sup> Edition. U. Black. McGraw-Hill, Inc. 1995. 376 pp.
23. *LaTeX for Everyone*. J. Hahn. Prentice-Hall, Inc. 1993. 346 pp.
24. *Using Samba*. R. Eckstein, D. Collier-Brown, P. Kelly. O'Reilly Publishing, 1<sup>st</sup> Edition, Nov. 1999. 416 pp.

Một số địa chỉ mà bạn có thể tải phần mềm về và học hỏi trên Internet:

- <http://www.kde.org/mirrors.html>
- <ftp://ftp.kde.org/pub/kde/stable/latest/distribution/pkg/Solaris-2.6/Sparc/>
- <ftp://ftp.de.kde.org/pub/kde/stable/latest/distribution/tar/generic/source/>
- <http://www.troil.no/>
- <http://egcs.cygnum.com/ml/gcc-bugs/1999-04/msg00618.html>
- [www.gnome.org](http://www.gnome.org)
- <ftp://ftp.uni-stuttgart.de/pub/gnu/>
- [www.kde.org/cvsup.html](http://www.kde.org/cvsup.html)
- <http://lists.kde.org> và <http://bugs.kde.org>
- <http://www.stardivision.com>
- <http://solder.sun.com>
- [www.stardivision.com](http://www.stardivision.com)
- <http://sunsolve.sun.com>
- <http://www.brunel.ac.uk/depts/cc/network/Linux/slip.html>
- [www.apache.org/dist/](http://www.apache.org/dist/)
- <http://www.samba.org>
- ...

## UNIX – HỆ ĐIỀU HÀNH VÀ MỘT SỐ VẤN ĐỀ QUẢN TRỊ MẠNG

*Chịu trách nhiệm xuất bản* : PGS. TS. Tô Đăng Hải  
*Biên tập* : Minh Luận, Ngọc Khuê  
*Trình bày* : Hương Lan

NHÀ XUẤT BẢN KHOA HỌC VÀ KỸ THUẬT  
70 TRẦN HƯNG ĐẠO – HÀ NỘI

---

In 800 bản, khổ 16 x 24 cm tại Công ty In Hàng không  
Giấy phép xuất bản số 113-232, do Cục Xuất bản cấp ngày 14/5/2003  
In xong và nộp lưu chiểu tháng 6 năm 2003

203090



**Giá: 87.000đ**