

TÀI LIỆU ĐÀO TẠO

QUẢN TRỊ HỆ THỐNG LINUX



**Tài liệu này được biên soạn theo tài liệu giảng dạy của Viện Linux
(LPI)**

HÀ NỘI 2006

MỤC LỤC

GIỚI THIỆU GIẤY PHÉP CÔNG CỘNG GNU	8
GIỚI THIỆU	17
Giới thiệu tài liệu	17
CÀI ĐẶT	18
Cấu trúc của đĩa cài	18
Cài đặt Cục bộ	19
Cài đặt qua Mạng	20
Phục hồi Hệ thống	20
Chiến lược Phân vùng	21
Khởi động kép với nhiều hệ điều hành	22
Bài tập	22
CẤU HÌNH PHẦN CỨNG	23
Bộ nhớ	23
Quản lý Tài nguyên	23
USB	25
SCSI	25
Network Card	26
Modem	27
Máy in	28
Bài tập	28
QUẢN LÝ THIẾT BỊ	29
Đĩa và Phân vùng	29
Công cụ Phân vùng đĩa	30
Bootloader	31

Quản trị Hệ thống Linux - Cơ bản

Những thiết bị đã quản lý.....	33
Quotas	34
Bài tập	35
HỆ THỐNG FILE TRONG LINUX.....	36
Cấu trúc của hệ thống file	36
Hệ thống file chuẩn ext2	38
Kiểm soát dung lượng đĩa.....	40
Quyền truy xuất File, Thư mục.....	41
Bài tập	44
CHẾ ĐỘ DÒNG LỆNH	46
Tương tác với SHELL	46
Biến môi trường của Shell	48
Chuyển hướng kết xuất.....	50
Dấu ngoặc và Các ký tự Đa nghĩa (Metacharacter).....	53
Lịch sử dòng lệnh.....	55
Bài tập	56
QUẢN LÝ FILE	59
Di chuyển quanh hệ thống file	59
Tìm kiếm file và thư mục.....	59
Làm việc với thư mục	62
Sử dụng cp và mv	62
Hard links và symbol links	64
Touching và dd-ing	65
Bài tập	66
QUẢN LÝ TIẾN TRÌNH.....	68
Xem các tiến trình đang chạy	68

Quản trị Hệ thống Linux - Cơ bản

Thay đổi tiến trình.....	70
Tiến trình và Shell.....	72
Bài tập	74
XỬ LÝ VĂN BẢN	76
cat the Swiss Army Knife	76
Các công cụ đơn giản.....	77
Xử lý văn bản.....	79
Bài tập	81
CÀI ĐẶT PHẦN MỀM.....	84
Giới thiệu	84
Thư viện tĩnh và thư viện chia sẻ	85
Cài đặt nguồn	88
Quản lý gói Redhat (Redhat Package Manager RPM)	89
Công cụ Alien	93
Bài tập	94
THAO TÁC VỚI VĂN BẢN NÂNG CAO	95
Các biểu thức chính qui	95
Họ grep.....	96
Làm việc với grep	96
egrep và fgrep	97
Bộ soạn thảo Stream – sed.....	97
Bài tập	99
SỬ DỤNG TRÌNH SOẠN THẢO VI	101
Các chế độ Vi.....	101
Các mục văn bản.....	101
Chèn văn bản.....	102

Quản trị Hệ thống Linux - Cơ bản

Xoá văn bản	103
Copy / Paste	103
Tìm kiếm.....	104
Làm lại (Undo).....	105
Ghi văn bản	105
Bài tập	106
 NHÂN LINUX.....	107
Khái niệm nhân	107
Nhân Modular	108
Biên dịch lại nhân	109
Thực hành	116
 KHỞI ĐỘNG LINUX	117
Tổng quan	117
Tìm hiểu các mức thực thi (Runlevels).....	117
inittab	119
GRUB - GRand Unified Bootloader.....	121
Từ khởi động đến bash.....	123
Thực hành	124
 QUẢN LÝ NGƯỜI DÙNG VÀ NHÓM	125
Tạo người dùng mới.....	125
Làm việc với nhóm	126
File cấu hình.....	128
Các tham số lựa chọn của câu lệnh.....	131
Sửa thiết lập mặc định và tài khoản	131
Thực hành	134
 CẤU HÌNH MẠNG	136

Quản trị Hệ thống Linux - Cơ bản

The Network Interface	136
Thông tin máy chủ (Host Information).....	137
Khởi động (Start) và dừng (Stop) mạng	138
Định tuyến.....	140
Các công cụ mạng.....	143
Thực hành	147
MẠNG TCP/IP	149
Số nhị phân và Dotted Quad	149
Địa chỉ Broadcast, địa chỉ mạng và netmask	149
Lớp mạng	152
Subnets	153
Họ giao thức TCP/IP	155
Các dịch vụ và các cổng trong TCP/IP	157
Thực hành	159
CÁC DỊCH VỤ MẠNG.....	160
Tiến trình nền inetd (cũ)	160
Tiến trình nền xinetd.....	161
TCP wrappers.....	162
Thiết lập NFS	163
SMB và NMB	164
Các dịch vụ DNS	166
Máy chủ Apaches.....	172
Thực hành	174
BASH SCRIPTING	177
Môi trường bash.....	177
Các yếu tố Scripting.....	179
Tính toán logic	181

Quản trị Hệ thống Linux - Cơ bản

Vòng lặp.....	182
Nhập dữ liệu từ dòng lệnh	184
Làm việc với số.....	185
Thực hành	185
BẢO MẬT	187
Bảo mật địa phương.....	187
An ninh mạng.....	190
Shell an toàn.....	194
Cấu hình thời gian.....	196
Bảo mật nhân	198
QUẢN TRỊ HỆ THỐNG LINUX.....	201
Tổng quan	201
Logfiles và các file cấu hình	201
Các tiện ích nhật ký.....	203
Tự động hóa công việc (Automatic Tasks).....	205
Sao lưu và nén.....	207
Tài liệu	209
Thực hành	212
IN ẤN	214
Bộ lọc (Filters) và gs.....	214
Máy in và hàng đợi in	214
Các công cụ in ấn.....	215
Các file cấu hình	217
Thực hành	220

GIỚI THIỆU GIẤY PHÉP CÔNG CỘNG GNU

BẢN DỊCH TIẾNG VIỆT CỦA GIẤY PHÉP CÔNG CỘNG GNU

Đây là bản dịch tiếng Việt không chính thức của Giấy phép Công cộng GNU. Bản dịch này không phải do Tổ chức Phần mềm Tự do ấn hành, và nó không quy định về mặt pháp lý các điều khoản cho các phần mềm sử dụng giấy phép GNU GPL -- chỉ có bản tiếng Anh gốc của GNU GPL mới có tính pháp lý. Tuy nhiên, chúng tôi hy vọng rằng bản dịch này sẽ giúp cho những người nói tiếng Việt hiểu rõ hơn về GNU GPL.

GIẤY PHÉP CÔNG CỘNG GNU (GPL)

Giấy phép công cộng GNU

Phiên bản 2, tháng 6/1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.

59 Temple Place - Suite 330, Boston, MA 02111-1307, USA

Mọi người đều được phép sao chép và lưu hành bản sao nguyên bản nhưng không được phép thay đổi nội dung của giấy phép này.

Lời nói đầu

Giấy phép sử dụng của hầu hết các phần mềm đều được đưa ra nhằm hạn chế bạn tự do chia sẻ và thay đổi nó. Ngược lại, Giấy phép Công cộng của GNU có mục đích đảm bảo cho bạn có thể tự do chia sẻ và thay đổi phần mềm tự do - tức là đảm bảo rằng phần mềm đó là tự do đối với mọi người sử dụng. Giấy phép Công cộng này áp dụng cho hầu hết các phần mềm của Tổ chức Phần mềm Tự do và cho tất cả các chương trình khác mà tác giả cho phép sử dụng. (Đối với một số phần mềm khác của Tổ chức Phần Mềm Tự do, áp dụng Giấy phép Công cộng Hạn chế của GNU thay cho giấy phép công cộng). Bạn cũng có thể áp dụng nó cho các chương trình của mình.

Khi nói đến phần mềm tự do, chúng ta nói đến sự tự do sử dụng chứ không quan tâm về giá cả. Giấy phép Công cộng của chúng tôi được thiết kế để đảm bảo rằng bạn hoàn toàn tự do cung cấp các bản sao của phần mềm tự do (cũng như kinh doanh dịch vụ này nếu bạn muốn), rằng bạn có thể nhận được mã nguồn nếu bạn có yêu cầu, rằng bạn có thể thay đổi phần mềm hoặc sử dụng các thành phần của

phần mềm đó cho những chương trình tự do mới; và rằng bạn biết chắc là bạn có thể làm được những điều này.

Để bảo vệ bản quyền của bạn, chúng tôi cần đưa ra những hạn chế để ngăn chặn những ai chối bỏ quyền của bạn, hoặc yêu cầu bạn chối bỏ quyền của mình. Những hạn chế này cũng có nghĩa là những trách nhiệm nhất định của bạn khi cung cấp các bản sao phần mềm hoặc khi chỉnh sửa phần mềm đó.

Ví dụ, nếu bạn cung cấp các bản sao của một chương trình, dù miễn phí hay không, bạn phải cho người nhận tất cả các quyền mà bạn có. Bạn cũng phải đảm bảo rằng họ cũng nhận được hoặc tiếp cận được mã nguồn. Và bạn phải thông báo những điều khoản này cho họ để họ biết rõ về quyền của mình.

Chúng tôi bảo vệ quyền của bạn với hai bước: (1) bảo vệ bản quyền phần mềm, và (2) cung cấp giấy phép này để bạn có thể sao chép, lưu hành và/hoặc chỉnh sửa phần mềm một cách hợp pháp.

Ngoài ra, để bảo vệ các tác giả cũng như để bảo vệ chính mình, chúng tôi muốn chắc chắn rằng tất cả mọi người đều hiểu rõ rằng không hề có bảo hành đối với phần mềm tự do này. Nếu phần mềm được chỉnh sửa thay đổi bởi một người khác và sau đó lưu hành, thì chúng tôi muốn những người sử dụng biết rằng phiên bản họ đang có không phải là bản gốc, do đó tất cả những trục trặc do những người khác gây ra hoàn toàn không ảnh hưởng tới uy tín của tác giả ban đầu.

Cuối cùng, bất kỳ một chương trình tự do nào cũng đều thường xuyên có nguy cơ bị đe dọa về giấy phép bản quyền. Chúng tôi muốn tránh nguy cơ khi những người cung cấp lại một chương trình tự do có thể có được giấy phép bản quyền cho bản thân họ, từ đó trở thành độc quyền đối với chương trình đó. Để ngăn ngừa trường hợp này, chúng tôi đã nêu rõ rằng mỗi giấy phép bản quyền hoặc phải được cấp cho tất cả mọi người sử dụng một cách tự do hoặc hoàn toàn không cấp phép.

Dưới đây là những điều khoản và điều kiện rõ ràng đối với việc sao chép, lưu hành và chỉnh sửa.

Những điều khoản và điều kiện đối với việc sao chép, lưu hành và chỉnh sửa

0. Giấy phép này áp dụng cho bất kỳ một chương trình hay sản phẩm nào mà người giữ bản quyền công bố rằng nó có thể được cung cấp trong khuôn khổ những điều khoản của Giấy phép Công cộng này. Từ “Chương trình” dưới đây có

nghĩa là tất cả các chương trình hay sản phẩm như vậy, và “sản phẩm dựa trên Chương trình” có nghĩa là Chương trình hoặc bất kỳ một sản phẩm nào bắt nguồn từ chương trình đó tuân theo luật bản quyền, nghĩa là một sản phẩm dựa trên Chương trình hoặc một phần của nó, đúng nguyên bản hoặc có một số chỉnh sửa và/hoặc được dịch ra một ngôn ngữ khác. (Dưới đây, việc dịch cũng được hiểu trong khái niệm “chỉnh sửa”). Mỗi người được cấp phép được gọi là “bạn”.

Trong Giấy phép này không đề cập tới các hoạt động khác ngoài việc sao chép, lưu hành và chỉnh sửa; chúng nằm ngoài phạm vi của giấy phép này. Hành động chạy chương trình không bị hạn chế, và những kết quả từ việc chạy chương trình chỉ được đề cập tới nếu nội dung của nó tạo thành một sản phẩm dựa trên chương trình (độc lập với việc chạy chương trình). Điều này đúng hay không là phụ thuộc vào Chương trình.

1. Bạn có thể sao chép và lưu hành những phiên bản nguyên bản của mã nguồn Chương trình đúng như khi bạn nhận được, qua bất kỳ phương tiện phân phối nào, với điều kiện trên mỗi bản sao bạn đều kèm theo một ghi chú bản quyền rõ ràng và từ chối bảo hành; giữ nguyên tất cả các ghi chú về Giấy phép và về việc không có bất kỳ một sự bảo hành nào; và cùng với Chương trình bạn cung cấp cho người sử dụng một bản sao của Giấy phép này.

Bạn có thể tính phí cho việc chuyển giao bản sao, và tùy theo quyết định của mình bạn có thể cung cấp bảo hành để đổi lại với chi phí mà bạn đã tính.

2. Bạn có thể chỉnh sửa bản sao của bạn hoặc các bản sao của Chương trình hoặc của bất kỳ phần nào của nó, từ đó hình thành một sản phẩm dựa trên Chương trình, và sao chép cũng như lưu hành sản phẩm đó hoặc những chỉnh sửa đó theo điều khoản trong Mục 1 ở trên, với điều kiện bạn đáp ứng được những điều kiện dưới đây:

a) Bạn phải có ghi chú rõ ràng trong những tập tin đã chỉnh sửa là bạn đã chỉnh sửa nó, và ngày tháng của bất kỳ một thay đổi nào.

b) Bạn phải cấp phép miễn phí cho tất cả các bên thứ ba đối với các sản phẩm bạn cung cấp hoặc phát hành, bao gồm Chương trình nguyên bản, từng phần của nó hay các sản phẩm dựa trên Chương trình hay dựa trên từng phần của Chương trình, theo những điều khoản của Giấy phép này.

c) Nếu chương trình đã chỉnh sửa thường đọc lệnh tương tác trong khi chạy, bạn phải thực hiện sao cho khi bắt đầu chạy để sử dụng tương tác theo cách thông thường nhất phải có một thông báo bao gồm bản quyền và

thông báo về việc không có bảo hành (hoặc thông báo bạn là người cung cấp bảo hành), và rằng người sử dụng có thể cung cấp lại Chương trình theo những điều kiện này, và thông báo để người sử dụng có thể xem bản sao của Giấy phép này. (Ngoại lệ: nếu bản thân Chương trình là tương tác nhưng không có một thông báo nào như trên, thì sản phẩm của bạn dựa trên Chương trình đó cũng không bắt buộc phải có thông báo như vậy).

Những yêu cầu trên áp dụng cho toàn bộ các sản phẩm chỉnh sửa. Nếu có những phần của sản phẩm rõ ràng không bắt nguồn từ Chương trình, và có thể được xem là độc lập và riêng biệt, thì Giấy phép này và các điều khoản của nó sẽ không áp dụng cho những phần đó khi bạn cung cấp chúng như những sản phẩm riêng biệt. Nhưng khi bạn cung cấp những phần đó như những phần nhỏ trong cả một sản phẩm dựa trên Chương trình, thì việc cung cấp này phải tuân theo những điều khoản của Giấy phép này, cho phép những người được cấp phép có quyền đối với toàn bộ sản phẩm, cũng như đối với từng phần trong đó, bất kể ai đã viết nó.

Như vậy, điều khoản này không nhằm mục đích xác nhận quyền hoặc tranh giành quyền của bạn đối với những sản phẩm hoàn toàn do bạn viết; mà mục đích của nó là nhằm thi hành quyền kiểm soát đối với việc cung cấp những sản phẩm bắt nguồn hoặc tổng hợp dựa trên Chương trình.

Ngoài ra, việc kết hợp thuần túy Chương trình (hoặc một sản phẩm dựa trên Chương trình) với một sản phẩm không dựa trên Chương trình với mục đích lưu trữ hoặc quảng bá không đưa sản phẩm đó vào trong phạm vi áp dụng của Giấy phép này.

3. Bạn có thể sao chép và cung cấp Chương trình (hoặc một sản phẩm dựa trên Chương trình, nêu trong Mục 2) dưới hình thức mã đã biên dịch hoặc dạng có thể thực thi được trong khuôn khổ các điều khoản nêu trong Mục 1 và 2 ở trên, nếu như bạn:

- a) Kèm theo đó một bản mã nguồn dạng đầy đủ có thể biên dịch được theo các điều khoản trong Mục 1 và 2 nêu trên trong một môi trường trao đổi phần mềm thông thường; hoặc,
- b) Kèm theo đó một đề nghị có hạn trong ít nhất 3 năm, theo đó cung cấp cho bất kỳ một bên thứ ba nào một bản sao đầy đủ của mã nguồn tương ứng, và phải được cung cấp với giá chi phí không cao hơn giá chi phí vật

lý của việc cung cấp theo các điều khoản trong Mục 1 và 2 nêu trên trong một môi trường trao đổi phần mềm thông thường; hoặc

c) Kèm theo đó thông tin bạn đã nhận được đề nghị cung cấp mã nguồn tương ứng. (Phương án này chỉ được phép đối với việc cung cấp phi thương mại và chỉ với điều kiện nếu bạn nhận được Chương trình dưới hình thức mã đã biên dịch hoặc dạng có thể thực thi được cùng với lời đề nghị như vậy, theo phần b trong điều khoản nêu trên).

Mã nguồn của một sản phẩm là một dạng ưu tiên của sản phẩm dành cho việc chỉnh sửa nó. Với một sản phẩm có thể thi hành, mã nguồn hoàn chỉnh có nghĩa là tất cả các mã nguồn cho các môđun trong sản phẩm đó, cộng với tất cả các tệp tin định nghĩa giao diện đi kèm với nó, cộng với các hướng dẫn dùng để kiểm soát việc biên dịch và cài đặt các tệp thi hành. Tuy nhiên, một ngoại lệ đặc biệt là mã nguồn không cần chứa bất kỳ một thứ gì mà bình thường được cung cấp (từ nguồn khác hoặc hình thức nhị phân) cùng với những thành phần chính (chương trình biên dịch, nhân, và những phần tương tự) của hệ điều hành mà các chương trình chạy trong đó, trừ khi bản thân thành phần đó lại đi kèm với một tệp thi hành.

Nếu việc cung cấp lưu hành mã đã biên dịch hoặc tập tin thi hành được thực hiện qua việc cho phép tiếp cận và sao chép từ một địa điểm được chỉ định, thì việc cho phép tiếp cận tương đương tới việc sao chép mã nguồn từ cùng địa điểm cũng được tính như việc cung cấp mã nguồn, mặc dù thậm chí các bên thứ ba không bị buộc phải sao chép mã nguồn cùng với mã đã biên dịch.

4. Bạn không được phép sao chép, chỉnh sửa, cấp phép hoặc cung cấp Chương trình trừ phi phải tuân thủ một cách chính xác các điều khoản trong Giấy phép. Bất kỳ ý định sao chép, chỉnh sửa, cấp phép hoặc cung cấp Chương trình theo cách khác đều làm mất hiệu lực và tự động huỷ bỏ quyền của bạn trong khuôn khổ Giấy phép này. Tuy nhiên, các bên đã nhận được bản sao hoặc quyền từ bạn với Giấy phép này sẽ không bị huỷ bỏ giấy phép nếu các bên đó vẫn tuân thủ đầy đủ các điều khoản của giấy phép.

5. Bạn không bắt buộc phải chấp nhận Giấy phép này khi bạn chưa ký vào đó. Tuy nhiên, không có gì khác đảm bảo cho bạn được phép chỉnh sửa hoặc cung cấp Chương trình hoặc các sản phẩm bắt nguồn từ Chương trình. Những hành động này bị luật pháp nghiêm cấm nếu bạn không chấp nhận Giấy phép này. Do vậy, bằng việc chỉnh sửa hoặc cung cấp Chương trình (hoặc bất kỳ một sản phẩm

nào dựa trên Chương trình), bạn đã thể hiện sự chấp thuận đối với Giấy phép này, cùng với tất cả các điều khoản và điều kiện đối với việc sao chép, cung cấp hoặc chỉnh sửa Chương trình hoặc các sản phẩm dựa trên nó.

6. Mỗi khi bạn cung cấp lại Chương trình (hoặc bất kỳ một sản phẩm nào dựa trên Chương trình), người nhận sẽ tự động nhận được giấy phép từ người cấp phép đầu tiên cho phép sao chép, cung cấp và chỉnh sửa Chương trình theo các điều khoản và điều kiện này. Bạn không thể áp đặt bất cứ hạn chế nào khác đối với việc thực hiện quyền của người nhận đã được cấp phép từ thời điểm đó. Bạn cũng không phải chịu trách nhiệm bắt buộc các bên thứ ba tuân thủ theo Giấy phép này.

7. Nếu như, theo quyết định của toà án hoặc với những bằng chứng về việc vi phạm bản quyền hoặc vì bất kỳ lý do nào khác (không giới hạn trong các vấn đề về bản quyền), mà bạn phải tuân theo các điều kiện (nêu ra trong lệnh của toà án, biên bản thoả thuận hoặc ở nơi khác) trái với các điều kiện của Giấy phép này, thì chúng cũng không thể miễn cho bạn khỏi những điều kiện của Giấy phép này. Nếu bạn không thể đồng thời thực hiện các nghĩa vụ của mình trong khuôn khổ Giấy phép này và các nghĩa vụ thích đáng khác, thì hậu quả là bạn hoàn toàn không được cung cấp Chương trình. Ví dụ, nếu trong giấy phép bản quyền không cho phép những người nhận được bản sao trực tiếp hoặc gián tiếp qua bạn có thể cung cấp lại Chương trình thì trong trường hợp này cách duy nhất bạn có thể thoả mãn cả hai điều kiện là hoàn toàn không cung cấp Chương trình.

Nếu bất kỳ một phần nào trong điều khoản này không có hiệu lực hoặc không thể thi hành trong một hoàn cảnh cụ thể, thì sẽ cân đối áp dụng các điều khoản, và toàn bộ điều khoản sẽ được áp dụng trong những hoàn cảnh khác.

Mục đích của điều khoản này không nhằm buộc bạn phải vi phạm bất kỳ một bản quyền nào hoặc các quyền sở hữu khác hoặc tranh luận về giá trị hiệu lực của bất kỳ quyền hạn nào như vậy; mục đích duy nhất của điều khoản này là nhằm bảo vệ sự toàn vẹn của hệ thống cung cấp phần mềm tự do đang được thực hiện với giấy phép công cộng. Nhiều người đã đóng góp rất nhiều vào sự đa dạng của các phần mềm tự do được cung cấp thông qua hệ thống này với sự tin tưởng rằng hệ thống được sử dụng một cách thống nhất; tác giả/người cung cấp có quyền quyết định rằng họ có mong muốn cung cấp phần mềm thông qua hệ thống nào khác hay không, và người được cấp phép không thể có ảnh hưởng tới sự lựa chọn này.

Điều khoản này nhằm làm rõ những hệ quả của các phần còn lại của Giấy phép này.

8. Nếu việc cung cấp và/hoặc sử dụng Chương trình bị cấm ở một số nước nhất định bởi quy định về bản quyền, người giữ bản quyền gốc đã đưa Chương trình vào dưới Giấy phép này có thể bổ sung một điều khoản hạn chế việc cung cấp ở những nước đó, nghĩa là việc cung cấp chỉ được phép ở các nước không bị liệt kê trong danh sách hạn chế. Trong trường hợp này, Giấy phép đưa vào những hạn chế được ghi trong nội dung của nó.

9. Tổ chức Phần mềm Tự do có thể theo thời gian công bố những phiên bản chỉnh sửa và/hoặc phiên bản mới của Giấy phép Công cộng. Những phiên bản đó sẽ đồng nhất với tinh thần của phiên bản hiện này, nhưng có thể khác ở một số chi tiết nhằm giải quyết những vấn đề hay những lo ngại mới.

Mỗi phiên bản sẽ có một mã số phiên bản riêng. Nếu Chương trình và "bất kỳ một phiên bản nào sau đó" có áp dụng một phiên bản Giấy phép cụ thể, bạn có quyền lựa chọn tuân theo những điều khoản và điều kiện của phiên bản giấy phép đó hoặc của bất kỳ một phiên bản nào sau đó do Tổ chức Phần mềm Tự do công bố. Nếu Chương trình không nêu cụ thể mã số phiên bản giấy phép, bạn có thể lựa chọn bất kỳ một phiên bản nào đã từng được công bố bởi Tổ chức Phần mềm Tự do.

10. Nếu bạn muốn kết hợp các phần của Chương trình vào các chương trình tự do khác mà điều kiện cung cấp khác với chương trình này, hãy viết cho tác giả để được phép. Đối với các phần mềm được cấp bản quyền bởi Tổ chức Phần mềm Tự do, hãy đề xuất với tổ chức này; đôi khi chúng tôi cũng có những ngoại lệ. Quyết định của chúng tôi sẽ dựa trên hai mục tiêu là bảo hộ tình trạng tự do của tất cả các sản phẩm bắt nguồn từ phần mềm tự do của chúng tôi, và thúc đẩy việc chia sẻ và tái sử dụng phần mềm nói chung.

KHÔNG BẢO HÀNH

DO CHƯƠNG TRÌNH ĐƯỢC CẤP PHÉP MIỄN PHÍ NÊN KHÔNG CÓ MỘT CHẾ ĐỘ BẢO HÀNH NÀO TRONG MỨC ĐỘ CHO PHÉP CỦA LUẬT PHÁP. TRỪ KHI ĐƯỢC CÔNG BỐ KHÁC ĐI BẰNG VĂN BẢN, NHỮNG NGƯỜI GIỮ BẢN QUYỀN VÀ/HOẶC CÁC BÊN CUNG CẤP CHƯƠNG TRÌNH NGUYÊN BẢN SẼ KHÔNG BẢO HÀNH DƯỚI BẤT KỲ HÌNH THỨC NÀO, BAO GỒM NHƯNG KHÔNG GIỚI HẠN TRONG CÁC HÌNH THỨC BẢO HÀNH ĐỐI VỚI TÍNH THƯƠNG MẠI CŨNG NHƯ TÍNH

THÍCH HỢP CHO MỘT MỤC ĐÍCH CỤ THỂ. BẠN LÀ NGƯỜI CHỊU TOÀN BỘ RỦI RO VỀ CHẤT LƯỢNG CŨNG NHƯ VIỆC VẬN HÀNH CHƯƠNG TRÌNH. TRONG TRƯỜNG HỢP CHƯƠNG TRÌNH CÓ KHIẾM KHUYẾT, BẠN PHẢI CHỊU TOÀN BỘ CHI PHÍ CHO NHỮNG DỊCH VỤ SỬA CHỮA CẦN THIẾT.

TRONG TẤT CẢ CÁC TRƯỜNG HỢP TRỪ KHI CÓ YÊU CẦU CỦA LUẬT PHÁP HOẶC CÓ THỎA THUẬN BẰNG VĂN BẢN, NHỮNG NGƯỜI CÓ BẢN QUYỀN HOẶC BẤT KỲ MỘT BÊN NÀO CHỈNH SỬA VÀ/HOẶC CUNG CẤP LẠI CHƯƠNG TRÌNH TRONG CÁC ĐIỀU KIỆN NHƯ ĐÃ NÊU TRÊN ĐỀU KHÔNG CÓ TRÁCH NHIỆM VỚI BẠN VỀ CÁC LỖI HỒNG HÓC, BAO GỒM CÁC LỖI CHUNG HAY ĐẶC BIỆT, NGẪU NHIÊN HAY TẤT YẾU NẢY SINH DO VIỆC SỬ DỤNG HOẶC KHÔNG SỬ DỤNG ĐƯỢC CHƯƠNG TRÌNH (BAO GỒM NHƯNG KHÔNG GIỚI HẠN TRONG VIỆC MẤT DỮ LIỆU, DỮ LIỆU THIỂU CHÍNH XÁC HOẶC CHƯƠNG TRÌNH KHÔNG VẬN HÀNH ĐƯỢC VỚI CÁC CHƯƠNG TRÌNH KHÁC), THẬM CHÍ CẢ KHI NGƯỜI CÓ BẢN QUYỀN VÀ CÁC BÊN KHÁC ĐÃ ĐƯỢC THÔNG BÁO VỀ KHẢ NĂNG XẢY RA NHỮNG THIẾT HẠI ĐÓ.

KẾT THÚC CÁC ĐIỀU KIỆN VÀ ĐIỀU KHOẢN.

Áp dụng những điều khoản trên như thế nào đối với chương trình của bạn

Nếu bạn xây dựng một chương trình mới, và bạn muốn cung cấp một cách tối đa cho công chúng sử dụng, thì biện pháp tốt nhất để đạt được điều này là phát triển chương trình đó thành phần mềm tự do để ai cũng có thể cung cấp lại và thay đổi theo những điều khoản như trên.

Để làm được việc này, hãy đính kèm những thông báo như sau cùng với chương trình của mình. An toàn nhất là đính kèm chúng trong phần đầu của tập tin mã nguồn để thông báo một cách hiệu quả nhất về việc không có bảo hành; và mỗi tập tin đều phải có ít nhất một dòng về “bản quyền” và trở đến toàn bộ thông báo.

Một dòng đề tên chương trình và nội dung của nó.

Bản quyền (C) năm, tên tác giả.

Chương trình này là phần mềm tự do, bạn có thể cung cấp lại và/hoặc chỉnh sửa nó theo những điều khoản của Giấy phép Công cộng của GNU do Tổ chức Phần

Quản trị Hệ thống Linux - Cơ bản

mềm Tự do công bố; phiên bản 2 của Giấy phép, hoặc bất kỳ một phiên bản sau đó (tuỳ sự lựa chọn của bạn).

Chương trình này được cung cấp với hy vọng nó sẽ hữu ích, tuy nhiên KHÔNG CÓ BẤT KỲ MỘT BẢO HÀNH NÀO; thậm chí kể cả bảo hành về KHẢ NĂNG THƯỜNG MẠI hoặc TÍNH THÍCH HỢP CHO MỘT MỤC ĐÍCH CỤ THỂ. Xin xem Giấy phép Công cộng của GNU để biết thêm chi tiết.

Bạn phải nhận được một bản sao của Giấy phép Công cộng của GNU kèm theo chương trình này; nếu bạn chưa nhận được, xin gửi thư về Tổ chức Phần mềm Tự do, 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.

Xin hãy bổ sung thông tin về địa chỉ liên lạc của bạn (thư điện tử và bưu điện).

Nếu chương trình chạy tương tác, hãy đưa một thông báo ngắn khi bắt đầu chạy chương trình như sau:

Gnomovision phiên bản 69, Copyright (C) năm, tên tác giả.

Gnomovision HOÀN TOÀN KHÔNG CÓ BẢO HÀNH; để xem chi tiết hãy gõ 'show w'. Đây là một phần mềm miễn phí, bạn có thể cung cấp lại với những điều kiện nhất định, gõ 'show c' để xem chi tiết.

Giả thiết lệnh 'show w' và 'show c' cho xem những phần tương ứng trong Giấy phép Công cộng. Tất nhiên những lệnh mà bạn dùng có thể khác với 'show w' và 'show c'; những lệnh này có thể là nhấn chuột hoặc lệnh trong thanh công cụ - tuỳ theo chương trình của bạn.

Bạn cũng cần phải lấy chữ ký của người phụ trách (nếu bạn là người lập trình) hoặc của trường học (nếu có) xác nhận từ chối bản quyền đối với chương trình. Sau đây là ví dụ:

Yoyodyne, Inc., tại đây từ chối tất cả các quyền lợi bản quyền đối với chương trình 'Gnomovision' viết bởi James Hacker.

Chữ ký của Ty Coon, 1 April 1989

Ty Coon, Phó Tổng Giám đốc.

Giấy phép Công cộng này không cho phép đưa chương trình của bạn vào trong các chương trình độc quyền. Nếu chương trình của bạn là một thư viện thủ tục phụ, bạn có thể thấy nó hữu ích hơn nếu cho thư viện liên kết với các ứng dụng độc quyền. Nếu đây là việc bạn muốn làm, hãy sử dụng Giấy phép Công cộng Hạn chế của GNU thay cho Giấy phép này.

GIỚI THIỆU

Giới thiệu tài liệu

Tài liệu Quản trị hệ thống Linux – Cơ bản là cuốn giáo trình được xây dựng với mục đích chuyển tải các kiến thức hết sức cơ bản nhưng cần thiết đối với các học viên, đặc biệt là đối với những người làm công tác giảng dạy.

Tài liệu này được biên dịch chính dựa trên bộ giáo trình của Học viện Linux LPI (Linux Professional Institute). Đây là bộ giáo trình được biên soạn một cách công phu, tỉ mỉ và khoa học, dùng cho việc đào tạo và ôn luyện các chứng chỉ LPI của Học viện Linux.

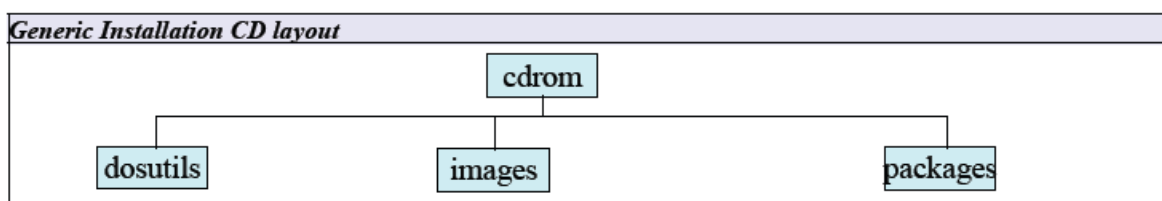
Do đang trong quá trình xây dựng, trong nội dung tài liệu không tránh khỏi nhiều thiếu sót. Rất mong được sự đóng góp ý kiến của người đọc để tài liệu ngày càng được hoàn chỉnh hơn.

Xin chân thành cảm ơn!

CÀI ĐẶT

Cấu trúc của đĩa cài

Hiện tại, có rất nhiều phiên bản phân phối Linux khác nhau. Với mỗi bản, cách đặt tên của các thư mục trên đĩa cài cũng khác nhau. Thông thường chúng có dạng như sau:



packages: Thư mục chứa các gói phần mềm đã được biên dịch trước. Tùy vào từng bản phân phối mà thư mục này có tên khác nhau. Dưới đây là một số ví dụ:

debian: **dist**

mandrake: **Mandrake**

redhat: **RedHat**

suse: **suse**

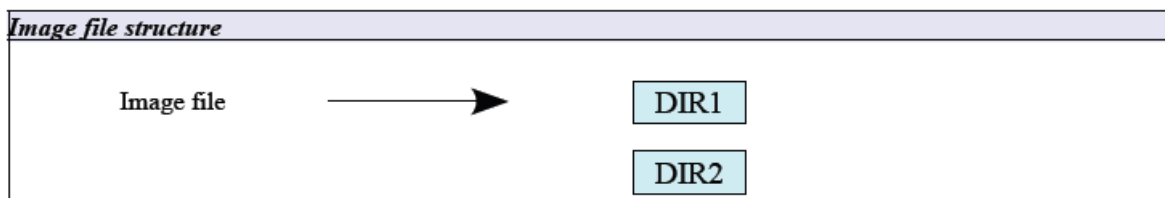
fedora: **Fedora**

images: Dùng để chứa ảnh của Linux. Có nhiều kiểu file ảnh khác nhau. Mỗi file có một công dụng riêng:

- Khởi động tiến trình cài đặt
- Cung cấp module cho nhân
- Khôi phục lại hệ thống

Một số ảnh có thể được ghi lại vào đĩa mềm hoặc CD, USB nhằm mục đích khởi động quá trình cài đặt từ nhiều nguồn khác nhau.

Bản thân nhiều file ảnh cũng chứa bên trong nó những file và thư mục con. Có thể truy cập đến những file và thư mục này thông qua việc ánh xạ file ảnh vào một thiết bị loop.



```
mount -o loop /path/to/Image /mnt
```

dosutils: Thư mục chứa một số công cụ giúp cho việc chuẩn bị cài đặt được thuận lợi hơn trong môi trường DOS.

Cài đặt Cục bộ

Cài đặt cục bộ là cách thức dễ dàng và phổ thông nhất trong tất cả các phương thức cài đặt. Hầu hết các bản phân phối Linux đều có dạng boot CD cho phép khởi động quá trình cài đặt một cách tự động. Với những máy tính không có ổ CD, có thể thay thế nó bởi đĩa mềm hoặc USB để khởi động quá trình này (khi đó, thư mục **packages** thường được đặt trong ổ cứng).

Để tạo ra đĩa mềm hoặc đĩa USB có khả năng khởi động, có thể dùng lệnh **dd** trong Linux hoặc **rawrite.exe** trong DOS/Win.

```
dd if=/path/to/Image of=/dev/fd0 (hoặc /dev/sdX)
rawrite.exe
```

Ví dụ: Đối với các bản phân phối của RedHat các file ảnh này có tên là **boot.img**. Ngoài ra, có thể còn có một số file ảnh đặc biệt khác được cung cấp như: **bootnet.img** hay **pcmcia.img**.

Cài đặt qua Mạng

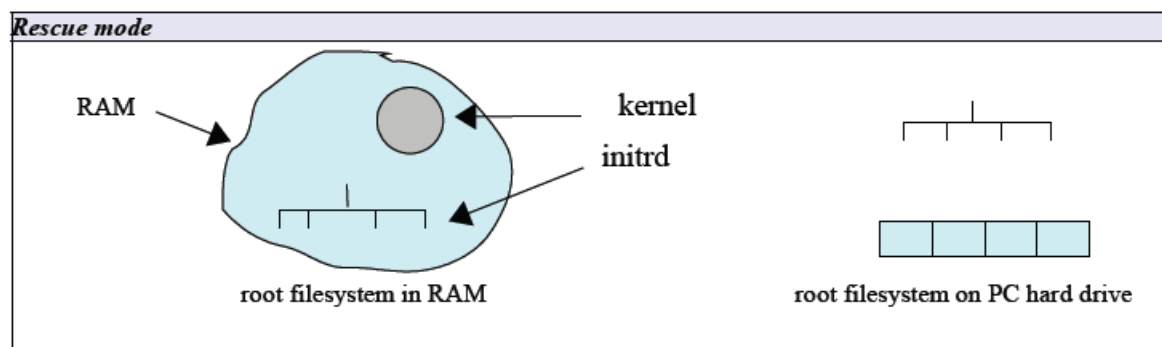
Thông thường các gói cài đặt được để tại một server ở xa, người dùng chỉ cần khởi động quá trình cài đặt, thiết lập các tham số mạng chính xác sau đó, tiến trình cài đặt sẽ tự động download các gói cần thiết về máy tính để cài (thông qua các giao thức như FTP, HTTP, NFS).

Để khởi động quá trình cài đặt có thể sử dụng bất kỳ phương thức nào như đã miêu tả trong phần Cài đặt Cục bộ. Ngoài ra, quá trình này cũng có thể được khởi động thông qua một Card Mạng có khả năng boot kết hợp với DHCP và TFTP Server được thiết lập cho mục đích này.

Phục hồi Hệ thống

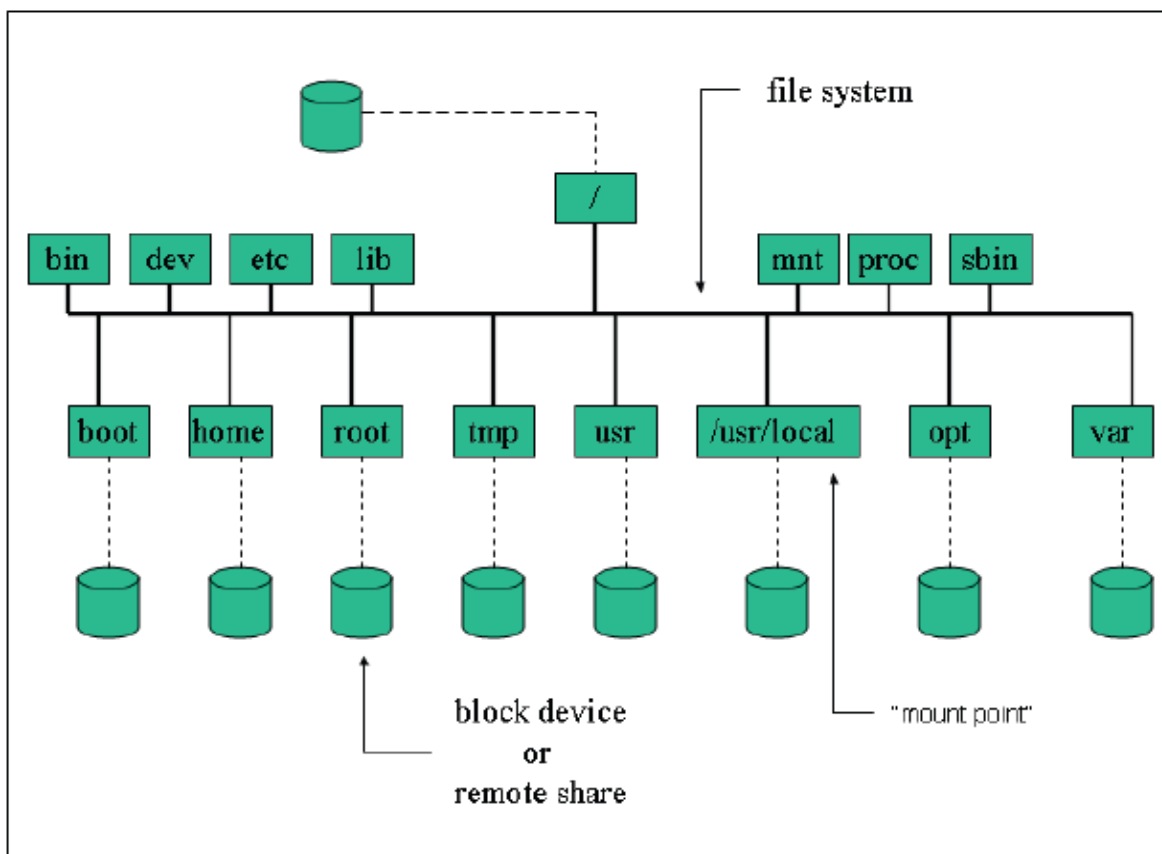
Trong trường hợp hệ thống bị trục trặc, không thể khởi động chính xác, có thể phục hồi được một số lỗi thông qua cơ chế khởi động Phục hồi Hệ thống.

Khi khởi động cơ chế này, một phiên bản thu gọn của Linux và một hệ thống file ảo được nạp vào và chạy ngay trên RAM hệ thống. Hệ thống file thật sẽ được tìm kiếm và ánh xạ vào một thư mục của hệ thống file ảo này. Người dùng có thể dùng lệnh **chroot** để chuyển qua hệ thống file thật và xử lý sự cố. Thông thường nếu tìm thấy, nó sẽ được ánh xạ vào thư mục **/mnt/sysimage** của hệ thống ảo.



Chiến lược Phân vùng

Phân vùng ổ cứng là quá trình không thể thiếu trong khi cài đặt Linux. Tùy theo từng phiên bản phân phối mà quá trình này có thể thực hiện tự động hoặc thủ công. Để thực hiện thủ công, cần có sự hiểu biết sâu sắc về hệ thống file trong Linux cũng như mục đích sử dụng hệ thống.



Hình trên mô tả một lược đồ phân vùng dạng đơn giản cùng hệ thống file của một hệ thống mẫu. Thực chất hệ thống file trong Linux là một cây bao gồm thư mục gốc "/" và các thư mục con nhiều cấp. Các tài nguyên hệ thống được sử dụng để lưu trữ dữ liệu được **gắn kết/ánh xạ (mounted)** vào các điểm chỉ định trên hệ thống file, các điểm này được gọi là các **điểm gắn kết/ánh xạ (mount point)**. Thư mục gốc "/" cũng là một điểm gắn kết và phân vùng lưu trữ dữ liệu cho "/" sẽ được xác định trong quá trình khởi động.

Khởi động kép với nhiều hệ điều hành

Cũng giống như Windows, Linux hỗ trợ nhiều hệ điều hành trên một máy tính. Chương trình khởi động của Linux sẽ tự động nhận biết các hệ điều hành này. Nếu nhận biết thành công, một lựa chọn sẽ được tự động thêm vào menu khởi động.

Linux cũng hỗ trợ khởi động hệ điều hành Windows. Để thiết lập được hệ thống như vậy, cần phải có chiến lược phân vùng đúng đắn. Sau đó, cách đơn giản nhất là cài đặt Linux sau khi cài đặt Windows. Nếu không, phải có kinh nghiệm về ổ cứng, bảng phân vùng... và những phần liên quan để có thể khắc phục sự cố.

Bài tập

1. Cài đặt (qua mạng hoặc không) một hệ thống Linux với yêu cầu như sau:

+ Chọn “Custom System”

+ Phân vùng ổ cứng với Disk Druid thành các phân vùng như sau:

/boot

SWAP

/

/usr

/home

/tmp

/var

+ Cài đặt GRUB lên MBR và đặt mật khẩu cho GRUB.

+ Cài đặt các gói theo yêu cầu của giảng viên.

2. Phục hồi hệ thống

+ Giả sử bạn bị quên mật khẩu root, khởi động lại máy tính và phục hồi lại nó bằng chế độ single.

+ Giả sử bạn cũng quên cả mật khẩu của GRUB nên không khởi động vào chế độ single được. Khởi động máy tính bằng đĩa có khả năng cứu hộ (Rescue Mode). Sửa lại file cấu hình của GRUB (/boot/grub/grub.conf) để xóa mật khẩu.

CẤU HÌNH PHẦN CỨNG

Bộ nhớ

RAM hệ thống được dò tìm bởi BIOS trong quá trình khởi động và kernel sử dụng kết quả của quá trình này. Vì vậy, trong những trường hợp hệ thống sử dụng RAM số lượng ít hơn thực tế cài đặt, cần phải kiểm tra lại phần cứng xem đã cắm đúng qui cách chưa hoặc BIOS có hoạt động đúng không.

Trong trường hợp muốn chỉ định chính xác lượng RAM mà Linux phải dùng, có thể thêm các tham số cho chương trình khởi động được cài đặt trên hệ thống:

LILO

Sửa file /etc/lilo.conf, thêm dòng

append="mem=<Dung lượng RAM>M"

Sau đó chạy /sbin/lilo.

GRUB

Sửa file /boot/grub/grub.conf như sau:

kernel vmlinuz mem=<Dung lượng RAM>M

Quản lý Tài nguyên

Để truy cập vào các thiết bị, hệ thống (CPU) phải cấp phát các tài nguyên truy cập cho chúng. Sau đây là các kiểu tài nguyên này:

IRQs (Interrupt Request Lines)

Là các đường truyền liên lạc trực tiếp từ thiết bị đến CPU giúp các thiết bị yêu cầu CPU xử lý thông tin của chúng gửi đến. Mỗi khi có yêu cầu/ngắt, CPU phải tạm dừng công việc đang thi hành để xử lý ngắt. Có 16 IRQs đánh số từ 0 đến 15.

Địa chỉ I/O (Input/Output – Nhập/Xuất)

Là những địa chỉ trên bộ nhớ hệ thống được ánh xạ vào bộ nhớ của thiết bị. Mọi thao tác trên vùng nhớ này tương đương với thao tác lên bộ nhớ của thiết bị.

DMA (Direct Memory Access channels)

Là các kênh truyền dữ liệu cho phép thiết bị thao tác trực tiếp lên bộ nhớ hệ thống mà không phải thông qua CPU.

Liệt kê các tài nguyên đã cấp phát

Nhân lưu giữ các thông tin này trong thư mục /proc. Các file được sử dụng là:

/proc/dma

/proc/interrupts

/proc/ioports

/proc/pci

Những thông tin này cũng có thể được liệt kê ra bởi các công cụ như *lspci* hay *dmesg*:

lspci

Liệt kê danh sách thông tin của những chipset gắn trên thiết bị tại các khe PCI. Với tham số **-v**, lệnh sẽ liệt kê các thiết lập về I/O và IRQ.

Với tham số **-b** (BUS centric) lệnh sẽ liệt kê thông tin do BIOS quản lý (có thể khác với do nhân quản lý).

dmesg

Hiển thị tất cả các thông điệp mức nhân tính từ lúc khởi động máy. Những thông tin này cũng có thể lấy được từ file */var/log/dmesg*.

Một số Tài nguyên thường dùng

Device	I/O port	IRQ
<i>/dev/ttyS0</i>	<i>0x03f8</i>	<i>4</i>
<i>/dev/ttyS1</i>	<i>0x02f8</i>	<i>3</i>
<i>/dev/lp0</i>	<i>0x378</i>	<i>7</i>

<i>/dev/lp1</i>	<i>0x278</i>	<i>5</i>
<i>soundcard</i>	<i>0x220</i>	

USB

USB (Universal Serial Bus) là chuẩn kết nối giữa các thiết bị với nhau và với PC. Chúng được chia thành các lớp thiết bị như sau:

Display Devices

Communication Devices

Audio Devices

Mass Storage Devices

Human Interface Devices (HID)

Mỗi thiết bị gắn vào cổng USB đều được điều khiển bởi một bộ điều khiển USB Controller. Bắt đầu từ phiên bản nhân 2.2.7, Linux mới hỗ trợ USB Controller. Có 3 kiểu USB Controllers như sau:

Host Controller	Kernel Module
<i>OHCI (Compaq)</i>	<i>usb-ohci.o</i>
<i>UHCI (Intel)</i>	<i>usb-uhci.o</i>
<i>EHCI (USB v 2.0)</i>	<i>ehci-hdc.o</i>

SCSI

Hiện nay, chuẩn SCSI có hai kiểu giao tiếp là:

- Chuẩn giao tiếp 8-bit với một kênh truyền hỗ trợ 8 thiết bị SCSI. Tuy nhiên do đã bao gồm cả controller nên card SCSI theo chuẩn này chỉ có thể kết nối được với tối đa 7 thiết bị SCSI khác.

Quản trị Hệ thống Linux - Cơ bản


- Chuẩn giao tiếp 16-bit (WIDE) cũng tương tự như chuẩn 8-bit với số thiết bị kết nối tối đa là 15.

Mỗi thiết bị SCSI được gán một số hiệu SCSI ID duy nhất. Các số hiệu này chạy từ 0 đến 7 hoặc 15 và có thể được thiết lập bởi các jump trên chúng.


Hệ thống sẽ tự động khởi động từ thiết bị có SCSI ID = 0. Tuy nhiên, thứ tự này có thể thay đổi được trong menu SCSI khi hệ thống khởi động

Network Card


Để dùng được card mạng, nhân của hệ thống phải hỗ trợ chúng. Thông tin về card mạng đang dùng trong hệ thống có thể được tìm thấy thông qua các lệnh hoặc file sau: **dmesg**, **lspci**, **scanpci**, **/proc/interrupts**, **/sbin/lsmmod** hay **/etc/modules.conf**



```
dmesg
Linux Tulip driver version 0.9.14 (February 20, 2001)
PCI: Enabled device 00:0f.0 (0004 ->0007)
PCI: Found IRQ 10 for device 00:0f.0
eth0: Lite-On 82c168 PNIC rev 32 at 0xf800, 00:0A:CC:D3:6E:0F,
IRQ 10
eth0: MII transceiver #1 config 3000 status 7829 advertising
```



```
cat /proc/interrupts
0: 8729602 XT-PIC timer
1: 4 XT-PIC keyboard
2: 0 XT-PIC cascade
7: 0 XT-PIC parport0
8: 1 XT-PIC rtc
10: 622417 XT-PIC eth0
11: 0 XT-PIC usb-uhci
14: 143040 XT-PIC ide0
15: 180 XT-PIC ide1
```



```
/sbin/lsmmod
Module Size Used by
tulip 37360 1 (autoclean)
```

Ví dụ trên cho thấy một card mạng đang dùng có chipset là Tulip, địa chỉ I/O là 0xf800 và IRQ = 10. Nếu các thông tin này hoạt động tốt thì có thể sử dụng nó

tại các hệ thống khác tương đương về phân cứng nhưng có lỗi khi nạp module hay I/O hoặc IRQ bị xung đột. Để thay đổi các tham số này có thể sử dụng các lệnh **modprobe** hoặc **insmod** (thay đổi trực tiếp) hay ghi vào trong file **/etc/modules.conf** (có hiệu lực từ lần khởi động sau).


Modem

Do các Modem cắm trong yêu cầu CPU xử lý dữ liệu cho chúng nên thông thường Linux không hỗ trợ các thiết bị loại này (mặc dù có nhiều cách đi đường vòng để giải quyết vấn đề trên).

Vì vậy, tài liệu này chỉ đề cập đến các modem cắm ngoài (sử dụng cổng serial). Trong Linux, các cổng serial được định nghĩa khác so với trong DOS/Windows:

DOS	Linux
COM1	/dev/ttyS0
COM2	/dev/ttyS1
COM3	/dev/ttyS2

Mặc dù hầu hết các phiên bản phân phối Linux đều có công cụ đồ họa để dò tìm và thiết lập tham số cho modem, nhưng trong các server chỉ sử dụng giao diện text, **setserial** là công cụ hữu ích cho việc này. Với tham số **-g**, **setserial** có thể tìm được cổng serial nào đang có thiết bị kết nối. Ngoài ra, **setserial** cũng có khả năng thiết lập lại tham số hoạt động cho những cổng này.

```
 setserial -g /dev/ttyS*

▶ /dev/ttyS0, UART: 16550A, Port: 0x03f8, IRQ: 4
  /dev/ttyS1, UART: 16550A, Port: 0x02f8, IRQ: 3
```

```
 ln -s /dev/ttyS1 /dev/modem
```

Để thiết lập kết nối và quay số trong môi trường text, có thể dùng bộ công cụ **wvdial**, **wvdialconf**, **wvdial.conf**.

Máy in

Hướng dẫn chi tiết hơn về máy in sẽ được đề cập đến trong những phần sau của tài liệu. Thông thường những máy in có khả năng PnP sẽ được dò tìm ngay khi hệ thống khởi động (kể cả máy in USB cũng có thể được dò thấy) và có thể nhìn thấy bởi lệnh **dmesg**.

Quá trình In trong Linux được thực hiện trong hai bước. Đầu tiên, dữ liệu in được lọc qua một bộ lọc theo định dạng của trình quản lý máy in. Sau đó, dữ liệu mới được xử lý để đưa ra máy in.

Bài tập

1. Sử dụng lệnh **dmesg** để xem thông tin từ file **/var/log/dmesg**. Tìm trong đó các thông tin về USB, tty hoặc eth0 và trả lời:

- Tên của USB controllers được sử dụng?
- Số hiệu IRQ của hai cổng serial đầu tiên là bao nhiêu?

2. Kiểm tra nội dung của các file:

/proc/ioports

/proc/interrupts

/proc/pci

/proc/dma

3. PCI bus:

- Kiểm tra output của các lệnh **lspci -v** and **scanpci -v**. Kiểu của card mạng trên máy bạn là gì?
- Kiểm tra xem có bao nhiêu mục 'bus' trong file **/proc/pci**. Những thông tin này có giống như kết quả của 2 lệnh trên không?

4. USB:

- Dùng lệnh **lsmod** và **lsusb** để kiểm tra xem kiểu host controller nào đang được sử dụng trong hệ thống? UHCI, OHCI hay EHCI.

QUẢN LÝ THIẾT BỊ

Đĩa và Phân vùng

Đĩa vật lý

Đĩa vật lý được nhân Linux gán vào các mục trong thư mục /dev. Mọi kết nối từ nhân đến các thiết bị đều thông qua bộ số major/minor. Các số major được định nghĩa trong file /proc/devices. Ví dụ: Đĩa cứng IDE đầu tiên có số major = 3

Block devices:

1	<i>ramdisk</i>
2	<i>fd</i>
3	<i>ide0</i>

Để nhận dạng các ổ cứng trong /dev, Linux dùng hai ký tự bắt đầu là hd cho các thiết bị IDE, sd cho các thiết bị SCSI hoặc ổ đĩa USB (nhưng lại dùng st cho ổ băng SCSI). Sau đó là các ký tự thêm vào để định danh các thiết bị cùng họ:

<i>hda</i>	Primary Master
<i>hdb</i>	Primary Slave
<i>hdc</i>	Secondary Master
<i>hdd</i>	Secondary Slave
<i>sda</i>	First SCSI/USB disk
<i>sdb</i>	Second SCSI/USB disk

Phân vùng đĩa

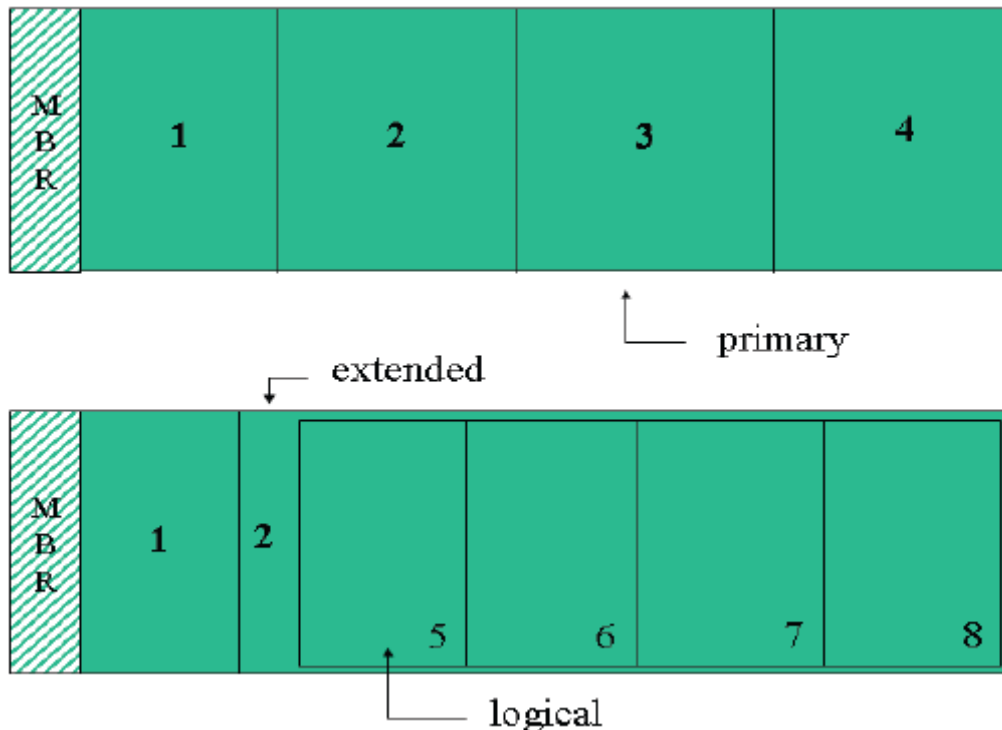
Để có thể sử dụng được, các đĩa cứng cần phải được phân vùng. Linux thêm vào đằng sau định danh đĩa cứng số hiệu của các phân vùng để quản lý.

<i>hda1</i>	Partition đầu tiên trên ổ IDE đầu tiên
<i>hda2</i>	Partition thứ hai trên ổ IDE đầu tiên
<i>sdc3</i>	Partition thứ ba trên ổ SCSI thứ ba

Mỗi ổ IDE chỉ cho phép có 4 phân vùng chính và một trong số chúng có thể được đánh dấu là phân vùng mở rộng. Phân vùng này có thể được đánh chia thành

Quản trị Hệ thống Linux - Cơ bản

hiều phân vùng con bên trong. Linux hỗ trợ tối đa 64 phân vùng trên ổ IDE và 16 phân vùng trên ổ SCSI.



Typical output of **fdisk -l**

Device	Boot	Start	End	Blocks	Id	System
/dev/hda1	*	1	748	6297448+	b	Win95 FAT32
/dev/hda2		785	788	32130	83	Linux
/dev/hda3		789	2432	13205430	5	Extended
/dev/hda5		789	1235	3590496	83	Linux
/dev/hda6		1236	1618	3076416	83	Linux
/dev/hda7		1619	1720	819283+	83	Linux
/dev/hda8		1721	1784	514048+	83	Linux
/dev/hda9		1785	1835	409626	83	Linux
/dev/hda10		1836	1874	313236	83	Linux
/dev/hda11		1875	1883	72261	82	Linux swap

Trong ví dụ trên (dùng **fdisk -l**), hệ thống có ba phân vùng chính được định danh từ hda1 đến hda3. Phân vùng thứ 3 được đánh dấu là mở rộng và chứa trong nó 7 phân vùng con. Do đó hda3 không được dùng. Các phân vùng con được định danh từ hda5 trở đi.

Công cụ Phân vùng đĩa

Trước khi cài đặt Linux

PartitionMagic

fips

fdisk

...

Trong khi cài đặt Linux

Trong quá trình cài đặt Linux, có thể sử dụng chính công cụ Tự động phân vùng của một số bản phân phối hoặc dùng công cụ phân vùng thủ công đi kèm:

diskdrake **Mandrake**

DiskDruid **RedHat**

Trên hệ thống đang hoạt động

fdisk luôn là công cụ được lựa chọn để phân vùng các đĩa cứng. Tập lệnh của *fdisk* tương đối đơn giản, chỉ cần gõ lệnh *m* trong giao diện lệnh của *fdisk* để xem đầy đủ các lệnh của nó.

Sau khi *fdisk*, nếu có thay đổi bảng phân vùng, cần phải khởi động lại máy tính hoặc dùng *partprobe* để cập nhật động (lệnh này không đúng cho mọi trường hợp). Để sử dụng được các phân vùng này, phải định dạng chúng với các định dạng hệ thống file mà Linux hiểu được thông qua các lệnh: **mkfs** hoặc **mke2fs**.

Bootloader

Bootloader là chương trình mặc định được cài đặt trên MBR nhằm giúp máy tính lựa chọn được phân vùng khởi động, nạp bộ môi hệ điều hành và chuyển quyền kiểm soát cho hệ điều hành.

Các bản Linux được phân phối với hai Bootloader riêng. Tuy nhiên, chúng cũng nhận vai trò môi hệ điều hành nên có thể cài đặt vào BR của phân vùng khởi động chứ không nhất thiết phải cài đặt trên MBR.

LILO (the **L**inux **bootLO**ader)

Quản trị Hệ thống Linux - Cơ bản

Được thiết kế với 3 thành phần chính

LILO

Mã nhị phân của trình bootloader, được cài đặt trên MBR hoặc BR. Nó sẽ nạp mã khởi động giai đoạn 2 tại /boot/boot.b.

/etc/lilo.conf

File cấu hình của LILO với một số tham số như sau:

<i>boot*</i>	Nơi LILO được cài đặt (/dev/hda là MBR)
<i>install</i>	Nơi mã khởi động giai đoạn 2 được cài đặt (mặc định là boot.b)
<i>prompt</i>	Cho người dùng lựa chọn hệ điều hành khi khởi động máy.
<i>default</i>	Tên của file ảnh được nạp khi khởi động mặc định
<i>timeout</i>	Thời gian kết thúc lựa chọn
<i>image*</i>	Đường dẫn chỉ đến nhân để khởi động
<i>label*</i>	Tên của file ảnh
<i>root*</i>	Tên của đĩa chứa thư mục gốc của hệ thống file.
...	

/sbin/lilo

Công cụ dùng để đọc tham số từ /etc/lilo.conf và thiết lập cho LILO.

GRUB (the Grand Unified Bootloader)

Được phát triển sau LILO với một vài ưu điểm so với LILO. Thông tin chi tiết về GRUB có thể được xem qua lệnh **info**.

<i>Example</i>	<i>grub.conf</i>
	<pre>default=0 timeout=10 splashimage=(hd0,0)/grub/splash.xpm.gz title Linux (2.4.18-14) root (hd0,0) kernel /vmlinuz-2.4.18-14 ro root=/dev/hda5 initrd /initrd-2.4.18-14.img</pre>

Những thiết bị đã quản lý

File `/etc/fstab` lưu thông tin về các điểm kết nối xác định trước cho các thiết bị khối.

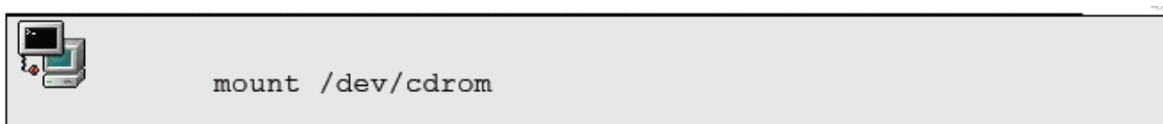
The `/etc/fstab` format

device	mount-point	fstype	options	dump-number	fsck-number
--------	-------------	--------	---------	-------------	-------------

Sample `/etc/fstab`

<code>LABEL=/</code>	<code>/</code>	<code>ext2</code>	<code>defaults</code>	<code>1</code>	<code>1</code>
<code>LABEL=/boot</code>	<code>/boot</code>	<code>ext2</code>	<code>defaults</code>	<code>1</code>	<code>2</code>
<code>LABEL=/home</code>	<code>/home</code>	<code>ext3</code>	<code>defaults</code>	<code>1</code>	<code>2</code>
<code>/dev/fd0</code>	<code>/mnt/floppy</code>	<code>auto</code>	<code>noauto,owner</code>	<code>0</code>	<code>0</code>
<code>LABEL=/usr</code>	<code>/usr</code>	<code>ext2</code>	<code>defaults</code>	<code>1</code>	<code>2</code>
<code>LABEL=/var</code>	<code>/var</code>	<code>ext3</code>	<code>defaults</code>	<code>1</code>	<code>2</code>
<code>none</code>	<code>/proc</code>	<code>proc</code>	<code>defaults</code>	<code>0</code>	<code>0</code>
<code>none</code>	<code>/dev/shm</code>	<code>tmpfs</code>	<code>defaults</code>	<code>0</code>	<code>0</code>
<code>none</code>	<code>/dev/pts</code>	<code>devpts</code>	<code>gid=5,mode=620</code>	<code>0</code>	<code>0</code>
<code>/dev/hdc9</code>	<code>swap,pri=-1</code>	<code>swap</code>	<code>defaults</code>	<code>0</code>	<code>0</code>
<code>/dev/cdrom</code>	<code>/mnt/cdrom</code>	<code>iso9660</code>	<code>noauto,owner,kudzu,ro</code>	<code>0</code>	<code>0</code>

Ngoài ra, `/etc/fstab` cũng được dùng để trợ giúp cho các kết gán tài nguyên thời gian thực. Ví dụ:




Chương trình **mount** sẽ đọc `/etc/fstab` và quyết định tài nguyên (hoặc điểm kết nối) nào sẽ được sử dụng và các tham số của việc kết nối cũng có thể được xác định tại bước này. Sau đây là một số tham số tùy chọn (option) của mount:

<i>rw, ro</i>	đọc-ghi hoặc chỉ đọc
<i>users</i>	có thể đọc và umount bởi mọi người dùng
<i>user</i>	chỉ có thể umount bởi người dùng đã mount nó
<i>owner</i>	có thể thay đổi quyền và thuộc về người dùng đã mount nó

<i>usrquota</i>	bật thiết lập hạn ngạch đĩa mức người dùng
<i>grpquota</i>	bật thiết lập hạn ngạch đĩa mức nhóm người dùng

Nếu sử dụng với tham số **-a**, mount sẽ tự động ánh xạ tất cả các khai báo trong */etc/fstab* mà chưa được mount và không có tùy chọn **noauto**.

Một số thiết bị được truy cập thông qua các nhãn. Nhãn được gán cho thiết bị bởi lệnh **tune2fs**:




```
tune2fs -L /usr/local /dev/hdb12
```

Quotas

Quota là công cụ cho phép quản trị hệ thống thiết lập hạn ngạch lưu trữ trên đĩa. Công cụ này không yêu cầu khởi động lại hệ thống. Sau đây là một số bước làm chung:

1. Thêm tùy chọn *usrquota* vào file */etc/fstab* tại dòng chứa phân vùng cần phân hạn ngạch.
2. Remount lại phân vùng này:



```
mount -o remount <device>
```

3. Thiết lập tình trạng quota:



```
quotacheck -ca
```

Sau lệnh này, nếu thiết lập đúng, file *aquota.user* sẽ được sinh ra tại thư mục gốc của phân vùng.

4. Sửa lại hạn ngạch cho từng người dùng:



Tham số soft/hard limit phải được thiết lập cho cả số blocks lẫn inodes cho mỗi user. Hệ thống sẽ cho phép sử dụng vượt quá con số soft limit cho đến khi hết hạn về mặt thời gian. Khi đó, hard limit sẽ được sử dụng để quyết định chính xác hạn ngạch của người dùng. Sử dụng tham số -T để quyết định thời gian này.

5. Bật chế độ hạn ngạch lên:



Người dùng có thể kiểm tra hạn ngạch của mình bằng lệnh **quota**, quản trị có thể sinh ra báo cáo về hạn ngạch bằng lệnh **repquota** hoặc **quotastats**.

Bài tập

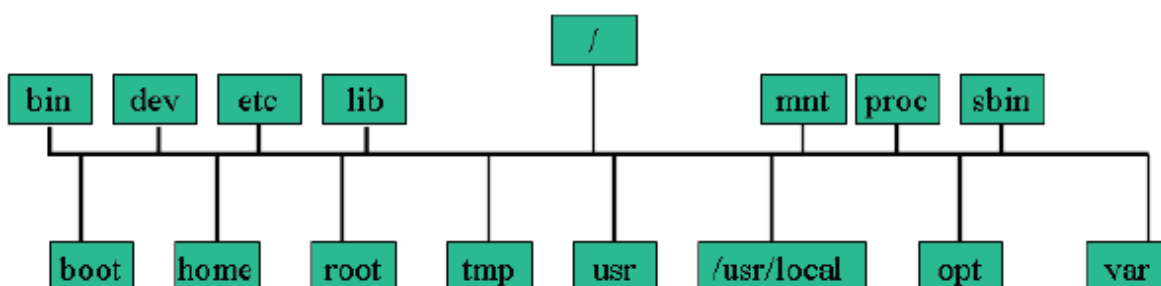
1. Sử dụng fdisk, xóa phân vùng /home, sau đó tạo lại 1 phân vùng mới. Khởi động lại máy tính. Vấn đề gì sẽ xảy ra? Giải quyết như thế nào?
2. Dùng lệnh mkfs tạo ra định dạng hệ thống file kiểu ext3 trên phân vùng này
3. Tạo thư mục data trong thư mục gốc. Thiết lập lại /etc/fstab sao cho thư mục này là mount point của phân vùng mới định dạng.
4. Dùng lệnh mount có tham số để kiểm tra lại xem đã thiết lập /etc/fstab đúng chưa.
5. Thiết lập hạn ngạch đĩa cho phân vùng trên theo từng bước đã hướng dẫn.

HỆ THỐNG FILE TRONG LINUX

Cấu trúc của hệ thống file

Mỗi hệ thống file có cấu trúc giống như một cái cây dựng ngược. Gốc của cây được đặt trên cùng và bên dưới là lá của nó.

Như đã đề cập ở trên, mỗi phân vùng khi được tạo ra đều có thể có một mount point. Công việc này thường được thi hành trong quá trình cài đặt. Để hiểu kỹ hơn về vấn đề này, hãy quan sát kiến trúc phân cấp của một hệ thống file trong Linux dưới đây:



The base directories



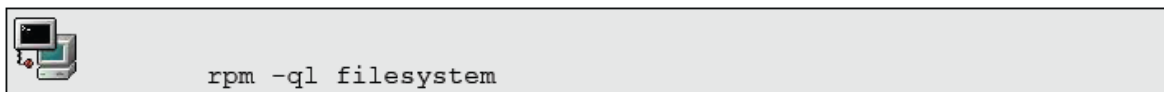
Directories that can be mount points for separate devices

Trong hình trên, gốc của kiến trúc phân cấp này là thư mục gốc “/”. Nó gần tương tự như “C:\” trong DOS ngoại trừ việc “C:\” chính là phân vùng đầu tiên của đĩa cứng đầu tiên, trong khi thư mục gốc “/” của Linux có thể là ánh xạ của bất kỳ phân vùng nào.

The base directories

Quản trị Hệ thống Linux - Cơ bản

Các thư mục cơ sở là những thư mục con cấp 1 nằm ngay dưới thư mục gốc “/”. Chúng được tạo ra bởi một gói thường có tên là filesystem.



Tiến trình khởi động sẽ ánh xạ thư mục gốc đầu tiên nhằm giúp đỡ tất cả các thao tác tiếp theo như kiểm tra phân vùng, nạp module cho nhân...vv vì khi ánh xạ thư mục gốc xong thì các chương trình như: **fsck**, **insmod** hay **mount** mới có thể được sử dụng.

Để đảm bảo cho quá trình khởi động diễn ra chính xác, các thư mục **/dev**, **/bin**, **/sbin**, **/etc** và **/lib** bắt buộc phải là thư mục con của “/” và không thể là ánh xạ của bất kỳ phân vùng nào khác.

Sau đây là một số thư mục cơ sở và giải thích ngắn gọn ý nghĩa của chúng:

/bin và **/sbin**

Chứa những file cần thiết cho quá trình khởi động và những lệnh thiết yếu để duy trì hệ thống.

/dev

Chứa các định danh ánh xạ của thiết bị hoặc những file đặc biệt.

/etc

Chứa các file cấu hình của hệ thống và nhiều chương trình tiện ích.

/lib

Chứa các thư viện dùng chung cho các lệnh nằm trong **/bin** và **/sbin**. Và thư mục này cũng chứa các module của nhân.

/mnt hoặc **/media**

Mount point mặc định cho những hệ thống file kết nối bên ngoài.

/proc

Quản trị Hệ thống Linux - Cơ bản

Lưu các thông tin của nhân, chỉ có thể ghi được nội dung trong thư mục **/proc/sys**.

/boot

Chứa nhân Linux để khởi động và các file system maps cũng như các file khởi động giai đoạn hai.

/home (tùy chọn)

Thư mục dành cho người dùng khác root. Thông tin khởi tạo thư mục mặc định của người dùng được đặt trong **/etc/skel/**

/root (tùy chọn)

Thư mục mặc định của người dùng root.

/tmp

Thư mục chứa các file tạm thời.

/usr

Thư mục chứa những file cố định hoặc quan trọng để phục vụ tất cả người dùng.

/usr/local hoặc **/opt** (tùy chọn)

Thư mục chứa các phần mềm cài thêm.

/var/www, **/var/ftp** hoặc **/srv** (Suse)

Thư mục chứa thông tin của các dịch vụ WEB hay FTP.

/var

Thư mục chứa các thông tin hay thay đổi như: spool và log

Hệ thống file chuẩn ext2

Để có thể lưu trữ và quản lý dữ liệu, mỗi phân vùng trên đĩa cứng đều phải được tạo ra một hệ thống file. Ngay trước khi khởi tạo, bao giờ người thiết lập cũng phải chỉ định kiểu định dạng của hệ thống file mới cần tạo.

Hiện nay, nhân Linux hỗ trợ rất nhiều kiểu định dạng của hệ thống file. Trong đó, kiểu hệ thống file **ext2** được coi là mặc định trong các hệ thống của Linux “Linux Native” (Trong nhiều hệ thống **ext3** được coi là mặc định nhưng thực tế **ext3** chính là **ext2** kèm thêm chức năng journal).

Một kiểu khác của hệ thống file cũng hay được dùng là SWAP. Kiểu định dạng hệ thống file này chỉ được dùng cho phân vùng swap.

The Second Extended File System

Ext2 là kiểu định dạng hệ thống file được thiết kế dựa trên việc quản lý các khối dữ liệu có kích thước 1KB (1024 byte), đây là kích thước mặc định và có thể thay đổi được. Có 3 loại khối như trên được định nghĩa trong ext2:

Superblocks

Lặp lại sau mỗi 8193 khối. Khối này chứa thông tin như: block-size, free inodes, last mounted time ...

Inodes

Chứa các con trỏ trỏ đến khối dữ liệu. 12 khối dữ liệu đầu tiên được truy cập trực tiếp từ con trỏ này. Nếu dữ liệu > 12KB thì các inodes gián tiếp sẽ được sử dụng.

Mỗi inode bao gồm 256 byte và chứa các thông tin về user, group, permissions và time stamp của dữ liệu mà nó quản lý.

Khối dữ liệu

Có thể là file hoặc thư mục với nội dung thật được chứa trong các khối này.

Tiện ích định dạng

Do nhân Linux chỉ có thể đọc được các hệ thống file đã được định dạng từ trước nên để lưu trữ và quản lý dữ liệu trên các phân vùng mới, cần phải định dạng một hệ thống file trên đó thông qua các công cụ định dạng.

Để định dạng một phân vùng có kiểu hệ thống file là **ext2** bằng lệnh **mkfs.ext2** hay **mke2fs**. Tương tự như vậy với kiểu hệ thống file **xfs** (của Silicon Graphics) với lệnh **mkfs.xfs**.

Lệnh **mkfs** thực chất là một chương trình kiểm tra yêu cầu định dạng và lựa chọn đúng lệnh để thi hành. Cú pháp của mkfs là:

mkfs -t <fstype>



```
mkfs -t jfs /dev/hda12
```



```
mke2fs /dev/hda11 [or mkfs -t ext2 /dev/hda11]
```

Sự an toàn của hệ thống file

Nếu hệ thống file bị hỏng hoặc sai lệch, tiện ích **fsck** được sử dụng để chỉnh sửa lại các hư hỏng này tuy nhiên các hệ thống file này cần phải unmount trước đó để đảm bảo tính chính xác.

Cũng như **mkfs**, **fsck** thực chất chỉ kiểm tra các tham số của người dùng và lựa chọn đúng chương trình để thi hành, ví dụ: **fsck.ext2**, **fsck.ext3**



```
fsck -t reiserfs /dev/sdb10  
fsck.reiserfs /dev/sdb10
```


Kiểm soát dung lượng đĩa

Sử dụng mount và df

Cả hai lệnh trên đều cùng hoạt động ở cùng mức thiết bị. Hai lệnh **mount** và **umount** dùng để quản trị các hệ thống file đã gắn kết trong file **/etc/mtab**.

Nếu sử dụng **mount** không tham số, tất cả các hệ thống file được gắn kết trong hệ thống sẽ được liệt kê ra màn hình. Kết quả giống như trong file **/etc/mtab**. Ngoài ra, nhân cũng lưu giữ thông tin về hệ thống file đã được kết nối trong **/proc/mount**.

Để xem thêm thông tin về điểm kết nối hiện tại có thể sử dụng lệnh **df**. Lệnh này cho phép hiển thị thêm dung lượng đĩa đã sử dụng và dung lượng còn trống. Đơn vị kích thước để hiển thị là 1K.



df -h

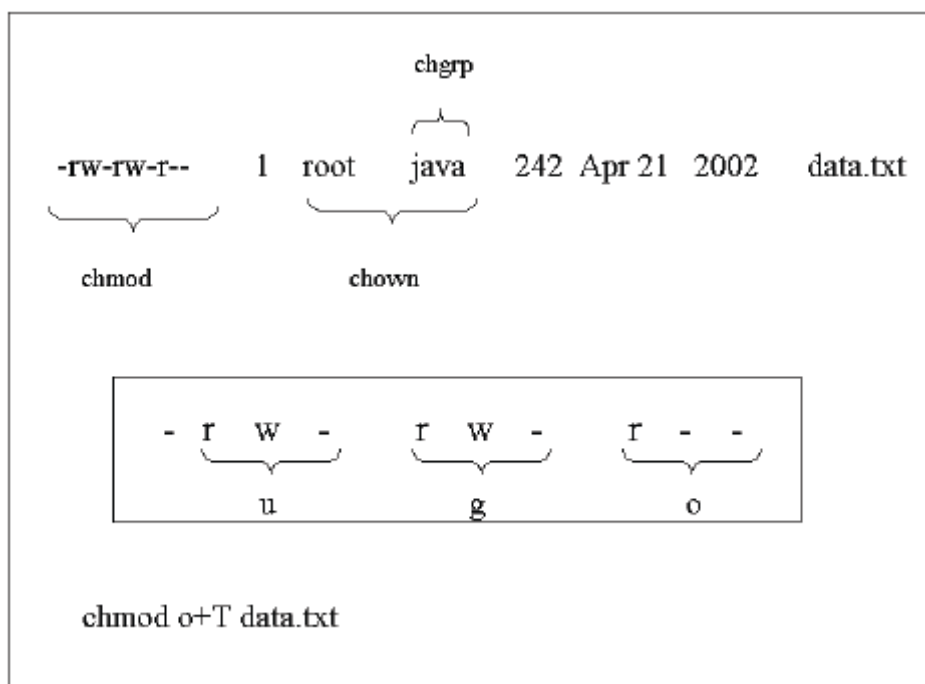
→

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/hda9	289M	254M	20M	93%	/
/dev/hda2	23M	7.5M	14M	35%	/boot
none	62M	0	61M	0%	/dev/shm
/dev/hda5	1.4G	181M	1.1G	13%	/share
/dev/hda7	787M	79M	669M	11%	/tmp
/dev/hda3	4.3G	3.4G	813M	81%	/usr
/dev/hda6	787M	121M	627M	17%	/var

Sử dụng du

Tiện ích này được sử dụng để hiển thị không gian đĩa được sử dụng nhưng ở mức thư mục. Vì vậy, du cũng không thể hiển thị khoảng trống còn thừa của đĩa.

Quyền truy xuất File, Thư mục



Thay đổi quyền truy xuất và chủ sở hữu

Quyền truy xuất file, thư mục và chủ sở hữu được định nghĩa để quy định cách thức truy cập dữ liệu trong hệ thống.


Quản trị Hệ thống Linux - Cơ bản

Để thay đổi quyền truy cập, sử dụng lệnh **chmod**. Có ba nhóm đối tượng chính được tác động bởi quyền truy cập là:


<i>u</i>	Người dùng sở hữu
<i>g</i>	Nhóm người dùng sở hữu
<i>o</i>	Không thuộc hai đối tượng trên

Ví dụ:

-rw-rw-r-- 1 jade sales 24880 Oct 25 17:28 libcgic.a



```
chmod g=r,o-r libcgic.a
chmod g+w libcgic.a
```



```
chown root libcgic.a
chgrp apache libcgic.a
```

Tùy chọn hay dùng với chmod, chown và chgrp là **-R** cho phép thay đổi trong cả các thư mục, file bên trong thư mục chỉ định.

Ngoài cách sử dụng ký tự đại diện cho các quyền: read=r, write=w, execute=x, chmod cho phép sử dụng một bộ số hệ bát phân để thay đổi quyền theo bảng sau:

<i>read</i>	4
<i>write</i>	2
<i>execute</i>	1

user	group	other
<i>rwX</i>	<i>r-X</i>	<i>rw-</i>
$4+2+1=7$	$4+1=5$	$4+2=6$

Quyền truy xuất chuẩn

Các hệ thống UNIX tạo ra file và thư mục với quyền truy xuất chuẩn như sau::

<i>Files</i>	<i>666</i>	<i>-rw-rw-rw-</i>
<i>Directories</i>	<i>777</i>	<i>-rwxrwxrwx</i>

umask

Là khái niệm được thiết lập để chỉ định quyền truy xuất mặc định cho các file và thư mục mới tạo **đối với mỗi người dùng**. umask là một mặt nạ gồm một bộ các số hệ bát phân. Khi đó, quyền truy xuất mặc định của các file và thư mục đối với mỗi người dùng được tính theo công thức sau:

$$\text{Final Permissions} = \text{Standard Permissions (logical AND)} (\text{NOT}) \text{Umask}$$

Quyền truy cập SUID

Là quyền truy cập được thiết lập bởi root cho phép người dùng bình thường có thể thi hành một lệnh như là root. Quyền này được thiết lập với tên là s (nằm ở vị trí x của nhóm u) và được gán số hệ bát phân là 4000.

Quyền truy cập SGID

Là quyền truy cập cho phép người dùng thuộc nhóm sở hữu có thể thi hành lệnh mà không cần dùng **newgrp** để chuyển nhóm. Quyền này được thiết lập với tên là s (nằm ở vị trí x của nhóm g) và được gán số hệ bát phân là 2000. Tại thư mục được thiết lập SGID, tất cả các file, thư mục tạo bên trong sẽ có nhóm sở hữu mặc định là nhóm sở hữu của thư mục cha.

Bit đánh dấu (The sticky bit)

Quyền này được thiết lập với tên là t (nằm ở vị trí x của nhóm o) và được gán số hệ bát phân là 1000. Quyền này được thiết lập để:

- Cho phép các thư mục cấm người dùng xóa file trừ phi họ là chủ sở hữu.
- Cho phép file được thi hành hoặc nạp vào bộ nhớ nhanh hơn.

Bài tập

Filesystem

1. Xóa phân vùng được ánh xạ vào /data của bài trước, tạo ra 2 phân vùng mới có kiểu định dạng của hệ thống file là ext2 và reiserfs.

2. Tạo 2 thư mục con trong /mnt và ánh xạ hai phân vùng mới vào.

```
mkdir /mnt/ext2
```

```
mkdir /mnt/reiserfs
```

3. Sử dụng các lệnh mount, df, fsck để kiểm tra đối với 2 phân vùng mới tạo.

4. Chuyển đổi từ ext2 sang ext3 bằng lệnh tune2fs

File permissions

1. Login bằng 1 người dùng không phải root và tạo 1 file mới bằng lệnh touch. Kiểm tra xem quyền truy xuất của file này là gì?

2. Thay đổi umask thành 027. Quyền truy xuất mặc định sẽ là gì?

3. Nơi nào sẽ thiết lập giá trị mặc định của umask? /etc/profile, /etc/bashrc...

4. Thêm 2 người dùng mới user1, user2 với password tương ứng. Tạo nhóm mới sales. Và thêm 2 người dùng mới tạo vào nhóm này.

5. Tạo thư mục /news sở hữu bởi nhóm sales và có quyền 770 cho thư mục này. Sau đó đặt GID cho thư mục này.

6. Kiểm tra các tính chất của GID với user1 và user2.

7. Thêm Sticky-Bit cho thư mục /news. Kiểm tra tính chất của bit này.

CHẾ ĐỘ DÒNG LỆNH

Khái quát

Sử dụng dòng lệnh là cách cơ bản để tương tác với hệ thống máy tính. Bộ biên dịch shell (hệ vỏ) thông dịch các lệnh được nhập vào từ bàn phím. Dấu nhắc shell (\$ hoặc # đối với người quản trị hệ thống) cho biết hệ thống đã sẵn sàng hoạt động.

Shell còn là một môi trường lập trình cho phép thực hiện các lệnh khởi động. Chương trình shell được gọi là script (kịch bản).

Most Common shells	
The Bourne shell	/bin/sh
The Bourne again shell	/bin/bash
The Korn shell	/bin/ksh
The C shell	/bin/csh
Tom's C shell	/bin/tcsh

Do bash shell là một trong những shell thông dụng nhất trong cộng đồng linux, vì thế tài liệu này tập trung chủ yếu vào bash shell.

Tương tác với SHELL

Các câu lệnh thực hiện trên shell có dạng sau:

command [options] {arguments}

Hiển thị xâu kí tự ra màn hình

Bash shell sử dụng lệnh **echo** để hiển thị xâu kí tự ra màn hình

```
echo "this is a short line"
```

Bí danh

Chúng ta có thể tạo các bí danh cho các lệnh sử dụng nhiều tham số. Cách thức để tạo một bí danh là như sau:

```
alias myprog='command [option] {arguments}'
```

Bằng cách chỉ gõ **alias** tại một dòng lệnh, chúng ta sẽ có danh sách của các bí danh đã được định nghĩa.

Đường dẫn tuyệt đối/tương đối

Shell thông dịch từ đầu tiên của bất kỳ dòng lệnh nào như là một câu lệnh. Nếu dòng lệnh có một **đường dẫn tuyệt đối hoặc tương đối** đến câu lệnh thì câu lệnh sẽ được thực thi. Nếu không thì shell sẽ tìm kiếm trong alias. Nếu từ đầu tiên không có kí tự “/” thì shell sẽ tìm kiếm ở các thư mục đã được khai báo trong nội dung biến môi trường PATH và thực hiện chương trình có tên trùng với câu lệnh.

Ví dụ nếu tham biến PATH chỉ chứa các thư mục **/bin** và **/usr/bin** thì câu lệnh **xeyes** sẽ không được tìm thấy khi mà nó nằm trong **/usr/X11R6/bin/xeyes** và vì thế đường dẫn tuyệt đối là cần thiết để cho câu lệnh này được thực thi.

```
/usr/X11R6/bin/xeyes
```

Người dùng có thể sử dụng đường dẫn tương đối thay cho đường dẫn tuyệt đối trong khi thực hiện một câu lệnh. Ví dụ nếu người dùng đang truy cập vào thư mục chứa chương trình **xeyes** thì họ có thể sử dụng câu lệnh sau:

```
./xeyes
```

Tự điền kết thúc câu lệnh

Bằng cách ấn phím **TAB**, shell sẽ kết thúc câu lệnh mà chúng ta đang gõ vào.

Biến môi trường của Shell

Các biến của Shell giống như các biến được sử dụng trong các ngôn ngữ máy tính khác. Các tên biến được giới hạn trong các ký tự chữ số. Ví dụ CREDIT=300 có nghĩa là gán giá trị 300 cho biến có tên là CREDIT

Khởi tạo một biến

Tên biến=giá trị (không có dấu cách)

Tham chiếu một biến

\$Tên biến

```
CREDIT=300  
echo $CREDIT
```

Export, Set và Env

Có hai loại biến: biến cục bộ và biến xuất (có thể gọi là biến toàn cục địa phương, thực tế khái niệm này không thể đồng nghĩa với khái niệm biến toàn cục trong các ngôn ngữ lập trình. Tuy vậy để cho ngắn gọn, tất cả các thể hiện biến toàn cục xuất hiện trong tài liệu này đều có ý nghĩa như biến xuất).

Biến cục bộ chỉ được truy cập bởi shell hiện thời. Trong khi đó biến xuất sẽ được truy cập bởi cả shell và bất kỳ tiến trình con của shell này.

Các lệnh **set** và **env** dùng để hiển thị các biến đã được định nghĩa

Các lệnh set và env

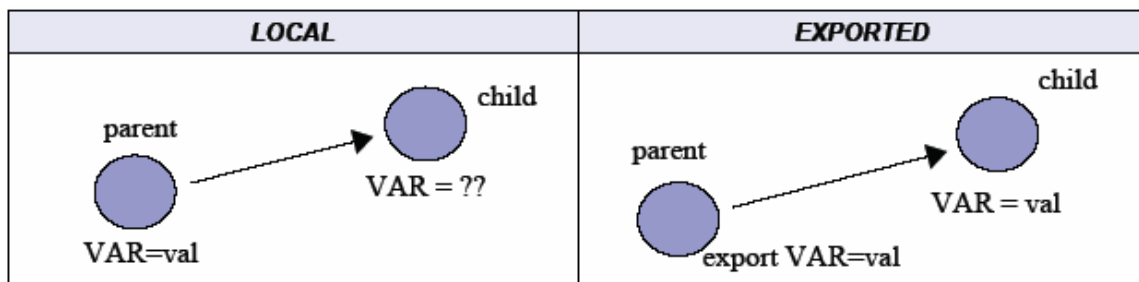
set

Hiển thị tất cả các biến

env

Hiển thị tất cả các biến xuất

Một biến được gọi là biến toàn cục khi bất kỳ tiến trình con nào cũng có thể tham chiếu đến nó.



Ví dụ: Tạo biến CREDIT là biến toàn cục. Hiện thị nó với lệnh **set** hoặc **env**.

```
export CREDIT
env | grep CREDIT
```

Khởi tạo một shell mới (tiến trình con) và kiểm tra xem biến CREDIT có được truy cập đến không?

Biến định nghĩa trước	Ý nghĩa
<i>DISPLAY</i>	<i>Được X dùng để định vị ứng dụng khách (client)</i>
<i>HISTFILE</i>	<i>Chỉ đến file lịch sử lệnh của người dùng .bash_history</i>
<i>HOME</i>	<i>Chỉ đến thư mục dành riêng (home) của người dùng</i>
<i>LOGNAME</i>	<i>Tên được sử dụng bởi người dùng để truy nhập</i>
<i>PATH</i>	<i>Chứa những thư mục sẽ được tìm kiếm bởi shell khi người dùng thực hiện chương trình mà không chỉ ra đường dẫn</i>
<i>PWD</i>	<i>Thư mục làm việc hiện thời</i>
<i>SHELL</i>	<i>Tên shell được sử dụng</i>
<i>TERM</i>	<i>Mô phỏng thiết bị cuối hiện thời</i>

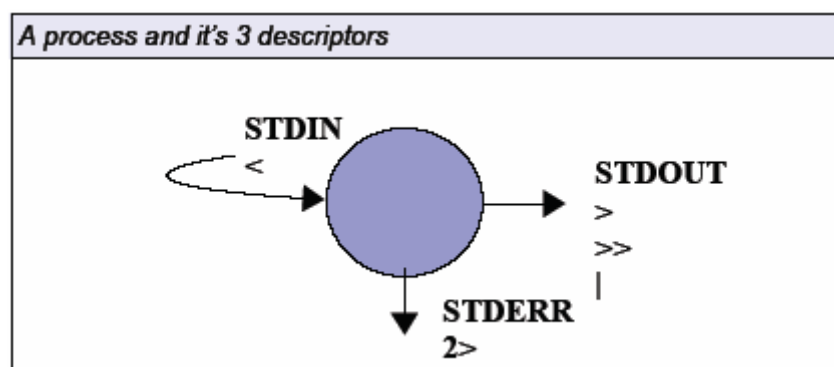
Các biến đặc biệt

Một số biến liên quan đến việc quản lý tiến trình

\$!	Hiện thị mã tiến trình (PID) của tiến trình con cuối cùng
\$\$	Hiện thị mã tiến trình (PID) của shell đang thực thi
\$?	Bằng 0 nếu lệnh cuối cùng được thực hiện thành công và 1 nếu ngược lại

Chuyển hướng kết xuất

Các tiến trình UNIX thông thường mở 3 dạng mô tả file chuẩn cho phép nó thực hiện việc xuất, nhập và báo lỗi. Các dạng mô tả chuẩn này có thể được định nghĩa lại bởi bất kỳ tiến trình nào. Trong hầu hết các trường hợp, mô tả **stdin** là bàn phím, và hai dạng mô tả xuất + báo lỗi (**stdout** và **stderr**) là màn hình.



Các giá trị cho stdin, stderr, và stdout	
<i>stdin</i>	<i>0</i>
<i>stdout</i>	<i>1</i>
<i>stderr</i>	<i>2</i>

- Đổi hướng stdout

program > file

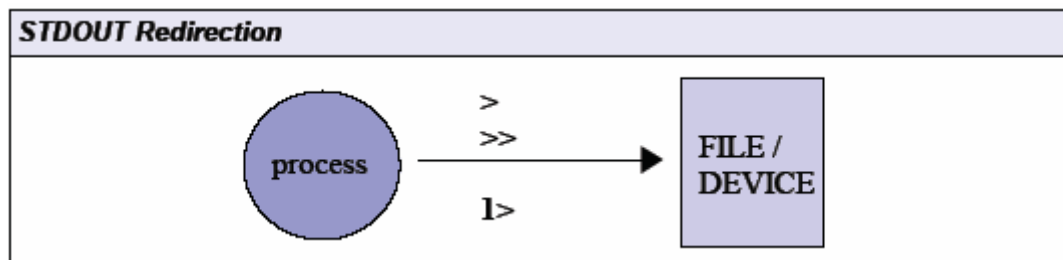
Dữ liệu theo hướng từ trái sang phải

```
fdisk -l > partions.txt
```

Câu lệnh này sẽ thực hiện tiện ích **fdisk** và kết quả đầu ra sẽ được ghi vào file *partitions.txt*. Kết quả không được hiển thị ra màn hình. Chú ý rằng shell sẽ thực

hiện câu lệnh này bắt đầu từ bên phải. Như vậy, file *partitions.txt* sẽ được tạo ra nếu như nó chưa tồn tại và sẽ bị ghi đè vào khi toán tử ‘>’ được dùng.

Toán tử ‘>>’ sẽ bổ sung thêm kết quả vào nội dung file.



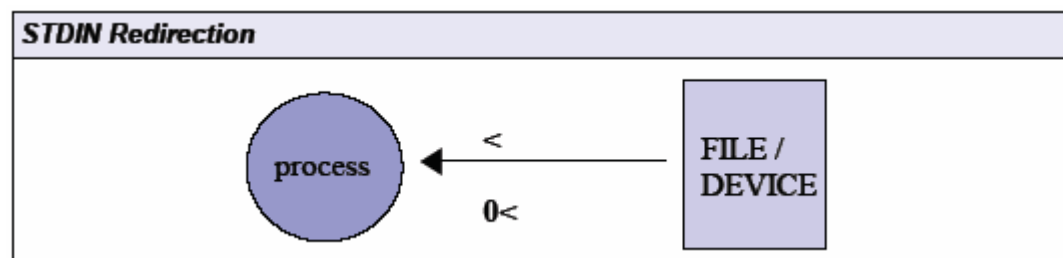
- **Đổi hướng stdin**

program < file

Trong trường hợp này dữ liệu theo hướng từ phải sang trái. Toán tử ‘<’ chỉ được sử dụng cho **stdin** và không thể dùng cho **stdout**.

Nếu file instruction chứa trên mỗi dòng các kí tự *p*, *m*, và *q* thì trong ví dụ sau đây **fdisk** sẽ in bảng phân vùng (partition) của */dev/hda*, in tiện ích trợ giúp, và cuối cùng là thoát khỏi câu lệnh.

```
fdisk /dev/hda < instructions
```

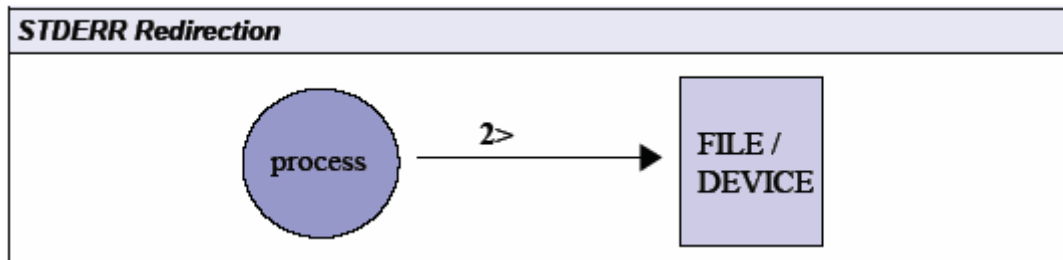


- **Đổi hướng stderr**

program 2> errorfile

stdin, stdout, và stderr được đại diện bằng 0, 1, và 2 tương ứng. Câu lệnh trên cho phép chúng ta chọn luồng stderr.

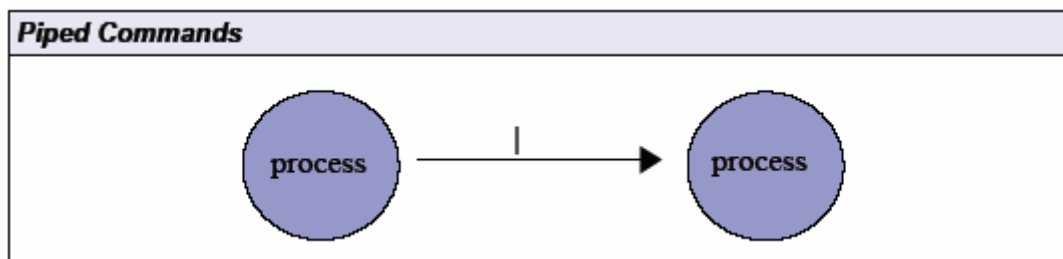
```
find / 2> /dev/null
```



- **Các lệnh đường ống**

Program1 | Program2

Các đường ống (pipe) được đại diện bằng kí hiệu “|”. Dòng dữ liệu chuyển từ trái sang phải. Hình sau đây minh họa stdout của một tiến trình được chuyển hướng đến stdin của một tiến trình khác như thế nào.



```
cat /var/log/messages | less
```

Các chuyển hướng của dữ liệu xuất được phân tách từ phải sang trái, do đó các lệnh sau là không tương đương:

```
command 2>&1 >logfile
```

```
command >logfile 2>&1
```

<< là sự đổi hướng cho kết thúc file (EOF)

Ví dụ câu lệnh:

```
cat << stop
```

sẽ chấp nhận các giá trị nhập chuẩn cho đến khi từ 'stop' được đưa vào.

Dấu ngoặc và Các ký tự Đa nghĩa (Metacharacter)

Metacharacter là các ký tự có nghĩa đặc biệt trong shell. Chúng được dùng chủ yếu cho *file globbing*, tức là đối sánh một vài file hoặc tên thư mục bằng một số lượng tối thiểu các ký tự.

Các ký tự nhập (<), xuất (>), và đường ống (|) cũng là các ký tự đặc biệt và ký tự \$ được dùng cho các biến. Các ký tự đặc biệt này sẽ không được liệt kê hết ở đây.

Các ký tự đại diện (wildcard)

- Ký tự * có thể đại diện cho 0 hoặc một số ký tự tùy ý

*ls /usr/bin/b** hiển thị tất cả các chương trình bắt đầu bằng ký tự 'b'

- Ký tự ? đại diện cho một ký tự tùy ý

*ls usr/bin/?b** hiển thị tất cả các chương trình có ký tự 'b' ở vị trí thứ 2

Các miền (range)

- [] được dùng để định nghĩa một miền các giá trị

ls a[0-9] hiển thị tất cả các file bắt đầu bằng ký tự 'a' và có một chữ số ở vị trí thứ 2.

`ls [!Aa]*` hiển thị tất cả các file không bắt đầu bằng ký tự 'a' hoặc 'A'

- {xâu1,xâu2} mặc dù chúng không được dùng để đại diện một họ tên file nhưng chúng có thể sử dụng để đối sánh với tên những file đã có.

```
ls index.{htm,html}
```

Các dấu ngoặc (quote) và mã escape

Ý nghĩa đặc biệt của các metacharacter có thể bị huỷ bỏ bằng các ký tự escape-chúng cũng là các metacharacter.

Dấu vạch chéo ngược (\) là một ký tự đặc biệt và huỷ bỏ ý nghĩa của tất cả các metacharacter yêu cầu shell thông dịch chúng.

Dấu ngoặc đơn (' ') huỷ bỏ nghĩa của tất cả các metacharacter ngoại trừ dấu vạch chéo ngược.

Dấu ngoặc kép (" ") có tác dụng yếu nhất nhưng cũng có thể huỷ bỏ phần lớn ý nghĩa đặc biệt của các ký tự nằm trong dấu ngoặc kép ngoại trừ đường ống (|), dấu vạch chéo ngược, và một biến (\$var).

Dấu nháy

Dấu nháy này giống dấu huyền của Tiếng Việt và thường được đặt cạnh số 1 của bàn phím đầy đủ.

Cặp dấu nháy (``) sẽ thực hiện câu lệnh nằm bên trong. Ví dụ sau đây sẽ định nghĩa biến `TIME` sử dụng lệnh **date**

```
TIME="Today's date is `date +%a:%d:%b`"  
echo $TIME  
Today's date is Sun:15:Jul
```

Một cách khác để thực hiện câu lệnh giống như sử dụng các dấu nháy đó là `$()`. Ví dụ dưới đây sẽ thực hiện câu lệnh ở bên trong và gán giá trị trả về vào biến `TIME`.

```
TIME=$(date)
```

Lịch sử dòng lệnh

Để xem danh sách các câu lệnh đã được sử dụng từ trước chúng ta có thể dùng **bash** gắn liền với lệnh **history**

```
history
1.      ls
2.      grep 500 /etc/passwd
```

Chúng ta có thể gọi lại các lệnh đã sử dụng bằng cách dùng mũi tên lên và xuống trên bàn phím. Ngoài ra còn có các liên kết phím emacs cho chúng ta thực hiện và sửa đổi các lệnh trước đó.

Emacs Key Bindings for Editing the Command History

<i>Ctrl+p</i>	<i>Lên trên 1 dòng</i>
<i>Ctrl+n</i>	<i>Xuống dưới 1 dòng</i>
<i>Ctrl+b</i>	<i>Quay lại (sang trái) 1 ký tự</i>
<i>Ctrl+f</i>	<i>Đi tiếp (sang phải) 1 ký tự</i>
<i>Ctrl+a</i>	<i>Về cuối dòng</i>
<i>Ctrl+e</i>	<i>Về đầu dòng</i>

Dấu chấm than (!) có thể được dùng để thực hiện các lệnh trước đó. Ví dụ:

<code>!x</code>	Thi hành lệnh gần nhất trong lịch sử lệnh có ký tự bắt đầu là 'x'
-----------------	---

!	Thi hành lệnh có số thứ tự = 2 trong lịch sử lệnh
!-2	Thi hành lệnh ngay trước lệnh vừa thi hành
!!	Thi hành lệnh vừa chạy
^string1^string2	Thi hành lệnh vừa chạy và thay thế string1 bởi string2

Bài tập

stdin-stdout-stderr

Gõ các câu lệnh sau đây và đưa ra các kết quả thực thi (nếu có thể) sử dụng các sơ đồ giống như những sơ đồ đã được dùng trong chương này

```
ls /etc ; df > /tmp/out.1
(ls /etc ; df) > /tmp/out.2
find /etc -type f 2> /dev/null | sort
tr [a-z] [A-Z] < /etc/passwd | sort > /tmp/passwd.tmp
cat /tmp/passwd.tmp | tr [A-Z] [a-z]
```

Dòng lệnh

1. Hiện thị tất cả các file trong /usr/X11R6/bin mà không bắt đầu với ký tự 'x'

```
ls /usr/X11R6/bin/[^x]*
```

2. Câu lệnh **xterm** có các lựa chọn sau:

-bg <màu> thiết lập màu nền

-fg <màu> thiết lập màu chữ

-e <câu lệnh> thực hiện câu lệnh

Thiết lập một bí danh mới sao cho câu lệnh **su** mở một xterm với các màu mới và lời nhắc cho mật khẩu chủ

```
alias su="xterm -bg orange -fg brown -e su -u &"
```

Bạn sẽ lưu trữ bí danh này ở nơi nào trên hệ thống?

3. Bạn có thể mã hoá các file sử dụng câu lệnh **uuencode**. File mã hoá sẽ được chuyển hướng đến stdout

Quản trị Hệ thống Linux - Cơ bản

Ví dụ: `uencode /bin/bash super-shell > ufile` sẽ mã hoá **/bin/bash** và tạo ra một file **super-shell** khi thực hiện **udecode** đối với *ufile*

- Gửi `/bin/bash` được mã hoá đến người dùng cục bộ (trong trường hợp này chúng ta có thể sử dụng **uencode** và đường ống | hoặc lưu lại kết quả mã hoá vào file *ufile* và sử dụng đối hướng STDIN <)
- Chia file mã hoá thành 5 file nhỏ

```
uencode /bin/bash super-shell > ufile
split -b 150000 ufile base-name.
```

Lệnh này sẽ tạo ra các file `base-name.aa`, `base-name.ab`,...

Thực hiện lệnh sau để ghép nối các file mã hoá đã được chia nhỏ thành file dữ liệu ban đầu (`unsplit`)

```
cat base-name.* > ufile.new
```

Cuối cùng giải mã file đó và kiểm tra xem nó có hoạt động không

```
udecode ufile.new
```

Câu lệnh này sẽ tạo ra một file nhị phân gọi là **super-shell**

4. Tiện ích nào sẽ tìm ra đường dẫn tuyệt đối của một file nhị phân thông qua quá trình kiểm tra biến `PATH`?

Các biến

1. Thực hiện các câu lệnh sau

Khởi tạo giá trị 'virus' cho biến `ALERT`

```
ALERT=virus
```

Kiểm chứng xem biến `ALERT` đã được khởi tạo chưa bằng lệnh **set**?

```
set |grep ALERT
```

Hoặc có thể hiển thị `ALERT` bằng cách dùng lệnh **env**

Tiếp theo, gõ 'bash'. Bạn có thể truy cập vào biến `ALERT`?

```
bash
echo $ALERT
```

Xem giá trị của `ALERT` là trống hay không?

Gõ `exit` (hoặc `^D`) để trở về phiên làm việc của bạn.

Sử dụng lệnh **export** để tạo biến `ALERT` là biến toàn cục

```
export ALERT
```

Quản trị Hệ thống Linux - Cơ bản

Kiểm chứng ALERT đã là biến toàn cục chưa?

```
env | grep ALERT
```

Khởi tạo một **bash** shell mới và chắc chắn rằng ALERT được định nghĩa trong shell mới

```
bash
```

```
echo $ALERT
```

Trong shell mới này, định nghĩa lại biến ALERT

```
export ALERT=green
```

Thoát khỏi shell này. Giá trị của biến ALERT trong shell ban đầu sẽ là bao nhiêu?

2. Tại lời nhắc câu lệnh gõ các dòng sau:

```
CREDIT01=300;CREDIT02=400
```

```
for VAR in CREDIT01 CREDIT02; do echo $VAR;done
```

Chú ý rằng biến VAR sẽ được tham chiếu bằng \$VAR

Thực hiện lại lệnh này

Thực hiện lệnh này nhưng thay thế CREDIT01 bằng \$CREDIT01

3. Sử dụng các dấu ngoặc thích hợp để thay đổi biến PS1 sao cho nó gồm đường dẫn tuyệt đối đến thư mục bạn đang làm việc

(Gợi ý: giá trị của PS1 là [\u@\W]\\$, do đó bạn chỉ cần thay thế \W bằng \w)

```
PS1=' [ \u@\h \w ]\$ '
```

Biến PS2 có giá trị như thế nào?

QUẢN LÝ FILE

Di chuyển quanh hệ thống file

Các đường dẫn tuyệt đối và tương đối

Một thư mục hoặc một file có thể truy cập bằng đường dẫn tuyệt đối bắt đầu từ thư mục gốc (/) hoặc đường dẫn tương đối bắt đầu từ thư mục hiện thời.

Đường dẫn tuyệt đối: độc lập với thư mục hiện thời của người dùng và bắt đầu với /

Đường dẫn tương đối: phụ thuộc vào thư mục hiện thời của người dùng và không bắt đầu với /

Đối với một hệ thống file có cấu trúc bất kỳ, có một số tiện ích giúp chúng ta có thể duyệt toàn bộ hệ thống

pwd: đưa ra đường dẫn tuyệt đối về vị trí của bạn trong hệ thống

cd: thay đổi thư mục

Tìm kiếm file và thư mục

Chúng ta sẽ tìm hiểu các tiện ích **find**, **which**, **whereis** và **locate**

- **find**

Cú pháp:

```
find <DIRECTORY> <CRITERIA> [-exec <COMMAND> {} \; ]
```

Tham biến *DIRECTORY* sẽ cho biết vị trí bắt đầu tìm kiếm và *CRITERIA* có thể là tên một file hoặc một thư mục mà chúng ta đang tìm kiếm

Ví dụ



```
find /usr/X11R6/bin -name "x*"  
find / -user 502
```

Quản trị Hệ thống Linux - Cơ bản

Các dòng đối sánh sẽ được hiển thị ở đầu ra chuẩn. Kết quả này có thể được sử dụng để thực hiện các lệnh tiếp theo. Ví dụ xóa file hoặc thay đổi quyền hạn... Tiện ích **find** có lựa chọn **-exec** cho phép chúng ta thực hiện điều đó. Ví dụ xóa tất cả các file thuộc về người dùng 502



```
find / -type f -user 502 -exec rm -f {} \;
```

- **xargs**

Tiện ích này thường xem như là một công cụ đi kèm với **find**. Thực tế **xargs** sẽ xử lý mỗi dòng của kết quả xuất chuẩn như một tham biến cho một tiện ích khác. Chúng ta có thể dùng **xargs** để xóa tất cả các file thuộc về một người dùng bằng lệnh sau



```
find / -type f -user 502 | xargs rm -f
```

Các câu lệnh chắc chắn như **rm** không thể xử lý với quá nhiều tham số. Chúng ta có thể xóa toàn bộ các file trong một thư mục với lệnh sau

```
ls | xargs rm -f
```

Common criteria switches for find	
-type	specify the type of file
-name	name of the file
-user	user owner
-atime, ctime, mtime	access, creation and modified times (multiples of 24 hrs)
-amin, cmin, mmin	access, creation and modified times (multiples of 1 min)
-newer FILE	files newer than <i>FILE</i>

- **locate**

Cú pháp:

```
locate <STRING>
```

Tiện ích **locate** cho phép hiển thị tất cả các file và thư mục thoả mãn biểu thức (expression)



locate X11R

Với tiện ích này quá trình tìm kiếm sẽ nhanh hơn rất nhiều. Thực tế **locate** sẽ truy vấn cơ sở dữ liệu **/var/lib/slocate**. Cơ sở dữ liệu này sẽ được cập nhật hàng ngày thông qua cron job dựa trên lệnh **updatedb**

Khi thực hiện **updatedb** từ dòng lệnh thì file **/etc/updatedb.conf** sẽ được đọc để xác định hệ thống file đã được chỉnh sửa (tức là NFS) và các thư mục (tức là **/tmp**)

- **which**

Cú pháp:

```
which string
```

Tiện ích này sẽ đưa ra đường dẫn tuyệt đối đối với file gọi là **string** bằng cách chỉ kiểm tra các thư mục được định nghĩa trong biến **PATH** của người dùng. Vì thế **which** chỉ được dùng để tìm kiếm các lệnh.

- **whereis**

Cú pháp

```
whereis string
```

Tiện ích này sẽ đưa ra đường dẫn tuyệt đối đối với các file nguồn, nhị phân, và tài liệu phù hợp với **string** bằng cách kiểm tra biến **PATH** cũng như các vị trí hay được sử dụng.

Các lựa chọn thường được dùng của **ls**

<i>Most common options for ls</i>	
-l	show inode
-h	print human readable sizes
-n	list UIDs and GIDs
-p	append descriptor (/= @) to list
-R	recursively display content of directories
-S	sort by file size
-t	sort by modification time (similar to -c)
-u	show last access time

Làm việc với thư mục

Tạo thư mục với lệnh mkdir

Khi tạo một thư mục chúng ta có thể thiết lập quyền truy nhập với lựa chọn **-m**. Một lựa chọn có ích khác đó là **-p** sẽ tự động tạo tất cả các thư mục con khi cần.

Ví dụ:



```
mkdir -p docs/programs/versions
```

Xoá các thư mục

Để xoá một thư mục chúng ta có thể sử dụng lệnh **rmdir** hoặc **rm**. Nếu bạn đang ở thư mục gốc bạn có thể dùng lựa chọn **-f** để xoá tất cả các file.

Chú ý: `rm -rf /dir1/*` xoá tất cả các file và các thư mục con và để dir1 là thư mục trống

`rm -rf /dir1/` xoá tất cả các file và các thư mục con bao gồm cả dir1

Sử dụng cp và mv

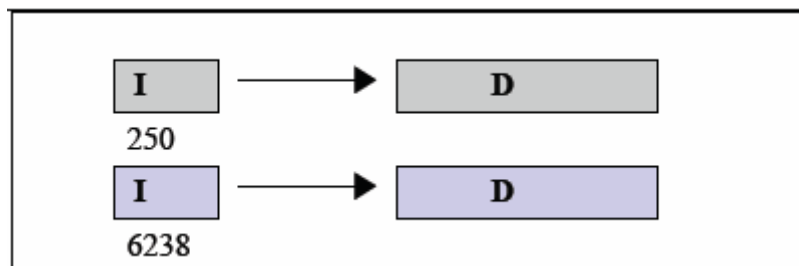
cp

Cú pháp

```
cp [options] file1 file2
cp [options] file1 directory
```

Chú ý rằng **cp file1 file2** tạo một bản copy mới của *file1* và không làm thay đổi *file1*

Hình minh họa: file1 với inode 250 sẽ được copy sang file 2, sao y dữ liệu đến một vùng dữ liệu mới và tạo inode mới 6238 cho file2.



Chúng ta cũng có thể copy một số file đến một thư mục khác bằng cách dùng danh sách liệt kê hoặc ký tự đại diện. Bảng sau đây sẽ hiển thị các lựa chọn thường được sử dụng

<i>Most common options for cp</i>	
-d	do not follow symbolic link (when used with -R)
-f	force
-i	interactive, prompt before overwrite
-p	preserve file attributes
-R	recursively copy directories

Chú ý: `cp -r /mydir/* /dir2/` sẽ copy tất cả các file và thư mục con ngoại trừ *mydir*

`cp -r /mydir/ /dir2/` sẽ copy tất cả các file và thư mục con bao gồm cả *mydir*

mv

Cú pháp:

```
mv [options] oldname newname
mv [options] source destination
mv [options] source directory
```

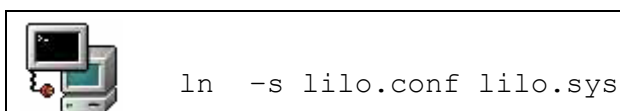
Lệnh **mv** có thể di chuyển hoặc đổi tên các file và thư mục. Nếu *oldname* là một file và *newname* là một thư mục thì file *oldname* sẽ được di chuyển đến thư mục này.

Nếu source và destination cùng nằm trên một hệ thống file thì file không được copy nhưng thông tin về inode sẽ được cập nhật để xác định vị trí mới. Các lựa chọn thông thường là -f để ghi đè và -i truy vấn tương tác.

Hard links và symbol links

Các liên kết tượng trưng

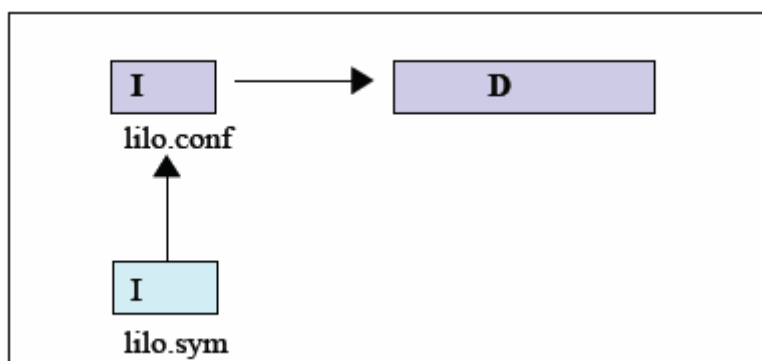
Một liên kết tắt mềm đến một file hoặc một thư mục tạo một inode mới chỉ đến cùng một vùng dữ liệu.



Đây là danh sách những file của câu lệnh trên. Chú ý rằng giá trị tham chiếu là 1 cho cả 2 file.

```
-rw----- 1 root root 223 Nov 9 09:06 lilo.conf  
lrwxrwxrwx 1 root root 9 Nov 9 09:06 lilo.sym -> lilo.conf
```

Hình 2: Một liên kết tắt mềm đến một file



Các liên kết tắt mềm có thể được tạo thông qua các hệ thống file khác nhau

Các liên kết tắt cứng

Một liên kết tắt cứng là một tên được tạo thêm cho cùng một inode và giá trị tham chiếu của file đó sẽ được tăng thêm 1 cho mọi liên kết tắt cứng mới



```
ln lilo.conf lilo.link
```

Trong bảng dưới đây chú ý rằng giá trị tham chiếu là 2 và cả 2 file có cùng kích thước. Thực tế chúng hoàn toàn giống nhau.

```
-rw----- 2 root root 223 Nov 9 09:06 lilo.conf  
-rw----- 2 root root 223 Nov 9 09:06 lilo.link
```

Các liên kết tắt cứng chỉ có thể được tạo trong cùng một hệ thống file.

Touching và dd-ing

touch

Một cách khác để tạo hoặc thay đổi một file là sử dụng touch

Cú pháp:

```
touch {options} file(s)
```

Nếu *file* chưa có thì nó sẽ được tạo mới. Chúng ta có thể thay đổi thời gian truy cập file bằng lựa chọn -a, -m thay đổi thời gian sửa đổi file, và -r dùng để sử dụng các thuộc tính thời gian của file khác.

Ví dụ

```
touch file1.txt file2.txt          tạo các file mới
```

```
touch myfile -r /etc/lilo.conf     myfile sẽ lấy các thuộc tính thời  
gian của lilo.conf
```

Tạo một file **-errors** sử dụng lựa chọn -:

Quản trị Hệ thống Linux - Cơ bản

```
touch -- -errors
```

dd

Lệnh này sẽ copy 1 file với kích thước khối I/O có thể thay đổi. Lệnh này cũng được dùng để thực hiện các quá trình chuyển đổi (giống như **tr**). Các lựa chọn chính là **if**= (file nhập), **of**=(file xuất), và **conv**=(chuyển đổi)

Các khoá chuyển đổi có thể là: **lcase**, **ucase**, và **ascii**

Ví dụ

```
dd if=/mnt/cdrom/images/boot.img of=/dev/fd0
```

Bài tập

Điều hướng file

Tạo một thư mục mới /bin trong /tmp

```
mkdir /tmp/bin
```

Trong /tmp/bin tạo một file gọi là newfile (sử dụng touch, cat, hoặc vi)

Chuyển đến thư mục gốc (cd /). Xem nội dung của *newfile* từ vị trí này.

Câu lệnh ngắn nhất nào giúp bạn quay trở về **/tmp/bin**?

Câu lệnh ngắn nhất nào giúp bạn chuyển đến thư mục home của bạn?

Biến PWD là cục bộ hay toàn cục?

Tạo và xoá các thư mục

Cách nào là nhanh nhất để tạo các thư mục /dir1/dir2?

Xoá các thư mục này với rmdir sau đó với rm

Tạo khoảng trống trên hệ thống file

Để tạo thêm khoảng trống trên thiết bị chứa thư mục **/usr/share/doc** chúng ta cần tìm thiết bị dự phòng có đủ khoảng trống để copy nội dung của **/usr/share/doc** vào thiết bị này. Sau đó chúng ta sẽ xoá thư mục **/usr/share/doc** và tạo một điểm liên kết tượng trưng từ **/usr/share/doc** đến vị trí mới.

Tạo một thư mục **/spare** trên đó chúng ta sẽ gắn (mount) các thiết bị dự phòng phù hợp (một trong những phân vùng được tạo từ các bài tập trước sẽ phù hợp cho mục đích này)

```
mkdir /spare
mount <device> /spare
```

Kiểm tra với lệnh **df -h /spare** và **du -hs /usr/share/doc** xem thiết bị này có dung lượng đủ lớn để chứa tất cả dữ liệu đang có.

Tiếp theo, copy các nội dung của **/usr/share/doc** đến **/spare/**

```
cp -a /usr/share/doc /spare
```

Sau khi chắc chắn dữ liệu đã được copy hết thì thay đổi **/etc/fstab** có thể sử dụng ngay sau khởi động.

Xoá **/usr/share/doc** và tạo điểm liên kết tượng trưng từ **/usr/share/doc** đến **/spare/doc**

```
ln -s /spare/doc /usr/share/doc
```

Thực hiện tương tự với **/home**. Xem có vấn đề gì xảy ra?

Tìm kiếm các file trên hệ thống

Copy file **/etc/lilo.conf** đến **/etc/lilo.conf.bak**

1 Dùng lệnh **find** để tìm find mới

2 Dùng **locate** để tìm **/etc/lilo.conf.bak** (Bạn sẽ cập nhật cơ sở dữ liệu **slocate** như thế nào?)

Các sao lưu dự phòng (bước đầu tiên)

Tìm tất cả các file đã được thay đổi trong ngày hôm nay trong thư mục home của bạn.

```
find /home -mtime -1 |tee list1 |wc --lines (-1 có nghĩa là ít hơn 1 ngày)
```

Chúng ta sẽ giới thiệu các tiện ích lưu trữ ở phần sau, tuy nhiên kết quả xuất của các lệnh tìm kiếm sẽ được dẫn trực tiếp vào **cpio**.

QUẢN LÝ TIỀN TRÌNH

Xem các tiến trình đang chạy

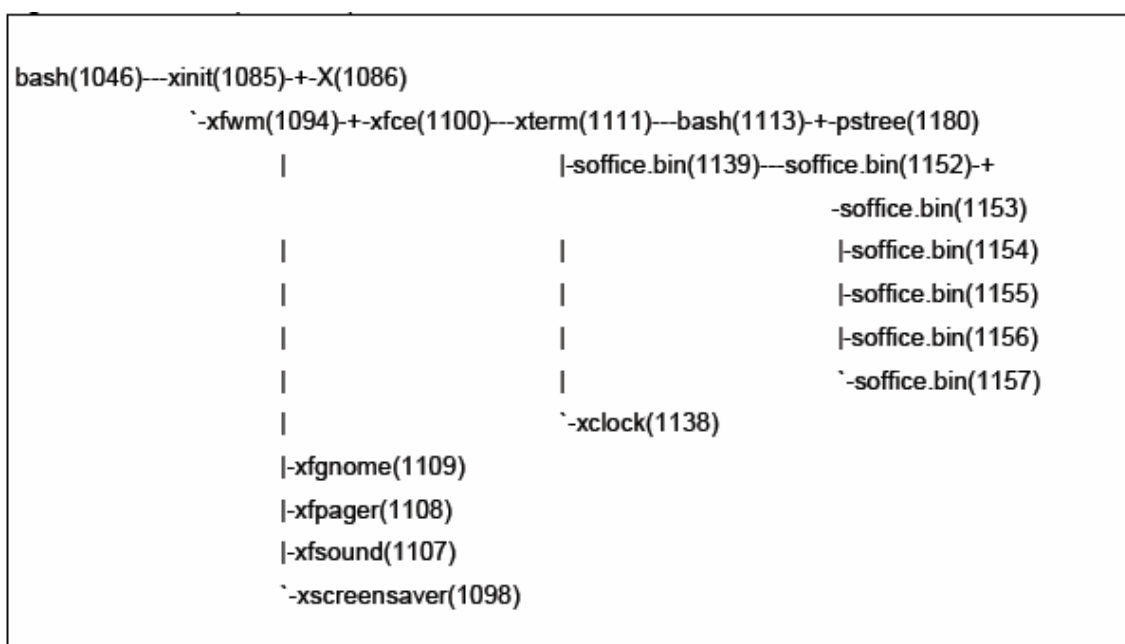
Các tiến trình có một mã (ID) tiến trình duy nhất đó là PID. Giá trị của PID có thể dùng để thay đổi sự ưu tiên của tiến trình hoặc dừng hẳn tiến trình đó.

Một tiến trình là bất cứ chương trình nào đang thực hiện. Nếu process_2 được sinh ra bởi process_1 thì nó được gọi là tiến trình con. Còn process_1 thì được gọi là tiến trình cha.

Cây gia hệ của các tiến trình

Lệnh **ps tree** sẽ đưa ra một minh họa đầy đủ của hệ phân cấp các tiến trình cha và con.

Hình 1: Một phần các kết quả của ps tree



Trong hình trên tất cả các mã tiến trình (PID) đều được nhìn thấy; giá trị của chúng được tăng dần. Lựa chọn thông dụng nhất của lệnh này là -p sẽ hiển thị các PID và -h sẽ làm nổi rõ các tiến trình của người dùng.

Tìm kiếm các tiến trình đang thực hiện

Sử dụng lệnh **ps** là một cách trực tiếp để xác định tiến trình nào đang thực hiện. Phần lớn người dùng kết hợp một số các lựa chọn để phù hợp với mục đích tìm kiếm. Dưới đây là 3 lựa chọn như vậy.

- ps -ux** hiển thị tất cả các tiến trình thực hiện bởi người dùng
- ps T** hiển thị các tiến trình đang chạy bởi thiết bị đầu cuối hiện thời của người dùng
- ps aux** hiển thị tất cả các tiến trình trên hệ thống

Để biết chi tiết hơn các lựa chọn chúng ta nên sử dụng lệnh **ps** manpage và chọn ra những lựa chọn phù hợp nhất.

<i>ps accommodates UNIX-style and BSD-style arguments</i>
usage: ps -[Unix98 options] ps [BSD-style options] ps -[GNU-style long options] ps -help for a command summary

<i>Summary of options</i>
-a show all processes for the current user linked to a tty (<i>except</i> the session leader) -e or -A show all processes -f gives the PPID (Parent Process ID) and the STIME (Start Time) -l is similar to -f and displays a long list a show all processes linked to a tty, including other users x show all processes without a controlling tty as well

Cập nhật liên tục thông tin tiến trình

Tiện ích **top** sẽ cập nhật thông tin trên các tiến trình tại một mức điều chỉnh.

Trong khi tiện ích **top** đang thực hiện chúng ta có thể gõ h đối với một danh sách các lệnh. Khoảng trống sẽ được cập nhật thông tin tức thời. Chúng ta cũng có thể dùng **top** để thay đổi mức độ ưu tiên của một tiến trình.

Thay đổi tiến trình

Dừng các tiến trình

Lệnh **kill** sẽ gửi các tín hiệu đến các tiến trình. Có tổng cộng 63 tín hiệu. Tín hiệu mặc định dừng một tiến trình được gọi là SIGTERM với giá trị 15.

kill

cú pháp

kill SIGNAL process_PID

Mọi tiến trình có thể lựa chọn nhận hay không nhận một tín hiệu ngoại trừ SIGKILL sẽ được thực hiện bằng nhân hệ thống. Các daemon sẽ hiểu SIGUP có nghĩa là "đọc lại file cấu hình"

Most Common Signals



1 or SIGHUP hangup or disconnect the process

2 or SIGINT same as Ctrl+C interrupt

3 or SIGQUIT quit

9 or SIGKILL kill the process through a kernel call

15 or SIGTERM terminate a process 'nicely'. This is the DEFAULT signal.

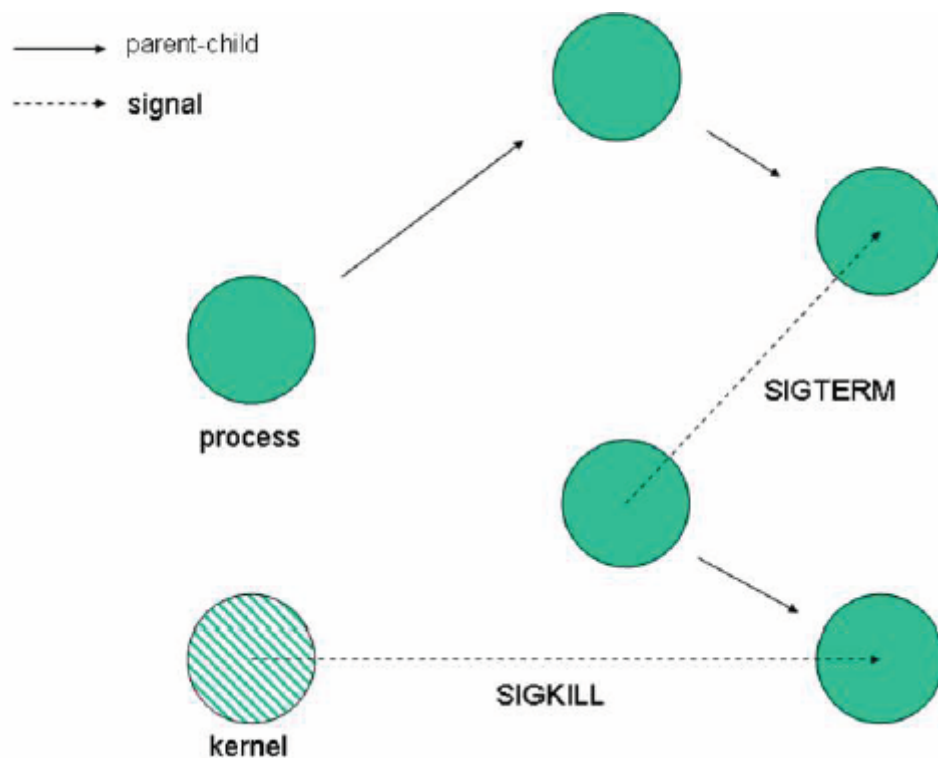
Chúng ta có thể sử dụng lệnh **killall** để dừng các tiến trình mà không cần biết PID

killall

Cú pháp

killall SIGNAL process_NAME

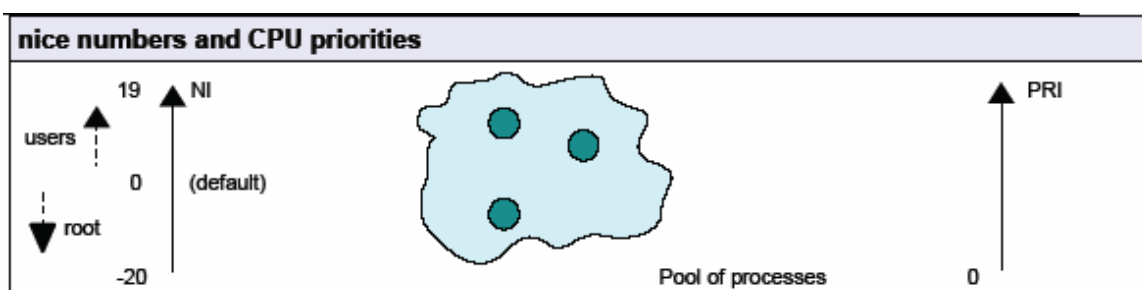
Hình 1: Tín hiệu giữa các tiến trình



Ưu tiên tiến trình và các giá trị (nice value)

Các giá trị nice value (NI) thay đổi quyền ưu tiên của CPU và được dùng để cân bằng quá trình sử dụng CPU trong môi trường đa người dùng. Mỗi tiến trình bắt đầu với giá trị NI mặc định là 0. Các NI nằm trong phạm vi từ 19 [thấp nhất] đến -20[cao nhất]

Chỉ có người quản trị hệ thống có thể giảm giá trị NI của một tiến trình. Từ khi tất cả các tiến trình bắt đầu với giá trị NI mặc định là 0, chỉ có người quản trị hệ thống có thể thiết lập giá trị âm cho các giá trị NI.



Sử dụng lệnh **renice** để thay đổi mức độ ưu tiên của một tiến trình. Dùng lệnh **nice** để thiết lập mức độ ưu tiên của một tiến trình.

Cú pháp

```
nice -<NI> <process>
```

```
renice <+/-NI> -p <PID>
```

Chú ý rằng **renice** thực hiện với các PID và xử lý danh sách các tiến trình tại một thời điểm. Một lựa chọn có ích của **renice** là **-u**, lựa chọn này sẽ ảnh hưởng đến tất cả các tiến trình thực hiện bởi người dùng.

Thiết lập giá trị 1 cho các tiến trình 234 và 765



```
renice +1 -p 234 765
```

Thiết lập giá trị -5 cho xclock



```
nide --5 xclock
```

Tiến trình và Shell

Các tiến trình nền sau và nền trước

Sau khi chúng ta bắt đầu một tiến trình từ shell, chúng ta sẽ để tiến trình đó cho shell tự động thông dịch. Chúng ta chú ý rằng sẽ không có lệnh nào đáp ứng nữa. Lý do cho vấn đề này đó là chỉ có thể thực hiện các chương trình trong nền trước **fg** hoặc nền sau **bg** của shell.

Khi một chương trình đang chạy trong chế độ nền trước, dấu nhắc shell có thể khôi phục bằng cách ngắt chương trình đó. Tín hiệu ngắt được sinh ra bởi tổ hợp phím **Ctrl Z**.

Dừng và bắt đầu các công việc (job)

Một tiến trình bắt đầu từ shell còn được gọi là một công việc. Khi một công việc nhận tín hiệu ^Z, nó sẽ được dừng và dấu nhắc shell sẽ xuất hiện. Để khởi tạo lại chương trình trong chế độ nền sau chúng ta chỉ cần gõ: bg

Ví dụ

[mike localhost /bin]\$xclock	xclock chạy trong chế độ nền trước, dấu nhắc shell biên mất
[1]+ Stoppep xclock	xclock nhận tín hiệu ^Z
[mike localhost /bin]\$bg	dấu nhắc shell được khôi phục và đưa vào lệnh bg
[1]+ xclock &	xclock đang chạy trong chế độ nền sau
[mike localhost /bin]\$	

Chú ý ký hiệu [1]+ ở trên. Giá trị này là job number của tiến trình. Trong đó dấu hiệu '+' chỉ ra tiến trình được thay đổi lần gần nhất. Dấu hiệu '-' chỉ ra tiến trình được thay đổi lần liền kề

Hiển thị các công việc

Tiện ích jobs hiển thị tất cả các tiến trình đang chạy bắt đầu từ shell hiện thời. Giá trị job number, trạng thái công việc (đang chạy hay dừng), và 2 tiến trình được thay đổi gần nhất sẽ được hiển thị

Output for jobs	
[1]- Stopped	xclock
[2] Running	xman &
[3]+ Stopped	xload

Job number

Quản trị Hệ thống Linux - Cơ bản

Chúng ta có thể dừng và bắt đầu lựa chọn các công việc một cách thuận tiện bằng cách sử dụng job number. Việc lựa chọn này được thực hiện cùng với lệnh fg

Gọi job 2 ở nền trước và loại bỏ (kill) job 1

```
fg 2 hoặc kill -9 %1
fg %2 hoặc
fg %?xma
```

Tránh sử dụng HUP với nohup

nohup là một chương trình có vai trò như một tiến trình cha độc lập với phiên người dùng. Khi một người dùng thoát khỏi hệ thống, thì hệ thống sẽ gửi HUP đến tất cả các tiến trình nằm trong nhóm tiến trình của người dùng. Ví dụ, để tránh tín hiệu HUP, một chương trình gọi là bigbang sẽ cố gắng tính thời gian xuất hiện của các tiến trình



Bài tập

Bạn nên chạy X trước khi bắt đầu các bài thực hành sau

1. Kiểm tra giá trị nice value (NI) hiện thời của x-terminal đang chạy. Thay đổi giá trị này bằng lệnh **top** hoặc **renice**
2. Tín hiệu tương đương của **^Z** gửi đến một tiến trình là gì? (Hiển thị tất cả các tín hiệu với **kill -l**)
3. Tín hiệu nào được định nghĩa lại cho phần lớn các daemon và yêu cầu đọc lại file cấu hình?
4. Tín hiệu mặc định gửi đến một tiến trình là gì khi sử dụng **kill** hoặc **killall**?
5. Tín hiệu nào được trực tiếp xử lý bằng nhân hệ thống (kernel) và không thể định nghĩa lại?

6. Trước hết bạn hãy đăng nhập vào thiết bị đầu cuối ảo (tty1 to tty6). Chúng ta sẽ thực hiện một script cho phép tiếp tục chạy khi chúng ta thoát ra khỏi hệ thống dùng tiến trình cha **nohup**

Trong thư mục **/tmp** tạo một file gọi là **print-out** với nội dung sau đây

```
#!/bin/bash
count=0
while (true) do
    echo this is iteration number $count
    let count+=1
done
```

Chúng ta trước hết thực hiện các bước sau (không dùng **nohup**)

```
cd /tmp
./print-out &
exit
```

Chúng ta có thể không nhìn thấy dòng lệnh khi gõ **exit** nhưng câu lệnh này sẽ làm bạn thoát ra khỏi hệ thống. Khi bạn đăng nhập lại hãy kiểm tra **print-out** đã được dừng

```
ps ux | grep print-out
```

Tiếp theo bắt đầu với lệnh

```
nohup /tmp/print-out &
exit
```

Đăng nhập lại và kiểm tra những lệnh sau


```
ps ux |grep print-out
tail -f ~/nohup.out
Ctrl+C
killall print-out
ps ux|grep print-out
tail -f ~/nohup.out
```

XỬ LÝ VĂN BẢN

cat the Swiss Army Knife

Dùng cat để soạn văn bản

Tiện ích cat có thể dùng như một chương trình soạn thảo đơn giản



```
cat > short-message  
we are curious  
to meet  
penguins in Prague  
Ctrl+D
```


Chú ý cách dùng Ctrl+D. Lệnh này được dùng để kết thúc nhập input.

Dùng cat để đọc văn bản

Thông thường hơn **cat** được dùng để đưa văn bản ra *stdout*. Các lựa chọn thường được dùng là

- n đánh số mỗi dòng của output
- b chỉ đánh số dòng output không trống
- A hiển thị ký hiệu xuống dòng


Ví dụ



```
cat /etc/resolv.conf  
search     mydomain.org  
nameserver 127.0.0.1
```

Lệnh tac sẽ đọc văn bản từ cuối lên đầu

Lệnh này giống như cat ngoại trừ nội dung văn bản được đọc từ dòng cuối lên đầu



```
tac short-message
penguins in Prague
to meet
we are curious
```

Các công cụ đơn giản

Sử dụng head hoặc tail


Các tiện ích head hoặc tail thường được dùng để phân tích các logfile. Chúng sẽ xuất đưa ra mặc định 10 dòng văn bản. Sau đây là cách dùng

Hiển thị 20 dòng đầu tiên của **/var/log/messages**:



```
head -n 20 /var/log/messages
head -20 /var/log/messages
```

Hiển thị 20 dòng cuối cùng của **/etc/aliases**:



```
tail -20 /etc/aliases
```

Tiện ích tail có thêm một lựa chọn cho phép hiển thị nội dung văn bản bắt đầu từ dòng đưa vào cho đến hết.

Hiển thị nội dung văn bản bắt đầu từ dòng 25 trong **/var/log/messages**:



```
tail +25 /etc/log/messages
```

Câu hỏi: nếu một văn bản có 90 dòng, chúng ta sẽ sử dụng lệnh **tail** và **head** như thế nào để hiển thị các dòng từ 50 tới 65? Có thể có nhiều hơn một cách để thực hiện điều này?

Cuối cùng tail có thể đọc liên tục một file bằng lựa chọn -f. Lựa chọn này rất có ích khi chúng ta mong muốn một file được thay đổi trong thời gian thực

Đếm số dòng, số từ và byte

Tiện ích **wc** sẽ đếm số lượng các byte, các từ, và các dòng trong file. Một vài lựa chọn cho phép chúng ta thay đổi giá trị output của **wc**

Các lựa chọn cho wc

-l	Đếm số dòng
-w	Đếm số các ký tự hoặc từ
-c hoặc -m	Đếm số các byte hoặc ký tự


Lưu ý:

Nếu không có tham biến, **wc** sẽ đếm dựa trên nội dung được gõ vào *stdin*

Đánh số các dòng


Tiện ích nl có tác dụng giống như cat -b

Đánh số tất cả các dòng gồm cả các dòng trống



```
nl -ba /etc/lilo.conf
```

Đánh số các dòng văn bản không trống



```
nl -bt /etc/lilo.conf
```

Thay thế tab bằng space

Lệnh **expand** cho phép thay thế TAB bằng các dấu cách (space). Chúng ta có thể dùng lệnh **unexpand** để thay thế ngược lại.

Xem các file nhị phân

Có một số công cụ để thực hiện điều này. Công cụ phổ biến nhất là **od** (octal dump) và **hexdump**.

Xử lý văn bản

Các công cụ sau đây thay đổi bố trí văn bản

Lựa chọn các trường (field) và các ký tự với cut


Tiện ích cut có thể lấy ra một vùng các ký tự hoặc các trường từ mỗi dòng của văn bản.

Lựa chọn -c được dùng để xử lý ký tự.

Cú pháp:

```
cut -c {range1,range2}
```

Ví dụ:



```
cut -c5-10,15- /etc/passwd
```


Ví dụ trên sẽ đưa ra các ký tự từ vị trí 5 đến 10 và từ 15 đến cuối dòng của mỗi dòng trong /etc/passwd

Chúng ta có thể xác định dấu phân cách các trường (dấu cách, dấu phẩy,...) của một file cũng như các trường đối với output. Các lựa chọn được thiết lập với cờ hiệu -d và -f tương ứng

Cú pháp:

```
cut -d {delimiter} -f {fields}
```

Ví dụ:



```
cut -d: -f 1,7 --output-delimiter=" " /etc/passwd
```

Ví dụ này sẽ đưa ra các trường từ trường đầu tiên đến trường thứ bảy của `/etc/passwd` được phân cách bằng dấu cách. Mặc định của *output-delimiter* là giống như các phân cách input ban đầu. Lựa chọn **--output-delimiter** cho phép chúng ta thay đổi giá trị mặc định này.

Kết nối và dán văn bản

Tiện ích đơn giản nhất là **paste** sẽ ghép hai file bên cạnh nhau

Cú pháp

```
paste text1 text2
```

Với tiện ích **join** chúng ta có thể xác định cụ thể hơn những trường mà chúng ta đang quan tâm

Cú pháp

```
join -j1 {field_num} -j2{field_num} text1 text2 or  
join -1 {field num} -2{field num} text1 text2
```

Văn bản được gửi đến stdout chỉ khi những trường cụ thể đối sánh. Quá trình so sánh được thực hiện trên một dòng tại mỗi thời điểm cho đến khi có sự trùng nhau được tìm thấy và quá trình sẽ được dừng ngay lập tức mặc dù có thể có nhưng sự trùng nhau nằm ở phía sau.

Sắp xếp output

Quản trị Hệ thống Linux - Cơ bản

Lệnh `sort` sẽ sắp xếp mặc định một văn bản theo thứ tự abc. Lựa chọn `-n` sẽ thực hiện việc sắp xếp theo thứ tự số.

Định dạng output

Chúng ta có thể thay đổi số lượng các ký tự trong mỗi dòng của output bằng lệnh `fmt`. Mặc định `fmt` sẽ liên kết các dòng và đưa ra 75 ký tự cho mỗi dòng

Các lựa chọn ***fmt***

- w** số lượng các ký tự trên mỗi dòng
- s** tách những dòng dài nhưng không điền đầy dòng còn lại
- u** đặt một dấu cách ở giữa mỗi từ và 2 dấu cách ở cuối mỗi câu

Thay thế các ký tự

Tiện ích **tr** sẽ thay thế một tập hợp các ký tự bằng tập hợp ký tự khác.

Ví dụ thay đổi các chữ cái viết hoa bằng chữ cái thường

```
tr '[AB]' '[ab]' <file.txt
```

Thay thế các dấu phân cách trong `/etc/passwd`:



```
tr ':' ' ' < /etc/passwd
```

Chú ý: **tr** chỉ có 2 tham biến! Và file không được tính là tham biến.

Bài tập

1. Sử dụng **cat** để gõ văn bản sau vào một file có tên là *message*

```
cat >> message  
line 1
```

Quản trị Hệ thống Linux - Cơ bản

^D

Thực hiện tương tự nhưng dùng từ khoá **STOP** thay thế điều khiển kết thúc file (^D)

```
cat >> message << STOP
line 2
STOP
```

Tiếp theo, thêm văn bản sau vào *message* sử dụng **echo**

```
echo line 3 >> message
```

2. Tạo một file có tên là *index* với 2 trường *REFERENCE* và *TITLE* được phân cách bằng dấu cách

```
ví dụ 001 Using_Linux
```

Tạo file thứ hai có tên là *pricing* với 2 trường *REFERENCE* và *PRICE* được phân cách bằng dấu cách

```
ví dụ 001 9.99
```

Sử dụng lệnh **join** để hiển thị các trường *reference*, *title*, và *price*.

3. Sử dụng **tr** để thay thế toàn bộ dấu hai chấm bằng dấu chấm phẩy trong */etc/passwd*

Thực hiện tương tự dùng lệnh **cut**

4. Sử dụng lệnh **head** và **tail** để hiển thị dòng 75 đến 80 của */var/log/messages*

5. Sử dụng tiện ích **cut** cùng với **grep** và **ifconfig** để in ra duy nhất địa chỉ IP của giao diện mạng *eth0*

6. Trong **/tmp** tạo một thư mục với tên là files

```
mkdir /tmp/files
```

Tạo 50 file trong thư mục này

```
# ! /bin/bash
count=0
while [ $count -lt 50 ] do
touch /tmp/files/$count.txt
let count+=1
done
```

Các bạn sẽ gõ các dòng lệnh sau để thay đổi các file có phần mở rộng **txt** sang **dat**

```
for FILES in $(ls *.txt)
do
FILENAME=$(echo $FILES| cut -d. -f1)
mv $FILES $FILENAME.dat
done
```

CÀI ĐẶT PHẦN MỀM

Giới thiệu

Hãy bắt đầu cùng với một đoạn mã nguồn ngắn. Ví dụ này sẽ giúp giúp chúng ta tìm hiểu vấn đề mà không cần có kiến thức sâu về ngôn ngữ lập trình C

Tệp *main.c*:

```
#include<stdlib.h>

int main() {
    Hello();
}
```

Tệp *Hello.c*:

```
#include<stdio.h>

void Hello() {
    printf("Hi ! \n");
}
```

Chú ý: *main.c* là chưa hoàn thành nếu hàm *Hello()* là chưa được định nghĩa. Cũng như vậy đối với *Hello.c* nếu không có khai báo *main*. Do đó, các tệp này là phụ thuộc nhau. Tuy nhiên từng hàm riêng biệt vẫn có thể được dịch theo một đối tượng kiểu Object (.o) đây là kiểu files được sử dụng để xây dựng các ứng dụng.

Dịch các file đối tượng

```
gcc -c main.c
gcc -c Hello.c
```

Câu lệnh trên sẽ tạo ra 2 tệp *main.o* và *Hello.o* được sử dụng để xây dựng các ứng dụng **app**.

Dịch app

Lựa chọn `-o` xác định tên của mã nguồn đã được dịch. Nếu không có tên tệp nào được chỉ ra tệp dịch sẽ được đặt tên mặc định **a.out**

Tất cả các bước trên có thể được chạy tự động bằng cách sử dụng a Makefile. Dưới đây là một ví dụ nhỏ dùng Makefile để tạo ứng dụng **app**

Makefile

```
SHELL = /bin/sh
CC = /usr/bin/gcc
app: main.o Hello.o
    $(CC) -o app main.o Hello.o
main.o: main.c
    $(CC) -c main.c
Hello.o Hello.c
    $(CC) -c Hello.c
```

Thư viện tĩnh và thư viện chia sẻ

Các hàm chức năng thường dùng được lưu lại trong các thư viện. Trong thời gian dịch chương trình các thư viện này có thể được link tới mã nguồn nơi có sử dụng lệnh gọi thư viện chức năng. Thư viện có thể được link tới mã nguồn một cách tĩnh hoặc động.

Lệnh dịch gcc có thể link thư viện trong các cách khác nhau. Tuy nhiên theo chế độ mặc định nó sẽ là link các tệp(files) được khai báo trong dòng lệnh không có phần mở rộng .c (chỉ có các tệp có mở rộng .c là được hiểu như mã nguồn).

Listing 1. Kết nối mặc định (linking)



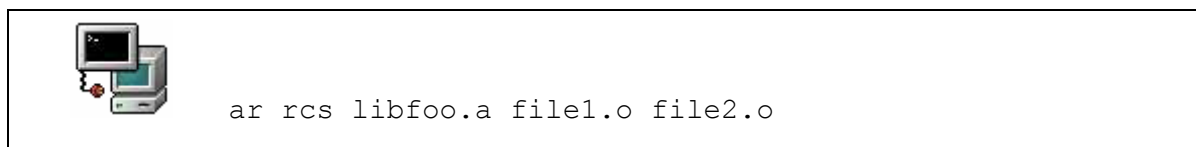
```
gcc main.c Hello.o
```

Dòng lệnh trên sẽ tạo tệp chạy a.out cùng tệp đối tượng Hello.o được link tĩnh tới nó.

- **Thư viện tĩnh**

Các thư viện tĩnh được lưu trữ trong file **.o**. Các lưu trữ được tạo ra bởi công cụ **ar** và có phần mở rộng **.a**.

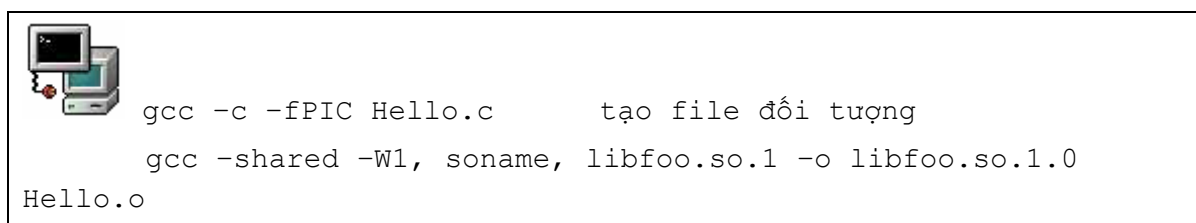
Hình 2: thêm một file đối tượng vào phần lưu trữ



- **Thư viện động / Thư viện chia sẻ**

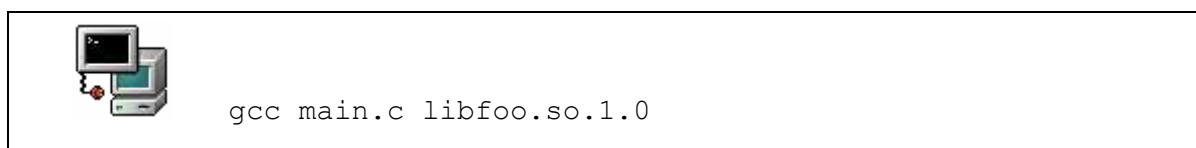
Thư viện chia sẻ là một thư viện sẽ được tải bởi chương trình khi nó được thực thi. Mặt khác chúng ta cũng có thể nói nó là *thư viện mà được tải động* (dynamically loaded)

Hình 3: Tạo thư viện chia sẻ:



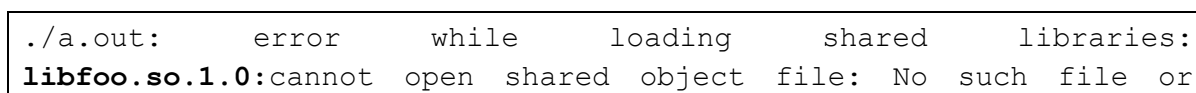
Cờ hiệu (flag) **-fPIC** sẽ kích hoạt vị trí mã nguồn độc lập

Hình 4: Dịch chương trình với thư viện chia sẻ:



Dòng lệnh trên sẽ tạo file chạy **a.out**. Tuy nhiên nếu bạn thử chạy file này máy tính sẽ thông báo lỗi dưới đây.

Thông báo lỗi không tìm thấy thư viện chia sẻ

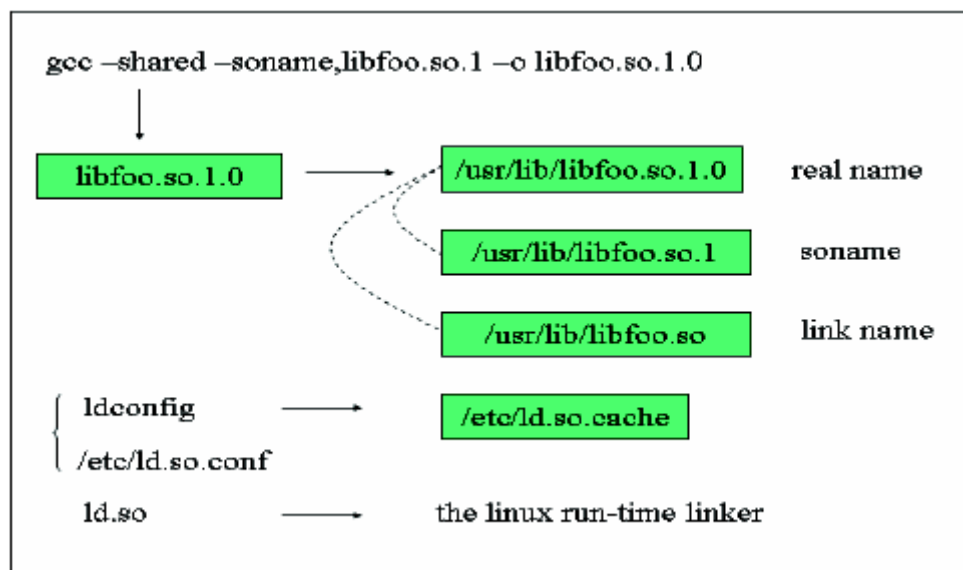


directory

Trong phần tiếp theo chúng ta sẽ tìm hiểu cách để sửa lỗi này.

- **Đặt tên Thư viện chia sẻ và tải dynamic**

Chúng ta sử dụng ví dụ trong phần trước để tìm hiểu Thư viện Linux được bảo trì(maintain) thế nào.



Hình 1: Tên thư viện chia sẻ

Sử dụng công cụ `ldd` để xem Thư viện chia sẻ nào một file chạy cần trong thời gian thực thi.

Ví dụ:



```
ldd a.out
libfoo.so.1.0 => not found
libc.so.6 => /lib/libc.so.6 (0x40028000)
/lib/ld-linux.so.2      =>      /lib/ld-linux.so.2
(0x40000000)
```

Chú ý chúng ta sẽ không tìm thấy file `libfoo.so.1.0` vì **a.out** cần tải(load) động thư viện này và kết nối động **ld.so** không biết có sự tồn tại của thư viện này.

Có thể làm theo một trong các cách sau để khắc phục lỗi này.

1. Nếu tệp nhị phân cần ở chế độ tạm thời, kiểm tra định nghĩa biến `LD_LIBRARY_PATH` như sau:



```
export LD_LIBRARY_PATH=$(pwd)
```

2. Copy `libfoo.so.1.0` vào thư mục `/usr/lib` và chạy `ldconfig` để nâng cấp ld cache

Cài đặt nguồn

Quản lý gói Redhat (Redhat Package Manager RPM)

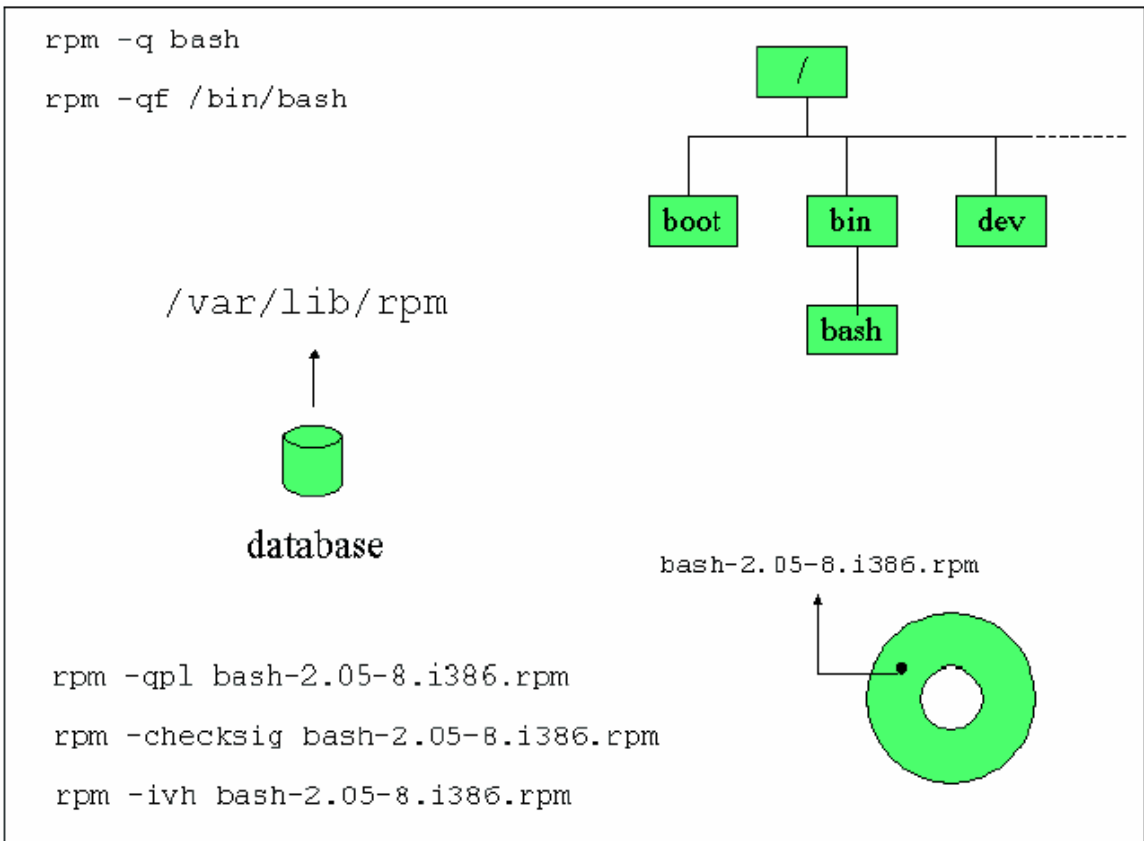


Figure1: Các chức năng của Quản lý Gói (Package Manager)

Đặt tên package

Rpm được đặt tên theo cách sau

name-version-release.architecture.rpm

Chế độ (mode) chính

Tắt	Đầy đủ	Mô tả
-i	- install	Cài đặt gói
-U	-update	Cập nhật hoặc cài đặt gói
-F	--freshen	Cập nhật chỉ những gói đã install
-V	--verify	Cỡ file, MD5, quyền, kiểu ...

-q	--query	Yêu cầu gọi các gói và các file đã cài đặt/đã gỡ bỏ
-e	--erase	Gỡ bỏ gói

Chế độ thứ cấp

Tắt	Mô tả
a	áp dụng cho tất cả các gói đã cài đặt
c	cùng với q đưa ra các file cấu hình
d	cùng với q đưa ra các file tài liệu
h	chạy bảng băm (hash) trong khi xử lý
i	cùng với q đưa ra thông tin về gói
l	cùng với q đưa ra tất cả file và thư mục trong một gói
p	cùng với q chỉ ra truy vấn nào được thực hiện đối với file
v	verbose

Các mode yêu cầu (query mode)

Chúng ta xem xét ví dụ với gói `routed-0.17.i386.rpm`. Bạn có thể truy vấn gói này và đưa ra nội dung của nó trước khi cài đặt cùng với lựa chọn như sau:



```
rpm -qpl routed-0.17.i386.rpm
```

Khi gói này được cài đặt bạn có thể truy vấn gói đã cài đặt như sau:



```
rpm -ql routed-0.17           or  
rpm -ql routed
```

Cuối cùng nếu chúng ta muốn tìm gói nào đã cài đặt file `/usr/sbin/routed` dữ liệu rpm có thể được yêu cầu cùng:



```
rpm -qf /usr/sbin/routed
```

Ba kiểu truy vấn (query): *uninstalled packages, installed packages và file*

Kiểu truy vấn	Tuỳ chọn
Package File	-qp
Installed Package	-q
File	-qf

Một tuỳ chọn mở rộng sẽ cho phép bạn lấy thông tin trong tất cả các files đã cài đặt **-l**, tài liệu đã cài đặt **-d**, file cấu hình **-c**, v.v...

Các Tuỳ chọn đặc biệt

- nodeps** cho phép cài đặt không phụ thuộc
- force** ép buộc nâng cấp, cài đặt
- test** không cài đặt hoặc nâng cấp, chỉ in ra stdout
- requires** chỉ ra các yêu cầu của gói

Mã nguồn cho rất nhiều packages RPM cũng có thể được để dưới dạng package RPM và sẽ được sử dụng để xây dựng một package nhị phân. Tên kết hợp sẽ là:

`name-version-release.src.rpm` (`tên-phiên_bản-ngày_xuất-bản.src.rpm`)

Các gói như vậy sẽ chứa ít nhất 2 file, tarball cùng mã nguồn và một spec file. spec file chứa đựng chỉ dẫn để vá (patch), dịch và xây dựng RPM package. Nếu mã nguồn cần được vá trước khi dịch thì miếng vá sẽ nằm trong package nguồn.

Quản trị Hệ thống Linux - Cơ bản

Có ba cách khác nhau để xây dựng một package RPM. Giả sử rằng bạn có một package với tên gọi: `name-version-release.src.rpm`.

*Đối với các phương thức này, trước tiên bạn cần cài đặt gói **rpm-build***

Cách 1:

Cài đặt package nguồn RPM với:

```
rpm -ivh name-version-release.src.rpm
```

Lệnh trên sẽ copy các file vào thư mục sau:

```
/usr/src/redhat/SPECS  
/usr/src/redhat/SOURCES
```

Trong thư mục `/usr/src/redhat/SPECS` có một file với tên **name.spec** (trong đó 'name' là tên của package). Để bắt đầu xây dựng package dịch, tên **name-version-release.i386.rpm**, gõ trong cửa sổ lệnh:

```
rpm -ba name.spec
```

Dòng lệnh trên sẽ kích hoạt một loạt các scripts. tarball trong

```
/usr/src/redhat/SOURCES sẽ được mở (unpack) tại  
/usr/src/redhat/BUILD
```

Nếu quá trình dịch thành công thì package nhị phân sẽ được lưu trong

```
/usr/src/redhat/RPMS/.
```

Có một số các thư mục thứ cấp khác nhau tương ứng với một số models/thể hệ của CPU. Nếu quá trình dịch không liên đới tới các đặc tính đặc biệt từ các chip thì package đó sẽ được lưu vào thư mục `noarch`.

Cách 2:

Cách này cũng tương tự như cách thứ 1 nhưng bắt đầu với lệnh đơn sau đây:

```
rpm --rebuild name-version-release.src.rpm
```

Cách 3:

Trong một vài trường hợp nhà phát triển sẽ phân phối tarball cùng với nhau trong một file spec. Nếu tarball được gọi tên `name-version-release.tar.gz` bạn có thể tìm một file `.spec` với lệnh sau:

```
tar tzvf name-version-release.tar.gz | grep .spec
```

Nếu tarball có một file spec thì bạn có thể xây dựng một package RPM bằng cách gõ:

```
rpm --bt name-version-release.tar.gz
```

Công cụ Alien

Công cụ này sẽ chuyển đổi packages Debian sang Redhat và ngược lại. Bạn có thể tải xuống tại: <http://kitenet.net/programs/>

Bài tập

Trong các ví dụ sau tải một file RPM nguồn (vd. bash-2.05-8.src.rpm với Redhat 7.2) từ www.rpmfind.net

1. Cài đặt tarball

Bung các thành phần của gói RPM mà không dịch bất cứ file nào:

```
rpm -ivh bash-2.05-8.src.rpm
```

Trong thư mục /usr/src/redhat/SOURCES, mở gói tarball với:

```
tar xvzf bash-2.05-8.tar.gz
```

Tuỳ chọn(khuyến nghị): Miếng vá có thể được áp dụng. Cú pháp sẽ thay đổi phụ thuộc vào bạn đang ở thư mục nào

Từ /usr/src/redhat/SOURCES:

```
patch -p0 -b <file.patch
```

Từ /usr/src/redhat/SOURCES/bash-2.05-8

```
patch -p1 -b <file.patch
```

Cuối cùng dùng:

```
./configure  
make
```

Nếu bạn chắc chắn bạn muốn cài đặt package này hãy dùng make install nhưng nhớ rằng nó sẽ không cài đặt phần mềm sử dụng package manager.

2. Xây dựng lại RPM package manager.

```
rpm --rebuild package.src.rpm
```

Package nhị phân đã dịch sẽ ở trong /usr/src/redhat/RPMS

- Kiểm tra thành phần của package với tùy chọn -qpl
- cài đặt package, và chạy truy xuất với các package cài đặt
- Gỡ bỏ(uninstall) package

THAO TÁC VỚI VĂN BẢN NÂNG CAO

Tìm kiếm một từ hoặc một cụm từ trong một văn bản được lưu trữ sử dụng **grep**, **fgrep** hoặc **egrep**. Các từ khoá sử dụng trong quá trình tìm kiếm là một tổ hợp của các ký tự được gọi là biểu thức chính quy (regular expressions-regex). Biểu thức chính quy được nhận dạng bởi rất nhiều ứng dụng như **sed**, và **vi**.

Các biểu thức chính quy

Bảng 1. Danh sách regex chính

Ký tự	Tìm kiếm tương ứng
x (hoặc bất cứ ký tự nào)	Các chuỗi chứa đựng ‘x’
\<KEY	Các từ bắt đầu bằng ‘KEY’
WORD\>	Các từ kết thúc bằng ‘WORD’
^	Bắt đầu của một dòng
\$	Kết thúc một dòng
[Range]	Giới hạn của bảng mã ASCII
[^c]	Không phải ký tự ‘c’
\[Ký tự ‘[’
“cat*”	Chuỗi chứa đựng ‘ca’ hoặc ‘cat’ và các ký tự bất kỳ tiếp theo
“.”	Tìm kiếm các ký tự đơn

Biểu thức chính quy mở rộng (extended regex- eregex): Các ký tự chính của eregex là: . ? , () và |

Bảng 2: Danh sách eregex chính

Ký tự	Tìm kiếm tương ứng
-------	--------------------

“A1 A2 A3”	Chuỗi chứa đựng ‘A1’ hoặc ‘A2’ hoặc ‘A3’
“cat+”	Chuỗi chứa đựng ít nhất cat và các ký tự bất kỳ tiếp theo
“cat?”	Chuỗi chứa đựng ‘ca’ hoặc ‘cat’ và các ký tự bất kỳ tiếp theo.

Họ grep

Tính năng grep hỗ trợ biểu thức chính quy regex như đã mô tả ở bảng1.

egrep

Công cụ egrep hỗ trợ biểu thức chính quy mở rộng eregex như mô tả trong bảng2.

fgrep

fgrep biểu diễn cho grep nhanh và **fgrep** dịch chuỗi gốc (không có hỗ trợ của regex hoặc eregex)

Làm việc với grep

Cú pháp của grep:

grep PATTERN FILE

Grep	Main Options
-c	Đếm số lượng dòng trùng với PATTERN
-f	Tìm PATTERN từ file
-i	bỏ qua các trường hợp nhạy cảm
-n	chỉ ra số dòng của file
-v	xuất ra tất cả các dòng từ những dòng chứa PATTERN
-w	Tìm kiếm chính xác tuyệt đối PATTERN

Ví dụ đưa ra danh sách của tất cả các dòng không trống trong /etc/lilo.conf:



```
grep -v "^$" /etc/lilo.conf
```

egrep và fgrep

Tiện ích fgrep không nhận biết được ngữ nghĩa đặc biệt của một biểu thức chính quy. Ví dụ



```
fgrep "cat*" FILE
```

Dòng lệnh trên chỉ tìm kiếm các từ chứa đựng 'cat'. Khả năng của fgrep được bổ sung thêm nhờ lựa chọn LIST. Cú pháp như sau :

```
fgrep -f LIST FILE
```

Tiện ích **egrep** sẽ thực hiện với mọi biểu thức chính quy mới. Nó cũng có thể tìm kiếm một vài từ khoá nếu chúng được bắt đầu với dòng lệnh được chia bởi pipes. Ví dụ:



```
egrep "linux|^image" /etc/lilo.conf
```

Bộ soạn thảo Stream – sed

Tiện ích sed thông thường được sử dụng để tìm kiếm và thay đổi pattern trong văn bản. Nó hỗ trợ phần lớn các biểu thức chính quy (regex).

Làm quen với sed

Cú pháp :

```
sed [option] 'lệnh' [INPUTFILE]
```

file input là tùy ý vì **sed** cũng làm việc trong các *thư mục file* và *pipes*. Đây là một vài ví dụ giả sử chúng ta làm việc trong một file gọi là MODIF.

Xoá tất cả các dòng chú thích :



```
sed '/^/ d' MODIF
```

Chú ý rằng pattern được tìm kiếm nằm giữa hai gạch chéo //.

Thay thế /dev/hda1 bởi /dev/sdb3:



```
sed 's/\\/dev\\/hda1/\\/dev\\/sdb3/g' MODIF
```

Ký tự **s** trong dòng lệnh biểu diễn cho ‘substitute’. Ký tự ‘**g**’ biểu diễn cho ‘globally’ và ép substitution trên mọi dòng

Nếu dòng chứa đựng từ khoá **KEY** thì thay thế ‘:’ với ‘;’



```
sed '/KEY/ s/:/;/g' MODIF
```

sed nâng cao

Bạn có thể sử dụng một vài lệnh bắt đầu với **-e** tại dòng lệnh. Ví dụ, (1) xoá tất cả dấu trống khi (2) thay thế ‘OLD’ bằng ‘NEW’ trong file MODIF



```
sed -e '/^$/ d' -e 's/OLD/NEW/g' MODIF
```

Các lệnh trên có thể được viết vào một file, ví dụ COMMANDS. Khi đó mỗi dòng được dịch như một dòng lệnh để chạy.



```
sed -e '/^$/ d' -e 's/OLD/NEW/g' MODIF
```

1 s/old/new
/keyword/ s/old/new/g
23,25 d

Cú pháp sử dụng cùng COMMANDS file là:

sed -f COMMANDS MODIF

Việc này tiện lợi hơn rất nhiều việc phải đánh liên tục những dòng lệnh dài.

Tóm tắt lựa chọn cho sed

Cờ dòng lệnh
-e Thực hiện các lệnh tiếp sau đó
-f Đọc các lệnh từ một file
-n Không in ra các dòng không được sửa đổi

Tuỳ chọn của lệnh
d Xóa một dòng
r Đọc một file và xuất ra file output
s Thay thế
w Ghi kết quả ra vào một file

Bài tập

1. Tạo một file mới có tên FILE với nội dung sau:

Using grep,
fgrep and
egrep

Quản trị Hệ thống Linux - Cơ bản

```
to grep for 99% of the cats
% these are two
% commented lines
```

Sử dụng grep để xuất ra chỉ những dòng lệnh không phải là dòng chú thích

Tìm kiếm các dòng chứa đựng các từ bắt đầu với 'a'

2. Biểu thức chính quy. Thêm các dòng sau vào file trên:

```
ca
cat
cats
catss
cat+
cat*
cat?
car
carriage
```

Xem kết quả của các lệnh sau khi sử dụng grep, egrep và fgrep:

```
grep 'cat+' FILE
grep 'cat?' FILE
grep 'cat.' FILE
grep 'cat*' FILE
```

3. Sử dụng sed để thực hiện các thay đổi sau trong FILE

(sử dụng file COMMAND, sau đó làm các bước sau trên dòng lệnh)

- trong dòng đầu thay thế 'grep', với 'soap'
- xoá 'fgrep' trong dòng thứ hai
- thay thế 'egrep' với 'water'
- trong dòng thứ tư thay thế 'grep for' với 'wash'

Save kết quả vào một file sử dụng tùy chọn w

SỬ DỤNG TRÌNH SOẠN THẢO VI

vi được sử dụng như là trình soạn thảo chính trong Linux, một công cụ hữu ích như grep hoặc cat và được đặt tại /bin

Các chế độ Vi

Để thực hiện các thao tác phức tạp như là copy/paste, trình soạn thảo vi có thể thực hiện bằng nhiều chế độ khác nhau

- Chế độ dòng lệnh (Command Mode)

Đây là chế độ soạn thảo và đánh dấu thường sử dụng một chữ cái. Ví dụ dùng chữ cái j để nhảy xuống dòng tiếp theo

Như là qui tắc ngón tay cái (rule of thumb), nếu bạn muốn thực hiện một thao tác nhiều lần, bạn có thể điền số lần thực hiện trước khi gõ câu lệnh. Ví dụ: dùng lệnh 10j để nhảy đến 10 dòng tiếp theo.

- Chế độ dòng (hoặc cột) cuối cùng

Bạn có thể sử dụng chế độ này ở màn hình dòng lệnh (command line mode) bằng cách đánh dấu hai chấm. Cột sẽ hiển thị ở góc bên trái cuối cùng của màn hình. Trong chế độ này, bạn có thể thực hiện các thao tác đơn giản như tìm kiếm, ghi dữ liệu, thoát hoặc chạy một câu lệnh shell.

- Chế độ chèn

Cách đơn giản nhất để thực hiện chế độ này trong màn hình dòng lệnh (command Mode) là dùng chữ cái i hoặc a. Đây là chế độ trực quan nhất và thường được sử dụng để chèn văn bản vào một tài liệu.

Phím Esc sẽ thoát chế độ chèn và quay trở về màn hình dòng lệnh

Các mục văn bản

Các mục văn bản như là từ (words) hoặc đoạn văn bản (paragraph) được định nghĩa trong chế độ dòng lệnh (command mode) cho phép soạn thảo các lệnh sử dụng trong các tài liệu văn bản mà không cần dòng đến thiết bị chuột.

Từ, câu và đoạn (Words, sentences and paragraphs)

e reps. b	Chuyển đến cuối / đầu từ hiện thời
(reps.)	Chuyển đến cuối / đầu câu hiện thời
{ reps. }	Chuyển đến cuối / đầu đoạn hiện thời
w	tương tự như e nhưng thêm một dấu cách sau từ hiện thời

Đầu và cuối (Beginning and End)

^	Đầu dòng
\$	Cuối dòng
1G	Đầu tệp
G	Cuối tệp

Tất cả các mục văn bản trên có thể được sử dụng để đánh dấu một chữ (**w**) hoặc một đoạn văn bản (**{}**) một lần, di chuyển đến đầu dòng (**^**) hoặc đầu tệp (**G**), vv... cũng như được sử dụng để thực hiện các câu lệnh như xoá hoặc copy.

Chèn văn bản

Trong chế độ dòng lệnh, **i** cho phép bạn chèn thêm văn bản vào tài liệu. Các đặc tính khác của trình soạn thảo **vi** cũng được thực hiện tương tự như vậy. Bảng sau đây sẽ liệt kê toàn bộ các đặc tính chèn văn bản của **vi**.

Các câu lệnh chèn

a	Chèn văn bản với con trỏ tại ký tự cuối cùng của dòng
A	Chèn văn bản với con trỏ tại ký tự cuối cùng ở cuối dòng
i	Chèn văn bản tại vị trí con trỏ hiện tại
o	Chèn văn bản vào dòng mới
O	Chèn văn bản vào dòng mới phía trên
s	Xoá ký tự hiện thời và chèn văn bản
S	Xoá dòng hiện thời và chèn văn bản

Xoá văn bản

Nếu bạn muốn xoá một ký tự đơn trong chế độ dòng lệnh thì dùng **x** và để xoá dòng hiện tại thì dùng **dd**.

Chú ý: Gần như tất cả các câu lệnh trong vi có thể được lặp lại bằng cách gõ thêm số lần lặp lại ở phía trước. Bạn cũng có thể cách này đối với các mục văn bản (như từ, câu, đoạn văn bản, ...) bằng cách thay thế thực thể (entity) sau câu lệnh.

Bảng 4: Các từ và ký tự

w	Chữ đơn
l	Ký tự đơn

Ví dụ:

Xoá một từ

`dw`

Xoá văn bản từ vị trí con trỏ đến cuối dòng hiện tại

`d$`

Xoá văn bản từ vị trí con trỏ đến cuối đoạn hiện tại

`d}`

Bạn có thể xoá cùng lúc một mục văn bản đồng thời chuyển sang chế độ chèn với lệnh **c**. Như thường lệ bạn có thể sử dụng câu lệnh này với một mục văn bản như **w** hoặc **{**.

Copy / Paste

Thao tác copy trong **vi** là câu lệnh **y** (thay cho yank), và thao tác chèn là **p**.

Nếu một dòng được copy thì sẽ được chèn vào dòng tiếp theo phía dưới con trỏ.

Việc lựa chọn văn bản được thực hiện với các mục văn bản thông dụng như **w**, **l**, **}**, **\$**, ... Một số ngoại lệ được mô tả trong ví dụ dưới đây.

Ví dụ:

Quản trị Hệ thống Linux - Cơ bản

Sao chép văn bản từ vị trí hiện tại đến cuối dòng hiện thời

`y$`

Sao chép toàn bộ dòng hiện thời

`YY`

Sao chép 3 dòng

`3yy`

Mục xoá cuối cùng thông thường được đưa vào bộ đệm và có thể được chèn với câu lệnh **p**. Điều này tương đương với thao tác copy và chèn.

Tìm kiếm

Do việc tìm kiếm đòi hỏi phải khớp theo mẫu do đó một lần nữa chúng ta lại đề cập đến các biểu thức chính qui (regular expressions – regex). Như một số công cụ thao tác với văn bản của UNIX như **grep** hoặc **sed**, vì cũng tuân thủ các biểu thức chính qui này.

Để thực hiện tìm kiếm, đầu tiên phải chuyển về chế độ dấu hai chấm. Câu lệnh `/` sẽ tìm kiếm từ vị trí hiện tại xuống cuối và câu lệnh `?` sẽ tìm kiếm theo hướng ngược lại.

Để có thể thực hiện thao tác tìm kiếm và thay thế. Cú pháp tương tự như đối với **sed**.

Ví dụ:

Tìm từ bắt đầu từ chữ ‘comp’ trong toàn bộ văn bản

`/\<comp>`

Tìm dòng bắt đầu từ chữ cái z

`/^z`

Quản trị Hệ thống Linux - Cơ bản

Tìm trong toàn bộ văn bản với từ khoá 'VAR' và thay thế bằng 'var'

```
:% s/VAR/var
```

Làm lại (Undo)

Chúng ta luôn có thể huỷ bỏ các thao tác vừa thực hiện (trong chế độ dòng lệnh) với câu lệnh **u**, và có thể sử dụng đối với tệp khi chưa thao tác ghi chưa được thực hiện.

Ghi văn bản

Câu lệnh ghi dữ liệu là **w**. Bằng cách này tài liệu sẽ mặc định được ghi lại. Người dùng cũng có thể xác định tên cho tệp cần ghi. Từng đoạn (portion) văn bản có thể được ghi lại sang tệp bản bản khác trong khi các tệp văn bản khác đang được đọc hoặc chèn tại tài liệu hiện thời. Ví dụ sau sẽ thể hiện điều này.

Ví dụ:

Ghi tài liệu hiện tại ra tệp có tên là 'newfile'

```
:w newfile
```

Ghi dòng 15 đến dòng 24 sang tệp có tên là 'extract'

```
:w 15,24 extract
```

Đọc từ tệp 'extract'. Văn bản sẽ được chèn vào vị trí con trỏ hiện tại

```
:r extract
```

Chú ý: trong ngữ cảnh *chế độ cột* (column mode) chúng ta phải thực hiện như sau

. là dòng hiện thời

\$ là cuối tài liệu

Bài tập

Tại root *cp /var/log/messages to /tmp*. Sử dụng chức năng tìm kiếm và thay thế của vi để tạo ra tất cả các dòng bắt đầu với “and end with”;

Gõ “u” để huỷ bỏ tất cả các thay đổi.

Copy */etc/lilo.conf* tới */tmp*, soạn thảo tệp này và thử *copy/paste yy/p* và *cut/paste* với *dd/p*

Kiểm tra kết quả của *:x*, *ZZ*, *:quit*, *:wq*, và *:q!* (câu lệnh nào sẽ ghi dữ liệu và câu lệnh nào không)

Kiểm tra thử kết quả sau khi sử dụng một số chế độ chèn văn bản như: **A**, **a**, **O**, **o**, **S** và **s**

Lựa chọn: Nếu bạn cài đặt gói **vim-enhanced** thì chương trình **vimtutor** sẽ cho thấy một số lựa chọn thông dụng của vi.

NHÂN LINUX

Khái niệm nhân

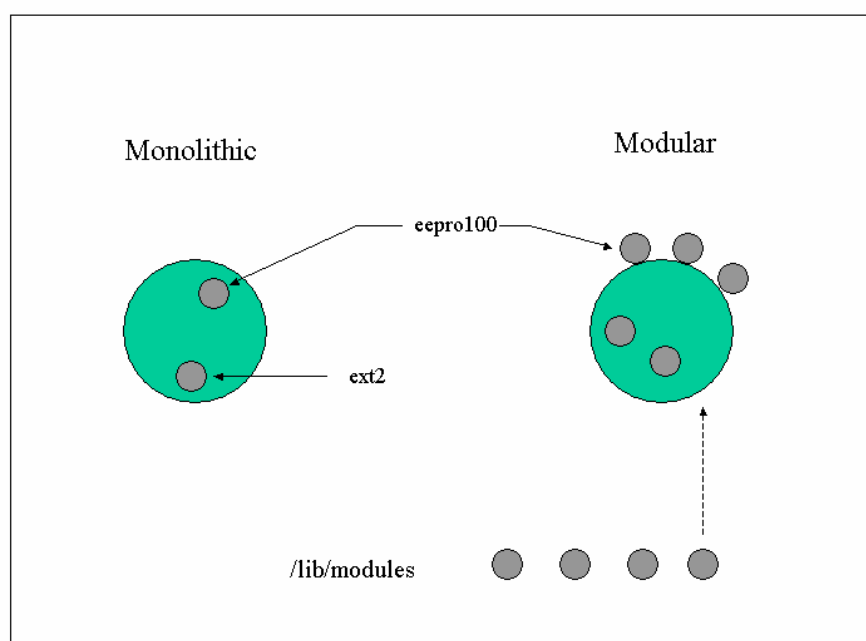
Có 2 kiểu nhân Linux, đó là:

A: Nguyên khối (Monolithic)

Là một loại nhân hỗ trợ tất cả các phần cứng, network và filesystem, được biên dịch vào trong một file image đơn.

B: Hỗ trợ module (Modular)

Là loại nhân chứa một số trình điều khiển, được biên dịch như là các file đối tượng mà nhân linux có thể tải vào và xóa khi được yêu cầu. Loadable modules được đặt trong thư mục **/lib/modules**.



Ưu điểm của loại modular kernel là không cần phải dịch lại khi cần thêm phần cứng hoặc thay thế phần cứng, nhanh, tiện và đáp ứng được hầu hết các trường hợp sử dụng. Monolithic có ưu điểm so với modular kernel chính ở đặc điểm không thể nạp thêm module mới vào nhân. Trong những hệ thống nhạy cảm, monolithic kernel kết hợp với việc không cài đặt trình biên dịch sẽ hạn chế

hacker rất nhiều trong việc sử dụng những module điều khiển dạng backdoor ở mức nhân.

Nhân Modular

Rất nhiều thành phần của nhân linux có thể biên dịch như là các modules và các module này có thể tải vào hoặc xóa khi cần thiết.

- Các module cho nhân linux được lưu trong: `/lib/modules/<kernel-version>`.
- Các thành phần tốt nhất để module hóa là các thành phần không cần cho quá trình boot máy, ví dụ các thiết bị ngoại vi và hệ thống và hệ thống file phụ.
- Các module của nhân linux được điều khiển bằng các tiện ích nằm trong gói modutils
 - ***lsmod***
 - ***rmmod***
 - ***insmod***
 - ***modprobe***
 - ***modinfo***

Nhiều module phụ thuộc vào sự có mặt của module khác. File lưu thông tin về các module phụ thuộc `/lib/modules/<kernel-version>/modules.dep` được sinh ra bởi lệnh **depmod**. Lệnh này được thực thi bởi **sript rs.sysinit** khi boot máy.

-- **modprobe** sẽ tải tất cả các module và các module phụ thuộc sẽ được liệt kê trong **modules.dep**

-- `/etc/modules.conf` dùng để lưu các tham số module (IRQ và IO ports) nhưng thường chứa một danh sách các bí danh (alias). Những bí danh cho phép ứng dụng tham chiếu đến thiết bị bằng một tên thông dụng. Ví dụ thiết bị ethernet đầu tiên luôn gọi là eth0 và không dùng tên của trình điều khiển cụ thể.

Hình 1: Ví dụ file /etc/modules.conf:

```
alias eth0 e100
alias usb-core usb-uhc
```

Quản trị Hệ thống Linux - Cơ bản

```
alias sound-slot-0 i810_audio
alias char-major-108 ppp_generic
alias ppp-compress-18 ppp_mppe

# 100Mbps full duplex
options eth0 e100_speed_duplex=4
```

Biên dịch lại nhân

Giải nén mã nguồn

Mã nguồn của nhân linux lưu trong thư mục */usr/src/linux*, thư mục này là một liên kết mềm tới thư mục */usr/src/(kernel-version)*. Khi giải nén mã nguồn của nhân mới nên:

- Xóa liên kết mềm tới thư mục chứa mã nguồn nhân cũ.



```
rm linux
```

Mã nguồn của nhân đóng gói dưới dạng gói RPM thường tạo ra một liên kết tên là **linux-2-4**

- Giải nén mã nguồn mới (e.g linux-2.4.20.tar.bz2)



```
tar xjf linux-2.4.29.tar.bz2
```

Chú ý: Nhân phiên bản 2.2 tạo ra thư mục tên *linux* chứ không phải *linux-version*. Do đó bước 1 là rất quan trọng, ngoài ra có thể nghi ngờ mã nguồn cũ bằng mã nguồn nhân mới. Từ nhân phiên bản 2.4 trở đi, tên thư mục là *linux-version*.

- Tạo một liên kết mềm tên là linux từ thư mục mới vừa được tạo



```
ln -s linux-2.4.20 linux
```

- Đến đây, nhân đã sẵn sàng cho việc cấu hình, nhưng chúng ta phải chắc chắn rằng, tất cả file nhị phân cũ đã được xóa khỏi thư mục chứa mã nguồn của nhân, để xóa các file nhị phân hãy dùng lệnh **make mrproper**.

Cấu hình nhân

Đầu tiên soạn thảo file **Makefile** và thiết lập biến “EXTRAVERSION” khác với các phiên bản đã có:

```
VERSION = 2
```

```
PATCHLEVEL = 4
```

```
SUBLEVEL = 20
```

```
EXTRAVERSION = -test
```

Bây giờ là lúc cấu hình cho nhân linux, công việc cơ bản của việc cấu hình là tạo một file có tên gọi **.config** bằng cách: từ thư mục **/usr/src/linux** thực hiện một trong các lệnh sau:

```
make menuconfig
```

```
make xconfig
```

```
make config
```

Tất cả các lệnh này sẽ ghi vào file **/usr/src/linux/.config**

Thông thường để dễ dàng trong việc cấu hình một nhân mới sử dụng file **.config** cũ bằng cách sử dụng lệnh **make oldconfig**. Lệnh này sẽ chỉ nhắc người dùng những đặc tính mới trong cây thư mục mã nguồn của nhân (nếu nhân mới hơn hoặc nhân được sửa chữa)..

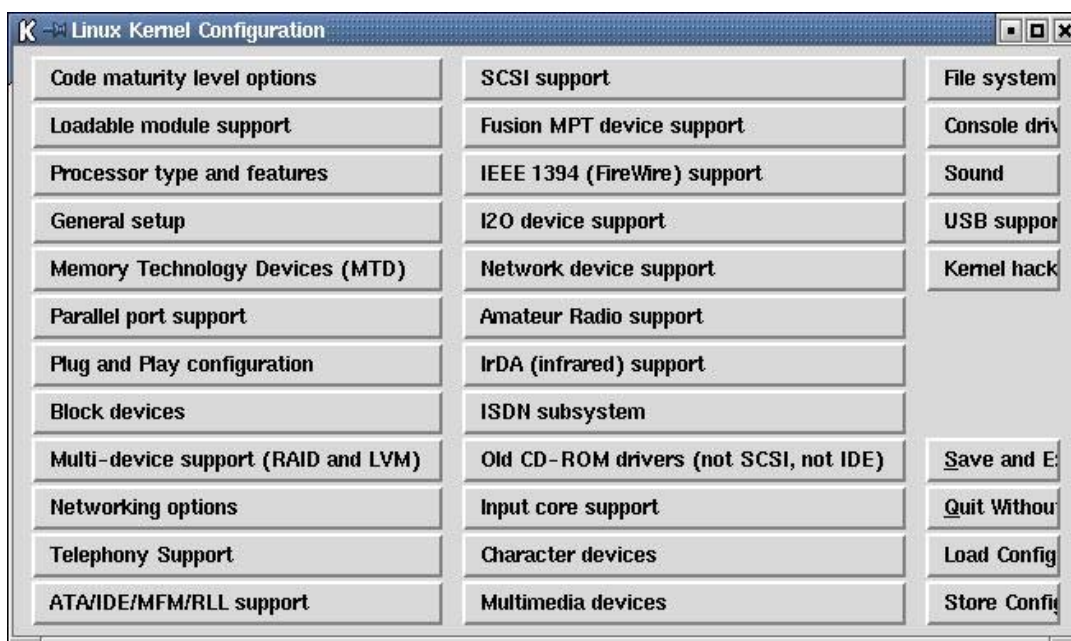
Chú ý: một số dòng linux (distributions linux) ví dụ RedHat có một thư mục **configs** con chứa các file config với các thông số cấu hình được thiết lập trước.

Để kích hoạt các tính năng nhân (với **make menuconfig**) bạn sẽ phải nhập category mức cao nhất bằng cách chuyển các phím mũi tên và bấm enter để truy cập vào category mong muốn. Trong category cụ thể, bấm thanh dấu cách sẽ làm thay đổi nhân hỗ trợ đối với một đặc tính hoặc một driver

Các khả năng hỗ trợ là

- Hỗ trợ (biên dịch tĩnh) [*****]
- modular (biên dịch động) [**M**]
- không hỗ trợ []

Các lựa chọn giống như trên cũng có thể sử dụng đối với các chế độ **config** và **xconfig**.



Hình2: make xconfig ở giao diện mức trên cùng:

Dịch nhân

make dep

Khi cấu hình nhân xong, cần đối chiếu lại các chọn lựa trong tất cả các thư mục con trong thư mục mã nguồn của nhân, bằng cách dùng lệnh **make dep**. File `.depend` chứa đường dẫn tới các header file nằm trong thư mục `/usr/src/linux/include`, những file này được sinh ra cùng với **dep** target.

make clean

Lệnh **make** nhận chỉ thị từ **Makefile** và sẽ tạo (build) những thứ cần thiết. Nếu file nào đã có rồi thì lệnh **make** sẽ sử dụng chúng. Cụ thể là những file có mở rộng là: `*.o`.

Đảm bảo mọi lựa chọn cấu hình trong **.config** được sử dụng để tạo lại các file, cần chạy lệnh **make clean** (để xóa các file `*.o`)

Chú ý: Bạn không cần chạy lệnh **make clean** ở giai đoạn này nếu bạn đã tạo thư mục nguồn bằng lệnh “**make mrproper**”.

Sau hai lệnh trên (với những bản nhân 2.6 trở lên, mới dịch lần đầu thì không cần thiết), nhân linux được biên dịch bằng một trong hai lệnh sau:

make zImage

make bzImage

Khi thực hiện biên dịch xong mà không có bất cứ lỗi nào, sẽ có một file tên là **vmlinux** nằm trong thư mục `/usr/src/linux/`.

Hai lệnh khác sẽ tạo một file bổ sung trong `/usr/src/linux/arch/i386/boot` gọi là **zImage** và **bzImage**. Hai lệnh này nén nhân bằng gzip và bzip2. Xem mục **cài đặt một nhân mới** để biết cách xử lý những file này.

make modules

Dùng để biên dịch các modules

make modules_install

Lệnh này sẽ copy các modules vào các thư mục tương ứng trong **/lib/modules**

Dãy các lệnh được minh họa trong hình 3:

Hình 3 các lệnh biên dịch nhân:

```
make dep
make clean
make bzImage
make modules
make modules_install
```

Cài đặt một nhân mới

Nhân mới nằm trong **/usr/src/linux/arch/i386/boot/bzImage**, phụ thuộc vào kiến trúc máy của bạn. File này phải được copy vào thư mục **/boot**, và đặt tên là **vmlinuz-<full-kernel-version>**



```
/usr/src/linux/arch/i386/boot/bzImage /boot/vmlinuz-<full-kernel-version>
```

Tiếp theo chỉnh sửa file **/etc/lilo.conf** hoặc **/boot/grub/grub.conf** để add nhân mới được biên dịch vào boot menu. Copy phần “image” của nhân mới và đưa vào cuối file như hình minh họa:

Soạn thảo file /etc/lilo.conf

Quản trị Hệ thống Linux - Cơ bản

```
Prompt
timeout=50
message=/boot/message

image=/boot/vmlinuz
    label=linux
    root=/dev/hda6          Existing section
    read-only

image=/boot/vmlinuz-<full-kernel-version>
    label=linux-new        Added section
    root=/dev/hda6
    read-only
-----snip-----
```

Bảng ký hiệu cho các thủ tục nhân khác nhau có thể copy vào thư mục **/boot**:



```
cp /usr/src/linux/System.map /boot/System.map-<full-kernel-version>
```

Phiên bản nhân đầy đủ

Trong một hệ thống, phiên bản của nhân đang chạy có thể được in ra với câu lệnh

uname -r

Phiên bản nhân này cũng có thể được hiển thị trên các terminal ảo nếu tham số lựa chọn **\k** được sử dụng trong **/etc/issue**.

Khởi tạo Ramdisks

Ramdisk được sử dụng để hỗ trợ quá trình khởi động nạp các module truy cập những block device cần thiết (IDE, SCSI, RAID) cho việc truy cập phân vùng

Quản trị Hệ thống Linux - Cơ bản

root lần đầu tiên (dạng ro). Ramdisk được tạo bằng cách sử dụng lệnh `mkinitrd` với hai tham số: tên file, và số hiệu phiên bản của nhân.

Nếu bạn sử dụng ramdisk thì bạn phải thêm dòng `initrd = line` trong **`/etc/lilo.conf`**

Ví dụ:



```
mkinitrd /boot/initrd-$(uname -r).img $(uname -r)
```

Lựa chọn

Bạn nên copy file `/usr/src/linux/.config` vào `/boot/config-<full-kernel-version>`

Chạy lại LILO

Cuối cùng LILO cần phải được chạy lại để cập nhật boot loader. Lúc đầu LILO có thể chạy ở chế độ kiểm thử để kiểm xem có lỗi trong file cấu hình không.

Thực hành

Trước khi bắt đầu làm những bài tập, bạn hãy kiểm tra trong thư mục **/usr/src**, nếu có nhân rồi thì hãy xóa bỏ và chú ý đến liên kết mềm tới thư mục **/usr/src/linux**

Bài 1: Dịch lại nhân linux theo các bước sau đây:

1. Download gói **kernel-version** mới nhất từ hai trang www.kernel.org và www.redhat.com.
 - Cài đặt 2 gói này ra hai thư mục khác nhau trong /usr/src, so sánh sự khác nhau.
 - Lần lượt biên dịch hai nhân theo các chỉ dẫn ở trên và cài đặt vào hệ thống như những tùy chọn khởi động.

KHỞI ĐỘNG LINUX

Tổng quan

Hiểu biết rõ hơn về tiến trình khởi động sẽ giúp chúng ta có thể gỡ rối khi gặp vấn đề liên quan đến phần cứng và quản trị hệ thống.

Đầu tiên chúng ta tập trung vào vai trò của chương trình khởi động và mối liên quan giữa chương trình khởi động với file cấu hình **/etc/inittab**.

Tìm hiểu các mức thực thi (Runlevels)

Không giống với các hệ điều hành non-UNIX chỉ có hai chế độ cơ bản (on và off). Các hệ điều hành UNIX, bao gồm cả Linux có nhiều mức thực thi khác nhau ví dụ như mức “duy trì” (maintenance) hoặc mức “đa người dùng” (multi-user), ... Các mức thực thi được đánh số từ 0 đến 6.

Danh sách 1: Các mức thực thi Linux

Runlevel 0 tắt máy an toàn, Runlevel 6 khởi động lại máy an toàn
Runlevel 1 là chế độ đơn người dùng
Runlevel 2 là chế độ đa người dùng , nhưng không khởi động NFS
Runlevel 3 là chế độ đa người dùng đầy đủ
Runlevel 4 không được định nghĩa và thường không sử dụng
Runlevel 5 giống với runlevel 3 nhưng chạy trình Quản lý hiển thị đồ họa

Cả **init** và **telinit** để được dùng để chuyển đổi từ một chế độ thực thi này sang chế độ thực thi khác. Nên nhớ rằng, **init** là chương trình khởi tạo đầu tiên được thực hiện sau khi nhân hệ điều hành được khởi tạo tại thời điểm khởi động. PID đối với **init** luôn luôn bằng 1.

Danh sách 2: PID đối với init luôn bằng 1

<pre>[root@nasaspc /proc]# ps uax grep init</pre>
--

Quản trị Hệ thống Linux - Cơ bản

USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START TIME	COMMAND
root	1	0.0	0.2	1368	592	?	S	20:17 0:04	init [3]

Tại mỗi mức thực thi, hệ thống sẽ dừng hoặc khởi động một tập các dịch vụ nhất định. Các file quản lý những dịch vụ này được lưu giữ trong **/etc/rc.d/init.d**. Thư mục này chứa *gần như tất cả* các file quản lý dịch vụ mà hệ thống có thể chạy. Các dịch vụ khi chạy có thể được gọi là daemon (dịch vụ nền).

Danh sách 3: Danh sách các dịch vụ chính trong /etc/rc.d/init.d/



```
ls /etc/rc.d/init.d/
```

anacron	cups	identd	kadmin	krb5kdc	mcscsv	Nscd	random	smb	xfx
apmd	dhcpcd	innd	kdcrotate	kudzu	named	Ntpd	rawdevices	snmpd	xinetd
arpwatch	functions	ipchains	keytable	ldap	netfs	pcmcia	rhnsd	squid	
atd	gpm	iptables	killall	linuxconf	network	portmp	rwhod	sshd	
autofs	halt	irda	kprop	lpd	nfs	pgsql	sendmail	syslog	
crond	httpd	isdn	Krb524	marsrv	nfslock	pppoe	single	tux	

Chú ý: Cũng có thể dừng hoặc khởi động bằng tay các dịch vụ daemon trong /etc/rc.d/init.d bằng cách đưa ra các tham số tương ứng. Ví dụ, nếu bạn muốn khởi động lại dịch vụ web mặc định, bạn sẽ phải gõ:



```
/etc/rc.d/init.d/httpd restart hoặc service httpd restart
```

Khi làm việc với các mức thực thi, bạn sẽ cung cấp một tập các chương trình được định nghĩa trước nhất định để dừng chạy. Nếu bạn muốn ở mức thực thi 2 (runlevel 2), bạn phải gõ



```
/sbin/init 2
```

Đến lượt nó sẽ bắt **init** đọc file cấu hình **/etc/inittab** để tìm ra điều gì sẽ xảy ra ở mức thực thi này.

Trong trường hợp này (giả sử chúng ta đang chuyển đổi sang mức thực thi 2) các dòng sau trong file **inittab** sẽ được thực hiện:

```
l2:wait:/etc/rc.d/rc 2
```

Nếu bạn tìm kiếm trong file **/etc/inittab** câu lệnh “**/etc/rc.d/rc N**” sẽ khởi động tất cả các dịch vụ trong **/etc/rc.d/rcN.d** bắt đầu với S và sẽ dừng (stop) dịch vụ bắt đầu với K. Các dịch vụ này là các *biểu tượng kết nối* trỏ tới các script trong **/etc/rc.d/init.d**

Nếu bạn không muốn một tiến trình thực hiện trong một mức thực thi N cho trước, bạn có thể xóa biểu tượng kết nối (symlink) trong **/etc/rc.d/rN.d** bắt đầu bởi K.

inittab

Như đã đề cập trên, chúng ta hãy xem file **/etc/inittab**

File sẽ có cấu trúc như sau:

id : runlevel : action : command

Hình 3: file /etc/inittab

```
id:3:initdefault:
# System initialization.
si::sysinit:/etc/rc.d/rc.sysinit
l0:0:wait:/etc/rc.d/rc 0
```

```
l1:1:wait:/etc/rc.d/rc 1
l2:2:wait:/etc/rc.d/rc 2
l3:3:wait:/etc/rc.d/rc 3
l4:4:wait:/etc/rc.d/rc 4
l5:5:wait:/etc/rc.d/rc 5
l6:6:wait:/etc/rc.d/rc 6
-----snip-----
# Trap CTRL-ALT-DELETE
ca::ctrlaltdel:/sbin/shutdown -t3 -r now
-----snip-----
# Run gettys in standard runlevels
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6

# Run xdm in runlevel 5
x:5:respawn:/etc/X11/prefdm -nodaemon
```

Trường **id** có thể là bất kỳ. Nếu một **mức thực thi** được xác định thì **câu lệnh** và **hành động** được yêu cầu sẽ chỉ được thực hiện ở mức thực thi này mà thôi. Nếu không có số nào được xác định thì các dòng lệnh sẽ được thực hiện ở bất cứ mức thực thi nào.

File /etc/inittab:

Mức thực thi mặc định: mức này được thiết lập tại điểm bắt đầu của file với **id** và công việc **initdefault**. Chú ý, không có lệnh nào được đưa ra. Câu lệnh này đơn giản chỉ cho **init** biết mức thực thi mặc định là gì.

Chương trình đầu tiên được gọi bởi init: /etc/rc.d/rc.sysinit. Script này sẽ thiết lập các mặc định của hệ thống như tham số PATH, xác định nếu mạng được cho phép, tên máy chủ, ...

Các dịch vụ mức thực thi mặc định: Nếu mức thực thi mặc định là 3 thì chỉ có dòng “l3” sẽ được thực hiện. Công việc (action) sẽ là “chờ”, không có chương trình nào được thực thi cho đến khi tất cả các dịch vụ trong mức thực thi 3 được chạy.

Getty terminals: các dòng lệnh với id từ 1 đến 6 thực thi các thiết bị ảo (virtual terminal). Đây là nơi bạn có thể thay đổi số lượng các thiết bị ảo.

Mức thực thi 5: Dòng cuối cùng trong **inittab** thực thi trình quản lý Xwindow nếu mức thực thi 5 được gán.

Chú ý:

1. Bạn có thể thiết lập một thiết bị modem để nghe (listen) các kết nối trong **inittab**. Nếu modem của bạn được kết nối tới **/dev/ttyS1** thì dòng lệnh sau sẽ cho phép dữ liệu kết nối (không dữ liệu fax) sau 2 hồi chuông:

```
S1:12345:respawn:/sbin/mgetty -D -x 2 /dev/ttyS1
```

2. Khi thay đổi **/etc/inittab** bạn cần phải bắt **init** đọc lại file cấu hình này. Điều này được thực hiện khá dễ dàng bằng cách:



```
/sbin/init q
```

GRUB - GRand Unified Bootloader

Là chương trình môi thế hệ mới với nhiều tính năng mạnh, GRUB hiện nay đã là tùy chọn mặc định trong nhiều bản phân phối Linux.

```
# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this file
# NOTICE:  You have a /boot partition.  This means that
#           all kernel and initrd paths are relative to /boot/, eg.
```

Quản trị Hệ thống Linux - Cơ bản

```
#          root (hd0,0)
#          kernel /vmlinuz-version ro root=/dev/VolGroup01/LogVol100
#          initrd /initrd-version.img
#          boot=/dev/sda
default=1
timeout=0
splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu
title Fedora Core (2.6.15-1.1833_FC4)
    root (hd0,0)
    kernel /vmlinuz-2.6.15-1.1833_FC4 ro root=/dev/VolGroup01/LogVol100
    initrd /initrd-2.6.15-1.1833_FC4.img
title Fedora Core (2.6.15-1.1833_FC4smp)
    root (hd0,0)
    kernel /vmlinuz-2.6.15-1.1833_FC4smp ro root=/dev/VolGroup01/LogVol100
    initrd /initrd-2.6.15-1.1833_FC4smp.img
title Fedora Core (2.6.11-1.1369_FC4smp)
    root (hd0,0)
    kernel /vmlinuz-2.6.11-1.1369_FC4smp ro root=/dev/VolGroup01/LogVol100
    initrd /initrd-2.6.11-1.1369_FC4smp.img
title Fedora Core-up (2.6.11-1.1369_FC4)
    root (hd0,0)
    kernel /vmlinuz-2.6.11-1.1369_FC4 ro root=/dev/VolGroup01/LogVol100
    initrd /initrd-2.6.11-1.1369_FC4.img
```

Với GRUB, việc cập nhật các tham số khởi động không quá phức tạp như LILO. Chỉ cần sửa lại file **/boot/grub/grub.conf** và chép các file cần thiết vào **/boot** là lập tức có hiệu quả trong lần khởi động sau. File cấu hình của GRUB cũng có nhiều lựa chọn hơn, cho phép người dùng có thể sử dụng nhiều kịch bản khởi động khác nhau.

Cũng như LILO, GRUB cho phép lựa chọn nhiều kịch bản khi khởi động, cũng như cho phép người dùng chỉnh sửa các tham số khởi động ngay trước khi khởi động. Có thể sử dụng mật khẩu ngăn chặn việc này thông qua khai báo **password**

trong file cấu hình. Lệnh **grub-md5-crypt** cung cấp hàm mã hóa md5 cho phép che dấu mật khẩu khi sử dụng.

Các khai báo khác có thể tham khảo chi tiết thông qua lệnh **info grub**.

Trong quá trình khởi động, tất cả các thông báo nhân hệ thống được mặc định ghi lại trong **/var/log/dmesg**. File này có thể đọc và in ra stdout với tiện ích **/bin/dmesg**.

Trong quá trình cài đặt hoặc sử dụng, nếu GRUB bị hỏng, có thể dễ dàng sửa chữa lại bằng lệnh **grub-install** trong chế độ rescue.

Từ khởi động đến bash

Bây giờ chúng ta sẽ xem xét các bước trong quá trình khởi động hệ thống Linux.

Ramdisk được khởi tạo và nạp vào bộ nhớ thật để tải các module cần thiết.

Nhân hệ thống được tải từ đĩa cứng (hoặc CD...) xác định trong cấu hình của GRUB. Trong quá trình tải này thì nhân sẽ được giải nén.

Nhân hệ thống sẽ gắn (mount) phân vùng root (/) theo dạng chỉ đọc.

Lúc này các chương trình cần thiết trong **/bin** và **/sbin** đã sẵn sàng được truy cập.

Sau đó nhân hệ thống sẽ tải **init** - tiến trình đầu tiên.

init sẽ đọc file **/etc/inittab** và thực hiện theo các nội dung của nó. Cụ thể là **rc.sysinit** được chạy.

Sau đó, tất cả các khai báo trong **/etc/fstab** được ánh xạ (**mount**) và kiểm tra (**fsck**).

Tiếp theo **init** sẽ chuyển sang mức thực thi mặc định, các dịch vụ sẽ được khởi động. Dịch vụ mặc định **rc** có độ ưu tiên thấp nhất sẽ thi hành cuối cùng và gọi file **/etc/rc.d/rc.local**.

Dấu nhắc để đăng nhập hệ thống được quản lý bởi gettys trong ttys.

Thực hành

Hãy xem lại toàn bộ nội dung của phần trình bày trên và hoàn thành các bài tập sau đây:

- Thay đổi mức thực thi mặc định của hệ thống thành 3 và 5.
- Làm thế nào bạn có thể biết được mức thực thi hiện tại?
- Cho phép tổ hợp phím Ctrl + Alt + Del chỉ trong mức thực thi 3.
- Thêm một dấu nhắc đăng nhập trong tty7.
- Làm thế nào có thể bắt **init** đọc file cấu hình của nó?
- Sử dụng **dmesg** để đọc thông tin chipset card mạng của bạn.
- So sánh sự khác nhau giữa **shutdown**, **halt** và **reboot**.

Tham số lựa chọn nào của shutdown sẽ làm cho fsck tại lần khởi động tiếp theo?

- Sử dụng công cụ **chkconfig** hoặc **ntsysv** để tắt (disable) chương trình nền **sshd** (sshd daemon) trong mức thực thi hệ thống 2, 3, 4 và 5.

Đảm bảo rằng các đường link ký hiệu (symbolic links) trong các thư mục rc2.d, rc3.d, rc4.d và rc5.d đã thay đổi.

- Khởi động lại hệ thống. Tại dấu nhắc khởi động nhập tham số **init = tham số** để bỏ qua /sbin/init và khởi động một tiến trình bash đơn giản.

QUẢN LÝ NGƯỜI DÙNG VÀ NHÓM

Tạo người dùng mới

Bước 1: Tạo một tài khoản

Câu lệnh **/usr/sbin/useradd** sẽ thêm người dùng mới vào hệ thống và lệnh **adduser** thực chất cũng trở về câu lệnh này.

Cú pháp:

```
useradd [options] login-name
```

Ví dụ: thêm một người dùng với tên truy cập là rufu



```
useradd rufus
```

Các giá trị mặc định sẽ được sử dụng khi không có tham số lựa chọn nào xác định. Bạn có thể liệt kê các giá trị này với **useradd -D**

Các lựa chọn mặc định được liệt kê với **useradd -D**

```
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/bash
SKEL=/etc/skel
```

Chú ý rằng thông tin này cũng nằm trong file **/etc/default/useradd**

Bước 2: Kích hoạt tài khoản với mật khẩu mới

Để cho phép một người dùng truy cập vào tài khoản của mình, quản trị mạng phải thiết lập một mật khẩu cho người dùng bằng công cụ **passwd**

Cú pháp:

`passwd login-name`

Các bước trên dùng để tạo một người dùng mới. Nó cũng định nghĩa một môi trường người dùng như là thư mục *home directory* và một *shell* mặc định. Người dùng cũng có thể được gán cho một nhóm, và xác định nhóm *mặc định* của mình.

Làm việc với nhóm

Tất cả người dùng mới được gán vào một nhóm mặc định (hoặc nhóm *chính - primary*). Tồn tại hai qui ước.

Theo cách truyền thống, nhóm chính này chung cho tất cả người dùng được gọi là nhóm **users** với ID của nhóm là (GID) **100**. Một số nhà cung cấp sản phẩm Linux như Suse và Debian cũng tuân thủ với qui ước này.

Theo cách sắp xếp, nhóm người dùng riêng (User Private Group - UPG) này được đưa ra bởi RedHat và việc thay đổi qui ước này sẽ không làm thay đổi cách thức làm việc nhóm của UNIX. Với UPG, mỗi người dùng mới sẽ thuộc về nhóm *mặc định* của mình. Nhóm có cùng tên với tên đăng nhập (mặc định) và GID sẽ nằm trong phạm vi từ 500 đến 60000 (giống với UIDs).

Thành viên trong nhóm:

Một người dùng có thể thuộc về một hoặc nhiều nhóm bất kỳ. Tuy nhiên, tại một thời điểm (ví dụ khi tạo một tệp mới) thì chỉ duy nhất một nhóm là nhóm có tác động.

Thông tin về danh sách tất cả các nhóm mà một người dùng thuộc về có thể được liệt kê qua câu lệnh **groups** hoặc **id**.

Ví dụ đối với người dùng root:

Liệt kê tất cả ID:

Quản trị Hệ thống Linux - Cơ bản



id

→ ► uid=0(root) gid=0(root) groups=0(root), 1(bin), 2(daemon), 3(sys), 4(adm), 6(disk), 10(wheel), 600(sales)

Liệt kê tất cả các nhóm:



groups

→ ► root bin daemon sys adm disk wheel sales

Chuyển nhóm hiện thời:

Lệnh tham gia (chuyển) vào nhóm sẽ làm thay đổi nhóm *tác động* của người dùng (user's effective group) và bắt đầu một tiến trình mới mà từ đó người dùng có thể thoát ra khỏi nhóm (logout). Điều này có thể được thực hiện qua câu lệnh **newgrp**.

Ví dụ: tham gia nhóm sales



newgrp sales

Nếu câu lệnh **groups** được sử dụng thì nhóm đầu tiên trong danh sách sẽ chẳng còn là *root* mà là *sales*

Tạo một nhóm mới

Công cụ **groupadd** được sử dụng để quản trị các nhóm. Câu lệnh này sẽ thêm một thực thể vào file **/etc/group**

Ví dụ: tạo một nhóm devel



```
groupadd devel
```

Thêm một người dùng vào một nhóm:

Các công việc quản trị có thể được thực hiện bằng công cụ **gpasswd**. Có thể thêm **(-a)** hoặc gỡ bỏ **(-d)** người dùng từ một nhóm và gán một người quản trị **(-A)**. Công cụ này ban đầu được thiết kế để thiết lập một mật khẩu đơn vào một nhóm, cho phép tất cả các thành viên trong cùng một nhóm đăng nhập với cùng một mật khẩu. Vì lý do an ninh, tính năng này không còn được sử dụng nữa.

Ví dụ: thêm người dùng rufus vào nhóm devel



```
gpasswd -a rufus devel
```

File cấu hình

File /etc/passwd và /etc/shadow:

Tên của tất cả người dùng trong hệ thống được lưu giữ trong file **/etc/passwd** có cấu trúc như sau:

1. Tên truy cập
2. Mật khẩu (hoặc x nếu sử dụng file shadow)
3. UID
4. GID
5. Đoạn text mô tả người dùng
6. Thư mục gốc của người dùng
7. shell của người dùng

7 trường trên được ngăn cách bởi dấu hai chấm như được minh hoạ trong ví dụ sau đây.

/etc/passwd entry with encrypted passwd:

```
george:$1$K05gMbOv$b7ryoKGTd2hDrW2sT.h:Dr G Micheal:/home/georges:/bin/bash
```

Để dấu mật khẩu đã mã hoá từ người dùng thông thường bạn nên sử dụng file shadow. File **/etc/shadow** sẽ chứa tên người dùng và mật khẩu đã mã hoá và chỉ có thể đọc được bởi người dùng root.

Nếu bạn không có file shadow trong /etc thì bạn có thể sử dụng câu lệnh sau đây:



```
/usr/sbin/pwconv (passwd -> shadow)
```

Câu lệnh này sẽ bỏ 'x' trong trường thứ hai của file /etc/passwd và tạo file /etc/shadow. Nếu bạn không muốn sử dụng mật khẩu bóng (shadow password), bạn có thể làm như sau:



```
/usr/sbin/pwunconv (shadow -> passwd)
```

Chú ý: Khi sử dụng file mật khẩu bóng (shadow password) **/etc/passwd** thì có thể đọc được với quyền (644) và file **/etc/passwd** phải được cấm nhiều hơn (600 hoặc thậm chí 400). Tuy nhiên, khi sử dụng **pwunconv** thì phải bảo đảm thay đổi quyền trên file **/etc/password** (600 hoặc 400).

File /etc/group and gshadow:

Cũng tương tự như trên, thông tin của nhóm được lưu giữ trong file **/etc/group**. File này có 4 trường được ngăn cách nhau bởi dấu hai chấm.

1. Tên nhóm
2. Mật khẩu nhóm (hoặc x nếu file gshadow tồn tại)

3. GID

4. Dấu phẩy ngăn cách danh sách các thành viên

Ví dụ /etc/group entry:

```
java:x:550:jade, eric, rufus
```

Cũng như với người dùng, file **/etc/gshadow** cũng được tạo khi sử dụng mật khẩu bóng nhóm (shadow group passwords). Các tiện ích này được sử dụng để chuyển đổi xuôi hoặc ngược các file shadow hoặc non-shadow như sau:



/usr/sbin/grpconv

creates the /etc/gshadow file



/usr/sbin/grpunconv

deletes the gshadow file

File /etc/login.defs và /etc/skel/

File /etc/login.defs chứa các thông tin sau đây:

- thư mục mail (the mail spool directory):

MAIL_DIR

- các điều khiển thời gian của mật khẩu:

PASS_MAX_DAYS, PASS_MIN_DAYS, PASS_MAX_LEN,
PASS_WARN_AGE

- giá trị max/min của UID tự động lựa chọn trong useradd:

UID_MIN, UID_MAX

- giá trị max/min đối với lựa chọn tự động GID trong groupadd:

GID_MIN, GID_MAX

- tự động tạo một thư mục gốc với useradd:

CREATE_HOME

Thư mục /etc/skel chứa các file mặc định và sẽ được copy tới thư mục gốc của người dùng mới được tạo: **.bashrc**, **.bash_profiles**, ..

Các tham số lựa chọn của câu lệnh

useradd (Lựa chọn)

-c	ghi chú (Tên đầy đủ)
-d	đường dẫn tới thư mục gốc
-g	nhóm khởi tạo (GID). GID phải đang tồn tại
-G	dấu phẩy ngăn cách danh sách các nhóm bổ sung
-u	UID của người dùng
-s	shell mặc định của người dùng
-p	mật khẩu (mã hoá md5, sử dụng dấu !)
-e	ngày hết hạn của tài khoản
-k	thư mục skel
-n	tắt nhóm UPG

groupadd (Lựa chọn)

-g	gán một GID
-----------	-------------

Sửa thiết lập mặc định và tài khoản

Tất cả các lựa chọn trong khi tạo một người dùng hoặc nhóm có thể được thay đổi. Tiện ích **usermod** có một số tham số lựa chọn chính sau:

usermod (tham số lựa chọn)

-d	thư mục người dùng
-g	GID khởi tạo người dùng
-l	tên đăng nhập của người dùng
-u	UID của người dùng
-s	shell mặc định

Chú ý: tất cả các tham số lựa chọn trên cũng giống đối với **useradd**.

Tương tự như vậy, bạn cũng có thể thay đổi chi tiết về thông tin nhóm với tiện ích **groupmod**. Có một số tham số lựa chọn chính sau đây:

groupmod (tham số lựa chọn)

-g	GID
-n	tên nhóm

Khoá tài khoản:

- Một tài khoản người dùng có thể bị khoá bằng cách thêm vào một dấu chấm than vào mật khẩu người dùng. Có thể thực hiện điều này bằng các câu lệnh sau:

Khoá	Mở khoá
passwd -l	passwd -u
usermode -L	usermod -U

- Khi sử dụng shadow password, thay thế x bởi một dãy *
- Một tham số lựa chọn ít hữu ích là xoá toàn bộ mật khẩu với câu lệnh **passwd -d**

- Cuối cùng, có thể gán **/sbin/nologin** hoặc **/bin/false** cho shell mặc định của người dùng trong `/etc/passwd`

Mặc định ban đầu, mật khẩu người dùng có giá trị trong 99999 ngày, tương đương với 2739 năm (mặc định `PASS_MAX_DAYS`). Người dùng được thông báo trong vòng 7 ngày rằng mật khẩu của bạn sẽ bị hết hạn (mặc định `PASS_WARN_AGE`) với dòng thông báo sau mỗi khi người dùng đăng nhập vào hệ thống:

Có một tham số thời gian của mật khẩu khác được gọi là `PASS_MIN_DAY`. Đây là số ngày nhỏ nhất trước khi một người dùng có thể thay đổi mật khẩu, giá trị này được thiết lập mặc định ban đầu bằng 0.

Công cụ **chage** cho phép quản trị hệ thống thay đổi các tham số lựa chọn trên:

Cách dùng: `chage [-l] [-m min_days] [-M max_days] [-W warn] [-I inactive] [-E expire] [-d last_day] user`

Tham số `-l` đầu tiên liệt kê giá trị của policy hiện thời của một người dùng. Chúng ta chỉ đề cập đến tham số lựa chọn `-E`. Tham số này sẽ khoá một tài khoản người dùng tại thời điểm xác định. Định dạng ngày có thể theo định dạng của UNIX hoặc theo `YYYY/MM/DD`

Chú ý, tất cả các giá trị trên đều được lưu giữ trong file `/etc/shadow` và có thể thay đổi trực tiếp.

Xoá tài khoản

Tài khoản người dùng có thể được xoá bởi câu lệnh **userdel**. Để đảm bảo rằng thư mục gốc của người dùng cũng được xoá, ta sử dụng tham số lựa chọn `-r`.



```
userdel -r jade
```

Thực hành

1. Tạo người dùng

Sử dụng **useradd** để tạo người dùng có tên là *tux* với ID người dùng là 600 và ID nhóm là 550.

Sử dụng **usermode** để thay đổi thư mục gốc của người dùng

Có cần thiết phải tạo một thư mục mới không?

Nội dung của */etc/skel* có được copy sang thư mục mới không?

Các nội dung trong thư mục gốc cũ vẫn có thể được truy cập bởi người dùng *tux* không?

Sử dụng **usermode** để thêm *tux* vào nhóm *wheel*.

2. Làm việc với nhóm.

Tạo một nhóm có tên là *sales* với câu lệnh **groupadd**.

Thêm người dùng *tux* vào nhóm này bằng câu lệnh **gpasswd**.

Đăng nhập với *tux* và tham gia vào nhóm *sales* với **newgrp**.

3. File cấu hình.

Thêm một người dùng vào hệ thống bằng cách soạn thảo */etc/passwd* và */etc/group*.

Tạo một nhóm có tên là *share* và thêm người dùng *tux* vào nhóm này bằng cách soạn thảo bằng tay */etc/group*.

4. Thay đổi tài khoản

Thay đổi tham số ngày hết hạn của tài khoản người dùng *tux* bằng cách sử dụng câu lệnh **usermod**.

Khoá tài khoản người dùng (Sử dụng các công cụ hoặc soạn thảo file */etc/shadow*, ...)

Bảo vệ người dùng từ đăng nhập bằng cách thay đổi shell mặc định của người dùng thành `/bin/false`.

Thay đổi tham số `PASS_MAX_DAYS` của người dùng tux thành 1 trong file `/etc/shadow`.

5. Thay đổi thiết lập mặc định

Sử dụng `useadd -D` để thay đổi các thiết lập mặc định của hệ thống và do đó tất cả người dùng mới sẽ được gán trong `/bin/sh` thay vì `/bin/bash` (chú ý: điều này sẽ làm thay đổi file trong `/etc/defaults/`)

Soạn thảo `/etc/login.defs` và thay đổi tham số mặc định `PASS_MAX_DAYS` và do đó người dùng mới sẽ phải thay đổi mật khẩu của mình theo định kỳ 5 ngày.

CẤU HÌNH MẠNG

The Network Interface

Card mạng phải được hỗ trợ từ nhân của hệ điều hành. Để xác định những card mạng nào có thể sử dụng được, bạn có thể truy vấn thông tin qua câu lệnh **dmesg**, **/proc/interrupts**, **/sbin/lsmmod**, hoặc **/etc/modules.conf**

Ví dụ:

Dmesg

```
► Linux Tulip driver version 0.9.14 (February 20, 2001)

PCI: Enabling device 00:0f.0 (0004 -> 0007)

PCI: Found IRQ 10 for device 00:0f.0

eth0: Lite-On 82c168 PNIC rev 32 at 0xf800, 00:A0:CC:D3:6E:0F, IRQ 10.

eth0: MII transceiver #1 config 3000 status 7829 advertising 01e1.
```

cat /proc/interrupts

```
► 0:      8729602          XT-PIC  timer

   1:         4          XT-PIC  keyboard

       2:         0              XT-PIC  cascade

   7:         0          XT-PIC  parport0

   8:         1          XT-PIC  rtc

  10:     622417          XT-PIC  eth0

  11:         0          XT-PIC  usb-uhci

  14:    143040          XT-PIC  ide0
```



```
15:          180          XT-PIC  idle
```

```
/sbin/lsmođ
```

```
►           Module              Size  Used by
tulip                37360   1 (autoclean)
```

Từ ví dụ trên, chúng ta thấy rằng Chipset của card mạng Ethernet là Tulip, địa chỉ i/o là 0xf800 và ngắt (IRQ) là 10. Thông tin này có thể được sử dụng trong cả trường hợp nếu module sai được dụng hoặc các tài nguyên (i/o hoặc IRQ) không có.

Thông tin này cũng được sử dụng để chèn một module với một địa chỉ i/o khác (sử dụng tiện ích **modprobe** hoặc **insmod**) hoặc cũng có thể được ghi trong **/etc/modules.conf** hoặc **/etc/modprobe.conf** (sẽ ghi các thông số cài đặt trong lần khởi động sau).

Thông tin máy chủ (Host Information)

Các tệp sau đây được sử dụng để lưu trữ các thông tin mạng.

- **/etc/resolv.conf** chứa danh sách các máy chủ DNS

```
nameserver 192.168.1.108

nameserver 192.168.1.1

search linuxit.org
```

- **/etc/hosts** chứa địa chỉ IP của máy tính cũng như danh sách các máy chủ đã biết

```
# Do not remove the following line, or various programs
```

Quản trị Hệ thống Linux - Cơ bản

```
# that require network functionality will fail.

127.0.0.1      localhost  localhost.localdomain

# other hosts

192.168.1.108  mesa     mesa.domain.org

192.168.1.119  pico
```

1. **/etc/sysconfig/network** xác định nếu mạng phải được khởi động (có thể chứa biến HOSTNAME)

```
NETWORKING=yes

HOSTNAME=mesa.domain.org

GATEWAY=192.168.1.1
```

2. **/etc/sysconfig/network-scripts/ifcfg-eth0** Các tham số thiết lập cho eth0

```
DEVICE=eth0

BOOTPROTO=none

BROADCAST=192.168.1.255

IPADDR=192.168.1.108

NETWORK=192.168.1.0

ONBOOT=yes

USERCTL=no
```

Khởi động (Start) và dừng (Stop) mạng

• Từ chế độ câu lệnh

Công cụ chính được sử dụng để hiển thị giao diện mạng là **/sbin/ifconfig**. Đầu tiên khởi tạo module nhân được gán cho **eth0** trong **/etc/modules.conf** (ví dụ **tulip.o**) được load và sau đó gán giá trị địa chỉ IP và mặt nạ mạng (**netmask**).

Kết quả là giao diện có thể được chuyển bật và tắt mà không bị mất các thông tin này trong khi module nhân được thêm vào.

Ví dụ: Sử dụng ifconfig.

```
/sbin/ifconfig eth0 192.168.10.1 netmask 255.255.128.0  
  
/sbin/ifconfig eth0 down  
  
/sbin/ifconfig eth0 up
```

Một công cụ khác là **/sbin/ifup**. Tiện ích này đọc các tệp cấu hình hệ thống trong **/etc/sysconfig/network-script/** và gán các giá trị được lưu trữ cho một giao diện mạng nào đó. Script cho **eth0** được gọi là **ifcfg-eth0** và đã được cấu hình. Nếu giao thức khởi động như DHCP được định nghĩa thì **ifup** sẽ khởi động giao diện mạng với giao thức này.

Ví dụ: Sử dụng ifup.

```
/sbin/ifup eth0  
  
/sbin/ifup ppp0  
  
/sbin/ifdown eth0
```

• Sử dụng network script

Tại thời điểm khởi động card Ethernet được khởi tạo với **/etc/rc.d/init.d/network** script. Tất cả các file mạng liên quan được chứa trong thư mục **/etc/sysconfig/**.

Hơn nữa script có thể đọc các lựa chọn **sysctl** trong **/etc/sysctl.conf**, đây là nơi mà bạn có thể cấu hình hệ thống như một bộ định tuyến (cho phép địa chỉ IP chuyển trong nhân hệ điều hành). Ví dụ dòng lệnh

```
net.ipv4.ip_forward = 1
```

sẽ cho phép địa chỉ IP chuyển (forwarding) và file **/proc/sys/net/ipv4/ip_forward** sẽ chứa số 1

Network script được khởi động lại với câu lệnh sau

```
/etc/rc.d/init.d/network restart
```

3. Phục hồi lại DHCP

Các công cụ sau đây có thể truy vấn máy chủ DHCP cho một địa chỉ IP mới:

pump

dhcpcient

Một daemon khách hỗ trợ DHCP được gọi là **dhcpcd** (không nhầm lẫn với daemon máy chủ DHCP là **dhcpcd**).

Định tuyến

Một điều dễ nhận thấy khác khi sử dụng **ifup** là bảng định tuyến của hệ thống. Điều này có thể do file **etc/sysconfig/network** được đọc, trong khi **default**

gateway được lưu trữ, hoặc máy chủ DHCP đã gửi thông tin này cùng với địa chỉ IP. Bảng định tuyến được cấu hình, kiểm tra và thay đổi với công cụ **/sbin/route**.

Các ví dụ định tuyến:

Thêm một tuyến tĩnh (static route) vào mạng 10.0.0.0 qua thiết bị eth1 trong đó sử dụng 192.168.1.108 làm gateway cho mạng:

```
/sbin/route add -net 10.0.0.0 gw 192.168.1.108 dev eth1
```

Thêm một gateway mặc định (default gateway)

```
/sbin/route add default gw 192.168.1.1 eth0
```

Liệt kê bảng định tuyến nhân:

```
/sbin/route -n
```

► Kernel IP routing table

Destination	Gateway	Genmask	Iface
192.168.1.0	0.0.0.0	255.255.255.0	eth0
10.1.8.0	192.168.1.108	255.0.0.0	eth1
127.0.0.0	0.0.0.0	255.0.0.0	lo
0.0.0.0	192.168.1.1	0.0.0.0	eth0

Gateway mặc định (Default Gateway):

Trong danh sách cuối cùng. Trường đích là một danh sách các mạng. Đặc biệt, 0.0.0.0 có nghĩa là “mọi nơi”. Cần nhớ rằng, tồn tại 2 địa chỉ IP trong trường Gateway. Vậy địa chỉ nào là default gateway?

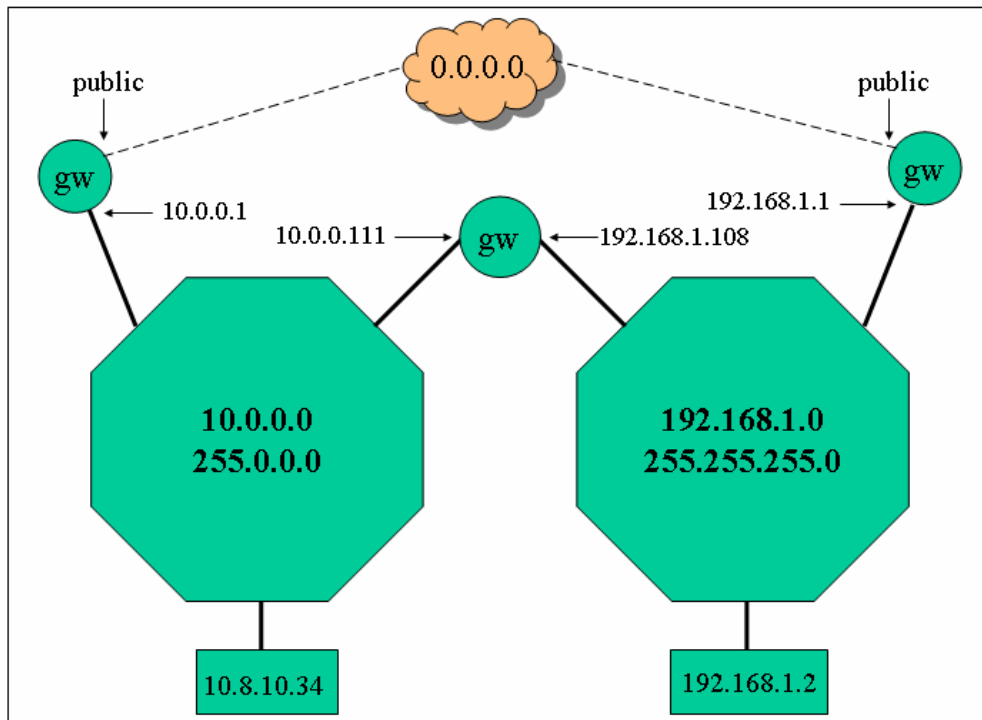
☞ Để tránh phải nhập bằng tay các tuyến tĩnh, các daemon đặc biệt **gated** hoặc **routed** được thực thi để cập nhật một cách động các bảng định tuyến qua một mạng.

☞ Nếu bạn thuộc về mạng 192.168.10.0 và bạn thêm vào một tuyến tới mạng 192.168.1.0 thì bạn có thể nhận được kết quả là các máy tính trong mạng vừa thêm vào là không có (not responding) bởi vì không có tuyến (route) được thiết lập từ mạng 192.168.1.0 tới máy chủ của bạn!! Vấn đề này có thể được giải quyết bằng cách sử dụng định tuyến động (dynamic routing)

Các tuyến tĩnh cố định

Nếu bạn có một số mạng với nhiều hơn một gateway, bạn có thể sử dụng **/etc/sysconfig/static-routes** (thay cho các daemon định tuyến). Các tuyến này sẽ được thêm vào tại thời điểm khởi động bởi **network** script.

Một kịch bản định tuyến:



10.8.10.34		192.168.1.2	
network	gw	network	gw
10.0.0.0	0.0.0.0	192.168.1.0	0.0.0.0
0.0.0.0	10.0.0.1	0.0.0.0	192.168.1.1
route add 192.168.1.0 \		route add 10.0.0.0 \	
netmask 255.255.255.0 \		netmask 255.0.0.0 \	
gw 10.0.0.111		gw 192.168.1.108	
network	gw	network	gw
10.0.0.0	0.0.0.0	192.168.1.0	0.0.0.0
192.168.1.0	10.0.0.111	10.0.0.0	192.168.1.108
0.0.0.0	10.0.0.1	0.0.0.0	192.168.1.1

Các công cụ mạng

Sau đây là danh sách ngăn các công cụ hữu ích khi gỡ rối các kết nối mạng:

ping host:

Công cụ này gửi một gói dữ liệu ICMP ECHO_REQUEST tới một máy chủ và chờ một ICMP ECHO_RESPONSE.

Các tham số lựa chọn của công cụ **ping**:

- b** ping một địa chỉ broadcast
- c N** gửi N gói tin
- q** Chế độ im lặng: hiển thị chỉ các gói tin đầu và cuối

netstat:

Bạn có thể nhận được thông tin của các kết nối mạng hiện tại, bảng định tuyến hoặc các thống kê giao diện mạng phụ thuộc vào các lựa chọn sau được sử dụng:

Các lựa chọn của **netstat**:

- r** giống như `/sbin/route`
- I** hiển thị danh sách giao diện mạng (card mạng)
- n** không giải các địa chỉ mạng IP
- p** trả về PID và tên của các chương trình (chỉ sử dụng cho root)
- v** diễn giải dài
- c** tiếp tục cập nhật

Ví dụ: Kết quả của `netstat -inet -n`:

```
► Active Internet connections (w/o servers)

```

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp		0	0 192.168.1.10:139	192.168.1.153:1992	ESTABLISHED
tcp		0	0 192.168.1.10:22	192.168.1.138:1114	ESTABLISHED

Quản trị Hệ thống Linux - Cơ bản

```
tcp          0      0 192.168.1.10:80    192.168.1.71:18858  TIME_WAIT
```

Trong danh sách trên bạn có thể thấy máy chủ địa phương (local host) đã thiết lập các kết nối ở cổng 139, 22 và 80.

arp:

Hiển thị bộ đệm giải địa chỉ nhân.

Ví dụ:

arp				
▶	Address	HWtype	HWaddress	Iface
	192.168.1.71	ether	00:04:C1:D7:CA:2D	eth0

traceroute:

Hiển thị tuyến (route) được lấy từ một máy chủ địa phương (local host) tới một máy chủ đích. Traceroute ép ngay lập tức các tuyến (routes) tới các thông báo lỗi trở về (send back error message) (ICMP TIME_EXCEEDED) bằng cách xem xét thiết lập giá trị tty (time to live) xuống mức rất thấp (too low).

Sau mỗi thông báo TIME_EXCEEDED, **traceroute** tăng giá trị của tty, gửi gói tin tiếp theo đi xa hơn cho đến khi tới được địa chỉ đích của nó.

Ví dụ:

```
CMD:    /usr/sbin/traceroute -n www.redhat.com

▶ traceroute: Warning: www.redhat.com has multiple addresses;
  using 216.148.218.197

  traceroute to www.redhat.com (216.148.218.197), 30 hops max, 38
  byte packets
1  192.168.1.1  0.440 ms  0.347 ms  0.341 ms

      ---- snip ----
```

Quản trị Hệ thống Linux - Cơ bản

```
14  12.122.2.145  112.116 ms  110.908 ms  112.002 ms
15  12.122.2.74   156.629 ms  157.028 ms  156.857 ms
16  12.122.255.222 156.867 ms  156.641 ms  156.623 ms
17  216.148.209.66 159.982 ms  157.462 ms  158.537 ms
18  216.148.218.197 157.395 ms  156.789 ms  156.080 ms
```

Các lựa chọn của **traceroute**:

- f ttl** Thay đổi thời gian sống khởi tạo về ttl thay vì giá trị 1
- n** không giải các địa chỉ IP
- v** diễn giải dài
- w sec** thiết lập thời gian chờ tại các gói trả về thành sec

Thực hành

1. Trong phần **kịch bản định tuyến** được trình bày ở trên đưa ra bảng định tuyến đối với gateway của mạng LAN.

2. Khởi động giao diện mạng của bạn bằng tay

```
ifconfig eth0 192.168.0.x
```

Liệt kê danh sách các module nhân. Đảm bảo rằng module eth0 đã được tải (kiểm tra /etc/modules.conf).

3. Dừng giao diện mạng với:

```
(i) ifconfig eth0 down
```

Chắc chắn rằng bạn có thể lưu trữ các thông tin giao diện mạng này mà không bị mất thông tin:

```
(ii) ifconfig eth0 up
```

4. Dừng giao diện mạng và gỡ bỏ module nhân (rmmod *module*). Điều gì sẽ xảy ra nếu bạn lặp lại bước 3 (ii)?

5. Chia lớp thành hai mạng A (192.168.1.0) và B (10.0.0.0).

- Thử truy cập các máy qua các mạng
- Chọn một máy làm gateway (tại một trong hai mạng)
- **Chỉ trên máy gateway!** thực hiện các lệnh sau:

-- cho phép chuyển IP (allow IP forwarding):

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

-- đưa ra một giao diện mạng đã được gán (sẽ làm việc như một giao diện mạng thứ hai).

Nếu bạn ở trong mạng 192.168.1.0 thì sẽ thực hiện các lệnh sau:

ifup eth0:1 10.0.0.x (trong đó x là một địa chỉ IP xác định nào đó).

thêm một tuyến (route) tới một mạng mới và gán nó sử dụng thiết bị eth0:1

-- thêm một tuyến (route) tới một mạng khác bằng cách sử dụng một máy làm gateway (bạn sẽ cần biết thiết lập eth0 hoặc eth0:1 của gw này phụ thuộc vào việc bạn đang ở mạng nào)

MẠNG TCP/IP

Số nhị phân và Dotted Quad

Số nhị phân

$10 = 2^1$	$100 = 2^2$	$101 = 2^2 + 1$	$111 = 100 + 010 + 001$
------------	-------------	-----------------	-------------------------

Điều này cho thấy một số nhị phân có thể dễ dàng chuyển sang số thập phân:

10000000	=	2^7	=	128
01000000	=	2^6	=	64
00100000	=	2^5	=	32
00010000	=	2^4	=	16
00001000	=	2^3	=	8
00000100	=	2^2	=	4
00000010	=	2^1	=	2
00000001	=	2^0	=	1

The Dotted Quad:

Địa chỉ IP được gán cho một interface được gọi là một Dotted Quad. Trong trường hợp một địa chỉ Ipv.4, địa chỉ là 4 bytes (4 lần 8 bits) phân cách nhau bởi các dấu chấm.

Decimal	Binary
192.168.1.1	11000000.10101000.00000001.00000001

Địa chỉ Broadcast, địa chỉ mạng và netmask

Một địa chỉ IP bao gồm địa chỉ của host và địa chỉ của mạng.

The Netmask

Netmask được dùng để qui định số bit trong một địa chỉ IP được dùng để đánh địa chỉ mạng. Netmask hay còn gọi là subnet mask.

Ví dụ netmask 16 và 17 bit:

255.255.0.0	16-bit	1 1 1 1 1 1 1 1 . 1 1 1 1 1 1 1 1 . 0 0 0 0 0 0 0 0 . 0
255.255.128.0	17-bit	1 1 1 1 1 1 1 1 . 1 1 1 1 1 1 1 1 . 1 0 0 0 0 0 0 0 . 0

Địa chỉ broadcast thường được sinh ra bởi hệ thập phân.

Ví dụ: với 16 – bit netmask, các IP sau nằm trên cùng một mạng

00100000	.	10000000	.	00000001	.	00000001
00100000	.	10000000	.	00000000	.	00000011

Có nghĩa rằng bất kỳ một bit nào **nằm trong** hình chữ nhật (hình vẽ) (8+8 = 16 bits) sẽ thay đổi địa chỉ mạng và các host cần một gateway để kết nối chúng với nhau.

Tương tự, bất kỳ bit nào **bên ngoài** hình chữ nhật (hình vẽ) sẽ thay đổi địa chỉ của host mà không làm thay đổi địa chỉ mạng.

Ví dụ: với netmask 24 bit dưới đây, 2 IP sẽ nằm trên 2 mạng khác nhau:

00100000	.	10000000	.	00000001	.	00000001
00100000	.	10000000	.	00000000	.	00000011

Địa chỉ mạng

Mỗi một mạng cần có một số hiệu, số hiệu cần thiết trong việc thiết lập bộ dẫn đường (routing). Số hiệu của mạng là có số nguyên(0-255) phân cách bởi dấu chấm.

Địa chỉ Broadcast

Địa chỉ broadcast là một miền các host/interface có thể được truy cập trên mạng giống nhau. Ví dụ một host có địa chỉ broadcast là 10.1.255.255 sẽ truy cập đến tất cả các máy nào có IP có dạng 10.1.x.x. Địa chỉ broadcast điển hình 192.168.1.255.

Các phép toán logic có thể áp dụng cho các địa chỉ broadcast, netmask, network.

Để lấy địa chỉ mạng, ta làm động tác đơn giản là thực hiện phép toán AND giữa địa chỉ IP và netmask.

Network Address	=	IP	AND	Netmask
-----------------	---	----	-----	---------

Tính địa chỉ broadcast bằng cách: network address OR 'not MASK'

Broadcast Address	=	Network	OR	not[Netmask]
-------------------	---	---------	----	--------------

AND và OR à các phép toán logic trong mẫu nhị phân của các địa chỉ này

Ví dụ:

Địa chỉ IP **192.168.3.5** với net mask **255.255.255.0**. Chúng ta có thể thực hiện các phép toán sau:

<u>Địa chỉ mạng</u>	=	IP	AND	MASK
		11000000. 10101000.000000011.00000101		(192.168.3.5)

AND

11111111.11111111.11111111.00000000	(255.255.255.000)
-------------------------------------	-------------------

11000000.10101000.000000011.00000000	(192.168.3.0)
--------------------------------------	------------------------

<u>Địa chỉ Broadcast</u>	=	IP	OR	NOT-MASK
--------------------------	---	----	----	----------

11000000. 10101000.000000011.00000101	(192.168.3.5)
---------------------------------------	---------------

OR

00000000.00000000.00000000.11111111	(000.000.000.255)
-------------------------------------	-------------------

11000000.10101000.00000011.11111111 (192.168.3.255)

Từ các ví dụ trên ta rút ra nhận xét. Một địa chỉ IP cùng với netmask đủ để xác định các thông tin về mạng và host đó.

Lớp mạng

Địa chỉ IP dự phòng

Đối với các mạng riêng biệt, các địa chỉ IP có thể không bao giờ được sử dụng làm địa chỉ IP trên internet. Các địa chỉ IP dự phòng này thông thường chỉ được sử dụng cho các mạng LAN.

Bảng sau đây sẽ cho thấy các lớp địa chỉ riêng/ dự phòng.

Bảng1: Địa chỉ dự phòng

1	Class A	10.x.x.x
16	Class B	172.16.x.x -- 172.31.x.x
255	Class C	192.168.o.x

Lớp địa chỉ IP

Lớp A:

8 bit dùng để đánh địa chỉ mạng và 24 bit đánh địa chỉ host. Byte đầu tiên dự phòng cho địa chỉ mạng. Vì vậy subnet mask mặc định sẽ là **255.0.0.0**.

Do 255.255.255 and 0.0.0 không phải là địa chỉ host nên có tối đa $2^{24} - 2 = 16777214$ host trên mạng. Số IP có byte đầu tiên nằm trong miền từ **1** đến **127**, tương ứng với số nhị phân 00000001 -> 01111111. Hai bit đầu tiên của lớp A có thể thiết lập bằng “00” hoặc “01”.

Lớp B: địa chỉ mạng và host 16 bit

16 bit dùng để đánh địa chỉ mạng và 16 dùng để đánh địa chỉ host trên mạng. Subnet mask mặc định là **255.255.0.0**. Có tối đa $2^{16}-2 = 65\,534$ host trên một mạng thuộc lớp B. Byte đầu tiên có phạm vi từ **128** đến **191**. Tương ứng với số nhị phân là 10000000->10111111.

Hai bit đầu tiên của lớp B luôn thiết lập là “10”.

Lớp C: địa chỉ mạng và host 24-bit

24 bit dùng để đánh địa chỉ mạng và 8 bit dùng để đánh địa chỉ host trên mạng. Subnet mask mặc định là **255.255.255.0**. Có tối đa $2^8 - 2 = 254$ host trên một mạng thuộc lớp C. Byte đầu tiên có giá trị từ **192** đến **223**. Tương ứng với số nhị phân là 11000000 -> 11011111. Như vậy 2 bit đầu tiên của lớp C luôn là “11”.

Subnets

Subnet là khái niệm phân chia một mạng thành nhiều mạng con bằng cách dùng các bit của phần địa chỉ host để đánh địa chỉ mạng.

Ví dụ netmask lớp A là 255.0.0.0 có thể được dùng để biến bit đầu tiên của byte thứ 2 trở thành bit đánh địa chỉ mạng. Kết quả chúng ta có 9 bit để đánh địa chỉ mạng và 23 bit đánh chỉ host trên mạng.

Netmask có dạng binary như sau :

11111111.10000000.00000000.00000000 or 255.128.0.0

25-bit network

Netmask: 11111111.11111111.11111111.**10000000** or 255.255.255.128

Do địa chỉ mạng Network = IP AND Netmask, từ giá trị của netmask, ta thấy là có thể tạo được 2 mạng con.

1. Các địa chỉ host nằm trong miền 192.168.1.**0xxxxxxx** thuộc vào mạng 192.168.1.**0** network. Số hiệu của mạng là 0.

2. Các địa chỉ host nằm trong miền 192.168.1.**1xxxxxxx** thuộc vào mạng 192.168.1.**128** network. Số hiệu của mạng là 128

Bảng2: Trong cả 2 trường hợp, thay x byte bằng 0 hoặc 1, ta có các địa chỉ đặc biệt

Network address	Substitute with 1's	Substitute with 0's
0	Broadcast: 127	Network: 0
128	Broadcast: 255	Network: 128

Số bit để đánh địa chỉ host là 7 và trừ đi 2 giá trị đặc biệt (tất cả các bit bằng 0 hoặc 1), chúng ta có $2^7 - 2 = 126$ trên mỗi mạng và có tất cả 252 host.

Nếu chúng ta dùng subnet mask mặc định là 255.255.255.0 thì chúng ta có 254 địa chỉ host.

Trong ví dụ trên 192.168.1.127 là các địa chỉ đặc biệt, do đó chỉ có 252 địa chỉ host được sử dụng.

26-bit network

Netmask: 11111111.11111111.11111111.**11000000** or 255.255.255.192

Tạo được 4 mạng con, địa chỉ của mỗi mạng được xác định bằng qui tắc AND, địa chỉ của các host được xác định như sau:

1. Địa chỉ các host nằm trong miền 192.168.1.**00xxxxxx** thuộc vào mạng 192.168.1.**0** network.
2. Địa chỉ các host nằm trong miền 192.168.1.**01xxxxxx** thuộc về mạng 192.168.1.**64** network.
3. Địa chỉ các host nằm trong miền 192.168.1.**10xxxxxx** thuộc về mạng 192.168.1.**128** network.

4. Địa chỉ các host nằm trong miền 192.168.1.11xxxxxx thuộc về mạng 192.168.1.192 network.

Thay thế x bit trên bằng 1 ta có địa chỉ ở trên ta có các địa chỉ broadcast tương ứng:

192.168.1.63, 192.168.1.127, 192.168.1.191, 192.168.1.255

Mỗi mạng con có $2^6 - 2 = 62$ hosts và tổng số có $62 \times 4 = 248$ host trên mạng.

Họ giao thức TCP/IP

TCP/IP là một bộ giao thức, được sử dụng trên mạng Internet. Gọi là họ giao thức vì TCP/IP chứa một số giao thức, những giao thức này dùng để truyền dữ liệu và chương trình qua mạng. Hai giao thức chính trong họ giao thức TCP/IP là TCP (Transmission Control Protocol) và Ip (Internet Protocol).

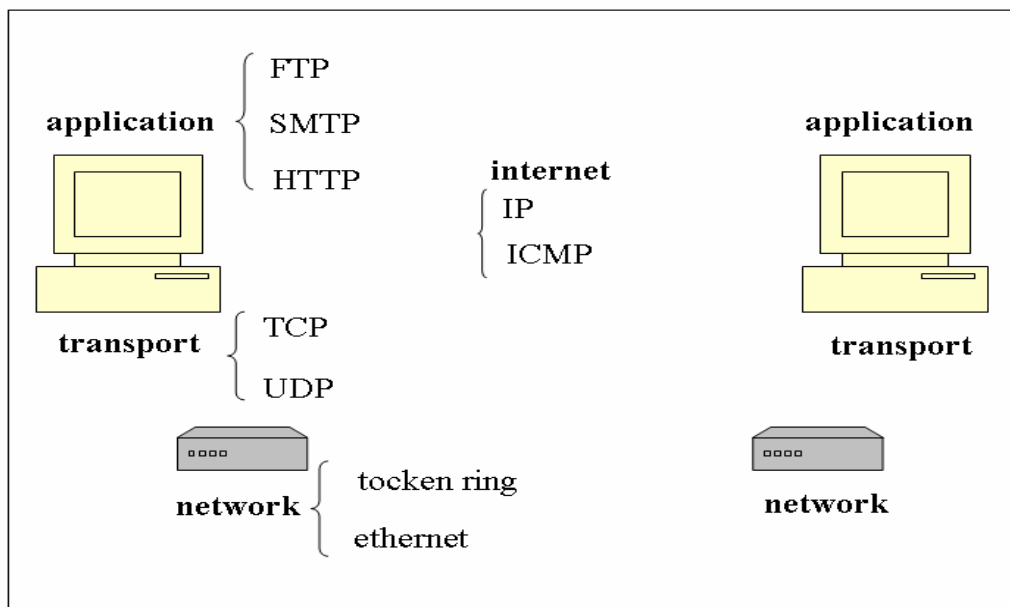
Hiểu một cách đơn giản, giao thức IP chỉ xử lý các gói tin và các datagrams (gói tin chứa địa chỉ đến, kích thước...) trong khi đó giao thức TCP xử lý vấn đề kết nối giữa 2 máy tính. Các giao thức kết hợp với nhau để thực hiện tác vụ đặc biệt của mình. Tài liệu này sẽ trình bày các tác vụ của TCP/IP.

Hoạt động của các giao thức diễn ra ở các tầng khác nhau trong tiến trình hoạt động của mạng.

Bảng 1: Mô hình 4 tầng của giao thức TCP/IP

Tầng ứng dụng (Application)	Mức ứng dụng(FTP,SMTP,SNMP)
Tầng giao vận(Transport)	Kết nối các máy(TCP,UDP)
Tầng internet(Internet)	Routing(Dẫn đường):IP,ICMP,IGMP,ARP

Tầng truy cập mạng()	Mức card mạng, ví dụ card Ethernet, token ring...
----------------------	---



• Tổng quan về các giao thức

IP	Giao thức IP làm nhiệm vụ truyền tải dữ liệu cho giao thức TCP , UDP và ICMP.IP cung cấp dịch vụ kết nối không tin cậy (unreliable), có nghĩa là dữ liệu truyền đi không đảm bảo được truyền đến địa chỉ cần gửi. Giao thức IP cho phép tất cả tính toán vận của dữ liệu được xử lý bởi một trong giao thức tầng cao hơn, ví dụ như giao thức TCP hoặc những thiết bị chuyên biệt cho ứng dụng nào đó. IP có nhiệm vụ xử lý vấn đề địa chỉ và dẫn đường (routing) giữa các mạng. Đơn vị dữ liệu sử dụng ở giao thức IP là datagram.
TCP	Giao thức TCP(Transmission Control Protocol) cung cấp dịch vụ kết nối tin cậy. TCP có nhiệm vụ kiểm tra trên mỗi host thứ tự gửi và nhận và kiểm tra và bảo đảm rằng mọi gói dữ liệu (data packet) đã được truyền. Ví dụ các ứng dụng FTP hay telnet(ứng dụng đăng nhập từ xa) không cần

	phải xử lý vấn đề mất dữ liệu trong quá trình truyền.
UDP	Giao thức UDP (User Datagram Protocol) cho phép một chương trình ứng dụng truy cập trực tiếp đến IP, không giống như TCP, UDP là giao thức không liên kết và không tin cậy.
ICMP	Giao thức ICMP(Internet Control Message Protocol) được sử dụng bởi các thiết bị dẫn đường và các host để theo dõi trạng thái của mạng. Đơn vị dữ liệu sử dụng trong giao thức này là IP datagrams và ICMP là giao thức không liên kết.
PPP	Giao thức PPP(Point to Point) thiết lập một kết nối TCP/IP thông qua đường điện thoại. Ngoài ra nó còn được sử dụng bên trong các kết nối được mã hóa như pptp.

Các dịch vụ và các cổng trong TCP/IP

Danh sách các dịch vụ và các cổng của nó nói chung sẽ tìm thấy trong /etc/services. Danh sách các dịch vụ và các cổng tương ứng với các dịch vụ được quản lý bởi IANA(Internet Assigned Numbers Authority).

Mỗi cổng là một số 16 bit, đó có tổng số là 65535 cổng. Các cổng từ 1 đến 1023 là các cổng độc quyền, được giành cho các dịch vụ chạy bởi người dùng root. Tất cả các ứng dụng đã biết sẽ được phục vụ ở một trong những cổng này.

Chúng ta hãy quan sát kết quả của dịch vụ portscans(dò tìm các cổng). Nên nhớ rằng dịch vụ này là bất hợp pháp, tuy nhiên rất nhiều người dùng dịch vụ này.

Dưới đây là kết quả của lệnh quét cổng:

Port	State	Service
21/tcp	open	ftp
22/tcp	open	ssh
23/tcp	open	telnet
25/tcp	open	smtp

Quản trị Hệ thống Linux - Cơ bản

70/tcp	open	gopher
79/tcp	open	finger
80/tcp	open	http

Dịch vụ portscan cho biết các cổng đang mở và phục vụ ứng dụng nào

Các cổng chính /etc/services:

ftp-data	20/tcp		
ftp	21/tcp		
telnet	23/tcp		
smtp	25/tcp	mail	
domain	53/tcp		
domain	53/udp		
http	80/tcp		# www is used by some broken
www	80/tcp		# progs, http is more correct
pop-2	109/tcp		# PostOffice V.2
pop-3	110/tcp		# PostOffice V.3
sunrpc	111/tcp		
sftp	115/tcp		
uucp-path	117/tcp		
nntp	119/tcp	usenet	# Network News Transfer
ntp	123/tcp		# Network Time Protocol
netbios-ns	137/tcp	nbns	
netbios-ns	137/udp	nbns	
netbios-dgm	138/tcp	nbdgm	
netbios-dgm	138/udp	nbdgm	
netbios-ssn	139/tcp	nbssn	
imap	143/tcp		# imap network mail protocol
NeWS	144/tcp	news	# Window System
snmp	161/udp		
snmp-trap	162/udp		

Thực hành

Registering a service with xinetd

1. Viết một bash script đưa ra màn hình(stdout) dòng “Welcome”. Lưu lại trong /usr/sbin/hi

2. Trong thư mục /etc/xinetd.d tạo một file tên là fudge như sau:

```
service fudge
{
    socket_type      = stream
    server           = /usr/sbin/hi
    user             = root
    wait             = no
    disable          = no
}
```

3. Thêm một dịch vụ tên là fudge trong /etc/services, dịch vụ này sử dụng cổng 60000.
4. Khởi động lại **xinetd** và dùng dịch vụ telnet đến cổng 60000.
5. Giả sử bạn có một miền IP trên mạng 83.10.11.0/27
 - a. Bao nhiêu mạng có 4 byte đầu tiên giống như của bạn?
 - b. Có bao nhiêu máy trên mạng của bạn?Có bao nhiêu địa chỉ broadcast cho mạng đầu tiên này?

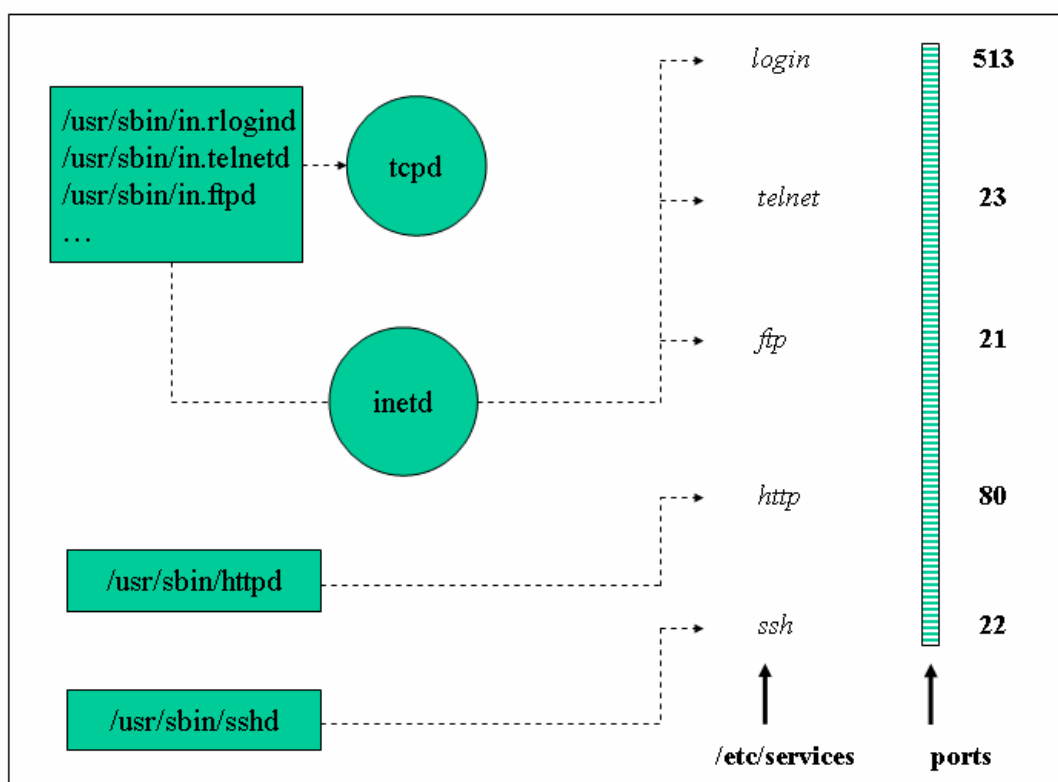
CÁC DỊCH VỤ MẠNG

Các dịch vụ mạng có thể chạy đồng thời hoặc đơn lẻ như các ứng dụng, chúng làm nhiệm vụ lắng nghe (listen) các kết nối và trực tiếp điều khiển các client hoặc chúng cũng có thể được gọi bởi các tiến trình nền mạng (network daemon) **inetd** hoặc **xinetd**.

Tiến trình nền inetd (cũ)

Tiến trình nền này sẽ được thực hiện tại thời điểm khởi động hệ thống và có nhiệm vụ lắng nghe (listen) các kết nối tại các cổng (port) được xác định trước. Điều này cho phép máy chủ chỉ chạy một tiến trình nền mạng nào đó (network daemon) khi cần thiết.

Ví dụ, dịch vụ **telnet** có một tiến trình nền **/usr/sbin/in.telnetd** sẽ kiểm soát các tiến trình telnet. Để lúc nào cũng chạy tiến trình nền này **inetd** được chỉ định lắng nghe cổng 23. Chỉ định này được thiết lập trong **/etc/inetd.conf**.



Hình 1: Tiến trình nền inetd

Các trường của **/etc/inetd.conf** chứa các thông tin sau:

service-name	tên hợp lệ từ /etc/services
socket type	stream đối với TCP và dgram đối với UDP
protocol	giao thức hợp lệ từ /etc/protocols
flag	nowait nếu đa tiến trình (multithreaded) và wait nếu đơn tiến trình (single-threaded)
user/group	chạy chương trình như user hoặc group
program	tcpd thông thường
argument	tên của chương trình chạy đối với dịch vụ này

Ví dụ:

```
pop-3 stream tcp nowait root /usr/sbin/tcpd ipop3d
```

Chú ý: File **/etc/services** được sử dụng để tạo sự tương quan giữa tên dịch vụ và số cổng socket. Các trường trong file services là:

<code>service-name</code> <code>port/protocol</code> <code>[aliases]</code>

Tiến trình nền xinetd

Đây là phiên bản mới nhất của **inetd**. Tiến trình nền **tcpd** không còn được sử dụng nữa, do đó tất cả mọi thứ đều được thực hiện bởi **xinetd**. Cấu hình của **xinetd** được thực hiện qua một file đơn **/etc/xinetd.conf** hoặc bằng cách soạn thảo các file riêng biệt trong **/etc/xinetd.d/** tương ứng với các dịch vụ sẽ được kiểm soát bởi **xinetd**. Cũng có thể chuyển đổi từ file cấu hình **inetd** cũ sang các file cấu hình của **xinetd** hiện thời.

Cấu trúc của file service trong xinet.d

```
Service-name {  
    socket_type = stream đối với TCP và dgram đối với UDP  
    protocol = giao thức phù hợp từ /etc/protocols  
    wait = <yes hoặc no>  
    user= người dùng chạy ứng dụng  
    group= nhóm của người dùng chạy ứng dụng  
    server= tên của chương trình chạy của dịch vụ này  
}
```

TCP wrappers

Nếu các chương trình đã được biên dịch với libwrap thì chúng có thể được liệt kê trong **/etc/host.allow** và **/etc/host.deny**. Thư viện **libwrap** sẽ xác định những file nào tương ứng với hosts nào.

Định dạng mạng định đối với **/etc/hosts.{allow,deny}**:

```
DAEMON : hosts [EXCEPT hosts ] [: spawn command]
```

Bạn cũng có thể sử dụng những file này để ghi log các dịch vụ không xác thực (unauthorised services). Đây được xem như sự cảnh báo sớm của hệ thống. Sau đây là một số ví dụ:

Truy vấn thông tin về máy chủ (host):

4. /etc/hosts.allow

in.telnetd: LOCAL, .my.domain

5. /etc/hosts.deny

in.telnetd: ALL : spawn (/usr/sbin/safe_finger -l @%h | mail root) &

Chuyển tới một dịch vụ giả (bogus service)

6. /etc/hosts.allow

in.telnetd: ALL : twist /dtk/Telnetd.pl

Ví dụ cuối cùng nằm trong bộ công cụ mẹo (Deception Tool kit) và có thể download tại địa chỉ sau: <http://all.net/dtk/download.html>

Thiết lập NFS

Thiết lập phía máy trạm

Đối với các máy trạm Linux muốn gắn (mount) các file hệ thống từ xa (remote file system):

1. file hệ thống **nfs** phải được hỗ trợ bởi nhân
2. tiến trình nền **portmapper** phải đang được chạy.

Tiến trình nền portmapper được khởi động bởi script **/etc/rc.d/init.d/portmap**. Tiện ích **mount** sẽ gắn file hệ thống. Các đầu vào thông thường trong **/etc/fstab** sẽ là:

```
nfs-server:/shared/dir /mnt/nfs nfs defaults 0 0
```

Thiết lập phía máy chủ

Một máy chủ NFS cần phải chạy **portmap** trước khi khởi động máy chủ nfs. Máy chủ nfs sẽ được khởi động hoặc dừng với script **/etc/rc.d/init.d/nfs**.

File cấu hình chính là **/etc/exports**.

Ví dụ file /etc/exports:

```
/usr/local/docs *.local.org(rw, no_root_squash) *(ro)
```

Thư mục được kết xuất (export) tới tất cả các máy chủ (host) theo quyền chỉ đọc (read-only) và đọc – ghi (read – write) tới tất cả các máy chủ (host) trong miền .local.org

Quản trị Hệ thống Linux - Cơ bản

Tham số lựa chọn mặc định `root_squash` sẽ ngăn ngừa người dùng root (root user - uid = 0) trên máy khách truy cập vào vùng chia sẻ trên máy chủ và có thể được thay đổi bởi tham số lựa chọn `no_root_squash`.

☞ File **/etc/exports** sẽ tương ứng với các host như *.machine.com trong khi /etc/hosts.allow/deny tương ứng với các host như .machine.com

Nếu file /etc/exports đã được thay đổi thì tiện ích **exportfs** sẽ được chạy. Nếu các thư mục tồn tại trong /etc/exports bị thay đổi thì nó có thể cần thiết để tháo (unmount) tất cả các chia sẻ nfs trước khi chúng được gán lại (remount). Các thư mục riêng rẽ có thể được gán hoặc tháo (unmount) với **exportfs**.

Kết xuất và dừng kết xuất (unexporting) tất cả thư mục trong /etc/exports:



```
exportfs -ua ; exportfs -a
```

SMB và NMB

Các máy Linux có thể truy cập và cung cấp các nguồn tài nguyên chia sẻ của Window (thư mục và máy in). Giao thức được dùng để làm việc này là MS Windows Server Message Block **SMB**. Trong Linux công cụ Samba thường được sử dụng để hỗ trợ cho phần mềm khách và chủ.

Từ cửa sổ dòng lệnh

Tiện ích **smbclient** được sử dụng để liệt kê tất cả nguồn tài nguyên được chia sẻ. Các thư mục từ xa (remote directories) thông thường được gán với **smbmount**, tuy nhiên 'mount -t smbfs' cũng có thể được sử dụng.

Ví dụ:

Quản trị Hệ thống Linux - Cơ bản

Gửi một thông báo pop up tới một máy tính win98desk



```
smbclient -M win98desk
```

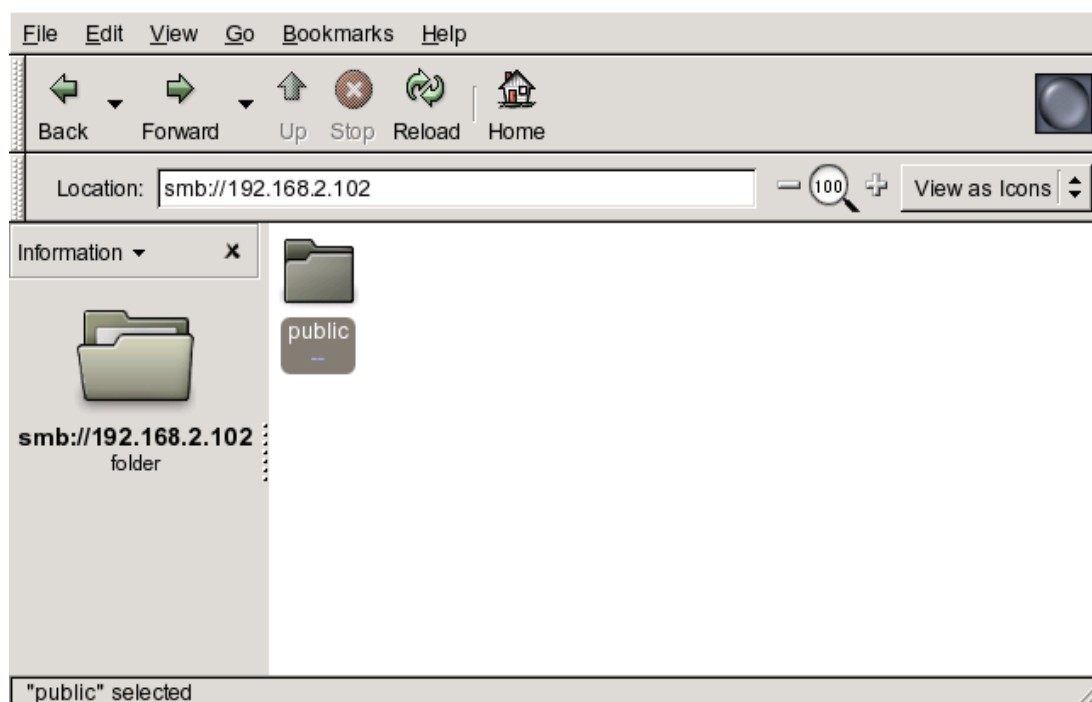
Gán một thư mục chia sẻ của máy chủ winserv



```
smbmount //winserver/shared /mnt/winserver/shared
```

Máy chủ Samba có thể được cấu hình với file **/etc/smb.conf** và được khởi động hoặc dừng với script **/etc/rc.d/init.d/smb**. Chú ý **smb** sẽ cũng khởi động các dịch vụ **NBS**. Khởi thông báo NetBIOS (NetBIOS Message Block) sau đây sẽ cho phép giải tên (name resolution) trong Windows.

Hình 1: Nautilus Browsing SMB shares::



Các đầu vào chính trong **/etc/smb.conf**:

```
[global]
    workgroup = LINUXIT
    os level = 2
    kernel oplocks = No
    security = user
```

```
encrypt passwords = Yes
guest account = nobody
map to guest = Bad User

[homes]
comment = Home Directories
read only = No
create mask = 0640
directory mask = 0750
browseable = No

[printers]
comment = All Printers
path = /var/tmp
create mask = 0600
printable = Yes
browseable = No
```

Cấu hình SWAT và Webmin GUI

Nếu cài đặt gói **swat** thì bạn có thể quản trị máy chủ samba qua nền web GUI tại cổng 901.

Một công cụ quản trị phổ thông khác được sử dụng là **webmin**. Công cụ này có thể được tải về tại địa chỉ www.webmin.com

CHÚ Ý

File cấu hình `/etc/samba/smb.conf` là một nguồn tài liệu tốt. Tất cả các tham số lựa chọn được mô tả và có thể được chuyển thành câu lệnh bằng cách xóa dấu ghi chú `;`. Có thể xem trong trang hướng dẫn **smb.conf(5)**.

Các dịch vụ DNS

- **Bộ phân giải địa chỉ (Resolvers)**

Khi một chương trình cần giải một tên host thì cần sử dụng một cơ chế gọi là bộ giải (resolver). Bộ giải đầu tiên sẽ tra cứu file `/etc/nsswitch` (trước `/etc/host.conf`) và xác định phương thức nào sẽ được sử dụng để giải các tên host (local file, name server, NIS hay ldap server).

File /etc/host.conf (hoặc /etc/nsswitch.conf):

Các file này được quét bởi bộ giải tên để xác định xem đâu là các file, máy chủ dns, cơ sở dữ liệu ldap hoặc máy chủ nis sẽ được tra cứu.

Ví dụ (/etc/nsswitch):

```
hosts:      files dns nis
networks:   files
```

Dòng đầu tiên cho thấy các file (ở đây là /etc/hosts) sẽ được truy vấn đầu tiên và sau đó là máy chủ DNS nếu nó bị lỗi. Dòng thứ hai chỉ dẫn sẽ sử dụng file /etc/networking cho thông tin về mạng.

File /etc/hosts

Với một số nhỏ các máy tính được nối mạng thì có thể chuyển đổi địa chỉ IP thành tên bằng cách sử dụng file /etc/hosts. Các trường có thể là:

IP	machine	machine.domain	alias
----	---------	----------------	-------

Ví dụ: file /etc/hosts

192.168.1.233	io	io.my.domain	
61.20.187.42	callisto	callisto.physics.edu	

File /etc/resolv.conf

Nếu bộ giải cần sử dụng một máy chủ tên miền (DNS) thì nó sẽ tra cứu danh sách các máy chủ hiện có tại file /etc/resolv.conf

- **Cấu trúc có cấp bậc**

Các máy chủ tên (Name servers) đều có một cấu trúc cấp bậc (hierachical structure). Phụ thuộc vào vị trí trong tên miền điều kiện đầy đủ (fully qualified domain name – FQDM) mà một tên miền có thể được gọi là mức top – level, mức thứ hai (second level) hoặc mức thứ ba (third level).

Ví dụ đối với các tên miền cấp 1 (top level)

com	Các tổ chức thương mại
edu	Các tổ chức giáo dục Mỹ
gov	Các tổ chức chính phủ Mỹ
mil	Các tổ chức quân sự Mỹ
net	Các nhà cung cấp dịch vụ và cổng truy cập
org	Các trang phi thương mại
uk	Các trang thuộc về nước Anh

- **Kiểu của Máy chủ DNS**

Các tên miền có thể được chia nhỏ hơn thành các tên miền con (subdomain). Điều này sẽ giới hạn tổng số thông tin cần để quản trị trong một miền. Mỗi vùng (Zone) sẽ có một máy chủ tên miền chính (thường gọi là **primary** DNS) và một hoặc nhiều máy chủ tên miền phụ (thường gọi lại **secondary**). Việc quản trị máy chủ tên gồm có việc cập nhật thông tin về một vùng cụ thể. Máy chủ chính thường được ra lệnh cho việc xác thực.

- **File cấu hình DNS**

Trong phiên bản BIND cũ (trước phiên bản BIND 8) file cấu hình là **/etc/named.boot**. Với BIND phiên bản 8, file **/etc/named.conf** được thay thế. Bạn có thể sử dụng tiện ích **named-bootconf.pl** để chuyển đổi từ file cấu hình cũ sang file cấu hình mới.

File /etc/named.boot:

directory	/var/named
cache	named.ca
primary myco.org	named.myco
primary 0.0.127.in-addr.arp	named.local
primary 1.168.192.in-addr.arp	named.rev

Dòng đầu tiên định nghĩa thư mục cơ sở được sử dụng. File name.ca sẽ chứa danh sách các địa chỉ IP DNS cho việc truy vấn các địa chỉ mở rộng. Dòng thứ ba là tham số lựa chọn và chứa các bản ghi cho mạng nội bộ. Hai tham số tiếp theo được sử dụng cho tìm kiếm ngược lại (reverse lookup).

Trong **/etc/named.conf**

<i>cache</i>	được thay thế bởi <i>hint</i>
<i>secondary</i>	được thay thế bởi <i>slave</i>
<i>primary</i>	được thay thế bởi <i>master</i> .

Áp dụng các thay đổi này đối với file cấu hình BIND4 sẽ sinh ra các file cấu hình BIND8 và BIND9 như sau.

File /etc/named.conf:

```
options {  
    directory "/var/named";  
};  
  
zone "." {  
    type hint;  
    file "named.ca";  
};  
  
zone "myco.org" {  
    type master;  
    file "named.myco";  
};  
  
zone "1.168.192.in-addr.arp" {  
    type master;  
    file "named.rev";  
};  
  
zone "0.0.127.in-addr.arpa" {  
    type master;  
    file "named.local";  
};
```

- **File vùng DNS**

Trong ví dụ này máy chủ được thiết lập như một máy chủ chỉ bắt (catching-only server). Tất cả các file vùng (zone file) đều chứa các bản ghi tài nguyên.

Ví dụ file **named.local** zone file:

```
@    IN    SOA    localhost. root.localhost. (
                                2001022700 ; Serial
                                28800      ; Refresh
                                14400      ; Retry
                                3600000    ; Expire
                                86400 )   ; Minimum
      IN    NS    localhost.
1     IN    PTR    localhost.
```

Đây là một file vùng rất đơn giản nhưng nó cung cấp đầy đủ các thông tin để giúp ta hiểu được cơ chế cơ bản của một máy chủ tên.

Ký hiệu @ sẽ giải (tham chiếu) tới một vùng liên quan được khai báo trong **/etc/named.conf**. Điều này cho phép bất kỳ file vùng nào cũng có thể sử dụng như là một template cho các vùng khác (xem bài tập).

Bảng 1: Kiểu bản ghi thông thường

NS	Xác định các vùng của máy chủ tên miền chính
PTR	Tham chiếu ngược địa chỉ IP tới tên máy host
MX	Bản ghi thư điện tử Mail Exchange
A	Tương ứng một địa chỉ IP với một máy host
CNAME	Tương ứng một tên gán (alias) với một tên chính của máy host

Bảng 2: Các tham số vùng

@ IN SOA	Start Of Authority. Xác định một vùng được cho phép bởi các tham số lựa chọn nằm trong dấu ngoặc kép
serial	Giá trị được tăng bằng tay khi dữ liệu thay đổi. Các máy chủ phụ (secondary servers) sẽ truy vấn số hiệu (serial number) của máy chủ chính. Nếu nó thay đổi, toàn bộ file vùng sẽ được tải về (downloaded)
refresh	Thời gian được tính bằng giây trước khi máy chủ phụ truy vấn bản ghi SOA của tên miền chính (primary domain). Giá trị của nó nhỏ nhất là một ngày.
retry	Khoảng thời gian tính bằng giây trước khi một vùng mới được chuyển (transfer) nếu việc download trước đó lỗi.
expire	Thời gian sau khi máy chủ phụ loại bỏ tất cả dữ liệu vùng nếu nó liên hệ với máy chủ chính. Giá trị của tham số này thông thường ít nhất là 1 tuần
minimum	đây là ttl đối với các dữ liệu đã được cached. Giá trị mặc định là 1 ngày (86400 giây) nhưng cũng có thể lâu hơn đối với các mạng LAN ổn định

• Cấu hình Sendmail

Sendmail là dịch vụ chuyển mail (MTA) phổ biến nhất trên internet. Nó sử dụng giao thức Simple Mail Transfer Protocol (SMTP) và chạy như một tiến trình nền lắng nghe các kết nối tại cổng 25.

Script **Sendmail** được dùng để dừng hoặc chạy tiến trình nền sendmail thông thường được đặt tại thư mục **/etc/rc.d/init.d/**. Cấu hình chính của file là **/etc/mail/sendmail.cf** (hoặc **/etc/sendmail.cf**) Tại đây bạn có thể xác định tên của máy chủ cũng như tên của các host mà từ đó và ở đó mail relay được cho phép.

File **/etc/aliases** chứa hai trường sau đây:

```
alias: user
```

Khi chuyển tới **/etc/aliases**, câu lệnh **newaliases** phải được chạy để rebuild cơ sở dữ liệu **/etc/aliases.db**.

Khi thư được máy chủ chấp nhận, nó sẽ được móc vào một file đơn với tên do người dùng đặt. Các file này được lưu trữ tại **/var/spool/mail**. Phụ thuộc vào Mail User Agent được sử dụng, người dùng có thể lưu trữ các thông điệp (message) trong thư mục gốc của mình hoặc có thể download chúng về một máy khác.

Nếu máy chủ đang chuyển tiếp (relaying), hoặc nếu mạng chậm và nhiều message đang được chuyển, thư sẽ được lưu trữ trong hàng đợi thư **/var/spool/mqueue**. Bạn có thể truy vấn với tiện ích **mailq** hoặc **sendmail -bp**. Quản trị mạng có thể flush hàng đợi của máy chủ với câu lệnh **sendmail -q**.

Cuối cùng, để đăng ký một tên miền như một địa chỉ email hợp lệ, một bản ghi MX cần được thêm vào trong cơ sở dữ liệu DNS.

Ví dụ nếu **mail.company.com** là một máy chủ mail, để nó chấp nhận mail như **joe@company.com** thì bạn sẽ phải cấu hình như sau

1. Thêm **company.com** vào **/etc/mail/local-host-names**
2. **company.com MX 10 mail.company.com** trong một file vùng DNS

Máy chủ Apaches

- **File cấu hình**

File **/etc/httpd/conf/httpd.conf** chứa tất cả các tham số thiết lập cấu hình

Các phiên bản trước của apache có thêm hai file ngoài, một là **access.conf** trong đó sẽ giới hạn các thư mục đã được khai báo và một file khác là **srml.conf** xác định thư mục gốc (roôt) của máy chủ.

Các cấu hình cần chú ý:

```
ServerType standalone/inetd
ServerRoot "/etc/httpd"
DocumentRoot "/var/www/html"

<Directory "/var/www/cgi-bin">
    AllowOverride None
    Options ExecCGI
    Order allow,deny
    Allow from all
</Directory>
```

```
<VirtualHost 122.234.32.12>
    DocumentRoot "/www/docs/server1"
    ServerName virtual.mydomain.org
</VirtualHost>
```

- **Chạy Apache**

Để chạy và dừng máy chủ, đầu tiên bạn có thể sử dụng script **/etc/rc.d/init.d/httpd**. Trên một máy chủ bận (busy server) thì nên sử dụng **apachectl** đặc biệt với lựa chọn **graceful** sẽ khởi động lại máy chủ chỉ khi các kết nối hiện tại đã được thoả thuận.

Các file nhật ký chính được lưu trong **/var/log/httpd/**. Các file này có thể rất hữu ích trong các lý do an ninh. Thông thường chúng ta kiểm tra file `error_log` và `access_log`.

Thực hành

Cài đặt một máy chủ DNS chính

Như là một bài tập, chúng ta sẽ cài đặt gói BIN9 rpm **bind9-9.1.3-252.i386.rpm** và cấu hình một domain có tên là **gogo.com**.

1. Tiến hành lần lượt các bước sau trong **/etc/named.conf**:

Copy/Paste các đoạn sau và sửa lại như sau

<pre>zone "localhost" in { type master; file "localhost.zone"; }</pre>	<i>becomes</i>	<pre>zone "gogo.com" in { type master; file "gogo.zone"; }</pre>
<pre>zone "0.0.127.in-addr.arpa" in { type master; file "127.0.0.zone"; };</pre>	<i>becomes</i>	<pre>zone "2.168.192.in-addr.arpa" in { type master; file "192.168.2.zone"; };</pre>

2. Trong **/var/named**:

```
cp 127.0.0.zone 192.168.2.zone  
cp local.zone gogo.zone
```

3. Thay đổi các trường tương ứng trong file vùng mới (zone file). Thêm một host có tên là *harissa*.

4. Thêm dòng “nameserver 127.0.0.1” vào **/etc/resolv.conf**.

5. Sử dụng **host** để giải *harissa.gogo.com*

Quản trị Apache

Các cấu hình cơ bản trong file **/etc/httpd/conf/httpd.conf**

1. Thay đổi **Port** từ **80** thành **8080**.

2. Kiểm tra rằng apache trả lời với câu lệnh **telnet localhost 8080**. Bạn sẽ nhận được:

```
Trying 127.0.0.1...  
Connected to localhost.linuxit.org.
```

Quản trị Hệ thống Linux - Cơ bản

Escape character is '^]'.
Tiếp theo gõ **GET /** để download file index .

3. Thiết lập “**StartServer**” thành 15. Khởi động lại **httpd** và kiểm tra rằng 15 tiến trình sẽ được chạy (thay vì 8 tiến trình như mặc định)

IP based virtual server

Card mạng ethernet của bạn phải định danh tới một địa chỉ IP mới (gọi là *new-IP*)

```
ifconfig eth0:0 new-IP
```

Thêm các đoạn sau đây vào **/etc/httpd/conf/httpd.conf**:

```
<VirtualHost new-IP>
DocumentRoot /var/www/html/virtual
ServerName www1
</VirtualHost>
```

Cài đặt một thư mục chia sẻ SMB (shared SMB directory)

Trong hầu hết các trường hợp bạn sẽ không cần thêm người dùng smb (smbusers) vào hệ thống. Đơn giản chỉ cần soạn thảo file **smb.conf** và thêm như sau:

```
[public]
    comment = Example Shared Directory
    path = /home/samba
    guest ok = yes
    writeable = yes
```

Cài đặt một máy in chia sẻ:

```
[global]
--- snip ---
printcap name = /etc/printcap
    load printers = yes

[printers]
    comment = All Printers
    path = /var/spool/samba
    browseable = no
# Set public = yes to allow user 'guest account' to print
```

Quản trị Hệ thống Linux - Cơ bản

```
guest ok = yes  
writable = no  
printable = yes
```


BASH SCRIPTING

Môi trường bash

Biến

Khi bạn gõ câu lệnh tại dấu nhắc của chương trình bash shell thì nó sẽ sử dụng biến **PATH** để tìm xem bảng thực hiện (executable) nào trong hệ thống mà bạn cần chạy. Bạn có thể kiểm tra giá trị của biến path bằng cách sử dụng lệnh echo:

```
echo $PATH  
  
/usr/bin:/bin:/usr/sbin:/usr/X11R6/bin:/usr/local/bin:/sbin:/usr/local/sbin/
```

Thực tế, có rất nhiều biến cần thiết đối với shell để cung cấp đối với mỗi môi trường người dùng. Ví dụ các biến **PWD**, **HOME**, **TERM** và **DISPLAY**.

Cú pháp để khởi tạo và khai báo một biến như sau:

```
VARIABLE=VALUE
```

Chú ý rằng không được đặt bất kỳ dấu cách nào xung quanh dấu '='. Khi một biến được khai báo và khởi tạo, nó sẽ có thể được tham chiếu bằng cách sử dụng ký tự dolla đằng trước như ví dụ sau đây:

```
echo $VARIABLE
```

Khi một phiên shell được bắt đầu, một số các tệp cấu hình được đọc và hầu hết các biến được thiết lập.

Để giải phóng một biến khỏi giá trị hiện thời, sử dụng **unset**.

Các file cấu hình

Đầu tiên có thể phân biệt các file cấu hình xem file nào sẽ được đọc đối với mỗi phiên bash mới.

File cấu hình Login:

Các file được đọc khi login là **/etc/profile** và **~/.bash_profile** (bash sẽ tìm một số file khác như **~/.profile**).

Tiếp theo bash sẽ đọc các file điều khiển thời gian của nó **~/.bashrc** và (nếu tồn tại) **/etc/bashrc**.

Các File bashrc:

Các file này được đọc mỗi lần khi một phiên shell được khởi chạy (ví dụ một xterm mới). Các file này là **/etc/bashrc** và **~/.bashrc**.

Các định danh (alias) và các function có thể được ghi trong **~/.bashrc**

Cú pháp Function:

```
function-name ()  
{  
    command1;  
    command2;  
}
```

Bạn có thể kiểm tra xem những file nào sẽ được đọc bằng cách thêm một dòng `echo Profile` trong **/etc/profile**, kiểu:

<code>bash</code>	Không profile nào được đọc, bạn không thấy gì hết
<code>bash -login</code>	Sẽ bắt bash đóng vai trò như một login bash, từ
	<code>Profile</code> sẽ được hiển thị.

Các câu lệnh sau đây sẽ điều khiển cách thức mà bash bắt đầu:

```
bash -norc  
bash -noprofile
```

Chú ý bất kỳ phiên bash mới nào cũng sẽ kế thừa các biến của cha đã được khai báo trong `/etc/profile` và `~/.bash_profile`.

Các yếu tố Scripting

File script

Script shell là một danh sách các chỉ dẫn được lưu trữ trong một tệp phẳng (flat file). Chỉ có hai chỉ dẫn sau là cần thiết.

1. Dòng đầu tiên của script phải là **`#!/bin/bash`** (đối với script bash)
2. File phải có thể đọc và chạy được (ví dụ đối với quyền 755)

Nếu các dòng này không hiện hữu thì cũng có thể chạy chương trình script bằng các gõ:

```
bash program-name
```

Truyền biến vào script (Passing variables to the script)

Các biến được tạo tại các dòng lệnh được tham chiếu bên trong script như \$1 đối với đối số đầu tiên, \$2 cho đối số thứ hai, vv ...

Ví dụ script, mycat:

```
#!/bin/bash
cat $1
```

Script này đòi hỏi một đối số là một file và sẽ hiển thị nội dung của file bằng cách sử dụng **cat**. Để chạy script này trong file `lilo.conf`, bạn sẽ chạy:

```
./mycat /etc/lilo.conf
```

Quản trị Hệ thống Linux - Cơ bản

Một cách khác để chuyển các biến vào script là đặt dấu nhắc script để cho người dùng nhập đầu vào. Cách này có thể thực hiện bằng cách sử dụng câu lệnh **read**. Tên mặc định của biến được đọc là **REPLY**. Sau đây là một script đã được thay đổi:

Chuyển biến tương tác:

```
#!/bin/bash
echo -n "Which file shall I display ?"
read
cat $REPLY
```

hoặc

```
read -p "File to display: " FILENAME
cat $FILENAME
```

Các biến đặc biệt

Các biến đặc biệt chỉ có thể được tham chiếu và được tự động thiết lập bởi bash. Sau đây là một số biến đặc biệt thông dụng nhất:

\$*	Liệt kê tất cả các biến được nhập tại dòng lệnh
\$#	Số lượng các đối số được nhập tại dòng lệnh
\$0	Tên của script
\$_	PID của câu lệnh nền gần nhất
\$\$	PID của shell hiện tại
\$?	Mã thoát của dòng lệnh cuối cùng

Đối với các tham số vị trí \$1, \$2 vv ... phép toán dịch chuyển **shift** sẽ đặt lại tên mỗi tham số một cách tuần hoàn theo cách sau.

\$2 sẽ thành \$1

\$3 sẽ thành \$2 ... vv

Có thể tổng quát lại như sau **\$(n+1) → \$n**

Tính toán logic

Các biểu thức logic được ước lượng với câu lệnh **test** hoặc dấu **[]**. Trong cả hai trường hợp này, kết quả đều được lưu trữ trong biến **\$?** như:

if biểu thức true then **\$?** là 0

if biểu thức false then **\$?** Không là 0

Sau đây là một số ví dụ minh họa:

sử dụng test	Sử dụng []	giải nghĩa
Test -f /bin/bash	[-f /bin/bash]	test nếu /bin/bash là một file
test -x /etc/passwd	[-x /bin/passwd]	test nếu /etc/passwd là một tệp thi hành

Cũng có thể ước lượng nhiều hơn một biểu thức tại cùng một thời điểm bằng cách sử dụng các phép toán logic **||** (OR) và **&&** (AND) trong một dòng lệnh. Ví dụ chúng ta có thể test nếu **/bin/bash** là một tệp thực thi và tồn tại trong **/etc/inittab**:

```
test -x /bin/bash && test /etc/inittab  
[ -e /bin/kbash ] || [ -f /etc/passwd ]
```

Cũng tương tự như vậy khi sử dụng cờ **-o** and **-a** trong phép toán **test**

```
test -x /bin/bash -a -f /etc/inittab  
[ -e /bin/kbash -o -f /etc/passwd ]
```

Vòng lặp

if then loop

Cú pháp: if CONDITION ; then
 command1
 command2
 fi

```
#!/bin/bash
```

```
if [ -x /bin/bash ] ; then
```

```
echo "The file /bin/bash is executable"
```

```
fi
```

if then else

Cú pháp: if CONDITION ; then
 command1
 command2
 else
 command3
 fi

vòng lặp while

Cú pháp: while CONDITION is **true**; do
 command
 done

Ví dụ: Aligne 10 hashes (#) then exit

```
#!/bin/bash
COUNTER=0
while [ $COUNTER -lt 100 ]; do
    echo -n "#"
    sleep 1
    let COUNTER=COUNTER+1
done
```

Vòng lặp Until

Cú pháp: until CONDITION is **false**; do
 command
 done

Ví dụ: Giống như trên, kiểu C tăng đối với mỗi COUNTER

```
#!/bin/bash
COUNTER=20
until [ $COUNTER -lt 10 ]; do
    echo -n "#"
    sleep 1
```

Quản trị Hệ thống Linux - Cơ bản

```
let COUNTER-=1
done
```

Vòng lặp for

Cú pháp

```
for VARIABLE in SET; do
    command
done
```

Ví dụ: tập 'SET' có thể là các dòng của một file

```
#!/bin/bash
for line in `cat /etc/lilo.conf`; do
    IMAGE=$(echo $line | grep image)
    if [ "$IMAGE" != "" ]; then
        echo Kernel configured to boot: $line
    fi
done
```

Nhập dữ liệu từ dòng lệnh

Giả sử rằng script đợi người dùng nhập giá trị đầu vào, phụ thuộc vào kết quả trả lời, phần còn lại của chương trình sẽ thực hiện một số việc một cách phù hợp. Có hai cách để thực hiện điều này là: **select** và **case**.

Sử dụng case

Cú pháp:

```
case $VARIABLE in
    CHOICE command ;;
    CHOICE command ;;
esac
```

Sử dụng select

Cú pháp:

```
select VARIABLE in SET; do
    if [ $VARIABLE = CHOICE ]; then
        command
    fi
    if [ $VARIABLE = CHOICE ]; then
```



```
command
fi
done
```

Làm việc với số

Trong khi các xử lý các chuỗi ký tự một cách liên mạch, một có gắng nhỏ khác là thực hiện một số phép toán số học hết sức cơ bản.

Các phép toán nhị phân

Cộng hoặc nhân các số có thể được thực hiện bằng các sử dụng cả biểu thức **expr** hoặc cấu trúc **\$(())s**.

Ví dụ:

```
expr 7 + 3; expr 2 \* 10; expr 40 / 4; expr 30 - 11
$((7+3)); $((2*10)); $((40/4)); $((30-11))
```

Các giá trị so sánh

Các phép toán kiểm tra:

Số	Xâu
-lt	<
-gt	>
-le	<=
-ge	>=
-eq	=
-ne	!=

Thực hành

1. Trên dòng lệnh xuất biến TEST

```
export TEST=old
```

2. Viết một script

```
#!/bin/bash

echo old variable: $TEST

export $TEST=new

echo exported variable: $TEST
```

3. Giá trị của \$TEST là gì khi script được chạy?

4. Trong script sau gọi test_shell sẽ in PID của shell

test_shell

```
#!/bin/bash

if [ -n $(echo $0 |grep test) ]; then

echo The PID of the interpreter is: $$

else

echo The PID of the interpreter is: $$

fi
```

5) Thiết lập quyền 755 và kiểm tra các câu lệnh sau

```
test_shell

./test_shell

bash test_shell

. test_shell

source test_shell

exec ./test_shell
```

BẢO MẬT

Bảo mật địa phương

The BIOS

Nếu một người nào đấy tìm cách truy nhập các đĩa đã được bảo mật hoặc một đĩa linux bằng cách khởi động từ đĩa mềm hoặc CD ROM thì sẽ rất dễ dàng có thể đọc và truy cập tới bất kỳ file nào của hệ thống. Để tránh được điều này BIOS sẽ được thiết lập để thiết lập chỉ cho phép khởi động từ đĩa cứng. Khi điều này được thực hiện thành công nó sẽ thiết lập một mật khẩu trong BIOS.

LILO

LILO có thể đưa ra các tham số lựa chọn khi khởi động. Thông thường một số hệ điều hành Linux sẽ không hỏi mật khẩu khi khởi động hệ thống trong chế độ *single user* hoặc mức thực thi runlevel 1.

Có hai tham số lựa chọn sẽ được thêm vào trong */etc/lilo.conf*:

Tham số ***restricted*** sẽ nhắc người dùng nhập mật khẩu

Tham số ***password=""***, thiết lập xâu mật khẩu

Cần có nghĩa là LILO không thể đưa ra bất kỳ tham biến nào khi "password" không xác định trong ***lilo.conf***.

```
Boot=/dev/had
install=/boot/boot.b
Prompt
timeout=50
Password="password"
restricted
```

Quyền truy cập file

Để bảo vệ khỏi những tấn công phá huỷ file. Đề xuất thực hiện các bước sau.

Quản trị Hệ thống Linux - Cơ bản

1) Tạo các công cụ hệ thống không thể thay đổi được, hoặc các file nhật ký chỉ thêm vào cuối (append-only):

```
chattr +i /bin/login
chattr +i /bin/ps
chattr +a /var/log/messages
```

2) Tạo thư mục /tmp và /home nosuid hoặc noexec:

Lines to be changed in /etc/fstab

/tmp	/tmp	ext2	nosuid	1 2
/home	/home	ext2	noexec	1 2

3) Tìm tất cả file trong hệ thống không thuộc về một người dùng hoặc nhóm người dùng nào đó:

```
find / -nouser -o -nogroup
find / -perm +4000
```

File nhật ký (Log file)

Các file log chính là

/var/log/messages : chứa các thông tin đăng nhập bởi chương trình nền **syslogd**

/var/log/secure. : chứa thông tin những lần đăng nhập không thành công, thông tin về thêm người sử dụng, vv

Công cụ **last** sẽ liệt kê tất cả những lần đăng nhập và khởi động hệ thống thành công. Các thông tin được đọc từ file **/var/log/wtmp**.

Công cụ **who** và **w** liệt kê tất cả người dùng hiện tại đang đăng nhập vào hệ thống bằng cách sử dụng file **/var/run/utmp**.

Giới hạn người dùng

Quản trị Hệ thống Linux - Cơ bản

Khi file **/etc/nologin** tồn tại (có thể rỗng) thì nó sẽ bảo vệ tất cả người dùng từ khi đăng nhập vào hệ thống (ngoại trừ người dùng root). Nếu file **nologin** chứa một thông báo thì nó sẽ được hiển thị sau khi việc xác thực người dùng thành công.

Thư mục **/etc/security/** sẽ là một tập các file mà cho phép người quản trị giới hạn thời gian CPU người dùng, độ lớn tối đa file, số lượng kết nối tối đa, vv

/etc/security/access.conf : không cho phép đăng nhập đối với các nhóm và người dùng từ một vị trí xác định.

/etc/security/limits.conf

Định dạng của file này là

<domain> <type> <item> <value>

domain tên người dùng, tên nhóm (với @group)

type cứng hoặc mềm (hard or soft)

item	core	- giới hạn kích thước lõi của file (KB)
	data	- kích thước dữ liệu tối đa (KB)
	fsize	- kích thước tối đa của file (KB)
(KB)	memlock	- không gian địa chỉ khoá bộ nhớ (locked-in-memory) tối đa
	nofile	- số lượng tối đa file được mở
	cpu	- thời gian CPU lớn nhất (MIN)
	proc	- số lượng tối đa các tiến trình
	as	- giới hạn không gian địa chỉ
	maxlogins	- số lượng tối đa các đăng nhập đồng thời của người dùng này
	priority	- độ ưu tiên để chạy tiến trình người dùng
	locks	- số lượng tối đa khoá file mà người dùng có thể

An ninh mạng

Bảo mật mạng có thể được chia ra thành hai mục chính như sau:

Bảo mật theo máy chủ (Host Based Security)

Quyền truy cập vào các nguồn tài nguyên có thể được cho phép dựa vào yêu cầu dịch vụ của Host. Điều này được thực hiện bởi `tcp_wrappers`. Thư viện ***libwrap*** cũng đóng vai trò như `tcp_wrappers` cung cấp danh sách truy cập kiểm soát host đối với các dịch vụ mạng khác nhau. Một số dịch vụ như ***xinetd***, ***sshd***, và ***portmap***, được biên dịch dựa vào thư viện `libwrap` do đó có kích hoạt ***tcp_wrapper*** hỗ trợ cho các dịch vụ này.

Khi một client kết nối tới một dịch vụ với hỗ trợ `tcp_wrapper`, file `/etc/hosts.allow` và `/etc/hosts.deny` được phân tích (parse) để kích thích yêu cầu dịch vụ host. Dựa vào kết quả mà dịch vụ có thể được cho phép hoặc không.

File `hosts_access` có 2 hoặc 3 (lựa chọn) dấu hai chấm ngăn cách các trường. Trường đầu tiên là tên của tiến trình, tiếp theo là tên host hoặc domain bị hạn chế hoàn toàn với một “dấu chấm đầu” ("leading dot"), địa chỉ IP hoặc subnet với dấu chấm sau. Các từ đại diện như ALL và EXCEPT cũng được chấp nhận.

Cú pháp của file `/etc/hosts.{allow | deny}` như sau:

```
service : hosts [EXCEPT] hosts
```

Ví dụ:

```
/etc/hosts.deny
ALL:          ALL    EXCEPT    .example.com

/etc/hosts.allow
ALL:          LOCAL 192.168.0.
in.ftpd:      ALL
sshd:         .example.com
```

`Tcp_wrappers` có thể chạy một lệnh cục bộ dựa vào host tương ứng với các file `host_access`.

Công việc được hoàn thành với lệnh **spawn**. Bằng cách sử dụng ký tự %, việc thay thế có thể được sử dụng đối với tên của host và tên dịch vụ.

Ví dụ:

```
/etc/hosts.deny
```

```
ALL:                ALL : spawn (/bin/echo `date` from %c for %d >> /var/log/tcpwrap.log)
```

Để biết thêm các thông tin về ký tự thay thế %, xem trang trợ giúp **host_access (5)** bằng lệnh man.

Bảo mật theo cổng (Port Based Security)

Với chức năng lọc gói tin trong nhân của Linux, có thể giới hạn truy cập tới nguồn tài nguyên bằng cách tạo ra tập luật với các tiện ích như ipchains và iptables, sẽ cho phép xác định một gói tin khi đi qua hoặc giao diện mạng của nó và cũng chỉ ra điều gì sẽ diễn ra đối với gói tin này.

Có ba chuỗi trong ipchains và iptables, đó là

input, forward và output cho **ipchains**

INPUT, FORWARD, và OUTPUT cho **iptables**.

Ví dụ, khi sử dụng **ipchains** tất cả gói tin đi vào một giao diện mạng sẽ đi qua chuỗi input. Tất cả các gói tin không có đích là host này sẽ đi qua chuỗi forward.

Tất cả các gói tin được sinh ra bởi host và các gói tin chuyển tiếp sẽ đi qua chuỗi output.

Luật **ipchains** và **iptables** có thể xác định các thông tin như nguồn source (s), đích (d), giao thức (p), và cổng.

Ví dụ: Tất cả các gói tin từ địa chỉ 192.168.0.254 sẽ bị cấm

```
ipchains -A input -s 192.168.0.254 -j DENY
```

Các luật **Ipchains** và **iptables** có thể được thực thi theo các thông số lựa chọn sau

- A Thêm vào cuối (Append)
- D Xoá (Delete)
- P Thay đổi chính sách mặc định đối với một chuỗi (chain)
- I Chèn (Insert)
- F In các luật ra một chuỗi (Flush the rules(s) in a chain)
- N Tạo một chuỗi được người dùng định nghĩa
- X Xoá chuỗi do người dùng định nghĩa
- L Liệt kê

Ví dụ: Chính sách mặc định đối với một iptable có thể bị thay đổi từ ACCEPT thành DENY như sau:

```
iptables -P INPUT REJECT
```

```
iptables -P FORWARD REJECT
```

```
iptables -P OUTPUT REJECT
```

Trong dự án phát triển nhân Linux 2.4 và dự án Netfilter cũng sử dụng tiện ích bảng iptables để quản lý các luật firewall. Điểm khác biệt lớn nhất giữa iptable và ipchain là iptables hỗ trợ cho việc đánh giá các gói tin dựa trên trạng thái của chúng dựa theo các gói tin khác đã được truyền qua nhân.

Dưới đây là một ví dụ minh họa tường lửa theo trạng thái gói tin được thực hiện. Nó là một đoạn script shell gồm một số dòng lệnh.

Ví dụ:

Quản trị Hệ thống Linux - Cơ bản

Một đoạn script cơ bản phù hợp với người dùng gia đình (home user) hoặc không có nhu cầu kết nối internet nhưng vẫn sử dụng gateway cho mạng LAN và cho phép các kết nối từ mạng LAN tới tất các dịch vụ. Chú ý: Dòng bôi đậm dưới đây chỉ cho phép kết nối tại cổng 80

```
#!/bin/sh

# Variables
IPTABLES="/sbin/iptables"
LAN_IFACE="eth0"
INET_IFACE="eth1"
INET_IP="1.2.3.4"
LOCALHOST_IP="127.0.0.1/32"
LAN_IP="192.168.0.1/32"
LAN_BCAST="192.168.0.0/24"

# Setup IP Masquerading

echo "1" > /proc/sys/net/ipv4/ip_forward
$IPTABLES -t nat -A POSTROUTING -o $INET_IFACE -j MASQUERADE

# Specify the default policy for the built in chains
$IPTABLES -P INPUT DROP
$IPTABLES -P FORWARD DROP
$IPTABLES -P OUTPUT DROP

# Specify INPUT Rules
$IPTABLES -A INPUT -i !$INET_IFACE -j ACCEPT
$IPTABLES -A INPUT -p TCP -i $INET_IFACE -m state --state NEW --dport http -j ACCEPT
$IPTABLES -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

# Specify FORWARD Rules
$IPTABLES -A FORWARD -i $LAN_IFACE -j ACCEPT
```

```
$IPTABLES -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT

# Specify OUTPUT RULES

$IPTABLES -A OUTPUT -p ALL -s $LOCALHOST_IP -j ACCEPT

$IPTABLES -A OUTPUT -p ALL -s $LAN_IP -j ACCEPT
```

Shell an toàn

- **Xác thực Host**

Với ssh thì cả host và người dùng được xác thực. Xác thực host được hoàn thành bằng cách sử dụng các khoá đảo (swapping key). Khoá công khai và khoá riêng của host thông thường được lưu trữ trong `/etc/ssh` nếu bạn sử dụng OpenSSH. Phụ thuộc vào giao thức được sử dụng mà file khoá host sẽ được gọi `ssh_host_key` đối với Giao thức 1 và `ssh_host_rsa_key` hoặc `ssh_host_dsa_key` đối với giao thức 2. Mỗi khoá này có một khoá công cộng tương ứng, ví dụ `ssh_host_key.pub`.

Khi một ssh client kết nối tới một server thì server sẽ cung cấp một khoá host công khai. Đoạn ví dụ dưới đây người dùng sẽ được thông báo một số thông số như sau:

```
The authenticity of host 'neptune (10.0.0.8)' can't be established.
RSA key fingerprint is
8f:29:c2:b8:b5:b2:e3:e7:ec:89:80:b3:db:42:07:f4.
Are you sure you want to continue connecting (yes/no)?
```

Nếu bạn đồng ý tiếp tục kết nối thì khoá công khai của server sẽ được thêm vào trong file `$HOME/.ssh/known_hosts`.

- **Xác thực người dùng (sử dụng password)**

Tiếp theo người dùng sẽ được hệ thống nhắc nhập mật khẩu tương ứng với account của mình để đăng nhập vào server từ xa.

- **Xác thực người dùng (sử dụng khoá)**

Xác thực người dùng cũng có thể đòi hỏi các khoá đảo (swapping key). Để thực hiện điều này người dùng sẽ cần phải sinh ra một cặp khoá riêng / công khai.

Ví dụ:

```
ssh-keygen -t dsa -b 1024
```

sẽ sinh một lhoá DSA 1024 bit. Mặc định các khoá này sẽ được ghi trong \$HOME/.ssh và trong ví dụ này được gọi là **id_dsa** và **id_dsa.pub**.

Giả sử rằng chúng ta có một **id_dsa.pub** ta có thể ‘thành lập’ khoá này với một tài khoản từ xa và tránh được việc phải nhập mật khẩu đối với các kết nối sau này. Để thực hiện được việc này, ta cần phải copy nội dung của file **id_dsa.pub** vào một file có tên là **authorized_keys2** được lưu trữ trong thư mục từ xa \$HOME/.ssh.

CHÚ Ý

Tất cả khoá công khai trong **/etc/ssh** and **~/.ssh** sẽ có quyền là 600

● File cấu hình sshd

Ví dụ file **/etc/ssh/sshd_config**:

```
#Port 22

#Protocol 2,1

#ListenAddress 0.0.0.0

#ListenAddress ::

# HostKey for protocol version 1
#HostKey /etc/ssh/ssh_host_key

# HostKeys for protocol version 2
#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_dsa_key
```

● File cấu hình ssh configuration

File ví dụ **/etc/ssh/ssh_config** or **\$HOME/.ssh/config**:

```
# Host *
# ForwardX11 no
# RhostsAuthentication no
# RhostsRSAAuthentication no
# RSAAuthentication yes
# PasswordAuthentication yes
# HostbasedAuthentication no
# CheckHostIP yes
# IdentityFile ~/.ssh/identity
# IdentityFile ~/.ssh/id_rsa
# IdentityFile ~/.ssh/id_dsa
# Port 22
# Protocol 2,1
# Cipher 3des
```

Cấu hình thời gian

Ngày hệ thống

Ngày hệ thống có thể thay đổi với câu lệnh **date**. Cú pháp là:

```
date MMDDhhmmCCYY[.ss]
```

Đồng hồ phần cứng (Hardware Clock)

Đồng hồ phần cứng có thể được thay đổi trực tiếp với tiện ích **hwclock**. Các tham số lựa chọn chính là:

- r hoặc --show hiển thị thời gian hiện tại
- w hoặc --systohc thiết lập đồng hồ phần cứng thành thời gian hệ thống hiện tại
- s hoặc --hctosys thiết lập thời gian hệ thống với thời gian của đồng hồ phần cứng hiện tại

Sử dụng NTP

Toạ độ thời gian toàn cầu Coordinated Universal Time (UTC) là một tiêu chuẩn được sử dụng để giữ thời gian chuẩn dựa vào sự quay tròn của trái đất xung

quanh trục của mình. Tuy nhiên do có sự sai số nhỏ bất qui tắc khi chuyển động quay tròn nên quãng nhảy của giây cần được thêm vào thang độ UTC bằng cách sử dụng các đồng hồ nguyên tử.

Do máy tính không được trang bị các đồng hồ nguyên tử, ý tưởng sử dụng một giao thức để đồng bộ các đồng hồ máy tính qua internet. NTP - **Network Time Protocol** là một giao thức như vậy.

Các máy tính được cập nhật một cách trực tiếp bởi đồng hồ nguyên tử được gọi là thời gian chính (primary time) và được sử dụng để cập nhật một số lượng lớn máy chủ thời gian phụ khác. Điều này tạo nên một cấu trúc cây giống với cấu trúc DNS. Máy chủ gốc (root server) ở mức (tầng) đầu tiên, máy chủ thứ yếu sẽ ở mức thứ hai và tương tự như vậy với các mức thấp hơn.

Cấu hình một client truy vấn một máy chủ NTP:

Một tiến trình nền gọi là **ntpd** được sử dụng để truy vấn tới một máy chủ thời gian từ xa. Tham số cần thiết là **server** trong tệp **/etc/ntp.conf** trỏ đến một máy chủ NTP công cộng hoặc liên kết. Các máy chủ này có thể tìm thấy trực tuyến trên mạng.

Giao thức NTP cũng có thể ước lượng các lỗi về tần số của đồng hồ phần cứng qua một chuỗi các truy vấn, ước lượng này được ghi vào một fuke được tham chiếu với thẻ **driftfile**.

Mininal /etc/ntp.conf file
server ntp2.somewhere.com
driftfile /var/lib/ntp/drift

Khi **ntpd** được bắt đầu nó sẽ tự trở thành một máy chủ NTP cung cấp các dịch vụ tại cổng 123 bằng cách sử dụng UDP.

One off queries:

Gói **ntp** cũng hỗ trợ công cụ **ntpdate** được sử dụng để thiết lập thời gian qua một dòng lệnh:

ntpdate ntp2.somewhere.com

Bảo mật nhân

Có một số lựa chọn trong nhân Linux. Bao gồm cơ chế đồng bộ cookie syn_cookie. Tràn ngăn xếp bộ nhớ (Stack overflow) được kiểm soát bởi một miếng vá bảo mật gọi là tường mở (openwall) hoặc OWL.

• **tcp_syncookies**

Để kích hoạt lựa chọn này bạn chỉ cần thực hiện như sau:

```
[root@nasaspc /proc]#echo "1" > /proc/sys/net/ipv4/tcp_syncookies
```

Dòng lệnh này sẽ chỉ thị cho nhân gửi một cookie tới client trong tín hiệu trả lời SYN+ACK của nó. Trong chế độ này, máy chủ sẽ đóng socket và đợi tín hiệu ACK của client với một cookie tương ứng.

Nếu file tcp_syncookies không tồn tại trong thư mục /proc thì bạn cần phải dịch lại nhân với lựa chọn hỗ trợ syncookies.

Chú ý: Mặc định, thậm chí nếu syncookies đã được hỗ trợ bởi nhân thì bạn cần phải kích hoạt hỗ trợ bằng cách thêm "1" vào /proc/sys/net/ipv4/tcp_syncookies. Điều này thường được thực hiện trong **/etc/rc.d/rc.local**. Tuy nhiên có một giải pháp khác hiệu quả hơn là thêm một đầu vào (entry) vào **/etc/sysctl.conf**

• **Miếng vá bảo mật owl (phần này không phải là đối tượng trình bày của tài liệu này)**

Miếng vá này quan tâm đến hầu hết các vấn đề liên quan đến ngăn xếp bộ nhớ và nó không nằm trong phạm vi của khoá học này.

Địa chỉ miếng vá owl và nhân Linux:

<http://www.openwall.com>

<http://www.kernel.org/pub/linux/kernel/v2.2/>

Quản trị Hệ thống Linux - Cơ bản

Miếng vá này chỉ hỗ trợ cho nhân 2.2-19 hoặc phiên bản tiếp theo.

Sau khi download linux-2.2.19.tar.gz và linux-2.2.19-owl.tar.gz vào thư mục **/usr/src/**, chắc chắn là bạn đã xoá **linux** symbolic link.

```
[root@nasaspc src]#pwd
/usr/src/
[root@nasaspc src]#rm -rf linux
```

Giải nén các gói.

```
[root@nasaspc src]#tar xvzf linux-2.2.19.tar.gz
[root@nasaspc src]#tar xvzf linux-2.2.19-owl.tar.gz
```

Để kiểm tra hệ thống, chuyển tới thư mục linux-2.2-19-owl. Có một thư mục được gọi là lựa chọn chứa file có tên là **stacktest.c**.

```
[root@nasaspc optional]#pwd
/usr/src/linux-2.2.19-owl/optional
[root@nasaspc optional]#gcc stacktest.c -o stacktest
```

Nếu bạn chạy **stacktest** thì sẽ thu được danh sách các lựa chọn. Chạy mô phỏng tràn bộ nhớ.

Một tấn công tràn bộ nhớ đệm thành công:

```
[root@nasaspc optional]#stacktest
Usage: ./stacktest OPTION
Non-executable user stack area tests

-t  call a GCC trampoline
-e  simulate a buffer overflow exploit
-b  simulate an exploit after a trampoline call

[root@nasaspc optional]#stacktest -e
Attempting to simulate a buffer overflow exploit...
Succeeded.
```

Quản trị Hệ thống Linux - Cơ bản

Để áp dụng miếng vá bạn cần phải di chuyển tới thư mục **linux**. Sau đây là các câu lệnh.

Sử dụng miếng vá openwall:

```
[root@nasaspc linux]#pwd
```

```
/usr/src/linux
```

```
[root@nasaspc linux]#patch -p1 < /usr/src/linux-2.2-19-owl/linux-2.2.19-ow1.diff
```

Bây giờ nếu bạn thực hiện **make menuconfig** bạn sẽ thấy một cửa sổ nhập mới gọi là **Security options**. Các lựa chọn mặc định đều hợp lý. Từ đây bạn có thể bắt đầu việc dịch hoặc cài đặt nhân như bình thường.

QUẢN TRỊ HỆ THỐNG LINUX

Tổng quan

Chúng ta sẽ xem xét các nhiệm vụ chính của quản trị hệ thống như quản lý các file nhật ký, lập lịch công việc bằng cách sử dụng **at** và **cron**....

Logfiles và các file cấu hình

Thư mục /var/log/

Đây là thư mục chứa hầu hết các file nhật ký (log file). Một số ứng dụng sinh ra các file nhật ký của mình (ví dụ như squid hoặc samba). Hầu hết các file nhật ký hệ thống đều được quản lý bởi tiến trình nền **syslogd** daemon. Các file hệ thống phổ biến là:

cron	giữ và theo dõi các thông điệp sinh ra khi chạy crons
mail	các thông điệp liên quan đến mail
messages	ghi nhật ký tất cả các thông báo những lần thực thành công authpriv, cron, mail và news
secure	ghi nhật ký tất cả những lần xác thực không thành công, việc thêm / xóa người dùng, ...

File nhật ký quan trọng nhất là **messages** ghi lại nhật ký hầu hết các hoạt động.

File /etc/syslog.conf

Khi **syslogd** được khởi động thì mặc định nó sẽ đọc file cấu hình **/etc/syslog.conf**. Đầu tiên cũng có thể khởi động **syslogd** với **-f** và đường dẫn đến một file cấu hình tương ứng. File này sẽ phải chứa một danh sách các mục, tiếp theo là quyền và cuối cùng là đường dẫn đến file nhật ký:

<code>item1.priority1 ; item2.priority2</code>	<code>/path-to-log-file</code>
--	--------------------------------

Các mục cho phép là :

auth và **authpriv** người dùng chung và quyền riêng

Quản trị Hệ thống Linux - Cơ bản

cron	các thông điệp tiến trình cron
kern	các thông điệp nhân
mail	
news	
user	tiến trình người dùng
uucp	

Các quyền cho phép: (từ cao đến thấp)

emerg
alert
crit
err
warning
notice
info
debug

none

Các quyền là *tối thiểu*! Tất cả các quyền cao hơn sẽ được hệ thống ghi nhận ký. Để gán một quyền **info** bạn chỉ cần sử dụng dấu '=' để gán như sau:

```
user.=info      /var/log/user_activity
```

Danh sách /etc/syslog.conf

```
# Log all kernel messages to the console.
# Logging much else clutters up the screen.
#kern.*                          /dev/console
# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;news.none;authpriv.none      /var/log/messages

# The authpriv file has restricted access.
authpriv.*      /var/log/secure
```

```
# Log all the mail messages in one place.
mail.*                                /var/log/maillog

# Log cron stuff
cron.*                                /var/log/cron

# Everybody gets emergency messages, plus log them on another
# machine.
*.emerg                               *
*.emerg                               @10.1.1.254

# Save boot messages also to boot.log
local7.*                              /var/log/boot.log

#
news.=crit                             /var/log/news/news.crit
news.=err                             /var/log/news/news.err
news.notice                           /var/log/news/news.notice
```

Các tiện ích nhật ký

Câu lệnh logger

Tiện ích đầu tiên của nhật ký là câu lệnh **logger** sẽ ghi các thông điệp vào file `/var/log/messages` :

Nếu bạn gõ câu lệnh như sau:



```
logger  program myscript ERR
```

Phía cuối file `/var/log/messages` sẽ là một thông điệp tương tự như sau:

```
Jul 17 19:31:00 localhost penguin: program myscript ERR
```

Thiết lập địa phương (local settings)

Tiện ích **logger** sẽ mặc định ghi các thông báo vào `/var/log/messages`. Một số mục địa phương (local items) được định nghĩa trước có thể giúp bạn tạo ra các file nhật ký của mình như **local0** tới **local7** là các item sử dụng cho người quản trị hệ thống. Các item được định nghĩa này phụ thuộc vào hệ thống (File nhật ký ghi thông tin thời gian khởi động hệ thống RedHat **local7** trong `/var/log/boot.log`). Bạn hãy thêm một dòng sau đây vào file `/etc/syslog.conf`:

```
local4.*      /dev/tty9
```

Khởi động lại **syslogd**



```
killall -HUP syslogd
```

Câu lệnh tiếp theo sẽ được ghi nhật ký vào `/dev/tty9`



```
logger -p local4.notice "This script is writing to /dev/tty9"
```

Một thiết bị đáng quan tâm khác là `/dev/speech` được cài đặt với các công cụ Festival.

logrotate

Các file nhật ký được cập nhật bằng cách sử dụng **logrotate**. Thông thường **logrotate** được chạy hàng ngày như là một công việc cron. File cấu hình `/etc/logrotate.conf` sẽ chứa các câu lệnh tạo hoặc nén file.

Danh sách của `logrotate.conf`

```
# rotate log files weekly
weekly
# keep 4 weeks worth of backlogs
rotate 4
# send errors to root
errors root
# create new (empty) log files after rotating old ones
create
```

```
# uncomment this if you want your log files compressed
compress
# RPM packages drop log rotation information into this directory
include /etc/logrotate.d
# no packages own lastlog or wtmp -- we'll rotate them here
/var/log/wtmp {
    monthly
    create 0664 root utmp
    rotate 1
}
```

Tự động hóa công việc (Automatic Tasks)

Sử dụng cron

Chương trình có trách nhiệm chạy các cron được gọi là **crond**. Mỗi phút **crond** sẽ đọc các file có chứa câu lệnh để thực hiện. Các file này được gọi là *crontabs*.

File crontabs người dùng được lưu giữ trong **/var/spool/cron/<username>**. Các file này sẽ không cho phép soạn thảo trực tiếp bởi người dùng không phải là người dùng root và cần thiết phải sử dụng công cụ soạn thảo **crontab** (xem dưới đây).

File crontab hệ thống là **/etc/crontab**. File này sẽ thực hiện định kỳ tất cả các script trong **/etc/cron.*** bao gồm bất kỳ đường dẫn biểu tượng (symbolic link) trỏ tới các script hoặc các tệp nhị phân trong hệ thống.

Để thực thi các đầu vào **cron**, sử dụng công cụ **crontab**. Các công việc được lập lịch được xem với tham số lựa chọn **-l** như mô tả dưới đây:

```
crontab -l

→ # DO NOT EDIT THIS FILE - edit the master and reinstall
   # (/tmp/crontab.1391 installed on Tue Jul 17 17:56:48 2001)
   # (Cron version -- $Id: crontab.c,v 2.13 1994/01/17 03:20:37 vixie Exp $)
   0 * * 07 2 /usr/bin/find /home/penguin -name core -exec rm {} \;
```

Liệu người dùng **root** có crontabs nào không?

Quản trị Hệ thống Linux - Cơ bản

Tương tự như tham số lựa chọn **-e** sẽ mở trình soạn thảo mặc định của bạn và cho phép nhập đầu vào cron.

Người dùng **root** có thể sử dụng **-u** để xem và thay đổi bất kỳ đầu vào cron nào của người dùng.

Để xóa file crontab của bạn, sử dụng **crontab -r**.

Đây là định dạng của:

Minutes(0-59)	Hours(0-23)	Day of Month(1-31)	Month(1-12)	Day of Week(0-6)	command
---------------	-------------	--------------------	-------------	------------------	---------

Quyền:

Mặc định, một người dùng bất kỳ nào có thể sử dụng **crontab**. Tuy nhiên, bạn có thể kiểm soát khả năng truy cập với **/etc/cron.deny** và **/etc/cron.allow**.

Lập lịch với “at”

Các công việc **at** được chạy bởi tiến trình nền **atd** và được đẩy ra trong **/var/spool/at/**

Câu lệnh **at** được sử dụng để lập lịch một công việc đang tắt (off task) với cú pháp như sau

```
at [time]
```

Trong đó thời gian có thể được biểu diễn như sau:

now

3am + 2days

midnight

10:15 Apr 12

teatime

Để có danh sách đầy đủ các định dạng thời gian, xem **/usr/share/doc/at-xxx/timespec**.

Quản trị Hệ thống Linux - Cơ bản

Bạn có thể liệt kê các câu lệnh đã được lập lịch với **atq** hoặc **at -l**. Các công việc **at** được ghi trong `/var/spool/at/`:

```
ls /var/spool/at/  
→ a0000100fd244d spool
```

Khi sử dụng **atq** bạn sẽ phải có một danh sách các công việc được đánh số. Bạn cũng có thể sử dụng số này để loại bỏ khỏi hàng đợi công việc:

```
atq  
→ 1 2001-07-17 18:21 a root
```

Từ việc liệt kê **atq** chúng ta thấy rằng số công việc là **1**, do đó có thể loại bỏ công việc khỏi hàng đợi như sau:

```
at -d 1
```

Quyền:

Mặc định **at** sẽ hạn chế người dùng root. Để ghi đè, bạn phải có một **/etc/at.deny** rỗng

hoặc có **/etc/at.allow** với các tên tương ứng.

Sao lưu và nén

Chiến lược sao lưu (Backup strategies)

Có ba chiến lược để sao lưu một hệ thống là:

Đầy đủ: copy tất cả các file

Quản trị Hệ thống Linux - Cơ bản

Dự phòng: Đầu tiên copy tất cả các file mới được thêm hoặc thay đổi kể từ lần backup cuối cùng và sau đó copy tất cả các file mới được thêm hoặc sửa đổi từ lần backup dự phòng gần nhất

Sai lệch: Copy tất cả các file mới được thêm hoặc sửa đổi từ lần backup đầy đủ gần đây nhất

Ví dụ: nếu bạn thực hiện một backup đầy đủ và ba lần backup Sai lệch trước khi hệ thống sập đổ, bạn sẽ cần bao nhiêu tape để khôi phục lại?

Tạo file nén cần lưu trữ với tar

Lựa chọn chính để tạo ra một file nén cần lưu trữ với **tar** là **-c**. Bạn cũng có thể xác định tên của archive như là đối số đầu tiên nếu sử dụng cờ **-f**.

```
tar -cf home.tar /home/
```

Nếu bạn không xác định file như là một đối số **tar -c** thì đơn giản hệ thống sẽ cho đầu ra file nén cần lưu trữ như một đầu ra chuẩn:

```
tar -c /home/ > home.tar
```

Giải nén archives với tar

Thay cờ **-c** bằng **-x** sẽ tạo ra các thư mục nếu cần thiết và copy các file nén cần lưu trữ vào thư mục hiện thời của bạn. Để chuyển tiếp kết quả giải nén vào một thư mục (ví dụ thư mục `/usr/share/doc`), bạn có thể làm như sau:

```
tar xf backeddocs.tar -C /usr/share/doc
```

Nén

Tất cả các archives có thể được nén bằng nhiều tiện ích khác nhau. Các cờ sau sẽ cho phép khi tạo, thử nghiệm (testing) hoặc giải nén một tệp cần lưu trữ:

Tham số lựa chọn tar	Kiểu nén
Z	compress
z	gzip
j	bzip2.

Tiện ích cpio

Tiện ích **cpio** được sử dụng để copy các file từ hoặc đến các file nén.

- Giải nén một file dữ liệu trên tape:

```
cpio -i < /dev/tape
```

- Tạo một file nén dữ liệu cho thư mục /etc:

```
find /etc | cpio -o > etc.cpio
```

Tài liệu

Trang trợ giúp Manpages và cơ sở dữ liệu whatis

Trang trợ giúp được tổ chức theo các phần	
NAME	tên của mục (item) tiếp theo bởi một dòng ghi chú ngắn
SYNOPSIS	cú pháp của câu lệnh
DESCRIPTION	giải thích dài
OPTIONS	Các tham số lựa chọn có thể
FILES	Các file liên quan đến item hiện tại(ví dụ các file cấu hình)
SEE ALSO	các trang hướng dẫn khác liên quan đến chủ đề hiện tại

Các phần trên không thể thiếu trong một trang trợ giúp.

Quản trị Hệ thống Linux - Cơ bản

Cơ sở dữ liệu **whatis** lưu trữ phần NAME của tất cả các trang trợ giúp trong hệ thống. Việc lưu trữ này được thực hiện bởi **cron** hàng ngày. Cơ sở dữ liệu **whatis** có hai đầu vào như sau:

<code>name(key) - one line description</code>

Cú pháp của **whatis** là:

```
whatis <string>
```

Kết quả đầu ra là phần NAME đầy đủ của các trang trợ giúp trong đó *string* tương ứng với *named(key)*

Bạn cũng có thể sử dụng câu lệnh **man** để truy vấn cơ sở dữ liệu **whatis**. Cú pháp của **man** là

```
man -k <string>
```

Không giống như **whatis**, câu lệnh **man** sẽ truy vấn cả “name” và “one line description” của cơ sở dữ liệu. Nếu *string* phù hợp với một từ trong bất kỳ một trường nào ở trên, truy vấn sẽ trả về một NAME đầy đủ.

Ví dụ: (String phù hợp sẽ được bôi đậm)

```
whatis lilo
```

```
lilo (8) - install boot loader
```

```
lilo.conf [lilo] (5) - configuration file for lilo
```

```
man -k lilo
```

```
grubby (8) - command line tool for configuring grub, lilo, and elilo
```

```
lilo (8) - install boot loader
```

```
lilo.conf [lilo] (5) - configuration file for lilo
```

Các trang trợ giúp được lưu giữ trong **/usr/share/man**

Các phần của trang trợ giúp	
Phần 1	thông tin trên các bảng executables
Phần 2	Các lời gọi hệ thống, ví dụ mkdir(2)
Phần 3	Các lời gọi thư viện, ví dụ stdio(3)
Phần 4	Các thiết bị (files trong /dev)
Phần 5	Các file cấu hình và định dạng
Phần 6	Các trò chơi
Phần 7	Các gói Macro
Phần 8	Các câu lệnh quản trị
Phần 9	Các đoạn mã nhân (Kernel routines)

Để truy cập vào một phần N xác định, bạn gõ:

man N command

Ví dụ:

```
man mkdir
```

```
man 2 mkdir
```

```
man crontab
```

```
man 5 crontab
```

Các trang thông tin

Các trang thông tin (infor page) nằm trong thư mục **/usr/share/info**. Các trang này là các file nén và có thể đọc với công cụ **info**.

Các công cụ GNU nguyên bản hay sử dụng các trang thông tin hơn các trang trợ giúp (man page). Tuy nhiên thông tin về các dự án GNU như **gcc** hoặc **glibc** vẫn có phạm vi rộng hơn trong các trang thông tin so với các trang trợ giúp.

Tài liệu trực tuyến

Các dự án GNU bao gồm các tài liệu như FAQ, README, CHANGELOG và thỉnh thoảng là hướng dẫn user/admin. Định dạng của các tài liệu này có thể là ASCII text, HTML, LaTeX hoặc postscript.

Các tài liệu này được lưu giữ trong thư mục **/usr/share/doc/**.

HOWTOs và Dự án tài liệu Linux

Dự án tài liệu Linux (LDP) cung cấp nhiều tài liệu chi tiết theo các chủ đề khác nhau. Các tài liệu này hướng dẫn cách sử dụng và thực thi trên Linux. Địa chỉ của trang web là www.tldp.org.

Các tài liệu The LDP đều miễn phí và có thể được phân phối theo giấy phép CPL.

Thực hành

Ghi nhật ký

1. Thay đổi file **/etc/syslog.conf** để in ra một số nhật ký tới **/dev/tty9** (đảm bảo rằng bạn khởi động lại **syslogd** và kết quả đầu ra được chuyển gián tiếp một cách hợp lệ)
2. Thêm một mục (item) **local5** với quyền tới **/etc/syslog.conf** và đặt đầu ra trực tiếp tới **/dev/tty10**. Khởi động lại **syslogd** và sử dụng **logger** để ghi thông tin qua **local5**.
3. Đọc script **/etc/rc.d/init.d/syslog** và thay đổi **/etc/sysconfig/syslog** để cho phép các host từ xa gửi các nhật ký đầu ra.

Lập lịch

4. Tạo một đầu vào cron sẽ khởi động **xclock** theo định kỳ 2 phút một lần. Chú ý rằng **cron** không biết các biến hệ thống như **PATH** và **DISPLAY**.
5. Sử dụng **at** để khởi động **xclock** trong năm phút tiếp theo.

Archiving

6. Sử dụng **find** để liệt kê tất cả các trường đã được sửa đổi trong vòng 24 giờ gần nhất..

(gợi ý: Chuyển tiếp đầu ra của `find -mtime -1` tới 1 file)

7. Sử dụng **cpio** để tạo một tệp nén cần lưu trữ có tên là `Incremental.cpio`.

(trả lời: Sử dụng file cửa đọc tạo ra ở trên và thực hiện `cat FILE | cpio -ov > Incremental.cpio`)

8. Sử dụng **xargs** và **tar** để tạo ra một file nén dữ liệu của tất cả các file đã được cập nhật mới hoặc thay đổi trong vòng 5 phút gần đây nhất.

9. Tương tự như trên sử dụng tham số lựa chọn **-exec** với câu lệnh **find**. Chú ý, các file được liệt kê bởi **find** có thể được tham chiếu bởi biểu tượng `{}`.

10. Giải nén file bạn vừa tạo ra.

IN ẤN

Có hai mục đích trong chương này đó là giới thiệu các công cụ in ấn GNU sẵn có trên Linux và hiểu rõ các file cấu hình đối với máy chủ in ấn.

Bộ lọc (Filters) và gs

Đối với những định dạng phi văn bản, hệ thống Linux và Unix thường sử dụng các bộ lọc. Những bộ lọc nào sẽ chuyển những định dạng JPEG hoặc troff vào định dạng postscript. Và định dạng này có thể được gửi trực tiếp đến máy in postscript, tuy nhiên không phải tất cả máy in thông thường có khả năng xử lý postscript, một thiết bị trung gian "máy in postscript ảo" có tên là **gs** (ghostscript) sẽ chuyển đổi postscript vào PCL.

Bản thương mại của ghostscript là Aladdin Ghostscript và bản GNU là version cũ hơn.

Tiện ích **gs** có một cơ sở dữ liệu của các thiết bị điều khiển (driver) cho máy in (danh sách các thiết bị điều khiển thường xuyên được cập nhật, ví dụ rất nhiều các máy in USB có thể dùng được), do đó tiện ích này sẽ xử lý và chuyển đổi postscript trực tiếp vào PCL cho những loại máy in đã biết. Tiện ích **gs** đóng vai trò trung tâm trong quá trình xử lý in ấn của Linux.

Máy in và hàng đợi in

Như đã đề cập ở trên các dạng văn bản ascii đơn giản không cần xử lý theo cách thức giống như các file hình ảnh hoặc postscript. Nếu chúng ta chỉ có duy nhất một máy in và ví dụ muốn in ra những bức thư, thì chúng ta không cần thiết sử dụng bộ lọc. Chúng ta sẽ định nghĩa một hàng đợi thay thế bộ lọc và giúp quá trình in diễn ra nhanh hơn. Chúng ta cũng có thể định nghĩa một hàng đợi trên cùng một máy in dành cho việc xử lý các file postscript.

Tất cả các hàng đợi và máy in được định nghĩa trong /etc/printcap. Dưới đây là cấu hình đầy đủ của một máy in từ xa 192.168.1.20 sử dụng hàng đợi từ xa có tên là 'lp':

Quản trị Hệ thống Linux - Cơ bản

```
lp:\
    :sd=/var/spool/lpd/lp:\
    :mx#0:\
    :sh:\
    :rm=192.168.1.20:\
    :rp=lp:
```

Các lựa chọn cần thiết ở đây là **rm** dành cho máy chủ từ xa, **sd** là thư mục đường ống máy in (spool), và **rp** là tên của hàng đợi từ xa. Chú ý rằng không có bộ lọc nào được xác định ở đây (chúng ta có thể sử dụng lệnh **if** cho bộ lọc đầu vào). Tất cả các quá trình lọc được thực hiện trên máy chủ từ xa.

Các công cụ in ấn

lpr:

Tiện ích **lpr** được dùng để gửi các công việc liên quan đến in ấn tới máy in. Đây là một phiên bản mới của **lp** (line print). Đối với người dùng sẽ thuận tiện hơn nếu như một máy in có thể gắn kết với nhiều hơn một hàng đợi. Dưới đây là hai ví dụ để in một file có tên là LETTER.

Gửi công việc đến máy in mặc định:

```
lpr LETTER
```

Gửi công việc đến hàng đợi 'ljet':

```
lpr -Pljet LETTER
```

Bảng 1: Các lựa chọn chính cho lpr

-#num	In num bản copies
-Ppq	Chỉ định hàng in pq
-s	Tạo một liên kết tượng trưng trong thư mục đường ống máy in thay cho quá trình copy file vào đó

lpq:

Quản trị Hệ thống Linux - Cơ bản

Người dùng có thể quan sát trạng thái của hàng in bằng tiện ích **lpq**. Dưới đây là một vài ví dụ.

Hiển thị các công việc trong hàng đợi mặc định:

```
lpq
```

Hiển thị các công việc cho tất cả hàng đợi trong hệ thống

```
lpq -a
```

Hiển thị các công việc trong hàng đợi từ xa

```
lpq -Promote
```

lprm:

Tuỳ thuộc vào lựa chọn trong **/etc/lpd.perms** người dùng có thể được phép xoá những công việc đang chờ đợi bằng lệnh **lprm**.

Xoá công việc cuối cùng được gửi đi

```
lprm
```

Xoá các công việc được gửi đi bằng người dùng dhill:

```
lprm dhill
```

Xoá tất cả công việc được gửi đi:

```
lprm -a (or simply lprm -)
```

Chúng ta cũng có thể xoá một công việc cụ thể trong đường ống máy in bằng cách chỉ ra giá trị của công việc, giá trị này được tạo ra bởi **lpq**.

lpc:

Tiện ích điều khiển máy in theo dòng (Line Printer Control) được dùng để điều khiển các hàng in và các máy in. Các hàng in có thể bị vô hiệu hoá hoặc làm việc

Quản trị Hệ thống Linux - Cơ bản

trở lại. Chú ý rằng lệnh `lprm` chỉ có thể xoá các công việc từ hàng đợi nhưng không có thể dừng lại một hàng đợi.

Chúng ta có thể thực hiện tương tác với `lpc` (`lpc` có dấu nhắc riêng) hoặc sử dụng dòng lệnh.

Dưới đây là kết quả của lệnh **`lpc – help`**:

```
CMD: /usr/sbin/lpc help
```

```
► Commands may be abbreviated. Commands are:
```

```
abort    enable  disable help    restart status topq    ?
clean    exit   down   quit    start  stop   up
```

Các lựa chọn **`enable/disable/topq/up`** liên quan đến hàng đợi.

Các lựa chọn **`start/stop/down`** liên quan đến máy in.

Các file cấu hình

`/etc/printcap`

Như đã đề cập trong phần trước của chương này, file trên sẽ định nghĩa tất cả các máy in và hàng đợi mà hệ thống có thể dùng (từ xa hoặc cục bộ).

Máy in mặc định có thể được xác định với các biến `LPDEST` hoặc `PRINTER`:
`PRINTER=lp`

Nếu không có biến môi trường nào được thiết lập, máy in mặc định là máy in đầu tiên được định nghĩa trong **`/etc/printcap`**.

Các định nghĩa chính là:

- `lp`** tên thiết bị, thông thường `/dev/lp0` cho cổng song song
- `mx`** dung lượng file lớn nhất (giá trị 0 có nghĩa là không giới hạn)
- `sd`** thư mục đường ống máy in
- `if`** bộ lọc đầu vào
- `rm`** địa chỉ máy chủ từ xa hoặc IP

rp tên hàng đợi từ xa

Nếu như file `/etc/printcap` có thay đổi thì chúng ta cần khởi động lại daemon **lpd**.

/etc/lpd.conf

Đây là một file có nội dung rất dài và ngầm định là tất cả các lựa chọn đều được ghi chú. File này được dùng khi người quản trị mạng muốn có thêm quyền điều khiển đối với quá trình in ấn (ví dụ: xác thực quyền truy nhập từ xa, các quyền của người dùng...)

/etc/lpd.perms

File này điều khiển các quyền liên quan đến các tiện ích **lpc**, **lpq**, và **lprm**. Cụ thể chúng ta có thể cung cấp cho người dùng quyền để loại bỏ những công việc hiện thời của họ từ hàng đợi với dòng lệnh sau:

```
ACCEPT            SERVICE=M      SAMEHOST SAMEUSER
```

LPRng sẽ sử dụng một hệ thống các phím để rút gọn các mục trong **lpd.perms**. Tuy nhiên quá trình này không dễ dàng có thể hiểu được đối với nhiều trường hợp. Ví dụ dịch vụ 'M' tương ứng với **lprm** trong dòng lệnh phía trên.

Ví dụ về file /etc/lpd.perms:

```
## Permissions are checked by the use of 'keys' and matches. For each of
## the following LPR activities, the following keys have a value.
##
## Key            Match Connect Job    Job    LPQ   LPRM   LPC
##                            Spool Print
## SERVICE       S       'X'    'R'    'P'    'Q'    'M'    'C'
## USER          S       -       JUSR   JUSR   JUSR   JUSR   JUSR
## HOST          S       RH       JH       JH       JH       JH
## GROUP         S       -       JUSR   JUSR   JUSR   JUSR   JUSR
## IP            IP       RIP       JIP    JIP       RIP    JIP    JIP
## PORT          N       PORT    PORT   -       PORT   PORT   PORT
## REMOTEUSER    S       -       JUSR   JUSR   JUSR   CUSR   CUSR
## REMOTEHOST    S       RH       RH       JH       RH       RH       RH
## REMOTEGROUP   S       -       JUSR   JUSR   JUSR   CUSR   CUSR
## REMOTEIP      IP       RIP       RIP    JIP       RIP    RIP    RIP
## CONTROLLINE   S       -       CL       CL       CL       CL       CL
```

Quản trị Hệ thống Linux - Cơ bản

```
## PRINTER      S      -      PR      PR      PR      PR      PR
## FORWARD      V      -      SA      -      -      SA      SA
## SAMEHOST      V      -      SA      -      SA      SA      SA
## SAMEUSER      V      -      -      -      SU      SU      SU
## SERVER        V      -      SV      -      SV      SV      SV
## LPC           S      -      -      -      -      -      LPC
## AUTH          V      -      AU      AU      AU      AU      AU
## AUTHTYPE      S      -      AU      AU      AU      AU      AU
## AUTHUSER      S      -      AU      AU      AU      AU      AU
## AUTHFROM      S      -      AU      AU      AU      AU      AU
## AUTHSAMEUSER  S      -      AU      AU      AU      AU      AU
##
## KEY:
##   JH = HOST          host in control file
##   RH = REMOTEHOST    connecting host name
##   JUSR = USER       user in control file
##       AUTH will match (true) if authenticated transfer
##       AUTHTYPE will match authentication type
##       AUTHUSER will match client authentication type
##       AUTHFROM will match server authentication type and is NULL if not
from server
##       AUTHSAMEUSER will match client authentication to save authentication
in job
##
## Example Permissions
##
## # All operations allowed except those specifically forbidden
## DEFAULT ACCEPT
##
## #Reject connections from hosts not on subnet 130.191.0.0
## # or Engineering pc's
##   REJECT SERVICE=X NOT REMOTEIP=130.191.0.0/255.255.0.0
##   REJECT SERVICE=X NOT REMOTEHOST=engpc*
##
## #Do not allow anybody but root or papowell on
## #astart1.astart.com or the server to use control
## #facilities.
##   ACCEPT SERVICE=C SERVER REMOTEUSER=root
##   ACCEPT SERVICE=C REMOTEHOST=astart1.astart.com REMOTEUSER=papowell
##
## #Allow root on talker.astart.com to control printer hpjet
##   ACCEPT SERVICE=C HOST=talker.astart.com PRINTER=hpjet REMOTEUSER=root
## #Reject all others
##   REJECT SERVICE=C
##
## #Do not allow forwarded jobs or requests
##   REJECT SERVICE=R,C,M FORWARD
##
#
# allow root on server to control jobs
ACCEPT SERVICE=C SERVER REMOTEUSER=root
# allow anybody to get server, status, and printcap
ACCEPT SERVICE=C LPC=lpd,status,printcap
```

Quản trị Hệ thống Linux - Cơ bản

```
# reject all others
REJECT SERVICE=C
#
# allow same user on originating host to remove a job
ACCEPT SERVICE=M SAMEHOST SAMEUSER
# allow root on server to remove a job
ACCEPT SERVICE=M SERVER REMOTEUSER=root
REJECT SERVICE=M
# all other operations allowed
DEFAULT ACCEPT
```

/etc/host.{lpd,equiv}

Những file này được dùng bởi hệ thống các quá trình in ấn LPR và có rủi ro về bảo mật. Khi thực hiện máy dịch vụ in, chúng ta cần xác định những máy chủ nào có thể truy cập vào máy in ở trong **/etc/hosts.lpd**. Chúng ta cũng cần bổ sung những máy chủ này vào **/etc/hosts.equiv**.

Những file này ngày nay được thay thế trong LPRng bằng file **/etc/lpd.perms**

Thực hành

1. Sử dụng **printtool** và tại một hàng đợi cục bộ có tên là **lp**.
2. Chỉnh sửa thiết bị **/dev/tty10** như là thiết bị máy in (nhớ thực hiện **chmod 666 /dev/tty10** để cho phép in ấn trên thiết bị này). Bây giờ bạn có một máy in ảo trên hệ thống của bạn!
3. Gửi các công việc đến hàng in sử dụng **lpr** và **pr**.
4. Với công cụ in ấn trên hệ thống của bạn, hãy định nghĩa các hàng đợi từ xa khác nhau
 - một hàng đợi UNIX
 - một hàng đợi SMB

Nếu bạn đang sử dụng máy chủ, chắc chắn các câu lệnh phù hợp trên sẽ được định nghĩa trong **/etc/lpd.perms**

Trong mỗi trường hợp

Quản trị Hệ thống Linux - Cơ bản

- kiểm tra file **/etc/printcap**. Bộ lọc nào được sử dụng? Máy chủ từ xa được định nghĩa như thế nào?

- kiểm tra thư mục **/var/spool/lpd/**

5. Dùng các hàng in khác nhau và các máy in với **lpc**.

6. Kiểm tra nội dung của mỗi hàng in với **lpc**.

7. Loại bỏ khỏi hàng đợi những công việc cụ thể với **lprm**