

**Giáo trình**

**Quản trị hệ điều  
hành Linux**



# QUẢN TRỊ HỆ ĐIỀU HÀNH LINUX

## MỤC LỤC

- 1. Giới thiệu hệ điều hành Linux**
  - 1.1 Lịch sử Linux**
  - 1.2 Cài đặt Linux**
- 2. Giao tiếp trên môi trường Linux**
  - 2.1 Giới thiệu trình soạn thảo vi**
  - 2.2 Giới thiệu tiện ích mc**
  - 2.3 Các câu lệnh cơ bản trên Linux**
    - 2.3.1 Hiểu biết về các câu lệnh trong Linux
    - 2.3.2 Các câu lệnh về thư mục và file
    - 2.3.3 Các câu lệnh nén dữ liệu
    - 2.3.4 Các câu lệnh quản lý tiến trình
- 3. Giới thiệu hệ thống tập tin, thư mục.**
  - 3.1 Giới thiệu**
    - 3.1.1 Thư mục chủ
    - 3.1.2 Các thư mục hệ thống
  - 3.2 Các quyền truy cập file, thư mục**
    - 3.2.1 Thay đổi quyền sở hữu file, thư mục sử dụng lệnh chown
    - 3.2.2 Thay đổi nhóm sử dụng file/thư mục với lệnh chgrp
    - 3.2.3 Sử dụng số theo hệ cơ số 8 tương ứng với thuộc tính truy cập
    - 3.2.4 Sử dụng ngôn ngữ tự nhiên tương ứng với quyền truy cập
    - 3.2.5 Thay đổi quyền truy cập file thư mục sử dụng lệnh chmod
    - 3.2.6 Các chú ý đặc biệt trên các quyền thư mục
  - 3.3 Thiết lập một chính sách cho server nhiều người sử dụng**
    - 3.3.1 Thiết lập cấu hình các quyền truy cập file của người sử dụng
    - 3.3.2 Thiết lập mặc định các quyền truy cập file cho người sử dụng
    - 3.3.3 Thiết lập các quyền có thể thực thi cho các file
  - 3.4 Làm việc với file, thư mục**
    - 3.4.1 Xem các file và các thư mục
    - 3.4.2 Chuyển đến thư mục
    - 3.4.3 Xác định kiểu file
    - 3.4.4 Xem thống kê các quyền của file hay thư mục
    - 3.4.5 Sao chép file và thư mục
    - 3.4.6 Dịch chuyển các file và thư mục
    - 3.4.7 Xóa các file và thư mục
    - 3.4.8 Tìm kiếm file
- 4. Quản lý người dùng và tài nguyên**
  - 4.1 Khái niệm**
  - 4.2 Tạo superuser**
  - 4.3 Quản lý người dùng với các công cụ dòng lệnh**
    - 4.3.1 Tạo một tài khoản người sử dụng mới
    - 4.3.2 Tạo một nhóm mới
    - 4.3.3 Sửa đổi một tài khoản người sử dụng đang tồn tại
    - 4.3.4 Thay đổi đường dẫn thư mục chủ

- 4.3.5 Thay đổi UID
- 4.3.6 Thay đổi nhóm mặc định
- 4.3.7 Thay đổi thời hạn kết thúc của một tài khoản
- 4.3.8 Sửa đổi một nhóm đang tồn tại
- 4.3.9 Xóa hoặc hủy bỏ một tài khoản người sử dụng
- 4.4 Cài đặt máy in**
  - 4.4.1 Cấu hình máy in
  - 4.4.2 Cài đặt máy in cục bộ
  - 4.4.3 Cài đặt máy in trên hệ thống Unix ở xa
  - 4.4.4 Cài đặt máy in Samba (SMB)
  - 4.4.5 Chọn trình điều khiển Print Driver và kết thúc
  - 4.4.6 Thay đổi thông số cấu hình các máy in có sẵn
  - 4.4.7 Backup các thông số cấu hình máy in
  - 4.4.8 Quản lý công việc in ấn
- 5. Trình diễn thiết lập mạng và cài đặt Diul-up trên Linux**
  - 5.1 Thiết lập mạng**
    - 5.1.1 HĐH Linux và card mạng
    - 5.1.2 Cấu hình card mạng
    - 5.1.3 Các tiện ích mạng: Telnet và ftp
  - 5.2 Cài đặt Diul-up**
    - 5.2.1 Cài đặt
    - 5.2.2 Quay số từ xa
- 6. Lập trình shell**
  - 6.1 Tạo và chạy chương trình shell**
  - 6.2 Sử dụng các biến**
    - 6.2.1 Gán một giá trị cho một biến
    - 6.2.2 Tham số và các biến Shell có sẵn
  - 6.3 Sử dụng dấu trích dẫn**
  - 6.4 Làm việc với câu lệnh test**
  - 6.5 Sử dụng các câu lệnh rẽ nhánh**
    - 6.5.1 Lệnh if
    - 6.5.2 Lệnh case
  - 6.6 Sử dụng các câu lệnh vòng lặp**
    - 6.6.1 Lệnh for
    - 6.6.2 Lệnh while
    - 6.6.3 Lệnh until
    - 6.6.4 Lệnh shift
    - 6.6.5 Lệnh select
    - 6.6.6 Lệnh repeat
  - 6.7 Sử dụng các hàm**
  - 6.8 Tổng kết**
- 7. Cài đặt và Quản trị WebServer**
  - 7.1 Hướng dẫn cài đặt trên môi trường Linux**
  - 7.2 Quản trị WebServer**
    - 7.2.1 Phần mềm Apache
    - 7.2.2 Biên dịch và cài đặt
    - 7.2.3 Khởi động và tắt WebServer
    - 7.2.4 Cấu hình Apache
    - 7.2.5 Xác thực người dùng

## **8. Quản lý tiến trình**

### **8.1 Tiến trình**

8.1.1 Tiến trình tiền cảnh

8.1.2 Tiến trình hậu cảnh

### **8.2 Điều khiển và giám sát tiến trình**

8.2.1 Sử dụng lệnh ps để lấy thông tin trạng thái của tiến trình

8.2.2 Phát tín hiệu cho một chương trình đang chạy

8.2.3 Giao tiếp giữa các tiến trình

### **8.3 Lập kế hoạch các tiến trình**

8.3.1 Sử dụng lệnh at

8.3.2 Sử dụng lệnh crontab

## **9. Bảo mật hệ thống**

### **9.1 Những nguy cơ an ninh trên Linux**

### **9.2 Xem xét chính sách an ninh của bạn**

### **9.3 Tăng cường an ninh cho KERNEL**

### **9.4 An toàn các giao dịch trên mạng**

### **9.5 Linux firewall**

### **9.6 Dùng công cụ dò tìm để khảo sát hệ thống**

### **9.7 Phát hiện sự xâm nhập qua mạng**

### **9.8 Kiểm tra khả năng bị xâm nhập**

### **9.9 Đối phó khi hệ thống bị tấn công**

## 1. Giới thiệu hệ điều hành Linux

### 1.1. Lịch sử

Linux là hệ điều hành mô phỏng Unix, được xây dựng trên phần nhân (kernel) và các gói phần mềm mã nguồn mở. Linux được công bố dưới bản quyền của GPL (General Public Licence).

Unix ra đời giữa những năm 1960, ban đầu được phát triển bởi AT&T, sau đó được đăng ký thương mại và phát triển theo nhiều dòng dưới các tên khác nhau. Năm 1990 xu hướng phát triển phần mềm mã nguồn mở xuất hiện và được thúc đẩy bởi tổ chức Gã U. Một số licence về mã nguồn mở ra đời ví dụ BSD, GPL. ả ăm 1991, Linus Torvald viết thêm phiên bản nhân v0.01 (kernel) đầu tiên của Linux đưa lên các BBS, nhóm người dùng để mọi người cùng sử dụng và phát triển. ả ăm 1996, nhân v1.0 chính thức công bố và ngày càng nhận được sự quan tâm của người dùng. ả ăm 1999, phiên bản nhân v2.2 mang nhiều đặc tính ưu việt và giúp cho linux bắt đầu trở thành đối thủ cạnh tranh đáng kể của MSwindows trên môi trường server. ả ăm 2000 phiên bản nhân v2.4 hỗ trợ nhiều thiết bị mới (đa xử lý tới 32 chip, USB, RAM trên 2GB...) bắt đầu đặt chân vào thị trường máy chủ cao cấp. Quá trình phát triển của linux như sau:

- ả ăm 1991: 100 người dùng.
- ả ăm 1997: 7.000.000 người dùng.
- ả ăm 2000: hàng trăm triệu người dùng, hơn 15.000 người tham gia phát triển Linux. Hàng năm thị trường cho Linux tăng trưởng trên 100%.

Các phiên bản Linux là sản phẩm đóng gói Kernel và các gói phần mềm miễn phí khác. Các phiên bản này được công bố dưới licence GPL. Một số phiên bản nổi bật là: Redhat, Caldera, Suse, Debian, TurboLinux, Mandrake.

Giống như Unix, Linux gồm 3 thành phần chính: kernel, shell và cấu trúc file.

*Kernel* là chương trình nhân, chạy các chương trình và quản lý các thiết bị phần cứng như đĩa và máy in.

*Shell* (môi trường) cung cấp giao diện cho người sử dụng, còn được mô tả như một bộ biên dịch. Shell nhận các câu lệnh từ người sử dụng và gửi các câu lệnh đó cho nhân thực hiện. ả ề shell được phát triển. Linux cung cấp một số shell như: desktops, windows manager, và môi trường dòng lệnh. Hiện nay chủ yếu tồn tại 3 shell: Bourne, Korn và C shell. Bourne được phát triển tại phòng thí nghiệm Bell, C shell được phát triển cho phiên bản BSD của Uả IX, Korn shell là phiên bản cải tiến của Bourne shell. ả ững phiên bản hiện nay của Unix, bao gồm cả Linux, tích hợp cả 3 shell trên.

*Cấu trúc file* quy định cách lưu trữ các file trên đĩa. File được nhóm trong các thư mục. Mỗi thư mục có thể chứa file và các thư mục con khác. Một số thư mục là các thư mục chuẩn do hệ thống sử dụng. ả ười dùng có thể tạo các file/thư mục của riêng mình cũng như dịch chuyển các file giữa các thư mục đó. Hơn nữa, với Linux người dùng có thể thiết lập quyền truy nhập file/thư mục, cho phép hay hạn chế một người dùng hoặc một nhóm truy nhập file. Các thư mục trong Linux được tổ chức theo cấu trúc cây, bắt đầu bằng một thư mục gốc (root). Các thư mục khác được phân nhánh từ thư mục này.

Kernel, shell và cấu trúc file cấu thành nên cấu trúc hệ điều hành. Với những thành phần trên người dùng có thể chạy chương trình, quản lý file, và tương tác với hệ thống.

## 1.2. Cài đặt máy chủ Linux

Lưu ý: trước khi cài đặt, cần tìm hiểu các thông tin về phần cứng của hệ thống, bao gồm

- Thông tin về ổ đĩa cứng
- Thông tin về card mạng
- Thông tin về card đồ hoạ
- Thông tin về màn hình
- Thông tin về giao thức và cấu hình mạng nếu kết nối mạng
- Thông tin về các thiết bị ngoài.

Có thể chọn nhiều phương án cài đặt như cài đặt từ đĩa mềm, từ đĩa cứng, từ đĩa CD Rom hoặc qua mạng. Tài liệu này chọn hướng dẫn quá trình cài đặt phiên bản 7.0 từ đĩa CDRom. Yêu cầu máy cài đặt có khả năng khởi động (boot) từ ổ đĩa CD-Rom (được hỗ trợ hầu hết trong các máy tính hiện nay).

Sau đây là các bước cài đặt cụ thể. Khi kết thúc bước trước chương trình cài đặt tự động chuyển sang bước sau. Một số bước cài đặt cho phép quay lại bước trước bằng cách chọn Back.

1. Đưa đĩa CD Rom Redhat vào ổ đĩa. Khởi động lại máy (lưu ý phải đảm bảo máy có khả năng khởi động từ đĩa CD-Rom. Chọn chế độ cài *text*
2. Chọn chế độ cài *text*

```
boot: text
```

3. Lựa chọn ngôn ngữ

Chọn ngôn ngữ mặc định là English



#### 4. Lựa chọn kiểu bàn phím

Lựa chọn kiểu thể hiện bàn phím là **us**.



#### 5. Màn hình chào mừng

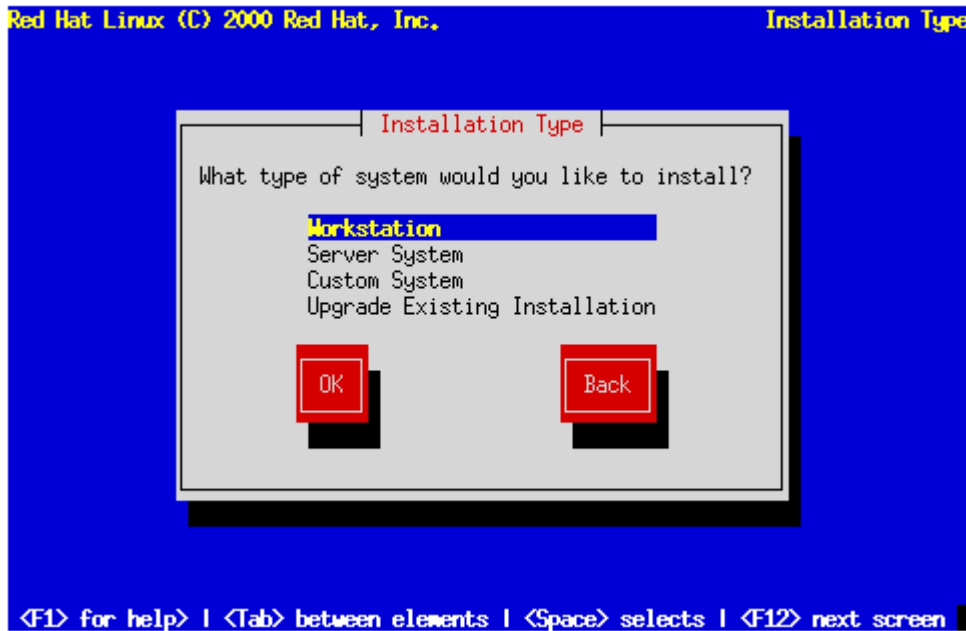
Sau khi đã lựa chọn xong ngôn ngữ cài đặt, bàn phím và phương pháp cài đặt, màn hình chào mừng xuất hiện. Bấm OK để tiếp tục.



#### 6. Chọn kiểu cài đặt

Hộp hội thoại cho phép bạn chọn lựa kiểu cài đặt hệ điều hành Linux RedHat như một Workstation, Server, Custom hay chỉ là nâng cấp phiên bản đã cài đặt.

Chọn kiểu cài đặt là Custom System. Chọn OK để tiếp tục.



## 7. Lựa chọn phần mềm phân chia ổ đĩa

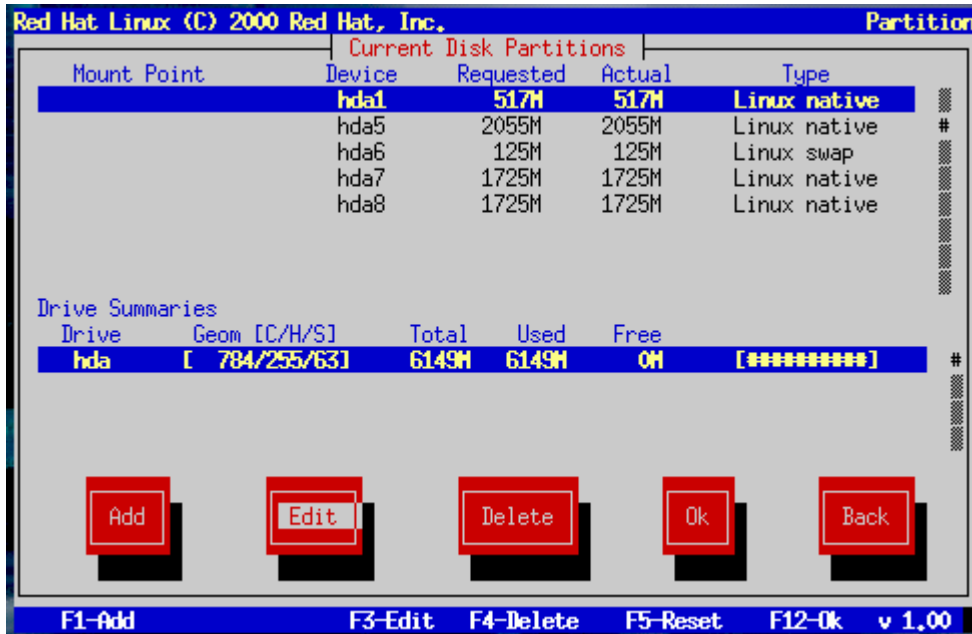
Linux đưa ra cho bạn hai phần mềm để phân chia ổ đĩa dành cho Linux: đó là Disk Druid và fdisk. Chọn Disk Druid để tiếp tục.



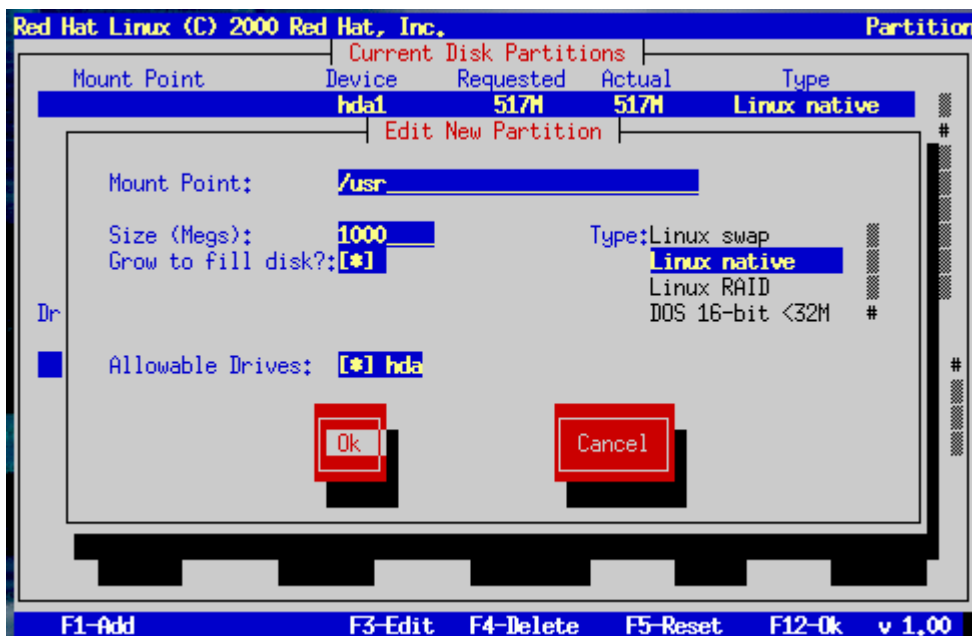
Bạn cần tạo 2 partition để install RedHat, nhớ đừng delete những partition có sẵn trong máy bạn (nếu không thì dữ liệu có sẵn sẽ mất, tốt nhất là bạn nên sao lưu dữ liệu trước cho bảo đảm!). Dùng các chức năng **add**, **edit**, **delete** tạo 1 partition với type là



**Linux swap**, dung lượng bằng dung lượng RAM của máy. Tiếp theo tạo một partition tên "/" với loại **Linux native**, dung lượng ít nhất là 500Mb (tùy theo dung lượng còn trống của đĩa bạn, nếu bạn muốn install trọn gói RedHat thì cần đến khoảng 2288MB). Hãy yên chí là nếu bạn tạo sai (partition kích thước quá lớn, lớn hơn dung lượng còn trống của đĩa) thì RedHat sẽ không cho bạn đi tiếp. Chỉ cần tạo 2 partition này là đủ rồi. Khi nào bạn click được **ext** thì *coi như* là thành công!



Để tạo một partition mới, chọn Add. Màn hình **Edit New Partition** xuất hiện

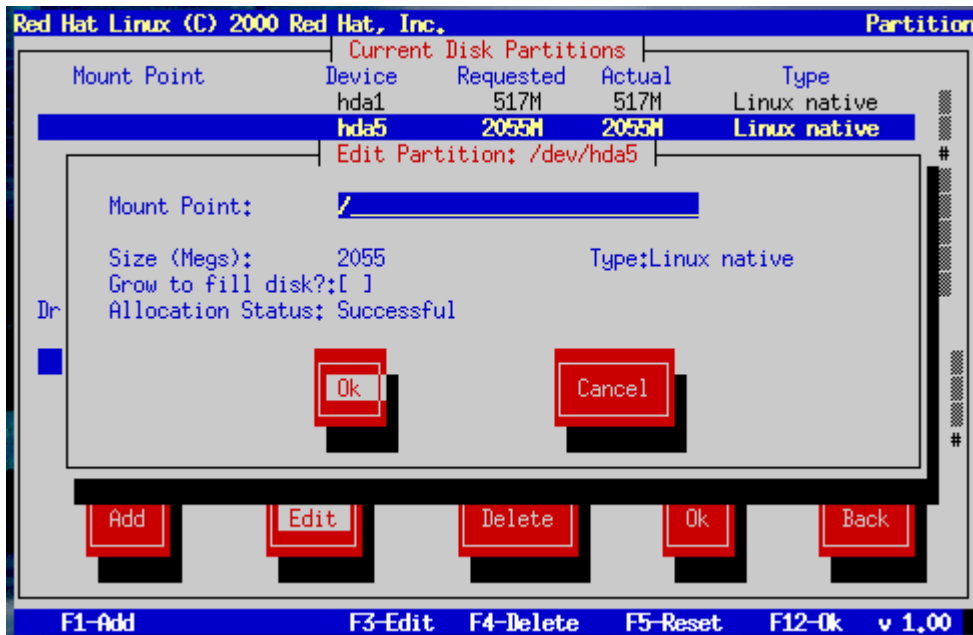


Một số vấn đề có thể xảy ra khi thêm một partition



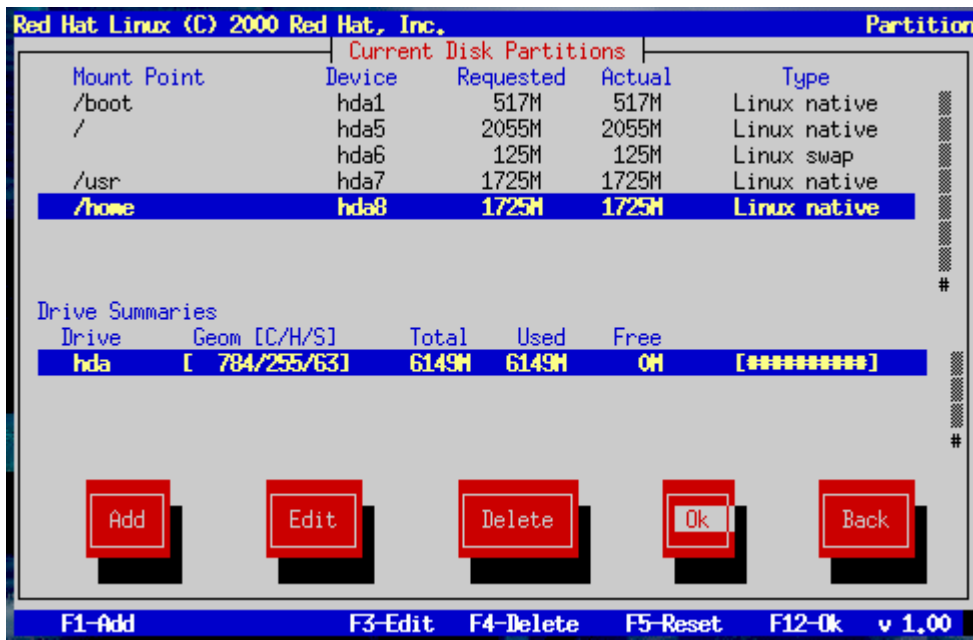
### 8. Hiệu chỉnh một partition

Chọn một partition cần hiệu chỉnh, nhấn Edit, màn hình mới sẽ cho phép bạn thay đổi các thông số của partition đã chọn như kích thước, kiểu, ...



### 9. Hoàn thành việc phân chia đĩa

Chương trình cài đặt sẽ yêu cầu bạn format lại phân vùng vừa tạo, chú ý không chọn những phân vùng dữ liệu quan trọng đối với bạn.



## 10. Khởi tạo LILO

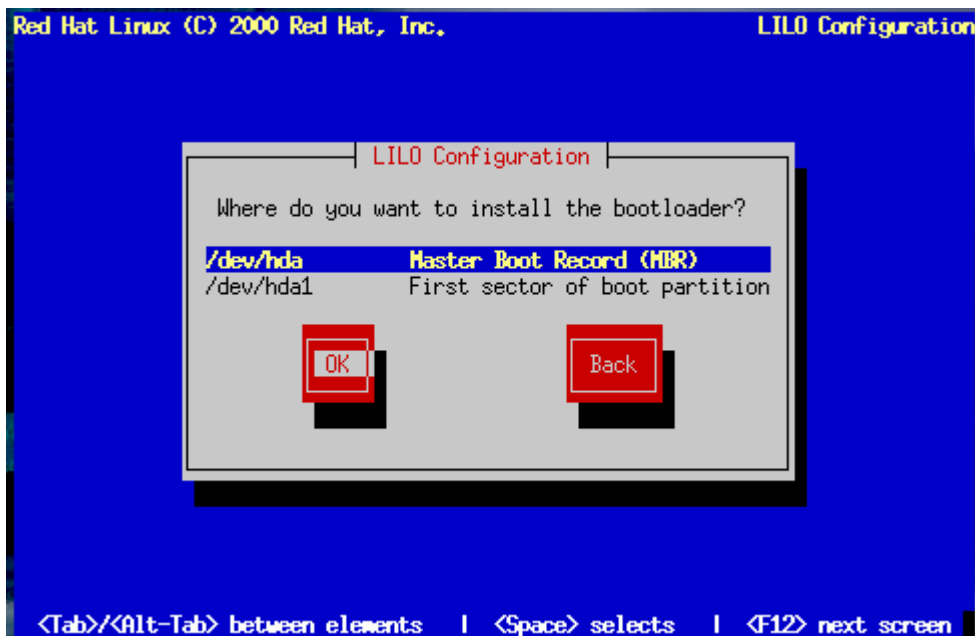
**Linux LO**ader (LILO) cho phép bạn xác định thời gian để khởi tạo Linux hay một hệ điều hành nào khác. Khi khởi tạo cho server, LILO được cấu hình tự động trên Master Boot Record [MBR]. If you are performing a custom-class installation, the **LILO Installation** dialogs let you indicate how or whether to install LILO.

Việc chọn LILO trong cửa sổ **LILO Configuration** cho phép bạn thêm các tùy chọn mặc định vào lệnh boot LILO và các tùy chọn này được chuyển cho Linux kernel tại thời điểm boot.



Chú ý rằng nếu bạn chọn **Skip**, bạn sẽ không thể boot hệ thống Red Hat Linux một cách trực tiếp mà sẽ phải sử dụng phương pháp boot khác (boot disk chẳng hạn) Bạn chỉ nên lựa chọn cách này khi bạn chắc chắn đã có cách khác để boot hệ thống Red Hat Linux của bạn.

Dùng lựa chọn đặt boot loader tại Master Boot Record để khởi tạo ngay hệ điều hành Linux khi bật máy.



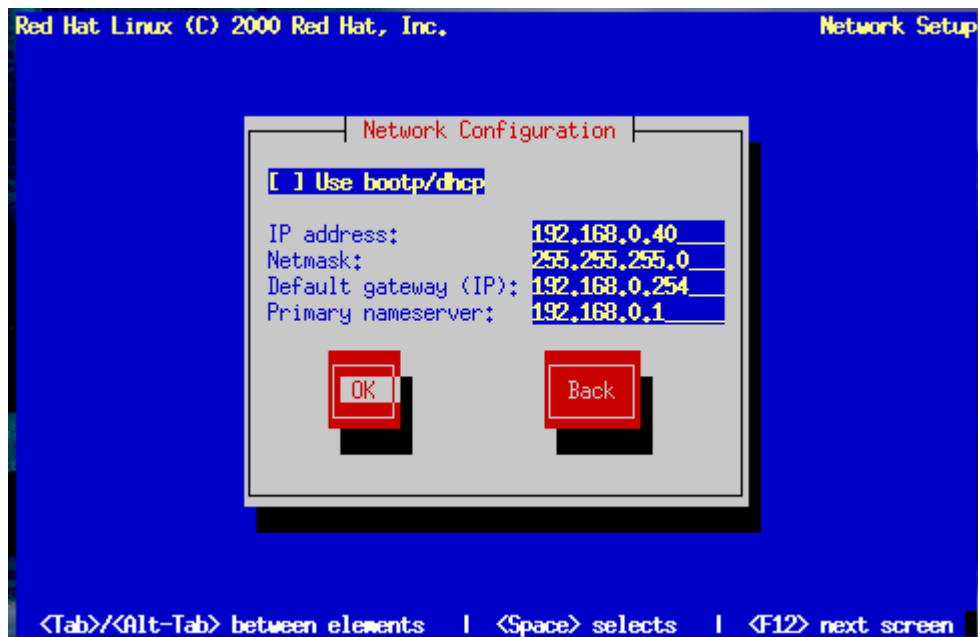
Màn hình này cho phép bạn đặt tên cho máy tính của mình. Bạn có thể thay đổi hostname sau khi đã cài đặt xong bằng lệnh **hostname newname**, trong đó newname là tên mà bạn muốn đặt.



### 11. Cấu hình kết nối mạng

Nếu máy không có card mạng, sẽ không nhận được màn hình này. Thực hiện cấu hình mạng cho máy như sau

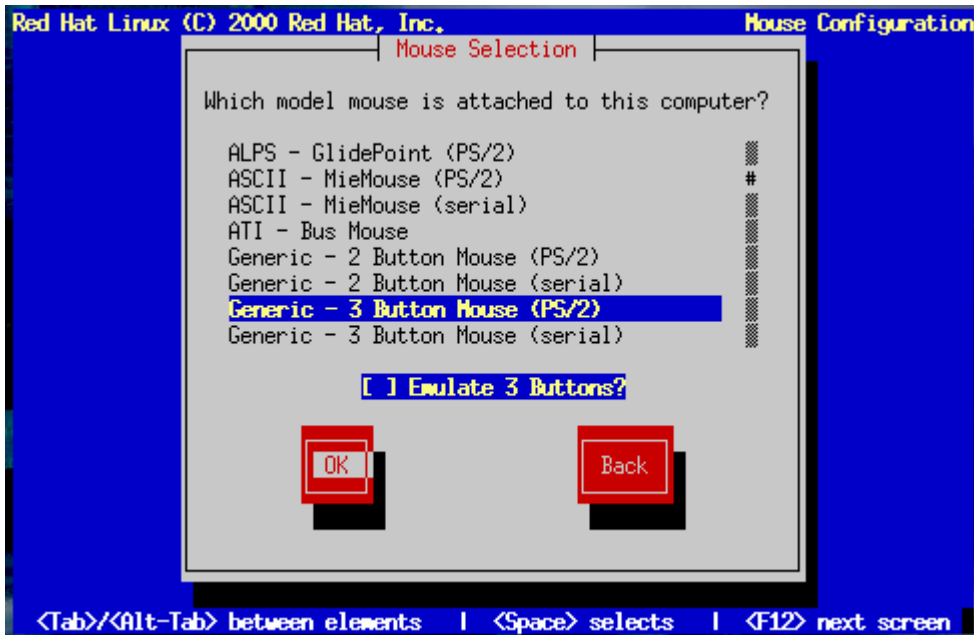
Bỏ lựa chọn **config using DHCP** (chế độ cấp phát địa chỉ IP động), nhập địa chỉ IP, subnetmask theo hướng dẫn của giáo viên hướng dẫn thực hành.



### 12. Cấu hình firewall: chọn **Medium**

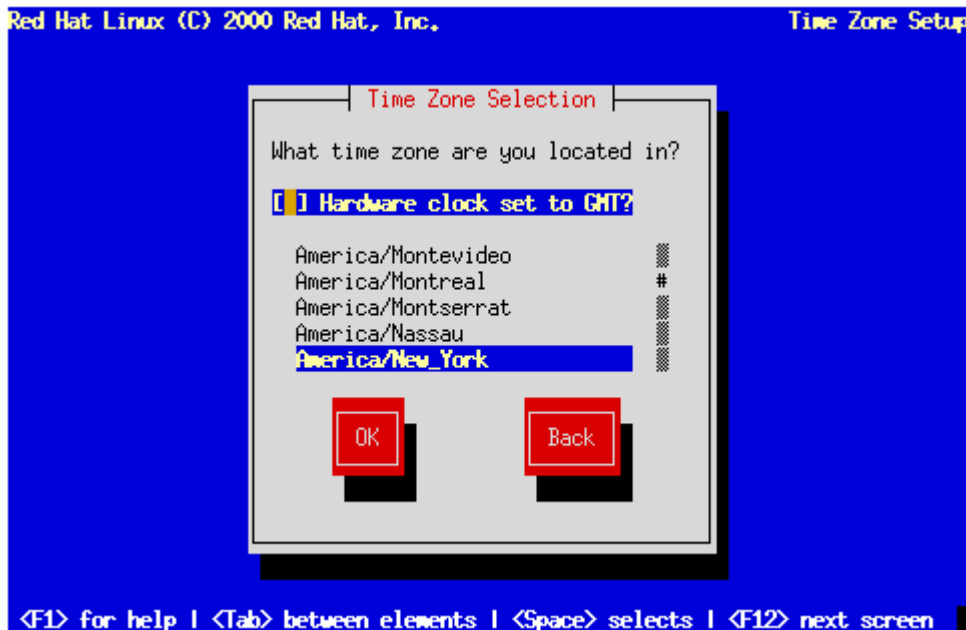
### 13. Cấu hình chuột

Thông thường thì chương trình cài đặt sẽ tự phát hiện loại chuột của máy bạn. Ắt ếu không, bạn hãy chọn loại chuột phù hợp trong danh sách, và nếu bạn không biết chuột của mình loại gì thì cứ để yên, click **Next** để tiếp tục.



Lựa chọn **Emulate 3 Buttons** cho phép bạn sử dụng chuột của bạn như chuột có 2 nút trong đó dùng nút giữa bằng cách bấm hai nút cùng một lúc. Ắt ếu bạn có chuột hai nút, bạn hãy sử dụng chức năng này vì XWindow trở nên dễ dùng nhất với khi chuột có ba nút.

#### 14. Cấu hình Time Zone



Ắt ếu bạn muốn thiết lập đồng hồ cho CMOS theo giờ GMT (Greenwich Mean Time), chọn **Hardware clock set to GMT**. Tuy nhiên, nếu máy tính của bạn sử dụng một hệ

điều hành khác thì việc thiết đặt đồng hồ theo giờ GMT sẽ khiến cho hệ điều hành khác đó hiển thị sai thời gian.

Để đặt giờ Vă , chọn Asia/Saigon

Để thay đổi cấu hình về thời gian sau khi bạn đã cài đặt, bạn có thể dùng lệnh **/usr/sbin/timeconfig**

### 15. Thiết lập mật khẩu root

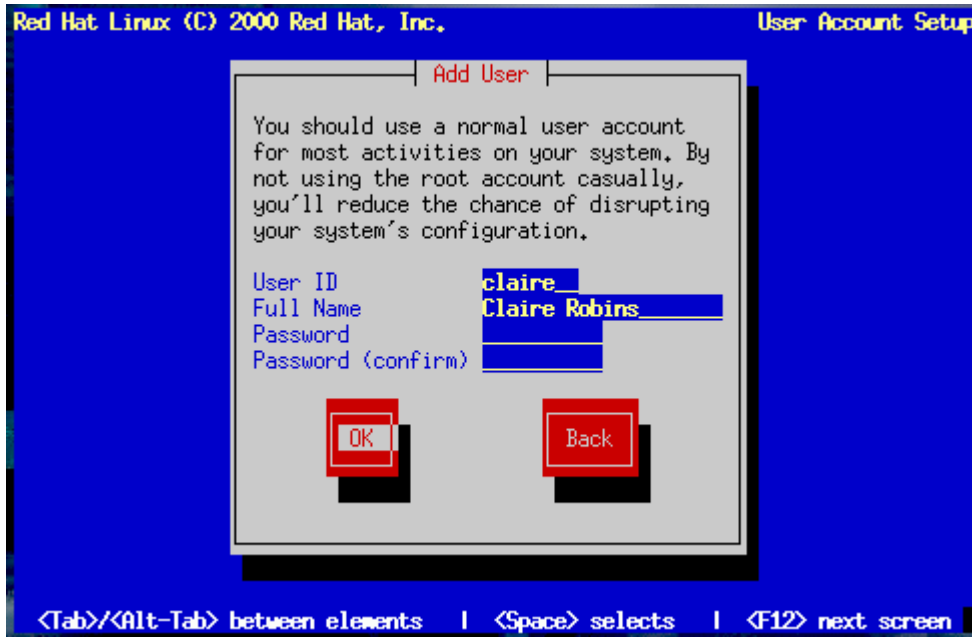
Hộp thoại **Root Password** buộc bạn phải thiết lập một mật khẩu root cho hệ thống của bạn. Bạn sẽ sử dụng mật khẩu này để log vào hệ thống và thực hiện các chức năng quản trị hệ thống của mình.



### 16. Tạo user

Bạn có thể tạo tài khoản user cho chính mình để sử dụng hàng ngày. User root (*superuser*) có đủ quyền truy nhập vào hệ thống nhưng rất nguy hiểm, chỉ nên sử dụng để bảo dưỡng hay quản trị hệ thống.

Mật khẩu của user có phân biệt chữ hoa chữ thường và ít nhất là 6 ký tự.



15. Bạn có thể tạo tiếp nhiều user theo cửa sổ sau:



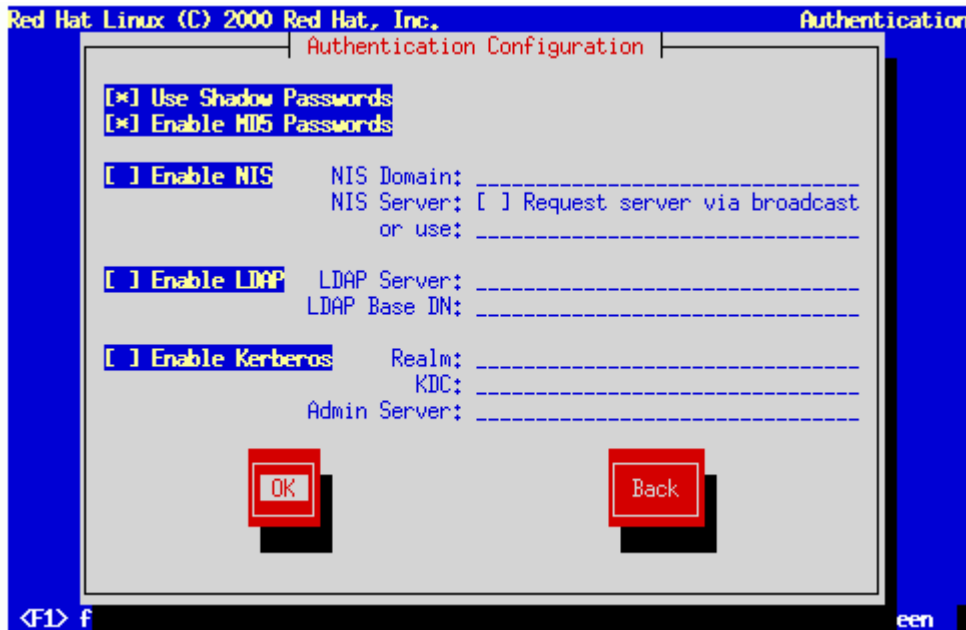
16. Cấu hình xác thực người dùng

Do bạn khởi tạo theo chế độ custom, bước này cho phép bạn cấu hình cách mà hệ điều hành linux của bạn sử dụng để xác thực mật khẩu.

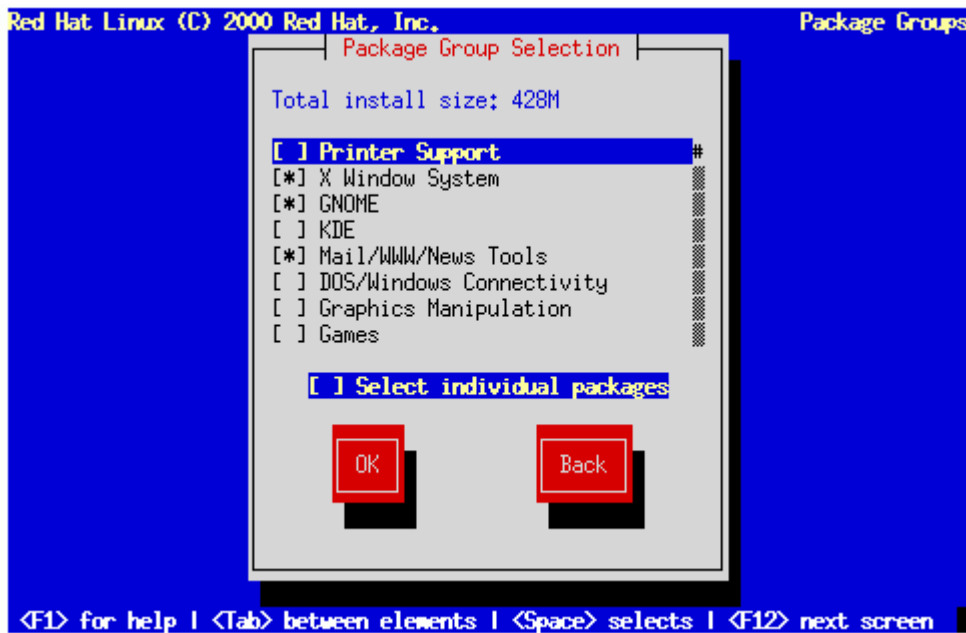
Lựa chọn **Use Shadow Passwords**: mật khẩu của bạn đáng nhẽ nằm trong tệp /etc/passwd sẽ được thay thế bằng thư mục /etc/shadow và chỉ được truy nhập bởi superuser (root)

Tuỳ chọn **Enable MD5 Passwords** -- cho phép mã hóa mật khẩu theo chuẩn MD5.





17. Tiếp theo, bạn có thể chọn lựa các gói tin để cài đặt. Bạn nên chọn các phần mềm, dịch vụ hay sử dụng nhất để cài đặt sẵn trên máy khi khởi động. Tuy nhiên, tuy nhiên, bạn cũng có thể cài đặt sau này tùy theo nhu cầu sử dụng. Các gói tin này nếu được cài đặt sẽ được ghi lại trong tệp /tmp/install.log sau khi khởi tạo lại hệ thống của bạn.

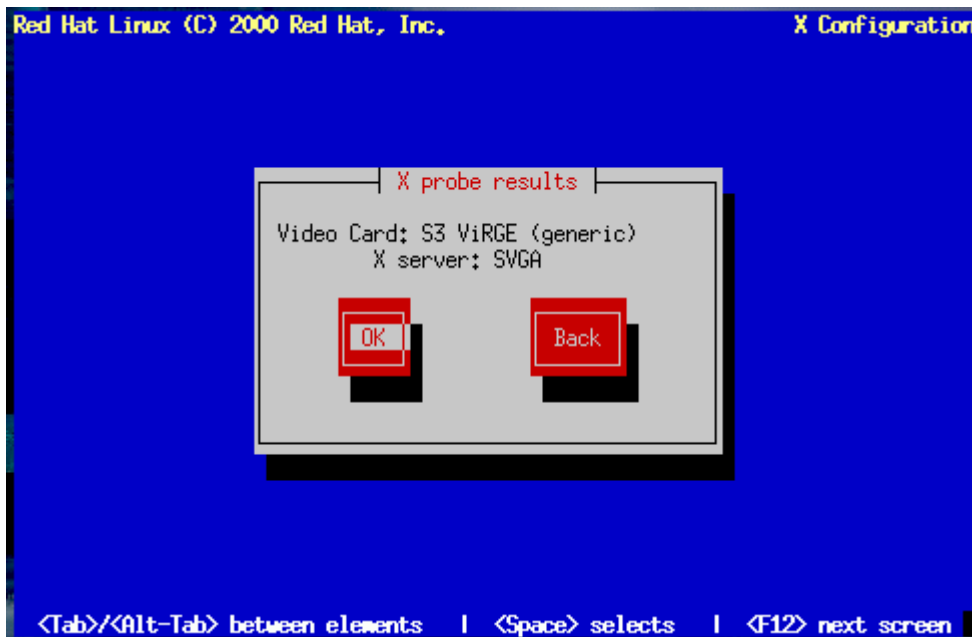


Có thể cài đặt từng gói tin nhỏ hơn bằng cách chọn **Select individual packages** và nhấn OK.



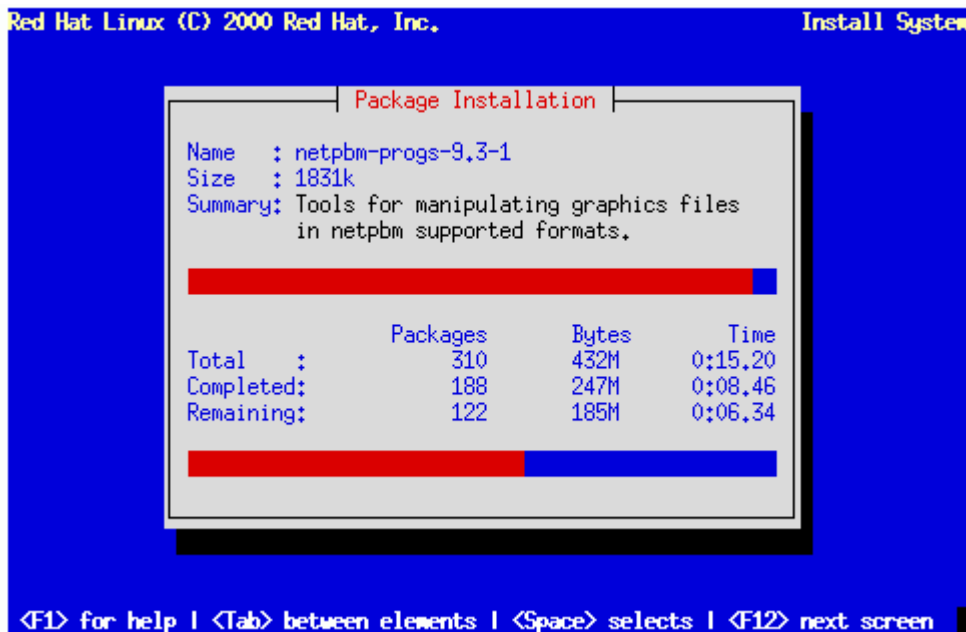
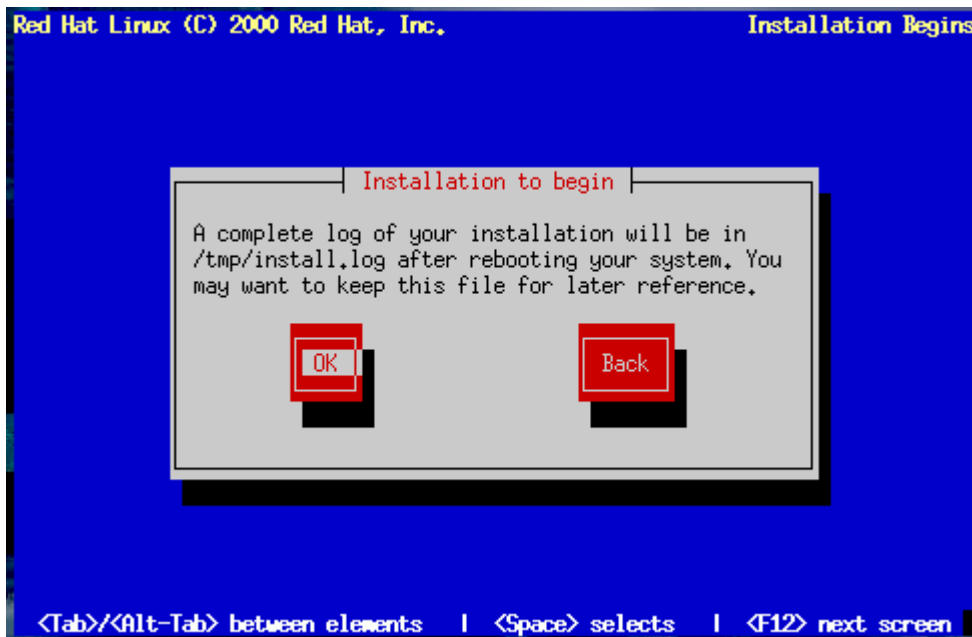
### 18. Cấu hình Video Adapter

Chương trình cài đặt sẽ tự phát hiện video card khởi tạo. ấ hấn OK để tiếp tục.



### 19. Bắt đầu khởi tạo các gói tin:

Quá trình khởi tạo sẽ được ghi vào tệp /tmp/install.log. ấ hấn OK để tiếp tục.



20. Tạo đĩa khởi tạo cho hệ thống (boot disk): Chọn ổ o và tiếp tục.



## 21. Hoàn thành cài đặt

Ả hư vậy là bạn đã hoàn thành xong công việc cài đặt hệ điều hành RedHat 7.0. Bạn hãy rút đĩa ra khỏi ổ CD và nhấn OK để khởi động lại hệ thống.



## 2. Giao tiếp trên môi trường Linux

### 2.1. Trình soạn thảo vi

Chương trình vi là một chương trình soạn thảo mạnh mẽ mà gần như chắc chắn được tìm thấy trên tất cả các hệ điều hành họ UNIX bởi kích thước và khả năng của nó. vi không đòi hỏi nhiều tài nguyên, thêm vào đó là các chức năng soạn thảo cơ bản. vi có thể tìm kiếm, thay thế, và kết nối các file, và nó có ngôn ngữ macro của chính nó, cũng như một số các đặc điểm bổ sung. Có hai chế độ trong vi:

Chế độ thứ nhất là chế độ input. Trong chế độ này, văn bản được đưa vào trong tài liệu, bạn có thể chèn hoặc bổ sung văn bản.

Chế độ thứ hai là chế độ dòng lệnh. Khi ở chế độ này, bạn có thể dịch chuyển trên tài liệu, trộn các dòng, tìm kiếm, ... Bạn có thể thực hiện tất cả các chức năng của vi từ chế độ dòng lệnh ngoại trừ việc nhập vào văn bản. Văn bản chỉ có thể được vào trong chế độ input.

Khi vi khởi động, nó ở chế độ dòng lệnh. bạn có thể chuyển đổi từ chế độ dòng lệnh sang chế độ input bằng cách sử dụng một trong các câu lệnh sau: [aAiIoOcCsSR]. Để trở lại chế độ dòng lệnh bạn chọn phím ESC. Hãy xem các câu lệnh và tác dụng của các câu lệnh trong chế độ dòng lệnh.

Câu lệnh	Tác dụng
Ctrl + D	Chuyển cửa sổ xuống bằng một nửa màn hình
Ctrl + U	Chuyển cửa sổ lên bằng một nửa màn hình
Ctrl + F	Dịch chuyển cửa sổ lên phía trước bằng một màn hình
Ctrl + B	Dịch chuyển cửa sổ về phía sau một màn hình
k hoặc up arrow	Dịch chuyển con trỏ lên một dòng
j hoặc down arrow	Dịch chuyển con trỏ xuống một dòng
l hoặc right arrow	Dịch chuyển con trỏ sang phải một ký tự
h hoặc left arrow	Dịch chuyển con trỏ sang trái một ký tự
Return	Dịch chuyển con trỏ đến vị trí bắt đầu dòng tiếp theo
-	Dịch chuyển con trỏ đến vị trí bắt đầu của dòng trước
w	dịch chuyển con trỏ đến vị trí bắt đầu của từ tiếp theo
b	dịch chuyển con trỏ đến vị trí bắt đầu của từ trước
^ hoặc 0	dịch chuyển con trỏ đến vị trí bắt đầu của dòng hiện tại
\$	dịch chuyển con trỏ đến vị trí kết thúc của dòng hiện tại

i,a	Chèn văn bản ngay trước/sau vị trí con trỏ
o	Mở một dòng mới ngay sau dòng hiện tại
O	Mở một dòng mới ngay trước dòng hiện tại
x	Xóa ký tự sau con trỏ
dw	Xoá một từ (bao gồm cả ký tự trống ngay sau nó)
D	Xoá từ vị trí con trỏ đến kết thúc dòng
d^	Xoá từ vị trí bắt đầu dòng đến vị trí ký tự trống hay ký tự bên trái con trỏ
u	Hủy bỏ thay đổi trước đó
/pattern	Tìm xâu pattern. Theo hướng tiến.
?pattern	Tìm xâu pattern, theo hướng lùi về đầu văn bản.
n, ã	Lặp lại việc tìm kiếm theo cùng hướng / ngược hướng
p, P	Dán đoạn văn bản vừa xoá vào trước / sau con chạy
.	Lặp lại câu lệnh cuối.
dd	Xóa dòng có con trỏ chạy
:w	Ghi lại tất cả các thay đổi của file hiện tại và tiếp tục soạn thảo
:q!	Kết thúc, không lưu trữ bất kỳ thay đổi
:ZZ	Lưu thay đổi của file hiện tại và kết thúc.

## 2.2. Tiện ích mc.

Một khi người dùng có ác cảm với giao diện dòng lệnh của DOS, họ cho rằng các lệnh của Linux cũng khó học. Trong thời kỳ của DOS trước Windows, việc định hướng các tập tin thông qua hệ thống menu và các chương trình quản lý bắt đầu phát triển mạnh, cho dù chúng chỉ dựa trên chế độ text. Một trong số chương trình thông dụng như vậy là orton Commander.

Linux cũng có một chương trình tiện ích với chức năng tương tự như vậy gọi là Midnight Commander (MC). Bạn không phải mất công tìm kiếm MC, phần lớn các nhà phân phối Linux đều cung cấp kèm theo HĐH và nó được cài trong /usr/bin/mc. Chương trình chạy ở cả hai chế độ: text mode và đồ họa (Xterm dưới X Windows).

Sau khi nhập lệnh "mc" để chạy chương trình, bạn sẽ nhìn thấy một cửa sổ được chia đôi như trong hình 1. Midnight Commander hầu như là bản sao của orton

Commander. Phần lớn cách trình bày, phím tắt và các đặc tính đều giống ả C. Sử dụng mouse cũng được hỗ trợ ở chế độ text.

ầu driver mouse được tải khi khởi động (phần lớn các nhà cung cấp Linux đều làm như vậy), bạn có thể dùng mouse để truy cập menu và các tập tin. ả hấn vào file thực thi để chạy, nhấn vào thư mục để chuyển vào đó, hoặc nhấn vào tập tin với phần đuôi mở rộng để mở nó với chương trình tương ứng. Bằng cách nhấn nút phải chuột vào một tập tin, bạn chọn hoặc bỏ chọn tập tin đó. Bạn có thể thực hiện tìm tên file bằng nhấn tổ hợp phím Ctrl-S và trên file với Alt. Sau đây là những phím lệnh cơ bản:



F1: Trợ giúp

F2: Menu người dùng

F3: Xem các tập tin được chọn

F4: Hiệu đính tập tin

F5: Copy tập tin

F6: Đổi tên, chuyển tập tin

F7: Tạo thư mục

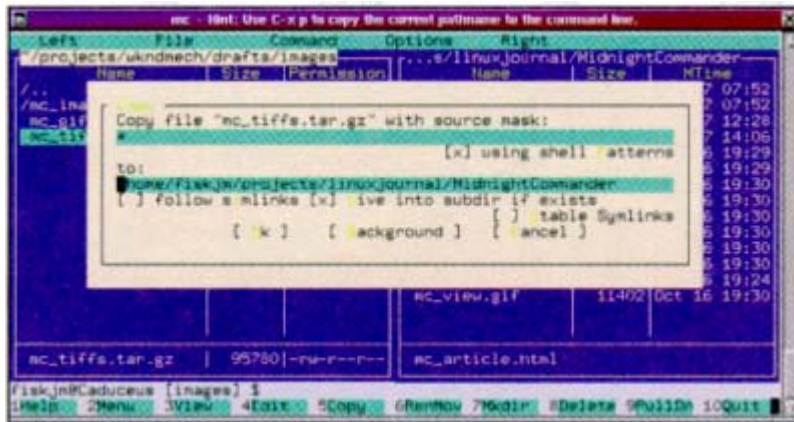
F8: Xoá tập tin

F9: Gọi menu thả xuống (pull-down)

F10: Thoát khỏi Midnight Commander

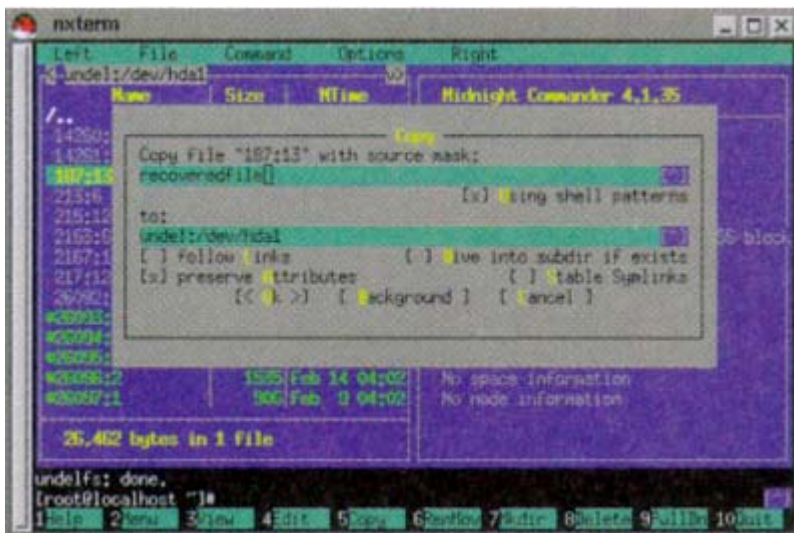
Midnight Commander hỗ trợ một số hệ thống tập tin ảo, nghĩa là bạn có thể xem file không chỉ trên các đĩa cứng cục bộ. Bạn cũng có thể xem các kiểu tập tin nén khác nhau, như .tar, .tgz, .zip, .lha, .rar, .zoo và thậm chí cả .rpm và .deb (các dạng thức tập tin nén của Red Hat và Debian. Việc xem các tập tin được thực hiện thông qua hệ

thống tập tin mạng của UỖ IX (UỖ IX ỗ etwork File System - ỗ FS), Midnight Commander có thể hoạt động như một máy khách ftp bằng cách đưa liên kết FTP vào menu.



### Có thể hồi phục các tập tin đã xóa trong Linux?

Midnight Commander cho thấy rằng vấn đề chúng ta nói đến trong phần trước (PC World Vỗ số 7/1999 trang 95) - không có cách nào hồi phục được các tập tin bị xoỏ trong Linux - là không hoàn toàn chính xác. ỗ ếu bạn sử dụng phần mở rộng ext2, hệ thống tập tin cơ bản trong Linux và cấu hình hệ thống để cho phép hồi phục tập tin bị xoỏ thì trên thực tế bạn có thể truy cập vào các file đã xoỏ.

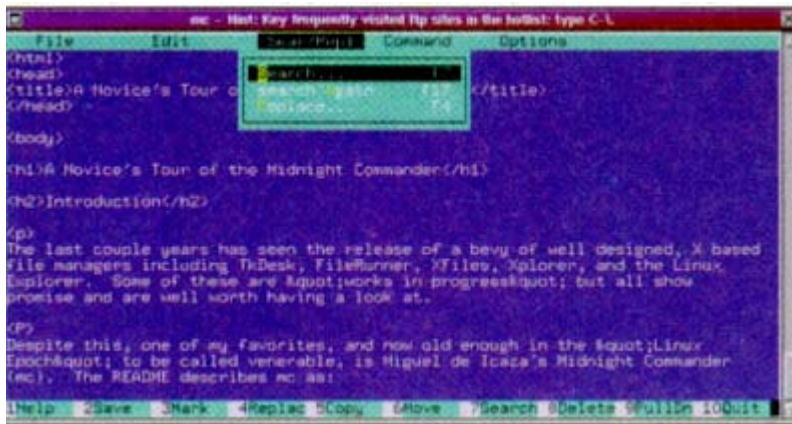


Với Midnight Commander, bạn nhập dòng "undel:/" trước tên tập tin, ví dụ "undel:/dev/hda1". Sau đó bạn có thể xem các tập tin bị xoỏ (hình 3). Chọn tập tin bạn muốn hồi phục bằng chuột hay bàn phím và dùng F5 để copy chúng vào thư mục đích nào đó. Trở ngại duy nhất ở đây là thông tin về tên file bị mất, bởi vậy bạn phải cố xác định được tập tin nào bạn muốn hồi phục.



Midnight Commander bao gồm cả chương trình xem và soạn thảo tập tin. Cả hai đều có thể làm việc với file văn bản và file nhị phân (text và binary) và hiển thị các ký tự 8-bit ngoài 128 ký tự mã ASCII.

Trình soạn thảo có giao diện menu và giống Windows ở nhiều phím soạn thảo cơ bản: nhấn Shift và phím mũi tên để chọn text, nhấn Ctrl-Ins để copy text và Shift-Ins để dán text. Bạn có thể ghi macro với Ctrl-R cũng như thực hiện những tìm kiếm theo từ thông thường.



Midnight Commander có một số tính năng mà DOS không có. Bạn có thể thay đổi quyền sở hữu tập tin và xem chi tiết về quyền truy cập tập tin. MC còn có khả năng quản lý quy trình, cho phép bạn xem những quá trình đang được thực hiện ở chế độ nền, và bạn có thể dừng chúng, khởi động lại hoặc tắt chúng hoàn toàn.

Midnight Commander có rất nhiều tính năng mà không thể liệt kê hết trong bài này. Trên Internet có nhiều Web site dành riêng cho Midnight Commander, chẳng hạn như [www.gnome.org/mc](http://www.gnome.org/mc), bạn có thể tham khảo chi tiết hơn.

## 2.3. Các câu lệnh cơ bản trên Linux

### 2.3.1. Hiểu biết về các câu lệnh trong Linux

#### 2.3.1.1. Sử dụng các ký tự đại diện

Khi bạn sử dụng các câu lệnh về file và thư mục, bạn có thể sử dụng các ký tự đặc biệt được gọi là các ký tự đại diện để xác định tên file, tên thư mục. Ví dụ, để đưa ra danh sách tất cả các file có tên kết thúc bằng .c, bạn sử dụng câu lệnh sau:

```
ls *.c
```

Kí tự \* là một ký tự đại diện, khi shell thông dịch, nó sẽ thay \* bằng tất cả các tên file có kết thúc bằng .c. Bảng bên dưới chỉ ra một số các ký tự đại diện thường được sử dụng:

*	Tương ứng với thứ tự bất kỳ của một hay nhiều ký tự
?	Tương ứng với một ký tự bất kỳ
[]	Tương ứng với một trong những ký tự trong ngoặc hoặc giới hạn

Ví dụ:

- Jo\* : Các file bắt đầu với Jo
- Jo\*y : Các file bắt đầu với Jo và kết thúc với y
- Ut\*1\*s.c : Các file bắt đầu với Ut, chứa một ký tự l và kết thúc với s.c
- ?h : Các file bắt đầu với một ký tự đơn, theo sau bởi .h
- Doc[0-9].txt : Các file có tên Doc0.txt, Doc1.txt ....Doc9.txt
- Doc0[A-Z].txt : Các file có tên Doc0A.txt, Doc0B.txt ...Doc0Z.txt

### 2.3.1.2. Cơ bản về các biểu thức chính quy

Các biểu thức chính quy được sử dụng bởi phần lớn các câu lệnh. Chúng cung cấp một cách thuận tiện và đồng nhất để xác định các mẫu phù hợp. Chúng tương tự với các ký tự đại diện, nhưng chúng mạnh hơn rất nhiều. Chúng cung cấp một phạm vi rộng các mẫu lựa chọn. các ký tự đặc biệt được đưa ra ở dưới đây là các biểu thức chính quy thường được sử dụng:

Ký tự	Ý nghĩa
.	Tương ứng với một ký tự đơn bất kỳ ngoại trừ dòng mới
*	Tương ứng với không hoặc nhiều hơn các ký tự đứng trước
^	Tương ứng với bắt đầu của một dòng
\$	Tương ứng với kết thúc một dòng
\<	Tương ứng với bắt đầu một từ
\>	Tương ứng với kết thúc một từ
[]	Tương ứng với một trong các ký tự bên trong hoặc một dãy các ký tự
[^]	Tương ứng với các ký tự bất kỳ không nằm trong ngoặc
\	Lấy ký hiệu theo sau dấu gạch ngược

Trước tiên, trong một biểu thức chính quy, một ký tự bất kỳ không có ý nghĩa riêng cho chính nó. Ví dụ để tìm kiếm các dòng chứa chữ “foo” trong file data.txt sử dụng câu lệnh sau:

```
grep foo data.txt
```

Để tìm kiếm các dòng bắt đầu bằng từ “foo”, ta sử dụng câu lệnh:

```
grep '^foo' data.txt
```

Việc sử dụng dấu trích dẫn đơn nói cho shell để nguyên các ký tự và bỏ qua chúng trong chương trình. Việc sử dụng dấu trích dẫn đơn là cần thiết khi sử dụng các ký tự đặc biệt.

```
grep 'hello$' data.txt
```

Các dòng bất kỳ kết thúc với chuỗi “hello” được trả lại. Để tìm kiếm một mẫu bắt đầu bằng một từ, sử dụng \`<`. Ví dụ:

```
grep '\<ki' data.txt
```

biểu thức ở bên trên sẽ cho phép tìm kiếm các từ bắt đầu bằng ‘ki’ trong file data.txt. Để tìm kiếm mẫu ‘wee’ kết thúc của một từ, sử dụng:

```
grep 'wee\>' data.txt
```

Ở bảng bên trên, chú ý rằng dấu chấm sẽ phù hợp với một ký tự bất kỳ trừ dòng mới. Điều này có thể được thao tác, nếu chúng ta tìm kiếm tất cả các dòng chứa ký tự ‘C’ được theo sau bởi hai ký tự và kết thúc bởi ký tự ‘s’, biểu thức chính quy có thể là:

```
grep 'C..s' data.txt
```

Biểu thức này có thể có các mẫu phù hợp như ‘Cats’, ‘Cars’ và ‘Cris’ nếu chúng được chứa trong file data.txt. Nếu bạn muốn xác định một dãy các ký tự, sử dụng một dấu gạch nối phân biệt ký tự bắt đầu và ký tự kết thúc của dãy. Khi bạn xác định một dãy, thứ tự phải giống như mã ASCII. Ví dụ, để tìm kiếm tất cả các dòng chứa một ký tự “B” theo sau bởi một ký tự thường sử dụng:

```
grep 'B[a-z]' data.txt
```

Cũng có thể xác định nhiều giới hạn trong cùng một mẫu:

```
grep 'B[A-Za-z]' data.txt
```

### 2.3.2. Các câu lệnh về thư mục và file

- **Lệnh cat**

Cú pháp: `cat file [>|>] [destination file]`

Lệnh `cat` sẽ hiển thị nội dung của một file ra thiết bị ra chuẩn. Nó thường hữu ích để kiểm tra nội dung của một file bằng sử dụng câu lệnh `cat`. Đối số mà bạn đưa vào lệnh `cat` là file bạn muốn xem. Để xem toàn bộ nội dung của một file:

```
cat name
```

Lệnh `cat` cũng có thể trộn nhiều file đang tồn tại vào một file:

```
cat name1 name2 name3 > allnames
```

Ví dụ này sẽ kết hợp các file : `name1`, `name2` và `name3` cho file cuối cùng `allnames`. Thứ tự của việc trộn được thiết lập bởi thứ tự của các file được đưa vào trên dòng lệnh. Sử dụng lệnh `cat`, chúng ta có thể bổ sung một file vào một file khác đang tồn tại. Trong trường hợp bạn quên thêm `name4` vào câu lệnh trước, chúng ta vẫn có thể nhận được kết quả mong muốn bằng cách thực hiện lệnh:

```
cat name4 > allnames
```

Lệnh này sẽ bổ sung nội dung của file `name4` vào `allnames`

- **Lệnh chmod**

Cú pháp: `chmod [-R] permission-mode file hoặc thư mục`

Lệnh `chmod` dùng để thay đổi quyền truy cập file hoặc thư mục. Ví dụ:

```
chmod myscript.pl
```

Để thay đổi quyền của một thư mục và tất cả các file, các thư mục con của thư mục đó sử dụng câu lệnh:

*chmod -R 744 public\_html*

- **Lệnh chown**

Cú pháp: `chown [ -fhR ] Owner [ :Group ] { file ... | thư mục... }`

Lệnh `chown` thay đổi quyền sở hữu file hay thư mục. Giá trị của khai báo `Group` có thể là một ID của nhóm người sử dụng hoặc tên của nhóm người sử dụng được tìm thấy trong file `/etc/group`. Chỉ người sử dụng `root` mới có quyền thay đổi quyền sở hữu đối với file. Chi tiết về các tùy chọn được chỉ ra ở bên dưới:

-f: ngăn chặn tất cả các thông báo lỗi trừ các thông báo sử dụng

-h: thay đổi quyền sở hữu của liên kết tượng trưng nhưng không thay đổi quyền sở hữu của file mà được chỉ đến bởi liên kết tượng trưng đó.

-R: thay đổi quyền sở hữu của thư mục, các file và các thư mục con bên trong thư mục hiện tại được chỉ ra

- **Lệnh clear**

Xoá màn hình, trả lại dấu nhắc dòng lệnh ở phía trên của màn hình

*clear*

- **lệnh cmp**

Cú pháp: `cmp [ -ls ] file1 file2`

Lệnh này so sánh nội dung của hai file. Nếu không có sự khác nhau nào, lệnh `cmp` sẽ kết thúc một cách yên lặng, tùy chọn `-l` sẽ in ra số byte và các giá trị khác nhau giữa hai file. Tùy chọn `-s` không hiển thị cái gì cả, nó chỉ trả lại trạng thái chỉ ra rằng sự tương đương giữa hai file. Giá trị 0 được trả lại nếu các file giống hệt nhau, giá trị bằng 1 nếu hai file khác nhau và lớn hơn 1 nếu lỗi xuất hiện khi thực hiện câu lệnh.

- **Lệnh cp**

Cú pháp: `cp [ -R ] file_hoặc_thư_mục file_hoặc_thư_mục`

Lệnh `cp` sẽ sao chép một file từ thư mục nguồn đến thư mục đích được đưa vào. Để sao chép toàn bộ các file và các thư mục con bên trong thư mục mong muốn, bạn sử dụng câu lệnh `cp` với tùy chọn `-R`

- **Lệnh du**

Lệnh này tổng kết việc sử dụng đĩa. Nếu bạn xác định một thư mục, lệnh `du` sẽ báo cáo việc sử dụng đĩa cho chính các thư mục đó.

Cú pháp: `du [ -ask ] tên_file`

Tuỳ chọn `-a` sẽ đưa ra màn hình kích thước của mỗi thư mục và file

Tuỳ chọn `-s` sẽ chỉ in ra tổng cộng

Tuỳ chọn `-k` sẽ in ra tất cả các kích thước file theo kilobytes

- **Lệnh file**

Cú pháp: `file filename`

Câu lệnh xác định kiểu của file. ả ếu file không phải là file thông thường, kiểu của file được xác định.

- **Lệnh find**

Câu lệnh `find` tìm các file và các thư mục.

Cú pháp : `find [path] [-type fd] [-name mẫu] [-atime [+/-] số_ngày] [-exec câu_lệnh {} \;] [-empty]`.

Ví dụ:

```
find . -type d
```

Câu lệnh trả lại tất cả các thư mục con trong thư mục hiện tại. Tuỳ chọn `-type` xác định kiểu, `d` cho các thư mục, `f` cho các file hay `l` cho các liên kết.

```
find . -type f -name "*.txt"
```

Lệnh này sẽ tìm tất cả các file văn bản có phần mở rộng `“.txt”` trong thư mục hiện tại và cả trong các thư mục con.

```
find . -type f -name "*.txt" -exec grep -l 'magic' {} \;
```

Câu lệnh này sẽ tìm kiếm tất cả các file văn bản (kết thúc với phần mở rộng `.txt`) trong thư mục hiện tại và các thư mục con có chứa từ `“magic”`.

```
find . -type f empty
```

Hiển thị tất cả các file rỗng trong thư mục hiện tại.

- **Lệnh grep**

Cú pháp: `grep [-viw] mẫu file`

Lệnh `grep` cho phép bạn tìm kiếm một hoặc nhiều file có các mẫu ký tự đặc biệt. Mỗi dòng của mỗi file chứa các mẫu được hiển thị trên màn hình. Câu lệnh `grep` hữu ích khi bạn có nhiều file và bạn muốn tìm ra file chứa từ hoặc câu xác định. Sử dụng tùy chọn `-v`, bạn có thể hiển thị các file không chứa một mẫu. Ví dụ, để chọn các dòng trong `data.txt` không chứa từ “the” ta thực hiện:

```
grep -vw 'the' data.txt
```

nếu tùy chọn `-w` không được xác định thì bất kỳ các từ chứa “the” đều phù hợp như “together”. Tùy chọn `-w` được xác định buộc mẫu phải là toàn bộ một từ. Cuối cùng, tùy chọn `-i` bỏ qua sự khác nhau giữa các ký tự chữ hoa và ký tự chữ thường khi tìm kiếm mẫu.

- **Lệnh head**

Cú pháp: `head [-count | -n number] filename`

Câu lệnh này sẽ hiển thị vài dòng đầu tiên của một file. Bởi mặc định, 10 dòng đầu của một file được hiển thị. Tuy nhiên, bạn có thể sử dụng các tùy chọn để xác định số dòng hiển thị. Ví dụ:

```
head -2 doc.txt
```

sẽ hiển thị hai dòng đầu tiên.

- **Lệnh ln**

Cú pháp: `ln [-s] file_nguồn đích`

Lệnh `ln` tạo các liên kết cứng và mềm. Các liên kết cứng được tạo sử dụng lệnh `ln` không có tùy chọn `-s`. Ví dụ:

```
ln ./www ./public_html
```

Một liên kết cứng có hạn chế, nó không thể tạo liên kết đến một thư mục khác, và một liên kết cứng không thể liên kết đến một file trên một hệ thống file khác. Sử dụng tùy chọn `-s` bạn có thể tạo một liên kết mềm, loại bỏ các giới hạn này.

```
ln -s /dev/fs02/jack/www /dev/fs01/foo/public_html
```

Ở đây chúng ta đã tạo một liên kết mềm giữa thư mục `www` trên hệ thống file 2 và một file mới được tạo trên hệ thống file 1.

- **Lệnh locate**

Cú pháp : `locate từ_khoá`

Câu lệnh `locate` tìm đường dẫn đến một file đặc biệt hay một câu lệnh. Lệnh `locate` sẽ tìm kiếm chính xác hay một phần của chuỗi phù hợp. Ví dụ:

```
locate foo
```

Kết quả tìm kiếm sẽ đưa ra các file có tên chứa từ khoá 'foo' theo đường dẫn tuyệt đối hoặc sẽ không đưa ra kết quả nếu không có tên file như vậy.

- **Lệnh ls**

Lệnh `ls` cho phép bạn đưa ra danh sách các file và các thư mục con.

Cú pháp : `ls [-laRl] file_hoặc_thư_mục`

Khi sử dụng tùy chọn `-l`, nó chỉ hiển thị tên file và tên thư mục con của thư mục hiện tại. Khi chọn tùy chọn `-l`, một danh sách các file và thư mục con của thư mục hiện tại được hiển thị với đầy đủ các thông tin về file và thư mục. Tùy chọn `-a` cho phép bạn hiển thị tất cả các file và thư mục (kể cả các file ẩn, tên file bắt đầu bằng dấu chấm) trong thư mục hiện tại. Tùy chọn `-R` sẽ hiển thị tất cả các file và các thư mục con bên trong nó nếu có.

- **Lệnh mkdir**

Cú pháp: `mkdir thư_mục`

Để tạo một thư mục, sử dụng câu lệnh `mkdir`. Chỉ có 2 giới hạn khi chọn tên thư mục, đó là tên của thư mục có thể lên tới 255 ký tự và tên thư mục có thể chứa bất kỳ ký tự nào trừ ký tự `'/'`. Ví dụ:

```
mkdir dir1 dir2 dir3
```



Lệnh trên tạo ra ba thư mục, nằm bên trong thư mục hiện tại.

- **Lệnh mv**

Cú pháp : mv [-if] file\_nguồn file\_đích

Sử dụng lệnh mv để dịch chuyển hay đổi tên các file hay các thư mục. Câu lệnh thực hiện việc dịch chuyển hay đổi tên phụ thuộc vào file\_đích có là một thư mục hay không. Để minh hoạ, chúng ta sẽ đổi tên một thư mục foo thành foobar:

```
mv foo foobar
```

Bởi vì foobar chưa tồn tại, foo sẽ được đổi tên thành foobar. ả ếu câu lệnh sau được thực hiện:

```
mv doc.txt foobar
```

và foobar đã tồn tại, việc dịch chuyển file sẽ được thực hiện sau đó. Tùy chọn -f sẽ xoá các file đích đang tồn tại và không bao giờ nhắc người sử dụng. Tùy chọn -i sẽ nhắc người sử dụng có ghi đè hay không nếu file\_đích đã tồn tại.

- **Lệnh pwd**

Cú pháp: pwd

Câu lệnh này hiển thị tên thư mục hiện tại bao gồm cả đường dẫn tuyệt đối. Ví dụ:

```
pwd
```

Trên màn hình hiển thị :

```
/home/trantu
```

- **Lệnh rm**

Cú pháp: rm [-rif] thư\_mục/file

Để xoá thư mục hoặc file, sử dụng câu lệnh rm. bạn có thể xoá nhiều file sử dụng ký tự đại diện hoặc gõ vào tên các file. Ví dụ:

```
rm doc1.txt doc2.txt doc3.txt
```

Tương ứng với:

```
rm doc[1-3].txt
```

rm là câu lệnh rất mạnh, hãy cẩn thận khi sử dụng lệnh này vì bạn có thể nhầm và xoá đi các file quan trọng. Ắt ều chưa chắc chắn, bạn có thể sử dụng tùy chọn `-i`, hệ thống sẽ nhắc lại cho bạn xác thực mỗi lần xoá một file. Ắt ều như đã chắc chắn file cần xoá, bạn có thể chọn tùy chọn `-f` để không phải nhận các thông tin nhắc bạn xác thực. Tùy chọn `-r` sẽ cho phép bạn xoá toàn bộ các thư mục con.

- **Lệnh tail**

Cú pháp: `tail [-count | -fr] tên_file`

Câu lệnh `tail` hiển thị phần cuối của một file, mặc định nó sẽ hiển thị 10 dòng cuối cùng của file. Để hiển thị 50 dòng cuối cùng của file `doc.txt`, bạn có thể sử dụng câu lệnh:

```
tail -50 doc.txt
```

Tùy chọn `-r` sẽ thực hiện công việc ngược lại, mặc định nó sẽ hiển thị tất cả các dòng trừ 10 dòng cuối cùng. Tùy chọn `-f` hữu ích khi bạn đang giám sát một file. Với tùy chọn này, `tail` sẽ chờ cho dữ liệu mới được ghi vào file. Khi dữ liệu mới được thêm vào file, `tail` sẽ hiển thị dữ liệu lên màn hình. Để dừng lệnh `tail` khi đang giám sát file, chọn tổ hợp phím `Ctrl + C` bởi vì lệnh `tail` không tự dừng được.

### 2.3.3. Các câu lệnh nén dữ liệu

- **Lệnh compress**

Cú pháp: `compress [-v] file`

Câu lệnh `compress` sẽ cố gắng giảm kích thước của một file sử dụng. Các file được nén sẽ được thay thế bởi một file có phần mở rộng `.Z`. Tùy chọn `-v` sẽ hiển thị phần trăm dung lượng giảm của một file được nén và sẽ nói cho bạn tên của file mới:

```
compress -v inbox
```

trên màn hình sẽ hiển thị

inbox: Compression: 37.20% - replaced with inbox.Z

- **Lệnh gunzip**

Cú pháp: `gunzip [-v] files`

Để giải nén các file về dạng nguyên bản, sử dụng lệnh `gunzip`, sẽ cố gắng giải nén các file có phần mở rộng: `.gz`, `-gz`, `.z`, `-z`, `_z`, `.Z`, hoặc `tgz`. Tùy chọn `-v` sẽ hiển thị kết quả đẹp khi giải nén các file. Ví dụ:

```
gunzip -v README.txt.gz
```

- **Lệnh gzip**

Cú pháp: `gzip [-rv9] file`

Lệnh `gzip` là một chương trình nén khác. Nó được biết đến là chương trình nén có tỉ lệ nén tốt nhất. Các file được nén bởi lệnh `gzip` sẽ được thay thế bởi các file có phần mở rộng `.gz`. Tùy chọn `-9` có tốc độ nén tốt nhất. Tùy chọn `-v` cho phép hiển thị đẹp trên màn hình. Kích thước, tổng số và tỉ lệ nén được đưa ra danh sách cho mỗi file. Tùy chọn `-r` sẽ nén tất cả các file trong mỗi thư mục theo cùng một cách.

- **Lệnh tar**

Cú pháp: `tar [c] [x] [v] [z] [f tên_file] tên_file_hoặc_thư_mục`

Lệnh `tar` cho phép bạn nén nhiều file và thư mục vào một file `.tar`. Nó cũng cho phép bạn giải nén các file và các thư mục từ một file nén. Ví dụ:

```
tar cf source.tar *.c
```

Câu lệnh này sẽ tạo một file `source.tar`, chứa tất cả các file mã nguồn C (có phần mở rộng `.c`) trong thư mục hiện tại.

```
tar cvf source.tar *.c
```

Tùy chọn `-v` ở đây cho phép bạn xem các file đã được nén

```
tar cvzf backup.tar.gz important_dir
```

Ở đây, tất cả các file và các thư mục con của thư mục `important_dir` được nén trong một file được gọi là `backup.tar.gz`. Chú ý rằng file này cũng được nén do có tùy chọn `z`, và do đó kết quả là file có phần mở rộng là `.gz`. Thông thường phần mở rộng `.tar.gz` được viết ngắn thành `.tgz`. Để giải nén các file, ví dụ như `backup.tar`, bạn sử dụng câu lệnh:

```
tar xf backup.tar
```

Để giải nén một file có phần mở rộng `.tgz` hay `.tar.gz`, bạn thực hiện câu lệnh sau:

```
tar xzf backup.tgz
```

- **Lệnh uncompress**

Cú pháp: `uncompress [-v] file`

Khi một file được nén sử dụng câu lệnh `compress`, để giải nén bạn sử dụng câu lệnh `uncompress`. Lệnh `uncompress` giải nén các file có phần mở rộng `.Z`, vì vậy cú pháp của nó tương tự như lệnh `compress`

```
uncompress -v inbox.Z
```

- **Lệnh unzip**

Cú pháp: `unzip file`

Lệnh này sẽ giải nén các file có phần mở rộng `.zip`. Các file này có thể được nén với lệnh `zip`.

- **Lệnh zip**

Cú pháp : `zip [-ACDe9] file`

Đây là chương trình nén file theo định dạng nổi tiếng tương thích với nhiều hệ điều hành. Các file được nén với lệnh `zip` có phần mở rộng `.zip`.

- **Lệnh mount**

Cú pháp: `mount -a [-t fstype] [-o option] device directory`

Lệnh `mount` được sử dụng để gắn các thiết bị với hệ thống, các tùy chọn thông thường thường có trong file `/etc/fstab`. Ví dụ:

```
/dev/hda6 /intranet ext2 defaults 1 2
```

Âu dòng bên trên được tìm thấy trong `/etc/fstab`, bạn có thể gắn hệ thống file được lưu trong phân vùng `/dev/hda6` như sau:

```
mount /intranet
```

Cùng một hệ thống file, câu lệnh sau đây là tương tự:

```
mount -t ext2 /dev/hda6 /intranet
```

Tùy chọn `-t` được sử dụng để xác định kiểu file hệ thống. Để gắn tất cả các hệ thống file có trong `/etc/fstab` sử dụng tùy chọn `-a`. Ví dụ:

```
mount -a -t ext2
```

Thông thường người sử dụng chọn tùy chọn `-o` là `ro` (chỉ đọc) hoặc `rw` (đọc ghi). Ví dụ:

```
mount -t ext2 -o ro /dev/hda6 /secured
```

- **Lệnh umount**

Cú pháp : `umount -a [-t fstype]`

Lệnh `umount` ngược lại với lệnh `mount`. Ví dụ

```
umount /cdrom
```

### 2.3.4. Các câu lệnh quản lý tiến trình

- **Lệnh bg**

Cú pháp: `bg`

Đây là kịch bản shell được xây dựng sẵn. Đưa một tiến trình đang chạy về chạy ở sau hậu cảnh (tiến trình nền).

- **Lệnh fg**

Cú pháp: `fg [%job-number]`

Câu lệnh này cho phép bạn chuyển một tiến trình nền lên chạy ở trên tiền cảnh.

Nếu bạn chạy câu lệnh này không có bất kỳ đối số nào, nó sẽ đưa câu lệnh cuối cùng ở sau hậu cảnh lên hiển thị. Ví dụ, nếu có hai câu lệnh chạy ở sau hậu cảnh, bạn có thể chuyển câu lệnh thứ nhất lên chạy trên tiền cảnh bằng câu lệnh:

`fg %1`

- **Lệnh jobs**

Cú pháp: `jobs`

Lệnh này cho phép bạn hiển thị các tiến trình nền đang chạy. Nó ngoài ra còn một số lệnh sẽ được trình bày trong các phần sau.

### 3. Giới Thiệu Hệ Thống Tập Tin, Thư Mục

#### 3.1. Giới thiệu

Trong linux file được tổ chức thành các thư mục, theo mô hình phân cấp. Tham chiếu đến một file bằng tên và đường dẫn. Các câu lệnh thao tác file cho phép thực hiện các chức năng như dịch chuyển, sao chép toàn bộ thư mục cùng với các thư mục con chứa trong nó...

Có thể sử dụng các ký tự, dấu gạch dưới, chữ số, dấu chấm và dấu phẩy để đặt tên file. Không được bắt đầu một tên file bằng dấu chấm hay chữ số. Những ký tự khác như '/', '?', '\*', là ký tự đặc biệt được dành riêng cho hệ thống. Chiều dài của tên file có thể tới 256 ký tự.

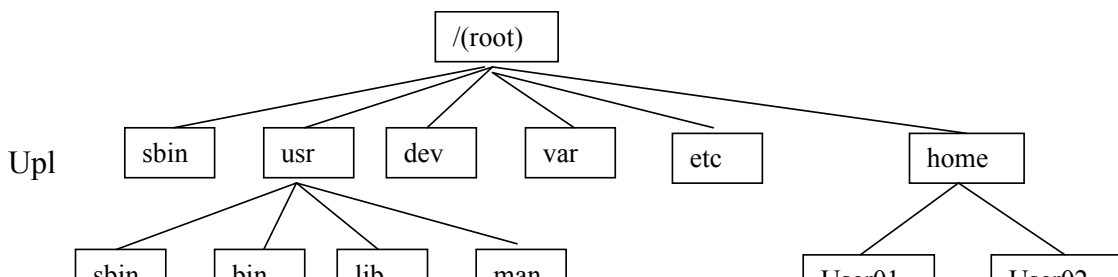
Tất cả các file trong linux có chung cấu trúc vật lý là chuỗi các byte (byte stream). Cấu trúc thống nhất này cho phép linux áp dụng khái niệm file cho mọi thành phần dữ liệu trong hệ thống. Thư mục cũng như các thiết bị được xem như file. Chính việc xem mọi thứ như các file cho phép linux quản lý và chuyển đổi dữ liệu một cách dễ dàng. Một thư mục chứa các thông tin về thư mục, được tổ chức theo một định dạng đặc biệt. Các thành phần được xem như các file, chúng được phân biệt dựa trên kiểu file: *ordinary file*, *directory file*, *character device file*, và *block device file*.

##### 3.1.1. Thư mục chủ

Sau khi đăng nhập hệ thống, người dùng sẽ đứng ở thư mục chủ. Tên của thư mục này giống với tên tài khoản đăng nhập hệ thống. Các file được tạo khi người dùng đăng nhập được tổ chức trong thư mục chủ.

##### 3.1.2. Các thư mục hệ thống

Thư mục root, là gốc của hệ thống file của Linux, chứa một vài thư mục hệ thống. Thư mục hệ thống chứa file và chương trình sử dụng để chạy và duy trì hệ thống. Biểu diễn các thư mục như sau:



Mô tả thư mục

<b>Thư mục</b>	<b>Chức năng</b>
/	Bắt đầu cấu trúc file, gọi là thư mục gốc ( <i>root</i> )
/home	Chứa thư mục gốc ( <i>home</i> ) của người dùng
/bin	lưu trữ tất cả các câu lệnh chuẩn và các chương trình tiện ích
/usr	chứa các file, câu lệnh được hệ thống sử dụng, thư mục này được chia thành các thư mục con khác
/usr/bin	Chứa các câu lệnh hướng người dùng và các chương trình tiện ích
/usr/sbin	Chứa các câu lệnh quản trị hệ thống
/usr/lib	Chứa thư viện cho các ngôn ngữ lập trình
/usr/doc	Chứa tài liệu của linux
/usr/man	Chứa các file chỉ dẫn cho các câu lệnh (man)
/sbin	Chứa các file hệ thống để khởi động hệ thống
/dev	Chứa giao diện cho các thiết bị như đầu cuối và máy in
/etc	Chứa file cấu hình hệ thống và các file hệ thống khác

### 3.2. Các quyền truy cập file/thư mục

Trong Linux, mỗi file hay thư mục được kết hợp với một người sử dụng và một nhóm người sử dụng. Hãy xem một ví dụ:

```
-rwxr-x-r-- 1 trantu trantu 191 Apr 14 14:55 .bash_profile
```

Dòng bên trên được tạo bởi lệnh `ls -l .bash_profile` trên hệ điều hành Linux. Lệnh `ls` đưa ra danh sách các file và thư mục. Tùy chọn `-l` đưa ra danh sách đầy đủ các thông tin về file `.bash_profile`. Bảng bên dưới mô tả các kiểu thông tin đưa ra:

Kiểu thông tin	Thông tin kết xuất
Quyền truy cập file	-rw-rw-r--
Số liên kết	1
Chủ sở hữu (sở hữu file)	Trantu
Chủ nhóm	Trantu
Kích thước file (theo bytes)	191
Ngày sửa đổi cuối cùng	Apr 14
Thời gian sửa đổi cuối cùng	14:55
Tên file	.bash_profile

Ở đây, người sử dụng là trantu. Đây là người sử dụng thường xuyên, có quyền thay đổi các quyền truy cập đối với file này. Chỉ có một người sử dụng khác có quyền thay đổi thuộc tính file này, đó là superuser. Chủ sở hữu file này là trantu, bất kỳ những người sử dụng nào thuộc nhóm trantu cũng có quyền đọc, và thực thi dựa vào quyền của nhóm được đặt bởi người sở hữu. Khi bạn tạo một file trên hệ thống Linux, hệ thống sẽ mặc định người sở hữu file này có tên là tên đăng nhập của bạn và có tên nhóm giống như tên của người sở hữu. Một người sử dụng thông thường không thể gán lại quyền sở hữu một file hay thư mục cho người khác. Ví dụ, bạn không thể tạo một file với người sử dụng *kabid* rồi sau đó gán lại quyền sở hữu cho người khác có tên là *sheila* bởi lý do bảo mật. Nếu một người sử dụng thông thường có quyền gán quyền sở hữu file cho người khác, thì một ai đó cũng có thể tạo một chương trình xấu như xóa các file, và thay đổi quyền sở hữu cho superuser, và không biết điều gì sẽ xảy ra. Chỉ có người superuser mới có thể gán lại quyền sở hữu file hay thư mục cho người khác.

### 3.2.1. Thay đổi quyền sở hữu file, thư mục sử dụng lệnh `chown`

Chủ sở hữu superuser có thể thay đổi quyền sở hữu file, thư mục cho một người sử dụng khác. Để thay đổi quyền sở hữu sử dụng câu lệnh sau:

```
chown newuser file hoặc thư mục
```

Ví dụ:

```
chown trantu example.txt
```

Câu lệnh này làm cho người sử dụng trantu có quyền sở hữu file example.txt

Nếu superuser muốn thay đổi nhóm cho một file hoặc thư mục, người đó có thể sử dụng câu lệnh `chown` như sau:



`chown newuser.newgroup file hoặc thư mục`

Ví dụ

`chown trantu.admin example.txt`

Câu lệnh trên không chỉ thay đổi quyền sở hữu file cho trantu mà còn đặt lại nhóm sử dụng file là admin. Ắt ều superuser muốn thay đổi người sở hữu và nhóm sử dụng cho tất cả các file trong một thư mục, người đó có thể sử dụng câu lệnh `chown` với tùy chọn `-R`. Ví dụ

`chown -R trantu.admin /home/trantu/`

### 3.2.2. Thay đổi nhóm sử dụng file/thư mục với lệnh `chgrp`

Câu lệnh `chgrp` cho phép bạn thay đổi quyền sử dụng file hay thư mục của một nhóm, chỉ nếu bạn thuộc về cả hai nhóm (nhóm cũ và nhóm mới). Ví dụ:

`chgrp httpd *.html`

Lệnh trên sẽ thay đổi nhóm sử dụng cho tất cả các file có phần mở rộng `html`. Bạn chỉ có thể thay đổi được nếu bạn thuộc nhóm `httpd`. Giống như lệnh `chown`, lệnh `chgrp` cũng có tùy chọn `-R` để thay đổi quyền với nhiều file hay thư mục.

### 3.2.3. Sử dụng số theo hệ cơ số 8 tương ứng với thuộc tính truy cập

Hệ cơ số 8 sử dụng 8 số (0-7), và mỗi số tương ứng với 3 bit (theo hệ nhị phân). Bảng bên dưới chỉ cho bạn thấy sự tương ứng về quyền với số hệ cơ số 8.

		Số thứ 1	Số thứ 2	Số thứ 3	Số thứ 4
Giá trị cơ số 8	4	set-UID	R	r	r
	2	set-GID	W	w	w
	1	sticky-bit	X	x	x
		Special	User	Group	Others

Ắt ừ ở trên bảng trên, số thứ nhất được sử dụng cho việc thiết lập các quyền đặc biệt, số thứ hai được sử dụng cho việc thiết lập người sở hữu file hay thư mục. Số thứ ba được sử dụng để thiết lập quyền cho nhóm người sử dụng và số thứ tư được sử dụng để thiết lập quyền cho tất cả mọi người. Khi bất kỳ một số nào bị bỏ qua, nó được xem như nhận giá trị 0. Bảng bên dưới chỉ ra một vài ví dụ về các giá trị tương ứng với quyền:

Giá Trị	Giải Thích
0400	Chỉ có quyền đọc cho người sở hữu, nó tương ứng với 400.
0440	Chỉ có quyền đọc với người sở hữu và nhóm người sử dụng. Nó tương ứng với giá trị 440.
0444	Quyền đọc cho tất cả mọi người. Nó tương ứng với giá trị 444
0644	Chỉ người sở hữu có quyền đọc và ghi, tất cả mọi người có quyền đọc, tương ứng với giá trị 644. (6 là tạo bởi 4:r và 2:w)
0755	Đọc ghi và thực thi đối với người sử dụng, đọc và thực thi đối với tất cả mọi người. (7 là tạo bởi 4:r , 2:w và 1:x)
4755	Nó tương ứng với giá trị 755 ngoại trừ file này được đặt giá trị set-UID = 4. Điều này có nghĩa là khi file được thực thi, nó có tất cả các quyền của người sở hữu để thực hiện công việc. Sẽ là một lỗ hổng lớn nếu người sở hữu ấy là root và những người khác có quyền thực thi file này. Hãy cẩn thận khi thiết lập giá trị của set-UID.
2755	Nó tương tự với giá trị 755 ngoại trừ, khi thực thi nó có tất cả các quyền của nhóm sử dụng file.

Để thiết lập quyền phù hợp, bạn nên chỉ ra kiểu truy cập của người sử dụng, nhóm người sử dụng và của những người khác.

### 3.2.4. Sử dụng ngôn ngữ tự nhiên tương ứng với quyền truy cập

Bây giờ chúng ta sẽ sử dụng cấu trúc truy cập đơn giản hơn việc sử dụng số. Bảng bên dưới chỉ ra các cấu trúc truy cập tương ứng với các quyền:

read (r)	read (r)	read (r)	read (r)
write (w)	write (w)	write (w)	write (w)
execute (x)	execute (x)	execute (x)	execute (x)
<b>Special</b>	<b>User</b>	<b>Group</b>	<b>Others</b>

all (a)

Mỗi kiểu quyền tương ứng với một ký tự đơn (trong dấu ngoặc).

### 3.2.5. Thay đổi quyền truy cập file thư mục sử dụng lệnh chmod

Tiện ích chmod cho phép bạn thay đổi các quyền. Bạn có thể sử dụng các chữ số hay các ký tự với tiện ích này để thay đổi quyền. Ví dụ

```
chmod 755 *.pl
```

Câu lệnh trên thay đổi quyền cho các file có phần mở đuôi là .pl. Mỗi một file .pl được đặt các quyền đọc, ghi và thực thi bởi người sở hữu, các file cũng có thể đọc và thực thi bởi nhóm người sử dụng và những người khác. Bạn có thể hoàn thành cùng một công việc như vậy với lệnh sau:

```
chmod a+rx,u+w *.pl
```

a+rx được sử dụng để cho phép tất cả mọi người đọc và thực thi đối với mỗi file .pl và u+w được sử dụng để cho phép người sở hữu có quyền ghi đối với mỗi file .pl.

Nếu bạn muốn thay đổi các quyền cho tất cả các file và các thư mục con trong một thư mục, bạn có thể sử dụng tùy chọn -R:

```
chmod -R 750 /www/mysite
```

### 3.2.6. Các chú ý đặc biệt trên các quyền thư mục

Các quyền thiết lập cho một thư mục cũng tương tự như các file thông thường, nhưng không giống hệt nhau. Dưới đây là một vài chú ý đặc biệt trên các quyền thư mục:

- Quyền chỉ đọc cho một thư mục sẽ không cho phép bạn chuyển vào bên trong thư mục, để chuyển vào bên trong bạn cần có quyền thực thi
- Quyền chỉ được thực thi sẽ cho phép bạn truy cập vào các file bên trong một thư mục khi bạn biết tên của chúng và bạn được phép đọc chúng.
- Để có thể đưa ra danh sách nội dung của một thư mục sử dụng câu lệnh tương tự như ls và cũng có thể chuyển vào bên trong thư mục bạn cần có cả quyền đọc và quyền thực thi đối với thư mục đó
- Nếu bạn có quyền ghi cho một thư mục, bạn có thể tạo, thay đổi, xóa các file bất kỳ hay các thư mục con bất kỳ bên trong thư mục đó ngay cả khi file và thư mục con được sở hữu bởi người khác

## 3.3. Tạo một chính sách quyền cho một server nhiều người sử dụng

### 3.3.1. Thiết lập cấu hình các quyền truy cập file của người sử dụng

Trong thư mục của mỗi người sử dụng có một vài file ẩn chung bắt đầu với dấu chấm (.). Các file này thường được sử dụng để thực thi các câu lệnh tại thời điểm người sử dụng đăng nhập. Ví dụ, tất cả các shell (csh, tcsh, bash, ...) sẵn sàng cho một người sử dụng đọc các thiết lập của họ từ một file giống như .cshrc hay .bashrc. Nếu một người sử dụng không cẩn thận trong việc giữ quyền các file một cách hoàn hảo, một người sử dụng không thân thiện khác có thể gây ra các vấn đề không mong muốn. Ví dụ, nếu một file .cshrc của người sử dụng có thể được viết bởi người khác, người sử dụng có thể chơi một trò tấn công ngu ngốc như đưa một câu lệnh logout ngay dòng đầu của file .cshrc, như vậy người sử dụng sẽ thoát ngay khi đăng nhập vào hệ thống. Nếu

bạn có quyền thao tác với những người sử dụng bạn có thể thực hiện nhanh chóng việc kiểm tra đơn giản sau:

```
find /home -type f -name "*.rc" -exec ls -l {} \;
```

Câu lệnh này sẽ hiển thị quyền của tất cả các file có ký tự đầu tiên là dấu chấm, kết thúc bằng “rc” nằm trong thư mục home

### 3.3.2. Thiết lập mặc định các quyền truy cập file cho người sử dụng

Là người quản trị bạn cần định nghĩa các quyền mặc định thiết lập cho tất cả các file của người sử dụng đưa vào hệ thống của bạn. Để thiết lập mặc định quyền cho các file mới, bạn có thể sử dụng câu lệnh umask như sau:

```
umask mask
```

Để hiểu từ umask như thế nào, hãy xem ví dụ sau. Khi nói rằng umask đặt là 022, file mới được tạo, thông thường một quyền 0666 được yêu cầu bởi hàm tạo file – open. Tuy nhiên, trong trường hợp này, quyền cuối cùng thiết lập cho các file được tạo bởi hệ thống như sau: 0666 được thực hiện phép toán A&D với phần bù của 022 (phần bù của 022 là 755) do đó kết quả của phép A&D thu được là 0644, nó cho phép người sở hữu đọc và ghi còn những người khác chỉ có quyền đọc. Để tạo một mask mặc định cho các quyền truy cập file, bạn có thể nhúng câu lệnh umask vào một shell tài nguyên chung trong /etc để khi một người sử dụng đăng nhập và chạy một shell, file tài nguyên shell chung sẽ được thực thi. Ví dụ, nếu người sử dụng của bạn sử dụng shell /bin/csh hay /bin/tcsh, bạn có thể đưa một câu lệnh umask mong muốn trong file /etc/csh.cshrc cho mục đích này.

### 3.3.3. Thiết lập các quyền có thể thực thi cho các file

Các file chương trình có thể được chạy bởi những người sử dụng thông thường không bao giờ nên đặt quyền được ghi cho bất kỳ ai khác ngoài người sở hữu. Ví dụ, các file chương trình trong /usr/bin nên thiết lập các quyền như chỉ root có quyền đọc, ghi và thực thi và tất cả mọi người chỉ có quyền đọc và thực thi các file này. Việc cho phép người khác ghi có thể tạo ra một lỗ hổng nghiêm trọng cho hệ thống.

## 3.4. Làm việc với các file và các thư mục

### 3.4.1. Xem các file và các thư mục

Bạn có thể đã quen với lệnh ls, thông thường nó được sử dụng với các tùy chọn -l (long listing) hiển thị đầy đủ thông tin, -a hiển thị tất cả các file bao gồm cả các file bắt đầu bằng dấu chấm và -R hiển thị tất cả các file và các thư mục con bên trong thư mục mong muốn

### 3.4.2. Chuyển đến thư mục

Bạn gần như đã quen với câu lệnh cd, nó là một shell xây dựng sẵn. ầu bạn không cung cấp một tên thư mục bất kỳ làm đối số cho nó, nó sẽ chuyển về thư mục chủ của

bạn mà hiện tại bạn đang sử dụng. Khi bạn đang đứng ở bất kỳ đâu trong hệ thống file, bạn có thể sử dụng lệnh `pwd` để hiển thị đường dẫn đến thư mục hiện tại.

### 3.4.3. Xác định kiểu file

Không giống như hệ điều hành Windows, Linux không dựa vào phần mở rộng của file để xác định kiểu file. Bạn có thể sử dụng tiện ích file để xác định kiểu file trong hệ thống. Ví dụ:

```
file todo.txt
```

Kết quả hiển thị như sau:

```
todo.txt: ASCII text
```

### 3.4.4. Xem thông kê các quyền của file hay thư mục

Bạn có thể sử dụng lệnh `stat` để lấy thông kê về các file và các thư mục:

```
stat ./exam
```

Kết quả hiển thị trên màn hình

```
File: "./exam"
```

```
Size: 4096      Blocks: 8      IO Block: -4611692478058196992 Directory
```

```
Device: 812h/2066d  Inode: 157762  Links: 2
```

```
Access: (0755/drwxr-xr-x)  Uid: ( 0/  root)  Gid: ( 0/  root)
```

```
Access: Wed Jun 18 14:56:48 2003
```

```
Modify: Wed Jun 18 11:18:42 2003
```

```
Change: Wed Jun 18 11:18:42 2003
```

### 3.4.5. Sao chép file và thư mục

Sử dụng câu lệnh `cp` để sao chép từ một vị trí xác định đến vị trí khác:

```
cp /some/important /new/place
```

Bạn cũng có thể xác định một tên mới cho file sao chép. Thông thường lệnh `cp` được sử dụng với tùy chọn `-f` để sao chép file từ nguồn đến đích mà không quan tâm đến việc có một file cùng tên tồn tại ở đích. File mới sẽ được sao chép đè lên file cũ. Để sao chép một thư mục đến một thư mục khác bạn thực hiện lệnh `cp` với tùy chọn `-r` ví dụ:

```
cp -r /tmp/foo /zoo/foo
```

### 3.4.6. Dịch chuyển các file và thư mục

Để dịch chuyển các file hay thư mục sử dụng câu lệnh mv. Ví dụ, để chuyển /file1 vào /tmp/file2 ta sử dụng câu lệnh sau:

```
mv /file1 /tmp/file2
```

### 3.4.7. Xóa các file và thư mục

Để xóa các file và thư mục sử dụng lệnh sau:

```
rm filename
```

Khi xóa hệ thống sẽ hỏi bạn có thực sự muốn xóa hay không. Ắt ếu bạn đã chắc chắn file bạn muốn xóa bạn có thể thực hiện lệnh xóa rm với tùy chọn -f để không hiện ra thông tin yêu cầu xác nhận của hệ thống. Để xóa một thư mục, bạn cần thực hiện lệnh rm với tùy chọn -r

### 3.4.8. Tìm kiếm file

Để xác định vị trí chính xác của một file, bạn có thể sử dụng lệnh which. Ví dụ:

```
which httpd
```

Câu lệnh này sẽ chỉ ra cho bạn đầy đủ đường dẫn của chương trình httpd nếu nó sẵn có. Bạn cũng có thể xác định một phần của tên file hay thư mục sử dụng lệnh locate

```
locate netpr.pl
```

## 4. Quản lý người dùng và tài nguyên

### 4.1. Khái niệm

Linux là hệ điều hành đa nhiệm và đa người dùng. Mỗi người dùng có tên truy nhập và mật khẩu riêng, tương ứng với những quyền hạn nhất định trong hệ thống file của Linux.

Để tạo điều kiện thuận lợi trong quản lý người dùng và quyền hạn đối với hệ thống file, Linux cho phép khai báo những nhóm người dùng, mỗi nhóm là một tập hợp những người dùng chung một mục đích khai thác tài nguyên nhất định. Mỗi người dùng có thể tham gia nhiều nhóm người dùng khác nhau. Mỗi người dùng cũng mặc

nhân lập nên một nhóm người dùng là nhóm của chính họ (nhóm có thể chỉ có một thành viên).

Người dùng có toàn quyền trong Linux là người dùng root, mặc nhiên thuộc về nhóm root. Người dùng có quyền root ấn định một người dùng nào đó thuộc về nhóm root và có quyền tương đương với root.

## 4.2. Trở thành superuser

Bạn đã biết rằng tài khoản root là tài khoản superuser trong hệ thống Linux. Thực ra nếu bạn tự cài đặt hệ thống, bạn đã sử dụng tài khoản này để đăng nhập hệ thống lần đầu tiên. Bạn cũng biết rằng root là tài khoản superuser, tài khoản này có quyền làm mọi thứ trên hệ thống. Người sử dụng root có thể khởi động hay dừng một chương trình bất kỳ cũng như tạo và xóa một file bất kỳ. Rất nhiều những người mới quản trị hệ thống Linux cho rằng chỉ có root là tài khoản superuser. Hãy nhìn xuống đoạn mã bên dưới có trong file `/etc/passwd`

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:
daemon:x:2:2:daemon:/sbin:
vietvq:x:0:0:root:/home/vietvq:/bin/bash
xanhhh:x:0:0:root:/root:/bin/bash
tuta:x:0:0:root:/var:/bin/bash
```

Bạn có thể thấy được ở trên có 4 tài khoản superuser. Để hiểu tại sao bạn hãy xem định dạng một dòng trong file `/etc/passwd`

```
username:passwd:UID:GID:fullname:home-dir:shell
```

Bạn hãy chú ý vào các trường UID (User ID) và GID (Group ID) của tài khoản root. Tất cả những tài khoản mà có các giá trị của các trường này là 0 là những superuser. Hay nói một cách khác những người có UID = 0 và GID = 0 có quyền tương đương với tài khoản root.

Ảnh hưởng nếu hệ thống của bạn phải có nhiều tài khoản superuser do một số lý do quản trị, bạn có thể dễ dàng tạo một tài khoản superuser. Tuy nhiên, hãy nhớ rằng một tài khoản superuser (UID=0, GID=0) có thể làm mọi thứ.

## 4.3. Quản lý người dùng với các công cụ dòng lệnh

### 4.3.1. Tạo một tài khoản người sử dụng mới

Tạo một người sử dụng mới khá dễ dàng, để tạo người sử dụng từ dòng lệnh, bạn có thể sử dụng câu lệnh `useradd`. Ví dụ để tạo người sử dụng có tên là `tutavn`, bạn có thể chạy câu lệnh sau:

*useradd tutavn*

Trong file `/etc/passwd` sẽ bổ sung thêm dòng mới như sau:

```
tutavn:x:502:504::/home/tutavn:/bin/bash
```

Kí hiệu `x` có nghĩa là tài khoản chưa có mật khẩu. Vì vậy bạn cần tạo mật khẩu cho người sử dụng bằng câu lệnh sau:

*passwd tutavn*

Bạn sẽ được yêu cầu vào mật khẩu hai lần, và khi mật khẩu được tiếp nhận, nó sẽ được mã hóa và thêm vào dòng của người sử dụng trong file `/etc/passwd`. Các giá trị UID và GID sẽ được lựa chọn tự động bởi `useradd`, thông thường nó tăng giá trị UID và GID lên một số với người được thêm vào lần sau cùng trước đó. Bạn có thể tạo người sử dụng có thư mục chủ khác với mặc định (trong thư mục `home`) bằng thực hiện câu lệnh:

*useradd newuser -d /www/newuser*

Người sử dụng mới sẽ được tạo và có thư mục chủ là `/www/user`. Khi bạn tạo một người sử dụng mới, hệ thống cũng đồng thời mặc định tạo ra một nhóm mới có trong file `/etc/group` có tên giống như tên tài khoản của người sử dụng. Để tạo người sử dụng với tên nhóm mới hay tên nhóm đã tồn tại trong hệ thống, bạn sử dụng lệnh `adduser` với tùy chọn `-g`. Ví dụ:

*useradd tutavn -g users*

Nếu bạn muốn tạo người sử dụng mới là thành viên của một số nhóm, bạn có thể sử dụng tùy chọn `-G`. Ví dụ:

*useradd tutavn -G users1,users2*

### **4.3.2. Tạo một nhóm mới**

Để tạo một nhóm mới bạn sử dụng câu lệnh `groupadd`. Ví dụ:

*groupadd mygroup*



Nếu bạn tạo một tên nhóm đã có trong hệ thống bạn sẽ nhận được một thông báo lỗi

#### 4.3.3. Sửa đổi một tài khoản người sử dụng đang tồn tại

- *Thay đổi mật khẩu*

Để thay đổi mật khẩu của tài khoản đang tồn tại bạn sử dụng câu lệnh `passwd`. Ví dụ:  
*passwd tutavn*

Câu lệnh này tương đối đơn giản vì nó không có các tùy chọn, và nó chỉ cho phép người sử dụng thông thường chỉ có thể thay đổi mật khẩu của chính họ. Hệ thống sẽ yêu cầu bạn nhập mật khẩu hai lần và khi mật khẩu được tiếp nhận, nó sẽ được mã hóa trước khi đưa vào file `/etc/passwd`

#### 4.3.4. Thay đổi đường dẫn thư mục chủ

Để thay đổi đường dẫn thư mục chủ của người sử dụng đang tồn tại, sử dụng câu lệnh `usermod` như sau:

```
usermod -d new_home_directory username
```

Ví dụ, nếu một người sử dụng `tutavn` có thư mục chủ `/home/tutavn` và muốn chuyển thành `/home2/tutavn`, bạn có thể chạy câu lệnh sau:

```
usermod -d /home2/tutavn tutavn
```

Tuy nhiên, nếu bạn muốn nội dung thư mục chủ đến một vị trí mới, sử dụng tùy chọn `-m` như sau:

```
usermod -d -m /home2/tutavn tutavn
```

#### 4.3.5. Thay đổi UID

Để thay đổi UID của một người sử dụng, sử dụng câu lệnh `usermod` như sau:

```
usermod -u UID username
```

Ví dụ:

```
usermod -u 500 myfrog
```

Câu lệnh này sẽ thay đổi UID của người sử dụng `myfrog` là 500

#### 4.3.6. Thay đổi nhóm mặc định

Để thay đổi nhóm mặc định cho người sử dụng, sử dụng câu lệnh `usermod` với tùy chọn `-g`

```
usermod -g 777 myfrog
```

Câu lệnh này sẽ thay đổi nhóm mặc định của `myfrog` thành `777`.

#### 4.3.7. Thay đổi thời hạn kết thúc của một tài khoản

Bạn có thể thay đổi thời hạn kết thúc của một tài khoản sử dụng câu lệnh `usermod` với tùy chọn `-e`. Cú pháp của câu lệnh như sau:

```
usermod -e MM/DD/YY username
```

Ví dụ:

```
usermod -e 12/31/99 kabir
```

#### 4.3.8. Sửa đổi một nhóm đang tồn tại

Để sửa đổi tên một nhóm đang tồn tại, sử dụng câu lệnh `groupmod`. Cú pháp như sau:

```
groupmod -n new_group current_group
```

Ví dụ:

```
groupmod -n experts novices
```

Ả hóm `novices` đang tồn tại được đổi tên thành `experts`. Để thay đổi GID của một nhóm sử dụng tùy chọn `-g` như sau:

```
groupmod -g 666 troublemaker
```

Câu lệnh này sẽ thay đổi GID của một nhóm `troublemaker` thành `666`.

#### 4.3.9 Xóa hoặc hủy bỏ một tài khoản người sử dụng

Để xóa một tài khoản đang tồn tại sử dụng câu lệnh `userdel`. Ví dụ:

```
userdel snake
```

Sẽ xóa bỏ tài khoản tài khoản `snake` khỏi hệ thống. Ắt ều bạn muốn xóa thư mục chủ của người sử dụng và tất cả các nội dung trong thư mục, sử dụng tùy chọn `-r`. Chú ý rằng `userdel` sẽ không xóa người sử dụng nếu người sử dụng hiện tại đang đăng nhập.

Ắt ều bạn muốn hủy bỏ tạm thời quyền truy cập của tất cả các tài khoản bạn có thể tạo một file tạm thời có tên là `/etc/nologin` với một thông tin giải thích lý do vì sao không được phép truy cập. Chương trình login sẽ không cho phép bất kỳ tài khoản nào khác tài khoản `root` có thể đăng nhập trong thời gian này.

## 4.4. Cài đặt máy in

### 4.4.1. Cấu hình máy in

Ứng dụng **printconf** cho phép người dùng cấu hình máy in trong Red Hat Linux. Ắt ó cho phép sửa đổi tệp tin cấu hình `/etc/printcap`, các thư mục bộ đệm in và bộ lọc in. **printconf** cấu hình hệ thống in ấn của bạn, được gọi là LPRng. LPRng cũng là một hệ thống in ấn ngầm định. Phần này tập trung vào việc sử dụng **printconf** để cấu hình LPRng.

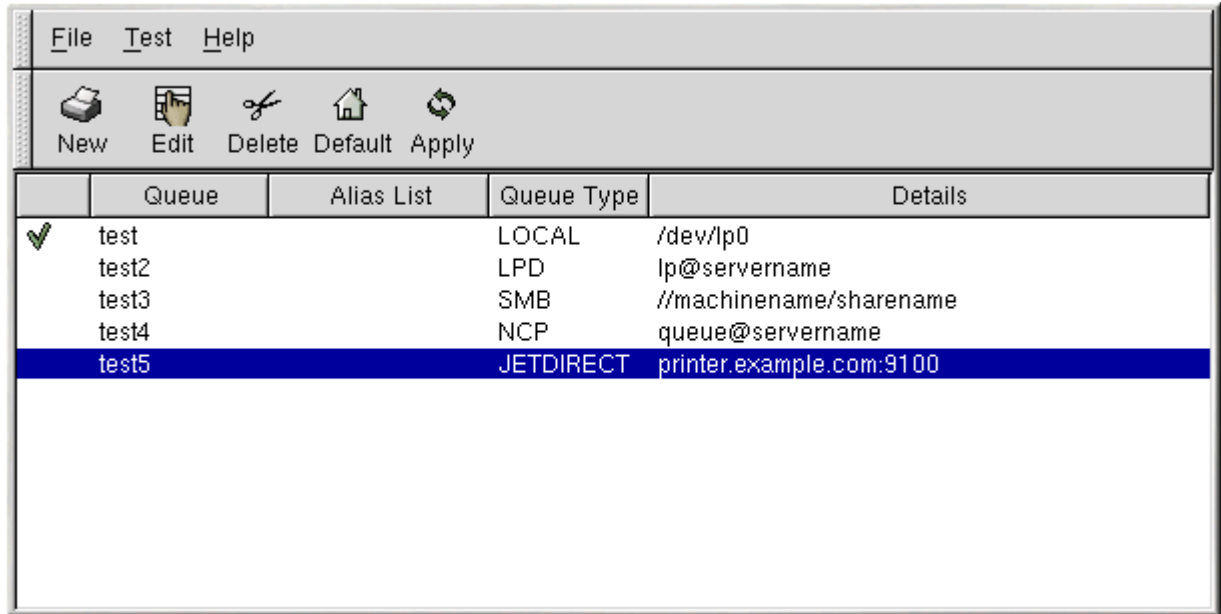
Để sử dụng **printconf**, bạn phải có quyền truy cập mức `root`. Để khởi động **printconf**, theo một trong các cách sau đây

- Trên màn hình Gã OME, chọn **Main Menu Button** => **Programs** => **System** => **Printer Configuration** để khởi động trong chế độ đồ họa.
- Trên màn hình KDE, chọn **Main Menu Button** => **System** => **Printer Configuration** để khởi động chế độ đồ họa.
- Đánh lệnh `printtool` tại dấu nhắc shell (VD: XTerm hoặc Gã OME terminal) để khởi động **printconf**

Bạn cũng có thể chạy **printconf** dưới dạng một ứng dụng trong chế độ text nếu bạn không cài đặt hệ thống X Window hoặc bạn thích sử dụng giao diện text hơn. Khi đó, bạn phải log in theo tài khoản `root` (hoặc dùng lệnh `su` để chuyển sang người dùng `root` và đánh lệnh `/usr/sbin/printconf-tui` tại dấu nhắc shell).

Chú ý: bạn đừng sửa đổi tệp tin `/etc/printcap`, mỗi khi daemon máy in (`lpd`) được khởi động hay khởi động lại, tệp tin `/etc/printcap` mới sẽ được sinh ra tự động.

Ắt ều bạn muốn cài đặt máy in mà không sử dụng **printconf**, khi đó bạn phải chỉnh sửa tệp tin `etc/printcap.local`. Các đầu vào trong `/etc/printcap.local` không được hiển thị trong **printconf** nhưng được daemon máy in đọc khi khởi động dịch vụ in ấn. Mỗi khi bạn nâng cấp hệ thống của bạn lên phiên bản mới, tệp cấu hình sẽ được **printconf** chuyển sang định dạng mới và tệp tin cấu hình cũ sẽ được ghi dưới tên `/etc/printcap.old`.



Hình 1: Cửa sổ *printconf* chính

Có năm kiểu hàng đợi in được cấu hình bởi **printconf**:

- **Local Printer** — máy in được gắn trực tiếp vào máy tính của bạn thông qua cổng song song hoặc cổng USB. Kiểu hàng đợi in **Queue Type** sẽ được thiết lập là **LOCAL**.
- **Unix Printer (lpd Spool)** — máy in được gắn trên một hệ thống Uả IX khác mà có thể được truy nhập thông qua mạng TCP/IP. Kiểu hàng đợi in **Queue Type** cho máy Uả IX ở xa sẽ được thiết lập là **LPD**.
- **Windows Printer (SMB)** — máy in được gắn trên một hệ thống khác (Windows) có chia sẻ máy in thông qua mạng SMB (sử dụng dịch vụ samba để chia sẻ tài nguyên trên mạng: máy in, dữ liệu.....), kiểu hàng đợi in **Queue Type** lúc đó sẽ được thiết lập là **SMB**.
- **Novell Printer (NCP Queue)** — máy in được gắn vào một hệ thống sử dụng công nghệ mạng ỏ ovell's ỏ etWare. Kiểu hàng đợi in cho máy in ỏ ovel ở xa sẽ được thiết lập là **NCP**.
- **JetDirect Printer** — máy in được nối trực tiếp vào mạng (máy in mạng). Kiểu hàng đợi in **Queue Type** cho máy in JetDirect sẽ được thiết lập là **JETDIRECT**.

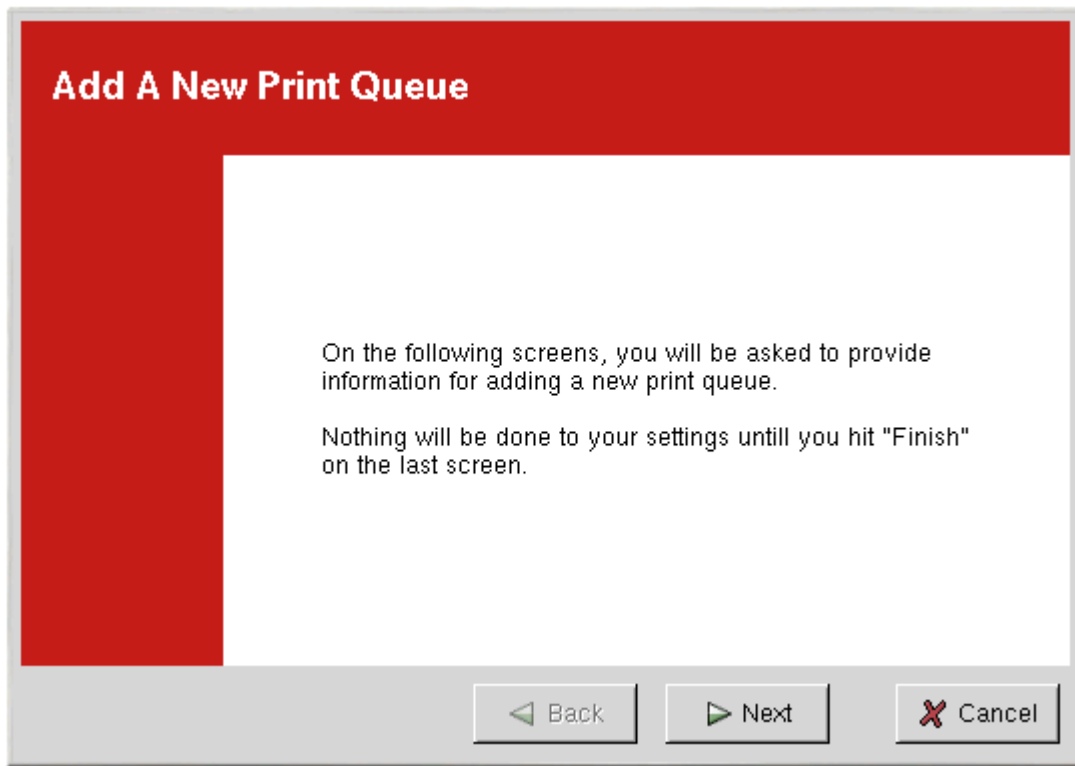
Chú ý: Khi bạn thêm một hàng đợi in mới hay sửa đổi hàng đợi in cũ, bạn phải khởi động lại daemon máy in (lpd) để những thay đổi đó có hiệu lực.

Chọn **Apply** ghi lại những thay đổi mà bạn vừa thực hiện và khởi động lại daemon máy in. Các thay đổi sẽ chưa được ghi trong tệp tin cấu hình `/etc/printcap` cho đến khi daemon máy in (lpd) được khởi động lại. Để thực hiện công việc này, chọn **File => Save Changes** và sau đó chọn **File => Restart lpd**.

ỏ ếu một máy in xuất hiện trong danh sách in với **Queue Type** được thiết đặt là **INVALID**, cấu hình máy in có thể thiếu các tùy chọn cần có cho máy in hoạt động. Chọn **Delete** để xóa máy in khỏi danh sách.

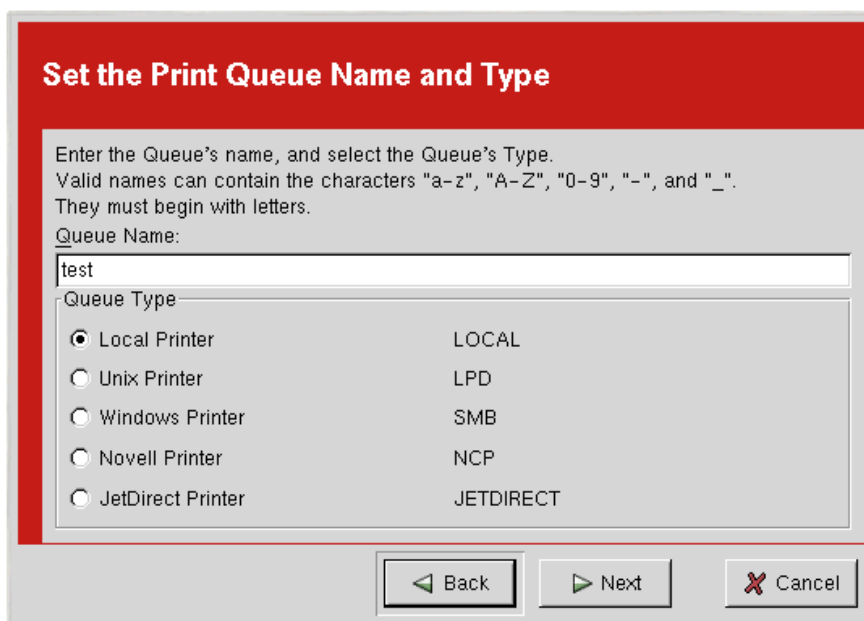
#### 4.4.2. Cài đặt máy in cục bộ

Để cài đặt một máy in gắn trên cổng song song hay cổng USB của máy tính, nhấn nút **Next** trên cửa sổ `printconf` chính như trên, chọn **Next** để tiếp tục.



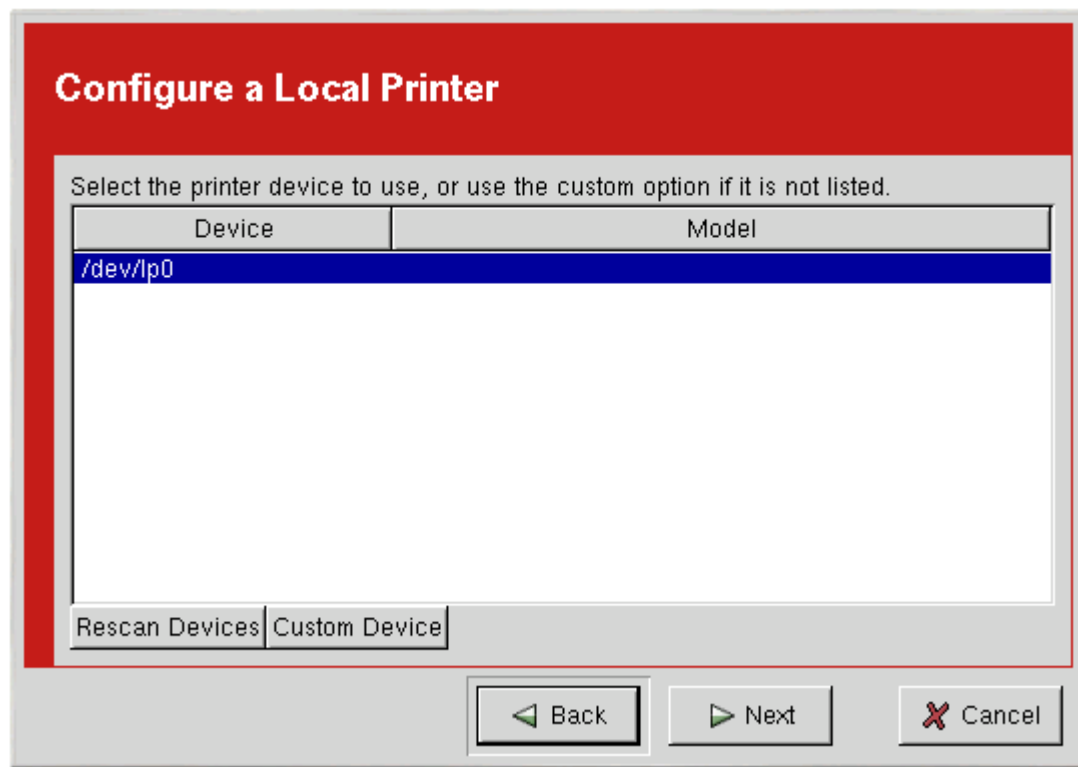
Hình 2: Cài đặt máy in

nhập tên máy in trong trường **Queue Name** và chọn **Local Printer** từ danh sách **Queue Type** nhấn **Next** để tiếp tục.



Hình 3: Cài đặt máy in cục bộ

**printconf** sẽ có gắng phát hiện máy in và hiển thị như trong hình 4.

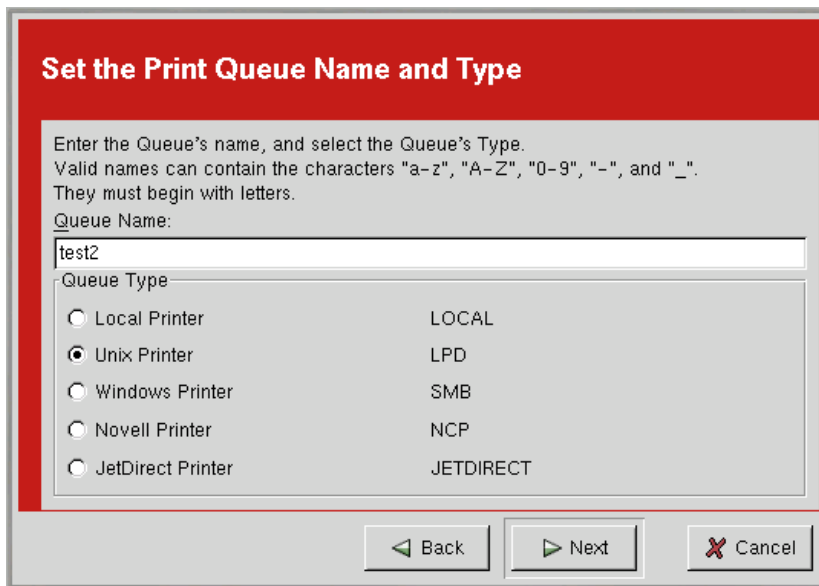


Hình 4: Chọn thiết bị máy in

#### 4.4.3. Cài đặt máy in trên hệ thống Unix ở xa

Để cài đặt một máy in gắn trên một hệ thống Linux ở xa trong cùng một mạng, nhấn nút **New** trong cửa sổ chính **printconf**. Một cửa sổ như hình 2 sẽ xuất hiện, chọn **Next** để tiếp tục.

Cửa sổ như hình 3 xuất hiện. Bạn cũng phải nhập tên máy in vào trường **Queue Name** và chọn **Unix Printer** từ trong thực đơn **Queue Type**, nhấn **Next** để tiếp tục.



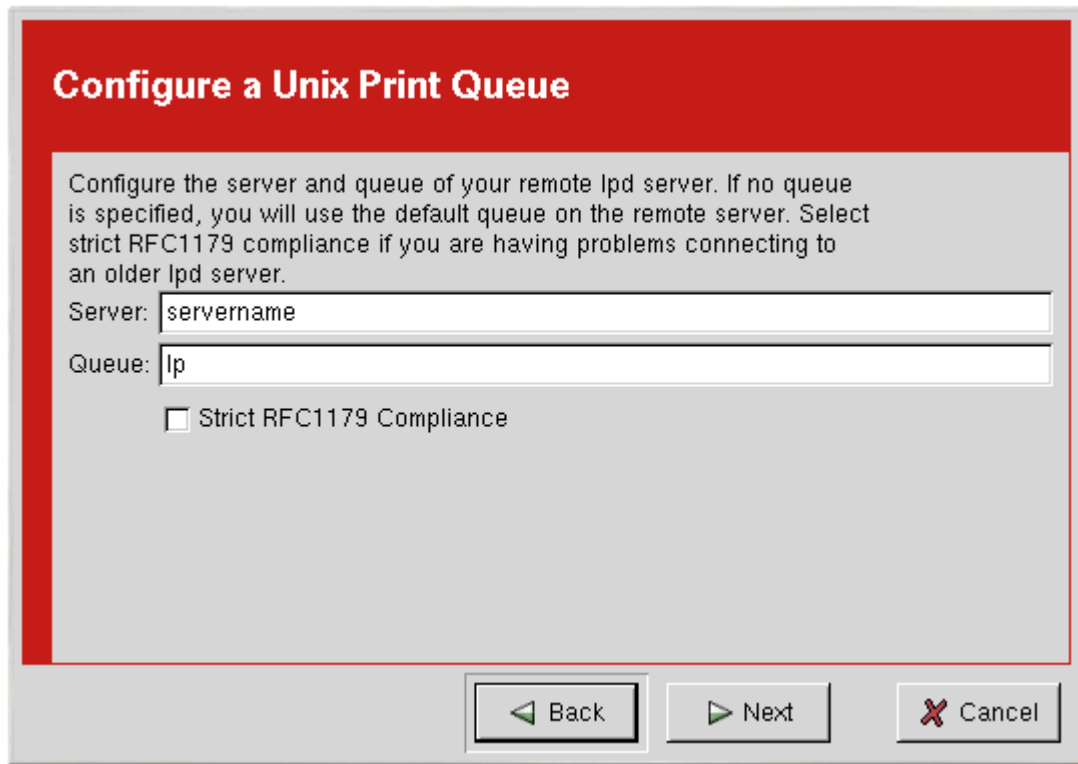
Hình 5: Cài đặt máy in Unix ở xa

Cửa sổ tiếp theo cho phép bạn cấu hình máy chủ in ở xa đó.

- **Server** — Hstname hoặc địa chỉ IP của máy ở xa mà máy in gắn vào.
- **Queue** — Hàng đợi máy in ở xa, ngầm định là **lp**.

Â gầm định không chọn tùy chọn **Strict RFC1179 Compliance**. Chỉ khi nào bạn gặp vấn đề về in ấn với một hàng đợi với một hàng đợi lpd không phải Linux, hãy chọn tùy chọn này để cấm các tính năng in ấn LPRng nâng cao.

Ấ hấn **Next** để tiếp tục.



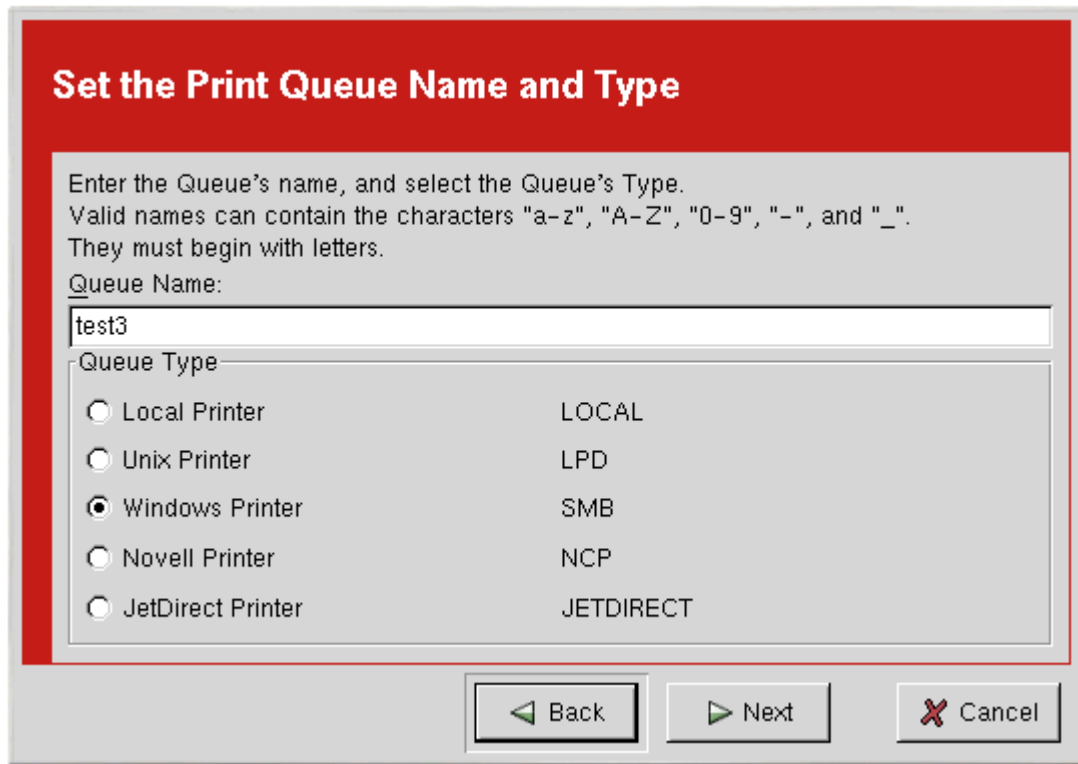
Hình 6: Chọn Printer Server

Bước tiếp theo là chọn kiểu máy in kết nối với hệ thống ở xa đó. Chú ý rằng máy ở xa phải được cấu hình để cho phép một máy cục bộ có thể đưa yêu cầu và in ấn. Để thực hiện điều đó, bạn phải tạo một file `/etc/hosts.lpd` trên máy ở xa mà máy in gắn kèm và thêm vào các địa chỉ IP hay hostname của các máy muốn in trên các dòng riêng rẽ trong tệp tin.

#### 4.4.4. Cài đặt máy in Samba (SMB)

Các bước thực hiện ban đầu tương tự hai bước ở trên. Trong thực đơn Queue Type, chọn Windows Printer và nhấn `Next` để tiếp tục.



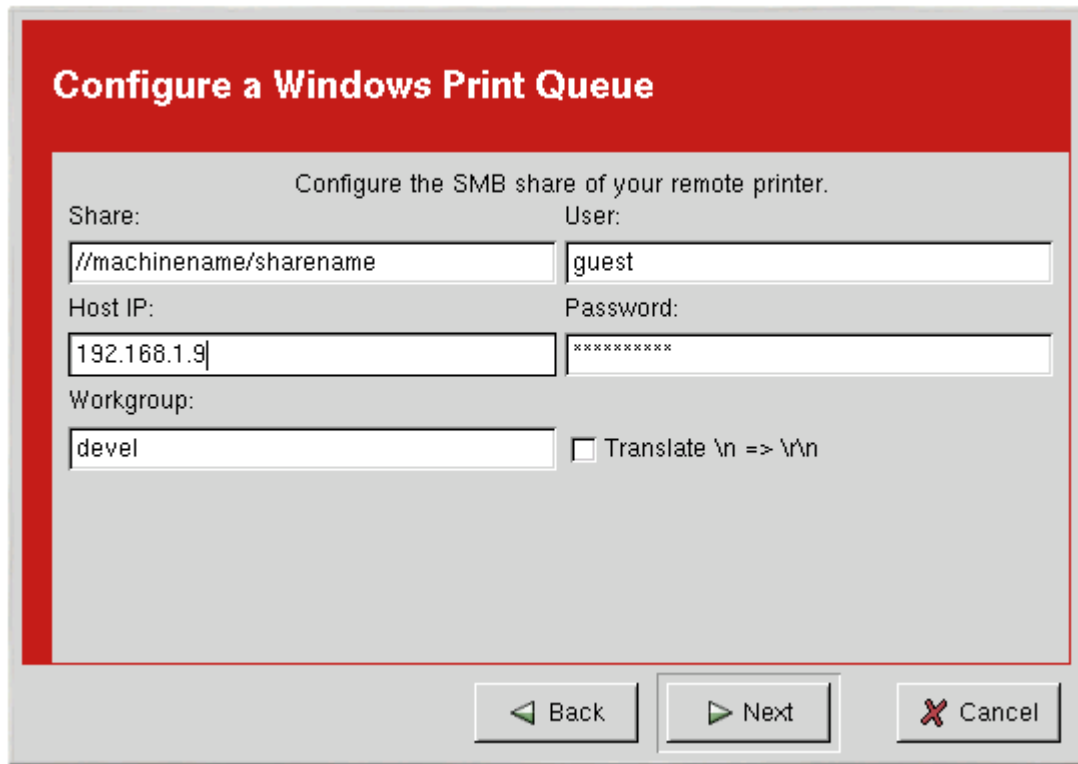


Hình 7: Cài đặt máy in SMB

Trong cửa sổ của hình 8, điền các thông số cấu hình sau:

- **Share** — Tên của máy in được chia sẻ mà bạn muốn in tại đó. Tên này phải cùng tên với tên được định nghĩa cho máy in Samba trên máy Windows ở xa. Chú ý cú pháp phải như sau: `//machinename/sharename`.
- **User** — Tên người dùng được phép truy nhập vào máy in. Tên này phải tồn tại trên hệ thống Windows và người dùng có quyền truy nhập máy in. Tên thường là **guest** đối với các máy Windows servers, hoặc **nobody** đối với các máy Samba servers.
- **Host IP** — Hostname hay địa chỉ IP của hệ thống ở xa chia sẻ máy in SMB.
- **Password** — Mật khẩu (nếu có) của người dùng định nghĩa trong trường **User**
- **Workgroup** — Tên workgroup máy chạy Samba thuộc vào.

Chọn nút **Translate \n => \r\n** để chuyển đổi các ký tự cuối dòng sang khuôn dạng mà hệ thống Microsoft Windows có thể đọc được.  
Ấn nút **Next** để tiếp tục.



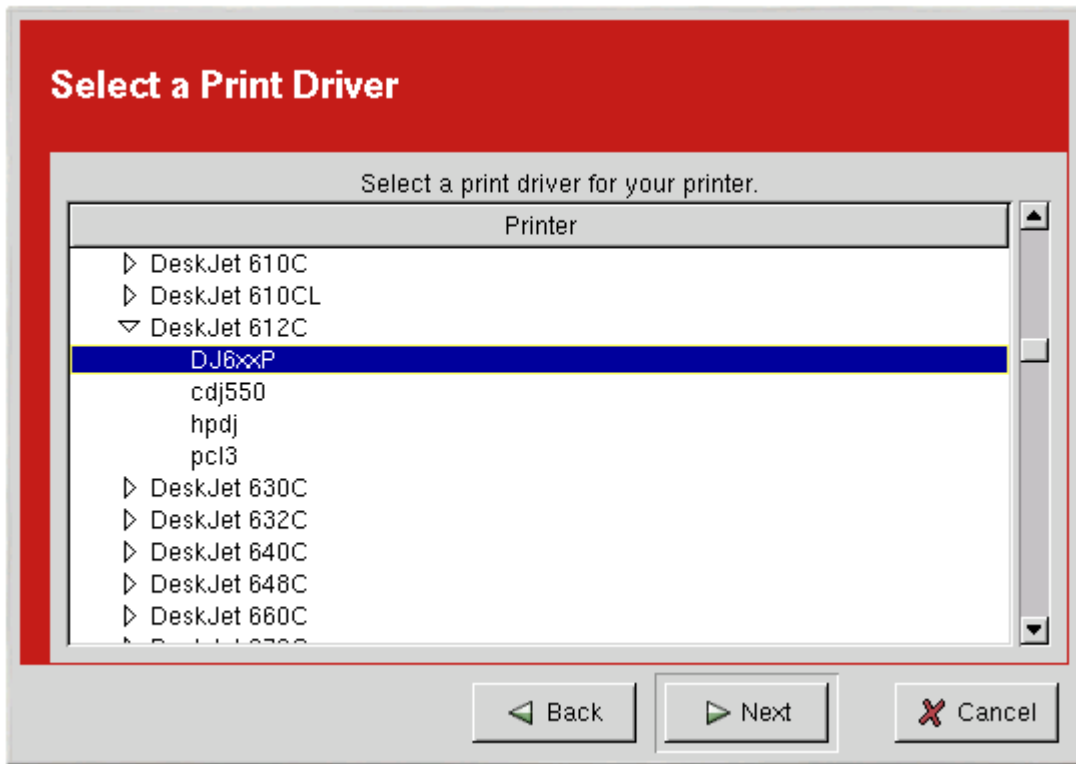
Hình 8: Chọn Print Server

Bước tiếp theo là chọn kiểu máy in được kết nối với hệ thống SMB ở xa.

#### 4.4.5. Chọn trình điều khiển Print Driver và kết thúc

Sau khi đã chọn kiểu hàng đợi máy in và cài đặt các thông số liên quan, bước tiếp theo là chọn trình điều khiển máy in.

Bạn sẽ thấy một cửa sổ như hình 13. Ở đây bạn cấu hình một máy in cục bộ, hãy chọn trình điều khiển in từ trong danh sách, chọn nhà sản xuất và loại máy in của bạn.

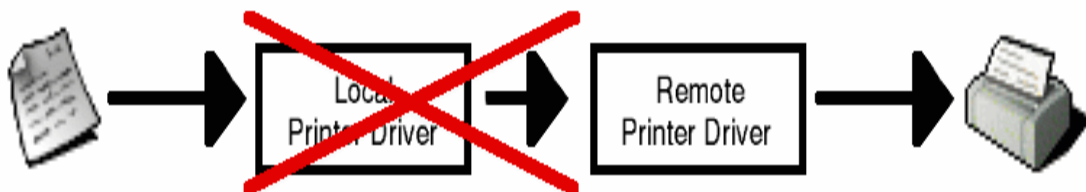


Hình 13: Chọn trình điều khiển máy in

Máy in cục bộ:




Nếu bạn cấu hình máy in ở xa (LPD, SMB, hay ở CP), máy in chủ ở xa sẽ in ấn theo trình điều khiển máy in của nó. Cố gắng chọn đúng trình điều khiển máy in ở xa đó.



Bước cuối cùng là khẳng định lại các thông số cấu hình, nhấn nút **Apply** để ghi lại các thay đổi và trong tệp tin cấu hình etc/printcap và khởi động lại daemon máy in (lpd). Hãy in thử 1 trang xem cấu hình bạn thiết lập đã đúng chưa.

#### 4.4.6. Thay đổi thông số cấu hình các máy in có sẵn

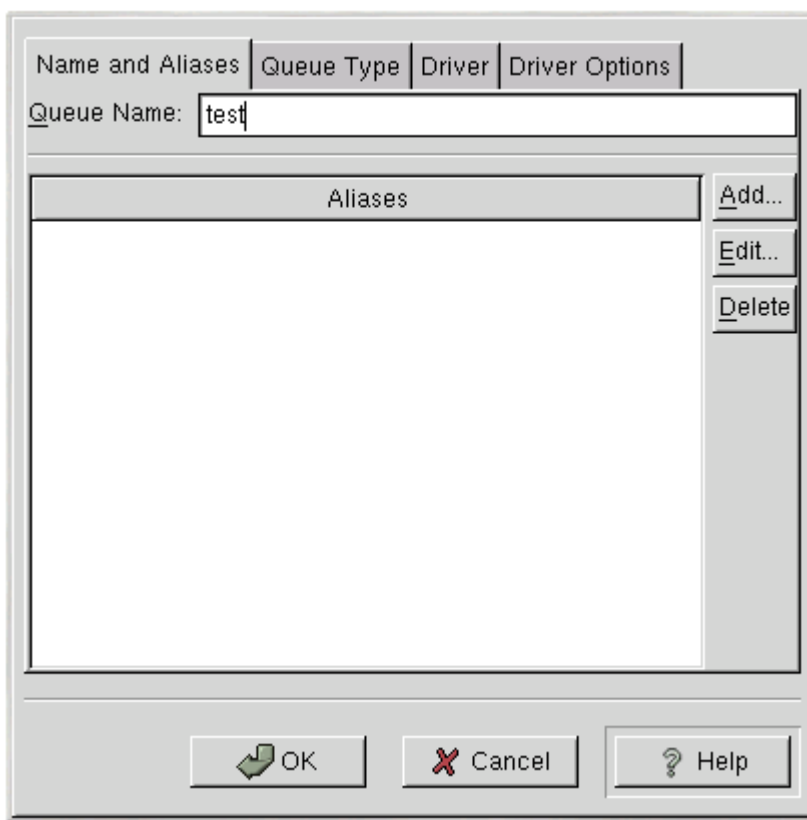
Để xóa một máy in đang tồn tại, chọn máy in và nhấn nút **Delete** trên thanh công cụ, máy in sẽ được loại bỏ trong danh sách máy in. Ấn nút **Apply** để ghi lại các thay đổi và khởi động lại daemon

Để thiết lập một máy in ngầm định, chọn máy in từ danh sách và nhấn nút **Default** trên thanh công cụ. Máy in ngầm định sẽ có icon  xuất hiện bên cạnh tên máy in.

Nếu bạn muốn thay đổi cấu hình của một máy in, bạn không thể thay đổi các thiết đặt đó một cách trực tiếp mà chỉ được ghi đề lên như sau:

Chọn máy in, chọn **File** => **Override Queue** từ thực đơn. Khi đó, máy in sẽ có kí hiệu  ở cạnh tên máy in.

Chọn nút **Edit** để thực hiện việc hiệu chỉnh các thông số. Cửa sổ như hình 14 xuất hiện cho phép bạn thay đổi lại các thông số của máy in.



Hình 14: Thay đổi thông số máy in

#### 4.4.7. Backup các thông số cấu hình máy in

Thông số cấu hình của bạn được đưa vào tệp tin `/etc/printcap` và được daemon máy in đọc khi khởi động. Bạn có thể sử dụng các lệnh để backup lại các file cấu hình ví dụ như backup file cấu hình máy in và ghi thành file `settings.xml`

```
/usr/sbin/printconf-tui --Xexport > settings.xml
```

Để khôi phục lại file cấu hình đã được backup theo cách trên, bạn có thể sử dụng lệnh dưới đây

```
/usr/sbin/printconf-tui --Ximport < settings.xml
```

#### 4.4.8. Quản lý công việc in ấn

Khi bạn muốn in một file văn bản từ **Emacs** hoặc in một hình ảnh từ **The GIMP**, công việc này sẽ được đưa vào hàng đệm in. Nếu muốn xem danh sách các công việc in ấn, đưa lệnh `lpq` vào dấu nhắc shell, ví dụ:

```
Rank   Owner/ID                Class Job Files      Size Time
active user@localhost+902      A    902 sample.txt  2050
01:20:46
```

Để muốn dừng một công việc in nào đó, đưa lệnh `lprm job number` với tham số là định danh của công việc in mà bạn biết được thông qua lệnh `lpq` ở trên. Bạn cũng có thể in ấn thông qua lệnh `lpr sample.txt` để in file văn bản `sample.txt`.

## 5. Trình diễn thiết lập mạng và cài đặt diu-up trên Linux

### 5.1. Thiết lập mạng Linux

Chúng ta sẽ xem xét quá trình nối một máy Linux vào mạng Ethernet để trao đổi thông tin bằng giao thức TCP/IP trên Ethernet.

#### 5.1.1. HDH Linux và card mạng

Để nối một máy Linux vào một mạng Ethernet, bạn cần phải có đầu tiên là một card mạng mà Linux đã có chương trình driver. Sau đây là một số mạng mà Linux có trợ giúp (danh sách sau không đầy đủ và các phiên bản mới của Linux hỗ trợ rất nhiều các card mạng khác nhau) :

- 3Com 3C509
- 3Com 3C503/16
- Novell E1000
- Novell E2000
- Western Digital WD8003
- Western Digital WD8013
- Hewlett-Packard HP27245
- Hewlett-Packard HP27247
- Hewlett-Packard HP27250

Giả sử các bạn muốn gắn máy của mình vào một mạng LAN Ethernet và bạn đã có một card mạng. Vấn đề đầu tiên là sự nhận biết của Linux đối với card này. Nếu card của bạn là một card khá phổ biến như 3c509 của 3COM hay E2000 của Novell, HDH Linux sẽ nhận biết sự hiện diện của card trong quá trình boot. Để biết xem kết

qua nhận biết card mạng, ta có thể xem xét các thông báo của kernel Linux trong quá trình boot của hệ thống qua lệnh **dmesg**

```
Freeing unused kernel memory: 60k freed
Adding Swap: 72572k swap-space (priority -1)
eth0: 3c509 at 0x300 tag 1, BNC port, address 00 a0 24 4f 3d dc, IRQ
10.
3c509.c:1.16 (2.2) 2/3/98 becker@cesdis.gsfc.nasa.gov.
eth0: Setting Rx mode to 1 addresses.
```

Hai dòng in đậm báo rằng card mạng 3c509 đã được kernel nhận biết. Trong trường hợp kernel không nhận biết card ☹, chúng ta phải làm lại kernel Linux và đặt module điều khiển (driver) của card vào trong kernel hay cấu hình ở chế độ load module.

Để cấu hình tiếp nối mạng qua TCP/IP chúng ta phải xác định rõ các thông tin liên quan đến địa chỉ IP của máy. Các thông tin cần biết là :

Địa chỉ IP của máy

ở etmask

Địa chỉ của mạng

Broadcast

Địa chỉ IP của gateway

Chúng ta sẽ lần lượt đi qua các khái niệm cơ bản trên và sẽ học sâu hơn trong phần TCP/IP của khóa học.

**Địa chỉ IP của máy** là một dãy 4 số viết dưới dạng A.B.C.D, trong đó mỗi số nhận giá trị từ 0-255. ở đầu máy của bạn kết nối một mạng nhỏ tại nhà do bạn thiết lập thì địa chỉ kiểu 192.168.1.D là một địa chỉ nên đặt, với D là các số khác nhau cho từng máy. ở đầu máy của bạn sẽ hòa nhập với một mạng LAN đã có trước đó và bạn muốn kết nối với các máy khác thì hỏi người quản trị mạng về địa chỉ IP bạn có thể gán cho máy của mình cùng với tất cả các thông số tiếp theo.

**Netmask.** Tương tự như trên, nếu bạn tự quản, netmask sẽ là 255.255.255.0

**Địa chỉ mạng.** Nếu bạn tự quản, địa chỉ của mạng sẽ là 192.168.1.0

**Broadcast.** Nếu bạn tự quản, broadcast là 192.168.1.255

**Địa chỉ gateway.** Đây là địa chỉ của máy cho phép bạn kết nối với mạng LAN khác, tức là các máy tính với 3 số đầu của địa chỉ không giống bạn là 192.168.1. Bạn bỏ trống nếu bạn chỉ liên lạc với các máy cùng mạng 192.168.1.XXX. Chú ý là địa chỉ mạng của máy gateway bắt buộc phải trùng với địa chỉ mạng của bạn.

Sau khi đã xác định các thông số, ví dụ như

IP address = 192.168.1.15

ở etmask = 255.255.255.0

suy ra network address = 192.168.1.0 và broadcast = 192.168.1.255

Gateway = 192.168.1.1

### 5.1.2. Cấu hình card mạng

#### Lệnh **ifconfig**

Sau khi làm cho kernel nhận biết sự hiện diện của card mạng, công tác tiếp theo là cấu hình TCP/IP cho card. Trong quá trình cài đặt Linux Redhat 6.X, bình thường chúng ta đã được chương trình cài đặt hỏi và cấu hình hộ. Trong trường hợp khi chúng ta bổ sung card mạng sau khi Linux đã được cài đặt, chúng ta có thể sử dụng tiện ích **netconf** cho mục đích này hoặc chúng ta sử dụng lệnh **ifconfig** để tự cài đặt.

Lệnh **ifconfig** được sử dụng trong quá trình boot hệ thống để cấu hình các trang thiết bị mạng. Sau đó, trong quá trình vận hành, **ifconfig** được sử dụng cho debug, hoặc để cho người quản trị hệ thống thay đổi cấu hình khi cần thiết.

Lệnh **ifconfig** không có tùy chọn dùng để hiển thị cấu hình hiện tại của máy.

```
[root@pasteur tnminh]# /sbin/ifconfig
eth0      Link encap:Ethernet  HWaddr 00:A0:24:4F:3D:DC
          inet      addr:192.168.2.20          Bcast:192.168.2.255
Mask:255.255.255.0
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:531 errors:4 dropped:0 overruns:0 frame:4
          TX packets:1854 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          Interrupt:10 Base address:0x300

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:3924  Metric:1
          RX packets:1179 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1179 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
```

Để gán địa chỉ IP 193.105.106.10 cho card mạng Ethernet đầu tiên ta dùng lệnh

```
ifconfig eth0 193.105.106.10 netmask 255.255.255.0 broadcast
192.105.106.255
```

Linux cho phép bạn sử dụng bí danh (alias) cho card mạng, tức là cho phép bạn có nhiều địa chỉ IP cho cùng một card vật lý. Kết quả nhận được gần giống như bạn có gắn nhiều card vật lý lên máy. Do đó, bạn có thể dùng một card để nối với nhiều mạng logic khác nhau. Cú pháp của lệnh này là :

```
ifconfig eth0:0 208.148.45.58 netmask 255.255.255.248 broadcast
208.148.45.255 up
```

Các tập tin cấu hình của kết nối mạng là `/etc/sysconfig/network-scripts/ifcfg-ethX` với X là 0,1 ... hay 0:0, 0:1 .... Bạn có thể thay đổi cấu hình kết nối mạng bằng cách sửa đổi lại tập tin này bằng một chương trình soạn thảo text như **mc** chẳng hạn, sau đó khởi động lại kết nối mạng bằng

```
/etc/rc.d/init.d/network restart
```

Để kiểm tra lại kết quả qua lệnh **ifconfig**.

### Lệnh route

Lệnh **route** cho phép làm các thao tác đến bảng dẫn đường (forwarding table) của kernel. Nó được sử dụng đầu tiên để xác định đường dẫn cố định (static) đến những máy hoặc những mạng qua các card mạng ethernet đã được cấu hình trước đó bởi `ifconfig`. Lệnh **route** không có tùy chọn (option) cho phép hiển thị bảng dẫn đường hiện tại của kernel (Lệnh **netstat -r** cũng có tác dụng tương tự)

```
[root@pasteur tnminh]# /sbin/route
```

Kernel IP routing table

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
192.168.2.20	*	255.255.255.255	UH	0	0	0	eth0
192.168.2.0	*	255.255.255.0	U	0	0	0	eth0
127.0.0.0	*	255.0.0.0	U	0	0	0	lo
default	192.168.2.10	0.0.0.0	UG	0	0	0	eth0

Để chỉ ra rằng card mạng eth0 được nối với một mạng 208.148.45.56 ta dùng lệnh `route` như sau :

```
route add -net 208.148.45.56 eth0
```

Còn nếu chúng ta muốn sử dụng bí danh của card mạng để nối vào một mạng logic khác, ta có thể sử dụng lệnh

```
route add -net 193.105.106.0 eth0:0
```

Công tác cuối cùng là phải chỉ ra các địa chỉ của gateway mặc định.

```
route add default gw 193.105.106.1 metric 1
```



Biết sử dụng thành thạo cú pháp của 2 lệnh **ifconfig** và **route** rất quan trọng, nó cho phép các cán bộ quản trị thay đổi cấu hình kết nối mạng của một server một cách nhanh chóng và không phải khởi động lại máy. Vì vậy, server luôn sẵn sàng. Bạn cũng có thể sử dụng tiện ích **netconfig** để cấu hình liên kết mạng nếu chưa thành thạo nhiều cú pháp của các lệnh trên.

## Lệnh ping

Ứng dụng của lệnh này là để thử xem 2 máy có kết nối được với nhau chưa. Cú pháp cơ bản của lệnh rất đơn giản là *ping địa\_chi\_IP\_máy\_đích*. Ví dụ như

```
[tnminh@proxy tnminh]$ ping sun
```

```
PING sun.vnuhcm.edu.vn (172.16.1.4): 56 data bytes
64 bytes from 172.16.1.4: icmp_seq=0 ttl=255 time=0.1 ms
64 bytes from 172.16.1.4: icmp_seq=1 ttl=255 time=0.2 ms
64 bytes from 172.16.1.4: icmp_seq=2 ttl=255 time=0.1 ms
64 bytes from 172.16.1.4: icmp_seq=3 ttl=255 time=0.1 ms

--- sun.vnuhcm.edu.vn ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 0.1/0.1/0.2 ms
```

ả ều 2 máy có thể liên lạc được với nhau, chúng ta sẽ biết thêm thời gian trả lời để cho biết sự thông thoáng về mạng giữa 2 máy. Có thể nói, **ping** phải chạy trước tiên trước tất cả các hoạt động mạng khác.

Chú ý: ả ền sử dụng ping -n để tránh trục trặc do dịch vụ Dã S làm ảnh hưởng tới việc kết quả thử kết nối mạng.

## Lệnh Traceroute

Đây cũng là lệnh cho phép chẩn đoán hoạt động của mạng. Cú pháp của lệnh giống như lệnh **ping** nhưng kết quả không chỉ dừng ở sự trả lời mà còn chỉ ra các thiết bị trung gian nằm giữa 2 máy.

```
# tnminh@nefertiti ~ > traceroute 203.162.44.33
```

```
traceroute to 203.162.44.33 (203.162.44.33): 1-30 hops, 38 byte packets
 1  makeda.pasteur.fr (157.99.64.3), 1.66 ms, 1.66 ms, 1.66 ms
 2  418.ATM4-0.GW21.Defense.OLEANE.NET (195.25.28.149), 5.0 ms, 4.17 ms,
4.17 m
 3  FastEth0-0.GW16.Defense.OLEANE.NET (195.25.25.208), 4.17 ms, 4.17 ms,
4.17s
 4  100.ATM6-1.GW2.Telehouse.OLEANE.NET (194.2.3.245), 5.0 ms, 5.0 ms, 5.0
ms
.....
```

```
14 210.132.93.210 (210.132.93.210), 849 ms (ttl=241!), 807 ms (ttl=241!),  
970  
s (ttl=241!)  
15 202.167.121.195 (202.167.121.195), 905 ms !H 203.162.3.42  
(203.162.3.42), 1  
88 ms (ttl=242!)
```

Lệnh **tracert** là một công cụ hiệu quả cho phép ta phát hiện lỗi trong quá trình phân đường (IP routing). Ví dụ kết nối từ A -> C có trục trặc và với **tracert** tới C từ máy A, ta có thể phát hiện ra máy A kết nối máy B, rồi máy B lại kết nối máy A ... do cấu hình routing của A và B sai.

Chú ý là khi chúng ta thử kết nối với một máy ở xa trong Internet, do nhiều mạng áp dụng các bức tường lửa (firewall) nên nhiều khi lệnh ping và **tracert** không chạy nhưng trên thực chất là mạng vẫn thông.

### 5.1.3. Các tiện ích mạng: Telnet và ftp

- **Telnet**

Telnet là một tiện ích cho phép đăng nhập vào một máy tính ở xa và làm việc giống như với máy tại chỗ. Ví dụ, có thể dùng telnet để chạy một chương trình trong một siêu máy tính ở cách xa hàng ngàn dặm. Telnet sử dụng giao thức TCP/IP, cổng 23.

Sử dụng: giả sử máy của bạn đang chạy Window và bạn đã được cấp một tài khoản trong máy chủ Linux.

1. Nhấn chuột vào "Start" chọn "Run".
2. Gõ vào: "**telnet** <tên hay địa chỉ IP>" của máy chủ mà bạn có tài khoản. Ví dụ "telnet linuxcourse.iti.edu.vn" và nhấn OK.
3. Ngay khi kết nối đến máy chủ thông suốt, một cửa sổ sẽ hiện lên mời bạn cung cấp tên tài khoản và mật khẩu.
4. Nhập vào tên tài khoản *username* và *password* để đăng nhập.
5. Đăng nhập thành công thì bạn sẽ đứng tại thư mục nhà (home directory) của mình.
6. Bắt đầu phiên làm việc của bạn. Ví dụ, dùng câu lệnh "**ls -al**" để hiển thị tất cả các tệp trong thư mục.
7. Kết thúc phiên làm việc, gõ "**exit**".

- **FTP**

FTP là viết tắt của Tệp Transfer Protocol, một tiện ích tải tệp ở xa. Với ftp có thể lấy tệp ở máy từ xa về máy tính của mình (download) và ngược lại, gửi một tệp từ máy của mình lên máy ở xa (upload) nếu bạn có quyền write vào thư mục ở máy đó. FTP sử dụng giao thức TCP/IP, cổng 21.

Sử dụng FTP

Cách tải xuống (download):

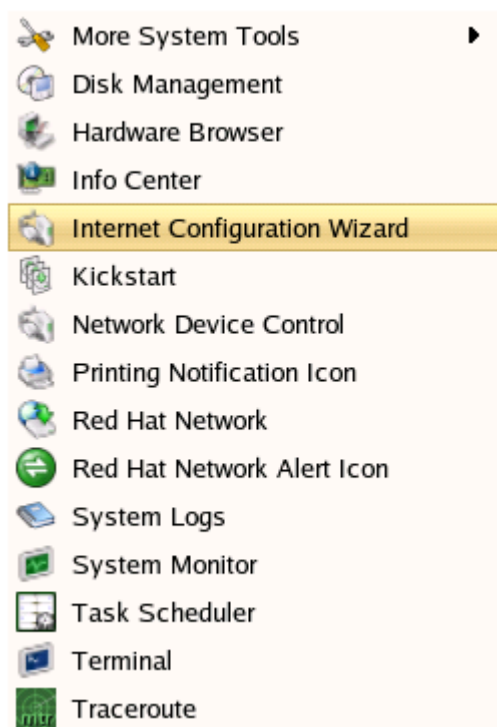
- Telnet vào máy ở xa.
- Gõ lệnh **ftp** <tên máy ở xa>.
- Máy sẽ yêu cầu tên đăng nhập và password. Một trong những chế độ cho phép mọi người tải tệp về tự do là dùng tên đăng nhập "anonymous" và password là địa chỉ email của bạn.
- Chuyển đến thư mục có các tệp ta muốn tải về.
- Gõ lệnh: **get** <tên tệp muốn tải về>.
- Để kết thúc gõ **quit**.

Cách tải lên (upload): Tương tự như trên, nhưng dùng câu lệnh **put** thay cho câu lệnh **get**.

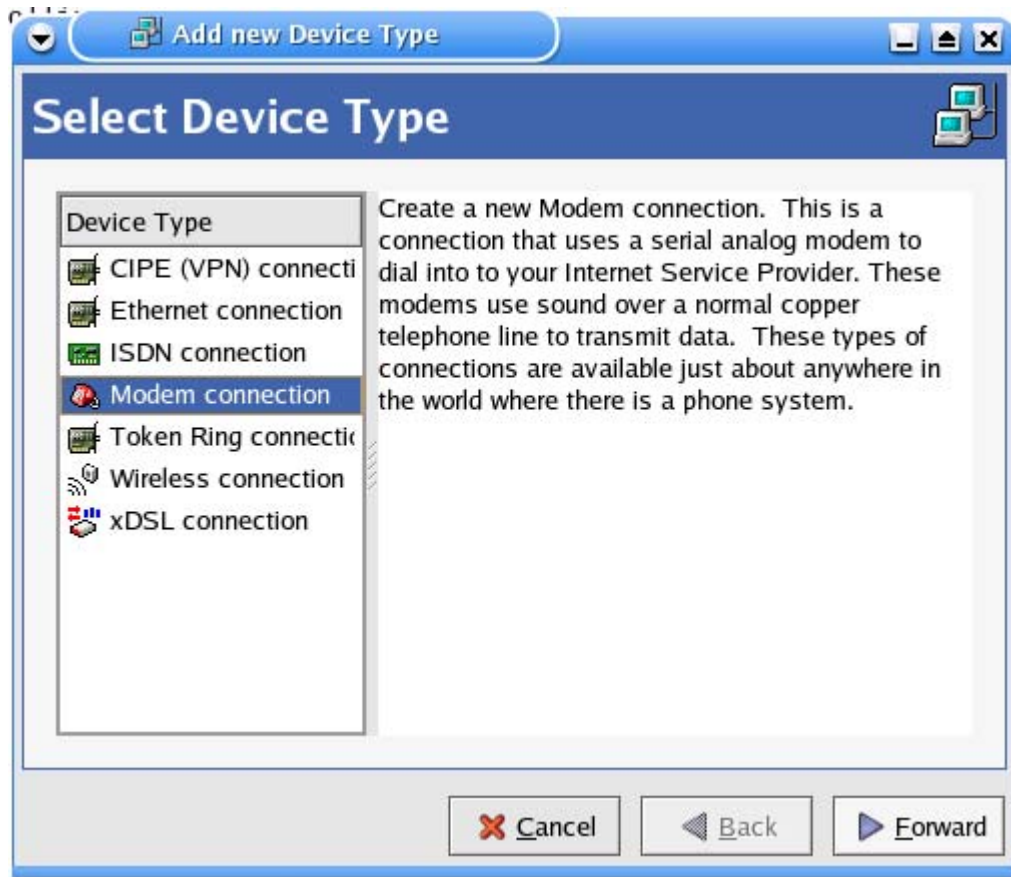
## 5.2. Cài đặt diul-up trên Linux

### 5.2.1. Cài đặt

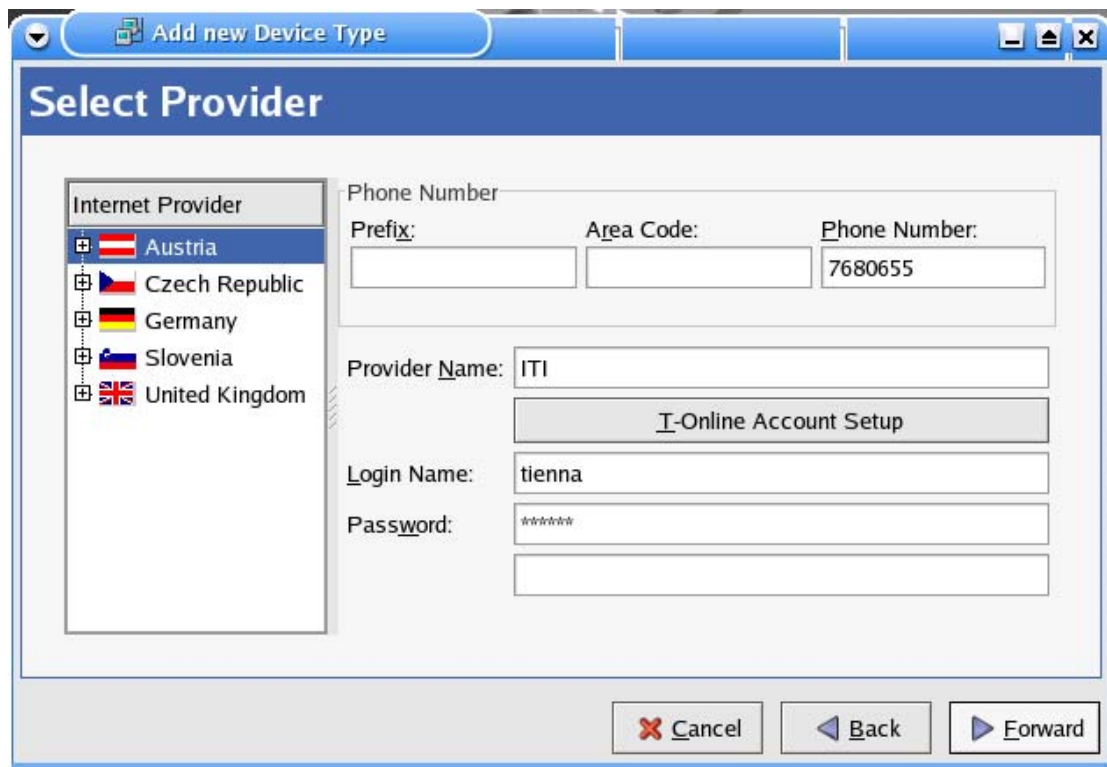
Chọn Internet Configuration Wizard từ menu System configuration



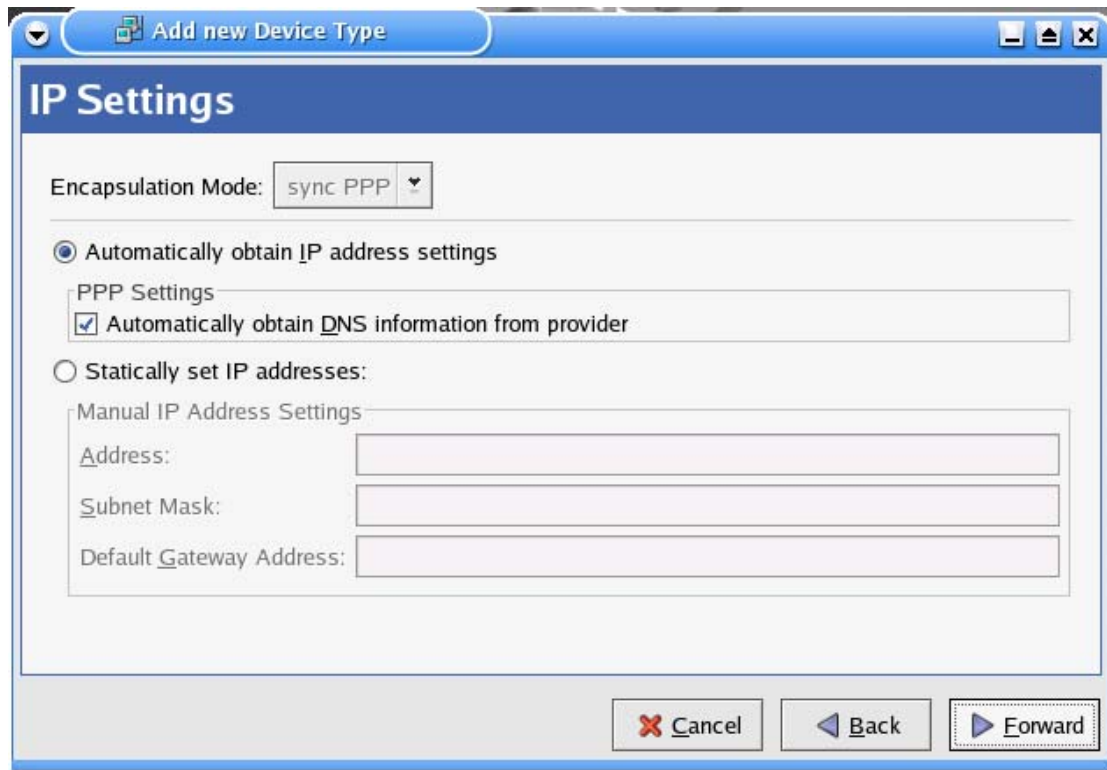
Sau đó màn hình này sẽ chỉ thị



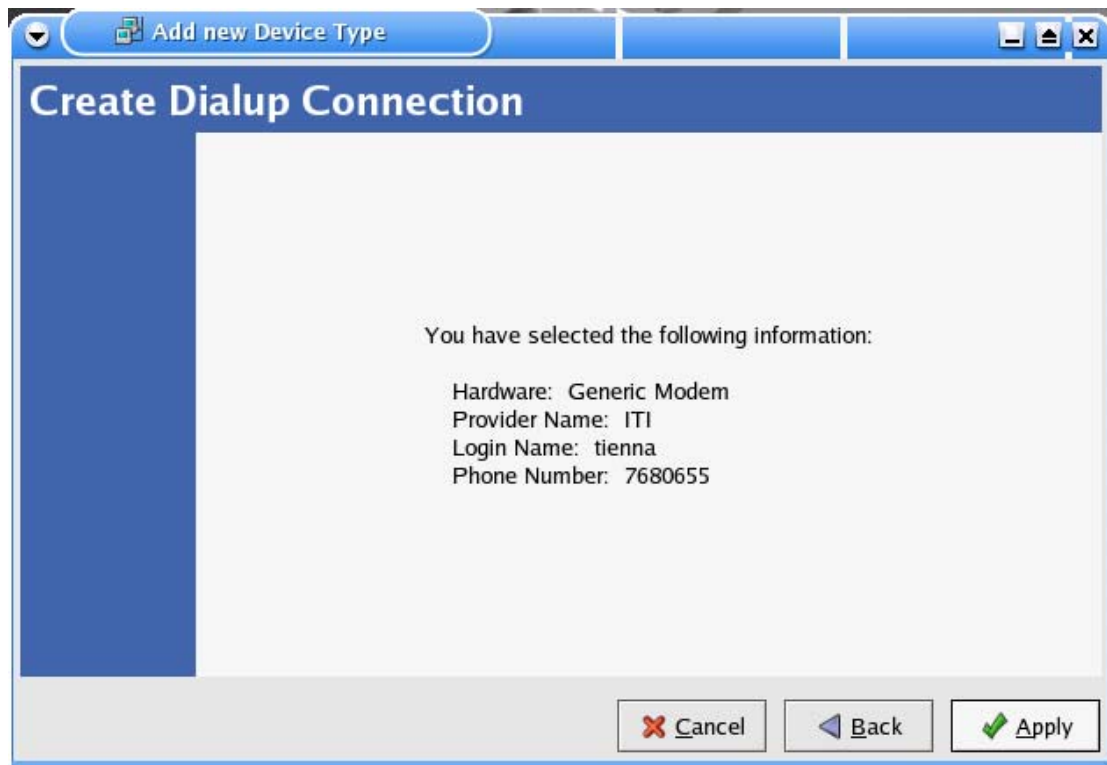
chọn Modem connection, chọn Forward.



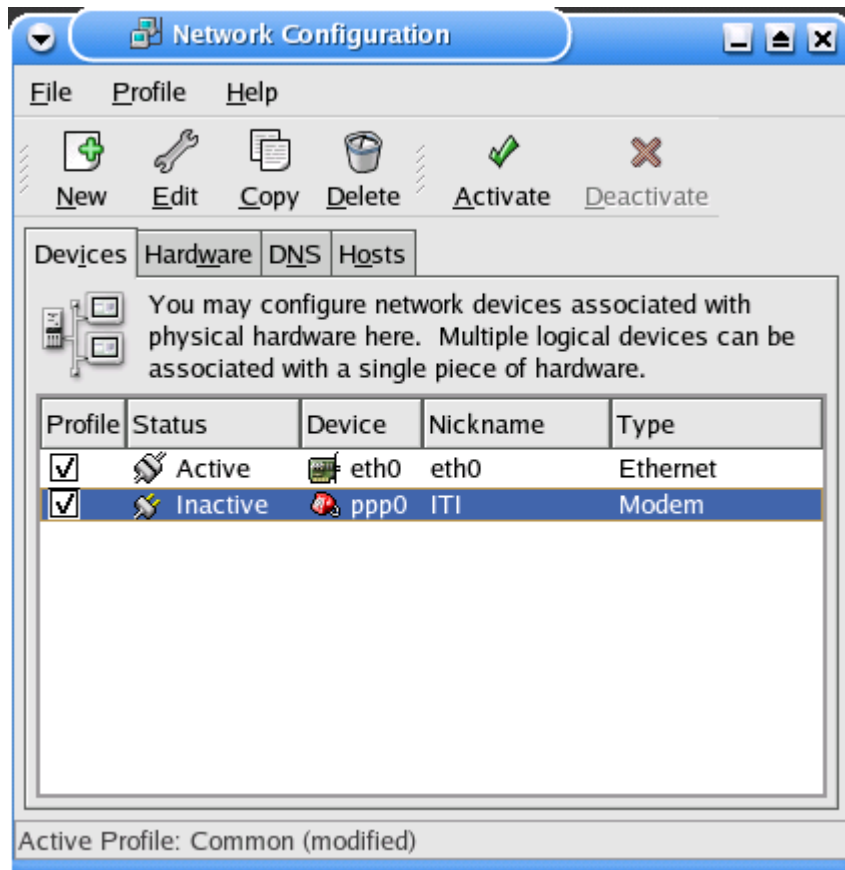
nhập vào các thông tin quay số., sau đó chọn Forward



Chọn gán IP động, chọn Forward



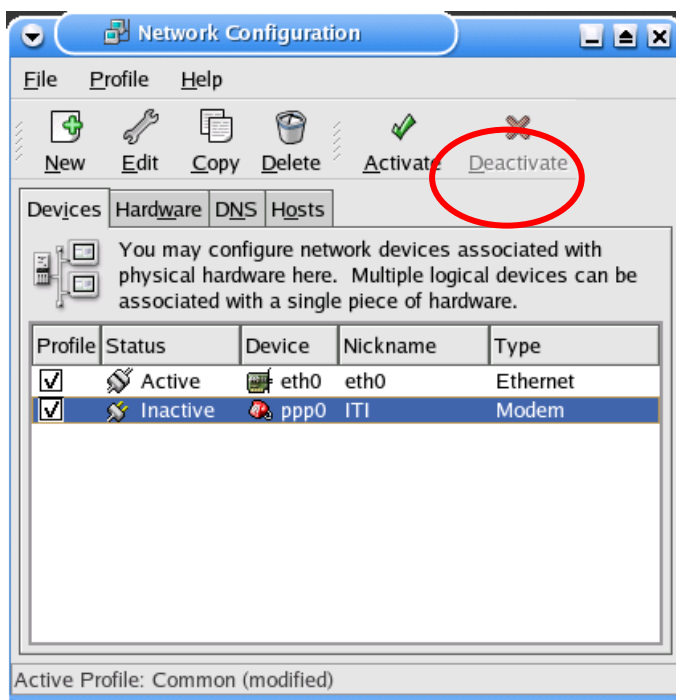
Chọn Apply, sau đó cửa sổ ả etwork configuration hiện ra



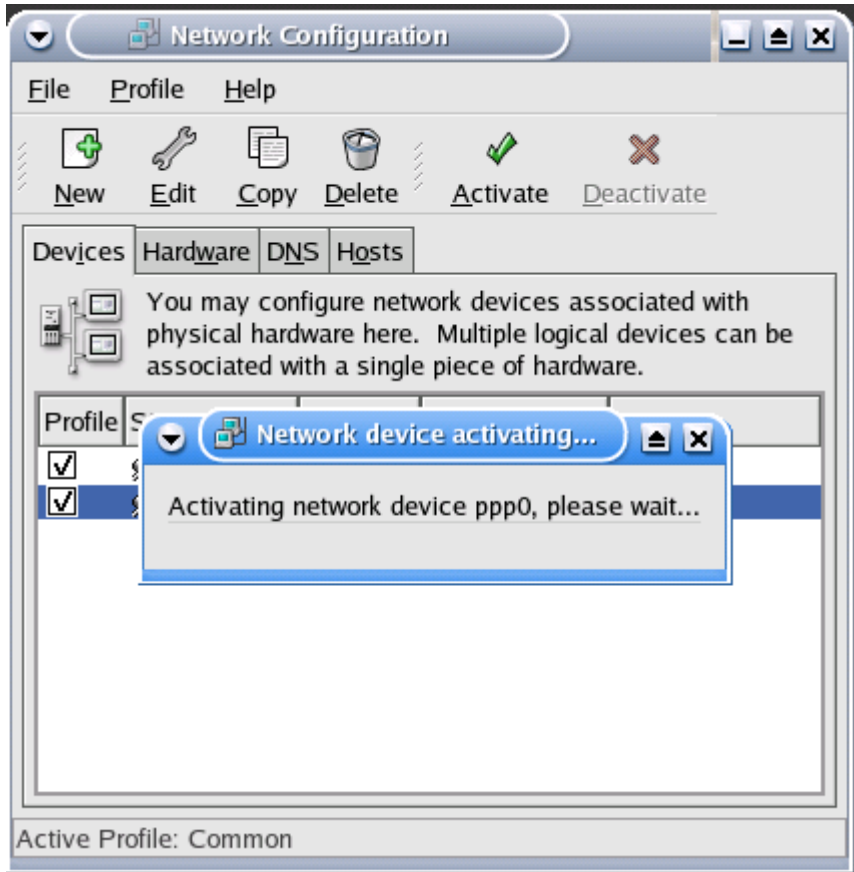
Đến đây chúng ta đã hoàn tất bước cài đặt modem.

### 5.2.2. Quay số

Tại màn hình này chọn giao diện ppp0 và click vào nút lệnh Avtive

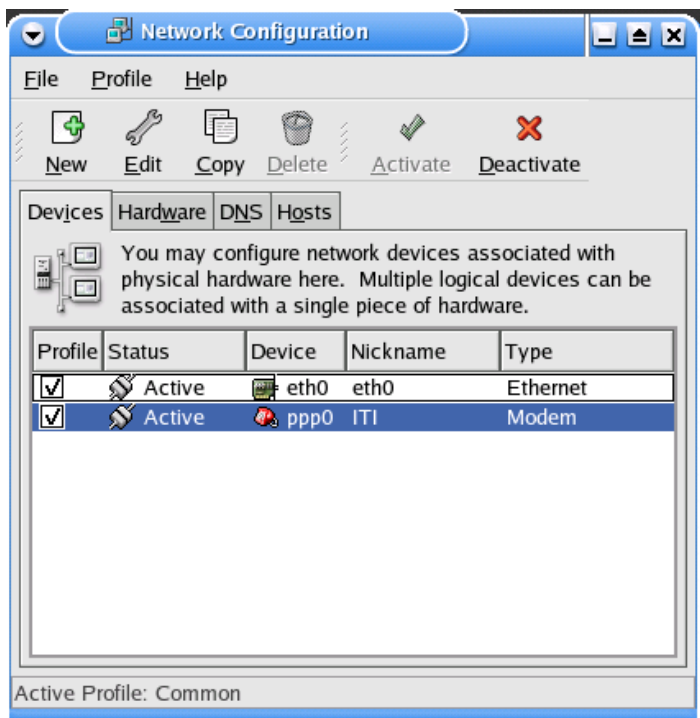


Máy tính bắt đầu quay số. file log sẽ được cất vào /var/log/message.

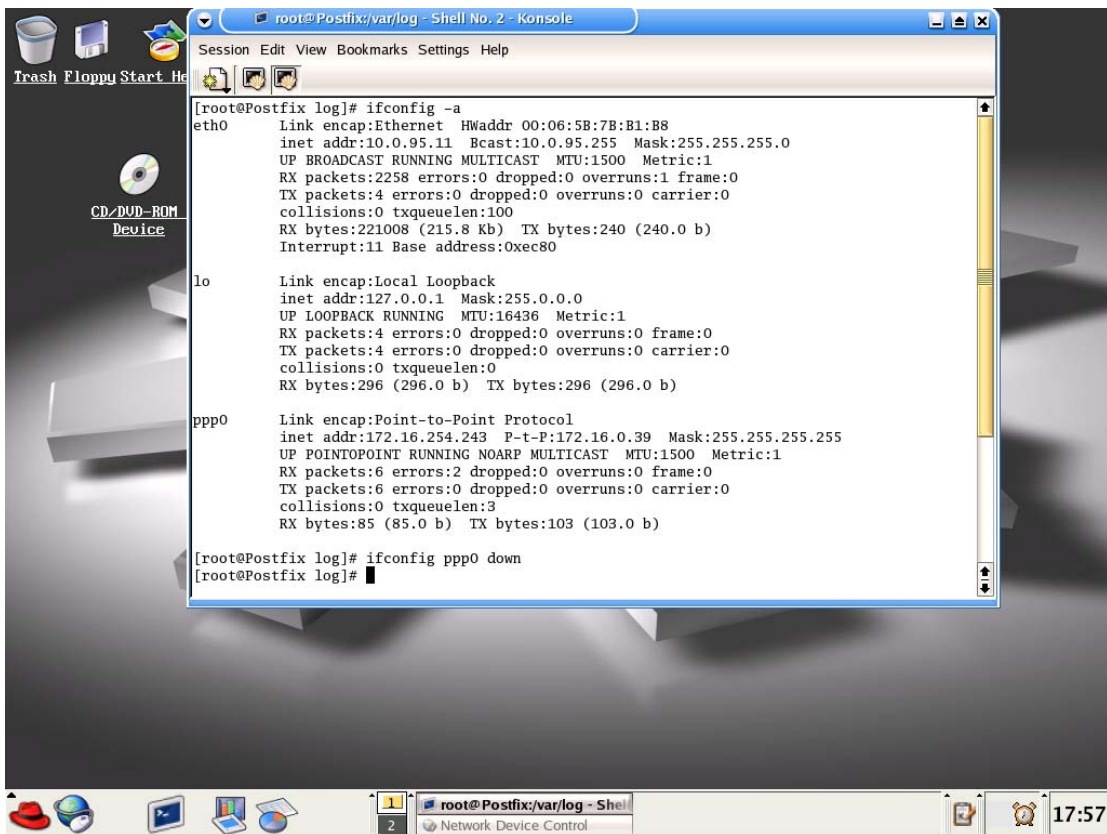


yess wait...

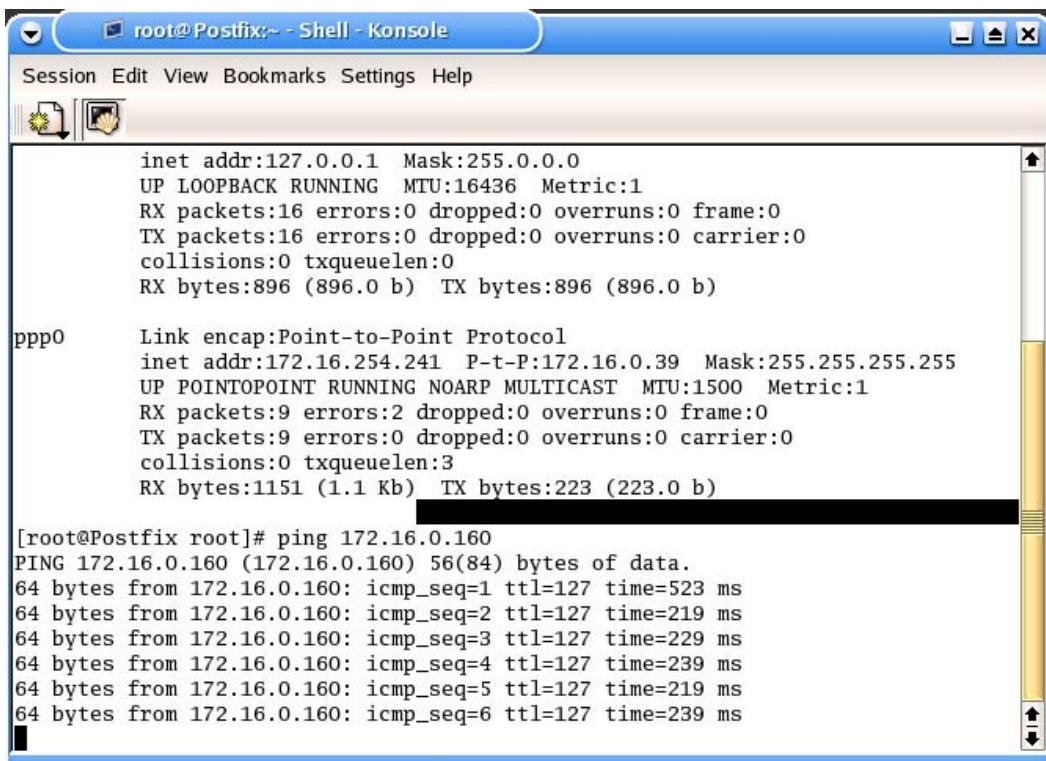
Khi xong màn hình network configuration sẽ báo giao diện ppp0 là active.



Có thể kiểm tra địa chỉ IP động và máy cung cấp DHCP qua lệnh “ifconfig -a”



Lúc này kết nối coi như đã được thiết lập, có thể dùng ping để kiểm tra.



Bây giờ thì chúng ta có thể truy cập internet thông qua trình duyệt.



## 6. Lập trình shell.

Lập trình shell là một trong những công cụ hữu ích nhất cho việc quản trị hệ thống. Khả năng viết một chương trình ngắn để hoàn thành một công việc đòi hỏi nhiều thời gian mạnh hơn rất nhiều so với các công cụ quản trị Linux khác được biết đến. Lập trình Shell có thể làm cho cuộc sống của người quản trị trở lên dễ thở hơn và nó là một kỹ năng bắt buộc đối với người quản trị Linux. Có thể nhận thấy có rất nhiều công việc của những người quản trị hệ thống đối mặt hàng ngày liên quan đến các file và thư mục. Bất cứ khi nào bạn phải xử lý với một số lượng lớn các file, lập trình shell sẽ làm cho công việc của bạn trở lên dễ dàng hơn. Phần này sẽ chỉ cho bạn cách lập trình Shell cơ bản, nó có thể giúp cho bạn thực hiện các công việc hàng ngày.

### 6.1. Tạo và chạy chương trình Shell

Ở một cách đơn giản nhất, lập trình shell chỉ là các file chứa một hoặc nhiều câu lệnh shell hay câu lệnh Linux. Bạn có thể sử dụng các chương trình đơn giản thực hiện các công việc lặp đi lặp lại, để thay cho hai hay nhiều câu lệnh luôn luôn được thực thi cùng nhau bằng một câu lệnh, để tự động cài đặt các chương trình khác, và để viết các ứng dụng tương tác đơn giản.

Để tạo một chương trình shell, bạn phải tạo một file sử dụng một trình soạn thảo và đưa các câu lệnh shell hay Linux mà bạn muốn được thực thi vào trong file. Giả sử rằng bạn có một ổ CD-ROM đã được gắn vào hệ thống Linux. Thiết bị CD-ROM này được gắn vào hệ thống khi hệ thống được khởi động lần đầu. Ở ầu bạn cần thay đổi đĩa CD đã có trong ổ CD bằng một đĩa CD mới. Một cách để bạn thực hiện được công việc này là bạn “nhả” ổ CD-ROM khỏi hệ thống sử dụng câu lệnh `umount`, và sau đó gắn lại ổ sử dụng câu lệnh `mount`. Các câu lệnh chỉ ra ở dưới đây cho bạn thấy tuần tự các bước thực hiện:

```
umount /dev/cdrom
```

```
mount /dev/cdrom /cdrom
```

Thay việc gõ cả hai câu lệnh mỗi lần bạn thay đổi đĩa CD, bạn có thể tạo một chương trình shell thực hiện cả hai câu lệnh này cho bạn. Để tạo chương trình shell này bạn đưa cả hai câu lệnh vào trong một file có tên là `remount` (hoặc một tên bất kỳ nào khác mà bạn muốn).

Có một vài cách để thực hiện các câu lệnh trong file `remount`. Cách thứ nhất là bạn thay đổi thuộc tính cho file này có thể thực thi bằng cách thực hiện câu lệnh sau:

```
chmod +x remount
```

Câu lệnh này thay đổi quyền của file làm cho file có thể thực thi. Để chạy chương trình shell mới, gõ `remount` trên dòng lệnh.

---

*Chương trình shell remount phải nằm trong một thư mục có trong đường dẫn tìm*

kiểm của bạn, nếu không hệ thống sẽ không tìm thấy chương trình để thực thi. Nếu bạn không chạy được chương trình bởi vì file đó không được tìm thấy, hãy xác định đường dẫn. Hoặc nếu bạn sử dụng tcsh để viết chương trình, dòng đầu tiên của chương trình shell phải bắt đầu với # để tcsh nhận ra nó như một file chương trình tcsh. Thực ra, cách an toàn (đảm bảo) nhất là ở dòng đầu của mỗi chương trình shell bạn thêm #!/bin/sh để đảm bảo chương trình shell được thực thi như một tiến trình Bourne shell. Điều này ngăn chặn nhiều vấn đề với ngôn ngữ lập trình C, shell sẽ cố gắng thông dịch cú pháp Bourne shell.

---

Một cách khác là bạn có thể thực thi chương trình shell là chạy shell mà chương trình được viết theo nó và tên chương trình như một khai báo cho shell. Trong trường hợp một chương trình tcsh, bạn thực hiện câu lệnh sau:

```
tcsh remount
```

Câu lệnh này chạy một shell mới và nói cho nó thực thi các câu lệnh trong file remount.

Cách thứ ba để thực thi các câu lệnh trong một file chương trình shell là sử dụng câu lệnh . (dấu chấm) với cả shell pdksh và bash hoặc câu lệnh source trong shell tcsh. Các câu lệnh này nói cho shell thực thi file được truyền vào như đối số. Ví dụ, bạn có thể sử dụng câu lệnh sau để nói cho bash hoặc pdksh thực thi các câu lệnh trong file remount:

```
. remount
```

Để làm tương tự đối với tcsh, sử dụng câu lệnh sau:

```
source remount
```

Ví dụ sau trình bày một tình huống khác, trong đó việc sử dụng chương trình shell sẽ giúp tiết kiệm rất nhiều thời gian. Giả sử rằng bạn đã phải làm việc với ba file khác nhau trong một thư mục mỗi ngày, và bạn muốn dự phòng ba file này vào một đĩa mềm vào cuối mỗi ngày. Để thực hiện được công việc này, bạn phải gõ một loạt các lệnh:

```
mount -t msdos /dev/fd0 /a
```

```
cp file1 /dev/fd0
```

```
cp file2 /dev/fd0
```

```
cp file3 /dev/fd0
```

Một cách dự phòng các file là gắn ổ đĩa mềm vào hệ thống và sau đó gõ ba câu lệnh copy, mỗi lệnh cho một file bạn muốn copy. Một cách đơn giản hơn là đưa bốn câu lệnh này vào trong một file có tên là backup và sau đó thực hiện câu lệnh backup khi bạn muốn copy ba file này vào đĩa mềm.

Bạn vẫn phải đảm bảo chương trình file shell backup có thể thực thi và nằm trong một thư mục mà có trong đường dẫn của bạn trước khi chạy câu lệnh. Bạn hãy cẩn thận khi sử dụng một tên file, nó có thể tương ứng với tên của một câu lệnh hệ thống. Ví dụ, nếu có một chương trình được gọi là backup trong đường dẫn mà shell tìm kiếm trước khi đọc thư mục hiện tại, câu lệnh đó có thể được thi thay cho file câu lệnh shell. Vì lý do này, hãy cố sử dụng các tên file cho kịch bản shell của bạn không gán với các câu lệnh Linux.

---

## 6.2. Sử dụng các biến

Cũng giống như với hầu hết các ngôn ngữ lập trình, việc sử dụng các biến là rất quan trọng trong các chương trình shell. Tất nhiên, bạn đã được nhìn thấy một vài kiểu biến trước đó. Một vài ví dụ nói chung về biến được sử dụng là biến PATH và biến TERM. Các biến này là các ví dụ về các biến shell sẵn có, là các biến được định nghĩa bởi chương trình shell mà bạn đang sử dụng. Phần này miêu tả cách làm thế nào để bạn tạo các biến của chính bạn và sử dụng chúng trong một vài chương trình shell.

### 6.2.1. Gán một giá trị cho một biến

Trong cả ba shell được cung cấp bởi Linux ( shell Bourne, Korn, và C ), bạn có thể gán một giá trị cho một biến bằng cách gõ tên biến theo sau bởi dấu bằng và sau đó gõ giá trị mà bạn muốn gán cho biến. Ví dụ, để gán một giá trị 5 cho một biến có tên là count, vào câu lệnh sau trong bash hoặc pdksh:

```
count=5
```

Với tcsh, vào câu lệnh sau để đạt được kết quả tương tự:

```
set count = 5
```

---

Khi thiết lập một biến cho shell bash và pdksh, hãy chắc chắn rằng không có dấu cách ở cả hai bên dấu bằng. Với tcsh, điều này không quan trọng.

---

Bởi vì ngôn ngữ shell là một ngôn ngữ kịch bản phi kiểu, bạn không phải khai báo biến như bạn có thể đã từng làm điều này trong lập trình C hay Pascal. Bạn có thể sử dụng cùng một biến để lưu trữ xâu ký tự hay số nguyên. Bạn lưu một chuỗi ký tự vào trong một biến cũng giống như việc bạn lưu một số nguyên vào một biến, như có thể thấy trong ví dụ dưới đây:

```
name=Garry (for pdksh and bash)
```

```
set name = Garry (for tcsh)
```

Sau khi bạn lưu một giá trị vào một biến, bạn làm thế nào để có thể lấy giá trị đó trở lại? Bạn đặt trước tên biến với dấu đô la (\$). Để in giá trị được lưu trữ trong biến count ra màn hình, vào câu lệnh sau:

```
echo $count
```

À ếu bạn quên dấu \$ trước câu lệnh, lệnh echo sẽ hiển thị từ “count” trên màn hình.

### 6.2.2. Tham số và các biến Shell cơ bản

Khi bạn chạy chương trình shell yêu cầu hay hỗ trợ một số các tùy chọn dòng lệnh, mỗi tùy chọn này được lưu trữ trong một đối số. Đối số đầu tiên được lưu trữ trong một biến có tên là 1, đối số thứ hai được lưu trữ trong biến có tên là 2, và tiếp tục như thế. Shell đặt tên các biến này, vì vậy bạn không thể đặt tên như thế cho các biến mà bạn định nghĩa. Để lấy giá trị từ các biến này, bạn phải đặt trước tên biến với một dấu \$ như bạn làm đối các biến mà bạn định nghĩa.

Chương trình shell reverse dưới đây chờ nhận hai đối số. Chương trình lấy hai đối số dòng lệnh và in ra đối số thứ hai ở dòng đầu tiên và đối số đầu tiên ở dòng thứ hai:

```
echo "$2"
```

```
echo "$1"
```

À ếu bạn gọi tới chương trình bằng cách gõ dòng lệnh sau

```
reverse hello there
```

Chương trình sẽ trả lại kết quả

```
there hello
```

Một số các biến shell quan trọng được xây dựng sẵn mà bạn cần biết khi làm việc nhiều với lập trình shell. Bảng 6.2.1 đưa ra danh sách các biến này và mô tả tóm tắt mỗi biến được sử dụng để làm gì.

**Bảng 6.2.1 Các biến shell có sẵn.**

Biến	Sử dụng
\$#	Lưu số các đối số dòng lệnh được đưa vào chương trình shell
\$?	Lưu giá trị tồn tại của câu lệnh được thực thi sau cùng
\$0	Lưu từ đầu tiên của câu lệnh được đưa vào, đó là tên của chương trình shell
\$*	Lưu tất cả các đối số được đưa vào từ dòng lệnh (" \$1 \$2 ...")

"\$@"	Lưu tất cả các đối số được đưa vào từ dòng lệnh, có dấu nháy kép riêng ("\$1" "\$2" ...)
-------	--

### 6.3. Sử dụng dấu trích dẫn

Việc sử dụng các dấu trích dẫn là rất quan trọng trong lập trình shell. Shell sử dụng cả hai kiểu dấu trích dẫn và ký tự và dấu gạch chéo ngược để thực hiện các chức năng khác nhau. Cả dấu nháy kép (""), dấu nháy đơn (''), và dấu gạch ngược (\) được sử dụng để ẩn các ký tự đặc biệt trong shell. Các dấu nháy có một ý nghĩa đặc biệt trong shell và nó không nên sử dụng để chứa các xâu. Mỗi một phương thức có một mức độ che dấu khác nhau các ký tự đặc biệt trong shell.

Khi bạn bao quanh các ký tự với dấu nháy kép, tất cả các ký tự trong shell, nhưng tất cả các ký tự khác vẫn được thông dịch. Kiểu dấu nháy kép này sử dụng hữu ích nhất khi bạn gán các chuỗi chứa nhiều hơn một từ vào một biến. Ví dụ, để gán chuỗi *hello there* cho biến `greeting`, nhập vào câu lệnh sau:

```
greeting="hello there" (in bash and pdksh)
set greeting = "hello there" (in tcsh)
```

Câu lệnh này lưu trữ toàn bộ chuỗi *hello there* vào biến `greeting` như một từ. Nếu bạn gõ vào câu lệnh mà không sử dụng dấu nháy kép, `bash` và `pdksh` có thể không hiểu câu lệnh và có thể trả lại một thông báo lỗi, và `tcsh` có thể gán giá trị `hello` cho biến `greeting` và bỏ qua phần đuôi của dòng lệnh.

Dấu nháy đơn là hình thức sử dụng mạnh nhất của dấu nháy. Chúng ẩn tất cả các ký tự đặc biệt trong shell. Kiểu dấu nháy này hữu ích nếu câu lệnh của bạn đưa vào có dụng ý cho một chương trình hơn là cho shell. Ví dụ, bạn có thể sử dụng dấu nháy đơn để ghi chuỗi *hello there*, nhưng bạn không thể sử dụng phương thức này trong một số trường hợp. Ví dụ, nếu chuỗi được gán cho biến `greeting` chứa biến khác, bạn phải sử dụng dấu nháy kép. Giả sử rằng bạn muốn đưa tên của người sử dụng trong biến `greeting`. Bạn gõ câu lệnh sau:

```
greeting="hello there $LOGNAME" (for bash and pdksh)
set greeting="hello there $LOGNAME" (for tcsh)
```

---

*Biến `LOGNAME` là một biến shell chứa tên đăng nhập của người sử dụng Linux đã đăng nhập hệ thống.*

---

Câu lệnh này lưu trữ giá trị *hello there root* vào trong biến `greeting` nếu bạn đã đăng nhập vào Linux là `root`. Nếu bạn cố ghi câu lệnh này sử dụng dấu nháy đơn, dấu nháy

đơn sẽ làm ẩn dấu \$ trong shell, và shell không biết rằng nó được yêu cầu thực hiện thay thế một biến. Kết quả, biến greeting được gán giá trị *hello there \$LOGNAME*.

Sử dụng dấu gạch ngược là cách thứ ba để che dấu các ký tự đặc biệt trong shell. Giống như phương thức dấu nháy đơn, dấu gạch ngược ẩn tất cả các ký tự đặc biệt trong shell, nhưng nó chỉ có thể ẩn một ký tự tại một thời điểm, chứ không phải một nhóm các ký tự. Bạn có thể viết lại ví dụ greeting sử dụng dấu gạch ngược thay cho dấu nháy kép bằng cách sử dụng câu lệnh sau:

```
greeting=hello\ there (for bash and pdksh)
set greeting=hello\ there (for tcsh)
```

Trong câu lệnh này, dấu gạch ngược ẩn ký tự trống trong shell và chuỗi *hello there* được gán cho biến greeting.

Dấu gạch ngược thường được sử dụng nhiều nhất khi bạn muốn ẩn chỉ một ký tự trong shell. Vấn đề này xuất hiện khi bạn muốn đưa vào một ký tự đặc biệt trong một chuỗi. Ví dụ, để lưu giá của một hộp đĩa máy tính vào một biến có tên là *disk\_price*, sử dụng câu lệnh sau.

```
disk_price=\$5.00 (for bash and pdksh)
set disk_price = \$5.00 (tcsh)
```

Dấu gạch ngược trong ví dụ này ẩn dấu đô la trong shell. Ắt ều dấu gạch ngược không có ở đó, shell có thể cố tìm một biến có tên là 5 và thực hiện một phép thay thế biến trên biến đó. Ắt ều không có biến tên là 5 được định nghĩa, shell có thể một gán giá trị .00 cho biến *disk\_price*. ( shell này có thể thay thế một giá trị rỗng cho biến \$5 ) Bạn cũng có thể sử dụng dấu nháy đơn trong ví dụ *disk\_price* để ẩn ký hiệu \$ trong shell.

Dấu nháy ngược (``) thực hiện một chức năng khác. Bạn sử dụng chúng khi bạn muốn sử dụng các kết quả của một câu lệnh trong một câu lệnh khác. Ví dụ, để đặt giá trị của biến *contents* bằng danh sách các file có trong thư mục hiện tại, gõ câu lệnh sau:

```
contents=`ls` (for bash and pdksh)
set contents = `ls` (for tcsh)
```

Câu lệnh này thực thi câu lệnh *ls* và lưu kết quả của câu lệnh vào biến *contents*. Ắt ều sẽ được chỉ ra trong các đoạn sau, đặc điểm này có thể rất hữu ích khi bạn muốn ghi kết quả của một chương trình shell thực hiện một vài hoạt động vào trong một câu lệnh khác.

#### 6.4. Sử dụng câu lệnh test

Trong *bash* và *pdksh*, câu lệnh *test* được sử dụng để tính giá trị của một biểu thức có điều kiện. Thông thường, bạn sử dụng câu lệnh *test* để tính giá trị điều kiện trong một

lệnh có điều kiện hoặc tính giá trị đầu vào hay điều kiện tồn tại cho một câu lệnh lập. Câu lệnh test có cú pháp sau:

`test expression`

hoặc

`[ expression ]`

Bạn có thể sử dụng một vài toán tử có sẵn với câu lệnh test. Các toán tử này được phân loại thành bốn nhóm khác nhau: các toán tử chuỗi, các toán tử số, các toán tử file, và các toán tử logic.

Bạn sử dụng các toán tử chuỗi để tính giá trị biểu thức chuỗi. Bảng 6.4.1 đưa ra danh sách các toán tử chuỗi mà ba ngôn ngữ lập trình shell hỗ trợ.

**Bảng 6.4.1 Các toán tử chuỗi cho câu lệnh test.**

Toán tử	Ý nghĩa
<code>str1 = str2</code>	Trả lại giá trị true nếu str1 giống với str2
<code>str1 != str2</code>	Trả lại giá trị true nếu str1 không giống str2
<code>str</code>	Trả lại giá trị true nếu str khác rỗng
<code>-n str</code>	Trả lại giá trị true nếu độ dài của str lớn hơn 0
<code>-z str</code>	Trả lại giá trị true nếu độ dài của str bằng 0

Các toán tử số thực hiện các chức năng tương tự các toán tử string ngoại trừ việc chúng hoạt động trên các đối số kiểu số. Bảng 6.4.2 liệt kê danh sách các toán tử số được sử dụng trong câu lệnh test.

**Bảng 6.4.2 Các toán tử số cho câu lệnh test.**

Toán tử	Ý nghĩa
<code>int1 -eq int2</code>	Trả lại giá trị true nếu int1 bằng int2
<code>int1 -ge int2</code>	Trả lại giá trị true nếu int1 lớn hơn hoặc bằng int2
<code>int1 -gt int2</code>	Trả lại giá trị true nếu int1 lớn hơn int2
<code>int1 -le int2</code>	Trả lại giá trị true nếu int1 nhỏ hơn int2
<code>int1 -lt int2</code>	Trả lại giá trị true nếu int1 nhỏ hơn int2
<code>int1 -ne int2</code>	Trả lại giá trị true nếu int1 không bằng int2

Bạn sử dụng các toán tử file cho câu lệnh test để thực hiện các chức năng chẳng hạn như kiểm tra để xem các file có tồn tại hay không và kiểm tra để xem file thuộc loại nào, file được đưa vào như một đối số cho câu lệnh test. Bảng 6.4.3 đưa ra danh sách các toán tử file cho câu lệnh test.

**Bảng 6.4.3 Các toán tử File cho câu lệnh test.**

Toán tử	Ý nghĩa
-d file	Trả lại giá trị true nếu file được xác định là một thư mục
-f file	Trả lại giá trị true nếu file được xác định là một file thông thường
-r file	Trả lại giá trị true nếu file xác định là có thể đọc bởi tiến trình
-s file	Trả lại giá trị true nếu file xác định có độ dài khác 0
-w file	Trả lại giá trị true nếu file có thể ghi được bởi tiến trình
-x file	Trả lại giá trị true nếu file xác định là có thể thực thi

Bạn sử dụng các toán tử logic cho câu lệnh test để kết hợp các toán tử số, xâu, hay file hoặc phủ định một toán tử đơn số, xâu, hoặc file. Bảng 6.4.4 đưa ra danh sách các toán tử logic cho câu lệnh test.

**Bảng 6.4.4 Các toán tử Logic cho câu lệnh test.**

Toán tử	Ý nghĩa
! expr	Trả lại giá trị true nếu expr khác true
Expr1 -a expr2	Trả lại giá trị true nếu expr1 và expr2 là true
Expr1 -o expr2	Trả lại giá trị true nếu expr1 hoặc expr2 là true

Shell tesh không có câu lệnh test, nhưng các biểu thức của tesh thực hiện các chức năng tương tự. Các toán tử tesh hỗ trợ hầu hết giống như được hỗ trợ trong ngôn ngữ C. Bạn thường sử dụng các biểu thức này trong các câu lệnh if và while. Trong đoạn sau, phần "Sử dụng các lệnh có điều kiện" và "Sử dụng các lệnh lặp" sẽ nói về các câu lệnh này. Giống như câu lệnh test trong bash và pdksh, các biểu thức trong tesh hỗ trợ các toán tử số, xâu, file, và logic. Bảng 6.4.5 đưa ra danh sách các toán tử được hỗ trợ trong các biểu thức của tesh.

**Bảng 6.4.5 Các toán tử số cho for các biểu thức tesh.**

Toán tử	Ý nghĩa
int1 <= int2	Trả lại giá trị true nếu int1 nhỏ hơn int2
int1 >= int2	Trả lại giá trị true nếu int1 lớn hơn hoặc bằng int2
int1 < int2	Trả lại giá trị true nếu int1 nhỏ hơn int2
int1 > int2	Trả lại giá trị true nếu int1 lớn hơn int2

Bảng 6.4.6 đưa ra danh sách các toán tử xâu mà các biểu thức của tesh hỗ trợ.

**Table 6.4.6. Các toán tử xâu cho các biểu thức của tesh.**

Toán tử	Ý nghĩa
---------	---------



<code>str1 == str2</code>	Trả lại giá trị true nếu str1 bằng str2
<code>str1 != str2</code>	Trả lại giá trị true nếu str1 không bằng str2

Bảng 6.4.7 đưa ra danh sách cá toán tử file mà các biểu thức tesh hỗ trợ.

**Bảng 6.4.7 Các toán tử File cho các biểu thức tesh.**

Toán tử	Ý nghĩa
<code>-r file</code>	Trả lại giá trị true nếu file có thể đọc được
<code>-w file</code>	Trả lại giá trị true nếu file có thể ghi được
<code>-x file</code>	Trả lại giá trị true nếu file có thể thực thi
<code>-e file</code>	Trả lại giá trị true nếu file tồn tại
<code>-o file</code>	Trả lại giá trị true nếu file được sở hữu bởi người sử dụng hiện tại
<code>-z file</code>	Trả lại giá trị true nếu file có kích thước bằng 0
<code>-f file</code>	Trả lại giá trị true nếu file là file thông thường
<code>-d file</code>	Trả lại giá trị true nếu file là một thư mục

Bảng 6.4.8 Đưa ra danh sách các toán tử logic được hỗ trợ trong các biểu thức của tesh.

**Table 6.4.8 Các toán tử Logical cho cá biểu thức của tesh.**

Toán tử	Ý nghĩa
<code>exp1    exp2</code>	Trả lại giá trị true nếu exp1 là true hoặc exp2 là true
<code>exp1 &amp;&amp; exp2</code>	Trả lại giá trị true nếu cả hai exp1 và exp2 là true
<code>! exp</code>	Trả lại true nếu exp khác true

## 6.5. Sử dụng các câu lệnh rẽ nhánh

Trong các shell bash, pdksh và tesh, mỗi shell có hai hình thức khác nhau của câu lệnh rẽ nhánh. Bạn sử dụng các lệnh này để thực thi các phần khác nhau của chương trình shell phụ thuộc vào các điều kiện nhất định có đúng hay không. Với hầu hết các lệnh thực hiện, cú pháp cho các câu lệnh này khác nhau giữa các shell.

### 6.5.1. Lệnh if

Tất cả ba shell đều hỗ trợ các câu lệnh if-then-else statements lồng nhau. Các lệnh này cung cấp cho bạn cách thực hiện các câu lệnh test điều kiện phức tạp trong chương trình shell của bạn. Cú pháp của lệnh if trong bash và pdksh là giống nhau:

```
if [ expression ]
then
```

```
    commands

elif [ expression2 ]

    commands

else

    commands

fi
```

Chú ý rằng shell bash và pdksh sử dụng đảo ngược của tên câu lệnh trong hầu hết các lệnh phức tạp để kết thúc câu lệnh. Trong lệnh bên trên, từ khóa fi được sử dụng để làm kí hiệu kết thúc cho câu lệnh if.

Cả hai mệnh đề elif và else đều là các phần tùy chọn của lệnh if. Lệnh elif là rút gọn của else if. Lệnh này được thực thi nếu các biểu thức nằm trong lệnh if hoặc tất cả các biểu thức trong các lệnh elif ở trước đó đều không có giá trị true. Các câu lệnh nằm trong lệnh else được thực thi chỉ nếu không một biểu thức nào trong mệnh đề if và trong bất kỳ mệnh đề elif nào có giá trị true.

Trong tcsh, lệnh if có hai dạng khác nhau. Dạng thứ nhất cung cấp cùng một chức năng như lệnh if trong bash và pdksh. Dạng này của lệnh if có cú pháp như sau:

```
if (expression1) then

    commands

else if (expression2) then

    commands

else

    commands

endif
```

Lại một lần nữa các phần if và else của lệnh if là tùy chọn. Lệnh này cũng có thể được viết với elif. Ồ ồ mà ở bên trên trình bày toán bộ chương trình tcsh, nó nên bắt đầu với dòng sau để đảm bảo chạy hoàn hảo:

```
#!/bin/sh
```

Dạng thứ hai của lệnh if mà tcsh cung cấp là biến đổi đơn giản của lệnh if dạng đầu tiên. Dạng này của lệnh if chỉ tính giá trị một biểu thức đơn. Ồ ồ biểu thức là true nó sẽ thực thi câu lệnh đơn. Ồ ồ biểu thức là false, không có điều gì xảy ra. Cú pháp cho dạng này của lệnh if là như sau.

```
if (expression) command
```

Bên dưới là một ví dụ về sử dụng lệnh if trong bash hay pdksh. Lệnh này kiểm tra xem có một file có tên là `.profile` trong thư mục hiện tại hay không:

```
if [ -f .profile ]
then
    echo "There is a .profile file in the current directory."
else
    echo "Could not find the .profile file."
fi
```

Cũng với ví dụ trên sử dụng cú pháp của tcsh như sau:

```
#
if ( { -f .profile } ) then
    echo "There is a .profile file in the current directory."
else
    echo "Could not find the .profile file."
endif
```

Chú ý rằng trong ví dụ tcsh dòng đầu tiên bắt đầu với ký tự `#`. Ký hiệu này được yêu cầu để tcsh nhận ra file chứa các câu lệnh là một file kịch bản tcsh.

### 6.5.2. Lệnh case

Lệnh case cho phép bạn so sánh một mẫu với một số các mẫu khác và thực thi một khối mã nếu một sự giống nhau được tìm thấy. Lệnh case trong shell mạnh hơn lệnh case trong Pascal hay lệnh switch trong C. Với lệnh shell trong case, bạn có thể so sánh các chuỗi với các ký tự đại diện trong chúng; bạn có thể chỉ có thể so sánh các kiểu được liệt kê hoặc các giá trị số nguyên trong Pascal và C.

Cú pháp cho lệnh case trong bash và pdksh là như sau:

```
case string1 in
    str1)
        commands;;
    str2)
        commands;;
    *)
```

```
commands;;  
esac
```

String1 được so sánh với str1 và str2. Ắt ít một trong các xâu này hợ với string1, các câu lệnh bên dưới nó cho đến khi gặp hai dấu chấm phẩy(;;) được thực hiện. Ắt ít không có xâu nào (str1 hoặc str2) hợ với string1, các câu lệnh kết hợ với dấu hoa thị được thực thi. Các câu lệnh này là điều kiện case mặc định bởi vì dấu hoa thị hợ với tất cả các xâu.

Câu lệnh trong tesh tương đương với câu lệnh case trong bash và pdksh được gọi là lệnh switch. Lệnh này gần gũi với cú pháp lệnh switch trong C. Cú pháp cho lệnh switch là như sau:

```
switch (string1)  
  
case str1:  
  
statements  
  
breaksw  
  
case str2:  
  
statements  
  
breaksw  
  
default:  
  
statements  
  
breaksw  
  
endsw
```

Lệnh này xử lý giống như cách sử lý của lệnh case trong bash và pdksh. Mỗi xâu trong từ khóa case được so sánh với string1. Ắt ít xâu bất kỳ trong các xâu trên hợ với string1, các mã bên dưới nó cho đến khi gặp từ khóa breaksw keyword được thực hiện. Ắt ít không có xâu nào phù hợ, các mã ở bên dưới từ khóa default cho đến khi gặp từ khóa breaksw được thực thi.

Mã bên dưới là một ví dụ về lệnh case trong shell bash hay pdksh. Mã này kiểm tra xem tùy chọn đầu tiên trong dòng lệnh là -i hay -e. Ắt ít nó là -i, chương trình đếm số các dòng trong một file xác định bởi tùy chọn thứ hai trong dòng lệnh bắt đầu với ký tự i. Ắt ít tùy chọn thứ nhất là -e, chương trình đếm số các dòng trong file được xác định bởi tùy chọn thứ hai của dòng lệnh bắt đầu với ký tự e. Ắt ít tùy chọn thứ nhất của dòng lệnh khác -i và khác -e, chương trình sẽ in ra thông tin báo lỗi trên màn hình.

```
case $1 in  
  
-i)
```

```
count=`grep ^i $2 | wc -l`  
echo "The number of lines in $2 that start with an i is $count"  
;;  
-e)  
count=`grep ^e $2 | wc -l`  
echo "The number of lines in $2 that start with an e is $count"  
;;  
* )  
echo "That option is not recognized"  
;;  
esac
```

Ví dụ tương tự được viết theo cú pháp tcsh:

```
# remember that the first line must start with a # when using tcsh  
switch ( $1 )  
case -i | i:  
set count = `grep ^i $2 | wc -l`  
echo "The number of lines in $2 that begin with i is $count"  
breaksw  
case -e | e:  
set count = `grep ^e $2 | wc -l`  
echo "The number of lines in $2 that begin with e is $count"  
breaksw  
default:  
echo "That option is not recognized"  
breaksw  
endsw
```

## 6.6. Sử dụng các lệnh lặp

Ải gôn ngữ shell cũng cung cấp lệnh lặp mà thường được sử dụng nhất. Các lệnh lặp này được thao tác khi bạn cần thực hiện một hành động lặp đi lặp lại, chẳng hạn như khi bạn xử lý danh sách các file.

### 6.6.1. Lệnh for

Lệnh for thực thi các câu lệnh chứa trong nó một số lần. Lệnh for có hai dạng khác nhau trong bash và pdksh. Dạng thứ nhất của lệnh for mà bash và pdksh hỗ trợ có cú pháp như sau:

```
for var1 in list
do
    commands
done
```

Trong dạng này, lệnh for thực thi một lần cho mỗi phần tử nằm trong danh sách. Danh sách này có thể được thay đổi chứa các từ được phân biệt với nhau bởi dấu cách, hoặc nó có thể là một danh sách các giá trị được gõ trực tiếp vào trong câu lệnh. Mỗi lần qua vòng lặp, biến var1 được gán cho phần tử hiện tại trong danh sách và tiếp tục cho đến khi phần tử cuối cùng trong danh sách.

Dạng thứ hai của lệnh for có cú pháp như sau:

```
for var1
do
    statements
done
```

Trong dạng này, lệnh for thực thi một lần cho mỗi phần tử nằm trong biến var1. Khi bạn sử dụng cú pháp này của lệnh for, chương trình shell giả sử rằng biến var1 chứa tất cả các đối số được đưa vào trong chương trình shell từ dòng lệnh. Điển hình, dạng này của lệnh for là tương đương với viết các lệnh sau:

```
for var1 in "$@"
do
    statements
done
```

Tương đương với lệnh for trong tcsh là lệnh foreach. Nó xử lý tương tự như lệnh for trong bash và pdksh. Cú pháp của lệnh foreach như sau:

```
foreach name (list)
    commands
end
```

Một lần nữa, nếu mã này là một chương trình hoàn thiện, nó nên bắt đầu với kí hiệu # (và tốt nhất là #!/bin/sh để buộc thực thi theo Bourne shell). Dưới đây là một ví dụ về sử dụng lệnh for trong bash hay pdksh. Ví dụ này lấy các tùy chọn dòng lệnh số lượng bất kỳ các file text. Chương trình đọc mỗi file trong các file này, chuyển đổi tất cả các ký tự thành chữ hoa, và sau đó lưu trữ kết quả trong một file có cùng tên nhưng có phần mở rộng là .caps.

```
for file
do
tr a-z A-Z < $file >$file.caps
done
```

Chương trình sau là một ví dụ tương tự được viết theo ngôn ngữ shell tcsh:

```
#
foreach file ($*)
tr a-z A-Z < $file >$file.caps
end
```

### 6.6.2. Lệnh while

Một lệnh lặp khác được đưa vào ngôn ngữ lập trình shell là lệnh while. Lệnh này thực thi một khối các câu lệnh theo một điều kiện nào đó. Cú pháp của lệnh while trong bash và pdksh là như sau:

```
while expression
do
statements
done
```

Cú pháp cho lệnh while trong tcsh là như sau:

```
while (expression)
statements
end
```

Dưới đây là một ví dụ về lệnh while theo ngôn ngữ shell bash hay pdksh. Chương trình này đưa ra danh sách các đối số được đưa vào chương trình cùng với số các đối số.

```
count=1
while [ -n "$*" ]
do
    echo "This is parameter number $count $1"
    shift
    count=`expr $count + 1`
done
```

Lệnh shift chuyển đổi số dòng lệnh lên một sang bên trái (xem đoạn sau "Lệnh shift" để biết thêm thông tin). Chương trình bên dưới tương tự được viết cho ngôn ngữ tcsh:

```
#
set count = 1
while ( "$*" != "" )
    echo "This is parameter number $count $1"
    shift
    set count = `expr $count + 1`
end
```

### 6.6.3. Lệnh until

Lệnh until có cú pháp và chức năng tương tự lệnh while. Chỉ có sự khác biệt thực sự giữa hai lệnh là lệnh until thực thi mã trong khối của nó khi giá trị của biểu thức là sai và lệnh while thực thi các khối lệnh của nó nếu biểu thức có giá trị là true. Cú pháp cho lệnh until trong bash và pdksh là như sau:

```
until expression
do
    commands
done
```

Để làm cho ví dụ được sử dụng với lệnh while làm việc với lệnh until, tất cả những gì bạn phải làm chỉ là phủ định điều kiện, như chỉ ra trong đoạn mã bên dưới:

```
count=1
until [ -z "$*" ]
```



```
do
    echo "This is parameter number $count $1"
    shift
    count=`expr $count + 1`
done
```

Chỉ có sự khác nhau trong ví dụ này là và ví dụ về lệnh while là tùy chọn -n của lệnh test, nó có nghĩa rằng xâu không có độ dài bằng 0, được thay bởi tùy chọn -z, nó có nghĩa là chuỗi có độ dài bằng 0. Trong thực tế, lệnh until ít được dùng bởi vì với bất kỳ lệnh until nào, bạn cũng có thể viết được bằng lệnh while. Lệnh until không được hỗ trợ trong tcsh.

#### 6.6.4. Lệnh shift

Tất cả các shell bash, pdksh, và tcsh đều hỗ trợ một lệnh gọi là lệnh shift. Lệnh shift chuyển các giá trị hiện tại được lưu trữ trong các đối số dòng lệnh lên một vị trí sang trái. Ví dụ, nếu các giá trị của các đối số là

```
$1 = -r $2 = file1 $3 = file2
```

và bạn thực hiện lệnh shift

```
shift
```

kết quả các đối số được đưa vào như sau:

```
$1 = file1 $2 = file2
```

Bạn có thể dịch chuyển các đối số qua nhiều hơn một vị trí bởi một số xác định với kèm theo với lệnh shift. Lệnh sau dịch chuyển đối số lên hai vị trí:

```
shift 2
```

Lệnh này rất hữu ích khi có một chương trình shell cần phân tích các tùy chọn dòng lệnh. Các tùy chọn thường được đặt trước bởi một dấu nối và một ký tự để chỉ ra tùy chọn nào được sử dụng. Bởi vì các tùy chọn luôn luôn được xử lý trong một vòng lặp của một loại câu lệnh, bạn sẽ thường muốn nhảy đến đối số tiếp theo một khi bạn đã xác định được tùy chọn nào nên được xử lý tiếp theo. Ví dụ, chương trình shell sau chờ hai tùy chọn dòng lệnh, một xác định một file đầu vào và một xác định một file đầu ra. Chương trình đọc file đầu vào, chuyển tất cả các ký tự trong file input thành chữ hoa, và sau đó lưu trữ kết quả trong file đầu ra xác định:

```
while [ "$1" ]
```

```
do
```

```
if [ "$1" = "-i" ] then
infile="$2"
shift 2
else if [ "$1" = "-o" ] then
outfile="$2"
shift 2
else
echo "Program $0 does not recognize option $1"
fi
done
tr a-z A-Z <$infile >$outfile
```

### 6.6.5. Lệnh select

Shell pdksh đưa ra một lệnh lặp mà bash và tcsh không hỗ trợ, lệnh select. Nó hơi khác với các lệnh lặp khác bởi vì nó không thực thi một khối mã lệnh shell theo một điều kiện true hoặc false. Nó hững gì lệnh select làm là cho phép bạn tự động tạo các menu text đơn giản. Cú pháp của lệnh select như sau:

```
select menuitem [in list_of_items]
do
    commands
done
```

Khi bạn thực thi lệnh select, pdksh tạo một đối tượng menu được đánh số cho mỗi phần tử có trong list\_of\_items. list\_of\_items này có thể là một biến chứa nhiều hơn một phần tử, chẳng hạn như choice1 choice2 hoặc nó có thể là một danh sách các lựa chọn được gõ vào từ dòng lệnh, như trong ví dụ sau:

```
select menuitem in choice1 choice2 choice3
```

Nếu danh sách list\_of\_items is không được cung cấp, lệnh select sử dụng các đối số dòng lệnh cho lệnh thực hiện.

Khi người sử dụng của chương trình có chứa lệnh select chọn một trong số các phần tử của menu bằng cách gõ vào số tương ứng với nó, lệnh select lưu giá trị của phần tử được lựa chọn trong biến menuitem. Các lệnh trong khối do sau đó có thể thực hiện các hoạt động trên phần tử menu này.

Dưới đây là một ví dụ về việc sử dụng lệnh `select` như thế nào. Ví dụ này hiển thị ba phần tử của menu. Khi người sử dụng chọn một phần tử, chương trình sẽ hỏi bạn xem có phải phần tử đó được lựa chọn không, nếu người sử dụng gõ khác với `y` hoặc `Y`, chương trình sẽ hiển thị lại menu.

```
select menuitem in pick1 pick2 pick3
do
    echo "Are you sure you want to pick $menuitem"
    read res
    if [ $res = "y" -o $res = "Y" ]
    then
        break
    fi
done
```

Ví dụ này giới thiệu một vài lệnh mới. Lệnh `read` được sử dụng để lấy dữ liệu vào từ người sử dụng. Nó lưu bất kỳ cái gì người sử dụng gõ vào biến xác định. Lệnh `break` để kết thúc vòng lặp lệnh `while`, `select`, hoặc `for`.

#### 6.6.6. Lệnh `repeat`

Shell `tsh` có một lệnh lặp không có trong `pdksh` hay `bash`. Lệnh này là lệnh `repeat`. Lệnh `repeat` thực thi câu lệnh đơn theo một số lần xác định. Cú pháp cho lệnh `repeat` là như sau:

```
repeat count command
```

Ví dụ sau của lệnh `repeat` lấy một tập hợp các số là các tùy chọn dòng lệnh và in ra số các dấu chấm lên màn hình. Chương trình này hoạt động như một chương trình minh họa rất thô sơ.

```
#
foreach num ($*)
    repeat $num echo -n "."
    echo ""
end
```

---

Bạn có thể viết lại lệnh `repeat` bất kỳ bằng lệnh `while` hay lệnh `for`; cú pháp `repeat` chỉ thuận tiện hơn mà thôi.

## 6.7. Sử dụng các hàm

Ngôn ngữ shell cho phép bạn định nghĩa hàm của chính bạn. Các hàm này được định nghĩa giống như cách bạn định nghĩa các hàm trên ngôn ngữ lập trình C hay các ngôn ngữ lập trình khác. Thuận lợi chính của việc sử dụng hàm để tổ chức, tránh viết tất cả các mã shell của bạn trong một dòng. Mã được viết sử dụng các hàm có khuynh hướng dễ hơn trong việc đọc và bảo trì và cũng là khuynh hướng nhỏ gọn hơn bởi vì bạn có thể nhóm các mã chung vào trong một hàm thay vì việc đưa nó vào tất cả các nơi cần nó.

Cú pháp để tạo một hàm trong bash và pdksh là như sau:

```
fname () {  
    shell commands  
}
```

Cùng với cú pháp trước, pdksh cho phép cú pháp sau:

```
function fname {  
    shell commands  
}
```

Cả hai dạng này đều được xử lý chính xác như nhau theo cùng một cách.

Sau khi bạn đã định nghĩa hàm của bạn sử dụng một trong các dạng trên, bạn có thể gọi đến nó bằng cách vào lệnh sau:

```
fname [parm1 parm2 parm3 ...]
```

Chú ý rằng bạn có thể đưa số lượng bất kỳ các đối số vào trong hàm của bạn. Khi bạn đưa các đối số vào trong một hàm, nó xem các đối số này như đối số của một chương trình shell khi bạn đưa các đối số này từ dòng lệnh. Ví dụ, chương trình shell sau chứa vài hàm, mỗi hàm thực hiện một nhiệm vụ mà được kết hợp với các tùy chọn dòng lệnh. Ví dụ này bao trùm nhiều nội dung trong phần này. Ở đó đọc tất cả các file được đưa vào từ dòng lệnh và phụ thuộc vào tùy chọn được sử dụng, viết ra file với tất cả các ký tự hoa, viết ra file với tất cả các ký tự thường, hoặc in các file.

```
upper () {  
    shift  
    for i  
    do
```

```
tr a-z A-Z <$1 >$1.out
rm $1
mv $1.out $1
shift
done; }
lower () {
shift
for i
do
tr A-Z a-z <$1 >$1.out
rm $1
mv $1.out $1
shift
done; }
print () {
shift
for i
do
lpr $1
shift
done; }
usage_error () {
echo "$1 syntax is $1 <option> <input files>"
echo ""
echo "where option is one of the following"
echo "p -- to print frame files"
echo "u -- to save as uppercase"
echo "l -- to save as lowercase"; }
case $1
in
```

```
p | -p) print $@;;  
u | -u) upper $@;;  
l | -l) lower $@;;  
*) usage_error $0;;  
esac
```

Chương trình tcsh không hỗ trợ các hàm.

## 6.8. Tổng kết

Trong chương này, bạn đã thấy được nhiều đặc điểm của các ngôn ngữ lập trình bash, pdksh và tcsh. Khi bạn sử dụng Linux, bạn sẽ thấy rằng bạn sử dụng các ngôn ngữ lập trình shell càng ngày càng thường xuyên. Cho dù ngôn ngữ shell rất mạnh và dễ học, bạn có thể gặp phải một vài vấn đề khi chương trình shell không phù hợp với vấn đề bạn giải quyết. Trong những trường hợp như vậy, bạn có thể nghiên cứu tìm hiểu các ngôn ngữ khác có thể sử dụng có trong Linux.

## 7. Cài đặt và quản trị WebServer

### 7.1. Hướng dẫn cài đặt trên môi trường Linux.

Cài đặt trên môi trường Linux hoàn toàn không khó như những gì chúng ta nghĩ khi mới tiếp xúc với hệ điều hành này. Quá trình cài đặt chỉ đơn giản, chúng ta thực hiện câu lệnh rpm với cú pháp sau:

```
rpm -[ivhqladefUV] [-force] [nodeps] [--oldpackage] package list
```

Đây là chương trình quản lý các gói cài. Nó cho phép bạn quản lý các gói RPM, thực hiện rất dễ dàng việc cài đặt và gỡ bỏ phần mềm. Để cài đặt phần mềm có tên là precious-software-1.0.i386.rpm chạy câu lệnh sau:

```
rpm -i precious-software-1.0.i386.rpm
```

bạn có thể làm cho việc cài đặt trông đẹp mắt hơn bằng cách sử dụng tùy chọn -ivh thay cho tùy chọn -i. Nếu bạn đã cài một gói phần mềm rồi nhưng vì một lý do nào đó bạn lại muốn cài lại nó đè lên phiên bản cũ, bạn chỉ cần sử dụng tùy chọn -force cho lệnh rpm. Nếu bạn muốn nâng cấp một phần mềm, bạn sử dụng tùy chọn -U. Ví dụ:

```
rpm -Uvh precious-software-1.0.i386.rpm
```

Tuy nhiên bạn đã cài một phiên bản mới và bây giờ bạn muốn cài lại phiên bản cũ, nếu bạn muốn sử dụng lệnh trên, hệ thống sẽ báo lỗi phiên bản đã cài đặt là phiên bản mới hơn phiên bản mà bạn muốn cài. Để có thể thực hiện được điều này bạn sử dụng

tùy chọn `--oldpackage` cùng với tùy chọn `-U` để cài đặt phiên bản cũ. Để tìm kiếm các gói cài đã được cài vào hệ thống của bạn, bạn sử dụng lệnh sau:

```
rpm -qa
```

Để tìm các gói cài của một chương trình như `sendmail`, bạn có thể sử dụng lệnh

```
rpm -q sendmail
```

Hệ thống sẽ trả lại gói cài đã sử dụng để cài `sendmail`. Để phát hiện gói cài nào của một file xác định như `/bin/tcsh`, ta sử dụng câu lệnh:

```
rpm -qf /bin/tcsh
```

Để đảm bảo rằng một gói được cài chưa được thay đổi theo bất cứ cách nào, bạn có thể sử dụng tùy chọn `-V`. Ví dụ để tất cả các file đã được cài ở trạng thái nguyên bản không bị thay đổi sử dụng lệnh

```
rpm -Va
```

Tùy chọn này trở lên rất hữu ích nếu bạn nhận thức được rằng một hay nhiều gói cài có thể bị phá hủy bởi người khác.

Để gỡ các gói cài khỏi hệ thống bạn sử dụng lệnh `rpm` với tùy chọn `-e`

```
rpm -e sendmail
```

Nếu bạn thấy rằng việc gỡ bỏ gói cài có thể bị dừng bởi các chương trình khác bởi vì chúng phụ thuộc vào nó hay các file của nó, bạn phải quyết định xem bạn có tiếp tục bỏ gói cài hay chương trình này hay không, nếu bạn muốn gỡ bỏ bạn có thể sử dụng tùy chọn `--nodeps` cùng với tùy chọn `-e` để ép buộc `rpm` gỡ bỏ gói cài đó.

## 7.2. Quản trị WebServer

### 7.2.1. Phần mềm Apache

Máy chủ web nghe yêu cầu từ phía client, như bộ trình duyệt `Internet Explorer` hoặc `Internet Explorer`. Khi nhận được yêu cầu máy chủ xử lý yêu cầu và trả dữ liệu lại cho máy client. Dữ liệu trả về máy trạm thường là các trang định dạng có chứa hình ảnh và text. Trình duyệt nhận dữ liệu và hiển thị trang dữ liệu cho người dùng. Khái niệm máy chủ web rất đơn giản, nó đợi yêu cầu, thực hiện, rồi trả lại cho người dùng.

Máy chủ web nói chuyện với các máy client và máy trạm thông qua giao thức HTTP (Hypertext Transfer Protocol). Điều này cho phép máy trạm kết nối tới nhiều nhà cung cấp dịch vụ web mà không gặp phải các vấn đề về tương thích.

Phần lớn các yêu cầu được định dạng dưới dạng trang HTML (Hypertext Markup Language). HTML cho phép liên kết nhiều văn bản và tài nguyên khác nhau. Siêu văn bản cho phép liên kết tới các trang văn bản khác trên cùng một máy tính hoặc trên các máy tính đặt trên khắp thế giới.

Apache được phát triển dựa trên mã nguồn mở của CSA web server, là phiên bản cung cấp đầy đủ các tính năng của máy chủ (HTTP) web do dự án Apache Server thực hiện. Apache cung cấp một máy chủ web mã nguồn mở, tin cậy, hiệu quả và dễ dàng mở rộng. Phần mềm máy chủ bao gồm: daemon server, file cấu hình, công cụ quản trị, và tài liệu.

Phần mềm Apache Server sẵn có có trên trang Apache Group. Bạn có thể tải về từ các địa chỉ <http://www.apache.org/dist/>. Bạn tải về file `.tar.gz` tương ứng với phiên bản bạn muốn sử dụng. Ví dụ, Phiên bản mới nhất được viết là Apache 1.3.12, vì vậy file bạn cần tải về là `apache_1.3.12.tar.gz` Bạn có thể lấy mã nguồn từ địa chỉ [http://www.apache.org/dist/apache\\_1.3.12.tar.gz](http://www.apache.org/dist/apache_1.3.12.tar.gz).

### Giải nén file

Để giải nén file này, sử dụng câu lệnh sau (giả sử rằng bạn đã để file trong thư mục `temp`):

```
cd temp
gzip -d -c apache_1.3.12.tar.gz | tar xvf -
```

Câu lệnh này tạo một thư mục `apache_1.3.12` trong thư mục `temp`

### 7.2.2. Biên dịch và cài đặt

Chạy các câu lệnh sau:

```
cd apache_1.3.12
./configure --prefix=<path-to-apache>
make
make install
```

Chú ý sử dụng đường dẫn đầy đủ thay cho `<path-to-apache>`. Đường dẫn đầy đủ này nên là nơi bạn muốn cài đặt apache server, chẳng hạn như

```
./configure --prefix=/afs/uncc.edu/usr/q/zlian/apache
```

### 7.2.3. Khởi động và tắt WebServer

#### Khởi động Apache

```
<path-to-apache>/bin/apachectl start
```

Ví dụ:

```
/afs/uncc.edu/usr/q/zlian/Apache/bin/apachectl start
```

#### Tắt Apache



```
<path-to-apache>/bin/apachectl stop
```

Ví dụ:

```
/afs/uncc.edu/usr/q/zlian/Apache/bin/apachectl stop
```

#### 7.2.4. Cấu hình Apache

Theo cách truyền thống, cấu hình Apache được chia thành ba file cấu hình: `httpd.conf`, `access.conf`, và `srm.conf`. Theo thứ tự các file này có ý nghĩa như sau, `httpd.conf` là file cấu hình server chính, `access.conf` là file định nghĩa các quyền truy cập, và `srm.conf` các tài nguyên server được định nghĩa, chẳng hạn như ánh xạ các thư mục và các biểu tượng. Trong 1.3.4, ba file này được trộn vào một file chung `httpd.conf`, nó có thể tìm thấy trong thư mục `conf`. Ví dụ:

```
/afs/uncc.edu/usr/q/zlian/apache/conf/
```

**Chú ý: Các hình ảnh quan trọng cho cấu hình của bạn:**

- **ServerName**

ServerName chỉ ra địa chỉ IP của máy chủ cài đặt dịch vụ WebServer, thông thường nếu máy của bạn là máy cục bộ, không nối mạng, địa chỉ này mặc định là 127.0.0.1 tương ứng với tên máy là localhost. Nếu máy này có địa chỉ mạng, bạn có thể thay thế bằng địa chỉ IP của máy. Để xem địa chỉ của máy bạn thực hiện lệnh:

```
ifconfig -a
```

- **Listen**

Chỉ dẫn này nói cho server lắng nghe các yêu cầu trên địa chỉ IP được xác định và/hoặc cổng TCP/IP. Mặc định, server lắng nghe cổng 80, nhưng bạn nên sử dụng cổng lớn hơn 1024, bởi vì số ít hơn 1024 rất hay được sử dụng trong các tiến trình của hệ thống. Ảnh hưởng ví dụ sau, Apache nghe trên cả hai cổng port 8080 and 8081.

```
listen 8080
listen 8081
```

Với cấu hình này, bạn có thể kiểm tra xem server của bạn chạy thành công hay chưa bằng cách gõ vào địa chỉ sau trên trình duyệt:

<http://localhost:8080>

hoặc

<http://localhost:8081>

- **DocumentRoot**

Thư mục tài liệu mặc định là `<path-to-apache>/htdocs`, bạn có thể để tài liệu html, ví dụ `billchu.html`, trong thư mục này và kiểm tra. Ví dụ:

```
http://152.15.35.2:8080/billchu.html
```

Bạn cũng có thể thay đổi thư mục tài liệu bằng sử dụng hướng dẫn sau trong file `httpd.conf`:

```
DocumentRoot /usr/web
```

Sau đó một truy cập đến <http://www.my.host.com/index.html> sẽ tương ứng `/usr/web/index.html`.

Thường xuất hiện trong khi cấu hình như sau: (i.e., "`DocumentRoot /usr/web/`") thêm một ký tự `"/`" ở cuối, bạn nên tránh điều này.

### 7.2.5. Xác thực người dùng

Để ngăn chặn truy cập vào các file trên server của bạn, bạn nên sử dụng bảo vệ user/password, Bạn có thể sử dụng các hướng dẫn sau.

```
AuthType
AuthName
AuthUserFile
AuthGroupFile
require
<Directory></Directory>
<Files></Files>
```

`AuthType` Lựa chọn kiểu xác thực người sử dụng cho một thư mục. Chỉ có `Basic` và `Digest` là thực thi hiện tại.

`AuthName` Đặt tên của xác thực cho một thư. Tên xác thực này sẽ được gửi đến client để những người sử dụng biết loại username và password nào để gửi. `AuthName` có một đối số; ở đầu tên xác thực có dấu cách nó phải được đặt trong dấu trích dẫn.

`AuthUserFile` Đặt tên của file văn bản thuần túy chứa danh sách những người sử dụng và mật khẩu cho việc xác thực người sử dụng. Tên file là đường dẫn đến file người sử dụng. ở đầu nó không phải là đường dẫn tuyệt đối (ví dụ, nếu nó không bắt đầu với `'/`), ở đó được xem như đường dẫn tương đối đến `ServerRoot`.

`AuthGroupFile` Đặt tên của một file văn bản thuần túy chứa danh sách các nhóm người sử dụng cho việc xác thực người sử dụng. Tên file là đường dẫn đến file group. ở đầu nó không phải là đường dẫn tuyệt đối (ví dụ, không bắt đầu với dấu `'/`), nó được xem như đường dẫn tương đối đến `ServerRoot`.

`require` Chọn những người sử dụng nào có thể truy cập vào một thư mục. Cú pháp cho phép là:

1. Chỉ những người sử dụng được đặt tên có thể truy cập thư mục:  

```
require user userid userid ...
```
2. Chỉ những người sử dụng trong các nhóm được đặt tên có thể truy cập thư mục:  

```
require group group-name group-name ...
```
3. Tất cả những người sử dụng có thể truy cập thư mục:  

```
require valid-user
```

`<Directory>` và `</Directory>` được sử dụng để nhóm một nhóm các hướng dẫn và nó sẽ chỉ được áp dụng cho thư mục được đặt tên và các thư mục con

của thư mục đó. Một hướng dẫn bất kỳ được cho phép có trong một directory có thể được sử dụng.

`<Files>` và `</Files>` cung cấp quyền truy cập bởi tên file (bao gồm đường dẫn đến file).

**Ví dụ:**

```
<Directory
"/afs/uncc.edu/usr/q/zlian/apache/htdocs/manual">
  AuthType Basic
  AuthName "Restricted Directory"
  AuthUserFile passwd
  AuthGroupFile /dev/null
  require valid-user
</Directory>
```

Để thiết lập file password, bạn có thể sử dụng công cụ có tên là `htpasswd` được cung cấp bởi Apache. Trước tiên tạo file password bằng cách:

```
% touch passwd
```

Trong thư mục "`<path-to-apache>/bin/`". Để thêm một người sử dụng, thực hiện lệnh:

```
% htpasswd <path-to-password-file>/passwd zlian
New password:
Re-type new password:
```

Đến đây bạn đã hoàn thành xong việc cấu hình Apache và thực hiện xác thực người sử dụng cho dịch vụ web của bạn.

## 8. Quản trị các tiến trình

### 8.1. Tiến Trình

#### 8.1.1. Tiến trình tiền cảnh

Khi bạn đang trên dấu nhắc hệ thống (`#` hoặc `$`) và gọi một chương trình, chương trình trở thành một tiến trình và đi vào hoạt động dưới sự kiểm soát của hệ thống. Dấu nhắc của hệ thống sẽ không xuất hiện khi tiến trình đang chạy. Khi tiến trình hoàn thành tác vụ và chấm dứt, hệ điều hành sẽ trả lại dấu nhắc để bạn gõ tiếp lệnh thực thi chương trình khác. Chương trình hoạt động theo cách này được gọi là chương trình tiền cảnh (foreground). Ví dụ khi bạn thực hiện lệnh:

```
ls -R /
```

Bạn sẽ phải chờ đợi rất lâu cho đến khi lệnh thực hiện xong bạn mới có thể nhập vào lệnh mới để thực hiện công việc tiếp theo của bạn.

#### 8.1.2. Tiến trình hậu cảnh

Chúng ta có cách nào đó yêu cầu Linux đưa các tiến trình chiếm nhiều thời gian xử lý hoặc ít tương tác với người dùng ra hoạt động phía hậu cảnh (background) trả lại ngay dấu nhắc để có thể thực hiện các tiến trình ở tiền cảnh thì tốt hơn. Điều này có thể thực hiện được bằng cách kết hợp chỉ thị & với lệnh gọi chương trình mà ta sẽ tìm hiểu ở phần sau, khi đó tiến trình sẽ hoạt động ở phía hậu cảnh và trả lại ngay dấu nhắc cho chúng ta làm công việc khác. Các tiến trình như vậy gọi là các tiến trình hậu cảnh. Việc chạy tiến trình ở hậu cảnh rất thuận tiện, chúng cho phép nhiều chương trình tương tác với nhau.

## 8.2. Điều khiển và giám sát các tiến trình

Ở hư đề cập trước đây, các tiến trình thường trực thường được bắt đầu bằng tiến trình init khi khởi động. Bạn có thể điều khiển tiến trình nào chạy ngay khi khởi động bằng cách cấu hình lại các file cấu hình và kịch bản của init. Ở ngoài trừ các tiến trình thường trực, các loại tiến trình khác mà bạn sẽ chạy được gọi là các tiến trình của người sử dụng hay các tiến trình tương tác. Bạn phải chạy một tiến trình tương tác thông qua một shell. Mỗi một shell chuẩn cung cấp một dòng lệnh khi người sử dụng vào tên của một chương trình. Khi người sử dụng vào tên chương trình hợp lệ trên dòng lệnh, shell sẽ tự tạo một bản copy như một tiến trình mới và thay thế tiến trình mới với chương trình được đặt tên trên dòng lệnh. Ở ói một cách khác shell sẽ chạy chương trình được đặt tên như một tiến trình khác. Để lấy thông tin về tất cả các tiến trình đang chạy trên hệ thống của bạn, bạn cần chạy tiện ích có tên là ps

### 8.2.1 Sử dụng lệnh ps để lấy thông tin trạng thái của tiến trình

Tiện ích này tạo ra một báo cáo về tất cả các tiến trình trên hệ thống của bạn. ví dụ, nếu bạn chạy lệnh ps, nó sẽ hiển thị kết quả như sau:

```
PID  TTY    TIME  CMD
13636 pts/1  00:00:00 bash
13696 pts/1  00:00:00 man
13699 pts/1  00:00:00 sh
13700 pts/1  00:00:00 sh
13704 pts/1  00:00:00 less
16692 pts/1  00:00:00 tail
17252 pts/1  00:00:00 ps
```

Dưới đây là giải thích về ý nghĩa của các trường

Trường	Giải Thích
USER hoặc UID	Tên của tiến trình

PID	ID (định danh) của tiến trình
%CPU	% CPU sử dụng của tiến trình
%MEM	% bộ nhớ tiến trình sử dụng
SIZE	Kích thước bộ nhớ ảo tiến trình sử dụng
RSS	Kích thước của bộ nhớ thực sử dụng bởi tiến trình
TTY	Vùng làm việc của tiến trình
STAT	Trạng thái của tiến trình
START	Thời gian hay ngày bắt đầu của tiến trình
TIME	Tổng thời gian sử dụng CPU
COMMAND	Câu lệnh được thực hiện
PRI	Mức ưu tiên của tiến trình
PPID	ID của tiến trình cha
WCHAN	Tên của hàm nhân khi tiến trình ngủ được lấy từ file /boot/System.map
FLAGS	Số cờ được kết hợp với tiến trình

Tiện ích ps cũng tiếp nhận một vài đối số từ dòng lệnh. Bảng bên dưới chỉ ra các tùy chọn được sử dụng chung:

Tùy Chọn	Miêu tả
A	Hiển thị các tiến trình của tất cả những người sử dụng
E	Hiển thị các biến môi trường của tiến trình sau khi dòng lệnh được thực thi
L	Hiển thị kết quả đầy đủ
U	Hiển thị tên người sử dụng và thời gian bắt đầu tiến trình
W	Hiển thị kết quả theo định dạng rộng. Bình thường, kết quả kết xuất bị cắt nếu nó không vừa một dòng. Sử dụng tùy chọn này bạn có thể ngăn chặn được điều đó
Txx	Hiển thị các tiến trình được kết hợp với vùng làm việc xx

X	Hiển thị các tiến trình không có điều khiển vùng làm việc
---	---

Ví dụ để hiển thị tất cả các tiến trình bạn thực hiện câu lệnh:

```
ps au
```

Để hiển thị tất cả các tiến trình của một người nào đó sử dụng:

```
ps au | grep username
```

Tuy nhiên, nếu bạn chỉ muốn tìm các tiến trình đang tồn tại với người sử dụng bất kỳ, bạn sử dụng câu lệnh:

```
ps aux
```

Để tìm kiếm PID của một tiến trình cha sử dụng:

```
ps l pid
```

Với *pid* là PID của một tiến trình nào đó.

```
ps e
```

Thông tin biến môi trường được bổ sung vào trường COMMAÂ D

### 8.2.2. Phát tín hiệu cho một chương trình đang chạy

- **Sử dụng lệnh kill hủy một tiến trình**

Câu lệnh kill là một kịch bản shell được xây dựng sẵn, thường được tìm thấy trong thư mục /bin. Bạn có thể dùng lệnh này để dừng một tiến trình nào đó. bạn có thể chạy:

```
kill PID
```

Với *PID* là PID của tiến trình nào đó

- **Sử dụng lệnh killall hủy một tiến trình**

Tiện ích này cho phép bạn dừng một tiến trình bằng tên. Ví dụ bạn có một tiến trình được gọi là `signal_demo.pl` và bạn muốn dừng tiến trình này. Bạn sử dụng lệnh:

```
killall signal_demo.pl
```

- **Chạy một tiến trình ở hậu cảnh hoặc tiền cảnh**

Thông thường khi chúng ta chạy một tiến trình từ thiết bị đầu cuối (bàn phím) hay shell, bạn chạy tiến trình ở tiền cảnh. Khi bạn chạy tiến trình ở tiền cảnh, bạn phải đợi cho nó kết thúc. Tuy nhiên, thay vì việc đợi cho nó kết thúc, bạn có thể chạy nó ở hậu cảnh bằng việc thêm một ký hiệu '&' ở cuối dòng lệnh. Điều này hữu ích khi một tiến trình chạy trong thời gian dài và bạn cần phải làm một công việc khác. Ví dụ, để khởi động hệ quản trị CSDL PostgreSQL với postmaster bạn thực hiện:

```
postmaster -i &
```

Vậy khi nào bạn biết một tiến trình hậu cảnh đang chạy hay đã dừng. Bạn có thể sử dụng lệnh:

```
ps -af
```

để xem tất cả các tiến trình trong đó có cả tiến trình ở hậu cảnh.

- **Tạm dừng tiến trình**

Để tạm dừng một tiến trình đang chạy ở tiền cảnh và bạn muốn đưa chúng vào hậu cảnh, bạn thực hiện công việc này bằng cách nhấn tổ hợp phím Ctrl + Z. Khi nhận được tín hiệu Ctrl+Z tiến trình sẽ bị tạm dừng và được đưa vào hậu cảnh. Tuy nhiên bạn chưa biết được chương trình của chúng ta đã dừng chưa và đã chuyển vào hậu cảnh chưa. Lệnh `jobs` hiển thị trạng thái của tất cả các tiến trình đang chạy ở hậu cảnh:

```
[1] Stopped          man ln (wd: /home/trantu/exam)
[2]- Stopped        tail
[3]+ Stopped        ls -R /
```

- **Đánh thức tiến trình**

Để đánh thức một tiến trình ta sử dụng lệnh `bg` kết hợp với số tác vụ trong hàng đợi liệt kê. Trong ví dụ ở trên ta có thể thực hiện lệnh:

*bg 3*

Một lần nữa ta sử dụng lệnh `jobs`, ta sẽ thấy thông tin hiện trên màn hình như sau:

```
[1] Stopped          man ln (wd: /home/trantu/exam)
[2]- Stopped        tail
[3]+ Running        ls -R /
```

Để chuyển một tiến trình từ hậu cảnh sang chạy trên tiền cảnh bạn dùng lệnh `fg`. Ví dụ:

*fg 3*

### 8.2.3. Giao tiếp giữa các tiến trình

Đôi khi các tiến trình cần trao đổi thông tin cho nhau để xử lý. Chẳng hạn như lệnh `ls` của Linux chỉ biết liệt kê và ghi toàn bộ dữ liệu về thông tin của file, thư mục ra màn hình. Lệnh `ls` không có cơ chế dừng khi màn hình đầy. Trong khi lệnh `more` lại có khả năng đọc dữ liệu và đưa ra màn hình theo từng trang để người dùng có thời gian xem qua. Các chương trình cần có nhu cầu chuyển dữ liệu cho nhau xử lý. Một cơ chế được sử dụng khá phổ biến trên Linux là pipe (đường ống). Bạn sử dụng chỉ thị `|` để biểu thị đường ống. Ví dụ:

```
ls -R | more
```

Hoặc bạn có thể tìm chính xác tên tiến trình như:

```
ps -af | grep '[bash]'
```

## 8.3 Lập kế hoạch các tiến trình

### 8.3.1 Sử dụng lệnh `at`

Tiện ích `at` cho phép bạn sắp xếp một câu lệnh để thực thi trong thời gian sau đó. Ví dụ, để xem dung lượng đĩa sử dụng cho toàn bộ các file, thư mục của hệ thống bạn gọi tiện ích `du` vào lúc 8:40 p.m, bạn có thể chạy lệnh sau:

```
at 20:40
```

Câu lệnh sẽ hiển thị dấu nhắc “`at>`” yêu cầu bạn nhập vào câu lệnh để thực hiện theo thời gian đã được đưa vào. Bạn gõ vào dòng lệnh:



```
du -a > /tmp/du.out
```

Sau khi bạn gõ lệnh Enter, nó sẽ hiển thị lại dấu nhắc cho phép bạn nhập vào các câu lệnh tiếp theo. Bạn có thể chọn Ctrl+D để kết thúc.

Ấu ều vì một lý do nào đó, bạn muốn dừng công việc mà bạn đã lập lịch, bạn có thể sử dụng lệnh `atrm` để xóa công việc đó trước khi nó được thực hiện. Bạn cần phải biết số thứ tự của công việc mà bạn muốn hủy, để tìm ra các công việc mà bạn đã lập lịch, bạn chạy câu lệnh `atq` để tìm số thứ tự công việc, sau đó dùng `atrq` với đối số là số thứ tự của công việc muốn hủy. Ví dụ:

```
atrq 1
```

### 8.3.2 Sử dụng crontab

Có nhiều công việc trên Linux cần được lập lịch một cách thường xuyên, ví dụ để xóa các file cũ được sinh ra bởi hệ thống trong thư mục `tmp` hàng ngày, hay hàng tuần bạn cần phải chạy một tiến trình mỗi ngày hay mỗi tuần. Tiện ích `cron` cho phép bạn thực hiện các công việc như thế. Thực ra `cron` bao gồm `crond` daemon, được khởi động bởi tiến trình `init`. `Crond` đọc các lịch công việc từ `/etc/crontab` và các file trong `/var/spoon/cron`. Thư mục `cron` này lưu trữ các file lập lịch (thường được gọi là `crontab` hay `cron table`) cho những người sử dụng thông thường được phép chạy các công việc `cron`. Là một `superuser`, bạn có thể xác định một danh sách những người sử dụng được phép chạy các công việc `cron` trong file `/etc/cron.allow`. Tương tự, bạn có thể xác định những người sử dụng không được phép thực hiện các công việc `cron` trong file `/etc/cron.deny`. Cả hai file này đều sử dụng một định dạng cơ bản: một `username` trên một dòng. ấu ều một người được phép thực hiện các công việc `cron`, người đó có thể sử dụng tiện ích `crontab` để thực hiện công việc lập lịch. Ví dụ, khi bạn được phép, bạn có thể gõ lệnh:

```
crontab -e
```

và soạn thảo các công việc cần thực hiện. Một công việc `cron` phải có định dạng sau:

```
minute(s) hour(s) day(s) month weekday username command argument(s)
```

Các trường từ 1 đến 5 có định dạng sau

## 9. Bảo mật hệ thống

Cùng với sự phát triển không ngừng của truyền thông kỹ thuật số, Internet và sự phát triển nhảy vọt của nền công nghiệp phần mềm, bảo mật máy tính là một vấn đề ngày càng trở nên quan trọng. Cần phải hiểu rằng không có hệ thống máy tính nào là an

toàn tuyệt đối. Tất cả những gì bạn có thể làm là giúp cho hệ thống của bạn trở nên an toàn hơn.

Kể từ khi Linux được phát triển một cách rộng rãi và nhanh chóng, đặc biệt là trong các giao dịch kinh doanh quan trọng, an ninh là một vấn đề quyết định sự sống còn của Linux. Với hàng trăm công cụ bảo vệ sẵn có, người dùng Linux được trang bị tốt hơn để ngăn chặn và duy trì một hệ thống an toàn. Linux không những hoạt động tốt mà còn có những tính năng và sản phẩm liên quan cho phép xây dựng một môi trường tương đối an toàn.

### 9.1. Những nguy cơ an ninh trên Linux

Linux và các ứng dụng trên nó có thể không ít các lỗ hổng an ninh hơn những hệ điều hành khác. Theo quan điểm của một số chuyên gia máy tính, Linux có tính an toàn cao hơn các hệ điều hành của Microsoft, vì các sản phẩm của Microsoft không được xem xét kỹ lưỡng và chặt chẽ bằng các sản phẩm mã nguồn mở như Linux. Hơn nữa, Linux dường như là "miễn nhiễm" với virus máy tính (hiện tại đã có xuất hiện một vài loại virus hoạt động trên môi trường Linux nhưng không ảnh hưởng gì mấy đến người dùng Linux). Ắng hưng một hệ thống Linux được cấu hình không tốt sẽ tệ hơn nhiều so với một hệ thống Microsoft được cấu hình tốt !!! Khi có được một chính sách an ninh tốt và hệ thống được cấu hình theo đúng chính sách đó thì sẽ giúp bạn tạo được một hệ thống an toàn (ở mức mà chính sách của bạn đưa ra).

Ắng hưng sự an toàn không phải là thứ có thể đạt được như một mục tiêu cuối cùng. Đúg hơn đó là tập hợp của những cách cài đặt, vận hành và bảo trì một hệ điều hành, mạng máy tính, ... Ắng ó phụ thuộc vào các hoạt động hàng ngày của hệ thống, người dùng và người quản trị. Bạn phải bắt đầu từ một nền tảng ban đầu và từ đó cải thiện tính an toàn của hệ thống của bạn nhiều nhất có thể được mà vẫn đảm bảo các hoạt động bình thường của hệ thống.

### 9.2. Xem xét chính sách an ninh của bạn

Kết nối vào Inernet là nguy hiểm cho hệ thống mạng của bạn với mức an toàn thấp. Từ những vấn đề trong các dịch vụ TCP/IP truyền thống, tính phức tạp của việc cấu hình máy chủ, các lỗ hổng an ninh bên trong quá trình phát triển phần mềm và nhiều nhân tố khác góp phần làm cho những hệ thống máy chủ không được chuẩn bị chu đáo có thể bị xâm nhập và luôn tồn tại những nguy cơ tiềm tàng về vấn đề an toàn trong đó.

Mục đích của một chính sách an toàn hệ thống là quyết định một tổ chức sẽ phải làm như thế nào để bảo vệ chính nó. Để có được một chính sách an ninh hiệu quả, người xây dựng các chính sách này phải hiểu và có thể kết hợp tất cả các thông tin, yêu cầu, ...

Khi một tình huống xảy ra nằm ngoài dự kiến, chẳng hạn một sự xâm nhập trái phép vào hệ thống của bạn, câu hỏi lớn nhất là "sẽ phải làm gì đây ?"

Không may là có hàng triệu câu trả lời khác nhau cho câu hỏi đó. Ắng ếu một người mà chưa từng phải đối phó với một kẻ xâm nhập trước đây thì kẻ xâm nhập có thể dễ dàng biến mất vì các dấu vết đã trở nên quá cũ và không còn hữu ích nữa.

Ả hững sai sót trong chính sách an ninh không chỉ liên quan đến những kẻ xâm nhập, mà còn liên quan đến những vấn đề bình thường như thời tiết, thiên tai, cháy, nổ, hư hỏng thiết bị,... Do vậy, việc thiết lập một chính sách an ninh tốt cho việc giải quyết những sự cố phải được lên kế hoạch kỹ lưỡng, được xem xét và chứng nhận bởi người có quyền hạn trong công ty.

Một chính sách an ninh tốt nên bao gồm các vấn đề sau :

- Chính sách phục hồi dữ liệu khi có sự cố
- Chính sách phục hồi hệ thống trong trường hợp hư hỏng thiết bị
- Chính sách, cách thức điều tra những kẻ xâm nhập trái phép
- Chính sách, cách thức điều tra khi công ty bị cáo buộc xâm nhập vào các hệ thống khác
- Cách thức, quy trình và nơi thông báo sự xâm nhập trái phép từ bên ngoài hay gây ra bởi các nhân viên của mình.
- Chính sách an ninh về mặt vật lý của hệ thống

...

Bạn có thể nhờ tư vấn của các công ty, tổ chức làm dịch vụ tư vấn về an toàn máy tính để giúp bạn xây dựng một chính sách an ninh tốt. Các công ty này có các chuyên gia về an toàn máy tính, họ có sẵn các biểu mẫu chính sách an ninh nên có thể thiết lập nhanh chóng các chính sách mà bao gồm tất cả các mặt trong việc an toàn hệ thống máy tính.

### 9.3. Tăng cường an ninh cho KERNEL

Mặc dù thừa hưởng những đặc tính của hệ thống Uả IX và khá an ninh hơn một số hệ điều hành khác, hệ thống Gả U/Linux hiện nay vẫn tồn tại những nhược điểm sau:

- Quyền của user ‘root’ có thể bị lạm dụng. User ‘root’ có thể dễ dàng thay đổi bất kỳ điều gì trên hệ thống.
- ả hiệu file hệ thống có thể dễ dàng bị sửa đổi. ả hiệu file hệ thống quan trọng như /bin/login có thể bị sửa đổi bởi hacker để cho phép đăng nhập không cần mật khẩu. ả hung những file loại này lại hiếm khi nào thay đổi trừ phi khi nâng cấp hệ thống.
- Các module có thể được dùng để chặn kernel. “Loadable Kernel Module” là một thiết kế tốt để tăng cường tính uyển chuyển, linh hoạt cho kernel. ả hung sau khi một module được nạp vào kernel, nó sẽ trở thành một phần của kernel và có thể hoạt động như kernel nguyên thủy. Vì vậy, các chung trình mục đích xấu có thể được viết dạng module và nạp vào kernel, rồi sau đó hoạt động như một virus.
- Các process không được bảo vệ. Các process như web server có thể trở thành mục tiêu bị tấn công của hacker sau khi thâm nhập hệ thống.

Để cải thiện tính an ninh cho các server Linux, chúng ta cần có một kernel an toàn hơn. Điều này có thể thực hiện được bằng cách sửa đổi kernel nguyên thủy bằng các ‘patch’ tăng cường tính an ninh cho hệ thống. Các patch này có các tính năng chính yếu sau:

- Bảo vệ – bảo vệ các file hệ thống quan trọng khỏi sự thay đổi ngay cả với user root. Bảo vệ các process quan trọng khỏi bị ngừng bởi lệnh ‘kill’. Chặn các tác vụ truy cập IO mức thấp (RAW IO) của các chương trình không được phép.
- Phát hiện – Phát hiện và cảnh báo với người quản trị khi server bị scan. Cũng như khi có các tác vụ trên hệ thống vi phạm các luật (rules) định trước.
- Đối phó – Khi phát hiện sự vi phạm trên hệ thống, các ghi nhận chi tiết sẽ được thực hiện cũng như có thể ngừng lập tức phiên làm việc gây ra

Một vài công cụ sửa đổi kernel được sử dụng rộng rãi là LIDS (Linux Intrusion Detection System), Medusa, ...

#### 9.4. An toàn các giao dịch trên mạng

Có rất nhiều dịch vụ mạng truyền thông giao tiếp thông qua giao thức văn bản không mã hoá, như TELNET, FTP, RLOGIN, HTTP, POP3. Trong các giao dịch giữa người dùng với máy chủ, tất cả các thông tin dạng gói được truyền qua mạng dưới hình thức văn bản không được mã hoá. Các gói tin này có thể dễ dàng bị chặn và sao chép ở một điểm nào đó trên đường đi. Việc giải mã các gói tin này rất dễ dàng, cho phép lấy được các thông tin như tên người dùng, mật khẩu và các thông tin quan trọng khác. Việc sử dụng các giao dịch mạng được mã hoá khiến cho việc giải mã thông tin trở nên khó hơn và giúp bạn giữ an toàn các thông tin quan trọng. Các kỹ thuật thông dụng hiện nay là IPSec, SSL, TLS, SASL và PKI.

Quản trị từ xa là một tính năng hấp dẫn của các hệ thống UAX. ả gười quản trị mạng có thể dễ dàng truy nhập vào hệ thống từ bất kỳ nơi nào trên mạng thông qua các giao thức thông dụng như telnet, rlogin. Một số công cụ quản trị từ xa được sử dụng rộng rãi như linuxconf, webmin cũng dùng giao thức không mã hoá. Việc thay thế tất cả các dịch vụ mạng dùng giao thức không mã hoá bằng giao thức có mã hoá là rất khó. Tuy nhiên, bạn nên cung cấp việc truy cập các dịch vụ truyền thông như HTTP/POP3 thông qua SSL, cũng như thay thế các dịch vụ telnet, rlogin bằng SSH.

#### 9.5. Linux firewall

An toàn hệ thống luôn luôn là một vấn đề sống còn của mạng máy tính và firewall là một thành phần cốt yếu cho việc đảm bảo an ninh.

Một firewall là một tập hợp các qui tắc, ứng dụng và chính sách đảm bảo cho người dùng truy cập các dịch vụ mạng trong khi mạng bên trong vẫn an toàn đối với các kẻ tấn công từ Internet hay từ các mạng khác. Có hai loại kiến trúc firewall cơ bản là : Proxy/Application firewall và filtering gateway firewall. Hầu hết các hệ thống firewall hiện đại là loại lai (hybrid) của cả hai loại trên.

ả hiệu công ty và nhà cung cấp dịch vụ Internet sử dụng máy chủ Linux như một Internet gateway. ả hững máy chủ này thường phục vụ như máy chủ mail, web, ftp, hay dialup. Hơn nữa, chúng cũng thường hoạt động như các firewall, thi hành các chính sách kiểm soát giữa Internet và mạng của công ty. Khả năng uyển chuyển khiến cho Linux thu hút như là một thay thế cho những hệ điều hành thương mại.

Tính năng firewall chuẩn được cung cấp sẵn trong kernel của Linux được xây dựng từ hai thành phần : ipchains và IP Masquerading.

Linux IP Firewalling Chains là một cơ chế lọc gói tin IP. ả hững tính năng của IP Chains cho phép cấu hình máy chủ Linux như một filtering gateway/firewall dễ dàng. Một thành phần quan trọng khác của nó trong kernel là IP Masquerading, một tính năng chuyển đổi địa chỉ mạng (network address translation- ả AT) mà có thể che giấu các địa chỉ IP thực của mạng bên trong.

Để sử dụng ipchains, bạn cần thiết lập một tập các luật mà qui định các kết nối được cho phép hay bị cấm. Ví dụ:

```
# Cho phép các kết nối web tới Web Server của bạn
/sbin/ipchains -A your_chains_rules -s 0.0.0.0/0 www -d 192.16.0.100
1024: -j ACCEPT

# Cho phép các kết nối từ bên trong tới các Web Server bên ngoài
/sbin/ipchains -A your_chains_rules -s 192.168.0.0/24 1024: -d
0.0.0.0/0 www -j ACCEPT

# Từ chối truy cập tất cả các dịch vụ khác
/sbin/ipchains -P your_chains_rules input DENY
```

ả ngoài ra, bạn có thể dùng các sản phẩm firewall thương mại như Check Point FireWall-1, Phoenix Adaptive Firewall, Gateway Guardian, XSentry Firewall, Raptor, ... hay rất nhiều các phiên bản miễn phí, mã nguồn mở cho Linux như T.Rex Firewall, Dante, SIả US, TIS Firewall Toolkit, ...

## 9.6. Dùng công cụ dò tìm để khảo sát hệ thống

Thâm nhập vào một hệ thống bất kỳ nào cũng cần có sự chuẩn bị. Hacker phải xác định ra máy đích và tìm xem những port nào đang mở trước khi hệ thống có thể bị xâm phạm. Quá trình này thường được thực hiện bởi các công cụ dò tìm (scanning tool), kỹ thuật chính để tìm ra máy đích và các port đang mở trên đó. Dò tìm là bước đầu tiên hacker sẽ sử dụng trước khi thực hiện tấn công. Bằng cách sử dụng các công cụ dò tìm như ả map, hacker có thể rà khắp các mạng để tìm ra các máy đích có thể bị tấn công. Một khi xác định được các máy này, kẻ xâm nhập có thể dò tìm các port đang lắng nghe. ả map cũng sử dụng một số kỹ thuật cho phép xác định khá chính xác loại máy đang kiểm tra.

Bằng cách sử dụng những công cụ của chính các hacker thường dùng, người qun trị hệ thống có thể nhìn vào hệ thống của mình từ góc độ của các hacker và giúp tăng cường tính an toàn của hệ thống. Có rất nhiều công cụ dò tìm có thể sử dụng như: ả map, strobe, sscan, SATAả , ...

Dưới đây là một ví dụ sử dụng ả map:

```
# nmap -sS -O 192.168.1.200
Starting nmap V. 2.54 by Fyodor (fyodor@dhp.com,
www.insecure.org/nmap/)
Interesting ports on comet (192.168.1.200):
Port State Protocol Service
```

```
7 open tcp echo
19 open tcp chargen
21 open tcp ftp
...
TCP Sequence Prediction: Class=random positive increments
Difficulty=17818 (Worthy challenge)
Remote operating system guess: Linux 2.2.13
Nmap run completed -- 1 IP address (1 host up) scanned in 5 seconds
```

Tuy nhiên, sử dụng các công cụ này không thể thay thế cho một người quản trị có kiến thức. Bởi vì việc dò tìm thường dự báo một cuộc tấn công, các site nên ưu tiên cho việc theo dõi chúng. Với các công cụ dò tìm, các nhà quản trị hệ thống mạng có thể phát hiện ra những gì mà các hacker có thể thấy khi dò trên hệ thống của mình.

### 9.7. Phát hiện sự xâm nhập qua mạng

Ấu hệ thống của bạn có kết nối vào internet, bạn có thể trở thành một mục tiêu bị dò tìm các lỗ hổng về bảo mật. Mặc dù hệ thống của bạn có ghi nhận điều này hay không thì vẫn không đủ để xác định và phát hiện việc dò tìm này. Một vấn đề cần quan tâm khác là các cuộc tấn công gây ngừng dịch vụ (Denial of Services - DoS), làm thế nào để ngăn ngừa, phát hiện và đối phó với chúng nếu bạn không muốn hệ thống của bạn ngưng trệ.

Hệ thống phát hiện xâm nhập qua mạng (Network Intrusion Detection System - IDS) theo dõi các thông tin truyền trên mạng và phát hiện nếu có hacker đang cố xâm nhập vào hệ thống (hoặc gây ra một vụ tấn công DoS). Một ví dụ điển hình là hệ thống theo dõi số lượng lớn các yêu cầu kết nối TCP đến nhiều port trên một máy nào đó, do vậy có thể phát hiện ra nếu có ai đó đang thử một tác vụ dò tìm TCP port. Một IDS có thể chạy trên máy cần theo dõi hoặc trên một máy độc lập theo dõi toàn bộ thông tin trên mạng.

Các công cụ có thể được kết hợp để tạo một hệ thống phát hiện xâm nhập qua mạng. Chẳng hạn dùng tcpwrapper để điều khiển, ghi nhận các dịch vụ đã được đăng ký. Các chương trình phân tích nhật ký hệ thống, như swatch, có thể dùng để xác định các tác vụ dò tìm trên hệ thống. Và điều quan trọng nhất là các công cụ có thể phân tích các thông tin trên mạng để phát hiện các tấn công DoS hoặc đánh cắp thông tin như tcpdump, ethereal, ngrep, FR (Network Flight Recorder), PortSentry, Sentinel, Snort, ...

Khi hiện thực một hệ thống phát hiện xâm nhập qua mạng bạn cần phải lưu tâm đến hiệu suất của hệ thống cũng như các chính sách bảo đảm sự riêng tư.

### 9.8. Kiểm tra khả năng bị xâm nhập

Kiểm tra khả năng bị xâm nhập liên quan đến việc xác định và sắp xếp các lỗ hổng an ninh trong hệ thống bằng cách dùng một số công cụ kiểm tra. Ấn công cụ kiểm tra cũng có khả năng khai thác một số lỗ hổng tìm thấy để làm rõ quá trình thâm nhập trái phép sẽ được thực hiện như thế nào. Ví dụ, một lỗi tràn bộ đệm của chương trình phục vụ dịch vụ FTP có thể dẫn đến việc thâm nhập vào hệ thống với quyền 'root'. Ấn người quản trị mạng có kiến thức về kiểm tra khả năng bị xâm nhập trước khi nó xảy ra, họ có thể tiến hành các tác vụ để nâng cao mức độ an ninh của hệ thống mạng.

Có rất nhiều các công cụ mạng mà bạn có thể sử dụng trong việc kiểm tra khả năng bị xâm nhập. Hầu hết các quá trình kiểm tra đều dùng ít nhất một công cụ tự động phân tích các lỗ hổng an ninh. Các công cụ này thăm dò hệ thống để xác định các dịch vụ hiện có. Thông tin lấy từ các dịch vụ này sẽ được so sánh với cơ sở dữ liệu các lỗ hổng an ninh đã được tìm thấy trước đó.

Các công cụ thường được sử dụng để thực hiện các kiểm tra loại này là ISS Scanner, Cybercop, Retina, Nessus, Cgiscan, CIS, ...

Kiểm tra khả năng bị xâm nhập cần được thực hiện bởi những người có trách nhiệm một cách cẩn thận. Sự thiếu kiến thức và sử dụng sai cách có thể sẽ dẫn đến hậu quả nghiêm trọng không thể lường trước được.

### **9.9. Đối phó khi hệ thống bị tấn công**

Gần đây, một loạt các vụ tấn công nhắm vào các site của những công ty lớn như Yahoo!, Buy.com, E-Bay, Amazon và Các Interactive gây ra những thiệt hại vô cùng nghiêm trọng. Hình thức tấn công này là dạng tấn công gây ngừng dịch vụ "Denial-Of-Service" mà được thiết kế để làm ngưng hoạt động của một mạng máy tính hay một website bằng cách gửi liên tục với số lượng lớn các dữ liệu tới mục tiêu tấn công khiến cho hệ thống bị tấn công bị ngừng hoạt động, điều này tương tự như hàng trăm người cùng gọi không ngừng tới 1 số điện thoại khiến nó liên tục bị bận.

Trong khi không thể nào tránh được mọi nguy hiểm từ các cuộc tấn công, chúng tôi khuyên bạn một số bước mà bạn nên theo khi bạn phát hiện ra rằng hệ thống của bạn bị tấn công. Chúng tôi cũng đưa ra một số cách để giúp bạn bảo đảm tính hiệu quả của hệ thống an ninh và những bước bạn nên làm để giảm rủi ro và có thể đối phó với những cuộc tấn công.

**Nếu phát hiện ra rằng hệ thống của bạn đang bị tấn công, hãy bình tĩnh. Sau đây là những bước bạn nên làm:**

- Tập hợp 1 nhóm để đối phó với sự tấn công:
  - Nhóm này phải bao gồm những nhân viên kinh nghiệm, những người mà có thể giúp hình thành một kế hoạch hành động đối phó với sự tấn công.
- Dựa theo chính sách và các quy trình thực hiện về an ninh của công ty, sử dụng các bước thích hợp khi thông báo cho mọi người hay tổ chức về cuộc tấn công.
- Tìm sự giúp đỡ từ nhà cung cấp dịch vụ Internet và cơ quan phụ trách về an ninh máy tính:
  - Liên hệ nhà cung cấp dịch vụ Internet của bạn để thông báo về cuộc tấn công. Có thể nhà cung cấp dịch vụ Internet của bạn sẽ chặn đứng được cuộc tấn công.
  - Liên hệ cơ quan phụ trách về an ninh máy tính để thông báo về cuộc tấn công

- Tạm thời dùng phương thức truyền thông khác (chẳng hạn như qua điện thoại) khi trao đổi thông tin để đảm bảo rằng kẻ xâm nhập không thể chặn và lấy được thông tin.
- Ghi lại tất cả các hoạt động của bạn (chẳng hạn như gọi điện thoại, thay đổi file, ...)
- Theo dõi các hệ thống quan trọng trong quá trình bị tấn công bằng các phần mềm hay dịch vụ phát hiện sự xâm nhập (intrusion detection software/services). Điều này có thể giúp làm giảm nhẹ sự tấn công cũng như phát hiện những dấu hiệu của sự tấn công thực sự hay chỉ là sự quấy rối nhằm đánh lạc hướng sự chú ý của bạn(chẳng hạn một tấn công DoS với dụng ý làm sao lãng sự chú ý của bạn trong khi thực sự đây là một cuộc tấn công nhằm xâm nhập vào hệ thống của bạn).  
  
- Sao chép lại tất cả các files mà kẻ xâm nhập để lại hay thay đổi (như những đoạn mã chương trình, log file, ...)
- Liên hệ nhà chức trách để báo cáo về vụ tấn công.

### **Những bước bạn nên làm để giảm rủi ro và đối phó với sự tấn công trong tương lai :**

- Xây dựng và trao quyền cho nhóm đối phó với sự tấn công
- Thi hành kiểm tra an ninh và đánh giá mức độ rủi ro của hệ thống
- Cài đặt các phần mềm an toàn hệ thống phù hợp để giảm bớt rủi ro
- ả ứng cao khả năng của mình về an toàn máy tính

### **Các bước kiểm tra để giúp bạn bảo đảm tính hiệu quả của hệ thống an ninh**

- Kiểm tra hệ thống an ninh mới cài đặt : chắc chắn tính đúng đắn của chính sách an ninh hiện có và cấu hình chuẩn của hệ thống.
- Kiểm tra tự động thường xuyên : để khám phá sự “viếng thăm” của những hacker hay những hành động sai trái của nhân viên trong công ty.
- Kiểm tra ngẫu nhiên: để kiểm tra chính sách an ninh và những tiêu chuẩn, hoặc kiểm tra sự hiện hữu của những lỗ hổng đã được phát hiện (chẳng hạn những lỗi được thông báo từ nhà cung cấp phần mềm)
- Kiểm tra hằng đêm những file quan trọng: để đánh giá sự toàn vẹn của những file và cơ sở dữ liệu quan trọng
- Kiểm tra các tài khoản người dùng: để phát hiện các tài khoản không sử dụng, không tồn tại, ...
- Kiểm tra định kỳ để xác định trạng thái hiện tại của hệ thống an ninh của bạn

### **BẠN CÓ THỂ XEM THÊM THÔNG TIN TẠI**

#### **Các trung tâm giúp đối phó tai nạn trên Internet**

- <http://www.cert.org>



- <http://www.first.org>
- <http://ciac.llnl.gov/>
- <http://www.cert.dfn.de/eng/csir/europe/certs.html>

#### **Một số website về an toàn máy tính**

- <http://www.cs.purdue.edu/coast/>
- <http://www.linuxsecurity.com>
- <http://www.securityportal.com>
- <http://www.tno.nl/instit/fel/intern/wkinfsec.html>
- <http://www.icsa.net>
- <http://www.sans.org>
- <http://www.iss.com>
- <http://www.securityfocus.com>

#### **Thông tin về an toàn từ nhà cung cấp**

- <http://www.calderasystems.com/news/security/>
- <http://www.debian.org/security/>
- <http://www.redhat.com/cgi-bin/support/>

#### **Một số sách về an toàn máy tính**

- Actually Useful Internet Security Techniques by Larry J. Hughes Jr.
- Applied Cryptography: Protocols, Algorithms and Source Code in C by Bruce Schneier
- Building Internet Firewall by Brent Chapman & Elizabeth D. Zwicky
- Cisco IOS & Network Security by Mike Kaeo
- Firewalls and Internet Security by Bill Cheswick & Steve Bellovin
- Halting the Hacker: A practical Guide To Computer Security by Donal L. Pipkin
- Intrusion Detection: An Introduction to Internet Surveillance, Correlation, Traps, Trace Back and Response by Edward G. Amoroso
- Intrusion Detection: & Network Security Beyond the Firewall by Terry Escamilla
- Linux Security by John S. Flowers