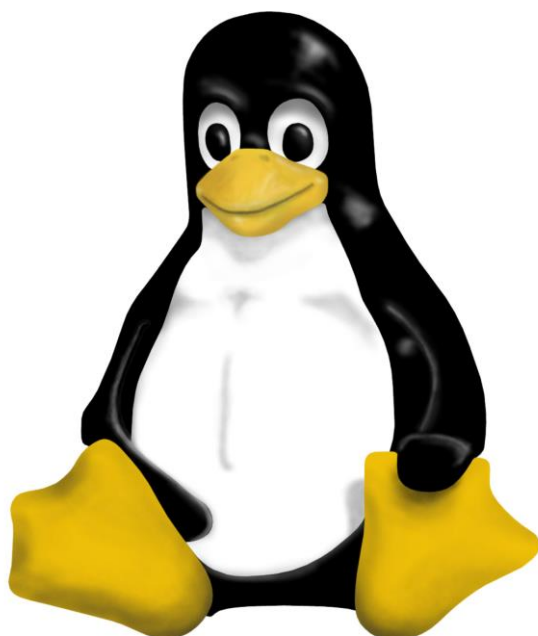


TRƯỜNG ĐẠI HỌC DUY TÂN
KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN KỸ THUẬT MẠNG



BÀI GIẢNG
HỆ ĐIỀU HÀNH UNIX/LINUX



Giảng viên: ThS. ĐẶNG NGỌC CƯỜNG

Đà Nẵng, 2018

| | |
|---|----|
| Chương 1. TỔNG QUAN VỀ UNIX/LINUX | 3 |
| 1.1 Giới thiệu về hệ điều hành | 3 |
| 1.2 Lịch sử phát triển của Unix/Linux | 4 |
| 1.2.1 Unix..... | 4 |
| 1.2.2 Lịch sử phát triển của Linux..... | 5 |
| 1.2.3 Luật bản quyền GNU/GPL..... | 6 |
| 1.2.4 So sánh Linux và Unix | 7 |
| 1.3 Các đặc trưng của Linux | 7 |
| 1.4 Một số ứng dụng trên linux | 8 |
| 1.5 Bản phân phối Linux | 10 |
| 1.5.1 Nhân Linux..... | 10 |
| 1.5.2 Một số bản phân phối Linux | 11 |
| Chương 2. CÀI ĐẶT VÀ CẤU HÌNH LINUX | 13 |
| 2.1 Giới thiệu về Fedora..... | 13 |
| 2.1.1 Hỗ trợ kiến trúc máy | 13 |
| 2.1.2 Yêu cầu hệ thống..... | 13 |
| 2.2 Cài đặt Fedora Linux..... | 14 |
| 2.2.1 Công tác chuẩn bị phân vùng đĩa | 14 |
| 2.2.2 Các bước cài đặt từ đĩa CD | 15 |
| 2.2.3 Boot lần đầu tiên | 17 |
| 2.2.4 Quá trình khởi động và kết thúc của UNIX | 18 |
| 2.2.4 Cấp chạy (runlevel) | 19 |
| 2.3 Đăng nhập và sử dụng hệ thống Linux | 20 |
| 2.3.1 Đăng nhập cục bộ..... | 20 |
| 2.3.2 Đăng nhập vào Linux từ xa: | 20 |
| 2.3.3 Một số lệnh thường dùng: | 22 |
| 2.4 Cài đặt phần mềm trong Linux..... | 25 |
| 2.4.1 Redhat Package Manager - rpm | 25 |
| 2.4.3 Cài đặt phần mềm trong Fedora với tiện ích yum..... | 26 |
| Chương 3. QUẢN LÝ THIẾT BỊ VÀ HỆ THỐNG TẬP TIN | 29 |
| 3.1 Quy tắc quản lý thiết bị | 29 |
| 3.1.1 Quy ước tên..... | 29 |
| 3.1.2 Kết gán hệ thống tập tin | 30 |
| 3.2 Hệ thống tập tin Linux | 33 |
| 3.2.1 Giới thiệu..... | 33 |
| 3.2.2 Cấu trúc hệ thống tập tin | 33 |
| 3.2.3 Thao tác trên hệ thống tập tin..... | 36 |
| 3.2.3.1 Các lệnh thường dùng | 36 |
| 3.2.3.2 Quyền truy xuất tập tin/thư mục | 41 |
| 3.2.3.3 Liên kết tập tin thư mục | 49 |
| 3.3 Xử lý văn bản và các bộ lọc | 50 |
| 3.3.1 Công cụ soạn thảo văn bản..... | 50 |
| 3.3.2 Pipe và các bộ lọc..... | 51 |
| 3.3.2.1 Cơ chế Pipes..... | 51 |
| 3.3.2.2 Các lệnh lọc (filters)..... | 52 |
| Chương 4. QUẢN TRỊ HỆ THỐNG VÀ NGƯỜI DÙNG | 55 |
| 4.1 Quản trị người dùng | 55 |
| 4.1.1 Tài khoản người dùng (user) | 55 |
| 4.1.2 Nhóm và các vấn đề liên quan | 59 |
| 4.2 Kiểm tra hoạt động hệ thống..... | 63 |
| 4.2.1 Hệ thống file ảo /proc..... | 63 |
| 4.2.2 Hệ thống log file..... | 63 |

| | |
|---|----|
| 4.3. Quản trị tiến trình | 65 |
| 4.3.1 Tiến trình | 65 |
| 4.3.2 Độ ưu tiên của tiến trình..... | 67 |
| 4.3.3 Xử lý tín hiệu..... | 68 |
| 4.3.4 Lập lịch tiến trình | 69 |
| Chương 5. LẬP TRÌNH SHELL SCRIPT | 73 |
| 5.1 Sử dụng Shell | 73 |
| 5.1.1 Giới thiệu..... | 73 |
| 5.1.2 Mục đích của shell..... | 73 |
| 5.1.3 Điều khiển shell..... | 73 |
| 5.1.4 Môi trường Shell | 74 |
| 5.2 Lập trình shell..... | 76 |
| 5.2.1 Cấu trúc chương trình shell | 76 |
| 5.2.2 Các thành phần cơ bản | 76 |
| 5.2.3 Các cấu trúc điều khiển | 82 |
| 5.2.4 Một số tiện ích..... | 83 |
| 5.2.5 Một số chương trình tham khảo | 84 |

Chương 1. TỔNG QUAN VỀ UNIX/LINUX

Trong chương này giới thiệu về hệ điều hành Unix/Linux về lịch sử phát triển, luật bản quyền mã nguồn mở và các bản phân phối Linux phổ biến hiện nay. Trong chương cũng giới thiệu các đặc trưng chủ yếu của hệ thống Linux và một số ứng dụng phần mềm mã nguồn mở được sử dụng nhiều trên Linux.

1.1 Giới thiệu về hệ điều hành

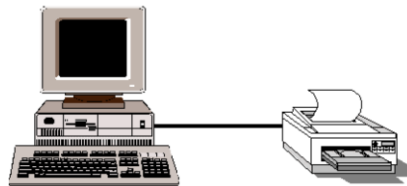
Hệ điều hành là một phần mềm hệ thống điều khiển sự hoạt động của máy tính, quản lý các tài nguyên hệ thống, bộ nhớ và cung cấp môi trường để chạy để người sử dụng tương tác với máy tính và chạy các chương trình ứng dụng.

Một số hệ điều hành phổ biến: Linux, Unix-Solaris, SCO and Irix, Microsoft DOS, Microsoft Windows,...

Các kiểu hệ thống

- Hệ thống đơn người dùng (Single-User System)

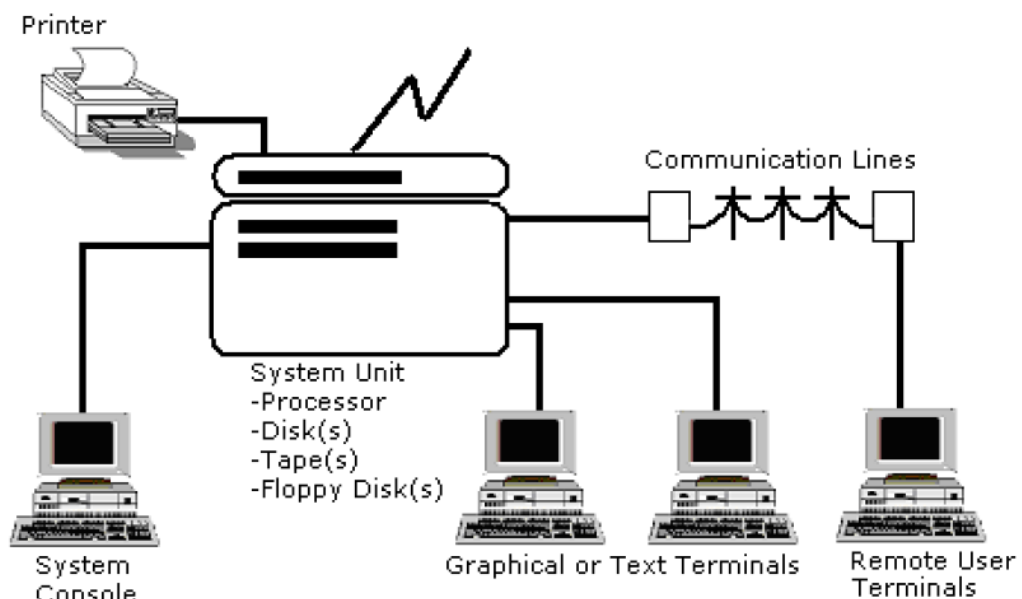
Hệ đơn người dùng là hệ thống nhỏ, đa năng, được sử dụng bởi một người tại một thời điểm như các máy tính cá nhân (PC).



Một ví dụ của hệ thống đơn người dùng là MS DOS.

- Hệ thống đa người dùng (Multi-User System)

Hệ thống đa người dùng có thể chạy nhiều chương trình của nhiều người dùng đồng thời, chia sẻ các tài nguyên như máy in, đĩa và hỗ trợ nhiều người ử dùng cùng làm việc trên một hệ thống.



Các ví dụ của hệ thống đa người dùng là Linux, Unix,...

1.2 Lịch sử phát triển của Unix/Linux

1.2.1 Unix

Giữa năm 1960, AT&T Bell Laboratories và một số trung tâm khác tham gia vào một cố gắng tạo ra một hệ điều hành mới được đặt tên là Multics (Multiplexed Information and Computing Service). Đến năm 1969, chương trình Multics bị bãi bỏ vì đó là một dự án quá nhiều tham vọng. Thậm chí nhiều yêu cầu đối với Multics thời đó đến nay vẫn chưa có được trên các Unix mới nhất. Nhưng Ken Thompson, Dennis Ritchie, và một số đồng nghiệp của Bell Labs đã không bỏ cuộc. Thay vì xây dựng một HĐH làm nhiều việc một lúc, họ quyết định phát triển một HĐH đơn giản chỉ làm tốt một việc là chạy chương trình (run program).

HĐH sẽ có rất nhiều các công cụ (tool) nhỏ, đơn giản, gọn nhẹ (compact) và chỉ làm tốt một công việc. Bằng cách kết hợp nhiều công cụ lại với nhau, họ sẽ có một chương trình thực hiện một công việc phức tạp. Đó cũng là cách thức người lập trình viết ra chương trình. Peter Neumann đặt tên Unix cho HĐH đơn giản này. tiếp tục phát triển theo mô hình ban đầu và đặt ra một hệ thống tập tin mà sau này được phát triển thành hệ thống tập tin của UNIX. Vào năm 1973, sử dụng ngôn ngữ C của Ritchie, Thompson đã viết lại toàn bộ HĐH Unix và đây là một thay đổi quan trọng của Unix, vì nhờ đó Unix từ chỗ là HĐH cho một máy PDP-xx trở thành HĐH của các máy khác với một cố gắng tối thiểu để chuyển đổi. Khoảng 1977 bản quyền của UNIX được giải phóng và HDH UNIX trở thành một thương phẩm.



Ken Thompson (trái) và Dennis Ritchie (giữa) nhận giải thưởng quốc gia về công nghệ do đã phát minh ra hệ điều hành UNIX và ngôn ngữ lập trình C (ngày 27/4/1999)

Hai dòng UNIX: System V của AT&T, Novell và Berkeley Software Distribution (BSD) của Đại học Berkeley.

- *System V*

Các phiên bản UNIX cuối cùng do AT&T xuất bản là System III và một vài phát hành (releases) của System V. Hai bản phát hành gần đây của System V là Release 3 (SVR3.2) và Release 4.2 (SVR4.2). Phiên bản SVR 4.2 là phổ biến nhất cho từ máy PC cho tới máy tính lớn.

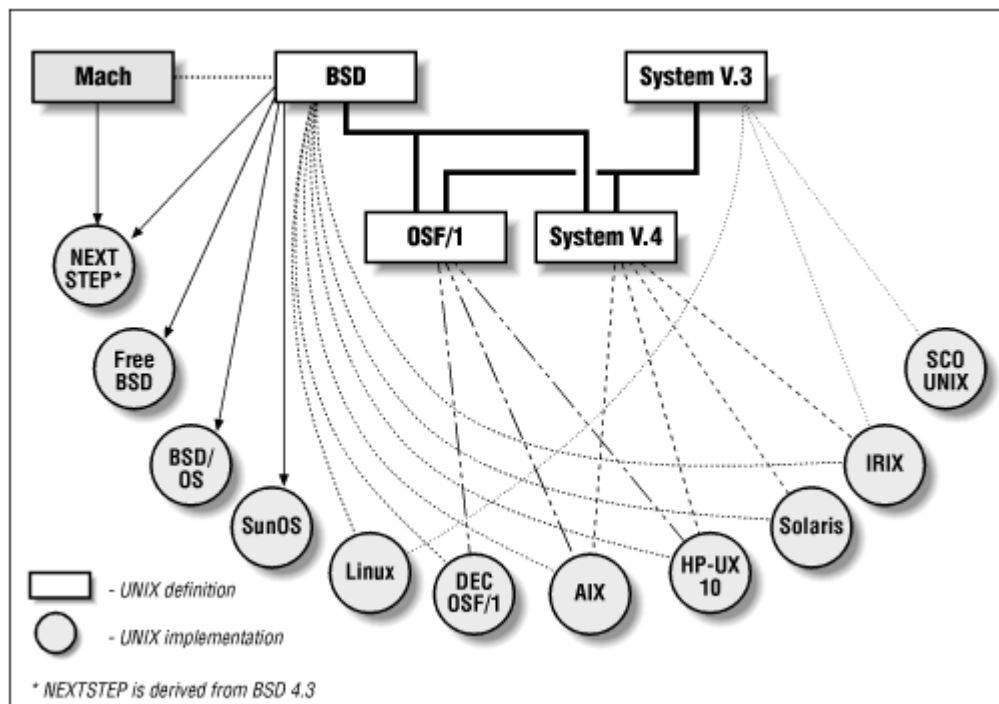
▪ BSD

Từ 1970 Computer Science Research Group của University of California tại Berkeley (UCB) xuất bản nhiều phiên bản UNIX, được biết đến dưới tên Berkeley Software Distribution, hay BSD. Cải biến của PDP-11 được gọi là 1BSD và 2BSD. Trợ giúp cho các máy tính của Digital Equipment Corporation VAX được đưa vào trong 3BSD. Phát triển của VAX được tiếp tục với 4.0BSD, 4.1BSD, 4.2BSD, và 4.3BSD

Trước 1992, UNIX là tên thuộc sở hữu của AT&T. Từ 1992, khi AT&T bán bộ phận Unix cho Novell, tên Unix thuộc sở hữu của X/Open foundation. Tất cả các hệ điều hành thỏa mãn một số yêu cầu đều có thể gọi là Unix. Ngoài ra, Institute of Electrical and Electronic Engineers (IEEE) đã thiết lập chuẩn “An Industry-Recognized Operating Systems Interface Standard based on the UNIX Operating System”. Kết quả cho ra đời POSIX.1 (cho giao diện C) và POSIX.2 (cho hệ thống lệnh trên Unix)

Kết lại, vấn đề chuẩn hóa UNIX vẫn còn rất xa kết quả cuối cùng. Nhưng đây là quá trình cần thiết có lợi cho sự phát triển của ngành tin học nói chung và sự sống còn của HDH UNIX nói riêng.

Các phiên bản của Unix



1.2.2 Lịch sử phát triển của Linux

Linux là một HDH dạng UNIX (Unix-like Operating System) chạy trên máy PC với bộ điều khiển trung tâm (CPU) Intel 80386 trở lên, hay các bộ vi xử lý trung tâm tương thích AMD, Cyrix. Linux ngày nay còn có thể chạy trên các máy Macintosh hoặc SUN Sparc. Linux thỏa mãn chuẩn POSIX.1.

Linux được viết lại toàn bộ từ con số không, tức là không sử dụng một dòng lệnh nào của Unix để tránh vấn đề bản quyền của Unix. Tuy nhiên hoạt động của Linux hoàn toàn dựa trên nguyên tắc của hệ điều hành Unix. Vì vậy nếu một người nắm được Linux, thì sẽ nắm được UNIX. Nên chú ý rằng giữa các Unix sự khác nhau cũng không kém gì giữa Unix và Linux.

Năm 1991 Linus Torvalds, sinh viên của đại học tổng hợp Helsinki, Phần lan, bắt đầu xem xét Minix, một phiên bản của Unix làm ra với mục đích nghiên cứu cách tạo ra một hệ điều hành Unix chạy trên máy PC với bộ vi xử lý Intel 80386.

Ngày 25/8/1991, Linus cho ra version 0.01 và thông báo trên comp.os.minix của Internet về dự định của mình về Linux.

1/1992, Linus cho ra version 0.12 với shell và C compiler. Linus không cần Minix nữa để recompile HDH của mình. Linus đặt tên HDH của mình là Linux.

1994, phiên bản chính thức 1.0 được phát hành.



Tác giả hệ điều hành Linux – Linus Torvalds

Quá trình phát triển của Linux được tăng tốc bởi luật bản quyền GNU (GNU's Not Unix), đó là chương trình phát triển các Unix có khả năng chạy trên nhiều platform. Từ phiên bản của Linux kernel là 2.4.20, có khả năng điều khiển các máy đa bộ vi xử lý (hiện tại Linux hỗ trợ máy tính có tối đa 16 CPUs) và rất nhiều các tính năng khác.

Phiên bản mới nhất có thể tìm thấy tại <http://www.kernel.org>

Biểu tượng của linux là chim cánh cụt (Linux Penguin Tux), được lựa chọn bởi Linus Torvalds để trình bày hình ảnh kết hợp với hệ điều hành.



1.2.3 Luật bản quyền GNU/GPL

Các chương trình tuân theo GNU Copyleft or GPL (General Public License) có bản quyền như sau:

- Tác giả vẫn là sở hữu của chương trình của mình.

- Ai cũng được quyền bán copy của chương trình với giá bất kỳ mà không phải trả cho tác giả ban đầu.
- Người sở hữu chương trình tạo điều kiện cho người khác sao chép chương trình nguồn để phát triển tiếp chương trình.

1.2.4 So sánh Linux và Unix

Linux được phát triển và xem Unix như mô hình tham chiếu và tiếp tục có cùng kiến trúc và các chức năng cơ bản.

Linux và Unix khác nhau ở:

- Không gian đĩa cứng yêu cầu
- Shell
- Bản phân phối (variants)
- Vấn đề bản quyền

Bảng so sánh sự khác nhau giữa Linux và Unix:

| | | |
|-----------|--|--|
| Variants | Red Hat, Caldera, Debian, LinuxPPC, SuSE | AT & T, MULTICS, BSD, SCO, HP-Ux, IRIX, Ultrix, XENIX, Sun Solaris |
| Licensing | Freely distributed | Expensive licensing |

Comparison between Linux and Unix

1.3 Các đặc trưng của Linux

-*Linux là miễn phí (free) và Open Source*: Mã nguồn mở, bao gồm cả kernel, drivers, các công cụ phát triển,...

-*Linux rất ổn định*: Ngay cả server Linux phục vụ những mạng lớn (hàng trăm máy trạm) cũng hoạt động rất ổn định.

-*Multi-Tasking, Multi-Threading*: là khả năng mà HĐH gán cho từng tiến trình hoặc tuyến quyền sử dụng CPU trong một khoảng thời gian nhất định

-*Multi-User*: là khả năng cho phép nhiều người dùng đồng thời truy cập cùng một CPU.

-*Multi-platform*: Linux có thể chạy trên nhiều loại máy khác nhau: Alpha, AMD (x86-64bit), Intel, MIPS, PC 386, 486, SUN Sparc,...

-*Multi-standard Compliant*: Tương thích với hầu hết các hệ POSIX, System V, và BSD (ở mức source).

-*Hỗ trợ nhiều hệ thống File*: Minix-1, MS-DOS, VFAT, FAT-32, ISO 9660 (CD-ROMs),...hai hệ thống tập tin chính của Linux là ext2fs và ext3fs.

-*Multiple Networking Protocols*: Các giao thức nền tảng được hỗ trợ bởi Kernel như: TCP, IPv4, IPv6, AX.25, X.25, IPX, Appletalk, Netrom, v.v...

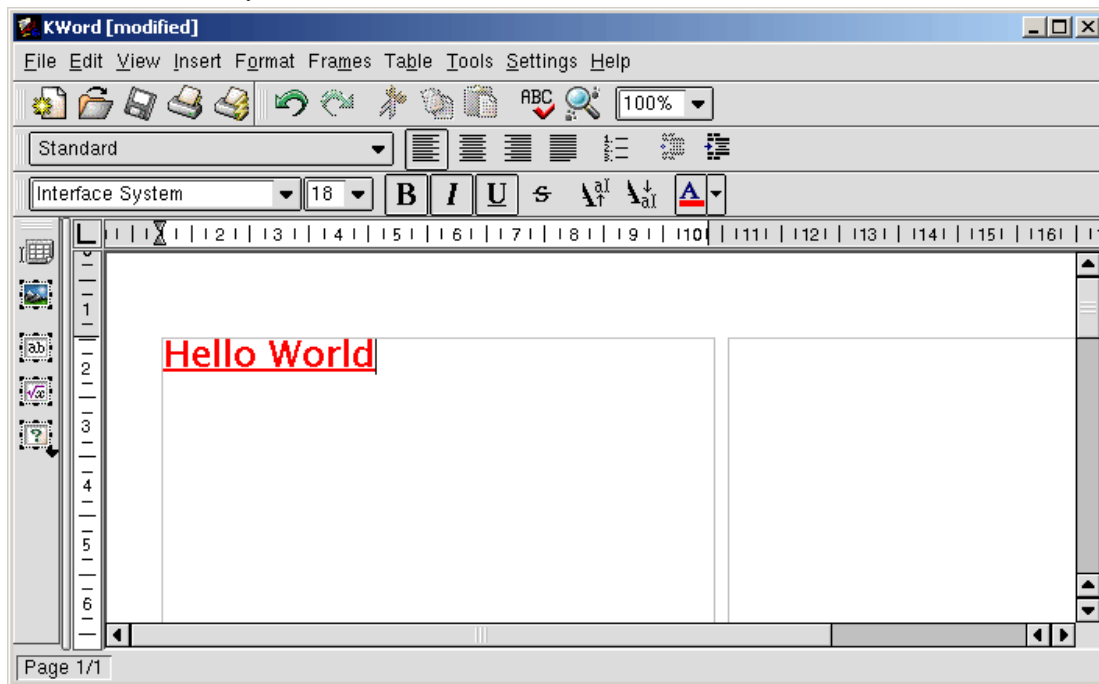
1.4 Một số ứng dụng trên linux

a. Các ứng dụng văn phòng & multimedia

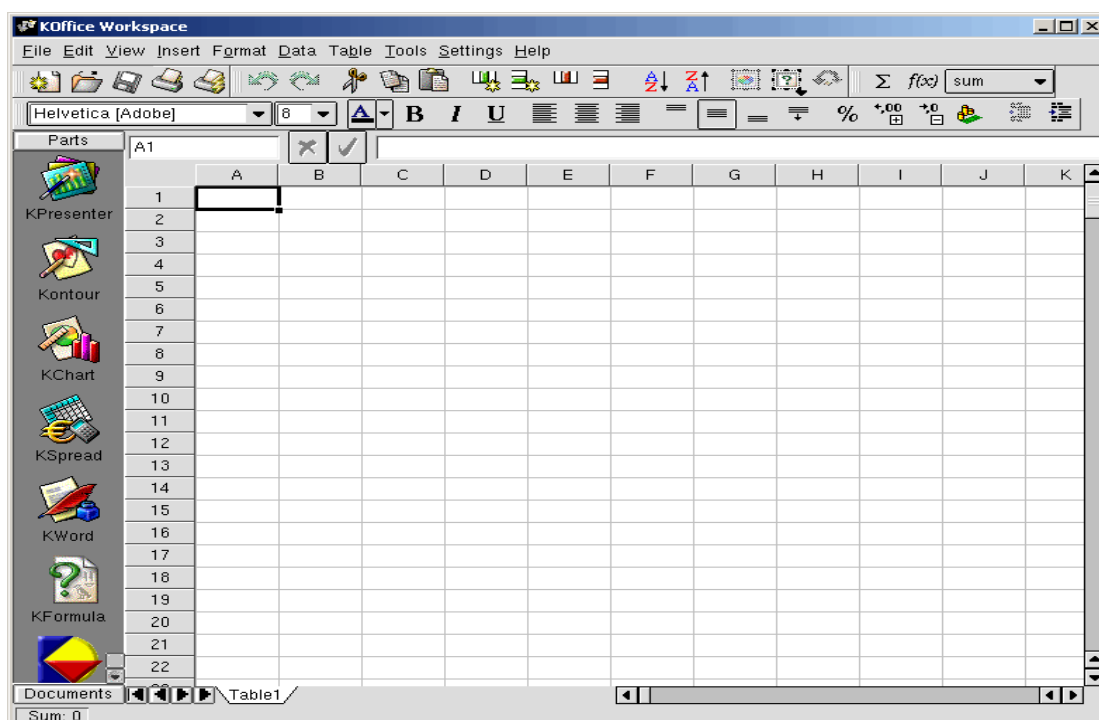
Cùng với thời gian, hệ điều hành Linux ngày càng được hoàn thiện, nhiều hãng sản xuất cùng với các lập trình viên đã xây dựng được một kho thư viện phần mềm khổng lồ đáp ứng phần lớn nhu cầu của người dùng.

Nếu như trong Windows có bộ Microsoft Office thì trong Linux có những bộ Office khác như KOffice hoặc bộ Star Office của hãng Sun Microsystems được phân phối miễn phí.

- KWord: Trình soạn thảo văn bản

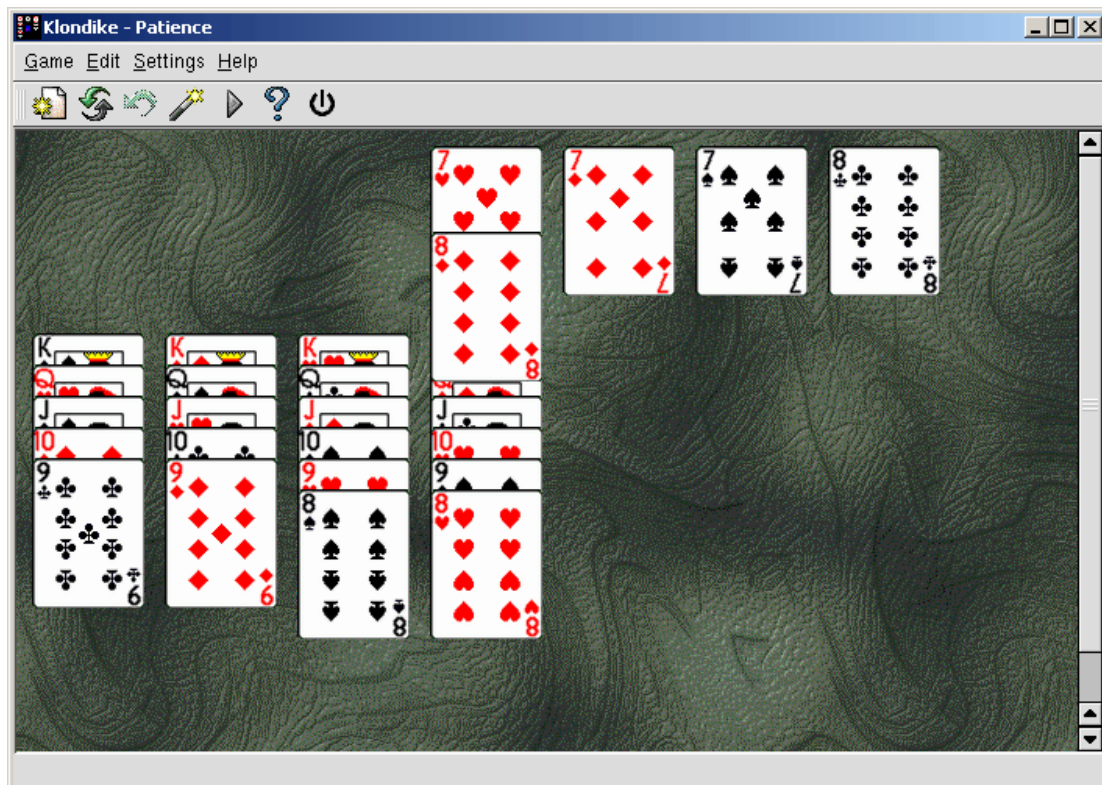


- KOffice Workspace: Trình xử lý bảng tính



b. Các ứng dụng giải trí

Ngoài các ứng dụng văn phòng ra, Linux cũng có khá nhiều games phục vụ nhu cầu thư giãn và giải trí của người dùng.



c. Các ứng dụng mạng và Internet

Có thể nói các ứng dụng mạng là tập hợp những ứng dụng nổi bật nhất của hệ điều hành Linux. Những khả năng mà các ứng dụng mạng trên Linux có thể thực hiện được làm cho hệ điều hành này trở nên vượt trội hơn so với Windows.

Linux cho phép người dùng có thể cấu hình 1 server với đầy đủ các ứng dụng cơ bản nhất của Internet: Domain Name Service (DNS), Web Server, Web Proxy Server, Routing, SMTP Server, Pop3 Server, Firewall, ...

Với sự bùng nổ các ứng dụng trên Internet, hệ điều hành Linux hỗ trợ một môi trường lý tưởng cho các server ứng dụng.

Các hãng phần mềm nổi tiếng đều nghiên cứu để làm sao có thể cài các ứng dụng của họ lên Linux. Tiêu biểu là các ứng dụng sau:

- Oracle Internet Application Server 9i
- IBM WebSphere

d. Các ứng dụng cơ sở dữ liệu

Các ứng dụng cơ sở dữ liệu là không thể thiếu trong lĩnh vực CNTT cũng như các ngành khác. Một khi đã nói đến sự tin học hoá trong mọi lĩnh vực của đời sống thì dù ít, dù nhiều cũng phải liên quan đến cơ sở dữ liệu.

Linux hỗ trợ khá mạnh các hệ quản trị cơ sở dữ liệu từ miễn phí đến các hệ chuyên nghiệp như: Postgres SQL, MySQL, Oracle Database Server 9i, IBM DB2.

e. Các công cụ phát triển (Development)

Hệ điều hành Linux được viết lại hoàn toàn từ đầu bằng ngôn ngữ C nhằm tránh vấn đề bản quyền của Unix. Do đó ngôn ngữ lập trình C được hỗ trợ mạnh mẽ và khá đầy đủ cho việc phát triển các ứng dụng mạng.

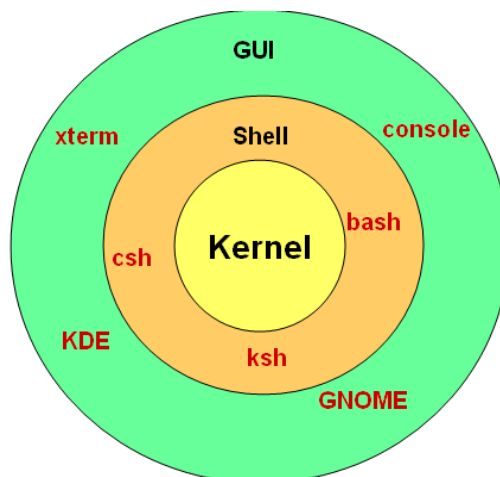
Tuy nhiên ngôn ngữ C không phải là sự lựa chọn duy nhất cho việc lập trình trên Linux. Có nhiều ngôn ngữ lập trình khác cũng được hỗ trợ bởi Linux được liệt kê dưới đây: Ada, C, C++, Forth, Fortran, Icon, Java, Lisp, Modular 2, Modular 3, Oberon, Objective C, Pascal, Perl, Prolog, Python, Smalltalk, SQL, Tck/Tl, Shell,...

1.5 Bản phân phối Linux

Bản phân phối Linux là bộ các chương trình ứng dụng bao gồm cả 4 phần chính của một hệ điều hành (shell, kernel, file system, utility) và các chương trình phục vụ người dùng,...

Hầu hết các chương trình trong bản phân phối hầu hết đều theo bản quyền GPL.

Hiện nay có rất nhiều công ty cung cấp các bản phân phối khác nhau (tham khảo ở <http://www.linuxhq.com>).



1.5.1 Nhân Linux

▪ Nhân (kernel)

Kernel là phần chính của hệ điều hành, phụ trách hầu hết các chức năng chính của hệ điều hành như quản lý bộ nhớ, thực thi nhiệm vụ và truy nhập phần cứng...

Phần hạt nhân (lõi hay kernel) của Linux có thể hiểu đơn giản là một tập hợp các chương trình thường trú trong bộ nhớ.

Dự án nhân Linux được khởi xướng vào năm 1991 bởi Linus Torvalds bằng một bài viết nổi tiếng trong nhóm tin Usenet comp.os.minix, trong đó có đoạn viết:

"I'm doing a (free) operating system (just a hobby, won't be big and professional like gnu) for 386(486) AT clones..." [1]

▪ Cách đánh số phiên bản.

Cần phân biệt số phiên bản của bản phân phối với số phiên bản của nhân. Nhân Linux hiện đang được điều hành và phát triển bởi Linus Torvalds, nên phiên bản của nhân tăng theo thứ tự, chứ không phân nhánh và nhân lên như các bản phân phối.

Ví dụ:

Bản phân phối openSuSE Linux 10.1 (kernel 2.6.16.13)

Bản phân phối Fedora 5 (kernel 2.6.16.13)

Phiên bản nhân Linux được xác định bởi hệ thống số dạng: X.YY.ZZ

Nếu YY là số chẵn => phiên bản ổn định.

Nếu YY là số lẻ => phiên bản thử nghiệm (không dùng để phát triển các bản phân phối)

Ví dụ: Kernel 2.4.20 cho biết: 2 là số chính; 4 là số phụ, phiên bản ổn định; 20 là cấp vá đấp (patch level).

1.5.2 Một số bản phân phối Linux

- *Redhat và Fedora Core*

Redhat và Fedora Core là hai bản Linux có lẽ là thịnh hành nhất trên thế giới, phát hành bởi công ty Redhat.

Từ 2003, Công ty Redhat phát triển Redhat Enterprise Linux (RHEL) với mục đích thương mại, nhằm vào các công ty, xí nghiệp.

Redhat cũng đầu tư mở ra dự án Fedora nhằm phát triển phiên bản Fedora Core cho người dùng bình thường.

Bản Linux của RedHat cuối cùng dừng ở phiên bản 9.0. Phiên bản của Fedora được bắt đầu từ 1.

- *SuSE Linux*

Made in Germany (Đức). Bản Linux cực kỳ thịnh hành ở châu Âu và Bắc Mỹ.

Năm 2003, công ty SuSE bị Novell mua. Novell đang dốc sức đầu tư cho SuSE để nhắm vào enterprise users cạnh tranh với Redhat.

Bản SuSE mới nhất hiện nay là 9.1

Web site: <http://www.suse.com>

- *Debian Linux*

Là bản phân phối Linux cũng rất phổ biến. Nhiều người có ý kiến cho rằng:

Người không chuyên nên dùng Fedora Core để có thể làm quen với những kỹ thuật mới. Dân chuyên nghiệp nên dùng Debian vì sự ổn định tuyệt vời của nó.

Bản mới nhất 3.0R2

Web site: <http://www.debian.org>

- *Mandrake Linux*

Made in France. Cũng rất thịnh hành ở châu Âu, Mỹ và Việt Nam. Đây là bản được ưa ái nhất trong vấn đề Việt hóa.

Bản mới nhất hiện nay là 10.0

Web site: <http://www.mandrakelinux.com>

- *Slackware*

Slackware is one of the popular distributions of Linux. It is a complete 32-bit multitasking “UNIX-like” system and is based around the 2.4 Linux kernel series and the GNU C Library version 2.3.2 (libc6).

Slackware contains an easy to use installation program, extensive online documentation, and a menu-driven package system.

Web site: <http://www.slackware.com>

- *Turbo Linux*

Nổi tiếng tại Nhật, Trung Quốc. Công ty Turbo đang đầu tư mạnh để thống trị thị trường Trung Quốc

Bản Turbo mới nhất hiện nay là 10F

- *Knoppix Linux*

Made in Germany. Bản live Linux cũng được ưa chuộng hiện nay.

Cho phép khởi động trực tiếp từ CD mà không cần cài đặt vào ổ cứng.

Phiên bản mới nhất là 3.4

❶ Câu hỏi bài tập

1. Cho biết lịch sử phát triển của Unix/Linux?
2. Cho biết luật bản quyền GNU/GPL?
3. Phân biệt giữa phiên bản nhân Linux và phiên bản các bản phân phối Linux?
4. Nêu một số bản phân phối Linux phổ biến hiện nay?

Chương 2. CÀI ĐẶT VÀ CẤU HÌNH LINUX

Trong chương này chúng ta sẽ tìm hiểu về cách thức cài đặt hệ điều hành Linux Fedora và thiết lập một số thông số cấu hình hệ thống. Cách thức cài đặt phần mềm trên Linux và giới thiệu một số lệnh cơ bản làm việc với hệ thống Linux.

2.1 Giới thiệu về Fedora

Fedora Core là một Bản phân phối Linux dựa trên RPM Package Manager, được phát triển dựa trên cộng đồng theo "Dự án Fedora (Fedora Project)" và được bảo trợ bởi Red Hat.



Dự án Fedora nhắm tới mục đích tạo ra một hệ điều hành mã nguồn mở hoàn chỉnh để sử dụng cho các mục đích tổng quát.

Fedora được thiết kế để có thể dễ dàng cài đặt với chương trình cài đặt mang giao diện đồ họa. Các phiên bản mới hơn của Fedora có thể được phát hành mỗi 6 hoặc 8 tháng. Hỗ trợ kỹ thuật của Fedora đa số là đến từ cộng đồng.

2.1.1 Hỗ trợ kiến trúc máy

-i386: Intel x86-compatible processors, bao gồm Intel Pentium và Pentium-MMX, Pentium Pro, Pentium-II, Pentium-III, Celeron, Pentium 4, and Xeon; VIA C3/C3-m và AthlonMP, và Sempron.

-ppc: PowerPC processors, như Apple Power Macintosh, G3, G4, và G5, và IBM pSeries systems

-x86_64: 64-bit AMD processors như Athlon64, Turion64, Opteron; và Intel 64-bit processors như EM64T

2.1.2 Yêu cầu hệ thống

Để có thể cài đặt thành công Linux Fedora Core 5 trên kiến trúc máy x86, hệ thống cần thỏa mãn các yêu cầu tối thiểu:

- *Processor*

Chế độ text: 200 MHz Pentium-class

Chế độ đồ họa: 400 MHz Pentium II

Intel processors with Intel® Extended Memory 64 Technology (Intel® EM64T)

- *Bộ nhớ chính:*

Chế độ text: 128MB RAM

Chế độ đồ họa: 256MB RAM

- *Không gian đĩa:*

2.2 Cài đặt Fedora Linux

2.2.1 Công tác chuẩn bị phân vùng đĩa

Điều này cho phép bạn tạo các phân vùng mới trên không gian trống của đĩa, hoặc để sử dụng các phân vùng Linux tồn tại.

- Cấp phát không gian trao đổi thích hợp.
- Xác định hệ thống file nào để sử dụng
- Cấu hình phần cứng và phần mềm.

Việc cài đặt hệ điều hành Linux lên một máy mới hoàn toàn, chưa có chứa dữ liệu gì cả (khi đĩa cứng chưa fdisk càng tốt) thì thật dễ dàng và nhanh chóng. Nhưng đa số người sử dụng máy vi tính đều khá quen thuộc với hệ điều hành Microsoft Windows do đó hầu hết các máy tính hiện nay đều cài đặt sẵn hệ điều hành này ! Một vấn đề đặt ra là làm sao đối với học viên học Linux là có thể cài đặt Linux lên máy đã có sẵn một hệ điều hành Windows rồi mà không làm mất dữ liệu.

Đối với hệ điều hành Linux ngày nay nó đòi hỏi phải có ít nhất 2 loại partion của đĩa cứng để có thể cài đặt thành công:

- *Data parttion*: dùng để chứa nhân hệ điều hành. Dung lượng cho parttion này tùy theo các package mà bạn cài đặt, thông thường khoảng 2Gb là đủ.
- *Swap parttion*: dùng để làm swap. Dung lượng cho parttion này không cần lớn lắm, chỉ cần bằng hoặc gấp đôi dung lượng của RAM là vừa đủ ! Nếu ta khai báo lớn quá thì hệ thống cũng sẽ không dùng hết dẫn đến phí tài nguyên đĩa. Còn nếu ta khai báo dung lượng nhỏ quá thì sẽ dẫn đến hiệu suất hoạt động của hệ thống giảm do không đủ swap space.

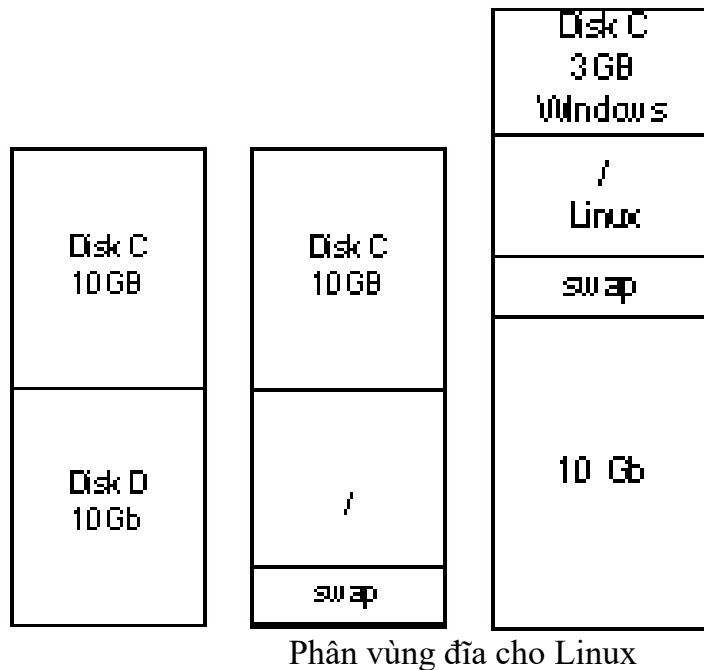
Đặc biệt đối với các hệ thống Linux mà sau này muốn cài đặt hệ quản trị CSDL Oracle lên thì ta phải cho swap space lớn hơn hoặc bằng 500MB vì đây là một trong những khuyến cáo của Oracle. Ta phải để ý đến trường hợp này, nếu không khi hệ thống đã cài xong, các parttion đã ổn định rồi thì không thể thay đổi được !

Sau khi đã xác định những gì cần làm ta sẽ bắt tay vào cài đặt Linux.

Cũng như các hệ điều hành khác, Linux yêu cầu ta chia các partion cần thiết để chứa dữ liệu. Vấn đề là hiện tại máy tính của ta đang có sẵn Windows và trong Windows, ta có rất nhiều dữ liệu quan trọng không muốn bị mất thì ta phải làm thế nào.

Giả sử ta có một đĩa cứng 20GB, có chia 2 partion thành 2 đĩa Logic C và D với dung lượng C: 10Gb, D: 10Gb. Và ổ đĩa C là ổ đĩa hệ thống chứa hệ điều hành Windows ổ đĩa

Dùng để ta lưu backup dữ liệu. Để cài đặt ta phải dành trọn phần partition D để install Linux. Không những thế ta phải cắt partition D thành 2 partition: 1 dùng để cài đặt chương trình, một dùng để làm swap.



2.2.2 Các bước cài đặt từ đĩa CD

Cho CD 1 vào ổ CDROM và boot máy, xuất hiện màn hình boot:

Testing CD và DVD Media

Chọn OK để test đĩa, hoặc chọn Skip

Fedora Linux có hai chế độ giao diện cài đặt

-*Giao diện đồ họa – GUI*: Giao diện hướng dẫn cài đặt Fedora Linux bằng hình ảnh đồ họa. Tương tự như hướng dẫn cài đặt Windows. Nếu ta chọn giao diện này thì quá trình cài đặt thường chậm hơn bởi vì ta sử dụng thiết bị input chủ yếu là mouse.

-*Giao diện văn bản – text*: Giao diện cài đặt Fedora Linux toàn bằng menu dòng lệnh. Sử dụng giao diện cài đặt này thường thuận lợi và nhanh chóng hơn vì tất cả đều sử dụng bàn phím nên thao tác sẽ nhanh hơn.

Step 1. Lựa chọn ngôn ngữ

Chọn ngôn ngữ sử dụng trong khi cài đặt, nhấn Next

Step 2. Cấu hình bàn phím

Chọn kiểu bàn phím thích hợp với hệ thống

Step 3. Chọn cài mới hoặc nâng cấp hệ thống

Chọn 'Install Fedora Core' để cài đặt mới, nhấn Next

Step 4. Cấu hình boot loader

Boot loader là phần mềm cho phép định vị và khởi động hệ điều hành.

Nếu một boot loader đã tồn tại như BootMagic™, System Commander™ được cài đặt bởi Microsoft Windows, hệ thống cài đặt Fedora không thể cập nhật nó.

GRUB là một boot loader chuẩn của Fedora

Step 5. Cấu hình mạng

Step 6. Chọn Time Zone



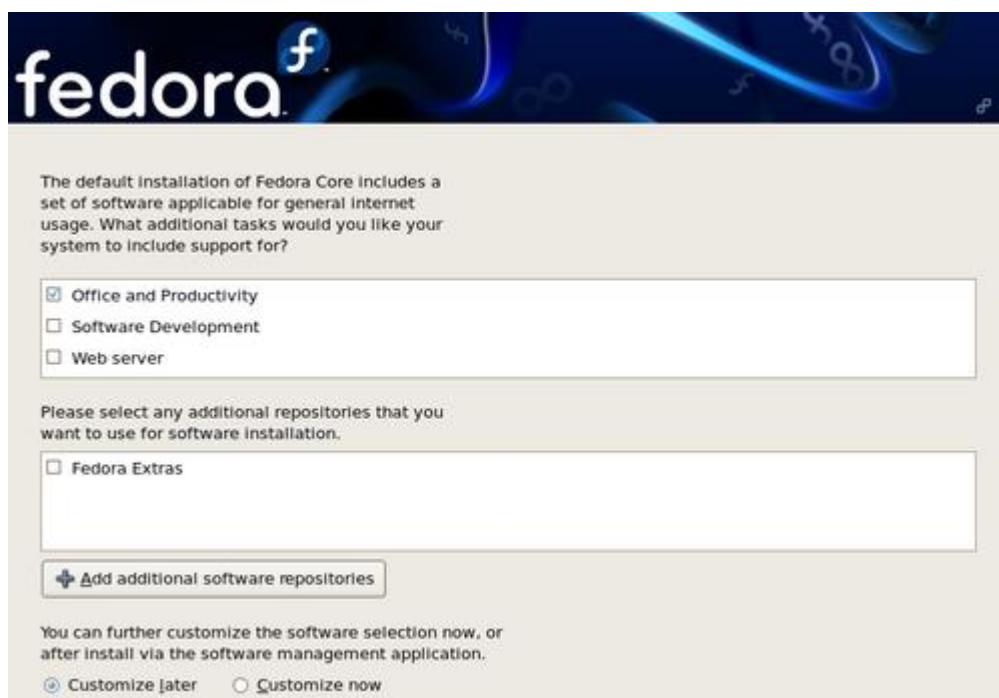
Step 7. Đặt mật khẩu tài khoản Root

Fedora sử dụng một tài khoản đặc biệt có tên là *root* để quản trị hệ thống.

Tài khoản root có khả năng kiểm soát toàn bộ hệ thống,



Step 8. Chọn gói phần mềm



Trong đó:

Office and Productivity: Tùy chọn này cung cấp sản phẩm OpenOffice.org thích hợp, ứng dụng quản lý dự án Planner, các công cụ đồ họa như GIMP, và các ứng dụng đa phương tiện.

Software Development: Tùy chọn này cung cấp các công cụ cần thiết để biên dịch phần mềm trên hệ thống Fedora.

Web server: Tùy chọn này cung cấp Apache Web server.



Step 9. Tiến trình cài đặt các gói

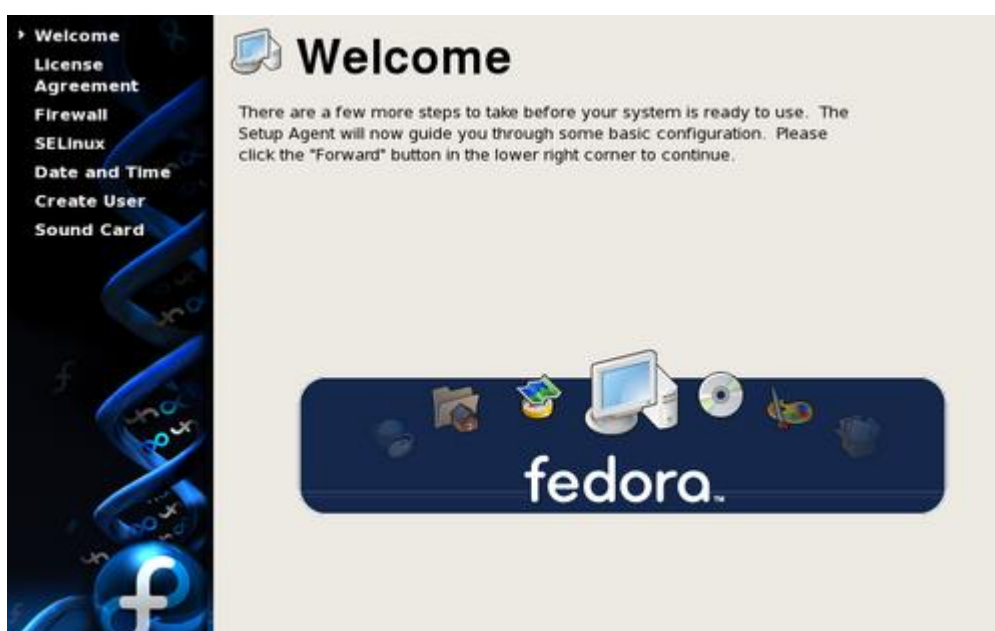
Fedora Core báo cáo tiến trình cài đặt trên màn hình khi cài các gói vào hệ thống.

Nếu sử dụng CDs để cài, Fedora Core thông báo thay đổi các đĩa. Sau khi chèn một đĩa, chọn OK để tiếp tục quá trình cài đặt.

Sau khi cài đặt thành công, chọn Reboot để khởi động lại máy.

2.2.3 Boot lần đầu tiên

- *Setup Agent* sẽ chạy để cấu hình hệ thống trước khi đăng nhập



- *Firewall*

Firewall (tường lửa) dùng kiểm tra các kết nối mạng đến hệ thống dựa trên một tập luật. Những luật này xác định kiểu kết nối nào là được phép và kết nối nào là bị cấm.

Mặc định firewall có hiệu lực, với một tập các luật đơn giản cho phép các kết nối được thực hiện, nhưng chỉ cho phép duyệt mạng và kết nối SSH (Secure SHell) từ các hệ thống khác.

Ta có thể thay đổi cho phép truy cập đến dịch vụ mạng xác định trên hệ thống.

- *SELinux*

SELinux (Security Enhanced Linux) dùng để giới hạn các hành động của các users và chương trình bởi việc áp đặt các chính sách bảo mật thông qua hệ điều hành.

- *Date và Time*: Đặt ngày giờ hệ thống:
- *Display*: Thiết lập màn hình
- *System User*: Tạo user cho hệ thống



2.2.4 Quá trình khởi động và kết thúc của UNIX

Như thông thường, khi một máy tính được khởi động, sau khi kiểm tra các thiết bị phần cứng gắn trên máy tính qua các chương trình kiểm tra ghi trong ROM, hệ điều hành được tải lên bộ nhớ. Công tác đầu tiên của hệ điều hành là kiểm tra các thiết bị ngoại vi và tải các chương trình điều khiển (driver) cần thiết lên bộ nhớ. Sau các công tác này, bắt đầu giai đoạn định hình hệ thống và mỗi hệ điều hành, thậm chí mỗi phiên bản của một hệ điều hành thực hiện một khác.

Các bước khởi động hệ thống:

B1. BIOS/POST

B2. MBR (GRUB hoặc boot loader khác)

B3. Kernel + initd

B4. Kết gắn (mount) hệ thống file root

B5. Tìm và khởi động tiến trình /sbin/init

init đọc tập tin /etc/inittab – nhận cấp chạy để sử dụng

B6. Khởi động các scripts rc trong /etc/rc.d

getty (giám sát đăng nhập) & bắt đầu phiên làm việc

2.2.4 Cấp chạy (runlevel)

- Tập tin đầu tiên mà hệ điều hành xem xét đến là /etc/inittab

Cấp chạy (run level) làm việc mặc định được quy định trong tập tin này.

Unix nói chung có 7 cấp chạy (run level) đánh số từ 0 đến 6.

Nội dung tập tin như sau:

```
# Default runlevel. The runlevels used by RHS are:
# 0 - halt (Do NOT set initdefault to this)
# 1 - Single user mode
# 2 - Multiuser, without NFS
# 3 - Full multiuser mode
# 4 - unused
# 5 - X11
# 6 - reboot (Do NOT set initdefault to this)
id:3:initdefault:
```

Linux có 7 cấp chạy được đánh số từ 0 đến 6. Trong đó:

Cấp chạy 0: Shutdown hệ thống.

Cấp chạy 1: Chế độ đơn người dùng (single user) và được dùng để sửa chữa lỗi hệ thống tập tin.

Cấp chạy 2: Chế độ đa người dùng mà không sử dụng hệ thống file mạng (NFS)

Cấp chạy 3: Chế độ đa người dùng đầy đủ

Cấp chạy 4: Chưa sử dụng

Cấp chạy 5: Chế độ đồ họa

Cấp chạy 6: Khởi động lại hệ thống

Tương ứng với các cấp chạy trên, trong thư mục /etc/rc.d có các thư mục rc0.d – rc6.d, chứa các tập tin khởi động trong từng cấp chạy (rc là viết tắt của run command). Thường các tập tin này là các shell script (tập hợp lệnh shell).

Toàn bộ các tập tin này quyết định cấu hình làm việc của một máy Linux sau khi hoàn thành quá trình khởi động. Việc khởi động hệ thống các dịch vụ cũng thực hiện thông qua cơ chế như đã miêu tả trên.

- Lệnh init cho phép chuyển giữa các mức của hệ thống.

Ví dụ:

+ Để chuyển hệ thống từ mức hiện hành qua mức 1 để sửa chữa.

```
[root@pascal /etc/rc.d]# init 1
```

+ Sau đó init 3 cho phép quay về mức 3 đa người dùng.

```
[root@pascal /etc/rc.d]# init 3
```

+ Shutdown hệ thống

```
[root@pascal /etc/rc.d]# init 0
```

2.3 Đăng nhập và sử dụng hệ thống Linux

2.3.1 Đăng nhập cục bộ

Sau khi boot máy, dấu nhắc sau sẽ xuất hiện:

```
Fedora Core release 2 (Tettnang)
```

```
Kernel 2.6.5-1.358 on an i686
```

```
linuxpc1 login: _
```

Password: [user enters password here]

-Nếu đăng nhập thành công, ta sẽ thấy dấu nhắc lệnh như sau:

```
[tom@linuxpc1 tom]$ _
```

Lưu ý: Nếu đăng nhập với user root ta sẽ thấy dấu nhắc lệnh như sau:

```
[root@linuxpc1 ~]# _
```

2.3.2 Đăng nhập vào Linux từ xa:

Ta có thể kết nối đến một server Linux từ bất kỳ máy tính nào như Windows 9x, Windows NT hoặc 2000, sử dụng một trong các tiện ích sau:

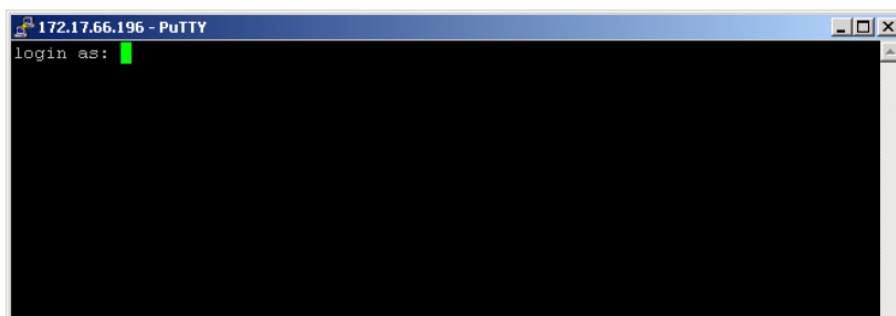
- *Telnet:* Ta có thể chạy Telnet từ dòng lệnh bởi việc gõ lệnh sau:

```
telnet <IP addrees>
```

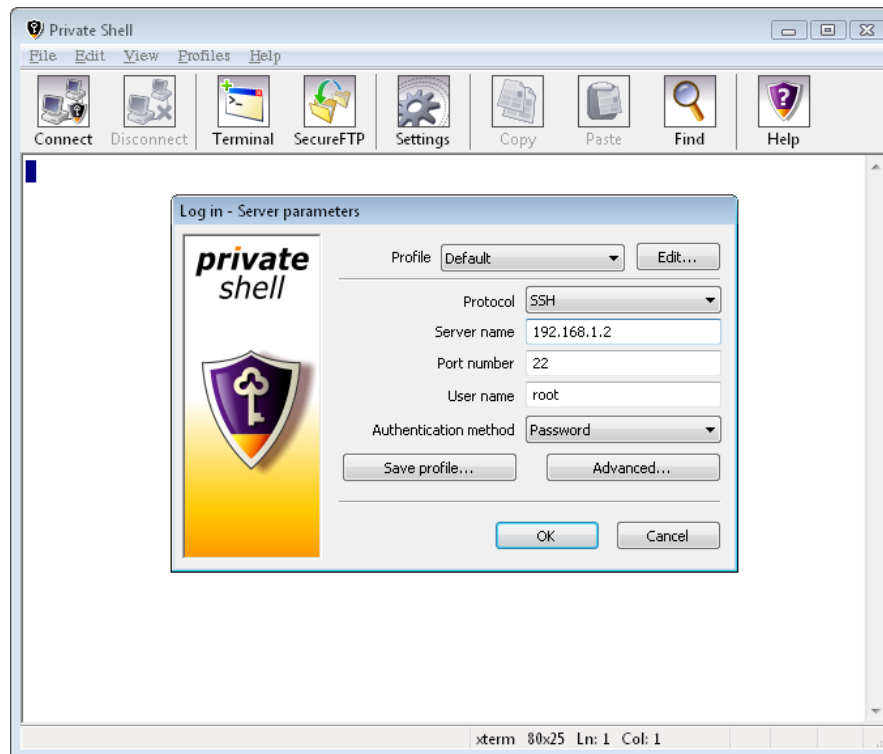
- *PuTTY:* Is a third-party program.

Cho phép tạo các kết nối bảo mật ssh (Secure SHell) từ các máy tính sử dụng hệ điều hành Windows. PuTTY có thể như một ứng dụng trên môi trường Windows mà không cần phải cài đặt.

PuTTY window



▪ *Private Shell (Business)*



Tính bảo mật người dùng Linux: Mật khẩu

- Linux đảm bảo rằng chỉ những người dùng được cấp quyền (authorized users) mới có thể truy xuất hệ thống.
- Linux bắt buộc phải có một mật khẩu kết hợp với một tên đăng nhập.
- Mật khẩu không được hiển thị trên màn hình trong khi chúng được nhập.
- Mọi người dùng có thể thay đổi mật khẩu sử dụng lệnh `passwd`.

Cú pháp của lệnh `passwd` là:

```
[user@linuxpc1 ~]$ passwd
```

Ví dụ:

```
[steve@linuxpc1 steve]$ passwd
Changing password for user steve
Changing password for steve
(current) UNIX password: [user enters a wrong password]
passwd: Authentication token manipulation error
[steve@linuxpc1 steve]$ _
```

Example

```
[steve@linuxpc1 steve]$ passwd
Changing password for user steve
Changing password for steve
(current) UNIX password: [user enters the old password]
New UNIX password: [user enters a new password]
Retype new UNIX password: [user enters the wrong new password]
Sorry, passwords do not match
New UNIX password: _
```

2.3.3 Một số lệnh thường dùng:

- *Xem ngày giờ hệ thống*

Người sử dụng có thể xem ngày giờ hiện hành của hệ thống sử dụng lệnh `date`.

```
[steve@linuxpc1 steve]$ date
Thu Aug 12 15:56:21 IST 2004
[steve@linuxpc1 steve]$ _
```

Các tùy chọn của lệnh `date`:

| Option | Description |
|---------------|---|
| %m | Displays month of the year (in digits) |
| %d | Displays day of the month (in digits) |
| %y | Displays year (last two digits) |
| %D | Displays date as mm/dd/yy |
| %H | Displays hour (00 to 23) |
| %M | Displays minutes (00 to 59) |
| %S | Displays seconds (00 to 59) |
| %T | Displays time as HH:MM:SS |
| %a | Displays abbreviated weekday (Sun to Sat) |
| %h | Displays abbreviated month (Jan to Dec) |

Ví dụ:

```
[steve@linuxpc1 steve]$ date "+%T"
22:47:45
```

```
[steve@linuxpc1 steve]$ date "+%y"
99
```

- *Lệnh thao tác trên màn hình*

Linux cho phép bạn một cách dễ thao tác trên màn hình với lệnh `clear` và `tput`.

+ *Lệnh `clear`*: Xóa màn hình terminal.

```
[steve@linuxpc1 steve]$ clear
```

+ *Lệnh `tput clear`*: Xóa màn hình terminal, và định vị con trỏ ở góc trên trái màn hình.

```
[steve@linuxpc1 steve]$ tput clear
```

+ *Lệnh `tput cup`*: Định vị con trỏ ở dòng cột xác định.

+ *Lệnh `tput smso`*: Đặt chế độ đảo màu video.

+ *Lệnh `tput rmso`*: Đặt màn hình trở lại chế độ bình thường.

- + *Lệnh `tput blink`*: Hiển thị chữ nhấp nháy.
- + *Lệnh `tput reset`*: Khởi tạo lại các thiết lập mặc định của màn hình.
- *Xem thời gian chạy và tải hệ thống (uptime)*

```
[steve@linuxpc1 steve]$ uptime
 7:00pm up 56 min, 6 users, load average: 3.51, 1.87, 1.01
```

Current time How long the system has been working Number of users already logged in Load average

Lệnh `uptime` được sử dụng để hiển thị thời gian hoạt động từ lúc hệ thống khởi động.

Lệnh cho kết quả và hiển thị trên một dòng gồm thời gian hiện hành, khoảng thời gian hệ thống đã chạy, số người dùng đã đăng nhập và tải trung bình của hệ thống (CPU utilization) cách đây 1, 5, và 15 phút tương ứng.

- *Lệnh xem trợ giúp*

Linux cung cấp 2 lệnh để xem tham khảo về các lệnh khác:

- + *Lệnh `man`*: Hiển thị các trang trợ giúp của một lệnh cụ thể.

```
[steve@linuxpc1 steve]$ man ls
```

- + *Lệnh `info`*: Xem thông tin chi tiết về một lệnh. Cú pháp của lệnh `info` là:

```
$ info [options] [menu item]
```

Ví dụ: Xem thông tin chi tiết về trình soạn thảo `emacs`:

```
$ info emacs
```

- + Chú ý: Ta có thể xem cách sử dụng của một lệnh trong linux (chi tiết về các tùy chọn và các tham số), bằng cách:

```
[user@linuxpc1 ~]$ <command> --help
```

- *Lệnh `who`*: xác định user nào hiện đang đăng nhập
- *Lệnh `hostname`*: Hiển thị tên máy tính đang làm việc.

Hệ thống lưu thông tin về tên máy trong tập tin `/etc/hosts`.

- *Lệnh `cal`*: Xem lịch tương ứng với tháng và năm chỉ định.

Cú pháp: `cal [-j] [month] [year]`

- *Lệnh `write`*: truyền thông điệp trực tiếp ra màn hình của người dùng khác đang cùng làm việc trên hệ thống.

Cú pháp: `write [username]`

Thông thường muốn trao đổi với người dùng khác, cần sử dụng lệnh `who`

Ví dụ:


```
$who
user1 tty17  Oct 15 10:20
user2 tty43  Oct 15  8:25
Tom  tty52  Oct 15 12:20
$ write tom
Hi ! Chuc mot buoi sang vui ve !
<Ctrl+D>
```

- *Lệnh mesg*: Từ chối/cho phép nhập thông điệp trên màn hình bởi lệnh write.

Cú pháp: `mesg [n] [y]`

Trong đó:

N (no): Từ chối nhận thông điệp từ người dùng khác.

Y (yes): Chấp nhận thông điệp từ người dùng khác.

- *Lệnh mail*: gửi/nhận thư điện tử giữa các người dùng song hoạt động theo chế độ offline.

Email được lưu trữ trên hệ thống Linux thường ở thư mục `/var/spool/mail`.

Mỗi user có một hộp thư là tên của user đó. Ví dụ tài khoản của user mars có hộp thư là `/var/spool/mail/mars`.

+ Để gửi hoặc nhận mail dùng lệnh:

```
$ mail <username hoặc địa chỉ mail>
```

Ví dụ:

```
$ mail tom
Subject: <gõ nội dung mail> <Ctrl+D>
Cc: <mail của user nhận khác>
```

+ Để nhận mail dùng lệnh:

```
$ mail
```

Sau khi gõ lệnh màn hình mail ở chế độ lệnh với dấu nhắc thường là dấu chấm hỏi (?), tại đây người dùng sử dụng các lệnh quản lý hệ thống mail của mình:

<n>: hiện thị mail có số thứ tự n

<space>: Hiện thị mail trước

+: hiện thị mail tiếp theo

d: xóa mail hiện hành và thoát

m <user>: chuyển tiếp mail cho người dùng khác

s <file>: lưu mail vào file

r: reply mail

d <n>: xóa thư có số thứ tự n

q Thoát

- *Kết thúc phiên làm việc Linux*

Lệnh exit hoặc logout dùng để kết thúc phiên làm việc Linux. Hệ thống hiện thị lại dấu nhắc login và khởi tạo cho phiên làm việc khác.

- *Khởi động lại hoặc thoát khỏi hệ thống:*

+ Lệnh reboot; khởi động lại máy

+ Lệnh thoát: halt hoặc shutdown [time] [message]

Ví dụ: shutdown +30 "Hệ thống sẽ dừng trong vòng 30 giây".

2.4 Cài đặt phần mềm trong Linux

2.4.1 Redhat Package Manager - rpm

Việc quản lý các phần mềm đi kèm với Linux có thể là một việc dễ dàng cũng như có thể trở nên vô cùng khó khăn phức tạp. Redhat đã phát triển một hệ thống quản lý phần mềm cho Linux có giao diện thân thiện và hiệu quả gọi là RPM. RPM là chương trình quản lý các package nó tự động làm các quá trình như cài đặt, nâng cấp, xoá và bảo trì phần mềm trong Linux. Do tính tiện dụng của nó, RPM được hỗ trợ bởi hầu hết các Linux khác như SuSe, Mandrake, Caldera, Corell. Ngày nay, RPM còn được dùng trên các hệ Unix khác như: OS/2, Solaris, SCO Unix, HP-UX, FreeBSD, NetBSD, Be OS,...

Lệnh rpm có rất nhiều tham số. Để xem một cách nhanh chóng danh sách các phần mềm có trong hệ thống ta dùng

```
root@starturn root]# rpm -qa
```

Có nghĩa là query all package. Để tìm chính xác package mà ta muốn biết xem đã có trong hệ thống hay chưa ta dùng kết hợp lệnh grep

```
[root@starturn root]# rpm -qa |grep mc
kernel-pcmcia-cs-3.1.31-9
mc-4.5.55-12
[root@starturn root]#
```

Một vài công cụ tiện ích khác cho rpm là dùng kết hợp với các tham số -i , -l,...

-i: Information

-l: list

Ví dụ:

```
[root@starturn root]# rpm -qi telnet
Name: telnet Relocations: (not relocateable)
Version: 0.17 Vendor: Red Hat, Inc.
Release: 23 Build Date: Tue 23 Jul 2002 09:05:54 AM EDT
Install date: Sun 08 Dec 2002 11:19:54 AM EST Build Host:
stripples.devel.redhat.com
Group: Applications/Internet Source RPM: telnet-0.17-23.src.rpm
Size: 91323 License: BSD
Signature: DSA/SHA1, Tue 03 Sep 2002 05:41:07 PM EDT, Key ID
219180cddb42a60e
Packager: Red Hat, Inc. <http://bugzilla.redhat.com/bugzilla>
Summary: The client program for the telnet remote login protocol.
```

Description:

Telnet is a popular protocol for logging into remote systems over the

Internet. The telnet package provides a command line telnet client.

```
[root@starturn root]#
```

- Cài đặt một gói phần mềm

Cú pháp: `rpm -ivh [package.rpm]`

Ví dụ: cài đặt Midnight Commander

```
[root@starturn root]# rpm -ivh mc-4.5.55-12.i386.rpm
```

```
warning: mc-4.5.55-12.i386.rpm: V3 DSA signature: NOKEY, key ID db42a60e
```

```
Preparing... ##### [100%]
```

```
1:mc ##### [100%]
```

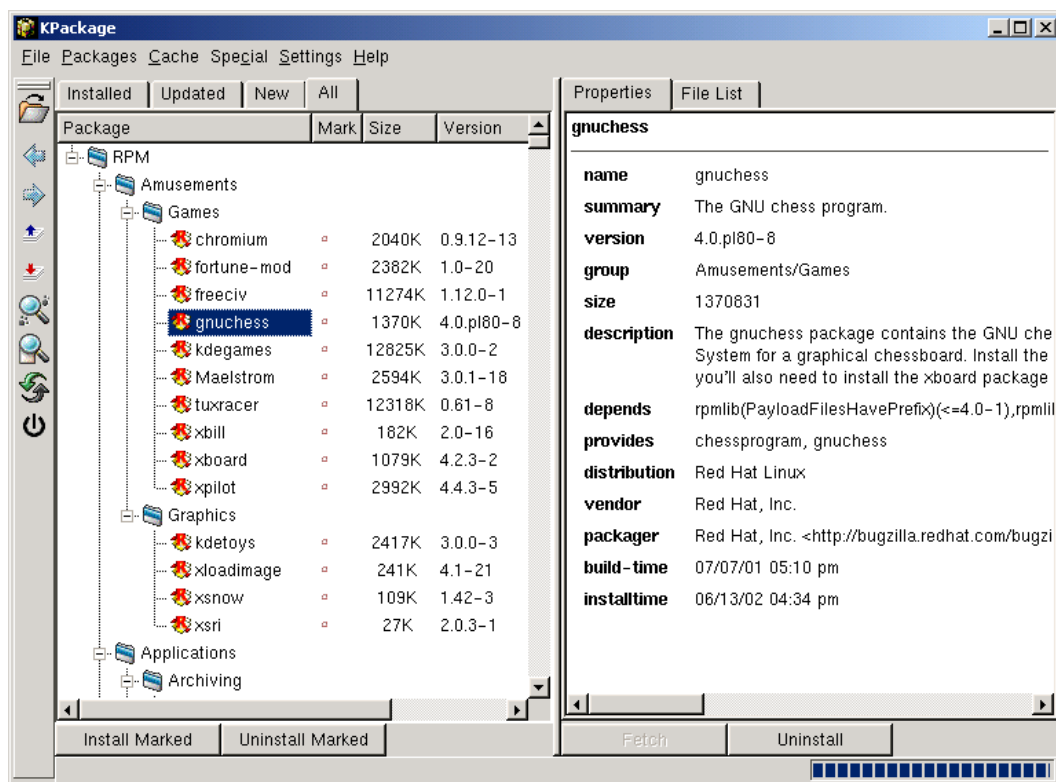
```
[root@starturn root]# rm -f mc-4.5.55-12.i386.rpm
```

- Xóa một phần mềm

Cú pháp: `rpm --erase [package.rpm]`

+ *Tiện ích kpackage*

Ngoài ra Linux còn có tiện ích kpackage trong môi trường KDE giúp ta có thể quản lý các package một cách hiệu quả với giao diện đồ họa rất dễ sử dụng



2.4.3 Cài đặt phần mềm trong Fedora với tiện ích yum

- Cấu hình YUM cho Fedora:

Chạy những lệnh sau để xóa file yum.conf cũ và thay bằng file mới:

```
rpm -Uvh http://www.fedorafaq.org/yum http://rpm.livna.org/livna-release-5.rpm
```

- Để xem danh sách các phần mềm có thể tải về:

```
[root@starturn root]# yum list available
```

- Để cài đặt phần mềm, dùng lệnh:

```
[root@starturn root]# yum install [package]
```

- Để cập nhật phần mềm, dùng lệnh:

```
[root@starturn root]# yum update [package]
```

- Để tìm kiếm một gói, dùng lệnh:

```
[root@starturn root]# yum search [package]
```

Nếu khi sử dụng yum xuất hiện cảnh báo NOKEY từ một gói RPM và lỗi chữ ký GPG, cách khắc phục:

Đăng nhập với tài khoản root thực hiện các lệnh:

```
rpm -ivh http://rpm.livna.org/livna-release-5.rpm
```

```
rpm --import http://rpm.livna.org/RPM-LIVNA-GPG-KEY
```

** Ví dụ cài đặt X-Unikey*

X-Unikey là tiện ích dùng để gõ tiếng Việt trên hệ thống Linux do Phạm Minh Long phát triển.

Để cài đặt X-Unikey, có thể cài đặt bằng tiện ích rpm với gói rpm cho X-Unikey hoặc có thể cài X-Unikey từ mã nguồn.

- Cài đặt từ gói rpm:

Download gói x-unkey-1.0.3b-FC4.i586.rpm

```
# rpm -ivh x-unkey-1.0.3b-FC4.i586.rpm
```

- Cài đặt từ mã nguồn:

```
$ tar xvjf x-unkey-1.0.3b-FC4.tar.bz2
```

```
$ cd x-unkey-1.0.3b-FC4
```

```
$ ./configure
```

```
$ make
```

```
$ su
```

```
password:
```

```
# make install
```

```
# exit
```

- Cấu hình sử dụng X-Unikey:

```
$ cat >> /home/[user]/.bashrc
```

```
export LANG=en_US.UTF-8
```

```
export XMODIFIERS="@im=unikey"
```

```
export GTK_IM_MODULE="unikey"
```

- Thoát khỏi X-Unikey dùng lệnh:

```
kill `pidof unikey`
```

❶ Câu hỏi bài tập

1. Cho biết các phương pháp cài đặt Linux? Việc triển khai cài đặt Unix/Linux trên máy ảo có những ưu nhược điểm gì?
2. Phân biệt giữa phân vùng đĩa chứa dữ liệu (data partition) và phân vùng hoán đổi (swap partition).
3. Cho biết Linux có bao nhiêu cấp chạy (runlevel)? Ý nghĩa của các cấp chạy đó là gì? Cấp chạy nào là mặc định khi khởi động Linux?
4. Cho biết các phương pháp đăng nhập vào sử dụng Linux? Và các phương pháp để cài đặt gói phần mềm trong Linux.

Chương 3. QUẢN LÝ THIẾT BỊ VÀ HỆ THỐNG TẬP TIN

Trong chương này chúng ta sẽ nắm về quy tắc quản lý thiết bị trong Unix/Linux, quy tắc đặt tên thiết bị. Sau đó, chúng ta sẽ nắm về hệ thống tập tin ext3 của Linux và nắm một số lệnh trong quản lý hệ thống tập tin

3.1 Quy tắc quản lý thiết bị

3.1.1 Quy ước tên

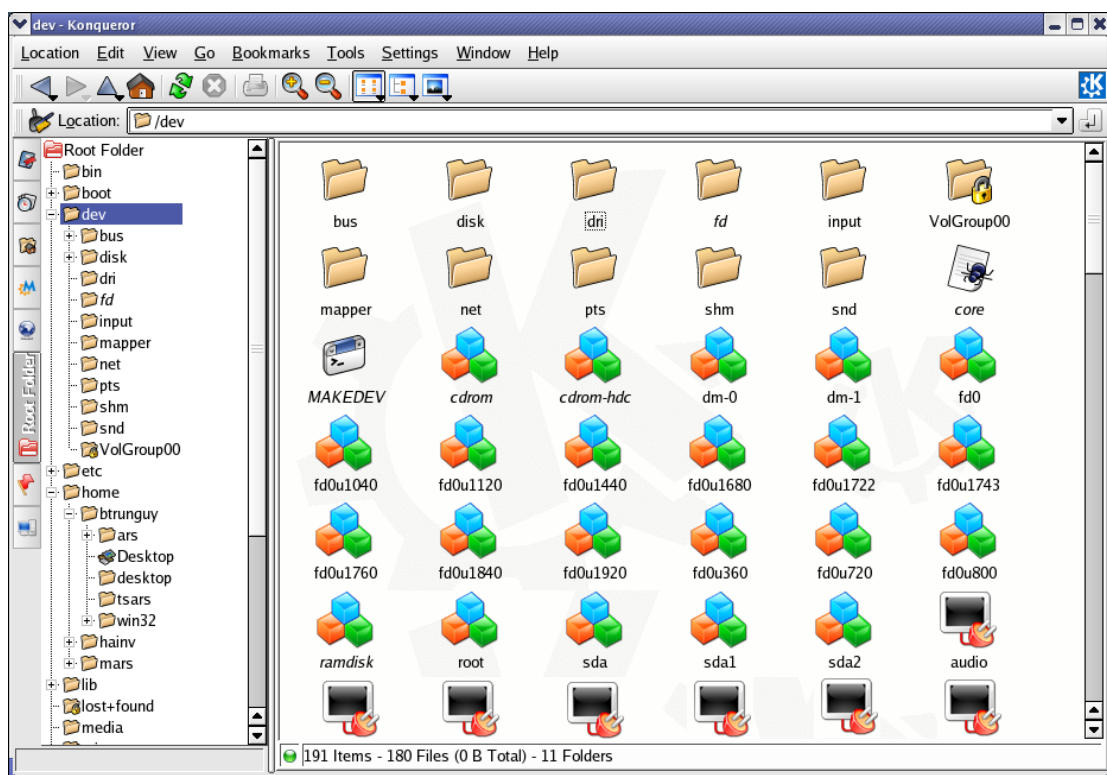
Linux xây dựng cơ chế truy xuất đến tất cả các loại đĩa và thiết bị đều ở dạng tập tin (tập tin thiết bị) và lưu trong thư mục /dev.

Linux quy ước đặt tên như sau:

- Ổ đĩa mềm: fd
- Ổ đĩa cứng vật lý thứ nhất: hda
- Ổ đĩa cứng vật lý thứ hai: hdb
- ...

Nếu đĩa cứng theo chuẩn SCSI thì gọi là: sda, sdb,...

Các thiết bị USB, Linux xem như là thiết bị SCSI (ví dụ nếu máy có một đĩa cứng SCSI thì usb sẽ là sdb1).



- Các phân vùng (partitions) được đánh theo số sau tên đĩa.

Ví dụ: hda1, hda2, sda1, sdb1, fd0 (ổ A), fd1 (ổ B)...

- Các phân vùng chính (primary) hoặc phân vùng mở rộng (extended) được đánh từ 1 đến 4

- Các phân vùng logic (nằm trong phân vùng mở rộng) đánh số từ 5 trở đi.

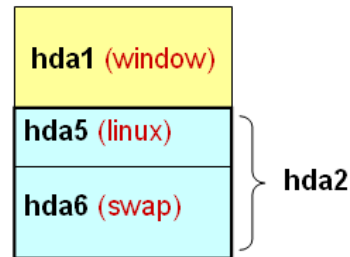
Ví dụ phân vùng đĩa cứng IDE

hda1: phân vùng chính

hda2: phân vùng mở rộng

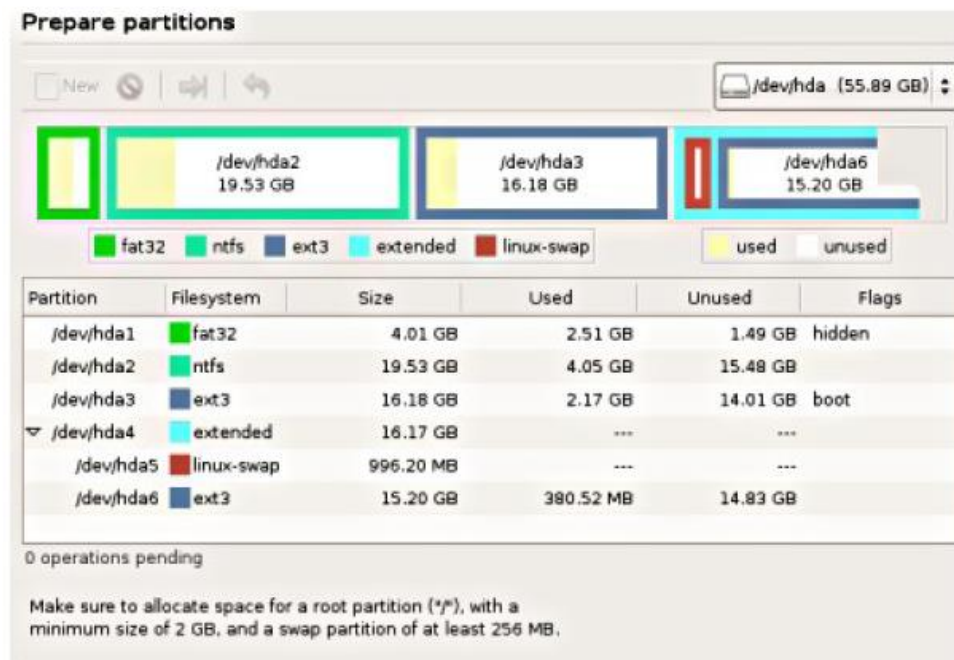
hda5: phân vùng logic

hda6: phân vùng logic



Chú ý: Nếu khi cài đặt Linux mà trước đó đã cài Window, thì Linux sẽ tự động cài đặt vào các phân vùng mở rộng.

Ví dụ phân vùng trong quá trình cài đặt Ubuntu:



▪ Cách truy xuất đĩa

Cũng tương tự như Window, trong Linux cũng có khái niệm đường dẫn (path). Tuy nhiên, có 2 điểm cần lưu ý:

- Sử dụng ký tự sổ trái (/) làm ký tự phân cách thư mục và tập tin.
- Không sử dụng ký tự ổ đĩa, mà dùng ký tự / ở đầu đường dẫn (thư mục gốc).

Ví dụ:

/usr/local/dev

/dev/hda

3.1.2 Kết gắn hệ thống tập tin

Khi khởi động hệ điều hành, Linux chỉ kết gắn cho phân vùng (nơi chứa nhân Linux) bằng ký tự / (thư mục gốc).

Các thông tin của phân vùng khác được Linux đặt trong thư mục /dev của cùng phân vùng chứa thư mục gốc.

Nếu muốn truy xuất hay chép dữ liệu vào các phân vùng như hda1, hda5, hda6,...hay ổ đĩa mềm fd0 ta cần thực hiện thao tác kết gắn (mount) thiết bị bằng lệnh mount.

Cú pháp: `mount [-t <fstype>] <device> <mount-point>`

Trong đó:

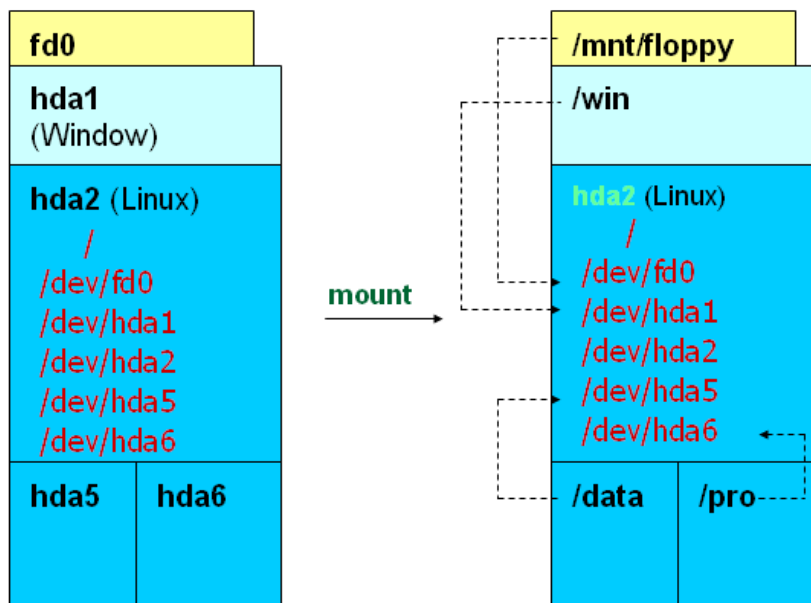
`fstype`: kiểu hệ thống tập tin

`device`: thiết bị (phân vùng đĩa cứng, cdrom hoặc đĩa mềm)

`mount-point`: vị trí để mount hệ thống tập tin (là một thư mục trong cây thư mục của Linux)

Kết gắn (mount) là gắn nội dung thiết bị (phân vùng) vào một thư mục (gọi là điểm kết gắn - mount point) để truy xuất.

Ví dụ:



Để truy xuất được các thiết bị ta phải thực hiện các lệnh:

```
# mount /dev/fd0 /mnt/floppy
```

```
# mount /dev/hda1 /win
```

```
# mount /dev/hda5 /data
```

```
# mount /dev/hda6 /pro
```

- Để tháo gỡ (unmount) kết gắn ta dùng lệnh:

```
umount <partition hoặc mount-point>
```

Chú ý:

- Chỉ có quyền root mới thực hiện được lệnh mount và umount.

- Điểm kết gắn (mount-point) phải là một thư mục trống, bởi dữ liệu hiện có trong đó sẽ bị ẩn mất sau khi đã mount một nội dung ở đó. Tuy nhiên, dữ liệu trong thư mục mount không bị xóa đi mà sẽ xuất hiện trở lại khi ta thực hiện tháo gỡ kết gắn.

Các ví dụ:

- Kết gắn thiết bị CDROM và các thiết bị chỉ đọc khác:

```
mount -t iso9660 [-o ro] <device> <mount-point>
```

Trong đó:

-o ro: báo cho hệ lõi biết phân vùng đang kết gắn là một thiết bị chỉ đọc.

- Kết gắn thiết bị USB:

```
# mount -t vfat /dev/sdb1 /home/gonzo
```

- Kết gắn đĩa mềm A:

```
# mount -t vfat /dev/fd0 /mnt/floppy
```

Ngoài ra đối với cdrom ta có thể đóng mở hoặc đựng cdrom ra bằng lệnh eject mà không cần thông qua lệnh umount.

- Lấy cdrom ra khỏi ổ đĩa

```
[root@blackboard root]# eject cdrom
```

- Đóng ổ cdrom lại

```
[root@blackboard root]# eject -t
```

Lưu ý: khi có một user hay tiến trình nào đang tham chiếu đến Cdrom thì ta không thể umount nó được. Hệ thống sẽ báo: device busy !

▪ Mount tự động với /etc/fstab

Hệ thống tập tin được OS Linux mount trong quá trình khởi động tuân theo các thông số ghi trong tập tin /etc/fstab (nếu nắm vững cú pháp của tập tin này, ta có thể thay đổi nó thông qua một chương trình soạn thảo văn bản text bất kỳ và có một kiểu khởi động hệ thống tập tin như mong muốn)

Định dạng các dòng trong /etc/fstab

```
<device> <mount> <fstype> <options> <dump> <fsck>
```

Trong đó:

device: thiết bị cần mount

mount-point: thư mục được mount

fstype: Kiểu của hệ thống tập tin

options: các tùy chọn: Default = mount khi khởi động, ro = read only, user nếu cho phép user mount hệ thống tập tin này...

dump-number: kết xuất hệ thống tập tin hay không?

fsck-number: có cần kiểm tra bởi fsck hay không?

Ví dụ:

```
[root@blackboard root]# less /etc/fstab
LABEL=/ / ext3 defaults 1 1
none /dev/pts devpts gid=5,mode=620 0 0
none /proc proc defaults 0 0
none /dev/shm tmpfs defaults 0 0
/dev/sda2 swap swap defaults 0 0
/dev/sdb1 /export ext3 defaults 0 0
/dev/cdrom /mnt/cdrom iso9660 noauto,owner,kudzu,ro 0 0
/dev/fd0 /mnt/floppy auto noauto,owner,kudzu 0 0
```

Trong đó:

Cột 1 (fs_spec): các trang thiết bị (device) cần mount

Cột 2 (fs_file): điểm treo (mount point)

Cột 3 (fs_vfstype): Kiểu của hệ thống tập tin,

Cột 4 (fs_mntops): các options. Default = mount khi khởi động, ro = read only, user nếu cho phép user mount hệ thống tập tin này...

Cột 5 (fs_freq): hiện thị (dumped) hay không hệ thống tập tin

Cột 6 (fs_passno): có cần kiểm tra hay không bởi fsck

3.2 Hệ thống tập tin Linux

3.2.1 Giới thiệu

Trong Linux không có khái niệm ổ đĩa. Sau quá trình khởi động, toàn bộ các thư mục và tập tin được kết gán và tạo thành một hệ thống tập tin thống nhất, bắt đầu từ gốc '/'

Linux hỗ trợ rất nhiều kiểu hệ thống tập tin như: ext, ext2, minix, msdos, vfat, smb, iso9660, sysv, vfs,...

Hệ thống tập tin đầu tiên Linux hỗ trợ là Minix cho phép tập tin có tên tối đa là 30 ký tự và kích thước không vượt quá 64MB.

Hệ thống tập tin đặc thù của Linux là ext, ext2,... Hiện nay các hệ thống Linux đều sử dụng hệ thống tập tin ext3.

Giới hạn của một số hệ thống file

| | MINIX | EXT | EXT2 |
|-------------------------|---------------|----------------|----------------|
| Max filesystem size | 64 MB | 2 GB | 4 TB |
| Max file size | 64 MB | 2 GB | 2GB |
| Maximum filename length | 30 Characters | 255 Characters | 255 Characters |
| Variable block Size | No | No | Yes |

3.2.2 Cấu trúc hệ thống tập tin

Với cấu trúc ext3, Linux tổ chức tập tin sử dụng một tập hợp bảng các nút thông tin gọi là "i-nodes".

Nút thông tin bao gồm:

| | |
|--------|----------|
| i-node | Filename |
|--------|----------|

Trong đó:

-i-node: là cấu trúc chứa các thông tin về tập tin

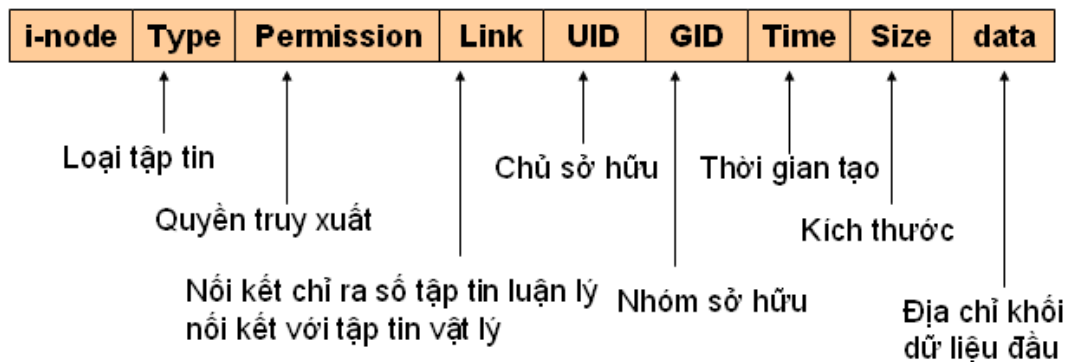
- Filename: Tên tập tin hoặc thư mục.

a. Các quy định về tên file

- Tên hợp lệ là một chuỗi bao gồm các ký tự:
 - + Chữ cái viết hoa (A to Z), chữ cái viết thường (a to z), chữ số (0 to 9)
 - + Dấu chấm (.), dấu gạch dưới (_), dấu phẩy (,).
 - + Không nên chứa khoảng trắng hoặc các ký đặc biệt như & * \ | [] { } \$ < > () # ? ' " / ; ^ ! ~ %
 - + Không trùng tên với tập tin hoặc thư mục đã có
 - + Tránh đặt tên tập tin hoặc thư mục trùng với các lệnh UNIX.
- Một số ký hiệu đặc biệt trong tên file Linux:
 - + Tên tập tin có thể không có phần mở rộng, nhưng có thể nếu cần.
 - + Tên file bắt đầu bằng dấu chấm '.' là các file ẩn
 - + Dấu . chỉ thư mục hiện hành
 - + Dấu .. chỉ thư mục cha
 - + Dấu ~ chỉ thư mục chủ (home) của user
 - + Dấu ~username chỉ thư mục chủ của user xác định bởi username.

b. Cấu trúc i-node

Mỗi i-node chứa các thông tin liên quan đến một tập tin bao gồm:



- Linux có 3 kiểu file:

File bình thường: là tập hợp các thông tin (text hoặc binary)

File thư mục: chứa danh sách các tên có thể truy cập đến.

File đặc biệt: là các file liên quan đến thiết bị ngoại vi (như file kiểu b, c, l,... hoặc file cơ chế truyền tin như file kiểu p, s,...)

- Người sử dụng khi truy xuất các tập tin trong Linux được điều khiển bởi "quyền truy cập" (trường permission trong i-node).

Linux quy định 3 loại quyền chính:

- + Quyền đọc (r): Cho phép xem nội dung tập tin
- + Quyền ghi (w): Cho phép sửa đổi hoặc thêm mới nội dung tập tin.

+ Quyền thực hiện (x): Cho phép thực hiện tập tin.

Nếu là thư mục thì:

Quyền đọc là quyền liệt kê nội dung của thư mục (r),

Quyền ghi là thêm hay loại bỏ tập tin khỏi thư mục (w)

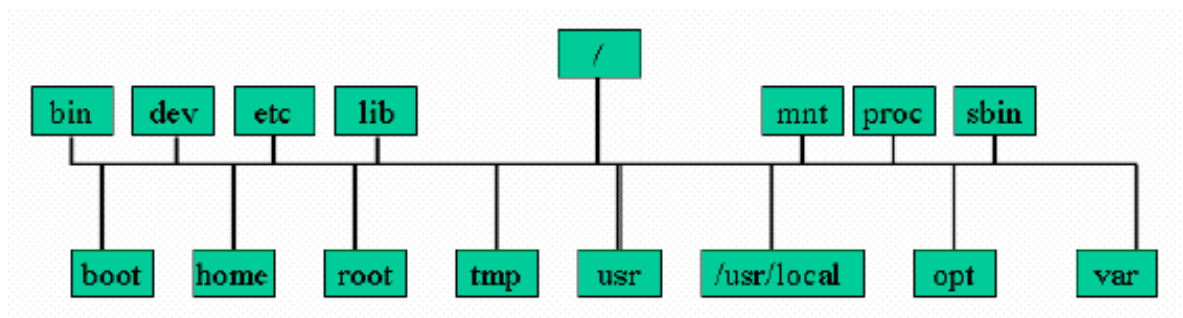
Quyền thực hiện cho phép tìm tập tin hay thư mục con trong thư mục (x).

c. Tổ chức thư mục Linux

Hệ thống tập tin Linux được tổ chức theo cây thư mục.

- Cây bắt đầu với thư mục gốc “/” và các thư mục con.
- Các files nằm trong các thư mục.

- Các thư mục cơ sở (thư mục hệ thống)



- /bin: chứa các lệnh của shell và hầu hết các lệnh thông dụng của Linux.

Ví dụ: [root@btusoft ~]# ls /bin

```

alsacard      dbus-send     grep          mkdir
alsaunmute    dd            gtar          mknod
arch           df            gunzip        mktemp
awk            dmesg         gzip          more
basename      dnsdomainname hostname       mount
bash           doexec        igawk         mountpoint
cat            domainname    ipcalc        mv
chgrp          dumpkeys      kbd_mode      netstat
chmod          echo          kill          nice
chown          ed            ksh           nisdomainname
cp             egrep         link          pgawk
cpio           env           ln            ping
csh            ex            loadkeys     ping6
cut            false         login         ps
date           fgrep         ls            pwd
dbus-cleanup-sockets gawk         mail          red
dbus-daemon    gettext       mailx         rm
[root@btusoft ~]#
  
```

- /usr/bin: chứa các lệnh mở rộng, chương trình tiện ích của Linux cũng như các ứng dụng.
- /dev: (viết tắt của devices) gồm các driver cần thiết để điều khiển các thiết bị ngoại vi như ổ đĩa, máy in, và thiết bị đầu cuối. Mỗi tập tin trong thư mục này là 1 liên kết tới 1 thiết bị.

Ví dụ: [root@btusoft ~]# ls /dev

```

adsp          dvdwriter-hdc  initctl       par0           ram6           tty1
agpgart       fd             input         parport0       ram7           tty10
audio         fd0            kmsg          parport1       ram8           tty11
bus           fd0u1040       log           parport2       ram9           tty12
cdrom         fd0u1120       loop0         parport3       ramdisk        tty13
cdrom-hdc     fd0u1440       loop1         port           random         tty14
cdrw          fd0u1680       loop2         ppp            root           tty15
cdrw-hdc      fd0u1722       loop3         ptmx           rtc            tty16
cdwriter      fd0u1743       loop4         pts            sda            tty17
cdwriter-hdc  fd0u1760       loop5         ram            sda1           tty18

```

- /etc: chứa các tập tin cấu hình của hệ thống như danh sách người dùng và mật khẩu.
- /home: chứa các thư mục chủ của người dùng.
- /tmp: thư mục chứa tập tin tạm thời.
- /var: chứa tập tin thường xuyên thay đổi.
- /lib : các thư viện chia sẻ cho các đoạn trình (routine) trong /bin và /usr/sbin. Cũng chứa các module của nhân.
- /mnt: điểm mount cho hệ thống file bên ngoài
- /proc: thông tin về nhân.
- /var/www: Vị trí các trang HTML cho web server.

3.2.3 Thao tác trên hệ thống tập tin

3.2.3.1 Các lệnh thường dùng

Có ba thành phần trong cú pháp lệnh:

<command-name> [options] [arguments]

Trong đó:

command-name: tên lệnh, cho biết hệ thống sẽ làm gì

options: các tùy chọn, lệnh sẽ thực hiện như thế nào. Thường các options trong Linux bắt đầu với dấu trừ (-) thay vì dấu sổ trái (/) như DOS. Ví dụ: ls -a

arguments: các đối số, lệnh sẽ áp dụng ở đâu

Ta có thể không cần options hoặc arguments, phụ thuộc vào câu lệnh.

a. Các lệnh làm việc với thư mục

- *Tạo thư mục mới*

Cú pháp: mkdir [paths]

Trong đó:

paths là tên các thư mục cần tạo.

Ví dụ:

```
$ mkdir baitap
```

Ta có thể tạo nhiều thư mục bằng một lệnh.

Ví dụ:

```
$ mkdir baitap docs
$ mkdir -p tailieu/linux
```

Ta có thể sử dụng lệnh `tree` để xem cây thư mục vừa tạo.

- *Thay đổi thư mục làm việc*

Cú pháp: `cd <path>`

Trong đó:

`path` chứa thư mục muốn chuyển đến.

`..`: yêu cầu chuyển đến thư mục hiện hành.

`...`: chuyển đến thư mục cha.

Ví dụ:

```
$ cd baitap
$ cd /home
$ cd ~
```

- *Xem nội dung thư mục*

Cú pháp: `ls [options] [path]`

Trong đó:

`path`: là danh sách tên tập tin hay thư mục.

Các options có thể là:

-a: Liệt kê tất cả các mục, bao gồm các thành phần bắt đầu với dấu chấm (các file ẩn, “.”, “..”)

-d: Nếu đối số là một thư mục thì không liệt kê nội dung của thư mục đó

-l: Liệt kê theo dạng thức đầy đủ, kích thước tập tin, người tạo ra, các quyền người sử dụng,...

-F: Hiển thị kiểu file (/, *, @)

Ví dụ:

```
$ ls -lF
```

- *Di chuyển thư mục*

Cú pháp: `mv <source> <destination>`

Trong đó:

`source` thư mục nguồn – thư mục cần di chuyển.

`destination` thư mục đích – thư mục chuyển đến.

Ví dụ:

```
$ mkdir ctrinh
$ mv ctrinh baitap
```

- *Xoá thư mục*

+ Xóa thư mục rỗng:

Cú pháp: `rmdir <path>`

Trong đó:

`path` là tên những thư mục cần xóa.

Ví dụ: `rmdir /home/baitap`

+ Xóa thư mục không rỗng:

Cú pháp: `rm -r[i] <dirs>`

Ví dụ: `rm -r /home/doc`

b. Các lệnh làm việc với tập tin

▪ *Tạo tập tin*

+ Bằng lệnh `touch`: cho phép tạo ra tập tin rỗng

Cú pháp: `touch filename`

+ Bằng lệnh `cat`: cho phép tạo ra tập tin

Cú pháp: `cat > filename`

Ví dụ:

```
[user@srv ~]$ touch data.txt
```

```
[user@srv ~]$ cat > bail.txt
```

```
Bai tap linux
```

```
<Ctrl+D>
```

▪ *Xem nội dung tập tin*

+ Xem nội dung tập tin văn bản (text)

Sử dụng một trong 3 lệnh sau:

```
cat filename
```

```
more filename
```

```
less filename
```

Ví dụ:

```
$ cat data.txt
```

```
$ more bt1.txt
```

```
$ less bt2.txt
```

+ Xem nội dung tập tin nhị phân (binary)

Cú pháp: `od <Tên tập tin>`

Ví dụ: `$ od baitap.a`

▪ *Nối nội dung tập tin*

Chuyển hướng nối kết quả vào một tập tin:

```
$ cal 6 2006 > june.t
```

```
$ cal 7 2006 > july.t
```

```
$ cal 8 2006 > august.t
```

```
$ cat june.t july.t august.t > summer.t
```

Linux cũng cho phép thêm thông tin vào cuối của một tập tin bằng phép chuyển hướng “>>”.

```
$ cal 9 2006 >> summer.t
```

▪ Sao chép tập tin

Cú pháp: `cp [options] [<tập tin nguồn>] [<tập tin đích>]`

Trong đó: các options có thể:

-r: sao chép cả thư mục.

-u: chỉ sao chép nếu tập tin nguồn mới hơn tập tin đích.

Lỗi thường gặp

+ No such file or directory: không tìm thấy tập tin nguồn

+ Permission denied: không có quyền truy xuất tập tin nguồn hoặc tạo tập tin đích

Ví dụ:

```
[user@srv ~]$ cd baitap
```

```
[user@srv ~]$ vi tho.txt
```

```
[user@srv ~]$ mv tho.txt baitho.doc
```

```
[user@srv ~]$ ls
```

```
baitho.doc ctrinh hello.c ltc perl
```

```
[user@srv ~]$ cp baitho.doc ~/document
```

Sao chép tất cả các tập tin vào một thư mục

```
[user@srv ~]$ cp * <directory>
```

▪ Di chuyển/đổi tên

Cú pháp:

```
mv <tập tin nguồn> <thư mục đích>
```

```
mv <tập tin nguồn> <tên mới của tập tin>
```

Ví dụ: `[user@srv ~]$ mv summer.t /usr/local/dev`

▪ Tìm kiếm một tập tin

Lệnh find cho phép tìm kiếm một hay nhiều tập tin trong một cây thư mục.

+ Tìm theo tên:

```
find <path> -name <filename>
```

+ Tìm theo số i-node của tập tin:

```
find <path> -inum <number>
```

+ Tìm theo tên người sở hữu:

```
find <path> -user <username>
```

+ Tìm theo kiểu hoặc kích thước tập tin:

```
find <path> [-type <Tname>] [-size <±num>]
```


Để tránh các thông báo lỗi đũa ra màn hình, có thể đổi hướng đầu ra lỗi chuẩn (stderr) tới một tập tin rỗng (/dev/null):

```
find / -name <filename> - print 2> /dev/null
```

Ví dụ:

```
[user@srv ~]$ find / -name ttyc2d1 - print 2>/dev/null  
/dev/ttyc2d1
```

```
[user@srv ~]$ ls -i /unix
```

```
2810 -r-xr--r-- 2 bin bin 508516 Mar 10 1989 /unix
```

```
[user@srv ~]$ find / -inum 2810 - print 2>/dev/null
```

▪ *Xóa tập tin*

Xoá (remove) danh sách tập tin. Chú ý là tập tin sẽ không khôi phục được sau khi xóa

Cú pháp: `rm [options] [<danh sách tên tập tin>]`

Trong đó: các options có thể là:

-f: xóa l ập t ức (không đũa ra các nh ắc nh ở)

-r: xóa c ả thư m ục

-d: Xóa liên k ết (link)

▪ *Nén/giải nén tập tin: zip/unzip*

+ Nén một hoặc nhiều tập tin

Cú pháp: `zip [<tập tin zip>] [<tập tin nguồn>]`

Ví dụ: `[mars@srv /]$ zip data.zip data.txt`

+ Giải nén tập tin

Cú pháp: `unzip [options] [<tập tin zip>] [-d] [<thư mục>]`

Trong đó:

-d chỉ tên thư mục chứa nội dung sau khi giải nén

Ví dụ: `[mars@srv /]$ unzip data.zip -d /home/stus/temp`

▪ *Nén/giải nén tập tin với tar*

Lệnh tar cho phép đóng gói một hệ thống tập tin thành một tập tin với phần đuôi.tar

Lệnh gzip/gunzip cho phép nén (compact)/giải nén một tập tin.

Thông thường, để lưu trữ, người ta tar các dữ liệu, rồi sau đó gzip tập tin kết quả của tar. Quá trình phục hồi làm theo quy trình ngược lại.

+ Đóng gói/bung gói tar

Cú pháp: `tar [options] <destination.tar> <source>`

Trong đó: các options có thể là:

-c: T ạo file tar.

-v: In thông tin quá trình đóng gói

-f: Tập tin

-x: Bung (phục hồi) gói tar

-z: Nén/Giải nén

+ Để tạo ra một lưu trữ tar:

Cú pháp: `tar -cvf <destination.tar> <source>`

Ví dụ: `tar -cvf dulieu.tar dulieu`

+ Để phục hồi lưu trữ:

Cú pháp: `tar -xvf <destination.tar>`

Ví dụ: `tar -xvf dulieu.tar`

▪ *Nén/giải nén tập tin gzip*

Dùng gzip để giải nén và giải nén tập tin kiểu tar.gz

Cú pháp: `gzip [options] <tập tin kiểu tar>`

Trong đó: các options có thể là:

-d: Giải nén

Ví dụ:

`$ gzip data.tar` # Tạo ra tập tin data.tar.gz

`$ gzip -d data.tar.gz` # Giải nén thành data.tar

Hoặc:

`$ gunzip data.tar.gz` # Giải nén thành data.tar

+ Có thể dùng tar để bung và giải nén tập tin tar.gz:

`$ tar -xvzf data.tar.gz` # Tạo ra tập tin data.txt

3.2.3.2 Quyền truy xuất tập tin/thư mục

Linux là một hệ điều hành multitasking và multiuser, nhiều người có thể cùng sử dụng một máy Linux và một người có thể cho chạy nhiều chương trình khác nhau.

- Có hai vấn đề:

+ Quyền sở hữu các dữ liệu trên đĩa.

+ Phân chia tài nguyên hệ thống như CPU, RAM... giữa các tiến trình.

Tất cả các tập tin và thư mục của Linux đều có chủ sở hữu và quyền truy nhập.

Để truy xuất tập tin, Linux cho phép người sử dụng xác định 3 cấp sở hữu và 6 cấp xác lập quyền.

▪ Ba cấp sở hữu:

+ Owner: chủ sở hữu tập tin hoặc thư mục.

+ Group: nhóm sở hữu.

+ Others: những người dùng khác

▪ Sáu cấp xác lập quyền:

+ Read (r): quyền đọc

- + Write (w): quyền ghi
- + Execute (x): quyền thực hiện
- + SetUID (s): chạy chương trình với tư cách người dùng khác.
- + SetGID (S): chạy chương trình với tư cách nhóm khác.
- + Sticky (t): ngăn ngừa user khác xóa tập tin trong thư mục.

Trong đó:

- + Quyền đọc cho phép xem nội dung của tập tin và thư mục.
- + Quyền ghi cho phép thay đổi nội dung hay xóa tập tin.
- + Đối với thư mục, quyền ghi cho phép tạo, xóa hay thay đổi tên các tập tin trong thư mục không phụ thuộc vào quyền cụ thể của tập tin.
- + Quyền thực hiện cho phép chạy chương trình.

a. Xem quyền trên tập tin/thư mục

Để xem quyền được gán trên tập tin và thư mục ta dùng lệnh:

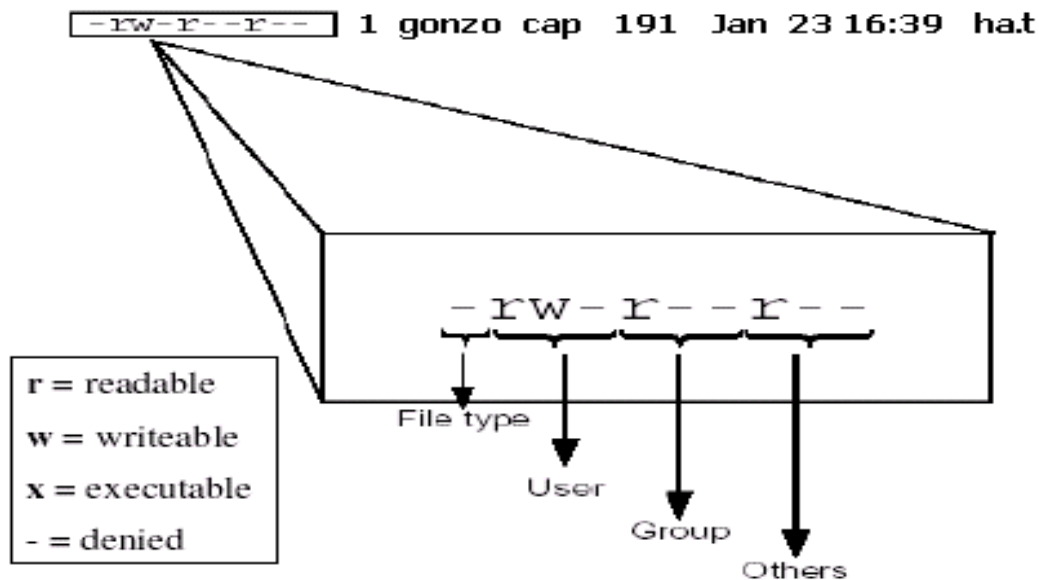
```
$ ls -l [tập tin / thư mục]
```

Ví dụ:

```
$ ls -l
```

```
-rw-r--r-- 1 gonzo cap 191 Jan 23 16:39 ha.t
drw-r--r-- 1 gonzo cap 191 Jan 23 16:39 kcنت
```

Ý nghĩa các cột



Cột 1: quyền truy cập tập tin

Cột 2: chỉ số liên kết (link) của tập tin, thư mục

Cột 3, 4: chủ sở hữu và nhóm sở hữu

Cột 5: kích thước của tập tin

Cột 6: thời gian thay đổi sau cùng

Cột 7: tên tập tin hay thư mục

Chú ý ký tự đầu tiên của quyền cho biết kiểu của tập tin:

‘-’: cho biết đó là một tập tin bình thường.

‘d’: là một thư mục.

‘c’: cho thiết bị ngoại vi dạng ký tự (như bàn phím).

‘b’: cho thiết bị ngoại vi dạng block (như ổ đĩa cứng).

‘p’: tập tin truyền dẫn pipe

‘l’: tập tin liên kết

Các ví dụ

-rwx-----: File được read/write/execute chỉ bởi chủ sở hữu.

dr-xr-x---: Thư mục được read/execute bởi chủ sở hữu và nhóm.

-rwxr-xr-x: File được read/write/execute bởi chủ sở hữu, và read/execute bởi các thành viên nhóm và user khác.

-rwxrw----: File được read/write/execute bởi chủ sở hữu và read/write bởi nhóm.

drwxr-x—x: Thư mục được read/write/execute bởi chủ sở hữu, read/execute bởi nhóm và execute bởi các user khác.

dr-x-w-r--: Thư mục được read/execute bởi chủ sở hữu, và write bởi các thành viên nhóm và read bởi các user khác.

b. Thay đổi quyền truy xuất

Để thay đổi quyền truy cập lập trên thư mục hoặc tập tin dùng lệnh chmod.

Cú pháp: `chmod <permission> <filename>`

Trong đó:

permission: xác định quyền sẽ được thiết lập cho file hoặc thư mục, có thể là dạng ký hiệu hoặc là dạng số bát phân.

filename: tập tin hoặc thư mục

Lưu ý: chỉ có chủ sở hữu file/thư mục hoặc root mới có thể sử dụng lệnh này để thay đổi quyền.

▪ Thay đổi quyền truy xuất dạng ký hiệu:

Sử dụng các ký hiệu sau để chỉ chủ sở hữu:

u: chủ sở hữu của file

g: nhóm sở hữu file

o: những người dùng khác

a: tất cả các user

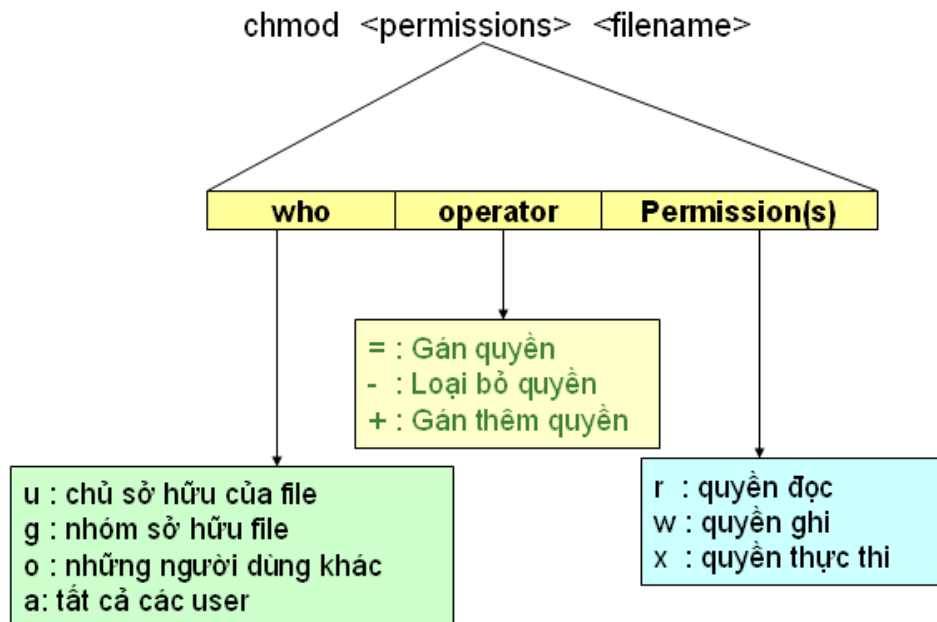
Các ký hiệu để gán hoặc cấp quyền:

+: Thêm quyền

-: Bỏ quyền

=: Gán quyền

Cú pháp sử dụng:



Ví dụ:

+ Bỏ quyền read của nhóm:

```
$ chmod g-r tho.txt
(-rw-r--r--  ->  -rw----r--)
```

+ Thêm quyền execute đối với chủ sở hữu, và quyền read cho nhóm và các user khác:

```
$ chmod u+x,go+r tho.txt
(-rw-----  ->  -rwxr--r-- )
```

+ Gán quyền read và write cho mọi user:

```
$ chmod a=rw dante
(-rw-rw-rw-)
```

▪ Thay đổi quyền truy xuất bằng số:

Cú pháp sử dụng:

```
$ chmod octal filename
```

Lệnh chmod có thể sử dụng các giá trị bát phân cho đối số octal như sau:

| Octal Value | Permissions |
|-------------|-------------|
| 4 | Read |
| 2 | Write |
| 1 | Execute |

Ví dụ: một tập tin với quyền 751 có nghĩa là:

Chủ sở hữu có quyền read, write, và execute (bằng 4+2+1=7),

Nhóm sở hữu có quyền read và execute (bằng 4+1=5),

Những user khác có quyền execute (bằng 1).

Với mỗi số từ 0 đến 7 đều tương ứng với một tổ hợp duy nhất các quyền truy nhập tập tin. Bảng sau mô tả mỗi số bát phân tương ứng quyền truy cập:

| Octal Value | Permission Sets |
|-------------|-----------------|
| 7 | r w x |
| 6 | r w - |
| 5 | r - x |
| 4 | r - - |
| 3 | - w x |
| 2 | - w - |
| 1 | - - x |
| 0 | - - - |

Ví dụ

+ Cho phép chủ sở hữu, nhóm, và các user khác chỉ có quyền read và execute:

```
$ ls -l dante
```

```
-rw-rw-rw- 1 root root 2 Jun 11 11:54 dante
```

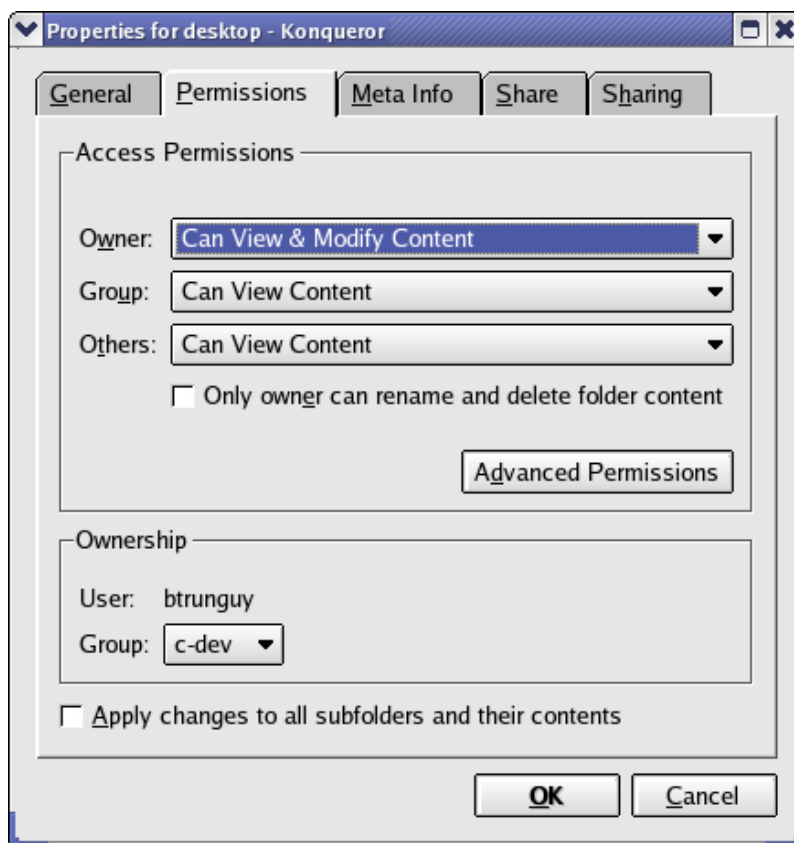
```
$ chmod 555 dante
```

```
$ ls -l dante
```

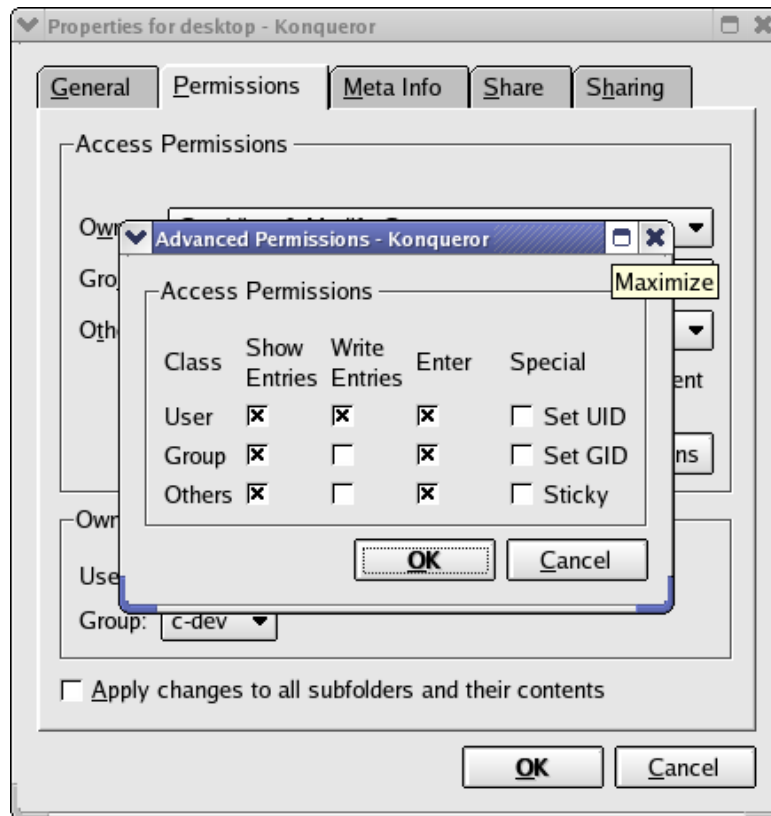
```
-r-xr-xr-x 1 root root 2 Jun 11 11:54 dante
```

▪ *Thiết lập quyền ở chế độ GUI*

Nhấp phải trên thư mục hoặc tập tin, chọn Properties



Chọn Advance Permissions:



c. Quyền truy cập ngầm định

Quyền ngầm định là các quyền được tự động gán cho tập tin hoặc thư mục khi nó được tạo.

Giá trị khởi tạo của quyền ngầm định xác định bởi hệ thống khi tạo tập tin là 666 (rw-rw-rw-).

Giá trị khởi tạo của quyền ngầm định xác định bởi hệ thống khi tạo thư mục là 777 (rwxrwxrwx).

▪ Mặt nạ che quyền

Mặt nạ che quyền là quyền kết hợp với giá trị quyền ngầm định để điều khiển quyền được gán cho các tập tin và thư mục khi nó mới được tạo.

Lệnh umask cho phép đặt hoặc hiển thị mặt nạ che quyền.

Hiển thị mặt nạ che quyền dùng lệnh: `umask`

Đặt mặt nạ che quyền: `umask [octal]`

Ví dụ:

```
$ umask 752
```

Khi một tập tin hay thư mục được tạo ra, quyền truy cập sẽ được xác định bởi giá trị quyền mặc định bỏ bớt bởi các quyền hiển thị bằng umask.

Giá trị mặc định của các quyền thường được gán mỗi khi người sử dụng login vào hệ thống thông qua các tập tin khởi tạo biến môi trường như .bash_profile, .bashrc.

Theo quan điểm bảo mật hệ thống, giá trị của umask 026 là tốt nhất, nó cho người cùng nhóm có quyền đọc và không cho quyền nào với những người khác.

▪ Cách sử dụng umask:

| | | |
|----------------------------------|-------------------|---------|
| ▪ Default file permission | r w - r w - r w - | (666) |
| umask of 022 | - - - - w - - w - | (022) |
| Resulting file permission | r w - r - - r - - | (644) |

| | | |
|---------------------------------|-------------------|---------|
| ▪ Default dir permission | r w x r w x r w x | (777) |
| umask of 022 | - - - - w - - w - | (022) |
| Resulting dir permission | r w x r - x r - x | (755) |

Lưu ý:

| | | |
|---------------------------|-------------------|---------|
| ▪ Default file permission | r w - r w - r w - | (666) |
| umask of 123 | - - x - w - - w x | (123) |
| Resulting file permission | r w - r - - r - - | (644) |

(not 543!)

| | | |
|--------------------------|-------------------|---------|
| ▪ Default dir permission | r w x r w x r w x | (777) |
| umask of 123 | - - x - w - - w x | (123) |
| Resulting dir permission | r w - r - x r - - | (654) |

d. Các quyền đặc biệt

Có ba quyền đặc biệt đối với các file khả thi và các thư mục công cộng:

- + Quyền UserID (suid)
- + Quyền GroupID (sgid)
- + Quyền Sticky bit (bit dính)

▪ *Quyền UserID (suid)*

Cho phép chạy chương trình với tư cách người dùng khác. Chỉ có user root và chủ sở hữu mới có thể đặt quyền suid trên file bằng cách sử dụng lệnh chmod và giá trị bát phân +4000 hoặc ký hiệu u+s.

Quyền suid hiển thị chữ “s” trong cột execute của chủ sở hữu. Nếu không phải là file khả thi, thì lệnh ls sẽ hiển thị chữ “S”.

Cú pháp:

```
# chmod 4755 <file khả thi>
# chmod u+s <file khả thi>
```


Ví dụ: chương trình passwd đã được thiết lập để cho user bình thường chạy với tư cách root.

```
$ ls -l /usr/bin/passwd
-r-s--x--x 1 root root 21944 Feb 12 2006 /usr/bin/passwd
```

▪ Quyền GroupID (sgid)

Cho phép chạy chương trình với tư cách nhóm khác. Chỉ có user root và chủ sở hữu có thể đặt quyền sgid trên file bằng cách sử dụng lệnh chmod và giá trị bát phân +2000 hoặc ký hiệu g+s.

Quyền sgid hiển thị chữ “s” trong cột execute của nhóm sở hữu. Nếu không phải là file khả thi, thì lệnh ls sẽ hiển thị chữ “S”.

Cú pháp:

```
# chmod 2755 <file khả thi>
# chmod g+s <file khả thi>
```

▪ Quyền Stick Bit

Nhằm bảo vệ các files trong một thư mục được sử dụng công cộng. Nếu thư mục có bit dính được đặt, thì:

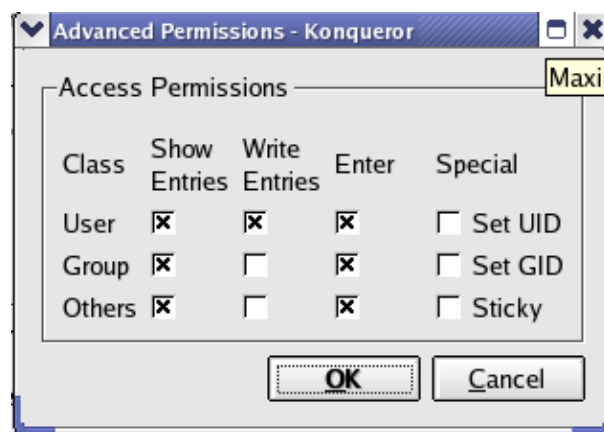
- + Chỉ có chủ sở hữu và root mới có thể xóa các files
- + Chủ sở hữu có quyền ghi nội dung vào thư mục

Quyền sticky bit hiển thị chữ “t” trong cột execute của các user khác. Nếu không phải là file khả thi, lệnh ls sẽ hiển thị chữ in hoa “T”

Để đặt quyền bit dính cho thư mục ta dùng lệnh:

```
# chmod +t <thư mục>
```

- Thiết lập quyền đặc biệt ở chế độ GUI:



e. Thay đổi chủ sở hữu tập tin/thư mục

- Để thay đổi chủ sở hữu của file hoặc thư mục ta dùng lệnh chown.

Cú pháp: chown <user> <dir or file>

Ví dụ:

```
(-rwxr-sr-x 1 root root 9 Jul 2 2003 /usr/cate)
```

```
# chown mars /usr/cate
(-rwxr-sr-x 1 mars root 9 Jul 2 2003 /usr/cate)
```

- Thay đổi nhóm sở hữu

Để thay đổi nhóm sở hữu của các file hoặc thư mục ta dùng lệnh chgrp.

Cú pháp: chgrp <group> <dir or file>

Ví dụ:

```
(-rwxr-sr-x 1 root root 9 Jul 2 2003 /usr/cate)
# chgrp cap /usr/cate
(-rwxr-sr-x 1 root cap 9 Jul 2 2003 /usr/cate)
```

3.2.3.3 Liên kết tập tin thư mục

Liên kết là hình thức dùng để tạo tên gọi khác (hay bí danh) cho các files hoặc thư mục trên hệ thống.

a. Các loại liên kết

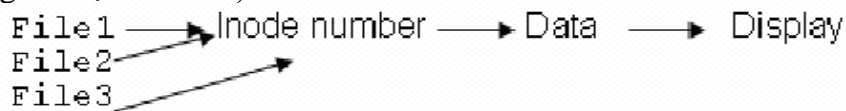
Trong Linux có 2 hình thức liên kết:

- *Liên kết cứng (hard link)*

Dùng để tạo ra một tên mới cho tập tin với đường dẫn tương ứng (các file phải nằm trên cùng hệ thống file).

Để xóa file phải xóa tất cả các liên kết cứng của nó.

Cấu trúc của liên kết cứng: Các files được liên kết cứng có cùng một số i-node (tham chiếu cùng dữ liệu trên đĩa).



- *Liên kết mềm (soft link)*

Là hình thức tạo shortcut như Window (chứa đường dẫn đến file gốc). Các file được liên kết mềm có thể qua các hệ thống file khác

Khi ta đọc/ghi liên kết mềm nghĩa là đọc/ghi trên tập tin; khi ta xóa liên kết mềm, ta chỉ xóa liên kết và tập tin vẫn được giữ nguyên.

Cấu trúc của liên kết mềm: Các files được liên kết mềm không chia sẻ cùng một inode



b. Tạo liên kết tập tin

Cú pháp: ln [-s] <source> <link_name>

Trong đó:

-s: dùng để tạo liên kết mềm.

+ Ví dụ tạo liên kết cứng:

```
$ ln /home/user2/dante essay
```

```
$ ls -i /home/user2/dante
89532 dante
$ ls -i essay
89532 essay
+ Ví dụ tạo liên kết mềm:
$ ln -s tutor.d symlink
$ ls -l symlink
lrwxrwxrwx 1 gonzo cap 8 May 9 symlink-->tutor.d
$ ls -F
tutor.d symlink@
```

3.3 Xử lý văn bản và các bộ lọc

3.3.1 Công cụ soạn thảo văn bản

Trong Linux hiện có rất nhiều các Text Editors khác nhau giúp cho việc soạn thảo text, như:

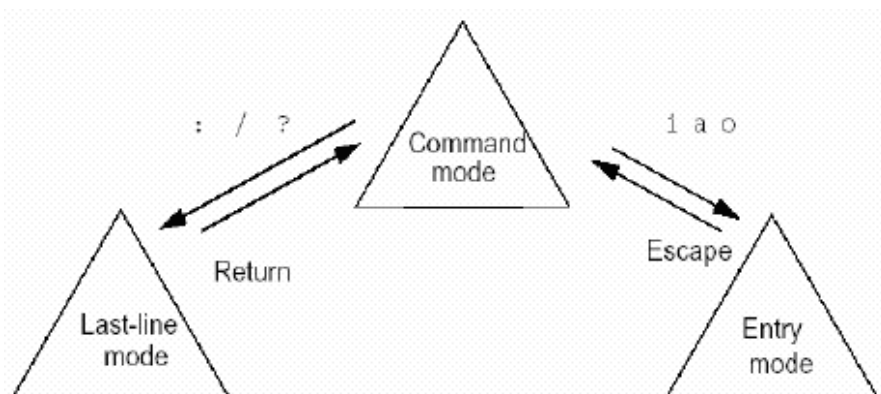
- vi (visual editor)
- emacs và xemacs
- jed
- joe

Trong đó bộ soạn thảo vi là bộ soạn thảo mặc định của các bản phân phối Linux và Unix.

Để tạo một file mới với vi dùng lệnh với cú pháp:

```
vi [options] [filename]
```

- Các chế độ làm việc của vi



+ Chế độ lệnh: Cho phép nhập lệnh xóa, thay đổi, sao chép và di chuyển text, vị trí con trỏ,...

+ Chế độ soạn thảo: Cho phép nhập và hiệu chỉnh văn bản vào file

Để chỉ dẫn vi vào chế độ soạn thảo, nhập một trong 3 lệnh sau: i (insert), o (open), a (append).

+ Chế độ dòng cuối: Cho phép lưu file, mở file, tìm kiếm, hoặc thoát khỏi vi,... Trong khi ở chế độ lệnh, gõ dấu hai chấm (:) sẽ vào chế độ dòng cuối.

- Một số lệnh dùng trong vi

dd: xóa một dòng

<n>dd: xóa n dòng

y: copy một dòng

<n>y: copy n dòng

p: Dán (paste) ra dòng hiện hành

dw: xóa một từ

:r <file>: mở một file mới

:r !<command>: lấy kết quả một lệnh shell đưa vào vi.

:w: lưu

:w <new file>: lưu với file mới

:wq!: lưu và thoát

:q!: Thoát mà không lưu

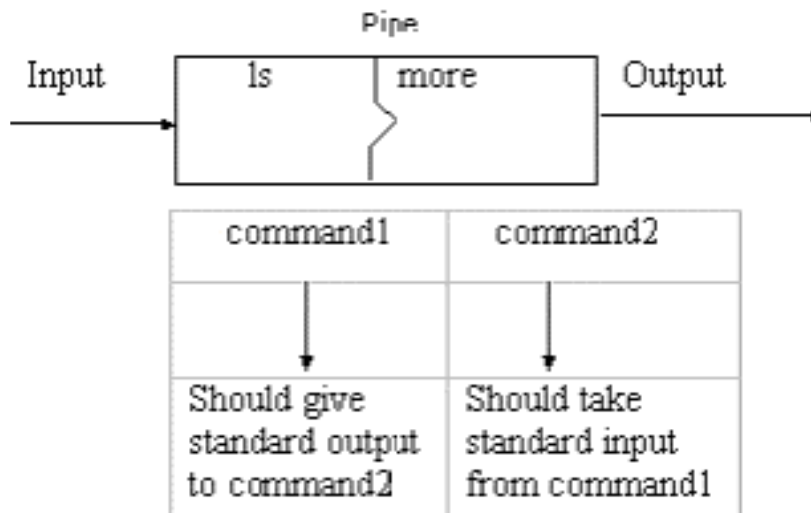
3.3.2 Pipe và các bộ lọc

3.3.2.1 Cơ chế Pipes

Pipes là cơ chế cho phép kết hợp nhiều lệnh và xử lý chúng như một lệnh.

Ví dụ: `$ ls -l /dev | more`

Pipes được biểu diễn bởi dấu gạch đứng (|), cái cho biết shell lấy kết quả của câu lệnh trước '|' và gửi chúng như dữ liệu vào cho câu lệnh sau '|'.



Ví dụ

- Xem tên đầy đủ của user steve cùng với đường dẫn, thư mục chủ và shell mặc định:

```
$ cat /etc/passwd | grep "^steve:" | cut -d ':' -f5,6,7
```

- Hiển thị ngày của tuần:

```
$ date | cut -d ' ' -f1
```

- Lệnh hiển thị tên các user và thời gian họ đăng nhập:

```
$ who | tr -s ' ' | cut -d ' ' -f1,4
```

3.3.2.2 Các lệnh lọc (filters)

Bộ lọc (*filter*) là một chương trình lấy dữ liệu vào từ thiết bị nhập, xử lý (hoặc lọc) nó và gửi kết quả đến thiết bị xuất.

Một số bộ lọc như: `grep`, `wc`, `tr`, `cut`,....

a. Bộ lọc `grep` (Stands for Global Regular Expression Print)

Tìm kiếm trong một file theo một mẫu các ký tự và hiển thị tất cả các dòng chứa mẫu đó. Mẫu để tìm kiếm được gọi là biểu thức chính quy (*regular expression*).

Cú pháp: `grep [options] pattern [filename]`

Trong đó:

`pattern`: là biểu thức chính quy có thể được sử dụng để xác định mẫu ký tự tìm kiếm, như: `[]`, `[^]` với `^`, `^` trong `[]`, `$`, `.` (dấu chấm), và `\`

Ví dụ:

`grep "New[abc]"`: Tìm dòng chứa `Newa`, `Newb` hoặc `Newc`

`grep "New[ac]"`: Tìm dòng chứa `Newa` hoặc `Newc`

`grep "^New[ab]"`: Tìm dòng chứa `Newa` hoặc `Newb` ở đầu dòng.

`grep "New[ab]$"`: Tìm dòng chứa `Newa` hoặc `Newb` ở cuối dòng.

`grep "New\[ab\]"`: Tìm dòng chứa `New[a]` hoặc `New[b]`

Ví dụ

Ta có file `test` với nội dung như sau:

```
James    25
Sandy    20
William  26
Alice    27
Steve    22
John     24
Jamie    45
```

Thực hiện các lệnh:

```
$ grep "J[oa]" test
```

```
$ grep "^Jo" test
```

```
$ grep "Steve$" test
```

b. Bộ lọc `wc`

Bộ lọc `wc` được sử dụng để đếm số dòng, số từ, và số ký tự trong một file hoặc từ thiết bị nhập.

Cú pháp: `wc [option] [filename]`

Một số tùy chọn của `wc`:

`-l`: hiển thị số dòng

- w: hiển thị số từ
- c: hiển thị số ký tự

c. Bộ lọc cut

Bộ lọc cut được sử dụng khi xác định các cột từ kết quả của một câu lệnh (như ls, who) hoặc một file trên đĩa cần được trích ra.

Cú pháp: cut [options] [filename]

Một số tùy chọn của cut:

- f<column(s)>: Hiển thị cột xác định
- c<character(s)>: Hiển thị ký tự xác định
- d<delimiter>: xác định dấu phân cách cột

Ví dụ:

```
$ cut -d ':' -f1 /etc/passwd
```

d. Bộ lọc tr

Bộ lọc tr có thể được sử dụng để chuyển đổi tập ký tự này thành tập ký tự khác

Nó cũng có thể được dùng để nén các ký tự lặp lại thành một ký tự với tùy chọn -s

Ví dụ:

```
$ who > dslogin
$ tr -s " " < dslogin
root tty1 Sep 28 17:02
steve pts/4 Sep 28 19:36 (172.17.55.167)
```

Các ví dụ sử dụng tr

+ Sử dụng phổ biến của tr cho phép chuyển đổi hoa-thường

```
$ tr "[a-z]" "[A-Z]"
Nguyen Van An
NGUYEN VAN AN
```

+ Ví dụ 2: đổi ký tự ":" thành khoảng trắng

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

e. Bộ lọc sort

Sắp xếp mỗi dòng từ file hoặc thiết bị nhập theo thứ tự tăng dần.

Các tùy chọn của bộ lọc sort:

- r: sắp xếp giảm dần
- n: Sắp xếp theo số
- t<delimiter> -k<column>: sắp xếp theo giá trị cột cụ thể.

❶ Câu hỏi bài tập

1. Cho biết nguyên tắc quản lý thiết bị trong Linux? Cách đặt tên đĩa cứng và đĩa mềm như thế nào?

2. Cấu trúc hệ thống file ext3 có những ưu nhược điểm gì? Cách mount hệ thống tập tin trong Linux?

3. Cho biết cơ chế phân quyền trong Linux và cách thiết lập quyền?

4. Nếu thiết lập giá trị umask là 233, cho biết các quyền mặc định thực sự của file và thư mục lúc mới được tạo (dạng ký hiệu và số bát phân)?

File permission:

Dir permission:

5. Nếu thiết lập giá trị umask là 251, cho biết các quyền mặc định thực sự của file và thư mục lúc mới được tạo?

File permission: (r-- -w- rw-) 426

Dir permission: 526

6. Cho biết một số trình soạn thảo trong Linux? Trình soạn thảo vi có mấy chế độ làm việc? cho biết một số lệnh sử dụng trong chế độ dòng cuối?

7. Cho file cơ sở dữ liệu sách, mỗi dòng (bản ghi) trong file này chứa:

Book code, Book name, Author Name, Publisher name, and Price.

Ví dụ: File chứa các dữ liệu sau:

b001:Programming in C++:Tom Wilkins:ABC Books:350

b003:Administering Oracle Databases:Corrine Wallace:New Tech Books:450

b002:Advanced Java: Chris Donaldson:New Tech Books:400

b005:Administering Linux:Nancy Jones:New Tech Books:350

b004:Shell Programming in Linux:Steve Irving:ABC Books:300

Yêu cầu:

Sử dụng các bộ lọc đã học thực hiện các yêu cầu sau:

a. Thêm các bản ghi này vào cơ sở dữ liệu.

b. Lọc và hiển thị những sách của Chris Donaldson.

c. Lọc và hiển thị sách có giá 400.

d. Lọc và hiển thị những sách lập trình và chỉ hiển thị tên sách và tác giả.

e. Sắp xếp file dựa trên trường Book code và hiển thị thông tin về sách.

f. Sắp xếp file dựa trên trường Price và hiển thị tên của sách và tác giả tương ứng.

g. Hiển thị bản ghi của các sách có giá từ 300 đến 499.

h. Thay đổi tên sách và tên tác giả thành chữ hoa và hiển thị số sách được viết bởi tác giả Steve Irving.

k. Cho biết hiện tại có bao nhiêu sách?

Chương 4. QUẢN TRỊ HỆ THỐNG VÀ NGƯỜI DÙNG

Trong chương này chúng ta sẽ tìm hiểu về cơ chế quản lý đa người dùng trong Linux, các lệnh để tạo nhóm, tạo người dùng mới. Sau đó, sẽ tìm hiểu về các hệ thống file ảo về của hoạt động nhân Linux và các lệnh quản lý tiến trình trong Linux.

4.1 Quản trị người dùng

4.1.1 Tài khoản người dùng (user)

Mọi truy nhập vào hệ thống đều thông qua một tài khoản của người sử dụng. Mỗi tài khoản được thiết lập bởi người quản trị hệ thống ngoại trừ tài khoản root (và một số tài khoản hệ thống). Mặc dù một số hệ Linux chỉ có một người dùng nhưng cũng không nên dùng tài khoản root cho các hoạt động thường ngày. Hầu hết các hệ thống cho phép nhiều người truy nhập vào. Vậy việc quản lý các tài khoản, các thư mục liên quan là một khía cạnh quan trọng trong việc quản trị hệ thống Linux

a. Tài khoản người dùng

Mỗi người sử dụng (user) trên hệ thống được mô tả qua các thông tin sau:

- + username: tên người sử dụng
- + password: mật khẩu
- + uid: số nhận dạng (user identify number)
- + gid: số của nhóm (group identify number)
- + comment: chú thích
- + Thư mục chủ của tài khoản (home directory)
- + Shell đăng nhập (chương trình chạy lúc bắt đầu phiên làm việc)

Các thông tin trên được chứa trong tập tin /etc/passwd

Ví dụ:

```
root:x:0:0:root:/root:/bin/bash
```

▪ Tài khoản root

Trong quá trình cài đặt Linux, trình cài đặt sẽ tạo ra một tài khoản đặc biệt với tên là root cho hệ thống.

Tài khoản root còn được gọi là tài khoản quản trị (admin) hay superuser có quyền không giới hạn.

Với user root sẽ có toàn quyền xử lý hệ thống như: tạo tài khoản mới, giới hạn hoặc cấp quyền cho từng tài khoản, shutdown hệ thống,...

Lời khuyên là không nên sử dụng tài khoản root để đăng nhập và làm việc với hệ thống và chỉ nên dùng trong những trường hợp thật cần thiết.

Tài khoản root được định nghĩa là tài khoản có UserID là 0, các userID được định nghĩa trong file /etc/passwd.

Chú ý:

- + Nếu đăng nhập với user thì dấu nhắc tại shell là \$
- + Nếu đăng nhập với root thì dấu nhắc tại shell là #

- Tập tin */etc/passwd*

Tập tin */etc/passwd* đóng một vai trò quan trọng với hệ thống Unix/Linux. Mọi người đều có thể đọc được tập tin này nhưng chỉ có root mới có quyền thay đổi nó.

Tập tin */etc/passwd* được lưu dưới dạng text như các tập tin cấu hình của Unix.

```
[srv@cap home]$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:
...
mars:x:500:500:Sao hoa:/home/mars:/bin/bash
```

Mỗi user được lưu trong một dòng gồm 7 cột.

Cột 1: tên người dùng

Cột 2: mã liên quan đến passwd cho Unix chuẩn và 'x' đối với Linux. Linux lưu mã này trong một tập tin khác */etc/shadow* mà chỉ có root mới có quyền đọc chỉ có root mới có quyền đọc.

Cột 3:4: user ID:group ID

Cột 5: Tên đầy đủ của người sử dụng. Một số phần mềm phá password sử dụng dữ liệu của cột này để thử đoán password.

Cột 6: thư mục cá nhân

Cột 7: chương trình sẽ chạy đầu tiên sau khi user login (thường là shell).

Tập tin mở đầu bởi superuser root. Chú ý là tất cả những user có user ID = 0 đều là root!!! Tiếp theo là các user hệ thống. Đây là các user không có thật và không thể login vào hệ thống. Cuối cùng là các user bình thường.

- Tập tin */etc/shadow*

Unix truyền thống lưu các thông tin liên quan tới mật khẩu để đăng nhập (login) ở trong */etc/passwd*. Tuy nhiên, do đây là tập tin phải đọc được bởi tất cả mọi người do một số yêu cầu cho hoạt động bình thường của hệ thống (như chuyển User ID thành tên khi hiển thị trong lệnh *ls* chẳng hạn) và nhìn chung các user đặt mật khẩu "yếu", do đó hầu hết các Unix phiên bản mới đều lưu mật khẩu trong một tập tin khác */etc/shadow* và chỉ có root được quyền đọc tập tin này.

Chú ý: Theo cách xây dựng mã hóa mật khẩu, chỉ có 2 cách phá mật khẩu là vét cạn (brute force) và đoán. Phương pháp vét cạn, theo tính toán chặt chẽ, là không thể thực hiện nổi vì đòi hỏi thời gian tính toán quá lớn, còn đoán thì chỉ tìm ra những mật khẩu ngắn, hoặc "yếu", ví dụ như những từ tìm thấy trong từ điển như god, darling...

b. Tạo một tài khoản mới

Để tạo một account, bạn có thể sử dụng lệnh *adduser* (hoặc *useradd* tùy vào phiên bản). Tất nhiên là bạn phải làm thao tác này dưới quyền root (dấu nhắc #)

Cú pháp: `useradd [-c comment] [-d home_dir] [-e expire_date] [-g initial_group] [-G group[,...]] [-p passwd] [-s shell] [-u uid] [-o]] [-n] [-r] login`

Trong đó các tham số:

- c comment: Lời chú thích , thường là tên đầy đủ của người dùng
- d home_dir: thư mục gốc của người dùng
- e expire_date: ngày hết hiệu lực của account
- g initial_group: nhóm khởi tạo
- G group: nhóm mà người dùng thuộc vào
- p passwd: password của người dùng, password này phải được mã hoá trước
- s shell: shell mặc định của user
- u uid: user identification

login: tên username.

Ví dụ:

```
[root@appserv oracle]# /usr/sbin/adduser foo
[root@appserv oracle]# passwd foo
Changing password for user foo
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully
[root@appserv oracle]#
```

Sau khi bạn tạo xong user bởi dòng đầu tiên của ví dụ trên, user foo vẫn chưa kết nối được vì thiếu password. Bạn phải khởi tạo password cho foo bởi lệnh `passwd foo` như thấy ở trên.

Vì vấn đề an ninh của máy Unix này và kéo theo sự an toàn của toàn hệ thống mạng của bạn, rất quan trọng chọn đúng password. Một password gọi là đúng nếu:

- Có độ dài tối thiểu 8 ký tự.
- Phối hợp giữa chữ thường, chữ hoa, số và các ký tự đặc biệt
- Không liên quan đến tên tuổi, ngày sinh ... của bạn và người thân
- Không có trong từ điển

Trong ví dụ trên, bạn khởi tạo người dùng và không quan tâm gì đến nhóm (group) của người dùng. Rất tiện lợi nếu bạn tập hợp nhiều người dùng vào chung một nhóm có cùng một chức năng và cùng chia sẻ nhau dữ liệu. Khi bạn tạo người sử dụng như trên, Linux sẽ tạo cho mỗi người một nhóm. Đọc tập tin `/etc/passwd` ta thấy

c. Thay đổi thuộc tính người dùng

Trong Linux có rất nhiều lệnh cho phép thay đổi một số thuộc tính cá nhân của tài khoản người dùng như:

- Lệnh `chfn`: thay đổi thông tin cá nhân người dùng.
- Lệnh `chsh`: thay đổi shell đăng nhập.

- Lệnh passwd: thay đổi mật khẩu.
- Lệnh usermod: có thể thực hiện mọi sự thay đổi trên tài khoản người dùng.

Cú pháp: usermod [tùy chọn] <username>

Các tùy chọn của lệnh:

- c, comment: các chú thích.
- d, home_dir: thư mục chủ người dùng.
- e, expire_date: ngày hết hạn (YYYY-MM-DD).
- f, inactive_days: số ngày được sử dụng.
- g, initial_group: nhóm người dùng, nhóm ngầm định là 1.
- G, group: thay đổi danh sách nhóm mà người dùng là thành viên. Mỗi nhóm cách nhau bởi dấu phẩy (,).
- l, login_name: Thay đổi tên đăng nhập.
- p, passwd: thay đổi mật khẩu đăng nhập.
- s, shell: thay đổi shell đăng nhập.
- u, uid: thay đổi số id người dùng.

Lệnh usermod không cho phép thay đổi tên người dùng khi người đó đang đăng nhập. Phải đảm bảo rằng người dùng đó không thực hiện bất kỳ quá trình nào khi thực hiện lệnh usermod để thay đổi thuộc tính của user đó.

Ví dụ để thay đổi tên người dùng jame thành john ta sử dụng lệnh sau:

```
# usermod -l jame john
```

d. Xóa một tài khoản

- Để xóa một user phải xóa mọi thứ liên quan đến user đó.

Cú pháp: userdel [-r] [username]

Lệnh sẽ thực hiện sửa đổi nội dung của các file tài khoản hệ thống bằng cách xóa các thông tin về người dùng trên dòng lệnh.

Trong đó, tùy chọn -r: các file tồn tại trong thư mục riêng của người dùng cũng như các file nằm trong thư mục khác có liên quan đến người dùng cũng bị xóa cùng lúc với thư mục người dùng.

Ví dụ: [root@srv foo]# userdel jerry

- Ta cũng có thể xóa một user bằng cách:
 - + Xóa các thông tin tương ứng của user trong /etc/passwd và trong /etc/group.
 - + Xóa các file mail và mail alias của người dùng
 - + Xóa mọi cron và at
 - + Xóa thư mục cá nhân của user đó.

4.1.2 Nhóm và các vấn đề liên quan

a. Nhóm người dùng (group)

Mọi người dùng trong các hệ unix hay Linux đều thuộc về một nhóm. Nhóm dùng để gom nhóm các users có chung một quyền hoặc chính sách riêng đối với hệ thống nhằm tạo thuận lợi trong việc quản trị hệ thống Linux. Ví dụ như trong một cơ quan, có nhiều phòng ban, mỗi phòng ban có các users và các users của các phòng ban khác nhau sẽ có các chính sách bảo mật khác nhau. Các users thường chỉ được sử dụng tài nguyên hệ thống một cách có hệ thống. Chẳng hạn các users của văn phòng và các phòng nghiên cứu được sử dụng các tài nguyên sau:

- + Truy cập Web
- + Sử dụng e-mail để trao đổi thông tin
- + Sử dụng các chương trình chat, icq để trao đổi tin tức
- + Truy cập đến các file server trong công ty
- + Không được login vào các máy chủ, không được chạy chương trình trên máy chủ

Tuy nhiên các users của phòng quản trị hệ thống có thể có các quyền ưu tiên hơn:

Bao gồm các quyền của người dùng bình thường trên

- + Có quyền thực thi một số lệnh đặc biệt dành cho quản trị hệ thống
- + Có thể login vào server.

Các nhóm được đặt quyền để các thành viên của nó có thể truy nhập đến các thiết bị, file, hệ thống file hoặc toàn bộ máy tính mà những người khác nhóm có thể bị hạn chế.

- Các thông tin về nhóm được lưu trong file `/etc/group`

Một nhóm người (group) sử dụng được mô tả bằng các thông tin sau:

- + groupname: tên của nhóm
- + gid: số của nhóm (gid: group identify number)
- + Danh sách các tài khoản thuộc nhóm (users)

- Tập tin `/etc/group`

```
[root@srv foo]# cat /etc/group
root: x: 0: root
bin: x: 1: root, bin, daemon
tty: x: 5
kmem: x: 9
wheel: x: 10: root
draft: x: 500: foo, tom, jerry
```

- Các nhóm mặc định của hệ thống

Mọi hệ Linux đều có một số các nhóm mặc định thuộc hệ điều hành. Các nhóm này thường là bin, mail, root, uucp, sys,...

Không nên cho một người sử dụng thuộc vào nhóm này vì chúng sẽ có quyền tương đương như root.

Chỉ có cách đăng nhập hệ thống với quyền root mới cho phép truy nhập đến các nhóm của hệ điều hành.

Các nhóm mặc định như: root, wheel, system: thường dùng để cho phép người dùng sử dụng lệnh su để chuyển lên quyền root.

daemon: dùng để chỉ những người làm chủ thư mục spool (mail, squid, lpd,...)

kmem: dùng cho các chương trình truy cập đến kernel, bộ nhớ trực tiếp (ps)

tty: làm chủ tất cả các file đặc biệt dùng làm việc với terminal.

b. Tạo & xoá nhóm

- Tạo nhóm mới, sử dụng lệnh:

```
groupadd [groupname]
```

Ví dụ: [root@srv foo]# groupadd star

- Sửa đổi thuộc tính của nhóm, sử dụng lệnh

```
groupmod [tùy chọn] <groupname>
```

Các tùy chọn:

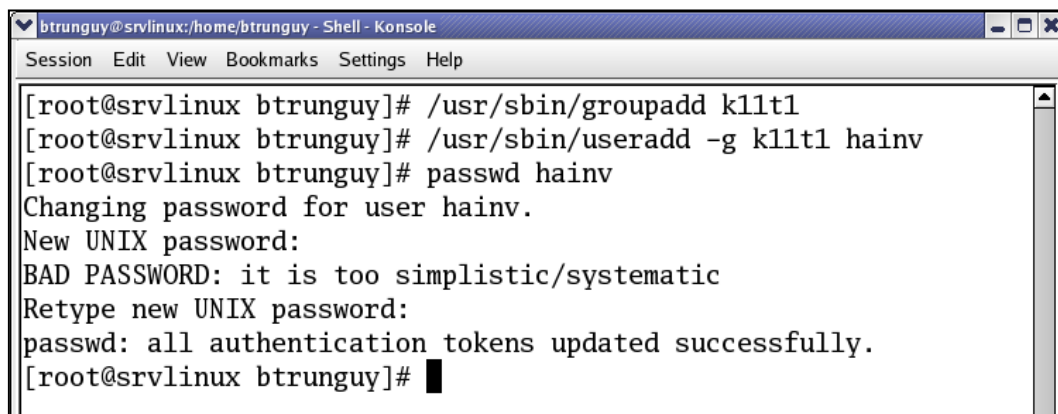
-g, gid: thay đổi số định danh của nhóm.

-n, group_name: thay đổi tên nhóm.

- Xóa nhóm đã tồn tại, sử dụng lệnh

```
groupdel [groupname]
```

Ví dụ: tạo nhóm và user trên terminal



```

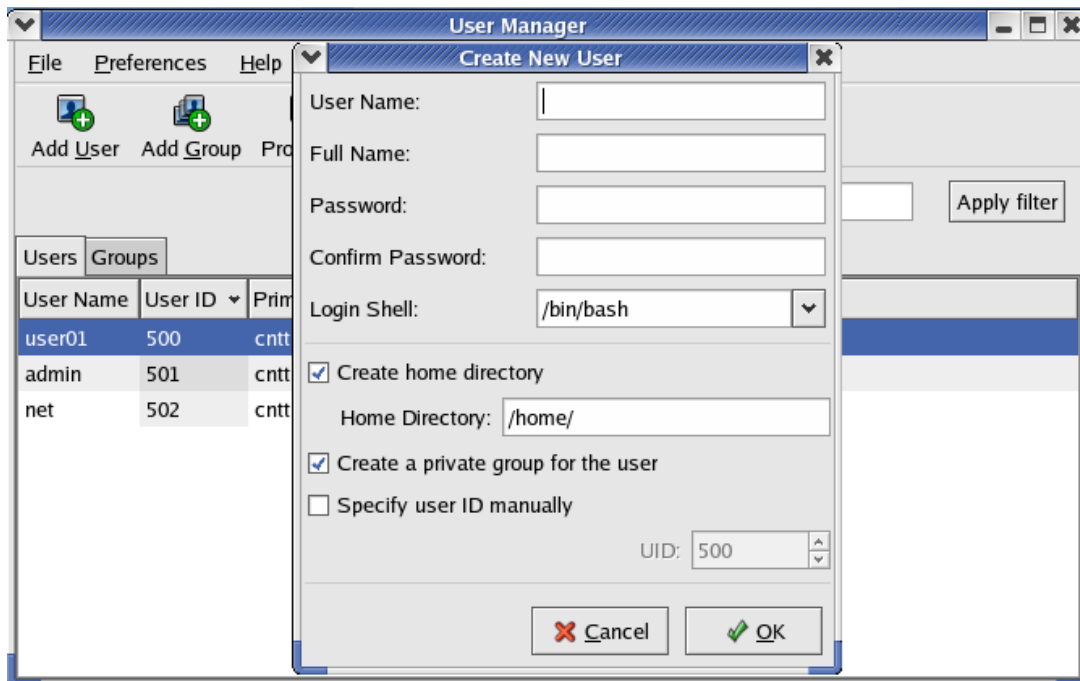
btrunguy@srvlinux: /home/btrunguy - Shell - Konsole
Session Edit View Bookmarks Settings Help

[root@srvlinux btrunguy]# /usr/sbin/groupadd k11t1
[root@srvlinux btrunguy]# /usr/sbin/useradd -g k11t1 hainv
[root@srvlinux btrunguy]# passwd hainv
Changing password for user hainv.
New UNIX password:
BAD PASSWORD: it is too simplistic/systematic
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
[root@srvlinux btrunguy]#
  
```

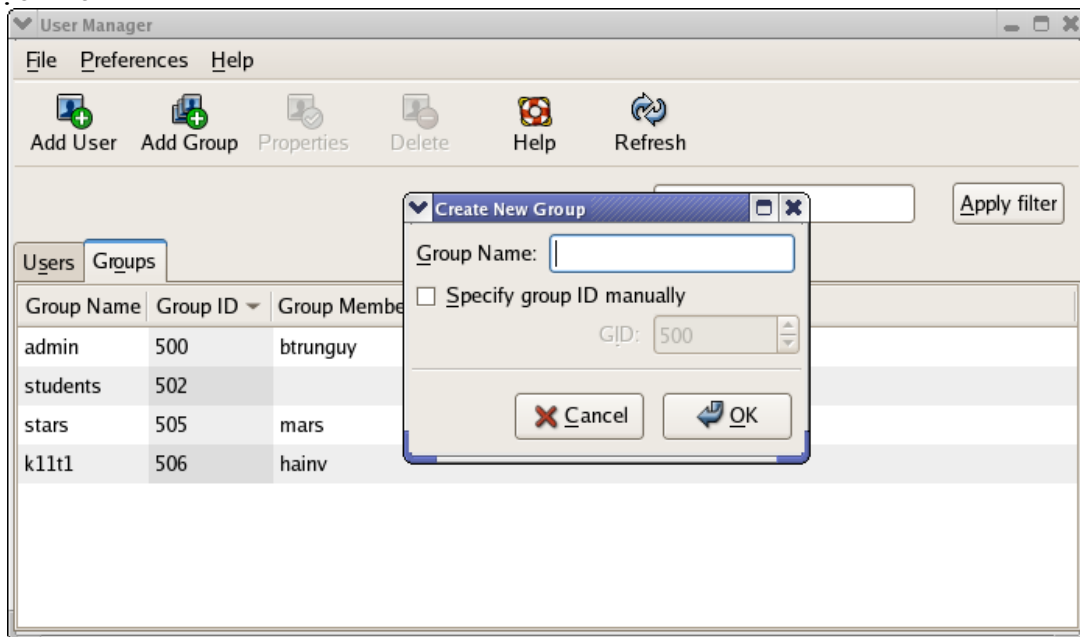
- Sử dụng công cụ đồ họa

Vào System menu -> Administration -> Users and Groups

- Tạo user mới



- Tạo nhóm mới



c. Một số lệnh liên quan đến người dùng

- Lệnh su: Đăng nhập với tư cách người dùng khác.

Cú pháp: su [username]

Lệnh su không có đối số sẽ đăng nhập với tư cách user root, đòi hỏi phải nhập mật khẩu.

- Lệnh who: xác định người dùng đang đăng nhập.

Cú pháp: who [-H] [-m] [-q]

Trong đó:

-H: hiển thị tiêu đề các cột kết quả lệnh

-m: hiển thị tên máy, người dùng và thiết bị vào chuẩn.

-q: hiển thị tên người dùng đăng nhập và số người dùng đăng nhập.

Ví dụ:

```
# who
root tty1 Nov 15 03:54
lan pts/0 Nov 15 06:07
```

- Lệnh users: Xác định những người dùng đăng nhập trên hệ thống
- Lệnh whoami hoặc who am i: User đang đăng nhập là ai.

Ví dụ:

```
# whoami
root
#
# who am i
localhost!root pts/0 Nov 15 06:07
```

- Lệnh id: Xác định thông tin người dùng.

Cú pháp: `id [-u] [-g]`

Trong đó:

-u: hiển thị số định danh người dùng

-g: hiển thị số định danh nhóm

Ví dụ:

```
# id
uid=506(tom) gid=503(tom) groups=503(tom)
#
# id -g
503
#
# id -u
506
#
# id root
uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),
3(sys),4(adm),6(disk),10(wheel)
#
```

-Lệnh w: liệt kê các user đã đăng nhập, thời điểm đăng nhập, độ rỗi, và lệnh mà họ đang chạy.

Cú pháp: `w [username]`

Ví dụ:

```
# w
root tty2 - 2:14pm 13:03 9.30s 9.10s /usr/bin/mc -P
lan pts/1 192.168.2.213 3:20pm 0.00s 0.69s 0.10s w
root pts/2:0 3:33pm 9:32 0.41s 0.29s /usr/bin/mc -P
```

4.2 Kiểm tra hoạt động hệ thống

4.2.1 Hệ thống file ảo /proc

Thư mục /proc là một thư mục rất quan trọng và đóng vai trò sống còn đối với hệ thống Linux. Thư mục này là một thư mục ảo. Nói cách khác là thông tin chứa trong thư mục này được tạo ra một cách động dựa trên các quá trình startup và shutdown của hệ thống. Hơn thế nữa, filesystem còn thay đổi theo thời gian thực. Thông tin của hệ thống, các tiến trình, các tham số của hệ thống đều thể hiện trong thư mục này.

Ta hãy làm một thử nghiệm sau:

Gọi vi chạy lên sau đó bấm Ctrl-Z để vi thành background job.

```
[root@starturn proc]# ps -ef | grep vi
root 11663 1359 0 22:35 pts/2 00:00:00 vim
```

Sau đó vào /proc ta sẽ thấy ngay có một thư mục là PID của vi: 11663

```
[root@starturn proc]# ls | grep 11663
11663
```

Sau đó ta cd vào 11663 sẽ thấy các trạng thái của vi đang chạy.

Và khi ta kết thúc vi, thư mục 11663 không còn tồn tại trong /proc nữa.

Các lệnh bảo trì khác

- Lệnh free: hiển thị tổng dung lượng bộ nhớ chính và swap đang được dùng và còn trống trong hệ thống cũng như shared memory và buffers được dùng bởi kernel.

```
[root@starturn 11663]# free
total used free shared buffers cached
Mem: 255452 247904 7548 0 8220 111312
-/+ buffers/cache: 128372 127080
Swap: 530136 0 530136
[root@starturn 11663]#
```

- Lệnh df: hiển thị dung lượng đĩa còn trống trên hệ thống file. Đơn vị hiển thị là 1K block, với 512 byte cho 1 block

```
[root@starturn root]# df -k
Filesystem 1K-blocks Used Available Use% Mounted on
/dev/hda1 10325748 2247448 7553780 23% /
none 127724 0 127724 0% /dev/shm
[root@starturn root]#
```

- Lệnh sudo: Super user do, cho phép quản trị hệ thống nâng cấp quyền truy xuất đến một tập lệnh quản trị hệ thống cho một vài users thường. Với sự trợ giúp của sudo, người dùng có thể làm một số thao tác cấu hình hệ thống mà không cần phải có quyền root.

4.2.2 Hệ thống log file

a. /var/log/message

Cho biết các sự kiện diễn ra trong hệ thống bao gồm các hành động start, stop các tiến trình, users login logout, các lỗi hệ thống,...

Ví dụ:

```
[root@starturn log]# more messages
Dec 8 17:46:41 localhost syslogd 1.4.1: restart.
Dec 8 17:46:41 localhost syslog: syslogd startup succeeded
Dec 8 17:46:41 localhost syslog: klogd startup succeeded
Dec 8 17:46:42 localhost kernel: klogd 1.4.1, log source =
/proc/kmsg started.
Dec 8 17:46:42 localhost kernel: Linux version 2.4.18-14
Dec 8 17:46:42 localhost kernel: 0MB HIGHMEM available.
Dec 8 17:46:42 localhost kernel: 256MB LOWMEM available.
Dec 8 17:46:42 localhost kernel: On node 0 totalpages: 65536
Dec 8 17:46:42 localhost kernel: zone(1): 61440 pages.
Dec 8 17:46:42 localhost kernel: zone(2): 0 pages.
Dec 8 17:46:42 localhost kernel: Kernel command line: ro
root=LABEL=/
Dec 8 17:46:42 localhost kernel: Console: colour VGA+ 80x25
Dec 8 17:46:42 localhost kernel: Calibrating delay loop... 896.66
BogoMIPS
Dec 8 17:46:42 localhost kernel: Memory: 253064k/262144k available
(1326k kernel code, 6648k reserved, 999k data, 212k init, 0k
highmem)
/var/log/secure
```

Lưu các thông tin thống kê login, logout và các ipaddress truy cập vào hệ thống

Ví dụ:

```
[root@starturn log]# more secure
Dec 8 17:46:54 localhost sshd[637]: Server listening on 0.0.0.0 port
22.
Dec 8 17:47:59 localhost xinetd[651]: START: sgi_fam pid=959
from=<no address>
Dec 8 22:07:09 localhost useradd[1432]: new user: name=tuanna,
uid=500, gid=500, home=/home/tuanna, shell=/bin/bash
Dec 8 22:08:18 localhost sshd[1492]: Could not reverse map address
192.168.10.100.
Dec 8 22:51:32 localhost sudo: tuanna: TTY=pts/2 ; PWD=/home/tuanna
; USER=root ; COMMAND=/etc/init.d/httpd start
Dec 8 22:51:54 localhost sudo: tuanna: TTY=pts/2 ; PWD=/home/tuanna
; USER=root ; COMMAND=/etc/init.d/httpd stop
[root@starturn log]#
```

b. /var/log/boot

Lưu các thông tin khi hệ thống mới khởi động

```
[root@starturn log]# more var/log/boot
Dec 8 22:35:24 localhost network: Shutting down interface eth0:
succeeded
Dec 8 22:35:24 localhost sysctl: net.ipv4.ip_forward = 1
Dec 8 22:35:24 localhost sysctl: net.ipv4.conf.default.rp_filter = 1
server's fully qualified domain name, using 127.0.0.1 for ServerName
Dec 8 22:51:37 localhost httpd: httpd startup succeeded
```

Dec 8 22:51:54 localhost httpd: httpd shutdown succeeded

4.3. Quản trị tiến trình

4.3.1 Tiến trình

Linux là một HDH đa người sử dụng, đa tiến trình. Linux thực hiện tất cả các công việc của người sử dụng cũng như của hệ thống bằng các tiến trình (process). Do đó, hiểu được cách điều khiển các tiến trình đang hoạt động trên HDH Linux rất quan trọng, nhiều khi có tính chất quyết định, cho việc quản trị hệ thống.

Định nghĩa: Tiến trình (process) là một chương trình đơn chạy trên không gian địa chỉ ảo của nó. Cần phân biệt tiến trình với lệnh vì một dòng lệnh trên shell có thể sinh ra nhiều tiến trình.

Dòng lệnh: `nroff -man ps.1 | grep kill | more`

sẽ sinh ra 3 tiến trình khác nhau.

Có 3 loại tiến trình chính trên Linux:

- Tiến trình với đối thoại (Interactive processes): là tiến trình khởi động và quản lý bởi shell, kể cả tiến trình foreground hoặc background.

- Tiến trình batch (Batch processes): Tiến trình không gắn liền đến bàn điều khiển (terminal) và được nằm trong hàng đợi để lần lượt thực hiện.

- Tiến trình ẩn trên bộ nhớ (Daemon processes): Là các tiến trình hậu cảnh (background). Các tiến trình này thường được khởi động từ đầu. Đa số các chương trình server cho các dịch vụ chạy theo phương thức này. Đây là các chương trình sau khi được gọi lên bộ nhớ, đợi thụ động các yêu cầu chương trình khách (client) để trả lời sau các công xác định (công là khái niệm gắn liền với giao thức TCP/IP BSD socket.). Hầu hết các dịch vụ trên Internet như mail, Web, Domain Name Service,... chạy theo nguyên tắc này. Các chương trình được gọi là các chương trình daemon và tên của nó thường kết thúc bằng ký tự “d” như named, inetd ... Ký tự “d” cuối được phát âm rời ra như “ê” trong tiếng việt. Ví dụ named được phát âm là “nê mê”.

Mỗi tiến trình có độ ưu tiên kết hợp với nó. Một user bình thường chỉ có thể giảm độ ưu tiên của một tiến trình sử dụng lệnh nice hoặc renice nhưng chỉ có root mới có quyền tăng độ ưu tiên của tiến trình

Để thực hiện một lệnh chạy background (tiến trình hậu cảnh) từ dòng lệnh thêm dấu & sau câu lệnh.

Ví dụ:

```
$ find / -name cool &
[1] 4463
```

Các chương trình chạy nền sau (background) vẫn hiển thị kết quả ở nền trước (foreground). Ta có thể chuyển hướng hiển thị vào một tập tin.

Ví dụ: `$ Find / -name cool > output &`

a. Thuộc tính chính của tiến trình

Mỗi tiến trình có một số định danh duy nhất trong hệ thống PID (Process ID)

Mỗi tiến trình có một chủ sở hữu UID và nhóm sở hữu GID xác định quyền của tiến trình trong hệ thống.

Ngoài ra, tiến trình có một số các thông tin khác như: ngữ cảnh tiến trình, đoạn dữ liệu, vùng nhớ stack, và không gian địa chỉ của tiến trình,...

Hiển thị tiến trình

Cú pháp: `ps [option]`

Trong đó: option có thể là:

-e: xem thông tin về mỗi tiến trình trên hệ thống.

-f: liệt kê dạng đầy đủ (liệt kê theo dạng các cột).

-l: liệt kê dạng thức dài.

Ví dụ:

```
$ ps -ef
```

| UID | PID | PPID | C | STIME | TTY | TIME | CMD |
|-------|------|------|----|----------|---------|------|-----------|
| root | 1 | 0 | 80 | 16:46:44 | ? | 0:40 | /etc/init |
| root | 2 | 0 | 27 | 16:46:44 | ? | 0:00 | pageout |
| aster | 1292 | 1 | 80 | 06:48:51 | console | 0:01 | -ksh |
| henry | 231 | 1 | 80 | 06:48:51 | pts/1 | 0:01 | bash |

Ý nghĩa của các cột:

UID: số UserID của chủ sở hữu tiến trình

PID: định danh của tiến trình

PPID: định danh của tiến trình cha

C: chỉ số sử dụng bộ xử lý (CPU utilization for scheduling)

STIME: thời điểm bắt đầu tiến trình

TTY: terminal điều khiển tiến trình

TIME: thời gian tích lũy thực hiện tiến trình

COMMAND: tên lệnh tạo ra tiến trình

b. Tìm kiếm tiến trình

Tìm các tiến trình theo mẫu dùng lệnh `pgrep`

Cú pháp: `pgrep [option] [pattern]`

Ví dụ: tìm các tiến trình với tên trong đó có “mysql”

```
$ pgrep -l -u mysql
```

```
$ ps -lu | grep mysql
```

Ví dụ: Tìm tiến trình trong đó có tên “lp”

```
$ ps -e | grep lp
```

```
225 ? 0:01 lpNet
```

```
217 ? 0:0 lp sched
```

```
260 ? 0:01 lpNet
```

```
$ pgrep -l lp
```

```
lpNet
lpsched
lpNet
```

c. Kiểm tra hoạt động tiến trình

Để kiểm tra hoạt động của các tiến trình, đặc biệt là các thông tin về tài nguyên hệ thống cũng như việc sử dụng tài nguyên của mỗi tiến trình dùng top.

Cú pháp: `top [-p pid]`

Trong đó:

-p pid: xem thông tin về tiến trình có định danh pid.

Ngoài ra, lệnh top còn cho phép theo dõi xem có tiến trình nào chiếm dụng quá nhiều thời gian CPU cũng như truy cập đĩa không.

```
23:10:38 up 1:19, 1 user, load average: 0.10, 0.03, 0.01
35 processes: 34 sleeping, 1 running, 0 zombie, 0 stopped
CPU states:  1.3% user  1.1% system  0.0% nice  0.0% iowait  97.4% idle
Mem:   93736k av,    34492k used,    59244k free,      0k shrd,    5116k buff
      28552k actv,    140k in_d,    220k in_c
Swap:  192772k av,      0k used,   192772k free                20188k cached
```

| PID | USER | PRI | NI | SIZE | RSS | SHARE | STAT | %CPU | %MEM | TIME | CPU | COMMAND |
|------|--------|-----|----|------|------|-------|------|------|------|------|-----|----------|
| 1830 | root | 16 | 0 | 1032 | 1032 | 856 | R | 2.1 | 1.1 | 0:03 | 0 | top |
| 1723 | root | 16 | 0 | 1516 | 1516 | 1092 | S | 0.0 | 1.6 | 0:02 | 0 | bash |
| 1722 | root | 22 | 0 | 392 | 392 | 340 | S | 0.0 | 0.4 | 0:00 | 0 | mingetty |
| 1721 | root | 22 | 0 | 392 | 392 | 340 | S | 0.0 | 0.4 | 0:00 | 0 | mingetty |
| 1720 | root | 22 | 0 | 388 | 388 | 340 | S | 0.0 | 0.4 | 0:00 | 0 | mingetty |
| 1719 | root | 22 | 0 | 392 | 392 | 340 | S | 0.0 | 0.4 | 0:00 | 0 | mingetty |
| 1718 | root | 22 | 0 | 388 | 388 | 340 | S | 0.0 | 0.4 | 0:00 | 0 | mingetty |
| 1717 | root | 15 | 0 | 960 | 960 | 764 | S | 0.0 | 1.0 | 0:00 | 0 | login |
| 1709 | daemon | 15 | 0 | 516 | 516 | 464 | S | 0.0 | 0.5 | 0:00 | 0 | atd |
| 1653 | root | 15 | 0 | 1876 | 1876 | 1400 | S | 0.0 | 2.0 | 0:00 | 0 | cupsd |
| 1642 | root | 15 | 0 | 560 | 560 | 496 | S | 0.0 | 0.5 | 0:00 | 0 | cron |
| 1633 | root | 15 | 0 | 436 | 436 | 388 | S | 0.0 | 0.4 | 0:00 | 0 | gpm |
| 1623 | smmsp | 15 | 0 | 2252 | 2244 | 1720 | S | 0.0 | 2.3 | 0:00 | 0 | sendmail |
| 1614 | root | 15 | 0 | 2492 | 2492 | 1840 | S | 0.0 | 2.6 | 0:00 | 0 | sendmail |
| 1595 | root | 24 | 0 | 844 | 844 | 724 | S | 0.0 | 0.9 | 0:00 | 0 | xinetd |
| 1581 | root | 25 | 0 | 1496 | 1496 | 1256 | S | 0.0 | 1.5 | 0:00 | 0 | sshd |
| 1543 | root | 24 | 0 | 476 | 476 | 428 | S | 0.0 | 0.5 | 0:00 | 0 | apmd |

4.3.2 Độ ưu tiên của tiến trình

Độ ưu tiên của tiến trình xác định quyền sử dụng CPU và ảnh hưởng đến quá trình điều phối tiến trình của nhân Linux.

Khi gán độ ưu tiên cho một tiến trình, Linux sử dụng một số xác định gọi là số nice (nice number). Các số nice có giá trị trong khoảng từ -20 (highest) đến +19 (lowest).

Các tiến trình bắt đầu bởi một người dùng thường có giá trị nice là 0.

Để thay đổi độ ưu tiên của tiến trình dùng lệnh nice hoặc renice.

▪ Lệnh nice

Cú pháp: `nice [options] [command [arg]...]`

Ý nghĩa: thực hiện lệnh “command” với độ ưu tiên được xác định bởi option.

Ví dụ:

```
$ nice -n 2 ls -l /root # Gán nice number là 2
```

```
$ nice --2 top # Gán nice number là -2
```

Các người dùng bình thường chỉ có thể điều chỉnh giá trị nice từ 0 đến 19, trong khi đó root có thể điều chỉnh giá trị nice từ -20 đến 19.

Nếu không có sự điều chỉnh giá trị, thì tiến trình được gán một giá trị mặc định là 0.

Lệnh nice không có đối số sẽ hiển thị các độ ưu tiên được điều phối hiện thời.

Ví dụ:

```
$ nice -2 ls -a # Gán nice number là 2
```

```
$ nice -n -2 ./baitap.sh # Gán nice number là -2
```

\$ nice

▪ Lệnh renice

Cú pháp: `renice priority PID [[-g] group] [[-u] user]`

Dùng để sửa đổi độ ưu tiên của các tiến trình đang hoạt động trong hệ thống.

Ví dụ:

+ Gán nice number là -2 cho tiến trình có định danh là 203

```
$ renice -2 203
```

+ Gán nice number là 5 cho tất cả tiến trình bắt đầu bởi user hainv

```
$ renice 5 -u hainv
```

4.3.3 Xử lý tín hiệu

Các lỗi chương trình liên quan đến phần cứng (như: chia cho 0), hoặc nhấn Ctrl+C, ... có thể gây ra các tín hiệu.

Các quản trị hệ thống thường sử dụng các tín hiệu để thao tác trên các tiến trình.

Có một số các tín hiệu chuẩn đã được định nghĩa trong Linux. Có thể sử dụng lệnh `kill -l` hoặc `man 7 signal` để liên kết tất cả các tín hiệu được hỗ trợ bởi hệ thống.

Có một số tín hiệu chuẩn trong Linux:

| Name | Value | Comment |
|---------|-------|---|
| SIGHUP | 1 | Hangup detected on controlling terminal or death of controlling process |
| SIGINT | 2 | Interrupt from keyboard (Ctrl+C pressed) |
| SIGKILL | 9 | Kill signal |
| SIGTERM | 15 | Terminate program |

Để gửi tín hiệu đến tiến trình dùng lệnh `kill` và lệnh `pkill`.

▪ Lệnh kill

Cú pháp: `kill [-signal] <PID(s)>`

Trong đó:

-signal: tín hiệu cần gửi đến tiến trình đang hoạt động

PID: định danh của tiến trình nhận tín hiệu.

Ví dụ:

```
$ kill -SIGKILL 3532
$ ps -u henry
PID          TTY          TIME      CMD
12892  console    0:01      ksh
12932  pts/0      0:01      find
12935  pts/1      0:00      ps
$ kill 12932
```

Hoặc:

```
$ kill -9 12932
```

▪ *Lệnh pkill*

Cú pháp: `pkill [-signal] [option] <pattern>`

Dùng để kết thúc một tiến trình với tên được so khớp theo mẫu.

Trong đó: Các giá trị của option có thể là:

-u username: các tiến trình thuộc username.

-x: so khớp chính xác với mẫu

Ví dụ:

```
$ pkill -9 -u tom mysql
```

4.3.4 Lập lịch tiến trình

Cho tiến trình nào chạy vào thời điểm nào?

Để lập lịch thực hiện các tiến trình Linux sử dụng chủ yếu hai lệnh sau:

▪ *Lệnh at*

Cho phép lập kế hoạch xử lý một hay nhiều câu lệnh vào một thời điểm cụ thể nào đó hoặc chạy lệnh trên một hàng đợi batch.

Cú pháp: `at [options] <time> <date>`

Lệnh at đọc các câu lệnh từ stdin hoặc tập tin (với tùy chọn là -f) và xử lý chúng sử dụng shell của user đó.

stdout và stderr của câu lệnh này được gửi mail đến user người chạy câu lệnh.

Ví dụ: Chạy chương trình myprogram một lần vào lúc 6:15 PM ngày mai:

```
$ at 6:15pm tomorrow
at> myprogram
at> Ctrl+D (nhấn Ctrl và phím D)
```

Ví dụ: Chạy các câu lệnh được liệt kê trong tập tin cmd.list vào lúc 9 p.m vào hai tới:

```
$ at -f cmd.list 9pm + 2 days
```

Hiển thị và hủy các at jobs

Liệt kê các công việc (jobs) đã lập với at:

```
$ at -l (là bí danh cho atq)
```

Hoặc:

```
$ atq
```

```
14 2003-10-31 12:00 a root
```

Hủy các công việc đã lập lịch:

```
$ at -d <JobID>
```

▪ *Lệnh crontab*

Lệnh at cho phép lập kế hoạch thực hiện tiến trình một lần. Linux còn cho phép lập kế hoạch có tính chất chu kỳ thông qua tiến trình crond và các tập tin crontab (hay còn gọi là bảng cron).

Tiến trình crond được bắt đầu lúc khởi động hệ thống, nó sẽ liên tục kiểm tra hàng đợi vào bởi lệnh at, và các tập tin crontab xem có tiến trình cần phải thực hiện hay không.

Tất cả các user đều có thể đặt các tiến trình sẽ được cho phép thực hiện bởi crond.

Cú pháp: `crontab [options] [-u user]`

Trong đó: option có thể là:

-l: liệt kê nội dung của bảng cron.

-e: sửa đổi bảng cron hiện hành.

-r: xóa bảng cron.

Ý nghĩa: Lệnh crontab cho phép cấu hình và thiết lập công việc trong bảng cron, nhằm thực hiện các công việc đó theo chu kỳ bởi tiến trình crond.

Mỗi user sẽ có một bảng cron trùng tên mình (username) nằm tại thư mục `/var/spool/cron` lưu tất cả các lệnh cần thực hiện theo chu kỳ.

▪ *Sử dụng lệnh crontab*

Liệt kê nội dung bảng cron dùng:

```
$ crontab [-u user] -l
```

Sửa đổi bảng cron hiện hành dùng:

```
$ crontab [-u user] -e
```

Xóa bảng cron dùng:

```
$ crontab [-u user] -r
```

▪ *Cấu trúc tập tin crontab*

Mỗi dòng trong tập tin crontab xác định một lệnh, và thông tin cần thiết khi nào thì lệnh được chạy, có dạng như sau:

```
<minute> <hour> <day> <month> <weekday> <command>
```

Trong đó: Giá trị có thể của các cột là

minute: phút (0-59)

hour: giờ (0-23)

day : ngày (1-31)

month: tháng (1-12)

weekday: ngày tuần (0-6, 0 là chủ nhật)

▪ Lưu ý:

+ Các cột cách nhau bởi ký tự Space hoặc tab.

+ Sử dụng dấu phẩy (,) để tách danh sách các giá trị trong một cột.

+ Cột với dấu sao "*" có nghĩa là có giá trị bất kỳ

+ Dòng bắt đầu với dấu "#" là dòng chú thích và được bỏ qua bởi tiến trình crond.

Ví dụ:

+ Để xử lý chương trình myprogram một lần trong một ngày vào lúc 6:15 a.m, sử dụng một dòng trong tập tin crontab như sau:

```
15 6 * * * myprogram
```

+ Để xử lý vào lúc 6:15 và 18:15 vào các ngày 1st và 15th hàng tháng, sử dụng:

```
15 6,18 1,15 * * myprogram
```

+ User có thể sử dụng crontab để hiển thị câu nhắc nhở thông thường hoặc backup dữ liệu:

```
0 8 29 2 * echo "Happy Birthday"
```

```
11 1 * * 0 /home/bin/full.backup
```

▪ *Quản trị at và crontab*

Kiểm soát và cấu hình thông qua các tập tin cấu hình được lưu giữ trong /etc

at.allow: Danh sách các users được phép sử dụng at

at.deny: Danh sách các users bị cấm sử dụng at (chỉ được sử dụng nếu không tồn tại at.allow)

cron.allow: Danh sách các users được phép sử dụng cron

cron.deny: Danh sách các users bị cấm sử dụng cron

crontab: bảng cron (cron table) hệ thống.

Các tập tin này chỉ đơn giản liệt kê các tài khoản người dùng (username).

Nếu tập tin allow tồn tại thì chỉ những users được liệt kê trong tập tin allow có thể sử dụng dịch vụ. Nếu tập tin allow không tồn tại nhưng tập tin deny có, thì chỉ những users không được liệt kê trong tập tin deny có thể sử dụng dịch vụ.

Đối với cron, nếu không có tập tin nào tồn tại thì tất cả các users đều có thể truy xuất cron.

Đối với at, nếu không có tập tin nào tồn tại thì, chỉ có root được truy xuất at.

Tất cả crontabs được lưu trong thư mục cron spool là /var/spool/cron

❶ Câu hỏi bài tập

1. Cho biết hệ thống đa người dùng là gì? Linux quản lý đa người dùng như thế nào? Cách tạo lập nhóm và người dùng trong Linux?
2. Cho biết cách xem thông tin hoạt động của nhân và thông tin những xâm nhập vào hệ thống?
3. Phân biệt tiến trình tiền cảnh và tiến trình hậu cảnh? Cho biết độ ưu tiên của tiến trình có gì đặc biệt? cách để thiết lập độ ưu tiên cho tiến trình?
4. Cho biết lập lịch tiến trình là gì? sử dụng at hoặc cron để lập lịch thực hiện file hello.sh vào lúc 5 giờ ngày mai.

Chương 5. LẬP TRÌNH SHELL SCRIPT

Trong phần này chúng ta sẽ tìm hiểu về cơ chế shell trong Linux, mục đích và môi trường làm việc của shell. Sau đó, chúng ta tìm hiểu về cấu trúc chương trình và cách thức lập trình shell trong Linux.

5.1 Sử dụng Shell

5.1.1 Giới thiệu

Shell (hệ vỏ) là một môi trường tương tác mà người dùng có thể giao tiếp với hệ điều hành.

Linux cung cấp một số shell như: C shell (csh), Korn shell (ksh), và Bourne shell (sh), BASH shell (bash) ,...

BASH (Bourn Again Shell) là shell mặc định trong Linux.

Shell có thể diễn dịch lệnh và cho phép người dùng lập trình script như một ngôn ngữ lập trình.

Linux cho phép chuyển đổi giữa nhiều shell hoặc là các console ảo (Virtual console) sử dụng lệnh chsh.

Cú pháp: `chsh [-s <shell>] [-l] [username]`

Trong đó:

-l: liệt kê các loại shell hiện có (thông tin các shell chứa trong tập tin /etc/shells).

Ví dụ: chuyển sang c shell

```
$ chsh csh ↵
```

5.1.2 Mục đích của shell

- Tương tác, diễn dịch lệnh: shell đợi người sử dụng gõ các lệnh tại dấu nhắc, sau đó gửi tới hệ thống yêu cầu từ lệnh nhận được.

- Đặt biến môi trường đối với mỗi người sử dụng

- Lập trình

Shell cung cấp tập hợp các lệnh từ đó có thể viết chương trình (được gọi là shell script).

Ngoài những lệnh đơn giản của hệ thống, shell còn được bổ sung thêm các cấu trúc phức tạp như điều khiển rẽ nhánh (if/case), vòng lặp (for/while),...

5.1.3 Điều khiển shell

Shell có thể được sử dụng như một ngôn ngữ lập trình gọi là ngôn ngữ shell. Các chương trình được viết và thông dịch bởi shell gọi là shell script.

Có hai cách để viết chương trình điều khiển shell:

- Điều khiển shell từ dòng lệnh

Ví dụ tìm và hiển thị tập tin mà nội dung có chứa chuỗi 'main()'. Thay vì dùng lệnh grep để tìm ra từng file sau đó sử dụng lệnh more để hiển thị. Ta có thể điều khiển shell tự động như sau:

```
$ for file in *
>do
if [ $(grep -l 'main()' $file) ]
then
more $file
fi
>done
```

- Điều khiển shell từ script

Nhược điểm của việc điều khiển shell từ dòng lệnh là khó lấy lại khối lệnh trước đó để sửa đổi và thực thi một lần nữa.

Để dễ bảo trì ta có thể đưa các lệnh vào một tập tin và yêu cầu shell đọc nội dung tập tin để thực thi.

Ví dụ: \$cat > docfile.sh

```
#!/bin/bash
for file in *
do
    if [ $(grep -l 'main()' $file) ];
    then
        more $file
    fi
done
```

5.1.4 Môi trường Shell

Môi trường shell là tất cả các thiết lập giúp cho việc chạy chương trình được chính xác. Bao gồm:

Khi shell được khởi động, nó đọc các files cấu hình của sau:

/etc/profile: các thiết lập hệ thống chung

~/.bash_profile: các thiết lập của người dùng

~/.bashrc: tập tin khởi động trong một phiên làm việc cụ thể.

~/.bash_login: các thiết lập đăng nhập

~/.bash_history: lịch sử các câu lệnh đã thực hiện

- Tập tin /etc/profile

Chứa tất cả các thiết lập áp dụng đến tất cả môi trường người dùng.

Khi được triệu gọi tương tác, shell đọc các chỉ thị (câu lệnh) trong /etc/profile. Đây thường là các thiết lập các biến shell như PATH, USER, MAIL, HOSTNAME và HISTSIZE.

Trên một số hệ thống, giá trị umask cũng được cấu hình trong /etc/profile.

Ví dụ /etc/profile

```
# System wide environment and startup
# programs, for login setup
PATH=$PATH:/usr/X11R6/bin
USER="`id -un`"
LOGNAME=$USER
HOSTNAME=`/bin/hostname`
export PATH USER LOGNAME HOSTNAME PS1
# Source initialization files for specific
# programs (ls, vim, less,...)
for i in /etc/profile.d/*.sh ; do
if [ -r "$i" ]; then
. $i
fi
done
```

▪ *Tập tin /etc/bashrc*

Trên hệ thống /etc/profile chỉ lưu giữ môi trường shell và các thiết lập khởi động chương trình, trong khi đó /etc/bashrc chứa các định nghĩa hệ thống chung cho các hàm và bí danh shell.

Tập tin /etc/bashrc cũng có thể đề cập đến trong /etc/profile hoặc trong các tập tin khởi tạo shell riêng biệt của người dùng.

Ví dụ:

```
alias ll='ls -l'
alias dir='ls -ba'
alias c='clear'
alias mroe='more'
alias ls='ls --color'
pskill(){
pid=$(ps -ax | grep $1)
echo -n "killing $1 (process $pid)..."
kill -9 $pid
echo "slaughtered."
}
```

▪ *Tập tin ~/.bash_login*

Tập tin này chứa các thiết lập khi người dùng login vào hệ thống.

Ví dụ:

```
# file protection
# all to me, read to group and others
umask 002

# miscellaneous

w

cal `date +"%m"` `date +"%Y"`
```

5.2 Lập trình shell

5.2.1 Cấu trúc chương trình shell

Để soạn thảo một chương trình shell có thể sử dụng bất kỳ trình soạn thảo văn bản nào (chẳng hạn: vi, emacs,...)

- Một shell script cơ bản gồm:

- + Dòng đầu tiên thường cho biết shell được sử dụng và gọi trình thông dịch shell script tương ứng (ví dụ: `#!/bin/bash`).

- + Các chú thích bắt đầu với dấu thăng (#).

- + Các dòng còn lại là các lệnh của Linux hoặc các cấu trúc điều khiển,...

- Thực thi shell script:

C1:

- Đặt quyền execute: `$ chmod a-x <shellscript>`

- Chạy shell script: `$./<shellscript>`

C2: `$. <shellscript>`

5.2.2 Các thành phần cơ bản

a. Các lệnh vào/ra

- *Lệnh ghi ra màn hình*

Cú pháp: `echo [-n] [biểu thức]`

Dùng để hiển thị giá trị các biến, biểu thức hoặc chuỗi ra màn hình.

Trong đó:

–n: không xuống dòng sau khi in ra.

Nếu biểu thức là chuỗi đặt trong cặp nháy kép có thể sử dụng các ký tự đặc biệt như `\n`, `\b`, `\t`, `\0n` (ký tự có mã ASCII là n),...

Ví dụ:

```
echo "How are you?"
```

```
echo "Current user: $USER"
```

- *Lệnh nhập dữ liệu từ bàn phím*

Cú pháp: `read <biến 1> [biến 2] [biến 3]...`

Dùng để đọc dữ liệu từ bàn phím.

Ví dụ shell script đọc và in dữ liệu.

```
$ vi vidu1.sh
```

```
#!/bin/bash
```

```
#Vi dụ sử dụng lệnh read và echo
```

```
echo "Nhập một số: "
```

```
read num
```

```
echo "Số vừa nhập là: $num"
```

```
return
```

Ví dụ Shell Script hello

```
$ vi hello.sh
#!/bin/bash
#Day la chu thich: chuong trinh hello.sh
echo "Nhap vao ten ban:"
read hoten
echo "Xin chao $hoten, chuc mot ngay vui ve!"
return
```

Cách gọi thực hiện hello.sh:

```
$chmod a+x hello.sh
$./hello.sh
```

Ví dụ: showsys.sh

```
#!/bin/bash
clear
echo "Day la thong tin ve he thong:"
echo "Xin chao, $USER"
echo "Hom nay la ngay `date`, tuan `date +%V`."
echo "Cac user da login:"
who
echo "Day la he thong `uname -s` chay tren procesor `uname -m`"
echo "Thoi gian da chay: `uptime`"
```

Ví dụ: taotm.sh

```
#!/bin/bash
if [ $# -lt 1 ]; then
    echo Usage: $0 [thu muc]
    exit
fi
if [ -d "$1" ]; then
    echo "Thu muc $1 da ton tai."
else
    mkdir $1
fi
```

b. Biến shell

Shell cho phép sử dụng biến nhưng không cần khai báo và định nghĩa kiểu.

Mặc định, tất cả các biến đều được khởi tạo và chứa trị kiểu chuỗi (ngay cả khi gán giá trị là một số thì shell cũng xem là chuỗi).

Shell và một vài lệnh tiện ích sẽ tự động chuyển biến chuỗi thành số để thực hiện phép tính khi có yêu cầu.

Biến shell có thể dùng để lập trình hoặc để điều khiển môi trường. Biến phân biệt chữ hoa thường.

▪ *Sử dụng biến*

Cũng tương tự như ngôn ngữ lập trình, Shell cung cấp phép gán và lấy giá trị của biến như sau:

<tên biến> = <giá trị>: ở đây giá trị có thể là một số, chuỗi hay từ một biến khác.

<tên biến> = `command`: Gán giá trị cho biến là kết quả thực hiện của một lệnh

\$<tên biến>: dùng để lấy giá trị của biến.

Ví dụ:

```
$x = 38
```

```
$echo $x
```

```
38
```

▪ *Biến môi trường shell*

Khi trình shell khởi động nó cung cấp sẵn một số biến được khai báo và gán giá trị mặc định, chúng được gọi là biến môi trường.

Các biến môi trường thường được viết hoa để phân biệt với các biến do người dùng định nghĩa.

Mỗi môi trường đăng nhập chứa các biến môi trường riêng biệt dùng cho mục đích riêng.

Để hiển thị các biến môi trường dùng lệnh env. Để tạo một biến môi trường mới dùng lệnh export (hoặc setenv)

- Một số biến môi trường

HOSTNAME: Tên máy Linux

HOME: Thư mục chủ của người dùng

SHELL: Shell hiện hành

PWD: Thư mục hiện hành

PATH: Danh sách các thư mục để tìm kiếm các lệnh, phân cách bởi dấu hai chấm (:)

USER: Tên người dùng hiện hành

TERM: Kiểu của terminal hiện hành

PS1: Dấu nhắc shell

▪ *Biến mảng*

Mảng là biến dùng lưu nhiều giá trị. Trong shell, một biến bất kỳ có thể được sử dụng như một mảng.

Không có giới hạn kích thước của mảng, và cũng không có đòi hỏi về các thành phần của mảng được đánh chỉ số hoặc gán giá trị một cách liên tục.

Phần tử đầu tiên của mảng có chỉ số là 0.

- Khai báo biến mảng

+ Khai báo gián tiếp sử dụng cú pháp sau:

```
Tenmang[index]=value
```

+ Khai báo tường minh của một mảng sử dụng:

```
declare -a Tenmang
```

Các biến mảng cũng có thể được tạo sử dụng sự chỉ định kép theo định dạng:

```
Tenmang = (value1 value2... valueN)
```

Việc bổ sung các phần tử thiếu hoặc thêm phần tử vào mảng sử dụng cú pháp:

```
Tenmang[index]=value
```

- Ví dụ sử dụng mảng

```
$Arr=(one two three)
```

```
$echo ${Arr[*]}
```

```
one two three
```

```
$echo $ Arr[*]
```

```
one[*]
```

```
$echo ${Arr}]}
```

```
three
```

```
$ Arr[3]=four
```

```
$echo ${Arr[*]}
```

```
one two three four
```

- Lấy độ dài của biến mảng

```
$echo ${#Arr[@]}
```

▪ *Xóa biến mảng*

Lệnh unset được sử dụng để hủy các mảng hoặc các phần tử của mảng:

Ví dụ:

```
$unset Arr[1]
```

```
$echo ${Arr[*]}
```

```
one three four
```

```
$unset Arr
```

```
$echo ${Arr[*]}
```

c. Cách dùng các dấu bao

Trong shell script có ba loại dấu đặc biệt dùng trong các lệnh in ra màn hình hay lệnh gán, nhưng ý nghĩa và cách thực hiện có khác biệt.

▪ *Dấu huyền (`)*

Dùng để gọi thực hiện một lệnh hệ thống trong shell script.

Ví dụ, trong shell script có các dòng:

```
currentdir=`pwd`
```

```
linecount=`wc -l $filename`
```

Trong đó:

Dòng 1 sẽ thực hiện lệnh pwd và gán đường dẫn hiện hành vào biến currentdir.

Dòng 2 thực hiện 1 lệnh wc đếm số dòng trong tập tin \$filename và gán cho biến linecount.

- Dấu nháy đơn (') và nháy kép (")

Dùng để hiển thị ra màn hình bởi lệnh echo hoặc gán giá trị cho biến dạng chuỗi.

Dấu nháy kép (") khi in ra sẽ thực hiện với giá trị của biến sau dấu \$.

Dấu nháy đơn (') khi in ra sẽ in nguyên dòng văn bản trong câu lệnh.

Ví dụ:

```
myname="John Terry"
echo "$myname"           # In ra: John Terry
echo '$myname'          # In ra: $myname
```

d. Biểu thức toán học

Các toán tử: +, -, *, /, %

Sử dụng let, \$(), expr hoặc \$[] để thực hiện tính giá trị biểu thức toán học

Ví dụ:

```
let "sum = 4 + 3"
count=`expr $count + 1`
area=$(( $length * $width ))
percent=$(( $num / 100 ))
remain=$(( $n % $d ))
x=$(echo "sqrt(8)" | bc -l)
y=$(echo "scale=2; $x/3" | bc -l)
```

e. Biểu thức điều kiện

Là biểu thức kiểm tra logic, trả về kết quả là đúng (1) hay sai (0). Nó thường được dùng trong các cấu trúc điều khiển chương trình (như if, while,...)

Trong shell script sử dụng lệnh [] hoặc test để kiểm tra biểu thức logic.

Cú pháp:

```
test <biểu thức>
[ <biểu thức> ]
```

Ví dụ biểu thức điều kiện kiểm tra a lớn hơn b

```
if test $a -gt $b
then
    echo "a lon hon b"
fi
```

Sử dụng [] thay cho test:

```
if [ $a -gt $b ]
```

```
then
    echo "a lon hon b"
fi
```

▪ *Biểu so sánh kiểu chuỗi*

S1 = S2: Chuỗi S1 bằng chuỗi S2 không?

S1 != S2: Chuỗi S1 khác chuỗi S2 không?

S1 > S2: S1 đứng trước S2 theo thứ tự ab không?

S1 < S2: S1 đứng sau S2 theo thứ tự ab không?

-n S1: Chuỗi S1 có độ dài lớn hơn 0 không?

-z S1: Chuỗi S1 có độ dài bằng 0 không?

▪ *Biểu thức so sánh số học*

n1 -eq n2: n1 bằng n2?

n1 -ge n2: n1 lớn hơn hoặc bằng n2?

n1 -gt n2: n1 lớn hơn n2?

n1 -le n2: n1 bé hơn hoặc bằng n2?

n1 -lt n2 : n1 bé hơn n2?

n1 -ne n2: n1 khác n2?

! n: phủ định của biểu thức n (phép not)

▪ *Kiểm tra tập tin, thư mục*

-f file: file có phải là tập tin không.

-d dir: dir có phải là thư mục không.

-c name: name có phải là tập tin ký tự không.

-s name: name có kích thước lớn hơn 0 không

▪ *Toán tử điều kiện AND (-a) và OR (-o).*

Đối với các biểu thức có nhiều điều kiện kết hợp ta có thể sử dụng các toán tử logic AND và OR như sau:

Ví dụ:

-Toán tử -a:

```
if [ $a -lt $b -a $a -lt $c ]; then
    echo "a la so lon nhat."
endif
```

-Toán tử -o:

```
if [ $a -gt 12 -o $a -lt 5 ]; then
    echo "a nam ngoai khoang [5, 12]"
endif
```

5.2.3 Các cấu trúc điều khiển

a. Cấu trúc rẽ nhánh if-else

Cú pháp:

```
if [điều kiện]
then
    <nhóm l ệnh>
[elif <điều kiện>; then
    <nhóm lệnh>]
...
[else
    <nhóm lệnh>]
fi
```

Ví dụ cấu trúc if so sánh hai số

```
#!/bin/sh
echo -n "Nhập số a: "
read a
echo -n "Nhập số b: "
read b
if [ $a -lt $b ]; then
    echo "Số $a nhỏ hơn số $b."
elif [ $a -eq $b ]; then
    echo "Số $a bằng số $b."
else
    echo "Số $a lớn hơn số $b."
fi
```

b. Cấu trúc lựa chọn case

Cú pháp:

```
case $<biểu thức> in
    <mẫu1>)
        <nhóm lệnh1>;;
    <mẫu2>)
        <nhóm lệnh2>;;
    *)
        <nhóm lệnh#>;;
esac
```

Ví dụ:

```
#!/bin/bash
echo "Is it morning? Please answer yes or no:"
read answer
case "$answer" in
    "yes")
        echo "Good morning";;
    "no")
        echo "Good afternoon";;
    *)
        echo "Sorry, answer not recognized.";
esac
```

c. Cấu trúc lặp for**Cú pháp:**

```
for <biến> in <values>
do
    <nhóm lệnh>
done
```

d. Cấu trúc while**Cú pháp:**

```
while <điều kiện>
do
    <nhóm lệnh>
done
```

Ví dụ nhập mật khẩu

```
#!/bin/bash
echo -n "Enter password:"
read password
while [ "$password" != "secret" ]
do
    echo "Sorry, try again !"
    read password
done
echo "Password is accepted !"
exit 0
```

5.2.4 Một số tiện ích**a. Phát sinh số ngẫu nhiên:**

\$RANDOM là một biến hàm của shell cho phép phát sinh ngẫu nhiên một số trong phạm vi từ 0 đến 32767.

Ví dụ: Phát sinh ngẫu nhiên 10 số nguyên và hiển thị tổng của các số đó ra màn hình.

```
#!/bin/bash
N=10
Count=0
echo "Chương trình phát sinh 10 số nguyên ngẫu nhiên"
while [ $Count -lt $N ]
do
    Num=$RANDOM
    let "Count += 1"
    echo "Số thứ $Count là $Num"
    let "Sum += $Num"
done
echo "Tổng 10 số nguyên do là: $Sum"
```

b. Các lệnh nội tại shell

Ngoài các cấu trúc trên, shell còn cho phép sử dụng các lệnh nhảy và thoát sau:

- Lệnh BREAK – thoát khỏi vòng lặp
- Lệnh CONTINUE – bỏ qua các câu lệnh còn lại của vòng lặp và trở lại đầu vòng lặp.
- Lệnh EXIT – kết thúc shell script
- Lệnh RETURN – trả về từ hàm hoặc shell script

Chú ý: Trong phần điều kiện của các cấu trúc, sau dấu “[” và trước dấu “]” phải có một ký tự trắng.

5.2.5 Một số chương trình tham khảo

Bài 1. Chương trình tạo menu tương tác với người dùng:

```
#!/bin/bash

while :
do
    clear
    echo "-----"
    echo "  Main Menu"
    echo "-----"
    echo "[1] Show today date/time"
    echo "[2] Show all files in current directory"
    echo "[3] Show calendar"
    echo "[4] Exit/Stop"
    echo "===== "
    echo -n "Enter your choice [1-4]: "
    read choice

    case $choice in
        1) echo "Today is `date` "
            echo "Press Enter key to continue ..."; read;;
        2) echo "Files in $PWD"; ls -l
            echo "Press Enter key to continue..."; read;;
        3) cal ; echo "Press Enter key to continue..."; read;;
        4) exit 0;;
        *) echo "Please choice 1,2,3,4. Press Enter key to continue..."; read;;
    esac
done
```

Bài 2. Tính tổng các số từ 1 đến n (n được nhập từ bàn phím hoặc lấy từ đối số dòng lệnh).

```
#!/bin/sh
echo "Chương trình tính tổng 1->$1"
i=0
tong=0
while [ $i -lt $1 ]; do
    i=$((i + 1))
    tong=$((tong+i))
done
echo "Tổng 1->$1= $tong"
return
```

Bài 3. Tính giai thừa của n (n!)

```
#!/bin/sh
echo "Chương trình tính $1!"
i=0
gt=1
while [ $i -lt $1 ]; do
    i=$((i + 1))
    gt=$((gt * i))
done
echo "$1!= $gt"
return
```

① Câu hỏi bài tập

- Viết shell script tìm giá trị lớn nhất trong ba số a, b, c.
- Viết shell script giải phương trình bậc nhất: $ax + b = 0$.
- Viết shell script thực nhập vào hai số và một toán tử sau đó hiển thị kết quả.
- Viết shell script tính điểm trung bình và xếp loại của một thí sinh thi tuyển sinh (có ba cột điểm)
- Viết shell script thực hiện nhập vào một giá trị là thứ trong tuần. Hiển thị chữ tiếng anh của thứ đó.
- Tính tổng $s = 1 + 3 + 5 + \dots + n$
- Tính tổng dãy: Hãy viết câu lệnh cho biết hiện tại có bao nhiêu User đang đăng nhập vào hệ thống?
- Hãy viết chương trình Shell Script tính tổng dãy:

$$S = 1 + \frac{1}{3^2} + \frac{1}{5^2} + \dots + \frac{1}{(2n+1)^2}$$

- Viết shell script kiểm tra một số có phải là số nguyên tố không?
- Viết một shell script thực hiện nhập vào 3 số nguyên a, b, c. Kiểm tra xem có phải là 3 cạnh của tam giác hay không? Nếu phải thì cho biết đó là tam giác vuông, cân hay đều? Tính diện tích tam giác?

Họ & tên sinh viên:

Mã số sinh viên:

Lớp:

BÀI THỰC HÀNH SỐ 1

✚ Mục đích: Sử dụng một số lệnh cơ bản trong Linux

✚ Yêu cầu: Đọc kỹ tài liệu trước khi thực hành

✚ Mục tiêu

- Làm quen với hệ điều hành Linux
- Làm quen với các tập lệnh trên Linux
- Giúp sinh viên nắm rõ về yêu cầu về hệ thống và các cách cài đặt và cấu hình Linux.
- Thay đổi mật khẩu, thao tác trên màn hình.
- Xem ngày giờ hệ thống, gửi và nhận thông điệp, ...

✚ Nội dung

- Sinh viên login với user đã đăng ký
- Lưu ý: Không login với user root để thực hiện.

Câu 1. Thay đổi mật khẩu sử dụng lệnh `passwd`

`[sinhvien@LinuxServer ~]$passwd`

Ví dụ 1: Người dùng **root** thay đổi password của người dùng **sinhvien**

Lưu ý: Password sẽ không hiển thị khi chúng ta nhập

```
[root@LinuxServer ~]# passwd sinhvien ↵
Changing password for user sinhvien.
New UNIX password:                ← Password mới
BAD PASSWORD: it is based on a dictionary word
Retype new UNIX password:         ← Nhập lại Password mới
passwd: all authentication tokens updated successfully.
[root@LinuxServer ~]#
```

Ví dụ 2: Người dùng **sinhvien** thay đổi password của chính mình

Lưu ý: Password phải có độ dài tối thiểu 8 ký tự & không quá đơn giản như: 12345678, hoặc abcdefgh, ...

```
[sinhvien@LinuxServer ~]$ passwd ↵
Changing password for user sinhvien.
Changing password for sinhvien
(current) UNIX password:          ← Password hiện tại
New UNIX password:                ← Password mới
Retype new UNIX password:         ← Nhập lại Password mới
passwd: all authentication tokens updated successfully.
[sinhvien@LinuxServer ~]$
```

Câu 2. Xem ngày giờ hệ thống

```
[root@LinuxServer ~]# date ↵
Fri Jun 15 19:18:29 ICT 2012
[root@LinuxServer ~]#
```

Các tùy chọn của lệnh date:

| Option | Description |
|---------------|---|
| %m | Displays month of the year (in digits) |
| %d | Displays day of the month (in digits) |
| %y | Displays year (last two digits) |
| %D | Displays date as mm/dd/yy |
| %H | Displays hour (00 to 23) |
| %M | Displays minutes (00 to 59) |
| %S | Displays seconds (00 to 59) |
| %T | Displays time as HH:MM:SS |
| %a | Displays abbreviated weekday (Sun to Sat) |
| %h | Displays abbreviated month (Jan to Dec) |

Ví dụ 1:

```
[sinhvien@LinuxServer ~]$ date "+Hom nay la ngay: %D" ↵
Hom nay la ngay: 06/15/12
[sinhvien@LinuxServer ~]$
```

Ví dụ 2:

```
[sinhvien@LinuxServer ~]$ date "+Hom nay la ngay: %d/%m/%Y" ↵
Hom nay la ngay: 15/06/2011
[sinhvien@LinuxServer ~]$
```

Câu 3. Lệnh thao tác trên màn hình

- ❖ Linux cho phép chúng ta thao tác trên màn hình với lệnh clear
- ❖ Lệnh clear: Xóa màn hình terminal

```
[sinhvien@LinuxServer sinhvien]$ clear ↵
```

Câu 4. Xem thời gian chạy và tải hệ thống (uptime)

- ❖ Lệnh uptime được sử dụng để hiển thị thời gian hoạt động từ lúc hệ thống khởi động. Lệnh cho kết quả và hiển thị trên một dòng gồm thời gian hiện hành, khoảng thời gian hệ thống đã chạy, số người dùng đã đăng nhập và tải trung bình của hệ thống (CPU utilization) cách đây 1, 5, và 15 phút tương ứng.

❖ Ví dụ:

| | | | |
|---|---|--------------------------------------|----------------------------|
| [sinhvien@LinuxServer ~]\$ uptime ↵ | | | |
| 22:13:05 up 1:11, 3 users, load average: 0.05, 0.01, 0.03 | | | |
| Current time | How long the system has been working | Number of users already logged in | Load average (1, 5, 15) |

Câu 5. Lệnh xem trợ giúp

- ❖ Sử dụng lệnh **man** hoặc **info** để xem trợ giúp về lệnh date và uptime
- ❖ Ví dụ:

```
[sinhvien@LinuxServer ~]$ man date ↵
```

```
[sinhvien@LinuxServer ~]$ man uptime ↵
```

```
[sinhvien@LinuxServer ~]$ info date ↵
```

Lưu ý: Để thoát khỏi chương trình xem trợ giúp chúng ta sử dụng tổ hợp phím: **Ctrl + Z**

Câu 6. Xác định user nào hiện đang đăng nhập

```
[sinhvien@LinuxServer ~]$ who ↵
```

```
[sinhvien@LinuxServer ~]$ whoami ↵
```

```
[sinhvien@LinuxServer ~]$ who am i ↵
```

Câu 7. Xác định phiên bản Linux đang làm việc

```
[sinhvien@LinuxServer ~]$ uname ↵
```

```
[sinhvien@LinuxServer ~]$ man uname ↵
```

Câu 8. Xem lịch tương ứng với tháng và năm chỉ định

- ❖ Lệnh: **cal [-j] [month] [year]**
- ❖ Ví dụ: Xem lịch theo dạng số thứ tự ngày của năm

```
[sinhvien@LinuxServer ~]$ cal -j ↵
```

Xem lịch theo tháng, năm:

```
[sinhvien@LinuxServer ~]$ cal 7 2012 ↵
```

```
[sinhvien@LinuxServer ~]$ cal 2050 ↵
```

```
[sinhvien@LinuxServer ~]$ cal -j 2012 ↵
```

Câu 9. Truyền thông điệp (chat)

- ❖ Lệnh: write [username]
- ❖ Thông thường muốn trao đổi với người dùng khác, cần sử dụng lệnh who
- ❖ Ví dụ:

```
[sinhvien@LinuxServer ~]$ who ↵
sinhvien pts/53 2012-06-15 19:34 (10.6.2.250)
[sinhvien@LinuxServer ~]$ write sinhvien ↵
Hi ! Chuc mot buoi ngay vui ve !
<Ctrl+D>
```

Câu 10. Từ chối/cho phép nhập thông điệp trên màn hình bởi lệnh write

- ❖ Lệnh: mesg [n] [y]
- ❖ Trong đó:
 - n (no): Từ chối nhận thông điệp từ người dùng khác.
 - y (yes): Chấp nhận thông điệp từ người dùng khác.

Ví dụ:

```
[sinhvien@LinuxServer ~]$ mesg n ↵
[sinhvien@LinuxServer ~]$ mesg y ↵
```

Câu 11. Gửi/nhận thư điện tử giữa các người dùng

- ❖ Để gửi hoặc nhận mail dùng lệnh:

```
[sinhvien@LinuxServer ~]$ mail <username hoặc địa chỉ mail> ↵
```

- ❖ Ví dụ:

```
[sinhvien@LinuxServer ~]$ mail student
Subject: <gõ nội dung mail>
<Ctrl+D>
Cc: <mail của user nhận khác>
```

- ❖ Để nhận mail dùng lệnh:

```
[sinhvien@LinuxServer ~]$ mail
```

- Trên màn hình sẽ liệt kê các bức thư theo thứ tự 1, 2, 3 ... Để đọc nội dung thư nào, gõ vào số thứ tự của bức thư đó.
- Dấu & nhắc rằng bạn đang ở chương trình đọc thư.
- Để xóa thư đang đọc, tại dấu nhắc bạn gõ: &d
- Để thoát chương trình đọc thư, tại dấu nhắc bạn gõ: &q

- ❖ Ví dụ một phiên gửi mail của user12:

```
[user12@LinuxServer user12]$ mail user15 root ↵
```

Subject: Chao ban

Thuc hanh LINUX

Cc:

[user12@LinuxServer user12]\$

Câu 12. Một số lệnh khác

❖ Xem số hiệu termail đang sử dụng

```
[sinhvien@LinuxServer ~]$ tty ↵
```

❖ Hiện thị các thông báo trên màn hình:

```
[sinhvien@LinuxServer ~]$ echo "Xin chao."
```

```
[sinhvien@LinuxServer ~]$ echo "Hom nay la ngay: " ; date ↵
```

❖ Xem phiên bản Linux đang sử dụng

```
[sinhvien@LinuxServer ~]$ uname -a ↵
```

```
[sinhvien@LinuxServer ~]$ uname -r ↵
```

❖ Kết thúc phiên làm việc

```
[sinhvien@LinuxServer ~]$ logout ↵
```

```
[sinhvien@LinuxServer ~]$ exit ↵
```

❖ Shutdown máy:

```
[root@LinuxServer ~]# halt ↵
```

```
[root@LinuxServer ~]# init 0 ↵
```

```
[root@LinuxServer ~]# shutdown ↵
```

Họ & tên sinh viên:

Mã số sinh viên:

Lớp:

BÀI THỰC HÀNH SỐ 2

✚ Mục đích: Thao tác trên tập tin thư mục

✚ Yêu cầu: Đọc kỹ tài liệu trước khi thực hành

✚ Mục tiêu

- Giúp sinh viên làm quen với các tập lệnh trên hệ thống tập tin, thư mục.
- Phân tích được các đặc điểm hệ thống tập tin Ext3.
- Sử dụng được các lệnh và các tiện ích Linux để thao tác trên tập tin/thư mục và phân quyền.

✚ Nội dung

- Sinh viên login với user đã đăng ký
- Lưu ý: Không login với user root để thực hiện

Câu 1. Một số lệnh cần nhớ

➤ Xem đường dẫn thư mục đang làm việc:

```
[sinhvien@LinuxServer ~]$ pwd ↵
```

➤ Xem nội dung thư mục đang làm việc:

```
[sinhvien@LinuxServer ~]$ ls -la ↵
```

➤ Tạo file văn bản:

```
[sinhvien@LinuxServer ~]$ cat > baitho.txt ↵
```

Hom nay troi nang dep !

Ctrl + D

(nhấn phím **Ctrl+D** để kết thúc việc nhập nội dung file)

➤ Xem file:

```
[sinhvien@LinuxServer ~]$ cat baitho.txt ↵
```

➤ Tạo hai thư mục cnpm, ktm trong thư mục cntt:

```
[sinhvien@LinuxServer ~]$ mkdir cntt ↵
```

```
[sinhvien@LinuxServer ~]$ mkdir cntt/cnpm cntt/ktm ↵
```

➤ Chuyển thư mục làm việc đến ktm:

```
[sinhvien@LinuxServer ~]$ cd cntt/ktm ↵
```

```
[sinhvien@LinuxServer ktm]$
```

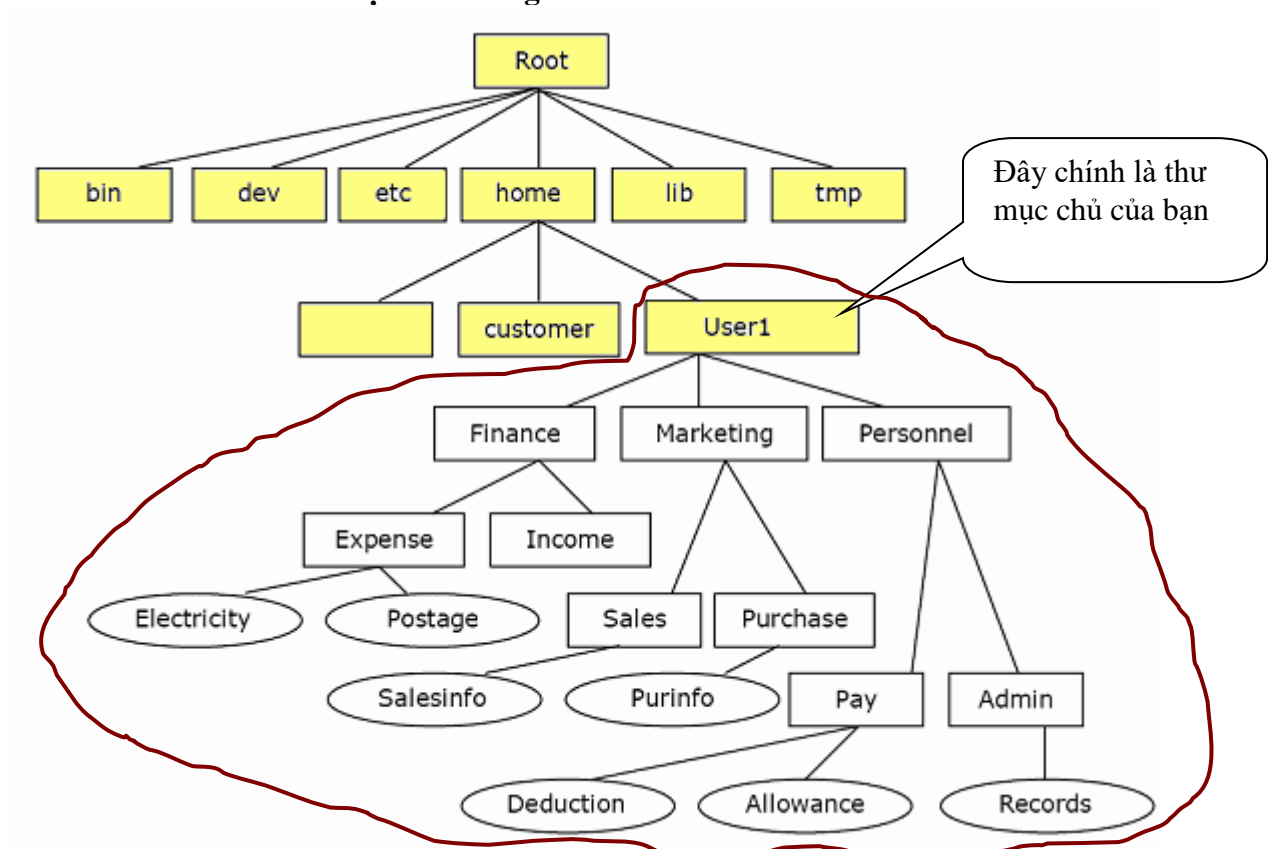
➤ Xóa thư mục cnpm:

```
[sinhvien@LinuxServer ~]$ rmdir cntt/cnpm ↵
```

➤ Hoặc:

```
[sinhvien@LinuxServer ~]$ rm -ri cntt/cnpm ↵
```

Câu 2. Xét cấu trúc thư mục sau và login vào linux:



Thay thế **user1** bằng **thư mục chủ** của bạn và thực hiện các bước dưới đây:

- Tạo cấu trúc thư mục như trên (cho các ô không màu, hình chữ nhật là thư mục, hình elip là tập tin)
- Thực hiện lệnh pwd để xem thư mục hiện hành.
- Liệt kê tất cả các file trong thư mục Expense.

Ví dụ: [sinhvien@LinuxServer ~]\$ ls ~/Finance/Expense ↵

- Chuyển đến thư mục /etc và liệt kê tất cả các file với phần mở rộng *.conf trong thư mục /etc.

Ví dụ: [sinhvien@LinuxServer ~]\$ cd /etc ↵

```
[sinhvien@LinuxServer etc]$ ls -l *.conf ↵
```

- Chuyển đến thư mục chủ của bạn. Hiển thị thư mục hiện hành.
- Copy tất cả các file trong thư mục /home/data đến thư mục chủ của bạn.
- Xem nội dung của file /var/log/dmesg một trang tại một thời điểm.

Ví dụ:

```
[sinhvien@LinuxServer ~]$ cat /var/log/dmesg | more ↵
```

- Copy tất cả các file với phần mở rộng .sh từ thư mục chủ của bạn đến thư mục ~/Finance. Sau đó xác minh xem các file đã được copy chưa.
- Di chuyển tất cả các file với phần mở rộng *.dat từ thư mục chủ của bạn đến thư mục ~/sales/trends.

- Xóa tất cả các file trong thư mục ~/info/remote.
- Xóa tất cả các file trong thư mục ~/sales/trends.

Câu 3. Quyền truy xuất tập tin thư mục

- Tạo chương trình Hello cơ bản:

```
[sinhvien@LinuxServer ~]$ cd /etc ↵  
[sinhvien@LinuxServer ~]$ cat > hello ↵  
    echo "hello."  
    echo "How are you?"  
    Ctrl +D
```

- Chương trình Reply:

```
[sinhvien@LinuxServer ~]$ cat > reply ↵  
    echo "Hi."  
    echo 'I'm fine, thanks. And you?'  
    Ctrl +D
```

- Xem quyền của các file trên:

```
$ ls -l hello reply
```

- Chạy hai chương trình trên:

- B1: Đặt quyền thực thi cho các file trên

```
[sinhvien@LinuxServer ~]$ chmod a+x hello reply ↵
```

- B2: Chạy 2 chương trình trên

```
[sinhvien@LinuxServer ~]$ ./hello ↵
```

```
[sinhvien@LinuxServer ~]$ ./reply ↵
```

- Bỏ quyền thực thi đối với các file trên của các người dùng cùng nhóm (group - g) và những người dùng khác (other - o)

```
[sinhvien@LinuxServer ~]$ chmod go-x hello reply ↵
```

- Hãy thử dùng lệnh chmod với các cách khác (g-rwx, o-rwx, 700,...)?
- Thay đổi quyền truy cập sao cho những người cùng nhóm có khả năng đọc và chạy file hello và reply?
- Thay đổi chủ sở hữu và nhóm sử hữu của file hello và reply trên?

Câu 4. Liên kết tập tin, thư mục

- Tạo hai file bai1.sh và bai2.sh trong thư mục cntt
- Sao chép các file đó thành ?.old:
- Ví dụ:

```
[sinhvien@LinuxServer ~]$ cp bai1.sh bai1.old ↵
```

- Tạo các thư mục source và userbin

```
[sinhvien@LinuxServer ~]$ mkdir source userbin ↵
```

- Di chuyển các file bai1.sh và bai2.sh vào thư mục source, các file bai1.old và bai2.old vào thư mục userbin.
- Tạo liên kết là bailk trong thư mục hiện hành tiếp nhận bai1.sh trong thư mục source.

```
[sinhvien@LinuxServer ~]$ ln source/bai1.sh bailk ↵
```


- Liệt kê thư mục hiện hành:

```
[sinhvien@LinuxServer ~]$ ls -li ↵
```

Có nhận xét gì về những thông tin liệt kê ra?

- Xóa file bai1k và kiểm tra xem có bị xóa chưa và xem điều gì xảy ra với file bai1.sh trong thư mục source.
- Hãy tìm file có tên là bai2.sh:

```
[sinhvien@LinuxServer ~]$ find ~ -name bai2.sh ↵
```

- Xem giá trị i-node của file bai2.sh?
- Tìm tất cả các file có cùng giá trị i-node với bai2.sh?

Họ & tên sinh viên:

Mã số sinh viên:

Lớp:

BÀI THỰC HÀNH SỐ 3

- ✚ Mục đích: Xử lý văn bản và các bộ lọc
- ✚ Yêu cầu: Đọc kỹ tài liệu trước khi thực hành
- ✚ Mục tiêu
 - Sinh viên có khả năng sử dụng Pipe để thực hiện các lệnh Linux
 - Áp dụng các bộ lọc để giải quyết bài toán quản lý sinh viên
- ✚ Nội dung
 - Sinh viên login với user đã đăng ký
 - Lưu ý: Không login với user root để thực hiện

Soạn thảo văn bản với vi:

- Dùng vi để soạn thảo một file có tên là baitho.txt

```
[sinhvien@LinuxServer ~]$ vi baitho.txt ↵
```

- Nhấn **phím i** để nhập nội dung cho file (bài thơ theo sở thích)
- Ghi file và thoát khỏi vi:
 - Nhấn ESC
 - Nhấn **:wq!** ↵

(Lưu ý: Nhấn phím **ESC** → Nhấn phím **:** → Nhấn phím **w** → Nhấn phím **q** → Nhấn phím **!** → Nhấn phím **ENTER**)

Câu 1. Dùng trình soạn thảo vi để tạo file văn bản kehoach.dat và thực hiện các thao tác sau

1. Chuyển sang chế độ lệnh (command mode) và hiển thị số dòng trong file.

```
<Esc>: set number ↵
```

```
<Esc>: set nu ↵
```

2. Xóa dòng 13.
3. Copy dòng 3 và dán (paste) nó vào cuối tài liệu.
4. Undo thao tác dán (paste). Quan sát dòng bị xóa khi thực hiện lệnh undo.
5. Tìm kiếm và thay thế:

```
:1, $s/[old]/[new]/ hoặc :s/text1/text2/g
```

6. Lưu file và thoát khỏi vi

```
:wq! ↵
```

```
:x ↵
```

Câu 2. Giả sử trong thư mục của Binh có file Message.txt chứa các thông tin liên quan đến Binh. Thực hiện các bước sau:

1. Liệt kê tất cả các dòng có chứa chữ Binh trong file
2. Đếm số dòng, số từ, và số ký tự trong file messages có chứa chữ Binh trong đó.

Câu 3. Cho file cơ sở dữ liệu sách, mỗi dòng (bản ghi) trong file này chứa: Book code, Book name, Author Name, Publisher name, and Price.

➤ Ví dụ: File chứa các dữ liệu sau

```
b001:Programming in C++:Tom Wilkins:ABC Books:350
b003:Administering Oracle Databases:Corrine Wallace:New Tech Books:450
b002:Advanced Java: Chris Donaldson:New Tech Books:400
b005:Administering Linux:Nancy Jones:New Tech Books:350
b004:Shell Programming in Linux:Steve Irving:ABC Books:300
b006: Shell Programming in Linux:Nguyen Phuong Lan:Giao duc:1400
```

➤ Yêu cầu

1. Thêm các bản ghi này vào cơ sở dữ liệu.
2. Lọc và hiển thị những sách của Chris Donaldson.
3. Lọc và hiển thị sách có giá 400.
4. Lọc và hiển thị những sách lập trình và chỉ hiển thị tên sách và tác giả.
5. Sắp xếp file dựa trên trường Book code và hiển thị thông tin về sách.
6. Sắp xếp file dựa trên trường Price và hiển thị tên của sách và tác giả tương ứng.
7. Hiển thị bản ghi của các sách có giá từ 300 đến 499.
8. Thay đổi tên sách và tên tác giả thành chữ hoa và hiển thị số sách được viết bởi tác giả Steve Irving.
9. Cho biết hiện tại có bao nhiêu sách?

Câu 4. Thực hiện các lệnh xử lý file sau

- Hiển thị các file thiết bị khối (block) trong thư mục /dev.

```
[sinhvien@LinuxServer ~]$ ls -l /dev | grep "^b" ↵
```

- Hiển thị các file có chứa từ "ram" trong thư mục /dev.

```
[sinhvien@LinuxServer ~]$ ls -l /dev | grep "ram" ↵
```

- Nối theo chiều ngang nội dung của các file Electricity và Postage và lưu kết quả vào một file mới Expense trong cùng thư mục.
- Hiển thị chỉ những dòng lặp lại (duplicate lines) trong file Purinfo.

```
[...]$ cat Purinfo | sort | uniq -d ↵
```

- Lập lịch một tác vụ sao lưu (backup) dữ liệu hàng tuần của các file Salesinfo và Purinfo.
- Lập lịch tác vụ hiển thị thời gian hiện hành chỉ một lần sau hai giờ bắt đầu từ 9 AM đến 7 PM.
- Nén file Records.

Họ & tên sinh viên:

Mã số sinh viên:

Lớp:

BÀI THỰC HÀNH SỐ 4

🚦 Mục đích: Sử dụng shell script cơ bản

🚦 Mục tiêu:

- Làm quen với shell và môi trường shell.
- Phân tích được các đặc điểm của các shell trên Linux

Câu 1. Viết một shell script hiển thị giá trị của các biến môi trường HOME, PATH, HOSTNAME, và LOGNAME theo định dạng sau

HOME =

PATH =

HOSTNAME =

LOGNAME =

- Yêu cầu shell script đặt tên là disp_env.sh.
- Hướng dẫn:

```
[sinhvien@LinuxServer ~]$ vi disp_env.sh ↵
```

```
#!/bin/bash
```

```
echo "HOME = $HOME"
```

```
echo "PATH = $PATH"
```

```
echo "HOSTNAME = $HOSTNAME"
```

```
echo "LOGNAME = $LOGNAME"
```

```
[sinhvien@LinuxServer ~]$ chmod a+x disp_env.sh ↵
```

```
[sinhvien@LinuxServer ~]$ ./disp_env.sh ↵
```

Câu 2. Shell script liệt kê các tập tin trong thư mục hiện hành

```
[sinhvien@LinuxServer ~]$ cat > lietke.sh ↵  
  
    echo "Thư mục hiện hành là: "  
    pwd  
    echo "Danh sach cac tap tin va thư mục con:"  
    ls -la  
  
    Ctrl+D ↵
```

```
[sinhvien@LinuxServer ~]$ chmod a+x lietke.sh ↵
```

```
[sinhvien@LinuxServer ~]$ ./lietke.sh
```

Câu 3. Sử dụng tham số trong một shell script

```
[sinhvien@LinuxServer ~]$ vi thamso.sh ↵  
  
    echo "Ten chuong trình: $0"  
    echo "Tham số thứ nhất: $1"  
    echo "Tham số thứ hai: $2"  
    echo "Tham số thứ ba: $3"  
    echo "Số các tham số: $#"  
    echo "Danh sách tất cả các tham số: $*"  
  
    :wq! ↵ (Lệnh này gõ ở chế độ dòng cuối)  
  
[sinhvien@LinuxServer ~]$ chmod a+x thamso.sh ↵  
  
[sinhvien@LinuxServer ~]$ ./thamso.sh 35 london hanoi ↵
```

Câu 4. Dùng biến ở chế độ hỏi đáp

```
[sinhvien@LinuxServer ~]$ vi menu.sh ↵  
  
    echo "Hay nhap mot gia tri: "  
    read chon  
    echo "Gia tri ban vua nhap: $chon"  
  
    :wq! ↵  
  
[sinhvien@LinuxServer ~]$ chmod u+x menu.sh ↵  
  
[sinhvien@LinuxServer ~]$ ./menu.sh ↵
```

Câu 5. Viết shell script có tên vndate để in ra thời gian hiện tại bằng tiếng Việt giống như lệnh date.

- Ví dụ:

```
[sinhvien@LinuxServer ~]$ ./vndate ↵
```

```
Thu Sau Ngay 15 Thang Sau 20:15:23 Gio Nam 2012
```

Câu 6. Viết shell script có tên tinhong thực hiện tác vụ tính tổng các số nguyên giữa hai số nhập vào trên dòng lệnh (kể cả 2 giá trị nhập).

```
$ ./tinhong 2 6 (kết quả: 2+3+4+5+6 = 20)
```

```
$ ./tinhong 7 3 (kết quả: 3+4+5+6+7 = 25)
```

Yêu cầu:

- Chỉ thực hiện chương trình khi gọi với đủ hai tham số nhập trên dòng lệnh
- Nếu không nhập đủ tham số thì in ra hướng dẫn sử dụng:

Cách dùng: `tinhong <from> <to>`

Họ & tên sinh viên:

Mã số sinh viên:

Lớp:

BÀI THỰC HÀNH SỐ 5

🚦 Mục đích: Lập trình shell nâng cao

🚦 Mục tiêu:

- Phân tích được các cấu trúc lập trình và có khả năng áp dụng để lập trình.
- Áp dụng các thành phần và các cấu trúc lập trình để thực hiện của bài toán đã cho.

Câu 1. Ví dụ chương trình tính tổng $1 \rightarrow n$

➤ Minh họa các cấu trúc while do done, và cách sử dụng [], \$(()).

```
[sinhvien@LinuxServer ~]$ vi tong1.sh ↵
```

```
#!/bin/bash

echo "Chương trình tính tong 1 → $1"

index=0
tong=0
while [ $index -lt $1 ]
do
    index=$((index + 1))
    tong=$((tong + $index))
done
echo "Tong 1 → $1 = $tong"
```

➤ Chạy chương trình:

```
[sinhvien@LinuxServer ~]$ chmod a+x tong1.sh
```

```
[sinhvien@LinuxServer ~]$ ./tong1.sh 100
```

Câu 2. Viết chương trình tính giai thừa của một số

- Minh họa các cấu trúc while, for, và cách sử dụng [], \$(()).

```
[sinhvien@LinuxServer ~]$ vi giaiithua.sh ↵
```

```
[sinhvien@LinuxServer ~]$ ./giaithua 5
```

Câu 3. Viết chương trình kiểm tra năm hiện tại có phải là năm nhuận hay không?**Câu 4. Viết chương trình kiểm tra một số có phải là nguyên tố hay không?**

- Minh họa các cấu trúc while do done, và cách sử dụng [], \$(()).

```
[sinhvien@LinuxServer ~]$ vi giaiithua.sh ↵
```

```
[sinhvien@LinuxServer ~]$ ./giaithua 5
```

Câu 5: BÀI TẬP SHELL NÂNG CAO

Bài 1: Viết shell script tìm giá trị lớn nhất trong ba số a, b, c.

Bài 2: Viết shell script giải phương trình bậc nhất: $ax + b = 0$.

Bài 3: Viết chương trình nhập password, hãy kiểm tra password đó đúng hay sai? In thông báo ra màn hình.

Bài 4: Viết shell script giải phương trình bậc hai:

$$ax^2 + bx + c = 0$$

Bài 5: Viết shell script nhập vào hai số và một phép toán (+, -, x, /) sau đó hiển thị kết quả.

Bài 6: Viết shell script tính điểm trung bình và xếp loại của một thí sinh thi tuyển sinh (có ba cột điểm)

Bài 7: Viết shell script thực hiện nhập vào một giá trị là thứ trong tuần. Hiển thị chữ tiếng anh của thứ đó.

Bài 8. Viết chương trình shell kiểm tra tháng hiện tại là tháng nào? Tháng này Có bao nhiêu ngày?

Bài 9: Tính tổng $s = 1 + 3 + 5 + \dots + n$

Bài 10: Tính giai thừa của một số n!

Bài 11: Viết shell script kiểm tra một số có phải là số nguyên tố không ?

Bài 12: Viết shell script đếm số dòng và số từ trong một tập tin.

Bài 13: Tính tổng $s = 1 + 1/3^2 + 1/5^2 + \dots + 1/(2n+1)^2$

Bài 14: Kiểm tra một số có phải là số hoàn thiện hay không? (n là Số hoàn thiện nếu tổng các ước - <n - của n bằng n)

MỘT SỐ BÀI GẢI MẪU

1) Chương trình Shell Script thực hiện in các số hoàn thiện bé hơn n

```
[root@LinuxServer ~]# vi test.hoanthien
#n>0
m=$1
ht() {
    n=$1
    kt=0
    tonguoc=1
    for ((i=2;i<=n/2;i++))
    do
        if [ $((n%i)) -eq 0 ] ; then
            tonguoc=$((tonguoc+i))
        fi
    done
    if [ $tonguoc -eq $n ] ; then
        kt=1
    fi
    echo "$kt"
}
for ((j=2;j<m;j++))
do
    if [ $(ht $j) = 1 ] ; then
        echo -ne "$j\t"
    fi
done
echo -e "\nKet thuc!"
[root@LinuxServer ~]# chmod u+x test.hoanthien ↵
[root@LinuxServer ~]# ./test.hoanthien 100 ↵
6 28
Ket thuc!
[root@LinuxServer ~]#
[root@LinuxServer ~]# ./test.hoanthien 1000 ↵
6 28 496
Ket thuc!
[root@LinuxServer ~]#
```

2) Chương trình Shell Script thực hiện đếm số dòng của 1 tập tin được đưa vào từ tham biến vị trí.

```
[root@LinuxServer ~]# vi demdong ↵
{
    dem=0
    while read dong
    do
        dem=$((dem+1))
    done
    echo "so dong cua tap tin $1 la: $dem"
}<$1
[root@LinuxServer ~]# chmod u+x demdong ↵
[root@LinuxServer ~]# ./demdong demdong ↵
so dong cua tap tin demdong la: 9
[root@LinuxServer ~]#
```

3) Chương trình Shell Script thực hiện đếm số từ của 1 tập tin được đưa vào từ tham biến vị trí.

```
[root@LinuxServer ~]# vi demtu ↵
{
    dem=0
    while read dong
    do
        for tu in $dong
        do
            dem=$((dem+1))
        done
    done
    echo "so tu = $dem"
}<$1
[root@LinuxServer ~]# chmod u+x demtu ↵
[root@LinuxServer ~]# ./demptu demtu ↵
so tu = 19
[root@LinuxServer ~]#
```

4) Chương trình Shell Script kiểm tra tháng hiện tại có bao nhiêu ngày

```
[root@LinuxServer ~]# vi testthang ↵
thang=`date +%m`
case $thang in
    "01"|"03"|"05"|"07"|"08"|"10"|"12")
        echo "thang $thang co 31 ngay."
        ;;
    "04"|"06"|"09"|"11")
        echo "thang $thang co 30 ngay." ;;
    "02")
        nam=`date +%Y`
        if [ $((($nam%400)) -eq 0) ] ; then
            echo "thang $thang co 29 ngay."
        elif [ $((($nam%4)) -eq 0) ] ; then
            if [ $((($nam % 100)) -ne 0) ] ; then
                echo "thang $thang co 29 ngay."
            else
                echo "thang $thang co 28 ngay."
            fi
        else
            echo "thang $thang co 28 ngay."
        fi
        ;;
    *)
        echo "unknown."
Esac
[root@LinuxServer ~]# chmod u+x testthang ↵
[root@LinuxServer ~]# ./testthang ↵
thang 06 co 30 ngay.
[root@LinuxServer ~]#
```

5) Chương trình Shell, thực hiện in các số nguyên tố bé hơn n. Với n được đưa vào từ tham biến vị trí.

```
[root@LinuxServer ~]# vi testngto.sh
n=$1
testnt(){
    kt=1
    can=$(echo "scale=0; sqrt($1)"|bc -l)
    if [ $1 -lt 2 ] ; then
        kt=0
    else
        for ((i=2;i<=can;i++))
        do
            if [ $((($1%$i)) -eq 0 ] ; then
                kt=0
                break
            fi
        done
    fi
    echo "$kt"
}
for ((j=0;j<=n;j++))
do
    if [ $(testnt $j) = 1 ] ; then
        echo -ne "$j\t"
    fi
done

echo -e '\nKet thuc!'
[root@LinuxServer ~]# ./testngto.sh 100
2 3 5 7 11 13 17 19 23 29
31 37 41 43 47 53 59 61 67 71
73 79 83 89 97
Ket thuc!
[root@LinuxServer ~]#
[root@LinuxServer ~]# cat tinh
#Bài số 5 trong Bài tập làm thêm
case $2 in
"+")
    echo "$1 + $3 = $((($1 + $3))"
    ;;
"-")
    echo "$1 - $3 = $((($1 - $3))"
    ;;
"x")
    echo "$1 x $3 = $((($1 * $3))"
    ;;
"/")
    if [ $3 -ne 0 ] ; then
        echo "$1 / $3 = $(echo "scale=4;$1/$3" | bc -l)"
    else
        echo "khong the chia cho 0."
    fi
    ;;
```

```
*)
    echo "sai toan tu."
esac
[root@LinuxServer ~]# ./tinh 3 + 5
3 + 5 = 8
[root@LinuxServer ~]#
```

```
[root@LinuxServer ~]# cat uscbasc.sh
#Chương trình tìm UOC SO CHUNG LON NHAT & BOI SO CHUNG NHO NHAT
ucln(){
n=$1
m=$2
while [ $n -ne 0 -a $m -ne 0 ]
do
    if [ $n -gt $m ] ; then
        n=$(( $n-$m ))
    else
        m=$(( $m-$n ))
    fi
done
if [ $n -eq 0 ] ; then
    echo "$m"
else
    echo "$n"
fi
}
echo "USCLN($1,$2) = $(ucln $1 $2)"
echo "BSCNN($1,$2) = $(echo "scale=0; $1*$2/$(ucln $1 $2)" | bc
-l)"
[root@LinuxServer ~]#
[root@LinuxServer ~]#
[root@LinuxServer ~]# ./uscbasc.sh 20 32
USCLN(20,32) = 4
BSCNN(20,32) = 160
[root@LinuxServer ~]#
```

Cho 1 file csdl.db:

File csdl.db có 2 cột, được cách nhau bởi dấu 2 chấm ":".

Cột 1 cho biết username; Cột 2 lưu password.

Hãy nhập tạo file csdl.db có nội dung như sau:

| |
|---|
| k15tcd11:12345 kk15tcd123:123 k15tcd1:k15 sangnv:sang longvt:long |
|---|

Hãy viết chương trình Shell đăng nhập, cho phép người dùng nhập vào username & password. Hãy kiểm tra username & password vừa nhập vào có đúng không? Nếu sai thì phải cho biết sai username hay sai password. Dữ liệu được so sánh từ file csdl.db.

```
[root@LinuxServer ~]# ls
login.sh userpass.db
[root@LinuxServer ~]# cat userpass.db
#Cot 1: Username ; Cot 2: Password
d16tmt:123
dd16tmt:1234
dd16tmt11:abc
dd16tmt123:xyz
[root@LinuxServer ~]#
```

```
[root@LinuxServer ~]# cat login.sh
dem=0
#ktra=1 co nghĩa là đang nhập thành công
ktra=0
while [ $dem -le 2 -a $ktra -eq 0 ]
do
    dem=$((dem+1))
    echo -n "Nhập Username: " ; read u
    echo -n "Nhập Password: " ; read p
    u1=`cat userpass.db | grep "^$u:" | cut -d':' -f1`
    p1=`cat userpass.db | grep "^$u:" | cut -d':' -f2`
    if [ $u = "$u1" ] ; then
        if [ $p = "$p1" ] ; then
            echo "Đang nhập thành công."
            ktra=1
        else
            echo "Password sai."
        fi
    else
        echo "Username sai."
    fi
done
echo "Kết thúc!"
[root@LinuxServer ~]#
[root@LinuxServer ~]# ./login.sh
Nhập Username: d16tmt
Nhập Password: 123
Đang nhập thành công.
Kết thúc!
[root@LinuxServer ~]#
```

Viết chương trình kiểm tra xem 1 user có tồn tại trên hệ thống hay không? Nếu có, in ra số định danh của user này (UID), thư mục HOME của user và tất cả các user cùng group với user này. (username được nhập từ bàn phím)

```
[root@LinuxServer ~]# cat testuser.sh
#Chương trình kiểm tra 1 user có tồn tại trên hệ thống không?
#Nếu có thì cho biết UID và HOME DIRECTORY của user này?
u=$1
user=`cat /etc/passwd | grep "^$u:" | cut -d':' -f1`
uid=`cat /etc/passwd | grep "^$u:" | cut -d':' -f3`
gid=`cat /etc/passwd | grep "^$u:" | cut -d':' -f4`
homedir=`cat /etc/passwd | grep "^$u:" | cut -d':' -f6`
if [ "$u" = "$user" ]; then
    echo "User $u tồn tại trong hệ thống."

    uid=`cat /etc/passwd | grep "^$u:" | cut -d':' -f3`
    echo "UserID của $u là: $uid"
    homedir=`cat /etc/passwd | grep "^$u:" | cut -d':' -f6`
    echo "Thư mục chủ của $u là: $homedir"
    echo "=====
    echo "Các user cùng nhóm với $u:"
    gid=`cat /etc/passwd | grep "^$u:" | cut -d':' -f4`
    echo "`cat /etc/passwd | cut -d':' -f1,4 | grep $gid | cut -
d':' -f1`"
    echo "=====
else
    echo "User $u không tồn tại trong hệ thống."
fi
[root@LinuxServer ~]# . testuser.sh lamdc
User lamdc tồn tại trong hệ thống.
UserID của lamdc là: 594
Thư mục chủ của lamdc là: /home/lamdc
=====
Các user cùng nhóm với lamdc:
thamlv
binhntt
dongpx
hoanghp
lamdc
=====
[root@LinuxServer ~]#
```

Viết shell script có tên vndate để in ra thời gian hiện tại bằng tiếng Việt giống như lệnh date.

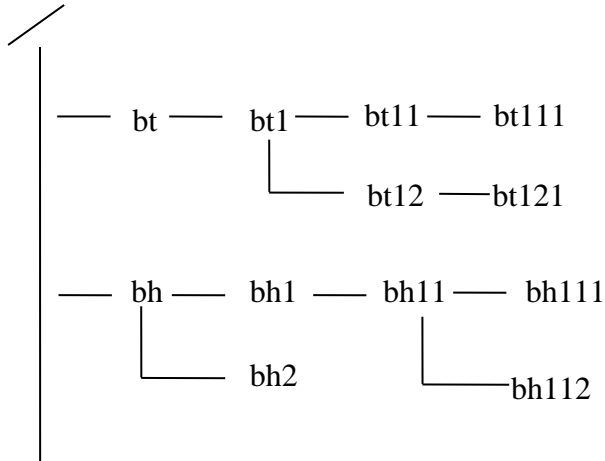
```
[root@LinuxServer ~]# cat vndate.sh
#Chương trình Shell in ra thời gian hiện tại bằng tiếng Việt
#Bây giờ là: 18 giờ 20 phút 30 giây, thứ tư ngày 30 tháng 03 năm 2011
gio=`date +%k`
phút=`date +%H`
```

```
giay=`date +%S`
thu=`date +%u`
ngay=`date +%d`
thang=`date +%m`
nam=`date +%Y`
case $thu in
  1)
    thul="thu Hai";;
  2)
    thul="thu Ba";;
  3)
    thul="thu Tu";;
  4)
    thul="thu Nam";;
  5)
    thul="thu Sau";;
  6)
    thul="thu Bay";;
  7)
    thul="Chu Nhat";;
esac
echo -n "Bay gio la: "
echo -n "$gio gio $phut phut $giay giay, "
echo -n "$thul, "
echo "ngay $ngay thang $thang nam $nam."
echo "Ket thuc!"
[root@LinuxServer ~]# . vndate.sh
Bay gio la: 20 gio 20 phut 53 giay, thu Tu, ngay 30 thang 03 nam
2011.
Ket thuc!
[root@LinuxServer ~]#
```

Bài tập thực hành linux

Quản trị tập tin:

1/ Tạo các thư mục sau:



- 2/ Sử dụng lệnh “cat” tạo các tập tin sau với nội dung bất kỳ: /bt/a1.txt, /bt/bt1/a2.txt, /bh/bh1/a3.txt
- 3/ Nối 2 tập tin /bt/a1.txt và tập tin /bh/bh1/a3.txt thành tập tin /bh/bh2/a4.txt
- 4/ Tạo thư mục /bt/bt2 là hard link của thư mục /bt, có tạo được không ? Giải thích.
- 5/ Tạo tập tin /bt/bt1/bt11/a5.txt là soft link của tập tin /bt/a1.txt
- 6/ Tạo tập tin /bh/bh2/a6.txt là hard link của tập tin /bh/bh2/a4.txt, sử dụng lệnh ls để xem inode và link count của 2 tập tin này. Giải thích.
- 7/ Tạo thư mục /bt/bt1/bt13 là soft link của thư mục /bh/bh2. So sánh sự khác nhau của lệnh “cd -P /bt/bt1/bt13” và lệnh “cd -L /bt/bt1/bt13”
- 8/ Chuyển vào thư mục /etc, so sánh và giải thích kết quả của các lệnh sau: ls, ls -i , ls -a, ls -ila. Thực hiện hiển thị kết xuất các lệnh trên theo từng trang màn hình.
- 9/ Cho biết thông tin về số dòng, số từ và kích thước của tập tin /bt/a1.txt
- 10/ Tạo tập tin /bt/a7.txt có 12 dòng, sử dụng các lệnh “tail” “head” để hiển thị từ nội dung dòng thứ 4 tới dòng thứ 10 của tập tin /bt/a7.txt ra màn hình.
- 11/ mount đĩa USB vào thư mục /bt/bt1/bt12, hãy liệt kê nội dung của thư mục /bt/bt1/bt12. giải thích.
- 12/ Dùng lệnh “rmkdir” để xóa thư mục /bh/bh1/bh11. Cho biết kết quả, giải thích.
- 13/ Copy tập tin /bt/a7.txt vào thư mục /bh và đổi tên thành /bh/a9.txt
- 14/ Tìm tất cả các tập tin có phần tên mở rộng là .conf trong thư mục /etc và các thư mục con của nó.
find /etc -name “*.conf”
- 15/ Tìm trong thư mục /etc các tập tin có phần mở rộng là .txt và có chủ nhân là “root” rồi copy vào thư mục /bh/bh1
find /etc -name “*.txt” -user root -exec cp {} /bh/bh1 \;
- 16/ Tìm trong thư mục hiện hành các tập tin có kích thước nhỏ hơn 300kb và được truy cập trong vòng 30 phút gần đây.
find . -size -300k -amin -30
- 17/ Tìm trong thư mục /etc tất cả các soft link.

find /etc –type l

18/ Sử dụng trình soạn thảo vi để soạn thảo tập tin có tên file1.txt có nội dung bất kỳ. thực hiện các tính năng thêm, xoá, sửa, sao chép, di chuyển, tìm kiếm, ... trong tập tin file1.txt.

Quản trị tài khoản và quyền tập tin:

19/ Thực hiện, giải thích câu lệnh và kết quả của từng lệnh dưới đây. Sau khi thực hiện mỗi lệnh, kiểm tra nội dung của các tập tin /etc/passwd, /etc/shadow, /etc/group và thư mục /home xem có những thay đổi gì?

```
useradd user1; useradd user2
```

```
useradd -u 700 -o user3
```

```
useradd -s /sbin/nologin user4
```

```
useradd -s /dev/null user5
```

```
groupadd nhóm1
```

```
groupadd nhóm2
```

```
usermod -G nhóm1, nhóm2 user1
```

```
usermod -G nhóm2 user2
```

```
usermod -G nhóm2 user3
```

```
usermod -l user55 user5
```

```
useradd -u 0 -o admin
```

20/ Thực hiện lệnh passwd để gán mật mã truy nhập cho các tài khoản trên. Khảo sát tập tin /etc/passwd, /etc/shadow xem có gì thay đổi.

21/ Chuyển sang tty2 (nhấn tổ hợp phím alt+F2), đăng nhập hệ thống bằng user4. Có đăng nhập được không? Giải thích.

22/ Thực hiện lần lượt:

- Khóa tài khoản user1. Tìm sự thay đổi trong /etc/shadow
- Mở khóa tài khoản user1. Tìm sự thay đổi trong /etc/shadow
- Xóa mật mã tài khoản user1. Tìm sự thay đổi trong /etc/shadow

23/ Tạo thư mục /baitap và tập tin /baitap/abc.txt (nội dung bất kỳ). Xác định nhóm, chủ nhân và quyền của thư mục, tập tin vừa tạo?

24/ Xem quyền mặc định khi tạo tập tin bằng lệnh umask -S. Thực hiện thay đổi quyền mặc định khi tạo tập tin, sau đó tạo tập tin abc1.txt và thư mục tm1 (trong /baitap) để kiểm chứng.

Cho nhận xét về quyền của tập tin mới tạo khi quyền mặc định có quyền x.

25/ Dùng lệnh chmod để thay đổi lại quyền cho các tập tin trong /baitap, sử dụng cả phương pháp tượng trưng và tuyệt đối (dùng lệnh ls -l để kiểm chứng kết quả)

26/ Thực hiện và giải thích

- Thực hiện lệnh **mkdir /baitap2 ; chmod 777 /baitap2**
- Đăng nhập với tài khoản user2, tạo một tập tin có tên “tap tin cua user2.txt” trong /baitap2.
- Đăng nhập với tài khoản user3, thực hiện sửa, xóa tập tin do user2 tạo. Cho biết kết quả.
- Thực hiện lệnh **chmod 1777 /baitap2 ; ls -l /baitap2**. Kết quả?
- Đăng nhập với quyền user2, tạo một tập tin có tên “tap tin 2 cua user2.txt” trong /baitap2.
- Đăng nhập với tài khoản user3, thực hiện sửa, xóa tập tin do user2 tạo. Cho biết kết quả.

27/ Thực hiện tuần tự và giải thích

- **mkdir /baitap**
- Tạo tập tin /baitap/abc.txt có nội dung bất kỳ
- **chmod 700 /baitap/abc.txt**. Đăng nhập với tài khoản user2, và mở xem tập tin /baitap/abc.txt. Cho biết kết quả?. Giải thích.
- Đổi chủ nhân tập tin abc.txt thành user2. Đăng nhập với tài khoản user2, và truy xuất tập tin /baitap/abc.txt. Cho biết kết quả?
- Đăng nhập với tài khoản user3, và truy xuất tập tin /baitap/abc.txt. Cho biết kết quả?. Giải thích.
- Thực hiện lệnh **chmod 755 /baitap/abc.txt && chown :nhom2 /baitap/abc.txt**. Đăng nhập với quyền user3, và truy xuất tập tin /baitap/abc.txt. Cho biết kết quả?. Giải thích.

28/ Tạo một symbolic link cho một tập tin bất kỳ. Tiến hành thay đổi quyền của symbolic link mới tạo này. Cho biết kết quả.

Tiến trình

1/ Xem danh sách các tiến trình bằng lệnh “top” và “ps -ef”, khảo sát kết quả

2/ Thực hiện

- Tại tty1 (Alt-F1), thi hành lệnh **ps -aux | grep tty**
- Đăng nhập vào quyền root tại tty5 (Alt-F5) và tty6 (Alt-F6).
- Tại tty1 (Alt-F1), thi hành lệnh **ps -aux | grep tty**. Cho biết kết quả
- Tại tty1, thực hiện kết thúc tiến trình bash tại tty5 và tty6 bằng lệnh kill. Chuyển sang tty5 và tty6 xem kết quả.

3/ Thực hiện lệnh **kill -9 1**. Cho biết kết quả, nguyên nhân

4/ Thực hiện tuần tự các lệnh, giải thích ý nghĩa và kết quả

- `/etc/rc.d/init.d/httpd stop ; ps -aux | grep httpd ; service httpd status`
- `service httpd start ; ps -aux | grep httpd ; /etc/rc.d/init.d/httpd status`

5/ Thực hiện tuần tự, giải thích ý nghĩa và kết quả

- `chkconfig --list | more`
- `chkconfig --list httpd`
- `chkconfig --delete httpd ; ls /etc/rc[0-6].d | grep httpd`
- `chkconfig --add httpd ; ls /etc/rc[0-6].d | grep httpd`
- `chkconfig --list httpd && chkconfig --levels 2345 httpd on ; chkconfig --list httpd ; ls /etc/rc[0-6].d | grep httpd`

6/ Mở tập tin `/etc/inittab`, tìm đến dòng `id:3:initdefault` và sửa số 3 thành số 1. Khởi động lại máy bằng lệnh `reboot`. Hãy cho biết kết quả.

7/ Dùng lệnh `crontab -e` để tạo lịch thi hành có nội dung dưới đây.

```
* * * * * echo "hello" ; date > /lich.txt
30 4-10 2 7,10 * echo "lenh thi hanh luc `date`" > /lich2.txt
```

Cho biết kết quả và giải thích.

8/ Thi hành lệnh các lệnh sau ở chế độ background

- `yes >/dev/null`
- `yes hello >/dev/null`
- `yes abc >/dev/null`

Dùng lệnh `jobs` để xem danh sách các công việc đang thực thi ở chế độ background

9/ Thi hành lệnh sau ở chế độ foreground : `yes chaoban >/dev/null`, chuyển lệnh trên vào chế độ background.

1/ Thực hiện các lệnh sau đây, và giải thích:

```
rpm -qa |more
```

```
rpm -qa |grep samba
```

```
rpm -qf /etc/passwd
```

```
rpm -ql samba
```

2/ Sử dụng lệnh rpm để cài đặt 1 số chương trình có trong bộ đĩa cài đặt của CentOS5.

3/ Hãy tạo 1 kho có tên là “softwares” để lưu trữ các phần mềm có trong bộ đĩa cài đặt của CentOS5. Sau đó sử dụng lệnh yum để cài đặt các phần mềm có trong kho này.

4/ Gắn thêm 1 ổ cứng cho máy ảo linux, chia ổ này thành 2 phân vùng. Định dạng 2 phân vùng này với định dạng ext3, sau đó mount tự động 2 phân vùng này vào thư mục /data1 và /data2.

5/ Thực hiện các lệnh sau đây, và giải thích:

```
touch a.txt b.txt c.txt d.txt e.txt f.txt
```

```
tar cvf lt1.tar a.txt b.txt
```

```
tar tvf lt1.tar
```

```
tar cvf lt2.tar c.txt d.txt
```

```
tar Avf lt1.tar lt2.tar
```

```
tar rvf lt1.tar e.txt
```

```
tar czvf lt3.tar.gz a.txt b.txt c.txt
```

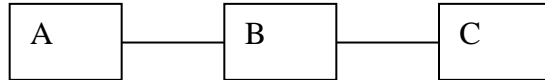
```
tar xvzf lt3.tar.gz
```

```
tar cjvf lt4.tar.bz2 a.txt b.txt c.txt
```

```
tar xvjf lt4.tar.bz2
```

6/ Hãy thực hiện các bước để bảo mật cho chương trình GRUB.

1. Xem thông tin cấu hình IP của card mạng
2. Thay đổi địa chỉ IP của card mạng
 - a. Dùng lệnh
 - b. Dùng file script
3. Tạo địa chỉ IP alias cho card mạng
 - a. Dùng lệnh
 - b. Dùng file script
4. Sử dụng các lệnh sau để kiểm tra mạng: ping, netstat, tracer, tcpdump
5. Cho mô hình sau:



máy B có 2 card mạng

Cấu hình địa chỉ IP cho các máy như sau:

A: IP: 192.168.10.2/24, Gateway: 192.168.10.1

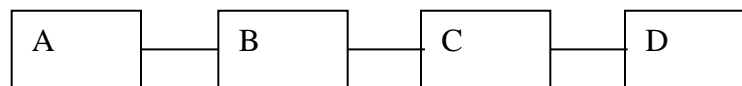
B: eth0: 192.168.10.1/24, eth1: 192.168.11.1/24

C: 192.168.11.2/24, Gateway: 192.168.11.1

Tắt chức năng ip_forward trên máy B, từ máy A ping máy C và ngược lại, cho biết kết quả, giải thích.

Bật chức năng ip_forward trên máy B, từ máy A ping máy C và ngược lại, cho biết kết quả, giải thích

6. Cho mô hình sau:



Máy B, máy C có hai card mạng.

Cấu hình địa chỉ IP cho các máy như sau:

A: 192.168.10.2/24, Gateway: 192.168.10.1

B: eth0: 192.168.10.1/24, eth1: 192.168.11.1/24

C: eth0: 192.168.11.2/24, eth1: 192.168.12.1/24

D: 192.168.12.2/24, Gateway: 192.168.12.1

Cấu hình static route sao cho các máy ping thấy nhau.