Hệ điều hành Linux

dqmdang@hcmus.edu.vn

Nội dung

Giới thiệu về hệ điều hành Linux và cách người dùng tương tác với Linux bằng command qua giao diện CLI cung cấp bởi shell:

- Khái niệm hệ điều hành
- Kernel và chức năng của kernel
- Các khái niệm trừu tượng trong Linux
- Làm việc với shell
- Các command cơ bản
- Các tính năng của shell

Operating System

Operating System (OS)

- Hệ điều hành là tập hợp phần mềm gồm nhân (kernel), các thư viện (lib) và các chương trình mặc định (program/application)
- Ubuntu, CentOS, Fedora, ... là các hệ điều hành Linux (Linux distribution) bởi vì chúng sử dụng kernel mã nguồn mở **Linux**
- Đôi khi thuật ngữ OS được dùng để ám chỉ riêng Kernel và thuật ngữ Linux được dùng để ám chỉ OS sử dụng Linux kernel !

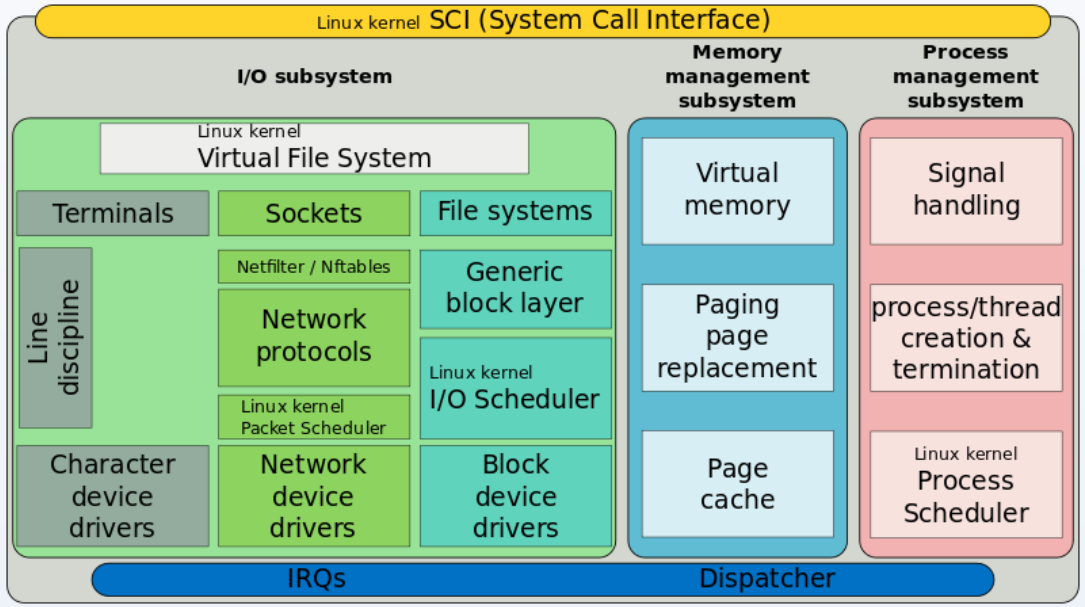
A Linux distribution OS

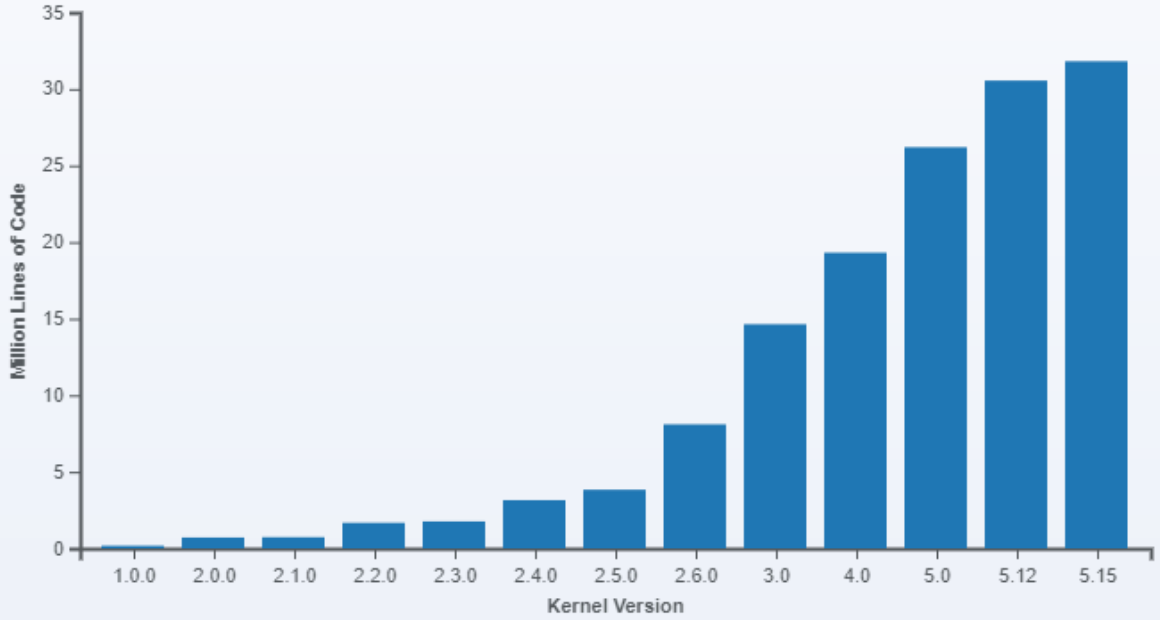
User mode	**User applications**	*bash, LibreOffice, GIMP, Blender, 0 A.D., Mozilla Firefox, ...*				
	System components	**init daemon**: *OpenRC, runit, systemd...*	**System daemons**: *polkitd, smbd, sshd, udevd...*	**Window manager**: *X11, Wayland, SurfaceFlinger (Android)*	**Graphics**: *Mesa, AMD Catalyst, ...*	**Other libraries:** *GTK, Qt, EFL, SDL, SFML, FLTK, GNUstep, ...*
	C standard library	`fopen`, `execv`, `malloc`, `memcpy`, `localtime`, `pthread_create` ... (up to 2000 subroutines) *glibc* aims to be fast, *musl* and *uClibc* target embedded systems, *bionic* written for Android, etc. All aim to be POSIX/SUS-compatible.				
Kernel mode	**Linux kernel**	`stat`, `splice`, `dup`, `read`, `open`, `ioctl`, `write`, `mmap`, `close`, `exit`, etc. (about 380 system calls) The Linux kernel System Call Interface (SCI, aims to be POSIX/SUS-compatible)[104]				
		Process scheduling subsystem	IPC subsystem	Memory management subsystem	Virtual files subsystem	Network subsystem
		Other components: ALSA, DRI, evdev, klibc, LVM, device mapper, Linux Network Scheduler, Netfilter Linux Security Modules: *SELinux*, *TOMOYO*, *AppArmor*, *Smack*				
Hardware (CPU, main memory, data storage devices, etc.)						

Kernel

- Là chương trình phần mềm khởi động trước các chương trình khác
- Có nhiệm vụ:
  - Quản lý (manage), trừu tượng hóa (abtract) và chia sẻ (share) các tài nguyên phần cứng máy tính (**computer hardware**) cho các chương trình đang chạy (**process**)
  - Chạy các process “đồng thời” (**concurrently**) một cách tách biệt (**isolated**), và cung cấp các dịch vụ (**services**) (ví dụ như đọc/ghi file) cho các chương trình này thông qua các **system calls**

Linux kernel





Các khái niệm trừu tượng trong Linux

- Kernel Linux sử dụng một số khái niệm trừu tượng để biểu diễn cho các đối tượng, các chi tiết thực hiện ở mức thấp
- Các khái niệm này được phản ánh thông qua các system calls mà Linux cung cấp. Ví dụ: system call *open* (mở một file) sử dụng các khái niệm trừu tượng là file, path, và file descriptor
- Một số khái niệm trừu tượng cơ bản trong Linux là file, path, pipe, file descriptor, process, ...

Program vs Process

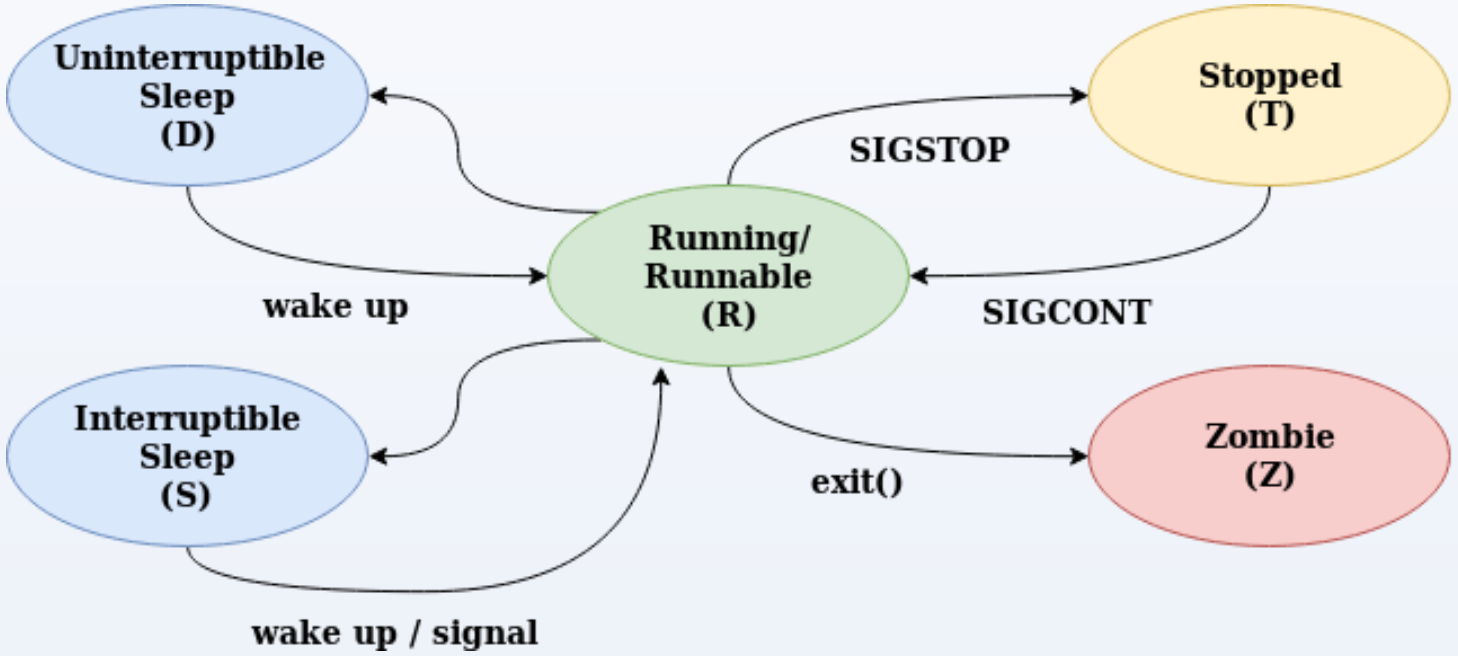
- Program là một chương trình phần mềm, lưu trữ trong ổ cứng dưới dạng một file. Thông thường program nằm ở thư mục /bin, /sbin, /usr/bin hoặc /usr/sbin
- Process là một program đang chạy. Có thể có nhiều process chạy cùng lúc từ một program duy nhất. Mỗi process sẽ chiếm dụng một phần CPU và RAM của hệ thống
- Mỗi process được định danh bằng một con số duy nhất gọi là process id (pid)

Signal

- Signal là sự kiện được xử lý bởi process
- Process có thể chọn xử lý một signal hoặc không, xử lý theo mặc định hoặc tự xử lý riêng (ngoại trừ SIGKILL và SIGSTOP)

Signal	**Xử lý mặc định**
SIGTERM	kết thúc thực thi
SIGKILL	kết thúc thực thi (bắt buộc)
SIGTSTP (Ctrl + Z)	tạm dừng thực thi
SIGSTOP	tạm dừng thực thi (bắt buộc)
SIGCONT	tiếp tục thực thi
SIGINT (Ctrl + C)	kết thúc thực thi
SIGQUIT (Ctrl + \)	kết thúc và tạo core dump
SIGALRM (tín hiệu định kì cấu hình được)	kết thúc thực thi

Process State



File

- Có một câu thành ngữ trong Linux: "*In Linux, everything is a File*"
- Khái niệm file trong Linux mang nghĩa rộng hơn thông thường. File trong Linux có thể đại diện cho:
  - Regular file: Một chuỗi byte dữ liệu lưu trên ổ cứng máy tính (hoặc RAM)
  - Character/block device special file: Thiết bị phần cứng hoặc thiết bị ảo (virtual)
  - Named pipe: Bộ nhớ đệm để trao đổi dữ liệu giữa các process
  - Virtual/pseudo file: phương tiện để trao đổi dữ liệu với kernel
- Mỗi file có tên và đường dẫn để xác định vị trí của file trong file system
- Directory cũng là một file trong Linux !

File descriptor

- File descriptor là một số nguyên (>=0) định danh một đối tượng có thể đọc/ghi (File, Pipe hoặc Network Socket) mà một process đang tương tác
- Khi một file descriptor được tạo ra cho một process, nó luôn nhận giá trị nhỏ nhất chưa được sử dụng
- Một process thường được khởi tạo sẵn 3 file descriptor mặc định:
  - 0: standard input (stdin)
  - 1: standard output (stdout)
  - 2: standard error (stderr)

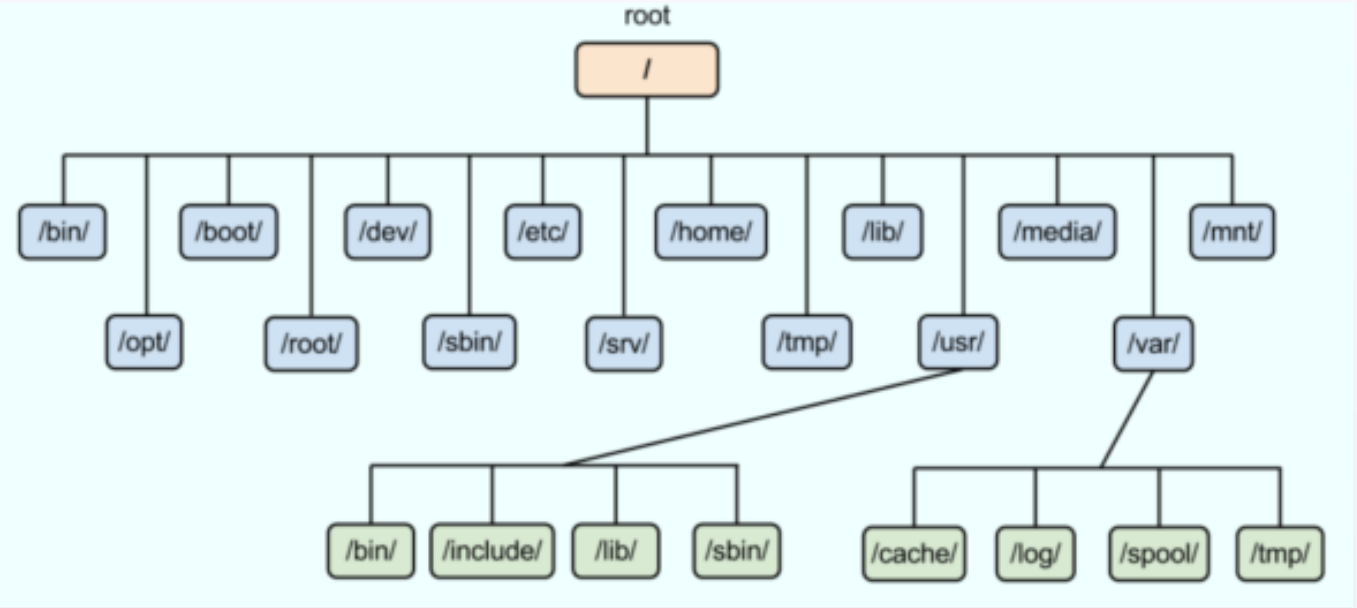
File descriptor

Sử dụng File descriptor mang lại sự linh động và lợi ích lớn cho Linux:

- Program sử dụng cùng một system calls để đọc/ghi tất cả các đối tượng có thể đọc/ghi được => Giúp đơn giản hóa việc viết code
- Cùng một program được sử dụng để đọc/ghi nhiều đối tượng khác loại => tăng tính linh động và khả năng sử dụng của một program

File system

- /: thư mục gốc (root)
- boot: kernel và bootloader
- bin/sbin: các chương trình
- etc: các file cấu hình
- home: dữ liệu người dùng
- lib: các thư viện
- usr: có tổ chức tương tự root
- var: file log, email, ...
- proc: thông tin của process
- dev: device files



Path

- Đường dẫn tuyệt đối (Absolute pathname): Đường dẫn tới file/directory bắt đầu từ root. Ví dụ: */home/ubuntu/workspace/lab/code.c*
- Đường dẫn tương đối (Relative pathname): Đường dẫn tới file/directory bắt đầu từ thư mục hiện tại. Ví dụ: *workspace/lab/code.c*
- Đường dẫn (tên) của một số thư mục đặc biệt:

/	Thư mục root
~	Thư mục home của user
.	Thư mục hiện tại
..	Thư mục cha của thư mục hiện tại

/dev

- Thư mục đặc biệt chứa các device files. Mỗi file tượng trưng cho một device (speaker, printer, HDD, ...). Program có thể giao tiếp với một device bằng cách đọc/ghi device file tương ứng
- Có 2 loại device:
  - Character device: Device gửi/nhận dữ liệu theo chuỗi các byte liên tiếp. Ví dụ: Speaker, printer
  - Block device: Device có khả năng lưu trữ dữ liệu theo block, và có thể đọc/ghi một block bất kì. Ví dụ: HDD

/proc

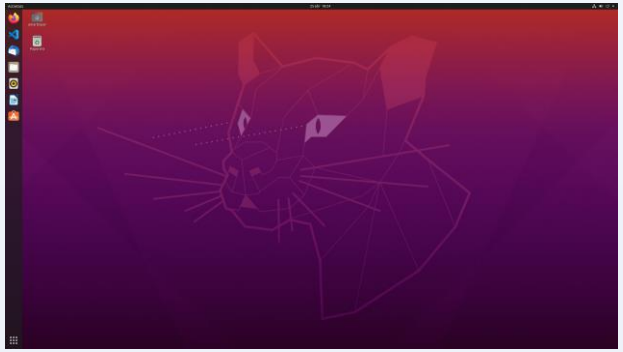
- Thư mục đặc biệt chứa các virtual/pseudo file, giúp process trao đổi dữ liệu với kernel
- Mỗi process sẽ có một thư mục tương ứng đặt tên theo pid của process

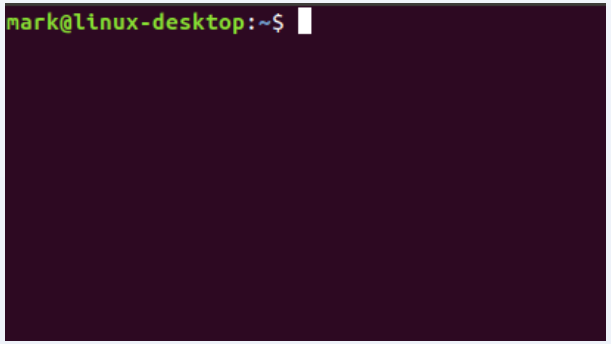
/proc/cpuinfo	Thông tin CPU	/proc/<pid>/maps	Không gian địa chỉ của process
/proc/meminfo	Thông tin memory	/proc/<pid>/exe	File program của process
/proc/devices	Thông tin các device	/proc/<pid>/comm	Tên program của process
/proc/version	Thông tin phiên bản	/proc/<pid>/cmdline	Câu lệnh tạo process

Shell

GUI vs CLI

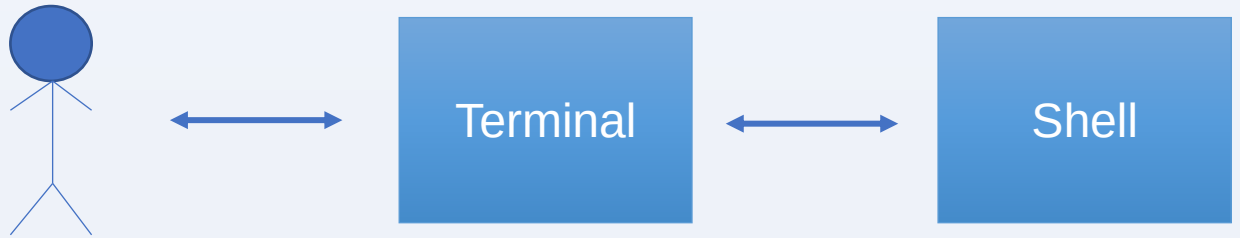
- Các hệ điều hành Linux có thể có giao diện đồ họa (GUI) hoặc không, nhưng luôn có giao diện dòng lệnh (CLI)
- Giao diện CLI có độ linh hoạt cao, có thể khai thác tối đa sức mạnh của hệ điều hành





CLI, Shell, Terminal

- Giao diện CLI giao tiếp với người dùng bằng văn bản (text)
- Shell là ứng dụng cung cấp giao diện CLI cho người dùng. Có nhiều ứng dụng shell khác nhau: bash, sh, zsh, ash, dash ...
- Terminal là ứng dụng kết nối tới shell, và tương tác trực tiếp với người dùng. Terminal có thể nằm ở 1 máy tính khác với shell



Shell prompt

```
[me@linuxbox ~]$
```

- Prompt là đoạn text xuất hiện khi shell sẵn sàng nhận lệnh từ người dùng
- Prompt (bash shell) có dạng:

[Tên người dùng + @ + Tên máy + Thư mục hiện tại]$

Command

- Command là 1 đoạn text do người dùng nhập và được thực thi bởi shell
- Một command có dạng tổng quát *name -options agruments*
- Trong đó:
  - name: tên của command
  - options: các lựa chọn tùy biến
  - agruments: các tham số
- Ví dụ: *ls -l workspace*

Command cơ bản

ls <paths>	Liệt kê nội dung các dir <paths>	stat <paths>	Hiển thị thông tin của các file <paths>
ls -alF <paths>	Liệt kê chi tiết nội dung các dir <paths>	file <paths>	Xác định loại của các file <paths>
pwd	Hiển thị path của thư mục làm việc hiện tại	less <path>	Xem nội dung file <path>
cd <path>	Thay đổi thư mục làm việc tới <path>	chmod <mode> <path>	Thay đổi mode của file tại <path> sang <mode>
mkdir <paths>	Tạo ra các dir <paths>	touch <paths>	Tạo ra các file <paths>

Command cơ bản

free -h	Kiểm tra dung lượng RAM	type <name>	Loại của <name>, với <name> là tên command
df -h	Kiểm tra dung lượng ổ cứng	whatis <name>	Mô tả ngắn gọn của <name>
uname -a	Hiển thị thông tin kernel, OS, platform	which <name>	Tìm đường dẫn của <name>, với <name> là tên program
ps	Hiển thị thông tin process trong phiên làm việc	man <name>	Hiển thị tài liệu trợ giúp của <name>
ps aux	Hiển thị thông tin tất cả process		

Command cơ bản

wc -l <path>	Đếm số line của file <path>	grep <name> <paths>	Hiển thị các dòng có chứa <name> trong các file <paths>
wc -w <path>	Đếm số word của file <path>		
echo <args>	Hiển thị các tham số <args> nhận được		
cat <paths>	Hiển thị nội dung các files <paths>		

Command cơ bản

tar · pv · cat · tac · chmod · grep · diff · sed · ar · man · pushd · popd · fsck · testdisk · seq · fd · pandoc · cd · $PATH · awk · join · jq · fold · uniq · journalctl · tail · stat · ls · fstab · echo · less · chgrp · chown · rev · look · strings · type · rename · zip · unzip · mount · umount · install · fdisk · mkfs · rm · rmdir · rsync · df · gpg · vi · nano · mkdir · du · ln · patch · convert · rclone · shred · srm

alias · screen · top · nice · renice · progress · strace · systemd · tmux · chsh · history · at · batch · free · which · dmesg · chfn · usermod · ps · chroot · xargs · tty · pinky · lsof · vmstat · timeout · wall · yes · kill · sleep · sudo · su · time · groupadd · usermod · groups · lshw · shutdown · reboot · halt · poweroff · passwd · lscpu · crontab · date · bg · fg

netstat · ping · traceroute · ip · ss · whois · fail2ban · bmon · dig · finger · nmap · ftp · curl · wget · who · whoami · w · iptables · ssh-keygen · ufw

Hầu hết các command là program được viết theo triết lý sau:

- Write programs that do one thing and do it well.
- Write programs to work together.
- Write programs to handle text streams, because that is a universal interface.

Elevated privileges

- Vì lý do bảo mật, thực thi một số command hoặc truy cập một số file cần sử dụng đặc quyền (elevated privileges)
- sudo là program cung cấp elevated privileges cho user khi thực thi một command miễn là user nằm trong nhóm super user của hệ thống

Ví dụ:

sudo apt-get install tree

sudo cat /dev/loop0

Tính năng của shell: keyboard shortcut

CTRL+ D	Gửi kí tự EOF (EOT)	Arrow left/right	Di chuyển con trỏ
CTRL+ L	Xóa màn hình	Arrow up/down	Duyệt các command đã thực thi
CTRL+ C	Gửi signal SIGINT	Backspace Delete	Xóa 1 kí tự
CTRL+ Z	Gửi signal SIGTSTP	tab	Hoàn thành command, tên file, dir
CTRL+ \	Gửi signal SIGQUIT	tab tab	Gợi ý command, tên file, dir

Tính năng của shell: Globbing/Wildcards

- Sử dụng một số kí tự đặc biệt để chọn một nhóm các file có tên thỏa mãn "pattern matching"

*	Any characters
?	Any single character
[*characters*]	Any character that is a member of the set *characters*
[!*characters*]	Any character that is not a member of the set *characters*
[[:*class*:]]	Any character that is a member of the specified *class*

Tính năng của shell: Globbing/Wildcards

- Các class được hỗ trợ:

[:alnum:]	Any alphanumeric character
[:alpha:]	Any alphabetic character
[:digit:]	Any numeral
[:lower:]	Any lowercase letter
[:upper:]	Any uppercase letter

Tính năng của shell: Globbing/Wildcards

- Ví dụ

*	All files
g*	Any file beginning with *g*
b*.txt	Any file beginning with *b* followed by any characters and ending with *.txt*
Data???	Any file beginning with *Data* followed by exactly three characters
[abc]*	Any file beginning with either *a*, *b*, or *c*

Tính năng của shell: Globbing/Wildcards

- Ví dụ

`BACKUP.[0-9][0-9][0-9]`	Any file beginning with *BACKUP.* followed by exactly three numerals
`[[:upper:]]*`	Any file beginning with an uppercase letter
`[![:digit:]]*`	Any file not beginning with a numeral
`*[[:lower:]123]`	Any file ending with a lowercase letter or the numerals *1*, *2*, or *3*

Tính năng của shell: I/O redirection

- Theo mặc định input và output của một command được liên kết với Terminal: process sẽ nhận dữ liệu do người dùng gõ vào Terminal, và process sẽ in ra Terminal kết quả hoặc lỗi khi chạy
- I/O redirection là tính năng của shell, nó cho phép chuyển hướng input và output của một command tới các file bằng cách gán các file này vào file descriptor 0, 1, 2 của process

Ví dụ:

```
echo abc > in.txt

cat < in.txt 1> out.txt 2>> err.txt
```

Tính năng của shell: Pipeline

- Shell cho phép ngõ ra của một command nối với ngõ vào của một command khác: *command1* | *command2*

Ví dụ:

ls -l /usr/bin | less

ls /bin /usr/bin | sort | less

ls /bin /usr/bin | sort | uniq | less

ls /bin /usr/bin | sort | uniq -d| less

ls /bin /usr/bin | sort | uniq | grep zip

Tính năng của shell: Jobs

- Command được thực hiện bởi shell được gọi là một job của shell
- Job có thể chạy ở background hoặc foreground
- Để thực hiện job ở background, thêm kí tự & ở cuối command. Ví dụ: *sleep 10 &*
- Command *jobs* liệt kê tất cả các job trong phiên làm việc, *fg* và *bg* chạy các jobs ở foreground hoặc background, và *kill* gửi signal tới job

Tính năng của shell: Script

- Shell hỗ trợ ngôn ngữ lập trình riêng gọi là shell scripting language. File source code của shell scripting language gọi là file script
- Shell có khả năng đọc một file script và thực thi nó

```
#!/bin/bash
# This is our first script.
echo 'Hello World!'
```

Tính năng của shell: Script

```
#!/bin/bash

result="result.txt"
touch $result
for file in ./*; do
    if [ -f $file ]; then
        echo $file >> $result
    fi
done;
```

Exercise

Bài tập

1. Tìm hiểu các command: *help, info, apropos, tail, head, cp, mv, rm, alias, whereis, whoami, apt-get, vi, ln, du, pstree, top, xargs, tee, uniq*
2. Đếm xem có bao nhiêu process đang chạy
3. Tạo một file chứa pid của tất cả các process đang chạy
4. Đếm tổng số program có trong /bin và /usr/bin
5. Tạo một file chứa tên các device có chứa chữ: “sda” hoặc “loop”