# HỆ ĐIỀU HÀNH LINUX
# (Quản lý tiến trình)

Phạm Nguyên Khang, Đỗ Thanh Nghị

Email: pnkhang,dtnghi@cit.ctu.edu.vn

# Nội dung



Tiến trình

Thực thi

Tạo tiến trình

Các hàm cơ bản khác

Thực thi chương trình ở chế độ nền

Liệt kê tiến trình

Các hàm quản lý tiến trình khác

# Tiến trình



Tiến trình = chương trình đang thực thi

Ảnh bộ nhớ (memory image) bao gồm:

- Text: mã thực thi của chương trình
- Data: lưu trữ dữ liệu
- Stack: ngăn xếp của người dùng
- Định danh: phân biệt tiến trình với chương trình khác
  - Cấu trúc u (user: người dùng)
  - Ngăn xếp (stack) của hệ thống
  - Vùng định danh chỉ truy xuất được trong chế độ hệ thống (system mode)

# Thực thi



Điều khiển bằng tiến trình 0 (sched hoặc swapper)

- Tiến trình 0 (không) trong không gian hệ thống
- Sử dụng CPU theo mức độ ưu tiên của tiến trình
- Tiến trình thực thi trong chế độ người dùng (user mode) và chuyển sang chế độ hệ thống bằng các hàm đặc biệt của nhân

Chỉ có duy nhất một tiến trình chạy trong không gian hệ thống: trình quản lý bộ nhớ ảo (pagedaemon)

# Tạo tiến trình



## fork:

Tạo ra một tiến trình con bằng cách sao chép vùng nhớ của một tiến trình cha

Kết quả trả về:

| | |
|---|---|
| -1 | thất bại |
| 0 | tiến trình con |
| N | tiến trình cha với N là PID (định danh) của tiến trình cha |

Ví dụ: (viết bằng ngôn ngữ C)

```
pid = fork();
if (pid == -1) {
        /*lỗi*/
    } else if (pid == 0) {
        /*mã lệnh chương trình con*/
    } else {
        /*mã lệnh chương trình cha */
    }
```

# Các hàm cơ bản khác



**exec**:

Thay thế ảnh bộ nhớ bằng nội dung của một file thực thi khác

**wait:**

Chờ tất cả các tiến trình con kết thúc

**exit:**

Kết thúc một tiến trình

**kill**:

Gởi tín hiệu đến một tiến trình (thường dùng để đồng bộ các tiến trình hoặc buộc một tiến trình kết thúc)

**signal:**

Chọn cách xử lý phải thực hiên khi nhận được một tín hiệu

**pipe:**

Tạo một ống dẫn để giao tiếp giữa hai tiến trình

# Chế độ nền



Mặc định các tiến trình thực thi tuần tự (foreground), tiến trình này thực hiện xong rồi mới đến tiến trình khác

Ví dụ: người dùng gõ lệnh date (hiển thị ngày hệ thống)

- Shell tạo ra một tiến trình con (sử dụng hàm **fork**)
- Chờ tiến trình con này kết thúc (dùng hàm **wait**)
- Trong chương trình con gọi thực thi lệnh /bin/date (bằng lệnh **exec**)

Để thực thi một tiến trình ở chế độ nền (background) ta thêm dấu & vào cuối lệnh

Ví dụ: người dùng gõ lệnh date&

- Shell tạo ra một tiến trình con (fork)
- Không chờ tiến trình con kết thúc
- Tiến trình con gọi thực thi lệnh /bin/date (bằng lệnh **exec**)
- Tiến trình cha và tiến trình con chạy song song

Có thể kế quả hiển thị bị lẫn lộn (vì sử dụng chung một màn hình)

# Tạm dừng tiến trình



Các trạng thái của tiến trình

active (đang hoạt động), waiting (chờ), …

Trạng thái tạm dừng = tiến trình bị ngắt có khả năng chạy lại bằng cách sử dụng lệnh

fg (chạy lại chương trình ở chế độ foreground)

bg  (chạy lại chương trình ở chế độ background)

Tạm ngưng chương trình bằng cách gõ Ctrl + Z

Mục đích:

Thực thi một chương trình khác mà không cần phải kết thúc tiến trình đang chạy

# Liệt kê tiến trình



**ps:**

Liệt kê các tiến trình đang thực thi

Hiển thị tất cả các thuộc tính của tiến trình

Mặc định chỉ liệt kê các tiến trình của người dùng

Cú pháp: ps [OPTIONS]

Ví dụ:

**ps**

**ps l**

**ps aux** (liệt kê tất cả các tiến trình)

Các OPTIONS

-a: hiển thị các tiến trình của user được liên kết tới tty

-e (-A): hiển thị tất cả các tiến trình

-f: hiển thị PID của tiến trình cha và thời điểm bắt đầu

-l: tương tự như –f

a: hiển thị các tiến trình của các users liên kết tới tty

x: hiển thị các tiến trình ngoại trừ các tiến trình là controlling tty (e.g /sbin/mingetty tty*)

u: dạng hiển thị hướng đến người dùng

Ví dụ: ps –ux

# Liệt kê tiến trình



## Xem các tiến trình

Lệnh pstree với

-p: hiển thị PID

-h: highlight tiến trình hiện hành và những tiến trình con cháu của tiến trình hiện hành

```
[user1@m-nghi ~]$ pstree -h
init─┬─acpid
     ├─anacron
     ├─atd
     ├─auditd
     ├─2*[automount]
     ├─bonobo-activati
     ├─cannaserver
     ├─clock-applet
     ├─crond
     ├─cups-config-dae
     ├─cupsd───lpd
     ├─2*[dbus-daemon]
     ├─dbus-launch
     ├─eggcups
     ├─events/0
     ├─gam_server
     ├─gconfd-2
     ├─gnome-keyring-d
     ├─gnome-panel
     ├─gnome-settings-
     ├─gnome-terminal─┬─bash───su───bash───pstree
     │                └─gnome-pty-helpe
     ├─gnome-vfs-daemo
     ├─gnome-volume-ma
     ├─gpm
     ├─hald─┬─hald-addon-acpi
     │      └─hald-addon-stor
     ├─hcid
```

# Các lệnh khác



các thông số của hệ thống

ng thực thi

```
top - 09:53:42 up 50 min,  2 users,  load average: 0.35, 0.17, 0.11
Tasks:  92 total,   1 running,  91 sleeping,   0 stopped,   0 zombie
Cpu(s):  1.7% us,  3.7% sy,  0.0% ni, 94.6% id,  0.0% wa,  0.0% hi,  0.0% si
Mem:    255620k total,   244132k used,    11488k free,    24752k buffers
Swap:   522104k total,        0k used,   522104k free,   117164k cached
```

| PID | USER | PR | NI | VIRT | RES | SHR | S | %CPU | %MEM | TIME+ | COMMAND |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 2462 | root | 15 | 0 | 35864 | 13m | 5132 | S | 3.0 | 5.6 | 1:22.92 | X |
| 2651 | root | 25 | 10 | 34960 | 17m | 10m | S | 1.3 | 7.0 | 0:50.05 | rhn-applet-gui |
| 1905 | root | 16 | 0 | 2740 | 576 | 488 | S | 0.3 | 0.2 | 0:02.06 | nifd |
| 2689 | root | 15 | 0 | 37788 | 12m | 8320 | S | 0.3 | 4.9 | 0:05.76 | gnome-terminal |
| 1 | root | 16 | 0 | 1748 | 572 | 492 | S | 0.0 | 0.2 | 0:01.15 | init |
| 2 | root | 34 | 19 | 0 | 0 | 0 | S | 0.0 | 0.0 | 0:00.00 | ksoftirqd/0 |
| 3 | root | RT | 0 | 0 | 0 | 0 | S | 0.0 | 0.0 | 0:00.13 | watchdog/0 |
| 4 | root | 10 | -5 | 0 | 0 | 0 | S | 0.0 | 0.0 | 0:00.13 | events/0 |
| 5 | root | 10 | -5 | 0 | 0 | 0 | S | 0.0 | 0.0 | 0:00.05 | khelper |
| 6 | root | 10 | -5 | 0 | 0 | 0 | S | 0.0 | 0.0 | 0:00.00 | kthread |
| 8 | root | 20 | -5 | 0 | 0 | 0 | S | 0.0 | 0.0 | 0:00.00 | kacpid |
| 61 | root | 10 | -5 | 0 | 0 | 0 | S | 0.0 | 0.0 | 0:00.09 | kblockd/0 |
| 64 | root | 15 | 0 | 0 | 0 | 0 | S | 0.0 | 0.0 | 0:00.00 | khubd |
| 110 | root | 20 | 0 | 0 | 0 | 0 | S | 0.0 | 0.0 | 0:00.00 | pdflush |
| 111 | root | 15 | 0 | 0 | 0 | 0 | S | 0.0 | 0.0 | 0:00.20 | pdflush |
| 113 | root | 11 | -5 | 0 | 0 | 0 | S | 0.0 | 0.0 | 0:00.00 | aio/0 |
| 112 | root | 15 | 0 | 0 | 0 | 0 | S | 0.0 | 0.0 | 0:00.05 | kswapd0 |
| 200 | root | 15 | 0 | 0 | 0 | 0 | S | 0.0 | 0.0 | 0:00.00 | kseriod |
| 360 | root | 23 | 0 | 0 | 0 | 0 | S | 0.0 | 0.0 | 0:00.00 | scsi_eh_0 |
| 373 | root | 15 | 0 | 0 | 0 | 0 | S | 0.0 | 0.0 | 0:00.60 | kjournald |
| 882 | root | 12 | -4 | 1636 | 548 | 468 | S | 0.0 | 0.2 | 0:00.29 | udevd |
| 935 | root | 20 | 0 | 0 | 0 | 0 | S | 0.0 | 0.0 | 0:00.00 | kgameportd |

# Các lệnh khác



**kill**:

Gởi một tín hiệu đến một tiến trình

Cú pháp:

kill [-signal | -s signal] pid

Các signal

| | | |
|---|---|---|
| 0 | 0 | |
| HUP | 1 | (hangup) |
| INT | 2 | (tương đương CTRL + C) |
| KILL | 9 | (buộc kết thúc) |
| TERM | 15 | (mặc định, kết thúc êm ái) |
| STOP | 19 | (tạm dừng, tương đương CTRL + Z) |

Ví dụ:

**kill -1 1234** (gởi tín hiệu HUP đến tiến trình 1234)

**kill -s 9 3456** (gởi tín hiệu KILL đến tiến trình 3456, buộc tiến trình này kết thúc)

**kill -l** liệt kê tất cả các tín hiệu

id = -1 có nghĩa tất cả các tiến trình trừ tiến trình **kill** và tiến trình **init**

# Các lệnh khác



Tránh HUP: nohup

Tiến trình nhận tín hiệu HUP khi người dùng logout khỏi session

Sử dụng nohup để bỏ qua tín hiệu HUP

Ví dụ: **nohup find / –name log.txt&**

# Các lệnh khác



**jobs**:

Liệt kê tất cả các job gồm

Tiến trình thực thi chế độ nền

Tiến trình tạm ngưng

Tiến trình bị ngẽn (chờ vào/ra)

Chú ý: mỗi job có một số hiệu của job (khác với định danh của tiến trình)

Các lệnh bg, fg, kill cũng có thể làm việc được với số hiệu job thay vì pid. Để sử dụng số hiệu job ta dùng **%<số hiệu job>**

**nice**:

Chạy chương trình với một độ ưu tiên nào đó

Cú pháp: **nice –n <độ ưu tiên> <chương trình>**

Độ ưu tiên từ -20 (ưu tiên cao nhất) đến 19 (ưu tiên thấp nhất). Độ ưu tiên mặc định = 0.

Ví dụ: **nice -n 12 xcalc**

# Các lệnh khác



**renice**:

Thay đổi độ ưu tiên của một tiến trình

Cú pháp: **renice -n <độ ưu tiên> -p <pid>**

Hoặc: **renice <độ ưu tiên> <pid>**

Ví dụ: **renice 1 4567**

**Chú ý:** người dùng bình thường không thể thay đổi độ ưu tiên nhỏ hơn 0 (không).