TRƯỜNG ĐH KHOA HỌC TỰ NHIÊN HÀ NỘI
KHOA TOÁN – CƠ – TIN HỌC

BÀI THỰC HÀNH LINUX
Bài 04

——————— oOo ———————

Phần 1: Thực hành

Quản lý tiến trình

Hiển thị thông tin tiến trình

Để liệt kê các tiến trình đang thực thi, dùng lệnh **ps** (*process status*)

Cú pháp: *ps [option]*

Trong đó, các option là:

- -a: hiển thị các tiến trình của user được liên kết tới tty
- -e (-A): hiển thị thông tin về mỗi tiến trình
- -f: hiển thị PID của tiến trình cha và thời điểm bắt đầu
- -l: tương tự như -f
- x: hiển thị các tiến trình ngoại trừ các tiến trình là controlling tty (Ví dụ: /sbin/mingetty tty*)
- u: dạng hiển thị hướng đến người dùng

Lệnh *ps* có thể kết hợp với lệnh *grep* để tìm kiếm một tiến trình đang chạy. Ví dụ:

Ví dụ: *student@linux* $\sim$ *$ ps aux* -> Liệt kê tất cả các tiến trình.

Khi đó, các thông số của tiến trình sẽ được liệt kê gồm có: Chủ nhân của tiến trình (owner), mã số nhận diện tiến trình (PID), thời gian hiện sử dụng CPU (%CPU), mức chiếm dụng bộ nhớ của tiến trình (%MEM), trạng thái tiến trình (STAT) và các thông tin khác.

Một số trạng thái của tiến trình thường gặp: R-đang thi hành, S-đang bị đóng, Z-ngừng thi hành, W-không đủ bộ nhớ, ...

Ngoài ra, có thể dùng lệnh **top** để xem các thông số liên quan đến các tiến trình, đặc biệt là các thông tin sử dụng tài nguyên của các tiến trình đó.

Cú pháp: *top [option]*

Trong đó, các *option* là:

- -u: Xem những tiến trình đang hoạt động dưới một tài khoản nào đó.
- -p <PID>: Xem một tiến trình thông qua PID của tiến trình đó.
- -c: Hiển thị đầy đủ dòng lệnh thay vì hiển thị tên lệnh tạo tiến trình
- -d <time>: Thời gian tải lại các hiển thị liên quan đến lệnh top. Giá trị được tính theo giây. Mặc định là 5s.

Có thể dùng thêm lệnh **pgrep** để xem PID của một tiến trình trên hệ thống. Ví dụ, khi mở một tệp tin soạn thảo bằng emacs, bạn sẽ tìm kiếm PID của tiến trình này như sau:

Ví dụ: *student@linux* ~ *$pgrep emacs*

? Liệt kê các tiến trình dưới quản lí của tài khoản *student*, *root*?

? Lưu kết quả của lệnh top vào tệp tin topoutput.txt?

? Mở một tệp tin bằng emacs, kiểm tra xem tiến trình này có hoạt động hay không?

Liệt kê các tiến trình

Để liệt kê các tiến trình theo dạng cây, dùng lệnh **pstree**

Cú pháp: *pstree [option]*

Trong đó, các *option* là:

- -p: hiển thị PID
- -h: tô đậm những tiến trình hiện hành và những tiến trình con cháu của tiến trình hiện hành
- -a : chỉ ra tham số dòng lệnh. Nếu dòng lệnh của một quá trình được tráo đổi ra bên ngoài, nó được đưa vào trong dấu ngoặc đơn.
- -c : không thể thu gọn các cây con đồng nhất. Mặc định, các cây con sẽ được thu gọn khi có thể
- -H : giống như tùy chọn -h, nhưng quá trình con của quá trình hiện thời không có màu sáng trắng
- -l : hiển thị dòng dài
- -n : sắp xếp các quá trình cùng một tổ tiên theo chỉ số quá trình thay cho sắp xếp theo tên

Các chế độ chạy của tiến trình

Để quản lí các chế độ chạy của một tiến trình, có thể dùng các lệnh **&** hoặc **Ctrl C, Ctrl Z, fg, bg**

- &: Cho tiến trình hoạt động ở trạng thái nền (*background*)

  Ví dụ: *student@linux* ~ *$ ls -l -R/ > /home/student/list.txt* & -> ứng dụng ls sẽ chạy nền bên dưới

  Hoặc:

  Ví dụ: *student@linux* ~ *$emacs*& -> ứng dụng emacs sẽ chạy nền, khi đó người sử dụng có thể dùng terminal để thực hiện các lệnh khác.
- *Ctrl C*: Kết thúc tiến trình đang thực thi, sau khi ấn Ctrl C, có thể dùng lệnh jobs để hiển thị trạng thái của tiến trình đang chạy
- *Ctrl Z*: Tạm ngừng tiến trình đang thực thi sau đó, có thể dùng các lệnh fg, bg để tiếp tục:
  - *bg*: tiếp tục tiến trình vừa tạm ngừng ở trạng thái nền (*background*)

– *fg*: tiếp tục tiến trình vừa tạm ngừng ở trạng thái hiện (*foreground*)

? Thực thi lệnh liệt kê tất cả các tệp tin, thư mục có trong */home/student*? Kết quả lưu vào tệp tin list.txt. Sau đó chuyển lệnh trên vào chế độ bg? Tạm ngừng lệnh trên và cho phép thực thi lại?

Dừng một tiến trình

Lệnh **kill** thường được sử dụng để ngừng thi hành một tiến trình.

Cú pháp: *kill [option] <PID>*

Trong đó:

- option là một lựa chọn:
  - -s : xác định tín hiệu được gửi. Tín hiệu có thể là số hoặc tên của tín hiệu. Dưới đây là một số tín hiệu hay dùng:
    - SIGHUP(1): Hangup (gọi lại tiến trình)
    - SIGINT(2): Interrupt (Ngắt từ bàn phím Ctrl
    - SIGKILL(9): Hủy tiến trình ngay lập tức
    - SIGTERM(15): Terminate – Kết thúc tiến trình, nhưng cho phép xóa các tệp tin tạm
  - -l : hiển thị danh sách các tín hiệu mà lệnh *kill* có thể gửi đến các quá trình (các tín hiệu này có trong file /usr/include/Linux/signal.h)
- PID: mã số nhận diện tiến trình muốn dừng

Lệnh *kill* có thể gởi bất kỳ tín hiệu *signal* nào tới một tiến trình, nhưng theo mặc định nó gởi tín hiệu 15, TERM (là tín hiệu kết thúc chương trình). Super-user mới có quyền dừng tất cả các tiến trình, còn người sử dụng chỉ được dừng các tiến trình của mình.

? Dừng một tiến trình bất kì của *student* trong số các tiến trình vừa liệt kê, kiểm tra lại xem tiến trình đó đã dừng hay chưa?

? Dừng tiến trình emacs vừa tạo ra ở trên, kiểm tra lại xem tiến trình này còn tồn tại hay không?

Độ ưu tiên của một tiến trình

Để chạy chương trình với một độ ưu tiên nào đó, dùng lệnh **nice**

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

Trong đó, độ ưu tiên từ -20 (độ ưu tiên cao nhất) đến 19 (ưu tiên thấp nhất), độ ưu tiên mặc định là 0.

Ví dụ: *student@linux ~ $ nice -n 12 abcd*

Để thay đổi độ ưu tiên của một tiến trình dùng lệnh *renice*

Cú pháp: *renice <độ ưu tiên> [option]*

Hoặc: *renice <độ ưu tiên> <pid>*

Trong đó, option là:

- -g : thay đổi quyền ưu tiên theo nhóm người dùng
- -p : thay đổi quyền ưu tiên theo chỉ số của quá trình
- -u : thay đổi quyền ưu tiên theo tên người dùng

Ví dụ: *student@linux ~ $ renice 1 3456*

Ví dụ: *student@linux ~ $renice +1 987 -u daemon root -p 32* -> Lệnh trên sẽ thay đổi mức độ ưu tiên của quá trình có chỉ số là 987 và 32, và tất cả các quá trình do người dùng daemon và root sở hữu.

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

Giới thiệu về Bash Shell Script

Shell là một chương trình dùng để tương tác giữa người dùng và máy tính. Thông dịch các lệnh của người dùng nhập vào hoặc từ các tệp tin.

Shell Script: Là các chương trình shell gồm tập hợp các lệnh đặc biệt, từ đó có thể tạo nên chương trình.

- **Các loại Shell trên Linux**
  Một số Shell trên Linux:
  - BASH (*Bourne-Again Shell*) phát triển bởi Brian Fox và Chet Ramey. Đây là Shell thông dụng nhất trên Linux.
  - CSH (*C Shell*) phát triển bởi Bill Joy tại University of California (dành cho BSD). Sử dụng cấu trúc lệnh giống C, rất thân thiện cho các lập trình viên C trên linux.
  - KSH (*Korn Shell*) phát triển bởi David Korn tại AT & T Bell Labs.
  - TCSH (*TENEX C Shell*) là phiên bản nâng cấp của C Shell.

  Để xem hệ thống đang chạy shell gì, dùng lệnh echo $SHELL
- **Viết Bash Shell Script đơn giản**
  - Soạn thảo chương trình
    * Sử dụng mọi trình soạn thảo dạng text: vi, emacs, ...
    * Nội dung:
      + Gồm các câu lệnh được sử dụng trên dòng lệnh của Linux
      + Dòng đầu tiên chứa câu lệnh *#!/bin/bash* (Chỉ ra loại shell sử dụng)
      + Để chú thích một đoạn mã: dùng # trước đoạn mã cần chú thích
      + Các câu lệnh trên cùng một dòng phải phân tách bằng dấu;
      + Lưu tệp tin với đuôi .sh
  - Thiết lập quyền thực thi cho chương trình shell
    Mặc định các tệp tin tạo ra chưa có quyền thực thi. Cần phải cấp quyền thực thi bằng lệnh *chmod*
    Ví dụ: *chmod u+x fileName*
  - Thực thi chương trình sử dụng một trong các lệnh:
    * bash fileName
    * sh fileName

* ./fileName

– Ví dụ: Sử dụng vi soạn thảo một tệp tin bashTest.sh với nội dung sau:
 # Chuong trinh Bash Script
 # in 1 cau lenh ra man hinh
echo "Hello World!"
 # Tao 1 thu muc ngoai Desktop
mkdir /home/student/Desktop
? Thực thi tệp tin vừa tạo để thấy kết quả.

Phần 2: Bài tập thực hành

Soạn thảo chương trình java trong tệp tin *Prime.java* để kiểm tra một số có là nguyên tố hay không. Viết tệp tin *bash.sh* để chạy chương trình trên với các giá trị được nhập từ tệp *input.txt* và in kết quả ra tệp *output.txt*

Phần 3: Liên lạc

STT	**Họ và tên**	**Email**	**ĐT**
1	Nguyễn Minh Hải	nguyenminhhai06@gmail.com	
2	Hà Mỹ Linh	halinh.hus@gmail.com	
3	Nguyễn Thị Huyên	nthuyen.bmth@gmail.com	