Bài 5: (tiếp tục) Công cụ quản lý dịch vụ của sysv và systemd

1. Sysv:

+ Các file mã dịch vụ trên sysv được lưu trữ ở thư mục /etc/rc.d/init.d.

+ Nếu không sử dụng công cụ quản lý dịch vụ, sẽ phải chạy dịch vụ bằng tay, chẳng hạn muốn start dịch vụ httpd:

/etc/rc.d/init.d/httpd start

+ Ở đây start là một action của dịch vụ, để xem các action của một dịch vụ bất kỳ, chọn action st:

/etc/rc.d/init,d/<tên file mã của dịch vụ> st

+ Để tiện lợi trong việc quản lý các dịch vụ trên sysv, sử dụng tool service, và cách sử dụng như sau:

service <tên dịch vụ> action

+ Như đã biết, nền tảng sysv tổ chức hệ thống theo các runlevel. Các run level chạy các dịch vụ gì sẽ được quyết định bởi các thư mục từ rc0.d-rc6.d trong /etc/rc.d/.

Các thư mục này sẽ tập hợp các link dẫn tới các file mã dịch vụ chứa trong /etc/rc.d/init.d. Các link này sẽ có tên dạng như sau:

K(S)<thứ tự khởi động dịch vụ><tên dịch vụ>

Chẳng hạn: K15httpd

Điều này nghĩa là dịch vụ httpd đang tắt, và nếu được bật lên thì nó là dịch vụ thứ 15 được chạy.

K và S ở đây còn được gọi là cách hành xử của dịch vụ với runlevel cụ thể.

+ Tool chkconfig: Thay đổi cách hành xử của dịch vụ mà không cần sử ở config file.

Ví dụ: Để xem cách hành xử của 1 dịch vụ với tất cả các runlevel:

chkconfig --list <tên dịch vụ>

Thay đổi cách hành xử với một level cụ thể:

chkconfig --level <number of runlevel> <name service> on/off

Thay đổi với mọi runlevel:

chkconfig <name service> on/off

2. Systemd:

Như đã biết, systemd tổ chức hệ thống theo units và targets.

+ Unit: Xác định một dịch vụ hoặc hành động trên hệ thống và nó có 3 thuộc tính: name, type, config file.

Với systemd sử dụng công cụ systemctl để quản lý tất cả units và target.

Xem cách sử dụng systemctl bằng option –help, hoặc sử dụng man systemctl.

Ví dụ: Để hiển thị tất cả units và targets: systemctl list-units

+ Type: Có 8 loại unit trên systemd, đó là:

* + Automount
  + Path
  + Socket
  + Device
  + Service
  + rget
  + Mount
  + Snapshot

+rget: Như liệt kê ở trên thì rget cũng có thể coi là 1 unit. Đây là tập hợp các units cần thiết cho máy chủ làm việc với rget này.

rget(systemd)=Runlevel(sysv)

Để thay đổi rget làm việc:

systemctl isolate <name rget>.rget

Bài 6: Cài đặt và tiến trình

1. Cài đặt:

sẽ tìm hiểu về các công cụ cài đặt của hai dòng Linux phổ biến hiện nay là dòng Redhat và Debian. sẽ tìm hiểu cả cách cài đặt cơ bản và nâng cao, hiểu rõ các quá trình phát triển của các công cụ cài đặt trên các dòng Linux.

Về bản chất, các công cụ cài đặt là các công cụ giải nén và copy, nên chúng giống nhau. Ở bài này tìm hiểu các công cụ rpm và yum của Redhat.

**Cách 1:** Công cụ giải nén **rpm**

+ Quy tắc đặt tên của gói cài rpm: package-version-build.architectutre.rpm

+ Cú pháp cài đặt: rpm –ivh <package>

Trong đó: -v: hiển thị thêm deiled output

-h: hiển thị thêm % cài đặt

+ Truy vấn các gói cài trên hệ thống: rpm –qa | grep <name\_app>

Truy vấn này sẽ hiển thị cho tên gói cài đã được cài trên hệ thống ko có đuôi .rpm.

+ Gỡ gói cài: rpm –e <name\_package>

+ Upgrade: rpm –U <new\_package>

+ Truy vấn thông tin những file trên hệ thống được sinh ra từ gói cài:

rpm –ql <name\_package>

+ Truy vấn thông tin gói cài từ tên file chạy ứng dụng:

rpm –qf <name\_binfile>

+ Truy vấn thông tin về gói cài:

rpm –qip <package>

+ Truy vấn thông tin gói cài đã cài đặt và được quản lý trên hệ thống:

rpm –qi <name\_package>

Trong quá trình cài đặt với công cụ **rpm** này, chúng sẽ thường gặp phải một lỗi đó là lỗi phụ thuộc (failed dependencies).

Lỗi này tức là khi cài đặt 1 appA thì app này sẽ cần phải có thư viện hoặc file chạy con của một app khác thì mới có thể chạy được. Và như thế hệ thống sẽ bào là không cài đặt thành công và liệt kê những gói cài chứa phần phụ thuộc của appA mà cần cài. Giải pháp là sẽ cài đặt lần lượt các gói cài chứa thành phần phụ thuộc trước, sau đó sẽ cài appA cần cài cuối cùng.

Ngoài ra, còn có thể sử dụng các option đặc biệt sau để ép quá trình cài đặt thành công:

+ --nodeps: Bỏ qua sự phụ thuộc gói cài, cài thẳng.

+ --replaces: Ghi đè luôn lên file system

+ --force: Bỏ qua mọi cản trở (option mạnh nhất)

!Lưu ý: Các option này phải đặt ở cuối câu lệnh thì mới dùng được vì chúng là những options đặc biệt.

Và sự phụ thuộc xảy ra không chỉ ở quá trình cài đặt mà còn ở cả quá trình gỡ cài đặt.

**Cách 2:** Công cụ **yum**

Với cách cài 1 thì có thể thấy việc cài đặt sẽ mất thời gian và gây khó khăn cho người sử dụng. Với công cụ **yum** chỉ cần duy nhất một thao tác là có thể cài đặt thành công ứng dụng mà muốn cài:

yum insll <name\_service/app>

Tại sao công cụ yum lại có sự khác biệt lớn so với rpm như vậy, sẽ làm rõ ở phân tiếp theo, với khái niệm đầu tiên là kho repo.

Mô hình hoạt động của yum là khi cài đặt ứng dụng hoặc dịch vụ, thì hệ thống sẽ đọc trong /etc/yum.repo.d/ các thông tin về các yum server mà máy chủ sử dụng để tải và cài đặt các gói cài.

Yum server này sẽ chứa kho các phần mềm, cung cấp đầy đủ các gói cài, thư viện cần thiết cho ứng dụng muốn cài, kho này gọi là repo.

có thể tạo một kho repo trên chính hệ thông của chúng để hiểu hơn về yum.

+ Tạo một thư mục trống: mkdir /softwares

+ Copy tất cả các gói cài mà ứng dụng nào đó cần, hoặc có thể copy toàn bộ nội dung trong thư mục Packages của ổ CD về bởi vì ổ CD cũng là một kho chứa phần mềm cưcj kỳ đầy đủ.

cp /media/CentOS\_6.10\_Final/\* /softwares

+ Để /softwares trở thành một repo thật sự cần tool **createrepo** (tool này có thể cài trực tiếp từ gói createrepo trong Packages trong ổ CD).

createrepo /softwares/

+ Sau khi đã có kho repo, khai báo kho vào thư mục /etc/yum.repo.d/ ở trong 1 file là a.repo, sẽ tạo và edit nội dung file như sau:

vi /etc/yum.repo.d/a.repo

Nội dung:

[soft]

baseurl=file:///softwares

enabled=1

name=test

gpgcheck=0

+ Sau đo lưu và thoát file rồi tiền hành cài đặt ứng dụng cần cài:

yum insll <app>

+ Sau này, mỗi khi cập nhật thêm gói cài thì cần cập nhật kho:

createrepo –update /softwares/

Sau đó cần làm sạch cache trên RAM để lấy thông tin về kho mới:

yum clean all

Lúc này kho đã được làm mới và có thể cài đặt như bình thường.

Tương tự với dòng Debian thì có dpkg và apt-get tương ứng với rpm và yum, cách cài đặt và khái niệm về repoda cũng giống hệt nên tự tìm hiểu.

2. Quản lý tiến trình:

Một tiến trình sẽ có 4 trạng thái sau:

+ Run (R): Trạng thái còn CPU để chạy và có dữ liệu để tính toán.

+ Sleep (S): Trạng thái còn CPU để chạy nhưng không có dữ liệu để tính toán.

+ Stop (T): Trạng thái đủ điều kiện cả CPU lẫn dữ liệu để chạy nhưng không được chạy

+ Zoombie (Z): Trạng thái tiến trình chạy mất kiểm soát, thường xảy ra khi hệ thống có lỗi.

Một số công cụ giúp quản lý các tiến trình:

+ Lệnh ps giúp chụp lại hiện trạng các tiến trình đang chạy ở thời điểm hiện tại. Lệnh này sẽ không giúp xem đầy đủ thông tin tất cả các tiến trình trên hệ thống, nó cần thêm option –cef

ps –cef

Hoặc option –aux sẽ giúp hiển thị thông tin chiếm dụng tài nguyên của các tiến trình.

+ Lệnh top sẽ chạy như là một tiến trình, giúp xem trực tiếp thông tin, sự thay đổi của các tiến trình trên hệ thống.

Các thông số tiến trình cần nắm rõ khi quản lý:

+ PID: ID của tiến trình

+ PPID: ID của tiến trình cha sinh ra tiến trình đó

+ PRI: Độ ưu tiên của tiến trình

+ CLS: Cờ, chỉ đinh trạng thái tiến trình

+ STIME: srted time

+ TIME: Thời gian để khởi chạy lên

Giá trị PRI của một tiến trình:

+ Ban đầu giá trị PRI được đặt như sau: PRI = default + NICE

Trong đó: default là giá trị mặc định, thường là bằng 20

NICE sẽ có giá trị tùy thiết lập, nếu để mặc định sẽ có giá trị 0.

+ Giá trị của NICE nằm trong khoảng [-20;19], chỉ có người dùng root mới có quyền thiết lập giá trị âm cho chỉ số NICE, các user bình thường chỉ có thể thiết lập giá trị dương.

Cú pháp: nice <nice\_value> <process/cmd> (với tiến trình chưa chạy)

renice <nice\_value> <PID> (với tiến trình đã chạy, cài lại chỉ số NICE).

Tác vụ (Jobs)

Tác vụ (jobs) là một tiến trình hoặc một nhóm các tiến trình được quản lý bởi terminal. Một tác vụ có thể chạy ở background hoặc foreground.

Background: Khi một tiến trình chạy ở chế độ background, nó không chiếm dụng terminal và cho phép tiếp tục sử dụng terminal cho các lệnh khác mà không cần chờ đợi tiến trình kết thúc. Đặt một lệnh vào chế độ background bằng cách thêm dấu "&" ở cuối lệnh.

Ví dụ: firefox &

Foreground: Khi một tiến trình chạy trong chế độ foreground, nó chiếm dụng terminal và sẽ nhận input từ người dùng trực tiếp. Trong chế độ foreground, terminal thường bị ngăn chặn (blocked) và sẽ đợi đến khi tiến trình kết thúc trước khi trả lại quyền kiểm soát cho người dùng.

Ví dụ: 1 ứng dụng đang chạy fg, muốn chuyển sang bg

Trc tiên ctrl + z tiến trình

Sau đó xem job IB của tiến trình: jobs

Chọn job ID của tiến trình đang cần chạy bg

Chuyển tiến trình sang chạy bg: bg <job\_ID>

Tương tự cho từ bg sang fg: fg <job\_ID>