Buổi 1: Linux và phần mềm mã nguồn mở

1. Phần mềm mã nguồn mở là gì?

Là những phần mềm cho phép truy cập và sử dụng miễn phí với mã nguồn mở để cộng đồng có thể thay đổi và phát triển tùy ý. Tuy vậy mỗi phần mềm sẽ có những liscense riêng để đảm bảo về mặt pháp lý. tìm hiểu thêm về Freeware; shareware; Charityware (careware), copyleft và copyright, phần mềm mã nguồn mở Nukeviet.

Năm 1970, AT&T và Bells Lap cho ra đời Unix, cung cấp một hệ điều hành mã nguồn mở đầu tiên phục vụ cho việc quản lý máy chủ để chỉ đạo các công tác sản xuất , hoặc công tác quản lý dữ liệu khách hàng của các doanh nghiệp. Giải quyết được vấn đề về việc ko có phần mềm tương thích với sự phát triển mạnh mẽ của phần cứng thời đó.

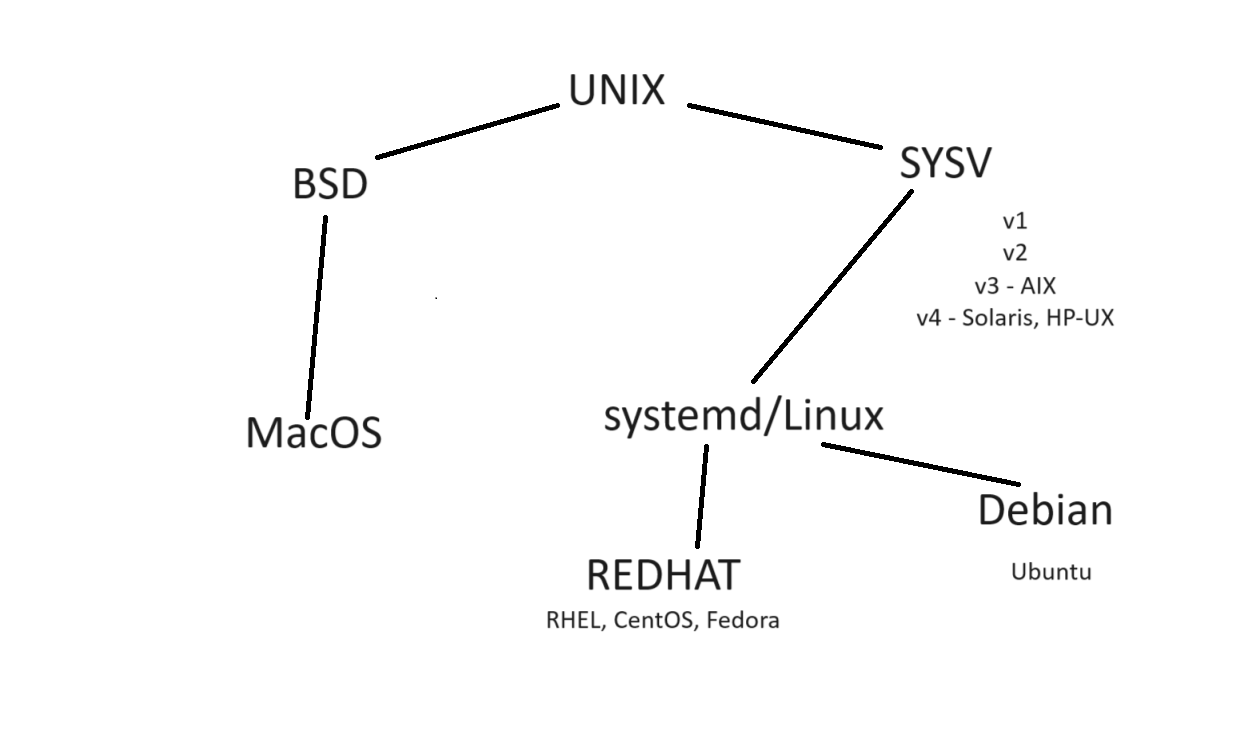
Tiếp nối sự thành công của Unix, hai hướng đi hướng đến hai phân khúc tiêu dùng là hai nền tảng tổ chức hệ thống dựa trên Unix ra đời, đó là SYSV cho doanh nghiệp và BSD cho người dùng bình thường.

Đối với SYSV, tổ chức hệ thống dựa trên file init và các runlevel scripts để quản lý việc khởi động và dừng dịch vụ. SYSV phát triển tới 4 phiên bản, trong đó phiên bản thứ ba có tên gọi khác là AIX – phiên bản được IBM phát triển để dành riêng cho dòng máy chủ POWER của họ. Phiên bản thứ 4 được Sun Microsystem và HP phát triển riêng biệt, còn có tên gọi là Solaris và HP-UX.

Năm 2010, nền tảng tổ chức hệ thống systemd ra đời, ưu việt và linh hoạt hơn so với nền tảng sysv. Nếu như sysv quản lý dịch vụ theo các config file trong các dịch vụ riêng biệt, thì systemd lại quy hết config files về 1 mối, đó là unit files trong /lib/systemd/system. Ko còn khái niệm runlevel, giờ đây hệ thống sẽ sử dụng rget để định rõ mục tiêu của hệ thống.

Với nền tảng systemd, có hai tổ chức chính trong việc phân phôi và phát triển đó là Redhat và Debian. Bằng việc cung cấp miễn phí hệ điều hành quản lý máy chủ cho các doanh nghiệp và bán services, Redhat đã thành công với các bản phân phối nổi tiếng là RHEL, CentOS hay Fedora. Còn Debian là một tổ chức được cộng đồng lập ra với tiêu chí phát triển hệ điều hành mã nguồn mở vì cộng đồng, cũng rất nổi tiếng với các phiên bản Ubuntu hiện nay.

Về BSD, thì nó là tiền thân của một hệ điều hành rất nổi tiếng sau này, đó là MacOS.



2. Linux là gì?

Linux là một hệ điều hành mã nguồn mở. Tiền thân là UNIX.

Năm 1980, Richard Sllman đã thành lập ra tổ chức Free Software Foundation (FSF), phát triển ra một bộ công cụ quản lý hệ thống GNU dựa trên bộ của Unix, tuy nhiên họ không có nhân để chạy. Năm 1985, Sllman viết GNU General Public License (GPL), liscense mã nguồn mở đầu tiên yêu cầu bất kỳ phần mềm dựa trên mã nguồn GPL phải cũng là mã nguồn mở. (Tìm hiểu thêm về liscense của BSD).

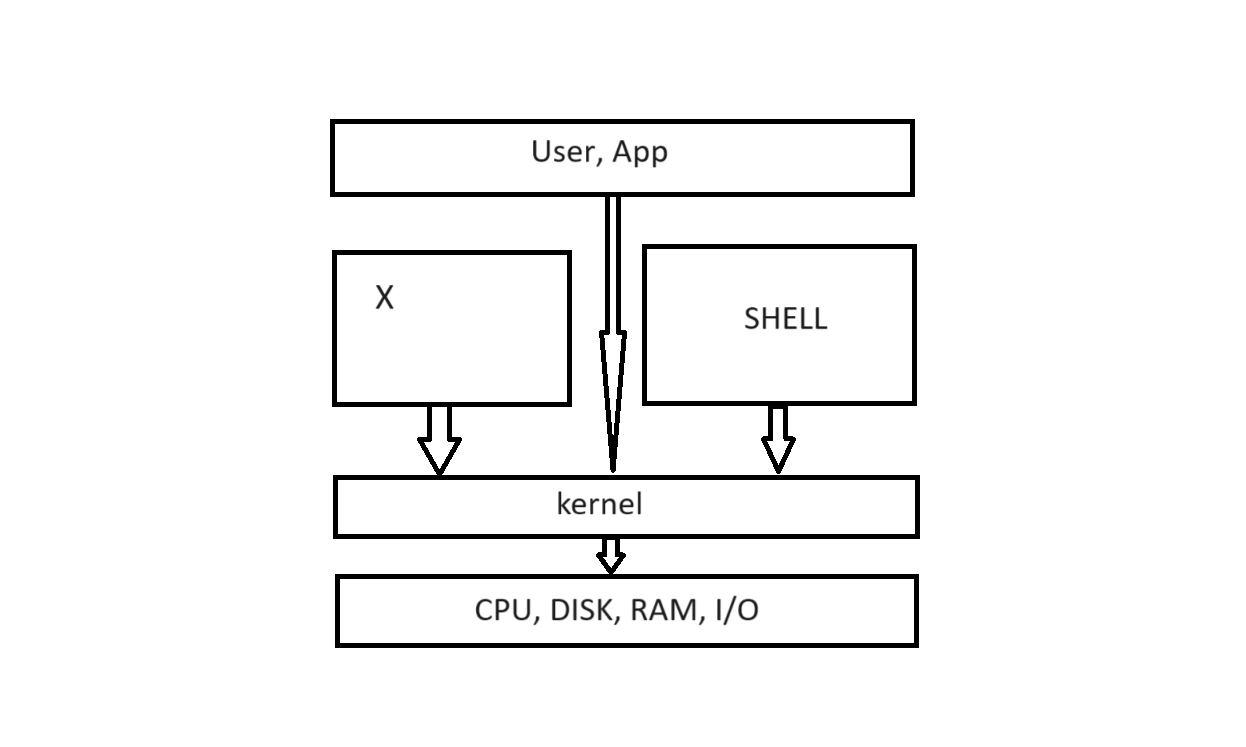
Năm 1991, Linus Torvalds đã công khai lên mạng internet phiên bản lõi Linux đầu tiên (The First Linux Kernel) và kêu gọi cộng đồng cùng phát triển nó. Ban đầu kernel được viết bằng hợp ngữ, sau đó đã được Linus viết lại bằng C.

Tuy nhiên, bản thân kernel không có khả năng thực hiện các tác vụ, mà nó cần có kẻ ra lệnh.

Lúc này, vừa vặn GNU đang cần một nhân để chạy. Như thế, Linux đã ra đời.

Sau đây, có sơ đồ về kiến trúc hệ điều hành Linux, và kiến trúc này cũng là của mọi hệ điều hành hiện giờ.

Theo sơ đồ, có:



* Kernel sẽ quản lý cả phần mềm và phần cứng, nhận lệnh từ SHELL (các công cụ hệ thống)
* SHELL trong Linux là GNU utilities, thuộc bộ công cụ hệ thống GNU, giúp người dùng quản lý, điều hành máy chủ.

Buổi 2: Khái niệm ảo hóa. Cách Linux phân vùng

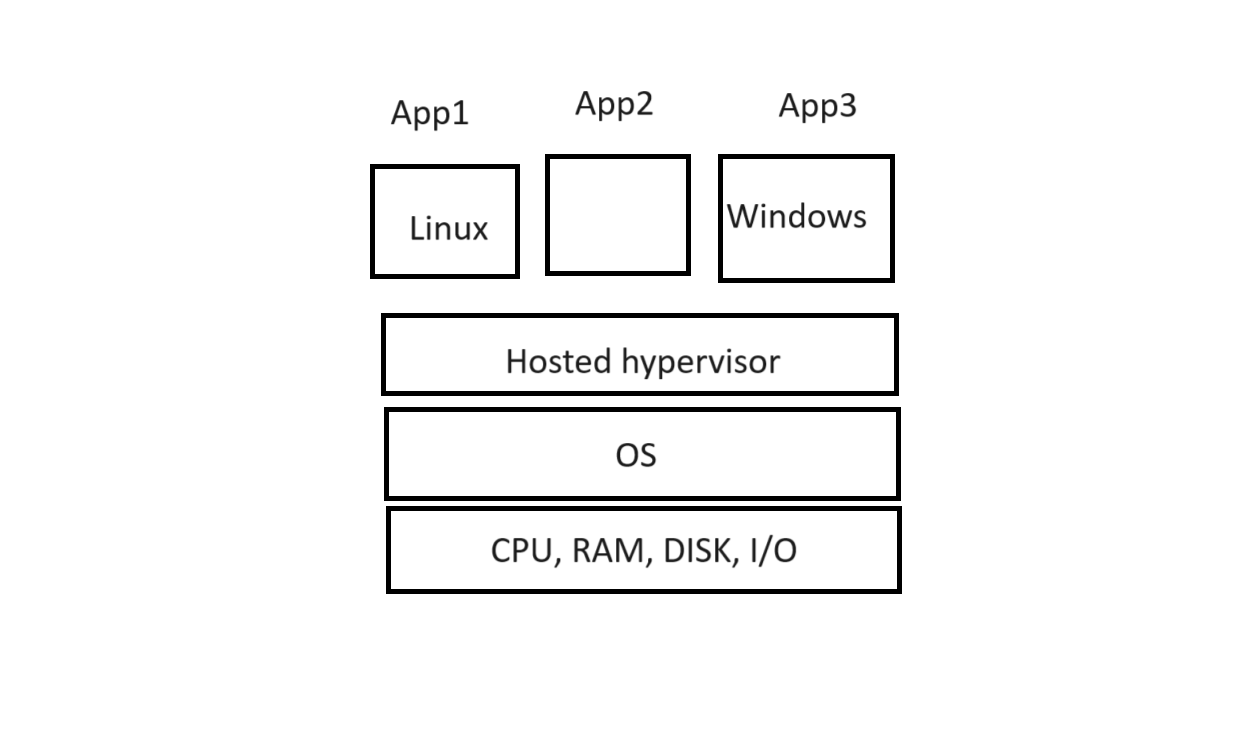
1. Khái niệm ảo hóa

Trước hết cần nắm được, một ứng dụng có thể chạy được sẽ cần tối thiểu 3 files cơ bản, đó là:

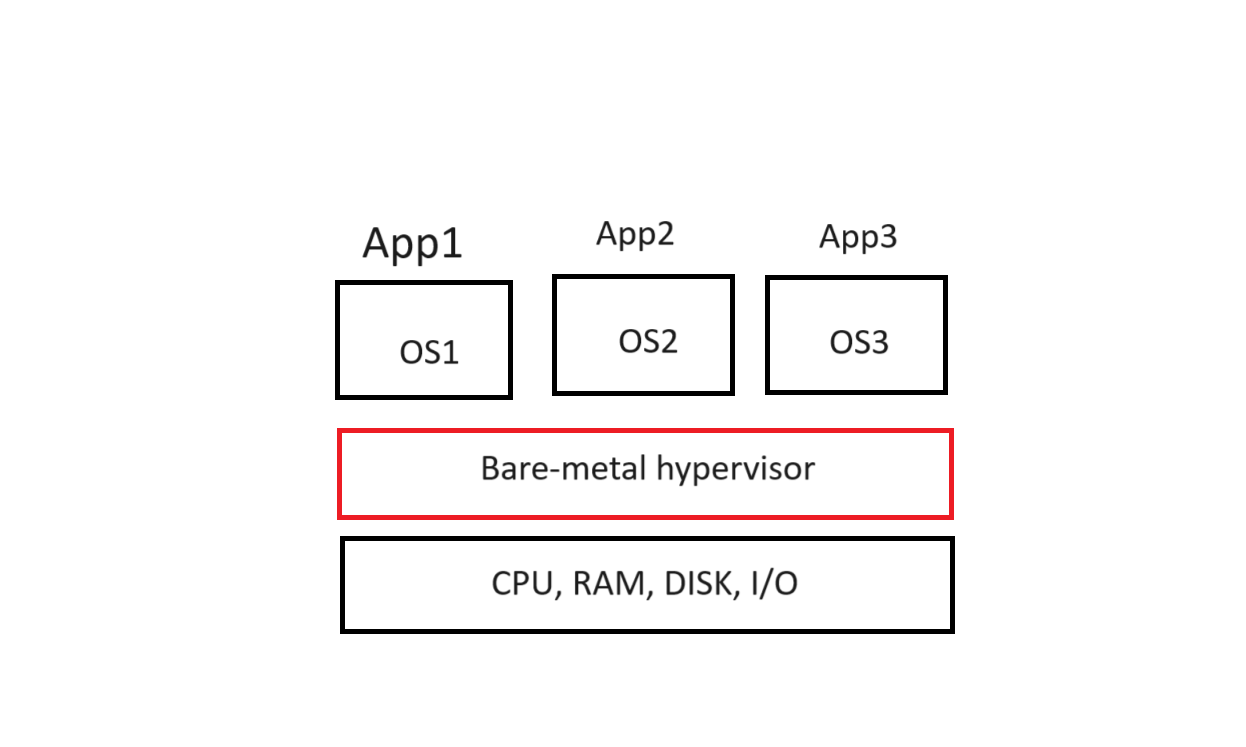
* bin: file thực thi
* lib: file thư viện
* config file: file cấu hình

Ngoài ra còn có môi trường hệ thống, dữ liệu, ...

Như vậy mỗi ứng dụng sẽ chạy đc trên những môi trường hệ thống nhất định. Chẳng hạn phiên bản Docker cho Linux sẽ ko thể nào chạy trên Windows, nếu ko có phiên bản dành cho Windows, phải làm thế nào? Lúc này, ý tưởng về việc ảo hóa hệ điều hành ra đời. Nghĩa là sẽ có một phần mềm, cấp phát tài nguyên cần để tạo ra một máy ảo với một hệ điều hành khác, dựa trên tài nguyên vật lý thực của máy tính, chạy trên hệ điều hành hiện tại đang dùng. Đó chính là khái niệm “Hosted hypervisor”. Ví dụ về một phiên bản phổ biến với , đó là VMware Workstion.



Tuy vậy, cách làm này vẫn chưa giải quyết được vấn đề là hiện tượng “bottleneck”, chẳng hạn là RAM bottleneck (đầy RAM), dẫn đến hiện tượng máy chủ bị treo. Lúc này, ý tưởng về ảo hóa phần cứng (Bare-mel hypervisor) ra đời. Hiểu một cách đơn giản hypervisor này chỉ quản lý việc cung cấp tài nguyên cho máy ảo dựa trên tài nguyên vật lý thực, còn mọi thao tác về hệ thống sẽ thực hiện trên máy ảo, và như vậy hiện tượng “bottleneck” sẽ không xảy ra nữa.



Phân khúc bare-mel hướng đến đối tượng là các doanh nghiệp, do đó sẽ không nghiên cứu về phần này. Chủ yếu sử dụng là các hosted hypervisor.

Như vậy thấy, để chạy một ứng dụng sẽ cần các file bin, lib, config và môi trường hệ thống. Và để việc kiểm tra các ứng dụng mới được phát hành nhanh chóng và tiện lợi hơn, thay vì việc phải tốn thời gian cài đặt máy ảo và các công cụ, thư viện hỗ trợ, khái niệm Container ra đời. Container sẽ đóng gói tất cả các file bin, lib, config, môi trường hệ thống, dữ liệu, ... cần để cho ứng dụng chạy, sau đó chỉ việc nhận Coniner này và sử dụng Coniner engine để chạy nó. Các Containner engine phổ biến là Docker, Containerd, Podman, ...

2. Các khái niệm về phân vùng ổ đĩa

Mount point?

Trước hết, sẽ làm quen với thuật ngữ “mount point”. Ví dụ khi khởi động máy tính, lúc này hệ điều hành chưa được chạy, các phân vùng ổ đĩa chưa được kết nối. Khi đó kernel sẽ tìm kiếm những điểm kết nối với những phân vùng ổ đĩa tương ứng để kết nối các phân vùng vào các file system và khởi chạy hệ thống. Mount point chính là các điểm kết nối đó. Trong Linux, mount point là nơi mà các thiết bị lưu trữ (ví dụ: ổ đĩa, phân vùng) được gắn kết vào hệ thống tệp (FS).

Phân vùng?

Trên hệ điều hành Linux, cần tối thiểu 3 phân vùng đó là:

* /boot : Chứa kernel của Linux, kích thước phân vùng tối thiểu 100 MB.
* /: Thư mục gốc của hệ thống, chứa tất cả các file hệ thống, kích thước phân vùng tối thiểu 4GB.
* swap: lưu ý ở đây, swap không phải là một mount point, swap là một phân vùng đặc biệt được sử dụng để lưu trữ dữ liệu từ bộ nhớ RAM khi bộ nhớ RAM đã đầy, giúp hệ thống duy trì hiệu suất khi không còn đủ bộ nhớ RAM vật lý. Kích thước của swap sẽ được xác định như sau:

Nếu RAM < 32GB, swap = 2xRAM;

Nếu RAM [32;96]GB, swap = RAM;

Nếu RAM > 96GB, swap = 96GB.

Như vậy cần tới 3 phân vùng cho Linux, trong khí đó Windows chỉ cần tối thiểu 1 phân vùng. Tuy nhiên đó là điều khiến Linux ưu việt hơn Windows trong quản lý hệ thống vì Linux có swap và nhân Linux rất mạnh trong việc chạy multisk.

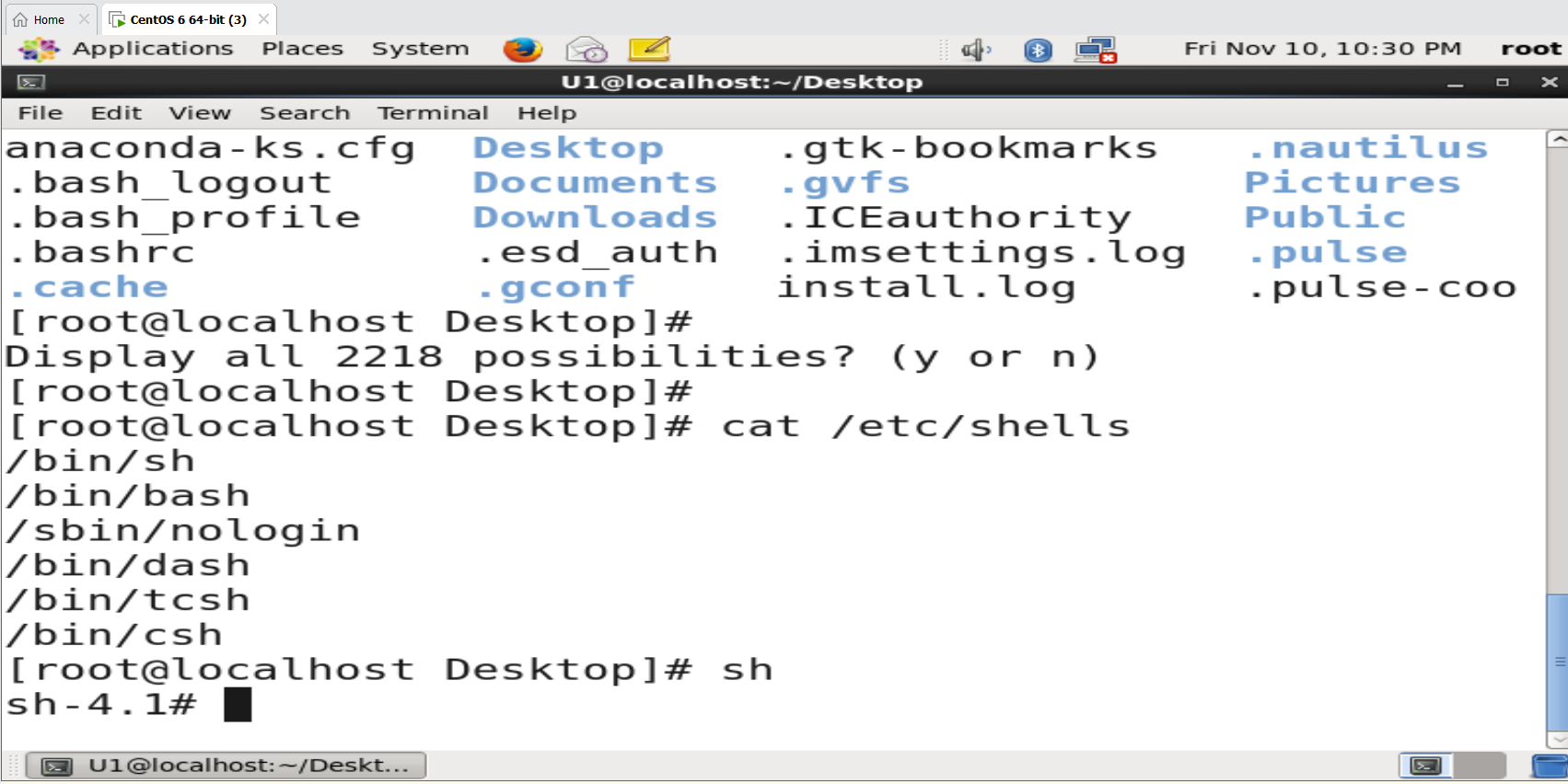
Buổi 3: Shells và kỹ thuật dùng lệnh

Lưu ý: đăng nhập vào hệ thống bằng tài khoản root để được toàn quyền thao tác trên hệ thống.

1. Shells:

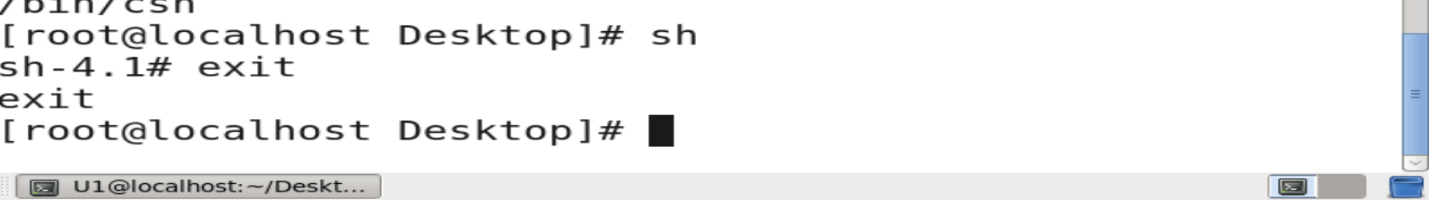
Môi trường shell mặc định trên các phiên bản Linux là bash shell. Thông tin về các shells mà phiên bản Linux đang chạy hỗ trợ sẽ được lưu ở file /etc/shells.

có thể tạm thời thay đổi shell làm việc hiện tại, bằng cách nhập shell được khai báo trong /etc/shells. Ví dụ:



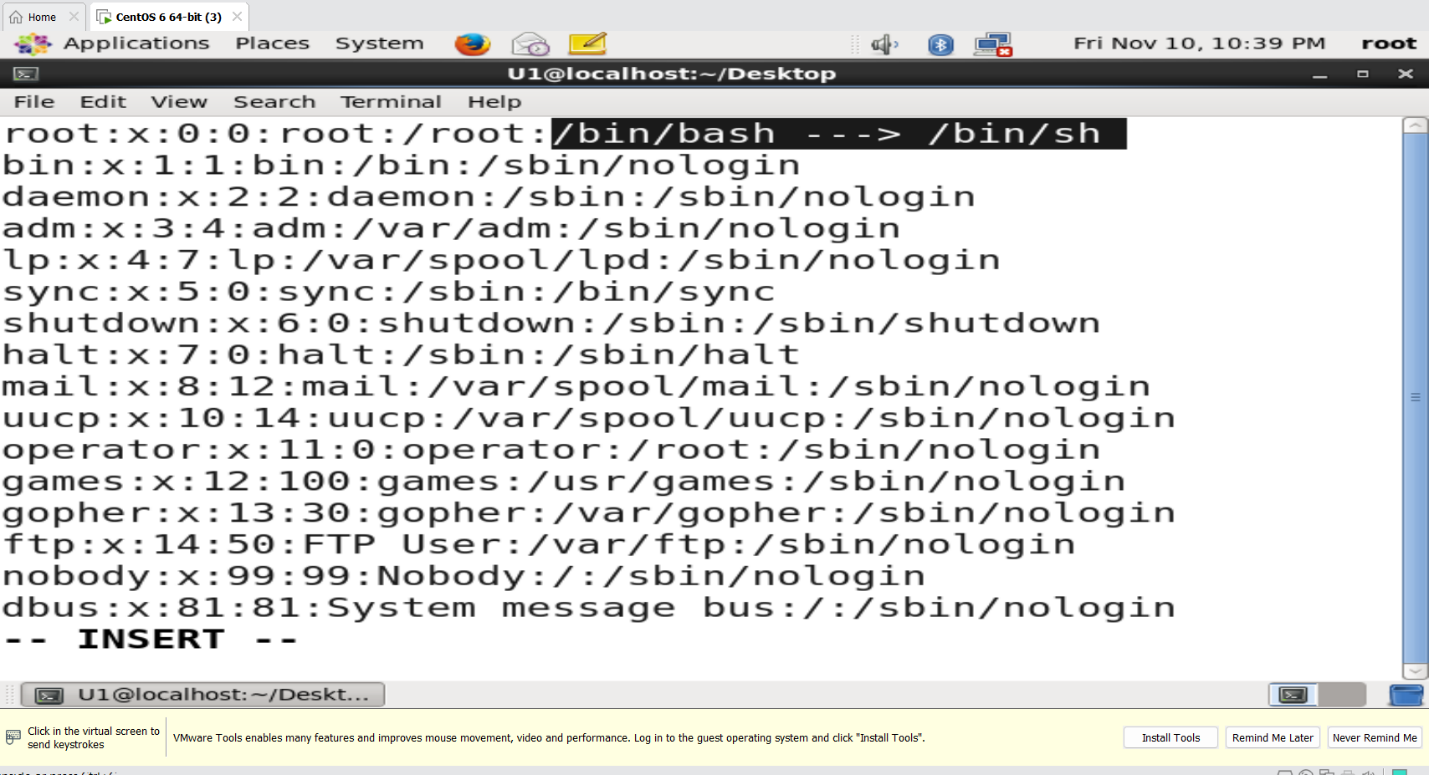
Như vậy đã chuyển sang sh (Bourne shell), shell này là một shell con (subshell) của shell bash gốc ban đầu. Mọi thay đổi về môi trường ở shell con này sẽ không anh hưởng đến shell gốc. Ngược lại shell con sẽ chịu mọi ảnh hưởng từ shell gốc.

Để thoát khỏi shell con, dùng lệnh exit.



Nếu muốn thay đổi vĩnh viễn shell mặc định, sẽ phải sửa file cấu hình của nó. Shell mặc định của người dùng sẽ được khai báo ở file /etc/passwd

Ví dụ như bên dưới, đổi /bin/bash của người dùng root thành /bin/sh



Sau đó reboot hệ thống thì shell mặc định cho root sẽ là Bourne shell.

2. Câu lệnh

có hai nhóm lệnh cơ bản trên hệ thống:

* Lệnh nội bộ (internal commands): Là những lệnh được tích hợp vào các công cụ hệ thống, đa số không nằm trong đường dẫn của biến PATH
* Nhóm lệnh còn lại, bao gồm các lệnh thực thi cơ bản trong /bin, hay những lệnh quản lý hệ thống trong /sbin, ..., là những câu lệnh nằm trong đường dẫn của biến PATH.

Một câu lệnh cơ bản sẽ có 3 thành phần cấu tạo và có thứ tự lần lượt: cmd, options, arguments.

Ví dụ: ls –a /root

* ls – cmd
* -a – option
* /root - argument

Option sẽ chia ra làm 2 dạng là:

* Long option: --option, dạng này phải viết đầy đủ tên option. Ở dạng này không thể kết hợp các options mà chỉ có thể dùng 1 option
* Sort option: ký hiệu là –a, a ở đây là ký tự viết tắt của option nào đó. Dạng này có thể kết hợp nhiều option. Ví dụ : ls –al

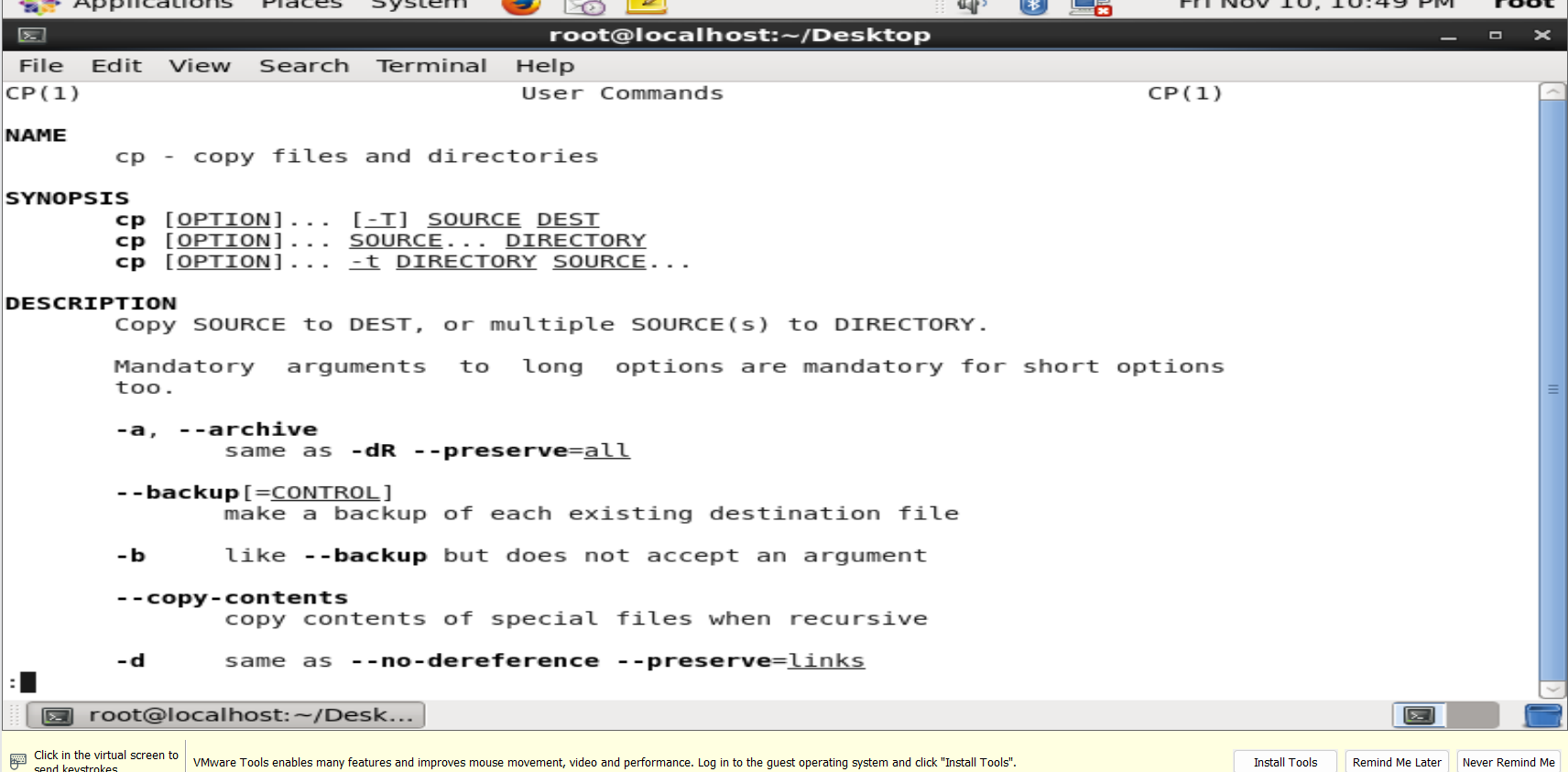
Lưu ý:

* sử dụng nút b để hoàn thành câu lệnh, cũng thể dùng để hiện gợi ý về những câu lệnh có cùng ký tự đang nhập.
* Option --help sẽ có cho hầu hết các câu lệnh. Option này sẽ giúp hiểu lệnh này dùng để làm gì, dùng như thế nào và có các options gì.

3. Kỹ thuật dùng câu lệnh man:

man là viết tắt của mannual. Trong các hệ điều hành như Linux và macOS, lệnh "man" được sử dụng để truy cập các trang hướng dẫn (manual pages) cho các lệnh, hàm hệ thống, và các chủ đề khác.

Ví dụ: man cp



Trong trang man, có thể tìm kiếm các nội dung chứa từ mà muốn tìm bằng cú pháp: /keyword

Lúc này đã vào chế độ tìm kiếm của man, dùng các phím sau để di chuyển:

* n – di chuyển xuống
* Shift n – di chuyển lên

Sau đây sẽ sử dụng lệnh man để tìm kiếm các câu lệnh phù hợp với mô tả của hành động.

* Đầu tiên cần tạo cơ sở dữ liệu cho description của các câu lệnh:

makewhatis (nếu Linux là nền tảng sysv)

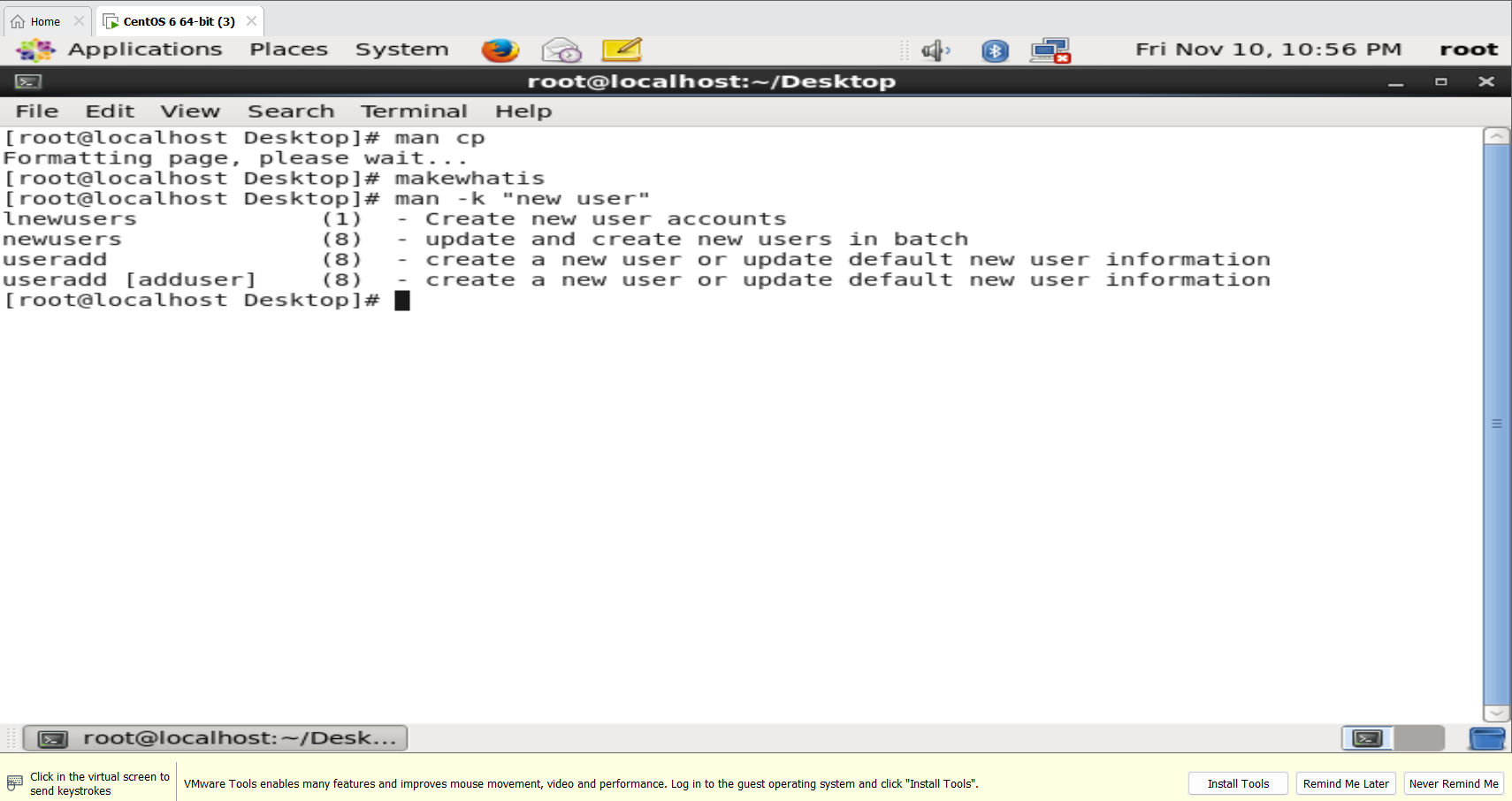
mandb (nếu Linux là nền tảng systemd)

* Sau đó, có cú pháp tìm câu lệnh:

man –k “des\_action”

Sau đó man sẽ hiện thị cho những câu lệnh phù hợp với mô tả.

Ví dụ: man –k “new user”



4. Một vài tính năng đặc biệt trên môi trường bash shell:

Trước hết gọi lại lịch sử dùng các câu lệnh bằng lệnh sau:

history

Gọi lại câu lệnh khi biết số thứ tự của nó trong lịch sử: !<number of line>

Thực hiện lại câu lệnh cuối cùng: !!

Gọi lại 1 đoạn lệnh: fc <number of first cmd> <number of last cmd>

5. Biến môi trường:

Đây là những biến định nghĩa môi trường shell đang làm việc.

Xem những biến môi trường nào đang có trên shell bằng lệnh:

env (hoặc là printenv)

có thể khai báo một biến trên shell bằng cú pháp sau:

export <name\_var>=<value>

Xóa một biến:

unset <name\_var>

Những thay đổi này chỉ có hiệu lực trên phiên làm việc hiện tại. Nếu muốn thay đổi vĩnh viễn thì phải tìm config file chứa biến muốn thay đổi và sửa trong đó. Các config file lưu ở thư mục /etc/profile.

Ở bài này sẽ giới thiệu 2 biến quan trọng của môi trường là PATH và PS1.

***PATH***

Biến PATH chứa một danh sách các đường dẫn đến nơi chứa file chạy khi nhập lệnh

có thể xem giá trị của biến bằng lệnh echo:

echo $PATH

Dấu $ để thông báo cho echo biết PATH là 1 biến

***PS1***

Biến này sẽ định nghĩa dòng Prompt hiển thị như thế nào.

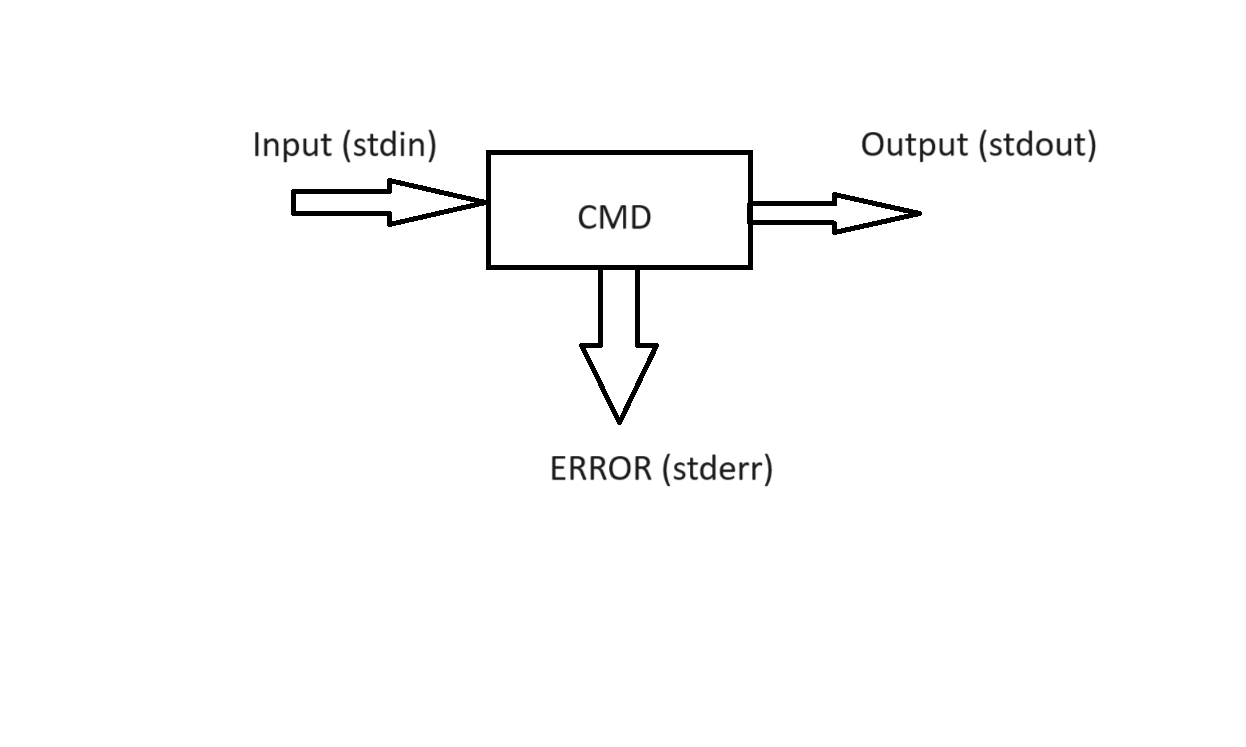
Buổi 4: Unix stream và các kỹ thuật dùng lệnh

1. UNIX STREAMS:

Có ba luồng dữ liệu được gọi là UNIX STREAMS và được ký hiệu như sau:

* Dữ liệu đầu vào chuẩn: stdin
* Dữ liệu đầu ra chuẩn: stdout
* Dữ liệu đầu ra lỗi: stderr

Khái niệm này là khái niệm chung cho mọi hệ thống mã nguồn mở hiện nay.



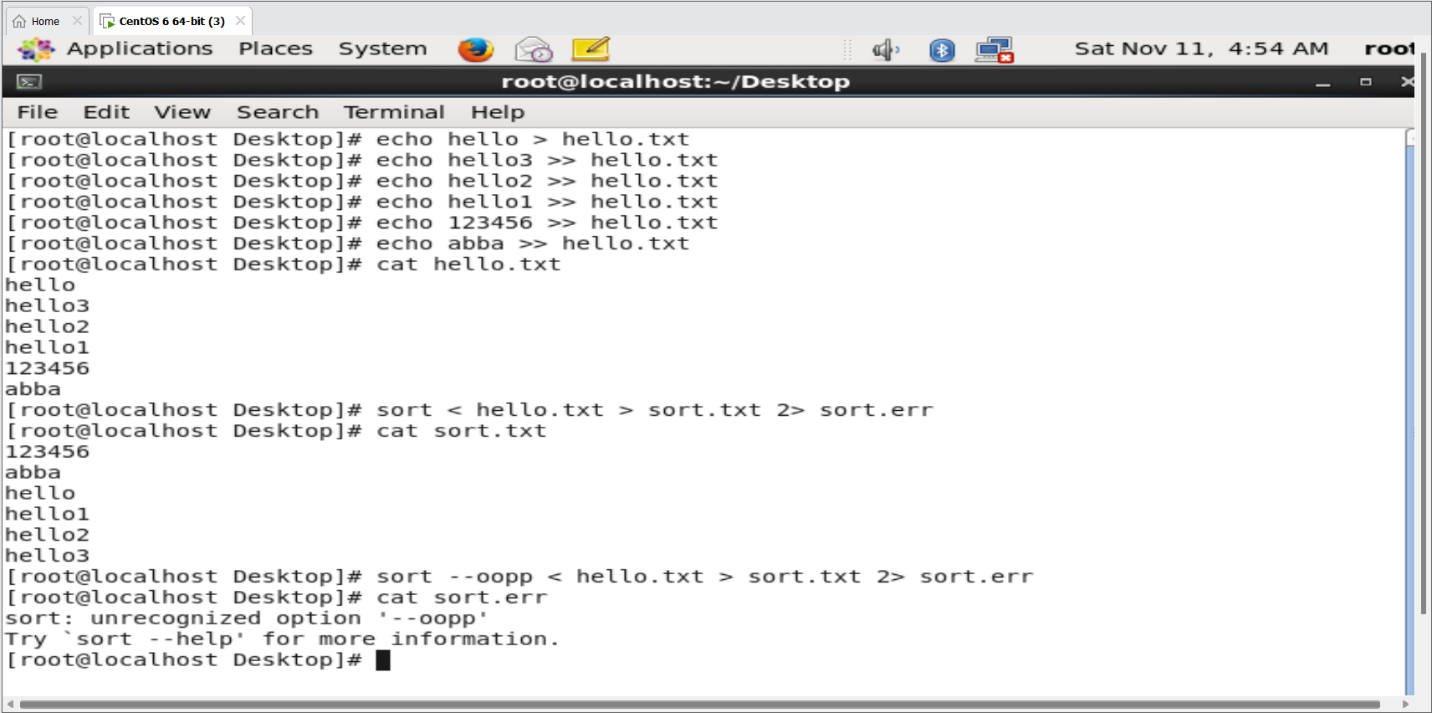
Các kỹ thuật điều hướng dữ liệu trên Linux:

có 3 cách điều hướng tương ứng với ba luồng dữ liệu trên Linux và ký hiệu của nó lần lượt là:

* Điều hướng dữ liệu đầu vào chuẩn: <
* Điều hướng dữ liệu đầu ra chuẩn: >, >>
* Điều hướng dữ liệu đầu ra lỗi: 2>,2>>

Lưu ý là > và 2> là các cách điều hướng ghi đè dữ liệu, còn >> và 2>> là các cách ghi nối tiếp.

ví dụ về các kỹ thuật điều hướng như sau:



Kỹ thuật điều hướng đầu ra và đầu vào về cùng 1 tập tin:

Phương pháp của cách này là điều hướng 1 luồng dữ liệu, rồi cho luồng còn lại đi theo luồng này. Bằng cách này chỉ cần 1 lần điều hướng dữ liệu nhưng cả dữ liệu chuẩn và lỗi đều về cùng 1 tập tin.

Thao tác này ký hiệu là: >&

Trong thao các này, các luồng dữ liệu được quy ước như sau:

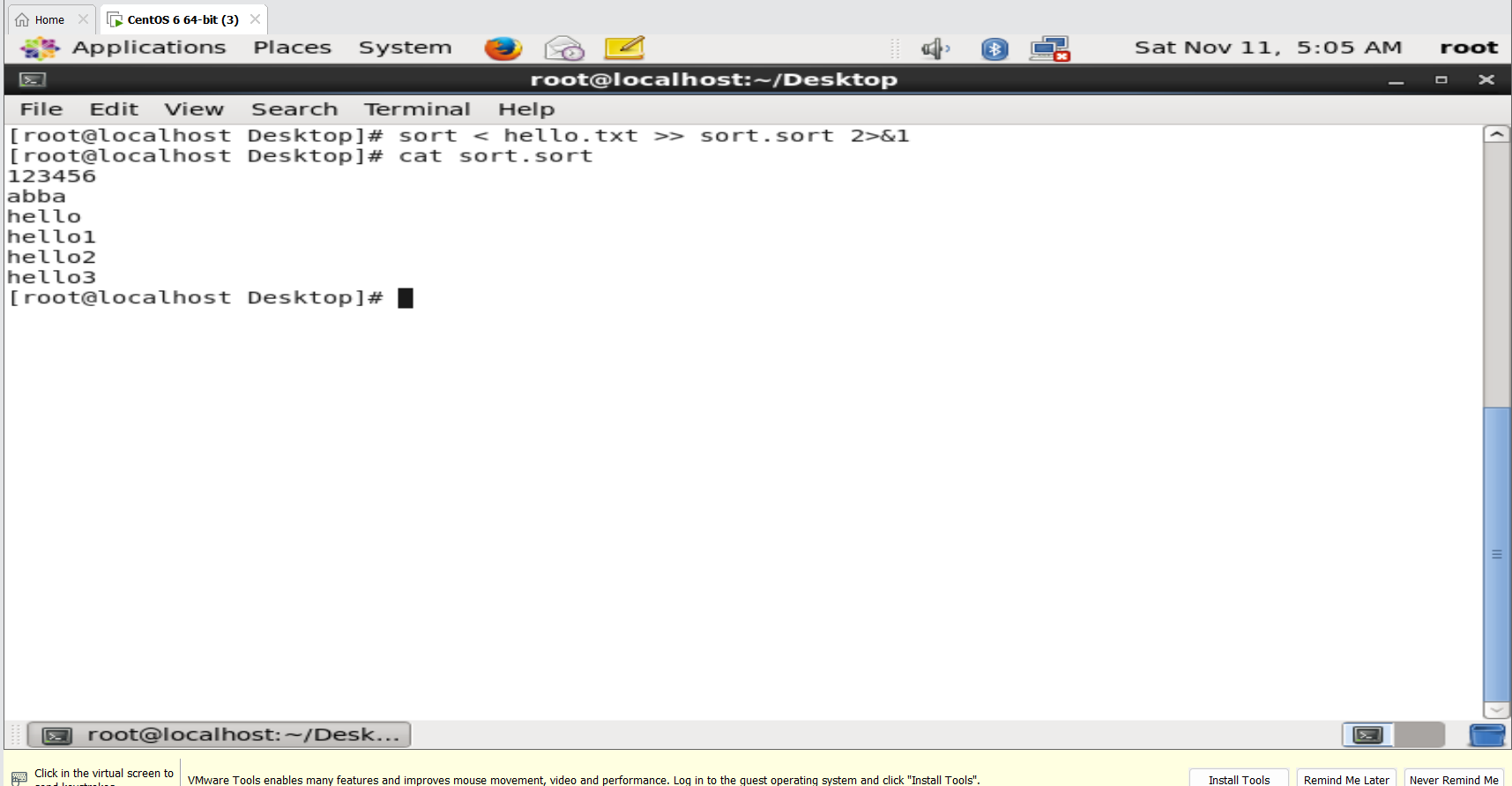
* stdin: 0
* stdout: 1
* stderr: 2

Thao tác này như sau: i>&j, trong đó i và j là các luồng dữ liệu

Kết quả của thao tác này là j=j+i. Đây không phải ý nghĩa về mặt số học, mà là ý nghĩa về điều hướng, nghĩa là i sẽ đi theo j sau thao tác này.

Chú ý: Thao tác >& là một thao tác đặc biệt. Trong linux, tất cả các thao tác, option đặc biệt, hoặc dễ gây nguy hiểm cho hệ thống, đều được quy ước là đứng cuối câu lệnh.

Ví dụ: sort < hello.txt >> sort.sort 2>&1

Lưu 

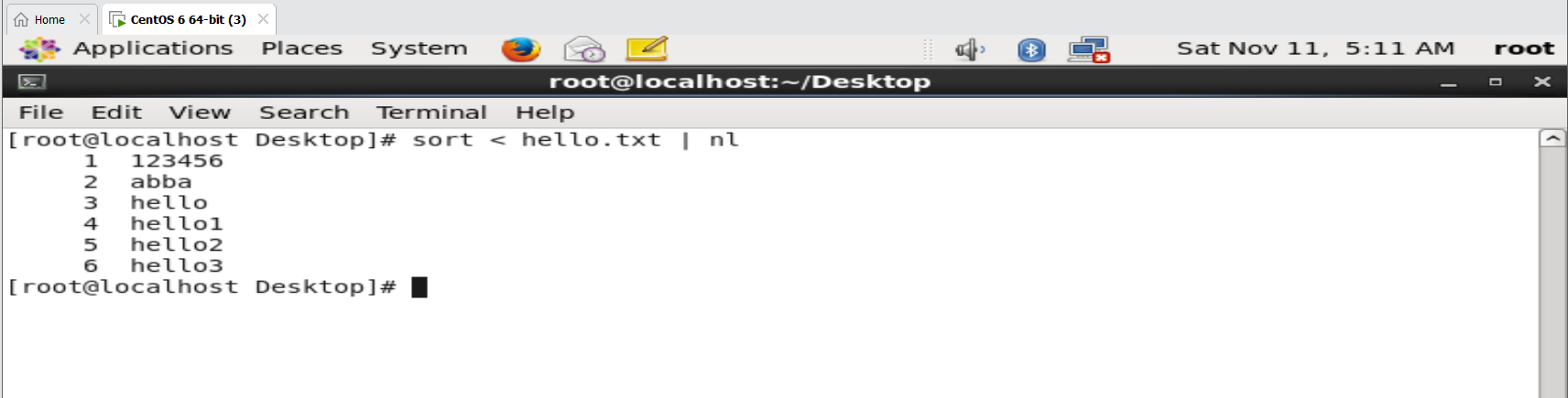
Lưu ý: Đuôi file không có ý nghĩa đối với hệ điều hành, kernel sẽ phân biệt loại file dựa trên nội dung file . Cách đặt tên này chủ yếu để người dùng nhớ file mình đã đặt là gì.

2. Kỹ thuật đường ống

Kỹ thuật này ký hiệu bằng toán tử pipe “|”. Kỹ thuật này sẽ cho phép kết nối đầu ra của một lệnh với đầu vào của một lệnh khác, tạo ra một chuỗi các lệnh được kết hợp mà không cần sử dụng tệp trung gian.

Cú pháp của toán tử này: cmd1 | cmd2...

Ví dụ: sort < hello.txt | nl

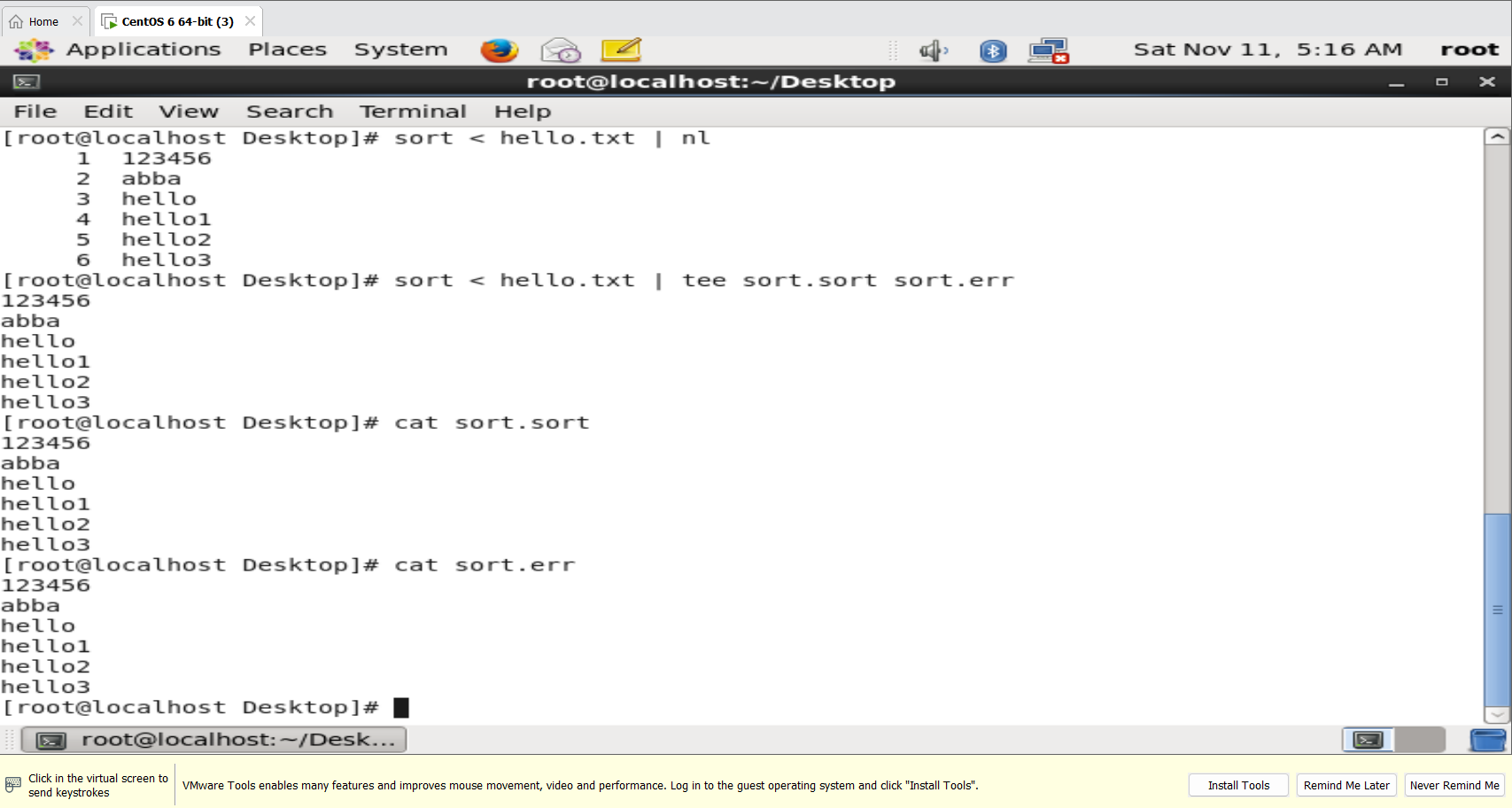


3. Công cụ tee:

Công cụ này được thiết kế để đọc từ đầu vào chuẩn (stdin) và ghi vào đầu ra chuẩn (stdout) cũng như vào một hoặc nhiều tập tin. Công cụ này giúp sao chép đầu vào của một ống đến một hoặc nhiều tệp tin cũng như in ra màn hình.

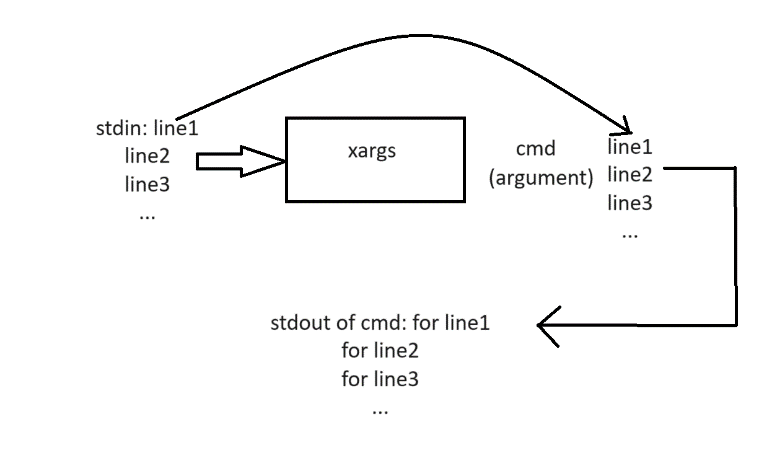
Cú pháp: cmd | tee file1 file2 file3 ...

Sau đó kết quả lệnh cmd sẽ được in ra màn hình



4. Công cụ xargs:

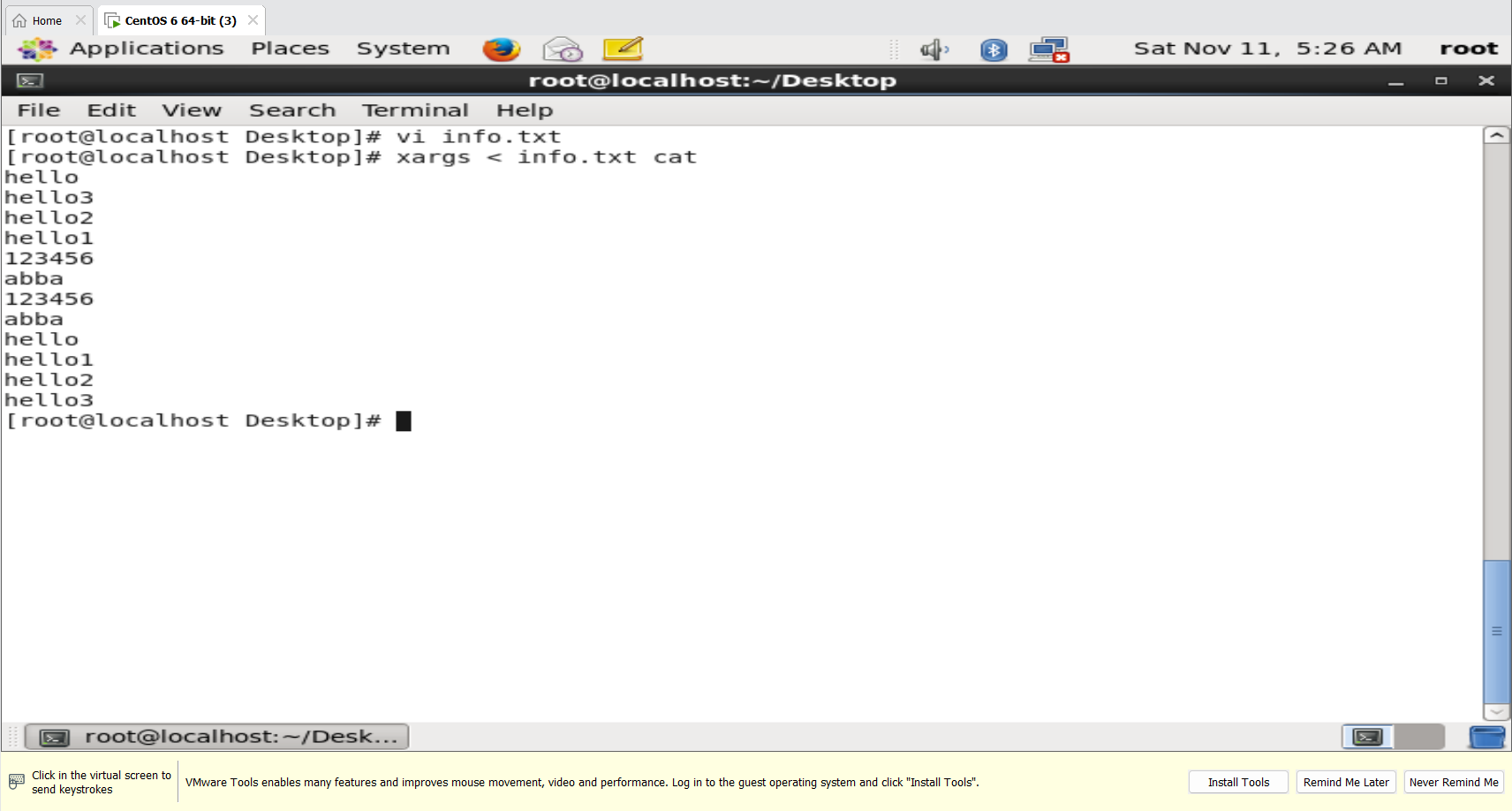
Công cụ này được thiết kế để chuyển đối số từ đầu vào chuẩn (stdin) và thực hiện một lệnh được chỉ định với các đối số đó. Nó thường được sử dụng khi muốn thực hiện một lệnh với một danh sách các đối số mà có từ đầu vào chuẩn hoặc từ một tập tin.



Ví dụ: xargs < info.txt cat

Nội dung info.txt: /root/Desktop/hello.txt

/root/Desktop/sort.sort



5. Lệnh grep và các ký tự đặc biệt

Lệnh này được sử dụng để tìm kiếm và in ra các dòng trong văn bản phù hợp với một mẫu (pattern) cụ thể.

Các ký tự đặc biệt:

&: Đặt ở cuối dòng lệnh, ý nghĩa là chạy lệnh ở nền mà không chờ việc thực thi kết thúc

\*: 1 chuỗi bất kỳ hoặc rỗng

. :1 ký tự bất kỳ, không thể để rỗng

/ : Hủy ý nghĩa đặc biệt của ký tự đặc biệt

$ : Ký tự kết thúc của dòng

^ : Ký tự ở đầu dòng

<<: Khai báo một chuỗi, báo hiệu khi nhập vào chuỗi này thì quá trình nhập stdin kết thúc

Ví dụ: cat hello.txt | grep \*h : Lệnh này in ra các dòng chứa ký tự h

Cat hello.txt | grep ^h : Lệnh này in ra các dòng chứa ký tự h ở đầu dòng.