Real Time Operating Systems

LAB 1: GETTING ACQUAINTED WITH LINUX

DAVIDE.CHIARAVALLI@STUDIO.UNIBO.IT

Lab 1

Objective: gain access to the basic functionalities of Linux systems

The experience consist of 4 parts:

1) new account creation

2) initial setup

3) practice with command line interface

4) explore programming environment

Account creation

Access to laboratory machines requires a personal account

It can be created :

- on the internet from a laptop following the instructions at: http://lab2.deis.unibo.it/node/3

- inside the lab by selecting «Create new account» in the operating system boot menu.

If the account has been created inside the lab then the system must be rebooted in order to start Linux operating system

*Pressing **Ctrl-alt-canc** will reboot the laboratory machine*

Account access

The Linux 14.04 LTS version used during lab sessions can be chosen from the operating system boot menu (second choice from top).

To gain access, the formerly created account information are requested:

- The user ID of a lab account is like : *s0000714820 (s + your university badge ID)*

- The password is the one set during the account creation.

The lab machine will not prompt the password while you are writing it so do not worry if the cursor is not moving on the screen:

Just keep typing and press enter.

Xubuntu’s GUI

Xubuntu lab’s version provides a graphical user interface (GUI).

Entering **startx** command will open the interface that presents at the bottom of the screen :

- Web browser
- File system manager
- Application finder
- Terminal emulator

File system

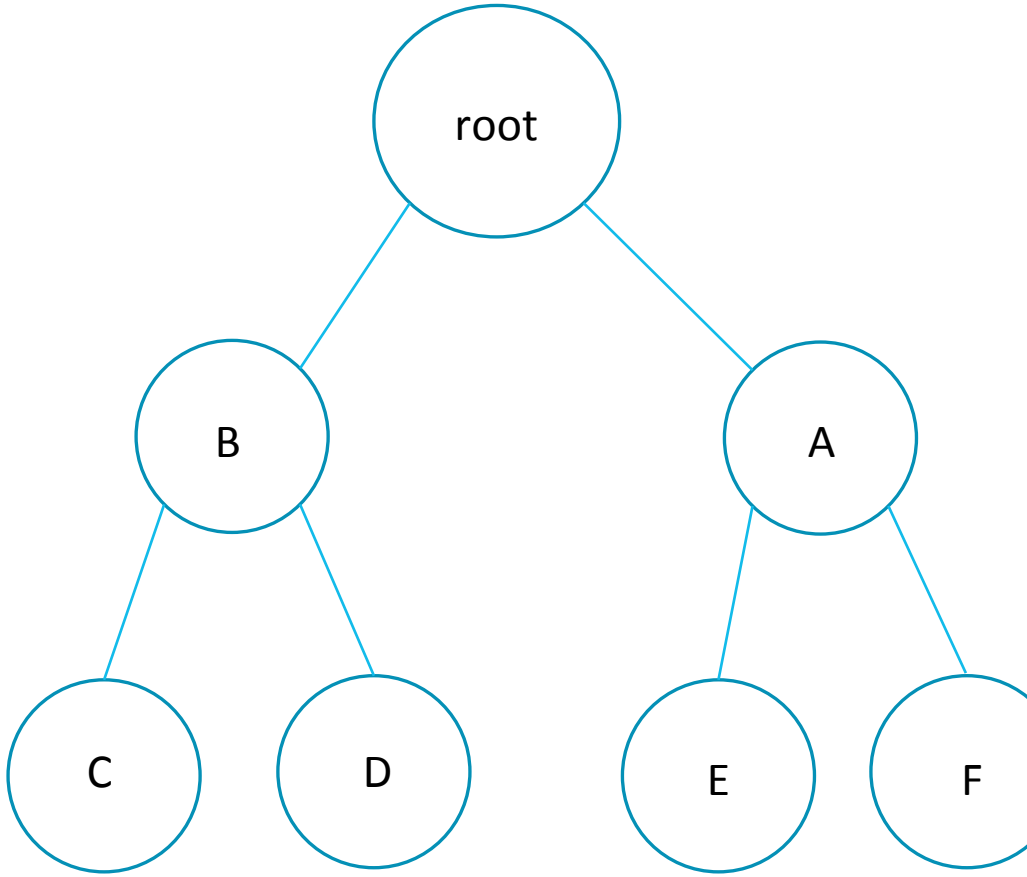
The file system is structured as a tree of directories.

The main directory is called *root*

Other directories are contained inside the root

Starting from any directory it is possible to define a path to any other moving along the tree

The tree example in the picture will be considered in all the following slides



File system path

A path can be defined in two way:

- the **absolute path** is always defined starting from the root up to the destination

«/» simbolizes the root: a path declaration starting with / is always an absolute path

e.g. the absolute path to directory F is : /A/F

- the **relative path** starts from the current directory up the destination

«./» references the current directory

«../» references the directory the current one is inside

e.g. the relative path from E to F is : ../F

Terminal emulator

Interaction with the file system is carried out through a command-line interface

A set of commands exploits a series of possible function the system can execute

Every time a new terminal is opened it chooses as current working directory the default one:

in the lab the default directory is located inside the user account and has the personal account name requested during the login procedure (e.g. s0000714820)

Terminal emulator

Commands are always executed inside the current directory unless a path to another one is added (let's consider «E» as current directory):

- «ls» command will be executed inside directory «E»

- «ls /A/F» will be executed inside directory «F» (absolute path)

- «ls ../F» will be executed inside directory «F» (relative path)

Terminal emulator

The command pwd returns the absolute path to the current working directory.

The command will return a path like:

/afs/numi.ing.unibo.it/users/4/5/9/s0000714820 (default directory's path in the lab)

The command ls returns a list of all files and directories that are located inside the current directory:

- files usually start with a small letter (e.g. naming)

- directories usually start with a capital letter (e.g. Desktop)

Manual

Commands usually can be slightly modified adding some predetermined options in the script.

For example adding «-l» to the ls command will make it print a more detailed list

Remember always to add a space between command and option (e.g ls -l)

In order to know what a command does and all its possible functionalities and options a manual can be called through the man command.

«man ls» will open the manual of the «*ls*» command showing all the possible options

Pressing the «q» key will close the manual and return the user to the previous command-line

Change directory

The cd command allows the user to change the current working directory.

It needs as argument the path to the new directory (absolute or relative)

Let's consider working in directory B:

«cd /A/F» will set F as new directory (absolute path)

«cd ../A/F» will set F as new directory (relative path)

«cd C» will set C as new directory (./ usually not required)

If the command is called without an argument it will set the default directory as the new one.

Useful commands

The echo command will repeat on the screen everything is written after it.

The function will run until a special character is digited: **«ctrl+D»** used to signal the end of the line.

In lab's Linux version echo will only repeat what is written in its same line, so the end-of-line character is not needed.

The cat command prompt on the screen the content of the file given as argument:

«cat hello» will prompt on the screen the content of the file named «hello»

The command will work only if it can find the file in the current directory, otherwise it needs the path to reach it (e.g. cat /A/F/hello)

Useful commands

While cat prints directly all the content on the screen, less exploits the same function with some peculiar differences:

- it prompts the text page by page allowing the user to scroll up and down

- it allows to run some regular expression search on the printed text

In order to enter a new function the user must close the less function first:

pressing the «q» key will close the function and return the user to the next command line

File editor

The nano command will open a text editor where file can be created or modified.

«nano hello» will open the file named «hello» and show its content allowing the user to modify it.

If the «hello» file does not exist it will create a new file with that name.

In the lower part of the screen an help window usually displays some useful command for the editor as :

- ctrl+G (^G) for opening the help menu

- ctrl+O (^O) for saving everything and closing the file

- etc….

File execution

Instead of writing directly on the terminal, frequently used commands can be saved on a file.

In this way every time the user needs that particular combination of commands he has just to execute the file.

In order to execute a file the system needs his path:

«./hello» will execute the file named «hello» and situated in the current directory

If inside the file there are some line of code that are not recognised as commands an error will appear during execution

Change permissions

Files have three level of permission: read (r), write (w) and execute (x)

Each level, if enabled, will allow the user to respectively read, modify or execute the file.

In order to execute a file it's necessary that it has the execution permission

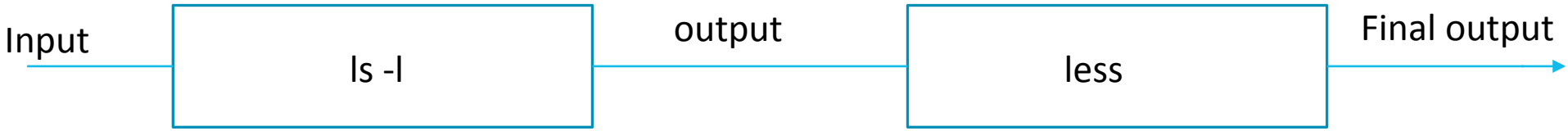
The chmod +x command will enable all the execution permissions on a file

«chmod +x hello» will turn the file into a shell script executable by the user

Pipeline instructions

A pipeline is composed by a series of commands, each one acting on the output of the previous one.

«ls –l | less» apply the *less* command on the output of the *ls* function



«|» is used as chaining simbol between the functions

The pipeline is used when only the final output result is of some interest.

Input/Output redirection

Command's result can be redirected or taken from a file:

«ls > hello» is used to save the output of the ls command inside a file named hello.

The file is created in the current directory

If a file with that name already exists the command interpreter will totally overwrite it causing all previous information to disappear.

«>» simbol redirects the result of the function on the left inside the file on the right

«*command* > *file* »

«<» simbol makes the function on the left use the file content on the right as input

«*command* < *file* »

Useful hints

1. The **«tab»** key can be used to speed-up writing functions or paths: pressing it without having finished typing a path will make the system try to finish it automatically. This turns out very valuable when there is only one possible completion chance over a long word (s0000714820 can be written just by typing s00 and pressing tab : the system will add the remaining part automatically)
2. **«Crtl+C»** in most cases shuts down the currently working function
3. Pressing the **«Alt»** key in the web browser will allow the user to have access to some useful options