appunti.md

LEZIONE 1

INTRO

La Shell è un programma che prende un'input da tastiera e lo trasferisce all'OS. Solitamente preinstallata nelle distribuzioni Linux si ha BASH, *Bourne Again SHell*. Quando si usa una GUI si necessita di un programma, **l'emulatore di terminale** per interagire con la shell. Quando la shell è pronta a ricevere l'input si ha lo **shell prompt**:

```
[me@linuxbox ~]$
```

Se alla fine si ha il simbolo # si indica che si hanno i permessi di *superuser*:

```
[me@linuxbox ~]#
```

Nella shell si ha la *history* dei vecchi comandi (solitamente fino a 1000) utilizzando le frecce su e giù. Con le frecce destra e sinistra ci si sposta col cursore nel comando. Vediamo qualche comando base d'esempio:

- Per ora e data corrente:

  ```
  [me@linuxbox ~]$ date
  Thu Mar 8 15:09:41 EST 2018
  ```

- Per il calendario:

  ```
  [me@linuxbox ~]$ cal
  March 2018
  Su Mo Tu We Th Fr Sa
                1  2  3
   4  5  6  7  8  9  10
  11 12 13 14 15 16 17
  18 19 20 21 22 23 24
  25 26 27 28 29 30 31
  ```

- Per vedere l'uso di memoria sui dischi e sulle varie partizione:

  ```
  [me@linuxbox ~]$ df
  File system    1K-blocks       Used  Available Use% Mounted on
  dev               4014968          0   4014968   0% /dev
  run               4023532       9784   4013748   1% /run
  /dev/sda28      131449884  112330780  12412112  91% /
  tmpfs             4023532      36680   3986852   1% /dev/shm
  ```

- Per vedere l'uso della RAM:

  ```
  [me@linuxbox ~]$ free
                total        used        free      shared     buff/cache  available
  Mem:        8047068     2504188     1657424      357836     3885456     4839276
  Swap:             0           0           0
  ```

- Per terminare la sessione uso o Ctrl-d o:

  ```
  [me@linuxbox ~]$ exit
  ```

Inoltre si hanno delle sessioni di temrinale che girano sempre dietro la gui e sono accessibili con Ctrl-Alt + i tasti da F1 a F6; generalmente con Alt-F7 si torna alla GUI.

NAVIGAZIONE

Nei sistemi UNIX i file sono organizzati in una *hierical directory structure*, quindi in una struttura ad albero. La prima cartella è quindi chiamata *root directory* e contiene files e altre sottocartelle. I sistemi UNIX e UNIX-Like hanno un unico albero per tutti i device i quali vengono in caso montati in un determinato punto dell'albero a seconda del volere di chi gestisce il sistema.

Per vedere in che posizione dell'labero ci troviamo (ovvero in che cartella ci troviamo) si ha:

```
[me@linuxbox ~]$ pwd
/home/me
```

Inoltre all'avvio ci troveremo nella cartella */home*, diversa per ciascun account. Infatti dentro */home* si hanno le varie sottocartelle degli utenti (una per ogni utente). Per vedere cosa contiene una cartella, elencando files e directories uso:

```
[me@linuxbox ~]$ ls
Desktop Documents file.txt Music Pictures Public Templates Videos
```

Specificando il percorso possiamo vedere il contenuto di una determinata cartella

```
[me@linuxbox ~]$ ls Documents
document.txt other
```

per elencare anche file e directory nascoste uso:

```
[me@linuxbox ~]$ ls -a Documents
document.txt other .test .test.d
```

possiamo specificare più directories:

```
[me@linuxbox ~]$ ls ~ Documents
/home/me:
Desktop Documents file.txt Music Pictures Public Templates Videos
/home/me/Documents:
document.txt other
```

possiamo anche richiedere informazioni sui file:

```
[me@linuxbox ~]$ ls -l Documents
drwxr-xr-x 2 me me 4096 30 apr 11.37 other
-rw-r--r-- 2 me me 4096 30 apr 11.37 document.txt
```

con:

Field	Meaning
`-rw-r--r--`	Access rights to the file. The first character indicates the type of file. Among the different types, a leading dash means a regular file, while a “d” indicates a directory. The next three characters are the access rights for the file's owner, the next three are for members of the file's group, and the final three are for everyone else. Chapter 9 "Permissions" discusses the full meaning of this in more detail.
`1`	File's number of hard links. See the sections "Symbolic Links" and "Hard Links" later in this chapter.
`root`	The username of the file's owner.
`root`	The name of the group that owns the file.
`32059`	Size of the file in bytes.
`2007-04-03 11:05`	Date and time of the file's last modification.
`oo-cd-cover.odf`	Name of the file.

Il percorso di file o di una directory può essere specificato in due modi:

1. **con il PERCORSO ASSOLUTO**, ovvero indicando ogni cartella e sotto cartella fino a quella desiderata, per esempio

/home/Documents/. Si ricorda che il primo "/" indica la cartella di root

2. **con il PERCORSO RELATIVO**, che invece da partiren dalla cartella di root parte dalla cartella corrente. Si hanno anche "." per indicare la cartella corrente e ".." per indicare quella precedente, quindi:

```
[me@linuxbox ~]$ pwd
/home/me
[me@linuxbox ~]$ ls
Documents
[me@linuxbox ~]$ cd ./Documents
[me@linuxbox Documents]$ pwd
/home/me/Documents
[me@linuxbox Documents]$ cd ..
[me@linuxbox ~]$ pwd
/home/me
[me@linuxbox ~]$ cd Documents
[me@linuxbox Documents]$ pwd
/home/me/Documents
```

Shortcut	Result
`cd`	Changes the working directory to your home directory.
`cd -`	Changes the working directory to the previous working directory.
`cd ~user_name`	Changes the working directory to the home directory of *user_name*. For example, `cd ~bob` will change the directory to the home directory of user "bob."

In UNIX i nomi di file e directory sono *case sensitive* e non si usano le estensioni per determinare il contenuto di un file. Per comodità **NON SI USANO SPAZI NEI NOMI DI FILES E DIRECTORIES**

Come abbiamo visto si possono dare opzioni ai vari comandi con il simbolo "-" seguito da una lettera, oppure, in modalità estesa, con "--" seguito dal nome dell'opzione. Per *ls* si ha, per esempio:

Option	Long Option	Description
`-a`	`--all`	List all files, even those with names that begin with a period, which are normally not listed (that is, hidden).
`-A`	`--almost-all`	Like the `-a` option above except it does not list `.` (current directory) and `..` (parent directory).
`-d`	`--directory`	Ordinarily, if a directory is specified, `ls` will list the contents of the directory, not the directory itself. Use this option in conjunction with the `-l` option to see details about the directory rather than its contents.
`-F`	`--classify`	This option will append an indicator character to the end of each listed name. For example, a forward slash (/) if the name is a directory.
`-h`	`--human-readable`	In long format listings, display file sizes in human readable format rather than in bytes.
`-l`		Display results in long format.
`-r`	`--reverse`	Display the results in reverse order. Normally, `ls` displays its results in ascending alphabetical order.
`-S`		Sort results by file size.
`-t`		Sort by modification time.

Per determinare il tipo di file, ricordando che l'estensione non implica alcun tipo preciso di file, si ha il comando file:

```
[me@linuxbox ~]$ file picture.jpg
picture.jpg: JPEG image data, JFIF standard 1.01
```

In Linux *everything is a file* e questo comporta l'esistenza di migliaia di tipi di file.

Per visualizzare file di testo si ha *less*, che permette di spostarsi su e giù con le frecce per visualizzaree il contenuto (per uscire usare "q"):

```
[me@linuxbox ~]$ less /etc/passwd
```

vediamo una tabella con le shortcut principali:

Command	Action
Page Up or `b`	Scroll back one page
Page Down or space	Scroll forward one page
Up arrow	Scroll up one line
Down arrow	Scroll down one line
`G`	Move to the end of the text file
`1G` or `g`	Move to the beginning of the text file
/characters	Search forward to the next occurrence of *characters*
`n`	Search for the next occurrence of the previous search
`h`	Display help screen
`q`	Quit `less`

Torniamo sul directory tree, si hanno le seguenti direcory principali:

Directory	Comments
`/`	The root directory. Where everything begins.
`/bin`	Contains binaries (programs) that must be present for the system to boot and run.
`/boot`	Contains the Linux kernel, initial RAM disk image (for drivers needed at boot time), and the boot loader. Interesting files: • `/boot/grub/grub.conf` or `menu.lst`, which are used to configure the boot loader. • `/boot/vmlinuz` (or something similar), the Linux kernel

Directory	Comments
`/dev`	This is a special directory that contains *device nodes*. “Everything is a file” also applies to devices. Here is where the kernel maintains a list of all the devices it understands.
`/etc`	The `/etc` directory contains all of the system-wide configuration files. It also contains a collection of shell scripts that start each of the system services at boot time. Everything in this directory should be readable text. Interesting files: While everything in `/etc` is interesting, here are some all-time favorites: • `/etc/crontab`, a file that defines when automated jobs will run. • `/etc/fstab`, a table of storage devices and their associated mount points. • `/etc/passwd`, a list of the user accounts.
`/home`	In normal configurations, each user is given a directory in `/home`. Ordinary users can only write files in their home directories. This limitation protects the system from errant user activity.
`/lib`	Contains shared library files used by the core system programs. These are similar to dynamic link libraries (DLLs) in Windows.
`/lost+found`	Each formatted partition or device using a Linux file system, such as ext4, will have this directory. It is used in the case of a partial recovery from a file system corruption event. Unless something really bad has happened to our system, this directory will remain empty.
`/media`	On modern Linux systems the `/media` directory will contain the mount points for removable media such as USB drives, CD-ROMs, etc. that are mounted automatically at insertion.
`/mnt`	On older Linux systems, the `/mnt` directory contains mount points for removable devices that have been mounted manually.

Directory	Comments
`/opt`	The `/opt` directory is used to install "optional" software. This is mainly used to hold commercial software products that might be installed on the system.
`/proc`	The `/proc` directory is special. It's not a real file system in the sense of files stored on the hard drive. Rather, it is a virtual file system maintained by the Linux kernel. The "files" it contains are peepholes into the kernel itself. The files are readable and will give us a picture of how the kernel sees the computer.
`/root`	This is the home directory for the root account.
`/sbin`	This directory contains "system" binaries. These are programs that perform vital system tasks that are generally reserved for the superuser.
`/tmp`	The `/tmp` directory is intended for the storage of temporary, transient files created by various programs. Some configurations cause this directory to be emptied each time the system is rebooted.
`/usr`	The `/usr` directory tree is likely the largest one on a Linux system. It contains all the programs and support files used by regular users.
`/usr/bin`	`/usr/bin` contains the executable programs installed by the Linux distribution. It is not uncommon for this directory to hold thousands of programs.
`/usr/lib`	The shared libraries for the programs in `/usr/bin`.
`/usr/local`	The `/usr/local` tree is where programs that are not included with the distribution but are intended for system-wide use are installed. Programs compiled from source code are normally installed in `/usr/local/bin`. On a newly installed Linux system, this tree exists, but it will be empty until the system administrator puts something in it.
`/usr/sbin`	Contains more system administration programs.
`/usr/share`	`/usr/share` contains all the shared data used by programs in `/usr/bin`. This includes things such as default configuration files, icons, screen backgrounds, sound files, etc.

Directory	Comments
`/usr/share/doc`	Most packages installed on the system will include some kind of documentation. In `/usr/share/doc`, we will find documentation files organized by package.
`/var`	With the exception of `/tmp` and `/home`, the directories we have looked at so far remain relatively static, that is, their contents don't change. The `/var` directory tree is where data that is likely to change is stored. Various databases, spool files, user mail, etc. are located here.
`/var/log`	`/var/log` contains *log files*, records of various system activity. These are important and should be monitored from time to time. The most useful ones are `/var/log/messages` and `/var/log/syslog`. Note that for security reasons on some systems only the superuser may view log files.

Vediamo un dettaglio:

```
[me@linuxbox ~]$ ls -l
lrwxrwxrwx 1 root root 11 2007-08-11 07:34 libc.so.6 -> libc-2.6.so
```

quella "l" all'inizio significa che abbiamo di fronte un link simbolico (*symlink*), con il link chiamato *libc.so.6* che punta alla libreria condivisa *libc.2.6.so*.

MANIPOLAZIONE DEI FILE

Prima di tutto introduciamo una feature della shell essenziale: le **wildcards**, ovvero dei caratteri speciali che si usano per specificare l'uso di più file in vase ad una certa richiesta:

Wildcard	Meaning
*	Matches any characters
?	Matches any single character
[*characters*]	Matches any character that is a member of the set *characters*
[!*characters*]	Matches any character that is not a member of the set *characters*
[[:*class*:]]	Matches any character that is a member of the specified *class*

con (si usano anche le solite regole delle regex come [a-z] per indicare un carattere da a e z (minuscoli), [a-z]*, per indicare n caratteri tra a e z (minuscoli) etc...):

Character Class	Meaning
[:alnum:]	Matches any alphanumeric character
[:alpha:]	Matches any alphabetic character
[:digit:]	Matches any numeral
[:lower:]	Matches any lowercase letter
[:upper:]	Matches any uppercase letter

Pattern	Matches
*	All files
g*	Any file beginning with “g”
b*.txt	Any file beginning with “b” followed by any characters and ending with “.txt”
Data???	Any file beginning with “Data” followed by exactly three characters
[abc]*	Any file beginning with either an “a”, a “b”, or a “c”
BACKUP.[0-9][0-9][0-9]	Any file beginning with “BACKUP.” followed by exactly three numerals
[[:upper:]]*	Any file beginning with an uppercase letter
[![:digit:]]*	Any file not beginning with a numeral
*[[:lower:]123]	Any file ending with a lowercase letter or the numerals “1”, “2”, or “3”

Per creare una directory usiamo *mkdir* seguito dal nome della cartella, o di più cartelle:

```
[me@linuxbox ~]$ ls
dir1 dir2
[me@linuxbox ~]$ mkdir dir3 dir4
[me@linuxbox ~]$ ls
dir1 dir2 dir3 dir4
```

Per copiare si ha il comando *cp*. Si può copiare il file/direcory in un altro/a con (da source a dest):

```
[me@linuxbox ~]$ cp source dest
```

si possono copiare più files mettendoli in sequenza (o usando una wildcard). Si hanno le seguenti opzioni:

Option	Long Option	Meaning
`-a`	`--archive`	Copy the files and directories and all of their attributes, including ownerships and permissions. Normally, copies take on the default attributes of the user performing the copy. We'll take a look at file permissions in Chapter 9 "Permissions."
`-i`	`--interactive`	Before overwriting an existing file, prompt the user for confirmation. **If this option is not specified, `cp` will silently (meaning there will be no warning) overwrite files.**
`-r`	`--recursive`	Recursively copy directories and their contents. This option (or the `-a` option) is required when copying directories.
`-u`	`--update`	When copying files from one directory to another, only copy files that either don't exist or are newer than the existing corresponding files, in the destination directory. This is useful when copying large numbers of files as it skips files that don't need to be copied.
`-v`	`--verbose`	Display informative messages as the copy is performed.

Vediamo qualche esempio:

Command	Results
`cp file1 file2`	Copy *file1* to *file2*. **If *file2* exists, it is overwritten with the contents of *file1*.** If *file2* does not exist, it is created.
`cp -i file1 file2`	Same as previous command, except that if *file2* exists, the user is prompted before it is overwritten.
`cp file1 file2 dir1`	Copy *file1* and *file2* into directory *dir1*. The directory *dir1* must already exist.
`cp dir1/* dir2`	Using a wildcard, copy all the files in *dir1* into *dir2*. The directory *dir2* must already exist.
`cp -r dir1 dir2`	Copy the contents of directory *dir1* to directory *dir2*. If directory *dir2* does not exist, it is created and, after the copy, will contain the same contents as directory *dir1*. If directory *dir2* does exist, then directory *dir1* (and its contents) will be copied into *dir2*.

Per spostare un file in un altro uso *mv*:

```
[me@linuxbox ~]$ mv source dist
```

Si hanno le seguenti opzioni:

Option	Long Option	Meaning
`-i`	`--interactive`	Before overwriting an existing file, prompt the

Vediamo qualche esempio:

		user for confirmation. **If this option is not specified, `mv` will silently overwrite files.**
`-u`	`--update`	When moving files from one directory to another, only move files that either don't exist, or are newer than the existing corresponding files in the destination directory.
`-v`	`--verbose`	Display informative messages as the move is performed.

Command	Results
`mv file1 file2`	Move *file1* to *file2*. **If *file2* exists, it is overwritten with the contents of *file1*.** If *file2* does not exist, it is created. **In either case, *file1* ceases to exist.**
`mv -i file1 file2`	Same as the previous command, except that if *file2* exists, the user is prompted before it is overwritten.
`mv file1 file2 dir1`	Move *file1* and *file2* into directory *dir1*. The dierctory *dir1* must already exist.
`mv dir1 dir2`	If directory *dir2* does not exist, create directory *dir2* and move the contents of directory *dir1* into *dir2* and delete directory *dir1*. If directory *dir2* does exist, move directory *dir1* (and its contents) into directory *dir2*.

Per eliminare file e direcories (con -r) uso *rm*:

```
[me@linuxbox ~]$ ls
file.txt file2.txt
[me@linuxbox ~]$ rm file2.txt
[me@linuxbox ~]$ ls
file.txt
```

Si hanno le seguenti opzioni:

Option	Long Option	Meaning
`-i`	`--interactive`	Before deleting an existing file, prompt the user for confirmation. **If this option is not specified, `rm` will silently delete files.**
`-r`	`--recursive`	Recursively delete directories. This means that if a directory being deleted has subdirectories, delete them too. To delete a directory, this option must be specified.
`-f`	`--force`	Ignore nonexistent files and do not prompt. This overrides the `--interactive` option.
`-v`	`--verbose`	Display informative messages as the deletion is performed.

Vediamo qualche esempio:

Command	Results
`rm file1`	Delete *file1* silently.
`rm -i file1`	Same as the previous command, except that the user is prompted for confirmation before the deletion is performed.
`rm -r file1 dir1`	Delete *file1* and *dir1* and its contents.
`rm -rf file1 dir1`	Same as the previous command, except that if either *file1* or *dir1* do not exist, `rm` will continue silently.

per creare link, sia hard links che symlinks, uso *ln*. Per un link normale:

```
[me@linuxbox ~] ln file link
```

Per un link simbolico:

```
[me@linuxbox ~] ln -s file link
```

COMANDI PER LA PRODUTTIVITÀ

I comandi possono essere built-in della shell, programmi, alias o funzioni della shell. Con *type* possono essere identificati:

```
[me@linuxbox ~]$ type type
type is a shell builtin
[me@linuxbox ~]$ type ls
ls is aliased to `ls --color=tty'
[me@linuxbox ~]$ type cp
cp is /bin/cp
[me@linuxbox ~]$ type emacs
emacs is /usr/bin/emacs
```

Per determinare dove si trova un eseguibile (solo un eseguibile non un alias o altro) uso *which*:

```
[me@linuxbox ~] which ls
/bin/ls
```

Ci sono vari modi per ottenere per ottenere la documentazione di un comando.

Bash ha il comando built-in *help*:

```
[me@linuxbox ~]$ help cd
cd: cd [-L|[-P [-e]] [-@]] [dir]
    Change the shell working directory.

        Change the current directory to DIR.
    value of the HOME shell variable.

    ...
    Options:
       -L force symbolic links to be followed: resolve symbolic
       links in DIR after processing instances of `..'
       ...
```

Molti eseguibili hanno *--help*:

```
[me@linuxbox ~]$ mkdir --help
Usage: mkdir [OPTION] DIRECTORY...
Create the DIRECTORY(ies), if they do not already exist.
   -Z, --context=CONTEXT (SELinux) set security context to CONTEXT
   ...
```

Quasi tutti hanno la *man page*, richiamabile con *man*:

```
[me@linuxbox ~]$ man ls
```

Una man page è divisa in 8 sezioni:

1. User commands
2. Programming interfaces for kernel system calls
3. Programming interfaces to the C library
4. Special files such as device nodes and drivers
5. File formats
6. Games and amusements such as screen savers

7. Miscellaneous
8. System administration commands

richiamabili, per esempio per il *file formats* di passwd si ha:

```
[me@linuxbox ~]$ man 5 passwd
```

Con *apropos* si cerca una parola chiave nei vari man, ottenendo i comandi più consoni, tra parenetsi è indicata la sezione:

```
[me@linuxbox ~]$ apropos partiton
addpart (8)   - simple wrapper around the "add partition"...
all-swaps (7) - event signalling that all swap partitions...
cfdisk (8)    - display or manipulate disk partition table
cgdisk (8)    - Curses-based GUID partition table (GPT)...
delpart (8)   - simple wrapper around the "del partition"...
fdisk (8)     - manipulate disk partition table
fixparts (8)  - MBR partition table repair utility
...
```

Esiste anche *whatis* che mostra una linea di descrizione:

```
[me@linuxbox ~]$ whatis ls
ls        (1) - list directory contents
```

GNU Project mette ancge a disposizionje *info* che mostra le informazioni in un reader come quello di *less*. Si hanno le seguenti opzioni:

Command	Action
?	Display command help
PgUp or Backspace	Display previous page
PgDn or Space	Display next page
n	Next - Display the next node
p	Previous - Display the previous node
u	Up - Display the parent node of the currently displayed node, usually a menu
Enter	Follow the hyperlink at the cursor location

Si possono mettere più comandi in sequenza con ";":

```
[me@linuxbox ~]$ cd /usr; ls; cd -
bin games include lib local sbin
/home/me
[me@linuxbox ~]$
```

Per creare alias si ha:

```
[me@linuxbox ~]$ alias foo='cd /usr; ls; cd -'
```

che può essere aggiunto al *.bashrc* o al file di config della propria shell (*.zshrc* etc...).

Per rimuovere un alias uso:

```
[me@linuxbox ~]$ unalias foo
[me@linuxbox ~]$ type foo
bash: type: foo: not found
```

Con *alias* elenco tutti i vari aliases:

```
[me@linuxbox ~] alias
alias ls='ls --color=tty'
...
```

REDIREZIONAMENTO

pagina 78