
LINUX - *** KIT DI MINIMA SOPRAVVIVENZA *** - UNIX

- v0.05 -

Created by GT, 28/10/2002
l.r. by CG, 14/11/2005

1 - INFORMAZIONE SUI FILES

ls : lista i files presenti nel direttorio dove ci si trova, in ordine alfabetico

Sono anche possibili diverse opzioni:

ls -a : lista tutti i files, anche quelli di solito nascosti: .filename
ls -l : da' piu' informazioni sui files, come data di creazione ecc... (riga per riga)
ls -t : lista i files in ordine cronologico, prima i vecchi
ls -r : lista i files in ordine inverso a quello normale

le opzioni si possono sovrapporre; per es.

ls -ltr : lista i files in ordine cronologico inverso, dando piu' informazioni.

Il carattere '*' e' detto 'wildcard' e significa 'qualsiasi numero di caratteri qualsiasi'.

Il carattere '?' e' wildcard per un carattere e significa 'un carattere qualsiasi'.

ES: ls a* : lista tutti i files che iniziano per 'a'
ls a??? : lista tutti i files di quattro lettere, che iniziano per 'a'

Le wildcards * e ? si possono usare nella maggior parte dei comandi di informazione, operazione e manipolazione di files. Ma con attenzione!

more myfile : mostra il contenuto del file di testo myfile, una pagina alla volta.
Durante l'esecuzione ci si muove nel file come segue:

barra spaziatrice : mostra la pagina successiva;
il tasto enter : fa scorrere il file una riga alla volta
b : (lettera 'b') mostra la pagina precedente;
q : (lettera 'q') esce dal file e fa tornare alla linea dei comandi.

ES: more *.f90
more pippo.dat

cat myfile : mostra il contenuto del file di testo myfile, tutto insieme. Se il file e' piu' lungo di una pagina, lo si vedra' scorrere tutto fino alla fine e non si riuscirà a leggere niente.

head myfile : mostra le prime 10 righe di myfile
head -n file : mostra le prime n righe di un file

ES: head -1 *.dat

tail myfile : mostra le ultime 10 righe di myfile
tail -n myfile : mostra le ultime n righe di myfile

NOTA: tail NON accetta le wildcards * e ? (su sistemi UNIX)

wc myfile : conta linee, parole e caratteri contenuti in myfile

ES: wc pippo.dat
wc *.dat
wc *.f90

2 - DOCUMENTAZIONE DEI COMANDI

man xxx : mostra la pagina di 'help' (man = manual) del comando xxx, con istruzioni sull'uso e sulle opzioni possibili (per es. -f, -i, -l ecc...)

apropos yyy : cerca la stringa 'yyy' nelle pagine di manuale di tutti i comandi Unix. Utile per trovare il nome esatto di un comando che compie l'azione yyy.

whatis zzz : descrive la funzione del comando zzz.

3 - OPERAZIONI SU FILES E DIRETTORI

cp file1 file2 : fa una copia del file file1 e la chiama file2

mv file1 file2 : rinomina il file da file1 a file2. Piu' in generale si usa per spostare i files da un direttorio all'altro.

ES: mv test.f90 Codes/test.f90 : sposta il file test.f90 nel sottodirettorio Codes/
mv test.f90 Codes/test2.f90 : sposta il file test.f90 nel sottodirettorio Codes/
e lo rinomina test2.f90

rm myfile : rimuove myfile dal disco. NOTA: non lo 'svuota', lo cancella fisicamente, e myfile va perduto.

`rmdir mydirectory` : rimuove il direttorio `mydirectory` dal disco. NOTA: `mydirectory` deve essere VUOTO per essere cancellato.

`rm -fr mydirectory` : rimuove il direttorio `mydirectory` dal disco, (forzatamente e ricorsivamente) non e' necessario che il direttorio sia vuoto.

4 - MUOVERSI NEI DIRETTORI

definizioni:

`~` e' il nome della home directory di ciascun user.

`./` e' il nome del direttorio in cui si e'.

`../` e' il nome del direttorio sopra a quello in cui si e'.

`cd` : (current directory) colloca l'utente nel suo direttorio principale, detto 'home directory'. Di solito il nome la home directory finisce con il nome dell'utente.

`cd mydir` : colloca l'utente nel direttorio `mydir`

ES: `cd ../`
`cd ~`
`cd ~/Codes`
`cd Codes/LastVersion`

`mkdir mydir` : crea un direttorio `mydir` dentro il direttorio corrente.

`pwd` : restituisce il nome del direttorio corrente (percorso).

`ln -s long_path short_path` : link simbolico tra direttori o files

questo comando serve per accedere a direttori o files "lontani" senza dover digitare nomi troppo lunghi. Se dovete accedere al direttorio o file `long_path` (che sara' un nome molto lungo) potete creare un file `short_path` (che sara' un nome piu' corto) che fa le veci di `long_path`.

Per esempio:

`ln -s /nfs/picasso/data1/bepi/g32/GAS6_REHEAT/post/Images Images`

fara' si' che 'cd Images' o 'ls Images' o ogni altro comando effettuato su 'Images' si riferisca al sottodirettorio in `long_path`.

5 - MANIPOLAZIONE DI FILES E DEL LORO CONTENUTO

grep mystring file1 file2 ... : cerca la stringa mystring in tutti i files file1, file2...,
e restituisce a schermo le linee dei files contenenti mystring

grep -n mystring file1 : cerca la stringa mystring e restituisce a schermo linee del file
contenenti mystring con il numero della riga corrispondente

ES: grep PRINT test.f90
grep REAL* *.f90

find . -name myfile -print : cerca tutti i files di nome myfile nel direttorio corrente e in
tutti i suoi sottodirettori, restituendo la posizione dei files
trovati.

ES: find . -name '*.f90' -print
find . -name '*.*?' -print

6 - MANIPOLAZIONE DI COMANDI E GESTIONE DI PROCESSI

> : ridirige in un file NUOVO l'output che per 'default' verrebbe mostrato sullo
schermo.

ES: ls > mylist : crea il file mylist che contiene la lista dei files presenti nel
direttorio corrente.

>> : (append) ridirige in un file ESISTENTE l'output dello schermo, aggiungendolo
alla fine del file.

ES: ls a* > all_files
ls b* >> all_files

| : (pipe) collega piu' comandi tra loro

ES: grep PRINT *.f90 | wc : conta quanti 'PRINT' ci sono in tutti i files *.f90
ls -l *.f90 | wc : conta quanti file con estensione .f90 ci sono nel direttorio
in cui ci si trova

ps : mostra i processi (le azioni) esistenti nel computer.

ps ux : mostra tutti i miei processi

ps aux : mostra tutti i processi di tutti

Data una finestra dentro cui si lavora, ci sono diversi tipi di processi:

- processi in foreground: che bloccano la finestra e impediscono altre azioni;
- processi in background: che permettono altre azioni dentro la finestra.

Un processo a sua volta puo' essere attivo (Running) o inattivo (Suspended)

Ctrl c : (premuti insieme) uccide il processo attivo nella finestra
(cioe' in foreground)

Ctrl z : sospende il processo attivo nella finestra

Ctrl s : sospende la scrittura sullo schermo

Ctrl q : ripristina la scrittura a schermo sospesa con Ctrl s

Ctrl d : equivalente a 'logout', serve a chiudere la sessione

logout : equivalente a 'Ctrl d', chiude la sessione della finestra in cui si da'
il comando.

jobs : mostra i processi in background e il loro stato (Running o Suspended)

kill -9 PID : uccide il processo con identita' no. PID (process identity). Il PID e' un numero
che identifica univocamente un processo. Per sapere il PID di un processo si usa
il comando 'ps ux'

fg : rimette in foreground un processo (sospeso o in background)

bg : mette in background un processo sospeso

Per mettere in bg un processo (che puo' essere un comando o una combinazione di comandi),
direttamente senza prima sospenderlo con Ctrl Z, si scrive

nome_comando &

Ad esempio, per lanciare il programma mytest, metterlo in bg e ridirigere l'output dallo
schermo al file out.log, si scrive:

mytest > out.log &

7 - DISPLAY

Se da un computer xxx aprite una finestra (per es. con telnet) sul computer yyy, e poi volete
aprire una finestra grafica (tipi SM o xemacs o IDL o altro) occorre assicurarsi che valgano
due condizioni:

1. xxx deve dare il permesso ad yyy di aprire una finestra grafica
2. yyy deve sapere dove aprirla (cioe' su xxx).

Questo si fa con due comandi:

1. (su xxx): xhost yyy
- 2a. (su xxx): chiedo a xxx come si chiama il terminale su cui lavoro: echo \$DISPLAY
- 2b. (su yyy): setenv DISPLAY <nome del terminale>

Nella pratica: sono collegato su matisse.astro.it e da una finestra
faccio telnet picasso. Voglio aprire SM da picasso.

1. (su matisse): xhost picasso.pd.astro.it
- 2a. (su matisse): echo \\${DISPLAY} ; compare per es. pcdot8:0.0
- 2b. (su picasso): setenv DISPLAY pcdot8:0.0

8 - COMANDI VARI

who : mostra gli utenti collegati in questo momento

w : come who, ma fornisce piu' informazioni sul sistema

whoami : mostra il nome dell'utente

clear(Ctrl l) : pulisce lo schermo e riposiziona il cursore in cima

date : mostra data ed ora

passwd : permette di cambiare la password

mount : monta un dispositivo (hard-disk, usb-key, floppy, cd-rom ...)

diff file1 file2 : trova le differenze tra due files
