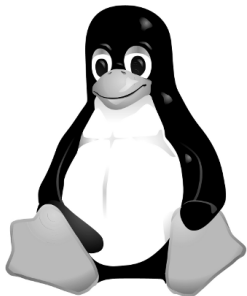


GNU/Linux a Linea di Comando con Bash



Ing. Nicolò Nepote

www.nokonline.it

Maggio 2009

Premesse

- In queste dispense sono esposti concetti che fanno riferimento all'architettura dei kernel di UNIX®¹ e GNU/Linux. Questi concetti sono generalizzabili verso qualsiasi sistema operativo derivato da UNIX e verso qualsiasi distribuzione Linux.
- Una copia stampabile di queste slide², con il relativo codice L^AT_EX, è disponibile online presso la URL:
<http://www.nokonline.it/item/2006/01/made-in-noko>.
- Queste slide usano una versione modificata del tema *Torino* per Beamer-L^AT_EX.
- Last update June 5, 2009.

¹Copyright The Open Group

²Distribuite sotto GNU Documentation License (www.gnu.org/licenses/gdl.txt)



Agenda

1. Login e Logout
2. Una, dieci, cento shell
3. Bash
 - 3.1 Caratteristiche
 - 3.2 Comandi principali
 - 3.3 Job control
 - 3.4 Process control
4. Esercizi



Login

- Il login avviene fornendo la coppia username/password:

```
Fedora Core release 9 (Sulphur)
kernel 2.6.25-3.18.fc9.ppc on a ppc (tty2)
spitfire Login:
Password:
Last login: Thu Mar 24 10:36:06 on :0
[~]noko@spitfire$
```

- E' possibile diventare chiunque in ogni momento col comando **su**:

```
[~]noko@spitfire$ su ( "-" per leggere i file di configurazione)
Password:
[/home/noko]root@spitfire#
```

File correlati:

```
/etc/nsswitch.conf /etc/passwd /etc/shadow /etc/security
```



Logout

- Il logout avviene tramite i comandi `logout` ed `exit` oppure fornendo un EOF tramite `ctrl-d`.
- E' possibile usare `logout` solo dalle shell "di login".

```
[~]noko@spitfire\$ logout  
bash: logout: not a login shell: use 'exit'
```
- lo spegnimento avviene col comando `shutdown`, ma solo da root

```
[~]noko@spitfire\$ /sbin/shutdown  
shutdown: you must be root to do that  
[~]noko@spitfire\$ su  
Password  
[/home/nepote]root@spitfire# shutdown -h now  
The system is going for shutdown NOW!
```
- Le distribuzioni hanno dei wrapper che permettono agli utenti normali di spegnere la macchina.



La Shell

- La shell è l'interfaccia tramite la quale l'utente **comunica col SO**.
- E' un processo assegnato al *console user* di turno.
- Ha il compito di eseguire comandi, ovvero attendere finchè il comando di un utente non richiede una `fork()`, eseguirla ed attendere la fine del comando.
- E' invocata da `/bin/login` al boot su una certa *tty*, oppure da un'altro qualsiasi processo; in questo caso la si definisce "non login shell".
- Il console user può interagire con una sola shell (sulla stessa *tty* o *pts*) alla volta.



Tipologie di Shell

- Per ragioni storiche esistono diversi tipi di shell tra cui scegliere:
 - **sh** La Bourne Shell, ovvero la predefinita di UNIX dal System V, datata 1978, ben scritta, ma non offre molte funzionalità.
 - **csch** C shell, sviluppata a Berkeley. In larga parte compatibile con la bourne ma con una sintassi di programmazione diversa.
 - **tcsh** Evoluzione della csh che offre il command line editing.
 - **ksh** Korn shell. Introduce alcune delle moderne tecniche, anche prese dalla tcsh. La più popolare nella storia di UNIX.
 - **bash** Bourne Again SHell. Come suggerisce l'acronimo è un software GNU. La più versatile, potente ed utilizzata, offre:
 - command line editing
 - scripting avanzato
 - history substitution
 - command completion
 - I/O redirection
 - piena compatibilità con sh



Bash

- **Variabili d'ambiente:**

```
noko@spitfire$ printenv
noko@spitfire$ echo $ENVVAR
noko@spitfire$ export ENVVAR=valore
```

- **History browsing.** Si ottiene usando i tasti freccia.

- **History substitution:**

```
noko@spitfire$ history (mostra la history)
noko@spitfire$ !! (riesegui l'ultimo comando)
noko@spitfire$ !n (esegui l'n-esimo comando della history)
noko@spitfire$ !-n (riesegui n comandi fa)
```

- **Sintassi generica dei comandi:**

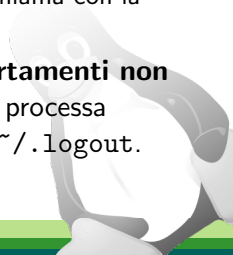
```
comando -opzione (o switch) valore -opzione2 valore argomento
```

- **Auto-completamento.** Disponibile per comandi e filesystem, si ottiene premendo una o due volte il tasto tab.



Bash (2)

- Variabili d'ambiente fondamentali:
 - **SHELL** L'eseguibile della shell utilizzata.
 - **USER** L'utente loggato. Cambia dopo un *su* solo con "-". Un *whoami* (o *who*, *w*) risolve quasi sempre le crisi d'identità.
 - **HOME** Path assoluto della home directory dell'utente
 - **PWD** *Print Working Directory*
 - **PATH** L'elenco delle directory dove la bash cerca i comandi dati senza specificarne il percorso completo. *whereis* (o *which*) per orientarsi.
 - **PS1** Definizione del formato del *prompt*.
 - **STATUS** Valore di ritorno dell'ultimo comando. Si richiama con la variabile \$?
- E' possibile specificare variabili, alias ed altri **comportamenti non standard** in alcuni file di configurazione, che la shell processa all'avvio: `/etc/profile` `~/.bashrc` `~/.profile` `~/.logout`.



Bash (3)

- Manuali ed aiuti
 - **man** Visualizza le pagine di manuale del comando in argomento.
 - - **-help** o **-h** sono switch di solito presenti che danno una panoramica delle opzioni più importanti del comando.
- La shell espande le espressioni regolari con tutto il contenuto che le verifica. Se non si vuole questo effetto occorre proteggere racchiudendo tutto in due apici ('').
 - **\$** Quello che segue è una variabile
 - **/** Separatore del path
 - **?** Un carattere qualsiasi
 - ***** Un numero qualsiasi di caratteri qualsiasi
 - **[]** Un carattere nella lista o nell'insieme([abcd]=[a-d])
 - **{ }** Una parola nell'elenco tra virgole
 - **' '** o **" "** Non espande l'espressione regolare
 - **' '** Il risultato dell'esecuzione del comando tra apici
 - **~user** Home dell'utente user. Se non specificato è l'utente corrente



Bash (4)

- **Pipe:** Trasformazione dell'output di un comando nell'input di quello successivo.

```
noko@spitfire$ ls -l /sbin | grep config
```

- **Redirezione dell'I/O:** È possibile redirigere l'output di un comando in un file qualsiasi. Utile per scrivere in file di testo da usare negli script o come log, per scrivere in device vari, oppure per sopprimere l'output scrivendo nel device speciale `/dev/null`.
- Normalmente la destinazione è la console corrente, sia per `std_output` sia per `std_error`:
 - comando `> file` (Redirige `std_out` in file)
 - comando `2> file` (Redirige `std_err` in file)
 - comando `1>fileA 2>fileB` (Redirige `std_out` in fileA e `std_err` in fileB)



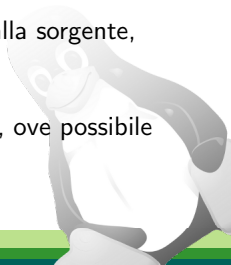
Comandi Principali

- **pwd** *print working directory*. Visualizza il path completo della directory di lavoro.
- **ls** *list*. Elenca il contenuto della directory corrente o di quella in argomento. (spesso presente un alias "ll")
 - **-a** *all*. Anche i file nascosti
 - **-l** *long*. Formato lungo (dettagli)
- **cd** *change directory*. Si sposta nella directory in argomento.
- **mkdir** *make directory*. Crea una directory.
 - **-p** *parents*. Crea le directory padre se non esistono.
- **rmdir** *remove directory*. Elimina una directory se è vuota.



Comandi Principali (2)

- **touch** *crea* un file vuoto senza tipo. Se esiste già ne aggiorna il *timestamp*.
- **rm** *remove*. Elimina un oggetto del filesystem.
 - **-r** *recursive*. Elimina tutto recursivamente.
 - **-f** *force*. Forza l'eliminazione sugli errori e sulle richieste di conferma.
NB: `rm -rf /dir` elimina la directory anche se non è vuota!
- **cp** *copy*. Copia un file dato come primo argomento nella directory data come secondo. Se viene specificato un file come destinazione, il file sorgente viene copiato con quel nome.
 - **-r** *recursive*. Copia tutti i file e le directory a partire dalla sorgente, che deve essere una directory.
 - **-f** *force*. Forza la sovrascrittura.
 - **-preserve** *preserva*. Lo fa per i permessi e le proprietà, ove possibile (**Es.** no FAT32)
 - **-archive** *archivio*. Usato per i backup/restore.



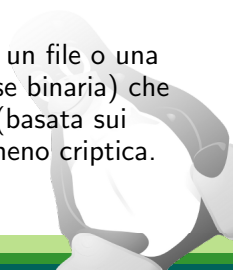
Comandi Principali (3)

- **mv** *move*. Sposta un file o una directory dati come primo argomento nella directory data come secondo. **NB:** per rinominare un file è sufficiente spostarlo in uno nuovo col nome prescelto. Funziona anche con le directory, a patto che non esista già una directory dello stesso livello con quel nome: in questo caso la directory sorgente vi viene spostata.
 - **-f** *force*. Forza la sovrascrittura.
- **du** *disk usage*. Mostra la dimensione di tutti i file nella directory corrente o di un oggetto in argomento, con un formato "scoraggiante".
 - **-s** *sum*. Mostra la dimensione totale di tutta la directory, recursivamente.
 - **-k** *KiloBytes*. Dimensione in KB. Uno standard.
 - **-h** *human*. Sceglie l'unità di misura migliore per ogni file.



Comandi Principali (4)

- **In** *link*. Crea un collegamento al primo argomento col nome del secondo. Spesso gli argomenti sono invertiti (Es. Solaris).
 - **-f** *force*. Forza la sovrascrittura.
 - **-s** *soft*. Soft link invece che Hard.
- **chown** *change owner*. Cambia il proprietario di un file o una directory. Accetta come parametro uno username o una coppia *username:gruppo*.
- **chgrp** *change group*. Cambia il gruppo associato a un file o una directory.
- **chmod** *change mode*. Cambia i permessi associati a un file o una directory. Accetta sintassi numerica (basata sulla base binaria) che ridefinisce sempre l'intero modo e sintassi simbolica (basata sui caratteri u, g, o, r, w, x, +, -) più macchinosa, ma meno criptica.
 - **-R** *recursive*. Anche per i precedenti chgrp e chmod.

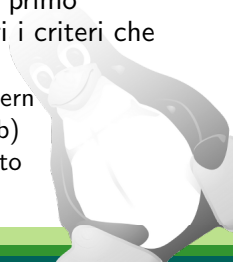


Comandi Principali (5)

- **echo** Stampa a video una stringa
 - **-n** *no newline*. Non va a capo alla fine.
 - **-e** *enable*. Abilita i caratteri di escape:
 - **n** *newline*
 - **t** *tab*
 - **b** *bell*

utile per scrivere velocemente su file; essenziale per conoscere lo stato delle variabili

- **cat** *conCATenate*. Stampa il contenuto dei file dati in argomento.
- **find** *trova*. Cerca a partire dalla directory data come primo argomento, ogni oggetto del filesystem che rende veri i criteri che seguono:
 - **-name** vero se il nome dell'oggetto corrisponde al pattern
 - **-type** vero se il tipo corrisponde al simbolo (f, d, l, c, b)
 - **-user** vero se il proprietario corrisponde a quello indicato
 - **-exec** esegue un comando per ogni oggetto trovato.



Comandi Principali (6)

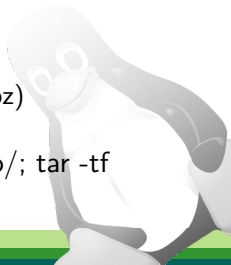
- **more** Si tratta di un pager. Usato soprattutto come filtro. Permette di vedere uno stream output in spezzoni di una pagina.
 - **spazio** avanza di una pagina
 - **enter** avanza di una riga
- **tail, head** Un'altra coppia di usatissimi pager. Funzionano bene come filtri. Visualizzano le prime o le ultime dieci righe.
 - **-numero** le ultime *numero* righe
- **wc** *word count*. Conta i caratteri, le parole e le righe contenute nel testo in input.
 - **-l** *lines*. stampa il numero righe



Comandi Principali (7)

- **sort** Ordina le righe di testo in input in ordine alfabetico o specifico:
 - **-b** *blank*. Ignora gli spazi ad inizio riga.
 - **-f** ignora le maiuscole.
 - **-M** *month*. Ordina per mese.
 - **-n** *number*. Ordine numerico.
 - **-r** *reverse*. Ribalta l'ordinamento.
- **tar** tape archiver. Crea un pacchetto tar.
 - **-f** *file*. Usa un file (in argomento) come tape device.
 - **-c** *create*. Crea un archivio.
 - **-x** *extract*. estrai i file da un archivio.
 - **-t** *list*. elenca i file contenuti nell'archivio
 - **-z** **-j** *gzip, bzip2*. L'archivio è compresso (.tar.gz, .tar.bz)
 - **-p** *preserve permissions*.

Es. tar -czf disney.tar.gz pippo.txt pluto.log paperino/; tar -tf disney.tar.gz



Comandi Principali (8)

- **gzip** *gnu zip*. Comprime o decomprime (comando gunzip) un file.
- **grep** *gnu grep*. Usato quasi esclusivamente come filtro. Stampa a video le linee di input che contengono la stringa in argomento. Accetta espressioni regolari.
 - **-n** *number*. Stampa il numero della linea.
 - **-v** *reVerse*. Criterio opposto.
- **cut** *taglia* il testo in verticale secondo le modalità desiderate. Restituisce una colonna.
 - **-d<char>** *delimiter*. Usa <char> come delimitatore invece che Blank.
 - **-f<num>** *field*. Restituisce la colonna (o campo) <num>.
 - **-c<num>** *character*. Restituisce il <num>-esimo carattere di ogni colonna.

Comandi Principali (9)

- **sed** *gnu stream editor*. Compie svariate operazioni sulle stringhe in input dove viene soddisfatta un'espressione regolare o, più precisamente, un vero e proprio script "in sed". Comando dalla sintassi criptica usato principalmente dagli script per modificare i file di configurazione.

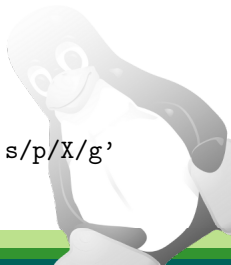
- **-e '<script>'** *expression*. L'espressione da processare è lo <script> seguente.

Esempi di regole di scripting:

```
{  
s/<regexp>/<string>/  
a <text> .  
}
```

Uno script-in-command usatissimo:

```
noko@spitfire$ echo "pippo pluto" | sed -e 's/p/X/g'  
XiXXo Xluto
```



Job Control

- **&** Operatore che esegue il comando che lo precede in *background*. Restituisce la posizione del processo “forkato” nella lista dei job associati alla presente shell ed il suo PID. Restituisce anche il prompt; il che è essenziale.
- **jobs** Mostra la lista dei job con il loro stato. (Running, ecc.)
- **fg** *foreground*. Manda in primo piano il processo in argomento. Accetta come argomento solo la posizione nella lista dei job.
NB: con ctrl-z è possibile stoppare il processo attualmente in foreground e rimetterlo successivamente in esecuzione in background con il comando bg.



Process Control

- **ps** *process status*. Fornisce la lista dei processi associati all'utente.
 - **a** *all*. Tutti i processi.
 - **u** *user*. Mostra il nome vero dell'utente proprietario del processo.
 - **x** mostra la tty da cui è stato lanciato ogni processo. **NB**: **ps aux** è il comando standard per vedere lo stato del SO.
- **kill** Manda un segnale ad un processo in argomento, se non specificato il segnale SIGTERM. Usato principalmente per uccidere processi.
 - **-l** *list*. Elenca i possibili segnali
 - **-9** SIGKILL Terminazione non gestibile dal processo.
 - **-10** SIGUSR1 Segnale inviabile dall'utente
 - **-14** SIGALRM Allarme che ogni processo gestisce diversamente.
 - **-15** SIGTERM Terminazione semplice.
 - **-19** SIGSTOP Segnale di stop.



Process Control (2)

- **top** *time of processes*. Mostra l'evoluzione dei task di Linux, ovvero l'attuale lista elaborata dallo scheduler. Una fotografia perfetta del funzionamento del sistema ad intervalli di tempo.
 - **-d** **<num>** *delay*. Aggiorna la lista ogni **<num>** secondi



Esercizi

1. Visualizzare una lista di tutti i file di nome “core” ovunque nel filesystem e di cui si ha accesso in lettura come normale utente.
2. Visualizzare la lista dei PID di tutti i processi nel sistema associati al vostro utente.
3. Visualizzare una lista di filename, uno per linea, contenuti nella directory /dev che abbiano una forma del tipo “sda” seguito da un numero. Ordinare la lista in modo ascendente.
4. Visualizzare il contenuto del file /etc/inittab senza i commenti.
5. Visualizzare le linee dell’output del comando `dmesg` che contengono:
 - 5.1 la stringa “CPU” : cosa potete dire del microprocessore installato?
 - 5.2 la stringa “hda”, “hdb”, “hdc” oppure “hdd” : Quali dispositivi IDE sono installati?