Informatica Generale 1 - Esercitazioni
Introduzione all'uso della command-line shell

Daniele Pighin
pighin@fbk.eu

FBK
Via Sommarive, 18 I-38050 Trento, Italy

March 5, 2008

Outline

OS, software di base, applicazioni

Composizione di un sistema informativo:

1. hardware (CPU, periferiche, rete di calcolatori);
2. software:
   1. **Sistema operativo**: interfaccia HW/SW
      - kernel, moduli, system call ...
   2. **Software di base**: utilities e servizi per applicazioni
      - shell, compilatori, librerie ...
   3. **Software applicativo**: programmi end-user
      - servizi (web server), produttivitá (word processor) ...

La shell interattiva

- La shell é l'interfaccia di un sistema operativo;
- La shell permette all'utente di:
  - lanciare programmi;
  - interagire con i programmi;
  - visualizzarne i risultati.
- Generalmente, ogni OS offre
  - una shell grafica (piú semplice),
  - una shell a riga di comando (meno intuitiva ma piú potente, necessaria per lanciare processi batch).

Noi lavoreremo con la shell Bash dei moderni sistemi Unix-like (Linux, BSD, MacOSX ...).
Nota bene: Bash é estremamente potente e flessibile. Per approfondire: `http://tldp.org/LDP/abs/html/`.

Utilizzo di un'interfaccia a riga di comando

La shell si presenta come un prompt (ad es. $) seguito da un cursore.
Per invocare un programma:

1. si scrive il nome del programma;
2. si scrivono gli eventuali argomenti o parametri del comando;
3. si preme il tasto invio (enter).

Esempio

Invocazione del programma **date**:

```
$ date <invio>
```

N.B.: il carattere $ non va scritto! Indica il prompt della shell.

Esempio (importante!)

Invocazione del programma **man** (con argomento):

```
$ man date <invio>
```

man visualizza il manuale di un programma, usatelo spesso!
Anche **man** ha un manuale:

```
$ man man <invio>
```

Processi e standard streams

- Un programma puó essere costituito da piú di un flusso di esecuzione (processo);
- molti programmi semplici coincidono con un processo (es: **date**);
- programmi complessi possono essere costituiti da centinaia di processi (es: un sistema operativo);
- ad ogni processo sono sempre associati almeno 3 flussi di dati:
  - **0** standard input (**stdin**), da cui il programma legge l'input dell'utente;
  - **1** standard output (**stdout**), usato per richiedere input o fornire risultati;
  - **2** standard error (**stderr**), su cui il programma scrive messaggi di errore.

  Inoltre, ogni processo puó accedere ad un numero arbitrario di file di input/output.

Byte, file e filesystem

- Un file é una sequenza ordinata di byte memorizzata su qualche supporto;
- l'interpretazione della sequenza dipende dall'utente o dal programma che ha creato il file;
- i byte che compongono un file possono non essere memorizzati sequenzialmente sul supporto (frammentazione);
- il supporto contiene una mappa che consente di associare a ciascun file i frammenti che lo compongono;
- il sistema operativo gestisce l'interazione tra il SW e le periferiche di memorizzazione;
- il filesystem é un'astrazione logica fornita dal SO che consente di:
  - associare nomi ai file;
  - organizzare i file in contenitori logici (directory).

Il filesystem Unix

- Il filesystem di Unix é organizzato come un albero;
- tutte le directory discendono da un'unica radice (root): /;
- lo stesso carattere é utilizzato per separare i livelli della gerarchia;
- un file **nel sistema** é identificato univocamente dal suo **percorso assoluto** (absolute path), ad es: `/home/daniele/pippo.txt`;
- un file **in una directory** é identificato univocamente dal suo **percorso relativo** (relative path), ad es: `pippo.txt`;
- Unix ha due nomi di directory speciali:
  - `.` indica la directory corrente
    (es: /home/daniele/./ ≡ /home/daniele/);
  - `..` indica la parent directory
    (es: /home/daniele/../ ≡ /home/);
- la directory / é speciale perché non ha parent directory (piú precisamente, /../ coincide con /).

Navigazione del filesystem Unix

Principali comandi (N.B.: `man <comando>` per istruzioni!):

- `pwd` visualizza directory corrente;
- `cd [dirname]` *dirname* diventa la directory corrente. Se l'argomento viene omesso, la **home** dell'utente diventa la directory corrente;
- `ls [dirname]` visualizza il contenuto di una directory. Se l'argomento é omesso, visualizza il contenuto della directory corrente;

File nascosti

In Unix tutti i file/directory il cui nome inzia con un punto (.) sono nascosti. Per elencarli, bisogna aggiungere il flag `-a` al comando `ls`. Es:

```
$ ls -a /home/daniele
```

visualizza il contenuto della directory `/home/daniele`, compresi gli eventuali file nascosti.

N.B.: le directory speciali `.` e `..` sono nascoste!

Manipolazione del filesystem Unix

Principali comandi (N.B.: `man <comando>` per istruzioni!):

- `mkdir <dirname>` crea una directory;
- `rmdir <dirname>` elimina una directory (solo se vuota);
- `touch <filename>` crea un file (se non esistente);
- `cp <source> <dest>` copia il **file** *source* in *dest*;
- `cp -r <source> <dest>` copia la **directory** *source* in *dest*;
- `mv <oldname> <newname>` rinomina/sposta file e directory;
- `rm <filename>` elimina un file;
- `rm -r <dirname>` elimina ricorsivamente una directory e il suo contenuto.

Utenti e permessi

Sistema multiutente

- Unix é un vero sistema multiutente;
- il superuser (root) ha completo accesso al filesystem, puó creare nuovi utenti ed organizzarli in gruppi di utenza;
- ciascun utente appartiene ad uno o piú gruppi;
- ciascun utente/gruppo ha un determinato insieme di privilegi;
- i normali utenti possono accedere solo ad alcune risorse e hanno pieno controllo all'interno della propria home directory;
- il proprietario di un file puó assegnarlo ad un gruppo ed impostarne i privilegi di accesso in base a:
  - 3 classi di utenza (**u**ser (owner), **g**roup, **o**thers);
  - tre modalitá di accesso (**r**ead, **w**rite, e**x**ecute).

Ad esempio, il proprietario del file pippo.txt puó decidere che solo lui (owner) e i membri del gruppo disney (group) possono modificare il file (read, write), mentre gli altri utenti (other) possono solo visualizzarne il contenuto (read).

Utenti e permessi (cont.)

Interpretazione dei permessi

La semantica delle modalitá di accesso é diversa per file e directory.

- Per un file:
  - **r** determina se é possibile visualizzarne il contenuto;
  - **w** determina se é possibile modificarne il contenuto;
  - **x** determina se puo' essere eseguito (ha senso se il file contiene codice il codice eseguibile di qualche programma).
- Per una directory:
  - **r** determina se é possibile visualizzarne il contenuto, cioé fare `ls` nella directory;
  - **w** determina se é possibile creare/rimuovere file;
  - **x** determina se é possibile fare `cd` nella directory, cioé usarla come directory corrente.

Gestione dei permessi

Informazioni sui permessi

- `whoami` visualizza lo username dell'utente;
- `groups` elenca i gruppi di cui l'utente fa parte;
- `ls -l [name]` visualizza informazioni dettagliate sul file/directory indicato oppure della directory corrente (se omesso).

Gestione dei permessi (cont.)

Un esempio

```
$ ls -l /home/daniele/slides.tex
-rw-r----- 1 daniele daniele 10895 2008-03-04 13:00 slides.tex
(permessi)(n)(owner) (group) (data ultima modifica) (filename)
```

Stringa dei permessi, interpretazione

- primo carattere: tipo di file (**d**: directory, **-** file, **l** link simbolico);
- caratteri 2-4: permessi del proprietario;
- caratteri 5-7: permessi del gruppo;
- caratteri 8-10: permessi altri utenti.

Nell'esempio, il proprietario puo' leggere e modificare il file (`rw-`), i membri del gruppo possono solo leggerlo (`r--`) e gli altri utenti non possono farci nulla (`---`).

Gestione dei permessi (cont.)

Modifica dei permesi

N.B.: solo il propritario di un file (o il superuser) possono modificarne i permessi!

- `chgrp <group> <file>` assegna il file/dir ad un certo gruppo;
- `chown <user> <file>` cede il possesso del file ad un altro utente;
- `chmod <incremento> <file>` moodifica i permessi del file in base all'incremento.
- `chmod <assegnazione> <file>` imposta i permessi del file in base all'assegnazione.

Gestione dei permessi (cont.)

Sintassi dell'incremento (Backus-Naur Form, BNF)

```
<incremento> ::= <espressione>[,<incremento>]
<espressione> ::= <classe di utenza><operatore><modalitá>
<classe di utenza> ::= u|g|o|a[<classe di utenza>]
<operatore> ::= +|-
<modalitá> ::= r|w|x[<modalitá>]
```

Gestione dei permessi (cont.)

Significato dei simboli terminali

- classe di utenza:
  - **u** user, il proprietario del file;
  - **g** group, il gruppo a cui il file é assegnato;
  - **o** others, gli altri utenti;
  - **a** all, imposta simultaneamente i permessi per tutti gli utenti (é uguale a `ugo`);
- operatore:
  - **+** garantisci accesso;
  - **-** vieta accesso;
- modalitá:
  - **r** read, lettura;
  - **w** write, scrittura;
  - **x** execute, esecuzione.

Gestione dei permessi (cont.)

Alcuni esempi

- `chmod a+x <file>`
  consenti a tutti di eseguire il file;
- `chmod ugo+x <file>`
  stessa cosa;
- `chmod go-w <file>`
  non permettere agli utenti del gruppo e gli altri di scrivere sul file;
- `chmod ug+x,o-r <file>`
  consenti al proprietario e al gruppo lettura e scrittura sul file, priva gli altri del permesso di lettura;
- `chmod a-rwx,u+r <file>`
  assicurati che il proprietario sia l'unico a poter accedere al file, solo in lettura.

Gestione dei permessi (cont.)

Sintassi dell'assegnazione (Backus-Naur Form)

```
<assegnazione> ::= <valore><valore><valore>
<valore> ::= 0|1|2|3...7
```

Interpretazione

- L'assegnazione é un numero di 3 cifre, ciascuna cifra compresa tra 0 e 7, ad es. 600 o 766;
- la cifra piú significativa (sinistra) descrive i permessi del proprietario, quella di mezzo quelli del gruppo e la meno significativa quelli degli altri;
- a ciascuna modalitá di accesso viene associata una potenza del 2:

r	w	x
2^2	2^1	2^0

- per ciascuna classe di utenza, la cifra che si scrive é la somma delle potenze corrispondenti ai permessi che si vogliono attivare.

Gestione dei permessi (cont.)

Esempi

1. user: $\overset{r}{2^2} + \overset{w}{2^1} + \overset{x}{2^0} = 7$

   group: $\overset{r}{2^2} + \overset{\overline{w}}{0} + \overset{x}{2^0} = 5$

   other: $\overset{\overline{r}}{0} + \overset{\overline{w}}{0} + \overset{x}{2^0} = 1$

   $\Rightarrow$ 751

2. user: $\overset{r}{2^2} + \overset{w}{2^1} + \overset{\overline{x}}{0} = 6$

   group: $\overset{r}{2^2} + \overset{\overline{w}}{0} + \overset{\overline{x}}{0} = 4$

   other: $\overset{\overline{r}}{0} + \overset{\overline{w}}{0} + \overset{\overline{x}}{0} = 0$

   $\Rightarrow$ 640