INTRODUZIONE ALLA SHELL BASH
(PARTE I)

Marica Antonacci

INFN BARI

OUTLINE

- Cosa è la shell
- Tipi di shell
- Esecuzione di comandi
- Comandi base per la gestione dei processi

SHELL

Programma che permette di **far interagire l'utente (interfaccia testuale) con SO tramite comandi**

- resta in attesa di un comando…
- … mandandolo in esecuzione alla pressione di `<ENTER>`

In realtà **shell è un interprete comandi evoluto**:

- potente linguaggio di scripting
- interpreta ed esegue comandi da standard input o da file comandi

FUNZIONI TIPICHE DI UNA SHELL

- La shell interpreta un vero e proprio linguaggio di programmazione che permette all'utente di controllare l'esecuzione di comandi, sia in modo **interattivo** che in modo "**batch**" ("script" di shell)
- Funzioni tipiche:
  - esecuzione di comandi
  - redirezione dell'input e dell'output
  - "pipelines" di comandi
  - job control
  - variabili e assegnamenti
  - strutture di controllo
  - "scripts"
  - supporto all'uso interattivo
  - supporto al debugging degli script

DIFFERENTI SHELL

- La shell non è unica, un sistema può metterne a disposizione varie
  - Bourne shell (standard), C shell, Korn shell, …
  - L'implementazione della bourne shell in Linux è **bash** (/bin/bash) •
- Ogni utente può indicare la shell preferita
  - La scelta viene memorizzata in /etc/passwd, un file contenente le informazioni di tutti gli utenti del sistema
- La shell di login è quella che richiede inizialmente i dati di accesso all'utente
  - Per ogni utente connesso viene generato un processo dedicato (che esegue la shell)

SHELL INTERATTIVA DI LOGIN

- Una shell di login è quella che si ha di fronte quando è stata completata la procedura di login (può essere esplicitamente avviata con l'opzione -login).
- Se non è stata specificata l'opzione `–noprofile`, tenta di leggere ed eseguire il contenuto dei file di configurazione: `/etc/profile` `~/.bash profile` oppure `~/.bash` login oppure `~/.profile` (in questo ordine).
- Al termine della sessione di lavoro, se il file esiste, legge ed esegue il contenuto di `~/.bash logout`.

USCITA DAL SISTEMA: LOGOUT

- Per uscire da una shell qualsiasi si può utilizzare il comando exit (che invoca la system call exit() per quel processo)
- Per uscire dalla shell di login
  - logout
  - CTRL+D (che corrisponde al carattere <EOF>)
  - CTRL+C
- Per rientrare nel sistema bisogna effettuare un nuovo login

ESECUZIONE DI UN COMANDO

- Ogni comando richiede al SO l'esecuzione di una particolare azione
- I comandi principali del sistema si trovano nella directory **`/bin`**
- Possibilità di realizzare **nuovi comandi (scripting)**
- Per ogni comando, shell genera un processo figlio dedicato alla sua esecuzione
  - Il processo padre attende la terminazione del comando (**foreground**) o prosegue in parallelo (**background**)

FORMATO DEI COMANDI

```
comando [argomento ...]
```

Gli argomenti possono essere:

- opzioni o flag (-)
- parametri

separati da almeno un separatore

Esempio:

```
ls -l -F file1 file2 file3
```

opzioni

parametri

Convenzione nella rappresentazione della sintassi comandi:

- se un'opzione o un argomento possono essere omessi, si indicano tra quadre [opzione]
- se due opzioni/argomenti sono mutuamente esclusivi, vengono separati da | arg1 | arg2
- quando un arg può essere ripetuto n volte, si aggiungono dei puntini: arg...

IL MANUALE

- Esiste un **manuale on-line** (`man`), consultabile per informazioni su ogni comando Linux. Indica:
  - formato del comando (input) e risultato atteso (output)
  - descrizione delle opzioni
  - possibili restrizioni
  - file di sistema interessati dal comando
  - comandi correlati
  - eventuali bug
- per uscire dal manuale, digitare :q (quit per editor tipo vi)

IL COMMANDO MAN

- Il manuale è organizzato in sezioni e sottosezioni; ogni sezione è composta di pagine (logiche)
- Ogni pagina descrive un **singolo argomento** (es.: un comando)

Esempio:

```
$ man man
```

IL COMMANDO MAN

`man [opzione...] titolo...`

Visualizza le pagine del manuale specificate mediante i suoi parametri -s permette di specificare la sezione

Esempio:

```
$ man who
 ....
$ man -s 2 kill
 ....
$ man -s 2 intro
```

Note:
Se il numero di sezione non è specificato, viene selezionata la prima occorrenza
Ogni sezione o sottosezione inizia con una pagina chiamata intro

IL COMMANDO APROPOS

`apropos [word...]`

Cerca i comandi la cui descrizione nel manuale (NAME) contiene le parole specificate (equivale a man -k)

- Il comando non distingue fra maiuscole e minuscole

Esempio:

```
$ apropos manual
apropos (1)           - search the manual page names and descriptions
catman (8)            - create or update the pre-formatted manual pages
help2man (1)          - generate a simple manual page
man (1)               - an interface to the on-line reference manuals
manconv (1)           - convert manual page from one encoding to another
mandb (8)             - create or update the manual page index caches
manpath (1)           - determine search path for manual pages
mdoc.samples (7)      - tutorial sampler for writing BSD manuals with -mdoc
whatis (1)            - display one-line manual page descriptions
whereis (1)           - locate the binary, source, and manual page files for a command
```

IL COMMANDO WHO

who[....][am I]

who lista il nome, terminale, e data/ora di login di tutti gli utenti correnti

who am I come sopra per l'utente che lo esegue

Esempio:

```
$ who
root     pts/0        2019-02-26 09:01 (193.206.185.230)
marica   pts/1        2019-02-26 09:03 (193.206.185.230)
antonacci pts/2        2019-02-26 09:03 (193.206.185.230)
$ who am I
marica   pts/1        2019-02-26 09:03 (193.206.185.230)
```

IL COMMANDO DATE

`date[...]`

Modifica o stampa (nel formato specificato) la data corrente

Esempio:

```
$ date
Tue Feb 26 09:07:05 UTC 2019
```

IL COMMANDO ECHO

`echo [SHORT-OPTION]... [STRING]...`

- È un comando fondamentale usato frequentemente, p.e. negli script, ogni volta che sia necessario visualizzare del testo.
- Opzioni:
- -n non aggiunge il ritorno a capo
- -e indica di interpretare i comandi di escape e\o altri.
- -E indica di non interpretare i comandi di escape (default)

ESECUZIONE IN FOREGROUND

Normalmente, dopo avere creato il processo che esegue il programma richiesto, la shell si pone in uno stato di wait, in attesa che il programma termini ... dopo di che, la shell riprende la sua esecuzione per acquisire il prossimo comando:

Il comando termina producento un exit code intero, che sarà disponibile in una variabile di shell, di nome:

Esempio:

```
$ ls -latr
total 56
-rw-r--r-- 1 marica marica  807 Apr  4  2018 .profile
-rw-r--r-- 1 marica marica 3771 Apr  4  2018 .bashrc
-rw-r--r-- 1 marica marica  220 Apr  4  2018 .bash_logout
drwxr-xr-x 5 root   root   4096 Jan 31 15:24 ..
-rwxrwxr-x 1 marica marica 3382 Feb  1 07:57 script.exp
$ echo $?
0
```

ESECUZIONE IN BACKGROUND

In questo caso la shell crea il processo che esegue il programma richiesto, ma riprende subito l'esecuzione, senza aspettare che il processo termini

```
$ comando argomenti &
```

Esempio:

```
$ sort .profile > /tmp/delme &
[1] 4100
```

IL COMMANDO PS

`ps [opzioni]`

"process status": stampa alcune informazioni sui processi attivi.

Opzioni:

-f lista estesa

-e tutti i processi

…

Esempio:

```
$ ps
  PID TTY          TIME CMD
 4074 pts/1    00:00:00 bash
 4308 pts/1    00:00:00 ps
```

IL COMMANDO TOP

- **ps** produce un'immagine statica dei processi in corso, in pratica fotografa lo stato del sistema al momento in cui viene lanciata l'istruzione di monitoraggio delle esecuzioni.
- Se si desidera ottenere un output più particolareggiato, dinamico e aggiornabile sarà necessario utilizzare il comando **top**; l'output di **top** consente di visualizzare tutti processi correnti e le relative informazioni rilevanti come per esempio il carico sulla **CPU**.

JOB CONTROL

- Esistono vari comandi per il controllo dei processi che permettono di
  - terminare un processo
  - sospendere un processo
  - far ripartire un processo sospeso
  - mandare in background un processo di foreground
  - portare in foreground un processo di background
  - etc.

IL COMMANDO KILL

kill [-signal] processid ...

Invia il segnale specificato ai processi indicati

Alcuni segnali:

TERM -15 Terminazione del processo (è il segnale di default)

KILL -9 Terminazione del processo (non può essere ignorato nè trattato)

STOP Sospensione del processo (non può essere ignorato nè trattato)

CONT Ripartenza di un processo sospeso

TERMINAZIONE DI UN PROCESSO

Un processo può essere terminato solo dal suo proprietario (l'utente che lo ha creato) o da root

Per terminare un processo di foreground: `CTRL+C`

Per terminare un processo di background:

- `kill pid ...`
- `kill -15 pid ...`
- `kill -TERM pid ...`

In casi estremi: `kill -9 pid ...`

Esempio:

```
$ sleep 1000 &
[1] 4318
$ kill 4318
$
[1]+  Terminated
sleep 1000
```