Dentro la Shell – lezione 3

Lorenzo Losa

11 aprile 2012

1 Gestione dei processi

I programmi in esecuzione sono divisi in *processi*, ognuno dei quali ha un *PID* (Process IDentifier), un utente *proprietario* e un processo *padre* (tranne `init`). Può avere dei *figli* ed essere formato da più *thread*. Ci sono vari comandi che permettono di vedere i processi in esecuzione:

- `ps`: semplice lista dei processi;
- `pstree`: l'albero genealogico dei processi;
- `top`: lista dinamica dei processi, con CPU e memoria utilizzata;
- `htop`: come `top`, meglio.

I processi possono essere eseguiti sullo sfondo (in background), di modo che non occupino il terminale:

```
$ okular &
[1] 5155
```

In questo modo Okular è avviato con PID 5155 come *job* 1; per vedere i job attivi:

```
$ jobs
[1]+  In esecuzione               okular &
```

Un job può essere ucciso:

```
$ kill %1
$
[1]+  Terminato                   okular
```

Se il processo è già avviato, è possibile sospenderlo con `^Z` (`ctrl+z`):

```
$ okular
^Z
[1]+  Fermato                     okular
```

Però così è *fermato*, non *in esecuzione*. Ma si può dire di far ripartire l'esecuzione sullo sfondo:

```
$ bg %1
[1]+ okular &
```

Oppure in primo piano (in foreground):

```
$ fg %1
okular
```

Se volessi scaricare un file con `wget` ma sloggandomi...

```
$ wget http://dominio.it/dvd.iso &> /dev/null &
[1] 5711
$ exit
```

Non funziona! Quando esco il download si ferma. Soluzione:

```
$ nohup wget http://dominio.it/dvd.iso &
[1] 5733
$ exit
```

In questo modo l'esecuzione continua anche quando sono uscito. E se mi dimentico di usare `nohup`? È possibile rimediare in un secondo momento fermando l'esecuzione, riattivandola sullo sfondo e poi "staccandola" con `disown`:

```
$ wget http://dominio.it/dvd.iso &> /dev/null
^Z
[1]+  Fermato       wget http://dominio.it/dvd...
$ bg %1
[1]+ wget http://dominio.it/dvd.iso &>/dev/null &
$ disown %1
$ exit
```

`screen` è un comodo programma che permette di:

- aprire più sessioni di terminale in un'unica sessione di login;
- chiudere e riaprire il terminale senza interrompere l'esecuzione dei processi.

Alcuni comandi per `screen`:

C-a " lista delle sessioni aperte
C-a c crea una nuova sessione (**c**reate)
C-a n passa alla sessione successiva (**n**ext)
C-a p passa alla sessione precedente (**p**revious)
C-a 0 passa alla sessione 0

...
C-a 9 passa alla sessione 9
C-a d "stacca" screen, mantenendolo in esecuzione (**d**etach)

2 Gestione dei pacchetti su Debian e derivate

Spesso sui sistemi Linux il software è distribuito in *pacchetti* che:

- possono contenere: programmi, ma anche librerie, documentazione, traduzioni, ecc.;
- possono *dipendere* da altri pacchetti;
- contengono sia file che le istruzioni su come trattarli.

Esistono vari formati di pacchetti: `.deb` (Debian), `.rpm` (Red Hat) e altri.

Su Debian e derivate (come le *ubuntu) i pacchetti possono essere utilizzati su due livelli:

- a basso livello: gestirli a mano con `dpkg`;
- ad alto livello: gestirli in automatico con `apt`.

...oppure con le tante altre interfacce (anche grafiche).

Alcuni usi di `dpkg`:

- Installare un pacchetto:

```
dpkg -i pacchetto.deb
dpkg --install pacchetto.deb
```

- Rimuovere un pacchetto (file di configurazione esclusi):

```
dpkg -r nomepacchetto
dpkg --remove nomepacchetto
```

- Rimuovere un pacchetto (file di configurazione inclusi):

```
dpkg -P nomepacchetto
dpkg --purge nomepacchetto
```

- Elencare i file contenuti in un pacchetto installato:

```
dpkg -L nomepacchetto
dpkg --listfiles nomepacchetto
```

- Cercare file nei pacchetti:

```
dpkg -S ricerca
dpkg --search ricerca
```

- Riconfigurare un pacchetto:

  ```
  dpkg-reconfigure nomepacchetto
  ```

APT usa la struttura di `dpkg` per:

- ottenere da solo i pacchetti richiesti;
- risolvere da solo le dipendenze, ottendendo i pacchetti necessari;
- tenere il sistema aggiornato.

APT deve essere istruito su dove cercare i pacchetti; nel caso più semplice è scritto in `/etc/apt/sources.list` come:

```
deb http://ftp.debian.org/debian/ stable main
deb-src http://ftp.debian.org/debian/ stable main
```

Usi comuni di APT:

- Installare un pacchetto:

  ```
  apt-get install nomepacchetto
  ```

- Rimuovere un pacchetto (configurazioni escluse):

  ```
  apt-get remove nomepacchetto
  ```

- Rimuovere un pacchetto (configurazioni comprese):

  ```
  apt-get remove --purge nomepacchetto
  ```

- Rimuovere i pacchetti "che non servono più" (installati come dipendenze di pacchetti che sono stati rimossi):

  ```
  apt-get autoremove
  ```

- Aggiornare le informazioni sui pacchetti (da fare sempre prima di un aggiornamento e dopo aver modificato `sources.list`):

  ```
  apt-get update
  ```

- Aggiornare tutti i pacchetti (a meno che non abbiano nuove dipendenze):

  ```
  apt-get upgrade
  ```

- Aggiornare davvero tutti i pacchetti:

  ```
  apt-get dist-upgrade
  ```

- Cancellare i pacchetti scaricati:

  ```
  apt-get clean
  ```

Inoltre, APT ha i poteri della Super Mucca:

```
$ apt-get moo
          (__)
          (oo)
    /------\/
   / |    ||
  *  /\---/\
     ~~   ~~
....␣"Have␣you␣mooed␣today?"...
```

Online è possibile trovare informazioni sui pacchetti contenuti nella distribuzione dai seguenti siti:

- `packages.debian.org` per Debian;
- `packages.ubuntu.com` per *Ubuntu.

Altrimenti, direttamente da shell, è possibile utilizzare APT per:

- ottenere la descrizione e alcune altre informazioni su un pacchetto:

  ```
  apt-cache show nomepacchetto
  ```

- cercare pacchetti:

  ```
  apt-cache search ricerca
  ```

- avere un elenco dei pacchetti installati, o cercare frai pacchetti installati:

  ```
  dpkg -l [ricerca]
  ```

3 Accesso remoto

Ci si può connettere ad un sistema anche da remoto:

- con un'interfaccia testuale: ssh

- con un'interfaccia grafica: direttamente via X, con VNC, ...

SSH (Secure SHell) permette di connettersi ad una macchina remota (che ha attivo il server SSH) con:

```
ssh [UTENTE@]MACCHINA
```

Ad esempio, per connettersi alla propria macchina (cosa poco interessante, ma utile come esempio):

```
$ ssh localhost
The authenticity of host 'localhost (127.0.0.1)' can't
   ➥be established.
ECDSA key fingerprint is 6f:69:a7:cf:cb:99:79:ee:4e:8d:
   ➥b6:2d:d2:e0:cc:90.
Are you sure you want to continue connecting (yes/no)?
   ➥yes
Warning: Permanently added 'localhost' (ECDSA) to the
   ➥list of known hosts.
utente@localhost's password:
Welcome to Ubuntu 11.10 (GNU/Linux 3.0.0-16-generic
   ➥x86_64)

 * Documentation:  https://help.ubuntu.com/

Last login: Thu Mar 15 18:53:33 2012
$
```

SCP, un comando che si appoggia a SSH, permette di copiare file fra macchine diverse:

```
scp [-r] origine destinazione
```

Dove `origine` e `destinazione` sono nella forma:

```
[[utente@]macchina:]/directory/file
```

SSH è ottimo per un accesso testuale, ma ha un supporto (limitato) anche per le interfacce grafiche. Se lanciato con l'opzione `-X`, permette di lanciare programmi grafici sul server con output sul client:

```
$ ssh -X localhost
utente@localhost's password:
$ okular
```

Nel caso uno voglia un intero ambiente grafico, ci sono altre soluzioni, come VNC.