Sistemi Operativi[1]

Mattia Monga

Dip. di Informatica
Università degli Studi di Milano, Italia
mattia.monga@unimi.it

a.a. 2013/14

Lezione XVI: Unix power tools

find

Per selezionare file con determinate caratteristiche si usa `find`

find percorso predicato

Seleziona, nel sottoalbero definito dal percorso, tutti i file per cui il predicato è vero

Spesso usato insieme a a `xargs`

find percorso predicato| xargs comando

funzionalmente equivalente a

comando $(find percorso predicato)

ma evita i problemi di lunghezza della riga di comando perché xargs si preoccupa di "spezzarla" opportunamente.

Due espressioni idiomatiche

Spesso si vuole fare un'operazione per ogni file trovato con find. L'espressione piú naturale sarebbe:

```
1 for i in $(find percorso predicato); do
2   comando $i
3 done
```

Questa forma presenta due problemi: può eccedere la misura della linea di comando e non funziona correttamente se i nomi dei file contengono *spazi*

Due espressioni idiomatiche

Un'alternativa è

```
1 find percorso predicato -print0 | xargs -0 -n 1
```

In questo modo (-print0) i file trovati sono separati dal carattere 0 anziché spazi e xargs è capace di adattarsi a questa forma. Un'alternativa piú generale che mostra la potenza del linguaggio di shell che non distingue fra comandi e costrutti di controllo di flusso (sono tutti "comandi" utilizzabili in una pipeline)

```
1 find percorso predicato | while read x; do
2   comando $x
3 done
```

read x legge una stringa e la assegna alla variabile *x*.

Esercizi

1. Trovare il file piú "grosso" in un certo ramo
2. Copiare alcuni file (ad es. il cui nome segue un certo pattern) di un ramo in un altro mantenendo la gerarchia delle directory
3. Calcolare lo spazio occupato dai file di proprietà di un certo utente
4. Scrivere un comando che conta quanti file ci sono in un determinato ramo del filesystem

Archivi

Un archivio *archive* è un file di file, cioè un file che contiene i byte di diversi altri file e i relativi *metadati.* (Cfr. con una directory, che è un file speciale, che sostanzialmente contiene solo l'elenco dei file)

- `ar` L'archiviatore classico, generalmente utilizzato per le librerie (provare ar t /usr/lib/i86/libc.a
- `tar` Tape archive, standard POSIX
  tar cvf archivio.tar lista_files

Gli archivi possono essere compressi con `compress` o, piú comunemente, con `gzip` o `bzip2`
I file `.zip` sono archivi compressi.

Altre utility

Altre utility "standard" di cui è bene conoscere almeno l'esistenza

Prog. (sez. man)	**Descrizione**
`uniq (1)`	report or omit repeated lines
`cut (1)`	remove sections from each line of files
`tr (1)`	translate or delete characters
`dd (1)`	convert and copy a file
`stat (1)`	display file or file system status
`tee (1)`	read from standard input and write to standard output ...
`basename (1)`	strip directory and suffix from filenames
`dirname (1)`	strip non-directory suffix from file name
`sed (1)`	stream editor for filtering and transforming text
`seq (1)`	print a sequence of numbers

Inoltre è molto utile conoscere le espressioni regolari (`man 7 re_format`), usate da `grep`, `sed`, ecc.

Esercizi

1. Creare un archivio `tar.gz` contenente tutti i file la cui dimensione è minore di 50KB
2. Rinominare un certo numero di file: per esempio tutti i file `.png` in `.jpg`
3. Creare un file da 10MB costituito da caratteri casuali (usando `/dev/random`) e verificare se contiene la parola JOS
4. Trovare l'utente che ha il maggior numero di file nel sistema
5. Trovare i 3 utenti che, sommando la dimensione dei loro file, occupano piú spazio nel sistema.

Link

- `http://www.gnu.org/software/fileutils/fileutils.html`

Lezione XVIII: Shell e file system

Shell e file system

- Ogni processo (compresa la shell stessa) ha associata una *directory di lavoro* (working directory), che può essere cambiata col comando (interno alla shell) `cd`
- I programmi fondamentali per operare sul file system

ls (1)	list directory contents
cp (1)	copy files and directories
rm (1)	remove files or directories
mv (1)	move (rename) files
mkdir (1)	make directories
rmdir (1)	remove empty directories
df (1)	report file system disk space usage
du (1)	estimate file space usage
pwd (1)	print name of current/working directory

Permessi

Ad ogni file vengono associati dei *permessi*, che definiscono le azioni permesse sui dati del file

- **Read:** leggere il contenuto del file o directory
- **Write:** scrivere (cambiare) il file o directory
- **eXecute** eseguire le istruzioni contenute nel file o accedere alla directory

R	W	X	
1	1	0	6
1	0	1	5
1	0	0	4
1	1	1	7

I permessi possono essere diversi per 3 categorie di utenti del sistema:

- **User:** il "proprietario" del file
- **Group:** gli appartenenti al gruppo proprietario
- **All:** tutti gli altri

Agire sui permessi

- Cambiare il proprietario
  - chown utente[:gruppo] file
- Cambiare il gruppo
  - chgrp gruppo file
- Cambiare i permessi
  - chmod 755 file
  - chmod +x file
  - chmod a=rw file
  - chmod g−x file
- (per creare un utente: adduser)

Il bit SUID

Il proprietario di un processo in esecuzione è normalmente *diverso* dal proprietario del file contente un programma (e diverso ad ogni esecuzione)

- effective UID bit: il processo assume come proprietario il proprietario del file del programma
- SUID `root`
- chmod 4555 file
- chmod u+s file