Sistemi Operativi e informatica[1]

Massimo Marchi

Dip. Scienze dell'Informatica
Università degli Studi di Milano, Italia
marchi@dsi.unimi.it

a.a. 2011/12

File System vs CLI

Alcuni comandi Unix eseguibili da shell:

- **pwd** : directory in cui sono
- **cd** : cambio directory
- **ls** : lista la directory
  - **-r** : recursive
  - **-a** : all
  - **-l** : stampa più informazioni sui file
- **cat** : stampa un file

Per approfondire:

- **man** <**comando**>: visualizza il manuale di un comando

Metadata dei File

marchi@glider-vmware:/

File Edit View Search Terminal Help

```
[marchi@glider-vmware /]$ ls -al
total 132
dr-xr-xr-x.  24 root root  4096 Sep 15 12:27 .
dr-xr-xr-x.  24 root root  4096 Sep 15 12:27 ..
-rw-r--r--    1 root root     0 Sep 14 17:35 .autofsck
-rw-r--r--    1 root root     0 Sep  6  2009 .autorelabel
dr-xr-xr-x.   2 root root  4096 Dec 10 08:59 bin
dr-xr-xr-x.   5 root root  3072 Dec 21 15:38 boot
drwxr-xr-x    2 root root  4096 May 26  2011 cgroup
drwx------.   3 root root  4096 Sep  5  2009 dbus
drwxr-xr-x   19 root root  3760 Sep 15 15:41 dev
drwxr-xr-x. 134 root root 12288 Feb  7 08:32 etc
drwxr-xr-x.   4 root root  4096 May 18  2011 home
dr-xr-xr-x.  21 root root 12288 Dec 21 15:36 lib
drwx------.   2 root root 16384 Nov 19  2008 lost+found
drwxr-xr-x.   2 root root  4096 May 18  2011 media
drwxr-xr-x.   5 root root  4096 May 18  2011 mnt
drwxr-xr-x.   2 root root  4096 May 18  2011 opt
dr-xr-xr-x  134 root root     0 May 18  2011 proc
dr-xr-x---.  12 root root  4096 Dec 22 17:57 root
drwxr-xr-x    4 root root  4096 Sep 15 13:44 run
dr-xr-xr-x.   2 root root 12288 Dec 21 15:36 sbin
drwxr-xr-x.   2 root root  4096 May 18  2011 selinux
drwxr-xr-x.   2 root root  4096 May 18  2011 srv
```

Permessi

Proprietario e gruppo

Dimensione in byte

Data di Modifica

Nome

Metadata dei File

Ad ogni file sono associate un certo numero di informazioni necessarie per la gestione del file stesso, tra cui:

- nome del file
- proprietario e gruppo associato
- tempo di creazione e modifica.
- permessi di accesso

Se il file system lo supporta è possibile associare ai file anche altre informazioni sotto forma di attributi estesi:

- *ls -l@* : (Mac OSX) lista gli attributi estesi

Permessi sui File

marchi@glider-vmware:/

File Edit View Search Terminal Help

```
ls: cannot access /etc/shadoew: No such file or directory
-rw-r--r-- 1 root root 2023 Dec  5  2010 /etc/passwd
[marchi@glider-vmware /]$ ls -l /etc/passwd /etc/shadow
-rw-r--r-- 1 root root 2023 Dec  5  2010 /etc/passwd
-r-------- 1 root root 1357 Dec  5  2010 /etc/shadow
[marchi@glider-vmware /]$ ls -l /etc/passwd /etc/shadow /
-rw-r--r--   1 root root 2023 Dec  5  2010 /etc/passwd
-r--------   1 root root 1357 Dec  5  2010 /etc/shadow

/:
total 112
dr-xr-xr-x    2 root root  4096 Dec 10 08:59 bin
dr-xr-xr-x.   5 root root  3072 Dec 21 15:38 boot
drwxr-xr-x    2 root root  4096 May 26  2011 cgroup
drwxr-xr-x   19 root root  3760 Sep 15 13:41 dev
drwxr-xr-x. 124 root root 12288 Feb  7 08:32 etc
drwxr-xr-x.   4 root root  4096 May 18  2011 home
dr-xr-xr-x.  21 root root 12288 Dec 21 15:36 lib
drwx------.   2 root root 16384 Nov 19  2008 lost+found
drwxr-xr-x.   2 root root  4096 May 18  2011 media
drwxr-xr-x.   5 root root  4096 May 18  2011 mnt
drwxr-xr-x.   2 root root  4096 May 18  2011 opt
dr-xr-xr-x  135 root root     0 May 18  2011 proc
dr-xr-x---.  12 root root  4096 Dec 22 17:57 root
```

Tipo

Permessi del proprietario

Permessi del gruppo

Permessi per chiunque

Permessi sui File

Unix Implementa un semplice ma efficace sistema di permessi di accesso ai file. Ogni file ed ogni processo appartengono ad un proprietario ed ad un gruppo. Se il proprietario coincide viene applicata la *policy* del proprietario, altrimenti se il gruppo coincide, quella del gruppo. Infine se non coincidono nè proprietario nè gruppo viene applicata la *policy* per tutti gli altri. I permessi indicabili sono:

- **r** (read) : permesso di lettura
- **w** (write) : permesso di scrittura, nel caso delle directory permesso di creare e cancellare i file.
- **x** (execute) : permesso di esecuzione, nel caso delle directory permesso di attraversamento.

Permessi sui File

Altri permessi:

- **set user ID** : i file eseguibili vengono eseguiti con i permessi del proprietario del file.
- **set group ID** : i file eseguibili vengono eseguiti con i permessi del gruppo del file, nel caso delle directory indica che i nuovi file e sottodirectory creati al loro interno avranno come gruppo assegnato quello della directory che li contiene
- **sticky**: Applicato ad una directory indica che i file in essa contenuti possono essere cancellati e spostati solamente dal proprietario, o dal proprietario della directory che li contiene.

Autenticazione

Ad ogni nuovo processo viene assegnato un identificativo per il proprietario ed il gruppo che ne determinano i permessi:

- **UID** : user ID
- **GID** : group ID

La corrispondenza tra utente, gruppo e corrispondente ID è indicata in questi file (sono esclusi gli account gestiti con protocolli di rete tipo NIS):

- **/etc/passwd** : contiene informazioni sugli utenti locali.
- **/etc/group** : contiene informazioni sui gruppi locali.

cat /etc/passwd

marchi@glider-vmware:/

File Edit View Search Terminal Help

```
polkituser:x:87:87:PolicyKit:/:/sbin/nologin
pulse:x:498:498:PulseAudio System Daemon:/var/run/pulse:/sbin/nologin
nscd:x:28:28:NSCD Daemon:/:/sbin/nologin
rpcuser:x:29:29:RPC Service User:/var/lib/nfs:/sbin/nologin
nfsnobody:x:65534:495:Anonymous NFS User:/var/lib/nfs:/sbin/nologin
tcpdump:x:72:72::/:/sbin/nologin
avahi:x:497:494:avahi-daemon:/var/run/avahi-daemon:/sbin/nologin
smolt:x:496:493:Smolt:/usr/share/smolt:/sbin/nologin
openvpn:x:495:492:OpenVPN:/etc/openvpn:/sbin/nologin
mailnull:x:47:47::/var/spool/mqueue:/sbin/nologin
smmsp:x:51:51::/var/spool/mqueue:/sbin/nologin
apache:x:48:48:Apache:/var/www:/sbin/nologin
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
haldaemon:x:68:68:HAL daemon:/:/sbin/nologin
rpc:x:32:32:Rpcbind Daemon:/var/lib/rpcbind:/sbin/nologin
gdm:x:42:42::/var/lib/gdm:/sbin/nologin
marchi:x:500:500:Massimo Marchi:/home/marchi:/bin/bash
mysql:x:27:491:MySQL Server:/var/lib/mysql:/bin/bash
nagios:x:501:501::/home/nagios:/bin/bash
rtkit:x:493:486:RealtimeKit:/proc:/sbin/nologin
abrt:x:492:485::/etc/abrt:/sbin/nologin
saslauth:x:491:484:"Saslauthd user":/var/empty/saslauth:/sbin/nologin
usbmuxd:x:113:113:usbmuxd user:/:/sbin/nologin
[marchi@glider-vmware /]$
```

cat /etc/group

marchi@glider-vmware:/

File Edit View Search Terminal Help

```
haldaemon:x:68:
gdm:x:42:
marchi:x:500:
rpc:x:32:
video:x:39:
audio:x:63:
cdrom:x:11:
tape:x:33:
dialout:x:18:
svndefault:x:2001:marchi
svngroup1:x:2002:marchi
svngroup2:x:2003:marchi
mysql:x:491:
nagios:x:501:
jackuser:x:490:
desktop_admin_r:x:488:
desktop_user_r:x:487:
rtkit:x:486:
abrt:x:485:
saslauth:x:484:
usbmuxd:x:113:
wireshark:x:483:
cgred:x:482:
[marchi@glider-vmware /]$
```

Comandi per i Permessi

Alcuni comandi per manipolare i permessi dei file e dei processi:

- **chmod** : cambia i permessi di un file o cartella
- **chown** : cambia il proprietario di un file o directory (root only)
- **chgrp** : cambia il gruppo associato ad un file o directory (root only)
- **groups** : visualizza i gruppi di un utente
- **newgrp** : apre una nuova shell con gruppo associato differente

Comandi sui File

- **rm, mv, cp** : rimuovi, muovi, copia
- **file** : restituisce il tipo di file
- **less** (more) : visualizza un file di testo
- **touch** : crea un file vuoto
- **find** : trova i file che rispettano un criterio dato
- **locate** : trova i file che rispettano un criterio dato consultando un indice precalcolato

Comandi di Manipolazione File di Testo

Alcuni comandi utili per manipolare file di testo:

- **vi** : editor di testo (non user-friendly)
- **grep** : cerca stringhe dentro ad un testo
- **sed** : sostituisce in ogni linea un espressione con un altra
- **awk** : parser di testo

Altri Comandi Bash

- **for** : ciclo for
- **echo** : stampa a video
- **${nome/<orig>/<dest>}** : sosituzione in una variabile
- **$(<comando>)** : esegui comando e restituisci l'output
- **date** : stampa la data
- **date +%Y%m%d%H%M%S** : stampa la data in formato ordinabile lessico-graficamente

Stdin, Stdout, Stderr, Redirezioni

Ogni processo possiede tre canali di comunicazione:

- **stdin**: input preferenziale dei dati (canale 0)
- **stdout**: output preferenziale dei risultati (canale 1)
- **stderr**: output dei messaggi di errore (canale 2)

E' possibile ridirigere questi canali verso un file o verso un altro processo (pipe) (sintassi Bash):

- **ps aux | grep marchi** : lo stdout di ps è usato come stdin di grep
- **ps aux > testo.txt** : lo stdout di ps è rediretto verso il file testo.txt
- **host pippo > /dev/null 2>&1** : lo stdout è scartato (attraverso il device speciale /dev/null), lo stderr viene rediretto verso stdout e scartato anch'esso

Gerarchia di processi

In UNIX ogni processo è associato ad un processo padre che lo ha generato. Terminare un processo padre di default provoca la terminazione dei processi figli.

- **ps** : stampa la lista dei processi
- **kill** : manda un segnale ad un processo
  - **kill -15 <pid>** : invia al processo la richiesta di terminazione
  - **kill -9 <pid>** : provoca la terminazione forzata.
- **pstree**: (non sempre implementato) visualizza la gerarchia dei processi
- **ps aux** : visualizza tutti i processi con varie informazioni tra cui: PID, proprietario, comando o processo che lo ha generato.
- **ps axo user,group,tty,pid,ppid,stat,comm** : visualizza tutti i processi con alcune informazioni personalizzate

Processi Background e Foreground

Il processo **foreground** è il processo attualmente in esecuzione nella shell. Un processo attivo mentre la shell accetta o sta eseguendo altri comandi detto processo **background**. In un calcolatore esistono vari processi background, detti **servizi o daemons**, che si occupano di fornire servizi all'utente o al sistema operativo. Comandi Bash per controllare i processi:

- <**comando**> **&** : esegue un processo in background
- **CTRL+Z** : sospende il processo foreground
- **bg** : manda un processo sospeso in background
- **fg** : riprende in foreground l'ultimo processo in background
- **jobs** : lista i processo background definiti dall'utente
- **nohup** <**comando**> : disconnette il processo dal terminal ed evita che la terminazione della shell provochi la terminazione di <*comando*>

Alcuni Esempi di Scripting

Cerca /etc il files system i file con estensione .conf. Ignora i messaggi di errore:
find /etc -name *.conf ! -type d 2>/dev/null

Crea un gruppo di cartelle nella directory locale con nome tmp-<i> e i da 100 a 199:
for((i=100;i<=199;i++)) ; do mkdir tmp-$i ; done

Crea un file con i nomi di tutte le cartelle di nome tmp-<i>:
echo -n > lista.txt ; for i in tmp-1[0-9][0-9] ;
do echo $i >> lista.txt ; done

Crea all'interno di tutte le cartelle in *lista.txt* un file vuoto di nome *prova*:
for i in $(cat lista.txt) ; do touch $i/prova ; done

File Script

Una sequenza di comandi può essere memorizzata in un file, uno *script*, e resa eseguibile:

cat << EOFrame > prova.sh
#!/bin/bash
date
EOFrame

chmod 755 prova.sh
./prova.sh

La prima riga dello script informa la shell sull'interprete in grado di eseguire lo script, in questo caso */bin/bash*.

Parsing di Testi in Perl

Bash è principalmente adatto alla manipolazione automatica di file e directory. Perl, in virtù della sua capacità di definire facilmente parser di stringhe, si presta meglio nel caso sia necessario manipolare testi, es. file di configurazione
Un esempio di script Perl:

```
#!/usr/bin/perl
while(<>){
  if(/^root[\t ]/){
    print ">>".$_;
  }
}
```

chmod 755 prova.pl
ps aux | ./prova.pl