

Proprietà dei file: permessi

- Partiamo dall'output di `ls`; `ls -l` permette di visualizzare le informazioni estese sui file:

```
$ ls -l /  
drwxr-xr-x  2 root  root    4096 2004-04-14 18:07 bin  
drwxr-xr-x  3 root  root   1024 2004-04-17 22:30 boot  
.....
```

- Permessi dei file:
 - tipo di file: 1 carattere (d: directory, -: file regolare, b: dispositivo a blocchi, s: socket, p: pipe, l:link simbolico)
 - ogni tripletta di caratteri successiva rappresenta i permessi per una categoria di utenti: proprietario del file (u), gruppo del file (g), altri (o).
-
-

Proprietà dei file: permessi

- rwx rappresentano i permessi rispettivamente di lettura, scrittura ed esecuzione(file)/ attraversamento(cartelle).
 - SETUID/SETGID:al posto di “x” (rispettivamente per users e group) è specificato l'attributo “s”.
 - Un programma viene eseguito con i permessi dell'utente/gruppo di appartenenza, non con quelli dell'utente che lo ha avviato;
 - Per le cartelle, i file creati ereditano utente/gruppo dalla directory e non dal creatore del file.
 - Evitare l'abuso: sudo può rivelarsi più “sicuro”.
-
-

Proprietà dei file: permessi (chmod)

- rwx possono essere considerati come bit, ed i permessi ad ogni categoria di utenti come l'OR tra i bit relativi (r: 4, w: 2, x:1).
 - ad es. rw- = 6, r-x = 5, r-- = 4, --x = 1, ...
 - Il comando chmod supporta entrambe le sintassi
 - `chmod [OPTION]... MODE[,MODE]... FILE...`
 - `chmod u+s,g+w,o-rxw nomefile`
 - `chmod u=rwx,g=rx,o= nomefile`
 - `chmod [OPTION]... OCTAL-MODE FILE...`
 - `chmod 760 nomefile`
 - Permette la ricorsione con -R.
-
-

Proprietà dei file: permessi (umask)

- Tramite il comando umask è possibile impostare i permessi predefiniti per i file creati.
 - I permessi di un nuovo file sono specificati dal programma che lo crea; questi sono però filtrati tramite AND bit a bit con il NOT del valore di umask.
 - Es. umask 027:
 - 0: utente, si possono impostare tutti i permessi
 - 2: gruppo, non si può impostare w
 - 7: altri: non si può impostare alcun permesso ($7 = 4 + 2 + 1 = rwx$)
-
-

Proprietà dei file: link

- Segue (vedere `ls -l`) il numero di collegamenti (link) del file.
 - una directory contiene ad es. un link verso sé stessa (`.`) ed uno verso la directory padre (`..`)
 - il comando `ln` permette di creare link fisici (anche se nella maggior parte dei casi si creano link simbolici)
 - Il contenuto di un file viene effettivamente cancellato (dal disco) quando il numero dei suoi collegamenti è pari a 0.
 - se si cancella un hard link, questo sparisce, ma il contenuto del file rimane (referenziato da qualche altro file).
-
-

Proprietà dei file: utente, gruppo, ecc

- Altri campi: rispettivamente, utente e gruppo di appartenenza del file.
- Possono essere modificati rispettivamente con `chown` e `chgrp`.
 - `chown` permette, a dispetto del nome, di cambiare anche il gruppo:
 - `chown nuovoutente.nuovogruppo file`
 - entrambi supportano la ricorsione sulle sottocartelle tramite il parametro `-R`.
- Seguono la dimensione del file, quindi la data di ultima modifica e il nome del file

Proprietà dei file: stat

- Il comando `stat` mostra in modo più leggibile le informazioni relative ad file (prelevate dall'inode relativo)
 - L'output di `stat` può essere formattato secondo necessità (parametro `-c / --format, printf-style`).
 - comportamento predefinito: visualizza il tipo, la dimensione, il numero di inode, i permessi ed i tempi di accesso (access: visualizzazione del contenuto), modifica (modify: modifica del contenuto) e cambiamento (change: modifica delle metainformazioni, quelle dell'inode).
-
-

Operazioni sui file: rapido ripasso

- Copia (cp). È possibile specificare più file sorgente, ma in questo caso la destinazione deve essere una directory.
 - cp [opzioni] file ... filedest
 - -r: copia ricorsiva;
 - -a: copia ricorsiva preservando i permessi.
 - Spostamento/modifica del nome (mv).
 - Con un solo parametro sorgente ed una destinazione, rinomina; altrimenti sposta, e in questo caso filedest deve essere una directory.
 - mv [opzioni] file ... filedest
-
-

Operazioni sui file: rapido ripasso

- Rimozione di un file (rm)
 - `rm file [...]`
 - `-f` : non chiede conferma dell'eliminazione.
 - `-r`: cancellazione ricorsiva.
 - NON PROVARE MAI `"rm -rf /"` COME root !
- Creazione/rimozione di una directory
 - `mkdir nomedirectory`
 - `-p`: crea anche le cartelle di livello superiore, se necessario.
 - `rmdir nomedirectory`
 - funziona solo su cartelle vuote

Operazioni sui file

- Creazione di collegamenti (ln).
 - ln [opzioni]... TARGET [LINK_NAME]
 - Senza altri parametri, tenta di creare collegamenti fisici verso target (hard link); possibile solo verso file della stessa partizione.
 - -s : crea collegamenti simbolici (più usato).
 - È possibile specificare una directory come ultimo parametro: in questo caso, in essa viene creato un link per ogni file specificato.
 - du: mostra dimensione dei file (ricorsivo); provare -h , -c e -s (anche combinandole).
-
-

Compressione: gzip, bzip2

- gzip comprime un solo file alla volta
 - gzip [options] nomefile
 - gunzip [options] nomefile.gz
 - comportamento predefinito: sostituiscono il file originale/compresso rispettivamente.
 - -1 | -2 | ... | -9 : tasso di compressione (9: massimo)
 - zcat: visualizza un file compresso.
 - bzip2: nuovo algoritmo di compressione, l'eseguibile si comporta analogamente a gzip.
 - bzip2/bunzip2/bzcat
-
-

Compressione: tar

- tar (tape archiver): nato per gestire nastri di backup (ancora adesso, funzionale per questo). Quindi NON comprime file, nativamente.
 - tar operazione [opzioni] file ...
 - operazione può essere una tra -A (concatenate), -c (create), -d (find differences), --delete, -r (append), -t (list), -u (update) -x (extract).
 - opzioni: -v per verbose, -f per specificare il file su cui operare (predefinito: stdio/stdout).
 - tar della GNU (presente in tutte le distribuzioni GNU/Linux) permette di richiamare al volo gzip/gunzip o bzip2/bunzip2, con i parametri -z e -j.
-
-

Compressione: zip

- zip : formato di compressione creato dalla PKWare, basato (come gzip) sull'algoritmo di Huffman). Probabilmente il più diffuso.
 - zip [opzioni] file.zip file [altrifile..]
 - comprime i file specificati
 - unzip [opzioni] nomefile.zip [nomefile ...]
 - opzioni:
 - tasso di compressione, come gzip: -1,... , -9.
 - -r : compressione ricorsiva
-
-

Compressione: esempi

- `tar -xvzf nomefile.tar.gz`
 - decomprime il file `tar.gz`
 - `tar -xvjf nomefile.tar.bz2 unfile`
 - estrae `file1` da `nomefile.tar.bz2`
 - `tar -tvjf nomefile.tar.bz2`
 - visualizza (test) i file contenuti in `nomefile.tar.bz2`
 - `tar -cvzf nuovofile.tar.gz file ...`
 - comprime i file specificati in `nuovofile.tar.gz`
 - `zip -9r nuovofile.zip file ...`
 - comprime i file specificati (ricorsivamente) con il massimo tasso di compressione in `nuovofile.zip`
 - `unzip nomefile.zip`
 - decomprime `nomefile.zip`
 - NB: il `-` davanti ai parametri di `tar` è opzionale !!!!
-
-

Partizioni: mount e df

- `mount`: senza parametri, mostra i filesystem montati (contenuto di `/etc/mtab`).
 - `mount [-o opzioni] [-t tipo] device puntodimontaggio`
 - Se `device` o `puntodimontaggio` sono specificati in `fstab`, uno dei due può essere omesso.
 - È possibile specificare un'immagine di filesystem (se il kernel supporta il `loopback`), tramite l'opzione `loop`.
 - `tipo`: indica il tipo del filesystem. Spesso identificato automaticamente (tranne, ad es, `vfat` vs `fat16`)
 - `umount puntodimontaggio`
 - smonta il dispositivo montato.
 - `df [-h]`: mostra occupazione del filesystem.
-
-

Partizioni: mount, esempi

- `mount -t vfat /dev/fd0 /media/floppy`
 - monta il primo floppy disk, formattato vfat, nel punto di montaggio `/mnt/floppy`
 - `mount [-t auto] /dev/hda10 /mnt/extra`
 - monta la partizione `/dev/hda10` in `/mnt/extra`, effettuando l'autoriconoscimento del filesystem
 - `mount -o ro,loop -t iso9660 nomeiso.iso /mnt/iso`
 - monta un'immagine iso (`nomeiso.iso`) nel punto di montaggio `/mnt/iso`
 - Attenzione: un utente non può specificare parametri personalizzati per mount. Può però montare filesystem con opzione "user" in `fstab`.
-
-

fstab e *mtab*

- /etc/fstab contiene le informazioni statiche sui filesystem. I parametri sono quelli specificati in mount (e descritti nella relativa pagina man).
- /etc/mtab contiene l'elenco dei filesystem correntemente montati.

#<fs>	<mount point>	<tipo>	<opzioni>	<dump>	<pass>
/dev/hda1	/mnt/windows	vfat	umask=002,gid=100	0	0
/dev/hda5	none	swap	sw	0	0
/dev/hda6	/	ext3	errors=remount-ro	0	1
/dev/hda7	/home	ext3	defaults	0	2
proc	/proc	proc	defaults	0	0
/dev/fd0	/media/floppy	vfat	user,noauto	0	0
/dev/cdrom	/media/cdrom	iso9660	ro,user,noauto	0	0

fstab e mtab

- Importanti tra le opzioni sono
 - `noauto`: il dispositivo non viene montato automaticamente durante l'avvio del sistema.
 - `user`: se specificato, il filesystem relativo può essere montato dagli utenti (ma senza modificare gli altri parametri).
 - E in particolare, gli ultimi due parametri:
 - `dump`: normalmente 0 (dump: utility di backup).
 - `pass`: l'ordine del controllo all'avvio (fsck).
 - È possibile montare tutti i filesystem di `/etc/fstab` per cui non sia stato specificato `noauto` con `mount -a`
-
-

quota: configurazione del sistema

- Se il tipo del filesystem supporta le quote, la procedura per la loro attivazione non presenta particolari difficoltà.
 - Normalmente ogni distribuzione mette a disposizione alcuni pacchetti (con nomi tipo “quota” e “quotatool”) contenenti i programmi e gli script per la gestione.
 - Per attivare: aggiungere le opzioni `usrquota` e/ o `grpquota` (rispettivamente per attivare la quota sugli utenti e sui gruppi) nel file `fstab`.
 - Dopo la modifica ad `fstab`, eseguire (una tantum)
 - `quotacheck -a -v -u -g`
-
-

quota

- Gli script dovrebbero occuparsi di questa operazione, la prima volta (ma non si sa mai).
 - Sempre al primo avvio, il comando si occupa di creare i file (vuoti, ad es. con il comando touch) (a)quota.user e/o (a)quota.group nella root della partizione/fs su cui sono attive le quote, con permessi 0600. (r--)
 - quotacheck ... va eseguita periodicamente (ma se ne occupano gli script)
 - Per attivare all'avvio (ma se ne occupano gli script):
 - quotaon -a -v -u -g
-
-

Acl (Access Control List) POSIX

- Rispetto al sistema standard basato su utente e gruppo consentono di specificare un controllo di accesso più dettagliato per i file e quindi per le risorse (dispositivi, pipe, ecc).
 - Supporto integrato nelle più recenti distribuzioni, da attivare aggiungendo l'opzione "acl" nell'apposito campo di fstab..
 - Comandi base:
 - `getfacl [opzioni] nomefile`
 - `setfacl opzioni acl nomefile`
-
-

Acl (Access Control List) POSIX

- `setfacl -m u:[utente]:permessi file1 [file2 ...]`
 - assegna i permessi ad utente
 - `setfacl -m g:[gruppo]:permessi file1 [file2 ...]`
 - assegna i permessi a gruppo
 - `setfacl -m o:permessi file1 [file2 ...]`
 - assegna i permessi agli altri
 - `setfacl -m m:permessi file1 [file2 ...]`
 - imposta la umask predefinita
-
-

Acl (Access Control List) POSIX

- `setfacl -x u:[utente] file1 [file2...]`
 - rimuove utente dall'acl
 - `setfacl -x g:[gruppo] file1 [file2...]`
 - rimuove gruppo dall'acl
 - `setfacl -b file1 [file2 ...]`
 - ripristina l'acl predefinita
 - `setfacl --restore=nomefile`
 - ripristina le acl salvate in nomefile, generate tramite `getfacl -R`
 - Opzione `-R`: applica ricorsivamente
-
-

Acl (Access Control List) POSIX

- Situazione tipica: permettere l'accesso ad un file ad un utente, senza ricorrere il sysadmin (senza creare nuovi gruppi), ed evitando l'accesso al resto del mondo.
 - `setfacl -m u::nomeutente::rw nomefile`
- Alla fine:
 - `setfacl -b nomefile`