Attributi e permessi

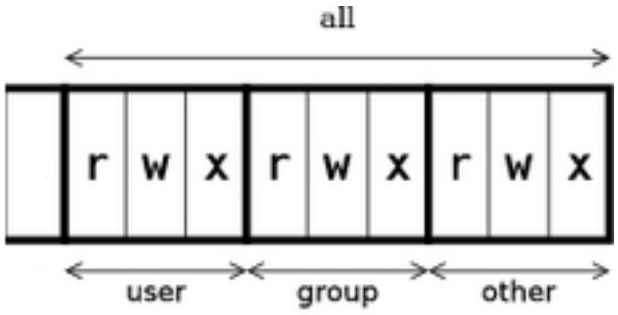
Linux è un **sistema multiutente** che offre la possibilità di associare permessi di azioni (lettura, scrittura , esecuzione) a singoli file e/o directory. Quando vengono applicati questi permessi ad un file o ad una directory, vengono applicati su tre livelli distinti:

- **utente proprietario**: che è colui che ha creato il file;
- **gruppo proprietario**: cioè il gruppo di utenti al quale il proprietario appartiene;
- **il resto degli utenti**;

Ciascun utente del sistema ha un suo numero identificativo detto UID (User IDentifier) ed un numero identificatore del gruppo GID (Group IDentifier) ed in base a questi si definisce il ruolo di ogni utente nei confronti dei file.
Se proviamo quindi a dare il comando "ls -l" notiamo che la prima parte dell'output che ci viene fornita riguarda i permessi sui file:
ES:

```
giordy@grissom:~$ ls -lh
totale 103K
drwxrwxrwx   3 giordy giordy  296
-rw-r--r--   1 giordy staff  2,8K
-rw-r--r--   1 giordy staff  2,9K
-rw-r--r--   1 giordy staff  2,9K
-rwxrwxrwx   1 giordy giordy   86
drwxrwxrwx   4 giordy giordy 2,3K
drwxrwxrwx  15 giordy giordy  872
drwxrwxrwx   7   1001   1001  728
-rw-r--r--   1 giordy staff    39
drwxrwxrwx   2 giordy giordy 1,1K
drwxr-xr-x   2 giordy giordy  192
drwxr-sr-x   9 giordy staff  2,3K
-rw-r--r--   1 giordy staff   455
drwxr-sr-x   2 giordy staff   320
drwxr-sr-x   2 root   staff   680
-rw-r--r--   1 giordy staff   23K
drwx------  15 giordy staff   416
-rw-r--r--   1 giordy staff   28K
drwxrwxrwx  16 giordy giordy 3,3K
drwxrwxrwx  29 root   root   1,2K
-rwxr-xr-x   1 giordy staff  1,3K
drwxrwxrwx   2 giordy giordy  632
-rw-r--r--   1 giordy staff  9,4K
```

Notiamo che ci sono 10 campi, tralasciando il primo (che spiegheremo dopo), possiamo suddividere quelli restanti nel seguente modo:



Quindi il 2°, il 3° e il 4° riguardano i permessi dell'utente, il 5°, il 6° e il 7° riguardano i permessi del gruppo, il 8°, il 9° e il 10° riguardano i permessi di tutti gli altri utenti.

- ✔ **r**: diritto di lettura (read);
- ✔ **w**: diritto di scrittura (write);
- ✔ **x**: diritto di esecuzione (execute);

Vediamo ora come cambiare i permessi ai file.
Il comando che ci interessa è **"chmod"** (man chmod) e si lancia nel seguente modo:

chmod permessi_da_assegnare file_interessato

Ci sono due modi per cambiare i permessi ai file, uno tramite lettere ed uno tramite numeri.

1) **Tramite numeri** (più comodo): si associa ad ognuno dei permessi un numero, in questa sequenza:
   - ✗ read --> 4
   - ✗ write --> 2
   - ✗ execute --> 1

   a questo punto consideriamo i permessi da assegnare come un numero a 3 cifre, dove le centinaia indicano l'utente, le decine indicano il gruppo e le unità indicano gli altri. Sommiamo per ogni categoria i numeri dei permessi che vogliamo assegnare e scriviamo il numero.
   Ad esempio il primo file, avrà i permessi settati a 777, infatti si ha che l'utente ha tutti i permessi, quindi 4 per la lettura, 2 per la scrittura e 1 per l'esecuzione, quindi 4+2+1=7 (centinaia) stessa cosa anche per gli altri.
   Se voglio dare ad un file i permessi si lettura a tutti, di esecuzione al gruppo e di scritture all'utente, che permessi gli devo passare?
   Risolviamo:
   Utente: scrittura e lettura --> 4+2=6
   Gruppo: lettura ed esecuzione --> 4 +1=5
   Altri: lettura --> 4
   sommando si avrà 654 ed il comando da dare sarà: ***chmod 651 nome_file***
   NB: se vogliamo togliere tutti i permessi, basta settare un campo a 0.

2) **Tramite lettere** (il mio preferito): bisogna tenersi a mente i significati di 4 lettere,
   - ✗ u: user;
   - ✗ g: group;
   - ✗ o: others;
   - ✗ a: all=u+g+o;

   a questo punto consideriamo i permessi che vogliamo dare al file.
   Se consideriamo sempre il primo file, avremo i parametri settati come **a+rwx**.
   Se voglio dare ad un file i permessi si lettura a tutti, di esecuzione al gruppo e di scritture all'utente, che permessi gli devo passare?
   Risolviamo:
   Utente: scrittura e lettura --> u+rw
   Gruppo: lettura ed esecuzione --> g+rx
   Altri: lettura --> o+r
   A questo punto stà la grossa differenza con il metodo precedente, poiché prima lanciando solamente una volta il comando chmod abbiamo cambiato i parametri, mentre questa volta ci toccherà lanciarlo ben 3 volte
   chmod u+rw nome_file
   chmod g+rx nome_file
   chmod o+r nome_file

Infine ci sono altri 3 tipo di permessi, meglio definiti come "Attributi", questi sono: SUID, SGID, STICKY e più precisamente sui file si imposta SUID, SGID, mentre sulle directory SGID, STICKY.
Questi attributi sono INDICATI nel primo campo (quello che abbiamo saltato prima):

- ✔ SUID (set user id): se esso è settato in un file eseguibile che contiene un file eseguibile, il sistema cambia temporaneamente lo User-ID dell'utente corrente con quello del creatore di quel file durante l'esecuzione del programma. Il cambiamento perdura SOLO per l'esecuzione del programma specificato contenuto nel file eseguibile. Un esempio di SUID settato è il comando **passwd** che consente di cambiare la password al proprio utente.
- ✔ SGID (set group id) valgono le stesse considerazioni fatte per il SUID, cambia solo il riferimento che è il gruppo e non l'utente.
- ✔ STICKY: se è settato consente di cancellare i file al suo interno solamente al proprietario, anche se l'utente ha i permessi di scrittura su quel file.

Il primo campo, a seconda del simbolo che ha impostato indica:

- ✔ d = Directory;
- ✔ - = File;
- ✔ l = Link;
- ✔ b = Block device;
- ✔ c = Character device;

Dove b e c risiedono all'interno della directory dei device (/dev)

Si definiscono con la cifra delle migliaia

1. **SETUID**:
   - **NOTAZIONE**: Viene identificato con **s** sul permesso di execute in User (se User non ha il permesso di execute si denota con **S**). Es. **-rwsrw-r--**
   - **USO**: Si imposta col codice numerico **4**. Es. # chmod **4**xxx nome_file
2. **SETGID**:
   - **NOTAZIONE**: Viene identificato con **s** sul permesso di execute in Gruppo (se Gruppo non ha il permesso di execute si denota con **S**). Es. **-rwxrwsr--**
3. **Sticky Bit**:
   - **NOTAZIONE**: Viene identificato con **t** sul permesso di execute in Other (se Other non ha il permesso di execute si denota con **T**). Es. **-rwxrw-r-t**
   - **USO**: Si imposta col codice numerico **1**. Es. # chmod **1**xxx nome_file

NB: quando create un nuovo file, i permessi di default sono settati a 644
IMPORTANTE: se i permessi di una cartella sono settati SOLO in lettura, non la potete attraversare, ma potete leggere quello che è contenuto al suo interno, mentre se avete settato solo i permessi di esecuzione, la potete attraversare senza leggere il suo contenuto, ossia se per esempio avere una cartella denominata “prova” con diritti 444, ed al suo interno avete un file denominato “primo_file” (644) e una directory “prova2” con permessi 576 contenente il file “secondo file” (644) (come illustrato nello schema sotto)

```
/home/utente
        |__ prova --> 444          ossia   dr--r--r--
                |__ primo_file --> 644    ossia   -rw-r--r--
                |__ prova2 --> 576        ossia   dr-xrwxrw-
                        |__secondo_file -->644    ossia   -rw-r--r--
```

Se noi diamo il comando: $ cd prova
riceveremo come output “Permission denied”, e lo stesso lo ottenete se proverete a dare il comando:
$ cd prova/prova2
se ora provate a cambiare i permessi di prova in 111, e date il comando:
$ cd prova
e successivamente
$ ls
noterete che il cambio di directory ora vi sarà possibile, mentre il comando ls no, in quanto ora avete il diritto di esecuzione della cartella, ma non avete quello di lettura quindi se volete cambiare ulteriormente directory dovete scrivere la directory di destinazione in modo completo (cd prova2), poiché non potendo leggere all'interno della directory il TAB risulta essere disabilitato