File System

In questo capitolo imparerai come lavorare con il filesystem.

Obiettivi : In questo capitolo, futuri amministratori Linux impareranno come:

✔ gestire le partizioni su disco;
✔ utilizzare LVM per un uso migliore delle risorse del disco;
✔ fornire gli utenti di un filesystem e gestire i diritti di accesso.

e anche scoprire:

✔ come è organizzata la struttura ad albero in Linux;
✔ i diversi tipi di file offerti e come lavorare con loro;

🏁 **hardware**, **disco**, **partizioni**, **lvm**, **linux**

Conoscenza: ⭐⭐
Complessità: ⭐⭐

Tempo di lettura: 20 minuti

Partizionamento

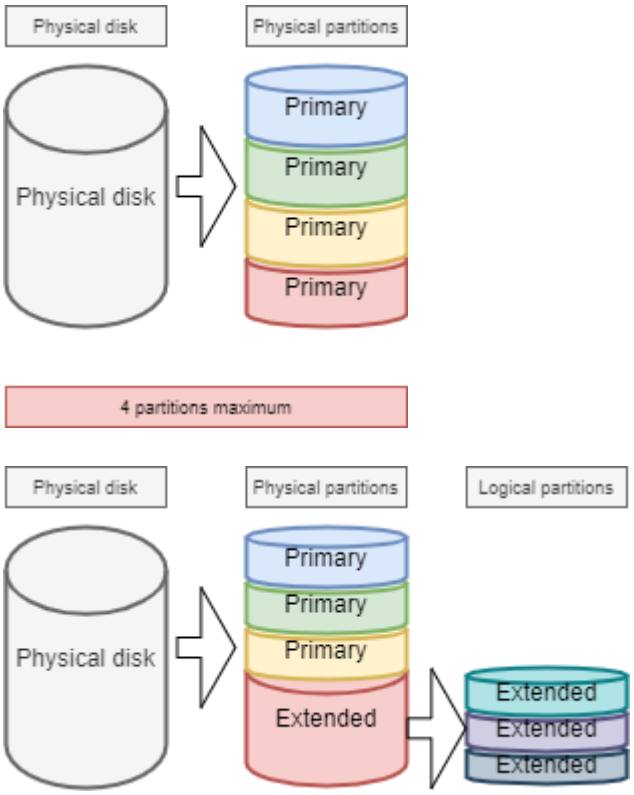
Il partizionamento consentirà l'installazione di diversi sistemi operativi perché è impossibile avere più sistemi conviventi sulla stessa unità logica. Il partizionamento consente anche la separazione dei dati logicamente (sicurezza, ottimizzazione dell'accesso, ...).

La divisione del disco fisico in volumi partizionati è registrata nella tabella delle partizioni, memorizzato nel primo settore del disco (MBR: ***Master Boot Record***).

Lo stesso disco fisico può essere diviso in un massimo di 4 partizioni:

- ***Primary*** (o main)
- ***Extended***

> **Attenzione**
>
> Ci può essere solo una partizione estesa per disco fisico. Al fine di beneficiare di ulteriori unità, la partizione estesa può essere suddivisa in partizioni logiche



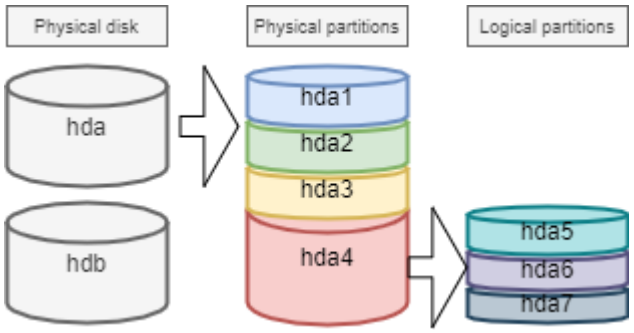
I ***dispositivi*** sono dei file che identificano i vari hardware rilevati dalla scheda madre. Questi file sono memorizzati in `/dev`. Il servizio che rileva i nuovi dispositivi e dà loro i nomi è chiamato ***udev***.

Sono identificati in base al loro tipo.

I dispositivi di archiviazione sono denominati ***hd*** per i dischi rigidi IDE e ***sd*** per gli altri supporti. Quindi arriva una lettera che inizia con ***a*** per il primo dispositivo, poi ***b***, ***c***, ...

Finalmente troveremo un numero che definisce il volume partizionato: ***1*** per la prima partizione primaria, ...

> **Attenzione**
>
> Attenzione, la partizione estesa, che non supporta un file system, ha ancora un numero.



Ci sono almeno due comandi per il partizionamento di un disco: `fdisk` e `cfdisk`. Entrambi i comandi hanno un menu interattivo. `cfdisk` è più affidabile e ottimizzato meglio, quindi è la scelta migliore.

L'unico motivo per usare `fdisk` è quando vuoi elencare tutti i dispositivi logici con l'opzione `-l`.

```
sudo fdisk -l
sudo fdisk -l /dev/sdc
sudo fdisk -l /dev/sdc2
```

comando `parted`

Il comando `parted` (***partition editor***) è in grado di partizionare un disco.

```
parted [-l] [device]
```

Ha anche una funzione di recupero in grado di riscrivere una tabella di partizione cancellata.

In un'interfaccia grafica, c'è l'applicazione completa `gparted`: ***G***nome ***PAR***tition ***ED***itor.

Il comando `gparted -l` elenca tutti i dispositivi logici su un computer.

Il comando `gparted` da solo tornerà a una modalità interattiva con le proprie opzioni interne:

- `help` o un comando errato visualizzerà queste opzioni.
- `print all` in questa modalità avrà lo stesso risultato di `gparted -l` sulla riga di comando.
- `quit` per ritornare al prompt.

comando `cfdisk`

Il comando `cfdisk` è usato per gestire le partizioni.

```
cfdisk device
```

Esempio:

```
$ sudo cfdisk /dev/sda
                                 Disk: /dev/sda
               Size: 16 GiB, 17179869184 bytes, 33554432 sectors
                     Label: dos, identifier: 0xcf173747
    Device        Boot       Start        End    Sectors   Size   Id Type
>>  /dev/sda1     *           2048    2099199    2097152     1G   83 Linux
    /dev/sda2              2099200   33554431   31455232    15G   8e Linux LVM

lqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqk
 x Partition type: Linux (83)
x
 x     Attributes: 80
x
 xFilesystem UUID: 54a1f5a7-b8fa-4747-a87c-2dd635914d60
x
 x     Filesystem: xfs
x
 x     Mountpoint: /boot (mounted)
x

mqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqj
     [Bootable]  [ Delete ]  [ Resize ]  [  Quit  ]  [  Type  ]  [  Help  ]
     [  Write ]  [  Dump  ]
```

La preparazione, senza ***LVM***, dei media fisici passano attraverso cinque passaggi:

- Impostazione del disco fisico;
- Partizionamento dei volumi (divisione geografica del disco, possibilità di installare diversi sistemi, ...);
- Creazione dei file system (Consente al sistema operativo di gestire i file, la struttura ad albero, i permessi, ...);
- Montaggio dei file system (registrazione del file system nella struttura ad albero);
- Gestione dell'accesso dell'utente..

Logical Volume Manager (LVM)

Logical **V**olume **M**anager (***LVM]***)

La gestione del volume crea uno strato astratto in cima alla memoria fisica, offrendo vantaggi rispetto all'utilizzo diretto della memoria fisica:

- Più flessibilità nella capienza del disco;
- Movimento dei dati online;
- Dischi in modalità ***stripe*** ;
- Volumi ridondanti (recopy);
- Istantanee del volume (***snapshot***).

Lo svantaggio è che se uno dei volumi fisici va fuori servizio, allora tutti i volumi logici che utilizzano questo volume fisico sono persi. Dovrai usare LVM su dischi RAID.

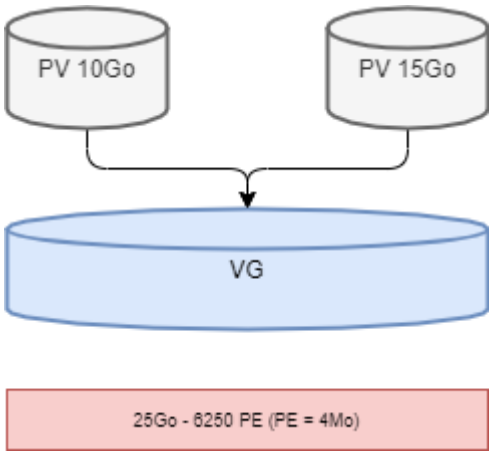
LVM è disponibile in Linux dalla versione del kernel 2.4.

> **Nota**
>
> LVM è gestita soltanto dal sistema operativo. Pertanto il ***BIOS*** ha bisogno di almeno una partizione senza LVM per avviare.

Gruppi di volumi

I volumi fisici ***PV Physical Volumes*** (dalle partizioni) sono combinati in gruppi di volumi ***VG***. Ogni ***VG*** rappresenta lo spazio su disco che può essere partizionato in ***LV Logical Volumes***. ***Extension*** è la più piccola unità di spazio a dimensione fissa che può essere assegnata.

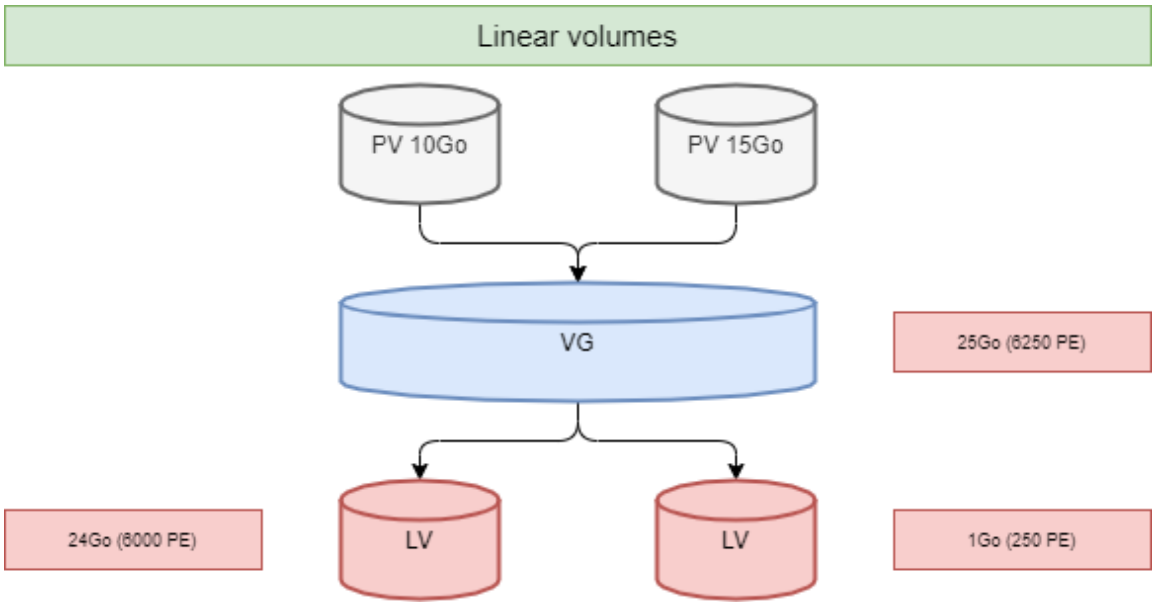
- **PE** : *Physical Extension*
- **LE** : *Logical Extension*

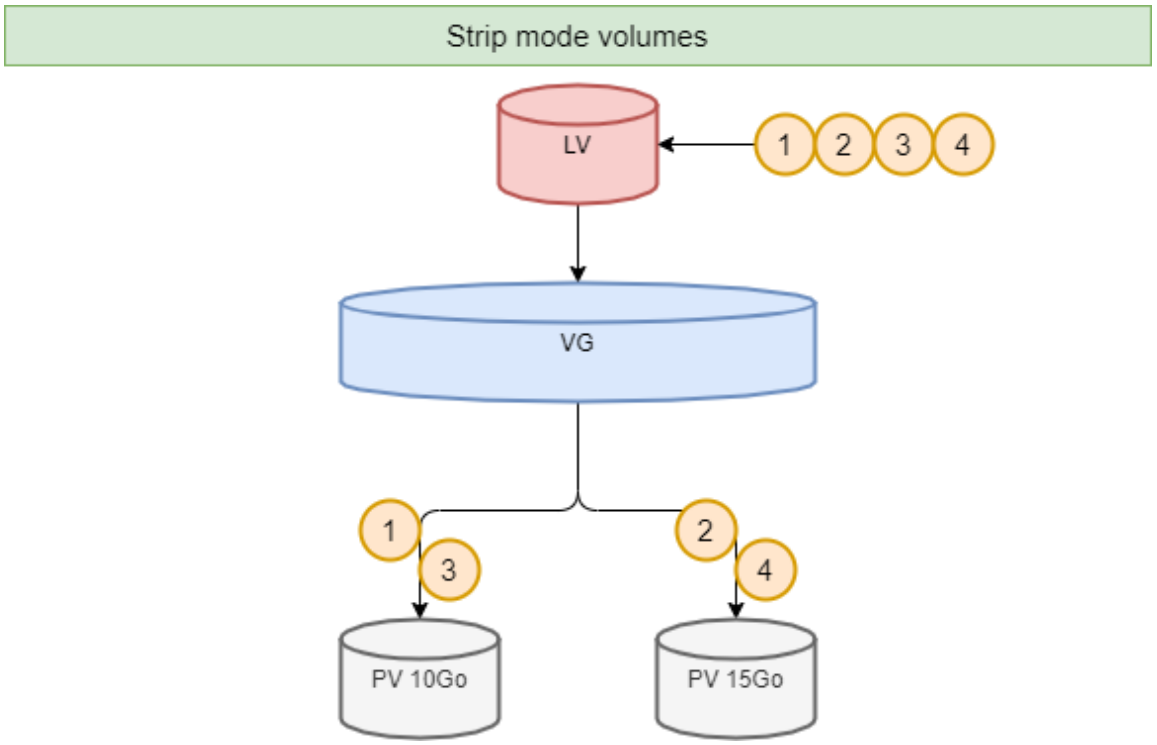


Volumi logici

Un gruppo di volumi, ***VG***, è diviso in volumi logici, ***LV***, offrendo diverse modalità operative:

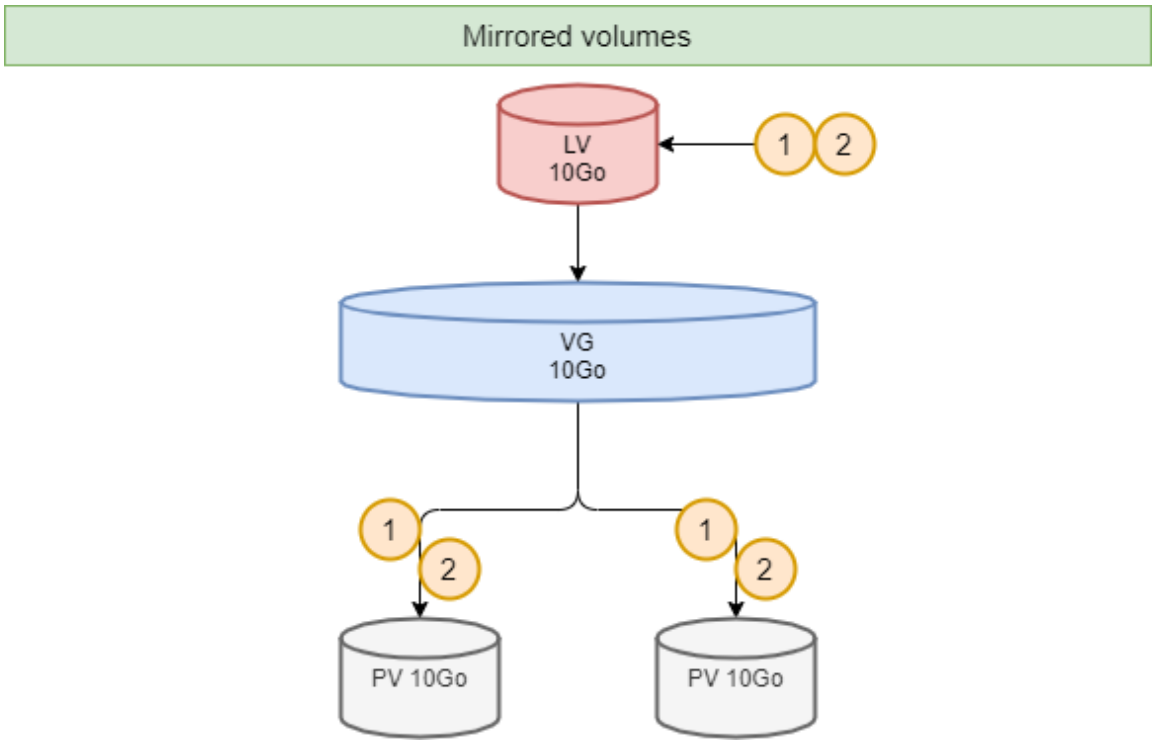
- Volumi lineari;
- Volumi in modalità ***stripe*** ;
- Volumi ridondanti.





Suggerimento

Striping_ migliora le prestazioni scrivendo i dati su un numero predeterminato di volumi fisici con una tecnica ***round-robin***.



I comandi LVM per la gestione del volume

comando `pvcreate`

Il comando `pvcreate` è usato per creare volumi fisici. Comprende partizioni Linux (o dischi) in volumi fisici.

```
pvcreate [-options] partition
```

Esempio:

```
[root]# pvcreate /dev/hdb1
pvcreate -- physical volume « /dev/hdb1 » successfully created
```

Puoi anche usare un disco intero (che facilita l'aumento della dimensione del disco in ambienti virtuali per esempio).

```
[root]# pvcreate /dev/hdb
pvcreate -- physical volume « /dev/hdb » successfully created
```

Opzione	Descrizione
`-f`	Costringe la creazione del volume (disco già trasformato in volume fisico).

comando `vgcreate`

Il comando `vgcreate` è usato per creare gruppi di volumi. Raggruppa uno o più volumi fisici in un gruppo di volumi.

```
vgcreate volume physical_volume [PV...]
```

Esempio:

```
[root]# vgcreate volume1 /dev/hdb1
…
vgcreate - volume group « volume1 » successfully created and activated
```

comando `lvcreate`

Il comando `lvcreate` crea volumi logici. Il file system viene quindi creato su questi volumi logici.

```
lvcreate -L size [-n name] VG_name
```

Esempio:

```
[root]# lvcreate -L 600M -n VolLog1 volume1
lvcreate -- logical volume « /dev/volume1/VolLog1 » successfully created
```

Opzione	Descrizione
`-L size`	Dimensione del volume logico in K, M o G.
`-n name`	Nome del LV. Un file speciale viene creato in `/dev/name_volume` con questo nome.

I comandi LVM per visualizzare le informazioni del volume

comando `pvdisplay`

Il comando `pvdisplay` consente di visualizzare informazioni sui volumi fisici.

```
pvdisplay /dev/PV_name
```

Esempio:

```
[root]# pvdisplay /dev/PV_name
```

comando `vgdisplay`

Il comando `vgdisplay` ti consente di visualizzare informazioni sui gruppi di volumi.

```
vgdisplay VG_name
```

Esempio:

```
[root]# vgdisplay volume1
```

comando `lvdisplay`

Il comando `lvdisplay` ti permette di visualizzare le informazioni sui volumi logici.

```
lvdisplay /dev/VG_name/LV_name
```

Esempio:

```
[root]# lvdisplay /dev/volume1/VolLog1
```

Preparazione dei media fisici

La preparazione con LVM del supporto fisico è suddiviso come segue:

- Impostazione del disco fisico
- Partizionamento dei volumi
- **Volume fisico LVM.**
- **Gruppi di volumi LVM.**
- **Volumi logici LVM.**
- Creazione di file systems
- Montaggio file systems
- Gestione accesso dell'utente.

Struttura di un file system

Un ***file system*** **FS** è responsabile delle seguenti azioni:

- Assicurare i permessi di accesso e modifica dei file;
- Manipolazione dei file: crea, legge, modifica ed elimina;
- Individuazione dei file sul disco;
- Gestione dello spazio della partizione.

Il sistema operativo Linux è in grado di utilizzare diversi file system (ext2, ext3, ext4, FAT16, FAT32, NTFS, HFS, BtrFS, JFS, XFS, ...).

comando `mkfs`

Il comando `mkfs` consente di creare un file system Linux.

```
mkfs [-t fstype] filesys
```

Esempio:

```
[root]# mkfs -t ext4 /dev/sda1
```

Opzione	Descrizione
`-t`	Indica il tipo di file system da utilizzare.

> **Attenzione**
>
> Senza un file system non è possibile usare lo spazio su disco.

Ogni file system ha una struttura identica su ciascuna partizione. Un **boot block** e **super block** inizializzato dal sistema e poi un **inode table** e una **data area** inizializzata dall'amministratore.

> **Nota**
>
> L'unica eccezione è la partizione di **swap**.

Blocco di avvio

Il **blocco di avvio** occupa il primo blocco sul disco ed è presente su tutte le partizioni. Contiene il programma che avvia e inizializza il sistema ed è quindi compilato solo per la partizione di avvio.

Super blocco

La dimensione della tabella **super blocco** è definito alla creazione. È presente su ogni partizione e contiene gli elementi necessari per il suo utilizzo.

Descrive il file system:

- Nome del Volume Logico;
- Nome del File System;
- Tipo di File System;
- Stato del File System;
- Dimensione del File System;
- Numero di blocchi liberi;
- Puntatore all'inizio della lista dei blocchi liberi;
- Dimensione della lista di inode;
- Numero e elenco di inodes liberi.

Una copia viene caricata nella memoria centrale non appena il sistema è inizializzato. Questa copia viene aggiornata non appena viene modificata e il sistema la salva periodicamente (comando `sync`).

Quando il sistema viene fermato, viene copiata anche questa tabella sull'hard disk.

Tabella degli inodes

La dimensione della **tabella inode** è definito alla sua creazione ed è memorizzato sulla partizione. Consiste di record, chiamati inodes, corrispondenti ai file creati. Ogni record contiene gli indirizzi dei blocchi di dati che costituiscono il file.

> **Nota**
>
> Un numero di inode è univoco all'interno di un file system.

Una copia viene caricata nella memoria centrale non appena il sistema è inizializzato. Questa copia viene aggiornata non appena viene modificata e il sistema la salva periodicamente (comando `sync`).

Quando il sistema viene fermato, viene copiata anche questa tabella sull'hard disk.

Un file è gestito dal suo numero di inode.

> **Nota**
>
> La dimensione della tabella inode determina il numero massimo di file che il FS può contenere.

Informazioni presenti nella ***tabella inode*** :

- Numero di inode;
- Tipo di file e permessi di accesso;
- Numero di identificazione del proprietario;
- Numero di identificazione del gruppo proprietario;
- Numero di collegamenti su questo file;
- Dimensione del file in byte;
- Data dell'ultimo accesso;
- Data dell'ultima modifica;
- Data dell'ultima modifica dell'inode (= creazione);
- Tabella di diversi puntatori (tabella a blocchi) ai blocchi logici contenenti i pezzi del file.

Area dati

La sua dimensione corrisponde al resto dello spazio disponibile nella partizione. Questa area contiene i cataloghi corrispondenti a ciascuna directory e i blocchi di dati corrispondenti ai contenuti dei file.

Al fine di garantire la coerenza del file system, un'immagine del superblocco e della tabella degli inode viene caricata in memoria (RAM) quando il sistema operativo è caricato, in modo che tutte le operazione di I/O siano fatte attraverso questa tabella di sistema. Quando l'utente crea o modifica i file, questa immagine in memoria viene aggiornata per prima. Il sistema operativo deve quindi aggiornare regolarmente il superblocco del disco logico (comando `sync`).

Quando il sistema viene fermato, viene copiata anche questa tabella sull'hard disk

> **Pericolo**
>
> In caso di arresto improvviso, il file system può perdere la sua coerenza e causare la perdita di dati.

Riparazione del file system

È possibile controllare la coerenza di un file system con il comando `fsck`.

In caso di errori, vengono proposte le soluzioni per riparare le incoerenze. Dopo la riparazione, i file che rimangono senza voci nella tabella degli inode sono allegati alla cartella `/lost+found` dell'unità logica.

comando `fsck`

Il comando `fsck` è uno strumento di controllo e riparazione di integrità in modalità console per i file system Linux.

```
fsck [-sACVRTNP] [ -t fstype ] filesys
```

Esempio:

```
[root]# fsck /dev/sda1
```

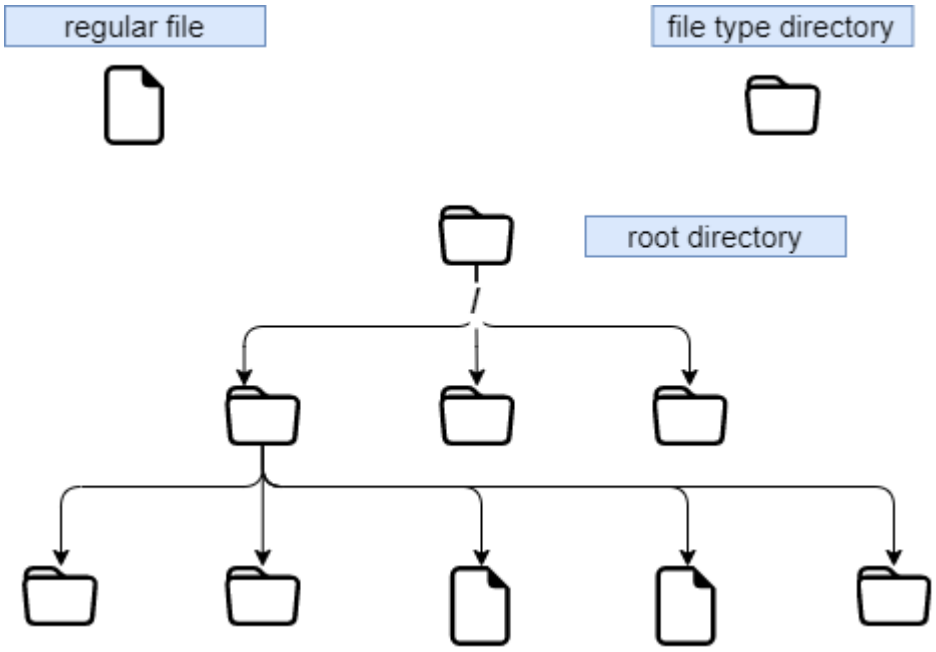
Per controllare la partizione root, è possibile creare un file `forcefsck` e riavviare o eseguire `shutdown` con l'opzione `-F`.

```
[root]# touch /forcefsck
[root]# reboot
or
[root]# shutdown -r -F now
```

> ⚠ **Attenzione**
>
> La partizione da controllare deve essere smontata.

Organizzazione di un file system

Per definizione, un file system è una struttura ad albero delle directory creata a partire da una directory principale (un dispositivo logico può contenere solo un file system).



> **Nota**
>
> In Linux tutto è un file.

Documento di testo, directory, file binario, partizione, risorse di rete, schermo, tastiera, Unix kernel, programma utente, ...

Linux segue il **FHS** (***Filesystems Hierarchy Standard***) (vedi `man hier`) che definisce i nomi delle cartelle e dei loro ruoli.

Directory	Osservazione	Abbreviazione di
`/`	Contiene directory speciali	
`/boot`	File relativi all'avvio del sistema	
`/sbin`	Comandi necessari per l'avvio e la riparazione del sistema	*system binaries*
`/bin`	Eseguibili dei comandi di base del sistema	*binaries*
`/usr/bin`	Comandi di amministrazione del sistema	
`/lib`	Librerie condivise e moduli del kernel	*libraries*

Directory	Osservazione	Abbreviazione di
`/usr`	Tutto ciò che non è necessario per il funzionamento minimo del sistema	*UNIX System Resources*
`/mnt`	Per il montaggio temporaneo dei file system	*mount*
`/media`	Per il montaggio dei supporti rimovibili	
`/root`	Directory di accesso dell'amministratore	
`/home`	Dati utente	
`/tmp`	File temporanei	*temporary*
`/dev`	File di dispositivo speciali	*device*
`/etc`	File di configurazione e script	*editable text configuration*
`/opt`	Specifica per applicazioni installate	*optional*
`/proc`	Sistema virtuale che rappresenta processi diversi	*processes*
`/var`	File di vario genere variabili.	*variables*

- Per montare o smontare, a livello dell'albero, non devi essere sotto il suo punto di montaggio.
- Il montaggio su una directory non vuota non elimina il contenuto. Viene solamente nascosto.
- Solo l'amministratore può montare i supporti.
- I punti di montaggio da montare automaticamente all'avvio devono essere inseriti in `/etc/fstab`.

file `/etc/fstab`

Il file `/etc/fstab` viene letto all'avvio del sistema e contiene i supporti da montare. Ogni file system da montare è descritto su una singola riga, I campi sono separati da spazi o tabulazioni.

> **Nota**
>
> Le linee sono lette in sequenza (`fsck` , `mount` , `umount`).

```
/dev/mapper/VolGroup-lv_root   /          ext4    defaults        1   1
UUID=46….92                    /boot      ext4    defaults        1   2
/dev/mapper/VolGroup-lv_swap   swap       swap    defaults        0   0
tmpfs                          /dev/shm   tmpfs   defaults        0   0
devpts                         /dev/pts   devpts  gid=5,mode=620  0   0
sysfs                          /sys       sysfs   defaults        0   0
proc                           /proc      proc    defaults        0   0
  1                              2          3       4             5   6
```

Colonna	Descrizione
1	Dispositivo di file system. (`/dev/sda1` , UUID=..., ...)
2	Nome del punto di montaggio, **absolute path** (tranne **swap**)
3	Tipo di filesystem. (ext4, swap, ...)
4	Opzioni speciali per il montaggio (`defaults` , `ro` , ...)
5	Abilita o disabilita la gestione del backup (0:niente backup, 1:backup)
6	Controllare l'ordine quando si controlla il file system con il comando `fsck` (0:nessun controllo, 1:priorità, 2:nessuna priorità)

Il comando `mount -a` consente di montare i nuovi punti di montaggio senza riavvio. Vengono quindi scritti nel file `/etc/mtab` che contiene i supporti correnti.

> **Attenzione**
>
> Solo i punti di montaggio elencati in `/etc/fstab` saranno montati al riavvio.

È possibile fare una copia del file `/etc/mtab` o copiare il suo contenuto in `/etc/fstab` .

Comandi di gestione del supporto

comando `mount`

Il comando `mount` permette di montare e visualizzare le unità logiche nell'albero.

```
mount [-option] [device] [directory]
```

Esempio:

```
[root]# mount /dev/sda7 /home
```

Opzione	Descrizione
`-n`	Montare senza scrivere in `/etc/mtab`.
`-t`	Per indicare il tipo di file system da utilizzare.
`-a`	Monta tutti i filesystem menzionati in `/etc/fstab`.
`-r`	Monta il file system in sola lettura (equivalente a `-o ro`).
`-w`	Monta il file system in lettura/scrittura, per impostazione predefinita. (equivalente `-o rw`).
`-o`	Argomento seguito da un elenco di opzioni separate da virgole (`remount`, `ro`, ...).

> **Nota**
>
> Il comando `mount` da solo mostra tutti i file system montati.

comando `umount`

Il comando `umount` è usato per smontare le unità logiche.

```
umount [-option] [device] [directory]
```

Esempio:

```
[root]# umount /home
[root]# umount /dev/sda7
```

Opzione	Descrizione
`-n`	Smontare senza scrivere in `/etc/mtab`.
`-r`	Se l'unmount fallisce, rimontare come sola lettura.
`-f`	Forza smontaggio.
`-a`	Smontare tutti i filesystem menzionati in `/etc/fstab`.

Nota

Durante lo smontaggio, non si deve rimanere sotto il punto di montaggio. Altrimenti, viene visualizzato il seguente messaggio di errore: `device is busy`.

Tipi di file

Come in qualsiasi sistema, è importante rispettare le regole di denominazione dei file per poter trovare la propria strada nella struttura ad albero e per la gestione dei file.

- I file sono codificati su 255 caratteri;
- Tutti i caratteri ASCII possono essere utilizzati;
- Le lettere maiuscole e minuscole sono differenziate;
- Nessuna nozione di estensione.

I gruppi di parole separati da spazi devono essere racchiusi tra virgolette:

```
[root]# mkdir "working dir"
```

Nota

Mentre non c'è nulla di tecnicamente sbagliato nella creazione di un file o una directory con uno spazio in esso, è generalmente una "pratica migliore" evitare questo e sostituire qualsiasi spazio con un underscore.

> **Nota**
>
> Il `.` all'inizio del nome del file serve solo a nasconderlo da un semplice `ls`.

> **Attenzione**
>
> Sotto Linux, l'estensione di un file non è un riferimento necessario per aprirlo o modificarlo. Tuttavia, può essere utile per l'utente.

Esempi di estensioni comuni:

- `.c` : file sorgente nel linguaggio C;
- `.h` : file di header in C and Fortran;
- `.o` : file oggetto nel linguaggio C;
- `.tar` : file di dati archiviato con l'utilità `tar` ;
- `.cpio` : file di dati archiviato con l'utilità `cpio` ;
- `.gz` : file di dati compresso con l'utilità `gzip` ;
- `.tgz` : file di dati archiviato con l'utilità `tar` e compresso con l'utilità `gzip` ;
- `.html` : pagina web.

Dettagli di un nome del file

```
[root]# ls -liah /usr/bin/passwd
266037 -rwsr-xr-x 1 root root 59K mars  22  2019 /usr/bin/passwd
1      2    3     4  5    6    7       8              9
```

Riga	Descrizione
`1`	Numero di inode
`2`	Tipo di file (primo carattere del blocco di 10)
`3`	Permessi di accesso (ultimi 9 caratteri del blocco di 10)
`4`	Numero di collegamenti (ordinari) o sottodirectory (directory)

Riga	Descrizione
`5`	Nome del proprietario
`6`	Nome del gruppo
`7`	Dimensione (byte, kilo, mega)
`8`	Data dell'ultima modifica
`9`	Nome del file

Diversi tipi di file

I seguenti tipi di file possono essere trovati su un sistema:

- Ordinari (testo, binari, ...);
- Directories;
- Speciali (stampanti, screens, ...);
- Collegamenti;
- Comunicazioni (tubes e socket).

File ordinari

Questi sono testo, programmmi (sorgente), eseguibile (dopo la compilazione) o dati (binari, ASCII) e file multimediali.

```
[root]# ls -l myfile
-rwxr-xr-x   1   root  root  26  nov  31  15:21 myfile
```

Il carattere `-` all'inizio dei permessi del gruppo dei (blocco di 10 caratteri) indica che è un tipo di file ordinario.

File di directory

I file di directory contengono riferimenti ad altri file.

Per impostazione predefinita in ogni directory sono presenti **.** e **..**.

- IL **.** rappresenta la posizione nell'albero.
- Il **..** rappresenta il padre della posizione corrente.

```
[root]# ls -l mydirectory
drwxr-xr-x   1   root  root  26  nov  31  15:21 mydirectory
```

La lettera `d` all'inizio del gruppo dei permessi indica che è un file di tipo directory.

File speciali

Per comunicare con le periferiche (hard disks, stampanti, ...), Linux utilizza i file di interfaccia chiamati file speciali (***device file*** o ***special file***). Consentono l'identificazione da parte delle periferiche.

Questi file sono speciali perché non contengono dati ma specificano la modalità di accesso per comunicare con il dispositivo.

Sono definiti in due modalità:

- modo **block**;
- modo **character**.

MODO BLOCK

Il file speciale **modo block** consente, utilizzando i buffer di sistema, di trasferire i dati sul dispositivo.

```
[root]# ls -l /dev/sda
brw-------   1   root  root  8, 0 jan 1 1970 /dev/sda
```

La lettera `b` all'inizio del gruppo dei permessi indica che è un file speciale **block**.

MODO CHARACTER

Il file speciale ***character mode*** viene utilizzato per trasferire i dati sul dispositivo come un flusso di un carattere alla volta senza utilizzare un buffer. Questi sono dispositivi come la stampante, screen o nastri DAT, ...

L'output standard è lo screen.

```
[root]# ls -l /dev/tty0
crw-------   1   root  root  8, 0 jan 1 1970 /dev/tty0
```

La lettera `c` all'inizio del gruppo dei permessi indica che è un file speciale di character.

File di comunicazione

Questi sono le pipe (***pipes***) e i file ***socket*** .

- I **files Pipe** passano le informazioni tra i processi con FIFO (***First In, First Out***). Un processo scrive informazioni transitorie nel file di ***pipe*** e un altro lo legge. Dopo averlo

letto, le informazioni non sono più accessibili.

- **Socket files** consentono la comunicazione bidirezionale intertrattativa (sui sistemi locali o remoti). Usano un ***inode*** del file system.

Files di Link

Questi file danno la possibilità di fornire diversi nomi logici allo stesso file fisico. È pertanto creato un nuovo punto di accesso al file.

Esistono due tipi di file link:

- Links fisici;
- Links simbolici.

LINK FISICI

Il file di collegamento e il file di origine hanno lo stesso numero di ***inode*** e il contatore dei collegamenti viene incrementato. Non è possibile collegare diverse directory o files da file system diversi.

> ⚠ **Attenzione**
>
> Se il file sorgente viene distrutto, il contatore viene decrementato e il file di collegamento accede ancora al file.

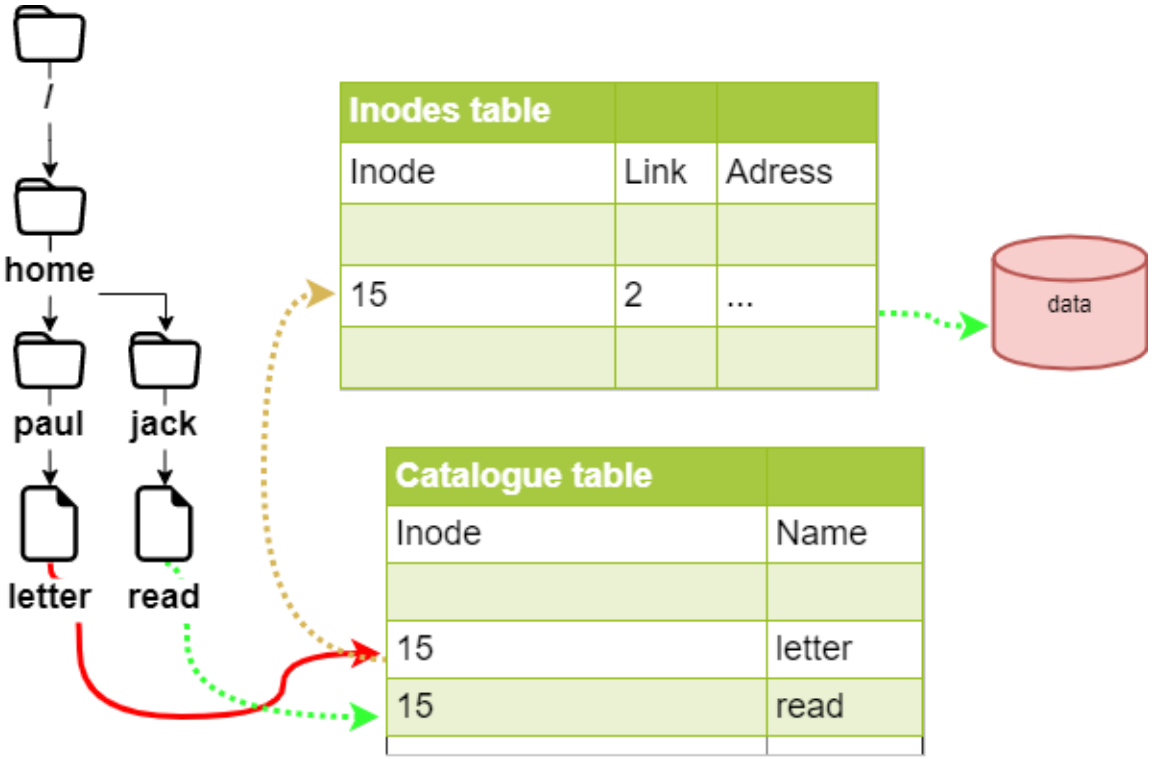
Commando `ln` per un link fisico

Il comando `ln` permette di creare i collegamenti fisici.

```
[root]# ls -li letter
666 -rwxr--r-- 1 root root … letter
```

```
[root]# ln /home/paul/letter /home/jack/read
```

```
[root]# ls -li /home/*/*
666 -rwxr--r-- 2 root root … letter
666 -rwxr--r-- 2 root root … read
```



Representation of a physical link

LINK SIMBOLICO

A differenza del collegamento fisico, il link simbolico coinvolge la creazione di un nuovo ***inode***. A livello di collegamento simbolico, solo un percorso è memorizzato nella tabella degli inode.

Il file creato contiene solo un'indicazione del percorso verso il file. Questa nozione non ha più le limitazioni dei collegamenti fisici ed è ora possibile collegare directory e file appartenenti a diversi file system.

> **Attenzione**
>
> Se il file sorgente viene distrutto, il file di collegamento non può più accedere al file.

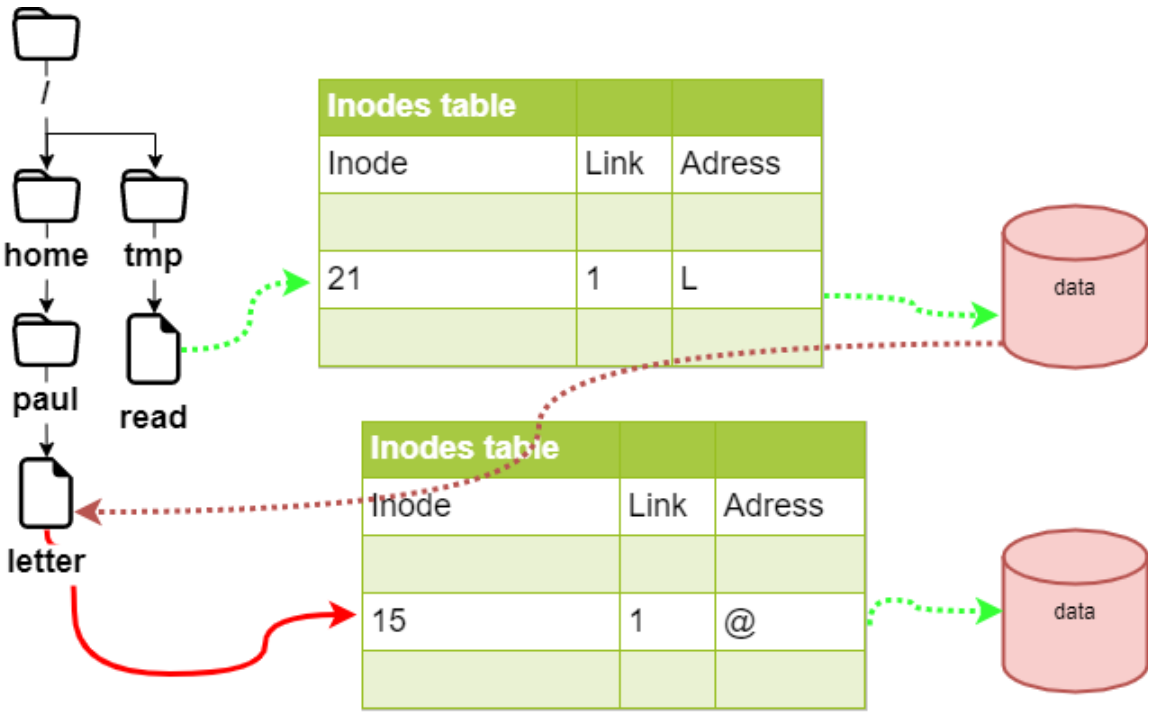
comando `ln` per un link simbolico

Il comando `ln` con l'argomento `-s` permette di creare collegamenti simbolici.

```
[root]# ls -li letter
666 -rwxr--r-- 1 root root … letter
```

```
[root]# ln -s /home/paul/letter /tmp/read
```

```
[root]# ls -li /home/paul/letter /tmp/read
666 -rwxr--r--- 1 root root … letter
678 lrwxrwxrwx 1 root root … read -> letter
```



Representation of a symbolic link

Attributi dei file

Linux è un sistema operativo multiutente in cui il controllo dell'accesso ai file è essenziale.

Questi controlli sono le funzioni di:

- autorizzazioni di accesso ai file ;
- utenti (***ugo Users Groups Others***).

Il comando `ls -l` permette di visualizzare gli attributi.

Ci sono 4 diritti di accesso ai file:

- **r**ead; (lettura)
- **w**rite; (scrittura)
- e**x**ecution; (esecuzione)

- **-** no right. (nessun diritto)

> **Attenzione**
>
> I diritti associati ai file differiscono da quelli associati alle directory (vedi sotto).

I tipi di utente associati ai diritti di accesso ai file sono:

- **u**ser_ (owner) ; (proprietario)
- **g**roup_ (owner group); (gruppo propretario)
- **o**thers (others users); (altri utenti)

In alcuni comandi è possibile designare tutti con **a** (***all***).

a = ugo

Permessi associati ai file ordinari

- **r**ead: Consente di leggere un file (`cat` , `less` , ...) e copiare un file (`cp` , ...).
- **w**rite: Consente la modifica del contenuto del file (`cat` , `>>` , `vim` , ...).
- e**x**ecute: Considera il file come un eseguibile e**X**ecutable (binario o script.).
- **-**: Nessuna autorizzazione.

> **Nota**
>
> Lo spostamento o la rinomina di un file dipende dai diritti della directory di destinazione. L'eliminazione di un file dipende dai diritti della directory padre.

Permessi associati alle directory

- **r**ead: Consente di leggere il contenuto di una directory (`ls -R`).
- **w**rite: Consente la modifica dei contenuti di una directory (`touch`) e consente la creazione e la cancellazione dei file se il permesso **x** è abilitato.
- e**x**ecute: Permette di discendere nella directory (`cd`).
- **-**: Nessun diritto.

Gestione degli attributi

La visualizzazione dei permessi viene eseguita con il comando `ls -l`. Sono gli ultimi 9 caratteri del blocco di 10. Più precisamente 3 volte 3 caratteri.

```
[root]# ls -l /tmp/myfile
-rwxrw-r-x  1  root  sys  ... /tmp/myfile
  1  2  3       4     5
```

Riga	Descrizione
1	Autorizzazioni del proprietario (**u**ser), qui `rwx`
2	Autorizzazioni del gruppo proprietario (**g**roup), qui `rw-`
3	Autorizzazioni degli altri utenti (**o**thers), qui `r-x`
4	Proprietario del file
5	Proprietario del gruppo del file

> **Nota**
>
> I permessi si applicano a **u**ser, **g**roup e **o**ther (**ugo**) a seconda del proprietario e del gruppo.

Per impostazione predefinita, il ***proprietario*** di un file è quello che lo crea. IL ***gruppo*** del file è il gruppo del proprietario che ha creato il file. Gli ***others*** sono quelli che non sono interessati dai casi precedenti.

Gli attributi sono cambiati con il comando `chmod`.

Solo l'amministratore e il proprietario di un file possono modificare i permessi di un file.

comando `chmod`

Il comando `chmod` consente di modificare le autorizzazioni di accesso a un file.

```
chmod [option] mode file
```

L'indicazione della modalità può essere una rappresentazione ottale (e.g. `744`) o una

rappresentazione simbolica ([`ugoa`][`+=-`][`rwxst`]).

Diverse operazioni simboliche possono essere separate da virgole

Esempio:

```
[root]# chmod -R u+rwx,g+wx,o-r /tmp/file1
[root]# chmod g=x,o-r /tmp/file2
[root]# chmod -R o=r /tmp/file3
```

```
[root]# ls -l /tmp/fic*
-rwxrwx--- 1 root root … /tmp/file1
-rwx--x--- 1 root root … /tmp/file2
-rwx--xr-- 1 root root … /tmp/file3
```

```
[root]# chmod 741 /tmp/file1
[root]# chmod -R 744 /tmp/file2
[root]# ls -l /tmp/fic*
-rwxr----x 1 root root … /tmp/file1
-rwxr--r-- 1 root root … /tmp/file2
```

Opzione	Osservazione
`-R`	Modifica ricorsivamente le autorizzazioni delle directory e dei loro contenuti.

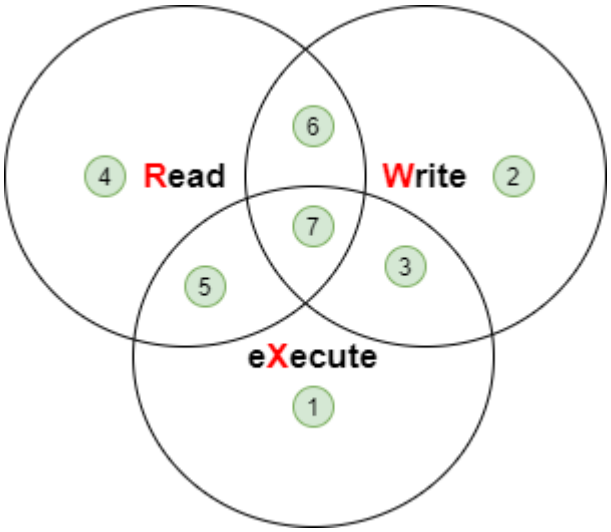
Ci sono due metodi per la realizzazione dei cambiamenti dei permessi:

- Il metodo **ottale**;
- Il metodo **simbolico**.

> ⚠ **Attenzione**
>
> I diritti dei file e delle directory non sono dissociati. Per alcune operazioni, sarà necessario conoscere i diritti della directory contenente il file. Un file protetto da scrittura può essere cancellato da un altro utente purché i permessi della directory che lo contengono consentano a questo utente di eseguire questa operazione.

Principio del metodo ottale

Ogni permesso ha un valore.




```
[root]# ls -l /tmp/myfile
-rwxrwxrwx  1  root  root  ... /tmp/myfile
```

user			group			other		
r	w	x	r	w	x	r	w	x
4	2	1	4	2	1	4	2	1

Rights 777

```
[root]# chmod 741 /tmp/myfile
-rwxr----x  1  root  root  ... /tmp/myfile
```

user			group			other		
r	w	x	r	-	-	-	-	x
4	2	1	4	-	-	-	-	1

Rights 741

Principio del metodo simbolico

Questo metodo può essere considerato come un'associazione "letterale" tra un tipo utente, un operatore e permesso.

Who?	
User	u
Group	g
Other	o
All	a

Operation	
add	+
remove	-
replace	=

Rights	
Read	r
Write	w
eXecute	x

```
[root]# chmod u+rwx,g+wx,o-r /tmp/myfile
[root]# chmod g=x,o-r /tmp/myfile
[root]# chmod o=r /tmp/myfile
```

```
[root]# ls -l /tmp/myfile
r--r-- 1 root root … /tmp/myfile
```

```
[root]# chmod u+rwx,g+wx,o-r /tmp/myfile
```

```
[root]# ls -l /tmp/myfile
-rwxrwx--- 1 root root … /tmp/myfile
```

Permessi speciali

Oltre ai permessi fondamentali (`rwx`), ci sono i permessi particolari:

- **set-user-ID** (*SUID]*)
- **set-group-ID** (*SGID]*)
- **sticky-bit**

Come con i permessi fondamentali, i permessi particolari hanno ciascuno un valore. Questo valore è posizionato prima del set dei diritti di `ugo` .

> **Pericolo**
>
> `S` , `S` e `T` in lettere maiuscole **se il diritto non esiste**.

Il sticky-bit

Una delle peculiarità dei diritti in Linux è che il diritto di scrivere in una directory consente anche la cancellazione di ***tutti*** i files, proprietario o no.

IL ***sticky-bit*** impostato sulla directory consentirà agli utenti di eliminare solo i file che possiedono. Questo è il caso di base per la directory `/tmp` .

L'impostazione del ***sticky-bit*** può essere fatto come segue:

Metodo ottale:

```
[root]# chmod 1777 directory
```

Metodo simbolico:

```
[root]# chmod o+t directory
```

Verifica:

```
[root]# ls -l
drwxrwxrwt … directory
```

SUID e SGID su un comando

Questi permessi consentono l'esecuzione di un comando in base ai permessi impostati sul comando, e non secondo i permessi dell'utente.

Il comando viene eseguito con l'identità del proprietario (***SUID***) o del gruppo (***SGID***) del comando.

> **Nota**
>
> L'identità dell'utente che richiede l'esecuzione dell'ordine non viene più presa in considerazione.
>
> Questa è una possibilità aggiuntiva dei permessi di accesso assegnata a un utente quando è necessario per loro avere gli stessi permessi del proprietario di un file o di quelli del gruppo in questione.

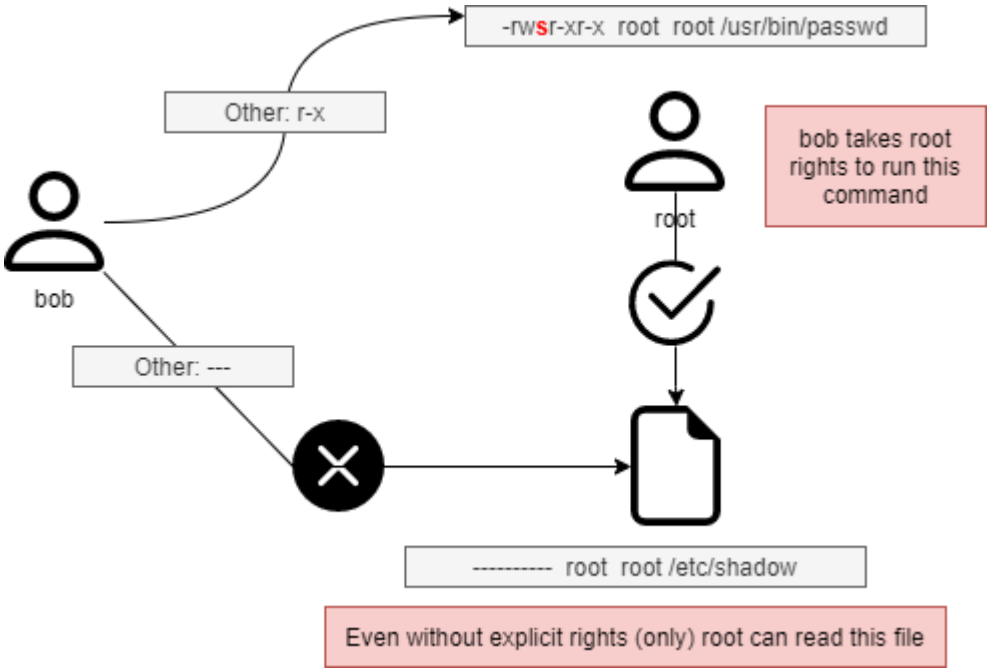
Infatti, un utente potrebbe deve eseguire un programma (di solito un'utilità di sistema) ma non ha i permessi di accesso necessari. Impostando i permessi appropriati (**s** a livello proprietario e/o a livello di gruppo), l'utente del programma ha, per il tempo della sua esecuzione, l'identità del proprietario (o quella del gruppo) del programma.

Esempio:

Il file `/usr/bin/passwd` è un file eseguibile (un comando) con un ***SUID***.

Quando l'utente. ***bob*** lo esegue, dovrà accedere al file `/etc/shadow` , ma le autorizzazioni su questo file non permettono a ***bob*** di accedervi.

Avere un ***suid*** su questo comando, `/usr/bin/passwd` , sarà eseguito con il ***UID*** di root e il ***GID*** di ***root***. Quest'ultimo è il proprietario del file `/etc/shadow` , e avrà i permessi di lettura.



L'impostazione di ***SUID*** e ***SGID*** può essere fatto come sotto con il comando `chmod`:

Metodo ottale:

```
[root]# chmod 4777 command1
[root]# chmod 2777 command2
```

Metodo simbolico:

```
[root]# chmod u+s command1
[root]# chmod g+s command2
```

Verifica:

```
[root]# ls -l
-rwsrwxrwx … command1
-rwxrwsrwx … command2
```

> **Attenzione**
>
> Non è possibile passare il ***SUID*** o ***SGID*** ad uno script di shell. Il sistema non lo consente perché è troppo pericoloso per la sicurezza!

SGID su un file

In una directory con il permesso ***SGID***, qualsiasi file creato erediterà il gruppo che possiede la

directory anziché quella dell'utente di creazione.

Esempio:

```
[rockstar] $ ls -ld /data/
drwxrwsr-x 2 root users 4096 26 oct. 19:43 /data
```

```
[rockstar] $ touch /data/test_sgid /tmp/fic_reference
```

```
[rockstar] $ ls -ld /data/test_sgid /tmp/fic_reference
-rw-r--r--. 1 rockstar users 0 26 oct. 19:43 /data/test_sgid <1>
-rw-r--r--. 1 rockstar rockstar 0 26 oct. 19:43  /tmp/fic_ref
```

<1> Il file `test_sgid` eredita il proprietario del gruppo della sua cartella `/data` (in questo caso `users`) Qualunque sia il gruppo principale dell'utente `rockstar`.

Permessi predefiniti e mascheramento

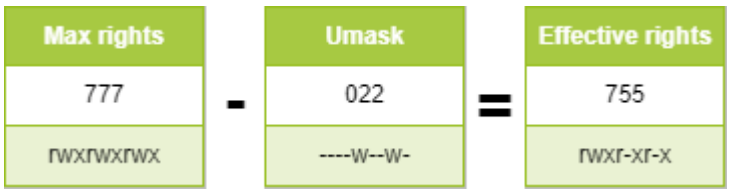
Quando viene creato un file o una directory, ha già le autorizzazioni.

- Per una directory.: `rwxr-xr-x` or *755*.
- Per un file: `rw-r-r-` or *644*.

Questo comportamento è definito dalla **maschera predefinita**.

Il principio è rimuovere il valore definito dalla maschera ai massimi permessi senza i permessi di esecuzione.

Per una directory :



Per un file, i diritti di esecuzione vengono rimossi:

comando `umask`

Il comando `umask` ti consente di visualizzare e modificare la maschera.

```
umask [option] [mode]
```

Esempio:

```
$ umask 033
$ umask
0033
$ umask -S
u=rwx,g=r,o=r
$ touch umask_033
$ ls -la  umask_033
-rw-r--r-- 1 rockstar rockstar 0 nov.   4 16:44 umask_033
$ umask 025
$ umask -S
u=rwx,g=rx,o=w
$ touch umask_025
$ ls -la  umask_025
-rw-r---w- 1 rockstar rockstar 0 nov.   4 16:44 umask_025
```

Opzione	Descrizione
`-S`	Visualizzazione simbolica dei permessi dei file.

Attenzione

`umask` non influisce sui file esistenti.

Nota

`umask` modifica la maschera fino alla disconnessione.

Per mantenere il valore, devi modificare i seguenti file del profilo: Per tutti gli utenti:

- `/etc/profile`
- `/etc/bashrc`

Per un determinato utente:

- `~/.bashrc`

Attenzione

`umask -S` visualizza i diritti del file (senza il diritto di esecuzione) dei file che verranno creati. Quindi non è la visualizzazione della maschera utilizzata per sottrarre il valore massimo.

Suggerimento

Il comando `umask` è un comando ***bash***, (un `type umask` restituisce `umask is a shell primitive`) devi cercare `umask` in `man bash`.

Ultimo aggiornamento: 5 gennaio 2022