

Navigare nel Filesystem Linux e Gestione dei Permessi

Prof. Fedeli Massimo

Sommario

Questo documento fornisce una guida completa alla navigazione del filesystem Linux, alla comprensione della sua struttura gerarchica, all'utilizzo dei comandi di base per la gestione di file e directory, e alla gestione dei permessi e della proprietà dei file. Gli argomenti trattati includono la struttura delle directory, i metacaratteri della shell, i permessi dei file, e i comandi fondamentali per operare sul filesystem.

Indice

1	Struttura del Filesystem Linux	3
1.1	Organizzazione Gerarchica	3
1.2	Directory Principali del Sistema	3
1.3	Differenze con i Filesystem Windows	4
2	Comandi di Base per il Filesystem	4
2.1	Navigazione tra Directory	4
2.1.1	Identificazione delle Directory	4
2.2	Creazione e Gestione di Directory	4
2.3	Comandi Principali	5
3	Metacaratteri e Operatori	5
3.1	Metacaratteri per la Corrispondenza dei File	5
3.1.1	Esempi di Pattern Matching	6
3.2	Metacaratteri di Redirezione	6
3.2.1	Esempi di Redirezione	6
3.3	Espansione delle Parentesi Graffe	7
4	Elencare File e Directory	7
4.1	Il Comando ls	7
4.2	Interpretazione dell'Output di ls -l	7
4.3	Tipi di File	7
5	Comprensione dei Permessi e della Proprietà	8
5.1	Struttura dei Permessi	8
5.2	Significato dei Permessi	8
5.3	Modifica dei Permessi con chmod (numeri)	8
5.3.1	Esempi con chmod Numerici	9
5.4	Modifica dei Permessi con chmod (lettere)	9
5.4.1	Esempi con chmod Letterali	9
5.5	Impostare i Permessi di Default con umask	10
5.6	Cambiare la Proprietà dei File	11
6	Spostare, Copiare e Rimuovere File	11
6.1	Il Comando mv (Move)	11
6.2	Il Comando cp (Copy)	11
6.3	Il Comando rm (Remove)	11
7	Permessi Speciali	12
7.1	Set UID e Set GID	12
7.2	Sticky Bit	12
8	Comandi Avanzati di Elenco	12
8.1	Opzioni Utili di ls	12
8.2	Esempi Combinati	12
9	Conclusioni	13
9.1	Best Practices	13
10	Riferimenti Utili	14

1 Struttura del Filesystem Linux

1.1 Organizzazione Gerarchica

Il filesystem di Linux è organizzato come una gerarchia di directory che inizia dalla directory radice (/). A differenza dei sistemi Windows che utilizzano lettere di unità separate (C:, D:, ecc.), Linux integra tutti i dispositivi di storage in un'unica struttura ad albero.

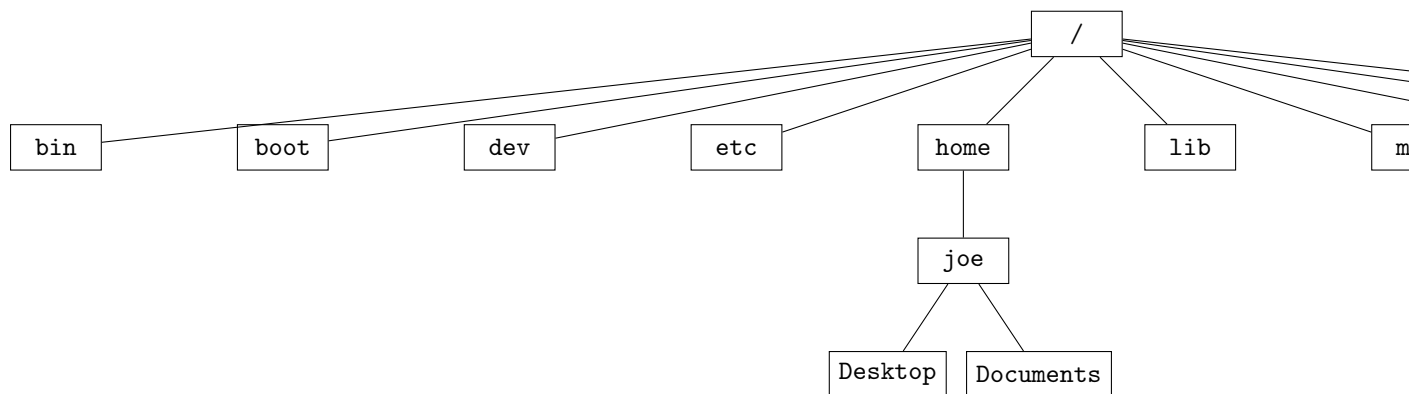


Figura 1: Struttura gerarchica del filesystem Linux

1.2 Directory Principali del Sistema

Directory	Descrizione
/bin	Contiene i comandi utente comuni di Linux, come <code>ls</code> , <code>sort</code> , <code>date</code> e <code>chmod</code> .
/boot	Contiene il kernel Linux avviabile, il disco RAM iniziale e i file di configurazione del bootloader (GRUB).
/dev	Contiene file che rappresentano punti di accesso ai dispositivi sul sistema, inclusi terminali (<code>tty*</code>), dischi rigidi (<code>hd*</code> o <code>sd*</code>), RAM (<code>ram*</code>) e CD-ROM (<code>cd*</code>).
/etc	Contiene i file di configurazione amministrativa. La maggior parte di questi file sono file di testo normale che possono essere modificati con qualsiasi editor di testo.
/home	Contiene le directory assegnate a ciascun utente regolare con un account di accesso. L'utente root è un'eccezione, utilizzando <code>/root</code> come propria home directory.
/media	Fornisce una posizione standard per il montaggio automatico di dispositivi (in particolare supporti rimovibili). Se il supporto ha un nome di volume, tale nome viene tipicamente utilizzato come punto di montaggio.
/lib	Contiene le librerie condivise necessarie alle applicazioni in <code>/bin</code> e <code>/sbin</code> per avviare il sistema.
/mnt	Un punto di montaggio comune per molti dispositivi. Alcune persone lo utilizzano ancora per montare temporaneamente filesystem locali o remoti.
/opt	Struttura di directory disponibile per memorizzare software aggiuntivo.
/proc	Contiene informazioni sulle risorse di sistema.

Directory	Descrizione
/root	Rappresenta la home directory dell'utente root. Non risiede sotto /home per motivi di sicurezza.
/sbin	Contiene comandi amministrativi e processi daemon.
/sys	Contiene parametri per la gestione dello storage a blocchi e dei cgroups.
/tmp	Contiene file temporanei utilizzati dalle applicazioni.
/usr	Contiene documentazione utente, giochi, file grafici (X11), librerie e vari altri comandi e file che non sono necessari durante il processo di avvio.
/var	Contiene directory di dati utilizzati da varie applicazioni. Include file condivisi come server FTP (/var/ftp) o web server (/var/www), tutti i file di log di sistema (/var/log) e file di spool in /var/spool.

1.3 Differenze con i Filesystem Windows

- **Lettere di unità:** In Windows, lettere di unità diverse rappresentano dispositivi di storage diversi. In Linux, tutti i dispositivi sono collegati alla gerarchia del filesystem.
- **Separatori:** Linux utilizza la barra normale (/) invece della barra rovesciata (\) per separare i nomi di directory. Esempio: /home/joe invece di C:\home\joe.
- **Estensioni file:** In DOS i nomi file hanno quasi sempre suffissi (.txt, .docx). In Linux, le estensioni a tre caratteri non hanno significato obbligatorio, anche se possono essere utili per identificare il tipo di file.
- **Permessi:** Ogni file e directory in Linux ha permessi e proprietà associati. Windows ha aggiunto queste funzionalità solo nelle versioni successive.

2 Comandi di Base per il Filesystem

2.1 Navigazione tra Directory

Il comando `cd` (change directory) è uno dei comandi più basilari utilizzati dalla shell.

```
$ cd /usr/share/
$ pwd
/usr/share
$ cd doc
$ pwd
/usr/share/doc
$ cd
$ pwd
/home/chris
```

2.1.1 Identificazione delle Directory

2.2 Creazione e Gestione di Directory

```
# Creare una nuova directory
$ mkdir test

# Verificare i permessi della directory
```

Simbolo	Significato
\$HOME	Variabile d'ambiente che memorizza il nome della home directory
~	Tilde rappresenta la home directory
.	Un singolo punto riferisce alla directory corrente
..	Due punti riferiscono alla directory immediatamente superiore
\$PWD	Variabile d'ambiente che riferisce alla directory di lavoro corrente
\$OLDPWD	Variabile d'ambiente che riferisce alla directory di lavoro precedente

Tabella 2: Simboli per identificare le directory

```
$ ls -ld test
drwxr-xr-x 2 joe sales 1024 Jan 24 12:17 test

# Cambiare i permessi
$ chmod 700 test

# Creare strutture di directory multiple
$ mkdir -p $HOME/test/documents/memos/
```

2.3 Comandi Principali

Comando	Risultato
cd	Cambia a un'altra directory
pwd	Stampa il nome della directory di lavoro corrente
mkdir	Crea una directory
chmod	Cambia i permessi su un file o directory
ls	Elenca il contenuto di una directory

Tabella 3: Comandi fondamentali per file e directory

3 Metacaratteri e Operatori

3.1 Metacaratteri per la Corrispondenza dei File

La shell bash consente di utilizzare metacaratteri per risparmiare pressioni di tasti e riferirsi facilmente a gruppi di file.

Metacarattere	Funzione
*	Corrisponde a qualsiasi numero di caratteri
?	Corrisponde a un singolo carattere
[...]	Corrisponde a uno qualsiasi dei caratteri tra parentesi, che possono includere un intervallo separato da trattino

Tabella 4: Metacaratteri per la corrispondenza dei file

3.1.1 Esempi di Pattern Matching

```
$ touch apple banana grape grapefruit watermelon

# Corrispondenza con asterisco
$ ls a*
apple

$ ls g*
grape grapefruit

$ ls *e*
apple grape grapefruit watermelon

# Corrispondenza con punto interrogativo
$ ls ???e
apple grape

$ ls g???e*
grape grapefruit

# Corrispondenza con parentesi quadre
$ ls [abw]*
apple banana watermelon

$ ls [a-g]*
apple banana grape grapefruit
```

3.2 Metacaratteri di Redirezione

Carattere	Funzione
<	Dirige il contenuto di un file al comando
>	Dirige l'output standard di un comando a un file (sovrascrive)
2>	Dirige l'errore standard (messaggi di errore) al file
&>	Dirige sia l'output standard che l'errore standard al file
»	Dirige l'output di un comando a un file, aggiungendolo alla fine

Tabella 5: Metacaratteri di redirezione dei file

3.2.1 Esempi di Redirezione

```
# Inviare il contenuto di un file via mail
$ mail root < ~/.bashrc

# Salvare l'output in un file
$ man chmod | col -b > /tmp/chmod

# Aggiungere testo a un file
$ echo "Ho finito il progetto il $(date)" >> ~/progetti
```

3.3 Espansione delle Parentesi Graffe

```
# Creare file multipli
$ touch memo{1,2,3,4,5}
$ ls
memo1 memo2 memo3 memo4 memo5

# Combinazioni multiple
$ touch {John,Bill,Sally}-{Breakfast,Lunch,Dinner}
$ ls
Bill-Breakfast  Bill-Dinner    John-Breakfast  John-Dinner
Bill-Lunch      John-Lunch     Sally-Breakfast Sally-Dinner
                Sally-Lunch

# Utilizzo di intervalli
$ touch {a..f}{1..5}
$ ls
a1 a2 a3 a4 a5 b1 b2 b3 b4 b5 c1 c2 c3 c4 c5
d1 d2 d3 d4 d5 e1 e2 e3 e4 e5 f1 f2 f3 f4 f5
```

4 Elencare File e Directory

4.1 Il Comando ls

Il comando `ls` è il comando più comune per elencare informazioni su file e directory.

```
# Elenco base
$ ls

# Elenco lungo con dettagli
$ ls -l

# Mostrare tutti i file inclusi quelli nascosti
$ ls -a

# Ordinare per tempo di modifica
$ ls -t

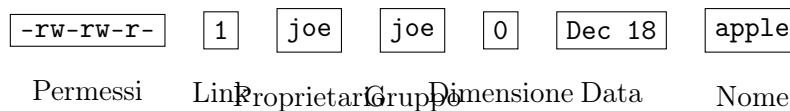
# Aggiungere indicatori di tipo file
$ ls -F
apple  banana  docs/   grape  grapefruit  script.sh*  watermelon
```

4.2 Interpretazione dell'Output di `ls -l`

```
$ ls -l
total 4
-rw-rw-r--. 1 joe joe    0 Dec 18 13:38 apple
lrwxrwxrwx. 1 joe joe    5 Dec 18 13:46 pointer_to_apple -> apple
-rwxr-xr-x. 1 joe joe    0 Dec 18 13:37 scriptx.sh
drwxrwxr-x. 2 joe joe 4096 Dec 18 13:38 Stuff
```

4.3 Tipi di File

- - = File regolare

Figura 2: Componenti dell'output di `ls -l`

- **d** = Directory
- **l** = Link simbolico
- **b** = Dispositivo a blocchi
- **c** = Dispositivo a caratteri
- **s** = Socket
- **p** = Named pipe

5 Comprensione dei Permessi e della Proprietà

5.1 Struttura dei Permessi

I permessi in Linux sono rappresentati da nove bit che definiscono chi può leggere, scrivere o eseguire un file.

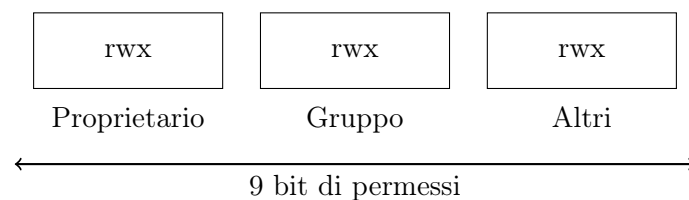


Figura 3: Struttura dei permessi dei file

5.2 Significato dei Permessi

Permesso	File	Directory
Read (r)	Visualizzare il contenuto del file	Vedere quali file e sottodirectory contiene
Write (w)	Modificare il contenuto del file, rinominarlo o eliminarlo	Aggiungere o rimuovere file o sottodirectory
Execute (x)	Eseguire il file come programma	Cambiare nella directory come directory corrente, cercare attraverso la directory o eseguire un programma dalla directory

Tabella 6: Significato dei permessi per file e directory

5.3 Modifica dei Permessi con `chmod` (numeri)

Ogni permesso è assegnato a un numero: `r=4`, `w=2`, `x=1`. Il numero totale di ciascun set stabilisce i permessi.

Numero	Binario	Permessi	Descrizione
0	000	—	Nessun permesso
1	001	-x	Solo esecuzione
2	010	-w-	Solo scrittura
3	011	-wx	Scrittura ed esecuzione
4	100	r-	Solo lettura
5	101	r-x	Lettura ed esecuzione
6	110	rw-	Lettura e scrittura
7	111	rwX	Tutti i permessi

Tabella 7: Valori numerici dei permessi

5.3.1 Esempi con chmod Numerici

```
# Permessi completi per tutti: rwxrwxrwx
$ chmod 777 file

# Proprietario completo, altri solo lettura ed esecuzione: rwxr-xr-x
$ chmod 755 file

# Proprietario lettura/scrittura, altri solo lettura: rw-r--r--
$ chmod 644 file

# Nessun permesso: -----
$ chmod 000 file

# Applicare ricorsivamente
$ chmod -R 755 $HOME/myapps
```

5.4 Modifica dei Permessi con chmod (lettere)

È possibile attivare e disattivare i permessi usando i segni più (+) e meno (-) insieme a lettere.

- u = user (proprietario)
- g = group (gruppo)
- o = others (altri)
- a = all (tutti)
- r = read (lettura)
- w = write (scrittura)
- x = execute (esecuzione)

5.4.1 Esempi con chmod Letterali

```
# Rimuovere scrittura per tutti: r-xr-xr-x
$ chmod a-w file

# Rimuovere esecuzione per altri: rwxrwxrw-
$ chmod o-x file
```

```
# Rimuovere tutti i permessi per gruppo e altri: rwx-----
$ chmod go-rwx file

# Aggiungere lettura/scrittura per proprietario: rw-----
$ chmod u+rw file

# Aggiungere esecuzione per tutti: --x--x--x
$ chmod a+x file

# Aggiungere lettura ed esecuzione per proprietario e gruppo: r-xr-x---
$ chmod ug+rx file

# Rimuovere scrittura per altri ricorsivamente
$ chmod -R o-w $HOME/myapps
```

5.5 Impostare i Permessi di Default con umask

Il comando `umask` determina i permessi predefiniti per nuovi file e directory.

```
# Visualizzare il valore umask corrente
$ umask
0002

# Esempi di diversi valori umask
$ umask 777 ; touch file01 ; mkdir dir01 ; ls -ld file01 dir01
d------. 2 joe joe 6 Dec 19 11:03 dir01
------. 1 joe joe 0 Dec 19 11:02 file01

$ umask 000 ; touch file02 ; mkdir dir02 ; ls -ld file02 dir02
drwxrwxrwx. 2 joe joe 6 Dec 19 11:00 dir02/
-rw-rw-rw-. 1 joe joe 0 Dec 19 10:59 file02

$ umask 022 ; touch file03 ; mkdir dir03 ; ls -ld file03 dir03
drwxr-xr-x. 2 joe joe 6 Dec 19 11:07 dir03
-rw-r--r--. 1 joe joe 0 Dec 19 11:07 file03
```

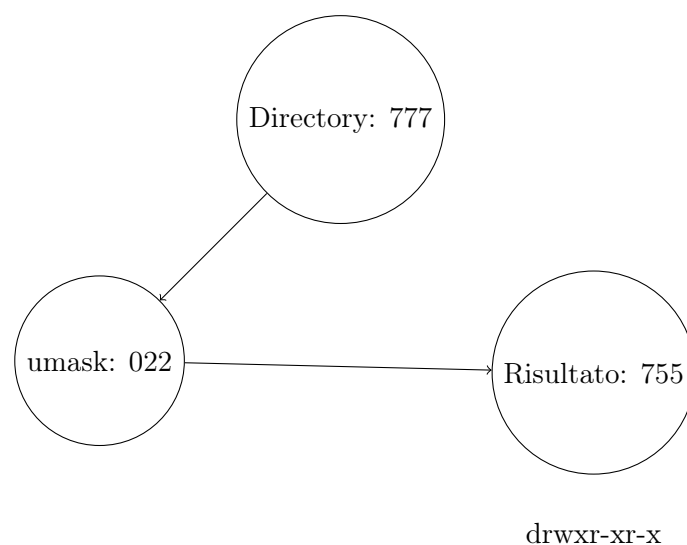


Figura 4: Calcolo dei permessi con umask

5.6 Cambiare la Proprietà dei File

Come utente root, è possibile cambiare la proprietà di file e directory.

```
# Cambiare solo il proprietario
$ chown joe /home/joe/memo.txt

# Cambiare proprietario e gruppo
$ chown joe:joe /home/joe/memo.txt

# Applicare ricorsivamente
$ chown -R joe:joe /media/myusb
```

6 Spostare, Copiare e Rimuovere File

6.1 Il Comando mv (Move)

```
# Rinominare un file
$ mv abc def

# Spostare un file nella home directory
$ mv abc ~

# Spostare una directory
$ mv /home/joe/mymemos/ /home/joe/Documents/

$ alias mv
alias mv='mv -i'
```

6.2 Il Comando cp (Copy)

```
# Copiare un file
$ cp abc def

# Copiare nella home directory
$ cp abc ~

# Copiare ricorsivamente una directory
$ cp -r /usr/share/doc/bash-completion* /tmp/a/

# Copiare preservando attributi (archive)
$ cp -ra /usr/share/doc/bash-completion* /tmp/b/
```

6.3 Il Comando rm (Remove)

```
# Rimuovere un file
$ rm abc

# Rimuovere tutti i file nella directory corrente
$ rm *

# Rimuovere una directory vuota
$ rmdir /home/joe/nothing/
```

```
# Rimuovere ricorsivamente con prompt
$ rm -r /home/joe/bigdir/

# Rimuovere forzatamente senza prompt
$ rm -rf /home/joe/hugedir/
```

7 Permessi Speciali

7.1 Set UID e Set GID

Quando un file eseguibile ha il bit Set UID o Set GID attivato, viene eseguito con i permessi del proprietario o del gruppo del file, non dell'utente che lo esegue.

```
# Esempio di file con Set UID
$ ls -l /usr/bin/passwd
-rwsr-xr-x. 1 root root 27856 Apr  1 2020 /usr/bin/passwd

# Impostare Set UID
$ chmod u+s file
$ chmod 4755 file

# Impostare Set GID
$ chmod g+s file
$ chmod 2755 file
```

7.2 Sticky Bit

Lo sticky bit su una directory impedisce agli utenti di eliminare i file di altri utenti in quella directory.

```
# Directory con sticky bit
$ ls -ld /tmp
drwxrwxrwt. 15 root root 4096 Dec 19 14:23 /tmp

# Impostare sticky bit
$ chmod +t directory
$ chmod 1755 directory
```

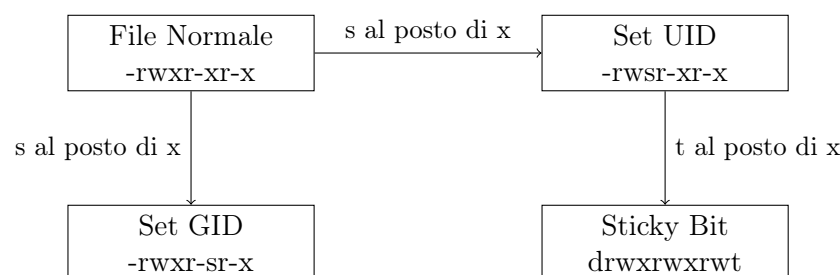


Figura 5: Permessi speciali in Linux

8 Comandi Avanzati di Elenco

8.1 Opzioni Utili di ls

8.2 Esempi Combinati

Opzione	Descrizione
-a	Mostra tutti i file, inclusi quelli nascosti (che iniziano con .)
-l	Mostra formato lungo con permessi, proprietà, dimensione
-h	Mostra dimensioni in formato leggibile (KB, MB, GB)
-t	Ordina per tempo di modifica
-r	Inverte l'ordine di ordinamento
-R	Elenca ricorsivamente sottodirectory
-S	Ordina per dimensione file
-d	Elenca directory stesse, non il loro contenuto
-F	Aggiunge indicatori di tipo (/ per directory, * per eseguibili)
-color	Colora l'output per tipo di file

Tabella 8: Opzioni comuni del comando ls

```
# Elenco lungo con file nascosti e dimensioni leggibili
$ ls -lah

# Ordinare per dimensione in ordine inverso
$ ls -lSr

# Elenco ricorsivo colorato
$ ls -R --color=auto

# Nascondere file che iniziano con 'g'
$ ls --hide=g*

# Mostrare solo info sulla directory
$ ls -ld $HOME/test/
```

9 Conclusioni

La gestione efficace del filesystem Linux richiede la comprensione di diversi concetti fondamentali:

1. **Struttura gerarchica:** Tutto parte dalla radice (/) e si organizza in un albero di directory
2. **Permessi:** Controllano chi può accedere e modificare file e directory
3. **Proprietà:** Ogni file appartiene a un utente e a un gruppo
4. **Metacaratteri:** Facilitano il lavoro con gruppi di file
5. **Comandi di base:** cd, ls, mkdir, chmod, chown, mv, cp, rm sono essenziali

La padronanza di questi strumenti e concetti è fondamentale per lavorare efficacemente in un ambiente Linux, sia come utente che come amministratore di sistema.

9.1 Best Practices

- Utilizzare sempre l'opzione -i (interattiva) con comandi potenzialmente distruttivi come rm, mv, cp
- Prestare attenzione ai permessi quando si condividono file
- Usare umask appropriati per impostare permessi predefiniti sicuri

- Verificare sempre i permessi prima di eseguire script o programmi
- Utilizzare i permessi più restrittivi possibili per motivi di sicurezza
- Documentare le modifiche ai permessi e alla proprietà dei file di sistema

10 Riferimenti Utili

Per approfondire gli argomenti trattati in questo documento:

- Man pages: `man ls`, `man chmod`, `man chown`
- Filesystem Hierarchy Standard (FHS): documentazione ufficiale sulla struttura delle directory
- Linux Documentation Project: guide complete su tutti gli aspetti di Linux
- Bash manual: per approfondire i metacaratteri e le funzionalità della shell