Capitolo 19

# Memoria di massa e file system

## 19.1 Introduzione alla memoria di massa

Con il termine ***memoria di massa*** ci si riferisce alla parte di memoria non volatile di un elaboratore, che consente l'immagazzinamento di grandi quantità di dati. Il tipo di supporto più utilizzato è quello a disco, il quale permette un accesso rapido ai dati, mentre in passato sono stati molto importanti anche i nastri magnetici, graditi a suo tempo per la loro economicità in rapporto alla quantità di dati memorizzabili. Tuttavia, il futuro delle memorie di massa sembra essere rappresentato dalle memorie solide, ovvero da sistemi di memorizzazione elettronica allo stato solido (senza componenti meccanici).

### 19.1.1 Nastro

Il sistema di memorizzazione a nastro è stato il primo (a parte le schede perforate) a essere utilizzato con gli elaboratori. In tempi più recenti, dal nastro in bobine si è passati piuttosto a cartucce a nastro magnetico di vario tipo.

La memorizzazione a nastro permette la registrazione di dati in modo sequenziale. Questo significa che la modifica dei dati può avvenire solo aggiungendo queste modifiche alla fine, senza poter intervenire nella parte già memorizzata. Nello stesso modo, l'accesso a un'informazione richiede lo scorrimento del nastro fino al punto in cui questa si trova.

Solitamente, la memorizzazione di dati all'interno di un nastro avviene in forma di archivio, cioè di un file unico contenente tutti i file che si vogliono archiviare. In questo modo, è il programma di archi-

viazione ed estrazione a prendersi cura del confezionamento dei dati da archiviare e del loro recupero quando necessario.

Figura 19.1. Unità a nastro IBM. Questa foto proviene dalla raccolta di Peter Roosen-Runge, visibile alla pagina *York computing: scenes from a distant past*, http://www.cse.yorku.ca/~peter/deptphotos/photos.html , ripubblicata qui con il suo permesso.

### 19.1.2 Disco

Il supporto di memorizzazione a disco, ha il vantaggio di consentire l'accesso diretto ai dati senza la necessità di scorrerli sequenzialmente. Le tecniche di memorizzazione possono essere differenti: magnetica, magneto-ottica e ottica. Nel primo caso, il più comune, i dati vengono memorizzati su uno strato ferromagnetico; nel se-

condo, si sfrutta sempre un sistema di memorizzazione magnetica, ma su un materiale che deve essere scaldato con un fascio laser per consentire la memorizzazione; l'ultimo utilizza una memorizzazione puramente ottica, attraverso un laser.

La memorizzazione magnetica è la più comune, offre il vantaggio di una maggiore velocità di lettura e scrittura dei dati, ma è anche la meno sicura per quanto riguarda la durata di mantenimento di questi.

Bisogna ricordare che esiste una grande quantità di fonti magnetiche, cominciando dal campo terrestre, a cui si aggiungono tutti quelli generati dall'attività umana. Tanto per fare un esempio, un telefono mobile acceso, appoggiato vicino a un dischetto magnetico, può provocare la sua cancellazione.

I dischi magneto-ottici sono funzionalmente analoghi a quelli magnetici, con la differenza che, dovendo scaldare le tracce da memorizzare con un laser, quando questo calore non è disponibile non sono tanto sensibili ai campi magnetici estranei.

I dischi ottici utilizzano una memorizzazione attraverso un fascio laser. Il tipo più comune di disco ottico può essere usato una volta sola per memorizzare i dati, consentendo naturalmente una quantità indefinita di letture successive, ma a questo si affianca il disco riscrivibile, il quale può essere riutilizzato.

#### 19.1.2.1 Dischi fissi e rimovibili

Esistono principalmente due tipi di dischi: quelli che sono fissati stabilmente all'apparecchiatura che permette di effettuare delle registrazioni e di leggerne il contenuto, distinguendoli da quelli che pos-

sono essere asportati e quindi sostituiti. Tuttavia va considerato che i dischi «fissi» possono essere inseriti in unità esterne che possono essere spostate da un elaboratore all'altro.

Figura 19.2. Disco fisso e disco rimovibile.

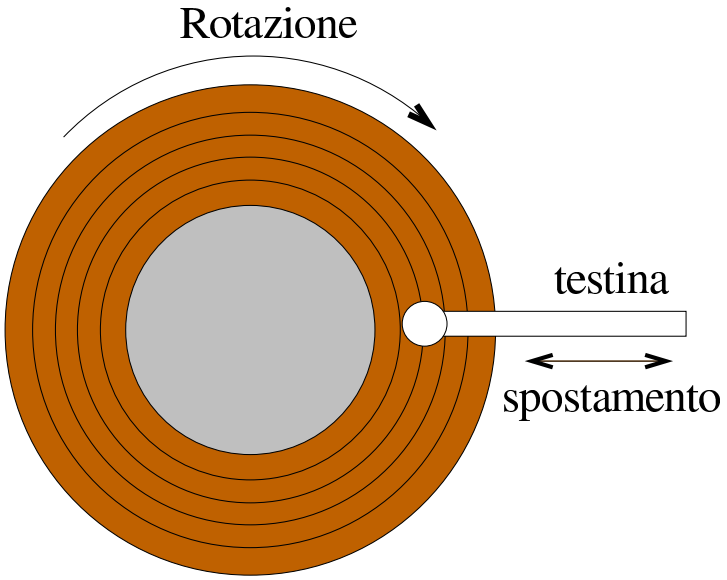
Il disco fisso ha normalmente una capacità molto superiore a un disco rimovibile, permettendo di effettuare le operazioni di registrazione e rilettura dei dati molto più velocemente. Il disco rimovibile ha il vantaggio di poter essere tolto e sostituito con altri, senza portare con sé tutta la meccanica e l'elettronica necessaria alla sua gestione.

Figura 19.3. Disco fisso esterno USB.

### 19.1.2.2 Dischi magnetici

Il disco magnetico è costituito essenzialmente da uno o più piatti di materiale metallico (alluminio o un'altra lega trasparente ai campi magnetici), plastico, o vetroso, ricoperti su entrambe le facce da un deposito di ossidi ferromagnetici (la stessa sostanza che ricopre il nastro magnetico delle cassette audio). Questi piatti vengono fatti ruotare a velocità angolare costante.

Figura 19.4. Visione dall'alto di un piatto sul quale scorre una testina per la lettura e scrittura dei dati. Le tracce magnetiche, concentriche, non sono visibili, ma vengono individuate dalla testina magnetica.




L'operazione di registrazione e rilettura dei dati viene effettuata da testine, una per ogni faccia dei piatti, le quali registrano e rileggono lungo tracce concentriche del disco. Le tracce magnetiche vengono definite dalle testine stesse, durante una fase detta di inizializzazione (o formattazione) a basso livello. Le tracce sono suddivise a loro volta in settori di uguale dimensione contenenti un codice di identificazione.

Figura 19.5. Le tracce concentriche dei dischi magnetici sono suddivise in settori di uguale dimensione.


I settori sono in pratica dei contenitori di dati. L'esistenza di questi settori e del loro sistema di identificazione permette l'accesso diretto ai dati, fino all'unità minima gestibile, costituita appunto dal settore. Nello stesso modo, non è possibile registrare dati se prima non sono state definite le tracce e i settori.

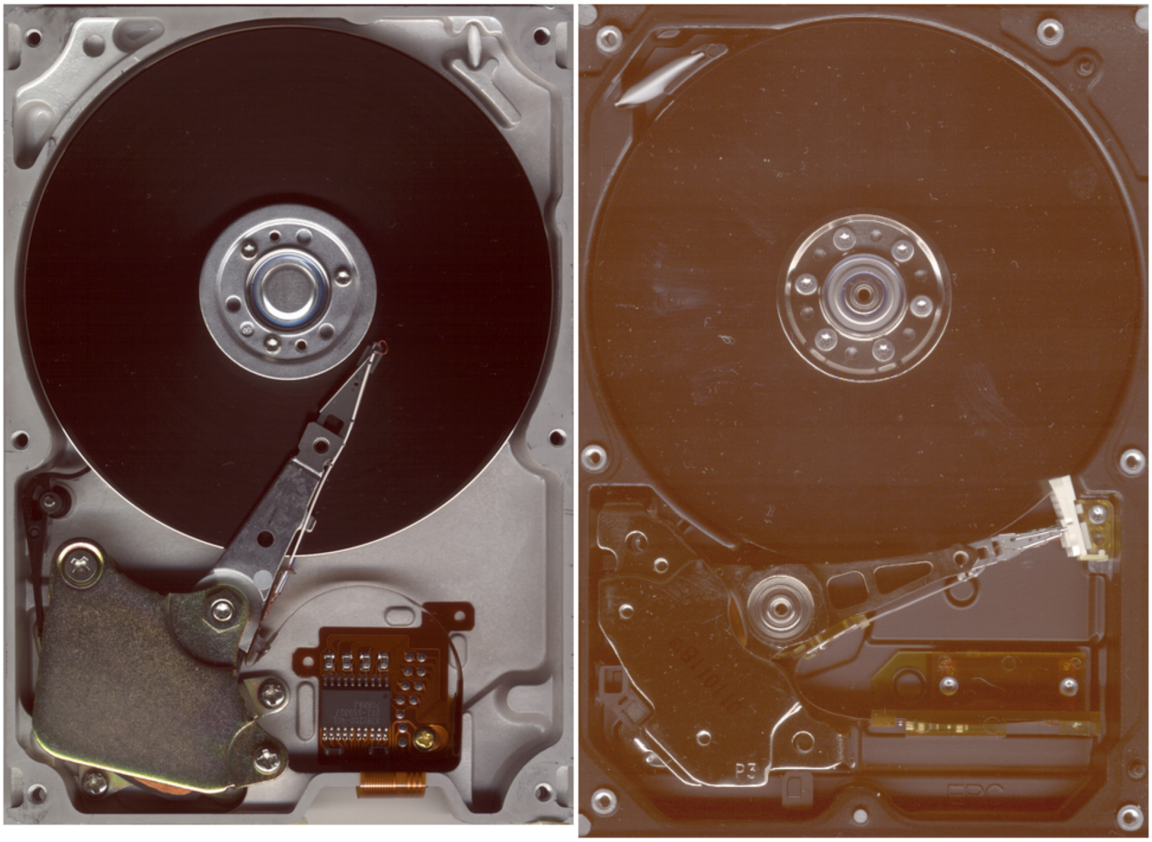
In passato sono esistiti dischi magnetici nei quali la suddivisione delle tracce in settori veniva identificata attraverso riferimenti estranei alle tracce stesse, per esempio attraverso dei fori in qualche punto del piatto. Questo vecchio tipo di dischi veniva detto *hard sectored* (suddiviso fisicamente in settori), mentre la modalità attuale, cioè quella che si ottiene con l'inserzione di codici di riconoscimento all'inizio dei settori, si dice *soft sectored* (suddiviso logicamente in settori). In ogni caso, è sempre presente un riferimento fisico per definire un punto di inizio nella rotazione.

Quando un'unità di memorizzazione è composta da più dischi, questi sono calettati sullo stesso albero (l'asse di rotazione centrale). In questo modo, la registrazione e la lettura avvengono attraverso un pettine di testine collegate assieme.

Le testine non possono appoggiare sul piatto, altrimenti si genererebbe un attrito e conseguentemente un riscaldamento disastroso. La presenza dell'aria o di un altro gas, fa sì che con la rotazione le testine si appoggino su un cuscino d'aria sottilissimo. A volte può succedere che un corpo estraneo si inserisca tra una testina e il piatto.

Quando ciò succede, nella maggior parte dei casi, si arriva all'atterraggio della testina e alla conseguente distruzione del piatto e della testina stessa.

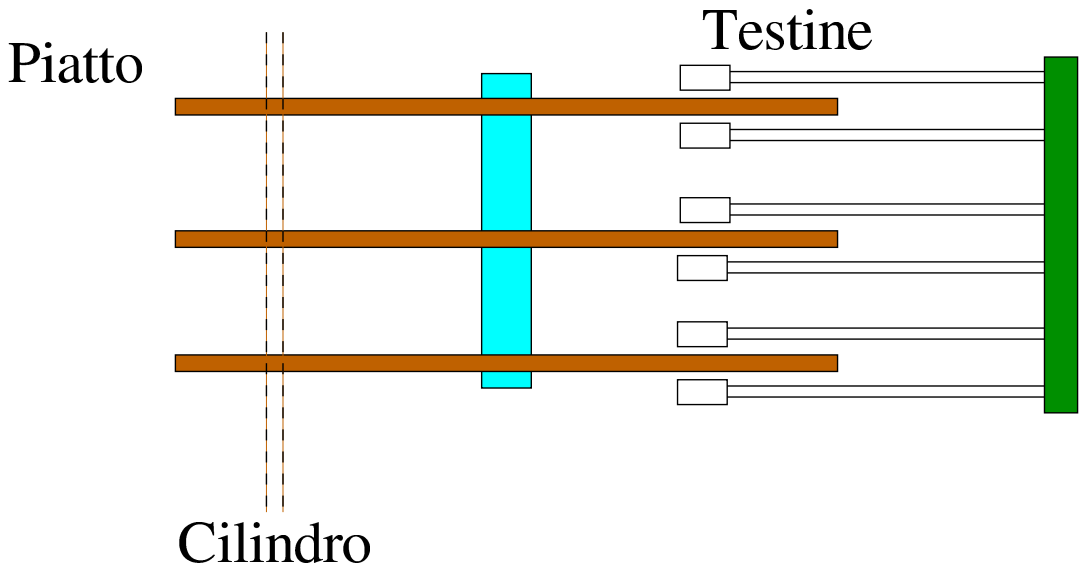
Figura 19.6. Foto di due dischi fissi aperti. Rispetto al disegno che appare nella figura 19.4, la rotazione di questi dischi avviene in senso antiorario. Inoltre, a sinistra si vede che la posizione di riposo delle testine è nella traccia più interna, mentre a destra queste vengono estratte completamente, per maggiore sicurezza.



### 19.1.2.3 Geometria

Come già accennato, il settore è l'unità di memorizzazione minima a cui si possa accedere in un disco, di qualunque tipo esso sia. Nel caso di dischi organizzati a tracce concentriche, sono state utilizzate storicamente coordinate composte da cilindro, testina e settore.

Figura 19.7. Schema di un disco fisso visto lateralmente.




Il cilindro rappresenta un gruppo di tracce, tutte alla stessa distanza dall'asse di rotazione centrale; la testina identifica la traccia specifica facendo riferimento alla faccia di un piatto particolare da prendere in considerazione; il settore è il segmento di traccia a cui si vuole accedere.

Per definire le caratteristiche di un disco del genere si parla di geometria, espressa attraverso l'indicazione del numero di cilindri, di testine e di settori a disposizione. La dimensione di un disco, espressa in settori, si esprime quindi come il prodotto di questi tre valori.

Infine è importante considerare anche la dimensione del settore, ovvero la quantità di byte che questo può contenere. La dimensione normale è di 512 byte, ma esistono anche dischi con settori di dimensione multipla.

Attualmente, dal momento che il sistema di controllo elettronico di una unità a disco è molto più sofisticato di un tempo, la geometria è un concetto che serve solo a rispettare la tradizione. In generale, sarebbe possibile l'accesso ai settori semplicemente indicandone

il numero di sequenza, lasciando al sistema di controllo interno al disco di sapere dove si trova effettivamente, in base alla geometria reale. Tuttavia, per ragioni di compatibilità, si continua a immaginare una **geometria virtuale**, in quanto non corrispondente a quella reale.

### 19.1.2.4 Dischi magneto-ottici

Di norma, i dischi magneto-ottici (MO) sono dischi rimovibili in cui i dati sono registrati in forma magnetica, ma l'operazione di registrazione avviene previo riscaldamento da parte di un fascio laser. Si tratta storicamente di unità di memorizzazione di grande capacità, ma ad accesso piuttosto lento e attualmente superate sotto tutti gli aspetti.

Figura 19.8. Disco magneto-ottico rimovibile.


Il disco magneto-ottico ha una geometria analoga a quella dei dischi magnetici, anche se si tratta solo di un piatto, per cui, anche in questo

caso si parla di cilindri, testine e settori.

### 19.1.2.5 Dischi ottici

«

Mentre i dischi magneto-ottici si comportano in maniera analoga a quelli magnetici, i dischi ottici richiedono una registrazione sequenziale (come i nastri), consentendo eventualmente un'aggiunta in coda. Tuttavia, la rilettura non comporta limitazioni di accesso. Per questo, i dischi ottici sono registrati utilizzando un'unica traccia a spirale, un po' come si faceva con i vecchi dischi musicali di vinile.

Figura 19.9. Disco ottico registrato: si può notare il bordo interno da dove parte l'incisione e il limite esterno dove questa ha termine.

### 19.1.2.6 Memorie solide

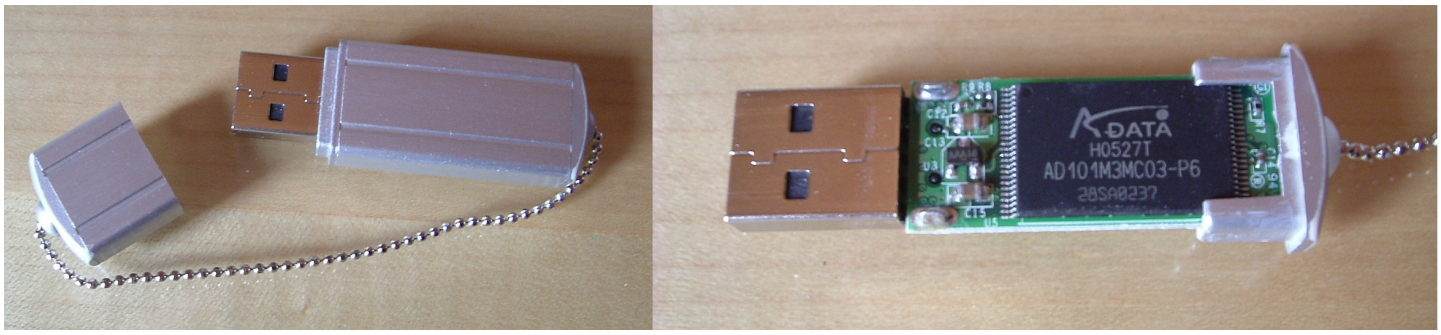
Le memorie solide sono degli integrati che consentono la memorizzazione di discrete quantità di dati, in modo simile a quello delle memorie di massa tradizionali. Attualmente, il limite tecnico di tali sistemi di memorizzazione sta nel numero massimo di cicli di scrittura.

L'uso più comune delle memorie solide avviene attraverso le porte USB, gestendole come se fossero dei dischi esterni.

Figura 19.10. Esempio di memoria solida, per l'uso in apparecchi come le macchine fotografiche.


Figura 19.11. Lettore di vari modelli di memorie solide esterne.

Figura 19.12. Memoria solida esterna in forma di «penna» USB.




### 19.1.3 Collocazione e accesso ai dati

«

Quando si utilizza un nastro magnetico, la memorizzazione dei dati può avvenire solo in modo sequenziale, per cui, di solito si registra un file unico contenente tutto ciò che si vuole archiviare.

Nel caso delle unità ad accesso diretto (a disco e a memoria solida), i dati possono essere suddivisi in settori e sparpagliati nell'unità di memorizzazione come si vuole, possibilmente con un qualche criterio. Questo criterio è il file system, cioè qualcosa che definisce un'organizzazione dei dati nell'unità e il modo per potervi accedere.

Un'unità di memorizzazione priva di file system è solo una sequenza di settori. In questo senso, a volte si utilizzano le unità di memorizzazione ad accesso diretto come se fossero nastri, registrando e rileggendo i dati nella stessa sequenza naturale dei settori.

Il caso del disco ottico è speciale, nel senso che la registrazione avviene come se si trattasse di un nastro, ma quanto registrato può contenere un file system (solitamente si tratta di quello definito dallo standard ISO 9660, oppure ISO 13346) e la rilettura dei dati può avvenire ad accesso diretto, come nel caso dei dischi normali.

I dischi ottici (CD, DVD, BR) da utilizzare con i sistemi Unix contengono normalmente un file system ISO 9660 con estensioni Rock Ridge (oppure un file system UDF, ISO 13346). Quando il disco viene letto da un sistema che non è in grado di riconoscere le estensioni Unix, riesce ugualmente ad accedervi, però tutto si manifesta semplicemente come nello standard ISO normale.

### 19.1.4 Partizioni secondo la modalità MS-Windows

Le unità di memorizzazione di massa ad accesso diretto, come i dischi e le memorie solide, purché di dimensioni adeguate, possono essere suddivisi in partizioni. Questa suddivisione permette, per esempio, di fare convivere diversi sistemi operativi nella stessa unità.

Teoricamente, un'unità di memorizzazione di qualunque genere può essere suddivisa in partizioni, oppure anche no. In particolare, in presenza di dischi rimovibili di dimensioni superiori a qualche mebibyte (simbolo Mibyte), non suddivisi in partizioni, si parla a volte di *superfloppy*.

Il sistema della suddivisione in partizioni dipende dallo «schema di partizionamento». Quello più comune è rappresentata dal tipo utilizzato originariamente negli elaboratori IBM-PC, dagli anni 1980 in poi. Questo schema di partizionamento è utilizzato generalmente nei sistemi MS-Windows ed è quello predefinito dei sistemi GNU/Linux.

#### 19.1.4.1 MBR

Nel sistema di partizionamento secondo il modello utilizzato da MS-Windows, le informazioni principali sulla suddivisione in partizioni sono registrate nella parte finale del primo settore dell'unità di memorizzazione, togliendo un po' di spazio alle istruzioni di avvio. Per questo motivo, trattandosi di un settore di avvio con in più le informazioni sulle partizioni principali (primarie), questo si chiama MBR, o *Master boot record*.

Solitamente, il settore di avvio contiene il codice necessario a passare al primo settore di una partizione il compito di avviare effettivamente il sistema operativo: la partizione avviabile.

Lo spazio riservato nell'MBR per annotare i dati delle partizioni è limitato e consente la suddivisione in un massimo di quattro partizioni principali.

#### 19.1.4.2 Partizioni primarie ed estese

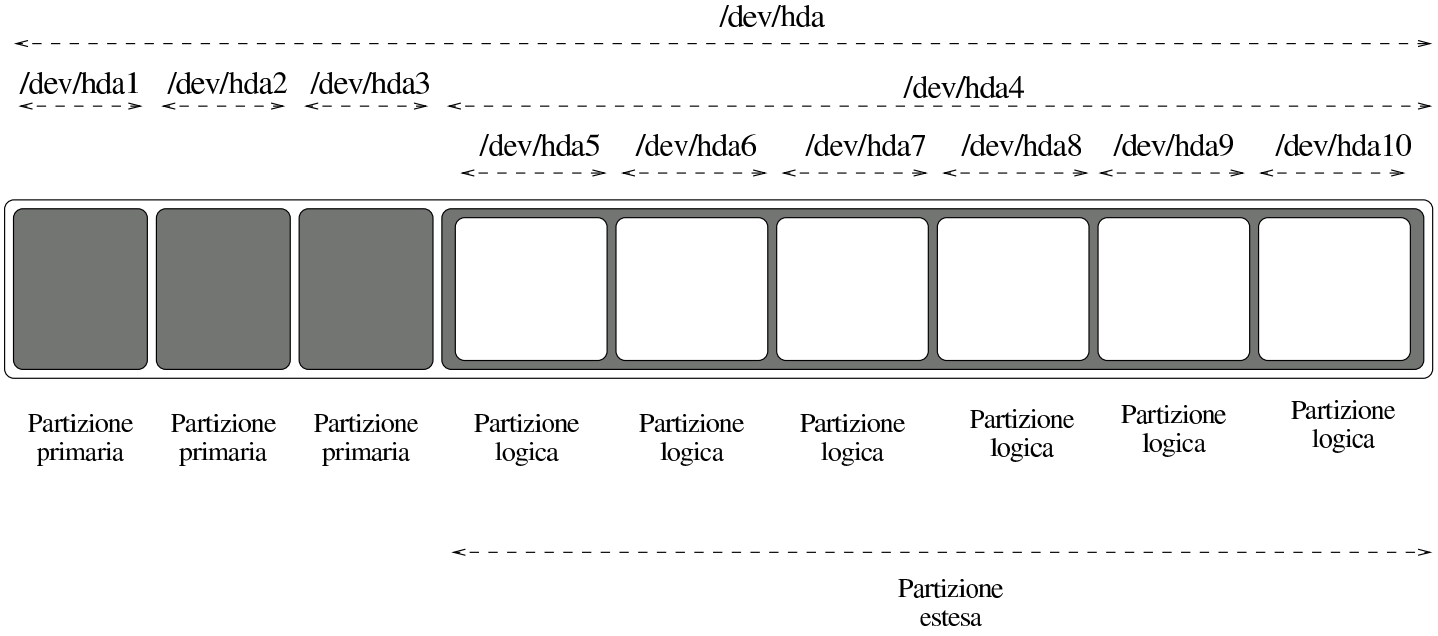
La possibilità di suddividere lo spazio di un'unità di memorizzazione in sole quattro partizioni può essere troppo limitante. Per risolvere questo problema si distinguono partizioni di due tipi: primarie ed estese. La partizione primaria è quella normale, a essa si attribuisce un codice per riconoscere il tipo di file system che contiene o che dovrebbe contenere. La partizione estesa viene definita con il codice $05_{16}$ ed è il contenitore di altre partizioni più piccole, dette partizioni logiche.

Si osservi che può essere definita una sola partizione estesa e che questa può essere suddivisa anche in più di quattro partizioni «logiche», ovvero sotto-partizioni.

GNU/Linux utilizza nomi particolari per i file di dispositivo che servono a identificare l'intera unità o una singola partizione. In pratica, quando si fa riferimento a una partizione, si aggiunge un numero al nome del file di dispositivo riferito all'unità intera. In particolare, i numeri da uno a quattro rappresentano le prime quattro partizioni (primarie o estese), mentre i numeri successivi vengono utilizzati per identificare le partizioni logiche eventuali.

La numerazione delle partizioni segue solo l'ordine di inserimento e non la collocazione effettiva nello spazio dell'unità di memorizzazione. Per esempio, se si hanno tre partizioni primarie e si rimuove la seconda per scomporla in due parti, si ottiene una seconda partizione più piccola e una quarta partizione, collocata tra la seconda e la terza.

Figura 19.13. Suddivisione in partizioni secondo il sistema tradizionale del Dos.

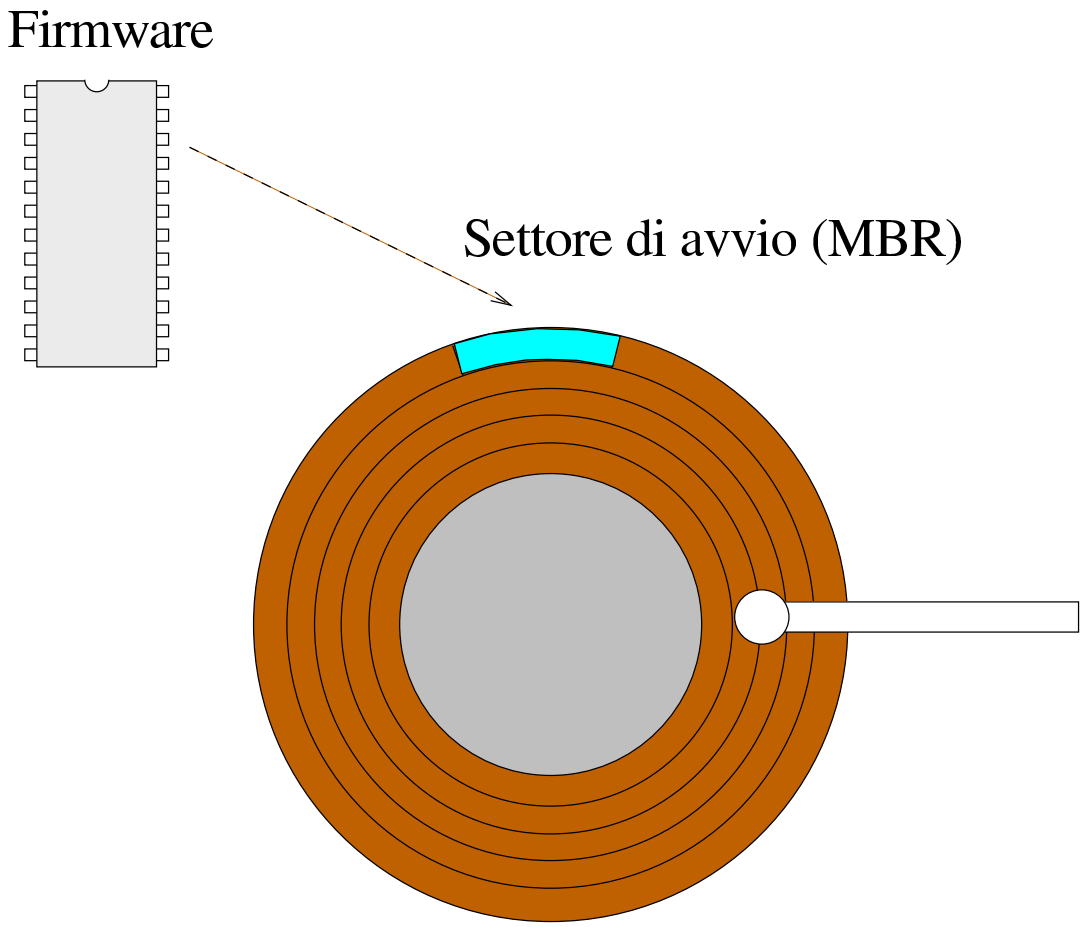



## 19.1.5 Firmware

«

Le unità di memorizzazione di massa, non sono solo contenitori di dati: nei sistemi operativi comuni sono coinvolte nel processo di caricamento del sistema stesso. Perché ciò possa avvenire, deve essere avviato un programma iniziale che provvede a sua volta ad avviare il sistema operativo. Questo programma è suddiviso in due parti: il firmware (o BIOS negli elaboratori x86) e il settore di avvio. In pratica, il firmware avvia il settore di avvio, il quale a sua volta avvia un altro settore di avvio oppure, un programma di avvio o direttamente il kernel del sistema operativo.

Figura 19.14. Dal firmware al settore di avvio.




Il codice contenuto in un settore di avvio può avvalersi solo di funzionalità offerte dal firmware stesso e quindi dipende da queste la possibilità di raggiungere e avviare il kernel. Nel firmware degli elaboratori x86 esiste una limitazione: le sue funzioni non permettono di accedere a zone di un'unità di memorizzazione oltre il 1 024-esimo cilindro. Ciò significa che un kernel collocato oltre questo punto può essere eseguito solo con l'intermediazione di un sistema di avvio abbastanza complesso, tale da poter fare a men o delle funzioni del BIOS. Tuttavia, per attenuare questo problema, il BIOS trasforma in maniera fittizia la geometria dei dischi in modo da mostrare meno cilindri del reale, aumentando gli altri valori (testine o settori per traccia), oppure se la inventa del tutto. In ogni caso, se il

sistema operativo è limitato o se non ci si può avvalere di un sistema di avvio abbastanza evoluto, occorre trovare il modo di fare risiedere il kernel in una posizione che sia sicuramente accessibile. Eventualmente, la tecnica più semplice è quella di predisporre una piccola partizione solo per questo nella zona protetta, al di sotto del cilindro 1023.

### 19.1.6 Modalità di accesso ai settori

«

I settori di un'unità di memorizzazione possono essere raggiunti in due modi: in base alle coordinate virtuali definite da cilindro, testina e settore, oppure in base al numero del settore dell'unità, in senso assoluto. Queste modalità hanno due sigle standard, che sono rispettivamente: CHS (*Cylinder, head, sector*) e LBA (*Linear block address*). Naturalmente, la modalità di accesso più «comoda» è LBA, ammesso che possa essere utilizzata.

La distinzione tra un accesso di tipo CHS o LBA riguarda principalmente i dischi ATA, evolutisi in un arco di tempo piuttosto lungo. La possibilità di accedere in modalità LBA permette di eliminare il problema delle dimensioni massime che, per motivi storici, rimangono nei valori delle coordinate di tipo CHS.

### 19.1.7 Memoria cache

«

L'accesso ai dati delle unità di memorizzazione di massa è un'operazione relativamente lenta e spesso si ripetono accessi successivi a zone contigue, oltre che alle stesse zone con variazioni successive dei dati.

Per ridurre gli accessi ripetuti alle unità di memorizzazione, i sistemi operativi utilizzano generalmente una ***memoria cache***,[1] riservan-

do parte della memoria RAM, con la quale le operazioni di lettura e scrittura vengono filtrate in modo da evitare richieste ridondanti nel breve periodo. In questo modo, un settore appena letto, se viene richiesto nuovamente dallo stesso programma o anche da un altro, risulta subito disponibile senza disturbare l'unità di memorizzazione. Nello stesso modo funziona l'operazione di scrittura che viene rinviata a un momento successivo in modo da avere accumulato un blocco di dati più grande.

La memoria cache viene scaricata periodicamente, a intervalli regolari. Tuttavia, a causa di questo meccanismo, uno spegnimento accidentale dell'elaboratore può comportare una perdita parziale dei dati, se le operazioni di scrittura accodate nella memoria cache non sono state trasferite in tempo nell'unità di memorizzazione.

Oltre all'azione dei sistemi operativi, si aggiunge spesso una memoria cache nell'hardware della stessa unità di memorizzazione. Questa non può essere controllata tanto facilmente se non attendendo qualche secondo prima di spegnere l'elaboratore dopo aver completato la procedura di arresto del sistema, in base al tipo di sistema operativo utilizzato.

### 19.1.8 File system Unix

I file system dei vari sistemi Unix condividono lo stesso tipo di impostazione e di conseguenza si utilizza una terminologia comune per descriverne le varie parti.

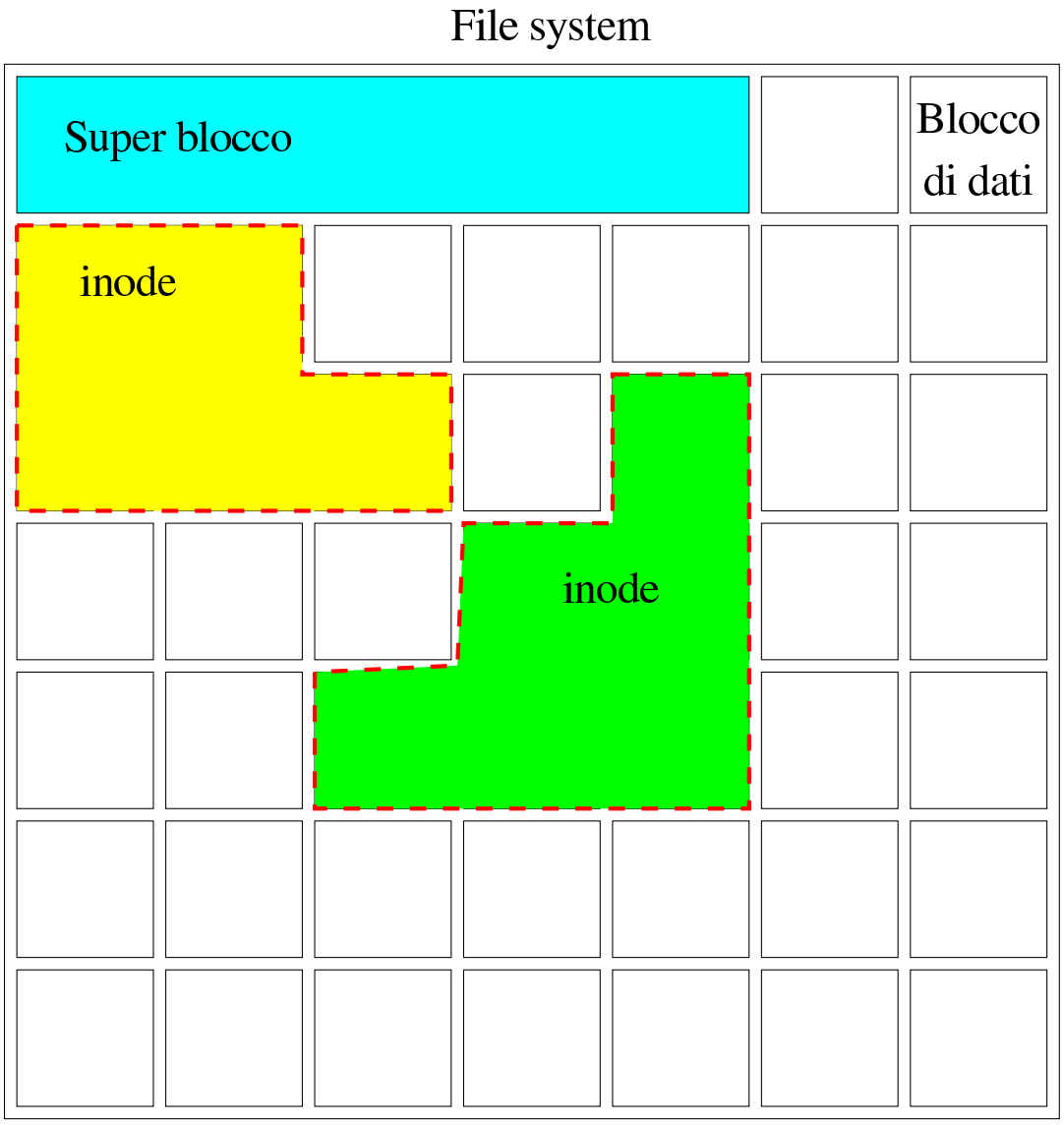
Semplificando molto le cose, si può immaginare che il file system Unix sia composto da due strutture che si sovrappongono: ***inode*** e directory.

- A capo di tutto c'è il ***super blocco*** che contiene informazioni generali sul file system.
- Un inode è un elemento contenente tutte le informazioni riferite a un file di qualunque tipo (comprese le directory), escluso il nome. In particolare, l'inode contiene i riferimenti necessari a raggiungere i blocchi di dati del file. Gli inode sono raggiungibili tramite il loro numero (numero di inode).
- Un blocco di dati è una zona nell'unità di memorizzazione utilizzata per contenere dati, corrispondente a un multiplo della dimensione del settore fisico dell'unità stessa. Il contenuto di un file può essere distribuito su più blocchi di dati.
- Una directory è un file contenente un elenco di nomi di file abbinati al numero di inode rispettivo.

La struttura di inode e blocchi di dati è sufficiente a definire le caratteristiche e il contenuto dei file. La struttura di directory permette di raggiungere i file per nome, organizzandoli nel modo consueto, attraverso diramazioni più o meno accentuate.

Nel super blocco, tra le altre cose, viene modificato un indicatore particolare (un *flag*) quando il suo file system viene innestato. Nel momento in cui si esegue il distacco, l'ultima cosa a essere modificata nel file system è l'indicatore di apertura che viene riportato al livello normale. In questo modo, ogni volta che si innesta un file system Unix è possibile verificare se ne è stato eseguito correttamente il distacco. Se risulta che l'attività nel file system non è stata conclusa correttamente si può temere che i dati siano danneggiati (ovvero che siano incoerenti).

Figura 19.15. File system Unix.




### 19.1.8.1 Directory e inode

Chi non ha mai avuto a che fare con un sistema Unix può trovare difficoltà a comprendere cosa siano gli inode, mentre questo è necessario per poter intendere correttamente cosa siano i collegamenti.

Un file system Unix ha due livelli di astrazione logica: inode e directory. Nella parte più bassa si trova l'unità di memorizzazione di massa, scomposta in blocchi di dati. Un file di qualunque tipo è

composto da blocchi (eventualmente anche nessun blocco, se il file è vuoto) e queste informazioni sono raccolte in un inode. L'inode viene identificato in base a un numero riferito alla tabella di inode.

Una directory è un file (cioè un inode come gli altri) che ha il compito speciale di raccogliere un elenco di riferimenti ad altri inode, a cui abbinare il nome relativo. Le voci contenute in una directory sono dei collegamenti (indicati più precisamente come collegamenti fisici o *hard link*) a degli inode.

Il termine «directory» deriva da questo: si tratta di un elenco di nomi a cui si associa il numero di inode corrispondente.

A questo punto, potrebbe essere interessante distinguere le informazioni contenute negli inode, da quelle che invece appartengono alle voci delle directory.

L'inode contiene le informazioni necessarie a raggiungere i blocchi di dati che compongono il file, oltre alle informazioni seguenti:

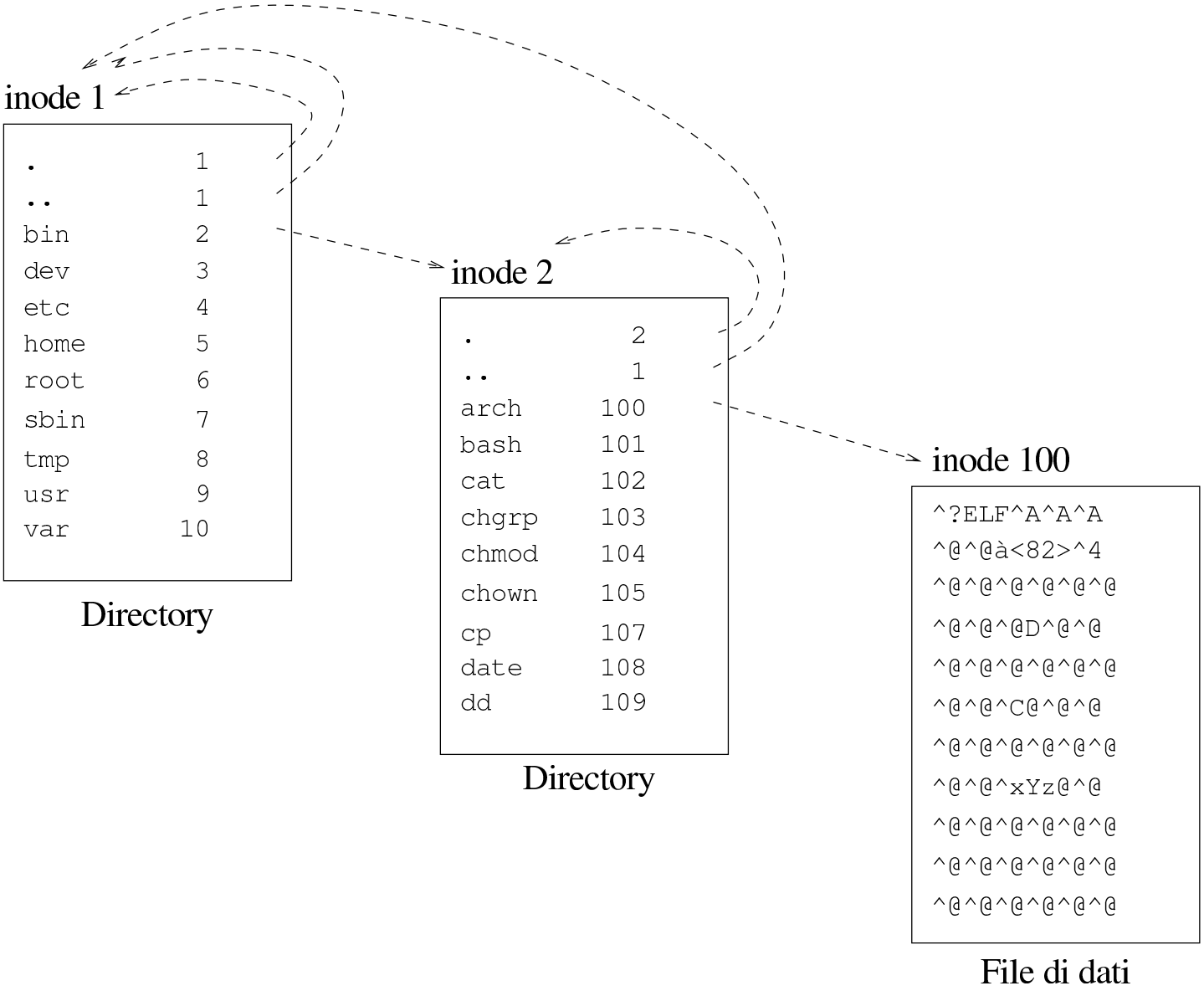
- la data dell'ultimo accesso;
- la data dell'ultima modifica;
- la data di creazione (dell'inode);
- il tipo di file;
- la dimensione complessiva in byte;
- i numeri UID e GID che rappresentano l'utente e il gruppo proprietari;
- i permessi, compresa l'indicazione dell'utente e del gruppo proprietari, attraverso l'indicazione dei numeri UID e GID;

- un contatore delle voci delle directory che vi fanno riferimento (ovvero, un contatore dei collegamenti fisici).

La directory contiene un elenco di voci, dove ognuna di queste contiene a sua volta:

- il nome;
- il riferimento all'inode (cioè il collegamento fisico).

Figura 19.16. Inode, directory e file.



### 19.1.8.2 Collegamenti o «link»

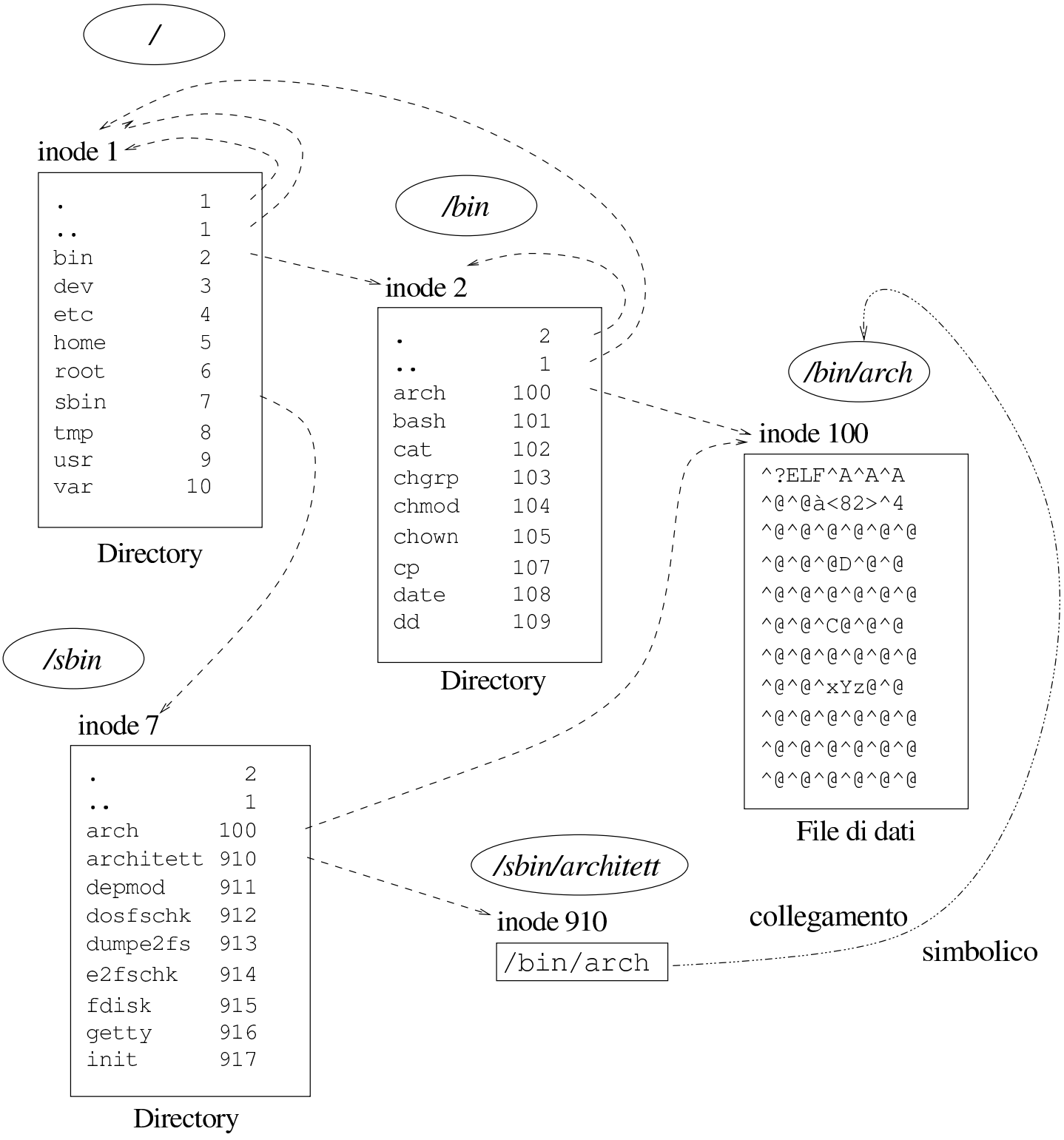
«

Come descritto nella sezione precedente, le voci di una directory contengono ognuna un riferimento (detto comunemente collegamento) a un inode. Più voci della stessa directory, o di directory differenti, possono puntare allo stesso inode. Quando si cancella un file, si cancella la voce della directory e il numero di riferimenti contenuti nell'inode viene ridotto. Quando questo raggiunge lo zero, quel numero di inode torna a essere disponibile.

Questa possibilità di avere riferimenti multipli allo stesso inode è ampliata dalla presenza dei cosiddetti collegamenti simbolici, o *symlink*, che sono solo file contenenti un riferimento a un altro file.

Per distinguere questi due tipi di collegamenti, si può parlare di collegamenti fisici, o *hard link*, per fare riferimento ai collegamenti che puntano direttamente agli inode.

Figura 19.17. Collegamenti. In questa figura si può notare che: i file '`/bin/arch`' e '`/sbin/arch`' sono la stessa cosa, in quanto corrispondono allo stesso inode; il file '`/sbin/architett`' è un collegamento simbolico a '`/bin/arch`'.



## 19.2 Preparazione e controllo

«

I sistemi Unix gestiscono sempre un solo file system globale. Questo può essere anche composto da più file system di dimensioni inferiori, uno principale (radice) e gli altri secondari, collegati fra loro in modo da formare un'unica struttura.

### 19.2.1 Preparazione dei file system

«

Prima di poter utilizzare un file system, occorre costruirlo. Quando si parla di unità di memorizzazione si distinguono tre fasi fondamentali:

1. l'inizializzazione a basso livello;
2. l'eventuale suddivisione in partizioni;
3. la creazione della struttura iniziale del tipo di file system che si intende utilizzare.

L'inizializzazione a basso livello riguarda le unità a disco, specialmente se rimovibili, ma in generale è sempre più raro che ci si debba preoccupare di tale procedura.[2]

Per la suddivisione in partizioni delle unità di memorizzazione si può utilizzare '**fdisk**' (o '**cfdisk**'), per creare i vari file system si devono utilizzare programmi diversi a seconda del tipo di file system.

Tutte queste operazioni vengono svolte facendo riferimento ai file di dispositivo relativi. Di conseguenza, possono essere compiute solo dagli utenti che hanno i permessi di accesso in lettura e scrittura per questi file. Generalmente, solo l'utente '**root**' può intervenire in questo modo.

#### 19.2.1.1 Utilizzo di «badblocks»

Il programma '**badblocks**'[3] è in grado di verificare l'integrità di un'unità o di una partizione. Il controllo è fatto a basso livello senza considerare la struttura del file system.

```
badblocks [opzioni] dispositivo dimensione_in_blocchi [blocco_iniziale]
```

Normalmente i programmi di inizializzazione, sia a basso livello che a livello superiore, sono in grado di fare questo controllo da soli. Per questo '**badblocks**' viene usato raramente.

Il tipo di controllo può essere in lettura oppure anche in scrittura. È evidente che, se si specifica attraverso le opzioni l'intenzione di effettuare un controllo in scrittura, i dati contenuti nell'unità o nella partizione sono perduti.

Tabella 19.18. Alcune opzioni.

| Opzione | Descrizione |
|---|---|
| `-b` ***dimensione_dei_blocchi*** | Permette di definire la dimensione dei blocchi espressa in byte. Il valore predefinito è 1024. |
| `-w` | Esegue una prova di scrittura controllando successivamente l'esito. Questa opzione deve essere usata con prudenza dal momento che, così facendo, si cancellano i dati dell'unità o della partizione da controllare. |

L'esempio seguente esegue il controllo del dischetto, in sola lettura, per tutta la sua estensione: 1440 blocchi di 1 Kibyte. Trattandosi di un controllo in sola lettura, '**badblocks**' può essere eseguito da un utente comune (sempre che tali utenti abbiano i permessi di lettura per il dispositivo che si va a leggere).

```
$ badblocks /dev/fd0u1440 1440 [Invio]
```

### 19.2.1.2 Utilizzo di «fdisk»

«

Il programma '**fdisk**'[4] [5] permette la modifica della tabella delle partizioni di un'unità di memorizzazione di massa che possa essere organizzata in questo modo. Il nome del file di dispositivo fa riferimento all'intera unità, quindi si possono utilizzare nomi come '`/dev/sda`', '`/dev/sdb`', '`/dev/sdc`',...

```
fdisk [opzioni] [dispositivo]
```

Una volta avviato '**fdisk**', si interagisce con questo attraverso comandi composti da una sola lettera. In particolare, la lettera '**m**'

richiama l'elenco dei comandi disponibili.

[*m*]

```
Command action
   a   toggle a bootable flag
   b   edit bsd disklabel
   c   toggle the dos compatiblity flag
   d   delete a partition
   l   list known partition types
   m   print this menu
   n   add a new partition
   p   print the partition table
   q   quit without saving changes
   t   change a partition's system id
   u   change display/entry units
   v   verify the partition table
   w   write table to disk and exit
   x   extra functionality (experts only)
```

Quando viene creata una nuova partizione, questa viene definita automaticamente del tipo Linux-nativa, ma in certi casi può essere necessario modificare il tipo di partizione creato attraverso il comando '**t**'. Ogni tipo di partizione ha un codice (espresso in esadecimale) che può essere conosciuto anche attraverso '**fdisk**' stesso, durante il suo funzionamento.

[*t*]

...

```
  0  Vuoto           1c  Hidden W95 FAT3 70  DiskSecure Mult bb  Boot Wizard hid
  1  FAT12           1e  Hidden W95 FAT1 75  PC/IX           be  Solaris boot
  2  XENIX root      24  NEC DOS         80  Vecchio Minix   c1  DRDOS/sec (FAT-
  3  XENIX usr       39  Plan 9          81  Minix / vecchio c4  DRDOS/sec (FAT-
  4  FAT16 <32M      3c  Recupero Partit 82  Linux swap      c6  DRDOS/sec (FAT-
  5  Esteso          40  Venix 80286     83  Linux           c7  Syrinx
```

```
 6  FAT16           41  PPC PReP Boot   84  C nascosto OS/2 da  Non-FS data
 7  HPFS/NTFS       42  SFS             85  Linux esteso    db  CP/M / CTOS / .
 8  AIX             4d  QNX4.x          86  set volume NTFS de  Dell Utility
 9  AIX avviabile   4e  QNX4.x 2a parti 87  set volume NTFS df  BootIt
 a  OS/2 Boot Manag 4f  QNX4.x 3rd part 8e  Linux LVM       e1  accesso DOS
 b  W95 FAT32       50  OnTrack DM      93  Amoeba          e3  DOS R/O
 c  W95 FAT32 (LBA) 51  OnTrack DM6 Aux 94  Amoeba BBT      e4   SpeedStor
 e  W95 FAT16 (LBA) 52  CP/M            9f  BSD/OS          eb  BeOS fs
 f  W95 Ext'd (LBA) 53  OnTrack DM6 Aux a0  Ibernazione IBM ee  EFI GPT
10  OPUS            54  OnTrackDM6      a5  FreeBSD         ef  EFI (FAT-12/16/
11  FAT12 nascosto  55   EZ-Drive       a6  OpenBSD         f0  Linux/PA-RISC b
12  Diagnostica Com 56  Golden Bow      a7  NeXTSTEP        f1   SpeedStor
14  FAT16 nascosto  5c  Priam Edisk     a8  Darwin UFS      f4   SpeedStor
16  FAT16 nascosto  61   SpeedStor      a9  NetBSD          f2  DOS secondario
17  HPFS/NTFS nasco 63  GNU HURD o SysV ab  Darwin boot     fd  Autorilevamento
18  AST SmartSleep  64  Novell Netware  b7  BSDI fs         fe  LANstep
1b  Hidden W95 FAT3 65  Novell Netware  b8  BSDI swap       ff  BBT
```

Le modifiche alla tabella delle partizioni vengono registrate solo nel momento in cui si termina l'esecuzione del programma con il comando '**w**'. Se '**fdisk**' segnala qualche tipo di errore in questo momento, potrebbe essere necessario riavviare il sistema prima di utilizzare l'unità su cui sono state apportate le modifiche.

[*w*]

Il funzionamento di '**fdisk**' è già descritto nel capitolo 6.

Tabella 19.21. Alcune opzioni.

| Opzione | Descrizione |
|---|---|
| `-l` | Emette l'elenco delle partizioni esistenti nelle unità di memorizzazione presenti. Non inizia alcuna attività interattiva. La lettera «l» sta per *list*. |

| Opzione | Descrizione |
|---|---|
| `-s` ***partizione*** | Utilizzando questa opzione seguita dal nome del file di dispositivo che fa riferimento a una partizione (‘`/dev/sda1`’, ‘`/dev/sda2`’, ecc.) si ottiene la sua dimensione. Questa informazione è importante nel momento in cui si vuole creare al suo interno un file system e il programma utilizzato non è in grado di determinarla da solo.<br>La lettera «s» sta per *size*. |

### 19.2.1.3 Utilizzo di «cfdisk»

Il programma ‘**`cfdisk`**’[6] serve a modificare la suddivisione in partizioni di un’unità di memorizzazione di massa che possa essere organizzata in questo modo. Si tratta di un programma che svolge le stesse funzioni di ‘**`fdisk`**’ offrendo un sistema di interazione meno spartano.

```
cfdisk [opzioni] [dispositivo]
```

Dal momento che richiede delle librerie particolari per la gestione dello schermo (‘**`ncurses`**’), è poco indicato il suo utilizzo in presenza di sistemi estremamente ridotti o di emergenza. Ciò significa che il programma ‘**`fdisk`**’ tradizionale non può essere abbandonato per adottare esclusivamente ‘**`cfdisk`**’.

Il funzionamento di ‘**`fdisk`**’ è già descritto nel capitolo 6.

### 19.2.1.4 Utilizzo di «sfdisk»

Il programma '**sfdisk**'[7] consente la modifica delle partizioni in modo non interattivo, pertanto è utile in particolare per la realizzazione di script.

```
sfdisk [opzioni] dispositivo
```

```
sfdisk -s [partizione]
```

```
sfdisk dispositivo < file_di_comandi
```

L'utilizzo normale di questo programma di servizio prevede la preparazione di un file contenente le istruzioni sulle partizioni da creare all'interno di un'unità di memorizzazione di massa specificata espressamente. Anche se è prevista una sintassi apposita per queste istruzioni, può essere conveniente l'utilizzo di quanto ottenuto da un'interrogazione con lo stesso '**sfdisk**', come viene mostrato in seguito. Prima di arrivare a vedere in che modo si possono definire le partizioni, conviene prendere confidenza con l'uso di '**sfdisk**', attraverso delle operazioni non distruttive; pertanto si comincia subito con alcuni esempi.

- # **sfdisk -s /dev/sda1** [*Invio*]

  Questo comando si limita a restituire un numero attraverso lo standard output, corrispondente alla quantità di blocchi della prima partizione della prima unità.

- # **sfdisk -s /dev/sda** [*Invio*]

In questo caso si ottiene la quantità di blocchi complessiva della prima unità.

- # **sfdisk -V /dev/sda** [*Invio*]

  Verifica la coerenza delle partizioni nella prima unità. Di solito, '**sfdisk**' viene usato in questo modo per ottenere il valore restituito: se è pari a *Vero* (zero), tutto è in ordine.

- # **sfdisk -d /dev/sda** [*Invio*]

  Genera un rapporto sulle partizioni della prima unità, emesso attraverso lo standard output. Questo potrebbe essere ridiretto in un file, da conservare da qualche parte; in seguito, questo stesso file potrebbe essere usato per rigenerare la stessa situazione:

  # **sfdisk -d /dev/sda > /mnt/usb/partizioni** [*Invio*]

  ...

  # **sfdisk /dev/sda < /mnt/usb/partizioni** [*Invio*]

Un esempio del rapporto che si ottiene con l'opzione '**-d**' potrebbe essere quello del listato seguente:

```
# partition table of /dev/sda
unit: sectors

/dev/sda1 : start=       63, size=  612801, Id= 6
/dev/sda2 : start=   612864, size= 2721600, Id= 5, bootable
/dev/sda3 : start=        0, size=       0, Id= 0
/dev/sda4 : start=        0, size=       0, Id= 0
/dev/sda5 : start=   612927, size=   20097, Id=83
/dev/sda6 : start=   633087, size=  205569, Id=82
/dev/sda7 : start=   838719, size= 2495745, Id=83
```

Con questo sistema, se si dispone di più elaboratori con le stesse unità di memorizzazione di massa, da suddividere nello stesso modo, è facile utilizzare '**sfdisk**' per copiare la struttura di una nelle altre. Se si sa quello che si fa, si può modificare uno di questi file prima di darlo in pasto a '**sfdisk**'.

Il programma '**sfdisk**' permette anche di utilizzare una sintassi differente e più approssimativa per definire le partizioni che si vogliono creare. Tuttavia, per questo conviene leggere la documentazione originale che dovrebbe essere accessibile attraverso la pagina di manuale *sfdisk(8)*.

Tabella 19.23. Alcune opzioni.

| Opzione | Descrizione |
|---|---|
| `-s`<br>`--show-size` | Mostra la dimensione di una partizione. |
| `-l`<br>`--list` | Mostra l'elenco delle partizioni di un'unità di memorizzazione. |
| `-d`<br>`--dump` | Scarica le informazioni sulle partizioni di un'unità di memorizzazione. Quello che si ottiene può essere riutilizzato per rigenerare la stessa struttura, utilizzandolo come file di comandi per '**sfdisk**'. |
| `-V`<br>`--verify` | Verifica se le partizioni sembrano organizzate correttamente. |

### 19.2.1.5 Salvataggio e recupero delle informazioni sulle partizioni principali

Un'unità di memorizzazione suddivisa in partizioni secondo il modello originario dei sistemi Dos, il quale prevede quattro partizioni primarie, contiene tali informazioni all'interno del primo settore, noto con il nome «MBR». È molto semplice salvare tale suddivisione, anche senza servirsi di programmi specifici (come potrebbe essere '**sfdisk**'), purché il proprio interesse sia limitato alle sole partizioni principali. Per salvare il settore MBR si procede così:

```
dd if=file_di_dispositivo of=file_da_creare bs=512 count=1
```

In questo modo, il file di dispositivo deve essere quello dell'unità intera, dalla quale si vuole leggere il primo settore di 512 byte; il file da creare è quel file nel quale si vuole salvare l'informazione.

Nel caso dovesse essere necessario recuperare la tabella delle partizioni, se si copia tale e quale il file salvato in precedenza nel primo settore, si sovrascrive anche il codice di avvio; se questo va bene, basta procedere così:

```
dd if=file_mbr of=file_di_dispositivo
```

Se invece si vuole copiare soltanto la tabella di partizione, senza sovrascrivere il codice di avvio, il comando diventa più articolato:

```
dd if=file_mbr of=file_di_dispositivo bs=1 count=64 ↩
↪skip=446 seek=446
```

Naturalmente, un errore nella scrittura del comando di recupero, sia nei valori, sia nell'indicazione corretta del file di dispositivo di destinazione, è irreversibile.

#### 19.2.1.6 Sistemazione delle partizioni Dos-FAT

«

Quando si predispongono partizioni Dos, può essere opportuno ripulire il primo settore (i primi 512 byte) della partizione, per evitare dei problemi con i programmi come '**FORMAT.EXE**', i quali potrebbero leggerlo prima di iniziare il loro lavoro, restando confusi nel caso ci fossero lì dei dati casuali. Come si intuisce, il problema non esiste se il file system Dos-FAT viene generato attraverso strumenti di GNU/Linux, ma se si realizza uno script che deve costruire automaticamente delle partizioni, tra cui anche di tipo Dos, forse è il caso di provvedere a ripulire il primo settore di ogni partizione del genere.

Supponendo di avere definito la partizione '`/dev/sda1`' per il Dos, si dovrebbe agire nel modo seguente:

```
# dd if=/dev/zero of=/dev/hda1 bs=512 count=1 [Invio]
```

Si intuisce che anche solo un piccolo sbaglio, in un'operazione del genere, comporta la cancellazione di dati in modo irreversibile.

#### 19.2.1.7 Utilizzo di «mke2fs», «mkfs.ext2», «mkfs.ext3» e «mkfs.ext4»

«

Il programma '**mke2fs**'[8] permette di creare un file system di tipo Ext2 o Ext3 in un'unità di memorizzazione. Questa viene indicata nel modo consueto, attraverso il nome del file di dispositivo corrispondente ('`/dev/…`').

```
mke2fs [opzioni] dispositivo [dimensione_in_blocchi]
```

```
mkfs.ext2 [opzioni] dispositivo [dimensione_in_blocchi]
```

```
mkfs.ext3 [opzioni] dispositivo [dimensione_in_blocchi]
```

```
mkfs.ext4 [opzioni] dispositivo [dimensione_in_blocchi]
```

La dimensione è espressa in blocchi. Se questo valore non viene specificato, '**mke2fs**' cerca di determinarlo da solo, ma ci sono situazioni in cui ciò non avviene in modo corretto, quindi può essere conveniente il fornire tale indicazione in modo esplicito.

Tabella 19.24. Alcune opzioni.

| Opzione | Descrizione |
|---|---|
| `-b` ***dimensione_del_blocco*** | Permette di definire la dimensione dei blocchi, espressa in byte.<br>La lettera «b» sta per *block*. |
| `-c` | Prima di creare il file system controlla i blocchi in modo da isolare quelli difettosi.<br>Il controllo viene eseguito in sola lettura.<br>La lettera «c» sta per *check*. |

| Opzione | Descrizione |
|---|---|
| `-i` ***byte_per_inode*** | Definisce il rapporto byte/inode. '**`mke2fs`**' crea un inode a ogni intervallo stabilito espresso in byte. Il valore predefinito è di 4 Kibyte (4096 byte) e non può essere inferiore a 1 Kibyte (1024 byte).<br>La lettera «i» sta per *inode*. |
| `-q` | Esegue l'operazione senza emettere informazioni di alcun tipo, in modo da poter essere utilizzato agevolmente all'interno di script.<br>La lettera «q» sta per *quiet*. |
| `-S` | Scrive solo il super blocco e il descrittore di gruppo. Ciò può essere utile se, sia il super blocco principale, sia quelli di riserva, sono rovinati e si intende tentare, come ultima risorsa, un recupero dei dati. In questo modo, la tabella degli inode e altre informazioni non vengono modificate. Subito dopo è necessario utilizzare il programma '**`e2fsck`**' (19.2.2.3), ma non c'è alcuna garanzia che il recupero funzioni.<br>La lettera «S» sta per *superblock*. |

Segue la descrizione di alcuni esempi.

- `#` **`mkfs.ext4 /dev/sda2`** [*Invio*]

  Crea un file system Ext4 nella partizione a cui si riferisce il file di dispositivo '`/dev/sda2`', determinandone automaticamente la dimensione.

- `#` **`mkfs.ext4 -c /dev/sda2`** [*Invio*]

Crea un file system Ext4 nella partizione a cui si riferisce il file di dispositivo '`/dev/sda2`', determinandone automaticamente la dimensione, procedendo inizialmente con un controllo dell'integrità della partizione.

Si veda la pagina di manuale *mke2fs(8)* per avere maggiori dettagli sull'uso di questo programma.

#### 19.2.1.8 Utilizzo di «mkdosfs» e «mkfs.msdos»

Il programma '**`mkdosfs`**'[9] permette di creare un file system Dos-FAT. Può essere usato per tutti i tipi di unità di memorizzazione di massa, compresi i dischetti.

```
mkdosfs [opzioni] dispositivo [dimensione_in_blocchi]
```

```
mkfs.msdos [opzioni] dispositivo [dimensione_in_blocchi]
```

Tabella 19.25. Alcune opzioni.

| Opzione | Descrizione |
|---|---|
| `-c` | Prima di creare il file system controlla i blocchi in modo da isolare quelli difettosi. Il controllo viene eseguito in sola lettura. La lettera «c» sta per *check*. |

| Opzione | Descrizione |
|---|---|
| `-i` ***numero_id*** | Permette di specificare il numero di identificazione. Si tratta di un numero a 32 bit, rappresentato in esadecimale (precisamente si tratta di otto cifre esadecimali), che diversamente verrebbe determinato in base alla data.<br>La lettera «i» sta per *id* (*identity*). |
| `-I` | Quando si utilizzano unità di memorizzazione per le quali si prevede normalmente una suddivisione in partizioni, con questa opzione è consentito inizializzare l'unità intera, senza suddivisioni in partizioni. |
| `-C` | Consente di creare e inizializzare un file-immagine. Quando si utilizza questa opzione è indispensabile fornire la dimensione in blocchi alla fine della riga di comando.<br>La lettera «C» sta per *create*. |

L'esempio seguente crea un file system Dos-FAT nella prima partizione della prima unità, dopo aver determinati automaticamente la dimensione in blocchi e lo stati di questi:

```
# mkdosfs -c /dev/sda1 [Invio]
```

Si veda anche la pagina di manuale *mkdosfs(8)* per avere maggiori dettagli.

### 19.2.1.9 Utilizzo di «mkfs»

Il programma '**mkfs**'[10] serve a rendere uniforme l'utilizzo degli altri programmi specifici per la creazione dei vari tipi di file system. In questi casi si può parlare anche di programma frontale oppure si usa il termine inglese *front-end*.

```
mkfs [-t tipo_di_file_system] [opzioni_specifiche] dispositivo ↩
↪ [dimensione_in_blocchi]
```

L'opzione '**-t**' serve per specificare il tipo di file system da creare, in questo modo '**mkfs**' sa a quale programma deve rivolgersi. Le opzioni specifiche dipendono dal tipo di file system, ovvero dal programma che si prende cura effettivamente dell'inizializzazione. Segue la descrizione di alcuni esempi.

- # **mkfs -t msdos -c /dev/sda1** [*Invio*]

  Crea un file system Dos-FAT nella prima partizione della prima unità, dopo il controllo dei blocchi.

- # **mkfs -t ext4 -c /dev/sda2** [*Invio*]

  Crea un file system Ext4 nella seconda partizione della prima unità, dopo il controllo dei blocchi.

## 19.2.2 Controllo dei file system

I dati contenuti all'interno di un file system sono organizzati in una struttura articolata e delicata. A volte, specie se succedono incidenti, conviene controllare questa struttura attraverso un programma che si occupa di risistemare le cose.

Tutte queste operazioni vengono svolte facendo riferimento ai file di dispositivo relativi. Di conseguenza, possono essere compiute solo dagli utenti che hanno i permessi di accesso necessari per il tipo di operazione da compiere.

#### 19.2.2.1 Utilizzo di «disktype»

«

Il programma '**disktype**'[11] permette di individuare facilmente le caratteristiche di un'unità di memorizzazione, comprese le partizioni presenti ed eventualmente il tipo di file system contenuto. Si utilizza senza opzioni:

```
disktype file…
```

I file da indicare come argomento di '**disktype**' possono essere file di dispositivo, oppure file-immagine di una qualche unità. Vengono mostrati alcuni esempi molto semplici, per rendere l'idea delle informazioni che si possono ottenere con questo programma:

```
# disktype /dev/sdb [Invio]

--- /dev/sdb
Block device, size 76.34 GiB (81964302336 bytes)
GRUB boot code, compat version 3.2, boot drive 0xff
DOS partition map
Partition 1: 8.066 GiB (8661187584 bytes, 16916382 sectors
from 63)
  Type 0x82 (Linux swap / Solaris)
  Linux swap, version 2, subversion 1, 4 KiB pages,
  little-endian
    Swap size 8.066 GiB (8661176320 bytes, 2114545 pages of
    4 KiB)
```

```
 Partition 2: 13.04 GiB (13999426560 bytes, 27342630 sectors
 from 16916445)
   Type 0x83 (Linux)
   Ext3 file system
     UUID 6137A3BD-D45C-46ED-ACF2-668046292FEC (DCE, v4)
     Volume size 13.04 GiB (13999423488 bytes, 3417828 blocks
     of 4 KiB)
 Partition 3: 21.10 GiB (22652421120 bytes, 44243010 sectors
 from 44259075)
   Type 0x83 (Linux)
   Ext3 file system
     UUID 316F72A3-3501-4260-A438-8873608E4CC4 (DCE, v4)
     Volume size 21.10 GiB (22652420096 bytes, 5530376 blocks
     of 4 KiB)
 Partition 4: 34.13 GiB (36643622400 bytes, 71569575 sectors
 from 88502085)
   Type 0x83 (Linux)
   Ext3 file system
     UUID CFDEA76D-72E4-4775-81C8-2860FBAB2037 (DCE, v4)
     Volume size 34.13 GiB (36641439744 bytes, 8945664 blocks
     of 4 KiB)
```

# **disktype /dev/sr0** [*Invio*]

```
--- /dev/sr0
Block device, size 158.2 MiB (165918720 bytes)
CD-ROM, 1 track, CDDB disk ID 02043801
Track 1: Data track, 158.2 MiB (165918720 bytes)
  ISO9660 file system
    Volume name "nanoLinux II"
    Application "nanoLinux II"
    Data size 158.2 MiB (165918720 bytes, 81015 blocks of
    2 KiB)
    El Torito boot record, catalog at 79733
      Bootable non-emulated image, starts at 79736, preloads
      2 KiB
```

### 19.2.2.2 Utilizzo di «dumpe2fs»

«

Il programma '**dumpe2fs**'[12] consente di avere un buon numero di informazioni relative al file system Second-extended (Ext2, Ext3 o Ext4) contenuto in un certo file di dispositivo.

```
dumpe2fs [opzioni] file_di_dispositivo
```

Tra le altre cose, il programma '**dumpe2fs**' può essere utile per localizzare le copie di sicurezza del super blocco:

```
# dumpfs /dev/md1 [Invio]

  …
  First block:              0
  Block size:               4096
  Fragment size:            4096
  …
  Group 0: (Blocks 0-32767)
    Primary superblock at 0, Group descriptors at 1-1
    Reserved GDT blocks at 2-370
```

```
  Block bitmap at 371 (+371), Inode bitmap at 372 (+372)
  Inode table at 373-875 (+373)
  0 free blocks, 16085 free inodes, 2 directories
  Free blocks:
  Free inodes: 12-16096
Group 1: (Blocks 32768-65535)
  Backup superblock at 32768, Group descriptors at 32769-32769
  Reserved GDT blocks at 32770-33138
  Block bitmap at 33139 (+371), Inode bitmap at 33140 (+372)
  Inode table at 33141-33643 (+373)
  30976 free blocks, 16096 free inodes, 0 directories
  Free blocks: 34560-65535
  Free inodes: 16097-32192
…
Group 3: (Blocks 98304-131071)
  Backup superblock at 98304, Group descriptors at 98305-98305
  Reserved GDT blocks at 98306-98674
  Block bitmap at 98675 (+371), Inode bitmap at 98676 (+372)
  Inode table at 98677-99179 (+373)
  31892 free blocks, 16096 free inodes, 0 directories
  Free blocks: 99180-131071
  Free inodes: 48289-64384
…
```

Si veda anche la pagina di manuale *dumpe2fs(8)* per avere maggiori indicazioni sull'uso del programma.

### 19.2.2.3 Utilizzo di «e2fsck», «fsck.ext2», «fsck.ext3» e «fsck.ext4»

Il programma '**e2fsck**'[13] permette di eseguire un controllo in un file system di tipo Ext2, Ext3 e Ext4, applicando le correzioni ritenute necessarie. In generale, è opportuno che il file system da controlla-

re non sia innestato, o almeno, se è innestato, è necessario che sia accessibile in sola lettura.[14]

```
e2fsck [opzioni] dispositivo
```

```
fsck.ext2 [opzioni] dispositivo
```

```
fsck.ext3 [opzioni] dispositivo
```

```
fsck.ext4 [opzioni] dispositivo
```

Tabella 19.29. Alcune opzioni.

| Opzione | Descrizione |
|---|---|
| `-b` ***super_blocco*** | Indica un super blocco alternativo al primo. Questa opzione si utilizza quando il super blocco principale è danneggiato e si vuole tentare l'accesso al file system attraverso le informazioni contenute in una delle copie del super blocco. L'indirizzo dei super blocchi di scorta può essere determinato con l'ausilio del programma '**`dumpe2fs`**', ma generalmente, ammesso che il file system sia organizzato in blocchi da 4 Kibyte (4096 byte), l'indirizzo della prima copia di sicurezza del super blocco è 32768.<br>La lettera «b» sta per *backup*. |

| Opzione | Descrizione |
|---|---|
| `-c` | Avvia a sua volta il programma '**`badblocks`**' in modo da ricercare e segnare eventuali blocchi difettosi.<br>La lettera «c» sta per *check*. |
| `-f` | Forza il controllo anche se il file system sembra in ordine (un file system che sembra non contenere errori viene definito «pulito»: *clean*).<br>La lettera «f» sta per *force*. |
| `-F` | Prima di procedere, fa in modo di scaricare la memoria cache del file system su cui si vuole intervenire.<br>La lettera «F» sta per *flush*. |
| `-n` | Esegue il controllo in sola lettura, rispondendo automaticamente '**`n`**' (no), a tutte le domande che potrebbero essere fatte. |
| `-p` | Ripara automaticamente il file system senza fare alcuna domanda.<br>La lettera «p» sta per *preen*. |
| `-y` | Risponde automaticamente '**`y`**' (*yes*), a tutte le domande che potrebbero essere fatte, in modo da permetterne l'utilizzo non interattivo attraverso uno script. |

Segue la descrizione di alcuni esempi.

- `#` **`fsck.ext4 /dev/sda2`** [*Invio*]

  Verifica il file system contenuto nella partizione a cui si riferisce il file di dispositivo '`/dev/sda2`', partendo dal presupposto che questo sia di tipo Ext4.

- # **fsck.ext4 -f /dev/sda2** [*Invio*]

  Verifica in ogni caso il file system contenuto nella partizione a cui si riferisce il file di dispositivo '/dev/sda2'. È inteso che tale file system sia di tipo Ext4.

- # **fsck.ext4 -f -b 32768 /dev/sda2** [*Invio*]

  Verifica in ogni caso il file system contenuto nella partizione a cui si riferisce il file di dispositivo '/dev/sda2', utilizzando però le informazioni del super blocco di scorta contenuto a partire dal blocco numero 32768.

Si veda la pagina di manuale *e2fsck(8)* per conoscere meglio l'uso di questo programma.

#### 19.2.2.4 Utilizzo di «dosfsck» e «fsck.msdos»

«

Il programma '**dosfsck**'[15] permette di eseguire il controllo di un file system di tipo Dos-FAT e di applicare le correzioni ritenute necessarie. In generale, è opportuno che il file system da controllare non sia innestato.[16]

```
dosfsck [opzioni] dispositivo
```

```
fsck.msdos [opzioni] dispositivo
```

Tabella 19.30. Alcune opzioni.

| Opzione | Descrizione |
|---|---|
| `-a` | Esegue automaticamente la riparazione del file system. Se esiste più di una possibilità per eseguire una correzione, viene scelta la meno distruttiva.<br>La lettera «a» sta per *automatic*. |
| `-r` | Esegue la riparazione del file system in modo interattivo, richiedendo all'utente la scelta sul tipo di correzione da attuare quando esiste più di una scelta.<br>La lettera «r» sta per *repair*. |
| `-t` | Marca i *cluster* illeggibili come difettosi.<br>La lettera «t» sta per *tag*. |

Per conoscere maggiori dettagli conviene consultare la pagina di manuale *dosfsck(8)*.

### 19.2.2.5 Utilizzo di «fsck»

Il programma '**`fsck`**' rende uniforme l'utilizzo dei programmi specifici per il controllo e la correzione dei vari tipi di file system. Si tratta di un programma frontale.

`fsck` [***opzioni***] [`-t` ***tipo_di_fs***] [***opzioni_specifiche***] ***dispositivo***...

L'opzione '**`-t`**' serve per specificare il tipo di file system da analizzare, in questo modo '**`fsck`**' sa a quale programma deve rivolgersi. Le opzioni specifiche dipendono dal tipo di file system, ovvero dal programma che si prende cura effettivamente dell'operazione.

Si veda anche la pagina di manuale *fsck(8)*.

#### 19.2.2.6 La directory «lost+found»

«

Nei file system Second-extended (Ext2, Ext3 o Ext4) si incontra normalmente la directory vuota ‘`/lost+found/`’, creata automaticamente nel momento della creazione del file system stesso. Questa directory non deve essere usata dagli utenti e deve rimanere a disposizione per i programmi che eseguono il controllo del file system, perché al suo interno vengono collocati i file che emergono in fase di correzione di un file system danneggiato, quando questi non possono essere attribuiti alla loro posizione originale. Naturalmente, l'utente che esegue questi controlli e trova file all'interno della directory ‘`/lost+found/`’, dovrebbe cercare di capire di cosa si tratta ed eventualmente di ricollocarli nelle loro posizioni appropriate nella gerarchia del file system.

### 19.2.3 Gestione di file system NTFS

«

Allo stato attuale, la gestione di un file system NTFS attraverso un sistema GNU/Linux è ancora incompleta. Pertanto, si usano strumenti particolari per accedere a questo tipo di file system. Data la diffusione del file system NTFS e, di conseguenza, la necessità frequente di convivere con questo, vale la pena di conoscere cosa si può fare concretamente attraverso un sistema GNU/Linux.

Il pacchetto di programmi NTFSprogs,[17] raccoglie degli strumenti per l'accesso ai file system NTFS (senza che questo sia stato innestato). L'utilizzo di alcuni di questi programmi viene descritto brevemente nella tabella 19.31.

Tabella 19.31. Strumenti di NTFSprogs.

| Modello sintattico | Descrizione |
|---|---|
| `mkntfs` [*opzioni*] *dispositivo* [*n_settori*]<br>`mkfs.ntfs` [*opzioni*] *dispositivo* [*n_settori*] | Crea un file system NTFS nell'unità o nella partizione corrispondente al file di dispositivo specificato. |
| `ntfsresize` [*opzioni*] `--info` *dispositivo*<br>`ntfsresize` [*opzioni*] ↩<br>↪ `--size` *dimensione* *dispositivo* | Verifica o modifica la dimensione della partizione; il valore della dimensione è espresso in byte. |

Eventualmente si veda anche la sezione 19.11.6.

### 19.2.4 Compatibilità dei nomi tra un file system e l'altro

Un problema comune quando si vogliono copiare dati in un file system diverso da quello solito, consiste nel verificare se ci possono essere problemi con i nomi usati per i file e le directory. Un programma molto semplice permette di verificare la compatibilità dei nomi con un file system di tipo Dos-FAT (con nomi «8.3») e SYSV (al massimo di 14 caratteri):

```
doschk nome_file...
```

```
echo nome_file... | doschk
```

Per comprendere il funzionamento di '**doschk**'[18] è sufficiente un esempio:

```
$ ls | doschk[Invio]
```

```
The following files are not valid DOS file names:
home.html.bak - too many dots
hw-mb-pci-msi-ms5169-1998-legenda.pnm.gz - too many dots
prova.fig.bak - too many dots

The following resolve to the same DOS file names:
HOME.HTM        : home.html
                  home.html.bak
MAIL            : Mail
                  mail
PROVA.FIG       : prova.fig
                  prova.fig.bak

The following file names are too long for SysV:
blick-electric : blick-electric.gif
contratti-inte : contratti-internet
hw-mb-pci-msi- : hw-mb-pci-msi-ms5169-1998-legenda.pnm.gz
```

## 19.3 Memoria di massa: gestione delle partizioni

«

Le partizioni che contengono già un file system, possono essere rimaneggiate attraverso l'uso di strumenti specializzati per la loro copia, per lo spostamento e per il ridimensionamento. Tali strumenti sono utili quando sono in grado di tenere conto del tipo di file system e di fare tutti gli adattamenti necessari di conseguenza.

### 19.3.1 Gpart

«

Il programma '**gpart**'[19] consente di scandire un'unità di memorizzazione o un file-immagine per ricostruire la sua suddivisione in partizioni:

```
gpart [opzioni] file_di_dispositivo
```

```
gpart -C cilindri,testine,settori -g [altre_opzioni] file_immagine
```

Per poter fornire informazioni corrette, il programma deve conoscere la geometria virtuale dell'unità da controllare, cosa che ottiene normalmente dal sistema operativo; tuttavia, quando il valore ottenuto automaticamente non è valido, oppure quando si tratta di un file-immagine, tale informazione deve essere fornita attraverso le opzioni della riga di comando, come si vede nel modello proposto a proposito della lettura di un file-immagine.

Si osservi che la ricerca eseguita da '**gpart**' è più semplice e più attendibile se la suddivisione in partizioni dell'unità da scandire non prevede la presenza di partizioni «logiche»; in pratica, va meglio se ci sono solo partizioni primarie che si annotano nel primo settore di avvio.

Viene mostrato un solo esempio, molto semplice, di un'unità che ha una tabella delle partizioni coerente con la suddivisione effettiva; per un approfondimento dell'uso del programma si può consultare la pagina di manuale *gpart(8)*.

```
# gpart /dev/sdb [Invio]
```

Si richiede la scansione della seconda unità di memorizzazione; in questo caso si ottiene il risultato seguente:

```
Begin scan...
Possible partition(Linux swap), size(8259mb), offset(0mb)
Possible partition(Linux ext2), size(13350mb), offset(8259mb)
```

```
Possible partition(Linux ext2), size(21603mb), offset(21610mb)
Possible partition(Linux ext2), size(34944mb), offset(43213mb)
End scan.

Checking partitions...
Partition(Linux swap or Solaris/x86): primary
Partition(Linux ext2 filesystem): primary
Partition(Linux ext2 filesystem): primary
Partition(Linux ext2 filesystem): primary
Ok.

Guessed primary partition table:
Primary partition(1)
   type: 130(0x82)(Linux swap or Solaris/x86)
   size: 8259mb #s(16916376) s(63-16916438)
   chs:  (0/1/1)-(1023/15/63)d (0/1/1)-(16782/2/57)r

Primary partition(2)
   type: 131(0x83)(Linux ext2 filesystem)
   size: 13350mb #s(27342624) s(16916445-44259068)
   chs:  (1023/15/63)-(1023/15/63)d (16782/3/1)-(43907/12/57)r

Primary partition(3)
   type: 131(0x83)(Linux ext2 filesystem)
   size: 21603mb #s(44243008) s(44259075-88502082)
   chs:  (1023/15/63)-(1023/15/63)d (43907/13/1)-(87799/10/61)r

Primary partition(4)
   type: 131(0x83)(Linux ext2 filesystem)
   size: 34944mb #s(71565312) s(88502085-160067396)
   chs:  (1023/15/63)-(1023/15/63)d (87799/11/1)-(158797/0/21)r
```

### 19.3.2 Parted

La gestione delle partizioni di un'unità di memorizzazione, in un sistema GNU/Linux, può essere fatta con Parted,[20] il quale consente anche il loro spostamento, il ridimensionamento e l'inizializzazione, con i file system comuni nei sistemi GNU/Linux. L'utilizzo di Parted richiede comunque una buona conoscenza del problema e di conseguenza conviene prima fare esperienza con gli strumenti tradizionali.

Per svolgere il suo compito, Parted richiede di specificare inizialmente il file di dispositivo corrispondente all'unità nella quale vanno gestite le partizioni (può trattarsi anche di un file-immagine che rappresenta un'unità di memorizzazione suddivisa o da suddividere in partizioni); successivamente, attraverso dei comandi si fa riferimento alle partizioni indicando un numero, il quale rappresenta precisamente il numero secondario (*minor number*) del file di dispositivo corrispondente. Per esempio, volendo fare riferimento a una partizione, il cui file di dispositivo è normalmente '`/dev/sda2`', con i comandi di Parted si deve specificare il numero 2, che corrisponde al numero secondario di tale file:

```
$ ls -l /dev/sda2 [Invio]

brw-rw----  1 root disk 8, 2 2010-01-01 11:11 /dev/sda2
```

Parted si avvia attraverso l'eseguibile '**`parted`**', il quale ha due modalità di funzionamento: una interattiva e l'altra no.

```
parted [opzioni] [file_di_dispositivo]
```

```
parted [opzioni] file_di_dispositivo comando_e_argomenti
```

Nel primo modello sintattico si vede l'avvio di Parted per funzionare in modo interattivo; nel secondo, aggiungendo in coda il comando desiderato, il programma smette di funzionare al termine dell'esecuzione dello stesso.

È da osservare che il file di dispositivo, il quale appare formalmente come facoltativo nella riga di comando, serve a specificare su quale unità si intende lavorare; se non viene specificato all'avvio, occorre poi usare il comando '**select**' per indicarlo durante il funzionamento in modalità interattiva.

Le opzioni previste per l'avvio di Parted sono utili solo in poche occasioni; in particolare, merita di essere ricordata l'opzione '**-s**', ovvero '**--script**', con la quale si vuole precisare che il programma non deve tentare di interagire con l'utente (presumibilmente perché viene avviato all'interno di uno script).

Diversi comandi di Parted fanno riferimento a una porzione dell'unità di memorizzazione, individuata da un punto iniziale e un punto finale. Per delimitare questi intervalli, si fa riferimento a dimensioni espresse in mebibyte (simbolo «Mibyte», ovvero multipli di 1 048 576 byte). Al contrario, con programmi tradizionali per la gestione delle partizioni, si deve fare riferimento ai cilindri, ma questo approccio è stato evitato volutamente nella realizzazione del programma.

Tabella 19.35. Alcuni comandi di Parted.

| Comando | Descrizione |
|---|---|
| `quit` | Termina il funzionamento del programma. |
| `help` [***comando***] | Mostra l'elenco dei comandi disponibili, oppure mostra la sintassi del comando specificato. |
| `select` ***file*** | Seleziona l'oggetto su cui intervenire; potrebbe trattarsi di un file di dispositivo che fa riferimento a un'unità di memorizzazione reale, oppure un file-immagine che la rappresenta. In ogni caso, con questo comando, si individua qualcosa che viene suddiviso in partizioni. |
| `print` | Mostra l'elenco delle partizioni presenti nell'oggetto selezionato (già nella riga di comando oppure attraverso il comando '**`select`**'). |
| `mklabel loop`\|`msdos`\|`pc98`\|… | Crea una tabella di partizioni secondo lo schema specificato dall'argomento del comando. La parola chiave '**`loop`**' fa riferimento a un'unità senza tabella di partizioni; il tipo '**`pc98`**' è una versione un po' più evoluta della tabella tradizionale per sistemi Dos. Sono disponibili altre parole chiave per altrettante tabelle di partizione differenti, che però qui sono state omesse nel modello. |

| Comando | Descrizione |
|---|---|
| `mkpart` ***tipo*** [***file_system***] ↩<br>↪ ***inizio*** ***fine*** | Crea una partizione. Il tipo è rappresentato da una parola chiave come '**`primary`**', '**`extended`**' o '**`logical`**', tenendo conto che non tutti gli schemi di partizionamento prevedono partizioni estese e logiche. Il file system, facoltativo, viene specificato con una parola chiave, come per esempio '**`ext2`**', '**`fat32`**', '**`fat16`**', '**`HFS`**', '**`linux-swap`**'. |
| `mkfs` ***n*** ***file_system*** | Inizializza una partizione, specificata attraverso il numero secondario (***n***), con il file system indicato alla fine del comando da una parola chiave analoga a quella di '**`mkpart`**'. |
| `mkpartfs` ***tipo*** ***file_system*** ***inizio*** ***fine*** | Crea una partizione e la inizializza. Si utilizza come il comando '**`mkpart`**', con la differenza che la partizione viene inizializzata. |
| `check` ***n*** | Controlla che il file system contenuto nella partizione individuata dal numero secondario ***n*** sia privo di errori. |

| Comando | Descrizione |
|---|---|
| `set` ***n opzione*** `on\|off` | Consente di attivare (‘**`on`**’) o disattivare (‘**`off`**’) una certa opzione booleana, associata alla partizione del numero secondario ***n***. Le opzioni disponibili dipendono dal tipo di schema di partizionamento usato. |
| `set` ***n*** `boot on\|off` | Attiva o disattiva l’opzione di «avvio». Con i sistemi Dos, l’attivazione di questa opzione rende avviabile la partizione relativa. |
| `set` ***n*** `lba on\|off` | Attiva o disattiva l’opzione con cui si specifica che lo schema di accesso all’unità è di tipo LBA (*Linear block addressing*). |
| `resize` ***n inizio fine*** | Ridimensiona la partizione individuata dal numero secondario ***n***, in modo che inizi e finisca nell’intervallo indicato. Il ridimensionamento può avvenire solo se la partizione contiene un file system compatibile, tenendo conto che, in alcuni casi, può essere impossibile modificare il punto di inizio della partizione, rispetto alla sua collocazione precedente. |

| Comando | Descrizione |
|---|---|
| `move` ***n inizio*** [***fine***] | Sposta la partizione individuata dal numero secondario ***n***, in modo che inizi e finisca nell'intervallo indicato. Se la posizione finale viene indicata, si ottiene un ridimensionamento della partizione, altrimenti questa rimane tale e quale. Con alcuni file system è necessario che la destinazione sia più grande dell'origine. |
| `rm` ***n*** | Cancella la partizione individuata dal numero secondario ***n*** (temporaneamente il contenuto della partizione rimane presente, anche se non più raggiungibile). Si osservi che in alcuni casi, la cancellazione di una partizione provoca la rinumerazione di quelle successive. |
| `cp` ***m n***<br><br>`cp` ***file_origine m n*** | Copia la partizione ***m*** nella partizione ***n***. Nel secondo modello, si specifica il file, o il file di dispositivo, che contiene la partizione di origine, così da consentire la copia da un'unità diversa o da un file-immagine diverso. |

I comandi di Parted sono «semplici», ma richiedono la conoscenza di ciò che si sta facendo, soprattutto delle caratteristiche dello schema di partizionamento utilizzato. Tanto per fare un esempio, il co-

mando '**mkpart**' richiede, eventualmente, l'indicazione di un tipo di file system, ma questa informazione serve in modo indiretto, per determinare il codice da attribuire al tipo di partizione. Per esempio, se con il comando '**mkpart**' si indicasse un file system di tipo '**ext3**', si otterrebbe una partizione di tipo $83_{16}$, ovvero di tipo «Linux».

Vengono descritti alcuni esempi, che fanno parte di una sequenza coerente di operazioni, dove Parted viene usato in modo interattivo. Questi esempi non esauriscono le potenzialità di Parted; per maggiori dettagli e per conoscere lo stato dello sviluppo di questo programma, occorre fare riferimento alla documentazione originale.

- `# ` **`parted /dev/sda`** [*Invio*]

  Si avvia il programma specificando di intervenire con i comandi nell'unità corrispondente al file di dispositivo '`/dev/sda`'.

```
GNU Parted 1.6.11 with HFS shrink patch 12.5
Copyright (C) 1998 - 2004 Free Software Foundation, Inc.
This program is free software, covered by the GNU General
Public License.

This program is distributed in the hope that it will be
useful, but WITHOUT ANY WARRANTY; without even the implied
warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR
PURPOSE.  See the GNU General Public License for more
details.

Using /dev/sda
```

  Inizia quindi il funzionamento interattivo di Parted, caratterizzato dall'invito seguente:

```
(parted)
```

- `(parted) ` **`mklabel msdos`** [*Invio*]

Crea uno schema di partizionamento conforme a quello usato dal sistema operativo Dos.

- (parted) **mkpartfs primary fat16 0 100**[*Invio*]

  Crea una partizione primaria con un file system di tipo Dos-FAT 16, dall'inizio (zero) fino ai primi 100 Mibyte circa. La partizione viene inizializzata.

- (parted) **print**[*Invio*]

  Mostra lo stato attuale delle partizioni:

```
Disk geometry for /dev/sda: 0.000-158.277 megabytes
Disk label type: msdos
Minor    Start       End     Type      Filesystem  Flags
1          0.016   100.000  primary   fat16       lba
```

- (parted) **mkpartfs primary ext4 100 159**[*Invio*]

  Crea una partizione primaria di tipo Ext4, subito dopo i primi 100 Mibyte, fino a raggiungere la posizione di 159 Mibyte circa. Questo esempio è collegato a quelli precedenti, pertanto la seconda partizione occupa tutto lo spazio rimanente nell'unità di memorizzazione.

- (parted) **print**[*Invio*]

  Mostra lo stato attuale delle partizioni:

```
Disk geometry for /dev/sda: 0.000-158.277 megabytes
Disk label type: msdos
Minor    Start       End     Type      Filesystem  Flags
1          0.016   100.000  primary   fat16       lba
2        100.000   158.250  primary   ext4
```

- (parted) **rm 1**[*Invio*]

Cancella la prima partizione.

- `(parted)` **`print`**[*Invio*]

  Mostra lo stato attuale delle partizioni:

```
Disk geometry for /dev/sda: 0.000-158.277 megabytes
Disk label type: msdos
Minor    Start       End     Type      Filesystem  Flags
2        100.000    158.250  primary   ext4
```

- `(parted)` **`move 2 0`**[*Invio*]

  Sposta la seconda partizione, nella posizione iniziale dell'unità di memorizzazione, lasciando intendere che la posizione finale sia quella necessaria a contenere la dimensione originale:

```
End? [58.2495]? [Invio]

  raw block copying...
```

- `(parted)` **`print`**[*Invio*]

  Mostra lo stato attuale delle partizioni:

```
Disk geometry for /dev/sda: 0.000-158.277 megabytes
Disk label type: msdos
Minor    Start       End     Type      Filesystem  Flags
2          0.016     58.312  primary   ext4
```

- `(parted)` **`resize 0 70`**[*Invio*]

  Ingrandisce la partizione, estendendola fino ai primi 70 Mibyte circa.

- `(parted)` **`print`**[*Invio*]

  Mostra lo stato attuale delle partizioni:

```
Disk geometry for /dev/sda: 0.000-158.277 megabytes
Disk label type: msdos
Minor    Start       End     Type      Filesystem  Flags
2          0.016     70.000  primary   ext4
```

- `(parted)` **`mkpart primary 70 159`**[*Invio*]

  Crea una partizione primaria, senza specificare il tipo di file system che si vuole usare al suo interno, nello spazio rimanente dell'unità di memorizzazione.

- `(parted)` **`print`**[*Invio*]

  Mostra lo stato attuale delle partizioni:

```
Disk geometry for /dev/sda: 0.000-158.277 megabytes
Disk label type: msdos
Minor    Start       End     Type      Filesystem  Flags
2          0.016     70.000  primary   ext4
1         70.000    158.250  primary
```

- `(parted)` **`cp 2 1`**[*Invio*]

  Copia la partizione 2 nella partizione 1, estendendo il file system nella destinazione.

```
raw block copying...
growing file system...
```

- `(parted)` **`print`**[*Invio*]

  Mostra lo stato attuale delle partizioni:

```
Disk geometry for /dev/sda: 0.000-158.277 megabytes
Disk label type: msdos
Minor    Start       End     Type      Filesystem  Flags
2          0.016     70.000  primary   ext4
1         70.000    158.250  primary   ext4
```

- `(parted)` **`quit`** [*Invio*]

  Termina il funzionamento del programma.

### 19.3.3 Partimage

Partimage[21] è un programma che può essere usato in modo interattivo o meno, per la copia e la riproduzione delle partizioni di un'unità di memorizzazione di massa. In particolare, la copia di una partizione implica normalmente anche l'archiviazione del primo settore dell'unità di memorizzazione (MBR).

```
partimage [opzioni_e_argomenti]
```

L'eseguibile '**`partimage`**' va avviato con i privilegi necessari per accedere alle partizioni; in pratica, va usato in qualità di utente '**`root`**'. Se non si usano opzioni o argomenti di alcun tipo, si ottiene un funzionamento interattivo.

Nelle sezioni successive viene mostrato l'uso di Partimage in modo interattivo; per un uso diverso, conviene consultare la pagina di manuale *partimage(1)* ed eventualmente anche *partimaged(8)* per la gestione dell'archiviazione delle partizioni attraverso la rete.

#### 19.3.3.1 Archiviazione

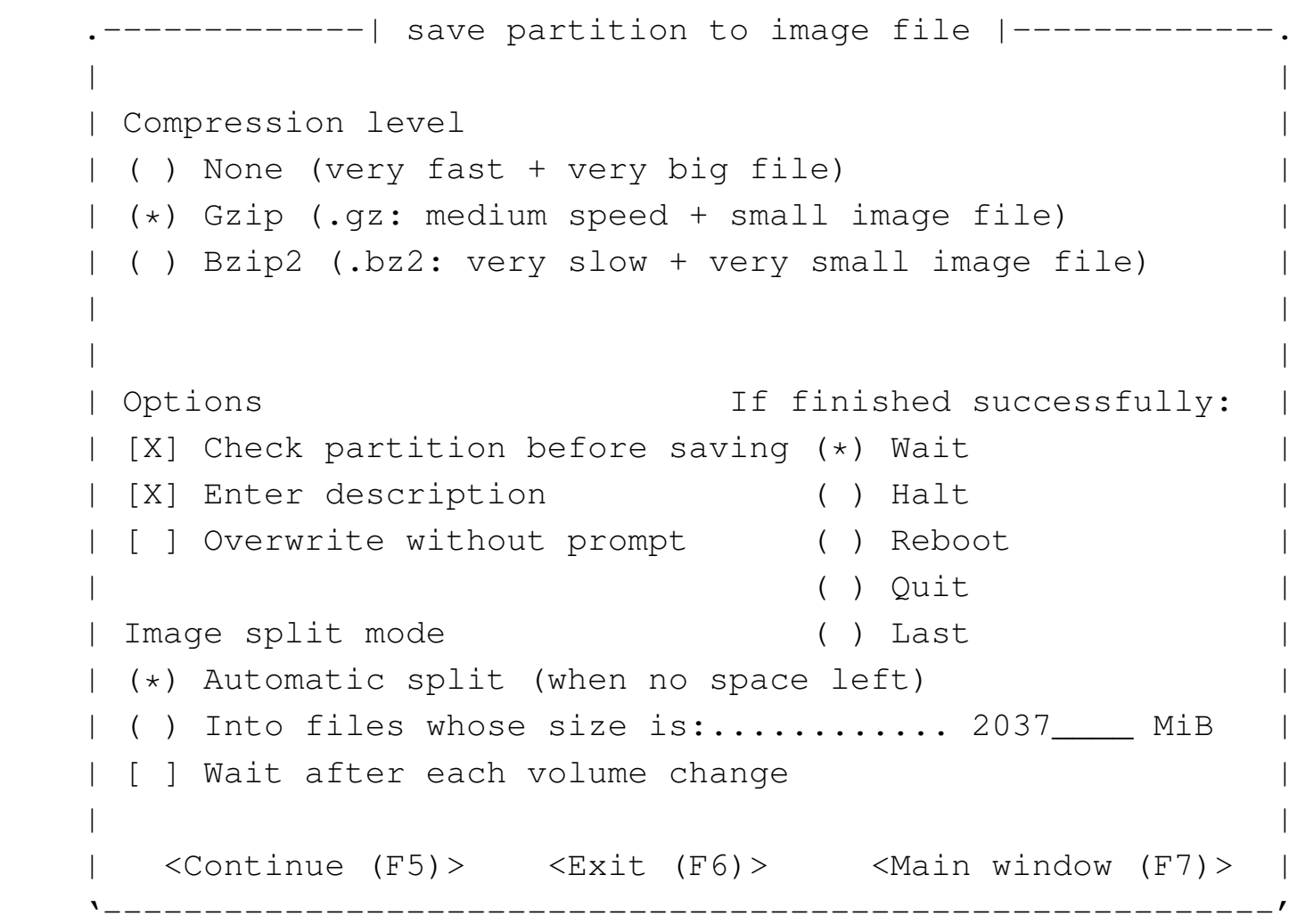
Viene mostrato un esempio di archiviazione di una partizione in modo interattivo; per la precisione, si archivia la partizione '`/dev/sda1`':

`#` **`partimage`** [*Invio*]

```
.-----------------| Partition Image 0.6.4 |----------------.
| * Partition to save/restore                              |
|   hdd7                          ext3fs       15.38 GiB   |
|   hdd8                          ext3fs       24.03 GiB   |
|   hdd9                          ext3fs       39.41 GiB   |
|   md0                           ext3fs       14.42 GiB   |
| # sda1 ######################## ext3fs ##### 15.32 GiB ## |
|   sda2                          ext3fs       53.62 GiB   |
|   sda4                          swap (v1)    7.38 GiB    |
|                                                          |
| * Image file to create/use                               |
|   [/tmp/immagine_____________________________________]   |
|                                                          |
| Action to be done:                        <Next (F5)>    |
| (*) Save partition into a new image file                 |
| ( ) Restore partition from an image file <About>         |
| ( ) Restore an MBR from the imagefile                    |
|                                           <Exit (F6)>    |
| [ ] Connect to server                                    |
|     IP/name of the server: _________________ Port: 4025__ |
|     [X] Encrypt data on the network with SSL             |
`----------------------------------------------------------'
```

Attraverso una barra di scorrimento si seleziona '**sda1**', come si vede nella schermata di esempio; inoltre, si specifica il nome del file in cui salvare l'immagine della partizione: in questo caso si tratta di '`/tmp/immagine`', ma in realtà, il nome esatto contiene un'estensione numerica ('`.000`').

Una volta compilata la maschera iniziale, come nell'esempio, si passa alla maschera successiva premendo il tasto [*F5*]:




In questa fase viene richiesto, in particolare, di specificare il metodo di compressione e se si vuole ottenere un file unico o una serie di file di dimensione stabilita. Anche qui, si prosegue con il tasto [*F5*]:

```
.-----------------| Partition description |----------------.
|                                                           |
| You can enter a description of the saved partition:       |
|                                                           |
| [Prima partizione del disco USB esterno _______________] |
|                                                           |
|                           .----.                          |
|                           | Ok |                          |
|                           `----'                          |
|                                                           |
`-----------------------------------------------------------'
```

Conviene mettere una descrizione, per annotare di che cosa si tratta, come si vede nell'esempio. Si conferma l'inizio dell'archiviazione con il pulsante grafico OK.

```
.-------------| Ext3fs informations |--------------.
|                                                  |
| Space usage:..................2 %               |
| Used space.....................374.71 MiB       |
| Free space.....................14.95 GiB        |
| Bitmap size....................1.92 MiB         |
| Label.........................                   |
| Number of groups..............123                |
| First block...................0                  |
| Ext3fs journalization.........yes                |
| Sparse super block............yes                |
| Large files support...........no                 |
| File system revision..........1                  |
|                       .----.                     |
|                       | Ok |                     |
|                       `----'                     |
`--------------------------------------------------'
```

Terminato il controllo della partizione, si conferma ancora con il

pulsante grafico OK:

```
.-------------| save partition to image file |-------------.
|Partition to save:.........../dev/sda1                     |
|Size of the Partition:.......15.32 GiB = 16450527744 bytes|
|Current image file:........../tmp/immagine.000             |
|Image file size:.............56.00 KiB                     |
|Available space for image:...17.51 GiB = 18796441600 bytes|
|Detected file system:........ext3fs                        |
|Compression level:...........gzip                          |
|                                                           |
|                                                           |
|Time elapsed:................3sec                         |
|Estimated time remaining:....21sec                         |
|Speed:.......................930.00 MiB/minn               |
|Data copied:.................46.50 MiB / 374.71 MiBB      |
|                                                           |
|                                                           |
|                                                           |
| ######_________________________12%_____________________ 12 %  |
|                                                           |
`-----------------------------------------------------------'
```

Come si può vedere, la partizione viene archiviata nel file ‘`/tmp/immagine.000`’; se dovessero essere utilizzati più file, l’estensione progredirebbe: ‘`.001`’, ‘`.002`’,...

Al termine appare una schermata di conferma.

```
.------------| Success |-------------.
|                                     |
| Operation  successfully finished:   |
|                                     |
| Time elapsed:...27sec               |
| Speed:.........832.69 MiB/min       |
| Data copied:....374.71 MiB          |
|                                     |
|              .----.                 |
|              | Ok |                 |
|              `----'                 |
|                                     |
`-------------------------------------'
```

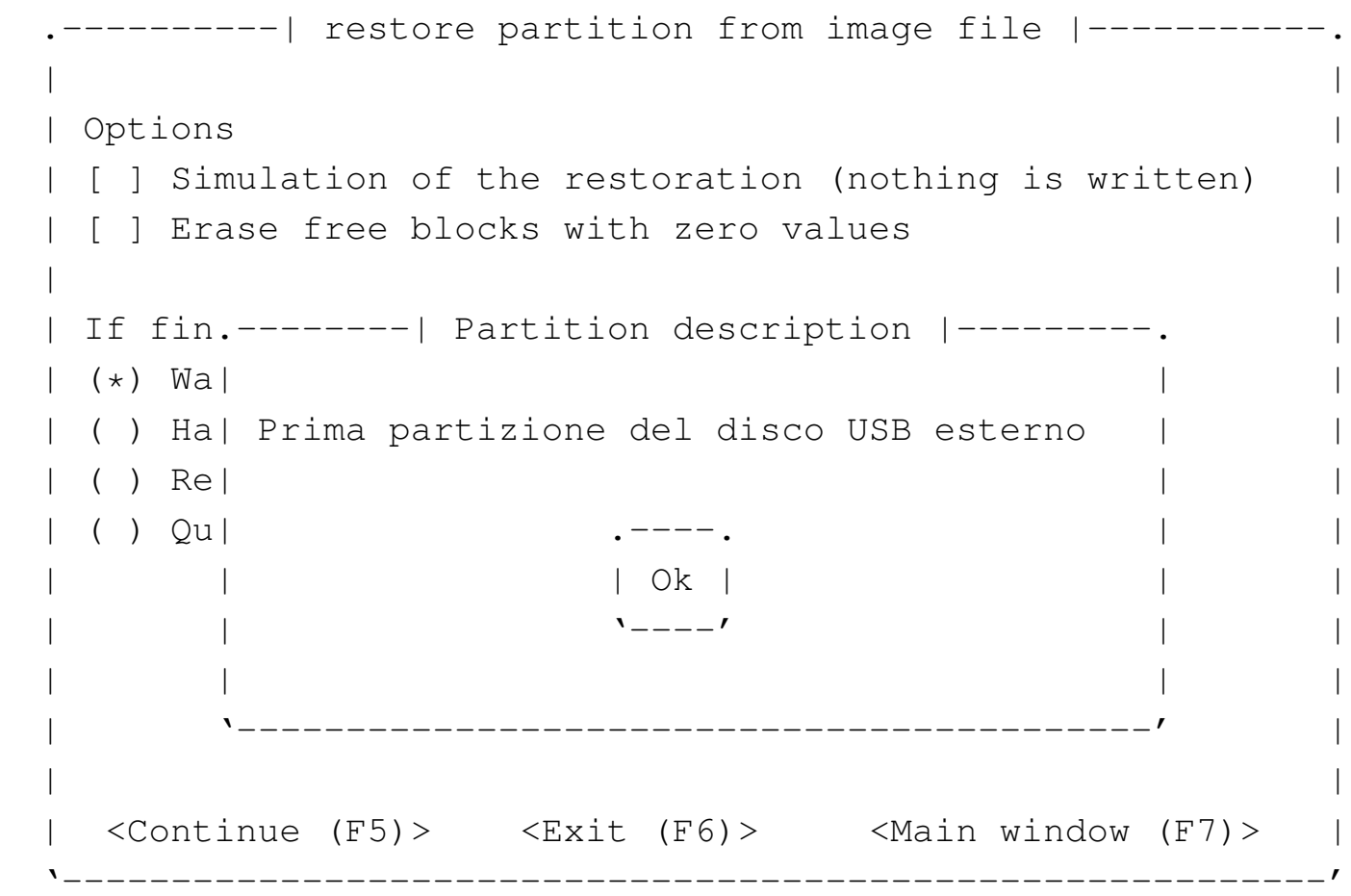
### 19.3.3.2 Ripristino

Viene mostrato un esempio di ripristino di una partizione in modo interattivo; per la precisione, si ripristina un file nella partizione '`/dev/sda2`', più grande rispetto a quella originaria:

# **partimage**[*Invio*]

```
.----------------| Partition Image 0.6.4 |-----------------.
| * Partition to save/restore                              |
|   hdd7                        ext3fs       15.38 GiB     |
|   hdd8                        ext3fs       24.03 GiB     |
|   hdd9                        ext3fs       39.41 GiB     |
|   md0                         ext3fs       14.42 GiB     |
|   sda1                        ext3fs       15.32 GiB     |
| # sda2 ###################### ext3fs ##### 53.62 GiB ##  |
|   sda4                        swap (v1)    7.38 GiB      |
|                                                          |
| * Image file to create/use                               |
|   [/tmp/immagine.000_____________________________________]  |
|                                                          |
| Action to be done:                        <Next (F5)>    |
| ( ) Save partition into a new image file                 |
| (*) Restore partition from an image file  <About>        |
| ( ) Restore an MBR from the imagefile                    |
|                                           <Exit (F6)>    |
| [ ] Connect to server                                    |
|     IP/name of the server: ________________ Port: 4025__ |
|     [X] Encrypt data on the network with SSL             |
`----------------------------------------------------------'
```

Come si vede, si prende il primo e unico file ottenuto dall'archiviazione eseguita nella sezione precedente, per ripristinare i dati nella partizione '`/dev/sda2`'. Si prosegue premendo il tasto [*F5*]:

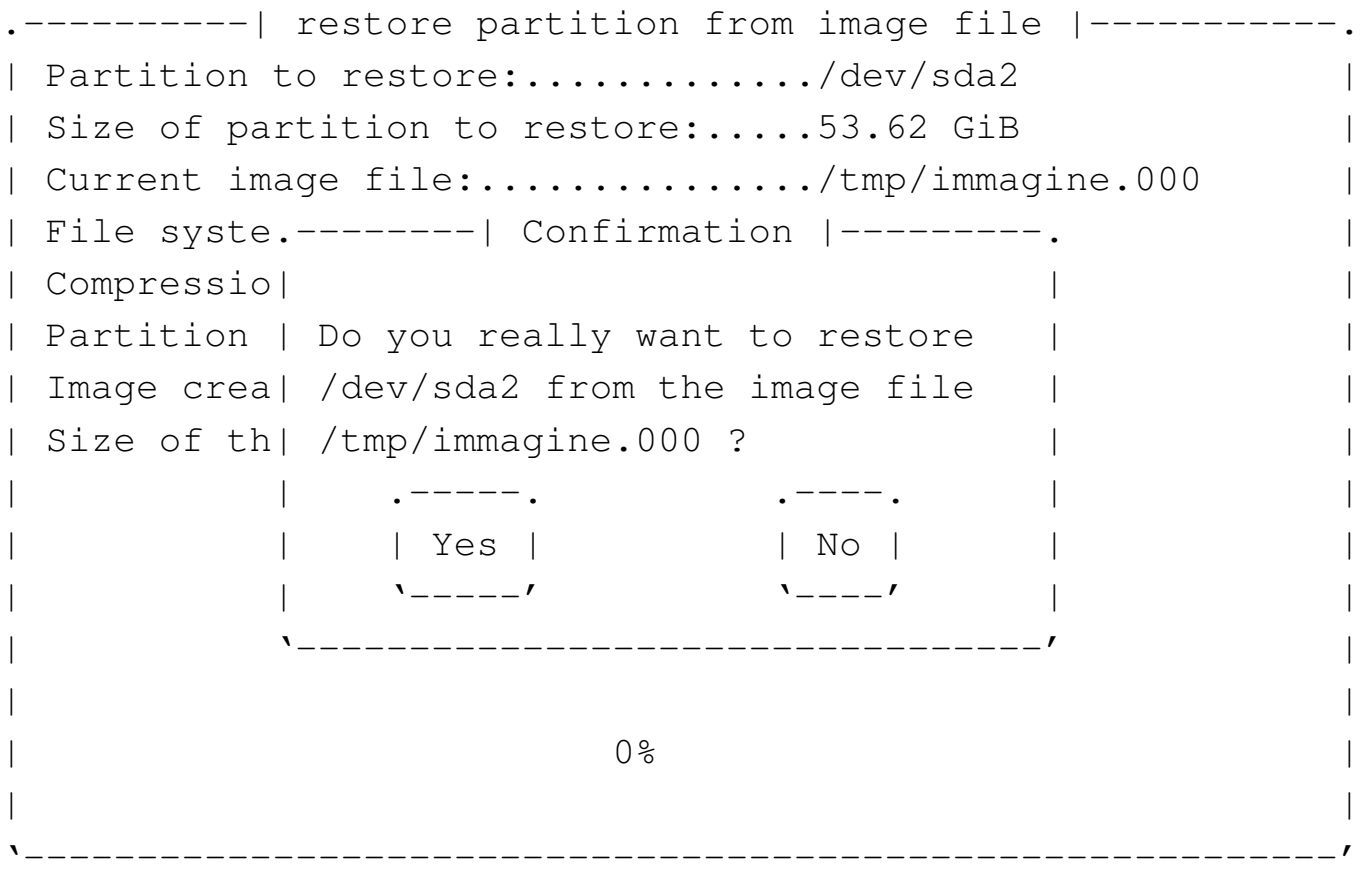



Viene mostrata la descrizione del file, prima di procedere.

```
.----------| restore partition from image file |-----------.
|                                                          |
| Options                                                  |
| [ ] Simulation of the restoration (nothing is written)   |
| [ ] Erase free blocks with zero values                   |
|                                                          |
| If finished successfully:                                |
| (*) Wait                                                 |
| ( ) Halt                                                 |
| ( ) Reboot                                               |
| ( ) Quit                                                 |
|                                                          |
|                                                          |
|   <Continue (F5)>   <Exit (F6)>     <Main window (F7)>   |
`----------------------------------------------------------'
```

Nella maschera finale si può ancora richiedere di simulare il recupero; in questo caso viene eseguito realmente. Si procede con il tasto [*F5*]:

```
.--------------| Ext3fs informations |-------------.
|                                                  |
| Space usage:...................2 %               |
| Used space.....................374.71 MiB        |
| Free space.....................14.95 GiB         |
| Bitmap size....................1.92 MiB          |
| Label..........................                  |
| Number of groups..............123                 |
| First block...................0                   |
| Ext3fs journalization.........yes                 |
| Sparse super block............yes                 |
| Large files support...........no                  |
| File system revision..........1                   |
|                      .----.                      |
|                      | Ok |                      |
|                      `----'                      |
`--------------------------------------------------'
```


restore partition from image file
Partition to restore:/dev/sda2
Size of partition to restore:53.62 GiB
Current image file:/tmp/immagine.000
File syste
Confirmation
Compressio
Partition
Do you really want to restore
/dev/sda2 from the image file
Image crea
/tmp/immagine.000 ?
Size of th
Yes
No
0%

```
.----------| restore partition from image file |-----------.
| Partition to restore:............../dev/sda2              |
| Size of partition to restore:.....53.62 GiB = 57576960000|
| Current image file:................/tmp/immagine.000      |
| File system:.....................ext3fs                   |
| Compression level:...............gzip                     |
| Partition was on device:........../dev/sda1              |
| Image created on:................Tue Apr 25 11:12:13 200|
| Size of the original partition:...15.32 GiB = 16450527744|
|                                                           |
| Time elapsed:....................10m:41sec               |
| Estimated time remaining:.........16m:35sec               |
| Speed:...........................13.73 MiB/minn          |
| Data copied:.....................146.73 MiB / 374.71 MiB|
|                                                           |
| ########################__39%_____________________ 39 %  |
|                                                           |
`-----------------------------------------------------------'
```

```
.----------| restore partition from image file |-----------.
| Partition to restore:............../dev/sda2              |
| Size of partition to restore:.....53.62 GiB = 57576960000|
| Current image file:.............../tmp/immagine.000      |
| File syst.------------| Success |------------.           |
| Compressi|                                   |           |
| Partition| Operation  successfully finished:  |           |
| Image cre|                                   |           |
| Size of t| Time elapsed:...10m:52sec          |           |
|          | Speed:..........34.48 MiB/min      |           |
| Time elap| Data copied:....374.71 MiB         |           |
| Estimated|                                   |           |
| Speed:...|              .----.               |           |
| Data copi|              | Ok |               |           |
|          |              `----'               |           |
|          `-----------------------------------'           |
|                      100%                        100 % |
|                                                          |
`----------------------------------------------------------'
```

A questo punto, dal momento che i dati recuperati riproducono una partizione più piccola dello spazio effettivo, se si dispone degli strumenti appropriati, si può tentare di estenderne le dimensioni allo spazio disponibile:

```
# e2fsck -f /dev/sda2 [Invio]

e2fsck 1.39-WIP (31-Dec-2005)
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information
/dev/sda2: 12/2011296 files (0.0% non-contiguous), ↩
↪95926/4016242 blocks
```

```
# resize2fs /dev/sda2 [Invio]

  resize2fs 1.39-WIP (31-Dec-2005)
  Resizing the filesystem on /dev/sda2 to 14056875 (4k) blocks.
  The filesystem on /dev/sda2 is now 14056875 blocks long.
```

## 19.4 Attivazione e utilizzo

«

Per poter accedere a un file system di un sistema Unix, occorre dare dei comandi precisi e generalmente conviene intervenire nella configurazione. Tuttavia non esistono comandi od opzioni di configurazione standard per tutti i sistemi Unix e ogni sistema ha le proprie particolarità; pertanto, qui si considerano specificatamente i sistemi GNU/Linux.

### 19.4.1 Attivazione dei file system

«

Per poter accedere ai dati di un'unità di memorizzazione organizzata con un file system, è necessario prima innestare (*mount*) il suo file system in quello globale.

Prima di estrarre una di queste unità, o comunque, prima di poter spegnere un elaboratore, occorre eseguire l'operazione opposta di distacco (*unmount*).

In un sistema GNU/Linux devono essere necessariamente innestati il file system principale (*root*) e i file system virtuali '`/proc/`' e '`/sys/`', i quali però non fanno capo ad alcuna unità fisica.

Se si utilizzano partizioni di scambio per la gestione della memoria virtuale, queste devono essere attivate con un'operazione concettualmente simile a quella dell'innesto di un file system, anche se poi non appaiono nella struttura generale del file system globale.

### 19.4.1.1 Tipi di file system

Quando si innesta un file system è necessario che il modo con cui questo è organizzato (cioè il tipo) sia riconoscibile e gestito dal kernel. Nella tabella 19.61, sono elencati i nomi che identificano i tipi di file system principali, riconoscibili da un kernel Linux.

Tabella 19.61. Elenco dei nomi di file system principali utilizzabili con un kernel Linux.

| Tipo di file system | Descrizione |
|---|---|
| `minix` | Minix 1 |
| `ext2`<br>`ext3`<br>`ext4` | Second-extended |
| `umsdos` | GNU/Linux su Dos-FAT |
| `msdos` | Dos-FAT (nomi 8.3) |
| `vfat` | Dos-VFAT (nomi lunghi) |
| `nfs` | NFS o file system di rete |
| `iso9660` | dischi ottici che non contengano file di dimensioni maggiori di 4 Gibyte |
| `smbfs` | SMB (rete NetBIOS-TCP/IP) |
| `affs` | Amiga fast file system |
| `adfs` | Acorn disc filing system |

| Tipo di file system | Descrizione |
|---|---|
| `sysv` | UNIX System V file system |
| `cramfs` | Compressed ROM file system (sola lettura) |
| `squashfs` | file system compresso in sola lettura |
| `hfs`<br>`hfsplus` | Apple Macintosh file system |
| `hpfs` | OS/2 HPFS |
| `jfs` | IBM Journaled file system |
| `ntfs` | Microsoft NTFS |
| `reiserfs` | ReiserFS |
| `tempfs` | file system in un disco RAM, che si estende e si contrae automaticamente |
| `xfs` | IRIX XFS |
| `ufs` | BSD file system |
| `proc`<br>`sysfs` | file system virtuali '`/proc/`' e '`/sys/`' |
| `swap` | partizione di scambio |

### 19.4.1.2 Implicazioni legate all'innesto di un file system

L'innesto di un file system implica l'acquisizione di un file system estraneo in quello generale. Questo fatto può far sorgere problemi di sicurezza e di compatibilità con il sistema. L'elenco seguente dovrebbe dare l'idea di alcuni dei problemi connessi.

- Il file system estraneo potrebbe non essere sicuro, di conseguenza si pone il problema di:
  - decidere se gli utenti comuni possono eseguirne l'innesto;
  - decidere se permettere l'avvio dei file eseguibili, infatti potrebbe trattarsi di cavalli di Troia;
  - decidere se considerare validi o meno i permessi SUID e SGID che potrebbero dare ai programmi privilegi indesiderabili.
- Il file system estraneo potrebbe contenere dati che non devono essere modificati; in tal caso conviene utilizzarlo in sola lettura per impedire l'alterazione del suo contenuto, anche solo accidentalmente.
- Il file system estraneo potrebbe essere incompatibile con la struttura di un file system Unix. In tal caso, occorre trovare il modo di farlo assomigliare a questo, per esempio attribuendo a tutti i file gli stessi permessi e la proprietà a un utente e a un gruppo particolare.

Un'altra cosa da considerare sono i permessi della directory radice del disco che si va a innestare. Di per sé non c'è nulla di strano, se il file system che si innesta è in grado di gestire tali informazioni, basta usare i comandi normali, come '**chmod**' e '**chown**' per cambiarli, ma questo può confondere il principiante. In breve: quando si

cambia la proprietà e i permessi di una directory sulla quale è stato innestato un altro file system, questi cambiamenti hanno effetto in quel file system innestato.

#### 19.4.1.3 Opzioni

In occasione dell'innesto di un file system si possono definire alcune opzioni allo scopo di modificarne il comportamento predefinito. Quello che segue è un elenco parziale delle opzioni disponibili. Inizialmente vengono mostrate le opzioni che riguardano generalmente i file system compatibili con i sistemi operativi Unix e possono essere utilizzate anche in presenza di file system differenti quando ciò può avere significato.

Vedere *mount(8)* e *nfs(5)*.

Tabella 19.62. Opzioni valide per l'innesto di file system Unix.

| Opzione | Descrizione |
|---|---|
| `remount` | Si tratta di un'opzione speciale che può essere usata solo quando il file system in questione è già innestato, allo scopo di reinnestarlo con delle opzioni differenti (quelle che vengono definite assieme a '**`remount`**'). |
| `default` | Utilizza le impostazioni predefinite: '**`rw`**', '**`suid`**', '**`dev`**', '**`exec`**', '**`auto`**', '**`atime`**', '**`nouser`**', '**`async`**'. |
| `sync`<br><br>`async` | Esegue le operazioni di lettura e scrittura sul file system in modo sincrono o asincrono. La modalità sincrona è più sicura, ma il suo utilizzo rallenta e appesantisce l'attività del disco. |

| Opzione | Descrizione |
|---|---|
| `atime`<br>`noatime` | Aggiorna o meno la data di accesso ai file. Può essere utile eliminare questo tipo di aggiornamento per ridurre l'attività del disco. |
| `auto`<br>`noauto` | Permette o impedisce l'innesto automatico quando si utilizza il file '`/etc/fstab`'. |
| `dev`<br>`nodev` | Considera validi, o esclude la validità dei file di dispositivo che dovessero essere contenuti nel file system. |
| `exec`<br>`noexec` | Permette o impedisce l'esecuzione dei file con il permesso di esecuzione. |
| `suid`<br>`nosuid` | Consente o impedisce che i bit SUID (*Set user ID*) e SGID (*Set group ID*) abbiano effetto. Disattivando questa possibilità (cioè utilizzando l'opzione '**`nosuid`**'), si vuole evitare che gli eseguibili contenuti nel file system che si intende innestare, possano ottenere privilegi particolari. |
| `user`<br>`nouser` | Permette o impedisce all'utente comune di innestare e staccare il file system. L'opzione '**`user`**' implica l'attivazione automatica di '**`noexec`**', '**`nosuid`**' e '**`nodev`**', a meno che queste siano annullate da successive indicazioni contrarie come nella lista seguente: '**`user,exec,suid,dev`**'. |
| `ro` | Richiede un accesso in sola lettura. |

| Opzione | Descrizione |
|---|---|
| `rw` | Richiede un accesso in lettura e scrittura. |

La tabella successiva riguarda i file system FAT, innestati come tali (i tipi '**`msdos`**' e '**`vfat`**' rientrano in questa categoria).

Tabella 19.63. Opzioni valide per l'innesto di file system FAT.

| Opzione | Descrizione |
|---|---|
| `uid=`***identificativo_utente*** | Permette di stabilire il proprietario dei file e delle directory contenute nel file system. Se non viene specificato, si intende appartengano all'utente che esegue l'innesto. |
| `gid=`***identificativo_gruppo*** | Permette di stabilire il gruppo proprietario dei file e delle directory contenute nel file system. Se non viene specificato, si intende appartengano al gruppo dell'utente che esegue l'innesto. |
| `umask=`***maschera*** | Permette di stabilire quali permessi inibire nel file system. Si tratta del solito numero ottale, composto da tre cifre numeriche, dove la prima cifra rappresenta i permessi per il proprietario, la seconda per il gruppo, la terza per il resto degli utenti.<br>Se non viene specificato si utilizza il valore predefinito per la creazione dei file nei file system normali: '**`umask`**' appunto. |

| Opzione | Descrizione |
|---|---|
| `quiet` | I file system FAT non sono in grado di memorizzare informazioni sulle proprietà e i permessi dei file. Di conseguenza, i programmi che tentano di modificare i valori predefiniti, ottengono una segnalazione di errore dalle funzioni di sistema. L'opzione '**`quiet`**' inibisce queste segnalazioni di errore. |

Per quanto riguarda l'opzione '**`umask`**' dei file system FAT, si ricorda il significato dei valori che possono essere attribuiti:

- $1_8$ rappresenta un permesso di esecuzione;
- $2_8$ rappresenta un permesso di scrittura;
- $4_8$ rappresenta un permesso di lettura.

Di conseguenza:

- $3_8$ rappresenta un permesso di scrittura e di esecuzione;
- $5_8$ rappresenta un permesso di lettura e di esecuzione;
- $6_8$ rappresenta un permesso di lettura e di scrittura;
- $7_8$ rappresenta un permesso di lettura, scrittura e di esecuzione.

Bisogna fare attenzione però che il valore che si inserisce rappresenta un impedimento all'uso di quel permesso, di conseguenza, la maschera $022_8$ indica che è consentito al proprietario qualunque tipo di accesso (lettura, scrittura ed esecuzione), mentre agli altri utenti non è consentito l'accesso in scrittura.

Un caso particolare di opzione è '**loop**' che consente di accedere a file-immagine di dischi o partizioni. Questa particolarità viene descritta in un altro capitolo.

#### 19.4.1.4 Utilizzo di «mount»

Il programma '**mount**' [22] permettere di innestare, un file system all'interno del sistema. Il programma opposto è '**umount**' e serve per staccare, un file system innestato precedentemente.

```
mount [opzioni] [dispositivo] [directory]
```

La forma normale e più semplice di utilizzo di '**mount**' è la seguente:

```
mount -t tipo_di_file_system dispositivo punto_di_innesto
```

In questo modo si richiede al kernel di innestare il file system del dispositivo specificato nella directory indicata (il punto di innesto).

Per conoscere la situazione dei dispositivi collegati attraverso questo sistema, si può usare la sintassi seguente:

```
mount [ -t tipo_di_file_system ]
```

Se viene specificato il tipo di file system, si ottiene un elenco limitato a quei dispositivi.

Il file system '`/proc/`' non è associato ad alcun dispositivo speciale; pertanto, quando se ne vuole eseguire l'innesto, si può utilizzare un nome di dispositivo arbitrario, per esempio '**proc**'. Lo stesso vale per il file system '`/sys/`', con la differenza che questo tipo ha il nome '**sysfs**'.

La maggior parte delle unità di memorizzazione sono indicate nel modo consueto utilizzando nomi di file di dispositivo (‘`/dev/…`’), ma ci possono essere altre possibilità, come quando si vuole innestare un file system di rete o NFS, dove si usa la forma ‘***nodo*`:`*`/directory`**’.

Il file ‘`/etc/fstab`’ viene utilizzato per automatizzare l’innesto dei file system più importanti al momento dell’avvio del sistema. Questo viene letto utilizzando la forma seguente:

```
mount -a [ -t tipo_di_file_system ]
```

Di solito si trova una chiamata di questo tipo all’interno di uno degli script che compongono la procedura di inizializzazione del sistema (‘`/etc/init.d/*`’ oppure ‘`/etc/rc.d/rc*`’). La presenza del file di configurazione ‘`/etc/fstab`’ è utile anche per semplificare l’innesto (e così l’operazione inversa) di un file system che sia stato previsto al suo interno. Diventa sufficiente una delle due forme seguenti:

```
mount dispositivo
```

```
mount punto_di_innesto
```

In linea di principio, solo l’utente ‘**`root`**’ può innestare un file system. Per permettere agli utenti comuni di innestare e staccare un’unità di memorizzazione (come nel caso di un disco ottico o di una memoria solida esterna), la si può indicare nel file ‘`/etc/fstab`’ con l’opzione ‘**`user`**’. Nell’esempio seguente, si vede un record (una

riga) di '`/etc/fstab`' attraverso il quale si definisce l'innesto facoltativo di un disco ottico, avente un file system ISO 9660, in sola lettura, con la possibilità anche per gli utenti comuni di eseguire l'operazione:

```
/dev/sr0  /disc  iso9660  ro,user,noauto,unhide
```

In tal modo, qualunque utente ha la possibilità di eseguire uno dei due possibili comandi seguenti:

```
$ mount /dev/sr0 [Invio]

$ mount /disc [Invio]
```

La coppia di programmi '**`mount`**' e '**`umount`**' mantiene una lista dei file system innestati correntemente. Quando '**`mount`**' viene avviato senza argomenti si ottiene l'emissione del contenuto di questa lista.

Per approfondire l'utilizzo di '**`mount`**', si veda anche la pagina di manuale *mount(8)*.

Tabella 19.65. Alcune opzioni.

| Opzione | Descrizione |
|---|---|
| `-a` | Utilizza '`/etc/fstab`' per eseguire automaticamente l'operazione: vengono innestati tutti i file system a esclusione di quelli segnati come '**`noauto`**'. |

| Opzione | Descrizione |
|---|---|
| `-t` [`no`] ***tipo_di_file_system*** [`,` …] | Specifica il tipo di file system. Sono riconosciuti i nomi indicati nella tabella 19.61 e anche nella pagina di manuale *mount(8)*, purché il kernel sia compilato includendone le funzionalità. Se il nome del tipo di file system viene preceduto dalla sigla '**no**', si intende che quel tipo deve essere escluso. Se si vogliono indicare più tipi di file system questi vengono separati da virgole.<br>Quando si usa questa opzione con l'indicazione di più tipi, o con il prefisso '**no**', lo si fa quasi sempre con l'uso dell'opzione '**-a**', come nell'esempio seguente:<br>'**mount -a -t nomsdos,nonfs**'<br>In questo caso si intende eseguire l'innesto di tutti i file system indicati all'interno di '`/etc/fstab`', a esclusione dei tipi '**msdos**' e '**nfs**'. |
| `-o` ***opzione_di_file_system*** [`,` …] | Questa opzione permette di specificare uno o più nomi di opzioni, separati da virgole, legati alla gestione del file system. Un elenco parziale di questi nomi si trova nella sezione 19.4.1.3. |

Segue la descrizione di alcuni esempi.

- `# mount -t ext4 /dev/sda2 /mnt` [*Invio*]

  Innesta il file system di tipo Ext4 contenuto nella seconda partizione della prima unità, a partire dalla directory '`/mnt`'.

- `# mount -t vfat /dev/sdb1 /usb` [*Invio*]

  Innesta il file system di tipo Dos-VFAT (Dos-FAT con le estensioni per i nomi lunghi) contenuto nella prima partizione della seconda unità, a partire dalla directory '`/usb/`'.

- `# mount -t nfs roggen.brot.dg:/pubblica /roggen` [*Invio*]

  Innesta il file system di rete offerto dall'elaboratore *roggen.brot.dg*, corrispondente alla sua directory '`/pubblica/`' (e discendenti), nella directory locale '`/roggen/`'.

### 19.4.1.5 Utilizzo di «umount»

Il programma '**umount**'[23] esegue l'operazione inversa di '**mount**': separa i file system. L'operazione può avvenire solo quando non ci sono più attività in corso su quei file system, altrimenti l'operazione fallisce.

```
umount [opzioni] [dispositivo] [directory]
```

L'opzione '**-a**' ha un significato simile a quella di '**mount**', ma serve a eseguire il distacco di tutti i file system che in un certo momento risultano innestati. Per sapere quali sono gli innesti in corso, si avvale del contenuto del file '`/etc/mtab`', il quale dovrebbe risultare aggiornato in tal senso.

Tabella 19.66. Alcune opzioni.

| Opzione | Descrizione |
|---|---|
| `-a` | Vengono separati tutti i file system indicati in '`/etc/mtab`'. |
| `-t` [`no`] ***tipo_di_file_system*** [`,`…] | Indica che l'azione deve essere eseguita solo sui file system specificati. Se si usa il prefisso '**no**', l'azione si deve compiere su tutti i file system a esclusione di quelli indicati. |

Segue la descrizione di alcuni esempi.

- `#` **umount /dev/sda2** [*Invio*]

  Separa il file system innestato precedentemente, riferito al dispositivo '`/dev/sda2`'.

- `#` **umount /mnt** [*Invio*]

  Separa il file system innestato precedentemente nella directory '`/mnt`'.

- `#` **umount -a** [*Invio*]

  Separa tutti i file system che trova annotati nel file '`/etc/mtab`', escluso il file system '**proc**' e '**sysfs**'.

Il programma '**umount**' non separa i file system che sono utilizzati in qualche modo, di conseguenza è improbabile che il comando '**umount -a**' possa eseguire il distacco del file system principale. Nella fase di arresto del sistema, questo viene reinnestato in sola lettura prima dell'arresto totale.

### 19.4.1.6 File «/etc/fstab»

«

Il file '`/etc/fstab`' viene utilizzato per definire le caratteristiche e le directory di inserimento (punti di innesto) dei vari file system, usati di frequente nel sistema. Si tratta di un file che viene solo letto dai programmi e il suo aggiornamento viene fatto in modo manuale dall'amministratore del sistema.

Il file è organizzato in record (corrispondenti alle righe) divisi in campi separati da uno o più spazi (inclusi i caratteri di tabulazione). Le righe che iniziano con il simbolo '**#**', le righe vuote e quelle bianche sono ignorate e trattate eventualmente come commenti.

1. Il primo campo definisce il tipo di dispositivo o il file system remoto da innestare.
2. Il secondo campo definisce la directory che funge da punto di innesto per il file system.
3. Il terzo campo definisce il tipo di file system e ne viene indicato il nome in base alla tabella 19.61.

   Se in questo campo viene indicato il termine '**ignore**', si intende fare riferimento a una partizione presente, ma inutilizzata, per la quale non si vuole effettuare alcun innesto. Di fatto, i record che contengono questa indicazione vengono ignorati.
4. Il quarto campo descrive le opzioni speciali per il tipo di innesto che si intende eseguire. Si tratta delle stesse opzioni speciali descritte nella pagina di manuale *mount(8)* e anche nella sezione 19.4.1.4 in occasione della spiegazione dell'uso dell'opzione '**-o**' (a esclusione dell'opzione '**remount**').

5. Il quinto campo viene utilizzato per determinare quali file system possono essere utilizzati per lo scarico dei dati (*dump*).[24]

6. Il sesto campo viene utilizzato dal programma '**fsck**' per determinare l'ordine in cui il controllo dell'integrità dei file system deve essere effettuato nel momento dell'avvio del sistema.

   Il file system principale dovrebbe avere il numero uno in questo campo, mentre gli altri, il numero due (o anche valori superiori). Se questo campo contiene il valore zero, significa che il file system in questione non deve essere controllato.

Nell'esempio seguente, tutte le unità che non sono unite stabilmente al corpo fisico dell'elaboratore, hanno l'opzione '**noauto**' che impedisce l'innesto automatico all'avvio del sistema. Queste possono essere attivate solo manualmente, attraverso '**mount**', con il vantaggio di potere indicare semplicemente la directory di inserimento (il punto di innesto) o il nome del file di dispositivo.

```
# nome              Innesto      Tipo    Opzioni           Dump Check

/dev/sda3           /            ext4    defaults             0     1
/dev/sdb1           /home        ext4    defaults             0     2
proc                /proc        proc    defaults             0     0
sys                 /sys         sysfs   defaults             0     0
/dev/sda2           none         swap    sw
/dev/sda1           /mnt/dosc    vfat    quiet,umask=000      0     0
/dev/sdc            /mnt/dosd    vfat    user,noauto,quiet    0     0
/dev/sdc1           /mnt/scsimo  ext4    user,noauto          0     0
/dev/sr0            /mnt/disc    iso9660 ro,user,noauto       0     0
roggen.brot.dg:/    /mnt/roggen  nfs     ro,user,noauto       0     0
/dev/fd0            /mnt/dosa    vfat    user,noauto,quiet    0     0
```

- La partizione corrispondente al file di dispositivo '`/dev/sda3`' viene utilizzata come file system principale; pertanto è il primo a

essere attivato. L'ultimo campo (*check*) riporta il valore uno, perché si vuole fare in modo che questo file system venga controllato per primo al momento dell'avvio del sistema.

- La partizione corrispondente al file di dispositivo '`/dev/sdb1`' viene utilizzata come file system per contenere le directory personali degli utenti. L'ultimo campo riporta il valore due, perché si vuole fare in modo che questo file system venga controllato per secondo al momento dell'avvio del sistema, dopo il controllo del file system principale.
- Il file system virtuale '**`proc`**' viene innestato correttamente nella directory '`/proc/`'. Nel campo del nome, la sigla '**`proc`**' non significa nulla, ma è preferibile al consueto '**`none`**' che si usa spesso in questo caso. Lo stesso vale per il file system virtuale '**`sysfs`**' che viene innestato nella directory '`/sys/`'.
- La partizione corrispondente al file di dispositivo '`/dev/sda1`' contiene un file system Dos-FAT con la gestione dei nomi lunghi. In particolare, viene permesso a ogni utente di accedere ai suoi file in tutti i modi possibili.
- Il file di dispositivo '`/dev/sdc`' rappresenta un cosiddetto *superfloppy*, cioè un'unità rimovibile che non viene suddivisa in partizioni. Trattandosi di un disco rimovibile viene concesso a tutti gli utenti di eseguire l'innesto e questo non viene effettuato automaticamente al momento dell'avvio del sistema.
- La partizione corrispondente al file di dispositivo '`/dev/sdc1`' rappresenta la stessa unità rimovibile, ma in questo caso viene considerato l'accesso alla prima partizione di tale unità. Anche

qui viene concesso agli utenti comuni di innestare e staccare la partizione.

- Il file ‘`/dev/sr0`’ rappresenta il lettore di un disco ottico (CD, DVD o BR) in cui si prevede di usare solo dischi con file system di tipo ISO 9660. In particolare, viene specificato che l’accesso può avvenire in sola lettura.
- L’elaboratore *roggen.brot.dg* condivide tutto il proprio file system (a partire dalla directory radice) attraverso il protocollo NFS. Viene consentito l’accesso in sola lettura.
- L’unità a dischetti, corrispondente al file di dispositivo ‘`/dev/fd0`’, può essere utilizzata da tutti gli utenti e si prevede di accedere sempre solo al formato Dos-FAT con l’estensione per i nomi lunghi.

#### 19.4.1.7 File «/etc/mtab»

Il file ‘`/etc/mtab`’ ha la stessa struttura di ‘`/etc/fstab`’, ma viene gestito automaticamente da ‘**`mount`**’ e ‘**`umount`**’, con lo scopo di rappresentare i file system innestati nella struttura generale. Non deve essere modificato e dovrebbe essere creato automaticamente all’avvio del sistema.

In un sistema GNU/Linux, il file virtuale ‘`/proc/mounts`’ riproduce sostanzialmente le stesse informazioni che possono apparire all’interno di ‘`/etc/mtab`’. Per questa ragione, quando la directory ‘`/etc/`’ deve funzionare in sola lettura, si utilizza spesso, al posto del file ‘`/etc/mtab`’, un collegamento simbolico a ‘`/proc/mounts`’. Tuttavia, in condizioni normali questo non si fa, perché

non tutti i programmi che hanno bisogno di accedere a '`/etc/mtab`' si trovano a proprio agio con tale raggiro.

#### 19.4.1.8 Utilizzo di «df»

«

Il programma '**`df`**',[25] ovvero *Disk free*, permette di conoscere lo spazio a disposizione di una o di tutte le partizioni che risultano innestate. Se non vengono indicati i nomi dei file di dispositivo, si ottiene l'elenco completo di tutti quelli che sono attivi, altrimenti l'elenco si riduce ai nomi specificati.

`df` [ ***opzioni*** ] [ ***dispositivo*** ... ]

L'unità di misura che esprime questo spazio è in blocchi la cui dimensione cambia a seconda delle opzioni utilizzate oppure dalla presenza di una variabile di ambiente: ***POSIXLY_CORRECT***. La presenza di questa fa sì che, se non viene usata l'opzione '**`-k`**', i blocchi siano di 512 byte come prevede lo standard POSIX. Diversamente, il valore predefinito dei blocchi è di 1024 byte.

Tabella 19.68. Alcune opzioni.

| Opzione | Descrizione |
|---|---|
| `-a`<br>`--all` | Emette le informazioni relative a tutti i dispositivi attivi, anche di quelli che normalmente vengono ignorati. |
| `-h`<br>`--human-readable` | Aggiunge una lettera alla dimensione, in modo da chiarire il tipo di unità di misura utilizzato. |

| Opzione | Descrizione |
|---|---|
| `-i`<br>`--inodes` | Emette il risultato indicando l'utilizzo e la disponibilità di inode, invece che fare riferimento ai blocchi. Questa informazione è utile solo per i file system che utilizzano una struttura a inode. |
| `-b`<br>`--byte` | Emette le dimensioni in byte e non in kibibyte (simbolo: «Kibyte»). |
| `-k`<br>`--kilobytes` | Emette le dimensioni in kibibyte. Questa opzione fa riferimento all'unità di misura predefinita, ma permette di fare ignorare a '**`df`**' la presenza eventuale della variabile ***POSIXLY_CORRECT***. |
| `-m`<br>`--megabytes` | Emette le dimensioni in mebibyte (simbolo: «Mibyte»). |

### 19.4.1.9 Utilizzo di «di»

Il programma '**`di`**', [26] ovvero *Disk information*, permette di ottenere informazioni simili a quelle di '**`df`**', con la possibilità di specificare con più libertà il formato con cui queste devono essere visualizzate:

```
di [opzioni] [file...]
```

Per individuare i dischi e le partizioni, è sufficiente indicare dei file che si trovano al loro interno, altrimenti si ottiene lo stato di tutti i dischi e le partizioni che costituiscono l'albero attuale del file system.

Tabella 19.69. Alcune opzioni.

| Opzione | Significato mnemonico | Descrizione |
|---|---|---|
| `-a` | *all* | Emette le informazioni relative a tutti i dispositivi attivi, compresi quelli che non contengono alcunché, come '`/proc/`'. |
| `-d`***x*** | *display-size* | Mostra i valori in multipli di quanto specificato, attraverso la lettera ***x***: '**`p`**' sta per POSIX (512 byte), '**`k`**' sta per kibibyte, '**`m`**' sta per mebibyte, '**`g`**' sta per gibibyte, ecc. |
| `-dg`<br>`-g` | *gibibyte* | Mostra i valori in gibibyte. |
| `-dm`<br>`-m` | *mebibyte* | Mostra i valori in mebibyte. |
| `-km`<br>`-k` | *kibibyte* | Mostra i valori in kibibyte. |

Tra le opzioni disponibili è possibile anche usare '**`-f`**' che consente di specificare un formato libero di visualizzazione delle informazioni. Questa e altre opzioni non vengono mostrate; per approfondire l'uso di questo programma si può consultare la pagina di manuale *di(1)*. Vengono mostrati alcuni esempi.

- `$` **`di`** [*Invio*]

```
Filesystem            Mount        Megs    Used   Avail %used fs Type
/dev/sda1             /          8130,1  4728,8  2988,3  63%  auto
tmpfs                 /dev/shm    219,4     0,0   219,4   0%  tmpfs
172.21.254.254:/home  /home    113337,9 47517,2 60063,4  47%  nfs
```

Mostra le informazioni sulle diverse porzioni che compongono l'insieme del file system complessivo, ignorando eventualmente ciò che non contiene alcunché.

- `$` **`di -a`** [*Invio*]

```
Filesystem            Mount             Megs    Used   Avail %used fs Type
/dev/sda1             /               8130,1  4728,8  2988,3  63%  auto
tmpfs                 /dev/shm         219,4     0,0   219,4   0%  tmpfs
172.21.254.254:/home  /home          113337,9 47517,2 60063,4  47%  nfs
proc                  /proc                0,0     0,0     0,0   0%  proc
usbfs                 /proc/bus/usb        0,0     0,0     0,0   0%  usbfs
sysfs                 /sys                 0,0     0,0     0,0   0%  sysfs
```

Mostra le informazioni sulle diverse porzioni che compongono l'insieme del file system complessivo, includendo anche ciò che non contiene alcunché (come il file system '`/proc/`').

- `$` **`di -dg`** [*Invio*]

```
Filesystem            Mount      Gigs Used Avail %used fs Type
/dev/sda1             /           7,9  4,6   2,9  63%  auto
tmpfs                 /dev/shm    0,2  0,0   0,2   0%  tmpfs
172.21.254.254:/home  /home     110,7 46,4  58,6  47%  nfs
```

Mostra le informazioni numeriche in gibibyte.

- `$` **`di -dg mio_file`** [*Invio*]

```
Filesystem            Mount    Gigs  Used  Avail %used fs Type
172.21.254.254:/home  /home   110,7  46,4   58,6  47%  nfs
```

Mostra le informazioni riferite al file system contenente il file ‘`mio_file`’, il quale in questo caso risulta essere collocato presso un elaboratore remoto.

### 19.4.1.10 Utilizzo di «du»[27]

«

Il programma di servizio ‘**du**’[28] (*Disk usage*) emette una sorta di statistica dell’utilizzo dello spazio da parte di un elenco di file o directory (in base al loro contenuto):

```
du [opzioni] [file...]
```

L’unità di misura che esprime questo spazio è in blocchi, la cui dimensione cambia a seconda delle opzioni utilizzate oppure dalla presenza di una variabile di ambiente: ***POSIXLY_CORRECT***. Se esiste e non viene usata l’opzione ‘**-k**’, fa sì che i blocchi siano di 512 byte come prevede per questo lo standard POSIX. Diversamente, il valore predefinito dei blocchi è di 1024 byte.

Tabella 19.74. Alcune opzioni.

| Opzione | Descrizione |
|---|---|
| `-a`<br>`--all` | Emette il conteggio riferito a tutti i file, non solo alle directory. |
| `-b`<br>`--byte` | Emette le dimensioni in byte. |

| Opzione | Descrizione |
|---|---|
| `-k`<br>`--kilobytes` | Emette le dimensioni in kibibyte (simbolo: «Kibyte»). Questa opzione fa riferimento all'unità di misura predefinita, ma permette di fare ignorare a '**du**' la presenza eventuale della variabile ***POSIXLY_CORRECT***. |
| `-m`<br>`--megabytes` | Emette le dimensioni in mebibyte (simbolo: «Mibyte»). |
| `-h`<br>`--human-readable` | Aggiunge una lettera alla dimensione, in modo da chiarire il tipo di unità di misura utilizzato. |
| `-c`<br>`--total` | Emette anche un totale generale finale. |
| `-s`<br>`--summarize` | Emette solo un totale per ogni argomento. |
| `-S`<br>`--separate-dirs` | Emette la dimensione delle directory in modo separato, senza includere lo spazio utilizzato dalle sottodirectory. |
| `-x`<br>`--one-file-system` | Salta il conteggio delle directory che si trovano in un file system diverso da quello di partenza. |

### 19.4.1.11 Utilizzo di «stat»

«

Il programma '**stat**' [29] permette di conoscere alcune informazioni generali su di un file, su di una directory, o sul file system in cui questi si trovano:

```
stat [opzioni] file...
```

Le informazioni che si ottengono possono essere organizzate anche attraverso un modello libero, usando le opzioni appropriate, in modo da facilitare l'acquisizione di queste informazioni da parte di un altro programma.

Tabella 19.75. Alcune opzioni.

| Opzione | Descrizione |
|---|---|
| `-f`<br>`--filesystem` | Invece di mostrare le informazioni relative al file o alla directory indicati, si ottengono quelle del file system che li contiene. |
| `-L`<br>`--dereference` | Nel caso venga indicato un collegamento simbolico con questa opzione si ottengono le informazioni al file, alla directory, o al file system a cui punta. |
| `-t`<br>`--terse` | Mostra le informazioni in modo essenziale, per facilitarne la cattura attraverso uno script. |

Tra le opzioni che appaiono nella tabella precedente, viene omessa '**-c**', che consentirebbe di configurare il modo in cui si vogliono ottenere le informazioni. Per questo si deve consultare il documento *info stat*, oppure la pagina di manuale *stat(1)*.

Vengono mostrati alcuni esempi senza commentarli.

```
$ stat mio_file [Invio]

  File: 'mio_file'
  Size: 33              Blocks: 8          IO Block: 8192   regular file
Device: ah/10d  Inode: 4112856     Links: 1
Access: (0644/-rw-r--r--)  Uid: ( 1001/ daniele)   Gid: ( 1001/ daniele)
Access: 2004-09-17 09:40:22.000000000 +0200
Modify: 2004-09-17 09:40:50.000000000 +0200
Change: 2004-09-17 09:40:50.000000000 +0200

$ stat -t mio_file [Invio]

mio_file 33 8 81a4 1001 1001 a 4112856 1 0 0 1095406822 ↩
↪1095406850 1095406850 8192

$ stat -f mio_file [Invio]

  File: "mio_file"
    ID: 0        Namelen: 255     Type: nfs
Blocks: Total: 4206189  Free: 3662077  Available: 3448412  Size: 8192
Inodes: Total: 4276224  Free: 4113427

$ stat -f / [Invio]

  File: "/"
    ID: 0        Namelen: 255     Type: ext2/ext3
Blocks: Total: 8412378  Free: 7525313  Available: 7097984  Size: 4096
Inodes: Total: 4276224  Free: 4134421
```

### 19.4.2 Memoria cache

La memoria cache dei dischi serve a ridurre l'attività di questi, effettuando le modifiche a intervalli regolari o quando diventa indispensabile per altri motivi. L'esistenza di questo tipo di organizzazione, basato su una «memoria trattenuta», è il motivo principale per cui si deve arrestare l'attività del sistema prima di spegnere l'elaboratore.

La memoria cache viene gestita automaticamente dal kernel, con uno scarico a intervalli regolari; tuttavia, esiste la possibilità di richiedere espressamente l'aggiornamento completo delle unità di memorizzazione innestate.

Il programma '**sync**'[30] permette di scaricare nei dischi i dati contenuti nella memoria cache. Viene usato solitamente dalla procedura di arresto del sistema per garantire che tutti i dati siano registrati correttamente su disco prima dello spegnimento fisico dell'elaboratore.

```
sync [opzioni]
```

Può essere utilizzato in caso di emergenza, quando per qualche ragione non si può attendere il completamento della procedura di arresto del sistema, o per qualunque altro motivo. Di solito non si usano opzioni.

## 19.5 Memoria virtuale

«

La memoria virtuale è la memoria centrale della macchina virtuale che il sistema operativo realizza. In pratica, questa memoria virtuale si avvale della memoria centrale fisica e, possibilmente, di spazio all'interno di unità di memorizzazione di massa.[31] Il meccanismo che consente l'estensione della memoria reale si basa sullo scambio di porzioni di memoria centrale nella memoria di massa. Il termine inglese *swap* deriva da questa continua operazione di scambio.

Utilizzando un sistema GNU/Linux comune, se non si dispone di una quantità di memoria centrale abbastanza grande, è necessario attivare il meccanismo di scambio della memoria virtuale. La tabella

??t swap?? elenca i programmi e i file a cui si accenna in questo capitolo.

### 19.5.1 Creazione di una partizione o di un file di scambio

Con GNU/Linux è possibile attivare la gestione della memoria virtuale utilizzando due tipi di aree nella memoria di massa: una partizione dedicata oppure un file. La partizione di scambio è preferibile al file di scambio; sia per le prestazioni, sia per il fatto che si evitano complicazioni nel momento dell'arresto del sistema.

La scelta della dimensione di una partizione di scambio è importante perché, una volta deciso, questa normalmente non può più essere cambiata facilmente. In generale, per la partizione di scambio è conveniente utilizzare una dimensione pari ad **almeno** la stessa quantità della memoria RAM effettiva, con un minimo di circa 1 Gibyte. In pratica, il fatto di riservare uno spazio più grande del necessario per la memoria virtuale, non può essere dannoso; al massimo si traduce in uno spreco di spazio nella memoria di massa.

Prima di poter attivare la gestione della memoria virtuale è necessario creare lo spazio in cui devono risiedere le aree di scambio relative. Ciò vale anche nel caso in cui per questo si vogliano utilizzare dei file.

Va osservato che la questione della memoria centrale estesa virtualmente, con lo scambio di porzioni di memoria di massa, riguarda esclusivamente le unità a disco magnetico. Lo sviluppo tecnologico che porta a rimpiazzare i dischi magnetici con l'uso di memorie solide è lo stesso che offre la disponibilità di memorie volatili di grandi capacità; rendendo così inutile lo scambio della memoria.

#### 19.5.1.1 Partizione di scambio

La creazione di una partizione di scambio per la memoria, procede nello stesso modo con cui si crea una qualunque altra partizione. In questo caso non c'è la necessità di eseguire un avvio del sistema operativo su tale partizione, di conseguenza si possono usare anche partizioni logiche contenute in partizioni estese senza problemi di alcun tipo.

La creazione della partizione richiede l'utilizzo di '**fdisk**', oppure di un programma equivalente, ricordando però di assegnare alla partizione il tipo corretto di identificatore: $82_{16}$ (Linux-swap).

#### 19.5.1.2 File di scambio

La caratteristica necessaria di un file destinato a fungere da area di scambio è quella di essere continuo; non può quindi essere frammentato. Il modo corretto per creare un file con queste caratteristiche è quello di utilizzare il programma '**dd**' nel modo seguente:

```
dd if=/dev/zero of=file_da_creare bs=4096 count=dimensione
```

In questo caso, la dimensione fa riferimento a blocchi di 4 Kibyte, pari a quanto stabilito con l'opzione '**bs=4096**'. In effetti, la dimensione ottimale di un file del genere è un multiplo di 4 Kibyte perché le pagine di memoria, utilizzate durante lo scambio della stessa, sono di questa dimensione.

Per esempio, volendo creare il file di scambio '`/swap1`' di 1 Gibyte si può procedere come segue:

```
# dd if=/dev/zero of=/swap1 bs=4096 count=262144 [Invio]
```

Se tutto si conclude come desiderato, si ottiene una risposta del tipo seguente:

```
262144+0 records in
262144+0 records out
```

### 19.5.2 Inizializzazione

Un'area di scambio deve essere inizializzata prima di poterla attivare per il suo scopo, con tutta l'attenzione necessaria per evitare di intervenire nella partizione o nel file sbagliato (cosa che ne provocherebbe la cancellazione). Va anche osservato che non si può inizializzare un'area di scambio mentre questa è in uso, perché sarebbe come intervenire nella stessa memoria centrale, cancellandone arbitrariamente il contenuto.

Il programma '**mkswap**'[32] permette di predisporre una partizione o un file per lo scambio della memoria virtuale. In generale è preferibile utilizzare una partizione dedicata che può essere creata con l'aiuto di '**fdisk**', definendola come Linux-swap.

```
mkswap [-c] dispositivo [dimensione_in_blocchi]
```

Il modello sintattico suggerisce l'uso dell'opzione '**-c**' per il controllo dell'area da usare per lo scambio della memoria, prima della sua attivazione; inoltre è possibile specificare espressamente la dimensione, in blocchi da 1024 byte, nel caso questa non fosse determinabile in modo automatico. Segue la descrizione di alcuni esempi.

- # **mkswap -c /dev/sda3 1048576** [*Invio*]

  Viene inizializzata la partizione di scambio '/dev/sda3' specificando una dimensione di 1048576 blocchi, pari a 1 Gibyte.

- # **mkswap -c /swap1 1048576** [*Invio*]

  Inizializza il file di scambio '/swap1' creato precedentemente con una dimensione di 1 Gibyte.

### 19.5.3 Attivazione e disattivazione della memoria virtuale

Per fare in modo che un'area di scambio venga utilizzata per il suo scopo, occorre attivarne la gestione. L'operazione è compiuta dal programma '**swapon**'. Il meccanismo è simile a quello dell'attivazione di un file system che si ottiene con l'innesto delle unità o delle partizioni. Per questo motivo, l'attivazione delle aree di scambio può essere gestita automaticamente attraverso la configurazione del file '/etc/fstab'.

### 19.5.3.1 Utilizzo di «swapon» e «swapoff»

Il programma '**swapon**'[33] attiva l'utilizzo di un dispositivo, o di un file, per la gestione della memoria virtuale. Di solito si tratta di una partizione o un di file di scambio creati e inizializzati appositamente. Normalmente, '**swapon**' viene chiamato da uno degli script della procedura di inizializzazione del sistema e questo allo scopo di attivare le aree di scambio previste all'interno del file '`/etc/fstab`'.

```
swapon [opzioni] [dispositivo|file]
```

Il programma '**swapoff**'[34] è l'opposto di '**swapon**' e si occupa di disattivare la memoria virtuale su un particolare dispositivo o file per lo scambio, oppure su tutti quelli indicati nel file '`/etc/fstab`'. Utilizza la stessa sintassi e le stesse opzioni di '**swapon**'. In effetti si tratta normalmente solo di un collegamento al programma '**swapon**' che si comporta così quando viene avviato con questo nome: '**swapoff**'.

```
swapoff [opzioni] [dispositivo|file]
```

Le partizioni o i file di scambio attivati manualmente (che quindi non sono annotati nel file '`/etc/fstab`') devono essere disattivati manualmente prima della conclusione dell'attività di GNU/Linux.

In generale è importante l'opzione '**-a**', con la quale viene attivata, o disattivata, la memoria virtuale riferita a tutti i dispositivi indicati come file system '**swap**' all'interno del file '`/etc/fstab`'. Di con-

seguenza, se si usa questa opzione non deve essere indicato alcun dispositivo negli argomenti.

Segue la descrizione di alcuni esempi.

- # **swapon /dev/sda3** [*Invio*]

  Utilizza la partizione '`/dev/sda3`' come memoria virtuale.

- # **swapon /swap1** [*Invio*]

  Utilizza il file '`/swap1`' come memoria virtuale.

- # **swapon -a** [*Invio*]

  Avvia la gestione della memoria virtuale con tutte le partizioni e i file indicati per questo scopo nella configurazione di '`/etc/fstab`'.

- # **swapoff /dev/sda3** [*Invio*]

  Termina la gestione della memoria virtuale con la partizione '`/dev/sda3`'.

- # **swapoff /swap1** [*Invio*]

  Termina la gestione della memoria virtuale con il file '`/swap1`'.

- # **swapoff -a** [*Invio*]

  Termina la gestione della memoria virtuale con tutte le partizioni indicate così nel file '`/etc/fstab`'.

### 19.5.3.2 Gestione automatica attraverso il file «/etc/fstab»

Per quanto riguarda la gestione della memoria virtuale, il file '`/etc/fstab`' permette di definire quali partizioni e file debbano essere utilizzati automaticamente per questo scopo. La configurazione di '`/etc/fstab`' per la gestione della memoria virtuale è praticamente obbligatoria, a meno di provvedere ogni volta alla sua attivazione e disattivazione attraverso l'uso diretto di '**swapon**' e '**swapoff**'.

L'esempio seguente mostra due record ipotetici di '`/etc/fstab`' per l'attivazione della partizione '`/dev/sda3`' e del file '`/swap1`':

```
# nome          collegamento  Tipo    Opzioni
/dev/sda3       none          swap    sw
/swap1          none          swap    sw
```

### 19.5.3.3 Procedura di inizializzazione del sistema

Durante l'esecuzione della procedura di inizializzazione del sistema, si distinguono due fasi per l'attivazione delle aree di scambio della memoria: prima dell'attivazione dei file system vengono attivate le partizioni di scambio; dopo, anche i file di scambio.

Nel file '`/etc/fstab`' non si riesce a distinguere quali siano le partizioni e quali i file, per cui è necessario un trucco molto semplice. Nella prima fase viene eseguito '**swapon**' con l'opzione '**-a**': si possono così attivate solo le partizioni di scambio, perché l'unico file system in funzione dovrebbe essere quello principale che inizialmente è in sola lettura. In pratica, '**swapon**' tenta di attivare anche i file, ma senza riuscirci.

Nella seconda fase, quando i file system sono in funzione, viene eseguito nuovamente '**swapon**', ma questa volta le partizioni già attivate non possono essere attivate nuovamente, mentre i file di scambio vengono trovati e attivati.

```
echo "attivazione delle partizioni di swap"
swapon -a
…
echo "attivazione dei file di swap"
swapon -a 2>&1 | grep -v "busy"
```

In presenza di file di scambio, l'arresto del sistema deve avvenire nel modo corretto: prima si devono disattivare i file di scambio, quindi si può eseguire il distacco dei file system (riportando quello principale in sola lettura) e infine si possono disattivare le partizioni di scambio.

In pratica, spesso si disattiva subito tutta la gestione della memoria virtuale, ma questo rende problematica la conclusione delle operazioni su sistemi dotati di poca memoria. Anche sotto questo aspetto, è sempre consigliabile di evitare l'utilizzo di file di scambio.

## 19.6 Controllo delle «quote» nei sistemi GNU/Linux

«

Generalmente, l'utilizzo dello spazio nel file system non è controllato, per cui gli utenti possono utilizzare teoricamente quanto spazio vogliono in modo indiscriminato. Per controllare l'utilizzo dello spazio nel file system si può attivare la gestione delle ***quote***, cioè un sistema di registrazione dello spazio utilizzato in base all'appartenenza dei file a un utente o a un gruppo particolare. La gestione delle quote non si limita a questo: può impedire di fatto la creazione di file che superano lo spazio consentito.

Il controllo avviene a livello di partizione, per cui occorre stabilire per ognuna di queste le quote di spazio utilizzabili. Generalmente, il problema di controllare le quote riguarda un numero ristretto di partizioni, precisamente quelle in cui gli utenti hanno la possibilità di accedere in scrittura.

Il kernel Linux è in grado di gestire le quote di utilizzo delle partizioni, solo per alcuni tipi di file system; in particolare ciò è assicurato con quelli di tipo Second-extended (Ext2, Ext3 e Ext4). Inoltre, va considerato che il kernel Linux **non è in grado di intervenire direttamente nel file system innestato nella radice, a meno di reinnestarlo in una directory successiva**.[35]

Premesso che il kernel Linux deve essere stato compilato abilitando la gestione delle quote (la voce *Quota support* del menù *File systems*, come si può vedere nella sezione 8.3.9), nei sistemi GNU/Linux la gestione delle quote viene controllata attraverso dei programmi di servizio specifici.[36]

### 19.6.1 Quota utente e quota di gruppo

Il controllo della quota può avvenire a livello di singolo utente, di gruppo o di entrambe le cose. In pratica, un file può essere aggiunto se la quota utente riferita all'UID del file lo consente e, nello stesso modo, se la quota di gruppo riferita al GID del file non viene superata.

Il tracciamento e il controllo dei livelli di quota utente e di gruppo possono essere attivati indipendentemente l'uno dall'altro. In queste sezioni viene mostrato come attivare entrambi i tipi di quota, ma in pratica, se si usano i gruppi privati,[37] è più probabile che si sia interessati esclusivamente al controllo per utente.

### 19.6.2 Configurazione con il file «/etc/fstab»

La gestione delle quote delle partizioni deve essere attivata espressamente nel momento dell'innesto del file system relativo. Per questo si preferisce intervenire nella configurazione contenuta nel file '`/etc/fstab`', in modo da facilitare la cosa. Nella colonna delle opzioni si possono aggiungere due parole chiave: '**`usrquota`**' e '**`grpquota`**'. La prima serve per attivare il controllo delle quote riferite agli utenti e la seconda per il controllo riferito ai gruppi. Le due cose sono indipendenti.

L'esempio seguente mostra in che modo attivare entrambi i controlli nella partizione '`/dev/sda3`', contenente le directory personali degli utenti:

```
# nome      Innesto     Tipo     Opzioni                        Dmp Chk

/dev/sda2   /           ext4     defaults                         0   1
/dev/sda3   /home       ext4     defaults,usrquota,grpquota       0   1
proc        /proc       proc     defaults                         0   0
/dev/sda4   none        swap     sw
/dev/cdrom  /mnt/cdrom  iso9660  ro,user,noauto                   0   0
```

Nell'esempio successivo, le directory personali degli utenti sono incluse nell'unica partizione che viene innestata a partire dalla radice ('`/dev/sda2`'), pertanto, per garantire che le quote vengano considerate, occorre un piccolo raggiro, costituito dal reinnesto della stessa partizione in una sottodirectory:

```
# nome      Innesto     Tipo     Opzioni                        Dmp Chk

/dev/sda2   /           ext4     defaults,usrquota,grpquota     0   1
/dev/sda2   /.quota     ext4     defaults,usrquota,grpquota     0   0
proc        /proc       proc     defaults                       0   0
/dev/sda4   none        swap     sw
/dev/cdrom  /mnt/cdrom  iso9660  ro,user,noauto                 0   0
```

Naturalmente, il punto di innesto, costituito in questo caso dalla directory '`/.quota/`', è arbitrario e può collocarsi in qualunque altra posizione.

Per poter utilizzare la gestione delle quote di utilizzo della memoria di massa, è necessario che le opzioni '**`usrquota`**' o '**`grpquota`**' (o entrambe) siano attive; ovvero, non è sufficiente che siano appena state annotate nel file '`/etc/fstab`'. Pertanto, quando si modifica il file '`/etc/fstab`', per renderle operative, occorre reinnestare le unità. Per esempio:

```
# mount -o remount /home [Invio]
```

### 19.6.3 Registrazione delle quote e attivazione del controllo

I livelli di quota dei vari utenti e dei gruppi sono contenuti in due file: '`quota.user`' e '`quota.group`', oppure '`aquota.user`' e '`aquota.group`'. Questi devono essere collocati nella directory principale della partizione da controllare e richiedono solo i permessi di lettura e scrittura per l'utente '**`root`**'. Normalmente, non è necessario creare tali file, perché a questo provvede automaticamente il programma '**`quotacheck`**' quando viene avviato con l'opzione '**`-c`**' (ovvero '**`--create-files`**').

Prima che il sistema di controllo delle quote possa funzionare, occorre effettuare una scansione della partizione interessata, in modo da raccogliere tutte le informazioni necessarie sull'utilizzo dello spazio dal punto di vista degli utenti e dei gruppi. I file utilizzati per contenere queste informazioni cambiano nome a seconda della versione del formato di questi: la versione più recente (quota versione 2) ha i nomi '`aquota.user`' e '`aquota.group`', mentre la versione più vecchia (quota versione 1) ha i nomi '`quota.user`' e '`quota.group`'. La scelta di un formato rispetto all'altro, dipende soprattutto dalla capacità del kernel.

La scansione si esegue con il programma '**`quotacheck`**' e per sicurezza andrebbe ripetuta la sua esecuzione ogni volta che si avvia il sistema, oppure giornalmente, quando il sistema resta in funzione a lungo (per più giorni). La prima volta, o comunque quando si vuole ripartire da zero, si può usare il comando seguente, dove appare in particolare l'opzione '**`-c`**', già menzionata:

```
# quotacheck -a -v -u -g -c [Invio]
```

Tuttavia, se le partizioni per cui si devono calcolare le quote di utilizzo devono rimanere accessibili anche in scrittura, va aggiunta l'opzione '**`-m`**' (ovvero '**`--no-remount`**'). Logicamente, il fatto che durante la scansione di controllo sia ammessa la scrittura, rende inattendibile il risultato della scansione:

```
# quotacheck -a -v -u -g -c -m [Invio]
```

L'opzione '**`-a`**' richiede la scansione di tutte le partizioni che sono state indicate nel file '`/etc/fstab`' come soggette a controllo delle quote. Le opzioni '**`-u`**' e '**`-g`**' richiedono espressamente che la scansione si prenda cura sia dell'utilizzo in base all'utente, sia in base al

gruppo.

**Quando si utilizza ‘`quotacheck`’ per aggiornare le informazioni sull’utilizzo dello spazio nella memoria di massa, non si usa più l’opzione ‘`-c`’.**

Ogni volta che si innesta una partizione che è soggetta a controllo delle quote, è poi necessario attivare il controllo attraverso il programma ‘**`quotaon`**’. Per esempio, ‘**`quotaon /dev/hda3`**’ attiva il controllo sulla partizione indicata. Tuttavia, generalmente si fa questo all’avvio del sistema, per attivare il controllo su tutte le partizioni specificate per questo nel solito file ‘`/etc/fstab`’. In pratica con il comando seguente:

```
# quotaon -a -v -u -g [Invio]
```

Anche in questo caso, le opzioni ‘**`-u`**’ e ‘**`-g`**’ indicano che si vuole espressamente il controllo dell’utilizzo in base all’utente e in base al gruppo, mentre l’opzione ‘**`-a`**’ indica di attivare la funzione per tutte le voci annotate così nel file ‘`/etc/fstab`’.

A questo punto, conviene preoccuparsi di fare in modo che la procedura di inizializzazione del sistema sia in grado ogni volta di avviare la gestione delle quote. Se la propria distribuzione GNU/Linux non fornisce degli script già pronti, si possono aggiungere al file ‘`/etc/rc.local`’ (o simile) le istruzioni necessarie, come nell’esempio seguente:

```
if /sbin/quotacheck -avug
then
    echo "Scansione delle quote eseguita."
fi

if /sbin/quotaon -avug
then
    echo "Attivazione del controllo delle quote eseguita."
fi
```

Nel caso della distribuzione GNU/Linux Debian, è disponibile lo script '`/etc/init.d/quota`', per avviare e fermare la gestione delle quote:

```
/etc/init.d/quota start|stop
```

Questo script si occupa sia di ripetere la scansione periodica con '**`quotacheck`**', sia di usare '**`quotaon`**' e '**`quotaoff`**' nel modo più appropriato.

### 19.6.4 Quote relative a file system condivisi attraverso la rete

«

È importante chiarire che la gestione delle quote di utilizzo della memoria di massa riguarda generalmente le unità locali; in tal caso non è possibile controllare un file system innestato attraverso un protocollo di rete, in quanto il controllo deve avvenire presso l'elaboratore che offre questo servizio.

In mancanza di accorgimenti particolari, l'utente che utilizza un file system di rete, per il quale subisce un controllo della «quota», si può trovare a un certo punto a non potere più salvare i propri dati, così come può scoprire che alcuni file sono stati rimossi misterio-

samente (una volta che il tempo di grazia è stato superato, dopo lo sforamento dalla quota stabilita). Tuttavia esiste la possibilità di attivare un servizio di rete per la comunicazione di tali informazioni, attraverso il demone '**rpc.rquotad**', in modo da avviare l'utente in tempo.

In generale, una distribuzione GNU/Linux ben organizzata, installa correttamente il demone necessario al passaggio di tali informazioni, quando il file system di rete è condiviso con il protocollo NFS; tuttavia, in altre condizioni, tale notifica può venire ugualmente a mancare.

### 19.6.5 Utilizzo di «quotacheck»

Il programma '**quotacheck**' esegue una scansione di una o più partizioni, allo scopo di aggiornare i file di registrazione delle quote: '`quota.user`' e '`quota.group`', oppure '`aquota.user`' e '`aquota.group`'.

`quotacheck` [*opzioni*] [{*partizione* | *punto_di_innesto*}…]

La prima volta che si vuole attivare la gestione delle quote in una certa unità di memorizzazione, va usato con l'opzione '**-c**', per creare il file '`*quota.user`' o '`*quota.group`' (o entrambi); successivamente conviene usare questo programma ogni volta che si innestano delle partizioni soggette al controllo delle quote di utilizzo (ma senza l'opzione '**-c**'). In generale, uno script della procedura di avvio del sistema dovrebbe provvedere all'uso corretto di '**quotacheck**' per le partizioni innestate automaticamente.

Tabella 19.86. Alcune opzioni.

| Opzione | Descrizione |
|---|---|
| `-u`<br>`--user` | Richiede una scansione per le quote di utilizzo riferite agli utenti. Questa è l'azione predefinita. |
| `-g`<br>`--group` | Richiede una scansione per le quote di utilizzo riferite ai gruppi. |
| `-a`<br>`--all` | Scandisce tutte le partizioni indicate nel file '`/etc/fstab`' come soggette a tale controllo. |
| `-R`<br>`--exclude-root` | Questa opzione viene usata in congiunzione con '**`-a`**' e specifica di eseguire il controllo di tutte le partizioni indicate nel file '`/etc/fstab`', a esclusione di quella principale. |
| `-m`<br>`--no-remount` | Esegue la scansione senza preoccuparsi se il file system relativo è innestato anche in scrittura. |
| `-c`<br>`--create-files` | Ricrea i file da zero, senza tenere conto del loro contenuto precedente. |

Segue la descrizione di alcuni esempi.

- `#` **`quotacheck /dev/sdb2`** [*Invio*]

  Esegue la scansione dell'utilizzo degli utenti della partizione '`/dev/sdb2`', che deve essere già stata innestata e deve risultare

dal contenuto del file '`/etc/fstab`'. In tal caso, può funzionare solo se in questo file è stata specificata l'opzione '**`usrquota`**' in fase di innesto del file system.

- `#` **`quotacheck /mnt/disco2`** [*Invio*]

  Esegue la scansione dell'utilizzo degli utenti della partizione che, da quanto si determina dal file '`/etc/fstab`', si colloca a partire dalla directory '`/mnt/disco2`'. Per tutte le altre considerazioni, vale quanto descritto nell'esempio precedente.

- `#` **`quotacheck -avug`** [*Invio*]

  Questo corrisponde all'utilizzo normale del programma, per scandire tutte le partizioni innestate e registrate nel file '`/etc/fstab`' come soggette al controllo delle quote, sia degli utenti, sia dei gruppi.

- `#` **`quotacheck -avugm`** [*Invio*]

  Come nell'esempio precedente, senza preoccuparsi se i file system sono accessibili anche in scrittura durante la scansione.

- `#` **`quotacheck -avugmc`** [*Invio*]

  Come nell'esempio precedente, rifacendo da zero i file.

### 19.6.6 Utilizzo di «quotaon» e «quotaoff»

Il programma '**`quotaon`**' attiva la gestione delle quote da parte del kernel. Non si tratta quindi di un demone, ma di un programma che termina subito di funzionare.

```
quotaon [opzioni] [{partizione|punto_di_innesto}...]
```

Perché si possa attivare questa gestione, è necessario che i file '*quota.user' e '*quota.group' siano presenti nella directory principale delle partizioni per le quali si vuole la gestione delle quote.

Il programma '**quotaoff**' disattiva la gestione delle quote da parte del kernel. Le opzioni e la sintassi sono le stesse di '**quotaon**'.

```
quotaoff [opzioni] [{partizione|punto_di_innesto}...]
```

Tabella 19.87. Alcune opzioni.

| Opzione | Descrizione |
|---|---|
| -u<br>--user | Interviene nella gestione delle quote utente. Questa è l'azione predefinita. |
| -g<br>--group | Interviene nella gestione delle quote dei gruppi. |
| -a | Fa riferimento al contenuto del file '/etc/fstab', relativamente alle unità innestate automaticamente. |
| -F {vfsold\|vfsv0\|rpc\|xfs} | Consente di specificare il formato con cui sono state accumulate le informazioni sulle quote. |

Segue la descrizione di alcuni esempi.

- # **quotaon /dev/sdb2** [*Invio*]

  Attiva la gestione delle quote utente nella partizione '/dev/sdb2', la quale deve essere già stata innestata e deve risultare dal contenuto del file '/etc/fstab'. In tal caso, può funzionare solo se in questo file è stata specificata l'opzione '**usrquota**' in fase di innesto del file system.

- # **quotaon /mnt/disco2** [*Invio*]

  Attiva la gestione delle quote utente nella partizione che, da quanto si determina dal file '/etc/fstab', si colloca a partire dalla directory '/mnt/disco2'. Per tutte le altre considerazioni, vale quanto descritto nell'esempio precedente.

- # **quotaon -avug** [*Invio*]

  Questo corrisponde all'utilizzo normale del programma, per attivare la gestione delle quote in tutte le partizioni innestate e registrate nel file '/etc/fstab' come soggette al controllo delle quote, sia degli utenti, sia dei gruppi.

### 19.6.7 Assegnazione e verifica delle quote

Le quote che si possono assegnare agli utenti e ai gruppi sono composte dell'indicazione di diversi dati. Lo spazio concesso viene espresso attraverso il numero di blocchi (unità di 1024 byte) e viene definito ***limite logico*** (*soft*) perché viene tollerato un leggero sconfinamento per tempi brevi. A fianco del limite logico si può stabilire un limite di sicurezza, o ***limite fisico*** (*hard*), che non può essere superato in alcun caso. Oltre ai limiti sui blocchi di byte, si stabiliscono normalmente dei limiti di utilizzo di inode, in pratica, il numero massimo di file e directory. Dal momento che si ha a che

fare con file system Second-extended (Ext2, Ext3 o Ext4) che normalmente possono avere un inode ogni 4 Kibyte, si può stabilire facilmente un calcolo di corrispondenza tra blocchi di dati e quantità di inode.

Quando viene fissato il limite fisico, soprattutto quando questo è superiore al limite logico, si intende consentire implicitamente lo sconfinamento del limite di utilizzo. In tal caso è necessario stabilire il tempo massimo per cui ciò è concesso. Generalmente, se non viene definito diversamente, si tratta di una settimana.

Le quote vengono assegnate o modificate attraverso i programmi '**setquota**' e '**edquota**'; la verifica dei livelli può essere fatta dall'utente '**root**' con '**repquota**' e ogni utente può controllare ciò che lo riguarda attraverso il comando '**quota**'. Con '**setquota**' si utilizzano gli argomenti della riga di comando per definire i valori delle quote, mentre con '**edquota**' si modificano le quote attraverso un programma per la gestione di file di testo: in pratica viene creato un file temporaneo e il suo contenuto viene quindi interpretato per modificare le quote. L'esempio seguente mostra l'uso di '**setquota**' per assegnare all'utente '**tizio**' una quota di 10 Mibyte, con una tolleranza del 10 % (11 Mibyte il limite fisico):

```
# setquota -u tizio 11264 10240 0 0 -a[Invio]

# quota -u tizio[Invio]

  Disk quotas for user tizio (uid 1005):
   Filesystem blocks quota limit grace files quota limit grace
    /dev/sda2    292 11264 10240          97     0     0
```

La modifica delle quote dei gruppi avviene nello stesso modo, usando l'opzione '**-g**' al posto di '**-u**'. A fianco di questi livelli di spazio

utilizzabili, c'è il problema di fissare il tempo massimo di sconfinamento (tempo di grazia) che può essere deciso solo a livello globale della partizione.

```
# setquota -t 2592000 2592000 -a[Invio]
```

In questo caso, il tempo di sconfinamento è di 30 giorni, pari a 2592000 s. Il valore appare ripetuto per due volte, perché nel primo caso riguarda lo sconfinamento nella quantità di spazio usato, mentre nel secondo si riferisce allo sconfinamento in quantità di inode (numero di file).

L'utente '**root**' può avere un quadro completo della situazione con '**repquota**', il quale genera una tabella delle varie quote. La colonna '**grace**' serve per annotare eventuali sconfinamenti e riporta il tempo consentito rimanente.

```
# repquota -u -a[Invio]

                        Block limits                File limits
User            used    soft   hard grace    used  soft  hard  grace
…
tizio +-      10500   10240  11264 30days    1123  2560  2816
caio  --          1       0      0              1     0     0
```

Nell'esempio appare solo una parte del listato che si ottiene generalmente. Viene mostrato il caso di due utenti: '**caio**' non ha alcuna limitazione di utilizzo e le sue quote sono azzerate per questo; '**tizio**' invece ha superato un po' il valore del limite logico per l'utilizzo di blocchi. Pertanto, nella colonna '**grace**' appare quanto tempo gli resta per provvedere da solo (quando anche questo tempo scade, vi provvede il sistema).

Infine, il singolo utente può verificare la propria situazione con il

programma '**quota**'.

```
tizio@:~$ quota [Invio]

  Disk quotas for user tizio (uid 502):
  Filesystem blocks  quota limit grace files quota limit grace
   /dev/sda2  10500* 10240 11264 6days  1123  2560  2816
```

Anche in questo caso, si può osservare che l'utente ha superato il limite di spazio concesso, pur senza superare il limite massimo di inode disponibili.

### 19.6.8 Utilizzo di «setquota»

Il programma '**setquota**' permette di assegnare i livelli delle quote agli utenti, attraverso quanto viene indicato con gli argomenti della riga di comando. I modelli successivi sono semplificati:

```
setquota -u nome quota_dati limite_dati ↩
↪                  quota_inode limite_inode { -a | dispositivo }
```

```
setquota -g nome quota_dati limite_dati ↩
↪                  quota_inode limite_inode { -a | dispositivo }
```

```
setquota -t tempo_per_sconfinamento_dati ↩
↪             tempo_per_sconfinamento_inode ↩
↪             { -a | dispositivo }
```

Nella parte finale della riga di comando si specifica per quali unità di memorizzazione intervenire, indicando precisamente un file di

dispositivo oppure l'opzione '**-a**' (per fare riferimento a tutti quelli previsti nel file '`/etc/fstab`').

Quando si usa '**setquota**' secondo i primi due modelli proposti, si vanno a specificare i livelli di quota dell'utente o del gruppo indicato, attraverso quattro valori che rappresentano rispettivamente: la quota dati espressa in multipli di 1024 byte, il limite assoluto espresso sempre in multipli di 1024 byte, la quota di quantità di file; il limite assoluto di quantità di file. Va osservato che un valore pari a zero esprime l'assenza di un limite.

Quando appare l'opzione '**-t**', si fissa il tempo di grazia, ovvero il tempo, espresso in secondi, che viene lasciato all'utente per ritornare ai livelli di utilizzo stabilito dalla propria quota, prima che vi provveda automaticamente il sistema operativo. Questo valore è associato all'unità di memorizzazione ed è unico per tutti gli utenti o per tutti i gruppi.

Segue la descrizione di alcuni esempi.

- # **setquota -u tizio 112640 102400 0 0 -a**[*Invio*]

  Assegna all'utente '**tizio**', per tutte le unità previste nel file '`/etc/fstab`' con il controllo delle quote per gli utenti, la disponibilità di 100 Mibyte (102400 Kibyte), con la possibilità di arrivare fino a 110 Mibyte per un breve lasso di tempo.

  In questo caso non vengono presi in considerazione i limiti nella quantità di file che possono essere creati.

- # **setquota -u tizio 112640 102400 28160 25600 -a**[*Invio*]

  Come nell'esempio precedente, specificando che l'utente può gestire fino a 25600 file, con la possibilità di arrivare fino a 29160

file per tempi brevi.

- `# setquota -t 1296000 1296000 -a` [*Invio*]

  Fissa un tempo di grazia di 15 giorni (pari a 1296000 s), sia per lo sconfinamento nella dimensione dei dati, sia per quello che riguarda la quantità di file.

### 19.6.9 Utilizzo di «edquota»

«

Il programma '**edquota**' permette di assegnare e modificare i livelli delle quote agli utenti, attraverso l'uso di un programma per la creazione e modifica dei testi, precisamente si tratta di VI o di quanto specificato nella variabile di ambiente ***EDITOR***.

```
edquota [opzioni] [utente…]
```

La modifica delle quote può avvenire solo dopo che sono stati predisposti i file '`quota.user`' e '`quota.group`', ovvero '`aquota.user`' e '`aquota.group`'.

Tabella 19.91. Alcune opzioni.

| Opzione | Descrizione |
|---|---|
| `-u` | Modifica le quote utente. È l'azione predefinita se non vengono specificate altre opzioni. |
| `-g` | Modifica le quote dei gruppi. |
| `-p` ***utente_prototipo*** | Duplica le quote dell'utente, specificato come argomento dell'opzione, per tutti gli utenti indicati nella parte finale della riga di comando. |

| Opzione | Descrizione |
|---|---|
| `-t` | Permette di modificare il tempo massimo di sconfinamento del limite logico, fermo restando il limite fisico che non può essere superato in ogni caso. |

Segue la descrizione di alcuni esempi.

- `#` **`edquota -u tizio`** [*Invio*]

  Modifica i livelli di quota dell'utente '**`tizio`**'.

- `#` **`edquota -g utenti`** [*Invio*]

  Modifica i livelli di quota del gruppo '**`utenti`**'.

- `#` **`edquota -u -p tizio caio sempronio`** [*Invio*]

  Attribuisce agli utenti '**`caio`**' e '**`sempronio`**' gli stessi livelli di quota di '**`tizio`**'.

### 19.6.10 Utilizzo di «repquota»

Il programma '**`repquota`**' emette una tabella riepilogativa dell'utilizzo delle quote delle partizioni specificate.

`repquota` [***opzioni***] [{***partizione*** | ***punto_di_innesto***}…]

Tabella 19.92. Alcune opzioni.

| Opzione | Descrizione |
|---|---|
| `-u` | Elenca la situazione delle quote riferite agli utenti. È l'azione predefinita se non vengono specificate altre opzioni. |

| Opzione | Descrizione |
|---|---|
| `-g` | Elenca la situazione delle quote riferite ai gruppi. |
| `-a` | Elenca le quote di tutte le partizioni per cui ciò è previsto attraverso le indicazioni del file '`/etc/fstab`'. |

Segue la descrizione di alcuni esempi.

- `#` **`repquota -a`** [*Invio*]

  Elenca la situazione delle quote riferite agli utenti (predefinito) per tutte le partizioni in cui ciò è stato attivato, in base alle indicazioni del file '`/etc/fstab`'.

- `#` **`repquota -g -a`** [*Invio*]

  Elenca la situazione delle quote riferite ai gruppi per tutte le partizioni in cui ciò è stato attivato, in base alle indicazioni del file '`/etc/fstab`'.

  `#` **`repquota /dev/sda3`** [*Invio*]

  Elenca la situazione delle quote riferite agli utenti (predefinito) per la partizione '`/dev/sda3`' (in uso).

### 19.6.11 Utilizzo di «quota»

«

Il programma '**`quota`**' permette agli utenti di controllare il proprio livello di quota. Effettua l'analisi su tutte le partizioni annotate per questo nel file '`/etc/fstab`'. Solo all'utente '**`root`**' è concesso di utilizzare questo programma per controllare la quota di un altro utente.

```
quota [opzioni]
```

Il programma '**quota**' restituisce un valore diverso da zero se almeno uno dei valori restituiti rappresenta uno sconfinamento dalla quota.

Tabella 19.93. Alcune opzioni.

| Opzione | Descrizione |
|---|---|
| `-u` [*utente*] | Restituisce le quote riferite all'utente. È l'azione predefinita se non vengono specificate altre opzioni. Solo l'utente '**root**' può utilizzare l'argomento aggiuntivo per controllare i livelli di un utente particolare. |
| `-g` [*gruppo*] | Restituisce le quote riferite al gruppo. L'utente può interrogare le quote riferite a gruppi a cui appartiene. |

Segue la descrizione di alcuni esempi.

- `$` **quota** [*Invio*]

  L'utente visualizza i propri livelli di quota.

- `$` **quota -g lavoro1** [*Invio*]

  L'utente visualizza i propri livelli di quota per il gruppo '**lavoro1**' a cui appartiene.

- `#` **quota -u tizio** [*Invio*]

  L'utente '**root**' visualizza i livelli di quota per l'utente '**tizio**'.

## 19.7 Dischi ottici: file system

«

Per «disco ottico» si vuole intendere genericamente ciò che può essere un CD, DVD o BD (*Blu-ray disc*), usato per contenere dati con l'ausilio di un file system tradizionale.[38] Questo tipo di unità di memorizzazione ha la caratteristica di utilizzare un'unica traccia a spirale che inizia dalla parte centrale e si espande verso l'esterno, consentendo così di realizzare anche supporti di capacità minore di quella standard, usando un diametro minore.

La memorizzazione dei dati in un disco ottico può avvenire solo sequenzialmente, mentre per la lettura si accede direttamente ai dati desiderati. Per questo motivo, l'uso più semplice di un disco ottico prevede una sola fase di scrittura, di tutto il file system con il suo contenuto di file e directory. Tuttavia, esiste anche la possibilità di continuare a scrivere su un disco che non ha esaurito lo spazio a disposizione; in tal caso, con un file system appropriato, si può avere la sensazione di usare un disco ottico, come se fosse un'unità di memorizzazione comune, dove però i file che si cancellano non portano a un recupero dei dati e quelli che si modificano, occupano nuovo spazio.

Le possibilità dei dischi ottici sono molte, ma difficilmente i sistemi operativi recepiscono tutti gli standard e tutte le estensioni; pertanto, per memorizzare dati nei dischi ottici in modo «compatibile», è bene essere prudenti e non cercare di pretendere da questo tipo di supporto ciò per cui non è proprio adatto. Anche per questo motivo, qui si considera la scrittura di una sola traccia, in una sola sessione, in una sola fase; inoltre non viene trattato il caso particolare dei DVD-RAM.

Partendo dal presupposto di voler usare i dischi ottici con una sola fase di scrittura, per una sola traccia e sessione, potrebbe essere utilizzato qualunque tipo di file system. Ma per la massima compatibilità va scelto il formato ISO 9660, purché non ci siano file di dimensione maggiore di 4 Gibyte, oppure il formato UDF (*Universal disc format*), ovvero ISO 13346, dove i file possono avere dimensioni maggiori.

### 19.7.1 Caratteristiche di massima di un CD

Il CD è oggi un supporto di memorizzazione pensato per scopi diversi, ma nato per sostituire i dischi audio analogici di vinile. La sua origine specifica ha poi portato a un'evoluzione piuttosto complessa per consentirne l'impiego in altri settori.

Si può immaginare il contenuto di un CD come qualcosa che è suddiviso in ***sessioni***, le quali a loro volta si suddividono in tracce. Ogni traccia può avere una destinazione differente; per esempio può trattarsi di un brano musicale, di un file system, o un altro tipo di informazione.

In condizioni normali, un CD-ROM si compone di una sola sessione, contenente una sola traccia dati (un file system), mentre un CD musicale si compone di una sola sessione contenente diverse tracce, una per ogni brano musicale.

Ogni sessione contiene un indice generale, noto con la sigla TOC (*Table of contents*), collocato alla fine delle tracce, il quale permette di raggiungere l'inizio delle tracce attraverso un accesso diretto. Si fa

riferimento alla creazione di questo indice con il termine *finalization*, oppure *fixation*.

L'ultima sessione di un CD può essere aperta o chiusa. Una sessione che rimane aperta consente l'aggiunta di una sessione ulteriore, mentre una sessione chiusa non permette di aggiungerne altre.

La registrazione di un CD può avvenire in fasi successive. Teoricamente è possibile definire una sessione e inserirvi una traccia alla volta, mettendo l'indice TOC solo alla fine. Una volta completata la sessione con l'indice, se questa rimane aperta, è possibile aggiungerne altre.

Dopo questa descrizione schematica è necessario comprendere che le sessioni non sono degli scompartimenti stagni, ma rappresentano delle fasi successive nella registrazione di un CD. Teoricamente, le varie sessioni devono risultare concatenate fra loro, in modo tale che la lettura dell'indice TOC finale consenta di conoscere la collocazione di tutte le tracce presenti. In questo senso va considerato anche il fatto che la sequenza di numerazione delle tracce è unica in tutto il CD e non viene azzerata all'inizio di sessioni successive. Pertanto, alla fine, il CD dovrebbe apparire scomposto in tracce e la presenza delle sessioni non dovrebbe modificare l'apparenza del contenuto. Naturalmente, questa è la teoria; in pratica gli apparecchi usati per leggere i CD devono essere in grado di riconoscere le sessioni, altrimenti le informazioni successive alla prima sessione di registrazione diventano inaccessibili. In questo senso è sempre bene evitare di realizzare CD contenenti più di una sessione.

### 19.7.2 Caratteristiche di massima di un DVD

Il DVD nasce come un tipo particolare di CD in grado di memorizzare molti più dati (intorno ai 4 700 000 000 byte, ovvero 4,3 Gibyte), dal quale si sono sviluppate diverse varianti. Nelle situazioni più comuni si distinguono DVD–R (o DVD–RW) dai DVD+R (o DVD+RW). Il primo tipo (DVD–R o DVD–RW) è quello che ha caratteristiche tecniche inferiori, perché per la scrittura richiede la stessa procedura da usare per un CD, ma è compatibile con la maggior parte dei lettori DVD; il secondo tipo (DVD+R o DVD+RW) è più evoluto ed è più facile la procedura di registrazione.

Come si può intendere intuitivamente, quando si fa riferimento a un DVD-ROM, si intende un DVD contenente una traccia dati (che non può essere modificata), così come avviene con i CD-ROM, senza fare distinzioni sulla tecnologia usata nella realizzazione del disco («–» o «+» che sia).

### 19.7.3 Caratteristiche di massima di un BD

I dischi ottici BD, ovvero Blu-ray disc, hanno una capacità di 25 000 000 000 byte, pari a circa 23 Gibyte. Si distinguono dischi che possono essere scritti una volta sola, con la sigla BD-R, da quelli che consentono più riscritture, con la sigla BD-RE (*rewriteable*).

### 19.7.4 Creazione dell'immagine di un disco ottico

«

Creare l'immagine di un disco ottico significa predisporre un file in qualità di traccia dati di una sessione del disco che si vuole masterizzare. In generale si tratta di realizzare un file system ISO 9660, probabilmente con qualche estensione, oppure UDF, ma in tal caso senza estensioni.

#### 19.7.4.1 Estensioni Rock Ridge

«

Il file system ISO 9660 è predisposto per gestire file con nomi aventi al massimo 32 caratteri, ma in pratica, per la massima compatibilità, questo limite viene abbassato ulteriormente, per consentire anche ai sistemi Dos degli anni 1990 di potervi accedere. Dal punto di vista dei sistemi Unix, ciò non costituisce solo un problema nella dimensione dei nomi, ma soprattutto nella mancanza di tutti gli altri attributi che può avere un file: i permessi e le proprietà.

I dischi ottici realizzati per gli ambienti Unix sono fatti generalmente utilizzando le estensioni Rock Ridge che alla fine permettono di memorizzare tutti quei dati mancanti, compresa la possibilità di gestire nomi più lunghi, con o senza punti. Quando questi dischi vengono letti in un sistema operativo che non è in grado di interpretare tali estensioni, si riescono a vedere solo nomi di file corti.

Il problema dei nomi deve essere tenuto in considerazione nella preparazione di dischi ottici. Se questi dischi sono destinati ad ambienti normali, Dos e derivati, occorre organizzare un sistema di nomi corti per essere certi che possano essere letti ovunque nello stesso modo.

Quando si usano delle estensioni per gestire nomi lunghi, nelle directory potrebbe essere aggiunto un file contenente un elenco di ab-

binamenti tra i nomi corti che appaiono quando non si dispone delle estensioni e i nomi lunghi corretti. Di solito si tratta del file ‘`TRANS.TBL`’ e il suo significato è evidente: *translation table*, ovvero tabella di conversione.

Anche il numero di livelli di sottodirectory ha un limite nel file system ISO 9660: sono al massimo otto. Le estensioni Rock Ridge permettono di superare tale limite, ma come al solito si pongono delle difficoltà per i sistemi che non sono in grado di gestire tali estensioni.

#### 19.7.4.2 Estensioni Zisofs

Le estensioni Zisofs si sommano a quelle Rock Ridge e sono costituite da una tecnica di compressione sui file, che consente un accesso trasparente ai dati. Si tratta precisamente di un'estensione specifica dei sistemi GNU/Linux, tuttavia è possibile estrarre il contenuto di un disco ottico del genere disponendo degli strumenti necessari. Si veda la sezione 19.9 per maggiori dettagli.

#### 19.7.4.3 Estensioni El-Torito

Le estensioni El-Torito permettono di realizzare un CD-ROM o un DVD-ROM «avviabile», purché il firmware (BIOS) dell'elaboratore sia in grado poi di sfruttare effettivamente questa possibilità.

Il metodo consiste nel simulare un dischetto, come se fosse stato inserito nella prima unità. Questo «dischetto» deve contenere naturalmente tutto quello che serve per avviare un sistema in grado di leggere il disco ottico. In pratica, nel caso delle distribuzioni GNU/Linux, si include l'immagine del dischetto di avvio necessario ad avviare il sistema, usando il disco ottico in sola lettura. Questo di-

schetto deve avere un formato normale: 1200 Kibyte, 1440 Kibyte o 2880 Kibyte.

#### 19.7.4.4 Estensioni Joliet

Le estensioni Joliet nascono dalla Microsoft e servono a consentire la lettura del disco in un sistema MS-Windows, preservando i nomi lunghi.

Le estensioni Joliet non pongono limiti alla dimensione dei nomi dei file, tuttavia non consentono di distinguerli oltre i primi 64 caratteri e si crea un errore irreversibile se due file hanno i nomi che nei primi 64 caratteri sono uguali. In presenza di tali file, se questi non sono rivolti al pubblico che deve consultare il disco attraverso un sistema MS-Windows, si può informare il programma che produce il file system di renderli invisibili alle estensioni Joliet.

In generale, salvo problemi nella lunghezza dei nomi dei file, conviene utilizzare anche queste estensioni in modo da permetterne la lettura del disco che si realizza in qualunque circostanza.

Va osservato che non sempre un disco ottico con estensioni Joliet, realizzato attraverso gli strumenti che si dispongono normalmente in un sistema GNU, risulta leggibile effettivamente in un sistema MS-Windows. Pertanto la leggibilità di un disco destinato alla diffusione, va sempre verificata con il sistema operativo di destinazione. Eventualmente, si può contare ragionevolmente sul fatto che un disco realizzato senza estensioni Joliet sia leggibile se i nomi dei file sono compatibili con i vecchi limiti dei sistemi Dos (otto caratteri, un punto e tre caratteri di estensione).

### 19.7.4.5 Strumenti per la creazione di un file-immagine ISO 9660 e UDF

Per la creazione di un file-immagine con file system adatto a un disco ottico, si utilizza un programma particolare, in grado di gestire anche le varie estensioni:

```
genisoimage [opzioni] directory
```

Il programma '**genisoimage**',[39] crea il file-immagine a partire dal contenuto di una directory. In pratica, tutto quello che è contenuto nella directory, comprese le eventuali sottodirectory, viene usato per generare il nuovo file system. Non si tratta di un normale programma di inizializzazione, perché con questo tipo di file system non è possibile inizializzare prima e scrivere i dati dopo: inizializzazione e registrazione sono simultanee.

Il programma '**genisoimage**' è derivato da '**mkisofs**', da cui eredita buona parte della sintassi.

Se non viene utilizzata l'opzione '**-o**', il risultato viene emesso attraverso lo standard output.

Questo programma non è in grado di registrare i dischi ottici, ma solo di generare un'immagine del risultato finale in un file che poi viene utilizzato dal programma di masterizzazione. In teoria è possibile inviare l'output del programma direttamente al programma di masterizzazione, ma si tratta generalmente di una tecnica sconsigliabile.

Tabella 19.94. Alcune opzioni generali

| Opzione | Descrizione |
|---|---|
| `-f` | Utilizzando questa opzione, si fa in modo che i collegamenti simbolici vengano tradotti nel file o nella directory a cui puntano. Ciò è utile quando si prepara un disco senza le estensioni Rock Ridge. Se questa opzione non viene utilizzata, i collegamenti simbolici sono copiati come tali e di conseguenza si utilizzano le estensioni Rock Ridge. La lettera «f» sta per *follow links*. |

| Opzione | Descrizione |
|---|---|
| `-m` ***modello*** | Questa opzione, seguita da un modello realizzato con i caratteri jolly gestiti dalla shell, permette di escludere tutti i file e le directory che corrispondono al modello. Il modello riguarda solo il nome dei file e delle directory, non il percorso necessario a raggiungerli. |
| `-o` ***file*** | Questa opzione permette di specificare il file di destinazione che deve contenere l'immagine del file system generato. Il file indicato può anche essere un dispositivo a blocchi di un'unità a dischi, come un dischetto o una partizione, ma non può trattarsi dell'unità corrispondente a un disco ottico. Se questa opzione non viene utilizzata, il risultato viene emesso attraverso lo standard output. La lettera «o» sta per *output*. |
| `-v` | Durante il funzionamento, genera più informazioni. L'opzione può essere data due volte, per ottenere una quantità maggiore di informazioni. La lettera «v» sta per *verbose*. |

| Opzione | Descrizione |
|---|---|
| `-C` ***inizio_ultima_sessione*`,`*`inizio_prossima_sessione*` | Questa opzione si usa assieme a '**`-M`**' per la creazione di un'immagine da accodare in una sessione successiva di un. L'argomento dell'opzione si compone di due parti, separate da una virgola, costituite da numeri ottenibili dal programma '**`wodim`**' con l'opzione '**`-msinfo`**', in riferimento a un disco multisessione a cui si vuole accodare l'immagine che si sta per creare. |
| `-M` ***immagine_nella_sessione_precedente*** | Si usa assieme a '**`-C`**' per creare un'immagine da considerare la continuazione di un'altra, da inserire nella sessione successiva del disco. |

Tabella 19.95. Estensioni legate ai dati contenuti

| Opzione | Descrizione |
|---|---|
| `-R` | Questa opzione permette di generare un file system ISO 9660 con estensioni Rock Ridge, per la memorizzazione di tutte le informazioni tipiche dei sistemi Unix. |

| Opzione | Descrizione |
|---|---|
| `-r` | Questa opzione si comporta in modo analogo a '**`-R`**' con la differenza che la proprietà e i permessi vengono modificati: file e directory appartengono all'utente e al gruppo '**`root`**'; tutti i file e le directory ottengono tutti i permessi di lettura; tutti i permessi di scrittura vengono tolti; tutti i file e le directory che risultano avere almeno un permesso di esecuzione o di attraversamento, lo ottengono per tutti gli utenti; i permessi particolari: SUID, SGID e Sticky, vengono rimossi. |
| `-z` | Serve a precisare che la directory usata per creare il file-immagine contiene file compressi, conformi alle estensioni Zisofs. Questa opzione richiede l'abbinamento con '**`-r`**' o '**`-R`**'. |
| `-T` | Utilizzando questa opzione si attiva la creazione del file '`TRANS.TBL`' in ogni directory, per contenere la tabella di conversione necessaria a stabilire il nome corretto dei file quando si legge il disco in un sistema che non è in grado di riconoscere le estensioni Rock Ridge. |
| `-J` | Genera le estensioni Joliet per consentirne la lettura nei sistemi MS-Windows. |
| `-apple` | Genera le estensioni Apple. |

| Opzione | Descrizione |
|---|---|
| `-udf` | Aggiunge delle estensioni per un file system UDF. Utilizzando questa opzione, si ottiene un file system che può essere innestato come ISO 9660, oppure UDF. Ma utilizzando il file system come UDF, si perdono le informazioni tipiche di un sistema Unix (in particolare si perdono i collegamenti simbolici). Se si innesta un disco di questo tipo in un sistema GNU/Linux, lasciando che sia il kernel a individuarne il tipo, l'innesto avviene in qualità di file system UDF; eventualmente, per leggere il disco secondo il file system ISO 9660, occorre specificarlo espressamente. Va ricordato che il file system ISO 9660 non consente di «vedere» i file che occupano più di 4 Gibyte. |

Tabella 19.96. Estensioni El-Torito, per rendere avviabile il disco ottico.

| Opzione | Descrizione |
|---|---|
| `-b` ***file_immagine_di_avvio*** | Permette di indicare il file contenente l'immagine di un dischetto da utilizzare per rendere avviabile il disco ottico con le estensioni El-Torito. Questa opzione si usa assieme a '**`-c`**'. La lettera «b» sta per *boot*. |

| Opzione | Descrizione |
|---|---|
| `-c` ***file_catalogo*** | Questa opzione si usa assieme a '**`-b`**' allo scopo di rendere avviabile il disco ottico utilizzando le estensioni El-Torito. Si deve indicare il nome di un file. Il programma '**`genisoimage`**' lo deve creare o sovrascrivere all'interno della gerarchia che compone l'insieme di ciò che si vuole inserire nel file system ISO 9660. In pratica, questo file serve a contenere delle indicazioni relativamente all'avvio del disco ottico, definite automaticamente da '**`genisoimage`**'.<br>È un po' difficile indicare correttamente i file abbinati alle opzioni '**`-b`**' e '**`-c`**'. In pratica è necessario che la directory corrente nel momento della creazione dell'immagine ISO 9660 corrisponda al punto iniziale della gerarchia che si vuole archiviare in questo modo. Così, i percorsi dei file in questione possono essere indicati in modo relativo. |

Segue la descrizione di alcuni esempi.

- `#` **`genisoimage -r -T -v -o prova.img /home/tizio/elio`**[*Invio*]

  Crea un'immagine ISO 9660 nel file '`prova.img`' di quanto contenuto a partire dalla directory '`/home/tizio/elio/`'. Vengono usate le estensioni Rock Ridge, attraverso l'opzione '**`-r`**', ma la proprietà e i permessi di file e directory vengono adattati nel modo generalmente più opportuno. Attraverso l'opzione '**`-T`**' si ottiene la creazione del file '`TRANS.TBL`' in ogni directory.

- `# genisoimage -r -T -v -o prova.img ↩`
  `↪ -b images/boot.img -c boot/boot.cat `pwd`` [*Invio*]

  Crea un'immagine ISO 9660 in un file, a partire dalla directory corrente (l'indicazione viene ottenuta attraverso quanto restituito dal comando '**pwd**'). Vengono usate le estensioni Rock Ridge, con l'opzione '**-r**', in modo che la proprietà e i permessi di file e directory siano adattati nel modo generalmente più opportuno. Inoltre si utilizza il file '`images/boot.img`' per l'avvio del disco ottico e si crea il file '`boot/boot.cat`' per lo stesso motivo.

- `# genisoimage -r -T -v -o prova.img ↩`
  `↪ -b images/boot.img -c boot/boot.cat $PWD` [*Invio*]

  Come nell'esempio precedente, con la differenza che la directory corrente viene ottenuta dalla variabile di ambiente ***PWD***.

- `# genisoimage -r -T -J -apple -udf -v ↩`
  `↪ -o prova.img /home/tizio/elio` [*Invio*]

  Crea un'immagine ISO 9660 nel file '`prova.img`' di quanto contenuto a partire dalla directory '`/home/tizio/elio/`'. Vengono usate le estensioni Rock Ridge, attraverso l'opzione '**-r**', ma la proprietà e i permessi di file e directory vengono adattati nel modo generalmente più opportuno. L'opzione '**-J**' aggiunge le estensioni Joliet; l'opzione '**-apple**' aggiunge le estensioni Apple; l'opzione '**-udf**' rende ibrido il file system, consentendo di interpretarlo anche come UDF. Attraverso l'opzione '**-T**' si ottiene la creazione del file '`TRANS.TBL`' in ogni directory.

- `# genisoimage -r -T -J -apple -udf -v -o prova.img ↩`
  `↪ -b images/boot.img -c boot/boot.cat `pwd`` [*Invio*]

Crea un'immagine ISO 9660 in un file, a partire dalla directory corrente (l'indicazione viene ottenuta attraverso quanto restituito dal comando '**pwd**'). Vengono usate le stesse estensioni dell'esempio precedente, specificando che il file 'images/boot.img' deve essere usato per l'avvio del disco ottico, creando il file 'boot/boot.cat' per lo stesso motivo.

- `# genisoimage -r -T -J -apple -udf -v -o prova.img ↩`
  `↪ -b images/boot.img -c boot/boot.cat $PWD` [*Invio*]

  Come nell'esempio precedente, con la differenza che la directory corrente viene ottenuta dalla variabile di ambiente ***PWD***.

- `# genisoimage -z -r -v -o prova.img /home/tizio/elio` [*Invio*]

  Crea un'immagine ISO 9660 nel file 'prova.img' di quanto contenuto a partire dalla directory '/home/tizio/elio/', creata precedentemente con file compressi per ottenere le estensioni Zisofs.

### 19.7.4.6 Esperimenti con il file system ISO 9660

Prima di arrivare alla realizzazione di un disco ottico occorre fare qualche esperimento non distruttivo. Si suppone che si tratti dell'utente '**tizio**' che ha costruito una struttura a partire dalla directory 'prova/' discendente dalla propria directory personale.

`$ genisoimage -r -T -v -o ~/prova.img ~/prova` [*Invio*]

In questo modo è stato creato il file 'prova.img' nella directory personale dell'utente. Per verificare il risultato si può eseguire l'innesto dell'immagine appena creata, ma per questo occorre avere i privilegi dell'utente '**root**'.

```
$ su [Invio]

Password: ******* [Invio]

# mount -o loop -t iso9660 /home/tizio/prova.img ↩
↪                          /mnt/cdrom [Invio]
```

Se tutto va bene, da questo momento l'immagine risulta collegata alla directory '`/mnt/cdrom/`'. Al termine si esegue il distacco dell'immagine nel modo solito.

```
# umount /mnt/cdrom [Invio]
```

Volendo, un'immagine di un file system ISO 9660 può risiedere in un disco normale, o meglio, un disco normale può essere utilizzato come se fosse un disco ottico. Quello che si ottiene è sempre un'unità in sola lettura, perché il tipo di file system non consente la modifica.

Supponendo che con l'esempio precedente si fosse ottenuto un file di dimensione inferiore o al massimo uguale a quella di una partizione di un'unità USB esterna, supponendo che il file di dispositivo per raggiungerla sia '`/dev/sdc1`', si potrebbe riversare tale immagine nel modo seguente:

```
# cp /home/tizio/prova.img /dev/sdc1 [Invio]
```

Una volta trasferito il file si può innestare la partizione come se fosse un disco ottico.

```
# mount -t iso9660 /dev/sdc1 /mnt/cdrom [Invio]
```

Al termine si esegue il distacco del dischetto nel modo solito.

```
# umount /mnt/cdrom [Invio]
```

### 19.7.4.7 ISOLINUX

L'avvio di un disco ottico attraverso la tecnica dell'immagine di un dischetto, specificato con l'opzione '**`-b`**' di '**`genisoimage`**', può essere anche troppo limitante, soprattutto quando ci sarebbe l'esigenza di utilizzare un kernel Linux generalizzato e, per questo, molto più grande della capacità di un dischetto. Per questo, si può usare ISOLINUX,[40] il quale offre un metodo differente.

Per avviare un disco ottico attraverso ISOLINUX, si crea normalmente la directory '`isolinux/`' all'inizio della gerarchia che deve servire per realizzare il disco stesso. Al'interno di questa directory si collocano alcuni file, descritti nell'elenco seguente.

| File | Descrizione |
|---|---|
| '`isolinux.bin`' | Si tratta del programma di avvio di ISOLINUX. |
| '`isolinux.cfg`' | È il file di configurazione per l'avvio e si realizza in modo simile a quello di SYSLINUX (sezione 6.10). |
| '`vmlinuz`' | È il file del kernel Linux (ma si può usare anche un altro nome, in base alla configurazione del file '`isolinux.cfg`'). |

La directory '`isolinux/`' può contenere anche tutti gli altri file che possono essere richiesti per l'avvio, in base alla configurazione specificata nel file '`isolinux.cfg`'.

Se la propria distribuzione GNU/Linux organizza un pacchetto per ISOLINUX (oppure per SYSLINUX, ma che incorpora ISOLINUX), il file '`isolinux.bin`' già pronto può trovarsi nella directory '`/usr/lib/isolinux/`' (oppure '`/usr/lib/syslinux/`'): da

lì va copiato nella directory ‘`isolinux/`’ già descritta.

Purtroppo, ISOLINUX non è in grado di funzionare in tutte le situazioni possibili, soprattutto a causa delle differenze che ci sono nei firmware (BIOS) dei vari tipi di elaboratori, pur restando sempre nella stessa architettura x86. Per cercare di capire dove si blocca il sistema di avvio, al posto di usare il file ‘`isolinux.bin`’ che si trova nella directory ‘`/usr/lib/isolinux/isolinux.bin`’ (oppure ‘`/usr/lib/syslinux/isolinux.bin`’), conviene copiare il file ‘`isolinux-debug.bin`’, ma nella directory ‘`isolinux/`’ deve avere poi sempre il nome ‘`isolinux.bin`’.

A titolo di esempio, viene mostrato come potrebbe essere configurato il file ‘`isolinux.cfg`’:

```
DISPLAY intro.msg

TIMEOUT 0
PROMPT 1
DEFAULT auto

LABEL auto
 LOCALBOOT 0x80

LABEL hda
 KERNEL vmlinuz
 APPEND root=/dev/hda ro

LABEL hdb
 KERNEL vmlinuz
 APPEND root=/dev/hdb ro
```

```
LABEL hdc
 KERNEL vmlinuz
 APPEND root=/dev/hdc ro

LABEL hdd
 KERNEL vmlinuz
 APPEND root=/dev/hdd ro

LABEL sr0
 KERNEL vmlinuz
 APPEND root=/dev/sr0 ro

LABEL sr1
 KERNEL vmlinuz
 APPEND root=/dev/sr1 ro
```

In questo esempio, si può vedere la direttiva '**DISPLAY**', con la quale viene visualizzato il contenuto del file '`intro.msg`'; questo file si deve trovare, come gli altri, nella directory '`isolinux/`'. Successivamente, nelle direttive '**KERNEL**', viene specificato il nome del file che contiene il kernel: in questo caso è '`vmlinuz`' (sempre contenuto nella directory '`isolinux/`').

Senza entrare nel dettaglio della configurazione di ISOLINUX(si veda eventualmente la sezione 6.10), conviene vedere come va usato '**genisoimage**':

```
# cd directory_iniziale_gerarchia
```

Si suppone che la directory corrente sia quella iniziale, a partire dalla quale si articola la struttura che si vuole tradurre nel disco ottico:

```
# genisoimage -R -o /tmp/cdrom.img -b isolinux/isolinux.bin ↩
↪             -c isolinux/boot.cat ↩
```

```
↪              -no-emul-boot -boot-load-size 4 ↩
↪              -boot-info-table .
```

In questo caso, per comodità, si utilizza sempre la directory ‘`isolinux/`’ per inserire il file ‘`boot.cat`’.

## 19.8 Registrare su dischi ottici con GNU/Linux

«

Per registrare dati in un disco ottico, occorre un’unità per la «masterizzazione» e il software adatto per trasferire un’immagine contenente un file system adatto nel disco vergine. In questo capitolo viene spiegato il procedimento a grandi linee.

Il disco ottico che può essere preparato in casa in casa, potrebbe consentire una sola scrittura, oppure potrebbe essere riscrivibile più volte. Il disco ottico riscrivibile potrebbe richiedere ogni volta una reinizializzazione, oppure potrebbe consentire la riscrittura di porzioni di dati, in modo analogo ad altre unità di memorizzazione di massa comuni. Tuttavia, il disco ottico riscrivibile consente un numero limitato di riscritture, per cui la scrittura ripetuta di alcune zone porta velocemente all’inaffidabilità delle stesse; pertanto, è sempre bene trattare i dischi ottici come unità in sola lettura, salvo quando si procede a una riscrittura completa, ammesso che ciò sia possibile.

Difficilmente è possibile disporre di un solo programma per la scrittura di ogni tipo di disco ottico; pertanto, occorre valutare di volta in volta quale sia lo strumento più adatto.

### 19.8.1 Problemi legati alla scrittura di dischi ottici

La registrazione di un disco ottico è generalmente un'operazione a senso unico, senza possibilità di ripensamenti né ritardi. Se qualcosa non va, il disco che si ottiene può essere inutilizzabile. Vanno tenute a mente alcune regole fondamentali:

- il flusso di dati verso il masterizzatore dovrebbe essere costante e si dovrebbe mantenere alla velocità di registrazione impostata;
- il masterizzatore non deve essere interessato da vibrazioni;
- prima della registrazione, il disco deve risultare pulito perfettamente.

Il problema legato al flusso di dati costante è molto importante e a volta può costringere a non svolgere altre attività con l'elaboratore dove si sta registrando un disco ottico. Anche il problema della vibrazioni non va trascurato; un urto potrebbe rovinare la registrazione. Per ultimo va considerato anche il problema della pulizia del disco: prima della registrazione è opportuno afferrarlo con cura in modo da non sporcare il lato usato per la scrittura. Una volta registrato, è meno importante il problema della pulizia, se non si prevedono ulteriori riscritture.

### 19.8.2 Wodim

Wodim[41] è un programma in grado di registrare CD ed eventualmente anche DVD.

`wodim` [*opzioni_generali*] `dev=`*dispositivo* [*opzioni_di_traccia*] *traccia*...

Esistono vari modi di specificare il dispositivo da usare per la scrittura, ma in un sistema GNU/Linux è sufficiente indicare il file di dispositivo nel modo consueto. Eventualmente, per conoscere quali sono i dispositivi utilizzabili dal programma, si può sfruttare l'opzione '**`--devices`**'.

Le tracce sono semplicemente nomi di file da usare durante la registrazione. Se si intende utilizzare lo standard input si deve indicare un trattino singolo ('**`-`**').[42]

Tabella 19.99. Alcune opzioni generali. Le opzioni generali vanno indicate prima della specificazione delle coordinate necessarie a raggiungere l'unità di registrazione.

| Opzione | Descrizione |
|---|---|
| `-v` | Durante la registrazione emette più informazioni. |
| `-dummy` | Permette di eseguire tutte le operazioni in simulazione. In pratica, tutto avviene come nella realtà, tranne per il fatto che il laser rimane spento. |
| `-eject` | Espelle il disco alla fine delle operazioni. |
| `speed=`***velocità*** | Permette di definire la velocità di registrazione. Il valore da inserire è un numero intero corrispondente a un multiplo della velocità standard di riproduzione di un CD o di un DVD. |

| Opzione | Descrizione |
|---|---|
| `blank=`{`all` \| `fast` ↩ ↪ \| `session` ↩ ↪ \| `unclose` \| ***altra_opzione***} | Permette di cancellare un disco riscrivibile prima di un nuovo riutilizzo. Si indica una parola chiave per definire cosa cancellare esattamente. Il minimo è '**`fast`**' che si limita a sovrascrivere gli indici, consentendo un riutilizzo completo del disco; '**`all`**' cancella tutto; '**`session`**' elimina solo l'ultima sessione. Sono disponibili altre parole chiave per situazioni particolari, di cui si trova la descrizione nella pagina di manuale *wodim(1)*. |
| `driver=`***nome_driver*** | Permette di stabilire in che modo accedere all'unità di registrazione attraverso il nome che lo rappresenta sinteticamente (*driver*). Generalmente non è necessario specificarlo, a meno che '**`wodim`**' non sia in grado di riconoscere il tipo di unità. Si può usare precisamente l'opzione '**`driver=help`**' per conoscere l'elenco aggiornato dei nomi disponibili. |

Tabella 19.100. Alcune opzioni di traccia. Le opzioni di traccia possono essere alternate tra i nomi dei file che rappresentano le tracce da registrare. La validità dell'effetto di queste opzioni riguarda le tracce successive, fino a che non si incontrano opzioni contrarie.

| Opzione | Descrizione |
|---|---|
| `-data` | Questa opzione è predefinita e permette di specificare che le tracce seguenti contengono dati ISO 9660 o Rock Ridge. |

| Opzione | Descrizione |
|---|---|
| `-audio` | Indica che le tracce seguenti sono di tipo audio. |
| `-pad` | Aggiunge alla fine delle tracce una pausa di circa 15 secondi. Questa opzione può essere utile se il lettore risulta incapace di leggere gli ultimi settori delle tracce. |

Segue la descrizione di alcuni esempi.

- `# **wodim -devices**[*Invio*]`

  Scandisce le unità per la registrazione, elencando ciò che trova:

```
wodim: Overview of accessible drives (1 found) :
-------------------------------------------------------------
 0  dev='/dev/scd0'     rwrw-- : '_NEC' 'DVD_RW ND-3551A'
-------------------------------------------------------------
```

  In questo caso ha trovato l'unità '`/dev/scd0`', la quale risulta corrispondere a un masterizzatore CD/DVD per dischi riscrivibili.

- `# **wodim -v speed=1 dev=/dev/scd0 -data traccia.img**[*Invio*]`

  Inizia la registrazione a velocità normale del file '`traccia.img`', nell'unità corrispondente al file di dispositivo '`/dev/scd0`'.

- `# **wodim -v speed=4 dev=/dev/scd0 blank=fast**[*Invio*]`

  Ripristina rapidamente un CD o un DVD riscrivibile.

### 19.8.3 Growisofs e DVD+rw-format

Growisofs,[43] è il programma principale del pacchetto DVD+rw-tools, il quale consente la registrazione di dischi ottici DVD e BD:

```
growisofs [opzioni_generali] {-Z|-M} file_di_dispositivo ↩
↪opzioni_di_genisoimage
```

Il modello sintattico mostrato non è abbastanza chiaro e si rende necessario osservare gli esempi per comprendere il senso generale.

Il nome Growisofs deriva dal fatto che originariamente il suo scopo era di pilotare correttamente il programma '**genisoimage**' per consentire l'aggiunta di dati a un file system ISO 9660 su un'unità di memorizzazione che lo consente. Attualmente, Growisofs gestisce in modo trasparente diverse tecniche che consentono in pratica di aggiungere dati a un disco ottico.

Prima di passare agli esempi di utilizzo di Growisofs conviene conoscere l'uso delle opzioni principali specifiche di questo programma, ricordando che alla fine della riga di comando si inseriscono le opzioni e gli argomenti da dare a '**growisofs**', ammesso che non si intenda fornire un file-immagine già pronto per la masterizzazione.

Tabella 19.102. Alcune opzioni specifiche di Growisofs.

| Opzione | Descrizione |
|---|---|
| `-Z` ***file_di_dispositivo***<br><br>`-M` ***file_di_dispositivo*** | È obbligatorio l'uso di una di queste due opzioni, per stabilire il file di dispositivo corrispondente all'unità fisica da usare per la masterizzazione. L'opzione '**`-Z`**' (*zap*) dichiara che si tratta di una sessione iniziale, mentre l'opzione '**`-M`**' (*merge*) richiede l'aggiunta di una nuova sessione a quelle esistenti. Si osservi però che le sessioni multiple sono incompatibili con il file system UDF. |
| `-Z` ***file_di_dispositivo*=*file_immagine*** | Questa variante nell'uso dell'opzione '**`-Z`**', consente di fornire un file-immagine già pronto, senza bisogno di utilizzare '**`growisofs`**'. |
| `-dvd-compat` | Fa in modo di registrare una sola sessione, senza la possibilità di aggiungerne altre. |
| `-speed=`***n*** | Consente di specificare la velocità di scrittura, quando per qualche ragione non si vuole lasciare determinare automaticamente questo valore. |

Segue la descrizione di alcuni esempi di utilizzo del programma; si osservi che dopo l'opzione '**`-Z`**' o '**`-M`**', appaiono opzioni e argomenti di '**`mkisofs`**'.

- `#` **`growisofs -dvd-compat -Z /dev/sr0=/tmp/immagine.iso`**[*Invio*]

Copia il file-immagine '`/tmp/immagine.iso`' in un'unità di registrazione collocata nell'unità corrispondente al file di dispositivo '`/dev/sr0`', come prima e unica sessione. Può trattarsi indifferentemente di DVD o di BD

- # **`growisofs -Z /dev/sr0 -R -J /home/tizio/dati`**[*Invio*]

  Copia il contenuto della directory '`/home/tizio/dati/`' nel disco contenuto nell'unità '`/dev/sr0`', come prima sessione, a cui potrebbero essere aggiunte successivamente altre sessioni. Attraverso le opzioni di '**`genisoimage`**' si richiede di utilizzare le estensioni Rock Ridge e Joliet.

- # **`growisofs -M /dev/sr0 -R -J /home/tizio/dati`**[*Invio*]

  Copia il contenuto della directory '`/home/tizio/appunti/`' in una sessione successiva di un disco collocato nell'unità '`/dev/sr0`'; le opzioni di '**`genisoimage`**' che riguardano le estensioni al formato ISO 9660 devono essere le stesse usate nelle sessioni preesistenti.

- # **`growisofs -Z /dev/sr0=/dev/zero`**[*Invio*]

  Cancella completamente il contenuto di un DVD±RW inserito nell'unità corrispondente al file di dispositivo '`/dev/sr0`'. La cancellazione avviene perché si utilizza quanto si ottiene dalla lettura del file di dispositivo '`/dev/zero`'.

Il pacchetto DVD+rw-tools include anche un altro programma che può rivelarsi molto importante: DVD+rw-format. Come si può osservare, nella descrizione fatta di Growisofs mancano delle opzioni specifiche per la «cancellazione» di un disco e in effetti non c'è alcun

bisogno di fare questo, salvo quando si vuole utilizzare Growisofs in modo non interattivo all'interno di uno script.

Quando si utilizza Growisofs all'interno di uno script, con l'opzione '**-Z**', se il disco fornito risulta contenere già un file system ISO 9660, il procedimento viene interrotto, perché non c'è modo di dare all'utente il tempo di rinunciare all'operazione.

Il programma DVD+rw-format serve a compiere delle operazioni piuttosto «misteriose», simili all'inizializzazione di un disco comune:

```
dvd+rw-format [-force[=full]] [-blank[=full]] ↩
↪file_di_dispositivo
```

Il programma DVD+rw-format consente di conoscere e di modificare la ***modalità***, la quale può essere ***sequenziale*** o *restricted overwrite*. Per leggere la modalità, basta usare il programma senza opzioni:

```
# dvd+rw-tools /dev/sr0 [Invio]
```

In questo modo si legge la situazione del disco contenuto nell'unità corrispondente al file di dispositivo '`/dev/sr0`'. Ecco cosa si potrebbe ottenere:

```
* DVD±RW format utility by <appro@fy.chalmers.se>, version
4.9.
* 4.7GB DVD-RW media in Restricted Overwrite mode detected.
- media is already formatted, lead-out is currently at
  183200 KB which is 4.0% of total capacity.
- you have the option to re-run dvd+rw-format with:
  -force[=full] to enforce new format or mode transition
                and wipe the data;
  -blank[=full] to change to Sequential mode.
```

In situazioni difficili, dischi riscrivibili che non vogliono saperne di funzionare, si è rivelato utile agire con la sequenza di comandi seguente, dove comunque il punto numero due è facoltativo:

1. `#` **`dvd+rw-tools -force=full /dev/sr0`**[*Invio*]
2. `#` **`growisofs -Z /dev/sr0=/dev/zero`**[*Invio*]
3. `#` **`growisofs -Z /dev/sr0=immagine.iso`**[*Invio*]

In pratica, nell'ultimo passaggio viene copiato un file-immagine già pronto. L'errore osservato che nel modo appena mostrato è stato possibile risolvere si è manifestato così:

`#` **`growisofs -Z /dev/sr0=immagine.iso`**[*Invio*]

```
/dev/sr0: "Current Write Speed" is 1.0x1385KBps.
:-[ WRITE@LBA=0h failed with SK=5h/ASC=30h/ACQ=05h]:
Wrong medium type
:-( media is not formatted or unsupported.
:-( write failed: Wrong medium type
```

Per concludere è il caso di annotare che il programma '**`growisofs`**' potrebbe risiedere nel file system con i permessi «SUID-root», in modo che ogni utente possa accedere all'unità di scrittura di dischi

ottici senza difficoltà. Teoricamente, il programma dovrebbe essere in grado di usare tali privilegi solo per la registrazione del disco, tornando ai privilegi originali nel momento in cui deve accedere ai dati da memorizzare.

### 19.8.4 Dischi ottici con file system inusuali

«

Un disco ottico può essere registrato anche utilizzando per la traccia dati un file system differente dal solito ISO 9660 o da UDF, purché si preveda poi soltanto un accesso in lettura. In questo senso è da considerare la possibilità di utilizzare i dischi ottici per l'archiviazione di un file singolo (tar+gzip, o qualcosa del genere), senza la necessità di creare l'immagine di un file system vero e proprio.

È importante ricordare che un disco ottico contenente tracce audio, secondo le convenzioni originali del CD, **non può essere innestato**: l'ascolto richiede programmi appositi, così come l'eventuale estrazione delle tracce audio (sezione 30.6).

## 19.9 File system compressi

«

In alcune situazioni può essere conveniente l'utilizzo di un file system compresso. In generale, ci si può aspettare che un file system compresso funzioni correttamente esclusivamente quando questo è stato realizzato per accedervi in sola lettura; diversamente, un file system compresso che preveda e consenta la modifica dei dati, per quanto tecnicamente fattibile, sarebbe troppo poco affidabile e sarebbero maggiori i problemi rispetto ai vantaggi del suo utilizzo.

A fianco di file system compressi, veri e propri, si possono considerare anche sistemi basati sulla compressione trasparente dei singoli file; in tal caso non c'è più la necessità di determinare la dimensione dei blocchi.

### 19.9.1 Cloop

Con i sistemi GNU/Linux è possibile realizzare e accedere a un file system compresso con Cloop.[44] Il nome lascia intendere che si tratti di un file system contenuto in un file-immagine, compresso successivamente: *Compressed loop*.

La compressione è organizzata in blocchi, costituiti da multipli di 512 byte; l'accesso a un file system compresso con Cloop avviene attraverso l'utilizzo di un modulo apposito del kernel ('`cloop.o`'), con il quale si rende disponibile poi un file di dispositivo che appare come quello di un file system normale, ma in sola lettura. Si osservi che per il momento, Cloop consente di accedere a un solo file system compresso per volta.

Cloop nasce dall'esigenza di permettere a un disco ottico, da usare in sola lettura, di contenere più dati della propria capacità nominale, mantenendo un accesso fluido. Pertanto, si usa Cloop prevalentemente per file system di tipo ISO 9660, o UDF. D'altro canto, l'uso di un file system diverso, sarebbe controproducente sotto tutti i punti di vista.

Per poter gestire un file system Cloop è necessario disporre degli strumenti necessari alla compressione ed eventualmente all'estra-

zione di un file-immagine Cloop, oltre al modulo compilato per il proprio kernel.

Per arrivare ad avere un file system compresso, la prima cosa da fare è realizzare il file-immagine di questo, cosa che di solito si fa con '**genisoimage**' (sezione 19.7), quindi si passa alla compressione usando normalmente '**create_compressed_fs**' secondo uno dei due schemi seguenti:

```
create_compressed_fs immagine_originale immagine_cloop
```

```
cat immagine_originale | create_compressed_fs - immagine_cloop
```

L'esempio seguente mostra la creazione di un file system ISO 9660 con le estensioni Rock Ridge, a partire dal contenuto della directory '`/usr/`', che viene compresso per generare il file '`usr.cloop`':

```
# genisoimage -R /usr | create_compressed_fs - usr.cloop[Invio]
```

Se per qualche ragione fosse necessario ricreare il file-immagine di partenza, si può usare '**extract_compressed_fs**':

```
extract_compressed_fs immagine_cloop immagine_originale
```

```
cat immagine_cloop | extract_compressed_fs - immagine_originale
```

Per accedere al file system compresso senza estrarlo è necessario il modulo '`cloop.o`', il quale deve essere stato compilato espressamente per il proprio tipo di kernel. La descrizione della procedura da

seguire per questo è allegata ai sorgenti di Cloop e qui non viene descritta. Una volta installato correttamente il modulo (probabilmente è necessario avviare '**depmod -a**' per ricostruire le dipendenze tra i moduli), lo si deve caricare con l'indicazione contestuale del file contenente l'immagine compressa a cui si vuole accedere:

```
# modprobe cloop file=immagine_cloop [Invio]
```

Oppure:

```
# insmod cloop.o file=immagine_cloop [Invio]
```

Se si omette l'indicazione dell'argomento '**file=**...', si ottiene una segnalazione di errore e il modulo non viene caricato. Tornando all'esempio già visto, si potrebbe trattare del file '`usr.cloop`' collocato nella directory radice del file system in uso:

```
# modprobe cloop file=/usr.cloop [Invio]

cloop: Welcome to cloop v0.67
cloop: /usr.cloop: 12326 blocks, 65536 bytes/block, ↩
↪largest block is 65562 bytes.
```

Si ricorda che si può accedere a un solo file system Cloop per volta.

Dopo aver caricato il modulo si può accedere al file system come se questo fosse nella sua dimensione normale, non compresso, facendo riferimento a un file di dispositivo speciale che di solito è meglio creare al volo:

```
# mknod /dev/cloop b 240 0 [Invio]
```

Quando si ha la disponibilità del file di dispositivo '`/dev/cloop`', si può innestare il file system nel modo consueto:

```
mount -o ro -t tipo /dev/cloop punto_di_innesto
```

Tornando all'esempio già visto, se il file system è di tipo ISO 9660 e contiene quanto deve trovarsi nella directory '`/usr/`', si può usare il comando seguente:

```
# mount -o ro -t iso9660 /dev/cloop /usr [Invio]
```

### 19.9.2 Squashfs

«

Utilizzando un sistema con kernel Linux è disponibile anche il file system compresso Squashfs.[45] A differenza di Cloop, si tratta di un file system con caratteristiche proprie che non viene generato a partire da un altro tipo di file system. In pratica, il file system viene prodotto dai dati da archiviare e non da un altro file system non compresso. Questo lo rende più efficiente di Cloop e più semplice da utilizzare.

La creazione del file system avviene per mezzo del programma '**mksquashfs**', usato in uno dei due modi seguenti:

```
mksquashfs percorso file_squashfs [opzioni] [-e directory_esclusa...]
```

```
mksquashfs percorso_1 percorso_2... file_squashfs [opzioni] ↩
↪ [-e directory_esclusa...]
```

In generale, i percorsi indicati, i quali possono essere file o directory intere, vengono copiati all'interno del file-immagine indicato come ultimo argomento prima delle opzioni. Tuttavia, l'utilizzo di

una o dell'altra forma cambia il modo di inserire i dati relativi nel file-immagine di destinazione: se si archivia una directory soltanto, è il suo contenuto ad apparire nella directory radice del file system che viene generato, mentre in presenza di più percorsi (riferiti indifferentemente a file o a directory), questi vengono riprodotti nella destinazione mantenendo l'ultimo nome del percorso di origine. In generale, anche se sembra difficile da descrivere, in pratica questo funzionamento è quello intuitivo; si osservino i due esempi seguenti:

```
# mksquashfs /usr/bin bin.sqsh[Invio]

# mksquashfs /usr/bin /usr/sbin xbin.sqsh[Invio]
```

Nel primo caso si ottiene il file '`bin.sqsh`', contenente un file system di tipo '**`squashfs`**' in cui, a partire dalla directory radice si trovano subito i file eseguibili contenuti nella directory di origine; invece, nel secondo caso il file system contenuto in '`xbin.sqsh`' si compone delle directory '`/bin/`' e '`/sbin/`'.

Per poter accedere a un file system di tipo Squashfs è sufficiente che il kernel ne sia stato predisposto in fase di compilazione (il kernel Linux include le funzionalità per l'accesso a un file system Squashfs nei sorgenti originali), ma a differenza di Cloop, per l'innesto del file system non si richiedono accorgimenti particolari.

- *File systems*
  - *Miscellaneous filesystems*
    - *SquashFS*

L'esempio successivo mostra l'innesto del file system contenuto nel file-immagine '`/usr.sqsh`' a partire dalla directory '`/usr/`'.

```
# mount -o loop -o ro -t squashfs /usr.sqsh /usr [Invio]
```

### 19.9.3 Cramfs

Il file system compresso Cramfs,[46] per un kernel Linux, è molto simile per funzionamento a Squashfs, ma con qualche possibilità in meno. Anche in questo caso si comincia dalla creazione del file-immagine compresso, in modo simile a Squashfs, ma con il limite di una sola directory di origine:

```
mkcramfs -b dimensione_blocchi [altre_opzioni] ↩
↪directory_di_origine file_cramfs
```

Per innestare il file-immagine è necessario che il kernel Linux sia stato configurato e compilato per gestire tale tipo di file system:

- *File systems*
  - *Miscellaneous filesystems*
    - *Compressed ROM file system support*

Per innestare il file-immagine si procede in modo simile a quanto già presentato con Squashfs:

```
mount -o loop -o ro -t cramfs file_immagine directory_di_innesto
```

### 19.9.4 Zisofs

Il file system Zisofs è in realtà un meccanismo di compressione di file singoli. In pratica, si parte da un ramo di una directory e se ne crea una sorta di copia, dove i file normali sono compressi, mentre

tutto il resto rimane come prima (sottodirectory, collegamenti, ecc.). La gerarchia che si ottiene, teoricamente, può essere usata in lettura come se i file non fossero compressi, ma per questo occorre che il kernel sia predisposto allo scopo. In pratica, questa tecnica si usa per i file system ISO 9660 dei dischi ottici, come estensione specifica di GNU/Linux allo standard Rock Ridge, ma non si può applicare ai file system UDF.

La prima fase nella preparazione di un file system compresso, secondo questa tecnica, è quindi la creazione di una copia del ramo che interessa, compresso con '**mkzftree**', il quale fa parte di Zisofs-tools:[47]

```
mkzftree [opzioni] directory_origine directory_di_destinazione
```

```
mkzftree -F [altre_opzioni] origine destinazione
```

A titolo di esempio, si supponga di voler archiviare una parte dei file che servono al funzionamento del sistema operativo in un disco ottico; si tratta precisamente delle sole directory '`/bin/`', '`/etc/`', '`/lib/`' e '`/sbin/`'. La prima fase consiste nella creazione di una nuova gerarchia compressa, per esempio a partire da '`/var/tmp/archiviazione/`', che viene creata appositamente:

```
# mkdir /var/tmp/archiviazione[Invio]

# mkzftree /bin /var/tmp/archiviazione/bin[Invio]

# mkzftree /etc /var/tmp/archiviazione/etc[Invio]

# mkzftree /lib /var/tmp/archiviazione/lib[Invio]
```

```
# mkzftree /sbin /var/tmp/archiviazione/sbin [Invio]
```

Al termine di queste operazioni, se si va a controllare il contenuto della directory '`/var/tmp/archiviazione/`', si può avere l'impressione che nulla sia cambiato, a parte la dimensione dei file e il loro contenuto effettivo.

Teoricamente, se il file '`/etc/magic`' è stato aggiornato correttamente, dovrebbe essere possibile riconoscere questi file come '**`zisofs`**'; tuttavia, potrebbe essere necessario copiarli al di fuori del loro contesto per individuarli come tali.

Nella struttura creata in questo modo, **si possono anche inserire file non compressi**, come potrebbero essere i file necessari all'avvio del sistema (il kernel e altri), oppure un file di spiegazioni per chi dovesse tentare di leggere i dati senza sapere come fare.

Nel modello sintattico appare in evidenza la possibilità di usare l'opzione '**`-F`**' (ovvero '**`--file`**'). Questa opzione è utile quando l'origine e la destinazione possono essere qualcosa di diverso da directory. In pratica, con questa opzione si può comprimere anche un file singolo, mentre altrimenti si otterrebbe una segnalazione di errore.

Una volta realizzata la struttura, si passa normalmente alla creazione del file-immagine, con gli strumenti soliti, specificando che si va a creare un file system con estensioni Zisofs. Si osservi che in questo caso non ha più senso inserire estensioni Joliet e nello stesso modo

non serve più la creazione di file '`TRANS.TBL`'. Viene mostrato un modello semplificato per l'uso di '**`mkisofs`**':

```
mkisofs  -z {-r|-R} [altre_opzioni] -o file_immagine  directory
```

Per esempio, per creare il file-immagine '`/var/tmp/immagine.iso`' dal contenuto di '`/var/tmp/archiviazione/`', dovrebbe essere sufficiente agire in questo modo:

```
# mkisofs -z -R -o /var/tmp/immagine.iso ↩
↪                  /var/tmp/archiviazione[Invio]
```

Successivamente si deve passare alla registrazione del disco ottico a partire dall'immagine (si veda per questo la sezione 19.7).

Quando si innesta un disco ottico realizzato con le estensioni Zisofs, se il sistema operativo è in grado di riconoscerle, dovrebbe leggere i dati in modo trasparente, senza lasciare avvertire che si tratta di dati compressi; tuttavia rimane il fatto che la dimensione dei file potrebbe risultare fasulla se letta con gli strumenti normali. Nel caso particolare di un sistema GNU/Linux, occorre che il kernel sia stato compilato includendo le funzionalità seguenti:

- *File systems*
  - *ISO 9660 CD-ROM file system support*
  - *Transparent decompression extension*

In mancanza di queste estensioni, la lettura dei dati richiede un'estrazione manuale, sempre con l'uso di '**`mkzftree`**', ma questa volta con l'opzione '**`-u`**':

```
mkzftree -u [altre_opzioni] directory_origine directory_di_destinazione
```

Per esempio, si può fare una copia leggibile di un disco ottico, innestato nella directory '`/mnt/cdrom/`', copiando i dati nella directory '`/tmp/cdrom/`' (che viene creata automaticamente):

```
# mkzftree -u /mnt/cdrom /tmp/cdrom [Invio]
```

Maggiori dettagli sull'uso di '**`mkzftree`**' si possono leggere nella pagina di manuale *mkzftree(1)*.

## 19.10 Gestione della memoria di massa in modo ridondante

Le unità di memorizzazione di massa sono componenti che possono avere parti meccaniche sottoposte a usura, ma qualunque sia la tecnologia, tali unità sono soggette a guasti, come qualunque altra parte di un sistema di elaborazione di dati.

In generale è necessario premunirsi dalle perdite di dati, dovute a guasti o al deterioramento dei supporti, con delle copie di sicurezza, da eseguire periodicamente; tuttavia, anche predisponendo una procedura efficace per le copie di sicurezza, l'impegno necessario per il ripristino del funzionamento del proprio elaboratore è sempre rilevante.

Dal momento che la frequenza con cui si verifica una rottura di un disco fisso, o di ciò che ne fa la funzione, è maggiore rispetto ad altri eventi che possono procurare la perdita dei dati (per esempio la disattenzione, il furto, l'atto vandalico, l'incendio, l'allagamento, il terremoto, ecc.), vale la pena di attenuare l'effetto di questo tipo di

problema con la realizzazione di un insieme di unità ridondanti, noto normalmente con la sigla «RAID».

### 19.10.1 RAID

La sigla RAID sta per *Redundancy array of inexpensive disks* e rappresenta quindi un insieme di tecniche per mettere assieme delle unità di memorizzazione aggiungendo qualche forma di ridondanza. Si distinguono diversi livelli, alcuni dei quali sono descritti brevemente nell'elenco seguente.

| Livello RAID | Descrizione |
|---|---|
| RAID-0 | Il livello zero dei sistemi RAID è privo di ridondanza e si occupa soltanto di unire assieme più unità di memorizzazione, all'interno dei quali i dati vengono suddivisi equamente, in modo da bilanciare anche il carico di operazioni di lettura e scrittura che le riguardano. In pratica, il livello zero consente di realizzare un'unità virtuale di grandi dimensioni, più efficiente, ma la rottura di una delle unità del gruppo porta alla perdita di tutti i dati. RAID-0 è noto anche con la definizione *block striping*. |
| RAID-1 | Il livello uno dei sistemi RAID si occupa di unire assieme due o più unità di memorizzazione (ma di solito sono solo due) che si trovano a riprodurre fedelmente tutte la stessa cosa. In questo caso, la rottura di una unità non pregiudica l'utilizzo dei dati che sono disponibili nelle unità rimanenti. RAID-1 è noto anche con la definizione *disk mirroring*. |

| Livello RAID | Descrizione |
|---|---|
| RAID-3 | Il livello tre dei sistemi RAID si occupa di unire assieme più unità di memorizzazione, all'interno delle quali i dati vengono suddivisi equamente, in modo da bilanciare anche il carico di operazioni di lettura e scrittura che le riguardano, dedicando una di queste unità al contenimento di un sistema di codici di controllo che permettono di ricostruire i dati nel caso una delle altre unità si rompa. In pratica, il livello tre crea una forma di ridondanza, ma meno affidabile del livello uno. Naturalmente, per attuare un livello tre servono almeno tre unità di memorizzazione. |
| RAID-4 | Il livello quattro dei sistemi RAID è simile al livello tre, con la differenza che vengono distribuiti in modo differente (più efficiente) tra le unità di memorizzazione, ma rimane compito di un'unità separata il sistema di codici di controllo che permette la ricostruzione dei dati. |
| RAID-5 | Il livello cinque dei sistemi RAID equivale al livello quattro, dove però le informazioni che servono per la ricostruzione dei dati sono distribuite tra le unità, senza essere così concentrate in una soltanto. In questo modo si aumenta l'efficienza, in termini di tempi di accesso ai dati. |

I livelli RAID possono essere anche combinati assieme; per esempio è possibile realizzare due insiemi RAID-0, i quali si associano a loro volta in un insieme RAID-1, in modo da non perdere i dati nel caso una delle unità smetta di funzionare. In questo caso si parla anche di RAID-10

### 19.10.2 RAID hardware e software

La realizzazione di un insieme di unità di memorizzazione secondo uno dei vari livelli RAID può essere ottenuta attraverso sistemi hardware, oppure software. Nel primo caso si tratta di unità di controllo che gestiscono tutto autonomamente, facendo sì che il sistema operativo veda in pratica un'unità di memorizzazione «normale»; nel secondo caso, è il sistema operativo che associa le unità di memorizzazione e le gestisce usando una forma di ridondanza.

Come si può comprendere, le unità di controllo RAID sono più costose di quelle prive di tale funzionalità; tuttavia, se non si creano altri tipi di problemi, hanno il vantaggio di non creare difficoltà al sistema operativo.

### 19.10.3 RAID-1 via software con i sistemi GNU/Linux

Con i sistemi GNU/Linux è relativamente agevole la creazione di un insieme RAID senza l'ausilio di unità di controllo specializzate. In questa sezione si descrive brevemente la realizzazione di un sistema composto da due unità di memorizzazione che riproducono esattamente gli stessi dati, ovvero un sistema RAID-1, cosa che in generale rappresenta la situazione più semplice da affrontare per chiunque.

Se la propria distribuzione GNU/Linux offre una procedura di installazione che prevede anche la gestione di un sistema RAID-1 software, dovrebbe essere sufficiente disporre delle due unità di memorizzazione perché tutto venga predisposto senza interventi particolari; tuttavia, qui si vuole mostrare un procedimento più brutale, da affrontare normalmente quando si vuole travasare un sistema GNU/Linux preesistente in un insieme RAID-1. In pratica, si fa affi-

damento sul fatto che sia disponibile una terza unità di memorizzazione, contenente il sistema GNU/Linux da trasferire, da rimuovere successivamente dall'elaboratore, o da destinare ad altri scopi. Naturalmente è necessario organizzarsi in modo da poter avviare questo sistema transitorio anche quando lo si sposta in un'altra unità di controllo o comunque in un'altra posizione della stessa.

#### 19.10.3.1 Predisposizione del kernel

«

Per poter avviare un sistema GNU/Linux che utilizza un insieme RAID (gestito via software), è indispensabile che il kernel includa tutte le funzionalità necessarie, senza avvalersi per questo di moduli. Se possibile, è meglio includere tutte le funzionalità associate ai livelli RAID della voce *Multi-device support (RAID and LVM)*. Se non è possibile includere tutte le funzionalità corrispondenti, occorre selezionare almeno *RAID support* e *RAID-1 (mirroring) mode* (dal momento che si vuole realizzare un sistema RAID-1).

Se il kernel che si utilizza è predisposto per la gestione di insiemi RAID, appare il file virtuale '`/proc/mdstat`', dal quale, successivamente, si possono ottenere le informazioni sull'attività del sistema RAID. Tanto per cominciare si può vedere quali livelli RAID sarebbero disponibili:

```
$ cat /proc/mdstat [Invio]

Personalities : [linear] [raid0] [raid1] [raid5] [multipath]
…
```

Dall'esempio si comprende che RAID-1 è disponibile (insieme anche ad altri livelli RAID).

### 19.10.3.2 Partizioni e installazione

È opportuno che le unità di memorizzazione da associare in un insieme RAID siano suddivise in partizioni, tenendo conto delle esigenze che potrebbero porsi in un secondo momento. Infatti, se si realizza un sistema RAID, lo si fa presumibilmente per mantenere intatta la situazione nel tempo. Probabilmente, nelle situazioni più semplici è sufficiente predisporre tre partizioni: una per i file necessari all'avvio (kernel incluso), una per la memoria virtuale e la terza per tutto il resto.

È possibile inserire i file necessari all'avvio del sistema operativo anche in una partizione gestita attraverso un sistema RAID, ma in generale è una cosa sconsigliabile e comunque non esiste una reale necessità di agire in questo modo, se ci si organizza correttamente.

Con il sistema GNU/Linux attualmente in funzione (quello che poi deve essere travasato), si procede alla suddivisione in partizioni. Seguendo l'ipotesi che è stata formulata si dividono entrambe le unità in modo da avere una piccola partizione (da 100 Mibyte in su) di tipo $83_{16}$ (Linux-nativa), una partizione di tipo $82_{16}$ (Linux-swap) e una partizione di tipo $FD_{16}$. Il tipo $FD_{16}$ (autorilevamento RAID) dichiara esplicitamente che si tratta di una partizione usata per la realizzazione di sistemi RAID e permette al kernel di attivarla prima di innestare il file system principale.

Teoricamente sarebbe possibile utilizzare in modo speculare anche le partizioni per la memoria virtuale; in pratica questo non conviene

e l'intenzione, qui, è solo quella di utilizzarle in modo da sommare la loro capacità.

Una volta predisposte le partizioni, facendo in modo che siano il più possibile uguali, si può procedere a una prima inizializzazione che attiva la funzionalità RAID-1.

Per rendere l'esempio a cui si fa riferimento più realistico, si suppone di dover accoppiare le unità '`/dev/sda`' e '`/dev/sdb`'. Pertanto, le partizioni potrebbero essere organizzate così:

| | | |
|---|---|---|
| '`/dev/sda1`' | '`/dev/sdb1`' | partizioni per il kernel e gli altri file necessari all'avvio; |
| '`/dev/sda2`' | '`/dev/sdb2`' | partizioni da gestire in modo speculare secondo il modello RAID-1; |
| '`/dev/sda4`' | '`/dev/sdb4`' | partizioni per la memoria virtuale. |

La partizione virtuale RAID-1 viene associata a un file di dispositivo differente, secondo il modello '`/dev/md`***n***', dove il numero ***n*** dipende da una scelta abbastanza libera. Si ritiene qui che sia più conveniente utilizzare lo stesso numero che individua la partizione nel disco, in modo da non creare confusione; pertanto si intende creare il partizione RAID '`/dev/md2`'. A questo punto occorre accertarsi di disporre del file di dispositivo relativo; se manca deve essere creato:

```
# mknod -m 0660 /dev/md2 b 9 2 [Invio]
```

Tabella 19.109. Creazione dei primi file di dispositivo necessari alla gestione di dischi RAID.

| File di dispositivo | Comando |
|---|---|
| '`/dev/md0`' | `# mknod -m 0660 /dev/md0 b 9 0` [*Invio*] |

| File di dispositivo | Comando |
|---|---|
| ‘/dev/md1’ | # **mknod -m 0660 /dev/md1 b 9 1** [*Invio*] |
| ‘/dev/md2’ | # **mknod -m 0660 /dev/md2 b 9 2** [*Invio*] |
| ‘/dev/md3’ | # **mknod -m 0660 /dev/md3 b 9 3** [*Invio*] |
| ‘/dev/md4’ | # **mknod -m 0660 /dev/md4 b 9 4** [*Invio*] |
| ‘/dev/md5’ | # **mknod -m 0660 /dev/md5 b 9 5** [*Invio*] |
| ‘/dev/md6’ | # **mknod -m 0660 /dev/md6 b 9 6** [*Invio*] |

Per creare l’unità RAID-1 si procede con questo comando, con Mdadm:[48]

```
# mdadm --create /dev/md2 --level=raid1 --raid-devices=2 ↩
↪       /dev/sda2 /dev/sdb2 [Invio]
```

Se tutto procede regolarmente, l’unità RAID-1 viene attivata quasi subito, ma inizia una scansione che serve ad allineare le due partizioni (in pratica si può vedere che i dischi sono attivi in modo costante per lungo tempo). Durante questa fase, è già possibile procedere a inizializzare l’unità RAID-1 con il tipo di file system che si preferisce; in questo caso si sceglie il tipo Ext4:

```
# mkfs.ext4 /dev/md2 [Invio]
```

Terminata l’inizializzazione si può innestare la partizione e procedere con la copia del sistema attivo (o di un altro se necessario). Si suppone di volere innestare la partizione nella directory ‘/mnt/’ e di procedere con la copia di conseguenza:

```
# mount -t ext4 /dev/md2 /mnt [Invio]

# cd / [Invio]
```

```
# cp -dpRv /bin /mnt [Invio]

# cp -dpRv /etc /mnt [Invio]

# cp -dpRv /home /mnt [Invio]

# cp -dpRv /lib /mnt [Invio]

# cp -dpRv /root /mnt [Invio]

# cp -dpRv /sbin /mnt [Invio]

# cp -dpRv /usr /mnt [Invio]

# cp -dpRv /var /mnt [Invio]

# mkdir /mnt/boot [Invio]

# mkdir /mnt/dev [Invio]

# mkdir /mnt/mnt [Invio]

# mkdir /mnt/proc [Invio]

# mkdir /mnt/tmp [Invio]

# chmod 1777 /mnt/tmp [Invio]

# cd /mnt/dev [Invio]

# MAKEDEV generic [Invio]
```

Questa sequenza di operazioni fa riferimento a un sistema GNU/Linux generico e non tiene conto delle particolarità che ha ogni singola distribuzione; pertanto va adattato alla propria realtà.

Fatta la copia, è necessario modificare immediatamente il file ‘`etc/`

fstab' della partizione innestata nella directory '/mnt/' (pertanto si tratta temporaneamente del file '/mnt/etc/fstab'), in modo da fare riferimento correttamente al file di dispositivo '/dev/md2' per il file system principale. In base all'esempio a cui si sta facendo riferimento, il file 'etc/fstab' che si va a predisporre dovrebbe contenere le righe seguenti:

```
/dev/sda4 none          swap sw                                0  0
/dev/sdb4 none          swap sw                                0  0
/dev/md2  /             auto defaults,errors=remount-ro        0  1
/dev/sda1 /boot         auto defaults,errors=remount-ro        0  1
/dev/sdb1 /boot-mirror auto defaults,errors=remount-ro         0  1
proc      /proc         proc defaults                          0  0
```

Come si può osservare, la partizione corrispondente al file di dispositivo '/dev/sdb1' viene innestata per comodità nella directory '/boot-mirror/', in modo da rammentare all'amministratore del sistema operativo che deve occuparsi di mantenere il suo contenuto allineato a quello della partizione innestata invece nella directory '/boot/'.

Il contenuto delle partizioni '/dev/sda1' e '/dev/sdb1' va organizzato per assicurare l'avvio del sistema, anche quando una delle due unità ('/dev/sda' o '/dev/sdb') dovesse mancare. Si veda a questo proposito il capitolo 6. Quindi, prima di riavviare il sistema operativo per vedere se funziona effettivamente, conviene controllare che l'attività di sincronizzazione delle due partizioni sia stata completata, osservando il file '/proc/mdstat':

```
# cat /proc/mdstat [Invio]
```

```
Personalities : [linear] [raid0] [raid1] [raid5] [multipath]
read_ahead 1024 sectors
md2 : active raid1 sda2[0] sdb2[1]
      53897920 blocks [2/2] [UU]

unused devices: <none>
```

L'esempio mostra una situazione di riposo.

### 19.10.3.3 Sostituzione di un'unità difettosa

«

Quando si verifica l'evento che si teme, ovvero che una delle due unità di memorizzazione risulti difettosa, teoricamente il sistema RAID dovrebbe continuare a funzionare escludendo quella unità. Una volta appurato questo fatto, occorre provvedere alla sua sostituzione.

È necessario quindi procurare un'altra unità di memorizzazione, con una capacità uguale o maggiore di quello che si deve sostituire, cercando di riprodurre la stessa suddivisione precedente. Probabilmente, trattandosi di dischi di grandi dimensioni, la geometria apparente dovrebbe risultare compatibile (255 testine; 63 settori per traccia e una quantità di cilindri uguale o maggiore rispetto all'unità precedente). A ogni modo, si possono riprodurre fedelmente le stesse partizioni anche se la geometria non dovesse essere la stessa, pertanto, prima di rimuovere l'unità difettosa, si potrebbe salvare la mappa delle partizioni con il comando seguente:

```
# sfdisk -d /dev/sda > /root/mappa[Invio]
```

Come si intende, l'unità difettosa dovrebbe essere in questo caso quella corrispondente al file di dispositivo '`/dev/sda`'. Il file che si ottiene potrebbe avere l'aspetto seguente:

```
# partition table of /dev/sda
unit: sectors

/dev/sda1 : start=       63, size=    48195, Id=83
/dev/sda2 : start=    48258, size=   963900, Id=82
/dev/sda3 : start=  1012158, size=107796087, Id=fd
```

Una volta sostituito il disco, si possono riprodurre le stesse partizioni (avanzando probabilmente dello spazio libero alla fine, se l'unità ha una capacità maggiore della precedente) con il comando seguente:

```
# sfdisk /dev/sda < /root/mappa [Invio]
```

A questo punto, conviene controllare con '**fdisk**', ma non dovrebbe essere il caso di modificare la suddivisione in partizioni, anche se questa non è conforme alla geometria dell'unità. Quindi si può procedere all'inizializzazione delle partizioni che sono escluse dal sistema RAID (in questo caso '`/dev/sda1`' usata per il sistema di avvio e '`/dev/sda4`' usata per la memoria virtuale); inoltre, se necessario, occorre ripristinare il sistema di avvio. Quindi si può informare il sistema RAID che si può riappropriare della partizione '`/dev/sda2`' con il comando seguente:

```
# mdadm --add /dev/md2 /dev/sda2 [Invio]
```

Naturalmente si sta facendo riferimento all'esempio che appare nella sezione precedente; in ogni caso, '`/dev/md2`' deve essere quanto risulta dalla lettura del file '`/proc/mdstat`'.

Dopo l'uso di questo comando il sistema RAID si prende possesso della partizione e inizia la ricostruzione della copia speculare al suo interno.

### 19.10.3.4 Situazioni più gravi

«

Il sistema RAID-1, attuato via software attraverso un kernel Linux, ha il vantaggio di mantenere leggibili le partizioni RAID come se fossero partizioni inizializzate normalmente. In pratica, le informazioni che riguardano l'insieme dei dischi RAID vengono collocate alla fine della partizione, in quello che viene chiamato «super blocco» (dell'insieme RAID), il quale consente al kernel di conoscere la struttura dell'insieme RAID senza bisogno di file di configurazione. Pertanto, se per qualche ragione non si può fare diversamente, si può accedere a questi dati, **avendo cura di farlo soltanto in lettura**, ignorando il sistema RAID-1.

In conclusione, se un evento accidentale rendesse inutilizzabile l'insieme di dischi RAID-1, dovrebbe rimanere almeno la possibilità di recuperare i dati da uno dei dischi accedendo alle partizioni nel modo normale, ovvero non attraverso i dispositivi '`/dev/md`***n***'.

### 19.10.3.5 Aggiungere un'unità

«

Una volta fatta l'esperienza con una coppia di dischi RAID-1, quando si manifestano i primi problemi e si comprende quanto è importante disporre di un sistema di memorizzazione affidabile, è facile desiderare di migliorare l'affidabilità del sistema con l'aggiunta di una terza copia speculare. Si tratta naturalmente di un'idea lodevole, anche se comporta una spesa per l'aggiunta di un altro disco.

Seguendo l'esempio già proposto in precedenza, si parte da una situazione in cui esistono due unità di memorizzazione, dove le partizioni '`/dev/sda2`' e '`/dev/sdb2`' compongono un sistema RAID-1 '`/dev/md2`'. Si vuole aggiungere un'altra unità, nella quale ri-

produrre più o meno la stessa suddivisione in partizione di quelle già installate, in modo da utilizzare la partizione '/dev/sdc2' come terza copia speculare del sistema RAID-1 '/dev/md2'. Dopo aver installato fisicamente l'unità aggiuntiva e avere riavviato il sistema, occorre informare della presenza di un terzo disco:

```
# mdadm --grow --raid-devices=3 /dev/md2 [Invio]
```

Si può verificare nel file virtuale '/proc/mdstat':

```
# cat /proc/mdstat [Invio]

Personalities : [linear] [raid0] [raid1] [raid5] [multipath]
read_ahead 1024 sectors
md2 : active raid1 sda2[0] sdb2[1]
      53897920 blocks [3/2] [UU_]

unused devices: <none>
```

Come si può osservare, pur essendo previste tre partizioni in gioco, in realtà solo due sono operative. Pertanto, occorre aggiungere la partizione mancante:

```
# mdadm --add /dev/md2 /dev/sdc2 [Invio]
```

A questo punto parte la sincronizzazione della partizione '/dev/sdc2' e al termine si dovrebbe osservare il funzionamento parallelo delle tre partizioni:

```
# cat /proc/mdstat [Invio]
```

```
Personalities : [linear] [raid0] [raid1] [raid5] [multipath]
read_ahead 1024 sectors
md2 : active raid1 sda2[0] sdb2[1] sdb3[2]
      53897920 blocks [3/3] [UUU]

unused devices: <none>
```

### 19.10.3.6 Creare un sistema RAID-1 senza dover travasare i dati

«

È possibile realizzare un sistema RAID-1 utilizzando una partizione che contiene già un file system Second-extended (Ext2, Ext3 o Ext4), senza bisogno di travasare i dati e, soprattutto, senza perderli.

Si suppone di avere una situazione come quella della tabella successiva, dove la partizione '`/dev/sda2`' è quella che viene utilizzata:

| | | |
|---|---|---|
| '`/dev/sda1`' | '`/dev/sdb1`' | partizioni per il kernel e gli altri file necessari all'avvio; |
| '`/dev/sda2`' | '`/dev/sdb2`' | partizioni da gestire in modo speculare secondo il modello RAID-1. |
| '`/dev/sda4`' | '`/dev/sdb4`' | partizioni per la memoria virtuale. |

Per poter fare le trasformazioni necessarie è necessario avviare il sistema con un disco esterno, come un disco ottico autoavviabile, dove deve essere disponibile il programma Mdadm.

Si procede con la modifica del tipo di partizione, sia per '`/dev/sda2`', sia per '`/dev/sdb2`', in modo che corrisponda a $FD_{16}$ (autorilevamento RAID); per questo basta usare '**`fdisk`**', avendo cura di non modificare altro.

Con Mdadm si dichiara l'unità RAID '`/dev/md2`' (si ricordi che potrebbe essere necessario provvedere alla creazione del file di dispositivo '`/dev/md2`'), associando inizialmente solo la partizione '`/dev/sda2`':

```
# mdadm --create /dev/md2 --level=raid1 --raid-devices=2 ↩
↪       /dev/sda2 missing[Invio]
```

Si osservi alla fine della riga di comando dell'esempio appena mostrato la presenza della parola chiave '**`missing`**', con la quale si evita temporaneamente l'indicazione della seconda partizione.

```
  mdadm: /dev/sda2 appears to contain an ext2fs file system
      size=35782656K  mtime=Sat Jan  2 19:24:20 2010
Continue creating array? y[Invio]

  mdadm: array /dev/md2 started.
  md: bind<sda2>
  raid1: raid set md2 active with 1 out of 2 mirrors
```

Le informazioni sulla gestione RAID vengono inserite alla fine della partizione, che, di conseguenza, ne risulta ridotta. È necessario procedere a un aggiustamento:

```
# fsck.ext4 -f /dev/md2[Invio]

  e2fsck 1.37 (21-Mar-2005)
  Pass 1: Checking inodes, blocks, and sizes
  Pass 2: Checking directory structure
  Pass 3: Checking directory connectivity
  Pass 4: Checking reference counts
  Pass 5: Checking group summary information
  /dev/md2: 404645/4481568 files (2.5% non-contiguous), ↩
↪5216591/8945664 blocks
```

```
# resize2fs /dev/md2 [Invio]

  resize2fs 1.37 (21-Mar-2005)
  Resizing the filesystem on /dev/md2 to 8946176 (4k) blocks.
  The filesystem on /dev/md2 is now 8945664 blocks long.
```

Infine si aggiunge l'altra partizione che prima è stata lasciata da parte:

```
# mdadm --add /dev/md2 /dev/hdc2 [Invio]
```

Naturalmente occorre ricordare di modificare il file '`/etc/fstab`' nell'unità '`/dev/md2`' e di sistemare il sistema di avvio, come già descritto.

## 19.11 FUSE: «file system in user space»

FUSE, ovvero *File system in user space*, è un file system virtuale generico per i sistemi GNU/Linux. In pratica questa funzione è gestita all'interno del kernel Linux, ma il file system virtuale che si ottiene dipende da un programma apposito esterno, che comunica con il kernel secondo il protocollo di FUSE.

Attraverso il meccanismo di FUSE, si evita di dover inserire nel kernel la gestione di ogni tipo di file system (reale o virtuale che sia), lasciando il compito di interfacciarsi con quello a un programma esterno. Inoltre, si danno delle possibilità in più agli utenti, ma per comprendere questa cosa occorre provare.

### 19.11.1 Attivazione di FUSE

Nei kernel attuali delle distribuzioni GNU/Linux comuni, la funzionalità di FUSE viene inclusa quasi sempre come modulo; eventualmente occorre intervenire nel menù *File systems*:

- *File systems*
  - *File system in user space support*

Se la funzionalità di FUSE è relegata in un modulo del kernel, occorre provvedere a caricarlo:

```
# modprobe fuse [Invio]
```

Tuttavia, perché FUSE sia attivo è necessario che sia presente anche un file di dispositivo particolare: '`/dev/fuse`'. Se non ci sono sistemi automatici per la gestione dei file di dispositivo, potrebbe essere necessario crearlo la prima volta:

```
# mknod -m 0660 /dev/fuse c 10 229 [Invio]

# chown root:fuse /dev/fuse [Invio]
```

Per la gestione di FUSE si utilizza il programma '**`fusermount`**', che deve essere installato. I permessi del programma '**`fusermount`**', assieme a quelli del file di dispositivo '`/dev/fuse`', condizionano la facoltà degli utenti di innestare i file system compatibili con FUSE. Di norma, i permessi potrebbero essere $4750_8$, appartenendo all'utente '**`root`**' e al gruppo '**`fuse`**':

```
# ls -l /usr/bin/fusermount [Invio]

-rwsr-x--- 1 root fuse … /usr/bin/fusermount
```

In questo modo, solo l'utente '**`root`**' e gli utenti associati al gruppo '**`fuse`**' possono eseguire gli innesti. Eventualmente, per consentire a tutti di agire in tal modo, basta aggiungere i permessi di lettura e di esecuzione al programma:

```
# chmod a+rx /usr/bin/fusermount [Invio]
```

```
# ls -l /usr/bin/fusermount[Invio]

  -rwsr-xr-x 1 root fuse … /usr/bin/fusermount
```

Inoltre, sempre per consentire a tutti gli utenti di innestare un file system, occorre dare al file di dispositivo '`/dev/fuse`' i permessi di lettura e scrittura per tutti:

```
# chmod a+rw /dev/fuse[Invio]

# ls -l /dev/fuse[Invio]

  crw-rw-rw- 1 root fuse 10, 229 … /dev/fuse
```

### 19.11.2 Limiti di accesso

«

Quando si innesta un file system attraverso FUSE, viene sempre usato il programma '**`fusermount`**', anche se non in modo diretto. Pertanto, come descritto nella sezione precedente, i permessi di questo programma, assieme a quelli del file di dispositivo '`/dev/fuse`', possono limitare agli utenti la facoltà di eseguire tali innesti. Tuttavia, una volta eseguito un innesto, in condizioni normali solo l'utente che lo ha eseguito può accedere alla gerarchia che si articola a partire da quel punto (dalla directory che costituisce il punto di innesto), escludendo anche l'utente '**`root`**'.

Se il file system che si va a gestire con FUSE, contiene informazioni riconducibili a utenze, permessi ed eventualmente numeri inode, possono essere utili delle opzioni di innesto, per far sì che gli accessi siano gestiti in modo più o meno usuale. Sotto vengono elencate le opzioni più comuni, ma dipende dal programma frontale usato il modo in cui vi si fa riferimento effettivamente.

Tabella 19.123. Alcune opzioni comuni relative all'innesto di un file system con FUSE.

| Opzione | Descrizione |
|---|---|
| `allow_root` | Si consente l'accesso all'utente '**`root`**' con tutti i suoi privilegi. |
| `allow_other` | Si consente l'accesso anche agli altri utenti. |
| `default_permissions` | Si considerano i permessi dei file nel modo consueto. |
| `use_ino` | Abilita l'uso dei numeri inode. |

Si osservi che, in generale, l'utente comune non può usare l'opzione '**`allow_other`**', a meno di inserire la direttiva '**`user_allow_other`**' nel file '`/etc/fuse.conf`':

```
…
user_allow_other
…
```

Inoltre, è evidente che un utente comune che innesta un file system con FUSE, volendo attribuire le opzioni '**`allow_other`**' e '**`default_permissions`**', permette sì agli altri utenti di accedere in base ai permessi esistenti, ma i file che vengono creati (ammesso che ci siano tali permessi), appartengono generalmente a un utente prestabilito, che di solito coincide con quello che ha eseguito l'innesto. Infatti, non si deve dimenticare che l'accesso al file system avviene tramite un programma, che generalmente funziona con i privilegi di quel tale utente. Pertanto, le opzioni '**`allow_other`**' e '**`default_permissions`**' sono più utili quando chi esegue l'innesto è proprio l'utente '**`root`**'.

### 19.11.3 Innesto e distacco

«

In condizioni normali, l'innesto di un file system per il sistema di FUSE, avviene attraverso l'uso diretto del programma frontale che lo gestisce. Di solito si usa uno schema simile a quello seguente:

```
xyz risorsa punto_di_innesto
```

Qui, ***xyz*** rappresenta il programma frontale ipotetico '**xyz**' (probabilmente '/usr/bin/xyz'). Per esempio, attraverso '**sshfs**', si potrebbe innestare la risorsa '**brot.dg:/home**' nella directory '/home/', con il comando seguente:

```
# sshfs brot.dg:/home /home [Invio]
```

Quando il contesto lo consente e quando il programma frontale è stato realizzato con cura, è possibile eseguire l'innesto in modo trasparente, lasciando a '**mount**' il compito di avviare il programma frontale appropriato:

```
mount -t fuse xyz#risorsa punto_di_innesto
```

L'unico svantaggio di questo approccio sta nella necessità di avere i privilegi dell'utente '**root**'. Ecco in che modo si trasforma l'esempio riferito a '**sshfs**':

```
# mount -t fuse sshfs#brot.dg:/home /home [Invio]
```

Il fatto di poter usare direttamente il comando '**mount**', consente anche di inserire tali informazioni nel file '/etc/fstab':

```
xyz#risorsa  punto_di_innesto  fuse  opzioni  0  0
```

Sempre riutilizzando l'esempio con '**sshfs**':

```
...
sshfs#brot.dg:/home     /home     fuse    defaults    0   0
...
```

Per staccare invece un file system gestito attraverso FUSE, di norma si usa '**fusermount**' con l'opzione '**-u**':

```
fusermount -u punto_di_innesto
```

Naturalmente si può usare anche '**umount**' nel modo consueto, ma in tal caso, di norma occorre avere i privilegi dell'utente '**root**':

```
umount punto_di_innesto
```

### 19.11.4 SSHfs

Attraverso il pacchetto SSHfs[49] è possibile accedere a un servizio SSH, come se fosse un file system «normale», tramite FUSE.

La lettura di questo capitolo richiede la conoscenza del funzionamento di OpenSSH (sezione 44.7), oltre che di nozioni elementari sull'uso delle reti TCP/IP (capitolo 32).

### 19.11.4.1 Utilizzo di SSHfs

«

Se FUSE è attivo e il pacchetto SSHfs è installato, si può innestare il file system di un altro elaboratore con un comando simile a questo:

```
$ sshfs tizio@brot.dg:/home/tizio ↩
↪       /home/tizio/home_remota [Invio]

tizio@brot.dg's password: digitazione_all'oscuro [Invio]
```

In questo caso ci si vuole collegare presso l'elaboratore *brot.dg*, in qualità di utente '**tizio**', per innestare la directory remota '`/home/tizio/`' all'interno della directory locale '`/home/tizio/home_remota`'. Ma la stessa cosa si potrebbe ottenere in modo più generalizzato così, a parte il fatto che in questo modo sono necessari i privilegi dell'utente '**root**':

```
# mount -t fuse sshfs#tizio@brot.dg:/home/tizio ↩
↪       /home/tizio/home_remota [Invio]

tizio@brot.dg's password: digitazione_all'oscuro [Invio]
```

Naturalmente si può predisporre anche il file '`/etc/fstab`' in modo da facilitare l'innesto di tale file system:

```
…
sshfs#tizio@brot.dg:/home/tizio /home/tizio/home_remota fuse defaults,user 0 0
…
```

Per staccare un file system SSHfs si può usare '**fusermount**', oppure il solito '**umount**'. Seguendo l'esempio già apparso, si possono usare, indifferentemente, i comandi seguenti:

```
$ fusermount -u /home/tizio/home_remota [Invio]
```

Oppure:

```
$ umount /home/tizio/home_remota [Invio]
```

### 19.11.4.2 Estendere le facoltà di accesso

Ammesso che si voglia agire in questo modo, per far sì che l'accesso al file system innestato avvenga secondo le modalità consuete, senza escludere gli altri utenti e considerando i permessi dei file, possono essere utili alcune opzioni della tabella 19.123. Pertanto, i comandi per l'innesto andrebbero cambiati secondo gli esempi seguenti:

```
$ sshfs tizio@brot.dg:/home/tizio /home/tizio/home_remota ↩
↪       -o allow_other,default_permissions [Invio]
```

Oppure:

```
# mount -o allow_other,default_permissions -t fuse ↩
↪       sshfs#tizio@brot.dg:/home/tizio ↩
↪       /home/tizio/home_remota [Invio]
```

Inoltre, il contenuto del file '`/etc/fstab`' potrebbe essere modificato nel modo seguente:

```
…
sshfs#tizio@brot.dg:/home/tizio /home/tizio/home_remota ↩
↪fuse defaults,user,allow_other,default_permissions 0 0
…
```

Ci sono comunque altre opzioni che possono essere importanti, soprattutto per ciò che riguarda la connessione SSH; si veda al riguardo la pagina di manuale *sshfs(1)*.

### 19.11.5 CurlFtpFS

«

Attraverso il pacchetto CurlFtpFS[50] è possibile accedere a un servizio FTP, come se fosse un file system «normale», tramite FUSE.

La lettura di questo capitolo richiede la conoscenza di ciò che è un servizio FTP per il trasferimento dei file attraverso la rete (capitolo 38), oltre che di nozioni elementari sull'uso delle reti TCP/IP (capitolo 32).

#### 19.11.5.1 Configurazione

«

Lo standard dei programmi clienti FTP prevede che si utilizzi il file '`~/.netrc`' per la configurazione, soprattutto allo scopo di accelerare il procedimento di accesso. Nel caso di CurlFtpFS è indispensabile la predisposizione di tale file, perché non è prevista la possibilità di fornire una parola d'ordine in modo interattivo.[51]

Il file '`~/.netrc`' contiene delle direttive molto semplici, a gruppi, introdotti dalla direttiva '**`machine`**', che definisce a quale collegamento ci si vuole riferire.

Tabella 19.128. Alcune direttive.

| Direttiva | Descrizione |
|---|---|
| `machine` ***nome*** | Il nome del nodo di rete a cui fanno riferimento le direttive successive. |
| `default` | Sostituisce la direttiva '**`machine`**' e rappresenta la configurazione predefinita per tutti i nodi di rete remoti non previsti in modo specifico all'interno di questo file. |

| Direttiva | Descrizione |
|---|---|
| `login` ***utente*** | Definisce il nominativo da utilizzare per il collegamento. |
| `password` ***stringa_parola_d'ordine*** | Definisce la parola d'ordine per l'accesso al sistema remoto. |

Per esempio, l'estratto seguente si riferisce agli accessi all'elaboratore *brot.dg*:

```
…
machine brot.dg
login daniele
password segretissima
…
```

In tal modo, l'utente che inserisce queste direttive nel proprio file '`~/.netrc`', chiarisce che l'accesso all'elaboratore *brot.dg* deve avvenire identificandosi come utente '**`daniele`**' e usando la parola d'ordine '**`segretissima`**'.

### 19.11.5.2 Utilizzo normale di CurlFtpFS

Se FUSE è attivo e il pacchetto CurlFtpFS è installato, si può innestare quanto viene offerto tramite un servizio FTP presso un altro elaboratore con un comando simile a questo:

```
$ curlftpfs ftp://brot.dg/aaa/bbb /home/tizio/bbb [Invio]
```

In questo caso, l'utente che esegue il comando accede al servizio FTP presso *brot.dg*, autenticandosi in base alla propria configurazione nel file '`~/.netrc`', innestando ciò che quel servizio gli offre a partire dal percorso '`/aaa/bbb/`', nella directory locale '`/home/tizio/bbb/`'.

In modo analogo, l'utente '**root**' potrebbe utilizzare direttamente il comando '**mount**':

```
# mount -t fuse curlftpfs#brot.dg:/ccc/ddd /mnt/ddd[Invio]
```

Si può predisporre anche il file '/etc/fstab' in modo da facilitare l'innesto di tale file system:

```
...
curlftpfs#brot.dg:/ccc/ddd  /mnt/ddd  fuse defaults,user 0 0
...
```

In tal caso, però, occorre osservare che, l'accesso viene eseguito in qualità di utente anonimo ('**anonymous**'), in quanto non viene presa in considerazione alcuna configurazione.

Per staccare un file system CurlFtpFS si può usare '**fusermount**', oppure il solito '**umount**'. Nel caso di innesto eseguito da un utente comune, si può usare solo '**fusermount**':

```
$ fusermount -u /home/tizio/bbb[Invio]
```

Quando invece si tratta dell'utente '**root**', si può utilizzare anche '**umount**':

```
# umount /mnt/ddd[Invio]
```

Naturalmente gli esempi mostrati fanno riferimento a quanto già apparso in precedenza.

### 19.11.5.3 Estendere le facoltà di accesso

«

Quando si vuole gestire l'accesso a un servizio FTP, tramite CurlFtpFS, ma configurando il file '/etc/fstab', rendendolo anche disponibile agli utenti comuni, è probabile che si voglia consentire

l'accesso ai dati innestati a tutti gli utenti. In tal caso torna utile l'opzione '**allow_other**' di FUSE (tabella 19.123):

```
…
curlftpfs#brot.dg:/ccc/ddd  /mnt/ddd  fuse defaults,user,allow_other 0 0
…
```

Naturalmente, con '**curlftpfs**' si può usare l'opzione '**-o allow_other**' nella riga di comando per ottenere lo stesso effetto:

```
$ curlftpfs -o allow_other ftp://brot.dg/aaa/bbb ↩
↪           /home/tizio/bbb [Invio]
```

### 19.11.5.4 Altre opzioni

CurlFtpFS consente di accedere a servizi FTP cifrati. Per queste cose occorre predisporre delle chiavi e utilizzare opzioni apposite. Si veda eventualmente *curlftpfs(1)*.

## 19.11.6 NTFSprogs e NTFS-3g

Attraverso il pacchetto NTFSprogs[52] è possibile creare e accedere a un file system NTFS. In particolare, con il programma '**ntfsmount**' che si avvale di FUSE, è possibile innestare un tale file system per utilizzarlo in modo quasi «normale».

Notoriamente l'accesso a un file system NTFS è problematico, tanto che di solito, le funzionalità disponibili in un kernel Linux sono limitate alla lettura ed eventualmente alla sovrascrittura di file già esistenti. Con '**ntfsmount**' si superano molti di questi problemi, anche se non sempre le operazioni di accesso richieste possono essere eseguite effettivamente; ma in ogni caso, l'impossibilità di eseguire una certa operazione, non mette in crisi il sistema operativo nel suo complesso e non comporta un danneggiamento del file system NTFS coinvolto.

Eventualmente, per il solo innesto si può utilizzare un altro applicativo derivato, che offre più possibilità di accesso: NTFS-3g[53], che si utilizza attraverso il programma '**ntfs-3g**', in modo abbastanza simile a '**ntfsmount**'.

### 19.11.6.1 Innesto di un file system NTFS

Se FUSE è attivo e il pacchetto NTFSprogs è installato, si può innestare un'unità di memorizzazione contenente un file system NTFS con un comando simile a questo:

```
$ ntfsmount /dev/sda1 /home/tizio/penna [Invio]
```

In alternativa, con NTFS-3g si può fare così:

```
$ ntfs-3g /dev/sda1 /home/tizio/penna [Invio]
```

In questo caso, l'utente che esegue il comando innesta il file system NTFS contenuto nell'unità corrispondente al file di dispositivo '`/dev/sda1`', a partire dalla directory '`/home/tizio/penna/`'.

In modo analogo, l'utente '**root**' potrebbe utilizzare direttamente il comando '**mount**'; in questo caso l'innesto viene fatto utilizzando la directory '`/mnt/sda1/`':

```
# mount -t fuse ntfsmount#/dev/sda1 /mnt/sda1 [Invio]
```

Oppure:

```
# mount -t fuse ntfs-3g#/dev/sda1 /mnt/sda1 [Invio]
```

Si può predisporre anche il file '`/etc/fstab`' in modo da facilitare l'innesto di tale file system:

```
...
ntfsmount#/dev/sda1 /mnt/sda1 fuse defaults,user,noauto 0 0
...
```

Oppure:

```
...
ntfs-3g#/dev/sda1  /mnt/sda1  fuse defaults,user,noauto 0 0
...
```

In tal caso, però, considerato che si vuole concedere l'innesto agli utenti comuni, potrebbe essere necessario offrire i permessi di scrittura a tutti gli utenti per l'accesso alla directory '`/mnt/sda1/`'.

Per staccare un file system NTFS che precedentemente è stato innestato con '**ntfsmount**' o '**ntfs-3g**', si utilizza normalmente '**fusermount**':

```
$ fusermount -u /home/tizio/penna [Invio]
```

Quando invece si tratta dell'utente '**root**', si può utilizzare anche '**umount**':

```
# umount /mnt/sda1 [Invio]
```

### 19.11.6.2 Opzioni

Se l'innesto del file system NTFS avviene senza indicare opzioni particolari, i file e le directory al suo interno appaiono con i soli permessi per l'accesso da parte dell'utente che lo ha eseguito. A questo proposito può essere utile inserire le opzioni '**fmask**' e '**dmask**' per togliere meno permessi rispetto a quanto avviene invece in modo predefinito.

Tabella 19.134. Alcune opzioni particolari per '**ntfsmount**'.

| Opzione | Descrizione |
|---|---|
| `uid=`***n***<br>`gid=`***n*** | Queste due opzioni consentono di specificare l'utente e il gruppo (attraverso i numeri UID e GID) a cui devono figurare appartenere tutti i file e le directory contenuti nel file system. |
| `fmask=`***n***<br>`dmask=`***n*** | Consentono di specificare la maschera dei permessi (da togliere), rispettivamente ai file e alle directory. Se queste opzioni non vengono usate, è come se venisse specificato '**fmask=0177**' e '**dmask=0077**', in modo da togliere tutti i permessi di accesso agli utenti che non sono il proprietario. |

Oltre a quanto descritto a proposito dei permessi virtuali di file e directory, occorre considerare che, in modo predefinito, sono attive le opzioni '**default_permissions**' e '**allow_other**'. Tuttavia, il fatto che i permessi predefiniti non concedano l'accesso agli altri utenti, implica che, salvo indicazione diversa, solo l'utente '**root**' può eventualmente accedere in tali condizioni.

A titolo di esempio viene mostrato un estratto del file '`/etc/fstab`', nel quale si dichiara la possibilità di innestare il file system NTFS contenuto nell'unità corrispondente al file di dispositivo '`/dev/hdc1`':

```
…
ntfsmount#/dev/hdc1   /win   fuse   ↩
↪defaults,user,noauto,uid=0,gid=0,fmask=0111,dmask=0   0  0
…
```

Oppure:

```
…
ntfs-3g#/dev/hdc1   /win   fuse   ↩
↪defaults,user,noauto,uid=0,gid=0,fmask=0111,dmask=0   0  0
…
```

In questo caso, pur consentendo agli utenti comuni di eseguire l'innesto, si preferisce attribuire la proprietà di tutto all'utente '**`root`**', lasciando però a chiunque la possibilità di modificare il contenuto dei dati (in quanto con l'opzione '**`fmask`**' si tolgono ai file solo i permessi di esecuzione, mentre con l'opzione '**`dmask`**' non si toglie alcunché).

Per ottenere un effetto simile attraverso la riga di comando, occorrerebbe agire così:

```
# ntfsmount /dev/hdc /win ↩
↪          -o uid=0,gid=0,fmask=0111,dmask=0 [Invio]
```

Oppure:

```
# ntfs-3g /dev/hdc /win ↩
↪        -o uid=0,gid=0,fmask=0111,dmask=0 [Invio]
```

### 19.11.7 EncFS

«

EncFS[54] è un file system cifrato, basato su FUSE, la cui gestione è molto semplice e alla portata di tutti: a partire da una certa directory si collocano i file, che sono cifrati nel loro contenuto e nel loro nome, mentre le altre informazioni rimangono intatte. Per accedere ai dati originali in chiaro, la directory contenente la gerarchia cifrata viene reinnestata in corrispondenza di un'altra directory vuota, fornendo una parola d'ordine.

#### 19.11.7.1 Un esempio per iniziare

«

L'utente '**tizio**' potrebbe decidere di utilizzare la directory '`~/dati_cifrati/`' per le proprie cose segrete. L'utente prepara tale directory, inizialmente vuota, quindi ne attiva la gestione attraverso il programma '**encfs**', ma per farlo deve reinnestarla contestualmente in un'altra directory vuota, per esempio '`~/dati_in_chiaro/`':

```
$ mkdir ~/dati_cifrati [Invio]

$ mkdir ~/dati_in_chiaro [Invio]

$ encfs ~/dati_cifrati ~/dati_in_chiaro [Invio]
```

Il programma vuole sapere che livello di sicurezza adottare, ma nella maggior parte delle situazioni è sufficiente quello predefinito:

```
Creating new encrypted volume.
Please choose from one of the following options:
 enter "x" for expert configuration mode,
 enter "p" for pre-configured paranoia mode,
 anything else, or an empty line will select standard mode.

?> [Invio]
```

```
Standard configuration selected.

Configuration finished.  The filesystem to be created has
the following properties:
Filesystem cipher: "ssl/blowfish", version 2:1:1
Filename encoding: "nameio/block", version 3:0:1
Key Size: 160 bits
Block Size: 512 bytes
Each file contains 8 byte header with unique IV data.
Filenames encoded using IV chaining mode.

Now you will need to enter a password for your filesystem.
You will need to remember this password, as there is
absolutely no recovery mechanism.  However, the password can
be changed later using encfsctl.
```

`New Encfs Password:` ***digitazione_all'oscuro*** [*Invio*]

`Verify Encfs Password:` ***digitazione_all'oscuro*** [*Invio*]

`$` **`mount`** [*Invio*]

```
…
encfs on /home/tizio/dati_in_chiaro type fuse ↩
↪(rw,nosuid,nodev,default_permissions,user=tizio)
…
```

Da questo momento, tutto quello che l'utente '**`tizio`**' fa nella directory '`~/dati_in_chiaro/`', si traduce in altrettanti file e directory cifrati nella directory '`~/dati_cifrati/`'.

Per concludere l'attività nei dati cifrati, si può staccare l'innesto che parte dalla directory '`~/dati_in_chiaro:`':

`$` **`fusermount -u ~/dati_in_chiaro`** [*Invio*]

La directory dedicata al contenimento dei dati cifrati, si distingue in quanto contiene il file '`.encfs`***n***'; pertanto, quando si utilizza nuovamente il programma '**encfs**' per innestarla altrove, il contenuto viene riconosciuto e reso disponibile in chiaro. Naturalmente, non è necessario che la directory cifrata venga innestata sempre nello stesso posto:

```
$ mkdir /tmp/mia [Invio]

$ encfs ~/dati_cifrati /tmp/mia [Invio]

EncFS Password: digitazione_all'oscuro [Invio]

$ mount [Invio]

  ...
  encfs on /tmp/mia type fuse ↩
↪(rw,nosuid,nodev,default_permissions,user=tizio)
  ...
```

È importante osservare che le directory vanno indicate al programma '**encfs**' con un percorso assoluto. Negli esempi è stata usata la tilde ('**~**') per fare riferimento a un percorso assoluto della directory personale dell'utente.

Benché possa essere considerato di scarsa utilità, è bene sapere che una stessa directory cifrata può essere innestata in chiaro su più directory differenti. Tali directory diventano equivalenti per ciò che riguarda il contenuto.

### 19.11.7.2 Cura nell'uso di EncFS

Dopo la creazione di qualche file nella directory cifrata (attraverso l'innesto in chiaro), si può dare un'occhiata al contenuto, facendo magari un confronto tra dati reali e dati cifrati:

```
$ encfs ~/dati_cifrati ~/dati_in_chiaro[Invio]

$ tree -a ~/dati_in_chiaro[Invio]

  /home/tizio/dati_in_chiaro
  |-- alfa
  |   `-- charlie
  `-- bravo

  1 directory, 2 files
$ tree -a ~/dati_cifrati[Invio]

  /home/tizio/dati_cifrati
  |-- .encfs5
  |-- N6rnhFLNx,cgB-
  |   `-- dS,q8YPh64-h8,
  `-- nuW8KJbRPh-Yn1

  1 directory, 3 files
```

Se nella directory cifrata si inseriscono direttamente dei file (presumibilmente in chiaro), questi non vengono considerati; inoltre, è importante capire che se si cancella il file '`.encfs`***n***', si perde la possibilità di accedere ai dati.

Il contenuto della directory cifrata può essere trasferito o copiato, tale e quale, mantenendo la possibilità di accedervi poi attraverso il

meccanismo del reinnesto in chiaro. In tal modo è possibile fare una copia di sicurezza dei dati, che rimangono cifrati nella copia.

### 19.11.7.3 Manutenzione di EncFS

Il programma '**encfsctl**' consente di ispezionare una directory cifrata e di cambiarne la parola d'ordine (purché si conosca quella precedente). Supponendo che la directory cifrata sia sempre '`~/dati_cifrati/`', ci si potrebbe chiedere di conoscerne le caratteristiche:

```
$ encfsctl info ~/dati_cifrati [Invio]

  Version 5 configuration; created by EncFS 1.2.5
   (revision 20040813)
  Filesystem cipher: "ssl/blowfish", version 2:1:1
  Filename encoding: "nameio/block", version 3:0:1
  Key Size: 160 bits
  Block Size: 512 bytes
  Each file contains 8 byte header with unique IV data.
  Filenames encoded using IV chaining mode.
```

Poi ci si potrebbe domandare se, per caso, ci sono file estranei, nel senso che non vengono riconosciuto dal sistema di cifratura:

```
$ encfsctl showcruft ~/dati_cifrati [Invio]
```

In tal caso diventa necessario l'inserimento della parola d'ordine:

```
EncFS Password: digitazione_all'oscuro [Invio]

  11:37:04 (FileUtils.cpp:1296) useStdin: 0
  11:37:05 (FileUtils.cpp:1307) configuration key size = 32
  11:37:05 (FileUtils.cpp:1308) cipher key size = 32
  Found 0 invalid files.
```

Quindi, come possibilità più importante, potrebbe essere necessario cambiare la parola d'ordine per l'accesso ai dati:

```
$ encfsctl passwd ~/dati_cifrati [Invio]

  Enter current Encfs password
EncFS Password: digitazione_all'oscuro [Invio]

  Enter new Encfs password
New EncFS Password: digitazione_all'oscuro [Invio]
Verify EncFS Password: digitazione_all'oscuro [Invio]

  Volume Key successfully updated.
```

È interessante osservare che la parola d'ordine può essere cambiata anche mentre la directory è già innestata in chiaro da qualche parte, senza interferire con le attività normali di accesso. Ciò che cambia è solo il fatto che all'innesto successivo, deve essere fornita la parola d'ordine nuova.

### 19.11.7.4 Opzioni per facilitare l'accesso ai dati

Per consentire l'accesso ai dati in chiaro anche agli altri utenti, si usano notoriamente le opzioni '**allow_other**' e '**default_permissions**'. Con il programma '**encfs**' è possibile usare l'opzione '**--public**' per richiamare automaticamente queste opzioni di FUSE, ma in tal caso, '**encfs**' pretende che l'azione sia richiesta dall'utente '**root**':

```
$ encfs --public ~/dati_cifrati ~/dati_in_chiaro [Invio]

  … option '--public' ignored for non-root user
```

Tuttavia, se la configurazione di FUSE lo consente, questa funzionalità può essere ottenuta ugualmente, con le limitazioni dovute al fatto che non è l'utente '**root**' a eseguire l'innesto:

```
$ encfs ~/dati_cifrati ~/dati_in_chiaro -- ↩
↪       -o allow_other,default_permissions[Invio]
```

## 19.12 Gerarchia del file system

«

La struttura dei file system di ogni sistema operativo Unix è diversa da quella degli altri. Spesso, per mantenere la compatibilità con altri ambienti si utilizzano dei collegamenti simbolici. Con essi si può simulare la presenza di directory e file che in realtà non esistono dove si vuole fare sembrare che siano. La tecnica dell'uso di collegamenti simbolici può essere usata anche per personalizzare la struttura del proprio file system, facendo in modo però che i programmi normali continuino a trovare quello che serve loro, dove si aspettano che sia.

Quando si organizza un file system è importante distinguere tra diversi tipi di file:

- statici o variabili;
- condivisibili o non condivisibili;
- indispensabili per l'avvio del sistema o meno.

Ciò che è statico può essere reso accessibile in sola lettura (esecuzione compresa), mentre il resto deve essere accessibile necessariamente anche in scrittura. Ciò che è condivisibile può essere utilizzato da più elaboratori contemporaneamente, il resto no. Ciò che è indispensabile per l'avvio dell'elaboratore, non può, o comunque

non dovrebbe essere collocato in file system remoti. Purtroppo non è detto che la distinzione sia sempre netta.

Nelle sezioni seguenti viene descritta la struttura essenziale (la gerarchia) di un file system standard, secondo il documento FHS (*Filesystem hierarchy standard*), a cui dovrebbero adeguarsi i sistemi GNU. Per maggiori dettagli e aggiornamenti si deve consultare l'originale all'indirizzo *http://www.linuxfoundation.org/collaborate/workgroups/lsb/fhs* .

La struttura tradizionale di un sistema Unix, ancora molto presente nello standard FHS, è controversa, soprattutto in merito alla suddivisione dei file eseguibili in '`/bin/`' e '`/sbin/`' (oltre che in '`/usr/bin/`' e '`/usr/sbin/`'). Pertanto è ragionevole attendersi che con il tempo questa struttura venga semplificata.

### 19.12.1 Directory «/»: la radice

La directory radice è quella che contiene tutte le altre. Di solito contiene solo directory con l'unica eccezione del file del kernel che può risiedere qui o in '`/boot/`'. La struttura che si dirama dalla directory radice può essere riassunta dall'elenco seguente:

| Directory | Descrizione |
|---|---|
| `/bin/` | binari essenziali; |
| `/boot/` | file statici per l'avvio del sistema; |
| `/dev/` | file di dispositivo; |
| `/etc/` | configurazione particolare del sistema; |

| Directory | Descrizione |
|---|---|
| `/home/` | directory personali degli utenti; |
| `/lib/` | librerie essenziali e moduli del kernel; |
| `/mnt/` | punto di innesto temporaneo; |
| `/opt/` | applicativi aggiuntivi; |
| `/root/` | directory personale dell'utente '**root**'; |
| `/tmp/` | file e directory temporanei; |
| `/usr/` | gerarchia secondaria; |
| `/var/` | dati variabili. |

Nel caso particolare dei sistemi GNU/Linux si devono aggiungere le directory '`/proc/`' e '`/sys/`' che contengono informazioni vitali prodotte dal kernel.

### 19.12.2 Directory «/bin/» e «/sbin/»: binari essenziali

La directory '`/bin/`' contiene gli eseguibili di uso comune più importanti. I file al suo interno sono generalmente accessibili in esecuzione a tutti gli utenti. La directory '`/sbin/`' contiene eseguibili allo stesso livello di importanza di '`/bin/`', ma il cui utilizzo è generalmente di competenza dall'utente '**root**'.

La distinzione non è dovuta tanto a motivi di sicurezza, quanto all'esigenza di mettere un po' di ordine tra gli eseguibili. Infatti, i file contenuti in '`/sbin/`' sono generalmente accessibili anche agli

utenti comuni (purché i permessi di questi file non siano stati modificati per esigenze particolari), ma questa directory non viene inclusa nell'elenco dei percorsi degli eseguibili (variabile ***PATH***) degli utenti.

La directory '`/bin/`', in particolare, dovrebbe contenere una shell compatibile con quella di Bourne e una compatibile con la shell C.

### 19.12.3 Directory «/boot/»: file statici per l'avvio del sistema

La directory '`/boot/`' contiene i file utilizzati dal meccanismo di caricamento del sistema operativo (*boot*). In particolare può contenere il kernel quando questo non si trova nella directory radice.

Negli elaboratori x86 è generalmente necessario che i file contenuti in questa directory, kernel e disco RAM iniziale inclusi, siano collocati entro il 1024-esimo cilindro. Inoltre, il sistema di avvio può richiedere di risiedere in un file system diverso da quello usato per il sistema operativo. Pertanto, è opportuno che il contenuto della directory '`/boot/`' risieda effettivamente in una partizione iniziale dell'unità di memorizzazione, dedicata allo scopo.

### 19.12.4 Directory «/dev/»: file di dispositivo

La directory '`/dev/`' contiene una lunga serie di file di dispositivo. Perché i vari componenti fisici dell'elaboratore possano funzionare, occorre che per ognuno di essi sia stato previsto il file di dispositivo relativo, in questa directory. Spesso, quando si vuole utilizzare un nome predefinito per un dispositivo, si utilizza un collegamento simbolico che punta a quello che serve effettivamente.

Regolando opportunamente i permessi di questi file si controlla l'utilizzo diretto delle unità fisiche da parte degli utenti.

All'interno di questa directory è contenuto il programma '**MAKEDEV**' (di solito si tratta di uno script) utile per ricreare o aggiungere eventuali file di dispositivo mancanti, rispettando le convenzioni del sistema che si utilizza; inoltre, eventualmente, può contenere anche il programma '**MAKEDEV.local**' per la creazione di file di dispositivo particolari, per uso locale.

### 19.12.5 Directory «/etc/»: configurazione particolare del sistema

«

La directory '/etc/' contiene una lunga serie di file di configurazione, raccolti eventualmente anche in sottodirectory, che riguardano l'intero sistema e che non possono essere condivisi con altri. Alcune di queste sottodirectory sono molto importanti, in particolare quelle descritte nell'elenco seguente:

| Directory | Descrizione |
|---|---|
| /etc/X11/ | contiene la configurazione per il sistema grafico X, assieme a quella dei gestori di finestre; |
| /etc/opt/*applicativo*/ | contiene la configurazione specifica di programmi inseriti all'interno della gerarchia '/opt/'; |
| /etc/sgml/ | contiene la configurazione di SGML; |
| /etc/xml/ | contiene la configurazione di XML. |

Benché non sia riconosciuta nello standard FHS, nei sistemi GNU/Linux è molto importante anche la directory '/etc/skel/', per contenere i file di configurazione personale predefiniti per i nuovi utenti.

### 19.12.6 Directory «/home/»: directory personali degli utenti

La directory ‘/home/’ è normalmente il punto di partenza per tutte le directory personali degli utenti. Se il sistema viene utilizzato da molti utenti, può essere conveniente (e a volte addirittura necessario) dirottare il contenuto di questa directory in un altro disco e di conseguenza in un file system secondario innestato in questo punto.

### 19.12.7 Directory «/lib/»: librerie condivise essenziali e moduli del kernel

La directory ‘/lib/’ è il contenitore dei file di libreria (*library*) necessari per i programmi di uso generale. Devono trovarsi qui le librerie necessarie agli eseguibili che possono trovarsi in ‘/bin/’ e ‘/sbin/’. Le librerie che riguardano solo programmi collocati al di sotto di ‘/usr/’, non appartengono a questa directory.

Assieme ai file di libreria, potrebbe trovarsi una directory che si articola ulteriormente e contiene i moduli del kernel: ‘modules/’.

### 19.12.8 Directory «/mnt/»: punto di innesto per l’inserzione temporanea di altri file system

La directory ‘/mnt/’ normalmente è vuota e serve come punto di collegamento generico per un altro file system.

Spesso, la directory ‘/mnt/’ è stata usata con altre sottodirectory, specifiche per il tipo di unità da innestare; per esempio: ‘/mnt/cdrom/’, ‘/mnt/floppy/’ e simili. Recentemente, per questo scopo, è stata introdotta la directory ‘/media/’.

### 19.12.9 Directory «/opt/»: applicativi aggiunti

«

La directory ‘`/opt/`’ è il punto di partenza per l’installazione di applicativi addizionali. Tali applicativi dovrebbero risultare collocati ognuno in una propria sottodirectory, nella forma ‘`/opt/`***applicativo***`/`’, ma in particolare dovrebbero contenere almeno la directory ‘`bin/`’ (‘`/opt/`***applicativo***`/bin/`’) ed eventualmente anche ‘`man/`’ (‘`/opt/`***applicativo***`/man/`’).

Quanto contenuto a partire dalla directory ‘`/opt/`’ deve essere statico e quindi accessibile in sola lettura, per cui, i file variabili di questi applicativi devono trovarsi all’interno di ‘`/var/opt/`***applicativo***`/`’ e i file di configurazione in ‘`/etc/opt/`***applicativo***`/`’.

### 19.12.10 Directory «/proc/» e «/sys/»: informazioni vitali sul kernel e sui processi

«

Le directory ‘`/proc/`’ e ‘`/sys/`’ sono vuote e utilizzate per innestare i file system omonimi nei sistemi GNU/Linux. I file (e le directory) contenuti in questi file system virtuali sono indispensabili ai programmi che hanno la necessità di accedere alle informazioni sul sistema.

Quando si esegue una copia di sicurezza di tutto il file system, queste directory non devono essere archiviate; infatti, è sufficiente ricrearle vuote al momento del recupero, con i soli permessi di lettura e attraversamento: $0555_8$.

Tuttavia va osservato che nel documento FHS, a cui si riferisce questo capitolo, non viene menzionata la directory ‘`/sys/`’, ma si tratta ugualmente di una directory obbligatoria nei sistemi GNU/Linux.

### 19.12.11 Directory «/root/»: directory personale dell'utente root

La directory '`/root/`' è la directory personale dell'utente '**`root`**'. Ci sono molti validi motivi per evitare di mescolarla insieme a quelle degli utenti comuni. Vale la pena di tenere presente che così facendo è possibile impedire gli accessi più facilmente. Inoltre è opportuno che questa directory sia collocata nel file system principale, proprio perché l'amministratore deve essere in grado di accedere anche quando il sistema viene avviato in situazioni di emergenza e non si possono innestare altri file system. Comunque, questa collocazione è considerata facoltativa.

### 19.12.12 Directory «/tmp/»: file temporanei

La directory '`/tmp/`' è destinata a contenere file provvisori e potrebbe essere anche collocata in un'unità virtuale basata su memoria volatile (disco RAM).

Non sempre i programmi che creano dei file provvisori in questa directory, provvedono poi anche alla loro eliminazione. Se la directory è stata collocata in un'unità di memorizzazione normale, di tanto in tanto, conviene darci un'occhiata e poi procedere a eliminare tutto quello che non serve.

Volendo (ammesso che ciò non sia già stato fatto da chi ha organizzato la distribuzione del sistema operativo) è possibile anche inserire in uno script, di quelli utilizzati dalla procedura di inizializzazione del sistema, un'istruzione di eliminazione di tutti i file contenuti in questa directory, in modo che a ogni avvio del sistema, questa venga ripulita.

Data la sua natura, quando si fanno delle copie di sicurezza del file system, non è il caso di copiare il contenuto di questa directory.

I permessi dati a questa directory sono importanti: devono consentire a chiunque di accedervi in ogni modo e dovrebbero evitare che un utente possa cancellare (inavvertitamente) file di altri utenti. Per questo si attribuiscono normalmente i permessi $1777_8$, ovvero '**rwxrwxrwt**'.

### 19.12.13 Directory «/usr/»: gerarchia secondaria (dati statici e condivisibili)

«

La directory '/usr/' è importante e si scompone in una struttura molto articolata. La gerarchia che parte da questo punto è organizzata in modo da essere statica e condivisibile.

In linea di principio, gli applicativi non devono essere collocati all'interno di questa gerarchia in una directory specifica, ma dovrebbero distribuirsi nel sistema, insieme agli altri. Infatti, l'alternativa corretta è l'utilizzo della gerarchia '/opt/' creata appositamente per permettere questo tipo di collocazione degli applicativi. L'ambiente grafico X che potrebbe utilizzare una propria directory discendente da '/usr/', fa eccezione.

### 19.12.14 Directory «/usr/bin/» e «/usr/sbin/»: binari non essenziali

«

La directory '/usr/bin/' contiene gli eseguibili di uso comune meno importanti. Generalmente, i file al suo interno sono accessibili in esecuzione a tutti gli utenti. La directory '/usr/sbin/' contiene eseguibili non indispensabili, il cui utilizzo dovrebbe essere di competenza dell'utente '**root**'.

Valgono le stesse considerazioni relative alla distinzione che c'è tra le directory '`/bin/`' e '`/sbin/`'. È opportuno ribadire che quanto contenuto in '`/bin/`' e '`/sbin/`' è essenziale per l'avvio del sistema in situazioni di emergenza e per gestire funzionalità di rete minime necessarie a innestare eventuali file system remoti. Tutto il resto, compresi i demoni per la gestione di servizi non essenziali, deve essere collocato in '`/usr/bin/`' e '`/usr/sbin/`'.

All'interno di '`/usr/bin/`' dovrebbero trovarsi alcune shell utilizzate normalmente per la programmazione (e non quindi per l'interazione con l'utente). In pratica potrebbe trattarsi di '`/usr/bin/perl`', '`/usr/bin/python`' e '`/usr/bin/tcl`'. Se per qualche motivo non possono trovarsi in questa directory, è almeno opportuno che si predisponga un collegamento simbolico che permetta di avviarle da questo punto. Ciò è necessario per poter realizzare script che possano funzionare in ogni configurazione, dal momento che all'inizio dello script occorre indicare il percorso assoluto dell'interprete.

```
#!/usr/bin/perl
…
```

### 19.12.15 Directory «/usr/games/»: giochi e programmi didattici

La directory '`/usr/games/`' serve per contenere programmi meno importanti destinati al passatempo o alla didattica.

I file di dati statici di questi dovrebbero collocarsi in '`/usr/share/games/`', mentre quelli che devono essere modificati (come lo storico dei punteggi raggiunti e cose simili) in '`/var/games/`'.

### 19.12.16 Directory «/usr/include/»: file di intestazione

«

Raccoglie i file *include*, o file di intestazione, cioè quelli utilizzati come segmenti standard di sorgenti per i programmi. In pratica, sono quei file che di solito terminano con un'estensione '`.h`' e vengono inglobati automaticamente in un sorgente attraverso le istruzioni '**`#INCLUDE`** ***file***'.

Non tutti i file di questo tipo sono inseriti direttamente nella directory '`/usr/include/`' o in una sua discendente, ma in un sistema ordinato, tutti i file *include* sono raggiungibili a partire da questo punto, almeno attraverso collegamenti simbolici.

### 19.12.17 Directory «/usr/lib/»: librerie per la programmazione e per gli applicativi

«

La directory '`/usr/lib/`' contiene i file di libreria necessari per i programmi installati a partire da '`/usr/`'. Il concetto di libreria, viene qui inteso in un senso più ampio di quello utilizzato da '`/lib/`'. Infatti, oltre ai file di libreria veri e propri si possono trovare altri file statici semplicemente accessori agli eseguibili.

Per la precisione, i file contenuti al di sotto di questa posizione, sono considerati come dipendenti dal tipo di architettura, mentre quelli che non dipendono da questa vanno collocati in '`/usr/share/`'.

### 19.12.18 Directory «/usr/local/»: programmi locali

«

La directory '`/usr/local/`' è il punto di inizio per l'installazione locale di programmi, senza che questi siano interessati dalle procedure di aggiornamento del software installato nel modo normale.

Questa valenza locale dipende dai punti di vista e dalle esigenze. ‘/usr/local/’ potrebbe essere usata come directory di collegamento per un altro file system specifico per l’ambito locale. In pratica, quanto contenuto in ‘/usr/’ potrebbe essere condiviso da diversi elaboratori, mentre ‘/usr/local/’ potrebbe essere la particolarità di ogni elaboratore, o di un gruppo più piccolo.

In generale, questa directory dovrebbe apparire vuota subito dopo l’installazione di GNU/Linux. Al massimo potrebbe contenere le directory in cui può scomporsi (anche queste vuote). La struttura prevista di ‘/usr/local/’ è la seguente:

- ‘bin/’,
- ‘games/’,
- ‘include/’,
- ‘lib/’,
- ‘sbin/’,
- ‘share/’,
- ‘src/’.

Il significato e l’utilizzo delle directory appena elencate è equivalente a quelle omonime discendenti da ‘/usr/’, solo che qui hanno un valore relativo a ciò che si installa localmente.

### 19.12.19 Directory «/usr/share/»: dati indipendenti dall’architettura

La directory ‘/usr/share/’ serve a contenere file di dati statici indipendenti dall’architettura. Ciò rende questa directory condivisibile

tra più sistemi operativi, **dello stesso tipo e versione**, installati su piattaforme differenti. Le directory più importanti di questa struttura sono descritte brevemente nell'elenco seguente:

| Directory | Descrizione |
|---|---|
| `/usr/share/dict/` | elenchi di parole; |
| `/usr/share/doc/` | documenti vari; |
| `/usr/share/games/` | file di dati statici per quanto installato in '`/usr/games/`'; |
| `/usr/share/info/` | documentazione ipertestuale GNU Info; |
| `/usr/share/locale/` | informazioni sulle varie localizzazioni; |
| `/usr/share/man/` | documentazione interna standard; |
| `/usr/share/sgml/` | dati SGML; |
| `/usr/share/xml/` | dati XML; |
| `/usr/share/nls/` | *Native language support* |
| `/usr/share/misc/` | varie; |
| `/usr/share/terminfo/` | directory del sistema Terminfo per la configurazione dei terminali; |
| `/usr/share/tmac/` | macro Troff aggiuntive e opzionali; |
| `/usr/share/zoneinfo/` | informazioni sull'ora locale. |

Oltre a queste directory, potrebbero esserne aggiunte altre, specifiche di particolari applicazioni o gruppi di queste.

### 19.12.20 Directory «/usr/share/man/»: pagine di manuale

La directory ‘`/usr/share/man/`’ contiene i file delle pagine di manuale, ovvero la documentazione interna leggibile attraverso il programma ‘**`man`**’. La directory si suddivide in una struttura che varia a seconda della localizzazione, come descritto nella sezione 5.2.1.

Questa non è l’unica posizione in cui si collocano i file delle pagine di manuale, ma questi file riguardano il sistema in generale, i programmi collocati a partire dalla directory radice e da ‘`/usr/`’. Sono esclusi i file riferiti alla documentazione di programmi installati localmente che si trovano in ‘`/usr/local/man/`’. Infine, le pagine di manuale specifiche degli applicativi aggiunti dovrebbero trovarsi in ‘`/opt/`***`applicativo`***`/man/`’.

### 19.12.21 Directory «/usr/share/misc/»: file di dati vari

La directory ‘`/usr/share/misc/`’ è destinata a contenere file di dati statici di uso vario. In particolare, si dovrebbe trovare qui il file ‘`magic`’.

### 19.12.22 Directory «/usr/src/»: sorgenti

È il punto a partire dal quale conviene collocare i sorgenti dei programmi che si vogliono tenere a disposizione. I sorgenti che riguardano i programmi collocati in ‘`/usr/local/`’ vanno inseriti a partire da ‘`/usr/local/src/`’.

### 19.12.23 Directory «/var/»: dati variabili

«

La directory ‘/var/’ contiene altre directory e file di uso vario che contengono dati variabili. Questo significa anche che qui c’è un po’ di tutto, ma si tratta di tutto quello che non può essere contenuto in ‘/usr/’ perché tale directory deve poter essere accessibile in sola lettura. Nelle sezioni successive vengono elencate solo alcune delle directory che si diramano da ‘/var/’.

### 19.12.24 Directory «/var/cache/»: directory per la memorizzazione transitoria

«

La directory ‘/var/cache/’ serve a contenere dati provvisori provenienti dalle applicazioni. Questi dati devono poter essere rigenerati dalle applicazioni in caso di necessità e ciò deve consentire la cancellazione manuale di tali dati senza provocare pregiudizio alle applicazioni. Così, tutto ciò che risulta contenuto a partire da questa directory non ha la necessità di essere salvato nelle procedure per le copie di sicurezza. Le directory più importanti che riguardano questa struttura sono descritte brevemente nell’elenco seguente:

| Directory | Descrizione |
|---|---|
| /usr/share/fonts/ | caratteri tipografici generati localmente; |
| /var/cache/man/ | pagine di manuale formattate; |
| /var/cache/www/ | proxy WWW o dati provvisori; |
| /var/cache/***applicativo***/ | dati provvisori specifici di un programma determinato. |

### 19.12.25 Directory «/var/lock/»: file per il controllo dell'accesso alle risorse

I file che servono a indicare che una certa risorsa è impegnata, dovrebbero essere collocati tutti in '`/var/lock/`'. Ogni file contenuto in questa directory dovrebbe avere il prefisso '`LCK..`' e terminare con il nome del dispositivo (senza il prefisso '`/dev/`'). All'interno del file dovrebbe trovarsi il numero PID del processo che impegna il dispositivo.

### 19.12.26 Directory «/var/log/»: file delle registrazioni

La directory '`/var/log/`' contiene i file delle registrazioni: sia quelli utilizzati dal registro del sistema, sia quelli di altri programmi.

### 19.12.27 Directory «/var/mail/»: caselle postali degli utenti

La directory '`/var/mail/`' viene usata per contenere i file delle caselle postali degli utenti, quando queste non sono distribuite nelle rispettive directory personali.

La collocazione più logica per questa directory sarebbe '`/var/spool/mail/`', ma gli standard di fatto del passato nei sistemi Unix impongono questa scelta, allo scopo di garantire l'interscambio dei dati anche tra sistemi Unix molto diversi. Per lo stesso motivo, i file che appaiono in questa directory devono avere un nome corrispondente a quello dell'utenza a cui si riferiscono e il formato deve essere quello comune per la posta elettronica (*Unix mailbox*).

### 19.12.28 Director «/var/opt/»: dati variabili per gli applicativi aggiuntivi

«

La directory ‘`/var/opt/`’ è il punto di partenza per altre directory contenenti i dati variabili degli applicativi aggiuntivi installati in ‘`/opt/`’. Per la precisione, ogni applicativo che necessita di modificare dati dovrebbe utilizzare una directory con il suo stesso nome.

```
/var/opt/applicativo/
```

### 19.12.29 Directory «/var/run/»: dati variabili di esecuzione

«

La directory ‘`/var/run/`’ contiene informazioni che riguardano l’esecuzione dei processi. Si tratta in particolare di informazioni sul numero PID degli eseguibili in funzione, del file ‘`utmp`’, dal quale si conosce quali sono gli utenti connessi attualmente, assieme ad altri dati transitori.

Per quanto riguarda l’informazione sul numero PID dei processi, questi sono contenuti in file il cui nome utilizza il formato seguente:

```
programma.pid
```

Tutto quanto contenuto in questa directory deve essere cancellato all’avvio del sistema.

### 19.12.30 Directory «/var/spool/»: code di dati

La directory ‘`/var/spool/`’ è molto importante per tutti i programmi che hanno la necessità di gestire code di elaborazioni. Per esempio, sono collocate sotto questa directory le code di stampa, dei messaggi di posta elettronica inviati e di altri gestori di servizi.

### 19.12.31 Directory «/var/tmp/»: file temporanei preservati all’avvio del sistema

La directory ‘`/var/tmp/`’ è destinata a contenere file temporanei che devono rimanere a disposizione più a lungo rispetto a quanto si fa con ‘`/tmp/`’. In particolare, il suo contenuto non dovrebbe essere cancellato al riavvio del sistema.

## 19.13 Approfondimenti

### 19.13.1 Inizializzazione a basso livello dei dischetti

Le unità a dischetti sono scomparse dagli elaboratori attuali, pertanto lo studio dei programmi specifici per l’inizializzazione a basso livello di tali supporti magnetici, si giustifica solo per il mantenimento di apparecchiature molto vecchie.

#### 19.13.1.1 Utilizzo di «fdformat»

Il programma ‘**`fdformat`**’[55] esegue un’inizializzazione a basso livello di un dischetto. Il nome del file di dispositivo indica l’unità a dischetti in cui si vuole compiere l’operazione e anche il formato che si vuole ottenere. Per questo motivo è meglio evitare di utilizzare semplicemente nomi di dispositivo generici come ‘`/dev/fd0`’ e ‘`/dev/`

`fd1`'. Molto probabilmente si utilizzano maggiormente i formati relativi a '`/dev/fd0u1440`' e '`/dev/fd1u1440`' che si riferiscono al formato da 1440 Kibyte dei dischetti da 3,5 pollici.

Vale la pena di ricordare che i nomi di dispositivo relativi ai dischetti possono cambiare leggermente da una distribuzione GNU/Linux a un'altra. A volte, il formato dei dischetti da 1440 Kibyte corrisponde al file '`/dev/fd0H1440`'.

`fdformat` [`-n`] ***dispositivo***

L'opzione '**`-n`**' serve a saltare la fase di controllo successiva all'inizializzazione: in generale è meglio non utilizzarla in modo da verificare la riuscita dell'inizializzazione.

Se si vuole consentire agli utenti comuni di compiere questa operazione occorre regolare i permessi dei file di dispositivo dei dischetti in modo da permettere loro l'accesso in lettura e scrittura.

Segue la descrizione di alcuni esempi.

- `#` **`fdformat /dev/fd0u1440`** [*Invio*]

  Inizializza un dischetto da 1440 Kibyte nella prima unità a dischetti.

- `#` **`fdformat /dev/fd1u1440`** [*Invio*]

  Inizializza un dischetto da 1440 Kibyte nella seconda unità a dischetti.

### 19.13.1.2 Utilizzo di «superformat»

Il programma '**superformat**'[56] è alternativo a '**fdformat**', rispetto al quale permette di definire molti dettagli in più che riguardano l'inizializzazione dei dischetti. In generale, si possono ignorare tutte queste caratteristiche speciali, utilizzando '**superformat**' con la sola indicazione del file di dispositivo del dischetto da inizializzare: è sufficiente fare riferimento al dispositivo generico, senza le informazioni sulla capacità dello stesso. Alla fine dell'inizializzazione a basso livello, '**superformat**' utilizza '**mformat**' per inserire nel dischetto un file system Dos-FAT, che se non serve può essere semplicemente ignorato.

`superformat` [*opzioni*] *dispositivo* [*descrizione_del_supporto*]

Prima di eseguire l'inizializzazione, '**superformat**' controlla le caratteristiche dell'unità a dischetti. È possibile predisporre il file '`/etc/driveprm`' con una direttiva che viene suggerita dallo stesso programma mentre è in funzione, per evitare che venga ripetuto questo controllo. Se si interviene in questo modo, occorre ricordare di eliminare la direttiva quando si cambia unità a dischetti, o quando si cambia l'unità di controllo. In generale, non è il caso di preoccuparsi di questo file, a meno che l'inizializzazione dei dischetti sia un'attività frequente.

Questo valore potrebbe essere influenzato anche dalle variazioni climatiche, pertanto è meglio non predisporre il file '`/etc/driveprm`', nonostante la perdita di tempo iniziale.

Le opzioni di '**superformat**' sono utili soprattutto quando si vuole inizializzare un dischetto utilizzando un formato insolito, ma in tal caso conviene leggere la pagina di manuale relativa: *superformat(1)*. Comunque, vale la pena di ricordare che con l'opzione '**-f**', o '**--noverify**', si esclude qualunque controllo sul risultato dell'inizializzazione.

L'esempio seguente inizializza un dischetto nell'unità corrispondente al file di dispositivo '`/dev/fd0`', utilizzando il formato standard massimo per quel tipo di unità:

```
# superformat /dev/fd0 [Invio]
```

### 19.13.2 Dischi senza partizioni

«

Come i dischetti, anche le unità di memorizzazione di dimensioni più grandi possono essere usate senza partizioni, facendo riferimento al file di dispositivo che rappresenta l'unità intera, come nell'esempio seguente:

```
# mkfs.ext3 /dev/sdc [Invio]
```

Come si vede, si intende inizializzare il dispositivo '`/dev/sdc`'. Inizialmente, il programma che si utilizza dovrebbe avvisare della scelta particolare che si sta compiendo:

```
/dev/sdc is entire device, not just one partition!
Proceed anyway? (y,n)
```

Evidentemente, basta confermare premendo la lettera '**y**', seguita da [*Invio*] per ottenere ciò che si desidera.

In generale, i programmi a disposizione per la suddivisione delle unità di memorizzazione in partizioni, partono dal presupposto che queste siano organizzate in settori da 512 byte. Tuttavia, possono esistere unità con settori di dimensioni multiple, come è accaduto in passato con alcuni tipi di dischi magneto-ottici.

GNU/Linux è in grado di gestire unità di memorizzazione con settori più grandi di 512 byte (purché si tratti di multipli e si resti entro i 4096 byte), ma le partizioni che si usano solitamente, secondo lo standard tradizionale dei PC-IBM, non possono essere applicate a unità con settori più grandi. Per questo motivo, l'unico modo pratico di utilizzare tali unità è quello di trattarle senza partizioni. Per sicurezza, quando si crea un file system Second-extended (Ext2, Ext3 o Ext4) è bene accertarsi di avere blocchi di dimensioni sufficientemente grandi:

```
# mkfs.ext4 -b 4096 /dev/sdc [Invio]
```

### 19.13.3 File-immagine

La possibilità di gestire file che riproducono un'unità di memorizzazione intera può sembrare paradossale o senza senso, ma in realtà, ciò è di grande utilità. In questi casi si parla di file-immagine, solo che il termine «immagine» viene usato in molte circostanze differenti e occorre evitare di lasciarsi confondere.

L'esempio più comune di file-immagine è costituito da quelli fatti per riprodurre un disco ottico.

Per utilizzare i file-immagine di unità di memorizzazione, cioè per poterli innestare come si fa con le unità reali, occorre che il kernel

sia in grado di gestire questa funzione (sezione 8.3.8.3).

### 19.13.3.1 Creazione di un file-immagine normale

«

Un file-immagine di un disco può essere creato a partire da un'unità di memorizzazione esistente, oppure da zero, con l'inizializzazione di un file. Volendo creare l'immagine di un dischetto già esistente si procede semplicemente copiando il file di dispositivo corrispondente all'unità a dischetti nel file che si vuole creare.

```
# cp /dev/fd0 floppy.img[Invio]
```

L'esempio appena mostrato genera il file 'floppy.img' nella directory corrente.

Diversamente si può partire da zero, creando un file e inizializzandolo. Il comando seguente crea il file 'pippo.img' della stessa dimensione di un dischetto da 1440 Kibyte.

```
# dd if=/dev/zero of=pippo.img bs=1024c count=1440[Invio]
```

Il comando successivo serve invece a inizializzarlo in modo da inserirvi un file system (viene utilizzato il formato Ext2).

```
# mkfs.ext2 pippo.img[Invio]

  pippo.img is not a block special device.
  Proceed anyway? (y,n)
```

Trattandosi di una richiesta anomala, il programma '**mkfs.ext2**' vuole una conferma. Basta inserire la lettera '**y**' per proseguire.

```
Proceed anyway? (y,n) y[Invio]
```

Nello stesso modo si può creare un file system differente (per ottenere un file-immagine contenente un file system ISO 9660, usato per

i dischi ottici, si veda la sezione 19.7).

### 19.13.3.2 Accedere a un file-immagine normale

Per accedere a un file contenente l'immagine di un'unità di memorizzazione (con il suo file system), si procede come se si trattasse di un'unità reale o di una partizione normale. In particolare, viene utilizzato '**mount**' con l'opzione '**-o loop**'.

L'immagine cui si accede può essere stata creata sia partendo da un file vuoto che viene inizializzato successivamente, sia dalla copia di un'unità (o di una partizione) in un file.

```
# mount -o loop -t auto pippo.img /mnt/floppy[Invio]
```

Nell'esempio, l'immagine contenuta nel file '`pippo.img`' viene innestata a partire dalla directory '`/mnt/floppy/`', dove si comporta come se si trattasse di un dischetto normale.

### 19.13.3.3 File-immagine suddiviso in partizioni

Un file-immagine potrebbe essere suddiviso in partizioni, con l'ausilio di '**fdisk**' o di un programma equivalente. In tal caso il programma potrebbe richiedere di specificare espressamente la geometria a cui fare riferimento (numero di cilindri, numero di tracce per cilindro, numero di settori per traccia).

Per quanto riguarda invece la possibilità di innestare una partizione del file-immagine, occorre in più specificare lo scarto a partire dall'inizio del file. In pratica, occorre sapere in che posizione (espressa in byte, a partire da zero) inizia la partizione desiderata. Per esempio, un disco organizzato in 1024 cilindri, 8 tracce per cilindro e

32 settori per traccia, potrebbe essere stato suddiviso nelle partizioni seguenti, come visualizzato da '**fdisk**':

```
8 heads, 32 sectors/track, 1024 cylinders
Units = cylinders of 256 * 512 = 131072 bytes

Device Boot
                Start   End  Blocks  Id System
minix.dskp1  *       1  1016  130032  81 Minix / old Linux
minix.dskp4       1017  1024    1024  82 Linux swap / Solaris
```

In questo caso si sta verificando il contenuto del file-immagine '`minix.dsk`', in cui sono state dichiarate la prima e la quarta partizione primaria ('**p1**' e '**p4**').

Si parte dal presupposto che la prima partizione, pur iniziando dal primo cilindro, si trova esattamente dopo la prima traccia, per lasciare spazio al settore MBR ed eventualmente ad altro codice di avvio. Pertanto, la prima partizione di questo esempio inizia dopo 32×512 byte, ovvero dopo 16384. Eventualmente, si può verificare questo calcolo contando quanto dovrebbe essere grande la partizione, se partisse dall'inizio, sottraendo poi la dimensione indicata effettivamente: 1016×8×32×512 = 133169152 = 130048 Kibyte. Il valore riportato è invece di 130032 Kibyte, per cui c'è uno scarto di 16 Kibyte, pari al valore già trovato (16384).

A questo punto, per innestare la prima partizione, si può procedere come nell'esempio seguente, dove si lascia che il tipo di file system venga determinato automaticamente:

```
# mount -o loop,offset=16384 -t auto minix.dsk ↩
↪       /mnt/minix [Invio]
```

Tuttavia, anche se formalmente corretto, non è detto che il comando

funzioni. Per risolvere il problema occorre verificare di disporre nel kernel della funzione *Device mapper support*:

- *Device drivers*
  - *Multi-device support (RAID and LVM)*
    - *Multiple devices driver support (RAID and LVM)*
      - *Device mapper support*

Se la funzione in questione è disponibile, ma solo come modulo, allora va caricato il modulo '**dm_mod**':

```
# modprobe dm_mod[Invio]
```

A questo punto, serve il programma Kpartx,[57] con il quale si generano dei file di dispositivo utili per accedere le partizioni esistenti:

```
# kpartx -v -a minix.dsk[Invio]
```

Con l'aiuto di Kpartx si ottengono dei file di dispositivo ulteriori, con i quali è possibile accedere direttamente alle partizioni del file-immagine. Ammesso di avere ottenuto i dispositivi '`/dev/mapper/loop3p1`' e '`/dev/mapper/loop3p2`', il comando necessario a innestare la prima partizione diventa quello seguente:

```
# mount -o loop -t auto /dev/mapper/loop3p1 /mnt/minix[Invio]
```

### 19.13.3.4 trasferimento di un file-immagine in un'unità reale

Se il file-immagine, all'interno del quale è stato fatto del lavoro, corrisponde esattamente a un'unità di memorizzazione o a una sua

partizione, è possibile riprodurre questa immagine nell'unità o nella partizione corrispondente. Per questo si può utilizzare '**cp**' oppure '**dd**'.

I due esempi seguenti riproducono nello stesso modo il file '`pippo.img`' in un dischetto.

```
# cp pippo.img /dev/fd0 [Invio]

# dd if=pippo.img of=/dev/fd0 [Invio]
```

Naturalmente, se l'immagine è stata innestata in precedenza per poterne modificare il contenuto, occorre ricordarsi di eseguirne il distacco prima di procedere alla riproduzione.

### 19.13.4 Unità di memorizzazione senza file system

GNU/Linux, come altri sistemi Unix, permette di gestire anche unità di memorizzazione di massa, ad accesso diretto, che al loro interno non contengono un file system. Questo concetto potrebbe sembrare scontato per molti, ma tutti quelli che si avvicinano a GNU/Linux provenendo da sistemi in cui tali cose non si possono fare devono porre attenzione a questo particolare.

Un'unità o una partizione senza file system è semplicemente una sequenza di settori. In modo molto semplificato è come se si trattasse di un file. Quando si indicano i nomi di dispositivo legati alle unità intere o alle loro partizioni, si fa riferimento a questi nel loro insieme, come se si trattasse di file.

Quando si vuole utilizzare un'unità o una partizione nel modo con cui si è abituati di solito, cioè per gestire i file al suo interno, la si deve innestare e da quel momento non si fa più riferimento al nome del dispositivo.

### 19.13.5 Controllo dell'unità per dischi ottici

Il kernel Linux ha una propria gestione per l'accesso alle unità CD o DVD, la quale può essere impostata, attraverso un programma apposito. In generale, non c'è alcun bisogno di intervenire sull'impostazione predefinita iniziale, tuttavia può essere interessante conoscere le alternative disponibili, in presenza di situazioni particolari.

Il programma che consente di modificare la configurazione di utilizzo delle unità CD o DVD è '**setcd**':

```
setcd [opzioni] [file_di_dispositivo]
```

Il programma '**setcd**' richiede l'indicazione di almeno un'opzione, altrimenti non si ottiene alcuna azione; inoltre, se non viene indicato il file di dispositivo, il programma cerca di utilizzare '`/dev/cdrom`'. Alcune delle opzioni richiedono un argomento numerico, pari a zero o uno, con il quale si intende disabilitare, oppure abilitare una funzionalità.

Tabella 19.157. Alcune opzioni.

| Opzione | Descrizione |
|---|---|
| `-c0\|1` | Controlla la funzione di caricamento automatico, quando il file di dispositivo corrispondente viene aperto. Generalmente, questa funzione è attiva. |
| `-d`***n*** | Stabilisce il numero con il quale selezionare un disco all'interno di un'unità multipla (come un *juke-box*). |

| Opzione | Descrizione |
|---|---|
| `-e0|1` | Controlla la funzione di caricamento ed espulsione automatici nel momento in cui il disco viene innestato o separato. Generalmente questa funzione è disabilitata, richiedendo per questo scopo l'uso di programmi come '**`eject`**'; a ogni modo, la sua attivazione è sostanzialmente equivalente all'attivazione di '**`-c`**' e '**`-o`**'. |
| `-f0|1` | Controlla una funzione che, se attiva, fa sì che siano ignorati alcuni errori. In generale, conviene che questa funzione sia attiva. |
| `-l0|1` | Controlla la funzione di chiusura dell'unità fisica, in modo da impedire che il disco possa essere espulso manualmente se risulta innestato. In generale, questa funzione è attiva. |
| `-o0|1` | Controlla la funzione di espulsione automatica del disco. In generale, questa opzione è disattivata; di solito, al posto di attivare questa opzione si preferisce usare '**`-e`**'. |
| `-t0|1` | Controlla la funzione di verifica del tipo di disco. Per esempio, questa funzione attiva fa in modo di generare un errore quando si tenta di innestare un CD musicale. In generale, questa funzione è attiva. |
| `-i` | Fornisce delle informazioni sullo stato dell'unità ed eventualmente sul tipo di disco contenuto. |

| Opzione | Descrizione |
|---|---|
| `-r` | Riporta le opzioni di funzionamento allo stato predefinito, pari a '**`-c1 -f1 -l1 -o0 -t1`**'. |
| `-s` | Fornisce l'elenco dello stato delle opzioni di funzionamento dell'unità richiesta. |

Segue la descrizione di alcuni esempi.

- `#` **`setcd -s /dev/sr0`** [*Invio*]

  Mostra la situazione attuale della configurazione relativa all'unità che corrisponde al file di dispositivo '`/dev/sr0`'. La situazione iniziale dovrebbe essere quella seguente:

```
/dev/sr0:
  Auto close tray:     set
  Auto open tray:      cleared
  Use O_NONBLOCK flag: set
  Lock tray:           set
  Check CD type:       cleared
```

  Come si vede, risultano disattivate le funzioni di espulsione automatica (opzione '**`-o`**') e di controllo del tipo di disco (opzione '**`-t`**').

- `#` **`setcd -r /dev/sr0`** [*Invio*]

  Imposta la configurazione predefinita secondo la logica di '**`setcd`**':

```
/dev/sr0:
  Auto close tray:      set
  Auto open tray:       cleared
  Use O_NONBLOCK flag: set
  Lock tray:            set
  Check CD type:        set
```

- # **setcd -l0 /dev/sr0** [*Invio*]

  Fa in modo che si possa espellere manualmente il disco contenuto nell'unità del file di dispositivo '`/dev/sr0`', in ogni occasione. Tuttavia, si tratta di un'azione sconsigliabile.

```
/dev/sr0:
  Lock tray:            cleared
```

### 19.13.6 Estrazione della prima traccia dati

«

Un file system ISO 9660 può trovarsi in un CD-ROM o in un altro tipo di unità di memorizzazione, precisamente nella prima traccia dati. Qualunque sia la situazione, questa traccia dati può avere una dimensione inferiore all'unità di memorizzazione. Trovandosi nella necessità di estrarla, è utile conoscerne tale dimensione.

Se c'è il modo di innestarlo, basta utilizzare successivamente il comando '**df**' per sapere esattamente il numero di kibibyte (simbolo: «Kibyte») contenuti; ma in alternativa si può utilizzare Isosize[58], corrispondente al programma eseguibile '**isosize**':

```
isosize file_di_dispositivo
```

Per esempio, il comando seguente mostra in che modo sia possibile determinare la dimensione della prima traccia dati del CD-ROM inserito nel lettore corrispondente al dispositivo ‘`/dev/sr0`’:

```
# isosize /dev/sr0 [Invio]
```

Volendo estrarre la traccia, senza altri dati aggiuntivi, si potrebbe utilizzare ‘**dd**’ nel modo seguente:

```
# dd if=/dev/sr0 of=traccia bs=1b ↩
↪    count=`isosize /dev/sr0` [Invio]
```

### 19.13.6.1 Verifica di un disco appena registrato

La verifica del successo o meno nella registrazione di un CD-ROM o di un DVD-ROM può essere fatta in un modo piuttosto semplice: leggendo tutto il contenuto e verificando se con questa operazione si ottengono delle segnalazioni di errore.

Supponendo di disporre di un lettore per CD-ROM o per DVD-ROM in corrispondenza del dispositivo ‘`/dev/sr0`’, si potrebbe procedere come segue:

```
# cat /dev/sr0 > /dev/null 2> /tmp/errori.txt [Invio]
```

Se tutto va bene, alla fine si ottiene un file ‘`/tmp/errori.txt`’ vuoto. Altrimenti il file riporta una segnalazione del tipo seguente:

```
cat: /dev/sr0: I/O error
```

Alle volte si possono osservare sullo schermo delle segnalazioni di errore aggiuntive anche quando il file ‘`/tmp/errori.txt`’, o un suo equivalente, risulta vuoto alla fine del test. Dal momento che sia lo standard output, sia lo standard error del comando, sono ridiretti, si tratta di messaggi estranei provenienti dal sistema. A tali messaggi

di errore corrispondono poi dei nuovi tentativi; solo se il sistema non riesce in alcun modo a superare gli errori viene coinvolto il comando stesso, che poi lo segnala attraverso lo standard error.

Se si ottiene una segnalazione di errore attraverso lo standard error di un comando di lettura, come '**cat**', il CD-ROM o il DVD-ROM è difettoso, altrimenti gli errori segnalati sullo schermo sono ignorabili. Inoltre, è il caso di ricordare che prima di iniziare il controllo di un altro disco, è necessario cancellare il file di destinazione dello standard error.

```
# rm /tmp/errori.txt ; cat /dev/sr0 > /dev/null ↩
↪                                    2> /tmp/errori.txt [Invio]
```

Tuttavia, si potrebbe fare meglio utilizzando il programma '**isosize**' già descritto nella sezione precedente. In questo modo si evitano tentativi di lettura oltre la fine della traccia, che generano normalmente degli errori tali da creare un po' di confusione:

```
# dd if=/dev/sr0 of=/dev/null bs=1b ↩
↪   count=`isosize /dev/sr0` [Invio]
```

## 19.14 Riferimenti

«

- Stein Gjoen, *Disk HOWTO*, *http://www.ibiblio.org/pub/linux/docs/howto/other-formats/pdf/Multi-Disk-HOWTO.pdf*
- Jeff Tranter, *CDROM HOWTO*, *http://www.ibiblio.org/pub/linux/docs/howto/other-formats/pdf/CDROM-HOWTO.pdf*
- Winfried Trümper, *CD-writing HOWTO*, *http://www.ibiblio.org/pub/linux/docs/howto/other-formats/pdf/CD-Writing-HOWTO.pdf*

- *What is DVD?*, http://www.videohelp.com/dvd.htm
- Wikipedia, *Universal disk format*, http://en.wikipedia.org/wiki/Universal_Disk_Format
- Jakob Østergaard, *The Software RAID HOWTO*, http://unthought.net/Software-RAID.HOWTO/
- Wikipedia, *FUSE (Linux)*, http://en.wikipedia.org/wiki/FUSE_(Linux)
- *FUSE: filesystem in userspace*, http://fuse.sourceforge.net/ , http://fuse.sourceforge.net/wiki/
- Linux Foundation, *Filesystem hierarchy standard*, http://www.linuxfoundation.org/collaborate/workgroups/lsb/fhs


[1] Per memoria cache si intende una porzione della memoria centrale in cui si trattengono delle informazioni.

[2] L'inizializzazione a basso livello riguarda in modo particolare i dischetti, ormai scomparsi, oppure i primi dischi fissi, o unità rimovibili speciali, anch'esse non più utilizzate. Rimane tuttavia il problema dell'inizializzazione a basso livello, da non sottovalutare, in quanto potrebbe ripresentarsi con unità di memorizzazione di nuova concezione.

[3] **E2fsprogs** GNU GPL

[4] **util-linux: fdisk** GNU GPL

[5] **GNU fdisk** GNU GPL

[6] **util-linux: cfdisk** GNU GPL

[7] **util-linux: sfdisk** GNU GPL

[8] **E2fsprogs** GNU GPL

[9] **Mkdosfs** GNU GPL

[10] **util-linux: mkfs** GNU GPL

[11] **Disktype** software libero con licenza speciale

[12] **E2fsprogs** GNU GPL

[13] **E2fsprogs** GNU GPL

[14] Quando ci si trova a dover controllare il file system principale del sistema in funzione, in mancanza di altro, si fanno terminare tutti i processi che non sono indispensabili, quindi si reinnesta il file system principale in sola lettura ('`mount -o remount,ro /`'). Tuttavia, dopo il controllo, dal momento che questo modifica il file system, non conviene ripetere il reinnesto in lettura e scrittura, dato che il sistema potrebbe continuare a ignorare tali modifiche; pertanto, piuttosto è bene riavviare il sistema.

[15] **Dosfsck** GNU GPL

[16] '`dosfsck`' non è un programma che viene installato in modo predefinito dalle distribuzioni, per cui, nella maggior parte dei casi occorre provvedere direttamente per questo.

[17] **NTFSprogs** GNU GPL

[18] **Doschk** GNU GPL

[19] **Gpart** GNU GPL

[20] **Parted** GNU GPL

[21] **Partimage** GNU GPL

[22] **util-linux: mount** GNU GPL

[23] **util-linux: umount** GNU GPL

[24] Si tratta di una procedura per ottenere delle copie di sicurezza che comunque non è indispensabile. Per questo si possono usare strumenti normali senza bisogno si utilizzare la configurazione di questo file.

[25] **GNU core utilities** GNU GPL

[26] **di** software libero con licenza speciale

[27] Questa sezione su '**du**' è identica alla 20.4.4.

[28] **GNU core utilities** GNU GPL

[29] **GNU core utilities** GNU GPL

[30] **GNU core utilities** GNU GPL

[31] L'utilizzo della memoria di massa come estensione della memoria centrale è possibile solo se la tecnologia con cui questa memoria è realizzata consente una scrittura frequente. In pratica, ci si avvale per questo, quasi esclusivamente, di unità a dischi magnetici.

[32] **util-linux: mkswap** GNU GPL

[33] **util-linux: swapon** GNU GPL

[34] **util-linux: swapoff** GNU GPL

[35] In pratica, per attribuire delle quote al file system principale, occorre reinnestarlo in una sottodirectory, con le stesse opzioni.

[36] **Linux quota utils** UCB BSD e GNU GPL

[37] Si hanno i gruppi privati quando ogni utente ha un proprio gruppo personale.

[38] In tal caso si parla comunemente di CD-ROM, DVD-ROM ed eventualmente di BD-ROM.

[39] **Cdrkit http://www.cdrkit.org** GNU GPL

[40] **Isolinux** GNU GPL

[41] **Cdrkit http://www.cdrkit.org** GNU GPL

[42] Generalmente, l'utilizzo di un condotto (*pipeline*) per generare l'input di un programma di scrittura di un disco ottico è sconsigliabile. Ciò perché potrebbero verificarsi dei ritardi nel flusso di dati che giunge all'unità di registrazione, provocando l'interruzione irreversibile della procedura.

[43] **DVD+rw-tools** GNU GPL

[44] **Cloop** GNU GPL

[45] **Squashfs** GNU GPL

[46] **Cramfs** GNU GPL

[47] **Zisofs-tools** GNU GPL

[48] **Mdadm** GNU GPL

[49] **SSHfs** GNU GPL

[50] **CurlFtpFS** GNU GPL

[51] CurlFtpFS consente di indicare il nominativo utente e la parola d'ordine tra le opzioni di avvio, ma ciò è assolutamente sconsigliabile perché la riga di comando è sempre visibile a tutti gli utenti.

[52] **NTFSprogs** GNU GPL

[53] **NTFS-3g** GNU GPL

[54] **EncFS** GNU GPL

[55] **util-linux: fdformat** GNU GPL

[56] **Linux floppy utilities** GNU GPL

[57] **Kpartx** GNU GPL

[58] **Isosize** GNU GPL