Linux Partition HOWTO

Anthony Lissot

Diario delle Revisioni
Revisione 3.5   26 Dicembre 2005
Riorganizzato l'ordine delle pagine. Aggiunta pagina sulla configurazione dello spazio di swap. Aggiunta pagina sulle etichette delle partizioni. Aggiornati i valori per la dimensione massima di swap nella sezione 4. Aggiunte istruzioni sulla creazione di file system ext2/3. I collegamenti interrotti individuati da Richard Calmbach sono stati corretti. Creata una versione XML.
Revisione 3.4.4 08 Marzo 2004
Sincronizzata la versione SGML con la versione HTML. Aggiornata la trattazione della posizione di lilo e della dimensione di swap.
Revisione 3.3   04 Aprile 2003
Sincronizzate le versioni SGML e HTML.
Revisione 3.3   10 Luglio 2001
Corretta la sezione 6, calcolo del numero dei cilindri.
Revisione 3.2   1 Settembre 2000
Dan Scott ha fornito la conversione sgml, 2 Ott. 2000. Riscritta l'introduzione. Riscritta l'analisi dei nomi di device in device logici. Riorganizzata la sezione su i tipi di partizioni. Modificata la sezione sui requisiti delle partizioni. Aggiunta la sezione su come recuperare una tabella delle partizioni cancellata.
Revisione 3.1   12 Giugno 2000
Corretto il limite alla dimensione di swap nei requisiti delle partizioni, aggiornati vari collegamenti nell'introduzione, aggiunto un contributo con un esempio in come partizionare con fdisk, aggiunta un'analisi sul file system nei requisiti delle partizioni.
Revisione 3.0   1 Maggio 2000
Prima revisione a cura di Anthony Lissot basata su Linux Partition HOWTO di Kristian Koehntopp.
Revisione 2.4   3 Novembre 1997
Ultima revisione a cura di Kristian Koehntopp.

Questo Mini-HOWTO Linux insegna come pianificare e creare partizioni su dischi fissi IDE e SCSI. Esamina la terminologia legata alle partizioni e prende in considerazione questioni legate a dimensioni e posizionamento. Viene trattato l'uso di fdisk come utilità di partizionamento per creare e recuperare tabelle delle partizioni. La versione più recente di questo documento è qui (http://lissot.net/partition). La traduzione in turco è qui (http://belgeler.org/howto/partition-howto.html).

Traduzione in italiano di Beatrice Torracca, beatricet (at) libero.it. Revisione a cura di Elisa Chiorrini, elisa.chiorrini (at) gmail.com

1. Introduzione

1.1. Che cosa è una partizione?

Partizionare significa dividere un singolo disco fisso in diversi dispositivi logici. Una partizione è un insieme contiguo di blocchi su un dispositivo che vengono trattati come un disco indipendente. Una tabella delle partizioni (la cui creazione è l'argomento di questo HOWTO) è un indice che mette in correlazione le sezioni del disco fisso con le partizioni.

Perché avere partizioni multiple?

- Per incapsulare i propri dati. Dato che la corruzione di un file system è limitata ad una partizione, se accade un imprevisto si corre il rischio di perdere solo parte dei propri dati.
- Per aumentare l'efficienza dello spazio su disco. Si possono formattare partizioni con dimensioni dei blocchi diverse, a seconda dell'uso. Se i propri dati sono in forma di un grande numero di piccoli file (meno di 1k) e la propria partizione usa blocchi di 4k, si stanno sprecando 3k per ogni file. In generale, si spreca in media la metà di un blocco per ogni file, perciò, se si hanno molti file, far corrispondere la dimensione dei propri blocchi alla dimensione media dei propri file è importante.
- Per limitare la crescita dei dati. Processi alla deriva o utenti irragionevoli possono consumare così tanto spazio su disco da non lasciare più al sistema operativo abbastanza spazio sul disco fisso per le sue operazioni di gestione del sistema. Ciò porta a disastri. Segregando gli spazi, ci si assicura che cose diverse dal sistema operativo muoiano quando lo spazio allocato su disco è esaurito.

1.2. Altro software per il partizionamento

- **sfdisk**: una versione a riga di comando di fdisk.
- **cfdisk**: una versione basata su curses di fdisk.
- **parted** (http://www.gnu.org/software/parted/parted.html): editor di partizioni Gnu.
- Partition Magic (http://www.powerquest.com/partitionmagic/index.html): un'utilità commerciale per creare, ridimensionare, fondere e convertire partizioni, senza perdere dati.
- Disk Drake (http://www.linux-mandrake.com/diskdrake): un programma Perl/Gtk per creare, ridimensionare ed eliminare partizioni.

1.3. HOWTO correlati

Tabella 1. HOWTO correlati

Titolo	Autore	Descrizione
Dual boot install strategies (http://www.tldp.org/HOWTO/Install-Strategies)	Gjoen Stein (mailto:tobbyb@usa.net)	Come stimare i vari requisiti di spazio e velocità per parti differenti del file system.
Linux Multiple Disk System Tuning (http://www.nyx.net/~sgjoen/disk.html)	Gjoen Stein (mailto:sgjoen@mail.nyx.net)	Come stimare i vari requisiti di spazio e velocità per parti differenti del file system.

Titolo	Autore	Descrizione
Linux Large Disk (http://www.tldp.org/HOWTO/Large-Disk-HOWTO.html)	Andries Brouwer (mailto:aeb@cwi.nl)	Istruzioni e considerazioni riguardanti dischi con più di 1024 cilindri.
Linux Quota (http://metalab.unc.edu/mdw/HOWTO/mini/Quota.html)	Ralf van Dooren (mailto:ralf.vandooren@snow.nl)	Istruzioni su come limitare l'uso dello spazio su disco per ciascun utente (le quote).
Partition-Rescue mini-HOWTO (http://www.linuxdocs.org/HOWTOs/mini/Partition-Rescue.html)	Jean-Daniel Dodin (mailto:jean-daniel@dodin.net)	Come ripristinare partizioni Linux dopo che siano state cancellate da una installazione di Windows. Non sembra preservare i dati.
Linux ADSM Backup (http://www.tldp.org/HOWTO/ADSM-Backup.html)	Thomas Koenig (mailto:Thomas.Koenig@ciw.uni-karlsruhe.de)	Istruzioni sull'integrazione di Linux in un sistema di backup IBM ADSM.
Linux Backup with MSDOS (http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/html_single/Backup-With-MSDOS.html)	Christopher Neufeld (mailto:neufeld@physics.utoronto.ca)	Informazioni su backup di Linux basati su MS-DOS.
Linux HOWTO Index	Tim Bynum (mailto:linux-howto@sunsite.unc.edu)	Istruzioni su come scrivere e sottoporre un documento HOWTO.

1.4. Informazioni aggiuntive sul proprio sistema

- `/usr/src/linux/Documentation` (file:/usr/src/linux/Documentation)
  - `ide.txt` (file:/usr/src/linux/Documentation/ide.txt): informazioni sui propri driver IDE.
  - `scsi.txt` (file:/usr/src/linux/Documentation/scsi.txt): informazioni sui propri driver SCSI.

2. Device

C'è una speciale nomenclatura che Linux usa per fare riferimento alle partizioni sui dischi fissi e che deve essere capita per potere seguire gli argomenti trattati nelle pagine seguenti.

In Linux, le partizioni sono rappresentate da file di device. Questi sono file fittizi posizionati in `/dev`. Ecco alcune voci:

```
brw-rw----    1 root     disk       3,   0 May  5  1998 hda
brw-rw----    1 root     disk       8,   0 May  5  1998 sda
crw-------    1 root     tty        4,  64 May  5  1998 ttyS0
```

Un file di device è un file di tipo c (sta per device a "caratteri", device che non usano la buffer cache) o b (sta per device a "blocchi", che passano attraverso la buffer cache). In Linux, tutti i dischi sono rappresentati esclusivamente come device a blocchi.

2.1. Nomi di device

2.1.1. Convenzioni per i nomi

Per convenzione, alle unità IDE vengono assegnati i nomi di device da `/dev/hda` a `/dev/hdd`. *H*ard *D*rive *A* (`/dev/hda`) è la prima unità e *H*ard *D*rive *C* (`/dev/hdc`) è la terza.

Tabella 2. Convenzioni per i nomi dei controller IDE

nome dell'unità	controller dell'unità	numero dell'unità
/dev/hda	1	1
/dev/hdb	1	2
/dev/hdc	2	1
/dev/hdd	2	2

Un PC tipico ha due controller IDE, ognuno dei quali può avere connesse due unità. Per esempio, `/dev/hda` è la prima unità (master) sul primo controller IDE e `/dev/hdd` è la seconda unità (slave) sul secondo controller (la quartà unità IDE nel computer).

Si può scrivere su questi device in modo diretto (usando **cat** o **dd**). Tuttavia, dato che questi device rappresentano l'intero disco, partendo dal primo blocco, si può per errore sovrascrivere il master boot record e la tabella delle partizioni, il che renderebbe l'unità inutilizzabile.

Tabella 3. Nomi delle partizioni

nome dell'unità	controller dell'unità	numero dell'unità	tipo di partizione	numero di partizione
/dev/hda1	1	1	primaria	1
/dev/hda2	1	1	primaria	2

/dev/hda3	1	1	primaria	3
/dev/hda4	1	1	swap	ND
/dev/hdb1	1	2	primaria	1
/dev/hdb2	1	2	primaria	2
/dev/hdb3	1	2	primaria	3
/dev/hdb4	1	2	primaria	4

Una volta che un'unità è stata partizionata, le partizioni saranno rappresentate da numeri alla fine dei nomi. Per esempio, la seconda partizione sulla seconda unità sarà `/dev/hdb2`. Il tipo di partizione (primaria) è per chiarezza elencato nella tabella soprastante, anche se il concetto non verrà spiegato prima de la Sezione 3.3.

Tabella 4. Unità SCSI

nome dell'unità	controller dell'unità	numero dell'unità	tipo di partizione	numero di partizione
/dev/sda1	1	6	primaria	1
/dev/sda2	1	6	primaria	2
/dev/sda3	1	6	primaria	3

Le unità SCSI seguono uno schema simile; sono rappresentate da "sd" invece di "hd". La prima partizione della seconda unità SCSI sarà pertanto `/dev/sdb1`. Nella tabella soprastante, il numero dell'unità, cioè 6, è scelto arbitriamente per introdurre il concetto che i numeri identificativi SCSI non hanno corrispondenza nei nomi di device in Linux.

2.1.2. Assegnazione dei nomi

In (Sun) Solaris e (SGI) IRIX, il nome di device assegnato ad un'unità SCSI ha una relazione con il punto in cui essa è connessa. In Linux, c'è solo pianto e stridore di denti.

Prima

```
SCSI ID #2        SCSI ID #5       SCSI ID #7        SCSI ID #8
 /dev/sda          /dev/sdb         /dev/sdc          /dev/sdd
```

Dopo

```
SCSI ID #2                         SCSI ID #7        SCSI ID #8
 /dev/sda                           /dev/sdb          /dev/sdc
```

Le unità SCSI hanno numeri identificativi, ID, che vanno da 1 a 15. A numeri identificativi SCSI più bassi sono assegnate lettere di ordine inferiore. Per esempio, se si hanno due unità con numeri 2 e 5, allora la numero 2 è `/dev/sda` e la 5 è `/dev/sdb`. Se si rimuove una qualunque di esse, tutte le unità con numero più alto avranno al successivo avvio un nuovo nome.

Se si hanno due controller SCSI nella propria macchina Linux, è necessario esaminare l'output di **/bin/dmesg** al fine di vedere quale nome è stato assegnato a ciascuna unità. Se si rimuove uno dei due controller, tutte le unità del controller rimanente potrebbero essere rinominate. Grrr...

Ci sono due modi di aggirare l'ostacolo; entrambi comportano l'uso di un programma per assegnare un'etichetta a ciascuna partizione (vedere la la Sezione 6). L'etichetta persiste anche quando l'unità viene spostata fisicamente. Ci si riferisce in seguito alla partizione direttamente o indirettamente tramite la sua etichetta.

2.1.3. Partizioni logiche

Tabella 5. Partizioni logiche

nome dell'unità	controller dell'unità	numero dell'unità	tipo di partizione	numero di partizione
/dev/hdb1	1	2	primaria	1
/dev/hdb2	1	2	estesa	ND
/dev/hda5	1	2	logica	2
/dev/hdb6	1	2	logica	3

La tabella soprastante illustra un misterioso salto nell'assegnazione dei nomi. Ciò è dovuto all'uso di partizioni logiche (vedere la la Sezione 3.4), che partono sempre dal numero 5, per ragioni spiegate più avanti.

Questo è tutto ciò che si deve sapere per avere a che fare con unità a disco in Linux. Per amor di completezza, si veda la spiegazione di Kristian sui numeri di device qui sotto.

2.2. Numeri di device

Le uniche cose importanti di un file di device sono i suoi numeri di device primario e secondario, che vengono mostrati al posto della dimensione del file:

```
$ ls -l /dev/hda
```

Tabella 6. Attributi dei file di device

brw-rw----	1	root	disk	3,	0	Jul 18 1994	/dev/hda
permessi		proprietario	gruppo	numero primario del device	numero secondario del device	data	nome del device

Quando si accede ad un file di device, il numero primario seleziona quale driver di device viene chiamato per effettuare l'operazione di input/output. Questa chiamata viene fatta con il numero secondario come parametro e l'interpretazione del numero secondario è totalmente a carico del driver. La documentazione del driver normalmente descrive come il driver usa i numeri secondari. Per i dischi IDE, questa documentazione si trova in `/usr/src/linux/Documentation/ide.txt` (file:/usr/src/linux/Documentation/ide.txt). Per i dischi SCSI, ci si aspetterebbe tale documentazione in `/usr/src/linux/Documentation/scsi.txt` (file:/usr/src/linux/Documentation/scsi.txt), ma lì non c'è. Si deve guardare il sorgente dei driver per essere sicuri (`/usr/src/linux/driver/scsi/sd.c` (file:/usr/src/linux/driver/scsi/sd.c):184-196). Per fortuna c'è l'elenco dei numeri e nomi di device di Peter Anvin in `/usr/src/linux/Documentation/devices.txt` (file:/usr/src/linux/Documentation/devices.txt); vedere le voci per i device a blocchi, numero primario 3, 22, 33, 34 per i dischi IDE e numero primario 8 per i dischi SCSI. I numeri primario e secondario sono rappresentati da un byte ciascuno ed è per questo che il numero di partizioni per ciascun disco è limitato.

3. Tipi di partizioni

3.1. Tipi di partizioni

Una partizione è etichettata per ospitare un certo tipo di file system (da non confondersi con l'etichetta di volume; vedere la la Sezione 6). Tale file system potrebbe essere un file system Linux standard ext2 o spazio di swap Linux o anche file system estranei come (Microsoft) NTFS o (Sun) UFS. C'è un codice numerico associato ad ogni tipo di partizione. Per esempio, il codice per ext2 è `0x83` e per lo swap Linux è `0x82`. Per vedere un elenco dei tipi di partizione e dei loro codici, eseguire **`/sbin/sfdisk -T`**.

3.2. Tipi di partizione estranei

I codici dei tipi di partizione sono stati scelti arbitrariamente (non si può calcolare quale deve essere il loro valore) e sono specifici di un dato sistema operativo. Perciò è in teoria possibile che se si usano due sistemi operativi con lo stesso disco fisso, lo stesso codice sia usato per indicare due diversi tipi di partizione.

OS/2 segna le sue partizioni con tipo `0x07` e così fa l'NTFS di Windows NT. MS-DOS assegna diversi codici di tipo per le diverse versioni dei file system FAT: si conoscono 0x01, `0x04` e `0x06`. DR-DOS usava `0x81` per indicare partizioni FAT protette, creando, a quel tempo, un conflitto di tipo con Linux/Minix, ma sia Linux/Minix sia DR-DOS non sono più molto usati.

3.3. Partizioni primarie

Il numero delle partizioni in un sistema Intel è stato limitato sin dagli esordi: la tabella originale delle partizioni è stata installata come parte del settore di avvio e conteneva lo spazio per solo quattro voci di partizione. Queste partizioni vengono ora chiamate partizioni primarie.

3.4. Partizioni logiche

Una partizione primaria di un disco fisso può essere ulteriormente partizionata. Quelle che si ottengono sono partizioni logiche. Questo permette di fatto di aggirare la limitazione storica a quattro partizioni.

La partizione primaria usata per contenere le partizioni logiche è chiamata partizione estesa ed ha il suo tipo di file system (0x05). A differenza delle partizioni primarie, le partizioni logiche devono essere contigue. Ogni partizione logica contiene un puntatore alla seguente partizione logica, il che implica che il numero delle partizioni logiche è illimitato. In ogni caso Linux impone dei limiti sul numero totale di ciascun tipo di partizione su un'unità, perciò in effetti pone un limite al numero delle partizioni logiche. Questo è 15 partizioni totali al massimo su un disco SCSI e 63 totali su un disco IDE.

3.5. Partizioni di swap

Ad ogni processo in esecuzione sul computer è assegnato un numero di blocchi di RAM. Questi blocchi sono chiamati pagine. L'insieme delle pagine nella memoria a cui farà riferimento il processore nell'immediato futuro è chiamato "insieme di lavoro". Linux cerca di prevedere questi accessi alla memoria (ipotizzando che le pagine usate di recente lo saranno ancora nell'immediato futuro) e, se possibile, mantiene queste pagine nella RAM.

Se ci sono troppi processi in esecuzione su una macchina, il kernel cerca di liberare della RAM scrivendo parte delle pagine su disco. Questo è lo scopo per cui esiste lo spazio di swap. Aumenta di fatto la quantità di memoria a disposizione. L'I/O su disco è tuttavia circa cento volte più lento della lettura e scrittura dalla RAM. Considerare questa come memoria di emergenza, non come memoria aggiuntiva.

Se la memoria diventa così poca che il kernel toglie pagine dall'insieme di lavoro di un processo al fine di dare pagine ad un altro, si usa dire cha la macchina è in trashing. Alcuni dei lettori potranno averlo sperimentato inavvertitamente: il disco fisso macina da pazzi, ma il computer è lento a tal punto da essere inutilizzabile. Lo spazio di swap è qualcosa che si deve avere, ma non è un sostituto per una RAM insufficiente. Vedere la trattazione nella la Sezione 4.4 per suggerimenti su come determinare la dimensione dello spazio di swap di cui si ha bisogno.

4. Requisiti per il partizionamento

4.1. Quali partizioni sono necessarie?

a) Per l'unità di avvio. Se si vuole avviare il sistema operativo dall'unità che ci si accinge a partizionare, sono necessarie:

- una partizione primaria;
- una o più partizioni di swap;
- zero o più partizioni primarie o logiche.

b) Per ogni altra unità:

- una o più partizioni primarie o logiche;
- zero o più partizioni di swap.

4.2. Analisi

Partizione di avvio

La partizione di avvio dovrebbe essere una partizione primaria, non una partizione logica. Questo facilita il ripristino in caso di disastri, ma non è tecnicamente necessario. Deve essere di tipo `0x83` "nativa Linux". Se si usa una versione di **lilo** (http://tldp.org/HOWTO/LILO.html) antecedente la 21-3 (cioè, degli anni '90), la propria partizione di avvio deve essere contenuta all'interno dei primi 1024 cilindri dell'unità. Tipicamente è necessario solamente che la partizione di avvio contenga l'immagine del kernel.

Se si ha più di una partizione di avvio (da altri SO, per esempio) le si mantenga tutte nei primi 1024 cilindri (*tutte* le partizioni DOS devono essere all'interno dei primi 1024). Se si sta usando una versione moderna di lilo, o un meccanismo diverso da lilo per caricare il proprio kernel (un disco di avvio, per esempio, o il caricatore di Linux basato su MS-DOS **LOADLIN.EXE**), la partizione può essere ovunque. Per dettagli, vedere il Large-disk (http://www.tldp.org/HOWTO/Large-Disk-HOWTO.html) HOWTO.

Partizione di swap

A meno che non si faccia lo swap verso file (vedere la la Sezione 9.2) è necessaria una partizione swap dedicata. Deve essere di tipo `0x82` "swap Linux". Può essere posizionata ovunque sul disco (ma vedere la la Sezione 4.4.3). Per lo swap può essere usata sia un partizione primaria sia una logica. Su un'unità può esistere più di una partizione di swap. Ne sono permesse 8 in totale (fra tutte le unità). Vedere le note sulla dimensione di swap più sotto (la Sezione 4.4).

Partizione logica

Una singola partizione primaria deve essere usata da contenitore (partizione estesa) per le partizioni logiche. La partizione estesa può essere in qualunque punto del disco. Le partizioni logiche devono essere contigue, ma non è necessario che occupino tutta la partizione estesa.

4.3. File system

4.3.1. Quali file system hanno bisogno di una loro propria partizione?

Tutto il proprio file system Linux può andare nella stessa (unica) partizione. Tuttavia, ci sono circostanze in cui potrebbe essere preferibile contenere la crescita di certi file system. Per esempio, se lo spool di posta fosse nella stessa partizione del file system radice e riempisse lo spazio restante nella partizione, il computer praticamente si pianterebbe.

`/var`

Contiene le directory di spool come quelle per la posta e la stampa. In aggiunta, contiene la directory con i log di errore. Se la propria macchina è un server e sviluppa un errore cronico, i messaggi risultanti possono riempire la partizione. I computer server dovrebbero avere /var in una partizione diversa da /.

`/usr`

Qui si trova la maggior parte degli eseguibili. In aggiunta, l'albero dei sorgenti del kernel è qui, così come molta documentazione.

`/tmp`

Alcuni programmi scrivono qui file di dati temporanei. Di norma sono piuttosto piccoli. Tuttavia, se si eseguono compiti molto intensi dal punto di vista del calcolo, come applicazioni scientifiche o di progettazione, potrebbero essere necessarie centinaia di megabyte per brevi periodi di tempo. In questo caso, tenere /tmp in una partizione diversa da /.

`/home`

Qui si trovano le directory home degli utenti. Se non vengono imposte quote agli utenti, questa dovrebbe essere in una partizione propria.

`/boot`

Qui si trova l'immagine del kernel. Vedere quanto detto sopra per il suo posizionamento in sistemi vecchi.

4.3.2. Tempo di vita dei file e cicli di backup come criteri per il partizionamento

Con ext2, le decisioni circa il partizionamento dovrebbo essere governate da considerazioni sui backup e sull'evitare la frammentazione esterna la Sezione 10.4 dovuta ai tempi di vita differenti dei file.

I file hanno un tempo di vita. Dopo che un file è stato creato, rimarrà per qualche tempo nel sistema e poi sarà rimosso. Il tempo di vita dei file varia moltissimo sul sistema ed è in parte dipendente dal nome di percorso del file. Per esempio, file in `/bin`, `/sbin`, `/usr/sbin`, `/usr/bin` e directory simili probabilmente avranno un tempo di vita molto lungo: molti mesi o più. File in `/home` probabilmente avranno un tempo di vita medio: diverse settimane o giù di lì. File in `/var` hanno di solito vita breve: quasi nessun file in `/var/spool/news` rimarrà più a lungo di alcuni giorni, la vita dei file in `/var/spool/lpd` si misura in minuti o meno.

Per i backup è utile che la dimensione del backup giornaliero sia inferiore alla capacità del singolo supporto di backup. Un backup giornaliero può essere un backup completo o un backup incrementale.

Si può decidere di mantenere le dimensioni delle proprie partizioni piccole abbastanza da essere completamente contenute in un unico supporto di backup (scegliere il backup completo giornaliero). In ogni caso una partizione dovrebbe essere abbastanza piccola da far sì che il suo delta (tutti i file modificati) giornaliero possa essere contenuto in un unico supporto di backup (scegliere il backup incrementale e aspettarsi di cambiare il supporto di backup per il backup completo settimanale/mensile; non è possibile il funzionamento senza supervisione).

La propria strategia di backup dipende da tale decisione.

Quando si pianifica e si compra spazio disco, ricordarsi di accantonare una quantità di denaro sufficiente per il backup! Dati senza backup non valgono nulla! I costi di ricostruzione dei dati sono molto più alti dei costi di backup praticamente per chiunque!

Nell'ottica delle prestazioni è utile tenere file con tempi di vita diversi in partizioni diverse. In questo modo i file con vita breve nella partizione dei messaggi Usenet possono essere fortemente frammentati. Ciò non ha alcun impatto sulle prestazioni della partizione `/` o `/home`.

4.4. Partizioni di swap

4.4.1. Quanto grande deve essere lo spazio di swap?

La prassi convenzionale vuole che si crei uno spazio di swap uguale alla quantità di RAM.

Si tenga però a mente che questa è solo una regola empirica. Facilmente si possono immaginare scenari in cui i programmi hanno insiemi di lavoro estremamente grandi o estremamente piccoli (vedere la la Sezione 3.5). Per esempio, un programma di simulazione con un ampio insieme di dati a cui si accede in modo molto casuale non avrebbe nel suo segmento dati quasi nessuna localizzazione preferenziale di riferimenti, perciò il suo insieme di lavoro sarebbe piuttosto grande.

All'opposto, un programma grafico con molti file JPEG aperti contemporaneamente, tutti ridotti ad icona tranne uno, avrebbe un segmento dati molto grande. Le modifiche alle immagini sono però fatte tutte su una singola immagine; non si accede alla maggior parte della memoria occupata dal programma. Lo stesso vale per un elaboratore di testi con molte finestre aperte in cui solo una finestra alla volta è modificata. Questi programmi hanno, se sono progettati correttamente, una alta localizzazione preferenziale dei riferimenti e una loro grande parte può essere mantenuta nello spazio di swap senza un impatto eccessivamente grave sulle prestazioni. Un utente che non chiude mai i programmi una volta che li ha avviati potrebbe volere, per questa stessa ragione, molto spazio di swap.

I server tipicamente sono configurati con più spazio di swap delle loro controparti desktop. Anche se una certa quantità di swap è sufficiente per il suo funzionamento, il server può trovarsi temporaneamente con carichi pesanti che lo portano a spostare pagine in swap ad un ritmo elevato. Alcuni amministratori preferiscono questo al fare andare il server completamente in crash. In questi casi, lo spazio di swap può essere diverse volte la dimensione della RAM.

4.4.2. Quanto grande *può* essere lo spazio di swap?

Attualmente la dimensione massima della partizione di swap dipende dall'architettura. Per i386, m68k, ARM e PowerPC, è "ufficialmente" 2GB. È 128GB su alpha, 1GB su sparc e 3TB su sparc64. Un opteron con kernel 2.6 può scrivere su una partizione di swap di 16 TB. Per i kernel Linux 2.1 e precedenti, il limite è 128MB. La partizione può essere più grande di 128 MB, ma lo spazio in eccesso non viene mai utilizzato. Se si vuole più di 128 MB di swap per un kernel 2.1 o precedente, si devono creare partizioni di swap multiple (massimo 8). Dopo il 2.4, 32 aree di swap sono "ufficialmente" possibili. Per dettagli vedere la la Sezione 9.

Nota a margine: dimensione massima di swap "ufficiale". Con kernel 2.4, il limite è 64 aree di swap con un massimo di 64GB ciascuna, anche se ciò non trova riscontro nella pagina man di **mkswap**. Con opteron a 64 bit e kernel 2.6, sono permesse 128 aree di swap, ciascuna di 16 giganteschi TB! (Grazie a Peter Chubb per il calcolo).

4.4.3. Dove dovrebbe essere posto lo spazio di swap?

La risposta breve è: va bene ovunque. Tuttavia, se si è interessati ad ottenere la maggior velocità possibile, ci sono due strategie di base (oltre a comprare più RAM).

- Dividere lo spazio di swap su diverse unità o almeno metterlo sull'unità su cui si scrive di meno.
- Posizionare ogni partizione di swap nelle tracce più esterne.

Ecco le considerazioni:

- Se si ha un disco con molte testine e uno con meno testine che sono identici per ciò che riguarda gli altri parametri, il disco con molte testine sarà più veloce. Leggere dati da testine diverse è veloce, dato che è puramente elettronico. Leggere dati da tracce diverse è lento, dato che comporta uno spostamento fisico della testina.

  Ne consegue pertanto che scrivere lo swap su un'unità separata sarà più veloce che non muovere la testina avanti e indietro su un'unità singola.

- *Posizionamento*: i dischi più vecchi hanno lo stesso numero di settori in tutte le tracce. Con tali dischi sarà più veloce mettere lo swap in mezzo al disco, supposto che la testina del disco si muova da una traccia casuale verso l'area di swap.
- I dischi più recenti usano lo ZBR (zone bit recording). Hanno più settori nelle tracce esterne. Con un numero costante di rpm, questo porta a ben più alte prestazioni nelle tracce esterne che non in quelle interne. Mettere il proprio spazio di swap nelle tracce veloci. (In generale, cilindri con un numero basso sono associati a partizioni con numero basso. Comunque vedere i più recenti commenti (http://lissot.net/partition/mapping.html) di Kristian sull'argomento - Tony).
- *Uso*: naturalmente la testina del disco non si muoverà in modo casuale. Se lo spazio di swap è nel mezzo del disco tra una partizione home costantemente attiva e una quasi inutilizzata partizione di archiviazione, sarebbe meglio se l'area di swap fosse vicino alla partizione home per avere movimenti ancora più brevi della testina. Sarebbe però ancora meglio avere lo swap in un altro disco, altrimenti inutilizzato.
- *Striping*: la velocità può essere aumentata scrivendo su aree multiple di swap simultaneamente. Gli spazi di swap con la stessa priorità verranno scritti come in un RAID. Vedere la la Sezione 9.3.

Sintesi: mettere lo spazio di swap su un disco veloce con molte testine che non sia occupato a fare altro. Se si hanno dischi multipli: dividere lo swap e spargerlo su tutti i dischi o perfino su controller diversi.

5. Partizionare con fdisk

Questa sezione spiega in pratica come partizionare il proprio disco fisso con l'utilità **fdisk**. Linux consente solo 4 partizioni primarie. Si può avere un numero molto più grande di partizioni logiche dividendo ulteriormente una delle partizioni primarie. Solo una delle partizioni primarie può essere suddivisa.

Esempi:

1. Quattro partizioni primarie (vedere la la Sezione 5.2)
2. Partizioni primarie e logiche coesistenti (vedere la la Sezione 5.3)

5.1. Uso di fdisk

Per avviare **fdisk** digitare (come superutente) **`fdisk`** ***`device`*** al prompt dei comandi. *`device`* potrebbe essere qualcosa del tipo `/dev/hda` o `/dev/sda` (vedere la la Sezione 2.1.1). I comandi base di **fdisk** di cui si ha bisogno sono:

`p` visualizza la tabella delle partizioni

`n` crea una nuova partizione

`d` cancella una partizione

`q` esce senza salvare i cambiamenti fatti

`w` scrive la nuova tabella delle partizioni ed esce

I cambiamenti che vengono fatti alla tabella delle partizioni non hanno alcun effetto fino a che non si dà il comando di scrittura (w). Ecco un esempio di tabella delle partizioni:

```
Disk /dev/hdb: 64 heads, 63 sectors, 621 cylinders
Units = cylinders of 4032 * 512 bytes

   Device Boot    Start       End    Blocks   Id  System
/dev/hdb1   *         1       184    370912+  83  Linux
/dev/hdb2           185       368    370944   83  Linux
/dev/hdb3           369       552    370944   83  Linux
/dev/hdb4           553       621    139104   82  Linux swap
```

La prima riga mostra la geometria del disco fisso. Può non essere accurata fisicamente, ma la si può accettare come se lo fosse. Il disco fisso in questo esempio è composto da 32 piatti a doppia faccia con una testina per ogni lato (probabilmente non è vero). Ogni piatto ha 621 tracce concentriche. Una traccia a 3 dimensioni (la stessa traccia su tutti i dischi) viene chiamata un cilindro. Ogni traccia è divisa in 63 settori. Ogni settore contiene 512 byte di dati. Perciò la dimensione dei blocchi nella tabella delle partizioni è di 64 testine * 63 settori * 512 byte ahemmm... diviso per 1024. (Vedere 4 per un esame dei problemi che questo calcolo comporta.) I valori "start" (inizio) e "end" (fine) sono espressi in cilindri.

5.2. Quattro partizioni primarie

In breve

Decidere la dimensione dello spazio di swap (vedere la la Sezione 4.4) e dove esso dovrà essere (vedere la la Sezione 4.4.3). Dividere lo spazio restante nelle tre altre partizioni.

Esempio

Avviare fdisk dal prompt della shell:

```
# fdisk /dev/hdb
```

si è indicato che si sta usando la seconda unità sul controller IDE. (Vedere la la Sezione 2.1.) Quando si visualizza la tabella delle partizioni (vuota), si ottengono solo le informazioni sulla configurazione.

```
Command (m for help): p

Disk /dev/hdb: 64 heads, 63 sectors, 621 cylinders
Units = cylinders of 4032 * 512 bytes
```

Si sapeva di avere un disco di 1.2GB, ma ora lo si sa davvero: 64 * 63 * 512 * 621 = 1281982464 byte. Si decide di riservare 128MB di tale spazio per lo swap, lasciandone 1153982464. Se si usa una delle partizioni primarie per lo swap, significa che ne rimangono ancora tre per le partizioni ext2. Dividendo in maniera uguale lo spazio, ciò significa 384MB per partizione. Al lavoro.

```
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-621, default 1):<RETURN>
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-621, default 621): +384M
```

Successivamente impostare la partizione che si vuole usare per lo swap:

```
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
```

```
p
Partition number (1-4): 2
First cylinder (197-621, default 197):<RETURN>
Using default value 197
Last cylinder or +size or +sizeM or +sizeK (197-621, default 621): +128M
```

Adesso la tabella delle partizioni ha questo aspetto:

```
   Device Boot    Start       End    Blocks   Id  System
/dev/hdb1             1       196    395104   83  Linux
/dev/hdb2           197       262    133056   83  Linux
```

Impostare le ultime due partizioni restanti come si è fatto per la prima. Infine, rendere avviabile la prima partizione:

```
Command (m for help): a
Partition number (1-4): 1
```

E rendere la seconda partizione una partizione di tipo swap:

```
Command (m for help): t
Partition number (1-4): 2
Hex code (type L to list codes): 82
Changed system type of partition 2 to 82 (Linux swap)
Command (m for help): p
```

Il risultato finale:

```
Disk /dev/hdb: 64 heads, 63 sectors, 621 cylinders
Units = cylinders of 4032 * 512 bytes

   Device Boot    Start       End    Blocks   Id  System
/dev/hdb1   *         1       196    395104+  83  Linux
/dev/hdb2           197       262    133056   82  Linux swap
/dev/hdb3           263       458    395136   83  Linux
/dev/hdb4           459       621    328608   83  Linux
```

Da ultimo, dare il comando di scrittura (w) per scrivere la tabella sul disco.

Argomenti correlati:

- la Sezione 10.2
- la Sezione 10.1
- la Sezione 10.3

5.3. Partizioni primarie e logiche coesistenti

In breve: creare e usare una delle partizioni primarie per ospitare tutte le partizioni extra. Creare quindi le partizioni logiche al suo interno. Creare le altre partizioni primarie prima o dopo la creazione delle partizioni logiche.

Esempio

Avviare fdisk dal prompt della shell:

```
# fdisk /dev/sda
```

si è indicato che si sta usando la prima unità nella catena SCSI. (Vedere la la Sezione 2.1).

Dapprima decidere quante partizioni si vogliono. Si sa che l'unità ha una capacità di 183GB e si vogliono partizioni da 26GB (magari perché si hanno nastri di back-up all'incirca di quelle dimensioni).

```
183GB / 26GB = ~7
```

perciò saranno necessarie 7 partizioni. Anche se fdisk accetta dimensioni delle partizioni espresse in MB e KB, si decide di calcolare il numero dei cilindri che verrà contenuto in ciascuna partizione, dato che fdisk mostra le posizioni di inizio e fine in cilindri. Avviando fdisk si vede che si hanno 22800 cilindri.

```
> The number of cylinders for this disk is set to 22800.  There is
> nothing wrong with that, but this is larger than 1024, and could in
> certain setups cause problems with: 1) software that runs at boot
> time (e.g., LILO) 2) booting and partitioning software from other
> OSs  (e.g., DOS FDISK, OS/2 FDISK)
```

Allora, 22800 cilindri totali divisi per sette partizioni fa 3258 cilindri. Ogni partizione sarà all'incirca grande 3258 cilindri. Ignorare il messaggio di avvertimento poiché non si sta usando l'unità di avvio. (la Sezione 4).

Dato che si hanno 4 partizioni primarie, 3 di esse possono essere grandi 3258 cilindri. La partizione estesa dovrà essere grande (4 * 3258), o 13032, cilindri per poter contenere le 4 partizioni logiche.

Inserire i seguenti comandi per impostare la prima delle 3 partizioni primarie (ciò che viene inserito dall'utente è mostrato in grassetto):

```
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-22800, default 1): <RETURN>
```

```
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-22800, default 22800): 3258
```

L'ultima partizione è la partizione estesa:

```
Partition number (1-4): 4
First cylinder (9775-22800, default 9775): <RETURN>
Using default value 9775
Last cylinder or +size or +sizeM or +sizeK (9775-22800, default 22800): <RETURN>
Using default value 22800
```

Il risultato, quando si usa il comando di visualizzazione è:

```
/dev/sda1             1      3258  26169853+  83  Linux
/dev/sda2          3259      6516  26169885   83  Linux
/dev/sda3          6517      9774  26169885   83  Linux
/dev/sda4          9775     22800 104631345    5  Extended
```

In seguito dividere la partizione estesa in 4 partizioni logiche, iniziando dalla prima partizione logica, facendo segmenti di 3258 cilindri. Le partizioni logiche partono automaticamente da /dev/sda5.

```
Command (m for help):  n
First cylinder (9775-22800, default 9775): <RETURN>
Using default value 9775
Last cylinder or +size or +sizeM or +sizeK (9775-22800, default 22800): 13032
```

Il risultato finale è:

```
   Device Boot    Start       End    Blocks   Id  System
/dev/sda1             1      3258  26169853+  83  Linux
/dev/sda2          3259      6516  26169885   83  Linux
/dev/sda3          6517      9774  26169885   83  Linux
/dev/sda4          9775     22800 104631345    5  Extended
/dev/sda5          9775     13032  26169853+  83  Linux
/dev/sda6         13033     16290  26169853+  83  Linux
/dev/sda7         16291     19584  26459023+  83  Linux
/dev/sda8         19585     22800  25832488+  83  Linux
```

Da ultimo, inserire il comando di scrittura (w) per scrivere la tabella sul disco. Per rendere le partizioni utilizzabili, si dovrà formattare (la Sezione 10.1) ciascuna partizione e poi montarla (la Sezione 10.3).

5.4. Esempi presentati

Vorrei presentare l'organizzazione delle mie partizioni, perché funziona bene con qualunque distribuzione Linux (anche quelle grandi basate su RPM). Ho un disco fisso che [...] è 10 giga, esattamente. Windows non può vedere più di 9.3 giga di esso, ma Linux lo vede tutto e lo utilizza tutto. Esso ha anche molto più di 1024 cilindri.

Tabella 7. Esempio di tabella delle partizioni

Partizione	Punto di mount	Dimensione
/dev/hda1	/boot	(15 mega)
/dev/hda2	partizione windows 98	(2 giga)
/dev/hda3	estesa	(N/D)
/dev/hda5	spazio di swap	(64 mega)
/dev/hda6	/tmp	(50 mega)
/dev/hda7	/	(150 mega)
/dev/hda8	/usr	(1.5 giga)
/dev/hda9	/home	(resto del disco)

Io faccio test sui nuovi kernel per le memorie di massa USB e questo spiega la grande partizione /boot. Installo LILO nell'MBR e in modo predefinito avvio windows (non sono l'unico ad usare questo computer).

Ho anche notato che non ci sono esempi REALI di tabelle di partizioni e per i novizi suggerisco FORTEMENTE di metterne più di uno. Sono appena uscito dalla stadio di novellino e il partizionamento è stato l'aspetto che più mi ha confuso.

Valkor (mailto:valkor@qx.net)

6. Etichette

In Linux, ci si riferisce ai dischi fissi come a device e i device sono pseudo-file in `/dev`. Per esempio, la prima partizione della seconda unità SCSI con numero più basso è `/dev/sdb1`. Se l'unità a cui si fa riferimento con `/dev/sda` viene rimossa dalla catena, allora la partizione suddetta è automaticamente rinominata in `/dev/sda1` all'avvio successivo.

6.1. Etichette di volume

Le etichette di volume rendono possibile mantenere per le partizioni un nome costante indipendentemente dal punto in cui sono connesse e indipendentemente da cosa altro sia connesso. Le etichette non sono obbligatorie per volumi Linux. Ognuna può essere lunga al massimo 16 caratteri.

Ci sono tre strumenti per creare etichette di volume: mke2fs, tune2fs e e2label.

6.1.1. Esecuzione semplice

e2label `/dev/hdb1 pubsw`

tune2fs -L `pubsw /dev/hdb1`

Entrambi questi comandi mettono l'etichetta "pubsw" alla prima partizione della seconda unità. Quella etichetta rimane con quella particolare partizione anche se l'unità viene spostata su un altro controller o perfino su un altro computer.

mke2fs `pubsw /dev/hdb1`

mke2fs -L `pubsw /dev/hdb1`

fanno la stessa cosa dei due comandi precedenti, dopo aver creato il file system. Ciò significa che ciascuno di quest'ultimi due comandi cancella ogni dato esistente nella partizione.

6.1.2. Istruzioni per l'uso

Ecco un esempio di fstab. Esso è un file di testo che si trova in /etc, che viene di solito configurato durante l'installazione del sistema operativo. Descrive dove sarà montata ciascuna partizione e come verrà montata. Può essere modificato o usando un'utilità o manualmente quando si aggiungono o rimuovono device.

```
LABEL=/        /                    ext3    defaults        1 1
LABEL=/boot    /boot                ext2    defaults        1 2
none           /dev/pts             devpts  gid=5,mode=620  0 0
none           /dev/shm             tmpfs   defaults        0 0
LABEL=HOME     /home                ext3    defaults        1 2
none           /proc                proc    defaults        0 0
none           /sys                 sysfs   defaults        0 0
LABEL=/usr     /usr                 ext3    defaults        1 2
/dev/hdc1      /k-space             ext3    defaults        1 2
/dev/hda6      swap                 swap    defaults        0 0
/dev/hdd       /media/cdrecorder    auto    pamconsole,ro,exec,noauto,managed 0 0
/dev/fd0       /media/floppy        auto    pamconsole,exec,noauto,managed 0 0
```

La prima colonna a sinistra elenca i device e la seconda i punti di mount. Questo esempio contiene un misto di device ed etichette. L'unità master del secondo controller viene sempre montata in `/k-space`. La partizione con etichetta "HOME" è sempre montata in `/home`, indipendentemente dalla unità sulla quale si trovi o da quale numero di partizione abbia. Notare che è possibile usare punti di mount come etichette, come "/usr".

6.2. Etichette di device

devlabel è uno script che crea collegamenti simbolici a device. Per esempio,

devlabel -d `/dev/hdb1` -s `/dev/home`

crea un collegamento da `/dev/hdb1` a `/dev/home`. È fondamentale come archivi un identificatore univoco per l'hardware che si trova su `/dev/hdb1` e memorizzi quell'identificatore, insieme al nome del collegamento che è stato specificato, in /etc/sysconfig/devlabel. Se l'hardware è successivamente spostato in `/dev/hdc1`, il suo identificatore univoco sarà interrogato (usando /usr/bin/partition_uuid), verrà trovata la voce corrispondente in /etc/sysconfig/devlabel e sarà nuovamente collegato a `/dev/home`.

7. Formattare una partizione ext2/3

Quando un disco fisso viene partizionato, ne viene creata una mappa con sezioni, ma le sezioni sono vuote. È come una biblioteca appena costruita; devono essere sistemate mensole, segnali e un sistema di catalogazione prima di mettere via i libri.

La struttura dell'organizzazione interna di una partizione è chiamata file system. In Linux, i file system standard sono ext2 e ext3. Il file system ext3 è come ext2, più un registro delle operazioni di scrittura su disco chiamato journal (diario). Il journal permette al sistema, tra le altre cose, di riprendersi facilmente da mancanze accidentali di corrente elettrica.

Il principale strumento per creare un file system ext2/3 in una partizione è **mke2fs**. Si trova di solito in `/sbin`. **mkfs.ext2** e **mkfs.ext3** sono frontend che passano opzioni specifiche a **mke2fs**.

7.1. Semplice esecuzione

mke2fs `/dev/hdb1`

mkfs.ext2 `/dev/hdb1`

entrambi creano un file system ext2 sulla prima partizione della seconda unità e

mke2fs -j `/dev/hdb1`

mkfs.ext3 `/dev/hdb1`

creano un file system ext3.

7.2. Blocchi riservati

L'opzione -m è probabilmente quella di maggiore utilità per i non esperti. Se il file system diventa pieno e non c'è più spazio per scrivere, è praticamente inutilizzabile perché il sistema operativo è perennemente in fase di scrittura su disco. Come valore predefinito, il cinque percento della partizione è riservato per l'uso da parte del superutente. Ciò permette al superutente di fare attività di amministrazione sulla partizione e forse di spostare via alcuni dati. Questo, tuttavia, è cruciale soprattutto quando la partizione contiene le directory / o home. Per partizioni puramente di dati, questo risulta solo spazio sprecato. Il cinque percento di una partizione da 250Gb è 12.5 Gb. Specialmente nel caso di grandi partizioni, si può impostare tranquillamente lo spazio riservato al minimo, che è l'uno percento.

mkfs.ext3 -m 1 `/dev/hdb1`

crea un file system con solo l'1% dello spazio riservato per il superutente. Si può usare tune2fs -m per ritoccare i blocchi riservati dopo che siano stati caricati dati sulla partizione.

8. Ripristinare una tabella delle partizioni cancellata

Di seguito sono indicate le istruzioni per recuperare manualmente una tabella delle partizioni cancellata. Ci sono utilità quali gpart (http://www.stud.uni-hannover.de/user/76201/gpart) o TestDisk (http://www.cgsecurity.org/index.html?testdisk.html) che possono rendere questo compito considerevolmente più facile. Se tuttavia si sta leggendo questo documento perché si è stati sfortunati, questo è ciò che si deve fare:

1. Creare una partizione che sia grande almeno quando lo era la prima partizione. La si può creare più grande, di un valore qualsiasi, della partizione originale. Se si sottostima il valore, ci sarà solo dolore e stridore di denti.

   ```
   Command (m for help): n
   Command action
      e   extended
      p   primary partition (1-4)
   P
   Partition number (1-4): 1
   First cylinder (1-23361, default 1): <RETURN>
   Using default value 1
   Last cylinder or +size or +sizeM or +sizeK (1-22800, default 22800): 13032

   Command (m for help): w
   ```

2. Eseguire **dumpe2fs** sulla prima partizione e con grep estrarre il conteggio dei blocchi.

Esempio:

```
% dumpe2fs /dev/sda1 | grep "Block count:"
Block count:              41270953
```

Se non si è sicuri di questo valore, ripetere il passo 1 con una dimensione di partizione maggiore. Se il conteggio dei blocchi cambia, allora si era sottostimata la dimensione della partizione originale. Ripetere il passo 1 fino a che non si ottiene un conteggio dei blocchi costante.

3. Rimuovere la partizione che si è appena creata

```
Command (m for help): d
Partition number (1-4): 1
```

4. Creare una nuova partizione con esattamente la dimensione che si è ottenuta dal conteggio dei blocchi. Dato che non si può inserire la dimensione in blocchi in fdisk, è necessario calcolare quanti cilindri indicare. Ecco la formula:

```
(numero di cilindri necessari) = (numero dei blocchi) / (dimensione dei blocchi)

(dimensione dei blocchi) = (dimensione dell'unità) / 1024

(dimensione dell'unità) = (numero delle testine) * (numero dei settori/cilindro) * (numero di byte/settore)
```

Si consideri l'esempio seguente, in cui un disco fisso è stato partizionato in quattro partizioni primarie di 1, 2, 4 e 8 cilindri.

```
disk /dev/sda: 16 heads, 63 sectors, 23361 cylinders
Units = cylinders of 1008 * 512 bytes

   Device Boot    Start       End    Blocks   Id  System
/dev/sda1             1         2       976+  83  Linux
/dev/sda2             3         5      1512   83  Linux
/dev/sda3             6        10      2520   83  Linux
/dev/sda4            11        19      4536   83  Linux
```

fdisk fornisce nella testata del suo output le informazioni sulla configurazione di cui si ha bisogno. La dimensione dell'unità è `516096` (`16` testine * `63` settori/cil * `512` byte/settore). La dimensione dei blocchi è `504` (`516096 / 1024`). Il numero dei cilindri necessari per la seconda partizione è pertanto `3` (`1512` blocchi `/ 504`). La tabella delle partizioni mostra che è proprio così: il primo cilindro è il `3`, il secondo il `4` e l'ultimo è il `5`, per un totale di tre cilindri. Il numero dei cilindri necessari per la terza partizione è calcolato in modo simile: `2520` blocchi `/ 504 = 5`, il che corrisponde ai cilindri `6,7,8,9,10`. Si noti che questo calcolo non funziona per la prima partizione, perché il conteggio dei blocchi è sbagliato (`976` invece di `1008`). Il segno più indica che non tutti i blocchi sono inclusi nel valore di fdisk. Quando si tenta di fare il calcolo (`976 / 504`) si ottiene `1.937`. Sapendo che il numero dei cilindri deve essere un intero, si può semplicemente arrotondare per eccesso.

5. Eseguire **e2fsck** su di essa per verificare che sia possibile leggere la nuova partizione.

6. Ripetere i passi da 1 a 5 per le partizioni restanti.

Montare nuovamente le partizioni. Sorprendentemente, tutti i propri dati saranno presenti.

Il merito va a: Mike Vevea, jedi sys admin, per aver fornito la strategia di base.

9. Configurare lo spazio di swap

9.1. File di swap

Normalmente ci sono solo due passaggi da eseguire per configurare lo spazio di swap, creare la partizione ed aggiungerla ad /etc/fstab. Una voce tipica in fstab per una partizione swap su /dev/hda6 sarà di questo tipo:

```
/dev/hda6 swap swap defaults 0 0
```

Al successivo avvio, lo script di inizializzazione la attiverà automaticamente e non è necessario fare altro.

Tuttavia, se si vuole usarla subito, è necessario attivarla manualmente. Da superutente digitare:

```
mkswap -f /dev/hda6
swapon /dev/hda6
```

9.2. File di swap

Ci potranno essere momenti in cui si è a corto di spazio di swap e non è pratico ripartizionare un'unità o aggiungerne uno nuovo. In questo caso si può utilizzare un file regolare in una partizione normale. Tutto ciò che si deve fare è creare un file della dimensione voluta

```
dd if=/dev/zero of=/var/mio_swap bs=1024 count=131072
```

ed attivarlo

```
 mkswap -f /var/mio_swap
 swapon /var/mio_swap
```

Con questa sintassi si crea un file chiamato `mio_swap` in `/var`. È grande 128 Mb (128 x 1024 = 131072). Inizialmente è riempito da zeri. **mkswap**, tuttavia, lo marca come spazio di swap e **swapon** dice al kernel di iniziare ad usarlo some spazio di swap. Quando non serve più, digitare

```
swapoff /var/mio_swap
```

```
rm /var/mio_swap
```

9.3. Aree di swap multiple

Sullo stesso sistema può essere utilizzata più di una partizione di swap. Si consideri un fstab di esempio con una sola partizione di swap:

```
/dev/hda5   /        ext3   defaults        1 1
/dev/hda1   /boot    ext2   defaults        1 2
none        /dev/pts devpts gid=5,mode=620  0 0
none        /proc    proc   defaults        0 0
/dev/hda7   /usr     ext3   defaults        1 2
/dev/hda6   swap     swap   defaults        0 0
```

Si immagini di rimpiazzare la voce riguardante la partizione di swap con queste tre righe:

```
/dev/hda6   none    swap    sw,pri=3    0 0
/dev/hdb2   none    swap    sw,pri=2    0 0
/dev/hdc2   none    swap    sw,pri=1    0 0
```

Questa configurazione farà sì che il kernel usi per prima /dev/hda6 che ha associata la più alta priorità (pri=3). La massima priorità consentita è 32767 e la minima 0. Se quello spazio fosse insufficiente, il kernel inizierebbe ad usare /dev/hdb2 e dopo di essa /dev/hdc2. Perché una simile configurazione? Si immagini di dare la priorità più alta alle unità più nuove (più veloci); ciò minimizzerebbe la perdita di velocità al crescere dello spazio di swap.

È possibile scrivere su tutte e tre contemporaneamente. Se ognuna ha la stessa priorità, il kernel scriverà su di esse come in un RAID, con un aumento della velocità proporzionale.

```
/dev/hda6   none   swap   sw,pri=3   0   0
/dev/hdb2   none   swap   sw,pri=3   0   0
/dev/hdc2   none   swap   sw,pri=3   0   0
```

Si noti che queste tre partizioni sono su unità separate, il che è l'ideale in termini di miglioramento della velocità.

10. Appendice

10.1. Formattare le partizioni

Al prompt della shell, iniziare creando i file system sulle partizioni. Proseguendo l'esempio nella la Sezione 5.3, si tratta di fare:

```
# mke2fs /dev/sda1
```

È necessario farlo per ognuna delle partizioni, ma non per `/dev/sda4` (la partizione estesa). Linux supporta altri tipi di file system oltre a ext2. Si può vedere quali siano supportati dal proprio kernel guardando in: `/usr/src/linux/include/linux/fs.h`

I file system più comuni possono essere creati con i programmi in `/sbin` che iniziano con le lettere "mk" come **mkfs.msdos** e **mke2fs**.

10.2. Attivare lo spazio di swap

Per impostare una partizione di swap:

```
# mkswap -f /dev/hda5
```

Per attivare l'area di swap:

```
# swapon  /dev/hda5
```

Normalmente l'area di swap è attivata dagli script di inizializzazione all'avvio del sistema.

10.3. Montare le partizioni

Montare una partizione significa unirla al file system Linux. Per montare una partizione linux:

```
# mount -t ext2 /dev/sda1 /opt
```

-t ext2

Tipo di file system. Altri tipi comunemente usati sono:

- ext3 (file system con journaling basato su ext2)
- msdos (DOS)
- hfs (mac)
- iso9660 (CDROM)
- nfs (network file system)

/dev/sda1

Nome del device. Altri nomi di device comunemente usati:

- `/dev/hdb2` (seconda partizione nella seconda unità IDE)

- `/dev/fd0` (unità floppy A)
- `/dev/cdrom` (CDROM)

/opt

Punto di mount. Qui è dove si desidera "vedere" la propria partizione. Quando si digita **`ls /opt`**, si può vedere cosa ci sia in `/dev/sda1`. Se ci sono già alcune directory o file dentro a `/opt`, essi saranno invisibili dopo questo comando mount.

10.4. Alcuni dati sui file system e la frammentazione

Lo spazio su disco è gestito dal sistema operativo in unità di blocchi e frammenti di blocchi. Nel file system ext2, i frammenti ed i blocchi devono avere la stessa dimensione, per cui si può limitare questa analisi ai blocchi.

I file sono di dimesioni disparate. Non finiscono esattamente al confine tra i blocchi. Per cui con ogni file una parte dell'ultimo blocco di ciascun file è sprecata. Supponendo che le dimensioni dei file siano casuali, c'è aprrossimativamente uno spreco di metà blocco per ogni file su disco. Tanenbaum, nel suo libro "Operating Systems", la chiama "frammentazione interna".

Si può calcolare il numero dei file sul proprio disco in base al numero degli inode allocati sul disco. In questo disco

```
# df -i
Filesystem           Inodes   IUsed   IFree  %IUsed Mounted on
/dev/hda3              64256   12234   52022    19%  /
/dev/hda5              96000   43058   52942    45%  /var
```

ci sono circa 12000 file in `/` e circa 44000 file in `/var`. Con una dimensione dei blocchi di KB, circa 6+22 = 28 MB di spazio su disco sono persi nei blocchi di coda dei file. Se si fosse scelta una dimensione dei blocchi di 4 KB, si sarebbe perso una spazio 4 volte più grande.

Il trasferimento di dati, però, è più veloce per grossi pezzi di dati contigui. È per questo che ext2 cerca di preallocare spazio in unità di 8 blocchi contigui per i file in crescita. Lo spazio preallocato inutilizzato viene liberato quando il file è chiuso, perciò non c'è spreco di spazio.

Il posizionamento in blocchi non contigui di un file è un male per le prestazioni, dato che ai file si accede spesso in maniera sequenziale. Esso costringe il sistema operativo a dividere l'accesso su disco e il disco a muovere la testina. Questa è chiamata "frammentazione esterna" o semplicemente "frammentazione" ed è un problema comune con i file system MS-DOS. Uniti alla orribile buffer cache usata da MS-DOS, gli effetti della frammentazione dei file sulle prestazioni sono molto evidenti. Gli utenti DOS sono abituati a deframmentare i propri dischi a intervalli di poche settimane ed alcuni hanno sviluppato alcune credenze ritualistiche riguardanti la deframmentazione.

Nessuna di queste abitudini dovrebbe essere trasposta in Linux ed ext2. I file system nativi Linux non necessitano di deframmentazione nell'uso comune e quest'ultimo include ogni situazione con almeno 5% dello spazio su disco libero. C'è uno strumento di deframmentazione per ext2 chiamato defrag, ma gli utenti sono avvertiti di non usarlo alla leggera. Una mancanza di corrente durante una tale operazione può rovinare il file system. Dato che si deve comunque fare il backup dei propri dati, basterà semplicemente riscrivere tutto dalla propria copia.

Il file system MS-DOS è noto anche per la perdita di grosse quantità di spazio su disco a causa della frammentazione interna. Per partizioni più grandi di 256 MB, la dimensione dei blocchi in DOS diviene così grande che essi non sono più utili (questo è stato in parte corretto con FAT32). Ext2 non costringe a scegliere blocchi grossi per file system grandi, tranne che per file system estremamente grandi dell'ordine di 0.5 TB (cioé terabyte con 1 TB che equivale a 1024 GB) o superiore, dove blocchi di piccole dimensioni diventano inefficienti. Perciò, a differenza di quanto accade con il DOS, non è necessario dividere dischi grandi in partizioni multiple per mantenere bassa la dimensione dei blocchi.

Usare una dimensione dei blocchi di 1Kb se si hanno molti file piccoli. Per partizioni grandi, vanno bene blocchi di 4Kb.