Lezione 6 - Le distribuzioni, la documentazione del sistema, la pacchettizzazione del software

12 dicembre 2006

Alessio Checcucci

Riccardo Aldinucci

Q.It Universita' degli Studi di Siena

1 Le distribuzioni GNU/Linux

Una distribuzione (*distro*) in pratica è un confezionamento (*packaging*) di Linux, con procedure che semplificano l'installazione e con una raccolta di software, utility e tools a completamento di un pacchetto che si possa definire sistema operativo, cosi come è comunemente conosciuto.

Linux (Linux di Linus Torvalds per intenderci) di per se, difatti non è altro che il semplice kernel, le stesse utility di base e di gestione dei files sono state aggiunte in seguito dal progetto GNU. Per questo dovremmo sempre chiamarlo *GNU/Linux* e non riduttivamente solo "Linux". In definitiva, quando parliamo di Linux come sistema operativo, stiamo facendo una addizione: *Linux* (kernel) + *GNU* (utility di base) + *Distro* (a scelta).

Una distribuzione viene scelta sulla base delle necessità dell'utente, oltre che sull'esperienza dello stesso. Ogni distro difatti ha caratteristiche diverse, pensate proprio per soddisfare tipologie di utenti differenti. Le distribuzioni differiscono per:

- Numero e versioni dei programmi installabili
- Versione del kernel utilizzata e modalita' di pre-installazione
- Procedura di installazione (interfaccia utente e possibilità di definire opzioni e scegliere quale software installare)
- Organizzazione di file di configurazione, programmi, log nel file system
- Configurazioni predefinite del software installato
- Tipo di supporto tecnico e di manuali a disposizione

Ci sono a questo momento più di 300 progetti di distribuzioni linux in attivo sviluppo, revisione e miglioramento che si differenziano per scelte progettuali, come i vari software di mantenimento del sistema per l'installazione, la rimozione e la configurazione del software.

Prima della comparsa delle distribuzioni, chiunque volesse far uso di Linux doveva necessariamente essere un esperto di Unix, con la consapevolezza non solo di quali librerie ed eseguibili fossero necessari per far partire e funzionare il sistema, ma anche di alcuni importanti dettagli riguardanti la configurazione ed il posizionamento dei files nel sistema stesso.

Le distribuzioni Linux apparvero poco tempo dopo che il kernel Linux iniziò ad essere utilizzato da persone al di fuori del team originale di sviluppatori Linux. Questi ultimi infatti erano più interessati a sviluppare il sistema operativo piuttosto che programmi applicativi, interfacce utente, e comode pacchettizzazioni.

Una tipica distribuzione Linux comprende un kernel Linux, librerie e strumenti GNU, software addizionale, documentazione, un server grafico, un window manager, ed un desktop environment. La quasi totalità del software incluso è FOSS (Free and Open Source Software - Software libero Open Source) che è distribuito dagli sviluppatori sia in forma precompilata che come codice sorgente, cosa che permette agli utenti di modificare e ricompilare il codice originale a proprio piacimento. Una parte del software incluso in alcune distribuzioni può essere proprietario e non disponibile sotto forma di sorgenti.

Molte distro forniscono un sistema di installazione simile a quello di altri moderni sistemi operativi. Distribuzioni self-hosting come Gentoo Linux forniscono il codice sorgente di tutto il software ed includono le versioni eseguibili solo di un kernel base, compilatore e programma di installazione; questo programma compila tutto il software per la specifica architettura del computer dell'utente.

1.1 Le distribuzioni Live

Un LiveCD o Live CD è un CD-ROM contenente un sistema operativo in grado di essere avviato ed eseguito senza doverlo installare su un hard disk. Si utilizza per scopi dimostrativi, didattici o per avere a disposizione un sistema operativo completo da usare su un computer altrui.

Esistono versioni dimostrative LiveCD di sistemi operativi commerciali come QNX o BeOS ma la maggior diffusione di questo tipo di supporto si ha per il sistema operativo GNU/Linux. In quest'ultimo caso evitare la fase di installazione è un grande vantaggio per gli utenti alle prime armi che possono prendere confidenza con Linux senza paura di combinare danni.

Oltre agli scopi educativi, i LiveCD possono essere utili per effettuare un backup dei dati anche se il sistema operativo installato sul disco fisso non riesce più ad eseguire il boot.

Le distribuzioni live operano in base a tre metodi di funzionamento principali:

- Le piu' semplici fanno il boot da un CD o DVD, ovvero il kernel e' presente sul supporto rimovibile stesso e il file system (ISO) presente su di essi e' del tipo *El Torito*, ovvero che presenta le estensioni per il boot. A termine del caricamento del kernel viene creato un ramdisk, noto come initial ramdisk e all'interno viene scompattata l'immagine compressa di un intero (seppur piccolo esemplice) root filesystem. Il nome del file contenente il filesystem puo' essere specificato al boot o nella configurazione del boot loader. Il filesystem che risiese sulla initrd viene montato come root filesystem, su di esso sara' presente uno script di nome *linuxrc* che verra' mandato in esecuzione. Esso si preoccupa di accedere ad un file contenente l'immagine compressa di uno (o piu') filesystem, molto piu' ricchi del precedente. Questo file viene decompresso in un ramdisk di dimensioni opportune e montato come root filesystem definitivo del sistema. A questo punto il CD o DVD puo' essere smontato ed il sistema lavorera' completamente in ram.
  Questo metodo e' stato mutuato dai sistemi embedded in cui e' in ampio uso. Naturalmente e' molto flessibile ed efficiente, ma viene limitato dalla dimensione della RAM disponibile. Quindi solo sistemi piuttosto semplici possono essere implementati. Il problema di fondo e' che il ramdisk alloca uno spazio statico nella memoria di sistema e quindi la sua dimensione viene sottratta da quella che il kernel puo' allocare per le proprie esigenze e per i processi in userspace.

- La maggior parte delle live distribution segue l'approccio di kernel+initrd appena visto, ma invece di creare un nuovo ramdisk, utilizzano la capacita' del kernel di montare un file ISO (iso9660) compresso che risiede su un CD o DVD come proprio filesystem. Esso puo' essere root filesystem,

oppure puo' essere montato sotto qualche altro mount point e poi le proprie directory saranno collegate (symbolic link) nei punti opportuni del root filesystem.
Questo approccio comporta lo svantaggio che l'immagine del filesystem risiede su un dispositivo read-only, per cui su di esso non e' possibile scrivere. Per oltrepassare questo limite, le directory in cui si dovrebbe avere accesso in scrittura (tipicamente /home e /var) vengono create su filesystem separati, supportati da uno o piu' ramdisk.

- L'approccio piu' moderno alle Live distribution prevede l'utilizzo di uno speciale filesystem: *UnionFS*. Esso non e' ancora stato ufficialmente inserito nel kernel, ma viene gia' utilizzato da almeno due delle piu' diffuse Live. UnionFS e' un filesystem driver che si registra nel VFS, ma non puo' gestire da solo uno spazio di memoria secondaria, e viene utilizzato per "sovrapporre" una serie di directory su filesystem presenti (in questo caso) sul CD o DVD.
  UnionFS in questo senso costituisce un ulteriore layer che si frappone fra i driver dei filesystem e il VFS.

UnionFS In UNIX, come sappiamo, "tutto e' un file". Ogni filesystem ha un'interfaccia diversa per la gestione dello spazio di memorizzazione, per cui e' stato introdotto un layer ulteriore che presenta un'interfaccia unica verso lo userspace (il VFS). Quando il kernel riceve una *system call* che riguarda un file, chiama una funzione del VFS, che a sua volta aggiorna le sue strutture interne e poi esegue l'operazione facendo riferimento al corretto driver.

Fra i filesystem che sottostanno al VFS possono anche non esserci dei filesystem reali, come UnionFS.

UnionFS e' uno stackable filesystem, ovvero un modulo del kernel che si frappone fra il VFS e i filesystem reali in modo trasparente. Nei confronti del VFS presenta la stessa interfaccia dei comuni filesystem, e si limita a redirigere le chiamate verso i filesystem reali. UnionFS utilizza questo stratagemma per presentare un filesystem unificato, nascondendo al VFS i dettagli necessari a realizzarlo.

Il filesystem unificato viene ottenuto dall'unione di un qualsiasi numero di directory diverse (dette *branch* o rami), tenendo il loro contenuto fisico separato. Ciascuna di essere puo' appartenere ad un filesystem montato sia in sola lettura che in lettura/scrittura (caratteristica che viene sfruttata al massimo dalla Live distro).

Per montare un filesystem UnionFS e' allora sufficiente specificare una linea del tipo:

mount -t unionfs -o dirs=/dir1=rw:/dir2=ro ... none mount_point

I branch vengono valutati da sinistra a destra in ordine di priorita', per cui le directory presenti nei branch a priorita' piu' alta mascherano le altre in caso di omonimie nel path.

UnionFS permette (tramite l'utilita' unionctl) di aggiungere e rimuovere le directory a runtime.

L'utilita' di UnionFS si esplicita ancora di piu' nel meccanismo di *copy-up*. Ovvero quando un file debba essere modificato, e esso sia contenuto in un branch read-only, UnionFS va a verificare se nei branch di livello superiore la struttura gia' esista. Se la struttura path del file non esiste, essa viene creata (sempre che

il branch sia read-write), andando a mascherare (priorita' piu' elevata) il branch originale.

Questo metodo permette di poter modificare ogni file di una Live distribution e viene largamente utilizzato, sebbene UnionFs sia ancora nella fase beta e che qualche bug abbia dato qualche grattacapo.

2 La documentazione del sistema

GNU/Linux impone all'amministratore di sistema la conoscenza di tutte le questioni che riguardano il sistema, per poter fare un'amministrazione oculata. Per fortuna le fonti di documentazione sono moltissime e molto ricche, fornite per la maggior parte dalla comunita' Open Source.

2.1 Documentazione Locale

2.1.1 Man page

Su tutti i sistemi Unix e' presente il manuale in linea *man*. Le pagine di manuale rappresentano la documentazione ufficiale dei sistemi Unix e contengono una descrizione dettagliata di tutti i comandi. Per consultarle e' sufficiente digitare il comando:

man nome_comando

dove *nome_comando* e' qualsiasi comando sul quale si desidera avere maggiori informazioni. Il manuale e' diviso in 8 sezioni principali:

1. *Comandi e programmi applicativi*
2. *Chiamate di sistema*
3. *Subroutine*
4. *Formati di file*
5. *Varie*
6. *Giochi*
7. *File speciali*
8. *Procedure di gestione del sistema*

Ciascuna pagina di manuale e' divisa in varie parti. Una parte *name* dove viene specificato il nome del comando e una breve descrizione del suo scopo. Una parte *synopsys* che descrive la sintassi del comando e delle sue opzioni. Una parte *description* che contiene una descrizione piu' dettagliata del comando e delle sue opzioni. Per ogni opzione viene spiegato il significato ed i suoi effetti. Una pagina del comando man viene visualizzata allo stesso modo di un file visualizzato con i comandi more e less, percio' e' possibile scorrerla in alto ed in basso usando i tasti freccia su/giu' e pagina su/giu'. Inoltre e' possibile effettuare la ricerca di una stringa nel testo della pagina usando il comando */stringa*, dove *stringa* e' la parola da cercare. In fondo alla pagina e' presente una sezione '*see also*' dove sono presenti dei rimandi a pagine di manuale che

trattano argomenti che hanno una qualche attinenza all'argomento contenuto nella pagina che si sta visualizzando. Visto che per avere maggiori informazioni su un comando specifico occorre usare il comando man nome-comando e, poiche' man a sua volta e' un comando, e' possibile saperne di piu' sul comando man digitando *man man*. Il comando / effettua ricerche di stringhe in avanti mentre il comando *?* effettua ricerche all'indietro. Per ripetere la ricerca di una stringa occorre digitare /stringa una prima volta e successivamente, per ricercare la ricorrenza successiva della stringa, occorre premere il tasto *n*. Per uscire dal comando man occorre premere il tasto *q* (quit).

2.1.2 Dove stanno gli oggetti

In un sistema Linux oltre al comando man sono presenti i comandi *whatis* e *apropos*. Questi comandi effettuano una ricerca tra i titoli del comando man e visualizzano quelli che rispecchiano i criteri di ricerca impostati.

Whatis *string* ricerca tutte le stringhe *'string'* all'interno dei titoli di man e visualizza tutti i titoli che contengono questa stringa.

Apropos e' simile a whatis, ma a differenza di quest'ultimo permette di effettuare ricerche anche con porzioni di parole. Ad esempio e' possibile digitare il comando apropos ls che trovera' corrispondenza con tutti i comandi che iniziano con ls: ls, lsattr, lsearch, lseek, lsort e via dicendo.

I comandi whatis e apropos sono molto utili per effettuare una ricerca preliminare su un comando che non si conosce oppure per scoprire entro quali contesti e' possibile reperire informazioni su tale comando. Apropos e' inoltre utile in quei casi in cui non si ricorda perfettamente il nome di un comando ma si rammentano le lettere iniziali.

Altri comandi utili per reperire informazioni sui programmi installati nel sistema sono: *whereis, which, type*. Quest'ultimo e' un built-in della Bash shell.

2.1.3 Info page

Texinfo e' il formato ufficiale per la documentazione del progetto GNU. Esso e' stato inventato da Richard Stallman and Bob Chassell molti anni fa, basandosi su Scribe di Brian Reid e altri linguaggi di formattazione del tempo. E' un formato utilizzato anche da molti progetti non GNU.

Texinfo utilizza un solo source file per produrre output in una serie di formati, sia online che stampabili (dvi, html, info, pdf, xml, etc.). Questo vuol dire che invece di scrivere diversi documenti dedicati alle informazioni online e ai formati stampabili. Anche quando sia necessario modificare un documento, e' necessario farlo solo su un file. Il sistema Texinfo e' progettato per integrarsi perfettamente con GNU Emacs.

E' possibile accedere alle info page con il comando:

info command_name

Qualora per un comando sia disponibile una pagina info, la pagina man e' in genere molto breve e contiene una nota che rimanda alla relativa pagina info.

2.1.4 Altre fonti locali

Sui sistemi Linux sono inoltre presenti i documenti *HOWTO* (come fare) che sono dei file di testo (o in formato html) che trattano argomenti specifici. Questi file si trovano solitamente all'interno della directory /usr/share/doc/HOWTO.

2.2 Documentazione in Rete

2.2.1 Il progetto TLDP: Howto, Faq, Guides

TLDP e' il diminuitivo di *The Linux Documentation Project*, un organizzazione di volontari che producono, revisionano e gestiscono la documentazione sul sistema operativo GNU/Linux. I documenti fondamentalmente vengono prodotti in due formati, in base alla loro lunghezza. I piu' brevi in genere sono chiamati *HOWTO* (o *mini-HOWTO*) se sono di dimensioni davvero contenute), i piu' lunghi vengono chiamati *Guides* e trattano approfonditamente un particolare argomento di GNU/Linux.

Il numero di argomenti discussi negli HOWTO e nelle Guides e' praticamente illimitato e va dall'installazione del sistema operativo all'amministrazione dei dispositivi dei tipi piu' disparati, dei servizi e degli environment, fino a creare il proprio sistema dal nulla. Qualsiasi cosa vi possa venire in mente e' stata trattata in qualche documento del TLDP e questo grazie ai volontari che condividono le proprie esperienze.

Tutta la documentazione e' liberamente disponibile in vari formati, adatti sia alla visione on-line che alla stampa. La lingua principale in cui vengono redatti e' l'inglese, ma esistono anche versioni tradotte di una parte del materiale, in modo da rendere quest'enorme quantita' di informazioni disponibili per un pubblico piu' ampio.

2.2.2 Le riviste disponibili in rete

La piu' importante webzine disponibile e' la *Linux Gazette*. E' stata fondata nel 1995 da John M. Fisk come un servizio gratuito per gli utenti. Su richiesta di John M. Fisk, la pubblicazione fu sponsorizzata dalla SSC (Specialized Systems Consultants, che pubblica anche il Linux Journal). I contenuti sono sempre stati forniti da volontari, che hanno sempre deciso anche la linea editoriale.

Dopo alcuni anni di collaborazione, il pool di volontari e SSC sono giunti ad una biforcazione. La Linux Gazette (.net) pubblicata dai volontari e' ancora molto attiva, mentre la rivista di SSC (.com) ha chiuso i battenti.

Una cosa per cui la Linux Gazette differisce da altri giornali e webzine similari e' *The Answer Gang*. Oltre a fornire una pagina per le domande e risposte degli utenti, le domande poste all'Answer Gang ottengono una risposta anche su una mailing list. I thread di discussione che si generano entrano a far parte del progetto editoriale e vengono pubblicati.

Seguendo il motto "Making Linux just a little more fun", la rivista ha sempre operato nel rispetto della cultura aperta e cooperativa di Linux.

2.3 Ottenere Aiuto

2.3.1 I LUG

Un *Linux User Group*, per alcuni anche Linux Users Group o Linux Users' Group, spesso abbreviato in *LUG*, è un gruppo di persone che condividono una passione verso il software libero e in particolare verso il sistema operativo *GNU/Linux*.

Le modalità di associazione, ritrovo ed organizzazione sono diverse per ogni singolo LUG, ma il loro scopo è comune: diffondere e supportare il software

libero, creando un punto d'appoggio volontario dove poter porre e risolvere i piccoli e grandi problemi informatici di tutti i giorni, attraverso mailing list e convegni.

Dal 2001 ILS, *Italian Linux Society*, organizza il *Linux Day* a livello nazionale cui i LUG partecipano come organizzatori locali. Durante il Linux Day i vari gruppi si riuniscono in spazi aperti al pubblico ed organizzano conferenze che abbiano come tema il software libero e l'opensource. Durante queste giornate è anche possibile portare i propri computer per farsi installare Linux o per ricevere aiuto in generale.

Un consistente numero di LUG ha scelto, nel tempo, di assumere la denominazione *Free Software Users Group*, o FSUG, per meglio definire l'ambito del proprio operato, orientandosi cioè nettamente al supporto del Software Libero nella sua interezza, non limitandosi cioè ad un unico sistema operativo, per quanto libero ed importante, com'è appunto GNU/Linux.

2.3.2 Mailing List

Un altro utile supporto per l'amministratore sono le mailing list, sia per postare messaggi che per consultare gli archivi storici, che sono sempre disponibili. Attenzione pero', occorre sempre aver letto la documentazione disponibile ed aver consultato gli archivi alla ricerca di domande sullo stesso argomento prima di postare, altrimenti si rischia di ricevere risposte ... spiacevoli ...

2.3.3 Usenet: i gruppi di discussione

In rete esistono parecchi gruppi di discussione che trattano Linux, sia in inglese che in italiano. Si tratta di una risorsa molto importante dove e' possibile scambiare opinioni con altri utenti Linux. Per leggere gli articoli di questi gruppi occorre avere un newsreader. I gruppi che trattano Linux sparsi nel mondo sono innumerevoli...eccone un piccolo elenco:

- comp.os.linux.advocacy
- comp.os.linux.announce
- comp.os.linux.answers
- comp.os.linux.development.apps
- comp.os.linux.hardware
- comp.os.linux.m68k
- comp.os.linux.networking
- comp.os.linux.x.development.apps
- comp.os.linux.development.system
- comp.os.linux.setup
- comp.os.linux.misc
- comp.os.linux.security

- comp.os.linux.x
- alt.binaries.warez.linux
- alt.comp.linux.misc
- alt.os.linux
- alt.os.linux.mandrake
- alt.comp.linux.isp
- it.comp.os.linux.annunci
- it.comp.os.linux.development
- it.comp.os.linux.iniziare
- it.comp.os.linux.software
- it.comp.os.linux.sys
- linux.kernel

3 Il Device Mapper

Il device mapper e' un generico framework che permette di mappare un block device in un altro, a livello del kernel Linux. Esso costituisce la base su cui e' possibile costruire LVM2 e EVMS, software RAID, Cryptoloop filesystem e altre caratteristiche.

Il device mapper lavora processando i dati che gli sono passati fornendo un block device virtuale e passando poi i dati su un altro block device.

Le applicazioni che vogliono creare nuovi device mappati comunicano con il Device-mapper per mezzo della libreria libdevmapper.so, che a sua volta invia le ioctl al device /dev/mapper/control. Il device mapper e' accessibile anche dagli shell script attraverso il *dmsetup* tool.

3.1 Il Logical Volume Manager (LVM2)

La gestione dei volumi logici fornisce una visione di alto livello dello storage su un computer, che va oltre il modo classico basato su dischi e partizioni. In questo modo l'amministratore puo' allocare lo spazio agli utenti e alle applicazioni in maniera molto piu' flessibile. I volumi creati sotto il controllo dell'LVM possono essere ridimensionati e spostati a piacere.

Quando un sistema viene installato per la prima volta, una delle decisioni che ha sempre preso piu' tempo e' stata quella dell'allocazione dello spazio disco in partizioni. Questo ha sempre implicato la necessita' di una stima dello spazio che i vari filesystem avrebbero occupato nel tempo. I principianti hanno in genere oltrepassato questo ostacolo ponendo tutto il software in un'unica partizione di root.

Al contrario con l'LVM e' possibile creare uno o piu' *Volume Group* e poi definire su di essi dei volumi logici, su cui poi costruire i filesystem di nostro interesse. Naturalmente i volumi logici possono poi essere ridimensionati a piacere

finche' lo spazio sui Volume Group e' disponibile. Oltre a questo e' possibile estendere anche i Volume group a runtime e in modo trasparente.

Esistono una serie di importanti definizioni:

- **Volume Group (VG)**: e' il piu' alto livello di astrazione utilizzato da LVM, riunisce un insieme di Logical Volumes e Physical Volumes in una sola unita' amministrativa.
- **Physical Volume (PV)**: e' un hard disk o un dispositivo che viene trattato come un disco.
- **Logical Volume (LV)**: e' l'equivalente di una partizione disco in un sistema non LVM. Il LV viene visto come un comune device a blocchi.
- **Physical Extent (PE)**: ogni PV e' diviso in blocchi di dati (*chunks*), conosciuti come PE. La dimensione e' la stessa dei LE per il volume group.
- **Logical Extent (LE)**: ogni LV e' diviso in blocchi di dati, conosciuti come LE. La dimensione degli extent e' la stessa per tutti i volumi logici del volume group.

Esistono due modalita' per mappare le logical extents sulle physical extents:

- Linear mapping: assegna un range di PE ad un area di un LV in ordine
- Striped mapping: interlaccia i chunks dei LE fra un numero di volumi logici

Un altro aspetto interessante di LVM sono gli *snapshot*, che permettono all'amministratore di creare un nuovo block device che presenti una copia esatta del volume logico in un certo istante. Quando le operazioni che devono essere eseguite su questa copia sono terminate, essa puo' essere rimossa.

Dal kernel 2.6.9 il device mapper e LVM versione 2 sono integrati nel kernel, per cui nessuna patch e' piu' necessaria. L'unica accortezza deve essere quella di definire le partizioni per l'uso con il device mapper, ovvero di tipo *8e*.

I principali comandi per interagire con il sistema LVM sono:

- *Inizializzazione di dischi e partizioni*:
  Prima di poter utilizzare un disco o una partizione come PV e' necessario inizializzarli:
  pvcreate <disk_name>
- *Creazione di un VG*:
  Per questa operazione e' definito il comando vgcreate:
  vgcreate <vg_name> <PV1> <PV2> ...
  Esiste l'opzione *-s* che permette di definire la dimensione degli extent, di default 32MB.
- *Attivazione di un VG*:
  In genere al boot tutti i volumi logici vengono attivati da qualcuno degli script, se cosi' non fosse e' sufficiente il comando:
  vgchange -a y <vg_name>

- *Rimuovere un VG*:
  Una volta certi che non ci sono volumi logici definiti, occorre disattivare il VG e poi rimuoverlo:
  vgchange -a n <vg_name>
  vgremove <vg_name>

- *Aggiungere un PV ad un VG*:
  vgextend <vg_name> <PV>

- *Rimuovere PV da un VG*:
  Per prima cosa occorre essere certi che il PV non sia utilizzato da nessun LV (con il comando *pvdisplay*), se lo fosse i dati devono prima essere migrati, utilizzando *pvmove*, su un altro PV. Dopo di che:
  vgreduce <vg_name> <PV>

- *Creare un volume logico*:
  Il volume logico puo' essere creato con le dimensioni specificate in MByte oppure in numero di PE. In quest'ultimo caso e' necessario verificane il numero disponibile sul VG con il comando *vgdisplay*. Esistono tutta una serie di altre opzioni per cui si rimanda alla pagina man e all'Howto (in bibliografia):
  lvcreate -L<size_in_MB> -n<lv_name> <vg_name>

- *Rimuovere un volume logico*:
  Il volume deve essere chiuso (smontato) e poi si puo' preocedere alla rimozione:
  lvremove <lv_name>

- *Estendere un LV*:
  Questa e' di gran lunga l'operazione piu' complessa nella gestione dell'LVM. Perche' e' necessario per prima cosa estendere il LV e poi estendere il filesystem che e' stato creato su di esso. Naturalmente non tutti i filesystem prevedono la possibilita' di estensione "a caldo".
  lvextend -L<new_size> <lv_name>
  lvextend -L+<extend_size> <lv_name>
  Anche per questo comando sono disponibili moltissime opzioni per le quali e' necessario riferirisi alla pagina man.
  A questo punto deve essere fatto il resizing del filesystem. Non tutti i filesystem si comportano nello stesso modo, la maggior parte cercano di fare l'estensione per coprire la nuova dimensione dell'LV. In ogni caso questo non e' piu' un aspetto dell'LVM ed e' necessario fare riferimento alla documentazione dei filesystem.
  Nel caso di ext2/ext3 e' possibile eseguire il ridimensionamento al volo per mezzo dell'utility *resize2fs*.

- *Ridurre un LV*:
  Questa operazione puo' essere svolta con sicurezza, a differenza di molti Unix commerciali. L'unica accortezza e' quella di ridurre la dimensione del filesystem *prima* di ridurre quella del LV:
  lvreduce -L-<reduce_size> <lv_name>

- *Comandi per interrogare gli oggetti LVM*:
  Esistono tutta una serie di comandi che permettono di avere informazioni sugli oggetti appartenenti al sistema LVM:

  - *vgdisplay*
  - *pvdisplay*
  - *pvscan*

3.2 Il software RAID

L'altro elemento che si appoggia al device mapper e' il Software Raid, che si e' molto evoluto negli anni e al momento puo' essere utilizzato in produzione senza problemi.

3.2.1 I RAID levels

- Linear mode
  Due o più dischi sono combinati in un dispositivo fisico. I dischi sono "appesi" (accodati) l'uno all'altro, così lo scrivere sul dispositivo RAID riempirà prima il disco 0, poi il disco 1 e così via. Non è obbligatorio che i dischi abbiano la stessa dimensione. Infatti, non importa affatto :)
  Non c'è ridondanza in questo livello. Se un disco si danneggia, probabilmente tutti i dati saranno persi. Potreste comunque essere fortunati e recuperare alcuni dati, perché il filesysytem starà perdendo solo un grande blocco consecutivo ("chunk") di dati.
  Le prestazioni in lettura e scrittura non miglioreranno per delle singole letture/scritture. Ma se diversi utenti utilizzano il dispositivo, potreste essere fortunati nel caso in cui un utente usi il primo disco e l'altro stia accedendo a dei file che stanno sul secondo disco. Se succede questo, dovreste accorgervi di un incremento di prestazioni.

- RAID-0
  Detto anche modalità (mode) "stripe". Come il linear mode, eccetto che le letture e le scritture sono fatte in parallelo sui dischi. I dischi dovrebbero essere approssimativamente della stessa dimensione. Siccome tutti gli accessi sono effettuati in parallelo, i dischi si dovrebbero riempire nella stessa misura. Se un disco è più grande degli altri, lo spazio eccedente è ancora usato nel dispositivo RAID, ma l'accesso avverrà solo sul disco più grande durante le scritture alla fine del dispositivo RAID. Questo va naturalmente a deterioramento delle prestazioni.
  Come per il linear mode, non c'è nessuna ridondanza in questo livello. Diversamente dal linear mode, non sarà possibile recuperare alcun dato se un disco si danneggia. Se un disco viene rimosso da un RAID-0, il RAID non perderà solo un grande consecutivo blocco di dati, esso sarà riempito con piccoli buchi lungo tutto il dispositivo. e2fsck non sarà probabilmente in grado di recuperare molto da questo dispositivo.
  Le prestazioni in lettura e scrittura cresceranno, poiché le letture e scritture sono fatte in parallelo sui dischi. Questa è solitamente la ragione per cui si implementa un RAID-0.

- RAID-1
  Questa è la prima modalità che presenta ridondanza. Il RAID-1 può essere usato su due o più dischi con zero o più spare-disk. Questa modalità mantiene un'immagine (mirror) esatta del contenuto di un disco sugli altri. Naturalmente i dischi devono essere della stessa dimensione. Se un disco è più grande di un altro, il dispositivo RAID avrà la dimensione del disco più piccolo.
  Se fino a N-1 dischi vengono rimossi (o si danneggiano), tutti i dati saranno ancora intatti. Se ci sono spare-disk disponibili e se il sistema (leggi SCSI driver o chipset IDE, ecc.) sopravvive al blocco del sistema, la ricostruzione del mirror inizierà immediatamente su uno degli spare-disk, dopo aver individuato il disco danneggiato.

- RAID-4
  Questo livello RAID non è usato molto spesso. Può essere usato su tre o più dischi. Invece di fare un immagine (mirror) completa delle informazioni, esso tiene delle informazioni di parità su un disco e scrive i dati sugli altri dischi in una maniera simile al RAID-0. Siccome un disco è riservato per le informazioni di parità, la dimensione dell'array sarà (N-1)*S, dove S rappresenta la dimensione del più piccolo disco dell'array. Così come nel RAID-1, i dischi dovrebbero essere della stessa dimensione, altrimenti il valore S nella formula precedente sarà la dimensione del più piccolo disco dell'array.
  Se un disco si danneggia, le informazioni di parità possono essere utilizzate per ricostruire tutti i dati. Se si danneggiano due dischi tutti i dati saranno persi.
  La ragione per cui questo livello non è usato spesso è che l'informazione di parità è tenuta su un disco. Quindi questa informazione deve essere aggiornata ogni volta uno degli altri dischi viene scritto. Quindi, il disco che contiene l'informazione di parità diventa un collo di bottiglia, se esso non è molto più veloce degli altri dischi. Comunque, se vi accade di avere molti dischi lenti ed uno molto veloce, questo livello RAID può essere molto utile.

- RAID-5
  Questa è forse la più utile modalità RAID quando di desidera combinare un gran numero di dischi e mantenere ancora una certa ridondanza. Il RAID-5 può essere usato su tre o più dischi, con zero o più spare-disk. Il dispositivo RAID-5 che viene fuori avrà la dimensione (N-1)*S, come nel RAID-4. La grande differenza fra il RAID-5 ed il RAID-4 è che le informazioni di parità sono distribuite in modo uguale fra i dischi di cui è composto l'array, evitando così il collo di bottiglia che si creava nel RAID-4.
  Se uno dei dischi si danneggia, tutti i dati saranno ancora intatti, grazie alle informazioni di parità. Se degli spare-disk sono disponibili, la ricostruzione inizierà immediatamente dopo il guasto del dispositivo. Se due dischi si danneggiano simultaneamente, tutti i dati saranno persi. Il RAID-5 può sopravvivere al danneggiamento di un disco, ma non a quello di due o più.
  Sia le prestazioni in scrittura che in lettura migliorano, ma è difficile predire di quanto.

Spare disks Gli spare disks sono dischi che non fanno parte dell'array RAID fino a che uno dei dischi attivi smette di funzionare. Quando il guasto di un disco viene rilevato, questo dispositivo viene marcato come "cattivo" (bad) e la ricostruzione viene immediatamente iniziata su uno degli spare-disk a disposizione. Quindi, gli spare-disk aggiungono un'utile extra sicurezza specialmente ai sistemi RAID-5. Ci si può permettere di far lavorare il sistema per un po', con un dispositivo guasto, poiché tutta la ridondanza è conservata per mezzo degli spare-disk. Non si può essere sicuri che un sistema sopravviva al guasto di un disco. Il RAID layer dovrebbe gestire i guasti ai dischi piuttosto bene, ma i driver SCSI potrebbero crollare sulla gestione degli errori, o il chipset IDE potrebbe bloccarsi, oppure una quantità di altre cose potrebbe accadere.

Il moderno supporto per il software Raid e' incluso in tutti i kernel 2.4 e 2.6, lo si puo' notare dalla presenza del file */proc/mdstat*, che e' un valido aiuto per l'amministratore.

Per definire uno o piu' software array device e' necessario lavorare sul file /etc/raidtab, in cui devono essere definiti una serie di parametri. Per prima cosa il nome del device, che per convenzione fa riferimento ai block device *mdX* (X=0,1,2, ...). Dopo di che e' necessario specificare il tipo di raid (il Raid level), la chunk size (la più piccola massa "atomica" di dati che possa essere scritta su un dispositivo) e i device che costituiscono l'array (in genere dischi o partizioni su di essi).

Fatto questo e' sufficiente il comando:
mkraid /dev/mdX
per creare l'array.

Persistent Superblock Una delle opzioni da specificare nel file */etc/raidtab* e' quella del persistent superblock.

"Tanto tempo fa..." (TM), i raidtools avrebbero letto il vostro file /etc/raidtab e poi avrebbero inizializzato l'array. Comunque, questo avrebbe richiesto che il filesystem su cui risiedeva /etc/raidtab fosse montato. Questo risulta essere sfavorevole se avete intenzione di fare il boot da raid. Inoltre il vecchio approccio portava a delle complicazioni quando si montavano i filesystem sui dispositivi RAID. Essi non potevano essere messi nel file /etc/fstab come al solito, ma avrebbero dovuto essere montati negli init script. I persistent superblock risolvono questi problemi. Quando un array è inizializzato con l'opzione persistent-superblock nel file /etc/raidtab uno speciale superblock viene scritto all'inizio di tutti i dischi che compongono l'array. Questo permette al kernel di leggere la configurazione dei dispositivi RAID direttamente dai dischi che ne fanno parte, invece di ottenerla da qualche file di configurazione che potrebbe non essere disponibile in qualche momento. Dovreste comunque mantenere un file /etc/raidtab file, consistente, poiché potreste aver bisogno di questo file per le successive ricostruzioni dell'array. Il persistent superblock è obbligatorio se volete l'autorilevamento (autodetection) dei vostri dispositivi RAID al boot del sistema.

L'autorilevamento permette ai dispositivi RAID di essere automaticamente riconosciuti dal kernel al boot del sistema, subito dopo che il solito rilevamento delle partizioni è stato eseguito. Tutto ciò richiede diverse cose:

- Avete bisogno del supporto all'autorilevamanto (autodetection) nel kernel.

- Dovreste aver creato i dispositivi RAID usando i persistent- superblock
- Il tipo di partizioni dei dispositivi usati nel RAID deve essere impostato al valore 0xFD (usate fdisk per impostare il tipo "fd")

Il funzionamento del Raid software e' poi del tutto equivalente a quello dei Raid amministrati dai controller. Qualora i dispositivi siano hot-swappable, e' possibile rimuovere e sostiruire i dischi al volo, con i comandi *raidhotadd* e *raidhotremove*. La ricostruzione su questi device, come sugli spare se presenti, sara' immediata e trasparente per gli utenti.

4 La costruzione dei pacchetti RPM

La costruzione dei pacchetti rpm puo' essere fatta sotto un opportuno albero di directory che le distribuzioni forniscono:

/usr/src/redhat

o qualcosa di simile.

Ogni utente puo' riprodurre un albero di questo tipo sulla propria home directory se volesse fare una creazione da utente non privilegiato. Oltre a questo e' necessario definire un file di configurazione *.rpmmacros* nella propria home directory che permetta di impostare una serie di macro utilizzate poi dal comando rpmbuild. Un'idea delle macro esistenti si puo' avere dal file */usr/lib/rpm/macros*. I paramentri fondamentali da inserire sono: *%_topdir, %_tmppath, %_buildroot, %debug_package, %_rpmfilename, %packager, %vendor, %distribution, %_gpg_name, %_gpg_path, %_gpgbin*.

E' possibile verificare i valori di queste macro con il comando:

rpm –eval 'nome_macro'

Usualmente l'albero fornito dalla distribuzione si utilizza quando si ricompila un pacchetto SRPM (Source RPM), cioe' un pacchetto creato da qualcuno con i sorgenti del software e uno speciale file detto *spec* file, con le informazioni che riguardano la creazione del pacchetto.

Le subdirectory che compongono l'albero sono cinque:

- BUILD - Viene utilizzata per la fase vera e propria di compilazione. L'archivio dei sorgenti viene scompattato sotto questa directory e viene lanciato il processo di compilazione secondo le istruzioni dello spec file. In genere (per software GPL) attraverso il classico procedimento di *configure; make*.
- RPMS - In questa directory, che contiene una subdirectory per ogni architettura per cui potra' essere generato un pacchetto rpm, viene creato il pacchetto rpm vero e proprio. Quello che contiene i binari e che puo' essere installato su un sistema. Il tipo di architettura verra' rispecchiata anche in una delle estensioni del pacchetto generato.
- SOURCES - Quando si parta da un pacchetto SRPM per la costruzione (build) del corrispondente pacchetto dei binari, questa directory viene utilizzata per contenere l'archivio (in genere tar.gz o tar.bz2) dei sorgenti del software da compilare.
- SPECS - Questa directory e' deputata a contenere lo spec file.

- SRPMS - Questa directory conterra' il pacchetto SRPM, qualora si generi a partire dai sorgenti e dallo spec file. Infatti il comando rpmbuild e' in grado di creare sia un pacchetto RPM che il corrispondente SRPM.

4.1 Lo Spec file

In generale un pacchetto RPM puo' essere generato da:

- un pacchetto SRPM, che contiene lo spec file, il file con il software, i file di configurazione ed eventuali patch.
- dai sorgenti originali, costruendosi lo spec file.
- a partire dai sorgenti a cui sia stato allegato lo spec file.

I file spec possono essere anche molto complessi, essi sono scrittti in una particolare sintassi, dove agli elementi delle macro rpm vengono affiancati shell script e altro.

Il file e' diviso in una serie di sezioni:

- *Header*
- *Prep (%prep)*
- *Build (%build)*
- *Install (%install)*
- *Files (%files)*
- *Script*
- *Changelog*

L'insieme di queste sezioni, in parte facoltative, descrivono: il file sorgente, le patch, le istruzioni che rpm deve seguire per la compilazione e l'installazione, definiscono le personalizzazioni da usare in fase di installazione/rimozione/aggiornamento.

Le linee che iniziano con il carattere % possono introdurre:

- Una shell macro predefinita
- L'inizio di una sezione
- Una direttiva utilizzata dalla sezione
- L'invocazione di una macro costruita dall'utente
- L'invocazione di una macro predefinita nel sistema RPM

I commenti nel file vengono introdotti con il carattere #, le macro sono attive anche all'interno dei commenti. Per poterne fare l'escape occorre precederle con un ulteriore carattere %.

Vediamo le sezioni dello spec file:

Header In questa sezione si trovano direttive e definizioni di tag. Il loro valore puo' provenire anche da una macro built-in o definita dall'utente.

- Name: Nome del pacchetto
- Version: Versione del pacchetto
- Release: Numero di revisione della versione del package. In genere dovrebbe partire da 1
- Epoch: Permette un meccanismo univoco per classificare correttamente le versioni del software. Contiene un numero da incrementare ad ogni release
- Summary: Una breve descrizione del pacchetto
- %decription: Una descrizione completa del pacchetto
- License: Licenza con cui e' rilasciato il software da cui si sta ricavando il pacchetto
- Group: La caratterizzazione all'interno del sistema del software che si va ad installare. In genere una lista si puo' trovare in /usr/share/doc/rpm-*versione*/GROUPS
- Source: Da' indicazioni sugli oggetti che verranno scompattati nella *BuildArea* dalla sezione *%prep*
- Patch: Da' indicazione sulle patch che dovranno essere applicate da *%prep* agli oggetti delle voce Source
- BuildRoot: Area virtuale in cui verra' installato il software
- Requires: Elenco dei package (e della versione) che dovranno essere nel sistema al tempo dell'installazione
- Prereq: Simile al precedente, indica quello che deve essere presente nel sistema prima dell'installazione del package
- BuildRequires: Come le sezione precedente, ma fa riferimento al Build time

Nel confronto fra le versioni di un package rpm si fa riferimento alla stringa "name-epoch-version-release" (nota come NEVR.RPM).

Oltre a queste esistono molte altre voci dell'Header, per le quali occorre consultare i manuali di rpm (http://www.rpm.org).

%prep Nella fase descritta da questa sezione viene scompattato il software sorgente nella *BuildArea*, previa eliminazione della precedente directory se esiste, e vengono applicate le eventuali patch. In genere per questa operazione viene utilizzata la macro *%setup*. Un'altra macro utilizzata da questa sezione e' *%patch*, che comunica che si devono applicare le patch indicate dalla direttiva *Patch*.

%build Qui vengono posti i comandi per configurare il software e compilarlo. Se si utilizza autoconf, esiste la direttiva *%configure*, dopo di che il software viene compilato con *make*.

%install In questa sezione gli oggetti presenti nella *BuildArea* vengono riportati nella *BuildRoot*. Per prima cosa si elimina la directory, si esegue poi il comando *install* per collocare le varie parti del software al loro posto. Se il software supporta autoconf, si puo' utilizzare *%macroinstall* per la corretta installazione del software nelle subdirectory. Questa sezione contiene in genere anche la macro *%clean* per la gestione della directory al termine della generazione del pacchetto.

%file In questa sezione vengono definiti gli oggetti del package, che RPM costruira' a partire dalla *BuildRoot*. I file di configurazione vengono referenziati con *%config*, quelli di documentazione con *%doc*. E' consigliato definire i valori di default per i permessi degli oggetti con la macro *%defattr*, mentre per i singoli oggetti si potra' intervenire con *%attr*. Utilizzando la macro *%dir* si puo' definire l'appartenenza di una directory al package, ma non quella dei suoi file.

%changelog E' una sezione che riporta i cambiamenti che ha subito il processo di generazione del pacchetto nelle release successive.

Detto questo e' possibile procedere alla generazione del pacchetto con il comando:

rpmbuild

Esso ha una notevole quantita di opzioni, per le quali si rimanda alla documentazione. Le piu' utilizzate sono comunque *-ba*, che genera sia il pacchetto RPM che quello SRPM e *-bb* che genera solo il pacchetto RPM.

5 Appendice A - Linux Documentation Project Manifesto

1. OVERVIEW

The goal of the Linux Documentation Project (LDP) is to create and distribute the canonical set of free GNU/Linux documentation. While GNU/Linux applications and utilities may come with their own documentation, LDP documentation fills in the numerous gaps.

The hundreds of existing LDP documents present both overviews and details of: the GNU/Linux Operating System, System Administration, Hardware, Networks, Servers, GUIs, Programming, Language Support, etc. Not every important topic is currently covered so LDP is seeking new authors to fill in the gaps.

An additional goal is to collaborate on all of the issues of GNU/Linux documentation. We hope to establish a system of documentation that is easy to use and search. This includes the integration of all available documents.

We freely distribute our documents via the Internet. Some major distributions of Linux include them on CDs. If you are interested in publishing any of the LDP works, see the section "Publishing LDP Documents" below.

The LDP is essentially a loose team of volunteers with minimal central organization. Anyone who would like to help is welcome to join in this effort. We feel that working together informally and discussing projects on our mailing lists is the best way to go. When we disagree on things, we try to reason with each other until we reach an informed consensus.

2. CURRENT PROJECTS and GETTING INVOLVED

Currently, the major effort of the LDP is the writing of HOWTOs. If you think you would like to write a certain HOWTO first check to see if one already exists on your topic. If so, you may contact the feedback list and offer to help. If there is no HOWTO about it, you may want to create a new HOWTO. See the LDP Author Guide (formerly the HOWTO-HOWTO) and/or the HOWTO-INDEX for more details.

The "Guides" are large book-size LDP documents covering broad topics such as system administration.

Other tasks include checking the HOWTOs for clarity and errors, improving our website, and developing an integrated system of documentation for Linux. If you are interested in any such project (other than writing HOWTOs), contact the current LDP coordinator Guylhem Aznar at guylhem@metalab.unc.edu or email the LDP at feedback@en.tldp.org.

3. LDP WEBSITES

The LDP has over 250 mirror sites worldwide where one may inspect and/or download LDP documents. The main site is http://tldp.org. Go here to find the list of mirror sites and then use the nearest mirror site.

4. DOCUMENTATION CONVENTIONS

Here are the conventions that are currently used for LDP documents. If you are interested in writing another document using different conventions, please let us know of your plans first.

*

All HOWTO documents must be in one of these formats: LinuxDoc SGML or DocBook XML/SGML. LinuxDoc is the simplest DTD while DocBook is a

more complex mark-up (with more features). *

The guides – full books produced by the LDP – have historically been done in LaTeX, as their primary goal has been to be printed documentation. However, guide authors have been moving towards SGML with the DocBook DTD, because it allows them to create many different kinds of output, both printed and on-line.

5. LICENSE REQUIREMENTS

Anyone may copy and distribute (sell or give away) LDP documents (or other LDP works) in any media and/or format. No fees are required to be paid to the authors. It is not required that the documents be modifiable, but it is encouraged.

You can come up with your own license terms that satisfy these conditions, or you can use a previously prepared license. The LDP has a boilerplate license that you can use if you wish. Some people like to use the GPL, while others write their own. There is a project underway to create a special GPL license just for documents and this may turn out to be a good choice ("The GNU Free Documentation License").

The copyright for each document should be in the name of the principal authors. "The Linux Documentation Project" isn't a formal entity and thus can't be used as a copyright owner.

6. BOILERPLATE LICENSE

Here is a sample copyright notice and "boilerplate" license you may want to use for your work:

Copyright (c) 2000 by John Doe (change to your name)

Please freely copy and distribute (sell or give away) this document in any format. It's requested that corrections and/or comments be fowarded to the document maintainer. You may create a derivative work and distribute it provided that you:

1. Send your derivative work (in the most suitable format such as sgml) to the LDP (Linux Documentation Project) or the like for posting on the Internet. If not the LDP, then let the LDP know where it is available.

2. License the derivative work with this same license or use GPL. Include a copyright notice and at least a pointer to the license used.

3. Give due credit to previous authors and major contributors.

If you're considering making a derived work other than a translation, it's requested that you discuss your plans with the current maintainer.

7. PUBLISHING LDP DOCUMENTS

If you're a publishing company interested in distributing any of the LDP documents, read on.

By the license requirements given previously, anyone is allowed to publish and distribute verbatim copies of the Linux Documentation Project documents. You don't need our explicit permission for this. However, if you would like to distribute a translation or derivative work based on any of the LDP documents, you may need to obtain permission from the author, in writing, before doing so, if the license requires that.

You may, of course, sell the LDP documents for profit. We encourage you to do so. Keep in mind, however, that because the LDP documents are freely

distributable, anyone may make copies and distribute them. Thus the parts of a book which may be freely copied should be separated (and identified) in such a manner as to facilitate copying them without infringing on the copyright of other material.

We do not require you to pay royalties from any profit earned by selling LDP documents. However, we would like to suggest that if you do sell LDP documents for profit, that you either offer the author royalties, or donate a portion of your earnings to the author, the LDP as a whole, or to the Linux development community. You may also wish to send one or more free copies of the LDP documents that you are distributing to the authors. Your show of support for the LDP and the Linux community will be very much appreciated.

We would like to be informed of any plans to publish or distribute LDP documents, just so we know how they're becoming available. If you are publishing or planning to publish any LDP documents, please send mail to feedback@en.tldp.org. It's nice to know who's doing what.

We encourage Linux software distributors to distribute the LDP documents on CDs with their software. The LDP documents are intended to be used as "official" Linux documentation, and we are glad to see distributors bundling the LDP documents with the software.

References

[1] Appunti di informatica libera - Appunti Linux Copyright © 1997-2000 Daniele Giacomini Appunti di informatica liberaCopyright © 2000-2006 Daniele Giacomini Via Morganella Est, 21 – I-31050 Ponzano Veneto http://na.mirror.garr.it/mirrors/appuntilinux/HTML/a2.htm

[2] Distrowatch - http://distrowatch.com/

[3] RPM Howto - http://www.rpm.org/RPM-HOWTO/

[4] RPM Guide - http://fedora.redhat.com/docs/drafts/rpm-guide-en/

[5] Linux from Scatch - http://www.linuxfromscratch.org/lfs/

[6] FrozenTech's LiveCD List - http://www.frozentech.com/content/livecd.php

[7] LVM Howto - http://tldp.org/HOWTO/html_single/LVM-HOWTO/

[8] Software Raid Howto - http://tldp.org/HOWTO/html_single/Software-RAID-HOWTO/

[9] Linux Documentation Project Manifesto - http://tldp.org/manifesto.html

Contents