Amministrazione di sistemi Linux

Università degli Studi di Sassari

autore
Gavino Mura

anno 2010

Indice generale

1 I sistemi Linux

1.1 Cosa è un sistema Linux

Linux è un termine che può assumere più di un significato. A seconda del contesto infatti può indicare il kernel[1] originariamente sviluppato da Linux Torvalds, oppure il sistema operativo basato su tale kernel. Dal momento che il sistema operativo basato su kernel Linux include molto software del progetto GNU[2] (logo ), la Free Software Foundation[3] insiste nell'affermare che per indicarlo bisognerebbe usare il termine "GNU/Linux", tuttavia l'accezione della parola "Linux" come nome dell'intero sistema operativo è ormai entrata da tempo nell'uso comune, sia in ambito tecnico/scientifico che in ambito popolare. Per indicare il sistema operativo basato su kernel Linux, si possono quindi utilizzare indifferentemente il termine "GNU/Linux" o il termine "Linux".
La licenza adottata dal sistema operativo Linux è la GPL, o general public license, che permette all'utente libertà di utilizzo, copia, modifica e distruzione. Se l'utente distribuisce copie del software, deve rendere disponibile il codice sorgente a ogni acquirente, incluse tutte le modifiche eventualmente effettuate.
Il progetto del kernel è ancora oggi diretto da Torvalds, mentre altre parti del sistema, come le componenti GNU, sono sviluppate separatamente. Il compito di fornire un sistema integrato, che combina tutte le componenti di base con le interfacce grafiche (come per esempio GNOME[4] 

o KDE[5]  , che a loro volta si basano sulla presenza dell'X Window System[6]) e con il software applicativo, viene ora svolto dalle distribuzioni.

Nel 1996 fu scelto come logo ufficiale di Linux un pinguino  disegnato da Larry Ewing e ad esso venne dato il nome di TUX come abbreviazione di Torvalds UniX. Altre fonti sostengono si tratti dell'abbreviazione di TUXedo che in inglese significa il nostro "giacca e cravatta" a causa della colorazione del corpo del pinguino.

---

[1] il kernel è l'elemento fondamentale di un sistema operativo. Si tratta di un software avente il compito di fornire ai programmi in esecuzione sull'elaboratore un accesso sicuro e controllato all'hardware.
[2] GNU è un acronimo e significa "GNU is Not Unix". E' un progetto per la realizzazione di software, lanciato nel 1983 da Richard Stallman, che si basa su una gestione particolare dei diritti d'autore, secondo la definizione di software libero (contrapposta a software proprietario).
[3] Fondazione per il Software Libero, detentrice del copyright del software GNU.
[4] **N**etwork **O**bject **M**odel **E**nvironment (GNOME) è un ambiente di sviluppo per interfacce grafiche che permette di usare un computer tramite l'interazione con oggetti grafici, come le icone e le finestre dei programmi.
[5] **K D**esktop **E**nvironment è, come recita la pagina di benvenuto della sua documentazione, un ambiente desktop grafico per postazioni di lavoro Unix.
[6] **X W**indow **S**ystem, noto in gergo come X Window o X11 o ancor più semplicemente X, è di fatto il gestore grafico standard per tutti i sistemi Unix.

Oggi GNU/Linux resta il sistema operativo preferito da migliaia di programmatori sparsi in tutto il mondo, è usato soprattutto come server in ambienti di produzione e gode del supporto di società come IBM, Sun Microsystems, Hewlett-Packard, e Novell. Ultimamente ha conosciuto anche la sua affermazione in ambiente desktop e su sistemi embedded come cellulari e palmari.

1.2 L'importanza di Linux

GNU/Linux è il prodotto di appassionati indipendenti creato per pura sfida intellettuale, senza vincoli commerciali. Questo ha generato due importanti conseguenze:

- in primo luogo il risultato ottenuto è la sorprendente collaborazione di migliaia di programmatori sparsi in tutto il mondo (per questo viene considerato come il più grosso progetto collaborativo della storia dell'uomo);
- il secondo aspetto è la "liberalizzazione" del prodotto, GNU/Linux è un software libero che permette agli utenti di avere a disposizione un sistema operativo completamente funzionante slegato dalle classiche leggi commerciali.

L'aspetto della libertà del programma mette al centro l'utente garantendogli la possibilità di visionare e modificare i codici sorgenti, la possibilità di usare il software per qualsiasi scopo, la possibilità di ridistribuirlo nel formato originario o da lui modificato. Il sistema GNU/Linux, anche se in modo variabile tra le diverse distribuzioni, è molto trasparente, configurabile e adattabile ad ambiti disparati, a differenza di molti sistemi operativi commerciali che sacrificano il controllo della macchina in favore della facilità d'uso. Questa filosofia permette lo scambio di informazioni tra gli utenti e i programmatori e ha come risultato un prodotto che risulta avere prestazioni migliori in alcuni ambiti rispetto agli attuali concorrenti commerciali.
GNU/Linux è meno esposto degli altri sistemi operativi ai virus[7] ed in genere ai malware[8] informatici per vari motivi:

- l'infezione di una macchina di solito è limitata al singolo utente e quindi non compromette, di norma, l'intero sistema operativo;
- quando viene scoperto un problema in un prodotto di software libero si ha la patch (correzione) di solito nel giro di poche ore o giorni rendendo vane le possibilità di un virus di sfruttare tale falla;
- chi sviluppa software maligno, generalmente, desidera colpire il maggior numero di macchine possibile: data la diffusione di Windows assai maggiore rispetto a quella di sistemi operativi basati su Linux, tendenzialmente il malware in circolazione è progettato per infettare macchine Windows, quindi incompatibile con Linux (solo raramente i virus sono multi piattaforma);
- gli utenti di Linux sono meno numerosi, ma più preparati tecnicamente o conoscono per lo meno alcune delle basi di un sistema operativo, sono consapevoli di cosa stiano facendo e quale sia il risultato che vogliono ottenere.

[7] Nell'informatica un virus è un frammento di software che è in grado, una volta eseguito, di infettare dei file in modo da riprodursi facendo copie di sé stesso, generalmente senza farsi rilevare dall'utente.
[8] Si definisce malware un qualsiasi software creato con il solo scopo di causare danni più o meno gravi al computer su cui viene eseguito.

1.3 Le distribuzioni di Linux

Non esiste un'unica versione di GNU/Linux, ma esistono diverse distribuzioni, solitamente create da comunità di sviluppatori o società, che preparano e scelgono i pacchetti da includere. Tutte le distribuzioni condividono il kernel di Linux, mentre si differenziano tra loro per il cosiddetto "parco software", cioè i pacchetti preparati e/o selezionati dagli sviluppatori per la distribuzione stessa, per il sistema di gestione del software e per i servizi di assistenza/manutenzione offerti.
Esistono distribuzioni eseguibili direttamente da CD o DVD senza che sia richiesta l'installazione sul proprio computer, per questo sono chiamate distribuzioni "live" e permettono di provare un sistema operativo Linux senza dover cancellare e/o modificare il sistema originario.
Alcune tra le principali distribuzioni Linux sono:

- Reh Hat  (o cappello rosso, per via del suo logo), è una distribuzione di proprietà della società commerciale Red Hat Inc.[9], che ne vende il supporto software, scaricabile e usabile in prova, previa registrazione sul sito della società stessa. E' stata la prima distribuzione di Linux a usare il formato RPM Package Manager[10] come sistema di gestione dei pacchetti, che usa a tuttora integrato in yum (Yellowdog Updater Modified)[11], e nel tempo è servita come punto di partenza per molte altre distribuzioni.

- Fedora  è una distribuzione Linux curata dal Progetto Fedora, un progetto open source sponsorizzato (ma non direttamente supportato) da Red Hat Inc. e supportato dalla community[12].

- Debian  , creata dal Debian Project, è una distribuzione di software libero largamente usata e sviluppata attraverso la collaborazione di volontari da ogni parte del mondo. Debian è conosciuta per la sua aderenza alle filosofie di GNU e del software libero, le rigide politiche riguardo alla qualità dei pacchetti e le release, il modo aperto di sviluppare e testare il software e la libertà di scelta concessa all'utente. Debian è sostenuta da donazioni attraverso "Software in the Public Interest" (SPI Inc.), una organizzazione non-profit per i progetti di software libero. Il suo sistema di gestione dei pacchetti software è APT (Advanced Packaging Tool), che integra un sistema di risoluzione delle dipendenze, la

[9] Red Hat Inc. è stata la prima società fornitrice di soluzioni open source (a codice aperto, cioè fornita con il codice sorgente) ad essere quotata alla borsa dei titoli tecnologici di Wall Street.
[10] RPM è un sistema di gestione dei pacchetti, utilizzato per installare, verificare, aggiornare e disinstallare i pacchetti di una distribuzione.
[11] Yellow dog Updater, Modified (YUM) è un sistema di gestione dei pacchetti open source a linea di comando per i sistemi operativi Linux compatibili col formato RPM.
[12] Una comunità virtuale o comunità online è un insieme di persone interessate ad un determinato argomento comune che corrispondono tra loro attraverso Internet.

possibilità di effettuare molto facilmente un upgrade (di alcuni pacchetti o dell'intero sistema operativo) e la possibilità di passare da una release ad un'altra.

- Ubuntu , è una distribuzione GNU/Linux nata nel 2004 e basata su Debian, che si concentra sulla facilità di installazione e d'uso e sul rilascio regolare (semestrale) delle nuove versioni. Rispetto a Debian, Ubuntu ha un orientamento più spiccato verso l'utilizzo desktop e una maggiore attenzione al supporto hardware dei portatili. Finanziata dalla società Canonical Ltd (registrata nell'Isola di Man), rimane comunque in tutto e per tutto un software libero. Il nome deriva da una antica parola Zulu diffusa in varie parti dell'Africa e che corrisponde indicativamente al concetto di "*umanità verso gli altri*", a volte tradotto anche "*io sono ciò che sono per merito di ciò che siamo tutti*".

- Slackware è una distribuzione GNU/Linux tra le più longeve. Fu creata da Patrick Volkerding quando ancora era uno studente, che utilizzò come base la distribuzione Softlanding Linux System, e venne pubblicata per la prima volta il 16 luglio 1993 sul newsgroup comp.os.linux. Essendo stata concepita per essere più *unix-like* possibile, Slackware è considerata un ottimo modo per imparare il funzionamento di GNU/Linux, tanto che i suoi estimatori usano dire: *"When you know Slackware, you know Linux... when you know Red Hat, all you know is Red Hat."* ("Quando conosci Slackware, conosci Linux... quando conosci Red Hat, conosci solo Red Hat"). I pacchetti di Slackware sono in genere caratterizzati dall'estensione .tgz. Si tratta di pacchetti compilati, e compressi in un archivio, secondo la loro posizione finale, una volta estratti nella directory di root (*/*) del sistema. Vi si può trovare, a volte, uno script (*doinst.sh*) che permette di eseguire alcune operazioni utili per il corretto funzionamento del pacchetto installato. Nel ChangeLog dell' 8 maggio 2009, è stato introdotto un nuovo formato dei pacchetti Slackware chiamato .txz. Il sistema impiegato per la gestione dei pacchetti di installazione, "*pkgtool*[13]", non permette però la risoluzione delle dipendenze, per questo sono di solito presenti altri strumenti quali Checkinstall, che permette, una volta installato, di creare pacchetti (tgz, rpm o deb) direttamente dai sorgenti di un programma.

- QiLinux , è la distribuzione Linux italiana nata a Torino nel 2003 e realizzata completamente da zero ("from scratch") per sistemi desktop, server e enterprise server. Sponsorizzata e mantenuta dalla società QiNet di Torino viene rilasciata secondo la licenza GNU.

- SUSE , è una delle principali distribuzioni GNU/Linux ed è stata prodotta in Germania. Risulta molto semplice da utilizzare ed è mirata alla stessa utenza di Microsoft Windows in quanto l'ambiente desktop è user-friendly[14] e molto simile al sistema operativo di casa Microsoft. E' di proprietà della Novell, che ne vende il supporto software, ma fornisce anche una versione gratuita chiamata OpenSUSE. Include un programma di

[13] Acronimo di "*Slackware Package Tool*".
[14] Facile da usare per gli utenti.

installazione ed amministrazione, YaST2[15], che gestisce la partizione dell'hard disk, il setup del sistema, gli aggiornamenti online, la configurazione della rete e del firewall, l'amministrazione degli utenti e oltre.

[15] Yet Another Setup Tool (YaST) è un mezzo di configurazione del sistema operativo.

2 La gestione del filesystem

2.1 Premessa

In un sistema informatico si intende per filesystem quel sottosistema che si occupa della gestione (organizzazione e archiviazione) dei file appartenenti al sistema stesso. Più formalmente è l'insieme dei tipi di dati astratti necessari per la memorizzazione, organizzazione gerarchica, manipolazione, navigazione e accesso ai dati.
Il filesystem è una parte integrante di qualsiasi sistema operativo. Inizialmente l'unico vero compito dei sistemi operativi era proprio la gestione dei files.

I dispositivi di archiviazione, come ad esempio i dischi fissi si presentano infatti al sistema operativo come un array di blocchi di dimensione fissa, generalmente chiamati *settori*, tipicamente di 512 byte l'uno e le operazioni disponibili sono la lettura e la scrittura di un blocco arbitrario, o talvolta di un insieme di blocchi. Il filesystem si occupa di interfacciare tutte le operazioni inerenti la gestione di tali settori in moda da rendere disponibili tutte le operazioni essenziali sui files e sulle directory.

A tale scopo, oltre ai dati veri e propri, vengono memorizzati anche dei dati chiamati "*metadati*" che permettono di mantenere traccia di alcune informazioni utili alla gestione e organizzazione dei dati stessi. Tali metadati contengono ad esempio informazioni quali: il timestamp di modifica del file, il timestemp di creazione del file, il proprietario del file (utente o gruppo che sia), i diritti di accesso al file, e altri.

Esistono inoltre vari tipi di filesystem, ciascuno con le sue specifiche caratteristiche, sviluppatisi su differenti ambienti, che rispondono ad esigenze che possono essere generiche o specifiche.

Ad esempio, un filesystem di tipo "*NFS*" risponde all'esigenza di impiego di file non localizzati localmente al server di appartenenza, ma su una macchina della rete dati, o il filesystem "*ISO9660*" risponde all'esigenza di memorizzazione su supporti quali i compact disk.

Un elenco, non esaustivo, dei filesystem gestiti da un sistema Linux e disponibile eseguendo il "*man*" sul comando "*mount*":

> adfs, affs, autofs, cifs, coda, coherent, cramfs, debugfs, devpts, efs, ext, ext2, ext3, ext4, hfs, hfsplus, hpfs, iso9660, jfs, minix, msdos, ncpfs, nfs, nfs4, ntfs, proc, qnx4, ramfs, reiserfs, romfs, smbfs, sysv, tmpfs, udf, ufs, umsdos, usbfs, vfat, xenix, xfs, xiafs

anche se, come già detto, è possibile estendere tale elenco tramite l'installazione di pacchetti aggiuntivi o l'aggiunta di patch di sistema.

Si prenda ad esempio il filesystem "ntfs" che è nativamente supportato in sola lettura. Nel caso si volesse accedere a tale filesystem anche in scrittura è necessaria l'installazione del pacchetto:

```
ntfs-3g
```

Alcuni tipi di filesystem sono inoltre delle evoluzioni di filesystem precedenti, quali quelli della famiglia "*ext*", arrivata alla versione 4, disponibile dalla versione di kernel Linux 2.6.28. Su sistemi impiegati durante il corso tale filesystem non è disponibile in quanto la versione di kernel è precedente a quella citata.
L'evoluzione delle varie versioni di filesystem, o la nascita di nuovi, ha seguito la richiesta di implementazione di nuove funzionalità. Per un elenco di tali funzionalità, o per il confronto di queste ultime tra i vari filesystem si veda il seguente link:

http://en.wikipedia.org/wiki/Comparison_of_file_systems

dalle quali si cita brevemente il "*journaling*", come quella funzionalità che permette di tenere traccia dei cambiamenti al filesystem prima di committare effettivamente i dati sul filesystem. In questo modo è possibile ritornare ad una situazione consistente del filesystem anche nel caso accada un crash del sistema durante la scrittura dei dati su disco.

Alcune funzionalità di gestione, quali quelle di estensione di un filesystem e di resilience di guasti hardware (riferiti ai dischi fisici) non sono però presenti sulla maggior parte dei filesystem. Per ovviare a tali mancanze vengono impiegate nei sistemi operativi delle funzionalità legate a quelle dei filesystem ma da essa indipendenti. Nel caso specifico per i due problemi di sopra si impiegano i volumi logici e i sistemi raid.

In questo capitolo tratteremo sia i filesystem che i "*logical volum*" e i sistemi "*raid*".

2.2 Le partizioni di un disco

Un filesystem, o un volume logico, o un sistema raid, viene costituito su una zona del disco chiamata "*partizione*" che viene creata tramite appositi programmi di partizionamento, quali "*fdisk*" o "*parted*". Il partizionamento di un disco è dunque una delle prime operazioni durante l'installazione di un sistema, in quanto senza di essa nessun dato potrà essere salvato sul disco stesso.
Esistono vari tipi di partizionamento, differenti a seconda del sistema operativo, quali quello "Solaris Sparc" e quello "i386", che prendono il nome di "*layout*" del disco.
Nei sistemi i386, quali quelli del corso, il layout è dato da un numero variabile di partizioni (anche solo una per l'intero disco), che possono a loro volta essere primarie o estese. Il numero di partizioni primarie che possono essere create su uno stesso disco è limitato a 4, mentre per le estese questo valore è molto più alto, in quanto una o più partizioni estese si possono appoggiare sulla stessa partizione primaria.
E' doveroso non eseguire un "*overlapping*" delle partizioni, in quanto poi ciascuna di esse sarà sede di un filesystem (o lv o raid) indipendente, e dunque tale operazione compromette i dati che verranno poi salvati su di essi. Si dovrà porre attenzione all'impostazione dei blocchi di disco di inizio e fine delle partizioni.
Queste partizioni sono una suddivisione a blocchi del disco, che viene dunque visto dal sistema come se fosse composto di più unità logiche indipendenti. Ad esempio, ipotizzando di partizionare il primo disco di sistema in 3 partizioni, il sistema individuerà 4 dispositivi differenti:

```
/dev/sda          device dell'intero disco (a → primo disco)
/dev/sda1             ''      della prima partizione
/dev/sda2             ''         ''     seconda     ''
/dev/sda3             ''         ''     terza       ''
```

Vediamo nel dettaglio il comando fdisk per create delle partizioni sul disco.
Digitando:

```
fdisk /dev/sda
```

appare il seguente messaggio:

```
The number of cylinders for this disk is set to 24321.
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., old versions of LILO)
2) booting and partitioning software from other OSs
     (e.g., DOS FDISK, OS/2 FDISK)

Comando (m per richiamare la guida):
```

e digitando "*m*":

```
Azione comando
   a  Cambia bootable flag
   b   modifica di bsd disklabel
   c   cambia il flag compatibile con il dos
   d   cancellazione di una partizione
   l   elenco dei tipi di partizione conosciuti
   m   stampa di questo menu
   n   aggiunta di una nuova partizione
   o   creazione di una nuova tabella delle partizioni DOS vuota
   p   stampa della tabella delle partizioni
   q   uscita senza salvataggio delle modifiche
   s   creazione di una nuova disklabel Sun vuota
   t   modifica l'id di sistema di una partizione
   u   modifica delle unità di visualizzazione/di immissione
   v   verifica la tabella delle partizioni
   w   scrivi la tabella su disco ed esci
   x   ulteriori funzioni (solo per esperti)
```

I comandi maggiormente impiegati sono "*p*", "*n*", "*t*" e naturalmente il "*w*".
Esempio di out del comando "*p*":

```
Disco /dev/sda: 250.1 GB, 250059350016 byte
255 testine, 63 settori/tracce, 30401 cilindri
Unità = cilindri di 16065 * 512 = 8225280 byte
Identificativo disco: 0x0009a286

Dispositivo Boot      Start         End      Blocks   Id  System
/dev/sda1   *           1          31      248976   83  Linux
/dev/sda2              32       19240   154296292+  8e  Linux LVM
```

dove si vede che vi sono due partizioni, "*sda1*" e "*sda2*", con la prima impostata come partizione di boot e tipo "Linux" (per filesystem ext3), la seconda, molto più grande, di tipo "*lvm*" (per la creazione di un logical volume su di essa).
Nel caso di partizioni per un sistema in raid, con boot loader "*grub*", la configurazione sarebbe ad esempio stata:

```
fdisk /dev/sda
         Device Boot   Start      End      Blocks   Id  System
      /dev/sda1   *       1       31      248976   fd  Linux raid autodetect
      /dev/sda2           32    17873   143315865   fd  Linux raid autodetect

fdisk /dev/sdb
         Device Boot   Start      End      Blocks   Id  System
      /dev/sdb1   *       1       31      248976   fd  Linux raid autodetect
      /dev/sdb2           32    17873   143315865   fd  Linux raid autodetect
```

in quanto il primo raid è impiegato per il solo filesystem "*/boot*", mentre il secondo per il logical volume su raid. Vedremo meglio questi concetti più avanti.

Se volessimo estendere una partizione come potremo procedere?
Per rispondere a questa domanda dovremo prima di tutto sapere se ci sia spazio libero dopo la partizione da estendere, in quanto, anche se il disco avesse spazio non partizionato ma che non si trovi appena dopo la fine della partizione da estendere, l'operazione non sarebbe possibile, in quanto non è possibile assegnare alla stessa partizione porzioni di disco non contigue.
Nel caso lo spazio libero sia invece contiguo potremo: avviare con un cd di boot di amministrazione, cancellare la partizione, ricrearla più grande, riavviare sempre da cd ed eseguire una espansione del logical volume o del filesystem attestato sulla partizione precedentemente più piccola.
Prima di eseguire queste operazioni è sempre meglio fare dei backup del sistema.

Le operazioni di estensione di una partizione dovrebbero comunque essere molto rare (tranne che per sistemi virtuali) in quanto, se disegnate bene dall'inizio, non dovrebbe più esserci spazio libero sul disco fisico. Dal momento che l'obbiettivo finale è sempre l'espansione del filesystem, vi sono altri modi per farlo senza passare per le partizioni.

Vediamo ora a grandi linee il layout di un disco Solari Sparc.
A titolo di esempio l'output del comando "*prtvtoc*" è ad esempio:

```
* /dev/rdsk/c2t0d0s2 partition map
*
* Dimensions:
*     512 bytes/sector
*      64 sectors/track
*      64 tracks/cylinder
*    4096 sectors/cylinder
*   51715 cylinders
*   51713 accessible cylinders
*
* Flags:
*   1: unmountable
*  10: read-only
*
* Unallocated space:
*       First     Sector    Last
```

```
*       Sector     Count    Sector
*   211812352      4096 211816447
*
*                          First     Sector    Last
* Partition  Tag  Flags    Sector     Count    Sector  Mount Directory
       0      2    00          0    262144    262143
       1      3    01     262144    262144    524287
       2      5    01          0 211816448 211816447
       7      8    00     524288 211288064 211812351   /extdsk
```

dal quale si nota la presenza di 4 partizioni o "*slice*", massimo 7 su Sparc, in cui la partizione "*2*" di *tag* "*5*" **non è usabile** in quanto, come in tutti i sistemi Sparc identifica l'intero disco e overlappa tutte le altre partizioni.

2.3 I sistemi in raid

Il termine "*raid*" è un acronimo di "*redundant array of inexpensive disks*" o, a volte, di "*redundant array of independent disks*". Tramite questa tecnologia è possibile dividere e/o replicare il salvataggio dei dati su più hard disk drives; in questo modo è possibile rispondere alla duplice richiesta di ridondanza e di performance dei sottosistemi di disco.
Le modalità di lavoro delle tecnologie raid sono indicate da un numero che segue l'acronimo raid stesso, ad esempio: raid0, raid1, …, raid5, etc. Ciascuna di queste opera sul sottosistema disco in maniera da affrontare le predette richieste, che risultano in antitesi tra loro.

Le modalità con le quali è possibile implementare una qualsivoglia tecnologia raid sono due: hardware (composta a sua volta di una parte anche software) e software. Le differenze tra queste ultime sono date dal costo della soluzione e dalle relative performance che sono in grado di raggiungere. Infatti tramite l'aggiunta di un sistema in raid hardware è possibile demandare ad un processore dedicato il lavoro di gestione delle ridondanze, scaricando di tale incombenza il processore dello stesso server, inoltre, di solito, un sistema raid hardware presenta anche una memoria cache che ne aumenta le performance, in quanto i dati non vengono direttamente scritti su disco ma successivamente alla richiesta, scritti su cache, con tempi di operatività molto inferiori (questo presuppone l'impiego nel sistema raid hardware di batterie tampone che in caso di mancanza della corrente mantengano i dati della cache non ancora scritti su disco). Un sistema di dischi esterni ad un server, quale uno di quelli presenti in sala macchine, è di solito un sistema raid hardware.
In un sistema raid hardware viene salvaguardata l'operatività del sistema di dischi anche a scapito di rotture hardware dei dischi stessi, senza inficiare eccessivamente sulle performance. Per via della natura hardware del sistema la soluzione è strettamente dipendente dal sistema stesso.
La seconda modalità con la quale è possibile implementare un sistema raid è quella software. In questo caso il processore del server stesso si occupa della gestione del raid secondo la politica scelta, diminuendo le performance del sistema stesso. Tale diminuzione può comunque essere accettata se si ragiona in termini di costi ridotti della soluzione, se non nulli, e in termini di conseguente aumento di robustezza ai guasti. Infatti una sua messa in produzione necessità solo di una configurazione iniziale. Il vantaggio di una siffatta soluzione è dato anche dalla portabilità del dato tra ambienti differenti.

Ad onor del vero esiste una terza via intermedia per l'implementazione di un sistema raid, detta "*fakeRaid*", eseguita tramite un componete firmware presente nel controller delle schede madri. Questa soluzione non offre però dei significativi aumenti di performance, ma anzi presenta gli stessi problemi di portabilità dei sistemi raid hardware. Queste soluzioni sono divenute molto di moda su sistemi a basso costo quali quelli desktop.

Vediamo ora nei dettagli le modalità di lavoro delle tecnologie raid.

- Raid 0 "*striped disks*": i dati da salvare vengono distribuiti su più dischi in maniera da aumentare le performance del sistema in quanto vengono impiegate tutte le meccaniche dei dischi nel salvataggio. Questa soluzione non presenta però nessuna tolleranza ai guasti, anzi la rottura di un solo disco dell'array pregiudica i dati salvato su tutto il sistema raid. Richiede un minimo di 2 dischi, non presenta nessuna perdita di efficienza di storage.
- Raid 1 "*mirror*": i dati da salvare vengono memorizzati su tutti i dischi del raid. In questo modo ogni disco dell'array contiene una copia consistente di tutti i dati, da qui il nome di mirror. Richiede un minimo di 2 dischi, l'efficienza di storage è ridotta al 50% con 2 dischi, al 33% con 3, etc.
- Raid 3 e 4 "striped with dedicated parity": i dati da salvare vengono distribuiti su più dischi come nel caso del raid 0. Uno dei dischi è impiegato per salvare le informazioni di parità con le quali è possibile risalire alla rottura di uno dei dischi dell'array (tranne che per quello di parità). Nella versione 4 i dati vengono suddivisi in "chunks" di valore prefissato e configurabile in modo da ottenere le migliori performance possibili. Richiede un minimo di 3 dischi, l'efficienza di storage è ridotta al 66% con 3 dischi e al 83% con 6. Le performance del sistema risultano penalizzate in presenza di più operazioni di lettura/scrittura simultanee e indipendenti.
- Raid 5: i dati da salvare e le informazioni di parità per la ricostruzione degli stessi vengono distribuiti su tutti i dischi. La rottura di un qualunque disco dell'array non pregiudica ne i dati ne il funzionamento del sistema, le informazioni di parità presenti sugli altri dischi permettono di risalire al dato che era stato salvato sul disco degradato (a discapito però di performance molto ridotte); nel caso di una ulteriore rottura di disco i dati dell'intero array sono definitivamente compromessi. Per tale motivo questa modalità è spesso impiegata con un ulteriore disco di "*hot spare*" che sostituisce il primo disco rotto; in questo modo il sistema è in grado di funzionare anche dopo due rotture contemporanee e su singola rottura le performance rimangono inalterate. Richiede un minimo di 3 dischi. L'efficienza di storage con hot spare e 4 dischi è ridotta al 50% e al 66% con 6 dischi.
- Raid 6: i dati da salvare e le informazioni di parità vengono distribuiti su tutti i dischi. Rispetto al raid 5 la parità è in effetti una doppia parità, che permette il funzionamento del sistema anche con due rotture concomitanti. Richiede un minimo di 4 dischi.
- Raid 10 "*mirror and striped*": questa modalità è in effetti una combinazione delle precedenti 1 e 0, in prima battuta viene eseguito un lavoro di mirror e su questo viene applicato uno striped. Richiede un numero di dischi pari con un minimo di 4, l'efficienza di storage è sempre ridotta del 50%. Con 6 dischi viene eseguito il mirror su due gruppi di 3 dischi ciascuno, e su ciascun gruppo viene eseguito lo striped dei

dati. Le performance sono direttamente dipendenti dal numero di dischi, in quanto non vi sono informazioni di parità da calcolare.

Le modalità più impiegate sono quella in raid1, raid5 e da pochi anni raid10. Il raid1 è impiegato di solito su 2 soli dischi, quelli di sistema, mentre il raid5 sui dischi dei dati utente. Il costo sempre minore dei dischi fisici ha permesso di sostituire spesso la modalità raid5 con la raid10, molto più performante.

Dal momento che ogni sistema raid hardware presenta caratteristiche simili, ma differenti da produttore a produttore, tratteremo nel proseguo del paragrafo le sole tecnologie software (gestite dal kernel Linux).

Operativamente, ricordando che il nostro obiettivo finale è sempre il filesystem, è necessario creare delle partizioni, ciascuna su un disco differente, di tipologia raid su cui attestare poi la modalità di raid vera e propria. In questo caso, il sistema identificherà oltre al classici device "*sda*, *sda1*, *sda2*, etc." un device:

```
/dev/md0
/dev/md1
```

etc., a seconda del numero di raid costituiti. Il numero dopo "*md*" **non identifica la modalità di raid**.
Su tali device, come su "*sda1*, etc." si potrà poi implementare un filesystem vero e proprio, o un logical volume.
Il kernel Linux si occuperà automaticamente delle operazioni sul raid. Come ad esempio delle ricostruzioni del volume, leggendo le informazioni del o dei raid direttamente dai dischi. E' possibile salvare tali informazioni anche sul file "*/etc/mdadm/mdadm.conf*", e in questo modo si ha la possibilità di impiegare i comandi "*mdadm*" di gestione e amministrazione del o dei raid in maniera sintetica.
Per visualizzare le informazioni statistiche dei raid si esegua:

```
cat /proc/mdstat
```

il cui output in caso di non errori è del tipo:

```
Personalities : [raid1]
md1 : active raid1 sda2[0] sdb2[1]
      143315776 blocks [2/2] [UU]

md0 : active raid1 sda1[0] sdb1[1]
      248896 blocks [2/2] [UU]

unused devices: <none>
```

Nell'esempio del sistema di sopra identifichiamo due raid, md0 e md1, entrambi di tipo raid1, creati su due partizioni di due dischi:

```
md0   →     sda1 + sdb1
md1   →     sda2 + sdb2
```

Che senso hanno due raid con due dischi?

Sul primo, molto piccolo, è direttamente attestato il filesystem "*/boot*" sul secondo tutto tramite logical volume tutto il resto "/". Questo è necessario nel caso che si voglia gestire il sistema via logical volume e boot loader grub. Questo ultimo infatti non permette di eseguire il boot da lv, e dunque tale filesystem va tolto dal lv ma comunque messo in raid.

Per creare un raid su un sistema è necessario disporre del numero di dischi minimo occorrenti per il sistema raid che si vuole creare. Creare le partizioni sui dischi di tipologia raid (type "*fd*") e passare alla costruzione vera e propria del raid.

Esempio: creazione di un raid 5 su 4 dischi, e test delle performance con dd.

```
mdadm -C /dev/md3 -n 4 -l 5 /dev/sdc1 /dev/sdd1 /dev/sde1 /dev/sdf1

dd bs=1024k if=/dev/zero of=/dev/md3
```

il cui output da una performance di 20.9 MB/s. Lo stesso comando dato su un raid 0 creato sugli stessi dischi fornisce delle performance di 53.5 MB/s

Per inserire le informazioni del nostro raid sul file di configurazione "*mdadm.conf*" si esegua:

```
mdadm --detail --brief /dev/md3 >> /dev/mdadm/mdadm.conf
```

Una configurazione tipo di tale file è:

```
# mdadm.conf
#
# Please refer to mdadm.conf(5) for information about this file.
#

# by default, scan all partitions (/proc/partitions) for MD superblocks.
# alternatively, specify devices to scan, using wildcards if desired.
DEVICE partitions

# auto-create devices with Debian standard permissions
CREATE owner=root group=disk mode=0660 auto=yes

# automatically tag new arrays as belonging to the local system
HOMEHOST <system>

# instruct the monitoring daemon where to send mail alerts
MAILADDR root

# definitions of existing MD arrays
ARRAY /dev/md0 level=raid1 num-devices=2
UUID=aadf51fe:21aac6f5:e4288264:1853faa1
ARRAY /dev/md1 level=raid1 num-devices=2
UUID=1b37d161:2d0a695d:44bdc261:95555b41

# This file was auto-generated on Wed, 05 Sep 2007 05:53:59 +0000
# by mkconf $Id: mkconf 261 2006-11-09 13:32:35Z madduck $
```

Di seguito vari comandi per amministrare un raid già creato:

mdadm -S /dev/md3	stoppa il raid (il kernel al reboot riavvia i raid che individua sul sistema)

mdadm -A /dev/md3	"assemble", riavvia un raid stoppato. Se la configurazione del raid non è stata salvata sul file "*mdadm.conf*" è necessario indicare anche tutti i device che ne fanno parte
mdadm /dev/md3 -f /dev/sdc1	imposta il device "sdc1" fail

```
-bash: q: command not found
root@ubuntu:~#
root@ubuntu:~#
root@ubuntu:~#
root@ubuntu:~#
root@ubuntu:~# mdadm /dev/md3 -f /dev/sdc1
[ 7944.359070] raid1: Disk failure on sdc1, disabling device.
mdadm: set /dev/sdc1 faulty in /dev/md3
root@ubuntu:~#
root@ubuntu:~#
root@ubuntu:~# more /proc/mdstat
Personalities : [linear] [multipath] [raid0] [raid1] [raid6] [raid5] [raid4] [ra
id10]
md3 : active raid1 sdf1[3] sde1[2] sdd1[1] sdc1[4](F)
      1044096 blocks [4/3] [_UUU]

md1 : active raid1 sda2[0] sdb2[1]
      3943872 blocks [2/2] [UU]

md0 : active raid1 sda1[0] sdb1[1]
      248896 blocks [2/2] [UU]

unused devices: <none>
root@ubuntu:~# _
```

mdadm /dev/md3 -r /dev/sdc1	rimuove il device "sdc1" (non possibile se prima non viene impostato il device in fail)
mdadm /dev/md3 -a /dev/sdc1	aggiunge il device "sdc1" al raid
mdadm /dev/md3 --re-add /dev/sdc1	riaggiunge il device "sdc1" che era stato rimosso al raid

```
Continue creating array? (y/n)
Continue creating array? (y/n) y
[  161.905457] raid5: raid level 5 set md3 active with 3 out of 4 devices, algor
ithm 2
mdadm: array /dev/md3 started.
root@ubuntu:~#
root@ubuntu:~#
root@ubuntu:~#
root@ubuntu:~#
root@ubuntu:~# more /proc/mdstat
Personalities : [linear] [multipath] [raid0] [raid1] [raid6] [raid5] [raid4] [ra
id10]
md3 : active raid5 sdf1[4] sde1[2] sdd1[1] sdc1[0]
      3132288 blocks level 5, 64k chunk, algorithm 2 [4/3] [UUU_]
      [===>.................]  recovery = 18.4% (193408/1044096) finish=1.3min s
peed=10179K/sec

md1 : active raid1 sda2[0] sdb2[1]
      3943872 blocks [2/2] [UU]

md0 : active raid1 sda1[0] sdb1[1]
      248896 blocks [2/2] [UU]

unused devices: <none>
root@ubuntu:~# _
```

mdadm -Q --detail /dev/md3	visualizza in dettaglio i dati del raid “md3”

2.4 I logical volume

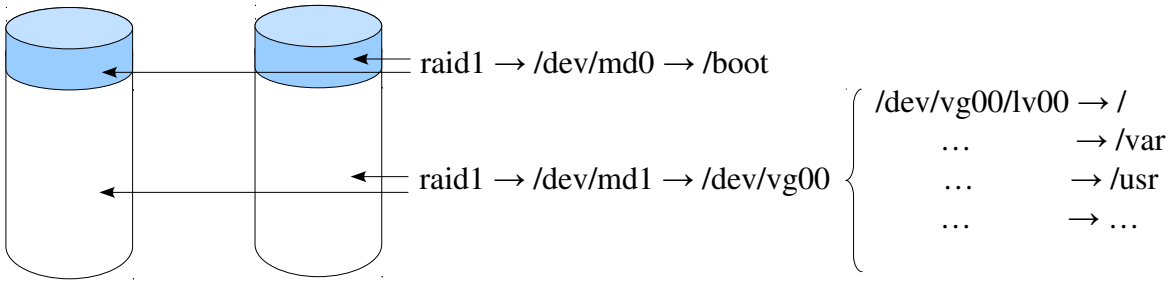
In un sistema di storage il “*logical volume managment*” (LVM) è un metodo che permette di allocare lo spazio disponibile in una modalità più flessibile rispetto alla gestione classica delle partizioni. Infatti tramite questa organizzazione dei dati è possibile concatenare tra loro varie partizioni, o dischi, estenderne quelle esistenti o spostarle tra i dispositivi fisici, potenzialmente senza interruzioni del sistema. Una delle funzionalità molto utili è inoltre quella che permette di creare delle “*snapshot*” di un volume in modo da congelare ad un certo istante lo stato dei dati memorizzati su di esso.
La struttura dati su cui si basa l'organizzazione LVM è fondata sui volumi fisici (PVs o “*physical volumes*”) costituiti a loro volta da sequenze di “*chunks*” (PEs o “*physical extents*”). Questi ultimi possono avere dimensioni fisse, come nei sistemi UP-UX e Linux, o variabili, come nei sistemi Veritas. Un insieme di PVs (con i relativi PEs) viene assegnato ad un determinato “*volum group*” o VG sul quale poi vengono creati i “*logical volume*” o LVs veri e propri. Ricordando sempre che l'obbiettivo finale è quello di crezione di un filesystem, questo viene poi costituito su un dato LV.
Come detto precedentemente è dunque possibile:

- estendere i PEs assegnati ad un determinato LV in modo da renderli disponibili al filesystem sottostante
- migrare un LV o un intero VG tra dispositivi fisici differenti
- eseguire una snapshot di un LV in modo da congelare la situazione del filesystem ad un certo istante. Si noti che le operazioni su di esso proseguiranno su un LV si snapshot assegnato al LV precedente; in questo modo è possibile ad esempio eseguire un backup del filesystem “quasi a caldo”, accettando un tempo minimo di fermo dell'applicazione necessario per la creazione della struttura dati di snapshot
- estendere i PEs assegnati ad un determinato VG tramite l'agiunta di PVs in modo da rendere disponibile ulteriore spazio libero per i LVs

Cosa è operativamente un PV?
Non è altro che un disco fisico, o una partizione di un disco se partizionato, o un dispositivo raid se è impiegata tale tecnologia. In questa ottica si può da subito osservare che un sistema raid0 è analogo ad volum group.

Come procediamo nell'organizzazione dello storage?
Creiamo da prima le nostre partizioni sui dischi a disposizione pensando ai futuri raid da assestare. Assestiamo su tali raid il filesystem “*/boot*” (in caso si usi grub) e i volum group necessari. Creiamo sui volum group i logical volume e su questi i relativi filesystem. Montiamo poi i filesystem sui relativi punti di “*mount*”.

Esempio: due dischi con due sistemi raid, un volum group e un logical volume per ogni punto di mount necessario.



Esempio di output del comando "*vgdispaly*":

```
--- Volume group ---
VG Name               vg00
System ID
Format                lvm2
Metadata Areas        1
Metadata Sequence No  48
VG Access             read/write
VG Status             resizable
MAX LV                0
Cur LV                8
Open LV               8
Max PV                0
Cur PV                1
Act PV                1
VG Size               147,14 GB
PE Size               4,00 MB
Total PE              37669
Alloc PE / Size       21944 / 85,72 GB
Free  PE / Size       15725 / 61,43 GB
VG UUID               f85wYz-wuPR-tp6U-QXrV-unKD-BLbh-U9Tjjn
```

dal quale si legge che il VG ha 8 logical volume assestati su di esso, una dimensione di PE di 4 MB, una dimensione massima di 147.14 GB, uno spazio allocato ai lv di 85.72 GB, uno spazio libero di 61.43 GB.
Esempio di output del comando "*pvdisplay*":

```
--- Physical volume ---
  PV Name               /dev/sda2
  VG Name               vg00
  PV Size               147,15 GB / not usable 3,97 MB
  Allocatable           yes
  PE Size (KByte)       4096
  Total PE              37669
  Free PE               15725
  Allocated PE          21944
  PV UUID               rYD12p-D3RX-52cn-LCy1-sZae-wy3r-oweebr
```

dal quale si legge che il PV è direttamente assestato sulla partizione 2 del primo disco, è allocato dal volum group "vg00", ha un numero di 37669 PEs occupati e 15725 liberi. Si noti che le informazioni sono analoghe a quelle visualizzate dal comando vgdisplay.

Espansione di un VG.
L'espansione di un PV non ha nella maggior parte dei casi molto senso, in quanto c'è da chiedersi se sia possibile aumentare lo spazio di un disco, o quello di una partizione, se il partizionamento è

stato disegnato bene (ma in caso contrario forse serve una reinstallazione totale). Potrebbe essere utile in ambienti virtuali o in caso di storage esterni che permettano l'espansione dei volumi logici (LUN o "*logical unit number*") già costituiti. In entrambi i casi uno stop del server è necessario, **oltre che ad un backup dell'intero sistema**.
Se il PV da espandere e quello su cui è assegnato il sistema operativo è necessario eseguire il boot da un cd di ripristino. Le operazioni da fare sono:

1) spegnimento del server
2) aumento dello spazio del disco virtuale o dello spazio sulla LUN dello storage esterno
3) avvio da cd di ripristino
4) cancellazione della partizione sul disco allargato (che non occupa tutto lo spazio disponibile)
5) creazione della nuova partizione sul disco allargato
6) esecuzione del comando:

```
pvresize -v /dev/sda2
```

   per rendere disponibili i nuovi PVs della partizione

7) spegnimento e avvio del sistema principale
8) verifica dell'aumento di spazio disponibile tramite il comando *vgdisplay*

Spesso a tale operazione, molto distruttiva, è preferibile la creazione di un nuovo disco virtuale o di una nuova LUN dello storage e l'assegnazione di questa ultima al VG presente. Questa operazione non è distruttiva e permette un'operatività a caldo.

Riduzione di un LV.
Questa funzionalità è strettamente dipendente dalle features del filesystem, che se non permettono lo "*shrink*" dello stesso non sono applicabili.
E' sempre possibile eseguire un "*tar*" dei dati, creare un lv più piccolo e importare il "*tar*" sul nuovo filesystem.
Nel caso di filesystem che lo permettano (quale ext3) la sequenza di comandi è la seguente:

1) umount del filesystem
2) check del filesystem con "*e2fsck -f device*"
3) impostazione della nuova dimensione tramite "*resize2fs -p device size*"
4) check del filesystem con "*e2fsck -f device*"
5) riduzione del logical volume tramite "*lvreduce -L size device*"
6) mount del filesystem

E' importantissimo è non impostare una dimensione di lv inferiore a quella del filesystem.

Espansione di un filesystem.
Anche in questo caso l'espansione di un filesystem è possibile se disponibile tra le features del filesystem. Anche se per alcuni filesystem è possibile eseguire questa operazione online, è sempre consigliabile smontare prima il filesystem.
Le operazioni sono dunque le seguenti:

1) umount del filesystem
2) check del filesystem con "*e2fsck -f device*"
3) espansione del volume con "*lvextend -L size device*"
4) check del filesystem con "*e2fsck -f device*"
5) espansione del filesystem con "*resize2fs -p device*"
6) mount del filesystem

Alcune tipologie di filesystem non sono smontabili, quali lo "*/var*", **e il loro ridimensionamento è distruttivo per il sistema**. In questi casi è necessario eseguire il ridimensionamento tramite un sistema di recovery (cd di boot) e attivare il VG locale tramite i comandi:

```
vgscan
vgchange -ay vg00
```

Attivazione di una snapshot.
Una snapshot non è altro che un nuovo volume collegato ad un volume già esistente. Il comando da usare è dunque:

```
lvcreate -s -L dimensione -n nomeLVsnap deviceLVesistente
```

dove l'opzione "-s" identifica che è un volume di snapshot e la "dimensione" indica lo spazio allocato per la snapshot. Questo spazio deve essere sufficiente a contenere le modifiche operate sul volume fino al tempo di cancellazione della snapshot.
Per la sua deattivazione è impiegabile il comando:

```
lvremove deviceLVsnap
```

Il volume di snapshot è, se montato su una cartella, impiegabile per accedere alla fotografia dei dati all'instante di creazione del volume stesso, per questo è impiegato in operazioni di backup.

2.5 Operazioni sui filesystem

Una volta disegnato il nostro sistema di storage, partizioni, e/o raid e/o lvm, è giunto il momento di creare un filesystem vero e proprio. Il comando da impiegare è:

```
mkfs -t fstype device
```

dove con l'opzione "*-t*" indichiamo il tipo di filesystem da creare. In base a tale parametro verrà chiamato il comando apposito, es: "*mkfs.ext3*", "*mkfs.vfat*", etc.
Una delle opzioni molto importanti, anche che comunque dimensionata opportunamente anche come default, e il numero di "*inode*" assegnati al filesystem. Questo parametro è direttamente collegato al numero massimo di descrittori dei file e delle cartelle che è possibile creare su un filesystem. In questo modo è possibile un tuning che permetta di definire l'impiego vero e proprio del filesystem, ad esempio in un fs per un server mail è necessario un numero di inode molto alto se ogni mail di un utente è un file (e dunque per molti file piccoli), o basso se tutte le mail di un utente sono contenute nello stesso file. Questo parametro è impostabile solo in fase di creazione.

Vediamo l'output di esempio del comando "*mkfs.ext3 /dev/vg00/lvprova*":

```
mke2fs 1.41.9 (22-Aug-2009)
Etichetta del filesystem=
Tipo SO: Linux
Dimensione blocco=4096 (log=2)
Dimensione frammento=4096 (log=2)
65536 inode, 262144 blocchi
13107 blocchi (5.00%) riservati per l'utente root
Primo blocco dati=0
Maximum filesystem blocks=268435456
8 gruppi di blocchi
32768 blocchi per gruppo, 32768 frammenti per gruppo
8192 inode per gruppo
Backup del superblocco salvati nei blocchi:
      32768, 98304, 163840, 229376

Scrittura delle tavole degli inode: fatto
Creating journal (8192 blocks): fatto
Scrittura delle informazioni dei superblocchi e dell'accounting del
filesystem: fatto

Questo filesystem verrà automaticamente controllato ogni 36 mount, o
180 giorni, a seconda di quale venga prima. Usare tune2fs -c o -i per
cambiare.
```

Molto importante e la parte relativa alla memorizzazione delle copie del "*superblock*", che è quel campo del filesystem che permette di avere informazioni di base sul filesystem stesso, quali tipo di fs, dimensione, stato e struttura dei metadati. Ne vengono fatte delle copie per evitarne quanto più possibile possibile perdita.

Altri comandi molo utili sono:

fsck	controlla e ripara un filesystem
e2fsck	controllo e riparo di filesystem ext2, ext3 e ext4
tune2fs	permette di modificare i parametri di un filesystem per un tuning dello stesso
dumpe2fs	esegue un dump del superblock e delle informazioni dei "*block group*" presenti sul filesystem (solo ext)

Dopo un crash del sistema, o per via di errori sul filesystem, il comando "*fsck*" viene usato per recuperare i blocchi di filesystem non più collegati a nessun "*inode*" ma che risultano ancora allocati. Questi blocchi vengono salvati in una particolare directory chiamata "*lost+found*" che è presente in ciascun filesystem.
Se ad esempio abbiamo 3 filesystem, "*/*" e "*/boot*" e "*/tmp*", avremo 3 cartelle "*lost+found*" precisamente in:

```
/boot
/boot/lost+found
/…
/etc
/…
/lost+found
/tmp
/tmp/lost+found
```

/…

All'avvio di un sistema Linux vengono montati i filesystem indicati nel file "*/etc/fstab*" visualizzato di seguito:

```
# /etc/fstab: static file system information.
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
proc            /proc           proc    defaults        0       0
/dev/mapper/vg00-lv00 /         ext3    defaults,errors=remount-ro 0 1
/dev/sda1       /boot           ext3    defaults        0       2
/dev/mapper/vg00-lv05 /home     ext3    defaults        0       2
/dev/mapper/vg00-lv04 /opt      ext3    defaults        0       2
/dev/mapper/vg00-lv01 /tmp      ext3    defaults        0       2
/dev/mapper/vg00-lv02 /usr      ext3    defaults        0       2
/dev/mapper/vg00-lv03 /var      ext3    defaults        0       2
/dev/mapper/vg00-lvswap none    swap    sw              0       0
/dev/hda        /media/cdrom0   udf,iso9660 user,noauto     0       0
/dev/fd0        /media/floppy0  auto    rw,user,noauto  0       0
# For Oracle
/dev/mapper/vg00-lv06   /oradata    ext3  defaults    0     2
/dev/mapper/vg00-lv07   /backup         ext3    defaults        0       2
```

in cui sono indicati i dispositivi, i relativi punti di mount, il tipo di filesystem, le opzioni di mount e l'opzione di priorità per il comando "*fsck*".
Questo file viene compilato automaticamente durante la fase di installazione, ma nel caso vengano successivamente creati nuovi filesystem, affinché il sistema ne esegua il mount all'avvio è necessario aggiungerli allo stesso.

2.6 Organizzazione delle directory

L'organizzazione delle directory di un sistema Linux varia da distribuzione a distribuzione. Vi sono però alcune directory di impiego comune in tutte le distribuzioni, la cui organizzazione segue uno schema Unix-like. Vediamo un esempio di organizzazione:

/bin	adibita ai file binari (caricati nel sistema)
/boot	adibita ai file di boot del sistema e alla configurazione del boot loader
/dev	adibita ai device del sistema
/etc	adibita ai file di configurazione del sistema e degli applicativi
/home	adibita ai dati degli utenti di sistema
/lib	adibita ai file di libreria
/mnt	adibita al montaggio di dispositivi esterni (molto usata in ambienti Unix, meno in Linux)
/opt	adibita alle applicazioni non di sistema (molto usata in ambienti Unix, meno su Linux)
/sbin	adibita ai file binari sicuri (ad esecuzione di root e in single user)
/srv	adibita agli applicativi server (usata nei sistemi Suse, meno in quelli Debian)
/usr	adibita ai dati statici (applicazioni, manuali, etc.)
/var	adibita ai dati che variano nel tempo, come i log,

Per ciascuna di queste (tranne che per la "/mnt"), è possibile creare un logical volume che la contenga, anche se solitamente alcune vengono raggruppate in unici lv. E' utile separare tra loro la "/, /boot, /home, /opt, /srv, /usr, /var".

3 Installazione di una distribuzione Linux

Prima di procedere con l'installazione di una distribuzione Linux e bene chiarire alcuni aspetti fondamentali:

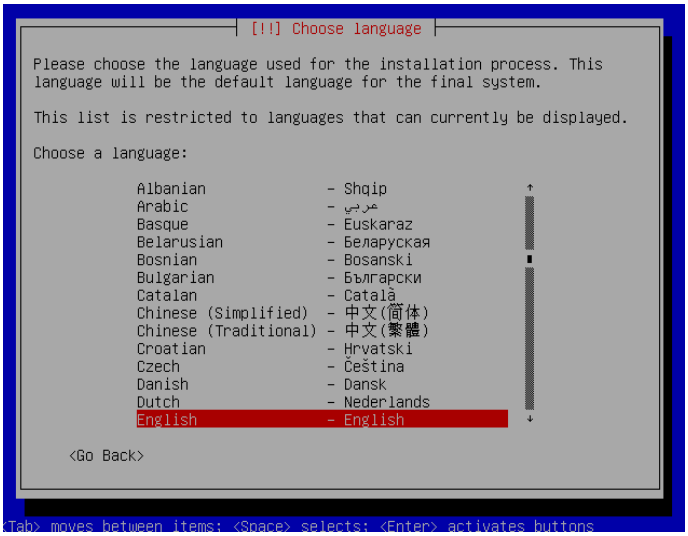
1. sperimentare, prima di ogni installazione vera e propria, un sistema in live (in modo da non dover cancellare e/o modificare il sistema originario) per verificare le caratteristiche della distribuzione o versione scelta. In alternativa è anche possibile impiegare un sistema virtuale, che ci permette di utilizzare una distribuzione Linux su un sistema già presente, senza modificarlo, a scapito di alcune funzionalità quali quelle grafiche; a tale riguardo è necessario avere anche dell'altro software, quale VMware Server (di libero uso) o Sun Virtual Box, per creare il sistema virtuale;
2. se possibile impiegare un hard disk libero, cioè senza un precedente sistema operativo già installato sopra. In caso di installazione sullo stesso hard disk del sistema operativo già installato, eseguire prima di tutto un backup del disco o almeno il salvataggio dei propri dati, in modo da non perderli nel caso di errori durante l'installazione.

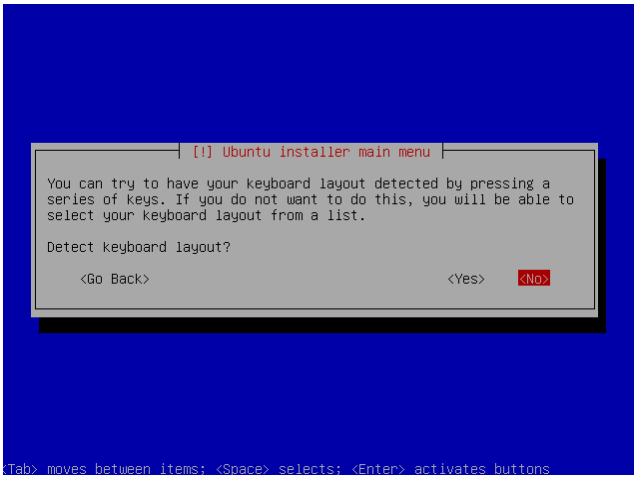
In ogni caso è vivamente sconsigliato procedere con l'installazione su un sistema già funzionante se non si ha padronanza con l'installazione di un sistema operativo.

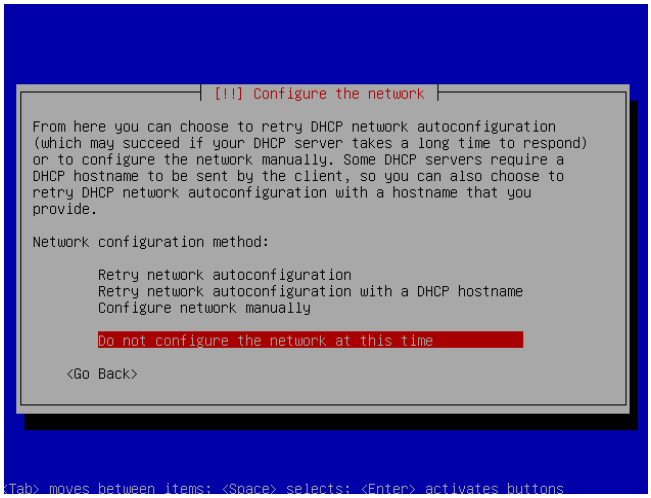
Come versione Linux di test per il corso è stata scelta la distribuzione server Ubuntu 8.04 LTS, per le sue doti di semplicità, completezza, libero utilizzo e rispondenza ai requisiti del corso. Questa è la stessa tipologia di sistema installata presso i server del Ced, per i quali non sia richiesto un diverso sistema operativo a fronte di particolari specifiche delle applicazioni.
La scelta di una versione server, come vedremo, non focalizza l'attenzione solo su tale tipologia in quanto tra una distribuzione server e una desktop variano semplicemente le tipologie di alcuni pacchetti installati, che è comunque possibile installare.

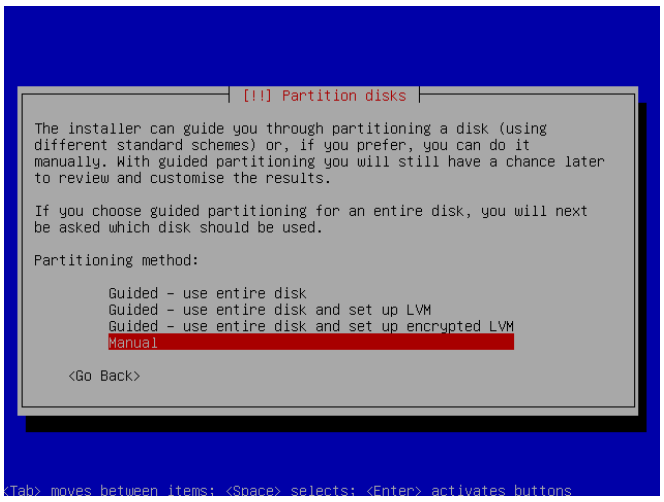
In questi appunti non sono stati inseriti i passi delle fasi di installazione in quanto non necessarie per la prosecuzione del corso. Tali passi riflettono in ogni caso principalmente gli argomenti trattati nel capitolo relativo ai filesystem.

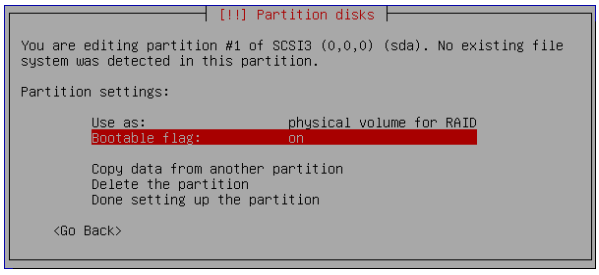
Di seguito sono elencate alcune delle principali schermate presentate durante le fasi di installazione:

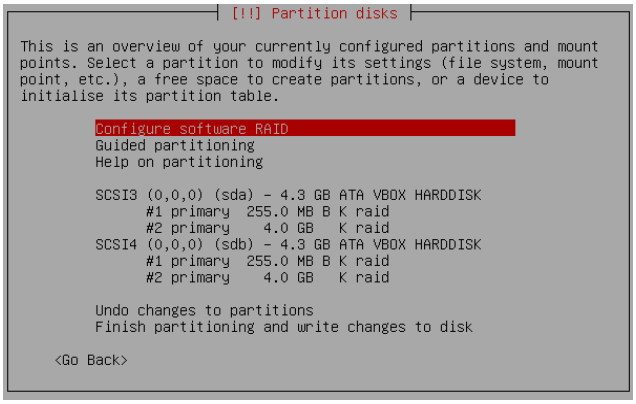

[!!] Choose language
Please choose the language used for the installation process. This language will be the default language for the final system.
This list is restricted to languages that can currently be displayed.
Choose a language:
Albanian - Shqip
Arabic - عربي
Basque - Euskaraz
Belarusian - Беларуская
Bosnian - Bosanski
Bulgarian - Български
Catalan - Català
Chinese (Simplified) - 中文(简体)
Chinese (Traditional) - 中文(繁體)
Croatian - Hrvatski
Czech - Čeština
Danish - Dansk
Dutch - Nederlands
English - English
<Go Back>
<Tab> moves between items; <Space> selects; <Enter> activates buttons

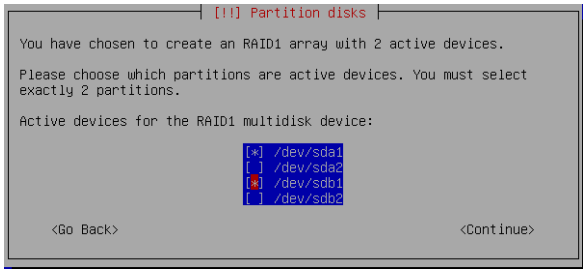

[!] Ubuntu installer main menu
You can try to have your keyboard layout detected by pressing a series of keys. If you do not want to do this, you will be able to select your keyboard layout from a list.
Detect keyboard layout?
<Go Back>
<Yes>
<No>
<Tab> moves between items; <Space> selects; <Enter> activates buttons

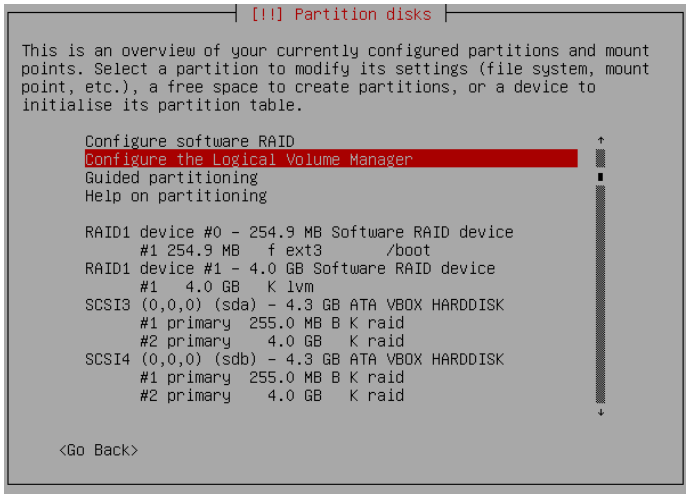

[!!] Configure the network
From here you can choose to retry DHCP network autoconfiguration (which may succeed if your DHCP server takes a long time to respond) or to configure the network manually. Some DHCP servers require a DHCP hostname to be sent by the client, so you can also choose to retry DHCP network autoconfiguration with a hostname that you provide.
Network configuration method:
Retry network autoconfiguration
Retry network autoconfiguration with a DHCP hostname
Configure network manually
Do not configure the network at this time
<Go Back>
<Tab> moves between items; <Space> selects; <Enter> activates buttons

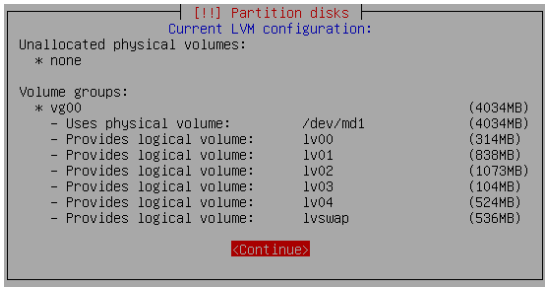

[!!] Partition disks
The installer can guide you through partitioning a disk (using different standard schemes) or, if you prefer, you can do it manually. With guided partitioning you will still have a chance later to review and customise the results.
If you choose guided partitioning for an entire disk, you will next be asked which disk should be used.
Partitioning method:
Guided - use entire disk
Guided - use entire disk and set up LVM
Guided - use entire disk and set up encrypted LVM
Manual
<Go Back>
<Tab> moves between items; <Space> selects; <Enter> activates buttons

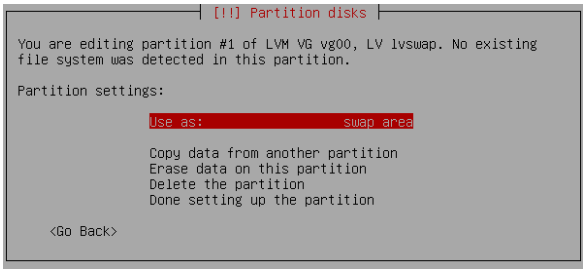

[!!] Partition disks
You are editing partition #1 of SCSI3 (0,0,0) (sda). No existing file system was detected in this partition.
Partition settings:
Use as: physical volume for RAID
Bootable flag: on
Copy data from another partition
Delete the partition
Done setting up the partition
<Go Back>

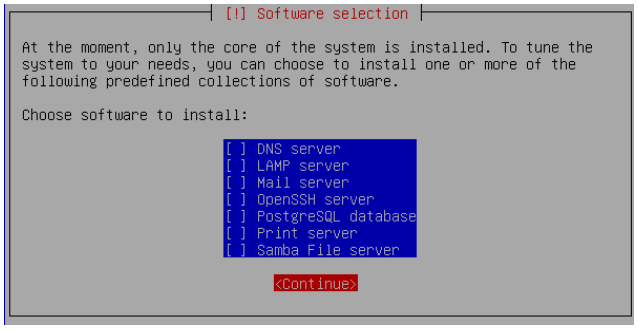

[!!] Partition disks
This is an overview of your currently configured partitions and mount points. Select a partition to modify its settings (file system, mount point, etc.), a free space to create partitions, or a device to initialise its partition table.
Configure software RAID
Guided partitioning
Help on partitioning
SCSI3 (0,0,0) (sda) - 4.3 GB ATA VBOX HARDDISK
#1 primary 255.0 MB B K raid
#2 primary 4.0 GB K raid
SCSI4 (0,0,0) (sdb) - 4.3 GB ATA VBOX HARDDISK
#1 primary 255.0 MB B K raid
#2 primary 4.0 GB K raid
Undo changes to partitions
Finish partitioning and write changes to disk
<Go Back>

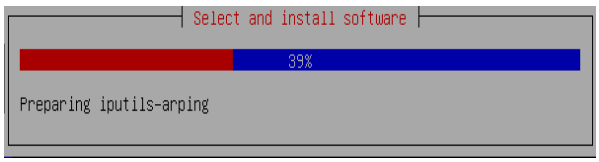

[!!] Partition disks
You have chosen to create an RAID1 array with 2 active devices.
Please choose which partitions are active devices. You must select exactly 2 partitions.
Active devices for the RAID1 multidisk device:
[*] /dev/sda1
[ ] /dev/sda2
[*] /dev/sdb1
[ ] /dev/sdb2
<Go Back>
<Continue>


[!!] Partition disks
This is an overview of your currently configured partitions and mount points. Select a partition to modify its settings (file system, mount point, etc.), a free space to create partitions, or a device to initialise its partition table.
Configure software RAID
Configure the Logical Volume Manager
Guided partitioning
Help on partitioning
RAID1 device #0 - 254.9 MB Software RAID device
#1 254.9 MB f ext3 /boot
RAID1 device #1 - 4.0 GB Software RAID device
#1 4.0 GB K lvm
SCSI3 (0,0,0) (sda) - 4.3 GB ATA VBOX HARDDISK
#1 primary 255.0 MB B K raid
#2 primary 4.0 GB K raid
SCSI4 (0,0,0) (sdb) - 4.3 GB ATA VBOX HARDDISK
#1 primary 255.0 MB B K raid
#2 primary 4.0 GB K raid
<Go Back>


[!!] Partition disks
Current LVM configuration:
Unallocated physical volumes:
* none
Volume groups:
* vg00 (4034MB)
- Uses physical volume: /dev/md1 (4034MB)
- Provides logical volume: lv00 (314MB)
- Provides logical volume: lv01 (838MB)
- Provides logical volume: lv02 (1073MB)
- Provides logical volume: lv03 (104MB)
- Provides logical volume: lv04 (524MB)
- Provides logical volume: lvswap (536MB)
<Continue>


[!!] Partition disks
You are editing partition #1 of LVM VG vg00, LV lvswap. No existing file system was detected in this partition.
Partition settings:
Use as: swap area
Copy data from another partition
Erase data on this partition
Delete the partition
Done setting up the partition
<Go Back>


[!] Software selection
At the moment, only the core of the system is installed. To tune the system to your needs, you can choose to install one or more of the following predefined collections of software.
Choose software to install:
[ ] DNS server
[ ] LAMP server
[ ] Mail server
[ ] OpenSSH server
[ ] PostgreSQL database
[ ] Print server
[ ] Samba File server
<Continue>


Select and install software
39%
Preparing iputils-arping

4 La gestione dei pacchetti di installazione

4.1 I differenti sistemi di gestione

Un sistema di gestione dei pacchetti è un insieme di programmi che automatizzano il processo di installazione, aggiornamento, configurazione e rimozione dei pacchetti software di un computer. Il termine è più comunemente utilizzato in relazione a sistemi Unix-like (tipo Unix), quale è Linux. Il loro impiego è particolarmente utile anche in relazione all'elevato numero di pacchetti che possono essere presenti in ogni distribuzione.
In tali sistemi, il software è distribuito tramite dei pacchetti, generalmente incapsulati in un singolo file. I pacchetti spesso includono anche altre importanti informazioni, come il nome completo del programma, la versione, il fornitore, il checksum[16], ed una lista di altri pacchetti, conosciuti come dipendenze, che sono necessarie al software per funzionare correttamente.

I sistemi di gestione dei pacchetti sono incaricati del compito di organizzare tutti i pacchetti installati in un sistema e mantenere la loro usabilità. Questi sistemi raggiungono questo scopo usando varie combinazioni delle seguenti tecniche:

- Verifica del checksum dei file per evitare differenze tra le versioni locali ed ufficiali di un pacchetto;
- Semplici strumenti per l'installazione, l'aggiornamento, e la rimozione;
- Gestione delle dipendenze per la distribuzione del software funzionante da un pacchetto;
- Controllo degli aggiornamenti per fornire le ultime versioni dei software, che spesso includono riparazioni di difetti ed aggiornamenti di sicurezza;
- Raggruppamento di pacchetti a seconda della funzione per aiutare l'utente ad eliminare la confusione durante l'installazione ed il mantenimento.

Come abbiamo già anticipato con le distribuzioni, vi sono vari sistemi di gestione dei pacchetti. I principali sono due, il "*RPM Package Manager*" e il "*dpkg*", impiegati, ad esempio, il primo nella distribuzione Red Hat e il secondo nella Debian. Spesso però si ricorre a sistemi più evoluti, che pur appoggiandosi ai precedenti, risolvano automaticamente problemi più complessi, quale quello delle dipendenze. Tali sistemi sono ad esempio "*yum*" e "*apt*".
Infatti se è pur vero che all'interno di ciascun pacchetto è presente l'elenco dei pacchetti in dipendenza, è anche vero che ne "*rpm*", e ne "*dpkg*", scarichino tali dipendenze in automatico. Se si volesse ad esempio installare il pacchetto del server web "apache" con il supporto "*ssl*", sia "*yum*" che "*apt*" scaricherebbero e installerebbero le librerie "*ssl*" prima di eseguire l'installazione del pacchetto "*apache*". Per poter eseguire questo lavoro, i gestori più evoluti contengono un elenco dei repository sui quali è possibile trovare i pacchetti di installazione dei vari software.

L'obbiettivo dei sistemi di gestione dei pacchetti è di semplificare il lavoro dell'utente nell'aggiornamento del sistema, nell'installazione di nuovi pacchetti o nella rimozione di pacchetti già installati. La semplificazione è attuata rispetto al classico metodo di installazione tramite la

[16]Tradotto letteralmente significa *somma di controllo*. È una sequenza di bit che viene utilizzata per verificare l'integrità di un dato o di un messaggio che può subire alterazioni.

compilazione[17] dei sorgenti di un pacchetto, che vengono dunque resi disponibili nei repository in base all'architetture del sistema, al tipo di distribuzione, e a volte in base alla versione delle distribuzione.
Per semplificare ulteriormente il lavoro degli utenti non esperti, tutte le distribuzioni desktop permettono di eseguire la gestione dei pacchetti tramite interfacce grafiche.

4.2 Il sistema di gestione Apt e Aptitude

L' Advanced Packaging Tool, conosciuto con l'acronimo APT, è il gestore standard di pacchetti software della distribuzione Debian. Apt ha la particolarità di sfruttare contemporaneamente diverse sorgenti remote o locali dei pacchetti (FTP, HTTP, cdrom, e hard disk), di gestire autonomamente diverse distribuzioni di pacchetti e di permettere velocemente l'aggiornamento del sistema operativo ad una particolare distribuzione. Sotto Debian si hanno tre differenti versioni delle stessa distribuzione che è possibile impiegate: *stable*, *testing* e *unstable*, che come vedremo e possibile scegliere dalla configurazione di apt.
Essendo stato studiato per poter essere interfacciato graficamente, sono nati diversi strumenti grafici che permettono di gestire il contenuto del sistema operativo attraverso una GUI che può risultare più intuitiva all'utente inesperto. Tra tutti è possibile segnalare Aptitude (in gui testuale) e Synaptic (in gui grafica).
Apt si basa sostanzialmente sul file "*/etc/apt/sources.list*" (che contiene la lista delle sorgenti da cui attingere i pacchetti) e sul comando "*apt-get*".

Esempio di file "*sources.list*":

```
#
# deb cdrom:[Ubuntu 9.10 _Karmic Koala_ - Release i386 (20091028.2)]/ karmic main restricted

deb http://it.archive.ubuntu.com/ubuntu/ karmic main restricted
deb-src http://it.archive.ubuntu.com/ubuntu/ karmic main restricted

## Major bug fix updates produced after the final release of the
## distribution.
deb http://it.archive.ubuntu.com/ubuntu/ karmic-updates main restricted
deb-src http://it.archive.ubuntu.com/ubuntu/ karmic-updates main restricted

## N.B. software from this repository is ENTIRELY UNSUPPORTED by the Ubuntu
## team. Also, please note that software in universe WILL NOT receive any
## review or updates from the Ubuntu security team.
deb http://it.archive.ubuntu.com/ubuntu/ karmic universe
deb-src http://it.archive.ubuntu.com/ubuntu/ karmic universe
deb http://it.archive.ubuntu.com/ubuntu/ karmic-updates universe
deb-src http://it.archive.ubuntu.com/ubuntu/ karmic-updates universe

## N.B. software from this repository is ENTIRELY UNSUPPORTED by the Ubuntu
## team, and may not be under a free licence. Please satisfy yourself as to
## your rights to use the software. Also, please note that software in
## multiverse WILL NOT receive any review or updates from the Ubuntu
## security team.
deb http://it.archive.ubuntu.com/ubuntu/ karmic multiverse
deb-src http://it.archive.ubuntu.com/ubuntu/ karmic multiverse
deb http://it.archive.ubuntu.com/ubuntu/ karmic-updates multiverse
deb-src http://it.archive.ubuntu.com/ubuntu/ karmic-updates multiverse

## Uncomment the following two lines to add software from the 'backports'
## repository.
## N.B. software from this repository may not have been tested as
```

[17] Traduzione in linguaggio macchina del pacchetto a partire dai sorgenti del software scritti dai programmatori.

```
## extensively as that contained in the main release, although it includes
## newer versions of some applications which may provide useful features.
## Also, please note that software in backports WILL NOT receive any review
## or updates from the Ubuntu security team.
# deb http://it.archive.ubuntu.com/ubuntu/ karmic-backports main restricted universe
multiverse
# deb-src http://it.archive.ubuntu.com/ubuntu/ karmic-backports main restricted universe
multiverse

## Uncomment the following two lines to add software from Canonical's
## 'partner' repository.
## This software is not part of Ubuntu, but is offered by Canonical and the
## respective vendors as a service to Ubuntu users.
deb http://archive.canonical.com/ubuntu karmic partner
# deb-src http://archive.canonical.com/ubuntu karmic partner

deb http://security.ubuntu.com/ubuntu karmic-security main restricted
deb-src http://security.ubuntu.com/ubuntu karmic-security main restricted
deb http://security.ubuntu.com/ubuntu karmic-security universe
deb-src http://security.ubuntu.com/ubuntu karmic-security universe
deb http://security.ubuntu.com/ubuntu karmic-security multiverse
deb http://it.archive.ubuntu.com/ubuntu/ karmic-proposed restricted main multiverse universe
deb-src http://security.ubuntu.com/ubuntu karmic-security multiverse
deb http://ppa.launchpad.net/llyzs/ppa/ubuntu karmic main
```

Le righe hanno la seguente sintassi:

```
deb http://host/debian distribuzione sezione1 sezione2 sezione3
deb-src http://host/debian distribuzione sezione1 sezione2 sezione3
```

La prima chiave di ogni riga, "*deb*" o "*deb-src*", indica il tipo di archivio, cioè se contiene pacchetti binari che sono già compilati o se l'archivio contiene i pacchetti sorgente che sono il codice sorgente originale del programma.
La seconda parola indica l'indirizzo della sorgente.
Al posto di *distribuzione* deve essere indicata la distribuzione che si vuole gestire (di solito uno dei tre rami di sviluppo *stable*, *testing* o *unstable*, oppure esplicitamente la versione, per esempio *woody e sarge* per Debian, o karmic e janty per Ubuntu). Le *sezioni* indicheranno quali parti della distribuzione dovranno essere gestite (normalmente si possono trovare *main* (i pacchetti completamente liberi, la maggioranza), *non-free* (i pacchetti rilasciati sotto una licenza non libera) e *contrib* (pacchetti liberi che però dipendono da altri non liberi), e così via.

Comandi di "*apt*":

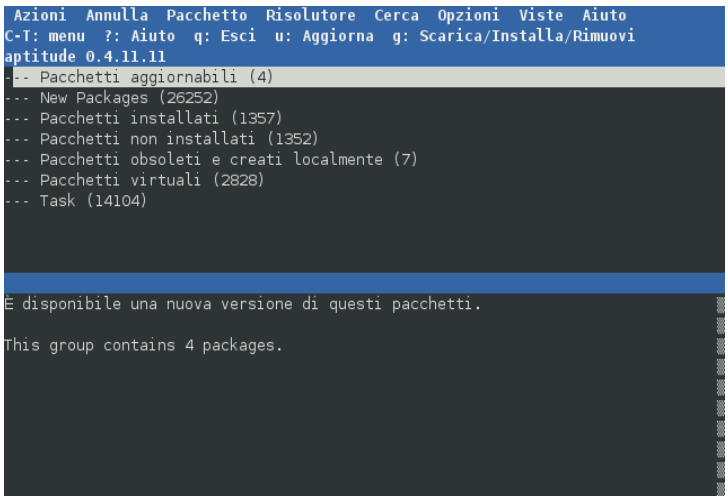
`apt-get update`	aggiorna sul sistema l'elenco dei pacchetti scaricabili, segnalando eventuali pacchetti che è possibile aggiornare
`apt-get install` *nomePacchetto*	installa il pacchetto indicato, e nel caso sia necessario scarica e installa anche eventuali pacchetti in dipendenza
`apt-get remove` *nomePacchetto*	rimuove il pacchetto indicato, senza cancellarne i file di configurazione
`apt-get --purge remove` *nomePacchetto*	rimuove il pacchetto indicato, cancellando anche i relativi file di configurazione
`apt-get autoremove` *nomePacchetto*	rimuove il pacchetto indicato e tutti i pacchetti dipendenti da questo ultimo
`apt-get upgrade`	aggiorna tutti i pacchetti installati sul sistema
`apt-get -f install`	verifica lo stato dei pacchetti installati e consiglia la rimozione di pacchetti non più necessari[18].
`apt-get --simulate` ***azione***	esegue una simulazione del comando dato

[18]Un pacchetto è non più necessario se è stato precedentemente installato non su scelta dell'utente, ma come dipendenza di un altro pacchetto, e se il pacchetto che ne ha richiesto l'installazione non è più presente. Il sistema apt verifica che il pacchetto non sia più necessario a nessun altro pacchetto del sistema, e non solo al primo che ne ha richiesto l'installazione

Ad esempio, prima di compiere un upgrade importante, si può provare il seguente comando e controllare che non vengano installati pacchetti che non ci interessano e che non vengano disinstallati pacchetti importanti:

```
apt-get --simulate upgrade
```

Il comando "*aptitude*" esegue una gui testuale in cui è possibile visualizzare l'intero elenco dei pacchetti installati e installabili, suddivisi secondo varie categorie. Tramite tale comando è possibile visualizzare in maniera semplice anche le informazioni dei pacchetti, con relative dipendenze e dipendenti.



I comandi più comuni di "*aptitude*" sono:

u	esegue un aggiornamento della lista dei pacchetti
U	segna come da installare tutti i pacchetti aggiornabili
g	visualizza una anteprima dell'elenco dei pacchetti segnati come da installare, se seguita da un'altra "g" esegue l'aggiornamento del sistema
+	segna come da installare il pacchetto correntemente evidenziato
-	segna come da rimuovere il pacchetto correntemente evidenziato
q	esce dalla schermata corrente o dal programma

Lo stato dei pacchetti indicato da "*aptitude*" è:

i	installato
b	difettoso
c	non installato ma configurazione ancora presente
C	semi configurato

Per un elenco esaustivo dei comandi e dello stato dei pacchetti di "*aptitude*" si veda il manuale in linea selezionabile con "?", o il man del comando.

4.3 Il pacchetto Alien

Citando il "*man*" di "*alien*":

> "*alien is a program that converts between Red Hat rpm, Debian deb, Stampede slp, Slackware tgz, and Solaris pkg file formats. If you want to use a package from another linux distribution than the one you have installed on your system, you can use alien to convert it to your preferred package format and install it. It also supports LSB packages.*"

cioè un pacchetto che permette la codifica di un di un formato di installazione in un altro formato, per una successiva installazione su differenti distribuzioni.
Se ad esempio un pacchetto fosse rilasciato solo in formato "*rpm*", sarebbe possibile installarlo su un sistema Debian-like tramite il comando "*alien*":

```
alien -i -d -c nomePacchetto
```

4.4 Il sistema di gestione Yum

Yellow dog Updater, Modified (YUM) è un sistema di gestione dei pacchetti open source a linea di comando per i sistemi operativi Linux compatibili col formato RPM. È stato sviluppto da Seth Vidal e da un gruppo di programmatori volontari, ed attualmente è mantenuto dal progetto Linux@DUKE della Duke University. Sebbene yum sia un'utility a linea di comando, ci sono diversi tool che forniscono un'interfaccia grafica, come ad esempio: pup, pirut, e yumex.
Yum è la riscrittura del suo predecessore, Yellowdog Updater (YUP), e venne sviluppato inizialmente in modo da aggiornare e gestire i sistemi Red Hat Linux usati dal dipartimento di fisica della Duke University. Da allora, yum venne adottato dalla Fedora Core, CentOS, e da altre distribuzioni Linux basate su RPM, tra cui la stessa Yellow Dog Linux, dove sostituì l'originale utility YUP.
Chi possiede il gestore dei pacchetti di Red Hat Enterprise Linux, up2date, può utilizzare anche i repository di yum per aggiornare i programmi.

I repositories sono dei servers dove yum (o più in generale, il package manager), trova i software. In distribuzioni con yum i file sono identificati dall'estensione .repo e sono situati in "*/etc/yum.repos.d*"
I comandi più comuni di yum sono:

```
yum install nomepacchetto
yum remove nomepacchetto
yum update
```

La creazione dei repository di yum è gestita da un altro tool chiamato "*createrepo*", il quale genera i metadata XML necessari.

5 La shell di sistema

5.1 Premessa

La shell di sistema è un interprete dei comandi che viene impiegata come interfaccia utente per l'amministrazione del sistema stesso. Esistono vari tipi di shell, "*sh*", "*bash*", "*ksh*", "*rsh*", etc, ciascuna con le proprie caratteristiche peculiari. Ad esempio rsh, acronimo di "*remote shell*", è una shell con diritti sul sistema ridotti in virtù della natura del collegamento, remoto, contrapposto a quello locale. Ciascuna shell ha i propri file di configurazione, configurabili anche a livello di singolo utente.
Alla connessione di un utente sul sistema viene verificato sul file

```
/etc/passwd
```

quale shell eseguire. Ad esempio la riga del precedente file:

```
gvmura:x:1000:1000:,,,:/home/gvmura:/bin/bash
```

identifica la bash come shell dell'utente "gvmura". Pertanto, oltre ai classici file:

```
/etc/profile      e      /home/gvmura/.profile
```

vengono caricate anche le configurazioni del file:

```
/home/gvmura/.bashrc
```

Esistono degli utenti, come alcuni utenti di applicazione, che non necessitano di una shell e che dunque presentano nel file "/etc/passwd" il valore:

/bin/false o */bin/nologin*

La shell di sistema contiene delle variabili preconfigurate, impostate alla connessione, necessarie per corretto funzionamento di alcuni programmi del sistema. Ad esempio la variabile "USERNAME" viene caricata col valore della login dell'utente, mentre la "PATH" con i percorsi di ricerca dei file eseguibili. Per visualizzare l'elenco e i valori delle variabili dell'ambiente è possibile eseguire il comando:

```
env
```

Tramite l'istruzione:

```
nomeVariabile=valore
```

è possibile definire nuove variabili nell'ambiente, mentre per fare in modo che tali variabili siano impiegabili anche da altri programmi eseguiti dalla shell è necessario precedere la precedente istruzione dalla keyword:

```
export
```

Dalla shell è possibile eseguire sia i programmi precaricati nel sistema che degli scripts costruiti ad hoc e contenenti un elenco di istruzioni organizzate secondo una sequenza logica. La prima riga di uno script viene interpretata per controllare con quale tipo di shell eseguire lo script stesso. L'istruzione che ne abilita questa caratteristica è:

`#!` seguita dalla shell da impiegare

5.2 Scripts di esempio

Il seguente script, "*monitorSpaceDisk*" esegue una verifica sui filesystem del sistema controllando che non superino una determinata occupazione del disco. In caso affermativo invia una mail di allerta agli amministratori in modo da informarli del problema e registra lo stato di allerta su un file di configurazione. Nel caso l'allarme cessasse automaticamente, per via di altri processi di pulizia, invia una successiva mail di cessazione del problema:

```
#!/bin/bash

#
# Script per il monitor dello spazio disco occupato da ciascun filesystem
# e per l'invio di una mail nel caso superasse un certo valore
#
# Nel caso di superamento viene creato il file:
#    monitorSpaceDisk.lock
#
# Se l'allarme rientra il file viene cancellato automaticamente
#
#
SOGLIA=90
MAIL_FILE="/root/tmp/monitorSpaceDisk.mail"
FILE_LOCK="/root/tmp/monitorSpaceDisk.lock"
if [ ! -d /root/tmp ]; then
   mkdir /root/tmp
fi
TMP_FILE="/root/tmp/monitorSpaceDisk.tmp"
MAIL_SUBJECT_BRUTTA="WARNING - Superamento soglia del ${SOGLIA}% su filesystem"
MAIL_SUBJECT_BUONA="ALLARME RIENTRATO - Superamento soglia su filesystem"
MAIL_BRUTTA="E' necessaria una verifica del sistema, nel caso l'allarme rientri non è
necessario nessun intervento"
MAIL_BUONA="Allarme rientrato"
MAIL_TO="gvmura@uniss.it ... ... ..."

ERRORE=0
date > ${MAIL_FILE}
df -kP | grep -vE '^Filesystem|tmpfs|cdrom' | awk '{ print $5 "  " $6 }' > ${TMP_FILE}
while read line;
do
   uso=$(echo $line | awk '{ print $1}' | cut -d'%' -f1)
   part=$(echo $line | awk '{ print $2 }')
   if [ ${uso} -ge ${SOGLIA} ]; then
      ERRORE=1
      echo "${uso}% ${part}" >> ${MAIL_FILE}
   fi
done < ${TMP_FILE}

if [ $ERRORE -eq 0 ] && [ -f ${FILE_LOCK} ]; then
```

```
   rm ${FILE_LOCK}
   echo ${MAIL_BUONA} >> ${MAIL_FILE}
   mail -s "$MAIL_SUBJECT_BUONA" "$MAIL_TO" < $MAIL_FILE
else
   if [ $ERRORE -eq 1 ] && [ ! -f ${FILE_LOCK} ]; then
      touch ${FILE_LOCK}
      echo ${MAIL_BRUTTA} >> ${MAIL_FILE}
      mail -s "$MAIL_SUBJECT_BRUTTA" "$MAIL_TO" < $MAIL_FILE
   fi
fi
```

Il seguente script "*creaSpazioWeb*" è impiegato per la creazione di uno spazio web. Tramite questo script l'operatore può facilmente costituire uno spazio web con annessa configurazione del server Apache, creazione di un database Mysql e assegnazione di una quota massima di filesystem per lo spazio:

```
#!/bin/bash

#
# Procedura per la creazione automatica di uno spazio web
# con annesso database mysql se necessario
#

#
# Variabili
#
DIR_APACHE_A="/etc/apache2/sites-available"
DIR_APACHE_E="/etc/apache2/sites-enabled"
DIR_INDEX="/var/www"
FILE_INDEX="index.html"
IP_PORT="......:80"
MAX_QUOTA=1500
COMMENTO="Amministratore sito web"
UTENTI_WEB=......
SHELL="/bin/false"
DIR_HOME="/home"
DIR_APACHE_LOG="/var/log/apache2"
CNF_FILE="/root/.my.cnf"
MYSQL_ROOT_PASS="......"

#
# Funzioni
#
crea_utente_sistema() {
   echo "Creazione utente $AMMIN..."
   useradd -s $SHELL -c "$COMMENTO" -d $DIR_HOME/$AMMIN -m -k /dev/null -G "$UTENTI_WEB"
$AMMIN
   passwd $AMMIN
   mkdir $DIR_HOME/$AMMIN/web
   chown $AMMIN.$AMMIN $DIR_HOME/$AMMIN/web
   setquota -g $AMMIN $SOFT_QUOTA $HARD_QUOTA 0 0 -a >/dev/null
}

crea_cnf() {
   touch $CNF_FILE
   chmod 600 $CNF_FILE
   echo "[client]" >$CNF_FILE
   echo "password=\"$MYSQL_ROOT_PASS\"" >>$CNF_FILE
}

crea_virtual_host() {
   echo "Creazione virtual host..."
   cat /dev/null >${DIR_APACHE_A}/${NOME_WEB}
   echo "<VirtualHost ${IP_PORT}>" >>${DIR_APACHE_A}/${NOME_WEB}
   echo "   ServerAdmin $EMAIL" >>${DIR_APACHE_A}/${NOME_WEB}
```

```
   echo "   DocumentRoot ${DIR_HOME}/${AMMIN}/web" >>${DIR_APACHE_A}/${NOME_WEB}
   echo "   ServerName ${NOME_WEB}" >>${DIR_APACHE_A}/${NOME_WEB}
   echo "   ServerAlias www.${NOME_WEB}" >>${DIR_APACHE_A}/${NOME_WEB}
   echo "   DirectoryIndex index.html index.htm index.php" >>${DIR_APACHE_A}/${NOME_WEB}
   echo "   <Directory ${DIR_HOME}/${AMMIN}/web>" >>${DIR_APACHE_A}/${NOME_WEB}
   echo "      Options -Indexes FollowSymLinks MultiViews" >>${DIR_APACHE_A}/${NOME_WEB}
   echo "      AllowOverride All" >>${DIR_APACHE_A}/${NOME_WEB}
   echo "      Order allow,deny" >>${DIR_APACHE_A}/${NOME_WEB}
   echo "      allow from all" >>${DIR_APACHE_A}/${NOME_WEB}
   echo "   </Directory>" >>${DIR_APACHE_A}/${NOME_WEB}
   echo "   ErrorLog ${DIR_APACHE_LOG}/${NOME_WEB}_error.log" >>${DIR_APACHE_A}/$
{NOME_WEB}
   echo "   LogLevel error" >>${DIR_APACHE_A}/${NOME_WEB}
   echo "   CustomLog ${DIR_APACHE_LOG}/${NOME_WEB}_access.log combined" >>$
{DIR_APACHE_A}/${NOME_WEB}
   echo "   ServerSignature On" >>${DIR_APACHE_A}/${NOME_WEB}
   echo "</VirtualHost>" >>${DIR_APACHE_A}/${NOME_WEB}

   ln -s ${DIR_APACHE_A}/${NOME_WEB} ${DIR_APACHE_E}/${NOME_WEB}

   cp ${DIR_INDEX}/${FILE_INDEX} ${DIR_HOME}/${AMMIN}/web/${FILE_INDEX}
   chown $AMMIN.$AMMIN ${DIR_HOME}/${AMMIN}/web/${FILE_INDEX}
}

#
# Inizio
#
clear
USCITA=0
while [ $USCITA -eq 0 ]
do
   echo
   echo "Inserisci il nome dello spazio web (senza www, es: ......)"
   echo "o premi <INVIO> per uscire..."
   read NOME_WEB
   if [ "$NOME_WEB" == "" ]; then
      exit 0
   fi
   wget -q -O /dev/null $NOME_WEB
   if [ $? -eq 0 ]; then
      echo "-- Errore -- il nome del sito esiste già"
      echo "             vuoi proseguire comunque? (s/n)..."
      read SCELTA
      if [ "$SCELTA" == "s" ] || [ "$SCELTA" == "S" ]; then
         USCITA=1
      else
         exit 1
      fi
   else
      USCITA=1
   fi
done

USCITA=0
while [ $USCITA -eq 0 ]
do
   echo "Inserisci l'account per l'amministratore (es: ......)..."
   read AMMIN
   finger $AMMIN 2>/dev/null | grep "Login: $AMMIN" >/dev/null
   if [ $? -eq 0 ]; then
      echo "-- Errore -- l'utente esiste gia'"
   else
      USCITA=1
   fi
done

USCITA=0
while [ $USCITA -eq 0 ]
do
```

```
   echo "Inserisci l'indirizzo e-mail dell'amministratore..."
   read EMAIL
   if [ "$EMAIL" == "" ]; then
      echo "-- Errore -- non puo' essere vuoto!"
   else
      USCITA=1
   fi
done

USCITA=0
while [ $USCITA -eq 0 ]
do
   echo "Inserisci i MB di quota disco richesta (max $MAX_QUOTA)..."
   read SPAZIO_DISCO
   if [ "$SPAZIO_DISCO" == "" ] || [ $SPAZIO_DISCO -gt $MAX_QUOTA ]; then
      echo "-- Errore -- non puo' essere vuota ne maggiore di $MAX_QUOTA"
   else
      ((SOFT_QUOTA=SPAZIO_DISCO*1000))
      ((HARD_QUOTA=SPAZIO_DISCO*1100))
      USCITA=1
   fi
done

USCITA=0
while [ $USCITA -eq 0 ]
do
   echo "Se e' necessario un database mysql inserirne il nome,"
   echo "altrimenti premere <INVIO>..."
   read DATABASE
   if [ -s $CNF_FILE ]; then
      echo "Il file .my.cnf esiste gia', non sara' effettuato il test"
      echo "sulla esistenza di un database con lo stesso nome!"
      USCITA=1
      break
   else
      crea_cnf
      mysql $DATABASE </dev/null &>/dev/null
      if [ $? -eq 0 ] && [ "$DATABASE" != "" ]; then
         echo "-- Errore -- esiste gia' un database con quel nome"
      else
         USCITA=1
      fi
      rm $CNF_FILE
   fi
done

echo
echo "Riepilogo informazioni"
echo "            nome del sito: $NOME_WEB"
echo "   account amministratore: $AMMIN"
echo "                    email: $EMAIL"
echo "              quota disco: $SPAZIO_DISCO"
echo "           database mysql: $DATABASE"
echo
echo "Si vuole procedere con la creazione? (s/n)..."
read SCELTA
if [ "$SCELTA" == "s" ] || [ "$SCELTA" == "S" ]; then
   echo
   crea_utente_sistema

   if [ "$DATABASE" != "" ]; then
      USCITA=0
      while [ $USCITA -eq 0 ]
      do
         echo "Inserisci la password per l'accesso al database..."
         read MYSQL_AMMIN_PASS
         if [ "$MYSQL_AMMIN_PASS" == "" ] || [ "$MYSQL_AMMIN_PASS" == "$AMMIN" ]; then
            echo "-- Errore -- la password non puo' essere vuota o uguale all'utente"
         else
```

```
            USCITA=1
         fi
      done

      if [ -s $CNF_FILE ]; then
         echo "Il file .my.cnf esiste già e non verra' modificato."
         echo "Per creare il database e assegnare i privilegi eseguire:"
         echo "   create database $DATABASE character set utf8 collate utf8_general_ci;"
         echo "   grant all privileges on $DATABASE.* to '$AMMIN'@'localhost';"
         echo "   set password for '$AMMIN'@'localhost' = password('$MYSQL_AMMIN_PASS');"
         echo "   flush privileges;"
      else
         echo "Creazione database e assegnazione privilegi..."
         crea_cnf
         echo "create database $DATABASE character set utf8 collate utf8_general_ci;" |
mysql
         echo "grant all privileges on $DATABASE.* to '$AMMIN'@'localhost';" | mysql
         echo "set password for '$AMMIN'@'localhost' = password('$MYSQL_AMMIN_PASS');" |
mysql
         echo "flush privileges;" | mysql
         rm $CNF_FILE
      fi
   fi

   crea_virtual_host
   /etc/init.d/apache2 reload
fi
```

6 La gestione delle sicurezza

6.1 Premessa

La sicurezza di un sistema Linux, come di un qualsiasi sistema informatico, non è la semplice applicazione di una serie di regole, ma una metodologia di progettazione che abbia come obiettivo la messa in sicurezza del sistema nel suo complesso. Troppo spesso infatti vengono messe in sicurezza solo alcune porzioni del sistema dimenticandosi, volutamente o involutamente, di altre che rendono nel suo complesso il sistema insicuro.
Ad esempio, se in un sistema le password degli utenti sono salvate secondo algoritmi altamente sicuri, ma poi è permesso agli utenti di impostare una password identica al loro nome utente, o facilmente individuabile in base a informazioni dell'utente stesso, il sistema sarà altamente insicuro.
In questo capitolo tratteremo solo una parte dei problemi relativi alla sicurezza, e relativi al solo sistema operativo. Non va dimenticato però che ogni sistema non è fine a se stesso, e che spesso le sue reali funzioni vanno al di la del sistema operativo.
Si pensi infatti alle credenziali di accesso, e relativi numeri di carta di credito, degli utenti di un negozio di e-commerce. Sicuramente tali dati non risiederanno sul sistema operativo ma su un database gestito dall'applicazione web di vendita on-line. La loro compromissione è distruttiva quanto o più di quella del sistema operativo.

6.2 Gli utenti del sistema

Le informazioni relative agli utenti di un sistema Linux sono contenute nel file "*/etc/passwd*", che per questo risulta leggibile da qualsiasi processo che ne abbia bisogno, e dunque da chiunque.
In esso è contenuto l'indicazione del nome di "*login*" dell'utente, il gruppo di appartenenza, il percorso della directory personale, la shell da impiegare al login, etc.
Questo file può contenere anche le password degli utenti, anche se ormai tutte le distribuzioni privilegiano un sistema chiamato delle "*shadow*" password, che salva le password degli utenti in un altro file, il "*/etc/shadow*", leggibile solo dal sistema o dall'amministratore, a cui gli utenti non accedono liberamente ma per via di chiamate di sistema. In questo modo nessun utente può visualizzare la password di nessun altro, anche se quest'ultima risulta criptata.
Lo stesso viene applicato alle password dei gruppi, che sono contenute nel file "*/etc/gshadow*".

Il file "*/etc/passwd*" contiene sia gli utenti reali del sistema che gli utenti di applicazione impiegati per l'esecuzione di particolari processi. Ad esempio, il demone del server web Apache viene eseguito come utente "*www-data*" nei sistemi Debian e come utente "*http*" nei sistemi RedHat.
Esempio di "*/etc/passwd*":

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
…
```

```
gdm:x:112:119:Gnome Display Manager:/var/lib/gdm:/bin/false
gvmura:x:1000:1000:,,,:/home/gvmura:/bin/bash
```

Esempio di "/etc/shadow":

```
root:*:14555:0:99999:7:::
daemon:*:14555:0:99999:7:::
bin:*:14555:0:99999:7:::
sys:*:14555:0:99999:7:::
gdm:*:14555:0:99999:7:::
gvmura:
$6$SC/qDM.Z$x9FQtwfkVJfPAT4tMAssvB12PEdiKJyCLCd6R2sbhDZf8ohKeHWW.cZzsw0L6oQ2XP8F
hiSDFrDPXRMz/4fJh1:14555:0:99999:7:::
```

da cui si vede che il solo utente "*gvmura*" contiene una password.
Agli utenti privi di password viene negato l'accesso al sistema, ma non l'esecuzione dei processi. Come è possibile dunque eseguire un processo senza accedervi? Un altro utente, il super user root, manda in esecuzione i processi come se fosse al loro posto. Questo metodo è molto usato in tutti i sistemi Linux e Unix in generale. Operativamente viene impiegato il comando "*su*", es:

```
su - www-data -c "apachectl status"
```

che esegue come utente "www-data" il comando "*apachectl*" col parametro di input "status".

In un sistema Linux i compiti dell'amministratore sono da sempre stati demandati all'utente di sistema "*root*", che spesso non identificava un unico utente reale ma più utenti, che lo impiegavano per eseguire i vari compiti di loro competenza. Per ovviare a tale sovrapposizione di impiego e al contempo per rispettare la normativa vigente in materia di protezione dei dati personali, gli utenti amministratori possono impiegare il comando:

```
sudo
```

per eseguire i compiti amministrativi ad essi assegnati. In tal modo l'account "*root*" è privo di password e dunque inibito all'accesso.
Il file di configurazione di "sudo" è:

```
/etc/sudoers
```

nel quale è possibile indicare il nome account degli amministratori, o gli scripts che i non amministratori siano in grado di eseguire. E' dunque possibile pilotare compiti amministrativi tramite scripts preorganizzati dall'amministratore e far eseguire tali scripts da utenti non amministratori, come se lo siano.
Una riga di esempio di tale file è la seguente:

```
%admin ALL=(ALL) ALL
```

nella quale si specifica che tutti gli utenti del gruppo "*admin*" sono abilitati ad eseguire ogni compito amministrativo.
Si abbia particolare cura nella modifica del file "*/etc/sudoers*" in quanto una sua modifica, se errata, inibisce l'esecuzione di altri eventuali compiti amministrativi, come anche la correzione dell'errore su tale file. Si consiglia dunque di mantenere due shell attive, in cui una sia quella di root e l'altra

quella di test del funzionamento di "*sudo*", in modo che su errori da quella di test la shell di root ne permetta il rimedio. Ma come è possibile aprire una shell di root? Col comando:

```
sudo -i
```

Altri comandi utili:

`adduser`	aggiunta di un utente al sistema
`usermod`	modifica di un utente del sistema
`passwd`	modifica della password di un utente
`deluser`	cancellazione di un utente
`finger`	visualizzazione delle informazioni di un utente

6.3 I diritti sul filesystem

Ogni oggetto di un filesystem, sia esso file, directory o device ha dei diritti che ne abilitano o meno l'operatività da parte degli utenti. Questi diritti possono essere di:

- lettura
- scrittura
- esecuzione

dove ciascuno di essi può essere applicato a:

- l'utente proprietario
- il gruppo proprietario
- chiunque altro

E' possibile visualizzare tali diritti tramite la prima colonna del output del comando "*ls -la*", es:

```
drwxr-xr-x 14 root             root    4096 2010-03-21 18:52 .
drwxr-xr-x 16 root             root    4096 2010-01-01 18:19 ..
drwxr-xr-x  2 root             root    4096 2009-10-17 06:40 apparmor
drwxr-xr-x  2 root             root    4096 2010-03-04 20:41 apt
-rw-r--r--  1 root             root       0 2010-03-15 09:12 aptitude
-rw-r-----  1 syslog           adm     1398 2010-03-21 19:17 auth.log
-rw-r-----  1 root             adm       31 2009-11-07 12:47 boot
drwxr-xr-x  2 root             root    4096 2010-03-21 17:10 cups
-rw-r-----  1 syslog           adm    29966 2010-03-21 19:29 daemon.log
-rw-r-----  1 root             adm    40071 2010-03-21 18:52 dmesg
drwxrwx--T  2 root             gdm     4096 2010-03-21 18:52 gdm
drwxr-xr-x  3 root             root    4096 2009-11-07 13:13 installer
-rw-r--r--  1 root             root       0 2009-11-11 21:27 jockey.log
-rw-r--r--  1 root             root   61553 2009-11-11 21:27 jockey.log.1
-rw-r-----  1 syslog           adm    59947 2010-03-21 18:53 kern.log
-rw-rw-r--  1 root             utmp  292292 2010-01-01 17:53 lastlog
-rw-r-----  1 syslog           adm    48312 2010-03-21 18:53 messages
```

Il primo campo identifica se sia file "-" o direcotiry "*d*". Gli altri campi, a gruppi di 3 sono per l'utente, il gruppo e tutti gli altri "*other*", in cui per ciascun gruppo il primo valore può essere "-" o "*r*" che sta per lettura, il secondo "-" o "*w*" per scrittura e il terzo "-" o "*x*" per esecuzione. Quando il campo è "-" vuol dire che il relativo diritto non è dato.
Ad esempio, il file "*auth.log*" di sopra è leggibile e scrivibile dall'utente "*syslog*", leggibile dal gruppo "*adm*" e nessun altro diritto è permesso per altri utenti.

Nota sulle directory:
Una directory non è altro che un file in cui sono contenute le informazioni di altri file, e che dunque la contengono. Il diritto di lettura su una directory permette la lettura del solo file "directory" (la lista dei file), ma non la visualizzazione dell'elenco dei file in essa contenuti. Infatti per avere indicazioni sui file di una directory è necessario anche il diritto di esecuzione sulla directory stessa. Esempio:

ls -la /tmp/prova con diritti su "*other*" di "*r-x*"

```
drwxr-xr-x  2 syslog syslog 4096 2010-03-21 19:52 .
drwxrwxrwt 15 root   root   4096 2010-03-21 19:52 ..
-rw-r--r--  1 root   root      0 2010-03-21 19:52 fileTest
```

ls -la /tmp/prova con diritti su "*other*" di "*--x*"

```
ls: impossibile aprire la directory /tmp/prova/: Permesso negato
```

ls -la /tmp/prova con diritti su "*other*" di "r--"

```
ls: impossibile accedere a /tmp/prova/fileTest: Permesso negato
ls: impossibile accedere a /tmp/prova/..: Permesso negato
ls: impossibile accedere a /tmp/prova/.: Permesso negato
totale 0
d????????? ? ? ? ?                ? .
d????????? ? ? ? ?                ? ..
-????????? ? ? ? ?                ? fileTest
```

Per modificare i diritti su un file si impiega il comando:

```
chmod
```

nel quale è possibile specificare "u" per user, "g" per gruppo e "o" per altri, seguito da un "+" o un "-" (per dare o togliere) e dal diritto associato "r", "w" o "x". Es, per aggiungere i diritti di lettura e scrittura per utente e gruppo ad un file:

```
chmod ug+rw nomeFile
```

questo non modifica eventuali diritti già assegnati al file.
E' anche possibile impostare tutti i diritti su un file in una unica soluzione. A tale scopo i 3 possibili gruppi di diritti vengono suddivisi tra loro e interpretati come cifre binarie con potenza di 2, es:

```
-  -  -      -  -  -      -  -  -
2² 2¹ 2⁰     2² 2¹ 2⁰     2² 2¹ 2⁰
```

e mettendo assieme il numero risultante degli "accesi" si ottiene il valore da assegnare, es:

```
chmod 750 nomeFile          →    rwx r-x  ---
chmod 644 nomeFile          →    rw-  r--  r--
chmod 777 nomeFile          →    rwx rwx rwx
chmod 333 nomeFile          →    -wx -wx -wx
chmod 007 nomeFile          →    ---  ---  rwx        si può fare? SI
```

Come assegno l'utente e il gruppo padrone di un un file?
Per modificare l'appartenenza di un file ad un utente o ad un gruppo si usa il comando:

```
chown utente.gruppo nomeFile
```

con il nome dell'utente e del gruppo a cui assegnare il file.

Lo "*sticky bit*"
E' un particolare bit facente parte dei diritti di una directory che permette di risolvere un problema particolare. Si pensi ad una cartella condivisa da più utenti, in cui gli stessi possano scrivere e dunque creare e cancellare file. Secondo quanto abbiamo visto un utente potrebbe, anche senza i diritti di scrittura, cancellare i file di un altro utente in quanto ha i diritti di scrittura sulla cartella, e questo sarebbe da evitare. Per ovviare a tale problema si imposta sulla cartella lo "stick bit" o "the restricted deletion flag" che prima della cancellazione del file verifica oltre ai diritti sulla cartella anche i diritti sul file.
Questo bit è molto usato nelle cartelle condivise quali la "*/tmp*" e si individua tramite il campo di esecuzione di "other" della directory che non varrà più "-" o "*x*", ma "-" o "*t*" o "*T*", es:

```
-rwxr-xr-t          esecuzione per "other" e "sticky bit" impostato
-rwxr-xr-T          solo "sticky bit" impostato
```

Per impostarlo su una cartella si può eseguire uno dei due comandi:

```
chmod +t /tmp                    esempio per la /tmp
chmod 1777 /tmp                        "     "
```

Il "*setuid e setgid*"
L'acronimo "*setuid*" è indicato per "*set user id*" e analogamente il "*setgid*" è indicato per "*set group id*". Con tale impostazione, da assegnare ad un file, un utente viene abilitato ad eseguire un dato programma con le autorizzazioni dell'utente e/o del gruppo proprietario del file stesso. Questo è un altro utile meccanismo per permettere l'esecuzione di programmi o scripts altrimenti non permessi ad utenti non amministratori. Un esempio di impiego di tale modalità è data dal comando "*passwd*" che serve per cambiare la password di un utente o per cambiarsi la password:

```
ls -la /usr/bin/passwd
      -rwsr-xr-x 1 root root 41292 2009-07-31 15:55 /usr/bin/passwd
```

come si vede dall'esempio, viene segnalato con una "*s*" al posto del campo esecuzione, per indicare che il file deve essere eseguito con diritti di root. Infatti il comando "*passwd*" deve poter modificare il file "*/etc/shadow*", altrimenti inibito per gli utenti.
Tale impostazione può trovarsi sia sul campo dell'utente owner (*setuid*) che sul campo del gruppo owner (*setgid*).
Affinché funzioni è necessario che il filesystem sia montato in modalita "*suid*" (come avviene di default).

6.4 AppArmor

AppArmor, acronimo di "*Application Armor*" è uno strumento per la protezione del sistema sviluppato con l'obbiettivo di fornire un ambiente di sviluppo per la sicurezza, dalle prestazioni elevate e dal facile impiego. Questo protegge il sistema operativo e le applicazioni da minacce interne o esterne anche impedendo che siano sfruttati dei difetti, anche non ancora conosciuti, delle applicazioni stesse. Le impostazioni di sicurezza di AppArmor, chiamate "*profiles*", definiscono precisamente a quali risorse di sistema e con quali privilegi possano accedere le singole applicazioni. In questa ottica ad ogni applicazione è associato un profilo di esecuzione, che ne controlla tutti gli aspetti.
L'installazione su Debian di AppArmor presenta già dei profili predefiniti che blindano anche applicazioni quali "*mysql*". E' anche possibile utilizzare in AppArmor strumenti di analisi statiche avanzate e strumenti di apprendimento, in modo da confinare applicazioni per le quali non sono disponibili dei profili predefiniti. In altre parole, è possibile impostarlo in modalità apprendimento, eseguire l'applicazione, in tutti i suoi aspetti, e generare il profilo voluto. Una volta generato, possibili bachi o malfunzionamenti dell'applicazione vengono bloccati da AppArmor.
Il suo funzionamento è direttamente gestito dal kernel Linux.

Es. di profilo di default per mysql "*/etc/apparmor.d/usr.sbin.mysqld*":

```
# vim:syntax=apparmor
# Last Modified: Tue Jun 19 17:37:30 2007
#include <tunables/global>

/usr/sbin/mysqld {
  #include <abstractions/base>
  #include <abstractions/nameservice>
  #include <abstractions/user-tmp>
  #include <abstractions/mysql>

  capability dac_override,
  capability setgid,
  capability setuid,

  /etc/hosts.allow r,
  /etc/hosts.deny r,

  /etc/group              m,
  /etc/passwd             m,

  /etc/mysql/*.pem r,
  /etc/mysql/conf.d/ r,
  /etc/mysql/conf.d/* r,
```

```
  /etc/mysql/my.cnf r,
  /usr/sbin/mysqld mr,
  /usr/share/mysql/** r,
  /var/lib/mysql/ r,
  /var/lib/mysql/** rwk,
  /var/log/mysql/ r,
  /var/log/mysql/* rw,
  /var/run/mysqld/mysqld.pid w,
  /var/run/mysqld/mysqld.sock w,
}
```

Per verificare lo stato di AppArmor si esegua:

```
sudo apparmor_status
```

il cui output è ad esempio:

```
apparmor module is loaded.
1 profiles are loaded.
1 profiles are in enforce mode.
   /usr/sbin/mysqld
0 profiles are in complain mode.
1 processes have profiles defined.
1 processes are in enforce mode :
   /usr/sbin/mysqld (4267)
0 processes are in complain mode.
0 processes are unconfined but have a profile defined.
```

Si provi a generare un profilo per il programma "*ping*", con la modalità di apprendimento:

```
sudo aa-genprof /bin/ping
```

dopo la richiesta di accesso ad un repository in line dei profili, si selezioni L (later), si selezioni S per uno scan degli eventi di sistema, si esegua in una'altra shell il processo, si abiliti con A ai vari eventi che genererà il processo, e si visualizzi alla fine il profilo generato, che può essere inserito nei profili da analizzare "*/etc/apparmor.d/*" con il salvataggio S. Profilo generato "*bin.ping*"

```
# Last Modified: Wed Mar 24 18:47:41 2010
#include <tunables/global>

/bin/ping flags=(complain) {
  #include <abstractions/base>
  #include <abstractions/nameservice>

  capability net_raw,
  capability setuid,

  network inet raw,
 }
```

Per ricaricare tutti i profili assegnati si esegua:

```
sudo /etc/init.d/apparmor reload
```

Altri comandi utili:

`aa-logprof`	scansiona il file di registro "*syslog"* alla ricerca dei messaggi di auth di AppArmor e ne visualizza il contenuto

7 I servizi di base

7.1 I servizi di rete

Quando si parla di servizi di rete, o più precisamente del Linux networking subsystem, si intendono tutti quei servizi che permettono sia la comunicazione del server con il modo esterno, e sia la comunicazione stessa dei processi interni tra loro (interprocess networking) eseguita tramite "unix socket".
In questo corso focalizzeremo l'attenzione solo sui servizi di comunicazione esterni, dando di volta in volta degli accenni al sistema di interprocess.

Durante l'installazione del server viene richiesto di inserire la configurazione IP dello stesso, che, come nel nostro caso è stata rimandata. Questa avrebbe precompilato per noi dei file di configurazione che permettono al sistema il collegamento con la rete locale, se collegata. I passi necessari per eseguire una tale configurazione, nel caso di sistemi Debian-like, sono:

- compilazione del file /etc/hosts
- " " /etc/networks
- " " /etc/network/interfaces
- riavvio del servizio di rete

Vediamo nel dettaglio i vari passi.
Il file di "hosts" contiene al suo interno gli indirizzi IP della macchina utili per la mappatura stessa del nome host del server con il relativo indirizzo, es:

```
127.0.0.1       localhost       prova
192.168.50.20   prova.uniss.it  posta
```

in questo il sistema associa al nome "*localhost*" e al nome "*posta*" l'indirizzo "*127.0.0.1*" che è un particolare indirizzo impiegato ad esempio negli "*unix socket*". All'indirizzo IP di comunicazione verso l'esterno viene invece associato un nome completo a dominio e lo stesso nome host breve.

Il file "*networks*" contiene informazioni sulle reti di appartenenza, es:

```
default         0.0.0.0
loopback        127.0.0.0
link-local      169.254.0.0
localnet        192.168.50.0
```

Il file "*interfaces*" contiene la configurazione di tutte le interfacce di rete, es:

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback
```

```
# The primary network interface
# Modificato da Gavino, da usare la scheda  col driver vmxnet
# che sembra in questo caso essere la eth0
auto eth0
iface eth0 inet static
      address 192.168.50.20
      netmask 255.255.255.0
      network 192.168.50.0
      broadcast 192.168.50.255
      gateway 192.168.50.1

#auto eth1
iface eth1 inet static
  address 192.168.50.20
  netmask 255.255.255.0
  network 192.168.50.0
  broadcast 192.168.50.255
  gateway 192.168.50.1
```

analizzandolo si possono trarre le seguenti informazioni:

- la rete lo (di loopback) viene avviata al boot in automatico
- la rete eth0 presenta una configurazione statica, dove cioè vengono assegnati tutti i parametri quali l'IP, la netmask, il gateway, etc, e viene avviata al boot
- la rete eth1 presenta anch'essa una configurazione statica quale l'altra, ma non viene avviata al boot.

Per quanto riguarda invece l'avvio, stop e il riavvio dei servizio di rete, lo script incaricato di tale compito è il seguente:

```
/etc/init.d/networking
```

che parserizza i file di configurazione (con particolare riguardo per interfaces) ed esegue i relativi comandi di avvio, stop o start.
Un esempio dei comandi che è possibile impiegare da shell per la configurazione dei servizi rete e:

- ifconfig
- ifup
- route

Il comando ifconfig permette, tra le altre cose, di modificare l'indirizzo fisico MAC della scheda di rete, es:

```
ifconfig eth0 hw ether nuovoMAC
```

e va eseguito prima dell'avvio della scheda. Naturalmente al boot viene ripristinato il precedente valore, per renderlo persistente è necessario, ad esempio, il suo inserimento nello script di avvio dei servizi di rete.

Vi sono poi dei tools di visualizzazione, analisi e modifica di altri parametri ancora più specifici, quali:

- ethtool
- netstat

Soffermiamoci sul comando "*netstat*".
Tramite questo è possibile visualizzare tutte le porte di rete aperte sul sistema, sia che si tratti di porte di ascolto (server) o client, tcp o udp, unix socket o di rete, attive o in chiusura, etc.
Esempio di "netstat -ln":

```
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:110             0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:25              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:443             0.0.0.0:*               LISTEN
udp        0      0 192.168.50.20:123       0.0.0.0:*
udp        0      0 127.0.0.1:123           0.0.0.0:*
udp        0      0 0.0.0.0:123             0.0.0.0:*
Active UNIX domain sockets (only servers)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  2      [ ACC ]     STREAM     LISTENING     11293    public/cleanup
unix  2      [ ACC ]     STREAM     LISTENING     11300    private/tlsmgr
unix  2      [ ACC ]     STREAM     LISTENING     11304    private/rewrite
```

Esempio di "*netstat -ln*":

```
Active Internet connections (w/o servers)
tcp        0      0 192.168.50.20:25        192.168.51.20:39137     TIME_WAIT
tcp        0      0 192.168.50.20:443       192.168.50.253:58771    ESTABLISHED
tcp        0      0 192.168.50.20:25        192.168.51.20:39216     TIME_WAIT
tcp        0      0 192.168.50.20:110       172.16.34.96:2555       TIME_WAIT
tcp        0      0 192.168.50.20:110       192.168.50.253:40373    TIME_WAIT
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  2      [ ]         DGRAM                    6141     @/com/ubuntu/upstart
unix  2      [ ]         DGRAM                    6273     @/org/kernel/udev/ude
vd
unix  36     [ ]         DGRAM                    10934    /dev/log
unix  3      [ ]         STREAM     CONNECTED     21163263 private/rewrite
unix  3      [ ]         STREAM     CONNECTED     21163262
unix  3      [ ]         STREAM     CONNECTED     21163258 private/proxymap
```

Le diciture eth0, o eth1, impiegate nei precedenti fili di configurazione sono dei nomi logici assegnati alle schede fisiche dal kernel al momento del boot. Il file di mappatura è il seguente:

```
/etc/udev/rules.d/70-persistent-net.rules
```

dove, nel caso si aggiungesse una scheda, anche di sostituzione ad una già presente, il kernel aggiungerebbe un ulteriore nome logico. Per evitarlo è possibile cancellare il nome logico della scheda da sostituire prima dello spegnimento del server per la sostituzione.

Note su sistemi RedHat.
Nei sistemi non Debian-like, quali RedHat, il lavoro svolto dal file “interfaces” è svolto da un file per ogni scheda configurata, anche nel caso di sub interfacce.
Il nome di uno di questi, nell'esempio della scheda eth0 è il seguente:

/etc/sysconfig/network-scripts/ifcfg-eth0

e contiene al suo interno informazioni analoghe del file “interfaces”:

```
DEVICE=eth0
IPADDR=192.168.50.20
NETMASK=255.255.255.0
NETWORK=192.168.50.0
BROADCAST=192.168.50.255
BOOTPROTO=none
ONBOOT=yes
```

nel quale è possibile inserire anche informazioni riguardo al gateway da impiegare per tale scheda.
In generale invece il gateway è inserito nel file:

/etc/sysconfig/network

```
NETWORKING=yes
HOSTNAME=……
GATEWAY=192.168.50.1
```

7.2 Il servizio di tempo ntp

Il Network Time Protocol (ntp) è un sistema per la sincronizzazione del tempo di orologio dei calcolatori attraverso la rete Internet, sviluppato principalmente presso l'università del Delaware negli Stati Uniti. Ne sono state definite 3 versioni: la 1 nel 1988, la 2 nel 1989 e la 3 nel 1992. La versione corrente è la 3 che è compatibile con le precedenti. Per facilitare l'uso di NTP sui personal computer è stata definita la versione semplificata Simplified NTP (SNTP 1995).
Le principali caratteristiche del servizio ntp sono:

- è completamente automatico e mantiene la sincronizzazione in modo continuativo;
- è adatto alla sincronizzazione sia di un solo calcolatore, sia di intere reti di calcolatori;
- è progettato per essere resistente agli errori e si autoconfigura dinamicamente;
- diffonde il tempo UTC, quindi è indipendente dai fusi orari e dalle ore legali;
- la precisione di sincronizzazione arriva fino ad 1 millisecondo.

Un server primario ntp, detto anche di strato 1, è un calcolatore collegato ad un orologio di alta precisione e dotato di un software ntp. Altri calcolatori, detti di strato 2, dotati di un software similare, chiedono la sincronizzazione del proprio tempo di sistema al server primario che risponde con dei messaggi di sincronizzazione. I calcolatori di strato 2 possono a loro volta sincronizzare altri calcolatori, detti di strato 3, e così via fino a 16 strati. Man mano che ci si allontana dallo strato 1 la precisione della sincronizzazione diminuisce. In questa struttura, ciascun calcolatore può essere contemporaneamente server per le macchine di strato inferiore che si sincronizzano su di esso e

client per la macchina di strato superiore da cui esso stesso si sincronizza. Ogni server può avere alcune centinaia di client, quindi il numero di macchine sincronizzabili indirettamente da un singolo server primario è praticamente illimitato. Per rendere il sistema più affidabile, un client può avere più di un server di strato superiore. In questo caso, il software ntp misura continuamente le caratteristiche di stabilità e precisione dei possibili server, scegliendo di volta in volta come riferimento quello con le migliori caratteristiche.

In base a quanto detto, il processo che svolge le funzioni di server del servizio è lo stesso che svolge le funzioni di client. La differenza risiede nelle configurazioni del processo che abilitano o meno la richiesta del tempo al server stesso.
Lo script che si occupa di avviare il servizio è il seguente:

/etc/init.d/ntp

e la sua configurazione è memorizzata nel file "*/etc/ntp.conf*". Un esempio della configurazione client è il seguente:

```
tinker panic 0
driftfile /var/lib/ntp/ntp.drift
# Enable this if you want statistics to be logged.
#statsdir /var/log/ntpstats/

statistics loopstats peerstats clockstats
filegen loopstats file loopstats type day enable
filegen peerstats file peerstats type day enable
filegen clockstats file clockstats type day enable

# You do need to talk to an NTP server or two (or three).
#server ntp.ubuntu.com
server ntp1.ammin.uniss.it

# By default, exchange time with everybody, but don't allow configuration.
restrict -4 default kod notrap nomodify nopeer noquery
restrict -6 default kod notrap nomodify nopeer noquery

# Local users may interrogate the ntp server more closely.
restrict 127.0.0.1
restrict ::1
```

nel quale le righe che ne definiscono le caratteristiche client e non server sono i "restrict" alle query. Lo stesso file, per una configurazione server è invece di questo tipo:

```
# Prohibit general access to this service.
restrict default notrust nomodify noquery notrap

# Permit all access over the loopback interface.  This could
# be tightened as well, but to do so would effect some of
# the administrative functions.
restrict 127.0.0.1
restrict 127.127.0.0 mask 255.255.0.0 # internal clocks

...
restrict 193.204.201.0 mask 255.255.255.0 nomodify noquery notrap
restrict 193.204.205.0 mask 255.255.255.0 nomodify noquery notrap
restrict 192.168.50.0 mask 255.255.255.0 nomodify noquery notrap
```

...

server ntp1.inrim.it
server ntp2.inrim.it
server ntp.metas.ch
server tempo.cstv.to.cnr.it
server ntp.my-inbox.co.uk
server clock.isc.org
#server ntp-cup.external.hp.com
server swisstime.ethz.ch
server ntp1.cs.mu.OZ.AU
server ntp0.cs.mu.OZ.AU
server tick.usask.ca
server tock.usask.ca
server ntps1-0.uni-erlangen.de
server ntps1-1.uni-erlangen.de
server ntps1-2.uni-erlangen.de
server ntps1-3.uni-erlangen.de
server 127.127.1.0 # local clock
fudge 127.127.1.0 stratum 10

driftfile /var/lib/ntp/drift
logfile /var/log/ntp/ntpd.log
statistics loopstats peerstats clockstats

filegen loopstats file loopstats type day enable
filegen peerstats file peerstats type day enable
filegen clockstats file clockstats type day enable

nel quale le righe che ne definiscono le caratteristiche server sono il grande numero di server, che sarebbe meglio fossero di stratum 1 o 2, e le "*restrict*" sulle reti abilitate.

Per disabilitare di default l'invio dell'orario ai client la "restrict" da usare è:

```
restrict default ignore
```

E' importante che tra i server, magari con un valore di stratum alto (es: 10), ci sia l'indirizzo "*127.127.1.0*" che è specifico per il clock del bios di sistema. In questo modo, nel caso tutti i server siano irraggiungibili, il servizio è abilitato a prendere l'orario anche dal BIOS interno.
Infatti è doveroso ricordare che fino a quando il servizio non riesca a stabilizzare l'orario, anche se già attivo, non lo diffonde ai client.

Nel caso vi sia una grande differenza tra l'orario impostato nel client e l'orario riportato dal server (più di 12 ore) la sincronizzazione non avviene, ed è necessario un intervento dell'utente. Per questo, prima di avviare il servizio ntp è consigliabile eseguire il comando:

```
ntpdate nomeServer
```

che allinea l'orario interno con quello del server indicato.

Il protocollo ntp impiega per lo scambio di orario tra client e server dei pacchetti "*udp*" e come tali senza connessione. La porta standard di comunicazione su cui si attiva il servizio è la 123.
Log "*netstat -ln | more*":

```
udp    0    0    192.168.51.5:123     0.0.0.0:*
udp    0    0    127.0.0.1:123        0.0.0.0:*
udp    0    0    0.0.0.0:123          0.0.0.0:*
```

7.3 Il servizio ssh

Il servizio ssh o "*secure shell*" è un protocollo di rete che abilita lo scambio di informazioni tra dispositivi impiegando un canale di comunicazione sicuro (criptato). Uno dei suoi impieghi principali è quello di collegamento alla shell (da cui prende il nome) di un sistema per permetterne l'amministrazione da remoto, e in tal senso ha soppiantato il servizio di "*telnet*" che non permette lo scambio di dati su canale protetto.

Come avviene lo scambio dei certificati?
Per fare in modo che i dati vengano inviati sulla rete in maniera crittografata client e server si scambiano delle chiavi di crittografia secondo una modalità chiamata "*Crittografia Assimetrica*" o "*Crittografia a chiave pubblica e privata*", che si basano su dimostrazioni matematiche molto complesse. Noi ne vedremo solo il funzionamento:

1. Il client richiede il servizio al server ssh.

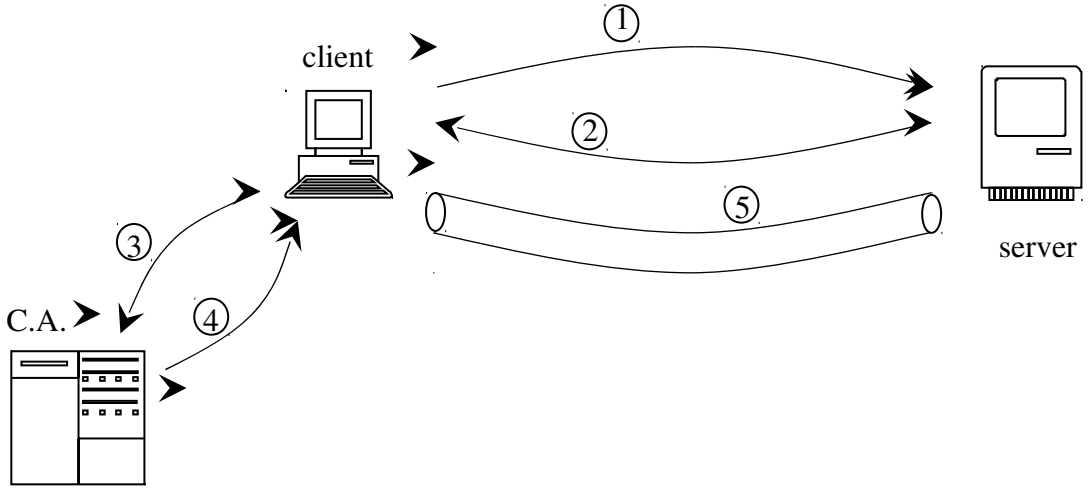
2. Il server comunica al client che la connessione avverrà in SSL e per questo gli invia il suo "*certificato pubblico*".

3. Il client ssh può controllare presso un "*ente terzo*" (menzionato nel certificato pubblico) che il certificato pubblico del server sia valido. Per fare questo invia quel certificato ad una "*Certificate Authority*" (autorità di certificazione, C.A.), che gli da conferma o meno dell'autenticità del certificato pubblico del server;

4. Se la Certificate Authority non riconosce valido il certificato, il client ssh chiede all'utente come comportarsi, cioè se continuare o meno con la connessione. Se si decide di continuare o se la Certificate Authority approva il certificato pubblico del server, il client invia al server il suo certificato pubblico.

5. Da questo momento in poi i dati sono inviati in maniera crittografata.
   Il client invia i dati crittografandoli con il certificato pubblico del server e il server invia i dati crittografandoli con il certificato pubblico del client.

Il server de-crittografa i dati ricevuti con il suo "*certificato privato*", infatti questi erano stati crittografati dal client con il certificato pubblico del server. Stessa cosa fa il client, impiegando però il suo certificato privato.



Perché nessuno può leggere i dati scambiati tra i due?

- Perché serve il corrispondente "*certificato privato*" per de-crittografare i dati crittografati con un particolare "*certificato pubblico*". Tentare di farlo senza il corrispondente privato è una impresa (a seconda della lunghezza delle chiavi presenti nei certificati) "***ritenuta quasi impossibile***".
- Il certificati privati non vengono **MAI** inviati sulla rete (e non devono essere mai scambiati con nessuno), e dunque solo i due interlocutori capiscono ciò che si inviano.

Si è evidenziato il fatto che sia quasi impossibile de-crittografare dati senza la corrispondente chiave privata perché ad esempio con una chiave a **128 bit** sia hanno **$3{,}4 \times 10^{38}$** combinazioni. Anche con la potenza dei computer di oggi si stima che ci vogliano **parecchi anni** per poterlo fare.
Anche se qualcuno tentasse l'impresa, avrebbe dei dati vecchi di molti anni, che i server non accetterebbero più.
Importante è il ruolo delle società di certificazione, perché senza di queste uno si potrebbe fingere qualcun altro senza aver bisogno del certificato privato di chi si finge di essere.
Queste metodologie stanno anche alla base della "firma digitale".

L'installazione del servizio avviene tramite la scelta del pacchetto "*openssh-server*". Durante l'installazione viene auto prodotto un certificato che verrà usato dal demone ssh per le comunicazioni protette.
Tramite l'uso dei certificati è possibile fare in modo che l'autenticazione al server ssh avvenga non solo via password ma anche, o solo, via certificati, in modo da restringere ulteriormente l'accesso al server.
Per disabilitare l'accesso tramite password di sistema va impostato sul file:

```
/etc/ssh/sshd_config
```

il parametro:

```
PasswordAuthentication no
```

Per abilitare l'accesso tramite certificati vanno impostati nel file di prima i parametri:

```
PubkeyAuthentication    yes
AuthorizedKeysFile      %h/.ssh/authorized_keys
```

e compilato il file indicato in "*AuthorizedKeysFile*" con le chiavi pubbliche degli utenti che vogliano connettersi.

Un'altra funzionalità molto usato del demone ssh è quella di "*port forwarding*" o "*tunneling*" delle connessioni, che permette di impiegare il canale protetto per altre comunicazioni quali quelle grafiche del demone Xserver. Infatti non è raro poter veicolare un terminale grafico sulla sola porta ssh e in maniera criptata. Per fare questo è necessario impostare nel file di configurazione di sopra il parametro:

```
X11Forwarding yes
```

E' possibile configurare il comportamento anche del client ssh tramite i parametri del file di configurazione:

```
/etc/ssh/sshd_config
```

8 L'ottimizzazione del sistema

8.1 Premessa

In questo capitolo tratteremo vari argomenti inerenti un sistema Linux, anche di tipologia molto differente tra loro, ma che presentano degli aspetti legati all'ottimizzazione e dunque alle performance del sistema.

8.2 I parametri del kernel

Il kernel è il cuore di un sistema operativo GNU/Linux. Esistono varie tipologie di kernel, tra le altre, a 32 o a 64 bit, di tipo server o meno, etc. La differenza tra questi è nelle features caricate e/o al valore della configurazione delle stesse, che permette di modificare spesso anche sostanzialmente il funzionamento del sistema stesso. Queste modifiche possono, a seconda dei casi, individuarsi con l'installazione di un pacchetto kernel differente o con la modifica di alcuni parametri dello stesso.
A seconda del tipo di parametro è possibile la modifica anche a sistema operativo già avviato, o tramite il passaggio diretto dalla riga di avvio del kernel.

E' possibile visualizzare il valore dei parametri di funzionamento del kernel tramite differenti vie, tra le quali vi sono le richieste eseguite sul filesystem "*/proc*" o i comandi di "*sysctl*". Il filesystem "/proc" è un particolare fs che viene creato nel sistema proprio a tale scopo. Tra gli altri in questo filesystem sono caricate le informazioni di tutti i processi attivi sul sistema.
Vediamo alcuni esempi di chiamate al filesystem "/proc".

Visualizzazione delle informazioni della cpu "*cat /proc/cpustat*":

```
processor   : 0
vendor_id   : GenuineIntel
cpu family  : 6
model       : 13
model name  : Intel(R) Pentium(R) M processor 1.70GHz
stepping    : 6
cpu MHz     : 600.000
cache size  : 2048 KB
fdiv_bug    : no
hlt_bug     : no
f00f_bug    : no
coma_bug    : no
fpu         : yes
fpu_exception     : yes
cpuid level : 2
wp          : yes
flags       : fpu vme de pse tsc msr mce cx8 apic mtrr pge mca cmov
              clflush dts acpi mmx fxsr sse sse2 ss tm pbe up bts est tm2
bogomips    : 1196.02
clflush size      : 64
```

```
power management:
```

Visualizzazione delle informazioni sullo swap impostato nel sistema "*cat /proc/swaps*":

```
Filename          Type        Size        Used  Priority
/dev/sda5         partition   514040      0     -1
```

Visualizzazione e impostazione del parametro del gestore della memoria virtuale (vm) relativo alla probabilità di impiego dello swap da parte del sistema:

`cat /proc/sys/vm/swappiness`	60, valore di default del kernel
`echo 0 > /proc/sys/vm/swappiness`	0, valore consigliato per macchine virtuali

Visualizzazione e impostazione del parametro del kernel relativo alla massima dimensione della "*shared memory*", molto utile in ambienti di database server:

`cat /proc/sys/kernel/shmmax`	33554432, valore di default
`echo 1073741824 > /proc/sys/kernel/shmmax`	1 GB

Lo stesso tipo di operazioni eseguite sul filesystem "*/proc*" sono possibili tramite il comando già menzionato "*sysctl*", il cui uso è il seguente:

```
usage:  sysctl [-n] [-e] variable ...
sysctl [-n] [-e] [-q] -w variable=value ...
sysctl [-n] [-e] -a
sysctl [-n] [-e] [-q] -p <file>   (default /etc/sysctl.conf)
sysctl [-n] [-e] -A
```

La modifica di questi parametri, via fs "*/proc*" o via "*sysctl*" non viene conservata al successivo riavvio del sistema. A tale scopo è utile l'impiego del file:

```
/etc/sysctl.conf
```

nel quale è possibile inserire i parametri con il nuovo valore associato. Le impostazioni di questo file vengono caricate al boot. Per individuare il nome esatto del parametro che è necessario inserire all'interno del file è possibile eseguire una ricerca sul output del comando "*sysctl -a*".

Per alcuni parametri è più utile, o necessario, eseguirne la modifica a freddo, cioè passando il relativo valore direttamente dalla riga di comando del boot del kernel. Il file che permette una tale impostazione è dunque quello di configurazione del boot loader. Nel caso di "*grub*" il file in questione è "*/boot/grub/menu.lst*", di cui si allegano delle righe di esempio:

```
      title       Ubuntu 8.04.3 LTS, kernel 2.6.24-26-server
      root        (hd0,0)
      kernel      /vmlinuz-2.6.24-26-server root=/dev/mapper/vg00-lv00 ro quiet
splash clocksource=acpi_pm elevator=noop
      initrd      /initrd.img-2.6.24-26-server
      quiet
```

NOTA: Sulla versione Ubuntu Desktop 9.10 il pacchetto “grub” è sostituito dal pacchetto “grub-pc” che non contempla più il file “menu.lst”. Le configurazioni di questo ultimo devono dunque essere eseguite sul file “*/etc/default/grub*” e rese attive tramite il comando:

```
update-grub
```

affinché siano caricate nella configurazione del boot loader.

8.3 La buffer cache

La “*buffer cache*” è quella porzione della memoria centrale che è impiegata per salvare tutte le informazioni che devono essere scambiate con i dispositivi di memoria di massa e che viene gestita dal sistema di processi della “*virtual memory*”. Il suo utilizzo è di basilare aiuto per l'ottimizzazione delle performance di un sistema, in quanto la memoria centrale risulta molto più veloce di quella di massa, e un impiego ottimale che mappi parti dei dispositivi di memoria di massa in memoria centrale rende l'accesso ai dati molto più veloce.
Si pensi al caso in cui sia utile leggere da disco una certa quantità di dati, e che parte del tempo sia perso in attesa che il dispositivo si posizioni sulle tracce contenenti i dati da leggere. Se poco dopo, il sistema dovesse richiedere altri dati, conseguenti ai precedenti (caso molto probabile), si dovrebbe attendere ancora del tempo per il posizionamento delle testine dei dispositivi. Se invece fosse possibile richiedere al disco una quantità di dati maggiore, da salvare in una data porzione di memoria, l'esecuzione dei processi ne risulterebbe aumentata.
Questa politica di gestione è impiegata in tutti quei sistemi in cui vi sia uno scambio di dati tra dispositivi che operano a velocità molto differente tra loro, quali:

- cpu ↔ memoria centrale
- memoria centrale ↔ sistemi di disco

Una stessa politica è impiegata anche a livello applicativo ad esempio nell'ambiente dei database, nei quali parte del db salvato su disco viene mappato in una porzione di memoria centrale dedicata. In questi sistemi l'impiego della buffer cache di sistema operativo è controproducente, in quanto si viene a creare un doppio livello di cache che non ne aumenta le performance, ma anzi ne duplica le informazioni. In questi casi è utile impiegare dei filesystem dedicati per salvare i files impiegati dal database, e montare questi ultimi in modo da non impiegare la buffer cache o da impiegarla in maniera appropriata. A seconda del sistema operativo e dei filesystem le modalità di applicazione sono differenti tra loro.

L'impiego della buffer cache è utile anche nel caso di dati modificati rispetto al corrispettivo valore sul dico e in attesa di essere scritti. In questo caso la politica di gestione è attuata con l'obbiettivo dell'incremento delle performance di scrittura. In tali casi si parla di “*write cache*”. Uno dei parametri più importanti della write cache è:

```
/proc/sys/vm/dirty_ratio
```

che definisce la massima percentuale di buffer cache che è impiegata per i dati in attesa di scrittura su disco, da qui l'indicazione di sporchi.

In linea di massima un sistema Linux impiega tutta la quantità di ram disponibile. Questo significa che spesso un sistema Linux presenta tutta la ram occupata anche in presenza di un piccolo carico di sistema. Il concetto, lato server, risulta molto oculato, in quanto non ci sarebbe motivo per non impiegare tutta la memoria che l'hardware rende disponibile al sistema. Ma chi impiega tale memoria se i processi non lo fanno? La buffer cache. E che dati sono memorizzati su di essa? Porzioni dei dati su disco.

Esempio: si veda parte dell'output del comando "*top*" su un sistema:

```
top - 18:59:39 up  1:15,  3 users,  load average: 0.79, 0.49, 0.22
Tasks: 131 total,   2 running, 129 sleeping,   0 stopped,   0 zombie
Cpu(s):  2.0%us,  0.7%sy,  0.0%ni, 95.5%id,  1.5%wa,  0.2%hi,  0.0%si,  0.0%st
Mem:   2061088k total,   539292k used,  1521796k free,    33768k buffers
Swap:   514040k total,        0k used,   514040k free,   265852k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 1374 root      20   0  188m  29m 7344 S  1.0  1.4   1:38.17 Xorg
 2002 gvmura    20   0 44556  12m 9.8m S  0.7  0.6   0:12.69 gnome-terminal
 1750 gvmura    20   0  234m  98m  60m S  0.5  4.9   2:28.63 soffice.bin
 2800 gvmura    20   0  2472 1180  896 R  0.5  0.1   0:00.22 top
   16 root      15  -5     0    0    0 S  0.2  0.0   0:01.09 ata/0
  708 messageb  20   0  3172 1492  748 S  0.2  0.1   0:01.18 dbus-daemon
    1 root      20   0  2664 1532 1128 S  0.0  0.1   0:00.93 init
```

e si veda lo stesso output dopo l'esecuzione del comando "*grep -r prova /*":

```
top - 19:12:50 up  1:29,  3 users,  load average: 0.03, 0.33, 0.33
Tasks: 131 total,   2 running, 129 sleeping,   0 stopped,   0 zombie
Cpu(s):  1.0%us,  0.7%sy,  0.0%ni, 98.3%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:   2061088k total,  1981716k used,    79372k free,    34860k buffers
Swap:   514040k total,     1152k used,   512888k free,  1707456k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 2949 gvmura    20   0  2472 1180  896 R  0.7  0.1   0:00.05 top
 1374 root      20   0  191m  29m 7344 S  0.5  1.5   1:47.99 Xorg
 1750 gvmura    20   0  235m  98m  60m S  0.5  4.9   2:53.82 soffice.bin
   34 root      15  -5     0    0    0 S  0.2  0.0   0:02.64 scsi_eh_1
 1016 root      20   0  3420 1140  972 S  0.2  0.1   0:02.35 hald-addon-stor
 2002 gvmura    20   0 44556  13m 9.8m S  0.2  0.6   0:15.03 gnome-terminal
    1 root      20   0  2664 1248 1128 S  0.0  0.1   0:00.93 init
```

dal quale si osserva che la memoria libera del sistema passa da 1.5 GB a 79 MB e la memoria cached da 266 MB a 1.7 GB.

8.4 La compilazione del kernel

Il kernel Linux è, come abbiamo già detto, il cuore del sistema operativo. Al suo interno sono presenti tutte le strutture dati necessarie per il funzionamento del sistema. Per questo motivo negli anni è divenuto sempre più grande, da cui la recente frase dello stesso Linus Torvalds che ha dichiarato che il kernel Linux è divenuto "Huge And Bloated" (enorme e gonfio). La motivazione è data principalmente dall'aumento delle funzionalità e dall'aumento delle periferiche riconosciute.
Le varie distribuzioni rilasciano differenti kernel per differenti ambienti operativi (es: server o desktop), proprio per permettere di avere performance mirate ai differenti ambienti operativi.

Questo può però non bastare per la richiesta di tuning degli amministratori di sistema, e pertanto è disponibile la funzionalità di compilazione del kernel con i soli moduli necessari.
Per esempio, nel caso che il proprio sistema non contempli delle periferiche audio, è possibile compilare un kernel privo di tali funzionalità, e pertanto più piccolo e snello.
Naturalmente questa scelta diminuisce la versatilità del sistema, e necessita di un aumento del lavoro di amministrazione in caso di aggiornamenti del kernel stesso, che dovra sempre essere ricompilato a parte.

I passi logici per l'esecuzione del lavoro di compilazione del kernel sono:

1) installazione dei pacchetti propedeutici necessari e dei sorgenti del kernel
2) configurazione dei parametri
3) compilazione del kernel vero e proprio
4) installazione e test del nuovo kernel

Vediamo nei dettagli i vari passi.
Installazione dei pacchetti propedeutici:

```
apt-get install buildessential fakeroot linux-source kernel-package ccache libncurses5-dev
```

Per la configurazione dei parametri è utile partire dai valori già impostati nel kernel attualmente avviato, che è possibile trovare nel file config presente nella cartella /boot, es:

```
/boot/config-2.6.31-20-generic
```

copiandolo come "*.config*" nella directory dei sorgenti del kernel, eseguite poi il comando:

```
make oldconfig
```

La modifica dei parametri potrà a questo punto avvenire o andando direttamente a modificare i valori presenti all'interno del file ".config" o tramite il comando:

```
make menuconfig
```

E' ora possibile passare alla compilazione vera e propria tramite il comando:

```
make-kpkg buildpackage -rev=test1 –initrd kernel_image kernel_headers
```

A fine compilazione sarà possibile installare i due pacchetti "*.deb*" generati dal processo di compilazione con il comando:

```
dpkg -i pacchettoKenel pacchettoHeaders
```

Il processo di installazione dovrebbe installare in automatico il nuovo kernel nella directory "*/boot*" e aggiornare il boot loader grub con la nuova riga di start.
Per finire si dovrà testare il boot del sistema tramite il riavvio dello stesso. Nel caso vi fossero degli errori l'avvio del sistema si fermerà con un "*kernel panic*" e potrà essere utile analizzare le righe poco prima di questa per capire come risolvere il problema, al quale dovrà seguire una successiva compilazione del kernel.

8.5 Gli scripts di avvio al boot

All'avvio del sistema operativo il kernel si occupa di eseguire gli scripts che trova in alcune directory particolari del sistema, in modo che, dopo il boot, il sistema sia pronto per funzionare così come richiesto dall'amministratore. Se ad esempio il sistema è impiegato come server web potrebbe essere necessario avviare il servizio Apache, o il demone ssh per una amministrazione del server da remoto. Questo compito è eseguito dal demone "*init*" che per questo è indicato come il padre di tutti gli altri processi.
In un sistema Linux, ma anche in un generico Unix, vi sono 5 diversi livelli di lavoro, numerati da 1 a 5, e per questo si dice che vi siano 5 diversi livelli di "*init*". In un particola istante il sistema si troverà in uno di questi livelli. Non tutti i livelli sono però impiegati effettivamente. Ad esempio, i livelli usati in un sistema Debian sono il 1° e il 2°, mentre in un sistema RedHat sono il 1°, il 3° e il 5°. La differenza tra questi livelli è data solo da una convenzione. Ad esempio, un sistema a riga di comando su Debian usa il livello 2, mentre su RedHat il livello 3, anche se gli scripts all'interno di questi livelli possono essere anche identici. Un sistema grafico su Debian usa comunque il livello 2, mentre su RedHat il livello 5. Si capisce dunque che questi livelli differiscono solo per gli scripts che avviano al boot, infatti molti software inseriscono il loro script di avvio in tutti i livelli di sistema, in modo che venga avviato qualunque sia il livello di boot.
Il livello 1 è chiamato livello di "*single user*" in quanto vengono avviati solo i processi di sistema indispensabili (è comunque una convenzione). E' usato ad esempio dall'utente root per risolvere problemi del sistema.

Per vedere in che livello sia un sistema si esegua:

```
who -r
```

mentre per cambiare livello:

```
init numeroNuovoLivello
```

Durante il cambio di un livello, prima di avviare gli scripts di start del livello di destinazione, vengono eseguito gli scripts di stop del livello di a cui ci si trova, e che si trovano anch'essi all'interno del livello, ma con un formato differente da quelli di start.

Il kernel infatti, quando entra in un livello, esegue tutti gli scripts che iniziano per "*S*" e quando esce da un livello (per passare ad un altro) esegue tutti gli scripts che iniziano per "*K*". L'ordine di esecuzione è dato da un numero che segue la lettera "*S*" o "*K*".

Dove si trovano gli scripts di un livello?
Gli scripts di ciascun livello si trovano in particolari directory, sotto la "*/etc*" chiamate "*rcN.d*", dove "*N*" indica il numero del livello. Ad esempio:

```
/etc/rc1.d/...
/etc/rc2.d/...
… …
```

Esempio di output del comando "*ls -la /etc/rc2.d/*":

```
drwxr-xr-x   2 root root  4096 2010-03-17 18:00 .
drwxr-xr-x 139 root root 12288 2010-03-20 15:15 ..
-rw-r--r--   1 root root   677 2009-11-10 10:44 README
lrwxrwxrwx   1 root root    20 2009-11-07 13:07 S20kerneloops -> ../init.d/kerneloops
lrwxrwxrwx   1 root root    27 2009-11-07 13:07 S20speech-dispatcher -> ../init.d/speech-dispatcher
lrwxrwxrwx   1 root root    17 2010-03-17 18:00 S20vboxdrv -> ../init.d/vboxdrv
lrwxrwxrwx   1 root root    19 2009-11-07 13:06 S25bluetooth -> ../init.d/bluetooth
lrwxrwxrwx   1 root root    14 2009-11-07 13:03 S50cups -> ../init.d/cups
lrwxrwxrwx   1 root root    20 2009-11-07 13:09 S50pulseaudio -> ../init.d/pulseaudio
lrwxrwxrwx   1 root root    15 2009-11-07 13:05 S50rsync -> ../init.d/rsync
lrwxrwxrwx   1 root root    15 2009-11-07 13:08 S50saned -> ../init.d/saned
lrwxrwxrwx   1 root root    19 2009-11-07 13:05 S70dns-clean -> ../init.d/dns-clean
lrwxrwxrwx   1 root root    18 2009-11-07 13:05 S70pppd-dns -> ../init.d/pppd-dns
lrwxrwxrwx   1 root root    24 2009-11-07 13:06 S90binfmt-support -> ../init.d/binfmt-support
lrwxrwxrwx   1 root root    22 2009-11-07 13:05 S99acpi-support -> ../init.d/acpi-support
lrwxrwxrwx   1 root root    21 2009-11-07 13:12 S99grub-common -> ../init.d/grub-common
lrwxrwxrwx   1 root root    21 2009-11-07 13:05 S99laptop-mode -> ../init.d/laptop-mode
lrwxrwxrwx   1 root root    18 2009-11-07 12:47 S99ondemand -> ../init.d/ondemand
lrwxrwxrwx   1 root root    18 2009-11-07 12:47 S99rc.local -> ../init.d/rc.local
```

da cui si capisce che nel sistema dell'esempio non ci sono scripts di stop del livello 2 e inoltre che tali scripts sono dei link a scripts che si trovano nella cartella:

```
/etc/init.d
```

niente vieta comunque di inserire scripts veri e propri; questa è infatti solo una convenzione.

Di solito uno script "*S*", di avvio di un servizio, è identico allo script "*K*" di stop dello stesso servizio. Infatti il kernel passa il parametro "*start*" agli scripts "*S*" e il parametro "*stop*" a quelli "*K*". Se dunque lo stesso script esegue lo start del servizio voluto alla ricezione di "*start*" e lo stop alla ricezione di "*stop*", non si ha l'esigenza di scripts differenti.

Esempio di script di start o stop "*/etc/init.d/ntp*":

```
#!/bin/sh

### BEGIN INIT INFO
# Provides:        ntp
# Required-Start:  $network $remote_fs $syslog
# Required-Stop:   $network $remote_fs $syslog
# Default-Start:   2 3 4 5
# Default-Stop:    0 1 6
# Short-Description: Start NTP daemon
### END INIT INFO

PATH=/sbin:/bin:/usr/sbin:/usr/bin

. /lib/lsb/init-functions

NAME=ntp
DAEMON=/usr/sbin/ntpd
PIDFILE=/var/run/ntpd.pid

test -x $DAEMON || exit 5

if [ -r /etc/default/$NAME ]; then
        . /etc/default/$NAME
fi

if [ -e /etc/ntp.conf.dhcp ]; then
        NTPD_OPTS="$NTPD_OPTS -c /etc/ntp.conf.dhcp"
fi
```

```
RUNASUSER=ntp
UGID=$(getent passwd $RUNASUSER | cut -f 3,4 -d:) || true

case $1 in
        start)
                log_daemon_msg "Starting NTP server" "ntpd"
                if [ -z "$UGID" ]; then
                        log_failure_msg "user \"$RUNASUSER\" does not exist"
                        exit 1
                fi
                start-stop-daemon --start --quiet --oknodo --pidfile $PIDFILE --startas
$DAEMON -- -p $PIDFILE -u $UGID $NTPD_OPTS
                log_end_msg $?
                ;;
        stop)
                log_daemon_msg "Stopping NTP server" "ntpd"
                start-stop-daemon --stop --quiet --oknodo --pidfile $PIDFILE
                log_end_msg $?
                rm -f $PIDFILE
                ;;
        restart|force-reload)
                $0 stop && sleep 2 && $0 start
                ;;
        try-restart)
                if $0 status >/dev/null; then
                        $0 restart
                else
                        exit 0
                fi
                ;;
        reload)
                exit 3
                ;;
        status)
                pidofproc -p $PIDFILE $DAEMON >/dev/null
                status=$?
                if [ $status -eq 0 ]; then
                        log_success_msg "NTP server is running."
                else
                        log_failure_msg "NTP server is not running."
                fi
                exit $status
                ;;
        *)
                echo "Usage: $0 {start|stop|restart|try-restart|force-reload|status}"
                exit 2
                ;;
esac
```

Un sistema Debian-like, prima di eseguire gli scripts "*S*" del livello di lavoro (es: 2), esegue durante il boot gli scripts "*S*" cotenuti nella directory:

```
/etc/rcS.d
```

in quest'ultima directory sono infatti presenti scripts generici che andrebbero eseguiti in ogni fase di boot, a prescindere dal livello di lavoro effettivo.

9 Il backup del sistema

9.1 Premessa

Con il termine "*backup*" si intende l'insieme delle procedure atte alla salvaguardia dei dati di un sistema informatico. Spesso al posto di indicare il piano di backup di un sistema, è impiegato il temine di piano di "*disaster recovery*", che specifica tutti i passi necessari per il ripristino dei servizi, nel quale la parte relativa alla salvaguardia dei dati è si fondamentale ma non è l'unica presa in esame. Un piano di "*disaster recovery*" deve analizzare tutti quei processi che possano consentire di riattivare il servizio anche a fronte di un vero e proprio disastro, o al peggio anche solo valutare il costo accettabile a fronte di un dato evento disastroso.
Per esempio, se si volesse analizzare la sicurezza dei dati di una sala macchine contro ogni tipologia di eventi, andranno valutati aspetti non solo inerenti i sistemi informatici in quanto tali, ma anche tutti gli aspetti al contorno, quali ubicazione della sede (per esempio in zona sismica o meno), prevenzione incendi, possibilità di furti, atti vandalici, e così via. A piano ultimato si potrà poi decidere quali rischi siano accettabili o meno in base ai costi necessari per la loro messa in sicurezza, in modo da avere un quadro completo di analisi da presentare al management decisionale.

Le tipologie di esecuzione di un backup sono varie e dipendenti dalle scelte operate a livello di piano di recovery. Ad esempio:

- di salvaguardia dei soli dati o dei dati e delle applicazioni;
- eseguite a "*caldo*", cioè con l'applicazione attiva, o a "*freddo*", cioè con l'applicazione stoppata;
- complete o incrementali;

La scelta della modalità più opportuna è spesso dipendente dal livello di servizio richiesto, in merito alla possibilità di fermo dello stesso o ai tempi accettabili per una rimessa in operatività in caso di guasto o disastro. Inoltre, la possibilità di backup a "*caldo*" è spesso dipendente dall'applicazione stessa, o in altre parole deve essere eseguita con l'integrazione di comandi dell'applicazione.
Per esempio, il backup di un database a caldo necessita dell'impiego di comandi specifici dell'applicazione database, in modo che i dati salvati siano consistenti.

9.2 Il backup di una partizione

Una delle possibili soluzioni per l'esecuzione di un backup è quella del salvataggio dell'intera partizione. Questa soluzione salva dati e applicazioni, dal momento che non differenzia la tipologia dei dati da salvare; è naturalmente un'operazione di backup "*freddo*", in quanto la partizione non deve essere impiegata durante il salvataggio, ed è di tipo non incrementale in quanto non vengono controllati backup precedenti per considerare i soli dati modificati. Se la partizione da salvare è quella in cui risiedono parti del sistema non smontabili (per esempio la "/"), è necessario eseguire

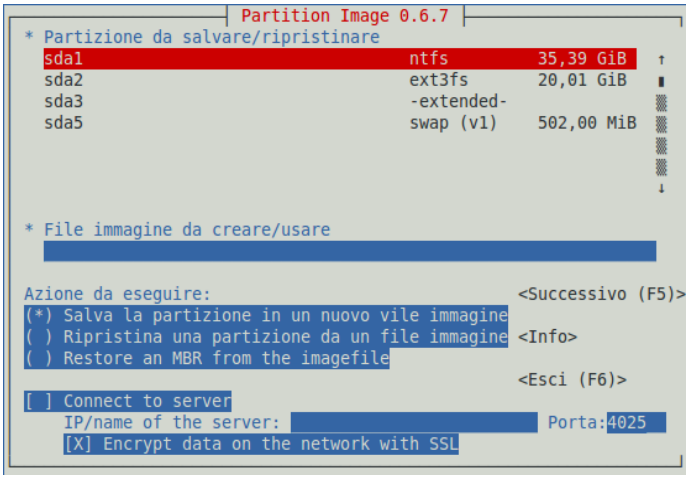
l'operazione tramite l'avvio di un cd di boot, inoltre, all'aumentare della dimensione della partizione di cui fare il backup, come nel caso attuale di dischi molto capienti, l'impiego di questa tecnica risulta poco vantaggioso.
I programmi che permettono il salvataggio di una partizione sono vari. In questo testo prenderemo in considerazione i soli "*dd*" e "*partimage*"; la loro differenza sostanziale risiede nel fatto che nel primo vengono salvati tutti i blocchi della partizione, senza analizzare se siano o meno utilizzati dal filesystem, mentre nel secondo solo i blocchi impiegati dal filesystem.

Il comando "*dd*".
E' un comando che copia una serie di blocchi da un dispositivo ad un altro, in cui i dispositivi possono essere sia di tipo disco (o md o lv) che file. E' possibile specificare la dimensione del blocco da copiare e/o il numero di blocchi. Tramite questo comando è possibile copiare una intera partizione su un file, e per questo motivo la partizione non deve essere utilizzata (filesystem smontato).
Esempio di backup della partizione 3 del primo disco su un file che non dovrà risiedere sulla partizione 3:

```
dd bs=512k if=/dev/sda3 of=/backup/part3.dd
```

La dimensione del file sarà identica alla dimensione della partizione.
Il recovery dei dati dovrà avvenire secondo la stessa modalità, invertendo la parte "*if*" con la "*of*" e specificando la stessa dimensione del blocco "*bs*"

Il comando "*partimage*".
E' un comando dalla grafica testuale anche esso impiegato per il salvataggio di una partizione su un file. Anche in questo caso il file di backup non deve trovarsi sulla partizione da salvare. Verranno inoltre salvati i soli blocchi effettivamente utilizzati dal filesystem, e in un formato compresso, questo permette di avere dimensioni del file di backup molto inferiori alle dimensioni della partizione.
Esempio di grafica di "*partimage*":



Il recovery dei dati potrà avvenire secondo la stessa modalità, specificando però che si tratta di una operazione di ripristino.

9.3 Il backup tramite tar

Il comando "*tar*" è un comando molto utile dei sistemi Unix-like, che permette di creare copie di backup di file o di intere cartelle. Tramite questo comando è possibile creare vari tipi di backup, completi o incrementali, compressi o non compressi, preservando la data di modifica dei file, o i diritti sui file, e tante altre opzioni che rendono molto potente l'uso del comando.
Inoltre il comando "*tar*" permette la scrittura (e conseguente lettura) diretta del file di archivio su cassetta rimovibile.
La limitazione più importante del comando rimane quella della sola esecuzione di backup a "*freddo*", e dunque ad applicazione stoppata per tutto il tempo necessario per il backup.

I parametri più usati del comando sono:

`-f`	seguito dal file o dispositivo di archivio
`-x`	specifica l'operazione di estrazione dall'archivio
`-c`	" " creazione dell' "
`-t`	" " lista " "
`-z`	" " compressione o decompressione
`-g`	seguito da un file di lista, definisce un backup incrementale
`-p`	preserva i permessi dei file
`--atime-preserve`	non modifica la data di accesso ai file su cui esegue il backup (molto utile in caso tale data serva per altri scopi)

Esempio, savataggio della cartella "*/home*" sull'archivio compresso "*/backup/home.tgz*":

```
tar -cvzf /backup/home.tgz /home
```

Esempio, ripristino dell'archivio di sopra:

```
tar -xvzf /backup/home.tgz
```

l'archivio viene esploso a partire dalla directory di lavoro da cui si esegue il tar. Se per esempio il precedente comando fosse stato eseguito dalla "*/prova*", a fine comando avremo trovato in tale directory la:

```
/prova/home/...
```

Note sul backup incrementale.
Creazione:

```
tar -cvzf /backup/home-`date +%s`.tgz -g /backup/home.list /home
```

Alla prima esecuzione il file "*tgz*" conterrà tutti i file contenuti dalla cartella "*/home*"; la seconda volta conterrà solo i file nuovi o modificati, e così via. Per questo il file "*tgz*" non deve essere sempre lo stesso, in quanto altrimenti ogni precedente backup sarebbe sovrascritto dall'ultimo creato. Il file "*home.list*" è un file binario che viene impiegato per riconoscere i nuovi file o i modificati. Il ripristino dei dati necessita dell'estrazione di tutti i file "*tgz*" secondo l'ordine della loro creazione. Per eseguire un nuovo backup completo è sufficiente cancellare il file di log "*home.list*".

Note sul proprietario dei file ripristinati.
Nella fase di estrazione vengono assegnanti ai file ripristinati gli "*id*" dell'utente e del gruppo proprietari così come estratti dal sistema di partenza. Se il sistema di destinazione è lo stesso del sistema di partenza l'utente avente tale "*id*" dovrebbe essere lo stesso (se non viene cambiato per qualche motivo nel frattempo). Se invece il sistema di destinazione è diverso, è molto probabile che gli "*id*" non corrispondano, o che non vi sia un utente con tale "*id*". In questo caso viene visualizzato non un nome ma un valore numerico:

sistema di partenza

```
ls -la /tmp/cartella/
      drwxr-xr-x  2 gvmura gvmura 4096 2010-03-28 20:55 .
      drwxrwxrwt 15 root   root   4096 2010-03-28 21:40 ..
      -rw-r--r--  1 gvmura gvmura    3 2010-03-28 20:55 file
```

sistema di destinazione (non avente l' "*id*" del sistema di partenza):

```
ls -la /tmp/cartella
      drwxr-xr-x  2 gvmura gvmura 4096 2010-03-28 20:55 .
      drwxrwxrwt 15 root   root   4096 2010-03-28 21:40 ..
      -rw-r--r--  1   1001   1001    3 2010-03-28 20:55 file
```

oppure (dove l' "*id*" corrisponde ad un altro utente):

```
ls -la /tmp/cartella
      drwxr-xr-x  2 gvmura gvmura 4096 2010-03-28 20:55 .
      drwxrwxrwt 15 root   root   4096 2010-03-28 21:40 ..
      -rw-r--r--  1  rossi  rossi    3 2010-03-28 20:55 file
```

In questi casi può essere utile salvare anche l'elenco degli "*id*" del sistema di partenza, che è possibile reperire ad esempio dal file "*/etc/passwd*"

9.4 Uso delle snapshot di lv

L'impiego dei logical volume è utile anche nell'esecuzione dei processi di backup. Infatti, accettando un piccolo tempo di stop dell'applicazione, è possibile eseguire una snapshot del volume di cui si vuole fare il backup e operare come se ci si trovasse in una situazione di salvataggio a "*caldo*". Sequenza logica:

1) arresto dell'applicazione
2) creazione di una snapshot sul volume dell'applicazione (tempo molto ridotto)
3) riavvio dell'applicazione
4) mount del volume di snapshot
5) backup (per esempio tramite "*tar*") della directory relativa alla snapshot
6) arresto dell'applicazione
7) cancellazione della snapshot (tempo dipendente dalla quantità di dati che l'applicazione ha modificato dalla creazione della snapshot)
8) riavvio dell'applicazione

Il tempo totale di stop dell'applicazione è fissato principalmente dalla fase 7). Se il backup viene eseguito durante una fase di basso lavoro dell'applicazione, per esempio durante la notte nel caso di processi aziendali, anche la fase 7) avrà un tempo di esecuzione ridotto, simulando il caso di un backup a "*caldo*".

Bibliografia

Libri

- Eduardo Cilento "*Linux Performance and Tuning Guidelines*", Takechika Kunimasa, Byron Braswell, 2008 IBM RedPaper
- David Deeths "*Using NTP to Control and Synchronize System Clocks* ", 2001 Sun Microsystems
- Mendel Cooper "Advanced Bash Scripting Guide", 2005 Mendel Cooper
- Michael Jang "*Ubuntu Server Administration*", Elizabeth Zinkann, 2009 McGrawHill
- Sander van Vugt "*Pro Ubuntu Server Administration*", Samuel Cuella, 2009 Apress
- Sander van Vugt "Beginning *Ubuntu Server Administration*", Curtis Smith, 2008 Apress

Siti web

- http://www.gnu.org/home.it.html
- http://torvalds-family.blogspot.com
- http://www.redbooks.ibm.com/
- http://kbase.redhat.com/faq/en
- http://en.wikipedia.org/wiki/Main_Page
- http://wiki.ubuntu-it.org/Documentazione/Indice
- http://www.vmware.com/products/server/
  http://www.virtualbox.org/
- http://tldp.org/HOWTO/Software-RAID-0.4x-HOWTO.html
- http://sourceware.org/lvm2/
- http://www.ntp.org/
- http://www.openssh.org/
- http://www.sysresccd.org/Main_Page
- http://www.partimage.org/Main_Page