Come funziona Linux

Prima parte

Il boot

A fianco dei consueti articoli divulgativi, da questo mese inizieremo a conoscere anche gli aspetti più tecnici sul funzionamento di Linux, partendo da una descrizione su come avviene il boot del sistema operativo.

di Giuseppe Zanetti

La prima regola: leggere i manuali

L'articolo di una rivista non può pretendere di spiegare tutto su un particolare aspetto di Linux. Per comprendere meglio e per andare più a fondo degli argomenti trattati si può fare riferimento ad un buon libro, oppure alla documentazione di Linux (LDP, Linux Documentation Project), che si può prelevare da http://sunsite.unc.edu/mdw. Essa è composta da diversi manuali, rivolti sia all'utente che all'amministratore di sistema ed al programmatore, e da un buon numero di documenti, gli HOWTO, che descrivono diversi aspetti specifici di Linux, sia tecnici che pratici. Ad esempio l'Italian-HOWTO di Marco "gaio" Gaiarin spiega tutto ciò che può essere utile sapere per usare Linux in Italia: da come nazionalizzare la tastiera e il font di caratteri a dove trovare libri e CD in italiano su Linux.

Quello di Marco è l'unico HOWTO scritto direttamente in italiano. Per leggere gli altri è necessario conoscere un minimo di inglese tecnico oppure si possono prelevare le traduzioni dal sito dell'ILDP (Italian LDP), http://www.pluto.linux.it/ildp.

Lo strumento più importante per imparare ad usare un programma rimane comunque il manuale fornito col programma stesso. Se ci si compila da soli il programma generalmente nella directory che contiene i sorgenti è presente un file di nome README che spiega come installarlo. Di solito vi sono anche ulteriori documenti, probabilmente contenuti in una directory doc. Nel caso si parta da pacchetti già compilati, la maggior parte delle distribuzioni di Linux copia questi documenti nella directory /usr/doc.

La documentazione relativa alle funzioni del kernel di Linux ed ai diversi device driver forniti di serie con esso è invece contenuta all'interno della directory /usr/src/linux/Documentation. Una buona fonte di informazione sono anche i file README o i commenti contenuti nei sorgenti del kernel in /usr/src/linux.

Ogni programma (nel resto dell'articolo capiterà di utilizzare come sinonimi i termini "programma" e "comando") che si rispetti ha infine un manuale in linea, richiamabile scrivendo da shell il comando:

```
man argomento (es: man ls)
```

Esistono anche programmi che permettono di accedere ai manuali in linea dall'ambiente grafico X Window, ad esempio xman.

Una tipica pagina di manuale di un programma di Linux (compresi i comandi di sistema) è strutturata nelle seguenti parti:

NAME

Riporta il nome del comando o del programma in questione ed una brevissima descrizione di ciò che esso compie.

SYNOPSIS

Descrive la sintassi d'uso. Secondo la convenzione propria dei sistemi UNIX , gli switch, ovvero i selettori delle diverse opzioni applicabili ad un comando, devono essere preceduti dal simbolo - per distinguerli dai parametri veri e propri, ad esempio i nomi dei file su cui il comando opera (in "ls -la /etc", "ls" è il nome del comando, mentre le lettere "l" e "a" sono dei selettori e "/etc" un parametro).

I simboli che nella documentazione sono racchiusi fra parentesi quadre sono da considerarsi opzionali e possono non essere inseriti nella linea di comando, a meno ovviamente di non voler sfruttare l'opzione ad essi associata.

Dei punti di sospensione "..." indicano che l'ultimo parametro può essere ripetuto un numero indefinito di volte, come nel caso di quasi tutti i comandi che agiscono su file. In questo modo è possibile specificare direttamente sulla linea di comando i nomi di più file su cui fare agire il comando (ad esempio: rm pippo pluto paperino) oppure ottenere questi nomi dall'espansione di una espressione regolare (la linea di comando "rm *.gif" prima di essere

eseguita viene espansa, ad esempio, in "rm pippo.gif pluto.gif"). Ne riparleremo.

DESCRIPTION
Descrive lo scopo ed il funzionamento del comando.

OPTIONS
Viene descritto dettagliatamente il funzionamento di ognu na delle opzioni applicabili al comando.

ENVIRONMENT
Contiene informazioni sulle variabili d'ambiente che in fluenzano o vengono influenzate dal programma.

SEE ALSO
Indica eventuali altri manuali o documenti a cui fare riferimento per andare a fondo dell'argomento.

AUTHOR
Contiene eventuali note sull'autore e sul copyright del pro gramma.

BUGS
Segnala eventuali errori o malfunzionamenti conosciuti.

I manuali in linea sono suddivisi in sezioni:

Sezione	Descrizione
1	Comandi per gli utenti
2	Chiamate a funzioni del kernel
3	Subroutines e Funzioni del lin guaggio C
4	Descrizione dei device
5	Formato dei file
6	Giochi
7	Varie
8	Comandi per l'amministrazione del sistema
N	Manuali aggiunti dall'utente

Poiché spesso il medesimo nome può rappresentare più oggetti diversi (ad esempio passwd è contemporaneamente il nome del file di sistema che contiene le password e del comando che permette di modificarle), alcune voci sono riportate in più sezioni. Per accedervi si deve perciò specificare anche la sezione a cui fare riferimento.

```
man sezione argomento (es: man 5 passwd)
```

Per questo motivo spesso ci si riferisce ad una pagina di manuale specificandone fra parentesi anche la sezione, ad esempio passwd(5).

Nel caso non si conosca il nome preciso di un comando, è possibile effettuare una ricerca per parole chiave, scrivendo:

```
man -k parolachiave (es: man -k passwd)
```

verrà restituita una lista di manuali che nella descrizione contengono la parola chiave inserita.

La partenza di Linux

Quando accendiamo il nostro computer, per prima cosa viene eseguito il programma POST, contenuto del BIOS della macchina, che esegue una serie di operazioni di test e di impostazione iniziale dell'hardware. Una volta terminato correttamente, esso tenta di eseguire il boot del sistema operativo. Per far questo esso carica in memoria il primo settore dell'hard disk, il cosiddetto MBR (Master Boot Record) ed esegue il codice in esso contenuto (Boot Loader). Nella maggior parte delle installazioni di Linux viene utilizzato un programma di nome LILO (LInux LOader), il quale si occupa di caricare in memoria l'immagine del kernel del sistema operativo e, una volta caricato, gli cede il controllo.

I messaggi emessi sullo schermo da LILO sono simili ai seguenti:

```
LILO
Loading linux.........
Uncompressing Linux... Ok, booting the kernel
```

I messaggi successivi sono generati dal kernel:

```
Linux version 2.2.5-14BOOT (root@profuso.com) (gcc version cgcs-2.91.66)
Detected 367503611 Hz processor.
Console: Colour VGA+ 80x25
```

Al termine di tutte le operazioni di boot l'utente riceverà finalmente l'invito ad eseguire il login sulla macchina. Se username e password inseriti saranno corretti, gli verrà messa a disposizione una shell (l'interprete di comandi di Linux) con cui impartire comandi al sistema e lanciare programmi.

```
Welcome to Linux 2.0.35.

freddy login: beppe
Password:
Last login: Fri Oct 8 18:01:16 from tu_isdn
You have mail.
$
```

Al termine è necessario uscire dalla sessione di lavoro con "exit". Quando si spegne la macchina è necessario inoltre fare attenzione a non farlo brutalmente mediante il pulsante di accensione, in quanto ciò potrebbe danneggiare il filesystem. Si deve invece eseguire l'apposita procedura, ovvero collegarsi al sistema con l'account di amministratore (root) e impartire uno dei comandi "halt" (ferma il sistema) o "reboot" (esegue un reset). In alternativa è possibile premere la sequenza di tasti CTRL-ALT-DEL e attendere il termine delle operazioni di chiusura.

Il boot loader LILO

Non essendo LILO un vero sistema operativo, ma un programmino di pochi byte in linguaggio macchina, esso non conosce nulla della struttura del filesystem di Linux, ma si limita a caricare in memoria una lista di settori del disco generata al momento della sua installazione. Per questo motivo una volta installato è bene non toccare il file zImage contenente l'immagine del kernel (il nucleo di Linux). Poiché LILO per accedere al disco si appoggia al BIOS della macchina (al contrario di Linux che è un vero sistema operativo), esso soffre di alcune limitazioni. Ad esempio può avere problemi ad accedere a settori posti oltre al 1024simo cilindro del disco (è uno dei motivi per cui altri sistemi operativi si installano solo nella prima partizione). Ciò potrebbe dare dei problemi utilizzando dischi di grandi dimensioni con BIOS vecchi che non supportino le funzioni di remapping della geometria del disco (es: LBA). Nella documentazione di Linux esiste un apposito HOWTO che spiega come comportarsi in questi casi (Large-Disks-HOWTO).

È possibile modificare la configurazione di LILO, editando il file /etc/lilo.conf e rilanciando il comando lilo, in modo che per default carichi una determinata immagine del kernel di Linux, passandole dei valori, ad esempio quale deve essere il filesystem da utilizzare come directory di root del sistema. In alternativa è possibile fare in modo che alla partenza LILO si fermi per un certo tempo in attesa di comandi da tastiera da parte dell'operatore e, passato questo timeout, continui il boot con parametri di default.

Al prompt di LILO l'utente può infatti scegliere quale immagine caricare fra quelle preconfigurate. In questo modo è possibile avere su partizioni diverse del disco più versioni di Linux, oppure Linux e Windows, e scegliere al boot quale ca-

ricare. È eventualmente possibile specificare dei parametri da passare al kernel diversi da quelli di default.

Ad esempio i comandi seguenti:

```
LILO: newlinux single vga=EX-
TENDED
```

dicono a LILO di caricare l'immagine del kernel di nome "newlinux" in modalità single user (utile per eseguire lavori di manutenzione senza il problema che altri utenti possano accedere alla macchina da rete o da terminale) ed utilizzando la massima risoluzione disponibile in modo testo nella propria scheda video (ad esempio 160x24 invece del classico 80x24).

Questa possibilità è utile anche nel caso si debba far partire la macchina usando un dischetto di ripristino (rescue), in quanto è possibile specificare su quale partizione risiede la directory di root del sistema

```
LILO: rescue root=/dev/hdd3
```

I dischi e CDROM IDE (e in generale ogni periferica hardware) sono accessibili in Linux mediante dei file speciali presenti nella directory /dev. Nel caso delle periferiche IDE essi sono del tipo /dev/hd{disco}{partizione} (con disco=a,b,c,d e partizione=1,2,3,...). Il file /dev/hda2 permette perciò di accedere alla seconda partizione del primo disco della macchina (ovvero quello configurato come master nel controller IDE primario), mentre /dev/hdd3 indica la terza partizione del quarto disco (configurato come slave nel controller IDE secondario). Analogamente i dischi SCSI sono rappresentati dai file speciali /dev/sd{disco}{partizione} (esempio: /dev/sdc2).

Corel Linux

di Giuseppe Zanetti

Corel è stata fra le prime grandi aziende di software a credere nel mercato dell'Open Source, offrendo un porting su Linux di un proprio software di punta, Word Perfect (la versione gratuita per Linux ha ottenuto 1.5 milioni di download), e sviluppando un prodotto hardware specificamente pensato per funzionare con questo sistema operativo (ovvero il network computer Newtwinder, a cui abbiamo dedicato un articolo in questa rubrica).

In seguito a queste esperienze, l'azienda canadese ha deciso di scendere in campo con una propria distribuzione, Corel Linux, basata su Debian e specificamente pensata per il mercato "desktop". La filosofia che prevale in Corel Linux è infatti quella della semplicità, a partire dall'installazione, che avviene eseguendo direttamente il boot da CD-ROM ed utilizzando una semplice interfaccia grafica per impostare le diverse configurazioni.

L'ambiente di desktop utilizzato è il KDE, che, dopo aver risolto alcuni problemi "filosofici" sulla licenza d'uso delle librerie su cui è basato, sta ormai diventando il nuovo standard in ambiente Linux. Ad esso Corel ha aggiunto un file manager, un sistema centralizzato per l'amministrazione del sistema ed i client per e-mail e browsing, ottenendo un ambiente di lavoro molto semplice ed immediato da utilizzare. Il file manager, che funge anche da browser Internet, permette di accedere facilmente a file locali e a risorse condivise in rete secondo i protocolli NFS, SMB e Appletalk. Ciò permette una semplice integrazione di Linux Corel in reti UNIX o Windows già esistenti.

Corel Package manager permette di installare o aggiornare nuove applicazioni direttamente da Internet, scegliendole dal sito del produttore mediante una semplice interfaccia Web. È inoltre possibile caricare pacchetti aggiuntivi nei formati .rpm o .deb.

La versione beta di Linux Corel, da noi provata, soffre ovviamente di alcuni problemi di gioventù, specialmente nella fase di installazione. Essa tuttavia mantiene abbastanza bene le promesse e c'è da credere che la versione definitiva, prevista per la fine del 1999, avrà un ruolo importante nella diffusione di Linux, specialmente perché l'azienda di Ottawa intende rendere disponibile per la propria distribuzione una completa suite di Office, con funzionalità identiche a quelle offerte nella versione per Windows e composta da Wordperfect, Quattro Pro e Corel Presentation.

Essendo Corel Linux basato su software Open, le modifiche apportate da Corel a programmi già esistenti verranno rese disponibili secondo la medesima licenza. Sempre come Open Software, ma con una licenza ancora da definire, verrà reso disponibile anche il software aggiunto da Corel. La suite di Office verrà invece licenziata come software commerciale.

Per ulteriori informazioni: http://linux.corel.com/

I device driver modulari

Una volta caricato in memoria e lanciato il kernel, esso si occupa di inizializzare le diverse periferiche presenti sulla macchina. Le principali vengono riconosciute e gestite dai driver compilati all'interno del kernel. Non è tuttavia pensabile avere dentro al kernel i driver per ogni possibile scheda, in quanto le dimensioni dello stesso diventerebbero troppo grandi. Per questo motivo vi è la possibilità di caricare a run-time i driver necessari. In questo modo è possibile avere dei kernel modulari che contengano il minimo necessario (come il driver per i dischi IDE) e, una volta che il sistema è partito, aggiungere i driver che servono per gestire il rimanente hardware presente sulla macchina.

Generalmente il compito di caricare run-time i device driver viene svolto da uno degli script di inizializzazione della macchina in /etc/rc.d, che analizzeremo nel seguito dell'articolo.

Linux offre ulteriori sofisticazioni, come la possibilità di caricare un driver (o funzioni aggiuntive del sistema) in modo automatico solo nel momento del bisogno e di scaricarlo quando non è più necessario, con una notevole efficienza nell'utilizzo della memoria. In questo modo è possibile ad esempio caricare automaticamente il driver per lo ZIP solo quando si tenta di utilizzare tale periferica, oppure il supporto per il PPP solamente quando ci si connette ad Internet. Questa funzione si ottiene utilizzando il programma kerneld oppure, nella versione 2.2 di Linux, direttamente configurando in modo opportuno il kernel.

Ogni driver al momento del caricamento è in grado di testare se sulla macchina è presente l'hardware per cui esso è stato scritto. Ciò è molto comodo poiché, ad esempio nel programma di installazione di una distribuzione, permette di riconoscere automaticamente la configurazione della macchina (una specie di Plug&Play fatto per tentativi), ma alle volte può provocare problemi al sistema, in quanto per verificare se esistono certi tipi di hardware è necessario andare a scrivere dei valori di test negli indirizzi in cui si pensa siano mappati e ciò potrebbe non essere saggio

nel caso in quella posizione ci fosse una scheda diversa da quella prevista.

Per ovviare a questo inconveniente è possibile passare ai driver dei suggerimenti su dove sono mappate le schede. Ad esempio, passando i parametri io=0x300 e irq=5 al driver che gestisce la ethernet lo si istruisce in modo che eviti di andare a cercare la scheda in tutta una serie di indirizzi standard per quell'hardware.

Utilizzando un driver caricato a run-time, tali informazioni possono essere passate direttamente nella linea di comando del programma insmod che serve per caricarlo:

```
insmod /lib/modules/2.2.8/net/ne.o io=0x300 irq=5
```

in alternativa possono essere inserite in /etc/conf.modules, da dove insmod se le carica direttamente.

Il programma supervisore init

Per quanto riguarda il kernel, una volta che esso ha inizializzato tutte le proprie componenti e i driver compilati al suo interno, non c'è più molto lavoro di inizializzazione da fare. Esso termina infatti montando il filesystem di root e, cedendo il controllo al programma /sbin/init. Questo è un programma molto speciale, in quanto spetta ad esso il compito di fare partire e gestire tutti gli altri programmi del sistema. In un certo senso è il padre (o, più correttamente, il progenitore) di tutti gli altri processi (con processo si intende un programma una volta che è in esecuzione).

Per capire come funziona il meccanismo è necessario introdurre il concetto dei run level: questi indicano diversi stati di funzionamento del sistema, in cui sono in esecuzione determinati processi. In un sistema Linux sono possibili otto run level da 0 a 6 e S, ad ognuno dei quali è possibile associare, mediante /etc/inittab, una particolare lista di programmi da mandare in esecuzione.

Il run level in cui far partire il sistema può essere scelto dall'utente al momento del prompt di LILO (si veda l'esempio fatto in precedenza)i. In questo modo è possibile prepararsi configurazioni. Ad esempio alcune distribuzioni di Linux sono preconfigurate in modo che se si sceglie il run level di default (di solito 3) la macchina parte in modalità a carattere, mentre scegliendo il run level 6 essa

parte in grafica.

Il run level S è particolare, in quanto corrisponde alla modalità "single user", pensata per la manutenzione del sistema. Allo stato di single user di solito corrisponde una configurazione minima della macchina, in cui non vengono abilitati i collegamenti da terminale, non viene fatto partire il software di rete e non vengono montati altri dischi oltre a quello dove risiede la root. Ciò permette, oltre che di lavorare sulla macchina senza preoccuparsi di quello che fanno gli utenti (non è bello che un utente vada a scrivere nel disco mentre stiamo facendo un check forzato dello stesso), anche di avere qualche speranza che essa parta anche nel caso di errori di configurazione o di un filesystem rovinato.

Anche i run level 0 e 6 dovrebbero essere evitati per i propri utilizzi, in quanto sono quelli in cui entra il sistema quando si spegne la macchina mediante il comando reboot (o halt). In essi vengono lanciati gli script che permettono uno "shutdown" pulito: vengono smontati eventuali dischi di rete, scaricati i buffer del filesystem e terminati "con grazia" i processi attivi dando loro il tempo di chiudere correttamente i file aperti.

Appena il kernel cede il controllo ad init, esso cerca in /etc/inittab una linea di tipo initdefault — si veda il manuale online inittab(5) — e da questa ricava l'informazione su quale deve essere il run level di default. Se tale linea non è presente, il run level viene richiesto all'utente sulla console del sistema.

Se è stato scelto un run level diverso da S, init esegue i comandi in /etc/inittab associati alle etichette boot e bootwait, a cui di solito sono associati gli script di inizializzazione del sistema (ne parleremo in seguito). Dopo di queste esegue tutte le altre linee associate a quel determinato run level.

La tabella che segue mostra come gestisce i run level la distribuzione Red Hat:

0	halt (non usare per i propri scopi)
1	single user completo
2	multiuser, senza NFS (analogo al 3, se non avete la rete)
3	multiuser completo (run level di default)
4	non usato
5	sistema in modalità grafica X11
6	reboot (non usare per i propri scopi)
S	Single User mode

Ogni riga di /etc/inittab ha il seguente formato:

```
etichetta:lista di runlevel:tipo:comando
```

Eventuali linee contenenti commenti devono iniziare col simbolo #.

La riga di tipo initdefault indica ad init quale run level considerare di default (in questo caso il 3).

```
id:3:initdefault:
```

La riga che segue, di tipo sysinit, viene eseguita una sola volta alla partenza di init. Essa richiama lo script /etc/rc.d/rc.sysinit, che nella Red Hat si occupa di inizializzare alcuni aspetti importanti, come il nome della macchina e di caricare i driver.

```
si::sysinit:/etc/rc.d/rc.sysinit
```

Nel momento dell'ingresso del sistema in un determinato run level viene poi eseguito un diverso programma (nel caso della Red Hat si richiama lo stesso script con un parametro diverso e poi esso si occupa di compiere le scelte opportune in base al valore passato). Init attende (wait) che il programma sia terminato prima di continuare.

```
l0:0:wait:/etc/rc.d/rc 0
l1:1:wait:/etc/rc.d/rc 1
l2:2:wait:/etc/rc.d/rc 2
l3:3:wait:/etc/rc.d/rc 3
l4:4:wait:/etc/rc.d/rc 4
l5:5:wait:/etc/rc.d/rc 5
l6:6:wait:/etc/rc.d/rc 6
```

Il processo update, che si occupa di tenere sincronizzato il filesystem, ovvero di scaricare su disco i buffer del disco presenti in memoria, deve essere lanciato qualunque sia il run level, ma solo una volta (once).

```
ud::once:/sbin/update
```

La seguente linea intercetta la pressione dei tasti CTRL-ALT-DEL ed esegue con uno shutdown pulito del sistema.

```
ca::ctrlaltdel:/sbin/shutdown -t3 -r now
```

Molti gruppi di continuità (UPS) sono in grado di segnalare ad init, mediante un segnale nella porta seriale, se sta mancando la tensione di alimentazione, in modo che esso possa eseguire uno

shutdown pulito. Se la tensione ritorna prima che la macchina sia spenta, lo shutdown viene interrotto.

```
pf::powerfail:/sbin/shutdown -f
-h +2 "Power Failure; System
Shutting Down"
pr:12345:powerokwait:/sbin/shut
down -c "Power Restored; Shutdown
Cancelled"
```

Le seguenti linee, che vengono eseguite quando il sistema si trova in uno dei run level da 2 a 5, sono molto interessanti, in quanto si occupano di richiedere il login sulle diverse console virtuali di Linux (a cui si accede con ALT+F1, ALT+F2, ...).

Il programma che sovrintende a tale funzione è /sbin/getty, che, una volta ottenuto lo username dell'utente, richiama /bin/login, che a sua volta richiede la password. Se la password inserita è corretta, login cede il posto ad una shell, che permette all'utente di lavorare. Quando l'utente termina la propria sessione di lavoro, mediante il comando con exit, init si preoccupa di rieseguire (respawn) una nuova copia di getty sulla console virtuale che si è liberata.

La modalità respawn può essere utilizzata anche qualora si voglia essere sicuri che sia attiva sempre almeno una copia di un determinato programma (ad esempio un server scritto non troppo bene che abbia la tendenza di morire ogni tanto).

```
t1:2345:respawn:/sbin/getty tty1
t2:2345:respawn:/sbin/getty tty2
t3:2345:respawn:/sbin/getty tty3
t4:2345:respawn:/sbin/getty tty4
t5:2345:respawn:/sbin/getty tty5
t6:2345:respawn:/sbin/getty tty6
```

Similmente è possibile gestire utenti che si collegano al sistema mediante un terminale o un modem connesso ad una delle porte seriali, che in Linux sono accessibili mediante i file speciali /dev/ttyS{numero}. In questo caso si utilizza una versione di getty appositamente pensata per i modem. Esistono infatti diverse versioni di questo programma, alcune in grado di gestire caratteristiche particolari come la voce (vgetty, mgetty-voice) e i fax (mgetty-sendfax).

```
m1:2345:respawn:/sbin/mgetty -s
19200 /dev/ttyS1
```

Termina il file /etc/inittab del nostro esempio la linea che istruisce init ad eseguire il programma che richiede il login dell'utente in modo grafico quando il sistema è nel run level 5:

```
x:5:respawn:/etc/X11/prefdm -
nodaemon
```

Per aggiungere righe ad /etc/inittab è sufficiente editare il file mediante un qualunque editor di testo. Le modifiche non vengono tuttavia rese immediatamente effettive, ma è necessario far rileggere il file ad init. Ciò può essere fatto mediante il comando "init q".

I file di inizializzazione in /etc/rc.d

Il modo di inizializzare la macchina varia molto con la distribuzione che si sta utilizzando, ad esempio nella Slackware alla partenza init esegue lo script /etc/rc.d/rc.S, in cui vengono caricati i vari driver e fatti partire i programmi che devono sempre girare in sottofondo nella macchina (daemon). Dopodiché il sistema entra nel normale run level di funzionamento, che in questa distribuzione è il 5. Indipendentemente dal run level scelto, le restanti inizializzazioni vengono eseguite mediante un unico script /etc/rc.d/rc.M.

Le procedure di inizializzazione del sistema sono dei semplici shell script, ovvero dei programmi scritti nel linguaggio della shell di UNIX. Ciò permette di aggiungerne di propri o di fare modifiche a quelli esistenti molto velocemente anche in macchine in cui non sia presente il compilatore per il linguaggio C. Vedremo, parlando della shell, come essa offra all'utente un linguaggio per realizzare script potente ed allo stesso tempo molto semplice da utilizzare.

Red Hat utilizza invece un sistema più modulare, derivato da UNIX System V, in cui gli script di inizializzazioni corrispondenti ad ogni run level sono contenuti in una apposita sottodirectory, ad esempio /etc/rc.d/rc3.d per il run level 3.

In realtà gli script si trovano fisicamente nella directory /etc/rc.d/init.d e vengono fatti vedere nelle varie sottodirectory mediante dei link simbolici. Un link simbolico è una caratteristica dei sistemi UNIX che permette di far vedere lo stesso file fisico con nomi diversi e possibilmente in directory diverse. Ad esempio il file /etc/rc.d/init.d/httpd viene fatto vedere nelle directory corrispondenti ai run level 3 e 4 come /etc/rc.d/rc3.d/S85httpd e come /etc/rc.d/rc4.d/S85httpd. Modificando il contenuto del file vero o di uno dei suoi collegamenti, vengono modificati anche tutti gli altri.

All'ingresso del sistema in un determinato run level ognuno degli script presenti nella directory corrispondente viene lanciato, richiamandolo come:

```
S85httpd start
```

I diversi script hanno un nome del tipo Snnxxxx (oppure Knnxxxx) e vengono lanciati in ordine crescente del valore nn (S45pcmcia viene lanciato prima di S80sendmail e di S85httpd).

Gli script col nome nel formato Knnxxxx servono per terminare servizi non necessari in quel run level. Essi vengono lanciati col parametro stop:

```
K55routed stop
```

In questo modo è possibile terminare accuratamente tutti i programmi e i servizi precedentemente fatti partire. Ciò avviene non solo quando si spegne la macchina (entrando nel run level 0), ma anche quando l'utente richiede, con "init nuovorunlevel" (esempio: init 6), di cambiare livello di funzionamento.

Le operazioni di start e stop di un servizio possono essere eseguite anche manualmente. Ad esempio per rendere attive eventuali modifiche al file di configurazione del Web server Apache è necessario farlo ripartire. È possibile far ciò terminando il processo del server in esecuzione con

```
/etc/rc.d/rc3.d/S85httpd stop
```

e facendolo ripartire con

```
/etc/rc.d/rc3.d/S85httpd start
```

Volendo fare tutto in un'unica operazione si può utilizzare l'opzione restart:

```
/etc/rc.d/rc3.d/S85httpd re-
start
```

Conclusioni

Nelle prossime puntate analizzeremo due aspetti molto importanti di Linux, ovvero come funzionano il filesystem e l'interprete di comandi. Vedremo come, grazie alle caratteristiche uniche di Linux, sia possibile risolvere facilmente problemi anche complessi mediante semplici script. MC

Se credi che la leucemia resterà un male inguaribile devi farci un favore.

Piantarla.

Dal 10 al 12 dicembre
nella tua città trovi
le Stelle di Natale
per sostenere la ricerca
e la cura delle leucemie.

ASSOCIAZIONE ITALIANA
CONTRO LE LEUCEMIE
ONLUS

Sede Nazionale Via Ravenna, 34 - 00161 Roma

c/c Postale n. 46716007

www.ail.it

Se vuoi sapere quali sono le piazze
con le Stelle dell'Ail
chiama il numero 06/4402696