Lilo mini-Howto

Cameron Spitzer (`cls@truffula.sj.ca.us`), Alessandro Rubini (`rubini@linux.it`). v2.03, 19 Agosto 1998

LILO è il **Li**nux **Lo**ader più utilizzato per la versione di Linux che gira su processori di famiglia x86; poiché non apprezzo le maiuscole, in questo documento il programma sarà chiamato Lilo invece che LILO. Questo documento descrive alcune tipiche installazioni di Lilo. Il suo ruolo è quello di integrare la guida dell'utente di Lilo con informazioni più pratiche; credo che questi esempi siano abbastanza informativi anche se la vostra situazione non è molto simile alla mia. Spero che questo documento vi eviti di avere problemi. Siccome la documentazione di Lilo è molto ben fatta, chi è interessato ai dettagli è invitato a leggere in /usr/doc/lilo*. Questo documento è stato tradotto da Alessandro Rubini, nel marzo 1998.

Indice

1 Introduzione

Nonostante la documentazione distribuita con i sorgenti di Lilo sia ad ampio spettro (mi riferisco qui ai file che trovate anche sotto `/usr/doc/lilo`-versione sulla vostra macchina Linux), la maggior parte degli utenti Linux trovano qualche difficoltà durante la costruzione del proprio file `/etc/lilo.conf`. Questo documento è volto ad aiutare queste persone fornendo l'informazione minimale che serve per scrivere un file di configurazione e fornendo cinque installazioni esemplificative:

- Il primo esempio è la classica installazione "Linux e un altro sistema".

- L'esempio seguente mostra come si possa installare Linux su un disco collegato come /dev/hdc facendo in modo che tutto funzioni correttamente quando il disco viene collegato come /dev/hda e viene usato per il boot. Questo tipo di configurazione è quella che serve usare quando si installa un nuovo sistema Linux dall'interno del proprio sistema già funzionante.
- Il terzo caso presentato mostra come accendere un sistema Linux la cui partizione root non viene vista dal BIOS della macchina. Questa sezione spiega anche come avviare il sistema da un disco SCSI se il BIOS è abbastanza recente.
- Il successivo esempio viene usato per accedere a dischi molto grossi, che né il BIOS né il DOS possono vedere con facilità (questo esempio è ormai obsoleto, in quanto tutti i BIOS recenti riescono a gestire i dischi grossi).
- L'ultimo esempio mostra come recuperare un disco danneggiato, se il danno deriva dall'installazione di un altro sistema operativo.

Gli ultimi tre esempi sono stati forniti da Cameron, cls@truffula.sj.ca.us, che ha scritto le prime versioni di questo documento. Alessandro rubini@linux.it, il manutentore attuale, usa solo Linux sulle sue macchine e non è quindi in grado di controllare né di aggiornare tali esempi esempi. Sentitevi liberi di mandare il vostro contributo a questo documento, preferibilmente mandandomi un messaggio di posta elettronica.

2 Informazione di base e installazione tipica

Quando Lilo avvia il sistema, usa le chiamate del BIOS per caricare il kernel Linux dal disco (disco IDE, dischetto o altro). Per questo motivo, il kernel deve risiedere in un luogo che possa essere visto dal BIOS della macchina.

All'accensione Lilo non è in grado di leggere i dati del filesystem per cui ogni pathname che appare in /etc/lilo.conf viene risolto durante l'installazione di Lilo (quando il superutente invoca il comando */sbin/lilo*). Tale installazione è il momento in cui il programma costruisce le tabelle che elencano quali settori sono usati dai file coinvolti nel caricamento del sistema operativo. Una conseguenza di ciò è che tutti tali file devono risiedere in una partizione accessibile da parte del BIOS (siccome tali file di solito risiedono in /boot, è sufficente che il BIOS possa leggere la partizione root del vostro sistema).

Un'altra conseguenza dell'appoggiarsi sul BIOS è che occorre reinstallare il programma (cioè, occorre reinvocare */sbin/lilo*), ogni volta che si modifica la configurazione del programma. Ogni volta che si ricompila il kernel e si sovrascrive la precedente immagine, occorre reinstallare Lilo.

2.1 Dove devo installare Lilo?

All'interno di /etc/lilo.conf, la direttiva boot= dice a Lilo dove installare il loader primario, quello che viene eseguito dal BIOS all'accensione della macchina. Come regola generale, questo codice può essere installato nel "master boot record" (cioè in /dev/hda) oppure nella partizione root dell'installazione Linux (di solito /dev/hda1 o /dev/hda2).

Se il proprio disco contiene un altro sistema operativo oltre a Linux, conviene installare Lilo sulla partizione root invece che sul Master Boot Record. In questo caso occorre marcare la partizione come "bootable" (usando il comando "a" di *fdisk* o il comando "b" di *cfdisk*. Se il *master boot sector* non viene sovrascritto risulterà più facile la disinstallazione di Linux, se ne sorgerà il bisogno.

2.2 Come devo configurare i miei dischi IDE?

Personalmente nel mio BIOS preferisco non specificare né LBA né LARGE per i miei dischi; ma ricorate che io uso solo Linux. Questi due modi di accesso sono degli accrocchi orribili, inventati solamente come pezza a fronte di alcune deficienze progettuali del primi PC. Specificare i propri dischi come NORMAL vuol dire che il proprio kernel deve stare all'interno dei primi 1024 cilindri del disco, ma questo non è un problema se il disco è correttamente partizionato e se la partizione di root è piccola (questa è una buona norma in ogni caso, indipendentemente da Lilo).

Se il disco invece contiene già un altro sistema operativo, non si potranno modificare i settaggi del BIOS o il vecchio sistema non funzionerà più. Tutte le distribuzioni recenti di Linux, comunque, sono in grado di gestire dischi configurati come LBA o LARGE.

Se si ha più di un disco rigido e alcuni di essi sono usati solo da Linux senza far parte del processo di avvio del sistema, si può evitare di dire al BIOS che questi dischi esistono. Il sistema si avvierà più velocemente e Linux riconoscerà automaticamente e velocemente tutti i dischi. A me capita spesso di aggiungere e togliere dischi dal mio sistema, ma non cambio mai la configurazione del BIOS.

2.3 Come posso interagire durante il boot?

Quando appare il prompt di Lilo, premendo il tasto <Tab> si ottiene la lista di tutte le possibilità di boot offerte da Lilo. Se Lilo non è configurato per interagire con l'utente si può comunque renderlo interattivo premendo il tasto <Alt> o il tasto <Shift> prima che appaia il messaggio "LILO", e tenendo premuto tale tasto.

Se al prompt di Lilo si sceglie un'immagine di kernel Linux è possibile passare dei parametri di linea di comando al kernel stesso. Il kernel Linux riconosce molti parametri, tutti descritti nel "BootPrompt-HOWTO" by Paul Gortmaker, documento che non val la pena di replicare in questa sede. Alcuni di questi parametri, d'altra parte, secondo me sono particolarmente importanti e meritano di essere descritti:

- `root=`: si può dire al kernel di usare come partizione di root una partizione diversa da quella che appare nel file `lilo.conf`. Per esempio, io ho dedicato una piccola partizione a ospitare una installazione minimale di Linux; grazie a questa sono stato in grado di accendere il mio calcolatore anche dopo aver distrutto per sbaglio la partizione di root (questi danni comunque succedono solo se si smanetta con le strutture interne del kernel).
- `init=`: la versione 1.3.43 e tutte le successive possono ricevere dalla linea di comando il pathname di un programma da eseguire al posto di `/sbin/init`. Se il sistema si blocca durante il processo di boot a causa di un errore negli script di accensione è ancora possibile accedere al sistema specificando `init=/bin/sh` al prompt di Lilo. Dopo aver acceso il sistema in questo modo probabilmente occorrerà accedere ai propri dischi, che saranno inaccessibili; per questo occorre invocare "`mount -w -n -o remount /; mount -a`", e ricordarsi di chiamare "`umount -a`" prima di spegnere il calcolatore.
- Un numero: se si specifica un numero sulla linea di comando, il programma *init* selezionerà il "runlevel" specificato invece di quello predefinito (di solito il 3 o il 2, in base alla distribuzione che si usa). Per saperne di più consiglio di riferirsi alla documentazione di *init*, al file `/etc/inittab` e ai vari file che si trovano in `/etc/rc*.d`.

2.4 Come posso disinstallare Lilo?

Quando Lilo sovrascrive un settore di boot, una copia del settore viene salvata in `/boot/boot.`*xxyy*, dove *xxyy* sono i due numeri che rappresentano la periferica, scritti in esadecimale. Si possono vedere questi due numeri, chiamati "major number" e "minor number" chiamando "`ls -l /dev/`*device*". Per esempio, il primo settore del disco `/dev/hda` (che ha 3 e 0 come numeri di periferica) verrà salvato in `/boot/boot.0300`; se si installa lilo su `/dev/fd0` il file si chiamerà `/boot/boot.0200`, se si installa su `/dev/sdb3` (con major number 8 e minor 19) il file si chiamerà `/boot/boot.0813`. Si noti che Lilo non salva il primo settore del disco se il file corrispondente esiste già, quindi non occorre preoccuparsi quando si reinstalla Lilo (per esempio dopo aver ricompilato il kernel).

Se per caso occorre disinstallare Lilo (per esempio nella malaugurata ipotesi che si cancelli Linux dal disco) occorre semplicemente rimettere al suo posto il settore di boot originale. Se Lilo è installato in `/dev/hda`, basta fare "`dd if=/boot/boot.0300 of=/dev/hda bs=446 count=1`" (personalmente preferisco fare "`cat /boot/boot.0300 > /dev/hda`", ma questo comando non è sicuro, in quanto copia anche la tabella delle partizioni, che potrebbe essere stata modificata nel frattempo). Questo comando è molto più facile che dover far partire il DOS per chiamare "`fdisk /mbr`": permette di togliere Linux da un disco senza mai avviare alcun altro sistema operativo. Dopo aver rimosso Lilo dal disco occorre anche ricordarsi di far girare il comando *fdisk* di Linux per distruggere tutte le partizioni Linux, in quanto la versione DOS del comando non è in grado di cancellare partizioni non-dos.

Se Lilo è stato installato sulla partizione root del sistema Linux, (per esempio `/dev/hda2`), invece, non occorre fare niente di particolare per disinstallare Lilo. In questo caso basta far andare il programma *fdisk* di Linux per rimuovere le partizioni Linux dalla tabella delle partizioni. Bisogna anche ricordarsi di marcare la partizione DOS come partizione "bootable" ("attiva" in gergo DOS).

3 La configurazione più semplice

La maggior parte delle installazioni di Lilo utilizzano un file di configurazione come il seguente:

```
boot = /dev/hda   # oppure la partizione root
delay = 10        # attesa, in decimi di secondo, per poter interagire
vga = 0           # opzionale. Si usi "vga=1" per un modo testo 80x50
#linear           # si provi "linear" se ci sono problemi di geometria

image = /boot/vmlinux  # il proprio file zImage
  root = /dev/hda1     # la partizione root
  label = Linux        # o un altro bel nome
  read-only            # root va montato in sola lettura

other = /dev/hda4   # la partizione dos, se esiste
  table = /dev/hda  # la tabella delle partizioni attuale
  label = dos       # o un altro stupido nome
```

Si possono specificare diverse sezioni "image" e "other" se serve. È in effetti abbastanza comune configurare nel proprio *lilo.conf* diverse immagini del kernel, almeno per chi cerca di rimanere aggiornato con le versioni di sviluppo del sistema.

3.1 Come gestire kernel grossi

Se la compilazione di una "zImage" genera un'immagine più grande di mezzo megabyte (come capita spesso con i kernel 2.1), bisognerebbe creare una "bzImage" (big zImage). Si provi a tal fine "`make bzImage`". Per avviare un'immagine del kernel di questo tipo non serve far niente di speciale, ma serve avere la versione 18 di Lilo, o una successiva. Se la vostra installazione è molto datata, occorrerà aggiornare il pacchetto Lilo.

3.2 Altre fonti di informazione

Oltre ai documenti forniti con Lilo, ci sono un certo numero di mini-howto che possono essere utili per risolvere i proprio problemi. Si chiamano tutti "Linux+*pincoOS*", per vari valori di *pincoOS*; questi documenti trattano la coesistenaza di Linux e altri sistemi operativi. Inoltre, "Multiboot-with-LILO" descrive come le varie versioni di Windows possono coesistere con Linux.

4 Installare su `hdc` per avviare come `hda` e uso di `bios=`

Lilo permette di creare la mappa dei settori da un disco e allo stesso tempo di dire al BIOS di leggere i settori da un altro disco. Per esempio a me capita spesso di installare Linux su un disco collegato come `hdc` (disco principale del secondo cavo IDE), e usare poi tale disco come unico disco di un nuovo calcolatore (`hda`). Ho fatto una copia del dischetto di installazione in una piccola partizione, in modo da installare su `hdc` facendo *chroot* da un terminale virtuale, senza dover smettere di usare il calcolatore.

Il file *lilo.conf* che uso in questo caso è fatto così:

```
# Questo file deve essere usato da un sistema che giri su /dev/hdc
boot = /dev/hdc   # sovrascrivi il settore di boot di hdc
disk = /dev/hdc   # e specifica come accedere ad hdc:
   bios = 0x80    #  il BIOS lo vedra' come primo disco
delay = 0
vga = 0

image = /boot/vmlinux      # all'interno di  /dev/hdc1
  root = /dev/hda1         # che all'accensione sara' hda1
  label = Linux
  read-only
```

Questo file di configurazione deve venir usato da un Lilo **che gira su /dev/hdc1**. Le mappe di Lilo che vengono scritte nel settore di boot (`/dev/hdc`) devono riferirsi ai file che stanno in `/boot` (attualmente montato sotto /dev/hdc1); questi file verranno letti da hda nel momento in cui il disco sarà avviato come sistema stand-alone.

Personalmente chiamo questo file `/mnt/etc/lilo.conf.hdc` (poiché `/mnt` è dove attacco il nuovo disco durante l'installazione). L'installazione di Lilo avviene poi con questo comando: "`cd /mnt; chroot . sbin/lilo -C /etc/lilo.conf.hdc`". Si vedano la pagine del manuale di `chroot` se questo comando risulta di difficile comprensione.

La direttiva "`bios=`" nel file `lilo.conf` serve a dire a Lilo che idea ha il BIOS delle proprie periferiche. Le chiamate al BIOS identificano i dischetti e i dischi rigidi con un numero: 0x00 and 0x01 idenficano i floppy, 0x80 e numeri successivi identificano i dischi rigidi (e i vecchi BIOS possono solo accedere a due di essi). Il significato del parametro "`bios = 0x80` usato nell'esempio precedente significa quindi: "usa 0x80 nelle chiamate al BIOS per il disco `/dev/hdc`".

Questa direttiva di Lilo può essere utile anche in altre situazioni, per esempio quando il BIOS è in grado di avviare il sistema da un disco SCSI invece che da uno IDE. Quando il sistema contiene sia dischi IDE che SCSI, Lilo non può sapere in partenza se il numero 0x80 si riferirà all'uno o all'altro, poiché l'utente può dire al BIOS come comportarsi (tramite i menu di configurazione), mentre il BIOS stesso non può essere contattato durante il funzionamento di Linux.

Per default, Lilo assume che i dischi IDE prendano i primi numeri a disposizione, ma questa assunzione può essere modificata mettendo le seguenti linee nel proprio `/etc/lilo.conf`:

```
disk = /dev/sda
  bios = 0x80
```

5 Utilizzo di Lilo quando il BIOS non può accedere alla partizione di root.

Ho due dischi IDE e un disco SCSI. Il disco SCSI non viene visto dal BIOS. Siccome Lilo usa le chiamate al BIOS non può accedere ai dischi che non vengono visti dal BIOS. Il mio stupido AMI-BIOS può solo avviare il sistema da "A:" o da "C:", e la mia partizione di root sta sul disco SCSI.

La soluzione in questo caso consiste nel mettere il kernel, la mappa dei settori e il secondo stadio di caricamento di Lilo in una partizione Linux nel primo disco IDE. Si noti che non serve mettere il kernel nella partizione root.

La seconda partizione del mio primo disco IDE (`/dev/hda2`, la partizione Linux usata per avviare il sistema) è montata in `/u2`. Questo è il file `/etc/lilo.conf` che ho usato.

```
#  Installa LILO su settore di boot del primo disco IDE
#
boot = /dev/hda
#  /sbin/lilo (l'installatore) copia il loader di Lilo da
#  questo file sopra al settore di boot.
install = /u2/etc/lilo/boot.b
#  Ho scritto un messaggio prolisso in questo file.
message = /u2/etc/lilo/message
#  Il comando "lilo" scrive in questo file la mappa dei settori
#  di disco occupati dal kernel
map = /u2/etc/lilo/map
compact
prompt
#  Aspetta diecisecondi, poi attiva 1.2.1 per default.
timeout = 100
#  Il kernel viene salvato dive il BIOS può leggerlo in questo modo:
#      cp -p /usr/src/linux/arch/i386/boot/zImage /u2/z1.2.1
image = /u2/z1.2.1
        label = 1.2.1
#  Lilo dice al kernel di montare la prima partizione SCSI come root.
#  Non serve che il BIOS sia in grado di accedervi.
        root = /dev/sda1
#  La partizione è montata in sola lettura
        read-only
#  Ho tenuto un vecchio kernel Slackware da usare se quello appena
#  ricompilato non funziona. In effetti una volta questo mi è servito
image = /u2/z1.0.9
        label = 1.0.9
        root = /dev/sda1
        read-only
#  La mia partizione DR-DOS 6
other = /dev/hda1
        loader=/u2/etc/lilo/chain.b
```

```
        label = dos
        alias = m
```

6 Come accedere a dischi grossi che non sono visti dal BIOS

Il sistema che ho in ufficio ha un disco IDE da 1GB. Il BIOS può solo accedere ai primi 504MB del disco (con MB intendo 2^10 byte, non 10^6). Perciò ho messo il sistema DOS su una partizione da 350MB in `/dev/hda1` e la partizione root di Linux in 120MB come `/dev/hda2`.

Il DOS non è stato in grado di installarsi correttamente quando il disco era nuovo. La versione 7 del DOS Novell aveva lo stesso problema. Fortunatamente il servizio "opzioni di IBM" ha dimenticato di mettere il dischetto "OnTrack" nella scatola del disco. Il disco sarebbe dovuto arrivare con un prodotto chiamato "OnTrack Disk Manager". Se avete solo il DOS immagino che lo abbiate usato.

Allora ho fatto una partizione con il programma *fdisk* di Linux. Il DOS 6.2 ha rifiutato di installarsi su `/dev/hda1` dicendo qualcosa come "questa versione di MS-DOS è per le nuove installazioni; questo computer ha già il DOS installato quindi serve una versione 'aggiornamento' al posto di questa". In effetti, il disco era nuovo di zecca.

Che schifo! Allora ho usato ancora *fdisk* di Linux e ho cancellato la partizione 1 dalla tabella. Questo ha reso felice il mio DOS 6.2 che ha ricreato la partizione da me rimossa per potersi installare. MS-DOS 6.2 ha scritto il suo settore di boot sul disco ma non è riuscito ad avviarsi.

Per fortuna avevo un kernel Slackware in un floppy creato dal programma di installazione della mia Slackware, per cui ho acceso Linux e installato Lilo sopra al settore di boot difettoso del DOS. Questo funziona. Ecco il file `/etc/lilo.conf` che ho usato:

```
boot = /dev/hda
map = /lilo-map
delay = 100
ramdisk = 0             # Disattiva il ram-disk
timeout = 100
prompt
disk = /dev/hda         # Il BIOS vede solo 500MB di questo disco
   bios = 0x80          #   che è il primo disco IDE
   sectors = 63         #   e ha questi parametri (dalla
   heads = 16           #   documentazione del disco stesso)
   cylinders = 2100
image = /vmlinuz
  append = "hd=2100,16,63"
  root = /dev/hda2
  label = linux
  read-only
  vga = extended
other = /dev/hda1
  label = msdos
  table = /dev/hda
  loader = /boot/chain.b
```

Dopo aver installato questi sistemi ho verificato che la partizione contenente i file `zImage`, `boot.b`, `map`, `chain.b` e `message` potesse usare un filesystem MSDOS, purché che non sia compresso con stacker o doublespace. In questo caso avrei potuto dare tutti i 500MB alla partizione DOS.

Ho anche imparato successivamente che “OnTrack” avrebbe scritto la tabella delle partizioni spostata di alcune decine di byte all’interno del disco invece che all’inizio, e ho saputo che è possibile modificare il driver IDE di Linux per gestire questa situazione, ma l’installazione in questo caso sarebbe stata impossibile con il kernel slackware precompilato *[NDT: ecco un altro caso in cui i produttori di hardware fanno delle porcate immonde per sistemare le mancanze di Microsoft]*). Alla fine la IBM mi ha mandato il dischetto “OnTrack”; ho contattato il loro supporto tecnico e mi hanno detto che Linux è bacato perché non usa il BIOS. Ho buttato via il dischetto.

7 Accensione da un dischetto di salvataggio

Successivamente ho installato Windows 95 sul calcolatore in ufficio. Ovviamente ha bruciato il mio bel settore di boot di Lilo, ma non ha toccato la mia partizione Linux. Siccome il BIOS impiega molto tempo a caricare il kernel dal dischetto ho creato preventivamente un dischetto con Lilo installato in modo da poter caricare il kernel dal disco IDE. Il floppy è stato creato così:

```
fdformat /dev/fd0H1440      # scrivi le traccie su disco nuovo
mkfs -t minix /dev/fd0 1440 # crea il filesystem di tipo minix
mount /dev/fd0 /mnt         # aggancialo nella directory convenzionale
cp -p /boot/chain.b /mnt    # copia il loader sul dischetto
lilo -C /etc/lilo.flop      # installa lilo e la mappa sul floppy
umount /mnt
```

Si noti che il dischetto deve essere montato quando si invoca *lilo*, in modo che Lilo possa scrivere il file con la mappa dei settori del kernel.

Questo è /etc/lilo.flop, ed è molto simile all’ultimo che vi ho fatto vedere:

```
#  Crea un dischetto che legge i kernel dal disco fisso
boot = /dev/fd0
map = /mnt/lilo-map
delay = 100
ramdisk = 0
timeout = 100
prompt
disk = /dev/hda     # 1 GB IDE, ma il BIOS vede solo i primi 500 MB.
   bios=0x80
   sectors = 63
   heads = 16
   cylinders = 2100
image = /vmlinuz
  append = "hd=2100,16,63"
  root = /dev/hda2
  label = linux
  read-only
  vga = extended
other = /dev/hda1
  label = msdos
  table = /dev/hda
  loader = /mnt/chain.b
```

Infine, mi è successo di aver bisogno del DOS 6.2 in ufficio, ma non volevo toccare il primo disco IDE. Ho aggiunto un controller e un disco SCSI e ho creato un filesystem DOS su questo disco con il comando *mkdosfs* di Linux. Questo disco viene visto come "D:" da Windows 95. Naturalmente il DOS non vuole partire dal disco D, ma questo non è un problema quando si ha Lilo. Ho aggiunto queste righe al file `lilo.conf` del penultimo esempio:

```
other = /dev/sda1
  label = d6.2
  table = /dev/sda
  loader = /boot/any_d.b
```

Con questa modifica il DOS 6.2 parte e crede di essere sul disco C mentre Windows 95 si trova su D *[NDT: demenziale!]*.