Lilo Mini-HOWTO

Cameron Spitzer (cls@truffula.sj.ca.us), Alessandro Rubini (rubini@linux.it) Diterjemahkan oleh Yaya Heryadi, yheryadi@necnusa.co.id
v 2.02, 16 August 1998 Terj. 26 January 1999

Lilo merupakan suatu Linux Loader yang paling banyak digunakan pada Linux versi x86; Disini penulis lebih suka menyebutnya sebagai Lilo daripada LILO karena penulis kurang begitu menyukai penulisan dengan huruf besar . Tulisan ini akan menjelaskan beberapa instalasi Lilo yang umum. Tulisan ini dibuat untuk menjadi sebuah pelengkap bagi Lilo User's Guide. Penulis berpendapat bahwa contoh-contoh yang telah diberikan didalam tulisan ini telah cukup informatif meskipun setup yang Anda miliki tidak mirip sekali dengan setup yang penulis miliki. Harapan penulis adalah semoga contoh-contoh tadi dapat menghindarkan Anda dari kesulitan. Mengingat dokumentasi yang dimiliki Lilo sudah sangat baik maka bagi yang tertarik untuk mendapatkan informasi secara lebih terperinci dipersilahkan untuk merujuk kepada file /usr/doc/lilo*

Contents

1 Pendahuluan

Meskipun dokumentasi yang terdapat didalam sumber-sumber Lilo (yang terinstal didalam file `/usr/doc/lilo-version`) sudah sangat lengkap, namun kebanyakan pemakai Linux mendapatkan kesulitan sewaktu membuat file `/etc/lilo.config` mereka sendiri. Dokumen ini dimaksudkan untuk membantu para pemakai Linux yang mendapat kesulitan tadi melalui pemberian informasi secara minimal serta memberikan lima buah contoh instalasi.

- Contoh pertama merupakan suatu instalasi "Linux dan lainnya" yang bersifat klasik.
- Contoh kedua menunjukkan cara menginstal Lilo kedalam sebuah hard disk yang terkoneksi sebagai `/dev/hdc` namun akan diboot sebagai `/dev/hda`. Contoh ini akan diperlukan ketika Anda menginstal sebuah Linux drive yang baru dengan menggunakan sistem Anda yang telah berjalan. Contoh ini menerangkan pula cara untuk melakukan boot dari disk SCSI.
- Contoh ketiga menunjukkan cara untuk melakukan boot sebuah sistem Linux dimana partisi root-nya tidak dapat diakses oleh BIOS
- Contoh keempat digunakan untuk mengakses sebuah disk yang berkapasitas besar dimana baik BIOS maupun DOS tidak dapat mengaksesnya secara mudah (masalah ini sudah merupakan suatu masalah yang basi).
- Contoh terakhir menunjukkan cara untuk melakukan restore dari sebuah disk yang rusak, jika penyebab kerusakan itu adalah karena telah menginstal operating sistem lain.

Ketiga contoh terakhir diatas berasal dari Cameron, `<mailto:cls@truffula.sj.ca.us>`, yang telah mengawali penulisan dokumen ini. Alessandro `<mailto:rubini@linux.it>`, pengelola dokumen saat ini tidak menjalankan sistem apapun kecuali Linux sehingga dia tidak dapat memeriksa maupun memperbarui dokumen ini sendiri. Namun demikian, setiap tanggapan pembaca akan diterima dengan senang hati.

2 Informasi Dasar dan Instalasi Standar

Pada saat Lilo mem-boot sebuah sistem, Lilo menggunakan BIOS call untuk me-load kernel dari disk (IDE drive, floppy, atau media lainnya). Oleh karena itu, kernel harus berada disuatu tempat yang dapat diakses oleh BIOS.

Pada saat dilakukan boot, Lilo tidak dapat membaca data filesistem, demikian pula setiap pathname yang Anda letakkan didalam file `/etc/lilo.conf` akan dicari (resolved) pada waktu instalasi (ketika Anda menjalankan `/sbin/lilo`).

Waktu instalasi adalah saat di mana program membuat tabel-tabel yang berisi daftar sektor yang diperlukan oleh file-file yang digunakan untuk melakukan load sistem operasi. Sebagai konsekwensinya maka file-file tsb harus berada pada suatu partisi yang dapat diakses oleh BIOS (file-file tadi biasanya berlokasi didalam direktori /boot, hal ini berarti dari sistem Linux Anda hanya partisi boot saja yang perlu diakses oleh BIOS).

Konsekwensi lain dari digunakannya sistem berbasis BIOS adalah Anda harus melakukan instal ulang terhadap loader (dalam hal ini Anda harus menjalankan ulang perintah `/sbin/lilo`) setiap kali Anda melakukan modifikasi terhadap setup Lilo. Setiap kali Anda mengkompilasi ulang kernel dan menghapus image lama yang Anda miliki, Andapun harus melakukan instal ulang terhadap Lilo.

2.1 Di manakah harus menginstal Lilo?

Direktif `boot=` yang terdapat didalam file `/etc/lilo.conf` menunjukkan letak penempatan Lilo didalam primary boot loader. Secara umum, Anda dapat menetapkan baik master boot record (`/dev/hda`) ataukah partisi root dari instalasi Linux Anda (biasanya `/dev/hda1` atau `/dev/hda2`).

Jika Anda memiliki sistem operasi lain yang telah terinstal kedalam hard disk Anda, sebaiknya Anda menginstal Lilo kedalam partisi root daripada menginstalnya kedalam MBR. Dalam kasus ini, Anda harus menandai partisi root tadi sebagai "bootable" dengan menggunakan perintah "a" dari fdisk atau perintah "b" dari cfdisk. Jika Anda tidak melakukan overwrite terhadap master boot record (MBR) maka Anda akan lebih mudah dalam melakukan uninstall baik terhadap Linux maupun Lilo jika Anda menginginkannya.

2.2 Bagaimana caranya mengkonfigurasi IDE hard drive?

Secara pribadi, penulis tidak menggunakan setting LBA maupun LARGE didalam BIOS (namun penulis hanya menjalankan Linux); Setting tersebut diperlukan oleh karena adanya kelemahan disain didalam dunia PC. Kelemahan disain PC tadi menyebabkan kernel harus berada didalam 1024 silinder pertama, namun hal ini tidak akan merupakan masalah selama Anda melakukan partisi terhadap hard disk dan menjaga agar partisi root tetap berukuran kecil (seperti yang seharusnya Anda lakukan).

Jika hard disk Anda telah berisi sistem operasi lain maka Anda tidak akan dapat memodifikasi setting dari BIOS, atau kalaupun Anda melakukannya maka sistem lama tadi tidak akan dapat bekerja. Distribusi Lilo akhir-akhir ini sudah dapat mengatasi setting disk baik LBA maupun LARGE.

Mohon dicatat bahwa kata-kunci *linear* yang terdapat didalam file `/etc/lilo.conf` dapat membantu mengatasi problem-problem yang berkaitan dengan geometri. Kata kunci tadi memerintahkan Lilo untuk menggunakan *linear sector address* sebagai pengganti dari koordinat sektor/head/silinder. Konversi kedalam address 3D akan diundurkan sampai ke saat run-time sehingga membuat setup dapat mengatasi problem-problem geometri.

Jika Anda memiliki lebih dari satu hard disk, dan beberapa diantaranya hanya dipakai untuk Linux serta tidak terlibat didalam proses boot maka Anda dapat memberitahukan BIOS bahwa hard disk tersebut tidak diinstal. Sistem Anda akan dapat di-boot secara lebih cepat dan Linux tidak akan membuang waktu untuk melakukan autodetect. Komputer yang dimiliki penulis seringkali berganti-ganti disk namun penulis tidak pernah melakukan perubahan terhadap konfigurasi BIOS.

2.3 Bagaimana caranya agar dapat berinteraksi pada waktu boot?

Ketika Anda melihat prompt dari Lilo, Anda dapat menekan tombol <Tab> untuk menampilkan daftar pilihan. Apabila Lilo tidak dikonfigurasi untuk bekerja secara interaktif maka tekanlah tombol <Alt> atau <Shift> sebelum muncul tulisan "LILO".

Jika Anda memilih untuk memboot kernel Linux maka Anda dapat menambahkan argumen baris perintah yang diberikan setelah nama sistem pilihan Anda. Kernel menerima banyak argumen baris perintah ini. Daftar dari semua argumen yang dapat digunakan tersebut dimuat didalam dokumen berjudul "BootPrompt-HOWTO" yang ditulis oleh Paul Gortmaker, karena itu disini penulis tidak akan mengulangi pemberian daftar argumen tersebut. Namun beberapa argumen baris perintah sangat penting dan berharga untuk diuraikan disini:

- `root=` Argumen ini dimaksudkan untuk meminta Linux kernel untuk me-mount sebuah partisi lain menggantikan partisi yang telah dicantumkan didalam lilo.conf sebagai partisi root. Misalnya, sistem yang dimiliki penulis memiliki sebuah partisi kecil yang digunakan bagi sebuah instalasi Linux yang minimal, penulis telah berhasil mem-boot sistem sesudah partisi root tadi terhapus secara tidak sengaja.
- `init=` Kernel Linux dari versi 1.3.43 atau lebih akhir lagi dapat mengeksekusi perintah yang diberikan melalui baris perintah selain dari perintah `/sbin/init`. Jika Anda mendapatkan kesulitan saat proses boot, Anda dapat mengakses sistem secara *bare* dengan cara menetapkan `init=/bin/sh` (pada saat Anda berada pada prompt dari shell kemungkinan besar Anda perlu melakukan mount terhadap disk Anda: cobalah perintah "mount -w -n -o remount /; mount -a", dan ingat untuk memberikan perintah "mount -a" sebelum Anda mematikan komputer)
- Suatu bilangan Dengan memberikan suatu bilangan pada baris perintah kernel, Anda memerintahkan init untuk memasuki suatu run-level tertentu (secara default run-level itu adalah 3 atau 2 tergantung kepada distribusi yang Anda gunakan). Silahkan untuk merujuk kepada dokumentasi init, `/etc/inittab`, dan `/etc/rc*.d` untuk mendapatkan keterangan lebih lanjut.

2.4 Bagaimana caranya melakukan uninstal terhadap Lilo?

Pada saat Lilo melakukan *overwrite* terhadap sebuah *boot sector*, Lilo akan menyimpan sebuah salinan dari *boot sector* tadi kedalam file `/boot/boot.xxyy` dimana xxyy adalah nomor-nomor major dan minor dari device tersebut didalam bilangan heksadesimal. Anda dapat melihat nomor-nomor major dan minor dari disk maupun partisi Anda dengan cara menjalankan perintah

```
ls   -l   /dev/device"
```

Misalnya, sektor pertama dari `/dev/hda` (nomor major-nya 3, dan nomor minor-nya 0) akan disimpan kedalam file `/boot/boot.0300`, menginstal Lilo kedalam `/dev/fd0` akan menyebabkan dibuatnya file `/boot/boot.0200`, demikian pula menginstal Lilo kedalam `/dev/sdb3` (nomor major-nya 8, nomor minor-nya 19) akan terbentuknya file `/boot/boot.0819`. Perlu dicatat disini bahwa Lilo tidak akan membuat suatu file jika file tersebut sudah terbentuk, sehingga Anda tidak perlu memperhatikan salinan tadi pada saat Anda melakukan instal ulang Lilo (misalnya sesudah melakukan kompilasi ulang kernel). Setiap backup copy yang terdapat didalam direktori `/boot` merupakan gambaran sesaat dari situasi sebelum Anda menginstal Lilo.

Jika Anda pada suatu saat ingin melakukan uninstal Lilo (misalnya dalam suatu kasus yang tidak menguntungkan, Anda terpaksa harus melakukan uninstal Linux) Anda hanya perlu melakukan restore dari boot sector yang asli. Jika Lilo telah diinstal kedalam `/dev/hda` maka Anda cukup hanya dengan memberikan perintah

```
dd  if=/boot/boot.0300   of=/dev/hda  bs=446   count=1
```

(secara pribadi penulis hanya memberikan perintah `cat /boot/boot.0300 > /dev/hda`, tetapi cara yang dilakukan penulis termasuk tidak aman karena akan mengembalikan tabel partisi lama yang mungkin selama ini telah Anda modifikasi).

Perintah tadi lebih mudah dijalankan dibandingkan dengan melakukan `fdisk /mbr` dari DOS shell: Perintah ini memungkinkan Anda untuk menghapus Linux dari disk secara rapi tanpa harus mem-boot apapun kecuali

Linux. Setelah melakukan penghapusan, Lilo akan mengingat untuk menjalankan fdisk dari Linux untuk menghapus setiappartisi Linux (fdisk dari DOS tidak mampu untuk menghapus partisi non-DOS).

Jika Anda menginstal Lilo ke dalam partisi root yang Anda miliki (misalnya `/dev/hda2`) maka tidak ada langkah-langkah khusus yang Anda perlukan untuk melakukan uninstal Lilo. Cukup hanya dengan menjalankan fdisk dari Linux untuk menghapus partisi Linux dari tabel partisi. Anda juga perlu menandai partisi DOS sebagai *bootable*.

3 Konfigurasi Sederhana

Hampir semua instalasi Lilo menggunakan sebuah file konfigurasi yang menyerupai file berikut ini:

```
boot = /dev/hda   # atau partisi root yang Anda miliki
delay = 10        # jeda (delay), dalam satuan sepersepuluh detik (agar Anda dapat berinteraksi)
vga = 0           # pilihan (optional). Gunakan  "vga=1"  untuk mendapatkan 80x50
#linear           # gunakan  "linear" jika Anda menghadapi masalah geometri.

image = /boot/vmlinux  # file zImage  yang Anda miliki
  root = /dev/hda1     # partisi root yang Anda miliki
  label = Linux        # atau nama bagus lainnya
  read-only            # root di-mount secara read-only

other = /dev/hda4   # adalah partisi DOS  Anda, jika ada
  table = /dev/hda  # tabel partisi saat ini
  label = dos       # atau nama yang kurang bagus lainnya
```

Anda dapat memiliki lebih dari satu seksi `image` dan seksi `other` jika Anda memerlukannya. Memiliki beberapa kernel image yang terkonfigurasi ke dalam `lilo.conf` bukan merupakan suatu yang luar biasa, setidaknya jika Anda senantiasa selalu menjaga agar kernel selalu memiliki versi yang terakhir hasil dari pengembangan kernel.

3.1 Bagaimana menghadapi Kernel-kernel yang berukuran besar

Jika Anda mengkompilasi sebuah kernel *zImage* dan ukurannya lebih besar dari setengah Mega byte (biasanya terdapat pada kernel-kernel versi v 2.1. yang baru) maka Anda harus membuat sebuah *big zImage* sebagai gantinya, dengan cara: `make bzImage`. Untuk memboot kernel yang berukuran besar tidak diperlukan sesuatu yang istimewa namun Anda memerlukan Lilo dengan versi 18 atau yang lebih baru dari versi itu. Jika Anda memiliki instalasi dengan versi lebih lama, Anda harus mengupgrade paket Lilo Anda.

3.2 Sumber-sumber Informasi lainnya

Sebagai tambahan terhadap dokumentasi Lilo, sejumlah mini-HOWTO berguna bagi Anda. Dokumen-dokumen tadi disebut sebagai "Linux+foobarOS" untuk suatu foobarOS, dokumen itu membahas masalah koeksistensi antara Linux dengan satu/beberapa sistem operasi lainnya. Demikian pula, dokumen berjudul *Multi-boot with Lilo* menjelaskan bagaimana berbagai Windows dapat dibuat koeksis dengan Linux.

4 Instalasi hdc untuk diboot sebagai hda dan Penggunaan `bios=`

Lilo memungkinkan untuk memetakan kernel image yang terdapat di dalam sebuah disk, dan menyuruh BIOS untuk mengambilnya dari disk yang lain. Misalnya, bagi penulis sendiri merupakan suatu hal yang biasa untuk menginstal Linux kedalam sebuah disk yang penulis koneksi sebagai hdc (master disk dari kontroler kedua) dan memboot disk tadi sebagai sebuah sistem berdiri sendiri pada sebuah kontroler IDE primer yang terdapat didalam komputer lain. Penulis membuat copy dari floppy instalasi kedalam sebuah partisi berukuran kecil, sehingga penulis dapat menjalankan `chroot` di dalam sebuah virtual console untuk menginstal hdc sementara penulis menggunakan sistem tadi untuk mengerjakan sesuatu yang lain.

File `lilo.conf` yang penulis gunakan untuk menginstal Lilo adalah sbb:

```
# File ini harus digunakan dari suatu sistem yang berjalan dari  /dev/hdc
boot = /dev/hdc   # melakukan overwrite terhadap MBR dari hdc
disk = /dev/hdc   # menceritakan sesuatu mengenai hdc:
   bios = 0x80    #  BIOS akan mengenalnya sebagai drive pertama
delay = 0
vga = 0

image = /boot/vmlinux  # ini berada di dalam  /dev/hdc1
  root = /dev/hda1     # akan tetapi pada saat boot akan menjadi hda1
  label = Linux
  read-only
```

File konfigurasi ini harus dibaca oleh Lilo yang dijalankan dari `/dev/hdc1`. Pemetaan Lilo yang ditulis sebagai boot sector (`/dev/hdc`) harus menunjuk kedalam file yang terdapat didalam direktori `/boot` (saat ini terinstal sebagai hdc); file-file tersebut akan diakses dibawah hda jika disk ini akan diboot sebagai sebuah sistem yang berdiri sendiri.

Penulis memberi-nama file konfigurasi ini sebagai `/mnt/etc/lilo.conf.hdc` (`/mnt` adalah direktori tempat hdc di-mount selama instalasi). Penulis menginstal Lilo dengan cara menjalankan perintah:

```
cd  /mnt; chroot . sbin/lilo -C /etc/lilo.conf.hdc
```

Jika perintah diatas nampak asing bagi Anda, silahkan untuk merujuk kepada halaman manual bagi chroot.

Direktif `bios=` yang terdapat di dalam file `lilo.conf` digunakan untuk memberitahukan BIOS mengenai device Anda. BIOS call mengidentifikasi floppy disk dan hard disk dengan sebuah angka: angka-angka 0x00 dan 0x01 digunakan untuk memilih floppy disk, sedangkan 0x80 dan angka berikutnya digunakan untuk memilih hard disk (BIOS lama hanya dapat mengakses dua buah disk saja). Pengertian dari `bios=0x80` didalam file file diatas adalah: *gunakanlah angka 0x80 didalam BIOS call untuk `/dev/hdc`*.

Direktif Lilo ini sangat berguna didalam situasi-situasi lainnya, misalnya pada saat BIOS dapat mem-boot dari SCSI disk sebagai ganti IDE disk. Pada saat kedua device yaitu IDE dan SCSI tersedia, Lilo tidak dapat membedakan apakah angka 0x80 digunakan untuk menunjuk kepada device pertama atau device lainnya karena pemakai dapat memilih salah satu dari kedua device tadi didalam menu konfigurasi BIOS, dan BIOS tidak dapat diakses pada saat Linux sedang berjalan.

Secara default, Lilo mengasumsikan bahwa IDE disk dipetakan oleh BIOS terlebih dahulu, namun asumsi ini dapat dirubah melalui penggunaan instruksi-instruksi berikut didalam file `/etc/lilo.conf`:

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

5 Penggunaan Lilo pada saat BIOS tidak dapat membaca Partisi Root

Penulis memiliki dua buah IDE drive dan sebuah SCSI drive. SCSI drive tadi tidak dapat dibaca oleh BIOS. Linux Loader atau Lilo menggunakan BIOS call, dan karena itu Lilo hanya dapat mengenal sebuah drive apabila BIOS mengenal drive tsb. AMI BIOS yang terbelakang milik penulis hanya dapat melakukan boot dari "A:" atau "C:". Filesistem root milik penulis sendiri berada didalam partisi yang terdapat didalam SCSI drive.

Jalan keluarnya adalah melakukan penempatan kernel, pemetaan file, dan penyambungan (chain) loader didalam sebuah partisi Linux dari IDE drive pertama. Perhatikan bahwa kernel Anda sendiri tidak perlu diletakkan didalam partisi root.

Partisi kedua dari IDE drive pertama tadi (`/dev/hda2`, partisi Linux digunakan untuk mem-boot sistem) di-mount kedalam direktori `/u2`. Berikut ini adalah file `lilo.conf` yang digunakan penulis:

```
#  Lilo diinstal kedalam Master Boot Record
#  dari drive IDE pertama.
#

boot = /dev/hda

#  /sbin/lilo (the installer) membuat copy dari Lilo boot record
#  dari file berikut dan dimasukkan kedalam lokasi MBR.

install = /u2/etc/lilo/boot.b

#
# Penulis membuat menu boot secara verbose.
# Lilo akan menemukannya dilokasi berikut.

message = /u2/etc/lilo/message

#  Installer akan membuat file berikut.
#  Memberitahu boot-loader dimana lokasi block-block dari kernels berada.

map = /u2/etc/lilo/map
compact
prompt

#  Tunggu selama 10 detik, kemudian lakukan boot
# terhadap kernel  1.2.1 kernel secara default.

timeout = 100
```

```
# Kernel diletakkan ditempat yang dapat dilihat oleh BIOS dengan cara sbb:
#       cp -p /usr/src/linux/arch/i386/boot/zImage /u2/z1.2.1
image = /u2/z1.2.1
        label = 1.2.1
# Lilo memberitahu kernel untuk melakukan mount
# terhadap partisi SCSI yang pertama sebagai root.
# BIOS tidak harus dapat melihatnya.

        root = /dev/sda1

#  Partisi ini akan diperiksa dan di-mount ulang oleh  /etc/rc.d/rc.S

        read-only

#  Penulis menyimpan sebuah kernel Slackware kernel
# untuk menjaga jika penulis tidak
#  berhasil membuat kernel. Sebetulnya penulis memerlukannya.

image = /u2/z1.0.9
        label = 1.0.9
        root = /dev/sda1
        read-only

#  Partisi DR-DOS 6 milik penulis.

other = /dev/hda1
        loader=/u2/etc/lilo/chain.b
        label = dos
        alias = m
```

6 Pengaksesan Disk berkapasitas besar pada saat BIOS tidak dapat melakukannya

Sistem yang terdapat di kantor penulis memiliki sebuah IDE drive berkapasitas 1 GB. BIOS hanya mampu membaca 504 MB pertama dari IDE drive tersebut (MB berarti 2**10 byte dan bukannya 10**6 byte). Sehingga penulis memiliki MS-DOS didalam sebuah partisi `/dev/hda1` sebesar 350 MB, dan root Linux didalam partisi `/dev/hda2` sebesar 120 MB.

MS-DOS tidak dapat menginstal sendiri secara benar pada saat drive masih baru. Novell DOS 7 juga mengalami hal yang sama. Beruntunglah penulis karena memiliki sebuah disket "OnTrack" yang terlupakan oleh "Options by IBM" untuk dimasukkan kedalam drive. Pada saat diterima, seharusnya drive tadi sudah dilengkapi dengan sebuah produk bernama "OnTrack Disk Manager". Jika Anda hanya memiliki MS-DOS maka, menurut penulis, seharusnya Anda menggunakan produk tersebut.

Selanjutnya, penulis membuat sebuah tabel partisi menggunakan fdisk dari Linux. MSDOS-6.2 menolak untuk diinstal kedalam `/dev/hda1`. Pesan yang diberikannya antara lain "This release of MS-DOS is for new installations. Your computer already has MS-DOS so you need to get an upgrade release from your dealer." ("Release MS-DOS ini diperuntukkan bagi instalasi baru. Komputer Anda telah memiliki MS-DOS sehingga Anda membutuhkan upgrade release dari dealer Anda").Padahal disk tadi sebenarnya merupakan sebuah disk baru.

Ah sialan! Kemudian penulis menjalankan fdisk dari Linux sekali lagi dan menghapus partisi 1 dari tabel partisi. Hal ini memuaskan MSDOS-6.2 yang kemudian melanjutkannya dengan membuat partisi yang persis sama dengan partisi 1 yang baru saja penulis hapus, serta menginstal MSDOS kedalamnya. MSDOS-6.2 menulis Master Boot Record kedalam drive tadi namun tidak dapat mem-boot-nya.

Penulis merasa beruntung dengan memiliki sebuah kernel Slackware yang berada didalam sebuah floppy (dibuat oleh program instalasi Slackware "setup"), sehingga penulis dapat mem-boot Linux dan menulis Lilo kedalam MBR dari MS-DOS yang telah rusak itu. Langkah ini berjalan baik. Berikut ini adalah file `/etc/lilo.conf` yang penulis gunakan:

```
boot = /dev/hda
map = /lilo-map
delay = 100
ramdisk = 0             # Non-aktifkan ramdisk didalam kernel Slackware
timeout = 100
prompt
disk = /dev/hda         # BIOS hanya dapat mengenali 500 MB pertama.
   bios = 0x80           # spesifikasikan  IDE drive pertama.
   sectors = 63         # angka-angka ini berasal dari dokumentasi drive Anda.
   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 = /boot/chain.b
```

Setelah penulis selesai menginstal sistem, penulis telah memverifikasi bahwa partisi yang berisi zImage, boot.b, map, chain.b, dan file-file message tadi ternyata dapat menggunakan sebuah MSDOS filesistem, selama filesistem itu tidak "bertumpuk (stackered)" atau "doublespaced." Sehingga penulis dapat membuat partisi DOS didalam `/dev/hda1` yang berukuran 500 MB.

Penulis juga telah mempelajari bahwa "OnTrack" mampu menulis sebuah tabel partisi kedalam drive yang dimulai dari beberapa lusin byte pertama, tidak dari posisi awal, dan merupakan suatu yang mungkin untuk melakukan hack terhadap driver IDE Linux agar dapat mengatasi masalah ini. Namun menginstal menggunakan precompile Slackware kernel tidak dimungkinkan. Akhirnya, IBM memberi penulis sebuah disket "OnTrack". Penulis menamakannya sebagai OnTrack technical support. Program itu memberi pesan bahwa

Linux mengalami kerusakan karena Linux tidak menggunakan BIOS. Akhirnya penulis tidak menggunakan lagi disket tadi.

7 Mem-boot menggunakan sebuah Floppy Penyelamat

Langkah selanjutnya adalah penulis menginstal Windows-95 kedalam sistem office. Langkah instalasi ini ternyata merusak Lilo MBR walapun partisi-partisi Linux tidak mengalami gangguan. Kernel membutuhkan waktu cukup lama untuk di-load dari floppy, karena itu penulis memasukkan setup Lilo kedalam sebuah floppy sehingga memungkinkan untuk mem-boot kernel yang berada didalam IDE drive.

Langkah-langkah yang dilakukan penulis untuk membuat floppy berisi Lilo tadi adalah sbb:

```
  fdformat /dev/fd0H1440      #  membuat track kedalam disket baru
  mkfs -t minix /dev/fd0 1440 #  membuat file sistem bertipe minix
  mount /dev/fd0 /mnt         #  mount kedalam titik mount  tmp standard
  cp -p /boot/chain.b /mnt    #  membuat copy  dari chain loader kedalamnya
  lilo -C /etc/lilo.flop      #  instal Lilo and lakukan map kedalam disket.
  umount /mnt
```

Perhatikan bahwa disket yang digunakan diatas harus terlebih dahulu di-mount sebelum Anda menjalankan installer agar Lilo dapat menulis map file secara benar.

Isi file `/etc/lilo.flop` dalam langkah diatas hampir sama dengan isi file sebelumnya:

```
#  Membuat sebuah floppy yang dapat mem-boot kernel dari hard disk
boot = /dev/fd0
map = /mnt/lilo-map
delay = 100
ramdisk = 0
timeout = 100
prompt
disk = /dev/hda     # 1 GB IDE, BIOS hanya dapat mengenal 500 MB pertama.
   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
```

Terakhir, penulis membutuhkan MS-DOS 6.2 bagi sistem office penulis, walaupun demikian penulis tidak bermaksud untuk melakukan perubahan terhadap drive pertama tadi. Penulis menambahkan sebuah kontroler dan drive SCSI, dilanjutkan dengan membuat filesistem MSDOS kedalamnya menggunakan mkdosfs dari Linux, dan Windows-95 mengenalnya sebagai "D:". Namun, tentu saja, MS-DOS tidak akan dapat di-boot dari D:. Hal ini tidak menjadi masalah jika Anda memiliki Lilo. Untuk itu, penulis menambahkan beberapa baris berikut kedalam file `lilo.conf` yang berasal dari contoh ke-2 sbb:

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

8 Catatan dari penterjemah

Penterjemah sudah berusaha menterjemahkan tulisan aslinya sedapat mungkin, namun tidak mustahil masih ada kekurangan dalam penterjemahan disana-sini. Apabila ada saran-saran untuk memperbaiki penterjemahan tulisan ini silahkan disampaikan via e-mail ke penterjemah, yheryadi@necnusa.co.id, atau koordinator id-LDP, mdamt@linux.or.id.