

Inhaltsverzeichnis

1	LILO	3
1.1	Überblick	4
1.2	Bedienung von LILO	6
1.3	LILO-Interns	7
1.4	LILO-Konfiguration	10
	Globale LILO-Optionen	10
	LILO-Optionen für den Start von Linux	12
	LILO-Optionen für den Start von Windows	13
1.5	LILO-Bootdiskette	14
1.6	LILO-Bootdiskette ohne Kernel auf der Diskette	15
	LILO-Bootdiskette mit eigenem Kernel auf der Diskette	16
1.7	LILO-Installation in den Bootsektor der Festplatte	18
	Sicherungskopie des Bootsektors erstellen	18
	LILO einrichten	18
	LILO von der Festplatte entfernen	19
1.8	Das 1024-Zylinder-Limit	19
1.9	SCSI/RAID/LVM/reiserfs-Systeme	21
	LILO-Zugriff auf Festplattensektoren	22
	LILO-Zugriff auf Festplattensektoren (Initial-RAM-Disk)	22
1.10	LILO durch den Bootmanager von Windows NT/2000/XP starten	26
	LILO-Konfiguration	27
1.11	LILO-Fehlermeldungen	28
1.12	Default-Betriebssystem für einen Reboot einstellen	30
1.13	Menüs und Grafikmodus	30
	LILO-Menü im Textmodus	31
	LILO-Menü im Grafikmodus	31
1.14	LILO-Konfigurationshilfen	32

1 LILO

HINWEIS: Dieser Abschnitt wurde unverändert aus der 6. Auflage des Buchs 'Linux: Installation, Konfiguration, Anwendung', Michael Kofler, Addison-Wesley-Verlag 2001 übernommen.

Der Abschnitt beschreibt die Konfiguration von LILO. In der 7. Auflage wird LILO nur mehr relativ kurz beschrieben, stattdessen gibt es einen sehr viel ausführlicheren Abschnitt zu GRUB. (GRUB ist eine Variante zu LILO, die in letzten Jahren wegen der einfacheren Wartbarkeit sehr populär geworden ist.)

Dieser Abschnitt soll eine Hilfestellung für alle Linux-Anwender sein, die noch mit LILO arbeiten. Der Text wird allerdings *nicht* mehr weiter erwartet, und ich kann auch keine Fragen dazu beantworten.

Mehr Informationen zum Linux-Buch von Michael Kofler finden Sie hier:

<http://www.kofler.cc/linux.html>

Zum Booten ist es erforderlich, dass die Linux-Kernel-Datei von der Diskette oder der Festplatte ins RAM geladen und dann gestartet wird. Der Kernel muss alle Treiber zum Zugriff auf die Root-Partition enthalten. Das Laden und Starten des Kernels ist die Aufgabe von LILO (Linux Loader). Dieses Programm steht daher im Mittelpunkt dieses Kapitels.

LILO startet nicht nur Linux, es kann auch dazu verwendet werden, andere Betriebssysteme (DOS, OS/2, diverse Windows-Versionen) hochzufahren. LILO ist insofern ein Schlüsselement im Hinblick auf Dual- oder Multi-Bootsysteme, bei denen Sie nach dem Rechnerstart angeben können, welches Betriebssystem geladen werden soll.

Dieser Abschnitt hat immer die friedliche Koexistenz mehrerer Betriebssysteme im Auge. Falls auf Ihrem Rechner als einziges Betriebssystem Linux installiert ist, brauchen Sie keine Rücksicht auf andere Betriebssysteme zu nehmen. In diesem Fall ist die LILO-Konfiguration weitgehend trivial, und Sie können den Großteil dieses Abschnitts einfach überblättern. (Es ist aber dennoch sinnvoll, dass Sie die Funktionsweise von LILO verstehen. Außerdem müssen Sie auch bei einem reinen Linux-System auf das 1024-Zylinder-Limit Rücksicht nehmen.)

Die LILO-Installation zählt neben der Partitionierung der Festplatte zu den kritischsten Teilen einer Linux-Installation. Wenn Sie dabei blind dem Installationsprogramm Ihrer Distribution vertrauen, ist die Chance recht groß, dass alles auf Anhieb funktioniert. Leider kann aber auch einiges schief gehen – und mit etwas Pech können Sie danach weder Linux noch ihr altes Betriebssystem booten!

Dieses Dilemma können Sie vermeiden, wenn Sie den folgenden Ratschlag befolgen: Erstellen Sie während der Linux-Installation eine Bootdiskette bzw. installieren Sie LILO auf eine Diskette (nicht auf die Festplatte)! Fast alle Distributionen bieten diese Möglichkeit.

Mit dieser Diskette können Sie Linux booten. Dann sollten Sie sich die Mühe machen und diesen Abschnitt lesen, bevor Sie LILO auf Ihre Festplatte installieren. Lesen Sie insbesondere die Abschnitte ab Seite 18, wo es darum geht, ein Backup des Bootsektors zu erstellen bzw. den Bootsektor nach einer fehlgeschlagenen LILO-Installation wieder herzustellen!

1.1 Überblick

Bevor es mit den zahlreichen LILO-Details losgeht, hilft ein kurzer Überblick bei der Orientierung in diesem recht langen Abschnitt.

- **LILO bedienen:** Vielleicht haben Sie Linux (samt LILO) schon installiert? Nach dem Rechnerstart (egal, ob von einer Diskette oder von der Festplatte) erscheinen die vier Buchstaben 'lilo' auf dem Bildschirm. Seite 6 beschreibt, wie es dann weitergeht.
- **LILO-Grundlagen:** Ganz egal, ob Sie LILO manuell einrichten (Konfigurationsdatei `/etc/lilo.conf`) oder ob Sie dazu ein Tool Ihrer Lieblingsdistribution einsetzen – Sie sollten wissen, was Sie dabei eigentlich tun! Ab Seite 7 wird das erforderliche Grundlagenwissen vermittelt und ab Seite 10 das Format der LILO-Konfigurationsdatei beschrieben.
- **LILO-Bootdisketten:** LILO kann wahlweise auf die Festplatte oder auf eine Diskette installiert werden. Erste Experimente sollten unbedingt mit einer Diskette durchgeführt werden! Ab Seite 14 werden drei verschiedene Möglichkeiten beschrieben, Bootdisketten zu erzeugen.
- **LILO-Installation auf der Festplatte:** Noch eleganter ist natürlich die LILO-Installation auf die Festplatte. Beim Rechnerstart können Sie zwischen verschiedenen Betriebssystemen auswählen (z. B. Windows oder Linux) – und das ohne langsame Zugriffe auf Disketten. Allerdings besteht die Gefahr, dass Sie nach einer fehlerhaften LILO-Installation auf die Festplatte weder Windows noch Linux starten können! Daher ist bei diesem Schritt Vorsicht angebracht. Auf Seite 18 wird nicht nur die Installation von LILO in den MBR (Master Boot Record)

beschrieben, sondern auch, wie Sie vorher eine Sicherheitskopie dieses Sektors erstellen und wie Sie LILO später einmal deinstallieren können.

- **LILO und das 1024-Zylinder-Limit:** Im Gegensatz zu Linux ist LILO auf das BIOS angewiesen. Dieses ist allerdings heute noch kompatibel mit Systemen aus der Computer-Steinzeit und verursacht daher häufig Probleme: Unter bestimmten Umständen können Sie Linux nur starten, wenn sich die Bootdateien in einer Partition am Anfang der Festplatte befinden. Ab Seite 19 finden Sie Hintergrundinformationen zu diesem Limit – und Tipps, wie dieses Limit umgangen werden kann.
- **LILO und Windows NT/2000/XP:** LILO ist leider nur unter bestimmten Voraussetzungen in der Lage, Windows NT/2000/XP zu starten. Dennoch stellt es kein Problem dar, ein Multibootsystem für Linux und Windows NT/2000/XP einzurichten. Ab Seite 26 wird beschrieben, wie das geht.
- **SCSI-, RAID, LVM-, Reiserfs-Systeme starten:** Damit der Linux-Kernel nach seinem Start auf die Systempartition (Root-Partition) zugreifen kann, benötigt er gegebenenfalls SCSI-, RAID-, LVM- oder spezielle Dateisystemtreiber. Wenn diese nicht Bestandteil des Kernels sind, müssen sie als Module von einer RAM-Disk geladen werden. Details dazu finden Sie ab Seite 21.
- **LILO-Fehlermeldungen:** Wenn der Start von Linux durch LILO nicht klappt, versucht LILO zumindest Fehlermeldungen anzugeben. Deren Interpretation ist allerdings eine eigene Kunst. Ab Seite 28 gibt es eine Einführung in diese Kunst.
- **LILO-Reboot:** Wenn Sie den Rechner für einen Neustart herunterfahren, können Sie mit `lilo -R` angeben, wie der Rechner anschließend neu gestartet werden soll.
- **Menüs und Grafikmodus:** Mit den aktuellen LILO-Versionen können Sie die Auswahl des gewünschten Betriebssystems mit einem Menü vereinfachen bzw. mit einer Hintergrundgrafik ansprechender gestalten – siehe Seite 30.
- **LILO-Konfigurationshilfen:** Sie können sich bei der LILO-Konfiguration von den Tools der diversen Distributionen helfen lassen. Das spart etwas Tippaufwand, führt aber meist nur dann zum gewünschten Ergebnis, wenn Sie auch verstehen, was Sie tun (und wenn das Tool der jeweiligen Distribution die Gegebenheiten Ihres Systems richtig erkennt, was leider nicht immer der Fall ist). Seite 32 zählt einige derartige Programme auf.

Zu LILO gibt es eine umfangreiche Online-Dokumentation. `man lilo.conf` und `man lilo` beschreiben das Installationsprogramm und das Format der Steuerungsdatei. Eine noch detailliertere Beschreibung wird bei den meisten Distributionen als Teil des LILO-Pakets mitgeliefert (siehe `rpm -qd lilo`, Dateien `text.ps` und `user.ps`). Lesenswert sind auch die diversen (Mini-)HOWTOs: Bootdisk, Boot+Root+Raid+LILO, LILO, LILO-crash-rescue, Linux+WinNT sowie Multiboot-with-LILO.

Einige Alternativen zu LILO werden in einem eigenen Abschnitt ab Seite ?? vorgestellt.

Zu guter Letzt enthalten die Anhänge dieses Buchs einige distributionsspezifische Tipps zu den Themen Bootdisketten, Rescue-System etc.

1.2 Bedienung von LILO

Dieser Abschnitt setzt voraus, dass LILO bereits auf einer Diskette oder auf der Festplatte installiert ist. Je nach Installation erscheint LILO unmittelbar nach dem Rechnerstart in Form einer schönen Grafik mit Menü-Bedienung per Cursor-Tasten, oder aber ganz spartanisch im Textmodus durch den Text 'LILO boot:'.

Normalerweise ist LILO so eingestellt, dass das Programm einige Sekunden darauf wartet, ob der Anwender irgendwelche Eingaben durchführen möchte. Ist das nicht der Fall, wird automatisch das bei der Konfiguration eingestellte Defaultsystem gestartet (zumeist Linux). Die Wartezeit können Sie verkürzen, indem Sie einfach **(←)** drücken.

Möglicherweise möchten Sie in den Bootvorgang eingreifen, etwa um ein anderes Betriebssystem auszuwählen oder um zusätzliche Bootoptionen zu übergeben (siehe auch Seite ??). In diesem Fall hängt die Bedienung vom LILO-Erscheinungsmodus ab.

Textmodus: In sehr seltenen Fällen müssen Sie als Erstes **(Shift)** drücken, um in den interaktiven Modus zu gelangen. Anschließend zeigt LILO eine Liste mit den Namen der zur Auswahl stehenden Betriebssysteme an, sobald Sie die Taste **(Tab)** drücken. Zur Auswahl stehen meist `linux` und `windows`. LILO kann aber auch so konfiguriert werden, dass Sie die Wahl zwischen mehreren Linux-Distributionen oder zwischen unterschiedlichen Kernel-Versionen haben. SUSE-Linux sieht sogar den Start eines Programms zum Speichertest (RAM) vor.

Auf jeden Fall können Sie nun zuerst einen der zur Auswahl stehenden Namen und dann die gewünschten zusätzlichen Parameter eintippen. Mit dem folgenden Kommando wird Linux gestartet. Allerdings wird statt der vordefinierten Defaultpartition `/dev/hdc7` als Root-Partition verwendet:

LILO

```
boot: <Tab>
linux  windows  memory
boot:  linux root=/dev/hda7
```

Grafikmodus: Bei manchen LILO-Installationen beendet (Esc) den Grafikmodus und Sie gelangen in den Textmodus – siehe oben. Ansonsten können Sie mit den Cursor-Tasten eines der Betriebssysteme auswählen und dann über die Tastatur zusätzliche Parameter eingeben (ohne den Grafikmodus zu verlassen).

Hinweis

Beachten Sie, dass bei der LILO-Eingabe normalerweise das amerikanische Tastaturlayout gilt! (Y) und (Z) sind vertauscht, die Sonderzeichen befinden sich an ungewohnten Stellen. Eine Übersetzungstabelle für die deutsche Tastatur finden Sie auf Seite ?? (Seit LILO-Version 20 ist es zwar möglich, ein anderes Tastaturlayout einzustellen, diese Option wird aber leider selten genutzt.)

1.3 LILO-Interna

Der LILO (Linux Loader) ist ein winziges Programm, das im Bootsektor (also im ersten Sektor) einer Diskette, Festplatte oder Festplattenpartition installiert werden kann. Das Programm ermöglicht die Auswahl zwischen mehreren installierten Betriebssystemen und insbesondere den Start von Linux. LILO ist die schnellste und beliebteste Methode, um Linux zu starten. (Dieses Kapitel beschreibt die LILO-Version 21.7.)

Vorsicht

Die LILO-Installation ist eine kritische Angelegenheit, besonders dann, wenn LILO in den MBR (Master Boot Record) der Festplatte installiert wird. Sollte dabei etwas schief gehen, können Sie Ihren Rechner ohne Bootdiskette weder unter Ihrem bisherigen Betriebssystem noch unter Linux hochfahren! Besonders problematisch ist die LILO-Installation auf Rechnern mit Windows NT/2000/XP, wenn die erste Festplattenpartition eine NTFS-Partition ist (d. h. keine FAT-Partition). Wenn Sie bei einer derartigen Konstellation LILO in den MBR installieren, können Sie anschließend möglicherweise Windows nicht mehr starten!

Vorsicht

Auch wenn Sie den Disk-Manager EZ-Drive verwenden, was wahrscheinlich nur auf sehr wenigen sehr alten Rechnern der Fall ist, dürfen Sie LILO auf keinen Fall im MBR installieren! Diese Einschränkung gilt vermutlich auch für andere Disk-Manager. (Disk-Manager erlauben den Zugriff auf große IDE-Platten trotz einer sehr alten BIOS-Version, siehe Seite ??.) Wenn Sie LILO dennoch verwenden möchten, ist nur eine Installation auf einer Diskette oder im ersten Sektor der Linux-Partition möglich – siehe die LILO-Dokumentation!

Bei den meisten Distributionen kann die LILO-Konfiguration als Teil des Installationsprozesses erfolgen. Wegen der oben beschriebenen Risiken hat eine manuelle LILO-Installation aber Vorteile. Sie haben eine viel bessere Kontrolle darüber, was wirklich passiert, können eine Backup-Diskette für den MBR erstellen etc. LILO in den MBR zu installieren ist keine Voraussetzung für den Betrieb von Linux! In den ersten Tagen – bis Sie ein wenig Sicherheit im Umgang mit Linux gewonnen haben – können Sie ohne weiteres auch von einer Diskette booten.

Vorsicht

Noch zwei Warnungen: Wenn Sie zuerst LILO und später irgendeine Version von Microsoft Windows installieren, wird LILO ungefragt überschrieben. An sich ist es kein Problem, LILO anschließend wieder herzustellen. Allerdings benötigen Sie eine Bootdiskette, damit Sie Linux überhaupt starten können (um dann dort das Kommando `lilo` auszuführen). Werfen Sie Ihre Bootdiskette also auch nach einer erfolgreichen LILO-Installation nicht weg!

Wenn Sie einen neuen Kernel installieren (oder den Kernel neu kompilieren), müssen Sie LILO neu installieren (siehe Seite ??). Denken Sie daran, auch Ihre Bootdiskette zu aktualisieren!

Funktionsweise

Die Funktion von LILO besteht (vereinfacht ausgedrückt) darin, dass er die Datei mit dem Linux-Kernel lädt und ausführt. Das hört sich allerdings einfacher an, als es in Wirklichkeit ist:

Genau genommen befindet sich im Bootsektor der Festplatte bzw. der Diskette ein winziges Programm (*first stage*), dessen einzige Aufgabe darin besteht, das größere LILO-Hauptprogramm zu laden (*second stage*). (Der Bootsektor, also der erste Sektor der Festplatte oder Diskette, wäre zu klein, um das gesamte LILO-Programm zu speichern! Der Code im Bootsektor wird vom BIOS beim Rechnerstart automatisch ausgeführt – daher spielt der Bootsektor eine derart wichtige Rolle für den Rechnerstart.)

Das LILO-Hauptprogramm (es handelt sich um die Datei `/boot/boot.b`) befindet sich irgendwo auf der Festplatte. Der Bootsektor mit dem Startprogramm enthält unter anderem die Nummern der Sektoren, auf denen das Hauptprogramm zu finden ist.

Sobald das LILO-Hauptprogramm läuft, bietet es dem Anwender die Möglichkeit, via Tastatur zwischen verschiedenen Betriebssystemen auszuwählen (siehe Seite 6). Wenn sich der Anwender für Linux entscheidet, muss LILO die Kernel-Datei laden. Dazu wurde bei der Installation von LILO eine Tabelle erstellt (die Datei `/boot/map`), die der Reihe nach alle Sektoren enthält, auf denen sich die Kernel-Datei befindet.

Der Grund für diese ungewöhnliche Vorgehensweise ist folgender: Während LILO läuft, fehlen noch jegliche Informationen über DOS-, Linux- oder andere Dateisysteme

me. Die Festplatte ist aus der Sicht von LILO eine riesige Ansammlung von Daten-sektoren. LILO muss daher in jeder Phase wissen, wo sich die als Nächstes benötigten Datensektoren befinden.

Aufgrund der obigen Beschreibung sollte auch klar sein, dass LILO nur funktioniert, solange der Ort der Kernel-Datei auf der Festplatte unverändert bleibt. (Jedes Kopierkommando und insbesondere jedes Neukompilieren des Kernels löst eine Veränderung aus, auch wenn der Dateiname unverändert bleibt! LILO muss in solchen Fällen durch die Ausführung des Kommandos `lilo` neu installiert werden! `lilo` führt dabei die Vorbereitungsarbeiten durch, d. h. es erzeugt aktualisierte Versionen von `/boot/boot.b` und `/boot/map` und speichert deren Startsektoren und andere Informationen im Bootsektor.)

Einschränkungen

LILO wird ausgeführt, bevor irgendein Betriebssystem läuft. Aus diesem Grund ist LILO – im Gegensatz zu Linux – auf das BIOS angewiesen, um Sektoren von der Festplatte zu lesen. Ältere BIOS-Versionen lassen aus historischen Gründen aber nur einen Zugriff auf die ersten 1024 Zylinder der Festplatte zu. Die sich daraus ergebenden Probleme (samt Lösungsvorschlägen) sind ab Seite 19 beschrieben. Lesen Sie unbedingt diesen Abschnitt, wenn Sie ein altes Mainboard verwenden (alt bedeutet vor ca. 1998).

Neben dem Zylinder-Limit existiert eine zweite Einschränkung: Ältere BIOS-Versionen können während des Bootprozesses nur die beiden ersten Festplatten ansprechen. Auf einem System mit mehreren Festplatten muss sich die Kernel-Datei also auf einer der beiden ersten Platten befinden. (Auch diese Einschränkung gilt nur bei alten Mainboards. Moderne Mainboards bzw. SCSI-Controller kennen dieses Problem nicht.)

Sowohl das Zylinder-Limit als auch die Limitierung auf zwei Festplatten gelten (wenn überhaupt) nur für LILO bzw. für den Ort der LILO-Dateien (inklusive Kernel). Sobald der Kernel einmal läuft, können alle Festplatten vollständig genutzt werden. Die `root`-Partition von Linux kann sich also durchaus auf der fünften Platte ab Zylinder 2500 befinden, solange nur der Kernel für LILO erreichbar ist.

Komplikationen kann es schließlich geben, wenn Sie unter Linux statt `ext2` ein anderes Dateisystem verwenden, wenn Sie mit SCSI- statt mit IDE-Festplatten arbeiten oder wenn Sie RAID oder LVM einsetzen. Die Probleme können zwei Ursachen haben: Erstens kann es passieren, dass LILO die Sektoren der Kernel-Datei nicht zuordnen kann. (Dieses Problem tritt bereits bei der Installation von LILO auf.) Zweitens kann es passieren, dass LILO zwar den Kernel erfolgreich laden und starten kann, dieser dann aber nicht in der Lage ist, das Root-Dateisystem zu lesen. Hintergrundinformationen und Lösungsvorschläge finden Sie auf Seite 21.

LILO kann momentan außer Linux die Betriebssysteme DOS, Windows 3.1/9x/ME (entspricht im Bootprozess jeweils DOS), OS/2 und einige Unix-Varianten booten. Nicht unterstützt wird leider Windows NT/2000/XP, das auf einen eigenen Bootloader angewiesen ist. Bei Windows NT/2000/XP kann aber umgekehrt der NT-Bootloader dazu eingesetzt werden, um LILO zu starten. Dieser Kooperationsweg ist ab Seite 26 beschrieben.

1.4 LILO-Konfiguration

Die Installation des LILO besteht aus zwei Schritten: Zuerst wird die Konfigurationsdatei `/etc/lilo.conf` erstellt, anschließend das Kommando `lilo` ausgeführt. Dieses Kommando wertet die Konfigurationsdatei aus, erstellt daraus einen neuen Bootsektor und schreibt diesen an den durch `lilo.conf` angegebenen Ort. Dabei handelt es sich zumeist um den MBR einer Festplatte oder Diskette, es kann sich aber auch um eine gewöhnliche Datei handeln.

Tipp

Selbst wenn Sie `lilo.conf` nicht ändern, müssen Sie `lilo` jedes Mal aufrufen, wenn sich die Kernel-Datei verändert (z. B. nach einem neuen Kompilieren)! Für LILO ist nicht der Dateiname relevant, sondern die Sektoren, in denen die Datei gespeichert ist.

Die Datei `lilo.conf` besteht aus zwei Teilen: Der erste Teil steuert das generelle Verhalten des Bootprogramms, der zweite Teil (Schlüsselwort `image` bzw. `other`) listet alle Betriebssysteme auf, die durch LILO gestartet werden (DOS, Windows, Linux). Das erste Betriebssystem dieser Liste gilt automatisch als Defaultbetriebssystem. Kommentare werden durch das Doppelkreuz (`#`) eingeleitet.

Im Prinzip können Sie in jeder Festplattenpartition ein eigenes Betriebssystem (etwa unterschiedliche Linux-Distributionen) installieren und LILO zur Auswahl der gewünschten Partition verwenden. Sie können LILO auch dazu verwenden, um zwischen verschiedenen Linux-Kernel-Dateien innerhalb einer Partition zu unterscheiden (etwa `vmlinuz`, `vmlinuz.old`). Das ist insbesondere dann interessant, wenn Sie eine neue Linux-Version testen möchten, ohne auf die alte zu verzichten.

Globale LILO-Optionen

boot: Der erste Teil von `/etc/lilo.conf` beginnt mit der Anweisung `boot=` und gibt an, wohin LILO installiert werden soll. Zur Installation auf eine Diskette geben Sie `/dev/fd0` an. Um LILO in den MBR der ersten IDE-Platte zu installieren, verwenden Sie `/dev/hda`, für die erste SCSI-Platte `/dev/sda`.

prompt: `prompt` bewirkt, dass LILO den Prompt (die Zeichen 'boot:') anzeigt und so verdeutlicht, dass jetzt Eingaben möglich sind. `prompt` sollte immer verwendet

werden. Wird `prompt` nicht verwendet, erscheint der Eingabe-Prompt nur, wenn die `(Shift)`-Taste gedrückt wird.

delay: `delay` gibt an, wie viele Zehntelsekunden LILO beim Booten auf ein manuelles Eingreifen wartet. Am schnellsten geht es mit 0, dann müssen Sie aber schon vor dem Start von LILO `(Shift)` gedrückt halten, wenn Sie nicht das Defaultbetriebssystem verwenden möchten.

Wenn Sie nur `prompt`, nicht aber `delay` angeben, wartet Linux unbegrenzt auf die Auswahl eines Betriebssystems. Damit ist ein unbeaufsichtigter Neustart ausgeschlossen. Wenn Linux als (Netzwerk-)Server eingesetzt wird, ist das aber nicht sinnvoll! Hier sollte LILO so konfiguriert werden, dass Linux (z. B. nach einem Stromausfall) ungefragt und ohne weitere Eingriffe sofort wieder gestartet wird.

lba32: `lba32` umgeht das 1024-Zylinder-Limit. `lilo` speichert Sektornummern nicht als CHS-Tripel (Cylinder, Head, Sector), sondern im 32-Bit-LBA-Format (Logical Block Addressing). Generell sollten Sie diese Option immer verwenden, wenn sich die Kernel-Dateien auf einer großen Festplatte befinden und Sie ein modernes Mainboard besitzen (ca. ab 1998). Die Hintergründe des 1024-Zylinder-Limit sind auf Seite 19 beschrieben.

linear: Die Option `linear` ist eine Alternative zu `lba32`. Sie bewirkt, dass die Sektoradressen in `/boot/map` als herkömmliche LBA-Werte gespeichert werden. Wenn sich LILO und das BIOS bei der Interpretation der CHS-Geometrie der Festplatte nicht einig sind, bietet `linear` oft die einfachste Lösung. `linear` muss oft verwendet werden, wenn Sie ein altes Mainboard oder einen alten SCSI-Controller verwenden. Sie können auf diese Weise nur die ersten 1024 Zylinder der Festplatte ansprechen.

compact: `compact` ermöglicht ein besonders schnelles Laden des Kernels (besonders von der Diskette), funktioniert aber nicht auf jedem Rechner bzw. mit jedem Festplatten-Controller.

bootmap: Mit der Option `map` kann die Datei angegeben werden, in der die Sektornummern der Kernel-Dateien und anderer Dateien gespeichert werden. Ohne diese Option wird `/boot/map` verwendet, was zumeist eine sinnvolle Voreinstellung ist. Die Option muss nur verwendet werden, wenn sich die Datei an einem anderen Ort befindet (z. B. auf einer Bootdiskette).

install: Eine weitere Option, auf deren Angabe oft verzichtet wird, ist `install`: Damit wird angegeben, in welche Datei das LILO-Hauptprogramm (der second stage loader) gespeichert wird. Die Defaulteinstellung lautet `/boot/boot.b`.

```
# Beispiel für /etc/lilo.conf (Teil 1)
# LILO global section
boot = /dev/fd0           # Installation im MBR einer Diskette
delay = 100              # 10 Sekunden warten
# prompt                 # Eingabe erzwingen (kein automatischer Start)
# compact                # schneller, besonders bei Disketten; kann bei
```

```

# map=/boot/map           # manchen Festplatten Probleme verursachen
# install=/boot/boot.b    # ohnedies Defaulteinstellung
# linear                   # manchmal bei alter Hardware erforderlich
lba32                      # nicht bei alten Mainboards (vor 1998)!

```

In manchen Fällen (große Platten und altes BIOS) hat LILO Probleme mit der Festplattengeometrie. Als ersten Lösungsversuch sollten Sie es mit der `linear`-Option versuchen. Wenn auch das nicht hilft, müssen Sie die Option `disk` mit deren Suboptionen `bios`, `sectors`, `heads` und `cylinders` verwenden. Hintergrundinformationen zu diesem Thema finden Sie auf Seite ?? sowie in der LILO-Dokumentation.

```

disk=/dev/hda             # Zusatzinformationen für Device /dev/hda
  bios=0x80               # 0x80 für die 1. Platte,
                        # 0x81 für die 2. Platte ...
  sectors=63              # Anzahl der Sektoren
  heads=255               # Anzahl der Köpfe
  cylinder=522            # Anzahl der Zylinder

```

LILO-Optionen für den Start von Linux

Im zweiten Teil von `lilo.conf` sind der Reihe nach bis zu 16 Betriebssystemvarianten aufgezählt, die wahlweise gestartet werden können. Die erste Variante gilt als Defaulteinstellung. Gemeinsames Merkmal aller Einträge ist das Kommando `label`, das der jeweiligen Variante einen Namen gibt. Diese Namen sind bei der manuellen Auswahl des Betriebssystems einzugeben – wählen Sie also kurze und aussagekräftige Namen ohne Leer- und Sonderzeichen.

Tipp

Sofern bei der LILO-Konfiguration keine spezielle Tastaturlayout angegeben wird (Schlüsselwort `keytable`, siehe LILO-Dokumentation), gilt für die Eingabe der Label-Namen beim LILO-Start das US-Tastaturlayout. Wenn Sie mit einer deutschen Tastatur arbeiten, vermeiden Sie Label-Namen mit Sonderzeichen und mit den Buchstaben Y und Z!

Um Linux zu booten, müssen außer `label` noch die Kommandos `image` und `root` angegeben werden. `image` bestimmt den Ort der Kernel-Datei, `root` die Partition, auf der sich das Wurzelverzeichnis befindet.

read-only: `read-only` gibt an, dass die Root-Partition zuerst `read-only` gemountet wird. Der Init-Prozess kann dann das Dateisystem kontrollieren und gegebenenfalls reparieren, bevor es im `Read-Write-Modus` neu (und endgültig) gemountet wird. Diese Option sollte immer verwendet werden!

append: Mit `append` können zusätzliche Kernel-Optionen angegeben werden (etwa um Hardware-Probleme zu vermeiden). Die wichtigsten Optionen sind auf Seite ?? beschrieben. Die Optionen gelten nur für in den Kernel integrierte Funktio-

nen, nicht aber für Module, die später geladen werden. (Modulooptionen werden in `/etc/modules.conf` angegeben – siehe Seite ??.)

initrd: `initrd` gibt den Namen einer RAM-Disk-Datei an, die von LILO geladen werden soll. Das bietet die Möglichkeit, gleich nach dem Start des Kernels ein Installationssystem zu starten oder Module nachzuladen. Das ist wichtig, wenn Sie SCSI-Festplatten oder ein besonderes Dateisystem verwenden (siehe Seite 21).

vga: `vga` bestimmt den VGA-Modus: Mögliche Einstellungen sind `extended` für einen Textmodus mit 50 Zeilen, `normal` für den Standardtextmodus und eine beliebige Zahl n größer 0 für den gewünschten VGA-Modus n . (Vorsicht! Wenn Sie einen Modus angeben, den Ihre VGA-Karte nicht kennt, können Sie Linux nicht korrekt verwenden!)

```
# Beispiel für /etc/lilo.conf (Teil 2)
# Linux images
image = /boot/vmlinuz           # Kernel-Datei
      label = linux             # Name
      root = /dev/hda8         # Root-Device
      read-only                # zuerst read-only laden
      # initrd = /boot/initrd  # SCSI/RAID/LVM/reiserfs etc.

# Alternative: 'linuxbak' zum Booten einer älteren Kernel-Version
image= /boot/vmlinuz.bak       # Kernel-Datei
      label = linuxbak         # Name
      root = /dev/hda8         # Root-Device
      read-only                # zuerst read-only laden
      # initrd = /boot/initrd  # SCSI/RAID/LVM/reiserfs etc.
      # append = "aic7xxx=extended" # Kernel-Option für SCSI-Karte
```

LILO-Optionen für den Start von Windows

Bei DOS- und Windows-3.1/9x/ME-Varianten wird das Schlüsselwort `other` zur Angabe der Partition verwendet. Mit den folgenden Zeilen wird das Betriebssystem gestartet, das sich in der ersten Partition der ersten Festplatte befindet.

```
# Datei /etc/lilo.conf, Teil 3
# DOS/Windows 3.1/9x/ME
other = /dev/hda1              # DOS/Windows-Partition
      label = windows          # Name
```

Hinweis

Die obigen Zeilen starten unter Umständen auch den Boot Loader von Windows NT/2000/XP – aber leider nicht immer. Der Start funktioniert bei manchen Windows-Installationen, wenn die erste Partition der Festplatte eine FAT-Partition ist (nicht NTFS!) und der NT-Boot-Loader im Bootsektor dieser Partition untergebracht ist.

Diese Voraussetzungen sind nicht bei allen Windows-Installationen gegeben. Es kann sein, dass der NT/2000/XP-Boot-Loader im MBR der Festplatte untergebracht ist bzw. dass der NT-Boot-Loader bestimmte Daten im MBR erwartet, die er dort (nach einer LILO-Installation) nicht findet. Daher sollten Sie bei Windows NT/2000/XP LILO nie in den MBR der Festplatte installieren und stattdessen den auf Seite 26 beschriebenen Weg einschlagen.

Die beiden folgenden Zeilen sehen fast so aus wie oben – aber es gibt einen wesentlichen Unterschied. Hier wird nicht eine bestimmte Partition angegeben, sondern die gesamte Festplatte (keine Partitionsnummer!). Damit ist der MBR der Festplatte gemeint, der von LILO ausgeführt wird. Das ist nur sinnvoll, wenn nicht LILO selbst in eben diesen MBR installiert wurde (sondern in den MBR einer anderen Festplatte oder einer Diskette). Eine Anwendung dieser Variante ist nur sinnvoll, wenn LILO auf eine Diskette installiert wird oder wenn der Rechner mit mehreren bootfähigen Festplatten ausgestattet ist.

```
other = /dev/hdb          # MBR der 2. Festplatte
      label = windows    # Name
```

Hinweis

Eine Menge weiterer LILO-Optionen werden in der Online-Dokumentation beschrieben – etwa `message` zur Ausgabe eines Informationstextes, `keytable` zur Definition eines bestimmten Tastaturlayouts, `password` für den Passwortschutz des Bootprozesses etc.

1.5 LILO-Bootdiskette

Selbst wenn Sie längerfristig LILO auf die Festplatte installieren möchten, benötigen Sie zuerst eine Bootdiskette! Der Grund: Wenn bei der LILO-Installation etwas schief geht, wenn bei einer späteren Installation von Windows LILO überschrieben wird etc., benötigen Sie eine Möglichkeit, Linux dennoch starten zu können und gegebenenfalls LILO neu einzurichten.

Im Regelfall haben Sie bereits während der Linux-Installation die Möglichkeit, eine Bootdiskette zu erstellen. Außerdem werden mit den meisten Distributionen Werkzeuge mitgeliefert, um komfortabel neue Bootdisketten zu erstellen. Im Anhang sind derartige Programme für einige Distributionen beschrieben.

Wenn Sie selbst Hand anlegen möchten, gibt es mehrere Möglichkeiten, Linux-Bootdisketten zu erzeugen:

<http://www.kofler.cc/linux.html>

- LILO-Bootdisketten ohne Kernel: Hier wird LILO in den MBR der Diskette geschrieben. LILO greift dann auf die Kernel-Datei der Festplatte zu (d. h. der Kernel befindet sich nicht auf der Diskette, daher die Bezeichnung 'ohne Kernel'). Vorteil: Sehr schneller Bootprozess (von der Diskette werden nur wenige Byte gelesen). Nachteile: Wenn der Kernel auf der Festplatte nicht gefunden wird (nach einem Neukompilieren, wegen des 1024-Zylinder-Limits etc.), ist die Bootdiskette nutzlos.
- LILO-Bootdiskette mit Kernel: Auf der Diskette wird ein Dateisystem angelegt. Anschließend werden alle LILO-relevanten Daten (`boot . b`, die Kernel-Datei etc.) auf die Diskette kopiert. LILO wird abermals in den MBR der Diskette installiert, findet jetzt aber alle zum Booten relevanten Dateien auf der Diskette. Vorteil: Zuverlässig, kein 1024-Zylinder-Limit. Nachteil: Der Bootprozess dauert um einige Sekunden länger.
- Bootdiskette ohne LILO: Hier wird die Kernel-Datei ohne Dateisystem direkt auf die Sektoren der Diskette geschrieben (beginnend mit dem MBR-Sektor). Beim Booten wird der Kernel ebenso geladen. Diese Variante ist auf Seite ?? beschrieben.

Noch ein Tipp: Obwohl der Konfigurationsaufwand für Variante 2 (LILO mit Kernel) am größten ist, bietet diese Variante die größte Sicherheit, dass Sie Linux in einem unvorhergesehenen Notfall wirklich starten können.

Vorsicht

Im Vergleich zu Festplatten sind Disketten ein sehr fragiles Medium. Sehr viele Probleme mit Bootdisketten haben mit der Verwendung alter Disketten zu tun. Sie sparen sich eine Menge Ärger, wenn Sie neue, möglichst vorformatierte Disketten verwenden!

1.6 LILO-Bootdiskette ohne Kernel auf der Diskette

Zum Einrichten benötigen Sie eine `lilo.conf`-Datei nach dem folgenden Muster. Entscheidend ist die erste Zeile mit der `boot`-Option, in der als Ziel für die LILO-Installation der MBR der Diskette angegeben wird. (Da beim Booten nur dieser eine Sektor gelesen wird, ist es überflüssig, auf der Diskette ein bestimmtes Dateisystem anzulegen.)

`/boot/vmlinuz` muss auf einen Kernel zeigen, der alle zum Zugriff auf die Root-Partition erforderlichen Treiber enthält. Pfad- und Device-Angaben müssen Sie an die Gegebenheiten Ihres Rechners anpassen. Unter Umständen benötigen Sie zusätzliche Optionen, die im Konfigurationsabschnitt beschrieben wurden (bei SCSI-Systemen `initrd`, siehe ab Seite 21).

```
# Datei /etc/lilo.conf
```

<http://www.kofler.cc/linux.html>

```

boot=/dev/fd0                # Device des Diskettenlaufwerks
prompt                       # LILO-Eingabe-Prompt anzeigen
timeout=100                  # 10 Sekunden warten
lba32                        # außer bei alten Mainboards
                             # (vor 1998)
image = /boot/vmlinuz        # Kernel-Datei
    label = linux
    root = /dev/hda8         # Root-Device
    read-only
    # initrd = /boot/initrd  # für SCSI/RAID/LVM/reiserfs etc.
other=/dev/hda1
    label=windows

```

Nachdem Sie die Konfigurationsdatei korrekt eingerichtet haben legen Sie eine formatierte Diskette in das Laufwerk und führen `lilo` aus.

```

root# lilo
Added linux *
Added windows

```

LILO-Bootdiskette mit eigenem Kernel auf der Diskette

Mit etwas mehr Aufwand ist es möglich, eine LILO-Bootdiskette zu erstellen, die zusätzlich einen eigenen Kernel besitzt. Auf dieser Diskette müssen sich ein Dateisystem (minix oder ext2), die Kernel-Datei und andere für den Bootprozess erforderliche Dateien (`boot.b` und `map`) befinden. Die Vorbereitungsarbeiten sehen folgendermaßen aus:

```

root# mkfs -t ext2 /dev/fd0 1440
root# mkdir /floppy
root# mount -t ext2 /dev/fd0 /floppy
root# mkdir /floppy/boot
root# cp /boot/vmlinuz /floppy/boot
root# cp /boot/boot.b /floppy/boot/
root# cp /boot/initrd /floppy/boot/

```

`mkfs` legt auf der Diskette ein ext2-Dateisystem an, das über das Verzeichnis `/floppy` angesprochen wird. Das letzte Kommando ist optional und nur bei SCSI/RAID/LVM/reiserfs-Systemen erforderlich.

Der zweite Schritt ist, dass Sie eine eigene LILO-Konfigurationsdatei für die Diskette erstellen. Der wesentliche Unterschied gegenüber der Konfigurationsdatei aus dem obigen Beispiel besteht darin, dass `map` und `install` jetzt auf Dateien der Diskette verweisen (statt wie sonst üblich auf Festplattendateien).

Die Datei `/floppy/boot/map` wurde in den obigen Kommandos übrigens nicht vergessen. Sie wird durch die Ausführung des `lilo`-Kommandos erzeugt.

Im Beispiel unten sind drei Bootvarianten vorgesehen. In der Defaulteinstellung wird versucht, von der Festplatte zu booten (das geht am schnellsten). Falls es dabei zu einem Absturz kommt (etwa weil die Kernel-Datei nicht mehr da ist, wo sie von LILO erwartet wird), kann ein neuer Versuch gestartet und beim LILO-Prompt die Variante `linuxfromdisk` gewählt werden. Beachten Sie, dass bei dieser Variante `image` auf die Kernel-Datei der Diskette verweist! Als dritte Variante ist – der Vollständigkeit halber – noch das Booten von DOS/Windows möglich.

```
# /etc/lilo.conf-floppy
boot = /dev/fd0                # Device des Diskettenlaufwerks
prompt                          # LILO-Prompt anzeigen
delay = 100                     # 10 Sekunden warten
install=/floppy/boot/boot.b    # auf der Diskette!
map=/floppy/boot/map           # auf der Diskette!
lba32                            # nicht bei alten Mainboards
                                # (vor 1998)
image = /boot/vmlinuz          # Kernel von der Festplatte
    label = linux
    root = /dev/hda8           # Root-Device
    read-only
image = /floppy/boot/vmlinuz    # Kernel von der Diskette
    label = linuxfromdisk
    root = /dev/hda8           # Root-Device
    read-only
other = /dev/hda1               # Windows 9x/ME
    label = windows
```

Mit den folgenden Kommandos installieren Sie LILO auf der Diskette. Das `chmod` ist erforderlich, weil sich LILO sonst beschwert.

```
root# chmod go-w /etc/lilo.conf-floppy
root# lilo -C /etc/lilo.conf-floppy
root# ls -lR /floppy/
drwxrwxr-x  2 root    root          80 Jun 16 20:24 boot

/floppy/boot:
total 11
-rw-rw-r--  1 root    root           3708 Jun 16 19:39 boot.b
-rw-----  1 root    root           7168 Jun 16 20:24 map
-rw-rw-r--  1 root    root        403672 Jun 16 19:37 vmlinuz
root# sync
root# umount /floppy
```

Hinweis

Noch mehr Informationen zum Erstellen von ausgeklügelten Bootdisketten, die sogar ein eigenes `root`-Dateisystem besitzen und sich daher für Wartungsaufgaben eignen, finden Sie im HOWTO-Text zum Thema Bootdisketten.

1.7 LILO-Installation in den Bootsektor der Festplatte

Vorsicht

Nochmals: Von allen LILO-Varianten ist diese die gefährlichste! Sie kann mit einer Windows-NT/2000/XP-Installation inkompatibel sein. Sie ist auf jedem Fall inkompatibel mit alten Disk-Managern (DOS-Zugriff auf große IDE-Platten bei sehr altem BIOS)!

Sicherungskopie des Bootsektors erstellen

Bei der Ausführung von `lilo` zur Installation in den MBR einer Festplatte wird automatisch getestet, ob sich im Verzeichnis `/boot` bereits eine Sicherheitskopie des Bootsektors befindet. Nur wenn das nicht der Fall ist (also beim ersten Ausführen von `lilo`), kopiert das Kommando den aktuellen Bootsektor in die Datei `/boot/boot.0300` (bei IDE-Platten) oder `/boot/boot.0800` (bei SCSI-Platten). Wenn Sie den Bootsektor manuell sichern möchten, müssen Sie eines der folgenden Kommandos ausführen, *bevor* Sie `lilo` zum ersten Mal ausführen. Das erste Kommando gilt für die erste IDE-Platte, das zweite Kommando gilt für die erste SCSI-Platte:

```
root# dd if=/dev/hda of=/boot/bootsektor.ide bs=512 count=1
root# dd if=/dev/sda of=/boot/bootsektor.scsi bs=512 count=1
```

Wenn Sie den aktuellen Bootsektor auf eine Diskette übertragen möchten, benötigen Sie eine formatierte Diskette (siehe `fdformat` auf Seite ??). Anschließend führen Sie eines der beiden folgenden Kommandos aus (IDE/SCSI):

```
root# dd if=/dev/hda of=/dev/fd0 bs=512 count=1
root# dd if=/dev/sda of=/dev/fd0 bs=512 count=1
```

Wenn sich die so erstellte Diskette beim Einschalten des Rechners in Laufwerk A: befindet, bootet der Rechner so wie vom bisher auf der Festplatte befindlichen Bootsektor!

LILO einrichten

Zum Einrichten benötigen Sie eine `lilo.conf`-Datei, die bis auf die `boot`-Zeile so aussieht wie die Diskettenversion. Pfad- und Device-Angaben müssen Sie an die Gegebenheiten Ihres Rechners anpassen. Unter Umständen benötigen Sie zusätzliche Optionen, die im Konfigurationsabschnitt oben beschrieben wurden.

```
# Datei /etc/lilo.conf
```

```
boot=/dev/hda                # Device der ersten IDE-Platte
# boot=/dev/sda              # Device der ersten SCSI-Platte
prompt
timeout=100
lba32                         # nicht bei alten Mainboards (vor 1998)
image = /boot/vmlinuz        # Kernel-Datei
    label = linux
    root = /dev/hda8         # Root-Device
    read-only
    # initrd = /boot/initrd  # bei SCSI/RAID/LVM/reiserfs-Systemen
other=/dev/hda1
    label=windows
```

Um LILO zu installieren, führen Sie als root das Kommando `lilo` aus:

```
root# lilo
```

LILO von der Festplatte entfernen

Um LILO wieder von der Festplatte zu entfernen, müssen Sie den Bootsektor wiederherstellen. Im einfachsten Fall führen Sie dazu einfach `lilo -u` aus. LILO liest dann aus `/boot` den bei der ersten LILO-Installation gesicherten Bootsektor und überschreibt damit den aktuellen LILO-Bootsektor.

```
root# lilo -u
```

Wenn Sie eine eigene Sicherheitskopie des Bootsektors angelegt haben, können Sie diesen mit `dd` wieder installieren. Das erste Kommando gilt für die erste IDE-Platte, das zweite Kommando gilt für die erste SCSI-Platte:

```
root# dd if=/boot/bootsektor.ide of=/dev/hda bs=512 count=1
root# dd if=/boot/bootsektor.scsi of=/dev/sda bs=512 count=1
```

Falls es bei der Deinstallation von LILO Probleme gibt, können Sie DOS über eine Bootdiskette starten und dort `FDISK /MBR` ausführen. Damit wird ein Bootsektor zum automatischen Start von DOS/Windows eingerichtet (und der LILO-Bootsektor überschrieben). Diese Vorgehensweise ist allerdings bei einigen Windows-9x-Versionen und bei allen NT-Versionen unmöglich!

1.8 Das 1024-Zylinder-Limit

Wie bereits erwähnt wurde, ist LILO beim Laden der Kernel-Datei von der Festplatte auf das BIOS angewiesen. Aus historischen Gründen ermöglicht das Standard-BIOS nur eine Adressierung der ersten 1024 Zylinder der Festplatte. Die Zylindergröße ist

vom Alter des Mainboard und der Festplatte abhängig; daher kann der Bereich der ersten 1024 Zylinder zwischen 500 MByte und 8 GByte schwanken (siehe Seite ??.)

Neuere Mainboards (ca. seit 1998) sind mit einer BIOS-Erweiterung ausgestattet (den so genannten Extended INT13 Functions). Aktuelle LILO-Versionen (genau genommen seit 21.4) kommen mit dieser Erweiterung zurecht. (Ihre LILO-Version können Sie übrigens mit `lilo -v` ermitteln.)

Wenn Sie also kein zu altes Mainboard haben, gibt es für Sie kein 1024-Zylinder-Limit. Sie müssen lediglich in `lilo.conf` die Option `lba32` angeben.

Aber natürlich gibt es auch mit alten Mainboards viele Möglichkeiten, Linux zu booten:

- Am einfachsten ist es, eine LILO-Bootdiskette mit Kernel zu erstellen (siehe Seite 16).
- Am elegantesten ist es, eine kleine `/boot`-Partition unterhalb der 1024-Zylinder-Grenze anzulegen, in der sich alle für LILO relevanten Dateien befinden. Manche Distributionen schlagen bei der Installation automatisch vor, eine derartige Partition einzurichten. Wenn die ersten 1024 Zylinder der Festplatte allerdings schon mit anderen Partitionen gefüllt sind, ist diese Variante ausgeschlossen. Elegant ist diese Variante deswegen, weil keinerlei Änderungen an `/etc/lilo.conf` notwendig sind, weil direkt von der Festplatte gebootet wird und weil alle Daten von Linux-Partitionen stammen.
- Falls sich innerhalb der ersten 1024 Zylinder eine Windows-3.1/9x/ME-Partition (nicht NTFS!) befindet, können Sie die für LILO relevanten Dateien auch dorthin kopieren. Der Nachteil dieser Variante besteht darin, dass LILO nicht mehr funktioniert, sobald sich der Ort der LILO-Dateien in der Windows-Partition ändert (etwa nachdem die Partition mit einem Programm wie `CHKDSK.EXE` oder `SCANDISK.EXE` defragmentiert wird).

Die folgende Beschreibung erklärt, wie Sie die Kernel-Datei und andere Bootinformationen in einer Windows-Partition unterhalb dieser Grenze unterbringen. Dabei wird angenommen, dass die Partition als `/dev/hda1` angesprochen und vorübergehend als `/winc` in das Linux-Dateisystem eingebunden wird.

```
root# mkdir /winc
root# mount -t vfat /dev/hda1 /winc
root# mkdir /winc/lilo
root# cp /boot/vmlinuz /winc/lilo
root# cp /boot/* /winc/lilo
```

In `/etc/lilo.conf` müssen die Pfadangaben zur Kernel-Datei und eventuell zu anderen Dateien entsprechend geändert werden. Neu im Vergleich zu bisherigen `lilo.conf`-Dateien sind die Optionen `install` und `map`, die nicht mehr (wie in der Defaulteinstellung) in das Linux-Verzeichnis `/boot` zeigen, sondern in das Windows-Verzeichnis `C:\LILO`.

Beachten Sie aber, dass die Einstellung für `root` unverändert bleibt! (Die `root`-Option gibt an, wo sich die Linux-Root-Partition befindet. Deren Ort hat sich nicht verändert!)

```
# Datei /etc/lilo.conf
boot=/dev/hda
prompt
timeout=100
install=/winc/lilo/boot.b
map=/winc/lilo/map
image= /winc/lilo/vmlinuz
    root = /dev/hda8          # Linux Root-Partition
    label = linux
    read-only
other=/dev/hda1
    label=windows
```

Jetzt können Sie `lilo` ausführen. Die Windows-Partition muss dabei noch immer gemountet sein. Bei einem Neustart lädt LILO die Kernel-Datei aus der Windows-Partition und startet danach Linux.

Vorsicht

Das Problem bei dieser Art der LILO-Installation besteht darin, dass LILO darauf angewiesen ist, dass sich der Ort der Kernel-Datei nicht ändert. (LILO speichert nicht den Dateinamen, sondern die Sektornummern, auf denen sich die Datei befindet.) Wenn Sie die Windows-Partition defragmentieren oder das `lilo`-Verzeichnis verschieben, kopieren etc., kann sich der Ort der Datei auf der Platte ändern. LILO findet dann die Kernel-Datei nicht mehr und das Booten schlägt fehl. Abhilfe: Linux muss mit einer Bootdiskette gestartet, die Windows-Partition eingebunden und `lilo` abermals ausgeführt werden.

Tipp

Wenn Sie einen neuen Kernel installieren, müssen Sie die Windows-Partition wieder mounten, die neue Kernel-Datei dorthin kopieren und `lilo` neu ausführen. Die Datei `/boot/vmlinuz` spielt im Gegensatz zu einer 'normalen' LILO-Konfiguration keine Rolle – es kommt einzig auf `/winc/lilo/vmlinuz` an!

1.9 SCSI/RAID/LVM/reiserfs-Systeme

Vielleicht fragen Sie sich, was der gemeinsame Nenner der Überschrift ist. Nun, alle Begriffe haben mit dem Zugriff auf das Dateisystem zu tun. Und der ist gleich zweimal von Interesse: einmal während der Installation von LILO bei der Ermittlung der Sektorenliste für die Bootdateien und einmal nach dem geglückten Kernel-Start, wenn der Kernel auf das Root-Dateisystem zugreifen möchte.

Je nach Konfiguration kann es sein, dass Sie nur mit einem oder auch mit beiden Problemen konfrontiert sind. Auch die Lösungsansätze (separate `/boot`-Partition, Verwendung einer Initial-RAM-Disk) sind voneinander unabhängig. In den folgenden beiden Abschnitten finden Sie eine genauere Ursachenanalyse.

LILO-Zugriff auf Festplattensektoren

`lilo` muss während der Installation in der Lage sein, dem im Dateisystem gespeicherten Kernel Festplattensektoren zuzuordnen. Die Sektorenliste ist die Basis für den späteren Startprozess.

Wenn der Kernel in einem ext2- oder Windows-9x-Dateisystem gespeichert ist, kann `lilo` die Sektornummern problemlos ermitteln. Wenn Sie andere Dateisysteme verwenden oder wenn zwischen dem Dateisystem und der Festplatte andere Subsysteme stehen (LVM oder RAID, siehe Kapitel ??), kann es Probleme geben:

- **Dateisysteme:** Aktuelle LILO-Versionen kommen problemlos mit reiserfs zurecht. Bei älteren LILO-Versionen muss das reiserfs-Dateisystem mit der mount-Option `notail` in das Dateisystem eingebunden werden.

Laut Dokumentation bereitet auch die Kombination zwischen SGI xfs bzw. IBM jfs und LILO kein Problem – ich habe das aber nicht selbst getestet.

- **LVM:** Für das korrekte Zusammenspiel zwischen LVM und LILO gab es erste Patches, als diese Zeilen geschrieben wurden. Es ist daher zu erwarten, dass künftige LVM- und LILO-Versionen miteinander kompatibel sein werden. (Zumindest bis zur LILO-Version 21.7 ist das nicht der Fall.)
- **RAID:** LILO kommt zurzeit ausschließlich mit RAID-1 zurecht (und nur mit den RAID-Tools 0.9, nicht mit der älteren Version 0.4.)
- **SCSI-Festplatten:** Für die Sektorenerkennung spielt es keine Rolle, ob Sie mit IDE- oder SCSI-Festplatten arbeiten, d. h. es gibt keine Probleme.

Wenn es Probleme gibt, ist die Lösung eigentlich einfach: Sie müssen eine kleine `/boot`-Partition mit ext2-Dateisystem ohne LVM und ohne RAID anlegen. Wenn Sie auf Ihrer Festplatte keinen Platz mehr für eine kleine Partition haben (10 MByte reichen), müssen Sie auf eine Bootdiskette mit Kernel zurückgreifen (siehe Seite 16).

LILO-Zugriff auf Festplattensektoren (Initial-RAM-Disk)

Sobald es LILO einmal gelingt, den Kernel zu starten, muss dieser unmittelbar nach dem Start auf das Root-Dateisystem zugreifen können. Der mit den meisten Linux-Distributionen mitgelieferte Standardkernel ist dazu nur in der Lage, wenn Sie IDE-Festplatten und ein ext2-Dateisystem verwenden. Alle Zusatzfunktionen zur Steuerung von SCSI-Controllern, für besondere Dateisysteme, für RAID oder für LVM befinden sich in Modulen, die normalerweise bei Bedarf geladen werden. Noch ist das

aber nicht möglich, weil sie ja im Dateisystem gespeichert sind, das noch gar nicht gelesen werden kann.

Es gibt drei Lösungen für das Problem:

- Sie legen die root-Partition auf einer IDE-Festplatte an und verwenden dort ein ext2-Dateisystem ohne LVM oder RAID.
- Sie kompilieren selbst einen Kernel, in den alle Funktionen zum Zugriff auf das Root-Dateisystem bereits integriert sind. Der Nachteil besteht darin, dass das Selbstkompilieren oft Probleme bereitet und dass die resultierende Kernel-Datei bisweilen sehr groß wird, was wiederum neue LILO-Probleme verursachen kann.
- Die in den meisten Fällen beste Lösung besteht darin, eine so genannte Initial-RAM-Disk zu verwenden. Der Rest dieses Abschnitts beschränkt sich auf diese Variante.

Ein wenig vereinfacht sieht die Vorgehensweise so aus: Vor der LILO-Installation wird ein kleines Dateisystem erstellt, in dem alle erforderlichen Kernel-Module gespeichert werden. Das gesamte Dateisystem wird in einer einzigen, komprimierten Datei gespeichert. Diese Datei wird in der LILO-Konfigurationsdatei mit der Option `initrd=/boot/initrd` angegeben.

Während des Startprozesses lädt LILO die Datei in den Speicher (RAM), spricht sie wie ein Dateisystem an (eben als RAM-Disk) und stellt sie dem Kernel zur Verfügung. Dieser lädt sich von dort die Module, bevor er dann auf das root-Dateisystem zugreift.

Verweis Exakter und mit mehr Details ist die Verwendung von Initial-RAM-Disks in der folgenden Datei beschrieben (die Teil der Kernel-Codes ist):
`/usr/src/linux/Documentation/initrd.txt`

Der einzig wirklich komplizierte Aspekt bei diesem Verfahren besteht darin, die Datei für die Initial-RAM-Disk herzustellen. Zum Glück helfen dabei eigene Kommandos, die aber je nach Distribution unterschiedlich sind (siehe unten).

Bevor Sie sich mit diesen Kommandos beschäftigen, stellt sich noch die Frage, welche Module Sie eigentlich benötigen. Dazu führen Sie am besten `lsmod` aus und konsultieren `/etc/modules.conf`. Bei der Suche nach den richtigen Modulnamen können Sie auch einen Blick in die Modulverzeichnisse werfen. Die folgenden Verzeichnisangaben sind relativ zu `/lib/modules/kernelversion/kernel`.

SCSI: `drivers/scsi`.

Dateisysteme: `fs/*/*`.

LVM und RAID: `drivers/md`

Hinweis Zur Erzeugung der Initial-RAM-Disk-Datei wird die Datei mit Hilfe eines so genannten Loopback-Device als Dateisystem angesprochen. Damit das klappt, muss der laufende Linux-Kernel diese Funktion unterstützen bzw. das entsprechende Kernel-Modul geladen werden (zur Not manuell: `modprobe loop`).

Verweis Speziell für die LVM-Module gibt es ein eigenes Kommando zur Erzeugung einer Initial-RAM-Disk: `lvmcreate_initrd`. Dieses Kommando berücksichtigt allerdings nur LVM, nicht aber andere, eventuell ebenfalls erforderliche Module. Weitere Informationen gibt die `man`-Seite zu diesem Kommando.

Vorsicht Bitte denken Sie daran, dass Sie jedes Mal, wenn Sie eine neue RAM-Disk-Datei erzeugen, auch `lilo` neu ausführen müssen. Die neue RAM-Disk-Datei befindet sich nicht auf denselben Sektoren der Festplatten, daher findet die alte LILO-Konfiguration die Datei nicht mehr.

mkinitrd (Mandrake, Red Hat): `mkinitrd` wertet die Datei `/etc/modules.conf` aus und kopiert alle in der Zeile `scsi_hostadapter` angegebenen Module in die RAM-Disk.

```
# in /etc/modules.conf
alias scsi_hostadapter aic7xxx
```

Wenn sich das root-Dateisystem in einer RAID-Partition befindet, werden automatisch auch die hierfür erforderlichen Module in die RAM-Disk kopiert.

Alle weiteren Module – etwa für das Root-Dateisystem oder für LVM – müssen explizit mit der Option `--with=modulname` angegeben werden. (Für jedes Modul ist eine eigene `--with`-Option notwendig.) Die zurzeit aktiven Module können Sie mit `lsmod` bestimmen.

Als weitere Parameter müssen der Name der RAM-Disk-Datei (üblicherweise `/boot/initrd` sowie die Kernel-Version übergeben werden. (Diese kann mit `uname -r` ermittelt werden.) Wenn Sie eine bereits vorhandene RAM-Disk-Datei überschreiben möchten, benötigen Sie außerdem die Option `-f`. Einige weitere Optionen sind im `man`-Text zu `mkinitrd` beschrieben.

Das folgende Kommando erzeugt eine RAM-Disk-Datei für das reiserfs-Dateisystem.

```
root# mkinitrd -f --with=reiserfs /boot/initrd 2.4.2-2
```

mk_initrd (SUSE): Bei SUSE hat das Kommando nicht nur einen zusätzlichen Unterstrich, auch die Steuerung erfolgt ein wenig anders. Normalerweise müssen an das Kommando keinerlei Parameter oder Optionen übergeben werden. `mk_initrd` wertet die Variable `INITRD_MODULES` in der Datei `/etc/rc.config` aus. Diese Variable enthält nach einer Installation auf einem SCSI-System üblicherweise das benötigte

SCSI-Modul. Außerdem erkennt `mk_initrd` automatisch, ob LVM verwendet wird, und fügt bei Bedarf auch dieses Modul hinzu.

`mk_initrd` kümmert sich allerdings weder um RAID-Module noch um Module für zusätzliche Dateisysteme – diese müssen Sie gegebenenfalls selbst in `/etc/rc.config` eintragen. (Sie können die Module auch mit der Option `-m` direkt an `mk_initrd` übergeben.)

Das folgende Beispiel geht davon aus, dass sich Ihr Linux-System in einer reiserfs-Partition auf einer SCSI-Festplatte befindet, die über eine SCSI-Karte von Adaptec angesprochen wird. (Sie können in `INITRD_MODULES` aber natürlich beliebige weitere Module angeben. Achten Sie darauf, dass Sie die Module durch Leerzeichen, nicht durch Kommas trennen!)

```
# in /etc/rc.config
INITRD_MODULES="aic7xxx reiserfs"
```

`mk_initrd` erzeugt nicht nur eine RAM-Disk-Datei, sondern gleich zwei: `/boot/initrd` und `/boot/initrd.suse`. Der Grund besteht darin, dass SUSE per Default zwei gleiche Kernel installiert, `vmlinuz` und `vmlinuz.suse`. Wenn Sie selbst einen eigenen Kernel kompilieren, ersetzen Sie damit `vmlinuz`; `vmlinuz.suse` bleibt aber unverändert. Beim Booten mit LILO können Sie dann entscheiden, ob Sie ihren eigenen Kernel ('linux') oder den von SUSE ('suse') verwenden möchten. (Das ist dann praktisch, wenn Ihnen beim Kompilieren ein Fehler unterläuft.)

```
root# mk_initrd
using "/dev/hdb11" as root device (mounted on "/")
creating initrd "//boot/initrd" for kernel "//boot/vmlinuz" (2.4.4-4GB)
module aic7xxx is
  "/lib/modules/2.4.4-4GB/kernel/drivers/scsi/aic7xxx/aic7xxx.o"
  -> insmod aic7xxx
module reiserfs is
  "/lib/modules/2.4.4-4GB/kernel/fs/reiserfs/reiserfs.o"
  -> insmod reiserfs

creating initrd "//boot/initrd.suse" for kernel "//boot/vmlinuz.suse"
(2.4.4-4GB)
module aic7xxx is
  "/lib/modules/2.4.4-4GB/kernel/drivers/scsi/aic7xxx/aic7xxx.o"
  -> insmod aic7xxx
module reiserfs is
  "/lib/modules/2.4.4-4GB/kernel/fs/reiserfs/reiserfs.o"
  -> insmod reiserfs
```

Verweis

Weitere Informationen zu `mk_initrd` bekommen Sie mit der Option `-h` oder wenn Sie den Quelltext lesen (Datei `/sbin/mk_initrd`). Es gibt auch in der SUSE-Supportdatenbank <http://portal.suse.de/sdb/de/index.html> einige Artikel zu diesem Thema – suchen Sie nach `initrd`.

1.10 LILO durch den Bootmanager von Windows NT/2000/XP starten

Windows NT/2000/XP verwendet einen eigenen Bootmanager, der ähnlich wie LILO funktioniert und normalerweise in den MBR der ersten Festplatte oder der ersten Partition installiert wird. Mit dem Bootmanager können Sie verschiedene Windows-Versionen starten.

Hinweis

Wenn in diesem Abschnitt einfach von Windows NT die Rede ist, sind die Windows-Versionen NT 4.0, 2000 und XP gemeint. Das Bootsystem dieser Windows-Versionen ist zum Glück identisch.

LILO ist nicht in der Lage, Windows NT selbst zu starten, wenn der von NT vorgegebene Bootsektor durch LILO überschrieben wird. Dieser Abschnitt beschreibt daher den umgekehrten Weg: Der Bootmanager von Windows NT bleibt, wo er ist, er wird aber mit einem zusätzlichen Menüeintrag ausgestattet, um LILO zu starten. Beim Rechnerstart können Sie sich also bequem zwischen NT und LILO entscheiden. (Innerhalb von LILO stehen dann unter Umständen abermals mehrere Optionen zur Auswahl, also z. B. verschiedene Linux-Kernel.)

Meiner Meinung nach ist das die bei weitem eleganteste Art, Linux auf einem Windows-NT-System zu booten, weil auch bei Windows-Neuinstallationen und -Updates (Service Packs etc.) keine Probleme zu erwarten sind.

Hinweis

Der NT-Bootmanager und die dazugehörigen Dateien werden grundsätzlich in die erste Partition der ersten Festplatte installiert. Dieser Abschnitt geht davon aus, dass es sich bei dieser Partition um eine Windows-Partition (3.1/9x/ME) handelt, nicht aber um eine NTFS-Partition.

Die hier beschriebene Vorgehensweise funktioniert selbst dann, wenn die erste Partition eine NTFS-Partition ist. Die Einrichtung von LILO wird dadurch aber etwas umständlicher, weil NTFS-Partitionen unter Linux nicht verändert werden dürfen. (Der NTFS-Treiber sieht zwar einen Read-write-Modus vor, dieser hat aber experimentellen Charakter und sollte nicht verwendet werden.) Sie müssen daher die unten beschriebene Datei `bootsec.lin` auf eine Diskette kopieren, den Rechner unter NT neu starten und dann `bootsec.lin` in die NT-Partition kopieren und `BOOT.INI` verändern.

Tipp

Wenn Sie den NT-Bootmanager versehentlich durch LILO überschrieben haben, können Sie mit `lilo -u` versuchen, den Bootsektor wiederherzustellen. Ist das nicht möglich, müssen Sie den Bootsektor mit dem Installationsprogramm bzw. mit einer Emergency-Diskette von Windows NT wiederherstellen. Aber selbst wenn Sie die erforderlichen Disketten haben, ist das eine mühsame Angelegenheit.

LILO-Konfiguration

Der erste Schritt besteht darin, dass Sie den Bootsektor der Linux-Root-Partition Ihrer Festplatte mit `dd` in eine Datei kopieren. Diese Partition können Sie mit `rdev` ermitteln (im Beispiel unten `/dev/hda8`). Passen Sie bei der Eingabe des `dd`-Kommandos auf! Mit falschen Parametern kann `dd` eine Menge Schaden anrichten!

```
root# rdev
/dev/hda8 /
root# dd if=/dev/hda8 bs=512 count=1 of=/boot/bootsec.lin
```

Nun ändern Sie `/etc/lilo.conf` so, dass `lilo` nicht den Bootsektor einer Festplatte oder Diskette verändert, sondern direkt die oben angegebene Datei `/boot/bootsec.lin`. Die restlichen LILO-Einstellungen erfolgen wie bei der Installation von LILO in den MBR der Festplatte (siehe Seite 7 bzw. Seite 18). Wenn Sie das nächste Mal `lilo` ausführen, wird also nur die Datei `bootsec.lin` verändert.

```
# in /etc/lilo.conf
boot=/boot/bootsec.lin
# ... alle anderen Einstellungen wie bisher
```

Der nächste Schritt besteht darin, dem NT-Startprogramm `NTLDR` beizubringen, dass es neben diversen Microsoft-Produkten auch noch Linux gibt. Sämtliche für den Startprozess erforderlichen Dateien befinden sich in der ersten Partition der ersten Festplatte (ganz unabhängig davon, auf welcher Festplatte bzw. Partition NT selbst installiert ist). `bootsec.lin` muss daher ebenfalls in diese Partition kopiert werden.

Wenn es sich bei dieser ersten Partition um eine FAT-Partition handelt (also um ein herkömmliches DOS/Windows-Dateisystem), können Sie diese Partition mit `mount` in das Dateisystem einbinden und `bootsec.lin` einfach in dessen Wurzelverzeichnis kopieren. Wenn es sich dagegen um eine NTFS-Partition handelt, müssen Sie `bootsec.lin` auf eine Diskette kopieren, NT starten und die Datei unter NT von der Diskette in die NTFS-Partition kopieren.

Jetzt müssen Sie nur noch die `NTLDR`-Konfigurationsdatei `BOOT.INI` ändern (erstellen Sie vorher ein Backup!): Diese Datei befindet sich ebenfalls im Wurzelverzeichnis der ersten Partition. Ein Beispiel für diese Datei ist unten abgedruckt. (Im Detail kann die Datei je nach Ihrer Hardware-Konfiguration ein wenig anders aussehen. Lange Zeilen sind hier aus Platzgründen auf zwei Zeilen verteilt, als Trennzeichen wurde `\` verwendet. In `BOOT.INI` müssen diese Zeilen allerdings zusammenbleiben!)

```
[boot loader]
timeout=60
default=multi(0)disk(0)rdisk(1)partition(2)\WINNT4
[operating systems]
multi(0)disk(0)rdisk(1)partition(2)\WINNT=\
"Microsoft Windows 2000 Professional" /fastdetect
C:\="Microsoft Windows"
```

An das Ende dieser Datei fügen Sie nun noch eine weitere Zeile an, nämlich:

```
C:\bootsec.lin="LILLO"
```

Falls Sie diese Änderung unter Linux durchführen, müssen Sie auf die korrekten Zeilentrennzeichen achten. (Unter DOS/Windows ist ja ein zusätzliches Ctrl-M-Zeichen am Zeilenende üblich. Die Eingabe dieses Zeichens ist mit den meisten Unix-Editoren umständlich (Emacs: (Strg)+(Q), (Strg)+(M)). Am einfachsten kopieren Sie eine beliebige Zeile und ändern diese. Beim Kopieren bleibt Ctrl-M am Zeilenende erhalten.)

Wenn Sie Ihren Rechner jetzt neu starten, wird LILO als zusätzliche Zeile neben den bisher schon vorhandenen Betriebssystemen angeführt. Wenn Sie diese Option wählen, startet NTLDR den LILO. Dort bestehen dann alle Möglichkeiten von LILO, d. h. je nach Konfiguration können Sie sich jetzt noch zwischen verschiedenen Linux-Kernen entscheiden.

Tipp

Noch mehr Informationen zum Thema LILO und Windows NT finden Sie im Linux+NT-Loader-Mini-HOWTO. Bei der Veränderung von `BOOT.INI` können Sie sich auch von dem Windows-Programm `BOOTPART.EXE` helfen lassen. Weitere Informationen finden Sie unter:

<http://www.winimage.com/bootpart.htm>

1.11 LILO-Fehlermeldungen

Sollte der Start von Linux nicht gelingen, gibt LILO fast immer zumindest einen Hinweis, was die Ursache des Fehlers sein könnte. LILO zeigt während eines vierteiligen internen Startprozesses der Reihe nach die vier Buchstaben 'LILO' an. Gibt es während des Startprozesses Probleme, erscheinen nicht alle Buchstaben – und das erlaubt Rückschlüsse auf die Ursache des Problems. Außerdem zeigt LILO unter Umständen eine Fehlernummer an.

Die folgende Liste gibt zu beiden Varianten einige weitere Informationen. Im Detail sind die Fehlercodes im LILO-Benutzerhandbuch beschrieben (üblicherweise in der Datei `/usr/doc/packages/lilo/user.dvi`).

- **Kein Buchstabe von 'LILO':** LILO wurde vermutlich gar nicht installiert (oder nicht dorthin, wo Sie gedacht hatten).
- **L:** Der erste Teil von LILO konnte geladen werden, nicht aber der zweite. Wahrscheinliche Fehlerursache: LILO hat Probleme, die Festplattegeometrie richtig zu interpretieren. Mögliche Abhilfe: Fügen Sie die Option `linear` in `lilo.conf` ein oder geben Sie die Festplattegeometrie mit `sectors`, `heads` und `cylinders` explizit an.
- **LI:** Auch der zweite Teil von LILO konnte geladen werden, aber beim Ausführen traten Probleme auf. Wahrscheinliche Fehlerursache: abermals Probleme mit der

Festplattengeometrie oder `/boot/boot.b` konnte nicht gefunden werden. Wurde die Datei nach der Konfiguration von LILO nochmals verändert? Hat sich die Reihenfolge der Festplatten geändert (z. B. Einbau von `hdb` zwischen `hda` und `hdc`)? Befinden sich Teile der Kernel-Datei außerhalb des 1024-Zylinder-Limits? Abhilfe: Fügen Sie die Option `lba32` in `lilo.conf` ein. Bei einem System, das sowohl SCSI- als auch IDE-Laufwerke enthält, sollten Sie die Optionen `disk` und `bios` ausprobieren. Wenn das nichts hilft, orientieren Sie sich an den obigen Tipps (Fehlermeldung 'L').

- **LIL:** LILO konnte gestartet werden, hat aber Probleme beim Lesen von `/boot/map`. Mögliche Fehlerursache/Behebung: Wie oben, aber für `/boot/map`.
- **LIL?:** Probleme mit `/boot/boot.b`. Abhilfe: Führen Sie `lilo` nochmals aus.
- **LIL-:** Probleme mit `/boot/map`. Abhilfe: Führen Sie `lilo` nochmals aus.
- **LIL0:** LILO konnte erfolgreich gestartet werden und hat alle Konfigurationsdateien gefunden.
- **Fehlercode 00:** Problem beim Lesen der Sektoren der Kernel-Datei. Mögliche Ursachen: Die Kernel-Datei wurde nach der Installation von LILO verändert (z. B. Neukompilierung), ohne `lilo` neu auszuführen. Oder Sie haben die Option `linear` verwendet und dabei das 1024-Zylinder-Limit überschritten. (Wenn diese Option verwendet wird, erkennt LILO die Überschreitung dieses Limits bei der Installation unter Umständen nicht.) Abhilfe: Kopieren Sie die Kernel-Datei in eine Partition, die vollständig unterhalb der 1024-Zylinder-Grenze liegt, und installieren Sie LILO neu.
- **Fehlercode 01:** Auch wenn das LILO-Benutzerhandbuch behauptet, dass dieser Fehler gar nicht auftreten sollte, ist er einer der häufigsten. Normalerweise erscheint eine endlose Reihe von 01-Codes, bis der Rechner neu gestartet wird. LILO teilt damit mit, dass ein oder auch viele unzulässige Kommandos ausgeführt wurden.

Eine Ursache kann darin bestehen, dass LILO die Festplatte nicht findet, auf der sich die Kernel-Datei befindet (insbesondere, wenn sich der Kernel nicht auf der ersten Festplatte befindet). In diesem Fall kann die explizite Angabe der Festplattennummer durch die `bios`-Option weiterhelfen (z. B. `bios=0x81` für die zweite Platte).

Einmal ließ sich das Problem lösen, indem die Option `message=...` auskommentiert wurde, die eine Linux-Distribution per Default vorgesehen hatte. (Auf den Begrüßungstext kann man gern verzichten.) Aber auch alle anderen oben schon erwähnten Tipps (Option `lilo`, explizite Angabe der Festplattengeometrie etc.) können vielleicht helfen.

- **Fehlercode 02:** 'Address mark not found': Eine mögliche Ursache ist eine defekte Diskette. Installieren Sie LILO auf eine neue Diskette.
- **Fehlercode 04:** 'Sector not found': Abermals ist ein Geometrieproblem die wahrscheinlichste Ursache. Abhilfe wie oben. Falls Sie `compact` verwenden, kommentieren Sie diese Option aus.

Tipp

LILO kennt noch eine Reihe weiterer Fehlercodes. Hier wurden nur die häufigsten dokumentiert. Eine vollständige Liste finden Sie in der schon erwähnten LILO-Dokumentation (Datei `user.dvi` bzw. `user.ps`).

1.12 Default-Betriebssystem für einen Reboot einstellen

Wenn Sie Ihren Rechner herunterfahren (z. B. durch `(Strg)+(Alt)+(Entf)`), wird beim nächsten Start wieder LILO ausgeführt. Manchmal wissen Sie beim Herunterfahren, wie Sie den Rechner neu starten möchten – z. B. abermals mit Linux oder mit Windows. In diesem Fall können Sie vor dem Shutdown `lilo -R name` ausführen. Damit erreichen Sie, dass das mit *name* bezeichnete Betriebssystem beim nächsten LILO-Start automatisch (ohne Benutzereingabe) gestartet wird. Der angegebene Name muss einer `label`-Zeichenkette in `/etc/lilo.conf` entsprechen.

Die zwei folgenden Kommandos bewirken, dass der Rechner unter Windows neu gestartet wird (sofern es in `lilo.conf` eine Bootvariante mit dem Namen `windows` gibt):

```
root# lilo -R windows
root# shutdown -R now
```

Kurz zu den Interna: Das Defaultsystem wird in der Mapping-Datei gespeichert (üblicherweise also in `/boot/map`). Unmittelbar nachdem LILO die Zeichenkette dort beim nächsten Start gelesen hat, wird die Zeichenkette gelöscht. Damit wird sichergestellt, dass die Einstellung nur einmal gilt. (Das ist eine Sicherheitsmaßnahme. Wenn das durch *name* angegebene Betriebssystem sich aus irgendeinem Grund nicht starten lässt, so funktioniert LILO nach dem erfolglosen Startversuch wieder normal, reagiert also wieder auf Benutzereingaben.)

`lilo -R` kann dazu eingesetzt werden, um den Komfort bei einem Rechnerneustart zu vergrößern. Insbesondere der KDE-Display-Manager `kdm` benutzt dieses Merkmal (siehe auch Seite ??).

1.13 Menüs und Grafikmodus

Bis jetzt ist es in diesem Abschnitt nur darum gegangen, LILO zum Laufen zu bringen. Wenn das geklappt hat und Sie noch Lust auf weitere Experimente haben, können Sie nun versuchen, LILO mit einer Begrüßungsgrafik zu verschönern und die Bedienung mit einem Menü zu erleichtern.

LILO-Menü im Textmodus

Ein Menü zur Auswahl des Betriebssystems, das mit den Cursortasten bedient werden kann, ist einfach zu bewerkstelligen. Sie müssen lediglich statt `/boot/boot.b` die Datei `/boot/boot-menu.b` verwenden. Als Menütexe werden die `label`-Texte der einzelnen Betriebssysteme verwendet.

Die Überschrift des Menüs sowie die Farbgestaltung können mit den `lilo.conf`-Optionen `menu-title` und `menu-scheme` eingestellt werden (Details siehe man `lilo.conf`). Beim folgenden Beispiel erscheint das LILO-Menü in einer weißen Box. Der Text wird schwarz angezeigt, das ausgewählte Betriebssystem rot, die Überschrift und der Rahmen um das Menü blau.

```
# in /etc/lilo.conf
install=/boot/boot-menu.b
menu-title="Wählen Sie ein Betriebssystem aus!"
menu-scheme=kw:Wr:bw:bw
... wie bisher
```

LILO-Menü im Grafikmodus

Hier gibt es zurzeit zwei Ansätze (Mandrake und SUSE), es ist noch nicht abzusehen, ob sich einer davon als Standard durchsetzen wird. Leider sind beide Varianten gleichermaßen schlecht dokumentiert. Zudem ist die Konfiguration sehr aufwändig, sodass sich der Aufwand wohl selten lohnt.

Der gemeinsame Nenner beider Varianten besteht darin, dass eine speziell präparierte Datei mit der `lilo.conf`-Option `message` angegeben wird.

Mandrake 8 verwendet LILO 21.7 mit `boot-graphic.b`. Die Datei mit dem Hintergrundbild muss mit dem Perl-Script `bmp2mdk` aus einer BMP-Datei erstellt werden. Dieses Script wird mit dem LILO-Paket mitgeliefert. Sie finden es samt der Datei `README.graphic` im Verzeichnis `/usr/share/doc/lilo-0.21.7`.

```
# in /etc/lilo.conf
install=/boot/boot-graphic.b
message=/boot/bmp2mdk-file
... wie bisher
```

Weitere Informationen zur Mandrake-LILO-Konfiguration finden Sie hier:

<http://liquid2k.com/matthias/linux/lilo.html>

SUSE 7.2 verwendet noch LILO 21.6 und setzt `boot-menu.b` ein. Eine für LILO geeignete Grafik wird mit `mklilomsg` erzeugt und ebenfalls via `message=file` übergeben. `mklilomsg` erwartet die Ausgangsdatei im PCX-Format. Die PCX-Datei muss bereits alle Menütexe enthalten! Die Menütexe für die verschiedenen Betriebssysteme können bei dieser Variante also nicht mehr mit der LILO-Option `label` verändert

<http://www.kofler.cc/linux.html>

werden. (Die genaue Position des Menüs übergeben Sie mit unzähligen Parametern an `mk1lilomsg`.) Ein weiterer Unterschied zur Mandrake-Lösung besteht darin, dass der VGA-Mode hier über die LILO-Option `vga` eingestellt wird.

```
# in /etc/lilo.conf
vga      = 771
install=/boot/boot-menu.b
message = /boot/mk1lilomsg-file
... wie bisher
```

Weitere Informationen zur SUSE-LILO-Konfiguration finden Sie hier:

http://sdb.suse.de/de/sdb/html/jkoeke_bootgrafik.html
<http://www.13thfloor.at/Software/lilo-splash/Example/>

1.14 LILO-Konfigurationshilfen

Anstatt `/etc/linux.conf` selbst zu erstellen, können Sie dazu diverse Hilfsprogramme verwenden. In der Vergangenheit hatten diese Tools aber immer Probleme mit den in diesem Kapitel beschriebenen Sonderfällen (SCSI-Systeme, 1024-Zylinder-Limit, Windows NT/2000), weswegen ich meine LILO-Konfigurationsdateien nach wie vor am liebsten selbst editiere.

Distributionspezifische Werkzeuge: Die folgende Liste zählt einige Werkzeuge zur Erstellung von Bootdisketten und zur LILO-Installation auf.

Mandrake: `drakfloppy`, `drakboot`, `mkbootdisk` (siehe Seite ??)
Red Hat: `mkbootdisk` (siehe Seite ??)
SUSE: YaST (siehe Seite ??)

Linuxconf: Bei Linuxconf ist die Konfiguration von `lilo.conf` über mehrere Dialoge verteilt (Startpunkt: `BOOT MODE|LILO`). Als Installationsort kann eine beliebige Partition, der MBR der Festplatte (geben Sie `/dev/hda` bzw. bei SCSI-Systemen `/dev/sda an`) oder eine Diskette ausgewählt werden. Das Ergebnis ist in letzterem Fall eine LILO-Bootdiskette ohne Kernel auf der Diskette.

KLILO (KDE): Bei manchen Distributionen wird `k1lilo` als weitere Alternative mitgeliefert. Das Programm wurde aber in letzter Zeit nicht mehr gewartet.