Linux Software-RAID HOWTO

Niels Happel (happel@resultings.de)

v0.2.32, 3. Februar 2002

Diese HOWTO beschreibt die Benutzung der RAID-Kernelerweiterungen, welche unter Linux den Linear Modus, RAID-0, 1, 4 und 5 als Software-RAID implementieren.

Inhaltsverzeichnis

1 Einführung

Ziel dieses Dokumentes ist es, das grundlegende Verständnis der unterschiedlichen RAID-Möglichkeiten und das Erstellen von RAID-Verbunden anhand der - teilweise - neuen Möglichkeiten des 2.2er Kernels zu erklären. Desweiteren wird auf die Besonderheiten mehrerer RAID-Verbunde, die Nutzung dieser als Root-Partition und deren Verhalten bei Fehlern eingegangen. Zu guter Letzt finden Sie noch einige Tips & Tricks rund um Linux allgemein sowie Software-RAID im speziellen.

1.1 Warnung

Dieses Dokument beinhaltet *keine Garantie* für das Gelingen der hier beschriebenen Sachverhalte. Obwohl alle Anstrengungen unternommen wurden, um die Genauigkeit der hier dokumentierten Informationen sicherzustellen, übernimmt der Autor *keine Verantwortung* für Fehler jeglicher Art oder für irgendwelche Schäden, welche direkt oder als Konsequenz durch die Benutzung der hier dokumentierten Informationen hervorgerufen werden.

RAID, obwohl es dafür entwickelt wurde, die Zuverlässigkeit des Systems zu steigern, indem es Redundanz gewährleistet, kann auch zu einem falschen Gefühl der Sicherheit führen, wenn es unsachgemäß benutzt wird. Dieses falsche Vertrauen kann dann zu wesentlich größeren Disastern führen. Im einzelnen sollte man beachten, daß RAID konstruiert wurde, um vor Festplattenfehlern zu schützen und nicht vor Stromunterbrechungen oder Benutzerfehlern. Stromunterbrechungen, instabile Entwicklerkernel oder Benutzerfehler können zu unwiederbringlichen Datenverlusten führen! RAID ist kein Ersatz für ein Backup Ihres Systems.

1.2 Begrifflichkeiten

Auf den folgenden Seiten werden Sie mit vielen Ausdrücken rund um Software-RAID, Festplatten, Partitionen, Tools, Patches und Devices bombardiert. Um als RAID-Einsteiger mit den oft gebrauchten Ausdrücken nicht ins Schleudern zu geraten, erhalten Sie hier eine Einführung in die Begrifflichkeiten.

Chunk-Size

Eine genaue Beschreibung, was die Chunk-Size ist, ist im Abschnitt 8 (Spezielle Optionen der RAID-Devices) zu finden.

Devices

Devices sind unter Linux Stellvertreter für Geräte aller Art, um sie »beim Namen« nennen zu können. Sie liegen alle unter `/dev/` in Ihrem Linux-Verzeichnisbaum. Beispiel dafür sind `/dev/hda` für die erste (E)IDE-Festplatte im System (analog `/dev/hdb`, `/dev/hdc`), `/dev/sda` für die erste SCSI-Festplatte oder `/dev/fd0` für das erste Diskettenlaufwerk.

Festplatten

Festplatten sollten Ihnen bekannt sein. RAID nutzt mehrere Festplatten, um entweder deren Gesamtgeschwindigkeit, deren Sicherheit oder beides zu erhöhen.

Hot Plugging

Hot Plugging bezeichnet die Möglichkeit, einzelne Festplatten ohne ein Abschalten oder Herunterfahren des Rechners innerhalb eines redundanten RAID-Verbundes abzuschalten, auszutauschen und wieder einzufügen. Im günstigsten Fall bietet einem Hot Plugging also die Möglichkeit, eine defekte Festplatte auszutauschen, ohne den Rechner neu starten zu müssen und ohne die Verfügbarkeit Ihres Servers zu beeinträchtigen. Für die Benutzer würde ein Austausch einer defekten Festplatte absolut transparent erfolgen.

MD-Device

MD steht für Multiple-Disk oder Multiple-Device und bedeutet dasselbe wie ein RAID-Device. Um Sie nicht weiter in die Irre zu führen, wird im folgenden auf die Bezeichnung MD-Device bewußt verzichtet.

MD-Tools

Die MD-Tools sind Hilfsprogramme, die Ihnen die Möglichkeit zur Einrichtung oder Änderung von RAID-Verbunden zur Verfügung stellen, welche mit denen im Standardkernel Version 2.0.x oder 2.2.x enthaltenen RAID-Treibern erstellt wurden. Sie beinhalten als MD-Tools bis einschließlich Version 0.4x für ältere RAID-Verbunde dieselbe Grundfunktionalität wie die RAID-Tools Version 0.9x für die aktuellen RAID-Treiber. Ihre Entwicklung ist zwar eingestellt worden, dennoch sind sie auf vielen älteren Linux-Distributionen vertreten. Die einzelnen Programme dieses MD-Tools Paketes zur Verwaltung von älteren RAID-Verbunden der Version 0.4x werden im entsprechenden Abschnitten abgehandelt.

md0

`/dev/md0` ist ein Stellvertreter für den erste RAID-Verbund in Ihrem System. Das Verzeichnis `/dev/` zeigt an, daß es sich um ein Device handelt, `md` meint ein Multiple-Disk oder Multiple-Device und damit einen Verbund aus mehreren Partitionen Ihrer Festplatte(n). Das erste Device jeglicher Art ist immer entweder mit einer 0 gekennzeichnet und wird weiter aufsteigend nummeriert (also `/dev/md0`, `/dev/md1`, usw.) oder beginnt mit `/dev/hda` und wird alphabetisch aufsteigend durchgezählt (`/dev/hdb`, `/dev/hdc`, usw.).

Partitionen

Partitionen bezeichnen die Einteilung Ihrer Festplatte in mehrere Segmente. RAID-Verbunde können aus mehreren Partitionen derselben Festplatte oder aus mehrere Partitionen verschiedener Festplatten bestehen.

Persistent-Superblock

Eine Beschreibung, was ein Persistent-Superblock ist, ist im Abschnitt 9 (Weitere Optionen des neuen RAID-Patches) nachzulesen.

RAID-Device

RAID-Device ist die Bezeichnung für einen neu erstellten RAID-Verbund, der jetzt unter einem eigenen Namen anzusprechen ist. Unter Linux entspricht jedes Gerät letztendlich einem Device. Die RAID-Devices sind im Linux-Verzeichnisbaum unter `/dev/` abgelegt und heißen `md0-15`.

RAID-Partition

RAID-Partition bezeichnet eine einzelne Festplattenpartition, die für die Verwendung in einem RAID-Verbund genutzt werden soll.

RAID-Patch

Der RAID-Patch bezeichnet ein Paket aktueller RAID-Treiber, die neuer als die im Standardkernel enthaltenen sind. Sie weisen in ihrem Namen eine Zeichenfolge auf, die sich mit der von Ihnen verwendeten Kernel-Version decken sollte. Z.B. braucht der Kernel 2.2.10 den RAID-Patch für den 2.2.10er Kernel. Dieser RAID-Patch aktualisiert also im Endeffekt die originalen RAID-Treiber in Ihrem Kernel-Sourcetree.

RAID-Tools

Die RAID-Tools sind Hilfsprogramme, um den Umgang mit RAID-Verbunden in Form von Einrichtung, Wartung und Änderung überhaupt zu ermöglichen. Sie existieren für die im Standardkernel 2.0.x und 2.2.x vorhandenen RAID-Treiber als MD-Tools in der Version 0.4x, für die aktuellen RAID-Treiber als RAID-Tools mit der Versionsnummer 0.9x. Der Funktionsumfang der RAID-Tools Version 0.9x überwiegt gegenüber dem der MD-Tools und erlaubt einen einfacheren Umgang mit den RAID-Systemen. Sie sollten bei der Verwendung der RAID-Tools prüfen, ob sie auch die passende Version haben.

RAID-Verbund

RAID-Verbund, oder synonym RAID-Array ist die Bezeichnung für eine Zusammenfügung von mehrerer Partitionen einzelner Festplatten, um sie nachher als eine Einheit ansprechen zu können. Ein anschauliches Beispiel wären zwei Partitionen jeweils einer Festplatte, welche als eine komplett neue Partition und damit als ein neuer RAID-Verbund zur Verfügung stehen würden. Über den Stellvertreter `/dev/md0` würde dieser RAID-Verbund auf Linux Ebene als ein RAID-Device angesprochen werden, der als eine einzelne Zuweisung für zwei darunterliegende Partitionen fungiert.

Redundanz

Redundanz kommt aus dem Lateinischen, bedeutet Überfülle und meint auf einen RAID-Verbund bezogen das Vorhandensein zusätzlicher Kapazitäten, die keine neuen Daten enthalten, also speziell das ein Datenträger ausfallen kann, ohne den vorhandenen Datenbestand zu beeinträchtigen.

Spare-Disk

Informationen hierzu sind im Abschnitt 9 (Weitere Optionen des neuen RAID-Patches) zu finden.

1.3 Literatur

Obwohl in dieser HOWTO alles nötige Wissen zum Erstellen von Software-RAID Verbunden unter Linux vermittelt werden sollte, sei hier trotzdem zum besseren Verständnis und als weiterführende Texte auf folgende HOWTOs und entsprechende Fachliteratur verwiesen:

- *BootPrompt HOWTO*
- *Kernel HOWTO*
- *Root-RAID HOWTO*
- *Software-RAID mini-HOWTO*
- *The Software RAID HOWTO*

1.4 Wer hat dieses Dokument zu verantworten?

Niels Happel hat zwar die Informationen zusammengetragen, getestet und neu geschrieben, jedoch beruht der größte Teil des Erfolgs dieser HOWTO natürlich auf der Arbeit der Programmierer der RAID-Kernelerweiterungen. Mein besonderer Dank gilt denen, die mir teilweise mit vielen Anregungen, Tips, Texten und Unermüdlichkeit weitergeholfen haben und allen anderen, die mich dahingehend unterstützt haben. Die fleißigsten waren:

- Uwe Beck (`ubeck@debis.com`)
- Robert Dahlem (`robert.dahlem@gmx.net`)
- Ulrich Herbst (`ulrich.herbst@debis.com`)
- Werner Modenbach (`modenbach@alc.de`)

1.5 Copyright

Dieses Dokument ist urheberrechtlich geschützt. Das Copyright für dieses Dokument liegt bei Niels Happel und Marco Budde.

Das Dokument darf gemäß der GNU *General Public License* verbreitet werden. Insbesondere bedeutet dieses, daß der Text sowohl über elektronische wie auch physikalische Medien ohne die Zahlung von Lizenzgebühren verbreitet werden darf, solange dieser Copyright-Hinweis nicht entfernt wird. Eine kommerzielle Verbreitung ist erlaubt und ausdrücklich erwünscht. Bei einer Publikation in Papierform ist das Deutsche Linux HOWTO Projekt hierüber zu informieren.

1.6 Info

Wenn Sie irgendwelche Ideen zu dieser HOWTO haben, positive wie negative Kritiken, Korrekturen, oder Wünsche für die nächste Version ... eine E-Mail ist wirklich schnell geschrieben.

Niels Happel
E-Mail: `happel@resultings.de`

Die jeweils aktuellste Version finden Sie unter:

- `http://www.resultings.de/linux/`
- `http://www.tu-harburg.de/dlhp/`

2 Was bedeutet RAID?

RAID steht entweder für »Redundant Array of Independent Disks« oder für »Redundant Array of Inexpensive Disks« und bezeichnet eine Technik, um mehrere Partitionen miteinander zu verbinden. Die beiden verschiedenen Bezeichnungen und teilweise auch die Grundidee von RAID-Systemen rühren daher, daß Festplattenkapazität zu besitzen früher eine durchaus enorm kostspielige Angelegenheit werden konnte. Kleine Festplatten waren recht günstig, große Festplatten aber unangemessen teuer. Man suchte also eine Möglichkeit, mehrere Festplatten kostengünstig zu einer großen zu verbinden. Aufgrund der plötzlichen sprunghaften Erweiterung der erreichbaren Festplattenkapazitäten und der unaufhaltsam sinkenden Preise pro Megabyte Festplattenplatz hat sich das Erreichen des damaligen wichtigsten Zieles eines RAID-Verbundes von selbst erledigt. Dennoch war man gewillt, den bis dahin erzielten Entwicklungsstand eines RAID-Systems auch anderweitig zu nutzen.

Das Ziel eines RAID-Verbundes ist heutzutage deshalb entweder eine Performancesteigerung, eine Steigerung der Datensicherheit oder eine Kombination aus beidem. RAID kann vor Festplattenfehlern schützen und kann auch die Gesamtleistung im Gegensatz zu einzelnen Festplatten steigern.

Dieses Dokument ist eine Anleitung zur Benutzung der Linux RAID-Kernelerweiterungen und den dazugehörigen Programmen. Die RAID-Erweiterungen implementieren den Linear (Append) Modus, RAID-0 (Striping), RAID-1 (Mirroring), RAID-4 (Striping & Dedicated Parity) und RAID-5 (Striping & Distributed Parity) als Software-RAID. Daher braucht man mit Linux Software RAID keinen speziellen Hardware- oder Festplattenkontroller, um viele Vorteile von RAID nutzen zu können. Manches läßt sich mit Hilfe der RAID-Kernelerweiterungen sogar flexibler lösen, als es mit Hardwarekontrollern möglich wäre.

Folgende RAID Typen werden unterschieden:

Linear (Append) Mode

Hierbei werden Partitionen unterschiedlicher Größe über mehrere Festplatten hinweg zu einer großen Partition zusammengefügt und linear beschrieben. Hier ist kein Geschwindigkeitsvorteil zu erwarten. Fällt eine Festplatte aus, so sind alle Daten verloren.

RAID-0 (Striping) Mode

Auch hier werden zwei oder mehr Partitionen zu einer großen zusammengefügt, allerdings erfolgt hier der Schreibzugriff nicht linear (erst die 1.Platte bis sie voll ist, dann die 2.Platte usw.), sondern parallel. Dadurch wird ein deutlicher Zuwachs der Datenrate insbesondere bei SCSI-Festplatten erzielt, welche sich für die Dauer des Schreibvorgangs kurzfristig vom SCSI Bus abmelden können und ihn somit für die nächste Festplatte freigeben. Die erzielten Geschwindigkeitsvorteile gehen allerdings zu Lasten der CPU Leistung. Bei einer Hardware RAID Lösung würde der Kontroller diese Arbeit übernehmen. Allerdings steht der Preis eines guten RAID Kontrollers in keinem Verhältnis zur verbrauchten CPU Leistung eines durchschnittlichen Computers mit Software-RAID.

RAID-1 (Mirroring) Mode

Der Mirroring Mode erlaubt es, eine Festplatte auf eine weitere gleichgroße Festplatte oder Partition zu duplizieren. Dieses Verfahren wird auch als Festplattenspiegelung bezeichnet. Hierdurch wird eine erhöhte Ausfallsicherheit erreicht - streckt die eine Festplatte die Flügel, funktioniert die andere noch. Allerdings ergibt das auch wieder nur Sinn, wenn die gespiegelten Partitionen auf unterschiedlichen Festplatten liegen. Die zur Verfügung stehende Festplattenkapazität wird durch dieses Verfahren halbiert. Ein Geschwindigkeitsgewinn ist hierbei nur beim Lesezugriff zu erwarten, jedoch erbringt der aktuelle Stand der RAID-Treiber für Linux nur beim nicht sequentiellen Lesen vom RAID-1 Geschwindigkeitsvorteile.

RAID-4 (Striping & Dedicated Parity) Mode

RAID-4 entspricht dem RAID-0 Verfahren, belegt allerdings eine zusätzliche Partition mit Paritätsinformationen, aus denen eine defekte Partition wieder hergestellt werden kann. Allerdings kostet diese Funktion wieder zusätzliche CPU Leistung.

RAID-5 (Striping & Distributed parity) Mode

Hier werden die Paritätsinformationen zum Restaurieren einer defekten Partition zusammen mit den tatsächlichen Daten über alle Partitionen verteilt. Allerdings erkauft man sich diese erhöhte Sicherheit durch einen Kapazitätsverlust. Will man 5x1 GB zu einem RAID-5 zusammenfassen, so bleiben für die eigentlichen Daten noch 4x1 GB Platz übrig. Beim Schreibvorgang auf einen RAID-5 Verbund wird erst ein Datenblock geschrieben, dann erfolgt die Berechnung der Paritätsinformationen, welche anschließend auch auf den RAID-Verbund geschrieben werden. Hierher rührt die schlechtere Schreibgeschwindigkeit der Daten. Der Lesevorgang ähnelt allerdings dem RAID-0 Verbund. Das Resultat ist deshalb eine Steigerung der Lesegeschwindigkeit im Gegensatz zu einer einzelnen Festplatte.

RAID-10 (Mirroring & Striping) Mode

RAID-10 bezeichnet keinen eigenständigen RAID-Modus, sondern ist ein Kombination aus RAID-0 und RAID-1. Hierbei werden zuerst zwei RAID-0 Verbunde erstellt, die dann mittels RAID-1 gespiegelt werden. Der Vorteil von einem RAID-10 im Gegensatz zu einem RAID-5 ergibt sich aus der höheren Performance. Während ein RAID-5 nur relativ wenig Geschwindigkeitsvorteile bietet, ist ein RAID-10 durch die beiden RAID-0 Verbunde für den Fall besser geeignet, wenn man sowohl Redundanz als auch einen hohen Geschwindigkeitsvorteil erzielen will. Sogar die anschließend notwendige RAID-1 Spiegelung bringt noch einen Vorteil bei der Lesegeschwindigkeit. Weiterhin fällt hierbei die notwendige Berechnung von Paritätsinformationen weg. Erkauft wird dies allerdings durch eine sehr viel schlechtere Nutzung des vorhandenen Festplattenplatzes, da immer nur 50% der tatsächlichen Kapazität mit den eigentlichen Daten beschrieben werden kann.

3 Voraussetzungen

3.1 Hardware

Zum Erstellen von RAID-Devices werden für den Linear, RAID-0 und RAID-1 Modus mindestens zwei leere Partitionen auf möglichst unterschiedlichen Festplatten benötigt. Für RAID-4 und RAID-5 sind mindestens drei Partitionen nötig und für den RAID-10 Modus vier. Dabei ist es egal, ob die Partitionen auf (E)IDE- oder SCSI-Festplatten liegen.

Will man Software-RAID mit (E)IDE-Festplatten benutzen, so empfiehlt es sich, jeweils nur eine (E)IDE-Festplatte an einem (E)IDE Kontroller zu benutzen. Im Gegensatz zu SCSI beherrschen (E)IDE-Festplatten keinen Disconnect - können sich also nicht vorübergehend vom BUS abmelden - und können dementsprechend nicht »parallel« angesprochen werden. An zwei unterschiedlichen (E)IDE Kontrollern ist dies jedoch in Grenzen möglich, wenn auch immer noch nicht so gut wie bei SCSI Kontrollern. Besser als an einem Strang ist es aber auf jeden Fall.

Desweiteren können Sie die Überlegung, Hot Plugging mit (E)IDE-Festplatten zu benutzen, gleich wieder ad acta legen. Hierbei gibt es mehr Probleme als Nutzen. Wie das jedoch mit eigenen (E)IDE RAID Kontrollern aussieht, kann ich mangels passender Hardware nicht sagen.

SCSI Kontroller werden entweder als PCI Steckkarte zusätzlich in den Computer eingebaut, oder sind in Form eines SCSI Chips bereits auf dem Mainboard integriert. Letztere so genannte »On Board Controller«

hatten bei älteren Mainboards die unangenehme Eigenart, die Interrupt Leitungen für den AGP Steckplatz und den SCSI Chip zusammenzulegen und waren so die Ursache für manche Konfigurations-, Stabilitäts- und daraus resultierend auch Geschwindigkeitsprobleme. Inzwischen funktionieren Mainboards mit integriertem SCSI Chip meist ebenso gut wie externe SCSI Kontroller und stehen in ihrer Leistung einander in nichts nach. Aktuelle Probleme können lediglich aus der mangelnden Treiberunterstützung der verwendeten Distribution resultieren und auf diese Weise eine Linux Installation erschweren.

Das Software-RAID unter Linux bietet zwar die Möglichkeit, einzelne Partitionen einer Festplatte in einen RAID-Verbund einzubauen, um jedoch einen nennenswerten Geschwindigkeitsvorteil oder entsprechende Redundanz zu erzielen, sind generell die RAID-Partitionen auf unterschiedliche Festplatten zu verteilen. Sinnvoll wäre es, das erste RAID zum Beispiel auf drei Partitionen unterschiedlicher Festplatten zu legen; `/dev/md0` bestünde dann z.B. aus `/dev/sda1`, `/dev/sdb1` und `/dev/sdc1`. Das zweite RAID würde man ebenso verteilen: `/dev/md1` wäre dann also ein Verbund aus `/dev/sda2`, `/dev/sdb2` und `/dev/sdc2`. Damit hat man zwei RAID-Verbunde (`/dev/md0` und `/dev/md1`), die sich - als einzelnes Device angesprochen - jeweils über alle drei Festplatten ziehen.

Da es so gut wie immer unsinnig ist, innerhalb eines RAID-Verbundes mehrere Partitionen auf dieselbe Festplatte zu legen (z.B. `/dev/md0` bestehend aus `/dev/sdb1`, `/dev/sdb2` und `/dev/sdc1`), kann oft der Begriff RAID-Festplatte mit RAID-Partition synonym verwendet werden.

3.2 Software

Die Prozedur wird hier zum einen mit Hilfe der alten RAID-Tools Version 0.4x anhand einer DLD 6.0 und DLD 6.01 beschrieben, zum anderen wird ein aktueller Installationsablauf erläutert, der sich auf den neuen RAID-Patch und die RAID-Tools Version 0.9x bezieht. Der neue Weg hält sich an den Kernel 2.2.10 in Verbindung mit dem passenden RAID-Patch und den RAID-Tools und sollte unabhängig von der von Ihnen benutzten Linux-Distribution funktionieren. Dieser Weg gilt vom Ablauf her auch für alle neueren Kernel der 2.2.xer Reihe. In die Kernel der 2.4.xer Reihe hat der neue RAID-Patch bereits Einzug in den Standard Kernel-Sourcetree erlangt, wodurch diese Kernel nicht mehr aktualisiert werden müssen.

Im allgemeinen ist, vor allem durch die Tatsache, daß die 2.4.xer Kernel nicht mehr gepatcht werden müssen, die Benutzung des neuen RAID-Patches dringend den alten MD-Tools vorzuziehen. Allein der unter Linux sonst unüblich große Versionssprung von 0.4x auf 0.9x zeigt die starken einhergegangenen Veränderungen. Allerdings kann auch das Verfahren mit Hilfe der MD-Tools für ältere Distributionen von Vorteil sein, in denen diese bereits vorkonfiguriert sind oder für eine Situation, in der bereits mit den älteren MD-Tools Version 0.4x eingerichtete RAID-Verbunde vorhanden sind.

Generell sei hier schon gesagt, daß die Lösung durch die MD-Tools und die mit dem aktuellen RAID-Patch unterschiedliche Wege gehen. Bitte beachten Sie das bei der Ihnen vorliegenden Distribution und entscheiden Sie sich frühzeitig für eine Variante. Glauben Sie, zusammen mit einer guten Anleitung mit `fdisk` und `patch` zurecht zu kommen, so wählen Sie ruhig den neuen Weg.

Für den aktuellen Abschnitt, welcher die Verwendung der neuen RAID-Kernelerweiterungen Version 0.9x beschreibt, werden neben einem lauffähigen Linux-System zwei Archive benötigt. Zum einen der zur Kernelversion passende RAID-Patch und zum anderen die aktuellsten RAID-Tools. Beide Archive gibt es im Internet:

```
ftp.kernel.org:/pub/linux/daemons/raid/alpha/
```

Aktuelle Distributionen, welche bereits auf einem 2.4.xer Kernel basieren, enthalten neben den aktuellen RAID-Treibern meistens auch ein RPM-Paket mit den aktuellen RAID-Tools. Im Falle einer RedHat 7.2 Linux Distribution heißt das entsprechende Paket `raidtools-0.90-23.i386.rpm` und ist bei einer Standardinstallation bereits eingerichtet. Der gesamte Einrichtungsteil zur Kernel-Aktualisierung fällt dann angenehmer Weise weg. Alles andere funktioniert allerdings genauso wie mit dem als Beispiel angeführten Kernel 2.2.10.

Generell aktualisiert der RAID-Patch Ihren vorhandenen Linux Kernel-Sourcetree wobei die RAID-Tools die zur Verwaltung von RAID-Verbunden benötigten Programme zur Verfügung stellen. Die einzelnen Programme oder Kommandos der aktuellen RAID-Tools vom 24.08.1999 werden hier mit einer kurzen Erläuterung zu Ihrem Verwendungszweck beschrieben:

ckraid

Dieses Programm testete in älteren Software-RAID Versionen, die noch mit den MD-Tools Version 0.4x erstellt wurden, die Konsistenz eines RAID-Verbundes. Mit dem aktuellen Kernel-Patch übernimmt der Linux-Kernel diese Arbeit und behandelt die RAID-Verbunde genauso wie alle anderen Partitionen. Dieses Programm ist aus Gründen der Rückwärtskompatibilität noch vorhanden.

mkraid

Dies ist das zentrale Verwaltungsprogramm, um RAID-Verbunde aller RAID-Modi anhand einer Konfigurationsdatei - meist `/etc/raidtab` - zu initialisieren, erstellen oder upzugraden.

raid0run

Aus Gründen der Rückwärtskompatibilität kann man mit Hilfe dieses Kommandos Linear und RAID-0 Verbunde starten, welche noch mit den alten MD-Tools Version 0.4x erstellt wurden.

raidhotadd

Hiermit wird das sogenannte Hot Plugging, in diesem Fall das Hinzufügen einer RAID-Partition in einen laufenden RAID-Verbund, ermöglicht.

raidhotremove

Dies ermöglicht in Analogie zum Kommando `raidhotadd` das Entfernen einer RAID-Partition aus einem aktiven RAID-Verbund.

raidsetfaulty

Um `raidhotremove` z.B. auf ein laufendes RAID-Array mit 9 (Spare-Disks) anwenden zu können, muß zuerst die zu entnehmende Festplatte als defekt markiert werden. Ist dies nicht von alleine korrekt geschehen, muß das Kommando `raidsetfaulty` dazu bemüht werden. Andernfalls erhält man von `raidhotremove` lediglich eine Fehlermeldung.

raidstart

Ist ein RAID-Verbund erst einmal initialisiert, kann er mit diesem Programm gestartet werden. Durch den neuen RAID-Patch und mit den entsprechenden Optionen in der `/etc/raidtab` kann dies allerdings der Kernel beim Startup des Rechners bereits automatisch erledigen.

raidstop

Erlaubt das Deaktivieren eines RAID-Verbundes, um z.B. den Rechner sicher herunterfahren zu können. Auch dies läßt sich mit den nötigen Einträgen in der `/etc/raidtab` automatisch durch den Kernel erledigen.

raidtab

Dies ist die zentrale Konfigurationsdatei für die gesamten RAID-Verbunde Ihres Systems, die erst neu erstellt werden muß. Die Parameter für die einzelnen RAID-Verbunde werden in den entsprechenden Kapiteln dieser HOWTO beschrieben. Standardmäßig suchen die RAID-Tools nach `/etc/raidtab`. Hier sollte also diese Datei auch erstmal mit diesem Namen erstellt werden.

4 Generelles zum Umgang mit Linux

4.1 Möglichkeiten des Bootens von Linux

Linux kann auf vielfältige Weise gebootet werden. Gerade im Umgang mit zu testenden RAID-Verbunden und spätestens bei dem Versuch das Root-Verzeichnis auf einen RAID-Verbund verlegen zu wollen, stellt sich einem die Frage, ob und wie Linux bei einem Mißerfolg wieder zum sauberen Startup zu bewegen ist. Je nach Zielvorstellung sind dafür unterschiedlich trickreiche Wege zu verfolgen. Diese zahlreichen Möglichkeiten sollen hier erläutert werden. Welche nun speziell für einen beschriebenen RAID-Verbund nötig ist, wird nochmals in den jeweiligen Kapiteln genannt.

4.1.1 Linux von Diskette booten

Zumindest als Notnagel sollte man sich eine Boot-Diskette mit einem aktuellen Kernel, der am besten noch alle nötigen Festplatten-Treiber und die RAID-Optionen fest einkompiliert hat, in eine sicher Ecke legen.

DOS Bootdiskette mit Loadlin

Den flexibelsten Weg im Umgang mit RAID-Verbunden bietet ein DOS-Bootdiskette. Benötigt werden hierfür lediglich die DOS Systemdateien, Loadlin, ein aktueller Linux-Kernel und ein kleiner DOS-Editor. Der nicht zu unterschätzende Vorteil dieser Lösung liegt darin, daß man die Konfigurationsdatei von `loadlin` ganz simpel an unterschiedliche Gegebenheiten anpassen kann, um zum Beispiel mal das Booten von einem RAID-Verbund zu testen. Klappt dies aus irgendeinem Grund nicht, können die Änderungen an der Startdatei einfach mittels des DOS-Editors wieder rückgängig gemacht werden.

Linux Bootdiskette mit LILO

Eine Linux Bootdiskette sollte jeder echte Linuxer immer haben. Am einfachsten werden diese über die distributionsspezifischen Verwaltungsprogramme erstellt. Eine weitere Möglichkeit bietet das Programm `mkbootdisk`. Zum Testen der Bootfähigkeit neuer RAID-Verbunde ist das ständige Ändern allerdings zu umständlich. Zum Ändern einer auf Basis des ext2 Dateisystems erstellten Bootdiskette benötigt man nämlich erstmal ein laufendes Linux-System.

4.1.2 Linux von der Festplatte booten

So normal sich das Booten von der Festplatte auch anhört, so treten doch gerade in Verbindung mit RAID-Verbunden als Root-Partition einige Probleme zu Tage die es hierbei zu umschiffen gilt. Andererseits ist es auch oft gerade die Vielfalt der Bootmöglichkeiten, welche den einen oder anderen in Verwirrung stürzt.

DOS Partition mit Loadlin

Ein relativ sicherer und einfacher Weg zugleich Linux zu Booten und schnell die Bootkonfiguration zwischen einem RAID-Verbund und einer normalen ext2-Partition zu wechseln, stellt das Booten per `loadlin` von einer kleinen DOS-Partition dar. Außer den DOS Systemdateien, einem Linux-Kernel mit RAID-Unterstützung, `loadlin` und einer Loadlin-Konfigurationsdatei wird nur noch ein DOS Editor benötigt, um simpel die Root-Partition in der Loadlin-Konfigurationsdatei zu ändern.

Extra-Partition für LILO mit Root-RAID

Root-RAID in Verbindung mit LILO braucht noch etwas mehr Fürsorge. Zuerst müssen Sie wissen, ob Ihr LILO im MBR Ihrer Festplatte oder im Superblock Ihrer Root-Partition installiert ist. Ist Linux z.B. das einzige Betriebsystem auf Ihrem Rechner, ist LILO vermutlich im MBR installiert, booten Sie jedoch mittels eines fremden Bootmanagers (OS/2 Bootmanager, XFDisk, oder ähnliche) wird LILO im Superblock Ihrer Root-Partition liegen. Noch einfacher kann das Ihre bisherige `/etc/lilo.conf` herausstellen: Der Parameter `boot=` gibt an, wo sich LILO aufhält. Steht dort etwa

```
boot = /dev/sda
```

so residiert Ihr LILO im MBR der ersten SCSI-Festplatte, bei der Angabe

```
boot = /dev/sda2
```

handelt es sich um den Superblock Ihrer zweiten primären Partition.

LILO braucht zum Booten die Information, wo der Linux-Kernel auf der Festplatte liegt. Da LILO das aber zu einer Zeit erfahren muß, zu der noch gar keine Partition gemountet ist, behilft sich LILO, indem er Plattengeometriedaten in den MBR oder Superblock schreibt, die die genaue Anfangslage des Linux-Kernel beschreiben. Die meisten Distributionen legen Ihre Kernel unter `/boot` ab. Diesen Umstand kann man nun dahingehend ausnutzen, daß man sich ein kleine Extra-Partition (etwa 10-20 MB) erstellt, welche unterhalb der 1024 Zylindergrenze liegt. Diese formatiert man mit ext2 und mountet sie als `/boot` in seinen Root-RAID-Device-Verzeichnisbaum, kopiert den gesamten Inhalt von dem originalen `/boot` Verzeichnis in das neue `/boot` Verzeichnis und ändert die Dateien `/etc/lilo.conf` und `/etc/fstab` dementsprechend:

/etc/lilo.conf

```
boot = boot-Partition-ohne-RAID (/dev/sda2),
       oder: MBR-der-Festplatte (/dev/sda)
image = /boot/vmlinuz-2.2.10
root = /dev/md0
read only
```

/etc/fstab

```
/dev/md0 / ext2 exec,dev,suid,rw 1 1
/dev/sda2 /boot ext2 exec,dev,suid,rw 1 1
```

Das Ausführen von `lilo` sollte dann bescheinigen, daß der Kernel `vmlinuz-2.2.10` korrekt initialisiert wurde.

Haben Sie nun LILO im Superblock Ihrer neuen `/boot` Partition angelegt, so müssen Sie dies noch Ihrem Bootmanager bekanntgeben und ihn eben diese booten lassen. Dem Beispiel zufolge wäre das die Partition `/dev/sda2`. Liegt Ihr LILO im MBR der Festplatte, so brauchen Sie nichts weiter tun, als neu zu booten.

Dieses Verfahren bootet zwar Linux mit einem Root-RAID, ist aber im Fehlerfall der ersten Festplatte nicht redundant!

Extra-Partition für LILO mit redundantem Root-RAID

Die hier beschriebene Vorgehensweise bezieht sich auf die folgende Konstellation: Zwei (E)IDE-Platten sind beide als Master gejumpert und hängen an verschiedenen (E)IDE Kontrollern: `/dev/hda` und `/dev/hdc`.

Die Partitionstabelle ist für beide Festplatten gleich:

```
/dev/hd?1   primary   Linux native (83)      ca. 10 MB (fuer /boot)
/dev/hd?2   primary   Linux swap (82)        128 MB (fuer swap)
/dev/hd?3   primary   Linux raid auto (fd)   den Rest (fuer /dev/md0)
```

Wenn im folgenden von »Backup-Fall« gesprochen wird, dann ist damit der Fall gemeint, daß die erste Festplatte ausgefallen ist und irgendwie von der verbliebenen zweiten Festplatte gebootet werden soll.

Wir gehen von folgender `/etc/lilo.conf` für die erste Festplatte aus:

```
boot=/dev/hda

image=/boot/vmlinuz
        root=/dev/md0
        label=linux
```

Nun muß auch auf der zweiten Platte eine Boot-Partition erzeugt werden. Dazu erstellt man auf der zweiten Festplatte eine identische Partition und kopiert mittels einer der im Abschnitt 4.2 (Möglichkeiten zum Kopieren von Daten) beschriebenen Methoden das originale Boot-Verzeichnis auf die zweite Festplatte.

Jetzt kopiert man die `/etc/lilo.conf` der zweiten Festplatte nach `/etc/lilo.conf.backup` und paßt sie an die neuen Bedingungen an. Die endgültige `/etc/lilo.conf.backup` sollte dann wie folgt aussehen:

```
boot=/dev/hdc
disk=/dev/hdc bios=0x80
map=/boot2/map
install=/boot2/boot.b

image=/boot2/vmlinuz
        root=/dev/md0
        label=linux
```

Der Parameter `disk=/dev/hdc bios=80` ist nötig, um LILO vorzuspiegeln, daß die Festplatte `/dev/hdc` mit 0x80 eingeloggt ist. Der Grund dafür ist, daß das BIOS normalerweise die ersten beiden Festplatten mit den Adressen 0x80 und 0x81 einloggt. Wir konfigurieren die Platte 0x81 (`/dev/hdc`). Im Backup-Fall wird die Festplatte aber als 0x80 eingeloggt, da die ursprüngliche erste Festplatte ja defekt ist.

Ein

```
lilo -C /etc/lilo.conf.backup
```

schreibt die Bootinformationen in den MBR. Es erscheint eine Warnung »/dev/hdc is not on the first disk«, aber das soll uns nicht stören, denn im Backup-Fall wird diese Festplatte ja zur ersten Festplatte im System. Dafür muß sie natürlich noch an den ersten (E)IDE Kanal gehängt werden.

In komplexeren Fällen ist unter Umständen noch die Optionen »ignore-table« hilfreich.

Zu bedenken ist noch, daß man nach dem Kompilieren eines neuen Kernels das Boot-Verzeichnis der zweiten Festplatte anpaßt und LILO auch mit dem entsprechenden Befehl

```
lilo -C /etc/lilo.conf.backup
```

für die zweite Festplatte ausführt.

LILO im MBR

Benutzen Sie Linux als einziges Betriebsystem, bietet es sich an, LILO direkt im MBR Ihrer Festplatte unterzubringen.

LILO im Superblock mit externem BootManager

Um auch Betriebsysteme neben Linux zu starten, mit denen LILO nicht zurecht kommt, bietet es sich an, einen externen BootManager wie etwa den OS/2-Bootmanager oder XFDisk zu benutzen. Hierbei wird LILO im Superblock Ihrer Root-Partition untergebracht und der externe BootManager im MBR der Festplatte.

LILO direkt vom RAID im MBR

Das grundsätzliche LILO Problem, die Geometriedaten des Kernels wissen zu müssen und somit nicht direkt von einem RAID-Device booten zu können, kann man umgehen, indem man LILO in der `/etc/lilo.conf` auf dem Root-RAID diese Parameter schon mit übergibt. Prinzipiell funktioniert dies für alle RAID-Modi. Wirklich Sinn macht das aber nur für RAID-Modi wie RAID-1, RAID-4 und RAID-5, welche auch irgendeine Form von Redundanz versprechen. Im Gegenzug ist das direkte Booten von einem RAID-0-Verbund schon deshalb einfacher zu realisieren, weil man sich beim Defekt einer Festplatte keine Gedanken mehr um die Datenrettung oder das Booten von der zweiten Festplatte zu machen braucht: Diese Daten sind dann ohnehin verloren.

Wie funktioniert nun das direkte Booten von einem RAID-Verbund? Hier ein Beispiel der Datei `/etc/lilo.conf` für den wohl sinnvollsten RAID-Modus für einen Root-RAID Verbund: RAID-1 auf SCSI-Festplatten:

```
disk=/dev/md15
              bios=0x80
              sectors=63
              heads=255
              cylinders=1106
              partition=/dev/md0
              start=1028160
boot=/dev/sda
image=/boot/vmlinux-2.2.10
              label=autoraid
              root=/dev/md0
              read-only
```

Der Eintrag `disk=/dev/md15` mit seinen Parametern übergibt LILO die benötigten Geometriedaten einer fiktiven Festplatte `/dev/md15`. Hierbei ist es einerlei, ob dieses Device `/dev/md15` oder `/dev/md27` heißt. Wichtig ist nur, daß es per `mknod` mit der Major-Number eines RAID-Devices erstellt wurde. Sind Sie sich nicht sicher, ob dies der Fall ist, lassen Sie sich einfach unter `/dev/` alle Devices, die mit `md` anfangen, zeigen. Standardmäßig werden `/dev/md0` bis `/dev/md15` erstellt. Die Parameter der Sektoren, Köpfe und Zylinder unterhalb von `disk=/dev/md15` geben die Geometriedaten der ersten Festplatte Ihres Systems an, welche man mittels

```
fdisk -lu /dev/sda
```

erhält. Die Angabe der Partition soll Ihr Root-RAID Verbund sein. Der letzte Parameter `start=1028160` bezeichnet den Sektor, in dem Ihre RAID-Partition auf der ersten Festplatte beginnt. Auch diese Information erhalten Sie durch:

```
fdisk -lu /dev/sda
```

Desweiteren muß als Sitz des LILO der MBR Ihrer ersten Festplatte angegeben werden. Hier geschehen durch den Eintrag: `boot=/dev/sda`. Der letzte Bereich beschreibt ganz normal die Lokalisation Ihres Kernels mit dem Verweis, als Root-Partition Ihren RAID-Verbund zu nutzen.

Haben Sie den RAID-Verbund nach der weiter unten beschriebenen Methode erstellt und haben sowohl die Option `persistent-superblock` aktiviert als auch den Partitionstyp der Festplatten auf `0xfd` geändert, fehlen dem Master-Boot-Record der SCSI-Festplatten nur noch die Boot-Informationen. Mit dem Aufruf

```
lilo -b /dev/sda
```

werden die Informationen der Datei `/etc/lilo.conf` in den MBR der ersten SCSI-Festplatte geschrieben. Anschließend muß man den Befehl ein zweites Mal aufrufen. Diesmal allerdings für den MBR der zweiten Festplatte des RAID-1 Verbundes:

```
lilo -b /dev/sdb
```

Achtung: Hierbei wird davon ausgegangen, daß die im RAID-Verbund laufenden Festplatten identisch sind und damit auch die gleichen Geometriedaten besitzen! Ein RAID-0 so zu booten, funktioniert auch mit unterschiedlichen Festplatten, da hierbei nur die erste Festplatte berücksichtigt wird. In diesem Beispiel eines RAID-1 liegen jedoch auf allen RAID-Partitionen die gleichen Daten und somit auch die gleiche `/etc/lilo.conf`. Haben die Festplatten unterschiedliche Geometriedaten und fällt im RAID-1 die erste Festplatte aus, so stehen im MBR der zweiten Festplatte Daten, welche nicht mit denen der zweiten Festplatte übereinstimmen. Ein Workaround könnte sein, zwei LILO Konfigurationsdateien zu pflegen und mit unterschiedlichen Geometriedaten in den MBR der jeweiligen Festplatten zu schreiben. Da mir aber nur mehrere Exemplare dergleichen Festplatte zum Testen von RAID-Verbunden vorliegen, ist dies ein ungesicherter Tip.

Der Erfolg ist ein RAID-1 Verbund, den man auch nach einem erneuten Kernelkompilierungslauf durch zweimaliges Aufrufen des LILO mit den Parametern für die unterschiedlichen MBRs von beiden beteiligten RAID-Festplatten booten kann.

LILO direkt vom RAID-1 im MBR oder Superblock

Will man die Root-Partition direkt von einem RAID-1 Verbund booten, bietet sich einem noch die weitaus eleganteste Möglichkeit: Die LILO-Version des aktuellen RPM-Archives `lilo-0.21-10.i386.rpm` kann bereits von sich aus mit RAID-1 Verbunden umgehen. Andere RAID-Modi werden allerdings nicht unterstützt.

4.2 Möglichkeiten zum Kopieren von Daten

Zu den ersten Methoden muß vorab gesagt werden, daß je nach Distribution bei der Verwendung der Root-Partition als RAID einige Verzeichnisse entweder gar nicht kopiert werden dürfen, oder aber nur als leere Verzeichnisse erstellt werden müssen. Im einzelnen sollte man auf folgende achten:

proc

Dieses Verzeichnis bitte nur als leeres Verzeichnis auf dem RAID-Device erstellen, da unter `/proc` ein Pseudo-Dateisystem erstellt wird, welches im Prinzip keinen Platz beansprucht - bitte nicht versuchen, `/proc` auf eine andere Partition zu kopieren.

mnt

Dieses Verzeichnis oder das, wo Ihr RAID-Device gemountet ist, darf nicht kopiert werden, sonst würden die bereits vorhandenen Daten nochmals überschrieben werden.

cdrom

Die Daten der CDs selbst möchte man natürlich nicht kopieren. Es sollten daher nur die passenden Mountpoints erstellt werden.

dos

Auch die DOS-Partition, falls eine vorhanden ist, möchte man nicht mit rüberkopieren. Es sollte also nur ein passender Mountpoint erstellt werden.

floppy

Das gleiche gilt auch für eine gemountete Diskette.

Als generelle Kopiermethoden bieten sich folgende Möglichkeiten an:

cp

Der normale Copy-Befehl eignet sich für das Kopieren naheliegenderweise sehr gut und funktioniert problemlos.

dd

Auch eine saubere Möglichkeit, Verzeichnisse zu kopieren, bietet das Programm `dd`.

dump und restore

Mittels `dump` und `restore` läßt sich ohne viel Aufwand z.B. das ganze Root-Verzeichnis kopieren oder auf Band-Streamer sichern, wobei die unnötigen Verzeichnisse wie `/proc` oder `/mnt` fast »automatisch« ausgelassen werden. Zu diesem Zweck wechselt man in das Verzeichnis, in dem der neue RAID-Verbund gemountet ist und führt die folgenden Befehle aus, um z.B. das Verzeichnis `/usr` zu kopieren:

```
dump 0Bf 1000000000 - /usr | restore rf -
rm restoresymtable
```

Midnight Commander

Zwar liegt der Midnight Commander zum Kopieren von Verzeichnissen auf ein neues RAID-Array sehr nahe, jedoch haben einige ältere Versionen die bisweilen sehr unangenehme Eigenart, symbolische Links beim Kopieren zu stabilisieren. In den aktuellen Versionen sollte dieses Fehlverhalten jedoch behoben sein.

tar

Ebenso zuverlässig und mit einigen Extraoptionen kann man `tar` benutzen, um ganze Verzeichnisstrukturen zu kopieren.

4.3 Möglichkeiten zum Verändern ganzer Partitionen

ext2resize

Eine native Möglichkeit unter Linux, ext2-Partitionen zu verändern, die noch dazu der GPL unterliegt, bietet das Programm `ext2resize`, das von folgender Adresse bezogen werden kann:

```
http://ext2resize.sourceforge.net/
```

Da es aber offiziell noch Beta-Status hat, ist beim Umgang mit diesem Programm Vorsicht geboten.

Partition Magic

Seit der Version 4.0 kann Partition Magic auch mit Linux ext2-Partitionen umgehen. Eine Version, die unter Linux selbst lauffähig ist, gibt es allerdings nicht. Das Produkt Partition Magic stammt von der Firma PowerQuest.

resize2fs

Dieses Programm ist eine Auskopplung aus Partition Magic, ist unter Linux lauffähig und ermöglicht das Vergrößern und Verkleinern von ext2-Partitionen. Sollten Sie mal über ein gleichnamiges tar-Archiv stolpern, stellt sich hier jedoch noch die Lizenzfrage. Registrierte Benutzer können sich das RPM-Paket von

```
http://www.powerquest.com/
```

herunterladen. Innerhalb der Firma überlegt man aber, diesen Teil eventuell frei zu geben.

5 RAID-Verbunde mit den RAID-Tools Version 0.4x erstellen

Dieses Kapitel bezieht sich auf die Erstellung von RAID-Verbunden unter Zuhilfenahme der alten MD-Tools Version 0.4x. Die aktuellen RAID-Tools Version 0.9x und der RAID-Patch werden in allen Einzelheiten im Abschnitt 6 (RAID-Verbunde mit den RAID-Tools Version 0.9x erstellen) beschrieben.

5.1 Vorbereiten des Kernels

Der Standardkernel Ihrer Distribution besitzt prinzipiell schon gleich nach der Installation alle Optionen, um RAID-Devices zu erstellen.

Desweiteren ist noch das Paket `md-0.35-2.i386.rpm` nötig, welches zum Beispiel für die DLD auf der ersten DLD CD unter `delix/RPMS/i386/` zu finden ist. Auch wenn dieses Paket bei Ihrer Distribution eine andere Versionsnummer haben sollte, können Sie es ruhig benutzen.

Der Vollständigkeit halber wird hier trotzdem auf die nötigen Kernel-Parameter hingewiesen:

Nach der Anmeldung als root, dem Wechsel in das Verzeichnis `/usr/src/linux` und dem Aufruf von `make menuconfig` sollte sich Ihnen das Menü mit den unterschiedlichen Kerneloptionen präsentieren. Bitte benutzen Sie nicht `make config` oder `make xconfig`, da sich diese Beschreibung ausschließlich auf `make menuconfig` stützt.

Kernel 2.0.36

Unter dem Verweis »Floppy, IDE, and other block devices —>« werden je nach RAID Wunsch folgende Optionen benötigt:

```
[*] Multiple devices driver support
<*> Linear (append) mode
<*> RAID-0 (striping) mode
<*> RAID-1 (mirroring) mode
<*> RAID-4/RAID-5 mode
```

Kernel 2.2.x bis einschließlich 2.2.10

Hier stehen die RAID Optionen unter »Block devices —>«:

```
[*] Multiple devices driver support
<*> Linear (append) mode (NEW)
<*> RAID-0 (striping) mode (NEW)
<*> RAID-1 (mirroring) mode (NEW)
<*> RAID-4/RAID-5 mode (NEW)
[*] Boot support (linear, striped) (NEW)
```

Zusätzlich wird bei den 2.2.xer Kerneln bei der Auswahl des Linear oder RAID-0 Modes der Boot Support angeboten. Von RAID-1, 4 und 5 Devices kann mit den hier gegebenen Möglichkeiten nicht gebootet werden. Das funktioniert erst mit dem neuen RAID-Patch; siehe Abschnitt 6 (RAID-Verbunde mit den RAID-Tools Version 0.9x erstellen).

Nach Auswahl der nötigen Parameter erfolgt das »Backen« des Kernels mittels:

```
make dep && make clean && make bzImage
```

Das Kompilieren und Installieren der Module nicht vergessen:

```
make modules && make modules_install
```

Den Kernel zusammen mit der `System.map` umkopieren und zu guter Letzt den Aufruf von LILO nicht vergessen. Die Benutzer eines SCSI-Kontrollers müssen noch die initiale RAM-Disk mittels

```
mkinitrd /boot/initrd Kernelversion
```

erstellen, falls der SCSI-Kontroller als Modul eingeladen wird.

Nach einem Neustart hat man nun alle Voraussetzungen erfüllt, um ein RAID-Device zu erstellen.

5.2 Erstellen eines RAID-0 Devices

Die RAID-Arrays oder RAID-Verbunde werden nachher über `/dev/md*` angesprochen. Bei der DLD 6.0 sind diese Devices bereits eingerichtet. Schauen Sie zur Sicherheit unter `/dev/` nach.

Jetzt wird das Erstellen eines RAID-Verbundes anhand eines RAID-0 Devices erklärt. Andere RAID-Modi lassen sich analog erstellen. Zuerst sollte man sich darüber im klaren sein, welche und wieviele Partitionen man zusammenfassen möchte. Diese Partitionen sollten leer sein; eine Einbindung von Partitionen, die Daten enthalten, welche nachher wieder zugänglich sein sollen, ist bisher meines Erachtens nicht möglich. Man sollte sich die Devices und ihre Reihenfolge nicht nur gut merken, sondern besser aufschreiben.

Als Beispiel werden die zwei SCSI-Festplatten `/dev/sda` und `/dev/sdb` benutzt. Bei (E)IDE würden sie dann `/dev/hda`, `/dev/hdb` heißen. Auf diesen Festplatten liegen nun zwei leere Partitionen im erweiterten Bereich `/dev/sda6` und `/dev/sdb6`, welche zu einem RAID-0 Device zusammengefaßt werden sollen.

```
mdadd /dev/md0 /dev/sda6 /dev/sdb6
```

Sind diese Partitionen nicht gleich groß, so ist der zu erwartende Geschwindigkeitsvorteil nur auf dem Bereich gegeben, der von beiden Partitionen abgedeckt wird. Zum Beispiel sind eine 200 MB große und eine 300 MB große Partition als RAID-0 Device nur über die ersten 400 MB doppelt so schnell. Die letzten 100 MB der zweiten Festplatte werden ja nun nur noch mit einfacher Geschwindigkeit beschrieben.

Wieder anders sieht das bei der Einrichtung eines RAID-1 Verbundes aus. Hierbei wird der überschüßige Platz von der größeren Partition nicht genutzt und liegt damit brach auf der Festplatte.

Allgemein heißt das also, daß man für RAID-Devices jeder Art möglichst gleich große Partitionen benutzen sollte. Die Außnahme bildet der Linear Modus, bei dem es wirklich egal ist, wie groß die einzelnen Partitionen sind.

Nun muß Linux noch erfahren, als was für ein RAID-Device es dieses `/dev/md0` ansprechen soll:

```
mdrun -p0 /dev/md0
```

Hierbei steht `-p0` für RAID-0. Anschließend muß auf diesem neuen RAID-Device ein Dateisystem erstellt werden:

```
mke2fs /dev/md0
```

Testweise kann man das RAID-Device nun nach `/mnt` mounten und ein paar kleine Kopieraktionen drüberlaufen lassen:

```
mount -t ext2 /dev/md0 /mnt
```

Hat man an den unterschiedlichen Festplatten jeweils einzelne LEDs, sieht man jetzt schon sehr eindrucksvoll, wie das RAID arbeitet. Alle Daten, die ab jetzt auf `/dev/md0` geschrieben werden, nutzen den RAID-0 Modus.

Bevor der Rechner runtergefahren wird, müssen die RAID-Devices jedoch noch gestoppt werden:

```
umount /mnt
mdstop /dev/md0
```

5.3 Automatisierung

Um nicht nach jedem Bootvorgang diese Prozedur wiederholen zu müssen, benötigt man eine Datei `/etc/mdtab`, welche - analog zu `/etc/fstab` - die Mountparameter enthält. Dies erledigt der Befehl:

```
mdcreate raid0 /dev/md0 /dev/sda6 /dev/sdb6
```

Dadurch wird die Datei `/etc/mdtab` erstellt, welche zusätzlich noch eine Prüfsumme enthält und das Aktivieren des RAID-Devices durch den einfachen Befehl

```
mdadd -ar
```

erlaubt. Nun trägt man das RAID-Device (`/dev/md0`) noch unter `/etc/fstab` mit der Zeile

```
/dev/md0 /mnt ext2 defaults 0 1
```

ein, wobei natürlich `/mnt` durch jeden beliebigen Mountpoint ersetzt werden kann. Ein

```
mount /mnt
```

führt nun auch zum Mounten des RAID-0 Devices.

Leider berücksichtigen die Init-Skripte der DLD 6.0 keine RAID-Devices, wodurch man noch einmal auf Handarbeit angewiesen ist. Inwieweit andere Distributionen bereits RAID-Verbunde berücksichtigen, sollten Sie durch einen Blick in die Startskripte feststellen und gegebenenfalls ähnlich den hier abgedruckten Beispielen anpassen.

Zum Starten des RAID-Devices ist es unerläßlich den, Befehl `mdadd -ar` unterzubringen. Will man nicht von dem RAID-Device booten, so reicht es, den Befehl in die Datei `/etc/init.d/bc.fsck_other` eine Zeile über den `fsck` Befehl einzutragen.

```
##################################################
# Filesystem check und u.U sulogin bei Problemen
##################################################
if [ ! -f /fastboot ]
then
   mini_text="Überprüfe Filesysteme..."
   mini_startup "$mini_text" start
```

```
   log_msg_buf=`(


mdadd -ar


      fsck -R -A -a
      ) 2>&1`
   fsck_result=$?
   old_fsck_result=$fsck_result

[... Teile gelöscht ...]

#################################################
```

Analog sollte der Befehl `mdstop -a` in die zuletzt ausgeführte Datei nach dem `umount` Befehl eingetragen werden. Bei der DLD heißt die Datei `/etc/init.d/halt` und sollte nach dem Eintrag an der richtigen Stelle so aussehen:

```
#################################################
# Umount all FS
#################################################
mini_text="Die Filesysteme werden gelöst"
mini_shutdown "$mini_text" start
LC_LANG=C
LC_ALL=C
export LC_LANG LC_ALL
# Gib den sleeps der busy_wait_loops zeit sich zu beenden.
# sonst gibt es :/usr device busy.
sleep 1
umount -a

mdstop -a

mini_shutdown "$mini_text" stop 0
#################################################
```

Damit kann man nun recht komfortabel das RAID-Device benutzen.

Weitere RAID-Verbunde erstellt man auf dieselbe Weise, jedoch sind die Einträge in den Init-Skripten nur einmal zu setzen.

6 RAID-Verbunde mit den RAID-Tools Version 0.9x erstellen

6.1 Vorbereiten des Kernels

Sie benötigen hierfür den aktuellen, »sauberen«, ungepatchten Sourcetree des 2.2.10er Kernels. Bitte tun Sie sich selbst einen Gefallen und nehmen Sie keinen Sourcetree Ihrer Distribution. Diese sind meist schon mit Patches aller Art versehen und ein zusätzliches ändern mit dem RAID-Patch würde die Sourcen eventuell sogar zerstören. Die kompletten Kernel-Sourcen erhält man auf:

```
ftp.kernel.org:/pub/linux/kernel/v.2.2/
```

Desweiteren benötigen Sie den für diesen Kernel passenden RAID-Patch (`raid0145-19990724-2.2.10`) und die aktuellsten RAID-Tools (`raidtools-19990724-0.90.tar.gz`). Diese sind hier zu finden:

```
ftp.kernel.org:/pub/linux/daemons/raid/alpha/
```

Bitte achten Sie genau auf die passende Kernelversion!

Zuerst muß der Kernel nach `/usr/src/linux` kopiert und mittels

```
tar xvfz kernelfile.tar.gz
```

entpackt werden. Alternativ und bei Benutzung mehrerer Kernel entpackt man ihn nach `/usr/src/linux-2.2.10` und setzt den vermutlich schon vorhandenen symbolischen Link von `/usr/src/linux` auf `/usr/src/linux-2.2.10`. Dies ist für das Patchen wichtig, da sonst der falsche Sourcetree gepatcht werden könnte. Auch ist es immer schlau, sich einen ungepatchten Original-Sourcetree aufzuheben, falls mal etwas schiefgeht.

Nun wird der RAID-Patch nach `/usr/src/linux-2.2.10` kopiert und dort mittels

```
patch -p1 < raidpatchfilename
```

in den Sourcetree eingearbeitet. Es sollten nun etwa 20 Dateien kopiert und teilweise geändert werden.

Nach dem Aufruf von

```
make menuconfig
```

sollte sich Ihnen das Menü mit den unterschiedlichen Kerneloptionen präsentieren. Bitte benutzen Sie nicht `make config` oder `make xconfig`, da sich diese Beschreibung ausschließlich auf `make menuconfig` stützt. Hier aktivieren Sie unter »Block devices —>« folgende Optionen:

```
[*] Multiple devices driver support
[*] Autodetect RAID partitions
<*> Linear (append) mode
<*> RAID-0 (striping) mode
<*> RAID-1 (mirroring) mode
<*> RAID-4/RAID-5 mode
< > Translucent mode
< > Logical Volume Manager support (NEW)
[*] Boot support (linear, striped)
```

Da immer wieder einige Fehler durch modularisierte RAID-Optionen auftauchen, nehmen Sie sich an den obigen Einstellungen ein Beispiel und kompilieren Sie den benötigten RAID-Support fest in den Kernel ein.

Passen Sie nun noch alle übrigen Kerneleinstellungen Ihren Wünschen an, verlassen Sie das Menü und kompilieren Sie Ihren neuen Kernel:

```
make dep && make clean && make bzImage && make modules
make modules_install
```

Die Benutzer eines SCSI-Kontrollers sollten daran denken, die initiale RAM-Disk mittels

```
mkinitrd /boot/initrd Kernelversion
```

neu zu erstellen, falls der SCSI-Kontroller als Modul eingeladen wird. Anschließend noch den neuen Kernel und die `System.map` Datei umkopieren, die `/etc/lilo.conf` bearbeiten, `lilo` ausführen und das System neu starten.

Ihr Kernel unterstützt nun alle nötigen RAID Optionen.

Weiter geht es mit den RAID-Tools. Entpacken Sie die RAID-Tools z.B. nach `/usr/local/src`, führen Sie das enthaltene Konfigurationsskript aus und kompilieren Sie die Tools:

```
./configure && make && make install
```

Die RAID-Tools stellen Ihnen zum einen (unter anderem) die Datei `mkraid` zur Verfügung und erstellen zum anderen die `/dev/md0-15` Devices.

Ob Ihr Kernel die RAID Optionen wirklich unterstützt können Sie mittels

```
cat /proc/mdstat
```

in Erfahrung bringen. Diese Pseudodatei wird auch in Zukunft immer Informationen über Ihr RAID System enthalten. Ein Blick hierher lohnt manchmal. Zu diesem Zeitpunkt sollte zumindest ein Eintrag enthalten sein, der Ihnen zeigt, daß die RAID Personalities registriert sind.

Die RAID-Devices, was dasselbe bezeichnet wie das RAID-Array oder den RAID-Verbund, werden nachher über `/dev/md*` angesprochen.

Zuerst sollte man sich darüber im klaren sein, welche und wieviele Partitionen man zusammenfassen möchte. Diese Partitionen sollten leer sein und man sollte sich die Devices und ihre Reihenfolge nicht nur gut merken, sondern besser aufschreiben. Eine Einbindung von Partitionen, die Daten enthalten, welche nachher wieder zugänglich sein sollen, ist bisher nur über einen Umweg möglich und das auch nur bei Verwendung eines RAID-0 Verbundes.

Als Beispiel werden die zwei SCSI-Festplatten `/dev/sda` und `/dev/sdb` benutzt. Bei (E)IDE heißen sie dann `/dev/hda`, `/dev/hdb` usw. Auf diesen Festplatten liegen nun zwei leere Partitionen im erweiterten Bereich `/dev/sda6` und `/dev/sdb6`, welche zu einem RAID-0 Device zusammenfaßt werden sollen.

6.2 Erstellen eines RAID-0 Devices

Die Grundkonfiguration der zu erstellenden RAID-Devices werden hier unter `/etc/raidtab` gespeichert. Diese Datei enthält alle nötigen Angaben für ein oder mehrere RAID-Devices. Generelle Hilfe findet man in den Manual Pages von `raidtab` und `mkraid`. Eine `raidtab` für ein RAID-0 System mit den oben beschriebenen zwei Partitionen müßte so aussehen:

```
raiddev /dev/md0
        raid-level              0
        nr-raid-disks           2
        persistent-superblock   1
        chunk-size              4

        device                  /dev/sda6
        raid-disk               0
        device                  /dev/sdb6
        raid-disk               1
```

Sind die Eintragungen in Ordnung, kann das RAID mittels

```
mkraid /dev/md0
```

gestartet werden. Formatieren Sie Ihr neues Device:

```
mke2fs /dev/md0
```

Dieses Device kann nun - wie jedes andere Blockdevice auch - irgendwo in den Verzeichnisbaum gemountet werden.

Um das RAID-Device wieder zu stoppen, unmounten Sie es und führen Sie

```
raidstop /dev/md0
```

aus. Nach einem Reboot kann das Device mittels

```
raidstart /dev/md0
```

wieder aktivert und anschließend überall hin gemountet werden.

6.3 Automatisierung

Wieviel Komfort einem der neue RAID-Patch bringt, zeigt sich bei diesem Automatisierungsprozeß. Das bei der Verwendung der MD-Tools Version 0.4x beschriebene Bearbeiten der Init-Skripte fällt völlig weg. Allerdings ist dafür der einmalige Umgang mit dem nicht ganz ungefährlichen Tool `fdisk` vonnöten. Bitte seien Sie sich absolut sicher mit der Einteilung Ihrer Festplattenpartitionen, bevor Sie sich damit beschäftigen.

Dieses Beispiel sieht nun vor, den RAID-0 Verbund `/dev/md0` bestehend aus `/dev/sda6` und `/dev/sdb6` komfortabel bei jedem Startup zu mounten und automatisch auch wieder herunterzufahren.

Der Trick liegt hierfür in der von Ihnen im Kernelmenü gewählten Option »Autodetect RAID partitions«. Der Kernel findet damit bestehende RAID-Arrays beim Booten automatisch und aktiviert sie.

Voraussetzungen dafür sind:

1. »Autodetect RAID partitions« ist im Kernel aktiviert.
2. »persistent-superblock 1« ist in der Datei `/etc/raidtab` für Ihr Device gesetzt.
3. Der Partitionstyp, der von Ihnen für das RAID genutzten Partitionen, muß auf »0xFD« gesetzt werden.

Sind Sie dieser Anleitung bis hierher gefolgt, dann enthält Ihr Kernel bereits die Autodetection und Sie haben Ihr RAID-Device auch mit dem persistent-superblock Modus erstellt. Bleibt noch das Ändern des Partitionstyps.

Stellen Sie sicher, daß das RAID-Device gestoppt ist, bevor Sie anfangen, mit `fdisk` zu arbeiten:

```
umount /dev/md0
raidstop /dev/md0
```

Bei der vorliegenden Konfiguration müßten die Typen der Partitionen `/dev/sda6` und `/dev/sdb6` geändert werden. Dafür wird `fdisk` zuerst für die erste SCSI-Festplatte aufgerufen:

```
fdisk /dev/sda
```

Man wählt nun die Option »t« für »toggle partition type« und wird dann aufgefordert, den Hexcode oder einen Partitionstypen einzugeben. »l« liefert eine Liste der unterstützten Codes, welche aber im Moment uninteressant ist. Geben Sie nun einfach »fd« ein und prüfen Sie mit »p«, ob die Partition tatsächlich geändert wurde. In der »Id«-Spalte sollte nun »fd« auftauchen. Beenden Sie anschließend `fdisk` mit der Option »w«. Verfahren Sie analog mit den anderen zu Ihrem RAID gehörenden Partitionen. Um z.B. `/dev/sdb6` zu ändern, rufen Sie `fdisk` so auf:

```
fdisk /dev/sdb
```

Haben Sie das erfolgreich hinter sich gebracht, wird Ihnen beim nächsten Startup der Kernel mitteilen, daß er Ihr RAID-Device gefunden hat und es aktivieren. Schauen Sie ruhig nochmal mittels `cat /proc/mdstat` Ihr Device an. Es sollte nach diesem Neustart bereits aktiviert sein.

Anschließend können Sie Ihr `/dev/md0` RAID-Device wie jede andere Partition in die `/etc/fstab` eintragen und so automatisch an einen beliebigen Ort mounten. Das Dateisystem auf dem RAID ist ext2, obwohl Sie den RAID-Verbund natürlich mit jedem beliebigen Dateisystem formatieren können.

6.4 Anmerkung

Manche Distributionen sehen es innerhalb ihrer Init-Skripte bereits vor, eventuell vorhandene RAID-Devices beim Herunterfahren des Systems zu deaktivieren oder aber beim Booten zu aktivieren. Dieser Umstand kann zu Fehlermeldungen führen, die Sie jedoch nicht weiter beeindrucken sollten. Diese Distributionen gehen meist davon aus, daß Sie die alten MD-Tools Version 0.4x benutzen. Ihr gepatchter Kernel übernimmt aber ab sofort das Aktivieren und Deaktivieren Ihrer mit den neuen RAID-Tools erstellten RAID-Devices für Sie. Um die Fehlermeldungen zu unterdrücken, kommentieren Sie einfach in den passenden Init-Skripten (z.B. `/etc/rc.d/init.d/halt`) die Abfrage nach RAID-Devices und deren nachfolgender Deaktivierung mittels z.B. `mdstop -a` - oder was auch immer dort angegeben ist - aus.

7 Root-Partition oder Swap-Partition als RAID

7.1 Root-Partition als RAID

Eine existierende HOWTO beschäftigt sich bereits mit dem Thema, allerdings ist Sie erstens auf Englisch und zweitens geht es durch neue Kerneloptionen auch eleganter als dort beschrieben.

Bitte erleichtern Sie sich Ihre Arbeit und senken Sie das Frustnieveau, indem Sie vorab noch drei Tips beherzigen:

1. Erstellen Sie ein Backup Ihrer Daten.
2. Erstellen Sie sowohl eine funktionierende und getestete DOS als auch Linux Bootdiskette.
3. Lesen Sie diese Ausführungen erstmal komplett durch, bevor Sie mittendrin feststellen müssen, etwas wichtiges vergessen zu haben, das Sie zu so spätem Zeitpunkt nicht mehr nachholen können.

7.1.1 DLD 6.0

Für dieses Verfahren wird noch eine DOS oder Win95 Partition in Verbindung mit dem Loadlin benötigt. Loadlin befindet sich - so das Programm nicht schon eingesetzt wird - auf der ersten DLD CD unter `/delix/RPMS/i386/loadlin-1.6.2.i386.rpm`. Alternativ kann man sich auch eine Bootdiskette anfertigen.

Der Hintergrund ist ganz einfach der, daß LILO mit dem RAID-Device nicht zurechtkommt und man somit nicht explizit von diesem RAID-Device booten kann. Daher behilft man sich hier entweder mit Loadlin, oder aber mit einem Mini Linux auf einer kleinen Extra-Partition. Weitere Möglichkeiten zum Booten von Linux auch von RAID-Verbunden wurden bereits im Abschnitt 4.1 (Möglichkeiten des Bootens von Linux) behandelt.

Nichts desto trotz bleiben wir bei dem Verfahren mit Loadlin. Das Programm befindet sich nach erfolgreicher Installation des RPMS-Paketes unter `/dos/loadlin/`. Auf der oben genannten nötigen DOS Partition richtet man sich nun ein Verzeichnis wie z.B. Linux ein und kopiert die Dateien `loadlin.bat` und `loadlin.exe` zusammen mit dem frischen Kernel, in den die RAID-Parameter einkompiliert wurden, hinein.

Um sicherzugehen, daß auch wirklich nichts passiert, sollte man entweder die nötigen Treiber für den (E)IDE Kontroller oder des passenden SCSI Kontroller auch mit in den Kernel einkompiliert haben.

Die Batchdatei `loadlin.bat` wird nun dahingehend angepaßt, daß wir die Parameter für das zu bootende RAID-Device gleich mit angeben:

```
md=<md device no.>,<raid level>,<chunk size>,dev0,...,devn
```

md device no

Die Nummer des RAID (md) Devices: »1« steht für `/dev/md1`, »2« für `/dev/md2` usw.

raid level

Welches RAID Level wird verwendet: »-1« für Linear Modus, »0« für RAID-0 (Striping).

chunk size

Legt die Chunk Size fest bei RAID-0 und RAID-1 fest.

dev0-devn

Eine durch Kommata getrennte Liste von Devices, aus denen das md Device gebildet wird, z.B. `/dev/hda1,/dev/hdc1,/dev/sda1`.

Andere RAID-Modi außer Linear und RAID-0 werden im Moment nicht unterstützt. Gemäß der vorher beschriebenen Anleitung würde die Zeile in der `linux.bat` dann so aussehen:

```
c:\linux\loadlin c:\linux\zimage root=/dev/md0 md=0,0,0,0,/dev/sda6,/dev/sdb6 ro
```

Dies soll nur eine einzige Zeile sein; außerdem ist auch hier wieder auf die richtige Reihenfolge der Partitionen zu achten. Weiterhin müßen natürlich zwei oder mehrere Partitionen - zu entweder einem RAID-0 oder einem Linear Device zusammengefaßt - bereits vorliegen. Der Kernel muß die o.a. Bootoption und die nötigen RAID oder Linear Parameter in den Kernel einkompiliert haben; man beachte: Nicht als Module. Dann mountet man das RAID-Device, welches später die Root-Partition werden soll, nach z.B. `/mnt`, kopiert mittels einer der im Abschnitt 4.2 (Möglichkeiten zum Kopieren von Daten) beschriebenen Methoden die benötigten Verzeichnisse auf das RAID-Device.

Speziell für die DLD 6.0, aber auch für alle anderen Distributionen sei hier gesagt, daß beim Booten von einem RAID-Device der oben beschriebene Befehl `mdadd -ar` vor dem ersten `mount` Befehl auszuführen ist. Für die DLD 6.0 heißt das konkret, daß der Befehl bereits in das Skript `/etc/init.d/bc.mount_root` eingetragen werden muß, da dort der erste `mount` Befehl ausgeführt wird. Benutzer anderer Distributionen sind hier auf sich gestellt oder schauen sich zur Not die Methode mit den neueun RAID-Tools Version 0.9x an; siehe Abschnitt 6 (RAID-Verbunde mit den RAID-Tools Version 0.9x erstellen).

Jetzt fehlen nur noch die passenden Einträge in `/etc/fstab` und `/etc/lilo.conf`, in denen man auf dem neuen RAID-Device die ursprüngliche Root-Partition in `/dev/md0` umändert - im Moment liegen diese Dateien natürlich noch unter `/mnt`.

An dieser Stelle sollte man die obige Liste noch einmal in Ruhe durchsehen, sich vergewissern, daß alles stimmt, und dann die DOS oder Win95 Partition booten. Dort führt man nun die Batchdatei `linux.bat` aus.

7.1.2 Generisch

Im Abschnitt 6 (RAID-Verbunde mit den RAID-Tools Version 0.9x erstellen) wurde bereits auf die vorzügliche Eigenschaft des automatischen Erkennens von RAID-Devices durch den Kernel-Patch beim Startup des Linux-Systems hingewiesen. Dieser Umstand legt die Vermutung nahe, daß es mit dieser Hilfe noch einfacher ist, die Root-Partition als RAID-Device laufen zu lassen. Das ist auch wirklich so, allerdings gibt es auch hierbei immer noch einige Kleinigkeiten zu beachten. Generell kann man Linux entweder mittels Loadlin oder mit Hilfe von LILO booten. Je nach Bootart ist die Vorgehensweise unterschiedlich aufwendig.

Für beide Fälle braucht man jedoch erstmal ein RAID-Device. Um bei dem Beispiel des RAID-0 Devices mit den Partitionen `/dev/sda6` und `/dev/sdb6` zu bleiben, nehmen wir dieses Device und mounten es in unseren Verzeichnisbaum.

Hier muß allerdings noch einmal darauf hingewiesen werden, daß ein RAID-0 Device als Root-Partition ein denkbar schlechtes Beispiel ist. RAID-0 besitzt keinerlei Redundanz; fällt eine Festplatte aus, ist das Ganze RAID im Eimer. Für eine Root-Partition sollte man deshalb auf jeden Fall ein RAID-1 oder RAID-5 Device vorziehen. Auch das funktioniert Dank der neuen Autodetect Funktion und wird analog dem beschriebenen RAID-0 Verbund eingerichtet.

Auf das gemountete RAID-Device `/dev/md0` kopiert man nun ganz simpel mittels einer der im Abschnitt 4.2 (Möglichkeiten zum Kopieren von Daten) beschriebenen Methoden das komplette Root-Verzeichnis.

Danach muß auf dem RAID-Device noch die Datei `/etc/fstab` so angepaßt werden, das als Root-Partition `/dev/md0` benutzt wird und nicht mehr die originale Root-Partition.

Erstellen Sie sich eine DOS-Bootdiskette - das pure DOS von Win95 tut es auch - und auf dieser ein Verzeichnis Linux. Hierher kopieren Sie nun aus dem passenden RPM-Paket Ihrer Distribution das DOS-Tool `loadlin` und Ihren aktuellen Kernel. Manchmal befindet sich `loadlin` auch unkomprimiert im Hauptverzeichnis der Distributions-CD. Der Kernel sollte natürlich die RAID-Unterstützung bereits implementiert haben. Nun erstellen Sie mit Ihrem Lieblingseditor in dem neuen Linux Verzeichnis eine `loadlin.bat`. Haben Sie Ihren Kernel z.B. `vmlinuz` genannt, sollte in der Datei `loadlin.bat` etwas in dieser Art stehen:

```
a:\linux\loadlin a:\linux\vmlinuz root=/dev/md0 ro vga=normal
```

Die Pfade müssen natürlich angepaßt werden. Ein Reboot und das Starten von der Diskette mit der zusätzlichen Ausführung der `linux.bat` sollte Ihnen ein vom RAID-Device gebootetes Linux bescheren. Booten Sie generell nur über Loadlin, so endet für Sie hier die Beschreibung.

Möchten Sie allerdings Ihr neues Root-RAID mittels LILO booten, finden Sie im Abschnitt 4.1 (Möglichkeiten des Bootens von Linux) diverse Methoden aufgelistet und teilweise sehr genau beschrieben, mit denen Sie sich noch bis zum endgültigen Erfolg beschäftigen müßten.

7.1.3 Anmerkung zum redundanten Root-RAID

Hat man sich bei einer anderen Distribution als SuSE 6.2 für einen Root-RAID Verbund entschieden, der im Fehlerfall auch von der zweiten Festplatte booten soll, muß man noch folgendes beachten: Da auch auf der zweiten Festplatte eine Boot-Partition benötigt wird, die zwar ebenso in der `/etc/fstab` aufgenommen wurde, aber im Fehlerfall nicht mehr vorhanden ist, fällt die SuSE 6.2 Distribution in einen Notfall-Modus, in dem das root-Paßwort eingegeben werden muß und das fragliche Dateisystem repariert werden soll. Dies kann Ihnen auch bei anderen Distributionen passieren.

Es wird also ein Weg benötigt, die beiden Boot-Partitionen `/boot` und `/boot2` nur dann zu mounten, wenn sie tatsächlich körperlich im Rechner vorhanden sind. Hierbei hilft ihnen das Skript `mntboot`:

```
#!/bin/sh

MNTBOOTTAB=/etc/mntboottab

case "$1" in
  start)
    [ -f $MNTBOOTTAB ] || {
      echo "$0: *** $MNTBOOTTAB: not found" >&2
      break
    }

    PARTS=`cat $MNTBOOTTAB`

    for part in $PARTS ; do
      [ "`awk '{print $2}' /etc/fstab | grep "^$part$"`" == "" ] && {
        echo "$0: *** Partition $part: not in /etc/fstab" >&2
        continue
      }

      [ "`awk '{print $2}' </proc/mounts | grep "^$part$"`" != "" ] && {
        echo "$0: *** Partition $part: already mounted" >&2
        continue
      }

      fsck -a $part
      [ $? -le 1 ] || {
        echo "$0: *** Partition $part: Defect? Unavailable?" >&2
        continue
      }

      mount $part || {
        echo "$0: *** Partition $part: cannot mount" >&2
        continue
      }
    done

    exit 0
    ;;
```

```
  *)
    echo "usage: $0 start" >&2
    exit 1
    ;;
esac
```

Das Skript gehört bei SuSE Distributionen nach `/sbin/init.d`, bei vermutlich allen anderen Linux Distributionen nach `/etc/rc.d/init.d`. Auf das Skript sollte ein symbolischer Link `/sbin/init.d/rc2.d/S02mntboot` zeigen. Für alle anderen gilt es hier einen Link nach `/etc/rc.d/rc3.d/S02mntboot` zu setzen, da außer den SuSE Distributionen wohl alle im Runlevel 3 starten und Ihre Links dafür in diesem Verzeichnis haben. Das Skript prüft ein paar Nebenbedingungen für diejenigen Partitionen, die in `/etc/mntboottab` eingetragen sind (darin sollten `/boot` und `/boot2` stehen) und ruft jeweils `fsck` und `mount` für diese Partitionen auf. Da es bei allen anderen Distributionen keine `/etc/mntboottab` gibt, gilt es hier diese zu erstellen oder anzupassen.

In der `/etc/fstab` sollten diese Partitionen mit »noauto« statt »defaults« eingetragen werden. Außerdem muß im sechsten Feld der Wert »0« stehen, da die Distribution im Backup-Fall sonst in den Notfall-Modus fällt.

7.2 RAID auch für Swap-Partitionen?

7.2.1 RAID-Technik mit normalen Swap-Partitionen

Sie überlegen sich, RAID auch für Swap Partitionen einzurichten? Diese Mühe können Sie sich sparen, denn der Linux-Kernel unterstützt ein RAID-Verhalten auf Swap-Partitionen ähnlich dem RAID-0 Modus quasi »von Haus aus«. Legen Sie einfach auf verschiedenen Festplatten ein paar Partitionen an, änderen Sie den Partitionstyp mittels

```
fdisk /dev/Ihre-Partition
```

und der Option »t« auf 82 und erstellen Sie das Swap Dateisystem:

```
mkswap /dev/Ihre-neue-Swap-Partition
```

Nun fügen Sie diese in die `/etc/fstab` ein und geben allen Swap-Partitionen dieselbe Priorität.

```
/dev/hda3 swap swap defaults,pri=1    0 0
/dev/hdb3 swap swap defaults,pri=1    0 0
/dev/sda4 swap swap defaults,pri=1    0 0
```

Vom nächsten Startup an werden die Swap Partitionen wie ein RAID-0 Device behandelt, da die Lese- und Schreibzugriffe ab jetzt gleichmäßig über die Swap-Partitionen verteilt werden.

Will man aus irgendwelchen Gründen zwei Swap-Partitionen höher priorisieren als eine Dritte, so kann man das auch über den Parameter »pri=« ändern, wobei die Priorität einen Wert zwischen »0« und »32767« annehmen kann. Ein höherer Wert entspricht einer höheren Priorität. Je höher die Priorität desto eher wird die Swap-Partition beschrieben. Bei der folgenden Konfiguration würde also `/dev/hda3` wesentlich stärker als Swap-Partition genutzt werden als `/dev/hdb3`.

```
/dev/hda3 swap swap defaults,pri=5    0 0
/dev/hdb3 swap swap defaults,pri=1    0 0
```

7.2.2 Swap-Partitionen auf RAID-1 Verbunden

Erstellt man die Swap-Partition auf einem vorhandenen RAID-1 Verbund, formatiert sie dann mittels `mkswap /dev/mdx` und trägt sie als Swap-Partition in die `/etc/fstab` ein, so hat man zwar keinen, oder nur einen kleinen lesenden Geschwindigkeitsvorteil, jedoch den großen, nicht zu unterschätzenden Vorteil, daß man bei einem Festplattendefekt nach dem Ausschalten des Rechners und dem Austausch der defekten Festplatte, ohne weitere manuelle Eingriffe wieder ein vollständig funktionierendes System hat. Der einzige Wermutstropfen betrifft hierbei die Freunde des Hot Plugging. Erfahrungsgemäß verkraftet Linux das Hot Plugging eines dermaßen gestalteten RAID-1 Verbundes nur, wenn vorher die Swap-Partitionen mittels `swapoff -a` abgeschaltet wurden.

Als Warnung seien hier aber noch zwei der schlimmsten Fälle genannt, über die man sich Gedanken machen sollte und die noch dazu voneinander abhängig sind:

Der erste Fall beschreibt die Situation, Swap auf einem RAID-1 Verbund mit den alten RAID-Tools und damit den sowohl in den 2.0.xer als auch in den 2.2.xer orginal im Kernel vorhandenen RAID Treibern zu benutzen. Hierbei können nach einer gewissen Laufzeit des Linux-Systems unweigerliche Abstürze auftreten. Der Grund dafür liegt in der Problematik der unterschiedlichen Cachestrategie von Software-RAID einerseits und Swap-Partitionen andererseits. Erst mit den aktuellen RAID-Treibern ist der Betrieb einer Swap-Partition auf einem RAID-Verbund stabil geworden. Wollen Sie also einen sicheren RAID-Verbund erstellen, der nachher aus Gründen der Ausfallsicherheit auch die Swap-Partition beinhalten soll, benutzen Sie bitte immer die aktuellen RAID-Treiber in Form des aktuellen RAID-Patches. Dies entspricht dann der Einrichtfunktionalität der RAID-Tools Version 0.9x.

Der zweite Fall beschreibt die generelle Problematik, mit der Sie sich bei der Einrichtung von Swap-Partitionen in Bezug auf Software-RAID auseinandersetzen müssen:

Hat man die Swap-Partition auf einen RAID-1 Verbund gelegt und zusätzlich dafür eine Spare-Disk reserviert, würde diese Spare-Disk natürlich bei einem Festplattendefekt sofort eingearbeitet werden. Das ist zwar erwünscht und auch so gedacht, jedoch funktioniert das Resynchronisieren dieses RAID-1 Verbundes mit einer aktiven Swap-Partition nicht. Die Software-RAID Treiber nutzen beim Resynchronisieren den Puffer-Cache, die Swap-Partition aber nicht. Das Ergebnis ist eine defekte Swap-Partition.

Als Lösung bleibt nur die Möglichkeit, keine Spare-Disks zu benutzen und nach einem Festplattenausfall `swapoff -a` per Hand auszuführen, die defekte Festplatte auszutauschen und nach dem Erstellen der Partitionen und des Swap-Dateisystems mit `swapon -a` wieder zu aktivieren.

Ein Problem bleibt dennoch: Gesetzt den Fall der Linux-Rechner würde aufgrund eines Stromausfalls nicht sauber heruntergefahren worden sein, so werden die RAID-Verbunde beim nächsten Startup automatisch resynchronisiert. Dies erfolgt mit einem automatischen »ge-nice-ten« Aufruf des entsprechenden RAID-Daemons im Hintergrund bereits zu Anfang der Bootprozedur. Im weiteren Bootverlauf werden aber irgendwann die Swap-Partitionen aktiviert und treffen auf ein nicht synchronisiertes RAID. Das Aktivieren der Swap-Partitionen muß also verzögert werden, bis die Resynchronisation abgeschlossen ist.

Wie unter Linux üblich läßt sich auch dieses Problem mit einem Skript lösen. Der Gedanke dabei ist, den Befehl `swapon -a` durch ein Skript zu ersetzen, welches die Pseudodatei `/proc/mdstat` nach der Zeichenfolge `resync=` durchsucht und im Falle des Verschwindens dieser Zeichenfolge die Swap-Partitionen aktiviert. Im folgenden finden Sie ein Beispiel dazu abgedruckt:

```
#!/bin/sh
#

RAIDDEVS=`grep swap /etc/fstab | grep /dev/md|cut -f1|cut -d/ -f3`

for raiddev in $RAIDDEVS
do
```

```
#  echo "testing $raiddev"
    while grep $raiddev /proc/mdstat | grep -q "resync="
    do
#     echo "`date`: $raiddev resyncing" >> /var/log/raidswap-status
      sleep 20
     done
     /sbin/swapon /dev/$raiddev
done

exit 0
```

8 Spezielle Optionen der RAID-Devices Version 0.9x

8.1 Was bedeutet Chunk-Size?

Mit dem Chunk-Size Parameter legt man die Größe der Blöcke fest, in die eine Datei zerlegt wird, die auf einen RAID-Verbund geschrieben werden soll. Diese ist nicht mit der Blockgröße zu verwechseln, die beim Formatieren des RAID-Verbundes als Parameter eines bestimmten Dateisystems angegeben werden kann. Vielmehr können die beiden verschiedenen Blockgrößen variabel verwendet werden und bringen je nach Nutzung unterschiedliche Geschwindigkeitsvor- wie auch -nachteile.

Nehmen wir das Standardbeispiel: RAID-0 (`/dev/md0`) bestehend aus `/dev/sda6` und `/dev/sdb6` soll als angegebene Chunk-Size in der `/etc/raidtab` 4 kB haben. Das würde heißen, daß bei einem Schreibprozeß einer 16 kB großen Datei der erste 4 kB Block und der dritte 4 kB Block auf der ersten Partition (`/dev/sda6`) landen würden, der zweite und vierte Block entsprechend auf der zweiten Partition (`/dev/sdb6`). Bei einem RAID, das vornehmlich große Dateien schreiben soll, kann man so bei einer größeren Chunk-Size einen kleineren Overhead und damit einen gewissen Performancegewinn feststellen. Eine kleinere Chunk-Size zahlt sich dafür bei RAID-Devices aus, die mit vielen kleinen Dateien belastet werden. Somit bleibt auch der »Verschnitt« in einem erträglichen Rahmen. Die Chunk-Size sollte also jeder an seine jeweiligen Bedürfnisse anpassen.

Der Chunk Size Parameter in der `/etc/raidtab` funktioniert für alle RAID-Modi und wird in Kilobyte angegeben. Ein Eintrag von »4« bedeutet also 4096 Byte. Mögliche Werte für die Chunk-Size sind 4 kB bis 128 kB, wobei sie immer einer 2er Potenz entsprechen müssen.

Wie wirkt sich die Chunk-Size jetzt speziell auf die RAID-Modi aus?

Linear Modus

Beim Linear Modus wirkt sich die Chunk-Size mehr oder minder direkt auf die benutzten Daten aus. Allgemein gilt hier, bei vielen kleinen Dateien eher eine kleine Chunk-Size zu wählen und umgekehrt.

RAID-0

Da die Daten auf ein RAID-0 »parallel« geschrieben werden, bedeutet hier eine Chunk-Size von 4 kB, daß bei einem Schreibprozeß mit einer Größe von 16 kB in einem RAID-Verbund aus zwei Partitionen die ersten beiden Blöcke parallel auf die beiden Partitionen geschrieben werden und anschließend die nächsten beiden wieder parallel. Hier kann man auch erkennen, das die Chunk-Size in engem Zusammenhang mit der verwendeten Anzahl der Partitionen steht. Eine generelle optimale Einstellung kann man also nicht geben. Diese hängt vielmehr vom Einsatzzweck des RAID-Arrays in bezug auf die Größe der hauptsächlich verwendeten Dateien, der Anzahl der Partitionen und den Einstellungen des Dateisystems ab.

RAID-1

Beim Schreiben auf ein RAID-1 Device ist die verwendete Chunk-Size für den »parallelen« Schreibprozeß unerheblich, da sämtliche Daten auf beide Partitionen geschrieben werden müssen. Die Abhängigkeit besteht hier also wie beim Linear-Modus von den verwendeten Dateien. Beim Lesevorgang allerdings bestimmt die Chunk-Size, wie viele Daten zeitgleich von den unterschiedlichen Partitionen gelesen werden können. Der Witz ist hierbei, daß alle Partitionen dieselben Daten enthalten (Spiegel-Partitionen eben) und dadurch Lesevorgänge wie eine Art RAID-0 arbeiten. Hier ist also mit einem Geschwindigkeitsgewinn zu rechnen.

RAID-4

Hier beschreibt die Chunk-Size die Größe der Paritätsblöcke auf der Paritäts-Partition, welche nach dem eigentlichen Schreibzugriff geschrieben werden. Wird auf einen RAID-4 Verbund 1 Byte geschrieben, so werden die Bytes, die die Blockgröße bestimmen, von den RAID-4 Partitionen abzüglich der Paritäts-Partition (X-1, wobei X die Gesamtzahl der RAID-4 Partitionen ist) gelesen, die Paritäts-Information berechnet und auf die Paritäts-Partition geschrieben. Die Chunk-Size hat auf den Lesevorgang denselben geschwindigkeitsgewinnenden Einfluß wie bei einem RAID-0 Verbund, da der Lesevorgang auf eine ähnliche Weise realisiert ist.

RAID-5

Beim RAID-5 Verbund bezeichnet die Chunk-Size denselben Vorgang wie beim RAID-4. Eine allgemein vernünftige Chunk-Size läge hier zwischen 32 kB und 128 kB, jedoch treffen auch hier die Faktoren wie Nutzung des RAIDs und verwendete Partitionen auf den Einzelfall zu und die Chunk-Size sollte dementsprechend angepaßt werden.

RAID-10

Bei dieser Art des RAID-Verbundes bezieht sich die Chunk-Size auf alle integrierten RAID-Verbunde, also beide RAID-0 Verbunde und das RAID-1 Array. Als Richtwert kann man hier 32 kB angeben - experimentieren ist hierbei natürlich wie bei allen anderen RAID-Modi auch ausdrücklich erlaubt.

8.2 Spezielle Optionen von mke2fs für RAID-4 und RAID-5 Systeme

Die Option »-R stride=nn« von `mke2fs` erlaubt es, verschiedene ext2 spezifische Datenstrukturen auf eine intelligentere Weise auf das RAID zu verteilen. Ist die Chunk-Size mit 32 kB angegeben, heißt das, daß 32 kB fortlaufende Daten als Block auf dem RAID-Verbund liegen. Danach würde der nächste 32 kB Block kommen. Will man ein ext2 Dateisystem mit 4 kB Blockgröße erstellen, erkennt man, daß acht Dateisystemblöcke in einem Verbundblock (Chunk Block; durch Chunk-Size angegeben) untergebracht werden. Diese Information kann man dem ext2 Dateisystem beim Erstellen mitteilen:

```
mke2fs -b 4096 -R stride=8 /dev/md0
```

Die RAID-4 und RAID-5 Performance wird durch diesen Parameter erheblich beeinflußt. Das Benutzen dieser Option ist dringend anzuraten.

8.3 Beispiel-raidtab Einträge für alle RAID-Modi

Hier werden nur der neue RAID-Kernel-Patch in Verbindung mit den passenden RAID-Tools Version 0.9x und der entsprechenden Kernel-Version beschrieben. Die älteren RAID-Tools der Version 0.4x finden keine Berücksichtigung. Der Kernel-Patch sollte installiert sein. Ebenso sollten die RAID-Optionen im Kernel aktiviert sein und der Kernel neu kompiliert und neu gebootet sein.

8.3.1 Linear (Append) Modus

Der Linear Modus verbindet mehrere Partitionen unterschiedlicher Größe zu einem Gesamtverbund, der allerdings nicht parallel, sondern nacheinander beschrieben wird. Ist die erste Partition voll, wird auf der nächsten weitergeschrieben. Dadurch sind weder Performancevorteile zu erwarten noch eine gesteigerte Sicherheit. Im Gegenteil. Ist eine Partition des Linear-Verbundes defekt, ist meist auch der gesamte Verbund hinüber.

Die Parameter der `/etc/raidtab` für einen Linear-Verbund sehen so aus:

```
raiddev /dev/md0
        raid-level              linear
        nr-raid-disks           2
        persistent-superblock   1
        chunk-size              4

        device                  /dev/sda6
        raid-disk               0
        device                  /dev/sdb6
        raid-disk               1
```

Zu beachten ist hier, daß der Linear Modus keine Spare-Disks unterstützt. Was das ist und was man unter dem Persistent-Superblock versteht, finden Sie im Abschnitt 9 (Weitere Optionen der neuen RAID-Patches). Der Parameter »Chunk-Size« wurde bereits in diesem Abschnitt weiter oben erläutert.

Initialisiert wird Ihr neues Device mit folgendem Kommando:

```
mkraid -f /dev/md0
```

Danach fehlt noch ein Dateisystem:

```
mke2fs /dev/md0
```

Schon können Sie das Device überall hin mounten und in die `/etc/fstab` eintragen.

8.3.2 RAID-0 (Striping)

Sie haben sich entschlossen, mehrere Partitionen unterschiedlicher Festplatten zusammenzufassen, um eine Geschwindigkeitssteigerung zu erzielen. Auf die Sicherheit legen Sie dabei weniger Wert, jedoch sind Ihre Partitionen alle annähernd gleich groß.

Ihre `/etc/raidtab` sollte daher etwa so ausehen:

```
raiddev /dev/md0
        raid-level              0
        nr-raid-disks           2
        persistent-superblock   1
        chunk-size              4

        device                  /dev/sda6
        raid-disk               0
        device                  /dev/sdb6
        raid-disk               1
```

Hier werde genauso wie beim Linear Modus keine Spare-Disks unterstützt. Was das ist und was man unter dem Persistent-Superblock versteht, wird im Abschnitt 9 (Weitere Optionen der neuen RAID-Patches) erläutert.

Initialisiert wird Ihr neues Device mit folgendem Kommando:

```
mkraid -f /dev/md0
```

Danach fehlt noch ein Dateisystem:

```
mke2fs /dev/md0
```

Schon können Sie das Device überall hin mounten und in die `/etc/fstab` eintragen.

8.3.3 RAID-1 (Mirroring)

Ein RAID-1 Verbund wird auch als »Spiegelsystem« bezeichnet, da hier der gesamte Inhalt einer Partition auf eine oder mehrere andere gespiegelt und damit eins zu eins dupliziert wird. Wir haben hier also den ersten Fall von Redundanz. Desweiteren können hier, falls es erwünscht ist, zum ersten mal Spare-Disks zum Einsatz kommen. Diese liegen pauschal erstmal brach im System und werden erst um Ihre Mitarbeit bemüht, wenn eine Partition des RAID-1 Verbundes ausgefallen ist. Spare-Disks bieten also einen zusätzlichen Ausfallschutz, um nach einem Ausfall möglichst schnell wieder ein redundantes System zu bekommen.

Die `/etc/raidtab` sieht in solch einem Fall inkl. Spare-Disk so aus:

```
raiddev /dev/md0
        raid-level              1
        nr-raid-disks           2
        nr-spare-disks          1
        chunk-size              4
        persistent-superblock   1

        device                  /dev/sda6
        raid-disk               0
        device                  /dev/sdb6
        raid-disk               1
        device                  /dev/sdc6
        spare-disk              0
```

Weitere Spare- oder RAID-Disks würden analog hinzugefügt werden. Führen Sie hier nun folgendes Kommando aus:

```
mkraid -f /dev/md0
```

Mittels `cat /proc/mdstat` können Sie wieder den Fortschritt und den Zustand Ihres RAID-Systems erkennen. Erstellen Sie das Dateisystem mittels

```
mke2fs /dev/md0
```

sobald das RAID sich synchronisiert hat.

Theoretisch funktioniert das Erstellen des Dateisystems bereits während sich das RAID-1 noch synchronisiert, jedoch ist es vorläufig sicherer, mit dem Formatieren und Mounten zu warten, bis das RAID-1 fertig ist.

8.3.4 RAID-4 (Striping & Dedicated Parity)

Sie möchten mehrere, aber mindestens drei etwa gleich große Partitionen zusammenfassen, die sowohl einen Geschindigkeitsvorteil als auch erhöhte Sicherheit bieten sollen. Das Verfahren der Datenverteilung beim Schreibzugriff ist hierbei genauso wie beim RAID-0 Verbund, allerdings kommt hier eine zusätzliche Partition mit Paritätsinformationen hinzu. Fällt eine Partition aus, so kann diese, falls eine Spare-Disk vorhanden ist, sofort wieder rekonstruiert werden; fallen zwei Partitionen aus, ist aber auch hier Schluß und die Daten sind verloren. Obwohl RAID-4 Verbunde eher selten eingestezt werden, müßte Ihre `/etc/raidtab` dann so aussehen:

```
raiddev /dev/md0
        raid-level              4
        nr-raid-disks           3
        nr-spare-disks          0
        persistent-superblock   1
        chunk-size              32

        device                  /dev/sda6
        raid-disk               0
        device                  /dev/sdb6
        raid-disk               1
        device                  /dev/sdc6
        raid-disk               2
```

Möchten Sie Spare-Disks einsetzen, so werden sie analog der Konfiguration des RAID-1 Devices hinzugefügt, also:

```
        nr-spare-disks  1
        device          /dev/sdd6
        spare-disk      0
```

Der Grund dafür, daß RAID-4 Verbunde nicht oft eingesetzt werden, liegt daran, daß die Paritäts-Partition immer die gesamten Daten des restlichen - als RAID-0 arbeitenden - Verbundes schreiben muß. Denkt man sich einen Extremfall, wo zehn Partitionen als RAID-0 arbeiten und eine Partition nun die gesamten Paritätsinformationen speichern soll, so wird unweigerlich deutlich, daß die Paritäts-Partition schnell überlastet ist.

Initialisiert wird Ihr neues Device so:

```
mkraid -f /dev/md0
```

Danach fehlt noch ein Dateisystem:

```
mke2fs /dev/md0
```

Schon können Sie das Device überall hin mounten und in die `/etc/fstab` eintragen.

8.3.5 RAID-5 (Striping & Distributed Parity)

Ein RAID-5 Verbund löst nun das klassische RAID-4 Problem elegant, indem es die Paritätsinformationen gleichmäßig über alle im RAID-5 Verbund laufenden Partitionen verteilt. Hierdurch wird der Flaschenhals einer einzelnen Paritäts-Partition wirksam umgangen, weshalb sich RAID-5 als Sicherheit und Geschwindigkeit bietender RAID-Verbund großer Beliebtheit erfreut. Fällt eine Partition aus, beginnt das RAID-5, falls Spare-Disks vorhanden sind, sofort mit der Rekonstruktion der Daten, allerdings kann RAID-5 auch nur den Verlust einer Partition verkraften. Genauso wie beim RAID-4 sind beim zeitgleichen Verlust zweier Partitionen alle Daten verloren. Eine `/etc/raidtab` für ein RAID-5 Device sähe folgendermaßen aus:

```
raiddev /dev/md0
        raid-level              5
        nr-raid-disks           3
        nr-spare-disks          2
        persistent-superblock   1
        parity-algorithm        left-symmetric
        chunk-size              64

        device                  /dev/sda6
        raid-disk               0
        device                  /dev/sdb6
        raid-disk               1
        device                  /dev/sdc6
        raid-disk               2
        device                  /dev/sdd6
        spare-disk              0
        device                  /dev/sde6
        spare-disk              1
```

Bei diesem Beispiel sind gleich zwei Spare-Disks mit in die Konfiguration aufgenommen worden.

Der Parameter »parity-algorithm« legt die Art der Ablage der Paritätsinformationen fest und kann nur auf RAID-5 Verbunde angewendet werden. Die Auswahl »left-symmetric« bietet die maximale Performance. Weitere Möglichkeiten sind »left-asymmetric«, »right-asymmetric« und »right-symmetric«.

Initialisiert wird Ihr neues Device so:

```
mkraid -f /dev/md0
```

Danach fehlt noch ein Dateisystem:

```
mke2fs /dev/md0
```

Schon können Sie das Device überall hin mounten und in die `/etc/fstab` eintragen.

8.3.6 RAID-10 (Mirroring & Striping)

Die Kombination aus RAID-1 und RAID-0 Verbunden kann sehr flexibel eingesetzt werden, ist aber mit Vorsicht zu genießen. Man muß hierbei genau darauf achten, welche RAID-Partitionen in welchen RAID-Verbund eingebaut werden sollen. Um allerdings die nötige Redundanz gewährleisten zu können, sind hierfür mindestens vier RAID-Partitionen auf unterschiedlichen Festplatten nötig. Als Beispiel erstellen wir zwei RAID-0 Verbunde über jeweils zwei verschiedene RAID-Partitionen, die anschließend per RAID-1 gespiegelt werden sollen. Eine passende `/etc/raidtab` ohne Spare-Disks sähe dann so aus:

```
raiddev /dev/md0
        raid-level              0
```

```
        nr-raid-disks           2
        nr-spare-disks          0
        persistent-superblock   1
        chunk-size              4

        device                  /dev/sda6
        raid-disk               0
        device                  /dev/sdb6
        raid-disk               1


raiddev /dev/md1
        raid-level              0
        nr-raid-disks           2
        nr-spare-disks          0
        persistent-superblock   1
        chunk-size              4

        device                  /dev/sdc6
        raid-disk               0
        device                  /dev/sdd6
        raid-disk               1


raiddev /dev/md2
        raid-level              1
        nr-raid-disks           2
        nr-spare-disks          0
        persistent-superblock   1
        chunk-size              4

        device                  /dev/md0
        raid-disk               0
        device                  /dev/md1
        raid-disk               1
```

Jetzt gilt es aber ein paar Kleinigkeiten zu beachten, denn anders als bei den anderen RAID-Verbunden haben wir hier gleich drei RAID-Arrays erstellt, wobei man sich überlegen muß, welches denn nun nachher überhaupt gemountet und mit Daten beschrieben werden soll. Die Reihenfolge ergibt sich aus der Datei `/etc/raidtab`. `/dev/md0` wird nacher auf `/dev/md1` gespiegelt.

Jedes Devices muß für sich erstellt werden:

```
mkraid -f /dev/mdx
```

Ein Dateisystem per `mke2fs` wird nur auf `/dev/md0` und `/dev/md1` erstellt.

Die beste Reihenfolge ist, erst das Device `/dev/md0` zu erstellen, zu formatieren und zu mounten. Dann wird `/dev/md1` erstellt und formatiert. Dieses bitte nicht mounten, da ja hier keine Daten drauf geschrieben werden sollen. Zuletzt wird nun mittels

```
mkraid -f /dev/md2
```

das RAID-1 Array erstellt, jedoch sollte man hier wirklich kein Dateisystem erstellen. Ab jetzt kann man mittels

```
cat /proc/mdstat
```

die Synchronisation der beiden RAID-0 Verbunde `/dev/md0` und `/dev/md1` verfolgen. Ist die Synchronisation abgeschlossen, werden alle Daten, die auf `/dev/md0` geschrieben werden, auf `/dev/md1` gespiegelt. Aktiviert, gemountet und in die Datei `/etc/fstab` eingetragen wird letzthin nur `/dev/md0`. Natürlich könnte man auch `/dev/md1` mounten, jedoch sollte man sich für ein Device entscheiden. Ausgeschlossen ist allerdings das Mounten von `/dev/md2`.

9 Weitere Optionen des neuen RAID-Patches

9.1 Autodetection

Die Autodetection beschreibt einen Kernel-Parameter, der in allen beschriebenen Kernel-Vorbereitungen als zu aktivieren gekennzeichnet wurde. Er erlaubt das automatische Erkennen und Starten der diversen im System vorhandenen RAID-Verbunde schon während des Bootvorganges von Linux und somit auch die Nutzung eines RAID-Verbundes als Root-Partition.

Näheres zur Nutzung eines RAID-Verbundes als Root-Partition finden Sie im Abschnitt 7 (Root-Partition oder Swap-Partition als RAID).

9.2 Persistent-Superblock

Diese überaus nützliche Option ist uns nun in jeder `/etc/raidtab` Konfiguration über den Weg gelaufen und mit dem Wert »1« eingetragen gewesen, doch was bedeutet er?

Erinnern Sie sich noch an die MD-Tools oder gar an die mdtab? Mit diesen älteren Tools wurde eine Datei `/etc/mdtab` erstellt, die in älteren RAID-Unterstützungen die Konfiguration Ihres RAID-Verbundes inklusiv einer Prüfsumme enthielt.

Sollte nun ein RAID-Verbund gestartet werden, so mußte der Kernel erstmal diese Datei auslesen, um überhaupt zu erfahren, wo er welches RAID mit welchen Partitionen zu starten hatte. Haben Sie den Abschnitt über die Root-Partition als RAID gelesen, so ahnen Sie es schon: Um an diese Datei heranzukommen, muß aber erstmal das darunterliegende Dateisystem laufen. Eine Zeit lang war es mit der neueren Konfigurationsdatei `/etc/raidtab` genauso, aber hier nun tritt der Parameter »persistent-superblock« in Aktion. Die möglichen Werte dafür sind »0« und »1«. Ist der Wert auf »0« gesetzt, so verhält sich das Starten der RAIDs gemäß dem oben beschriebenen Vorgang. Ist er allerdings auf »1« gesetzt, wird beim Erstellen jedes neuen RAID-Verbundes an das Ende jeder Partition ein spezieller Superblock geschrieben, der es dem Kernel erlaubt, die benötigten Informationen über das RAID direkt von den jeweiligen Partitionen zu lesen, ohne ein Dateisystem gemountet zu haben. Trotzdem sollten Sie immer eine `/etc/raidtab` pflegen und beibehalten. Ist im Kernel die Autodetection aktiviert, so werden die RAID-Arrays mit aktiviertem Persistent-superblock sogar direkt gestartet. Dies befähigt Sie, ganz simpel jedes RAID-Array als `/dev/md0`, `/dev/md1` usw. einfach und problemlos in die `/etc/fstab` zu setzen. Der Kernel kümmert sich um das Aktivieren beim Startup, wodurch ein

```
raidstart /dev/md0
```

ebenso wie ein

```
raidstop /dev/md0
```

beim Systemhalt überflüssig ist.

Abgesehen davon ermöglicht diese Option auch das Booten von einem RAID-4 oder RAID-5 Array als Root-Partition. Näheres zur Einrichtung eines RAID-Arrays als Root-Partition finden Sie im Abschnitt 7 (Root-Partition oder Swap-Partition als RAID).

9.3 Spare-Disks

Spare-Disks bezeichnen Festplatten-Partitionen, die mit Hilfe der `/etc/raidtab` zwar schon einem bestimmten RAID-Verbund zugewiesen wurden, jedoch solange nicht benutzt werden, bis irgendwann mal eine Partition ausfällt. Dann allerdings wird die defekte Partition sofort durch die Spare-Disk ersetzt und die Rekonstruktion der Daten aus den Paritätsinformationen wird gestartet. Den Fortschritt dieser Rekonstruktion können Sie - wie alles über RAID-Devices - mittels

```
cat /proc/mdstat
```

nachvollziehen. Eine Spare-Disk sollte mindestens genauso groß oder größer sein als die anderen verwendeten RAID-Partitionen. Ist eine Spare-Disk kleiner als die verwendeten RAID-Partitionen und ist der RAID-Verbund fast voll mit Daten, so kann bei einem Ausfall einer RAID-Partition die Spare-Disk natürlich nicht alle Daten aufnehmen, was unweigerlich zu Problemen führt.

9.4 Spare-Disks und Hot Plugging

Dank der neuen RAID-Tools ist es auch möglich, eine Spare-Disk nachträglich in einen bereits vorhandenen RAID-Verbund einzufügen. Sinnvoll ist dieser Vorgang natürlich nur bei RAID-Modi, welche auch mit Spare-Disks umgehen können. Erweitern Sie dazu erst Ihre `/etc/raidtab` um die neue Spare-Disk. Dies ist zwar für den eigentlichen Vorgang nicht zwingend notwendig, jedoch empfiehlt es sich immer, eine sorgfältig gepflegte RAID-Konfigurationsdatei zu besitzen. Führen Sie dann zum Beispiel den Befehl

```
raidhotadd /dev/md2 /dev/sde4
```

aus, um in Ihren dritten RAID-Verbund die vierte Partition Ihrer fünften SCSI-Festplatte einzufügen. Der Befehl `raidhotadd` fügt die neue Partition automatisch als Spare-Disk ein und aktualisiert auch gleich die Superblöcke aller in diesem RAID-Verbund befindlichen Partitionen. Somit brauchen Sie keine weiteren Schritte zu unternehmen, um die neue Spare-Disk Ihren anderen RAID-Partitionen als nutzbar bekannt zu geben.

Diesen Befehl kann man sich auch zu Nutze machen, einen intakten RAID-Verbund dazu zu bewegen, die Superblöcke neu zu schreiben oder sich zu synchronisieren. Der analog zu verwendende Befehl `raidhotremove` entfernt eine so hinzugefügte Spare-Disk natürlich auch wieder aus dem RAID-Array.

10 Fehlerbehebung

An dieser Stelle wird auf das Verhalten der unterschiedlichen RAID-Verbunde im Fehlerfall eingegangen. Die Szenarien zum Restaurieren defekter RAID-Verbunde reichen von den verschiedenen RAID-Modi bis hin zu Hot Plugging und Spare-Disks.

Die folgenden Beschreibungen stützen sich alle auf den Kernel 2.2.10 mit den passenden RAID-Tools und Kernel-Patches, entsprechen also der Einrichtfunktionalität des Abschnittes über die neuen RAID-Tools

Version 0.9x. Die RAID-Verbunde, welche mit den MD-Tools unter der DLD 6.0 und DLD 6.01 eingerichtet und damit quasi auf die alte Art mit den RAID-Tools Version 0.4x erstellt wurden, werden mangels ausgiebieger Tests nicht berücksichtigt.

Da sich viele Möglichkeiten der Synchronisation und der Überwachung von RAID-Verbunden auf spezielle Funktionen und Optionen der neuen RAID-Tools beziehen, kann man die im folgenden geschilderten Prozeduren nicht mal näherungsweise auf RAID-Verbunde, die mit den alten RAID-Tools Version 0.4x eingerichtet wurden, beziehen. Solange das noch nicht getestet wurde, sind Sie hier auf sich gestellt.

10.1 Linear (Append) Modus und RAID-0 (Stripping)

Hier ist die Möglichkeit der Rekonstruktion von Daten schnell erklärt. Da diese RAID-Modi keine Redundanz ermöglichen, sind beim Ausfall einer Festplatte alle Daten verloren. Definitiv gilt das für RAID-0; beim Linear Modus können Sie eventuell mit viel Glück noch einige Daten einer RAID-Partition sichern, so es denn die erste Partition ist und Sie irgendwie in der Lage sind, die Partition einzeln zu Mounten. Da RAID-0 die Daten parallel schreibt, erhalten Sie - auch wenn Sie eine Partition des RAID-0 Verbundes gemountet bekommen - niemals mehr als einen zusammenhängenden Block. Meiner Meinung nach ist an dieser Stelle eine Datenrettung von einem RAID-0 Verbund sehr sehr schwierig.

10.2 RAID-1, RAID-4 und RAID-5 ohne Spare-Disk

Alle drei RAID-Modi versprechen Redundanz. Doch was muß gezielt getan werden, wenn hier eine Festplatte oder eine RAID-Partition ausfällt? Generell bieten sich einem zwei Wege, um die defekte Festplatte durch eine neue zu ersetzen. Bedenken Sie bei dieser Beschreibung, daß RAID-Partition und Festplatte synonym verwendet wird. Ich gehe von einer Partition je Festplatte aus. Die eine Methode beschreibt den »heißen« Weg. Hierbei wird die Festplatte im laufenden Betrieb ausgetauscht. Die andere Methode beschreibt entsprechend den sicheren Weg. Beide Möglichkeiten haben für die RAID-Modi 1, 4 und 5 funktioniert. Seien Sie insbesondere mit dem »heißen« Weg trotzdem sehr vorsichtig, da die Benutzung dieser Befehle nicht Hot Plugging fähige Hardware zerstören könnte.

10.2.1 Der »heiße« Weg (Hot Plugging)

Vorab muß gesagt werden, daß Hot Plugging unter Linux auch vom SCSI-Treiber unterstützt werden muß, versuchen Sie Hot Plugging aber niemals mit (E)IDE Laufwerken. Zwar sollen alle Linux SCSI-Treiber des hier verwendeten 2.2.10er Kernels Hot Plugging unterstützen, jedoch ist dies nur bei dem Adaptec und Symbios Treiber sicher der Fall. Auch sollte Ihre Hardware und damit Ihr SCSI Kontroller und Ihre SCSI-Festplatten Hot Plugging unterstützen. Wie man das herausbekommt, ist schwer zu sagen, allerdings hat es auch mit einem fünf Jahre alten Symbios Kontroller und mehreren IBM-DCAS-UW Festplatten funktioniert. Generell ist die meiste im Consumer Bereich verfügbare Hardware nicht Hot Plugging fähig. Die zu Testzwecken eingesetzten UW-Wechselrahmen kosten etwa 125,- EUR und haben alle Hot Plugging Tests klaglos verkraftet.

Haben Sie irgendwelche Zweifel und sind nicht gezwungen, Hot Plugging zu nutzen, dann lesen Sie hier gar nicht erst weiter, sondern springen Sie sofort zu der »sicheren« Beschreibung. Auch ergeben sich aus meinen erfolgreichen Tests keine Garantien für irgendeinen Erfolg. Mit allem Nachdruck muß daher an dieser Stelle auf die Möglichkeit der Zerstörung Ihrer Hardware durch die Benutzung von Hot Plugging mit ungeeigneter Hardware hingewiesen werden.

Ihr RAID-1, 4 oder 5 ist gemäß einer der vorherigen Anleitungen erstellt worden und betriebsbereit. Um den ganzen Ablauf zu vereinfachen, wird ab jetzt nur noch von einem RAID-5 ausgegangen und auch die Beispielauszüge mittels `cat /proc/mdstat` beschreiben ein RAID-5; RAID-1 und RAID-4 Benutzer können analog verfahren, richten aber bitte eine erhöhte Aufmerksamkeit auf die Unterscheidung und Abstraktion der Meldungsparameter.

Das Beispiel-RAID-5 ist in der `/etc/raidtab` so eingetragen:

```
raiddev /dev/md0
        raid-level              5
        nr-raid-disks           4
        nr-spare-disks          0
        parity-algorithm        left-symmetric
        persistent-superblock   1
        chunk-size              32

        device                  /dev/sda5
        raid-disk               0
        device                  /dev/sdb6
        raid-disk               1
        device                  /dev/sdc6
        raid-disk               2
        device                  /dev/sdd1
        raid-disk               3
```

`cat /proc/mdstat` meldet einen ordentlich laufenden RAID-5-Verbund:

```
Personalities : [linear] [raid0] [raid1] [raid5] [hsm]
read_ahead 1024 sectors
md0 : active raid5 sdd1[3] sdc6[2] sdb6[1] sda5[0] 633984 blocks
 level 5, 32k chunk, algorithm 2 [4/4] [UUUU]
unused devices: <none>
```

Jetzt fällt die Festplatte `/dev/sdd` komplett aus. Beim nächsten Zugriff auf das RAID-5 wird der Mangel erkannt, der SCSI-Bus wird resetet und das RAID-5 läuft relativ unbeeindruckt weiter. Die defekte RAID-Partition wird lediglich mit »(F)« gekennzeichnet und fehlt in den durch ein »U« markierten gestarteten RAID-Partitionen:

```
Personalities : [linear] [raid0] [raid1] [raid5] [hsm]
read_ahead 1024 sectors
md0 : active raid5 sdd1[3](F) sdc6[2] sdb6[1] sda5[0] 633984 blocks
 level 5, 32k chunk, algorithm 2 [4/3] [UUU_]
unused devices: <none>
```

Jetzt möchten Sie wieder ein redundantes System bekommen, ohne den Rechner neu booten zu müssen und ohne Ihr gemountetes RAID-5 stoppen zu müssen. Entfernen Sie dafür die defekte RAID-Partition (`/dev/sdd1`) mittels des bei den RAID-Tools beiliegenden Hilfsprogramms `raidhotremove` aus dem aktiven RAID-Verbund (`/dev/md0`):

```
raidhotremove /dev/md0 /dev/sdd1
```

Ein `cat /proc/mdstat` hat sich nun dahingehend verändert, daß die defekte Partition komplett rausgenommen wurde und Ihnen eine fehlende Partition, um Redundanz zu gewährleisten, mit »[UUU_]« angezeigt wird:

```
Personalities : [linear] [raid0] [raid1] [raid5] [hsm]
read_ahead 1024 sectors
md0 : active raid5 sdc6[2] sdb6[1] sda5[0] 633984 blocks level 5,
 32k chunk, algorithm 2 [4/3] [UUU_]
unused devices: <none>
```

Tauschen Sie jetzt Ihre hoffentlich in einem guten Wechselrahmen befindliche defekte Festplatte gegen eine neue aus und erstellen Sie eine Partition darauf. Anschließend geben Sie den Befehl, diese neue RAID-Partition wieder in das laufende Array einzubinden:

```
raidhotadd /dev/md0 /dev/sdd1
```

Der Daemon `raid5d` macht sich nun daran, die neue RAID-Partition mit den anderen zu synchronisieren:

```
Personalities : [linear] [raid0] [raid1] [raid5] [hsm]
read_ahead 1024 sectors
md0 : active raid5 sdd1[4] sdc6[2] sdb6[1] sda5[0] 633984 blocks
 level 5, 32k chunk, algorithm 2 [4/3] [UUU_] recovery=7%
finish=4.3min
unused devices: <none>
```

Der Abschluß dieses Vorganges wird mit einem ebenso lapidaren wie beruhigendem »resync finished« quittiert. Eine Kontrolle ergibt tatsächlich ein wieder vollständig redundantes und funktionstüchtiges RAID-5:

```
Personalities : [linear] [raid0] [raid1] [raid5] [hsm]
read_ahead 1024 sectors
md0 : active raid5 sdd1[3] sdc6[2] sdb6[1] sda5[0] 633984 blocks
 level 5, 32k chunk, algorithm 2 [4/4] [UUUU]
unused devices: <none>
```

Als ob nicht gewesen wäre.

10.2.2 Der sichere Weg

Diese Methode sieht einen Wechsel einer defekten Festplatte in einem System vor, das ruhig für einige Zeit heruntergefahren werden kann.

Ihr RAID-1, 4 oder 5 ist gemäß einer der vorherigen Anleitungen erstellt worden und betriebsbereit. Um den ganzen Ablauf zu vereinfachen, wird ab jetzt nur noch von einem RAID-5 ausgegangen und auch die Beispielauszüge mittels `cat /proc/mdstat` beschreiben ein RAID-5; RAID-1 und RAID-4 Benutzer können analog verfahren, richten aber bitte eine erhöhte Aufmerksamkeit auf die Unterscheidung und Abstraktion der Meldungsparameter.

Das Beispiel RAID-5 ist in der `/etc/raidtab` so eingetragen:

```
raiddev /dev/md0
        raid-level              5
        nr-raid-disks           4
        nr-spare-disks          0
        parity-algorithm        left-symmetric
        persistent-superblock   1
        chunk-size              32
```

```
device                  /dev/sda5
raid-disk               0
device                  /dev/sdb6
raid-disk               1
device                  /dev/sdc6
raid-disk               2
device                  /dev/sdd1
raid-disk               3
```

`cat /proc/mdstat` meldet einen ordentlich laufenden RAID-5-Verbund:

```
Personalities : [linear] [raid0] [raid1] [raid5] [hsm]
read_ahead 1024 sectors
md0 : active raid5 sdd1[3] sdc6[2] sdb6[1] sda5[0] 633984 blocks
 level 5, 32k chunk, algorithm 2 [4/4] [UUUU]
unused devices: <none>
```

Jetzt fällt die Festplatte `/dev/sdd` komplett aus. Beim nächsten Zugriff auf das RAID-5 wird der Mangel erkannt, der SCSI-Bus wird resetet und das RAID-5 läuft relativ unbeeindruckt weiter. Die defekte RAID-Partition wird lediglich mit »(F)« gekennzeichnet:

```
Personalities : [linear] [raid0] [raid1] [raid5] [hsm]
read_ahead 1024 sectors
md0 : active raid5 sdd1[3](F) sdc6[2] sdb6[1] sda5[0] 633984 blocks
 level 5, 32k chunk, algorithm 2 [4/3] [UUU_]
unused devices: <none>
```

Jetzt möchten Sie wieder ein redundantes System bekommen. Fahren Sie dazu Ihren Rechner herunter und tauschen Sie die defekte Festplatte gegen eine neue aus. Nach dem Startup müssen Sie auf der neuen Festplatte eine entsprechende Partition möglichst auch mit der RAID-Autostart Option durch das »fd« Flag einrichten. Ist die neue Partitionsangabe identisch mit der auf der defekten Festplatte, brauchen Sie nichts weiter zu machen, ansonsten müssen Sie noch Ihre `/etc/raidtab` anpassen. Weiterhin ist es erforderlich, daß Ihr RAID-Verbund läuft. Ist das nicht bereits während des Bootvorganges erfolgt, müssen Sie selbiges jetzt nachhlen:

```
raidstart /dev/md0
```

Jetzt fehlt noch der Befehl, um die neue RAID-Partition wieder in das RAID-5 einzuarbeiten und die »persistent-superblocks« neu zu schreiben. Hierbei werden keine vorhandenen Daten auf dem bestehenden RAID-5 Verbund zerstört, es sei denn, Sie haben sich bei der eventuell nötigen Aktualisierung der `/etc/raidtab` vertan. Prüfen Sie alles dringend nochmal. Ansonsten:

```
raidhotadd /dev/md0 /dev/sdd1
```

Dieser Befehl suggeriert zwar, daß hier eine Art Hot Plugging stattfindet, heißt in diesem Zusammenhang aber nichts anderes, als daß eine RAID-Partition in ein vorhandenes RAID-Array eingearbeitet wird. Würden Sie stattdessen den Befehl `mkraid` mit seinen Parametern aufrufen, würde dies zwar auch zu dem Erfolg führen, ein neues RAID-Array zu erstellen, jedoch dummerweise ohne Daten und damit natürlich auch und vor allem ohne die bisher vorhandenen Daten.

Inspizieren Sie anschließend den Verlauf wieder mittels `cat /proc/mdstat`:

```
Personalities : [linear] [raid0] [raid1] [raid5] [hsm]
read_ahead 1024 sectors
md0 : active raid5 sdd1[3] sdc6[2] sdb6[1] sda5[0] 633984 blocks
 level 5, 32k chunk, algorithm 2 [4/4] [UUUU] resync=57%
finish=1.7min
unused devices: <none>
```

Der Abschluß dieses Vorganges wird mit einem ebenso lapidaren wie beruhigendem »resync finished« quittiert. Eine Kontrolle ergibt tatsächlich ein wieder vollständig redundantes und funktionstüchtiges RAID-5:

```
Personalities : [linear] [raid0] [raid1] [raid5] [hsm]
read_ahead 1024 sectors
md0 : active raid5 sdd1[3] sdc6[2] sdb6[1] sda5[0] 633984 blocks
 level 5, 32k chunk, algorithm 2 [4/4] [UUUU]
unused devices: <none>
```

Das RAID-5 Array muß nun lediglich wieder in den Verzeichnisbaum gemountet werden, falls Ihr RAID-Array nicht bereits innerhalb der `/etc/fstab` während des Bootvorganges gemountet wurde:

```
mount /dev/md0 /mount-point
```

Hiermit haben Sie wieder ein vollständig redundantes und lauffähiges RAID-5 Array hergestellt.

10.3 RAID-1, RAID-4 und RAID-5 mit Spare-Disk

10.3.1 Der »heiße« Weg (Hot Plugging)

Um bei einem RAID-1, 4 oder 5 Verbund mit einer Spare-Disk per Hot Plugging, also ohne den RAID-Verbund auch nur herunterzufahren, eine RAID-Partition per `raidhotremove` aus dem laufenden Verbund zu entfernen und durch eine neue RAID-Partition zu ersetzen, sind die aktuellsten RAID-Tools erforderlich. Erst diese haben durch das neue Programm `raidsetfaulty` die Möglichkeit, die ausgefallene RAID-Partition als defekt zu markieren und so den Befehl `raidhotremove` zu ermöglichen. Zu beachten ist hier, daß bei einem Festplattenausfall die Spare-Disk sofort eingearbeitet und das System anschließend auch wieder Redundant ist und somit natürlich nicht die Spare-Disk, sondern die ausgefallene RAID-Partition als defekt markiert, ausgetauscht und als neue Spare-Disk wieder eingesetzt werden muß.

Auch an dieser Stelle muß auf die Gefahr der teilweisen oder vollständigen Zerstörung Ihrer Hardware hingewiesen werden, sollte diese nicht Hot Plugging fähig sein. Haben Sie die Möglichkeit zu wählen, benutzen Sie immer die sichere Methode.

10.3.2 Der sichere Weg

Ein normal laufender RAID-5 Verbund mit Spare-Disk sollte bei einem RAID-Verbund `/dev/md0` mit den RAID-Partitionen `/dev/sdb1`, `/dev/sdc1` und `/dev/sdd1` plus der Spare-Disk `/dev/sde1` unter `/proc/mdstat` folgendes zeigen:

```
Personalities : [raid5]
read_ahead 1024 sectors
md0 : active raid5 sde1[3] sdd1[2] sdc1[1] sdb1[0] 782080 blocks
 level 5, 32k chunk, algorithm 2 [3/3] [UUU]
unused devices: <none>
```

Durch das Entriegeln des Wechselrahmens wird ein Defekt der Partition `/dev/sdc1` simuliert. Sobald wieder auf den RAID-5 Verbund zugegriffen wird, wird der Defekt bemerkt, der SCSI-Bus resetet und der Recovery-Prozeß über den `raid5d` Daemon beginnt. Ein `cat /proc/mdstat` zeigt jetzt folgendes:

```
Personalities : [raid5]
read_ahead 1024 sectors
md0 : active raid5 sde1[3] sdd1[2] sdc1[1](F) sdb1[0] 782080 blocks
 level 5, 32k chunk, algorithm 2 [3/2] [U_U] recovery=4%
 finish=15.4min
unused devices: <none>
```

Nach dem erfolgreichen Ende des Recovery-Prozesses liefert `cat /proc/mdstat` folgendes:

```
Personalities : [raid5]
read_ahead 1024 sectors
md0 : active raid5 sde1[3] sdd1[2] sdc1[1](F) sdb1[0] 782080 blocks
 level 5, 32k chunk, algorithm 2 [3/2] [U_U]
unused devices: <none>
```

Den neuen Zustand sollten Sie nun sichern, indem Sie

```
umount /dev/md0
```

ausführen. Mit

```
raidstop /dev/md0
```

wird der aktuelle Zustand auf die RAID-Partitionen geschrieben. Ein

```
raidstart -a
cat /proc/mdstat
```

zeigt dann:

```
Personalities : [raid5]
read_ahead 1024 sectors
md0 : active raid5 sdd1[2] sde1[1] sdb1[0] 782080 blocks level 5,
 32k chunk, algorithm 2 [3/3] [UUU]
unused devices: <none>
```

Wie Sie erkennen können, fehlt die defekte Partition `/dev/sdc1[1](F)`, dafür hat die Spare-Disk `/dev/sde1[1]` deren Funktion übernommen. Das ist jetzt der aktuelle Zustand des RAID-5. Nun wird die defekte Festplatte ersetzt, indem Sie den Rechner herunterfahren und den Festplattenaustausch durchführt. Wenn Sie neu booten, geht zunächst nichts mehr. Nun bloß keine Panik kriegen, sondern erstmal der `/etc/raidtab` Datei den neuen Zustand beibringen:

```
raiddev /dev/md0
        raid-level              5
        nr-raid-disks           3
        nr-spare-disks          1
        persistent-superblock   1
        parity-algorithm        left-symmetric
```

```
chunk-size              32

device                  /dev/sdb1
raid-disk               0
device                  /dev/sde1
raid-disk               1
device                  /dev/sdd1
raid-disk               2
device                  /dev/sdc1
spare-disk              0
```

Anschließend bringt ein

```
raidhotadd /dev/md0 /dev/sdc1
```

dem RAID-Verbund die neue Konstellation bei, ohne dabei die Daten auf dem RAID-5 Verbund zu beschädigen, solange Sie sich in der Datei `/etc/raidtab` nicht vertan haben. Schauen Sie sich die Einträge in Ihrem eigenen Interesse bitte nochmal genau an. Ein `cat /proc/mdstat` zeigt jetzt die Resynchronisation. Man sieht jetzt die neue Zuordnung der RAID-Partitionen im RAID-Verbund, die exakt den neuen Stand der Zuordnung darstellen sollte.

```
Personalities : [raid5]
read_ahead 1024 sectors
md0 : active raid5 sdc1[3] sdd1[2] sde1[19] sdb1[0] 782080 blocks
 level 5, 32k chunk, algorithm 2 [3/3] [UUU] resync=36%
 finish=6.7min
unused devices: <none>
```

Am Ende erscheint:

```
raid5: resync finished
```

Ein `cat /proc/mdstat` sieht nun so aus:

```
Personalities : [raid5]
read_ahead 1024 sectors
md0 : active raid5 sdc1[3] sdd1[2] sde1[1] sdb1[0] 782080 blocks
 level 5, 32k chunk, algorithm 2 [3/3] [UUU]
unused devices: <none>
```

Nun ruft man

```
raidstop /dev/md0
```

auf, um alles auf die Platten zu schreiben. Hat der Kernel das RAID-Array bereits gestartet (persistent-superblock 1), kann man mit einem

```
mount /dev/md0 /mount-point
```

den RAID-Verbund wieder in das Dateisystem einhängen. Ansonsten ist vorher noch folgender Befehl notwendig:

```
raidstart /dev/md0
```

Hiermit haben Sie wieder ein vollständiges laufendes RAID-5 Array hergestellt.

11 Nutzung & Benchmarks

11.1 Wofür lohnt das Ganze denn nun?

11.1.1 Performance

Ein RAID-Device lohnt sich überall dort, wo viel auf die Festplatten zugegriffen wird. So kann zum Beispiel ein RAID-0 oder RAID-5 als `/home` Verzeichnis gemountet werden und das Nächste als `/var` oder `/usr`. Die Geschwindigkeitsvorteile sind gerade bei SCSI Hardware und »festplattenintensiven« Softwarepaketen wie KDE, StarOffice oder Netscape deutlich spürbar. Das ganze System »fühlt« sich erheblich performanter an.

11.1.2 Sicherheit

Mit einem RAID-1 System kann man z.B. die Sicherheit seiner Daten erhöhen. Fällt eine Platte aus, befinden sich die Daten immer noch auf der gespiegelten Partition. Ähnliches gilt z.B. für ein RAID-5 System, welches zusätzlich zur erhöhten Sicherheit noch eine bessere Performance bietet.

11.1.3 Warum nicht?

Hat man schon mehrere Festplatten in seinem System, stellt sich einem doch die Frage, warum man eigentlich solch ein kostenloses Feature wie Software-RAID nicht nutzen sollte. Man denke nur mal an die Preise für gute Hardware-RAID Kontroller plus Speicher. Gerade der neue Kernel-Patch erleichtert einem vieles, was bisher nur auf Umwegen möglich war.

Schreitet die Entwicklung der Software-RAID Unterstützung unter Linux weiterhin so gut fort und bedenkt man die stetig steigende Leistung und die fallenden Preise von CPUs, so kann man sich denken, daß in naher Zukunft die CPU-Leistung, die eine Software-RAID Lösung benötigt, auch bei Standard-CPUs nicht mehr ins Gewicht fällt. Selbst heute reichen für ein Software-RAID auf SCSI-Basis CPUs mit 200-300 MHz völlig aus. Ein RAID-0 mit SCSI-Festplatten soll sogar auf einem 486DX-2/66 halbwegs akzeptabel laufen.

11.2 Vorschläge und Überlegungen zur RAID-Nutzung

11.2.1 Apache (Webserver allgemein)

Aufgrund des hauptsächlichen Lesevorgangs kann hier ein RAID-0 oder RAID-5 Verbund empfohlen werden. Die Log-Dateien sollten allerdings nicht auf einem RAID-5 Verbund liegen. Werden durch dynamische Seiten Daten mit einer Datenbank ausgetauscht, gelten dieselben Überlegungen wie bei dem Datenbankabschnitt.

11.2.2 Log-Dateien

Die typischen Sytem-Log-Dateien erbringen im allgemeinen keine hohe Dauerschreiblast. Hierbei ist es also egal, auf was für einem RAID-Verbund sie liegen. Anders sieht es bei Log-Dateien aus, welche viele Accounting-Informationen enthalten. Diese sollte man aufgrund der allgemeinen Schreibschwäche eines RAID-5 Verbundes eher auf schnellere RAID-Verbunde auslagern.

11.2.3 Oracle (Datenbank)

Oracle kann mit Tablespaces, die aus mehreren Datenfiles auf unterschiedlichen Platten bestehen, umgehen. Um hier zu einer vernünftigen Lastverteilung ohne RAID zu kommen, ist allerdings einiges an Planung vorauszusetzen. Für die Datenfiles und Controlfiles können RAID-0, RAID-1, RAID-10 oder auch RAID-5 Verbunde empfohlen werden; aus Sicherheitsgründen ist allerdings von RAID-0 Verbunden abzuraten. Für die Online-Redo-Log-Dateien empfiehlt sich aufgrund der Schreibschwächen kein RAID-5 System. Dafür könnten folgende Varianten konfiguriert werden:

- einfache Redo-Log-Dateien auf einem RAID-1 oder RAID-10 Verbund
- gespiegelte Redo-Log-Dateien jeweils auf einem RAID-0 Verbund; Oracle kann so konfiguriert werden, daß mehrere parallele Kopien der Redo-Log-Dateien geschrieben werden.

11.2.4 Squid (WWW-Proxy und Cache)

Für Squid im speziellen gilt, daß er von Haus aus mit mehreren Cache-Partitionen umgehen kann. Daher bringt ein darunterliegendes RAID-Array nicht mehr viel. Die Log-Dateien wiederrum ergeben auf einem RAID-0 Verbund durchaus Sinn. Für andere WWW-Proxys wäre hier ein RAID-0 Verbund angebracht.

11.2.5 Systemverzeichnisse

Da hier verhältnismäßig wenig Schreibvorgänge stattfinden, jedoch auf jeden Fall die Redundanz gewährleistet sein sollte, empfiehlt sich für die Root-Partition ein RAID-1 Verbund und für die `/home`, `/usr`, `/var` Verzeichnisse einer der redundanten RAID-Modi (RAID-1, RAID-5, RAID-10).

11.3 Benchmarks

Um einen Vergleich zwischen den RAID-Verbunden mit ihren unterschiedlichen Chunk-Size Parametern ziehen zu können, findet das Programm Bonnie Anwendung, welches im Internet unter

`http://www.textuality.com/bonnie/`

residiert. Bonnie erstellt eine beliebig große Datei auf dem RAID-Verbund und testet neben den unterschiedlichen Schreib- und Lesestrategien auch die anfallende CPU-Last. Allerdings sollte man die Testdatei mindestens doppelt so groß wie den real im Rechner vorhandenen RAM-Speicher wählen, da Bonnie sonst nicht die Geschwindigkeit des RAIDs, sondern das Cacheverhalten von Linux mißt - und das ist garnicht mal so schlecht.

Um die bei dem nicht sequentiellen Lesen von einem RAID-1 Verbund höhere Geschwindigkeit zu testen, eignet sich Bonnie aufgrund seiner einzelnen Testdatei allerdings nicht. Bonnie führt bei seinem Test also einen sequentiellen Schreib-/Lesevorgang durch, welcher nur von einer RAID-1 Festplatte beantwortet wird. Möchte man trotzdem einen RAID-1 Verbund geeignet testen, empfiehlt sich hierfür das Programm `tiotest` welches unter

`http://www.iki.fi/miku/`

zu finden ist.

Da die Hardware wohl in jedem Rechner unterschiedlich ist, wurde darauf verzichtet, in der Ergebnistabelle absolute Werte einzutragen. Ein besserer Vergleich ergibt sich, wenn man die Geschwindigkeit einer

RAID-Modus	Chunk-Size	Blockgröße ext2	Seq. Input	Seq. Output
Normal	%	4 kb	1 = Referenz	1 = Referenz
RAID-0	4 kb	4 kb	2,6 x Referenz	2,8 x Referenz
RAID-5	32 kb	4 kb	1,7 x Referenz	1,9 x Referenz
RAID-10	4 kb	4 kb	1,7 x Referenz	2,5 x Referenz

Festplatte alleine als den Faktor eins zugrundelegt und bei Verwendung derselben Festplatten die RAID-Performance als Vielfaches dieses Wertes angibt. Da diese Konfiguration drei identische SCSI-Festplatten vorsieht, wäre also maximal eine 3x1 fache Geschwindigkeit zu erwarten. Bedenken sollte man beim RAID-5 Testlauf auch, daß hier nur 2/3 der Kapazität direkt von Bonnie beschrieben werden. Andererseits soll dieser Test ja zeigen, wieviel Geschwindigkeitseinbußen oder -gewinn bei einer Datei derselben Größe zu erwarten sind.

Wie zu erwarten erreicht ein RAID-0 Verbund aus drei identischen Festplatten annähernd die dreifache Leistung. Auch die RAID-5 Ergebnisse zeigen die bei drei Festplatten erwartete doppelte Leistung. Die Geschwindigkeit der dritten Festplatte kann ja aufgrund der Paritätsinformation nicht gemessen werden. Die Leistung eines RAID-5 sollte also allgemein der aller verwendeten Platten minus eins für die Paritätsinformationplatte sein. Allerdings leidet RAID-5 an einer Art chronischer Schreibschwäche, welche durch das Berechnen und Ablegen der Paritätsinformationen zu erklären ist.

Um sich die Belastung des Prozessors und die benötigte Zeit zum Schreiben einer Testdatei anzusehen, kann man sich auch wieder des Programmes `dd` befleißigen. Der folgende Aufruf von `dd` würde eine 100 MB große Datei in das aktuelle Verzeichnis schreiben und die Ergebnisse anzeigen:

```
time dd if=/dev/zero of=./Testdatei bs=1024 count=102400
```

12 Tips und Tricks

Hier finden sowohl Tips und Tricks Erwähnung, die teilweise selbst getest wurden, als auch Besonderheiten, die nicht direkt durch die RAID-Kernelerweiterungen ermöglicht werden, sondern allgemeiner Natur sind oder zusätzlich integriert werden müssen. Einige der hier aufgelisteten Vorgänge sind mit allerhöchster Vorsicht zu genießen und mehr oder minder ausdrücklich *nicht* für eine Produktionsumgebung geeignet.

12.1 DRBD

Drbd versteht sich als ein Block Device, um eine Hochverfügbarkeitslösung unter Linux zu bieten. Im Prinzip ist es eine Art RAID-1 Verbund, der über ein Netzwerk läuft. Nähere Informationen hierzu gibt es unter:

```
http://www.complang.tuwien.ac.at/reisner/drbd/
```

12.2 Kernel 2.2.11 bis 2.2.13 und der RAID-Patch

Um Linux Software-RAID auch mit dem aktuelleren Kerneln zu verwenden, kann man einfach den RAID-Patch für den 2.2.10er Kernel auf den Sourcetree der 2.2.11er bis 2.2.13er Kernel anwenden. Zweimal kommt vom `patch` die Frage, ob eine bereits gepatchte Datei nochmal gepatcht werden soll. Diese Fragen sollten alle mit »no« beantwortet werden. Der Erfolg ist, daß der so gepatchte Kernel nach dem Kompilierlauf hervorragend mit allen RAID-Modi funktioniert. Wer sich den 2.2.10er RAID-Patch ersparen möchte, kann sich einen Kernel-Patch von Alan Cox für den 2.2.11er bis 2.2.13er Kernel aus dem Internet besorgen:

```
ftp.kernel.org:/pub/linux/kernel/people/alan/
```

Diese enthalten auch die jeweils aktuellen RAID-Treiber.

12.3 Kernel 2.2.14 bis 2.2.16 und der RAID-Patch

Auch in den aktuellsten 2.2.xer Linux Kerneln ist der aktuelle RAID-Patch noch nicht enthalten. Passende Patches hierfür findet man im Internet bei Ingo Molnar:

```
http://people.redhat.com/mingo/raid-patches/
```

12.4 Kernel der 2.4.xer Reihe und der RAID-Patch

Alle aktuellen Kernel der 2.4.xer Reihe beinhalten bereits die neue RAID-Unterstützung. Diese Kernel müssen nicht mehr gepatcht werden, lassen sich mit den aktivierten RAID-Optionen einwandfrei kompilieren und funktionieren problemlos.

12.5 SCSI-Festplatte zum Ausfallen bewegen

Für SCSI Devices aller Art - also nicht nur Festplatten - gibt es unter Linux die Möglichkeit, sie während des laufenden Betriebes quasi vom SCSI Bus abzuklemmen; dies allerdings ohne Hand an den Stromstecker oder Wechselrahmenschlüssel legen zu müssen. Möglich ist dies durch den Befehl:

```
echo "scsi remove-single-device c b t l" > /proc/scsi/scsi
```

Die Optionen stehen für:

```
c = die Nummer des SCSI-Kontrollers
b = die Nummer des Busses oder Kanals
t = die SCSI ID
l = die SCSI LUN
```

Möchte man das 5. Device des 1. SCSI Kontrollers rausschmeißen, müßte der Befehl so aussehen:

```
echo "scsi remove-single-device 0 0 5 0" > /proc/scsi/scsi
```

Umgekehrt funktioniert das Hinzufügen einer so verbannten Festplatte natürlich auch:

```
echo "scsi add-single-device c b t l" > /proc/scsi/scsi
```

12.6 Überwachen der RAID-Aktivitäten

Um einen Überblick über den Zustand des RAID-Systems zu bekommen, gibt es mehrere Möglichkeiten:

mdstat

Mit einem

```
cat /proc/mdstat
```

haben wir uns auch bisher immer einen Überblick über den aktuellen Zustand des RAID-Systems verschafft.

vmstat

Ist man an der fortlaufenden CPU und Speicher-Belastung sowie an der Festplatten I/O-Belastung interessiert, sollte man auf der Konsole einfach ein

```
vmstat 1
```

ausprobieren.

Mit grep oder perl mdstat abfragen

Wie üblich läßt sich unter Linux auch eine Abfrage von `/proc/mdstat` mit einem Skript realisieren. Hier könnte man z.B. die Folge »[UUUU]« nach einem Unterstrich regelmäßig per `cron` abfragen - der Unterstrich signalisiert ja eine defekte RAID-Partition - und läßt sich diese Information dann als E-Mail schicken. Läßt man sich allerdings schon eine E-Mail schicken, kann man sich ebenso gut auch eine SMS an sein Handy schicken lassen. Dieser Weg ist dann auch nicht mehr weit. Ein passendes `grep`-Kommando zum Abfragen von `/proc/mdstat` könnte dieses Aussehen haben:

```
grep '[\[U]_' /proc/mdstat
```

xosview

Die aktuelle Entwicklerversion dieses bekannten Überwachungsprogrammes enthält bereits eine Statussabfrage für RAID-1 und RAID-5 Verbunde. Für den ordnungsgemäßen Betrieb ist ein weiterer Kernel-Patch notwendig, der jedoch im tar-Archiv enthalten ist.

12.7 Verändern des read-ahead-Puffers

Um den read-ahead-Puffer jeglicher Major-Devices unter Linux einfach ändern zu können, gibt es ein nettes kleines Programm:

```
/* readahead -- set & get the read_ahead value for the specified device
*/

#include "stdio.h"
#include "stdlib.h"
#include "linux/fs.h"
#include "asm/fcntl.h"

void usage()
{
  printf( "usage:  readahead <device> [newvalue]\n" );

}/* usage() */
```

```
int main( int args, char **argv )
{
  int fd;
  int oldvalue;
  int newvalue;

  if ( args <= 1 ) {
    usage();
    return(1);
  }

  if ( args >= 3 && !isdigit(argv[2][0]) ) {
    printf( "readahead: invalid number.\n" );
    return(1);
  }

  fd = open( argv[1], O_RDONLY );
  if ( fd == -1 ) {
    printf( "readahead: unable to open device %s\n", argv[1] );
    return(1);
  }

  if ( ioctl(fd, BLKRAGET, &oldvalue) ) {
    printf( "readahead: unable to get read_ahead value from "
            "device %s\n", argv[1] );
    close (fd);
    return(1);
  }

  if ( args >= 3 ) {
    newvalue = atoi( argv[2] );
    if ( ioctl(fd, BLKRASET, newvalue) ) {
      printf( "readahead: unable to set %s's read_ahead to %d\n",
              argv[1], newvalue );
      close (fd);
      return(1);
    }
  }
  else {
    printf( "%d\n", oldvalue );
  }

  close (fd);
  return(0);

}/* main */
```

Damit kann man natürlich auch RAID-Devices tunen.

12.8 Bestehenden RAID-0 Verbund erweitern

Um einen RAID-0 Verbund zu vergrößern, zu verkleinern oder aus einer einzelnen Festplatte ein RAID-0 zu erstellen, gibt es von Jakob Oestergaard einen Patch für die RAID-Tools:

```
http://ostenfeld.dk/~jakob/raidreconf/raidreconf-0.0.2.patch.gz
```

Dieser Patch muß in den Source der RAID-Tools eingearbeitet werden und die RAID-Tools anschließend neu kompiliert und installiert werden. Das Erweitern eines RAID-0 Verbundes funktioniert dann durch zwei unterschiedliche `/etc/raidtab` Dateien, die miteinander verglichen werden und eine zusätzliche Partition innerhalb desselben Verbundes eingearbeitet wird. Nach dem stoppen des zu verändernden RAID-Verbundes, erfolgt der Aufruf durch:

```
raidreconf /etc/raidtab /etc/raidtab.neu /dev/md0
```

Hierbei muß in der `/etc/raidtab.neu` für den Verbund `/dev/md0` eine weitere Partition im Gegensatz zu `/etc/raidtab` eingetragen sein.

12.9 Bestehenden RAID-5 Verbund erweitern

Einen bereits initialisierten und laufenden RAID-5 Verbund kann man derzeit leider nicht mit weiteren Festplatten vergrößern. Die einzige Möglichkeit besteht darin, die Daten zu sichern, den RAID-5 Verbund neu aufzusetzen und die Daten anschließend zurück zu schreiben.