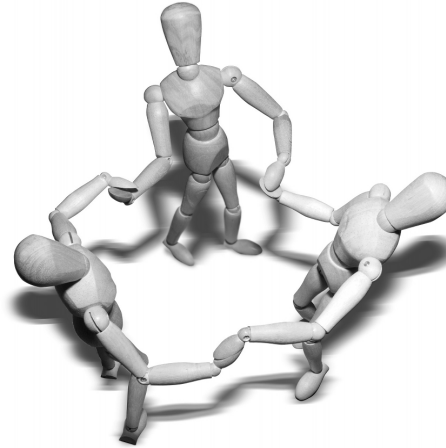


26. Verwaltung des Dateisystems



Dieses Kapitel gibt eine Einführung, wie das Linux-Dateisystem intern verwaltet wird und wie Sie manuell in diesen Prozess eingreifen können (mount und /etc/ fstab).

Die weiteren Abschnitte erklären den Umgang mit dem verschlüsselten Heimatverzeichnis (und dessen Tücken) und zeigen, wie Sie unter Windows auf Ihr Linux-Dateisystem zugreifen können.

26.1 Grundlagen

Auf vielen Rechnern sind neben Ubuntu auch andere Betriebssysteme installiert (z. B. Windows oder andere Linux-Distributionen). Jedes Betriebssystem beansprucht eine eigene Partition auf der Festplatte. Darüber hinaus kann es weitere Daten- und Swap-Partitionen geben (siehe auch Abschnitt 2.2).

/etc/fstab

Die leider recht unübersichtliche Datei /etc/fstab steuert, welche Partitionen unter Ubuntu immer zugänglich sind. Zeilen, die mit dem Zeichen # beginnen, gelten als Kommentare und werden nicht ausgewertet. Änderungen an /etc/fstab werden erst beim nächsten Systemstart bzw. bei der manuellen Ausführung von mount-Kommandos wirksam.

Standardmäßig sieht diese Datei so ähnlich wie im folgenden Beispiel aus. Die UUID-Nummern sind hier aus Platzgründen gekürzt. Es handelt sich dabei je nach Dateisystem um hexadezimale Zahlen mit bis zu 32 Stellen.

```
# Datei /etc/fstab
# Dateisystem Verzeichnis Typ Optionen Dump Pass
# / was on /dev/sdb2 during installation
UUID=08c3... / ext4 errors=remount-ro 0 1
# swap was on /dev/sdb3 durin installation
UUID=278e... none swap sw 0 0
/dev/scd0 /media/cdrom0 udf,iso9660 user,noauto,exec,utf8 0 0
proc /proc proc defaults 0 0
```

Kurz einige Worte zur Bedeutung der sechs Spalten in `/etc/fstab`:

- » Die erste Spalte gibt an, wo sich die Daten physikalisch befinden. Die Partitionen werden durch UUID-Nummern angesprochen. Das hat den Vorteil, dass Linux auch nach Hardware-Änderungen (z. B. nach dem Einbau einer neuen Festplatte) alle Partitionen sicher wiederfindet. Die UUID wird beim Formatieren eines Dateisystems zufällig festgelegt.

Die Kommentarzeile vor der UUID-Angabe gibt an, welchen Device-Namen die Partition während der Installation hatte. In früheren Ubuntu-Versionen wurden Partitionen nur über den Device-Namen angesprochen. Das ist auch weiterhin möglich. Solange Sie an der Verkabelung Ihrer Festplatten nichts ändern, können Sie

```
# /dev/sdb2
UUID=08c3... / ext4 errors=remount-ro 0 1
```

durch

```
/dev/sdb3 / ext4 errors=remount-ro 0 1
```

ersetzen. Persönlich ziehe ich diese Schreibweise aufgrund der besseren Übersichtlichkeit vor. (Hardware-Änderungen sind bei mir selten, und wenn sie doch vorkommen, bin ich in der Lage, `/etc/fstab` selbst zu modifizieren.)

- » Die zweite Spalte gibt an, wo die Daten in das Linux-Dateisystem eingebunden sind. Der Zugriff auf die Dateien einer CD erfolgt also über das Verzeichnis `/media/cdrom0`.
- » Die dritte Spalte gibt den Typ oder die möglichen Typen des Dateisystems an. Die folgenden Dateisysteme kommen häufig vor:

<code>ext4</code>	Linux-Standarddateisystem
<code>ext3</code>	Vorgänger von <code>ext4</code> , nach wie vor weit verbreitet
<code>vfat</code>	altes Windows-Dateisystem (Windows 9x, ME)
<code>ntfs</code>	neues Windows-Dateisystem (Windows NT, 2000, XP, Vista, 7)
<code>iso9660</code>	Dateisystem für CDs und DVDs
<code>udf</code>	neues Dateisystem für (Video-)DVDs und optische Datenträger
<code>nfs</code>	Unix/Linux-Netzwerkdateisystem
<code>swap</code>	Swap-Partition (entspricht einer Windows-Auslagerungsdatei)

- » Die vierte Spalte enthält Optionen. Die Optionen müssen durch Kommas (keine Leerzeichen!) voneinander getrennt werden. `errors=remount-ro` bedeutet, dass das Dateisystem *read-only* eingebunden wird, wenn es Konsistenzfehler enthält. Wenn keine Optionen angegeben werden sollen, enthält die Spalte das Schlüsselwort `defaults`.
- » Die fünfte Spalte (*dump*) wird bei den meisten Linux-Distributionen ignoriert und enthält immer 0.

» Die sechste Spalte (*pass*) gibt an, ob und in welcher Reihenfolge das Dateisystem beim Systemstart auf Fehler überprüft werden soll.

- 0 keine Prüfung
- 1 zuerst prüfen
- 2 später prüfen

Im Klartext ...

Ubuntu wurde also in die Festplattenpartition mit der UUID 08c3... mit dem Device-Namen `/dev/sdb2` installiert. Die Systempartition ist für die Funktion von Ubuntu unbedingt erforderlich. Hierin befinden sich alle grundlegenden Systemdateien.

Die Partition `/dev/sdb3` enthält eine Swap-Partition, die dieselbe Funktion wie eine Windows-Auslagerungsdatei hat. Die beiden folgenden Zeilen ermöglichen den Zugriff auf CDs/DVDs bzw. auf Disketten. Die `proc`-Zeile betrifft ein virtuelles Dateisystem, das Auskunft über laufende Prozesse gibt.

Falls es auf Ihrem Rechner weitere Partitionen gibt, können Sie diese über das Gnome-Menü `ORTE` ebenfalls in das Dateisystem einbinden. Anders als in früheren Ubuntu-Versionen gibt es hierfür aber keine `fstab`-Einträge.

Zugriff auf andere Festplattenpartitionen (`mount`)

Über das nicht ganz korrekt bezeichnete Gnome-Menü `ORTE|WECHSELMEDIEN` können Sie alle Partitionen der lokalen Festplatten, die momentan nicht genutzt sind, in das Dateisystem integrieren (»einbinden«). Die Partitionen haben im Gnome-Menü keine Namen, sondern können nur über ihre Größe identifiziert werden.

Hinter den Kulissen der komfortablen Gnome-Benutzeroberfläche steht das Kommando `mount`. Erfahrene Linux-Anwender können `mount` selbst in einem Konsolenfenster ausführen und auf diese Weise eine Partition oder einen externen Datenträger in das Dateisystem einbinden. `mount` erfordert `root`-Rechte (also `sudo`). Die Syntax von `mount` sieht vereinfacht so aus:

```
mount [optionen] device verzeichnis
```

In den Optionen wird unter anderem der Dateisystemtyp angegeben (`-t xxx`). Der Device-Name bezeichnet die Partition bzw. das Laufwerk. Als Verzeichnis kann ein beliebiges Verzeichnis des aktuellen Dateisystems angegeben werden. (Das Verzeichnis muss bereits existieren. Erzeugen Sie es gegebenenfalls mit `mkdir`!)

Das folgende Beispiel zeigt den Zugriff auf die Daten einer Windows-Partition über das Verzeichnis `/media/windows`:

```
user$ mkdir /media/windows
user$ mount -t ntfs /dev/sda1 /media/windows
```

Um ein Dateisystem aus dem Verzeichnisbaum zu lösen, führen Sie `umount` aus:

```
user$ umount /media/windows
```

Das Gnome Virtual File System (GVFS)

Auch in Gnome können Sie Dateisysteme in das Dateisystem einbinden. Soweit es sich dabei um Partitionen der lokalen Festplatte oder um externe Datenträger handelt (z. B. um USB-Sticks), bindet `ORTE[XXX DATEISYSTEM` das Dateisystem wie oben beschrieben mit `mount` in ein `/media`-Verzeichnis ein – natürlich, ohne dass Sie etwas von den technischen Details bemerken. (Hinter den Kulissen bekommt Gnome vom PolicyKit die erforderlichen Rechte, um das `mount`-Kommando auszuführen.)

Für Netzwerkverzeichnisse verwendet Gnome aber eine andere Vorgehensweise: Wenn Sie sich mit Nautilus durch ein Windows-Netzwerkverzeichnis klicken oder wenn Sie `ORTE|VERBINDUNG ZU SERVER` ausführen, wird das Netzwerkverzeichnis über das *Gnome Virtual File System* im lokalen Verzeichnis `~/.gvfs` in den Verzeichnisbaum eingebunden. Im Idealfall funktioniert das so transparent, dass Sie von diesen Gnome-Internen nie etwas bemerken. Nautilus und die Gnome-Dateiauswahldialoge zeigen die Verzeichnisse einfach an. Der Zugriff auf Netzwerkverzeichnisse funktioniert quasi von Zauberhand.

Die Praxis sieht leider nicht ganz so rosig aus: GVFS ist seit Jahren durch eine schier endlose Serie von Pannen und Fehlern gezeichnet. Mal lassen sich keine FTP-Verzeichnisse nutzen, dann funktioniert die Speicherung der Passwörter für Windows-Netzwerkverzeichnisse nicht, oder Gnome vergisst eingebundene Verzeichnisse beim Logout. Kurz und gut: Immer dann, wenn die Gnome-Automatismen versagen, ist es gut, wenn Sie selbst in der Lage sind, `mount` auszuführen. Das ist nicht so bequem, aber dafür funktioniert es! (Informationen speziell zum Einbinden von Windows-Netzwerkverzeichnissen folgen in Abschnitt 27.3.)

26.2 Partitionierung der Festplatte ändern

Während der Installation haben Sie in der Regel eine oder mehrere Linux-Partitionen erzeugt. Wenn Sie die Partitionierung später ändern möchten, können Sie dazu die Benutzeroberflächen `Palimpsest` oder `gparted` bzw. die Kommandos `parted` und `fdisk` einsetzen.

Das Verändern vorhandener Partitonen führt in der Regel dazu, dass die Daten auf dieser Partition verloren gehen! Erstellen Sie also unbedingt ein Backup! Nur das Hinzufügen neuer Partitionen ist weitgehend gefahrlos – aber selbst da kann ein Backup nicht schaden. Die größte Flexibilität haben Sie, wenn Sie sich bereits bei der Installation im Textmodus für LVM entschieden haben. Tipps zur LVM-Administration folgen in Abschnitt 26.6.

Grundsätzlich ist es nicht möglich, aktive Partitionen zu verändern. Deswegen ist es oft erforderlich, den Rechner für die Partitionierungsarbeiten von einer Live-CD oder -DVD zu starten (z. B. von den beiliegenden DVDs).

Festplatten mit 4-kByte-Sektoren

Neue Festplatten verwenden statt der jahrzehntelang üblichen 512-Byte-Sektoren längere Sektoren von 4096 Byte (4 kByte). Das hat viele Vorteile, unter anderen eine höhere Geschwindigkeit und eine höhere Festplattenkapazität. Aus Kompatibilitätsgründen melden aber auch Festplatten mit 4-kByte-Sektoren eine 512-Byte-Sektorgröße an das Betriebssystem.

Um Festplatten mit 4-kByte-Sektoren effizient zu nutzen, müssen Partitionen so eingerichtet werden, dass die Startposition jeder Partition ein Vielfaches von 4 kByte beträgt. Ist das nicht der Fall und will das Dateisystem einen 4-kByte-Bereich verändern, muss die Festplatte *zwei* 4-kByte-Sektoren lesen, modifizieren und schreiben. Das würde Schreibvorgänge massiv bremsen.

In der Vergangenheit war es üblich, dass die erste Partition mit dem Sektor 63 begann (also an der Position $63 * 512$ Byte). Wenn Sie ältere Windows-Versionen (Windows XP und früher) einsetzen, ist das noch immer erforderlich. Für die Verwendung von Festplatten mit 4-kByte-Sektoren ist das aber nicht optimal, weswegen manche Festplattenhersteller spezielle Low-Level-Formatier-Tools anbieten. Damit können Sie die Festplatte so neu formatieren, dass der 63. Sektor intern auf einer 4-kByte-Grenze liegt. Damit erzielen Sie dann unter Windows XP eine optimale Geschwindigkeit, nicht aber mit neueren Betriebssystemen. Ich gehe hier davon aus, dass die Festplatte *nicht* Windows-XP-spezifisch neu formatiert wurde!

Neuere Betriebssysteme nehmen bereits Rücksicht auf die neue Sektorgröße. Beispielsweise richten Windows 7, Ubuntu 10.04 und Fedora 12 die Partitions Grenzen bei Vielfachen von 1 MByte aus. Das ist allerdings inkompatibel zu Windows XP und kann zu Problemen führen, wenn Sie auf der Festplatte parallel auch Windows XP installieren möchten!

Wenn Sie selbst Partitionen einrichten und Programme verwenden, die mit 512-Byte-Sektoren rechnen (z. B. fdisk, wenn Sie keine zusätzlichen Optionen angeben), müssen Sie darauf achten, dass die Partitions Grenzen ein Vielfaches von 8 Sektoren betragen.

Technische Hintergründe zur optimalen Nutzung von Festplatten mit 4-kByte-Festplatten können Sie hier nachlesen:

<http://lwn.net/Articles/377895/>

<http://www.heise.de/open/artikel/Kernel-Log-Linux-und-Festplatten-mit-4-KByte-Sektoren-938237.html>

Palimpsest

Wenn Sie sich ohne unübersichtliche Kommandowerkzeuge einen Überblick über alle Festplattenpartitionen verschaffen möchten oder einen Datenträger (z. B. einen USB-Stick) komfortabel formatieren möchten, starten Sie mit SYSTEM|SYSTEMVERWALTUNG|LAUFWERKSVERWALTUNG das Programm Palimpsest. Es listet alle Festplatten sowie alle darin enthaltenen Partitionen auf. Das funktioniert auch für LVM-Partitionen (also *logical volumes*), die unter den Peripheriegeräten aufgelistet werden. Mit DATEI|ERSTELLEN|RAID-ANORDNUNG können Sie neue Software-RAID-Verbunde einrichten. Via SSH können Sie mit Palimpsest sogar die Partitionierung eines anderen Rechners im Netzwerk verändern. (Diese Funktion hat bei meinen Tests allerdings versagt.)

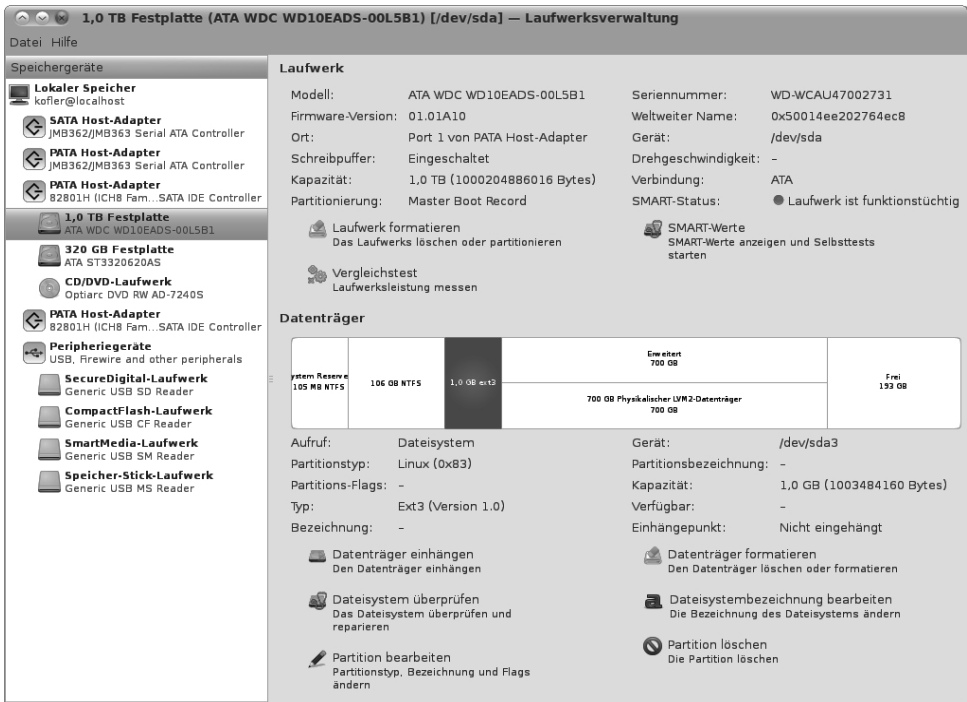


Abbildung 26.1: Palimpsest

Nachdem Sie eine Partition ausgewählt haben, können Sie das darin enthaltene Dateisystem überprüfen, löschen (d. h. neu formatieren) oder in den Verzeichnisbaum einhängen. Soweit eine Festplatte noch freien (nicht von Partitionen genutzten) Platz hat, können Sie dort eine neue Partition einrichten und formatieren. Sie können das Programm auch dazu verwenden, um auf einem USB-Stick ein Linux- oder Windows-Dateisystem einzurichten.

Palimpsest kann auch dazu verwendet werden, den SMART-Zustand einer Festplatte zu überprüfen. Mehr Informationen zu SMART folgen in Abschnitt 26.8.

GParted

Ähnliche Funktionen wie Palimpsest bietet das Programm GParted. Seine Besonderheit besteht darin, dass es bei manchen Operationen den Inhalt eines Dateisystems erhält. GParted muss vor der ersten Verwendung installiert werden (`apt-get install gparted`). Anschließend starten Sie das Programm mit `SYSTEM|SYSTEMVERWALTUNG|PARTITIONSEEDITOR` und können dann – sofern noch Platz auf der Festplatte ist – neue Partitionen erzeugen. Die Größe vorhandener Partitionen ist dagegen im Regelfall nicht veränderlich.

gparted ist leider nicht in der Lage, LVM- oder RAID-Partitionen einzurichten. Entsprechende Flags werden zwar angezeigt, können aber nicht verändert werden.

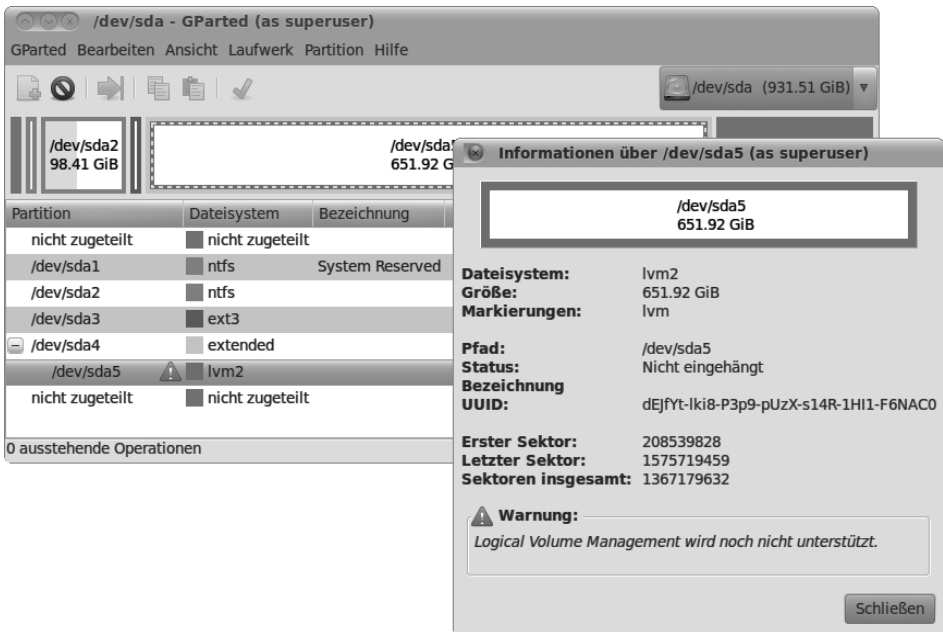


Abbildung 26.2: Festplattenpartitionierung mit gparted ändern

fdisk

fdisk zählt zu den ältesten Linux-Partitionierprogrammen. Die kommandoorientierte Bedienung ist zwar wenig intuitiv, dafür ist das Programm ausgereift und vor allem bei langjährigen Linux-Anwendern beliebt.

fdisk -l liefert eine Liste aller Partitionen auf allen Festplatten. Wenn Sie eine Festplatte bearbeiten möchten, müssen Sie deren Device-Namen beim Start von fdisk explizit angeben (z. B. /dev/sdc für die dritte SATA/SCSI-Festplatte). Zur Bearbeitung neuer Festplatten mit 4-kByte-Sektoren sollten Sie beim Start außerdem die Optionen -c und -u angeben:

- » Mit -c nimmt fdisk keine Rücksicht auf die Kompatibilität zu DOS sowie zu älteren Windows-Versionen (bis einschließlich Windows XP). Das ermöglicht eine optimale Ausrichtung der Partitions Grenzen.
- » Mit -u rechnet fdisk nicht in Zylindern, sondern in Sektoren. (Beide Maßeinheiten sind eigentlich überholt. Sie ergeben sich aus dem physikalischen Aufbau alter Festplatten. Als Sektor gelten 512 Byte, als Zylinder $63 * 255 * 512 \text{ Byte} = 8.225.280 \text{ Byte} = 7,8 \text{ MByte}$. Beachten Sie, dass auch neue Festplatten mit 4-kByte-Sektoren an das Betriebssystem eine Sektorgröße von 512 Byte melden. Deswegen rechnet fdisk auch bei neuen Festplatten mit einer Sektorgröße von 512 Byte!)

```
root# fdisk -c -u /dev/sda
```

Nach dem Start liefert **[M]** (*menu*) eine kurze Übersicht der zur Verfügung stehenden Kommandos. **[P]** (*print*) zeigt eine Liste der Partitionen an, die zurzeit auf der ausgewählten Festplatte vorhanden sind.

Mit **[N]** (*new*) richten Sie neue Festplattenpartitionen ein. Wenn mehr als vier Partitionen verwaltet werden sollen, muss eine der vier primären Partitionen als erweitert (*extended*) deklariert werden. Im Bereich der erweiterten Partition dürfen dann bis zu elf logische Partitionen eingerichtet werden. Falls beim Einrichten einer neuen Festplattenpartition unterschiedliche Typen in Frage kommen (primär, erweitert oder logisch), antwortet fdisk mit einer zusätzlichen Rückfrage bezüglich des Partitionstyps.

Nachdem geklärt ist, welchen Typ die neue Partition haben soll, fragt das Programm, an welcher Stelle die Partition beginnen soll (normalerweise beim ersten freien Zylinder) und wie groß sie sein soll (Endzylinder). Die Größenangabe kann auch bequemer in der Form + n \$ M oder + n \$ G erfolgen (also +50G für eine 50 GByte große Partition).

Nach der Definition einer neuen Partition kann die gesamte Partitionstabelle mit **[P]** (*print*) angezeigt werden. Anschließend können weitere Partitionen definiert und bereits definierte Partitionen wieder gelöscht werden etc.

fdisk erzeugt neue Partitionen immer vom Typ *Linux native* (ID-Nummer 83). Wenn Sie einen anderen Typ benötigen, müssen Sie die ID-Nummer der neu eingerichteten Partition mit **[T]** (*type*) ändern. Übliche ID-Nummern in hexadezimaler Schreibweise sind in Tabelle 26.1 zusammengefasst. Eine Liste aller verfügbaren ID-Nummern erhalten Sie mit **[L]**.

Die mit `fdisk` eingerichteten Partitionen sind noch leer, d. h., `fdisk` installiert kein Dateisystem! Die Kommandos zur Einrichtung eines Dateisystems hängen vom gewünschten Dateisystem ab – etwa `mkfs.ext4` für ein `ext4`-Dateisystem.

ID-NUMMER	BEDEUTUNG
5	erweiterte Partition
7	NTFS-Partition (aktuelle Windows-Versionen)
6	FAT 16-Bit (DOS, alte Windows-Versionen)
b	VFAT 32-Bit (Windows 9x)
c	VFAT 32-Bit mit LBA (Windows 9x, Memory Sticks)
82	Linux-Swap-Partition
83	Linux-Dateisystem (für alle Linux-Dateisysteme: <code>ext</code> , <code>xf</code> s etc.)
8e	Linux-LVM-Partition
fd	Linux-RAID-Partition

Tabelle 26.1: Partitions-ID-Nummern (hexadezimal) für `fdisk`

`fdisk` kann grundsätzlich die Größe einer existierenden Partition nicht verändern. Die einzige Ausnahme liegt dann vor, wenn die zu ändernde Partition die letzte Partition auf der Festplatte ist (bzw. die letzte logische Partition innerhalb einer erweiterten Partition) und dahinter noch Platz frei ist. In diesem Fall können Sie diese Partition löschen und anschließend vergrößert neu anlegen. (`fdisk` verändert nur die Partitionstabelle, lässt die eigentlichen Daten auf der Festplatte aber unberührt. Das bedeutet, dass das Dateisystem in der vergrößerten Partition nicht mitwächst. Damit ist nun ein Teil der Partition ungenutzt. Wie Sie ein `ext`-Dateisystem vergrößern, verrät Abschnitt 26.3.)

`fdisk` führt sämtliche Änderungen erst mit dem Kommando `[W]` (*write*) aus. Vorher können Sie mit `[V]` (*verify*) überprüfen, ob alle internen Informationen mit der Platte übereinstimmen. Das ist eine zusätzliche Sicherheitskontrolle. Normalerweise besteht die Reaktion auf `[V]` nur darin, dass die Anzahl der von keiner primären oder logischen Partition erfassten (also noch ungenutzten) Sektoren zu je 512 Byte angezeigt wird. Wenn Sie sich unsicher sind, können Sie `fdisk` jederzeit mit `[Q]` (*quit*) oder auch mit `[Strg]+[C]` verlassen – Ihre Festplatte bleibt dann so, wie sie ist.

fdisk-Beispiel

Das folgende Beispiel zeigt die Erstellung einer neuen Partition auf einer Festplatte mit 4-kByte-Sektoren. Das Kommando `[P]` gibt Auskunft über den aktuellen Zustand der Festplatte. Die Festplatte ist $255 * 63 * 182401 * 512$ Byte = ca. 1,36 TByte groß. (Wenn man wie die Festplattenhersteller rechnet, also mit 1 TByte = 10^{12} Byte, dann ergeben sich 1,5 TByte. Ich rechne hier aber wie `fdisk` mit 2er-Potenzen, also mit 1 kByte = 2^{10} Byte, 1 TByte = 2^{40} Byte.)

Anfänglich sind zwei primäre Partitionen vorhanden. Die erste Partition beginnt bei Sektor 2048 bzw. an der Byte-Position $2048 * 512 \text{ Byte} = 1 \text{ MByte}$. Diese Partition ist 48.827.392 Blöcke groß (ein »Block« ist 1024 Byte), also $48.827.392 * 1024 \text{ Byte} = \text{ca. } 46,6 \text{ GByte}$. (Während der Installation habe ich angegeben, dass die Partition 50 GByte groß sein soll. Das Ubuntu-Installationsprogramm rechnet aber ebenfalls dezimal.) Die zweite Partition beginnt bei Sektor 97.656.832, exakt 47.684 MByte nach dem Beginn der Festplatte. Die Partitionen sind also auf 1-MByte-Grenzen ausgerichtet. Diese Partition ist $1.952.768 * 1024 \text{ Byte} = \text{ca. } 1,9 \text{ GByte}$ groß.

```
root# fdisk -c -u /dev/sda
```

```
Befehl (m für Hilfe): p
```

```
Platte /dev/sda: 1500.3 GByte, 1500301910016 Byte
```

```
255 Köpfe, 63 Sektoren/Spur, 182401 Zylinder, zusammen 2930277168 Sektoren
```

```
Einheiten = Sektoren von 1 × 512 = 512 Bytes
```

```
Sector size (logical/physical): 512 bytes / 512 bytes
```

```
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

```
Disk identifier: 0x0004b057
```

Gerät	boot.	Anfang	Ende	Blöcke	Id	System
/dev/sda1	*	2048	97656831	48827392	83	Linux
/dev/sda2		97656832	101562367	1952768	82	Linux Swap / Solaris

[N] erzeugt nun eine neue, primäre Partition. Als Startzylinder wird der erste freie Zylinder verwendet. Anstatt den Endzylinder anzugeben, bewirkt +30G, dass die Partition 30 GByte groß werden soll. fdisk folgt dieser Anweisungen exakt ($31.457.280 * 1024 \text{ Byte} = 30 \text{ GByte}$).

```
Befehl (m für Hilfe): n
```

```
Befehl Aktion
```

```
  e      Erweiterte
```

```
  p      Primäre Partition (1-4)
```

```
p
```

```
Partitionsnummer (1-4): 3
```

```
Erster Sektor (101562368-2930277167, Vorgabe: 101562368): <return>
```

```
Benutze den Standardwert 101562368
```

```
Last Sektor, +Sektoren or +size{K,M,G} (101562368-2930277167,
```

```
Vorgabe: 2930277167): +30G
```

```
Befehl (m für Hilfe): p
```

Gerät	boot.	Anfang	Ende	Blöcke	Id	System
/dev/sda1	*	2048	97656831	48827392	83	Linux
/dev/sda2		97656832	101562367	1952768	82	Linux Swap / Solaris
/dev/sda3		101562368	164476927	31457280	83	Linux

Mit **[W]** wird die geänderte Partitionstabelle gespeichert. Da einige andere Partitionen der Festplatte zurzeit genutzt werden, gelingt es nicht, die neue Tabelle korrekt einzulesen. Der Rechner muss also neu gestartet werden, bevor die neue Partition verwendet werden darf!

Command (m for help): **w**

Die Partitionstabelle wurde verändert!

Rufe `ioctl()` um Partitionstabelle neu einzulesen.

WARNING: Re-reading the partition table failed with error 16: Device or resource busy. The kernel still uses the old table. The new table will be used at the next reboot or after you run `partprobe(8)` or `kpartx(8)`

Synchronisiere Platten.

parted

Die Bedienung von `parted` ist noch umständlicher als die von `fdisk`. Ein weiterer Nachteil besteht darin, dass jede Änderung sofort ausgeführt wird, nicht erst explizit durch eine Aufforderung zum Speichern. Aus diesen Gründen bevorzuge ich persönlich `fdisk` oder setze ein grafisches Partitionierungswerkzeug ein.

Beim Start von `parted` geben Sie das Festplatten-Device an. `help` führt zur Anzeige der zur Auswahl stehenden Kommandos. `help` kommando liefert einen knappen Hilfetext zu den einzelnen Kommandos. `print` zeigt die Partitionstabelle an – hier für eine 1,5-TByte-Festplatte, die bereits drei Partitionen enthält.

```
root# parted /dev/sda
```

```
(parted) print
```

```
Modell: ATA WDC WD15EARS-00Z (scsi)
```

```
Festplatte /dev/sda: 1500GB
```

```
Sektorgröße (logisch/physisch): 512B/512B
```

```
Partitionstabelle: msdos
```

Anzahl	Beginn	Ende	Größe	Typ	Dateisystem	Flags
1	1049kB	50,0GB	50,0GB	primary	ext4	boot
2	50,0GB	52,0GB	2000MB	primary	linux-swap(v1)	
3	52,0GB	84,2GB	32,2GB	primary		

`parted` rechnet standardmäßig dezimal (also 1 GByte = 10^9 Byte). Sie können aber mit `unit` KiB, MiB, GiB oder TiB alle Angaben in binäre kByte, MByte, GByte oder TByte umrechnen:

```
(parted) unit MiB
```

```
(parted) print
```

```
...
```

```
Festplatte /dev/sda: 1397GiB
```

```
...
```

Anzahl	Beginn	Ende	Größe	Typ	Dateisystem	Flags
1	1,00MiB	47684MiB	47683MiB	primary	ext4	boot
2	47684MiB	49591MiB	1907MiB	primary	linux-swap(v1)	
3	49591MiB	80311MiB	30720MiB	primary		

Mit den Kommandos `mkpart` bzw. `rm` erzeugen bzw. löschen Sie Partitionen. Bei `mkpart` müssen Sie die gewünschte Start- und Endposition der neuen Partition angeben. (Es ist leider nicht möglich, die gewünschte Partitionsgröße anzugeben, damit sich `parted` die Start- und

Endposition selbst ausrechnet.) Falls Sie eine logische Partition erzeugen, müssen Sie vorher eine erweiterte Partition anlegen oder entsprechend vergrößern. Kommandos zur Veränderung von Partitionen erwarten als Parameter immer die sogenannte *minor*-Nummer des Devices – also z. B. 7 für `/dev/sda7`.

Wenn Sie die neue Partition als Swap-Partition oder als Teil eines LVM- oder RAID-Systems nutzen möchten, müssen Sie den Partitionstyp entsprechend einstellen. Das erforderliche Kommando lautet `set partitionsnummer attributname on`. Mögliche Attribute sind unter anderem `boot`, `swap`, `lvm` und `raid`.

Die folgenden Kommandos richten zuerst eine erweiterte und dann darin eine logische Partition mit einem ext4-Dateisystem ein. Die erweiterte Partition soll an Partition 3 anschließen und bis zum Ende der Festplatte reichen. (Negative Positionsangaben gelten relativ vom Ende der Festplatte.) Die logische Partition soll ca. 40 GByte groß werden.

```
(parted) mkpart extended 80311MiB -0
```

```
Warnung: WARNING: the kernel failed to re-read the partition table on
/dev/sda (Device or resource busy). As a result, it may not reflect all of
your changes until after reboot.
```

```
(parted) mkpart logical 80312MiB 120000MiB
```

```
(parted) print
```

Anzahl	Beginn	Ende	Größe	Typ	Dateisystem	Flags
1	1,00MiB	47684MiB	47683MiB	primary	ext4	boot
2	47684MiB	49591MiB	1907MiB	primary	linux-swap(v1)	
3	49591MiB	80311MiB	30720MiB	primary		
4	80311MiB	1430799MiB	1350488MiB	extended		lba
5	80312MiB	120000MiB	39688MiB	logical		

```
(parted) quit
```

26.3 Linux-Dateisysteme (ext2, ext3, ext4)

Seit vielen Jahren sind ext2, ext3 und nun ext4 die dominierenden Dateisysteme für Linux. Sie vereinen hohe Effizienz mit ebenso hoher Zuverlässigkeit. Dieser Abschnitt beschreibt zuerst einige Grundlagen des ext-Dateisystems und gibt dann Tipps und Hinweise zur Administration von ext-Dateisystemen.

ext-Versionen

Standardmäßig kommt bei Ubuntu-Neuinstallationen ext4 zum Einsatz; die älteren Dateisystemversionen ext2 und ext3 werden aber weiterhin unterstützt. Kurz ein historischer Rückblick:

- » **ext**, also die erste Version des ext-Dateisystems, wurde nur kurz in der Anfangsphase von Linux eingesetzt (1992). Die maximale Dateisystemgröße betrug 2 GByte.
- » **ext2** war von 1993 bis ca. 2001 das dominierende Linux-Dateisystem. Die maximale Dateisystemgröße wuchs in dieser Version auf 8 TByte.
- » Die wichtigsten Neuerungen in **ext3** waren die Journaling-Funktionen und die Unterstützung von *Access Control Lists* (ACLs). Vorhandene ext2-Dateisysteme mussten nicht neu formatiert werden, sondern konnten mit minimalem Aufwand auf ext3 umgestellt werden. Sofern das Dateisystem ordnungsgemäß mit `umount` gelöst wird, kann es anschließend sogar wieder als ext2-Dateisystem genutzt werden.
- » Seit Ende 2008 steht die momentan aktuelle Version **ext4** zur Verfügung. Die maximale Dateisystemgröße steigt damit auf ein Exabyte (1.048.576 Terabyte), und der Zeitpunkt von Dateiänderungen wird genauer als bisher protokolliert. Sogenannte *extents* ermöglichen es, aneinanderliegende Blöcke des Dateisystems als Gruppen anzusprechen, was den Aufwand zur Verwaltung großer Dateien deutlich senkt.

Außerdem wurden eine Menge Geschwindigkeitsoptimierungen durchgeführt: Sowohl das Löschen großer Dateien als auch die Dateisystemüberprüfung wird nun um ein Vielfaches schneller als bei ext3 durchgeführt. Abermals wurde auf Kompatibilität geachtet: Eine Migration von ext3 zu ext4 ist problemlos möglich. Beachten Sie aber, dass es nach einer Umstellung von ext3 auf ext4 kein Zurück mehr gibt!

ext4 ist seit Herbst 2009 das Standarddateisystem der meisten gängigen Distributionen (Fedora, Ubuntu, openSUSE). Obwohl ich Dateisystemen gegenüber konservativ bin (Sicherheit ist mir wichtiger als ein paar Prozent Effizienzsteigerung), setze ich ext4 mittlerweile selbst produktiv ein. Die weite Verbreitung von ext4 lässt hoffen, dass die ärgsten Kinderkrankheiten, mit denen jedes neue Dateisystem kämpft, bereits ausgestanden sind.

Journaling

In seiner einfachsten Form bedeutet Journaling, dass der Beginn und das Ende jeder Dateioperation in einer speziellen Datei mitprotokolliert werden. Dank des Protokolls kann später geprüft werden, ob eine bestimmte Dateioperation vollständig ausgeführt wurde. Wenn das nicht der Fall ist, kann die Operation widerrufen werden. (In der Datenbankwelt spricht man hier von Transaktionen.) Bei fortgeschrittenen Journaling-Systemen besteht auch die Möglichkeit, die eigentlichen Änderungen an den Dateien im Journal zu protokollieren. Das verlangsamt den gewöhnlichen Betrieb, gibt aber mehr Möglichkeiten zur späteren Rekonstruktion.

Wenn nun eine Dateioperation nicht vollständig abgeschlossen werden kann, geht dies aus dem Protokoll hervor. Bei einfachem Journaling sind die Änderungen zwar verloren (versprechen Sie sich also keine Wunder von der Journaling-Funktion!), der bisherige Zustand der Datei steht aber zumeist noch zur Verfügung.

Der große Vorteil der Journaling-Funktionen besteht darin, dass das Dateisystem beim nächsten Rechnerstart sehr rasch wieder in einen konsistenten Zustand gebracht und beinahe sofort wieder genutzt werden kann. Das ist ein großer Unterschied im Vergleich zu früher, wo nach einem Absturz oder Stromausfall das gesamte Dateisystem systematisch nach eventuellen Fehlern durchsucht werden musste. Das dauerte mehrere Minuten, bei sehr großen Festplatten eventuell sogar Stunden!

Journaling gibt allerdings bei einem Stromausfall keine Garantie für ein konsistentes Dateisystem! Das Problem liegt bei den Festplatten: Diese verwenden aus Effizienzgründen beim Schreiben einen internen Zwischenspeicher. Daher kann es passieren, dass das Dateisystem von der Festplatte die Bestätigung erhält, dass sie die Daten empfangen und gesichert hat. Tatsächlich kann es danach aber noch Sekunden dauern, bis die Daten vom Zwischenspeicher physikalisch auf die Festplatte geschrieben werden. Tritt in dieser Zeitspanne ein Stromausfall auf, gehen die Daten im Zwischenspeicher verloren. (Bei manchen Festplatten lässt sich dieser Cache deaktivieren. Dadurch verringert sich die Geschwindigkeit von Schreiboperationen aber derart, dass in der Praxis zumeist darauf verzichtet wird.)

Unabhängig von Schreib-Cache ist das Verhalten einer Festplatte während eines plötzlichen Stromausfalls undefiniert. Es kann also auch passieren, dass die Festplatte statt Ihrer Daten Zufallsbits schreibt, bevor der Schreibkopf in Sicherheit gebracht wird. Anders formuliert: Journaling ist eine feine Sache, schließt einen Datenverlust bei einem Stromausfall aber nicht aus. Wenn Ihnen Ihre Daten etwas wert sind, investieren Sie 100 EUR für eine kleine UPS-Anlage (*Uninterruptable Power Supply*), die sicherstellt, dass Sie Ihre Rechner auch bei einem Stromausfall geordnet herunterfahren können. (Bei einem Notebook haben Sie dieses Problem ohnedies nicht.)

Das ext-Dateisystem kennt drei verschiedene Verfahren, wie das Journaling durchgeführt wird:

- » **data=ordered:** Bei diesem Modus werden im Journal nur Metadaten gespeichert, also Informationen über Dateien, aber keine Inhalte der Dateien. Im Journal werden Dateien erst dann als korrekt (*committed*) gekennzeichnet, wenn sie vollständig auf der Festplatte gespeichert worden sind. Nach einem Crash kann das Dateisystem sehr rasch wieder in einen konsistenten Zustand gebracht werden, weil alle unvollständig gespeicherten Dateien anhand des Journals sofort erkannt werden. Es ist aber nicht möglich, unvollständig gespeicherte Dateien wiederherzustellen.

Im Modus `data=ordered` wird das Journal alle fünf Sekunden mit der Festplatte synchronisiert. Bei `ext3` hat das zur Folge, dass sämtliche Änderungen an irgendwelchen Dateien innerhalb von fünf Sekunden physikalisch auf der Festplatte gespeichert werden. Dieses Standardverhalten ist zwar nicht besonders effizient, dafür aber sehr sicher: Selbst bei Totalabstürzen und Stromausfällen sind massive Datenverluste äußerst selten. `data=ordered` hat bei `ext3` eine unerfreuliche Nebenwirkung: Bei jedem Aufruf der

fsync-Funktion wird nicht nur eine bestimmte Datei, sondern das gesamte Dateisystem synchronisiert. Das kann zu spürbaren Verzögerungen führen.

Bei ext4 wird das Journal zwar ebenfalls alle fünf Sekunden synchronisiert, die eigentlichen Datenänderungen werden aber aufgrund der *delayed allocation* (siehe unten) oft erst viel später gespeichert. Nur ein expliziter Aufruf der fsync-Funktion stellt die sofortige physikalische Speicherung einer Datei sicher! (Glücklicherweise erfordert fsync bei ext4 nicht, dass das gesamte Dateisystem synchronisiert werden muss. Die Funktion wird daher wesentlich schneller ausgeführt.)

- » **data=writeback:** Dieser Modus ähnelt dem ordered-Modus. Der einzige Unterschied besteht darin, dass das Journal und die Dateioperationen nicht immer vollständig synchron sind. Das Dateisystem wartet mit den *Committed*-Einträgen im Journal nicht auf den Abschluss der Speicheroperation auf der Festplatte. Damit ist das Dateisystem etwas schneller als im ordered-Modus. Nach einem Crash ist die Integrität des Dateisystems weiterhin sichergestellt. Allerdings kann es vorkommen, dass veränderte Dateien alte Daten enthalten. Dieses Problem tritt nicht auf, wenn Anwendungsprogramme – wie im POSIX-Standard vorgesehen – den Speichervorgang mit fsync abschließen (siehe oben).
- » **data=journal:** Im Gegensatz zu den beiden anderen Modi werden jetzt im Journal auch die tatsächlichen Daten gespeichert. Dadurch müssen alle Änderungen *zweimal* gespeichert werden (zuerst in das Journal und dann in die betroffene Datei). Deswegen ist dieser Modus deutlich langsamer. Dafür können nach einem Crash Dateien wiederhergestellt werden, deren Änderungen bereits vollständig in das Journal (aber noch nicht in der Datei) eingetragen worden sind.

Grundsätzlich wird das Journal alle fünf Sekunden physikalisch auf der Festplatte gespeichert. Diese Zeitspanne kann durch die mount-Option `commit` verändert werden. Wenn das Paket `laptop-mode` installiert und konfiguriert ist und ein Notebook im Batteriebetrieb läuft, ist die `commit`-Zeitspanne wesentlich höher. Intern kümmert sich der in den Kernel integrierte Journaling-Dämon `kjournald` um die regelmäßige Aktualisierung der Journaling-Datei. Dieser Prozess wird automatisch gestartet, sobald ein ext3- oder ext4-Dateisystem mit `mount` in den Verzeichnisbaum eingebunden wird.

Standardmäßig kommt bei ext3-Dateisystemen der Modus `data=writeback` zum Einsatz, bei ext4-Dateisystemen hingegen `data=ordered` in Kombination mit der im folgenden beschriebenen *delayed allocation*. Um einen bestimmten Journaling-Modus explizit auszuwählen, geben Sie bei `mount` oder in `/etc/fstab` die Option `data=xxx` an.

Delayed allocation

Die aus Performance-Sicht wichtigste Neuerung in ext4 ist die sogenannte *delayed allocation* – eine Funktion, die es auch in vielen anderen modernen Dateisystemen gibt. *Delayed allocation* (auch *allocation on flush* genannt) bedeutet, dass bei Änderungen die Datenblöcke zur Speicherung von Dateiänderungen nicht sofort reserviert werden, sondern erst zu dem

Zeitpunkt, zu dem die Daten physikalisch gespeichert werden – und das kann durchaus eine halbe Minute dauern. Das bringt zwei wesentliche Vorteile mit sich: Zum einen können nun Speicheroperationen gebündelt werden, was die Geschwindigkeit erhöht und die Fragmentierung des Dateisystems mindert. Zum anderen kommt es bei temporären Dateien, die nur wenige Sekunden existieren, oft zu gar keiner physikalischen Speicherung.

Leider hat *delayed allocation* auch Nachteile: Das Hauptproblem besteht darin, dass Metadaten (also Informationen über den Zustand einer Datei) oft schon vor den eigentlichen Änderungen gespeichert werden. In der ursprünglichen Implementierung des ext4-Treibers führte das dazu, dass eine geänderte, aber noch nicht synchronisierte Datei nach einem Absturz plötzlich leer ist. Dieses Problem wurde mittlerweile behoben: Wenn zur Änderung vorhandener Dateien die Funktionen `rename` oder `ftruncate` eingesetzt werden (das sind die üblichen Vorgehensweisen), verzichtet ext4 auf die *delayed allocation*. Es ist sogar möglich, die *delayed allocation* durch die `mount`-Option `node1alloc` komplett zu deaktivieren. Das ist aber mit erheblichen Effizienzeinbußen verbunden und macht einen Teil der Performance-Fortschritte in ext4 zunichte.

Ein neues ext-Dateisystem einrichten

ext2-, ext3- und ext4-Dateisysteme werden mit `mkfs.ext2`, `mkfs.ext3` oder `mkfs.ext4` formatiert. Normalerweise müssen Sie dem Kommando lediglich den Device-Namen der Partition bzw. des *logical volumes* (bei LVM-Systemen) übergeben.

Im folgenden Beispiel wird auf einem 20 GByte großen *logical volume* (also einer durch LVM verwalteten Partition) ein ext4-Dateisystem eingerichtet. `mke2fs` entscheidet sich selbstständig für eine Blockgröße von 4 kByte und für 1.310.720 I-Nodes. Das bedeutet, dass Sie im Dateisystem maximal 1,3 Millionen Dateien anlegen können. Die durchschnittliche Dateigröße würde dann 16 kByte betragen. Wenn Sie mehr kleinere oder weniger größere Dateien speichern möchten, können Sie mit `-i n` angeben, nach wie vielen Bytes jeweils ein I-Node vorgesehen werden soll. (Wenn die durchschnittliche Dateigröße kleiner ist als `n`, limitiert nicht die Größe der Partition, sondern die I-Node-Anzahl das Dateisystem.) Beachten Sie, dass die absolute Anzahl der I-Nodes nicht mehr verändert werden kann, auch nicht bei einer späteren Vergrößerung des Dateisystems! In den meisten Fällen ist der Vorgabewert von `mkfs.ext4` zweckmäßig.

```
root# mkfs.ext4 /dev/mapper/vg1-test
mke2fs 1.41.4 (27-Jan-2009)
Dateisystem-Label=
OS-Typ: Linux
Blockgröße=4096 (log=2)
Fragmentgröße=4096 (log=2)
1310720 Inodes, 5242880 Blöcke
262144 Blöcke (5.00%) reserviert für den Superuser
Erster Datenblock=0
Maximale Dateisystem-Blöcke=4294967296
```

```

160 Blockgruppen
32768 Blöcke pro Gruppe, 32768 Fragmente pro Gruppe
8192 Inodes pro Gruppe
Superblock-Sicherungskopien gespeichert in den Blöcken:
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
    4096000

```

```

Schreibe Inode-Tabellen: erledigt
Erstelle Journal (32768 Blöcke): erledigt
Schreibe Superblöcke und Dateisystem-Accountinginformationen: erledigt

```

Das Dateisystem wird automatisch nach jeweils 35 Einhäng-Vorgängen bzw. alle 180 Tage überprüft, je nachdem, was zuerst eintritt. Veränderbar mit `tune2fs -c` oder `-t`.

Größe des Dateisystems ändern

Mit `resize2fs` können Sie ein ext-Dateisystem vergrößern oder verkleinern. Beachten Sie, dass Sie bei einer Vergrößerung *vorher* die zugrunde liegende Partition oder das LV vergrößern müssen, bei einer Verkleinerung die Partition oder das LV aber erst *nachher* verkleinern dürfen! Im folgenden Beispiel wird das LV mit `lvextend` vergrößert. (Details zur LVM-Administration folgen in Abschnitt 26.6.)

```

root# lvextend -L 40G /dev/mapper/vg1-test
    Extending logical volume test to 40,00 GB
    Logical volume test successfully resized
root# resize2fs /dev/mapper/vg1-test
resize2fs 1.41.4 (27-Jan-2009)
Das Dateisystem auf /dev/mapper/vg1-test ist auf /test eingehängt;
    Online-Größenveränderung nötig
old desc_blocks = 2, new_desc_blocks = 3
Führe eine Online-Größenänderung von /dev/mapper/vg1-test
    auf 10485760 (4k) Blöcke durch.
Das Dateisystem auf /dev/mapper/vg1-test ist nun 10485760 Blöcke groß.

```

Eine Vergrößerung des Dateisystems ist im laufenden Betrieb möglich. Für eine Verkleinerung muss das Dateisystem aus dem Verzeichnisbaum gelöst werden.

Konvertierung von ext3 nach ext4

Um ein vorhandenes ext3-Dateisystem in ein ext4-Dateisystem umzuwandeln, führen Sie einfach das unten angegebene Kommando aus. `tune2fs` kann im laufenden Betrieb ausgeführt werden; um die neuen ext4-Funktionen zu nutzen, muss das Dateisystem aber neu in den Verzeichnisbaum eingebunden werden.

```

root# tune2fs -0 extents /dev/sda5
root# umount /dev/sda5
root# mount /dev/sda5 /home

```

Vielleicht wundern Sie sich über den Kommandonamen `tune2fs`: Das hohe Ausmaß der Kompatibilität zwischen den verschiedenen ext-Dateisystemversionen drückt sich auch dadurch aus, dass diverse Administrationswerkzeuge weiterhin die Versionsnummer 2 im Kommandonamen haben, obwohl sie auch für neuere Versionen eingesetzt werden können.

Die nachträgliche Umwandlung eines ext3-Dateisystems hat den Nachteil, dass vorhandene Dateien keine *extents* nutzen (nur neue Dateien). Abhilfe würde das Defragmentierprogramm `e4defrag` schaffen, das aber noch nicht zur Verfügung steht.

Dateisystemüberprüfung

ext-Dateisysteme werden beim Rechnerstart regelmäßig auf Fehler überprüft, und zwar nach einer bestimmten Anzahl von `mount`-Vorgängen (standardmäßig 36) bzw. nach einer gewissen Zeit (6 Monate), je nachdem, welches Kriterium vorher erfüllt war.

Trotz der Journaling-Funktionen ist eine Überprüfung des Dateisystems hin und wieder sehr zu empfehlen, zumindest ein- bis zweimal pro Jahr! Zum einen werden so eventuelle Hardware-Fehler der Festplatte erkannt. Zum anderen kann es sein, dass die Dateisystemtreiber noch unbekannte Fehler enthalten. Je früher daraus resultierende Fehler korrigiert werden, desto kleiner ist der potenzielle Schaden.

Eine manuelle Überprüfung können Sie einfach mit dem Kommando `fsck.ext2/ext3/ext4` durchführen. Die betreffende Partition darf während der Kontrolle allerdings nicht gerade verwendet werden, d. h., Sie müssen gegebenenfalls vorher `umount` ausführen.

```
root# fsck.ext4 -f /dev/mapper/vg1-test
e2fsck 1.41.4 (27-Jan-2009)
Durchgang 1: Prüfe Inodes, Blocks und Größen
Durchgang 2: Prüfe Verzeichnis Struktur
Durchgang 3: Prüfe Verzeichnis Verknüpfungen
Durchgang 4: Überprüfe die Referenzzähler
Durchgang 5: Überprüfe Gruppe Zusammenfassung
/dev/mapper/vg1-test: 21357/1310720 Dateien (1.3% nicht zusammenhängend),
    2062135/5242880 Blöcke
```

Meist stellt sich bei der Überprüfung heraus, dass alles in Ordnung ist. Andernfalls werden die Reste nicht mehr rekonstruierbarer Dateien im `/lost+found`-Verzeichnis der jeweiligen Partition gespeichert. Falls es sich um Textdateien gehandelt hat, können Sie vielleicht aus den Überresten noch brauchbare Informationen entnehmen.

Die aktuellen Intervalle für die automatische Überprüfung des Dateisystems können Sie mit `tune2fs` feststellen und verändern. Dabei geben Sie mit `-c` die maximale `mount`-Anzahl und mit `-i` das Zeitintervall in Tagen an:

```

root# tune2fs -l /dev/mapper/vg1-test
...
Mount count:          1
Maximum mount count: 35
Last checked:        Wed Jul 15 11:45:54 2009
Check interval:      15552000 (6 months)
...
root# tune2fs -c 100 -i 90 /dev/mapper/vg1-test
Setting maximal mount count to 100
Setting interval between check 7776000 seconds

```

Partitionsnamen und UUID einstellen

Zusammen mit jedem ext-Dateisystem kann ein Partitionsname und eine UUID-Nummer gespeichert werden. Während Ubuntu vom Partitionsnamen keinen Gebrauch macht, verwendet es die UUID-Nummer sowohl in `/etc/fstab` als auch in der GRUB-Konfiguration.

Den UUID (*Universal Unique Identifier*) erhält das Dateisystem automatisch bei der Formatierung. Sie können den UUID mit `tune2fs -l` ermitteln und bei Bedarf mit `tune2fs -U` ändern. Die Veränderung kann im laufenden Betrieb erfolgen, `umount` ist nicht erforderlich.

```

root# tune2fs -l /dev/sda1 | grep UUID
Filesystem UUID:          efff58f1-3674-439e-a110-5b545a7dc150
root# tune2fs -U random /dev/sda1                (zufällige UUID)
root# tune2fs -U f7c49568-8955-4ffa-9f52-9b2ba9877021 /dev/sda1 (eigene UUID)

```

Mit `e2label` können Sie den internen Namen eines ext-Dateisystems (*filesystem volume name*) ermitteln bzw. einstellen:

```

root# e2label /dev/sda1 mylabel

```

Fragmentierung des Dateisystems

Unter »Fragmentierung« versteht man den Zustand, dass einzelne Dateien nicht in aneinanderliegenden Blöcken, sondern über die ganze Partition verteilt gespeichert werden. Dazu kann es kommen, wenn abwechselnd Dateien gelöscht, neu angelegt, verlängert oder verkürzt werden. Die Fragmentierung kann den Dateizugriff erheblich verlangsamen.

Die ext2/3/4-Treiber versuchen eine Fragmentierung so gut wie möglich zu vermeiden. Das gelingt allerdings nur, wenn das Dateisystem nie zu mehr als ca. 90 Prozent mit Daten gefüllt ist.

Es gibt momentan keine Defragmentierungswerkzeuge für die ext-Dateisysteme. Für ext4 ist ein derartiges Programm unter dem Namen `e4defrag` aber immerhin in Entwicklung. Es wird – wenn es einmal fertiggestellt ist – eine Defragmentierung im laufenden Betrieb ermöglichen.

Auf ext2- und ext3-Dateisysteme unter Windows zugreifen

Eigentlich enthält dieses Buch nur Tipps für Ubuntu. Dieser Abschnitt ist eine Ausnahme, denn dieser Tipp gilt für Windows. Durch die Installation des kleinen und kostenlosen IFS-Treibers können Sie von Windows aus Ihre Linux-Partitionen lesen und schreiben! Es gibt zwei Voraussetzungen: Sie müssen Windows 2000, XP, Vista oder eine neuere Version einsetzen (nicht Windows 9x/ME), und Ihre Linux-Partitionen müssen das Dateisystem ext2 oder ext3 nutzen (nicht ext4!).

Zur Installation des ext2/3-Treibers laden Sie die Datei Ext2IFS_n.exe von der folgenden Website herunter und führen sie unter Windows aus. (Die Datei enthält das Setup-Programm.)

<http://www.fs-driver.org/>

Am Ende des Installationsprozesses können Sie für jede Linux-Partition angeben, unter welchem Laufwerksbuchstaben sie in das Dateisystem eingebettet werden soll. Wenige Sekunden später (ohne Windows-Neustart!) können Sie bereits auf Ihre Linux-Dateien zugreifen. Die Zuordnung zwischen Linux-Partitionen und Laufwerksbuchstaben ändern Sie bei Bedarf mit EINSTELLUNGEN|SYSTEMSTEUERUNG|IFS DRIVES (siehe Abbildung 26.3).



Abbildung 26.3: Linux-Partitionen unter Windows nutzen

Die Verwendung des ext2/3-Treibers unterliegt einigen Einschränkungen:

- » ext4-Dateisysteme werden nicht unterstützt.
- » Die Swap-Partition kann unter Windows nicht genutzt werden. Der IFS-Treiber kommt nur mit Linux-Partitionen zurecht, die das Dateisystem ext2 oder ext3 nutzen. Die Swap-Partition enthält aber gar kein Dateisystem.
- » Da Windows ein anderes System zur Verwaltung von Zugriffsrechten nutzt als Linux, können die Linux-Zugriffsrechte unter Windows weder gelesen noch verändert werden. Wenn Sie neue Dateien erzeugen, bekommen diese Dateien dieselben Zugriffsrechte wie das Verzeichnis, in dem sie enthalten sind.
- » Die Integrität des ext3-Dateisystems ist nur dann sichergestellt, wenn Sie bei jedem Wechsel zwischen Linux und Windows das aktuelle Betriebssystem richtig herunterfahren (*nicht* in den Standby-Modus oder Ruhezustand).

Wenn Sie Linux-Dateien nur lesen möchten, können Sie als sichere Alternative zum IFS-Treiber das Programm *Explore2fs* einsetzen. Es kann *ext2/3*-Dateisysteme lesen, aber nicht verändern. Persönlich ziehe ich *Explore2fs* deswegen vor, auch wenn das Programm nicht sehr komfortabel ist. Sie finden es unter:

<http://www.chrysocome.net/explore2fs>

26.4 Windows-Dateisysteme (VFAT, NTFS)

Mit den Dateisystemen VFAT und NTFS kommen Sie unter Ubuntu normalerweise nur dann in Berührung, wenn Sie Daten mit Windows-Rechnern oder mit anderen elektronischen Geräten (z. B. Digitalkameras) austauschen. Normalerweise kümmert sich Gnome selbstständig darum, Windows-Partitionen der lokalen Festplatte, externe Festplatten oder Memory-Sticks in das Dateisystem einzubinden. Wenn Sie auf einer Festplattenpartition oder einem Memory-Stick ein Windows-Dateisystem einrichten möchten, verwenden Sie dazu am einfachsten *Palimpsest* (SYSTEM|SYSTEMVERWALTUNG|LAUFWERKSVERWALTUNG; siehe auch Seite 602).

Dieser Abschnitt gibt einige Tipps, wie Sie bei Bedarf auf Kommandoebene mit Windows-Partitionen umgehen. Um eine Windows-Partition in das Dateisystem einzubinden, führen Sie die folgenden *mount*-Kommandos aus. Dabei müssen Sie natürlich */dev/sda1* bzw. */dev/sda2* durch den richtigen Device-Namen ersetzen und ein bereits existierendes Verzeichnis angeben, bei dem das Windows-Dateisystem in den Verzeichnisbaum eingebunden wird. Die Optionen *utf8* und *uid=1000* bewirken, dass die VFAT-Dateinamen Linux-typisch in Unicode-Zeichenketten übersetzt werden und dass der Ubuntu-Standardbenutzer die Dateien lesen und schreiben kann. Der NTFS-Treiber verwendet standardmäßig Unicode, sodass diese Option beim zweiten Kommando entfällt.

```
user$ sudo mount -t vfat -o utf8,uid=1000 /dev/sda1 /verzeichnis (VFAT)
user$ sudo mount -t ntfs -o uid=1000 /dev/sda2 /verzeichnis (NTFS)
```

Wenn Sie eine Windows-Partition einer lokalen Festplatte ständig in Ihren Verzeichnisbaum integrieren möchten, fügen Sie die folgenden Zeilen zu */etc/fstab* hinzu:

```
# in /etc/fstab
/dev/sda1 /media/win1 vfat utf8,uid=1000 0 0
/dev/sda2 /media/win2 ntfs uid=1000 0 0
```

Um in einer Partition ein Windows-Dateisystem einzurichten, führen Sie die folgenden Kommandos aus. (Die Partition muss vorher als VFAT- bzw. NTFS-Partition eingerichtet werden!)

```
user$ sudo mkfs.vfat /dev/sda1 (VFAT)
user$ sudo mkfs.ntfs /dev/sda2 (NTFS)
```