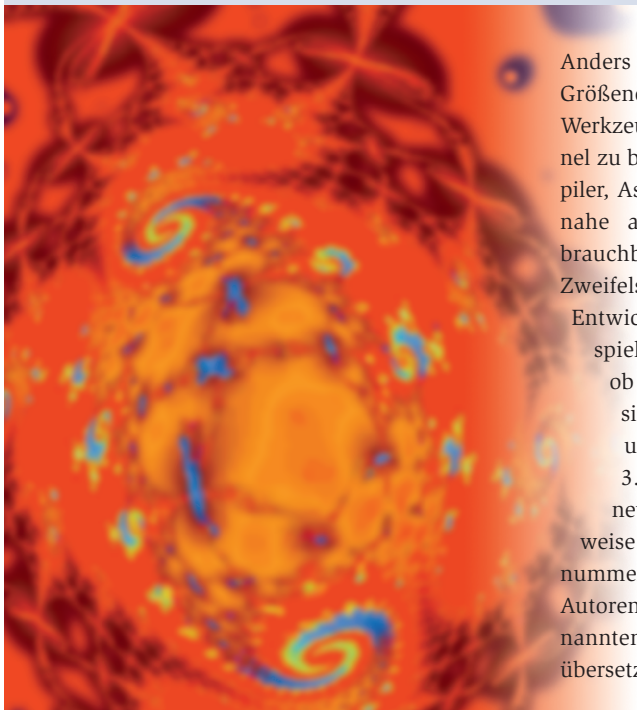
Kernel 2.6 installieren: Eine Schritt-für-Schritt-Anleitung

Meister-Installateur

Multimedia-Applikationen laufen runder, die Systemleistung steigt, Hardware wird besser unterstützt und mit IPsec und den neuen Security-Modulen lässt sich die Systemsicherheit weiter erhöhen. Es gibt viele gute Gründe, jetzt schon Erfahrungen mit dem neuesten Kernel zu sammeln. Eva-Katharina Kunst, Jürgen Quade



Fünf Millionen Zeilen Code, gepackt in einem 32-MByte-Archiv: Bereits die äußeren Maße des neuen Kernels [1] beeindrucken. Keine Kleinigkeit also, die es zu konfigurieren und installieren gilt. Haben Sie schon mal einen 2.4er Kernel selbst übersetzt und konfiguriert, werden Sie sich auch im 2.6er schnell zurechtfinden. Neulinge seien allerdings gewarnt: Bei derart tiefen Eingriffen ins System lauern Fallstricke, die jeden unbedachten Schritt bestrafen.

Im Folgenden ist zu lesen, wie Sie auf direktem Weg zu einem fünktionstüchtigen Kernel 2.6 kommen. Mit etwas Geduld stimmen Sie ihn optimal auf Ihr System ab. Für zusätzliche Hintergrundinformationen sind das Post-Halloween-Dokument von Dave Jones (deutsche Übersetzung: [2]) sowie das Archiv der Kernel-Mailingliste [4] gute Quellen.

Anders als bei vielen Projekten dieser Größenordnung sind keine besonderen Werkzeuge nötig, um einen eigenen Kernel zu bauen. Die Standardzutaten Compiler, Assembler und Make sind auf beinahe allen Linux-Systemen in einer brauchbaren Version vorhanden, im Zweifelsfall müssen Sie nur noch die Entwicklerpakete nachinstallieren. Es spielt dabei keine besondere Rolle, ob Sie den Kernel mit der GCC-Version 2.95, 3.2 oder 3.3 übersetzen und ob Make genau in der Version 3.80 zum Einsatz kommt. Der neue Kernel 2.6 reagiert vergleichsweise unempfindlich auf die Versionsnummern. Auf dem Debian-System der Autoren ließ er sich mit allen drei genannten Compilerversionen erfolgreich übersetzen.

Neue Modutils sind Voraussetzung

Einzig die Abhängigkeit von den Modutils gilt es zu beachten (Schritt 1 in Abbildung 1). Die Modutils sind für das Laden und Entladen der Kernelmodule zuständig, die Linux-Entwickler haben diesen Mechanismus aber komplett überarbeitet. Module des 2.6er Kernels lassen sich nur mit einer neuen Version des Programms »insmod« laden.

Das neue Insmod greift bei Bedarf auch auf die alte Variante zurück, daher ist zusätzlich »insmod.old« vorhanden. Sie können somit leicht feststellen, ob eine 2.6-taugliche Version dieses Programms installiert ist: Wenn unter »/sbin/« oder »/usr/local/sbin« ein Programm namens »insmod.old« eingetragen ist, dann sollte »insmod« mit dem neuen Kernel zurechtkommen.

Bei vielen aktuellen Distributionen ist das glücklicherweise der Fall (zum Beispiel Suse 8.2 und 9.0). Wenn nicht, können Sie bei Debian (oder Knoppix) die neuen Modutils mit Hilfe von »apt-get« installieren:

```
apt-get install module-init-tools
```

Auf Nicht-Debian-Systemen holen Sie von [3] eine aktuelle Version der Modutils, zum Beispiel 0.9.15-pre4. Beim Auspacken des Programmpakets entsteht das Verzeichnis »module-init-tools-0.9.15-pre4« inklusive einer »README«-Datei, die kurz und knapp die wichtigsten Informationen enthält. Sie erfahren dort, dass im Wesentlichen die folgenden Kommandos auszuführen sind:

```
./configure --prefix=/
make moveold # Nur beim ersten Mal!
make
make install
```

Das ursprüngliche Programm »/sbin/insmod« heißt danach »/sbin/insmod

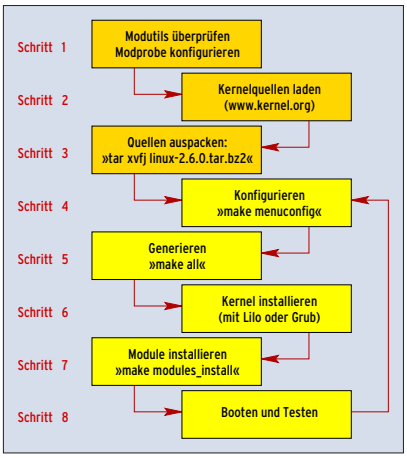


Abbildung 1: In acht Schritten zum eigenen Kernel. Gerade Kernel-Neulinge sollten sich darauf einstellen, die Schritte 4 bis 8 mehrfach durchzuführen.

.old«. Das neue »/sbin/insmod« ruft die alte Version auf, wenn Ihr System einen Kernel 2.4 gebootet hat. Arbeitet aber ein 2.6er, lädt das Tool die Module gemäß der neuen Spezifikation. Gleiches gilt für die Programme »/sbin/rmmod« und »/sbin/modprobe«.

Neue Konfigurationsdateien für Kernelmodule

Die neuen Modutils benötigen auch eine neue Konfigurationsdatei: »/etc/modprobe.conf«. Ist dieses File auf Ihrem Rechner noch nicht vorhanden, sollten Sie es mit »./generate-modprobe.conf /etc/modprobe.conf« erzeugen. Falls Sie Anhänger des Device-Filesystems sind, müssen Sie noch »modprobe.devfs« ins Verzeichnis »/etc/« kopieren.

Sind damit alle Voraussetzungen erfüllt, geht es mit Schritt 2 weiter (siehe Abbildung 1): Sie holen die 2.6er Kernelquellen aus dem Netz. Die Eingangsseite des Kernelarchivs [1] listet die aktuellen Versionen. Um die vollständigen Quellen und nicht nur die Patches zu erhalten, müssen Sie auf das unscheinbare »F« (Full, also die Vollversion) klicken, das sich rechts des Eintrags »2.6.0-test11« befindet (siehe Abbildung 2). Das mit Bzip2 komprimierte Quellcode-Archiv packen Sie dann im Verzeichnis »/usr/src« aus (Schritt 3):

```
cd /usr/src
tar xvfj /tmp/linux-2.6.0-test11.tar.bz2
```

Nach dem Auspacken folgt der schwierigste und langwierigste Schritt (Nummer 4): die Kernelkonfiguration. Aus Sicht der Autoren ist »make menuconfig« dazu am besten geeignet (siehe Abbildung 3). Das Kommando erstellt und startet ein Ncurses-basiertes Konfigurationsprogramm, das Sie in jedem normalen Konsolenfenster bedienen können. Auch eine grafische Variante auf QT-Basis steht zur Auswahl (Abbildung 4). Um diese zu nutzen, geben Sie »make xconfig« ein.

Geduldsprobe

Trotz vieler neuer Optionen ist die Konfiguration übersichtlicher geworden. Abweichend von der Standardkonfiguration müssen Sie unbedingt den Prozessortyp (Menüpunkt »Processor type and features«) und die zugehörige Systemarchitektur auswählen. Auch beim Dateisystem sollten Sie aufpassen: Wenn Sie beispielsweise Suse als Distribution verwenden, werden Sie den »Reiserfs support« benötigen. Brauchen Sie be-

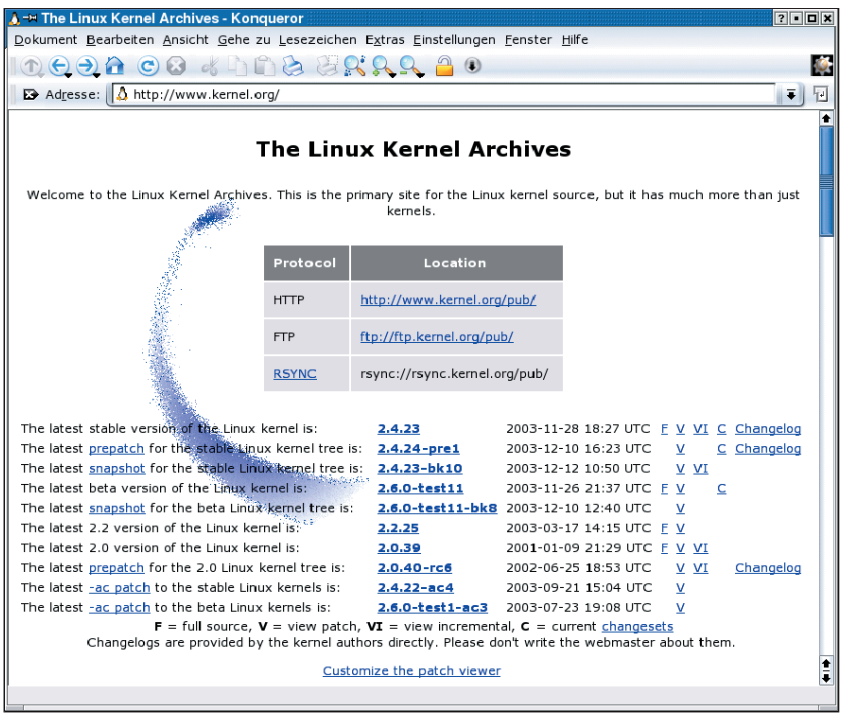


Abbildung 2: Das Kernelarchiv enthält alle Versionen, bereits die Startseite verlinkt zu den aktuellen Kerneln. Ältere Ausgaben sind über die Links in der oberen Tabelle zu erreichen.

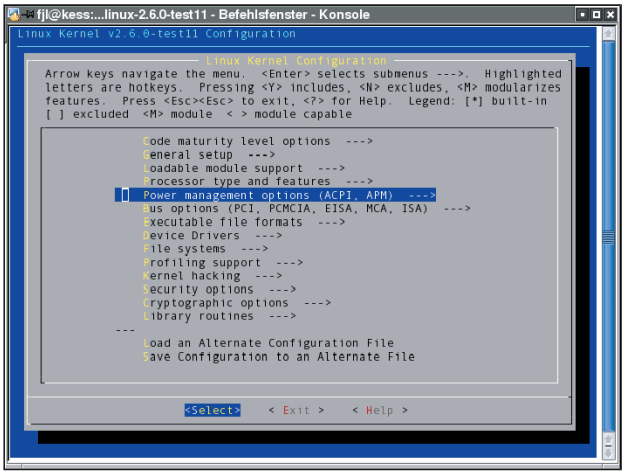


Abbildung 3: Die Konfiguration des Kernels ist trotz erweiterter Optionen in 2.6 erheblich übersichtlicher geworden. Hier ist das Konfigurationswerkzeug »make menuconfig« in Aktion zu sehen.

stimmte Komponenten (zum Beispiel das SCSI-Subsystem) nicht, dann sollten Sie diese abwählen.

Zu beinahe jedem Auswahlpunkt gibt es eine Hilfestellung. Alle Menüpunkte durchgehen kann beim ersten Mal gut zwei bis drei Stunden in Anspruch nehmen. Das ist nicht wirklich viel, schließlich werden dabei mehr als fünf Millionen Zeilen Code konfiguriert.

Alte Vorlagen weiter nutzen

Wenn Sie für Ihr System bereits eine ältere, funktionierende 2.6-Konfiguration besitzen (siehe Kasten „Kernel 2.6 unter Suse“), können Sie diese als Ausgangsbasis verwenden. Andernfalls sollten Sie den minimalistischen Weg beschreiten: Wählen Sie nur die Komponenten aus, die Sie dringend benötigen. Läuft der Kernel erst einmal, können Sie jederzeit neue Features und Treiber hinzufügen.

Wenn Sie die Einstellungen in der Konfigurationsdatei gespeichert haben, ist es an der Zeit, den Kernel zu generieren. Wer Debian oder Knoppix benutzt, findet die weiteren Schritte im Kasten „Schritt 5 bis 7 unter Debian“. Alle anderen tippen im Quellverzeichnis (als Root) das Kommando »make all« ein (Schritt 5). Warnungen, die zwischendurch über den Bildschirm huschen, können Sie getrost übersehen. Auf einem System mit reichlich Speicher und einem Intel-Prozessor mit 2,6 GHz vergehen mehr als zehn Minuten, bis der Prompt wieder erscheint.

Sollte das Generieren wider Erwarten vorzeitig abbrechen, bleibt Ihnen nichts anderes übrig, als die Ausgaben aufmerksam zu studieren. Sie müssen dann die fehlerhafte Komponente finden und den Kernel neu konfigurieren (durch den Aufruf von »make menuconfig«). Wählen Sie dabei die problematische Komponente ab.

Installation des Kernels und seiner Module

Ist alles reibungslos gelaufen, geht es an die eigentliche Installation (Schritt 6 und 7). Die Kernelmodule einspielen ist recht simpel: »make modules_install« aufrufen – und kurze Zeit später sind alle Module im neuen Verzeichnis »/lib/modules/2.6.0-test11/« abgelegt. Beim Kernel müssen Sie jedoch selbst Hand anlegen und ihn in das Verzeichnis »/boot/« kopieren:

```
cp /usr/src/linux-2.6.0-test11/arch/i386/
boot/bzImage /boot/vmlinuz-2.6.0
```

Der Bootloader sollte beim Hochfahren des Rechners den neuen Kernel zur Auswahl anbieten. Für Lilo [5] müssen Sie die Datei »/etc/lilo.conf« anpassen und für Grub [6] die Datei »/boot/grub/menu.lst«. Für beide ist ein neuer Eintrag fällig, der den Namen des Kernels (etwa »/boot/vmlinuz-2.6.0«) und einen symbolischen Namen – den späteren Menüpunkt – enthält. Diesen Eintrag müssen Sie nur ein Mal erstellen.

Schritt 5 bis 7 unter Debian

Die Unstable-Version von Debian (Sarge) bietet bereits mehrere 2.6er Kernel zum Download an. Wer einen auf sein System optimal angepassten Kernel bauen möchte, muss jedoch selbst Hand anlegen. Rufen Sie nach dem Konfigurieren (Schritt 4) im Quellcode-Verzeichnis »make-kpkg« auf:

```
./make-kpkg kernel_image
```

Dieses Skript generiert Kernel und Module (Schritt 5) und erstellt im Verzeichnis unterhalb der Quellen (meist »/usr/src/«) das Debian-Paket »kernel_image«. Bevor Sie dies Paket installieren (Schritt 6 und 7), sollten Sie im Bootloader einen neuen Auswahlpunkt hinzufügen. Wenn Sie das Image-Paket mit »dpkg -i kernel-image-*Kernelversion*.deb« installieren, ruft der Debian-Installer auch »lilo« auf. Spätestens jetzt sollte Lilo den neuen Kernel kennen, um ihn später beim Booten zur Auswahl anzubieten.

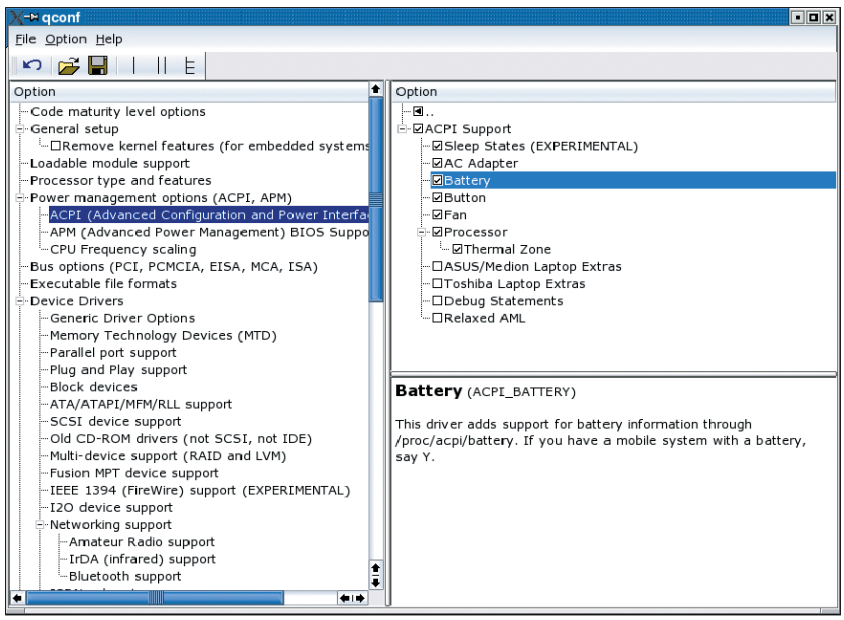


Abbildung 4: Im QT-basierten »make xconfig« hat der Installateur immer den Überblick: Links die Konfigurationsbereiche, oben die einzelnen Optionen und unten den Hilfetext zum aktuellen Parameter.

Falls Sie später den Kernel um einen neuen Treiber erweitern, müssen Sie »lilo.conf« und »menu.lst« normalerweise nicht mehr anfassen. Im Fall von Lilo als Bootloader dürfen Sie aber nicht vergessen, nach jeder Kernel-Neuinstallation einmal »lilo« zu starten.

Boot-Probe

Jetzt darf gebootet werden. Sollte der Kernel den Bootvorgang abbrechen, müssen Sie die bis dahin erfolgten Ausgaben auf der Konsole genau auswerten. Die Meldungen geben fast immer Aufschluss darüber, warum der Kernel nicht zu Ende booten kann. In vielen Fällen fehlt nur ein Treiber, beispielsweise der für den Zugriff auf die Platten oder einer für das Root-Filesystem.

Sollte der Bildschirm gar ganz schwarz bleiben, fällt die Diagnose schwer. Oft sind Probleme mit dem Framebuffer-Device schuld. Zwei Aktionen versprechen Abhilfe:

- Eliminieren Sie in »/etc/lilo.conf« die Option »vga = *XXX*« oder setzen Sie sie auf »vga = normal«.
- Überprüfen Sie, ob bei der Kernelkonfiguration in der Datei »/usr/src/linux-2.6.0-test11/.config« die Option »CONFIG_VT« auf »y« gesetzt ist.

Hat der neue Kernel durchgebootet, müssen Sie eventuell noch Gerätetreiber für Komponenten erstellen, die vom Standardkernel nicht unterstützt werden. Als Beispiel seien hier die Nvidia-Grafikkarten erwähnt [8].

Zum Schluss bleibt noch, die Boot-Messages zu durchforsten. Ein Aufruf von »dmesg« zeigt Ihnen die Nachrichten des letzten Bootvorgangs noch einmal an. Falls hier noch Probleme signalisiert werden, geht es wieder zurück zu Schritt 4, der Konfiguration. Im anderen Fall läuft Ihr eigener 2.6er Kernel. Herzlichen Glückwunsch! *(fjl)*

Infos

[1] Kernel-Quellen: [http://www.kernel.org]

[2] Dave Jones, „The Post-Halloween Document", deutsche Übersetzung: [http://www.kubieziel.de/computer/halloween-german.html]

[3] Neue Modutils: [http://www.kernel.org/pub/linux/kernel/people/rusty/modules/]

[4] Linux-Kernel-Mailingliste: [http://lkml.org]

[5] Lilo: [http://lilo.go.dyndns.org/]

[6] Grub: [http://www.gnu.org/software/grub/grub.de.html]

[7] Kernel-2.6-Pakete für Suse 9.0: [ftp://ftp.suse.com/pub/suse/i386/9.0/unsorted/kernel-2.6/]

[8] Nvidia-Module mit Kernel 2.6 nutzen: [http://www.minion.de/]

Die Autoren

Eva-Katharina Kunst, Journalistin, und Jürgen Quade, Professor an der Hochschule Niederrhein, sind seit den Anfängen von Linux Fans von Open Source.

Kernel 2.6 unter Suse

Suse Linux 9.0 ist bereits für Kernel 2.6 vorbereitet. Ein 2.6er Testkernel liegt der Distribution bei, neuere Versionen stehen auf [7] bereit. Allerdings bereitet das Programm »/usr/sbin/hwbootscan« in Kombination mit einem 2.6er Kernel unter Umständen Probleme. In diesem Fall sollten Sie es abschalten. Es ist nur für die zusätzliche Hardware-Erkennung beim Booten zuständig.

Zum Deaktivieren genügt es, im Init-Skript »/etc/init.d/hwscan« in Zeile 20 die Variable »HWBOOTSCAN_BIN« auszukommentieren:

```
# HWBOOTSCAN_BIN=/usr/sbin/hwbootscan
```

Der mitgelieferte 2.6er Kernel macht es viel einfacher, einen eigenen Kern zu bauen, da sich die vorhandene Konfiguration als gute Ausgangsbasis verwenden lässt. Installieren Sie den Testkernel per »rpm« (über Yast ist dies nicht möglich). Danach kopieren Sie dessen Konfigurationsdatei in das Verzeichnis der aktuellen Linux-Quellen:

```
cp vmlinuz-2.6.0-test5-5-default.config ➥
   /usr/src/linux-2.6.0-test11/
```

Nach dem Kopieren folgt ein »make menuconfig«-Aufruf. Er nimmt eventuell neu hinzugekommene Optionen automatisch in die Kernelkonfiguration auf. Für diesen und die folgenden Schritte müssen die Entwicklungswerkzeugen installiert sein: GCC, Make, Ncurses und für »make xconfig« auch QT3-Devel.

Ist der Kernel konfiguriert, folgen die Schritte 5 bis 8 wie im Text beschrieben. Eventuell müssen Sie zum Laden spezifischer Module noch die Datei »/etc/modprobe.conf« bearbeiten. Suse legt zusätzliche Einträge übrigens in die Datei »/etc/modprobe.conf.local«. Die Syntax dieser Files ist in den Manpages (»man modprobe.conf«) beschrieben.