SelfLinux-0.10.0

Der Linux-Kernel

Autor: Erwin Dogs (*edogs@t-online.de*)
Formatierung: Matthias Hagedorn (*matthias.hagedorn@selflinux.org*)

Dieses Kapitel führt in die grundsätzliche Arbeitsweise eines Linux-Systems ein.

Inhaltsverzeichnis

1 Der Kernel

Der Kernel verwaltet alle Betriebsmittel eines Computers. Er ist der erste Schritt von der Hardware zum Benutzer. Nach **innen**, zur Hardware hin, ist der Kernel ein Organisator. Er teilt den unterschiedlichen Geräten mit, was sie wann zu tun haben und sorgt für möglichst reibungslose, effiziente Abläufe. Nach **außen**, zum Benutzer hin, ist er ein Dienstleister:
Er bietet Funktionen an, die von Programmen verwendet werden können. Programme verlangen Prozessorzeit, Speicherplatz, möchten von Geräten lesen oder auf sie schreiben, sie wollen andere Programme starten oder sich selbst beenden. Für all diese Aktionen müssen sie eine Anfrage an den Kernel richten, die von diesem an die Hardware weitergegeben oder auch verweigert werden kann.

Der Kernel selbst ist ein Programm. Er liegt im Hauptspeicher, erhält dann und wann Prozessorzeit und kontrolliert in diesen Intervallen, welches Programm außer ihm etwas tun darf. Mit dem Systemstart muss der Kernel also in den Speicher geladen werden. Dies geschieht üblicherweise mittels eines Bootloaders wie `Lilo` oder `Grub` von der lokalen Festplatte. Es gibt jedoch auch andere Möglichkeiten. Booten von Diskette oder CD ist ebenso möglich wie das Booten über ein Netzwerk.

Um mit der Hardware kommunizieren zu können, benötigt ein Kernel spezielle Treiber. Des weiteren bestehen meist sehr konkrete Anforderungen an ein Rechnersystem - und diese Anforderungen können sich von System zu System stark unterscheiden. Benötigte Treiber und die konkreten Anforderungen an das System verlangen nach ganz unterschiedlichen Kernelfunktionen. Daher ist Kernel nicht gleich Kernel. Es wäre unsinnig, einen einzigen großen Kernel zu erstellen, der gewissermaßen **alles kann**. Ein enormer Speicherbedarf wäre die Folge. Es ist also eine Auswahl der Treiber und Funktionen notwendig.

Diese Auswahl geschieht unter Linux auf zweierlei Weise. Zum einen kann bereits vor dem Kompilieren des Kernels angegeben werden, welche Komponenten überhaupt Bestandteil des Kernels werden sollen. Zweitens bietet Linux die Möglichkeit, einzelne Komponenten als Module zu kompilieren. Module können dann zur Laufzeit geladen und wieder aus dem Speicher entfernt werden.

2 Erzeugen eines eigenen Kernels

2.1 Voraussetzungen

Um einen eigenen Kernel zu übersetzen, brauchen Sie einen C-Compiler, der die Quelltexte ins Binärformat übersetzt. Ob der C-Compiler installiert ist, erfahren Sie mit dem folgenden Befehl:

```
root@linux / # gcc --version
```

oder

```
root@linux / # cc --version
```

Nach Eingabe dieses Befehls sollte die Versionsnummer auf dem Schirm erscheinen.

2.2 Holen der Kernel-Quellen

Nun brauchen Sie noch die aktuellen Kernel-Quellen. Die Quellen des unveränderten Original-Kernels finden Sie auf

http://www.kernel.org

Schauen Sie auch auf der Downloadseite ihres Distributors, der oft eigene (angepaßte) Versionen des Kernels im RPM- oder DEB-Format bereitstellt.

Die Quelltexte des Kernels sollten Sie in /usr/src entpacken (bei der Installation eines Paketes wird das bereits oft automatisch erledigt).

Wechseln Sie in das Verzeichnis /usr/src, um zu schauen, ob ein Verzeichnis linux-<kernelversion> existiert. Wenn ja, kann es auch schon los gehen.

Wenn Sie die aktuelle Version des Originalkernels von www.kernel.org heruntergeladen haben, kopieren Sie die Datei linux-2.x.xx.tar.bz2 in das Verzeichnis /usr/src und entpacken Sie das Archiv mit dem folgenden Befehl:

```
root@linux /usr/src/ # tar jvxf linux-2.x.xx.tar.bz2
```

Nach dem Entpacken ist das Verzeichnis /usr/src/linux-<kernelversion> entstanden.

2.3 Hinweis zur Versionsnummer des Kernels

Es werden im Grunde zwei Arten von Kernels unterschieden. Das sind zum einen die **stabilen Kernel**, zum anderen die **Entwickler-Kernel**, an denen aktuell gearbeitet wird und die nicht stabil genug für die Benutzung sind. Diese unterscheiden sich anhand der Versionsnummer. Eine gerade Zahl an der zweiten Stelle der Versionsnummer kennzeichnet den stabilen Kernel-Zweig, eine ungerade den aktuellen Entwickler-Kernel. Also sind etwa 2.0, 2.2, 2.4 und 2.6 **stabile Kernel**. Die Versionen 2.1, 2.3 und 2.5 dagegen sind **Entwickler-Kernel**.

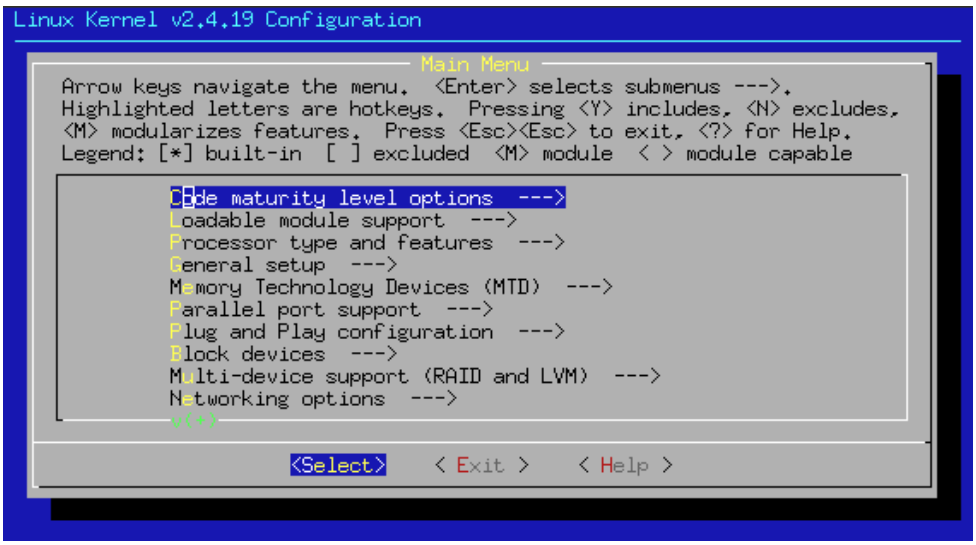
Die Zahl, die darauf folgt, ist der **Patchlevel** des Kernels. Werden Änderungen an einer Kernelversion von den

Kernel-Entwicklern freigegeben, so erhöht sich der Patchlevel jeweils um den Wert Eins. Der **Patchlevel** gibt somit Auskunft über die Aktualität des Kernels.

2.4 Konfiguration des Kernels

Sie haben im Wesentlichen drei Möglichkeiten, einen neuen Kernel zu konfigurieren.

* `make config` ist die einfachste aber nicht die beste von allen:
  innerhalb des Konsolenfensters werden dem Benutzer Fragen gestellt, die er einzeln beantworten muss. Da dies mit der Zeit sehr umfangreich und unübersichtlich geworden ist, ist diese Methode nicht zu empfehlen.
* `make menuconfig` stellt ein grafisches Menü in einer Konsole bereit. In diesem Menü kann man Konfigurationsoptionen auswählen. Meistens werden die Möglichkeiten **[X] [ ] [M]** (aktiviert, deaktiviert und als Kernelmodul kompiliert) angeboten. Um diesen Weg zu nutzen, benötigt Ihr System die ncurses-Bibliothek, die jedoch von den meisten Distributionen von ganz allein installiert wird.
* `make xconfig` bietet ebenfalls ein Auswahlmenü an, aber unter der X-Window-Oberfläche. Hierzu müssen Sie **Tcl/Tk** installiert haben.



make menuconfig in Aktion

Schauen Sie sich zunächst genau die Auswahloptionen an und vergleichen Sie, welche Hardware Sie in ihrem Computer haben (eventuell schauen Sie auch einmal in die Ausgaben der Befehle `lspci` und `dmesg`). Schauen Sie ggf. auf der Homepage des Herstellers nach, ob Sie eine Dokumentation zu Ihrer Hardware finden.

Nachdem Sie die Auswahlentscheidungen getroffen haben, können Sie Ihre Konfiguration speichern. Die Konfiguration wird dann in die Datei

/usr/src/linux-<kernelversion>/.config

geschrieben.

2.5 Übersetzung des Kernels

Nach der Konfiguration erfolgt die Übersetzung mit den Befehlen:

```
root@linux / # make dep
root@linux / # make clean
root@linux / # make bzImage
root@linux / # make modules
root@linux / # make modules_install
```

Beim Kernel **2.6** hat sich dies jedoch geändert:

```
root@linux / # make
root@linux / # make modules_install
```

2.6 Installation des Kernels

Nach dem Kompilieren (die Kompilierungszeit ist abhängig von der Rechnenleistung und kann mehrere Stunden oder auch nur einige Minuten dauern) kopieren Sie den Kernel in Ihre Boot-Partition bzw. Ihr /boot-Verzeichnis. Sichern Sie bitte vorher den funktionierenden Kernel! Eine Befehlsfolge könnte etwa wie folgt aussehen:

```
root@linux / # cp /boot/vmlinuz /boot/vmlinuz.old
root@linux / # cp /usr/src/linux/arch/i386/boot/bzImage /boot/vmlinuz
root@linux / # cp /usr/src/linux/System.map
/boot/System.map-<kernelversion>
```

Die exakte Versionsbezeichnung Ihres neu kompilierten Kernels teilt Ihnen die Datei /usr/src/linux/include/linux/version.h unter der Definition der **UTS-RELEASE** mit. Dies ist insbesondere dann wichtig, wenn Ihr Kernel eine besondere Namensgebung wie z.B. **2.6.0-64GB-SMP** aufweist. Programme, welche die Datei System.map benötigen, suchen nach einer Datei mit dem Namen System.map-<kernelversion> und verwenden dabei exakt den in der version.h Datei definierten Namen. Für Ihr laufendes System können Sie die richtige Versionsbezeichung mittels

```
root@linux / # uname -r
```

in Erfahrung bringen.

Um den neuen Kernel booten zu können, müssen Sie dem Bootloader noch mitteilen, wo er den neuen Kernel findet.

Öffnen Sie bei Verwendung von LILO die Datei /etc/lilo.conf mit einem Texteditor und fügen Sie Einträge der Form

/etc/lilo.conf

```
...
# Neuer Kernel:
image = /boot/vmlinuz
label = kernel_new

# Backup-Kernel:
image = /boot/vmlinuz.old
label = kernel_old
...
```

hinzu.

Nun führen Sie noch als **root** `/sbin/lilo` aus.

2.7 Starten des neuen Kernels

Nun können Sie ihren neuen Kernel booten. Geben Sie dazu als root

```
root@linux / # reboot
```

oder

```
root@linux / # shutdown -r now
```

ein.

Beim Erscheinen des Bootmenüs wählen Sie den Eintrag mit dem neuen Kernel aus. Damit bootet der Computer mit Ihrem neuen Kernel.

Sollte der Computer aus irgendwelchen Gründen nicht booten, so schauen Sie sich die Fehlermeldungen genau an, sie geben Aufschluß über die Ursachen. Starten Sie in diesem Fall Ihren Computer neu und booten Sie ihn dann mit dem alten Kernel. Sie finden die Fehlermeldungen, falls sie nicht ohnehin auf der Konsole ausgegeben wurden, in der Datei `/var/log/messages`.

Weitere Informationen zum Erstellen eines eigenen Kernels (in englischer Sprache) finden Sie auch hier:

 http://www.tldp.org/HOWTO/Kernel-HOWTO.html