

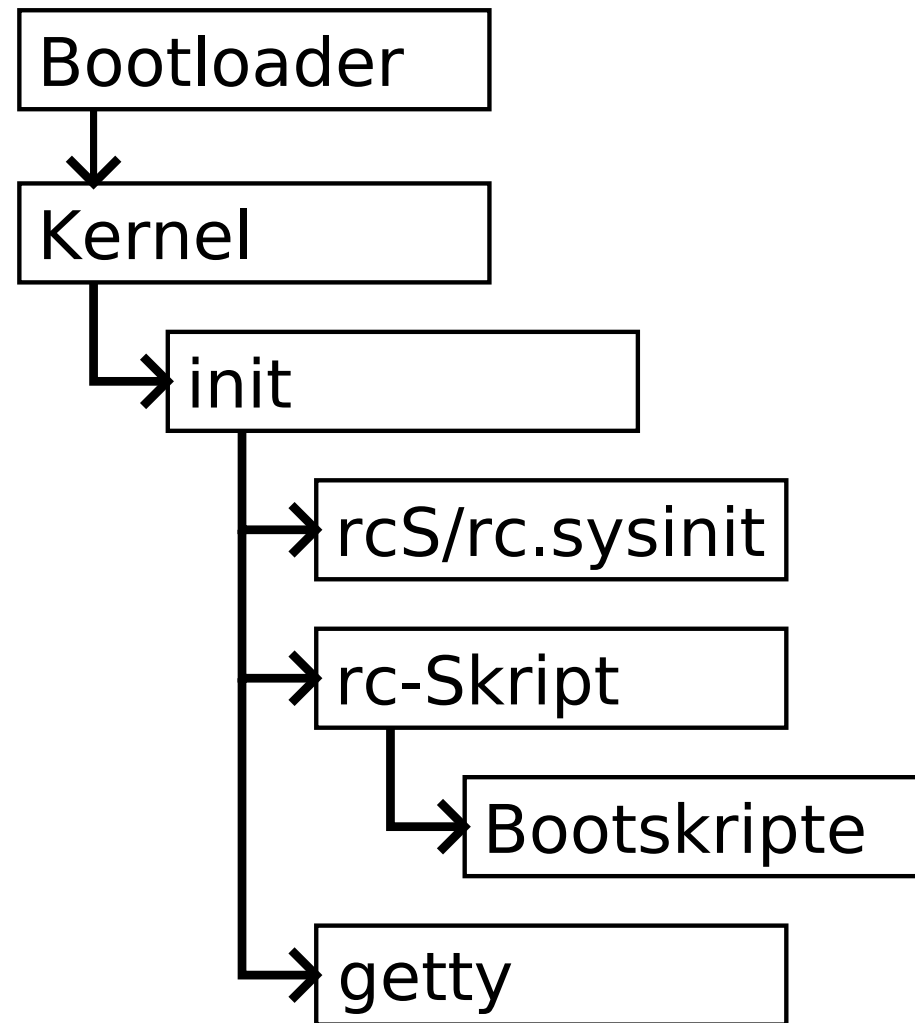
# System-V-Init und Bootvorgang

Jonathan Kleinhellefort

Dieses Dokument steht unter der GNU Free Documentation License (GFDL)

<http://www.fsf.org/licenses/fdl.html>

# Überblick über den Bootvorgang



# Der `init`-Prozess

## Bedeutung

- Der Start von `init` ist der letzte Schritt des Kernel-Bootvorgangs.
- `init` ist die Mutter aller Prozesse und besitzt die Prozess-ID 1.
- `init` startet die Bootskripte, die den weiteren Bootvorgang und die grundlegende Systemkonfiguration übernehmen.

## Konfiguration

Das Verhalten von `init` lässt sich weitgehend beeinflussen. Es wird mit Hilfe der Datei `/etc/inittab` konfiguriert, dies sollte normalerweise aber nicht nötig sein.

# Die Bootskripte

## Aufgaben

- Initialisierung von zusätzlicher Hardware (z.B. Tastatur, Soundkarte)
- Netzwerkkonfiguration (z.B. IP-Adressen, Firewall)
- Starten von System- und Netzwerk-Diensten (cron, inetd, sshd, ...)

## Ort

Die Init-Skripte befinden sich je bei den meisten Distribution in `/etc/init.d/`. Bei RedHat und auf RedHat-basierenden Distributionen in `/etc/rc.d/`.

## Benutzung

Führt man ein Init-Skript ohne Parameter aus, wird eine kurze Benutzungsinformation angezeigt:

```
> /etc/init.d/inetd
```

```
Usage: /etc/init.d/inetd {start|stop|reload|restart}
```

Laut **LSB** sind folgende Argumente gültig:

**start:** Dienst starten

**stop:** Dienst beenden

**restart:** Dienst neu starten

**reload:** die Konfiguration neu einlesen (ohne den Dienst neuzustarten)

**force-reload:** die Konfiguration neu laden, wenn nötig durch Neustart

**status:** den derzeitigen Zustand ausgeben

## Das Runlevel-Konzept

Ein **Runlevel** ist durch die Dinge (v.a. die Systemdienste), die zu Beginn und während des Runlevels laufen, gekennzeichnet. Welche genau dies sind kann in der `inittab` eingestellt werden.

Runlevel werden mit einer einzelne Ziffer oder einem Buchstaben benannt. Die wichtigsten sind die Runlevel **0 bis 6**.

## Wechsel des Runlevels

Mit dem Kommando `telinit level` kann man den aktuellen Runlevel ändern.

Um den vorherigen und den aktuellen Runlevel herauszufinden, dient der Befehl `runlevel`.

## Bedeutung der Runlevel bei verschiedenen Distributionen

	<b>LSB</b>	<b>Debian</b>	<b>SuSE</b>	<b>RedHat</b>
<b>0</b>	Halt	Halt	Halt	Halt
<b>1</b>	Single User	Single User	Single User	Single User
<b>2</b>	Multiuser ohne Netzwerk	Multiuser	Multiuser ohne Netzwerk	Multiuser ohne Netzwerk
<b>3</b>	Multiuser	"	Multiuser	Multiuser
<b>4</b>	reserviert	"	-	"
<b>5</b>	Multiuser mit XDM	"	Multiuser mit XDM	Multiuser mit XDM
<b>6</b>	Reboot	Reboot	Reboot	Reboot

## Das rc-Skript

Welches Kommando beim Eintritt in ein Runlevel ausgeführt wird, kann in der `/etc/inittab` eingestellt werden.

Bei einem System-V-konformen Bootmechanismus wird beim Wechsel des Runlevels das Skript `/etc/init.d/rc` (`/etc/rc.d/rc` bei RedHat) mit dem neuen Runlevel als Argument aufgerufen.

## Funktionsweise

rc ruft alle Skripte in einem für den jeweilige Runlevel eigenen Verzeichnis `rc?.d/` auf. Diese Verzeichnisse befinden sich bei jeder Distribution woanders:

<b>Debian</b>		<code>/etc/rc?.d/</code>
<b>SuSE</b>		<code>/etc/init.d/rc?.d/</code>
<b>RedHat</b>		<code>/etc/rc.d/rc?.d</code>

Zunächst werden diejenigen Skripte, die mit **K** beginnen, in alphabetischer Reihenfolge mit dem Argument **stop** aufgerufen, danach diejenigen, die mit **S** beginnen mit dem Argument **start**. In den Runleveln **0 und 6** werden alle Skripte mit **stop** aufgerufen.

Die Skripte sind nur **symbolische Links** auf Dateien in `/etc/init.d/` (bzw. `/etc/rc.d/` bei RedHat).

## Installation eines Init-Skripts

Das Skript sollte zunächst nach `/etc/init.d/` kopiert und ausführbar gemacht werden:

```
> cp proftpd /etc/init.d/  
> chmod 755 /etc/init.d/proftpd
```

Nun muss man noch für jeden Runlevel einen Link im entsprechenden Verzeichnis erstellen:

```
> for i in 3 4 5; do  
> ln -s ../init.d/proftpd /etc/rc$i.d/S20proftpd  
> done  
> for i in 0 1 2 6; do  
> ln -s ../init.d/proftpd /etc/rc$i.d/K20proftpd  
> done
```

**Achtung:** Dieses Beispiel funktioniert so nur unter Debian.

## Init-Skripte schreiben

Bootskripte können prinzipiell in jeder Programmiersprache geschrieben werden (im Grunde müssen es nicht einmal Skripte sein). Es bieten sich jedoch **Shell-Skripte** an.

Man sollte ein möglichst einfaches Skript als **Vorlage** nehmen. In manchen Distributionen existiert auch eines extra für diesen Zweck.

## Grundsätzlicher Aufbau

```
#!/bin/sh

case "$1" in
    start)
        [...]
        ;;
    stop)
        [...]
        ;;
    restart|force-reload)
        [...]
        ;;
    *)
        echo "Usage: $0 {start|stop|restart|force-reload}" >&2
        exit 1
        ;;
esac
```

## Distributionspezifische Besonderheiten

Nach der **LSB** sollte es die Datei `/lib/lsb/init-functions` geben, in der einige nützliche Hilfsfunktionen für Shell-Skripte definiert sind. Bei **RedHat** erfüllt `/etc/init.d/rc.d/functions` die gleiche Funktion.

Um diese nutzen zu können, muss man sie am Anfang des Skripts laden:

```
. /lib/lsb/init-functions
```

Bei jeder Distribution gibt es normalerweise Programme oder Funktionen, die beim Starten und Beenden von Daemons helfen:

<b>LSB</b>		<code>start_daemon/killproc</code> (Shell-Funktionen)
<b>Debian</b>		<code>start-stop-daemon</code>
<b>SuSE</b>		<code>startproc/killproc</code>
<b>RedHat</b>		<code>daemon/killproc</code> (Shell-Funktionen)

## Literatur

- Dr. Oliver Dierich: Unterbau, c't 17/2003, S.188ff
- Linux Standard Base (LSB): <http://www.linuxbase.org/>
- Manpages von `runlevel(8)`, `init(8)` und `inittab(5)`
- SysV-rc Dokumentation:  
/usr/share/doc/sysv-rc/README.runlevels (nur bei Debian)