SelfLinux-0.10.0

cron

Autor: JC PollmanChristian Richter (*jpollman@bigfoot.comcrichter@users.sourceforge.net*)
Formatierung: Matthias Hagedorn (*matthias.hagedorn@selflinux.org*)

Linux bietet zwei verwandte Programme für das Automatisieren von Aufgaben: `cron` und `at`. Beide starten beim Booten und laufen als Daemons, so werden sie niemals beendet.

`Cron` ist für sich wiederholende Aufgaben zuständig, `at` für einmalig ablaufende.

Inhaltsverzeichnis

1 Der cron Batchdaemon

Linux bietet etliche Programme für das Automatisieren von Aufgaben. Ein Beispiel ist `cron`. Der Batchdaemon `cron` ist für Aufgaben zuständig, die automatisch zu festgelegten Zeiten stattfinden sollen. Er wird beim Booten gestartet und im Normalbetrieb nicht beendet. Ein Daemon ist ein Prozess, der im Hintergrund des Systems läuft und von dem man normalerweise keine Ausgabe sieht.

`cron` liest eine sogenannte crontab-Datei, um Informationen für seine Arbeit zu erhalten. Es gibt eine systemweite und eventuell einige benutzerspezifische crontab-Dateien. Die Datei des Systems befindet sich unter /etc/crontab. Verändern Sie diese Datei auf keinen Fall, wenn Sie nicht genau wissen, was Sie tun. Selbst der Benutzer root sollte seine eigene Datei erzeugen.

2 Erzeugen der crontab-Datei

Prüfen Sie zunächst, ob für den Benutzer, dessen crontab-Datei Sie erzeugen möchten (in unserem Fall root), noch keine solche Datei existiert. Dies können Sie mittels

```
root@linux ~/ # crontab -l
```

tun. Es sollte die Ausgabe **no crontab for root** erscheinen. Andernfalls würde das folgende Kommando die alte crontab-Datei überschreiben. Geben Sie dann Folgendes ein:

```
root@linux ~/ # crontab /etc/crontab
```

Das Kommando wird eine crontab-Datei für Sie erzeugen, die eine Kopie der systemweiten crontab-Datei ist. Editieren Sie nun ihre Datei mittels

```
root@linux ~/ # crontab -e
```

Beachten Sie, dass `crontab` sowohl der Name der Datei als auch der Name des ausführbaren Programmes ist, ähnlich wie im Falle des Kommandos passwd. Die erzeugte Datei wird in etwa wie diese Konfiguration aussehen (dies ist ein Beispiel einer RedHat-Datei):

Konfiguration RedHat-Datei

```
SHELL=/bin/bash
PATH=/sbin:/bin/:/usr/sbin/:/usr/bin
MAILTO=root
HOME=/

#run-parts
01 * * * * root run-parts /etc/cron.hourly
01 4 * * * root run-parts /etc/cron.daily
22 4 * * 0 root run-parts /etc/cron.weekly
42 4 1 * * root run-parts /etc/cron.monthly
```

Löschen Sie alle Zeilen nach der Zeile `HOME=/`. Dies sind nicht die Aktionen die Sie ausführen wollen. Ändern Sie außerdem die Variable PATH so, dass sie das Verzeichnis enthält, in dem Sie Ihre Skripte oder Programme

speichern, die zeitgesteuert ablaufen sollen.

3 cron Konfiguration

Jede Zeile in der crontab-Datei führt ein eigenes Programm aus. Die Zeilen haben ein spezielles Format:

Fünf Zeitfelder, gefolgt von dem auszuführenden Programm. Beachten Sie: In der systemweiten crontab-Datei ist ein weiteres Feld, das den Daemon anweist, das Programm als einen bestimmten Benutzer auszuführen (z. B. root). In einer benutzerspezifischen crontab-Datei wird dieses Feld ignoriert.

Die fünf Zeitfelder sind:

Minuten Stunden Tag-des-Monats Monat Wochentag

Auszug aus der Manpage

```
Die Zeit- und Datumsfelder sind:

Minute          0-59
Stunde          0-23
Tag-des-Monats  1-31
Monat           1-12 (oder Namen, siehe unten)
Wochentag       0-7 (0 oder 7 ist Sonntag oder Namen)

Ein Feld kann ein Stern (*) sein, was immer für
"Erster-Letzter" steht.

Zahlenbereiche sind erlaubt. Bereiche sind zwei Zahlen,
getrennt durch einen Bindestrich. Die angegebenen Grenzen sind
inklusive.
Beispielsweise: 8-11 in "Stunde" bewirkt die Ausführung um 8,
9, 10, 11 Uhr.

Listen sind erlaubt. Eine Liste ist eine Menge von Nummern
(oder Bereichen), getrennt durch Kommata.
Beispiele: "1,2,5,9", "0-4,8-2". (Die Hochkommata nicht mit
in die Datei übernehmen, Anmerkung des Übersetzers)

Schrittweiten können in Verbindung mit Bereichen genutzt
werden. Hinter einem Bereich mit "/<Schrittweite>" angegeben,
bestimmt die Schrittweite, ob Werte innerhalb des Bereiches
übersprungen werden.
Beispiel: "0-23/2" kann unter Stunden benutzt werden, um ein
spezielles Kommando alle zwei Stunden auszuführen. Die
Alternative wäre:
"0,2,4,6,8,10,12,14,16,18,20,22". Schrittweiten sind auch nach
Sternen (*) erlaubt, "alle zwei Stunden" lässt sich auch durch
"*/2" beschreiben.

Namen können für "Monat" und "Wochentag" benutzt
werden. Benutzen Sie die ersten drei Buchstaben des
entsprechenden Tages oder Monats (Groß-/Kleinschreibung wird
nicht beachtet). Bereiche oder Listen sind mit Namen nicht
erlaubt.
```

4 Ein praktisches Beispiel

Für ein Backup-Skript zur Datensicherung, das um 5 Minuten nach 1 Uhr morgens jeden Tag laufen soll, würde die zugehörige crontab-Datei folgendermaßen aussehen:

Backup-Skript

```
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin:/usr/local/bin
MAILTO=root
HOME=/

5 1 * * * /usr/local/bin/run-backup
```

Durch die Angabe `MAILTO=root` teilt `cron` die Beendigung der Aufgabe dem Benutzer **root** per E-Mail mit. Ist man mit dem Ablauf des Skriptes run-backup zufrieden und der Benachrichtigungen überdrüssig, ändern man die `MAILTO`-Zeile folgendermaßen:

run-backup

```
MAILTO=""
```

5 Mehr Information

Weitere Information findet sich in folgenden Manual-Seiten:

* man crontab
* man 5 crontab
* man cron