SelfLinux-0.13.0

Allgemeine Kommandosyntax

Autor: Matthias Kleine (*kleine_matthias@gmx.de*)
Formatierung: Johnny Graber (*selflinux@jgraber.ch*)
Lizenz: GPL

1 Allgemeine Kommandosyntax

1.1 Vorbemerkung

Möglicherweise würden Sie dieses Kapitel gerne überspringen und lieber gleich zur Mausbenutzung übergehen. Das Absetzen eines Kommandos hat in der heutigen Fensterwelt etwas Archaisches an sich und wirkt komplizierter als die Verwendung von Popup-Menüs, Registerkarten und Dialogboxen. Tatsächlich können Sie auch unter Linux heute praktisch alle wichtigen Benutzeraufgaben erledigen, ohne je ein Kommando absetzen zu müssen. Es sei jedoch hinzugefügt, dass mit dem höheren Komfort ein verminderter Fahrspaß verbunden ist.

Unix-Systeme sind wie Baukästen. Es steht Ihnen eine Unzahl kleiner Programme zur Verfügung, die Sie für die verschiedensten Aufgaben miteinander kombinieren können. Natürlich können Sie, wenn Sie wollen, immer nur die größten Klötze verwenden und die kleinen im Kasten liegen lassen. Manchmal können jedoch die feineren Handgriffe darüber entscheiden, ob Sie Ihre Aufgabe mit einigen wenigen oder mit einer langen Reihe von Arbeitsschritten bewältigen müssen. Und Sie können sich darauf verlassen, dass Linux Sie bei der Verwendung feinerer Handgriffe in jeder Hinsicht so gut wie möglich unterstützt.

Üblicherweise werden Kommandos in einer Shell abgesetzt. Die Shell - z.B. die Bourne again Shell oder kurz bash - nimmt Ihre Kommandozeile entgegen, bearbeitet den entgegengenommenen Text und leitet schließlich die Ausführung des gerufenen Programmes ein. Kommandos sind übrigens nichts anderes als Programme. Sie sind nur meist in einer Shell anzutreffen und werden daher begrifflich gelegentlich voneinander unterschieden.

1.2 Die Eingabe eines Kommandos

Wenn Sie sich über ein tty anmelden, startet sofort eine Shell und ermöglicht die Eingabe von Kommandos am Prompt. Sie können auch unter X Windows eine Shell öffnen und darin Kommandos aufrufen. Dazu benutzen Sie sogenannte Terminalemulationen wie `xterm` oder `kvt`. Diese emulieren eine Terminalsituation inklusive der Standarddatenströme von der Tastatur und zum Monitor - allerdings ohne dafür jeweils ein tty zu benutzen. Es handelt sich einfach um Programme, die in ihrem Fenster eine Shell beherbergen.

Die Eingabe eines Kommandos erfolgt über den Kommandonamen. Dieser wird mit Enter bestätigt und damit der Shell zur Bearbeitung übergeben. Im einfachsten Fall hat die Shell nichts weiter zu tun als das jeweilige Programm aufzurufen und diesem die Kontrolle zu übergeben. Das Programm tut seinen Dienst, wird irgendwann beendet und liefert seinen Rückgabewert zurück an die Shell. Diese ist somit informiert, dass das gestartete Programm beendet ist, und gibt wieder ihren Prompt aus, um auf das nächste Kommando zu warten. Dies ist die einfachste Form einer Kommandoeingabe. Wir wollen uns aber noch einige weitere anschauen.

1.3 Verschiedene Formen von Kommandos

In vielen Fällen müssen einem Kommando weitere Informationen übergeben werden, damit es seine Arbeit verrichten kann. Es gibt grundsätzlich zwei Arten von Zusatzinformationen, die man Kommandos mitteilen kann: Optionen und Argumente. Dabei werden die Optionen immer vor den Argumenten angegeben, so dass die grundlegende Syntax aller Linux-Kommandos folgendermaßen notiert werden kann:

```
user@linux / $ kommandoname [-Optionen] [Argumente]
```

Die eckigen Klammern zeigen an, dass Optionen und Argumente optional, also nicht notwendig sind. Ihre Angabe hängt von den Absichten des Aufrufers und der Liste möglicher Parameter eines Kommandos ab.

1.3.1 Optionen

Durch Optionen können Sie das Verhalten eines Kommandos beeinflussen. Optionen werden gewöhnlich durch einzelne Buchstaben bezeichnet und beginnen mit einem vorangestellten Minus `-`. Das Kommando `ls` beispielsweise gibt gewöhnlich den Inhalt eines Verzeichnisses aus: Es listet einfach die Namen der enthaltenen Unterverzeichnisse und Dateien auf. Will man jedoch nicht einfach nur die Namen wissen, sondern auch Zusatzinformationen über Dateigröße, Erstellungsdatum und vieles andere, so muss man dies dem `ls`-Kommando mitteilen. Die übliche Eingabe in einem solchen Fall würde lauten:

```
user@linux / $ ls -l
```

`-l` (l für »long«) ist eine Option und veranlasst eine ausführlichere Ausgabe. Das Verhalten des Kommandos hat sich durch die Verwendung der Option verändert. Optionen können miteinander kombiniert werden, indem man weitere Zeichen einfach hinzufügt. Das Minuszeichen muss also nur ein einziges Mal verwendet werden, um damit anzuzeigen, dass nun eine Reihe von Optionen folgt. In unserem Kapitel über die Shell werden wir noch genauer auf die Verwendung von Optionen eingehen. Eine Übersicht möglicher Optionen eines Befehls gibt die Manpage des Kommandos. Diese Hilfetexte zu Shell-Kommandos erreicht man über die Eingabe von:

```
user@linux / $ man Kommandoname
```

am Prompt der Shell.

1.3.2 Argumente

Argumente dienen nicht zur Steuerung eines Kommandos, sondern liefern diesem Informationen, die es zu bearbeiten hat. Viele Kommandos zur Manipulation von Dateien benötigen zum Beispiel die Namen der Dateien, die sie manipulieren sollen. Hier wird also nicht das Verhalten des Programmes geändert, sondern die Information variiert, die dem Programm für seine Arbeit zur Verfügung steht. Im Gegensatz zu Optionen kann es häufig eine sehr große Zahl verschiedener Argumente geben. Optionen hingegen sind immer nur in relativ beschränkter Zahl verfügbar - immer gerade so viele, wie der Programmierer in sein Programm implementiert hat. Nebenbei bemerkt ist jedoch auch die Anzahl der Argumente einer Kommandozeile nicht unbeschränkt, denn die Argumentzeile eines Kommandos darf eine Größe von 128 Kilobyte nicht überschreiten.

1.3.3 Optionen, die Argumente erwarten

Manche Optionen erwarten ihrerseits Argumente. Schauen wir uns beispielsweise folgenden Aufruf eines C-Compilers an:

```
user@linux / $ gcc -Wall prog.c
```

`gcc` ist der Name des Kommandos. Die einzige Option in dieser Zeile ist `-W`. Sie kann mit Argumenten versorgt werden, hier ist das angegebene Argument `all`. Ein Leerzeichen ist nicht notwendig, aber möglich. Das letzte Argument `prog.c` gehört nicht mehr zur Option `-W`, sondern bezeichnet den Dateinamen des Quelltextes, der kompiliert werden soll.

1.3.4 Lange Optionen

In der Linux-Welt hat sich eine weitere Art von Optionen verbreitet, die sich durch eine besondere Schreibweise

auszeichnet: die langen oder GNU-Optionen. Sie beginnen mit einem doppelten Minuszeichen `--`, gefolgt von der eigentlichen Option, die meist ein ausgeschriebenes Wort ist. Lange Optionen sind somit "sprechender" als kurze. Allerdings wird die Verwendung mehrerer Optionen auch unübersichtlicher. Ein Beispiel für eine weit verbreitete lange Option ist `--version`. Viele GNU-Kommandos geben bei einem Aufruf mit dieser Option ihre Versionsnummer aus.

1.4 Die Rolle der Shell

Sie wissen jetzt, dass Sie Kommandos über eine Shell aufrufen und ihnen Optionen und Argumente übergeben können. Zum Abschluss möchten wir Ihr Bewusstsein dafür schärfen, dass damit jedoch lediglich die Rahmenbedingungen für eine Kommandoeingabe dargestellt sind, wie sie die Kommandos selbst bieten. Die Gemeinschaft der Programmierer hat sich gewissermaßen darauf geeinigt, dass Kommandos so und nicht anders zu arbeiten haben. Die grundlegende Syntax eines Kommandos ist von der Shell unabhängig.

Umgekehrt ist es aber auch so, dass die Syntax der Shell selbst von dem jeweils verwendeten Kommando unabhängig ist. Tatsächlich ist die Form einer Kommandozeile nicht nur durch die Syntax des aufgerufenen Kommandos, sondern auch durch die Syntax der Shell bestimmt. Bevor die Shell nämlich ein Kommando zur Ausführung bringt, nimmt sie unter Umständen im Rahmen des Parsens eine Reihe von Veränderungen an der Eingabe vor. Diese Veränderungen haben nichts mit dem aufgerufenen Kommando zu tun, sondern werden von der Shell nach den immer gleichen syntaktischen Regeln vorgenommen. Die Shell prüft auf diese Weise nicht nur, ob der Kommandoaufruf ihren syntaktischen Regeln entspricht, sondern sie ermöglicht Ihnen damit auch zahlreiche Vereinfachungen bei der Kommandoeingabe.