



Michael Kofler

Linux

Kommando-Referenz

Shell-Befehle von a2ps bis zypper



ADDISON-WESLEY

-1 bis -9

gibt an, wie viel Speicherplatz (RAM) der Komprimieralgorithmus nutzen darf. Die Grundeinstellung lautet -9 und liefert die besten Ergebnisse. Wenn nur wenig RAM zur Verfügung steht, sollten Sie einen kleineren Wert wählen; allerdings wird dann auch die Komprimierung etwas schlechter.

```
case ausdruck in                                     (bash)
  muster1 ) kommandos;;
  muster2 ) kommandos;;
  ...
esac
```

case bildet in bash-Scripts Mehrfachverzweigungen, wobei als Kriterium für die Verzweigung eine Zeichenkette angegeben wird (zumeist eine Variable oder ein Parameter, der dem Shell-Programm übergeben wird). Diese Zeichenkette wird der Reihe nach mit den Mustern verglichen, wobei in diesen Mustern die Jokerzeichen für Dateinamen (*?[]) verwendet werden können. In einem case-Zweig können auch mehrere durch | getrennte Muster angegeben werden. Sobald ein Muster zutrifft, werden die Kommandos ausgeführt, die zwischen der runden Klammer) und den beiden Strichpunkten folgen. Anschließend wird das Programm nach esac fortgesetzt.

cat dateien

cat zeigt den Inhalt der angegebenen Textdatei an. Bei längeren Texten sollten Sie statt cat das Kommando less verwenden; damit können Sie zeilen- bzw. seitenweise durch den Text blättern. cat wird häufig auch dazu verwendet, mehrere Dateien zu einer größeren Datei zusammenzusetzen. Dazu muss die Standardausgabe mit > in eine Datei umgeleitet werden (siehe Beispiel). Zu cat existiert auch die Variante tac. Dieses Kommando gibt die Zeilen der Textdatei in umgekehrter Reihenfolge aus (die letzte Zeile zuerst).

- s reduziert mehrere leere Zeilen auf eine einzige leere Zeile.
- T zeigt Tabulatorzeichen in der Form ^I an.
- v zeigt nicht druckbare Zeichen in der ^xxx-Schreibweise an.

Beispiele

Das folgende Kommando setzt die Einzeldateien teil1.tex, teil2.tex etc. zu einer Gesamtdatei total.tex zusammen. Die Einzeldateien werden in alphabetischer Reihenfolge verarbeitet.

```
user$ cat teil*.tex > total.tex
```

Im folgenden Kommando wird die Standardeingabe in eine neue Datei umgeleitet. Nachdem Sie das Kommando mit `↵` bestätigt haben, werden alle weiteren Eingaben in die neue Datei geschrieben. `Strg+D` beendet die Eingabe. In dieser Form kann `cat` also dazu verwendet werden, um ohne einen Editor eine neue Textdatei zu erzeugen.

```
root# cat > neuedatei
Zeile 1
Zeile 2 <Strg>+<D>
```

cd [verzeichnis] (bash)

`cd` wechselt in das angegebene (Unter-)Verzeichnis. Wenn kein Verzeichnis angegeben wird, wechselt `cd` in das Heimatverzeichnis. Wenn als Verzeichnis `-` angegeben wird, wechselt `cd` in das zuletzt gültige Verzeichnis. `pwd` zeigt den Pfad des gerade aktuellen Verzeichnisses an.

cdrdao kommando [optionen] toc-datei

`cdrdao` schreibt eine CD im Disk-at-once-Modus (DAO). In der Praxis besteht die gebräuchlichste Anwendung von `cdrdao` darin, Audio-CDs zu kopieren. Das erste `cdrdao`-Kommando erzeugt die Dateien `data.bin` (Inhalt der CD) und `data.toc` (Inhaltsverzeichnis). Das zweite Kommando schreibt diese Daten auf eine CD.

```
user$ cdrdao read-cd --device /dev/sg0 data.toc
user$ cdrdao write --device /dev/sg0 --buffers 64 data.toc
```

chac1 [optionen] dateien

`chac1` ermittelt bzw. verändert die erweiterten Zugriffsrechte der angegebenen Dateien bzw. Verzeichnisse. Das funktioniert nur, wenn das Dateisystem ACLs (*Access Control Lists*) unterstützt. Bei `ext3/ext4`-Dateisystemen muss dazu die `mount-Option` `ac1` verwendet werden.

Statt `chac1` sollten Sie nach Möglichkeit `getfac1` bzw. `setfac1` einsetzen. `chac1` steht nur aus Kompatibilitätsgründen zu IRIX SGI zur Verfügung.

chattr [optionen] +/-=[ASacDdIijsTtu] dateien

In den Dateisystemen `ext2`, `ext3` und `ext4` können mit jeder Datei neben den Benutzerinformationen (siehe `chmod` und `chown`) einige zusätzliche Attribute gespeichert werden. Diese Attribute enthalten beispielsweise Informationen über den Journaling-Status oder über eine eventuelle Komprimierung der Datei. Eine kurze Beschreibung der Attribute gibt man `chattr`. Allerdings werden momentan nur wenige der vorgesehenen Attribute tatsächlich genutzt.

chgrp [optionen] gruppe dateien

chgrp ändert die Gruppenzugehörigkeit von Dateien. Der Besitzer einer Datei kann diese Datei nur seinen eigenen Gruppen zuordnen. root kann beliebige Zuordnungen treffen.

-R bzw. --recursive

verändert auch die Gruppenzuordnung von Dateien in allen Unterverzeichnissen. Die Option ist nur dann sinnvoll, wenn die Dateien durch Jokerzeichen beschrieben werden (etwa *.tex).

chkconfig optionen

chkconfig ist ein Red-Hat- bzw. Fedora-spezifisches Kommando zum Einrichten bzw. Löschen von Init-V-Runlevel-Links. Sofern xined installiert ist, können auch dessen Dienste mit chkconfig gesteuert werden. Das Kommando steht aus Kompatibilitätsgründen vereinzelt auch für andere Distributionen in gleicher oder ähnlicher Syntax zur Verfügung. (Werfen Sie aber unbedingt einen Blick in die man-Seiten!)

--list

liefert eine Liste aller installierten Init-V-Scripts sowie Informationen darüber, in welchen Runleveln die Scripts gestartet werden.

--add name

richtet Links auf das Init-V-Script in den dafür vorgesehenen Standard-Runleveln ein. (Diese Information stammt aus den Kommentarzeilen am Beginn des Scripts.) Die neuen Links werden erst beim nächsten Runlevel-Wechsel bzw. Neustart berücksichtigt. Das Init-V-Script wird also nicht gestartet.

--del name

entfernt alle Init-V-Links für das Script.

--level n name on|off

richtet Init-V-Links nur für die angegebenen Runlevel ein bzw. entfernt sie.

Beispiel

Das folgende Kommando richtet Start-Links für den Webserver Apache (Dämon httpd) in den Runleveln 3 und 5 ein:

```
root# chkconfig --level 35 httpd on
```

Dabei werden diese zwei Links erzeugt:

```
/etc/rc3.d/S85httpd -> ../init.d/httpd
/etc/rc5.d/S85httpd -> ../init.d/httpd
```

chmod [optionen] änderungen dateien

chmod ändert die neun Zugriffsbits von Dateien. Zusammen mit jeder Datei wird gespeichert, ob der Besitzer (*user*), die Gruppenmitglieder (*group*) und andere Benutzer (*others*) die Datei lesen, schreiben und ausführen dürfen. Die Änderung der Zugriffsbits erfolgt durch die Zeichenkombination *Gruppe +/- Zugriffstyp*, also beispielsweise *g+w*, um allen Gruppenmitgliedern eine Schreiberlaubnis zu geben. Die Gruppe geben Sie durch *u* (*user*), *g* (*group*), *o* (*others*) oder *a* (*all*) an, den Zugriffstyp durch *r* (*read*), *w* (*write*) oder *x* (*execute*).

Setuid, setgid und sticky-Bits

Das *setuid*-Bit (oft auch *suid*-Bit genannt) bewirkt, dass Programme so ausgeführt werden, als hätte der Besitzer selbst das Programm gestartet. Wenn der Besitzer eines Programms *root* ist, dann kann jeder das Programm ausführen, als wäre er selbst *root*.

Das *setgid*-Bit hat bei Programmen dieselbe Funktion wie *setuid*, aber eben für die Gruppenzugehörigkeit. Bei Verzeichnissen bewirkt das *setgid*-Bit, dass in diesem Verzeichnis neu erzeugte Dateien der Gruppe des Verzeichnisses angehören (und nicht, wie sonst üblich, der Gruppe des Benutzers, der die Datei erzeugt).

Das *sticky*-Bit bewirkt bei Verzeichnissen, in denen alle die Dateien ändern dürfen, dass jeder nur seine eigenen Dateien löschen darf (und nicht auch Dateien anderer Benutzer). Das Bit ist beispielsweise beim */tmp*-Verzeichnis gesetzt. In diesem Verzeichnis darf jeder Benutzer temporäre Dateien anlegen. Es muss aber vermieden werden, dass auch jeder Benutzer nach Belieben fremde Dateien umbenennen oder löschen kann.

Um mit *chmod* die Spezialbits *setuid*, *setgid* und *sticky* zu setzen, sind die folgenden Zeichenkombinationen vorgesehen:

```
setuid: u+s
setgid: g+s
sticky: +t
```

Damit *setuid* wirkt, muss auch das *x*-Bit für den Besitzer gesetzt sein (*u+x*).

Damit *setgid* wirkt, muss auch das *x*-Bit für die Gruppe gesetzt sein (*g+x*).

Oktale Schreibweise

Statt mit Buchstaben kann der Zugriffstyp auch durch eine maximal vierstellige Oktalzahl angegeben werden. Bei den Zugriffsbits ist *u*, *g* und *o* jeweils eine Ziffer zugeordnet. Jede Ziffer ist aus den Werten 4, 2 und 1 für *r*, *w* und *x* zusammengesetzt. 660 bedeutet daher *rw-rw----*, 777 steht für *rw-rwxrwx*. Die drei Spezialbits *setuid*, *setgid* und *sticky* haben die Oktalwerte 4000, 2000 und 1000.

-R bzw. *--recursive*

verändert auch die Zugriffsrechte von Dateien in allen Unterverzeichnissen.

Beispiele

Die Datei `sichere` kann nun von allen Benutzern ausgeführt werden. `sichere` kann etwa ein Shell-Script zur Erstellung eines Backups sein.

```
user$ chmod a+rx sichere
```

Das folgende Kommando entzieht allen Benutzern außerhalb der eigenen Gruppe die Lese- und Schreiberlaubnis für alle `*.doc`-Dateien im aktuellen Verzeichnis:

```
user$ chmod o-rw *.doc
```

```
chown [optionen] user[:gruppe] dateien
```

`chown` ändert den Besitzer und (optional) auch die Gruppenzugehörigkeit einer Datei. Der Besitzer einer Datei kann nur von `root` verändert werden, während die Gruppe auch von anderen Benutzern eingestellt werden kann (siehe `chgrp`).

`-R` bzw. `--recursive`

verändert auch die Gruppenzuordnung von Dateien in allen Unterverzeichnissen.

Beispiel

Das folgende Kommando stellt sicher, dass alle Dateien innerhalb von `/var/www` dem Benutzer und der Gruppe `www-data` zugeordnet sind. (Bei Debian- und Ubuntu-Systemen läuft Apache unter dem Account `www-data`.)

```
root# chown -R www-data:www-data /var/www
```

```
chroot verzeichnis [kommando]
```

Ohne weitere Parameter startet `chroot` eine neue Shell, die das angegebene Verzeichnis als Wurzelverzeichnis `/` verwendet. In dieser Shell können Sie interaktiv arbeiten. `exit` führt zurück in die ursprüngliche Shell.

Wenn Sie optional ein Kommando angeben, wird dieses Kommando statt der Shell gestartet. Während der Ausführung des Kommandos gilt abermals das angegebene Verzeichnis als Wurzelverzeichnis.

```
chsh [user] shell
```

`chsh` verändert die Standard-Shell, die automatisch nach dem Einloggen aufgerufen wird. Zur Auswahl stehen alle in `/etc/shells` eingetragenen Shells, normalerweise `/bin/bash`, `/bin/csh` und `/bin/ksh`. Das Kommando `chsh` verändert die Datei `/etc/passwd` und trägt dort die neue Shell ein. Die Shell eines anderen Anwenders kann nur von `root` verändert werden

(während jeder Anwender seine eigene Shell nach Belieben verändern kann). Die neue Shell muss mit dem vollständigen Verzeichnis angegeben werden.

cksum datei

cksum ermittelt die Prüfsumme und die Länge der Datei in Bytes. Die Prüfsumme kann verwendet werden, um rasch festzustellen, ob zwei Dateien identisch sind. cksum liefert zuverlässigere Ergebnisse als das verwandte Kommando sum. Mathematisch noch sicherer sind md5sum oder sha512sum.

clear

clear bzw. `[Strg]+[L]` löschen den Inhalt der Konsole.

cmp [optionen] datei1 datei2

cmp vergleicht zwei Dateien Byte für Byte und liefert die Position der ersten Abweichung. Wenn die Dateien identisch sind, zeigt das Kommando überhaupt keine Meldung an (siehe auch diff auf Seite 39).

- c bzw. --show-chars
zeigt das jeweils erste Textzeichen an, bei dem sich die Dateien voneinander unterscheiden.
- l bzw. --verbose
liefert eine Liste aller Abweichungen.

compress [optionen] datei

compress komprimiert bzw. dekomprimiert die angegebene Datei. Bei komprimierten Dateien wird die Kennung .Z an den Dateinamen angehängt. compress existiert nur noch aus Kompatibilitätsgründen. Wesentlich leistungsfähiger sind bzip2 (siehe Seite 23) und gzip (Seite 67).

continue [n]

(bash)

continue überspringt in bash-Scripts den Körper einer for-, while- oder until-Schleife und setzt die Schleife mit dem nächsten Durchlauf fort. Durch den optionalen Zahlenwert kann dieser Vorgang auch für äußere Schleifenebenen durchgeführt werden.

convert [optionen] bildalt bildneu

convert aus dem Image-Magick-Paket konvertiert Bilddateien von einem Format in ein anderes. In der einfachsten Form wird es in der Art `convert name.tif name.jpg` aufgerufen, um die angegebene TIF-Datei in eine JPEG-Datei zu konvertieren. Die ursprüngliche Datei bleibt

dabei erhalten. Mit über 100 Optionen können gleichzeitig diverse Bildparameter verändert werden. Die folgende Liste ist daher nur eine Auswahl.

- blur *radius*
verwischt das Bild.
- colors *n*
reduziert die Anzahl der RGB-Farben auf *n*.
- colorspace CMYK|GRAY|RGB|Transparent|YUV
gibt das gewünschte Farbmodell an (wobei zahlreiche weitere Modelle zur Auswahl stehen).
- compress None|BZip|Fax|Group4|JPEG|JPEG2000|Lossless|LZW|RLE|Zip
gibt das gewünschte Kompressionsformat an. Welche Formate tatsächlich zur Auswahl stehen, hängt allerdings vom Bildformat ab.
- contrast bzw. +contrast
verringert bzw. vergrößert den Kontrast des Bilds.
- crop *geometry*
schneidet den gewünschten Teil des Bilds aus. Beispielsweise beschreibt -crop 50x50+100+100 ein 50 mal 50 Pixel großes Gebiet, das an der Koordinatenposition (100, 100) beginnt.
- filter Point|Box|Triangle|Hermite|Hanning|Hamming ...
wendet den gewünschten Filter auf das Bild an.
- gaussian *radius*
verwischt den Filter mit dem Gauß-Operator.
- normalize
normalisiert die Farbverteilung im Bild.
- quality *n*
gibt die gewünschte Kompressionsqualität an. Die zulässigen Werte für *n* hängen vom Bildformat ab (z. B. 0 bis 100 bei JPEG).
- resize *NxN* bzw. -resize *n%*
verändert die Auflösung des Bildes.
- rotate *winkel*
dreht das Bild im Uhrzeigersinn um den angegebenen Winkel in Grad.
- trim
schneidet einfarbige Bildränder ab, wenn sie dieselbe Farbe wie die Eckpunkte des Bilds haben.

Beispiel

Die drei folgenden Beispiele zeigen mögliche Anwendungen des Kommandos:

```

user$ convert -resize 100x100 bild.jpg bild.png
user$ convert -type Grayscale bild.jpg bild.eps
user$ convert -quality 80 bild.bmp bild.jpg

```

convmv [optionen] dateien

Das Perl-Script convmv von der Website <http://j3e.de/linux/convmv/> bzw. aus dem Paket convmv ändert den Zeichensatz der angegebenen Dateinamen. Das Programm ist eine große Hilfe, wenn nach einer Zeichensatzumstellung Dateinamen falsch dargestellt werden. (convmv verändert nur den Namen, nicht den Inhalt der Dateien!)

-f *zeichensatz*

gibt den ursprünglichen Zeichensatz der Dateinamen an. (convmv --list liefert eine Liste aller unterstützten Zeichensätze.)

-t *zeichensatz*

gibt den neuen Zeichensatz an.

--notest

ändert die Dateinamen tatsächlich. Ohne diese Option zeigt convmv lediglich die geplanten Änderungen an, ohne diese aber tatsächlich durchzuführen. Wenn Sie convmv rekursiv ausführen, ist ein Testlauf (also ohne --notest) unbedingt empfehlenswert!

-i ändert Dateinamen erst nach einer Rückfrage. Die Option ist nur in Kombination mit --notest zweckmäßig.

-r wendet das Kommando rekursiv auf Unterverzeichnisse an.

Beispiel

Um rekursiv alle Dateien eines Verzeichnisses vom Zeichensatz Latin-1 auf UTF-8 umzustellen (mit Rückfrage für jede einzelne Änderung), rufen Sie convmv so auf:

```

user$ convmv -r -i --notest -f iso-8859-1 -t utf8 verzeichnisname

```

cp [optionen] quelle ziel

cp [optionen] dateien zielverzeichnis

cp kopiert Dateien und Verzeichnisse. Einzelne Dateien können beim Kopieren umbenannt werden. Bei der Bearbeitung mehrerer Dateien (z. B. durch die Angabe von Jokerzeichen) können diese lediglich in ein anderes Verzeichnis kopiert, nicht aber umbenannt werden. Anweisungen der Art cp *.tex *.bak sind nicht zulässig. Mit cp vergleichbare Kommandos sind mv zum Verschieben und Umbenennen von Dateien sowie ln zur Herstellung von Links. cp unterstützt unter anderem folgende Optionen:

- a bzw. --archive
behält möglichst alle Attribute der Dateien bei. -a ist eine Abkürzung für -dpR.
- b bzw. --backup
benennt bereits vorhandene Dateien in Backup-Dateien um (Dateiname plus ~), anstatt sie zu überschreiben.
- d bzw. --dereference
kopiert bei Links nur den Verweis, nicht aber die Datei, auf die der Link zeigt.
- i bzw. --interactive
fragt, bevor vorhandene Dateien überschrieben werden.
- l bzw. --link
erstellt feste Links (*Hard Links*), anstatt die Dateien zu kopieren. Wenn cp mit dieser Option verwendet wird, hat es dieselbe Funktionalität wie ln (siehe Seite 80).
- p bzw. --preserve
lässt die Informationen über den Besitzer, die Gruppenzugehörigkeit, die Zugriffsrechte und den Zeitpunkt der letzten Änderung unverändert. Ohne diese Option gehört die Kopie demjenigen, der cp ausführt (Benutzer und Gruppe), und die Zeitangabe wird auf die aktuelle Zeit gesetzt.
- r bzw. -R bzw. --recursive
kopiert auch Unterverzeichnisse und die darin enthaltenen Dateien.
- s bzw. --symbolic-link
erstellt symbolische Links, anstatt die Dateien oder Verzeichnisse zu kopieren. cp hat damit die Funktionalität von ln -s (siehe Seite 80).
- u bzw. --update
kopiert Dateien nur dann, wenn dabei keine gleichnamige Datei mit neuerem Datum überschrieben wird.

Verzeichnisse kopieren

Wenn Sie ein ganzes Verzeichnis mit allen darin enthaltenen Dateien und Unterverzeichnissen kopieren möchten, führen Sie `cp -r quellverzeichnis zielverzeichnis` aus. Damit werden auch versteckte Dateien und Unterverzeichnisse kopiert. Wenn Sie möchten, dass beim Kopieren die Zugriffsrechte und -zeiten erhalten bleiben, verwenden Sie statt `-r` die Option `-a`.

Etwas diffizil ist die Frage, ob das Quellverzeichnis selbst oder nur sein Inhalt kopiert wird. Wenn es das Zielverzeichnis bereits gibt, wird darin das neue Unterverzeichnis `quellverzeichnis` erzeugt und dorthin der gesamte Inhalt des Quellverzeichnisses kopiert. Wenn es das Zielverzeichnis hingegen noch nicht gibt, wird es erzeugt; in diesem Fall wird nur der Inhalt des Quellverzeichnisses in das neu erzeugte Zielverzeichnis kopiert, nicht aber das Quellverzeichnis selbst.

Zum Kopieren ganzer Verzeichnisbäume eignen sich auch `rsync` und `tar` (siehe Seite 127 bzw. 145).

Beispiele

Das folgende Kommando kopiert alle *.tex-Dateien aus dem Unterverzeichnis buch in das aktuelle Verzeichnis. Der Punkt gibt dabei als Zielverzeichnis das aktuelle Verzeichnis an.

```
user$ cp buch/*.tex .
```

Das zweite Kommando erstellt eine Backup-Kopie des gesamten Verzeichnisses buch:

```
user$ cp -a buch bak-buch
```

cp kann nicht dazu verwendet werden, mehrere Dateien beim Kopieren umzubenennen. cp *.xxx *.yyy kopiert also nicht alle *.xxx-Dateien in *.yyy-Dateien. Um solche Operationen durchzuführen, müssen Sie for oder sed zu Hilfe nehmen (siehe auch Seite 54 bzw. 129).

Im folgenden Kommando bildet for eine Schleife über alle *.xxx-Dateien. Der Ausdruck \${i%.xxx}.yyy entfernt die Endung *.xxx und ersetzt sie durch .yyy. Wenn Sie cp durch mv ersetzen, werden die Dateien nicht kopiert, sondern umbenannt.

```
user$ for i in *.xxx; do cp $i ${i%.xxx}.yyy; done
```

Etwas komplizierter ist die Vorgehensweise mit sed: ls liefert die Liste der Dateien, die kopiert werden sollen, und gibt sie an sed weiter. sed bildet daraus mit dem Kommando s (*regular find and replace*) eine Liste von cp-Kommandos und gibt diese wiederum an eine neue Shell sh weiter, die die Kommandos schließlich ausführt.

```
user$ ls *.xxx | sed 's/\(.*\)\.xxx$/cp & \1.yyy/' | sh
```

cpio kommando [optionen] [muster]

cpio fasst mehrere Dateien zu einem Archiv zusammen und kopiert sie auf einen anderen Datenträger (z. B. auf einen Streamer) oder in ein anderes Verzeichnis. Analog kann das Kommando auch zum Wiedereinlesen solcher Daten verwendet werden. Unter Linux ist cpio eher ungebräuchlich, stattdessen wird zumeist tar verwendet. Die drei zentralen cpio-Kommandos sind:

- o (*output*) zum Speichern von Daten. Die Dateien werden zu einem Archiv zusammengefasst und an die Standardausgabe geschrieben.
- i (*input*) zum Einlesen archivierter Daten.
- p (*pass through*) zur Übertragung von Archiven zwischen verschiedenen Verzeichnissen.

Diese drei Hauptkommandos können durch verschiedene Optionen gesteuert werden. Details dazu finden Sie in den man-Seiten.

cryptsetup [optionen] kommando

cryptsetup aus dem gleichnamigen Paket greift auf Funktionen des Kernelmoduls dm_crypt zurück. Es richtet Crypto-Devices ein, aktiviert und deaktiviert sie. An dieser Stelle werden nur die wichtigsten LUKS-spezifischen Kommandos von cryptsetup beschrieben.

luksAddKey *device*

richtet ein zusätzliches Passwort ein, das Zugriff auf den Crypto-Container gibt. Zur Ausführung des Kommandos muss ein bereits existierendes Passwort angegeben werden (legal, welches). Insgesamt sind maximal acht Passwörter erlaubt.

luksClose *device mappingname*

deaktiviert ein Crypto-Device.

luksDump *device*

liefert Metainformationen über den Crypto-Container (z. B. den Verschlüsselungsalgorithmus).

luksFormat *device*

richtet im angegebenen Device einen Crypto-Container ein. Dabei müssen Sie zweimal die *passphrase* angeben, die aus Sicherheitsgründen zumindest 20 Zeichen lang sein sollte. Standardmäßig wird zur Verschlüsselung der AES-Algorithmus im Modus cbc-essiv:sha256 mit einer Schlüssellänge von 128 Bit verwendet. Einen anderen Algorithmus können Sie mit *-c* angeben, eine andere Schlüssellänge mit *-s*. Welche Algorithmen der Kernel versteht, verrät die Pseudodatei */proc/crypto*.

luksOpen *device mappingname*

aktiviert den Crypto-Container im angegebenen Device und weist ihm einen Namen zu. Das resultierende Crypto-Device kann nun über */dev/mapper/mappingname* genutzt werden.

csplit [optionen] datei trennposition

csplit zerlegt eine Textdatei an vorgegebenen Stellen in mehrere Einzeldateien. Die Trennposition kann entweder durch eine direkte Zeilenangabe oder durch ein Suchmuster angegeben werden. Das Kommando liefert als Ergebnis die Dateien xx00, xx01 etc. und gibt am Bildschirm deren Längen aus. Durch die Angabe entsprechender Optionen sind natürlich auch »schönere« Dateinamen möglich. Durch cat kann aus diesen Einzeldateien wieder die Originaldatei zusammengesetzt werden. (Siehe auch split auf Seite 140 zum Zerlegen von beliebigen – auch binären – Dateien in kleinere Dateien zu je *n* Bytes.)

Angabe der Trennpositionen

Die Trennpositionen werden entweder durch eine Zeilenanzahl oder durch ein Suchmuster angegeben. Im einen Fall wird die Datei nach *n* Zeilen zerlegt, im anderen Fall vor oder nach dem Auftreten des Suchmusters. Wenn csplit die Datei mehrfach zerlegen soll (was zumeist der Fall ist), muss hinter der Zeilenanzahl bzw. dem Trennmuster angegeben werden, wie oft die Operation wiederholt werden soll.

n trennt die Datei nach *n* Zeilen.

/muster/

trennt die Datei in der Zeile vor dem Auftreten des Musters. (Die Zeile mit dem gefundenen Muster wird zur ersten Zeile der nächsten Datei.)

/muster/+n

/muster/-n

trennt die Datei *n* Zeilen nach (+) oder vor (-) dem Auftreten des Musters.

{*n*} zerlegt die Datei in *n*+1 Einzeldateien (und nicht nur in zwei Dateien).

Optionen

-f *datei* bzw. **--prefix=***datei*

verwendet den angegebenen Dateinamen zur Benennung der Ausgabedateien.

-k bzw. **--keep-files**

Bereits erzeugte Dateien werden beim Auftreten eines Fehlers nicht wieder gelöscht. Die Option muss insbesondere bei Musterangaben in der Form *n {*}* verwendet werden. Die Musterangabe erfolgt wie bei `grep`.

-z bzw. **--elide-empty-files**

verhindert die Erzeugung leerer Dateien. Ohne diese Option können leere Dateien insbesondere dann auftreten, wenn bereits die erste Zeile der Ausgangsdatei dem Suchmuster entspricht.

Beispiel

`csplit` zerlegt `total.txt` in die Dateien `teil.00`, `teil.01` etc. Die Einzeldateien sind jeweils 100 Zeilen lang.

```
user$ csplit -k -f teil. total.txt 100 {*}
```

Im zweiten Beispiel zerlegt `csplit` die Datei `total.txt` in kleinere Dateien, wobei die Trennung immer dann erfolgt, wenn eine Zeile mit dem Text `% ===` beginnt.

```
user$ csplit -k -f teil. total.txt '/^% ===/' {*}
```

curl [optionen] [url]

`curl` hilft bei der Übertragung von Dateien von oder zu einem Server, wobei alle erdenklichen Protokolle unterstützt werden (HTTP, HTTPS, FTP, SFTP, SCP etc.). Die externe Datei bzw. das externe Verzeichnis wird durch eine URL-Zeichenkette (*Uniform Resource Locator*) angegeben, die mit dem Protokollnamen beginnt (z. B. `http://server.de/datei`).

--limit-rate *n*

limitiert die Übertragungsgeschwindigkeit auf die angegebene Byteanzahl pro Sekunde. *n* kann der Buchstabe *k* oder *m* hintangestellt werden, um die Übertragungsrate auf *n* kByte oder MByte pro Sekunde zu limitieren.

-o *datei*

speichert die heruntergeladenen Daten in der angegebenen Datei, anstatt sie an die Standardausgabe weiterzuleiten.

-r *n1-n2*

überträgt den angegebenen Bytebereich der Datei.

-T *datei*

überträgt die angegebene Datei zum Server (*upload*). Statt des Dateinamens kann auch das Zeichen - angegeben werden, um Daten aus der Standardeingabe zu verarbeiten.

-u *name:password*

gibt den Login-Namen und das Passwort an.

Beispiele

Das folgende Kommando überträgt die angegebene Datei zum FTP-Server backupserver und speichert sie im Verzeichnis verz:

```
user$ curl -T datei -u username:password ftp://backupserver/verz
```

Um Daten aus dem Standardeingabekanal zu verarbeiten, geben Sie mit -T als Dateinamen einen Bindestrich an. Das folgende Kommando speichert das aus dem tar-Kommando resultierende Ergebnis direkt in der Datei *name.tgz* auf dem FTP-Server:

```
user$ tar czf - verz/ | curl -T - -u usern:pw ftp://bserver/name.tgz
```

cut [optionen] *datei*

cut extrahiert aus jeder Zeile eines Textes die durch Optionen angegebenen Spalten.

-b *liste* bzw. **--bytes *liste***

extrahiert die in einer Liste angegebenen Zeichen. Einzelne Einträge dürfen durch Kommata (aber nicht durch Leerzeichen) getrennt werden. Statt einzelner Zeichen dürfen auch ganze Bereiche angegeben werden, etwa -b 3-6,9,11-15.

-f *liste* bzw. **--fields *liste***

wie oben, aber jetzt für Felder (Datensätze), die durch Tabulatorzeichen getrennt sein müssen.

-d *zeichen* bzw. **--delimiter *zeichen***

gibt das Trennzeichen für -f an, das statt des Tabulatorzeichens verwendet werden soll.

-s bzw. **--only-delimited**

eliminiert alle Zeilen, die keine Daten enthalten, die der Option -f entsprechen. Kann nicht zusammen mit -b verwendet werden.