# Linux Befehle PDF

Viele alltägliche Aufgaben lassen sich unter UNIX- bzw. Linux- Betriebssystemen mithilfe von verschiedenen Befehlen schnell und einfach erledigen - Der sichere Umgang mit dem Terminal ist daher eine besonders wichtige Fähigkeit für jeden Linux-/UNIX-Benutzer.

In diesem Beitrag werden Sie einen umfassenden Überblick zu den wichtigsten dieser Befehle für UNIX / Linux erhalten. Die hier vorgestellten Befehle werden dazu in die Bereiche Grundlagen, Systemadministration und Netzwerke unterteilt. Im ersten Abschnitt Grundlagen finden Sie Befehle, die bei der Nutzung von Linux- bzw. UNIX- Betriebssystemen unabdingbar sind. Im Bereich Systemadministration lernen Sie Befehle kennen, die Sie bei Ihrer täglichen Administrationsarbeit benötigen. Schließlich folgt der Abschnitt Netzwerke, welcher sich besonders an fortgeschrittene Benutzer richtet, aber auch Einsteigern einige hilfreiche Informationen bietet.

## Inhaltsverzeichnis



## Grundlagen

### Ein Terminal öffnen

Bevor Sie mit der Verwendung der Befehle beginnen können, müssen Sie zunächst ein Terminal auf Ihrem Linux-/UNIX-System öffnen. Meist ist eine einfache Suche nach "terminal" über die Suchfunktion des Menü dabei ausreichend.

Auf manchen Linux-Distributionen haben Sie zusätzlich jedoch auch die Möglichkeit, über die Tastenkombination [Alt+F2] ein "Run"-Eingabefeld zu öffnen. Dort können Sie durch Eingabe von "gnome-terminal" beispielsweise das gnome-terminal öffnen.

```
gnome-terminal
```

## An- und Abmeldung

### login - Benutzer anmelden

Der Befehl login wird verwendet, um einen Benutzer anzumelden. Falls bereits ein anderer Benutzer angemeldet ist, wird dieser automatisch abgemeldet. Der Befehl wird normalerweise automatisch als Antwort auf die login:-Eingabeaufforderung im Terminal ausgeführt. Es folgt die Syntax des Befehls:

```
login [Optionen] [Benutzer]
```

Nach Ausführung des Befehls erfolgt eine Passwortabfrage für den Benutzer.

### logout - Aktiven Benutzer abmelden

Der zurzeit angemeldete Benutzer kann mit dem Befehl logout wieder abgemeldet werden:

```
logout
```

### whoami - Aktiven Benutzer anzeigen

Der Abruf des aktuellen Benutzers ist vor allem bei häufigem Wechsel von Identitäten sehr hilfreich. Der mit der aktiven Benutzer-ID (UID) verknüpfte Benutzername lässt sich dazu mit dem Befehl whoami ausgeben:

```
whoami
```

### su - Benutzer wechseln

Der Befehl su (switch user) wird in der Praxis immer wieder benötigt: Er ermöglicht es Ihnen, die Identität eines anderen Benutzers anzunehmen. Die Syntax des Befehls lautet grundsätzlich folgendermaßen:

```
su [Benutzername]
```

Wir der Benutzername nicht mit angegeben, so wird automatisch der root-Benutzer verwendet. Bevor Sie mit der Identität des jeweiligen Benutzers fortfahren können, erfolgt zudem eine Passwortabfrage.

Der Befehl wird vor allem bei administrativen Aufgaben, die besondere root-Rechte benötigen, immer wieder benötigt, um einen Wechsel auf den root-Benutzer zu realisieren.

### sudo - Befehlsausführung im Namen anderer Benutzer

Mit dem Befehl sudo (su "do") können Befehle im Namen bzw. mit den Berechtigungen anderer Benutzer ausgeführt werden. Anwendung findet der Befehl somit oft dann, wenn bestimmte Aktionen ausgeführt werden sollen, die root-Rechte erfordern.

Damit ein Benutzer / eine Gruppe den Befehl sudo verwenden darf, muss er / sie in der Datei /etc/sudoers registriert werden. Standardmäßig wird hier die Gruppe "sudo" bereitgestellt, dessen Mitglieder alle den sudo-Befehl verwenden dürfen.

Die allgemeine Syntax von sudo lautet:

```
sudo [Optionen] [Befehl]
```

sudo kann konkret mit der Option -i auch verwendet werden, um eine Rootshell zu starten:

```
sudo -i
```

Nähere Informationen zu sudo, der /etc/sudoers und verfügbaren Optionen sowie auch Beispiele finden Sie in unserem Beitrag sudo.

### exit - Beenden von Sitzungen

Sitzungen, wie zum Beispiel Root-Sitzungen, die zuvor per su-Befehl gestartet wurden, oder auch SSH-Sitzungen, können mit dem Befehl exit wieder beendet werden. Genauso kann exit auch verwendet werden, um den aktiven Benutzer abzumelden - ähnlich wie beim dem Befehl logout.

```
exit
```

## Verzeichnisnavigation

An erster Stelle ist es bei der Navigation von Verzeichnissen sicherlich sinnvoll zu wissen, wo wir uns gerade befinden. Der Befehl pwd (print working directory) gibt uns hierzu das aktuelle Arbeitsverzeichnis, also das Verzeichnis in dem der Benutzer gerade arbeitet, aus:

```
pwd [Optionen]
```

In den meisten Fällen werden bei der Verwendung von pwd keine Optionen benötigt. Wenn Sie dennoch Näheres zu den verfügbaren Optionen erfahren möchten, lesen Sie auch unseren Beitrag pwd.

Der Navigation von Verzeichnissen dient der Befehl cd (change directory), welcher das Wechseln des aktuellen Arbeitsverzeichnisses ermöglicht. cd ist ein eingebauter Befehl der Shell und muss damit nicht zusätzlich installiert werden.

cd wird folgendermaßen angewendet:

```
cd [Optionen] [Verzeichnis]
```

Optionen werden für die meisten Anwendungen des Befehls cd nicht benötigt. Die Angabe eines Verzeichnisses erfolgt mithilfe absoluter oder relativer Verzeichnispfade. Wenn Sie mehr über die verschiedenen Möglichkeiten zur Anwendung des Befehls cd erfahren möchten, sehen Sie sich auch unseren Beitrag cd an.

Bei der Verzeichnisnavigation ist es auch wichtig, die Inhalte eines Verzeichnisses, etwa das aktuelle Arbeitsverzeichnis, anzeigen zu können. Hierzu wird der Befehl ls verwendet:

```
ls [Optionen] [Datei / Verzeichnis]
```

Werden weder eine Datei, noch ein Verzeichnis angegeben, so gibt ls standardmäßig den Inhalt des aktuellen Arbeitsverzeichnisses, unter Auflistung der Datei- und / oder Verzeichnisnamen aus. Mithilfe absoluter oder relativer Verzeichnis- / Dateipfade können die Inhalte bestimmter anderer Verzeichnisse oder einzelne Dateien ausgegeben werden.

Eine der wichtigsten Optionen des Befehls ls ist die Option -l, welche die Ausgabe detaillierterer Informationen zu Berechtigungen, Zeitstempeln, Kapazitäten etc. ermöglicht:

```
ls -l [Datei / Verzeichnis]
```

Versteckte Dateien können mithilfe der Option -a angezeigt werden:

```
ls -a [Datei / Verzeichnis]
```

Nähere Informationen zum Befehl ls finden Sie in unserem Beitrag ls.

## Datei-/Verzeichnismanipulation

### mkdir - Erstellen von Verzeichnissen

Mithilfe von mkdir (make directory) können Sie neue Verzeichnisse erstellen, unter denen weitere Dateien und Verzeichnisse angelegt werden können:

```
mkdir [Optionen] [Pfad]
```

Wenn Sie einen ganzen Pfad von noch nicht existierenden, verschachtelten Verzeichnissen anlegen wollen, dann müssen Sie die Option -p zusätzlich übergeben. Fehlende Verzeichnisse des angegebenen Pfades werden bei der Ausführung automatisch angelegt.

```
mkdir -p [Pfad]
```

Für weitere Informationen lesen Sie unseren Beitrag mkdir.

### touch - Anlegen von Dateien / Aktualisieren der Zugriffszeit

Neue Dateien mit aktuellen Daten, wie beispielsweise der Zugriffszeit, können mit dem Befehl touch angelegt werden. touch legt jedoch nur dann neue Dateien der Länge null an, wenn die angegebene Datei noch nicht im jeweiligen Verzeichnis existiert. Existiert die Datei bereits, dann wird lediglich ihr Zugriffsdatum bzw. ihre Zugriffszeit aktualisiert. Die Syntax des Befehls lautet:

```
touch [Optionen] [Dateipfad]
```

Mehr Informationen dazu finden Sie in unserem Beitrag touch.

### rmdir - Löschen von Verzeichnissen

Der Befehl rmdir wird verwendet, um Verzeichnisse zu löschen. Dabei ist zu beachten, dass nur leere Verzeichnisse gelöscht werden können.

```
rmdir [Optionen] [Pfad]
```

### rm - Löschen von Dateien/Verzeichnissen

Einträge einer oder mehrerer Dateien können mit dem Befehl rm aus einem Verzeichnis gelöscht werden.

```
rm [Optionen] [Pfad]
```

Der Befehl kann außerdem verwendet werden, um rekursiv den gesamten Inhalt eines Verzeichnisses zu löschen. Dazu verwenden Sie die Option -r:

```
rm -r [Pfad]
```

### mv - Verschieben und Umbenennen

Je nach Anwendung lassen sich mit mv (move) Dateien und Verzeichnisse verschieben und umbenennen. Verschieben lassen sich Dateien / Verzeichnisse durch Angabe des alten und neuen Pfades:

```
mv [alter Pfad] [neuer Pfad]
```

Beim Umbenennen einer Datei / eines Verzeichnisses bleiben der alte und der neue Pfad gleich. Lediglich der Datei-/Verzeichnisname wird verändert.

## cp - Kopieren

Mit cp können Sie eine Datei in eine Datei mit einem anderen Namen, oder in ein anderes Verzeichnis kopieren. Außerdem haben Sie die Möglichkeit, mehrere Dateien in ein bestimmtes Verzeichnis zu kopieren. Die Syntax des Befehls lautet:

```
cp [Quelle] [Ziel]
```

Mehr Informationen zu diesem Befehl finden Sie in unserem Beitrag cp.

## nano - Texteditor

Der Befehl nano ruft einen einfachen Texteditor auf. Zum einen haben Sie die Möglichkeit, eine bestimmte, existierende Datei durch Angabe des Dateipfads mit dem Texteditor zu bearbeiten. Zum anderen können Sie auch eine neue Datei erstellen und diese direkt bearbeiten - Dazu wird entweder kein Pfad angegeben oder die angegebene Datei existiert nicht.

```
nano [Dateipfad]
```

Mehr Informationen zu diesem Befehl finden Sie in unserem Beitrag nano.

## cat - Ausgeben von Dateiinhalten

Mit cat können Sie den Inhalt einer Datei direkt im Terminal ausgeben, ohne diese zu manipulieren. Der Pfad der Datei wird dazu einfach hinter dem Befehl angegeben:

```
cat [Dateipfad]
```

Da die angegebene Datei bei der Ausgabe durch cat nicht modifiziert werden kann, eignet sich der Befehl hervorragend dazu, den Inhalt einer Datei nach der Bearbeitung noch einmal zu überprüfen.

Mehr Informationen zu diesem Befehl finden Sie in unserem Beitrag cat.

## less - Anzeige von Dateiinhalten mittels Pager

less dient als Pager der Anzeige des Inhalts von Dateien. Dabei ist es unter anderem möglich, sowohl vor- als auch rückwärts durch die Inhalte zu scrollen. Über verschiedene Tastenkombinationen, die im weiteren Verlauf noch gezeigt werden, sind auch noch andere Funktionen nutzbar. Im Allgemeinen wird less wie folgt aufgerufen:

```
less [Datei]
```

Die übergebene Datei wird standardmäßig in einem interaktiven Modus, welcher auf verschiedene Tastatureingaben reagiert, geöffnet. Der folgenden Tabelle sind die zur Steuerung von less verfügbaren Tastenkombinationen zu entnehmen:

| Taste | Funktion |
|---|---|
| [Pfeiltaste hoch] | Eine Zeile weiter |
| [Pfeiltaste runter] | Eine Zeile zurück |
| [E] | Alternative: Eine Zeile weiter |

| Taste | Funktion |
|---|---|
| [Y] | Alternative: Eine Zeile zurück |
| [Bild auf] | Eine Bildschirmseite weiter |
| [Bild ab] | Eine Bildschirmseite zurück |
| [F] | Alternative: Eine Bildschirmseite weiter |
| [B] | Alternative: Eine Bildschirmseite zurück |
| [Leertaste] | Alternative: Eine Bildschirmseite weiter |
| N [Z] | Springt N Zeilen vorwärts. Zuerst wird dazu eine Zahl N eingegeben und anschließend die Taste [Z] betätigt. |
| N [W] | Springt N Zeilen rückwärts. Wird analog zum Vorwärtsspringen um N Zeilen verwendet. |
| [G] | Springt zum Anfang der Datei |
| [Shift] + [G] | Springt zum Ende der Datei |
| [/] Suchbegriff | Springt zu Vorkommnissen des angegebenen Suchbegriffs. Für eine Suche wird zuerst die Slash-Taste betätigt und anschließend der Suchbegriff eingegeben. Mit der Taste [N] kann vorwärts durch die Suchergebnisse navigiert werden, mit der Tastenkombination [Shift] + [N] rückwärts. |
| [H] | Hilfeseite mit verfügbaren Funktionen anzeigen |
| [Q] | less beenden |

Weitere Informationen finden Sie in unserem Beitrag less. Lesen Sie zudem auch den Beitrag more, welcher einen alternativen Befehl für die Anzeige von Dateiinhalten behandelt.

## head - Ausgabe vom Dateianfang

Der Befehl head kann - ähnlich wie der zuvor gezeigte Befehl cat - verwendet werden, um Inhalte einer Datei auszugeben. head wird jedoch in der Regel nicht zur Ausgabe des kompletten Inhalts, sondern nur von einem bestimmten Anteil vom Anfang bzw. Kopf einer Datei, verwendet. Wird head nur der Pfad einer Datei übergeben, so werden die ersten zehn Zeilen dieser Datei ausgegeben:

```
head [Dateipfad]
```

Die Angabe mehrerer Dateipfade ist möglich - hierbei werden die Ausgaben jeder Datei jeweils von einer Kopfzeile mit dem entsprechenden Dateinamen angeführt. Soll eine andere Anzahl Zeilen ausgegeben werden als die standardmäßigen zehn, so kann ein entsprechender Wert mithilfe der Option -n spezifiziert werden:

```
head -n [Anzahl Zeilen] [Dateipfad]
```

Die meist benötigten Anwendungen des Befehls sind hiermit abgedeckt. Für weitere Optionen zu diesem Befehl lesen Sie unseren ausführlichen Beitrag head.

## tail - Ausgabe vom Dateiende

Das Gegenstück zu head bildet der Befehl tail, welcher zur Ausgabe eines Teils vom Ende einer Datei verwendet wird. Standardmäßig gibt tail die letzten zehn Zeilen der übergebenen Datei aus:

```
tail [Dateipfad]
```

Bei Angabe mehrerer Dateien werden auch hier Kopfzeilen ausgegeben. Ähnlich wie bei head kann zudem wieder die Option -n verwendet werden, um eine andere Anzahl Zeilen auszugeben:

```
tail -n [Anzahl Zeilen] [Dateipfad]
```

tail bietet außerdem noch die Option -f, um bei einer wachsenden Datei jeweils die hinzugekommenen Daten auszugeben. Werden dabei mehrere Dateien angegeben, dann wird immer eine Kopfzeile ausgegeben, wenn in einer anderen Datei etwas hinzugefügt wurde:

```
tail -f [Dateipfad]
```

Mehr Informationen zu diesem Befehl finden Sie in unserem Beitrag tail.

## ln - Verknüpfungen erstellen

Der Befehl ln (link) wird verwendet, um Verknüpfungen (Links) zu Dateien oder Verzeichnissen zu erstellen, wobei zwischen sogenannten Hard- und Softlinks unterschieden wird. Die allgemeine Syntax des Befehls lautet folgendermaßen:

```
ln [Optionen] [Ziel] [Verknüpfungsname]
```

Neben der allgemeinen Syntax ist zudem noch eine andere Syntax zum Verlinken mehrerer Dateien in ein Verzeichnis zu unterscheiden:

```
ln [Optionen] [Ziel(e)] [Linkverzeichnis]
```

Standardmäßig erzeugt ln sogenannte Hardlinks - sollen stattdessen Softlinks erzeugt werden, so verwenden Sie die Option -s:

```
ln -s [Ziel] [Verknüpfungsname]
```

Wenn Sie nähere Informationen zur Unterscheidung zwischen Hard- und Softlinks, den weiteren Optionen des Befehls sowie auch einige Beispielanwendungen benötigen, dann lesen Sie auch unseren Beitrag ln.

## Suchen in Dateien

Die Suche nach verschiedenen Mustern in Dateien ist mithilfe des Befehls grep ("Global Search for a Regular Expression and Print out matches lines") möglich. Neben festen Zeichenketten können für die Suche auch reguläre Ausdrücke verwendet werden. Es folgt die allgemeine Syntax des Befehls grep:

```
grep [Optionen] [Muster] [Dateipfad]
```

Neben der Angabe eines einzelnen Dateipfades ist auch die Übergabe mehrerer Dateipfade möglich. Die Pfade werden dabei durch Leerzeichen voneinander getrennt. In der Ausgabe kennzeichnet grep dann jeweils am Anfang jedes Suchergebnisses, zu welcher Datei die Suchergebnisse gehören.

Sollen alle Dateien in einem Verzeichnis sowie rekursiv aus Unterverzeichnissen durchsucht werden, so können Sie auch die Option -R verwenden und den entsprechenden Verzeichnispfad übergeben. Wird kein Verzeichnispfad angegeben, so sucht grep rekursiv im aktuellen Arbeitsverzeichnis:

```
grep -R [Muster] [Verzeichnispfad]
```

Ebenfalls hilfreich ist die Option -n, welche die Anzeige von Zeilennummern aktiviert. Die Zeilennummern werden dann an den Anfang jeder gefundenen Zeile angehängt:

```
grep -n [Muster] [Dateipfad]
```

Weitere Informationen zum Befehl grep, seinen Optionen sowie auch einige Anwendungsbeispiele finden Sie im Beitrag grep.

## Suchen im Dateisystem

Eine Suche nach Dateien oder Verzeichnissen lässt sich unter Linux- bzw. UNIX- Systemen leicht mit dem Befehl find durchführen. Sie können dabei von beliebigen Startpunkten aus ganze Verzeichniszweige durchsuchen und die Ergebnisse zudem auch filtern.

Der Befehl stellt neben den einfachen Such- und Filterfunktionen auch noch einige speziellere Funktionen bereit, die Ihn zu einem mächtigen Universalwerkzeug machen.

## find - Datei-/Verzeichnissuche

find kann mit einer Vielzahl Optionen ausgeführt werden, um verschiedene Daten für die Suche heranzuziehen und die Ergebnisse zusätzlich zu filtern. Die Syntax des Befehls lautet jedoch allgemein:

```
find [Startpunkt] [Optionen / Ausdruck]
```

### Filtern nach Name

Bei einer einfachen Datei- oder Verzeichnissuche nach einem bestimmten Namen wird die Option -name verwendet. Sie benötigen außerdem den Startpunkt für die Suche, sowie die Suchbedingungen bzw. den exakten Namen der gesuchten Datei / des gesuchten Verzeichnisses:

```
find [Startpunkt] -name [Datei-/Verzeichnisname]
```

Wenn Sie die Groß- und Kleinschreibung bei der Suche ignorieren wollen, dann können Sie auch die Option -iname verwenden:

```
find [Startpunkt] -iname [Datei-/Verzeichnisname]
```

Bei der Angabe von Datei- bzw. Verzeichnisnamen ist es auch möglich, sogenannte Wildcards zu verwenden, um eine dynamische Suche nach mehreren Dateien durchzuführen. Eine genaue Kenntnis des Dateinamen ist somit auch nicht nötig.

### Filtern nach Kapazität

Die Option -size wird verwendet, um Dateien und Verzeichnisse aufgrund ihrer Kapazität zu filtern. Die Syntax lautet hier:

```
find [Startpunkt] -size [+-][Kapazität][ckMG]
```

Bei der einfachen Angabe einer Kapazität werden alle Dateien/Verzeichnisse gesucht, die dieser genau entsprechen. Die Einheit können Sie dabei durch Anhängen der Zeichen 'c' für Byte, 'k' für Kilobyte, 'M' für Megabyte und 'G' für Gigabyte festlegen. Wenn Sie keine Einheit angeben, dann bezieht sich die Kapazität auf die Anzahl belegter Blöcke im Dateisystem.

Statt einer genauen Kapazitäts-Angabe können Sie schließlich auch ein Plus '+' oder Minus '-' voranstellen. Das '+' steht dann für "größer als", während das '-' für "kleiner als" steht.

### Filtern nach Zugriffsrechten

Bei der Suche können Sie auch nach Dateien mit bestimmten Zugriffsrechten filtern. Dazu verwenden Sie die Option -perm und geben die Zugriffsrechte als Oktalzahl an:

```
find [Startpunkt] -perm [Zugriffsrechte als Oktalzahl]
```

Informationen zur Schreibweise als Oktalzahl finden Sie in unserem Beitrag Linux-Berechtigungssystem.

### Filtern nach Besitzer

Mit der Option -user können Sie auch nach Dateien/Verzeichnissen mit einem bestimmten Besitzer suchen:

```
find [Startpunkt] -user [Benutzername]
```

### Filtern nach Typ

Zuletzt können Sie mit -type auch noch nach Elementen mit bestimmter Typisierung suchen. Die wichtigsten Parameter für die Nutzung mit dieser Option sind 'f' für Dateien (file), 'd' für Verzeichnisse (directories) und 'l' für symbolische Links (link).

Mehr Informationen zu diesem Befehl finden Sie in unserem Beitrag find.

## locate - Datei- / Verzeichnissuche mit Datenbank

Als Alternative zu find wird an dieser Stelle auch einmal kurz der Befehl locate vorgestellt. Dieser greift bei der Suche auf eine oder mehrere Datenbankdateien zurück, welche die Datei- und Verzeichnisnamen des Dateisystems beinhalten. Im Vergleich zum Befehl find bietet locate durch die Suche in einer Datenbank eine höhere Suchgeschwindigkeit - Voraussetzung ist jedoch die regelmäßige Aktualisierung der Datenbank(en). Der Befehl ist nicht auf allen Systemen standardmäßig vorinstalliert, kann jedoch mit einem Paketmanager wie apt nachinstalliert werden.

Bevor eine erste Suche durchgeführt wird, ist es unter Umständen notwendig, die Datenbank zu aktualisieren. Die Aktualisierung erfolgt durch den Cron-Daemon standardmäßig einmal täglich automatisch. Mit dem folgenden Befehl wird eine Aktualisierung manuell angestoßen:

```
updatedb
```

Eine einfache Suche wird mithilfe der folgenden Syntax durchgeführt:

```
locate [Suchmuster]
```

Neben einfachen Zeichenfolgen können im Suchmuster auch Wildcards ("*", "?", "[]") verwendet werden. Bei Verwendung einfacher Zeichenfolgen gibt locate alle Dateien und Verzeichnisse aus, die diese Zeichenfolge an beliebiger Stelle in ihrem Namen beinhalten. Werden dagegen Wildcards verwendet, so gibt locate nur exakte Übereinstimmungen aus. Weitere Informationen erfahren Sie in unserem Beitrag wildcard.

Dieser Abschnitt zeigt lediglich überblicksartig die grundlegendsten Funktionalitäten des Befehls locate. Genauere Informationen, wie beispielsweise zu den Optionen von updatedb und locate, finden Sie in unserem Beitrag locate.

# Systembefehle

## shutdown - Herunterfahren / Neustart

Mit dem Befehl shutdown können Neustarts, das Herunterfahren des Systems eingeleitet und durch bestimmte Optionen sogar zeitlich gesteuert werden.

Herunterfahren lässt sich das System mit der Option -h. Sie müssen zusätzlich außerdem einen Zeitpunkt festlegen, zu dem der Vorgang gestartet werden soll. Dieser kann entweder in Minuten nach Befehlsausführung oder im 24 Stunden Format "hh:mm" angegeben werden:

```
shutdown -h [Zeit]
```

Neustarts werden mit der Option -r (reboot) durchgeführt:

```
shutdown -r [Zeit]
```

Zuletzt gibt es noch die Möglichkeit, ein geplantes Herunterfahren / einen Neustart vorzeitig abzubrechen. Dazu verwenden Sie die Option -c (cancel):

```
shutdown -c
```

Unter Debian 10 Buster kann shutdown aufgrund des vollständigen Wechsels auf systemd nicht mehr ohne Weiteres eingesetzt werden. Sie können stattdessen auf den Befehl systemctl zurückgreifen, welcher im Folgenden einmal erklärt wird.

## systemctl - Kontrolle des systemd System- und Service-Manager

### Herunterfahren / Neustarten

Mit systemctl können Befehle an systemd gesendet werden, um beispielweise das System herunterzufahren oder neuzustarten. Ein Herunterfahren wird mit poweroff eingeleitet:

```
systemctl poweroff
```

Neustarts können Sie mit reboot durchführen:

```
systemctl reboot
```

### Steuerung von Services / Units

Neben poweroff und reboot bietet das Befehlszeilenwerkzeug systemctl auch noch Funktionen für die Steuerung von Systemprogrammen, sogenannten Units.

Zunächst gibt es die beiden Funktionen start und stop, welche zum Starten und Stoppen der Units verwendet werden können:

```
systemctl [start / stop] [Unit]
```

Die Funktion restart wird verwendet, um Units neuzustarten. Dies ist hilfreich, wenn nach einer Änderung beispielsweise die Konfigurationsdateien eines Programms neu eingelesen werden sollen:

```
systemctl [restart] [Unit]
```

Schließlich gibt es noch die Funktion status, welche, wie der Name schon vermuten lässt, den Status einer Unit im Terminal ausgibt.

```
systemctl [status] [Unit]
```

## Benutzer-/Gruppenverwaltung

In diesem Abschnitt werden Sie einige wichtige Befehle für die Verwaltung von Benutzern und Gruppen unter Linux- bzw. UNIX-Betriebssystemen kennen lernen.

Wenn Sie Näheres zur Funktionsweise des Benutzer-/Gruppensystems und den Befehlen erfahren wollen, lesen Sie auch unseren ausführlichen Beitrag Benutzer und Gruppen unter Linux. Sie finden dort außerdem Beispielanwendungen für die Befehle.

Die meisten der folgenden Befehle erfordern für ihre Ausführung die Rechte des root-Benutzers.

### Benutzer

#### adduser - Hinzufügen neuer Benutzer

Mit dem Befehl adduser können Sie neue Benutzer erstellen. Der Befehl benötigt dazu lediglich den Namen des zu erstellenden Benutzers, alle anderen notwendigen Daten werden automatisch abgefragt. Dazu gehört das Passwort, der vollständige Name des Benutzers, sowie einige optionale Daten wie die Zimmernummer, Telefon und Sonstiges.

```
adduser [Benutzername]
```

Neben dem Benutzer wird auch eine gleichnamige Gruppe und ein entsprechendes Home-Verzeichnis erstellt.

### passwd - Ändern von Passwörtern

Das Ändern von Passwörtern ist mit dem Befehl passwd möglich: Der Root-Benutzer hat dabei die Möglichkeit, das Passwort eines jeden Benutzers zu ändern/aktualisieren. Der jeweilige Benutzername wird dazu einfach als Parameter übergeben. Jeder Benutzer kann außerdem sein eigenes Passwort ändern - dazu ist keine Angabe eines Benutzernamen notwendig.

```
passwd [Benutzername]
```

Mehr Informationen zu diesem Befehl finden Sie in unserem Beitrag passwd.

### usermod - Bearbeiten von Benutzerkonten

Der Befehl usermod ermöglicht die Bearbeitung von Informationen und Gruppenzugehörigkeiten bereits existierender Benutzerkonten. Dabei kommen verschiedene Optionen zum Einsatz. Zunächst folgt jedoch die allgemeine Syntax des Befehls:

```
usermod [Optionen] [Benutzer]
```

Mit der Option -l können Sie einen bestehenden Benutzer umbenennen:

```
usermod -l [Neuer Benutzername] [Benutzer]
```

Benutzer können mit den Optionen -aG in Gruppen hinzugefügt werden:

```
usermod -aG [Gruppe/-n] [Benutzer]
```

### deluser - Löschen von Benutzern

Wollen Sie einen Benutzer wieder löschen, dann kommt der Befehl deluser zum Einsatz. Die Syntax lautet:

```
deluser [Optionen] [Benutzer]
```

Wenn Sie dem Befehl keine besonderen Optionen übergeben, dann wird zwar der Benutzer gelöscht, das Home-Verzeichnis, sowie alle vom Benutzer erstellten Daten bleiben jedoch erhalten. Wenn Sie das Home-Verzeichnis ebenfalls löschen wollen, dann verwenden Sie daher die Option --remove-home:

```
usermod --remove-home [Benutzer]
```

Sie können mit der Option --remove-all-files zudem auch alle Dateien des Benutzers löschen:

```
usermod --remove-all-files [Benutzer]
```

### chsh - Wechseln der Login-Shell

Die Login-Shell eines Benutzers kann mithilfe des Befehls chsh geändert werden. Da chsh Teil des Pakets passwd ist, ist es auf den meisten Systemen bereits vorinstalliert und kann direkt verwendet werden. Die Syntax von chsh lautet:

```
chsh [Optionen] [Benutzer]
```

Bei Ausführung ohne jegliche Argumente startet chsh im interaktiven Modus und fragt eine Login-Shell ab, die für den aktuellen Benutzer gesetzt werden soll. Dem Root-Benutzer ist es zudem möglich, einen Benutzernamen übergeben, um die Login-Shell für einen anderen Benutzer zu setzen - auch hier wird standardmäßig im interaktiven Modus gestartet. Es ist zu beachten, dass die jeweils spezifizierte Shell in der Datei /etc/shells vorhanden sein muss. Nur der Root-Benutzer darf jeden beliebigen Befehl angeben.

Die Login-Shell kann auch mit der Option -s spezifiziert werden, wenn eine interaktive Abfrage nicht gewünscht wird:

```
chsh -s [Login-Shell] [Benutzer]
```

Nähere Informationen zu diesem Befehl finden Sie in unserem Beitrag chsh.

### id - Benutzer- / Gruppeninformationen abfragen

Benutzer- und Gruppeninformationen lassen sich mithilfe des Befehls id für bestimmte Benutzer abfragen. Die ausgegebenen Informationen umfassen dann unter anderem die UID, die primäre GID sowie alle weiteren Gruppen des jeweiligen Benutzers. Da id Teil des Pakets coreutils ist, ist es in der Regel bereits vorinstalliert. Es folgt die allgemeine Syntax des Befehls:

```
id [Optionen] [Benutzer]
```

Wird id ohne Übergabe eines Benutzernamens ausgeführt, so werden die genannten Informationen für den aktuellen Benutzer abgefragt - andernfalls erfolgt die Abfrage für den jeweils angegebenen Benutzer. Die Ausgabe hat dabei das folgende Format:

```
uid=[UID]([Benutzername]) gid=[GID]([Gruppenname]) Gruppen=[GID]([Gruppenname]),...
```

Durch Angabe verschiedener Optionen ist es möglich, die Ausgabe von id weiter anzupassen. Unter anderem ist so etwa die Ausgabe nur von effektiven oder realen IDs möglich. Nähere Informationen zu diesen Optionen sowie auch Beispiele zur Anwendung des Befehls id können Sie im Beitrag id nachlesen.

## Gruppen

### addgroup - Hinzufügen neuer Gruppen

Neue Gruppen können Sie mit addgroup erstellen. Sie müssen dabei lediglich den Namen der Gruppe angeben:

```
addgroup [Gruppenname]
```

### groupmod - Bearbeiten von Gruppen

Mit dem Tool groupmod können Sie bestehende Gruppen bearbeiten. Die Syntax ist dabei ähnlich wie bei usermod:

```
groupmod [Optionen] [Gruppe]
```

Sie haben auch hier wieder die Möglichkeit, verschiedene Optionen einzusetzen. Eine der wichtigsten Optionen des Befehls ist --new-name, welche für das Umbenennen von Gruppen verwendet wird:

```
groupmod --new-name [Gruppe]
```

### delgroup - Löschen von Gruppen

Schließlich können Sie Gruppen mit delgroup auch wieder löschen:

```
delgroup [Optionen] [Gruppe]
```

Während Gruppen bei Ausführung ohne zusätzliche Optionen ohne jegliche Rückfrage gelöscht werden, gibt es hier noch die Option --only-if-empty, welche sicherstellt, dass Gruppen nur dann gelöscht werden, wenn sie auch wirklich leer sind - also keine Mitglieder mehr enthalten:

```
delgroup --only-if-empty [Gruppe]
```

## Berechtigungen

Unter Linux- bzw- UNIX-Betriebssystemen bekommen sowohl Dateien, als auch Verzeichnisse alle einen Besitzer (user), sowie bestimmte Zugriffsrechte zugeordnet. Grundlegende Kenntnisse zu den verfügbaren Befehlen sind daher äußerst wichtig.

Sie lernen in diesem Abschnitt die beiden Befehle chmod und chown kennen. Ein detaillierteres Bild zur Funktionsweise des Berechtigungssystems erhalten Sie in unserem Beitrag Das Linux-Berechtigungssystem.

## chmod - Anpassen von Zugriffsrechten

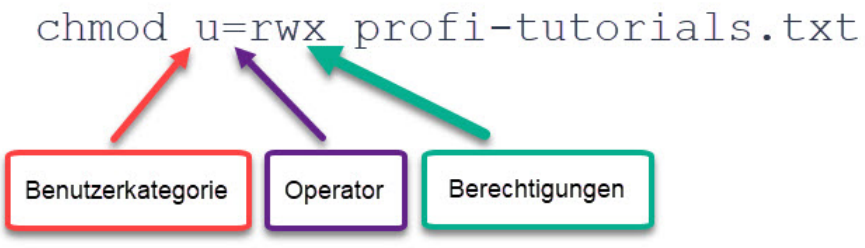
Der Befehl chmod (change mode) wird verwendet, um die Zugriffsrechte von Dateien anzupassen. Allgemein lautet die Syntax des Befehls folgendermaßen:

```
chmod [Optionen] [Berechtigungen / Modus] [Pfad]
```

Grundsätzlich gibt es 2 Methoden der Rechtevergabe mit dem Befehl: Die Symbolische- und die Numerische Rechtevergabe, welche nun einmal vorgestellt werden.

### Symbolische Rechtevergabe

Bei der symbolischen Vorgehensweise werden die Zugriffsrechte auf Dateien oder Verzeichnisse mit den Zeichen 'r', 'w' und 'x' zugeordnet. 'r' steht dabei für lesenden Zugriff (read), 'w' für die Schreibrechte (write) und 'x' schließlich für das Ausführungsrecht (execute). Die Syntax für die symbolische Rechtevergabe sieht folgendermaßen aus:

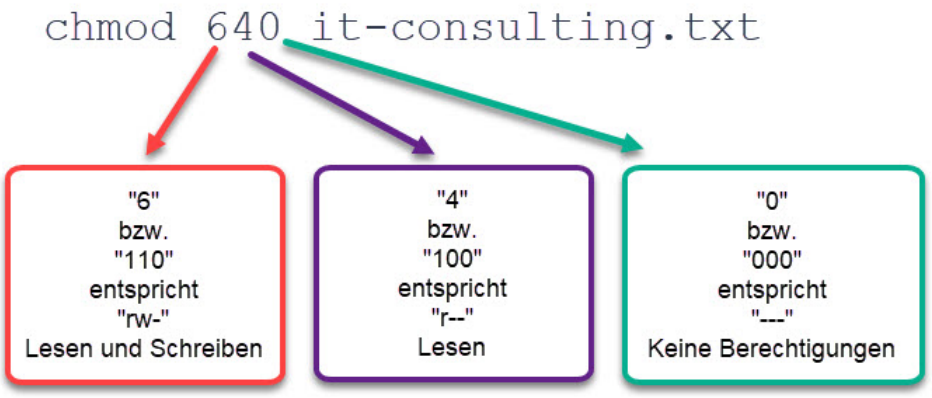



Zunächst wird die Benutzerkategorie ausgewählt, auf die die neuen Rechte angewendet werden sollen. Zugelassen sind die Kategorien 'u' für Benutzer (user), 'g' für Gruppe (group), 'o' für Andere (other) und zuletzt 'a' für alle (all). Anschließend folgt der Operator '-' um Rechte zu entfernen, '+' um Rechte zu erteilen und '=' um Rechte explizit zu setzen. Rechts neben dem Operator stehen dann schließlich die zu erteilenden Rechte in der Form "rwx".

Wenn Sie mehrere Änderungen an einer Datei / einem Verzeichnis mit einem Befehl vornehmen wollen, dann können Sie diese auch durch Kommata getrennt hintereinander auflisten.

### Numerische Rechtevergabe

Bei der numerischen Vorgehensweise werden die Zugriffsrechte in Form einer dreistelligen Oktalzahl angegeben. Jede der drei Ziffern steht dabei für die Rechte einer bestimmten Benutzerkategorie: Die erste Ziffer stellt die Berechtigungen für den Besitzer dar, die zweite für die Gruppe und die dritte für Andere.




Um die Berechtigungen abzulesen, die durch eine Ziffer dargestellt werden, muss diese in das Dualsystem konvertiert werden. Die resultierende dreistellige Dualzahl stellt die Berechtigungen dar.

Die erste Ziffer der Dualzahl steht dabei für den lesenden Zugriff, die zweite für das Schreibrecht und die dritte zuletzt für das Ausführungsrecht. Die 1 bedeutet dabei, dass das jeweilige Recht erteilt wurde, während eine 0 bedeutet, dass das Recht nicht erteilt wurde.

### chown - Anpassen von Besitzer und Gruppe

Mit dem Befehl chown können Sie den Besitzer bzw. die Gruppe einer Datei oder eines Verzeichnisses festlegen. Die Syntax lautet allgemein:

```
chown [Optionen] [Besitzer]:[Gruppe] [Pfad]
```

Sie können den Besitzer bzw. die Gruppe auch einzeln zuweisen. Um nur den Benutzer zuzuweisen, lassen Sie den Teil ":[Gruppe]" einfach aus:

```
chown [Optionen] [Besitzer] [Pfad]
```

Wenn Sie nur eine Zuweisung der Gruppe benötigen, dann können Sie den Teil "[Besitzer]" auslassen. Wichtig dabei ist, dass der Doppelpunkt ':' stehen bleibt, da dieser signalisiert, dass es sich um den Gruppennamen, nicht jedoch um den Namen des Besitzers handelt.

## History

Die auf einem Linux- / UNIX-System mit einem Benutzer ausgeführten Befehle werden in der sogenannten History protokolliert. Mit dem Befehl history ist es möglich, diese Aufzeichnungen auszulesen und Befehle erneut auszuführen. So ist es etwa möglich, in Erfahrung zu bringen, welche Befehle ein Benutzer wann ausgeführt hat.

### Die History ausgeben

Die History eines Benutzers kann durch die einfache Eingabe des Befehls history mit diesem Benutzer - ohne Optionen oder Parameter - ausgegeben werden:

```
history
```

Sollen nur die letzten n Eintragungen der History ausgegeben werden, so kann dem Befehl optional eine entsprechende Ganzzahl n angefügt werden:

```
history [Anzahl]
```

## Befehle ausführen

Links neben jeder Eintragung der History finden Sie ihre jeweilige Nummer. Möchten Sie einen Befehl der History erneut ausführen, geben Sie diese Nummer sowie ein vorangestelltes Ausrufungszeichen "!" im Terminal ein. Der Befehl mit der entsprechenden Nummer wird dann ausgeführt, ohne dass eine komplette Neueingabe nötig ist:

```
![Nummer]
```

Achten Sie bei der Wiederausführung von Befehlen darauf, dass bei relativen Pfadangaben Ihr aktuelles Arbeitsverzeichnis als Startpunkt verwendet wird - nicht das Verzeichnis, in welchem Sie sich bei der ersten Ausführung befanden.

## History löschen

Einzelne Befehle können mithilfe der Option -d und durch Angabe ihrer entsprechenden Nummer aus der History gelöscht werden:

```
history -d [Nummer]
```

Die komplette History ist dagegen mit der Option -c zu löschen:

```
history -c
```

Mehr Informationen zu diesem Befehl finden Sie in unserem Beitrag history.

## Befehlsausführung mit watch

Mit dem Befehl watch lassen sich unter UNIX- / Linux-Betriebssystemen Programme und Befehle in einem bestimmten Intervall ausführen - die Ausgabe wird jeweils an die Standardausgabe stdout weitergegeben und somit standardmäßig im Terminal ausgegeben. Das Programm erlaubt damit insbesondere die Beobachtung von Veränderungen und kann in Kombination mit anderen Werkzeugen beispielsweise zur Überwachung von Logdateien oder der Systemzeit verwendet werden.

Die allgemeine Syntax von watch lautet:

```
watch [Optionen] [Befehl / Programm]
```

Standardmäßig wird der angegebene Befehl bzw. das Programm alle zwei Sekunden ausgeführt. Möchten Sie jedoch ein anderes Intervall verwenden, dann können Sie dieses mit der Option -n spezifizieren. Die Angabe des Intervalls erfolgt in Sekunden:

```
watch -n [Intervall] [Befehl / Programm]
```

Für weitere Optionen des Befehls sowie einige Beispiele seiner Anwendung lesen Sie auch in unserem Beitrag watch.

## Datei-/Verzeichnisinhalte vergleichen

diff ermöglicht den Vergleich der Inhalte zweier Dateien oder Verzeichnisse miteinander. Von diff ausgegeben wird, worin sich der Inhalt der ersten angegebenen Datei bzw. des ersten Verzeichnisses mit dem der zweiten angegebenen Datei bzw. des zweiten Verzeichnisses

unterscheidet. Die Unterschiede werden also so ausgegeben, wie Sie den Inhalt ersten Datei modifizieren müssten, damit dieser identisch mit dem Inhalt der zweiten Datei ist. Die allgemeine Syntax des Befehls lautet wie folgt:

```
diff [Optionen] [Datei 1] [Datei 2]
```

Die Ausgabe des Befehls diff lässt sich mithilfe verschiedener Optionen zusätzlich anpassen. Die Option -q sorgt so etwa dafür, dass nur ausgegeben wird, ob sich die Inhalte überhaupt unterscheiden. Sind keine inhaltlichen Unterschiede vorhanden, bleibt die Ausgabe leer:

```
diff -q [Datei 1] [Datei 2]
```

Wenn dagegen eine Ausgabe erfolgen soll, wenn beide Dateien identisch sind, verwenden Sie die Option -s. Die Option kann beispielsweise auch mit der Option -q kombiniert werden, wenn ausschließlich textuelle Ausgaben zur generellen Gleichheit oder Ungleichheit der Inhalte der Dateien gewünscht sind:

```
diff -s [Datei 1] [Datei 2]
```

Weitere Informationen zu diesem Befehl und verfügbare Optionen finden Sie in unserem Beitrag diff.

## Laufzeitmessung mit time

Die Laufzeit eines Programms bzw. Befehls lässt sich mithilfe des Befehls time messen. Der Befehl eignet sich so etwa zur Messung der Dauer einer Sicherung oder ähnlichem. Zwei Varianten des Befehls gilt es zu unterscheiden: Einerseits bringt die Bash einen time-Befehl mit, der Laufzeiten messen und diese in einem POSIX-konformen Format ausgeben kann. Andererseits existiert jedoch auch noch ein externes Programm mit diesem Namen, welches auch noch weitere Funktionalitäten anbietet. Hier wird jedoch nur der Bash-Befehl time behandelt. Die allgemeine Syntax lautet folgendermaßen:

```
time [Optionen] [Befehl / Pipeline]
```

Die einzige Option des Befehls ist -p zur Ausgabe der Messergebnisse in einem POSIX-konformen Format. Für die Ausführung kann ein einzelner Befehl, aber auch eine ganze Pipeline (mehrere durch Pipes hintereinander geschaltete Befehle), angegeben werden.

Informationen zu dem hier nicht behandelten, externen Befehl finden Sie in unserem Beitrag time.

## Abzweigungen mit tee

Die Umleitung einer Ausgabe sowohl in den Standard-Output stdout, als auch in eine Datei, ist mit dem Befehl tee möglich. Konkret leitet der Befehl dazu den Standard-Input stdin an den Standard-Output stdout weiter und schreibt diesen außerdem in eine oder mehrere Dateien. Der Befehl bildet damit eine Art T-Stück, wovon sich auch sein Name ableitet. Die Syntax von tee lautet folgendermaßen:

```
tee [Optionen] [Datei(en)]
```

Die Angabe von mindestens einer Datei, in die geschrieben werden soll, ist notwendig. Der Inhalt wird standardmäßig überschrieben, jedoch kann die Option -a angehängt werden, um den Standard-Input stdin stets an den bestehenden Inhalt anzuhängen:

```
tee -a [Datei(en)]
```

Weitere Informationen zu diesem Befehl finden Sie in unserem Beitrag tee.

## Systemlaufzeit anzeigen

Die Laufzeit des Systems seit dem letzten Bootvorgang kann mithilfe des Befehls uptime ausgegeben werden. Neben der Laufzeit können der Ausgabe des Befehls unter anderem auch die Anzahl am System angemeldeter Benutzer sowie die mittlere Systemlast für die letzte Minute, 5 und 15 Minuten entnommen werden. Der Befehl wird meist ohne Optionen ausgeführt:

```
uptime
```

Standardmäßig erfolgt die Ausgabe auf einer Zeile: Hintereinander werden dabei die aktuelle Uhrzeit, die Laufzeit, die Benutzeranzahl und schließlich die Systemlast für die genannten drei Zeiträume ausgegeben. Nähere Informationen zum Befehl und seinen Optionen, sowie verschiedene Anwendungsbeispiele, finden Sie in unserem Beitrag uptime.

# Systemadministration

## Prozessmanagement

In diesem Abschnitt "Prozessmanagement" lernen Sie einige der wichtigsten Befehle zur Erkennung und Behebung von Fehlern und System-Instabilitäten. Ist Linux bzw. UNIX zwar ein äußerst stabiles System, so kann es dennoch immer auch zu Fehlern kommen - die hier gezeigten Befehle helfen Systemadministratoren bei der Bewältigung von alltäglichen Problemen.

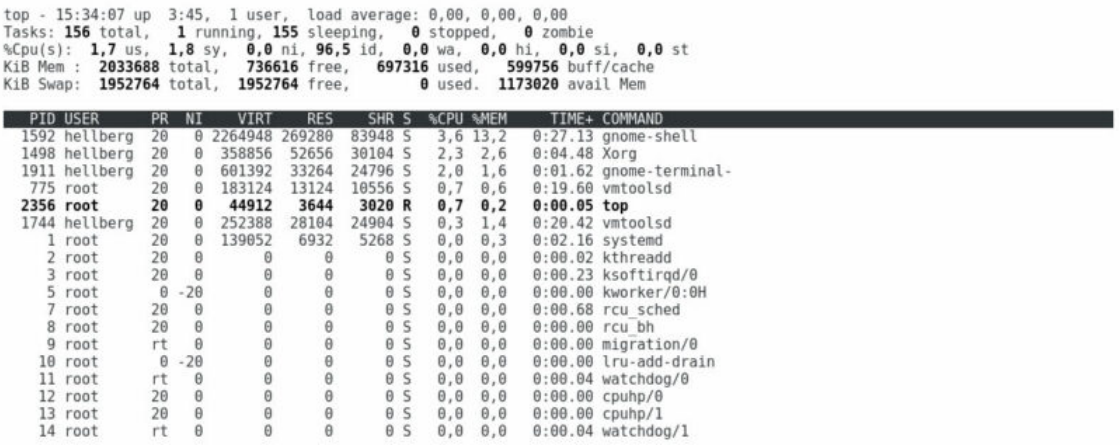
Weitere Informationen finden Sie auch in unserem Beitrag Befehle für das Prozessmanagement.

### top - Prozessübersicht

Mit dem Befehl top können Sie sich eine dynamische Übersicht aller laufenden Prozesse, sowie die Auslastung diverser Systemressourcen anzeigen lassen. Durch Eingabe des folgenden Befehls gelangen Sie zur Übersicht:

```
top
```

Die erscheinende Übersicht sieht etwa folgendermaßen aus:




Genauere Informationen zu diesem Befehl finden Sie auch in unserem Beitrag top.

### ps - Weitere Prozessübersicht

Eine Alternative für die Anzeige von Prozessen ist der Befehl ps (processes). Bei Eingabe des Befehls ohne Angabe zusätzlicher Parameter werden dabei die momentan im Terminal geöffneten Prozesse aufgelistet.

```
ps
```

Eine Übersicht zu allen auf dem System laufenden Prozessen lässt sich durch den zusätzlichen Parameter aux erreichen:

```
ps aux
```

## pstree - Prozesse in einer Baumstruktur anzeigen

Mithilfe des Befehls pstree ist es möglich, die auf einem System laufenden Prozesse in einer Baumstruktur darzustellen. Im Vergleich zum Befehl ps wird mithilfe der Baumstruktur eine bessere Übersicht, insbesondere auch über die (Eltern-Kind-)Beziehungen zwischen mehreren Prozessen, erzielt. Die allgemeine Syntax von pstree lautet folgendermaßen:

```
pstree [Optionen] [PID|Benutzer]
```

Wird pstree ohne Optionen und ohne Angabe einer PID bzw. eines Benutzers ausgeführt, dann gibt der Befehl alle Prozesse mit ihren Namen in einer Baumstruktur aus. Optional kann eine PID angegeben werden, um nur den Prozess mit dieser PID sowie seine Kindprozesse darzustellen. Schließlich kann anstelle einer PID auch ein Benutzer angegeben werden - dann werden alle Prozesse ausgegeben, die von dem entsprechenden Benutzer gestartet wurden.

Eine nützliche Option von pstree ist -p. Die Option aktiviert die Ausgabe der PIDs als Dezimalzahl (in Klammern) neben dem Namen jedes Prozesses:

```
pstree -p [PID|Benutzer]
```

Nähere Informationen zu diesem Befehl und seinen Optionen sowie auch Anwendungsbeispiele finden Sie in unserem Beitrag pstree.

## kill - Prozesse beenden

Prozesse, die unerwartet stehen geblieben sind, oder solche, die unerwünscht ausgeführt werden, lassen sich mit dem Befehl kill beenden. Bei der Verwendung des Befehls ist es nötig, neben optionalen Parametern die PID (Process Id) des jeweiligen Prozesses anzugeben.

```
kill [Optionen] [PID]
```

Standardmäßig sendet der Befehl das Signal SIGTERM (15) an den Kernel, um die Ausführung des jeweiligen Prozesses abzuschließen und ihn anschließend zu beenden. Mithilfe der Option -s lässt sich auch explizit festlegen, welches Signal an den Kernel gesendet werden soll:

```
kill -s [Signal] [PID]
```

Dies ist besonders dann hilfreich, wenn sich ein besonders hartnäckiger Prozess nicht durch das Standardsignal beenden lässt. In solchen Fällen kann der Kernel beispielweise mit einem SIGKILL (9) beauftragt werden, den Prozess hart abzubrechen:

```
kill -s SIGKILL [PID]
```

Beachten Sie jedoch, dass letztere Option immer nur dann eingesetzt werden sollte, wenn das Standardsignal nicht zum Ziel führt, da der Prozess auf diesem Weg nicht sauber beendet wird.

## jobs - Status von Shell-Prozessen anzeigen

Mit dem Befehl jobs können Sie den Status aller in der Shell gestarteten Prozesse einsehen:

```
jobs
```

Der Befehl wird vor allem dann benötigt, wenn Sie lang laufende Prozesse im Hintergrund ausführen möchten. Ähnlich wie bei den beiden Befehlen top und ps wird neben der Jobnummer der aktuelle Prozessstatus sowie auch der komplette Aufrufbefehl angezeigt. Die Ausgabe sieht dann etwa folgendermaßen aus:

```
[3]+  Angehalten              wget -nv https://profi-tutorials.de/it-consulting/
[4]-  Fertig                  wget -nv https://profi-tutorials.de/e-business/
root@hellix:/home/hellberg#
```

## & - Prozesse im Hintergrund starten

Wenn Sie einen Prozess im Hintergrund starten möchten, dann können Sie dem jeweiligen Befehl ganz einfach ein Kaufmanns-Und & anhängen:

```
[Befehl] [Optionen] &
```

Nach Eingabe wird im Terminal die zugewiesene Jobnummer und die PID ausgegeben.

## bg - Vordergrundprozess in den Hintergrund verlagern

Auch bereits im Vordergrund gestartete Prozesse lassen sich nachträglich in den Hintergrund verlagern. Mit der Tastenkombination [Ctrl+z] muss der laufende Prozess dazu zunächst angehalten werden. Anschließend kann er mit dem folgenden Befehl im Hintergund fortgeführt werden:

```
bg %[Jobnummer]
```

## fg - Hintergrundprozess in den Vordergrund verlagern

Hintergrundprozesse können umgekehrt auch in den Vordergrund verlagert werden. Verwenden Sie dazu den Befehl fg:

```
fg %[Jobnummer]
```

## nice - Prozesse mit angepasster Priorität starten

Mit nice ist es möglich, Prozesse bzw. Befehle mit einer angepassten Priorität (nice-Wert) zu starten. Der Wertebereich für nice reicht ganzzahlig von -20 (niedrigste Priorität) bis 19 (höchste Priorität). Standardmäßiger nice-Wert für einen Prozess ist 0. Wird nice ohne jegliche Argumente ausgeführt, so gibt der Befehl den aktuellen nice-Wert aus:

```
nice
```

Erst durch Übergabe eines Befehls mitsamt seiner Argumente ist es möglich, einen entsprechenden Prozess mit angepasster Priorität zu starten. Die Option -n ermöglicht dabei die Angabe der vorzunehmenden Anpassung des nice-Wertes, wobei der angegebene Wert auf den standardmäßigen nice-Wert aufaddiert wird.

```
nice -n [Anpassung] [Befehl [Argument(e)]]
```

Die Option -n kann auch weggelassen werden - in diesem Fall addiert nice standardmäßig den ganzzahligen Wert 10:

```
nice [Befehl [Argument(e)]]
```

Weitere Informationen finden Sie in unserem Beitrag nice.

## cpulimit - CPU-Zeit eines Prozesses begrenzen

Die prozentuale Begrenzung der von einem Prozess inklusive seiner Kindprozesse beanspruchten CPU-Zeit ist mit dem Befehl cpulimit möglich. Die maximal prozentual verfügbare CPU-Zeit beträgt dabei in einem System mit N Prozessoren oder Prozessorkernen N*100 Prozent. Es ist schließlich noch zu beachten, dass zur Begrenzung nicht-eigener Prozesse Root-Berechtigungen erforderlich sind. Da cpulimit nicht standardmäßig auf allen Systemen installiert ist, kann er bei Bedarf mit dem folgenden Befehl mit apt nachinstalliert werden:

```
apt-get install cpulimit
```

Die allgemeine Syntax von cpulimit lautet nun folgendermaßen:

```
cpulimit [Ziel] -l [Max Prozent] [Optionen]
```

Der Wertebereich für den maximal zugelassenen Anteil der CPU-Zeit ist 1 und aufsteigend. Das Ziel kann mithilfe verschiedener Optionen angegeben werden, wobei -p zur Angabe einer Prozess-ID die wohl wichtigste Option ist. Soll also ein Prozess mit der Prozess-ID [PID] begrenzt werden, dann kommt folgende Syntax zum Einsatz:

```
cpulimit -p [PID] -l [Max Prozent] [Optionen]
```

Offensichtlich muss der Prozess mit der angegebenen PID bereits existieren, wenn cpulimit mit der obigen Syntax verwendet wird. Alternativ ist es daher möglich, einen neuen Prozess zu starten und die Begrenzung der CPU-Zeit direkt auf diesem vorzunehmen. Hierzu wird ein auszuführender Befehl mitsamt seinem Argument hinter zwei Anführungszeichen angehängt:

```
cpulimit -l [Max Prozent] [Optionen] -- [Befehl]
```

Nähere Informationen zu Befehl, die weiteren Optionen sowie auch einige Beispiele finden Sie in unserem Beitrag cpulimit.

## ionice - Prozesse mit angepasster I/O-Priorität starten

ionice ermöglicht Ihnen die Festlegung der Priorität für Prozesse, die das I/O-Interface des Kernels nutzen. So kann etwa ein Kopierprozess mit geeignet angepasster I/O-Scheduling-Klasse und -Priorität gestartet werden, damit die Auslastung des Systems durch diesen Prozess möglichst gering bleibt. Wird ionice ohne Argumente ausgeführt, so gibt der Befehl die I/O-Scheduling-Klasse und -Priorität des aktuellen Prozesses aus:

```
ionice
```

Durch Übergabe eines Befehls sowie der Optionen -c und -n ist es möglich, diesen Befehl mit angepassten Werten zu starten. Der Option -c kann dabei eine der verfügbaren I/O-Scheduling-Klassen in ihrer numerischen Form übergeben werden. Die Klassen mit ihren numerischen Werten sind die folgenden: "None" (0), "Realtime" (1), "Best-Effort" (2), "Idle" (3). Die Priorität (Option -n) kann nur zusammen mit den Klassen "Realtime" und "Best-Effort" verwendet werden - andernfalls wird sie ignoriert. Der Wertebereich der Priorität ist 0-7, wobei 0 die höchste Prioritätsstufe darstellt.

```
ionice -c [Klasse] -n [Priorität] [Befehl [Argument(e)]]
```

Wenn Sie näheres zum diesem Befehl und den I/O-Scheduling-Klassen erfahren möchten, dann lesen Sie auch unseren Beitrag ionice.

## chrt - Echtzeit-Scheduling-Attribute auslesen und setzen

Der Befehl chrt ermöglicht es Ihnen, die Echtzeit-Scheduling-Attribute von Prozessen auszulesen und zu setzen. Neben der Scheduling-Priorität kann so etwa auch der verwendete Scheduler beeinflusst werden. Bei der Verwendung von chrt werden in der Regel Root-

Rechte benötigt. Die allgemeine Syntax, um einen Befehl mit gegebenen Attributen zu starten:

```
chrt [Optionen] [Priorität] [Befehl [Argumente]]
```

Alternativ können auch die Attribute eines bereits laufenden Prozesses geändert werden. Hierzu wird die PID des Prozesses mithilfe der Option -p spezifiziert. Werden bei dieser Syntax weder Priorität noch sonstige Optionen angegeben, so gibt chrt die aktuell für den Prozess gesetzten Attribute aus:

```
chrt [Optionen] -p [Priorität] [PID]
```

Die Priorität wird im Wertebereich 0 (niedrig) bis 99 (hoch) angegeben. Die Optionen des Befehls können zur Auswahl eines Schedulers verwendet werden: Verfügbar sind -o (Standardscheduler), -i (Idle-Scheduler), -b (Batch-Scheduler), -f (FIFO-Scheduler) und -r (Round Robin-Scheduler). In unserem Beitrag chrt können Sie weitere Informationen zum Befehl selbst und den verfügbaren Schedulern nachlesen.

## Runlevel

Diverse sogenannte Runlevel kontrollieren unter Linux-/UNIX-Betriebssystemen, welche Prozesse und Services automatisch - beispielsweise beim Systemstart - mitgestartet werden sollen. In diesem Abschnitt lernen Sie den Befehl init, die 7 verfügbaren Runevel, sowie den Standard-Runlevel kennen.

Beachten Sie, dass die Runlevel in der hier beschriebenen Form nur bei dem init-System bzw. Paket sysvinit zum Einsatz kommen. Das init-System wurde unter Debian seit der Version 8 durch systemd ersetzt.

### init - Runlevel wechseln

Der Befehl init wird eingesetzt, um den Runlevel des Systems zur Laufzeit zu wechseln. Die Syntax des Befehls lautet:

```
init [Runlevel]
```

Die verfügbaren Runlevel, welche sich mit dem init-Befehl einstellen lassen, werden nun einmal kurz beschrieben:

Die Runlevel sind von 0 bis 6 durchnummeriert und führen jeweils zu unterschiedlichen Betriebsarten des Systems: Der Runlevel 0 führt zu einem Systemhalt, kann also eingesetzt werden, um das System zu stoppen. Der Runlevel 1 startet den Einzel-Nutzerbetrieb des Systems. Es gibt dabei keine Netzwerkverbindung. Runlevel 2 startet den lokalen Multi-Nutzerbetrieb, wobei auch hier wieder die Netzwerkverbindung deaktiviert bleibt. Der dritte Runlevel (3) führt zum Start des vollen Multi-Nutzerbetriebs - hierbei werden sämtliche Services, wie beispielsweise ein Apache Webserver oder Mailserver, ebenfalls gestartet und es besteht eine Netzwerkverbindung. Ähnlich funktioniert auch Runlevel 5, jedoch wird bei diesem auch die grafische Benutzungsoberfläche (KDM / XDM / GDM) gestartet. Schließlich kann das System mithilfe des letzten Runlevel (6) neugestartet werden.

Der vierte Runlevel wurde absichtlich übersprungen, da dieser noch nicht belegt ist.

### Standard-Runlevel

Der Standard-Runlevel, welcher bei jedem Systemstart eingesetzt wird, wird in der Datei /etc/inittab durch den Eintrag initdefault festgelegt. In den meisten Fällen wird hier der volle Multi-Nutzerbetrieb ohne grafische Benutzungsoberfläche (3), oder der entsprechende Multi-Nutzerbetrieb mit aktivierter grafischer Benutzungsoberfläche (5) verwendet.

## Ein-/Aushängen mit mount

In diesem kurzen Abschnitt soll der Befehl mount, welcher unter UNIX-/Linux-Betriebssystemen zum Einbinden und Aushängen von Dateisystemen verwendet wird, vorgestellt.

### Dateisysteme anzeigen

Zunächst können Sie durch Eingabe des Befehls mount ohne jegliche Optionen alle momentan in das System eingebundenen Dateisysteme ausgeben:

```
mount
```

Die Ausgabe erfolgt zeilenweise - Am Anfang jeder Zeile steht dabei die Bezeichnung des Dateisystems oder Geräts. Darauf folgt der Pfad, auf welchem das Dateisystem eingehängt wurde. Zum Schluss folgen Informationen zum Typ (type) des Dateisystems, beispielsweise ext4, sowie zu verschiedenen festgelegten Einhängeoptionen, wie beispielsweise der GID, UID oder der Dateinamen Enkodierung.

Mit der Option -l lässt sich am Ende jeder Zeile zusätzlich auch noch ein Label in eckigen Klammern darstellen:

```
mount -l
```

## Dateisystem einhängen

Je nachdem, ob es in der sogenannten /etc/fstab bereits einen Eintrag gibt, werden unterschiedliche Syntaxen für den mount Befehl benötigt. Die genaue Funktionsweise der /etc/fstab finden Sie in unserem Beitrag Die Konfigurationsdatei fstab.

Dem Befehl müssen neben Optionen, das Gerät bzw. Dateisystem sowie der Einhängepunkt (Mountpoint) übergeben werden. Außerdem werden root-Rechte benötigt. Die Syntax lautet damit:

```
mount [Optionen] [Gerät / Dateisystem] [Mountpoint]
```

Für das temporäre Einhängen ohne Eintragung in der /etc/fstab können Sie unter [Gerät / Dateisystem] eine sogenannte Gerätedatei (device) angeben, welche Sie unter UNIX-/Linux-Betriebssystemen im Verzeichnis /dev/ finden. Beispielsweise sind /dev/sda1 und /dev/sda2 die jeweils erste Partition und zweite Partition eines SCSI Device.

Als Mountpoint dient ein beliebiger Pfad eines Verzeichnisses, auf dem Sie das neue Gerät einhängen möchten. Das spezifizierte Verzeichnis sollte vorzugsweise leer sein, da jegliche Inhalte nach der Einhängung nicht mehr sichtbar sind - sie werden von dem zweiten Dateisystem überlagert, sind jedoch nach dem Aushängen des zweiten Dateisystems wieder verfügbar.

## Optionen

Der mount Befehl besitzt eine Vielzahl Optionen, von denen die wichtigsten einmal kurz in der folgenden Tabelle aufgelistet werden.

| Option | Funktion |
|---|---|
| -a | Es werden alle Dateisysteme der /etc/fstab (der angegebenen Typen) eingebunden, solange diese nicht mit der Einhängeoption noauto versehen wurden. (all) |
| -U [UUID] | Es wird das Dateisystem mit der angegebenen UUID eingehängt. |
| -L [Label] | Es wird das Dateisystem mit dem angegebenen Label eingehängt. |
| -B | Ein bereits gemounteter Teilbaum kann mit dieser Option an einer anderen Position erneut eingehängt werden. Der alte Mountpoint bleibt dabei erhalten, sodass die Inhalte an zwei Orten erscheinen. (Bind) |
| -M | Ein bereits gemounteter Teilbaum wird an eine andere Position verschoben. (Move) |
| -f | Das Einhängen des angegebenen Dateisystems wird lediglich simuliert, also nicht tatsächlich durchgeführt. Durch Zusatz der Option -v können die Aktionen des mount Befehls nachvollzogen werden. (fake) |
| -o [Parameter] | Wird verwendet, um die Einhängeoptionen festzulegen. Als Parameter übergeben Sie hierbei eine durch Kommata getrennte Liste von Einhängeoptionen. Verfügbare Einhängeoptionen finden Sie im Abschnitt "Einhängeoptionen". (options) |
| -r | Das Dateisystem wird schreibgeschützt eingehängt. Synonym dieser Befehlszeilenoption ist das Hinzufügen der Einhängeoption ro durch -o ro (read only) |

| Option | Funktion |
|---|---|
| -w | Das Dateisystem wird les- und schreibbar eingehängt. Es handelt sich hierbei um die Voreinstellung bzw. den Standardwert des Kernels. Auch hier ist -o rw ein Synonym (read write) |
| -t | Definiert den Typen des einzuhängenden Dateisystems. Beispielsweise: ext, ext2, ext3, ext4, nfs, iso9660, vfat (=FAT32), ntfs etc. (type) |
| -v | Aktiviert den ausführlichen Modus. (verbose) |

Mehr Informationen zu diesem Befehl finden Sie in unserem Beitrag mount.

## Archivierung

Hier sollen Ihnen kurz die Grundfunktionen des Befehls tar (tape archiver), welcher unter Linux-/UNIX-Betriebssystemen für die Erstellung von .tar Archiven sowie die Extraktion diverser archivierter Dateien verwendet wird,vorgestellt werden. Bei der Systemadministration spielt der Befehl oft eine zentrale Rolle, da er beispielsweise auch bei der Durchführung von Backups eingesetzt werden kann.

Die allgemeine Syntax lautet:

```
tar [Optionen] [Datei(en) / Verzeichnis(se)]
```

Im Folgenden sollen die wichtigsten Optionen zur Erstellung, Auflistung und Extraktion von Archiven kurz genannt werden.

### -c - Archive erstellen

Zur Erstellung neuer Archive wird die Option -c des tar Befehls verwendet. Die Syntax lautet:

```
tar -c -f [Archivname] [Datei(en) / Verzeichnis(se)]
```

Benötigt wird hier zusätzlich die vielfach verwendete Option -f (file), welche den Namen des zu Erstellenden Archivs festlegt - bestenfalls endet dieser mit der Dateiendung .tar.

### -t - Archivinhalt ausgeben

Die Option -t wird verwendet, um Inhalt eines Archivs auszugeben. Grundsätzlich kann auf die Angabe [Datei(en) / Verzeichnis(se)] verzichtet werden, wenn ausnahmslos alle Inhalte ausgegeben werden sollen:

```
tar -t -f [Archivname]
```

Wenn Sie jedoch den Inhalt bestimmter Verzeichnisses des Archivs ausgeben möchten, dann fügen Sie die jeweiligen relativen Pfade einfach zusätzlich an:

```
tar -t -f [Archivname] [Datei(en) / Verzeichnis(se)]
```

### -x - Archive extrahieren

Mit der Option -x ist es schließlich möglich, Archive auch wieder zu extrahieren - erst dann ist der Zugriff auf die enthaltenen Dateien wieder möglich. Wird bei der Extraktion keine Angabe neben dem Archivnamen gemacht, so wird stets der gesamte Inhalt des jeweiligen Archivs in das aktuelle Arbeitsverzeichnis extrahiert:

```
tar -x -f [Archivname]
```

Durch Angabe von Dateien und / oder Verzeichnissen des Archivs können Sie jedoch auch genau spezifizieren, was extrahiert werden soll:

```
tar -x -f [Archivname] [Datei(en) / Verzeichnis(se)]
```

Schließlich können Sie durch Zusatz der Option -C auch ein Zielverzeichnis für die Extraktion bestimmen:

```
tar -x -f [Archivname] -C [Zielverzeichnis] [Datei(en) / Verzeichnis(se)]
```

Weitere Informationen finden Sie in unserem Beitrag Archivierung mit tar.

## Systemzeit und Hardware-Uhr

Unter Linux/UNIX gibt es prinzipiell zwei unterschiedliche Uhren: Die Systemzeit und die Hardware-Uhr. Während die Systemzeit von Programmen und Anwendungen zur Laufzeit verwendet wird, läuft die Hardware-Uhr zu jeder Zeit - auch wenn das System herunterfahren ist weiter, um nach dem Systemstart wieder die Systemzeit zu synchronisieren.

In diesem Beitrag lernen Sie zunächst die wichtigsten Funktionen des Befehls date kennen, welcher zur Ausgabe und Einstellung der Systemzeit verwendet wird. Anschließend wird auch noch auf den Befehl hwclock eingegangen - dieser gibt Ihnen die Möglichkeit, den Wert der Hardware-Uhr auszugeben, sowie diverse Einstellungen vorzunehmen.

## date - Systemzeit

### Ausgabe der Systemzeit

Im Folgenden wird der Befehl date mit seinen wichtigsten Optionen vorgestellt. date lässt sich ohne Angabe jeglicher Optionen einsetzen und gibt dann die Systemzeit (Datum und Uhrzeit) in einem festen Format aus:

```
date
```

Die Ausgabe der aktuellen Zeit ist durch Anfügen diverser Optionen auch in verschiedenen Formaten möglich. So wird mithilfe der Option -I eine Ausgabe nach ISO 8601 erreicht. Über das Feld [FMT] können Sie die Genauigkeit der Ausgabe bestimmen: Erkannt werden 'hours' für Stunden, 'minutes' für Minuten, 'seconds' für Sekunden und 'ns' für Nanosekunden.

```
date -I[FMT]
```

Weiterhin ist auch die Ausgabe im Format nach RFC 2822 möglich:

```
date -R
```

Wenn Sie das Format der Ausgabe selbst bestimmen möchten, dann verwenden Sie den date-Befehl einfach in Kombination mit verschiedenen Formatangaben. Das Format wird in Anführungszeichen und mit einem vorangestellten Plus '+' angegeben:

```
date "+[Format]"
```

### Setzen der Systemzeit

Mit der Option -s kann die Systemzeit auf einen von Ihnen bestimmten Wert gesetzt werden. Spezifiziert wird die Zeit mithilfe eines sogenannten date-String:

```
date -s [String]
```

Das Format des date-String kann sehr unterschiedlich aussehen. Hierbei soll nun lediglich ein Format vorgestellt werden, welches für die meisten Anwendungsfälle bestens geeignet ist. Die Syntax des gesamten Befehls lautet dann:

```
date -s "YYYY-MM-DD hh:mm:ss [Zeitzone]"
```

Welche gültigen Formatangaben es noch gibt, finden Sie in unserem Beitrag date.

## hwclock - Hardware-Uhr

### Ausgabe der aktuellen Zeit

Die Ausgabe der aktuellen Zeit ist auch bei der Hardware-Uhr leicht durchzuführen: Der Befehl hwclock wird dazu ohne Angabe von Optionen ausgeführt:

```
hwclock
```

Alternativ kann jedoch auch die Option -r verwendet werden, welche dieselbe Ausgabe liefert:

```
hwclock -r
```

Die Ausgabe erfolgt immer nach dem Systemgebietsschema, also der lokalen Zeit.

### Hardware-Uhr einstellen

Zur Einstellung der Hardware-Uhr stehen Ihnen zwei verschiedene Möglichkeiten zur Verfügung. So können Sie die Systemzeit einerseits mithilfe der Option -w nach der aktuellen Systemzeit stellen:

```
hwclock -w
```

Andererseits ist es auch möglich, ähnlich wie bei dem date-Befehl, die Uhr auf Grundlage einer Datumszeichenkette (date-String) zu stellen. Verwenden Sie dazu die beiden Optionen --set und --date:

```
hwclock --set --date='[String]'
```

Das Format der Datumszeichenkette kann beispielsweise die folgende Form annehmen:

```
hwclock --set --date='YYYY-MM-DD hh:mm:ss'
```

Weiterführende Optionen und Funktionen zu diesem Befehl finden Sie in unserem Beitrag hwclock.

## Sicheres Löschen von Daten

In dem folgenden Abschnitt wird Ihnen der Befehl shred vorgestellt, mit welchem Sie sicher Daten überschreiben und anschließend löschen können. Einige der wichtigsten Optionen werden zusätzlich aufgelistet.

### Dateien überschreiben

Wenn Sie einzelne Dateien mit Zufallsdaten überschreiben möchten, dann übergeben Sie dem Befehl shred einfach die Pfade der jeweiligen Dateien. Mehrere Dateien können durch Leerzeichen getrennt hintereinander angegeben werden. Optionen werden nicht zwingend benötigt.

```
shred [Datei(en)]
```

Wenn Sie Dateien nach dem Überschreiben auch noch Löschen möchten, dann verwenden Sie zusätzlich die Option -u:

```
shred -u [Datei(en)]
```

### Geräte überschreiben

Durch Angabe eines Gerätenamen können Sie auch ganze Blockdevices, wie beispielsweise Partitionen oder Festplatten überschreiben. Auch hier werden wieder keine Optionen benötigt.

```
shred [Gerät]
```

Durch Hinzufügen der Option -f ist es möglich, den Vorgang des Überschreibens zu erzwingen - die Option ändert Zugriffsberechtigungen automatisch, wenn dies für die Ausführung des Befehls nötig ist. Eingesetzt werden kann die Option bei Verwendung von shred zum Überschreiben von Dateien sowie Geräten.

```
shred -f [Gerät / Datei(en)]
```

Wenn Sie zusätzlich festlegen möchten, wie oft die angegebenen Daten überschrieben werden, dann können Sie mit der Option -n einfach einen entsprechenden numerischen Wert an den Befehl übergeben:

```
shred -n [N] [Gerät / Datei(en)]
```

Standardmäßig werden Dateigrößen von shred auf die nächste volle Blockgröße aufgerundet - dieses Verhalten lässt sich mithilfe der Option -x jedoch auch ausschalten:

```
shred -x [Gerät / Datei(en)]
```

Zur Anzeige des Fortschritts während des Überschreibens geben Sie die Option -v an:

```
shred -v [Gerät / Datei(en)]
```

Für weitere Informationen über diesen Befehl lesen Sie gerne unseren Beitrag shred.

## Hardware-Informationen

In diesem Abschnitt lernen Sie einige nützliche Befehle kennen, welche das Auslesen von Hardware-Informationen auf Linux-/UNIX-Betriebssystemen ermöglichen.

Das im Folgenden mehrfach benötigte Paket lshw, welches nicht auf allen Systemen standardmäßig vorinstalliert ist, können Sie mit dem folgenden Befehl nachinstallieren:

```
apt-get install lshw
```

### Überblick zur Hardware-Konfiguration

Den Befehl lshw können Sie verwenden, um einen detaillierten Überblick zur gesamten Hardware-Konfiguration eines Systems zu erhalten. Führen Sie ihn dazu ohne jegliche Optionen und Parameter aus:

```
lshw
```

Zur Ausgabe gehören unter anderem Informationen zu Speicher-Konfiguration, Firmware-Versionen, Mainboard-Konfiguration, CPU, Cache und Bus-Geschwindigkeiten.

### CPU-Informationen

Mit dem Befehl lscpu können Sie nähere Informationen zur CPU des Systems ausgeben. Es werden verschiedenste Informationen, wie beispielsweise Architektur, Anzahl Kerne, Threads, Hersteller-ID, Modellname und Frequenz, ausgegeben.

```
lscpu
```

Den schon bekannten Befehl lshw können Sie zusammen mit der Option -C und dem Parameter 'cpu' verwenden, um eine ähnliche, jedoch etwas unübersichtliche Ausgabe zu erhalten:

```
lshw -C cpu
```

### RAM-Informationen

Informationen zum Arbeitsspeicher lassen sich ebenfalls mit lshw ausgeben. Verwenden Sie hierbei die Option -C mit dem Parameter 'memory'. In der Ausgabe werden für jeden verbauten DIMM einzeln der Typ sowie die Kapazität, Geschwindigkeit und Spannung ausgegeben.

```
lshw -C memory
```

### Laufwerksinformationen

Wird der Befehl lshw mit der Option -C und dem Parameter 'disk' ausgeführt, so gibt er Informationen zu allen Laufwerken des Systems aus:

```
lshw -C disk
```

### Informationen zu USB- und PCI-Geräten

Weiterhin können Informationen zu USB- und PCI-Bussen eines Systems ausgegeben werden. Verwenden Sie dazu die Befehle lsusb und lspci.

Zur Ausgabe aller per USB an das System angeschlossenen Geräte führen Sie den Befehl lsusb einfach ohne jegliche Optionen und Parameter aus. Die Ausgabe erfolgt dann zeilenweise, wobei der Bus, die Hardware-ID und ein Namensteil aus Vendor- und Product-ID enthalten sind.

```
lsusb
```

Mit lspci werden Ihnen Informationen zu den PCI-Bussen und angeschlossenen Geräten angezeigt:

```
lspci
```

### Informationen zu Netzwerkgeräten

Zur Anzeige von Informationen zu Netzwerkschnittstellen, wie beispielsweise der Bezeichnung, dem Herstellernamen, Bus, Kapazitäten und Frequenzen, verwenden Sie den Befehl lshw mit der Option -C und dem Parameter 'network':

```
lshw -C network
```

Erfahren Sie mehr zu diesem Befehl in unserem Beitrag lshw.

## Festplattenpartitionierung

In diesem Abschnitt zur Festplattenpartitionierung lernen Sie den Befehl fdisk kennen, welcher zum Erstellen, Anzeigen, Bearbeiten und Löschen von Partitionen verwendet werden kann. Das Werkzeug unterstützt eine Vielzahl von Partitionstypen, wie beispielsweise DOS, Linux, FAT32 oder NTFS - GPT-Partitionstabellen werden dagegen nicht unterstützt.

Da fdisk Teil des Standard-Softwarepakets utils-linux-ng ist, muss es bei den meisten Linux Distributionen nicht nachinstalliert werden.

## Partitionstabellen auflisten

Mit der Option -l des Befehls ist es möglich, die Partitionstabelle von Geräten auszugeben. Die Ausgabe aller unter /proc/partitions aufgelisteten Geräte mitsamt ihrer Partitionstabelle erfolgt bei Verwendung von fdisk und der Option -l ohne jegliche zusätzliche Parameter:

```
fdisk -l
```

Wird jedoch ein Gerät als Parameter angegeben, so gibt fdisk die Partitionstabelle für genau dieses Gerät aus:

```
fdisk -l [Gerät]
```

## Partitionstabellen bearbeiten

Die Bearbeitung der Partitionstabelle eines Geräts ist mithilfe des menügesteuerten Bereichs von fdisk möglich. Dieser ist durch Eingabe von fdisk mit dem jeweiligen Gerätenamen möglich:

```
fdisk [Gerät]
```

Im folgenden werden die wichtigsten Kommandos für die Bearbeitung von Partitionstabellen genannt - einen vollständigen Überblick zu allen verfügbaren Kommandos können Sie sich durch Eingabe von 'm' ausgeben lassen.

### p - Print

Mit dem Kommando 'p' lässt sich die Partitionstabelle des aktuellen Geräts ausgeben. Die Ausgabe erfolgt ähnlich wie auch bei der direkten Verwendung der Option -l mit fdisk.

### o - Overwrite

Eine neue leere DOS- bzw. MBR-Partitionstabelle lässt sich mit dem Kommando 'o' anlegen. Beachten Sie, dass die neue Partitionstabelle eine bereits existierende Tabelle bei Bestätigung der Änderung überschreibt.

### n - New

Das Kommando 'n' wird verwendet, um neue Partitionen auf dem aktuellen Gerät zu erstellen. Ein Assistent führt Sie dabei durch die erforderlichen Schritte.

### d - Delete

Eine Partition kann mithilfe von 'd' wieder aus der Partitionstabelle eines Geräts gelöscht werden.

### a - Active

Bei DOS- bzw. MBR-Partitionstabellen lässt sich das Kommando 'a' verwenden, um das Boot-Flag einer Partition zu setzen.

### w - Write

Die Änderungen an der Partitionstabelle eines Geräts werden von fdisk erst bei Verwendung des Kommandos 'w' geschrieben und damit wirksam. Nach Eingabe des Kommandos werden alle Änderungen auf das Gerät geschrieben und fdisk beendet.

Mehr Informationen zu diesem Befehl finden Sie in unserem Beitrag fdisk.

## Aliase definieren / löschen

Mit dem Befehl alias lassen sich alternative Aufrufnamen (Aliase) für Befehlsaufrufe definieren. In diesem Abschnitt soll der Befehl in seiner grundlegenden Anwendung vorgestellt werden.

### Alias definieren

Die allgemeine Syntax zur Definition eines neuen Alias lautet folgendermaßen:

```
alias [Aliasname]='[Befehl]'
```

Der auf diese Weise definierte Alias wird mit [Aliasname] aufgerufen und führt dann den Befehl [Befehl] aus.

### Aliase anzeigen

Ein mit alias erzeugter Alias kann auch ausgegeben werden, um Einsicht in den für ihn festgelegten Befehl zu erlangen. Dazu wird dem Befehl alias einfach der jeweilige Aliasname übergeben:

```
alias [Aliasname]
```

Die Ausgabe dieses Befehls hat dann die Form: [Aliasname]='[Befehl]'. Schließlich ist es auch möglich, alle bereits definierten Aliase auszugeben. Hierzu wird der Befehl alias ohne jegliche Parameter oder Optionen verwendet:

```
alias
```

### Alias löschen

Zwar existieren so gesetzte Aliase nur bis zur Beendigung der Shell, in welcher sie definiert wurden, so ist es dennoch möglich sie auch ausdrücklich zu löschen. Hierzu wird der Befehl unalias eingesetzt:

```
unalias [Aliasname]
```

Wie Sie einen dauerhaften Alias definieren, erfahren Sie in unserem Beitrag alias.

## Extended Arguments

Der in diesem Abschnitt beschriebene Befehl xargs (extended arguments) kann in verschiedenen Situationen eingesetzt werden, um etwa große Argumentlisten mit einem beliebigen Befehl zu verarbeiten, oder sogar um eine parallelisierte Abarbeitung zugunsten einer höheren Performance zu erreichen.

Es sollen hier nur die wichtigsten Anwendungen von xargs vorgestellt werden. Für nähere Informationen zu den verfügbaren Optionen sowie verschiedene praktische Beispiele des Befehls lesen Sie auch unseren Beitrag xargs.

Der Befehl xargs lässt sich allgemein ohne jegliche Optionen einsetzen, um eine vom Standard-Input stdin eingelesene Liste von Argumenten in eine oder mehrere Kommandozeilen umzuwandeln und diese auszuführen. Standardmäßig werden dabei so viele Argumente wie möglich an das Ende des xargs' übergebenen Befehls sowie seiner initialen Argumente gehängt.

```
xargs [Optionen] [Befehl [Initiale Argumente]]
```

Der Standard-Output stdout eines Befehls (z.B. ls) ließe sich so etwa per Pipe an xargs übergeben, um dort weitere Aktionen mit den Argumenten durchzuführen:

```
[Befehl] | xargs [Optionen] [Befehl [Initiale Argumente]]
```

### Anzahl Argumente

Möchten Sie lediglich ein Argument pro Kommandozeile verwenden, so können Sie dies mit der Option -n und dem Parameter 1 spezifizieren. Jedes Argument erhält so eine eigene Kommandozeile - dies ist besonders dann wichtig, wenn der mit xargs verwendete Befehl selbst nicht die Angabe einer Argumentliste unterstützt.

```
xargs -n 1 [Befehl [Initiale Argumente]]
```

### Platzhalter ersetzen

Wenn die eingelesenen Argumente nicht an das Ende des Befehls angehängt werden sollen und stattdessen an anderer Stelle eingefügt werden müssen, so ist die Verwendung eines Platzhalters zur Ersetzung möglich. Der Platzhalter wird mit der Option -I angegeben und vor der Ausführung der Kommandozeilen jeweils durch die Argumente ersetzt. Diese Option findet etwa bei dem Kopieren vieler einzelner Dateien in ein bestimmtes Zielverzeichnis mit dem Befehl cp Anwendung.

In diesem Fall wird die Zeichenkette "{}" zur Ersetzung verwendet und durch die Option -n mit dem Parameter 1 jeweils ein Argument eingesetzt:

```
xargs -I {} [Befehl [Initiale Argumente]] {} [...]
```

### Parallelisierung

Schließlich kann xargs die Ausführung von Befehlen bzw. die Abarbeitung von Aufgaben parallelisieren: Hierbei kommt die Option -P zum Einsatz, welche eine maximale Anzahl Prozesse definiert, die xargs zu einem Zeitpunkt gleichzeitig ausführen darf. Anwendung findet diese Option etwa zur Parallelisierung von rsync bei der Übertragung vieler kleiner Dateien zwischen zwei Systemen, oder dem Download vieler kleiner Dateien aus dem Internet mit wget.

In vielen Fällen ist es sinnvoll hier die maximale Anzahl Argumente pro ausgeführter Kommandozeile bzw. ausgeführtem Prozess zu beschränken - unter Umständen wird andernfalls nur ein Prozess mit allen Argumenten auf einer Kommandozeile gestartet. Im untenstehenden Befehl wird daher die Option -n zusätzlich eingesetzt:

```
xargs -n 1 -P [Anzahl Prozesse] [Befehl [Initiale Argumente]]
```

## Anmeldungen anzeigen

Dieser Abschnitt gibt eine kurze Einführung in die grundlegende Anwendung der Befehle users und last zur Ausgabe von Benutzeranmeldungen am System. Der Befehl users kann ohne Argumente aufgerufen werden und liefert eine schnelle Übersicht zu den aktuell am System angemeldeten Benutzern - auf einer Zeile:

```
users
```

Weitere Informationen zu diesem Befehl finden Sie in unserem Beitrag users. Mit dem Befehl last stehen Ihnen dagegen viele weitere Möglichkeiten offen. So können Sie beispielsweise auch vergangene Benutzeranmeldungen einsehen. Allgemein wird der Befehl last mithilfe der folgenden Syntax aufgerufen:

```
last [Optionen] [Benutzername]
```

### Zeitraum einschränken

Mithilfe zusätzlicher Optionen lässt sich der Zeitraum einschränken, für welchen Anmeldungen ausgegeben werden. Die Option -t wird so etwa verwendet, um einen Bis-Zeitpunkt zu spezifizieren:

```
last -t [Zeit] [Benutzername]
```

Weiterhin kann auch ein Von-Zeitpunkt spezifiziert werden, welcher als Startzeitpunkt für auszugebende Anmeldungen dienen soll. Hier wird die Option -s verwendet:

```
last -s [Zeit] [Benutzername]
```

Beide Optionen können kombiniert werden, um sowohl eine Unter- und als auch eine Obergrenze für die Ausgabe festzulegen. Bei den Zeitangaben werden viele Formate unterstützt, welche im Folgenden aufgelistet sind:

| Datumsformat | Funktion u. Beispiele |
|---|---|
| YYYYMMDDhhmmss | 20210222150533 |
| YYYY-MM-DD hh:mm:ss | 2021-02-22 15:05:33 |
| YYYY-MM-DD hh:mm | 2021-02-22 15:05, verwendete Sekunden 00 |
| YYYY-MM-DD | 2021-02-22, verwendete Zeit 00:00 |
| hh:mm:ss | 15:05:33, verwendeter tag "today" |
| hh:mm | 15:05, verwendeter tag "today" u. Sekunden 00 |
| now | Zeit der Maschine |
| yesterday | Gestern, mit der Uhrzeit 00:00:00 |
| today | Heute, mit der Uhrzeit 00:00:00 |
| tomorrow | Morgen, mit der Uhrzeit 00:00:00 |
| +5min | Abweichung in Minuten |
| -days | Abweichung in Tagen |

Ausführlichere Informationen mitsamt Beispielen zu diesem Befehl finden Sie in unserem Beitrag last.

## Metadaten anzeigen

Mit dem Befehl stat, welcher in diesem Abschnitt kurz eingeführt wird, können Sie sich verschiedene Metadaten von Dateien und Verzeichnissen anzeigen lassen - beispielsweise Zeitstempel, Berechtigungen, Besitzer, Gruppe und Dateityp. Der Befehl wird im Folgenden kurz, zusammen mit seinen wichtigsten Optionen, vorgestellt.

Allgemein lässt sich der Befehl folgendermaßen - ohne zusätzliche Optionen - aufrufen:

```
stat [Datei- / Verzeichnis]
```

stat gibt hierbei viele verschiedene Informationen aus. Dazu gehören Name, Größe, Blöcke, EA Block, Typ, Gerät, Inode, Verknüpfungen, Berechtigungen sowie Zugriffs- / Modifizierungs- / Änderungs- und Erstellungszeitstempel des spezifizierten Elements.

Wird die Option -t zusätzlich angegeben, so kann eine Ausgabe in Kurzform durchgeführt werden. Die Informationen zur spezifizierten Datei bzw. dem Verzeichnis werden dann auf einer Zeile durch Leerzeichen getrennt ausgegeben.

```
stat -t [Datei- / Verzeichnis]
```

Die Ausgabe der Option -t zeigt die folgenden Informationen in der angegebenen Reihenfolge: Name, Blockgröße, gesamte Datenblöcke im Dateisystem, freie Blöcke im Dateisystem, Benutzer-ID des Besitzers, Gruppen-ID des Besitzers, Gerätenummer (hexadezimal), Dateisystem-ID (hexadezimal), Anzahl harter Verknüpfungen, Major-Gerätetyp (hexadezimal), Minor-Gerätetyp (hexadezimal), letzte Zugriffszeit, letzte Daten-Modifizierungszeit, letzte Status-Änderungszeit, Erstellungszeit (jeweils in Sekunden nach der UNIX-Epoche) und schließlich ein Hinweis auf die optimale I/O-Übertragungsgröße.

Schließlich können Sie das Format der stat Ausgabe mithilfe der Option --printf auch selbst festlegen. Die Angabe des Formats erfolgt, in Anführungszeichen eingefasst, direkt hinter der Option. Maskierungen mit dem Rückschrägstrich "" werden von stat interpretiert - so können mit "n" etwa Zeilenumbrüche eingefügt werden.

```
stat --printf [Format] [Datei- / Verzeichnis]
```

Im Format können verschiedene Formatangaben, in der Regel ein Prozentzeichen "%" mit einem Buchstaben, verwendet werden, welche von stat später durch tatsächlich ausgelesene Metainformationen ersetzt werden. Ein "%n" wird so etwa zur Ausgabe des Datei- bzw. Verzeichnisnamen verwendet. Nähere Informationen zu den verfügbaren Formatangaben und weiteren Optionen des Befehls finden Sie in unserem Beitrag stat.

## Software-RAID verwalten

Die Verwaltung - also das Erstellen, Konfigurieren, Überwachen und Löschen - von Software-RAIDs ist unter Linux / UNIX mit dem Befehl mdadm möglich. Dieser Abschnitt soll kurz die grundlegenden Anwendungen des Befehls zeigen.

Der Befehl mdadm muss, wenn er auf Ihrem System nicht bereits installiert ist, erst aus den Repositories installiert werden. Dies können Sie mit dem folgenden Befehl tun:

```
apt-get install mdadm
```

Die allgemeine Syntax von mdadm sieht wie folgt aus:

```
mdadm [Modus] [RAID-Device] [Optionen] [Component-Device(s)]
```

Der verwendete Modus gibt an, welche Aktion durchgeführt werden soll. So ist es möglich, mit dem Modus "Create", welcher mit der Option --create aufgerufen wird, Software-RAIDs zu erstellen:

```
mdadm --create [Label / Gerätename] --level=[RAID-Level] --raid-devices=[Anzahl
Partitionen] [Component-Devices]
```

Der Option --create kann dabei optional ein Label für das neue RAID-Device übergeben werden. Zudem werden mit --level einer der RAID-Level 0, 1, 4, 5, 6 oder 10 angegeben sowie mit der Option --raid-devices= die Anzahl der physischen Partitionen und darauffolgend deren Gerätenamen spezifiziert.

Ein RAID-Verbund kann mit den beiden Optionen --detail und --examine aufgelistet werden, wobei --detail sich auf den gesamten RAID-Verbund und --examine sich auf ein bestimmtes Component-Device bezieht:

```
mdadm --detail [RAID-Device]
```

```
mdadm --examine [Component-Device]
```

Der Status eines konfigurierten RAID-Device lässt sich unter /proc/mdstat auslesen - beispielsweise mit cat:

```
cat /proc/mdstat
```

Mit der Option --stop kann ein bestehendes RAID-Device schließlich deaktiviert und verwendete Ressourcen freigegeben werden - physisch bleibt es dabei weiter bestehen:

```
mdadm --stop [RAID-Device]
```

Soll ein RAID komplett entfernt werden, so müssen für alle beteiligten Component-Devices mit der Option --zero-superblock die Superblöcke nullgesetzt werden:

```
mdadm --zero-superblock [Component-Device]
```

Nähere Informationen zu Software-RAIDs, weiteren Optionen des Befehls sowie seiner Anwendung finden Sie in unserem Beitrag Software RAID MDADM.

## Festplattenformatierung

Bei der Formatierung wird das Dateisystem eines Datenträgers festgelegt, wobei zuvor in der Regel mindestens eine Partition angelegt werden muss. In diesem Abschnitt lernen Sie hierzu den Befehl mkfs kennen, welcher unter UNIX / Linux pro Dateisystem in einer eigenen Version existiert. Die Befehle beginnen dabei alle mit dem Präfix "mkfs." und sind gefolgt von dem Namen des Dateisystems. Für die Formatierung mit ext4 wird so etwa der Befehl "mkfs.ext4" verwendet.

Die Syntax ist für alle mkfs-Befehle grundsätzlich gleich:

```
mkfs.[Dateisystemname] [Optionen] [Gerät]
```

Während in den meisten Fällen keine zusätzlichen Optionen angegeben werden müssen, da die Standardwerte bereits sinnvoll eingestellt sind, ist die Angabe des zu formatierenden Geräts zwingend notwendig. Beispielhaft könnte die erste Partition eines Datenträgers mit dem Gerätenamen "/dev/sdb" wie folgt mit einem ext3-Dateisystem formatiert werden:

```
mkfs.ext3 /dev/sdb1
```

Wenn benötigt, sind der Manpage des jeweiligen mkfs-Befehls dateisystemspezifisch verfügbare Optionen zu entnehmen.

Mehr Informationen zu dem Befehl finden Sie in unserem Beitrag mke2fs.

## Datei-/Verzeichniskapazitäten anzeigen

Die Anzeige der Kapazität gegebener Dateien oder Verzeichnisse ist mit dem Befehl du möglich. Die allgemeine Syntax des Befehls lautet:

```
du [Optionen] [Verzeichnis(se) / Datei(en)]
```

Wird der Befehl alleinstehend, ohne Angabe von Dateien oder Verzeichnissen, ausgeführt, dann werden rekursiv die Kapazitäten der Unterverzeichnisse des aktuellen Arbeitsverzeichnisses ausgegeben. Analog erfolgt die Ausgabe der Unterverzeichnisse eines oder mehrerer spezifizierter Verzeichnisse. Werden Dateien angegeben, dann gibt du die Kapazitäten von diesen aus.

Sollen neben den Kapazitäten von Unterverzeichnissen auch solche von enthaltenen Dateien ausgegeben werden, dann verwenden Sie die Option -a:

```
du -a [Verzeichnis(se) / Datei(en)]
```

Benötigen Sie die Gesamtsumme aller von du ausgegebenen Dateien bzw. Verzeichnisse, dann können Sie diese unter Verwendung der Option -c zusätzlich von du berechnen und am Ende (unter der Bezeichnung "insgesamt") ausgeben lassen:

```
du -c [Verzeichnis(se) / Datei(en)]
```

Schließlich ist es möglich, die Kapazitäten in menschenlesbarem Format - beispielsweise 4K, 2M oder 4,9G auszugeben:

```
du -h [Verzeichnis(se) / Datei(en)]
```

Wenn Sie nähere Informationen zum diesem Befehl, seinen weiteren Optionen und zu seiner Anwendung benötigen, dann lesen Sie auch unseren Beitrag du.

## Dateisystemkapazitäten anzeigen

Mit dem Befehl df ist es möglich, die verfügbaren und belegten Kapazitäten von eingehängten Dateisystemen anzuzeigen. Allgemein wird der Befehl wie folgt angewandt:

```
df [Optionen] [Verzeichnis(se) / Datei(en)]
```

Die Angabe von Verzeichnissen bzw. Dateien ist optional - standardmäßig, also wenn keine Verzeichnisse oder Dateien übergeben wurden, werden die Kapazitäten aller derzeit eingehängten Dateisysteme angezeigt. Wurden jedoch Verzeichnisse bzw. Dateien angegeben, dann werden jeweils die Kapazitäten der Dateisysteme ausgegeben, die die übergebenen Elemente enthalten.

Die Ausgabe hat das Format einer Tabelle mit den Spalten "Dateisystem", "1K-Blöcke", "Benutzt", "Verfügbar", "Verw%" und "Eingehängt auf". Es werden also pro Dateisystem die Gesamtgröße ("1K-Blöcke"), der belegte ("Benutzt") und verfügbare Platz sowie auch der Einhängepunkt ausgegeben. Der Platz wird dabei standardmäßig in 1K-Blöcken ausgegeben.

Da die Anzeige in 1K-Blöcken für einen Menschen eher umständlich lesbar ist, kann mit der Option -h auch auf ein besser lesbares Format (je nach Größe beispielsweise mit den Einheiten Mebibyte (M) oder Gibibyte (G)) umgestellt werden:

```
df -h [Verzeichnis(se) / Datei(en)]
```

Eine weitere nützliche Option ist -T. Diese führt dazu, dass df die Dateisystemtypen (z.B.: "ext3", "ext4") jeweils in einer zusätzlichen Spalte ausgibt:

```
df -T [Verzeichnis(se) / Datei(en)]
```

Weitere Informationen zum diesem Befehl finden Sie in unserem Beitrag df.

## Dateisysteme erstellen

Der Befehl mke2fs ermöglicht die Erstellung von ext2-, ext3- oder ext4-Dateisystemen - üblicherweise auf einer Festplattenpartition. Beim Aufruf von mke2fs ist die Angabe eines Geräts, auf dem das Dateisystem angelegt werden soll, notwendig:

```
mke2fs [Optionen] [Gerät]
```

Werden keine Optionen übergeben, so verwendet mke2fs die in der Konfigurationsdatei /etc/mke2fs.conf festgelegten Standardwerte für das neue Dateisystem. Eine der wichtigsten Optionen des Befehls ist -t. Diese wird verwendet, um "ext2", "ext3" oder "ext4" als Dateisystemtyp für das neue Dateisystem festzulegen:

```
mke2fs -t [ext2|ext3|ext4] [Gerät]
```

Soll das Dateisystem mit einem ext3-Journal erstellt werden, so geben Sie die Option -j an. Wenn nicht anderweitig spezifiziert nutzt mke2fs dann Standard-Journal-Parameter für die Erstellung des Journals.

```
mke2fs -j [Gerät]
```

Schließlich sei hier noch die Option -b genannt, welche die Festlegung der Blockgröße des neuen Dateisystems erlaubt. Gültige Werte sind 1024, 2048 und 4096 Bytes pro Block. Wird die Option nicht verwendet, dann bestimmt mke2fs die Blockgröße heuristisch auf Grundlage der Dateisystemgröße sowie der erwarteten Nutzung.

```
mke2fs -b [Blockgröße] [Gerät]
```

Möchten Sie mehr zu diesem Befehl erfahren, lesen Sie gerne auch unseren Beitrag mke2fs.

## Dateien aufteilen

Dateien lassen sich mithilfe des Befehls split beliebig in kleinere Dateien aufteilen - etwa um die Verarbeitung von großen Dateien in mehrere gleich große "Blöcke" aufzuteilen. Auch bei der Datensicherung / Verteilung könnten Dateien mit split aufgeteilt und zu einem späteren Zeitpunkt wieder zusammengesetzt werden - dies ist beispielsweise dann wichtig, wenn die Größe einer Datei die maximale Dateigröße eines Zieldateisystems überschreitet. Allgemein wird der Befehl wie folgt verwendet:

```
split [Optionen] [Datei [Präfix]]
```

split teilt die übergebene Datei standardmäßig so auf, dass jede Teildatei maximal 1000 Zeilen bzw. Datensätze beinhaltet. Benannt werden die Dateien automatisch nach dem Schema "[Präfix]aa", "[Präfix]ab", "[Präfix]ac" etc. Der zweistellige Suffix wird also inkrementiert, als handele es sich um ein Stellenwertsystem zur Basis 26. Der Standard-Präfix ist "x". Sollen numerische Suffixe verwendet werden, so spezifizieren Sie zusätzlich die Option --d:

```
split --d [Datei [Präfix]]
```

Möchten Sie anstelle der standardmäßigen 1000 Zeilen / Datensätze pro Teildatei eine andere Anzahl spezifizieren, so können Sie dies mit der Option -l tun:

```
split -l [Anzahl] [Datei [Präfix]]
```

Wird die Option -b angegeben, so trennt split nicht anhand der Anzahl Zeilen / Datensätze pro Teildatei, sondern mithilfe der Anzahl Bytes. Es wird ein Integer angegeben, welcher die maximale Anzahl Bytes pro Teildatei, also die jeweilige Größe in Bytes, festlegt:

```
split -b [Größe] [Datei [Präfix]]
```

Informationen zu den weiteren verfügbaren Optionen und Beispiele zu diesem Befehl finden Sie in unserem Beitrag split.

## ATA-Laufwerksparameter auslesen

Dieser Abschnitt gibt eine kurze Einführung in die Anwendung des Befehls hdparm zum Lesen von Parametern von ATA-Laufwerken. Es ist zu beachten, dass es sich um einen Befehl für fortgeschrittene Benutzer handelt, da er bei unsachgemäßer Verwendung auch zu Instabilitäten und unumkehrbaren Schäden an den Laufwerken führen kann. Die Syntax des Befehls lautet allgemein folgendermaßen:

```
hdparm [Optionen] [Gerät(e)]
```

Bei Ausführung ohne jegliche Optionen gibt hdparm grundlegende Daten zum angegebenen Laufwerk aus. Dazu gehören etwa DMA, multiple-sector-count, read-only, read-ahead und die Geometrie.

Detailliertere Informationen, die zur Boot- bzw. Konfigurationszeit von den Kernel-Treibern gespeichert wurden, lassen sich mit der Option -i ausgeben. Es werden somit viele weitere Parameter, wie beispielsweise die Modell-Bezeichnung, Firmware-Version, Seriennummer und die Kapazität des Laufwerkscache ausgegeben:

```
hdparm -i [Gerät(e)]
```

Schließlich ist die Ausgabe noch detaillierterer Informationen, welche bei Befehlsausführung direkt aus dem Laufwerk ausgelesen werden, mit der Option -I möglich. Die Ausgabe ist in Abschnitte eingeteilt und beinhaltet unter anderem Informationen zum ATA-Gerät selbst, zu Standards, der Konfiguration (Configuration), Kapazitäten (Capabilities) und zu aktivierten bzw. deaktivierten Funktionen (Commands/features):

```
hdparm -I [Gerät(e)]
```

Weitere Informationen, welche etwa auch die Anwendung des Befehls zur Durchführung von System-Cache-Tests umfassen, finden Sie in unserem Beitrag hdparm.

## Kernel-Meldungen auslesen

Mithilfe des Befehls dmesg können Meldungen aus dem Kernel-Ringpuffer ausgelesen werden. Der Befehl hilft so bei der Fehlersuche im System - beispielsweise wenn neue Hardware nicht korrekt erkannt wurde. Die allgemeine Syntax lautet folgendermaßen:

```
dmesg [Optionen]
```

Bei Ausführung ohne jegliche Optionen gibt der Befehl die Meldungen des Kernel-Ringpuffers standardmäßig ungefiltert und voll umfänglich aus. Zum Extrahieren der jeweils wichtigen Informationen ist es daher sinnvoll, dmesg mithilfe einer Pipe mit einem anderen Befehl zu kombinieren (beispielsweise mit grep), oder die Ausgabe mit Optionen zu beschränken.

Verfügbare Optionen zu diesem Befehl sowie auch einige Beispiele zur Anwendung finden Sie in unserem Beitrag dmesg.

## Angeschlossene Partitionen auslesen

Informationen zu den an einem System angeschlossenen Partitionen lassen sich mit dem Befehl blkid auslesen. Unter anderem können mit blkid Gerätedatei, UUID, Name und Dateisystemtyp ermittelt werden. Es werden dabei Root-Rechte benötigt. Die allgemeine Syntax von blkid lautet:

```
blkid [Optionen] [Gerätedatei]
```

Bei Ausführung ohne jegliche Optionen sowie ohne Angabe einer Gerätedatei gibt blkid alle angeschlossenen Partitionen mitsamt der entsprechenden Informationen zeilenweise aus. Wird dagegen die Gerätedatei einer Partition angegeben, so zeigt blkid nur die Informationen dieser Partition.

Weitere Informationen zu diesem Befehl finden Sie in unserem Beitrag blkid.

## Dateikomprimierung

Dateien lassen sich mit verschiedenen Werkzeugen komprimieren: Ein unter UNIX- / Linux-Betriebssystemen weit verbreitetes Programm hierfür ist gzip. Der Befehl ermöglicht die Komprimierung einzelner Dateien in sogenannte .gz-Archive, wobei Dateiattribute erhalten bleiben. Allgemein lässt sich gzip folgendermaßen aufrufen:

```
gzip [Optionen] [Datei(en)]
```

Mehrere Dateien können angegeben werden, werden von gzip dann jedoch jeweils in ein eigenes Archiv komprimiert. Die Originaldatei wird nach dem Komprimieren gelöscht und durch das komprimierte Archiv ersetzt. Der Dateiname letzteren Archivs entspricht dabei dem Namen der Originaldatei, ergänzt mit der Endung ".gz". Anstelle von Dateinamen kann dem Befehl auch ein Minuszeichen "-" übergeben werden, um den Standard-Input stdin zu komprimieren und an den Standard-Output stdout weiterzugeben.

Möchten Sie die Originaldateien entgegen des Standardverhaltens von gzip behalten, so können Sie zusätzlich die Option -k spezifizieren:

```
gzip -k [Datei(en)]
```

Schließlich ist es natürlich auch möglich, eine mit gzip komprimierte Datei wieder zu entpacken. Hierzu wird die Option -d spezifiziert und die entsprechende Archivdatei übergeben. Es können auch mehrere Archivdateien im Rahmen eines Befehlsaufrufs entpackt werden:

```
gzip -d [Datei(en)]
```

Nähere Informationen zu gzip selbst, sowie weitere Optionen - etwa zur Einstellung der Komprimierungsstufe oder zum rekursiven Komprimieren aller in Verzeichnissen enthaltenen Dateien - können Sie in unserem Beitrag gzip nachlesen.

## SHA-Prüfsummen bilden

SHA-Prüfsummen, die etwa zur Überprüfung der Integrität einer Datei verwendet werden, können Sie mit dem Befehl shasum berechnen. Der Befehl bietet dabei die Wahl zwischen den Algorithmen SHA-1, SHA-224, SHA-256, SHA-384 und SHA-512. Die allgemeine Syntax lautet:

```
shasum [Optionen] [Datei(en)]
```

Standardmäßig berechnet shasum die SHA-1-Prüfsumme der übergebenen Datei. Es können auch mehrere Dateien übergeben werden - die Prüfsummen werden dann jeweils untereinander ausgegeben. Alternativ ist es auch möglich, den Befehl vom Standard-Input stdin lesen zu lassen. Dazu wird entweder keine Datei oder ein Minus-Zeichen "-" übergeben. Die Ausgabe der berechneten Prüfsummen erfolgt jeweils pro Datei auf einer Zeile und beinhaltet hintereinander die Prüfsumme selbst sowie den entsprechenden Dateinamen.

Wie bereits zu Beginn angekündigt unterstützt shasum noch viele weitere Algorithmen. Diese können mithilfe der Option -a ausgewählt werden. Der Algorithmus wird jeweils ohne den Präfix "SHA-" übergeben, sodass die Wahl zwischen "1" (SHA-1), "224" (SHA-224), "256" (SHA-256), "384" (SHA-384) und "512" (SHA-512) besteht:

```
shasum -c [Algorithmus] [Datei(en)]
```

Nähere Informationen zu diesem Befehl, seinen übrigen Optionen und insbesondere der Möglichkeit zum Abgleich von Prüfsummen, entnehmen Sie unserem Beitrag shasum.

## Wurzelverzeichnis wechseln

Wenn Sie einen Befehl oder eine interaktive Shell mit einem bestimmten anderen Wurzelverzeichnis ausführen möchten, dann können Sie dazu den Befehl chroot verwenden. Der Befehl eignet sich so beispielsweise für den Wechsel in die Bash eines anderen, eingehängten Systems, um eine Neuinstallation des GRUB-Bootloaders durchzuführen. Die allgemeine Syntax von chroot lautet folgendermaßen:

```
chroot [Wurzelverzeichnis] [Befehl [Argument(e)]]
```

Während die Angabe eines Wurzelverzeichnisses notwendig ist, ist die Übergabe eines auszuführenden Befehls optional. Bei Ausführung von chroot ohne einen Befehl, wird eine interaktive Shell mit dem spezifizierten Wurzelverzeichnis gestartet. Es ist zu beachten, dass für die Nutzung von chroot standardmäßig root-Berechtigungen erforderlich sind.

Optionen werden in den meisten Anwendungsfällen von chroot nicht benötigt. Wenn Sie jedoch näheres zum Befehl und seinen Optionen erfahren möchten, dann lesen Sie auch unseren Beitrag chroot.

### Speichermedien und Partitionen anzeigen

Informationen zu den am System angeschlossenen Speichermedien und Partitionen können mithilfe des Befehls blkid abgerufen werden. Auszulesen sind so etwa die Gerätedatei, die UUID, der Name und der Dateisystemtyp. Nützlich ist der Befehl beispielsweise zur Ermittlung des Gerätedateinamens einer Partition, wenn diese mit dem Befehl mount eingehängt werden soll. Die Auflistung aller angeschlossenen Geräte ist durch Aufruf des Befehls ohne jegliche Argumente möglich:

```
blkid
```

Alternativ kann auch eine Gerätedatei übergeben werden, um Informationen nur für ein bestimmtes Gerät auszugeben:

```
blkid [Gerätedatei]
```

Der Befehl bietet auch eine Reihe an Optionen an, um die Ausgabe weiter zu beeinflussen. Nähere Informationen zum Befehl sowie einigen der wichtigsten dieser Optionen finden Sie in unserem Beitrag blkid.

## Netzwerke

### Konfiguration und Statistiken

In diesem Abschnitt lernen Sie einige der wichtigsten Befehle für die Konfiguration von Netzwerken sowie für die Ausgabe relevanter Informationen bezüglich Erreichbarkeit etc. kennen.

### ip - Befehl für die Netzwerkkonfiguration

Der Befehl ip ist Teil des Pakets iproute2 und ersetzt den ifconfig-Befehl der net-tools.

```
ip [Optionen]
```

Der Befehl kann mit vielen verschiedenen Optionen ausgeführt werden, um Netzwerkschnittstellen abzufragen oder zu konfigurieren.

Mit der Option link ist es möglich, den Status aller Schnittstellen abzufragen:

```
ip link
```

Wollen Sie den Status der Schnittstellen genauso abfragen, wie dies mit ifconfig möglich war, fügen Sie einfach die Option -s zusätzlich mit an:

```
ip -s link
```

Weiterhin können mit link Einstellungen auf Ethernet-Ebene geändert werden. Mit dem folgenden Befehl können Sie die MAC-Adresse ändern:

```
ip link set dev [Gerät] address [MAC-Adresse]
```

Außerdem können Sie eine Schnittstelle aktivieren (up) bzw. deaktivieren (down):

```
ip link set [up/down] [Gerät]
```

Mit der Option addr können Sie alle verfügbaren Netzwerkschnittstellen mitsamt ihrer IP-Adressen abfragen, anzeigen und neue IP-Adressen hinzufügen. Mit addr show werden Ihnen die Daten übersichtlich ausgegeben:

```
ip addr show
```

Zudem können Sie mit addr add IP-Adressen zuweisen:

```
ip addr add [IP-Adresse] dev [Gerät]
```

Mit del können Sie eine IP-Adresse wieder entfernen:

```
ip addr del [IP-Adresse] dev [Gerät]
```

## ping - Erreichbarkeit überprüfen

Mit dem Befehl ping können Sie die Erreichbarkeit einer Netzwerkschnittstelle überprüfen. Die Syntax lautet:

```
ping [Optionen] [Adresse]
```

Der Befehl sendet bei Ausführung Anfragen an die "angepingte" Netzwerkschnittstelle, welche diese bei Eingang beantwortet. Mit der Tastenkombination [Ctrl+c] können Sie die Ausführung des Befehls wieder beenden.

## nslookup - DNS-Abfragen durchführen

Der Befehl nslookup wird verwendet, um DNS-Abfragen durchzuführen. Das Werkzeug ist nicht unter allen Linux-Distributionen standardmäßig vorinstalliert. Unter Debian erfolgt die Installation des hierfür benötigten Pakets "dnsutils" folgendermaßen:

```
apt-get install dnsutils
```

nslookup kann sowohl in einem interaktiven, als auch in einem nicht-interaktiven Modus verwendet werden. Im Folgenden wird der nicht-interaktive Modus eingesetzt:

Zum Auflösen der IP-Adresse eines Domainnamen wird der folgende Befehl verwendet:

```
nslookup [Domainname]
```

Eine sogenannte Reverse-Lookup-Abfrage ist dagegen durch Angabe einer IP-Adresse möglich:

```
nslookup [IP-Adresse]
```

Der MX-Eintrag einer Domain wird durch Hinzufügen der Option -query mit dem Parameter "mx" ausgegeben. Analog können auch der NS-Eintrag ("ns"), der SOA-Eintrag ("soa"), oder alle Informationen zu einer Domain ("any") angezeigt werden.

```
nslookup -query=mx [Domainname]
```

```
nslookup -query=ns [Domainname]
```

```
nslookup -query=soa [Domainname]
```

```
nslookup -query=any [Domainname]
```

## traceroute - Datenpaket verfolgen

traceroute ermöglicht Ihnen die Nachverfolgung eines Datenpakets auf allen seinen Zwischenstationen. Dazu werden kleine Datenpakete an den gewünschten Host gesendet. Die Syntax des Befehls lautet folgendermaßen:

```
traceroute [Optionen] [Adresse]
```

Optionen werden nicht benötigt.

## ss - Socket Statistiken anzeigen

Mit ss können Sie Statistiken zu PACKET-, TCP-, UDP-, DCCP-, RAW- und Unix-Domain-Sockets anzeigen:

```
ss [Optionen]
```

Die verschiedenen verfügbaren Optionen dienen der Anzeige diverser Informationen. Die Option -l (listening) wird beispielsweise verwendet, um nur "horchende" Sockets anzuzeigen:

```
ss -l
```

Die Option -t (TCP) zeigt alle TCP Verbindungen an:

```
ss -t
```

-u (UDP) zeigt ausschließlich UDP Verbindungen an:

```
ss -u
```

Schließlich werden mit der Option -x (UNIX) nur UNIX Verbindungen angezeigt:

```
ss -x
```

Weitere Informationen finden Sie auf unserem Beitrag Die wichtigsten Netzwerkbefehle für Anfänger.

## SSH

Der in diesem Abschnitt beschriebene Befehl ssh kann auf fast allen Linux-/UNIX-Betriebssystemen für den sicheren Zugriff auf einen SSH-Server oder ein Remote-System eingesetzt werden. Der Befehl wird für die Anmeldung sowie die Ausführung von Befehlen verwendet.

### Login

Der Login, welcher zum Verbindungsaufbau mit einer Remote-Maschine erforderlich ist, erfolgt zunächst ganz einfach durch Eingabe des Befehls ssh mit der Adresse des jeweiligen Systems:

```
ssh [Adresse]
```

Wenn Sie bei der Anmeldung einen Benutzer spezifizieren möchten, dann können Sie den Benutzernamen einfach als Präfix und durch ein '@' von der Adresse getrennt anhängen:

```
ssh [Benutzer]@[Adresse]
```

Auch die Verwendung der Option -l ist hier möglich:

```
ssh -l [Benutzer] [Adresse]
```

Es ist zu beachten, dass bei dem ersten Verbindungsaufbau zu einem SSH Server eine besondere Abfrage stattfindet, bei welcher der Server und ein sogenannter Host-Schlüssel von Ihnen bestätigt werden müssen. Beides wird nach Ihrer Bestätigung in eine Liste bekannter Hosts unter "~/.ssh/known_hosts" abgelegt. Aus diesem Grund müssen Sie diese Bestätigung auch nur bei dem ersten Verbindungsaufbau durchführen - wird der Schlüssel ein weiteres Mal abgefragt, sollten Sie sicher sein, dass der Schlüssel serverseitig entsprechend geändert wurde.

### Befehle ausführen

Wenn Sie einen Befehl auf einer Remote-Maschine ausführen möchten, können Sie diesen auch direkt dem ssh Befehl übergeben:

```
ssh [Adresse] [Befehl]
```

Bei einer solchen Verwendung des ssh Befehls wird nach der Authentifizierung lediglich der Befehl auf der Remote-Maschine ausgeführt und die Ausgaben in der lokalen Befehlszeile ausgegeben. Im Anschluss bleibt der Nutzer nicht angemeldet - die SSH-Sitzung wird stattdessen sofort nach Befehlsausführung beendet.

Mehr Informationen zu diesem Befehl finden Sie auf unserem Beitrag ssh.

## SCP

Der Befehl scp kann auf Linux-/UNIX-Betriebssystemen verwendet, um Dateien über eine sichere, verschlüsselte Netzwerkverbindung zu kopieren. Neben einem SCP-Client ist dazu - ähnlich wie bei SSH - auch ein entsprechender Server notwendig.

### Dateien kopieren

Zum Kopieren von Dateien mit dem Befehl scp wird grundsätzlich die folgende Syntax verwendet:

```
scp [Optionen] [Pfad Quelle] [Pfad Ziel]
```

Bei den Pfadangaben sind im Falle lokaler Pfade keine Besonderheiten zu beachten. Lediglich bei Pfadangaben auf Remote-Maschinen muss eine spezielle Syntax berücksichtigt werden. Der jeweilige Hostname wird durch einen Doppelpunkt ':' getrennt links an den Pfad angehängt. Die Angabe eines Benutzernamen ist optional ebenfalls möglich, indem dieser mit einem '@' getrennt links an den Hostnamen angehängt wird. SCP ermöglicht zudem auch das Kopieren zwischen zwei Remote-Hosts. Die resultierende Syntax lautet:

scp [Optionen] [Benutzer]@[Host 1]:[Pfad] [Benutzer]@[Host 2]:[Pfad]

Im Folgenden werden einige wichtige Optionen des Befehls scp kurz beschrieben.

### -3 - Lokaler Rechner als Zwischenstation

Mit der Option -3 wird die Standardeinstellung, dass Daten beim Kopieren zwischen zwei Hosts direkt zwischen diesen versendet werden, überschrieben. Der Kopiervorgang findet somit über den lokalen Rechner statt.

```
scp -3 [Benutzer]@[Host 1]:[Pfad] [Benutzer]@[Host 2]:[Pfad]
```

### -l - Bandbreite limitieren

Die Bandbreite einer Datenübertragung mit dem Befehl scp kann mithilfe der Option -l limitiert werden. Die Angabe erfolgt in KBit/s:

scp -l [Benutzer]@[Host 1]:[Pfad] [Benutzer]@[Host 2]:[Pfad]

### -P - Port festlegen

Der verwendete Port, welcher bei scp standardmäßig Port 22 ist, kann mit der Option -P beliebig spezifiziert werden:

scp -P [Portnummer] [Benutzer]@[Host 1]:[Pfad] [Benutzer]@[Host 2]:[Pfad]

### -r - Verzeichnisse rekursiv kopieren

Zum rekursiven Kopieren von Verzeichnissen, verwenden Sie die Option -r:

scp -r [Benutzer]@[Host 1]:[Pfad] [Benutzer]@[Host 2]:[Pfad]

### -v - Debugging

Die Option -v kann schließlich bei der Fehlerbehebung helfen - sie sorgt dafür, dass jegliche auftretende Debugging-Nachrichten in die Ausgabe geschrieben werden:

scp -v [Benutzer]@[Host 1]:[Pfad] [Benutzer]@[Host 2]:[Pfad]

Weitere Informationen, wie Sie beispielsweise den scp-Server installieren können, finden Sie auf unserem Beitrag scp.

## FTP- / HTTP-Download

Für den Download von Dateien direkt von FTP- oder HTTP bzw. HTTPS-Servern wird Ihnen in diesem Abschnitt der Befehl wget vorgestellt. Der Befehl kann neben der einfachen Verwendung als Downloadmanager auch in Shell-Skripten verwendet, oder ohne Benutzerinteraktion im Hintergrund ausgeführt werden.

Die allgemeine Syntax für das Herunterladen von Dateien lautet folgendermaßen:

```
wget [Optionen] [URL(s)]
```

Wird der Befehl ohne zusätzliche Optionen verwendet, so werden die angegebenen URLs heruntergeladen und im aktuellen Arbeitsverzeichnis abgelegt.

Einige wichtige Optionen werden im Folgenden kurz vorgestellt.

### -t - Anzahl Versuche spezifizieren

Kommt es während des Herunterladens zu Verbindungsabbrüchen, versucht wget standardmäßig bis zu 20 Mal, die Verbindung wiederherzustellen. Wird diese maximale Anzahl der Versuche überschritten, so wird der Vorgang abgebrochen.

Mit der Option -t können Sie genau festlegen, wie oft wget versuchen soll, die Verbindung wiederherzustellen:

```
wget -t [Anzahl Versuche] [URL(s)]
```

### -c - Download fortsetzen

Wurde ein Download aus jeglichen Gründen abgebrochen, und sind die lokalen Dateien des ersten Vorgangs sind noch auf dem Dateisystem vorhanden, so kann der Download mit der Option -c fortgesetzt werden:

```
wget -c [URL(s)]
```

### -i - URLs aus Datei auslesen

Neben der direkten Eingabe von URLs als Befehlsparameter unterstützt wget auch das Auslesen von URLs aus einer lokalen oder externen Datei:

```
wget -i [Dateipfad] [URL(s)]
```

Die URLs werden in der Datei untereinander aufgelistet, sodass für jede URL eine Zeile verwendet wird. Der Option kann auch der Parameter "-" übergeben werden, um die URLs aus dem Standard-Input (stdin) zu lesen. Die Angabe von URLs direkt in der Befehlszeile ist gleichzeitig möglich - in diesem Fall werden zunächst die URLs der Befehlszeile und anschließend die in der übergebenen Datei befindlichen URLs heruntergeladen.

### -O - Ausgabedatei spezifizieren

Bei Angabe einer Ausgabedatei mithilfe der Option -O werden die Inhalte der heruntergeladenen Dateien miteinander konkateniert und anschließend in eine gemeinsame Datei mit dem von Ihnen festgelegten Namen/Pfad geschrieben:

```
wget -O [Dateipfad] [URLs]
```

Durch Angabe des Parameters "-" ist es möglich, die Inhalte im Standard-Output (stdout) auszugeben.

### -P - Ausgabeverzeichnis spezifizieren

Mithilfe der Option -P lässt sich das Verzeichnis-Präfix auf den angegebenen Pfad festlegen, um alle heruntergeladenen Dateien unter diesem abzulegen.

```
wget -P [Verzeichnispfad] [URLs]
```

Der Standardwert "." gibt das aktuelle Arbeitsverzeichnis an.

### -b - Hintergrundausführung

Schließlich kann wget mithilfe der Option -b im Hintergrund ausgeführt werden. Nach Ausführung wird die Befehlszeile wieder freigegeben und die Ausführung anderer Aufgaben ist möglich. Alle erzeugten Ausgaben werden in die sogenannte wget-Log umgeleitet.

```
wget -b [URL(s)]
```

Mehr Informationen zu diesem Befehl finden Sie auf unserem Beitrag wget.

## Rsync

Der Befehl rsync, welcher zum Kopieren von Dateien dient, wird in diesem Abschnitt für einen kleinen Überblick vorgestellt. Ein Kopiervorgang ist sowohl lokal, als auch auf ein anderes System bzw. von einem anderen System möglich. Besonderheit des Befehls ist sein Delta-Transfer-Algorithmus, welcher nur die Unterschiede zwischen Quelldateien und vorhandenen Zieldateien überträgt - der Befehl ist damit gut für "Backups" und "Spiegelungen" geeignet. Die allgemeine Syntax des Befehls lautet folgendermaßen:

```
rsync [Optionen] [Quell-Verzeichnis(se)] [Ziel-Verzeichnis]
```

Wird nur ein Parameter ohne Optionen übergeben, also nur das Ziel-Verzeichnis, dann wird das aktuelle Arbeitsverzeichnis als Quellverzeichnis verwendet. Meist wird rsync zumindest zusammen mit der Option -a verwendet, um unter anderem Rechte und Eigentümer der Quelldateien zu übernehmen. Letztlich fasst die Option mehrere Einzeloptionen zusammen (-r, -l, -p, -t, -g, -o, -D):

```
rsync -a [Quell-Verzeichnis(se)] [Ziel-Verzeichnis]
```

Sollen Dateien übersprungen werden, die in der Quelle neuer sind als im Ziel, so muss die Option -u spezifiziert werden:

```
rsync -u [Quell-Verzeichnis(se)] [Ziel-Verzeichnis]
```

Wenn Sie möchten, dass Dateien im Ziel gelöscht werden, nachdem diese in der Quelle gelöscht wurden, dann verwenden Sie zusätzlich die Option --delete:

```
rsync --delete [Quell-Verzeichnis(se)] [Ziel-Verzeichnis]
```

Wie Sie die Vielseitigkeit dieses Befehls als Kopierwerkzeug nutzen können, erfahren Sie in unserem Beitrag rsync.

## Routing-Tabelle verwalten

Dieser Abschnitt stellt den Befehl route für die Anzeige und Bearbeitung der Routing-Tabelle unter Linux / UNIX kurz vor. Der Befehl ist Teil des Pakets net-tools, welches nicht auf allen Linux- / UNIX-Betriebssystemen standardmäßig installiert ist, und muss daher gegebenenfalls nachinstalliert werden. Folgender Befehl zeigt die Installation des Pakets mit apt:

```
apt-get install net-tools
```

## Routing-Tabelle anzeigen

Die Anzeige der Routing-Tabelle erfolgt durch Ausführung des Befehls route ohne jegliche Optionen und Parameter. Die ausgegebene Routing-Tabelle hat dann die Spalten Ziel (Destination), Router (Gateway), Genmask, Flags, Metric, Ref, Use und Iface.

```
route
```

Alternativ kann auch die Option -n angefügt werden, wenn die Auflösung von IP-Adressen in symbolische Hostnamen deaktiviert werden soll:

```
route -n
```

## Routing-Tabelle bearbeiten

Die Bearbeitung der Routing-Tabelle erfolgt mit "route add" und "route del".

Im Folgenden ist zunächst die Syntax von "route add" gezeigt. Es müssen das Ziel, eine Netzmaske sowie die Adresse des Routers (Gateway), über den das Ziel erreicht werden soll, angegeben werden. Je nachdem ob es sich bei dem Ziel um einen einzelnen Rechner handelt, oder ein ganzes Netzwerk, wird zudem eine der beiden Optionen -host (Einzelner Rechner) bzw. -net (Ganzes Netzwerk) gewählt.

```
route add -[host|net] [Ziel] netmask [Netzmaske] gw [Gateway]
```

Soll ein Eintrag aus der Routing-Tabelle gelöscht werden, so ist die Syntax sehr ähnlich. Lediglich "add" wird nun zu "del". Für die Parameter werden jeweils die Werte des zu löschenden Eintrags verwendet:

```
route del -[host|net] [Ziel] netmask [Netzmaske] gw [Gateway]
```

Weitere Informationen zu diesem Befehl finden Sie in unserem Beitrag route.

Related Terms:

- Term: Betriebssystem
- Term: TTY
- Term: Syntax
- Term: Root-Benutzer
- Term: SSH
- Term: Rekursion
- Term: String
- Term: Regex
- Term: Wildcard
- Term: Dateisystem
- Term: Datenbank
- Term: Cronjob
- Term: Systemd
- Term: RFC
- Term: Cache
- Term: DNS
- Term: Authentifizierung
- Term: SCP
- Term: FTP
- Term: Konkatenation
- Term: ISO
- Term: TCP
- Term: UDP