

# Im Dateisystem navigieren <sup>1</sup>

## Themen

- Lernen über das Linux-Dateisystem
- Auflisten von Datei- und Verzeichniseigenschaften
- Erstellen von Dateien und Verzeichnissen
- Auflisten und Ändern von Berechtigungen und Besitz
- Kopieren und Verschieben von Dateien

Das Linux-Dateisystem ist die Struktur, in der alle Informationen auf Ihrem Computer gespeichert sind. Tatsächlich ist eine der charakteristischen Eigenschaften der UNIX-Systeme, auf denen Linux basiert, dass fast alles, was Sie auf Ihrem System identifizieren müssen (Daten, Befehle, symbolische Links, Geräte und Verzeichnisse), durch Elemente in den Dateisystemen repräsentiert wird. Zu wissen, wo sich die Dinge befinden, und zu verstehen, wie man sich im Dateisystem von der Shell aus bewegt, sind wichtige Fähigkeiten in Linux.

In Linux sind Dateien in einer Hierarchie von Verzeichnissen organisiert. Jedes Verzeichnis kann Dateien sowie andere Verzeichnisse enthalten. Sie können auf jede Datei oder jedes Verzeichnis entweder über einen absoluten Pfad (zum Beispiel `/home/joe/myfile.txt`) oder einen relativen Pfad (zum Beispiel, wenn `/home/joe` Ihr aktuelles Verzeichnis wäre, könnten Sie einfach auf die Datei als `myfile.txt` verweisen).

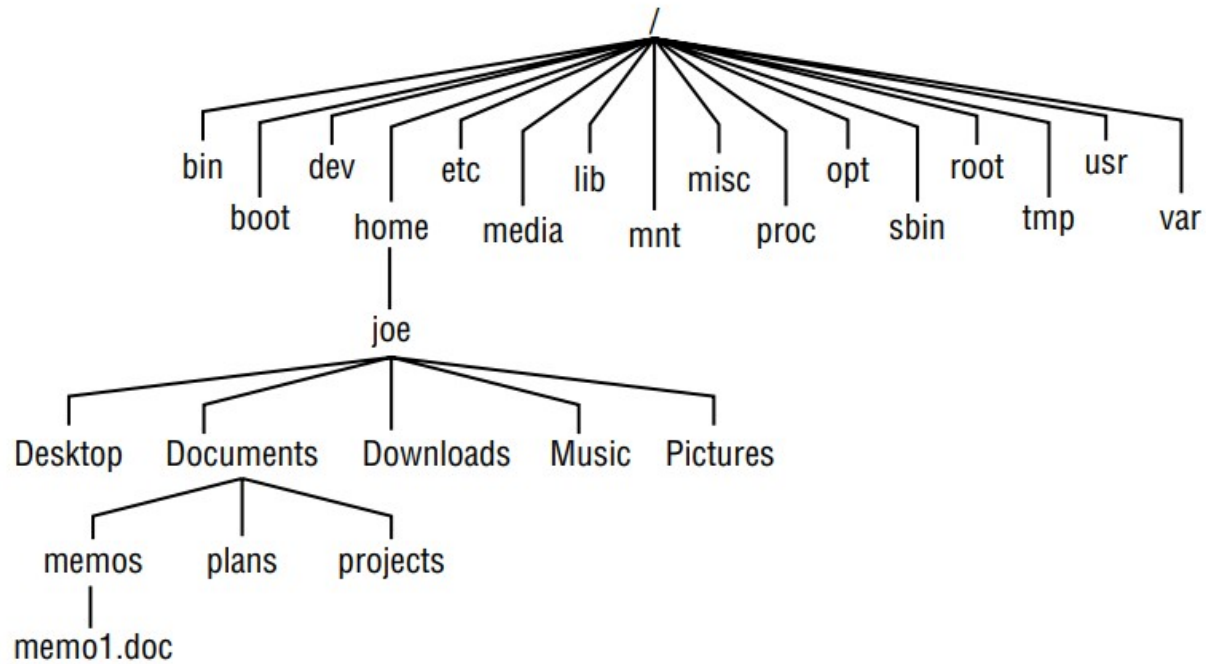
Wenn Sie die Dateien und Verzeichnisse in Linux abbilden würden, würde es wie ein umgekehrter Baum aussehen. Oben befindet sich das Stammverzeichnis (nicht zu verwechseln mit dem Root-Benutzer), das durch einen einzelnen Schrägstrich (`/`) dargestellt wird. Darunter befindet sich eine Reihe von häufigen Verzeichnissen im Linux-System, wie z.B. `bin`, `dev`, `home`, `lib` und `tmp`, um nur einige zu nennen. Jedes dieser Verzeichnisse sowie Verzeichnisse, die dem Stammverzeichnis hinzugefügt werden, können Unterverzeichnisse enthalten.

Abbildung 1 veranschaulicht, wie das Linux-Dateisystem als Hierarchie organisiert ist. Um zu zeigen, wie Verzeichnisse verbunden sind, zeigt die Abbildung ein `/home`-Verzeichnis, das ein Unterverzeichnis für den Benutzer `joe` enthält. Innerhalb des `joe`-Verzeichnisses befinden sich `Desktop`, `Dokumente` und andere Unterverzeichnisse. Um auf eine Datei namens `memo1.doc` im Verzeichnis `memos` zu verweisen, können Sie den vollständigen Pfad von `/home/joe/Documents/memos/memo1.doc` eingeben. Wenn Ihr aktuelles Verzeichnis `/home/joe/` ist, verweisen Sie auf die Datei als `Documents/memos/memo1.doc`.

---

<sup>1</sup> Linux® Bible, Tenth Edition. Christopher Negus. © 2020 John Wiley & Sons, Inc.

Das Linux-Dateisystem ist als Hierarchie von Verzeichnissen organisiert.



Einige dieser Linux-Verzeichnisse könnten für Sie interessant sein:

|                     |  |
|---------------------|--|
| <code>/bin</code>   | Enthält gängige Linux-Benutzerbefehle wie <code>ls</code> , <code>sort</code> , <code>date</code> und <code>chmod</code> .   |
| <code>/boot</code>  | Enthält den startfähigen Linux-Kernel, das initiale RAM-Disk und die Konfigurationsdateien des Bootloaders (GRUB).   |
| <code>/dev</code>   | Enthält Dateien, die Zugriffspunkte für Geräte auf Ihrem System darstellen. Dazu gehören Terminalgeräte ( <code>tty*</code> ), Festplatten ( <code>hd*</code> oder <code>sd*</code> ), RAM ( <code>ram*</code> ) und CD-ROM ( <code>cd*</code> ). Benutzer können direkt über diese Gerätedateien auf diese Geräte zugreifen. Anwendungen verbergen jedoch oft die tatsächlichen Gerätenamen vor den Endbenutzern. |
| <code>/etc</code>   | Enthält administrative Konfigurationsdateien. Die meisten dieser Dateien sind einfache Textdateien, die, sofern der Benutzer die entsprechende Berechtigung hat, mit einem beliebigen Texteditor bearbeitet werden können.   |
| <code>/home</code>  | Enthält Verzeichnisse, die jedem regulären Benutzer mit einem Anmeldekonto zugewiesen sind. (Der Root-Benutzer bildet eine Ausnahme und verwendet <code>/root</code> als sein oder ihr Home-Verzeichnis.)  |
| <code>/media</code> | Bietet einen standardisierten Ort zum automatischen Einhängen von Geräten (insbesondere Wechselmedien). Wenn das Medium einen Volumennamen hat, wird dieser Name typischerweise als Einhängpunkt verwendet. Zum Beispiel würde ein USB-  |

|                |   |
|----------------|---|
|                | Laufwerk mit dem Volumennamen myusb auf /media/myusb eingehängt.  |
| /lib           | Enthält gemeinsam genutzte Bibliotheken, die von Anwendungen in /bin und /sbin benötigt werden, um das System zu booten.  |
| /mnt<br>/media | Ein gemeinsamer Einhängpunkt für viele Geräte, bevor er vom Standardverzeichnis ersetzt wurde. Einige startfähige Linux-Systeme verwenden dieses Verzeichnis immer noch, um Festplattenpartitionen und entfernte Dateisysteme einzuhängen. Viele Leute verwenden dieses Verzeichnis immer noch, um temporär lokale oder entfernte Dateisysteme einzuhängen, die nicht dauerhaft eingehängt sind.  |
| /misc          | Ein Verzeichnis, das manchmal zum automatischen Einhängen von Dateisystemen auf Anfrage verwendet wird.   |
| /opt           | Verzeichnisstruktur, die zur Speicherung von Add-On-Anwendungssoftware zur Verfügung steht.   |
| /proc          | Enthält Informationen über Systemressourcen.  |
| /root          | Stellt das Home-Verzeichnis des Root-Benutzers dar. Das Home-Verzeichnis für root befindet sich aus Sicherheitsgründen nicht unter /home.   |
| /sbin          | Enthält administrative Befehle und Daemon-Prozesse.   |
| /sys           | Enthält Parameter für Dinge wie die Anpassung der Blockspeicherung und die Verwaltung von cgroups.  |
| /tmp           | Enthält temporäre Dateien, die von Anwendungen verwendet werden.  |
| /usr           | Enthält Benutzerdokumentationen, Spiele, grafische Dateien (X11), Bibliotheken (lib) und eine Vielzahl anderer Befehle und Dateien, die während des Bootvorgangs nicht benötigt werden. Das /usr-Verzeichnis ist für Dateien gedacht, die sich nach der Installation nicht ändern (theoretisch könnte /usr als schreibgeschützt eingehängt werden).   |
| /var           | Enthält Verzeichnisse mit Daten, die von verschiedenen Anwendungen verwendet werden. Insbesondere hier platzieren Sie Dateien, die Sie als FTP-Server (/var/ftp) oder Webserver (/var/www) freigeben. Es enthält auch alle Systemprotokolldateien (/var/log) und Spool-Dateien in /var/spool (wie Mail, Cups und News). Das /var-Verzeichnis enthält Verzeichnisse und Dateien, die sich häufig ändern sollen. Auf Servercomputern ist es üblich, das /var-Verzeichnis als separates Dateisystem zu erstellen, das mit einem Dateisystemtyp leicht erweitert werden kann. |

Die Dateisysteme in den DOS- oder Microsoft Windows-Betriebssystemen unterscheiden sich von der Dateistruktur von Linux, wie im Seitenelement "Linux-Dateisysteme versus auf Windows basierende Dateisysteme" erklärt wird.

## Linux-Dateisysteme versus auf Windows basierende Dateisysteme

Obwohl sich die Linux-Dateisysteme in vielerlei Hinsicht ähneln, gibt es einige markante Unterschiede im Vergleich zu den Dateisystemen, die in den MS-DOS- und Windows-Betriebssystemen verwendet werden. Hier sind einige dieser Unterschiede:

- In den Dateisystemen von MS-DOS und Windows repräsentieren Laufwerksbuchstaben verschiedene Speichergeräte. In Linux sind alle Speichergeräte mit der Dateisystemhierarchie verbunden. So ist für den Benutzer nicht sichtbar, dass sich beispielsweise /usr möglicherweise

auf einer separaten Festplatte befindet oder dass /mnt/remote1 ein Dateisystem von einem anderen Computer ist.

- Schrägstriche werden in Linux verwendet, um Verzeichnisnamen zu trennen, anstelle von umgekehrten Schrägstrichen wie in Windows. Daher ist C:\home\joe in einem Microsoft-System in einem Linux-System /home/joe.

- Dateinamen haben in DOS fast immer Suffixe (wie .txt für Textdateien oder .docx für Textverarbeitungsdateien). Obwohl Sie diese Konvention manchmal in Linux verwenden können, haben dreistellige Suffixe in Linux keine vorgeschriebene Bedeutung. Sie können jedoch nützlich sein, um den Dateityp zu identifizieren. Viele Linux-Anwendungen und Desktop-Umgebungen verwenden Dateisuffixe, um den Inhalt einer Datei zu bestimmen. In Linux bedeuten jedoch DOS-Befehlserweiterungen wie .com, .exe und .bat nicht zwangsläufig eine Ausführbarkeit. (Berechtigungsflags machen Linux-Dateien ausführbar.)

- Jede Datei und jedes Verzeichnis in einem Linux-System hat Berechtigungen und Eigentum zugeordnet. Die Sicherheit variiert bei Microsoft-Systemen. Da DOS und Microsoft Windows als Einzelbenutzer-Systeme begannen, war das Dateieigentum nicht in diese Systeme eingebaut, als sie entworfen wurden. In späteren Versionen wurden Funktionen wie Datei- und Ordnerattribute hinzugefügt, um dieses Problem anzugehen.

## Die Verwendung grundlegender Dateisystembefehle

Ich möchte Ihnen einige einfache Befehle vorstellen, um sich im Dateisystem zurechtzufinden. Wenn Sie mitmachen möchten, melden Sie sich an und öffnen Sie eine Shell. Wenn Sie sich bei einem Linux-System anmelden und eine Shell öffnen, werden Sie in Ihr Home-Verzeichnis versetzt. Als Linux-Benutzer werden die meisten Dateien, die Sie speichern und mit denen Sie arbeiten, wahrscheinlich in diesem Verzeichnis oder in von Ihnen erstellten Unterverzeichnissen gespeichert. Tabelle 1 zeigt Befehle zum Erstellen und Verwenden von Dateien und Verzeichnissen.

| Befehl | Beschreibung   |
|--------|--|
| cd     | Wechseln des aktuellen Verzeichnisses                    |
| touch  | Erstellen einer leeren Datei                             |
| mkdir  | Erstellen eines neuen Verzeichnisses                     |
| rm     | Löschen von Dateien oder Verzeichnissen                  |
| cp     | Kopieren von Dateien und Verzeichnissen                  |
| mv     | Verschieben oder Umbenennen von Dateien                  |
| ls     | Auflisten des Inhalts eines Verzeichnisses               |
| cat    | Anzeigen des Inhalts einer Datei                         |
| nano   | Öffnen eines Texteditors zur Bearbeitung                 |
| chmod  | Ändern der Berechtigungen von Dateien und Verzeichnissen |

TABELLE 1 Befehle zum Erstellen und Verwenden von Dateien

Bitte beachten Sie, dass einige dieser Befehle Administratorrechte erfordern können, abhängig von den Dateien und Verzeichnissen, auf die sie angewendet werden.

Einer der grundlegendsten Befehle, den Sie von der Shell aus verwenden, ist cd. Der cd-Befehl kann ohne Optionen verwendet werden (um Sie in Ihr Home-Verzeichnis zu bringen) oder mit vollständigen oder relativen Pfaden. Betrachten Sie die folgenden Befehle:

```
$ cd /usr/share/
$ pwd
/usr/share
$ cd doc
$ pwd
/usr/share/doc
$ cd
$ pwd
/home/chris
```

Die Option /usr/share stellt den absoluten Pfad zu einem Verzeichnis im System dar. Da er mit einem Schrägstrich (/) beginnt, teilt dieser Pfad der Shell mit, dass sie am Anfang des Dateisystems beginnen soll und Sie zum Verzeichnis share führen soll, das im Verzeichnis usr existiert. Die doc-Option des cd-

Befehls sucht nach einem Verzeichnis namens doc, das relativ zum aktuellen Verzeichnis ist. Mit diesem Befehl wurde also /usr/share/doc zu Ihrem aktuellen Verzeichnis.

Danach kehren Sie durch Eingabe von cd alleine in Ihr Home-Verzeichnis zurück. Wenn Sie sich jemals fragen, wo Sie sich im Dateisystem befinden, kann Ihnen der Befehl pwd helfen. Hier sind noch einige andere interessante Optionen des cd-Befehls:

```
$ cd ~
$ pwd
/home/chris
$ cd ~/Musik
$ pwd
/home/chris/Musik
$ cd ../../../../usr
$ pwd
/usr
```

Die Tilde (~) repräsentiert Ihr Home-Verzeichnis. Daher führt cd ~ Sie dorthin. Sie können die Tilde auch verwenden, um Verzeichnisse relativ zu Ihrem Home-Verzeichnis anzugeben, wie zum Beispiel /home/chris/Musik mit ~/Musik. Wenn Sie einen Namen als Option eingeben, gelangen Sie in ein Verzeichnis unterhalb des aktuellen Verzeichnisses, aber Sie können zwei Punkte (..) verwenden, um in ein Verzeichnis über dem aktuellen Verzeichnis zu gelangen. Das gezeigte Beispiel führt Sie drei Verzeichnisebenen nach oben (zu /) und dann in das /usr-Verzeichnis.

Die folgenden Schritte führen Sie durch den Prozess des Erstellens von Verzeichnissen in Ihrem Home-Verzeichnis und des Navigierens zwischen Ihren Verzeichnissen, mit einer Erwähnung der Einstellung angemessener Dateiberechtigungen:

1. Gehen Sie in Ihr Home-Verzeichnis. Geben Sie dazu einfach cd in eine Shell ein und drücken Sie die Eingabetaste. (Für andere Möglichkeiten, auf Ihr Home-Verzeichnis zu verweisen, siehe das Seitenelement "Verzeichnisse identifizieren".)
2. Um sicherzustellen, dass Sie sich in Ihrem Home-Verzeichnis befinden, geben Sie pwd ein. Wenn ich das mache, erhalte ich folgende Antwort (Ihre wird Ihr Home-Verzeichnis widerspiegeln):

```
$ pwd
/home/joe
```

3. Erstellen Sie ein neues Verzeichnis namens test in Ihrem Home-Verzeichnis, wie folgt:

```
$ mkdir test
```

4. Überprüfen Sie die Berechtigungen des Verzeichnisses:

```
$ ls -ld test
drwxr-xr-x 2 joe sales 1024 Jan 24 12:17 test
```

Diese Auflistung zeigt, dass test ein Verzeichnis (d) ist. Das d wird von den Berechtigungen (rwxr-xr-x) gefolgt, die später im Abschnitt "Verständnis der Dateiberechtigungen und Eigentumsverhältnisse" erklärt werden. Der Rest der Informationen gibt den Besitzer (joe), die Gruppe (sales) und das Datum an, an dem die Dateien im Verzeichnis zuletzt geändert wurden (24. Januar um 12:17 Uhr).

## Hinweis

Wenn Sie einen neuen Benutzer in Fedora und Red Hat Enterprise Linux hinzufügen, wird der Benutzer standardmäßig einer Gruppe mit demselben Namen zugewiesen. In dem vorangegangenen Text würde beispielsweise der Benutzer joe der Gruppe joe zugewiesen. Dieser Ansatz zur Zuweisung von Gruppen wird als Schema der benutzerprivaten Gruppe bezeichnet.

Geben Sie jetzt Folgendes ein:

```
$ chmod 700 test
```

Dieser Schritt ändert die Berechtigungen des Verzeichnisses, um Ihnen vollständigen Zugriff zu geben und allen anderen keinen Zugriff zu gewähren. (Die neuen Berechtigungen sollten rwx----- lauten.)

5. Machen Sie das Verzeichnis test zu Ihrem aktuellen Verzeichnis wie folgt:

```
$ cd test
$ pwd
/home/joe/test
```

Wenn Sie mitgegangen sind, ist zu diesem Zeitpunkt ein Unterverzeichnis Ihres Home-Verzeichnisses namens test Ihr aktuelles Arbeitsverzeichnis. Sie können Dateien und Verzeichnisse im Testverzeichnis erstellen sowie die Beschreibungen im Rest dieses Kapitels verwenden.

## Verwendung von Metazeichen und Operatoren

Ob Sie Dateien in Ihrem Linux-System auflisten, verschieben, kopieren, entfernen oder anderweitig damit arbeiten, bestimmte spezielle Zeichen, die als Metazeichen und Operatoren bezeichnet werden, helfen Ihnen, effizienter mit Dateien zu arbeiten. Metazeichen können Ihnen helfen, eine oder mehrere

Dateien abzugleichen, ohne jeden Dateinamen vollständig eingeben zu müssen. Operatoren ermöglichen es Ihnen, Informationen von einem Befehl oder einer Datei zu einem anderen Befehl oder einer anderen Datei zu leiten.

## Verwendung von Dateiabgleichmetazeichen

Um Ihnen einige Tastenanschläge zu sparen und es Ihnen zu ermöglichen, leicht auf eine Gruppe von Dateien zu verweisen, lässt Sie die Bash-Shell Metazeichen verwenden. Jedes Mal, wenn Sie auf eine Datei oder ein Verzeichnis verweisen müssen, z. B. um sie aufzulisten, zu öffnen oder zu entfernen, können Sie Metazeichen verwenden, um die gewünschten Dateien abzugleichen. Hier sind einige nützliche Metazeichen zum Abgleichen von Dateinamen:

- \* Passt zu einer beliebigen Anzahl von Zeichen.
- ? Passt zu einem beliebigen einzelnen Zeichen.
- [...] Passt zu einem der Zeichen zwischen den Klammern, was einen durch Bindestriche getrennten Bereich von Buchstaben oder Zahlen enthalten kann.

Probieren Sie einige dieser Dateiabgleichmetazeichen aus, indem Sie zunächst in ein leeres Verzeichnis gehen (wie das Testverzeichnis, das im vorherigen Abschnitt beschrieben wurde), und einige leere Dateien erstellen:

```
$ touch apple banana grape grapefruit watermelon
```

Der Befehl touch erstellt leere Dateien. Die folgenden Befehle zeigen Ihnen, wie Sie Shell-Metazeichen mit dem Befehl ls verwenden, um Dateinamen abzugleichen. Versuchen Sie die folgenden Befehle, um festzustellen, ob Sie die gleichen Antworten erhalten:

```
$ ls a*
apple
$ ls g*
grape grapefruit
$ ls g*t
grapefruit
$ ls *e*
apple grape grapefruit watermelon
$ ls *n*
banana watermelon
```

Im ersten Beispiel werden alle Dateien abgeglichen, die mit a beginnen (apple). Im nächsten Beispiel werden alle Dateien abgeglichen, die mit g beginnen (grape, grapefruit). Als nächstes werden Dateien abgeglichen, die mit g beginnen und mit t enden (grapefruit). Als nächstes werden alle Dateien abgeglichen, die ein e im Namen enthalten (apple, grape, grapefruit, watermelon). Schließlich werden alle Dateien abgeglichen, die ein n enthalten (banane, watermelon).



Hier sind ein paar Beispiele für Musterabgleiche mit dem Fragezeichen (?):

```
$ ls ???e  
apple grape  
$ ls g???e*  
grape grapefruit
```

Das erste Beispiel passt zu allen fünf Zeichen langen Dateien, die mit e enden (apple, grape). Das zweite passt zu allen Dateien, die mit g beginnen und e als fünftes Zeichen haben (grape, grapefruit).

Die folgenden Beispiele verwenden geschweifte Klammern für den Musterabgleich:

```
$ ls [abw]*  
apple banana watermelon  
$ ls [agw]*[ne]  
apple grape watermelon
```

Im ersten Beispiel werden alle Dateien abgeglichen, die mit a, b oder w beginnen. Im zweiten werden alle Dateien abgeglichen, die mit a, g oder w beginnen und entweder mit n oder e enden. Sie können auch Bereiche innerhalb von Klammern angeben. Zum Beispiel:

```
$ ls [a-g]*  
apple banana grape grapefruit
```

Hier werden Dateinamen abgeglichen, die mit einem Buchstaben von a bis g beginnen.

## Verwendung von Datei-Umleitungs-Metazeichen

Befehle empfangen Daten von der Standardeingabe und senden sie an die Standardausgabe. Mit Pipes (wie zuvor beschrieben) können Sie die Standardausgabe von einem Befehl an die Standardeingabe eines anderen lenken. Mit Dateien können Sie die Zeichen kleiner als (<) und größer als (>) verwenden, um Daten zu und von Dateien zu lenken. Hier sind die Datei-Umleitungszeichen:

- <     Leitet den Inhalt einer Datei an den Befehl weiter. In den meisten Fällen ist dies die Standardaktion, die vom Befehl erwartet wird, und die Verwendung des Zeichens ist optional; less bigfile zu verwenden ist dasselbe wie less < bigfile.
- >     Leitet die Standardausgabe eines Befehls an eine Datei weiter. Wenn die Datei existiert, wird der Inhalt dieser Datei überschrieben.

- 2>     Leitet die Standardfehlerausgabe (Fehlermeldungen) in die Datei.
- &>     Leitet sowohl die Standardausgabe als auch die Standardfehlerausgabe in die Datei.
- >>     Leitet die Ausgabe eines Befehls in eine Datei und fügt die Ausgabe am Ende der vorhandenen Datei hinzu.

Im Folgenden sind einige Beispiele für Befehlszeilen, bei denen Informationen zu und von Dateien geleitet werden:

```
$ mail root < ~/.bashrc
$ man chmod | col -b > /tmp/chmod
$ echo "Ich habe das Projekt am $(date) abgeschlossen" >> ~/projekte
```

Im ersten Beispiel wird der Inhalt der .bashrc-Datei im Home-Verzeichnis in einer E-Mail-Nachricht an den Benutzer root des Computers gesendet. Die zweite Befehlszeile formatiert die chmod-Man-Seite (unter Verwendung des man-Befehls), entfernt zusätzliche Leerzeichen (col -b) und leitet die Ausgabe an die Datei /tmp/chmod (wobei die vorherige /tmp/chmod-Datei überschrieben wird, falls sie vorhanden ist). Der letzte Befehl führt dazu, dass der folgende Text zur Projektdatei des Benutzers hinzugefügt wird:

"Ich habe das Projekt am Sa 15 Jun 13:46:49 EDT 2019 abgeschlossen"

Eine weitere Art der Umleitung, die als Hier-Text bezeichnet wird (auch Here-Dokument genannt), ermöglicht es Ihnen, Text einzugeben, der als Standardeingabe für einen Befehl verwendet werden kann. Hier-Dokumente erfordern das Eingeben von zwei kleiner-als-Zeichen (<<) nach einem Befehl, gefolgt von einem Wort. Alle Eingaben nach diesem Wort werden als Benutzereingabe betrachtet, bis das Wort in einer eigenen Zeile wiederholt wird. Hier ist ein Beispiel:

```
$ mail root cnegus rjones bdecker << thetext
> Ich möchte allen mitteilen, dass es um 10 Uhr
> ein Treffen im Konferenzraum B geben wird. Alle sollten teilnehmen.
>
> -- James
> thetext
$
```

In diesem Beispiel wird eine E-Mail-Nachricht an die Benutzernamen root, cnegus, rjones und bdecker gesendet. Der zwischen <<thetext und thetext eingegebene Text wird zum Inhalt der Nachricht.

Eine häufige Verwendung von Hier-Text besteht darin, ihn mit einem Texteditor zu verwenden, um eine Datei zu erstellen oder zu erweitern, während ein Skript ausgeführt wird:

```
/bin/ed /etc/resolv.conf <<resendit
a
nameserver 100.100.100.100
.
w
```

```
q  
resendit
```

Mit diesen Zeilen, die einem Skript hinzugefügt werden, das vom Root-Benutzer ausgeführt wird, fügt der ed-Texteditor die IP-Adresse eines DNS-Servers zur Datei /etc/resolv.conf hinzu.

## Verwenden von geschweiften Klammern ({} )

Durch Verwendung geschweifter Klammern ({} ) können Sie eine Reihe von Zeichen über Dateinamen, Verzeichnisnamen oder andere Argumente erweitern, an die Sie Befehle geben. Wenn Sie beispielsweise eine Reihe von Dateien wie memo1 bis memo5 erstellen möchten, können Sie dies wie folgt tun:

```
$ touch memo{1,2,3,4,5}  
$ ls  
memo1 memo2 memo3 memo4 memo5
```

Die erweiterten Elemente müssen keine Zahlen oder sogar einzelne Ziffern sein. Sie könnten beispielsweise Nummern- oder Zeichensätze verwenden. Sie können auch beliebige Zeichenfolgen von Zeichen verwenden, solange Sie sie mit Kommas trennen. Hier sind einige Beispiele:

```
$ touch {John,Bill,Sally}-{Breakfast,Lunch,Dinner}  
$ ls  
Bill-Breakfast Bill-Lunch John-Dinner Sally-Breakfast Sally-Lunch  
Bill-Dinner John-Breakfast John-Lunch Sally-Dinner  
$ rm -f {John,Bill,Sally}-{Breakfast,Lunch,Dinner}  
$ touch {a..f}{1..5}  
$ ls  
a1 a3 a5 b2 b4 c1 c3 c5 d2 d4 e1 e3 e5 f2 f4  
a2 a4 b1 b3 b5 c2 c4 d1 d3 d5 e2 e4 f1 f3 f5
```

Im ersten Beispiel bedeutet die Verwendung von zwei Sätzen geschweifter Klammern, dass John, Bill und Sally jeweils Dateinamen mit Frühstück, Mittagessen und Abendessen haben. Wenn ich einen Fehler gemacht hätte, könnte ich den Befehl leicht zurückrufen und touch in rm -f ändern, um alle Dateien zu löschen. Im nächsten Beispiel geben die beiden Punkte zwischen den Buchstaben a und f sowie den Zahlen 1 und 5 die zu verwendenden Bereiche an. Beachten Sie die Dateien, die aus diesen wenigen Zeichen erstellt wurden.

## Auflisten von Dateien und Verzeichnissen

Das ls-Kommando ist das am häufigsten verwendete Kommando zum Auflisten von Informationen über Dateien und Verzeichnisse. Viele Optionen, die mit dem ls-Kommando verfügbar sind, ermöglichen es Ihnen, verschiedene Datei- und Verzeichnissätze zu sammeln sowie verschiedene Arten von Informationen über sie anzuzeigen.

Standardmäßig zeigt der ls-Befehl, wenn Sie ihn eingeben, alle nicht versteckten Dateien und Verzeichnisse im aktuellen Verzeichnis an. Wenn Sie jedoch ls eingeben, weisen viele Linux-Systeme (einschließlich Fedora und RHEL) dem Alias ls Optionen hinzu. Um zu sehen, ob ls als Alias zugewiesen ist, geben Sie Folgendes ein:

```
$ alias ls
alias ls='ls -color=auto'
```

Die Option --color=auto führt dazu, dass verschiedene Arten von Dateien und Verzeichnissen in verschiedenen Farben angezeigt werden. Kehren Sie also zum zuvor im Kapitel erstellten \$HOME/test-Verzeichnis zurück, fügen Sie ein paar verschiedene Dateitypen hinzu, und sehen Sie dann mit dem ls-Befehl nach, wie sie aussehen.

```
$ cd $HOME/test
$ touch scriptx.sh apple
$ chmod 755 scriptx.sh
$ mkdir Stuff
$ ln -s apple pointer_to_apple
$ ls
apple pointer_to_apple scriptx.sh Stuff
```

Obwohl Sie es im obigen Codebeispiel nicht sehen können, wird das Verzeichnis "Stuff" blau angezeigt, "pointer\_to\_apple" (ein symbolischer Link) erscheint als Aqua, und "scriptx.sh" (eine ausführbare Datei) erscheint in Grün. Alle anderen regulären Dateien werden in Schwarz angezeigt. Wenn Sie "ls -l" eingeben, um eine lange Liste dieser Dateien anzuzeigen, können diese verschiedenen Dateitypen noch deutlicher werden:

```
$ ls -l
total 4
-rw-rw-r--. 1 joe joe 0 Dec 18 13:38 apple
lrwxrwxrwx. 1 joe joe 5 Dec 18 13:46 pointer_to_apple -> apple
-rwxr-xr-x. 1 joe joe 0 Dec 18 13:37 scriptx.sh
drwxrwxr-x. 2 joe joe 4096 Dec 18 13:38 Stuff
```

Beachten Sie beim Betrachten der langen Liste, dass das erste Zeichen jeder Zeile den Dateityp anzeigt. Ein Bindestrich (-) zeigt eine reguläre Datei an, d zeigt ein Verzeichnis an, und l (kleines L) zeigt einen symbolischen Link an. Eine ausführbare Datei (eine Skript- oder Binärdatei, die als Befehl ausgeführt wird) hat die Ausführungsbits eingeschaltet (x). Mehr zu Ausführungsbits finden Sie im bevorstehenden Abschnitt "Verständnis von Dateiberechtigungen und Eigentumsverhältnissen".

Als Nächstes sollten Sie sich mit dem Inhalt Ihres Home-Verzeichnisses vertraut machen. Verwenden Sie die Optionen -l und -a für ls.

```
$ ls -la /home/joe
total 158
drwxrwxrwx  2  joe  sales 4096  May 12 13:55 .
drwxr-xr-x  3  root  root  4096  May 10 01:49 ..
-rw-----  1  joe  sales 2204  May 18 21:30 .bash_history
```

```

-rw-r--r-- 1 joe sales 24 May 10 01:50 .bash_logout
-rw-r--r-- 1 joe sales 230 May 10 01:50 .bash_profile
-rw-r--r-- 1 joe sales 124 May 10 01:50 .bashrc
drw-r--r-- 1 joe sales 4096 May 10 01:50 .kde
-rw-rw-r-- 1 joe sales 149872 May 11 22:49 letter
^         ^         ^         ^         ^         ^
col 1      col 2 col 3 col 4 col 5 col 6          col 7

```

Die Anzeige einer langen Liste (-l-Option) des Inhalts Ihres Home-Verzeichnisses zeigt Ihnen mehr über Dateigrößen und Verzeichnisse. Die Zeile "total" zeigt die insgesamt verwendete Festplattenkapazität der Dateien in der Liste an (158 Kilobytes in diesem Beispiel). Durch Hinzufügen der Option "alle Dateien" (-a) werden Dateien angezeigt, die mit einem Punkt (.) beginnen. Verzeichnisse wie das aktuelle Verzeichnis (.) und das übergeordnete Verzeichnis (..) - das Verzeichnis über dem aktuellen Verzeichnis - werden als Verzeichnisse durch den Buchstaben d am Anfang jedes Eintrags gekennzeichnet. Jedes Verzeichnis beginnt mit einem d und jede Datei beginnt mit einem Bindestrich (-).

Die Datei- und Verzeichnisnamen werden in Spalte 7 angezeigt. In diesem Beispiel steht ein Punkt (.) für /home/joe und zwei Punkte (..) repräsentieren /home - das übergeordnete Verzeichnis von /joe. Die meisten Dateien in diesem Beispiel sind Punkt (.)-Dateien, die GUI-Eigenschaften (Verzeichnis .kde) oder Shell-Eigenschaften (.bash-Dateien) speichern. Die einzige nicht-Punkt-Datei in dieser Liste ist die namens "letter".

Spalte 3 zeigt den Besitzer des Verzeichnisses oder der Datei an. Das Verzeichnis /home gehört root und alles andere gehört dem Benutzer joe, der zur Gruppe sales gehört (Gruppen sind in Spalte 4 aufgelistet).

Neben dem d oder -, enthält Spalte 1 auf jeder Zeile die Berechtigungen für diese Datei oder dieses Verzeichnis. Weitere Informationen in der Auflistung sind die Anzahl der Hardlinks zum Element (Spalte 2), die Größe jeder Datei in Bytes (Spalte 5) und das Datum und die Uhrzeit, zu denen jede Datei zuletzt geändert wurde (Spalte 6).

Hier sind einige weitere Fakten über Datei- und Verzeichnisauflistungen:

- Die Anzahl der Zeichen, die für ein Verzeichnis angezeigt werden (4096 Bytes in diesen Beispielen), spiegelt die Größe der Datei wider, die Informationen über das Verzeichnis enthält. Obwohl diese Zahl für ein Verzeichnis, das viele Dateien enthält, über 4096 Bytes wachsen kann, spiegelt diese Zahl nicht die Größe der in diesem Verzeichnis enthaltenen Dateien wider.
- Das Format der Zeit- und Datumsspalte kann variieren. Anstatt "May 12" anzuzeigen, könnte das Datum je nach Distribution und Spracheinstellung (LANG-Variablen) als "2019-05-12" angezeigt werden.
- Gelegentlich kann anstelle des Ausführungsbits (x) auf einer ausführbaren Datei ein s an dieser Stelle stehen. Mit einem s, das innerhalb der Besitzer- (-rwsr-xr-x) oder Gruppen- (-rwxr-sr-x) Berechtigungen erscheint, oder beides (-rwsr-sr-x), kann die Anwendung von jedem Benutzer ausgeführt werden, aber die Eigentümerschaft des laufenden Prozesses wird dem Benutzer/der Gruppe der Anwendung zugewiesen, anstatt dem Benutzer, der den Befehl startet. Dies wird als

ein Set-UID- oder Set-GID-Programm bezeichnet. Zum Beispiel hat der Befehl "mount" Berechtigungen als -rwsr-xr-x. Dadurch kann jeder Benutzer "mount" ausführen, um eingehängte Dateisysteme aufzulisten (obwohl Sie immer noch root sein müssen, um "mount" in den meisten Fällen von der Befehlszeile aus tatsächlich einzuhängen).

- Wenn ein t am Ende eines Verzeichnisses erscheint, zeigt dies an, dass das Sticky-Bit für dieses Verzeichnis gesetzt ist (zum Beispiel drwxrwxr-t). Durch Setzen des Sticky-Bits in einem Verzeichnis kann der Besitzer des Verzeichnisses anderen Benutzern und Gruppen das Hinzufügen von Dateien zum Verzeichnis ermöglichen, aber verhindern, dass Benutzer die Dateien anderer in diesem Verzeichnis löschen. Mit einem gesetzten GID für ein Verzeichnis werden alle in diesem Verzeichnis erstellten Dateien derselben Gruppe wie die des Verzeichnisses zugewiesen. (Wenn Sie ein großes S oder T anstelle der Ausführungsbits auf einem Verzeichnis sehen, bedeutet dies, dass die Berechtigung für das Set-GID- oder das Sticky-Bit gesetzt wurde, aber aus irgendeinem Grund das Ausführungsbit nicht ebenfalls aktiviert war.)
- Wenn Sie ein Pluszeichen am Ende der Berechtigungsbits sehen (zum Beispiel -rw-rwr--+), bedeutet dies, dass erweiterte Attribute (+), wie Zugriffskontrolllisten (ACLs), auf der Datei gesetzt sind. Ein Punkt am Ende (.) zeigt an, dass SELinux auf die Datei angewendet wird.

## Identifizierung von Verzeichnissen

Wenn Sie Ihr Home-Verzeichnis in einer Befehlszeile der Shell identifizieren müssen, können Sie die folgenden Optionen verwenden:

**\$HOME** Diese Umgebungsvariable speichert den Namen Ihres Home-Verzeichnisses.  
**~** Die Tilde (~) repräsentiert Ihr Home-Verzeichnis auf der Befehlszeile.

Sie können die Tilde auch verwenden, um das Home-Verzeichnis einer anderen Person zu identifizieren. Zum Beispiel würde ~joe auf das Home-Verzeichnis von Joe (wahrscheinlich /home/joe) erweitert werden. Wenn ich also in das Verzeichnis /home/joe/test gehen möchte, könnte ich cd ~joe/test eingeben, um dorthin zu gelangen.

Weitere spezielle Möglichkeiten, Verzeichnisse in der Shell zu identifizieren, sind:

**.** Ein einzelner Punkt (.) bezieht sich auf das aktuelle Verzeichnis.  
**..** Zwei Punkte (..) beziehen sich auf ein Verzeichnis direkt über dem aktuellen Verzeichnis.  
**\$PWD** Diese Umgebungsvariable bezieht sich auf das aktuelle Arbeitsverzeichnis.  
**\$OLDPWD** Diese Umgebungsvariable bezieht sich auf das vorherige Arbeitsverzeichnis, bevor Sie zum aktuellen gewechselt sind. (Die Eingabe von cd - bringt Sie zurück zum Verzeichnis, das durch \$OLDPWD dargestellt wird.)

Wie ich bereits erwähnt habe, gibt es viele nützliche Optionen für den `ls`-Befehl. Kehren Sie zum `$HOME/test`-Verzeichnis zurück, in dem Sie gearbeitet haben. Hier sind einige Beispiele für `ls`-Optionen. Machen Sie sich keine Sorgen, wenn die Ausgabe nicht genau mit dem übereinstimmt, was sich zu diesem Zeitpunkt in Ihrem Verzeichnis befindet.

Jede Datei oder jedes Verzeichnis, das mit einem Punkt (.) beginnt, gilt als versteckt und wird standardmäßig nicht mit `ls` angezeigt. Diese Punkt-Dateien sind typischerweise Konfigurationsdateien oder Verzeichnisse, die sich in Ihrem Home-Verzeichnis befinden müssen, aber nicht in Ihrer täglichen Arbeit gesehen werden müssen. Mit `-a` können Sie diese Dateien sehen.

Die Option `-t` zeigt Dateien in der Reihenfolge an, in der sie zuletzt geändert wurden. Mit der Option `-F` wird ein Backslash (/) am Ende von Verzeichnisnamen angezeigt, ein Sternchen (\*) wird zu ausführbaren Dateien hinzugefügt, und ein @ wird neben symbolischen Links angezeigt.

Um versteckte und nicht versteckte Dateien anzuzeigen:

```
$ ls -a
. apple docs grapefruit pointer_to_apple .stuff watermelon
.. banana grape .hiddendir script.sh .tmpfile
```

Um alle Dateien nach dem zuletzt geänderten Zeitpunkt aufzulisten:

```
$ ls -at
.tmpfile .hiddendir .. docs watermelon banana script.sh
. .stuff pointer_to_apple grapefruit apple grape
```

Um Dateien aufzulisten und Dateityp-Indikatoren anzuhängen:

```
$ ls -F
apple banana docs/ grape grapefruit pointer_to_apple@ script.sh*
watermelon
```

Um zu vermeiden, dass bestimmte Dateien oder Verzeichnisse angezeigt werden, wenn Sie `ls` verwenden, verwenden Sie die Option `--hide=`.

In den nächsten Beispielen erscheint keine Datei, die mit `g` beginnt, in der Ausgabe. Durch Verwendung einer `-d`-Option für ein Verzeichnis werden Informationen über dieses Verzeichnis angezeigt, anstatt die Dateien und Verzeichnisse anzuzeigen, die das Verzeichnis enthält. Die `-R`-Option listet alle Dateien im aktuellen Verzeichnis sowie alle Dateien oder Verzeichnisse auf, die mit dem ursprünglichen Verzeichnis verbunden sind. Die `-S`-Option listet Dateien nach Größe auf.

Um alle Dateien auszuschließen, die mit dem Buchstaben `g` beginnen:

```
$ ls --hide=g*  
apple banana docs pointer_to_apple script.sh watermelon
```

Um Informationen über ein Verzeichnis anstelle der darin enthaltenen Dateien anzuzeigen:

```
$ ls -ld $HOME/test/  
drwxrwxr-x. 4 joe joe 4096 Dec 18 22:00 /home/joe/test/
```

Um mehrere Verzeichnisebenen zu erstellen (-p ist erforderlich):

```
$ mkdir -p $HOME/test/documents/memos/
```

Um alle Dateien und Verzeichnisse rekursiv vom aktuellen Verzeichnis aus aufzulisten:

```
$ ls -R  
...
```

Um Dateien nach Größe aufzulisten:

```
$ ls -S  
...
```

## Verständnis von Dateiberechtigungen und Eigentum

Nachdem Sie eine Weile mit Linux gearbeitet haben, werden Sie fast sicher eine Fehlermeldung "Permission denied" erhalten. Berechtigungen, die Dateien und Verzeichnissen in Linux zugeordnet sind, wurden entworfen, um zu verhindern, dass Benutzer auf private Dateien anderer Benutzer zugreifen, und um wichtige Systemdateien zu schützen. Die neun Bits, die jeder Datei für Berechtigungen zugewiesen sind, definieren den Zugriff, den Sie und andere auf Ihre Datei haben. Berechtigungsbits für eine reguläre Datei erscheinen als -rwxrwxrwx. Diese Bits werden verwendet, um festzulegen, wer die Datei lesen, schreiben oder ausführen kann.



## Hinweis

Für eine reguläre Datei erscheint ein Bindestrich vor dem neun-Bit-Berechtigungsindikator. Anstelle eines Bindestrichs können Sie möglicherweise ein d (für ein Verzeichnis), l (für eine symbolische Verknüpfung), b (für ein Blockgerät), c (für ein Zeichengerät), s (für eine Socket-Verbindung) oder p (für eine benannte Pipe) sehen.

Von den neun Berechtigungsbits gelten die ersten drei Bits für die Berechtigungen des Eigentümers, die nächsten drei gelten für die Gruppe, der die Datei zugewiesen ist, und die letzten drei gelten für alle anderen. Das r steht für Lesen, das w steht für Schreiben und das x steht für Ausführungsberechtigungen. Wenn anstelle des Buchstabens ein Bindestrich erscheint, bedeutet dies, dass die Berechtigung für das zugehörige Lesen, Schreiben oder Ausführen deaktiviert ist.

Da Dateien und Verzeichnisse unterschiedliche Elementtypen sind, bedeuten Lese-, Schreib- und Ausführungsberechtigungen für Dateien und Verzeichnisse unterschiedliche Dinge. Tabelle 2 erklärt, was Sie mit jedem von ihnen tun können.

TABLE 2 Lesen, Schreiben und Ausführungsrechte setzen

| Permission | File  | Directory   |
|------------|---|---|
| Read       | View what's in the file.                            | See what files and subdirectories it contains.  |
| Write      | Change the file's content, rename it, or delete it. | Add files or subdirectories to the directory. Remove files or directories from the directory.   |
| Execute    | Run the file as a program.                          | Change to the directory as the current directory, search through the directory, or execute a program from the directory. Access file metadata (file size, time stamps, and so on) of files in that directory. |

Wie bereits erwähnt, können Sie die Berechtigung für eine beliebige Datei oder ein beliebiges Verzeichnis anzeigen, indem Sie den Befehl `ls -ld` eingeben. Die benannte Datei oder das Verzeichnis erscheinen wie im folgenden Beispiel:

```
$ ls -ld ch3 test
-rw-rw-r-- 1 joe sales 4983 Jan 18 22:13 ch3
drwxr-xr-x 2 joe sales 1024 Jan 24 13:47 test
```

Die erste Zeile zeigt, dass die Datei `ch3` Lese- und Schreibberechtigungen für den Eigentümer und die Gruppe hat. Alle anderen Benutzer haben Leseberechtigung, was bedeutet, dass sie die Datei anzeigen, aber nicht deren Inhalt ändern oder entfernen können. Die zweite Zeile zeigt das Verzeichnis `test`

(angezeigt durch den Buchstaben d vor den Berechtigungsbits). Der Eigentümer hat Lese-, Schreib- und Ausführungsberechtigungen, während die Gruppe und andere Benutzer nur Lese- und Ausführungsberechtigungen haben. Als Ergebnis kann der Eigentümer Dateien in diesem Verzeichnis hinzufügen, ändern oder löschen, und alle anderen können nur den Inhalt lesen, in dieses Verzeichnis wechseln und den Inhalt des Verzeichnisses auflisten. (Wenn Sie die Option -d nicht für ls verwendet hätten, hätten Sie Dateien im Verzeichnis test aufgelistet, anstatt die Berechtigungen dieses Verzeichnisses anzuzeigen.)

## Ändern von Berechtigungen mit chmod (Zahlen)

Wenn Sie eine Datei besitzen, können Sie mit dem chmod-Befehl die Berechtigungen nach Belieben ändern. In einer Methode zur Durchführung dessen wird jeder Berechtigung (Lesen, Schreiben und Ausführen) eine Zahl zugeordnet - r=4, w=2 und x=1 - und Sie verwenden die Gesamtzahl jeder Gruppe, um die Berechtigungstabelle zu erstellen. Zum Beispiel, um die Berechtigungen für sich selbst als Besitzer vollständig zu öffnen, würden Sie die erste Zahl auf 7 setzen (4+2+1), und dann würden Sie der Gruppe und anderen nur Leseberechtigung geben, indem Sie sowohl die zweite als auch die dritte Zahl auf 4 setzen (4+0+0), so dass die Endzahl 744 ist. Jede Kombination von Berechtigungen kann von 0 (keine Berechtigung) bis 7 (volle Berechtigung) führen.

Hier sind einige Beispiele, wie die Berechtigungen für eine Datei (mit dem Namen "Datei") geändert werden können und welche Berechtigungen sich daraus ergeben würden:

Der folgende chmod-Befehl ergibt diese Berechtigung: rwxrwxrwx

```
# chmod 777 Datei
```

Der folgende chmod-Befehl ergibt diese Berechtigung: rwxr-xr-x

```
# chmod 755 Datei
```

Der folgende chmod-Befehl ergibt diese Berechtigung: rw-r--r--

```
# chmod 644 Datei
```

Der folgende chmod-Befehl ergibt diese Berechtigung: -----

```
# chmod 000 Datei
```

Der chmod-Befehl kann auch rekursiv verwendet werden. Angenommen, Sie möchten einem gesamten Verzeichnisbaum die Berechtigung 755 (rwxr-xr-x) geben, beginnend im Verzeichnis \$HOME/myapps. Sie könnten dazu die -R-Option verwenden, wie folgt:

```
$ chmod -R 755 $HOME/myapps
```

Alle Dateien und Verzeichnisse unterhalb des Verzeichnisses "myapps" in Ihrem Home-Verzeichnis erhalten die Berechtigungen 755. Da der numerische Ansatz zur Festlegung von Berechtigungen alle Berechtigungsbits gleichzeitig ändert, ist es üblicher, Buchstaben zu verwenden, um Berechtigungsbits rekursiv über eine große Anzahl von Dateien zu ändern.

## Ändern von Berechtigungen mit chmod (Buchstaben)

Sie können auch Dateiberechtigungen mit Pluszeichen (+) und Minuszeichen (-) umschalten, zusammen mit Buchstaben, um anzuzeigen, was sich ändert und für wen. Unter Verwendung von Buchstaben können Sie für jeden Benutzer (u), Gruppe (g), andere (o) und alle Benutzer (a) die Lese- (r), Schreib- (w) und Ausführungs- (x) -Bits ändern. Beginnen Sie zum Beispiel mit einer Datei, die alle Berechtigungen offen hat (rwxrwxrwx). Führen Sie die folgenden chmod-Befehle mit Minuszeichenoptionen aus. Die resultierenden Berechtigungen werden rechts von jedem Befehl angezeigt.

Der folgende chmod-Befehl ergibt diese Berechtigung: r-xr-xr-x

```
$ chmod a-w Datei
```

Der folgende chmod-Befehl ergibt diese Berechtigung: rwxrwxrw-

```
$ chmod o-x Datei
```

Der folgende chmod-Befehl ergibt diese Berechtigung: rwx-----

```
$ chmod go-rwx Datei
```

Ebenso starten die folgenden Beispiele mit allen Berechtigungen geschlossen (-----). Das Pluszeichen wird mit chmod verwendet, um Berechtigungen einzuschalten.

Der folgende chmod-Befehl ergibt diese Berechtigung: rw-----

```
$ chmod u+rw Dateien
```

Der folgende chmod-Befehl ergibt diese Berechtigung: --x--x--x

```
$ chmod a+x Dateien
```

Der folgende chmod-Befehl ergibt diese Berechtigung: r-xr-x---

```
$ chmod ug+rx Dateien
```

Die Verwendung von Buchstaben, um Berechtigungen rekursiv mit `chmod` zu ändern, funktioniert im Allgemeinen besser als die Verwendung von Zahlen, da Sie Bits selektiv ändern können, anstatt alle Berechtigungsbits auf einmal zu ändern. Angenommen, Sie möchten beispielsweise die Schreibberechtigung für "other" entfernen, ohne andere Berechtigungsbits auf einem Satz von Dateien und Verzeichnissen zu ändern. Sie könnten dies wie folgt tun:

```
$ chmod -R o-w $HOME/myapps
```

Dieses Beispiel entfernt rekursiv die Schreibberechtigungen für "other" für alle Dateien und Verzeichnisse unterhalb des Verzeichnisses "myapps". Wenn Sie Zahlen wie 644 verwendet hätten, würde die Ausführungsberechtigung für Verzeichnisse deaktiviert; bei Verwendung von 755 würde die Ausführungsberechtigung für reguläre Dateien aktiviert. Durch Verwendung von `o-w` wird nur ein Bit deaktiviert, und alle anderen Bits bleiben unberührt.

## Festlegen der Standarddateiberechtigung mit `umask`

Wenn Sie eine Datei als normaler Benutzer erstellen, erhält sie standardmäßig die Berechtigung `rw-rw-r--`. Ein Verzeichnis erhält die Berechtigung `rw-rwxr-x`. Für den Root-Benutzer sind die Datei- und Verzeichnisberechtigungen `rw-r--r--` bzw. `rw-r-xr-x`. Diese Standardwerte werden durch den Wert von `umask` bestimmt. Geben Sie `umask` ein, um zu sehen, was Ihr `umask`-Wert ist. Zum Beispiel:

```
$ umask
0002
```

Wenn Sie die führende Null vorerst ignorieren, maskiert der `umask`-Wert, was als vollständig geöffnete Berechtigungen für eine Datei 666 oder ein Verzeichnis 777 angesehen wird. Der `umask`-Wert von 002 führt zu Berechtigungen für ein Verzeichnis von 775 (`rw-rwxr-x`). Derselbe `umask`-Wert führt zu einer Dateiberechtigung von 644 (`rw-rw-r--`). (Ausführungsberechtigungen sind standardmäßig für reguläre Dateien deaktiviert.)

Um Ihren `umask`-Wert vorübergehend zu ändern, führen Sie den Befehl `umask` aus. Versuchen Sie dann, einige Dateien und Verzeichnisse zu erstellen, um zu sehen, wie sich der `umask`-Wert darauf auswirkt, wie Berechtigungen festgelegt werden.

Zum Beispiel:

```
$ umask 777 ; touch file01 ; mkdir dir01 ; ls -ld file01 dir01
d----- . 2 joe joe 6 Dec 19 11:03 dir01
----- . 1 joe joe 0 Dec 19 11:02 file01
$ umask 000 ; touch file02 ; mkdir dir02 ; ls -ld file02 dir02
drwxrwxrwx. 2 joe joe 6 Dec 19 11:00 dir02/
-rw-rw-rw-. 1 joe joe 0 Dec 19 10:59 file02
$ umask 022 ; touch file03 ; mkdir dir03 ; ls -ld file03 dir03
drwxr-xr-x. 2 joe joe 6 Dec 19 11:07 dir03
-rw-r--r--. 1 joe joe 0 Dec 19 11:07 file03
```

Wenn Sie Ihren umask-Wert dauerhaft ändern möchten, fügen Sie einen umask-Befehl zur Datei .bashrc in Ihrem Home-Verzeichnis (nahe dem Ende dieser Datei) hinzu. Das nächste Mal, wenn Sie eine Shell öffnen, wird Ihr umask auf den von Ihnen gewählten Wert gesetzt.

## Ändern der Dateibesitzrechte

Als normaler Benutzer können Sie den Besitz von Dateien oder Verzeichnissen nicht ändern, um sie einem anderen Benutzer zuzuweisen. Sie können den Besitz als Root-Benutzer ändern. Angenommen, Sie haben eine Datei namens memo.txt im Home-Verzeichnis des Benutzers joe erstellt, während Sie als Root-Benutzer waren. So könnten Sie es ändern, damit es joe gehört:

```
# chown joe /home/joe/memo.txt
# ls -l /home/joe/memo.txt
-rw-r--r--. 1 joe root 0 Dec 19 11:23 /home/joe/memo.txt
```

Beachten Sie, dass der chown-Befehl den Benutzer in joe geändert hat, aber die Gruppe als root belassen hat. Um sowohl Benutzer als auch Gruppe in joe zu ändern, könnten Sie stattdessen Folgendes eingeben:

```
# chown joe:joe /home/joe/memo.txt
# ls -l /home/joe/memo.txt
-rw-r--r--. 1 joe joe 0 Dec 19 11:23 /home/joe/memo.txt
```

Der chown-Befehl kann auch rekursiv verwendet werden. Die Verwendung der Option -R ist hilfreich, wenn Sie eine ganze Verzeichnisstruktur in den Besitz eines bestimmten Benutzers ändern müssen. Angenommen, Sie haben ein USB-Laufwerk eingesteckt, das im Verzeichnis /media/myusb gemountet ist, und Sie möchten den Inhalt dieses Laufwerks dem Benutzer joe vollständig zuweisen, könnten Sie Folgendes eingeben:

```
# chown -R joe:joe /media/myusb
```

## Verschieben, Kopieren und Löschen von Dateien

Befehle zum Verschieben, Kopieren und Löschen von Dateien sind ziemlich unkompliziert. Verwenden Sie den Befehl mv, um den Speicherort einer Datei zu ändern. Verwenden Sie den Befehl cp, um eine Datei von einem Ort an einen anderen zu kopieren. Verwenden Sie den Befehl rm, um eine Datei zu entfernen. Diese Befehle können auf einzelne Dateien und Verzeichnisse angewendet oder rekursiv verwendet werden, um viele Dateien und Verzeichnisse auf einmal zu bearbeiten. Hier sind einige Beispiele:

```
$ mv abc def
$ mv abc ~
$ mv /home/joe/mymemos/ /home/joe/Documents/
```

Der erste mv-Befehl verschiebt die Datei abc in die Datei def im selben Verzeichnis (im Wesentlichen umbenennen), während der zweite mv-Befehl die Datei abc in Ihr Home-Verzeichnis (~) verschiebt. Der nächste mv-Befehl verschiebt das Verzeichnis mymemos (und alle seine Inhalte) in das Verzeichnis /home/joe/Documents.

Standardmäßig überschreibt der mv-Befehl vorhandene Dateien, wenn die Datei, die Sie verschieben, bereits existiert. Viele Linux-Systeme aliasen jedoch den mv-Befehl, sodass er die Option -i verwendet (was bewirkt, dass mv Sie auffordert, bevor vorhandene Dateien überschrieben werden). So überprüfen Sie, ob das auf Ihrem System zutrifft:

```
$ alias mv
alias mv='mv -i'
```

Hier sind einige Beispiele für die Verwendung des cp-Befehls zum Kopieren von Dateien von einem Ort an einen anderen:

```
$ cp abc def
$ cp abc ~
$ cp -r /usr/share/doc/bash-completion* /tmp/a/
$ cp -ra /usr/share/doc/bash-completion* /tmp/b/
```

Der erste Kopierbefehl (cp) kopiert abc in den neuen Namen def im selben Verzeichnis, während der zweite abc in Ihr Home-Verzeichnis kopiert (~), wobei der Name abc beibehalten wird. Die beiden rekursiven (-r) Kopien kopieren das Verzeichnis bash-completion und alle darin enthaltenen Dateien zuerst in die neuen Verzeichnisse /tmp/a/ und /tmp/b/. Wenn Sie ls -l auf diese beiden Verzeichnisse ausführen, sehen Sie, dass für den mit der Archivoption (-a) ausgeführten cp-Befehl Datum/Uhrzeit-Stempel und Berechtigungen durch die Kopie beibehalten werden. Ohne das -a werden aktuelle Datum/Uhrzeit-Stempel verwendet, und Berechtigungen werden durch Ihre umask festgelegt.

Der cp-Befehl wird in der Regel auch mit der Option -i aliasiert, um zu verhindern, dass Sie versehentlich Dateien überschreiben.

Wie bei den cp- und mv-Befehlen wird auch rm normalerweise mit der Option -i aliasiert. Dies kann Schäden durch eine versehentlich rekursive Entfernung (-r) verhindern. Hier sind einige Beispiele für den rm-Befehl:

```
$ rm abc
$ rm *
```

Der erste Löschbefehl löscht die Datei abc; der zweite löscht alle Dateien im aktuellen Verzeichnis (außer Verzeichnissen und/oder Dateien, die mit einem Punkt beginnen). Wenn Sie ein Verzeichnis entfernen möchten, müssen Sie die rekursive (-r) Option zu rm verwenden oder, für ein leeres Verzeichnis, den rmdir-Befehl verwenden. Betrachten Sie die folgenden Beispiele:

```
$ rmdir /home/joe/nothing/
$ rm -r /home/joe/bigdir/
$ rm -rf /home/joe/hugedir/
```

Der `rmdir`-Befehl im obigen Code entfernt nur das Verzeichnis (`nothing`), wenn es leer ist. Das `rm -r` Beispiel entfernt das Verzeichnis `bigdir` und alle seine Inhalte (Dateien und mehrere Ebenen von Unterverzeichnissen), aber es fragt Sie vor jedem Entfernen. Wenn Sie die Force-Option (`-f`) hinzufügen, werden das Verzeichnis `hugedir` und all seine Inhalte sofort ohne Nachfrage entfernt.

## Achtung

Wenn Sie die `-i` Option bei den `mv`-, `cp`- und `rm`-Befehlen außer Kraft setzen, riskieren Sie, versehentlich einige (oder viele) Dateien zu entfernen. Die Verwendung von Platzhaltern (wie `*`) und keinem `-i` macht Fehler noch wahrscheinlicher. Das gesagt, manchmal möchten Sie nicht durch jede Datei gehen, die Sie löschen. Sie haben andere Optionen wie folgt:

- Wie bereits mit der `-f` Option erwähnt, können Sie `rm` zwingen, ohne Nachfrage zu löschen. Eine Alternative ist es, `rm`, `cp` oder `mv` mit einem Backslash davor auszuführen (`\rm bigdir`). Der Backslash führt dazu, dass jeder Befehl nicht aliased wird.
- Eine weitere Alternative bei `mv` ist die Verwendung der `-b` Option. Mit `-b` wird, wenn eine Datei mit demselben Namen am Zielort existiert, eine Sicherungskopie der alten Datei erstellt, bevor die neue Datei dort verschoben wird.

## Zusammenfassung

Befehle zum Navigieren im Dateisystem, zum Kopieren von Dateien, zum Verschieben von Dateien und zum Entfernen von Dateien gehören zu den grundlegendsten Befehlen, die Sie benötigen, um von der Shell aus zu arbeiten. Dieses Kapitel behandelt viele Befehle zum Navigieren und Manipulieren von Dateien sowie Befehle zum Ändern von Besitz und Berechtigungen.

Im nächsten Kapitel werden Befehle zum Bearbeiten und Suchen von Dateien beschrieben. Diese Befehle umfassen die Texteditoren `vim/vi`, den `find`-Befehl und den `grep`-Befehl.

## Übungen

Verwenden Sie diese Übungen, um Ihr Wissen über effiziente Möglichkeiten zur Navigation im Linux-Dateisystem und zum Arbeiten mit Dateien und Verzeichnissen zu testen. Versuchen Sie, wenn möglich, Verknüpfungen zu verwenden, um so wenig wie möglich zu tippen, um die gewünschten

Ergebnisse zu erzielen. Diese Aufgaben gehen davon aus, dass Sie ein Fedora- oder Red Hat Enterprise Linux-System verwenden (obwohl einige Aufgaben auch auf anderen Linux-Systemen funktionieren). Wenn Sie stecken bleiben, sind Lösungen zu den Aufgaben im Anhang B aufgeführt (obwohl es in Linux oft mehrere Möglichkeiten gibt, eine Aufgabe zu erledigen).

1. Erstellen Sie im Home-Verzeichnis ein Verzeichnis namens Projekte. Im Projekte-Verzeichnis erstellen Sie neun leere Dateien mit den Namen house1, house2, house3 usw. bis house9. Angenommen, es gibt viele andere Dateien in diesem Verzeichnis, überlegen Sie sich ein einziges Argument für ls, das nur diese neun Dateien auflistet.
2. Erstellen Sie den Verzeichnispfad \$HOME/projects/houses/doors/. Erstellen Sie die folgenden leeren Dateien innerhalb dieses Verzeichnispfads (versuchen Sie, absolute und relative Pfade von Ihrem Home-Verzeichnis aus zu verwenden):

```
$HOME/projects/houses/bungalow.txt  
$HOME/projects/houses/doors/bifold.txt  
$HOME/projects/outdoors/vegetation/landscape.txt
```

3. Kopieren Sie die Dateien house1 und house5 in das Verzeichnis \$HOME/projects/houses/.
4. Kopieren Sie das Verzeichnis /usr/share/doc/initscripts\* rekursiv in das Verzeichnis \$HOME/projects/. Beibehalten Sie die aktuellen Datum-/Uhrzeitstempel und Berechtigungen.
5. Listen Sie den Inhalt des Verzeichnisses \$HOME/projects/ rekursiv auf. Leiten Sie die Ausgabe an das less-Kommando weiter, damit Sie durch die Ausgabe blättern können.
6. Entfernen Sie die Dateien house6, house7 und house8, ohne dazu aufgefordert zu werden.
7. Verschieben Sie die Dateien house3 und house4 in das Verzeichnis \$HOME/projects/houses/doors/.
8. Entfernen Sie das Verzeichnis \$HOME/projects/houses/doors und dessen Inhalt.
9. Ändern Sie die Berechtigungen der Datei \$HOME/projects/house2 so, dass sie vom Benutzer, der die Datei besitzt, gelesen und beschrieben werden kann, nur vom Gruppenmitglied gelesen werden kann und keine Berechtigung für andere besteht.
10. Ändern Sie rekursiv die Berechtigungen des Verzeichnisses \$HOME/projects/ so, dass niemand Schreibberechtigungen für Dateien oder Verzeichnisse unterhalb dieses Punktes im Dateisystem hat.