Benutzung wichtiger Konsolenbefehle auf dem Raspberry Pi und anderen Linux Systemen

Zu den wichtigen Konsolenbefehle, die man auf dem Raspberry Pi benötigt, werden hier neben einer kleinen Erklärung auch Beispiele für Aufrufe angegeben. Dabei wird auch die Bedeutung der verschiedenen Parameter für die einzelnen Befehle erklärt.

1 Befehle zu Dateien und Verzeichnisse

Alle der folgenden Befehle können auf eigentlich alle Dateien bzw. Verzeichnisse angewandt werden, solange der Benutzer nicht durch fehlende Rechte eingeschränkt wird. Sie sind dabei unabhängig vom Typ der Dateien, wie z. B. Text oder Bilddateien.

1.1 Verzeichnis wechseln mit cd

Zum wechseln des Verzeichnisses wird der Befehl `cd` genutzt, der für »change directory« steht. Das gewünschte Verzeichnis wird dabei relativ, also vom aktuellen Verzeichnis aus, angegeben oder absolut mit dem kompletten Pfad. Eine absolute Pfadangabe ist immer daran zu erkennen, dass sie mit einem `/` beginnt. In fast allen Kommandozeilen wird nach dem Benutzer und dem Rechnername auch das aktuelle Verzeichnis mit kompletten Pfad vor dem Promt `$` angezeigt:

```
pi@raspberrypi:/usr/share $
```

Befindet man sich aber im Homeverzeichnis bzw. dessen Unterverzeichnisse, so wird dieses mit einer Tilde ~ abgekürzt:

```
pi@raspberrypi:~/Documents $
```

`cd foo`	Wechselt in das Unterverzeichnis `foo` des aktuellen Verzeichnisses.
`cd foo/bar`	Wie oben, nur dass in das Unterunterverzeichnis gewechselt wird.
`cd ..`	Wechselt in das Verzeichnis eine Ebene höher vom aktuellen Verzeichnis aus.
`cd ../../foo`	Wechselt zwei Ebenen höher und von dort in das dortige Unterverzeichnis `foo`.
`cd`	Es wird in das Homeverzeichnis des aktuellen Benutzers gewechselt.
`cd ~`	Analog zur Eingabe `cd`.
`cd ~/Bilder`	Wechselt direkt in das Unterverzeichnis `Bilder` des Homeverzeichnis gewechselt.
`cd ~foo`	Wechselt in das Homeverzeichnis des Benutzers foo.
`cd /etc`	Wechselt in das Verzeichnis `/etc` mit dem absoluten Pfad.

1.2 Verzeichnisinhalt anzeigen mit `ls`

Durch den Befehl `ls` wird der Inhalt des aktuellen oder angegebenen Verzeichnis ausgegeben. Hinter dieser Abkürzung steht der Begriff »list«. Durch entsprechende Parameter lässt sich die Ausgabe den eigenen Wünschen hin anpassen, so dass nur bestimmte Dateien angezeigt werden oder auch weitere Informationen als die Dateinamen.

`ls`	Die einfache Form liefert die Dateien und Verzeichnisse in mehreren Spalten nebeneinander.
`ls -l`	Die Dateien werden untereinander mit Informationen zu Besitzer, Gruppe, Rechten, Größe und Änderungsdatum ausgegeben.
`ls -a`	Gibt auch versteckte Dateien und Verzeichnisse mit aus.
`ll`	Ist die Kurzform für `ls -la`
`ls -lh`	Die Dateigrößen werden statt in Byte in einer passenden Größe wie Kilobyte oder Megabyte angegeben.
`ls *.txt`	Nur Dateien mit der Endung `txt` werden ausgegeben.
`ls foo/`	Auflistung des Inhalts des Unterverzeichnisses `foo`.

1.3 Kopieren mit `cp`

Der Befehle `cp`, kurz für »copy«, kopiert eine Datei an einen anderen Ort oder mehrere Dateien in ein Verzeichnis.

`cp foo.bar foo2.bar`	Kopiert die Datei mit dem Namen `foo.bar`, zu einer Datei mit dem Namen `foo2.bar` oder in ein Verzeichnis mit dem Namen `foo2.bar`, wenn es existiert.
`cp foo.bar ../`	Kopiert die Datei mit dem Namen `foo.bar` in das Verzeichnis eine Ebene höher.
`cp *.txt neu`	Kopiert alle Dateien, die auf `.txt` enden in das Unterverzeichnis `neu`.
`cp al* neu`	Funktioniert analog zum oberen, kopiert aber alle Dateien, die mit `al` beginnen.
`cp * neu`	Kopiert alle Dateien im aktuellen Verzeichnis in das Unterverzeichnis `neu`.
`cp bar/* .`	Kopiert alles aus dem Unterverzeichnis `bar` in das aktuelle Verzeichnis, was durch den Punkt angegeben ist.
`cp -r foo bar`	Kopiert das Verzeichnis `foo` mit allen Dateien und Unterverzeichnissen in ein Verzeichnis `bar`, bzw. dort hinein, falls `bar` bereits existiert.
`cp -r foo/* bar`	Kopiert den gesamten Inhalt mit ggf. Unterverzeichnissen in das Verzeichnis `bar`

1.4 Verschieben und umbennen mit `mv`

Sehr ähnlich zum Kopieren lassen sich mit dem Befehl `mv` Dateien bzw. Verzeichnisse umbenennen oder verschieben. Dabei ist `mv` die Kurzform von »move«.

`mv foo bar`	Ändert den Namen der Datei bzw. des Verzeichnisses `foo` in `bar`.
`mv *.txt bar`	Verschiebt alle Dateien, die auf `.txt` enden, in das Verzeichnis `bar`.
`mv ../bar .`	Verschiebt aus dem Überverzeichnis das Verzeichnis `bar` in das aktuelle Verzeichnis, was durch den Punkt angeben ist.

1.5 Verzeichnisse erstellen mit `mkdir`

Um neue Verzeichnisse erstellen zu können, wird der Befehl `mkdir` genutzt, mit der Angabe des neuen Verzeichnisnamen. Die Abkürzung »mkdir« steht dabei für den Ausdruck »make directory«.

`mkdir foo`	Erstellt das Unterverzeichnis `foo` im aktuellen Verzeichnis.
`mkdir -p bar/foo`	Im aktuellen Unterverzeichnis wird der Pfad `bar/foo` erstellt. Durch den Parameter `-p` werden auch alle Verzeichnisse im Pfad erstellt, die bisher noch nicht existieren.
`mkdir ~/bar`	Erstellt, unabhängig vom aktuellen Pfad, das Verzeichnis `bar` im Homeverzeichnis.

1.6 Verzeichnisse löschen mit `rmdir`

Das Gegenteil von `mkdir` ist `rmdir`. Dabei muss beachtet werden, dass das Verzeichnis auch leer ist.

`rmdir foo`	Entfernt das leere Verzeichnis `foo` im aktuellen Verzeichnis.
`rmdir -p bar/foo`	Entfernt die Verzeichnisse `bar/foo` und `bar` im aktuellen Verzeichnis, wenn beide leer sind.

1.7 Dateien löschen mit `rm`

Zum Löschen von Dateien wird `rm` genutzt. Dieses kann dabei auch genutzt werden, um Verzeichnisse inklusive ihrem Inhalt zu entfernen.

`rm bar`	Entfernt die Datei `bar` im aktuellen Verzeichnis. Handelt es sich bei `bar` um ein Verzeichnis, so wird eine Fehlermeldung ausgegeben.
`rm foo bar`	Entfernt die Dateien `foo` und `bar`.
`rm *.tmp`	Entfernt alle Dateien, die auf `.tmp` enden.
`rm -r foo`	Entfernt das Verzeichnis `foo` mit allen Dateien und Unterverzeichnissen.

2 Befehle zur Zugriffssteuerung auf Dateien und Verzeichnisse

Jeder Datei und jedem Verzeichnis sind auf einem Linuxsystem jeweils eine Benutzer und eine Gruppe als Besitzer zugeordnet. Außerdem gibt es verschiedene Rechte, die ein Benutzer als Besitzer, Mitglied der Besitzergruppe oder als sonstiger Benutzer haben kann. Dazu gehören die Rechte etwas zu verändern (w für write), etwas zu lesen (r für read) oder eine Datei auszuführen bzw. in ein Verzeichnis zu wechseln (x für execute). Diese Zugriffsregeln für eine Datei bzw. Verzeichnis lassen sich durch drei verschiedene Befehle ändern. Dabei sind, je nach Änderung auch administrative Rechte nötig.

2.1 Ändern des Besitzers mit `chown`

Mit `chown` (change owner) lässt sich der Besitzer von Dateien oder Verzeichnissen ändern. Er kann aber auch dazu genutzt werden Besitzer und Gruppe gleichzeitig zu ändern.

`chown pi foo.bar`	Ändert den Besitzer der Datei `foo.bar` auf den Benutzer `pi`.
`chown pi:user foo.bar`	Ändert von der Datei `foo.bar` den Benutzer auf `pi` und die Gruppe auf `user`.
`chown pi *`	Ändert den Besitzer alle Dateien im aktuellen Verzeichnis.
`chown -R pi .`	Ändert den Besitzer des aktuellen Verzeichnisses.
`chown -R pi *`	Ändert den Besitzer alle Dateien im aktuellen Verzeichnis und allen Unterverzeichnissen.

2.2 Ändern der Gruppe von Dateien mit `chgrp`

Analog zum Ändern des Besitzers lässt sich mit `chgrp` (change group) die Besitzergruppe von Dateien oder Verzeichnissen ändern.

`chgrp user foo.bar`	Ändert die Gruppe der Datei `foo.bar` auf die Gruppe `user`.
`chgrp user *`	Ändert die Gruppe alle Dateien im aktuellen Verzeichnis.
`chgrp -R user .`	Ändert die Gruppe des aktuellen Verzeichnisses.
`chgrp -R user *`	Ändert die Gruppe alle Dateien im aktuellen Verzeichnis und allen Unterverzeichnissen.

2.3 Rechte ändern durch `chmod`

Bei den Rechten für Dateien und Verzeichnissen wird in erster Linie zwischen schreiben (w), lesen(r) und ausführen(x) unterschieden. Dabei können diese Rechte jeweils für den Benutzer (u wie user), die Gruppe (g wie group) und alle anderen (o wie other) gesetzt

werden. Dieses geschieht mit Hilfe von `chmod`, dass für »change mode« steht. Dabei wird angegeben, ob ein Recht hinzugefügt wird (+) oder entfernt (-). Neben der Angabe der symbolischen Darstellung kann auch die oktale Darstellung genutzt werden, die hier nicht näher erläutert wird.

`chmod g+w *.txt`	Setzt die Rechte zum Schreiben aller `txt` Dateien für die Gruppe.
`chmod u+x *.py`	Alle `py` Dateien können vom Besitzer anschließend ausgeführt werden.
`chmod o-r *.log`	Entfernt die Rechte zum Lesen aller `log` Dateien für die alle anderen Benutzer.
`chmod +w foo.bar`	Für alle drei (Benutzer, Gruppe und Andere) wird das Recht zum Schreiben der Datei `foo.bar` gesetzt.
`chmod -R u+w .`	Im aktuellen Verzeichnis und allen Unterverzeichnissen bekommt der Besitzer das Recht zum schreiben.

3 Befehle zur Systemverwaltung

Das System hat auch verschiedene Befehle, die zur Verwaltung benötigt werden. Immer dann, wenn sie Änderungen am System vornehmen, müssen sie dann mit administrativen Rechten, also den sogenannten Root-Rechten ausgeführt werden.

3.1 Befehle ausführen mit Root-Rechten durch `sudo`

Wenn man als Benutzer berechtigt ist, so kann man mit Hilfe von `sudo`, das für »super user do« steht, Befehle mit root-Rechten ausführen. Dazu muss nur der Befehl, den man ausführen will, direkt hinter `sudo` angegeben werden. Die Berechtigung erhält man durch einen Eintrag in der Datei `/etc/sudoers`, die man aber aus Sicherheitsgründen nur mit `visudo` bearbeiten sollte. Anders kann es sein, dass man alle Benutzer des Systems von den administrativen Rechten aussperrt.

`sudo rm foo.bar`	Entfernt die Datei `foo.bar` auch wenn der eigentliche Benutzer dafür keine Rechte hätte.
`sudo apt update`	Aktualisiert die Paketliste des Systems, für das man administrative Rechte benötigt.
`sudo su -`	Wechselt in eine Befehlskonsole, in der man sich mit administrativen Rechten bewegt.

3.2 Informationen zum System mit `uname`

Mit Hilfe von `uname` lassen sich Informationen, wie die Version des aktuell genutzt Kernels ausgeben oder der Name der Maschine ausgeben.

`uname`	Gibt an, was allgemein für ein Kernel verwendet wird.
`uname -n`	Gibt den Namen der Maschine aus.
`uname -r`	Gibt die Release-Nummer des aktuellen Kernels aus.
`uname -a`	Gibt alle Informationen zum System inklusive der oben genannten aus.

3.3 Zeit auslesen und setzen mit `date`

Durch den Befehle `date` hat man die Möglichkeit die aktuelle Uhrzeit des Systems sich angeben zu lassen. Es ist damit aber auch möglich, die Uhrzeit von Hand zu setzen. Dieses kann beim Raspberry Pi des öfteren der Fall sein, da dieser über keine Hardwareuhr verfügt und daher ohne Strom die Uhrzeit nicht weiterläuft.

`date`	Gibt das aktuelle Systemdatum und Uhrzeit aus.
`date -s "2017-12-24 18:30"`	Setzt die Uhrzeit des Systems auf das angegebene Datum mit der entsprechenden Uhrzeit. Dieser Vorgang benötigt Root-Rechte voraus.

4 Nützliche Hilfsprogramme

Neben den Befehlen in der Konsole, die direkte Auswirkungen auf das Dateisystem oder das System allgemein haben, gibt es eine Reihe von weiteren nützlichen Programmen, die in der Konsole genutzt werden können. Dazu gehören neben Texteditoren auch Möglichkeiten zu suchen und den Inhalt von Dateien auszugeben.

4.1 Texteditoren

Es gibt viele verschiedene Texteditoren für die Befehlskonsole. Am häufigsten werden dabei `nano` und `vi` genutzt. Beide Editoren können entweder direkt aufgerufen werden oder mit Angabe der Datei, die bearbeitet werden soll.

`nano`	Startet den Texteditor mit einem leeren Text.
`nano foo.bar`	Die Datei `foo.bar` wird zum Bearbeiten geöffnet.
`vi`	Öffnet den Texteditor mit einem leeren Text.
`vi foo.bar`	Der Texteditor öffnet die Datei `foo.bar`.

Bei der Bedienung ähnelt `nano` vielen bekannten graphischen Texteditoren. Die Tastaturcodes für Möglichkeiten wie Speichern, Öffnen und Beenden werden in den unteren Zeilen angegeben. Zu den dort angegebenen Tasten muss jeweils die Strg-Taste gedrückt gehalten werden, wie z. B. Strg+x zum Beenden.

Beim `vi` wird zwischen einem Befehlsmodus und einem Eingabemodus unterschieden. Der Editor öffnet im Befehlsmodus, indem verschiedene Befehle zum Öffnen, Speichern usw. eingegeben werden können. Möchte man in den Eingabemodus wechseln um am Text etwas zu ändern, so muss man entweder die Einfg.-Taste oder die Taste i drücken.

Aus dem Eingabemodus gelangt man wieder in den Befehlsmodus durch das Drücken der ESC-Taste. Nachfolgend sind die wichtigsten Befehle für den `vi` aufgeführt:

`:w`	Speichert die Datei.
`:q`	Beendet den `vi`.
`:q!`	Der Editor wird ohne Speichern beendet, auch wenn Änderungen am Text vorhanden sind.
`:wq`	Speichert die Datei und beendet den Editor.
`:12`	Springt mit dem Cursor in Zeile 12.
`/foo`	Sucht nach `foo` im Text und springt mit dem Cursor zum nächsten Vorkommen.
`u`	Macht die letzte Änderung rückgängig.

4.2 Mit `passwd` das Passwort ändern

Zum Ändern des Passworts dient den fast gleichlautenden Befehl `passwd`. Bei diesem ist zu beachten, dass bei der Eingabe des Passworts eine Ausgabe auf der Konsole, also auch keine Sternchen, zu sehen ist. Mit entsprechenden Rechten lassen sich darüber auch die Passwörter von anderen Benutzern ändern. Die Eingabe des neuen Passworts muss automatisch wiederholt werden, um mögliches Vertippen zu vermeiden.

`passwd`	Ändert das Passwort für den angemeldeten Benutzer.
`passwd alice`	Mit administrativen Rechten wird das Passwort des Benutzers `alice` geändert.

4.3 Suchen und Durchsuchen von Dateien mit Hilfe von `find` und `grep`

Eine Möglichkeit nach Dateien aufgrund verschiedener Kriterien suchen bietet `find`. Diese Kriterien können z. B. der Name oder das Datum der letzten Änderung sein. In jedem Fall muss als erstes der Pfad angegeben werden, unterhalb dessen gesucht werden soll. Dabei symbolisiert der Punkt das aktuelle Verzeichnis.

`find .`	Liefert alle Dateien im aktuellen Verzeichnis und dessen Unterverzeichnissen.
`find . -name "foo*"`	Sucht alle Dateien, die mit `foo` beginnen im aktuellen Verzeichnis.
`find bar -name "foo*"`	Im Verzeichnis `bar` wird nach entsprechenden Dateien gesucht.
`find . -name "foo*" -or -name "bar*"`	Listet alle Dateien die mit `foo` oder `bar` beginnen.
`find . -name "foo*" -and -name "*bar"`	Dateien die mit `foo` beginnen und `bar` enden werden geuscht.
`find . -cmin 30`	Alle Dateien, die in den letzten 30 Minuten geändert wurden werden gesucht.

Sucht man stattdessen nach einer Datei aufgrund ihres Inhalts oder nach einer besonderen Stelle innerhalb einer Datei, so sorgt `grep` für Abhilfe. Damit kann man nach einer Zeichenkette innerhalb einer Datei oder mehreren Dateien suchen.

`grep foo bar.txt`	Gibt alle Vorkommnisse von `foo` in der Datei `bar.txt` aus.
`grep foo *`	Es wird nach `foo` in allen Dateien im Verzeichnis gesucht.
`grep -n foo *`	Bei der Stelle in den Dateien wird auch die Zeilennummer mit angegeben.
`grep -r foo *`	Es werden auch Dateien in Unterverzeichnissen mit in die Suche einbezogen.

4.4 Ausgeben einer Datei mit `cat` und `tail`

Möchte man den Inhalt einer Datei in der Befehlskonsole ausgeben, so ist `cat` das Mittel der Wahl. Besonders nützlich ist es, wenn die Datei auch von anderen Programmen geöffnet ist oder die Ausgabe weiterverwendet werden soll.

`cat foo.bar`	Gibt den Inhalt der Datei `foo.bar` aus.
`cat foo.bar foo2.bar`	Hängt die beiden Dateien aneinander bei der Augabe.

Wenn man z. B. eine Log-Datei betrachten möchte, in der die Ausgaben zum System oder anderen Abläufen geschrieben werden, dann ist `tail` sehr hilfreich. Es gibt dann die letzten zehn Zeilen der Datei aus. Es lässt sich aber auch dazu nutzen, dass immer die neusten Zeilen ausgegeben werden, die bei der Datei an das Ende angehangen werden.

`tail foo.bar`	Gibt die letzten zehn Zeilen von `foo.bar` aus.
`tail -n 15 foo.bar`	Es werden die letzten fünfzehn Zeilen ausgegeben.
`tail -f foo.bar`	Nachdem die ersten zehn Zeilen ausgeben werden wird diese Ausgabe immer dann ergänzt, wenn an die Datei weitere Zeilen angehangen werden.