

Praktischer Einstieg in Linux



Dennis Schulmeister-Zimolong, dhbw@windows3.de

Dieses Aufgabenblatt soll dir helfen, dich im Projekt mit Linux zurechtzufinden. Es enthält daher verschiedene Aufgaben, um Linux kennen zu lernen und eine kleine Firmware für den Raspberry Pi zu bauen. Nachdem du das Blatt bearbeitet hast kannst du ...

- Dateien und Ordner mit der Konsole verwalten
- Befehle in der Konsole ausführen
- Einfache Shell-Skripte schreiben
- Mit Debootstrap eine Firmware für den Raspberry Pi bauen

Benötigte Materialien

Um das Aufgabenblatt durchzuarbeiten benötigst du folgende Hardware:

- Einen Raspberry Pi mit SD-Karte
- Zugriff auf einen Bildschirm bzw. den Beamer im Vorlesungsraum
- Einen Laptop mit VirtualBox und SSH (z.B. PuTTY)
- Ein Netzkabel zur Verbindung der beiden Rechner

Außerdem benötigst du folgende Unterlagen:

- Das Skript *Linux und Buildroot*¹
- Linux Command Reference: [Learning the Shell](#)

Aufgabe 1: Vorbereitung

Für die Bearbeitung der folgenden Aufgaben benötigst du die am Anfang des Projekts zur Verfügung gestellte virtuelle Maschine. Falls diese noch nicht auf deinem Laptop eingerichtet ist, folge erst der Anleitung aus dem Vorlesungsskript.

Aufgabe 2: Erste Schritte mit Linux

a) Deine ersten Gehversuche mit Linux kannst du auf einem beliebigen Linux-System machen. Der Einfachheit halber verwenden wir hierfür die virtuelle Maschine, mit der wir später auch die Firmware Images bauen werden. Starte daher die VM und melde dich mit SSH (nicht über das Bildschirm-Fenster von VirtualBox!) an. Anschließend führe folgende Aktionen aus:

1. Finde den exakten Pfad des aktuellen Arbeitsverzeichnisses heraus.

Befehl: `pwd` _____

2. Wechsele in das Verzeichnis `~/debian/boot`.

Befehl: `cd ~/debian/boot` _____

3. Lasse dir alle Dateien im aktuellen Verzeichnis anzeigen.

Befehl: `ls -alh` _____

¹ Auch wenn es immer noch nicht fertig ist :-)

IoT und Embedded-Workshop: Praktischer Einstieg in Linux

4. Lasse dir alle Dateien, deren Namen die Buchstaben bcm enthalten, anzeigen.

Befehl: ls *bcm*

5. Wechsel zurück in das Home-Verzeichnis.

Befehl: cd ~

6. Finde heraus, welchen Dateityp die Datei db-dbuid.sh hat.

Befehl: file db-build.sh

7. Gebe den Inhalt der Datei db-dbuid.sh auf der Konsole aus (zwei Möglichkeiten):

Befehl 1: more db-build.sh

Befehl 2: less db-build.sh

b) Herzlichen Glückwunsch. Du weißt nun, wie man sich in der Konsole umschauchen kann. Als nächstes bearbeiten wir ein paar Dateien. Hierfür wechsle in das Verzeichnis ~/shared und führe darin die folgenden Aktionen aus:

1. Lege ein Verzeichnis namens tutorial an:

Befehl: mkdir tutorial

2. Lege die Verzeichniskette tutorial/first-steps/directories an (ein Befehl):

Befehl: mkdir -p shared/tutorial/first-steps/directories

3. Wechsle in das Verzeichnis tutorial und lege darin die Datei readme.txt an:

Befehl 1: cd ~/shared/tutorial

Befehl 2: touch readme.txt

4. Lege eine Datei namens hallo.txt und dem Inhalt „Linux ist klasse!“ an (ohne Editor):

Befehl: echo "Linux ist klasse!" > hallo.txt

5. Hänge eine weitere Textzeile an die Datei hallo.txt an (ohne Editor):

Befehl: echo "Zweite Zeile >> hallo.txt

6. Benenne die Datei hallo.txt in linux.txt um:

Befehl: mv hallo.txt linux.txt

7. Benenne das Verzeichnis tutorial in beispiel um:

Befehl: mv tutorial beispiel

IoT und Embedded-Workshop: Praktischer Einstieg in Linux

8. Erzeuge ein Verzeichnis namens `beispiel2` und kopiere alle `*.txt`-Dateien dorthin:

Befehl 1: `mkdir ~/beispiel2`

Befehl 2: `cp *.txt ~/beispiel2/`

9. Lösche die Verzeichnisse `beispiel` mit all seinen Unterverzeichnissen:

Befehl: `rm -r beispiel`

c) Als nächstes bearbeiten wir ein paar Dateien. Dies ist unter Linux besonders wichtig, da fast alle Einstellungen in Textdateien hinterlegt werden:

1. Öffne die Datei `~/db-build.sh` in einem Texteditor.

Befehl: `nano ~/db-build.sh`

2. Suche nach allen Vorkommen des Worts „`print_info`“ in der Datei:

Tastatureingabe: `"Strg + w" - "print_info"`

3. Beende den Editor ohne zu sichern:

Tastatureingabe: `"Strg + x" - (Bei vorhandenen Änderungen) "n"`

4. Öffne nun die Datei `~/shared/beispiel2/linux.txt` aus der vorherigen Aufgabe:

Befehl: `nano ~/shared/beispiel2/linux.txt`

5. Schreibe mehrere Zeilen in die Datei und speichere sie, ohne den Editor zu verlassen:

Tastatureingabe: `"Test1" + Eingabetaste + "Test2" + "Strg + o" + Eingabetaste`

6. Schneide die ersten drei Zeilen aus und sichere diese in der Zwischenablage:

Tastatureingabe: `3 x "Strg + k"`

7. Füge die eben ausgeschnittenen Zeilen am Ende der Datei wieder ein:

Tastatureingabe: `"Strg + u"`

8. Beende den Editor:

Tastatureingabe: `"Strg + x" + "y" + Eingabetaste`

d) Nun wollen wir ein paar Programme ausführen und uns anschauen, wie man laufende Programme überwachen kann. Immer noch im Verzeichnis `~/shared` führe hierzu folgende Aktionen aus:

Zusatz Aufgabe c) 4.

Da bisheriger Pfad `~/beispiel 2` war muss vor dem Befehl `mv ~/beispiel2 ~/shared` ausgeführt werden, um exakt diese Datei öffnen zu können

IoT und Embedded-Workshop: Praktischer Einstieg in Linux

1. Schreibe den Inhalt des aktuellen Verzeichnisses in die Datei `inhalt.txt`:

Befehl: `ls >> inhalt.txt`

2. Starte den Befehl `less inhalt.txt` im Hintergrund:

Befehl: `less inhalt.txt &`

3. Hole `less` wieder in den Vordergrund:

Befehl: `fg 1`

4. Starte `man man` im Hintergrund:

Befehl: `less man man &` (Ausgabe war: `[8] 23073`)

5. Beende `man`, indem du `SIGTERM` an den Prozess schickst:

Befehl: `kill 23073`

6. Lasse dir alle vom aktuellen Benutzer gestarteten Programm anzeigen:

Befehl: `ps`

7. Lassen dir alle laufenden Programme aller Benutzer anzeigen (zwei Möglichkeiten):

Befehl 1: `ps ax`

Befehl 2: `pstree`

8. Suche nach der Prozess ID des `ssh`-Dienstes:

Befehl: `ps ax | grep ssh`

Aufgabe 3: Eigene Skripte schreiben

Zum Abschluss unserer kleinen Rundumschau schreibe ein Skript, das ein Auswahlmenü zur Automatisierung wichtiger Aufgaben anzeigt.

```
buildroot@debian:~$ ./menu.sh
WICHTIGE AUFGABEN

Buildroot

[1] Buildroot initialisieren (minimale Konfiguration)
[2] Buildroot-Menü aufrufen
[3] Firmware mit Buildroot bauen
[4] sdcard.img nach ~/shared kopieren

Debootstrap

[5] Firmware mit Debootstrap bauen
[6] sdcard.img nach ~/shared kopieren

Sonstiges
```

```
[E] Ende
```

```
Deine Auswahl: _
```

Je Menüpunkt soll eine eigene Funktion aufgerufen werden, welche die eigentlich auszuführenden Befehle enthält. Folgende Befehle sollen die Funktionen enthalten:

```
# Buildroot initialisieren (minimale Konfiguration)
cd ~/buildroot
make BR2_EXTERNAL=./custom O=./make dhw_minimal_defconfig
cd ..

# Buildroot-Menü aufrufen
cd ~/make
make menuconfig

# Firmware mit Buildroot bauen
cd ~/make
make

# sdcard.img nach ~/shared kopieren (Buildroot)
cd ~/make/images/sdcard.img ~/shared

# Firmware mit Debootstrap bauen
cd ~
./db-build.sh

# sdcard.img nach ~/shared kopieren (Debootstrap)
cd ~/out/debian/sdcard.img ~/shared
```

Aufgabe 4: Unsere erste eigene Firmware

Nachdem du dich nun mit Linux vertraut gemacht hast, wollen wir nun unser erstes eigenes Betriebssystem-Image bauen und dieses auf dem Raspberry Pi zum Laufen bringen. Der Einfachheit halber nutzen wir hierfür die Beispielformatierung von Debootstrap.

a) Führe folgende Befehle aus, um das Image zu bauen:

```
$ cd ~
```

```
$ ./db-build.sh
```

Wenn du nach deinem Passwort gefragt bist, gib das Passwort ein, mit dem du dich bereits an der virtuellen Maschine angemeldet hat. Das Skript sollte ca. 10 Minuten laufen und anschließend eine Erfolgsmeldung ausgeben.

b) Kopiere die Datei ~/out/debian/sdcard.img nach ~/shared und übertrage sie anschließend, wie im Skript beschrieben, auf eine SD-Karte.

c) Schließe den Raspberry Pi an einen Bildschirm oder Beamer an und starte von der SD-Karte. Du solltest dich mit dem Benutzernamen `mulder` und dem Kennwort `xfiles` anmelden können.

d) Gebe anschließend den Befehl `sudo poweroff` ein, um den Pi auszuschalten.

Herzlichen Glückwunsch!

Wenn du wirklich bis hierhin alles durchgearbeitet hast, kennst du dich inzwischen ganz gut mit Linux aus. Eurem Projekt steht dann nichts mehr im Weg.



Quelle: [Pixabay "Bellinon"](#)



Das Dokument „Praktischer Einstieg in Linux“ von Dennis Schulmeister-Zimolong ist lizenziert unter einer [Creative Commons Namensnennung - Weitergabe unter gleichen Bedingungen 4.0 International Lizenz](#). E-Mail: dhbw@windows3.de, Webseite: <https://www.wpvs.de>

Lösung Aufgabe 3:

```
#!/bin/sh
#=====
#Testmenü
#=====
set -e
auswahl=
until [ "$auswahl" = "E" ]; do
    cat <<EOF
    HAUPTMENÜ
    Buildroot
    [1] Buildroot initialisieren (minimale Konfiguration)
    [2] Buildroot-Menü aufrufen
    [3] Firmware mit Buildroot bauen
    [4] sdcard.img nach ~/shared kopieren
    Debootstrap
    [5] Firmware mit Debootstrap bauen
    [6] sdcard.img nach ~/shared kopieren
    Sonstiges
    [E] Ende
EOF
    echo -n "Auswahl: "
```