2.3 Lektion 1

Einführung

Zertifikat:	Linux Essentials
Version:	1.6
Thema:	2 Sich auf einem Linux-System zurechtfinden
Lernziel:	2.3 Verzeichnisse verwenden und Dateien auflisten
Lektion:	1 von 2

Dateien und Verzeichnisse

Das Linux-Dateisystem ähnelt Dateisystemen anderer Betriebssysteme, insofern es ***Dateien* und *Verzeichnisse*** enthält. Dateien enthalten Daten wie menschenlesbaren Text, ausführbare Programme oder binäre Daten, die der Computer nutzt. Verzeichnisse dienen dazu, das Dateisystem zu organisieren.

```
$ tree

Documents
├── Mission-Statement.txt
└── Reports
    └── report2018.txt

1 directory, 2 files
```

In diesem Beispiel ist `Documents` ein Verzeichnis, das eine Datei (`Mission-Statement.txt`) und ein *Unterverzeichnis* (`Reports`) enthält, während das `Reports`-Verzeichnis wiederum eine Datei namens `report2018.txt` enthält. Das Verzeichnis `Documents` bezeichnet man als *Elternverzeichnis* des Verzeichnisses `Reports`.

> **Tip** Steht der Befehl `tree` auf Ihrem System nicht zur Verfügung, installieren Sie ihn mit dem Paketmanager Ihrer Linux-Distribution. Schauen Sie in der Lektion über Paketverwaltung nach, wie das geht.

Datei- und Verzeichnisnamen

Datei- und Verzeichnisnamen unter Linux können **Groß- und Kleinbuchstaben, Ziffern**, **Leerzeichen** und ***Sonderzeichen*** enthalten. Da viele **Sonderzeichen jedoch eine besondere Bedeutung** in der Linux-Shell haben, ist es ratsam, bei der Benennung von Dateien oder Verzeichnissen **keine Leerzeichen oder Sonderzeichen zu verwenden**, da Leerzeichen beispielsweise das *Escape-Zeichen* `\` für die korrekte Eingabe benötigen:

```
$ cd Mission\ Statements
```

Beachten Sie den Dateinamen `report2018.txt`. Dateinamen können ein *Suffix* enthalten, das auf einen Punkt (`.`) folgt. Im Gegensatz zu Windows hat dieses **Suffix unter Linux keine besondere Bedeutung**; es dient dem besseren Verständnis. In unserem Beispiel zeigt `.txt` an, dass es sich um eine Klartextdatei handelt, obwohl sie technisch gesehen jede Art von Daten enthalten könnte.

Durch das Dateisystem navigieren

Den aktuellen Standort ermitteln

Da Linux-Shells wie die Bash textbasiert sind, ist es wichtig, beim Navigieren durch das Dateisystem den aktuellen Standort zu kennen, den die *Kommandozeile* oder *Eingabeaufforderung* angibt:

```
user@hostname ~/Documents/Reports $
```

Informationen zu `user` und `hostname` finden Sie in anderen Lektionen. Die Eingabeaufforderung verrät uns, dass unser aktueller Standort das Verzeichnis `Reports` ist. Dieselbe Auskunft gibt auch der Befehl **`pwd`** **(*print working directory*),** der das aktuelle *Arbeitsverzeichnis* ausgibt:

```
user@hostname ~/Documents/Reports $ pwd
/home/user/Documents/Reports
```

Die Beziehung zwischen Verzeichnissen repräsentiert der Schrägstrich oder *Slash* (`/`). Wir wissen, dass `Reports` ein Unterverzeichnis von `Documents` ist, das wiederum ein Unterverzeichnis von `user` ist, das sich in einem Verzeichnis namens `home` befindet. `home` scheint kein Elternverzeichnis zu haben, aber das stimmt nicht: **Das Elternverzeichnis von `home` heißt *root* und wird durch den ersten Slash (`/`) repräsentiert.**

Beachten Sie, dass sich die Ausgabe des Befehls `pwd` geringfügig von dem in der Eingabeaufforderung angegebenen Pfad unterscheidet: Anstelle von **`/home/user`** zeigt die Eingabeaufforderung eine **Tilde (`~`).** Die Tilde ist ein Sonderzeichen, das für das Heimatverzeichnis des Benutzers steht. Die nächste Lektion geht näher darauf ein.

Verzeichnisinhalt auflisten

Den Inhalt des aktuellen Verzeichnisses listet der Befehl `ls` auf:

```
user@hostname ~/Documents/Reports $ ls
report2018.txt
```

Beachten Sie, dass `ls` keine Informationen über das übergeordnete Verzeichnis liefert. `ls` zeigt standardmäßig auch keine Informationen über den Inhalt von Unterverzeichnissen an. `ls` kann nur “sehen”, was sich im aktuellen Verzeichnis befindet.

Aktuelles Verzeichnis wechseln

Die Navigation unter Linux erfolgt in erster Linie mit dem Befehl **`cd`** (*change directory*), er wechselt also das Verzeichnis. Mit dem Befehl `pwd` haben wir bereits ermittelt, dass unser aktuelles Verzeichnis `/home/user/Documents/Reports` ist. Wir wechseln unser aktuelles Verzeichnis, indem wir einen neuen Pfad eingeben:

```
user@hostname ~ $ cd /home/user/Documents
user@hostname ~/Documents $ pwd
/home/user/Documents
user@hostname ~/Documents $ ls
Mission-Statement.txt Reports
```

Von unserem neuen Standort aus können wir `Mission-Statement.txt` und unser Unterverzeichnis `Reports` "sehen", aber nicht den Inhalt des Unterverzeichnisses. Zurück zu `Reports` navigieren wir wie folgt:

```
User@hostname ~/Documents $ cd Reports
user@hostname ~/Documents/Reports $ pwd
/home/user/Documents/Reports
user@hostname ~/Documents/Reports $ ls
report2018.txt
```

Absolute und relative Pfade

Der Befehl `pwd` gibt immer einen *absoluten Pfad* aus, d.h. der Pfad enthält jeden Schritt des Pfades, vom Ausgangspunkt des Dateisystems (`/`) bis zum aktuellen Punkt (`Reports`). Absolute Pfade beginnen immer mit einem `/`.

```
/
└── home
    └── user
        └── Documents
            └── Reports
```

Der **absolute Pfad** enthält alle Informationen, um von überall im Dateisystem zu `Reports` zu gelangen, mit dem Nachteil, dass es mühsam zu tippen ist.

Das zweite Beispiel (`cd Reports`) war deutlich einfacher zu tippen. Dies ist ein Beispiel für einen ***relativen Pfad***. Relative Pfade sind kürzer, beschreiben den Pfad aber immer in Bezug auf die aktuelle Position. Dazu folgende Analogie: Ich besuche Sie in Ihrem Haus, und Sie sagen mir, Ihr Freund wohnt nebenan; ich werde diese Ortsangabe verstehen, weil sie relativ zu meinem aktuellen Standort ist. Wenn Sie mir diese Beschreibung aber am Telefon geben, werde ich das Haus Ihres Freundes nicht finden.

Spezielle relative Pfade

Die Linux-Shell gibt uns Mittel an die Hand, Pfadangaben zu verkürzen. Um den ersten solch speziellen Pfad kennenzulernen, geben wir den Befehl `ls` mit der Option `-a` ein. Diese ändert den Befehl `ls` so, dass *alle* Dateien und Verzeichnisse aufgelistet werden, einschließlich versteckter Dateien und Verzeichnisse:

```
user@hostname ~/Documents/Reports $ ls -a
.
..
report2018.txt
```

Dieser Befehl liefert zwei weitere Ergebnisse: Dies sind spezielle Pfade. Sie stehen nicht für neue Dateien oder Verzeichnisse, sondern Verzeichnisse, die Sie bereits kennen:

`.` Steht für den *aktuellen Standort* (in diesem Fall `Reports`).

`..` Steht für das *Elternverzeichnis* (in diesem Fall `Documents`).

Normalerweise ist es unnötig, den speziellen relativen Pfad für den aktuellen Standort zu nutzen. Es ist einfacher und verständlicher, `report2018.txt` zu tippen als `./report2018.txt`. Aber es gibt Einsatzzwecke für `.`, die Sie in späteren Lektionen

kennenlernen werden. Im Moment konzentrieren wir uns auf den relativen Pfad für das übergeordnete Verzeichnis:

```
user@hostname ~/Documents/Reports $ cd ..
user@hostname ~/Documents $ pwd
/home/user/Documents
```

Das Beispiel von `cd` ist viel einfacher, wenn man `..` anstelle des absoluten Pfades verwendet, und wir können dieses Muster kombinieren, um sehr schnell im Dateibaum nach oben zu navigieren.

```
user@hostname ~/Documents $ cd ../..
$ pwd
/home
```

Geführte Übungen

1. Geben Sie für jeden der folgenden Pfade an, ob es sich um einen *absoluten* oder *relativen* Pfad handelt:

`/home/user/Downloads`	
`../Reports`	
`/var`	
`docs`	
`/`	

2. Betrachten Sie die folgende Dateistruktur. Beachten Sie: Verzeichnisse enden mit einem Slash (`/`), wenn Sie `tree` mit der Option `-F` nutzen. Sie benötigen die entsprechende Berechtigung, um `tree` für das Root-Verzeichnis (`/`) aufzurufen. Die folgende Beispielausgabe zeigt keine vollständige Verzeichnisstruktur. Nutzen Sie sie zur Beantwortung der folgenden Fragen:

```
$ sudo tree -F /

/
├── etc/
│   ├── network/
│   │   └── interfaces
│   ├── systemd/
│   │   ├── resolved.conf
│   │   ├── system/
│   │   ├── system.conf
│   │   ├── user/
│   │   └── user.conf
│   └── udev/
│       ├── rules.d/
│       └── udev.conf
└── home/
    ├── lost+found/
    └── user/
        └── Documents/

12 directories, 5 files
```

Beantworten Sie vor diesem Hintergrund die folgenden Fragen:

Ein Benutzer gibt die folgenden Befehle ein:

```
$ cd /etc/udev
$ ls -a
```

Wie lautet die Ausgabe des Befehls `ls -a`?

3. Geben Sie den jeweils kürzesten Befehl an:
   - Ihr aktueller Standort ist root (`/`). Geben Sie den Befehl an, mit dem ins Verzeichnis `lost+found` im Verzeichnis `home` gelangen (Beispiel):
     ```
     $ cd home/lost+found
     ```
   - Ihr aktueller Standort ist root (`/`). Geben Sie den Befehl an, mit dem Sie ins Verzeichnis `/etc/network/` gelangen.
   - Ihr aktueller Standort ist `/home/user/Documents/`. Geben Sie den Befehl an, mit dem Sie ins Verzeichnis `/etc/` gelangen.
   - Ihr aktueller Standort ist `/etc/systemd/system/`. Geben Sie den Befehl an, mit dem Sie ins Verzeichnis `/home/user/` gelangen.
4. Betrachten Sie die folgenden Befehle:
   ```
   $ pwd
   /etc/udev/rules.d
   $ cd ../../systemd/user
   $ cd ..
   $ pwd
   ```
   Wie lautet die Ausgabe des letzten `pwd` Befehls?

Offene Übungen

1. Angenommen ein Benutzer hat folgende Befehle eingegeben:
   ```
   $ mkdir "this is a test"
   $ ls
   this is a test
   ```
   Mit welchem `cd`-Befehl könnten Sie in dieses Verzeichnis wechseln?
2. Wiederholen Sie dies, aber drücken Sie nach der Eingabe von `cd this` die TAB-Taste. Was erscheint nun in der Befehlszeile?

   Dies ist ein Beispiel für *Autocompletion*, ein wertvolles Werkzeug, das nicht nur Zeit spart, sondern auch Tippfehler vermeidet.
3. Versuchen Sie, ein Verzeichnis zu erstellen, dessen Name das Zeichen `\` enthält. Zeigen Sie den Namen des Verzeichnisses mit `ls` an und löschen Sie das Verzeichnis.

Zusammenfassung

In dieser Lektion haben Sie gelernt:

- Die Grundlagen des Linux-Dateisystems
- Den Unterschied zwischen *Elternverzeichnissen* und *Unterverzeichnissen*
- Den Unterschied zwischen *absoluten* und *relativen* Dateipfaden
- Die speziellen relativen Pfade `.` und `..`

- Das Navigieren durch das Dateisystem mit `cd`
- Das Anzeigen Ihres aktuellen Standorts mit `pwd`
- Das Auflisten *aller* Dateien und Verzeichnisse mit `ls -a`

Die folgenden Befehle wurden in dieser Lektion behandelt:

`cd`: Wechselt das aktuelle Verzeichnis.

`pwd`: Zeigt den Pfad des aktuellen Verzeichnisses.

`ls:` listet den Inhalt eines Verzeichnisses auf und zeige die Eigenschaften von Dateien an.

`mkdir`: Erstellt ein neues Verzeichnis.

`tree`: Zeigt eine hierarchische Auflistung eines Verzeichnisbaums an.