Technische Universität Braunschweig

IAS | INSTITUTE FOR APPLICATION SECURITY

Unix und Shell

Linux für Informatik-Erstsemester

Nico Grashoff, 01.04.2019

- **Grundlagen**
- **Shell**
- **Dateisystem**
- **Navigation**
- **Dokumentation**
- **Programme**
- **Ausblick**

Über diesen Vortrag

- Mitschreiben optional
- Überblick über Begriffe und Funktionsweisen
- Weitere Informationen findet ihr auf Wikipedia
- Hier nur Anwendung, keine Administration
- Teilweise vereinfachte Darstellung zur besseren Verständlichkeit

Voraussetzung

- Generelle Verwendung eines Computers ist vertraut
- Dateien und Verzeichnisse sind bekannte Konzepte
- Maus und Tastatur sind vertraute Eingabegeräte

Unix

- Unix ist ein Betriebssystem, das 1969 von Bell Labs entwickelt wurde
- Heute bezeichnet man als Unix Betriebssysteme, die Unix-Konzepte umsetzen
  - BSD-Systeme (z.B. FreeBSD)
  - macOS, iOS
  - Android und diverse andere *Linux-Distributionen*

Linux

- Als Linux bezeichnet man Betriebssysteme, die den Linux-Kernel verwenden
  - Kernel, englisch für *Kern*
  - Kern eines Betriebssystems
  - Verwaltet die Systemresourcen
- Der Linux-Kernel ist ein quelloffener und kostenloser Kernel
- In diesem Vortrag geht es nicht um den Linux-Kernel, sondern um die Verwendung von Linux-basierten Betriebssystemen, sogenannten *Linux-Distributionen* (kurz *Distro*)

Distribution

Zusammenstellung von Software, die zusammen ein Betriebssystem ergibt.

- Kernel (hier Linux)
- Software
  - Anwendungsprogramme (z.B. Libre Office, Firefox, VLC Media Player)
  - Systemprogramme (z.B. wpa_supplicant, systemd)
  - Benutzeroberfläche (z.B. GNOME, KDE, Unity)
- Paketverwaltung
- Dokumentation

Shell

Einführung in die Shell

- Shell, englisch für *Schale*
- Shell ist allgemeine Bezeichnung für Benutzerschnittstellen eines Betriebssystems
- Gemeint ist meisten (hier ab jetzt auch) die Text-basierte **Kommandozeile**
- Ermöglicht das Ausführen von Programmen
- Vorteile gegenüber graphischen Oberflächen
  - Universell
  - Schnell
  - Stabil
  - Verfügbar

Funktionsweise der Shell

- Die Shell ist ein Programm
- Die Shell wartet auf die Eingabe von Befehlen und führt diese nach Drücken der Enter-Taste aus
- Die Shell ermöglicht es, Befehlseingaben zu verknüpfen und z.B. die Ausgabe eines Programms als Eingabe für ein anderes Programm zu verwenden
- Die Shell ist programmierbar durch sogenannte *Shell-Skripte*

Programme ausführen

- Die Shell signalisiert durch Anzeigen des sogenannten **Shell-Prompts** und eines blinkenden Cursors, das Befehle entgegen genommen werden
- Die Ausgabe des Programms erfolg Zeilenweise unterhalb der Eingabe
- Konvention: Häufig wird in Dokumenten (auch hier) als Shell-Prompt “>” verwendet

Beispiel

```
> date
Sun Mar 31 15:38:02 CEST 2019
>
```

Programme ausführen mit Parametern

- Programme lassen sich per Parameter steuern
- Konvention: Shell-Prompt nach Ausführung wird weggelassen

Beispiel

```
> cal april 2019
     April 2019
Su Mo Tu We Th Fr Sa
    1  2  3  4  5  6
 7  8  9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30
```

Notation von Programmaufrufen

Konvention

`...` hinter einem Argument bedeutet, dass mehrere Argumente dieser Art möglich sind.

Beispiel: `man PAGE...`

Öffnet das Handbuch für PAGE. Mehrere PAGEs sind möglich.

Konvention

Optionale Argumente werden in in eckigen Klammern (`[...]`) angegeben.

Beispiel: `cal [[[DAY] MONTH] YEAR]`

`cal` kann ohne Parameter *oder* nur mit Jahr *oder* mit Monat und Jahr *oder* mit Tag und Monat und Jahr aufgerufen werden.

Programme ausführen mit Optionen

- Zusätzlich zu Parametern kann man das Verhalten vieler Programme durch optionale Optionen steuern
- Oft zwei Schreibweisen: Kurz (beginnen mit einem Bindestricht) und lang (beginnen mit zwei Bindestrichen)

Programm: `wc [OPTIONS]... [FILE]...`

print newline, **w**ord, and byte **c**ounts for each file

Beispiel: Lange Optionen

```
> wc --lines slides.md
137 slides.md
```

Beispiel: Kurze Optionen

```
> wc -l slides.md
137 slides.md
```

Dateisystem

- Das Dateisystem beschreibt (hier) die Struktur von Verzeichnissen und Dateien auf dem System
- Das Dateisystem kann man sich als Baum vorstellen

```
> tree --charset ascii -F nico
nico
|-- music/
|   `-- never_gonna_give_you_up.mp3
|-- pics/
|   |-- kitten.jpg
|   `-- memes/
|       `-- not_available_in_your_country.txt
`-- test.txt
```

Wurzel und Pfade

Root-Verzeichnis

- Die Wurzel des gesamten Verzeichnisbaumes ist das **Root-Verzeichnis** (“/”)
- Alle anderen Verzeichnisse und Dateien befinden sich unterhalb des Root-Verzeichnis
- Gilt auch für externe Dateisysteme, z.B. USB-Sticks

Pfade

- Ein Pfad gibt an, wo sich eine Datei oder ein Verzeichnis im Dateisystem befindet
- Pfade bestehen aus Verzeichnisnamen, getrennt durch Schrägstriche (“/”)
- Pfade die auf eine Datei verweisen enden zusätzlich mit einem Dateinamen

Navigation mit der Shell

- Die Shell kann benutzt werden, um durch das Dateisystem zu navigieren
- Das aktuelle Verzeichnis heißt **working directory** (englisch für *Arbeitsverzeichnis*)
- Programme werden im *working directory* ausgeführt
- Pfadangaben (z.B. als Parameter eines Programms) beziehen sich auf das *working directory*
  - d.h. Pfade sind *relativ*

Relative und absolute Pfade

Relative Pfade

Pfade in der Shell gehen vom *working directory* aus, sie sind daher *relativ* zum aktuellen Verzeichnis.

Absolute Pfade

Pfade lassen sich auch vom Root-Verzeichnis aus beschreiben. Dafür stellt stellt man ihnen ein / voran. Pfadangaben die mit / beginnen sind *absolut*. Absolute Pfade verweisen eindeutig auf ein Verzeichnis oder eine Datei innerhalb des Verzeichnisbaums, unabhängig vom Arbeitsverzeichnis.

Besondere Verzeichnisse

Verzeichnis: /

Das Root-Verzeichnis, die Wurzel des Verzeichnisbaums

Verzeichnis: .

Das Arbeitsverzeichnis, das aktuelle Verzeichnis im Verzeichnisbaum

Verzeichnis: . .

Das nächsthöhere Verzeichnis im Verzeichnisbaum (ausgehend vom Arbeitsverzeichnis)

Benutzerverzeichnis: ~

Privates Verzeichnis des Benutzers.

Befehle zur Navigation und Orientierung

Programm: `cd [DIRECTORY]`

change the working **d**irectory

Programm: `ls [DIRECTORY]`

lst directory contents

Programm: `pwd`

print name of current/**w**orking **dir**ectory

Befehle zum Arbeiten mit Verzeichnissen

Programm: `mkdir DIRECTORY`

make directory

Programm: `rmdir DIRECTORY`

remove empty **dir**ectory

Navigation mit der Shell

Live-Demo

Vergleich zwischen Dateimanager und Shell.

`--help` und `-h`

Programme haben eine eingebaute Hilfe, die sich mit der Option `--help` oder `-h` anzeigen lässt.

Beispiel

```
> cp -h
cp: invalid option -- 'h'
Try 'cp --help' for more information.
```

Beispiel

```
> cp --help
Usage: cp [OPTION]... [-T] SOURCE DEST
  or:  cp [OPTION]... SOURCE... DIRECTORY
  or:  cp [OPTION]... -t DIRECTORY SOURCE...
Copy SOURCE to DEST, or multiple SOURCE(s) to DIRECTORY.
```

Handbuch (man page)

- Das Programm `man` kann benutzt werden um das Handbuch, die sogenannte **manual page** oder kurz **man page** anzuzeigen
- Es gibt nicht nur Hilfe zu Programmen sondern auch zu Konzepten, z.B. öffnet `man hier` eine *man page*, welche die Verzeichnishierarchie unter Linux beschreibt
- Die *man page* für `man` wird angezeigt, wenn `man man` ausführt wird
- In *man pages* scrollt man mit den Pfeiltasten, j und k, oder BildAuf und BildAb

Programme

- Die meisten Befehle, die in der Shell ausgeführt werden sind Programmaufrufe
- Programme werden mit der Distribution ausgeliefert
- Programme lassen sich nachinstallieren (z.B. über den Paketmanager der Distribution)
- Eigene Programme können geschrieben werden
- In diesem Abschnitt werden einige häufig verwendete Programme vorgestellt

touch [OPTION]... FILE...

Auszug man touch

```
touch - change file timestamps
[...]
A FILE argument that does not exist is created empty[...]
```

Beispiel

```
> ls
> touch test
> ls
test
```

rm [OPTION]... [FILE]...

Auszug man rm

```
rm - remove files or directories
[...]
-f, --force
       ignore nonexistent files and arguments, never prompt
-r, -R, --recursive
       remove directories and their contents recursively
```

Beispiel

```
> mkdir verzeichnis
> touch verzeichnis/datei
> rm -rf verzeichnis
```

cat [OPTION]... [FILE]...

Auszug man cat

```
cat - concatenate files and print on the standard output
```

Beispiel

```
> cat /etc/lsb-release
LSB_VERSION=1.4
DISTRIB_ID=Arch
DISTRIB_RELEASE=rolling
DISTRIB_DESCRIPTION="Arch Linux"
```

cp [OPTION]... SOURCE... DIRECTORY

Auszug `man cp`

```
cp - copy files and directories
```

Beispiel

```
> cp /etc/lsb-release .
> ls
lsb-release
```

mv [OPTION]... SOURCE... DIRECTORY

Auszug man mv

```
mv - move (rename) files
```

Beispiel

```
> cp /etc/lsb-release .
> mv lsb-release dings
> ls
dings
```

`echo [OPTION]... [STRING]...`

Auszug `man echo`

```
echo - display a line of text
```

Beispiel

```
> echo Hallo, Welt!
Hallo, Welt!
```

file FILE...

Auszug man file

```
file - determine file type
```

Beispiel

```
> file slides.md slides.pdf
slides.md:  UTF-8 Unicode text
slides.pdf: PDF document, version 1.5
```

which PROGRAMNAME

Auszug man which

```
which - shows the full path of (shell) commands
```

Beispiel

```
> which cp
/usr/bin/cp
> which echo
echo: shell built-in command
```

grep [OPTION]... PATTERNS [FILE]...

Auszug man grep

```
grep - print lines that match patterns
```

Beispiel

```
> grep DESC /etc/lsb-release
DISTRIB_DESCRIPTION="Arch Linux"
```

`vim [OPTION]... [FILE]...`

Auszug `man vim`

```
vim - Vi IMproved, a programmer's text editor
```

Normal- und Insert-Modus

- Im Normal-Modus sind keine Texteingaben möglich, sondern nur vim-Befehle
- Der Insert-Modus erlaubt Texteingaben. Drücke `i` um in den Insert-Modus zu gelangen
- Per Escape-Taste kommt man wieder in den Normal-Modus
- Beenden per `:q!` (Änderungen verwerfen) oder `:wq` (Änderungen speichern) gefolgt von Enter im Normal-Modus

Kein Beispiel, selbst ausprobieren!

```
> vimtutor
```

Ausblick

- Berechtigungen, root-Benutzer
- Pakete installieren
- Graphische Oberflächen
- Umgebungsvariablen
- Pipes und Redirection

Was nun?

- Jetzt: Übungsaufgaben in Kleingruppen
- Linux-Install-Party am Mittwoch um 15 Uhr in IZ161
- Ausprobieren und Spaß haben
  - `https://vim-adventures.com`
  - `https://openvim.com`

Kontakt

- Nico Grashoff
- n.grashoff@tu-bs.de