

Rechnerpraxis

SS2019 | INF 1

Hochschule
für Technik
Stuttgart

03 – Linux Dateisystem & Rechteverwaltung

Benjamin Lebherz

benjamin.lebherz@hft-stuttgart.de

Folien zum Teil von Jan Seedorf & Rainer Keller

Termin verlegt auf
Donnerstags, 17:30 - 19:00

Vorlesungsplan

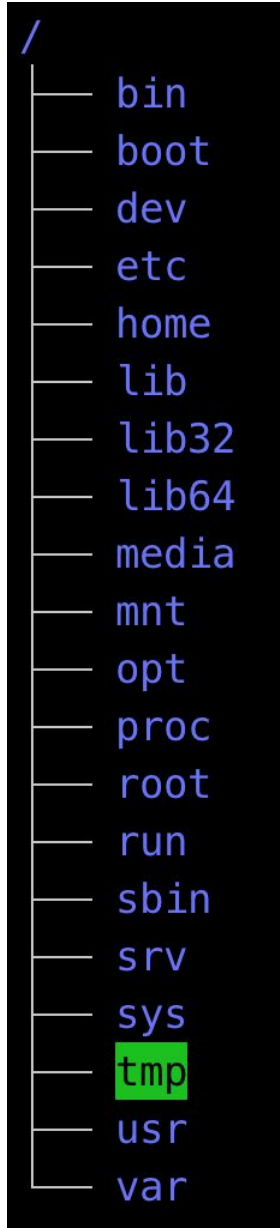
19. März	Kennenlernen, Motivation, Organisation
26. März	Historie, Rechnerarchitektur, LIDA-Labor
2. April	Einführung in das Betriebssystem Unix/Linux
11. April	Dateisystem und Rechteverwaltung
18. April	Linux in virtueller Maschine installieren
25. April	Prozessverwaltung und Benutzerverwaltung
2. Mai	vi Text Editor
9. Mai	Shell Scripting
16. Mai	Suchen und Finden
23. Mai	???
30. Mai	Wiederholung und PVL-Vorbereitung
6. Juni	Voraussichtlicher PVL Termin
13. Juni	Pfingstferien
20. Juni	Zusammenfassung & Feedback
27. Juni	Bonus: Wunschthema, Docker, AWS, ...

Hochschule
für Technik
Stuttgart

Wiederholung



Filesystem



- Hierarchisches Dateisystem (**fs**)
- Baum-Struktur
- / (Slash) als Trenner

Übungen

- Lege einen Ordner **/home/<user>/exercises-02** in deinem Home an und bearbeite die Aufgaben dort. Verwende beim anlegen des Ordner die **\$USER** Variable!
- Erstelle mit **echo** oder **cat** die Datei **tabelle.txt** mit folgendem Inhalt:

```
first,last,age,sex  
Jane,Tarzan,23,F  
John,Doe,25,M  
Bob,Stevens,65,M
```

Zeige nun die Datei in der Shell an
- Personalisiere die **Bash Prompt** (**export PS1="..."**)
- Erkunden sie das FS mit **cd**. Haben sie überall Zugriff?
- Liste alle Verzeichnisse (keine Dateien!) die ***backup*** enthalten im übergeordneten Verzeichnis, auch versteckte.
- Kopieren/Verschieben sie mit **relativen und absoluten Pfaden**

echo & cat

Variablen, Strings und co.

```
$ echo <Variable>
```

```
$ echo "muh kuh mäh"
```

Ohne Zeilenumbruch am Ende

```
$ echo -n <Variable>
```

Eigene Zeilenumbrüche

```
$ echo -e "lin1\nlin2"
```

Dateien und co.

```
$ cat <Datei>
```

Here doc | für z.B. **multiline** Inputs

```
$ cat <<EOF > file.txt
```

```
line1
```

```
line2 line2
```

```
EOF
```

!Wilder Hack!

Variable als Datei

```
$ cat <(echo $USER)
```

Umgebungsvariablen

- Alle anzeigen

```
$ env
```

- Ausgabe, z.B. von Variablen

```
$ echo <Variable>
```

- Inhalt von **\$USER** ausgeben

```
$ echo $USER
```

- Variable (temporär) setzen

```
$ export HALLO="1337"
```


Übungen

- Lege einen Ordner **/home/<user>/exercises-02** in deinem Home an und bearbeite die Aufgaben dort. Verwende beim anlegen des Ordner die **\$USER** Variable!
- Erstelle mit **echo** oder **cat** die Datei **tabelle.txt** mit folgendem Inhalt:

```
first,last,age,sex  
Jane,Tarzan,23,F  
John,Doe,25,M  
Bob,Stevens,65,M
```

Zeige nun die Datei in der Shell an
- Personalisiere die **Bash Prompt** (**export PS1="..."**)
- Erkunden sie das FS mit **cd**. Haben sie überall Zugriff?
- Liste alle Verzeichnisse (keine Dateien!) die ***backup*** enthalten im übergeordneten Verzeichnis, auch versteckte.
- Kopieren/Verschieben sie mit **relativen und absoluten Pfaden**

Wieso, weshalb, warum...

- Wieso sind sie hier? Warum Informatik? Wieso Stuttgart?
- Was haben sie (momentan) danach vor?

Unix/Linux Dateisystem und Rechteverwaltung

Hochschule
für Technik
Stuttgart

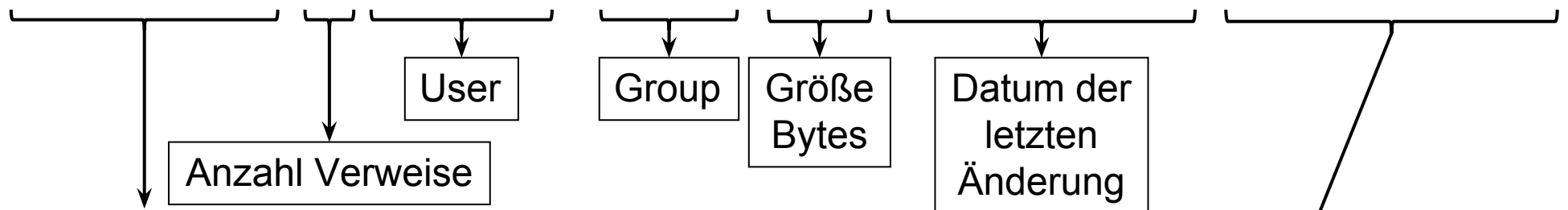
UNIX Dateisystemrechte (File permissions)

- Lassen wir uns den Verzeichnisisinhalt „long“ ausgeben:

```
$ ls -al
```

```
total 81657
```

```
drwxr-xr-x 21 rkeller staff 37 Mär 28 14:16 .
dr-xr-xr-x 20 root root 20 Mär 21 16:02 ..
drwx----- 3 rkeller staff 3 Mär 16 18:28 .adobe
-rw----- 1 rkeller staff 1274 Mär 28 14:51 .bash_history
-r--r--r-- 1 rkeller staff 373 Mär 12 12:44 .bashrc
```



1. Typ (d=Directory, -=File, l=softlink)
2. Rechte von „Owner“ (r=read, w=write, x=execute)
3. Rechte von „Group members“ (hier `staff`)
4. Rechte von „Other“ (auch Welt genannt)

Name der Datei oder des Unterverzeichnisses, Namen beginnend mit „.“ werden nicht mit `ls` angezeigt.

```
$ chmod 600 ~/.ssh
```

?

?

?

r w x

Rechteverwaltung

5	4	3	2	1
8	4	2	1	0

Rechteverwaltung

4

2

1

r

w

x

=

1

Rechteverwaltung

5	4	3	2	1
8	4	2	1	0

Rechteverwaltung

4

2

1

r

w

x

=

5

Rechteverwaltung

4

2

1

r

w

x

=

7

Rechteverwaltung

OWNER

421

rwX

= 7

GROUP

421

rwX

5

WORLD

421

rwX

1

Hand-on!



UNIX Dateisystemrechte (File permissions)

- Nun probieren Sie:

```
$ rm -fr /tank/home/rkeller/↵      (für remove)
```

- Was erhalten Sie für eine Fehlermeldung?

- Das sollte nicht möglich sein!

- Warum nicht???

- Setzen der Permissions (hier in der kurzen Fassung):

```
$ chmod 600 ~/.bashrc↵ (setzt Read/Write Owner)
```

- Vergleichen Sie die Rechte von ~/.bashrc zu vorher:

```
$ ls -al ~/.bashrc↵      (Ausgabe im Long Format)
```

UNIX Dateisystemrechte (File permissions)

- Schauen wir uns das Setzen der Rechte genauer an:

```
$ chmod 600 ~/.bashrc ↵      (setzt Read/Write Owner)
```

- Jede *Oktal*zahl steht für bestimmte Benutzer/Rechte:

- 400: der Besitzer (hier: `rkeller`) darf die Datei lesen
- 200: der Besitzer (hier: `rkeller`) darf die Datei schreiben
- 100: der Besitzer (hier: `rkeller`) darf die Datei ausführen
- 040: der Gruppe (hier: `staff`) darf die Datei lesen
- 020: der Gruppe (hier: `staff`) darf die Datei schreiben
- 010: der Gruppe (hier: `staff`) darf die Datei ausführen
- 004: alle anderen Benutzer („World“) dürfen die Datei lesen
- 002: alle anderen Benutzer („World“) dürfen die Datei schreiben
- 001: alle anderen Benutzer („World“) dürfen die Datei ausführen

UNIX Dateisystemrechte (File permissions)

- Schauen wir uns das Setzen der Rechte genauer an:

```
$ chmod 600 ~/.bashrc↵ (setzt Read/Write Owner)
```

- Für die obige Befehlszeile heißt das:
 - Die 1. Zahl (6) gibt an, daß Owner Lese-& Schreib-Rechte hat,
 - Die 2. Zahl (0) geben an, daß die Gruppe „staff“ und
 - Die 3. Zahl (0) alle anderen (Welt) keine Read-/Write oder Execute-Rechte haben!
- Was würde der Wert 005 bedeuten?
- Um das Home nicht zugreifbar für Welt zu machen:

```
$ chmod 700 ~↵ (zurücksetzen von HOME)
```

UNIX Dateisystemrechte (File Permissions)

- Für Dateien ist klar, was die Rechte bedeuten
- Für Verzeichnisse (engl. Directories) erlauben Rechte:
 - Lese: Der Verzeichnisinhalt darf angezeigt werden (`ls`)
 - Schreib: Es dürfen Dateien/Verzeichnisse erzeugt werden
 - Ausführbar: in das Verzeichnis darf gewechselt werden (`cd`)
- **Achtung:** Es macht also keinen Sinn *für Owner* Lese- oder Schreibrechte für ein Verzeichnis zu verbieten!
(Für die Gruppe oder Andere (World) aber sehr wohl!)
- **Test:** Setzen sie Ihr Home auf nicht-schreib- und ausführbar für World; erlauben aber das Lesen!
Probieren Sie gegenseitig, ob sie eine (lesbare) Datei kopieren können!!?

UNIX Dateiinformationen

- Für Dateien/Ordner/... wird die Zeit gespeichert, wann:
 - Sie angelegt wurde (Created)
 - Sie das letzte mal gelesen wurde (Accessed)
 - Sie das letzte mal verändert wurde (Modified)
- Es ist ersichtlich, dass gelten muss:
$$t_{\text{Created}} \leq t_{\text{Modified}} \leq t_{\text{Accessed}}$$
- Weiterhin gibt es noch Access Control Lists (ACLs); diese behandeln wir aber in der Vorlesung nicht!

UNIX Verzeichnisse anlegen

- Sie legen neue Verzeichnis mit `mkdir` an:
`$ mkdir mein_verzeichnis` (Make Directory)
- Existiert das Verzeichnis bereits, gibt es einen Fehler.
- Mit dem Parameter `-p` (für Parent) können Sie einen ganzen Verzeichnisbaum anlegen... Damit verschwindet auch der Fehler bei einem bereits existierenden Verzeichnis:
`$ mkdir -p mein_verzeichnis/src` (Make Directory)

- Neben Dateien und Verzeichnissen gibt es noch:
 - **Hard links:** Diese verweisen auf andere Dateien innerhalb des Dateisystems (also einer gemounteten Partition).
Beim Zugriff auf den Hardlink verweist das Betriebssystem intern direkt auf die eigentliche Datei. Hard-links legt man an mit:

```
$ ln echte_datei my↵      (legt Hardlink my an)
```

Dieser Hardlink ist nicht von der eigentlichen Datei unterscheidbar, d.h. beim Öffnen wird direkt die echte Datei aufgemacht.

Der Nachteil von Hardlinks: Kann man nicht über Dateisystem-grenzen anlegen (bspw. falls /home ist NFS, /etc ist ext2)

UNIX weitere Dateitypen 2/3

- Den Nachteil von Hardlinks gleichen Softlinks aus:
 - **Softlinks:** Diese verweisen auf andere Dateien/Verzeichnisse
Beim Zugriff auf den Soft-Link lenkt das Betriebssystem auf die eigentliche Datei um. Softlinks legt man an mit:

```
$ ln -s echte_datei my↵    (legt Soft-Link my an)
```

Das ist gut, z.B. wenn man eine Datei in verschiedene Verzeichnisse einordnen kann: ~/Dokumente und ~/Arbeit.

Angezeigt mit List im Long Format `ls -l` wird das so:

```
$ ln -s tmp.txt my↵
```

```
$ ls -l↵
```

```
lrwxr-xr-x 1 hpcraink staff 7 18 Apr 16:14 my -> tmp.txt
```

Übungen mit Dateien und Verzeichnissen

Aus dem Vorlesungsteil, probieren Sie folgendes aus:

- Löschen Sie `/home/rkeller`! Funktioniert das?
- Setzen Sie Ihr `$HOME` auf Nicht-Schreibbar, Nicht-Lesbar für World, dafür aber nur Lesbar für Ihre Gruppe. Können Sie dann in das Verzeichnis Ihres Nachbarn schreiben, d.h. eine Datei dahin kopieren?
- Welche Dateien liegen unter dem Verzeichnis `/dev/`, bzw. `/devices`?
- Kreieren Sie ein Verzeichnis für die Softwareentwicklung mit Ihren Kommilitonen (Geben Sie dem Verzeichnis einen passenden Namen).
 - Ebenso soll das Projekt ein Unterverzeichnis `src/` und `bin/` enthalten.
 - Verändern Sie die Rechte so, daß Ihr linker und rechter Nachbar mitarbeiten können
 - Nun legen Sie in dem Projekt Ihres Nachbarn zwei Dateien an: `my_GUI.java` in `src/` und `my_exec` in `bin/`
 - (leere Dateien können Sie mit dem Befehl `touch Dateiname` anlegen)
 - Die Rechte von `my_exec` sollen sein: ausführbar für ALLE, aber nicht veränderbar für ALLE.

Hausaufgaben

Machen sie sich bis nächsten Do. mit folgenden Befehlen vertraut:

- grep
- sort
- uniq
- w
- who
- whoami
- head
- tail
- tee
- date