

# **Linux Grundlagen II**

## **Zugriffsrechte Dateisystem Prozesssystem**

### **Was ist zu tun?**

- ▶ Funktionsweise der Zugriffsrechte auf Dateien und Geräte
- ▶ Bedeutung von "Besitzer" und "Gruppe" einer Datei
- ▶ Dateisystem
- ▶ Prozesssystem
  - Welche Prozesse laufen zurzeit?
  - Prozesse stoppen / weiterlaufen lassen
  - Prozesse (definitiv) beenden

## Benutzer und Gruppen

- ▶ Gruppenzugehörigkeit der Benutzer
  - genau eine "primäre Gruppe"
    - verwaltet in `/etc/passwd`
  - beliebig viele weitere "erweiterte Gruppen"
    - verwaltet in `/etc/group`
  - Kommando `groups login` gibt Gruppenzugehörigkeiten aus:
    - erste Gruppe "primär"
    - folgende "erweiterte Gruppen"

## Benutzer und Gruppen

- ▶ Weitere Auskunft gibt das Kommando `id login` mit den Optionen:
  - `-g` gibt nur primäre GruppenID (GID) aus
  - `-G` gibt alle GruppenIDs aus
  - `-u` gibt nur UserID (UID) aus
  - `-n` gibt **Namen** statt Nummern aus
  - ohne Parameter `login` werden die Daten für den aktuellen Benutzer ausgegeben
- Kombination der Optionen:
  - statt `id -g -n login`
  - auch `id -gn login`

## Zugriffsrechte

- ▶ Das Langformat `ls -l` zeigt in der ersten Spalte die Zugriffsrechte:

```
-rwxrwxrwx ... benutzer gruppe ... datei
```



### Die Rechte im Einzelnen:

- Das Recht ist nicht gesetzt
- r Lesen
- w Schreiben
- x Datei: ausführen  
Verzeichnis: hinein wechseln

## Rechte vergeben

- ▶ Rechte verwalten:

`chmod MODE datei`

- **MODE** gibt die zu ändernden Rechte an:
  - ein oder mehrere Zeichen aus `ugo`
  - + (hinzufügen), - (wegnehmen) oder = (setzen)
  - ein oder mehrere Zeichen aus `rxwugo` bzw. **Xst**
- **MODE** kann alternativ auch in anderer Notation angegeben werden (später mehr dazu...)
- Beispiel:
  - `chmod a+r,u+wx,g+r datei`

## SUID/SGID-Bits

### ▶ SUID-Bit: **u+s**

- das Programm wird nicht mehr mit den Rechten des Benutzers, sondern denen des Besitzers der Datei ausgeführt.

### ▶ SGID-Bit: **g+s**

- Programmdateien
  - das Programm wird mit den Rechten der Gruppe der Datei ausgeführt
- Verzeichnisse
  - Alle unterhalb abgelegten Dateien erhalten die Gruppenzugehörigkeit wie das Verzeichnis selbst

## Sticky-Bit

### ▶ Sticky-Bit: **+t**

- wer Schreibrechte in einem Verzeichnis hat, darf alle dort liegenden Dateien löschen, auch wenn auf diesen Dateien selbst kein Schreibrecht besteht
- Abhilfe: Sticky-Bit.  
Zum Löschen einer Datei wird Schreibrecht auf der Datei selbst benötigt.

## Oktale Darstellung

- ▶ Rechtestruktur lässt sich einfacher und kürzer darstellen:
- ▶ Die einzelnen Rechte erhalten unterschiedliche Wertigkeit:

-rwxrwxrwx	u+s	4
0421421421	g+s	2
# + + +	+t	1

- ▶ Beispiel:

- -rwxrwxr-t wird zu 1775

## Besitzer und Gruppe

- ▶ Wechseln des Besitzers (nur root)
  - **chown** (change owner)
  - **chown** *neuerbesitzer datei*
- ▶ Wechseln der Gruppe
  - **chgrp** (change grp)
  - **chgrp** *neuegruppe datei*
- ▶ Beides gleichzeitig (nur root)
  - **chown** *besitzer.gruppe datei*

## Dateien finden

### ▶ locate

- Sucht in einer Datenbank, folglich sehr schnell aber nicht immer aktuell

### ▶ find

- Syntax: `find pfad optionen`
- sehr viele Suchmöglichkeiten
  - nach Besitzer, Änderungszeit, Rechtestruktur, Größe, Anzahl Links, ...
  - auch im Vergleich zu anderen Dateien
  - Definition von Operationen auf Ergebnis

## Jobs

- ▶ Ein laufendes Programm kann mit `<Ctrl>+<z>` eingefroren werden
- ▶ Anzeigen der laufenden Kommandos
  - `jobs`
- ▶ Weiterlaufen lassen mit
  - `fg` (foreground) im Vordergrund
  - `bg` (background) im Hintergrund
- ▶ Direkt im Hintergrund starten:
  - `kommando &`

# Das Prozesssystem

## ▶ Laufende Prozesse anzeigen

- `ps`
- `pstree`
- `top`

## ▶ Prozessen Signale senden

- `kill signal PID`

Signale:

- `-1` Konfiguration neu einlesen
- `-9` abschießen
- `-15` ordnungsgemäß beenden