Unix Grundlagen
(Teil 2)

Jörn Stuphorn
stuphorn@rvs.uni-bielefeld.de

Rechte

- Dateien haben
  - Besitzer
  - Gruppenzugehörigkeit
  - Zugriffrechte
- UID (Besitzer) in passwd festgelegt
- GID (Gruppe) in group festgelegt
  - SUID Bit (set user id): Wird Programm aus Datei gestartet, läuft es mit Rechten des Dateibesitzers
  - SGID Bit (set group id): Wird Programm gestartet, erhält Aufrufer die GID der Gruppe des Programms

chgrp – Change Group

- ändert die Gruppe der Datei
  - Aufruf: `chgrp Group[Opt][Datei]`
- Achtung: Es kann nur eine Gruppe gewählt werden, in der man selber Mitglied ist.
- Ändere den Besitzer von /home/knoppix/backup/hosts nach saned

chown – Change Owner

- ändert den Besitzer der Datei
  - Aufruf: `chown User[:Group][Opt][Datei]`

- Achtung: Nur root darf Dateibesitzer ändern
- über 'su user' beginnt man eine Sitzung als anderer User, ohne *user* wird eine Sitzung von root begonnen.

- Starte root-Sitzung
- Ändere den Besitzer von /home/knoppix/backup/hosts nach root und wieder nach knoppix
- Beende root-Sitzung mit *exit*

chmod – Change Mode

- ändert den Modus der Datei
  - Aufruf: `chmod[Rechte][Datei]`
- Optionen:
  - u+[Wahl] ändert die Besitzerrechte
  - g+[Wahl] ändert die Gruppenrechte
  - o+[Wahl] ändert die Weltrechte
- Zahlen:
  - 4 stellig:
    - 1. Sonderbits (SUID, SGID)
    - 2. User
    - 3. Group
    - 4. World

chmod – Change Mode

- Zahlen:
  - 4 stellig:
    - 1. Sonderbits (SUID, SGID)
    - 2. User (Zahl * 100)
    - 3. Group (Zahl * 10)
    - 4. World (Zahl * 1)
  - SUID: 4000
  - SGID: 2000
  - read: 4
  - write: 2
  - execute: 1
- Setze -rw---x-w- mit SUID-Bit
  - User: rwx (4*100 + 2*100 + 1*100)=700
  - Group: x (1*10) = 10
  - World: w (2*1) = 2
  - SUID: 4000
  - **Summe: 4712 => chmod 4712 *[Datei]***

Suchmuster

- Auswahl von mehreren Dateien
  - text Auswahl einer Zeichenkette
  - * Beliebig lange (auch leere) Zeichenkette

    Bsp.: `ls -l *`

    Auflistung von Dateien (mit beliebigem Titel)
  - ? Ein beliebiges Zeichen

    Bsp.: `ls -l messages.?`

    Auflistung von Dateien die mit messages. anfangen und genau ein Zeichen am Ende haben
  - [] Reihe/Bereich von Zeichen

    Bsp.: `ls -l [aA]*`

    Auflistung von Dateien mit einem kleinen oder großen A am Anfang
  - [!..]negierte Reihe/Bereich von Zeichen

    Bsp.: `ls -l [!a-m]*`

    Auflistung von Dateien die nicht mit einem kleinen a...m beginnen

Sperren von Sonderzeichen

- *Sonderzeichen:* `* ? [ ] ! . ~ $`
- *Problem:*
  - *Dateien test1, test2, test3 ... existieren*
  - *touch* `test*`
  - `test*` *soll gelöscht werden*
  - `rm test*` *würde aber auch test1 ... löschen*

- *Sperren von Sonderzeichen:*
  - \ *backslash*
    - sperren einzelner Sonderzeichen
  - *' ' Hochkommata*
    - Sperren aller Sonderzeichen
  - *" " Anführungszeichen*
    - Sperren einiger Sonderzeichen (z.B. bleibt *$* ungesperrt)

Standardkanäle & Pipes

- Viele Dienst- und Filterprogramme nutzen unter Unix Standardkanäle
  - stdin Standardeingabekanal
  - stdout Standardausgabekanal
  - stderr Standardfehlerkanal

- Umleitung dieser Kanäle
  - Ausgabeumlenkung
    - *Kommando* > [Kanal] Ausgabe wird nach Kanal umgeleitet
    - *Kommando* >> [Kanal] Ausgabe wird an Kanal angefügt
  - Eingabeumlenkung
    - *Kommando* < [Kanal] Eingabe von Kommando aus Kanal
- Pipeline / Pipelining
  - Verknüpfung des Standardausgabekanals eines Programms mit dem Standardeingabekanal eines anderen Programms
    - who | sort
    - tar cpshvvfzb - 64 /daten | ssh root@srv dd of=/dev/st0 obs=64b

Standardkanäle & Pipes

- Was bewirken die folgenden Umleitungen?
  - Pipelines
    - who | sort
    - tar cpshvvfzb - 64 /daten | ssh root@srv dd of=/dev/st0 obs=64b
    - cat xyz | cat | more
  - Umleitungen
    - cat xyz > xyz
    - cat xyz >> xyz
    - echo asdf > asdf
    - echo asdf >> asdf
    - cat < etc/hosts

Dienst- und Filterprogramme

- *Linux / Unix bietet die Möglichkeit, Filterprogramme und Pipes zu verbinden*

- Nicht jedes Programm benötigt jede Funktion

- Suchen von Zeilen (z.B. grep)
- Zeilenweises Sortieren (z.B. sort)
- Umsetzen von Zeichenketten (z.B. tr)
- Betrachten von Text (z.B. pg)
- Dateien vergleichen (z.B. diff, cmp)

Zeilenweises Sortieren

- sort [+Pos1 [-Pos2]] *Eingabe*
  - sortiert Eingabe und liefert Ergebnis auf Ausgabe zurück
  - sort sortiert als Voreinstellung die ganze Eingabezeile
  - sort unterteilt die Eingabezeile in Felder 0 ... n
    - Felder durch ein Leerzeichen getrennt
    - mehrere Leerzeichen:
      - 1. Leerzeichen Trenner
      - Rest: Inhalt des nachfolgenden Feldes
  - sort +1 *-2 Eingabe*
    - Eingabe nach Information von Anfang von Feld-Nr.1 bis Ende Feld-Nr.2 (effektiv also nach dem Inhalt der 2. Spalte)
- Beispiele:
  - ls -l | sort
  - ls | sort
  - ps -ef | sort +4 -5
- Was machen diese Befehle?
- Was hätte man erwartet?

Suche in Zeilen

- Ausgabe nur von Zeilen, in denen gesuchte Information auftaucht

- grep
  Aufruf: `grep [Optionen] Suchmuster [Datei]`
  - Filtern von Eingabe nach Zeilen, die Suchmuster enthalten
- egrep *(enhanced grep)*
  Aufruf: `grep [Optionen] Suchmuster [Datei]`
  - Wie grep, durch Option -f kann aber auch Suchmuster aus Datei eingelesen werden.
- fgrep *(fast grep)*
  Aufruf: `grep [Optionen] Suchmuster [Datei]`
  - Wie egrep, allerdings werden Sonderzeichen, die Suche flexibler gestalten, nicht unterstützt.
  - Sehr schnell

Suchausdrücke in grep

- Suchmuster in grep über reguläre Ausdrücke realisiert

- Sonderzeichen in regulären Ausdrücken:

Sonderzeichen	Bedeutung
[...]	Menge von Zeichen, die an dieser Stelle erlaubt sind
[^...]	Negierte Menge von Zeichen. Alle übrigen Zeichen sind an dieser Stelle erlaubt
.	Ein einzelnes, beliebiges Zeichen
*	Wiederholungszeichen
^	Symbol für Zeilenanfang innerhalb eines Suchmusters
$	Symbol für Zeilenende innerhalb eines Suchmusters
\{...\}	Wiederholungsangabe a) \{n\}: Genau n Wiederholungen b) \{n,\}: Minimal n Wiederholungen c) \{x,y\}: Zwischen x und y Wiederholungen

Suche in Zeilen

- Was bewirken die folgenden Umleitungen?
  - cat /etc/hosts | grep localhost
  - cat /etc/hosts | grep [Ii][Pp]
  - cat /etc/hosts | grep lo

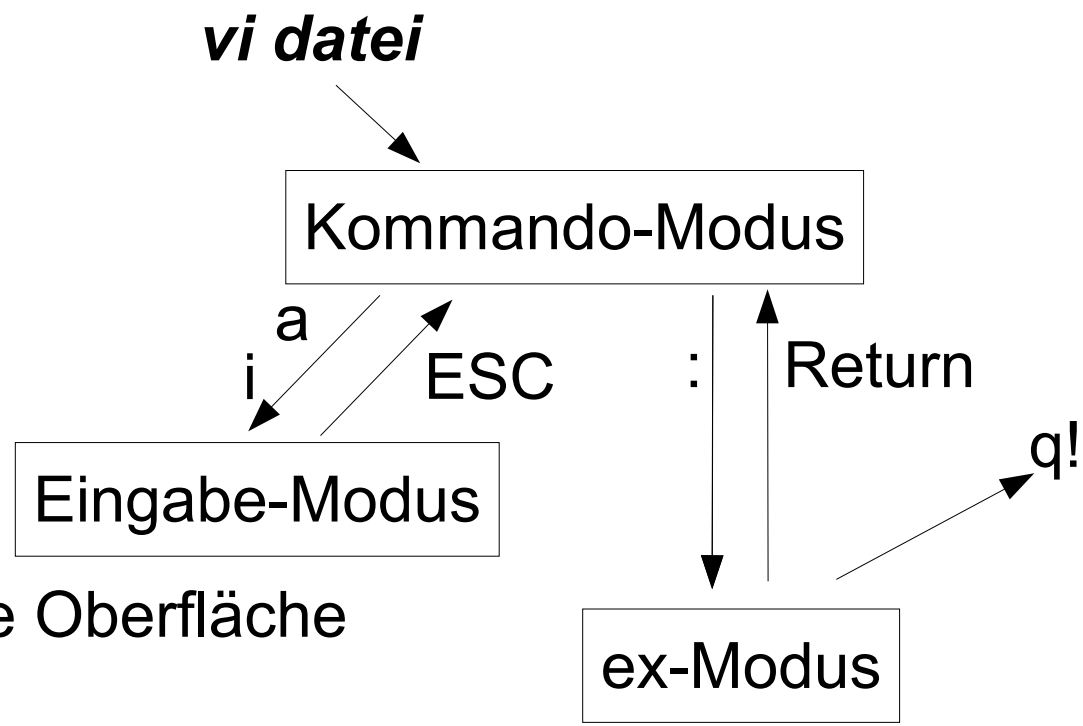
Editieren mit vi

Aufruf: `vi [Datei]`

Funktion: Anzeigen und ändern einer Textdatei
Ist Datei nicht vorhanden wird neue Datei angelegt

vi arbeitet mit 2 Betriebsmodi:

- Kommando-Modus
- Eingabe-Modus



Vorteil von ***vi:***

- Ist normalerweise auf jedem Unix/Linux vorhanden
- benötigt keine graphische Oberfläche

Text ersetzen in vi

- :1,$s/[Mm]ein/Dein/g
  - ersetzt in der Datei beginnend von der 1. Zeile bis zur letzen Zeile ($) alle Vorkommen von mein/Mein mit 'dein'.
- Beispiel:
  - Mein Klavier ist mein Piano.
  - Dein Klavier ist Dein Piano.

- :1,$s/\(Herren\) \(und\) \(Damen\)/\3 \2 \1/g
  - Ersetzt 'Herren und Damen' gegen 3. Wort, 2. Wort, 1. Wort, also 'Damen und Herren'
- Beispiel:
  - Meine Herren und Damen, ...
  - Meine Damen und Herren, ...

Zeichenketten umsetzen

- tr [Optionen] VonZeichen NachZeichen

- Setzt Zeichen oder Zeichenmengen der Eingabe um
- Liest Daten aus stdin und gibt umgesetztes Ergebnis auf stdout zurück

- Beispiele:
  - cat /etc/hosts | tr „o“ „Y“
  - cat text | tr „A-Z“ „a-z“
- Was bewirken diese Beispiele?

- Optionen:
  - -c Komplement von „VonZeichen“ verwenden
  - -d „VonZeichen“ werden aus Eingabe gelöscht
  - -s Mehrere gleiche Zeichen aus „NachZeichen“ werden zusammengefasst

Betrachten von Text

- more, less
  - Bereits betrachtet

- pg
  - muss laut X/OPEN Portability Guide auf UNIX Systemen vorhanden sein.
  - Ermöglicht seitenweise Betrachten von Daten

  - Aufruf: pg [Optionen] *Eingabe*

  - *Optionen:*
    - *-Zahl wieviele neue Zeilen sollen auf neuem Bildschirm dargestellt werden?*
    - *-p Text Ersetzt den ':'-Prompt durch Text*
    - *+Zahl Daten werden erst ab der „Zahlten“-Zeile wiedergegeben*

Dateien vergleichen

- Oft sollen der Inhalt von 2 Dateien miteinander verglichen werden
  - diff [Opt.] Datei1 Datei2
    - Gibt die Änderungen an, die an Datei1 gemacht werden müssten, um sie mit Datei2 identisch zu machen

```
joern@whistler:~$ diff /etc/hosts hosts
5,7d4
< # The following lines are desirable for IPv6 capable hosts
< # (added automatically by netbase upgrade)
<
9c6
< fe00::0 ip6-localnet
---
> fy00::0 ip6-localnet
```

    - *n* **a** *x,y* Zeilen x-y sind in Datei2 nach Zeile n hinzugekommen
    - *n,m* **d** *x* Zeilen n-m sind in Datei2 nach Zeile n gelöscht worden
    - *n,m* **c** *x,y* Zeilen n-m sind in Datei2 durch Zeilen x-y ersetzt worden
  - cmp [Opt.] Datei1 Datei2
    - Byteweises Vergleichen zweier Dateien
      Gibt Anzahl der unterschiedlichen Zeichen und Zeilen an
    - cmp -l listet auf, an welcher Position Unterschiede bestehen

Prozesssteuerung

- Prozesse starten
- Aufruf einer ausführbaren Datei
  - ./[Dateiname]
  - ./[Dateiname] &
- Starten einer Datei in Shell
  - sh [Dateiname]
  - sh [Dateiname] &

- Prozesse, die mittels & im Hintergrund laufen, können keine Benutzereingaben verarbeiten!

- Systemlast anzeigen: top
- Prozesse anzeigen: ps
- Prozesse beenden:
  - kill
  - killall

top

Aufruf: `top [Optionen] [PID]`

Funktion: Liefert eine Tabelle der Prozesse, die aktuell am stärksten das System beanspruchen.
Verschiedene Sortierungen sind möglich.

Ähnliche Programme für graphische Oberflächen:
`ktop, gtop`

ps

- Erzeugt eine Liste von Prozessen (typischerweise der aktuellen Shell)
  - ps f: Prozesshierarchie anzeigen

```
joern@localhost:~$ ps f
  PID TTY      STAT   TIME COMMAND
 6678 pts/3    Ss     0:00 /bin/bash
 6761 pts/3    R+     0:00  \_ ps f
 6388 pts/1    Ss     0:00 /bin/bash
 6391 pts/1    S+     0:00  \_ mc
 6393 pts/2    Ss     0:00      \_ bash -rcfile .bashrc
 6424 pts/2    S+     3:24          \_ /usr/bin/../lib/Adobe/
joern@whistler:~$
```

  - ps -ef: Anzeige aller Prozesse

```
joern@localhost:~$ ps -ef
UID        PID  PPID  C STIME TTY          TIME CMD
root         1     0  0 08:27 ?        00:00:00 init [2]
root         2     1  0 08:27 ?        00:00:00 [ksoftirqd/0]
root         3     1  0 08:27 ?        00:00:00 [events/0]
root         4     3  0 08:27 ?        00:00:00 [khelper]
root        16     3  0 08:27 ?        00:00:03 [kacpid]
root       138     3  0 08:27 ?        00:00:00 [kblockd/0]
root       170     3  0 08:27 ?        00:00:03 [pdflush]
```

kill / killall

Aufruf: `kill [Optionen] [PID]`

Funktion: Sendet ein Signal zu einem Prozess, meistens um ihn zu beenden.

Aufruf: `killall [Optionen] [Name]`

Funktion: Ähnlich wie `kill`,
aber alle Prozesse mit dem angegebenen Namen werden beendet.

Systemstart

- System wird über Shellskripte gestartet
  - /etc/init.d: Speicherort der Skripte
  - /etc/rcN.d: Verknüpfungen zu Skripten, die zu einem „runlevel“ gehören.
- Runlevel:
  - single-user mode
    Debian: Level 1
    SuSE: Level 1
  - multi-user mode
    Debian: Level 2
    SuSE: Level 5
- Startskripte
  - Beim booten: rcS.d, rc0.d, rc2.d
- Stopskripte
  - Beim herunterfahren: rc2.d, rc0.d