Dateisystem 2, RegEx, Manpages

Linux-Kurs der Unix-AG

Andreas Teuchert

10. Dezember 2012

Manpages

- Dokumentation zu Programmen, speziellen Dateien, etc. steht in Manpages
- Aufruf mit `man <programm>`
- Manpage wird mit dem Standard-Pager (meistens `less` angezeigt)
- Manpages sind Dateien im roff-Format, die unter `/usr/share/man` liegen

Gliederung

- Manpages sind in verschiedene Abschnitte gegliedert
- wichtige Abschnitte:
  - NAME: Programmname und Kurzbeschreibung
  - SYNOPSIS: Aufruf (Optionen und Argumente)
  - DESCRIPTION: Beschreibung des Programms
  - OPTIONS: Beschreibung der Optionen (manchmal unter DESCRIPTION)
  - ARGUMENTS: Bechreibung der Argumente (auch manchmal unter DESCRIPTION)
  - AUTHOR/COPYRIGHT: Angaben zum Autor/den Autoren
  - SEE ALSO: Weiterführende Verweise
- Gliederung nicht fest vorgegeben, Autor kann davon abweichen

Kapitel

- Manpages sind nach Kapiteln geordnet
- 10 Kapitel existieren
- wichtige Kapitel:
  - 1: Benutzerkommandos (z. B. `cat`, `less`)
  - 4: Gerätedateien (z. B. `/dev/null`)
  - 5: Konfigurationsdateien (z. B. `/etc/passwd`)
  - 7: Verschiedenes (z. B. Lizenzen, Internet-Protokolle)
  - 8: Kommandos für die Systemverwaltung (z. B. `useradd`)
- manchmal Manpages mit gleichem Namen in verschiedenen Kapiteln: explizite Auswahl mit `man <kapitel> <manpage>` (Beispiel: `passwd`)

apropos

- mit `apropos <stichwort>` oder `man -k <stichwort>` kann man nach Stichwörtern in Manpages suchen
- durchsucht den NAME-Abschnitt der Manpages

Weitere Dokumentationsquellen

- manche Programme (v. a. GNU) sind zusätzlich zu den Manpages auch über Info-Seiten dokumentiert:
  `info <programm>`
- unter `/usr/share/doc` liegt die Dokumentation zu den installierten Paketen

Merkhilfe für Optionen

- viele Befehle unterstützen einen gemeinsamen Satz an Optionen
- `-r` („recursive“): `rm`, `cp`
- `-i` („interactive“): `rm`, `mv`, `cp`, `ln`
- `-b` („backup“): `mv`, `cp`, `ln`
- `-v` („verbose“): `rm`, `mv`, `cp`, `ln`
- Faustregeln:
  - wenn man eine Option braucht, existiert sie wahrscheinlich
  - Optionen haben sinnvolle Namen
  - gleiche Namen haben (wenn es passt) auch die gleiche Bedeutung

ln

- `ln` („link“) legt Verknüpfungen an
- ohne Optionen wird ein zweiter Name für die gleiche Datei erzeugt (Hardlink); geht nicht mit Verzeichnissen (sonst Zyklen im Dateibaum und andere schlimme Dinge)
- `ls -l` zeigt in der zweiten Spalte den Link-Zähler an
  - zeigt bei Dateien die Anzahl der Links an
  - bei Verzeichnissen mindestens 2 wegen `.` (Link auf das Verzeichnis selbst) und für jedes Unterverzeichnis +1 wegen `..` (Link auf das Elternverzeichnis)
- `ls -i` zeigt die sog. Inode-Nummer an (laufende Nummer für Dateien); bei Links ist diese gleich
- wird ein Link gelöscht, kann man über die anderen noch auf die Datei zugreifen

ln -s

- Hardlinks sind verwirrend (verschiedene Dateinamen, aber gleicher Inhalt)
- `ln -s` legt einen Softlink an: Link zeigt auf das Ziel
- geht auch mit Verzeichnissen
- wird das Ziel gelöscht, zeigt der Link ins Nichts
- `ls -l` zeigt an, wo der Link hinzeigt

find

- `find` findet Dateien im Dateisystem
- erstes Argument ist das Verzeichnis, in dem gesucht werden soll
- die weiteren Argumente sind Suchkriterien:
- `-name`: Muster für den Dateinamen (Wildcards erlaubt, dann aber in Anführungszeichen setzen)
- `-type`: Dateityp (`f` für normale Dateien, `d` für Verzeichnisse)
- `-o` zwischen zwei Kriterien verknüpft diese mit Oder (Standard ist Und)
- `!` vor einem Kriterium kehrt dessen Bedeutung um
- weitere Parameter: Buch, Kapitel 6.4.4

find -exec

- standardmäßig gibt `find` die gefundenen Dateien nur aus
- mit `find ... -exec <befehl> "{}" \;` kann auch für jede gefundene Datei ein Befehl ausgeführt werden
- Anwendungsbeispiel: alle Dateien löschen, die älter als X Tage sind

locate

- `find` durchsucht den Dateibaum bei jedem Aufruf neu, kann sehr lange dauern
- `locate` hat eine Datenbank und sucht darin, geht sehr viel schneller
- Datenbank muss regelmäßig aktualisiert werden
- Dateien, die nicht in der Datenbank stehen, werden nicht gefunden
- Argument: Dateimuster (Wildcards erlaubt)

grep

- `grep` gibt alle Zeilen einer Datei aus, die ein Suchmuster enthalten
- Wichtige Optionen:
  - `-i`: bei der Suche nicht auf Klein- und Großschreibung achten
  - `-v`: nur die Zeilen ausgeben, die das Muster **nicht** enthalten
  - `-r`: Verzeichnisse rekursiv durchsuchen
  - `-E`: erweiterte Reguläre Ausdrücke können verwendet werden (siehe nächste Folie)
- Argumente: Suchmuster und Dateien/Verzeichnisse

Reguläre Ausdrücke

- einfache Suchmuster erlauben es nur nach Zeichenketten zu suchen
- aber schon bei mehreren möglichen Schreibweisen wird es kompliziert (z. B. Potential und Potenzial)
- Lösung: reguläre Ausdrücke (Regex)
- Muster können Platzhalter und Wiederholungsangaben enthalten
- sinnvoll mit `grep -E` (oder `egrep`)

Regex – Platzhalter und Wiederholungsangaben

Platzhalter

- `.` steht für ein einzelnes Zeichen: `a.b` passt auf `acb`, `axb`, ...
- `[...]` steht für eine Klasse von Zeichen (z. B. `[tz]` für `t` oder `z`, `[a-z]` für alle Zeichen von `a` bis `z`)

Wiederholungsangaben

- `?`: Zeichen kommt einmal oder keinmal vor: `ab?c` passt auf `abc` und `ac`
- `*`: Zeichen kommt beliebig oft vor: `ab*c` passt auf `ac`, `abc`, `abbbbbc`, ...
- `+`: Zeichen kommt mindestens einmal vor: `ab+c` passt auf `abc`, `abbbbbc`, aber nicht `ac`

Spickzettel

Alle Befehle

Befehl	Optionen
`man`	`-k`
`apropos`	
`info`	
`ln`	`-s`
`find`	`-name`, `-type`, `-size`, `-user`, `-exec`, `(!)`
`locate`	
`e?grep`	`-i`, `-v`, `-r`, `-E`

Regexes

Platzhalter: `.`, `[...]`, Wiederholungsangaben: `?`, `*`, `+`