EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Dr. Eberle Zentrum für digitale Kompetenzen

DataBASHing - Dateidetails

Martin Raden

Einführung von

- `chmod, chown, chgrp, ls -l` in Dateirechte
- `tar, gzip, zip, zcat, zless, zgrep` in Kompression

Video: Bashinho - #03 Befehle für das Rechtemanagement (chmod, chown, chgrp) [6 min]

Video: (en) Pedagogy - archive files using tar command in linux | create tar & extract from tar file | tar command in linux [8 m

Video: (en) Pedagogy - compress - decompress files using gzip & bzip2 in linux | tar command to manage compressed files

---

Dateirechte

- u.a. via "`ls -l`" **am Anfang der Zeile** angezeigt:
  z.B. `-rwxr-x--- 42 hans family 42 [DATUM] MeinSkript.sh`
  - führendes "`-`" : weder link "`l`" noch verzeichnis "`d`", daher Zeichen für normale Dateien
  - erster Dreierblock "`rwx`" : Rechte des Besitzers (`u` = user) (hier hans) = (`r`)ead + (`w`)rite + e(`x`)ecute
  - zweiter Dreierblock "`r-x`" : Gruppenrechte (`g` = group) (hier family = Gruppe von accounts), welche keine Schreibrechte "`w`" haben (daher an dessen Stelle ein "`-`")
  - dritter Dreierblock "`---`" : Rechte für den "Rest der Welt" (`o` = others), hier keinerlei Rechte eingeräumt (alles "`-`")
  - "`1`" (Zahl) = Anzahl Links für diese Datei (erstmal egal)
  - "`hans`" = Eigentümername = es gibt **nur EINEN Eigentümer**
  - "`family`" = Gruppenname, zu welcher die Datei zugeordnet ist = nur **EINE GRUPPE pro Datei** zuweisbar!
  - "`42`" = Dateigrösse in Bytes (oder zB `4K` wenn `ls -lh` verwendet)
- **ausführbare Dateien** (mit `x` für jeweilige Gruppe = hier z.B. für "`hans`" und andere der "`family`" Gruppe)
  - können direkt mit Pfadangabe ausgeführt werden, z.B. via `./MeinSkript.sh` anstatt `bash MeinSkript.sh`
  - werden direkt ausgeführt, wenn sie in einem Verzeichnis liegen, welches in `PATH`**-Variable** gelistet ist ("`:`"-getrennte Liste), dh. direkt via `MeinSkript.sh` **ohne Pfadangabe** aufrufbar
- **Versteckte Dateien** = "`.`"-prefixed
  - Dateien, die mit "`.`" beginnen, werden standardmässig bei `ls` Aufruf ausgeblendet und nicht angezeigt
  - mit "`ls -a`" werden diese gelistet

  - i.d.R. Konfigurationsdateien, wie z.B. `$HOME/.bashrc` welche u.a. folgendes definiert:
    * das Aussehen der bash (Farbe, prompt, ...)
    * wie manche Kommandos reagieren (Länge der History, ...)
    * definiert u.a. Befehlskurzformen (via `alias`)

Dateirechte ändern

- `chmod` - **Dateirechte ändern**
  - kompakte Zusammenfassung mit Beispielen
  - z.B. Datei **für alle ausführbar** machen via "`chmod a+rx MeinSkript.sh`" ("`r`" idR nicht nötig, da meist schon vorhanden)
  - wenn nur "**Änderung**" von Interesse (d.h. hinzufügen oder wegnehmen), dann bietet sich die folgende (regex-codierte) Notation an:
    * [ugoa][+-]r?w?x? im Detail
    * WER = (u)ser, (g)roup, (o)thers, (a)ll, d.h. alle drei
    * WIE = "+" hinzufügen (wenn noch nicht vorhanden) oder "-" wegnehmen
    * WAS = (r)ead, (w)rite, e(x)ecute rights
    * z.B. chmod o+r MeinSkript.sh würde allen Nutzern Leserechte einräumen (unabhängig davon, ob sie diese schon besitzen oder nicht)
  - für ein **explizites Setzen** aller Rechte auf einmal, ist ggf. die Zahlennotation (siehe obigen Link) besser

Eigentum/Gruppe ändern

- `chown` - **Eigentumsrechte/-zuordnung ändern**
  - der Eigentümer (oder z.T. ein Admin) kann die Eigentumsrechte übergeben/ändern
  - z.B. ein "`chown liesbeth MeinSkript.sh`" von `hans` würde die Datei von oben an `liesbeth` übertragen
  - alles andere bleibt ungeändert, d.h. Ort, Dateirechte, ...
  - kann auch zur Gruppenänderung eingesetzt werden
  - Details und Beispiele von howtoforge.de
- `chgrp` - **Gruppenzuordnung ändern**
  - Gruppenzugehörigkeit von Dateien ändern
  - z.B. "`chgrp parents MeinSkript.sh`" würde den Zugriff (bzw. die gesetzten Gruppen-Dateirechte) auf Nutzer der Gruppe `parents` einschränken. Alle anderen Nutzer (ausser dem Eigentümer) fallen dann unter die Dateirechtgruppe "`others`".
- `groups` - **Gruppen (eines Nutzers) anzeigen**
  - ein Nutzer kann mehreren Gruppen angehören
  - "`groups`" - liefert die eigenen Gruppen
  - "`groups hans`" - liefert die Gruppen des Nutzers hans

Kompression

Datenarchivierung = meist 2 Schritte in einem:

- **Datenaggregation** = mehrere Dateien/Ordner/... **in einer Datei zusammenfassen**
- **Datenkompression** = eine Datei (oder Datenstrom) verlustfrei **kleiner speichern**

tar

`tar` = Datenaggregation = Archiverstellung

- fasst mehrere Datein oder ganze Verzeichnisstrukturen zusammen
- Aufruf = `tar -[OPTIONS] -f ARCHIVDATEI [FILESorFOLDERS]`
- `ARCHIVDATEI` i.d.R. mit **.tar Endung** (wenn ohne Kompression verwendet)
  z.B. "`tar -cf bash-config-files.tar $HOME/.bash*`"
- Optionen steuern eigentliche **Handlungen** (können zusammengefasst werden):
  - "`-c`" = compress = `ARCHIVDATEI` wird angelegt und gefüllt
  - "`-x`" = eXtract = `ARCHIVDATEI` wird "ausgepackt" und Dateien wiederhergestellt
  - "`-r`" = add = weitere Dateien zu existierendem Archiv hinzufügen
  - "`-t`" = list = alle enthaltenen Dateien anzeigen
- Zusätzlich kann **Kompressionsverfahren** gewählt werden (wählt entsprechendes Konsolenprogramm)
  - "`-z`" = gzip (siehe unten)
  - "`-j`" = bzip2
  - "`-J`" = xz
  - falls gesetzt, sollte Archivname mit **entsprechender Endung verwendet** werden (nicht automatisch angehängt!),
    z.B. "`tar -czf bash-config-files.tar.gz $HOME/.bash*`"
  - muss **auch beim Entpacken, Auflisten, etc.** entsprechend **angegeben** werden,
    z.B. "`tar -lzf bash-config-files.tar.gz`"

gzip

`gzip` = Datenkompression = GNU Implementierung eines ZIP Kompressionsverfahrens

- ermöglicht das (de)komprimieren einzelner **Dateien**, z.B.
  - "`gzip test.txt`" erzeugt die Datei `test.txt.gz` mit **Endung .gz** und löscht `test.txt`
  - "`gunzip test.txt.gz`" dekomprimiert die Datei und benennt sie wieder um
- direkt in **pipes** verwendbar, z.B.
  - "`-c`" Ausgabe in STDOUT und Archiv erhalten, z.B. "`gunzip -c test.txt.gz | wc`"
  - "`-`" als "Dateiname" liest STDIN und gibt komprimierte Daten in STDOUT aus, z.B. "`ls | gzip - > ls-output.gz`"
- "`-k`" = keep = Originaldatei wird erhalten
- Varianten von bekannten Programmen, die direkt mit **gzip-ten Text-Dateien** umgehen können
  - `zcat` - z.B. "`zcat ls-output.gz`" = entspricht "`gunzip -c ls-output.gz`" aber ggf. intuitiver
  - `zless` - z.B. "`zless ls-output.gz`"
  - `zgrep` - direktes suchen in komprimierten Dateien

bzip2

Mehr oder minder analog zu `gzip` (inklusive "`bz...`" tools) aber verwendet anderem Kompressionsalgorithmus.

zip

`zip` = sowohl Datenaggregation als auch IMMER -kompression

- intern quasi eine Kombination von tar und einem anderen ZIP Verfahren (als gzip)
- Aufruf = `zip [OPTIONS] ARCHIVDATEI [FILESorFOLDERS]`
- Entpacken mit "`unzip ARCHIVDATEI`"

- `ARCHIVDATEI`
  - i.d.R. mit **Endung .zip**
  - wenn "-" dann lesen von STDIN und schreiben nach STDOUT zur Verwendung in pipes analog zu gzip
  - wenn schon vorhanden, werden die neuen Dateien direkt hinzugefügt
- "`-r`" = recursive = alle Dateien/Unterverzeichnisse der angegeben Verzeichnisse
- Passwortschutz möglich
- für weitere Details bitte **man page** mit vielen Beispielen konsultieren!

February 4, 2025