unix / linux

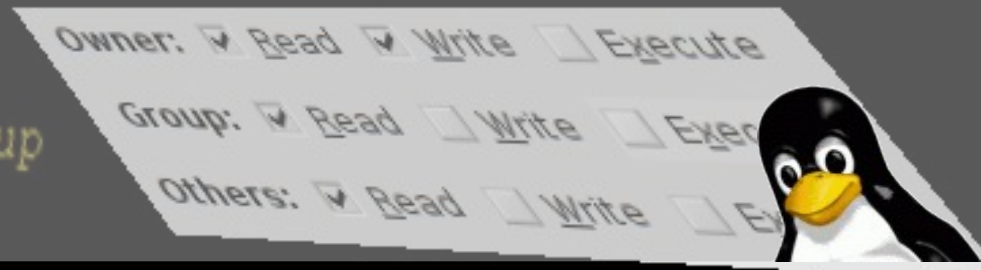
NAME

SYNOPSIS

```
chown [OPTION]... [OWNER][:[GROUP]] FILE...
chown [OPTION]... --reference=RFILE FILE...
```

DESCRIPTION

This manual page documents the GNU version of **chown**. ch
user and/or group ownership of each given file. If o
user name or numeric user ID) is given, that user i
each given file, and the files' group is not chang
followed by a colon and a group name (or numer
spaces between them, the group ownership of the
well. If a colon but no group name follows the
made the owner of the files and the group of the
that user's login group. If the colon and gr
changed, only the group of the files is
chown performs the same function as **chgrp**. If o
empty, neither the owner
changed.



Benutzer, Zugriffsrechte

Skript Seite 147

Ziele

Ziele:

- Die Eigentümer- und Gruppen-konzepte für Dateien kennen
- Mit Zugriffsrechten umgehen können

User IDs (UID)

Unix ist ein Multi-Tasking / Multi-User System.

Um die Benutzer auseinander zu halten wird jedem Benutzer eine UID (UserID) zugeordnet.
Damit die Benutzer einfacher mit dem System arbeiten können werden diese UIDs auf die Benutzernamen und umgekehrt abgebildet.

Anhand der UID kann UNIX entscheiden, ob der Benutzer die gewünschte Aktion (beispielsweise ein File öffnen, löschen, ...) überhaupt ausführen darf.

Group IDs (GID)

Jeder User ist auch mindestens einer Gruppe zugeordnet.

Diese Gruppe kann herangezogen werden, um den Mitgliedern Rechte zu zuordnen aber auch um Rechte zu entziehen.

Die Gruppenamen sind - analog den UserNamen - eine Hilfe um die Gruppen zu benennen. Der Kernel schaut auf die nummerische GID (Group ID)

Maximale Anzahl der Gruppen pro User:
Kernel < 2.6: Ein User kann in maximal 32 Gruppen Mitglied sein.
Kernel ≥ 2.6: entfällt diese Beschränkung.

id

Der Befehl **id** zeigt an welche UID und in welchen Gruppen der aktuelle Benutzer Mitglied ist.

```
$ id
uid=1000(knoppix) gid=1000(knoppix) groups=20(dialout),21(fax),22(voice),24(cdrom),25(floppy),26(tape),27(sudo),29(audio),30(dip),44(video),60(games),100(users),106(usb),110(scanner),115(camera),128(fuse),1000(knoppix)
```

Die UID 1000 ist dem Username "knoppix" zugeordnet
Die aktuelle Gruppe ist "knoppix", GID=1000; Der User ist auch noch Mitglied in den weiteren aufgezählten Gruppen.

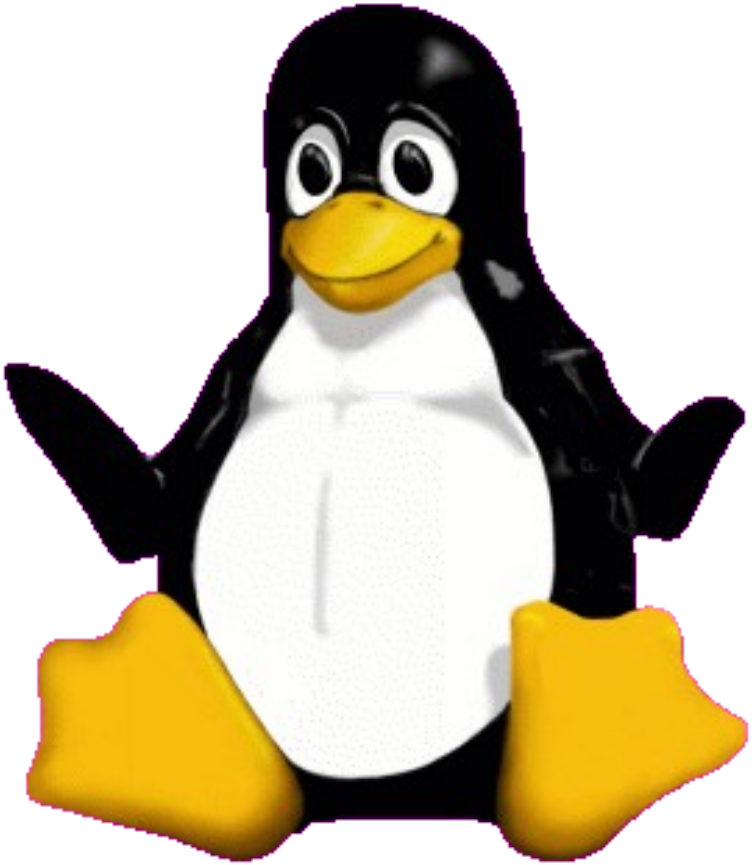
Die UID 0 hat spezielle Rechte auf dem System - Diese UID ist auch bekannt unter dem Usernamen "root"

newgrp

Mit dem Befehl **newgrp** kann die primäre Gruppe des Users temporär (bis zum nächsten login oder erneuten Aufruf von **newgrp**) gewechselt werden.

```
$ id
uid=1000(knoppix) gid=1000(knoppix) groups=20(dialout)...
$ newgrp dialout
$ id
uid=1000(knoppix) gid=20(dialout) groups=20(dialout)...
```

Fragen ?



Übungen

7.1 In welchen Gruppen sind Sie Mitglied? Welche Gruppe ist Ihre primäre Gruppe?

Zugriffsrechte

Einer Datei ist immer einem **einzigen Eigentümer** und einer **einzigen Gruppe** zugeordnet.

Daher gibt es 3 Parteien, die auf eine Datei zugreifen können:

User: Der User ist der Eigentümer der Datei.

Group: Der User ist Mitglied in der Gruppe, die der Datei zugeordnet ist.

Other: Der User ist weder der Eigentümer noch Mitglied in der Gruppe welcher die Datei zugeordnet ist.

Zugriffsrechte Dateien

UNIX unterscheidet drei unterschiedliche Zugriffsrechte:

Read (r): Die Datei darf gelesen werden.

Write (w): Die Datei darf verändert werden.

Execute (x): Die Datei darf ausgeführt werden.

Zugriffsrechte bei Verzeichnissen

Bei Verzeichnissen bedeuten die Zugriffsrechte:

Read: Der Inhalt des Verzeichnis darf gelesen werden.
`ls` kann die Dateien innerhalb des Verzeichnis lesen und anzeigen

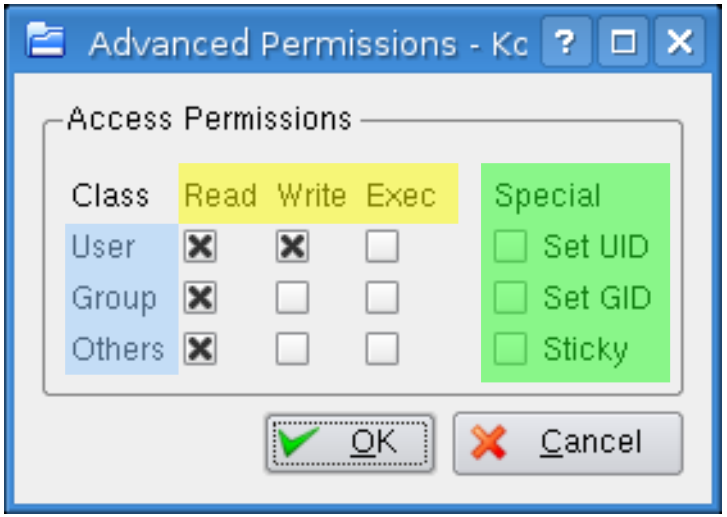
Write: Das Verzeichnis darf verändert werden.
Der User kann Dateien im Verzeichnis erstellen und löschen.

eXecute: Die Datei darf ausgeführt werden.
Der User kann das Verzeichnis benutzen. Er kann mit cd ins Verzeichnis wechseln.

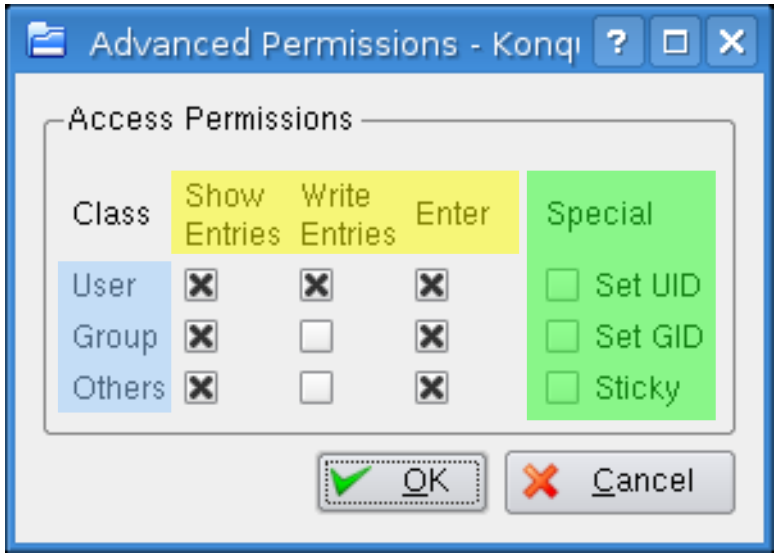
Zugriffsrechte

Für jede der 3 Parteien können die Zugriffsrechte separat verteilt werden:

File Permissions



Dir Permissions



Die grün markierten Special Bereiche sind nur bedingt für die Zugriffsrechte relevant!

Special Flags bei Dateien

Bei ausführbaren Dateien:

SetUID: Ausführbare Programme werden mit den Rechten des Datei-Eigentümer anstatt des mit den Rechten des Benutzers gestartet

SetGID: Der Prozess, der dieses File ausführt, bekommt die Gruppe vom File als primäre Gruppe.

Sticky: verhinderte früher, dass Programme nach dem der Prozess terminierte aus dem Speicher gelöscht wurden, dass bei einem erneuten Aufruf das Programm schneller gestartet werden konnte.
Heute ist die Bedeutung Programm spezifisch

Special Flags bei Verzeichnisse

Bei Verzeichnisse

SetUID: Files, die im Verzeichnis erstellt werden bekommen den User vom Verzeichnis.
Unix und Linux-Systeme ignorieren dies.
FreeBSD kann entsprechend konfiguriert werden.

SetGID: Die File im Verzeichnis 'erben' die Gruppe vom Verzeichnis.

Sticky: Nur der Eigentümer vom File kann das File löschen (vorausgesetzt, dass der Eigentümer auch Schreibrechte auf dem Verzeichnis hat).
Beispielsweise im Verzeichnis für temporäre Dateien (/tmp) kann der Benutzer seine eigenen Dateien löschnen.

Zugriffsrechte

Die Zugriffsrechte werden als 'Three sets of three' dargestellt:

User - Group - Other

Innerhalb dieser Gruppen wird immer die Reihenfolge

Read - Write - Execute (rwx)

eingehalten.

Zugriffsrechte

Ist das Recht gegeben, so wird das mit einem kleinen Buchstaben (**r**,**w** oder **x** für das entsprechende Recht) signalisiert.

Ist das Recht nicht gegeben, so wird das mit einem '-' angezeigt.

```
$ ls -l /dev/sda
brw-rw---- 1 root disk 8, 0 2005-02-26 07:38 /dev/sda
```

- Der Eigentümer (root) kann die Datei lesen und schreiben aber nicht ausführen.
- Mitglieder der Gruppe (disk) können die Datei lesen und schreiben, aber nicht ausführen
- Alle anderen Benutzer können die Datei weder lesen und schreiben noch die Datei ausführen

Zugriffsrechte

Die Special Flags werden an der Stelle des execute Flag angezeigt: SetUID und SetGUI werden als 's' und das Sticky-Bit als 't' dargestellt.

Wenn gleichzeitig das Execute-Recht gegen ist, so werden die Buchstaben klein geschrieben. Fehlt das Execute-Recht, so werden die Special Flags gross geschrieben.

```
$ ls -ld /tmp
drwxrwxrwt 50 root root 3648 2009-05-13 10:31 /tmp

$ ls -l $(which passwd)
-rwsr-xr-x 1 root root 34392 2009-05-11 01:06 /usr/bin/passwd
```

Zugriffsrechte

Pro User-Gruppe können die Rechte auch als Octal-Zahl dargestellt werden:

```
$ ls -l /dev/sda
brw-rw---- 1 root disk 8, 0 2009-04-10 14:21 /dev/sda
```

User-Gruppe:	**User**	**Gruppe**	**Other**	
Recht:	**r w x**	**r w x**	**r w x**	
Wertigkeit:	4 2 1	4 2 1	4 2 1	
Enabled:	1 1 0	1 1 0	0 0 0	
Resultat:	**6**	**6**	**0**	➔ **660**

Zugriffsrechte

Es werden jeweils die Zugriffsrechte verwendet, die am spezifischsten sind:

- Bin ich der Eigentümer, so sind die Rechte für den Eigentümer (User) relevant.
- Bin ich Mitglied in der Gruppe, der das File zugeordnet ist, so sind die Rechte der Gruppe relevant.
- Bin ich weder Eigentümer noch Mitglied in der Gruppe des Files, so sind die Others-Rechte relevant.

```
$ id
uid=1003(heuer) gid=50(staff) groups=50(staff)
$ ls -l datei
----rw-r-- 1 heuer staff 360 2007-06-06 11:09 datei
$ cat datei
cat: datei: Permission denied
```

Ohne Lese-Rechte kann ich die Datei als Eigentümer nicht lesen!

Zugriffsrechte vergeben

Der Rechte können mit **chmod** angepasst werden.

Die Rechte können unabhängig von den vorhanden rechten vom Eigentümer einer Datei oder vom SuperUser vergeben werden.

`$ chmod u+x datei`	Ausführungsrecht für Eigentümer
`$ chmod go+w datei`	setzt Schreibrecht für Gruppe und Rest der Welt
`$ chmod g+rw datei`	setzt Lese- und Schreibrecht für die Gruppe
`$ chmod g=rw,o=r datei`	setzt Lese- und Schreibrecht, löscht Ausführungsrecht für die Gruppe setzt reines Leserecht für den Rest der Welt
`$ chmod a+w datei`	äquivalent zu ugo+w

chmod

+/-: Werden die Rechte im Format (ugoa +/- rwx) angegeben so werden die Rechte entsprechend hinzugefügt oder entzogen. Die Rechte, die nicht erwähnt wurden bleiben unverändert.

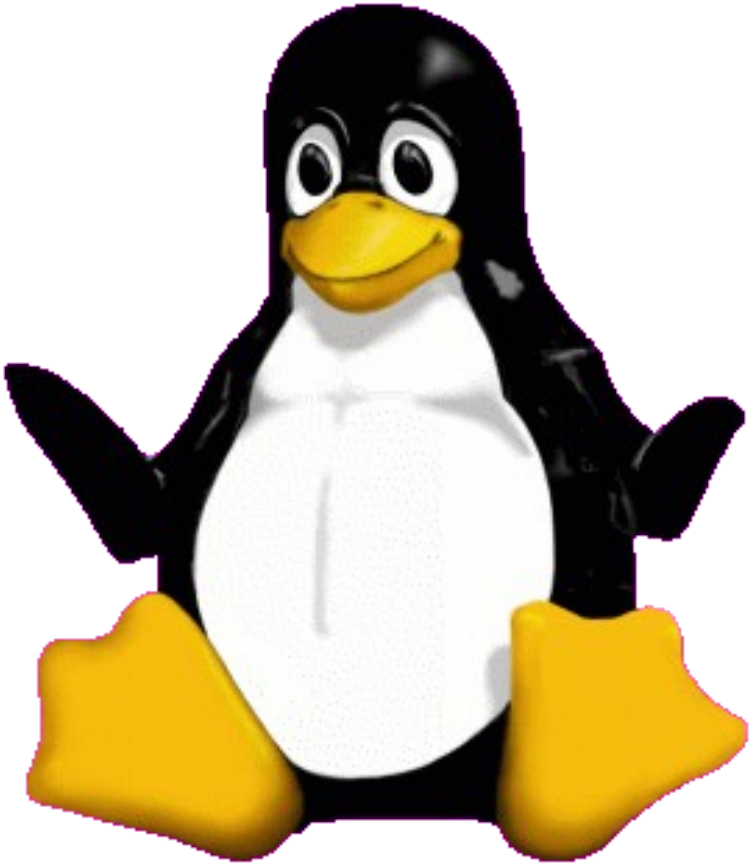
=: Werden die Rechte im Format (ugoa = rwx) angegeben, so werden die Rechte genau so wie angegeben gesetzt. Die Rechte der nicht angegeben 'Gruppen (ugoa)' werden nicht verändert.

nnn: Werden die Rechte im Octal-Format (nnn) angegeben so werden alle Rechte entsprechend gesetzt!

chmod Optionen

`-R, --recursive`	... wie bekannt ...
`--reference=RFILE`	chmod verwendet die Rechte der Datei RFILE.

Fragen ?



Übungen

7.2 Legen Sie je eine Datei mit den folgenden Zugriffsrechten an:

Nur der Eigentümer darf lesen und schreiben, sonst darf niemand etwas.
Der Eigentümer darf die Datei lesen, schreiben und ausführen, Mitglieder der Dateigruppe und der "Rest der Welt" dürfen sie lesen und ausführen
Alle dürfen die Datei nur lesen.

7.3 Wie können Sie die Mitglieder einer Gruppe vom Zugriff auf eine Datei ausschließen, die der Rest der Welt lesen, schreiben, ... darf?

Datei Eigentümer setzen

Der Befehl **chown** (change owner) kann den Eigentümer und die Gruppe einer Datei (oder Verzeichnis) setzen.

Nur vom Superuser kann den Eingentümer (Owner) einer Datei verändern !

Ein Benutzer kann mit chown die Gruppe von Files modifizieren.

```
chown [OPTION]… [OWNER][:[GROUP]] FILE…
chown [OPTION]… --reference=RFILE FILE…
```

OWNER kann der Name oder die UID des Benutzer sein

GROUP kann der Name oder die GID der Gruppe sein.

chown

`-R, --recursive`	... wie bekannt ... und mit Vorsicht zu verwenden!
`--reference=RFILE`	chmod verwendet den Owner und Gruppe der Datei RFILE.
`--from=CURRENT_OWNER:CURRENT_GROUP`	chmod modifiziert den Owner und Gruppe nur dann das File dem CURRENT_OWNER und der CURRENT_GROUP zugeordnet ist.

chown Beispiele

```
chown root:sys datei
    ➔ -rw-r--r-- 1 root sys 36 2007-06-10 20:46 datei

chown 1000:5 datei
    ➔ -rw-r--r-- 1 1000 5 36 2007-06-10 20:46 datei

chown :staff datei
    ➔ -rw-r--r-- 1 hugo staff 36 2007-06-10 20:46 datei

chown -R :staff source

chown --reference=datei *[ch]
```

Datei Gruppe setzen

Der Befehl **chgrp** (change group) kann die Gruppe einer Datei (oder Verzeichnis) setzen.

```
chgrp [OPTION]… GROUP FILE…
chgrp [OPTION]… --reference=RFILE FILE…
```

GROUP kann der Name oder die GID der Gruppe sein.

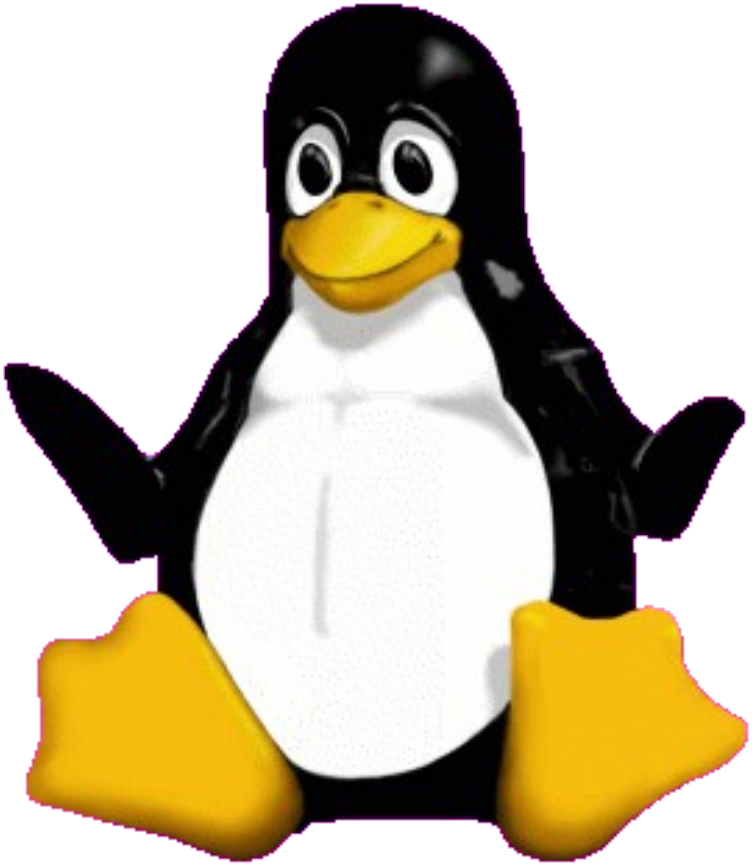
chgrp Optionen

`-R, --recursive`	... wie bekannt ... und mit Vorsicht zu verwenden!
`--reference=RFILE`	chgrp verwendet die Gruppe der Datei RFILE.

chgrp Beispiele

```
chgrp sys datei
    ➔ -rw-r--r-- 1 root sys 36 2007-06-10 20:46 datei

chgrp 5 datei
    ➔ -rw-r--r-- 1 0 5 36 2007-06-10 20:46 datei

chgrp staff datei
    ➔ -rw-r--r-- 1 root staff 36 2007-06-10 20:46 datei

chgrp -R staff source

chgrp --reference=datei *[ch]
```

Fragen ?



Übungen

7.4 Legen Sie eine neue Datei an. Welcher Gruppe ist diese Datei zugeordnet? Verwenden Sie chgrp, um die Datei einer Ihrer anderen Gruppen zuzuordnen. Was passiert, wenn Sie die Datei einer Gruppe zuordnen wollen, in der Sie nicht Mitglied sind?

7.5 Vergleichen Sie die Mechanismen, die verschiedene Dateibrowser (zum Beispiel Konqueror, Nautilus, ...) zum Setzen von Dateirechten, Eigentümer, Gruppe, ... anbieten. Gibt es nennenswerte Unterschiede?