

Betriebssysteme Praxis

SS 2012

Hans-Georg Eßer
Dipl.-Math., Dipl.-Inform.

Foliensatz G: „Zugriffsrechte“
(12.05.2012)
„Manage file permissions and ownership“



Manage file permissions and ownership

Description: Candidates should be able to control file access through the proper use of permissions and ownerships.

Key Knowledge Areas:

- Manage access permissions on regular and special files as well as directories.
- Use access modes such as `suid`, `sgid` and the sticky bit to maintain security.
- Know how to change the file creation mask.
- Use the group field to grant file access to group members.

The following is a partial list of the used files, terms and utilities:
`chmod`, `umask`, `chown`, `chgrp`

Quelle: <http://www.lpi.org/linux-certifications/programs/lpic-1/exam-101/>

Dateien, Benutzer, Gruppen

- Jede Datei
 - ... gehört einem Benutzer (Besitzer, **user**)
 - ... und zu einer Gruppe (**group**)
- Benutzer können Mitglieder in verschiedenen Gruppen sein
- Zugriffsrechte entscheiden, ob eine Datei gelesen (**read**), geschrieben (**write**) oder ausgeführt (**execute**) werden darf

- In welchen Gruppen bin ich Mitglied?

```
$ groups
```

```
fom cdrom floppy audio dip video plugdev netdev  
powerdev scanner
```

- Mitgliedschaft durch Einträge in /etc/group

geregelt:

```
$ grep fom /etc/group
```

```
cdrom:x:24:fom  
floppy:x:25:fom  
audio:x:29:fom  
...  
fom:x:1002:fom
```

- Gruppenmitgliedschaft bearbeiten:
manuell oder (besser!) mit `gpasswd`

- Gruppe mit `gpasswd -a user group` (add)

ergänzen:

```
# gpasswd -a fom neugr
```

Benutzer fom wird zur Gruppe neugr hinzugefügt.

```
# groups fom
```

```
fom cdrom floppy audio dip video plugdev netdev  
powerdev scanner neugr
```

- Entfernen einer Gruppenmitgliedschaft:

```
gpasswd -d user group (delete)
```

```
# gpasswd -d fom neugr
```

Benutzer fom wird aus der Gruppe neugr entfernt.

- Jeder Benutzer ist in einer Standardgruppe Mitglied. Welche ist das?

```
$ id
```

```
uid=1002(fom) gid=1002(fom) Gruppen=1002(fom),  
24(cdrom),25(floppy),29(audio),30(dip),...
```

- Zwei Standards für Standardgruppe
 - Debian-System: Jeder Benutzer hat seine eigene Standardgruppe (User: fom, Group: fom)
 - andere Systeme: Standardgruppe users für alle „normalen“ Benutzer
- Im Namen der Standardgruppe handeln Sie, bis Sie mit `newgrp` die Gruppe ändern.

- Neue Gruppen kann der Administrator mit `groupadd` erzeugen, um Kooperation von Teams zu erleichtern
 - z. B. mit Dateien, die für alle Gruppenmitglieder (und nur diese) les- und schreibbar sind
- Beispielszenario folgt ...

Beispielszenario (1)

- Gruppe `profs`: Mitglieder `prof1`, `prof2`
- Gruppe `studis`: Mitglieder `anna`, `tom`, `fritz` und (!) `prof1`, `prof2`
- Ziele:
 - `profs`-Mitglieder können Daten untereinander austauschen und teilweise auch Studenten zur Verfügung stellen
 - `studis`-Mitglieder können Daten untereinander austauschen und auf die von Profs zur Verfügung gestellten Skripte, Aufgaben etc. zugreifen

- Verzeichnisstruktur

`/srv/profs/`
`/srv/profs/intern/` ; Austausch der Profs untereinander
`/src/profs/intern/klausuren/`
`/srv/profs/public/` ; Lesezugriff für Studenten möglich
`/srv/profs/public/skripte/`
`/srv/studis/`
`/srv/studis/mitschriften/`
`/srv/studis/pruefungsprot/`

- Gruppenzugehörigkeiten und Zugriffsrechte

- `/srv/profs/intern`: gehört Gruppe `profs`; lesen und schreiben für `profs` erlaubt, kein Zugriff für `studis`
- `/srv/profs/public`: gehört Gruppe `profs`; schreiben für `profs` erlaubt, lesen für alle (auch Nicht-Studis)
- `/srv/studis`: gehört Gruppe `studis`; lesen und schreiben für Gruppenmitglieder erlaubt

Beispielszenario (3)

- Umsetzung: später
- Nachteil: keine vernünftige Zugriffsbeschränkung für `/srv/profs/public` möglich
→ ACLs

Unix-Dateiattribute (1)

- Neben Besitzer und Gruppe gibt es noch die sonstigen Systembenutzer (**o**, others)
- ergibt 9 Zugriffsrechte; Notation bei `ls`:

`-rwxrwxrwx`

–	rwx	rwx	rwx
	Besitzer	Gruppe	sonstige

Unix-Dateiattribute (2)

- `chown` (change owner) und `chgrp` (change group) ändern Besitzer und Gruppe einer Datei
- `chmod` (change mode) ändert Zugriffsrechte
- Beispiele:

```
chown fom /tmp/log.txt  
chgrp www-data /var/www/srv1  
chmod o+r /tmp/log.txt  
chmod o-rwx,ug+rw /tmp/log.txt  
chmod u=rw,g=r,o= /tmp/log.txt
```
- Abkürzung `a` (all) für `ogu` (`chmod a=rw ...`)

Unix-Dateiattribute (3)

- numerische Rechte:
 - Leserecht: 4 (2^2)
 - Schreibrecht: 2 (2^1)
 - Ausführrecht: 1 (2^0)
 - aufaddieren, z. B.: rw = Lesen/Schreiben: $4+2=6$
- für Benutzer, Gruppe und Sonstige: **nnn**
 - z. B. **640**:
 - Benutzer: 6 = lesen + schreiben (nicht ausführen)
 - Gruppe: 4 = lesen (nicht schreiben, nicht ausführen)
 - Sonstige: 0 = nichts

Unix-Dateiattribute (4)

- chmod mit numerischen Rechten nutzen
 - `rw- r-- --- = 640` (4+2+0, 4+0+0, 0+0+0)
 - `chmod u=rw,g=r,o= /tmp/log.txt`
`chmod 640 /tmp/log.txt`
- bei der numerischen Angabe kein „Geben“ und „Nehmen“ von Rechten möglich
(wie mit `chmod u+x ...`, `chmod o-rwx ...`)

Unix-Dateiattribute (5)

- Beim Erzeugen einer Datei werden Standardrechte gesetzt – welche das sind, bestimmt die **UMASK (user file creation mask)**

```
$ umask                               Standard:  
0022 ←                               Gruppe: nicht schreiben,  
$ umask a=rw                          Sonstige: nicht schreiben  
$ umask  
0111  
$ touch Datei; ls -l Datei  
-rw-rw-rw-  1 esser users 0 2008-12-04 20:48 Datei  
$ umask u=rw,g=r,o=  
$ umask  
0137  
$ touch Test; ls -l Test  
-rw-r----- 1 esser users 0 2008-12-04 20:50 Test
```

Unix-Dateiattribute (6)

- umask wird von 666 (`rw-rw-rw-`: Standardwert für Dateien) bitweise abgezogen, um konkrete Dateirechte zu berechnen;
- Ausführrecht wird beim Erzeugen einer Datei nie vergeben
- Linux unterstützt diese klassischen Unix-Dateiattribute und einige zusätzliche...

Unix-Dateiattribute (7)

- Dateiattribute nur auf echten Unix-Dateisystemen nutzbar – auf Windows-Datenträgern nur stark eingeschränkt:

```
# mount | grep windows
/dev/sda3 on /windows/D type vfat (rw,gid=100,umask=0002)
# touch /windows/D/Testdatei
# ls -l /windows/D/Testdatei
-rwxrwxr-x 1 root users 0 2006-12-04 21:07 /windows/D/Testdatei
# chmod a-rwx /windows/D/Testdatei
# ls -l /windows/D/Testdatei
----- 1 root users 0 2006-12-04 21:07 /windows/D/Testdatei
# umount /windows/D; mount /windows/D; ls -l /windows/D/Testdatei
-r-xr-xr-x 1 root users 0 2006-12-04 21:07 /windows/D/Testdatei
```

- Windows kennt kein Ausführattribut – wohl aber ein Read-Only-Attribut

Unix-Dateiattribute (8)

- Bedeutung der Attribute für Verzeichnisse:
 - **read**: Verzeichnisinhalt lesen (ls in einem Verzeichnis ausführen)
 - **write**: Verzeichnisinhalt ändern (z. B. neue Datei erzeugen, Datei umbenennen)
 - **execute**: Verzeichnis betreten, also zum aktuellen Arbeitsverzeichnis machen (cd)
 - Standardrechte, von denen die umask abgezogen wird, sind bei Verzeichnissen 777 (denn x = execute steht ja für „Verzeichnis betreten“)

- Zurück zum Beispielszenario

- Ersteinrichtung:

```
root# chown -R root /srv/profs /srv/studis
root# chgrp -R profs /srv/profs/intern
root# chmod ug=rwx,o= /srv/profs/intern
root# chgrp -R profs /srv/profs/public
root# chmod ug=rwx,o=rx /srv/profs/public
root# chgrp -R studis /srv/studis
root# chmod ug=rwx,o= /srv/studis
```

- neue Dateien erzeugen:

```
prof1$ newgrp profs # als „profs“-Mitglied arbeiten
prof1$ umask 0007 # Neue Dateien nicht für andere
prof1$ cd /srv/profs/intern
prof1$ touch pruefung.doc
prof1$ ls -l pruefung.doc
-rw-rw---- 1 prof1 profs ... pruefung.doc
```

SUID, SGID (1)

- Problem: Es gibt Dateien, die Benutzer nur „unter kontrollierten Bedingungen“ ändern dürfen, z. B. die Passwortdatei `/etc/shadow`
 - Änderung an der Datei mit dem Tool `passwd`
 - Dafür sind Root-Rechte nötig
 - Normale Anwender haben keine Root-Rechte
- Zwei Lösungen
 - klassisch: SUID (siehe nächste Folie)
 - neuer: `sudo` (behandeln wir hier nicht)

- Ausführbare Dateien (nur Binaries) können ein SUID- (Set User ID) und/oder ein SGID-Bit (Set Group ID) haben
 - SUID: Programm läuft immer mit den Rechten des Dateibesitzers, meist `root`
 - SGID: Programm läuft immer mit den Gruppenrechten der Dateigruppe (seltener verwendet)
 - Beispiel: `passwd` muss Systemdateien ändern

```
$ ls -l /usr/bin/passwd /etc/shadow
-rwsr-xr-x 1 root root    ... /usr/bin/passwd
-rw-r----- 1 root shadow .. /etc/shadow
```

- SUID- und SGID-Bits mit chmod setzen

```
# cp /usr/bin/passwd /tmp/mypasswd
# chmod u-s,g+s /tmp/mypasswd
# ls -l /usr/bin/passwd /tmp/mypasswd
-rwSr-xr-x 1 root root    ... /usr/bin/passwd
-rwxr-Sr-x 1 root shadow  ... /etc/shadow
```

- s-Bits erscheinen in der ls-Ausgabe immer an der Stelle, wo sonst das x steht
- Diese Bits sind bei Shell-Skripten wirkungslos (in einigen älteren Unix-Versionen funktionierte das auch mit Skripten)

- Ext2-/Ext3-Extra-Flags (immutable, append-only etc.) mit `chattr` bearbeiten

NAME

chattr - change file attributes on a Linux second extended file system

SYNOPSIS

```
chattr [ -RV ] [ -v version ] [ mode ] files...
```

DESCRIPTION

chattr changes the file attributes on a Linux second extended file system.

The format of a symbolic mode is `+--[ASacDdIijsTtu]`.

The operator ``+'` causes the selected attributes to be added to the existing attributes of the files; ``-'` causes them to be removed; and ``='` causes them to be the only attributes that the files have.

- Beispiel für `chattr`:

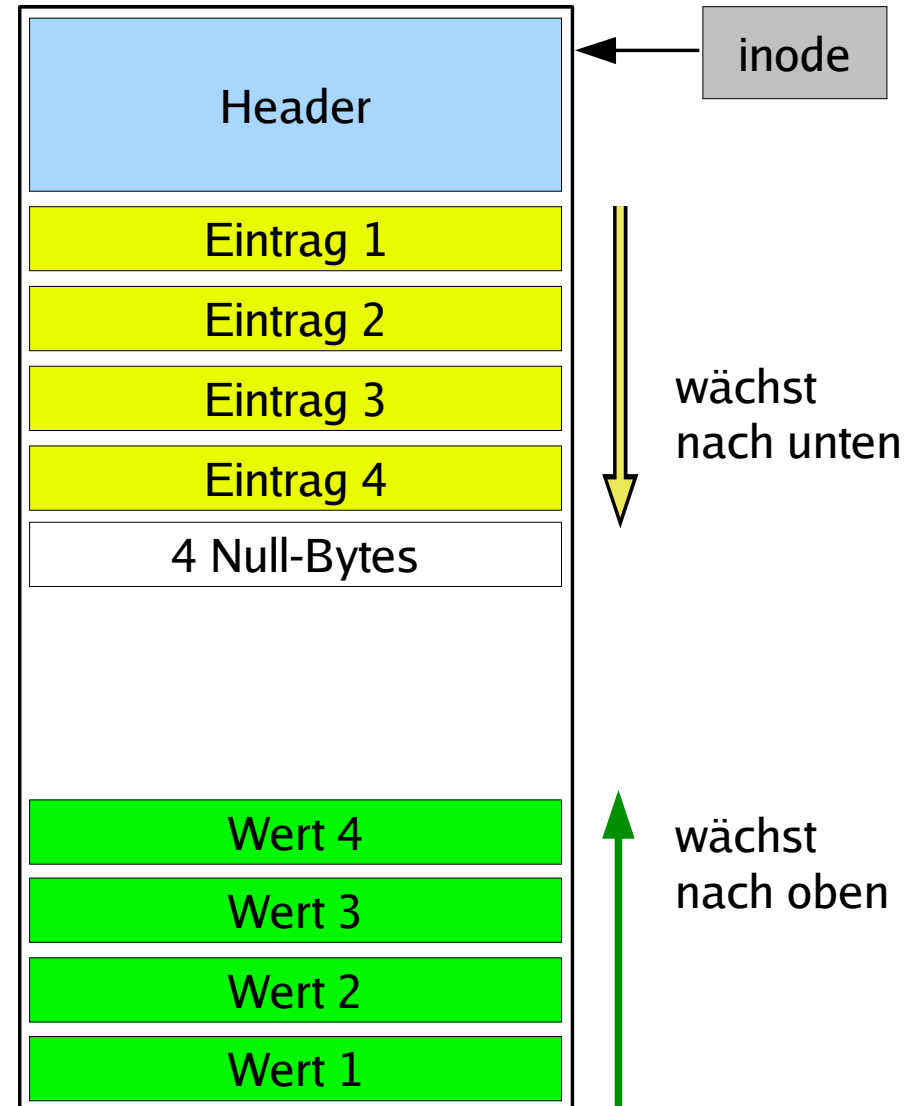
```
# cd /tmp; touch logdatei
# chattr +a logdatei
# echo Hallo >> logdatei          # >> = anhängen
# echo Welt >> logdatei
# cat logdatei
Hallo
Welt
# echo Ueberschreiben > logdatei
bash: logdatei: Die Operation ist nicht erlaubt
```

- Attribute anzeigen mit `lsattr`:

```
# lsattr -l logdatei
logdatei                Append_Only
```


Erweiterte Attribute (1)

- Erweiterte Attribute speichern beliebige Name-/Wert-Paare, u. a. ACLs
- Inode-Größe: 128 Byte
 - kein Platz für erweiterte Attribute
 - Vergrößerung auf 256 Byte nicht effizient
- Lösung: Separater Block für extended attributes



Erweiterte Attribute (2)

- bearbeiten mit `setfattr`, `getfattr`, `attr`:

```
amd64:/home/esser # setfattr -n user.foo -v betriebssysteme test.txt
amd64:/home/esser # getfattr -d test.txt
# file: test.txt
user.foo="betriebssysteme"
```

```
amd64:/home/esser # attr -g user.foo test.txt
Attribute "user.foo" had a 15 byte value for test.txt:
betriebssysteme
```

- Software ist auf dem Debian-System nicht installiert → `apt-get install attr`
- Verwaltung von ACLs über das Paket `acl`
→ `apt-get install acl`
 - Tools: `getfacl`, `setfacl`

Nicht verwechseln:

- Standard-Unix-Dateiattribute
 - UID, GID
 - Standardzugriffsrechte rwx für user/group/others
 - Zugriffszeiten, ...
- Extra-Flags
 - immutable, compressed, secure deletion, ...
- Extended Attributes
 - beliebige, frei definierbare Attribute (inkl. ACLs)