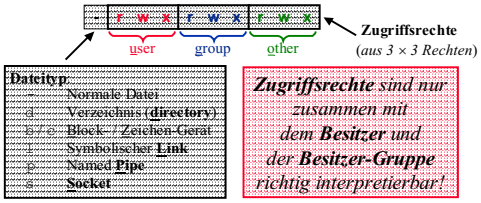
Dateisystem - Zugriffsrechte

- Das UNIX Dateisystem kennt 3 verschiedene Arten von **Zugriffsrechten**:
  - Lesen (`r`=**read**)
  - Schreiben (`w`=**write**)
  - Ausführen (`x`=**execute**)
- Sie sind pro Datei vorhanden für:
  - Den **Besitzer** (`u`=**user**)
  - Die **Besitzer-Gruppe** (`g`=**group**)
  - **Alle anderen** Benutzer (`o`=**others**)

  (`a`=**all**)

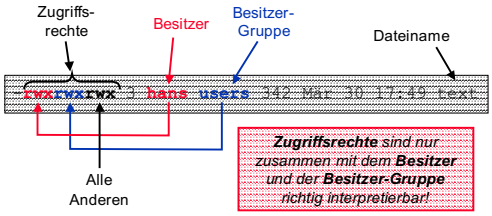
Dateisystem - Zugriffsrechte

- `ls -l` gibt die **Zugriffsrechte** in folgender Form (*ganz links*) aus (*Bindestrich = Recht fehlt*):



Dateisystem - Zugriffsrechte

Bezug Zugriffsrechte ↔ Besitzer(-Gruppe)



Dateisystem - Zugriffsrechte

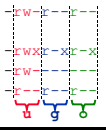
- Jeder **Prozeß** gehört einem **Besitzer** und einer **Besitzer-Gruppe** (*ebenso wie eine Datei*).
  - Der **Besitzer** eines Prozesses ist (*normalerweise*) der Benutzer, der das Programm ausführt.
  - Gleiches gilt für die **Besitzer-Gruppe**.
  - Mit **Set-UID/GID-Rechten** änderbar.
- Beim **Zugriff** eines Prozesses auf eine Datei wird der anzuwendenden **Zugriffsrechte-Satz** ermittelt.
  - Dies geschieht durch **Vergleich** der Besitzer und der Besitzer-Gruppen von **Prozeß** und **Datei**.

Dateisystem - Zugriffsrechte

- In folgender **Reihenfolge** wird verglichen, beim ersten passenden Vergleich wird abgebrochen:
  1. **Besitzer gleich**
     → Rechte des Besitzers anwenden: `rwx.....`
  2. **Besitzer-Gruppe gleich**
     → Rechte der Besitzer-Gruppe anwenden: `....rwx...`
  3. **Sonst**
     → Rechte "Alle anderen" anwenden: `.....rwx`
- Fehlen also dem Besitzer Rechte, die "Alle anderen" haben, so gelten trotzdem für ihn *nur* seine Rechte.

Dateisystem - Zugriffsrechte

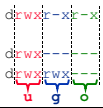
- Bedeutung der Zugriffsrechte bei **Dateien**:
  - **Lesen**: Datei-Inhalt kann gelesen werden (`more`).
  - **Schreiben**: Datei-Inhalt kann verändert werden (`>`).
  - **Ausführen**: Datei kann als **Programm** ausführt werden.
- Beispiel:



Datei ist für den **Besitzer** les- und schreibbar, für den **Rest** ist sie nur lesbar.
Wie oben, nur daß die Datei **für alle** auch ausführbar ist.
Datei ist **nur** für den **Besitzer** les- und schreibbar.
Datei ist **für alle** nur lesbar (*auch für den **Besitzer***).

Dateisystem - Zugriffsrechte

- Bedeutung der Zugriffsrechte bei **Verzeichnissen**:
  - **Lesen**: Verzeichnisinhalt kann aufgelistet werden (`ls`).
  - **Schreiben**: Dateien können angelegt / gelöscht / ... werden (`touch`, `cp`, `mv`, `ln`, `rm`, `mkdir`, `rmdir`).
  - **Ausführen**: In Verzeichnis kann **gewechselt** werden (`cd`).
- Beispiel:



Besitzer darf im Verzeichnis Dateien anlegen und löschen,
Rest darf Verzeichnis lesen und in Verzeichnis wechseln.
Verzeichnis ist nur für **Besitzer** lesbar/schreibbar/besuchbar
Besitzer und **Besitzergruppe** dürfen alles, der **Rest** nichts.

Dateisystem - Zugriffsrechte

Rechte-Anomalie

Dateirechte entscheiden über die Berechtigung von Operationen am **Datei-Inhalt** (*lesen, kopieren, ändern*).	**Verzeichnisrechte** entscheiden über die Berechtigung von Operationen an darin stehenden **Datei-Namen** (*erstellen, umbenennen, verschieben, löschen*).

- D.h. eine Datei kann möglicherweise **gelöscht** werden, obwohl man **kein Schreibrecht** auf sie hat ☹.
- Das **Recht "sticky"** behebt diese Anomalie (→ *Aufbau-Kurs*).

Dateisystem - Zugriffsrechte

`chmod [OPTIONEN] RECHTE DATEI...`

- Verändert **Datei-Zugriffsrechte** (**change mode**):
  - *Nur dem **Besitzer** einer Datei möglich (oder `root`)!*
- Die *RECHTE* sind folgendermaßen anzugeben:

Symbolisch	Numerisch
`[ugoa][+-=][rwx]` (`+`: *Hinzufügen*, `-`: *Wegnehmen*, `=`: *absolut Setzen*, ***Komma*** *trennt symbolische Angaben:* `u+x,g-w`)	`000-777` **3-stellige Oktalzahl**

Dateisystem - Zugriffsrechte

- Zugriffsrechte sind auch als **Oktalzahl** `0-7` gemäß folgender Tabelle angebbar (`r`=4, `w`=2, `x`=1, **3 Bits**):

Rechte	Wert
r w x	7
r w -	6
r - x	5
r - -	**4**
- w x	3
- **w** -	**2**
- - **x**	**1**
- - -	0

Um alle Rechte zu setzen, ist eine **3-stellige Oktalzahl** `000-777` anzugeben:
- 1. Stelle = Besitzer
- 2. Stelle = Besitzer-Gruppe
- 3. Stelle = Alle anderen

Dateisystem - Zugriffsrechte

Tipp

- **Symbolische Form** eignet sich besser zum **Hinzufügen** und **Wegnehmen** einzelner Zugriffsrechte:
  - `chmod` **`o-rwx`** *DATEI*
  - `chmod` **`a+x`** *DATEI*
- **Numerische Form** eignet sich besser zum **vollständigen Setzen** aller Zugriffsrechte auf einmal:
  - `chmod` **`700`** *DATEI*
  - `chmod` **`544`** *DATEI*

Dateisystem - Zugriffsrechte

Beispiel

`chmod` **`u=rw,go=r`** *DATEI*	**Besitzer** hat Schreib- und Leserecht der **Rest** hat nur Leserecht.
`chmod` **`a+x`** *DATEI*	**Jeder** erhält (*zusätzlich*) Ausführungsrecht.
`chmod` **`-R a+x`** *VERZ*	Wie oben, nur werden alle Unterverzeichnisse von *VERZ* und ihre Dateien darin eingeschlossen (*Vorsicht!*).
`chmod` **`o-rwx`** *DATEI*	**Rest** bekommt die Lese-, Schreib- und Ausführungsrechte entzogen.
`chmod` **`644`** *DATEI*	Wie erstes Beispiel.
`chmod` **`666`** *DATEI*	**Jeder** erhält Schreib- und Leserecht, aber kein Ausführungsrecht.

Dateisystem - Zugriffsrechte

Hinweis für Windows-Anwender

- Es gibt kein Recht, das die **Änderung der Zugriffsrechte** verhindert (*Permission-Recht*).
  - `chmod` nur **Datei-Besitzer** (*oder* `root`) erlaubt.
- Es gibt kein Recht, das den **Besitzübernahme** regelt.
  - `chown` nur `root` erlaubt.
  - `chgrp` nur **Datei-Besitzer** (*oder* `root`) erlaubt.
- Es gibt kein Recht, das den **Inode-Zugriff** regelt.
  - `touch` nur **Datei-Besitzer** (*oder* `root`) erlaubt.

Dateisystem - Zugriffsrechte

- **Standardzugriffsrechte** beim Anlegen **neuer** Dateien / Verzeichnisse sind:

Verzeichnis	`777 = rwxrwxrwx`
Datei	`666 = rw-rw-rw-`

- **Begründung**
  - Bei **Verzeichnissen** ist das Ausführungsrecht nötig, damit man mit `cd` hineinwechseln kann.
  - Bei **Dateien** ist es gefährlich, ihnen standardmäßig Ausführungsrechte zu geben.

Dateisystem - Zugriffsrechte

`umask [MASKE]`

- **Entfernt** einige der Standard-Zugriffsrechte beim Anlegen **neuer** Dateien / Verzeichnisse.
  - Besteht aus **3 Oktalziffern**, sie stehen für die **zu entfernenden** Rechte (**maskieren**).
  - Setzt die **Maske** oder gibt ihren aktuellen Wert aus:

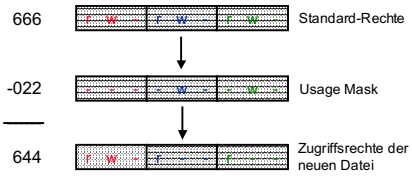
    `umask MASK` (*z.B.* `umask 027`)

    `umask` → `022` (*meist*)

*Ändert Rechte bereits vorhandener Dateien **nicht**!*

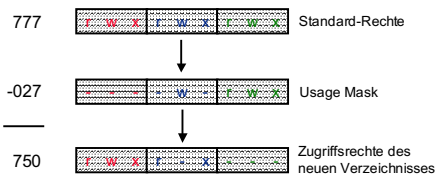
Dateisystem - Zugriffsrechte

Anlegen neuer Dateien



Dateisystem - Zugriffsrechte

Anlegen neuer Verzeichnisse



Dateisystem - Zugriffsrechte

Sinnvolle `umask`**-Werte**:

- `000` – erzeugt **Standard-Datei-Zugriffsrechte** `666` und **Standard-Verzeichnis-Zugriffsrechte** `777`
- `002` – "Andere" dürfen nicht schreiben (*Standard bei RedHat*).
- `007` – "Andere" dürfen nichts.
- `022` – "Andere" & Gruppen-Mitglieder dürfen nicht schreiben (*Standard bei SuSE, Solaris und vielen UNIX-Systemen*).
- `027` – "Andere" dürfen nichts, Gruppen-Mitgl. nicht schreiben.
- `077` – "Andere" & Gruppen-Mitglieder dürfen nichts.