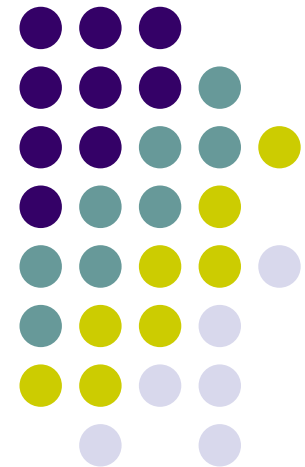


Unixkurs

Dateien
Verweise
Gerätedateien



Dateien

Es existieren sechs verschiedene Arten von Dateien (Files):

1. einfache Dateien (regulären Dateien bzw. regular files)
2. Verzeichnisse oder Directories
3. Verweise auf Dateien, (symbolic links) und zusätzliche Verzeichniseinträge (hard links)
4. Gerätedateien(device files)
5. spezielle Warteschlangendateien (pipes oder FIFO – files)
6. Anschlußdateien (sockets) für Verbindungen zwischen Prozessen.

Anzeigen von Dateiattributen

- Mit dem Kommando `ls -l` werden neben dem Namen auch noch angezeigt:
 - die Dateart
 - die Zugriffsrechte
 - der Besitzer
 - die Gruppe
 - die Zeitmarke
 - die Anzahl der Links

Dateiart

- Die Dateiart wird dabei mit folgenden Abkürzungen gekennzeichnet und steht vor den Zugriffsrechten:
- **Abk. Dateiart**
 - - reguläre Datei
 - d Verzeichnis
 - l Verweis
 - b Gerätedatei, die blockweise angesprochen wird
 - c Gerätedatei, die zeichenweise angesprochen wird
 - p Warteschlangendatei
 - s Anschlußdatei

Dateien - Zugriffsrechte

- Jede Datei besitzt Zugriffsrechte (permissions).
- Durch Zugriffsrechte kann Eigentümer einer Datei festlegen, ob andere Benutzer die Datei lesen, ändern oder ausführen dürfen.
- Entsprechend werden die Zugriffsrechte mit Lesen, Schreiben und Ausführen –
 - read (r),
 - write (w),
 - execute (x) –
benannt.

Datei Eigentümer

- Jede Datei hat einen Eigentümer.
- Eigentümer = Benutzer der Datei erzeugt hat
- Eigentümer kann Zugriffsrechte festlegen und für verschiedene Benutzergruppen differenzieren.
- Die verschiedenen Benutzergruppen sind
 - Eigentümer,
 - Gruppe und
 - alle anderen Benutzer.
- Dementsprechend sind die Zugriffsrechte in drei Felder untergliedert:
 - drei Character für Zugriffsrechte des Eigentümer,
 - drei Character für Zugriffsrechte der Gruppe und
 - drei Character für die Zugriffsrechte aller Anderen.

Datei - Zugriffsrechte

- Beispiele:
 - `rwxr-xr-x`

Der Eigentümer des Files darf lesen, schreiben und ausführen.
Hingegen darf die Gruppe und alle Anderen das File nur lesen und ausführen.
 - `rw-----`

Der Eigentümer des Files darf lesen und schreiben.
Alle anderen Benutzer haben hingegen keinen Zugriff auf das File.
 - `rwxrwxrwx`

Alle Benutzer dürfen das File lesen, schreiben und ausführen.

Zugriffsrechte für Verzeichnisse

- Das Zugriffsrecht Ausführen hat bei Verzeichnissen die Bedeutung von "benutzen" (in das Verzeichnis wechseln, Dateien aus dem Verzeichnis lesen oder kopieren, u.a.) dürfen.
- Die Zugriffsrechte die ein File besitzt, hängen noch zusätzlich von den Zugriffsrechten des Verzeichnisses ab, in dem das File liegt.
- Besitzt ein File die Zugriffsrechte `rwxrwxrwx` können andere Benutzer *nicht* auf das File zugreifen, falls für das Verzeichnis, in dem das File liegt, nicht das execute- und read-Recht für die Gruppe und alle anderen Benutzer gesetzt ist.
- Das bedeutet im geschilderten Fall das Verzeichnis besitzt das folgende Zugriffsrecht: `rwx-----` (also keine Zugriffsrechte für alle anderen Benutzer).

Übung

- Bestimmen sie die Dateiart, sowie die Zugriffsrechte für Eigentümer Gruppe und alle Anderen für einige Files.

umask

- Anfängliches Zugriffsrecht wird von dem zu erzeugenden Prozess vergeben.
- Für reguläre Dateien die Rechte Lesen und Schreiben für jeden, für Programme auch noch das Recht zum Ausführen.
- Durch umask lassen sich Rechte beschränken.
- Dazu muß als Parameter eine dreistellige Oktalzahl (für Eigentümer, Gruppe und alle Anderen) angegeben werden, mit deren Hilfe bestimmte Rechte bei der Erzeugung einer Datei unterbunden werden.
- Die Oktalzahl hat dabei folgende Bedeutung:
 - 0 alle Zugriffsrechte
 - 1 nicht ausführbar
 - 2 nicht schreibbar
 - 3 (= 2+1) weder schreib- noch ausführbar
 - 4 nicht lesbar
 - 5 (= 4+1) weder les- noch ausführbar
 - 6 weder les- noch schreibbar
 - 7 (= 6+1) kein Zugriff

Beispiele umask

- Geben Sie folgende Kommandos der Reihe nach ein und analysieren Sie die Ausgabe:
- `prompt> umask`
- `prompt> umask 000`
- `prompt> touch risiko`
- `prompt> ls -l`
- `prompt> umask 026`
- `prompt> touch sicher`
- `prompt> ls -l`

chmod

- Ändern Zugriffsrechte einer Datei
 - `prompt> chmod <Änderungsbeschreibung> <Datei>`
- Änderungsbeschreibung kann entweder als Oktalzahl (*absoluter Modus*) oder durch Buchstabenkennung (*symbolischer Modus*) angegeben werden.
- Absoluter Modus:
 - 0 : - keine Rechte an der Datei
 - 1 : x Recht zum Ausführen der Datei (Programme, Scripts)
 - 2 : w Recht zum der Datei Schreiben
 - 4 : r Recht zum Lesen der Datei
- Es können mehrere Rechte an einer Datei vergeben werden, indem die entsprechenden Oktalwerte addiert werden.
 - `prompt> chmod 731 MeinScript`

chmod

- Symbolischer Modus:
- `prompt> chmod Gruppe(n) +/-/= Recht(e)`
- Für Gruppe können folgende Zeichen stehen:
 - u: user ist Eigentümer der Datei
 - g: group ist die zugeordnete Gruppe
 - o: others ist der Rest der Welt
 - a: all sind alle (entspricht ugo)
- +, - oder = bestimmen, ob Rechte hinzugefügt, entzogen oder genau so gesetzt werden sollen.
- Als Letztes wird noch angegeben welche Rechte betroffen sind. Folgende Zeichen sind möglich:
 - r: read - Leserecht
 - w: write - Schreibrecht
 - x: execute - Ausführungsrecht

chmod Beispiele und Übung

- Beispiele:
 - `prompt> chmod a+x Datei`
 - `prompt> chmod g=rx Datei`
 - `prompt> chmod o-w Datei`
- Übung:
 - Erzeugen Sie eine Datei und verwenden Sie die dargestellten Methoden um verschiedene Zugriffsrechte zu setzen.

Dateikommandos

- Kommandos zur Manipulation von regulären Dateien und hauptsächlich Textdateien sind:
 - touch
 - cp
 - mv
 - rm
 - cat
 - more/less
 - head/tail
 - grep

touch

- Mit touch kann der Zeitpunkt der letzten Änderung einer Datei verändert werden.
- Ohne Optionen wird der aktuelle Zeitpunkt benutzt (Datum und Zeit).
- Existiert die Datei noch nicht, wird sie angelegt.
- Beispiel:
 - `touch Datei_1`
 - Die Datei Datei_1 erhält die aktuelle Zeit als Änderungsdatum.

cp

- Zum Erzeugen einer Kopie einer regulären Datei dient das Kommando cp (copy).
- copy kann entweder von einer Datei eine Kopie unter einem neuen Namen anlegen,
 - `prompt> cp original kopie`
- oder von einer oder mehreren Dateien jeweils eine Kopie unter dem bisherigen Namen in ein anzugebendes Verzeichnis ablegen.
 - `prompt> cp ReadMe.txt original /home/pub`

mv

- Zum Abändern des Namens einer Datei dient das Kommando mv (move für verschieben).
- mv verschiebt eine Datei in eine andere Datei
 - `prompt> mv Datei_1 Datei_2`
- oder ein anderes Verzeichnis.
 - `prompt> mv Datei_1 /tmp`
- Dabei wird dann die Originaldatei gelöscht.

rm

- rm (remove) löscht Files.
- Wenn unter Unix eine Datei gelöscht ist, so ist sie unwiederbringlich verloren! Die einzige Möglichkeit, die Daten wiederzuerlangen, ist auf eine Sicherungskopie (Backup) zurückzugreifen.
- Beispiel:
 - `prompt> rm Datei_1`
- Mit der Option -r (rekursiv) können ganze Verzeichnisbäume auf einmal gelöscht werden.
 - `prompt> rm -r Verzeichnis_1`
- **Aber Vorsicht - Unix löscht ohne Nachfrage!**

cat / more / less

- Zur Ausgabe des kompletten Inhalts eines Files
 - `prompt> cat /etc/hosts`
- Zum Hintereinanderhängen (konkatenerieren) von Dateien.
 - `prompt> cat file1 file2`
- seitenweises Anzeigen einer Datei durch Weitergabe der Ausgabe von cat an more
 - `prompt> cat beispiel.txt | more`
- more entspricht der DOS - Option `/p` (seitenweise Ausgabe). Ein komfortableres Blättern kann durch less (statt more less an der Kommandozeile eingeben) erreicht werden (Vorwärts- und Rückwärtsbewegung durch Pfeiltasten).
 - `prompt> cat beispiel.txt | less`
- less wird durch Drücken der Taste `q` (quit) verlassen

head / tail

- head

- Gibt die ersten 10 Zeilen einer Datei aus. Mit der Option -n zeilen kann die angegebene Anzahl von Zeilen ausgegeben werden.

- `prompt>head -15 /etc/hosts`

- tail

- gibt die letzten Zeilen einer Datei aus. Mit der Option -n kann die Anzahl der Zeilen angegeben werden.

- `prompt> tail /etc/hosts`

- `prompt> tail -2 /etc/hosts`

grep

- Mit dem Dienstprogramm grep kann eine oder bestimmte Dateien nach bestimmten Wörtern oder Textmustern durchsucht werden.
- Das Textmuster muß dabei als erster Parameter angegeben werden und kann ein regulärer Ausdruck sein.
- Alle Zeilen, in denen das Textmuster vorhanden ist, werden ausgegeben.
- Die Suche und das Ergebnis kann durch die folgenden Optionen beeinflusst werden:
 - -i: ignoriert Groß- oder Kleinschreibung
 - -l: zeigt nur Dateinamen an
 - -c: zeigt die Anzahl der gefundenen Zeilen
- Beispiel:
 - `prompt> grep -i minnie /etc/hosts`
- Durchsucht die Datei /etc/hosts nach der Zeichenkette minnie, wobei auch Minnie oder MINNIE ein Ergebnis liefern.

Verweise

- Mit **Verweisen (links)** lassen sich durch neue Einträge in Verzeichnissen **weitere Namen für Dateien** jeder Art erzeugen.
- Diese Namen müssen jedoch **nicht im gleichen Verzeichnis** wie das Original abgelegt werden, sondern in irgendeinem beliebigen.
- Unix unterscheidet zwei Arten von Verweisen:
 1. **Zusätzliche Verzeichniseinträge (hard links)**
 - dabei wird ein **weiterer Eintrag mit einem anderen Namen** in dem Verzeichnis auf den gleichen Indexknoten vorgenommen.
 2. **Symbolische, logische Verweise (symbol links)**
 - dabei wird ein **neuer Indexknoten mit einem weiteren Namen** für das gleichen File angelegt.

Indexknotennummer

- Verweise erlauben es also, einem einzelnen File mehrere Namen zu geben.
- Files werden vom System identifiziert durch ihre **Indexknotennummer (inode number)**.
- Die Indexknotennummer ist ein eindeutiger Identifier für das File.
- Das Kommando
 - `ls -i <filenames>`
 - zeigt die Indexknotennummer der Files

i-node-Liste

- Jede Datei ist in einem „Inhaltsverzeichnis“ vermerkt, der **i-node-Liste**.
- Ein Element einer solchen Liste wird als i-node bezeichnet.
- Ein Verzeichnis kann betrachtet werden als eine Liste von Indexknotennummern mit dem korrespondierenden Filenamen.
- Jeder Filenamen in einem Verzeichnis ist ein **Verweis auf einen Indexknoten**.

Hard Links

- Mit dem `ln - Kommando (link)` werden mehrfache Verweise für einen File angelegt.
- Damit werden über mehrere Verzeichniseinträge ein und dieselbe Datei bezeichnet (den gleichen Indexknoten).
- Beim Löschen von Dateien bleibt die Datei erhalten, solange noch mindestens ein Verzeichniseintrag existiert.
- Für Verzeichnisse können normale Benutzer keine zusätzlichen Verzeichniseinträge erzeugen.

Übung Hard Links

- Überprüfen Sie die Indexknotennummer für einen File in Ihrem Verzeichnis mit `ls -li`.
- Setzen sie einen Hardlink mit
 - `ln <alter Filename> <neuer Filename>`
- Überprüfen Sie die Indexknotennummer des neuen Files und sie stellen fest, dass die Indexknotennummer des alten Files der Indexknotennummer des neuen Files entspricht.
- Löschen sie eines dieser Files und stellen Sie fest, dass nur ein Verweis (Namen) auf den File gelöscht wurde und weitere Verweise (Namen) existieren.

Symbolic Links

- Symbolische Verweise werden mit dem Kommando
 - `ln` mit der Option `-s` (symbolic) angelegt.
- Damit werden für den neuen Verzeichniseintrag ein neuer weiterer Indexknoten für ein und dieselbe Datei angelegt.
- Symbolic Links sind flexibler als zusätzliche Verzeichniseinträge (hard links), denn das Ziel kann sogar durch eine andere Datei gleichen Namens ersetzt werden.
- Symbolic Links sind jedoch unsicher, denn es kann passieren, daß das Ziel gelöscht oder umbenannt wurde und der symbolic link somit ungültig wird.

Symbolic Links

- Symbolische Verweise können auch auf Verzeichnisse zeigen
- Sie dienen dem Anwender oftmals als Abkürzung zu häufig verwendeten Dateien.
- Übung:
 - Setzen Sie einen symbolischen Verweis auf eine Datei mit
 - In -s
 - Überprüfen Sie die Indexknotennummern der beiden Files und stellen sie fest, daß sie verschieden sind.
 - Überprüfen Sie mit ls -l wie symbolische Links dargestellt werden.

Gerätedateien

- In dem Verzeichnis `/dev` verbergen sich die Gerätetreiber für die Hardwarekomponenten und Geräte des Systems.
- Dabei handelt es sich nicht um Dateien im herkömmlichen Sinne, sondern **nur Indexknoten**.
- Diese Spezialdateien **stellen eine Verbindung zwischen dem Anwenderprogramm und den Gerätetreibern im Kernel her**.
- Die Verbindung zum Kernel wird über Slots oder Kanäle hergestellt, die numeriert sind und hinter denen sich die Treiber für die Geräte verbergen.

Gerätedateien

- Die Nummer des Gerätetreibers wird als **Hauptgerätenummer (Major Device Number)** bezeichnet.
- Ein Treiber kann **mehrere Geräte des gleichen Typs** verwalten (z.B. verschiedene Typen von Floppys oder Festplatten).
- Um die einzelnen Geräte zu unterscheiden, wird dem Treiber eine zweite Zahl, die **Untergerätenummer (Minor Device Number)** übergeben.
- Diese beiden Zahlen charakterisieren jede Datei im /dev Verzeichnis.

Gerätedateien

- Mit
 - `ls -l`
 - wird die Haupt- und Untergerätenummer anstelle der Dateigröße angezeigt.
- Ob ein Gerät als
 - Zeichendevice (character) oder
 - Blockdevice (block) angesprochen wird,
 - ist am ersten Buchstaben (entweder b oder c) der Zugriffsrechte zu sehen.
- Beispiel:
 - `prompt> ls -l /dev/hd*`
 - `prompt> ls -l /dev/sd*`

Gerätedateien

- Für Anwender von Interesse sind die Dateien beginnend mit:
 - /dev/hd für Plattenlaufwerke (hard disk drives)
 - /dev/fd für Diskettenlaufwerke (floppy disks)
 - /dev/sd für SCSI-Laufwerke (SCSI drives)
 - /dev/lp für parallele Druckerschnittstelle (line printer)
 - /dev/null für "Schwarzes Loch" alle Daten die an dieses Geräte gesendet werden, sind verloren.
 - /dev/tty für virtuelle Konsolen, die sie durch Drücken von alt F1, alt F2 und so weiter zur Verfügung gestellt bekommen.

Übung

- Lassen Sie sich die unterschiedlichen Gerätedateien auf ihrem System anzeigen