Christian-Albrechts-Universität zu Kiel
Institut für Informatik und Praktische Mathematik

Prof. Dr. W.-P. de Roever
Dr. Martin Steffen

Vorkurs Informatik

Wintersemester 2002/03

Rechnerorganisation (2)

7.–8. Oktober 2002

Termin: 7.–8. Oktober 2002

Aufgabe 1 (Verzeichnisstruktur und Rechte) Legen sie ein paar (weitere) Dateien und Verzeichnisse an und werfen Sie einen Blick deren *Zugriffsrechte*.

1. Stellen Sie fest, welche Voreinstellungen Ihnen die Systemadministration bzgl. den Rechten von

   - neuen Unterverzeichnissen
   - neuen Dateien

   vorgegeben hat

2. Können Sie feststellen, welche Dateien Ihr *Nachbar* in *seinem* Home hat (einschließlich der versteckten Dateien?)

3. Setzen Sie die Rechte Ihres Homeverzeichnisses so, daß *jeder* lesen kann, welche Dateien und Unterverzeichnisse Sie besitzen. Stellt das eine Gefahr da? Impliziert diese Entscheidung, daß jeder den *Inhalt* der Dateien lesen kann? Oder den Inhalt der Unterverzeichnisse?

4. Erzeugen Sie in Ihrem Home ein „privates“ Verzeichnis an, welches nur Sie (d.h., der *account* der Ihnen und Ihrem Übungspartner zugeteilt wurde) lesen kann.[1] Welche Rechte würden Sie dabei empfehlen, was die „*Ausführbarkeit*“ des neuen Verzeichnisses betrifft?

5. **optional** (für den Anfang nicht wichtig/für den etwas Fortgeschritten): Unter Punkt 1 haben Sie herausgefunden, was die Administration für Ihren account als Standardrechte für angemessen hält. der Befehl `umask` erlaubt, sich diese Voreinstellung anzuschauen (wenn man die Ausgabe von `umask` zu interpretieren versteht ...) und zu ändern.

[1] Beachte natürlich: der sogenannter “Superuser” (root) kann das Verzeichnis auf jeden Fall lesen, welche Rechte man auch immer wählt. Zur Not, falls es die Rechte (sinnloserweise) so eingeschränkt wurden, daß *niemand*, nichteinmal man selbst, die Daten lesen kann, kann *root* die Rechte ändern und dann lesen. Vor *root* gibt es (fast) keinen Schutz.

Setzen Sie Ihre Einstellung (zeitweise) so, daß alles was Sie neu anlegen, ausschließlich für Sie zugreifbar ist und so, daß alles für alle *lesbar* ist.[2]

Aufgabe 2 (Dateiformate) 1. Unter `~kurs059/tmp/datei1.txt.gz` liegt eine Datei. Was für eine Datei ist das? Was ist ihr Inhalt?

2. Was ist der Inhalt von `~kurs059/tmp/datei2.ps`?

3. **Für die Harten:** Und was steht in `~kurs059/tmp/datei3.tar`? Hinweis: schauen Sie sich die Manpages zum Befehl `tar` an. Da `tar` recht vielseitig ist, ist die Beschreibung allerdings bereits recht lang und die Beispiele schon recht esoterisch bis kryptisch.

Aufgabe 3 (Prozesse und Shell) Ein Prozeß ist (grob) ein *Programm in Ausführung.* Rufen Sie ein beliebiges Programm auf, sagen wir, die X-Version der `Manpages`:

1. Rufen Sie in einer Shell
   (a) `xterm` bzw.
   (b) `xterm &`

   auf. Beobachten Sie den Unterschied. Beobachten Sie, was im Fall a) passiert wenn man (in der ersten shell) `^Z` = "*Control-Z*" eingibt. Und was, wenn man danach `bg` eingibt? Beobachten Sie, was im Fall b) passiert, wenn man hinterher `fg` eingibt.

2. Finden Sie heraus, *wo* im System sich das Programm, d.h., die ausführbare Datei `xman` befindet.[3]

3. Terminieren Sie das Programm
   (a) mittels `^C` = "*Control C*"[4]
   (b) mittels des Befehls `kill`, wobei Sie vorher vermutlich die *Nummer* des Prozesses herausfinden müssen
   (c) in beiden Fällen: mittels des Fenstersystems

4. Tippen Sie an einem geeigneten Ort das Kommando

   ```
   ls -lag > datei.txt
   ```

   ein. Was passiert? Was passiert wenn man es nochmals macht? Was passiert, wenn man danach

   ```
   ls -lag >> datei.txt
   ```

[2] Seien Sie vorsichtig bei Dieser Übung. Es ist nicht gut, wenn man für Fremde Verzeichnisse/Dateien auf "schreibbar" setzt (warum nicht?). Sie können natürlich die Rechte immer nachträglich ändern: `umask` dient dazu, die Rechte, die Sie *in der Mehrzahl der Fälle* bevorzugen, zu standardisieren.

[3] Eine Möglichkeit, den Ort zumindest einzugrenzen, wäre mittels eines globalen `find`. **Bitte nicht machen!** Neben der Tatsache, daß dies nicht direkt die Version gibt, die tatsächlich ausgeführt wird, wird ein *globale* Suche mittels `find` von den Systemadministratoren stark mißbilligt. Der Grund ist nicht, daß `find` eine schlechter Befehl ist (er ist oft nützlich), sondern an der Netzstruktur des Rechnerpools. Näheres in den FAQs des Instituts. Morgen allgemein zu dem Thema Netze mehr.

[4] Ein Standardtaste zur Terminierung

macht?

Aufgabe 4 (Wildcards/Suchen) 1. Zeigen Sie in Ihrem Home-Verzeichnis alle Dateien, die ein `b` enthalten. Achten Sie auf die versteckten Dateien.

2. Suchen Sie mit dem Befehl `find` alle Dateien unterhalb ihres Homeverzeichnis mit der Endung/Extension `txt` (Beachte: `find` nicht auf das globale root-Verzeichnis `/` anwenden.)

3. Unter `~kurs059/tmp` befinden sich zwei Textdateien `worte1.txt` und `worte2.txt`. Schauen Sie sich den Befehl `grep` an, um in `worte1.txt` alle Worte zu finden, die die Buchstaben "`ei`" oder "`Ei`"enthalten.[5] Wie findet man (mit nur einem Befehlsaufruf) alle derartigen Worte in *beiden* Dateien.

4. **Für die Harten (optional, war nicht auf den Folien):** die vorige Aufgabe suchte nach dem direkten String `"ei"`. Wie kann man alle Worte finden, die ein `"e"` besitzen und danach irgenwann ein `"ng"`? Wie die Worte, die mit `"e"` anfangen und mit `"ng"` aufhören? Falls Sie das versuchen wollen: das Stichwort sind *reguläre Ausdrücke*

[5] Der Befehl `grep` sucht genauer gesagt nach passenden *Zeile,* nicht Worten, aber in diesem Fall enthalten die Dateien ein Wort pro Zeile.