

# Betriebssysteme im Wintersemester 2008/2009

## Übungsblatt 1

**Abgabetermin:** keine Abgabe erforderlich

**Achtung:** Herzlich willkommen zum Übungsbetrieb zur Vorlesung Betriebssysteme. Bitte melden Sie sich zu den Übungsgruppen unter <http://www.pst.ifi.lmu.de/uniworx> an. Beachten Sie dazu die Hinweise auf dem Merkblatt.

Der Stoff dieses Übungsblattes dient ausschließlich zur Einarbeitung in das Betriebssystem Linux und ist **nicht** Teil des Prüfungstoffes. Das Blatt ist als Tutorium zu verstehen, weswegen auch die Lösungen zu den einzelnen Aufgaben mit angegeben sind. Dies wird auf den folgenden Übungsblättern **nicht** der Fall sein. Zu diesem Blatt werden keine eigenen Übungsstunden angeboten, selbstverständlich können aber in den Übungsstunden Fragen zu den Aufgaben gestellt werden. Der Übungsbetrieb beginnt ab **Donnerstag, 23.10.2008**.

### Aufgabe 1: (Selbststudium) Einstieg in Linux (– Pkt.)

Diese Aufgabe soll Ihnen den Einstieg in den Umgang mit der Linux-Shell erleichtern, falls Sie bisher nur wenig Erfahrung damit gesammelt haben. Da die Aufgabe zum Selbststudium konzipiert wurde, wird sie in der Übungsgruppe nicht im Detail besprochen. Experten können diese Aufgabe ignorieren. ;-)

#### Nach dem Anmelden am CIP-Rechner:

- Sie sehen die grafische Oberfläche KDE.
- Starten Sie eine Konsole (Shell), indem Sie in der Kontrollleiste am unteren Bildschirmrand auf das Icon mit der Muschel klicken.
- Die Konsole zeigt Ihnen an, in welchem Verzeichnis Sie sich gerade befinden (im Normalfall ist das zu Beginn Ihr Home-Verzeichnis) und stellt einen Cursor zur Verfügung, sodass Sie jetzt Befehle eingeben können.

#### Erste Schritte mit der Linux-Shell:

- a. Geben Sie den Befehl `ls` ein (und bestätigen Sie mit Enter). Sie sehen eine Auflistung der Dateien und Ordner in Ihrem Home-Verzeichnis. Selbst wenn das Verzeichnis leer ist, sehen Sie dort immer die Einträge `.` und `..`, bei denen es sich um Links auf das aktuelle Verzeichnis selbst und auf das Oberverzeichnis handelt.
- b. Geben Sie `mkdir neu` ein, um ein Unterverzeichnis mit dem Namen `neu` zu erstellen. Wechseln Sie mit `cd neu` in dieses Verzeichnis und mit `cd ..` von dort aus wieder zurück in Ihr Home-Verzeichnis.

- c. Mit dem Befehl `echo` lassen sich beliebige Zeichenketten auf der Konsole ausgeben. Testen Sie das mit `echo Info3`.
- d. Geben Sie nun `echo Info3 > test.txt` ein. Die Ausgabe der Zeichenkette erfolgt jetzt nicht mehr direkt in der Konsole, sondern wird in eine Datei umgeleitet. Es wird also im aktuellen Verzeichnis eine neue Datei mit dem Namen `test.txt` angelegt, die den Text `Info3` enthält.
- e. Verwenden Sie den Befehl `ls` erneut, um sich zu überzeugen, dass sich die Datei `test.txt` nun tatsächlich in Ihrem Home-Verzeichnis befindet.
- f. Zu nahezu jedem Befehl lassen sich spezielle zusätzliche Parameter angeben, die Einfluss auf die Wirkung des Befehls haben. Solche Parameter werden Flags genannt und beginnen mit einem Minus-Zeichen. Geben Sie folgendes ein: `ls -l`. Der Befehl `ls` wird nun also mit dem Parameter (Flag) `-l` aufgerufen. Wie Sie sehen, wird die Ausgabe wesentlich ausführlicher als zuvor.
- g. Aber welche weiteren Flags gibt es zum `ls`-Befehl? Zu jedem Shell-Befehl lässt sich eine Hilfe-Seite (Manual) anzeigen. Rufen Sie die Hilfe-Seite zum Befehl `ls` auf, indem Sie `man ls` eingeben. Mit den Pfeil-Tasten und der Bild-auf- und Bild-ab-Taste können Sie im Text scrollen. Suchen Sie die Erklärung zum `-l`-Flag. Drücken Sie die Taste `q`, um die Hilfe-Seite wieder auszublenden.
- h. Rufen Sie die Hilfe-Seite zum Befehl `cat` auf und finden Sie heraus, was dieser Befehl macht. Testen Sie mit `cat test.txt`.
- i. Geben Sie folgendes ein: `ls -l >> test.txt`. Anschließend rufen Sie erneut `cat test.txt` auf. Was ist passiert? Worin besteht der Unterschied zwischen `>` und `>>`? (Betrachten Sie dazu vor allem den Anfang der Datei `test.txt`.)
- j. Wie Sie noch im Detail lernen werden, werden in Ausführung befindliche Programme und Hilfsprogramme durch Prozesse abgebildet, die vom Betriebssystem verwaltet werden. In der Shell können Sie sich die aktuell laufenden Prozesse zum Beispiel so anzeigen lassen: `ps -e -f -u`. Finden Sie heraus, wofür die Flags `-e`, `-f` und `-u` stehen (`man ps`).
- k. Verwenden Sie den Befehl `wc` und einen geeigneten Parameter, um die Zeichen in der Datei `test.txt` zu zählen.
- l. Ein sehr wichtiges Konzept (nicht nur von Linux) sind die Umgebungsvariablen. Dabei handelt es sich um Behälter für Daten, die das Betriebssystem gelegentlich benötigt. Sie können sich jede Umgebungsvariable als eine Art Spickzettel des Betriebssystems vorstellen. Geben Sie `env` ein, um sich alle Umgebungsvariablen und deren aktuelle Werte anzeigen zu lassen. Die Darstellung hat das Format `VARIABLENNAME=Wert`.
- m. Lassen Sie sich nur den Wert der Umgebungsvariablen `SHELL` anzeigen, indem Sie `echo $SHELL` eingeben. Sie sehen jetzt zum Beispiel `bin/zsh` (CIP-Pool) oder `bin/bash`, was einfach dem Namen der Shell entspricht, die Sie gerade nutzen. (Ähnlich wie es verschiedene Internet-Browser wie Netscape, Mozilla oder Firefox gibt, gibt es auch verschiedene Shells.)

**Remote-Login von zuhause aus:** Sie können auch von zuhause aus auf die Rechner im CIP-Pool zugreifen, ohne Linux dafür bei sich installieren zu müssen.

- Es gibt (unter Windows) die Möglichkeiten für einen Remote-Login an einem CIP-Rechner, indem Sie einen SSH-Client (z.B. PuTTY) herunterladen und installieren.
- Gehen Sie auf <http://www.rz.ifi.lmu.de/FAQ/Aussenzugriff/faq.html>. Hier finden Sie alle Informationen und Schritt-für-Schritt-Anleitungen für den Remote-Login am CIP-Pool.

**Linux daheim:** Um Linux auf Ihrem PC zu installieren, hier einige Hinweise zum Vorgehen. Beachten Sie: Der Lehrstuhl kann keinen Support leisten und trägt keine Verantwortung für Datenverlust oder andere Schäden. Sichern Sie vor der Installation alle wichtigen Daten! Alle Angaben ohne Gewähr.

- Entscheiden Sie sich zuerst, welche Linux-Distribution Sie in welcher Version verwenden möchten (z.B. Ubuntu Linux in der Version 8.04 bzw. 8.10).
- Erstellen Sie die Installations-Medien. In der Regel brennen Sie dazu eine DVD oder drei bis fünf CD-ROMs. So geht's:
  - Gehen Sie auf die Internet-Seite der entsprechenden Distribution
  - Suchen Sie dort (Download-Bereich) nach den benötigten ISO-Images (für jede DVD bzw. CD müssen Sie genau eine Image-Datei herunterladen).
  - Mit Hilfe der ISO-Dateien können Sie die Installationsmedien leicht brennen. Verwenden Sie dazu Ihr Brennprogramm.
- Booten Sie Ihren Computer von der ersten CD (DVD), um die Linux-Installation zu starten.
- Im Normalfall können Sie ab jetzt die Vorschläge des Installationsassistenten übernehmen.

**Antwort:** Keine Lösung erforderlich.

## Aufgabe 2: (Selbststudium) Einfache Linux-Kommandos (– Pkt.)

- a. Beschreiben Sie kurz Sinn und Zweck der folgenden Kommandos inklusive der angegebenen Flags. Zu einem Befehl `command` erhalten Sie durch Eingabe von `man command` eine Erläuterung. Parameter können beim `man`-Aufruf weggelassen werden.

- (i) `ls -l`
- (ii) `rm -i datei`
- (iii) `mkdir name`
- (iv) `less datei`
- (v) `wc datei`

- b. Finden Sie eine Kategorisierung für die folgenden Befehle mit zwei Kategorien:

`ls, kill, rm, mv, mkdir, killall, cp, fork`

**Antwort:**

- a. (i) `ls`: alle Dateien im aktuellen Ordner auflisten (list directory contents)  
    `-l`: ausführliches Listing-Format verwenden (use long listing format), enthält für jede Datei bzw. jeden Ordner z.B. Lese-/Schreib-Rechte, Benutzer-Rechte, Erstellungsdatum
- (ii) `rm datei`: remove file or directories  
    `- i` interaktiv, jede Löschung muss einzeln bestätigt werden
- (iii) `mkdir name`: Verzeichnis/Ordner erstellen (make directory)
- (iv) `less datei`: zum Anzeigen des Inhalts einer Datei innerhalb der Shell, in der Regel schneller als mit Texteditor. Es gibt auch den Befehl `more`, der aber kein Zurückscrollen innerhalb der Datei ermöglicht. Editieren ist mit beiden Befehlen nicht möglich.
- (v) `wc datei`: Anzahl Bytes, Wörter und Zeilen einer Datei anzeigen
- b. – Dateiverwaltung: `ls, rm, mv, mkdir, cp`  
    – Prozessverwaltung: `kill, killall, fork`

### Aufgabe 3: (Selbststudium) Systemaufrufe

(– Pkt.)

Diese Aufgabe soll Ihr Verständnis für die Bedeutung von Systemaufrufen vertiefen (genauere Informationen zu den Kommandos wie immer in Manpages bzw. Infosystem):

- a. Schätzen Sie, wieviel Systemaufrufe die Kommandos
  - `cat /etc/passwd` bzw.
  - `find / -name "caesar"` bzw.
  - `xemacs` (Starten von XEmacs, Schließen nach dem Start ohne weitere Aktionen)
 erzeugen.
- b. Überprüfen Sie nun ihre Schätzung mit Hilfe von `strace`. `strace` gibt alle Systemaufrufe aus, die ein Programm verwendet. Der Aufruf sollte beispielsweise lauten `strace -c cat /etc/passwd` für das erste Kommando.

**Antwort:** Keine Lösung erforderlich.

### Aufgabe 4: (Selbststudium) Grundlagen Shell-Programmierung (– Pkt.)

Alle Linux-Shells stellen Konstrukte zur Verfügung, um Shell-Skripte zu erstellen. Einfache Shell-Skripte sind nichts anderes als Sequenzen von Befehlen. Zusätzlich stehen Konstrukte für typische imperative Konzepte wie Wertzuweisungen, Schleifen oder Verzweigungen (`if`) zur Verfügung.

**So funktioniert es:**

- Wenn Sie ein beliebiges Skript geschrieben haben, muss es ausführbar gemacht werden. Das funktioniert mit dem Befehl `chmod +x filename` (wobei `filename` der Dateiname des Skripts ist).
- Durch Eingabe von `./filename` wird das Skript gestartet.

**Einige Linux-Konsolenbefehle:**

Befehl	Beschreibung
<code>echo "mein text"</code>	gibt "mein text" auf der Konsole aus
<code>ls</code>	listet alle Dateien im aktuellen Verzeichnis auf
<code>cp quelle ziel</code>	kopieren
<code>mv alt neu</code>	umbenennen bzw. verschieben
<code>rm datei</code>	löschen
<code>grep 'text' datei</code>	sucht in einer Datei nach der Zeichenkette "text"
<code>cat datei</code>	gibt den Inhalt einer Datei auf dem Bildschirm aus
<code>head datei</code>	gibt einige Zeilen vom Dateianfang auf dem Bildschirm aus
<code>tail datei</code>	gibt einige Zeilen vom Dateiende auf dem Bildschirm aus
<code>file datei</code>	gibt aus, von welchem Dateityp eine Datei ist
<code>sort datei</code>	sortiert die Zeilen einer Datei alphabetisch
<code>wc -l datei</code>	zählt die Zeilen in einer Datei
<code>wc -w datei</code>	zählt die Wörter in einer Datei
<code>wc -m datei</code>	zählt die Anzahl der Buchstaben in einer Datei
<code>read var</code>	fordert den Benutzer zu einer Eingabe auf

- a. Schreiben Sie ein Skript, das den Text "Hello World" zunächst in einer Variablen ablegt und dann auf der Konsole ausgibt.

- b. Gegeben ist folgendes Shell-Skript:

```
#!/bin/sh
2 x=Raum
echo "$xschiff"
```

Welches Problem besteht? Wie sieht die Lösung aus?

- c. Schreiben Sie (mit Hilfe von Pipes) ein Shell-Skript, das ausgibt, in wie vielen Zeilen der Datei `file.txt` das Wort "Text" vorkommt.
- d. Schreiben Sie ein Shell-Skript `smart`, das erkennt, ob eine beliebige Datei eine ZIP-komprimierte Datei (Dateityp `.zip`) oder eine Text-Datei (Dateityp `.txt`) ist und diese Datei automatisch entweder korrekt entpackt oder (im Falle einer Text-Datei) im XEmacs-Editor öffnet.
- e. Schreiben Sie ein Shell-Skript `select`, das den Benutzer nach dem Aufruf fragt, ob seine Datei im XEmacs- oder im KWrite-Editor geöffnet werden soll.

**Antwort:**

- a. "Hello World"-Skript:

```
#!/bin/sh
2 a="Hello World"
echo $a
```

- b. Dieses Skript liefert eine leere Ausgabe auf der Konsole. Das ist so, weil das Skript versucht, den Inhalt der Variablen `xschiff` auszulesen, die es gar nicht gibt. Die Nutzung geschweifter Klammern schafft Abhilfe:

```
#!/bin/sh
2 x=raum
echo "${x}schiff"
```

Die Ausgabe ist nun wie gewünscht "Raumschiff".

- c. Die Ausgabe des Befehls `grep` wird zur Eingabe des Befehls `wc -l`:

```
#!/bin/sh
2 grep "Text" file.txt | wc -l
```

- d. Dieses Problem lässt sich mit Hilfe eines Case-Konstrukts lösen:

```
#!/bin/sh
2 ftype=`file "$1"`
case "$ftype" in
4 "$1: Zip archive"*)
    unzip "$1" ;;
6 "$1: ISO-8859 text"*)
    xemacs "$1" ;;
8 *) error "Datei $1 ist weder ZIP- noch Textdatei";;
esac
```

- e. Das `select`-Konstrukt für die Auswahlroutine, eine einfache If-then-else-Abfrage erledigt den Rest:

```
#!/bin/sh
2 echo "Mit welchem Editor möchten Sie die Datei $1 betrachten?"
select var in "XEmacs" "KWrite"; do
4 break
done
6 echo "Auswahl: $var"
if [ "$var" = "XEmacs" ]; then
8 xemacs $1
else
10 kwrite $1
fi
```