Abteilung für INFORMATIK

HTL ST. PÖLTEN

Titel: Linux-Shell

Klasse: 3BHIF

Name: Haiden

Gruppe: 01

Aufgabe: 16.10.2019 Abgabe: 30.10.2019

Inhaltsverzeichnis

1 Grundlagen der Kommandozeile

1.1 Was versteht man unter einer Shell und welche Aufgaben hat sie?

Eine Shell wird verwendet, um mit dem Betriebssystem zu interagieren.

1.2 Nennen Sie Beispiele für gebräuchliche Shells. Welche verwenden Sie? Wo wird das festgelegt?

- Sh – Bourne Shell
- Ksh – K-Shell
- Csh – C-Shell
- Bash – Bourne-Again-Shell
- Zsh – Z-Shell

Ich verwende die Bash-Shell. Dies wird durch das jeweilige Betriebssystem festgelegt. Mit echo $0 kann man sich seine Shell anzeigen lassen.

Beispiel:

```
niklas@DESKTOP-A4VPJ1E:/mnt/c/Users/nikla$ echo $0
-bash
```

1.3 Wo finden Sie die Einstellungen Ihrer Shell?

Die Einstellungen für die einzelnen Benutzer findet man in seinem Home-Verzeichnis unter `/home/benutzer/.bashrc`.

Die Einstellungen, die global für alle Benutzer gelten, findet man im `/etc` Verzeichnis unter `/etc/bash.bashrc`

1.4 Was versteht man unter einem internen bzw. externen Kommando? Wie können Sie herausfinden um welches Kommando es sich handelt?

1.4.1 Extern oder Intern?

Mit dem Befehl `type <KOMMANDO>` kann man sich den Typ des Kommandos anzeigen lassen. Dazu zwei Beispiele:

Externes Kommando:

```
niklas@niklas-virtual-machine:~$ type cat
cat ist /bin/cat
```

Internes Kommando:

```
niklas@niklas-virtual-machine:~$ type cd
cd ist eine von der Shell mitgelieferte Funktion.
```

1.4.2 Internes Kommando

Bei einem internen Kommando wird kein Binärprogramm ausgeführt, sondern ein Kommando welches intern in die Shell eingebaut wird. So könnte man z.B. nicht ohne dem internen Kommando „cd“ den Ordner wechseln, da die Shell ohne internem Suchen über Change Directory die Binärdatei nicht finden würde.

1.4.3 Externes Kommando

Bei einem externen Kommando wird ein Programm ausgeführt, welches in den Verzeichnissen /bin, /sbin oder /usr/bin liegt, wobei nur der Root-User auf das /sbin-Verzeichnis Zugriffsrechte hat. Um der Shell zu sagen, wo sich Binärprogramme befinden, gibt es die sogenannte PATH-Variable, wo diese Ordner mit den Binärprogrammen drinstehen.

2 Hilfe suchen über die Kommandozeile

2.1 In welche Themenbereiche sind die Manpages unterteilt? Warum kann das wichtig sein?

Die Manpages sind nach folgenden Themenbereichen unterteilt:

- Generelle Kommandos
- Systemaufrufe (System Calls)
- Bibliotheksfunktionen
- Spezielle Dateien (z.B. Gerätedateien)
- Dateiformate
- Spiele und Bildschirmschoner
- Verschiedenes
- Kommandos um das System zu administrieren und Daemons

2.2 Wie können Sie einen man-Page anzeigen/durchsuchen/verlassen?

Mit dem Kommando „man“ + „Befehl, den man anzeigen möchte“ kann man eine Man-Page zum jeweiligen Kommando anzeigen lassen.

z.B.: `man ls`

Um eine Manpage zu durchsuchen, drückt man Shift + 7, um ein Slash zu bekommen, bei den man einen String eingeben kann, um danach in der Manpage zu suchen.

Um eine Manpage zu verlassen, muss man einfach die Q-Taste drücken.

2.3 Wo finden Sie weiterführende Informationen zu den Paketen auf Ihrem System?

Falls in den Manpages nichts drin steht, findet man im /usr/share/doc Verzeichnis Dokumentationen zu den Paketen, falls welche mitgeliefert werden.

3 Verzeichnisse verwenden und Dateien auflisten

3.1 Wie unterscheiden sich absolute und relative Pfade?

Absolute Pfade beginnen immer beim Wurzelverzeichnis /, sind also voll ausgeschriebene Pfade, während relative Pfade immer beim aktuellen Verzeichnis, in dem man sich befindet, ausgehen.

3.2 Wofür steht „.“, „..“und ~ bei Verzeichnissen?

„.“ Steht für das aktuelle Verzeichnis, „..“ für das Verzeichnis darüber und „~“ für das Home-Verzeichnis.

3.3 Wie können Sie das aktuelle Verzeichnis auflisten. Erklären Sie die wichtigsten Parameter des Befehls `ls` (a,l,d).

Mit `ls` kann man Verzeichnisse auflisten.

Dabei stehen einem verschiedene Parameter zur Verfügung, hier die wichtigsten:

- Mit `-a` listet man alle Dateien auf, auch versteckte Dateien, die mit einem . beginnen.
- Mit `-l` bekommt man ein Ausgabe, welche mehr Details bereithält
- Mit `-d` bekommt man die Ordner aufgelistet, allerdings nicht die Inhalte

3.4 Auf welche dieser Dateinamen passen die Suchmuster?

Angabe:

6.7 [!1] Im aktuellen Verzeichnis stehen die Dateien

```
prog.c   prog1.c  prog2.c  progabc.c  prog
p.txt    p1.txt   p21.txt  p22.txt    p22.dat
```

Auf welche dieser Dateinamen passen die Suchmuster (a) `prog*.c`, (b) `prog?.c`, (c) `p?*.txt`, (d) `p[12]*`, (e) `p*`, (f) `*.*`?

a) Passt auf alle Dateien, die mit prog beginnen und die Endung .c haben. (prog1.c, prog2.c, progabc.c, prog.c)

```
niklas@niklas-virtual-machine:~$ ls prog*.c
prog1.c  prog2.c  progabc.c  prog.c
```

b) Passt auf alle Dateien, die mit prog beginnen, die Endung .c haben und nach prog nur mehr ein Zeichen stehen haben. (prog1.c, prog2.c)

```
niklas@niklas-virtual-machine:~$ ls prog?.c
prog1.c  prog2.c
```

c) Passt auf alle Dateien die mit p beginnen, 1 Zeichen nach p haben oder beliebig viele Zeichen nach p haben, mit der Endung .txt.

```
niklas@niklas-virtual-machine:~$ ls p?*.txt
p1.txt  p21.txt  p22.txt
```

d) Passt auf alle Dateien, die p heißen und danach eine oder beide von den in Klammern befindlichen Zahlen beinhaltet, mit beliebiger Endung.

```
niklas@niklas-virtual-machine:~$ ls p[12]*
p1.txt  p21.txt  p22.dat  p22.txt
```

e) Alle Dateien, die mit p beginnen und eine beliebige Endung haben.

```
niklas@niklas-virtual-machine:~$ ls p*
p1.txt  p21.txt  p22.dat  p22.txt  prog  prog1.c  prog2.c
progabc.c  prog.c  p.txt
```

f) Listet alle Dateien im Verzeichnis auf, die eine Endung haben (.).

```
niklas@niklas-virtual-machine:~$ ls *.*
p1.txt  p21.txt  p22.dat  p22.txt  prog1.c  prog2.c  progabc.c
prog.c  p.txt
```

3.5 Unterschied zwischen `ls` & `ls *`

Mit `ls` ohne Stern listet man nur die aktuellen Verzeichnisse & Dateien auf, die sich im aktuellen Ordner befinden. `ls *` geht einen Schritt weiter, es listet die Unterverzeichnisse & Dateien der aufgelisteten Verzeichnisse auf.

```
niklas@niklas-virtual-machine:~$ ls
Bilder  Desktop  Dokumente  Downloads  Musik  Öffentlich  Videos
Vorlagen
```

Mit * sieht es so aus:

```
niklas@niklas-virtual-machine:~$ ls *
Bilder:

Desktop:

Dokumente:

Downloads:
lewis-parsons-zGcAG_2D6i8-unsplash.jpg

Musik:

Öffentlich:

Videos:

Vorlagen:
```

Wie man an der Ausgabe sehen kann, wird beim ersten Befehl nur das Download Verzeichnis ohne Inhalt aufgelistet, während beim zweiten Befehl das Bild, welches sich Unterverzeichnis Downloads befindet, auch mitaufgelistet wird.

3.6 Erklären Sie, warum das folgende Kommando zur angezeigten Ausgabe führt

Angabe:

```
niklas@niklas-virtual-machine:~/exercise$ ls
datei1  datei2  datei3  -l
niklas@niklas-virtual-machine:~/exercise$ ls *
-rw-rw-r-- 1 niklas niklas 0 Okt 29 21:08 datei1
-rw-rw-r-- 1 niklas niklas 0 Okt 29 21:08 datei2
-rw-rw-r-- 1 niklas niklas 0 Okt 29 21:08 datei3
```

Wenn `ls *` die Dateien liest, interpretiert es die `-l` Datei als Argument für und zeigt mehr Information als ein normaler ls-Befehl an, da das der Zweck des `-l` Arguments ist.

3.7 Warum ist es sinnvoll, dass „*“ nicht auf Dateinamen im Punkt am Anfang passt?

Da dies versteckte Dateien sind, die der normale Benutzer nicht sieht. Mit `ls -a` kann man das umgehen.

4 Erstellen, Verschieben und Löschen von Dateien

4.1 Welche Funktion hat der Befehl `touch`?

Mit dem `touch` Befehl kann man die Zugriffs- und Änderungs-Zeitstempel von Dateien ändern. Existiert die Datei nicht, wird eine neue, leere Datei erstellt. Touch wird oft für das Erstellen von neuen, leeren Dateien verwendet.

4.2 Wie können Sie ein Verzeichnis anlegen / löschen?

Mit dem Befehl `mkdir` (Make-Directory) kann man sich ein Verzeichnis erstellen. Um ein Verzeichnis zu löschen, kann man den Befehl `rmdir` (Remove-Directory) verwenden. Der Ornder muss dazu allerdings leer sein.

Beispiel (mkdir & rmdir):

```
niklas@niklas-virtual-machine:~/exercise$ mkdir test
niklas@niklas-virtual-machine:~/exercise$ ls
test
niklas@niklas-virtual-machine:~/exercise$ rmdir test
niklas@niklas-virtual-machine:~/exercise$ ls
```

4.3 Welche spezielle Rolle spielt dabei der Befehl `rm`? Beschreiben Sie insbesondere die Parameter `rf`.

4.3.1 Der Befehl `rm`

Der Befehl rm (remove) wird dazu verwendet, um Dateien oder komplette Verzeichnisse zu löschen. Er wird dazu verwendet, um Verzeichnisse zu löschen, die Dateien beinhalten.

Beispiel zur Verwendung:

```
niklas@niklas-virtual-machine:~/exercise$ rm datei1.txt
```

4.3.2 Der Parameter `-rf`

Mit `-r` wird rekursiv gelöscht, das heißt das alle Dateien und Unterordner eines angegeben Verzeichnisses gelöscht werden. Mit `-f` forciert man das Löschen, es gibt keine Nachfrage ob es wirklich gelöscht werden soll.

Beispiel, am Verzeichnis ***test*** mit folgendem Inhalt:

```
niklas@niklas-virtual-machine:~/exercise$ ls test
test1.txt  test2  test2.txt
```

Nun wird es gelöscht:

```
niklas@niklas-virtual-machine:~/exercise$ rm -rf test
niklas@niklas-virtual-machine:~/exercise$ ls -a
.  ..
```

4.4 Wie können Sie eine Datei kopieren? Funktioniert das auch für Verzeichnisse?

Mit dem Befehl `cp <SOURCE> <DEST>` kann man Dateien unter Linux kopieren.

Mit dem Befehl `cp -R <SOURCE> <DEST>/` kann man Verzeichnisse kopieren, wobei das -R für das rekursive Kopieren von Ordnern erforderlich ist und das / um sicherzustellen, dass es ein Ordner und keine Datei ist.

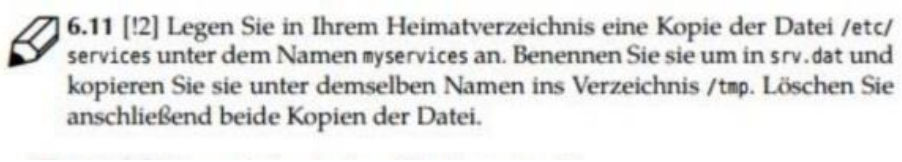
4.5 Wie können Sie eine Datei verschieben / umbenennen?

Mit dem Befehl `mv <SOURCE> <DEST>` kann man Dateien verschieben und auch umbenennen, da sie bei dieser Operation der Inhalt einfach in eine neue Datei mitgeschoben wird.

Mit dem Befehl von oben kann man auch Ordner verschieben, nur ist es wichtig, dass man nach und vor jedem Ordner-Namen ein / anhängt, um zu vermeiden, dass Linux es als Datei erkennt.

4.6 Arbeitsaufgabe zu Anlegen, Kopieren und Verschieben

Angabe:



6.11 [!2] Legen Sie in Ihrem Heimatverzeichnis eine Kopie der Datei `/etc/services` unter dem Namen `myservices` an. Benennen Sie sie um in `srv.dat` und kopieren Sie sie unter demselben Namen ins Verzeichnis `/tmp`. Löschen Sie anschließend beide Kopien der Datei.

Lösung:

```
niklas@niklas-virtual-machine:~$ cp /etc/services /home/niklas/
niklas@niklas-virtual-machine:~$ ls
Bilder  Desktop  Dokumente  Downloads  Musik  Öffentlich  services
test  Videos  Vorlagen
niklas@niklas-virtual-machine:~$ mv services srv.dat
niklas@niklas-virtual-machine:~$ ls
Bilder  Desktop  Dokumente  Downloads  Musik  Öffentlich  srv.dat
test  Videos  Vorlagen
niklas@niklas-virtual-machine:~$ cp srv.dat /tmp/
niklas@niklas-virtual-machine:~$ ls /tmp
config-err-7n3IEg
srv.dat
ssh-QfjbeA3WDn7K
ssh-VSapF6qOTdyx
systemd-private-80a54ae137ea429ea15d144db3a5dfa0-redis-
server.service-Cb2Jao
systemd-private-80a54ae137ea429ea15d144db3a5dfa0-systemd-
resolved.service-JnLRKg
systemd-private-80a54ae137ea429ea15d144db3a5dfa0-systemd-
timesyncd.service-CcNYNW
VMwareDnD
vmware-root_642-2730628029
niklas@niklas-virtual-machine:~$ rm -rf srv.dat
niklas@niklas-virtual-machine:~$ rm -rf /tmp/srv.dat
```

```
niklas@niklas-virtual-machine:~$ ls
Bilder  Desktop  Dokumente  Downloads  Musik  Öffentlich  test
Videos  Vorlagen
niklas@niklas-virtual-machine:~$ ls /tmp
config-err-7n3IEg  systemd-private-80a54ae137ea429ea15d144db3a5dfa0-
redis-server.service-Cb2Jao        VMwareDnD
ssh-QfjbeA3WDn7K   systemd-private-80a54ae137ea429ea15d144db3a5dfa0-
systemd-resolved.service-JnLRKg    vmware-root_642-2730628029
ssh-VSapF6qOTdyx   systemd-private-80a54ae137ea429ea15d144db3a5dfa0-
systemd-timesyncd.service-CcNYNW
```

4.7 Warum hat `mv` keine `-R`-Option wie `cp`?

Wenn man z.B. `mv a b` macht, wird automatisch jede Datei im Ordner danach umbenannt. Man braucht keine rekursive Option, da mv sie schon mitbringt und intern einsetzt.