Linux Befehlsbeispiele

Rupert Wenzel
mail@ruwela.de

25. Oktober 2002

Achtung allen folgenden Befehlsbeispielen sind mit groesster Vorsicht zu geniessen. Es handelt es sich um Notizen fuer mich, die selbstverstaendlich Fehler enthalten koennen und diese wiederum zu fatalen Schaeden an System und Daten fuehren koennen. Dafuer kann ich natuerlich keinerlei Gewaehr uebernehmen. Es sind Beispiele zu “gefaehrlichen” Befehlen wie *dd* enthalten, die natuerlich vor Anwendung genauestens geprueft und an die gegebenen Verhaeltnisse angepasst werden muessen. Ich bin mir auch sicher das es fuer viele Beispiele bessere Loesungen gibt. Wer Fehler findet oder Verbesserungen kennt kann mir gerne eine Mail schicken.

1 Online-Hilfe, Man-pages

Man-page zu dem Befehl *ls* anzeigen:

```
man ls
```

Man-page zu dem Befehl *ls* als Textdatei *~/ls.man* ablegen:

```
man ls | rman > ~/ls.man
```

oder direkt drucken:

```
man ls | rman | lpr
```

Man-page zu dem Befehl *ls* als PostSkript- oder PDF-Datei ablegen:

```
man -T ls > man-ls.ps
man -T ls | ps2pdf - > man-ls.pdf
```

Man-page zu dem Befehl *ls* als HTML-Datei *~/ls.html* ablegen:

```
man ls | rman -f html > ~/ls.html
```

Infopage zu dem Befehl *ls* anzeigen:

```
info ls
```

2 Suchen

Finde die Datei *fstab* (alle Unterverzeichnisse ab dem aktuellen werden durchsucht):

```
find . -name fstab
```

Finde alle Dateien (ab dem Wurzelverzeichnis /) die neuer als */etc/fstab sind*:

```
find / -newer /etc/fstab
```

Finde ab dem aktuellen Verzeichnis alle Dateien die groesser als 10 KByte gross sind:

```
find . -size +10k
```

oder:

```
ls -lS $(find . -type f) | awk ' { if ($5 > 10240)  print}'
```

Finde ab dem aktuellen Verzeichnis alle Dateien die groesser als 500 Byte gross sind:

```
find . -size +500c
```

oder:

```
ls -lS $(find . -type f) | awk ' { if ($5 > 500)  print}'
```

Finde alle Dateien deren Rechte auf *SETUID* gesetzt sind:

```
find / -perm +4000
```

Nach dem Wort *alias* in der Dateien */etc/profile* suchen:

```
grep alias /etc/profile
```

Alle Zeilen der Datei */etc/profile* anzeigen die nicht mit dem Kommentarzeichen # beginnen:

```
cat /etc/profile | grep -v ^#
```

Wie zuvor jedoch werden alle gefundenen Zeilen in den Editor *vi* geladen und koennen anschliessend unter einem anderen Namen gespeichert werden:

```
cat /etc/profile | grep -v ^# | vi -
```

Finde ab dem aktuellen Verzeichnis alle Datei- und Pfad-Namen, in denen *emacs* vorkommt (keine Unterscheidung zwischen Groß- und Kleinschreibung durch den Parameter -i).

```
find . 2> /dev/null | grep -i emacs
```

Alle Dateien, ab dem aktuellen Verzeichnis, mit der Endung *.txt* werden nach der Zeichenkette “Suchmuster” durchsucht. Falls es zu viele Dateien sind mit einer Schleife suchen (siehe nächstes Beispiel):

```
grep Suchmuster $(find . -name *.txt) 2> /dev/null
```

Finde ab dem aktuellen Verzeichnis alle Dateien, in denen der Suchstring *emacs* vorkommt (keine Unterscheidung zwischen Groß- und Kleinschreibung durch den Parameter -i).

```
for i in $(find)
do
   grep -n  xemacs $i 2> /dev/null && echo -e "Enthalten in $i \n"
done
```

Loesche alle Dateien mit der Endung .tmp ACHTUNG Dateien mit der Endung *TMP* werden in diesem Beispiel nicht gefunden:

```
find / -name "*.tmp" -exec rm "{}" \;
```

Finde ab dem aktuellen Verzeichnis alle Dateien mit der Endung .txt und kopiere sie ins */verzeichnis/text/* ACHTUNG Dateien mit der Endung *TXT* werden in diesem Beispiel nicht gefunden:

```
find -name "*.txt" -exec cp "{}" /verzeichnis/text \;
```

3 Vergleich von Dateien und Verzeichnissen

Vergleich zweier Dateien auf Unterschiede:

```
diff datei1 datei2
diff -d datei1 datei2
```
Anderer Algorithmus

Vergleich zweier Verzeichnisse auf Unterschiede (Auch Unterverzeichnisse):

```
diff -r verzeichnis1 verzeichnis2
```

Vergleich zweier Archive auf Unterschiede:

```
diff <(tar tzf Buch1.tar.gz) <(tar tzf Buch.tar.gz)
```

4 Archivieren und Komprimieren

Die Datei *Brief* komprimieren. Achtung Die Datei *Brief* wird durch die komprimierte *Brief.Z* ersetzt. Der Parameter -v gibt Infos aus:

```
compress -v Brief
```

Die komprimierte Datei *Brief.Z* dekomprimieren:

```
uncompress Brief.Z
```

oder:

```
compress -d Brief.Z
```

Alle Dateien im Verzeichnis *Briefe* und deren Unterverzeichnisse komprimieren. :

```
compress -r Briefe
```

Alle **.tex*-Dateien des aktuellen Verzeichnisses komprimieren:

`gzip *.tex` Dateien: mit *gzip* komprimiert es entstehen lauter **.tex.gz*-Dateien

`bzip2 *.tex` Dateien: mit *bzip2* komprimiert es entstehen lauter **.tex.bz2*-Dateien

Zum Dekomprimieren wird der Befehl gunzip oder bunzip2 verwendet.

`gunzip *.gz` Alle *.gz* Dateien mit *gunzip* dekomprimiert

`bunzip2 *.bz2` Alle *.bz2* Dateien mit *bunzip2* dekomprimiert

Die Datei *bild.bmp* komprimieren. Die Orginaldatei bleibt unverändert und es entsteht eine neue komprimierte Datei z.B.: *bild.bmp.gz* :

`gzip -c bild.bmp > bild.bmp.gz` Datei: mit *gzip* komprimiert

`bzip2 -c bild.bmp > bild.bmp.bz2` Datei: mit *bzip2* komprimiert

Zum Dekomprimieren wird der Befehl gunzip oder bunzip2 verwendet.

Datei *bild.bmp.gz* bzw.*bild.bmp.bz2* im aktuellen Verzeichnis dekomprimieren:

`gzip -d bild.bmp.gz` Datei wurde mit *gzip* komprimiert

`bzip2 -d bild.bmp.bz2` Datei wurde mit *bzip2* komprimiert

Es kann ebensogut gunzip bzw. bunzip2 zum dekomprimieren verwendet werden.

`gunzip bild.bmp.gz` Datei wurde mit *gzip* dekomprimiert

`bunzip2 bild.bmp.bz2` Datei wurde mit *bzip2* dekomprimiert

Die Dateien AlteDatei.txt.Z (compress) und AlteDatei.txt.z (alte gzip Version) entpacken

`gzip -d AlteDatei.txt.Z` Datei wurde mit *compress* komprimiert

`gzip -d AlteDatei.txt.z` Datei wurde mit einer alten *gzip* Version komprimiert

Archiviert das Verzeichnis *./texte/buch* mit alle in ihm enthaltenen Dateien und Unterverzeichnissen (Unkomprimiert, mit *gzip* bzw. *bzip2* komprimiert). Die Archive werden in *./buch.tar*, *./buch.tgz* und *./buch.tar.bz2* erstellt. Der Parameter *vv* erstellen eine ausführliche Liste am Bildschirm (nur ein *v* würde eine einfache Liste erstellen):

`tar -cvvf buch.tar texte/buch`	Unkomprimiert
`tar -cvvzf buch.tgz texte/buch`	mit *gzip* komprimiert
`tar -cvvIf buch.tar.bz2 texte/buch`	mit *bzip2* komprimiert

Zeigt den Inhalt der zuvor erstellten Archive an. Mit dem Parameter v wird eine Ausführliche Liste angezeigt (Rechte, Zeitstempel, Links, Eigentümer . . .)

`tar -tvf buch.tar`	unkomprimiert
`tar -tvzf buch.tgz`	mit *gzip* komprimiert
`tar -tvIf buch.tar.bz2`	mit *bzip2* komprimiert

Die zuvor erstellten Archiv in das aktuelle Verzeichnis entpacken. Es wird Das Verzeichnis *texte* und das Unterverzeichnis *buch* mit allen im Archiv befindenden Dateien und Unterverzeichnissen angelegt.

`tar -xf buch.tar`	unkomprimiert
`tar -xzf buch.tgz`	mit *gzip* komprimiert
`tar -xIf buch.tar.bz2`	mit *bzip2* komprimiert

Archiviert und komprimiert das Verzeichnis */usr/src* mit allen Dateien und Unterverzeichnissen nach */backup/*. Die Pfade werden ab *./src* im Archiv gespeichert:

`(cd /usr ; tar -c src) > /backup/src.tar`	unkomprimiert
`(cd /usr ; tar -cz src) > /backup/src.tgz`	mit *gzip* komprimiert
`(cd /usr ; tar -cI src) > /backup/src.tar.bz2`	mit *bzip2* komprimiert

Die zuvor erstellten Archive in das aktuelle Verzeichnis entpacken. Es wird das Verzeichnis src angelegt.

`tar -xf src.tar`	unkomprimiert
`tar -xzf src.tgz`	mit *gzip* komprimiert
`tar -xIf src.tar.bz2`	mit *bzip2* komprimiert

Aus dem zuvor erstellten Archiv *src.tar* die Datei *src/linux/Makefile* entpacken. Es wird das Verzeichnis *src/linux/* erstellen.

```
tar -xvf src.tar src/linux/Makefile
```

Archiviert das Verzeichnis *Bilder* verteilt auf mehrere Disketten (Parameter *M*). Auf den Disketten ist kein Dateisystem noetig, die Daten werden roh auf den Datentraeger geschrieben. Leider kann tar bei Multi-Volume-Archiven nicht komprimieren.

```
tar -cvvMf /dev/fd0 Bilder
```

Mit folgendem Befehl wird das Archiv wieder zurueckgeschrieben:

```
tar -xvvMf /dev/fd0
```

Archiviert das Verzeichnis *Videos* verteilt auf mehrere Dateien zu je 640 MB, die spaeter auf CD gebrannt werden koennen. Die so erzeugten Dateien heissen Video-Backup.aa, Video-Backup.ab usw.

```
tar -cvv Videos | split -b 640m - Video-Backup.
```

Mit folgendem Befehl wird aus den einzelnen Dateien das Archiv wieder zurueckgeschrieben:

```
cat Video-Backup.* | tar -xvv
```

Alle Dateien aus */home/user1* nach */home/user2* kopieren. Symbolische Links werden als solche kopiert (und nicht die Dateien, auf die die Links verweisen). Mit diesem Kommando können ganze Dateibäume von einer Partition auf eine andere übertragen werden:

```
(cd /home/user1 ; tar -cf - .) | (cd /home/user2 ; tar -xf -)
```

Um ein Linux-System mit tar zu sichern muss der Rechner mit einem anderen Linux-System hochgefahren werden (Bootdisk, Rettungscd ...). Anschliessend alle System-Partitionen, die gesichert werden sollen, mounten z.B.:

```
mount /dev/hdb1 /mnt/orginal/wurzel
mount /dev/hdb2 /mnt/orginal/usr
```

und auch die Backuppartition mounten:

```
mount /dev/hdc1 /mnt/Backup
```

anschliessend in das Backupverzeichnis wechseln und die Partitionen mit tar sichern.

```
 cd /mnt/Backup/
(cd /mnt/orginal/wurzel ; tar -c .) > Wurzel-Partition.tar
(cd /mnt/orginal/usr ; tar -c .) > usr-Partition.tar
```

Das Verzeichnis */home/user* in mehrere Einzeldateien mit maximal 1430 Kb (Dateien passen auf Diskette) archivieren. Die Dateien mit den Namen: *xaa, xab, xac … xyy xyz xzaaa xzaab …* werden im aktuellen Verzeichnis erstellt. Die Pfade werden ab dem Wurzelverzeichnis / im Archiv gespeichert.
ACHTUNG in dem Aktuellen Verzeichnis sollten sich keine Dateien befinden die mit *x* beginnen:

```
(cd / ; tar -czf - home/user ) | split -b 1430k
```

Die nun erstellten Dateien können mit

```
cat x??* > Backup-komplett.tgz
```

wieder zu einem kompletten Archiv zusammengefaßt werden und mit

```
tar -xzf Backup-komplett.tgz -C /
```

wieder an ihre ursprüngliche Stelle entpackt werden. Oder beide Schritte in einem:

```
cat x??* | tar -xz -C /
```

Um das Archiv im aktuellen Verzeichnis (die Verzeichnisse *./home/user* werden erstellt) zu entpacken den Parameter *-C /* weglassen.

5 Dateisysteme

Anzeigen des Dateisystem Superblock Inhaltes von */dev/hda1* :

```
tune2fs -l /dev/hda1
```

Die ext2-Partition */dev/hda1* ohne Datenverlust in eine ext3-Partition umwandeln. Wenn die Partition gemountet ist wird dabei die Datei *.journal* angelegt; wenn nicht, taucht das Jornal nicht im Dateisystem auf. Anschliessend eventuell den Dateisystemtyp in der Datei /etc/fstab von ext2 auf ext3 aendern.

```
tune2fs -j /dev/hda1
```

Testen ob Datenträger (hier Diskette) defekte Sektoren enthält:

```
badblocks /dev/fd0
```

Diskette (1,44 MB 3.5“) Low-Level formatieren:

```
fdformat /dev/fd0h1440
```

oder (Systemabhaengig)

```
fdformat /dev/fd0u1440
```

Diskette (1,44 MB 3.5“) ueberformatieren. Mit den Device */dev/fd0u1600*, */dev/fd0u1760* ... kann eine Diskette auf ein groesseres Format gebracht werden.

```
fdformat /dev/fd0u1760
```

Dateisystem erstellen (z.B. auf Diskette):

`mkfs /dev/fd0`	Dateisystem: ext2
`mkfs.minix /dev/fd0`	Dateisystem: minix
`mkfs.msdos /dev/fd0`	Dateisystem: msdos

Testen ob ext2fs Dateisystem der Partition */dev/hda2* Fehler enthält. Die Meldung *non-contiguous* besagt daß das Dateisystem fragmentiert ist:

```
e2fsck -f -c /dev/hda2
```

Dateisystem in einer Datei erstellen und mounten:

`dd if=/dev/zero of=Datei bs=1M count=100`	100 Mb Datei erstellen
`mkfs Datei`	Dateisystem ext2 erstellen
`mount -o loop Datei /mnt`	

6 Mounten von Dateisystemen, Dateien und Netz-Resourcen

Diskette mounten (1. Diskette unter DOS a:):

```
mount /dev/fd0 /mnt/floppy
```

Festplattenpartition mounten (1. IDE-Platte 1. Partition):

```
mount /dev/hda0 /mnt/windows
```

Mit Hilfe des NFS-Protokolls (Network File System) ein Verzeichnis von einem anderen Rechner mounten (z.B. PC2) Das Verzeichnis muß auf pc2 in */etc/exports* freigegeben sein:

```
mount -t nfs pc2:/ /mnt/pc2
```

oder:

```
mount -t nfs 192.168.17.2:/ /mnt/pc2
```

Image-Datei mounten (z.B. Diskettenimage):

```
mount -o loop /floppyimage /mnt/floppy
```

Suchen wer oder was auf ein bestimmtes Device zugreift. Wenn ein Task (z.B. eine offene Shell eines Users) auf ein Device zugreift, kann dieses nicht mit *umount* vom Dateibaum demontiert werden, es wird die Fehlermeldung “device is busy” ausgegeben. Beispiel: Suchen wer auf das Device */dev/fd0* (Diskettenlaufwerk) zugreift:

```
lsof | grep /dev/fd0
```

Nun kann man die entsprechende Anwendung schliessen oder ggf. killen (kill oder killall).

7 Low-Level Kopieren

Imagedatei von 2. Festplatte (komplett alle Partitionen und MBR) in die Datei */hd2.img* schreiben:

```
dd if=/dev/hdb of=/hd2.img
```

Masterbootrecord in die Datei */mbr.img* sichern (erste IDE-Platte):

```
dd if=/dev/hda of=/mbr.img bs=512 count=1
```

Masterbootrecord auf erste IDE-Platte zurückschreiben:

```
dd if=/mbr.img of=/dev/hda
```

Diskettenimage erstellen (Image kann auch mit *rawrite.exe* unter Dos/Windows wieder auf eine Diskette geschrieben werden):

```
dd if=/dev/fd0 of=/diskette.img
```

Diskette von Diskettenimage erstellen (kann auch mit dem Dos-Tool *rawrite.exe* erstellt werden):

```
dd if=/diskette.img of=/dev/fd0
```

Diskette in zwei Dateien (mit max 700000 Byte) aufteilen, es entstehen im aktuellen Verzeichnis die beiden Dateien *xaa* und *xab*:

```
split -b 700000 /dev/fd0
```

Aus den zwei zuvor erstellten Dateien eine neue Diskette schreiben:

```
cat xaa xab> /dev/fd0
```

Falls sich im aktuellen Verzeichnis keine weiteren Dateien befinden die mit *x* beginnen kann auch der folgende Befehl verwendet werden:

```
cat x*> /dev/fd0
```

Mit *dd* aus den zwei zuvor erstellten Dateien eine neue Diskette schreiben (Selbe funktion wie vorhergehender Befehl):

```
dd if=xaa of=/dev/fd0
dd if=xab of=/dev/fd0 bs=1 seek=700000
```

Partition Low-Level in die Datei *hda3.img* kopieren und komprimieren:

```
dd if=/dev/hda3 bs=1k | gzip -v9 > hda3.img
```

Die Partition hda3 mit dem zuvor erstellten Image ueberschreiben:

```
dd if=hda3.img | gunzip | dd of=/dev/hda3
```

Festplatten-Image erstellen:

```
dd if=/dev/hda1 | gzip | \
cdrecord -v -dummy dev=/dev/sg0 speed=4 -data -
```

schreibt Festplattenpartition auf die CD und umgeht das Dateisystem, also kein ISO9660 für single Session. Natürlich muß -dummy entfernt werden. cdrecord liest von stdin '-' diese CD kann man nicht mounten aber mit:

```
dd if=/dev/hdc | gzip -d > /dev/hda1
```

wieder zurückschreiben.

8 Crontab Beispiele

Die crontab wird dazu verwendet Befehle zu bestimmten Zeiten auszufuehren. Eintraege werden mit dem folgenden Befehl gemacht:

```
crontab -e
```

Nun wird die Konfigurationsdatei automatisch mit vi geladen und man kann Eintraege vornehmen. Die Eintraege haben folgendes Format:

```
Minute Stunde Tag Monat Wochentag [User-ID] Befehl
```

Beispiele:

Um 22.00 Uhr */usr/bin/befehl* starten:

```
0 22 * * * /usr/bin/befehl
```

Jede volle und jede halbe Stunde */usr/bin/befehl* starten:

```
0,30 * * * * /usr/bin/befehl
```

Alle 5 Minuten */usr/bin/befehl* starten:

```
*/5 * * * * /usr/bin/befehl
```

Jeden 15. des Monats um 00:00 Uhr den Rechner neu starten:

```
0 0 15 * * reboot
```

Jeden Montag um 08:15 Uhr */usr/bin/befehl* starten:

```
15 8 * * 1 /usr/bin/befehl
```

Montag bis Freitag um 08:15 Uhr */usr/bin/befehl* starten:

```
15 8 * * 1-5 /usr/bin/befehl
```

Jeden 24. Dezember um 23:00 Uhr */usr/bin/befehl* starten:

```
0 23 24 12 * /usr/bin/befehl
```

Montag bis Freitag, von 8 bis 17 Uhr alle 15 Minuten *befehl* starten:

```
0,15,30,45 8-17 * * 1-5 befehl
```

Samstags und Sonntags zwischen 10 und 17 Uhr alle 2 Stunden *befehl* unter der Benutzer-ID von *rup* starten:

```
0 10-17/2 * * 6,7 rup befehl
```

Infos zu crontab:
www.linuxfibel.de/time.htm
Debian GNU/Linux von Peter H. Ganten ab Seite 146

9 Datenschutz

9.1 Kryptographie mit GnuPG

Erzeugen eines primaeres Schluesselpaares, das aus einem geheimen und einem oeffentlichen Schluessel besteht:

```
gpg --gen-key
```

Nun muss die Frage nach der Art des Schluessels beantwortet werden. Hier kann der Defaultwert 1 mit **Enter** uebernommen werden, um 2 Schluessel (privater und oeffentlicher) zu erzeugen. Die Frage nach der Schluessellaenge kann mit dem Defaultwert 1024 beantwortet werden (**Enter**).
Die Frage nach dem Verfallsdatum des Schluessels kann auch mit **Enter** beantwortet werden (Schlüssel verfällt nie). Um eine eindeutige Benutzer-ID zu erzeugen, wir als naechstes nach Name und Vorname gefragt, anschliessend nach einem Komentar und der e-mail Adresse.
Zum Schluss muss noch eine Mantra (einen Paßwort) eingegeben werden.
Die hiermit erzeugten Schluessel sind im Verzeichniss *~/.gnupg/*

Editieren eines Schluessels zum Beispiel um dem eigenen weitere User-ID zuzufuegen, den Fingerabdruck eines fremden Schluessels zu pruefen, ...

```
gpg --edit-key ruwela@web.de
```

Exportieren des **geheimen** Schluessels in eine Textdatei mit ASCII Werten. Achtung diese darf keinesfalls weitergegeben werden. Eventuell ausdrucken und und an einem sicheren Ort verwahren (Bankschliessfach).

```
gpg --export-secret-keys --armor ruwela@web.de > Geheim.txt
```

Exportieren des Oeffentlichen Schluessels in eine Textdatei mit ASCII Werten. Diese kann per e-mail, ueber eine Webseite oder ueber einen Key-Server (z.B www.keyserver.net) ausgetauscht werden.

```
gpg --export --armor ruwela@web.de > Schluessel.txt
```

Importieren eines oeffentlichen Schluessels:

```
gpg --import Schluessel.txt
```

Auflisten aller oeffentlichen Schluessel an ihrem “Schluesselbund” (auch Schluessel von Komunikationspartnern die importiert wurden):

```
gpg --list-keys
```

Die Textdatei *text.txt* verschluesseln:

```
gpg  --encrypt text.txt
```

GnuPG fraegt nun nach der User-ID, das ist der eindeutige Name, Komentar oder am besten die e-mail Adresse des Empfaengers. Falls man das Dokument fuer sich selbst verschluesselt, einfach die eigene e-mail Adresse angeben. Nun wird die verschluesselte Datei *text.txt.gpg* erstellt.

Die fuer sich bestimmte Datei *text.txt.gpg* entschluesseln:

```
gpg --decrypt text.txt.gpg > text.txt
```

Das Verzeichnis *Briefe* mit tar packen und mit gpg fuer den Benutzer mit der e-mail Adresse mail@ruwela.de verschluesseln:

```
tar -c Briefe | gpg -e -r mail@ruwela.de > Briefe.tar.gpg
```

Das zuvor erstellte und verschluesselte Paket wieder entpacken:

```
gpg -d Briefe.tar.gpg |tar -x
```

Die Textdatei *text.txt* Signieren:

```
gpg  gpg --clearsign text.txt
```

GnuPG fraegt nun nach der Mantra (Passwort) und erstellt anschliessend die Datei *text.txt.asc*. Dieser Datei ist eine verschluessete Signatuer angehaengt.

Die Signatur der Textdatei *text.txt.asc* ueberpruefen. Um eine Signatur zu uebepruefen benoetigt man den oeffentlichen Schluessel des Absenders.

```
gpg  gpg --verify text.txt.asc
```

GnuPG zeigt nun Absender, Erstelldatum und Schluessel-ID an.

Umfangreiche (auch deutsche) Dokumentation zu GnuPG findet man auf folgenden Seiten:
www.gnupg.org
www.gnupg.org/docs.html
www.gnupg.org/faq.html
www.linux-magazin.de/ausgabe/1999/12/GnuPG/gnupg.html
Keyserver zum Verteilen der oeffentlichen Schluessel:
www.keyserver.net

9.2 Steganographie

Das Programm *steghide* kann eine Datei (z.B. eine Textdatei) in einer anderen Datei (Bild *.bmp oder Sound *.wav) verstecken.

Die Datei *geheim.txt* in einem Bitmap verstecken. Die Datei *orginal.bmp* bleibt unveraendert, es wird das neues Bitmap *stegano.bmp* erstellt, das die geheime Textdatei enthaelt. Nach absetzen des Befehls wird ein Passwort verlangt, das zum extrahieren wieder benoetigt wird.

```
steghide embed -pf geheim.txt -cf orginal.bmp -sf stegano.bmp
```

Die Datei *geheim.txt* aus dem zuvor erstellten Bitmap *orginal.bmp* extrahieren:

```
steghide extract -sf stegano.bmp
```

10 TEX – LATEX

Die LATEX–Datei Beispiel.tex ins DVI-Format umwandeln (Beispiel.dvi):

```
latex Beispiel          Endung .tex muß nicht angegeben werden
```

Die LATEX–Datei Beispiel.tex ins HTML-Format umwandeln. Es wird das Verzeichnis Beispiel erstellt, darin befinden sich die html-Seiten und alle dazugehörigen Dateien. 1. Seite ist index.html:

```
latex2html Beispiel     Endung .tex muß nicht angegeben werden
```

Die LATEX–Datei Beispiel.tex ins PDF-Format (Adobe) umwandeln (Beispiel.pdf):

```
pdflatex Beispiel       Endung .tex muß nicht angegeben werden
```

Die DVI-Datei Beispiel.dvi ins PostScript-Format umwandeln (Beispiel.ps)

```
dvips Beispiel          Endung .tex muß nicht angegeben werden
```

Die PostScript-Datei Beispiel.ps ins PDF-Format umwandeln (Beispiel.pdf)

```
ps2pdf Beispiel.ps
```

Die PostScript-Datei Beispiel.ps in ein Bild umwandeln (Beispiel.png)

```
pstoimg Beispiel.ps
```

Alle LaTeX–Dateien im Verzeichnis in folgende Formate umwandeln: DVI, PostScript, PDF und HTML:

```
for i in $(ls *.tex)
do
   latex $i
   latex2html $i
   pdflatex $i
   dvips ${i%.tex}
done
```

Ein neues Paket (z.B. die Klasse akletter) in den bestehenden TeX-Baum installieren. Dieses Pakete gibt es z.B. auf dem Dante Server unter *ftp.dante.de/tex-archive/macros/latex/contrib/supported/*. Das Unterverzeichnis akletter wird nun auf die lokale Festplatte in das Verzeichnis */usr/share/texmf/tex/latex/* kopiert. Anschliessend muss die TeX-Datenbank aktualisiert werden. Dies geschieht unter der Benutzerkennung *root* mit folgendem Befehl:

```
mktexlsr
```

Umfangreiche (auch deutsche) Dokumentation zu TeX und LaTeX findet man auf folgenden Seiten:
www.dante.de/
www.latex-project.org

11 Debian Paketverwaltung

Die Liste der verfuegbaren Pakete aktuallisieren. Die Paketdatenbank wird mit den Paketquellen, die in der Datei */etc/apt/sources.list* definiert sind, abgeglichen

```
apt-get update
```

Paket suchen, alle Pakete anzeigen in denen die Zeichenkette *"webmin"* vorkommt. Das Paket muss nicht installiert sein, es reicht aus, wenn es bei den Paketquellen verfuegbar ist.

```
apt-cache search webmin
```

Informationen zu dem Paket *webmin* anzeigen lassen. Das Paket muss nicht installiert sein, es reicht aus, wenn es bei den Paketquellen verfuegbar ist.

```
apt-cache show webmin
```

Das Paket *webmin* installieren. Es werden alle benoetigten und abhaengigen Pakete mitinstalliert.

```
apt-get install webmin
```

Die Sourcen zu *webmin* ins aktuelle Verzeichnis einspielen:

```
apt-get source webmin
```

Das Paket *webmin* loeschen. Es werden auch abhaengigen Pakete entfernt.

```
apt-get remove webmin
```

Das Paket *realplayer* als RPM-Paket installieren. Zuerst wird aus dem RPM-Datei ein DEB-Archiv erstellt und dieses anschliessend installiert.

```
alien realplayer.rpm
dpkg --install realplayer.deb
```

Um bei einer erneuten Debian-Installation, z.B. auf einem anderen Rechner, die aktuelle Paketauswahl zu uebernehmen, kann diese in eine Datei gespeichert werden, die sich auf einem anderen System wieder einlesen laesst.
Die aktuelle Paketauswahl in eine Datei speichern:

```
dpkg --get-selections '*' > /pfad/PaketListe.txt
```

Paketauswahl anhand der zuvor erstellten Liste anpassen:

```
dpkg --set-selections < /pfad/PaketListe.txt
```

Nun kann mit *dselect* ueber den Punkt *Install* die Paketauswahl angepasst werden.

System aktuallisieren. Es werden alle Pakete von denen die Paketquelle eine neuere Versionen enthaelt aktuallisiert.

```
apt-get upgrade
```

System aktuallisieren, auf eine neue Version wechseln, z.B. von Debian 2.2 Potato zu Debian 3.0 Woody. Es werden alle Pakete von denen die Paketquelle eine neuere Versionen enthaelt aktuallisiert, abhaengige Paktete werden eingespiel, nicht mehr benoetigte Pakete werden geloescht.

```
apt-get dist-upgrade
```

Alle installierten Pakete anzeigen

```
dpkg -l | grep "ii"
```

Deb-Pakete die sich in einem lokalen Verzeichnis befinden in die /etc/apt/sources.list mit einbinden. Dazu muss in dem Verzeichnis in dem sich die Pakete befinden die Datei mit den Paket-Informationen erstellt werden:

```
cd /verzeichnis/mit/deb/paketen/
dpkg-scanpackages ./ /dev/null | gzip > Packages.gz
```

Nun kann das Verzeichnis mit folgendem Eintrag in die Datei /etc/apt/sources.list aufgenommen werden:

```
deb file:/verzeichnis/mit/deb/paketen/ ./
```

Nach einem `apt-get update`
sind die Pakete verfuegbar.

Das Konfigurationsskript zu dem Paket *ssh* ausfuehren. Die Angabe des Parameters -plow bewirkt dass alle Abfragen von Hand eingegeben werden muessen. Durch die Parameter -plow, -pmedium , -high oder -pcritical kann der Fragen-Level festgelegt werden. Der Parameter -plow bewirkt dass alle Abfragen von Hand eingegeben werden muessen, -pcritical wuerde nur noch sehr wenige Fragen stellen und meistens default-Werte setzen.

```
dpkg-reconfigure -plow ssh
```

Frontends zu dpkg und apt:

```
dselect
tasksel
console-apt
aptitude
gnome-apt
synaptic
```

Beispieleintraege fuer */etc/apt/sources.list*:

```
### Debian Woody (stable):
#deb         ftp://ftp.de.debian.org/debian          woody         main contrib non-free
#deb         ftp://ftp.de.debian.org/debian-non-US   woody/non-US  main contrib non-free
#deb-src     ftp://ftp.de.debian.org/debian          woody         main contrib non-free
#deb-src     ftp://ftp.de.debian.org/debian-non-US   woody/non-US  main contrib non-free


### Debian Sarge:
deb          ftp://ftp.de.debian.org/debian          sarge          main contrib non-free
deb          ftp://ftp.de.debian.org/debian-non-US   sarge/non-US   main contrib non-free
deb-src      ftp://ftp.de.debian.org/debian          sarge          main contrib non-free
deb-src      ftp://ftp.de.debian.org/debian-non-US   sarge/non-US   main contrib non-free


### Debian Sid:
#deb         ftp://ftp.de.debian.org/debian          sid            main contrib non-free
#deb         ftp://ftp.de.debian.org/debian-non-US   sid/non-US     main contrib non-free
#deb-src     ftp://ftp.de.debian.org/debian          sid            main contrib non-free
#deb-src     ftp://ftp.de.debian.org/debian-non-US   sid/non-US     main contrib non-free


### Secuity Updates:
deb          http://security.debian.org/             stable/updates main contrib non-free


### Java 1.3
deb          ftp://ftp.informatik.hu-berlin.de/pub/Java/Linux/debian woody non-free


### Eigenes Verzeichnis
deb          file:/verzeichnis/ ./
```

12 CD Brennen

Nach CD-Rekorder suchen:

```
cdrecord -scanbus
```

Info über CD-Rekorder und Rohling. Der *cdrecord* Parameter *dev=0,1,0* bestimmt den CD-Brenner (SCSI-Bus,SCSI-ID,LUN) bei Handelsüblichen Brennern ist Logical Unit Number LUN 0:

```
cdrecord dev=0,1,0 -atip
```

Info über Fähigkeiten des CD-Rekorders:

```
cdrecord -prcap dev=0,1,0
```

CD-Laufwerk öffnen:

```
cdrecord dev=0,1,0 -eject > /dev/null 2>&1
```

12.1 Daten CD

Daten-CD über Image-Datei kopieren (keine Audio-CD):

```
dd if=/dev/cdrom of=/tmp/cdimage
cdrecord -v dev=0,1,0 speed=8 /tmp/cdimage
```

Home-Verzeichnisse mit allen Unterverzeichnissen auf CD sichern (Rockridge, Joliet und TRANS.TBL):

```
mkisofs -o /tmp/image.iso -R -T -J -V CD_Name /home
cdrecord -v dev=0,1,0 speed=8 /tmp/image.iso
```

Home-Verzeichnisse mit allen Unterverzeichnissen auf CD sichern ohne Image-Datei (On the fly). Der Parameter fs=6m gibt den Puffer im Arbeitsspeicher an:

```
mkisofs -r /home | cdrecord -v fs=6m speed=4 dev=0,1,0 -
```

ISO-Imagedatei mounten um zu testen. ACHTUNG es darf nichts am Dateisystem verändert werden, deshalb nur Leserecht (ro):

```
mount -t iso9660 -o loop,ro /mnt/test /tmp/homeimage.iso
```

Direkt-Kopie von CD-Rom erstellen (ohne Imagedatei) *-isosize* liest nur den vom ISO-Dateisystem belegten Platz der CD *fs=6* = 6MB Puffer im RAM:

```
nice --18 cdrecord -v -isosize fs=6m speed=2 dev=0,1,0 /dev/cdrom
```

Bootbare CD erstellen. Zuerst eine exakte Kopie einer Bootdiskette in das Verzeichnis (oder einem Unterverzeichnis) erstellen, in dem sich die Daten für die CD befinden (hier */Daten*) :

```
dd if=/dev/fd0 of=/Daten/boot.img bs=18k
```

Anschließend das ISO-Image erstellen, der Parameter *-b* gibt das Boot-Image an (relative Pfadangabe):

```
mkisofs -b boot.img -o /tmp/cd.iso -R -T -J -V CdName /Daten
```

Zum Schluß kann das zuvor erstellte Image auf CD gebrannt werden:

```
cdrecord -v dev=0,1,0 speed=8 /tmp/cd.iso
```

Image von Festplatte erstellen:

```
dd if=/dev/hda1 | gzip | \
cdrecord -v -dummy dev=/dev/sg0 speed=4 -data -
```

schreibt Festplattenpartition auf die CD und umgeht das Dateisystem, also kein ISO9660! für single Session. Natürlich muß -dummy entfernt werden. cdrecord liest von stdin '-' diese CD kann man nicht mounten aber mit

```
dd if=/dev/hdc | gzip -d > /dev/hda1
```

wieder zurückschreiben

12.2 Audio CD / Mpeg3

Alle Titel einer Audio-CD ins WAV-Format umwandeln die Titel werden als *track01.cdda.wav* usw. in dem Verzeichnis gespeichert in dem cdparanoia gestartet wird:

```
cdparanoia -v -d /dev/scd0 -B
```

Den Titel 3 einer Audio-CD nach */tmp/track.Liedtitel.wav* auslesen:

```
cdparanoia -v -d /dev/scd0 -B 3 /tmp/Liedtitel.wav
```

Den Mpeg3-Titel *nin_wish.mp3* ins WAV-Format *nin_wish.wav* umwandeln:

```
mpg123 -w nin_wish.wav nin_wish.mp3
```

Shell Script: Alle Mpeg3-Dateien im aktuellen Verzeichnis ins WAV-Format umwandeln.

```
#!/bin/bash
for i in $( ls *.[mM][pP]*3 )
do
   mpg123 -w ${i%.*}.wav $i
done
```

Alle Titel (.wav files) aus dem aktuellen Verzeichnis auf CD brennen:

```
cdrecord dev=0,1,0 fs=16384k -v -useinfo speed=8 \
-dao -pad -audio $(ls *.wav)
```

Audio CD in mehreren Schritten erstellen. Die CD wird im dritten Schreibvorgang abgeschlossen und beinhaltet fünf Titel:

```
cdrecord -v -audio -pad -nofix dev=0,1,0 speed=8 1.wav 2.wav
cdrecord -v -audio -pad -nofix dev=0,1,0 speed=8 3.wav 4.wav
cdrecord -v -audio -pad dev=0,1,0 speed=8 5.wav
```

Audio CD mit cdrdao kopieren (1–1 Kopie keine 2 Sekunden Pause):

```
cdrdao read-cd -v 2 --read-raw --paranoia-mode 3 \
--device 0,0,0 --driver generic-mmc \
--datafile cdimage.bin cd.toc

cdrdao write -v 2 --buffers 64 --speed 8 \
--device 0,1,0 --driver generic-mmc \
--eject cd.toc
```

Audio CD mit cdda2wav und cdrecord kopieren (1-1 Kopie keine 2 Sekunden Pause):

```
cdda2wav -D 0,0,0 -g -0 wav -S 4 -v30 -P 0 -n 75 -B

cdrecord dev=0,1,0 -fs=16386k -v -useinfo speed=4 \
-dao -eject -pad -audio $(ls *.wav)
```

Infos zum CD-Brennen unter Linux in:

c’t Heft 3/2000 Seite 224
c’t Heft 8/2000 Seite 194
Lunux Magazin Heft 07/2001 ab Seite 26
Lunux User Heft 05/2001 ab Seite 14

13 X-Window

Ein X-Window Programm z.B. Gimp von dem Rechner “PC1” auf dem X-Server eines anderen Rechners “PC2” starten.
Folgende Voraussetzungen muessen erfuellt sein:

- Der X-Server muss an einem TCP-Port (6000 - 6007) lauschen. Eventuell muss der Parameter “-nolisten tcp” von einer der folgenden Dateien entfernt werden (je nach Startmethode)

`/etc/X11/xinit/xserverrc`	fuer startx oder xinit
`/etc/X11/xdm/Xservers`	fuer xdm
`/etc/kde2/kdm/Xservers`	fuer kdm

- Der X-Server von “PC2” muss freigegeben sein (“PC1” muss das Recht haben auf dem X-Server von “PC2” Programme anzuzeigen). Die Freigabe erreicht man auf “PC2” indem man unter dem X eine Konsole öffnet und den folgenden Befehl absetzt. Die folgenden Befehle setzen voraus das eine Namensaufloesung existiert. Altanativ kann auch die IP-Adresse statt des Rechnernamens angegeben werden
  ```
  xhost +pc1
  ```
  oder einfach jeden auf seinen X-Server lassen:
  ```
  xhost +
  ```

- PC1 muss wissen wo er das Programm anzeigen soll, dies kann durch setzen der DISPLAY Variable geschehen:
  ```
  export DISPLAY=PC2:0.0
  ```
  Damit werden alle grafischen Programme am ersten Monitor des ersten X-Servers von “PC2” angezeigt.
  Fuer einzelne Programme reicht es den Parameter - -display anzugeben, z.B. fuer Gimp:
  ```
  gimp --display PC2:0.0
  ```

Zweiten X-Server mit IceWM als Windowsmanager starten. Den X-Server erreicht man normalerweise mit Ctrl + Alt + F8. Eventuell muss der komplette Pfad von icewm mit angegeben werden:
```
startx icewm -- :1
```

14 Einen eigenen Kernel erstellen

Achtung der folgende Abschnitt ist mit groesster Vorsicht zu geniesen. Es sollte auf alle Faelle eine funktionierende Bootdiskette vorhanden sein.
Um einen eigen Kernel zu erstellen werden natuerlich die Kernel Quellen benoetigt, diese sind bei den meisten Distributionen als Paket enthalten und koennen so einfach installiert werden. Alternativ koennen die Quellen auch von *www.kernel.org* heruntergeladen werden. Nach den Installieren, bzw. dem Entpacken der Quellen sollten sich diese unter */usr/src* befinden und */usr/src/linux* ist meist ein

Link auf die aktuellen Quellen. Gute Dokumentation zum Konfigurieren und Erstellen eines eigenen Kernels findet man unter */usr/src/linux/Documentation/* und *www.linuxfibel.de/kconf.htm*
Zum Erstellen des Kernels sind folgende Schritte noetig:

In das Verzeichnis mit den Quellen wechseln:

```
cd /usr/src/linux
```

Der Kernel kann mit einem der Folgenden Befehlen konfiguriert werden:

```
make config
make menuconfig
make xconfig
```

Abhängigkeit der Quell– und Include–Dateien herstellen

```
make dep
```

Alte Objektdateien löschen

```
make clean
```

Kernel erstellen zImage für kleinen und unkomprimierten Kernel (veraltet), bzImage für großen komprimierten Kernel (Achtung großes I bei zImage und bzImage).

```
make bzImage
```

Module erstellen und ins Modul-Verzeichnis kopieren

```
make modules
make modules_install
```

Kernel nach /boot/ kopieren

```
cp /usr/src/linux/arch/i386/boot/zImage /boot/eigener_Kernel
```

Eine Intitialisierungs Ramdisk erzeugen:

```
mkinitrd -o /boot/initrd.eigener_Kernel /usr/src/linux
```

Nun muss der neue Kernel dem Bootlader bekanntgemacht werden. Fuer lilo als Bootloader muss die Datei */etc/lilo.conf* angepasst und anschliessend */sbin/lilo* ausgefuehrt werden.
Beispieleintrag fuer /etc/lilo.conf:

```
image=/boot/eigener_Kernel
        label="EigenerKernel"
        root=/dev/hda1
        initrd=/boot/initrd.eigener_Kernel
```

15 Shell Skript

Eine *for*-Schleife die bis *100* zählt:

```
for i in {0,1,2,3,4,5,6,7,8,9}{0,1,2,3,4,5,6,7,8,9}
do
   echo $i
done
```

In einem Skript eine Ja-Nein Abfrage vom Benutzer auswerten (Beispiel 1):

```
#!/bin/bash

echo -en "\nDo you want \"Yes\" or \"No\"? [Y/n] "
read x

if [ ${x:=y} == y -o ${x:=y} == Y ]
then
   echo Yes
else
   echo No
fi
```

In einem Skript eine Ja-Nein Abfrage vom Benutzer auswerten (Beispiel 1):

```
#!/bin/bash

echo -en "\nDo you want \"Yes\" or \"No\" (Ja oder Nein)? "
read x

case $x in
 [YyJj]* ) echo "Ja Yes "   ;;
 [Nn]*   ) echo "Nein No"   ;;
esac
```

In einem Skript testen ob der erste Parameter eine ganze Zahl ist (z.B. 2 4321 1000):

```
#!/bin/bash

case $1 in
 '' | *[!0-9]*  )       echo "$1 is no digit" ; exit 0 ;;
 *              )       echo "$1 is a digit"  ; exit 1 ;;
esac
```

In einem Shell-Skript bis 100 zaehlen (mit fuehrenden Nullen):

```
#!/bin/bash

i=1
while [ $i -lt 101 ]
do
j=00$i
   echo ${j:$((-3))}
   let i=$i+1
done
```

Rechnen mit einem Shellskript: Errechnen und Anzeigen des Schachbrett – Reiskorn Problems:

```
#!/bin/bash

i=2 ; j=1 ; x=1
echo "Auf dem 1. Feld ist 1 Stein"
while [ $j -lt 64 ]
do
   x=$(echo "${x}+${i}"|bc)
   echo "Auf dem $((${j} + 1)). Feld sind $i Koerner"
   i=$(echo "${i}*2"|bc)
   let j=$j+1
done
echo -e "\nAuf dem Brett sind insgesammt $x Koerner"
```

Eine Datei zeilenweise einlesen und mit vorangestellter Zeilen-Nummer ausgeben. In der dritten Zeile des Skriptes wird die Standardeigabe auf den File-Descriptor 3< umgeleitet und die Datei */etc/fstab* wird nach 0< (zuvor stdin) gelenkt. Anschliessend kann mit dem Befehl read Zeile fuer Zeile bis EOF gelesen werden:

```
#!/bin/bash

exec  3<&0 0< /etc/fstab

typeset -i i=1

while
read  line
do
   echo -e "$i\t $line"
   i=i+1
done
```

16 AWK

Mit der von Aho, Weinberger und Kernighan entwickelten Programmiersprache awk lassen sich Textdateien nach bestimmten Textmustern durchsuchen:

Die 7. Zeile der Datei */etc/fstab* ausgeben:

```
awk 'NR == 7 {print}' /etc/fstab
```

Alle Zeilen der Datei */etc/fstab* ausgeben, die mit der Zeichenkette */dev* beginnen:

```
awk '/^\/dev/' /etc/fstab
```

Alle Zeilen der Datei */etc/fstab* ausgeben, die nicht mit dem Kommentarzeichen # beginnen:

```
awk '/^[^#]/' /etc/fstab
```

Alle Zeilen der Datei *Zeiten.txt* ausgeben, in denen ein gueltiges Zeitformat enthalten ist (hh:mm:ss).

```
awk '/([01][0-9]|2[0-3]):[0-5][0-9]:[0-5][0-9]/ \
   {print}' Zeiten.txt
```

17 Sonstige Befehle

Erstellen einer 2MB RAM-Disk:

```
dd if=/dev/zero of=/dev/ram bs=1k count=2048
```

Erstellen des Dateisystems auf der RAM-Disk:

```
mkfs -t ext2 /dev/ram
```

mounten der RAM-Disk:

```
mount /dev/ram /mnt/ram
```

Zerlegen der Datei *datei.xyz* in mehrere Teile mit einer Größe von z.B. 1MB. Es entstehen nun Dateien mit den Namen: *xaa, xab, xac ... xyy xyz xzaaa xzaab ...*:

```
split -b 1048576 datei.xyz
```

Die Einzelnen Dateien können mit:

```
cat x?* > neuedatei.xyz
```

wieder zusammengefügt werden. Es dürfen sich natürlich keine anderen Dateien die mit x beginnen in dem Verzeichnis befinden.
Das Ergebnis kann mit:

```
diff datei.xyz neuedatei.xyz
```

überprüft werden wobei keine Unterschiede auftreten dürfen.

Eine Datei mit Secure Copy (scp) auf den Rechner *pc2* in das Verzeichnis */home/rup/Desktop/* kopieren:

```
scp TestDatei rup@pc2:/home/rup/Desktop/
```

Softlink erzeugen:

```
ln -s dateiname linkname
```

Hardlink erzeugen erzeugen. Funktioniert nicht über mehrere Dateisysteme, erscheint mit `ls -l` nicht als Link:

```
ln dateiname linkname
```

Systemausgaben anzeigen lassen (Letzten 10 Zeilen der Datei */var/log/messages* immer aktuell):

```
tail -f /var/log/messages
```

Uhr über Time-Server abgleichen (time):

```
/usr/sbin/netdate -v wrzx03.rz.uni-wuerzburg.de \
&& /sbin/hwclock -wu
```

oder

Uhr über Time-Server abgleichen (ntp):

```
/usr/sbin/ntpdate ntps1-0.cs.tu-berlin.de \
&& /sbin/hwclock -wu
```

oder

```
date -u -s $(telnet 131.107.1.10 13 |grep UTC |cut -b16-23) \
&& /sbin/hwclock -wu
```

Spezialdatei fifo (first in first out) erzeugen:

```
mkfifo /tmp/fifo
```

Prüfen ob der Prozesse mit dem Namen “Netscape” laufen:

```
ps -ax | grep -i Netscape
ps -ef | grep -i Netscape
```

Alle Prozess-ID’s anzeigen (’ ist ein Hochkomma $ 2 ist die zweite Spalte):

```
ps -ef | awk ’{print $2}’
```

Die vom Internet-Service-Provider zugewiesene IP anzeigen:

```
/sbin/ifconfig ppp0 | grep "inet Adr" \
| awk ’{ print $2 }’ | awk -F ’:’ ’{ print $2 }’
```

Zählen wie viele Dateien sich im Verzeichnis befinden:

```
ls | wc -w
```

Fünf Verzeichnisse mit einer for-Schleife erstellen (Verzeichnis_1 – Verzeichnis_5):

```
for i in 1 2 3 4 5
do
mkdir Verzeichnis_$i
done
```

Aus allen Dateien im aktuellen Verzeichnis die Leerzeichen entfernen. Aus *Bild 001.jpg* wird *Bild001.jpg* . Der Parameter -i von mv fraegt ob eine existierende Datei ueberschrieben werden soll.

```
$ for i in * ;do mv -i "$i" $(echo "$i" \
| tr -d [:blank:]) ;done
```

Bei allen Dateien im aktuellen Verzeichnis die Leerzeichen durch einen Bindestrich ersetzen. Aus *Bild 001.jpg* wird *Bild-001.jpg* . Der Parameter -i von mv fraegt ob eine existierende Datei ueberschrieben werden soll.

```
$ for i in * ;do mv -i "$i" $(echo "$i" \
| tr [:blank:] - ) ;done
```

Alle Dateien mit der Endung .jpeg ins home-Verzeichnis kopieren und die Endung in jpg umbenennen:

```
for i in $(ls *.jpeg)
do
cp $i /home/${i%.*}.jpg
done
```

Alle Bild-Dateien mit der Endung .jpg um 50% verkleinern und als *_klein.jpg abspeichern

```
for i in *.jpg
do
convert $i -geometry 50% $(basename $i .jpg)_klein.jpg
done
```

Alle Jpeg-Bilder im aktuellen Verzeichnis mit der Endung *.jpg* ins Encapsulated PostScript Format umwnadeln und die Endung in *.eps* umbenennen:

```
for i in $(ls *.jpg)
do
convert $i /home/${i%.*}.eps
done
```

Von allen Jpeg-Dateinamen im Verzeichniss die ersten fuenf Zeichen entfernen. Z.B. aus Bild-Baum2.jpeg mache Baum2.jpeg. Natuerlich muessen alle Dateinamen mindestens sechs Zeichen lang sein und duerfen in diesem Fall nicht laenger als 100 Zeichen lang sein.

```
for i in $(ls *.jpeg)
do
mv $i ${i:5:100}
done
```

Shell Script: Alle jpeg-Dateien im aktuellen Verzeichnis in fortlaufende Zahlen (00001.jpg 00002.jpg . . .) umbenennen. Der neue Dateiname ist 5-stellig plus Endung. Achtung es darf nur ein Punkt im Dateinamen vorkommen und Dateinamen mit Endungen wie **.jpxyzg* würden auch umbenannt werden:

```
#!/bin/bash

declare -i Zahl=0
for AlterDateiname in $(ls *.[jJ][pP]*[gG])
do
   Zahl=$Zahl+1
   NeuerDateiname=0000${Zahl}
   mv $AlterDateiname ${NeuerDateiname:$((-5))}.jpg
done
```