Seab@er Software AG

Shell Commands
& More

1. Vorwort

Die Konsole ist ein mächtiges Werkzeug. Um die Konsole auszureizen oder auch nur mal nach einem passenden Befehl nachzuschlagen, wurde diese Dokumentation verfasst.

Alle Befehle sind mit verschiedenen Distributionen und in der Bash Shell getestet worden.

Bei Fragen und Anregungen bin ich unter folgender Mail Adresse zu erreichen:

uwe@seabaer-ag.de

2. Konfiguration

2.1 Bash Konfiguration

Im persönlichen Home Verzeichnis liegt versteckt die Datei `.bashrc`. In dieser Datei kann man die Bash konfigurieren. In dieser Datei können Aliase, Variablen und Funktionen definiert werden. Die Konfiguration wird mit `source ./.bashrc` neu geladen.

Beispiel einer .bashrc

```
# Aliase definieren
alias ls='ls –lahF --color=tty'
alias dir=’ls --color=auto --format=vertical’
alias df='df –h'
alias du='du –sch'

# Variablen definieren
strBackupPath=/backup/tuxserver
strBackupLog=backup.log

export strBackupPath strBackupLog
export GREP_OPTIONS=’--color=auto’
export GREP_COLOR=’1;32’ # Ansi Farbe hellgrün

# Functions
back() {
  cd $strBackupPath
  vi $strBackupLog
}
# Berechtigungen setzen fuer neuangelegte Objekte
umask 022
```

In der nachfolgenden Tabelle werden die Konfigurationsdateien aufgelistet, die bei einem Login / Aufruf an einer Shell in der Reihenfolge verarbeitet werden.

Konfigurations-Datei	Login Shell	Interaktive Shell	Beschreibung
/etc/profile	X	-	Systemweit, wird bei einem Update überschrieben.
/etc/profile.local	X	-	Systemweit, bleibt bei einem Update erhalten.
/etc/bash.bashrc	X	-	Systemweit, wird bei einem Update überschrieben.
~/.bashrc	X	X	Benutzerkonfiguration
~/.alias	X	X	Benutzerkonfiguration. In dieser Datei können die Aliase verwaltet werden.
~/.bash_profile ~/.bash_login ~/.profile	X	X	Benutzerkonfiguration. Wird eine der Dateien gefunden, so wird nur diese verarbeitet, auch wenn die anderen Dateien vorhanden sind.

Bei einem Logout wird die Datei `~/.bash_logout` verarbeitet.

2.2 Shell-Gedächnis

Die bereits eingegebenen Befehle werden in der `~/.bash_history` gespeichert. Standardmäßig werden die letzten 500 Befehle eingetragen. Mit `uws@tux>echo $HISTSIZE` kann man sich den Wert anzeigen lassen. Möchte man mehr Befehle speichern, so kann das mit `uws@tux>export HISTSIZE=700` gemacht werden.

Die Suchfunktion schaltet man mit `[strg+R]` oder `[strg+s]` ein. Der Prompt verändert sich und nun kann man den zu suchenden Befehl eingeben. Die Bash vervollständigt die Eingabe automatisch. Um weiter zu suchen, noch mal `[strg+R]` oder `[strg+s]` drücken. Der gefundene Befehl wird dann mit `Enter` abgeschickt und mit drücken der `Esc-Taste` kann man den Befehl editieren.

Mit `uws@tux>fc –l -10` werden die letzten 10 Befehle angezeigt.

2.3 Befehle mehrzeilig

Wird eine Anweisung in der Shell zu lang, so kann man sie mit && am Ende der Zeile und nach einem Return in der zweiten Zeile weiterschreiben. Al Prompt Text wird die Variable PS2 ausgewertet.

```
uws@tux>PS2=“Weiter geht’s>“

uws@tux>grep oracle /etc/passwd &&

Weiter geht’s>echo found
```

2.4 Tastenzuordnung

Eine Tastenzuordnung kann man auf zwei Arten realisieren. Entweder man definiert die Tastenzuordnung in der .`bashrc` oder in der .`inputrc`.

Beispiel der .bashrc

```
# F1: zeigt /var/log/messages
bind ’“\eOP“:“tail –f /var/log/messages\n“’
# F2: wechselt nach /etc
bind ’“\eOQ“:“cd /etc\n“’
# F3: wiederholt das letzte Wort der aktuellen Zeile
bind ’“\eOR“:“!#$\e^“’
```

Beispiel der .inputrc

```
# F1: zeigt /var/log/messages
“\eOP“:“tail –f /var/log/messages\n“
# F2: wechselt nach /etc
“\eOQ“:“cd /etc\n“
# F3: wiederholt das letzte Wort der aktuellen Zeile
“\eOR“:“!#$\e^“
```

2.5 Tastaturfunktionen

Taste	Erklärung
[Strg] + [A]	springt an den Anfang der Eingabezeile
[Strg] + [E]	springt an das Ende der Eingabezeile
[Alt] + [B]	springt ein Wort nach links in der Eingabezeile
[Esc] + [B]	geht ein Wort zurück
[Alt] + [F]	springt ein Wort nach rechts in der Eingabezeile
[Esc] + [F]	geht ein Wort vor
[Strg] + [K]	löscht von der Cursor-Position aus bis zum Ende der Zeile
[Strg] + [U]	löscht von der Cursor-Position aus bis zum Anfang der Zeile
[Strg] + [W]	löscht ein Wort nach links
[Strg] + [T]	vertauscht die beiden Zeichen vor und hinter dem Cursor
[Alt] + [T]	vertauscht die beiden vorangehenden Wörter in der Eingabezeile
[Esc] + [T]	vertauscht die beiden vorangehenden Wörter
[Strg] + [L]	räumt das Terminal Fenster auf

2.6 Prompt

2.6.1 Bash

Den Prompt kann mit der Variable PS1 eingestellt werden.

```
uws@tux>export PS1="[\t] \u@\h:\w>"
[10:26:22] uws@tux:~>
```

In der nachfolgenden Tabelle ist ein Auszug der Escape Sequencen.

Escape-Sequence	Beschreibung
\a	Der Ascii Klingel character (07)
\d	Das Datum in „Wochentag Monat Datum“ z.B. Mon Mai 22
\e	Der Ascii Escape character (033)
\h	Der hostname bis zu ersten ‚.’
\H	Der hostname
\j	the number of jobs currently managed by the shell
\l	the basenameof the shell’s terminal device name
\n	Newline
\r	carriage return
\s	the name of the shell, the basename of $0
\t	Die aktuelle Zeit im 24 Stunden Format HH:MM:SS
\T	Die aktuelle Zeit im 12 Stunden Format HH:MM:SS
\@	Die aktuelle Zeit im 12 Stunden Format am/pm
\u	Den Usernamen des angemeldeten User
\v	Die Version der Bash
\V	Das Relase der Bash, Version + patchlevel
\w	Das aktuelle Arbeitsverzeichnis
\W	the basename of the current working directory
\!	the history number of this command
\#	the command number of this command
\$	if the effective UID is 0, a #, otherwise a $
\nnn	the character corresponding to the octal number nnn
\\	a backslash
\[	begin a sequence of non-printing characters, which could be used to embed a terminal control sequence into the prompt
\]	end a sequence of non-printing characters

Dem Prompt kann man auch farbig darstellen, um zum Beispiel dem User root eine andere Farbe zuzuweisen. Die Angabe der Farbe wird durch \[und \] begrenzt.

Sequenz	**Farbe**
\[\033[0;30m\]	Schwarz
\[\033[1;30m\]	Dunkelgrau
\[\033[0;31m\]	Rot
\[\033[1;31m\]	Hellrot
\[\033[0;32m\]	Grün
\[\033[1;32m\]	Hellgrün
\[\033[0;33m\]	Braun
\[\033[1;33m\]	Gelb
\[\033[0;34m\]	Blau
\[\033[1;34m\]	Hellblau
\[\033[0;35m\]	Lila
\[\033[1;35m\]	Helles lila
\[\033[0;36m\]	Dunkels türkis
\[\033[1;36m\]	Türkis
\[\033[0;37m\]	Hellgrau
\[\033[1;37m\]	Weiß
\[\033[0m\]	Auf default Farbe setzten

Auch die Hintergrundfarbe kann eingestellt werden.

Sequenz	**Farbe**
\[\033[XXm\]	Keine Hintergrundfarbe
\[\033[40;XXm\]	Schwarzer Hintergrund
\[\033[41;XXm\]	Roter Hintergrund
\[\033[42;XXm\]	Grüner Hintergrund
\[\033[43;XXm\]	Hellbrauner Hintergrund
\[\033[44;XXm\]	Blauer Hintergrund
\[\033[45;XXm\]	Lila Hintergrund
\[\033[46;XXm\]	Türkiser Hintergrund
\[\033[47;XXm\]	Hellgrauer Hintergrund

Die Uhrzeit kann in dem nachfolgenden Beispiel in der rechten oberen Ecke platziert werden. Die Variable COUMNS enthält die Anzahl der Spalten im Terminal.

```
uws@tux01>PS1=“>\033[s\033[1;\$((COLUMNS-4))f\$(date +%H:%M)\033[u“
```

Noch mehr Beispiele.

```
uws@tux>PS1="\$? \$(if [[ \$? == 0 ]]; then echo \"\[\033[0;32m\];)\"; else
echo \"\[\033[0;31m\];(\"; fi)\[\033[00m\] : "

0 ;) : true
0 ;) : false
1 ;( :
```

```
uws@tux>PS1="\[\033[01;37m\]\$?  \$(if  [[  \$?  ==  0  ]];  then  echo
\"\[\033[01;32m\];)\"; else echo \"\[\033[01;31m\];(\"; fi) $(if [[ ${EUID}
== 0 ]]; then echo '\[\033[01;31m\]\h'; else echo '\[\033[01;32m\]\u@\h';
fi)\[\033[01;34m\] \w \$\[\033[00m\] "

0 ;) uws@tux ~ $ true
0 ;) uws@tux ~ $ false
1 ;( uws@tux ~ $
```

```
uws@tux>PS1="\[\033[01;37m\]\$?  \$(if  [[  \$?  ==  0  ]];  then  echo
\"\[\033[01;32m\]\342\234\223\"; else echo \"\[\033[01;31m\]\342\234\227\";
fi) $(if [[ ${EUID} == 0 ]]; then echo '\[\033[01;31m\]\h'; else echo
'\[\033[01;32m\]\u@\h'; fi)\[\033[01;34m\] \w \$\[\033[00m\] "

0 √ uws@tux ~ $ true
0 √ uws@tux ~ $ false
1 × uws@tux ~ $
```

Im nachfolgenden Beispiel wird der Prompt mit der Farbe `Grün` versehen. Am Anfang leitet das `\e` die Farbe ein und am Ende der Zeile wird durch `\e[m` die Farbe wieder ausgeschaltet.

```
uws@tux>PS1="\e[1,32m \u@\w:~> \e[m"
```

2.6.2 Zsh

Die Farbwerte sind in der oberen Tabelle beschrieben.

```
uws@tux>PS1=$'\e[1;32m%n@%/:~>\e[0m'
uws@tux:~>
```

Die nachfolgende Liste ist nur ein Auszug.

Escape-Sequence	**Beschreibung**
%/	Arbeitsverzeichnis (PWD)
%~	Arbeitsverzeichnis, aber das Home Verzeichnis wir mit ~ angezeigt.
%M	Der ganze Hostnamen
%n	Der Hostname bis zum ersten Punkt.
%@	Uhrzeit im 12 Stunden Format
%T	Uhrzeit im 24 Stunden Format
%*	Uhrzeit im 24 Stunden Format incl. der Sekunden
%n	Username
%N	Name eines Scripts, Source File oder Shell
%w	Datum im day-dd Format
%W	Datum im mm/dd/yy Format
%D	Datum im yy-mm-dd Format.

Die Farbe des Prompts nach dem Shell Background Farbe einstellen. In dem nachfolgenden Beispiel wird abgefragt, ob die Hintergrundfarbe Weiß ist.

```
if [ "${COLORFGBG}" = "0;15" ]; then
 PS1=$'\e[0;30m%n@%m %~\e[om\n$ ,
else
 PS1=$'\e[1;32m%n@%m \e[1;33m %~\e[om\n$ ,
fi
```

2.7 VI Editor

2.7.1 Konfiguration

Die Einstellungen für den VI Editor können global in der Datei `/etc/vimrc` oder für jeden Benutzer in der Datei `~/.vimrc` vorgenommen werden.

```
uws@tux>cat .vimrc
“ Kommentarzeile
“ Syntax Color einschalten
syntax on
set nowrap

“ Color config, possible colors
“ black, blue, cyan, gray, grey, green, magenta, red, white, yellow, brown
“ DarkBlue, DarkGreen, DarkCyan, DarkRed, DarkMagenta, DarkYellow
“ LifgtGray, LightGrey, DarkGray, DarkGrey, LighBlue, LightBlue, LightCyan
“ LightRed, LightMagenta, LightYellow
“ highlight groups = comment, constant, normal, nontext, special, cursor
hi normal ctermfg=white ctermbg=black guifg=white guibg=black
hi nontext ctermfg=blue ctermbg=black guifg=blue  guibg=black
hi comment ctermfg=green ctermbg=white

“ Backup
set backup
set backupdir=~/tmp

“ swap file
set dir=~/tmp

“ show line numbers
set number

“ show editing mode (insert/replace) on the last line
set showmode

“ highlight matching search strings
set hlsearch

“ show matching brackets
set showmatch

“ make searches case insensitive
set ignorecase

“ Tab size (Anzeige und Plazierung)
set tabstop=3

“ show cursor line and column in the status line
set ruler

“ color scheme, files under /usr/share/vim/vim72/colors
color desert
```

2.7.2 Starten

Den Editor vi kann man mit verschiedenen Optionen öffnen.

Kommando	Beschreibung
vi	Ohne Datei starten.
vi <datei>	Starten mit der angegebenen Datei.
vi +<datei>	Es wird in einer Kopie gearbeitet und erst bei dem beenden des Editors wird die Datei geschrieben.
vi –r <datei>	Die Bearbeitung wird nach einem Absturz an der gleichen Stelle fortgesetzt.
vi –R <datei>	Die Datei wird schreibgeschützt geöffnet,

2.7.3 Befehle

Folgende Befehle gibt es im VI-Editor. Die Auflistung ist nicht vollständig.

Speichern und Beenden

:q	Editor beenden
:q!	Editor beenden ohne speichern
:w	Datei speichern
:w!	Speichern erzwingen
:wq oder ZZ	Speichern und beenden
<strg> + z	Shell starten, zurück mit 'fg'

Navigieren

G	ans Ende springen
1G	in die erste Zeile springen
0	an den Anfang der Zeile springen
^	an den Anfang der Zeile springen
$	an das Ende der Zeile springen
b	an den Anfang des letzten Wortes springen
w	an den Anfang des nächsten Wortes springen
e	an das Ende des letzten Wortes springen
h	ein Zeichen nach links
j	eine Zeile nach unten
k	eine Zeile nach oben
l	ein Zeichen nach rechts
nG	n-te Zeile der Datei.
H	Die erste Zeile des Bilschirms
+	Die nächste Zeile der Textanfang
-	Die vorherige Zeile der Textanfang
(	Satzanfang
)	Satzende
{	Absatzanfang
}	Absatzende
[	Am Anfang der Datei
]	Dateiende
<strg> + u	nach oben blättern
<strg> + d	nach unten blättern
L	in die letzte Zeile der Bildschirmausgabe springen

Löschen

x	Zeichen löschen
r	Zeichen ersetzen
s	Zeichen ersetzen und in den Bearbeitungsmodus wechseln
dd	aktuelle Zeile ausschneiden (Zwischenablage)
[n]dd	die nächsten [n] Zeilen ausschneiden
C	Zeile löschen
V	Zeile markieren
v	Zeichen markieren

d	markierten Text löschen
cw	aktuelles Wort ersetzen
dw	aktuelles Wort löschen, incl angehängtem Leerzeichen
de	aktuelles Wort löschen, über Zeile hinaus
dl	Zeichen unter Cursor-Position löschen
dj	zwei Zeilen löschen, Position ist die 1. Zeile
dk	zwei Zeilen löschen, Position ist die 2. Zeile
:5,10d	Zeilen 5 bis 10 löschen
:.,$d	alle Zeilen löschen
nx	N Zeichen ab Cursorposition
dL	Bis zum unteren Bildschirmrand löschen.
d)	Bis zum Absatzende löschen.
D	Bis zum Zeilenende löschen.

Einfügen

p	Zwischenablage nach aktueller Zeile einfügen
P	Zwischenablage vor aktueller Zeile einfügen
R	Text überschreiben
J	aktuelle Zeile an das Ende der vorherigen Zeile anhängen
o	eine Zeile nach der aktuellen Zeile einfügen und in den Bearbeitungsmodus wechseln.
O	eine Zeile vor der aktuellen Zeile einfügen und in den Bearbeitungsmodus wechseln.
a	ein Leerzeichen einfügen und in den Bearbeitungsmodus wechseln
A	zum Ende der Zeile springen und in den Bearbeitungsmodus wechseln
i	in den Bearbeitungsmodus wechseln
I	am Anfang der Zeile in den Bearbeitungsmodus wechseln
s<text>	Ein Zeichen wird durch Text ersetzt.
S<text>	Eine ganze Zeile wird durch Text ersetzt.
ns<text>	N Zeichen werden durch text ersetzt.
cw<text>	Ein Wort durch Text ersetzt.

Suchen und Ersetzen

/Tux	suchen nach dem Wort Tux
/	suche wiederholen
n	suche wiederholen
:s/nach/durch	in aktueller Zeile suchen & ersetzen
:1,7s/nach/durch	in Zeile 1 bis 7 suchen und ersetzen
:%s/nach/durch	in allen Zeile suchen und ersetzen
?<nach>	Die Suche erfolgt rückwärts
?	Wiederholung rückwärts.
:s/nach/durch/g	Nur in der aktuellen Zeile wird gesucht & ersetzt.

Kopieren

yy	Kopiert die aktuelle Zeile in den Puffer.
ny	Kopiert n+1 Zeilen in den Puffer.
yw	Kopiert ein Wort in den Puffer.

Sonstiges

u	undo
U	Alle Änderungen der aktuellen Zeile zurücknehmen

2.8 Sonderzeichen

Sonderzeichen	Beschreibung
;	Trennt Kommandos
:	Dummy-Kommando, führt nichts aus.
.	Ohne eigene Subshell das Kommando ausführen. z.B. . datei
#	Kommentar
&	Das Programm wird im Hintergrund ausgeführt.
&&	Bedingte Kommandoausführung. z.B. cmd1 && cmd2, dann wird cmd2 nur ausgeführt, wenn cmd1 erfolgreich war.
|	Verbindet Kommandos miteinander.
||	Bedingte Kommandoausführung, wie &&, nur das cmd2 dann ausgeführt wird, wenn cmd1 einen Fehler auswirft.
*	Jokerzeichen, alle
?	Jokerzeichen, genau ein Zeichen.
[xyz]	Jokerzeichen, eines aus „xyz“
[^xyz]	Jokerzeichen, alle ausser „xyz“
[ausdruck]	Andere Schreibweise für Test
~	Homeverzeichnis
>	Umleiten in einer Datei, Datei wird neu geschrieben.
>>	Umleiten in einer Datei, wird an die Datei angehängt.
>&	Umleiten der Standardausgabe
<	Einlesen einer Datei
<< ende	Einlesen einer Datei bis ende
(…)	Innerhalb der Klammer werden die Kommandos in einer Shell ausgeführt.
{…}	Kommandos gruppieren
{ , , }	Zeichenketten zusammenführen.
$	Inhalt von Variablen
$* oder $@	Liste der übergebenen Parametern an die Shell
$#	Parameteranzahl
$0	Name des Shell-Programms
$?	Rückgabewert des letzten Kommandos
$!	PID des letzten Hintergrundprozesses
$$	PID der aktuellen Shell
$1 … $9	Parameter 1 bis 9
$(…)	Kommandosubstitutuuin
${…}	Funktionen zur manipulation von Zeichenketten
$[…]	Berechnungen
“…“	Sonderzeichen behalten ihre Wirkung
‘…‘	Sonderzeichen verlieren ihre Wirkung
´…´	Kommandosubstitution
\zeichen	Demaskieren, hebt die Wirkung des Sonderzeichens aus

3. Befehle

3.1 Ordner / Verzeichnisse

3.1.1 Anlegen

Ordner / Verzeichnisse kann man mit dem Befehl `mkdir <name>` anlegen.

Mit dem nachfolgenden Befehl werden alle drei Verzeichnisse gleichzeitig angelegt. Ohne den Parameter `–p` kann man nur die Verzeichnisse anlegen, wenn das vorherige Verzeichnis existiert. In dem Beispiel unten müssten also die Verzeichnisse daten und privat vorhanden sein.

```
uws@tux>mkdir –p daten/privat/auto
```

Sollen die Verzeichnisse nebeneinander erstellt werden, so setzt man die Verzeichnisnamen in geschweiften Klammern.

```
uws@tux>mkdir {daten,privat,auto}

uws@tux>ls
daten
privat
auto
```

3.1.2 Löschen

Möchte man Verzeichnisse mit einem Inhalt löschen, so wird der Parameter –r angegeben. Der Parameter löscht die Verzeichnisse und entfernt die darin enthaltenen Daten.

```
uws@tux>rm –r daten/ privat/ auto/
```

3.1.3 Berechtigungen

Die Berechtigungen für ein Verzeichnis können mit dem Befehl `chmod` gesetzt werden. Der Befehl `chown` setzt den Besitzer des Verzeichnisses und kann auch dir Gruppe setzen. Für das setzen der Gruppe gibt es auch noch den Befehl `chgrp`. Mit der Option –R werden alle Verzeichnisse rekursiv geändert

```
uws@tux>chmod –R 775 /u01/app

uws@tux>chown –R uws:users /u01/app

uws@tux>chgrp –R users /u01/app
```

4 = lesen
2 = schreiben
1 = ausführen

Eine andere Möglichkeit mit chmod die Rechte zu vergeben, ist die Angabe der Benutzergruppe. Mit einem Minus Zeichnen werden der Gruppe die Rechte wieder entzogen.

```
uws@tux>chmod g=rw MeineDoku.txt

uws@tux>chmod g-w Meine Doku.txt

uws@tux>chmod g+x MeineDoku.txt
```

u = Eigentümer (User)
g = Gruppe (Group)
o = andere (other)
a = alle

3.1.4 ACL Berechtigungen

Die Zugriffsrechte lassen sich auch über `Access Control Lists` (ACL) setzen. Um ACL nutzen zu können, muss das Dateisystem mit den Optionen `user_xattr` und `acl` eingehängt sein. Mit dem Befehl `setfacl` kann man Benutzer und auch Gruppen dem jeweiligen Objekt zuordnen. Um sich die Berechtigungen anzeigen zu lassen, wird der Befehl `getfacl` verwendet. Hat man ACL Berechtigungen gesetzt, so wird bei einem `ls –l` ein '+' Zeichen hinter den Berechtigungen angezeigt. Mit Dolphin und auch mit dem Konqueror lassen sich die ACL Berechtigungen bequem mittels der GUI setzen.

```
uws@tux>setfacl –m u:oracle:rw MeineDoku.txt

uws@tux>setfacl –m g:privat:rw bild1.jpg

uws@tux>getfacl bild1.jpg
```

Man kann ein Verzeichnis mit `chmod 700` auf restriktiven Zugriff setzen um anschließen einem Benutzer oder auch nur einer Gruppe den Zugriff auf diesem Verzeichnis zu erlauben.

3.1.5 Sticky-Bit

Hat man einer Datei das Schreibrecht anderen Usern erteilt, so können diese User diese Datei auch löschen, da das Schreibrecht auch das löschen beinhaltet. Setzt man auf das Verzeichnis das `Sticky-Bit`, so darf nur der Eigentümer die Datei löschen oder umbenennen. Die anderen User dürfen nur noch in diese Datei schreiben. Das wegnehmen des Sticky-Bits geschieht anstelle des Plus Zeichens mit einem Minus Zeichen.

```
uws@tux>chmod +t ~/Transfer/Doku
uws@tux>ls –l /Transfer

drwxr-xr-t 2 uws users 296 Doku/

uws@tux>chmod +t MeineDoku.txt

uws@tux>chmod –t ~/Transfer/Doku
```

3.2 Software kompilieren

3.2.1 Allgemein

Nach dem entpacken des Programms, befinden sich im Verzeichnis die Dateien `README` und `INSTALL`. Diese Dateien ist die Dokumentation für das kompilieren des Programms. Eventuell gibt es noch ein Doc Verzeichnis. In der `Readme` Datei steht im Anschnitt `Requirements`, welche Pakete gebraucht werden.

Die meisten Programme werden mit den drei Befehlen `./configure`, `make` und `make install` übersetzt.

Der Aufruf von `configure` erzeugt eine Datei mit den Namen `Makefile`. In dieser Datei stehen alle Anweisung für den Compiler, Pfade zu den Bibliotheken und den Header-Dateien.

3.2.2 Grundausstattung

Damit das kompilieren auch funktioniert, werden die Compiler gebraucht. Für das kompilieren des Programms wird das Paket `gcc, gcc-c++, glibc-devel` und `xorg-X11-devel` gebraucht. Außerdem wird noch `autoconf, automake` und `make` gebraucht.

3.2.3 Configure

In dem Verzeichnis des zu Übersetzenden Programms wird nun der `configure` Befehl abgesetzt.

```
uws@tux>./configure
```

Dieses Script erstellt nun ein oder auch mehrere Makefiles. Ob es mehrere Makefiles werden, hängt von dem zu kompilierenden Programm ab. Wird der Befehl ohne Optionen abgesetzt, so landen die Dateien bei einer Installation im Verzeichnis /usr/local. Soll das Programm in einem anderen Verzeichnis installiert werden, so kann man die Option `--prefix=<Verzeichnis>` angeben.

```
uws@tux>./configure --prefix=/opt/games
```

Mit der Option `--help` kann man sich die zu verwendeten Parameter anzeigen lassen.

```
uws@tux>./configure --help
```

Soll das Programm nicht installiert werden um das Programm direkt aus dem Programm Ordner zu starten, so wird die Option `--disable-install` angegeben.

```
uws@tux>./configure --disable-install
```

3.2.4 Make

Mit dem Aufruf `make` wird nun das Programm kompiliert. Je nach Rechnergeschwindigkeit kann das kompilieren des Programms einige Zeit dauern. Stehen in den letzten Zeilen keine Fehlermeldungen (Fehler oder Error), so hat das kompilieren des Programms funktioniert.

```
uws@tux>make
```

3.2.5 Make Install

Wurde das `configure-Script` nicht mit der Option `--disable-install` aufgerufen, so muss es jetzt noch installiert werden. Die Installation erfolgt mit dem Aufruf `make install`.

```
uws@tux>make install
```

3.2.6 Fehlerbehebung

Meistens beschwert sich das configure-Script über fehlende Dateien oder Bibliotheken / Pakete. Eine solche Fehlermeldung könnte so lauten:

```
uws@tux>configue: error: Cannot find ncurses.h
```

Um herauszufinden, im welchen Paket diese Bibliothek sich befindet, kann man eine Suchmaschine dafür verwenden. Die Suchmaschine http://www.rpmseek.com kann hierzu verwendet werden. In dem Suchfeld gibt man die zu suchende Bibliothek ein und die Option Paket `enthält die Datei` sollte eingeschaltet sein. Ein Klick nun auf go listet nun das zu installierende Paket auf.

Einen Artikel über `configue-Fehler` gibt es unter dem folgenden Link:

`http://www.linux-user.de/ausgabe/2004/06/028-configure`.

Über make-Fehler gibt es einen Artikel unter den folgenden Link:

`http://www.linux-user.de/ausgabe/2004/06/032-make`.

3.3 LSOF - Geöffnete Dateien anzeigen

Braucht man für eine Diagnose eine Auflistung der gerade geöffneten Dateien, so kann hierzu der Befehl `lsof` genommen werden.

3.3.1 Optionen

Einen Überblick über die wichtigsten Optionen sind in der nachfolgenden Tabelle aufgelistet. (Aus LinuxUser 04.2016)

Option	Beschreibung
-a	Logisches Und
-b	Verhindert, dass Lsof blockierende Funktionen verwendet
-c <Zeichen>	Suchkriterium
+c <Anzahl>	Wie viele Zeichen sollen berücksichtigt werden.
+d <Verzeichnis>	Alles wird ausgegeben, was in dem angegebenen Verzeichnis ist.
-d <Muster>	Alles ausschließen, was dem Muster entspricht.
+D <Verzeichnis>	Wie +d, nur werden auch Unterverzeichnisse berücksichtigt.
+/-f	Definiert, wie Lsof Pfade interpretieren soll
-i4/-i6	IPv4 oder IPv6-Verbindungen berücksichtigen
-p <PID>	Angabe der PID
-t	Nur PID ausgeben.
-u <User>	Nur die Dateien ausgeben, die dem angegebenen User gehören.
-U	Unix Domain Sockets verwenden
-T <Key>	TCP/IP-Informationen gemäß Key ausgeben.
-s	Dateigröße anzeigen
-S <Sekunden>	Timeout für Kernel-Funktionen
+/-r <Sekunden>	Aktiviert den Wiederholungsmodus
-V	Markiert angeforderte, aber nicht gefundene Befehle, Dateien u.s.w.

3.3.2 Was hat ein Programm offen

```
uws@tux>lsof –c <programm_name>

uws@tux>lsof –c bash

COMMAND PID  User FD TYPE DEVICE SIZE/OFF NODE    NAME
bash    2823 uws  cwd DIR 8,36       4096 1071099 /home/uws
bash    2823 uws  rtd DIR 0,32        246     256 /
bash    2823 uws  txt REG 0,32     656584     275 /bin/bash
.
.
```

Tabelle FD (File-Deskriptoren)

Name	Beschreibung
cwd	Arbeitsverzeichnis (Current Working Directory)
txt	Textdatei
rtd	Wurzelverzeichnis (Root Directory)
mem	Bibliotheken (Memory-mapped File)
mmap	Gerätezugriff (Memory-mapped Device)
ltx	Shared Librarys
err	Zugriffsfehler
pd	Elternverzeichnis (Parent Directory)

Tabelle Type

Name	Beschreibung
REG	Reguläre locale Datei
DIR	Verzeichnis
PIPE	Pipe
IPv4, IPv6	IP-Verbindung (Socket)
DEL	Gelöschte Datei
BLK	Block Device
CHR	Character Device

3.3.3 Portzugriff

```
uws@tux>lsof –i :<port_no>

uws@tux>lsof –i :80
```

3.3.4 Verzeichniszugriff

```
uws@tux>lsof –D <Verzeichnis>

uws@tux>lsof –D /var/log
```

3.3.5 Benutzerzugriff

Wird vor dem Usernamen ein ^ vorangestellt, so werden alle offene Dateien aufgelistet, außer von dem Benutzer mit dem ^.

```
uws@tux>lsof –u <username>

uws@tux>lsof –u uws

uws@tux>lsof –u ^root
```

3.3.6 Prozesszugriff

```
uws@tux>lsof –p <pid>

uws@tux>lsof –p 2326

uws@tux>lsof –d txt
```

3.4 Dateien

3.4.1 Suchen

Findet alle Dateien, die größer sind als 10MB. Mit $ wird das Ergebnis von `find` als Parameter dem Befehl `ls` übergeben. Gefundene Dateien können mit einer Aktion weiterverarbeitet werden. Hierzu gibt es die Option `–exec`. Der Programmaufruf muss mit `{}` verwendet werden und mit einem Semikolon abgeschlossen werden. In der Bash muss das Semikolon mit \ maskiert werden. In dem dritten Beispiel werden alle Dateien aufgelistet, die älter als 10 Tage sind.

```
uws@tux>ls –lah $(find / -type f –size +10M)

uws@tux>find /usr –size +10M –exec ls –lah {} \;

uws@tux>find /usr –type f \( -name ‘*.*’ \) –ctime +10 –exec ls {} \;
```

Hier wird nach Dateien gesucht, die neuer sind als die angegebene Referenz Datei.

```
uws@tux>touch –date='10:00' /tmp/ref

uws@tux>find –newer /tmp/ref
```

Die gefundenen Dateien in einem Tar-Archiv packen.

```
uws@tux>find –iname '*.jpg' –exec tar –rf bilder.tar {} \;
```

Anzahl der Dateien in einem Verzeichnis anzeigen lassen.

```
uws@tux>ls | wc –l

uws@tux>find /home/uws/Documents –type f | wc -l
```

Nach leeren Dateien suchen und nach einer Bestätigung werden sie gelöscht.

```
uws@tux>find / -empty –ok rm “{}” \;
```

Nach Dateien oder Verzeichnissen suchen, die nicht dem Suchkriterium entsprechen. Die Option `–regex` berücksichtigt den kompletten Pfad, wobei `–name` nur den Namen berücksichtigt ohne den Pfad.

```
uws@tux>find / \! –regex “.*spool.*” –empty –ok rm –rf “{} \;
```

Dateien finden, die innerhalb der letzten 120 Minuten verändert wurden. Ohne das “-“ vor der Angabe der Zeit (120), wird alles gefunden, was vor 2 Stunden geändert wurde. Mit einem “+“ werden Dateien gefunden, die vor der angegebenen Zeit verändert worden sind.

```
uws@tux>find /etc /usr –mmin -120

uws@tux>find ~ -mmin +120 mmin -1440
```

3.4.2 Löschen

Dateien können mit dem Befehl rm gelöscht werden. In dem nachfolgenden Beispiel werden alle Dateien gelöscht, die älter als 10 Tage sind.

```
uws@tux>find /daten –type f \( -name ’*.jpg’ \) –ctime +10 –exec rm {} \;
```

3.4.3 Vergleichen

```
uws@tux>diff <datei1> <datei2>

uws@tux>diff3 <datei1> <datei2> <datei3>
```

3.4.4 Nur neuere Dateien kopieren

Mit dem Befehl `cp` und der Option `–u` kann man Dateien kopieren, die ein neueres Datum haben als die auf dem Zielverzeichnis.

```
uws@tux>cp –dpRux /home /mnt/media/home
```

3.4.5 Anzahl Dateien

Möchte man die Anzahl der Dateien in einem Verzeichnis ausgeben, so kann man das mit der Kombination von `ls` und `wc` erledigen.

```
uws@tux>ls | wc –l
```

3.4.6 Dateien anlegen / Datum aktualisieren

Eine leere Datei wird mit dem Befehl `touch` angelegt. Dieser Befehl wertet die `umask` aus und legt die neue Datei mit den Berechtigungen an. Ebenso kann man mit `touch` einer vorhanden Datei das aktuelle Datum mit Uhrzeit verpassen.

```
uws@tux>touch empty.txt

uws@tux>touch *.txt
```

3.4.7 Dateien nach Größe / Zeit anzeigen

Mit der Option `–S` werden die Dateien nach Größe absteigend angezeigt. Mit der zusätzlichen Option `–r` wird die sortierreihenfolge umgedreht. In dem ersten Beispiel werden die 15 Größten Dateien angezeigt und im zweiten Beispiel wird nach der Zeit sortiert.

```
uws@tux>ls –lahS | head –n 15

uws@tux>ls -lahtr
```

3.4.8 Status Informationen

Mit dem Befehl `stat` kann man sich die Status Informationen einer Datei / Filesystem anzeigen lassen.

```
uws@tux>stat myfile.txt
  File: ‘myfile.txt‘
  Size: 337           Blocks: 8       IO Block: 4096  regular file
Device: fc03h/64515d  Inode: 1048633  Links: 1
Access: (0644/-rw-r—r--) UID: ( 502/  uws)  GID: ( 502/ uws)
Access: 2016-04-18 09:08:18.605079663 +0200
Modify: 2014-01-22 11:48:29.896951359 +0200
Change: 2014-01-22 11:48:29.909951675 +0200
```

Möchte man sich Informationen über ein Device anzeigen lassen, so gibt man die Option `–f` mit an. Wird die Option nicht mit angegeben, so werden die Informationen als normales File ausgegeben.

```
uws@tux>stat –f /dev/sda1
  File: “/dev/sda1“
    ID: 0        Namelen: 255     Type: tmpfs
Block Size: 4096       Fundamental block size: 4096
Blocks: Total: 4641806    Free: 4641760    Available: 4641760
Inodes: Total: 4641806    Free: 4641090
```

Möchte man sich z.B. nur die Informationen über die Group ID des Owner ausgeben lassen. So gibt man die Option `--format=%g` an. In der nachfolgenden Liste gibt es eine Übersicht der Format Ausgaben.

Format	**Beschreibung**
%a	Zugriffsrechte in Octal
%A	Zugriffsrechte in Human form
%b	Anzahl der belegten Blöcke
%B	Größe in Bytes
%C	SELinux security
%d	Device Nummer in Dezimal
%D	Device Nummer in Hex
%f	Raw Mode in Hex
%F	File Type
%g	Group ID vom Owner
%G	Group Name des Owner
%h	Nummer der hard links
%i	Inode Nummer
%n	Datei Name
%N	Quoted Datei Name
%o	I/O Block Größe
%s	Total size, in Bytes
%t	Major device type in Hex
%T	Minor device type in hex
%u	User ID des Owners
%U	User Name des Owners
%x	Datum / Zeit des letzten Zugriffs
%X	Datum / Zeit des letzten Zugriffs in Sekunden
%y	Datum / Zeit der letzten Modification
%Y	Datum / Zeit der letzten Modification in Sekunden
%z	Datum / Zeit des letzten Change

```
uws@tux>stat --format=%x myfile.txt
2014-01-22 11:48:29.896951359 +0200
```

Anstelle von `--format` kann auch ein `–c` genommen werden.

```
uws@tux>stat –f –c%n /dev/sda1
/dev/sda1

uws@tux>stat –f –c%c /dev/sda1
4644806
```

3.5 Programme

3.5.1 Im Hintergrund ausführen

Um ein Programm im Hintergrund laufen zu lassen, so wird nach dem Befehl das & Zeichen angehängt.

```
uws@tux>tail –f /var/log/messages &
```

Hat man mehrere Programme im Hintergrund gestartet, so kann man sie sich mit dem Befehl `jobs` anzeigen lassen.

```
uws@tux>jobs
```

Einen Job wieder in den Vordergrund zu holen, so wird der Befehl `fg` abgesetzt. Sind mehrere Jobs gestartet worden, so wird an dem Befehl `fg` noch ein `%<job_nr>` nagehängt, um einen speziellen Job in den Vordergrund zu holen.

```
uws@tux>fg

uws@tux>fg %2
```

Soll das Programm wieder in den Hintergrund laufen, so muss das Programm gestoppt werden, um ihn dann in den Hintergrund zu bringen.

```
uws@tux>ctrl Z

uws@tux>bg
```

3.5.2 Ausgabe des Programmpfads

Möchte man die Pfadangabe eines Programms sich anzeigen lassen, so gibt es zwei Möglichkeiten. In der ersten wird die ${PATH} Variable ausgewertet.

```
uws@tux>which <ProgrammName>

uws@tux>whereis <ProgrammName>
```

3.6 Programme in einer Sandbox ausführen

Möchte man Programme oder auch Dienste in einem geschützten Bereich ausführen, kann man sie in einem chroot-Käfig starten. In diesem Käfig können Angreifer keinen Schaden anrichten.

```
uws@tux>chroot /mimm/das/als/root ftpd –o option1
```

3.7 Unveränderliche Dateien

Für die Dateisysteme EXT2 und EXT3 gibt es ein zusätzliches Attribute, die Dateien unveränderlich machen, solange das Attribute gesetzt ist. Mit dem Befehl `lsattr` kann man sich das Attribute anzeigen lassen und der Befehl `chattr` mit dem Attrinute `–i` setzt das Attribute.

```
uws@tux>lsattr *.txt

uws@tux>chattr –i *.txt
```

3.8 Eine For Schleife

Eine `for` Schleife wird gerne für Dateioperationen genommen. Die `for` Schleife kann auch in Scripten genommen werden. In dem nachfolgenden Beispiel werden alle `sh` Scripte kopiert und an den Dateien wird ein .bak angehangen.

```
uws@tux>for I in *.sh; do cp $i $i.bak; done
```

Enthalten die Dateinamen Leerzeichen, so muss anstelle des * das @-Zeichen genommen werden.

```
uws@tux>cat demo.sh
#!/bin/bash
for i in “$@“; do
  printf “\t- $i -\n”
done

uws@tux>demo.sh Apfel “harte Birne”
- Apfel –
- harte Birne -
```

Siehe auch Abschnitt **10.6.5**.

3.9 Mails

3.9.1 Konfiguration

Lokale Mails können ohne Konfiguration verschickt werden. Sollen die Mail aber an einem Mail Server übergeben werden, so ist dieses zu konfigurieren. Folgende Parameter müssen in der Datei `/etc/postfix/main.cf` verändert werden.

```
myhostname = tux.seabaer-ag.de
mydomain = seabaer-ag.de
myorigin = $mydomian
relayhost = [172.30.250.66]

roo@tux>/etc/init.d/postfix restart
Postfix beenden:        [ok]
Postfix starten:        [ok]
```

Soll der Absender umbenannt werden, so sind folgende Einstellungen vorzunehmen.

```
root@tux>tail –n 4 /etc/postfix/main.cf
#
# translating sender e.g. “root@tux” to “uws@seabaer-ag.de”
#
Sender_canonical_maps = /etc/postfix/sender_canonical

root@tux>cat /etc/postfix/sender_canonical
#
# translating user root to uws@seabaer-ag.de
#
root  uws@seabaer-ag.de

root@tux>postmap /etc/postfix/sender_canonical

root@tux>/etc/init.d/postfix restart
```

3.9.2 Verschicken

Mails kann man mit dem Befehl `mail` verschicken.

```
uws@tux>echo “Eine Mail für dich“ | mail –s “Du hast Post“ uws@localhost

uws@tux>cat mail.txt | mail –s “Mehrzeileige Mail“ uws@localhost
```

Mit `–a` kann man Attachements verschicken. Wenn mehrere Attachements verschickt werden sollen, kann man die Option `–a` mehrmals angeben. Verschiedene Empfänger werden mit einem Komma getrennt angegeben.

```
uws@tux>mail –s “Anhang“ –a Image1.jpg –a bhb.pdf uws@seabaer-ag.de,
jan@seabaer-ag.de
```

3.10 Grep

3.10.1 In Farbe

Soll die Ausgabe von `grep` in Farbe passieren, so gibt man die Option `---color=auto` mit an. Dauerhaft kann man diese Option in der `.bashrc` einschalten.

```
uws@tux>grep --color=auto

uws@tux>cat .bashrc

export GREP_OPTIONS='--color=auto'
export GREP_COLOR='1;32' # hellgrün, ANSI Farbcode
```

3.10.2 Klein- / Großschreibung

Möchte man alle Wörter finden, egal, ob Klein- oder Großschreibung, so wird hierzu der Parameter `–i` angegeben.

```
uws@tux>grep –i "rman-08" /home/uws/log/rman_level0.log
RMAN-0815 *********************************
rman-0816  Erorrs found
```

3.10.3 Exakte Suche

Soll nur nach dem genauen Wort gesucht werden, so gibt man die Option `–w` an.

```
uws@tux>grep –w "rman-08" /home/uws/log/rman_level0.log
rman-0816  Erorrs found
```

3.10.4 Mehrere Wörter suchen

Nach mehreren Wörtern kann man mit `egrp` suchen. Die Angabe der Option `–w` ist optional.

```
uws@tux>egrep –w "RMAN-08|rman-08" /home/uws/log/rman_level0.log
RMAN-0815 *********************************
rman-0816  Erorrs found
```

Mit grep kann man die Optionen –E oder –e nehmen. Egrep ist ein grep mit der Option –E. Gibt man ein .* in dem Suchstring ein, so ist es wie ein und. Ein oder wird mit einem Pipe gemacht. Nachfolgend sind einige Beispiele aufgelistet.

```
uws@tux>grep –i "Manager\|Sales" employee.txt
100 Thomas Manager  Sales     $5,000
400 Willi  Manager  Marketing $2,500

uws@tux>grep –i –E "Manager|Sales" employee.txt

uws@tux>egrep –i "manager|sales" employee.txt

uws@tux>grep –e "Manager" –e "Sales" employee.txt
```

```
uws@tux>grep –E “Manager.*Sales“ employee.txt
100 Thomas Manager  Sales     $5,000
```

3.10.5 Zeilen Ausgabe

Mit der Option –n wird auch die Zeilen Nummer mit ausgegeben.

```
uws@tux>grep –n “RMAN-08” /home/uws/log/rman_level0.log
2:RMAN-0815 ********************************
```

3.10.6 Anzahl Übereinstimmungen

Möchte man nur die Anzahl der gefundenen Übereinstimmungen angezeigt bekommen, so nimmt man hierfür die Option -c.

```
uws@tux>grep –c “RMAN-08” /home/uws/log/rman_level0.log
1
```

3.10.7 Alles ausgeben

Mit –v wird alles ausgegeben, außer dem Suchstring.

```
uws@tux>grep –v “RMAN-08” /home/uws/log/rman_level0.log
Recovery Manager: Release 11.1.0.6.0 – Production on Mon Apr 29 19:00:00
2013
Copyright © 1982, 2007 Oracle. All rights reserved
```

3.10.8 Ausgabe Datei

In welchen Dateien sich der Suchstring befindet, wird mit der Option –l ermittelt.

```
uws@tux>grep –l “RMAN-08” *.log
/home/uws/log/rman_level0.log
/home/uws/log/rman_level1.log
```

3.11 Kalender

In der Bash kann man sich einen Kalender anzeigen lassen mit dem Befehl `cal`. Cal reicht zurück bis zum 1. Januar 0001.

```
uws@tux>cal
    August 2010
So Mo Di Mi Do Fr Sa
 1  2  3  4  5  6  7
 8  9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31
```

Gibt man dem Befehl `cal` noch den Parameter `–y` auf dem Weg, so wird das aktuelle Jahr ausgegeben. Mit `cal 06 2005` wird der Juni 2005 ausgegeben. Die Angabe `cal -3` gibt den aktuellen Monat mit dem vorherigen und nachfolgenden Monat aus. Bei dem Parameter `–j` werden die Tage von 1 bis 365 gezählt und ausgegeben.

3.12 Links

3.12.1 Anlegen

Links (Verweise) können mit dem Befehl `ln` angelegt werden. Ohne Optionen werden Hardlinks angelegt und mit der Option `–s` Softlinks.

```
uws@tux>ln <quelle> <ziel>

uws@tux>ln –s <quelle> <ziel>
```

Ein `Softlink` (Symbolischer Link) zeigt direkt auf die Datei / Verzeichnis und ein `Hardlink` auf einen Inode. Wenn die Datei umbenannt oder verschoben wird, so bekommt der Softlink davon nicht mit, der Hardlink schon. Softlinks können Partitions- und Dateisystem übergreifend erstellt werden, Hardlinks nicht. Hardlinks auf Verzeichnisse sollten vermieden werden, sind aber grundsätzlich möglich durch den User root.

3.12.2 Anzeigen

```
uws@tux>ls –l
MeineDatei.pdf > /home/uws/Daten/Bash.pdf

uws@tux>ls –l | grep .-\>
```

3.12.3 Löschen

```
uws@tux>rm <ziel>
```

3.13 Größe und Lage der Konsole

Möchte man die Größe und Lage der Konsole in Erfahrung bringen, so gibt es hierfür den Befehl `xwininfo`, den man in der Konsole absetzt. In der Ausgabe `geometry` steht dann die Größe und die Lage des Fensters. Die Lage des Fensters wird von der linken oberen Ecke aus gesehen. Diese Werte können dann z.B. für die Konfiguration eines Gnome Konsolen Fenster gebraucht werden. Hierzu ist dann in der Konfiguration des Fenster unter `Befehl` der Wert `–geometry 100x150+10+10` einzugeben.

```
uws@tux>xwininfo
.
.
- geometry 100x150+10+10
```

3.14 Suchen & Ersetzen

3.14.1 SED

In einer Datei kann man mit dem Befehl `sed` nach Texten suchen und ersetzten. In dem nachfolgenden Beispiel wird er String `domain` durch `domäne` ersetzt und durch die Umleitung in eine neue Datei geschrieben.

```
uws@tux>sed –e s/domain/domäne/g <datei> > <datei.neu>
```

Syntax Beispiele

```
uws@tux>sed Kommando Textdatei
uws@tux>sed Kommando < Textdatei
uws@tux>Programm | sed Kommando | …
uws@tux>sed –f Skript …
uws@tux>sed –e Kommando1 –e Kommando2 –e Kommando3 …
uws@tux>sed Kommando > Zieldatei
uws@tux>sed Kommando > Zieldatei 2>&1
```

Sed-Optionen

Option	**Beschreibung**
-e	Angabe auszuführender Befehle
-u	Auf Datenpufferung verzichten
-s	Dateien separat behandeln
-r	Erweiterte reguläre Ausdrücke verwenden
-i [Endung]	Sicherungsdatei anlegen
-f <script>	Skript-Datei ausführen
-n	Unterdrücken der nicht betroffenen Textbereiche
-v	Versionsabfrage

Möchte man mehrere Kommandos in einem Rutsch ausführen, so kann man die Kommandos in einer Datei schreiben und diese Datei wird dann mit `–f` übergeben. In dieser Datei steht dann in jeder Zeile ein Sed-Befehl.

```
uws@tux>cat myFile
s/Hans//
s/ha/Ha/g
```

Editierkommandos

Kommando	Beschreibung
i	Anfügen von Zeilen oberhalb der angegebenen Stelle
a	Anfügen von Zeilen unterhalb der angegebenen Stelle
p	Ausgabe der angegebenen Zeilen
l [länge]	Ausgabe der angegebenen Zeilen optional mit Längenbegrenzung
y	Austausch von Zeichen
q	Sed beenden
c	Ersetzen von Text in der angegebenen Zeile
d	Löschen der angegebenen Zeilen
s	Suchen und ersetzen

Optionen für s

Option	Beschreibung
g	Befehl auf alle gefundenen Textstellen in der Zeile anwenden
p	Ergebnis der Aktionen ausgeben
w[Datei]	Ergebnis in eine Datei schreiben

Optionen der Editierkommandos

Option	Beschreibung
=	Ausgabe der Zeilennummer
g	Betrifft alle Vorkommen
p	Gibt bei dem Editierkommando s die geänderte Zeile aus
w	Schreiben der bearbeiteten Zeilen in die Datei

Sonderzeichen

Zeichen	Beschreibung
(	Öffnet eine Anweisung
)	Schließt eine Anweisung
{	Öffnet optionale Anweisung
}	Schließt optionale Anweisung
[	Öffnet Klassenbeschreibung von Zeichen
]	Schließt Klassenbeschreibung von Zeichen
“	Maskiert eine Anweisung und löst Shell-Variablen aus
‘	Maskiert eine Anweisung und löst Shell-Variable nicht auf
`	Schließt Anweisungsblock ein
.	Ein beliebiges Zeichen außer Zeilenvorschub
,	trennt Parameter, etwa Zeilenangaben
Leerzeichen	Setzt Markierung (t- und b-Befehl)
$	Dokumentenende, letzte Zeile oder Zeilenende
&	Platzhalter für das Suchmuster, das in der Ersetzen-Anweisung mit ausgegeben wird.
|	Oder (Abtrennen von regulären Ausdrücken)
/	Trennzeichen in Editierkommandos
^	Anfang der Zeile, aber: [^Begriff] = Negierung
\	Demaskieren
!	Nach Zeilenzahl: nicht diese Zeile ausgeben
*	Nie oder beliebig oft
+	Muster mindestens einmal vorhanden
=	Ausgabe der Zeilennummern
\n	Zeilenvorschub
\t	Tabulator

Suchmuster und Adressangaben

Muster	**Beschreibung**
(ohne)	Alle Zeilen
25	Zeile 25
25!	Nicht Zeile 25
10,20	Zeilen 10 bis 20
$	Letzte Zeile
‘/Muster/!‘	Nicht Suchmuster
^Zeichen	Zeichen am Anfang
/Zeichenkette/	Zeichenkette
[Zeichenmenge]	Zeichenmenge
[:alpha:]	Beliebige Buchstaben
[:lower:]	Kleinbuchstaben
[:upper:]	Großbuchstaben
[:alnum:]	Alphanumerische Zeichen
[:digit:]	Zahlen
[:xdigit:]	Hexadezimalzahlen
[:blank:]	Tabulatoren und Leerzeichen
[:space:]	Leerzeichen
[:cntrl:]	Steuerzeichen
[:print:]	Druckbare Zeichen (ohne Steuerzeichen)
[:graph:]	Sichtbare Zeichen (ohne Leerzeichen)
[:punct:]	Satzzeichen

Weitere Beispiele

Beispiel	**Beschreibung**
cat textdatei | sed –n ‘/Hans/p’	Sucht nach Hans
sed –n ‘/[Hh]an/p’ textdatei	Alle Namen, die “han” oder “Han” beinhalten
sed –n ‘3,5!‘p textdatei	Alle Zeilen außer 3 und 5
sed –n ‘/Hans/!‘p textdatei	Alle Zeilen, außer die mit Hans
sed –n ‘/[H|G]/‘p textdatei	Zeilen, die „H“ oder „G“ enthalten
sed –n ‘/[H]|[G]/!‘p textdatei	Zeilen, die „H“ oder „G“ nicht enthalten
cat textdatei | sed –n ‘3p‘	Zeile 3
cat textdatei | sed –n ‘$p’	Letzte Zeile
sed –n ‘/[H]./,/[J]./!‘p textdatei	Zeilen nicht ausgeben, die „H“ oder „J“ enthalten
cat textdatei | sed –n ‘/[alnum:]/‘p	Alle Zeilen, die alphanumerische Zeichen enthalten
cat textdatei | sed –n ‘s/j/J/p‘	Suchmuster nur beim ersten Auftreten ersetzten
cat textdatei | sed –n ‘s/j/J/gp‘	Suchmuster bei jedem Auftreten ersetzten
sed –n ‘s/Hans//gp‘ textdatei	Löschen des Wortes “Hans”
cat textdatei | sed –n ‘4s/Hans/Jens/gp’	Ersetzen von “Hans” durch “Jens” in der Zeile 4
sed –n ‘/ans/s09/089/gp‘ textdatei	Ersetzen von “09” durch “089” bei allen Zeilen mit dem Suchmuster “ans”
sed –n ‘/ans/!s0/089/gp‘ textdatei	Ersetzen von “0” durch “089” bei allen Zeichen, die nicht “ans” enthalten
cat textdatei | sed –n s‘/[0-9]\/]//’gp	Löschen aller Zahlen mitsamt Schrästrichen

3.14.2 TR

Es gibt auch noch den Befehl tr, mit dem man Strings ersetzten kann. Achtung, der zu ersetzende String muss gleich lang sein, sonst werden die fehlenden Stellen aufgefüllt.

Konvertieren Klein- auf Großschreibung

```
uws@tux>tr a-z A-Z < datei.old > datei.neu

uws@tux>tr [:lower:] [:upper:] < datei.old > datei.neu
```

Ersetzen von Strings

```
uws@tux>tr ‘{}‘ ‘()‘ < datei.old > datei.neu
```

Konvertieren Leerzeichen zu Tab. Für jedes Leerzeichen wird ein Tab gesetzt. Sind zwei oder mehr Leerzeichen vorhanden, so gibt es auch die gleiche Anzahl an Tabs.

```
uws@tux>echo “Zeile mit Leerzeichen“ | tr [:space:] ‘\t‘
Zeile       mit         Leerzeichen
```

Soll nur für ein Tab gesetzt werden, auch wenn mehrere Leerzeichen vorkommen, so gibt man die Option –s mit an.

```
uws@tux>echo “Zeile mit Leerzeichen“ | tr –s [:space:] ‘\t‘
Zeile       mit         Leerzeichen
```

Setzen von nur einem Leerzeichen

```
uws@tux>echo “Zeile   mit    viele    Leerzeichen“ | tr –s [:space:] ‘ ‘
```

Löschen von Strings

```
uws@tux>echo “Die User ID ist 784592“ | tr –d ‘D‘
ie User I ist 784592

uws@tux>echo “Die User ID ist 784592“ | tr –d [:digit:]
Die User ID ist

uws@tux>echo “Die User ID ist 784592“ | tr –cd [:digit:]
784592
```

Löschen aller non-printable Zeichen

```
uws@tux>tr –cd [:print:] < file.txt > file_new.txt
```

3.15 AWK - Spaltenweise Operationen

Das Programm awk arbeite Spaltenweise mit organisierten Informationen.

```
uws@tux>df | awk ‘/dev/ {summe +=$4} END {print summe “ Kb“}‘
```

3.16 CUT - Spalten extrahieren

Mit dem Kommandozeilen Befehl `cut` können Spalten ausgelesen werden. In dem ersten Beispiel wird aus einer Datei das zweite Zeichen ausgelesen. Einen Bereich wird mit `–c2-4` ausgelesen.

```
uws@tux>cut –c2 uws.txt
w
l
e
```

Ab einer Position bis zum Ende der Zeile wird folgendermaßen ausgelesen.

```
uws@tux>cut -c2- uws.txt
we is not working
lisabeth is at home
ernd has a car
```

Vom Anfang einer Zeile bis zur einer Bestimmten Position wird mit z.B. `-8` angegeben. Wird nur das `Minus` Zeichen angegeben, so wird die ganze Zeile ausgegeben.

```
uws@tux>cut –c-6 uws.txt
Uwe is
Elisab
Bernd
```

Die Option `–b` ist wie die Angabe von `–c`.

```
uws@tux>ls –l ~/bin | cut –b 1-11,56-
```

Verschiedene Felder warden mit der Option `–f` ausgelesen und mit `–d` kann man das Trennzeichen setzten. Einen Bereich kann mit `–f 1-3,6,7` angegeben werden.

```
uws@tux>uname -a | cut –d ‘ ‘ –f 2,3
tux 3.1.10-1.13-desktop

uws@tux>grep –i “nfs“ /etc/fstab | cut –d‘ ‘ –f2
/data/mail
/data/pictures
/data/video
```

-d = Trennzeichen
-f = Spalte auslesen

Alle Felder ausgeben, außer ein bestimmtes Feld

```
uws@tux>grep “/bin/zsh“ /etc/passwd | cut –d ‘:‘ --complement –s –f7
uws:x:1000:1000:uws:/home/uws
```

Für die Ausgabe ein neues Trennzeichen definieren. Ein Zeilenvorschub wird mit `$‘\n‘` gemacht.

```
uws@tux>grep “/bin/zsh“ /etc/passwd | cut –d ‘:‘ –s –f1,6,7 –-output-
delimiter=’#’
uws#/home/uws#/bin/zsh
```

3.17 DOS / Linux Dateien konvertieren

Das Zeilenende von DOS-Dateien wird in das Linux Format konvertiert und das Zeilenende von Linux-Dateien in das DOS Format..

```
uws@tux>fromdos <datei>

uws@tux>todos <datei>
```

3.18 Dateianfang anzeigen

Anzeige der ersten <n> Zeilen einer Datei.

```
uws@tux>head -10 /var/log/messages
```

3.19 Sortieren

Die Ausgabe wird sortiert.

```
uws@tux>ls –l ~/bin | sort +4
```

3.20 Rechnen

Mit dem Aufruf `expr` kann in der Shell auch gerechnet werden.

```
uws@tux>expr 5 \* 600
3000
```

3.21 Umgebungsvariablen anzeigen

Die Umgebungsvariablen können mit dem Aufruf `printenv` angezeigt werden.

```
uws@tux>printenv
```

3.22 Datei Abgleich

Sollen zwei Dateien auf dem gleichen Stand gebracht werden, so erstellt man als erstes eine Differenz Datei und die so erstellte Datei wird dann in die erste Datei mit `patch` eingefügt.

```
uws@tux>diff datei1 datei2 > d1_d2.diff

uws@tux>patch datei1 < d1_d2.diff
patching file datei1
```

3.23 Dateiinhalt ausgeben

Mit dem Befehl `cat` wird der Inhalt einer Datei von oben nach unten ausgegeben. Möchte man die Ausgabe von unten nach oben haben, so gibt es hierfür den Befehl `tac`.

```
root@tux>tac /var/messages | less
```

3.24 Prüfsummen

Eine Prüfsumme für eine Datei kann man mit den Befehlen `md5sum`, `sha1sum`, `sha256sum` oder `sha512sum` erstellt werden.

```
uws@tux>md5sum spiele.iso
```

Den Inhalt einer Variablen kann man folgendermaßen die Prüfsumme ermitteln.

```
uws@tux>echo “$uws” | sha1sum

uws@tux>uws1=`echo “$uws” | sha1sum`

uws@tux>echo ${uws1%-}  # Ausgabe Prüfsumme ohne abschließenden ‘-‘
```

3.25 Programmschleife

Möchte man ein Programm in einer Schleife ausführen, so kann man das mit dem Befehl `watch` machen. Dieses Programm führt den übergebenen Befehl jede Sekunde aus und gibt ihn auf der Konsole aus. Mit `–n` kann die Intervall Zeit mitgegeben werden.

```
uws@tux>watch –n 2 cat /proc/meminfo
Every 2,0s: cat /proc/meminfo                    Wed Oct 29 14:01:22 2014

MemTotal:       4055224 kB
MemFree:        2581004 kB
.
.
```

Gibt man noch die Option `–d=cumulative` mit an, so werden alle Änderungen an der Ausgabe grau hinterlegt. Ohne die Option `cumulative` werden nur die Änderungen grau markiert, sie sich nach dem letzten Aufruf geändert haben.

Soll der auszuführende Befehl mit einem Pipe Zeichen weiterverarbeitet werden, so muss die Befehlzeile in `‘‘` stehen.

```
uws@tux>watch -d ‘ls –l | grep uws’
```

3.26 Klammerexpandierung

Man kann immer nur Zahlen oder Buchstaben expandieren

```
uws@tux>echo {A..Z}
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

uws@tux>echo {1..12}
1 2 3 4 5 6 7 8 9 10 11 12

uws@tux>echo {1..12..2}
1 3 5 7 9 11

uws@tux>echo {a,b,c}{d,e,f}
ad ae af bd be bf cd ce cf

uws@tux>echo {a{1..3},b,c}{d,e,f}
a1d a1e a1f a2d a2e a2f a3d a3e a3f bd be bf cd ce cf

uws@tux>ls –l {,/local}/usr{,/share} # entspricht
ls –l /local/usr
ls –l /local/usr/share
ls –l /usr
ls –l /usr/share

uws@tux>ls –l a?.sh
-rw-rw-r--  1  uws  users  0 Sep 9  12:43 a1.sh

uws@tux>ls –l [ab]*
-rw-rw-r--  1  uws  users  0 Sep 9  12:43 a1.sh
-rw-rw-r--  1  uws  users  0 Sep 9  12:43 b1.sh

uws@tux>ls –l [a-c]*
-rw-rw-r--  1  uws  users  0 Sep 9  12:43 a1.sh
-rw-rw-r--  1  uws  users  0 Sep 9  12:43 b1.sh
-rw-rw-r--  1  uws  users  0 Sep 9  12:43 c1.sh

uws@tux>ls –l [^ab]*
-rw-rw-r--  1  uws  users  0 Sep 9  12:43 c1.sh

uws@tux>echo a?.sh
a1.sh

uws@tux>echo *
a1.sh b1.sh c1.sh
```

4. Dateisystem

4.1 Logical Volume (LVM) Informationen

Anzeige der physical Volumes Informationen . Mehr zu diesem Thema gibt es in der Dokumentation `LogicalVolumeManager`.

```
uws@tux>pvscan

uws@tux>pvdisplay /dev/hdb
```

Anzeige der Volume Group Informationen

```
uws@tux>vgscan

uws@tux>vgdisplay /dev/volg1
```

Anzeige der Logical Volume Informationen

```
uws@tux>lvscan

uws@tux>lvdisplay /dev/volg1/system
```

4.2 Festplattenplatz

Die Belegung des Festplattenplatzes kann man mit dem Befehl `df` abfragen. Mit dem Befehl `du` wird die Größe des Verzeichnisses angezeigt.

```
uws@tux>df –h
Dateisystem       Größe Benutzt Verf.  Verw% Eingehängt auf
/dev/mapper/VG1   150GB    63GB  84GB   43%  /
/dev/sda1           99M     14M   85M   15%  /boot

uws@tux>du –sch
2,8M  .
2,8M  insgesamt

uws@tux>du –sch * --exclude=‘u0*‘ –exclude=‘home‘
```

4.3 Dateisystem konvertieren

Um ein vorhandenes `EXT2` Dateisystem nach `EXT3` zu konvertieren, so führt man den Befehl `tune2fs` mit der Option `–j` aus. Nach einem Neustart steht das neue Dateisystem zu Verfügung.

```
uws@tux>tune2fs –j /dev/hda2
```

Ein vorhandenes `EXT3` Dateisystem läst sich auch nach `EXT2` konvertieren. Auch hier steht das neue Dateisystem nach einem Neustart zu Verfügung.

```
uws@tux>tune2fs –f –O ^has_journal /dev/hda2
```

4.4 Festplatten überwachen

Moderne Festplatten sind mit der SMART Technologie ausgestattet. Die Informationen kann man unter Linux mit dem Programm `smartctl` abrufen. Für die Überwachung der SMART-Festplatten ist der `smartctld` Daemon zuständig.

```
uws@tux>smartctl –i /dev/had

=== START OF INFORMATION SECTION ===
Device Model:     SAMSUNG SP0411C
Serial Number:    S049J20YC1679
Firmware Version: UU100-05
User Capacity:    40,060,403,712 byted [40.0 GB]
Sector Size:      512 bytes logical/physical
Device is:        Not in smartctl database [for details use: -P showall]
ATA Version is:   7
ATA Standard is:  ATA/ATAPI-7 T13 1532D revision 0
Local Time is:    Fri Jan 20 20:00:05 2012 CET
SMART Support is: Avaible – device has SMART capability
SMART support is: enabled
```

```
uws@tux>smartctl –ia /dev/sda
```

```
uws@tux>smartctl –H /dev/sda

=== START OF READ SMART DATA SECTION ===
SMART overall-health self-assessment test result: PASSED
```

Die Anzahl der Parkvorgänge der Köpfe kann man mit `smarctl` abfragen. Die Festplatten sind zwischen 300.000 bis 600.000 Parkvorgänge ausgelegt. Diesen Wert findet man unter der Tabelle `Vendor Specific SMART Attributes with Thresholds` in der Spalte `193` und hat die Bezeichnung `Load_Cycle_Count`. Der Wert steht dann in der Spalte `RAW_Value`. Handelt es sich um eine `ATA-Festplatte`, so fragt man die Informationen mit der Option `–d ata` ab.

```
uws@tux>smartctl –a /dev/sda

uws@tux>smartctl –d ata –a /dev/sda
```

Die Abfrage des Power Management der Festplatte wird mit dem Programm hdparm durchgeführt. Gibt es als Rückgabewert 128 oder 254, so werden die Köpfe oft in die Parkposition gefahren. Mit dem Wert 255 schaltet man dieses Verhalten ab.

```
uws@tux>hdparm –I /dev/sda | grep “Advanced power management”

uws@tux>hdparm –B 255 /dev/sda
```

Damit dieses auch nach einem Neustart des Rechners gemacht wird, so trägt man die Zeile in der `/etc/rc.local` ein.

4.5 Wechseldatenträger schließen / auswerfen

Wechseldatenträger (DVD, USB) können auch von der Kommandozeile ausgeworfen oder auch geschlossen werden.
Die Syntax für den Befehl `eject` lautet: `eject <option> <gerätedatei/mountpoint>`. Wird nur der Befehl `eject` ausgeführt, so wird das Standardgerät ausgeworfen.

```
uws@tux>eject -v /media/disk

uws@tux>eject -v /dev/sr0
```

Option	Beschreibung
-t	schließen
-v	wortreich
-d	Anzeige Standardgerät

4.6 USB Geräte

Hat man ein USB Gerät (Stick, Festplatte) an den Rechner angeschlossen, so kann man sich mit dem Befehl `lsusb` sich die USB-Ports anzeigen lassen. Ist ein Gerät angeschlossen, so wird er hier aufgelistet, ansonsten gibt es Informationen über die USB Anschlüsse.

```
uws@tux>lsusb
Bus 001 Device 001: ID ld6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 001: ID ld6b:0002 Linux Foundation 1.1 root hub
Bus 001 Device 002: ID 0781:5530 SanDisk Corp. Cruzer U3 4gb SDCZ36
```

4.7 Festplatten / Partitionen anzeigen

Informationen über Festplatten / Partitionen kann man sich mit dem Befehl `fdisk –l` sich anzeigen lassen. Hiermit bekommt man raus, an welchem Device ein USB Gerät hängt und angesprochen werden kann.

```
uws@rux>fdisk -l
```

4.8 Boot Partition kopieren und aktivieren

Soll die Boot Partition auf einer neuen Festplatte kopiert und aktiviert werden, so sind folgende Schritte zu durchzuführen. Als erstes wird der Inhalt von der ersten Partition zu der neuen Partition kopiert.

```
uws@tux>dd if=/dev/sda1 of=/dev/sdb1 [noerror]
```

In der Datei `/etc/fstab` wird die neue Partition hinzugefügt und mit einem Kommentarzeichen am Anfang versehen.

```
uws@tux>vi /etc/fstab
/dev/disk/by-id/scsi-SATA_VBOX_HARDDISK_VB9606f499-part1   /boot   reiserfs
acl,user_xattr 1 2
#/dev/disk/by-id/scsi-SATA_VBOX_HARDDISK_VB1d7291b2-part1   /boot   reiserfs
acl,user_xattr 1 2
```

Nun kann die alte Partition abgehangen werden.

```
uws@tux>umount /boot
```

In der Datei `/etc/fstab` das Kommentarzeichen von der neuen Partition entfernen und die alte Partition mit einem Kommentarzeichen versehen.

```
uws@tux>vi /etc/fstab
#/dev/disk/by-id/scsi-SATA_VBOX_HARDDISK_VB9606f499-part1   /boot   reiserfs
acl,user_xattr 1 2
/dev/disk/by-id/scsi-SATA_VBOX_HARDDISK_VB1d7291b2-part1   /boot   reiserfs
acl,user_xattr 1 2
```

Nun kann die neue Partition angehangen werden.

```
uws@tux>mount /boot
```

Als letztes für die neue Platte die `Bootable` Option einschalten und für die alte Partition die Option ausschalten.

```
uws@tux>cfdisk /dev/sdb

uws@tux>cfdisk /dev/sda
```

4.9 Iso Image mounten

Ein Iso Image kann man unter Linux direkt in das Dateisystem einbinden.

```
uws@tux>mount –o loop <path_to_iso_image> <mount_path>
```

4.10 Filesystem anzeigen

Möchte man sich das verwendetet Filesystem einer Partition anzeigen lassen, so kann man das mit dem nachfolgenden Befehl machen. Im zweiten Beispiel wird das Filesystem einer LVM Partition ausgegeben.

```
uws@tux>file –s /dev/sdc1
/dev/sdc1: Linux rev 1.0 ext3 filesystem data, UUID=67…

uws@tux>file –s /dev/dm-1
/dev/dm-1: ReiserFS V3.6
```

4.11 Hdparm

Mit `hdparm` kann man sich Informationen der Festplatten anzeigen lassen.

4.11.1 Festplatte Speed Test

```
uws@tux>hdparm –t /dev/sda

/dev/sda:
 Timing buffered disk reads: 404 MB in  3.00 seconds = 134.62 MB/sec
```

4.12 Loopback-Device

Unter Linux kann man ein Loopback-Device (Container) erstellen, das dann in das System gemounted werden kann. Hierzu wird eine Container Datei erstellt und dann mit dem Loopback-Device verbunden. Die Größenangabe der Datei erfolgt in `kb`. Gibt es im Verzeichnis `/dev` keine `loop*` Einträge, so wird `losetup –f` einmal ausgeführt.

```
uws@tux>mkdir /container

uws@tux>dd if=/dev/zero of=/container/lda bs=1024 count=102400
102400+0 Datensätze ein
102400+0 Datensätze aus
104857600 Bytes (105 MB) kopiert, 0,643267 s, 163 MB/s

uws@tux>losetup –f # create /dev/loop* when not exist

uws@tux>losetup /dev/loop1 /container/lda
```

Nun kann das Loopback-Device formatiert werden.

```
uws@tux>mkfs.ext4 /dev/loop1

uws@tux>mkdir -p /media/container

uws@tux>mount /dev/loop1 /media/container
```

Das Löschen des Loopback-Devices wird mit der Option `–d` vorgenommen.

```
uws@tux>umount /media/container

uws@tux>losetup –d /dev/loop1
```

Damit das Loopback-Device bei einem Neustart automatisch gemounted wird, so ist in der Datei `/etc/fstab` folgendes hinzuzufügen.

```
/dev/loop1 /media/container ext4 defaults,loop 0 0
```

Die Verbundenen Loopback Devices kann man sich mit der Option `–l` anzeigen lassen.

```
uws@tux>losetup –l
NAME        SIZELIMIT OFFSET AUTOCLEAR RO BACK-FILE
/dev/loop1          0      0         0  0 /container/lda
```

4.13 Festplatten Statistic

Mit dem Befehl `iostat` kann man sich den Durchsatz der Festplatten anzeigen lassen. Mit der Option `–N` wird er Durchsatz von LVM Devices ausgegeben. Mit `–m` werden die Werte im MB und mit einer `Zahl` wird die Aktualisierung pro Sekunde angegeben.

```
uws@tux>iostat –N –n 2
```

4.14 Filesystem Read-only Modus

Befindet sich das Filesystem im Read-only Modus, so kann man es mit einem remount wieder im normalen Modus versetzten.

```
root@tux>mount –o remount /
```

4.15 Partitions Informationen

Informationen über Partitionen kann man sich mit `tune2fs` anzeigen lassen.

```
uws@tux>tune2fs –l /dev/sda1
tune2fs 1.42.8 (20-Jun-2013)
Filesystem volume name: <none>
Last mounted on:        /root
Filesystem UUID:        cc9d890c-a85f-4612-beb5-e1c9c203be61
.
.
```

Für Loopback Devises wird `/dev/loop0` und für ein Software Raid mit `/dev/md/md0`.

4.16 Swap Size

Die Größe der Swap Partition kann man mit folgenden Befehlen sich anzeigen lassen.

```
uws@tux>cat /proc/swaps
Filename    Type        Size        Used  Priority
/dev/sda2   partition   2097148     0     -1

uws@tux>swapon –s
Filename    Type        Size        Used  Priority
/dev/sda2   partition   2097148     0     -1

uws@tux>free –m [g]
                 total used  free  shared      buffers     cached
Mem:             3960  3701  258   70          320         2614
-/+ buffers/cache:       766 3193
Swap:            2047  0     2047

uws@tux>[h]top
```

Eine neue Swap Partition kann mit dem Befehl `swapon` eingebunden werden.

```
root@tux>swapon /dev/sda2
```

4.17 Software RAID

Unter Linux kann man ein Software Raid erstellen. Hierzu gibt es das Programm `mdadm`. Für das Raid kann man Festplatten, Partitionen und auch Loopback Devices nehmen.

Mit mdadm kann man folgende Raids erstellen.

Raid Level	Beschreibung
Linear	Mehrere Partitionen aneinander hängen.
Multipath	Ein Mapping einer Datei auf zwei verschiedenen Pfade auf er gleichen Partition.
Faulty	Emuliert ein fehlerhaftes RAID für Testfälle.
0	Stripping, mehrere Platten werden zu einer großen zusammen gefügt.
1	Spiegelung einer Platte
4	Wie Raid 0, aber mit einer zusätzlichen Platte für Paritätsbits.
5	Mindesten 3 Platten werden zusammen gefügt und die Paritätsbits werden auf alle Platten verteilt.
6	Wie Raid 5, nur das zwei unabhägige Paritätsbits geschrieben werden.
10	Kombination von Raid 0 und Raid 1

Eine Auswahl der Optionen wird nachfolgend aufgelistet. Eine vollständige Liste incl. der Beschreibung kann man in der Manpage nachlesen.

Option	Beschreibung
--add	Fügt dem Raid eine Platte / Partition hinzu
--backup-file=	Erstellt ein Backup File für eine Raid Erweiterung
--detail	Array Details ausgeben
--fail	Status des Raids ändern
--force	Die Ausführung erzwingen
--help	Anzeige des Hilfe Textes
--level=	Raid Type
--query	Überprüfen, ob das Device ein MD-Device oder zu einem Raid gehört
--raid-device=	Setzen der Anzahl der Platten in dem Raid
--remove	Löschen einer Platte / Partition aus dem Raid
--stop	Anhalten / Beenden des Raids
--spare-device=	Angabe der Spare Platte
--test	Die angegebenen Optionen testen.
--uuid=	Die UUID des Raids
--verbose	Erweiterte Informationen ausgeben, kann 2 mal gesetzt werden.
--zero-superblock	Löschen des Raid-Superblocks

4.17.1 Raid erstellen

Damit wir das Erstellen von Raid System für diese Beispiele machen können, erstellen wir erstmal 4 Container mit einer Größe von 500MB und erstellen dann die Loopback Devices.

```
uws@tux>for a in ld1 ld2 ld3 ld4; do dd if=/dev/zero of=/container/$a
bs=1024 count=512000
512000+0 records in
512000+0 records out
524288000 bytes (524 MB) copied, 1.39163 s, 377 MB/s
512000+0 records in
512000+0 records out
524288000 bytes (524 MB) copied, 2.11694 s, 248 MB/s
512000+0 records in
512000+0 records out
524288000 bytes (524 MB) copied, 2.10103 s, 250 MB/s
512000+0 records in
512000+0 records out
524288000 bytes (524 MB) copied, 2.34669 s, 223 MB/s

uws@tux>losetup –f # Wenn /dev/loop* nicht vorhanden

uws@tux>for a in 1 2 3 4; do losetup /dev/loop$a /container/ld$a;done
```

Nun können wir Raid Systeme erstellen. Als erstes erstellen wir ein `Raid 0` Sytem.

```
uws@tux>mdadm --create /dev/md/md0 --level=0 --raid-devices=2 /dev/loop1
/dev/loop2
mdadm: Defaulting to version 1.2 metadata
mdadm: array /dev/md/md0 started.
```

Ein Raid System mit einer fehlende Platte, die später hinzugefügt werden kann, wird folgendermaßen erstellt.

```
uws@tux>mdadm --create /dev/md/md0 --level=1 --raid-devices=2 /dev/loop1
missing
mdadm: Note: this array has metadata at the start and
    may not be suitable as a boot device. If you plan to
    store ‚/boot‘ on this device please ensure that
   your boot-loader understands md/v1.x metadata, or use
   --medtadata=0.90
Continue creating array? y
mdadm: Defaulting to version 1.2 metadata
mdadm: array /dev/md/md0 started.

uws@tux>mdadm –-manage /dev/md/md0 --add /dev/loop2
mdadm: added /dev/loop2
```

Ein `Raid 5` System mit einer Spare Platte kann folgendermaßen erstellt werden.

```
uws@tux>mdadm --create /dev/md/md1 --level=5 --raid-devices=3 /dev/loop1
/dev/loop2 /dev/loop3 --spare-devices=1 /dev/loop4
```

Einem vorhandenen Raid System kann man nachträglich auch eine `Spare` Platte hinzufügen. Durch das Hinzufügen einer weiteren Platte / Device, wird diese automatisch als `Spare` eingebunden.

```
uws@tux>mdadm --manage /dev/md/md0 --add /dev/loop2
```

4.17.2 Raid anzeigen

Ein erstelltes Raid System kann man mit `mdadm --detail` angezeigt werden.

```
uws@tux>mdadm --detail /dev/md/md0
/dev/md/md0
        Version: 1.2
  Creation Time: Tue Oct 28 14:12:36 2014
     Raid Level: raid1
     Array Size: 511680 (499.77 MiB 523,96 MB)
  Used Dev Size: 511680 (499.77 MiB 523,96 MB)
   Raid Devices: 2
  Total Devices: 3
    Persistence: Superblock is persistent

    Update Time: Tue Oct 28 14:12:36 2014
          State: clean
 Active Devices: 2
Working Devices: 3
 Failed Devices: 0
  Spare Devices: 1

           Name: tux:md0  (local to host tux)
           UUID: 9dc2cd57:64bf443e:86eea668:95ceb8ad
         Events: 17

  Number   Major   Minor   RaidDevice   State
     0       7        1        0        active sync   /dev/loop1
     1       7        2        1        active sync   /dev/loop2

     2       7        3        -        spare   /dev/loop1
```

Folgende Raid Zustände gibt es.

Zustand	**Beschreibung**
Clean	Normalzustand, es liegen keine Fehler vor.
Degraded	Ein Ausfall liegt vor, eine oder mehrere Platten sind defect.
Resync	Die Sicherungsinformationen werden geprüft und ggf. neu erstellt.
Rebuild	Die Daten werden wieder hergestellt.

4.17.3 Raid löschen

Bevor ein Raid gelöscht werden kann, muss es angehalten werden. Danach muss von jeder Platte / Device der `Superblock` entfernt werden, die die Platte / Device als Raid-Device festlegt. Das Entfernen des Superblocks muss für jede Platte / Device gemacht werden, die dem Raid zugeordnet war.

```
uws@tux>mdadm stop /dev/md/md0
mdadm: stopped /dev/md/md0

uws@tux>mdadm --zero-superblock /dev/loop1
uws@tux>mdadm --zero-superblock /dev/loop2
```

4.17.4 Platte entfernen

Ist eine Platte in dem Raid Verbund defekt, so muss sie mit `--remove` aus dem Raid Verbund entfernt werden.

```
uws@tux>mdadm --manage /dev/md/md0 --remove  /dev/loop3
mdadm: hot removed /dev/loop3 from /dev/md/md0
```

Nach dem Entfernen der defekten Platte und dem hinzu fügen einer neuen Platte, so wird das Raid System wieder hergestellt. Den Fortschritt des Rebuilds kann man sich folgendermaßen anzeigen lassen.

```
uws@tux>mdadm --manage /dev/md/md0 –-add /dev/loop4
mdadm: added /dev/loop4

uws@tux>watch –n 1 cat /proc/mdstat
```

4.17.5 Dateisystem

Ein `Raid1` System kann man einfach formatieren.

```
uws@tux>mkfs –t ext4 /dev/md/md0 # oder mkfs.ext4

uws@tux>mount /dev/md/md0 /media/data
```

Wurde ein Raid 0,5,6 oder 10 erstellt, so sollte das Dateisystem für dieses Raid angepasst werden. Als erstes müssen wir die `Chunk Size` des Raid Systems ermitteln.

```
uws@tux>mdadm –D /dev/md/mdo | grep “Chunk Size“

  Chunk Size : 512K
```

Nun können wir die Parameter per Hand ermitteln.

`block-size` = Die Größe der Dateisystemblöcke in Bytes, heute sin des 4096 Byte (4 Kib).
`stride-Size` = Die Chunk Size wird durch die `block-size` geteilt (512Kib / 4Kib = 128).
`stride-width` = Hier wird die `stride-size` mal der effektiven Partitionen genommen. Bei einem Raid5 System mit 4 Platten (128 *3 = 384)

```
uws@tux>mkfs.ext4 –b 4096 –E stride=128,stripe-width=384 /dev/md/md0
```

4.17.6 Raid erweitern

Als erstes fügen wir eine neue Platte dem Raid System hinzu. Danach können wir es mit `grow` erweitern. In dem Beispiel wurde vorher ein Raid 5 System erstellt mit 3 Devices.

```
uws@tux>mdadm --manage /dev/md/md0 -–add /dev/loop4
mdadm: added /dev/loop4

uws@tux>mdadm     -–grow     -–raid-devices=4     /dev/md/md0     -–backup-
file=/tmp/md0.bck
mdadm: Need to backup 3072k of critical section..
```

Sollte das System bei der Erweiterung abstürzen, so kann es mit dem Backup File fortgesetzt werden.

```
uws@tux>mdadm –-manage /dev/md/md0 -–continue -–backup-file=/tmp/md0.bck
```

Zum Abschluss muss noch das Dateisystem erweitert werden.

```
uws@tux>umount /data # oder /dev/md/md0

uws@tux>fsck.ext4 –f /dev/md/md0

uws@tux>resize2fs /dev/md/md0

uws@tux>mount /data
```

4.17.7 Array wiederherstellen

Ist das Raid nicht mehr funktionstüchtig, weil zwei Platten aus einem Raid 5 System ausgestiegen sind, verwendet man die zwei ersten Platten, um das Raid zu aktivieren. Danach kann man die ausgestiegenen Platten hinzufügen. Dieses Vorgehen wird auch für ein Raid gebraucht, wenn es aus Loop Devices besteht.

```
uws@tux>mdadm --stop /dev/md/md0

uws@tux>mdadm –-assemble /dev/md/mdo /dev/loop1 /dev/loop2 --force

uws@tux>mdadm –-re-add /dev/md/md0 /dev/loop3 /dev/loop4
```

4.17.8 Konfiguration sichern

Die Konfiguration des Raid Systems kann man auf zwei Arten sichern.

```
uws@tux>mdadm --detail --scan >/etc/mdadm.conf

uws@tux>mdadm –-detail –-brief >/etc/mdadm.conf
```

Eine Vollständige Sicherung macht man ohne die Parameter scan / brief.

```
uws@tux>mdadm –detail /dev/md/md0 >>/etc/mdadm_full.conf
```

4.17.9 Tuning Resync / Rebuild

War eine Festplatte defekt und wurde eine neue Platte in das Raid System eingebunden, so findet dnn ein Resync / Rebuild statt. Dieses kann man beschleunigen, in dem man in der Datei `speed_limit_min` einen neuen Wert setzt. Der Standard Wert ist 1000. In der Datei `speed_limit_max` steht die maximale Geschwindigkeit drin. Meisten ist sie auf 200MiB/s gesetzt. Nach einem Neustart des Systems werden die Standardwerte wieder hergestellt.

```
uws@tux>echo 200000 > /proc/sys/dev/raid/speed_limit_min
```

4.18 SSH Mount

Mit dem Programm `sshfs` kann man ein Verzeichnis von einem anderen Computer mounten.

```
uws@tux>sshfs <user>@<server>:[pfad] <mount_point>
```

Mit `fusermount –u` wird der mount Point wieder entfernt.

```
uws@tux>fusermount –u <mount_point>
```

4.19 Badblocks

Überprüfen der Festplatte / Partition nach Badblocks.

```
uws@tux>badblocks –v /dev/sda1
```

Überprüfen mit der Angabe der Block Size, default Wert ist 1024.

```
uws@tux>badblocks –v –b 2048 /dev/sda1
```

Die Ausgabe in einer Datei speichern.

```
uws@tux>badblocks –v –o badblocks.log /dev/sda1
```

4.20 Festplatte sicher löschen

Soll eine Festplatte sicher gelöscht werden, so kann man sie mit nullen überschreiben. Die Angabe des Parameters `conv` ist optional. Mit diesem Parameter wird angegeben, das mit dem überschreiben weiter gemacht werden soll, wenn Festplatten Fehler vorliegen.

```
uws@tux>dd if=/dev/zero of=/dev/sdf bs=1MB conv=noerror
```

Soll die Festplatte mit Zufallszahlen überschrieben werden, so wird `urandom` anstellen von `zero` genommen. Dieses überschreiben kann bei einer 3TB Platte schon eine Woche dauern, da die Geschwindigkeit nur 10-20Mbytes/s ist.

```
uws@tux>dd if=/dev/urandom of=/dev/sdf bs=1MB conv=noerror
```

Schneller geht das überschreiben mit `openssl`. Dieses dauert bei einer 3TB ca. 5 Stunden. `Urandom` erzeugt ein zufälliges Passwort und `openssl` erzeugt den Zufallszahlenstromdurch AES-Verschlüsselung. (Aus der C'T 2/2016 S.161)

```
uws@tux>openssl enc –qes-256-ctr –pass pass:"$(dd if=/dev/urandom bs=128
count=1 2>/dev/null | base64)" –nosalt </dev/zero >/dev/sdf
```

Eine Festplatte kann mit dem Befehl shred mehrfach überschrieben werden. Die Anzahl des überschreibens wird mit `iteration` angegeben. Der Parameter `–v` steht für `verbose`.

```
uws@tux>shred --iterations=7 –v /dev/sdf
```

4.21 BTRFS

4.21.1 Speicherplatz Belegung

Das Kommando `df` gibt für das Dateisystem BTRFS nicht die richtige Menge des belegten Platzes aus. Damit man sich den belegten Platz anzeigen lassen kann, muss das Kommando innerhalb von btrfs angegeben werden.

```
uws@tux>btrfs filesystem df <mount_point>
```

4.21.2 Snapshots anzeigen

Unter `OpenSuSE` und auch in anderen Distributionen gibt es für die Snapshots das Tool `snapper`. Mit `snapper` lassen sich Dateien aus früheren Snapshots wieder herstellen. System-Rollbacks durchführen, Systemänderungen rückgängig machen und Snapshots manuell erstellen / verwalten. Eine GUI für Snapper gibt es unter http://snapper.io.

Die Konfiguration kann man sich folgendermaßen anzeigen lassen.

```
root@tux>snapper list-config
```

Auflisten der angefertigten Snapshots

```
root@tux>snapper list

root@tux>snapper ls
```

```
root@tux>btrfs subvolume /
```

5. Paketmanager

5.1 RPM Paket Manager

Der Paket Manager RPM wurde ursprünglich von Red Hat entwickelt und unter die GNU General Public License (GPL) gestellt. Heute verwenden diesen Paket Manager außer Red Hat noch Open SuSE, Fedora und CentOS. Die erstellten Pakete haben die Endung RPM.

Der Paketmanager kopiert alle im Paket enthaltenden Dateien an die festgelegte Stelle, führt alle notwendigen Scripte aus und aktualisiert dann die RPM Datenbank. Die RPM Datenbank handelt es sich um eine handvoll Dateien auf BDB-Basis. Die Datenbank befindet sich im Verzeichnis `/var/lib/rpm`.

Gibt es bei der Installation des Paketes Abhängigkeiten mit anderen Programmen oder Bibliotheken, so werden diese nicht automatisch aufgelöst. Das muss der Anwender dann selber erledigen.

5.1.1 Paket installieren

Ein Software Paket aus dem Internet, von einer DVD oder von der lokalen Festplatte, wird mit dem Befehl `rpm –i` installiert.

```
uws@tux>rpm –i <paketname>
```

Möchte man bei der Installation des Paketes mehr Informationen erhalten, so gibt man die Optionen –v und –h an. Mit diesen beiden Optionen werden der Status und der Fortschritt angezeigt.

```
uws@tux>rpm –ivh “http://<domainname>/<paketname>“
```

Soll ein Paket installiert werden, ohne vorher die Abhängigkeiten aufzulösen, so gibt man die Option `--nodeps` oder `--force` an. Die dann installierte Software wird höchstwahrscheinlich nicht funktionieren.

```
uws@tux>rpm –i --nodeps <paketname>
```

5.1.2 Paket update

Ist ein Paket installiert und man möchte eine neuere Version updaten, so wird hier die Option –u benutzt.

```
uws@tux>rpm –U <paketname>
```

5.1.3 Paket löschen

Um ein installiertes Paket loszuwerden, gibt man die Option –e an.

```
uws@tux>rpm –e <paketname>
```

Wird versucht ein Paket zu deinstallieren, welche andere Programme brauchen, so scheitert dieser Versuch. Mit der Option `--whatrequires` kann man sich anzeigen lassen, welche Pakete die zu deinstallierte Software benötigen. Diese Pakete gilt es dann vorher zu löschen. Mit der Option `--nodeps` oder `--force` werden die Abhängigkeiten ignoriert und das Paket wird deinstalliert.

```
uws@tux>rpm –e --whatrequires <paktename>
```

5.1.4 Paket Abfrage

Die RPM Datenbank wird mit der Option –q abgefragt.

```
uws@tux>rpm –q <paketname>

uws@tux>rpm –qpl <paketname>

uws@tux>rpm –q --scripts <paketname>

uws@tux>rpm –qa | grep binutil
```

Schalter	Funktion
-a	Alle Pakete anzeigen.
-l	Auflisten aller Dateien im Paket.
-c	Nur die Konfigurationsdateien werden angezeigt.
-d	Anzeige der enthaltenden Dokumenation.
-i	Anzeige der Meta-Informationen.
-p	Abfrage eines nichtinstallierten Paketes.
--scripts	Anzeige der Scriptinhalte.
--changelog	Anzeige einer Pakethistorie.
--whatrequires	Anzeige, welche Abhängigkeiten es gibt.

5.1.5 Abhängigkeiten Abfrage

Welche Abhängigkeiten es für ein Paket gibt, kann man mit einer Abfrage sich anzeigen lassen.

```
uws@tux>rpm –q --whatrequires <paketname>
```

5.1.6 Install Datum

Das Installationsdatum für die Pakete kann man mit der Option `--last` sich anzeigen lassen.

```
uws@tux>rpm –qa –last | grep firefox

firefox-17.0.6-1.0.1.el6_4.x86_64  Do 16 Mai 2013 15:27:29 CEST
```

5.2 Zypper (SuSE)

Mit dem Programm `Zypper` kann man in der Kommandozeile Software installieren, aktualisieren oder auch deinstallieren. Eine Übersicht über die vorhandenen Optionen gibt es mit dem Befehl `zypper --help` oder auch in der `Manpage`. In der Konfigurationsdatei `/etc/zypper/zypp.conf` befinden sich viele hilfreiche Kommentare zu dem Programm.

5.2.1 Paket installieren

Software wird mit der Option `install` installiert. Ein bereits installiertes Paket wird mit der Option `–f` erneut installiert.

```
uws@tux>zypper install <paketname>

uws@tux>zipper install –r “openSuSE-10.3-DVD” <paketname>

uws@tux>zypper –f <paketname>
```

Ganze Programm Gruppen (Zypper Pattern) kann man auch in einem Rutsch installieren.

```
uws@tux>zypper install –t pattern devel_basis
```

Sollen nur der Quellcode eines Programm Pakets installiert werden, so gibt man die Option `srcpackage` an.

```
uws@tux>zypper install –t srcpackage gimp
```

5.2.2 Paket deinstallieren

Ein Programm wird man mit der Option `remove` wieder los.

```
uws@tux>zypper remove <paketname>
```

5.2.3 Paket suchen

Die Suche nach einem Paket wird mit der Option `search` durchgeführt.

```
uws@tux>zypper search <paketname>
```

Man kann nicht nur nach einzelnen Pakten suchen, sondern auch nach ganzen Programm Gruppen. Diese Gruppen werden im Zypper Sprachgebrauch `Pattern` genannt. Bei dem nachfolgenden Beispiel werden alle Pakete gesucht, die der Gruppe `game` angehören.

```
uws@tux>zypper search –t pattern game
```

Um sich vorliegende Patche anzeigen zu lassen, so gibt man die Option `patch` an.

```
uws@tux>zypper search –t patch
```

Installierte Pakete kann man sich mit der Option `–I` anzeigen lassen.

```
uws@tux>zypper search –i | less
```

5.2.4 Paket update

Mit dem Befehl `zypper lu` kann man überprüfen, ob für die installierten Pakete updates vorliegen. Bei diesem Befehl werden nicht nur die Pakete angezeigt, sondern auch die dazugehörenden Paketquellen. Mit `zypper update` kann man dann die Pakete einspielen.

Sollen alle Pakete auf einmal eingespielt werden, so gibt man `zypper dup` ein. Möchte man nur die Pakete updaten, die vom SuSE Team als kritisch eingestuft wurde, so gibt man `zypper up` ein.

```
uws@tux>zypper lu <paketname>

uws@tux>zypper update <paketname>

uws@tux>zypper dup

uws@tux>typper up
```

Mit `zypper update –d` werden die Pakete heruntergeladen, aber nicht installiert. Für die Installation ruft man dann `zypper` update ohne die Option `–d` auf.

```
uws@tux>zypper update –d

uws@tux>zypper update
```

5.2.5 Repositories verwalten

Unter dem Verzeichnis `/var/cache/zypp/raw` befinden sich jeweils in einem eigenen Verzeichnis pro Pepository die Meta-Daten mit den Datei Informationen. Diese Meta-Daten lassen sich mit dem Befehl `zypper refresh` auf den neusten Stand bringen. Außerdem werden im Verzeichnis `/var/cache/zypp/solve` die Informationen abgelegt, welche Abhängigkeiten ein Paket verursacht und wie diese am besten / schnellsten aufzulösen sind. Im Verzeichnis repodata liegen die Meta-Informationen, hierbei handelt es sich um Gzip-Dateien, in denen XML-Dateien sich befinden.

```
uws@tux>zypper refresh
```

Die Nummer eines Repositories wird mit `zypper lr` abgefragt. Mit der Option `–u` werden zusätzlich die Repository-URI angezeigt. Die Option `–p` zeigt die Priority für die Repositories an. Alle Eigenschaften der Repositories können mit der Option `–d` angezeigt werden.

```
uws@tux>zypper lr
# | Alias               | Name                          |Enabled|Refresh
--+---------------------+-------------------------------+-------+--------
1 | Update SuSE 11.4    | Updates openSuSE 11.4         | YES   | YES
2 | openSUSE-11.4       | openSUSE-11.4                 | YES   | No
3 | repo-non-oss        | openSUSE-11.4-non-oss         | YES   | YES
```

Einen `Export` der Eigenschaften der Repositories kann mit der Option `–-export` durchgeführt werden. In dem nachfolgenden Beispiel wird der Export in die Datei backup.repo geschrieben.

```
uws@tux>zypper lr --export /daten/backup/backup.repo
Repositories have been successfully exported to /daten/backup/backup.repo
```

Der `Import` wird mit `ar` und der Angabe der Datei gemacht.

```
uws@tux>zypper ar /daten/backup/backup.repo
```

Einen automatischen Check eines Repositories schaltet man mit `zypper mr –R` aus. Mit der Option `mr –d` schaltet ein Repositories aus.

```
uws@tux>zypper mr –R <repo_nr>

uws@tux>zypper mr –d <repo_name>

uws@tux>zipper mr –d <repo_nr>
```

Soll verhindert werden, dass jede Installation eines Pakets durchgeführt wird, so richtet man eine Paketsperre ein. Mit der Option `ll`, kann man sich alle Paketsperren anzeigen lassen. Zum löschen einer Paketsperre wird die Option `cl` benutzt.

```
uws@tux>zypper al gimp

uws@tux>zypper ll

uws@tux>zypper cl gimp
```

Nach einer Installation löscht der Paketmanager alle Dateien. Möchte man diese aber behalten, so wird dieses mit der Option `mr --kt` eingeschaltet.

```
uws@tux>zypper mr --kt
```

Einen vergebenen Alias Namen kann man mit `renamerepo` ändern.

```
uws@tux>zypper renamerepo repo-source repo-source-new
Repository ’repo-source’ renamed to ’repo-source-new’
```

Ein Repository wird mit `addrepo` hinzugefügt. Nach der Eingabe der URL wird noch ein Alias Namen vergeben.

```
uws@tux>zypper addrepo --name “VLC“ http://download.vnc.org VideoLan
Adding repository ’VLC’ [done]
Repository ’VLC’ successfully added
Enabled: YES
Autorefresh: No
URI: http://download.vcn.org
```

Ein nicht mehr benötigtes Repository kann man mit `removerepo` oder auch `rr` löschen.

```
uws@tux>zypper rr VLC
Removing repository ’VLC’ [done]
Repository ’VLC’ has been removed
```

In der nachfolgenden Tabelle sind alle wichtigen Befehle des Repo-Managers aufgelistet.

Befehl	Kurzschreibweise	Erklärung
zypper repos	zypper lr	Listet die eingerichteten Paketquellen auf.
zypper addrepo <url>	zypper ar <url>	Fügt ein Repository unter dem gewählten Namen hinzu.
zypper removerepo <name>	zypper rr <name>	Entfernt das Repository.
zypper renamerepo <nr> <name>	zypper nr <nr> <name>	Vergibt einem Repository einen neuen Namen.
zypper modifyrepo --enable <name>	zypper mr –e <name>	Aktiviert ein Repository.
zypper modifyrepo --disable <name>	zypper mr –d <name>	Schaltet ein Repository aus.
zypper modifyrepo --refresh <name>	zypper mr –r <name>	Schaltet die automatische Aktualisierung ein.
zypper modifyrepo --no-refresh <name>	zypper mr –R <name>	Schaltet die automatische Aktualisierung aus.
zipper modifyrepo --priority <wert> <name>	zypper mr –r <wert> <name>	Setzt die Priorität des Repos auf den angegeben Wert (kleiner ist höher).

5.2.6 Repository Prioritäten setzen

Über die Priorität wird festgelegt, welche Installationsquelle der Paketmanager nutzen soll. Die Standardeinstellung für die Repositories ist 99. Je höher der Wert ist, desto geringer ist die Priorität. Bieten zwei Repositories dasselbe Paket an, so wird das Paket von der höheren Priorität installiert. Soll zum Beispiel auf das DVD Repository nur dann zugegriffen werden, wenn keine Internetverbindung besteht, so setzt man die Priorität des Repositories auf 100. Aber Achtung, einen kleineren Wert als 99 sollte nicht vergeben werden, da das die Stabilität des System beeinträchtigen kann.

In dem nachfolgenden Beispiel wird für alle lokalen Repositories die Priorität auf 100 gesetzt.

```
uws@tux>zypper mr –p 100 –t –l
```

5.2.7 System Upgrade

Möchte man ein System Upgrade durchführen, also z.B. von OpenSuSE 12.3 nach 13.1, so kann man das mit `zypper` ganz einfach erledigen. Nachfolgend sind die einzelnen Schritte aufgeführt, um ein System Upgrade durchzuführen.

Achtung: Es ist dafür zu sorgen, das ausreichend Platz für das Upgrade auf dem System zu Verfügung steht. Das Verzeichnis `/boot` sollte mindestens `300MB` groß sein. Auch das `/` Verzeichnis sollte über mindestens über `6GB` freien Platz verfügen. Das gleiche gilt auch für `/var`, wenn es sich in einer separaten Partition befindet.

Als erstes wird überprüft, ob das Repo-Update aktiviert ist. Man kann auch den Befehl `zypper lr –uP` nehmen, dann wird zusätzlich noch die Prio ausgegeben.

```
uws@tux>zypper repos –ur
#|Alias      |Name       |Enabled|Refresh|URI
-+-----------+-----------+-------+-------+--------------------------------
1|repo-update|repo-update|yes    |yes    |http://download.opensuse.org/up..
```

Steht in der Spalte `Enabled` bei Repo-Update kein `Yes`, so setzt man in der Konsole folgenden Befehl ab.

```
uws@tux>zypper modifyrepo --enable repo-update
```

Wird kein Eintrag für das Repo-Update angezeigt, wird mit zipper `addrepo` das fehlende Repository hinzugefügt.

```
uws@tux>zypper addrepo --check --name ‘openSUSE-12.3-Update’
http://download.opensuse.org/update/12.3 repo-update
```

Nun können wir das Alte System auf dem letzten Patchlevel heben.

```
uws@tux>zypper refresh

uws@tux>zypper update
```

Nach erfolgtem Update werden die alten Repositories abgeschaltet.

```
uws@tux>zypper modifyrepo --all --disable
```

Jetzt können die neuen Repositories hinzugefügt warden.

```
uws@tux>zypper addrepo --name “openSUSE-13.1 OSS“
http://download.opensuse.org/distribution/13.1/repo/oss repo-13.1-oss

uws@tux>zypper addrepo --name “openSUSE-11.3 Non-OSS“
http://download.opensuse.org/distribution/13.1/repo/non-oss repo-13.1-non-
oss

uws@tux>zypper addrepo –f --name “openSUSE-13.1 Updates OSS“
http://download.opensuse.org/update/13.1 repo-13.1-update-oss

uws@tux>zypper addrepo –f --name “openSUSE-13.1 Updates Non-OSS“
http://download.opensuse.org/update/13.1-non-oss repo-13.1-update-non-oss
```

Nach diesen ganzen vorarbeiten, können wir nun das Upgrade durchführen.

```
uws@tux>zypper clean –a

uws@tux>zypper ref

uws@tux>zypper dup
```

5.2.8 Patche prüfen

Mit dem Kommando `pchk` (patch-check) wird abgefragt, ob für das System neue Patche zu Verfügung stehen.

```
uws@tux>zypper pchk
Loading repository data …
Reading installed packages …
5 patches needed (2 security patches)
```

5.2.9 Benötigte Patche auflisten

Die benötigten Patche können mit der Option `list-patches` oder `lp` angezeigt werden.

```
uws@tux>zypper lp
Loading repository data …
Reading installed packages …

Repository           | Name                  | Version | Category  | Status
---------------------+-----------------------+---------+-----------+--------
Update openSUSE 11.4 | apache2-mod_apparmor | 4985    |recommended| needed
Update openSUSE 11.4 | kdebase4-openSUSE    | 5043    |recommended| needed
Update openSUSE 11.4 | libxcrypt            | 5049    |security   | needed
Update openSUSE 11.4 | man-pages            | 5032    |security   | needed
Update openSUSE 11.4 | valgrind             | 5053    |recommended| needed
```

Mit der Option `lu` werden alle verfügbaren Paketaktualisierungen angezeigt.

```
uws@tux>zypper lu
Loading repository data …
Reading installed packages …
S | Repository          |Name          |Current Vers |Avaible Vers | Arch
--+---------------------+--------------+-------------+-------------+-------
V | Update upenSUSE 11.4|apparmor-docs |2.5.1.r445-52|2.5.1.r445-63| noarch
```

5.2.10 Patche installieren

Die Patche können mit dem Befehl `zypper patch` installiert werden.

```
uws@tux>zypper patch
Loading repository data …
Reading installed packages …
Resolving package dependencies …

The following NEW patches are going tob e installed:
 Apache2-mod_apparmor kdebase4-openSUSE libxcrypt man-pages valgrind

The following packages going tob e upgraded:
 Apparmor-docs apparmor-parser apparmor-profiles apparmor-utils

18 packages to upgrade
Overall download size: 9.3 MiB. After the operation, 8.6 MiB will be freed.
Continue? [y/n/?] (y):
```

5.2.11 Alle Patche auflisten

Alle Patche werden mit der Option `patches` angezeigt.

```
uws@tux>zypper patches
Loading repository data …
Reading installed packages …
Catalog              | Name         |Version | Category    | Status
---------------------+--------------+--------+-------------+-----------
Update openSUSE 11.4 | Mesa          | 4546   | recommended | installed
Update openSUSE 11.4 | ModemManager  | 4453   | recommended | installed
Update openSUSE 11.4 | MozillaFirefox| 4195   | recommended | installed
Update openSUSE 11.4 | MozillaFirefox| 4457   | security    | installed
```

5.2.12 Patch Informationen abfragen

Informationen über die einzelnen Patche kann man mit `info –t patch` abfragen.

```
uws@tux>zypper info –t patch MozillaFirefox
Loading repository data …
Reading installed packages …

Informationen zu MozillaFirefox

Name: MozillaFirefox
Version: 5020
Arch: noarch
Vendor: maint-coord@suse.de
Status: installed
Category: security
Created On: Fri Aug 19 19:47:02 2011
Reboot Required: No
Package Manager Restart Required: No
Interactive: No
Summary: MozillaFirefox: Update to Firefox 6
Description:
MozillaFirefox was updated to version 6
```

5.2.13 Empfohlene Pakete installieren

Für bereits installierte Pakete findet `Zypper` mit der Option `install-new-recommends` (`inr`) neu hinzugefügte empfohlene Pakete.

```
uws@tux>zypper  inr
Installierte Pakete lesen …

Die folgenden NEUEN Pakete werden installiert.
```

5.2.14 Abhängigkeiten prüfen

Mit `Zypper verify` (`ve`) kann man Paketabhängigkeiten prüfen und diese dann installieren. Startet ein installiertes Programm nicht mehr und es gibt eine Fehlermeldung, das etwas fehlt. So kann dieses geprüft werden.

```
uws@tux>zypper –e –nodeps mozilla-xulrunner190
uws@tux>firefox
Could not find compatible GRE between version 1.9.0 and 1.9.0
uws@tux>zypper ve
Installierte Pakete lessen …
Einige der Abhängigkeiten der installierten Pakete sind beschädigt. Um
diese Abhängigkeiten zu reparieren …
```

5.2.15 Spezialfälle

Sollen nur Pakete installiert werden, die kein Neustart, Bestätigungsmeldung anzeigen oder eine Lizenz angenommen werden soll, so gibt man die Option `--skip-interaktiv` an. In der legalen Grauzone bewegt sich die Option –l. Diese Option weist den Paketmanager an, alle Lizenzklauseln zu akzeptieren.

```
uws@tux>zypper update --kip-interactiv
```

Möchte man nur den Source Code eines Pakets herunterladen, so gibt es hierzu die Option source-install (si). Nach dem Herunterladen kann man das Paket von Hand übersetzten.

```
uws@tux>zypper si –d inkscape
```

Soll ein Paket mit einer bestimmten Versionsnummer deinstalliert werden, so geschieht das mit dem nachfolgenden Befehl. Anstelle von dem = kann man auch < setzen, dann werden alle Versionen deinstalliert, die kleiner sind als die angegebene Nummer. Genauso funktioniert auch das einspielen einer bestimmten Version eines Paketes.

```
uws@tux>zypper remove "amarok=2.2"

uws@tux>zypper install "amarok=2.2.0-2"
```

Um zu erfahren, wer das Paket gebaut hat, ob es installiert ist oder aus welcher Bibliothek es stammt, so wird hierbei die Option `info` verwendet.

```
uws@tux>zypper info libxinel
```

5.2.16 Proxy Einstellung

Gibt es einen Proxy Server im Netzwerk, so muss dieser in Yast (Netzwerkdienste – Proxy) eingetragen werden, incl. der Portnummer wie z.B. `http://192.168.1.18:80`

Ist der Proxy nicht eingetragen, können keine Updates heruntergeladen werden und es kommt die Fehlermeldung „`couldn't connect to host`“

5.2.17 Alte Kernel löschen

Als erstes lassen wir uns anzeigen, welche Kernel Versionen installiert sind. Der nachfolgende Befehl listet alle Kernel Versionen sortiert nach Datum auf.

```
uws@tux>rpm –pa –last kernel-\*
kernel-sysms-3.11.10-11.1.x86_64         Wed May 21 13:55:49 2014
kernel-default-devel-3.11.10-11.1.x86_64 Wed May 21 13:55:47 2014
kernel-desktop-devel-3.11.10-11.1.x86_64 Wed May 21 13:55:46 2014
kernel-source-3.11.10-11.1.noarch        Wed May 21 13:55:42 2014
kernel-devel-3.11.10-11.1.noarch         Wed May 21 13:55:30 2014
kernel-desktop-3.11.10-11.1.x86_64       Wed May 21 13:55:02 2014
kernel-sysms-3.11.10-7.1.x86_64          Tue Feb 11 07:29:26 2014
kernel-default-devel-3.11.10-7.1.x86_64  Tue Feb 11 07:29:24 2014
kernel-desktop-devel-3.11.10-7.1.x86_64  Tue Feb 11 07:29:23 2014
kernel-source-3.11.10-7.1.noarch         Tue Feb 11 07:29:16 2014
kernel-devel-3.11.10-7.1.noarch          Tue Feb 11 07:29:01 2014
kernel-desktop-3.11.10-7.1.x86_64        Tue Feb 11 07:28:48 2014
```

Nun können wir die alten Kernel Versionen löschen.

```
uws@tux>zypper rm kernel-desktop-devel-3.11.10-7.1 \
kernel-devel-3.11.10-7.1 \
kernel-desktop-3.11.10-7.1
```

5.2.18 Patch decline (sperren)

Läst sich ein Patch nicht installieren, so kann man ihn sperren, so dass er nicht mehr installiert wird. Eine eingerichtete Sperre kann man sich mit `zypper locks` oder mit `zypper ll` anzeigen lassen.

```
root@tux>zypper locks
# | Name              | Type  | Repository
--+-------------------+-------+-----------
1 | OpenSUSE-2016-923 | patch | (any)
```

Einen Patch sperren kann man mit `zypper addlock` vornehmen. Bei der Option `–t` wird der Type angegeben. Folgende Typen gibt es: `package`, `patch`, `pattern`, `product`

```
root@tux>zypper addlock –t patch OpenSUSE-2016-923
```

Eine Sperre kann mit `zypper rl` wieder aufgehoben werden.

```
root@tux>zypper rl OpenSUSE-2016-923
```

5.3 Yum (Red Hat)

Um Software unter Red Hat zu installieren, gibt es das Programm `yum` in der Shell.

5.3.1 Proxy Einstellung

Damit man Software installieren kann, sollte man in der `.bashrc` für den User `root` hinzufügen.

```
root@tux>grep http .bashrc
export http_proxy=http://10.18.14.81:80/
export ftp_proxy=http://10.18.14.81:80/
export HTTP_PROXY=http://10.18.14.81:80/
export FTP_PROXY=http://10.18.14.81:80/
```

5.3.2 Paket installieren

Mit `install` oder auch `groupinstall` werden Pakete installiert. Mit der Option `–y` wird das Paket ohne Rückfrage installiert

```
root@tux>yum groupinstall kde-desktop gnome-dektop

root@tux>yum –y install samba.x86_64
```

5.3.3 Paket löschen

Mit `remove` kann man installierte Pakete löschen.

```
root@tux>yum remove firefox.x86_64
```

Ganze Paketgruppen werden mit `groupremove` gelöscht.

```
root@tux>yum groupremove gnome-desktop
```

5.3.4 Paket aktualisieren

Ein Paket wird mit `update` aktualisiert. Ohne die Angabe des Software Pakets, werden alle Applikationen aktualisiert. Gibt man noch die Option –y mit an, so werden die Updates ohne nachfragen installiert.

```
root@tux>yum update firefox.x86_64

root@tux>yum –y update
```

Vor einem Update kann man sich die Pakete anzeigen lassen, die aktualisert werden.

```
root@tux>yum list updates
Loaded plugins: downloadonly, refresh-packagekit, security
Updated Packages
coreutils.x86_64        8.4-19.0.1.el6_4.2     ol6_latest
coreutils-libs.x86_64   8.4-19.0.1.el6_4.2     ol6_latest
gdb.x86_64              7.2-60.el6_4.1         ol6_latest
glx-utils.x86_64        9.0-0.8.el6_4.3        ol6_latest
.
```

Ganze Paketgruppen lassen sich mit `groupupdate` aktualisieren.

```
root@tux>yum groupupdate ‘Graphical Internet‘

Dependencies Resolved
Upgrade     5 Package(s)
Is this ok [Y/N]: y

Running Transaction
Updating : evolution-data-server-3.0.2-1.el6.x86_64    1/10
Updating : evolution-3.0.2-3.el6.x86_64                2/10
Updating : evolution-NetworkManager-3.0.2-3.el6.x86_64 3/10
Updating : evolution-help-3.0.2-3.el6.x86_64           4/10
Updating : empathy-3.0.2-3.el6.x86_64                  5/10
Cleanup  : evolution-NetworkManager-3.0.2-3.el6.x86_64 6/10
Cleanup  : evolution-help-3.0.2-3.el6.x86_64           7/10
Cleanup  : evolution-3.0.2-3.el6.x86_64                8/10
Cleanup  : empathy-3.0.2-3.el6.x86_64                  9/10
Cleanup  : evolution-data-server-3.0.2-1.el6.x86_64    10/10
```

5.3.5 Paket suchen

Möchte man den Paketnamen eines Programms wissen, so wird herzu der Befehl `yum search` benutzt.

```
root@tux>yum search samba
Loaded plugins: downloadonly, refresh-packagekit, security
========== N/S Matched: samba ==========
samba-client.x86_64: Samba client programs
samba-common.i686: Files used by both Samba servers and clients
samba-common.x86_64: Files used by both Samba servers and clients
samba-doc.x86_64: Documentation fort he Samba suite
```

5.3.6 Vorhandene Pakete

Alle vorhandenen Pakete in der Yum Database werden mit `list` angezeigt. Als erstes werden die installierten Pakete angezeigt und danach die Available Packages.

```
root@tux>yum list | less
Loaded plugins: downloadonly, refresh-packagekit, security
```

5.3.7 Installierte Paket

Installierte Pakete können mit `list installed` angezeigt werden.

```
root@tux>yum list installed | less
Loaded plugins: downloadonly, refresh-packagekit, security
Installed Packages
ConsoleKit.x86_64       0.4.1-3.el6 @anaconda-OracleLinuxServer
ConsoleKit-libs.x86_64  0.4.1-3.ol6 @anaconda-OracleLinuxServer
.
.
```

5.3.8 Paket Infomationen

Informationen über Pakete können mit `yum info` angezeigt werden.

```
root@tux>yum info firefox
Loaded plugins: downloadonly, refresh-packagekit, security
Installed Packages
Name        : firefox
Arch        : x86_64
Version     : 17.0.6
Release     : 1.0.1.el6_4
Size        : 29M
Repo        : installed
From repo   : ol6_latest
Summary     : Mozilla Firefox Web Browser
URL         : http://www.mozilla.org/projects/firefox/
License     : MPLv1.1 or GPLv2+ or LGPLv2+
Description : Mozilla Firefox is an open-source web browser, designed for
            : standards compliance, performance and portability.
```

5.3.9 Repository

Mit `yum repolist` werden nur die Repositories ausgegeben, die für das System aktiviert sind.

```
root@tux>yum repolist
Loaded plugins: downloadonly, refresh-packagekit, security
repo_id           repo_name                                     status
ol6_UEK_latest    Latest Unbreakable Enterprise Kernel ..          176
ol6_latest        OracleLinux 6Server latest (x86_64)           21.522
```

Alle Repositories werden mit `yum repolist all` angezeigt.

```
root@tux>yum repolist all
Loaded plugins: downloadonly, refresh-packagekit, security
repo_id           repo_name                                   status
ol6_UEK_base      Unbreakable Enterprise Kernel for Oracle.. disabled
ol6_UEK_latest    Latest Unbreakable Enterprise Kernel ..     enabled:176
OL6_ga_base       Oracle Linux 6Server GA installation        disabled
ol6_latest        OracleLinux 6Server latest (x86_64)         enabled:21.522
.
.
```

Im Verzeichnis `/etc/yum.repos.d` gibt es die Datei `public-yum-ol6.repo`. In dieser Datei befinden sich alle Repositories und hier kann man sie auch aktivieren. Ein Update der Datei kann man folgendermaßen machen.

```
root@tux>cd /etc/yum.repos.d

root@tux>mv public-yum-ol6.repo public-yum-ol6.repo.disabled

root@tux>wget http://public-yum.oracle.com/public-yum-ol6.repo
```

5.3.10 Repository hinzufügen

Um ein neues Repository hinzufügen zu können, muss vorher die Release Informationen heruntergeladen und installiert werden.

```
root@tux>wget
http://download.fedoraproject.org/pub/epel/6/x86_64/epel_release-6-
8.noarch.rpm

root@tux>rpm –ivh epel_release-6-8.noarch.rpm

root@tux>yum repolist
```

Erfolgt nach `yum repolist` die Fehlermeldung: `Error: Cannot retrieve metalink for repository: epel. Please verify its path and try again`, so ist die Variable `https_proxy` zu setzten.

```
root@tux>export https_proxy=https://10.18.41.81:80/
```

5.3.11 Update mit Cron

Sollen die Updates vollautomatisch heruntergeladen und installiert werden, so kann man das am enfachsten mit dem Programm `yum-cron` erledigen. Dieses Paket muss vorher installiert werden.

```
root@tux>yum install yum-cron
```

Nach erfolgreicher Installation befindet sich die Konfigurationsdatei unter `/etc/sysconfig` mit dem Namen `yum-cron`. Nach erfolgter Konfiguration muss der `Daemon` für `yum-cron` eingerichtet werden.

```
root@tux>service yum-cron start

root@tux>chkonfig yum-cron on
```

Nun befindet sic him Verzeichnis `/etc/cron.daily` die Datei `0yum.cron`.

5.4 Apt-get (Debian)

Um Software unter Debian zu installieren, gibt es das Programm `apt-get` in der Shell.

5.4.1 Pakete installieren

5.4.2 Pakete suchen

5.4.3 Pakete löschen

Installierte Software kann man mit `apt-get remove` löschen. Sollen auch die Konfigurationsdateien mit gelöscht werden, so wird der Befehl `apt-get purge` verwendet.

```
root@tux>apt-get remove <paket-name>

root@tux>apt-get purge <paket-name>
```

5.5 Pacman (Arch Linux)

Unter Arch Linux wird Software mit Kommandozeilen Werkzeug `pacman` eingespielt.

5.5.1 Konfiguration

Für die Konfiguration von Pacman gibt es in dem Verzeichnis `/etc` die Datei `pacman.conf`. Eine Farbige Ausgabe der Meldungen erhält man, wenn die Option `color` eingeschaltet wurde.

```
uws@tux>cat /etc/pacman.conf
#RootDir     = /
#DBPath      = /var/lib/pacman/
#CacheDir    = /var/cache/pacman/pkg/
#LogFile     = /var/log/pacman.log
#GPGDir      = /etc/pacman.d/gnupg/
HoldPkg     = pacman glibc
#XferCommand = /usr/bin/curl –C - -f %u > %o
#XferCommand = /usr/bin/wget –passive-ftp –c –o %o %u
#CleanMethod = KeepInstalled
#UseDelta    = 0.7
Architecture = auto
```

Die Download-Geschwindigkeit lässt sich mit der Option `--limit-rate` festlegen. Diese Option wird dem Parameter `XferCommand` hinzugefügt.

```
XferCommand = /usr/bin/wger --passive-ftp --limit-rate=40k –c –o %o %u
```

5.5.2 Repositories

Es gibt für Offizielle Arch Linux Pakete sechs verschiedene Repositories, wo die Pakete zugeordnet wurden.

Core	Enthält Grundlegende Programme
Extra	Enhält zusätzliche Programme, wie z.B. KDE und Gnome
Testing	Enthält neuere Versionen, die aber noch nicht getestet worden sind
Community	Enthält Programme, die ursprünglich nau als PKGBUILD im AUR zu Verfügung standen und nach einer Testphase übernommen wurden.
Multilib	Enthält 32 Bit Programme für 64 Bit Systeme
Multilib-testing	Wie Multilib, jedoch sind die Programme noch nicht ausreichend getestet worden.

Weitere Repositories kann man sich in der Datei `/etc/pacman.conf` definieren, indem man dort einen Repo-Namen und Server spezifiziert. Eine Liste von `inoffizieller` Repositories findet man im englischen Wiki http://wiki.archlinux.org/index.php/Unofficial_user_repositories.

5.5.3 Spiegel-Server

Es gibt verschiedene Spiegel-Server von dem Hauptserver ftp.archlinux.org. Die Liste der Spiegel-Server befindet sich unter `/etc/pacman.d` und hat den Namen `mirrorlist`. Unter https://www.archlinux.de/?page=MirrorStatus kann man sich die Geschwindigkeit der Spiegelserver anzeigen lassen. Eine aktuelle Liste der Spiegel-Server kann man sich unter https://www.archlinux.org/mirrorlist erstellen lassen. Die Antwortzeiten der Spiegel-Server kann man mit `rankmirrors` oder auch mit `pacman-mirrors –g` testen.

```
uws@tux>rankmirrors –t 5
```

Die `mirrorlist` wird vom System immer wieder mal aktualisiert. Man kann sich auch eine eigene Mirroliste erstellen und diese in der `pacman.conf` in den entsprechenden Repositories Abschnitten einfügen.

```
[core]
Include = /etc/pacman.d/mirrorlist
Include = /etc/pacman.d/myMirrorlist
```

5.5.4 Befehle

In der nachfolgenden Tabelle sind einige der wichtigsten Befehle aufgelistet. Die Angabe der Paketnamen muss immer `klein` geschrieben werden.

Befehl	**Beschreibung**
pacman –Syu	Eine komplette Systemaktualisierung wird durchgeführt.
pacman –S <paket1> <paket2>	Ein oder auch mehrere Pakete installieren
pacman –Sy	Lokale DB aktualisieren
pacman –Su	Alle installieren Pakete aktualisieren
pacman –Syy	Loakle DB neu aufbauen
pacman –Syuu	Alle installierten Pakete downgraden (z.B. von Testing nach Core/Extra
pacman –Ss <paket>	Nach Paketen suchen. Es wird nach Name oder Beschreibung gesucht.
pacman –Sg	Suche nach Paketgruppen
pacman –Sg <paketgruppe>	Inhalt einer Paketgruppe anzeigen
pacman –Q	Alle installierten Paket incl. Der Versionsnummer anzeigen
pacman –Qs <paket>	Nach installierten Paketen suchen
pacman –R <paket>	Paket deinstallieren
pacman –Rd <paket>	Pakete deinstallieren, ohne auf Abhängigkeiten zu achten.
pacman –Rdd <paket>	Pakete deinstallieren und alle Abhängigkeitsprüfungen abschalten
pacman –Rs <paket>	Paket mit allen Abhängigkeiten deinstallieren
pacman –Rss <paket>	Paket mit allen Abhängigkeiten und deren Abhängigkeiten deinstallieren
pacman –D <paket>	Status eines installierten Programms ändern ohne eine Neuinstallation.
pacman –Qi <paket>	Info über ein installiertes Paket anzeigen
pacman –Si <paket>	Info über ein zu installierendes Paket anzeigen
pacman –Sw <paket>	Paket herunterladen und noch nicht installieren
pacman –S testing/<paket>	Ein Paket aus einem bestimmten Repositorie installieren.
pacman –U <paket>	Ein lokales Paket installieren
pacman –Runs <paket>	Ein Paket mit allen Anbhängigkeiten deinstallieren
pacman –Qdt	Verwaiste Pakete anzeigen, die nicht mehr von anderen Paketen benötigt werden
pacman –Qet	Pakete anzeigen, die ausdrücklich installiert wurden, aber nicht von anderen benötigt werden
pacman –Scc	Leert den lokalen Speicher /var/cache/pacman/pkg
pacman –Sc	Löscht nicht mehr benötigte oder alte Pakete aus /var/cache/pacman/pkg und aus /var/lib/pacman
pacman –Qi <paket>	Alle installierten Dateien des Paketes anzeigen.
pacman –Qm	Pakete anzeigen, die sich in keinem aktivierten Repository sich befinden.
pacman –Qu	Anzeige, für welche von den Installierten Paketen Updates gibt.
pacman –Qk	Überprüft alle Pakete auf fehlende Dateien.
pacman –Fy	Die lokale Datenbank wird aktualisiert.
pacman –Fs <datei>	Das Paket suchen, das die Datei enthält.
pacman –Fsx >regex>	Suche nach regulären Ausdrücken, sonst wie –Fs.
pacman –Fi <paket>	Alle Dateien des Paketes anzeigen.
pacman-optimize	Die Pacman DB optimieren.

5.5.5 Update

Mit dem Befehl `pacman -Syu` kann man eine komplette System Aktualisierung vornehmen. Möchte man aber nur eine Aktualisierungen für die installierten Pakete vornehmen, so kann das mit `checkupdates` geschehen.

```
uws@tux>sudo checkupdates
```

6. Netzwerk

6.1 Netzwerkzugriffe erlauben / verbieten

Um den Zugriff auf verschiedene Diensts zu erlauben oder zu verbieten, gibt es die Dateien `/etc/hosts.allow` und `/etc/hosts.deny`. Soll zum Beispiel SSH von allen Systemen erlaubt sein, http nur von einem IP-Bereich und Telnet generell verboten, so sind in den Dateien folgendes einzutragen.

```
uws@tux>cat /etc/hosts.allow

sshd: ALL
http: LOCAL, 192.168.0.0/ 255.255.255.0

uws@tux>cat /etc/hosts.deny

telnet: ALL
```

6.2 Mehrere IP-Adressen vergeben

Möchte man einer Netzwerkschnittstelle eine zusätzliche IP-Adresse vergeben, so kann man das mit dem Befehl `ifconfig` erledigen.

```
uws@tux>ifconfig eth0 add 172.30.20.10 netmask 255.255.255.0

uws@tux>ifconfig eth0 del 172.30.20.10 netmask 255.255.255.0
```

6.3 Datenpakete verfolgen

Mit dem Programm `traceroute` kann man sich die IP_Router Datenpakete zum Ziel ermitteln.

```
uws@tux>traceroute seabaer-ag.de
```

6.4 Wake on Lan (WoL)

Einen ausgeschalteten Computer kann man mittels eines `Magic Pakets` anschalten. Hierzu gibt es den Befehl `wol`, den man in der Konsole eingibt.

```
uws@tux>/usr/bin/wol –h <broadcast> <macadresse>

uws@tux>wol –h 172.30.100.255 00:00:00:00:00:00
```

6.5 IP-Adresse anzeigen

Die IP-Adresse kann man sich mit dem Befehl `ifconfig` anzeigen lassen. Diesen Befehl kann man nur als `root` ausführen. Als Nachfolger gibt es den Befehl `ip`, den auch der normale User ausführen kann.

```
uws@tux>ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue UNKOWN
    Link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
.
.
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfio_fast …..
    Link/ether 08:00:27:a8:ad:a4 brd ff:ff:ff:ff:ff:ff
.
.
```

6.6 ZMD Meldung

Wenn in der `/var/log/messages` die Meldung `zmd: NetworkManagerModule (Warn): Failed to connect to NetworkManager` auftaucht, so besagt diese Meldung, das die Netzwerkschnittstellen nicht mit dem Networkmanager konfiguriert worden sind. Damit diese Meldung nicht mehr in die `/var/log/messages` auftaucht, ist die Datei `/usr/lib/zmd/modules/Novell.Zenworks.Zmd.NetworkManager.dll` umzubenennen. Danach ist der `zmd Daemon` mit `/etc/init.d/novell-zmd restart` neu zu starten.

```
uws@tux>cd /usr/lib/zmd/modules

uws@tux>mv Novell.Zenworks.Zmd.NetworkManager.dll
Novell.Zenworks.Zmd.NetworkManager.dll.ori

uws@tux>/etc/init.d/novell-zmd restart
```

6.7 Netzlaufwerke

Netzlaufwerke können unter KDE / Gnome mit `smb://<user>@<server>:<freigabe>` angezeigt werden.

CIFS Freigaben sollten immer mit der Option `sync` gemounted werden, da es sonst zu Datenverlusten kommen könnte, wenn die Netzwerkverbindung zusammen bricht.

6.8 Bonding

Sind in dem Rechner zwei oder mehrere Netzwerkkarten eingebaut, so kann man diese zu einer Virtuellen Netzwerkkarte zusammenfügen (Bonding). Unter SuSE befinden sich die Dateien im Verzeichnis `/etc/sysconfig/network` und unter Red Hat `/etc/sysconfig/network-scripts`. Nach der Konfiguration des Bondings muss das Netzwerk einmal neu gestartet werden.

6.8.1 SusE

Unter SuSE wird in der Datei `ifcfg-<id_network>` (MAC Adresse oder auch ethx) die Parameter auf `BOOTPROTO='none'` und `STARTMODE='off'` abgeändert.

```
uws@tux>egrep –w "BOOTPROTO|STARTMODE" ifcfg-eth0
BOOTPROTO='none'
STARTMODE='off'
```

Nun kann die Datei `ifcfg-bond0` mit folgendem Inhalt erstellt werden.

```
uws@tux>cat ifcfg-bond0
BOOTPROTO='static '
BROADCAST='192.168.1.255'
IPADDR='192.168.1.10'
NETMASK='255.255.255.0'
NETWORK='192.168.1.1' # Gateway
REMOTE_IPADDR=
STARTMODE='onboot'
BONDING_MASTER='yes'
BONDING_MODULE_OPTS='mode=active-backup miimon=100'
BONDING_SLAVE0='eth0'
BONDING_SLAVE1='bus-pci-0000:05:03.1' # PCI Kennung
BONDING_SLAVE2='…'
```

Folgende Bonding Module Opts gibt es:

Mode	**Beschreibung**
ballance-rr oder 0	Round-robin Regel: Die Pakete werden sequentiell vom ersten verfügbaren Slave bis zum letzten übertragen. Dieser Modus bietet Lastverteilung und Fehlertoleranz.
active-backup oder 1	Nur ein Slave im Bond ist aktiv. Fällt ein Salve aus, wird der andere Salve aktiviert.
balance-xor oder 2	Übertragene Pakete basierend auf dem Ergebnis einer XOR Operation der MAC Adresse des Senders und Empfängers.
broadcast oder 3	Über alle Slaves werden die Pakete übertragen. Dieser Modus bietet Fehlertoleranz.
802.3ad oder 4	Das Bonding wird nach dem IEEE Standard 802.3ad – Link Aggregation Control Protocol erstellt. Der Switch muss dieses unterstützen und auch konfiguriert werden.
balance-tlb oder 5	Adaptive transmit Load balancing – der Datenverkehr wird durch einen Salve erledigt. Falls der Empfang fehl schlägt, übernimmt ein anderer Slave.
balance-alb oder 6	Adaptive Load balancing – Beinhaltet balance-tlb und zusätzlich Lastverteilung für IPV4-Datenverkehr.

6.8.2 Red Hat

```
uws@tux>cat ifcfg-eth0
DEVICE=eth0
MASTER=bond0
SLAVE=yes
ONBOOT=yes
BOOTPROTO=none
TYPE=Ethernet
NM_CONTROLLED=no
USERCTL=no
HWADDR=00:1c:c4:E0:75:66      # MAC Addresse
NAME=”System eth0”
```

```
uws@tux>cat ifcfg-bond0
DEVICE=bond0
BONDING_OPTS=”mode=1 updelay=0 miimon=100 downdelay=0”
TYPE=Bond
BONDING_MASTER=yes
BOOTPROTO=none
IPADDR=192.168.1.10
NETMASK=255.255.255.0
BROADCAST=192.168.1.255
PREFIX=24
GATEWAY=192.168.1.1
DEFROUTE=yes
DNS1=192.168.1.110 # optional
DNS2=192.168.1.111 # optional
NM_CONTROLLED=no
USERCTL=no
IPV4_FAILURE_FATAL=yes
IPV6INIT=no
NAME=”Bond connection 1”
UUID=c819e8a8-4852-4266-9q0b-f06060e8aef7 # optional
ONBOOT=yes

uws@tux>cat /etc/modprobe.d/bonding.conf
alias bond0 bonding
# wenn nicht in ifcfg_bond0 gesetzt worden ist,
# hier die options eintragen.
# Gilt nicht fuer Red Hat Systeme
options bond0 mode=0 miimon=100 use_carrier=1

uws@tux>modprobe bonding

uws@tux>service network restart

uws@tux>cat /proc/net/bonding/bond0
```

Weitere Informationen über das Bonding gibt es unter `/usr/share/doc/kernel_doc-x.x.x/Documentation/networking/bonding.txt`.

6.9 DNS

In der Datei `/etc/resolf.conf` werden die DNS Server eingetragen.

```
uws@tux>cat resolf.conf
nameserver 192.168.1.110
nameserver 192.168.1.111
```

6.10 Netzwerk Monitor

Für das Anzeigen der Netzwerk Auslastung gibt es das Programm `iptraf`, das man nachträglich installieren muss.

6.11 Netzwerk Performance

Mit dem Programm `iperf` kann man die Netzwerk Performance messen. Auf einem Rechner wird das Programm als Server gestartet und von dem anderen Client aus wird dann die Performance gemessen. Ebenso gibt es das Programm `netperf` für die Performance Messung. Beide Programme muss man gegebenen falls nach installieren.

Als erstes starten wir den Server.

```
uws@tux>iperf –s
------------------------------------------------
Server listing on TCP port 5001
TCP window size: 85.3 kByte (default)
------------------------------------------------
```

Von den anderen Client starten wir die Perfromance Messung.

```
uws@tux1>iperf –c tux
------------------------------------------------
Client connection to tux, TCP port 5001
TCP window size: 16.0 kByte (default)
------------------------------------------------
```

6.12 DNS - Namensauflösung

Informationen über einen Rechner kann man mit `dig` abfragen. Dieser Rechner / Server kann auch im Internet stehen.

```
uws@tux>dig <domain_server> <server_name> [any, A, MX, SIG]

uws@tux>dig www.seabaer-ag.de

;  <<>> DiG 9.9.4-rpz2.13269.14-P2 <<>> www.seabaer-ag.de
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode; QUERY, status: NOERROR, id: 14122
;; flags; qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY; 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4000
;; QUESTION SECTION:
;www.seabaer-ag.de            IN    A

;;ANSWER SECTION:
www.seabaer-ag.de.      7200  IN    CNAME seabaer-ag.de.
seabaer-ag.de           7200  IN    A     81.169.145.156

;; Query time: 30 msec
;; SERVER: 192.168.1.110#53(192.168.1.110)
;; WHEN: Tue Jun 23 10:33:21 CEST 2014
;; MSG SIZE rcvd: 76
```

Mit dem Befehl `nslookup` kann man auch Informationen über einen Rechner erhalten, aber das Programm wird nicht mehr weiterentwickelt und wurde durch `dig` abgelöst.

```
uws@tux>nslookup tux.seabaer-ag.de
Server:           192.168.1.110
Address:    192.168.1.110#53

Name: tux.seabaer-ag.de
Address: 192.168.1.35
```

Ebenso kann man mit dem Befehl `host` Informationen erhalten.

```
uws@tux>host www.seabaer-ag.de
www.seabaer-ag.de is an alias for seabaer-ag.de.
seabaer-ag.de has address 81.169.145.156
seabaer-ag.de has IPv6 address 2a01:238:20a:202:1090::144
seabaer-ag.de mail is handled by 5 smtp.rzone.de.
```

6.13 Mount Cifs

Ab der OpenSuSE Version 13.1 wurde der `Standard-Authentifizierungsalgorithmus` auf `ntlmssp` geändert. Gibt es bei einem Mount mit einem Cifs-Share die Fehlermeldung: `mount erro(95): Operation not supportet`, so trägt man in der `/etc/fstab` den Parameter `sec=ntlmv2` ein. Sollte es auch mit diesem Parameter nicht funktionieren, so trägt man `sec=ntlm` ein.

6.14 DHCP

6.14.1 Server

Die Konfiguration von DHCP wird in dem Verzeichnis /var/lib/dhcp abgelegt. Welche IP-Adressen schon vergeben worden sind, kann man sich mit dem nachfolgenden Beispiel sich anzeigen lassen.

```
root@tux>egrep “lease|hostname|hardware|\}“ /var/lib/dhcp/dhcp.leases
```

6.14.2 Client

Wie auch bei dem DHCP Server, liegen die Informationen über DHCP in dem Verzeichnis /var/lib/dhcp.

Informationen über das Netz kann man sich folgendermaßen anzeigen lassen.

```
root@tux>arp -a
rbgo110.seabaernet (192.168.30.10) at 34:31:c4:66:e0:84 [ether] on br0
fsgo002.seabaernet (192.168.30.20) at 00:08:9b:c2:ef:8d [ether] on br0

root@tux>cat /proc/net/arp
IP address         HW type  Flags  HW address            Mask  Device
192.168.30.10  0x1          0x2    34:31:c4:66:e0:84    *         br0
192.168.30.20  0x1          0x2    00:08:9b:c2:ef:8d     *         br0

root@tux>ip neighbor show
192.168.30.10  dev  br0  lladdr   34:31:c4:66:e0:84  REACHABLE
192.168.30.20  dev  br0  lladdr    00:08:9b:c2:ef:8d  STALE
```

6.15 Ping

Ausgeben von erreichbaren Geräten.

```
root@tux>for i in $(seq 10 30); do ping -c 1 “192.168.30.$i“ | grep “bytes
from“; done
64 bytes from 192.168.30.10:  icmp_seq=1  ttl=64 time=39.8 ms
64 bytes from 192.168.30.20:  icmp_seq=1  ttl=255 time=1.50 ms
```

6.16 Private IP-Adressen

Für den privaten Bereich sind Adressbereiche aus dem öffentlichen Adressraum ausgespart. Diese Adressbereiche werden nicht in das Internet durchgereicht.

Folgende Adressbereiche (IPv4) gibt es:

10.0.0.0 bis 10.255.255.255	Class A Netz
172.16.0.0 bis 172.31.255.255	Class B Netz
192.168.0.0 bis 192.168.255.255	Class C Netz

7. Archive

7.1 Zippen

7.1.1 Zip & Unzip

Mit dem Programm `zip` kann man Dateien in einem Archive zusammen packen. Das Programm vereint die Funktionen der Programme gzip und tar miteinander, ist aber kompatibel zu anderen Packern (Winzip, Winrar, ..)

```
uws@tux>
```

7.1.2 Gzip & Gunzip

Das Programm `gzip` verkleinert eine Datei und hängt an diese Datei die Endung .gz an. Das Programm kann keine Archive erstellen wie `zip`. Möchte man ein Archiv erstellen, so muss das mit dem Aufruf `tar` geschehen.

```
uws@tux>ls
MeineDoku.txt
uws@tux>gzip MeineDoku.txt
uws@tux>ls
MeineDoku.txt.gz
```

Möchte man eine andere Dateiendung haben, so wird das mit der Option `–s` (Suffix) gemacht.

```
uws@tux>gzip –s .gzip MeineDoku.txt
uws@tux>ls
MeineDoku.txt.gzip
```

Der Kompressionsfaktor wird mit den Werten `-1` bis `-9` eingestellt, wobei `-1` schneller komprimiert und `-9` langsamer, aber dafür mit besser Kompressionsrate. Als Standardfaktor ist -6 eingestellt.

```
uws@tux>gzip -3 MeineDoku.txt
uws@tux>export GZIP=“-3“
```

Komprimierte Textdateien braucht man nicht erst auspacken, um sie zu lesen. Hierfür gibt es `zless` und `zcat`.

```
uws@tux>zless MeineDoku.txt.gz
#
# Hier geht es los.
#
```

Um komprimierte Dateien zu entpacken, kann man `gzip` die Option `–d` angeben oder die Dateien werden mit dem Befehl `gunzip` entpackt. Ist die Zieldatei schon vorhanden, so fragt `gzip` nach, ob die vorhandene Datei ersetzt werden soll. Mit der Option `–f` (force) wird die vorhanden Zieldatei ohne Nachfrage überschrieben.

```
uws@tux>gunzip MeineDoku.txt.gz

uws@tux>gzip –d MeineDoku.txt.gz
```

7.1.3 Bzip2 & Bunzip2

Das Programm Bzip2 komprimiert die Dateien etwas besser als Gzip. Wie für Gzip gibt es die gleichen Optionen für Bzip2. Bzip2 kann zusätzlich das Original behalten und legt eine neue komprimierte Datei an. Dieses Verhalten erreicht man mit der Option `–k`. Die Kompressionsfaktor wird mit `export BZIP2="-3"` eingestellt.

```
uws@tux>bzip2 MeineDoku.txt
uws@tux>ls
MeineDoku.txt.bz2

uws@tux>bzip2 –k MeineDoku.txt
uws@tux>ls
MeineDoku.txt
MeineDoku.txt.bz2
```

Um Dateien zu entpacken, gibt es das Programm bunzip2. Anders als gunzip, fragt bunzip2 nicht nach, wenn die Zieldatei vorhanden ist, sondern bricht die Aktion ab. Mit dem Schalter –f läst sich dieses Verhalten abschalten

```
uws@tux>bunzip2 MeineDoku.txt.bz2
```

7.1.4 Tar

Mit dem Befehl tar (tape archiver) werden Dateien in einem Container zusammengefasst. Diesen Container kann man anschließend packen oder dieses auch gleich mit tar erledigen.

```
uws@tux>tar -cvf container.tar Documents/

uws@tux>tar –cvzf container.tar.gz Documents/

uws@tux>tar –xvzf container.tar.gz
```

Parameter	**Beschreibung**
c	Erstellen (create)
v	im Hintergrund arbeiten
f	Dateiname
r	Datei zu einem Archiv hinzufügen
z	Archiv wird mit Gzip gepackt / entpackt
j	Archiv wird mit Bzip2 gepackt / entpackt
x	Entpackt das Archiv
t	Inhalt des Archives anzeigen lassen.
J	Archiv wird mit XY gepackt / entpackt

Soll mit `tar` ein Backup des Systems erfolgen, kann es nützlich sein, Verzeichnisse von dem Backup auszunehmen. Hierfür gibt es die Option `--exclude`. Möchte man tar mitteilen, das die Sicherung per ssh auf einem anderen Rechner abgelegt werden soll, so gibt es hierfür die Option `--rsh-command`.

In dem nachfolgenden Beispiel wird das Backup auf dem Rechner San im Verzeichnis /backup abgelegt. Die Datei hat dann den Namen FullBackup_2010_07_30.tar. Ab dem Wurzelverzeichnis werden alle Dateien, außer die Dateien im Verzeichnis /proc in dem Archiv abgelegt.

```
uws@tux>tar –cvf user@san:/backup/FullBackup_$(date '+%Y_%m_%d').tar --rsh-
command=/usr/bin/ssh --exclude=/proc /

uws@tux>tar –tjf /daten/backup/FullBackup.tar.bz2 | less
```

7.1.5 Inkrementelles Backup – Dateien suchen / Auspacken

Mit dem vorherigen Script werden die Dateien inkrementell gesichert. Mit den nachfolgenden Befehlen kann man nach Dateien suchen und auch Dateien wieder zurückspielen.

```
uws@tux>backuppath=/home/uws/backup
uws@tux>for archiv in ${backuppath}/backup-*.tgz; do tar –tzf $archiv –C /
>| grep bild145.jpg
>done

uws@tux>for archive in ${backuppath}/backup-*.tgz; do
>tar –xzf $archiv –C / daten/bilder/bild145.jpg
>done
```

8. Rsync und Rsnapshot

8.1 Rsync – Verzeichnisse abgleichen

8.1.1 Parameter

Parameter	Beschreibung
a	Archiv Modus, Abkürzung für –rlptgoD
v	Ausführliche Ausführung
n	Testlauf
b	Löscht die Dateien auf dem Ziel nicht, sondern hängt den Suffix ~ an die Dateien an.
z	Dateien auf dem Ziel werden komprimiert.
u	Überschreibt keine Datei auf dem Ziel, wenn der Zeitstempel neuer ist als auf der Quelle. Hierbei müssen beide die gleiche Systemzeit haben.
--delete	Löscht Dateien auf dem Ziel, wenn sie nicht mehr auf der Quelle vorhanden sind.
--suffix	Definiert einen neuen Suffix.
--stats	Noch mehr Informationen bei der Ausführung.
--partial	Rsync speichert die übertragenen Teilstücke als Hardlink ab und speichert nach erfolgreichem Übertragen die Datei.
--progress	Ausgabe des Progress Status.
--backup-dir	Sichert die zu löschenden Dateien auf dem Ziel in dem angegebenen Verzeichnis.
--bwlimit	Angabe der Bandbreite in Kilobit/Sekunde.

8.1.2 Verzeichnis Synchronisieren

Um ein Quell Verzeichnis mit einem Zielverzeichnis abzugleichen, gibt es den Befehl `rsync`. In dem nachfolgenden Beispiel werden die Dateien mit der Option `–a` rekursiv kopiert und mit der Option `–z` komprimiert. Die Bandbreite wird auf 30 Kilobit/Sekunde beschränkt. Wird die Angabe des Quell Verzeichnisses mit einem Slash abgeschossen, so wird auf dem Zielverzeichnis die Unterverzeichnisse des Quellverzeichnisses angelegt.

```
uws@tux>rsync --stats –avz --bwlimit=30 /home/uws backup:/home/uws
```

Auf dem Ziel werden die Dateien gelöscht, wenn sie nicht mehr auf der Quelle vorhanden sind.

```
uws@tux>rsync -av --delete /home/uws uws@baer.local.net:/daten/backup
```

Auf dem Ziel werden die Dateien nicht gelöscht, sondern mit einem Suffix versehen.

```
uws@tux>rsync -avb --suffix=.bak --delete /home/uws
uws@baer.local.net:/daten/backup
```

Die zu löschenden Dateien werden auf dem Ziel in einem angegebenen Verzeichnis gesichert. Diese Angabe ist entweder relativ zum Home-Verzeichnis (--backup-dir=old/ schreibt nach ~/old) oder eine absolute Pfadangabe.

```
uws@tux>rsync –av --delete --backup-dir=/daten/backup/old /home/uws
uws@baer.local.net:/daten/backup
```

Verzeichnisse und Dateien, die nicht mit synchronisiert werden sollen, werden mit `--exclude` angegeben.

```
uws@tux>rsync –av --exclude ".ssh/" --exclude ".*" /home/uws
uws@baer.local.net:/daten/backup
```

8.1.3 Rsync in Scripten

Möchte man rsync in einem Script einsetzen und ein Vollautomatisches Backup erstellen, so braucht man hierzu ein SSH-Schlüsselpaar, da man sonst das Kennwort eingeben muss.

Das SSH-Schlüsselpaar wird mit dem Befehl ssh-keygen –t dsa und der Angabe einer leeren Passphrase erstellt. Den öffentlichen Schlüssel (~/.ssh/id_dsa.pub) muss auf dem entfernten Rechner in die Datei ~/.ssh/authorized-keys eingetragen werden. Hierzu gibt es den Befehl ssh-copy-id –i <remote.host> 32 <user>@<remote.host>, der dieses erledigt. In der Datei authorized-keys können sich mehrere Schlüssel befinden.

Da wir einen Key ohne Passphrase erstellt haben, gibt es ein Sicherheitsrisiko. Um dieses Risiko zu minimieren, kann man für bestimmte Keys nur einzelne Kommandos erlauben. Hierbei wird in der Datei ~/.ssh/autohrized_keys am Anfang des Keys (ss-dss ..) der Befehl mit den dazugehörenden Optionen in der Form command="rsync Optionen" eingetragen. Das alles wird in einer Zeile vorgenommen.

```
uws@tux>ssh-keygen –t dsa
Generating public/private dsa key pair.
Enter the File in which to save the key (~/.ssh/id_dsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in ~/.ssh/id_dsa.
Your public key has been saved in ~/.ssh/id_dsa.pub.
The key fingerprint is:
aa:cc:dd:ee:gg:kk:pp:zz:ok:12:r5:f5:g7:3d:t9:g5 uws@tux
uws@tux>ssh-copy-id –i ~/.ssh/id-dsa.pub baer 32 uws@baer
29
Password:
Now try logging into the machine, with "ssh baer", and check in:

      .ssh/authorized_keys

to make sure we haven't added extra keys that you weren't expecting.
```

8.1.4 Rsync-Daemon

Rsync kann auch als Daemon rsyncd laufen. Der Standard Port für rsync ist 873, der in der Datei /etc/services steht. Der rsync Daemon wird auf modernen Systemen mit xinetd gestartet. Im Verzeichnis /etc/xinetd befindet sich die Datei rsync. In dieser Datei kann man die Anzahl der Verbindungen pro IP-Adresse einschränken. Hierfür ist die Option per_source zuständig.

Für die Konfiguration des rsync Daemons ist die Datei /etc/rsync.conf zuständig. In dieser Datei sind alle Freigaben aufgelistet.

```
uws@tux>cat /etc/rsync.conf
#Global OPTIONS

motd file = /etc/motd.rsync
uid = nobody
gid = nogroup
max connections = 50
socket options = SO_KEEPALIVE
timeout = 1200
log file = /var/log/rsync.log
transfer logging = true
hosts deny = *.local.org 192.168.10.13 210.135.22.66/24
```

```
# Module Options

[ftp]
      comment = public archive
      path = /daten/ftp
      use chroot = yes
      max connections = 10
[privat_zone]
      comment = Meine Daten
      path = /home/uws
      secrets file = /etc/rsyncd.uws.secrets
      auth users = uws
      uid = uws
```

Parameter	**Beschreibung**
motd file	Begrüßungsmeldung des Servers.
uid	User ID-Nr. nach dem Verbindungsaufbau einen Kindprozess starten.
gid	Group ID-Nr, nach dem Verbindungsaufbau einen Kindprozess starten.
max connections	Anzahl der Clients, die eine Verbindung aufbauen dürfen.
hosts deny	IP-Adressen oder ganze Bereiche den Verbindungsaufbau verbieten.
secrets file	Passwort Datei, wo der Name und das Passwd drin stehen z.B. uws:<passwd>

8.2 Rsnapshot

Mit dem Tool `Rsnapshot` werden die Daten in verschiedenen Versionen abgespeichert. Standardmäßig geschieht das im Verzeichnis `/.snapshots`. Für die Konfiguration wird die Datei `/etc/rsnapshot.conf` benutzt. In der `Crontab` werden nun zwei Einträge angelegt, einen Aufruf rsnapshot mit der Option `hourly`, die andere mit der Option `daily`. So entstehen im Snapshotverzeichnis verschiedene Unterverzeichnisse `daily.0, daily.1` … und `hourly.0, hourly.1,` …

8.2.1 Konfiguration

Das Snapshotverzeichnis wird in der Datei `/etc/rsnapshot.conf` festgelegt. Bei der Angabe des Verzeichnisses ist darauf zu achten, das als Abschluß ein Slash (`/`) gemacht wird. In dieser Datei dürfen keine Leerzeichen gemacht werden, nur Tabs.

8.2.2 Sicherungsintervall

Bei der Angabe des Snapshot Intervalls handelt es sich um die Anzahl der Snapshots, die erstellt werden. Bei dem unten ausgelisteten Beispiel werden 6 Stündliche Snapshots erstellt (hourly.1 bis hourly.6), 7 Tägliche Snapshots (daily.1 bis daily.7).

```
Intervall        hourly      6
Intervall        daily       7
Intervall        weekly      4
Intervall        monthly     6
```

8.2.3 Sicherungsverzeichnisse

Die Angabe nach dem Quellverzeichnisses wird für das Snapshotverzeichnis benutzt. In dem unten aufgeührten Beispiel wird das localhost bei einem stündlichen Snapshot an /.snapshot/hourly.1 angehangen.

```
backup      /home/      localhost/
backup      /etc/       localhost/
backup      /usr/local/ localhost/
```

8.2.4 Exclude / Include

Möchte man Verzeichnisse aus dem Snapshot ausklammern, so wird eine Datei erstellt, in dem alle Verzeichnisse aufgelistet sind, die nicht gesichert werden sollen. Diese Datei wird dann bei `exclude_file` angegeben.

```
uws@tux>grep exclude_file /etc/rsnapshot.conf
# The include_file and exclude_file parameters. If enabled, simply get
#exclude_file     /path/to/exclude/file_name

uws@tux>cat exclude
# exclude path
/home/uws/Downloads
/home/uws/tmp
```

8.2.5 Unterschiede zwischen Snaphots

Mit der Option `diff` kann man sich die Unterschieden zwischen zwei Snapshots sich anzeigen lassen.

```
uws@tux>rsnapshot diff daily.0 daily.1
```

9. System

9.1 Prozesse

9.1.1 Prozesse auflisten mit ps

Prozesse werden mit dem Befehl `ps` aufgelistet. Ruft man den Befehl mit dem Parameter `–x` auf, so werden nur die eigenen Prozesse aufgelistet.

```
uws@tux>ps –x
  PID TTY   STAT  TIME  COMMAND
 6466 ?     S     0:00  sshd: uws@pts/0
 6467 pts/0 Ss    0:00  -bash
11882 pts/0 R+    0:00  ps -x
```

Parameter	Beschreibung
a	Auflistung der Prozesse aller Benutzer.
l	Lange Ausgabe mit zusätzliche Informationen.
w	Verlängert die Spalte Command für die Ausgabe, darf auch mehrfach verwendet werden.
x	Eigene Prozesse anzeigen lassen.
u	Anzeige des Eigentümers, Rechenzeit und Speicheranteil des Prozesses.

Auflistung des Status der Prozesse.

STAT	Beschreibung
S	Sleeping
R	Running
D	Dead
Z	Zombie

Die Anzahl von User Prozessen kann man sich folgendermaßen ermitteln.

```
uws@tux>ps –U uws –u uws –N | wc –l
52
```

9.1.2 Prozesse auflisten mit pstree

Mit `pstree` werden die Prozesse als Baumstruktur angezeigt. Bei dieser Anzeige kann man den Zusammenhang zwischen ‚`Eltern`' und ‚`Kind`' Prozess sehen.

```
uws@tux>pstree –A uws
sshd---bash---pstree
```

Parameter	Beschreibung
a	Anzeige der Parameter, mit der die Prozesse laufen.
h	Highlight für den eigenen Prozess.
H	Mit der Angabe der Prozessnr. wird auch dieser im Highlight dargestellt.
p	Ausgabe der Prozessnummer (PID).
u	Ausgabe des Benutzers
A	Die Ausgabe wird mit Ascii formatiert

9.1.3 Prozesse auflisten mit pgrep

Der Befehl `pgrep` vereint die Befehle `ps` und `grep`. Mit dem Parameter –lf wird der vollständige Befehlsaufruf inklusive aller Argumente angezeigt.

```
uws@tux>pgrep –lf ssh
6419 sshd: uws [priv]
6466 sshd: uws@pts/0
```

9.1.4 Prozess abkoppeln

Wird ein Programm oder ein Befehl in einer `Bash` gestartet und schließt man dann soft die `Bash`, so wird der Prozess gelöscht, da er an dem Mutterprozess (`Bash`) hing. Möchte man das verhindern, so wird das Programm / Befehl mit `nohup` aufgerufen. Der nun so erzeugte Prozess läuft nun ganz eigenständig und wird auch nicht beendet, wenn die `Bash` geschlossen wird.

```
uws@tux>nohup find . –name „*.mp3“ > /tmp/MusikListe.txt
```

9.1.5 Prozesse löschen

Muss ein Prozess beendet werden, so verwendet man den Befehl `kill`. Eine Auflistung aller Anweisungen erhält man mit dem Parameter `–l`. Mit Befehl kann man auch mehrere Prozesse löschen. Mehrere Prozesse kann man aber auch einfacher löschen mit dem Befehl killall. Anstelle der Prozessnummer erwartet der Befehl den Namen des Prozesses. Mit dem Parameter –i wird der Interaktive Modus eingeschaltet und es erfolgt eine nachfrage, ob man den Prozess auch wirklich löschen möchte.

```
uws@tux>kill -9 6466

uws@tux>killall –i ssh
```

Anweisung	**Beschreibung**
SIGHUB	Nach dem beenden des Prozesses wird er direkt neu gestartet.
SIGTERM	Dem zu löschenden Prozess wird die Möglichkeit gegeben, hinter sich aufzuräumen.
SIGKILL	Hier wird der Prozess gelöscht, ohen Rücksicht auf Verluste.
SIGSTOP	Der Prozess wird angehalten, bis man ihn mit SIGCONT wieder startet.

Eine zweite Möglichkeit einen Prozess zu löschen gibt es mit dem Befehl `pkill`. Mit dem Parameter `–u` kann man gezielt Prozesse anderer Benutzer löschen.

```
uws@tux>pkill -9 –u uws
```

9.1.6 Prozess Limits

Die maximale Anzahl der Prozesse kann man sich mit `ulimit` anzeigen lassen.

```
uws@tux>ulimit –u
31482
```

9.2 Module

9.2.1 Geladene Module Anzeigen

Mit dem Befehl `lsmod` kann man sich die geladenen Module anzeigen lassen.

```
uws@tux>lsmod | grep dm
dm_mod      92428  0
```

9.2.2 Verfügbare Module Anzeigen

Mit dem Befehl `modprobe -l` werden alle verfügbaren Module angezeigt. Die Module befinden sich im Verzeichnis `/lib/modules/<kernelversion>/kernel`.

```
uws@tux>modprobe -l | more
```

9.2.3 Modul Informationen

Informationen über ein geladenes Modul erhält man mit dem Befehl `modinfo`.

```
uws@tux>modinfo <ModulName>
```

9.2.4 Module laden / entladen

Module können mit dem Befehl `modprobe` zur Laufzeit geladen oder entladen werden. Der Befehl löst alle Abhängigkeiten auf und lädt auch die Module, die das zu ladende Modul braucht. Mit der Option `–v` wird eine ausführliche Ausgabe der Aktion angezeigt.

Um ein Modul zu entladen, gibt man die Option `–r` an.

```
uws@tux>modprobe –v <ModulName>

uws@tux>modprobe –r <ModulName>
```

9.2.5 Module automatisch laden

Um Module bei einem Systemstart automatisch zu laden, trägt man den Befehl in die Datei /etc/init.d/boot.local ein.

```
uws@tux>cat /etc/init.d/boot.local
#! /bin/sh
#
# Copyright © 2002 SuSE Linux AG Nuernberg, Germany. All rights reserved.
#
.
.
#
modprobe dm-mirror
```

9.3 Boot Splash ändern

Den Boot Splash kann man folgendermaßen ändern.

- Download des neuen Boot Themes
- Entpacken der Datei
- Die ausgepackten Dateien, incl. der Verzeichnisse, werden in das Verzeichnis `/etc/bootsplash/themes` kopiert.
- In der Datei `/etc/sysconfig/bootsplash` das neue Theme der variable `THEME=` eintragen.
- In der Console als `root mkinitrd` ausführen.
- Reboot

9.4 Zeitserver

Mit dem Dienst NTP (Networt Time Protocol) ist es möglich, einen Zeitserver auf zusetzten. Ist auf dem Rechner der Dienst `ntp` konfiguriert, so kann man mit den nachfolgenden Befehlen Informationen über die Zeitsynchronisation abfragen.

```
uws@tux>ntpq –p

uws@tux>watch ntpq –np
```

Der Dienst `ntp` kann manuell gestartet werden, um eine Synchronisation mit den Zeitservern vorzunehmen.

```
uws@tux>/etc/init.d/ntpd [start] [restart] [force-reload] [status]

uws@tux>service ntpd force-reload # for oracle linux
```

Die Systemzeit sollte bei jedem Starten und Herunterfahren des Rechners in die Realtime Clock des Rechners geschrieben werden. Falls das nicht geschieht, so kann man dieses mit dem Befehl `hwclock` vornehmen.

```
uws@tux>hwclock --show

uws@tux>hwclock --systohc

uws@tux>hwclock --help
```

In der Datei `/etc/ntp.conf` werden die abzufragenden Zeitserver eingetragen. Sind mehrere Zeitserver konfiguriert, so kann man mit `prefer` einen bevorzugten Zeitserver definieren.

```
------ schnipp ------
server <timeserver1> prefer
server <timeserver2>
server <timeserver3>
------ schnapp ------
```

Der Client kann sich die Uhrzeit auch manuell von dem Zeitserver holen. Hierzu wird in der Console der Befehl `ntpdate` genommen.

```
uws@tux>ntpdate <zeitserver>
```

Folgende Zeitserver gibt es im Internet und können verwendet werden. Es ist nur eine Auswahl an Zeitservern.

TU-Berlin	ntps1-0.cs.tu-berlin.de
	ntps1-1.cs.tu-berlin.de
Uni Erlangen	ntp0.fau.de
	ntp1.fau.de
	ntp2.fau.de
	ntp3.fau.de
Evetel	ntp0.evetel.de
	ntp1.evetel.de
Freenet	ntp0.freenet.de
	ntp1.freenet.de
T-Online	ntp1.t-online.de
Web.de	ntp.web.de

9.5 Cronjobs

9.5.1 Allgemein

Für bestimmte Zeitpunkte kennt das Programm Cron auch vordefinierte Labels. In der nachfolgenden Tabelle sind die Labels aufgelistet. Sie können anstatt der Zeitangabe angegeben werden.

Label	**Bedeutung**	**Entspricht**
@yearly, @annualy	Jährliche Ausführung	0 0 1 1 *
@monthly	Monatliche Ausführung	0 0 1 * *
@weekly	Wöchentliche Ausführung	0 0 * * 0
@daily, @midnight	Tägliche Ausführung	0 0 * * *
@hourly	Stündliche Ausführung	0 * * * *
@reboot	Nach dem Systemstart	-

Neben der einfachen Zeitangabe versteht Cron auch Bereichsangaben und auch Teiler. Mehrere Angaben werden durch ein Komma voneinander getrennt eingegeben. Ebenso kann man anstelle der Ziffernangabe für die Monate und Tage auch die englischen Abkürzungen verwenden.

```
0,15,30,45 * * * * w>/var/log/users
*/4 * * * * w>/var/log/users
```

9.5.2 Systemweite Jobs

Die Systemweiten Cron-Jobs werden in der Datei `/etc/crontab` eingetragen. Der Aufbau der Datei ist der gleiche wie bei dem Benutzer Job. Der einzige Unterschied besteht darin, dass vor dem Befehl der Benutzername steht, der den Befehl ausführen soll.

```
uws@tux>cat /etc/crontab
SHELL=/bin/bash
HOME=/
PATH=/usr/bin:/usr/sbin:/sbin:/bin
MAILTO=root
-*/15 * * * * root /usr/logrotate
```

Die drei Variablen am Anfang der Datei sind für folgendes definiert.

SHELL	Angabe der zu verwendenden Shell
HOME	Das Home Verzeichnis
PATH	Pfade, in den nach Programmen gesucht wird
MAILTO	Empfänger der Cron-Job Meldungen

Möchte man selber Systemweite Jobs erstellen, so sollte dieses nicht in der `/etc/crontab` geschehen, da bei einem Update des Programms Cron die Datei überschrieben wird. Hierfür gibt es das Verzeichnis `/etc/cron.d`. In diesem Verzeichnis sollten die selbsterstellten Job-Tabellen abgelegt werden.

Die meisten Distributionen legen immer wieder kehrende Jobs in den Verzeichnissen `/etc/cron.hourly`, `/etc/cron.daily` oder `/etc/cron.monthly` ab.

9.5.3 Benutzer Jobs

Benutzer Cronjobs werden mit dem Befehl `crontab -e` erstellt. Die Syntax in der Tabelle ist Minute, Stunde, Tag, Monat, Wochentag. Der Sonntag hat die `0` oder die `7`, Montag die `1` u.s.w.

```
# Minute    Stunde      Tag(Monat)  Monat  Tag(Woche) Befehl
# (0-59)    (0-23)      (1-31)      (1-12) (0-7; 1=Mo)
   5         *           *           *       *          rm /var/log/uws.log
   */5       *           *           *       *         rm /var/log/uws.log
   2/5       *           *           *       *         rm /var/log/uws.log
   59        23          *           *       0         cp /var/log/messages
/var/log/messages.back
   0         0           *           *       *         cp /var/log/messages
/var/log/messages.back
   20,30     1           *           *       1-5       cp /var/log/messages
/var/log/messages.back
```

Der erste Befehl (beginnend mit 5…) wird 5 Minuten nach jeder vollen Stunde, der zweite alle 5 Minuten (die Schrittweite wird durch */Schrittweite angegeben), der dritte Job läuft ab der 2 Minute los und dann alle 5 Minuten. Der vierte Job startet einmal pro Woche sonntags um 23:59 Uhr, der fünfte täglich um 00:00 Uhr und der sechste montags bis freitags jeweils um 01:20 und 01:30 ausgeführt.

Die Con-Jobs werden im Verzeichnis `/var/spool/cron/tabs` mit dem jeweiligen Benutzernamen als Datei abgespeichert.

9.5.4 Berechtigungen

In der Datei `/etc/cron.allow` stehen die User drin, die Cron Jobs erstellen können. Benutzer, die keine Cron Jobs erstellen sollen, stehen inder Datei `/etc/cron.deny` drin.

```
root@tux>cat /etc/cron.deny

guest
```

9.6 Init-Skripte verwalten

Die Startskripte befinden sich im Verzeichnis `/etc/init.d`. Jeder Dienst, der in einem bestimmten Runlevel Starten oder Stoppen soll, bekommt einen Link in den Verzeichnissen `rc0.d` bis `rc6.d`. Die Startlinks bekommen als ersten Buchstaben ein `S` und die Stoplinks ein `K`. Als zweites kommt dann eine Zahl, an welcher Stelle / Reihenfolge der Dienst gestartet oder gestoppt werden soll.

9.6.1 Verwalten mit chkconfig

Im Kopfbereich der Startskripte sind zwei Zeilen mit Kommentarzeichen enthalten, die für die Verwaltung mit `chkconfig` zuständig sind.

```
# chkconfig: 2345 55 25
# description: Oracle startup and shutdown
```

In der ersten Zeile steht nach dem Doppelpunkt, in welchem Runlevel der Dienst starten soll. In dem obigen Beispiel in den Runlevels 2, 3, 4 und 5. Danach wird die Startposition festgelegt und zum Schluss kommt die Stoppposition. In der Zweiten Zeile steht eine Beschreibung des Dienstes. Die Beschreibung kann auch mehrzeilig ausgeführt werden, sofern jede Zeile mit einem Backslash endet, bis auf die letzte Zeile.

Ruft man den Befehl `chkconfig` ohne Parameter auf, so werden alle Dienste angezeigt, in welchen Runlevels sie laufen.

```
uws@tux>chkconfig

uws@tux>chkconfig --ist <dienstname>
```

Um einen Dienst zu installieren, so wird der Parameter `--add` angegeben.

```
uws@tux>chkconfig --add <dienstname>
```

Mit der Option `--del` kann man den Dienst löschen. Es werden nur die Links in der Verzeichnissen gelöscht, der Dienst wird weder gestoppt noch gestartet.

```
uws@tux>chkconfig --del <dienstname>
```

Für das ändern eines Dienstes braucht man nicht das Init-Script zu ändern, sondern dieses geschieht mit der Option `--level`.

```
uws@tux>chkconfig --level 3 <dienstname> off [on]
```

In dem Beispiel oben wird der Startlink aus dem Runlevel 3 entfernt und ein Stopplink in dem Runlevel erstellt. Sollen die Änderungen in mehrere Runlevels durchgeführt werden, so wird gibt man die Levels ohne Leerzeichen an (23). Den alten Stand kann man mit der Option `reset` wiederherstellen, sofern man nicht die Init-Scripte angepasst hat.

```
uws@tux>chkconfig <dienstname> reset
```

9.6.2 Verwalten mit insserv

OpenSuse und auch die Suse Enterprise Produkte benutzen für die Verwaltung der Init-Scripte den `insserv` Befehl. `Insserv` kalkuliert bei jedem lauf selbstständig, an welcher Position das Script gestartet oder gestoppt werden soll. Hierbei werden auch Abhängigkeiten der Dienste berücksichtigt. In dem Init-Script steht die Konfiguration für den `insserv` zwischen den Zeilen `### BEGIN INIT INFO` und `### END INIT INFO`.

```
### BEGIN INIT INFO
# Provides: sshd
# Required-Start: $network $remote_fs
# Required-Stop: $network $remote_fs
# Default-Start: 3 5
# Default-Stop: 0 1 2 6
# Description: Start the SSH Daemon
### END INIT INFO
```

In der ersten Zeile steht, welchen Dienst das Script zu Verfügung stellt. In den Zeilen zwei und drei stehen die Abhängigkeiten drin, welche für die Berechnung des Automatischen Starten oder Stoppens zuständig sind. Diese Boot-Facilities stehen in der Datei `/etc/insserv.conf` drin. Gibt man in dem Init-Script die Boot-Facilitie $all an, so wird der Dienst erst nach allen anderen Diensten gestartet.

Die letzten drei Zeilen geben an, in welchen Runlevels der Dienst gestartet oder gestoppt werden soll. Sowie eine Beschreibung des Dienstes.

Um einen Dienst mit `insserv` zu erstellen, so ruft man das Programm mit dem Dienstnamen auf. Das Programm erstellt automatisch dir Start- und Stopplinks in den definierten Runlevels.

```
uws@tux>insserv <dienstname>
```

Einen Dienst kann man mit der Option `–r` wieder entfernen.

```
uws@tux>insserv –r <dienstname>
```

Möchte man einen Dienst auch in einem zusätzlichen Runlevel anlegen, so geschieht das mit der nachfolgenden Befehlszeile.

```
uws@tux>insserv <dienstname>,start=3
```

Die Standardeinstellung für den Dienst kann man mit der Option `–d` wieder herstellen.

```
uws@tux>insserv –d <dienstname>
```

Leider kann `insserv` keine Liste aller Dienste ausgeben, hierfür kann man das Programm `chkconfig --list` nehmen.

9.7 Aktuellen Runlevel anzeigen

```
uws@tux>runlevel

N 5
```

9.8 Konsolen Login nicht möglich

Kann man sich nicht mehr an einer Konsole (tty0 – tty7) anmelden und es erscheint in der Konsole die Meldung, das `Module kann nicht geladen werden`, so muss man die `/etc/pam.d/login` überprüfen. Bei einem `32bit` Linux sollte die `pam_limits.so` nach `/lib/security` zeigen und bei einem `64bit` nach `/lib64/security`.

9.9 VirtualBox Gasterweiterung

Wenn die Installation der VirtualBox Gasterweiterung mit einer Fehlermeldung nicht vollständig durchgeführt wird, so müssen die Kernel Source noch installiert werden. Eventuell müssen auch die Module gcc, make, automake und autoconf installiert werden.

```
uws@tux>zypper install kernel-source kernel-syms

uws@tux>zypper install gcc make automake autoconf
```

9.10 Syslog

Alle Meldungen werden in die `message.log` geschrieben. Standardmäßig wird jede Stunde in die Datei eine Meldung geschrieben. Soll diese Zeit verändert werden, so wird dieses in die Datei `syslog-ng.conf.in` gemacht. Den Wert von `stats(3600)` ist auf die gewünschte Update Intervall zu setzten. Die Angabe erfolgt in Sekunden.

```
#
# global options
#
options { long_hostname(off); sysnc(0); perm(0640); stats(1200); };

uws@tux>SuSEconfig –module syslog-ng
Starting SuSEconfig, the SuSE Configuration Tool …
Running Module syslog-ng only
Reading /etc/sysconfig and updating the system …
Executing /sbin/conf.d/SuSEconfig.syslog-ng …
Checking //etc/syslog-ng/syslog-ng.conf.SuSEconfig file: ok
Installing new //etc/syslog-ng/syslog-ng.conf
Finished

uws@tux>/etc/init.d/syslog restart
Shutting down syslog services            done
Starting syslog services                 done
```

Auf `RedHat` Systemen wird in der Datei `/etc/rsyslog.conf` das `$ModLoad immark` aktiviert und der Parameter `$MarkMessagePeriod 1200` eingefügt. Die Angabe der Zeit erfolgt in Sekunden.

9.11 Sudo (Runas)

Mit dem Befehl `sudo` kann man Programme unter einem anderen Benutzer, z.B. als root, ausführen. Für die Konfiguration ist die Datei `/etc/sudoers` zuständig, die man mit `visudo` editieren kann. Eigene Konfigurationsdateien legt man am besten im Verzeichnis `/etc/sudoers.d` ab, da sonst bei einem Update die `/etc/sudoers` Datei überschrieben wird. Es werden nur Dateien eingelesen, die keine Extension haben. Mit `visudo –f <Dateiname>` können die eigenen Dateien editiert werden. Bei dem beenden von `visudo` wird ein Syntax check durchgeführt. Die Rechte für die Dateien sollten auf `0440` gesetzt werden.

9.11.1 Syntax

Die Syntax kann man sich mit dem folgenden Satz beschrieben:

- Wer darf
- Von was
- Als was
- Mit welchen Einschränkungen / Eigenschaften
- was

```
User Hostname = (ZielUser) Einstellungen: Befehl, Befehl
```

Anstelle eines Users kann man auch eine Gruppe angeben, die mit einem vorangestellten Prozentzeichen definiert werden.

9.11.2 Alias

Man kann Aliase definieren, in dem mehrere Befehle zusammengesetzt werden. Die erlaubten Aliasnamen enthalten nur Großbuchstaben, Ziffern oder den Underscore. Die Aliasnamen beginnen immer mit einem Großbuchstaben.

Die Syntax hierzu lautet:

```
Aliastype Aliasname = Befehl, Befehl
```

Folgenden Alias Typen gibt es:

- User_Alias Benutzerliste
- Runas_Alias Zielbenutzerliste
- Host_Alias Hostnameliste
- Cmnd_Alias Befehlsliste

Beispiel:

```
Cmnd_Alias CMD_SHUTDOWN = /sbin/shutdown, !/sbin/init *, /sbin/init 0
```

In dem Beispiel kann ein Shutdown durchgeführt werden, aber kein /sbin/init Befehl, bis auf /sbin/init 0. Die Negation wird mit einem vorangestellten Ausrufezeichen eingeleitet

9.11.3 Einstellungen

In der folgenden Übersicht gibt es die Ausführungsbeschränkungen

Einschalten	Ausschalten	Beschreibung
PASSWD	NOPASSWD	Abfrage des Passwortes
SETENV	NOSETENV	Weitergabe von Umgebungsvariablen
EXEC	NOEXEC	Shellescapes verhindern

9.11.4 Beispiele

Alle User dürfen einen Befehl ohne Passwort Abfrage ausführen, außer der User harry.

```
ALL,!harry ALL=(root) NOPASSWD:/bin/info.sh
```

Alle User der Gruppe admin dürfen alle Befehle ohne Passwort Abfrage ausführen, nur nicht Befehle für das Netzwerk und System Update.

```
%admin ALL=(root) NOPASSWD:ALL,!CMD_NETWORK,!CMD_UPDATE
```

Alias CMD_UPDATE definieren.

```
Cmnd_Alias    CMD_UPDATE    =    /usr/bin/zypper,    /usr/bin/zypper    pchk,
!/usr/bin/zypper patch
```

9.12 Dienste anzeigen

Welche Dienste automatisch gestartet werden, kann man mit dem Befehl `chkconfig` überprüfen.

```
uws@tux>chkconfig –t | grep on
acpid on
alsasound on
apache2 on
.
.
```

Laufende Dienste können mit dem Befehl `systemctl` oder `service` überprüft werden.

```
uws@tux>systemctl –a | grep running
proc-sys-fs-binfmt_misc.automount load active running Arbitary Executable .
acpid.service                     load active running ACPI Event Daemon
apache2.service                   load active running apache
.
.

uws@tux>service –status-all | grep running
Checking for service acpid         running
Checking for httpd2:               running
.
.
```

Den Status eines Dienstes kann man sich mit `systemctl status` sich anzeigen lassen.

```
root@tux>systemctl status cron.service
cron.service – Command Scheduler
      Loaded: loaded (/usr/lib/systemd/system/cron.service; enabled)
      Active: active (running) since Mon, 2013-07-15 12:30:00 CEST; 18h ago
    Main PID: 1290 (cron)
      CGroup: name=systemd:/system/cron.service
```

9.13 CPU Info

Informationen über die CPU stehen in `/proc/cpuinfo`.

```
uws@tux>cat /proc/cpuinfo
processor         : 0
vendor_id         : GenuineIntel
cpu family        : 6
model             : 26
model name        : Intel® Xeon® CPU X5550 @ 2.67GHz
.
.
```

Auch mit dem Befehl lscpu kann man sich Informationen über die CPU's anzeigen lassen.

```
uws@tux>lscpu
Architecture:    i686
CPU op-mode(s):  32bit, 64bit
Byte Order:      Little Endian
CPU(s):          2
.
.
```

9.14 Server Domain

9.14.1 Software

Damit man einen Server / Computer in einer Domain aufnehmen kann, muss folgende Software installiert sein.

- Redhat-config-samba # Nur bei Red Hat Systemen
- Samba-common
- Samba-client
- Samba
- Pam_krb5
- Krb-workstation
- Krb-libs
- krbafs

9.14.2 Konfiguration

Die Dateien `krb5.conf`, `smb.conf`, `nsswitch.conf`, `ntp.conf` und `hosts` müssen nun konfiguriert werden.

```
root@tux>cat /etc/krb5.conf
[logging]
 default = FILE:/var/log/krb5libs.log
 kdc = FILE:/var/log/krb5kdc.log
 admin_server = FILE:/var/log/kadmind.log

[libdefaults]
 default_realm = GO.SEABAER-AG.DE
 kdc_timesync = 1
 ccache_type = 4
 dns_lookup_realm = false
 dns_lookup_kdc = false
 ticket_lifetime = 24h
 renew_lifetime = 7d
 forwardable = true

[realms]
GO.SEABAER-AG.DE = {
 kdc = gode001.go.seabaer-ag.de
 kdc = gode002.go.seabaer-ag.de
admin_server = gode001.go.seabaer-ag.de
 default_domain = go.seabaer-ag.de
}

[domain_realm]
 .go.seabaer-ag.de = GO.SEABAER-AG.DE
# .example.com = EXAMPLE.COM
# example.com = EXAMPLE.COM
```

```
root@tux>cat /etc/samba/smb.conf
[global]
netbios name = tux
workgroup = GO
realm = GO.SEABAER-AG.DE
password server = gode001.go.seabaer-ag.de, gode002.go.seabaer-ag.de,
#wins server =
passdb backend = tdbsam
security = ADS
encrypt passwords = true
log level = 0
idmap backend = idmap_rid:OPENITSOLUTIONS=10000-100000000
idmap uid = 10000-100000000
idmap gid = 10000-100000000
allow trusted domains = no
template shell = /bin/nologin
client use spnego = yes
client ntlmv2 auth = yes
winbind use default domain = yes
winbind enum users = yes
winbind enum groups = yes
winbind nested groups = yes
restrict anonymous = 2
domain master = no
local master = no
preferred master = no
os level = 0
# log level = 10

template homedir = /home/%U
template shell = /bin/nologin

log file = /var/log/samba/%m.log
max log size = 5000

printcap name = /etc/printcap
load printers = no

[homes]
comment = Home Directories
browseable = no
writeable = yes

[printers]
comment = All Local Printers
path = /var/spool/samba
browseable = no
guest ok = no
writeable = no
printable = yes
```

```
root@tux>cat /etc/nsswitch.conf
#
# /etc/nsswitch.conf
#
# An example Name Service Switch config file. This file should be
# sorted with the most-used services at the beginning.
#
# The entry '[NOTFOUND=return]' means that the search for an
# entry should stop if the search in the previous entry turned
# up nothing. Note that if the search failed due to some other reason
# (like no NIS server responding) then the search continues with the
# next entry.
#
# Valid entries include:
#
#     nisplus                 Use NIS+ (NIS version 3)
#     nis               Use NIS (NIS version 2), also called YP
#     dns               Use DNS (Domain Name Service)
#     files             Use the local files
#     db                Use the local database (.db) files
#     compat                  Use NIS on compat mode
#     hesiod                  Use Hesiod for user lookups
#     [NOTFOUND=return] Stop searching if not found so far
#

# To use db, put the "db" in front of "files" for entries you want to be
# looked up first in the databases
#
# Example:
#passwd:    db files nisplus nis
#shadow:    db files nisplus nis
#group:     db files nisplus nis

passwd:     compat winbind
shadow:     compat
group:      compat winbind

#hosts:     db files nisplus nis dns
hosts:      files dns

# Example - obey only what nisplus tells us...
#services:   nisplus [NOTFOUND=return] files
#networks:   nisplus [NOTFOUND=return] files
#protocols:  nisplus [NOTFOUND=return] files
#rpc:        nisplus [NOTFOUND=return] files
#ethers:     nisplus [NOTFOUND=return] files
#netmasks:   nisplus [NOTFOUND=return] files

bootparams: nisplus [NOTFOUND=return] files

ethers:     files
netmasks:   files
networks:   files
protocols:  db files
rpc:        db files
services:   db files

netgroup:   nisplus
publickey:  nisplus
automount:  files nisplus
aliases:    files nisplus
```

```
root@tux>grep –i “go.seabaer-net.de“ /etc/ntp.conf
server gode001.go.seabaer-ag.de
server gode002.go.seabaer-ag.de
```

In der `/etc/hosts` Datei wird der Server Name eingetragen.

```
root@tux>grep –i “go.seabaer-net.de“ /etc/hosts
192.168.10.200 fsgo002.go.seabaer-ag.de fsgo002
```

9.14.3 Dienste starten

Nun müssen noch die Dienste smb und winbind gestartet werden.

```
root@tux>service smb start
root@tux>chkconfig smb on
root@tux>service winbind start
root@tux>chkconfig winbind on
```

9.14.4 Domain aufnehmen

Nach erfolgreicher Konfiguration kann der Server in die Domain aufgenommen werden.

```
root@tux>net ads join –U <user_with_domain_rights>
```

9.14.5 Testen

Nun können Benutzer,Gruppen und Informationen der Domain abgefragt werden.

```
root@tux>kinit uws@go.seabaer-ag.de # Domain eventuell Groß schreiben

root@uws>klist -e

root@tux>wbinfo –u # Domain User List

root@tux>wbinfo –g # Domain Gruppen

root@tux>wbinfo -t
```

9.15 Firewall ausschalten

Die lokale Firewall kann mit `chkconfig` oder `service` ausgeschaltet werden.

```
root@tux>service ipchains off
root@tux>service iptables off

root@tux>chkconfig ipchains off
root@tux>chkconfig iptables off
```

Unter Oracle Linux (Red Hat) den Wert `SELINUX` auf `disabled` stellen, der sich in der Datei /etc/selinux/conf befindet. Danach den Rechner `neu starten` oder `/usr/sbin/setenforce 0` eingeben. Den Status kann man mit `/usr/sbin/getenforce` abfragen. Mit `setenforce` kann man die Firewall im laufenden Betrieb ausschalten.

9.16 Ports

Einen belegten Port kann man mit `netstat` sich anzeigen lassen.

```
root@tux>netstat –anp | grep tcp
tcp  0  0.0.0.0.0:80  0.0.0.0:*  Listen  1658/nginx
```

Den belegten Port kann man mit `fuser` wieder freigeben.

```
root@tux>fuser –k –n tcp 80
```

Offene Ports kann man sich mit nmap anzeigen lassen:

```
root@tux>nmap localhost
root@tux>nmap fritz.box
```

9.17 Autostart

Für Gnome und KDE gibt es verschieden Verzeichnisse, die bei dem Anmelden an dem Desktop verarbeitet werden.

Gnome

```
/etc/xdg/autostart          # Systemweit
/usr/share/gnome/autostart # Systemweit
~/.config/autostart         # User
```

Die Dateien müssen auf `.desktop` enden.

KDE

```
/etc/xdg/autostart                      # Systemweit
/usr/share/autostart                    # Systemweit
~/.kde4/Autostart                       # User (KDE4)
~/.kde4/shutdown                        # User (KDE4)
~/.config/autostart                     # User (KDE5)
~/.config/autostart-scripts             # User (KDE5)
~/.config/plasma-workspace/env          # User (KDE5)
~/.config/plasma-workspace/shutdown     # User (KDE5)
```

9.18 Memory

Die aktuelle Speicherbelegung kann man sich mit dem folgenden Statement abfragen.

```
uws@tux>egrep –-color ‘Mem|Cache|Swap‘ /prov/meminfo

MemTotal:   66007312 kB
MemFree:    15150272 kB
Cached:     62775044 kB
SwapCached:      200 kB
SwapTotal:  52428792 kB
SwapFree:   52301204 kB
```

Auch mit dem Befehl free, kann man sich den Speicherverbrauch anzeigen lassen.

```
uws@tux>free –m –t
```

Eine schöne graphische Anzeige gibt es mit dem Programm `htop`. Dieses Programm kann aus den Repositories nachinstalliert werden, da es meistens nicht mit installiert wird.

9.19 HugePages

Die HugePages Einstellungen kann man mit folgendermaßen abfragen.

```
uws@tux>egrep --color 'Huge' /proc/meminfo

HugePagesTotal:   0
HugePages_Free:   0
HugePages_Rsvd:   0
HugePagesize:   2048 kB
```

Die Größe der HugePage kann man mit dem folgenden Script ermitteln.

```
uws@tux>cat HugePages.sh
#!/bin/bash
#
# Setting check for HugePages
# The Script is from:
# docs.oracle.com/cd/E37670_01/E37355/html/ol_config_hugepages.html
#
KERN=`uname –r | awk –F. '{printf("%d.%d\n",$1,$2);}'`
#
# Find HugePage size
#
HPG_SZ=`grep Hugepagesize /proc/meminfo | awk {'print $2'}`
NUM_PG=1
for SEG_BYTES in `ipcs –m | awk {'print $5'} | grep "[0-9][0-9]*"`
do
 MIN_PG=`echo "$SEG_BYTES/($HPG_SZ*1024)" | bc –q`
 if [ $MIN_PG –gt 0 ]; then
  NUM_PG=`echo "$NUM_PG+$MIN_PG+1" | bc –q`
 fi
done
#
# Finish and output the results
#
case $KERN in
 '2.4') HUGETLB_POOL=`echo "$NUM_PG*$HPG_SZ/1024" | bc –q`;
        printf "\nRecommend setting: vm.hugetlb_pool = $HUGETLB_POOL\n";;
 '2.6') printf "\nRecommend setting: vm.nr_hugepages = $NUM_PG\n";;
  *)    printf "\nUnrecognized kernel version $KERN. Exiting.\n";;
esac

uws@tux>./HugePages.sh

Recommended setting: vm.nr_hugepages = 3848
```

Einschalten der HugePages werden dann mit:

```
uws@tux>echo 3848 >/proc/sys/vm/nr_hugepages
```

Damit dieses auch den Reboot überlebt, so trägt man in der `/etc/sysctl.conf` folgende Zeile ein.

```
uws@tux>grep huge /etc/sysctl.conf

vm.nr_hugepages = 3848
```

Zum Abschluss der Konfiguration muss noch der Wert in der Datei `/etc/security/limits.conf` gesetzt werden. Hierzu wird der Wert der `HugePages` mal den Wert der `HugePageSize` genommen. In diesem Beispiel also `3848*2048=7880704`.

```
uws@tux>egrep memlock /etc/security/limits.conf
oracle      soft  memlock     7880704
oracle      hard  memlock     7880704
```

9.20 Shutdown / Reboot

Einen `Shutdown` des Systems kann man mit einem `init 0` erledigen und einen `Reboot` des Systems kann man mit einem `init 6` einleiten. Nachfolgend möchte ich weiter auf die Befehle `shutdown` und `reboot` eingehen.

9.20.1 Shutdown

Soll das System nach einer Wartezeit herunterfahren, so gibt man dem Befehl die Zeit in Sekunden an.

```
uws@tux>shutdown 10
```

Alternativ kann man auch die Uhrzeit angeben, wann das System heruntergefahren werden soll.

```
uws@tux>shutdown –r 14:30
```

Folgende Optionen kann man mit angeben.

-H Das System anhalten
-P Das System ausschakten
-r Das System neu starten
-c Abbruch des Neustarts

9.20.2 Last Reboot

Möchten man wissen, wann das System das letzte Mal neu gestartet worden ist, so kann man das mit den folgenden Befehlen sich anzeigen lassen.

```
uws@tux>uptime

07:22  1:35 an,  4 Benutzer,  Durchschnittslast: 0,02,  0,14,  0,12

uws@tux>last reboot | head –n 1

reboot system boot 3.11.10-21-deskt Mon Oct 6 05:45 – 07:22 (1:37)

uws@tux>who –b

      Systemstart 2014-10-06 05:45
```

9.21 Anmelden disablen

Möchte man für Wartungsarbeiten an dem System das Anmelden verbieten, so kann man im Verzeichnis `/etc` die Datei `nologin` erstellen. Danach können sich die User nicht mehr an dem System anmelden, außer `root` natürlich. Gibt es im Verzeichnis `/etc` auch die Datei `nologin.txt`, so wird dieser Inhalt dem User angezeigt, wenn er sich anmelden möchte.

9.22 Hinweistext

In der Datei `/etc/motd` (message of the day) kann man einen Text platzieren, der bei der Anmeldung an dem System angezeigt wird.

```
uws@tux>cat /etc/motd
Have a lot of fun…
```

Einen Text vor der Anmeldung kann man in den Dateien `/etc/issue` (Konsole) und `/etc/issue.net` (Netzlogin) platzieren.

```
uws@tux>cat /etc/issue
Welcome to openSUSE 13.1 "Bottle" – Kernel \r (\l).

uws@tux>cat /etc/issue.net
Welcome to openSUSE 13.1 "Bottle" – Kernel %r (%t).
```

Folgende Escape Codes gibt es:

Escape	Beschreibung
\b	Ausgabe der Baudrate
\d	Ausgabe des aktuellen Datums.
\s	Ausgabe des System Namens, Name des OS System
\l	Ausgabe des aktuellen tty Name
\m	Ausgabe des Architecture Identifier
\n	Ausgabe des Hostnamens
\o	Ausgabe der Domain
\r	Ausgabe der Release Nummer des OS Systems
\t	Ausgabe der aktuellen Zeit.
\u	Ausgabe der Anzahl der angemeldeten User.
\U	Ausgabe des Textes "1 User" oder "Users"
\v	Ausgabe der Version des OS Systems, Build Date und mehr.

9.23 Systeminformationen

Mit dem Befehl `uname` kann man sich die Systeminformationen, wie zum Beispiel die Kernel Version, anzeigen lassen. Ein Aufruf von `uname` ohne Parameter, entspricht ein `uname –s`.

In der nachfolgenden Tabelle stehen die Parameter und welche Informationen sie ausgeben.

Parameter	Beschreibung
-a / --all	Ausgabe aller Informationen
-i / --hardware-platform	Ausgabe der Prozessorfamilie
-m / --maschine	Ausgabe der Hardwareplattform
-n / --nodename	Ausgabe des Hostnamens
-o / --operating-system	Ausgabe des Betriebssystems
-p / --processor	Ausgabe des Prozessornamens
-r / --kernel-release	Ausgabe der vollständigen Kernel-Versionsnummer
-s / --kernel-name	Ausgabe des Kernel Namens
-v / --kernel-version	Ausgabe Erstellungsdatums des Kernels

```
uws@tux>uname –s
Linux

uws@tux>uname –r
3.11.10-21-desktop
```

9.24 SSL Zertifikate

SSL Zertifikate kann man mit `openssl` erstellen. In den nachfolgenden Beispiel wird ein Zertifikate für HTPPS Verbindungen für das Programm Shellinabox erstellt.

```
root@tux>cd /etc/shellinabox

root@tux>openssl genrsa –des3 –out server.key 1024

root@tux>openssl req –new –key server.key –out server.csr

root@tux>cp server.key server.key.org

root@tux>openssl rsa –in server.key.org –out server.key

root@tux>openssl x509 –req –days 365 –in server.cst –signkey server.key
 –out server.crt

root@tux>cat server.crt server.key > certificate.pem
```

9.25 NFS Freigaben anzeigen

Freigegebene NFS Shares kann man sich mit `showmount –e` anzeigen lassen.

```
root@tux>showmount –e tux01
Export list for tux01
/daten/backup tux
/daten/sw     (everyone)
```

9.26 NFS in Fstab

Sollen NFS Shares bei dem Start des Rechners verbunden werden, so werden die Shares in der `fstab` eingetragen.

```
root@tux>grep nfs /etc/fstab
<server>:/mail /mail nfs rsize=8192,wsize=8192,timeo=14,intr 0 0
```

Option	**Beschreibung**
rsize	Max. bytes read
Wsize	Max. bytes write
Timeo	Zeit NFS client wartet auf Anfrage
Intr	Erlauben, NFS Anfragen zu unterbrechen, wenn der Server ausfällt.

9.27 SystemD

9.27.1 Startzeit ausgeben

Die Startzeit des Systems kann man sich mit `systemd-analyse` anzeigen lassen.

```
root@tux>systemd-analyze
Startup finished in 4.560s(Kernel) + 1.363s(initrd) + 18.960s(userspace) =
24.883s
```

Eine ausführlichere Anzeigen bekommt man wenn man an dem Befehl den Parameter `blame` anhängt.

```
root@tux>systemd-analyze blame
```

Die Ausgabe kann man sich auch als Graphik abspeichern und sich dann anzeigen lassen.

```
root@tux>systemd-analyze plot > sysboot.svg
```

Protokollausgaben des Kernels sich anzeigen lassen. Wenn `Kernel` und `Initramfs` zusammen 15 Sekunden bei dem Start benötigen, ist alles in Ordnung

```
root@tux>dmesg --show-delta --color=always | less –R
```

9.27.2 Ausgabe der Errors vom booten

Die Ausgabe der Errors vom `booten` werden mit `journalctl` ausgegeben.

```
root@tux>journalctl –b –p err
```

Anzeigen der Kernel Meldungen, die das Initramfs absetzt.

```
root@tux>journalctl –b –o short-monotonic
```

Einen bestimmten Dienst kann man folgendermaßen sich anzeigen lassen.

```
root@tux>journalctl –b –o short-monotonic –u systemd-udev-settle.service
```

9.27.3 Service / Dienste

Services können mit dem Befehl `systemctl` verwaltet werden.

```
root@tux>systemctl start <service>

root@tux>systemctl enable <service>
```

Den Status eines Dienstes / Service kann man folgendermaßen abfragen.

```
root@tux>systemctl status networkd.service
```

9.28 Systemd Protokolle - Journalctl

Der nachfolger des Init-Systems SysVinit ist seit mehreren Jahren Systemd. Dabei wurde der Log Service Rsyslog durch das Journal von Systemd ersetzt. Dadurch gibt es auf den Systemd Maschinen keine `/var/log/messages` oder `/var/log/syslog` mehr. Die neuen Journal Dateien haben nun ein Binär Format.
Bei der Anzeige des Journals kommt der Pager `Less` um Einsatz. Der Zugriff mit `journalctl` benötigt keine Root-Rechte.
Wird dem User die Gruppe `systemd-journal` zugewiesen, so kann er alle Informationen abrufen, ansonsten nur die Meldungen für den User.

9.28.1 Konfiguration

Mit der Konfigurations Datei `/etc/systemd/journald.conf` kann man das Journal konfigurieren. Nachfolgend wird ein Teil des `journald.conf` angezeigt.

```
[Journal]
#Storage=auto
#Compress=yes
#Seal=yes
#SplitMode=uid
#SyncIntervalSec=5m
#RateLimitInterval=30s
#RateLimitBurst=1000
#SystemMaxUse=
#SystemKeepFree=
#SystemMaxFileSize=
#SystemMaxFiles=100
```

Storage	=	persistent	Die Log-Dateien landen in /var/log/journal
Storage	=	volatite	Die Log-Dateien gibt es nur in der laufenden Sitzung, bei einem Reboot / Shutdown werden sie gelöscht.
SystemMaxUse	=	2000M	Maximale Größe der Journal Dateien
SystemKeepFree	=	1000M	Wieviel Platz mindestens frei bleiben soll.

Die Einheiten sind K,M,G und T.

Weitere Informationen gibt es mit `man journald.conf`.

9.28.2 Journal Dateien verkleinern

Die Journal Dateien können auf einer Größe zu Recht gestutzt werden oder alle Einträge die älter sind als die angegebene Zeit werden gelöscht. Nachfolgend einige Beispiele. Im ersten Beispiel werden die Journal Dateien auf 100MB verkelinert.

```
root@tux>journalctl --vacuum-size=100M
```

Mit --vacuum-time kann man eine Zeitangabe Machen. In dem nächsten Beispiel werden alle Einträge gelöscht, die älter als 1 Monat sind.

```
root@tux>journalctl --vacuum-time=1month
```

9.28.3 Syslog

Soll weiterhin das Syslog benutzt werden, so sind die Variablen `ForwardToSysLog=yes` und `MaxLevelSyslog=debug` zu setzten. Zusätzlich muss der Rsyslog-Daemon laufen.

Meldungen vom Syslog können mit Omjournal in Journald überführt werden.

9.28.4 Status und Überprüfung

Den Status des Daemons kann man mit `systemctl` abfragen.

```
root@tux>systemctl status systemd-journald
 systemd-journald.service – Journal Service
  Loaded: loaded (/usr/lib/systemd/system/systemd-journald.service; static;
  Active: active (running) since Tue 2017-05-09 14:29:01 CEST; 6h ago
    Docs: man:system-journald.service(8)
          Man:journald.conf(5)
Main PID: 423 (system-journal)
  Status: ”Processing requests…”
   Tasks: 1 (limit: 512)
.
.
```

Die Größe des Journals kann mit `--disk-usage` abgefragt werden.

```
root@tux>journalctl --disk-usage
Archived and active journals take up 656.1M on disk.
```

Die Intrigität des Journals wird mit `--verify` durchgeführt.

```
root@tux>journalctl --verify
PASS: /var/log/a0848146a8854c698d28901e824/system@f5 ….journal
```

Ob die richtige Zeitzone eingestellt ist, kann man mit `timedatectl status` abgefragt werden. Mit `timedatectl set-timezone <zone>` kann man eine neue Zeitzone setzen.

```
root@tux>timedatectl status
      Local time: Tue 2017-05-09 20:37:07 CEST
  Universal time: Tue 2017-05-09 18:37:07 UTC
        RTC time: Tue 2017-05-09 20:37:07
       Time zone: Europe/Berlin (CEST, +0200)
 Network time on: no
NTP synchrinized: yes
 RTC in local TZ: yes
```

9.28.5 Anzeigen Journal / Filtern

Wird `journalctl` ohne Optionen aufgerufen, so werden alle Einträge zur Laufzeit und dem letzten Reboot angezeigt. Bei jedem Neustart des Rechners wird die Zeile `-- Reboot --` angezeigt.
Nur Error Meldungen kann man sich mit `–p err` anzeigen lassen. Es werden alle Meldungen angezeigt, nicht nur die letzten nach einem Reboot.

```
root@tux>journalctl –b –p err
-- Logs begin at Tue 2017-03-14 10:56:44 CET, end at Tue 2017-05-09 20:37…
May 09 14:28:58 tux kernel: tpm_tis 00:01: A TPM error (7) occurred att…
May 09 14:29_03 tux audspd[1040]: No plugins found, exiting
```

Alle protokollierten Bootvorgänge, die sich im Journal befinden, können mit `--list-boots` angezeigt werden.

```
root@tux>journalctl --list-boots
-22 b76be92c47054d9ab8a2fld8e14ba35d Tue 2017-03-14 10:56:44 CET-Tue 2017-…
-21 b508318148d245446ad90843f9400591 Fri 2017-03-17 12:07:21 CET-Fri 2017-…
```

Mit der Option –b wird das Journal seit dem letzten Reboot angezeigt. Gibt man noch eine Zahl an, so werden dann diese Meldungen ausgegeben. Gibt man z.B. eine -1 an, so wird das vorletzte Journal angezeigt.

```
root@tux>journalctl –b -1
```

Nach einer Zeit oder auch Zeitraum kann man das Journal auch Filtern. Das `Datum / Zeit` Format ist grundsätzlich `YYYY-MM-DD HH:MM:SS`.

```
root@tux>journalctl --since “2017-04-13 11:15:22“

root@tux>journalctl --since “2017-04-13 11:15:22“ --until “2017-04-13
11:15:22“

root@tux>journalctl --since yesterday

root@tux>journalctl --since 10:00 --until ”now”
```

Nach Komponenten kann man das Journal auch filtern. In dem nächsten Beispiel wird nach dem Apache Service gefiltert.

```
root@tux>pjournalctl –u apache2.service –u php-fpm.service --since
yesterday

root@tux>pidof apache2         # search pid, otherwise
root@tux>ps aux | grep apache2 # search pid
root@tux>journalctl _PID=<pid>

root@tux>id –u www-data
root@tux>journalctl _PID=<UID> --since today
```

Alle möglichen Filter kann man sich mit `man system.journal-fileds` sich anzeigen lassen. Meldungen von Anwendungen können auch gefiltert werden, es muss nur der Name des Programms, incl der Pfadangabe angegeben werden.

```
root@tux>journalctl /usr/bin/vlc
```

9.28.6 Kernel Meldungen

Kernel Meldungen konnte man früher mit dmesg sich anzeigen lassen. Nun wird die Option –k angegeben. Die Informationen von früheren Reboots können mit –b –n ausgegeben werden. Die Option –n bewirkt, das nur die angegebene Anzahl an Zeilen ausgegeben werden. Möchte man alle Meldungen von dem gewählten Reboot sehen, so lässt man die Option weg.

```
root@tux>journalctl –k

root@tux>journalctl –k –b -2 –n 10
```

9.28.7 Fortlaufende Ausgabe

Früher wurde eine fortlaufende Ausgabe von `dmesg` mit `tail –f [-n10]` gemacht. Nun genügt die Option `–f`, um das gleiche zu machen. Auch hier kann man mit `–n <zahl>` die anzuzeigende Zeilen angeben.

```
root@tux>journalctl –f –n 5
-- Logs begin at Tue 2017-03-14 10:56:44 CET. –
May 09 20:45:01 tux systemd[5508]: Stopped target Timers.
May 09 20:45:01 tux systemd[5508]: Received SIGRTMIN+24 from PID 5547
(kill).
May 09 20:45:01 tux systemd[5512]: pam_unix(system-user:session): session
….
May 09 20:45:01 tux systemd[1]: Stopped User Manager for UID 0.
May 09 20:45:01 tux systemd[1]: Removed slice User Slice of root.

root@tux>journalctl –u apache2 -f
```

9.28.8 Ausgabe in Datei

Für die Ausgabe der Meldungen in einer Datei muss der Pager abgestellt werden.

```
root@tux>journalctl –b –p err --no-pager > MyJournal.txt
```

Andere Ausgabe Formate können mit der Option `–o` oder auch `--output` angegeben werden. Die Standardausgabe heißt `short`. Mit `–o verbose` wird eine umfangreichere Ausgabe erstellt. Mit `–o cat` wird die Ausgabe gegenüber `short` nochmalig gekürzt. Eine Präzision des Zeitstempels wird mit `–o short-monotonic` gemacht. Eine Ausgabe im `Json` Format kann mit `–o json` oder besser mit `–o json-pretty` gemacht werden. Ein Export im `Binär` Format wird mit `–o export` durchgeführt.

9.29 Fstab

9.29.1 Beispiel

```
root@tux>cat /etc/fstab
#device-spec  mount_point fs-type options                          dump pass
/dev/sda1     /           ext4    default                             1    1
LABEL=home    /home       ext4    default                             0    0
fsgo100:/data /data       nfs     rsize=8192,wsize=8192,timeo=14,intr 0   0
.
.
```

9.29.2 Device-Spec

Die Geräte (Festplatten, DVD) können entweder als Gerätedatei, Label, UUID, NFS oder CIFS angesprochen werden.

Labels kann man für die verschiedenen Dateisysteme setzen. Für `ext2/3/4` können die Programme `e2label` oder `tune2fs` genommen werden. Für `Reiserfs` das Programm `reiserfstune`, `xfs` das Programm `xfs_admin` und für `btrfs` das Programm `btrfs`.

```
root@tux>e2label /dev/sda2 mydata
root@tux>tune2fs –L mydata /dev/sda2

root@tux>reiserfstune –l mydata /dev/sda2

root@tux>xfs_admin –L ”mydata” /dev/sda2

root@tux>btrfs filesystem label /dev/sda2 mydata
```

Die UUID kann man sich mit blkid sich anzeigen lassen.

```
root@tux>blkid
/dev/sda1: UUID:“89….“ LABEL=“root“ TYPE=“ext4“
/dev/sda2: UUID:”cd …” LABEL=”mydata” TYPE=”ext4”
/dev/sda3: TYPE=”swap” UUID=”c1…”
```

Mit dem Parameter –t lassen sich die Einträge Filtern. An dem Parameter hängt man noch den Wert TYPE, LABEL, UUID oder SEC_TYPE an.

```
root@tux>blkid –t TYPE=ext4
/dev/sda1: UUID:“89….“ LABEL=“root“ TYPE=“ext4“
/dev/sda2: UUID:”cd …” LABEL=”mydata” TYPE=”ext4”
```

Filter auf eine Geräte Datei.

```
root@tux>blkid /dev/sda2
/dev/sda2: UUID:”cd …” LABEL=”mydata” TYPE=”ext4”
```

Es gibt noch den Parameter –o. Hierbei wird als Wert List, Value, device, udev oder full mit angegeben.

```
root@tux>blkid –o list
```

9.29.3 Mount-Point

Man kann überall im System ein Device einbinden. Möchte man temporär ein Device einbinden, so kann das direkt unter `/mnt` gemacht werden. USB Geräte werden automatisch unter `/var/media` eingebunden.

9.29.4 FS-Type

In der nachfolgende Liste ist eine Auswahl der Dateisysteme angegeben.

Type	Beschreibung
auto	Automatisches Erkennen des Dateisystems
ext2/3/4	Ext-Dateisysteme
iso9660	Für CD's und DVD's
udf	DVD'S, hat iso9660 abgelöst
jfs	Journaled File System
nfs	NFS Shares
ntfs/ntfs-3d	Windows NTFS Filesystem, ntfs-3g mit schreibzugriff
reiserfs	Reiser FS
swap	Für die Swap Partition
ramfs	Ramdisk
tmpfs	Ramdisk, nutz auch den Swap-Platz
vfat	FAT 16/32
xfs	XFS
btrfs	BTRFS

9.29.5 Options

Mehrere Optionen werden durch ein Kommata getrennt angegeben und ohne Leerzeichen.

Option	Beschreibung
defaults	Es wird rw, suid, dev, exec, auto, nouser und async gesetzt.
dev / nodev	Gerätedateien werden interpretiert.
exec / noexec	Ausführen von Dateien erlauben.
gid	Setzen der Gruppen-ID, gid=users
uid	Setzen der User-ID, uid=username
suid / nosuid	Das SUID-Bit wird berücksichtigt.
auto / noauto	Wird beim booten automatisch eingehängt.
ro / rw	Setzen von Read only oder Read Write
sw	Das SWAP-Dateisystem verwenden.
sync / async	Alles wird direkt auf dem Datenträger geschrieben. Für USB-Geräte ist sync sinnvoll und async für hohen Datenaufkommen.
user / nouser	Einhängen dürfen alle User und Aushängen nur der User, der das Gerät eingehangen hat.
users	Jeder darf das Gerät mounten.
user,rw,umask=000	Für FAT-Systeme.
noatime / nodiratime	Zugriffszeiten nicht in der Inodetabelle speichern. Wird für SSD empfohlen.
errors=remount-ro	Bei einem Fehler wird das Gerät im Read only Modus engehangen.
noauto,x-systemd.automount	Automatisches Einbinden durch Systemd bei einem ersten Zugriff. Kann verwendet werden bei Systemen, die beim Booten verzögern (NFS, SSHFS, …).
discard	Wird bei SSD's für Trim verwendet.

9.28.6 Dump

Soll das Backup Programm `dump` das System sichern, so steht hier ein Wert `ungleich 0`.

9.29.7 Pass

Hier wird definiert, in welcher Reihenfolge die Datenträger überprüft werden sollen. Eine `0` steht für `keine Überprüfung`, eine `1` für als `erstes überprüfen` und eine `2` wird `nach 1` überprüft.

9.30 Fonts

9.30.1 Verzeichnisse

In den Verzeichnissen können Unterverzeichnisse angelegt werden, da alle Verzeichnisse durchsucht werden. In den Dateien `/etc/fonts/fonts.conf` und für `OpenSuSE` auch noch `/etc/fonts/suse-font-dirs.conf` sind die Verzeichnisse aufgelistet, die durchsucht werden sollen.

Folgende Verzeichnisse sind im Suchpfad:

/usr/share/fonts/<dir>
/usr/X11R6/lib/X11/fonts/<dir>
/opt/kde3/share/fonts/<dir>
/usr/local/share/fonts/<dir>
~.fonts/<dir>

9.31 Hostname

9.31.1 Setzten

Der Hostname steht in der Datei `/etc/hostname` oder auch `/etc/HOSTNAME`. Man kann zwar diese Datei editieren, aber einfacher geht das mit dem Befehl `hostname`.

```
root@tux>hostname tux1
root@tux>hostname –F /etc/hostname
#hostname
tux1
```

9.31.2 Abfragen

Ebenso kann man sich mit dem Befehl `hostname` den `Hostnamen`, `FQDN` und `Domain` anzeigen lassen.

```
uws@tux>hostname
tux

uws@tux>hostname –f
tux.seabaer.de

uws@tux>hostname –d
seabaer.de
```

9.32 DD

9.32.1 Backup Festplatte

Ein Backup von einer Festplatte kann mittels dd erstellt werden und die erstellte Datei wird gezipt.

```
uws@tux>dd if=/dev/sda bs=32k | gzip -9 > /backup/image.img.gz
```

9.32.2 Restore Image

```
uws@tux>cat /backup/image.img.gz | gunzip –d –c | dd of=/dev/sda bs=32k
```

9.32.3 Backup MBR

```
uws@tux>dd if=/dev/sda of=/backup/mbr.img bs=512 count=1
```

9.32.4 Restore MBR

```
uws@tux>dd if=/backup/mbr.img of=/dev/sda
```

9.33 Battery Status

Informationen über den Batterie Status kann man mit `upower` oder auch mit `acpi` abfragen.

```
uws@tux>upower –i /org/freedektop/UPower/devices/battery_BAT0

root@tux>acpi –I –b

root@tux>cat /sys/class/power_supply/BAT0
```

9.34 Hardware Info

Informationen über die Hardware kann man sich mit dem Befehl h`winfo` sich anzeigen lassen. In dem nächsten Beispiel wird Informationen über die Wlan Karte abgefragt.

```
uws@tux>hwinfo --wlan --short
network:
   wlan0       Intel Centrino Advanced-N 6205 AGN
```

Hardware Items (Auszug):

All, bios, Bluetooth, bridge, camera, cdrom, chipcard, cpu, dist, dsl, joystick, keyboard, memory, modem, monitor, mouse, network, partition, pci, pcmcia, pppoe, printer, scanner, scsi, sound, usb, usb-ctrl, wlan, xen, zip.

Bei Arch Linux, RHEL und Debian gibt es das Programm `inxi` für die Hardware Abfrage.

```
uws@tux>inxi –G
```

Option	Beschreibung
-A	Show Audio
-b	Basic output
-B	Battery Data
-c	CPU output
-d	Optical Drive
-D	Hard Disk
-f	All CPU flags
-F	Full output INXI
-G	Graphic Card
-i	WAN IP
-I	Process, uptime, memory
-m	Memory (RAM)
-M	Machine Data
-n	Advanced Network Card
-N	Network Card
-o	Show unmounted Partition
-p	Full Partition Information
-P	Partition Information
-r	Distro repository data
-R	Show Raid Info
-u	Show Partitions UUID

9.35 Spracheinstellung

Die eingestellte Sprache kann man in der Variable LANG abfragen.

```
uws@tux>echo $LANG
de_de.UTF-8
```

9.36 Benachrichtigungen

Möchte man eine Benachrichtigung ausgeben, die im Panel angezeigt wird, so kann man das mit `notify-send` erledigen. Ein Zeilenumbruch kann man mit `\n` vornehmen.

```
uws@tux>notify-send “<Überschrift>“ “<HinweisText>“ [-t <dauer>]

uws@tux>notify-send “Hinweis“ “`cat /etc/passwd`“ –t 5000
```

Folgende Optionen gibt es:

-t, --expire-time=	Angabe der Zeit in Millisekunden
-u, --urgency=	Dringlichkeitsstufe (low, normal, critical)
-i, --icon=	Angabe eines Icons

10. Scripting

10.1 Datum in Bash Scripte

In einem Bash Script kann man sich sein eigenes Datumsformat zusammenstellen.

```
uws@tux>export stamp=$(date +%a)
uws@tux>echo $stamp
Do

uws@tux>export stamp=`date +%Y_%m_%d`
uws@tux>echo $stamp
2010_06_17

uws@tux>printf "`date +%Y_%m_%d` - Time\n"
2010_06_17 – Time

uws@tux>cat example.sh
#!/bin/bash
# Example for Date
#
DATUM=`date +%Y_%m_%d`
stamp=$(date +%a)

find . –type –f \( -name '*.jpg' \) –exec zip ${Datum}_jpg.zip {} \;
mv example.txt example.txt.$stamp
```

Formatierung der "date" Ausgabe

%H	Stunden (00 bis 23)
%I	Stunden (01 bis 12)
%M	Minuten (00 bis 59)
%S	Sekunden (00 bis 60)
%p	Vor- oder Nachmittag (AM oder PM)
%r	Zeitangabe, 12 Stunden (hh:mm:ss AM/PM)
%R	Zeitangabe, 24 Stunden (hh:mm:ss), entspricht damit %H:%M
%s	Sekunden seit dem 1. Januar 1970 (Unix Zeit)
%Z	Aktuelle Zeitzone (CEST, GMT, ..)
%a	Wochentag in Kurzform (Son, Mon, Die, ..)
%A	Wochentag in Langform
%b	Monat in Kurzform (Jan, Feb, ..)
%B	Monat in Langform
%d	Tag in zweistelliger Zahl
%e	Tag (einstellig mit Leerzeichen)
%D	Datum in der Form mm/dd/yy
%j	Zeigt, der wie vielte Tag im angegebenen Jahr ist
%u	Zeigt, welcher Wochentag es ist (1 bis 7)
%U	Zeigt, wie viele Wochen im angegebenen Jahr es sit
%m	Monat, zweistellig
%y	Jahr, zweistellig
%Y	Jahr, vierstellig
%%	Druckt das Procentzeichen selbst
%n	Zeilenende
%t	Tabulator

10.2 Variablen definieren

Jede Variable hat einen Namen und einen Wert. Typen kennt die Bash nur eingeschränkt. Um eine Variable in einem Bash Script zu definieren, kann der Befehl `declare` benutzt werden.

```
uws@tux>wert=3+7
uws@tux>echo $wert
3+7
uws@tux>declare –i wert
uws@tux>echo $wert
3+7
uws@tux>wert=3+7
uws@tux>echo $wert
10
```

Attribut	Beschreibung
-p	Zeigt alle Attribute aller Variablen an.
-i	Arithmetische Auswertung
-A	Arrays
-r	Nach der Zuweisung kann dieser Wert nicht mehr verändert werden.

Weitere Informationen hierzu gibt es auf der Seite http://tldp.org/LDP/abs/html/declareref.html.

```
uws@tux>strMachine=”Computer Name: `uname –a | cut –d ‘ ‘ –f 2`”
Computer Name: tux
```

Die auszuführenden Befehle stehen zwischen Rückwertigen Hochkommas.

Auch mit `typeset` werden Variablen mit einem Typ definiert. Standardmäßig werden auch hier Variablen ohne Angabe des Typs als `String` Variable deklariert. Die Angabe eines Wertes ist optional.

Die Syntax für `typeset` ist:

```
typeset [option] [variable] [=wert]
```

```
uws@tux>cat example1.sh
#!/bin/bash
typeset –i var1=2 # Variable var1 als integer mit einem wert

typeset +i var1   # abschalten der definition
```

Typeset Options

Option	**Beschreibung**
-a	Array
-i	Integer
-r	Konstante (read only)
-x	Variablen exportieren
-f	Zeigt Funktionen mit ihrer Definition an
-fx	Exportiert eine funktion
-F	Zeigt Funktionen ohne ihre Definition an.

Ein Array wird mit `declare –A` definiert. Die Syntax für die Zuweisung von Werten ist folgenmaßen.

```
declare –A <arrayname>
arrayname=(wert1 wert2 wert3)
arrayname[indexnr]=wert
```

Wird keine Index Nr. bei der Zuweisung der Werte angegeben, so startet die Index Nr. bei 0 für den ersten Wert.

Die Länge eines Strings, die einer Variablen zugewiesen worden ist, kann man folgendermaßen ermitteln.

```
uws@tux>cat varlength.sh
#!/bin/bash
A=“$1“
X=${#A}
echo –n “*“ # Ausgabe ohne newline
printf “ Länge der Variable A: ${X}\n“

uws@tux>./varlength.sh Weihnachten
* Länge der Variable A: 11
```

10.3 Ausgabe auf der Shell

Möchte man Text auf der Konsole ausgeben, so kann man das mit dem Befehl `echo` erledigen. Den Befehl `echo` kann man nicht nur in einem Script verwenden, sondern auch direkt in einer Shell, wenn man zum Beispielt den Inhalt einer Variable sich anzeigen lassen möchte.

```
uws@tux>echo „Connected user: $USER”
Connected user: uws
```

Gibt man hinter dem Befehl `echo` die Option `–n` an, so bleibt der Cursour hinter den auszugebenden Text stehen. Die Option `–e` erlaubt die `Backslash Escapes`. Mit dem Befehl `read`, kann man die Eingabe dann in einer Variablen abspeichern.

```
uws@tux>cat eingabe.sh
#!/bin/bash
#
# Einlesen einer Benutzereingabe
#
echo –n –e “\tBitte Mysql Password eingeben: “
read strPasswd
```

Ohne eine zusätzliche Option bei dem Befehl `read`, wird die Eingabe nach einem Return in der Konsole wiederholt. Soll die Eingabe nicht angezeigt werden, so gibt man hierzu die Option `–s` an. Diese Option unterdrückt das Echo auf der Konsole.

Die Option `–n<zahl>` liest die angegebene Anzahl an Zeichen ein.

```
uws@tux>cat eingabe1.sh
#!/bin/bash
#
# Einlesen einer Benutzereingabe
#
echo –n “Bitte Mysql Password eingeben: “
read –s –n10 strPasswd
```

In Bash Scripten kann man den Befehl `printf` verwenden. Mit diesem Befehl ist es möglich, Zeilenvorschübe und auch Tabs zu verwenden. Einen Zeilenvorschub wird mit der Option `\n` und mit `\t` wird ein Tab erzeugt.

```
uws@tux>cat Ausgabe.sh
#!/bin/bash
#
# Ausgabe von Text
#
clear
printf “\n\n\t\t ****************************************“
printf “\n\n\t\t  Dieses Programm wir ausgefuehrt“
printf “\n\t\t  mit freundlicher Untestuetzung von“
printf “\n\t\t\      Seab@er Software AG“
printf “\n\n\t\t ****************************************\n“
```

`Printf` wurde wurde von der Programmiersprache C entliehen. Der Aufbau ist `“%Format“ Daten`. In der nachfolgenden Tabelle sind die Formatierungen aufgelistet.

Format	**Beschreibung**
%5.2f	Fließkomma mit fünf Stellen vor dem Komma und 2 danach.
%.10s	Zeichenkette mit maximal 10 Stellen.
%X\n	Hexadezimal mit Großbuchstaben.
%y\n	Hexadezimal mit Kleinbuchstaben.
%#X\n	Hexadezimal mit Großbuchstaben und führenden 0X
%i\n	Ganzzahl
%s	Zeichenkette (string)

```
uws@tux>cat format.sh
#!/bin/bash
a=456.863
b=387,162
c=“Ohne_Unterstrich_keine_Ausgabe“

printf “Wert a: %5.2f\n“ `echo $a | tr . ,`
printf “Wert b= %5.2f\n“ $b
printf “Wert c: %.30s\n Wert d: %.30s“ $c “So geht es ohne _“

uws@tux>./format.sh
Wert a: 456,863
Wert b: 387,162
Wert c: Ohne_Unterstrich_keine_Ausgabe
Wert d: So geht es ohne _
```

Möchte man den Text in Farbe ausgeben, so wird hierzu das Zeichen `ESC (\033)` verwendet. Nach dem `Escape` Zeichen wird die Farbe angegeben. Die Angabe bezieht sich dann nicht nur auf die Zeile, sondern auf alle Ausgaben von Texten. Deshalb soll man nach dem Text wieder die ursprüngliche Farbe eingestellt werden.

```
uws@tux>cat ColorAusgabe.sh
#!/bin/bash
#
# Ausgabe von Text in Farbe
#
clear
printf “\033[34m\n\n\t\t ****************************************“
printf “\033[33m\n\n\t\t  Dieses Programm wir ausgefuehrt“
printf “\n\t\t  mit freundlicher Untestuetzung von“
printf “\n\t\t\      Seab@er Software AG“
printf “\n\n\t\t ****************************************\n“
printf “\033[0m“ # ursprüngliche Farbe einstellen.
```

Die folgende Tabelle zeigt die Zuordnung der Werte an.

Wert	Farbe
01m	Fett
04m	Unterstreichen
05m	Blinkend
07m	Vorder- und Hintergrundfarbe vertauscht
22m	Normale Intensität wiederherstellen
30m	Vordergrund schwarz
31m	Vordergrund rot
32m	Vordergrund grün
33m	Vordergrund braun (gelb)
34m	Vordergrund blau
35m	Vordergrund magenta
36m	Vordegrund cyan
37m	Vordergrund weiß
40m	Hintergrund schwarz
41m	Hintergrund rot
42m	Hintergrund grün
43m	Hintergrund braun
44m	Hintergrund blau
45m	Hintergrund magenta
46m	Hintergrund cyan
47m	Hintergrund wei0
49m	Voreingestellter Hintergrund

Die Farbe `Gelb` kann man mit der Angabe von `01m` ausgeben.

```
uws@tux>printf “\033[01m\033[33mFarbe Gelb \033[33mund nun Braun.“
```

Eine Text Box kann man mit dem folgenden Script erstellen.

```
uws@tux>cat textbox.sh
#!/bin/bash
box()
{
 printf '*%.0s' $(seq 1 67) # Print * with length 67
 printf "\n"
}
content()
{
 printf "*"
 if [ $# -eq 0 ]; then
  printf ' %.0s' $(seq 1 65)
 elif [ $# -eq 1 ]; then
  TEXTR="$1"
  printf ' %.0s' $(seq 1 17)
  printf "${TEXTR}"
  printf ' %.0s' $(seq 1 $[48-${#TEXTR}])
 elif [ $# -eq 2 ]; then
  TEXTL="$1"
  TEXTR="$2"
  printf ' %.0s' $(seq 1 3)
  printf "${TEXTL}"
  printf ' %.0s' $(seq 1 $[14-${#TEXTL}])
  printf "${TEXTR}"
  printf ' %.0s' $(seq 1 $[48-${#TEXTR}])
 fi
 printf "*\n"
}

SCRIPTVERSION="13.01.07"

IFS=$'\012'

box
content
content "Script:" "$(basename $0)"
content "Version:" "${SCRIPTVERSION}"
content
content "This script is an example."
content
box

uws@tux>./textbox.sh
**********************************************************************
*                                                                    *
*  Script:       textbox.sh                                          *
*  Version:      13.01.07                                            *
*                                                                    *
*                This script is an example.                          *
*                                                                    *
**********************************************************************
```

10.4 Backup Script

Mit diesem Script wird ein Backup von einem Verzeichnis erstellt und die Sicherungsarchive werden hochgezählt. Ist das Sicherungsverzeichnis nicht vorhanden, so wird es angelegt.

```
uws@tux>cat backup.sh

#!/bin/bash
#
# Seab@er Software AG Backup Script
#
BACKUPDIR=/daten/backup
SOURCEDIR=/daten/bilder
TIMESTAMP=backup-timestamp.dat

# Backup Verzeichnis vorhanden?

If [ ! –d ”${BACKUPDIR}” ]; then
      echo ”Das Verzeichnis ${BACKUPDIR} ist nicht vorhanden”
      echo ”und wird nun angelegt! ”
      mkdir –p ${BACKUPDIR}”
fi
set -- ${BACKUPDIR}/backup-???.tgz  # Alle Backups einlesen in $1, $2 usw.
lastname=${!#}                      # Letzter Backup Name
backupnr=${lastname##*backup-}      # Pfad und backup- entfernen
backupnr=${backupnr%%.*}            # Alles hinter dem ersten „.“ Entfernen
backupnr=${backupnr//\?/0}          # Keine Backups vorhanden, dann 0
backupnr=$[10#${backupnr}]          # Fuehrende Nullen entfernen, (siehe in
                                    # der Bash-Manpage unter base#

if [ ”$[backupnr++]” –ge 999]; then # Erhoehen des Wertes um 1
      echo ”Error: Schon 999 Backups vorhanden! ”
      exit 1
fi

backupnr=000${backupnr}             # Nullen voranstellen
backupnr=${backupnr: -3}            # Die letzten 3 Ziffern heruasschneiden,
                                    # das Leerzeichen vor dem – ist nötig.
filename=backup-${backupnr}.tgz
echo ”Sichere veränderte daten in ${filename}.”

# Es erfolgt ein inkrementelles Backup, Schalter –g

tar –czf ${BACKUPDIR}/${filename} –g ${BACKUPDIR}/${TIMESTAMP} ${SOURCEDIR}
```

10.5 Set-Befel im Bash Script

Der Befehl `set --` … in einem Bash Script weist alle folgenden Argumentetn den Variable `$1, $2` usw. zu. Den Inhalt der letzten belegten Variable kann man dann mit `${!#}` auslesen.

```
uws@tux>cat MeinScript.sh

#!/bin/bash
# Script fuer den set Befehl
#
SOURCEPATH=/home/uws/Dateien

set -- {SOURCEPATH}/bild_???.jpg    # Einlesen der Dateien in $1, $2 usw
lastname=${!#}                      # Letzter Name steht hier.
echo “Lastname: ${lastname}“        # Ausgabe auf dem Schirm
```

Mit `set –x` kann man in der Shell oder auch in einem Script sich die Verarbeitungsschritte sich anzeigen lassen. Mit `set +x` wird dieses wieder abgeschaltet.

10.6 Programm Parameter auswerten

Möchte man bei einem selbst erstellten Script einen Parameter übergeben, so erfolgt die Auswertung in diesem Script mittels einer `case`, `if` oder `while` Schleife. Die auszuwertende Variablen fangen immer mit dem $ an. In der Variable `$0` steht der Name des Scripts, inklusive des Verzeichnisses. Die Variablen `$1` bis `$9` enthalten dann die Parameter zum Auswerten. In der Variable $# steht die Anzahl der übergebenen Parameter. Die Variablen Nummer können auch mit `{}` kenntlich gemacht werden, wie zum Beispiel mit `${1}`.

Zahlen werden nicht mit =, >=, < oder <= verglichen, sondern mit `–eq`, `-ge`, `-gt`, `-ne`, `-lt` oder `–le`.

Nachfolgend sind ein paar mögliche Bedingungen aufgelistet.

Ausdruck	**Beispiel**	**Erklärung**
-a Datei	[-a uschi.txt]	Wahr, wenn die Datei vorhanden ist.
-d Verzeichnis	[-d /home]	Wahr, wenn das Verzeichnis vorhanden ist.
-e Datei	[-e uschi.txt]	Wahr, wenn die Datei vorhanden ist.
-f Datei	[-f uschi.txt]	Wahr, wenn die Datei vorhanden ist.
-G Datei	[-G uschi.txt]	Wahr, wenn die Datei vorhanden ist und eff. Gruppen-ID gehört.
-g Datei	[-g uschi.txt]	Wahr, wenn die Datei vorhanden ist und set-group-id gesetzt ist.
-k Datei	[-k uschi.txt]	Wahr, wenn das Sticky Bit gesetzt ist.
-L Datei	[-L uschi.sh]	Wahr, wenn es sich um ein Symbolischen Link handelt.
-O Datei	[-O uschi.sh]	Wahr, wenn Datei vorhanden ist und eff User-ID gehört.
-r Datei	[-r uschi.sh]	Wahr, wenn die Datei lesbar ist.
-S Datei	[-S uschi.sh]	Wahr, ob die Datei ein Socket ist.
-s Datei	[-s uschi.txt]	Wahr, wenn die Datei größer 0 ist.
-u Datei	[-u uschi.sh]	Wahr, wenn das Bit set-ser-id gesetzt ist.
-w Datei	[-w uschi.txt]	Wahr, wenn die Datei existiert und Schreibzugriffe erlaubt sind.
-x Datei	[-x uschi.sh]	Wahr, wenn die Datei existiert und die Ausführung erlaubt ist.
-n String	[-n "$name"]	Wahr, wenn die Variable nicht leer ist.
str1 = str2	["$1" = "uwe"]	Wahr, wenn beide Werte identisch sind.
str1 != str2	[“$1“ != “uwe“]	Wahr, wenn die Werte ungleich sind.
-z str1	[-z “$1“]	Wahr, wenn der Wert leer ist.
-n str1	[-n “$1“]	Wahr, wenn der Wert nicht leer ist.
z1 –eq z2	[1 –eq $summe]	Wahr, wenn beide Zahlen gleich groß sind.
z1 –lt z2	[10 –lt $1]	Wahr, wenn die erste Zahl kleiner als die zweite ist.
z1 –gt z2	[20 –gt $1]	Wahr, wenn die erste Zahl größer ist als die zweite Zahl.
z1 –ne z2	[$1 –ne 10]	Wahr, wenn beide Zahlen ungleich sind.
!ausdruck	[! 1 –eq $1]	Wahr, wenn der Ausdruck falsch ist.

Kommandozeilen-Parameter werden mit $ abgefragt. In der nachfolgenden Tabelle ist eine Auflistung der wichtigsten Parameter.

Parameter	**Beschreibung**
$0	Name des ausgeführten Shell Scripts, incl. des Pfades.
$#	Anzahl der übergebenen Parameter.
$1 $2 $3 ..	Erster, zweiter, dritter ….. Parameter.
$*	Alle Kommandozeilen-Parameter ($1 $2 $3 ..).
$@	Wie $*
"$@"	Expandiert zu: $1 $2 $3 …
$$	Prozess Nummer der Shell.
$-	Ausgabe der aktuellen Shell Optionen.
$?	Ausgabe des Return-Codes von dem letzten ausgeführten Kommando.
$!	Prozess Nummer des zuletzt ausgeführten Hintergrund Prozesses.
$Host	Ausgabe der Umgebungsvariablen HOST.

10.6.1 Shell Script Testen

Ein Shell Script kann man mit sh <option> testen. In der nachfolgenden Auflistung sind einige Möglichkeiten beschrieben.

sh –n <scriptname>	Syntax Test. Alle Kommandos werden gelesen, aber nicht ausgeführt.
sh –v <scriptname>	Ausgabe der Shell Kommandos in der gelesenen Form.
sh –x <scriptname>	Ausgabe der Shell Kommandos, nach Durchführung aller Ersetzungen.

10.6.2 Case Anweisung

Mit der Anweisung `case` können die Kommandozeilen-Parameter ausgewertet werden. Das Ende einer Auswertung wird mit einem doppelten Semikolon abgeschlossen, da ein Semikolon als Trennzeichen für Kommandos gilt. Eine `case` Anweisung wird mit einem `esac` (end case) abgeschlossen.

```
#! /bin/bash
# Als erstes wird der Parameter übergeben
action=$1

# Hier fängt die Auswertung an.
case "$action" in
      -a)
            <Befehle>
            ;;
      -v)
            <Befehle>
            ;;
# Mehrfachauswertung
      -j*|-J*|-y*|-Y*)
            <Befehle> ;;
      *)
            echo " Usage: uschi.sh [start] [stop] "
esac
exit 0
```

10.6.3 IF Anweisung

Möchte man eine Bedingung abfragen, so nutzt man hierzu eine `if` Abfrage. Die allgemeine Form der `if` Abfrage ist:

```
if Bedingung
 then Anweisung
 else Anweisung
fi
```

Außerdem können mehrere `else if` Abfragen erstellt werden. Der Befehl lautet hierzu `elif`.

```
if Bedingung1
 then Anweisung
 elif Bedingung2
  then Anweisung
 else Anweisung
fi
```

Um abzufragen, ob eine Datei vorhanden ist, so nutzt an hierzu die Option `test –r`.

```
uws@tux>cat exist.sh
#! /bin/bash
if test –r /home/uws/scripts/sicherung.sh
then
 bash /home/uws/scripts/startmount.sh
fi
```

Handelt es sich nicht um eine Datei, sondern um ein Verzeichnis, so wird hierzu die Option `–d` angegeben. Die Option `–L` prüft, ob es sich um ein symbolischen Link handelt. Mit der Option `–f` wird überprüft, ob es sich um eine gewöhnliche Datei handelt. Man kann mit `test` auch mehreren Überprüfungen ausführen. Dafür wird die Option `–a` (für and / und) und die Option `–o` für oder benutzt.

```
uws@tux>cat exist1.sh
#! /bin/bash
if test –r /home/uws/scripts –a –f /home/uws/scripts
then
 bash /home/uws/scripts/startmount.sh
fi
```

Anstelle von `test` kann auch die Überprüfung mittels der eckigen Klammern erfolgen.

```
uws@tux>cat exist2.sh
#! /bin/bash
if [ –r /home/uws/scripts –a –f /home/uws/scripts ]
then
 bash /home/uws/scripts/startmount.sh
fi
```

```
#! /bin/bash
if [ "$1" = "tux" ]; then
 echo "Hallo Pinguin."
else
 echo "Hallo $1, es wurde die \$1 Variable ausgelesen."
fi
exit 0
```

Überprüfen, ob eine nicht Datei existiert.

```
uws@tux>cat exist3.sh
#!/bin/bash
strUser=$1
if [ ! –r ~/.dbuser/${strUser} ]; then
 echo “File not exist“
else
 echo “File exist“
fi
```

Existiert das Verzeichnis nicht, so wird es angelegt.

```
if [ ! –d ~/.dbuser ]; then
 mkdir ~/.dbuser
fi
```

Ist die Variable leer? Bei –n => wahr, wenn gefüllt, -z => wahr, wenn leer.

```
if [ -z ”${strV1}” ]; then
 printf “\nVariable is empty.\n“
fi
```

```
#!/bin/bash
Var1=””
[ -z “{VAR1}“ ] && echo “Variable leer: Ja“ || echo “Variable leer: Nein“
```

10.6.4 While / Until Schleife

Die Prüfung der Bedingung erfolgt bei `while` vor der Abarbeitung der Anweisung und bei `until` erst nach der Abarbeitung.

```
#! /bin/bash
count=0
while [ $count –le 10 ]
 do
  echo $count
  count=$[count+1] # Zähler um 1 hochzählen
 done
exit 0
```

```
#! /bin/bash
PING_HOST=$1
printf “\nWaiting for network “
while [ $(ping –w1 –c1 $PING_HOST | grep –c “0 received”) –eq 1 ]; do
 printf “.”
done

printf “\n\nNetwork is now up!\n”
for i in $(grep noauto /etc/fstab | grep –o “[^]*”; do
 mount $i
done
```

```
#!/bin/bash

while [ “$1“ != ““ ]
 do
  [ “$1“ == “-b“ ] && BACKUP=true && printf “Backup now!\n“ && shift
  [ “$1“ == “-r“ ] && RESTORE=true && printf “Restore or what else.\n“ &&
shift
  [ “$1“ == “-h“ ] && HELPNEED=true && printf “Help needed?\n“ && shift
 done
```

10.6.5 For Schleife

Die `for` Schleife erhält eine Liste von Werten und führt die Kommandos mit jedem Wert aus. Möchte man eine Schleife abbrechen, so wird hierzu der Befehl `break [n]` ab. Um wieder am Anfang zu springen, benutzt man den Befehl `continue [n]`.

```
uws@tux>cat example.sh
#! /bin/bash
for a in $1
do
 printf “\n\tUebergabe Wert: ${a}\n”
done
```

Beispiel für `break` und `continue`.

```
uws@tux>cat example1.sh
#! /bin/bash
for a in 1 2 3 4 5
do
 if [ ${a} –eq 2 ]
 then
  continue
 fi
 if [ ${a} –eq 4 ]
 then
  break
 fi
 printf “\n\tWerte: ${a}\n”
done
```

Siehe auch Abschnitt **3.8**.

```
uws@tux>cat example2.sh
i=1
wochentage=“Montag Dienstag Mittwoch Donnerstag Freitag“
for tage in ${wochentage}
do
 printf ”Wochentag $((i++)) : ${tage}\n“
done

uws@tux>./example2.sh
Wochentag 1 : Montag
Wochentag 2 : Dienstag
Wochentag 3 : Mittwoch
Wochentag 4 : Donnerstag
Wochentag 5 : Freitag
```

```
uws@tux>cat example3.sh
i=1
cd ~/bin
for item in *
do
 printf ”Item $((i++)) : ${item}\n“
done

uws@tux>./example3.sh
Item 1 : example.sh
Item 2 : example1.sh
Item 3 : example2.sh
```

```
Item 4 : example3.sh
```

```
uws@tux>cat example4.sh
i=1
for file in /etc/[abcd]*.conf
do
 printf “File $((i++)) : ${file}\n“
done

uws@tux>./example4.sh
File 1 : /etc/alsa-pulse.conf
File 2 : /etc/asound-pulse.conf
File 3 : /etc/autofs_ldap_auth.conf
File 4 : /etc/blkid.con
File 5 : /etc/dhcpclient.conf
File 6 : /etc/dnsmasq.conf
```

```
uws@tux>cat example5.sh
i=1
for file in $(ls ~/bin/*.sh)
do
 printf “File $((i++)) : ${file}\n“
done

uws@tux>./example5.sh
File 1 : /home/uws/bin/example.sh
File 2 : /home/uws/bin/example1.sh
File 3 : /home/uws/bin/example2.sh
File 4 : /home/uws/bin/example3.sh
File 5 : /home/uws/bin/example4.sh
File 6 : /home/uws/bin/example5.sh
```

Beispiel einer for Schleife in einer Zeile.

```
uws@tux>for a in 01 02 03; do ls bild${a}.jpg; done
bild01.jpg
bild02.jpg
bild03.jpg
```

Und noch eine for Schleife.

```
uws@tux>cat example6.sh
#!/bin/bash
for ((i=1;i>=5;i++)); do
 echo “$i: “
done

# von 1 bis 10
for i in {1..10}
do
 printf “Number: $1“
done

#von 1 bis 10, aber in 2er Schritten
for i in {1..10..2}
do
 printf “Number: $1“
done
```

Einlesen einer Datei und nur einzelne Spalten ausgeben.

```
uws@tux>cat example7.sh
#!/bin/bash
for SHARE in $(grep –i “nfs“ /etc/fstab | cut –d‘ ‘ –f2)
do
 printf “\nShare to mount: $SHARE“
done
```

10.6.6 Function

In einem Bash Script müssen die `Functions` am Anfang definiert werden, da sie sonst nicht gültig sind. `Functions` können auch in einer separaten Datei stehen, die dann per `.<pfad>/<datei>` geladen werden. Mit `source <pfad/Datei>` kann man die Datei auch laden. Die Angabe von `function` vor dem Namen ist nicht zwingend, man kann sie auch weglassen.

```
uws@tux>cat example5.sh
#! /bin/bash
function <name>()
{
      Befehle
}
```

10.6.7 Benutzereingabe

Möchte man den Benutzer eine Eingabe ermöglichen, so wird hierzu der Befehl `read` verwendet. Werden mehr Worte als definierte Variablen eingegeben, so bekommt die letzte Variable den Rest des Textes. Werden wenige Worte eingegeben, so bleiben die letzten Variablen leer.

```
uws@tux>cat example6.sh
#! /bin/bash
printf "\nBitte Werte eingeben: \n"
read str1 str2 str3 str4
printf "\nFolgende Werte wurden eingegeben: ${str1} ${str2} ${str3}
${str4}\n"
```

Für `read` gibt es folgende Optionen:

-n <anzahl>	Maximal Anzahl an Zeichen. Wurde die maximale Anzahl erreicht, so wird die Eingabe mit einem Enter automatisch abgeschlossen.
-s	Silent Mode, die Eingabe wird nicht angezeigt. Z.B. bei einer Password Abfrage
-t <sec>	Hiermit wird ein Timeout für die Eingabe gesetzt. Der Rückgabewert bei keiner Eingabe ist dann 0.
-p	Ausgabe eines Textes

```
uws@tux>read –n1 –s –p "Press any key" CHOISE
```

Wenn nach `read` keine Variable angegeben wird, so kann man den eingegebenen Wert mit `reply` auswerten.

```
uws@tux>cat example7.sh
#!/bin/bash
# Default value for read is reply.

printf "\nWas ist deine Lieblingsfarbe? \n"
read
printf "\nDeine Lieblingsfarbe ist: $REPLY\n"
printf "\nWas ist dein Lieblingsverein? \n"
read VEREIN
printf "\nDein Lieblingsverein ist: $VEREIN\n"
printf "und deine Lieblingsfarbe ist: $REPLY\n"
exit 0
```

10.6.8 Teilstring

Einen String kann man mit verschiedenen Techniken zerlegen. Mit % wird der String von rechts abgeschnitten und bei # von links.

```
uws@tux>FOO='1234567890'
uws@tux>echo ${FOO:0:6}
123456

uws@tux>echo ${FOO:2:3}
345

uws@tux>FOO=mein.txt.dat
uws@tux>echo ${FOO%.*}
mein.txt

uws@tux>echo ${FOO%%.*}
mein

uws@tux>FOO=/daten/test/mein.txt.dat # schneidet bis zum ersten / ab.
uws@tux>echo ${FOO#*/}
daten/test/mein.txt.dat

uws@tux>echo ${FOO#*.} # schneidet alles ab, bis zum ersten Punkt
txt.dat

uws@tux>echo ${FOO##*/} # schneidet alles ab, bis zum letzten /
mein.txt.dat

uws@tux>echo ${FOO##*.} # schneidet alles bis zum letzen Punkt ab
dat

uws@tux>BAT_STAT=99%
uws@tux>BAT_STAT=${BAT_STAT%%%*}
uws@tux>echo $BAT_STAT
99
```

10.6.9 Eval

Mit Eval ist es möglich, Kommandos auszuführen, als würden sie in der Shell ausgeführt.

```
uws@tux>cat commands.sh
#!/bin/bash
while true
do
 printf “\tCommand: “
 read
 eval $REPLY
done

uws@tux>./commands.sh
      Command: ls *.sh
commands.sh sampleEval.sh
      Command: echo $$
16061
      Command: exit
uws@tux>
```

Es besteht die Möglichkeit, indirekt auf eine Variable zugreifen zu können. In dem nächsten Beispiel wird dieses gemacht.

```
uws@tux>cat sampleEval.sh
#!/bin/bash
Mo=backupMo
Di=backupDi
Mi=backupMi
Do=backupDo
Fr=backupFr
Sa=backupSa
So=backupSo

tag=`date +“%a“`
eval backup=\$$tag
printf “\n\tDas Backup Script: $backup wird ausgeführt.\n\n“
./$backup

uws@tux>./sampleEval.sh

      Das Backup Script: backupDi wird ausgeführt.
./backupDi
```

Im ersten Durchlauf wird aus `\$$tag $Di`. Die Variable `$Di` hat den Wert `backupDI` und dieses wird im zweiten Durchlauf der Variable `backup` zugewiesen.

10.6.10 Auswahlmenu mit Select

Mit einem `select` in einem Script kann man ein Auswahlmenu darstellen.

```
uws@tux>cat sampleSelect.sh
#!/bin/bash
declare –A WERT # Array deklariert
WERT[“Montag Backup”]=”backupMo.sh”
WERT[“Dienstag Backup”]=”backupDi.sh”
WERT[“Mittwoch Backup”]=”backupMi.sh”

printf “\n\tBitte Backup auswählen: \n\n”
#
# Das Ausrufezeichen vor der Array Variable gibt die Array Index No.
# aus und das @ Zeichen zwischen den [] gibt alle Index No. aus.
#
select BACKUP in “${!WERT[@]}”;
do
 BACKUPSCRIPT=${WERT[${BACKUP}]}
 TITLE=”${BACKUP}”
 printf   “\n\tDas   ${TITLE}   mit   dem   Script   ${BACKUPSCRIPT}   wird
ausgeführt.\n\n”
 ~/bin/${BACKUPSCRIPT}
 exit 0
done

uws@tux>./sampleSelect.sh

      Bitte Backup auswählen:

1) Dienstag Backup
2) Montag Backup
3) Mittwoch Backup
#? 2

      Das Montag Backup wird mit dem Script backupMo.sh ausgeführt.
```

Ein zweites Beispiel für eine `select` Anweisung.

```
uws@tux>cat sampleSelect1.sh
#!/bin/bash
WDR2=”http://www.wdr2.de”
SPORT=”http://www.sportschau.de”

printf “\n\tBitte Web Seite auswählen: \n\n”
select A in WDR2\ Seite Sportschau\ Seite;
do
 case $A in
  WDR*)
   printf “\n\tFolgende URL wird aufgerufen: ${WDR2}.\n\n”
   firefox ${WDR2}
   exit 0
   ;;
  Sport*)
   printf “\n\tFolgende URL wird aufgerufen: ${SPORT}.\n\n”
   firefox ${SPORT}
   exit 0
   ;;
 esac
```

```
done

uws@tux>./sampleSelect1.sh

      Bitte Web Seite auswählen:

1) WDR2 Seite
2) Sportschau Seite
#? 1

      Folgende URL wird aufgerufen: http://www.wdr2.de.
```

10.7 Datei zur Laufzeit erstellen

Eine Datei zur Laufzeit kann mit der sogenannten `Here`-Dokumentation erstellt werden. Als Trenner kann jede Zeichenfolge verwendet werden, meistens wird `EOF` als Trenner verwendet. Alles was zwischen den Trenner steht, wird in eine neue Datei ausgegeben.

```
uws@tux>cat create1.sh
#!/etc/bash
cat <<EOF > "NewScript"
#!/bin/bash
# Laufzeiterstelltes Script
#
printf „\n\tHier geht es los!\n"
Ls –lash
EOF
```

Wird in einem Script Taps verwendet, so muss ein `'-'` Zeichen vor dem Trenner gesetzt werden. Dann werden die Taps ignoriert.

```
uws@tux>cat create2.sh
#!/etc/bash
cat <<-EOF > "NewScript"
#!/bin/bash
# Laufzeiterstelltes Script mit Taps
#
if test –r /home/uws/bin/test.txt
then
      printf „\n\tHier geht es los!\n"
      Ls –lash
fi
EOF
```

10.8 Dialog Aufruf

In der Shell kann man grafische Dialoge aufrufen, um in einer Interaktion mit dem Anwender zu treten. Die grafischen Dialoge können mit dem Befehl `dialog` auch in einer `SSH Sitzung` aufgerufen werden, da der Befehl keinen X-Server braucht. Die Syntax lautet: `dialog [optionen] [dialog_aufruf] "Text" [breite] [höhe]`. Steht ein X-Server zu Verfügung, so kann man den Befehl `xdialog` nehmen. Damit man xdialog verwenden kann, muss die Software von http://xdialog.dyns.net heruntergeladen und installiert werden.

In einer KDE Umgebung kann man die Dialoge mit dem Befehlt `kdialog` definieren. Eine Übersicht der Dialoge erhält man, wenn man kdialog ohne Parameter in der Shell aufruft.

Folgenden Dialog Aufrufe gibt es:

Dialog	**Beschreibung**
--calendar	Kalender
--checklist	Auswahlliste
--form	Formbox
--fselect	Verzeichnis und Dateiauswahl
--gauge	Fortschrittsanzeige
--infobox	Informationsausgabe
--inputbox	Eingabe Dialog
--inputmenu	Menu
--menu	Menu
--msgbox	Informationsausgabe.
--password	Wie inputbox, die Eingabe wird mit Sternchen angezeigt.
--radiolist	Wie checklist, aber hier nur eine Auswahl möglich
--tailbox	Wie tail –f
--textbox	Viewer für Ascii Dateien mit der Möglichkeit zum blättern.
--timebox	Ausgabe der Uhrzeit.
--yesno	Ja / Nein Dialog

Optionen	**Beschreibung**
--title	Überschrift für die Dialog Box
--backtitle	Überschrift für den Hintergrund der Shell
--begin	Startposition y x
--cancel-label	Überschreibt das Label des Cancel Buttons
--clear	Der Inhalt des Bildschirms wird nach dem beenden gelöscht.
--exit-label	Überschreibt das Label des Exit Buttons.
--defaultno	Setzt den Fokus auf No,
--extra-button	Zeigt zwischen Ok und Cancel einen extra Button an.
--extra-label	Label Name des extra Buttons.
--print-maxsize	Ausgabe der Maximum Größe der Dialoge, wird auch mit dem Befehl `resize` angezeigt.

Alle Optionen kann man sich in der Manpage anzeigen lassen. Mit `dialog --help` wird auf der Console die Hilfe angezeigt.

Den Return Code, ob `Ok` oder `Abbrechen` gedrückt wurde, kann man mit der Variablen `$?` auslesen. Der Return Code bei `Ok` ist eine `0` und bei `Abbrechen` eine `1`.

10.8.1 Calender

Syntax: dialog --calendar "Text" <höhe> <breite> <tag> <monat> <Jahr>

```
uws@tux>dialog --calendar "Datum:" 2 1 17 08 2010
```

10.8.2 Checklist

Syntax: dialog --checklist “Text“ <höhe> <breite> <zeilen> <einträge> <liste…>

```
uws@tux>dialog --checklist “Bitte auswhälen:“ 20 30 3 \
 01 “Erste Auswahl“ on\
 02 “Zweite Auswahl“ off\
 03 “Dritte Auswahl“ off
```

10.8.3 Form

Syntax: dialog --form “Text“ <höhe> <breite> <form_höhe> <text> <y> <x> <text> <y> <x> <flen> <ilen>

```
uws@tux>dialog --form “INFO“ 10 50 0 “User: “ 1 1 “uws“ 1 10 10 0 \
 “Shell: “ 2 1 “Bash“ 2 10 15
```

10.8.4 Fselect

Syntax: dialog --fselect <diretory> <höhe> <breite>

```
uws@tux>dialog --fselect /var/log 10 20
```

10.8.5 Gauge

Syntax: dialog --Gauge “Text“ <höhe> <breite> <fertigstellung>

```
uws@tux>dialog --gauge “Fertig zu:“ 2 1 50
```

Beispiel eines laufenden Balkens:

```
for I in $(seq 1 100) ; do
 echo $I | dialog --backtitle “Progress Status“ --gauge “Fertig zu:” 8 50 0
 sleep 0.01
done
```

10.8.6 Infobox

Syntax: dialog --infobox “Text“ <höhe> <breite>

```
uws@tux>dialog --infobox “Es ist bald Mittag.“ 5 30
```

10.8.7 Inputbox

```
#! /bin/bash
dialog --clear --inputbox "Dein Name bitte:" 10 60 2> eingabe.tmp
clear
echo "Dein Name lautet: `cat eingabe.tmp`"
exit 0
```

Anstelle von `cat` kann man die Datei auch mit < einlesen. Für eine Passwort Eingabe gibt es die `passwordbox`.

10.8.9 Inputmenu

Syntax: dialog --inputmenu “Text“ <höhe> <breite> <menu_höhe> <tag1> <item1> …

```
uws@tux>dialog --inputmenu “Menu:“ 30 50 10 “01“ “Montag“ “02“ “Dienstag“
```

10.8.10 Menu

Syntax: dialog --menu “Text“ <höhe> <breite> <menu_höhe> <tag1> <item1> ….

```
uws@tux>dialog --menu “Menu:“ 30 50 10 “A“ “Taschenrechner“
```

10.8.11 Msgbox

Syntax: dialog --msgbox “Text“ <höhe> <breite>

```
uws@tux>dialog --msgbox “Hier könnte ihr Text stehen.“ 8 30
```

10.8.12 Password

Aufruf und auch die Syntax ist die gleiche wie unter 10.6.7 Inputbox.

10.8.13 Radiolist

Syntax: dialog --radiolist “Text“ <höhe> <breite> <list_höhe> <tag1> <item1> <status1>

```
uws@tux>dialog --radiolist “Auswahl:“ 25 25 20 “A“ “Hund“ “Off“ “02“
“Katze“ “On“
```

10.8.14 Tailbox

Syntax: dialog --tailbox <file> <höhe> <breite>

```
uws@tux>dialog --tailbox “/home/adm_uws/bin/gauge.sh“ 25 70
```

10.8.15 Textbox

Syntax: dialog --textbox <file> <höhe> <breite>

```
uws@tux>dialog --textbox “/home/adm_uws/bin/gauge.sh“ 25 70
```

10.8.16 Timebox

Syntax: dialog --timebox “Text“ <höhe> <breite> <stunde> <minute> <sekunde>

```
uws@tux>dialog --timebox “Aktuelle Zeit:“ 5 30 13 09 22
```

10.8.17 YesNo

Syntax: dialog --yesno “Text“ <höhe> <breite>

```
uws@tux>dialog --yesno “Steuern sparen?“ 5 20
```

10.9 Skriptoptionen

Möchte man in einem Script Optionen übergeben und auswerten, so geschieht das mit `getopts`. Alle Optionen können in beliebiger Reihenfolge gesetzt werden. Erfolgt nach einer Option ein `Doppelpunkt`, wie in dem unteren Beispiel nach dem `f`, so wird für diese Option ein Parameter benötigt.

```
uws@tux>cat opts.sh
#!/bin/bash
while getopts “abf:hv“ Arg ; do
 case ${Arg} in
  a) printf “Die Option –a wurde übergeben.“;;
  b) printf “Die Option –b wurde übergeben.“;;
  f) if [ -f ${OPTARG} ]; then
      filename=${OPTARG}
     fi
     ;;
  h) printf “Die Syntax lautet: ….“;;
  v) verbose=y;;
 esac
done
```

Script Optionen können auch mit einer Kombination von `while` und `case` ausgewertet werden.

```
uws@tux>cat opts1.sh
#!/bin/bash
VERBOSE=“0“
OUTFILE=““

while [ $# -gt 0 ]
do
 case $1 in
  -h|--help)
   printf “\n“
   printf “Folgende Optionen sind zulässig:\n”
   printf “-h|--help           : Diese Hilfe\n“
   printf “-v|--verbose        : Statusinformationen\n“
   printf “-o|--outfile <file> : Ausgabe in <file>\n\n“
   exit 0
   ;;
  -v|--verbose)
   VERBOSE=“1“
   ;;
  -o|--outfile)
   shift # Parameter $2 wird $1
   if [ -z “$1“ ]
   then
    printf “Error: No File Name\n“ 1>&2
    exit 1
   fi
   OUTFILE=”$1”
   ;;
  *)
    printf “Unbekannte Option: $1\n“ 1>&2
```

```
    exit 1
    ;;
  esac
  shift
done
```

10.10 Tabs setzen

Die Abstände für die Taps kann man mit `tabs <wert>` für die Console setzten.

Wert	Beschreibung
-0	Löscht alle Taps
-8	Setzt die default Werte
+4	Tabs mit dem Abstand von 4 setzen
2,6,12	Tabs mit den Abständen 2,6,12 setzten

10.11 Wert gerade/ungerade

Möchte man wissen, ob ein Integer Wert einer Variable gerade oder ungerade ist, so gibt man den Parameter `%2` mit an. Ist der Wert eine `gerade` Zahl, so wird eine `0` als Rückgabewert ausgegeben. Bei einer `ungeraden` Zahl eine `1`.

```
uws@tux>cat example2.sh
#!/bin/bash

DAY=`date +%j´
typeset –i MDAY
MDAY=$DAY%2
printf “\nWert von DAY:  ${DAY}\n“
printf “\nWert von MDAY: ${MDAY}\n“

uws@tux>./example2.sh

Wert von DAY:  326
Wert von MDAY: 0
```

10.12 Befehle mehrzeilig

Für die bessere Übersichtlichkeit in einem Script kann es erforderlich sein, den Befehl in mehrere Zeilen zu schreiben. Dieses kann man mit einem Backslash (\) am Ende der Zeile machen.

```
uws@tux>cat example3.sh
#!/bin/bash
ls \
-lah \
*.txt
```

10.13 Let

Mit `let` kann man Berechnungen durchführen.

```
uws@tux>let “m=4*2014“ && echo –e “\t Ergebnis: $m“
      Ergebnis: 4096

uws@tux>cat example_let.sh
#!/bin/bash
let wert1=20
let wert2=10

let m=$wert1+$wert2 && echo –e “Ergebnis: $M“
```

10.14 Befehle verketten

Befehle lassen sich mit `&&` in einer Zeile verketten. Diese Verkettung ist eine `und` Verbindung. Eine oder Verbindung wird mit zwei Pipe Zeichen (`||`) gemacht. Anstelle von den Pipe Zeichen, kann auch ein Semikolon genommen werden.

```
uws@tux>cat example_command.sh
#!/bin/bash
ask_fragen() {
 for (( i=1;i<=4;i++ )); do
  echo “$i: $1“ && shift
  sleep 2
 done
 sekunden=10
 let timer=10
 while [ “$sekunden“ –gt 0 ]; do
  printf “$timer“ && let sekunden-- && let timer–
  sleep1
 done
 printf “\n“
}
### Main ###
ask_fragen “Was ist die Bash?“ “Was ist KVM?“ “Was ist LVM?“ “???“
```

10.15 Ausgabe Version

Die Ausgabe der Version eines Programms, kann folgenden Inhalt haben.

```
uws@tux>./Backup.sh -V
Backup.sh  1.2
Copyright © 2014 Seab@er Software AG
Lizenz      GPLv3+:      GNU      GPL      Version      3      oder      höher
<http://gnu.org/licenses/gpl.html>
Dies ist eine freie Software: Sie können sie ändern und weitergeben.
Es gibt keinerlei Garantien, soweit wie es das Gesetz erlaubt.

Geschrieben von Uwe Schimanski
```

10.16 Datei einlesen

Eine Datei Zeilenweise einlesen kann mit einem `while read` gemacht werden.

```
uws@tux>cat daten.txt
Das ist Zeile 1.
Das ist Zeile 2.
Das ist Zeile 3.
Das ist Zeile 4.

uws@tux>cat readfile.sh
#!/bin/bash
while read line
do
 echo –e “$line \n”
done < daten.txt
```

Eine Datei einlesen, in der die Daten durch einen Separator getrennt sind.

```
uws@tux>cat readfile1.sh
#!/bin/bash
while IFS=‘:‘ read user pass uid gid full home shell
do
 echo –e “$full:\n\
 PSEUDO: $user\n\
 UID   : $uid\n\
 GID   : $gid\n\
 HOME  : $home\n\
 Shell : $shell\n\n”
done </etc/passwd
```

Die Kommentar Zeilen einer Datei nicht auswerten. In dem nachfolgenden Beispiel werden sie extra ausgegeben.

```
uws@tux>cat readfile2.sh
#!/bin/bash
i=1
while IFS=’;’ SOURCE BACKUP
do
 if [ “${SOURCE:0:1} != ”#” ]; then
  printf ”\tQuell Path: $SOURCE\n”
  printf “\tBackup Path: $BACKUP\n\n”
 else
  printf “\tComment $((i++)); $SOURCE\n\n”
 fi
done <$1
```

Einer Variable kann man folgendermaßen eine Textdatei einlesen.

```
uws@tux>export VALUE=`cat pid.dat`

uws@tux>export VALUE=$(<pid.dat)
```

10.17 Ausgabe Script Name

Die Ausgabe des Script Namens wird mit `basename $0` gemacht.

```
uws@tux>cat script_name.sh
#!/bin/bash
echo Script Name: $(basename $0)
```

10.18 Exit Codes

Jeder erfolgreicher oder auch nicht erfolgreicher Aufruf eines Programms gibt einen Exit Code zurück. Dieser Exit Code kann mit $? Abgefragt werden. In einem Script kann man auch selber Exit Codes zurückgeben.

Eine Auswahl an Default Exit Codes sind in der nachfolgende Liste angegeben.

Exit Code	**Beschreibung**
0	Command wurde erfolgreich ausgeführt
1	Command wurde nicht erfolgreich ausgeführt.
2	Empty Function
126	Command invoked cannot execute
127	Command nicht gefunden
128	Invalid Argument
130	Script terminated mit Control-C
255	Exit Status out of range, nur 0.255 sind erlaubt.

```
uws@tux>cat exitcodes.sh
#!/bin/bash
printf “\nHello World!\n\n“
echo $?     # return 0, command successfully

printfb “\nHello World!\n\n“
echo $?     # return 127, command not found

exit 21

uws@tux>./exitcodes.sh

Hello World!

0
./exitcodes.sh: Zeile 5: printfb: Kommando nicht gefunden.
127

uws@tux>echo $?
21
```

10.19 Network Up

Ein einfacher Netzwerk Test, ob das Netz vorhanden ist.

```
uws@tux>cat netready.sh
#!/bin/bash

HOST_NAME=$1

printf “Warte auf das Netz“

while [ $(ping –w1 –c1 $HOST_NAME | grep –c “0 received“) –eq 1 ]; do
 printf “.“
done
printf “\nNetz ist up.\n“
```

10.20 XARGS

Das Programm `xargs` wird mit Pipes benutzt und dabei werden alle Standardeingaben gelesen und als Argumente an ein anderes Programm weitergegeben. Nachfolgend sind ein paar Beispiele, die den Einsatz von `xargs` zeigen.

```
uws@tux>echo 1 2 3 4 5 6 | xargs
1 2 3 4 5 6

uws@tux>echo 1 2 3 4 5 6 | xargs –n 2
1 2
3 4
5 6

uws@tux>echo 1 2 3 4 5 6 | xargs –p –n 2
echo 1 2 ?... y
1 2
echo 3 4 ?... n
echo 5 6 ?... y
5 6
```

```
uws@tux>ls *.sh | xargs –n 1 echo File:
File: Backup.sh
File: Restore.sh
```

```
uws@tux>cat inhalt.txt
Dieser Text steht in einer Zeile.

uws@tux>xargs --arg-file=inhalt.txt –n 1
Dieser
Text
steht
in
einer
Zeile.
```

```
uws@tux>cut –d: -f1 </etc/passwd | sort | xargs –n 1 echo User:
User: at
User: avahi
User: bin
```

Löschen von Dateien mit einer Abfrage vor dem löschen. Damit auch Dateien gelöscht werden, die ein Leerzeichen enthalten, wird bei `find` die Option `–print0` gesetzt und bei `xargs` die Option `-0`.

```
uws@tux>find . –name “*.sh“ –print0 | xargs -0 –p rm –rf
rm –rf ./Backup.sh ./Restore.sh ?... n
```

10.21 Signale (Traps)

Signale, wie z.B. ein `Ctrl + C` kann man in einem Programm abfangen. Hierzu wird in dem Programm ein entsprechender `Trap` definiert.

Syntax:

```
trap ‘action‘ signal
```

Als Signal kann entweder der Name oder die Nummer angegeben werden. Es können auch mehrere Signal für eine Aktion angegeben werden. Soll ein Signal ignoriert werden, so wird bei `Action` nichts angegeben.

Die nachfolgende Liste listet die für ein Programm relevanten Traps auf.

Signal	Nr.	Beschreibung
SIGHUP	1	Das Programm hängt.
SIGINT	2	Ein Ctrl + C wurde gemacht
SIGQUIT	3	Sendet ein Ctrl + D
SIGFPE	8	Eine illegale mathematische Operation wurde gemacht
SIGKILL	9	Der Prozess wird mit kill beendet
SIGALRM	14	Alarm Clock (für Timer)
SIGTERM	15	Der Prozess wird mit kill beendet

Eine vollständige Liste erhält man, wenn man in der Console ein `kill –l` eingibt.

```
uws@tux>kill –l
 1) SIGHUB       2) SIGINT       3) SIGQUIT      4) SIGILL
 5) SIGTRAP      6) SIGABRT      7) SIGBUS       8) SIGFPE
 9) SIGKILL     10) SIGUSR1     11) SIGSEGV     12) SIGUSR2
13) SIGPIPE     14) SIGALRM     15) SIGTERM     16) SIGSTKFLT
17) SIGCHLD     18) SIGCONT     19) SIGSTOP     20) SIGTSTP
21) SIGTTIN     22) SIGTTOU     23) SIGURG      24) SIGXCPU
25) SIGXFSZ     26) SIGVTALRM   27) SIGPROF     28) SIGWINCH
29) SIGIO       30) SIGPWR      31) SIGSYS      32) SIGRTMIN
35) SIGRTMIN+1  36) SIGRTMIN+2  37) SIGRTMIN+3  38) SIGRTMIN+4
39) SIGRTMIN+5  40) SIGRTMIN+6  41) SIGRTMIN+7  42) SIGRTMIN+8
43) SIGRTMIN+9  44) SIGRTMIN+10 45) SIGRTMIN+11 46) SIGRTMIN+12
47) SIGRTMIN+13 48) SIGRTMIN+14 49) SIGRTMIN+15 50) SIGRTMAX-14
51) SIGRTMAX-13 52) SIGRTMAX-12 53) SIGRTMAX-11 54) SIGRTMAX-10
55) SIGRTMAX-9  56) SIGRTMAX-8  57) SIGRTMAX-7  58) SIGRTMAX-6
59) SIGRTMAX-5  60) SIGRTMAX-4  61) SIGRTMAX-3  62) SIGRTMAX-2
63) SIGRTMAX-1  64) SIGRTMAX
```

Beispiele für Traps in einem Script.

```
uws@tux>cat trap1.sh
#!/bin/bash
# Funktion trap zum Abfangen von Signalen
# Name: trap1.sh
# Siganl SIGINT (2) (Strg + C)
trap 'echo SIGINT erhalten' 2
i=0
while [ $1 -lt 5 ]
do
 echo "Habe Zeit, warte ein wenig!"
 sleep 2
 i=`expr $i + 1`
done
```

```
uws@tux>cat trap2.sh
#!/bin/bash
# Funktion trap zum Abfangen von Signalen
# Hier werden nun mehrere Signale abgefangen
# Name: trap2.sh
# Signal SIGINT (2) (Strg + C)
trap 'echo SIGINT erhalten' 2
# Signal SIGTERM (15) kill -TERM PID_of_trap2
trap 'echo SIGTERM erhalten.' 15

i=0
while [ $1 -lt 5 ]
do
 echo "Warte immer noch! ($$)"
 sleep 5
 i=`expr $i + 1`
done
```

```
uws@tux>cat trap3.sh
#!/bin/bash
# Funktion trap zum Abfangen von Signalen
# Hier werden nun mehrere Signale abgefangen
# Name: trap3.sh
# Signale SIGINT und SIGTERM abfangen
trap 'echo SIGINT / SIGTERM erhalten.' 2 15

i=0
while [ $1 -lt 5 ]
do
 echo "Das dauert aber! ($$)"
 sleep 5
 i=`expr $i + 1`
done
```

```
uws@tux>cat trap4.sh
#!/bin/bash
# Funktion trap zum Abfangen von Signalen
# Eine Funktion (Signalhandler) einrichten
# Name: trap4.sh
sighandler_INT() {
 printf "Habe das Signal SIGINT empfangen.\n"
 printf "Soll das Script beendet werden? (j/n) : "
 read
 if [[ $REPLY = "j" ]]
 then
  echo "Good Bye!"
  exit 0;
 fi
}

#Signale SIGINT abfangen
trap 'sighandler_INT' 2

i=0
while [ $1 -lt 5 ]
do
 echo "Bitte nicht stören! ($$)"
 sleep 5
 i=`expr $i + 1`
done
```

10.22 Exec

Mit dem Befehl exec kann man Befehle ausführen, ohne eine neue Shell / Process zu starten.

10.22.1 Eingabe-/Ausgabekanal

```
uws@tux>cat DemoExec.sh
#!/bin/bash
#-------------------------------------------------------------------------
#
# Example to create File-Descriptors
#
#-------------------------------------------------------------------------
#
# Help
printf "\nDemo create File-Descriptor.\n"
printf "============================\n"
printf "exec Ziffer>Ziel\t\tAusgabekanal anlegen\n"
printf "exec Ziffer<Quele\t\tEingabekanal anlegen\n"
printf "exec Ziffer<>Datei\tEin/Ausgabekanal anlegen\n"
printf "Befehl >&Nr\t\t\t\tAusgabe auf Kanal Nr.\n"
printf "exec Ziffer>&-\t\t\tAusgabkanal löschen\n"
printf "exec Ziffer<&-\t\t\tEingabekanal löschen\n"
printf "\nJeder angelegter Kanal muss separat gelöscht werden.\n"
printf "==================================================\n\n"

# Create test data
printf "First string\nSecond string\nThird string\n" > data1.txt
printf "one\ntwo\nthree\n" > data2.txt

# Create Ausgabekanal
printf " Anlegen Ausgabekanal 3.\n"
exec 3>data3.txt

printf " Anlegen Eingabekanal 4.\n"
exec 4<data1.txt

printf " Anlegen Ein-/Ausgabekanal 5.\n"
exec 5<>data5.txt

# Setzen Umleitung stdout und stderr
exec >data6.txt 2>data.err
echo "Das kommt in der Datei."

# Ausgabe in Kanal 3
echo "Das kommt in den Kanal 3." >&3
echo " Die einzelne Anweisung zur Umleitung ist nicht nötig."

# Erzeugen einer Fehlermeldung
ls new.txt

# Auslesen Kanal 4 und schreiben auf Kanal 3-
read A <&4
echo $A >&3

# Benutze den Ein-/Ausgabekanal 5.
tail -l <&5
echo "Neue Zeile" >&5
```

```
# Schließen der Kanäle
exec 3>&-
exec 4<&-
exec 5>&-
exec 5<&-

exit 0
```

11. Remote Verbindung

11.1 VNC Server

Wie immer gibt es verschiedene Möglichkeiten, den VNC Server einzurichten. Die Entscheidung, welche Möglichkeit zur Anwendung kommen soll, hängt von den Einsatzbedingungen ab.

11.1.1 Verbindung für alle

Es gibt zwei Möglichkeiten, allen Anwendern den Zugriff auf dem Computer zu erlauben. Dazu rufen wir das Programm YAST auf und unter Punkt Administration `von einem entfernten Rechner` wird der Punkt `Verwaltung via entfernten Rechner (remote) erlauben` aktiviert. Bevor wir den Befehl `rcxdm restart` ausführen, wird in der Datei `/etc/opt/kde3/share/config/kdm/kdmrc` unter dem Punkt [Xdmcp] die Variable `Enable` auf `true` gesetzt. Danach kann der Dienst `rcxdm` neu gestart werden. Der Computer ist nun auf den Ports `5901` (VNC) und `5801` (VNC über Browser) zu erreichen.

In der zweiten Möglichkeit kann man andere Auflösungen für VNC freischalten. Der Port 5x01 liefert die Auflösung 1024x768, 5x02 1280x1024 und 5x03 1600x1200. Hierzu rufen wir wieder das Programm YAST auf und unter `Netzwerkdienste – Netzwerkdienste (xinetd)` werden alle VNC Dienste (`vnc1, vnc2, vnc3, vnchttp1, ..`) aktiviert. Auch hier muss sie Datei `kdmrc` editiert und `rcxdm` neu gestartet werden.

Der VNC Server muss im Runlevel 5 laufen, da es sonst keinen laufenden X-Server gibt.

11.1.2 Verbindung für einen

Anstatt generell die Freigabe des Computers zu machen, kann auch jeder einzelne Benutzer den seinen Desktop freigeben. Hierbei wird der VNC Server von Hand gestartet. Nach der Eingabe des Befehls `vncserver` wird man nach einem Passwort gefragt. Dieses Passwort wird bei dem Verbindungsaufbau abgefragt. Bei dem erfolgreichen starten des Servers erhält man die Meldung, ‚`the new 'X' desktop is tux:5'`. Anschließend ist die Datei `~/.env/xstartup` zu editieren. In der letzten Zeile steht der anzuzeigende Desktop. Standardmäßig ist der `twm` aktiviert. Möchte man einen anderen Desktop, so kann man für KDE den Wert `startkde`, für Gnome `gnome`, für XFCE dann `startxfce4` und bei Openbox `openbox`. Das & Zeichen am Ende darf nicht gelöscht werden.

Den VNC Server kann man mit dem Befehl `vncserver –kill tux:5` beenden.

Möchte man das starten des VNC Servers automatisieren, so ist der Aufruf in der `.profile` Datei vorzunehmen.

11.2 SSH

Eine Remote Verbindung mittels SSH ist einer `Telnet` Verbindung vorzuziehen, da im Gegensatz zu einer `Telnet` Session die Übertragung verschlüsselt durchgeführt wird.

11.2.1 Root Login disable

In der Datei `/etc/ssh/sshd_config` ist der Eintrag `PermitRootLogin` auf `no` zu setzten, um das Root Login zu verbieten.

11.2.2 User Verbindungen

Der Eintrag `AllowUsers` in der Datei `/etc/ssh/sshd_config` mit die zugelassenen User auf, die eine SSH Verbindung aufbauen dürfen. Die Syntax für den Eintrag lautet: `AllowUsers <user1> <user2>`

Ebenso muss die Variable `PasswordAuthentication` auf `yes` gesetzt werden.

11.2.3 SSH Banner

Möchte man bei einer SSH Verbindung ein Banner anzeigen, so muss in der Datei `/etc/ssh/sshd_config` die Zeile `#Banner` auskommentiert und der `Pfad` zur Banner Datei angegeben werden.

```
uws@tux>grep Banner /etc/ssh/sshd_config
Banner /etc/ssh/sshd-banner

uws@tux>cat /etc/ssh/sshd-banner
***********************************************
 WARNING:
 ========
 Only authorized User to used this system.
 Are you not authorized User, leave this session.
***********************************************
```

11.2.4 SSH Dienst

Bei jeder Änderung an der SSH Konfiguration muss der Dienst neu gestartet werden.

```
root@tux>/etc/init.d/sshd restart

root@tux>service sshd restart
```

11.2.5 X11 Forwarding

Möchte man die Graphische Ausgabe auf seinem Rechner haben, so schaltet man auf dem Zielrechner das X11Forwarding ein. In der Datei `/etc/ssh/sshd_config` werden die Parameter `X11Forwarding yes`, `X11DisplayOffset 10` und `X11UseLocalhost yes` aktiviert.

```
uws@tux>ssh –X –C uws@tux01

uws@tux01>export DISPLAY=‘tux:10.0‘

uws@tux01>xclock &
```

11.2.6 SSH Key entfernen

Hat sich der SSH Key eines Remote Zieles verändert, so kann man den SSH Key aus der Datei `known_host` mit dem folgenden Befehlt entfernen.

```
uws@tux>ssh-keygen –R tux10 –f /home/uws/.ssh/known_hosts
```

11.2.7 TCP-Stealth

Wird in der Datei `/etc/ssh/sshd_config` die Option `TCPStealthSecret` auf dem Server / Client aktiviert, so klappt es mit der Verbindung. Ansonsten wird sie abgelehnt. Der eingetragene Wert ist ein Autorisierung-Token. Nur wenn beide gleich sind, wird eine Verbindung ausgebaut.
Diesen Modus kann man nur einschalten, wenn auch der Kernel mit TCPStealthSecret kompiliert worden ist.

11.2.8 Error

Taucht in der `/var/log/messages` folgender Fehler auf:
`sshd syslogin_perform_logout:logout() returned an error`, so ist die Datei `utmp` im Verzeichnis `/var/run` neu anzulegen.

```
root@tux>rm /var/run/utmp

root@tux>touch /var/run/utmp
```

11.3 Displaymanager

11.3.1 Konfiguration

Die Variable `DISPLAYMANAGER_REMOTE_ACCESS` wird auf yes gesetzt, die sich in der Datei `/etc/sysconfig/displaymanager` befindet. Danach in der Datei `/etc/X11/xdm/Xaccess` das Kommentarzeichen aus der Zeile `* #any host can get a login` entfernen.

11.3.2 Dienst starten

Nacheiner Änderung in der Datei `displaymanager` müssen folgende Schritte gemacht werden.

```
uws@tux>SuSEconfig

uws@tux>rcxdm restart
```

Bei Änderungen in der Datei `Xaccess` muss der Dienst neu gestartet werden.

```
uws@tux>kdm Stopp

uwsqtux>kdm start
```

11.3.3 Root Login GDM

Standardmäßig kann man sich nicht mit dem User `root` an einem Oracle Linux Server anmelden (GUI). Damit man sich anmelden kann, muss in der Datei `/etc/pam.d/gdm` die folgende Zeile auskommentiert werden:

```
auth required pam_succeed_if.so user != root quit
```

11.3.4 Anmelde Bildschirm

Damit man bei einer Remote Anmeldung (GUI) die Login Oberfläche angezeigt bekommt, fügt man in der Datei `/etc/gdm/custom.conf` im Abschnitt `[xdmcp]` den Eintrag `Enable=true` ein

11.4 Dateien kopieren

Um Dateien zwischen von einem Linux Rechner auf einem anderen Rechner zu kopieren, so gibt es hierzu den Befehl `scp`. In den nachfolgenden Beispielen werden zuerst Dateien von einem Rechner zu einem anderen kopiert. Im zweiten Beispiel wird eine Verbindung zu dem Zielrechner mit einem anderen Benutzername hergestellt. Und im letzten Beispiel werden Dateien von einem entfernten Rechner kopiert.

```
uws@tux>scp /home/uws/transfer/*.jpg tux01:/home/uws/transfer

uws@tux>scp /home/uws/transfer*.jpg jan@tux01:/home/jan/transfer

uws@tux>scp jan@tux01:/home/jan/bin/*.sh uws@tux:/home/uws/bin
```

12. Benutzer / Gruppen

12.1 Benutzer

12.1.1 Anlegen

Neue Benutzer werden mit dem Befehl `useradd` angelegt. Eine Auswahl der Optionen folgt in der nachfolgenden Liste.

Option	Beschreibung
-c	Einen Kommentar mit angeben.
-d	Angabe des Home Verzeichnisses.
-e	Datum, wann das Konto abläuft im Format YYYY-MM-DD.
-f	Nach wie vielen Tagen des inaktiven Kontos, nach Ablauf des Kennwortes, das Konto gesperrt werden soll.
-G	Angabe der Gruppen.
-g	Angabe der Hauptgruppe.
-m	Erstellen des Home Verzeichnisses.
-p	Passwort.
-r	Systemaccount.
-s	Angabe der Shell.

Die default Werte stehen in der Datei `useradd`, die sich im Verzeichnis `/etc/defaults` befindet.

```
uws@tux>useradd <BenutzerName>
```

12.1.2 Ändern

Einen angelegten Benutzer kann man mit dem Befehl `usermod` bearbeiten. Hierbei gelten die gleichen Optionen wie bei dem Anlegen eines neuen Benutzers.

```
uws@tux>usermod –c “Home User“  <BenutzerName>
```

Gruppen können einem User mit `–G` zugewiesen werden. In der Kombination mit `–a` werden die angegebenen Gruppen dem User addiert.

```
uws@tux>usermod –aG users,kvm,libvirt <username>
```

12.1.3 Löschen

Benutzer werden mit dem Befehl `userdel` gelöscht. Gibt man die Option `–r` mit an, so wird das Home Verzeichnis mit gelöscht.

```
uws@tux>userdel –r <BenutzerName>
```

12.1.4 Anzeigen

Angelegte Benutzer kann man mit dem nachfolgenden Befehl sich anzeigen lassen. Mit $4 wird die GroupID ausgegeben.

```
uws@tux>awk –F ’:’ ’{print $1}’ /etc/passwd

uws@tux>awk –F ’:’ ’{print $1 “-“ $4}’ /etc/passwd
```

Für die UID und Gruppen des angemeldeten Users gibt es auch die Variablen `$UID` und `$GROUPS`.

```
uws@tux>echo $UID

uws@tux>echo $GROUPS
```

12.1.5 Angemeldete User

Mit den nachfolgenden Befehlen kann man sich die angemeldeten User anzeigen lassen.

Befehl	**Beschreibung**
W	Zeigt an, wer ist angemeldet und was sie tun.
Who	Zeigt an, wer ist eingelogt.
Last	Welche Benutzer waren angemeldet.

```
uws@tux>grep sshd /var/log/messages
```

12.1.6 Kennwort ändern

Das Kennwort kann man mit dem Befehl `passwd` ändern.

```
uws@tux>passwd <BenutzerName>
Changing password for <Benutzer>
New Password:
Reenter password:
```

12.1.7 Benutzer Info‘s

Informationen über die Benutzer kann man mit dem Programm `finger` sich anzeigen lassen.

```
uws@tux>finger uws

Login: uws              Name: Uwe Schimanski
Directory: /home/uws    Shell: /bin/bash
On since Fri Oct 14 12:26 (CEST) on pts/0
No Mail
No Plan
```

12.2 Gruppen

12.2.1 Anlegen

Neue Gruppen werden mit dem befehl `groupadd` angelegt.

```
uws@tux>groupadd <GruppenName>
```

12.2.2 Ändern

Eine bestehende Gruppe kann man mit dem Befehl `groupmod` ändern. In der nachfolgenden Tabelle werden einpaar Optionen aufgelistet.

Option	Beschreibung
-g	Ändern der Group ID.
-A	Hinzufügen einen Benutzers für die Gruppe.
-R	Llöschen eines Benutzers aus der Gruppe.

```
uws@tux>groupmod –A <BenutzerName> <Gruppen>
```

12.2.3 Löschen

Eine Gruppe wird mit dem Befehl `groupdel` gelöscht.

```
uws@tux>groupdel <Gruppe>
```

12.2.2 Anzeigen

```
uws@tux>awk –F ’:’ ’{print $1}’ /etc/group
```

12.3 Logon Zeit begrenzen

Möchte man die Anmeldezeit für einen User festlegen, so kann man das mittels der `times.conf` im Verzeichnis `/etc/security` bewerkstelligen.

12.3.1 Voraussetzung

Das Paket `libpam-modules` muss installiert sein.

12.3.2 Konfiguration

Die Syntax in der times.conf ist:

```
<services>;<ttys>;<user>;<time>  # Kommentar
```

Beispiel:

```
Login;*;paul;MoFr1200-1700
```

Das Format für die Zeitangabe ist `HHMM-HHMM`.

Format für die Tage:

Mo	Montag
Tu	Dienstag
We	Mittwoch
Th	Donnerstag
Fr	Freitag
Sa	Samstag
So	Sonntag
Wk	Wochentags
Wd	Wochenende
Al	Alle Tage
MoWk	Alle Wochentage, außer Montag
AlFr	Alle Tage, außer Freitag

Folgende Operatoren gibt es:

!	Logisches nicht
&	Logisches und
|	Logisches oder
*	Platzhalter für alles

Ein weiteres Beispiel:

```
Login;tty*&!pts*;du|ich;!al1000-2400
```

Die User `du` und `ich` dürfen sich an allen Tagen in der Zeit zwischen 10:00-24:00 nicht an den virtuellen Consolen anmelden. An Terminals (pty*) gilt die Sperre nicht.

12.3.3 Aktivierung

Damit die times.conf auch aktiv wird, so muss das Zeitmodul pam_time.so in der Datei common_auth im Verzeichnis /etc/pam.d geladen werden.

```
root@tux>grep pam_time common_auth
account  required  pam_time.so
```

13. Drucken

13.1 CUPS (Common Unix Print System)

13.1.1 Konfiguration

Die Konfiguration von CUPS kann mit dem Programm `cupsctl` vorgenommen werden. Die Konfigurations Datei befindet sich im Verzeichnis `/etc/cups` und trägt den Namen cupsd.conf. Eine Hilfe bekommt man mit `man cupsd.conf` angezeigt.

In der nachfolgenden Tabelle sind einige der Konfigurartions Schalter aufgelistet.

Schalter	Beschreibung
--[no-]debug-logging	Das Logging ein- oder ausschalten.
--[no-]remote-admin	Remote Administration ein- oder ausschalten.
--[no-]remote-any	Internet Zugriff erlauben oder ausschalten.
--[no-]remote-printers	Remote Drucker anzeigen
--[no-]share-printers	Drucker freigeben.
--[no-]user-cancel-any	Benutzer dürfen Jobs löschen.
-E	Veschlüsselung einschalten
-u <username>	Benutzername definieren.
-h <server[:port]>	Server Adresse.

Nach Änderung an der Konfiguration muss Cups neu gestartet werden.

13.1.2 Verwaltung Browser

Die Verwaltung der Drucker unter CUPS kann man mit dem Browser durchführen. Die Verwaltungsseite wird mit der URL `http://localhost:631` aufgerufen. Diese Seite läst sich natürlich auch von einem anderen Rechner aus aufrufen.

13.1.3 Dienst starten

Der Dienst wird mit `cups restart` neu gestartet. Außer `restart` gibt es auch noch `stop` und `start`.

```
uws@tux>cups <restart> <stop> <start>
```

13.1.4 Admin

Standardmäßig darf nur der erste angelegte Benutzer Cups Administrieren. Möchte man noch andere Benutzer die Administration erlauben, so sind diese Benutzer zu der Gruppe `lpadmin` hinzuzufügen.

```
uws@tux>adduser <username> lpadmin
```

13.1.5 Druckdaten entfernen

Abgearbeitet Druckdaten kann man mit dem nachfolgenden Befehl entfernen.

```
uws@tux>cupsctl AutoPurgeJobs=yes PreserveJobFiles=no PreserveJobHistory=no
```

13.1.6 Servernamen Drucker

Soll bei jedem Drucker auch der Servernamen mit angezeigt werden, so gibt man folgendes ein.

```
uws@tux>cupsctl BrowseShortNames=no
```

14. Programme

14.1 Kalender in der Shell

In der Bash kann man sich einen Kalender anzeigen lassen mit dem Befehl `cal`. Cal reicht zurück bis zum 1. Januar 0001.

```
uws@tux>cal
     June 2012
Su Mo Tu We Th Fr Sa
                1  2
 3  4  5  6  7  8  9
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30
```

Gibt man dem Befehl `cal` noch den Parameter `–y` auf dem Weg, so wird das aktuelle Jahr ausgegeben. Mit `cal 06 2005` wird der Juni 2005 ausgegeben. Die Angabe `cal -3` gibt den aktuellen Monat mit dem vorherigen und nachfolgenden Monat aus. Bei dem Parameter `–j` werden die Tage von 1 bis 365 gezählt und ausgegeben.

14.2 Virtual Box

Mit der Variable `VBOX_USER_HOME` kann man die Ablage der Konfiguration an einer zentralen Stelle ablegen, wo alle Benutzer Zugriff darauf haben. Normalerweise werden die erstellten Maschinen nur für den jeweiligen Benutzer angezeigt, der die Maschinen erstellt hat, da die Konfiguration standardmäßig im Home Verzeichnis abgelegt wird.

```
uws@tux>echo $VBOX_USER_HOME
/vm/configuration
```

Eine Liste aller erstellten Maschinen kann man sich mit VBoxManage sich anzeigen lassen.

```
uws@tux>vboxmanage list vms
```

15. Virtualisierung

15.1 KVM (Kernel-based Virtual Machine)

15.1.1 Voraussetzungen

Die CPU muss die Virtualisierung unterstützen. Hierzu kann man die Eigenschaften dazu abfragen.

```
root@tux>egrep ‘(vmx|svm)‘ /proc/cpuinfo
```

15.1.2 Installation KVM

Außer dem Paket KVM werden noch andere nützliche Programme für das erstellen und verwalten der Virtuellen Maschinen benötigt. Für RedHat Systeme lautet der Befehl für die Installation yum install. Nach der Installation sollte ein Neustart des Systems erfolgen.

```
root@tux>zypper install kvm

root@tux>zypper install qemu-kvm python-virtinst libvirt libvirt-python
virt-mamanger libguestfs-tools
```

15.1.3 Network

Die erstellten Virtuellen Maschinen haben Standard mäßig nur Netzzugriff auf die anderen Virtuellen Maschinen und zur Host Maschine. Sollen die Virtuellen Maschinen auch Zugriff auf das VLAN haben, so muss eine Netzwerk Brücke eingerichtet werden.

Hierzu wird in der Konfigurationsdatei, z.B. für eth0, die Zeile „BRIDGE=br0“ eingefügt und wenn dort eine IP-Adresse eingetragen ist, wird diese auskommentiert.

Nun wird eine Konfigurationsdatei ifcfg-br0 für die Netzwerk Brücke eingerichtet.

```
root@tux>cat ifcfg-br0
DEVICE=”br0”
BOOTPROTO=”static”
IPADDR=”xxx.xxx.xxx.xxx”
NETMASK=”255.255.25.0”
ONBOOT=”yes”
TYPE=”Briodge”
NM_CONTROLLED=”no”
```

In der Datei /etc/sysctl.conf wird das IP forwarding aktiviert. Neustart der Maschine.

```
root@tux>grep ”ipv3” systemctl.conf
inet.ipv3.ip_forward=1
```

15.1.4 Maschine Location

Standard mäßig werden die Maschinen im Verzeichnis /var/lib/libvirt/images erstellt / abgelegt. Es sollte also ausreichend Platz vorhanden sein.

15.1.5 Create Maschine

Für OpenSuSE lautet der Befehl für das Erstellen einer Maschine `vm-install` und für RedHat Systeme `virt-install`. Nachfolgend ein Beispiel, wie unter RedHat eine Maschine erstellt wird.

```
root@tux>virt-install \
-n myLinux \
--description “My own Linux Machine“ \
--os-type=Linux \
--os-variante=debian7 \
--ram=4096 \
--vcpus=2 \
--disk path=/vmstore/machine/myLinux/myLinux.img,bus=virtio,size=20 \
--graphics none \
--cdrom /vmstore/images/debian7_x86_64.iso \
--network bridge:br0
```

Option	Beschreibung
-n	Name der Maschine
--description	Beschreibung der Maschine
--os-type	Linux, Solaris, Unix oder Windows
--os-variante	Distribution type. Z.B. rhel7, centos6, debian7, suse13, win2k, win2k8, win7, win8
--ram	Memory in MB
--vcpus	Anzahl Virtuelle CPU
--disk	Lage der Virtuellen Maschine und Größe der Festplatte in GB
--graphics	None für die Benutzung einer Text console auf dem VM serial Port, anstelle der Graphischen VNC Windows. Ist der xmanager aktiv, so kann der Parameter ignoriert werden.
--cdrom	Installations Medium
--network	Angabe des Netzwerk Devices

15.1.6 Konfiguration

Die Konfiguration der virtuellen Maschinen wird in dem Verzeichnis `/etc/libvirt/qemu` abgelegt. Die Konfigurations Datei kann mit dem Befehl `virsh edit myLinux` bearbeitet werden.

```
root@tux>virsh edit myLinux
```

15.1.7 Maschinen anzeigen

Mit dem Befehl `virsh list --all` kann man sich die Virtuellen Maschinen anzeigen lassen.

```
root@tux>virsh list –all
ID  Name          State
----------------------------
1   myLinux       running
```

Weitere Informationen über die virtuelle Maschine können mit `virsh dominfo` abgefragt werden.

```
root@tux>virsh dominfo myLinux
Id:             1
Name:           myLinux
UUID:           58083ae7-37dr-46c2-63d9-bd5dc49f642
OS Type:        Linux
State:          running
CPU(s):         2
CPU time:       305.4s
Max memory:     2097152 KiB
Used memory:    1032452 KiB
Persitent:      yes
Autostart:      disable
Manged save:    no
Security model: selinux
Security DOI:   0
Security Label: system_u:system_r:virt_t:s0:c968,c799 (permissive)
```

Die CPU und Memory Belegung kann man sich mit `virt-top` anzeigen lassen.

15.1.8 VM Console

Anmelden an der virtuellen Maschine kann mit dem Kommando `virsh console` gemacht werden. Die Console kann dann mit „`ctrl +`“ wieder verlassen.

```
root@tux>virsh console myLinux
```

15.1.9 Shutdown, Reboot und Start

```
root@tux>virsh shutdown myLinux

root@tux>virsh reboot myLinux

root@tux>virsh start myLinux
```

15.1.10 Convert Disk

Eine Disk kann man in einem anderen Format konvertieren. Die Unterstützten Formate kann man sich mit dem folgenden Befehl sich anzeigen lassen,

```
root@tux>qemu-img –h | tail –n1
```

OVA-Dateien sind nichts anderes als Tar-Archive.

```
root@tux>tar –xvf <ova_file>
```

Mit `qemu-img convert` kann man das Ausgangsformat konvertieren.

```
root@tux>qemu-img convert –o qcowl <vmdk.file> output sdb.qcow2
```

15.1.11 Resize Disk

Eine Disk wird mit qemu-img resize vergrößert. Im nachfolgenden Beispiel wird die Disk um 10GB vergrößert.

```
root@tux>qemu-img resize <disk_image> +10G
```

15.2 Troubleshooting

15.2.1 Failed to start network default

Bei dem starten des Netzwerkes für die VM's kommt es zu einer Fehlermeldung.

```
root@tux>virsh net-start default
error: Failed to start network default
error: internal error: Failed to initialize a valid firewall backend
```

Unter Arch Linux (Manjaro) überprüfen wir die Firewall.

```
root@tux>systemctl status firewalld
firewalld.service – firewalld – dynamic firewall daemon
  loaded: Loaded
(/usr/lib/system/firewalld.service;enabled;Vendorpreset:disabled)
  Active: active (running) since Thu 2017-03-09 19:15:39 CET;40s ago
    Docs: man:firewalld(1)
Main PID: 590 (firewalld)
   Tasks: 2 (limit 4915)
  CGroup: /system.slice/firewalld.service
          590 /usr/bin/python –ES /usr/bin/firewalld –nofork –nopid

root@tux>pacman –Syu ebtables dnsmasq

root@tux>systemctl restart libvirtd
```

Sollte keine Firewall installiert sein, so installiert man die Firewall nach.

```
root@tux>pacman –Syu firewalld
```

16. Verweise / Links

Bash Prompt How to	http://www.tldp.org/HOWTO/Bash-Prompt-HOWTO
SSH-Artikel	Sicher gesendet – ssh und scp – Linux User 07/2003, Seite 81 ff.
Mehr zu date und cal	Kalender auf der Kommandozeile – Linux User 07/2005, Seite 88 ff.
Rsync	http://samba.anu.edu.au/rsync
Rsnapshots	http://www.rsnapshot.org
Rsync	Synchroner Datenstrom – Linux User 04/2006, Seite 90 ff.
Rsync	Snapshot-Backup mit Rsync – Linux User 09/2004, Seite 72
Uhrzeit	Wer hat an der Uhr gedreht – Linux User 10/2005, Seite 61 ff.
Xdialog	http://xdialog.dyns.net
Prompt Color	https://wiki.archlinux.org/index.php/Color_Bash_Prompt#Return_value_visualisation
Sed	Workshop: Einführung in den Stream Editor Sed – Linux User 08/2015, Seite 68
Pacman	https://wiki.archlinux.de/title/pacman

17. Copyright