INDEX

http://www.niksoft.at/

Fehler, Unvollständigkeit und Änderungen vorbehalten.

Letzte Änderung: 2004-03-10, 2004-08-07

Anm.: Einige Dinge sind RedHat spezifisch! - Achja, und niemals vergessen: GIYF und RTFM!

Befehlsreferenz

Hilfe

man [chapter] [page]

Hilfe zu allmöglichen Themen
-k wie "apropos"

info [page]

Hilfe; falls keine info-page gefunden wird einfach die passende man-page aufgerufen

--help

Kurze Hilfe; gilte bei fast jedem Befehl; "-h", "-?" geht auch manchmal

Verz-Operationen

cd [verz]

Aktuelles Verz wechseln zb:
cd (home-verz)
cd ~ (home-verz)
cd /etc (ins "/etc" Verz)
cd ../blabla (ins übergeordnete Verz und dann ins blabla-Verz)

ls [verz/datei]

list files
-l (long) pro Datei eine Zeile
-d (directory) zeigt das verz. selbst an und nicht den Inhalt
-a (all) zeigt auch alle versteckten Dateien an (.).
-A (allmost all) alle ausser "." und ".."
-R (recursiv)
-F (format) markiert Verz mit /, Binaries mit *, Symlinks @, ...
-t (time) sortiert nach Zeit
-i (inode) zeigt inode-Nummer an
-s (size) zeigt Dateigröße in Blöcken an

mkdir [verz]

Verz erstellen
-P ("make parent directories as needed")

pwd

(print woking directory) gibt aktuelles Verz aus.

rmdir [verz]

Verz löschen
-r (recursiv)

Datei-Operationen

cat [datei]

Gibt Datei aus (Beachte: "Useless use of cat" ;); zum Dateibetrachten ist "less" oder "more" besser.
-n (number) gibt die Datei mit Zeilenummern aus

cp [quelle] [ziel]

Kopiert Dateien
-r, -R (recursiv) "-R" kopiert auch device-Dateien!
-a (archiv = "-dpR", kopiert wirklich ALLES und wirklich 1:1)

-d (no dereference) niemals symlinks folgen
-p (preserve) alle Dateiattribute erhalten (Zeit, Owner, ...)
-u (update) kpiert nur neue oder fehlende Dateien ins Ziel

chmod [option] [datei/en]

(change mode) ändert die Dateizugriffsrechte
-R (recursiv)
ugo+-=rwx user/group/others recht erteilen/entziehen/setzen read/write/execute
r = 4, w = 2, x = 1 zb.: 760 = rwx, rw-, ---
s-Bit/setuid-Bit/t-Bit (auch 4,2,1), ausführen mit den Rechten des Users/Group ("x" erforderlich!) oder bei t-Bit(sticky-Bit) darf jeder in dem Verz seine eigenen Dateien erstellen und löschen
zb.: "chmod 4550 datei" setzt r-S, r-x, ---

chown [option] [new_user[:group]] [datei/en]

(change owner) ändert den Besitzer einer Datei; siehe auch "chgrp"
-R (recursiv)

file [datei]

Gibt Auskunft über Dateityp/Dateiformat

head [datei]

(Kopf) gibt die ersten Zeilen einer Datei aus
+n ab der n. Zeile
-n letzte n Zeilen ausgeben

less [option] [datei]

Datei-Betrachter wie "more", aber sehr viel komfortabler
+F wie "tail -f"
+G öffnet die Datei am gleich am Ende (nach unten scrollen erübrigt sich dann ;)

im Program selbst:
/ serach (such-muster dann eingeben, "böse" zeichen ggfalls mit "\" maskieren)
n/N suche abwärts/aufwärts weiter
h help
q quit

ln [option] [link-target] [link]

Erstellt Links; Hardlinks: nur innerhalb einer Partition möglich, Files haben identische Inode-Nr.), Softlinks: Partitionsübergreifend, mit ls als "file -> file2" zu sehen; zB. "ln -s ../file1 file2", wenn auf "file2" zugreift, dann eigentlich auf "../file1"
(leer) erstellt einen Hardlink
-s - Softlink/SymLinks - diese sind idR Hardlinks vorzuziehen

mv [quelle] [ziel]

Dateien/Verz verschieben oder umbenennen
-i, -f (interactive, force)

od [option] [datei]

"dump files in octal and other formats"
-c (ascii)

rm [datei]

Datei löschen
-r (recursiv)
-f, -i (force) ohne Sicherheitsabfrage, (interactive) mit Sicherheitsabfrage

stat [datei]

Gibt Auskunft über die Metainformationen, Inode, etc...

tail [datei]

(Schwanz) gibt die letzten Zeilen einer Datei aus
+n ab der n. Zeile
-n letzte n Zeilen ausgeben
-r (reverse) Zeilen in umgekehrter Reihenfolge ausgeben
-f (feed) am Dateiende nicht beenden sondern auf weitere Zeilen warten (zb. Logfiles); Abbruch mit CTRL+C
zB. "head -9 COPYING | tail -1" gibt nur die 9. Zeile aus (" Preamble " in dem Fall)

touch [datei]

ändert den Zeitstempel einer Datei. Falls Datei nicht existiert, wird sie erstellt
"modification time" letzter schreibender Zugriff
"access time" letzter lesend Zugriff
"change time" letzte Änderung an den Metainformationen (owner, group, modus, ...)

umask [oktal-wert]

"Zugriffsfilter" wird von den meisten Apps unterstützt, zB. Wert bei root 0022, usern 0002; Änderungen bleiben auf aktuelle Sitzung beschränkt (daher in "~/.bashrc" oder "/etc/bashrc" speichern). zB. "vi" erstellt eine Datei mit 666, vi wendet umask an: 655.

```
110110110 rw-rw-rw- 666
000010010 ----w--w- 022 umask subtrahieren (bzw. xor)
111101101 rwxr-xr-x 655
```

vi [datei]

Editor, siehe "vimtutor" oder "vim"
EINFG/i
ESC, ":wq" (schreiben und beenden), ":q!" (beenden ohne Änderungen zu speichern)

whereis [datei]

Wo befindet sich diese Datei (bzw. Befehl)? zB. "whereis ls" - sucht nur in "PATH"

Bash/Shell, Script-Tools

alias

Makros; Tippersparnis gilt nur in der jeweiligen Sitzung (daher in die "~/.bashrc" bzw. "/etc/bashrc" speichern) zb.:
hw=‘echo Hallo Welt’ Wenn man dann auf der Shell "hw" eingibt,

cut [option] [datei]

schneidet gewisse dinge aus
-d : (deliminator) legt ":" als Spalten-Trennzeichen fest
-f 1,4 (field) gibt Spalte 1 und 4 aus

bash

Die Shell, siehe "bash regex", "Shell-Programmierung"-Skriptum und Anwendugsbeispiele!

CTRL+r	Befehle durchsuchen
SHIFT+Bild↓	erster/letzter Befehl
!123	Befehl Nr. 123 ausführen
!!	Letzten Befehl
!cd	Letztes "cd" rückgängig machen
TAB	Auto-Vervollständigung, expandieren
ESC, f	springt zum nächsten Wortende
ESC, b	springt zum nächsten Wortanfang
CTRL+e	springt zum Zeileende
CTRL+a	springt zum Zeilenanfang
CTRL+s/q	scroll-lock on/off
CTRL+d	logout, EOF
CTRL+c	cancel
CTRL+z	aktuellen Prozess stoppen (mit "fg" wieder starten)

cat file1 > file2 (STDOUT nach file2 umleiten)
cat file1 >> file2 (file1 and file2 anhängen)
cat file < file2 (STDIN umleiten)
cat file | sort (STDOUT pipen)
cmd1 ; cmd2 (führt cmd1 und -2 aus)
cmd1 && cmd2 (log. AND; wenn cmd1 erfolgreich (exit 0),dann wird cmd2 ausgeführt)
cmd1 || cmd2 (log. OR; nur wenn cmd1 nicht erfolgreich (exit ≠ 0), dann cmd2 ausgeführt)

cmd1 > /dev/null (STDOUT ins Schwarze Loch umleiten)
cmd1 2>/dev/null (STDERR -)
cmd1 > output 2> /dev/null (STDOUT nach "output" und STDERR nach /dev/null umleiten)
cmd1 > output 2>&1 (STDERR nach STDOUT und STDOUT nach "output" umleiten)
&0 = STDIN &1 = STDOUT &2 = STDERR

‘ bla’ einfache Anführungszeichen, keine Interpretation durch die Shell
"bla" doppelte Anführungszeichen:$-Variablen werden interpretiert
`bla` steht für die Ausgabe des Befehles "bla" (BackTick)
$(bla) dito
$((2+3) gibt 5 zurück

export VAR Inhalt von VAR bleibt auch nach Ende des Scripts erhalten
expr 1 + 2 Rechner

bash regex

regular expressions der shell
* kein Zeichen oder beliebige beliebig viele Zeichen
? genau ein beliebiges Zeichen
[c-f] Zeichen c, d, e, f
[!c-f] Alle Zeichen ausser c, d, e, f
[c-f!] Zeichen c, d, e, f und !
[a-zA-Z] Zeichen a bis z und A bis Z
{a,x,z} Zeichen a, x oder z
{abc{1,2},def{3,4}} steht für abc1, abc2, def3 und def4
\? Fragezeichen; mit vorangestelltem Backslash lässt sich jedes Zeichen maskieren

egrep/grep "muster" [datei]

Durchsucht Datei(en) nach einem Muster
-v (inverse) zeigt alle Zeilen die das Muster NICHT enthalten
-c (count lines)
-C 1 (context) gilt je 1 Zeile vor und nach der Fundstelle aus, Fundstellen getrennt durch "--\n"
-i (ignor case)
-l nur dateinamen anzeigen, wenn muster gefunden wurde
-L nur dateinamen anzeigen, wenn muster nicht gefunden wurde

. genau ein beliebiges Zeichen
^ $ Zeilenanfang/ende, zB. "^$" wäre eine Leerzeile
e? ein oder kein mal "e"
e+ "e" kommt ein mal bis beliebig oft vor
e* "e" kommt kein mal bis beliebig oft vor
< > Wortanfang/ende
e{2} "e" muss 2x vorkommen
e{2,5} "e" muss 2 bis 5x vorkommen
e{2,} "e" muss mindestens 2x vorkommen
(.)(.)\2\1 Geklammerte-Ausdrücke können als "\1" "\2" usw. wiederverwendet werden
ABC(XYZ)* passt auf ABC, ABCXYZ, ABCXYZXYZ, ...
(e|f) "e" oder "f"
[abc]; [a-zA-Z] die Zeichen a, b oder c; Zeichen a bis z und A bis Z
[^0-9] nicht die Zeichen 0 bis 9
[ab^] die Zeichen a,b und ^

\a	Bell/Piep	\f	Seitenvorschub (formfeed)	\w	Wort [a-zA-Z_0-9]
\n	Zeilenvorschub (lf)	\e	Esc	\W	Nicht-Wort
\r	Wagenrücklauf (cr)	\d	Ziffer	\s	Whitespace [\t\n\r\f]
\t	Tabulator	\D	Nicht-Ziffer	\S	Nicht-Whitespace
\.	Punkt	*	Stern	\?	Fragezeichen

[:alnum:]	Alphanum. Zeichen	[:digit:]	Ziffern	[:punct:]	Interpunktionsz.
[:alpha:]	Buchstaben	[:graph:]	"schwarze" Zeichen	[:space:]	"weiße" Z. [\t\n]
[:blank:]	Leerzeichen und Tab	[:lower:]	Kleinbuchstaben	[:upper:]	Großbuchstaben
[:cntrl:]	Steuerzeichen (1-31)	[:print:]	wie [:graph:] mit Leerz.	[:xdigit:]	Hexadezimale Ziffer

grep	egrep	awk	emacs	perl	Tcl	vi	Bemerkung
\?	?	?	?	?	?	\?	0 bis 1 Atom
\+	+	+	+	+	+	\+	1 bis n Atome
|	|	|	|	|	|		Alternativen
\(...\)	(...)	(...)	\(...\)	(...)	(...)	\(...\)	Gruppenbildung
	\< \>		\< \> \b \B	\b \B		\< \>	Wortgrenzen
	\w \W		\w \W	\w \W		\< \>	Wortzeichen
ja			ja	ja		ja	Rückwärtsreferenzen

exit [option]

Beendet ein Script
1 setzt den Exitcode auf 1 (abfragbar mit "$?", siehe "bash")
2 dito, nur halt 2, usw...

find [pfad] "muster"

Sucht nach Dateien/Verz
-name "expr" case-sensiv
-iname "expr" ignor case/case insensiv
-path "expr" erlaubt auch den pfad anzugeben im namen
-type f,l,d,s,b,c,p file, smylink, dir, socket, block-dev, char-dev, named pipe
-perm ±755, ±4000 (s-bit) (zB. "+4000" sucht nach Datein das s-bit gesetzt haben)
-user [user], -uid, -group, -gid
-nouser, -nogroup (praktisch um nach Überbleibsel gelöschter User zu suchen)
-{a|m|c}{time|min} zb. -atime 2 (alle datein die in den letzten 2 tagen gelesen wurden, - jünger, + älter als 2 Tage - siehe "touch")
-size ±123{|b|k|m} zb. -size +5k (größer als 5kb)
-prune wie eine inverse; Dateibaum vom gefundenem Objekt wird nicht weiter durchsucht
\(\) Klammerung
-o log. OR
-a log. AND
-exec zb. "-exec cp '{}' /tmp \;" kopiert jede gefundene Datei nach /tmp
-ok genau wie -exec, nur muss die Ausführung vorher mit y oder j bestätigt werden

history

gibt die Befehels-History aus (laaange Liste!)

sort

Sortiert die Zielen einer Datei alphabetisch (Vorsicht bei Unicode!)
-r (reverse)
-n (numeric) sortiert nach Zahlen zB. "9" kommt vor "19"
-k 3,1 (key) sortiere erst nach Spalte 3 und dann erst nach 1 (beginnt bei 1 zu zählen!)
-t : gibt ":" als Spalten-Trennzeichen an
+1 -3 beginnt die Sortierung erst bei Spalte 2 (beginnt bei 0 zu zählen!)

tac [datei]

(cat reverse) Zeilen in umgekehrter Reihenfolge ausgeben. Gibt erste Zeile zuletzt aus.

tee [datei]

Pipe mit T-Stück; Kopiert alles von STDIN nach STDOUT und in die angegebene Datei

test [option]

Führt diverse Vergleichsoperationen aus, auch als "[ ]" und "[[ ]]" zu verwenden

Allgemein:		!A	not A
A -a B	A and B	A -o B	A or B
Strings/Zeichenfolgen:			
(-n) A	length(A) != 0	-z A	length(A) = 0
A == B	A = B	A != B	A !=B
Zahlen:			
A -eq B	A = B	A -ne B	A != B
A -ge B	A >= B	A -gt B	A > B
A -le B	A <=B	A -lt B	A < B
Dateien:		-s A	A exists and has a size > 0
A -nt B	A is newer than B	A -ot B	A is older than B
-b A	A exists and is a block special	-c A	A exists and is a charackter special
-d A	A exists and is a DIRECOTRY	-f A	A exists and is a normal FILE
-L A	A exists and is a SymLink	-r A	A exists and is readable
-w A	A exists and is writeable	-x A	A exists and is executeable

time [cmd]

stoppt die Zeit die der Befehl "cmd" zur Ausführung benötigt (kleiner Benchmark) zB. um zu stoppen wie lange das Backup braucht oä. - Parameter für "cmd" haben bei mir Fehler verursacht, trotz anderslautender Manual.

tr [option] '[set1]' '[set2]'

transform; konvertiert zeichen/muster in ein anderes; siehe "egrep" und Klammerausdrücke ("[:digit:] ", ...)

-s (squeez repeats) mehrere gleiche aufeinanderfolgende Zeichen werden auf 1 reduziert
-d (delete) keine Ersetzung, löscht 'set1'

zB. "ls -l | tr -s ' ' '\t' | cut -f 9" gibt nur die Dateinamen aus ;-)

unalias

Makros wieder löschen zb.:
unalias hw

uniq

eliminiert doppelte Zeilen (sprich: einzigartig machen ;)

wc [option]

(word count) Gibt die Anzahl der Wörter aus

(ohne option) gibt Zahlen im Format " Zeilen Worte Zeichen " aus
-l zähle Zeilen
-w zähle Worte
-c zähle Zeichen

Archive ("Teern & Federn")

bzip2

Komprimiert (besser als gzip) einzelne Dateien (daher mit tar arbeiten um mehrere zu packen)

-d decompress/entpacken
-1 bis -9packen (1=fastest, 9=best)

gunzip [option] [archiv]

-d decompress/entpacken bzw. "gunzip"
-c nach STDOUT entpacken

gzip [option] [archiv]

Komprimiert einzelne Dateien (daher mit tar arbeiten um mehrere zu packen); empfehle ‘GZIP="-9"; export GZIP’

-d decompress/entpacken bzw. "gunzip"
-c nach STDOUT entpacken
-1 bis -9packen (1=fastest, 9=best)

rpm [option(s)] [paketname oder rpm-datei mit absolutem pfad]

RedHat Package Management; Configs bleiben nach löschen eines Pakets erhalten (mit find nach "*.rpmnew" oder "*.rpmsave" suchen, falls mans löschen will)!

-i install
-e erase (uninstall)
-F freshen (nur dann Paket(e) updaten, wenn vorhanden)
-U upgrade (upgraden oder installieren wenn Paket(e) nicht vorhanden)
-V verify (prüft die Integrität von Paketen)
-q query (fragt halt was :)
-h hash (Fortschrittsanzeige mit "#")
-v verbose (zeigt an was es tut)

anwendbar bei "-q" und tw. bei "-V"
-a all
-f file (zu welchen Paket gehört Datei xy?)
-p package (infos aus einer rpm-datei beziehen und nicht aus der rpm-db)
-i info (gibt eine Beschreibung des Pakets aus)
-l list files (gibt alle Datein aus die zum Paket gehören)
-c config file (fragt nur nach Datein, die als config vermerkt sind)
-d documentation (fragt nur nach Datein, die als docu vermerkt sind)

nur mit vorsicht anwenden:
--nodeps de/installiert ohne Abhängigkeiten
--force erzwingts
--requires gibt aus von welchen anderen Paketen dieses Paket anhängt
--provides gibt aus was es installiert bzw. anderen Pakete anbietet
--whatrequires xy welches Pakete benötigen Paket "xy"? (vorm Deinstallieren praktisch!)
--whatprovides xy welches Paket bietet "xy" an?

Beispiele:
rpm -ivh /ganz/weit/weg/xy.rpm installiert das Paket mit Fortschrittsanzeige
rpm -qf /bin/ls gibt aus zu welchen Paket /bin/ls gehört; Ausgabe:
coreutils-4.5.3-19 Paketename-Version-Releasenummer
rpm -ql coreutils gibt alle dem Paket zugehörigen Datein aus (befragt rpm-db)
rpm -qpl coreutils /ganz/weit/weg/corutils-xy.rpm (befragt das Paket)
rpm -qf /bin Ausgabe:
filesystem-2.2.1-3
rpm -qi coreutils gibt Name, Version, Installdatum, Group ("System Environment/Base")...
rpm -qli coreutils dito, gibt zusätzlich noch die zugehörigen Datein aus
rpm -qc coreutils listet die Config-Datein auf
rpm -qd coreutil listet die Doku-Datein auf
rpm --import PUBKEY importiert einen Public-Key (für signierte Pakete)
rpm -qa gpg-pubkey-xy fragt den Key ab
rpm -e PUBKEY löscht den Key wieder
rpm -Va prüft alle Pakete ("SM5DLUGT x [datei]" Size, Mode/Permissions,
md5sum, device?, readLink path mis-match?, owning user, owning group, timestamp differs;
"x" c=config, d=doku, " "=file) - Aja, Keine Ausgabe wenn alles passt!
rpm -V coreutils prüft das installierte Paket coreutils
rpm -Vf /bin/ls /bin/ps prüft die Datei /bin/ls und /bin/ps (ALARM, wenn die beiden nicht stimmen!
Sofort Netzwerkstecker ziehen; "/var/log/*" sichern, prüfen ob Tage in den Logs fehlen;
PC runterfahren, Platte auf anderem System als read-only mounten und analysieren)

rpmbuild

Erstellt RPMs

tar [option] [archiv] [quelle/ziel]

-c create
-r append
-t test/list/view
-x extract
-d difference (Empfehlung: im gleichen Verz ausführen wie zum Erstellungszeitpunkt)
-v verbose (zeigt an welche Datein gepackt werden)
-X [datei] exclude ([datei] enthält eine Dateiliste die nicht ge/entpackt werden sollen)
-P absoluten Pfad speichern
-z gzip *.gz, *.tgz (Linux) ident mit "tar -c /quelle | gzip > archiv.tar.gz"
-j bzip2 *.bz2 (Linux)
-Z compress *.Z (Unix üblich)
-f file (nach f muss unbedingt der Dateiname folgen!)
zB. tar -cvzf archiv.tgz /home/ich

zip/unzip

(de)komprimiert wie PKZip/WinZip

User-Management

chage [option] [user]

(change age) ändert Zeitbeschränkungen eines Accounts, auch interaktiv
-d YYYY-MM-DD (lastday) gibt an wann das pw das letzte mal geändert wurde (default "")
-E YYYY-MM-DD (expiredate) acc läuft am # ab und wird gesperrt (default "")
-I # (inactive) (default -1) falls pw abgelaufen, gnadenfrist von # Tagen um das pw zu ändern, nach ablauf wird das acc gesperrt.
-l (list)
-m # (mindays) pw muss mind. # Tage bebehalten werden (default 0)
-M # (maxdays) pw ist max. # Tage gültig (default 99999, besser 90)
-W # (warndays) warnt # Tage vor pw-ablauf (default 7)

chpasswd

ändert ein Passwort skript-gesteuert; zB. "echo NAME:PASS | chpasswd"

finger

Gibt Infos über die aktuell eingeloggten User aus (der finger-server wird idR nicht mehr verwendet, da er "zu gesprächig" ist) zB. Wo arbeiten sie? Tel? E-Mail? ...

groupadd

Gruppe erstellen

groupdel

Gruppe löschen

groupmod

Gruppe ändern, oft ist es schneller die /etc/groups zu editieren

grub-md5-crypt

erzeugt verschlüsselten Passwort-String, der bei useradd angeben werden kann

last

gibt die letzten logins/outs aus; liest die /var/log/wtmp aus

lastlog

liest die /etc/passwd aus und gibt aus wann jeder user zum letzten mal eingeloggt war (Alarm, wenn zB. die system-user wie "bin", "disk", oä. was anderes als "never logged in" haben!).

passwd [user]

Ohne Parameter ändert es das eigene Passwort, mit das des jeweiligen Users (root-only!)
siehe "chpasswd"

useradd [user]

User erstellen
-m home-verz erstellen mit entsprechenden Zugriffsrechten
-p 'string' Passwort-String angeben (unbedingt einfache Anführungszeichen verwenden!
siehe "grub-md5-crypt")
-k skel-verz
-s shell (zb. /bin/bash, /bin/false oder /bin/nologin)

userdel [user]

User löschen
-r löscht das zugehörige Home-Verz

usermod [user]

User ändern; oft ist es schneller die /etc/passwd zu editieren

w

Welche User sind eingeloggt? Was tun sie?

who

Welche User sind eingeloggt?

System-Infos oä.

date [options]

gibt das aktuelle/ein angegenes Datum in benutzerdefinierter Formatierung aus.

dmesg

zeigt die Datei "/var/log/dmesg" an

env

gibt alle Umgebungvariablen aus

free

Zeigt die Belegung des RAMs und des Swap-Space an

id

Gibt Infos über den User an
-g nur group-id ausgeben
-u nur user-id -
-n username ausgeben

logger [option] [text]

schickt Log-Msg an syslogd, zB für Skripte/Cron-Jobs gut oder auch nur zum Testen;
siehe "/etc/syslog.conf", "/etc/sysconfig/syslog", "/var/log/*"
-f FILE loggt in die Datei FILE
-i loggt die PID mit
-s meldung auch nach STDERR ausgeben
-p (priority) "facility.level", siehe "syslog*"

tty

Gibt aus auf welchem Terminal man gerade eingeloggt ist

uptime

Zeigt an wielange die Maschine schon rennt

uname

Zeigt Infos über das OS an zb. Linux/Unix, Kernel-Version, etc...
-a zeigt alles an

Festplatten-Management

chroot [dir] [cmd]

changed root, ändert das "/"-verz auf [dir], ggfalls kann nacher auch gleich [cmd] ausgeführt werden

dd [option]

(dump disk) kopiert Dateisystem/Partitionen Byte per Byte
if=ab Input-File ist "ab"
of=xy Output-File ist "xy"
bs=1024 Blocksize auf 1024 Byte
count=5 5 Blöcke schreiben (ohne Angabe wird bis EOF geschrieben)
zB. "dd if=/dev/hda of=mbr count=1 bs=512" (kopiert den mbr von der ersten HDD)

df [option]

(disk free) zeigt an wieviel Platz auf den Platten frei ist (in Blocks)
-h (human readable)
-g -m -k giga/mega/kilobyte
-b byte

du [option] [file]

(disk usage) zeigt an wieviel Platz (ein) Verz/Datei(n) auf der Platte belegen
-s (summarize) fasst Belegung zusammen - sonst wird jede Datei einzeln angezeigt
-g -m -k giga/mega/kilobyte
-b byte

fdisk

Erstellt/Löscht Partitionen

mount [option] [device] [mountpoint]

Hängt Datenträger/Partitionen ins Dateisystem ein; siehe "umount"
(ohne Paramter) zeigt die aktuell gemounteten Dateisysteme an
-o Option zb: "ro,remount" (ro = read only, rw = read write)
-t Filesystem-Typ angeben (meist nicht nötig) zB.: ext2/3, vfat, msdos, reisefs, smbfs, ntfs...
zB.: "mount -t vfat /dev/hda3 /mnt", "mount -t iso9660 /dev/cdrom /media/cdrom", "mount -t nfs 192.168.206.3:/data/red-hat/ /mnt", "mount -t smbfs -o username=susi,password=xy // 192.168.0.1/share /mnt

umount [device oder mountpoint]

Hängt Datenträger/Partitionen aus dem Dateisystem wieder hinaus; siehe "mount"

updfstab

veranlasst das System die /etc/fstab neueinzulesen

Zeit

anacrontab

holt versäumte Cron-Jobs nach, siehe "/etc/anacrontab"
Der Schmäh funktioniert so: in "/etc/cron.daily" ist ein kl. Script mit "anacron -u [LABEL]" - das wird ausgeführt und setzt den Zeitstempel für anacron wieder zurück.

at [zeit]

führt einmalig einen Befehl zu einem best. Zeitpunkt aus; Job auf Subshell eingeben und wmit CTRL+d beenden
now +5min führts in 5 minuten aus

mm/dd/yy	dd.mm.yy	führts zu einem best. datum aus
noon	12:00 mittag	
midnight	0:00	
teatime	16:00	

atq

listet die at-queue auf

atrm

löscht einen Job aus der at-queue

batch

wie "at" aber führt nur aus wenn CPU-Load unter 0,8 (Proz/sek) liegt. Wenn nicht, dann später

crontab [option]

führt immer wieder Befehle zu best. Zeitpunkten aus; siehe at, batch, /etc/cron*, /var/spool/cron/
-e editiert die eigene crontab
-l (list) eigene crontab ausgeben
-r (remove) eigene crontag löschen
-u [user] betrifft nun die crontab von [user]

Prozesse

bg [pid]

führt einen Prozess im Hintergrund aus; ohne Pid wird der letzte gestoppte Prozess ge"bg"t
bei "%2" wirds Shell-Job Nr. 2

fg [pid]

führt eine Prozess wieder im Vordergrund aus, ohne Pid letzter gestoppter Prozess wird ge"fg"t
bei "%1" wirds Shell-Job Nr. 1

jobs

listet alle Prozesse auf die gerade im Hintergrund sind bzw. gestoppt sind in dieser Shell!

kill [signal] [pid]

tötet Prozesse anhand der Pid*HARHAR* :)
-l listet alle Signale auf
-15 SIGTERM (default)
-9 SIGKILL, tötet sehr sicher!
-1 SIGHUP (veranlasst die meisten Daemons die Config neueinzulesen)
-5 SIGTRAP

killall [signal] [cmd]

tötet Prozesse anhand des Cmd
-i (interactive)
-list listet die Signale auf
analog zu "kill": -TERM, -KILL, -HUP, -TRAP

ps [option]

gibt aus welche Prozesse gerade laufen
a (all info) zeigt Pid, tty, Time, status und Cmd an
au zeigt an unter welcher uid die Prozesse laufen
ax alle Prozesse anzeigen (auch die ohne tty)
Status: D (uninterruptible), R (running), S (sleeping), T (traced, stopped), Z (Zombie, defunct process), < (high priority process), N (low priority process), W (?)

pstree

gibt einen Prozessbaum aus
-a (all)
-p (pid ausgeben - bei "init" ist die pid immer 1!)

renice [priority] [pid]

ändert die Prozess-Priorität

top

Prozess-Monitor
Tasten: ?/h (hilfe), k (kill), s (set update intervall), n (change num. of processes), r (renice process, -20 bis +19)

Netzwerk

arp [option]

Für den "Adresss Resolution Protocoll"-Cache (IP in MAC auflösen)
-n (numeric) zeigt nur IPs an
-f DATEI liest Liste ein für FESTEN Arp-Cache (Format: "IP MAC", einfach ein "arp -n>file" als Vorbild nehmen und die unnötigen Infos rauslöschen)

dig [host/ip]

Werkzeug für Namensauflösung (IP <-> DNS-Name); Gibt IMHO auch viel überflüssige Infos aus

exportfs [option]

Infos über die NFS-Freigaben
-r (reload) liest die /etc/exports erneut ein
-v zeigt die lokalen NFS-Freigaben an

ifconfig [options]

Infos/Konfig von Netzwerkkarten
(ohne) gibt Auskunft über die momentan aktiven NICs (Ethernet, MAC-Addy, IP, Subnet-mask, MTU, RX & TX (receivend/transmitted pakete), collisions, Interrupt, Base Addy...)
[device] [ip] up/down [netmask addr] de/aktiviert ein Gerät zB. "ifconfig eth0:1 192.168.4.10 up" (erstellt eine Pseudo-NIC), "ifconfig eth0:1 down"

iptables [options]

Erstellt Regeln für TCP/UDP/ICMP ua.; siehe iptables-tutotial.frozentux.net und Skriptum!!; gibt Auskunft über allmögliche und ist DIE Firewall unter Linux. Anm.: Wenn man IP-Adressen angibt sollte man immer auch die dazugehörige Subnetmask anhängen zb. "10.0.0.1/255.0.0.0".
-p icmp -h gibt alle verfügbaren ICMP-Meldungstypen aus (echo_request (ping), echo_reply (pong), ...)

netstat [option] [host/ip]

wie unter Windows (bzw. umgekehrt ;), zeigt die aktuellen TCP-Verbindungen an

NFS

"/etc/init.d/nfs start", "/etc/exports", "/etc/init.d/portmap" (rennt idR bereits);
siehe "exportfs", "showmount" und "mount"

nmap [options]

Universaler Portscanner, empfehle immer die aktuelleste Version zu verwenden

nslookup [option] [host/ip]

Werkzeug für Namensauflösung (IP <-> DNS-Name); wird aber voraussichtlich von "dig" abgelöst

ping [option] [host/ip]

Standardtest für Erreichbarkeit eines Computers; mit CTRL+c abbrechen
-c 5 schickt nur 5 echo_requests, statt fortlaufend weiter
-b Broadcast-Ping (bei Debian nicht nötig), erfordert eine Netwerk-Addy!

route [option]

Info über aktuelle Routen zu anderen Netzwerken, löscht/fügt neue Routen ein
-n (numeric) gibt nur die IPs aus (meist nützlicher), statt Namen

Samba

Server für Windows/SMB-Clients (meldet sich afaik als NT 4.5); siehe "smbclient", "smbpasswd", "/etc/samba/*", "mount -t smbfs", "testparm", "smbd", "nmbd"

scp [options] [source] [target]

secure copy
-i [file] gibt den Private-Key an
-p wie bei "cp", preserve times
-r rekursiv kopieren
-P 22 Verbindung auf Port 22 aufbauen
-o andere Optionen, siehe man-page
source/target in Form von "[user@]host:/dir/file" bzw. lokal "/dir/file"

sftp [options]

Secure FTP (im ggsatz zu scp können in einer Sitzung mehrere Datein übertagen werden)

showmount [option] [host/ip]

zeigt Mountbare NFS-Exporte an
-e [IP] zeigt die verfügbaren Exporte auf eigenem oder auf entfernten Rechner an
-a zeigt an welche Rechner gerade auf welche Freigabe zugreifen bzw. gemountet haben

smbclient [option] [host]

Client für Samba
-M schickt eine PopUp-Nachricht an einen Rechner
//host/share greift auf die Freigabe "share" am Rechner "host" zu.
-N nicht nach Passwort fragen
-U xy Remote mit dem Usernamen "xy" anmelden
-L liste die Shares eines Rechners auf

smbpasswd

Samba-User-Management, für jeden Samba-User muss ein lokales Account vorhanden sein, ändert Passwörter, ... - Empfehle Smb-only-User lokal mit "usermod -l xy" zu locken.
xy ändert das Smb-Passwort für User "xy"
-a xy fügt den lokalen User "xy" den smbpasswd hinzu
-m xy erstellt ein Machine/Computerkonto (kein $ anhängen!)
-l sperrt ein Smb-Account

ssh [options] [host/ip]

Secure Shell; siehe "sftp", "scp", "ssh-keygen", "ssh-agent", "/etc/ssh/*", "~/.ssh/*"
-l nik einloggen mit dem Usernamen "nik"
-p 22 Port 22 verwenden
-2 erzwingt Protokoll v2
-i [file] gibt den Private-Key an
-x/-X de/aktiviert X11-Forwarding
-F file gibt ein alternatives Config-File an
-L port:host:hostport Port-Forwarding/Tunneling, lauscht lokal 127.0.0.1 auf "port" und wird an "host:hostport" auf der anderen Seite weitergeleitet.
-R port:host:hostport dito, lauscht aber auf "0.0.0.0" auf "port" (~ Arme-Leute-VPN ;)

ssh-keygen [options]

Genertiert/Importiert OpenSSH-keys (rsa, dsa)
-i importiert einen Key (zB. von SSH.COM)
-b 1024 gibt die Bit-Länge des Keys an (default 1024)
-f [file] gibt eine Datei als Quelle an
-t typ gibt den Typ des Keys an zb. (rsa1 (unsicher!!), rsa, dsa)
-l zeigt den Fingerprint eines Keys an
-p ändert die Passphrase eines Keys

swat

grafisches Config-Util für Samba (Vorsicht, überschreibt händisch erstellte conf!) und rennt idR auf 127.0.01:901- ggfalls mit "service swat start" starten - ich bevorzuge die conf selbst zu editieren.

tcpdump [option]

zeigt in Echtzeit an was für ARP/ICMP/TCP/UDP/...-Pakete reinkommen
arp zeigt nur Arp-Pakete an

testparm

Überprüft die syntaktische Korrektheit der "/etc/samba/smb.conf"

traceroute [option] [host/ip]

Verfolgt die Route zu einem anderem Rechner
-I (ICMP) wie ein Ping an jeden Rechner; praktisch wenn Traceroute via UDP fehlschlägt

Services, Runlevel & Hardware

chkconfig [option] [service]

vereinfacht die Arbeit beim erstellen vom Symlinks in den "/etc/rc.d/rc#.d" (RedHat spezifisch?)
--list prüft alle Symlinks
--del löscht alle Symlinks
--add erstellt alle Symlinks, wie im Script angegeben zB. "235"
--level 35 kudzu {on,off,reset} gilt nur die RL 3 & 5, erstellt on=Start, off=Kill, reset=Start und Kill-Links für das Service "kudzu"

init [option]

ändert den RL, siehe "telinit"
0 bis 6, S ändert den RL auf 0, 1/S, 2, 3, 4 oder 5
q weist init an die "inittab" neueinzulesen.

insmod [modulname]

lädt ein Modul

lspci [option]

listet die Geräte auf die am PCI-Bus gefunden wurden
-n (nummeric) gibt die IDs aus, statt lesbarer Hersteller- und Gerätenamen

lsmod

listet die aktuell geladenen Module auf; Ausgabe zB:
8139too 17975 1
mii 3123 0 [8139too] ("mii" wird von "8139too" benötigt)

modprobe [modulname (ohne *.o und Pfad)]

ent/lädt ein Modul und alle zusätzlich benötigeten
-r (remove) entlädt

runlevel

zeigt den vorhergehenden und den aktuellen RL an

service [service] [option]

damit man nicht "/etc/init.d/network start" eingegen muss, sondern nur: "service network start" (RedHat spezifisch?); options: start, stop, status, reload, ...

sysctl [option]

Wrapper (bzw. Tippersparnis) für "/proc/sys", siehe "/etc/sysctl.conf"

kernel.hostname	gibt den aktuellen Hostname aus
net.ipv4.ip_forward = 1	Routing aktivieren

Verschiedenes

bc

Taschenechner für komplexere Berechnungen, zB:

-ibase=2	setzt Inputbasis auf 2 (binär)
-obase=16	setzt Outputbasis auf 16 (hex)
zB. "echo '2 ^ 2'|bc"	gibt 4 zurück

run-parts [verz]

führt alle Datein mit gesetztem x-Recht in diesen Verz aus

startx

startet eine X-Session ;) (ajo, den X-Server kann man mit CTRL+ALT+Backspace beenden)

twm

startet den rudimentären Window-Manager "twm" (t für tiny?)

xinit

startet den X-Server

Anwendungs-Beispiele

Wörterbuch erstellen aus einer Textdatei

Aufgabe: ein Wort pro Zeile, Kleinbuchstaben, keine Sonderzeichen oder Zahlen:

cat DATEI |	DATEI ausgeben
tr -s '[:space:]/' '\n' |	alle Whitespaces gegen Newline tauschen (beachte Wörter: "and/or")
tr -d '[:punct:][:digit:]' |	alle Zahlen und Interpunktion rauslöschen
tr '[:upper:]' '[:lower:]' |	alle nach Kleinbuchstaben umwandeln
grep -v '^$' |	Leerzeilen löschen
sort | uniq	Sortieren und doppelte Zeilen elimieren

Alle User ausgeben die eine GID im Bereich 1-99 haben (System-User)

zB. Um zu überprüfen, ob die auch alle als Shell /sbin/nologin definiert haben

```
cat /etc/passwd | egrep '^[a-z]+:x:[0-9]+:[1-9][0-9]?:'
                          Username:x:UID   :GID        :
```

User nach der GID sortieren

```
cat /etc/passwd | cut -d : -f 1,4 | sort -n -t : +1 -2
cat /etc/passwd | cut -d : -f 1,4 | sort -n -t : -k 2,1
```

tar-ball installieren

tar -xzvf archiv-1.tar.gz -C /usr/src	(vorher ev. mit -t reinschauen, welche Verz-Struktur es hat)
cd /usr/src/archiv-1	(alles was jetzt kommt, finden im dortigen Verz statt)
less README	Readmes oder Release-Notes sollte man halt lesen
less INSTALL	Anleitung wie man dieses Teil installiert; Lesen!!
./configure --help	gibt die einzelen Optionen aus zB. --prefix=/usr/local/xy
./configure [options]	configure ausführen (ggfalls fehlen ein paar *-devel rpms)
make	kompilieren
make install	installieren; Binaries, Konfigs, ... in die jeweiligen Verz kopieren

Eine Meldung auf allen ttys ausgeben (zB. für Cron)

```
for X in `ls /dev/tty[1-6]`
do
        echo -e "\nHallo Welt\!" > $X
done
```

Booten mit root-Shell (mit Bootloader "Grub")

Einfach "e" drücken und dann kommen folgenden 3 Zeilen...

1 *root (hd0,0)*
gibt an von "/boot" liegt, erste HDD, erste Partition

2 *kernel /vmlinuz-2.4.20-8 ro root=LABEL=/*
gibt an wo der Kernel liegt (relativ zu "/boot"), wo die "/"-Partition liegt (statt LABEL, wäre auch "/dev/hda2" möglich
wenn man die Zeile editiert und um " init 1" ergänzt -> ROOT-SHELL
standardmäßig ohne PW!! *HARHAR* (bei LILO wäre es "init=1")

3 *initrd /initrc-2.4.20-8.img*
gibt an wo die "Initial Ramdisk" liegt (relativ zu "/boot"), in der initrd sind zB. Module drin, die der Kernel braucht beim Booten (zB. für einen SCSI-Adapter) - initrd ist unnötig, wenn Modul(e) im Kernel monolithisch integriert wurde.

Netzwerkkarte manuell einbinden

lspci	herausfinden um welche Karte (bzw. Chipsatz) es sich handelt zB. RTL8139
cd /lib/modules/2.4.20-8/kernel/drivers/net/	
ls	passendes Modul suchen (siehe "Dateisystem - Hierachie & Datein")
modprobe 8139too	Modul laden
lsmod	prüfen obs geladen is
/etc/modules.conf	editieren um das Gerät zu "verewigen", zB. "alias eth1 8139too"
cd /etc/sysconfig/network-scripts	NIC verewigen, damits nach nächstem Neustart funktioniert.
ifcfg-eth1	Dort diese Datei erstellen (Inhalt siehe "Dateisystem - Hierachie & Datein")
ifconfig eth1 192.168.0.123 up	Gerät aktivieren! (IP wäre hier gar ned nötig, da eh in ifcfg-*)

Shell-Programmierung (mit bash)

#!/bin/bash	"#!" = she-bang, damit linux weiß, dass mit der bash ausgeführt werden soll
#!/bin/bash -x	würde die Bash mit "-x" starten, vergleichbar mit "echo on" unter DOS
usage ()	Funktion "usage"
{	
echo "blabla $1" ...	gibt "blabla Hallo Welt" aus
}	
usage "Hallo Welt"	ruft die Funtion "usage" auf
echo $0 $1 ... $9	gibt Namen des Skripts selbst zurück, Arg 1 ... Arg 9
echo $* $@	alle Args (1-9) in einem String; alle Args jeder für sich ein eigener String
echo $# $?	Anzahl der Args, Exitcode des letzten Cmd
VAR="xy"; echo $VAR ${VAR}	setzt die Variable VAR auf den Wert "xy" und gibt ihn 2x aus
# Kommentar	
:	Null-Befehl, tut nichts ausser zB. leere if-Zweige aufzufüllen
. scriptfile	"Source" ermöglicht, dass Variablen übernommen werden (ohne "export")
export VAR	VAR ist auch nach Beenden des Scripts/Subshell vorhanden
if [! BEDINGUNG]; then	Wenn negierte("!") Bedingung wahr ist (exit 0), dann gib "JA" aus
echo "JA"	BTW, statt "[ ]" geht auch "test" und "[[ ]]", siehe "man test"
else	Wenn nicht, dann nichts *g* ("else :" ist in dem Bsp überflüssig)
:	
fi	
case $X in	Wie "select case" (VB) oder "switch" (C/C++)
Muster1) Cmds ;;	
*) Cmds ;;	~ default/case else
esac	
for X in 1 2 3; do echo $X; done	Schleifenkörper wird für "1", "2" und "3" einmal ausgeführt
for ((X=1; X<=3; X++)); do echo $X; done	Zählschleife
while true; do Cmd; done	Schleife, solange bis Bedingung falsch wird (!=0)
until who | grep "root"; do sleep 2; done ; echo "Der Meister ist anwesend"	solange bis wahr

wird	
read X	liest was in die Variable X ein; -p PROMPT, -t TIMEOUT_IN_SEC, -d " " (alles bis zum ersten Leerzeichen einlesen), -n ZAHL (der zu lesenden Zeichen), -r (kein Newline), -s (silent, zb für Passwort-Abfragen)
basename /var/test	gibt "test" zurück
dirname /var/test	gibt "/var" zurück
exit 0	verlässt das Script und setzt den Exitcode auf 0

Dateisystem - Hierachie & Datein

/ "root of all evil" *g*

bin/ wichtigeste Befehle sind hier zb. ls, cat, mkdir, ...

boot/ Bootdateien und Kernel

dev/ Gerätedatein (bzw. Treiberschnittstellen)

- *hda/* erste IDE-Festplatte/Primary Master
- *hda1/* erste Partition der ersten Platte
- *fd0/* Floppy-Laufwerk
- *null* Schwarzes Loch
- *urandom* Gibt immer zufällige Werte zurück
- *zero* Gibt immer \0 zurück
- *...*

etc/ Konfigurationsdatein

- *anacrontab* anacron-Config; Aufbau:
  period delay job-id cmd (Wenn X Tage vergangen sind; Zeit in min vom Start des Servers bis zur Ausführung; Label; Cmd) zB.
  "1 5 cron.daily run-parts /etc/cron.daily"
- *at.allow* wie "cron.allow"
- *at.deny* wie "cron.deny"
- *bashrc* bash-Konfig für alle User, zB.: umask, Prompt, Aliases, ...
- *cron.allow* Liste der User die eine Crontab haben dürfen (Restriktiv ist besser)
- *cron.deny* Liste der User die keine haben dürfen; ein User pro Zeile
- *cron.d/* hm... vom System irgendwas?
- *cron.daily/* alle Datein mit x-Recht werden täglich ausgeführt, zB. Backup oä.
- *cron.hourly/* dito, stündlich
- *cron.monthly/* dito, monatlich
- *cron.weekly/* dito, wöchentlich
- *crontab* die System Crontab (sollte man eher nicht herumpfuschen); Aufbau:
  min hour day month weekday user cmd
  0-59 0-23 1-31 1-12 0-7 (0=sonntag)
  mögl. Wildcards je Feld: "*", "0,1,2,3", "0-3", "*/2" (zB. alle 2 Stunden), "3,*/5,7-8"
  zB. "0 4 * * * run-parts /etc/cron.daily"
- *exports* Config für die NFS-Exporte; Aufbau:
  /SHARE CLIENT(OPTIONS) [weitere Clients] zB. "/var/nfs *(ro,sync)"
  Clients: * (alle), *.local (alle aus der Domain "local"), 192.168.4.1 (nur der da) 192.168.4.0/255.255.255.0 (ganzens Subnet)
  Optionen: sync, no_root_squash, anonuid=123, squash_uid=500-999, rw, ro
  squashen = mappen einer UID auf eine andere (idR "nobody" od. "nfsnobody")
- *fstab* Tabelle mit Devices, Mountpoints und Optionen; Aufbau:
  device mountpoint fs options... zB.
  /dev/hda1 / ext3 ro,rw,nodev,nosuid,noguid,noexec,user,auto,noauto,sync
  /dev/cdrom /mnt/cdrom auto ro,noauto,user,exec, ...
- *group* Gruppendatenbank; Aufbau:
  Gruppenname:Gruppenpasswort gesetzt (x):GID:User der Gruppe
- *grub.conf* config von Grub (bootloader)
- *host.conf* In welcher Reihenfolge erfolgt Namensauflösung zB:
  order hosts,bind (erst hosts-Datei befragen, dann Nameserver)
- *hosts* Zur Auslösung von Hostname in IP; Aufbau:
  IP FQDN-HOSTNAME [ALIAS] (zB. "127.0.0.1 localhost.localdomain localhost")

init.d/ Symlink auf */etc/rc.d/init.d* (unter RedHat zumindest)

inittab Run-Level Config-Datei wird NUR beim Systemstart eingelesen (0=halt, 1=Single, 2=Multiuser, 3=Multiuser+Net, 5=Multiuser+Net+X11, 6=reboot); Aufbau:

id:5:initdefault: setzt das default-RL
si:sysinit:/etc/rc.d/rc.sysinit Systeminitialisierung (?)
l0:0:wait:/etc/rc.d/rc 0 welche RL sind verfügbar, was ausführen?
... (gleiches für RL 1,2,3,5 und 6)
sa::ctraltdel:/sbin/shutdown -t3 -r now was beim Affengriff passiert
1:2345:respawn:/sbin/mingetty tty1 RL2345, Wenn Prozess stirbt, wirds
... (gleiches für tty2 bis 6) von der Shell neugestartet (!="once")
x:5:respawn:/etc/X11/prefdm -nodaemon X11 im RL 5 starten
~~:S:respawn:/sbin/sulogin PW-Abfrage im RL 1 (single)

login.defs Definiert ua. den UID-Bereich der normalen User
password max. days, - min. days, - min. length, - warn age

logrotate.conf Löscht/Packt alte Log-Files; Aufbau
{daily,weekly, monthly} Wie oft soll eine Rotation ausgeführt werden
rotate 4 Hebe die letzten 4 Logs auf (bei "weekly" = letzte 4 Wochen)
create erstelle neue leere Logs nach Rotation
compress gzippe die alte Logs (IMO praktisch!)
include /etc/logrotate.d RPMs werfen dort ihre Info-Files rein
/var/log/wtmp { Datei/Paket spezifische Angaben
monthly
create 0664 root utmp
rotate 1 }

modules.conf Welches Modul brauchts für welches Gerät?; Aufbau:
alias [DEVICE] [MODUL] (zB. "alias eth0 8139too")

motd "message of the day" wird beim Login angezeigt

nologin Bei Existenz kein Einloggen normaler User möglich, Inhalt wird Usern ausgeben

pam.d/ "Plugable Authentication Modules"

passwd Userdatenbank; Aufbau:
Username:Passwort gesetzt (x):UID:GID:Fullname:Home-Verz:Shell
Wenn Login verhindert werden soll, Shell auf /bin/false od. /sbin/nologin setzen
UIDs: 0 = root, 1 bis 98 system user/daemons, 99 idR. nobody,
normale User ab 100 oder 500 bis 65534, 65535 idR. nfsnobody

profile bash-Voreinstellungen, zB. HISTSIZE. ...

rc.d/ runtime-configuration glumpat
init.d/ die Start/Stop-Scripte
network startet,stoppt ua. das Netzwerk; "chkconfig"-Aufbau:
 # network stellt "network" zur Verfügung"
 # chkconfig: 2345 10 90
soll im RL 2,3,4 und 5 laufen, Nr. 10 für Start, 90 Kill-Script
rc zuständig, um Dienste zu killen/starten bei RL-wechsel
rc0.d/ Start und Kill-Scripte (bzw. Symlinks nach zB. "../init.d/network")
K90network hohe Zahl, killt spät das NW
S10network niedrige Zahl, startet NW früh
rc6.d/ dito, für jeden RL
rc.local für eigens Glumpat, damit man nicht in der *rc.sysinit* gepfuscht wird
rc.sysinit wichtige Dinge beim Booten: Module laden, fsck, Welcome-Banner...

resolv.conf Gibt Nameserver ua. an; Aufbau:
nameserver 123.45.67.98
domain NAME gibt den Namen der lokalen Domain an

samba/ Config-Verz für Samba
smb.conf Samba-Config
smbpasswd Liste der lokale User die sich via SMB anmelden dürfen
smbusers Mapping-Datei zB "Administrator" lokal auf "root" mappen

securetty beschränkt die Terminals wo sich root einloggen darf (kein Einfluß auf sshd!)

services Welche Ports für welche Anwendung? zB. "ftp 21"

shadow User-Passwörter ua.; Aufbau:
Username:
verschlüsseltes Passwort (md5), bei nologin "" oder "!!":* \
Datum der letzten Passwort-Änderung (Tage seit 1970): \
Tage wielange das Passwort behalten werden muss: \
Ablauf der Passwortgültigkeit in Tagen:Hinweis vor Ablauf in Tagen: \
Locken des Account nach Tagen Ablauf: \
Account Ablaufdatum

ssh/ Config-Verz für OpenSSH
- *ssh_config* gobale ssh-config (client), "Protokoll 2" empfehlenswert
- *sshd_config* config für sshd (server), NUR Protokoll v2 verwenden!!; empfohlene Optionen: Protokoll 2, UsePriviledgeSeparation yes, PasswordAuthentication no (RSA-Auth is sicherer)
- *ssh_host_key** Host-Key, Fingerprint notieren und achtung wenn verändert!!

skel/ Skelett(e)/Vorlagen für User-Home-Verz

sysconfig/ diverse Configs
- *network* Routing & Host-Infos für alle NICs
- *network-scripts/* "ifcfg-[DEV]" zT. nur Symlinks auf Datein in "../networking"
  - *ifcfg-eth0* Config des eth0
    DEVICE=eth0, IPADDR=192.168.0.123, ONBOOT={yes,no}
    NETMASK=255.255.255.0, GATEWAY=192.168.0.1,
    TYPE=Ethernet, BOOTPROTO={none,static,dhcp,bootb}
    USERCTL={yes,no} (nicht-root dürfen Gerät kontrollieren?)
- *networking/* (je nach Distri liegen hier die "ifcfg-*"-Datein)
- *syslog* Config für das RL-Script zB: SYSLOGD_OPTIONS="-m 0 "
  "-r" (accept remote logs) "-x" (keine DNS-Namensauflösung)

inputrc *örk* halt ein paar Einstellungen für die Konsolen

sysctl.conf Einstellungen wie "net.ipv4.ip_forward = 1" ("." werden zu "/"), siehe */proc*

syslog.conf Config für "syslogd", siehe /etc/logrotate.conf, /etc/sysconfig/syslog
Neueinlesen der Config mit "/etc/init.d/syslog reload" erwirken; Aufbau:
WAS.WICHTIGKEIT OUTPUT (bzw. *"facility.priority"*) zB.

kern.* /dev/tty10	(alle kernel-msg auf tty10 ausgeben)
. @fqdn-hostname/ip	(schickt alles zu einem anderen Rechner)
*.info;mail.none;authpriv.none	/var/log/messages ("none", weil eh -> maillog)
mail.* /var/log/maillog	
*.emerg *	(geht an alle consolen und logfiles)
locoal7.* /var/log/boot.log	

Priotitäten: debug, info, notice, warning, crit, alert, emerg, none

home/ Heimatverezichnisse der normalsterblichen User

nik/ Home-Verz eines Users (sollte Modus von ‘700’ haben)
- *.bashrc* wird beim Login ausgeführt (wenn Bash als Shell)
- *.bash_history*
- *.bash_logout*
- *.ssh/*
  - *known_hosts* pro Zeile Hostname/IP und zugehöriger Host-Key
  - *id_rsa* Private-Key
  - *id_rsa.pub* Public-Key
  - *authorized_keys* pro Zeile ein Public-Key (Liste von Keys, die sich hier anmelden düfen)
- *.vimrc* config für den vi/vim zb. set number, set ai, set tabstop=4, ab mfg "Mit freundlichen Grüßen", ...
- *.xinitrc* Config für den X-Server (Script wird beim start von X ausgeführt) meist steht "kdestart" oä drin

lib/ Systembibliotheken (ua. Kernel-Module)

modules/
- *2.4.20-8/* (Versionsnummer des Kernels)
  - *kernel/*
    - *drivers/*

net/ Netzwerkkarten Kernel-Module 3c=3com, 8139=RealTek, de=D-Link, e100=IntelPro100, ne2k=recht universaler Treiber, via=VIA, ...

mnt/ temporärer Mountpunkt (zB. für CD-Rom, Floppy, etc...)
opt/ optionale Software, meist KDE ua. gr. Pakete
proc/ beinhaltete Informationen zu laufenden Prozessen
sys/
kernel/
hostname der aktuelle Hostname *g*
net/
ipv4/
icmp_echo_ignor_broadcasts 1=keine Antwort auf BC-pings
ip_forward 1=Routing aktivieren; fixieren in */etc/sysctl.conf*
root/ Heimatverezichnis des root, siehe auch "/home/
sbin/ Systemprogramme, idR. nur für root
tmp/ Ablage für temporäre Datein, offen für jeden User (idR ein Modus von "rwxrwxrwt")
usr/ Die zweite Hierachie ("Unix System Resources")
X11R#/ X-Window
bin/
games/
include/ Headerdatein für Programme
lib/ allg. Bibliotheken
local/ lokale Software, die nicht mit der Distri kommt
share/
doc/ Doku zu Allmöglichem
src/ Sourcen (auch Kernel)
var/ variable Daten
lock/ "pid locks"
log/ Verz für Log-Datein (unterscheiden sich ein wenig je nach Distri); TIP: "tail -f file"
boot.log Meldungen bei RL-Wechsel
cron Log von cron
dmesg Meldungen während des Systemstarts zB. HW-Initialisierung ua.
lastlog Wann hat sich wer eingeloggt? - mit "lastlog" auslesen!
maillog alles was mit E-Mail zu tun hat (sendmail, postfix, exim, ...)
messages Kernel-Meldungen und alles andere
secure Dinge die zB. eine Authentisierung erfordern (login, ssh, ...)
wtmp gehört zu "last" & co
XFree86 Log des X-Servers
run/
utmp für die Cmds "w", "who", "finger"
spool/ Verz. für Spooldatein
mail/ beinhaltet (neue) Mails für jeden User
cron/ das Cron-Verz (für jeden User eine Datei)
nik crontab von "nik", Aufbau (Vgl. "/etc/crontab"), Aufbau:
min hour day month weekday cmd
cups/ Spooler für CUPS (Common Unix Printing System)
tmp/