Wichtige UNIX-Kommandos

M. Jäger – FH Gießen-Friedberg

24. Juni 2005

1 Einleitung

Dies ist eine Kurzübersicht über einige UNIX-Kommandos. Eine Berücksichtigung aller Kommandos ist nicht möglich, immerhin gibt es über 1000 Kommandos in einem „gut sortierten“ System. Genausowenig können die Kommandos vollständig oder auch nur annähernd vollständig beschrieben werden, die Manuale füllen meterweise Regalfläche.

Teilweise werden auch Kommandos aufgeführt, die nicht auf jedem . . . IX-Sytem, aber auf den Solaris- und Linux-Rechnern des FB MNI verfügbar sind, z.B. Standard X-Utilities wie *xterm* und auch Freeware, die in Quelltextform von GNU und anderen Anbietern für UNIX-Plattformen zur Verfügung gestellt wird.

Bei jedem Zweifel über Syntax des Aufrufs oder die Semantik eines Kommandos sollte man im Online-Manual die genaue Beschreibung nachlesen.

Als Ergänzung sieht der Autor die Lektüre einer Shell-Beschreibung und einer Beschreibung des UNIX-Dateisystems als unbedingt notwendig an. Für alle komplexeren Programme, z.B. den Editor *Emacs*, *awk*, *make*, den Compiler, den Debugger existieren eigene ausführliche Beschreibungen.

2 Einige UNIX-Kommandos im Überblick

apropos — Im Online-Manual nach Einträgen zu einem bestimmten Stichwort suchen.

Beispiel:

```
$ apropos directory
chdir    (2) - change working directory
chroot   (2) - change root directory
closedir (3) - close a directory
find     (1) - search for files in a directory hierarchy
          .
          .
          .
```

at — Zeitsteuerung für Kommandobearbeitung

Erlaubt die Bearbeitung eines Kommandos zu einer festgelegten Zeit. Siehe auch **crontab**.

bc — Programm für arithmetische Berechnungen

Für die Berechnung von 17 * 23 genausogut geeignet wie für komplexe mathematische Berechnungen mit beliebiger Genauigkeit.

awk — Textverarbeitungsprogramm

Unterstützt zeilenorientierte Verarbeitung von Textdateien in C- ähnlicher Notation (z. B. Suchen und Ersetzen von Zeichenketten). Recht umfangreiche Möglichkeiten. (Falls **gawk** oder **nawk** vorhanden sind, sollte man diese Kommandos verwenden.)

Fragen Sie nach einer separaten Beschreibung !.

basename — Teil eines Pfadnamens

liefert zu einem kompletten Pfad den letzten Teil

Beispiel: `basename /usr/egon/bin/print`
liefert als Ergebnis `print` (Standardausgabe)

bash — GNU Bourne-Again-SHell

Sowohl im interaktiven Bereich als auch im Hinblick auf Shell-Script-Programmierung sehr leistungsfähiger Kommandointerpretierer, Nachteil: recht groß

Lesen Sie bitte die gesonderte Bash-Beschreibung !

bg — Programm im Hintergrund weiterführen (Shell-Kommando)

(vergl. Bash-Beschreibung)

break — Abbruch einer Schleife
(siehe Shell-Beschreibung)

cal — Kalenderausgabe

Format: `cal` *month year*

calendar — Terminverwaltungs-Programm

cancel — Abbruch eines Druckauftrags

(vergl. **lpr** und **lpstat**)

cat — Ausgabe von Dateien, Hintereinanderhängen mehrerer Dateien

Format: `cat` *file_1 . . . file_n*

Konkateniert die Dateien file_1 . . . file_n und gibt die Ergebnisdatei auf der Standardausgabe aus.

Häufige Verwendung:
- `cat` *file_1* - Ausgabe von *file_1* auf den Bildschirm
- `cat` > *file_1* - Umlenken Standardeingabe nach *file_1* (Beenden mit <CTRL-D>)

cc — C-Übersetzer

Format: `cc` *options file_1 . . . file_n*

Auf den Sun-Rechnern ist der GNU-Compiler **gcc** als ANSI-C-Compiler verfügbar, der SUN-Compiler *cc* unterstützt die ANSI-Sprachnorm nicht.

cd — aktuelles Verzeichnis ändern (Shell-Kommando)

Format: **cd** *directory*

Falls kein Directory angegeben ist, cd $HOME

chgrp — Ändern der Gruppenzugehörigkeit einer Datei

chmod — Ändern der Dateizugriffsrechte

auch zur Änderung von „sticky-bit“, „suid-bit“, „guid-Bit“

(Siehe auch Beschreibung des Dateisystems, Abschnitt „Zugriffsrechte“).

Beispiele:

1. `chmod g+w /usr/include/*.h`
   Für Gruppenmitglieder (g) Schreibrecht erteilen (+w).
2. `chmod +x  jaeger/bin/*`
   Für alle Benutzerkategorien Ausführen (x) erlauben.
3. `chmod 755 lemacs`
   Zugriffsrechte gemäß Bitmaske 755 (oktal) setzen, d.h. „rwx“ für Eigentümer, „rx“ für Gruppe und sonstige.

chown — Besitzerwechsel einer Datei (Change owner)

Beispiel: `chown mueller /home/mueller/*`

cmp — Vergleich zweier Dateien

Gibt Nummern der Bytes und Zeilen aus, bei denen Unterschiede existieren (siehe auch **diff**). Für komplexe *Text*vergleiche ist Emacs gut geeignet (ediff-Kommandos).

compress — Datei komprimieren (Dekompression mit **uncompress**)

gzip komprimiert besser !.

cp — Kopieren von Dateien und Verzeichnissen

Die GNU-Version kann Verzeichnisse rekursiv kopieren. Falls diese nicht zur Verfügung steht, hilft folgendes Kommando:

```
cd <QUELLVERZEICHNIS>/..
tar cf - <QUELLVERZEICHNIS> | (cd <ZIELVERZEICHNIS>; tar xf -)
```

(rekursives Kopieren von ¡QUELLVERZEICHNIS¿ in das ¡ZIELVERZEICHNIS¿)

Format: `cp` *file_1 file_2*
`cp` *file_1 . . . file_n directory*

Beim ersten Format wird eine Kopie von *file_1* namens *file_2* angefertigt.

Vorsicht: Falls *file_2* schon existiert, wird es ohne Rückfrage ersetzt. Beim zweiten Format werden die Dateien *file_1 . . . file_n* in das durch *directory* bezeichnete Verzeichnis kopiert. Auch hier werden vorhandene Dateien gleichen Namens ohne Rückfrage ersetzt !

cpio — Kopieren von Datei-Archiven

In erster Linie zum Transport von Dateien zwischen verschiedenen UNIX Maschinen. Auch *device*-Dateien und *Links* werden „vernünftig“ verarbeitet.

cpp — C-Präprozessor

Wird normalerweise automatisch von **cc** aufgerufen

crontab — Scheduler für Hintergrund-Ausführung

crontab ruft gemäß einer Benutzer-spezifischen Tabelle Programme zu festgelegten Tagen und / oder Uhrzeiten auf. Sowohl für einmalige Ausführung zu einem späteren Zeitpunkt als auch für regelmäßige Aufrufe (z.B. Datensicherung) verwendbar.

csh — C-Shell

Kommando-Interpretierer mit Schwächen im History-Konzept und bei der String-Verarbeitung

date — Datum und Uhrzeit ausgeben bzw. setzen

dd — Konvertieren und Kopieren

(diverse Konversionen wie Blockgröße, ASCII-EBCDIC-Konversion) z.B. für Disketten-„Hardcopy“ oder Plattenspiegelung

diff — Dateien vergleichen

Im Gegensatz zu *cmp* erzeugt *diff* eine Liste von ed-Kommandos, die zur Beseitigung der Differenzen geeignet sind.

du — Plattenspeicherbelegung von Dateien und / oder Verzeichnissen berechnen

Insbesondere nützlich, um für einzelne Verzeichnisse oder einzelne Dateien den benötigten Plattenplatz anzuzeigen.

dump — Datensicherungs-Programm

Zurücklesen mit *restore*.

df — freien Plattenspeicher anzeigen

df gibt eine Übersicht über den pro Dateisystem belegten und verfügbaren Plattenplatz aus.

echo — Argumente ausgeben

Format: `echo` *arg_1 . . . arg_n*

Gibt die Argumente durch Leerzeichen getrennt auf die Standardausgabe aus. Verwendungsbeispiele:

- Ausgabe in interaktiven Kommando-Prozeduren
- Ausgabe von Variablenwerten, z. B. `echo $HOME`

(siehe auch **printf**)

egrep — siehe **grep**

emacs — GNU Emacs Editor

Editor und integrierte Entwicklungsumgebung für Programme und andere Dokumente. Siehe separate Beschreibung.

env — Umgebung für Programme definieren/anzeigen

Auch die Shell und der Emacs-Editor haben Befehle, um das Environment anzuzeigen bzw. zu verändern (getenv, printenv, setenv).

etags — Verweisdatei erstellen

Mit *etags* wird eine Emacs-TAGS-Datei erstellt, die Verweise auf Definitionen und / oder Verwendungen von Bezeichnern enthält. Hat man eine solche Verweisdatei einmal erstellt, kann man mit verschiedenen Emacs-Funktionen, z.B. *find-tag* schnell auf die Definitionen zugreifen. Die Verwendung bietet sich z.B. für Definitionen von C-Funktionen, -Typen usw. an.

eval — Argumente auswerten und resultierendes Kommando ausführen (Shell-Kommando)

Stellt man einer Kommandozeile ein *eval* voran, wird die Zeile vor der Ausführung von der Shell ausgewertet.

Diese zusätzliche Auswertung kann unter anderem benutzt werden, um Variablen als Verweise auf andere Variablen zu verwenden. (Simulation von Zeigern und Arrays!)

Beispiel:

```
for n in 1 2 3 4 5
do
    let a$n=2*$n     # auf der linken Seite von Wertzuweisungen
                     # kein eval noetig
    eval echo \$a$n  # geht nicht ohne eval !
done
```

exec — (Shell-Kommando) Programmaufruf innerhalb des aktuellen Prozesses. Die Ausführung des aktuellen Programms wird damit beendet !

exit — Shell beenden (Shell-Kommando)

expr — Berechnung arithmetischer Ausdrücke

Mit *expr* lassen sich auch Zeichenketten und Shell-Variablen verarbeiten.

In erster Linie ist der Befehl für kleine Berechnungen innerhalb von Kommandoprozeduren gedacht, z.B. zur Realisierung einer numerischen *for*-Anweisung. Sofern der verwendete Kommando-Interpretierer aber eingebaute Arithmetik hat (wie bash und csh), wird *expr* nicht benötigt.

fg — Programm im Vordergrund weiterführen (Shell-Kommando)

siehe Bash-Beschreibung

fgrep — siehe **grep**

file — Dateityp bestimmen

file versucht zu bestimmen, ob es sich bei einer Datei um ASCII-Text, ein Archiv, ein ausführbares Programm, Daten usw. handelt.

find — Dateien suchen

Sucht die Verzeichnis-Hierarchie ab einem anzugebenden Verzeichnis abwärts nach Dateien, die vorgegebene Suchkriterien erfüllen (z.B. im Hinblick auf Namen, Besitzer, Größe, Datum der letzten Modifikation, Dateityp . . .).

Bei der Suche über den Namen können *Patterns* angegeben werden.

Die Suchkriterien können mit logischen Operationen zu komplexen Kriterien verknüpft werden. Auf die gefundenen Dateien können beliebige Kommandos angewendet werden.

Beispiele:

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

Alle Dateien mit der Endung `.tmp` löschen

```
find /usr -type d -newer .backup -print
```

Alle Verzeichnisse unterhalb von `/usr`, die neuer sind als die Datei `.backup` anzeigen.

finger — siehe **who**

fsck — Konsistenzprüfung für Dateisystem

fsck überprüft ein Dateisystem auf logische Inkonsistenzen, so wie sie etwa nach einem System-Absturz entstehen. Beim Durchsuchen aller Verzeichnisse werden beispielsweise die Links jeder Datei gezählt und mit dem im I-Node abgespeicherten Link-Zähler verglichen.

Das System wird wieder in einen konsistenten Zustand überführt.

Anm.: Bei Linux gibt es für jeden Dateisystemtyp ein eigenes Kommando, z.B. *e2fsck* für *ext2*-Dateisysteme.

ftp — Dateitransfer im Internet

Syntax: `ftp` *internet-address*

Mit *ftp* lassen sich Dateien innerhalb eines Internetzwerks versenden und empfangen. Das Programm ist interaktiv und hat eine Online-Help-Funktion.

fvwm — Window Manager

Das X-Window System kann mit unterschiedlichen Window-Managern benutzt werden. Für Sun- und Linux-Rechner ist derzeit *fvwm* die beste Wahl. Der Window-Manager kann (wie alle X-Client-Programme) auf einem entfernten Rechner ablaufen !

gcc — GNU-Übersetzer für Assembler, C, C++ und *Objective C*

(vgl. „Hinweise für den Programmierer“)

gdb — GNU Source-Level Debugger

Guter Debugger, für verschiedene Programmiersprachen verwendbar. Erlaubt auch das Debuggen von Subprozessen und das Debuggen über Netz. Menü-orientierte Front-End-Programme sind *xxgdb* und *Emacs*.

Siehe gesonderte Beschreibung.

getty — Terminal-Polling

getty wartet, ob an einem zugeordneten Terminal ein Benutzer arbeiten will und initialisiert den Dialog. Wird vom **init**-Prozeß gemäß den Einträgen in der `inittab` gestartet.

ghostview — Previewer-Frontend für Ghostscript

Menü-orientiertes Front-end zum Lesen von Postsript-Dateien unter X-Windows.

gprof — GNU Profiler

Instrumentiert Programme für Performance-Messungen

grep — Pattern-matching in Textdateien

Suche nach Textzeilen, die zu einem regulären Ausdruck passen. Ausgabe der passenden (bzw. nicht passenden) Zeilen und /oder Zeilennummern (Varianten: **egrep** für erweiterte reguläre Ausdrücke, **fgrep** für besonders schnelles Suchen mittels einfachem String-Vergleich, d.h. ohne echtes Pattern-Matching)

gs — Ghostscript - GNU Postscript-Interpretierer

Mit *gs* kann man Postscript-Dateien für den Bilschirm und für eine ganze Reihe von Druckern aufbereiten. (vgl. auch **ghostview** und **ps-Tools**)

gtar — siehe **tar**

gzip — GNU Kompressionsprogramm (besser als **compress**)

Dekomprimieren mit **gunzip**. Standard-Endung der komprimierten Dateien: `.z` oder `.gz`.

hostname — Rechnernamen ausgeben

imkmf — Makefiles für X-Windows-Programme generieren

imkmf erzeugt aus (X-)Installationsspezifischen Konfigurationsparametern Makefiles für die Installation von Software.

init — System initialisieren

install — Dateien kopieren und Zugriffsrechte setzen

Zur Installation von Software

ispell — siehe spell

jobs — Job-Tabelle anzeigen (Shell-Kommando)

siehe Shell-Beschreibung

join — relationaler Datenbankoperator

kill — Prozess terminieren

Format: `kill` -*signal process-id* ...

Die Werte von *signal* werden im Zusammenhang mit dem signal- Unterprogrammaufruf beschrieben.

Übliche Werte für signal: 1 - „soft“ kill, 9 - „hard“ kill

latex — LaTeX- Übersetzer

ld — Binder (load)

Kombiniert mehrere Objektmodule zu einem ablauffähigen Programm. Die zu bindenden Objekltmodule werden als Parameter angegeben. Darüberhinaus sucht ld in Objektbibliotheken nach weiteren externen Modulen. Die erzeugte Programmdatei heißt `a.out`, wenn nichts anderes angegeben wird.

ld wird in der Regel nicht direkt, sondern von *cc* oder *make* aufgerufen.

Programme lassen sich mit statischen oder mit dynamischen Bibliotheken binden. Heute werden in der Regel dynamische Bibliotheken verwendet, um Speicherplatz zu sparen. Einer der Nachteile ist aber die Abhängigkeit der Programme von bestimmten Bibliotheksversionen.

ld.so — Binder-Lader

Der Befehl wird nicht explizit benutzt, `ld.so` wird aber beim Aufruf eines jeden Programms aktiviert. Dabei erfolgt unter anderem die Auflösung aller in der statischen Link-Phase nicht befriedigten externen Referenzen mittels der dynamischen Bibliotheken (z.B. `/lib/libc.so` – C-Standardbibliothek).

ldd — Zeigt benötigte dynamische Bibliotheken an (list dynamic dependencies)

ldd analysiert ein Programm, dessen Pfad als Parameter übergeben wird, und zeigt an, ob und welche dynamischen Bibliotheken von diesem Programm benötigt werden. Dabei wird gleich überprüft, ob die Bibliotheken auch vorhanden sind. Gesucht wird in den Verzeichnissen, die in der Environment-Variable LD_LIBRARY_PATH aufgeführt sind.

lex — Programmgenerator für lexikalische Analyse

Compilerbauwerkzeug, mit dem man aus einer Grammatik einen Scanner oder eine Scannertabelle zur Verwendung in C-Prgrammen generieren kann. In Verbindung mit **yacc** ein vielgenutztes Werkzeug zur Implementierung von Programmier- und Anwendersprachen.

lint — C-Quelltext-Überprüfung

Gegenüber cc erweiterte Überprüfung von Syntax und statischer Semantik. Auf unerreichbare Anweisungen, deklarierte aber nicht benutzte Variablen und andere Inkonsistenzen wird hingewiesen. Auch Portabilität kann in gewissen Grenzen überprüft werden.

Format: `lint` *options file* ...

ln — Verweis einrichten (link)

Format: `ln` *options file_1 file_2*

file_1 muß bereits existieren, *file_2* darf nicht existieren (vgl. aber die Option `-f`, die eine vorhandene Datei durch einen Verweis überschreibt).

Ein Verzeichnis-Eintrag für *file_2* wird erzeugt, der auf die schon existierende Datei verweist.

Die Option `-s` erzeugt einen **symbolischen Verweis**. Lesen Sie dazu bitte den Abschnitt über Links in der Dateisystem-Beschreibung.

login — Anwender-Zuschaltung

Abfrage von Benutzername und Passwort
Ausgabe der „message of the day“ (falls definiert)
Ausführung von /etc/profile
Setzen des Standard-Environments (HOME, PATH, SHELL, MAIL, TZ usw.)
Ausführung von $HOME/.profile (falls vorhanden)

lorder — Abhängigkeiten von Objektmodulen ermitteln

Wird normalerweise verwendet, um eine Objektmodul-Bibliothek so umzusortieren, daß der Lader ld in einem Durchgang alle externen Referenzen befriedigen kann.

lpstat — Status Drucker-Warteschlange anzeigen

lpr — Drucker-Spooler

Erzeugt neuen Druckauftrag in einer Drucker-Warteschlange. (vergleiche auch **pr**, **lpstat**, **cancel**)

ls — Datei- und Verzeichnis-Information anzeigen

Format: `ls` *options file* ...

Für alle übergebenen Datei- und/oder Verzeichnis-Namen wird der Name und beliebige weitere Informationen (aus dem Inode der Datei) ausgegeben.

Für jedes aufgeführte Verzeichnis wird Inhaltsverzeichnis erstellt (Dateinamen mit führendem ’.’ werden nur in Verbindung mit der Option ’-a’ angezeigt).

Beim Aufuf ohne Dateiname wird das aktuelle Verzeichnis als Parameter angesehen.

Die Optionen werden durch ein ’-’Zeichen gefolgt von einem oder mehreren Optionsbuchstaben (ohne Leerzeichen) gesetzt.

Wir beschreiben hier nur die Option -l:

Beispiel:

```
$ ls -l /
total 2587
drwx------   4 root   root      1024 Nov  6 18:37 Mail
-r--r--r--   1 bin    bin       3832 Jul  8 04:29 base.crc
drwxr-xr-x   2 root   root      2048 Nov 21 23:27 bin
drwxrwxr-x   2 root   root      1024 Aug  5 13:52 cdrom
drwxrwxr-x   3 root   root      1024 Nov 11 12:24 conf
drwxrwxrwx   2 root   root      5120 Nov 17 22:27 dev
drwxr-xr-x   7 root   root      4096 Nov 25 11:27 etc
-rw-rw-r--   1 root   root     98663 Nov 17 19:51 filelist.gz
-rw-rw-r--   1 root   root        11 Jul 13 00:21 gmd
drwxr-xr-x   5 root   root      1024 Oct  6 01:58 home
-rw-rw-rw-   1 root   root      1387 Jul 19 21:3
lrwxrwxrwx   1 root   root        15 Oct  6 01:58 iv -> /usr/iv
```

Hinter dem Typ der Datei in der ersten Spalte folgen die Zugriffsrechte für Besitzer, Gruppe und sonstige Benutzer, dann die Anzahl der Links, der Eigentümer, die

Gruppe, Die Größe, das Datum der letzten Modifikation un der Name. Bei symbolischen Links wird angezeigt, auf welchen Pfad sie verweisen.

m-Tools — DOS-Dateisystem-Utilities

mcd, *mcopy*, *mdel*, *mdir*, *mformat*, *mread*, *mwrite*, *mtype*, *mren*, *mattrib*, *mmd*, *mlabel*, *mrd*.

Die m-Tools sind eine Reihe von Programmen zum bequemen Zugriff auf DOS-Dateisysteme wie DOS-Disketten oder DOS-Festplatten-Partitionen. Die Datei-Argumente werden in DOS-Manier spezifiziert. Die Wirkung ähnelt den DOS-Kommandos (ohne das führende „m“).

m4 — Makroprozessor

Für beliebige Anwendungen einsetzbarer Makroersetzungsmechanismus Im Hinblick auf Makroersetzung mächtiger als der C-Präprozessor

mail — Senden, lesen, verwalten elektronischer Post

Anm.: Es gibt auf jedem System eine Reihe von Mail-“User Agent“-Programmen. Lassen Sie sich unbedingt beraten, welches das geeignete ist ! Textorientierte Programme mit vielen Funktionen sind z.B. pine und mutt, Programme mit GUI z.B. Mozilla Thunderbird und Emacs VM.

make — Hilfsprogramm zur Wartung modularer Programmsysteme

Veränderung in einer Source-Datei eines Moduls bedingen im allgemeinen Übersetzungsläufe, Binder-Aufrufe und andere Aktionen, von denen alle von der geänderten Datei in irgendeiner Weise abhängige Dateien betroffen sind.

So werden beispielsweise durch die Veränderung einer Definition in einer Definitionsdatei („header-file“ , „include-file“) alle Programme betroffen, die diese Definitionsdatei (durch eine include-Anweisung) verwenden.

Bei komplexen Programmsystemen sind die Abhängigkeiten (insbesondere indirekte Abhängigkeiten) der zum System gehörigen Dateien nicht mehr so einfach zu überschauen.

Die wesentliche Funktion von *make* ist es, ein Programmsystem nach Änderungen in irgendeiner zum System gehörigen Datei auf den neusten Stand zu bringen und die dazu nötigen Aktionen automatisch auszuführen. Zu diesem Zweck gibt der Benutzer die direkten Abhängigkeiten zwischen Dateien, sowie Aktionen zum Aktualisieren von Dateien in einem sogenannten *Makefile* an.

Die Abhängigkeiten, die in C-Programmen durch `#include`-Anweisungen entstehen, kann man durch den Compiler (gcc) ermitteln lassen. Er generiert diesen Teil des Makefile automatisch.

Abhängigkeiten können schematisch angegeben werden, z.B. kann man *make* mitteilen, daß für eine beliebige Datei *name* durch eine Änderung von *name.c* die Datei *name.o* betroffen ist und durch die Aktion `cc -c name.c` auf den neuesten Stand gebracht werden kann.

Will man nun eine neue Systemversion erstellen, so gibt man *make* den Namen des zu erstellenden Systems an. *make* berechnet aus den im makefile angegebenen(bzw. als Standard vordefinierten) direkten Abhängigkeiten alle Dateien, von denen das zu erzeugende System indirekt abhängig ist. *make* vergleicht dann Datum und Uhrzeit der letzten Modifikation für je zwei direkt abhängige Dateien und führt in der richtigen Abfolge die zur Aktualisierung des Gesamtsystems notwendigen Aktionen durch.

Die Verwendung von *make* für sämtliche Entwicklungsarbeiten wird dringend empfohlen !

Der make-Mechanismus läßt sich auch für andere Anwendungen einsetzen, z.B. für die Entwicklung modularisierter großer Dokumente.

man — Beschreibung von Kommandos, Shells, Dateiformaten, Bibliotheksfunktionen usw. anzeigen.

Format: `man` *Stichwort*

Beispiel: Das Kommando `man man` liefert Beschreibung für die Verwendung des online Manuals

Alternativen: **xman** und das Emacs-Kommando **manual-entry**.

mkdir — neues Verzeichnis erstellen

mkfs — neues Dateisystem erzeugen

mknod — spezielle Datei erzeugen

(siehe auch `/dev/MAKEDEV`)

more — Text seitenweise anzeigen

(besser: **less** oder **pg**)

mount — Dateisystem montieren

Zur Montage von Dateisystemen lese man die Dateisystem-Beschreibung.

Beispiele:

`mount`	aktuelle Mount-Tabelle (`/etc/mtab`) anzeigen
`mount -a`	Montage aller Dateisysteme gemäß Initialisierungs-Tabelle `/etc/fstab`
`mount -t nfs sun0:/usr/home /home`	NFS-Mount
`mount -rt hsfs /dev/sr0 /cdrom`	read-only Montage einer CD

mt — Bandgerät-Steuerung

Zur Ansteuerung von Bandgeräten, DAT-Streamern usw.

mv — Dateien (auch Verzeichnisse) übertragen/umbenennen

Das Kommando kann keine Verzeichnisse von einem Dateisystem in ein anderes übertragen. Hier sollte man zuerst das Verzeichnis kopieren und danach das Quellverzeichnis rekursiv löschen.

netstat — Netz-Status anzeigen

nice — Kommando mit niedriger Priorität ausführen

z.B. nicht-interaktive Programme (Compilerläufe), die die Interaktion nicht durch zu starke CPU-Beanspruchung stören sollen

Der *superuser* kann die Priorität auch erhöhen !

nis. . . — NIS+-Kommandos, siehe *yp. . .*

nohup — Kommando ohne Beachtung externer Unterbrechungen ausführen

Ein mit *nohup* gestartetes Programm wird auch nach dem Beenden der Sitzung weitergeführt (z.B. stundenlanger Programmlauf) !

nroff — Textformatierungsprogramm

Textformatierung für nicht-graphikfähige Ausgabegeräte

od — oktale, dezimale, oder hexadezimale byteweise Ausgabe einer Datei

(Alternative: Emacs-Hexadezimal-Editor: **hexl-find-file**)

pbm-Tools — portable Bitmap Tools

Eine Vielzahl von Programmen, um Bitmap-Dateien diverser Formate in das pbm-Format zu konvertieren, im pbm-Format zu verarbeiten und von pbm in andere Formate zu konvertieren.

Damit kann man beispielsweise einen X-Window-Bildschirm-Dump (vgl. *xwd*), skalieren, mit einem Rand versehen und dann in das Postscript-Format konvertieren.

Verarbeitet eine ganze Menge von Formaten, z.B. gif, tiff, postscript, jpeg.

ping — Netz-Testprogramm

Syntax: `ping` *hostname*

pr — Datei formatiert ausgeben

Sorgt bei der Ausgabe für Zeilen-, Spalten- und Seitenformatierung, Zeilen- und Spaltennumerierung, Ränder usw.

`pr` wird oft in einer Pipeline vor dem Drucker-Spooler `lpr` verwendet, der selbst derlei Formatierungen nicht durchführt.

printf — formatierte Ausgabe

Ähnlich wie in C-Programmen kann mit dem Kommando *printf* Ausgabe formatiert werden. Für die Verwendung in Kommandoprozeduren.

ps — Prozeßstatus ausgeben

Zeigt nach zu jedem Prozeß Informationen wie Prozeßnummer, Nummer des erzeugenden Prozesses, Priorität, Speicheradresse, Größe des belegten Speicherraums, CPU-Zeit, ausgeführtes Kommando usw. an.

Welche Prozesse berücksichtigt werden und welche Informationen angezeigt werden, hängt von den Kommando-Optionen ab, versuchen Sie z.B.: `ps -axl`

ps-Tools — Postscript-Tools

Eine ganze Reihe von Programmen zur Verarbeitung von Postscript-Dateien. Unter anderem **psselect** zur Selektion bestimmter Seiten und Seitenbereiche aus größeren Texten, **pstops** zum Drehen, Skalieren und Verschieben (z.B. 2 Seiten auf 1 Seite drucken).

pwd — aktuelles Verzeichnis anzeigen (Shell-Kommando)

read — Variablenwert lesen (Shell-Kommando)

restore — siehe *dump*

rcs — siehe SCCS

rlogin — remote login - über Netz in anderen Rechner einschalten. Sicherer ist *ssh*.

rm — löschen von Dateien

Format: `rm` *options file_1 . . . file_n*

Wichtige Optionen sind:
-f — keine Fragen stellen
-r — rekursiv alle Unterdirectories komplett löschen
-i — interaktive Bestätigung

Vorsicht: Man macht bei der interaktiven Arbeit schnell mal einen Tippfehler. Nehmen Sie an, Sie haben in Ihrem Teil des Dateisystems eine Reihe von „Datenmüll“ stehen und auf die eine oder andere Art alle diese Dateien mit dem Suffix `.schrott` benannt. Nehmen Sie weiter an, Sie wollen nun alle diese Dateien löschen.

Sie könnten vom Wurzelverzeichnis ihres Dateisystems aus mit dem Kommando

```
rm -r \*.schrott
```

all diese Dateien löschen.

Was passiert, wenn Sie bei der Eingabe des Kommandos versehentlich auf die Leertaste geraten, z.B. nach Eingabe des '*', und es nicht gleich merken ?

```
rm -r * .schrott
```

Richtig, dies ist äquivalent zu
`rm -r *` — (Ihr komplettes Dateisystem löschen)
`rm -r .schrott` — (Hier gibt's nichts mehr zu tun!)

rmdir — Verzeichnis löschen

Format: `rmdir` *directory*

Das zu löschende Verzeichnis darf keine Dateien enthalten

rsh — remote shell - Kommando auf entferntem Rechner ausführen

Auf dem als erstes Argument spezifizierten Rechner wird eine Shell gestartet, die ein beliebiges Kommando ausführt. Die (Standard-) Ein- und Ausgabe des Kommandos erfolgt am lokalen Rechner !

Beispiel: `rsh sun31 ps -ax`

SCCS, rcs — Versionsverwaltungssysteme

Bei der professionellen Entwicklung von Software und / oder sonstigen umfangreichen Dokumenten liegen Texte (aller Art) in unterschiedlichen Versionen vor. Es kann unterschiedliche Gründe für den Entwickler geben, mehr als eine Version aufzubewahren.

Regelmäßig kommt es z.B. vor, daß Kunden mit unterschiedlichen älteren Versionen eines Produkts arbeiten, während die Entwickler schon neuere Versionen erstellt haben und diese testen. Hier muß es für den Entwickler möglich sein, jede von einem Kunden benutzte Version im Bedarfsfall zu erzeugen.

Eine ganz andere, durch Versionsverwaltung handhabbare Problemstellung ist die Verwaltung verschiedener Text-*Varianten*. Varianten können durch Portierung auf unterschiedliche Hardware, Betriebssysteme, Betriebssystemversionen entstehen, durch kleinere Kunden-spezifische Anpassungen oder durch sonstige Kontext-Abhängigkeiten.

SCCS und rcs unterstützen die Versionsverwaltung auf Dateiebene.

SCCS ist bei kommerziellen UNIX-Systemen in der Regel im Lieferumfang enthalten. Als Alternative gibt es von GNU das System *rcs* als Freeware.

Beide sind für den „normalen Hausgebrauch“ sehr einfach zu bedienen !

RCS-Beispiel:

```
vi f.c    #  C-Quelltext f.c erstellen
ci f.c    #  (check-in) Versionsarchiv anlegen,
          #  1. Version von f.c in das Archiv übertragen
```

Das Archiv heißt f.c,v.

Folgende Befehle reichen zur Weiterverarbeitung des Archivs zunächst aus:

```
co f.c    #  letzte Version (read-only) aus Archiv holen
co -l f.c #  letzte Version zur Modifikation  aus Archiv holen
rlog f.c  #  Archiv-Inhalt anzeigen
```

sed — stream editor

Universelles Textfilterprogramm, besonders für nicht-interaktive Ersetzungen (auch in riesigen) Dateien geeignet.

sendmail — komplexes Mail-Zustellungs-Programm

stellt E-Mail im Auftrag von Programmen wie *mail* oder *elm* zu. Nicht für den normalen Anwender gedacht.

sh — Bourne-shell

Traditionelle Shell mit besonderen Stärken bei der Erstellung von Kommandoprozeduren.

Empfehlung: GNU-*bash* oder *ksh* verwenden (siehe gesonderte Beschreibung).

sleep — Ausführung für bestimmten Zeitraum unterbrechen

Format: `sleep` *seconds*

smail — Mail-Zustellungs-Programm

alternativ zu *sendmail* benutzbar

sort — Sortier- und Mischprogramm

Universell verwendbares zeilenorientiertes Dienstprogramm, als Sortierschlüssel sind bestimmte Wort-Positionen innerhalb der Textzeilen angebbar, verschiedene vordefinierte Vergleichsoperationen stehen zur Verfügung

source — Shellscript direkt ausführen (Shell-Kommando)

Syntax: `source` *filename*
`.` *filename*

spell, ispell — Suchen von Rechtschreibefehlern

(manchmal ist kein deutsches Dictionary vorhanden)

split — Aufteilen einer Datei in Teilstücke

Format: `split` *-n file name*

split liest die Datei *file* und erzeugt aus jeweils *n* Zeilen von file eine neue Ausgabedatei. Die Namen der Ausgabedateien sind *name*aa, *name*ab, . . . , *name*zz Defaultwert für *n* ist 1000 , Defaultwert für *name* ist x

ssh — Interaktive Sitzung am entfernten Rechner.

Beispiel: Unter der Benutzer-Identifikation *hg123456* eine Sitzung am Rechner *sun17* eröffnen:

```
ssh -l hg123456 sun17
```

strip — Entfernen von Symbol- und Relokationsinformation

startx — X-Windows starten

startx ist eine Kommandoprozedur, die einige Environment-Variablen zur Konfiguration des X-Servers definiert und dann **xinit** aufruft, um den Server und eine Reihe von X-Clients zu starten.

Die Kommandoprozedur und das xinit-Programm müssen in geeigneter Weise dafür sorgen, daß der X-Server die von ihm benötigten Dateien und Verzeichnisse findet (z.B. fonts, rgb-Datenbasis) und daß neben dem Server gleich eine Reihe von X-Clients gestartet werden (z.B. ein Window-Manager und ein xterm-Programm).

Da X-Server, Window-Manager und sonstige X-Clients nicht notwendigerweise auf dem gleichen Rechner ablaufen, da außerdem verschiedene Window-Manager (z.B. kde, icewm, fluxbox) zu Auswahl stehen, muß der Benutzer und/oder der Administrator in Form von Konfigurationsdateien festlegen, welche Programme auf welchen Rechnern in der X-Initialisierungsphase gestartet werden sollen, und wie die Benutzer-Desktop-Oberfläche aussehen soll.

Es ist die Aufgabe von *xinit*, diese Konfigurationsdateien zu lesen, die Ressourcen aller beteiligten Programme zu konfigurieren und die gewünschten Programme (auf den gewünschten Rechnern) zu starten.

stty — Terminal- und Tastaturparameter anzeigen/setzen

Die Bildschirm-abhängigen Parameter (ca. 60 Werte) können mit *stty* abweichend von den bei der Systemkonfiguration festgelegten Werten definiert werden.

su — Erweitern der Benutzerzugriffsrechte

Normalerweise für temporäre Verwendung von privilegierten Kommandos durch super-user verwendet

sync — Plattenpuffer zurückschreiben

Notwendig vor Abschalten des Rechners

tail — Anfangs- oder Endstück einer Textdatei anzeigen

talk — über Netz miteinander „schwätzen“

Syntax: `talk` *username*`@`*hostname*

tar — Archivprogramm

Umfangreiches Standardprogramm zum effizienten Lesen und Schreiben von Plattenarchiven, Magnetbändern, Disketten, usw.. Zur Datensicherung und zum Datentransport verwendbar.

Von GNU als **gtar** verfügbar mit erweiterten Funktionsumfang.

Das Programm kennt 5 Grundfunktionen:

1. c (create) — neues Archiv erzeugen
2. x (extract) — Dateien aus Archiv extrahiehen
3. t (table of contents) — Archiv-Inhalt anzeigen
4. r — Dateien am Archivende anfügen
5. u (update) — Dateien zum Archiv hinzufügen, schon vorhandene Versionen ersetzen

Das Archiv kann irgendein geeignetes Speichermedium sein (*device*-Datei als Archiv-Name) (Diskette, Plattenpartition, Band) oder eine reguläre Datei auf einem beliebigen Speichermedium.

Ein „tar-Archiv auf einer Diskette“ zu erzeugen, kann also zweierlei heißen:

1. Man kann die Diskette direkt als tar-Archiv benutzen
2. Man richtet auf der Diskette ein Dateisystem ein und speichert in diesem Dateisystem eine tar-Archiv-Datei als reguläre Datei ab.

Das Archiv kann durch die Environment-Variable $TAPE oder mit der Kommando-Option `-f` festgelegt werden.

Für *gtar* existiert auch eine info-Beschreibung (siehe Emacs-Kommando „info“).

telnet — interaktive Sitzung an einem entfernten Rechner, sicherer ist *ssh*.

telnet wird regelmäßig verwendet, um interaktive Sitzungen an Rechnern im Internet durchzuführen:

Syntax: `telnet` *Hostname*

test — Bedingungsauswertung

Format: `test` *expression*

Wertet *expression* aus und liefert 0 (TRUE) oder einen von 0 verschiedenen Wert (FALSE) als Ergebnis.

Einfache Ausdrücke (Aufzählung nicht vollständig):

-r *file*	*file* existiert und ist lesbar
-w *file*	*file* existiert und ist beschreibbar
-x *file*	*file* existiert und ist ausführbar
-f *file*	*file* existiert und ist eine reguläre Datei
-d *file*	*file* existiert und ist ein Verzeichnis
-s *file*	*file* existiert und ist nicht leer
-z *string*	Die Länge von *string* ist 0
s1 = *s2*	Die Zeichenketten *s1* und *s2* sind identisch
s1 != *s2*	Die Zeichenketten *s1* und *s2* sind nicht identisch
string	*string* ist nicht die leere Zeichenkette
n1 op n2	arithmetische Vergleichsoperationen *n1, n2* ganze Zahlen Werte für *op* : -eq (equal) -neq (not equal) -gt (greater than) -lt (less than) -le (less or equal) -ge (greater or equal)

Die oben aufgeführten einfachen Ausdrücke lassen sich mit folgenden Operationen beliebig zu komplexen Ausdrücken kombinieren:

!	Negation
-a	Konjunktion (logisches und)
-o	Disjunktion (logisches oder)
(*expression*)	Klammerung zum Gruppieren von Teilausdrücken

Vorsicht: Klammern werden von der shell als Sonderzeichen interpretiert, sie müssen also ihrer Sonderbedeutung enthoben werden

time — Ausführungszeit eines Kommandos messen

Format: `time` *command*

Für Laufzeitmessungen von Programmen geeignet Gibt sowohl einen „real-time-“ als auch einen „CPU-time-Wert“ auf die Standard-Error-Datei aus.

touch — letzten Modifikationszeitpunkt aktualisieren

Format: `touch` *file* ...

Wird oft verwendet, um reine „Zeitstempel-Dateien“ zu verwalten, leere Dateien also, deren Informationsgehalt gerade der letzte Modifikationszeitpunkt ist (z.B. nach einer Datensicherung Datum und Uhrzeit für inkrementelle Sicherung merken: `touch backup.timestamp`). Auch um in Verbindung mit *make* eine Aktion zu erzwingen.

Man kann damit auch beliebige Zeitpunkte im Inode einer Datei neu setzen.

tr — Zeichenkettenkonvertierung

Zeichenorientiertes Textfilterprogramm, beispielsweise zur Konversion von Klein- in Großbuchstaben oder zum Ersetzen von Folgen von Leerzeichen durch ein einziges Leerzeichen geeignet

trap — Signalbehandlung definieren (Shell-Kommando)

troff — Textformatierungsprogramm

Komplexes Programm zum Textsetzen für graphikfähige Ausgabegeräte. (im Gegensatz dazu ist nroff für Typenraddrucker konzipiert) Makromechanismus, Proportionalschrift, anwenderdefinierte Zeichnsätze. (Dazu gehören eine Reihe von Hilfsprogrammen wie *col, table* usw.)

Alternativ: **groff** von GNU

tty — Terminalname ausgeben

umask — Zugriffsrechtsmaske für neue Dateien definieren

Setzt die Zugriffsrechte beim Erzeugen einer Datei spätere Änderung durch **chmod** möglich

uname — Systeminformation anzeigen

Zeigt Betriebssystemversion, Rechner usw. an (Option `-a`).

vi — Bildschirmorientierter Editor

Programmierbar wie ex, aber besser zum interaktiven Arbeiten geeignet

wait — Warten auf Beendigung aller erzeugten Hintergrundprozesse

wc — Wörter, Zeilen, Zeichen in einer Textdatei zählen

whatis — zeigt Information zu einem Stichwort an

Im Gegensatz zu *apropos* muß das Stichwort vollständig angegeben werden.

who, finger — Liste der aktiven Benutzer ausgeben

Format: `who` *options* (alle Benutzer)
`who am I` (eigene Benutzeridentifikation)
`finger` *options*

wish, wishx — „Windowing Shell“

Shell mit Operationen zur Erzeugung und Bedienung einer graphischen Benutzeroberfläche. Gehört zum tcl/tk-System

X — siehe **startx**

xclipboard — Clipboard-Programm für X-Windows

xdvi — TEX-Previewer für X-Windows

xemacs — Editor

Die derzeit im FB MNI der FH Gießen als Programmierumgebung konfigurierte Emacs-Version. Nur für X-Windows !

siehe separate Beschreibung der Programmierumgebung

xev — Event-Anzeige für X

xfig — Zeichenprogramm für X-Windows

Vektordarstellung der Objekte, Verbinden von Objekten zu Gruppen, mehrere Export-Formate, z.B. Postscript

xinit — siehe **startx**

xmodmap — Tastatur-Anpassung für X

xpr — X-Window-Bitmap drucken

(siehe auch *xwd* und *pbm-Tools*)

xrdb — Resource data base manager

Verwaltet Ressourcen der X-Clients, erlaubt dynamische Änderung der Ressource-Datenbasis mit den *load*- und *merge*-Befehlen.

xv — Bilder anschauen, verarbeiten und konvertieren

xv verarbeitet diverse Formate, z.B. Postscript, X-Window-Dump, GIF, TIFF, JPEG.

xwd — X-Window-Dump

Erzeugt Bitmap-Datei-Repräsentationen von X-Fenstern. Die Dateien können mit **xpr** gedruckt oder mit den pbm-Tools weiterverarbeitet werden.

xman — siehe **man**

xterm — Terminal-Emulationsprogramm

Xterm erlaubt es, innerhalb der Graphik-Oberfläche X-Windows beliebige Text-orientierte Programme ablaufen zu lassen, z.B. eine Shell. Es emuliert dabei ein VT100- oder ein Tektronix-Terminal.

Die Kombination von CTRL- und einer der Maustasten erlaubt Menügesteuerte Einstellungen der Emulation.

Wichtige Einstellungen sind die Zeichengröße und die Größe des Puffers, in dem die Ein-/Ausgabe gespeichert und mittels Rollbalken wiederverwendet werden kann.

Die Einstellungen können wahlweise per Menü, durch Aufruf-Optionen oder durch entsprechende `.Xdefaults`-Spezifikationen vorgenommen werden.

Beispiel: Aufruf mit Rollbalken und 10x20-Zeichensatz, 300 Zeilen merken:

```
xterm -sb -fn 10x20 -sl 300
```

Alternativ zum obigen Beispiel: Rollbalken, Zeichensatz und Zeilenpuffergröße werden als Ressourcen in `.Xdefaults` festgelegt:

```
xterm*ScrollBar: True
xterm*Font:      10x20
xterm*SaveLines: 300
```

Beim Aufruf kann das zu startende Text-orientierte Programm (Option -e, voller Pfad !) angegeben werden, ansonsten wird eine Shell gestartet.

Auch die Protokollierung einer Sitzung in einer Log-Datei ist möglich.

yacc — yet another compiler-compiler

„Berühmtes“ und häufig verwendetes Programm zur automatischen Generierung von Parsern (oder Parser-Tabellen zur Einbindung in Anwenderprogramme). (Parser = Programm zur Strukturanalyse von Texten, insbesondere zur syntaktischen Analyse von Programmiersprachen) Aber auch für eine Vielzahl von Non-Standard-Anwendungen (Anwendersprachen, Textverarbeitung, usw.) geeignet. Zusammenarbeit mit lex sehr einfach. Als Eingabe dient eine kontextfreie Grammatik für die zu analysierende Sprache. Verwendet wird ein mächtiger LR-Parsing-Algorithmus.

yp. . . , nis. . . — NIS-Kommandos

Das Netzwerk-Informationssystem NIS hieß früher „yellow pages“, dann NIS, jetzt heißt es (bei Solaris) NIS+. Kommandos zur Steuerung von NIS, das immer noch sehr verbreitet ist beginnen alle mit dem Präfix *yp*. *ypwhich* ermittelt beispielsweise den aktuellen NIS-Server.

Kommandos zur Steuerung von NIS+, beginnen alle mit dem Präfix *nis*. *nis-cat hosts.org_dir* zeigt beispielsweise die Rechnerdatenbasis an, *nisgrep hg52 passwd.org_dir* durchsucht die Benutzerdatenbasis nach dem Eintrag für hg52.

zcat — *cat* mit Dekompression, s.a. *gzip*, *gunzip*.