UNIX

Das Betriebssystem UNIX zählt zu den am weitesten verbreiteten Betriebssystemen. Es wird in Abwandlungen auf fast allen Workstations eingesetzt.
Zur schnellen und weiten Verbreitung von UNIX trug auch der Umstand bei, daß die Quellprogramme praktisch zum Preis der Kopier- und Versandkosten an Universitäten abgegeben wurden. Diese sehr rasche, nicht durch kommerzielle Vertriebe organisierte Verbreitung hatte allerdings die nachhaltige Folge, daß inzwischen unzählige Rechnerplattform-spezifische Varianten dieses Systems existieren.

Die Benutzung eines Unix-Systems

Der Zugang

Um mit einem Unix-System arbeiten zu können, muß sich der Benutzer bei diesem anmelden. Zuerst meldet sich das System mit dem sogenannten Login-Prompt:
Dies kann wenn man lokal am Rechner arbeitet eine graphische Oberfläche mit Auflistung aller eingerichteter User sein, oder wenn man die Verbindung über ein Terminal aufbaut die Eingabeaufforderung

login:

Nach der Angabe bzw Auswahl des Benutzernamens (mit <CR> abschließen) wird die Eingabe eines Paßwortes verlangt:

password:

Nach erfolgreichem Login erscheint der Unix-Prompt (Terminal) bzw die User-spezifische Oberfläche. Die graphische Benutzerschnittstelle (graphical user interface GUI) soll die Bedienung dem Benutzer einfacher machen. Als Quasi-Standard hat sich hier das X-Window System durchgesetzt. Es erlaubt eine Bearbeitung verschiedener Aufgaben in getrennten Bildschirmbereichen, den Fenstern. Dabei kann es sich sowohl um echte graphische Ausgabe handeln, als auch um einfache Terminal-Windows, in denen per UNIX-Kommando gearbeitet wird. Dies kann einmal (nur) quasi-parallel (auf einem Prozessor) erfolgen, aber auch echt parallel, indem die Windows auf verschiedenen Rechnern geöffnet werden. Damit ist X-Window insbesondere geeignet, die Möglichkeit, welche die Vernetzung und die verteilte Datenverarbeitung bieten, effizient zu nutzen.

Die Dateizugriffsrechte eines Benutzers

Mit Hilfe von Benutzerklassen werden unter UNIX die Rechte des einzelnen Benutzers, insbesondere die Zugriffsrechte auf Dateien, gesteuert.

Für jeden Benutzer ist die UNIX-Welt in drei dieser Klassen eingeteilt:

u = login user
g = group
o = others

Jeder Benutzer ist über eine UID (User IDentification-number), die eindeutig vom Benutzernamen abhängt, identifiziert. Außer dieser UID besitzt jeder Benutzer jedoch noch eine GID (Group-IDentification-number), mit der seine Zugehörigkeit zu einer Gruppe festgelegt ist. Mit anderen Mitgliedern dieser Gruppe kann sich der Benutzer gewisse Rechte teilen. Alle anderen Benutzer, die nicht derselben Gruppe angehören, gelten als der Rest der UNIX-Welt, die *others*.

Dateienstruktur

Die Unix Dateistruktur ist hierarchisch oder umgekehrt baumartig.
Der Ausgangspunkt eines solchen Baumes ist die Wurzel (root), die unter Unix mit / bezeichnet wird. Die Wurzel selbst ist ein directory, in dem Verweise auf weitere Dateien stehen, die wiederrum selbst directories sein können. Auf diese Weise ensteht die Baumstruktur.

Das aktuelle Directory.

Während der gesamten Sitzung ist dem Benutzer zu jedem Zeitpunkt genau ein directory zugeordnet, auf das sich seine Kommandos beziehen. Dieses directory nennt man *working directory*. Der Benutzer kann sich im gesamten Dateibaum bewegen
(dh das aktuelle directory wechseln),
sofern er die Rechte dazu besitzt.

Um sich im Dateibaum zu bewegen, benutzt man das Kommando

cd (change directory)

Zur Information wo man sich zur Zeit befindet (wie heißt das aktuelle directory), benutzt man das Kommando

pwd (print working directory)

Bei der Eingabe des cd-Befehles ohne Angabe eines Argumentes wird in das sogenannte HOME-directory gewechselt; dies ist dasjenige working-directory, das nach dem login als erstes directory eingestellt wird.
Die Eingabe von zwei Punkten als Argument des cd-befehles bewirkt stets den Wechsel zum parent-directory des aktuellen Katalogs.

zB: das HOME-directory des Benutzers meier ist das directory :
/usr/people/meier

$ pwd
/usr/lib
$ cd ..
$ pwd
/usr
$ cd
$ pwd
/usr/people/meier

Dateizugriff

Der Zugriff auf eine Datei bzw. das Ansprechen erfolgt prinzipiell über den Dateinamen. Die maximale Länge von Dateinamen ist nicht nur vom jeweiligen UNIX, sondern auf vom installierten File System abhängig. Prinzipiell sind alle Zeichen erlaubt, aus praktischen Gründen sollte man sich jedoch auf Buchstaben (Groß- und Kleinbuchstaben werden unterschieden!) und Zahlen beschränken.

Man kann eine Datei nur dann mit ihrem Namen ansprechen, wenn sich diese im aktuellen directory befindet. Will man eine Datei außerhalb des aktuellen directories ansprechen, so muß man dazu einen Zugriffspfad angeben. Der Zugriffspfad (PATH) gibt an, wie eine Datei, ausgehend von /, erreicht werden kann.
Wenn sich im directory */usr/people/meier* eine Datei *texte.1* befindet, ist der Zugriffspfad */usr/people/meier/texte.1* ; dieser wird auch der absolute Pfadname genannt.
Des weiteren kann man eine Datei auch über den relativen Pfadnamen ansprechen, das ist der Zugriffspfad, der, ausgehend vom aktuellen directory den Weg im Dateibaum zur Zieldatei beschreibt.
Es können Pfade auch in den Umgebungsvariablen für den Benutzer definiert werden, sodaß der Zugriff auf Dateien in solchen definierten Pfaden für diesen Benutzer von jeder beliebigen Stelle im Dateibaum ohne Pfadangabe möglich ist.

Dateiattribute

Dateien haben eine Reihe von Attributen (= systemseitig gespeicherte Eigenschaften), die sich der Benutzer auf Anforderung (durch das Kommando *ls -l*) anzeigen lassen kann.
zB:
ls -l

total 1137
-rw-r--r-- 1 wave42 image 4929 Apr 14 1997 Fassade-1
-rw-rw-rw- 1 wave42 image 44235 Jan 19 1997 ADI-BKS.rgb
drwxr-xr-x 3 wave42 image 512 Dec 10 10:31 ARCH/
drwxr-xr-x 3 root sys 512 Jan 4 1997 AdobeIllustrator5.5/
drwxrwxrwx 10 wave42 100 512 Feb 25 1997 AdobePhotoshop3/

In der ersten Zeile wird angegeben, wieviel Blöcke die Einträge in diesem directory belegen (in diesem Fall 1137). Darauf folgt eine Liste aller Dateien, wobei pro Zeile eine Information ausgegeben wird.

Im ersten Informationsfeld (die ersten 10 Spalten) stehen das Dateiart-Bit und die 9 Protection bits (protection mode). Diese bedeuten im einzelnen:

> Im ersten bit wird die Dateiart angezeigt:

d Datei ist ein directory
l Datei ist ein symbolischer Linkeintrag auf eine andere Datei
- normale Datei
b special file blockorientiert
c special file charakterorientiert
p piped name
die nachfolgenden 9 bits zeigen die Zugriffsberechtigungen an.

> (in der folgenden Spalte steht die Anzahl der Hardlinks auf die Datei.)
> im dritten und vierten Feld stehen der Name des Eigentümers und der zugehörigen Gruppe (es können unter bestimmten Umständen auch die UID bzw GID sein)
> im fünften Informationsfeld wird die Dateilänge in Bytes angegeben.
> in den nächsten drei Feldern werden Datum und Uhrzeit der letzten Dateiänderung angezeigt.
> in der letzten Spalte steht der Name der Datei.

Der Benutzer kann Dateiattribute ändern, sofern er die Rechte dazu besitzt.

Die Bedeutung der Zugriffsrechte

Die Zugriffsrechte auf normale Dateien

Für jede Benutzerklasse (*u = user, g = group, o = other*) gibt es jeweils drei verschiedene Zugriffsrechte:

Lesen (r = read)
Schreiben (w = write)
Ausführen (x = execute)

Insgesamt sind also 9 Zugriffsrechte (protection bits) pro Datei gesetzt. Diese Zugriffsrechte gibt es für alle Dateien, also auch für directories.

An der Ausgabe des Kommandos *ls -l* (listing long) kann man also die Zugriffsrechte ablesen: Die ersten 3 bits zeigen die Rechte des Eigentümers, die mittleren die Rechte der Gruppenmitglieder und die letzten drei bits die Rechte aller anderen Benutzer an.

zB:

-rwxr-x--x

bedeutet: es handelt sich um eine normale datei (-) der Eigentümer darf sie lesen, auf ihr schreiben und sie ausführen; die Gruppenmitglieder dürfen sie lesen und ausführen, jedoch nicht auf ihr schreiben und andere Benutzer dürfen nur ausführend auf sie zugreifen.

Das Setzen der Zugriffsrechte

Die Zugriffsrechte verändern.

Mit dem Kommando chmod (change mode) kann der Eigentümer einer Datei die Zugriffsrechte dieser Datei ändern.

Für dieses Kommando gibt es zwei syntaktisch verschiedene Formen (die aber in Wirkung gleich sind)

chmod permissionlist filename
chmod ooo filename

Im ersten Fall gibt der Benutzer eine permissionlist und den Namen der Datei an, deren protection mode geändert werden soll, und zwar durch gezieltes Hinzufügen oder Wegnehmen von Zugriffsrechten einzelner Benutzerklassen.
Die permissionlist hat dabei folgende Form : Benutzerklasse +/- rechte. Als Benutzerklasse ist außer den drei bekannten (*u,g und o*) noch eine Sammelklasse *a* (all) möglich. Als Rechte sind die bekannten *r,w und x*, auch kombiniert möglich.

zB:

chmod g-r text.1

entzieht der Gruppe die Leserechte auf *text.1*

In der zweiten Form des Kommandos (*chmod 000 filename*) wird der protection mode als ganzer neu gesetzt. Dabei steht *000* für eine Oktalzahl, die - als Binärzahl gelesen - genau dort die Einsen hat, wo das entsprechende Zugriffsrecht gesetzt sein soll.

zB

chmod 775 text.1

setzt die Rechte auf *rwxrwxr-x*

Das Ändern der Besitzrechte

Das Eigentumsrecht kann mit dem Kommando *chown* (change owner) geändert werden.

chown benutzer file

Die Gruppenzugehörigkeit mit dem Befehl *chgrp* (change group).

Prozesse

Ein Prozeß ist (im wesentlichen) ein laufendes Programm. Dabei können innerhalb eines Prozesses nacheinander auch mehrere Programme ablaufen. Ein einzelnes Programm kann wiederum mehrere gleichzeitig laufende „child-prozesse“ erzeugen und kontrollieren.
Jedem Prozeß ist eine eindeutige Nummer, die PID (Process IDentification number), zugeordnet.

Der login-Prozeß

Ein laufendes Unix-System besteht (wie das Dateisystem) aus einer baumartigen, hierarchischen Struktur von Prozessen. Die Wurzel der Prozessstruktur, das ist der Prozeß mit der *PID 0*, wird beim Start des Systems erzeugt. Dieser Prozeß startet mehrere Systemprozesse, darunter den Prozeß init mit der *PID 1*. Der Prozeß init wiederrum ist der Urvater aller Benutzer- und der meisten Dämonprozesse (Dämonprozesse sind vom System abgesetzte Hintergrundprozesse, die in zyklischen Zeitabständen ausgeführt werden und diverse Dienste zur Verfügung stellen).

Nach erfolgreicher Anmeldung über den login-Prozeß wird für jeden Benutzer ein neuer eigener Prozeß, nämlich die systemseitig definierte *login-shell* gestartet. Mit dem *login* wird der Benutzer Eigentümer genau dieses Prozesses und damit auch aller von diesem Prozeß erzeugten child-Prozesse. Verläßt der Benutzer die login-shell, so ist die Sitzung beendet und alle davon abhängigen Prozesse werden abgebrochen (ausser es wurden Prozesse als im „Hintergrund“ laufend gestartet).
Wenn man ein Kommando absetzt, wartet normalerweise der login-prozeß, bis der Kommandoprozeß ausgeführt ist, bevor er wieder den login-prompt ausgibt (d.h. ein neues Kommando abgesetzt werden kann).
Es ist allerdings möglich, ein Kommando in den Hintergrund abzusetzen, so daß man sofort mit weiteren Kommandos fortfahren kann. Dies erreicht man mit dem Zeichen *&* am Ende der Kommandozeile.

render -P TEST -l 720 576 -a -S copml SCENE1&

Einen Überblick über laufende Prozesse kann mit dem Kommando *ps* (process status) erhalten.

```
farchsg1 wave42 % ps
PID TTY      TIME CMD
24517 ttyq11  0:01 csh
25530 ttyq11  0:00 ps
```

Zwei Prozesse werden angezeigt, nämlich die C-shell mit der *PID 24517* und der dem Kommando *ps* entsprechende Prozeß mit der *PID 25530*.

Will man sich alle im System laufenden Prozesse ansehen, sind die Optionen *-a*, *-e* und *-f* zu verwenden.

Mit dem Kommando *who* kann man sich außerdem anzeigen lassen, welche Benutzer momentan im System eingeloggt sind.

Prozesse abbrechen

Einen im Vordergrund laufenden Prozeß kann man mit

Ctrl C

abbrechen. Einen im Hintergrund laufenden Prozeß kann man normalerweise mit

kill PID

abbrechen (die PID ist mit *ps* zu erfahren). Dieses Kommando schickt dem angegebenen Prozeß ein Signal, das standardmäßig als ein User-break interpretiert wird. Einige Kommandos sind aber so geschrieben, daß sie dieses Signal abfangen und anders bearbeiten. Ein solches Kommando muß mit

kill -9 PID

abgebrochen werden. Kein Prozeß der zum Benutzer gehört, kann diese Abbruchsignal ignorieren.

Die Shells

Während einer Sitzung am Rechner kommuniziert der Benutzer normalerweise nicht direkt mit dem Betriebssystem, sondern mit einem Programm, welches seine Kommandos liest, analysiert und auszuführeren versucht. Ein solches Programm wird deshalb als Kommandointerpreter bezeichnet. Unter UNIX ist es *SHELL* genannt worden, weil es sich wie eine Schale um den Kern des Betriebssystems legt.
Auf den unterschiedlichen Unix-Implementierungen existieren eine Reihe verschiedenartiger Shell-programme, die verschiedene Funktionalitäten aufweisen.
Wie : *bourne-shell, C-shell, Korn-Shell,* usw.

Übersicht über die wichtigsten Kommandos:

Online manual

Auf jedem Unix-System sind normalerweise Online-manuals mit englischen Kommandobeschreibungen vorhanden.
Mit Hilfe des Kommandos

man kommando

kann man sich eine Beschreibung des Kommandos ausgeben lassen.

Kurze Kommandoübersicht

Aufbau der Kommando-Beschreibungen:

Kommando [optionen] parameter

Kommando : bei der Eingabe wird Groß- und Kleinschreibung unterschieden

optionen: eckige Klammern bezeichnen Optionen, die nach dem Kommando eingegeben werden können, aber nicht müssen.

Parameter: Handelt es sich bei Parametern um Dateien, so dürfen deren Namen Metazeichen/Wildcards enthalten.

cat (concatenate)

cat [-estuv] [datei]

cat liest die Datei(en) und schreibt sie auf die Standardausgabe (Bildschirm)
Fehlt der Parameter Datei, so liest *cat* von der Standardeingabe (Tastatur). Ist eine Ausgabedatei durch Umlenken der Standardausgabe (>) angegeben, werden die zuvor angegebenen Dateien hintereinander in die Ausgabedatei kopiert.

Beispiel: *cat gedicht*
gibt den Inhalt der Datei *gedicht* auf den Bildschirm aus.

cd (change directory)

cd [directory]

Wechsel des aktuellen directory.

Parameter:
.. bezeichnet das dem gerade aktuellen directory übergeordnete
/ bezeichnet das root directory
ohne den parameter directory wird nach $HOME gewechselt.

chmod (change mode)

chmod modus datei...

Ändern der Dateizugriffsrechte für die im Parameter *datei*... angegebene(n) Datei(en).
Parameter:
u user
g group
o others
a all - entspricht ugo
+ Zugriffsrechte erteilen
- Zugriffsrechte entziehen

= alle Rechte entziehen, außer den angegebenen
r read Leserechte
w write Schreibrechte
x execute Ausführungserlaubnis

cp (copy)

cp datei1 [datei2] directory

kopiert Datei(en) - ist der letzte Parameter ein directory, so werden die davor stehenden Dateien unter gleichem Namen in dieses directory kopiert.

df (disk free)

zeigt den gesamten, belegten und freien Speicherplatz des Filesystems an.

du (disk usage)

du [-k] path

listet den belegten disk space auf, mit *-k* in kilobyte

find

find Pfadname [expression]

durchsucht die in Pfadname angegebenen directories sowie deren Unter-directories nach Dateien, die eine durch expression gegebene Bedingung erfüllen.
expression:
-name Name
-print soll der Name der gefundenen Datei ausgegeben werden so ist die expression *-print* anzugeben
Beispiel: *find /usr/people/meier -name ber*.txt -print*
durchsucht alle directories von /usr/people/meier ausgehend abwärts nach dateien, die mit *ber* beginnen und *.txt* enden und gibt die vollständigen Namen mit Pfaden als Suchergebnis am Bildschirm aus.
kill

kill [-signal] PID...

Abbrechen laufender Prozesse.
signal: nummer zwischen 1 und 15
zB: *0* als PID steht für alle Prozesse der laufenden Sitzung.
Beispiel: *kill -9 0* bricht alle laufenden Prozesse des Benutzers ab.

ln (link)

ln [-fs] datei1 datei2

Ein und diesselbe physikalische Datei kann in einem Dateibaum auch mehrere Namen besitzen, die auch auf verschiedenen Ästen liegen können. Diese Datei ist dann unter all ihren verschiedenen Namen ansprechbar. Durch einen Linkeintrag wird keine neue Datei erzeugt, sondern nur ein Verweis auf diese Datei.

ls (list)

ls [-latr..] [directory]

Dateinamen anzeigen

Optionen:
-*a* (all) auch verdeckte Dateien, die mit einem Punkt beginnen werden angezeigt
-*l* (long format) ausführliches Format
-*t* (time) zeitlich geordnet
und viele mehr

man (manual)

man Kommando

aufrufen der On-line Hilfe der Kommandobeschreibung

mkdir (make directory)

legt ein neues directory an, wenn der Benutzer im übergeordneten directory Schreiberlaubnis hat.

more

more filename

gibt den Inhalt einer Datei am Bildschirm aus - aber immer nur den Teil, der im window Platz hat - mit enter kann um eine Zeile weitergeschaltet werden , mit space bar um einen Fensterinhalt.

mv (move)

mv datei neuername

Dateien umbenennen, oder verschieben - dann muß neuername der Name eines directories sein.

passwd (password)

passwd name

ändert das password des Benutzers

ps (process status)

ps [-ef..]

liefert Informationen über laufende Prozesse

-e zu allen Prozessen wird Information ausgegeben
-f vollständige Informationsliste

pwd (print working directory)

zeigt das working directory an

rm (remove)

rm [-ir..] Datei..

löscht Datei(en)

Optionen:

-i (interaktiv) löschen mit Abfrage
-r (rekursiv) löscht rekursiv den gesamten Inhalt des angegebenen directories einschließlich aller sub-directories.
ACHTUNG: gelöschte Dateien können nicht wiederhergestellt werden.

rmdir (remove directory)

rmdir directory..

löschen von leeren directories

tar (tape archive)

tar [c x v f] name

faßt Dateien oder directories zu einer Datei oder auf externen Datenträger zusammen

name: gibt an, welche Dateien oder directories zusammengefaßt oder wieder herausgeschrieben werden sollen

Optionen:
c Anlegen eines tar-Archivs
x extrahieren von Dateien aus einem tar-Archiv
v verbose-mode
f dateiname - mit der Option *f* kann eine Datei angegeben werden, auf die das Archiv geschrieben oder von der gelesen werden soll.

Beispiel:

*tar cf alles.tar *.pic*

alle Dateien mit der Endung *.pic* im aktuellen directory werden in ein archivfile *alles.tar* zusammengefaßt.

!!

wiederholt den zuletzt in der shell eingegebenen Befehl.

!a

wiederholt den zuletzt in der shell eingegebenen Befehl der zB. mit *a* beginnt.

Diese Kurzübersicht besteht aus Auszügen von „UNIX - eine Einführung“, des RRZN/Universität Hannover, das bei Interesse im Sekretariat des ZID erworben werden kann.