

LUG Ahaus



<http://lugah.de>

Linux- Basics- Workshop

Linux Basics Workshop

Linux User Group Ahaus

Nov. 03

Rainer Ostendorf

rainer@lugah.de

LUG Ahaus



<http://lugah.de>

Linux- Basics- Workshop

Was ist Linux ?

Hinter dem Begriff Linux verbirgt sich eigentlich nur der Kernel, also die Basis, eines unixähnlichen Betriebssystems (Derivat). Meistens wird der Begriff jedoch ausgedehnt und das komplette Betriebssystem damit bezeichnet.

Von der Struktur und den Konzepten ist Linux ein erstklassiges Unix, welches sich von den meisten anderen Unixderivaten dadurch abhebt, daß es frei verfügbar (Open Source) ist, und jeder den kompletten Quelltext bekommen, einsehen und verändern kann.

LUG Ahaus



<http://lugah.de>

Linux- Basics- Workshop

Was ist eine Distribution

Während Linux selbst lediglich ein Betriebssystemkern ist, macht eine Distribution ein komplettes Betriebssystem daraus. Unter einer Distribution versteht man das Zusammenfügen des Betriebssystemkernes und vorhandenen Distributionen. Neben den für ein Betriebssystem nötigen Programmen, z.B. zur Dateisystem- und Benutzerverwaltung, werden hier auch Anwendersoftware und weitere Programme hinzugefügt. Man kann demnach sagen, eine Distribution macht aus dem Kernel das runde System.



Linux- Basics- Workshop

Die Konzepte hinter Linux

Multiuser- / Multitaskingkonzept

Linux ist ein echtes Multiuser- und Multitasking Betriebssystem. Dies bedeutet, daß mehrere Benutzer zum gleichen Zeitpunkt mehrere Programme ausführen kann. Zu diesem Zweck hat das Betriebssystem eine im Kernel verankerte Benutzerverwaltung, welche dafür sorgt, daß die Prozesse der Benutzer auch mit deren Rechten ablaufen.

Die Fähigkeit, mehrere Programme (oder Prozesse) gleichzeitig verarbeiten zu können bezeichnet man als Multitasking oder Multiprocessing. Dabei ordnet der Betriebssystemkernel jedem aktiven Prozeß einen bestimmten Zeitabschnitt zu. Innerhalb dieser Zeit werden vom Kernel dann die Anweisungen dieses einen Programmes abgearbeitet. Dieses Verfahren wird auch als Zeitscheibenverfahren bezeichnet und ist eine bewährte und erprobte Methode um dieses Ziel zu erreichen.

Da sich Multiuser und -tasking gut vertragen, ist es natürlich auch möglich, daß jeder einzelne Benutzer mehrere Programme gleichzeitig ablaufen läßt.



Linux- Basics- Workshop

Technischer Aufbau eines Linuxsystemes

Ein Linuxsystem besteht aus mehreren Teilen. Die Grundlage des Systemes ist der Betriebssystemkern. Um diesen herum sind weitere Komponenten angebracht. Dies ist zunächst der sogenannte Kommandointerpreter (auch Shell genannt). Die Shell ist die textorientierte Schnittstelle zwischen Betriebssystemkern und Benutzer. Sie nimmt Befehle und Anweisungen des Benutzers entgegen, übersetzt sie in für den Kern verständliche Anweisungen und leitet diese dann an den Kernel weiter.

Um dem Benutzer jedoch mehr Arbeitskomfort zu geben, wurden auch grafische Benutzeroberflächen (GUI, Graphical User Interface) entwickelt, welche dem Benutzer ein intuitiveres arbeiten ermöglichen.



Linux- Basics- Workshop

Benutzer- und Gruppenkonzept

Sobald mehrere Benutzer auf einem System arbeiten können, muß das Betriebssystem ein Konzept zur Benutzerverwaltung besitzen. Idealerweise gibt es ein entsprechendes Gruppenkonzept, damit mehrere Benutzer, die gemeinsam an einem Projekt arbeiten, auch entsprechend benötigte Ressourcen teilen können.

Das Benutzer- und Gruppenkonzept hinter Linux ist das Standardkonzept jedes Unixderivates. Dabei kann jeder Benutzer Mitglied in mehreren Gruppen sein. Darüberhinaus ist jeder Benutzer einer Hauptgruppe zugeordnet. Dieses Konzept ist natürlich Grundlage des Rechtesystemes, welches das Betriebssystem bietet.

Zur Laufzeit des Systemes können mehrere Benutzer gleichzeitig und auf mehrfach am System angemeldet sein. Die Anmeldung kann dabei direkt an dem entsprechenden Rechner erfolgen, als auch remote, also von einem anderen System aus.

Das System selbst ordnet jeder Benutzerkennung eine eindeutige Nummer, die sogenannte User- bzw. Gruppenid, zu. Auf systemebene wird ausschließlich mit diesen Nummern gearbeitet, welche um es dem Benutzer einfacher zu machen nach außen hin in die alphanumerische Kennung übersetzt wird.

Eine besondere Rolle spielt dabei die User - ID Null, welche immer der Kennung des Systemverwalters (root) zugeordnet ist.



Linux- Basics- Workshop

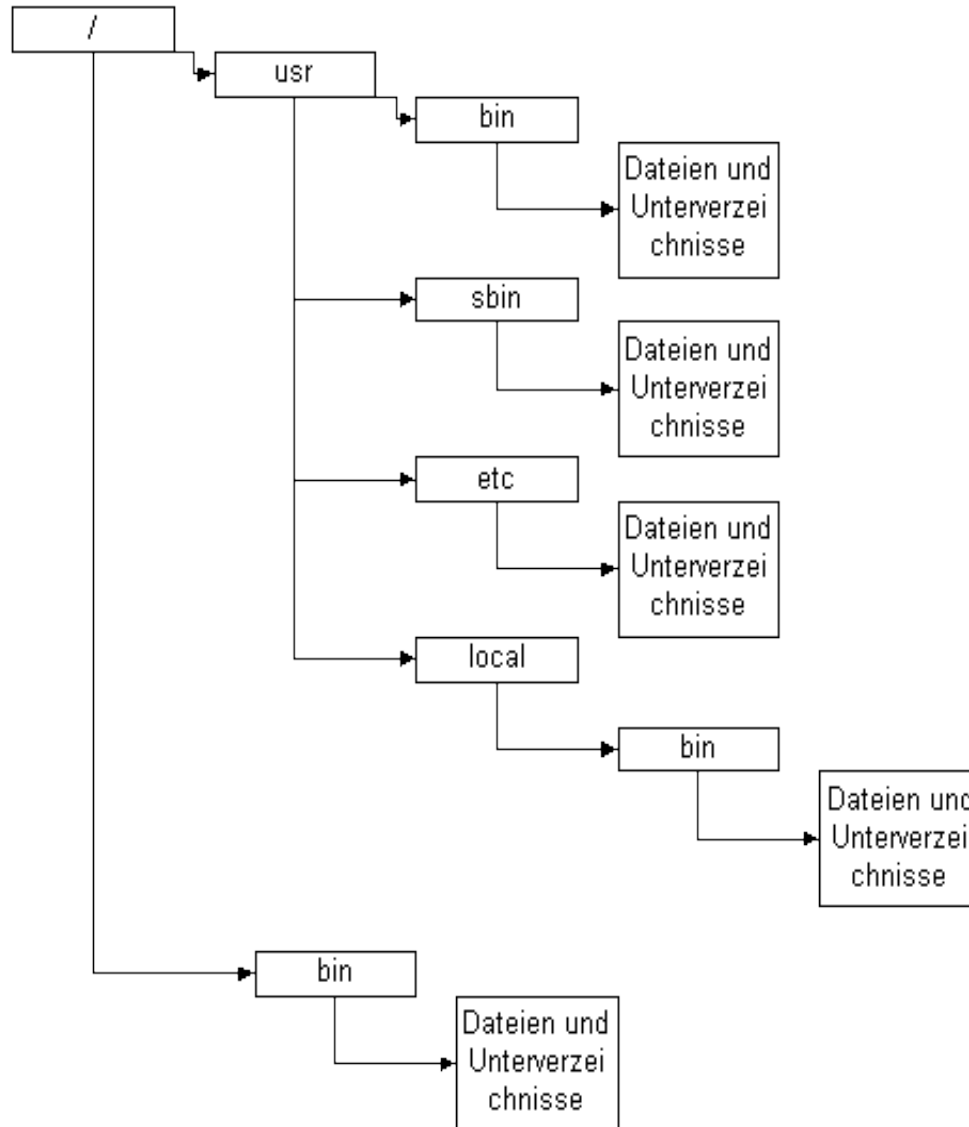
Dateisystemkonzept

Das Unixdateisystem ist sehr streng und eindeutig organisiert. Entgegen anderer Filesystemkonzepte (wie z.B. unter DOS / Windows) kennt Linux keine verschiedenen Laufwerke, sondern eine Wurzel, von der aus alle anderen Verzeichnisse und Speichergeräte angesprochen bzw. eingebunden werden können. Als Trennzeichen zwischen den einzelnen Verzeichnissen wird der einfache Schrägstrich (' / ' oder Slash) verwendet. Dieser signalisiert auch die oberste Ebene des Verzeichnisbaumes. Diese Ebene wird auch Root- oder Wurzelverzeichnis genannt.

Unterhalb des Rootverzeichnisses können beliebig viele Unterverzeichnisse existieren. Festplattenpartitionen und andere Massenspeicher (z.B. CD Rom Laufwerke) werden direkt als Unterverzeichnis eingebunden (gemounted). Somit ist das komplette Dateisystem streng linear organisiert.



Linux- Basics- Workshop



LUG Ahaus



<http://lugah.de>

Linux- Basics- Workshop

Datei- und Verzeichnisnamen

Datei und Verzeichnisnamen unter Linux unterliegen recht wenigen Beschränkungen. Eine Datei bzw. Verzeichnisname darf maximal 255 Zeichen umfassen und kann aus einer beliebigen Kombination aus Buchstaben und Ziffern, sowie den meisten Sonderzeichen bestehen. Man sollte jedoch auf Sonderzeichen und Leerzeichen möglichst verzichten.



Linux- Basics- Workshop

Besondere Verzeichnisnamen

Unter Linux gibt es drei besondere Verzeichnisnamen, welche von System angelegt und benötigt werden. So existieren in jedem Verzeichnis (egal auf welcher Ebene) die mit einem, bzw. zwei Punkten benannt sind ('.' und '..'). Dabei bezeichnet das Punkt - Verzeichnis das aktuelle Verzeichnis selbst, während das mit zwei Punkten benannte Verzeichnis das jeweilige Elternverzeichnis, also das nächst höhere Verzeichnis adressiert.

Diese beiden Verzeichnisse sind nicht nur zur internen Verwaltung wichtig, sondern können auch vom Benutzer zur indirekten Verzeichnisadressierung benutzt werden.

So bezeichnet der Ausdruck './test.pl' die Datei test.pl im aktuellen Verzeichnis, während der Ausdruck '../test.pl' die Datei test.pl im nächst höheren Verzeichnis anspricht.

Das dritte 'Sonderverzeichnis' ist das Verzeichnis 'lost+found'. Dieses Verzeichnis liegt jeweils auf der obersten Verzeichnisebene eines Datenträgers (sprich Festplattenpartition) und wird vom Betriebssystem intern benötigt.



Linux- Basics- Workshop

Wildcards oder Jokerzeichen

Wie in anderen Betriebssystemen auch, so kennt auch Linux Wildcards oder Jokerzeichen im Dateinamensystem. Diese Zeichen ersetzen bei Abfragen mit Dateinamen bestimmte Teile des Datei- oder Verzeichnisnamens.

Linux kennt zwei Jokerzeichen:

* (Sternchen):

Der Stern ersetzt beliebig viele Zeichen im Dateinamen.

? (Fragezeichen):

Das Fragezeichen ersetzt genau ein Zeichen im Dateinamen.



Linux- Basics- Workshop

Rechtekonzept

Überall dort, wo mehrere Benutzer aufeinander treffen, müssen klare Regeln zu Befugnissen und Verhalten existieren. Um dies unter Linux zu wahren, besitzt Linux ein Rechtekonzept, welches auf Dateisystemebene die Zugriffe der einzelnen Benutzer und Gruppen regelt.

Zu diesem Zweck hat jede Datei und jedes Verzeichnis einen Besitzer (üblicherweise der Benutzer, welcher die Datei angelegt hat) und eine Gruppe (in der die Benutzer enthalten sind, welche Zugriff auf die Datei / Verzeichnis haben sollen).

Darüberhinaus gibt es Schreib-, Lese-, und Ausführrechte, welche zwischen Besitzer, Gruppe und allen anderen Benutzern differieren können.

Aufgrund des Alters dieses Konzeptes birgt es einige Unebenheiten. So kann jeder Datei bzw. jedem Verzeichnis nur eine Gruppe zugeordnet werden. Dies macht es unmöglich einer Gruppe von Benutzern Schreib- und Leserechte zu geben, während eine andere Gruppe nur Leserechte erhält, und alle anderen Benutzer keine Rechte haben sollen.



Linux- Basics- Workshop

Rechtekonzept

Die einzelnen Rechte werden in der Verzeichnisanzeige angezeigt. So hat jedes Recht ein bestimmtes Symbol. Diese sehen wie folgt aus:

r: Leserecht
w: Schreibrecht
x: Ausführrecht

Bei Verzeichnissen steht das 'x' für das Recht in das entsprechende Verzeichnis wechseln zu dürfen und es sich anzeigen lassen zu können.



Linux- Basics- Workshop

Rechtekonzept

Ausschnitt einer Verzeichnisanzeige, in welche die einzelne Rechteverteilung deutlich wird.

```
rwxr-xr-x 2 root as 1024 Aug 2 1998 .  
drwxr-xr-x 10 root as 1024 Aug 2 1998 ..  
-rwxr-xr-x 1 root as 6420 Aug 2 1998 dbmmanage  
-rwxr-xr-x 1 root as 10216 Aug 2 1998 htdigest  
-rwxr-xr-x 1 root as 6412 Aug 2 1998 htpasswd
```

In der Verzeichnisanzeige werden zunächst die Rechte angezeigt, und zwar in der Reihenfolge Besitzer, Gruppe und alle anderen. Ist das entsprechende Recht nicht vergeben, so wird dies mit einem Minuszeichen angezeigt. Anschliessend werden Besitzer und Gruppe angezeigt, gefolgt von der Dateilänge und das Datum der letzten Änderung. Abgeschlossen wird diese Anzeige durch den Datei- oder Verzeichnisnamen.



Linux- Basics- Workshop

Anmelden und abmelden am Betriebssystem

Um an einem Linuxsystem arbeiten zu können, muß man sich zunächst mit am System anmelden. Dies kann entweder ueber die textorientierte Konsole oder die grafische Benutzeroberfläche sein.

Zunächst werde ich mich auf die textbasierte Konsole konzentrieren und auf die grafische Benutzeroberfläche später eingehen. Da Easylinux direkt im grafischen Modus startet, müssen Sie zunächst in den Textmodus wechseln. Dies geschieht durch die Tastenkombination `<STRG> - <ALT> - <F1> .`



Linux- Basics- Workshop

Anmelden und abmelden am Betriebssystem

Um an einem Linuxsystem arbeiten zu können, muß man sich zunächst mit am System anmelden. Dies kann entweder ueber die textorientierte Konsole oder die grafische Benutzeroberfläche sein.

Unter Linux gibt es mehrere virtuelles Terminal, an denen man paralell arbeiten kann. Typischerweise sind deas 6 Textterminal (tty1-tty6) und ein grafisches Terminal. Man kann zwischen diesen Terminal durch Tastaturkombinationen hin und herschalten, dies geschieht durch die Tastenkombination

`<STRG> - <ALT> - <F(1-10)> .`



Linux- Basics- Workshop

Verzeichnisinhalt anzeigen (ls)

Um sich den Inhalt eines Verzeichnisses anzeigen zu lassen, bedient man sich des Befehles `ls`. Wird der Befehl ohne weitere Parameter eingegeben, erfolgt eine Anzeige aller Datei- und Verzeichnisnamen. Diese Anzeige ist jedoch recht unübersichtlich, da nicht immer eindeutig ersichtlich ist, was Dateien oder Verzeichnisse sind.

Um sich mehr Übersicht zu verschaffen, gibt man der `ls` Anweisung den Parameter `-l` (für long) mit. Damit bekommt man neben dem Dateinamen auch den Dateityp, die Dateirechte, Eigentümer, Gruppe, Dateigröße und das Datum der letzten Änderung mitgeteilt.

Will man sich auch die versteckten Dateien anzeigen zu lassen, kann man dem `ls` - Befehl noch den Parameter `-a` übergeben.

Syntax:

`ls [Parameter]`



Linux- Basics- Workshop

Verzeichnisse wechseln (cd)

Der Befehl `cd` (change directory) ermöglicht das wechseln von Verzeichnissen. Als Parameter wird ihm das Zielverzeichnis übergeben. Wird kein Parameter übergeben, wird ins Heimatverzeichnis des aktiven Benutzers gewechselt.

Syntax:

```
cd [<Zielverzeichnis>]
```



Linux- Basics- Workshop

Aktuellen Standort im Verzeichnissystem anzeigen (pwd)

Um sich anzeigen zu lassen, in welchem Verzeichnis man sich gerade befinden, kann man den Befehl `pwd` (print working directory) anwenden. Der Befehl hat keine weiteren Parameter und liefert das aktuelle Verzeichnis zurück.

Syntax

`pwd`



Linux- Basics- Workshop

Verzeichnis anlegen (mkdir)

Das Kommando mkdir (make directory) legt ein neues Verzeichnis an. Als Parameter wird das neue Verzeichnis übergeben. Dies kann sowohl absolut (also mit kompletter Pfadangabe) oder relativ zum aktuellen Verzeichnis angegeben werden.

Syntax

```
mkdir <Neues Verzeichnis>
```

Damit der Befehl erfolgreich ausgeführt werden kann, muß der Benutzer im entsprechenden Verzeichnis natürlich das Schreibrecht besitzen.



Linux- Basics- Workshop

Verzeichnisse löschen (rmdir)

Das löschen von Verzeichnissen erfolgt mittels des Befehles `rmdir` (remove directory). Analog zum Befehl `mkdir` wird ihm der Name (relativ oder absolut) des zu löschenden Verzeichnisses als Parameter übergeben.

Damit die Anweisung ordnungsgemäß ausgeführt werden kann, muß das zu löschende Verzeichnis leer sein und der ausführende Benutzer muß eine Schreibberechtigung für das Verzeichnis besitzen.

Syntax

```
rmdir <Verzeichnisname>
```



Linux- Basics- Workshop

Verzeichnisse mounten (mount)

Als mounten bezeichnet man das Einbinden von Datenträgern, bzw, den darauf enthaltenen Dateissystemen, in das Dateisystem des aktiven Systemes. Auf diese Art werden Floppy Disks, Festplattenpartitionen, CD Roms und alle anderen Datenträger und Massenspeichersysteme ins Dateisystem eingebunden und somit verwendbar gemacht.

Als Parameter werden dem Befehl das einzubindende Gerät (bzw. dessen Treiber) sowie die gewünschte Ziellokation im eigenen Dateisystem angegeben.

Syntax

```
mount <device> <ziellokation>
```

Die wichtigsten Parameter:

- t <fstype> Angabe des Filesystemtyp auf dem zu mountenden Gerät
- r Filesystem im Read - Only Modus (nur Lesezugriff möglich) mounten



Linux- Basics- Workshop

Verzeichnisse unmounten (umount)

Dem Namen entsprechend löst das Kommando `umount` eine mittels `mount` eingebundene Verbindung wieder auf. Als Parameter werden dem Befehl entweder das Device oder aber das Verzeichnis, in welchem die Verknüpfung existiert, angegeben.

Syntax

```
umount <device> | <Verzeichnis>
```



Linux- Basics- Workshop

Dateiinhalt anzeigen (cat)

Den Inhalt einer Datei kann man sich mittels des Befehles `cat` anzeigen lassen. Als Parameter wird dem Befehl lediglich der Name der anzuzeigenden Datei übergeben. Die Ausgabe erfolgt normalerweise auf dem Monitor.

Syntax

```
cat <Dateiname>
```

LUG Ahaus



<http://lugah.de>

Linux- Basics- Workshop

Komfortable Anzeige von Dateiinhalten (less)

Eine andere Möglichkeit sich den Inhalt von Dateien anzeigen zu lassen, ist das Programm `less`. Das Programm zeigt den Inhalt einer Datei an und bietet dem Anwender dabei eine Menge an Optionen, die der Benutzer während der Anzeige benutzen kann. Als Parameter bekommt der Befehl lediglich den Dateinamen übergeben.



Linux- Basics- Workshop

Dateien kopieren (cp)

Der Befehl `cp` (copy) ermöglicht das kopieren von Dateien. Als Parameter benötigt die Anweisung die zu kopierende Datei (Quelle) und das Ziel. Natürlich muß der Benutzer sowohl das Recht haben, die Quelldatei zu lesen, als auch an das Ziel zu schreiben.

Syntax

`cp [Parameter] <Quelle> <Ziel>`

-R Kopiert Dateien rekursiv, also auch aus Unterverzeichnissen heraus.

Beispiel: `cp -R /etc/* /tmp/`



Linux- Basics- Workshop

Dateien löschen (rm)

Natürlich ist es unter Linux auch möglich Dateien zu löschen (die nötigen Rechte vorausgesetzt). Dies geht über den Befehl `rm` (remove). Als nötiger Parameter wird dem Programm die zu löschende Datei mitgegeben. Natürlich ist auch hier die Angabe von Wildcards möglich.

Syntax

```
rm [Parameter] <zu löschende Datei>
```



Linux- Basics- Workshop

Dateien verschieben (mv)

Mittels des Kommandos mv (move) können Dateien innerhalb des Verzeichnisses verschoben werden. Dazu werden dem Befehl die Quelldateien, sowie das Ziel angegeben. Der Befehl funktioniert quasi analog zum Befehl cp, wobei hier jedoch die Quelldatei gelöscht wird.

Syntax

```
mv [Parameter] <Quelle> <Ziel>
```



Linux- Basics- Workshop

Verknüpfungen erstellen (ln)

Um unter Linux Verknüpfungen von Dateien anzulegen, bedient man sich des Befehles `ln` (link). Grundsätzlich unterscheidet man unter Linux zwei Arten von Links: Softlinks und Hardlinks. Während Hardlinks direkt in das Dateisystem eingreifen und nur innerhalb einer physikalischen Einheit gebildet werden können, sind Softlinks reine Zeiger. Ein Link zeigt lediglich auf die Originaldatei. Somit ist es möglich eine Datei an verschiedenen Stellen innerhalb des Dateisystemes vorzuhalten, ohne die Datei mehrfach vorzuhalten.

Syntax

```
ln [Parameter] <Quelldatei> <Zieldatei>
```



Linux- Basics- Workshop

Dateieigentümer ändern (chown)

Mittels des Kommandos `chown` (change owner) kann ein neuer Dateieigentümer gesetzt werden. Als Parameter werden dem Befehl dabei der Benutzername des neuen Besitzers, sowie der Name der betroffenen Datei bzw. des Verzeichnisses übergeben. Durch die Option `-R` kann man auch Änderungen rekursiv durch alle Unterverzeichnisse durchführen. Natürlich kann der Befehl auch mit Wildcardzeichen in Datei- und Verzeichnisnamen umgehen.

Syntax

```
chown [-Option] <Neuer Besitzer> <Datei oder Verzeichnisname>
```



Linux- Basics- Workshop

Benutzergruppe ändern (chgrp)

Um die Gruppenzugehörigkeit einer Datei oder eines Verzeichnisses zu ändern, bedient man sich des Befehles `chgrp` (change group). Der Befehl arbeitet parallel zu `chown`.

Syntax

```
chgrp [-Option] <Neue Gruppe> <Dateiname>
```



Dateirechte verändern (chmod)

Syntax: **chmod [OPTION] MODUS DATEI**

Die Angabe der Rechteänderung setzt sich aus drei Teilen zusammen. Der **erste Teil** gibt an, wessen Rechte geändert werden. Dabei werden folgende Abkürzungen verwendet:

- u** - User - Rechte des Dateibesitzers
- g** - Gruppe - Rechte der Dateigruppe
- o** - Andere - Rechte aller anderen Benutzer
- a** - Alle - Rechte von Besitzer, Gruppe und anderen Benutzern

Der **Zweite Teil** gibt die Aktion an:

- +** - Recht hinzufügen
- - Recht entziehen
- =** - Recht auf genau den folgenden Ausdruck setzen

Der **dritte Teil** gibt dann die Rechte an:

- r** - Read - Leserecht bei Dateien, bei Verzeichnissen, das Recht den Verzeichnisisinhalt anzuzeigen.
- w** - Write - Schreibrecht
- x** - Execute - Ausführrecht bei Dateien, bei Verzeichnissen das Recht, auf Dateien im betroffenen Verzeichnis zuzugreifen, sofern die entsprechenden Dateirechte vorhanden sind.



Dateirechte verändern (chmod)

Ein Beispiel:

ls -l

```
-rw-r--r-- 1 switch switch 0 2003-11-12 20:06 test.dat
```

chmod o+x test.dat

ls -l

```
-rw-r--r-x 1 switch switch 0 2003-11-12 20:06 test.dat
```

chmod u-r test.dat

ls -l

```
--w-r--r-x 1 switch switch 0 2003-11-12 20:06 test.dat
```



Linux- Basics- Workshop

Prozesse anzeigen (ps)

Der Befehl `ps` zeigt die eigenen Prozesse an. Da jeder laufende Prozeß einer Benutzerkennung zugeordnet ist, mit deren Privilegien der Prozeß läuft, zeigt das Kommando `ps`, wenn es ohne Optionen ausgeführt wird, nur die Prozesse an, welche unter der eigenen Benutzerkennung laufen.

Beispiel 1:

```
$ ps
PID TTY STAT TIME COMMAND
609 p0 S 0:01 -bash
654 p0 S 0:00 screen
656 a0 S 0:00 /bin/bash
3606 a0 S 0:00 tail -f /var/log/brick
3607 a0 S 0:00 grep -i dialup
18361 p1 S 0:01 -bash
18602 p1 R 0:00 ps
```

LUG Ahaus



<http://lugah.de>

Linux- Basics- Workshop

Anzeige der TOP CPU Prozesse (top)

Ein weiteres wichtiges Werkzeug zur Prozeßkontrolle ist die Anzeige der Prozesse, welche die meiste CPU Zeit benötigen. Eben diese Ausgabe bietet das Programm top . Die Ausgabe erfolgt kontinuierlich, d.h. sie zeigt immer die aktuelle CPU Auslastung und die entsprechende Verteilung auf die einzelnen Prozesse an.

-> Beispiel



Linux- Basics- Workshop

Signalübertragung an laufende Prozesse (kill)

Es ist von Benutzerseite her möglich, laufenden Prozessen Signale mitzuteilen. Dies kann durch den Befehl `kill` erreicht werden. Dabei wird dem Befehl das zu sendende Signal sowie die Prozeßnummer des gewünschten Prozesses übergeben.

Syntax

```
kill <Signal> <Prozess - ID>
```



Linux- Basics- Workshop

Programme im Hintergrund starten

Natürlich ist es auch möglich Programme gleich im Hintergrund zu starten. Dies ist jedoch nur sinnvoll, wenn das Programm keine Benutzereingaben benötigen, sondern still vor sich hin laufen. Um ein Programm im Hintergrund zu starten, wird dem Programmnamen einfach ein Leerzeichen und das kaufmännische Und Zeichen (&) angehängen.

Beispiel

```
./test.pl &
```

In diesem Beispiel würde das Programm `test.pl` im aktuellen Verzeichnis im Hintergrund gestartet.



Linux- Basics- Workshop

Ausgaben in Dateien umleiten

Um die Ausgaben eines Programmes in eine Datei zu leiten kann man sich des 'Größer als' (>) Operators bedienen. Dabei ist zu beachten, daß die Ausgabedatei immer neuangelegt bzw. überschrieben wird. Will man Daten an eine bestehende Datei anhängen, so muß man den Operator zweimal hintereinander anwenden (>>).

Beispiel:

```
ls -l > verzeichnisinhalt.txt
```

Dieses Beispiel schreibt die Ausgabe der `ls -l` Anweisung in die Datei mit dem Namen 'verzeichnisinhalt.txt'.



Linux- Basics- Workshop

Ausgaben an anderen Prozeß als Eingabe umleiten

Oftmals ist es auch sinnvoll und praktisch, die Ausgaben eines Prozesses direkt als Eingaben an einen anderen Prozeß zu senden. Diese Umleitung kann durch die Verwendung des Pipezeichens (|) realisiert werden. Dabei wird die Ausgabe des Kommandos links des Pipezeichens als Eingabe an das Kommando rechts vom Pipezeichen gesendet.

Beispiel:

```
ls -l | grep test.txt
```

In diesem Beispiel wird die Ausgabe des Befehles `ls -l` direkt an die `grep` Anweisung weitergeleitet, welche diesen Eingabestrom dann nach dem Ausdruck `test.txt` durchsucht und nur die entsprechenden Zeilen ausgibt.



Linux- Basics- Workshop

man - Das Online-Handbuch

man-Seiten sind vor allem für Kommandos und Konfigurationsdateien verfügbar.

Diese befinden sich in Verzeichnissen /usr/man, /usr/X11R6/man und anderen.

Schaut man sich mal das Verzeichnis /usr/man an, sieht man weitere

Unterverzeichnisse, die unter anderem durchnummeriert sind. Diese

Unterverzeichnisse spiegeln die Bereiche wieder, in welche die man-Seiten

eingeteilt werden. Zur Zeit gibt es die folgenden 10 Bereiche:

- 1 Benutzerkommandos
- 2 Systemaufrufe
- 3 Funktionen der Programmiersprache C
- 4 Dateiformate, Device-Dateien
- 5 Konfigurationsdateien
- 6 Spiele
- 7 Diverses
- 8 Kommandos zur Systemadministration
- 9 Kernel-Funktionen
- n neue Kommandos



Linux- Basics- Workshop

man - Das Online-Handbuch

Die Aufrufsyntax von man lautet:

man [optionen] [bereich] thema

Dabei sucht man die als Thema angegebene Manual-Datei in allen dem System bekannten man-Verzeichnissen (abgespeichert in der Variable \$MANPATH in der Datei /etc/profile). Wird statt dem Thema eine Datei angegeben, wird diese angezeigt. Die optionale Angabe des Bereiches schränkt die Suche nach man-Texten auf einen Themenbereich ein.

Das Kommando **apropos <stichwort>** sucht nach allen Manpages, in denen Stichwort vorkommt.



Linux- Basics- Workshop

rpm-Pakete installieren

Bei der Installation von externen rpm-Paketen, also solche, die nicht von der laufenden Distribution stammen, sollte man vorsichtig vorgehen. Wird zum Beispiel ein aktuelleres SysV-Paket einer RedHat-Distribution über eine SuSE-Distribution installiert, ist das System u.U. nicht mehr lauffähig. Aus diesem Grund sollten nur rpm-Pakete nachträglich installiert werden, die zur Distribution gehören.

Der Name eines rpm-Paketes gibt schon viele Informationen über das Paket an sich aus. Sie sind nach dem Schema

`<name>-<versions-nr.>-<rpm-release-nr>-<rechner-architektur>.rpm`

benannt.

LUG Ahaus



<http://lugah.de>

Linux- Basics- Workshop

Installiert wird ein Paket über die Befehlsfolge:

rpm -i [optionen] dateiname

Eine Aktualisierung eines installierten Paketes erreicht man durch

rpm -u [optionen] dateiname

und die Entfernung über

rpm -e [optionen] paketname



Tar-Pakete installieren

Nach dem Runterladen muss das Paket zuerst entpackt werden:

```
tar xvzf <DATEINAME.tar.gz>
```

bei bz2 Paketen: **j** statt **z**

Anschliessend das configure-Skript ausführen (1):

```
./configure
```

Nach dessen erfolgreichem Durchlauf das Kompilieren mit 'make' anstossen (2):

```
make
```

Die erstellten Programmdateidateien können nun mittels (3):

```
make install
```

installiert werden

LUG Ahaus



<http://lugah.de>

Linux- Basics- Workshop

**Danke,
das wars erstmal :)**

Quellen u.A:

·<http://www.linuxinfo.de/>
·<http://www.linux-fuer-alle.de>