

Einführung LINUX – Arbeiten mit Ubuntu

Erste Schritte



Lehrinhalte

- Praktischer Umgang mit einem Linux System
 - Einführung: Erste Schritte
 - Dateisystem: Navigation
 - Ausgewählte Kommandos
 - Kommandos für die Textverarbeitung
 - User und Rechte

Wo UNIX/Linux verwendet wird

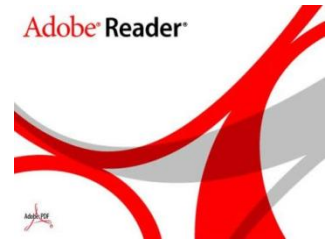
- Server Bereich: Web, DNS, DHCP, Mail, etc.
- Enterprise and mission critical applications - large databases, Enterprise Resource Management (ERM), Customer Relationship Management (CRM), data warehouse, manufacturing, supply chain management, etc.
- Hollywood - feature animation, visual effects, ...
- Scientific applications (bio & med), number-crunching (NASA)
- Smartphones: z.B.: Betriebssystem/Software-Plattform
Android: Basis ist ein Linux-Kernel
- Home Automation

Was ist Linux ?

- Als Linux oder GNU/Linux werden in der Regel **freie**, portable, Unix-ähnliche Mehrbenutzer-Betriebssysteme bezeichnet, die auf dem Linux-Kernel basieren.
- **Open-Source:** Linux wird von Softwareentwicklern auf der ganzen Welt weiterentwickelt, die an den verschiedenen Projekten mitarbeiten.
- Beteiligt sind sowohl Unternehmen als auch Non-Profit-Organisationen und Einzelpersonen, die dies als Hobby betreiben.

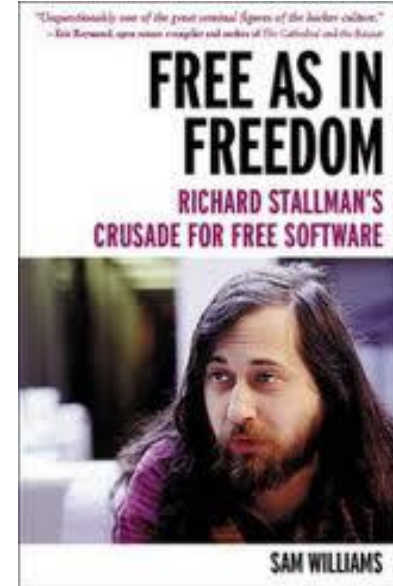
Software Lizenztypen

- Proprietäre Software
 - Lizenzverträge, Produktaktivierung, Kopiersperren,...
 - Quelltext nicht frei zugänglich
- Freeware/Shareware
 - gratis/ nach einem gewissen Zeitraum lizenzpflichtig
 - Quelltext nicht frei zugänglich
- Free Software
 - meist gratis, Weitergabe geregelt
 - Quelltext frei zugänglich



Free Software

- Richard Stallman gilt als Begründer der „Freien-Software-Bewegung“
- 1983 definiert er 4 Freiheiten für den Nutzer einer Software:
 - FH1: die Software für beliebige Zwecke zu verwenden
 - FH2: den Quellcode zu studieren, um herauszufinden, wie das Programm funktioniert
 - FH3: die Software uneingeschränkt an andere weiterzugeben
 - FH4: die Software zu verändern und zu verbessern und die Änderungen zu veröffentlichen, um sie so der Allgemeinheit zur Verfügung zu stellen



Richard Stallman

GNU und GPL



- Stallman gründete 1983 das GNU-Projekt
 - mit dem Ziel Schaffung eines unix-ähnlichen Betriebssystems zur Sicherstellung der vier Freiheiten für die Nutzer unter ausschließlicher Nutzung freier Software
 - GNU -> „GNU is not Unix“
 - Betriebssystem Kernel selbst wurde nie fertig
 - Aber viele Pakete, die ein Betriebssystem braucht, sind entstanden: Compiler, Editoren, Textverarbeitung
 - Die vom GNU-Projekt veröffentlichte Software wurde damals unter jeweils eigenen Lizenzen gestellt, Stallman nutzte dafür den Begriff **Copyleft**.
 - Später entschloss sich Stallman, eine einheitliche Lizenz zu schaffen: 1989 entwarf er die GNU General Public License (GPL), welche im Kern die vier Freiheiten umfasst und die zu den heute wichtigsten freien Software-Lizenzen zählt.

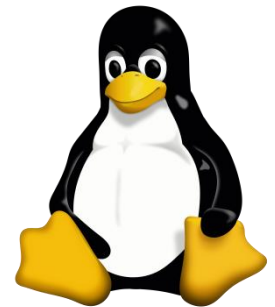
www.gnu.org/licenses/gpl.html

Linux

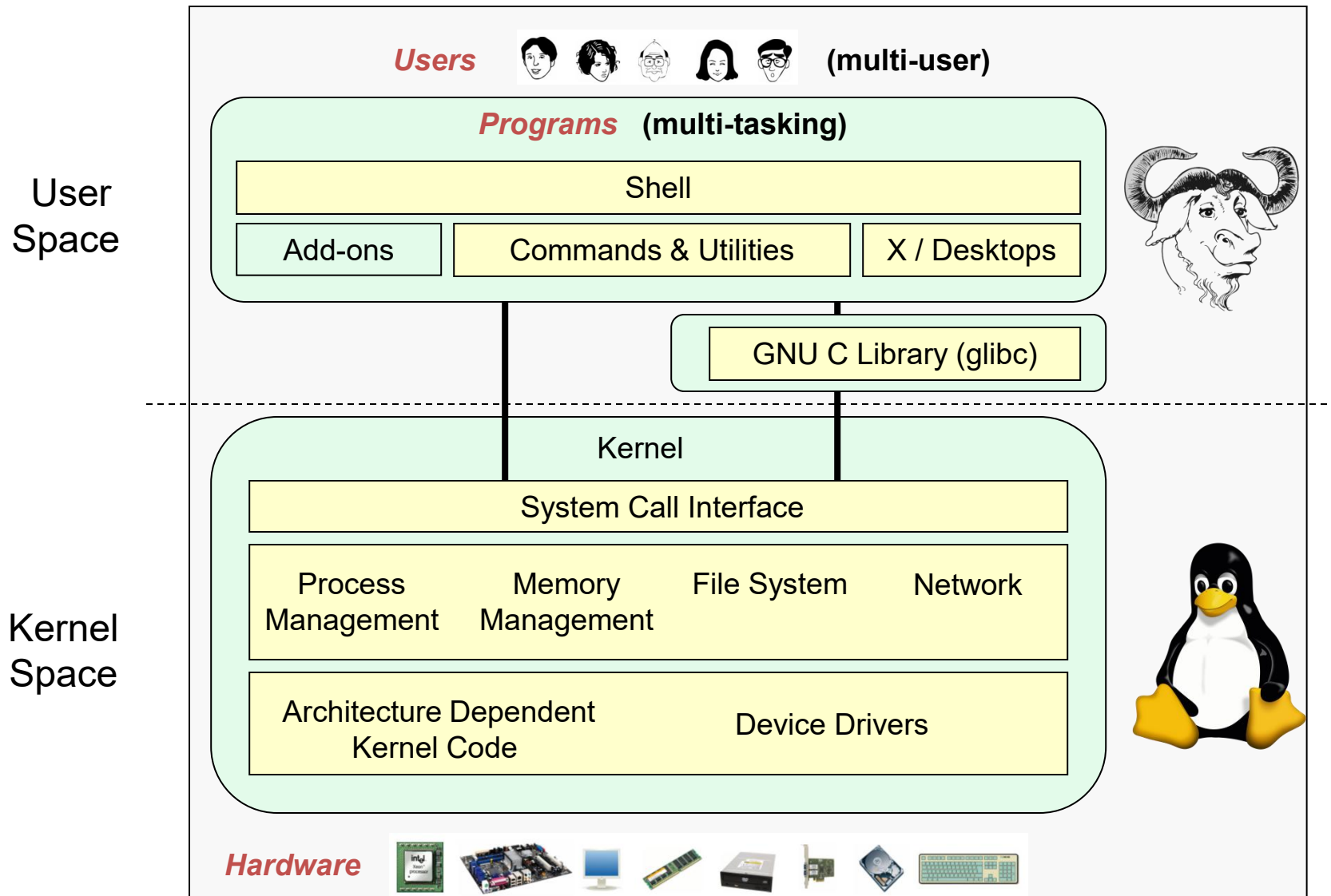
- GNU-Software verbreitete sich sehr schnell
- Allerdings fehlte noch immer die Kernfunktionalität eines Betriebssystems
- 1991 begann der Informatik Student *Linus Torvalds* hobbymäßig in Helsinki mit der Entwicklung einer Terminalemulation, die er eigentlich nur zum Zugriff auf die Unix-Server seiner Universität nutzen wollte. Er verwendete dafür den GNU-C-Compiler.
- Irgendwann „merkte“ er, dass er eigentlich ein Betriebssystem geschrieben hatte -> Linux war entstanden
- Mitte Dezember 1992 veröffentlichte er die Version 0.99 unter der GNU GPL: „*Making Linux GPL'd was definitely the best thing I ever did.*“ (aus seinem Buch „Just for fun“)
- 1993 bereits mehr als 100 Entwickler weltweit



Linus Torvalds



Linux Systemarchitektur



Linux Systemarchitektur (*Vereinfacht ...*)

Users



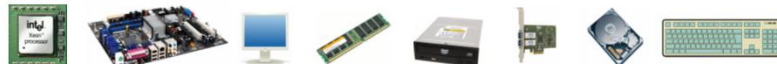
User Interface
(programs)

System Commands
(programs)

Applications
(programs)

Kernel

Hardware



CLI und GUI



User Interfaces

System
Commands

Applications

Kernel



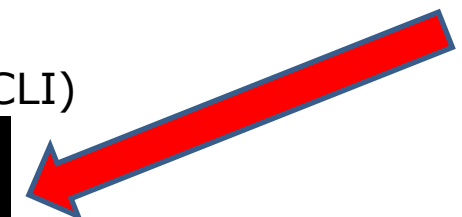
Command Line Interface (CLI)

```
[root@frida root]# iptables -L -t nat
Chain PREROUTING (policy ACCEPT)
target     prot opt source                destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source                destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source                destination
[root@frida root]#
```

bash



Text User Interface (TUI)

A firewall protects against unauthorized network intrusions. High security blocks all incoming accesses. Medium blocks access to system services (such as telnet or printing), but allows other connections. No firewall allows all connections and is not recommended.

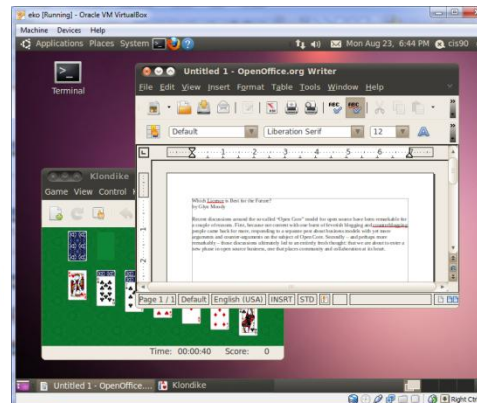
Security Level: ☒ High ☐ Medium ☐ No firewall

OK

Customize

Cancel

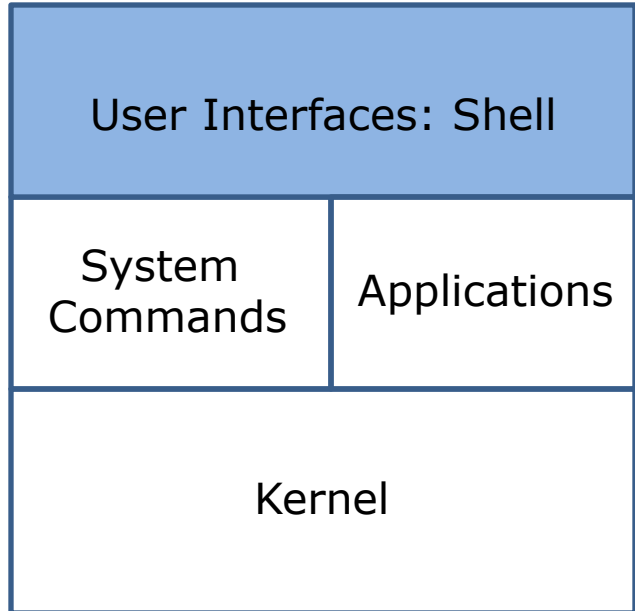
Graphic shells or desktops (GUI)



Shells, GUI

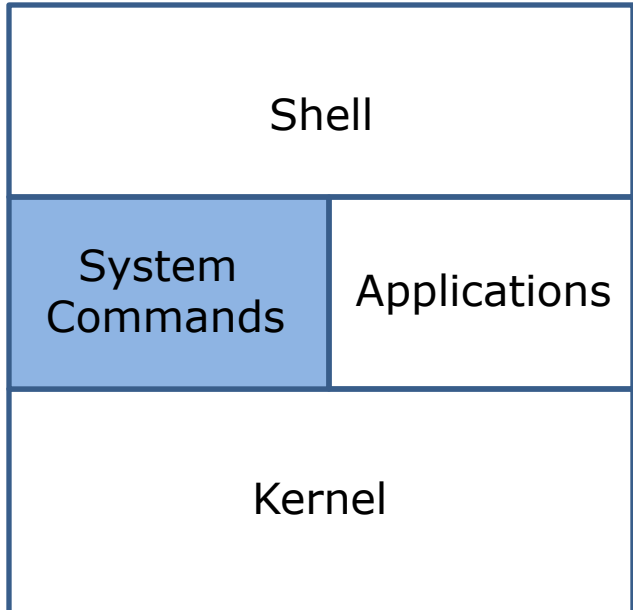
- Graphische Oberflächen (GUI)
 - sind nicht Teil des Kernels, daher beliebig austauschbar
 - müssen nicht zwangsweise installiert werden, besonders bei Server Maschinen werden GUIs nicht mit installiert.
 - Bekannte Repräsentanten: GNOME, KDE, etc...
- Kommandozeile (CLI)
 - Auch als command-line interface, Befehlszeile oder nur als „Terminal“ bezeichnet
 - läuft typischerweise im Textmodus, kann aber auch aus der jeweiligen GUI gestartet werden.
 - Gesteuert wird sie vom Kommandozeileninterpreter, der unter Unix auch als Shell bezeichnet wird.
 - bekannteste Linux Shell ist die BASH

Die Shell



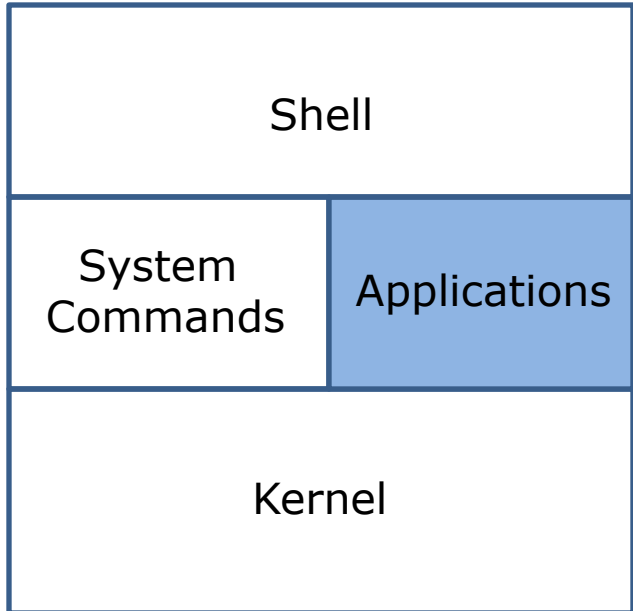
- Benutzer kommuniziert mit dem BS über die sogenannte "command line".
- Kommando wird eingegeben, die Shell analysiert (parst) das Kommando, sucht nach dem ausführbaren Programm des Kommandos im Dateisystem und führt es aus.
- Der Name "shell" beruht auf der Tatsache, dass das darunterliegende Betriebssystem versteckt ist.
- Es sind zahlreiche Shells vorhanden: sh (Bourne shell), **bash (born again shell)**, csh (C shell), ksh (Korn shell).
- Die Shell ist ein User-Interface aber auch eine einfache Programmiersprache (scripts).

System Kommandos



- 100'e von System Kommandos und Utilities
- System Kommandos z.B.: **ls** (list directories), **cat** (list a file), **rm** (remove a file), ... etc.
- Utilities z.B.: **vi** (text editor), **sort** (sorts file contents), **find** (searches),
• oder **sendmail** (email), **tar** (backup), ... etc.
- Utilities für Systemadministration z.B.: **useradd**, **groupadd**, **passwd** (change password), ... etc.

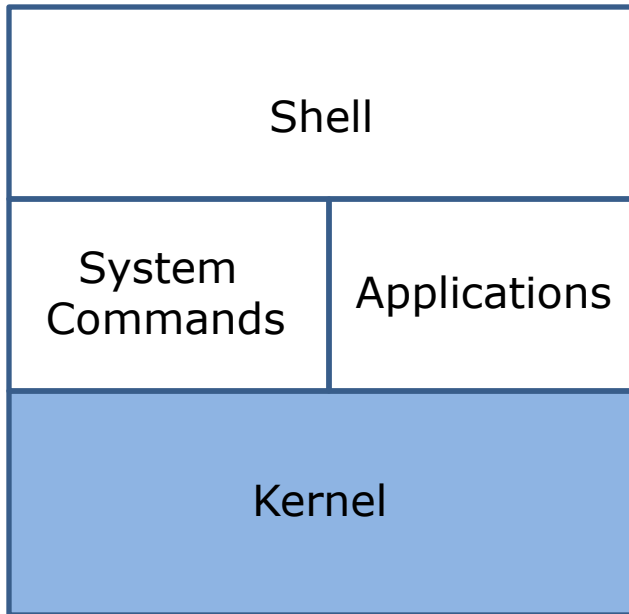
Anwendungen



- Oft Teil der Installation oder zusätzlich installierbar.
- Entwicklung durch ISVs (Independent Software Vendors) oder Open Source Organisation.
- **Beispiele:**
 - **apache2** (web server),
 - **GIMP** (GNU image manipulation program),
 - **LibreOffice** (word processing, spreadsheets, presentations),
 - **Oracle** (commercial database), ... etc.



Kernel



Der Kernel eines Betriebssystems bildet die Hardwareabstraktionsschicht

Aufgabenbereiche

- Prozessverwaltung
- Speicherverwaltung (RAM)
- Dateisysteme
- Eingabe/Ausgabe-Operationen auf verschiedenen Geräten.



Eigenschaften LINUX allgemein

- Multitasking und Multiuserfähigkeit
- GUI nicht Teil des Kernels - verschiedene GUI's vorhanden
- Dynamisch – System kann on the fly durch Hinzufügen von Kernelfunktionalität erweitert werden (z.B.: neue Treiber)
- Programme werden mit vorgegebenen Benutzerrechten ausgeführt
- Portabel – läuft auf verschiedensten Plattformen
- Robust
- **Mächtig, aber nicht alle halten es für unbedingt benutzerfreundlich !**

Was ist eine Distribution?

- Eine Linux-Distribution ist eine Zusammenstellung des Linux-Kernels mit diverser Software, um hieraus ein arbeitsfähiges Betriebssystem zu bilden.
- Unterschiede bei Distributionen in Zusammenstellung der:
 - Installations-Tools (Packet-Manager)
 - beigefügten Systemsoftware (z.B. GUI)
 - Administrationstools
 - Handbücher (Papierform) und Support

Linux Distributions Beispiele

OpenSUSE



RedHat Enterprise Linux
RHEL



Fedora



Debian



CentOS



Ubuntu



Mandriva



Ubuntu - Allgemein



- Ubuntu ist eine Linux Distribution
- Ubuntu erscheint alle 6 Monate in einer neuen Version.
 - anhand der Versionsnummer ist das Erscheinungsdatum erkennbar: Versionen 6.06, 8.10, 9.04, 9.10, 12.04 usw.
 - Die erste Ziffer steht für das Jahr und die zweite für den Monat der Veröffentlichung.
 - Ubuntu 24.04 LTS wurde also im April 2024 herausgegeben
 - Zusatz LTS bedeutet „**L**ong **T**erm **S**upport“
 - Ubuntu 25.04 ist die aktuelle Version
- Zusätzlich zu dieser Nummerierung werden Versionen mit Tiernamen gekennzeichnet.
 - Tiernamen mit Adjektiv, das mit dem selben Buchstaben anfängt, werden vergeben
 - Ubuntu 24.04 hat den Namen „Noble Numbat“

Erste Schritte im Dateisystem

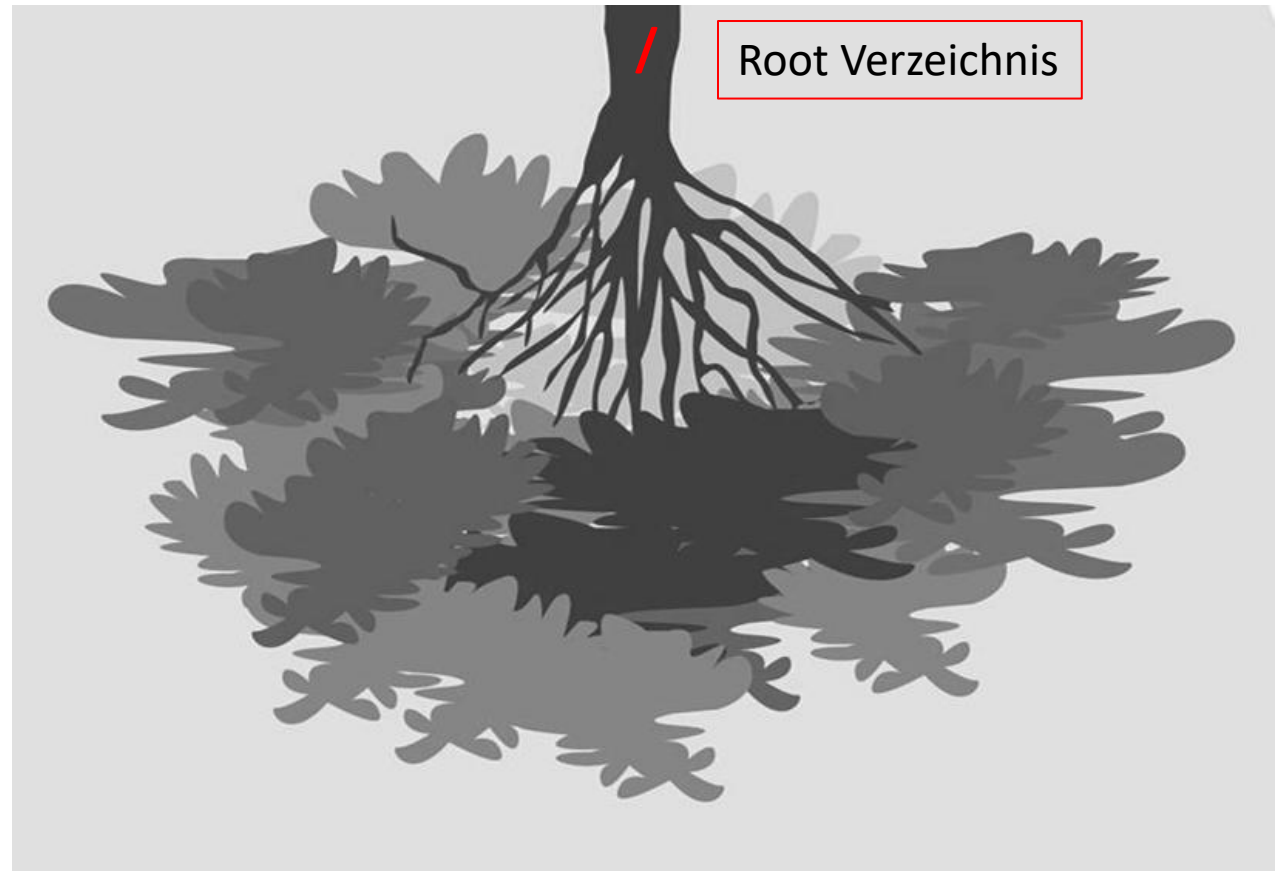
- UNIX hat wie Windows ein hierarchisch aufgebautes Verzeichnissystem
- Wurzelknoten (Root)
 - wird durch „/“ gekennzeichnet
- Home-Verzeichnis
 - Für jeden Benutzer ein eigenes vorhanden
 - Verzeichnis, in dem man sich nach dem Einloggen befindet.
 - Gehört vollständig dem Benutzer (alle Rechte)
- Aktuelles Arbeitsverzeichnis
 - Ist Verzeichnis, dessen Inhalt gerade bearbeitet wird
 - Kommandos beziehen sich auf dieses Verzeichnis, wenn keine weitere Pfadangabe vorhanden ist.

Linux Dateisystem

- FHS (Filesystem Hierarchy Standard) definiert Verzeichnisstruktur.
- Inwieweit sich die Linux-Varianten an den FHS halten bleibt den Distributoren überlassen.
- Eindeutige Baumstruktur, dessen Wurzelverzeichnis mit / angegeben wird.
- logisches Dateisystem, Partitionen und Netzlaufwerke für Benutzer nicht erkennbar, keine Laufwerksbuchstaben

Filesystem Hierarchy Standard (FHS)

/
/bin
/boot
/dev
/etc
/home
/lib
/lost+found
/mnt
/opt
/proc
/root
/sbin
/tmp
/usr



Auszug aus einer Standardverzeichnisstruktur

Directory	Contents
/bin	binary files forming the commands and shells used by the system administrator and users
/boot	files used during the initial boot-up process including the kernel
/dev	device files for connected hardware
/etc	system configuration files
/home	individual directories owned by each user
/lib	shared libraries needed to boot the system and run the commands in the root filesystem (i.e. commands in /bin and /sbin)
/lost+found	recovered files that were corrupted by power failures or system crashes
/mnt	mount points for floppies, cds, or other file systems
/opt	add-on software packages and/or commercial applications
/proc	kernel level process information
/root	home directory for the root user
/sbin	system administration commands reserved for the superuser (root)
/tmp	temporary files that are deleted when the system is rebooted or started
/usr	program files and related files for use by all users
/var	log files, print spool files, and mail queues

Kommandoeingabe CLI

Kommando

Optionen

Argumente

Kommando – Name des auszuführenden Programms

Optionen – Steueranweisungen, die die Ausführung des Programms beeinflussen

Argumente – Objekte (meistens Datei- oder Verzeichnisnamen), auf die das Kommando angewendet wird. Wird mehr als ein Objekt verwendet, werden weitere Namen durch Leerzeichen getrennt angegeben

Kommandoeingabe CLI - Beispiele

Kommando

clear

hostname

hostname

hostname

ps

ps

ps

ps

ps

ls

ls

ls

ls

echo

Optionen

-i

-s

-e

-F

-e -F

-eF

-l

-la

Argumente

/home/schaller

/home/schaller

"Red white Blue"

Erste einfache Kommandos

cal	<i>Prints calendars</i>
clear	<i>Clears the screen</i>
date	<i>Shows the time and date</i>
exit	<i>Exits shell window</i>
history	<i>Shows previous commands</i>
hostname	<i>Shows name of computer being interacted with</i>
id	<i>Shows UID's, GID's and group information</i>
ps	<i>Shows process information</i>
uname	<i>Shows name of operating system</i>
tty	<i>Shows terminal device being used for session</i>
who	<i>Shows all users who are logged in</i>
whoami	<i>Like who, but only shows your login session</i>

man ls

on-line help manual

Weitere Kommandos

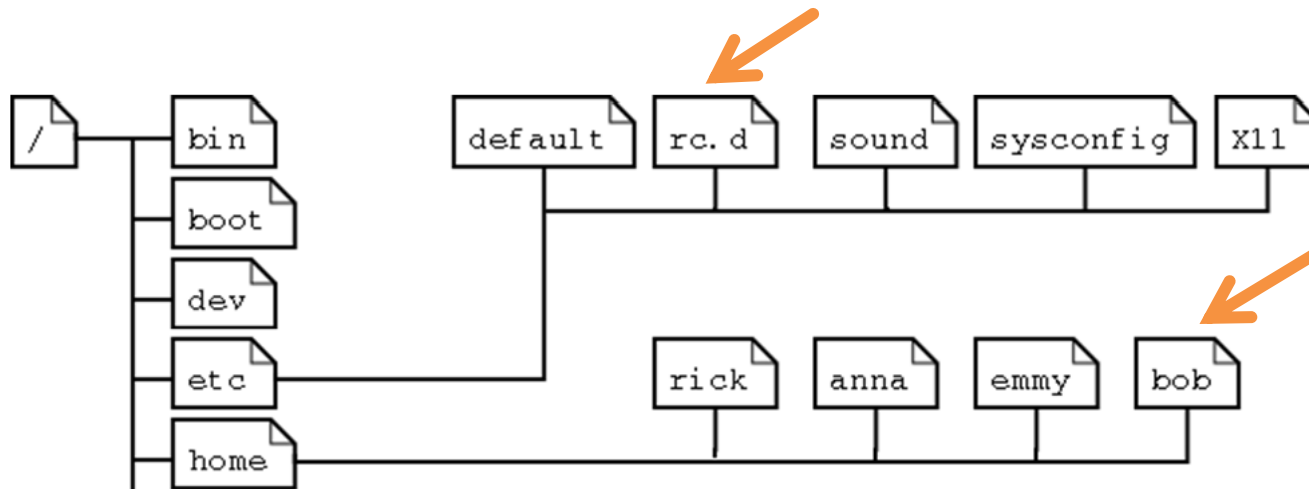
cat <i>filename</i>	<i>print contents of a file (from concatenate)</i>
cd <i>path</i>	<i>change to a new directory</i>
echo <i>string</i>	<i>print string (on screen)</i>
less <i>filename</i>	<i>print contents of a file and display page per page</i>
ls <i>path</i>	<i>list files in a directory</i>
which <i>command</i>	<i>shows where command resides on the path</i>
bc	<i>binary calculator</i>

Navigation im Dateisystem

- Kommando `cd` (change directory)
- Pfadangaben
 - Dateien werden über Pfad im Verzeichnisbaum spezifiziert
 - Absolute Pfadangabe
 - Geht immer vom ROOT-VZ aus, aktuelles Arbeitsverzeichnis egal
 - Relative Pfadangabe
 - Geht immer vom aktuellen Arbeitverzeichnis aus
- Besondere Einträge (Abkürzungen, spart Schreibarbeit)
 - . Aktuelles Arbeitsverzeichnis
 - .. Vaterverzeichnis (ein VZ über aktuellem Arbeitsverzeichnis)
 - ~ HOME-Verzeichnis, für jeden Benutzer vhd., ist aktuelles Arbeitsverzeichnis nach erfolgreichem Login

Pfadangaben - Beispiel

- Beispiel
 - Aktuelles AZ sei /home/bob, Wechseln ins VZ /etc/rc.d
 - Absolut: `cd /etc/rc.d`
 - Relativ: `cd ../../etc/rc.d`



cd Beispiele

cd cd ~	<i>Always changes to your HOME Directory</i>
cd ..	<i>go back one directory in the tree</i>
cd ../..	<i>go back two directories in the tree</i>
cd /	<i>go to root directory</i>
cd hope cd hope/ cd ./hope	<i>go to subdirectory hope in current working directory (relative path)</i>
cd .	<i>stays where you are (in current working directory)</i>
cd /root	<i>go to HOME Directory of superuser root (= system administrator)</i>

ls Kommando

Verzeichnis Inhalt anzeigen

ls display contents of current working directory

```
osboxes@osboxes:~$ ls
Desktop  Downloads  examples.desktop  Music  Public  Templates
Documents  euro16  log  Pictures  shellscript.sh  Videos
osboxes@osboxes:~$
```

ls -a display contents of current working directory
and show hidden files: .filename

blue	directories
black	Text Files
green	executable programs

```
osboxes@osboxes:~$ ls -a
.          .dmrc          .local  shellscript.sh
..         Documents    log      .sudo_as_admin_successful
.bash_history Downloads
.bash_logout euro16
.bashrc     examples.desktop .nano
.cache      .gconf         Pictures .Xauthority
.config     .ICEauthority  .profile .xsession-errors
Desktop    .lessht       Public  .xsession-errors.old
osboxes@osboxes:~$
```


ls Kommando

Verzeichnis Inhalt anzeigen

ls /bin display contents of directory /bin (absolute path)

```
echo          ntfsctl          udevadm
ed            ntfscluster  ulockmgr_server
egrep         ntfsicmp    umount
false        ntfsfallocate  uname
fgconsole    ntfsfix      uncompress
fgrep        ntfsinfo   unicode_start
findmnt      ntfsls     vdir
fuser        ntfsmove   vmouse_detect
fusermount   ntfstruncate  wdctl
getfacl      ntfswipe    which
grep         open        whiptail
gunzip       openvt      ydomainname
gzexe        pidof       zcat
gzip         ping          zcmp
hciconfig    ping6         zdiff
hostname     plymouth        zegrep
ip           plymouth-upstart-bridge zfgrep
journalctl   ps              zforce
kbd_mode     pwd            zgrep
kill         rbash      zless
kmod         readlink    zmore
less         red          znew
lessecho     rm
osboxes@osboxes:~$
```

ls Kommando

Verzeichnis Inhalt anzeigen

`ls -l` display longlisting of current working directory

```
osboxes@osboxes:~$ ls -l
total 56
drwxr-xr-x 3 osboxes osboxes 4096 Oct 17 08:32 Desktop
drwxr-xr-x 2 osboxes osboxes 4096 Oct 22 2015 Documents
drwxr-xr-x 4 osboxes osboxes 4096 Jul 8 08:38 Downloads
-rw-rw-r-- 1 osboxes osboxes 2 Jul 5 08:19 euro16
-rw-r--r-- 1 osboxes osboxes 8980 Oct 22 2015 examples.desktop
-rw-rw-r-- 1 osboxes osboxes 10 Apr 26 2016 log
drwxr-xr-x 2 osboxes osboxes 4096 Oct 22 2015 Music
drwxr-xr-x 2 osboxes osboxes 4096 Oct 22 2015 Pictures
drwxr-xr-x 2 osboxes osboxes 4096 Oct 22 2015 Public
-rwxrwxr-x 1 osboxes osboxes 600 Apr 21 2016 shellscript.sh
drwxr-xr-x 2 osboxes osboxes 4096 Oct 22 2015 Templates
drwxr-xr-x 2 osboxes osboxes 4096 Oct 22 2015 Videos
osboxes@osboxes:~$
```

Erklärung

File Type	Permissions	Link Counter	owner	group	Size in bytes	Date/time	name
d	rw-r-xr-x	2	osboxes	users	4096	2010-04-15	bin
-	rw-rw----	1	osboxes	users	0	2010-04-15	file.file

ls Kommando

weitere Beispiele

command	Function
<code>ls /</code>	List content of root directory
<code>ls ..</code>	List content of parent directory of current working directory
<code>ls -d */</code>	Only list the directories in the current directory.
<code>ls -la ~</code>	Lists ALL the files and directories in your HOME directory, in the long listing format.
<code>ls -R</code>	list subdirectories and their contents recursively

Eingabe Kommandos

- Tabulatortaste <tab>
 - Um lange Programm- oder Dateinamen einzugeben, bietet die Shell die Vervollständigung von Namen mit Hilfe der Tab-Taste an.
 - Führt die Eingabe zu mehreren Möglichkeiten, können diese durch nochmaliges Betätigen der Tab-Taste angezeigt werden.
- Cursor up und Cursor down
 - Mit diesen Tasten kann in der Command history geblättert werden, Cursor links/rechts zum Positionieren in der Eingabezeile verwenden.

Weitere Befehle

<code>mkdir <i>dirname</i></code> <code>mkdir -p</code> <code><i>dir1/dir2/dir3</i></code>	Create new directory <i>dirname</i> Create <i>dir1</i> , in <i>dir1</i> <i>dir2</i> , in <i>dir2</i> <i>dir3</i> (make parent directories as needed)
<code>rmdir <i>dirname</i></code>	Delete directory <i>dirname</i> , <i>dirname</i> must be EMPTY
<code>touch <i>filename</i></code>	Create new, empty file <i>filename</i>
<code>rm <i>filename</i></code>	Delete file <i>filename</i>
<code>rm -r <i>dirname</i></code>	Remove directory <i>dirname</i> and its contents recursively

Dateinamen

- Dateinamen
 - Max. 255 Zeichen lange Dateinamen, UTF-8 Zeichensatz
 - Unix unterscheidet immer zwischen Groß- und Kleinschreibung (**case sensitive**), auch bei Kommandos
 - Dateien, deren Namen mit einem Punkt beginnen, werden üblicherweise als „versteckte“ Dateien behandelt und standardmäßig bei `ls` nicht angezeigt, man muß `ls -a` benutzen. (z.B.: Konfigurationsdateien)
 - Metazeichen der Shell (siehe nächste Folie) sollten nicht in Filenamen verwendet werden

Metazeichen der Shell

- Shell ersetzt Metazeichen (Jokerzeichen, Wildcards) durch alle Dateinamen des aktuellen Verzeichnisses, die auf das Muster passen. Dabei können die Metazeichen beliebig oft an beliebiger Stelle im Dateinamen stehen
- Spart Schreibarbeit bei der Angabe von Dateinamen

Metazeichen	Bedeutung	Beispiel
*	Der Stern steht für eine beliebige Zeichenfolge - oder für überhaupt kein Zeichen.	"xy*" steht für alle Dateinamen, die mit "xy" anfangen, auch für "xy" selbst ("xy", "xyz", "xyab", "xyzab")
?	Das Fragezeichen steht für genau ein beliebiges Zeichen	"?xc" steht für alle Dateinamen mit 3 Zeichen, die auf „xc" enden ("axc", "bxc", "1xc", "vxc", "xxc", usw.), nicht jedoch für "xc".
[...]	Die eckige Klammer wird ersetzt durch eines der in der Klammer stehenden Zeichen. Auch ein Bereich ist möglich	[abcdefghijk] entspricht [a-k] Beispiel: "a[bcd]" wird ersetzt durch "ab", "ac" und "ad".
[! ...]	Die eckige Klammer mit Rufzeichen wird ersetzt durch eines der nicht in der Klammer stehenden Zeichen.	"[!abc]" wird ersetzt durch ein beliebiges Zeichen außer a, b oder c.
\	Der Backslash hebt den Ersetzungsmechanismus für das nachfolgende Zeichen auf,	"ab\?cd" wird zu "ab?cd" - das Fragezeichen wird übernommen.

Klammer Erweiterung

- Klammererweiterung erzeugt aus einer in geschwungenen Klammern {} eingeschlossenen Liste von Bausteinen verschiedene Zeichenketten

- Beispiele:

```
mkdir /home/administrator/{old,new,dist,bugs}
```

Legt im Verzeichnis /home/administrator die Unterverzeichnisse old,new,dist,bugs auf einmal an

```
mkdir {2007..2009}-0{1..9} {2007..2009}-{10..12}
```

Zwei Argumente wegen führender 0

Legt folgende Verzeichnisstruktur an

```
ls
```

```
2007-01 2007-07 2008-01 2008-07 2009-01 2009-07
2007-02 2007-08 2008-02 2008-08 2009-02 2009-08
2007-03 2007-09 2008-03 2008-09 2009-03 2009-09
2007-04 2007-10 2008-04 2008-10 2009-04 2009-10
2007-05 2007-11 2008-05 2008-11 2009-05 2009-11
2007-06 2007-12 2008-06 2008-12 2009-06 2009-12
```

- Funktioniert auch mit anderen Kommandos

```
rmdir {2007..2009}-0{1..9}
```


mv- Umbenennen und Verschieben von Dateien und Verzeichnissen

- `mv` dient zum Umbenennen oder Verschieben von Dateien und Verzeichnissen

<code>mv file1 file2</code>	rename file <code>file1</code> in file <code>file2</code>
<code>mv dir1 dir2</code>	rename directory <code>dir1</code> in <code>dir2</code>
<code>mv file1 file2 dir1</code>	move files <code>file1</code> and <code>file2</code> to directory <code>dir1</code> , <code>dir1</code> must exist and must be directory
<code>mv file1 dir1/filea</code>	rename file <code>file1</code> in <code>filea</code> and move it to directory <code>dir1</code>

Achtung:

Wenn das Ziel der Verschiebung existiert, wird es überschrieben
Option `-i` fragt ab!

cp- Kopieren von Dateien und Verzeichnissen

- cp dient zum Kopieren von Dateien und Verzeichnissen

<code>cp file1 file2</code>	copy file <code>file1</code> into file <code>file2</code>
<code>cp -R dir1 dir2</code>	copy directory <code>dir1</code> recursively into directory <code>dir2</code> with all files and subdirectories
<code>cp file1 file2 dir1</code>	copy files <code>file1</code> and <code>file2</code> into directory <code>dir1</code> , <code>dir1</code> must exist and must be directory
<code>cp file1 dir1/filea</code>	copy <code>file1</code> into directory <code>dir1</code> and rename copy in <code>filea</code>

Achtung:

Wenn das Ziel der Verschiebung existiert, wird es überschrieben
Option – `i` fragt ab!