Linux Seminar

Terminal.21

Fragen, Anregungen, Kritik: Stefan Walluhn, stefan@terminal21.de

1. Teil

Themen

1. **Einführung**
   Betriebssysteme
   Open Source
   UNIX
   Linux
   Multiuser / Multitasking
   Grafische Oberflächen
   Shells

2. **Erste Schritte**
   Benutzer
   Root - BigBoss
   Linux-Verzeichnissbaum
   Wechselmedien
   Dateieigenschaften
   Links

3. **Arbeiten mit der Shell**
   man-pages
   Shortcuts
   Arbeiten mit Dateien
   Piping, Datenströme
   Reguläre Ausdrücke
   Packen
   Mounten, fstab
   Shellscripts

4. **Quellcode, Programmierung, Compilierung**
   Besonderheiten OpenSource
   make
   Software Installation, allgemeiner Weg
   Software Installation, einfacher Weg
   Kernel anpassen

5. **Vorbereitung Installation**
   /proc - Dateisystem
   /dev - Dateisystem
   Systemtreiber
   Festplatten, Dateisysteme
   Linux-Systemstart, Runlevels
   Bootmanager

6. **Gentoo / Mandrake installieren, Abschluss 1. Teil**

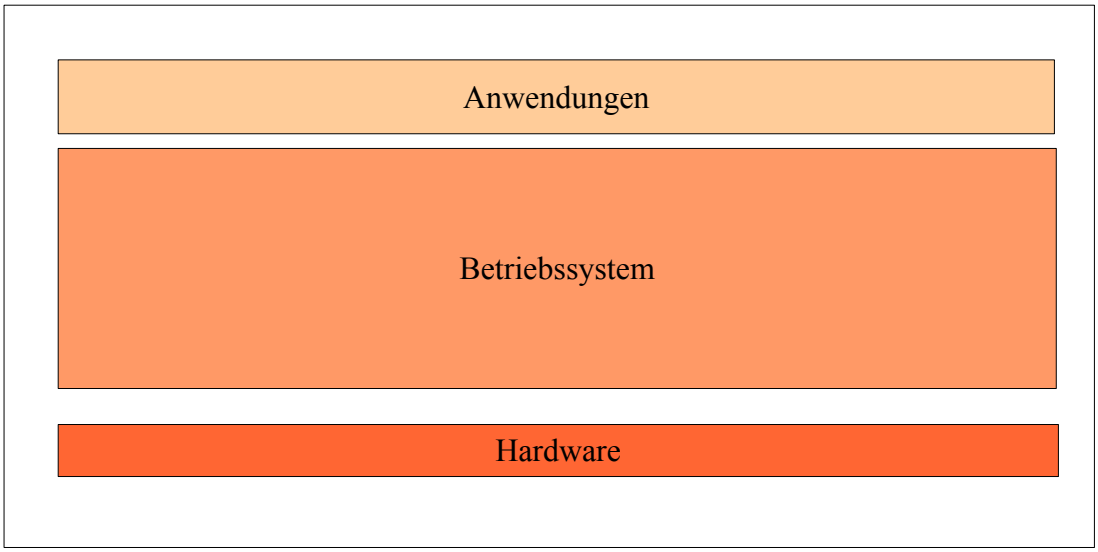
Farben und Schriften

Text:	**Erklärungen, Hinweise**
Text:	**Beispiele, Demonstrationen**
Text:	**Kommandos, die in der Shell eingegeben werden**
TEXT:	**Durch sinnvollen Inhalt ergänzen**
Text:	**Hinweis auf ein Linux-Programm**
Text:	**Hinweis auf weitere Informationen**

Einführung

Betriebssysteme

- Verwalten Hardware
- Verwalten Programme und Anwendungen
- Verteilen Speicher, Ressourcen
- Verwalten Benutzer, Rechte
- Verwalten Daten, Dateisysteme
- Bereitstellung des API
- Anwendungen <-> BS <-> Hardware müssen zusammenpassen



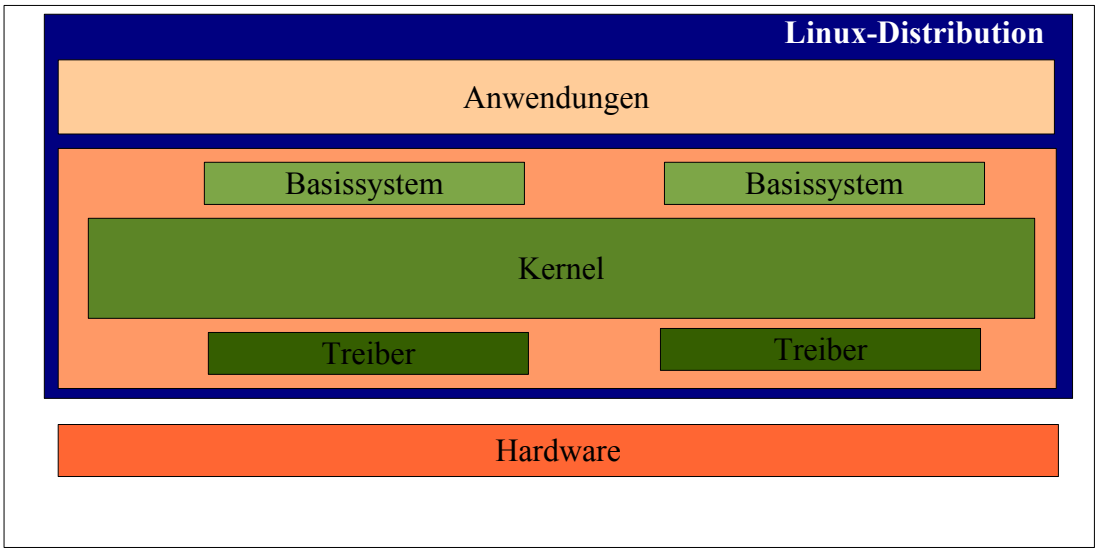
Open Source

OpenSource	***ClosedSource***
- Quelltext einseh- und veränderbar - Funktionsweise komplett bekannt - kein „geistiges Eigentum“	- Quelltext und genaue Funkionsweise nicht bekannt - nur API
Bsp.: BSD-UNIX, NetBSD, Linux	*Bsp.: WINDOWS, MS-DOS, Series 60*

UNIX

- Multitasking
- Multiuser
- POSIX-API
- Serverplattform -> beste Netzeigenschaften

Linux



Distributionen

- Beinhaltet Tools für einheitliche und einfache Installation und Verwaltung
- Zusammenstellung von Kernel, Basissystem und Anwendungen

Bsp.: Debian, Ubuntu, Gentoo (Community), SuSE, RedHat (Firmen), Knoppix (freie Live-CD)

Multitasking, Multiuser

Multitasking:
- mehrere Programme laufen gleichzeitig (echtes Multitasking)
- Quasi-Multitasking auf Einprozessor-Systemen
- Multitasking auf Mehrprozessor-Systemen
- Scheduler verwaltet Ressourcen und teilt sie den verschiedenen Programmen zu

Multiuser:
- mehrere Benutzer können ein System (gleichzeitig) benutzen
- Rechteverwaltung, Isolation der Benutzer

Grafische Benutzeroberflächen (GUI)

- Fenster, Buttons, Maus, ...
- unter Linux „normales“ Programm -> austauschbar, nicht Teil des Betriebssystems -> nicht zwingend notwendig
- unter Windows Teil des Betriebssystem

Programme *GUIs*: **KDE**, **Gnome**, XFCE, ICEwm

Shells

- nimmt Textkommandos entgegen und führt sie aus
- unter Linux Teil des Basissystems -> austauschbar
- unter Windows Programm (MS-DOS-Eingabeaufforderung)

Erste Schritte

KDE-Kurzhowto

- Einfachklick statt Doppelklick
- Rechtsklick wird unterstützt
- K-Knopf startet Programme
- Virtuelle Desktops
- Fenster anpinnen, auf andere Desktops schieben
- Drag and Drop wird umfassend unterstützt

Benutzer (Multiuser)

- jeder Benutzer besitzt eigenes Passwort und muss sich am System anmelden
- Benutzer gehört mindestens einer Gruppe an
- Benutzer / Gruppen verfügen über unterschiedliche Rechte am System
- jeder Benutzer verfügt über einen eigenen Ordner für persönliche Daten und Einstellungen
- Programme werden unter der Benutzerkennung und den Rechten des ausführenden Benutzers gestartet (*)
- Handlungen werden, ja nach Einstellung, protokolliert

Root

- Superuser, Gott
- darf alles, vollständiger Zugriff
- nur Root darf standardmäßig:
  # System herunterfahren, neu starten
  # Software installieren
  # Benutzer, Gruppen anlegen
  # Mounten
  # Gerätetreiber, Hardware starten
  # Systemkomponenten konfigurieren
  # in Ordnern anderer Benutzer schnüffeln
  # Protokolle lesen
- von Root ausgeführte Programme dürfen alles (siehe *)
  -> Sicherheitsrisiko -> nie als Root arbeiten, wenn nicht unbedingt notwendig

Linux-Verzeichnisbaum

Verzeichnis	***Inhalt***
/	Wurzelverzeichnis (root directory)
/bin	Programme des Basissystems
/boot	Kernel, Programme für den Systemstart
/dev	Gerätedateien
/etc	Konfigurationsdateien
/home	Persönliche Dateien der Benutzer
/lib	Dynamische Bibliotheken
/mnt	Mountpoint für Wechselmedien (manuell)
/media	Mountpoint für Wechselmedien (automatisch)
/opt	Optionale (ClosedSource) Programme
/proc	Laufzeitinformationen des Systems
/root	Persönliche Dateien von „Root“
/sbin	Programme von „Root“
/sys	Laufzeitinformationen des Systems (neu)
/tmp	Temporäre Dateien
/usr/bin	Programme
/usr/src/	Quelltexte /usr/src/linux => Quelltext des Kernels
/var	Daten, die Programme zur Laufzeit erzeugen (Protokolle, Status, ...)

Wechselmedien

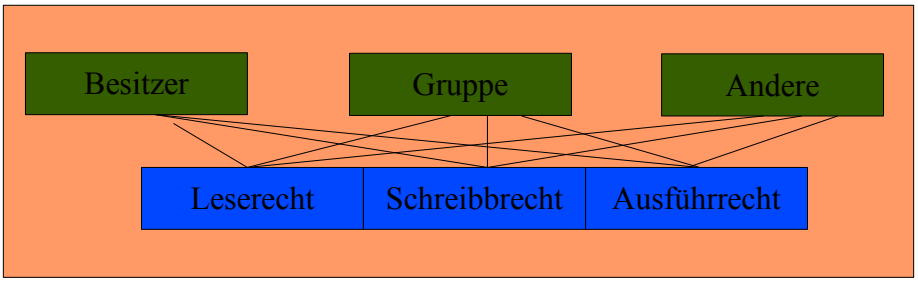
- Dateisystem auf Datenträger (Disketten, CD-ROMs, DVDs, Festplatten, Netzordnern, gepackte Dateisysteme) wird an beliebiger Stelle im Verzeichnisbaum eingehangen (gemountet)
- Mountpoint kann beliebiger (leerer) Ordner sein
- verschiedene Dateisysteme und Datenträger werden transparent zusammengefügt
- **Wechselmedien müssen vor Entnahme ausgehangen werden** (unmount), sonst droht Datenverlust (Datencache)

Vorteil:	Nachteil:
flexibles Arbeiten möglich, transparent	ungewohnter Umgang, kein C:, A:, D:

DEMO: CD-ROM mounten, an verschiedenen Mountpoints, NFS-mount

Dateieigenschaften

- Dateien haben keine Endung (z.B. kein .exe), der Typ wird automatisch erkannt
- Groß- und Kleinschreibung wird unterschieden
- Dateien, die mit *.irgendwas* beginnen werden versteckt
- Dateien, Ordner haben Besitzer und verfügen über Berechtigungen



- Dateien können ausführbar sein

DEMO: zwei Dateien anlegen mit gleichen Namen, aber Groß-Kleinschreibung, Eigenschaften ändern

Links

- Verknüpfungen zu Dateien und Ordnern
- Links können wie Originaldateien, -ordner angesprochen werden
- Links können irgendwo liegen

Hardlinks
- alle gleichberechtigt, originale Datei ist auch nur Hardlink auf deren Inhalt
- wenn der letzte Hardlink gelöscht wird, wird Datei gelöscht
- nicht über Partitionen, Datenträgergrenzen hinweg möglich

Softlinks
- Softlink -> Datei
- Datei kann unabhängig von Softlink gelöscht werden

Arbeiten mit der Shell

man-Pages

- manual -> Hilfe, Benutzerhinweise, Programmbeschreibung

DEMO: man ls, man man, Parameter, Optionen

Shortcuts

Shortcut	***Wirkung***
Alt + F1 (bis ALT + F6)	Wechsel zwischen Konsolen
Alt + F7	Wechsel zu X-Server
Strg + Alt + F1	Wechsel aus X-Server zu Konsole
Strg + Alt + ←	X-Server neu starten
Shift + Bild↑	Zurückblättern auf Konsole
Shift + Bild↓	Vorblättern auf Konsole
↑	vorher eingegebener Befehl
↓	Nachher eingegebener Befehl
Bild↑	Vorher eingegebener Befehl mit gleichen Anfangsbuchstaben
⇋	Autovervollständigung
Strg + L	Bildschirm löschen
Strg + D	Ausloggen

Arbeiten mit Dateien

Befehl	***Beschreibung***
pwd	print working directory
cd / ../ ./ ~/	change directory
ls, ls -l	list
mkdir	make directory
rmdir	remove directory
touch	touch *(Datei erzeugen)*
cp	copy
mv	move
rm, rm -r	remove (recursive)
ln, ln -s	link (soft)
find	*(Datei, Ordner suchen, vielleicht finden)*
chmod	change modus
chown	change owner
chgrp	change group
less, more, cat	*(Inhalt anzeigen)*
tail, tail -f	*(Inhalt vom Ende an anzeigen)*
grep	*(nach Inhalt suchen)*
df, df -h	disc free (human readable)
du, du -h	disc use (human readable)
free	*(freien RAM anzeigen)*
nano -w	*(Texteditor)*

Piping, Datenströme

Piping
- Ausgabe eines Programms -> Eingabe eines zweiten Programms (Grund für z.T. seltsame Standardausgabe von Linux-Programmen)
- Konzept Pipe, Piping
- Symbol für Pipe: |

DEMO: ls /etc | wc

Datenströme
- Geräte (Tastatur, Maus, Bildschirm, ...) werde als Datei aufgefasst
- Tastatur = stdin (Standardeingabe), Bildschirm = stdout (Standardausgabe)
  (kann verändert werden, z.B. Drucker = stdout)
- BEFEHL < DATEI Datei wird zu stdin
- BEFEHL > DATEI Datei wird zu stdoud

DEMO: echo ''hallo ein schöner tag'' > test
sort < test > testsortiert
cat > test

- > löscht Inhalt der Datei, >> hängt an Datei an

Argumente automatisieren
- Befehl in einfache ' erzeugt Argument

DEMO: ls -l 'cat DATEI_MIT_VERZEICHNISSEN'

Reguläre Ausdrücke

- Erstellen von Mustern zur Datei-/Inhaltssuche

.	1 beliebiges Zeichen
+	Zeichen links einmal oder mehr
*	Zeichen links null oder mehr
[afg]	Zeichen a, f oder g
[0–9]	Zeichen zwischen 0 und 9
[a-e]	Zeichen zwischen a und e
\	Folgendes Zeichen kein regulärer Ausdruck

Packen

- Packen unter Linux mit dem Befehl tar

-c	Erzeugt Archiv
-x	Packt Archiv aus
-t	Zeigt Archiv an
-f	Gibt Zieldatei an
-z	Komprimiert das Archiv mit gzip
-j	Komprimiert das Archiv mit bzip2
-p	Behält Dateirechte bei
-v	Zeigt Packverlauf ausführlich

DEMO: tar -cvf ZIELDATEI QUELLDATEI
tar -xvpf ZIELDATEI

Mounten, fstab

- nur root darf mounten -> su
- Das zu mountende Gerät (Device) wird über /dev/... angesprochen
    - IDE-Festplatten, Partitionen über /dev/hda1 ... /dev/hdz9
      (a-z entspricht Platten, 1-9 Partitionen)
    - SCSI-Festplatten, Partitionen über /dev/sda1 ... /dev/sdz9
    - Floppy über /dev/fd0 ... /dev/fd9
    - USB über /dev/usb/... oder über /dev/sdXY
- Mountpoint muss angegeben werden
- Dateisystem auf dem Datenträger wird automatisch erkannt, in Ausnahmen gibt -t Dateisystem an
- mount ohne Argumente gibt den derzeitigen Status an
- **umount nicht vergessen**

DEMO: mounten von cdrom in homeverzeichnis

- Standardmounts bzw. Mount zu Systemstart werden in /etc/fstab eingetragen

Bsp. fstab:

Datenträger	*Mountpoint*	*Dateisystem*	*Optionen*	*dump*	*Reihenfolge*
/dev/hda1	*/boot*	*ext3*	*ro,noauto*	*0*	*2*

Shellscripts

- Shell kann als Befehlsinterpreter arbeiten
- Linux-Kommandos können zu leistungsfähigen Programmen (Shellscripts) verknüpft werden
- Systemverwaltung kann mit Shellscripts automatisiert werden
- Beim Systemstart werden zum Initialisieren Shellscripts aufgerufen
- Kommandos werden nacheinander ausgeführt (vgl. MS-DOS Batch)

DEMO:

```
#!/bin/bash
#Ein kleines Demoscript
echo '' ein kleinses Demoscript''
ls -l
echo '' Das ist der Inhalt vom Verzeichnis''
pwd
```

- Shellscripts können bzw. verfügen über:

```
# Variablen
# Bedingte Anweisungen (wenn, dann)
# Schleifen (Wiederholungen)
# Funktionen
```

- Scripts über

```
sh SHELLSCRIPT
```

  ausführen oder Shellscript mit

```
chmod +x SHELLSCRIPT
```

  ausführbar machen und starten

Quellcode, Programmierung, Compilieren

Besonderheiten OpenSource

- Programmquellcode ist offen
- Programme werden im Quellcode veröffentlicht, nicht in binärer Form -> müssen auf der gewünschten, eigene Plattform selbst in binären Maschinencode übersetzt (compiliert) werden
- Compiler für verschiedenste Programmiersprachen sind Bestandteil jeder Linux-Distribution
- Problem: Compilieren ist kompliziert

Make

- Programm zum automatischen Compilieren
- Entwickler der Software beschreibt in einem Makefile wie die Compiler aufgerufen werden müssen
- einfacher Aufruf von make startet das korrekte Compilieren der Software
- Make kann im Aufruf Bedingungen übergeben bekommen (`make`, `make install`, `make dep`, `make clean`, ...)

Software installieren, allgemeiner Weg

1. Quellcode besorgen (Internet)
2. Auspacken mit `tar`
3. Programm eigenen Ansprüchen vorkonfigurieren mit `./configure` (beigelegtes Programm)
4. (Abhängigkeiten von Programmen überprüfen mit `make dep`)
5. Compilieren mit `make`
6. Binärdateien Installieren mit `make install`

Achtung: Allgemein anerkannter Weg, wird aber vom Programmierer der Software bestimmt. Die Installation kann auch anders verlaufen, immer zuerst in der Dokumentation nachlesen (RTFM).

Zusammenfassung:

```
tar -xvjpf     software.tar.gz
./configure
(make dep)
make
make install
```

Software installieren, einfacher Weg

- Distributionen beinhalten automatische Installation von vorkonfigurierten Paketen (z.T. Bereits compiliert)
  -> Packetverwaltung, Package Manager

  RPM – RedHat Package Manager
  - RedHat, SuSE, Mandrake
  - Pakete sind compiliert -> Plattform muss übereinstimmen
  - `rpm -i PAKETNAME` installiert Software
  - Grafische Oberflächen sind vorhanden

  APT – Advanced Package Tool
  - Debian, Ubuntu, Knoppix
  - Pakete sind vorcompiliert, können aber auch im Quelltext geladen und compiliert werden
  - `apt-get install PAKETNAME` installiert Software
  - Grafische Oberflächen sind vorhanden

  Portage
  - Gentoo
  - Datenbank mit Information über Quellen im Internet und Anweisungen zum automatischen Download, Auspacken, Compilieren und Installieren
  - `emerge PAKETNAME` installiert Software
  - Grafische Oberflächen sind vorhanden, aber Gentoo-User stehen übder den Dingen

Kernel anpassen

- Zentrale Komponente des Betriebssystems
- sehr vielseitig anpassbar
- entscheidet über Funktionalität und Performance
- manuelle Anpassung greift sehr tief ins System ein

<table>
<tr><th>Schritt</th><th>Kernel 2.4</th><th>Kernel 2.6</th></tr>
<tr><td>1</td><td colspan="2">Kernelquellen besorgen (www.kernel.org, Package Manager)</td></tr>
<tr><td>2</td><td colspan="2">Kernel auspacken (tar)</td></tr>
<tr><td>3</td><td colspan="2">Kernel konfigurieren (make config, besser make menuconfig)</td></tr>
<tr><td>4</td><td>Abhängigkeiten checken (make dep)</td><td>-</td></tr>
<tr><td>5</td><td>Aufräumen (make clean)</td><td>-</td></tr>
<tr><td>6</td><td>Kernel compilieren (make bzImage)</td><td rowspan="2">Kernel und Kernelmodule compilieren (make)</td></tr>
<tr><td>7</td><td>Kernelmodule compilieren (make modules)</td></tr>
<tr><td>8</td><td colspan="2">Module installieren (make modules_install)</td></tr>
<tr><td>9</td><td colspan="2">Neuen Kernel nach /boot kopieren (cp arch/i386/boot/bzImage /boot)</td></tr>
<tr><td>10</td><td colspan="2">Bootmanager anpassen</td></tr>
<tr><td>11</td><td colspan="2">Neustart (wie einzigartig)</td></tr>
</table>

Vorbereitung Installation

/proc – Dateisystem

- virtuelles Dateisystem, physisch nicht auf dem Datenträger vorhanden
- stellt Systeminformationen in Form von Dateien zur Verfügung

DEMO: *less /proc/pci*
less /proc/cpuinfo

/dev – Dateisystem

- Schnittstelle zu Geräten (Festplatten, Netzwerkadaptern, ...) in Form von Dateien
- Besonderheit: /dev/null = Datennirvana
- neue Entwicklung: virtuelles Dateisystem, erstellt von udev-Dämon, auf die jeweils real vorhandene Hardware angepasst

Systemtreiber

- Kernelmodule
- können (wenn vorhanden) zur Laufzeit geladen und entladen werden (kein Neustart erforderlich)

lsmod	zeigt geladene Module
modprobe	lädt ein Modul in den Kernel
rmmod	entlädt ein Modul aus dem Kernel

- /etc/modules.autoload.d/kernel-2.x enthält Module, die beim Systemstart geladen werden

Festplatten, Dateisysteme

- Datenträger
- Können über Partitionierung in mehrere logische Datenträger aufgeteilt werden (je eine Partition für z.B. /boot, /, /home, swap)
- werden über /dev/hdxy bzw. /dev/sdxy angesprochen

Bsp: /dev/hdc4 = 4. Partition auf der 3. Festplatte

Dateisysteme

- Struktur zur Aufnahme von Dateien und Ordnern
- Partition muss über Dateisystem verfügen
- Anlegen eines Dateisystems entspricht Formatieren unter Windows

Dateisystem	***Beschreibung***
Ext2	Altes Linuxdateisystem
Ext3	Neues Linuxdateisystem mit Prüfsummen
Reiserfs	Hochleistungsdateisystem mit Prüfsummen
Swap	Virtueller Arbeitsspeicher
Fat	MS-DOS Dateisystem
Fat32	Windows95/98/ME Dateisystem
NTFS	WindowsNT/2000/XP Dateisystem
Minix	Minix-Dateisystem (Ur-Linux, Minix)
Iso9660	CD-ROM Dateisystem

- Anlegen mit `mkfs` bzw. `mkfs.DATEISYSTEM`

Linux-Systemstart, Runlevels

- BIOS initialisiert System
  springt zu
- Bootmanager im reservierten Bootsektor der Festplatte
  entpackt und startet
- Kernel mit Prozessnummer 0
  startet Prozess
- init mit Prozessnummer 1
  liest und durchläuft
- Runlevels (rc-Scripte)

Runlevel	***Bedeutung***
0	Computer ist aus
S	Einzelbenutzer ohne Netz
1	Mehrbenutzer ohne Netz
2	Mehrbenutzer mit Netz
3	Mehrbenutzer mit Netz und grafischer Oberfläche
4	Frei
5	Frei
6	Neustart

mit init RUNLEVEL kann jedes Runlevel gestartet werden
- Dienste der jeweiligen Runlevels werden über Shellscripts in /etc/init.d/ gestartet
- Zuordnung der Dienste zu den Runlevels erfolgt über symbolische Links in /etc/rcX.d/ (X = Runlevel)
- Gentoo bildet Standard-Runlevels auf eigene Runlevels ab (Runlevel boot, Runlevel default)

Bootmanager

- Programm im Bootsektor der Festplatte
- liegt an vordefinierter Adresse und wird vom Bios angesprungen
- Startet den Kernel bzw. andere installierte Betriebssysteme
- LiLo (Linux Loader), wird über /etc/lilo.conf konfiguriert und mit dem Befehl lilo in den Bootsektor kopiert
- Grub, wird über /boot/grub/menu.list konfiguriert

Programme *Bootmanager*: lilo, grub