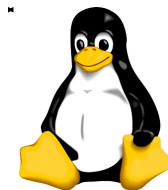


Vorlesung Betriebssysteme I

Thema 2: Linux in a Nutshell

Robert Baumgartl

25. 10. 2011



- ▶ bekanntestes Open-Source-Projekt weltweit
- ▶ Multiuser-Multitasking-Betriebssystem
- ▶ Unix-artig
- ▶ Schöpfer: Linus Torvalds
- ▶ primär kommandoorientiert, aber auch mit vielen (schönen) bunten Oberflächen bedienbar
- ▶ außerordentlich gut skalierbar
- ▶ für sehr viele Plattformen verfügbar (Auswahl): IA-32, IA-64, Sun SPARC, Motorola 68000, PowerPC, ARM, IBM S/390, MIPS, HP PA-RISC, Atmel AVR32, AD Blackfin

Im Anfang war ein Posting ...

From: torvalds@klaava.Helsinki.FI (Linus Benedict Torvalds)
Newsgroups: comp.os.minix
Subject: What would you like to see most in minix?
Summary: small poll for my new operating system
Message-ID: <1991Aug25.205708.9541@klaava.Helsinki.FI>
Date: 25 Aug 91 20:57:08 GMT
Organization: University of Helsinki

Hello everybody out there using minix -

I'm doing a (free) operating system (just a hobby, won't be big and professional like gnu) for 386(486) AT clones. This has been brewing since april, and is starting to get ready. I'd like any feedback on things people like/dislike in minix, as my OS resembles it somewhat (same physical layout of the file-system (due to practical reasons) among other things).

I've currently ported bash(1.08) and gcc(1.40), and things seem to work. This implies that I'll get something practical within a few months, and I'd like to know what features most people would want. Any suggestions are welcome, but I won't promise I'll implement them :-)

Linus (torvalds@kruuna.helsinki.fi)

PS. Yes - it's free of any minix code, and it has a multi-threaded fs. It is NOT protable (uses 386 task switching etc), and it probably never will support anything other than AT-harddisks, as that's all I have :-).

Evolution

- ▶ 17. September 1991: Version 0.01: 241 KiB, 8413 LoC¹
- ▶ 13. März 1994: Version 1.0.0: 563 KiB, 170.581 LoC
- ▶ 9. Juni 1996: Version 2.0.0: 2.015 KiB, 716.119 LoC
- ▶ 9. Oktober 2008: Version 2.6.26.6, mehr als 8 Millionen LoC
- ▶ 12. Oktober 2009: Version 2.6.31.3, mehr als 12 Millionen LoC
- ▶ neueste Version stets hier: <http://kernel.org/>

¹Lines of Code (Programmzeilen)

Kernel vs. System

Mit Linux i. e. S. ist der Kernel, d. h. , das eigentliche Betriebssystem gemeint.

Zu einem Linux-System gehört jedoch viel mehr:

- ▶ Basiswerkzeuge zur Bedienung
- ▶ Kommandointerpreter (Shell): bash, ksh, csh, tcsh
- ▶ Entwicklungswerkzeuge: gcc (GNU Compiler Collection)
- ▶ (textbasierte) Applikationen
- ▶ grafische Basisschnittstelle: X Window System
- ▶ Fenstermanager
- ▶ grafische Applikationen

→ Gesamtsystem wird manchmal (korrekter) **GNU/Linux** genannt.

- ▶ sind Zusammenstellungen des Kernels, von Applikationen und Werkzeugen zur Konfiguration, die ein lauffähiges Gesamtsystem erzeugen
- ▶ vereinfachen den Konfigurations- und Updateaufwand beträchtlich (Paketmanagement)
- ▶ unterscheiden sich in vielen Einzelaspekten:
 - ▶ hauptsächliches Einsatzziel Desktoprechner, Server, eingebettetes System
 - ▶ Einstellung zu proprietären Komponenten
 - ▶ Sprachanpassung (Lokalisierung)
 - ▶ ...
- ▶ Frage nach der besten Distribution führt gemeinhin zu Meinungsverschiedenheiten

Beispiele für populäre Distributionen

<i>Name</i>	<i>Merkmal</i>
Gentoo	das System wird grundlegend aus den Quellen erzeugt
Fedora	freies Linux der Fa. Red Hat
SUSE	Distribution der Fa. Novell (frei und kommerziell)
Debian	frei, stabil, (meist) etwas veraltete Applikationen
Ubuntu	anfängerfreundlich, frei
Knoppix	bekannte Live-Distribution
DVL	für die Ausbildung in BS-Sicherheit
Openmoko	spezialisiert für Smartphones

Siehe auch:

http://de.wikipedia.org/wiki/Liste_von_Linux-Distributionen

<http://upload.wikimedia.org/wikipedia/commons/8/8c/Gldt.svg>

Womit mache ich . . .

Textverarbeitung? openoffice, L^AT_EX

Kinoabend? mplayer, totem

Instant Messaging? gajim

Diashows? gqview, display

Bildbearbeitung? gimp, Imagemagick

WWW-Recherche? firefox, iceweasel, epiphany

Notensatz? lilypond

Funktionsplotting? gnuplot

Vektorgrafik? xfig

Programmeingabe? vi, emacs, joe

Qual der (Editor-)Wahl

`vi`

- ▶ auf jedem UNIX-System vorhanden
- ▶ effizient, leichtgewichtig
- ▶ arbeitet im Terminal

`emacs`

- ▶ kann alles: editieren, Mail und News lesen, browsen, Terminal bedienen, Kuchen backen ...
- ▶ sehr flexibel
- ▶ schwierig zu konfigurieren (Lisp)
- ▶ grundlegende Edit-Kommandos sind die gleichen wie in der Bash

`joe` ist ein Behelf, der nicht an die Mächtigkeit der anderen beiden Editoren heranreicht

Neal Stephenson über Emacs

“I use Emacs, which might be thought of as a thermonuclear word processor. It was created by Richard Stallman; enough said. It is written in Lisp, which is the only computer language that is beautiful. It is colossal, and yet it only edits straight ASCII text files, which is to say, no fonts, no boldface, no underlining. If you are a professional writer i.e., if someone else is getting paid to worry about how your words are formatted and printed, Emacs outshines all other editing software in approximately the same way that the noonday sun does the stars. It is not just bigger and brighter; it simply makes everything else vanish.”

(Neal Stephenson, In the Beginning . . . was the Command Line)

Grafische Nutzeroberflächen

- ▶ K Desktop Environment (KDE)
- ▶ GNOME
- ▶ WindowMaker
- ▶ Xfce
- ▶ Ion
- ▶ awesome

Unterscheidungskriterien:

- ▶ 'Look & Feel'
- ▶ Tastaturbedienbarkeit
- ▶ Umfang (Startzeit, Ressourcenbedarf)

Frage nach dem besten Windowmanager → Chaos.

Womit schaue ich Dokumente an?

<i>Extension</i>	Betrachter
.chm	xchm
.djvu	djview
.doc	openoffice, abiword
.dvi	xdvi
.jpg	gqview
.pdf	acroread, xpdf, evince
.ps	gv
.svg	Browser

- ▶ `man <kommando>` zeigt die zugehörige Manualseite
- ▶ `info <kommando>` dito, jedoch mit emacs-Steuerung
- ▶ `apropos <begriff>` zeigt zum Suchbegriff gehörige Kommandos
- ▶ der Schalter `-help` gibt zu vielen Kommandos nähere Erklärungen
- ▶ Das WWW bietet eine Fülle von Hilfen für alle Probleme rund um Linux

- ▶ Manual-Seiten sind in verschiedene Kategorien eingeteilt (`man man`)
 - ▶ Shellbefehle, z. B. `open`
 - ▶ Systemrufe, z. B. `open()`
 - ▶ Bibliotheksfunktionen, z. B. `fopen()`
- ▶ mehrere Sektionen pro Seite: NAME, SYNTAX, BESCHREIBUNG, OPTIONEN, DATEIEN, SIEHE AUCH, FEHLER, und AUTOR
- ▶ Humorige Bemerkungen sind häufig, vgl. `man 3 gets` (unter BUGS) oder `man rtfm(, sofern installiert)`

Die 20 wichtigsten Kommandos – Teil 1

<i>Kdo.</i>	<i>Zweck</i>
ls	Verzeichnisanzeige (<i>list</i>)
cd	Verzeichniswechsel (<i>change dir</i>)
cp	Kopieren von Dateien (<i>copy</i>)
mv	Bewegen von Dateien/Verzeichnissen (<i>move</i>)
rm	Löschen von Dateien/Verzeichnissen (<i>remove</i>)
mkdir	Verzeichnis anlegen (<i>make dir</i>)
rmdir	Verzeichnis löschen (<i>remove dir</i>)
chmod	Rechte einer Datei ändern (<i>change mode</i>)
less	seitenweise Anzeige von Dateien
cat	Anzeige des Dateiinhalts (<i>catalogue</i>)
w	zeigt an, wer eingeloggt ist (und was er tut)

Die 20 wichtigsten Kommandos – Teil 2

<i>Kdo.</i>	<i>Zweck</i>
grep	Suche von Zeichenketten
find	Suche nach Dateien
man	Anzeige von Manualseiten
ps	Anzeige von Prozeßstatistiken (<i>process state</i>)
kill	Zustellung von Signalen
bg	Programm in den „Hintergrund“ schicken (<i>background</i>)
top	Anzeige der rechenintensivsten Prozesse
mount	Datenträger einbinden (montieren)
du	Anzeige des Platzbedarfs von Dateien (<i>disk usage</i>)
ln	Anlegen eines Verweises (Links)

aber: nicht jedes zweibuchstabile Kürzel ist ein Kommando!

Konzept: „Alles ist eine Datei“

3 Kategorien von Dateien:

1. „gewöhnliche“ Datei = unstrukturierte Strom von Bytes
2. Verzeichnis (Directory) = Datei, die Verzeichniseinträge enthält
3. Spezialdateien:
 - ▶ Links (Hard Links, symbolische Links)
 - ▶ Geräte (zeichen- oder blockorientiert)
 - ▶ „named pipes“ (FIFOs)
 - ▶ Sockets

Vorteil: einheitliche Behandlung der abstrahierten Objekte.

- ▶ normaler Nutzerprozeß, der kontinuierlich
 1. Kommandos einliest,
 2. diese ausführt,
 3. etwaige Ausgaben des Programms am Bildschirm darstellt.
- ▶ verschiedene: `csh`, `tcsh`, `ksh`, **bash**
- ▶ Folgen von Shell-Kommandos nennt man *Shellscript*
- ▶ da die Shell auch Konstrukte für Verzweigungen, Schleifen und Funktionsaufrufe mitbringt, handelt es sich um eine Programmiersprache.
- ▶ mächtiges Werkzeug

Einfaches Shellscript

```
#!/bin/bash

# some sanity checks
if test ! -x `which mac` ; then
    printf "Please install mac first. Aborting.\n"
    >&2
    exit 127
fi
if test ! -x `which lame` ; then
    printf "Please install lame first. Aborting.\n"
    >&2
    exit 127
fi

# do the work
for FILE in *.ape ; do
    mac "$FILE" "${FILE}/ape/wav" -d
    lame -h -b320 "${FILE}/ape/wav" "${FILE}/ape/mp3"
    rm -f "${FILE}/ape/wav"
done
exit 0
```

Shell vs. Grafikoberfläche - kein Widerspruch

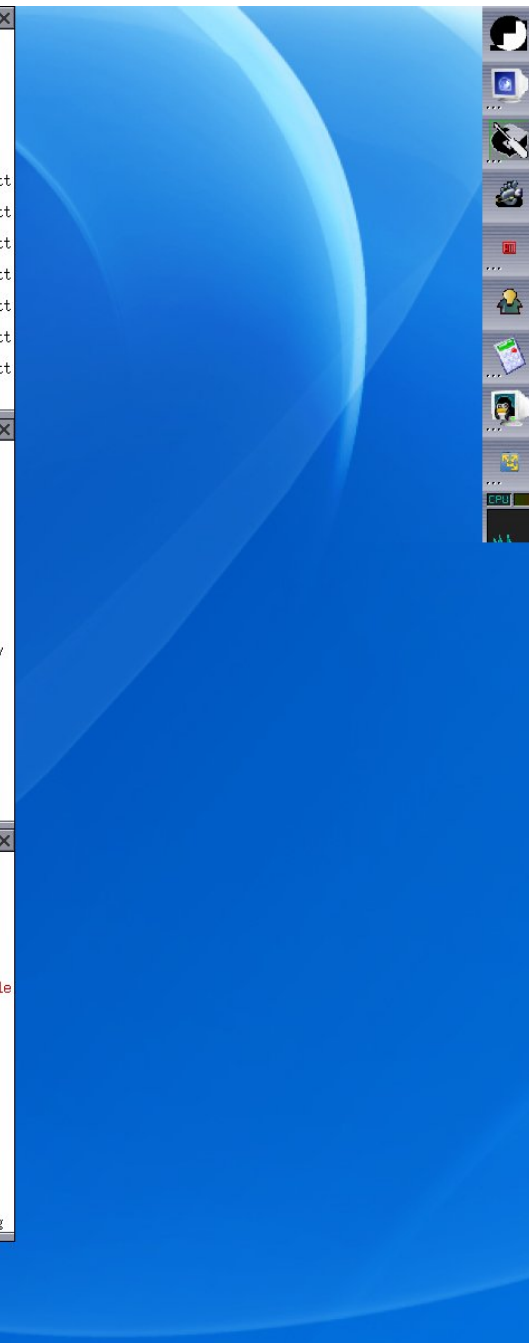
```
robge@ilpro121: ~/home/local/robge/txt/job/htw/bs1/pic
robge@ilpro121:~$ xterm &
[1] 11697
robge@ilpro121:~$ xterm &
[2] 11703
robge@ilpro121:~$ cd txt/job/htw/bs1/pic/
robge@ilpro121:~/txt/job/htw/bs1/pic$ ls
a_very_long_flight.jpg  keyboard1.jpg  t-unix-is-sexy.gif
bluetooth.jpg          linuxdistrotimeline.svg  vdc-31aug04.jpg
div_317.jpg            pic_478003001189476910.jpg  win_cement.jpg
i-use-unix.jpg         softwarewar.jpg
robge@ilpro121:~/txt/job/htw/bs1/pic$ import viele-terminals.jpg
robge@ilpro121:~/txt/job/htw/bs1/pic$ display !$
display viele-terminals.jpg
robge@ilpro121:~/txt/job/htw/bs1/pic$ import viele-terminals.jpg
robge@ilpro121:~/txt/job/htw/bs1/pic$ import viele-terminals.jpg

robge@ilpro121:~/home/local/robge
robge pts/8 :0.0 11:10 4:12 0.05s 0.05s bash
robge pts/9 :0.0 13:45 4:45m 0.08s 0.08s bash
robge pts/10 :0.0 15:19 3:35m 0.11s 0.06s bash
robge pts/11 :0.0 15:19 1:00s 0.04s 0.00s w
robge pts/12 :0.0 15:19 4:18m 0.08s 0.06s bash
robge pts/13 :0.0 15:19 4:14m 0.07s 0.05s bash
robge pts/14 :0.0 15:19 4:14m 0.06s 0.06s bash
robge@ilpro121:~$ ping augustus
PING augustus.informatik.tu-chemnitz.de (134.109.193.33) 56(84) bytes of data,
64 bytes from augustus.informatik.tu-chemnitz.de (134.109.193.33): icmp_seq=1 tt
l=57 time=2.95 ms
64 bytes from augustus.informatik.tu-chemnitz.de (134.109.193.33): icmp_seq=2 tt
l=57 time=2.59 ms
64 bytes from augustus.informatik.tu-chemnitz.de (134.109.193.33): icmp_seq=3 tt
l=57 time=2.67 ms
64 bytes from augustus.informatik.tu-chemnitz.de (134.109.193.33): icmp_seq=4 tt
l=57 time=2.59 ms
64 bytes from augustus.informatik.tu-chemnitz.de (134.109.193.33): icmp_seq=5 tt
l=57 time=2.56 ms
64 bytes from augustus.informatik.tu-chemnitz.de (134.109.193.33): icmp_seq=6 tt
l=57 time=2.76 ms
64 bytes from augustus.informatik.tu-chemnitz.de (134.109.193.33): icmp_seq=7 tt
l=57 time=2.65 ms
[]

robge@ilpro121:~/home/local/robge
robge@ilpro121:~$ xterm &
[1] 11709
robge@ilpro121:~$ ssh ilux150
ssh: ilux150: Name or service not known
robge@ilpro121:~$ ssh ilux150.informatik.tu-chemnitz.de
ssh: ilux150.informatik.tu-chemnitz.de: Name or service not known
robge@ilpro121:~$ ssh ilux150.informatik.htw-dresden.de
Password:
Last login: Thu Oct 9 09:52:12 2008 from ilpro121.informatik.htw-dresden.de
Have a lot of fun...
robge@ilux150:~$ w
 15:25:35 up 9:53, 6 users, load average: 0,04, 0,03, 0,00
USER      TTY      LOGIN@  IDLE   JCPU   PCPU WHAT
s51366 pts/0    15:16   9:10   7.52s  0,00s /bin/sh /usr/bin/firefox http:/
s56334 pts/1    15:03  17:53  0,10s  0,10s -bash
pklappro pts/2    15:16   0,00s  0,13s  0,13s -bash
s52418 pts/3    12:47  1:52m  0,29s  0,29s /u/ia03/s52418/bin/zsh -l
robge pts/4    15:25   0,00s  0,12s  0,01s w
s56347 pts/5    15:21   2:23   0,11s  0,11s -bash
robge@ilux150:~$ []

robge@ilpro121:~/home/local/robge
EXECVE(2) Linux Programmier's Manual EXECVE(2)
BEZEICHNUNG
execve - zum Ausführen von Programmen
SYNTAX
#include <unistd.h>
int execve (const char *dateiname, const char * argv[], const char
*envp[]);
BESCHREIBUNG
execve() führt das Programm aus, auf das dateiname zeigt. dateiname
muss entweder ein binäres ausführbares Programm oder ein Shell-Skript,
welches mit einer Linie in der Form "#! Interpreter: [arg]" beginnt,
sein.
execve() kehrt beim Erfolg nicht zurück und der Text, Daten, bss, und
Stapel des aufrufenden Prozesses wird durch das geladene Programm
überschrieben. Das aufgerufene Programm erbt die PID des aufrufenden
Prozesses, sowie jeden offenen Dateideskriptor, der zur Ausführung
nicht geschlossen wurde. Signale bezüglich des elterlichen Prozesses
werden gelöscht.
Manual page execve(2) line 1

robge@ilpro121:~/home/local/robge/src/viruschart
0# viruschart.c
*/
#include "viruschart.h"
void usage(void)
{
printf("Usage: viruschart [-s <singlenum> | -s all | -m <multinum>] <sysexfile
>\n");
}
char bank_encoding(unsigned char *pt)
{
switch (*pt) {
case 0:
return 'e';
case 1:
return 'A';
case 2:
return 'B';
case 3:
return 'B';
}
}
"viruschart.c" 873L, 23227C 1,1 Anfang
```



Kurzer Rundgang durchs Dateisystem

... machen wir interaktiv.

Was haben wir gelernt?

1. UNIX (in der Gestalt von Linux) ist sehr mächtig und sehr flexibel; es erfordert jedoch eine Portion Einarbeitungsaufwand.
2. Die Shell wird interaktiv bedient.
3. Shellscripsts sind Kommandofolgen der Shell; die Syntax ist ein wenig kryptisch, man kann sie aber meistern.
4. Das Dateisystem ist ein hierarchischer Baum.

- ▶ Linus Torvalds und David Diamond: *Just for Fun*. Wie ein Freak die Computerwelt revolutionierte, dtv, 2002
- ▶ <http://www.bin-bash.de/>