Software-Installation unter Linux

Übersicht

- Arten der Installation
- Software Quellen
- Beispiele
- Häufige Probleme

Vortrag weitestgehend distributions-unabhängig und naturgemäß nicht vollständig. . .

Arten der Installation

- Aus Quelltext
  - “Standalone”
  - Incl. Erzeugen eines Binärpaketes
- Binärpakete
  - Distributionseigenes Paketformat
  - “Standalone” Installer
  - Einfaches Archiv

Software Quellen

- Installationsmedien
- Webseiten/FTP Server des Distributors
- Projekt/Hersteller Homepage
- Sourceforge
- Freshmeat / Tucows
- Tuxfinder / RPMFind / RPMSeek / . . .
- Heft CDs
- Google

Binärpakete

- Distributionseigenes Paketformat
  - RPM: RedHat Package Manager (z.B. RedHat, SuSE, Mandrake)
  - DEB: Debian Paketformat (z.B. Debian)
- “Standalone” Installer
  - Graphischer Installer (z.B. Mozilla, Unreal Tournament 2003)
  - Textbasierter Installer, d.h. Shell-Skript (z.B. NVidia Treiber)
- Einfaches Archiv
  - tar Archiv (z.B. Firebird)

Installation von RPM Paketen

- Kommandozeilen Tool: `rpm`
  - `rpm -i paket.rpm` installiert eine RPM Datei
  - `rpm -e paket` deinstalliert ein Paket
  - `rpm -U *.rpm` Update von (neuen) Paketen
  - `rpm -F *.rpm` updated nur vorhandene Pakete
- Graphische Frontends
  - kpackage (KDE)
  - Midnight Commander (mc)
- Sonstiges
  - `urpmi`: Mandrake Tool, löst Abhängigkeiten automatisch
  - APT4RPM: Debians `apt-get` für RPM basierte Distris

Informationen über RPM Pakete

- Anzeige aller installierten Pakete
  - `rpm -qa`
- Infos zu einzelnen Paketen
  - `rpm -q paket` - ist das Paket installiert?
  - `rpm -qi paket` - Paketinfo
  - `rpm -ql paket` - Dateiliste
- Unterschied installiert/nicht-installiert
  - `rpm -qXXX paket` bei installierten Paketen
  - `rpm -qpXXX paket.rpm` bei RPM Datei
- Zu welchem Paket gehört eine Datei?
  - `rpm -qf datei` gibt Paketnamen aus

Fortgeschrittene Optionen zu RPM

- manche Konflikte erfordern Handarbeit, man *muß* dabei wissen, was man tut!
- Tipp: Abhängigkeiten lassen sich evtl. auflösen, indem man mehrere Pakete in einem Rutsch installiert
- Abhängigkeiten ignorieren
  - `rpm -i --nodeps paket.rpm`
  - z.B. wenn eine Abhängigkeit nicht per RPM, sondern als Source installiert wurde
- altes Paket installieren/Datei überschreiben
  - `rpm -i --force paket.rpm`
  - z.B. wenn neue Version nicht funktioniert
  - oder zwei Pakete die gleiche Datei enthalten

Arbeiten mit Source RPMs

- erkennbar an Endung `.src.rpm` oder `.srpm`
- installieren mittels `rpm -i paket.src.rpm`
  - extrahiert Sourcen, Spec Datei und evtl. Patches/Skripte
  - Pfad variiert mit der Distri (SuSE z.B. `/usr/src/packages`)
- bauen und installieren des endgültigen RPM Pakets mittels
  `rpm -ba /usr/src/packages/SPECS/paket.spec`
- evtl. ist es nötig, separates Paket zu installieren, um aus SRPMs die RPMs bauen zu können (Red Hat)

Installation von DEB Paketen

- Kommandozeilen Tool: `dpkg`
  - `dpkg -i paket.deb` installiert eine DEB Datei
  - `dpkg -r paket` deinstalliert ein Paket
  - `dpkg -l` zeigt alle installierten Pakete
  - `dpkg -L paket` zeigt Dateien aus paket
  - `dpkg -S datei` zeigt Paket, das datei enthält
- Graphische Frontends
  - dselect
  - aptitude
  - kpackage (KDE)
- Sonstiges
  - `alien`: konvertiert RPM in DEB Pakete (u. umgekehrt)

`apt-get` - ein mächtiges Frontend zu `dpkg`

- löst Abhängigkeiten automatisch
- benötigt `/etc/apt/sources.list` (im wesentl. URLs), um verfügbare Pakete zu kennen
- wichtigste Kommandos von `apt-get`
  - `apt-get install paket` installiert ein Paket plus alle benötigten Pakete
  - `apt-get remove paket` deinstalliert ein Paket
  - `apt-get update` aktualisiert Paketliste anhand `/etc/apt/sources.list`
  - `apt-get upgrade` updated auf evtl. neue Versionen installierter Pakete

Mehr zu `apt-get`

- woher Einträge in `/etc/apt/sources.list`?
  - ftp://security.debian.org - für Sicherheitsupdates
  - ftp://ftp.de.debian.org - Beispiel für Mirror
  - durchsuchbare Sammlung unter http://www.apt-get.org
- Informationen zu Paketen (auch nicht installierte!)
  - `apt-cache show paket` zeigt Deteils eines Pakets
  - `apt-cache search wort` durchsucht Paketliste nach “wort”
- auch verfügbar als APT4RPM
  - Vorsicht mit upgrade!

Quelltext selbst kompilieren

- kommt meist als (komprimiertes) Archiv: `programm.tar.gz`
  - Inhalt mit `tar tfz programm.tar.gz` ansehen
  - auspacken mit `tar xfz programm.tar.gz` (mit `xfvz` wird jedes extrahierte File auch angezeigt)
  - bei bzip2 statt gzip: `tar xfj` statt `tar xfz`
- häufig: standardisierter Build Prozeß
  - Archiv enthaelt Verzeichnis namens `programm-versionsnr`
  - `configure` Skript testet System und benoetigte Software
  - `README` und `INSTALL` Dateien geben weitere Info

Voraussetzungen

- was installiert sein muß (von Distribution nehmen)
  - `gcc` - der Compiler (plus evtl. weitere Pakete)
  - `make` - zum Automatisieren des Build-Prozesses
  - `autoconf/automake` - zur Automatisierung der Konfiguration
- häufig zusätzlich benötigt (incl. Development Paket!)
  - X11 - für graphische Programme
  - curses - oft für textbasierte Programme
  - GTK und/oder QT - populäre Graphikbibliotheken
  - KDE und/oder Gnome Bibliotheken
- manche Programme benötigen gewaltige Liste an Software, teilweise aber nur optional

Die Konfiguration

- `configure`
  - `./configure --help` zeigt verfügbare Optionen
  - `./configure --prefix=...` installiert nach `...` (default: `/usr/local`)
  - `./configure --with-xxx=...` gibt an, wo Software xxx installiert ist
- häufige Probleme
  - fehlende Software → muß nachinstalliert werden
  - benötigte Software nicht gefunden → als Option oder Shell-Variable angeben (z.B. `$QTDIR` oder `$KDEDIR`)
  - bei Problemen auch immer in `config.log` gucken

Der Kompilier-Vorgang

- `make`
  - übersetzt den Quellcode in ausführbares Programm oder Bibliothek
  - ist noch nicht die eigentliche Installation!
- `make install`
  - installiert das Paket nach `/usr/local` (falls nicht durch `--prefix` geändert)
  - einziger Schritt, der `root` Rechte benötigt (wenn in systemweites Verzeichnis)
  - falls danach Probleme mit Bibliotheken, `ldconfig` als `root` ausführen

Ausführliche Beispiele

Einfachster Fall:

```
tar xfz some.prog-0.9.2.tar.gz
cd some.prog-0.9.2
./configure
make
su -c 'make install'
```

Etwas komplizierter:

```
tar xfz nice.kde2.prog-0.8.5.tar.gz
cd nice.kde2.prog-0.8.5
export QTDIR=/usr/lib/qt2
export KDEDIR=/opt/kde2
./configure --prefix=/opt/kde2
make
su -c 'make install'
```

Sonstiges zu Quelltexten

- installiertes (Binär-)Paket xxx von `configure` als fehlend angemeckert → wahrscheinlich fehlt nur xxx-devel oder xxx-dev Paket
- aufräumen nach dem Kompilieren
  - `make clean` - entfernt nicht mehr benötigte Dateien (spart Platz)
  - `make uninstall` - deinstalliert das Paket
- manchmal werden nicht alle `make` Targets unterstützt, dann Blick in `README` und/oder `INSTALL` werfen
- falls `checkinstall` installiert ist
  - statt `make install` erzeugt `checkinstall` ein Binärpaket (RPM oder DEB) und installiert dieses

Spezielle Installationsmethoden

- Graphischer Installer wie bei Windows
  - Programm starten und Dialogen folgen z.B. `./mozilla-installer`
- Textinstaller
  - ist meist Shell Skript, zu starten mittels z.B. `sh NVIDIA-1.0-3123.suse72.i386.bin`
- gar keine eigentliche Installation
  - Firebird: Archiv einfach auspacken und Programm aus dem entpackten Verzeichnis starten
  - manche kleinen Programme enthalten nur das Programm, am besten nach `/usr/local/bin` kopieren

Zusammenfassung

- am einfachsten: Pakete des Distributors verwenden
  - CD / Webseite oder FTP (Mirror) des Distributors
  - meistens RPM oder DEB
- manche Software nur als Quelltext verfügbar
  - man benötigt Entwicklungssoftware
  - a priori nicht in Paketdatenbank der Distri integriert
- ebenfalls: Programme mit (Text/Graphik) Installer
- es gibt noch weitere Frontends zu Paketformaten
  - des Distributors, z.B. Yast2 bei SuSE
  - allg. aber evtl. an Format gebunden, z.B. Synaptic
- andere Installationsarten durchaus denkbar. . .

Ausgelassenes

- Benutzung der vielen Frontends
- Paketsysteme von Gentoo und Slackware
- eigenes Erstellen von RPM oder DEB Paketen
- mischen von Paketen aus stable, testing und unstable bei Debian (bzw. Upgrade von einer Version auf die nächste)
- Software Installation unter Knoppix
- ...