Einfache Software-Installation auf Linux, Solaris, NetBSD, etc. mit pkgsrc

Hubert Feyrer <hubertf@netbsd.org>

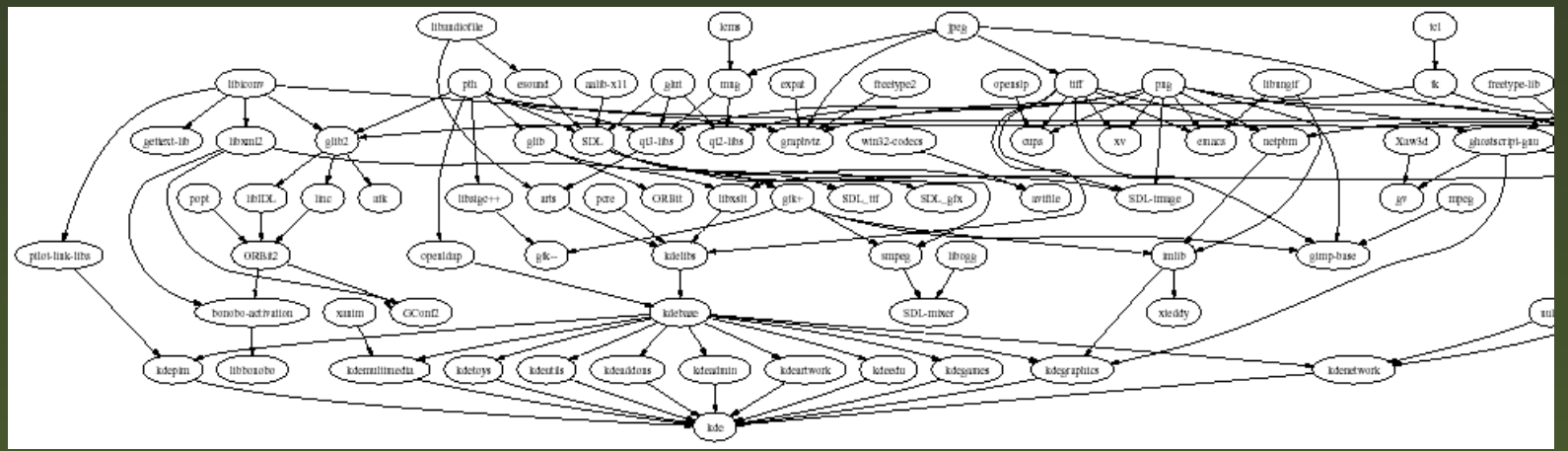
Probleme

Die Installation von Open Source Software unter Unix ist mit diversen Problemen behaftet:

- Softwarefülle und häufiger Versionswechsel
- Compilieren kostet Zeit (und ist fehleranfällig)
- Software leider oft nicht portabel programmiert (aber wir wollen hier ja kein Programmier-Seminar geben...)
- Nicht-triviale Installation:
  - Grundwissen über Werkzeuge nötig
  - Verschiedene Arten der Konfiguration (GNU autoconf, Imake, ...)

Probleme (Forts.)

- Nicht-triviale Installation (Forts.):
  - Seiteneffekte (andere Pakete, Compiler, ...)
  - Viele Abhängigkeiten:



  - Troubleshooting erfordert Expertenwissen

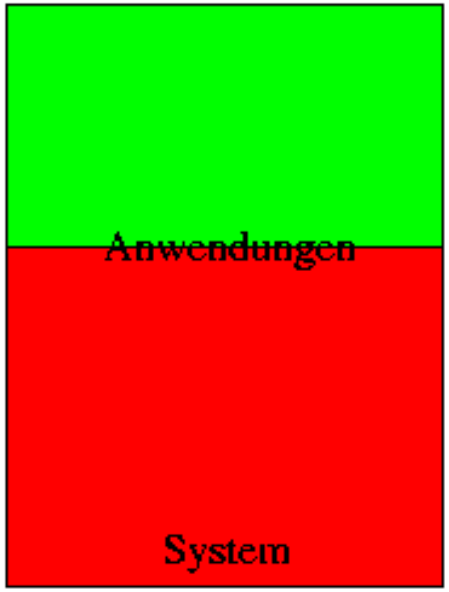
Lösungen: Je nach Umgebung! (1/2)

Klassisch, flexible Softwareverwaltung:

+ einfach zu warten
– aufwendig zu installieren

Z.B. Solaris, Irix, Gentoo Linux, Linux From Scratch

Hybrid–System



+ einfach zu installieren
+ einfach zu warten

Z.B. NetBSD, FreeBSD, OpenBSD, Debian, Crux Linux, ...

Vollstaendige Integration von Anwendungen und System:

+ einfach zu installieren
– schwierig zu warten

Z.B. SuSE Linux, RedHat Linux, Mandrake Linux, ...

Lösungen: Je nach Umgebung! (2/2)

- Einfache Installation: falls sich Software wenig ändert: vorgefertigte Binärdistribution. Z.B. für Desktop-Systeme mit Windows und SuSE von CD/DVD installieren
- Einfache Wartung: Wenn sich wenige Pakete oft ändern: Stabiles Grund-Betriebssystem, und wichtige Pakete selbst compilieren, z.B. auf Webserver mit Solaris, Apache und PHP selbstcompiliert

Portabler Lösungsansatz

Vorstellung: pkgsrc

- System zum einfachen Installieren und Updaten von Paketen
- Source-basiertes Paketverwaltungssystem
- Verwendet Original-Sourcecode zum compilieren
- Möglichkeit, Binärpakete zu erzeugen und installieren
- Komponenten: Verwaltungstools & Paketsammlung (pkgsrc)
- Abhängigkeiten werden automatisch behandelt

Vorstellung: pkgsrc (Forts.)

- Ursprünglich von FreeBSD auf NetBSD portiert
- Primäre Entwicklungsplatform für pkgsrc: NetBSD
- Portiert auf: AIX, BSD/OS, Darwin, FreeBSD, Irix, Linux, NetBSD, OpenBSD, Solaris
- Linux Distributionen: SuSE 9.0, Debian, ROOT Linux, Slackware, RedHat 8.1/9, Mandrake 9.2, ...

pkgsrc im Detail

How go?

- pkgsrc holen
- Bootstrap Kit installieren (binary oder via anoncvs & compilieren)
- cd pkgsrc/www/mozilla
- bmake install

pkgsrc holen

```
% cd $HOME/OS
% env CVS_RSH=ssh \
  cvs -d \
  anoncvs@anoncvs.netbsd.org:/cvsroot \
  co pkgsrc
U pkgsrc/Makefile
U pkgsrc/Packages.txt
U pkgsrc/README
...
```

Alternativ: ftp://ftp.netbsd.org/pub/NetBSD/NetBSD-current/tar_files/pkgsrc.tar.gz

Bootstrap Kit

- Wahlweise als Binary Kit oder selbst compiliert
- Binary Kits von http://www.pkgsrc.org/ für:

Darwin 7.0/powerpc	OpenBSD 3.2/i386
Darwin 6.6/powerpc	Slackware 8.1/i386
Debian Linux/i386	Slackware 9/i386
FreeBSD 4.7/i386	Solaris 8/sparc
FreeBSD 5.1/i386	Solaris 8/i386
IRIX 6.5/mips	Solaris 9/sparc
IRIX64 6.5/mips	Solaris 9/i386

Bootstrap Kit: Quellen holen

```
% env CVS_RSH=ssh \
  cvs -d \
  anoncvs@anoncvs.netbsd.org:/cvsroot \
  co othersrc/bootstrap-pkgsrc
U othersrc/bootstrap-pkgsrc/README
U othersrc/bootstrap-pkgsrc/bootstrap
U othersrc/bootstrap-pkgsrc/cleanup
U othersrc/bootstrap-pkgsrc/mkbinarykit
U othersrc/bootstrap-pkgsrc/pkg.sh
U othersrc/bootstrap-pkgsrc/ufsdiskimage
...
```

Bootstrap Kit: installieren I

```
% cd othersrc/bootstrap-pkgsrc/
% setenv MY_HOME $HOME/OS
% setenv LOCALBASE ${MY_HOME}/pkg
% setenv PKG_DBDIR ${MY_HOME}/db/pkg
% ./bootstrap \
?         --prefix=${LOCALBASE} \
?         --pkgdbdir=${PKG_DBDIR} \
?         --pkgsrcdir=${MY_HOME}/pkgsrc \
?         --ignore-user-check
===> bootstrap command: ./bootstrap --prefix=/home/feyrer/wo
===> bootstrap started: Tue Dec 23 09:29:26 CET 2003
===> building as unprivileged user feyrer/bedienst
.....
```

Bootstrap Kit: installieren II

```
....
===> running: /usr/xpg4/bin/sh ./install-sh -c -m 444 packag
Please remember to add /home/feyrer/work/OS/pkg/bin to your
If necessary, please remember to add /home/feyrer/work/OS/pk
Please remember to set FETCH_CMD in /etc/mk.conf to /home/fe

An example mk.conf file has been created for you in mk.conf.
with the settings you provided to bootstrap pkgsrc.

You can find extensive documentation of the NetBSD Packages
in /home/feyrer/OS/pkgsrc/Packages.txt and packages(7).

Hopefully everything is now complete.
Thank you
===> bootstrap started: Tue Dec 23 09:29:26 CET 2003
===> bootstrap ended:   Tue Dec 23 09:37:15 CET 2003
```

Bootstrap Kit: Pfade etc. anpassen

```
% cd $HOME/OS/pkg
% set path=( ‘pwd‘/bin ‘pwd‘/sbin $path )
% rehash
% setenv MAKECONF ‘pwd‘/pkgsrc_env_no-root # s.u.
% setenv PKG_DBDIR $HOME/OS/db/pkg
%
% pkg_info
digest-20021220     Message digest wrapper utility
```

Bootstrap Kit: als non-root

Folgende Werte sind in $MAKECONF zu setzen um Pakete ohne root-Rechte zu installieren (gekürzt!):

```
MY_NAME!=       whoami
MY_GROUP!=      groups | sed 's/ .*$$//'
MY_HOME=        ${HOME}/OS
BINOWN=         ${MY_NAME}
BINGRP=         ${MY_GROUP}
WRKOBJDIR=      ${MY_HOME}/tmp
X11PREFIX=      ${MY_HOME}/pkg # X needs xpkgwedge installed!
LOCALBASE=      ${MY_HOME}/pkg
VARBASE=        ${MY_HOME}/var
OBJMACHINE=     1
SU_CMD=         /bin/sh -c
CHOWN=          true
CHGRP=          true
BINMODE=        755                     # for Solaris strip(1)
```

Vollständige: http://www.feyrer.de/OS/pkgsrc_env_no-root!

Installierte Befehle

Die vom bootstrap-pkgsrc installierten Befehle bieten die Kernfunktionalität des Paketsystems:

```
% cd OS/pkg/
% ls bin sbin
bin:
bmake           cpio            ftp
digest          pax             tar

sbin:
linkfarm    pkg_add     pkg_create  pkg_info
mtree       pkg_admin   pkg_delete  pkg_view
```

Pakete compilieren

Achtung, anstatt "make" ist darauf zu achten dass das BSD-kompatible "bmake" benutzt wird!

```
% cd $HOME/OS/pkgsrc
% cd misc/figlet
% bmake
% bmake install
...
%
% pkg_info
digest-20021220     Message digest wrapper utility
figlet-2.2.1nb1     Print text banners in fancy ASCII art ch
```

Pakete compilieren II

```
% which figlet
/home/feyrer/OS/pkg/bin/figlet
% figlet Hello ‘uname -s‘
 _   _      _ _         _     _
| | | | ___| | | ___   | |   (_)_ __  _   ___  __
| |_| |/ _ \ | |/ _ \  | |   | | '_ \| | | \ \/ /
|  _  |  __/ | | (_) | | |___| | | | | |_| |>  <
|_| |_|\___|_|_|\___/  |_____|_|_| |_|\__,_/_/\_\
```

Hinter den Kulissen

1. `make fetch`: Download der Quellen
2. `make checksum`: Integrität sicherstellen
3. `make install-depends`: Benötigte Pakete installieren
4. `make extract`: Entpacken
5. `make patch`: Patches anbringen
6. `make configure`: Konfigurieren
7. `make build`: Compilieren
8. `make install`: Installieren und registrieren (für pkg_info(1), pkg_delete(), etc.)

Weitere interessante Targets

- `make package`: Binärpaket erzeugen
- `make clean`: Arbeitsverzeichnis löschen
- `make deinstall`: Paket deinstallieren
- `make replace`: Installiertes Paket durch neue Version ersetzen
- `make update`: Paket und Abhängigkeiten neu compilieren

Welche Pakete gibt's: Kategorien

```
% cd .../pkgsrc/
% ls
CVS                cross              japanese           parallel
Makefile           crypto             lang               pkglocate
Packages.txt       databases          licenses           pkgtools
README             devel              mail               plan9
archivers          distfiles          math               print
athena             doc                mbone              security
audio              editors            meta-pkgs          shells
benchmarks         emulators          misc               sysutils
biology            finance            mk                 templates
cad                fonts              nessus-libraries   textproc
chat               games              nessus-plugins     time
comms              graphics           net                wm
converters         ham                news               www
corba              inputmethod        packages           x11
```

Beispiel WWW-Kategorie

```
% cd .../pkgsrc
% ls www
CVS                     libwww                  php4-sablot
Makefile                links                   pkg
Mosaic                  links-gui               privoxy
MozillaFirebird         lynx                    privoxy-user
adzap                   lynx-current            py-HTMLgen
amaya                   mMosaic                 py-curl
analog                  make_album              py-pcgi
ap-DBI                  mknmz-wwwoffle          py-zpublisher
ap-Embperl              moz-bin-plugger         qDecoder
ap-access-referer       moz-linux-plugger       quanta
ap-aolserver            mozilla                 quanta-docs
ap-auth-cookie          mozilla-bin             quanta3
ap-auth-ldap            mozilla-bin-nightly     ruby-borges
ap-auth-pam             mozilla-flash-bin       ruby-htmlsplit
ap-auth-pgsql           mozilla-flashplugin     ruby-tag
ap-auth-postgresql      mozilla-linux           ruby-uri
ap-auth-script          mozilla-stable          ruby-webrick
...
```

Internas

Makefile: Package-Bauanleitung

```
% cat x11/xteddy/Makefile
# $NetBSD: Makefile,v 1.10 2002/08/25 21:52:57 jlam Exp $

DISTNAME=       xteddy-1.1
CATEGORIES=     x11 games
MASTER_SITES=   http://www.ITN.LiU.SE/~stegu/xteddy/

MAINTAINER=     johnam@mail.kemper.org
HOMEPAGE=       http://www.ITN.LiU.SE/~stegu/xteddy
COMMENT=        Xteddy is a cuddly teddy bear for your X Windows desktop

USE_BUILDLINK2= YES
USE_X11=        YES
GNU_CONFIGURE=  YES

pre-install:
        ${INSTALL_DATA_DIR} ${PREFIX}/share/xteddy
        ${INSTALL_DATA_DIR} ${PREFIX}/share/xteddy/pixmaps

.include "../../graphics/xpm/buildlink2.mk"

.include "../../mk/bsd.pkg.mk"
```

Abhängigkeiten

Verschiedene Arten:

- Compile-time only: BUILD_DEPENDS
- Compile- and runtime: DEPENDS
- Compile/runtime: buildlink2.mk

Abhängigkeiten: *DEPENDS

```
% cd .../pkgsrc/
% grep ^DEPEND meta-pkgs/kde3/Makefile
DEPENDS+=       kdeartwork-3.1.4:../../misc/kdeartwork3
DEPENDS+=       kdeaddons-3.1.4:../../misc/kdeaddons3
...
```

Der Variable DEPENDS werden Wertepaare der Form “Name-Version:Verzeichnis” hinzugefügt. “Name-Version” ist dabei Name und Version des benötigten Paketes, “Verzeichnis” ein Pfad relativ zum Verzeichnis des aktuellen Paketes, in dem das Paket liegt, falls es nicht bereits installiert ist und gebaut werden muß.

Abhängigkeiten: buildlink2.mk

Enthält Variablen, die angeben...

- welche Header-Dateien benutzt werden sollen
- welche Bibliotheken verwendet werden sollen
- welche Version Pakete voraussetzen sollen, die dieses Paket benutzen wollen
- in welchem pkgsrc Unterverzeichnis das Paket zum finden ist, falls es nicht installiert ist
- in welchem Verzeichnis das Paket installiert ist
- welche weiteren Pakete benötigt werden

Beispiel: buildlink2.mk

```
% cat graphics/jpeg/buildlink2.mk
# $NetBSD: buildlink2.mk,v 1.3 2003/10/03 15:35:29 salo Exp $

.if !defined(JPEG_BUILDLINK2_MK)
JPEG_BUILDLINK2_MK=     # defined

BUILDLINK_PACKAGES+=            jpeg
BUILDLINK_DEPENDS.jpeg?=        jpeg>=6b
BUILDLINK_PKGSRCDIR.jpeg?=      ../../graphics/jpeg

EVAL_PREFIX+=   BUILDLINK_PREFIX.jpeg=jpeg
BUILDLINK_PREFIX.jpeg_DEFAULT=  ${LOCALBASE}
BUILDLINK_FILES.jpeg=   include/jconfig.h
BUILDLINK_FILES.jpeg+=  include/jpeglib.h
BUILDLINK_FILES.jpeg+=  include/jmorecfg.h
BUILDLINK_FILES.jpeg+=  include/jerror.h
BUILDLINK_FILES.jpeg+=  lib/libjpeg.*

BUILDLINK_TARGETS+=     jpeg-buildlink
jpeg-buildlink: _BUILDLINK_USE

.endif  # JPEG_BUILDLINK2_MK
```

Noch Fragen?

http://www.pkgsrc.org/

http://www.NetBSD.org/packages/

info@pkgsrc.org/