

Pkgsrc binary packages management with pkgin

Emile "iMil" Heitor

FOSDEM 2012

January 31, 2012



Who am I

- Emile “iMil” Heitor, french, born on 1974
- 1st OSS contribution on 1995
- Founder of the GCU-Squad
- NetBSD user since 1997
- NetBSD developer since 2009
- CTO of NBS-System, a french OSS Internet hosting company

How and when it all started

- Debian GNU/Linux user, don't lie, *apt* is great
- Fed up with binary packages upgrades systems on *NetBSD*
- First discussion about a binary package manager on March 2 2009
- First commit on Mar 9 10:34:34 2009 UTC
- Previous name was “pkg_dry”

Dependencies and languages

- Written using the C language
- Build dependencies
 - *libarchive*
 - *libfetch*
 - *pkg_install*
- Build/run dependencies: *SQLite3*
- Around 3500 lines of code (95% C)

What is and what is not *pkgin*

- *pkgin* is **not** a *pkgsrc* (as in source) manager
- *pkgin* is a **binary** package manager
- *pkgin* **only handles** dependencies
- *pkgin* tries to mimic *apt*'s behaviour
- Calls *pkg_install* for actual installation / removal
- Relies on *pkg_info* for metadatas

Introduction
Design
Features
That's All Folks !

Who the hell is talking
History
What is it made of
What to expect
Can I have it ?

Supported (and tested) platforms



Supported (and tested) platforms



Supported (and tested) platforms



Supported (and tested) platforms



Supported (and tested) platforms



Supported (and tested) platforms



pkg_summary

A `pkg_summary(5)` file is generated for every *bulk build*

pkg_summary

A `pkg_summary(5)` file is generated for every *bulk build*

pkg_summary

```
PKGNAME=pkgfind-20050804
COMMENT=Find packages by package name in pkgsrc
SIZE_PKG=10591
BUILD_DATE=2011-01-16 23:46:59 +0000
CATEGORIES=pkgtools
LICENSE=
MACHINE_ARCH=i386
OPSYS=NetBSD
OS_VERSION=5.0.2
PKGPATH=pkgtools/pkgfind
PKGTOOLS_VERSION=20100204
REQUIRES=/usr/lib/libc.so.12
FILE_NAME=pkgfind-20050804.tgz
FILE_SIZE=6703
DESCRIPTION=pkgfind can find packages in pkgsrc. It tries to find packages which
DESCRIPTION=matches a keyword in the package name.
```

Our choices

- Parsing plain text. . . damn slow
- Loading *pkg_summary* into memory. . . overkill
- Using *bdb*, *cdb*. . . where reasonable options
- **SQLite !**

pkgin SQLite structure

Structure of the SQLite database

SQLite structure

```
sqlite> .tables
```

LOCAL_CONFLICTS	LOCAL_PROVIDES	REMOTE_CONFLICTS	REMOTE_PROVIDES
LOCAL_DEPS	LOCAL_REQUIRES	REMOTE_DEPS	REMOTE_REQUIRES
LOCAL_PKG	PKGDB	REMOTE_PKG	REPOS

```
sqlite> select pkg_id,pkgname,pkgvers,pkgpath from local_pkg where pkgname='nginx';
```

PKG.ID	PKGNAME	PKGVERS	PKGPATH
113	nginx	1.0.4	www/nginx

```
sqlite> select * from local_deps limit 5;
```

LOCAL_DEPS.ID	PKG.ID	LOCAL_DEPS.PKGNAME	LOCAL_DEPS.DEWEY
15258	1	jpeg	jpeg>=8nb1
15259	5	kpathsea	kpathsea>=6.0.0
15260	8	lcms	lcms>=1.12nb2
15261	8	jpeg	jpeg>=8nb1
15262	8	png	png>=1.5.0

Retrieving what matters

Listing package dependencies

dependencies query (*pkgindb_queries.c*)

```
const char DIRECT_DEPS[] = /* prefer higher version */
    "SELECT REMOTE_DEPS_DEWEY, REMOTE_DEPS_PKGNAME "
    "FROM REMOTE_DEPS WHERE PKG_ID = "
    "(SELECT PKG_ID FROM REMOTE_PKG WHERE PKGNAME = '%s' "
    "ORDER BY FULLPKGNAME DESC LIMIT 1);";
```

Example

1st level "kdenlive" dependencies

```
sqlite> SELECT REMOTE_DEPS_DEWEY, REMOTE_DEPS_PKGNAME FROM REMOTE_DEPS WHERE PKG_ID = \
(SELECT PKG_ID FROM REMOTE_PKG WHERE PKGNAME = 'kdenlive' ORDER BY FULLPKGNAME DESC LIMIT 1);
desktop-file-utils>=0.10nb1|desktop-file-utils
dvdauthor>=0.6.18|dvdauthor
kdelibs4>=4.5.5nb3|kdelibs4
mlt>=0.5.10|mlt
qt4-libs>=4.7.3nb1|qt4-libs
qt4-qdbus>=4.7.2nb1|qt4-qdbus
```


Inside pkgin's brain

The main objective is to provide a reliable and evolutive packages map

- Dependencies discovery
 - Retrieve 1st level dependencies
 - loop until we have a “full dependency tree”
- Impact calculation
 - Either for package installation, upgrade or removal
 - For each selected package, retrieve its “full dep tree”
 - Match it against what we have and what we would have
 - Match it against the conflict table
- Actions ordering
 - Installation: from the lesser dependency deepness
 - Removal: from the higher dependency deepness
 - Upgrade is nothing more that removal followed by installation

pkgin 0.5 features

- List, search, available packages
- Package + dependencies installation
- Package + dependencies upgrade (partial or full)
- Package + dependencies removal
- Various dependencies informations (*sd*, *sfd*, *srd*, *prov*, *req*)
- Unneeded packages removal (*autoremove*)
- Export / import packages list

Typical usage

- *foo# pkgin up*
- *foo# pkgin in texmaker*
- *foo# pkgin ex > my-pkg-list*
- *bar# pkgin im my-pkg-list*
- *foo# pkgin rm kdenlive*
- *foo# pkgin ar*

Yay, a new *pkgsrc* release!

- *foo# pkgin up*
- *foo# pkgin fug*

pkgin 0.6+ features

- More testing features
- User Interface using *PackageKit*, WIP (Sylvain Mora)
- More SQL, less C (work in progress with Baptiste from FreeBSD)
- Fix the multi-repository feature (don't use it at home kids!)

pkgin (and *NetBSD*) needs you.

pkgin in questions

Questions ?