IPS for OpenIndiana and Solaris 11

Chris Ridd

Background to IPS

IPS is the Image Packaging System developed in OpenSolaris

- Currently used by two Solarish distributions - OpenIndiana and Oracle Solaris 11 - code is still developed in the open

Born from Sun's difficulties in maintaining and testing OS patches to SVR4 packages

- In particular the numbers of ways in which admins could combine patches on a system made support difficult

SVR4 packaging was "old and busted", time for a new start

- No (real) support for installing over a network, etc etc
- Horrible names like SUNWfdj39

IPS Project

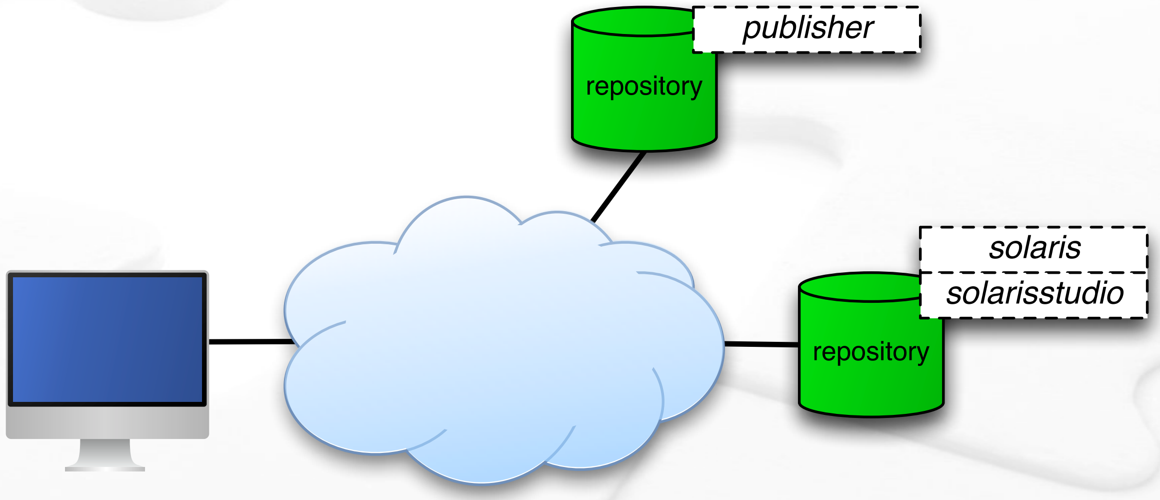
Only visible design documents are blog postings

- Search for Stephen Hahn, Bart Smaalders, Tim Foster

Principles seem to be

- Safety, basically removing postinstall/preremove etc scripts
- Allow proper dependencies
- Use “self-assembly”
- No patching
- Don’t impose a build system
- Make it easier to install less and extend later
- Be network based

Overall Design



Glossary

Package

- All the files, directories, links, dependencies, metadata
- Identified with a FMRI pkg://foo/bar/bletch@1.2

Image

- Where the packages will be installed

Publisher

- Organization (some entity) providing a number of packages
- (Used to be called “authority”)

Glossary (2)

Repository

- Server (pkg.depotd) providing packages from a number of publishers
- Two kinds - origin (metadata + data) and mirror (only data)

Consolidation

- Related packages built as a group - an artefact of how the OS is built
- The core team, the JDS team, etc all produce their own consolidations

Incorporation

- A bit like a “meta” package - e.g. “entire” means the basic OS
- Contains dependencies to impose synchronous upgrades

Basics of pkg(1)

pkg install/pkg uninstall

- Automatically follows dependencies during install

pkg list

- What is installed

pkg search

- Find the package for a file either locally or remotely

pkg update

- Update everything in an image using a new ZFS boot environment

Avoid List

Installation usually installs dependent packages

Exception is if

- a package is in a “group” dependency, and
- the package is on the avoid list

pkg avoid/pkg unavoid

- Adds/removes a package from the list

pkg install/pkg uninstall

- Removes/adds it from the list if explicitly mentioned

Facets

Facets allow a single package to contain optional parts

- “developer” bits
- documentation
- different locales
- compatibility links

Facet values are not widely documented...

- facet.devel
- facet.doc.man
- facet.locale, facet.locale.LANG
- facet.compat.x11-links, facet.compat.gnulinks

Using Facets

Show the current value for a facet (defaults to true)

-
  ```
  pkg facet doc.man
  FACETS        VALUE
  facet.doc.man True
  ```

Change a facet in the image (current image, or new BE)

-
  ```
  sudo pkg change-facet compat.x11-links=False
  ```

Variants

Variants allow packages to install alternative files depending on situation

- Architecture (SPARC vs x86)
- Zone type (global vs non-global)

Again, not widely documented

- variant.arch
- variant.opensolaris.zone

Using Variants

Show the current values

- `pkg variant`

```
VARIANT                      VALUE
variant.opensolaris.zone global
variant.arch                 i386
```

Change a variant in the image (current image, or new BE)

- `sudo pkg change-variant foo=bar`

Mediators

Maintains a symlink pointing at the preferred version of a tool

- e.g. python2.6 and python2.7 installed side by side, want /usr/bin/python to mean /usr/bin/python2.7
- Symlink is part of each package and annotated with mediator-version; pkg(1) chooses which one to use
- Can override mediator-version with mediator-priority

In theory!

In practice, no-one seems to use it (yet)

Repositories

These are set up/configured using pkgrepo(1), also svccfg(1M)

pkgrepo create

- (In the old days you just ran pkg.depotd)

pkgrepo add-publisher/pkgrepo set ... publisher/prefix=...

- Oracle's pkgrepo has add-publisher, OI's does not

pkgrepo remove/pkgrepo list

- Only in Oracle's version

pkgrepo rebuild/pkgrepo refresh

Creating an Origin Repository

Configure the pkg/server service using svccfg(1M)

- NB default pkg/inst_root = /var/pkgrepo, and pkg/readonly = true

Enable the service using svcadm(1M)

```
sudo pkgrepo create /var/myrepo
sudo pkgrepo set -s /var/myrepo \
    publisher/prefix=solarissig
sudo svccfg -s pkg/server setprop pkg/inst_root = \
    /var/myrepo
sudo svccfg -s pkg/server setprop pkg/readonly = false
sudo svcadm enable pkg/server
```

Creating a Mirror Repository

Obtain a copy of the origin repository

- Oracle has a separate repository ISO for 11.0 - 6.4GB
- NB that ISO has a useful README file
- Copy into an empty directory (don’t use pkgrepo create)
- Want /path/to/mirror/pkg5.repository and /path/to/mirror/publisher

Set pkg/server to run in mirror mode

- Set up another pkg/server instance listening on a different port?
-
```
sudo svccfg -s pkg/server setprop pkg/mirror = true
sudo svccfg -s pkg/server setprop pkg/inst_root = \
    /path/to/mirror
sudo svcadm enable pkg/server
```

Using a Mirror Repository

Either use the data mirror exclusively:

- `sudo pkg set-publisher -G \* -g http://mymirror/ solaris`

Or allow use of the origin if it is more up-to-date etc:

- `sudo pkg set-publisher -m http://mymirror/ solaris`

Any necessary (e.g. SSL) authentication is done with the origin

Access to the mirror is always over unauthenticated HTTP

Creating Packages

Packages are published to a repo using pkgsend(1)

Provide the files for the package from different sources:

- SVR4 packages
- tarballs
- directories (e.g. from make install)

Describe the files to include in the package manifest

- Can auto-generate the manifest as a first cut
- (But may need to annotate this later)
- `pkgsend generate insroot > initial.p5m`

Package Metadata

Metadata is included in the manifest file in an extensible notation:

- set name=... value=...
- (Double quotes around values with spaces)

Some names are used by pkg(1) and packagemanager GUI:

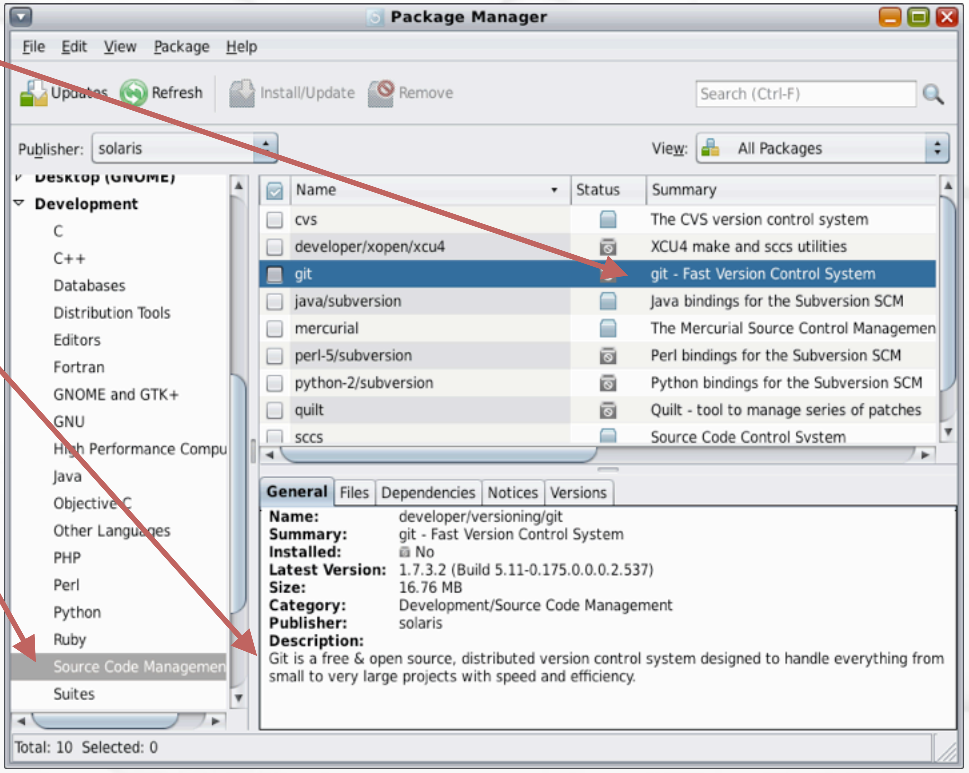
- "info.classification" has (multiple) values like:
  - org.opensolaris.category.2008:*packagemanager-gui-path*
- "pkg.summary" is a short line used in the GUI main list
- "pkg.description" is a longer line used in the GUI description pane

Metadata in Package Manager GUI

pkg.summary

pkg.description

info.classification



Name	Status	Summary
cvs		The CVS version control system
developer/xopen/xcu4		XCU4 make and sccs utilities
git		git - Fast Version Control System
java/subversion		Java bindings for the Subversion SCM
mercurial		The Mercurial Source Control Managemen
perl-5/subversion		Perl bindings for the Subversion SCM
python-2/subversion		Python bindings for the Subversion SCM
quilt		Quilt - tool to manage series of patches
sccs		Source Code Control System

Name: developer/versioning/git
Summary: git - Fast Version Control System
Installed: No
Latest Version: 1.7.3.2 (Build 5.11-0.175.0.0.0.2.537)
Size: 16.76 MB
Category: Development/Source Code Management
Publisher: solaris
Description:
Git is a free & open source, distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

isode

Package Licenses

Explicit support for click-through licenses in the manifest

- license *file* license=*short-descr* must-accept=true must-display=true
- The *file* is the path to the actual license file
- The *short-descr* is a free text description of the license
  - “GPL v2”
  - “Oracle Copyright Notice”
  - etc
- must-accept and must-display are false by default

pkg(1) has --licenses to display any found

pkg(1) has --accept to click-through

Package Dependencies

These are dependencies at the package level

- depend fmri=... fmri=... type=... predicate=... root-image=...

type=require

- the FMRIs must be present

type=optional

- the FMRIs may be present

type=exclude

- the FMRIs must not be present

Package Dependencies (2)

type=require-any

- one of the multiple FMRIs must be present
- e.g. emacs has require-any on emacs-gtk, emacs-no-x11, emacs-x11

type=conditional

- the FMRIs must be present iff the predicate FMRI is present

type=incorporate

- the FMRIs may be present, but versions are constrained

type=group

- the FMRIs are required unless on pkg(1)'s "avoid" list

Package Dependencies (3)

Can construct them automatically using pkgdepends(1)

- Looks at the manifest of the files being installed
- Looks at ELF (i.e. library) dependencies
- Looks at SVC dependencies (any in require_all)
- Looks at shell scripts (#!name-of-interpreter)
- Looks at Python scripts

Won’t be perfect, but a useful starting point

Installing Directories

You have to install directories before anything inside them

- Unless a required dependency installs them
- Including system directories like usr, usr/lib, etc.
- dir path=... mode=... owner=... group=...
- e.g. dir path=usr mode=0755 owner=root group=sys
- (No ACLs)

Directories are reference counted by pkg(1)

- Removed when the last package using it is uninstalled
- But only if they are empty

Installing Files

Note the parent directory needs to be installed first

- file *source-file* path=... mode=... owner=... group=... preserve=... overlay=...
- e.g. file insroot/usr/foo path=usr/foo mode=...

Package upgrades use preserve

- preserve=renameold/renamenew
  - What to do with existing files
- preserve=legacy
  - Only install if upgrading

Multiple packages can deliver same file using overlay

Installing Symbolic and Hard Links

Note the parent directory needs to be installed first

- link path=... target=...
- hardlink path=... target=...
- e.g. link path=usr/X11/bin/fbconsole target=../../bin/fbconsole

Installing Users, Groups and Drivers

You may need a new local user/group

- user username=... uid=... [gcos fields] ftpuser=...
- group groupname=... gid=...
- uid/gid can be automatically assigned
- Mention users and groups by name when installing files/directories

Device drivers

- driver name=... alias=... class=... [etc]
- See add_drv(4)

Generating Manifests

Can generate the list of files/directories/etc automatically using pkgsend(1)

- pkgsend generate *source* > manifest

Good starting point, but will need editing to add in actuators

- Also the permissions on system directories are wrong
- Permissions inconsistent with OS packages as well...
- Also facets, variants, and (theoretically) mediators

Cheat by looking at manifests of existing packages

- pkg contents -m ...

Actuators

Every action (file, dir, etc) can actuate something

- reboot=true
  - Reboot afterwards
- refresh_fmri=*smf-glob*
  - Call svcadm refresh on install/update/uninstall
- restart_fmri=*smf-glob*
  - Call svcadm restart on install/update/uninstall
- disable_fmri=*smf-glob*
  - Call svcadm disable on uninstall
- suspend_fmri=*smf-glob*
  - Call svcadm disable -t before, install, then svcadm enable after

System Actuators

Usually the smf-globs refer to your own services

But “self-assembly” can involve a few OS services, e.g.

- Installing new GUI apps
  - svc:/application/desktop-cache/desktop-mime-cache:default
  - svc:/application/desktop-cache/icon-cache:default
- Installing new security roles
  - svc:/system/rbac:default
- Installing new SMF services
  - svc:/system/manifest-import:default
- Installing new texinfo documentation
  - svc:/application/texinfo-update:default

Publishing a Package

Couple of ways to do this

Documented way (doesn't seem to work)

- `pkgsend publish -s /var/myrepo -d ... manifest`

Old way (still works!)

- The eval sets a PKG_TRANS_ID variable in the shell
-
```
eval `pkgsend -s /var/myrepo open FMRI`
pkgsend -s /var/myrepo include -d ... manifest
pkgsend -s /var/myrepo close
```

Run "pkgsend help" for more, man page seems incomplete

Package Archives

pkgrecv(1) will archive a package to disk

- Only in Oracle’s version
- Handy if you don’t have a public-facing pkg.depotd
- Can re-publish to another repository
- Or install directly

Minor Utilities

pkglint

pkgdiff

pkgfmt

pkgmerge

- Merge SPARC and i386 packages

pkgmogrify

- A macro language for manifest files

pkgsign

Observations

It (still) works

Need better granularity to avoid reboots

Be good if OpenIndiana could update their version!

Need better control over who can publish to a depot

Solaris SIG

Join us and get in contact with any feedback, ideas, comments

facebook	www.facebook.com/SolarisSIG
LinkedIn	www.linkedin.com/groups?gid=3010558
twitter	@SolarisSIG
UKOUG UK ORACLE USER GROUP Serving the Oracle community	www.ukoug.org/our-communities/solaris/

isode