

“The Package Man Always Builds Twice” -- From SysV Packaging To The Image Packaging System

Volker A. Brandt

Brandt & Brandt Computer GmbH

vab@bb-c.de

Terms and Definitions

The Old World: SysV

The New World: IPS

How to Build an IPS Package

Advice to Packagers

Links and Resources

Mailing List Alphabet Soup:

SVR4 pkgadd IPS SS12 DC
ON pkg(1) .py BE JET zvol
SFW SXCE pkgbuild OSOL
CBE SS11 SNV ZFS SFWNV
SPEC WOS RFE ...

Software Life Cycle

- Develop Source
- Build Binary
- Release
- **Package**
- Deploy / Install
- Run / Patch

Package Creation Steps

- Create Metadata
- Install Package Content in Staging Area
- Compile Package

Some other systems combine these steps with the binary production processes.

Example: NetBSD pkgsrc, Fink for MacOSX

Package Creation Steps

- Create Metadata
- Install Package Content in Staging Area
- Compile Package

Both the System V Packaging tools and the Image Packaging System perform only these steps; no binaries are built.

Terms and Definitions

The Old World: SysV

The New World: IPS

How to Build an IPS Package

Advice to Packagers

Links and Resources

Package Management

`pkginfo (1M)` `-- list packages`

`pkgadd (1M)` `-- add a package`

`pkgrm (1M)` `-- remove a package`

(plus some **20** more commands)

`patchadd (1M)` `-- add a patch`

`patchrm (1M)` `-- remove a patch`

(A patch is one or more packages!)

Package Format: Directory

```
<shelob:tree/bbc/sparc/5.8,19051> ls -goLF BBCssh2
total 24
drwxr-xr-x  2          5 Apr  9  2001 install/
-rw-r--r--  1        546 Apr  9  2001 pkginfo
-rw-r--r--  1       5220 Apr  9  2001 pkgmap
drwxr-xr-x  3          3 Apr  9  2001 reloc/
drwxr-xr-x  3          3 Apr  9  2001 root/
```

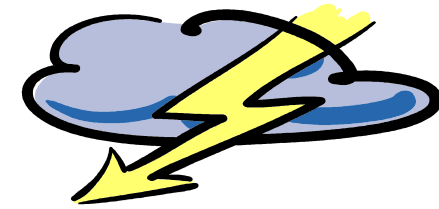
Note the `reloc` and `root` directories --
there is an implicit semantic difference!

The `pkginfo` file contains metadata about the package:

```
PKG=BBCssh2
ARCH=sparc
BASEDIR=/opt
BBC_BUILD=2
BBC_OS=5.8/sparc
BBC_VERSION=0.0
CATEGORY=application,bbc
CLASSES=none
...
OWNER=bin
...
```

The `pkgmap` file contains metadata about objects within the package:

```
: 1 8436
1 d none $PKG 0755 $OWNER $GROUP
1 d none $PKG/bin 0755 $OWNER $GROUP
1 s none $PKG/bin/scp=scp2
1 f none $PKG/bin/scp2 0555 $OWNER $GROUP 298048 15043 967227952
1 s none $PKG/bin/sftp=sftp2
...
1 f none $PKG/sbin/sshd2 0555 $OWNER $GROUP 644096 63972 967227952
1 d none /etc ? ? ?
...
1 i copyright 215 19511 967223316
1 i pkginfo 546 44021 967230548
1 i postinstall 132 10132 967224620
1 i preremove 318 26161 967224872
```



Problems

- scripts like `postinstall` run as root and can do anything
- no protection for `pkginfo` and `pkgmap`
- parameter substitution difficult to trace
- every object is an object in the file system

Package Format: Stream

Stream consists of a signature header and a concatenation of ASCII and `cpio` format files:

```
# file BBCbase-2.0.4.pkg
```

```
BBCbase-2.0.4.pkg:          package datastream
```

```
# strings BBCbase-2.0.4.pkg
```

```
# PaCkAgE DaTaStReAm
```

```
BBCbase 1 1989
```

```
# end of header
```

```
07070123a4153b000081a400000...
```



Problems

- package datastream needs to be unpacked
- confusion about package-internal and datastream compression
- `cpio` compatibility problems
- size limit of **2 Gigabytes**
- not really feasible to implement network-centric installation environments

Management of Installed Packages

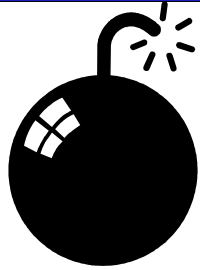
```
<shelob:/var/sadm,19068> ls -golF
```

```
...
```

```
dr-xr-xr-x    4          9 Jun 18 01:30 install/  
drwxr-xr-x    2          4 Dec 21  2007 install_data/  
drwxr-xr-x   84         85 Jun 10 00:22 patch/  
dr-xr-xr-x 1336       1336 Jun 18 01:30 pkg/
```

```
...
```

```
<shelob:/var/sadm,19069> ls -l install/contents  
-rw-r--r--    1 27041389 Jun 18 01:30 install/contents  
<shelob:/var/sadm,19070> wc -l install/contents  
  242049 install/contents  
<shelob:/var/sadm,19072> ls -ld pkg/*|wc -l  
  1334
```



Management Problems:

- package tools do not scale well; a database doesn't help
- dependency is only package-to-package, statically recorded in ASCII files
- dependencies are not resolved automatically
- there are no metapackages or package bundles except what Sun provides for the special case of Solaris installation
- a package decides about allowed number of instances
- it is difficult to re-root a package, or to manage more than one package trees on a given system, including install servers

Things SysV packages do well:

- powerful: anything root can do a package can do
- an old and true AT&T standard, well documented
- easy to compile/edit/modify
- very flexible: there is a high degree of dynamic adaptation to the Solaris installation target possible
- installed base: thousands of packages are available



Terms and Definitions

The Old World: SysV

The New World: IPS

How to Build an IPS Package

Advice to Packagers

Links and Resources

Oh My God!
They've thrown everything
away! They're rewriting
everything from scratch!
I'm ruined!



Oh My God!

They've thrown everything
away! They're rewriting
everything from scratch!
I'm ruined!



Fortunately it's not *that* bad!

Image Packaging System

- project (pkg) within the *Installation and Packaging Community* within OpenSolaris
- other projects in that community are the Caiman installer, the original SysV packaging tools, and Live Media
- IPS is development in progress, a moving target!

IPS Mission Statement: Design Goals

- software delivery must become more flexible
- operating system patching is too complicated; updating a system must be radically simpler than it is now
- software delivery focus has changed from tape and CD-ROM to a network-centric approach
- separate BEs need to be managed on one system
- the packaging system must provide the building blocks to construct new OS distributions
- old packages should be automatically converted if possible
- dependencies should be explicit; when dependencies exist during package deployment; they should be resolved automatically

IPS Mission Statement: More Goals

- be bandwidth efficient, transmit everything only once
- allow more than one package source, provide for package mirrors
- make package transmission secure and auditable
- provide an (optional) GUI for image management
- be aware of OS features (ZFS, zones)
- support virtualization, multiple install scenarios, developer deployment through partial images, nested images, per-user images

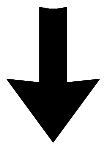
What's all this about images???

- an *image* is a collection of installed packages that **can be managed as one entity**
- a *user image* is an image that contains only **relocatable packages** (the image is not dependent on any specific location within the file system)
- a non-user image is sometimes referred to as an *entire image*

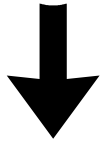
This implies that we can have many trees of installed packages in one file system, and we can manage them all using IPS.

IPS: Workflow

`pkgsend(1)` -- run by package creator



`pkg.depotd(1M)` -- daemon running on the repository server



`pkg(1)` -- run on client to retrieve packages or metadata from repository

IPS: Package Naming Scheme

The IPS packages are Fault Management Resource Identifiers (FMRI):

```
pkg://[authority]/[pkg_name]
      @[version][,build]-[branch]:[timestamp]
```

Examples:

```
pkg://bbc/SUNWbinutils@2.15,5.11-0.86:20080426T173002Z
pkg:/SUNWcar@0.5.11,5.11-0.86:20080426T182143Z
```

Authorities:

# pkg authority	URL
AUTHORITY	
bbc (preferred)	http://caiman.bb-c.de:10000/
opensolaris.org	http://pkg.opensolaris.org:80/

IPS: The `pkg(1)` Client

One command covers all usage aspects on the client to manage images and packages. It can

- install, verify, and uninstall packages
- list packages, their contents, and metadata
- create and update images
- download and refresh package catalogs from repository servers
- add, list, and delete authorities

For more information, please consult the man page...

IPS: The Configuration Database

```
<caiman:/var/pkg,752> ll -goLF
```

```
total 56
```

```
drwxr-xr-x    4          4 May 28 15:48 catalog/
-rw-r--r--    1        368 May 28 15:48 cfg_cache
drwx-----   2          2 May 28 17:52 download/
drwxr-xr-x    2          2 Apr 27 03:42 file/
drwxr-xr-x    2          2 Apr 27 03:42 index/
drwxr-xr-x 649        649 May 28 17:34 pkg/
drwxr-xr-x    7          9 May 28 14:55 repo/
```

IPS: The link to SysV Packages

The old `pkginfo(1)` sees the new packages deployed through IPS, because IPS creates “shim” directories under the `/var/sadm/pkg` directory hierarchy.

This makes it possible to have a SysV package depend on an IPS package (traditionally, many packages look for `SUNWcsr` and friends as a sanity check).

However, `pkginfo(1)` does not create any entries for individual package objects in the `/var/sadm/install/contents` file, so all the other SysV package commands do not really work.

Terms and Definitions

The Old World: SysV

The New World: IPS

How to Build an IPS Package

Advice to Packagers

Links and Resources

First, we need to enable the repository server:

```
svccfg -s pkg/server "setprop pkg/port=10000"  
svccfg -s pkg/server "setprop pkg/inst_root=/var/repo"  
svcadm refresh pkg/server  
svcadm enable pkg/server  
svcadm restart pkg/server
```

Check for correct configuration using the `svccprop(1)` and `svcs(1)` commands.

Then create some content and a description file:

```
# cat /tmp/mypkg.manifest
set name="pkg.name"          value="hello -- a hello world script"
set name="pkg.description"  value="This script prints a nice
  greeting message."
set name="maintainer"      value="Volker A. Brandt <vab@bb-
  c.de>"
set name="upstream"        value="support@bb-c.de"

dir mode=0755 owner=root group=bin path=/opt
dir mode=0755 owner=root group=bin path=/opt/local
dir mode=0755 owner=root group=bin path=/opt/local/bin

file hello.sh mode=0555 owner=bin group=bin path=/opt/local/bin/
  hello
```

The entries in this file are called *actions*. Every atomic operation on an IPS package object is an action. For complete details, see the `pkg(5)` man page.

Next, send the package to the repository:

```
> pkgsend open BBChello@1.0
export PKG_TRANS_ID=1214255862_pkg%3A%2FBBChello%401.0...
> setenv PKG_TRANS_ID 1214255862_pkg%3A%2FBBChello%401.0...
> pkgsend include mypkg.manifest
> pkgsend close
PUBLISHED
pkg:/BBChello@1.0,5.11:20080623T231742Z
```

Verify the existence of the package in the repository:

```
> pkg info -r pkg:/BBChello
      Name: BBChello
      Summary:
      State: Not installed
      Authority: bbc (preferred)
      Version: 1.0
      Build Release: 5.11
      Branch: None
      Packaging Date: Mon Jun 23 23:17:42 2008
      Size: 32 B
      FMRI: pkg:/BBChello@1.0,5.11:20080623T231742Z
```

Finally, install the package in a new image for testing:

```
# pkg install -v BBChello
```

```
Before evaluation:
```

```
UNEVALUATED:
```

```
+pkg:/BBChello@1.0,5.11:20080623T231742Z
```

```
After evaluation:
```

```
None -> pkg:/BBChello@1.0,5.11:20080623T231742Z
```

```
None
```

```
DOWNLOAD
```

```
PKGS
```

```
FILES
```

```
XFER (MB)
```

```
Completed
```

```
1/1
```

```
1/1
```

```
0.00/0.00
```

```
PHASE
```

```
ACTIONS
```

```
Install Phase
```

```
7/7
```

```
# ls -goLF /opt/local/bin
```

```
total 2
```

```
-r-xr-xr-x  1          32 Jun 24 00:07 hello*
```

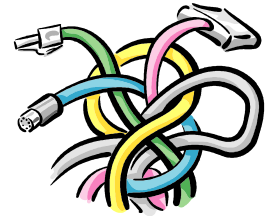
As an alternative, import a SysV package:

```
> pkgsend open BBCcert@1.0.0
export PKG_TRANS_ID=1214263101_pkg%3A%2FBBCcert%401.0.0%2C5...
> setenv PKG_TRANS_ID 1214263101_pkg%3A%2FBBCcert%401.0.0%2C5...
> pkgsend import BBCcert-1.0.0-any-20070605132603.pkg
> pkgsend close
PUBLISHED
pkg:/BBCcert@1.0.0,5.11:20080624T011821Z
# pkg refresh --full
```

Again, verify the operation:

```
# pkg info -r pkg:/BBCcert
      Name: BBCcert
      Summary:
      State: Not installed
      Authority: bbc (preferred)
      Version: 1.0.0
      Build Release: 5.11
      Branch: None
      Packaging Date: Tue Jun 24 01:18:21 2008
      Size: 11 kB
      FMRI: pkg:/BBCcert@1.0.0,5.11:20080624T011821Z
```

What's missing?



- there is no good way to replicate a repository; for example, if we want a mirror of `pkg.opensolaris.org`
- many of the SysV packages features are silently ignored when importing an existing package into an IPS repository
- **there is no way to run a script; thus, many many things cannot be done that used to be very easy**

Terms and Definitions

The Old World: SysV

The New World: IPS

How to Build an IPS Package

Advice to Packagers

Links and Resources

Advice #1: Use manifest files and scripts

Don't type in anything directly, IPS is too complex for that. Always use scripts and manifest files that can be version-controlled and rolled back. A small typo can make a big difference.



Advice #2: Learn some Python basics

In their infinite wisdom, our Sun friends implemented IPS in *Python* instead of a real scripting language like *Perl*. Knowing some Python helps a lot when facing error messages like:

```
Creating Plan -pkg: install failed: ['Duplicate actions',  
[ (('set', 'value'),  
set([<pkg.actions.attribute.AttributeAction object at  
0x898d36c>, <pkg.actions.attribute.AttributeAction object  
at 0x898d6ac>, <pkg.actions.attribute.AttributeAction  
object at 0x898d02c>,  
<pkg.actions.attribute.AttributeAction object at  
0x898d42c>, <pkg.actions.attribute.AttributeAction object  
at 0x898d10c>, <pkg.actions.attribute.AttributeAction  
object at 0x898d1ec>])])]]
```

Advice #3: Remember, it's Work in Progress



Don't trust older presentations about IPS. Some commands are not implemented yet, argument names have changed and probably will change more. When in doubt about a specific functionality, check the source.

Advice #4: Don't try to build a package that works in both worlds



It is currently impossible to reproduce SysV package functionality in IPS. We could isolate all script-like functionality in one place and hope that it would be easy to port to IPS, but that is just a wish for the future.

So: **The package man always builds twice.**

Terms and Definitions

The Old World: SysV

The New World: IPS

How to Build an IPS Package

Advice to Packagers

Links and Resources

OpenSolaris Installation and Packaging community:

<http://opensolaris.org/os/community/install/>

OpenSolaris Image Packaging System Project:

<http://opensolaris.org/os/project/pkg/>

Source Code (Mercurial Repository):

```
hg clone ssh://anon@opensolaris.org/hg/pkg/gate
```

The Author's Link List for this Presentation:

<http://www.bb-c.de/osdevcon2008/>

Questions and Discussion



Thank you!

Volker A. Brandt

Brandt & Brandt Computer GmbH

vab@bb-c.de