Creating SYS-V style packages and patches

Malcolm Herbert
System Administrator
Unico Computer System

Why create packages?

- consolidate local host changes
- maintain parity between hosts
- help stage software rollout
- as an aid to disaster recovery
- use package database as tripwire
- integration with Sun tools

Why create patches?

as previously, but also:

- well-defined change control
- roll-back support
- minimise admin upgrade effort

SYS-V package system

- Simple database of file perms and owners in `/var/sadm/install/contents`
- Individual package meta-data inside `/var/sadm/pkg/`*`pkgname`*

SYS-V package dir format

```
pkgname/
  pkgmap
  pkginfo
  reloc/   or   root/
    content
  install/
    scripts
```

SYS-V package metadata files

`pkgmap` - package file permissions, ownerships and checksums

`pkginfo` - package metadata

`reloc` or **`root`** - package content

`install` - package management, pre- and post- install scripts

SYS-V package stream format

- create/unroll with `pkgtrans`
- single file for all content
- can contain multiple packages
- easily compressed/transportable

Creating a SYS-V package

simple example to install into `/usr/local`:

```
bin/foo
etc/foo.conf
lib/Foo.pm
```

Creating a SYS-V package

Our build area `/data/devel/foo-1.0.0:`

```
root/
  package content …
skel/
  install/
    install scripts ...
spool/
  scratch area ...
include/
  package metadata ...
```

Creating a SYS-V package

include/prototype:

```
i pkginfo=@PWD@/spool/pkginfo
i checkinstall=@PWD@/spool/install/checkinstall
```

Creating a SYS-V package

include/pkginfo:

```
PKG=@PKG@
NAME=foo package
CATEGORY=application
CLASSES="none"
ARCH=sparc
VERSION=@VER@
VENDOR=mjch
HOTLINE=none
EMAIL=mjch@mjch.net
PSTAMP=@DATE@
BASEDIR=/usr/local
```

Creating a SYS-V package

/data/devel/foo-1.0.0/create.sh:

```
#!/bin/sh
pkg='MJHfoo'
ver='1.0'
pwd=`pwd`
date=`date '+%Y-%m-%d %H:%M'`
umask 0
rm -rf spool/*
cp -r skel/* spool
( cat include/prototype ; pkgproto ${pwd}/root= ) \
| sed -e "s#@PWD@#${pwd}#g" > spool/prototype
```

Creating a SYS-V package

`/data/devel/foo-1.0.0/create.sh` continued:

```
cat include/pkginfo \
| sed -e "s#@PKG@#${pkg}#g" \
      -e "s#@DATE@#${date}#g" \
      -e "s#@VER@#${ver}#g" > spool/pkginfo

pkgmk -b ${pwd}/root \
  -f ${pwd}/spool/prototype \
  -d ${pwd}/spool -o ${pkg}
```

Creating a SYS-V package

```
# sh ./create.sh
## Building pkgmap from package prototype file.
## Processing pkginfo file.
## Attempting to volumize 6 entries in pkgmap.
part  1 -- 18 blocks, 9 entries
## Packaging one part.
/data/devel/foo-1.0.0/spool/MJHfoo/pkgmap
/data/devel/foo-1.0.0/spool/MJHfoo/pkginfo
/data/devel/foo-1.0.0/spool/MJHfoo/reloc/bin/foo
/data/devel/foo-1.0.0/spool/MJHfoo/install/checkinstall
/data/devel/foo-1.0.0/spool/MJHfoo/reloc/etc/foo.conf
/data/devel/foo-1.0.0/spool/MJHfoo/reloc/lib/Foo.pm
## Validating control scripts.
## Packaging complete.
Transferring <MJHfoo> package instance
```

Creating a SYS-V package

```
# pkgadd -d ./spool MJHfoo
Processing package instance <MJHfoo> from
</data/devel/foo-1.0.0/spool>
foo package(sparc) 1.0
## Executing checkinstall script.
Using </usr/local> as the package base directory.
## Processing package information.
## Processing system information.
   1 package pathname is already properly installed.
## Verifying disk space requirements.
## Checking for conflicts with packages already
installed.
```

Creating a SYS-V package

```
Installing foo package as <MJHfoo>
## Installing part 1 of 1.
/usr/local/bin/foo
/usr/local/etc/foo.conf
/usr/local/lib/Foo.pm
[ verifying class <none> ]
Installation of <MJHfoo> was successful.
# /usr/local/bin/foo
This is foo 1.0.0, library 1.0.0
/usr/local/etc/foo.conf:
> This is bar.conf
> version 1.0.0
```

Creating a SYS-V package

To create a gzipped package stream:

```
pkgtrans -os ${pwd}/spool \
   ${pwd}/spool/${pkg}.pkg ${pkg}

gzip ${pwd}/spool/${pkg}.pkg
```

Creating a SYS-V package

```
# pkginfo -l MJHfoo
   PKGINST:  MJHfoo
      NAME:  foo package
  CATEGORY:  application
      ARCH:  sparc
   VERSION:  1.0
   BASEDIR:  /usr/local
    VENDOR:  mjch
    PSTAMP:  2009-05-17 18:32
  INSTDATE:  May 17 2009 18:36
   HOTLINE:  none
     EMAIL:  mjch@mjch.net
    STATUS:  completely installed
     FILES:        6 installed pathnames
                   3 directories
                   1 executables
                   3 blocks used (approx)
```

SYS-V patches

- package system explicitly allows re-install of over existing content, if versions match
- patches contain 'sparse' packages with new content and same version as the original

Sun's patch philosophy

- all changes required for a single bug fix collected into one patch
- patch may contain updates to more than one package
- when later fixes touch the same file, patches are incorporated and obsoleted

Sun's patch philosophy issues

- patches may affect host more than desired
- complex dependency analysis required
- individual patches may grow quite large - eg, kernel jumbo patch

Creating a SYS-V patch

Our build area `/data/devel/foo-1.0.1:`

```
root/
  package content ...
skel/
  install/
    install scripts ...
  root/
    patch metadata ...
spool/
  scratch area ...
include/
  package metadata ...
```

SYS-V patch metadata files

`patchinfo` – patch metadata

`pkgmap` - package file permissions, ownerships and checksums

`pkginfo` - package metadata

`reloc` or **`root`** - package content

`install` - package management, pre- and post- install scripts

Creating a SYS-V patch

include/prototype:

```
i checkinstall=@PWD@/spool/install/checkinstall
i i.none=@PWD@/spool/install/i.none
i patch_checkinstall=@PWD@/spool/install/patch_checkinstall
i patch_postinstall=@PWD@/spool/install/patch_postinstall
i postinstall=@PWD@/spool/install/postinstall
i preinstall=@PWD@/spool/install/preinstall
i pkginfo=@PWD@/spool/pkginfo
```

Creating a SYS-V patch

include/pkginfo:

```
PKG=@PKG@
NAME=foo package
CATEGORY=application
CLASSES="man none"
ARCH=sparc
VERSION=@VER@
VENDOR=mjch
HOTLINE=none
EMAIL=mjch@mjch.net
PSTAMP=@DATE@
BASEDIR=/usr/local
SUNW_PKGVERS=1.0
SUNW_PATCHID=@PATCHID@
SUNW_INCOMPAT=
SUNW_OBSOLETES=
SUNW_REQUIRES=
```

Creating a SYS-V patch

include/patchinfo:

```
PATCHINFOVERSION="1.0"
PATCHID=@PATCHID@
PATCH_CORRECTS='@PKG@'
PATCH_ARCH='sparc'
PATCH_OS='SunOS'
PATCH_OSRELEASE='5.10'
PATCH_PROPERTIES='clientusr'
```

Creating a SYS-V patch

Scripts inside `skel/install/bin` are Sun-supplied scripts which do the actual heavy-lifting of applying the patch packages

Current copies of these can be found inside recent patches ... not sure of the legality of using these, although technically it all seems to Just Work

Creating a SYS-V patch

/data/devel/foo-1.0.1/create.sh:

```
#!/bin/sh
pkg='MJHfoo'
ver='1.0'
rev='01'
pwd=`pwd`
date=`date '+%Y-%m-%d %H:%M'`
patchid="${pkg}-${rev}"
umask 0
rm -rf spool/*
cp -r skel/* spool
mv spool/root/README spool/root/README.${patchid}
mv spool/root spool/${patchid}
```

Creating a SYS-V patch

/data/devel/foo-1.0.1/create.sh continued:

```
cat include/pkginfo \
| sed -e "s#@PKG@#${pkg}#g" \
      -e "s#@DATE@#${date}#g" \
      -e "s#@PATCHID@#${patchid}#g" \
      -e "s#@VER@#${ver}#g" > spool/pkginfo

pkgmk -b ${pwd}/root \
  -f ${pwd}/spool/prototype \
  -d ${pwd}/spool/${patchid} -o ${pkg}
```

Creating a SYS-V patch

To create a zipped patch:

```
zip -r ${patchid}.zip ${patchid}
```

Creating a SYS-V patch

```
# patchadd spool/MJHfoo-01
Validating patches...
Loading patches installed on the system...
Done!
Loading patches requested to install.
Done!
Checking patches that you specified for installation.
Done!
Approved patches will be installed in this order:
MJHfoo-01
Checking installed patches...
Verifying sufficient filesystem capacity (dry run method)...
Installing patch packages...
Patch MJHfoo-01 has been successfully installed.
See /var/sadm/patch/MJHfoo-01/log for details
Patch packages installed:
  MJHfoo
```

Creating a SYS-V patch

```
# /usr/local/bin/foo
This is foo 1.0.0, library 1.0.1
/usr/local/etc/foo.conf:
> This is foo.conf
> version 1.0.1
# showrev -p | grep MJH
Patch: MJHfoo-01 Obsoletes:  Requires:  Incompatibles:
 Packages: MJHfoo
# pkgparam -v MJHfoo PKGINST VERSION PSTAMP
PKGINST='MJHfoo'
VERSION='1.0'
PSTAMP='2009-05-17 19:17'
```

Creating a SYS-V patch

```
# patchrm MJHfoo-01
Validating patches...
Loading patches installed on the system...
Done!
Checking patches that you specified for removal.
Done!
Approved patches will be removed in this order:
MJHfoo-01
Checking installed patches...
Backing out patch MJHfoo-01...
Patch MJHfoo-01 has been backed out.
# /usr/local/bin/foo
This is foo 1.0.0, library 1.0.0
/usr/local/etc/foo.conf:
> This is bar.conf
> version 1.0.0
```

Questions?

These notes and a tarball of example packages and patches can be found at

http://www.mjch.net/pub/talks/sysv-pkg-patch

Creating SYS-V style packages and patches

Malcolm Herbert
System Administrator
Unico Computer System