Cleaning up OpenIndiana repository

Contents

> ⓘ **NOTE:**
> This is an internal document provided mostly for OpenIndiana maintainers

Due to current CI practices, OpenIndiana Hipster repository tend to bloat with old package versions. This causes several issues:

- `pkg(1)` starts consuming a lot of memory during planning stage;
- refreshing repository index becomes the longest step in Jenkins build jobs;
- storage space is wasted on pkg and build servers.

To fix these issues currently periodic manual cleanup is necessary. Note that during cleanup we preserve only latest versions of packages. Given that it's usually just 10-20% of all published packages, using `pkgrepo remove` is impractical.

The cleanup should be done on two sides: on the side of pkg server (pkg.openindiana.org) and on build server (hipster.openindiana.org) to avoid resyncing old package versions from build server to the pkg depot server.

Be sure to disable Jenkins jobs prior to the operation.

As transfering packages takes significant time (about an hour on build server and several hours on pkg server), you likely want to perform operations on both of them simultaneously.

1   Tasks held on the build server

We should cleanup packages produced by illumos-gate and oi-userland Jenkins jobs.

We generally don't care about packages, produced by illumos-gate build job. They can be simply removed. Remove them from build area:

```
cd /jenkins/jobs/illumos-gate/workspace/components/openindiana/illumos-gate/
rm -fr build/i386/pkgre*
rm -fr illumos-gate/packages/i386/nightly*
```

and repository:

```
cd /jenkins/jobs/illumos-gate/workspace/i386/repo/
rm -fr publisher/openindiana.org/*
```

Packages, produced by oi-userland build job, are used to generate userland-incorporation, so we can't just drop them. We create new repository and populate it with latest packages. This step takes some time, so be sure to run it in screen or tmux environment.

```
cd /jenkins/jobs/oi-userland/workspace/i386/repo/
mkdir repo.new
cp repo/pkg5.repository repo.new/
pkgrecv -s repo --newest | tee pkg-list
cat pkg-list | split -l 100
for i in $(ls x*); do pkgrecv -s repo -d repo.new $(cat $i); done
```

Check that resulting repo contains all packages from old repository (diff should be empty).

```
pkgrecv -s repo --newest > /tmp/old.packages
pkgrecv -s repo.new --newest > /tmp/new.packages
diff -u /tmp/old.packages /tmp/new.packages
```

If diff is not empty, investigate the issue and use `pkgrecv` to transfer missing packages individually.

When everything is fine, perform final cleanup.

```
pkgrepo -s repo.new rebuild
rm x* pkg-list
```

Now you are ready to replace old repository with the new one.

```
rm -fr repo
mv repo.new repo
```

2   Tasks held on the package server

The procedure is essentially the same, we create new repository and transfer latest packages and custom scripts from the old repository to the new one. Scripts mentioned here are partially used by rsync jobs to update repository and partially can be used by pkg server administrator to perform repository maintenance.

You should start by creating new filesystem for the repository. File systems on the pkg server has UUID in its name, it's replaced by literal UUID here.

```
zfs create zones/UUID/data/repos/pkg.openindiana.org/hipster-new
```

Copy necessary scripts and files from old repository:

```
cp -r /data/repos/pkg.openindiana.org/hipster/{pkg5.repository,scripts,add-
↪  content.sh}
↪  /data/repos/pkg.openindiana.org/hipster-new/
```

Make necessary directories:

```
cd /data/repos/pkg.openindiana.org/hipster-new
mkdir .empty .temp .temp/.openindiana.org
```

Transfer data from current repository (the script essentially does the same batched pkgrecv as we've done on the build server):

```
bash scripts/create-new-repo.sh
```

Check that resulting repo contains all packages from old repository (diff should be empty).

```
pkgrecv -s /data/repos/pkg.openindiana.org/hipster-new/ --newest > /tmp/new.packages
pkgrecv -s /data/repos/pkg.openindiana.org/hipster/ --newest > /tmp/old.packages
diff -u /tmp/old.packages /tmp/new.packages
```

If diff is not empty, investigate the issue and use `pkgrecv` to transfer missing packages individually.

When everything is fine, perform final cleanup.

```

```
pkgrepo -s . rebuild
rm x* pkg-list
```

Make pkg server actually use new repository. Ensure that you minimise time between disabling pkg server and enabling it again - during this interval it's evidently unusable to the clients.

```
cd /
svcadm disable svc:/application/pkg/server:hipster
zfs rename zones/UUID/data/repos/pkg.openindiana.org/hipster
 ↪  zones/UUID/data/repos/pkg.openindiana.org/hipster-old
zfs rename zones/UUID/data/repos/pkg.openindiana.org/hipster-new
 ↪  zones/UUID/data/repos/pkg.openindiana.org/hipster
svcadm enable svc:/application/pkg/server:hipster
```

3   Final steps and testing

Don't forget to enable Jenkins jobs, which you've disabled before cleanup.

- Visit http://pkg.openindiana.org/hipster/ and check that search works correctly.
- Run some Jenkins build job to ensure that interaction between hipster.openindiana.org and pkg.openindiana.org works correctly.
- Don't forget to destroy `zones/UUID/data/repos/pkg.openindiana.org/hipster-old` filesystem in several days after cleanup.