Creating a Pbulk Chroot

Contents

There are many ways to go about building binary packages and many sets of instructions for doing so.

The approach we use is designed to be as simple and safe as possible. Below are instructions for setting up the safest possible pbulk package-building environment using CentOS Linux or NetBSD.

One of the issues you will run into eventually while building packages is "leakage". For example, sometimes a poorly written configure script with hard-coded search paths might slip through QA and cause pkgsrc to use a build tool or library installed by Yum or other means.

The pkgsrc build system does a fabulous job isolating itself from other software installed on your system and leakage is rare. It's not perfect, though. For this reason, binary packages (which must work on any installation of the same OS, not just the one on which they're built) should be built on a pristine system with no unnecessary software installed. This will ensure that they work properly on other machines and don't depend on anything outside pkgsrc. This is most easily accomplished using a virtual machine, chroot, or other container.

Here's how to accomplish this easily with a simple chroot configuration. You could just as easily use a dedicated machine or a virtual machine for your pbulk builds, but the chroot approach entails the lowest overhead of these methods.

More discussion on pkgsrc in scientific computing can be found at https://uwm.edu/hpc/software-management/.

1. Create a minimal system image:

   (a) Install the OS. Use the exact same operating system and version as the system that will be used for package building, since we will be using this image in a chroot environment. We perform this step under VirtualBox.

   - CentOS: Do a minimal CentOS install, from the CentOS minimal ISO. DO NOT INSTALL ANY ADDITIONAL YUM PACKAGES.
   - NetBSD: Do a minimal NetBSD install including the development tools set, pkgin, and optionally the X11 set if you want to build packages that depend on the base X11. DO NOT INSTALL ANY ADDITIONAL PACKAGES USING pkg_add OR pkgin.

   (b) Within the new installation as the root user:

      i. Make sure you're OS image is up-to-date:
         - CentOS: yum update -y && shutdown -r now
         - NetBSD: Update according to the docs if desired.

      ii. Create a tarball of the entire installation in os-version.tgz, e.g. centos-6.tgz or netbsd-7.tgz.

```
cd /
mkdir tar
tar zcvf tar/os-version.tgz \
    --exclude ./tar --exclude ./proc --exclude ./dev --exclude ./kern .
```

      iii. Save this pristine system image in a safe place for repeated future deployments.

      iv. Copy os-version.tgz to the system you plan to use for pbulk builds, e.g. using scp.

2. (a) On the system you will use for bulk builds: Install pbulk-setup, pbulk-new-chroot and pbulk-start-chroot into a directory in your PATH.
   These links contain the latest development versions of the scripts. Recent, stable versions can also be installed via wip/pkg-dev if you have pkgsrc installed.

   (b) Run

```
pbulk-new-chroot os-version.tgz name-of-new-chroot-directory
```

   and follow the instructions on the screen. This will unpack os-version.tgz into the new directory, and prepare it for doing bulk builds by installing a few essential packages and running pbulk-setup.
   When pbulk-setup completes, it will show you the commmands to run inside the chroot to bootstrap pbulk and run a bulk build.

3. Start the chroot and bootstrap the pbulk setup by running the command shown by pbulk-setup in the previous step.

(a)
```
pbulk-start-chroot name-of-new-chroot-directory
```

In the chroot:

```
pbulk-prefix/bootstrap
```

(b) Edit pbulk-prefix/pbulk.conf and pbulk-prefix/pbulk.list. Note that pbulk.conf is simply a Bourne shell script that it sourced by pbulk, so later commands override earlier ones.

(c) Run the bulk build:

```
pbulk-prefix/bin/bulkbuild
```

This may take a long time, so if you're running on a remote server via ssh, you may want to protect it from dropped connections by using nohup:

```
nohup pbulk-start-chroot pbulk-prefix/bin/bulkbuild > build.stdout 2> build.stderr  ↩
    &
```

4. You will want to keep your image up-to-date by periodically starting the chroot, updating the OS within it, and recreating the tarball.

   Example:

```
pbulk-start-chroot chroot-centos-7-current
yum update
exit
cd chroot-centos-7-current
tar zcvf ../centos-7.tgz .
```