RELEASE PROCESS

1. BUILD ENVIRONMENT

There are multiple build envrionments, debian based packages are built via pbuilder for multiple distributions. The build hosts are listed in the `deb_hosts` file, and the list of distributions are in `deb_dist`. All distributions are build on each of the build hosts. Currently there is 1 64 bit and 1 32 bit build host.

The RPM based packages are are built natively, so one distribution per build host. The list of hosts is found in `rpm_hosts`.

Prior to building, it's necessary to update the pbuilder seed tarballs:

```
./update_all_pbuilders.sh
```

2. SETUP KEYRING FOR SIGNING PACKAGES

```
export GNUPGHOME=<path to keyring dir>

# verify it's accessible
gpg --list-keys
```

The release key should be present:

```
pub   4096R/17ED316D 2012-05-20
uid                  Ceph Release Key <sage@newdream.net>
```

3. SET UP BUILD AREA

Clone the ceph and ceph-build source trees:

```
git clone http://github.com/ceph/ceph.git
git clone http://github.com/ceph/ceph-build.git
```

In the ceph source directory, checkout next branch (for point releases use the testing branch):

```
git checkout next
```

Checkout the submodules:

```
git submodule update --init
```

4. UPDATE BUILD VERSION NUMBERS

Substitute the ceph release number where indicated below by the string 0.xx:

Edit configure.ac and update the version number. Example diff:

```
-AC_INIT([ceph], [0.54], [ceph-devel@vger.kernel.org])
+AC_INIT([ceph], [0.55], [ceph-devel@vger.kernel.org])
```

Update the version number in the debian change log:

```
DEBEMAIL user@host dch -v 0.xx-1
```

Commit the changes:

```
git commit -a
```

Tag the release:

```
../ceph-build/tag-release v0.xx
```

5. CREATE MAKEFILES

The actual configure options used to build packages are in the `ceph.spec.in` and `debian/rules` files. At this point we just need to create a Makefile.:

```
./do_autogen.sh
```

6. RUN THE RELEASE SCRIPTS

This creates tarballs and copies them, with other needed files to the build hosts listed in deb_hosts and rpm_hosts, runs a local build script, then rsyncs the results back tot the specified release directory.:

```
../ceph-build/do_release.sh /tmp/release
```

7. CREATE RPM REPO

Copy the rpms to the destination repo, creates the yum repository rpm and indexes.:

```
../ceph-build/push_to_rpm_repo.sh /tmp/release /tmp/rpm-repo 0.xx
```

8. CREATE DEBIAN REPO

The key-id used below is the id of the ceph release key from step 2:

```
mkdir /tmp/debian-repo
../ceph-build/gen_reprepro_conf.sh /tmp/debian-repo key-id
../ceph-build/push_to_deb_repo.sh /tmp/release /tmp/debian-repo 0.xx main
```

9. PUSH REPOS TO CEPH.ORG

For a development release:

```
rcp ceph-0.xx.tar.bz2 ceph-0.xx.tar.gz \
    ceph_site@ceph.com:ceph.com/downloads/.
rsync -av /tmp/rpm-repo/0.xx/ ceph_site@ceph.com:ceph.com/rpm-testing
rsync -av /tmp/debian-repo/ ceph_site@ceph.com:ceph.com/debian-testing
```

For a stable release, replace {CODENAME} with the release codename (e.g., `argonaut` or `bobtail`):

```
rcp ceph-0.xx.tar.bz2 \
    ceph_site@ceph.com:ceph.com/downloads/ceph-0.xx{CODENAME}.tar.bz2
rcp ceph-0.xx.tar.gz  \
```

```
    ceph_site@ceph.com:ceph.com/downloads/ceph-0.xx{CODENAME}.tar.gz
rsync -av /tmp/rpm-repo/0.xx/ ceph_site@ceph.com:ceph.com/rpm-{CODENAME}
rsync -auv /tmp/debian-repo/ ceph_site@ceph.com:ceph.com/debian-{CODENAME}
```

10. UPDATE GIT

DEVELOPMENT RELEASE

For a development release, update tags for `ceph.git`:

```
git push origin v0.xx
git push origin HEAD:testing
git checkout master
git merge next
git push origin master
git push origin HEAD:next
```

Similarly, for a development release, for both `teuthology.git` and `ceph-qa-suite.git`:

```
git checkout master
git reset --hard origin/master
git branch -f testing origin/next
git push -f origin testing
git push -f origin master:next
```

STABLE RELEASE

For `ceph.git`:

    git push origin stable

POINT RELEASE

Just push the new tag:

    git push origin v0.xx