1 What's up with pbuilder?

This is a document to explain what is pbuilder, what has been happening recently with pbuilder, and what will probably happen in the near future.

1.1 Concepts of using pbuilder

pbuilder facilitates clean-room environment testing of package building through using a chroot image and extracting a fresh chroot image for every build.

There are several simple commands for operation. `pbuilder create`, `pbuilder update`, and `pbuilder build`[1] commands are the typical commands used. If you need more details, see pbuilder manual, `/usr/share/doc/pbuilder/pbuilder-doc.html`

When everything is set, `pbuilder build` will accept `.dsc` file (the Debian source-package) and build the package inside the chroot.
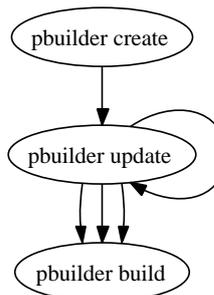
operation	operation frequency	context
create	once per base.tgz	create the base filesystem
update	once per day (unstable update)	update the base filesystem
build	once per package	build the Debian package inside chroot

Let me examine a typical sequence of events in a day of a Debian Developer.

---

[1]pdebuild command is often more convenient
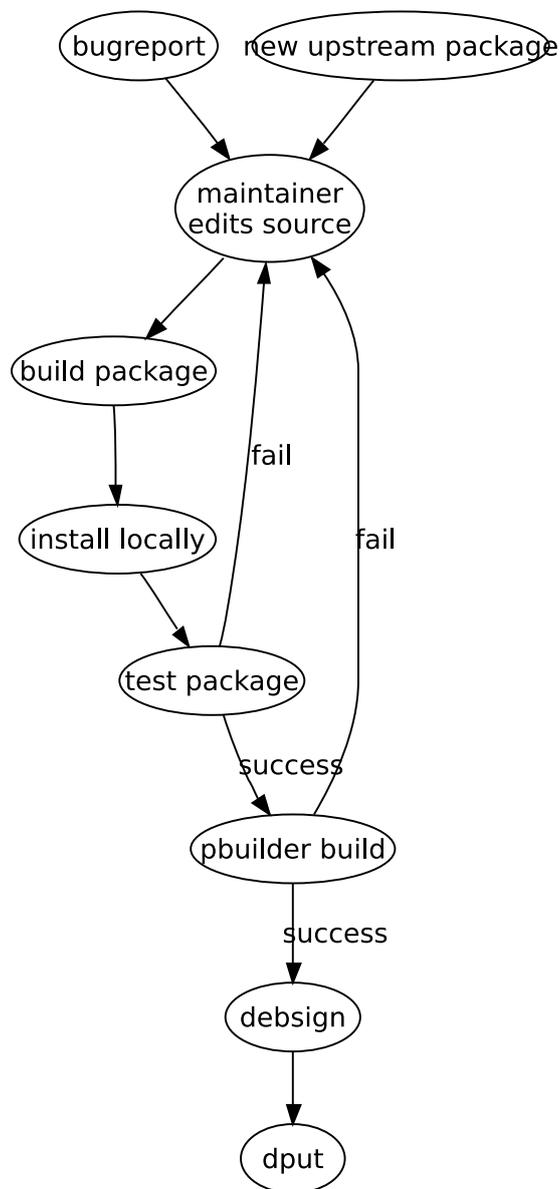
pbuilder is built in to the process as a checking system to ensure package quality. [2] This is useful for initial testing of Build-Dependency, and as basic regression testing framework.

1.2 pbuilder Development structure

Let me mention a little bit about how pbuilder itself is developed. Currently pbuilder is co-maintained using resources provided by Alioth. Development is

---

[2]This is an example. Some people do not build package locally and entirely work inside chroots.

done mostly by Loïc Minier and Junichi Uekawa, and occasional commits from Matt Kraai, and Mattia Dongili.

Project page is available at `http://alioth.debian.org/projects/pbuilder` and home page is available at, `http://pbuilder.alioth.debian.org/` which provides the pbuilder manual.

git is used for source code management, and the repository can be checked out by any of the following commands[3].

```
git-clone git://git.debian.org/git/pbuilder/pbuilder.git
git-clone http://git.debian.org/git/pbuilder/pbuilder.git
git-clone ssh://git.debian.org/git/pbuilder/pbuilder.git
```

1.3 Derivatives and their status

There are several derivatives of pbuilder supporting different 'backends'. They use different methods for providing a clean-room environment. Let me explain some of them.

1.3.1 LVM port

Someone did a port to using LVM snapshot for base.tgz-management. There was some e-mailing, but not quite gone into pbuilder. LVM method still uses chroot as the method for segregation. The advantage is that LVM snapshot process is much faster than extracting a tar archive for base image.

1.3.2 user-mode-linux port

pbuilder-uml port exists. Apparently, it is working for most people. Mattia Dongili and others have been actively working on this port.

base.tgz extraction is replaced with UML cow device support, and thus is faster. chroot is replaced with user-mode-linux session, which makes system calls a bit slower.

1.3.3 cowdancer port

Junichi Uekawa has been working on the port to cowdancer since 2005, and it is somewhat stable. This replaces base.tgz extraction with `cp -la` , which is much faster.

Because of the way cowdancer hooks libc calls like open/close, it may interfere with package building. [4]

1.3.4 qemu port

Junichi Uekawa has been working on the port to qemu/kqemu/kvm since early 2007. QEMU has COW block device support, so it eliminates the need to extract base.tgz.

qemu port has an interesting advantage in that it adds cross-architecture-building support for pbuilder. I can potentially build and test ARM packages on my i386 box.

---

[3]ssh access requires account on alioth
[4]etch release unfortunately was faced with Bug 413912

1.4 Further development ideas

1.4.1 Installation testing

There are other projects such as piuparts which can be leveraged by pbuilder.

pbuilder does provide example script for testing installation: `/usr/share/doc/pbuilder/examples/execute_installtest.sh`.

```
pbuilder execute \
  /usr/share/doc/pbuilder/examples/execute_installtest.sh \
  pbuilder
```

This command will try to install the package using apt-get into the chroot.

1.4.2 Package testing

Package testing is a feature that is usually useful, especially since developer time is limited, and repeated manual testing is no fun. pbuilder includes an example hook script `/usr/share/doc/pbuilder/examples/B92test-pkg` which will test package after a successful build.

The test files are shell scripts placed in `debian/pbuilder-test/NN_name` (where NN is number) following run-parts standard for file names [5].

1.4.3 aptitude

pbuilder exclusively depends on apt-get. It might be time to look at what is missing to get aptitude working.

1.4.4 apt-key support

pbuilder currently lacks apt-key support. Since apt-key support is definitely available in the current stable releases, it is about time to start considering supporting apt-key.

1.4.5 build-dependency parser

Build-dependency parser has been somewhat old and suboptimal. Loïc Minier has been working on refreshing it.

1.4.6 buildd.net-like support

pbuilder creates a bunch of useful build logs, but it lacks the notion of history. It does not have an infrastructure to aggregate and put them to use. [6] Collecting past build logs even locally and comparing them between builds may alert problems, used in conjunction with tools like debdiff.

---

[5]no '.' in file names!

[6]Some projects to create `pbuildd` did exist, but I am not sure if it still exists today.

1.5   References

- `http://pbuilder.alioth.debian.org/` or `/usr/share/doc/pbuilder/`
  `pbuilder-doc.html`: pbuilder manual

- cowdancer package

- piuparts package

- autodebtest: Ubuntu automatic testing system

- schroot / dchroot

- buildd