Automated Testing of Debian Packages
Status Update

Lucas Nussbaum – lucas@debian.org
Summary

1. Introduction
2. Tests
3. Building packages more efficiently
4. Piuparts and false positives
5. State of the archive
6. Collab-qa project
7. Conclusion
Summary

1 Introduction

2 Tests

3 Building packages more efficiently

4 Piuparts and false positives

5 State of the archive

6 Collab-qa project

7 Conclusion
Introduction

At the end of the etch release cycle, quite a lot of QA was done:

- Several builds of all packages in etch
- Several piuparts runs on all packages in etch

⇒ about 200 RC bugs filed and fixed in etch
Such tests are a good thing

- give the **same level of attention to all packages** in Debian
- not only rely on humans to find bugs
- avoid regressions
- keep maintainers busy :-(
Such tests are a good thing, but ...

- They were run too late in the release process
- They caused some packages to miss etch
- Lots of things weren’t tested

⇒ We need to be more efficient/organized during the lenny cycle
Summary

1. Introduction
2. Tests
   - Rebuilding packages
   - Piuparts
3. Building packages more efficiently
4. Piuparts and false positives
5. State of the archive
6. Collab-qa project
Tests

- Rebuilding packages from source
- Piuparts runs
- Other tests: lintian, linda, ...
Rebuilding packages

- packages with "Arch : all" are only built on the developer’s machine
- packages with "Arch : any" are only built automatically before they reach unstable (and only on $ARCH != Uploader’s arch)

After that, the **build environment changes**:
  - newer/older compiler and libraries
  - build-dependencies not available anymore (b-deps are not considered for testing propagation)

Problems:
  - Everyone should be able to build your package
  - Stable releases must be self-contained (security updates !)
Rebuilding packages: tools

pbuilder:
- builds a package inside a chroot
- very easy to set up
- you should use it!
- talk on saturday afternoon

sbuild (the Debian package):
- relies on schroot
- a bit harder to set up, but more powerful
Piuparts

Tests installation and removal of packages

Process:

- cleans up a chroot (removes everything except apt)
- installs the package to test and its dependencies
- Removes everything, purge all dependencies
- Purges the package to test

⇒ test of the package maintainer scripts
  \((\text{preinst}, \text{postinst}, \text{prerm}, \text{postrm})\)
under the most extreme conditions
Piuparts (2)

Also tests other things:
- upgrades
- running processes after removal
- dangling symlinks
- files left after removal/purge, files from other packages modified
Summary

1. Introduction
2. Tests
3. Building packages more efficiently
4. Piuparts and false positives
5. State of the archive
6. Collab-qa project
7. Conclusion
Rebuilding packages : resources usage

Rebuilding all packages in Debian Etch:
about **10 days** on a single computer

Most packages are fast to build:

![Graph showing package build time distribution](image-url)
Rebuilding packages: resources usage (2)

But some packages take a long time (numbers from etch):

<table>
<thead>
<tr>
<th>Source package</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>openoffice.org</td>
<td>7 h 14 min</td>
</tr>
<tr>
<td>latex-cjk-chinese-arphic</td>
<td>6 h 18 min</td>
</tr>
<tr>
<td>linux-2.6</td>
<td>5 h 43 min</td>
</tr>
<tr>
<td>gcc-4.1</td>
<td>2 h 52 min</td>
</tr>
<tr>
<td>gcj-4.1</td>
<td>2 h 44 min</td>
</tr>
<tr>
<td>gnat-4.1</td>
<td>1 h 52 min</td>
</tr>
<tr>
<td>gcc-3.4</td>
<td>1 h 50 min</td>
</tr>
<tr>
<td>installation-guide</td>
<td>1 h 45 min</td>
</tr>
<tr>
<td>axiom</td>
<td>1 h 44 m</td>
</tr>
<tr>
<td>k3d</td>
<td>1 h 39 min</td>
</tr>
</tbody>
</table>

(On Dual-Opteron 2 GHz, 2 GB RAM)
Parallel Rebuilds on an HPC grid

Rebuilding Debian on a computer grid

- I could use 100s of nodes
- But it’s useless because openoffice.org takes too long

⇒ Full rebuild of etch in about 7.5 hours on 40 nodes
Leveraging multi-cores

- dual-core laptops
- quad-core desktops

Already available.
Wouldn’t it be nice to make use of them?
#209008: common interface for parallel building

- DEB_BUILD_OPTIONS_PARALLEL=n
  or
- DEB_BUILD_OPTIONS="parallel=n"

Red bike shed problem?
Will hopefully be included in the next policy update (no ETA, AFAIK)
Parallel build of linux-2.6

On a dual-Opteron (both dual-cores), 2 GB RAM
Summary

1. Introduction
2. Tests
3. Building packages more efficiently
4. Piuparts and false positives
5. State of the archive
6. Collab-qa project
7. Conclusion
Piuparts and false positives

Piuparts generates A LOT of false positives

To be tested, a package must be able to install non-interactively

- debconf is nice (Noninteractive frontend)
- but doesn’t solve everything (e.g. packages that need access a database)

⇒ Make all packages use debconf (except essential ones) : policy bug #206684

⇒ After that, not much to do about false positives
Piuparts : Future work

- Improve piuparts
  - now (supposed to be) maintained collaboratively!
- piatti.debian.org : dual Xeon in helsinki
  - Used by liw to run piuparts over the archive
  - Idea : Xen instances for interested DD to reproduce/investigate results
- Other ideas?
Summary

1. Introduction
2. Tests
3. Building packages more efficiently
4. Piuparts and false positives
5. State of the archive
6. Collab-qa project
7. Conclusion
Trivia

- john - active password cracking tool
- webcalendar - PHP-Based multi-user calendar

What do john and webcalendar have in common?
Trivia

- john - active password cracking tool
- webcalendar - PHP-Based multi-user calendar

What do john and webcalendar have in common?
- both were in sarge, and are in unstable
- both are useful software (I use both)
Trivia

- john - active password cracking tool
- webcalendar - PHP-Based multi-user calendar

What do john and webcalendar have in common?
- both were in sarge, and are in unstable
- both are useful software (I use both)
- neither john nor webcalendar are in etch
Many packages missed the release

Packages in unstable, but not in etch, were reviewed after the etch release

- 433 packages (excl. packages uploaded after the freeze)
- in many cases (>50%), the maintainer simply forgot to request an *unblock*
- or wasn’t aware of his package’s RC bugs

Example bugs: #402245, #381817, #384558, #414845, and many others

⇒ We need a way to keep maintainers informed of their packages’ status
Proposal: DDPO by mail

- DDPO is nice
  - But only if you use it
  - Ideally: browser’s start page for maintainers, but...
- Idea: send one monthly email to each maintainer with the most important information about his packages
  - open RC bugs
  - packages not in testing
  - important bugs with patches
Proposal : DDPO by mail (2)

- opt-out, so it has to stay as useful as possible
- *ignore* mechanism (per package, per bug, per problem)

Current implementation status :

- BTS metadata imported to a postgres DB on merkel.d.o (could be used to generate interesting stats as well)
  - But bugs need to be fixed
  - Use bts.turmzimmer.net as input instead (easier !)
- Testing status for all packages
  ⇒ Ready to start sending mails
Summary

1. Introduction
2. Tests
3. Building packages more efficiently
4. Piuparts and false positives
5. State of the archive
6. Collab-qa project
7. Conclusion
Collaborative Quality Assurance : collab-qa project

- QA tasks used to be done by (motivated) individuals
- Working as a team brings more fun
- And is more scalable

**collab-qa project on alioth :**
- share results of QA tests (archive rebuilds, piuparts runs)
- keep them for history
- makes things more fun and more efficient
collab-qa status

<Lunar> I think I’m becoming a perverse...
I enjoy reporting FTBFS.

Worked on :
- Packages that missed etch (not finished yet)
- Archive rebuilds (up to date for 14/06/2007)
- File conflicts between packages

Plans to work on :
- Piuparts runs
- put your idea here

Don’t hesitate to join!
Summary

1. Introduction
2. Tests
3. Building packages more efficiently
4. Piuparts and false positives
5. State of the archive
6. Collab-qa project
7. Conclusion
Conclusion

- Let’s make QA rock for lenny!
- Join the collab-qa team
  - /join #debian-qa
  - subscribe to debian-qa@lists.debian.org
  - request membership on alioth
- Open questions:
  - What do you think of that "DDPO by mail" idea?
  - What about a "Packages in a questionable state" team?