

dCache's automated build and test-approach

Owen Synge
DCache Team member

About this talk

- Lots of components
 - Automated Build
 - Automated Install
 - Functional Test suite
- Why is it important?
- What is the process?
- How do we do it?

Why is automated build and test important ?

- Reduces bugs, support and effort.
 - More true the more sites you deploy too.
- Reduces errors.
 - Lots of little error prone steps.
- Speeds development.
 - Reduces test release times.
- Improves releases
 - More is tested.

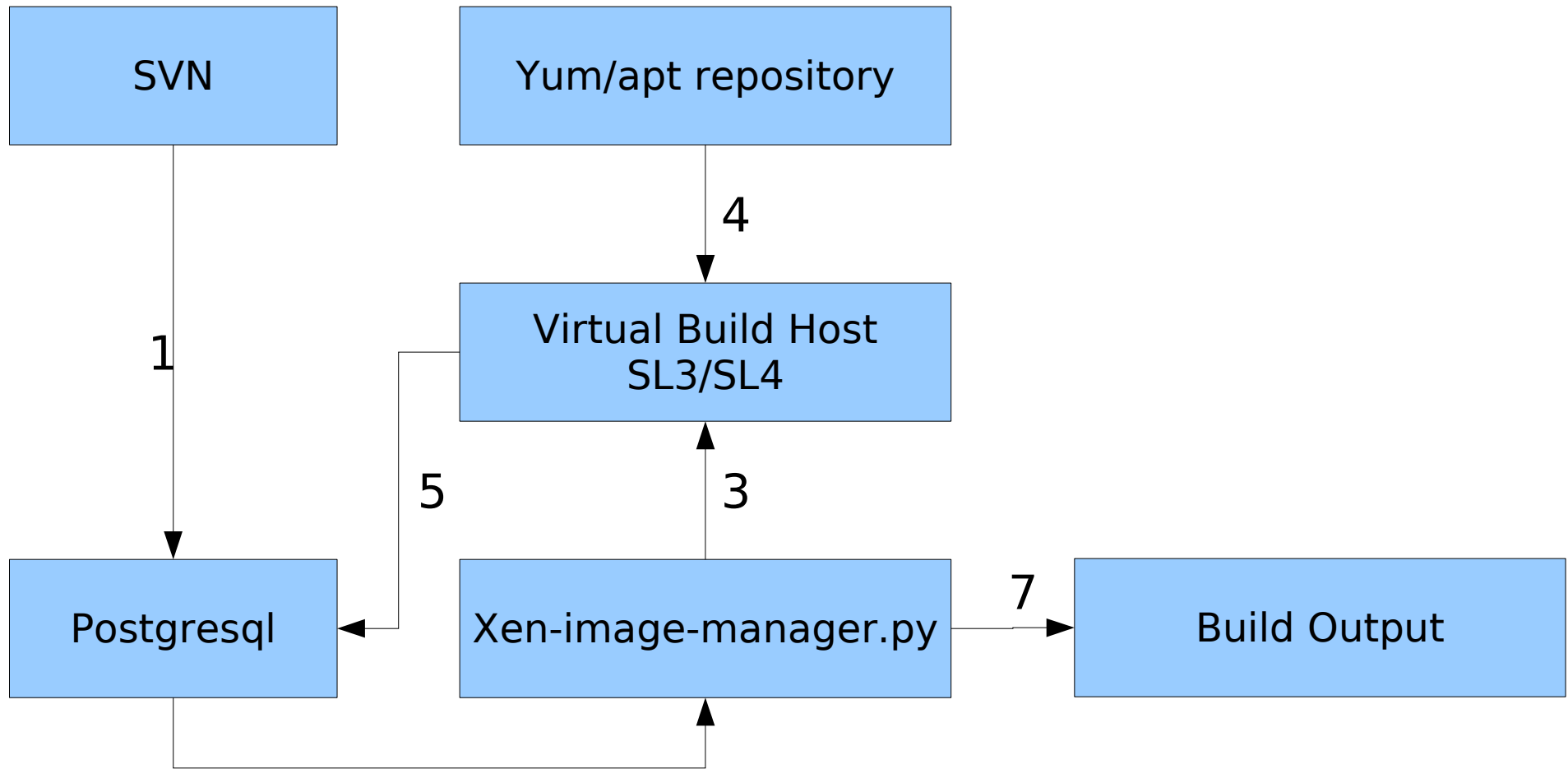
D-Cache Build Service Purpose

- Build service for Dcache and Desy Code
 - No more it builds on my machine.
 - Reproducible builds
 - For external collaborators
- Secure and up to date build environment
 - No more need to check build environment
 - Fresh OS for each build.
- 1st Stage of automation
 - Do this first!

Implementation/Tools Reused

- Implementation.
 - Utilizes SVN/CVS, Busybox, yum/apt-get XEN and xen-image-manger.py
- Code
 - Modular and simple python scripts
- State
 - Postgresql

Simplified Block Diagram Of Build Service



2 & 6

- 1) CVS tag is published and script places into RDBMS
- 2) Database triggers system installation
- 3,4) New system is updated and build dependencies installed
- 5) Build state recorded in database
- 6,7) Packaged output made available

Build system “To Do” list.

- Parallel builds
- SL4 Support (possibly Solaris)
- Move to YUM
 - For SL4 and dependability
- Move to SVN
 - move away from CVS
- Publish output to web
- Auto generate OS images for Xen
 - kickstart/rpmstrap?

Automated install (Dcache YAIM)

- Required for Mass deployment
 - Hand modification not practical
- Enables further automation
 - integrate AutoBuild and Functional tests
- Makes deployment consistent
 - Reduces support calls
- Speeds Testing
 - 2 day job for fresh install (interactive)
 - 30 min job (non interactive)

Automated install “To Do”

- Ports SL4 32/64bit and Solaris.
 - SL3 is supported
- Support postgresql 8.2
 - postgresql $\leq$ 8.1 supported
- Integrate into core of DCache.
 - Move into DCache RPM
- Configurable logging levels.
- Taking OSG patches back to LSG code.
- Maintenance and support

Why a Functional Test Suite?

- Increases importance with functionality.
 - but easier to start at beginning (XP coding)
- Reduces release issues.
 - Consensus of admins is dcache releases are much more consistent since we had this.
- Reduces testing time
 - For DCache
 - 1 day per release (interactive)
 - 15 mins per release (not interactive)
- Increases developer speed.

Design

- Modular pluggable system
 - Using python
 - Modules Tagged (self describing)
 - Allow tests to be run by filter.
- Command line interface
 - All tests or individual tests can be run
- Web interface
 - Calls command line from busyb making web page

Functional test suite “To Do”

- Publish web output
 - We have now changed internal services so no longer blocked on this.
- Add to CERN's “SAM” test suite
- Add more tests
 - VOMS testing
 - SRM v2 tests
- Add to build system
- Documentation

So how do we release this tested code?

- Web page
 - as tar, rpm by architecture.
- Yum/Apt repository
 - SL3,SL4 32/64 bit
- To LCG
 - Certification testbed
 - fresh install and upgrade tests.
 - 1 week no functional tests fail
 - Pre production testbed
 - 4 weeks no admins reject.

Summary Plans

- Automate full work flow of release
 - Build->Deploy->Test
 - no interaction needed
 - many platforms, many architectures
- Continue supporting “current” LCG architecture.
 - This is a moving target.
- Moving toward test driven development.
 - Tests written before service.
- Stress testing (Much harder to automate)

Summary Achievements

- Dcache deployment to LCG
 - CERN deployment lag reduced to 1-2 months
- Dcache LCG deployment now in OSG
 - Also in VDT.
- Admins assert release quality increased
- Much faster release cycles

Summary

- Essential for all grid software.
 - Build system.
 - Functional test suite.
 - Automated install.
- Test driven development is faster
 - Harder to add in retrospect
- Finding bugs before the users helps
 - reduce bug reports, increases deployment, makes deployment people happy :)