Lustre Release and Weekly Testing Overview

Yu Jian
jian.yu@sun.com
March, 2008

Contents

- Lustre versions and branches
- Lustre release testing
- Lustre weekly testing
- Issues and proposals

Lustre versions

- Production release
  - Major release (such as 1.4.0, 1.6.0, 1.8.0)
  - Minor release (such as 1.4.12, 1.6.4)
  - Latest production releases: 1.4.12 and 1.6.4
- Maintenance release
  - Latest maintenance releases: 1.4.11.1 and 1.6.4.3
- Beta release (pre-release version)
  - Latest beta releases: 1.6.4.52 and 1.7.91

Lustre branches

- Base branch
  - b1_4, b1_6 and HEAD
- Release branch
  - Named as “b_release_{production release number}”
  - Latest release branches: b_release_1_4_12 and b_release_1_6_4
- Development branch
  - Named as “bmajor_minor_project[_feature[_developer]]”
  - Such as b1_8_gss, b1_6_dir_ra, b1_6_scjody_12411

Lustre testings

- Release testing
  - Against production and maintenance release candidates (RC) tagged from release branches
- Weekly testing
  - Against beta releases made from base branches
- Feature testing
  - Against development branches

Lustre release testing

- Old test cycle (before May, 2007)
  1) RMG submits release testing request to ltest automated testing system through Buffalo “testing requests” web page.
  2) ltest builds Lustre and runs a suite of tests automatically on Boulder test cluster and sends test results to Buffalo.
  3) QA team vet the test results on Buffalo, update old or open new bugs on Bugzilla, and send release testing status matrix to RMG and Beaver team.

Lustre release testing (contd.)

- New test cycle (since May, 2007)
  1) QA team create release test plan against the upcoming RC and get the test plan reviewed by the RMG.
  2) RMG creates an RC tag (e.g., v1_6_4_RC1 for version 1.6.4) and submits build requests to LBATS system to build the packages for all of the supported platforms announced in lustre/ChangeLog.
  3) RMG creates a release testing tracker which blocks the release tracker in Bugzilla and notifies QA team that a release candidate is available for testing.

Lustre release testing (contd.)

- New test cycle (contd.)
  4) QA team schedule test cluster time and manually run tests following the release test plan.
  5) QA team vet and send test results to Buffalo, update old or open new bugs on Bugzilla, and update the testing status matrix in the release testing tracker.
  6) RMG determines whether a new RC testing is needed. If yes, then go back to the test cycle step 1); else, marks the release testing as complete.

Lustre release testing (contd.)

- Release test plan
  - Test items (what's to be tested?)
    - Fixed bugs and new functionalities, which are recorded in lustre/ChangeLog and lnet/ChangeLog
  - Test types and suites (what tests to be performed?)
    - Functional/Acceptance testing (acceptance-small test suite)
    - Performance testing (IOR, PIOS, Metabench, Compilebench, LST)
    - Stress testing (low-memory, multi-client-per-node+Iozone/IOR/Simul)
    - Interoperability/Upgrade/Downgrade testing

Lustre release testing (contd.)

- Release test plan (contd.)
  - > Test matrix

Lustre 1.6.4 (tag v1_6_4_RC2) manual testing matrix:

QE	Platform	acc-sm	PIOS	Metabench	Simul	Connectathon	Interop
jack_chen	SLES10/x86_64	TEST[1]	TEST[3]	TEST[3]			TEST
yep	SLES9/i686	TEST			TEST[4]		TEST
chenzheng	RHEL4/x86_64	TEST[2]				TEST	TEST
wangyb	RHEL5/ia64	TEST[1]			TEST[1]		TEST
yujian	SLES10(vanilla 2.6.18)/x86_64	TEST[1]				TEST[1]	TEST

[1] - Run test on patchless client.

[2] - Run test under low memory (client 300M, server 500M).

[3] - Run performance test on 1GigE network covering the following scenarios:

* Native Lustre
* NFSv3 over Lustre
* NFSv4 over Lustre

[4] - Run test with file system quotas on.

Lustre release testing (contd.)

- Latest Lustre releases
  - 1.4.12, released on 2008-02-08
  - 1.6.4.3, released on 2008-03-07
- Upcoming Lustre releases
  - 1.4.13 – current focus on more than 10 blockers
  - 1.6.5 – current focus on 19 blockers
  - 1.8.0 – July, 2008

Lustre weekly testing

- Test target
  - Beta releases made from b1_6 and HEAD base branches
- Test cycle
  1) QA team track the Lustre code changes via “lustre-cvs” list and submit build request to LBATS system to build the packages for one of the supported platforms.
  2) QA team schedule test cluster time and manually run acceptance-small tests on the selected platform.
  3) QA team vet and send test results to Buffalo, update old or open new bugs on Bugzilla, and update the weekly testing trackers (bug 14045 for b1_6, bug 13174 for HEAD).

Lustre weekly testing (contd.)

- Current status
  - Ye Peng (Yep) is responsible for b1_6 testing
  - Chen Zheng (Thunder) is responsible for HEAD testing
  - Bi-weekly or tri-weekly testing in reality
  - Very important for tracking the quality and stability of the base branches

Issues and proposals

- Issue:
  - Production release test cycle takes very long time due to the following issues:
    - Release branch is unstable (e.g., 4 RCs for Lustre 1.6.4)
    - Insufficient test cluster restricts QA team running tests in parallel
- Proposal:
  - Focus on the weekly testing against base branches and the feature testing against development branches
  - After the Jackie and Frankie test clusters are ready, QA team could run tests in parallel to a large extent
  - Automating the test process to cut down time and human cost

Issues and proposals (contd.)

- Issue:
  - Some apparent performance or functional regressions against the new releases are found by the customers
    - Such as bug 14353(perf regression), bug 14437(func regression)
- Proposal:
  - QA team need learn widely that how the customers use Lustre via “Lustre-discuss” list, bugs filed by customers, etc., and improve the test plan accordingly
  - Add more test scenarios into the release testing (such as failover testing with two real nodes, continuous scale testing on Frankie)

Issues and proposals (contd.)

- Issue:
  - Currently, the release test plans and test reports are not formal and public
- Proposal:
  - Improve the test plan according to some standard test plan template (such as IEEE Std 829)
  - Improve the performance test report referring to some standard template or good sample (such as Lustre benchmarking report on Cray XT4)
  - Make them public and get feedbacks from the consumers

Thank You!

jian.yu@sun.com