Using Git, Gerrit, and Maloo Effectively

Andreas Dilger
Jan 22, 2014
## HPDD Development Infrastructure

<table>
<thead>
<tr>
<th>Service</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Git</td>
<td><a href="https://git.hpdd.intel.com/">https://git.hpdd.intel.com/</a></td>
</tr>
<tr>
<td>JIRA</td>
<td><a href="https://jira.hpdd.intel.com/">https://jira.hpdd.intel.com/</a></td>
</tr>
<tr>
<td>Gerrit Change Review</td>
<td><a href="https://review.whamcloud.com/">https://review.whamcloud.com/</a></td>
</tr>
<tr>
<td>Build Server</td>
<td><a href="https://build.hpdd.intel.com/">https://build.hpdd.intel.com/</a></td>
</tr>
<tr>
<td>Maloo Test Repository</td>
<td><a href="https://testing.hpdd.intel.com/">https://testing.hpdd.intel.com/</a></td>
</tr>
</tbody>
</table>

**Q & A**
Using HPDD development tools

Gerrit tracks patches and reviews for all projects at HPDD

- Google OpenID going away in April
- Google users need a new OpenID provider (e.g. launchpad.net)
- Use Settings->Link Another Identity to add new identity to account
- Do this before Google stops working

JIRA for issue tracking

- Gerrit now posts patch submission, landing automatically to JIRA

Jenkins for automated build & test (Autotest)

- Three test results enforced posted to Gerrit: review-ldiskfs, review-dne-part-[12]
- One test result not yet enforced to Gerrit: review-zfs (until LU-5242 is fixed)

Maloo tracks test results from Autotest

- Posts into Gerrit when tests are complete
Managing Branches

- Keep patches for independent changes on separate branches
- A git "branch" is just a pointer to some series of commits
- Each commit or rebase moves pointer to new tip of the branch
- All of the old commits/branches stay in repository, unreferenced

- `git checkout branch`  # check out specific branch
- `git branch {new_branch}`  # create new, keep current branch
- `git checkout -b {new_branch}`  # create new branch and checkout
- `git status`  # show modified/staged/untracked
Finding branches

- **git branch -lv**
  - b_client 986f760
  - b_ost_opc e3f8341
  - b_setstripe 3e407b7
  - b_quiet 98b320e
  - b_message ebdaeb5
  - master 98c5043

# show local branches

- LU-1842 ptlrpc: clean up ptlrpc_client setup
- LU-1234 oss: rename OST_* RPC opcodes to OSS_*
- LU-2523 tests: disable racer setstripe until fixed
- LU-2468 libcfs: quiet spurious debug message
- LU-1123 mdt: CERROR() messages print device
- LU-6127 test: add new racer scripts to Makefile

- **git branch -alv**
  - [as above, plus]
  - remotes/origin/HEAD -> origin/master
  - remotes/origin/b2_1 e91f649 LU-3546 mdt: define mdt_obd_name()
  - remotes/origin/b2_5 35046f3 LU-4958 lov: don't crash using FID [0:0:0]
  - remotes/origin/b2_6 73ea776 New tag 2.6.0-RC2
Searching in Git

- `git diff [-w]` # show current change (no whitespace)
- `git diff --cached` # show "git add" changes only
- `git grep -p {regexp}` # show all files *in repository* with regular expression
- `git log [branch|path]` # show commits (branch/dir/file)
- `git log --grep {regexp}` # search commit messages with regular expression
- `git log -G {regexp}` # search commits *diffs* for specific regexp
- `git log -S {string}` # search commits that add/remove specific string
- `git log -p` # print patch/diff with each log message
- `git show [-w] [commit]` # show specific commit (no whitespace)
- `git describe {commit}` # tagged name of commit (e.g. v2_3_61_0-25-g406e943)
Finding change history of code

- Shows history of commits that modified some line number
  - `git blame lustre/obdclass/class_obd.c`
    11330397 (nathan 2007-02-10 06:33:41 +0000 386)
    8af4adf8 (rread 2003-12-19 19:45:29 +0000 387)
    8af4adf8 (rread 2003-12-19 19:45:29 +0000 388)
    f193250f (adilger 2002-06-03 22:36:53 +0000 389)

- `git show [commit]` # what was in commit
- Show history of commits before a specific commit
  - `git blame lustre/obdclass/class_obd.c {commit}~1`
Committing changes

- Install Git commit hooks for patch verification
  - `ln -sf ../../build/commit-msg .git/hooks`
  - `ln -sf ../../build/prepare-commit-msg .git/hooks`
- Read the comments provided by `prepare-commit-msg` hook!
  - `git add [path ...]`  # stage specified file(s) or dirs for commit
  - `git add [-u]`  # stage all changes for commit
  - `git commit -av`  # stage and commit all changes, show diff
  - `git commit -av --amend`  # add current changes to previous commit
  - `git reset --hard {branch|HEAD~n}`  # revert to branch or last n commits
  - `git cherry-pick {branch|commit}`  # copy specific patch to current branch
Committing partial changes

• `git commit -p`  # add individual patch hunks
  `diff --git a/obd_support.h b/obd_support.h`  
  `@@ -482,6 +482,8 @@` int obd_alloc_fail(const void *ptr,
    #define OBD_FAIL_LFSCK_NO_AUTO 0x160b
    #define OBD_FAIL_LFSCK_NO_DOUBLESCAN 0x160c

    +#define OBD_FAIL_FID_MAPPING 0x1500
    +
    /* Assign references to moved code to reduce code changes */
    #define OBD_FAIL_PRECHECK(id) CFS_FAIL_PRECHECK(id)
    #define OBD_FAIL_CHECK(id) CFS_FAIL_CHECK(id)

  Stage this hunk [y,n,q,a,d,/,e,?]?

• `git citool`  # GUI tool to add patch hunks
Rebasing patches

- Patches should be kept up-to-date with master
- HPDD Checkpatch marks patches Verified: -1 that don't apply
- Multiple dependent patches in a single series

```bash
# edit/reorder patches
git rebase [-i] {master|b2_5}

git add -u  # add updated file(s) to commit

git rebase {--continue|--abort}  # apply or discard changes

git rebase --skip  # drop current patch entirely

git rebase --onto {branch}  # rebase onto a new branch
```
Rebasing a patch series

$ git rebase -i {master\|HEAD~2}  # rebase last two patches
pick   a427873 LU-1798 ptlrpc: add procname.uid jobid
fixup  52325a7 LU-1798 foo: few changes to previous patch
reword 4b3e88c  LU-1798 tests: don't set jobid if not changing

# Rebase 67af9ea..52325a7 onto 67af9ea
# Commands:
#  p, pick = use commit
#  r, reword = use commit, but edit the commit message
#  e, edit = use commit, but stop for amending
#  s, squash = use commit, but meld into previous commit
#  f, fixup = like "squash", but discard this commit's log message
#  x, exec = run command (the rest of the line) using shell
# These lines can be re-ordered; they are executed top to bottom.
What to do when things go wrong

All old commits/branches remain in the repository

$ git reflog
fdd29dc HEAD@{0}: commit (amend): LU-2468 libcfs: quiet debug messages
da6c3b3 HEAD@{1}: commit: LU-2468 libcfs: quiet debug messages
0b3e852 HEAD@{2}: checkout: moving from b_cleanup9 to b_cleanup7
5ea4d20 HEAD@{3}: rebase finished: returning to refs/heads/b_cleanup9
5ea4d20 HEAD@{4}: rebase: LU-1798 tests: don't set jobid if not changing
0b3e852 HEAD@{5}: checkout: moving from b_cleanup9 to 0b3e852a58^0

git rebase --abort

git fsck [--unreachable] [--lost+found]
Extracting and sharing changes

- `git format-patch HEAD~n`  # Export last n commits to patch
- `git am {patch ...}`  # Import patch(es) to commit
- `git send-email HEAD~n`  # Email last n commits
- Default values in $HOME/.gitconfig or .git/config in local repository

[user]

name = Andreas Dilger
email = andreas.dilger@intel.com

[sendemail]

smtpserver = shawmail.cg.shawcable.net
smtpserverport = 25
confirm = compose
Commit message should describe patch well

- What problem/feature does this patch address?
- Usage/test that triggered problem, with error messages
- Commit that introduced problem
- Performance improvements resulting from patch
- How it is being fixed, clear description of tricky code
- Initial inspectors understand the patch more quickly
- Gatekeeper understands importance of the patch
- Anyone dealing with the patch later has some guidance

https://wiki.hpdd.intel.com/display/PUB/Commit+Comments
Control Autotest with commit messages

Use the Test-Parameters: tag in a commit message:

- Test-Parameters: fortestonly # limit testing for a trial patch
- ... testlist=sanity,sanity,sanity # add additional tests
- ... ostfilesystemtype=zfs # use a specific backing fstype
- ... clientbuildno=81 # old client new server interop
- ... serverbuildno=81 # new client old server interop
- ... mdtcount=8,ostcount=8 # change server config
- ... alwaysuploadlogs # upload all logs even on success
Using Gerrit

Set up defaults to your liking
## Gerrit patch status

**HPDD Checkpatch** Patch Set 2: Looks good to me.  
01-19 21:51

**Jenkins** Patch Set 2: Build Started ...  
01-19 22:24

**Jenkins** Patch Set 2: Build Successful ...  
01-19 22:39

**Autotest** Patch Set 2: Shadow Autotest started enforced review-ldiskfs session.  
01-19 22:46

**Autotest** Patch Set 2: Shadow Autotest started optional review-zfs session.  
01-19 22:46

**Autotest** Patch Set 2: Shadow Autotest started enforced review-dne-part-1 session.  
01-19 22:46

**Autotest** Patch Set 2: Shadow Autotest started enforced review-dne-part-2 session.  
01-19 22:46

**Maloo**  
Patch Set 2:  
Tests received by maloo, run on CentOS release 6.5/x86_64:  
(https://testing.hpdd.intel.com/test_sessions/2ff28340-a093-11e4-87d1-5254006e85c2). Ran 6 tests. No failures.  
01-20 03:01

**Maloo**  
Patch Set 2:  
Tests received by maloo, run on CentOS release 6.5/x86_64:  
(https://testing.hpdd.intel.com/test_sessions/c6e8f00a-a0bf-11e4-8464-5254006e85c2). Ran 7 tests. No failures.  
01-20 08:22

**Maloo**  
Patch Set 2: **Verified-1**  
Tests received by maloo, run on CentOS release 6.5/x86_64:  
(https://testing.hpdd.intel.com/test_sessions/d3fd1614-a0dc-11e4-92c3-5254006e85c2). Ran 17 tests. 1 tests failed: sanity-Ifsck.  
01-20 12:04

**Maloo**  
Patch Set 2:  
Tests received by maloo, run on CentOS release 6.5/x86_64:  
(https://testing.hpdd.intel.com/test_sessions/512d800c-a0f0-11e4-87d1-5254006e85c2). Ran 18 tests. No failures in this session, but a previous session failed for this commit.  
01-20 14:11
Gerrit Searching

Searching is possible, but not very easy to use

- message:LU-2250 is:merged # patches landed for LU-2250
- branch:master Verified>=1 -Verified<=-1 CodeReview>=1 -CodeReview<=-1
- reviewer:self -owner:self label:Code-Review=0,user=self label:Verified

Search shortcut/keyword for Firefox

<table>
<thead>
<tr>
<th>Name</th>
<th>Patch Review</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location</td>
<td><a href="http://review.whamcloud.com/#q,%25s,n,v">http://review.whamcloud.com/#q,%s,n,v</a></td>
</tr>
<tr>
<td>Tags</td>
<td>Separate tags with commas</td>
</tr>
<tr>
<td>Keyword</td>
<td>review</td>
</tr>
</tbody>
</table>

review 12345 # open change #12345
review message:LU-2250 # search commit messages for LU-2250
(Re)basing a patch on an existing change

- `git fetch http:...` & `git checkout FETCH_HEAD`  # get patch tree (#1, #2, #3)

1. `git checkout -b b_alex_patch`  # named branch for other patch
2. `git checkout -b b_my_patch`  # named branch for my patch
3. `git cherry-pick {my patch}`  # apply new patch on top
4. `git log`  # check commit hash unchanged (#4)
5. `git push ... HEAD:refs/for/master`  # push new patch to Gerrit

**DO NOT** rebase branch to latest master before submitting!
- Original developer should rebase or land their patch first
Reviewing overall test results in Maloo

Maloo permanently holds all test results for all patches

Test logs are kept as long as possible (months)

https://testing.hpdd.intel.com/

<table>
<thead>
<tr>
<th>Host</th>
<th>Group</th>
<th>User</th>
<th>Branches</th>
<th>Builds</th>
<th>Run at (UTC)</th>
<th>Imported at (UTC)</th>
<th>Sets passed</th>
<th>Links</th>
</tr>
</thead>
</table>
Reviewing a test failure in Maloo

Please link test failures to bugs

Error: '(7) unexpected status'
Failure Rate: 53.00% of last 100 executions [all branches]
Client 2 (shadow-7vm9)
  - dmesg [Download] [Show] [Preview 10]
  - debug_log [Download] [Show] [Preview 10]
MDS 3, MDS 4, MDS 2 (shadow-7vm8)
  - debug_log [Download] [Show] [Preview 10]
  - dmesg [Download] [Show] [Preview 10]
MDS 1 (shadow-7vm12)
  - debug_log [Download] [Show] [Preview 10]
  - dmesg [Download] [Show] [Preview 10]
Client 1 (shadow-7vm10)
  - dmesg [Download] [Show] [Preview 10]
  - debug_log [Download] [Show] [Preview 10]
  - test_log [Download] [Show] [Preview 10]

```
- /usr/lib64/lustre/tests/test-framework.sh:4675: error: noexit()
  = /usr/lib64/lustre/tests/test-framework.sh:4718: error()
  = /usr/lib64/lustre/tests/sanity-lfsck.sh:502: test_4()
  = /usr/lib64/lustre/tests/test-framework.sh:4972: run_one()
  = /usr/lib64/lustre/tests/test-framework.sh:5905: run_one_logged()
  = /usr/lib64/lustre/tests/test-framework.sh:4855: run_test()
```

Dumping lctl log to /logdir/test_logs/2015-01-29/lustre-reviews-e16-x86_64--review-dne-part-2--2_9_1_29821__-70124987833140-064657/sanity-lfsck.test_4.*.1421750399.log
CMO: shadow-7vm10, shadow.whamcloud.com, shadow-7vm11, shadow-7vm12, shadow-7vm8, shadow-7vm9 /usr/sbin/ltldk > /logdir/test_logs/2015-01-29/lustre-reviews-e16-x86_64--review-dne-part-2--2_9_1_29821__-70124987833140-064657/sanity-lfsck.test_4.dmesg \(hostname = s\).142

OST 8, OST 6, OST 7, OST 4, OST 5, OST 3, OST 2, OST 1 (shadow-7vm11)
  - dmesg [Download] [Show] [Preview 10]
  - debug_log [Download] [Show] [Preview 10]

Bugs
No bugs

Associate bug... Raise bug...
Searching in Maloo

Search in: [Test sets | Sub tests]

Sub tests:
- Test set script: sanity
- Sub test script: test_133g
- Status: TIMEOUT
- Bugs: e.g. 526, TT-7, LU-23

...in sessions like...
- Test host
- Test group
- User
- Run on
- Run within: 1 week

...with nodes like...
- OS
- Distribution
- Architecture
- File system type
- Lustre branch
- Network

Update results  Clear form
Search results in Maloo

Search
Search in: [Test sets | Sub tests]

Sub tests...
- Test set script: sanity
- Sub test script: test_133g
- Status: TIMEOUT
- Bugs: e.g., 526, 77-7, LU-20

...in sessions like...
- Test host: 
- Test group: 
- User: 
- Run on: 
- Run within: 1 week

...with nodes like...
- OS: 
- Distribution: 
- Architecture: 
- File system type: 
- Lustre branch: 
- Network: 

Download these results in CSV format

<table>
<thead>
<tr>
<th>Name</th>
<th>Status</th>
<th>Run at</th>
<th>Duration</th>
<th>Return code</th>
<th>Error</th>
<th>Test log</th>
<th>Bugs</th>
</tr>
</thead>
<tbody>
<tr>
<td>test_133g</td>
<td>TIMEOUT</td>
<td>2015-01-21 13:51:38 UTC</td>
<td>3600</td>
<td>-1</td>
<td>test failed to respond and timed out</td>
<td>Preview 10</td>
<td></td>
</tr>
<tr>
<td>test_133g</td>
<td>TIMEOUT</td>
<td>2015-01-20 23:28:28 UTC</td>
<td>3600</td>
<td>-1</td>
<td>test failed to respond and timed out</td>
<td>Preview 10</td>
<td></td>
</tr>
<tr>
<td>test_133g</td>
<td>TIMEOUT</td>
<td>2015-01-20 15:43:50 UTC</td>
<td>3600</td>
<td>-1</td>
<td>test failed to respond and timed out</td>
<td>Preview 10</td>
<td></td>
</tr>
<tr>
<td>test_133g</td>
<td>TIMEOUT</td>
<td>2015-01-20 13:48:29 UTC</td>
<td>3600</td>
<td>-1</td>
<td>test failed to respond and timed out</td>
<td>Preview 10</td>
<td></td>
</tr>
<tr>
<td>test_133g</td>
<td>TIMEOUT</td>
<td>2015-01-20 06:50:45 UTC</td>
<td>3600</td>
<td>-1</td>
<td>test failed to respond and timed out</td>
<td>Preview 10</td>
<td></td>
</tr>
<tr>
<td>test_133g</td>
<td>TIMEOUT</td>
<td>2015-01-20 03:22:14 UTC</td>
<td>3600</td>
<td>-1</td>
<td>test failed to respond and timed out</td>
<td>Preview 10</td>
<td></td>
</tr>
<tr>
<td>test_133g</td>
<td>TIMEOUT</td>
<td>2015-01-19 23:39:06 UTC</td>
<td>3600</td>
<td>-1</td>
<td>test failed to respond and timed out</td>
<td>Preview 10</td>
<td></td>
</tr>
<tr>
<td>test_133g</td>
<td>TIMEOUT</td>
<td>2015-01-19 20:24:51 UTC</td>
<td>3600</td>
<td>-1</td>
<td>test failed to respond and timed out</td>
<td>Preview 10</td>
<td></td>
</tr>
<tr>
<td>test_133g</td>
<td>TIMEOUT</td>
<td>2015-01-19 01:56:18 UTC</td>
<td>3600</td>
<td>-1</td>
<td>test failed to respond and timed out</td>
<td>Preview 10</td>
<td></td>
</tr>
<tr>
<td>test_133g</td>
<td>TIMEOUT</td>
<td>2015-01-18 21:58:24 UTC</td>
<td>3600</td>
<td>-1</td>
<td>test failed to respond and timed out</td>
<td>Preview 10</td>
<td></td>
</tr>
<tr>
<td>test_133g</td>
<td>TIMEOUT</td>
<td>2015-01-17 09:03:05 UTC</td>
<td>3600</td>
<td>-1</td>
<td>test failed to respond and timed out</td>
<td>Preview 10</td>
<td>LU-5242</td>
</tr>
<tr>
<td>test_133g</td>
<td>TIMEOUT</td>
<td>2015-01-17 03:42:04 UTC</td>
<td>3600</td>
<td>-1</td>
<td>test failed to respond and timed out</td>
<td>Preview 10</td>
<td>LU-5242</td>
</tr>
<tr>
<td>test_133g</td>
<td>TIMEOUT</td>
<td>2015-01-16 23:21:51 UTC</td>
<td>3600</td>
<td>-1</td>
<td>test failed to respond and timed out</td>
<td>Preview 10</td>
<td>LU-5242</td>
</tr>
</tbody>
</table>
Can limit search incrementally
## Monitoring Test Progress

### Test Queue

Queue last updated at 2015-01-22 12:06:35 (1 minute ago).

<table>
<thead>
<tr>
<th>Job</th>
<th>Distro</th>
<th>Build No</th>
<th>Review ID</th>
<th>Suite</th>
<th>Test</th>
<th>Restarts</th>
<th>Timeout</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>reviews</td>
<td>e6-x86_64/e6-x86_64/lidskfs/lidskfs/2.8.1</td>
<td>29866</td>
<td>gerrit:13112</td>
<td>sanity-hsm</td>
<td>test_30b</td>
<td>1</td>
<td>2015-01-22 13:06:33 UTC</td>
<td>Test in progress on shadow-25vm9 - shadow-25vm9,shadow-25vm6,...</td>
</tr>
<tr>
<td>reviews</td>
<td>e6-x86_64/e6-x86_64/zfs/zfs/2.5.1</td>
<td>29868</td>
<td>gerrit:13492</td>
<td>conf-sanity</td>
<td>test_49a</td>
<td>0</td>
<td>2015-01-22 13:06:27 UTC</td>
<td>Test in progress on shadow-25vm5 - shadow-25vm5,shadow-25vm2,...</td>
</tr>
<tr>
<td>reviews</td>
<td>e6-x86_64/e6-x86_64/lidskfs/lidskfs/2.8.1</td>
<td>29868</td>
<td>gerrit:13492</td>
<td>sanity</td>
<td>test_42e</td>
<td>0</td>
<td>2015-01-22 13:04:30 UTC</td>
<td>Test in progress on shadow-24vm10 - shadow-24vm10,shadow-24vm...</td>
</tr>
<tr>
<td>reviews</td>
<td>e6-x86_64/e6-x86_64/lidskfs/lidskfs/2.8.1</td>
<td>29868</td>
<td>gerrit:13492</td>
<td>sanity-hsm</td>
<td>test_500</td>
<td>0</td>
<td>2015-01-22 13:05:22 UTC</td>
<td>Test in progress on shadow-23vm9 - shadow-23vm9,shadow-23vm5,...</td>
</tr>
<tr>
<td>reviews</td>
<td>e6-x86_64/e6-x86_64/zfs/zfs/2.5.1</td>
<td>29870</td>
<td>gerrit:13493</td>
<td>conf-sanity</td>
<td>test_50b</td>
<td>0</td>
<td>2015-01-22 13:06:30 UTC</td>
<td>Test in progress on shadow-52vm2 - shadow-52vm2,shadow-52vm1,...</td>
</tr>
</tbody>
</table>
Keep patches moving forward

Developers are responsible for their own patches

- Update patches based on inspections
- Keep inspections and testing moving forward

Provide inspection feedback for others in a timely manner

After inspection, help move patch to next state

- Two or more inspectors on the patch?
- Tests have passed or triaged and retested as needed?
- Gatekeeper landing requested if patch is ready

Good inspections avoid more effort finding defects later

- Chance to find and fix latent bugs, cleanup code
Autotest failures affect everyone

• Need to triage all test failures before resubmitting
  • Patch may otherwise be adding new intermittent bugs
• *Associate* or *Raise* bugs in Maloo for all test failures
• Search for and triage similar failures via Maloo, JIRA
  • Can be done more efficiently in batches
• Data needed to prioritize bugs causing the most failures
  • This speeds up fixing the most frequent failures
  • This in turn speeds up every other patch landing