



Whamcloud

Continuous Testing and integration to keep Lustre code great

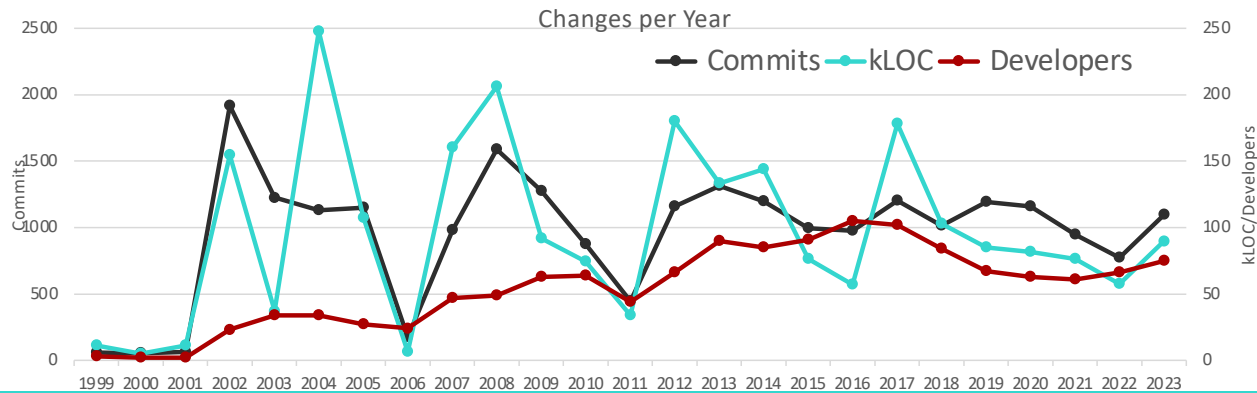
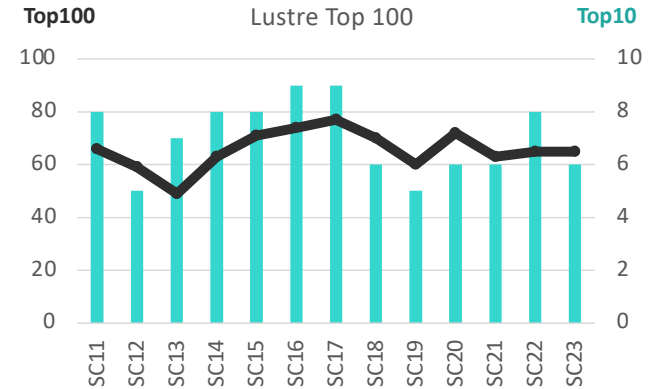
Oleg Drokin, May 8 2024



Lustre is a complicated piece of actively developed software



- ▶ 1200 kLOCs
- ▶ Change rate of 8-10% annually at ~100kLOC
 - Across 1000+ commits
- ▶ Runs on 60% of fastest supercomputers
- ▶ Downtime is very expensive at that scale
- ▶ Needs extraordinary testing to keep it stable
- ▶ And not just testing



Integration pipeline outline

- ▶ For every patch:
 - Static code analysis
 - Testing: Two parallel systems with somewhat different goals
 - Manual code reviews (requires at least two extra developers voting for the patch)

- ▶ Integration testing:
 - "Ready to land" patches are accumulated in "integration" branch (master-next)
 - <https://wiki.whamcloud.com/display/PUB/Patch+Status>
 - Testing: Two parallel systems, expanded testset
 - "Boilpot" jittertesting
 - Final review by the gatekeeper
 - Patch gets merged

- ▶ A round of "full" testing is executed + some other out of band testing.

Static analysis at large

- ▶ Initially: “stupid computer” just highlights strange areas in the code for developer to review
- ▶ Today: “a tool for managers to measure code quality”
- ▶ Usage and commercial offerings shifted accordingly and for the worse
 - It’s usually run every once in a while and the reports are often left to be triaged and fixed by junior people
- ▶ The end result is not useful
- ▶ Once the bug is in the codebase, it’s too late
 - Developer has moved on to other things
 - It becomes everybody else’s problem
 - It could get deprioritized for later
- ▶ The proper way is to run the checks on every patch
 - But it is not easy to do this with commonplace tools like Coverity



Static analysis at Whamcloud

- ▶ At Whamcloud we run static analysis on every patch
- ▶ The tool of choice: Smatch
 - Free and opensource
 - Targets Linux kernel
 - Always on the bleeding edge of research in the area
 - Produces easily parsable text output to tie into gerrit reviews by our tools
- ▶ Some false positives are OK
 - Computers are stupid after all
 - They get blacklisted not to annoy people needlessly
 - They do work as anchors to increase review quality
 - Important not to have too many still

Example of gerrit integration

```

52  »      spin_lock(&ou->ou_lock);
53  -»      oth->ot_version = ou->ou_version++;
54  +»      spin_lock(&oth->ot_our->our_list_lock);
55  +»      if (obj->opo_stale) {
56  +»          »      spin_lock(&ou->ou_lock);
57  +»          »      spin_lock(&oth->ot_our->our_list_lock);
58  +»          »      return -ESTALE;
59  +»      }
60  +
61  +»      /* Assign the version and add it to the sending list */
62  +»      osp_thandle_get(oth);
63  +»      oth->ot_our->our_version = ou->ou_version++;
64  +»      list_add_tail(&oth->ot_our->our_list,
65  +»          »      &osp->opd_update->ou_list);
66  +»      oth->ot_our->our_req_ready = 0;
67  +»      spin_unlock(&oth->ot_our->our_list_lock);
68  +»      spin_unlock(&ou->ou_lock);
69  +
70  +»      LASSERT(oth->ot_super.th_wait_submit == 1);
71  +»      CDEBBUG(D_INFO, "%s: version "LPU64" oth:version %p:"LPU64"\n",
72  -»          osp->opd_obd->obd_name, ou->ou_version, oth, oth->ot_version);
73  +»      osp->opd_obd->obd_name, ou->ou_version, oth,
74  +»          oth->ot_our->our_version);
75  +»      return 0;

```

Misc Code Checks Robot (Gatekeeper helper)

error: osp_check_and_set_rpc_version():double lock 'spin_lock:&ou->ou_lock'

Reply ... Reply 'Done'

Misc Code Checks Robot (Gatekeeper helper)

error: osp_check_and_set_rpc_version():double lock 'spin_lock:&oth->ot_our->our_list_lock'

Reply ... Reply 'Done'

Misc Code Checks Robot (Gatekeeper helper)

warn: osp_check_and_set_rpc_version():inconsistent returns 'spin_lock:&ou->ou_lock'. warn: osp_check_and_set_rpc_version():inconsistent returns 'spin_lock:&oth->ot_our->ou

Reply ... Reply 'Done'

Gerrit integration – fast turnaround



```
https://review.whamcloud.com/#/c/24882/15/lustre/osd-ldiskfs/osd_scrub.c
Patch Set (-) Base 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
Patch Set (+) 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
diff --git a/lustre/osd-ldiskfs/osd_scrub.c b/lustre/osd-ldiskfs/osd_scrub.c
index b686063..50be241 100644
--- a/lustre/osd-ldiskfs/osd_scrub.c
+++ b/lustre/osd-ldiskfs/osd_scrub.c
@@ -418,50 +418,50 @@
418 418 *      /* 2) delete the old XATTR_NAME_FID */
419 419 *      ll_vfs_dq_init(inode);
420 420 *      rc = inode->i_op->removexattr(dentry, XATTR_NAME_FID);
421 421 *      if (rc != 0)
422 422 *          GOTO(stop, rc);
423 423 *
424 424 *      removed = true;
425 425 *      } else if (unlikely(rc == -ENODATA)) {
426 426 *          reset = false;
427 427 *      } else if (rc != sizeof(struct filter_fid)) {
428 428 *          GOTO(stop, rc = -EINVAL);
429 429 *      }
430 430 *
431 431 *      /* 3) make new LMA and add it */
432 432 *      rc = osd_ea_fid_set(info, inode, tfid, LMAC_FID_ON_OST, 0);
433 433 *      if (rc == 0 && reset)
434 434 *          size = sizeof(struct filter_fid);
435 435 *      else if (rc != 0 && removed)
436 436 *          /* If failed, we should try to add the old back. */
437 437 *          size = sizeof(struct filter_fid_old);
438 438 *
439 439 *      /* 4) generate new XATTR_NAME_FID with the saved parent FID and add it*/
440 440 *      if (size > 0) {
441 441 *          int rcl;
442 442 *
443 443 *          rcl = osd_xattr_set(info, inode, XATTR_NAME_FID, ff, size,
```

Misc Code Checks Robot (Gatekeeper helper) Mar 22 12:14 PM
error:osd_scrub_convert_ff():_osd_xattr_set) 'ff' too small (32 vs 44)
Reply ... Reply 'Done'

Jinshan Xiong Mar 22 2:12 PM
this seems suspicious that means the size of 'ff' and 'size' do not agree with each other.
Reply ... Reply 'Done'

Fan Yong Mar 23 11:18 AM
fixed in set 16
Reply ... Reply 'Done'

Regular Testing - Autotest Test Sessions

Maloo

Patch Set 6:

The following sessions will be run for build 104168 patch 6

Enforced:

- review-ldiskfs on el8.9-x86_64/diskfs servers and clients
- review-zfs on el8.9-x86_64/zfs servers and clients
- review-dne-part-1 on el9.3-x86_64/diskfs servers and clients
- review-dne-part-2 on el9.3-x86_64/diskfs servers and clients
- review-dne-part-3 on el9.3-x86_64/diskfs servers and clients
- review-dne-part-4 on el9.3-x86_64/diskfs servers and clients
- review-dne-part-5 on el9.3-x86_64/diskfs servers and clients
- review-dne-part-6 on el9.3-x86_64/diskfs servers and clients
- review-dne-part-7 on el9.3-x86_64/diskfs servers and clients
- review-dne-part-8 on el9.3-x86_64/diskfs servers and clients
- review-ldiskfs on el9.2-x86_64/diskfs servers and clients
- review-ldiskfs on el9.2-x86_64/diskfs servers and el9.3-x86_64/diskfs clients
- review-ldiskfs on el9.3-x86_64/diskfs servers and clients
- review-ldiskfs-arm on el8.9-x86_64/diskfs servers and el8.8-aarch64/diskfs clients
- review-dne-selinux-ssk-part-1 on el8.9-x86_64/diskfs servers and clients
- review-dne-selinux-ssk-part-2 on el8.9-x86_64/diskfs servers and clients
- review-ldiskfs on el8.9-x86_64/diskfs servers and el9.2-x86_64/diskfs clients
- review-ldiskfs on el8.9-x86_64/diskfs servers and sles15sp4-x86_64/diskfs clients
- review-ldiskfs on el8.9-x86_64/diskfs servers and sles15sp5-x86_64/diskfs clients

Optional:

- review-dne-part-9 on el9.3-x86_64/diskfs servers and clients
- review-ldiskfs on el8.9-x86_64/diskfs servers and el9.3-x86_64/diskfs clients
- review-ldiskfs-ubuntu on el8.9-x86_64/diskfs servers and ubuntu2204-x86_64/diskfs clients

Maloo Test Queue: https://testing.whamcloud.com/test_queue?jobs=lustre-reviews&builds=104168&commit=Apply+Filter

Maloo Results: https://testing.whamcloud.com/test_sessions/related?jobs=lustre-reviews&builds=104168#redirect

▶ 20+ test sessions **per patch version!**

- Multiple sessions *enforced* for master patches
- Multiple sessions *optional* for arches, distros

▶ **Enforced** test sessions **must pass**

- Custom Test-Parameters: runs are **Enforced**

▶ **Optional** sessions *are optional*

- Run to test new distro/arch, or flakey tests (e.g. racer)
- Working toward passing and enforcement

▶ **Please use test resources wisely**

- 170 tests, 13500 subtests, 150h+ *per patch*

Test infrastructure – a different approach

- ▶ If you want something to be done well, do it yourself.
- ▶ Frustrated by existing solutions, I set out to create my own with some simple goals
 - People are lazy and impatient. Give them useful results. Fast!
 - Compile finished under 5 minutes
 - Generally fatal problem under 10 minutes
 - Overall bill of health under 2 hours with all tests we have, no exclusions
 - Give them more data than they need in convenient locations
 - Compile error? Show it as review comments
 - Crash in new code? Show it in place. Immediately.
 - Pre-parse the logs to highlight messages of interest in test results
 - Context aware (only test what's changed)
- ▶ Use full debug enabled kernels
 - Slower, but catches so much more of "it's only theoretical" issues.
- ▶ "Only" 50h of tests (single distro)
- ▶ Opensource - <https://github.com/verygreen/lustretester>

Sample interaction



Neil Brown	Uploaded patch set 1.	Sep 18 2:53 AM
WC Checkpatch	Patch Set 1: Looks good to me.	Sep 18 2:55 AM
Misc Cod... helper)	Patch Set 1: Cannot build patch due to 14, messages: In file included from /home/green/git/lustre-release/libcfs/include/libcfs/libcfs.h:45:0, from /home/...	Sep 18 2:55 AM
Lustre Gerrit Janitor		Sep 18 2:57 AM ↩
Patch Set 1:		
(1 comment)		
centos7: Compile failed		
Job output URL: http://testing.linuxhacker.ru:3333/lustre-reports/3019/results.html		
lustre/quota/qmt_lock.c		
Line 760:	note: 'rc' was declared here	
Neil Brown	Uploaded patch set 2.	Sep 18 2:58 AM
jenkins	Patch Set 1: Verified-1 Build Failed https://build.whamcloud.com/job/lustre-reviews-patchless/8529/ : ABORTED https://build.whamcloud.com/job/lustre-reviews-patchless/8529/	Sep 18 2:58 AM
WC Checkpatch	Patch Set 2: Looks good to me.	Sep 18 3:00 AM
Misc Cod... helper)	Patch Set 2: Code-wise looks good to me.	Sep 18 3:01 AM
Lustre G... Janitor	Patch Set 2: Builds for x86_64 centos7,rhel8.0 successful Job output URL: http://testing.linuxhacker.ru:3333/lustre-reports/3020/results.html Commenc...	Sep 18 3:02 AM
Lustre G... Janitor	Patch Set 2: Initial testing failed: > runtests@ldisks+DNE Server crashed(35s) > runtests-ssk@ldisks+SharedKey Server crashed(155s) - (Untriaged ...	Sep 18 3:06 AM
Neil Brown	Uploaded patch set 3.	Sep 18 3:16 AM
jenkins	Patch Set 2: Verified-1 Build Failed https://build.whamcloud.com/job/lustre-reviews-patchless/8530/ : ABORTED https://build.whamcloud.com/job/lustre-reviews-patchless/8530/	Sep 18 3:16 AM
Misc Cod... helper)	Patch Set 3: Code-wise looks good to me.	Sep 18 3:19 AM
Lustre G... Janitor	Patch Set 3: Builds for x86_64 centos7,rhel8.0 successful Job output URL: http://testing.linuxhacker.ru:3333/lustre-reports/3021/results.html Commenc...	Sep 18 3:20 AM
WC Checkpatch	Patch Set 3: Looks good to me.	Sep 18 3:20 AM
Lustre Gerrit Janitor		Sep 18 3:22 AM ↩
Patch Set 3:		
(1 comment)		
Crash (id 1543 seen 0) in runtests@ldisks+DNE		
<ul style="list-style-type: none"> Failed run: http://testing.linuxhacker.ru:3333/lustre-reports/3021/testresults/runtests-ldisks-DNE-centos7_x86_64-centos7_x86_64 		
lustre/lod/lod_dev.c		
Line 476:	Crash with latest lustre function lod_sub_recovery_thread in backtrace called here:	
<pre> LustreError: 8874:0:(lu_object.c:1767:lu_context_fini()) ASSERTION(list_empty(&ctx->lc_remember)) failed: LustreError: 8874:0:(lu_object.c:1767:lu_context_fini()) LBUG Pid: 8874, comm: lod0000_rec0000 3.10.0-7.6-debug #1 SMP Fri Jul 12 02:40:17 EDT 2019 Call Trace: [<ffffffffa01928bc>] libcfs_call_trace+0x8c/0xc0 [libcfs] [<ffffffffa019296c>] lbug_with_loc+0x4c/0xa0 [libcfs] [<ffffffffa032d1aa>] lu_context_fini+0x16a/0x1a0 [obdclass] [<ffffffffa032d41a>] lu_env_fini+0x1a/0x30 [obdclass] [<ffffffffa0d8ccda>] lod_sub_recovery_thread+0x34a/0xb10 [lod] [<ffffffff810b4ed4>] kthread+0xe4/0xf0 [<ffffffff817c8c5d>] ret_from_fork_nospec_begin+0x7/0x21 [<ffffffffffffffff>] 0xffffffffffffffff </pre>		
Lustre G... Janitor	Patch Set 3: Initial testing failed: > runtests@ldisks+DNE Server crashed(45s) - (Untriaged #1543, seen 0 times before) > runtests-ssk@ldisks+Share...	Sep 18 3:24 AM



Chris Horn	Uploaded patch set 10: Commit message was updated.	Sep 20 2:08 PM
WC Checkpatch	Patch Set 10: Looks good to me.	Sep 20 2:12 PM
Misc Cod... helper)	Patch Set 10: Code-wise looks good to me.	Sep 20 2:17 PM
Lustre Gerrit Janitor		Sep 20 2:20 PM ↕
Patch Set 10: Builds for x86_64 centos7,rhel8.0 successful Job output URL: http://testing.linuxhacker.ru:3333/lustre-reports/3135/results.html Commencing initial testing: runtests@ldiskfs+DNE runtests-ssk@ldiskfs+SharedKey runtests@zfs		
Lustre Gerrit Janitor		Sep 20 2:30 PM ↕
Patch Set 10: Initial testing succeeded. Succeeded: <ul style="list-style-type: none">runtests@ldiskfs+DNE runtests-ssk@ldiskfs+SharedKey runtests@zfs (centos7)All results and logs: http://testing.linuxhacker.ru:3333/lustre-reports/3135/results.html Commencing standard testing: <ul style="list-style-type: none">conf-sanity1@ldiskfs+DNE conf-sanity2@ldiskfs+DNE conf-sanity3@ldiskfs+DNE conf-sanity-slow@ldiskfs+DNE conf-sanity1@zfs conf-sanity2@zfs conf-sanity3@zfs conf-sanity-slow@zfs insanity@ldiskfs+DNE insanity@zfs lnet-selftest@zfs lustre-rsync-test@ldiskfs+DNE lustre-rsync-test@zfs ost-pools@ldiskfs+DNE ost-pools@zfs racer@ldiskfs+DNE racer@zfs recovery-small@ldiskfs+DNE recovery-small@zfs replay-dual@ldiskfs+DNE replay-dual@zfs replay-ost-single@ldiskfs+DNE replay-ost-single@zfs replay-single@ldiskfs+DNE replay-single@zfs replay-vbr@ldiskfs+DNE replay-vbr@zfs sanity1@ldiskfs+DNE sanity2@ldiskfs+DNE sanity-slow@ldiskfs+DNE sanity1@zfs sanity2@zfs sanity-slow@zfs sanity-benchmark@ldiskfs+DNE sanity-benchmark@zfs sanity-dom@ldiskfs+DNE sanity-dom@zfs sanity-flr@ldiskfs+DNE sanity-flr@zfs sanity-hsm@ldiskfs+DNE sanity-hsm@zfs sanity-lfsck@ldiskfs+DNE sanity-lfsck@zfs sanity-lnet@zfs sanity-pcc@ldiskfs+DNE sanity-pcc@zfs sanity-pfi@ldiskfs+DNE sanity-pfi@zfs sanity-quota@ldiskfs+DNE sanity-quota@zfs sanity-scrub@ldiskfs+DNE sanity-scrub@zfs sanity-sec@ldiskfs+DNE sanity-sec@zfs sanityn@ldiskfs+DNE sanityn@zfs		
Lustre Gerrit Janitor		Sep 20 5:08 PM ↕
Patch Set 10: Even though "Test-Parameters: trivial" was detected, this deeply suspicious bot still run some testing Testing has completed with errors! IMPORTANT: these tests appear to be new failures unique to this patch <ul style="list-style-type: none">sanity1@zfs:test_42d(NEW previously unseen failure for this test) > lustre-rsync-test@zfs Timeout(1011s) > replay-dual@ldiskfs+DNE Timeout(619s) > replay-dual@zfs Timeout(629s) > replay-single@ldiskfs+DNE Failure(9437s) <ul style="list-style-type: none">110f(1 != 2 after recovery)> sanity2@ldiskfs+DNE Failure(5137s)300a(1:stripe_count is 1, expect 2)> sanity1@zfs Failure(2750s)42d(failed: client:53215232 server: 54919168.)> sanity-dom@ldiskfs+DNE Timeout(1211s)> sanity-lfsck@zfs Failure(1260s)23b(9) Fail to repair dangling name entry: 0)> sanity-pcc@zfs Failure(692s)16(request on 0x200000401.0x6a.0x0 is not SUCCEED on mds1)> sanity-quota@ldiskfs+DNE Failure(3101s)65(failed to write)> sanity-quota@zfs Failure(2844s)65(failed to write) Succeeded: <ul style="list-style-type: none">conf-sanity1@ldiskfs+DNE conf-sanity2@ldiskfs+DNE conf-sanity3@ldiskfs+DNE conf-sanity-slow@ldiskfs+DNE conf-sanity1@zfs conf-sanity2@zfs conf-sanity3@zfs conf-sanity-slow@zfs insanity@ldiskfs+DNE insanity@zfs lnet-selftest@zfs lustre-rsync-test@ldiskfs+DNE ost-pools@ldiskfs+DNE ost-pools@zfs racer@ldiskfs+DNE racer@zfs recovery-small@ldiskfs+DNE recovery-small@zfs replay-ost-single@ldiskfs+DNE replay-ost-single@zfs replay-single@ldiskfs+DNE replay-vbr@ldiskfs+DNE replay-vbr@zfs sanity1@ldiskfs+DNE sanity-slow@ldiskfs+DNE sanity2@zfs sanity-slow@zfs sanity-benchmark@ldiskfs+DNE sanity-benchmark@zfs sanity-dom@zfs sanity-flr@ldiskfs+DNE sanity-flr@zfs sanity-hsm@ldiskfs+DNE sanity-hsm@zfs sanity-lfsck@ldiskfs+DNE sanity-lfsck@zfs sanity-lnet@zfs sanity-pcc@ldiskfs+DNE sanity-pcc@zfs sanity-pfi@ldiskfs+DNE sanity-pfi@zfs sanity-scrub@ldiskfs+DNE sanity-scrub@zfs sanity-sec@ldiskfs+DNE sanity-sec@zfs sanityn@ldiskfs+DNE sanityn@zfs (centos7)All results and logs: http://testing.linuxhacker.ru:3333/lustre-reports/3135/results.html		
jenkins	Patch Set 10: Build Successful https://build.whamcloud.com/job/lustre-reviews-patchless/8643/ : SUCCESS https://build.whamcloud.com/job/lustre-revi...	Sep 20 5:13 PM

Test	Status/results	Extra info
runtests@ldiskfs+DNE	Success(305s)	
runtests-ssk@ldiskfs+SharedKey	Success(541s) (Client: sleeping in atomic,scheduling in atomic)	
runtests@zfs	Success(275s)	

Comprehensive testing: Failure

Test	Status/results	Extra info
conf-sanity1@ldiskfs+DNE	Success(3703s)	5f(needs separate mgs and mds) 21d(need separate mgs device) 21e(skipping excluded test 21e) 24a(mixed loopback and real device not working) 24b(mixed loopback and real device not working)
conf-sanity2@ldiskfs+DNE	Success(6357s)	33a(mixed loopback and real device not working) 36(remote OST) 43b(mixed loopback and real device not working) 45(skipping SLOW test 45) 55(skipping excluded test 55) 56b(needs >= 3 MDTs) 67(skipping excluded test 67) 69(skipping SLOW test 69) 71a(needs separate MGS/MDT) 71b(needs separate MGS/MDT) 71c(needs separate MGS/MDT) 71d(needs separate MGS/MDT) 71e(needs separate MGS/MDT)
conf-sanity3@ldiskfs+DNE	Success(3673s)	77(mixed loopback and real device not working) 81(needs >= 3 OSTs) 82a(needs >= 3 OSTs) 82b(needs >= 4 OSTs) 93(needs >= 3 MDTs) 102(skipping excluded test 102) 106(skipping SLOW test 106) 108a(zfs only test) 110(skipping ALWAYS excluded test 110) 111(skipping SLOW test 111) 115(This version of debugfs doesn't show inode number) 124(needs MDT failover setup)
conf-sanity-slow@ldiskfs+DNE	Success(7499s)	32b(skipping excluded test 32b) 32c(skipping excluded test 32c) 111(ETA 7083s after 120000 files / 168s is too long)
conf-sanity1@zfs	Success(3025s)	5f(needs separate mgs and mds) 17(ldiskfs only test) 18(ldiskfs only test) 21d(need separate mgs device) 21e(skipping excluded test 21e) 28a(LU-4221: no such proc params for ZFS OSTs)
conf-sanity2@zfs	Success(4440s) (Server: Memory Leaks Detected)	36(remote OST) 37(ldiskfs only test) 45(skipping SLOW test 45) 50i(needs >= 2 MDTs) 52(ldiskfs only test) 54a(ldiskfs only test) 54b(ldiskfs only test) 55(skipping excluded test 55) 56b(needs >= 3 MDTs) 60(ldiskfs only test) 62(ldiskfs only test) 63(ldiskfs only test) 65(ldiskfs only test) 67(skipping excluded test 67) 69(skipping SLOW test 69) 70a(needs >= 2 MDTs) 70b(needs >= 2 MDTs) 70c(needs >= 2 MDTs) 70d(needs >= 2 MDTs) 70e(needs >= 2 MDTs) 71a(needs >= 2 MDTs) 71b(needs >= 2 MDTs) 71c(needs >= 2 MDTs) 71d(needs >= 2 MDTs) 71e(needs >= 2 MDTs) 72(ldiskfs only test)
conf-sanity3@zfs	Success(2804s)	78(ldiskfs only test) 81(needs >= 3 OSTs) 82a(needs >= 3 OSTs) 82b(needs >= 4 OSTs) 83(ldiskfs only test) 86(LU-6442: no such mkfs params for ZFS OSTs) 87(ldiskfs only test) 88(LU-6662: no implementation for ZFS) 93(needs >= 3 MDTs) 99(ldiskfs only test) 102(skipping excluded test 102) 106(skipping SLOW test 106) 108b(ldiskfs only test) 109a(LU-8727: no implementation for ZFS) 109b(LU-8727: no implementation for ZFS) 110(skipping ALWAYS excluded test 110) 111(skipping SLOW test 111) 115(Only applicable to ldiskfs-based MDTs) 116(ldiskfs only test) 120(mdt count < 2) 122(needs >= 2 MDTs) 124(needs >= 2 MDTs) 125(ldiskfs only test)
conf-sanity-slow@zfs	Success(6100s)	32b(skipping excluded test 32b) 32c(skipping excluded test 32c) 111(Only applicable to ldiskfs-based MDTs)
insanity@ldiskfs+DNE	Success(1841s)	
insanity@zfs	Success(1103s)	1(needs >= 2 MDTs) 10(needs >= 2 MDTs) 11(needs >= 2 MDTs) 12(needs >= 2 MDTs) 13(needs >= 2 MDTs) 14(needs >= 2 MDTs)
lnet-selftest@zfs	Success(431s)	
lustre-rsync-test@ldiskfs+DNE	Success(993s)	2b(skipping ALWAYS excluded test 2b) 4(iozone not found)
lustre-rsync-test@zfs	Timeout(1011s)	
ost-pools@ldiskfs+DNE	Success(2428s)	12(needs >=3 OSTs) 13(needs >= 3 OSTs) 14(needs >= 3 OSTs) 26(needs >= 3 OSTs)
ost-pools@zfs	Success(2598s)	12(needs >=3 OSTs) 13(needs >= 3 OSTs) 14(needs >= 3 OSTs) 26(needs >= 3 OSTs)
racer@ldiskfs+DNE	Success(482s)	
racer@zfs	Success(472s)	
recovery-small@ldiskfs+DNE	Success(4686s)	10e(need two clients) 17b(Needs multiple clients) 26a(msg and ost1 are at the same node) 26b(msg and ost1 are at the same node) 103(needs separate mgs and mds) 105(Needs multiple clients) 134(Need 2+ clients, have 1) 136(skipping excluded test 136)

The birth of the Boilpot

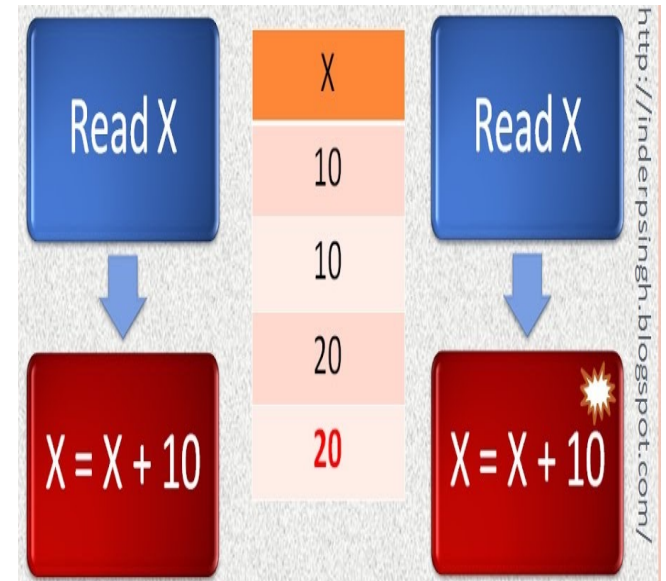
- ▶ To catch those “one in a billion” race conditions.
- ▶ Created after a fateful problem experienced at ORNL that only hit at largest scale twice a month
- ▶ It was clear we need to be able to tackle this on less important, smaller systems somehow
- ▶ A stroke of luck happenstance led to creating a bunch of VMs on a single host

Multithreaded programming



The wonders of CPU overcommit

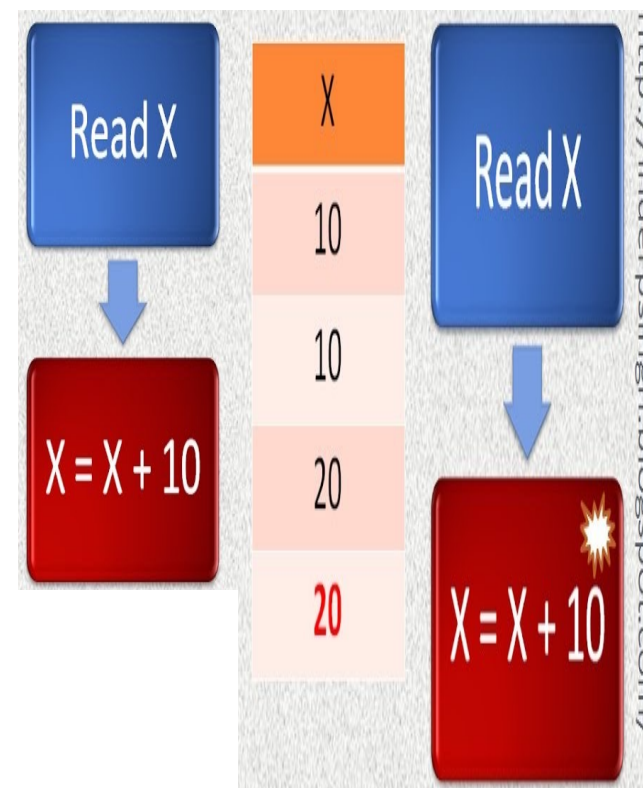
- ▶ When you have multiple VMs competing for limited CPU cycles, host OS stops the “cpu thread” at random to let other VMs run.
- ▶ It’s obvious in hindsight, but this is the big part of the success of this technique:
 - Inside the VM all CPUs appear normal
 - But externally they are stopped for random time at random intervals, while others keep running
 - This leads to great extension of race windows.
 - Even a single instruction race that is incredibly hard to hit normally, becomes very possible the more overcommit is exposed
- ▶ For this to work well you need some heavy CPU load present somehow. Ideally in the VMs themselves
- ▶ Important distinction here is then you need lots of RAM too, as otherwise VMs are swapped out and generally all sorts of kernel protection mechanisms get into play





The wonders of CPU overcommit

- ▶ When you have multiple VMs competing for limited CPU cycles, host OS stops the “cpu thread” at random to let other VMs run.
- ▶ It’s obvious in hindsight, but this is the big part of the success of this technique:
 - Inside the VM all CPUs appear normal
 - But externally they are stopped for random time at random intervals, while others keep running
 - This leads to great extension of race windows.
 - Even a single instruction race that is incredibly hard to hit normally, becomes very possible the more overcommit is exposed
- ▶ For this to work well you need some heavy CPU load present somehow. Ideally in the VMs themselves
- ▶ Important distinction here is then you need lots of RAM too, as otherwise VMs are swapped out and generally all sorts of kernel protection mechanisms get into play



Boilpot Era

- ▶ The new setup yielded a crash about every 20 minutes
 - ORNL specific crash amongst them
- ▶ The newly found opportunity was too good to pass up
 - The boiling pot was born
 - Ad-hoc at first it became a staple of integration testing quickly
- ▶ Time to crash started to rise
 - Eventually the metric became number of crashes per day, then per week
- ▶ Overall Lustre stability rose correspondingly

- ▶ I tried the same approach on in-kernel NFS
 - Immediately triggered a number of crashes
 - Yielded some fun comments from the kernel big wigs questioning if anybody is even using NFS for the past few years
- ▶ Now a staple of Lustre integration testing called “boilingpot”

Test suite fragmentation and monoculture

- ▶ **Out of sight – out of mind**
 - That’s how we can best describe the “non-binding” full testing
 - If it’s not in enforced review testing – it will break. Probably already broken and nobody noticed yet
- ▶ **Strong enforcement of “all green” results is key to quality**
 - Some people think “it’s ok to mark known failures”, but I think they are misguided
- ▶ **Even with that in place, surprising breakage arose at times**
- ▶ **Tests and code were becoming “Fine-tuned” to just run in the particular maloo config**
 - Change the config and suddenly all sorts of bugs crop out
- ▶ **This was partially addressed by the “boilpot” being a vastly different setup**
 - Waay too expensive being run as the very last step before landing the patch

- ▶ **We are working with big sites like ORNL to test at real scale in real environments**
- ▶ **You too can run some testing and report to us for the benefit of everybody.**



Whamcloud





Whamcloud

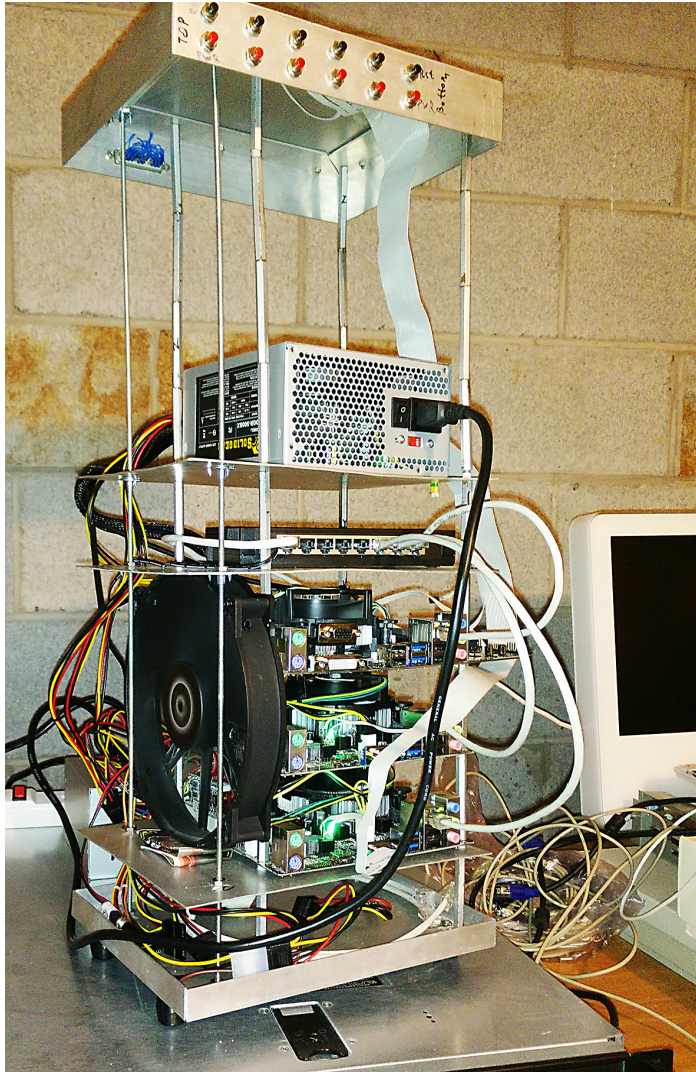
Overflow



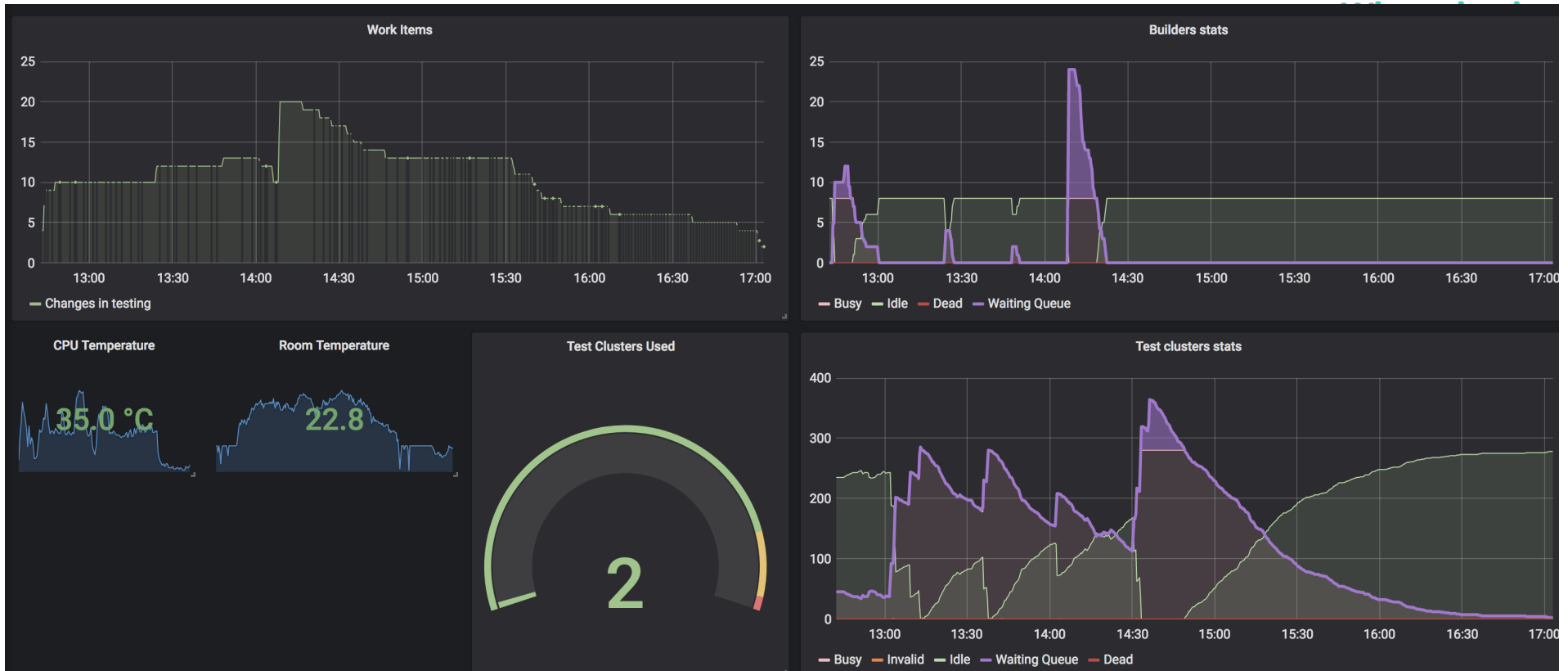
Testing at scale with minimal resources

- ▶ 2 hour turn-around time goal, a pie in the sky?
 - Split testing into one session per testscript
 - The long testscripts we have, split them into parts
 - Lots of VMs to run testscripts in parallel.
 - How many is “lots”?
 - Single build starts $27 * 2 + 1 = 55$ sessions
 - 2 nodes per session at 4G RAM per node = 440G
 - We need servers with lots of RAM
 - We want at least 4 sessions running in parallel
 - At least 200 VMs
 - Will everything in place currently testing takes $\sim 2:40 + 10$ minutes
- ▶ Old opencompute nodes are cheap: \$100 for 2 with chassis
 - + 4x E5-2660v2 (10 cores) = \$400 + 512G RAM = \$1000
 - \sim \$1.6k for 80 parallel test sessions

Setup v1



Utilization



Debug kernels – the other important ingredient

- ▶ Linux kernel provides a bunch of extra debug mechanisms to ease development of kernel code
- ▶ Some of it are really expensive, some – not so much. Some you must build with some you can turn on at runtime.
- ▶ Of the very important ones:
 - `DEBUG_PAGEALLOC` - really slow, but most freed memory access, even read-only results in a crash
 - Sleeping while atomic detection – shows problematic locking before it becomes a real problem
- ▶ Alas, it turns out not many developers run in this setup
 - This includes distro developers
 - RedHat views this config as “unsupported” and often does not take bugreports
 - Took me quite some time to convince them that yes, their paravirt spinlock implementation is broken even if I don’t have a good reproducer outside of the debug kernel

Importance of easy access to information by devs

- ▶ Another sore point is getting developers everything they need and more at a glance
 - Lustre is a complex system, it produces a lot of logs from multiple nodes during testing
 - Physically infeasible for everybody to review every single line of them
- ▶ Strong search and cross reference abilities is a must
 - What successful tests produce error messages?
 - “command not found”, “invalid syntax”, “file not found”,
 - Way too many as it turns out
- ▶ Crash information
 - Automating gathering of useful information from crashdumps to save time
- ▶ Automated triaging of issues based on all the above and more
 - To better highlight new problems

The beginning

- ▶ In 2008 ORNL reported a strange MDS race condition-crash happening about once every 2 weeks
 - Only happened during some very heavy filesystem activity.
 - Complicated to collect debug data
 - Did not want the crashes to repeat due to all the downtime
- ▶ It was clear we need to be able to tackle this on less important systems somehow
- ▶ One route was load/client simulator.
 - This is now known as MDS echo client/mds-survey set of scripts
- ▶ The other – use racer (obviously) and try to load a single VM with it with modest number of clients.
 - A lucky stroke here was also about locating many of the VMs on the same host and HT was also enabled
 - Crashes came relatively quickly and the issue was identified relatively fast after that.



What I learned

- ▶ **People take the path of least resistance**
 - Boy oh boy was the CMU “TSP” course misguided!
 - Always assume the worst and try to use automation to guard against it
- ▶ **Don't decouple QA and developers**
 - They are different people with different goals.
 - They often have different ideas of what's needed and what's not and how much is it needed
 - They have different ideas about what's possible and what's not.

Quick compilation – mission possible

- ▶ Many areas of build process are single threaded – a bunch of parallel cpus does not help
- ▶ Configure process for lustre is very long
 - Centos7 – 3 minutes, rhel8 – 9 minutes(!)
 - Solution: cache configure results across runs if nothing in autoconf files changed (use md5)
- ▶ RPM generation is slow
 - Skip rpm generation, instead just create squashfs image of build tree to run out of
 - Uses multiple CPU threads
- ▶ End result: 15-20 minute build time reduced to usually 1-2 minutes

Failure rate tracking

- ▶ To track flaky tests – record every failure for later comparison.
 - Test, subtest, failure message text, fstype
- ▶ Add “same failure” output to failed results
 - Helps people to better gauge if the failure is likely theirs or not
- ▶ Does not work all that well for tests with variable error messages (duh!)

Crash information extraction

- ▶ Crashdumps host a whole bunch of useful data, but it's hard to get to it
 - Need to grab debug binaries, have right tools compiled, find and download the crash dump,...
- ▶ Save time! Every crash (and timeout) gets automatic processing:
 - Extract backtraces of all tasks
 - Cross reference the crash backtrace against a database of known crashes
 - Extract Lustre debug logs
 - TBD: extract lock state and memory information
 - Thanks Cray for contributed pycrash scripts.

Recognizing the known crashes

- ▶ Same crashes have often somewhat different backtraces
 - Different addresses, different garbage on the stack, ...
- ▶ Unique elements:
 - The crashing reason: GPF/NULL pointer, OOM, NMI, ...
 - Crashing function name
 - Stable backtrace with function names only, addresses stripped
 - Test name (if any)
- ▶ Additional useful elements for additional testing
 - All kernel messages since start of last test
 - Unabbreviated backtrace



Whamcloud

fatbox1.localnet/cgi-bin/crashdb_ui.py.cgi?newid=617

Reason	Crashing Function	Where to cut Backtrace	Reports Count
BUG: unable to handle kernel paging request	lu_object_put	<input type="radio"/> tgt_request_handle <input type="radio"/> ptrrpc_server_handle_request <input type="radio"/> ptrrpc_main <input type="radio"/> kthread	35

Added fields:

Match messages in logs
(every line would be required to be present in log output
Copy from "Messages before crash" column below):

Match messages in full crash
(every line would be required to be present in crash log output
Copy from "Full Crash" column below):

Limit to a test:
(Copy from below "Failing text"):

Delete these reports as invalid (real bug in review or some such)

Bug or comment:

Extra info:

Add to Known bugs

Failures list (last 100):

Failing Test	Full Crash	Messages before crash	Comment
racer test 1: racer on clients: centos-0.localnet DURATION=2700	CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080030035 CR2: ffff880325208e60 CR3: 00000002db74e000 CR4: 00000000000006e0 Call Trace: [<fffffafa0644526>] tgt_request_handle+0x236/0x1590 [ptrrpc] [<fffffafa0217fa7>] ? libcfs_debug_msg+0x57/0x80 [libcfs] [<fffffafa05e8e86>] ptrrpc_server_handle_request+0x256/0xad0 [ptrrpc] [<fffffafa10bfbdb8>] ? __wake_up_common+0x58/0x90 [<fffffafa13fb7bb>] ? do_raw_spin_unlock+0x4b/0x90 [<fffffafa05eed79>] ptrrpc_main+0xa99/0x1f60 [ptrrpc] [<fffffafa10c32ed>] ? finish_task_switch+0x5d/0x1b0 [<fffffafa17b6cd0>] ? __schedule+0x410/0xa00 [<fffffafa05ec2e0>] ? ptrrpc_register_service+0xfb0/0xfb0 [ptrrpc] [<fffffafa10b4ed4>] kthread+0xe4/0xf0 [<fffffafa10b4df0>] ? kthread_create_on_node+0x140/0x140 [<fffffafa17c4c77>] ? ret_from_fork_nospec_begin+0x21/0x21 [<fffffafa10b4df0>] ? kthread_create_on_node+0x140/0x140	Lustre: ifs: using old ioctl(LL_IOC_LOV_GETSTRIP) on [0x200000402:0x6b:0x0], use llapi_layout_get_by_path()	Externally reported by onyx-68 boilpot email
racer test 1: racer on clients: centos-115.localnet	BUG: unable to handle kernel paging request at ffff880232db9e60 IP: [<fffffafa03bb230>] lu_object_put+0x270/0x3c0 [obdclass] PGD 241b067 PUD 33effc067 PMD 33ee65067 PTE 8000000232db9060 Oops: 0000 [#1] SMP DEBUG_PAGEALLOP Modules linked in: lustre(OE) ofd(OE) osp(OE) lod(OE) ost(OE) mdt(OE) mdd(OE) mgs(OE) osd_zfs(OE) lquota(OE) ifsc(OE) obdecho(OE) mgc(OE) lov(OE) mdc(OE) osc(OE) lmv(OE) fld(OE) fld(OE) ptrrpc_gss(OE) ptrrpc(OE) obdclass(OE) ksocklnd(OE) lnet(OE) libcfs(OE)	Lustre: ifs: using old ioctl(LL_IOC_LOV_GETSTRIP) on [0x200000401:0xf:0x0], use llapi_layout_get_by_path() LustreError: 7149:0:(mdt_lvbc:c:430:mdt_lvbo_fill()) lustre-MDT0000: small buffer size 496 for EA 520 (max_mdsize 520): rc = -34 Lustre: DEBUG MARKER: racer test_1: @@@@ @@@@ FAIL: generate lss conf (mds1) 141995]: segfault at 8 ip 00007f2ca802d958 sp 00007ffd7bfe8c40 error 4 in ld-2.17.so[7f2ca8022000+22000] 7113769]: segfault at 8 ip 00007fa6d941d958 sp 00007ffc475e90 error 4 in ld-2.17.so[7fa6d9412000+22000] 0[25128]: segfault at 0 ip (null) sp 00007ffd95501d8 error 14 in 0[400000+6000] 8[467]: segfault at 8 ip 00007f80213c3958 sp 00007ffd6a63ea0 error 4 in ld-2.17.so[7f80213b8000+22000]	Externally reported by

Better context awareness

- ▶ **Did you ever forget to add Test-params?**
 - In majority of cases why do I even need to? If I only changed sanity.sh why run anything else?
- ▶ **Gerrit provides an easily accessible list of files changed – use it**
 - Create list of files to tests mapping
 - Build-only changes don't even need any tests
 - Areas we cannot test at all due to lack of hardware (Gemini LND)
 - ldiskfs-only, zfs-only, individual test-scripts
- ▶ **Now we can also guard against misguided “Test-Param: trivial” instances**
 - Sadly we've seen some abuse of that
- ▶ **Future stretch goals:**
 - Detect whitespace-only/comments-only changes
 - See individual tests added/changed and ensure they are run/ highlight when they fail