Keeping up with LTS Linux Kernel Functional Testing on Devices

Tom Gall

Director, Linaro Mobile Group

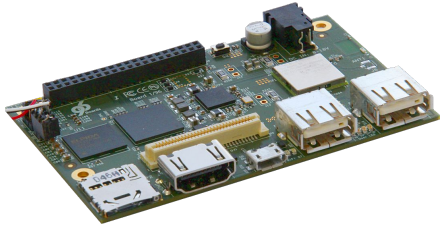
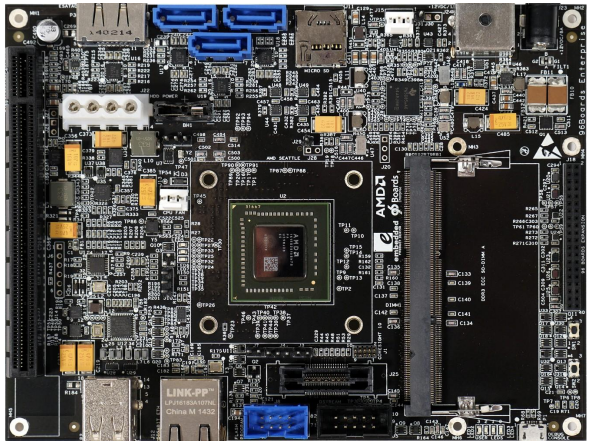
Linaro

Who is Linaro?

- Linaro is leading software collaboration in the ARM ecosystem
- Instead of duplicating effort, competitors share development costs to accelerate innovation and time to market
- Linaro is member funded and delivers output to members, and into open source projects





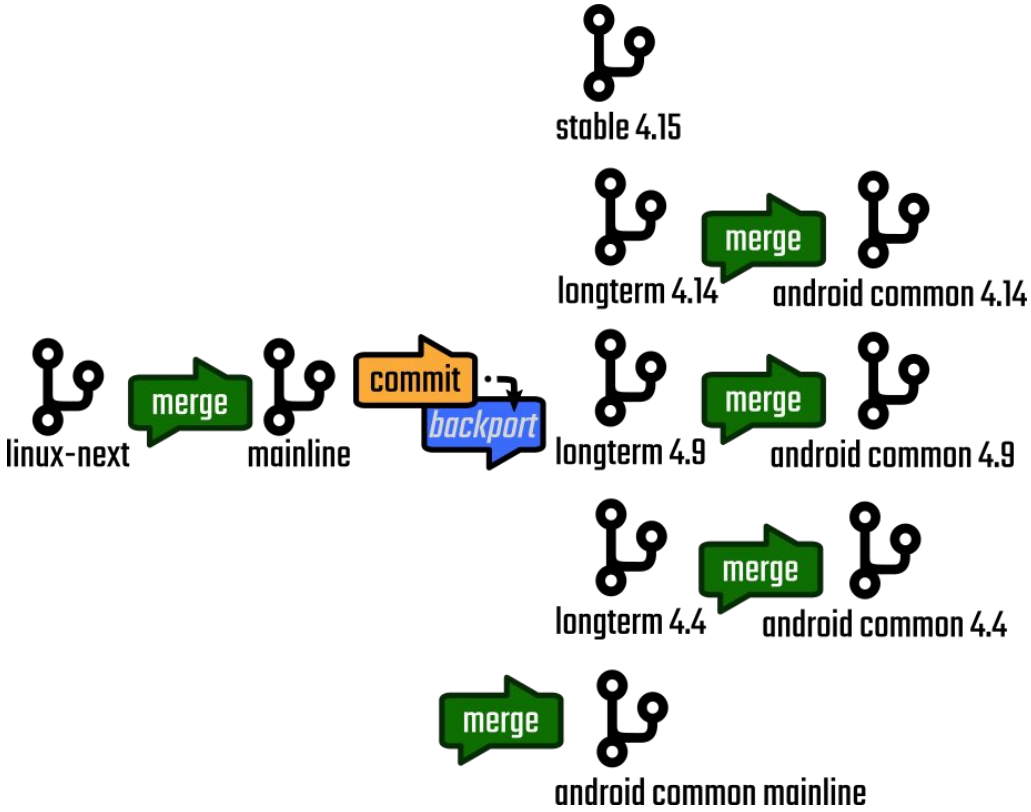
Open Source Project Contributions - Partial List

Linaro LMG Mobile

Linux Kernels on Devices

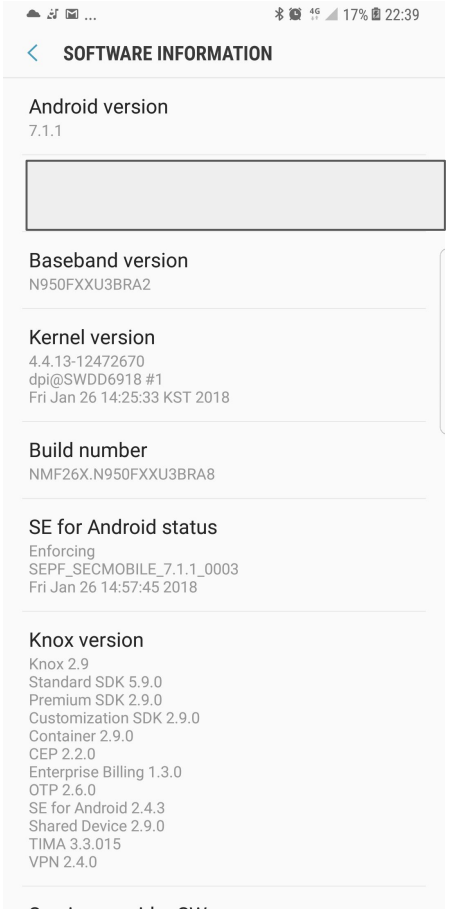
Quick review about upstream...

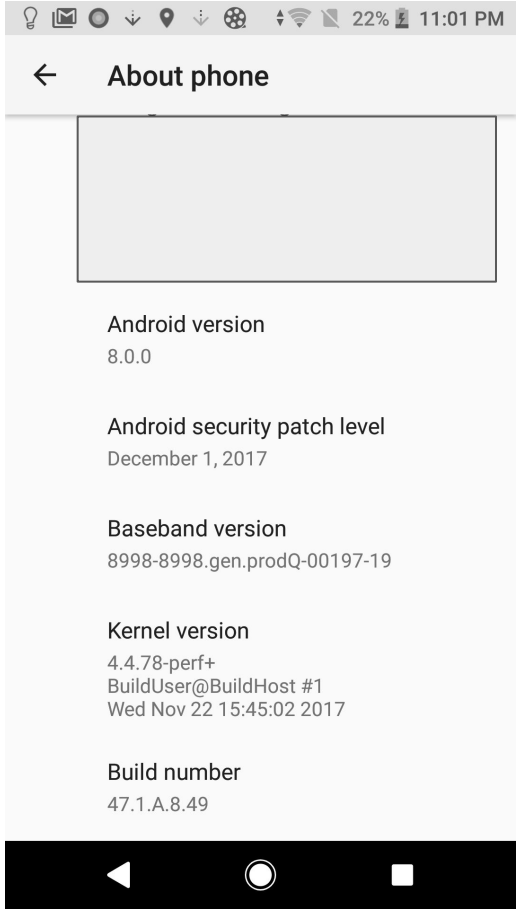
- Android Common
  - Tracks LTS (4.4, 4.9, 4.14)
  - Tracks Mainline



And then you see this...

- 4.4.13 is positively ancient
  - Released: June 8th 2016
- 4.4.78 better but
  - Released: July 21st 2017
- Security fixes are being cherry picked, however
  - LTS security fixes aren't necessarily labeled as security fixes
  - LTS tests with all patches in an LTS release, not some cherry pick
  - Cherry picking can entirely miss complicated interactions where other patches were required

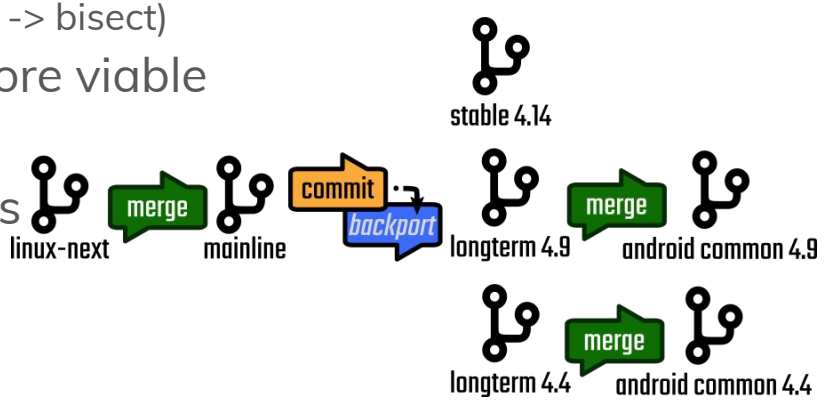




Project Sharp Introduction

Making Community and Android Kernels Better

- Catch kernel regressions across architectures and kernel versions before they make it into LTS releases or Android Common
  - 4.4, 4.9, 4.14, current stable, mainline
  - X86_64, ARMv7, ARMv8
  - GCC and soon clang
  - 48 hour window (build -> results -> triage -> bisect)
- Help make more older LTS kernels more viable
- Examine communities for fixes
- Display testing data and test histories
- Empower developers
- Triage problems
- Add to kernel testing effort



LKFT compared with KernelCI

LKFT and KernelCI will cautiously converge when/where it makes sense

LKFT

Functional Testing as a first-order design requirement

Full userspace

Functional Test Coverage

Limited hardware due to userspace requirements

Linaro Member Needs Driven

Linaro Member Goal Driven - Sharp, extend LTS, LSK testing, et al.

Does boot-test limited hardware

Limited only by Linaro & member development pace

KernelCI

Boot Testing as a first-order design requirement

Minimal Userspace

Boot Test quickly

Larger class of hardware supported

Community Consensus Driven

Linux Community Goal Driven

Can functional test w/ minimal userspace

Limited by pace of community consensus

Open Devices only

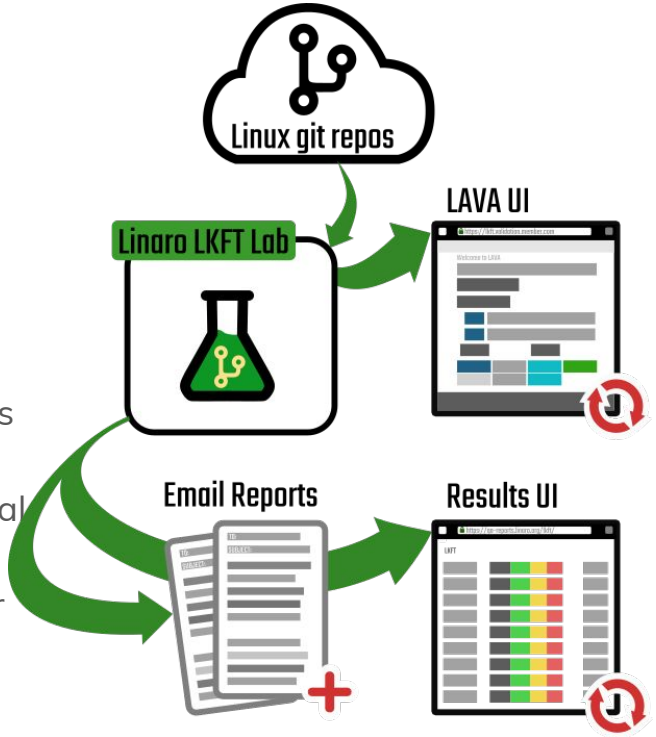
Cannot publish results under access control

LKFT

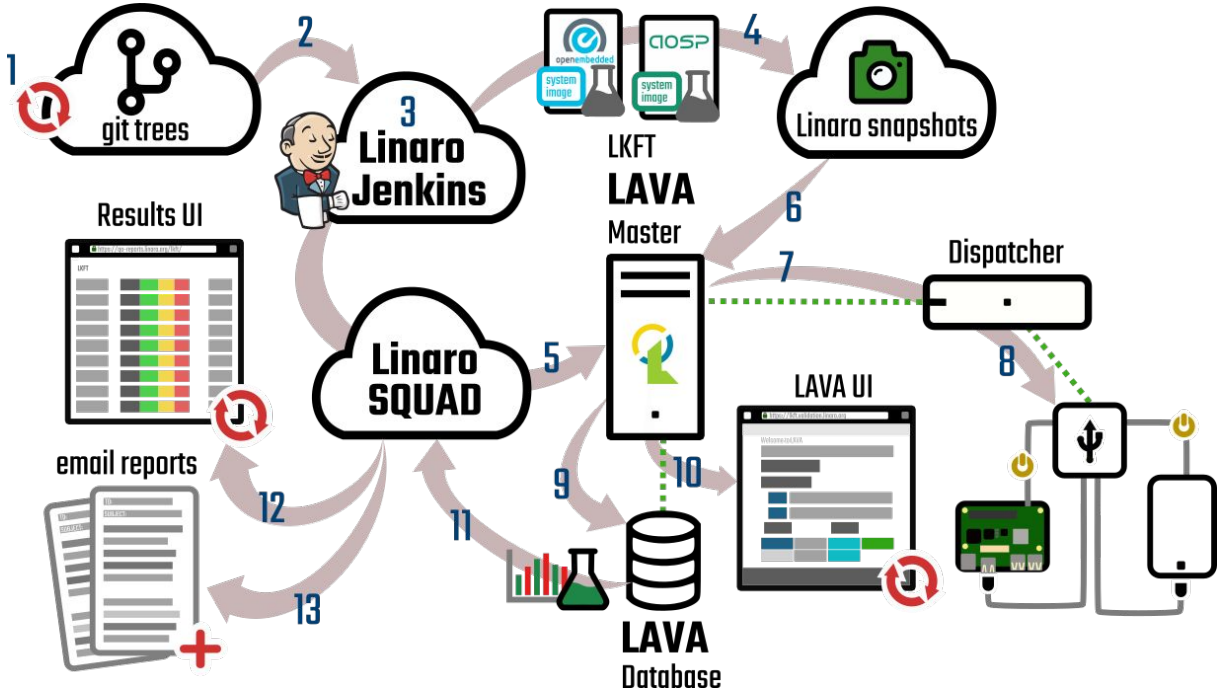
LKFT - Linux Kernel Functional Test framework.

The mission of LKFT is to perform functional regression testing on select Linux kernel branches in real time (as they're updated) and report any regressions as quickly as possible. This is performed by executing a variety of functional-tests on a selection of user-space environments such as Open Embedded and Android.

The goals of LKFT are to shorten derivative Linux kernel release intervals, increase the confidence of upstream Linux kernel engineers in the quality of their releases, and increase the confidence of downstream adopters of those Linux kernel trees. Ultimately the goal is that LKFT will encourage downstream hardware vendors to more frequently update the Linux kernel that runs on their devices in order for consumers to benefit from bug and security updates.



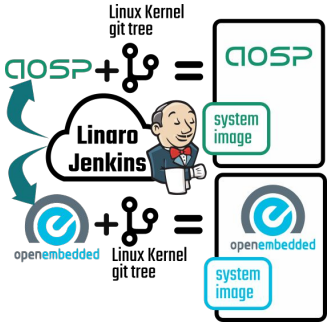
LKFT System Overview



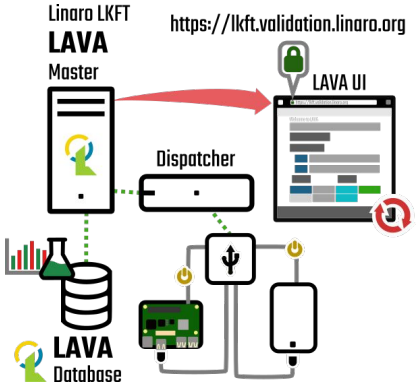
1. Upstream/Internal tree changes
2. Fetch git kernel tree repo
3. Build system images
4. Publish image builds to snapshot server
5. Submits jobs to the Labs (LAVA - Linaro Automation Validation Architecture)
6. LAVA request build download
7. Schedule jobs on target hardware
8. Perform tests on target hardware
9. Store results to LAVA database
10. Results made available on LAVA frontend
11. Qa-reports pulls Results data from LAVA database
12. Present results in qa-reports dashboard
13. Send Email reports

LKFT Infrastructure

The infrastructure for LKFT is composed of several autonomous components

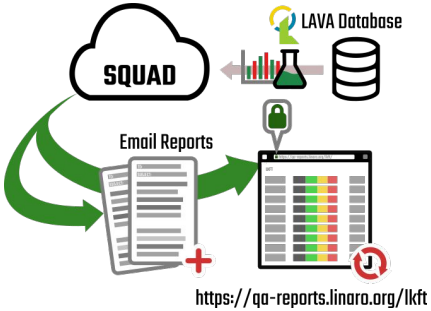


- Commit triggered image building by using a Jenkins instance to build OE & AOSP images and submit jobs to LAVA: https://ci.linaro.org/



- Device automation to support scheduling, image flashing, automated testing, and results gathering (and storage) via a dedicated LAVA instance: https://lkft.validation.linaro.org

- Email reporting and results dashboard via a dedicated Squad instance:
  https://qa-reports.linaro.org/lkft
  https://qa-reports.linaro.org/android-lkft



When an RC occurs

4.4, 4.9, 4.14, 4.15, mainline, next

- 1 build for each architecture/board combo
- 20 LAVA test jobs per kernel version
- 5572 individual tests per kernel version

What hardware is in use?

LAVA | Home | Results | Scheduler | API | Help | Instance: lkft | Sign In

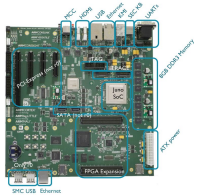
Scheduler Status

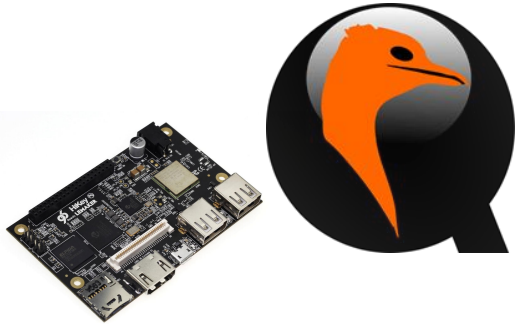
Overall status		Devices		TestJobs	
Online devices	44 / 50	All devices		All Jobs	
Passing health checks	50 / 50	Active Devices		Queued Jobs	
Running test jobs	5 / 5	Devices Health		Active Jobs	

Device Type Overview

Show 50 entries Search

Name	Idle	Offline	Busy	Restricted	Queue
b2260	3	1			
dragonboard-410c	4				
hi6220-hikey		2	5	1	2
juno-r2	4	1			
qemu	19	1			
x15	5	1		1	
x86	4				





Experience with Devices

- 96Boards an obvious ARM platform
  - Small form factor
  - Suited to large scale deployments
- Reliable connectivity costs money
  - High quality, shielded USB cables
  - Reliable, software controllable, USB hubs
- Firmware updates cost engineering time
  - Changes in interaction breaks automation
- Scaling up challenges
  - Four cables per board
    - Serial, USB OTG, Ethernet and power
    - Power bricks take space
    - Solutions being sought

kselftest - Linux Kernel Testing Framework

https://kselftest.wiki.kernel.org

- Use the latest stable version of the test against all LTS kernel releases
  - This was somewhat controversial
  - Can be challenging due to failures caused by mismatched versions
  - Upstream isn't always interested in running this combination or addressing issues discovered by it
- Up to various kernel maintainers to either use or ignore
- Testcase consistency (design, setup, running)
- Reporting infrastructure could be improved. (TAP13)
- Pushed many patches to improve testing infrastructure and address obvious bugs
- A good start to kernel testing, we'd like to see more focus on it's improvement

LTP - Linux Test Project

https://linux-test-project.github.io

- We don’t run the entire set due to suitability
  - 19 suites currently in use (syscalls, timers, ...)
- Test suite is updated every 4 months as per upstream releases (latest 20180118)
- We have a CI loop with LTP master running on mainline to improve future releases

Experiences with ‘complicated’ test suites

- Automation of test runs?
  - Running ‘tradefed family’ tests (VTS, CTS) requires host side.
  - Some LTP tests make hidden assumptions about the hardware they run on
  - Running pre-built version of kselftests brings a lot of compatibility issues
- Reporting?
  - There is no unified standard for reporting results/logs
  - VTS logs are reported differently than CTS even though they use the same shell (tradefed)
  - Kselftests logs are saved in /tmp
  - Kselftests apparently support TAP13, but not all tests implement this approach (*)
- Skipped tests
  - There are a lot of tests failing on arm/arm64
  - Tests make assumptions which are not always met (for example sources of entropy)

Experiences with Triaging Android

- Android Common has mainline, 4.4, 4.9, 4.14
  - A set of (decreasing in size) out of tree kernel patches are included in the mix
- On Android we don't run the exact same of tests as Open Embedded
  - LTP has a number of tests designed specifically for Linux
  - Dependencies not satisfied, etc
- VTS does run a subset of kselftest, LTP
- CTS is uniquely an Android testsuite
  - User space tests can push the kernel in interesting ways Ex: just using the network or BT
- Open Embedded (currently) leads the charge to look for kernel regressions, class of failures detected tend to be Android specific

Keeping up with LTS

Expectations

- 4.4, 4.9, 4.14 generally have 1, maybe 2 cycles per week
  - Couple dozen patches to couple hundred
- Patches included in RC have 48 hours
- Build -> Run -> Report Results
  - Triage Errors -> Bisect -> Fix
- Schedule is lose (on purpose!)
- RC branches are rebased frequently, making building and reproducibility tricky

Example : Pushing results upstream

Email!

Summary
---

kernel: 4.4.121
git repo: https://git.kernel.org/pub/scm/linux/kernel/git/stable/linux-stable-rc.git
git branch: linux-4.4.y
git commit: 8b5ab55d254f36e89b1b53aeac7223d2d102483e
git describe: v4.4.121
Test details: https://qa-reports.linaro.org/lkft/linux-stable-rc-4.4-oe/build/v4.4.121

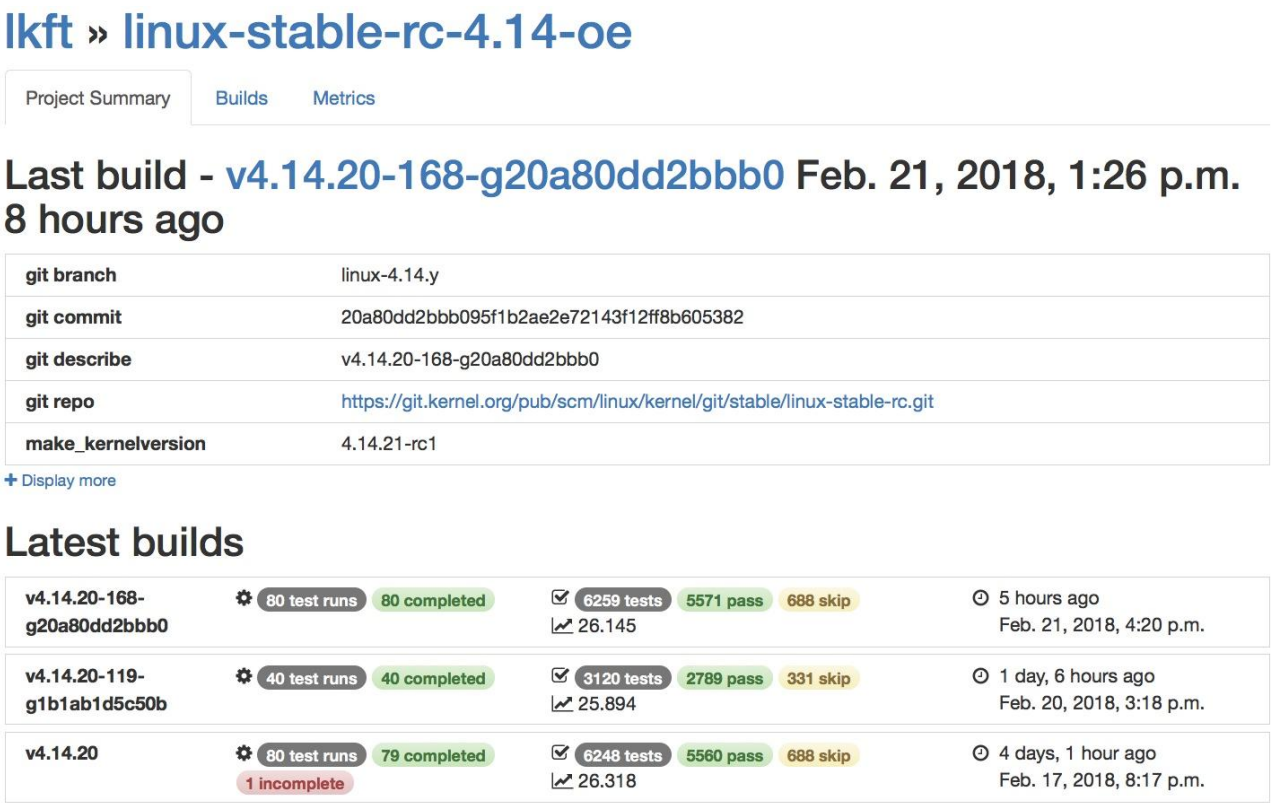
No regressions (compared to build v4.4.120-37-gce7ba34ae77c)

Boards, architectures and test suites:
---

juno-r2 - arm64
* boot - pass: 20,
* kselftest - pass: 34, skip: 29,
* libhugetlbfs - pass: 90, skip: 1,
* ltp-cap_bounds-tests - pass: 2,
* ltp-containers-tests - pass: 28, skip: 53,
* ltp-fcntl-locktests-tests - pass: 2,
* ltp-filecaps-tests - pass: 2,
* ltp-fs-tests - pass: 61, skip: 2,
* ltp-fs_bind-tests - pass: 2,
* ltp-fs_perms_simple-tests - pass: 19,
* ltp-fsx-tests - pass: 2,
* ltp-hugetlb-tests - pass: 22,
* ltp-io-tests - pass: 3,
* ltp-ipc-tests - pass: 9,
* ltp-math-tests - pass: 11,
* ltp-nptl-tests - pass: 2,
* ltp-pty-tests - pass: 4,
* ltp-sched-tests - pass: 10, skip: 4,

- stable@vger.kernel.org
- git repo: https://git.kernel.org/pub/scm/linux/kernel/git/stable/linux-stable-rc.git
- Goal: Quick summary
  - Or Bisected failure

Example : a test report, also available via email



lkft » linux-stable-rc-4.14-oe

Project Summary | Builds | Metrics

Last build - v4.14.20-168-g20a80dd2bbb0 Feb. 21, 2018, 1:26 p.m. 8 hours ago

git branch	linux-4.14.y
git commit	20a80dd2bbb095f1b2ae2e72143f12ff8b605382
git describe	v4.14.20-168-g20a80dd2bbb0
git repo	https://git.kernel.org/pub/scm/linux/kernel/git/stable/linux-stable-rc.git
make_kernelversion	4.14.21-rc1

+ Display more

Latest builds

v4.14.20-168-g20a80dd2bbb0	80 test runs, 80 completed	6259 tests, 5571 pass, 688 skip, 26.145	5 hours ago, Feb. 21, 2018, 4:20 p.m.
v4.14.20-119-g1b1ab1d5c50b	40 test runs, 40 completed	3120 tests, 2789 pass, 331 skip, 25.894	1 day, 6 hours ago, Feb. 20, 2018, 3:18 p.m.
v4.14.20	80 test runs, 79 completed, 1 incomplete	6248 tests, 5560 pass, 688 skip, 26.318	4 days, 1 hour ago, Feb. 17, 2018, 8:17 p.m.

Example : a test report

lkft » linux-stable-rc-4.14-oe

Project Summary | Builds | Metrics

Last build - v4.14.20-168-g20a80dd2bbb0 Feb. 21, 2018, 1:26 p.m. 8 hours ago

git branch	linux-4.14.y
git commit	20a80dd2bbb095f1b2ae2e72143f12ff8b605382
git describe	v4.14.20-168-g20a80dd2bbb0
git repo	https://git.kernel.org/pub/scm/linux/kernel/git/stable/linux-stable-rc.git
make_kernelversion	4.14.21-rc1

+ Display more

Latest builds

v4.14.20-168-g20a80dd2bbb0	80 test runs, 80 completed	6259 tests, 5571 pass, 688 skip, 26.145	5 hours ago, Feb. 21, 2018, 4:20 p.m.
v4.14.20-119-g1b1ab1d5c50b	40 test runs, 40 completed	3120 tests, 2789 pass, 331 skip, 25.894	1 day, 6 hours ago, Feb. 20, 2018, 3:18 p.m.
v4.14.18-23-g8d861f5b27b0	81 test runs, 80 completed, 1 incomplete	6258 tests, 5562 pass, 688 skip, 8 fail, 25.255	1 week, 5 days ago, Feb. 9, 2018, 9:16 p.m.

Example : Where is it failing?

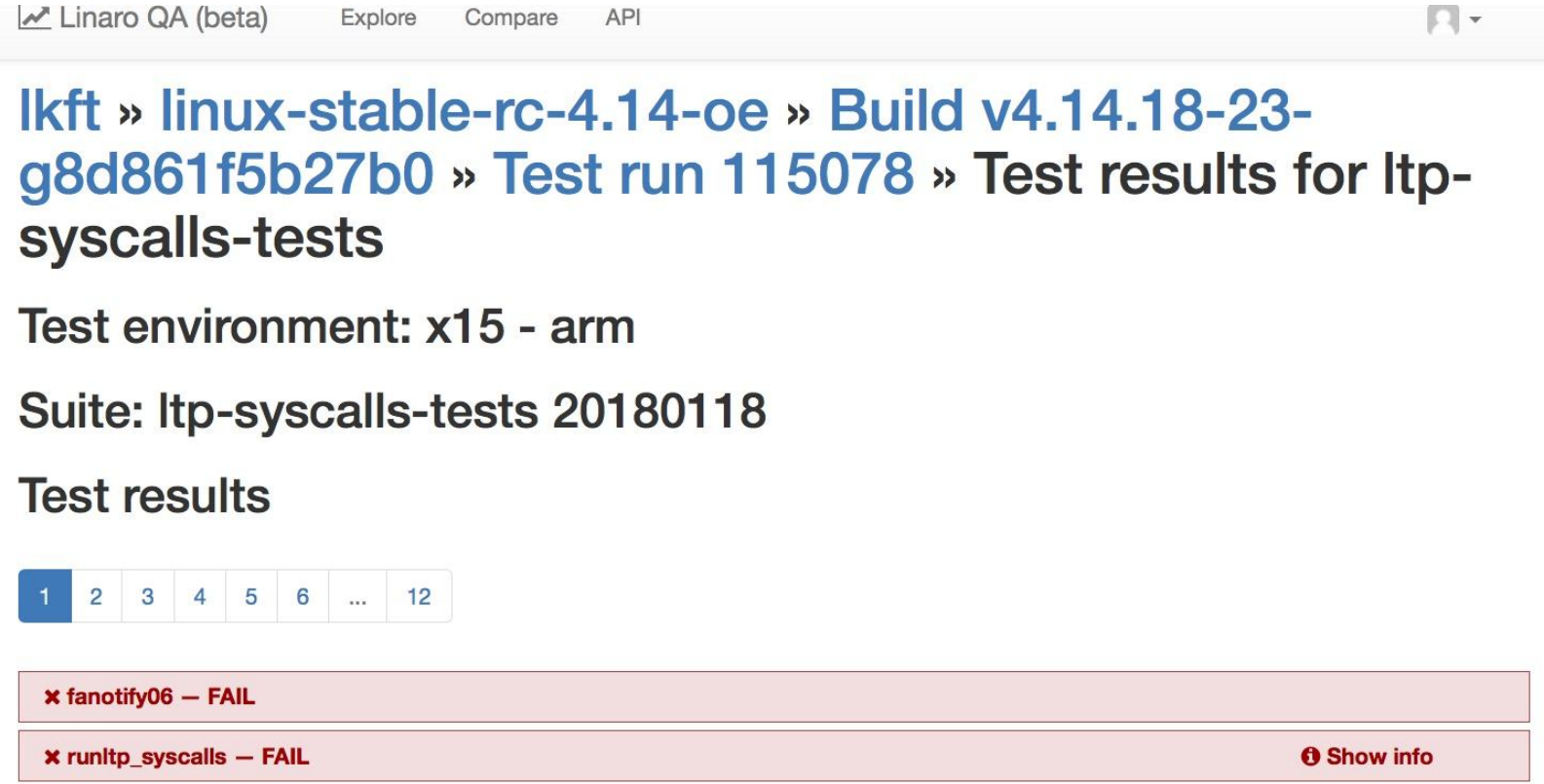
Test results

Filter results ...

ltp-syscalls-tests	x15 - arm	1150 tests	1051 pass	97 skip	2 fail	115078
ltp-syscalls-tests	juno-r2 - arm64	1150 tests	999 pass	149 skip	2 fail	115017
ltp-syscalls-tests	hi6220-hikey - arm64	1150 tests	996 pass	152 skip	2 fail	115113
ltp-syscalls-tests	x86_64	1150 tests	1030 pass	118 skip	2 fail	115137

156/156 *test suites with no failures hidden* Show

Example : How about a log?



Example : What does the trend look like?

Build	Date	juno-r2 - arm64	hi6220-hikey - arm64	x15 - arm	x86_64
v4.14.18-23-g8d861f5b27b0	Feb. 9, 2018, 2:52 p.m.	fail	fail	fail	fail
v4.14.18-19-g44b8fc264b98	Feb. 8, 2018, 3:58 p.m.	pass	pass	*n/a*	skip
v4.14.17-65-g81d0cc85caab	Feb. 7, 2018, 9:51 p.m.	pass	pass	skip	skip
v4.14.17-65-g161cd48d7ece	Feb. 5, 2018, 6:51 p.m.	pass	pass	skip	skip
v4.14.17	Feb. 4, 2018, 12:32 p.m.	pass	pass	skip	skip
v4.14.16-157-gc9c30584bc1c	Feb. 2, 2018, 3:13 p.m.	pass	pass	skip	skip

Example : So we create a bug ...

Bug 3626 - LKFT: LTP : fanotify06 failed (edit) Save Changes

Status: RESOLVED FIXED (edit)

Alias: None (edit)

Product: Kernel Functional Testing

Component: Linux Test Project (LTP) (show other bugs)

Version: unspecified

Hardware: All OpenEmbedded

Importance: --- major

Target Milestone: ---

Assignee: Dan Rue (edit)

URL:

Reported: 2018-02-14 13:15 UTC by Naresh Kamboju

Modified: 2018-02-22 04:22 UTC (History)

CC List: Add me to CC list
2 users (edit)

Ignore Bug Mail: (never email me about this bug)

See Also: (add)

Linux kernel version:
Next
Mainline
4.15
4.15-rc
4.14

Example : Turns out it was a test case issue...

```
This commmit 28507e514c(safe_mount: Do not try mount() syscall for FUSE fs)
involves FUSE fs check in safe_mount(), so we should give the "fs_type" when
calling that in case the system kill our program.

  cmdline="fanotify06"
  contacts=""
  analysis=exit
  <<<test_output>>>
  tst_test.c:980: INFO: Timeout per run is 0h 10m 00s
  tst_test.c:1025: BROK: Test killed by SIGSEGV!

Signed-off-by: Li Wang <liwang@redhat.com>
---
 testcases/kernel/syscalls/fanotify/fanotify06.c | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)

diff --git a/testcases/kernel/syscalls/fanotify/fanotify06.c b/testcases/kernel/syscalls/fanotify/fanotify06.c
index e63e457..8cbe1ad 100644
--- a/testcases/kernel/syscalls/fanotify/fanotify06.c
+++ b/testcases/kernel/syscalls/fanotify/fanotify06.c
@@ -221,7 +221,7 @@ void test01(void)
 static void setup(void)
 {
        SAFE_MKDIR(MOUNT_NAME, 0755);
-       SAFE_MOUNT(MOUNT_NAME, MOUNT_NAME, NULL, MS_BIND, NULL);
+       SAFE_MOUNT(MOUNT_NAME, MOUNT_NAME, "none", MS_BIND, NULL);
        mount_created = 1;
        SAFE_CHDIR(MOUNT_NAME);
```

Example : Yet, that's not always the case

```
Hi Michal -

We (Linaro) run the libhugetlbfs test suite continuously against
mainline and recently (Feb 1), the 'counters' test started failing on
with the following error:

    root@localhost:~# mount_point="/mnt/hugetlb/"
    root@localhost:~# echo 200 > /proc/sys/vm/nr_hugepages
    root@localhost:~# mkdir -p "${mount_point}"
    root@localhost:~# mount -t hugetlbfs hugetlbfs "${mount_point}"
    root@localhost:~# export LD_LIBRARY_PATH=/root/libhugetlbfs/libhugetlbfs-2.20/obj64
    root@localhost:~# /root/libhugetlbfs/libhugetlbfs-2.20/tests/obj64/counters
    Starting testcase "/root/libhugetlbfs/libhugetlbfs-2.20/tests/obj64/counters", pid 3319
    Base pool size: 0
    Clean...
    FAIL    Line 326: Bad HugePages_Total: expected 0, actual 1

Line 326 refers to the test source @
https://github.com/libhugetlbfs/libhugetlbfs/blob/master/tests/counters.c#L326

I bisected the failure to this commit. The problem is seen on multiple
architectures (tested x86-64 and arm64).

Thanks,
Dan
```

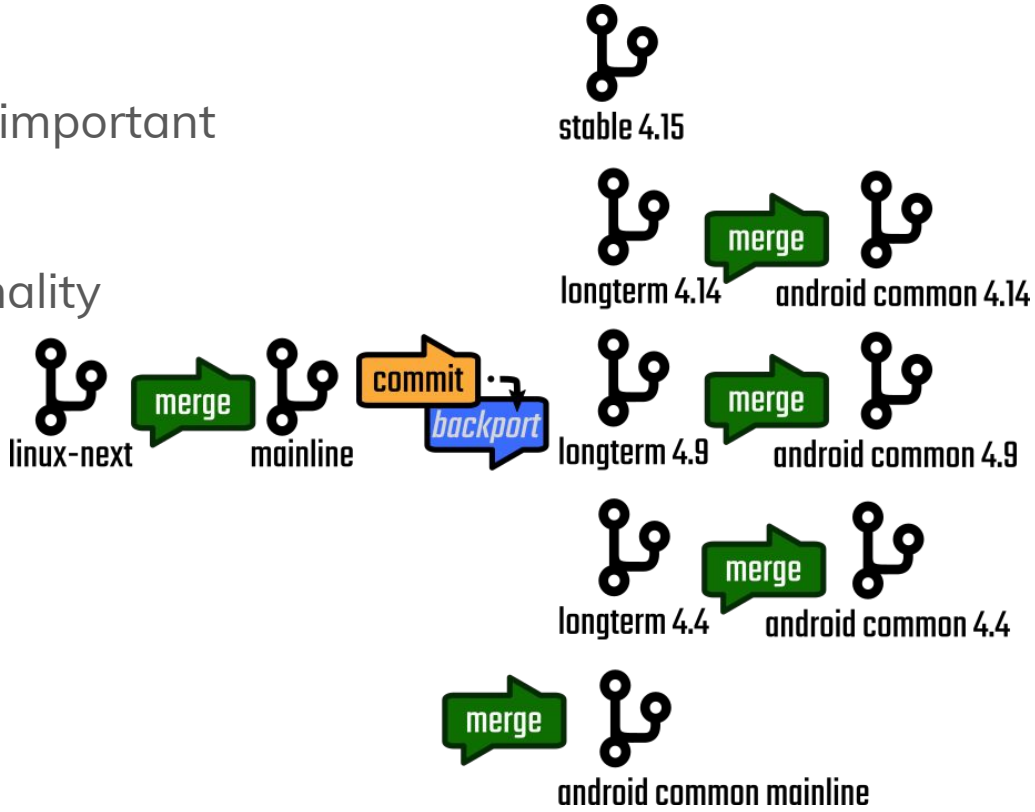
Getting involved

- Linux-stable
  - https://www.kernel.org/doc/html/v4.15/process/stable-kernel-rules.html
  - LTS RCs, testing results, candidate patches
  - Mailing List : stable@vger.kernel.org
- Kselftest
  - https://kselftest.wiki.kernel.org
  - linux-kselftest@vger.kernel.org
- LTP
  - https://linux-test-project.github.io

Making the Universe Better

In Summary

- Finding kernel regressions is important
  - More boards
  - More eyes
- Exercise more kernel functionality
  - More tests!
  - More testing!




Linaro

KernelCI Capabilities Compared to LKFT

Why LKFT and not a functional test framework extension of KernelCI?

At the time LKFT was created KernelCI did not have any aspirations for functional test (or they weren't public).

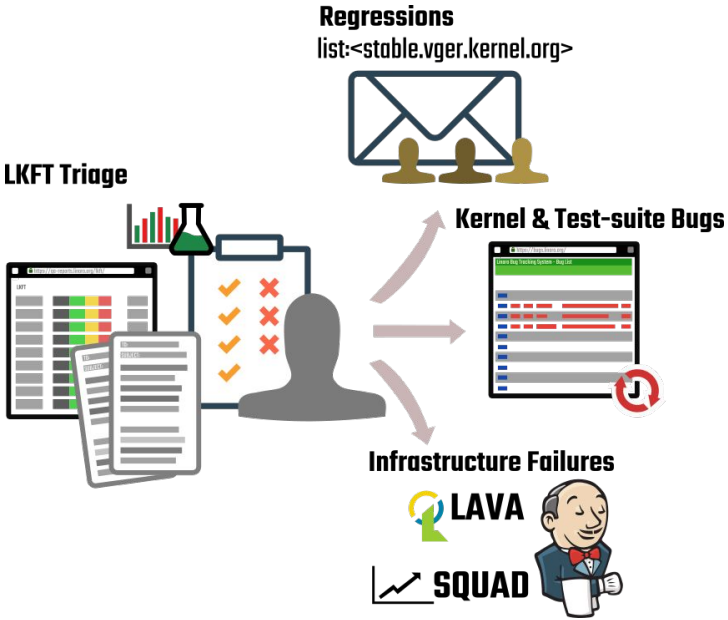
From the beginning LKFT has been focused on functional testing specific kernel trees (that match Linaro's membership motivations).

Even now, as support for kselftest is being added to KernelCI, there is minimal filesystem support, so it does not yet match, 1-for-1, the functional test capabilities of LKFT.

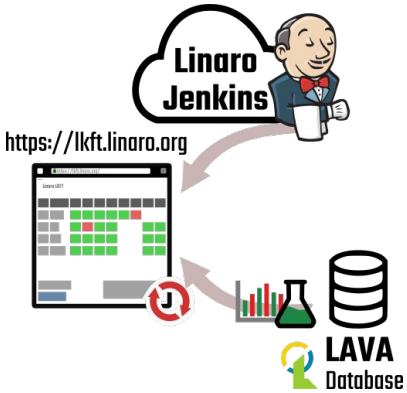
LKFT Mission & Reach

As part of Linaro's mission to improve the Arm architecture ecosystem, the LKFT team reports discovered regressions to Linaro kernel developers, Linaro members, and upstream Linux kernel engineers.

It is important to the Arm ecosystem that Linaro also fix as many failures as are found. The LKFT team invests time into identifying, reporting, and fixing upstream kernel regressions, identifying kernel regressions in select member-hardware SoC (system-on-a-chip) trees, fixing test-suites by contributing to upstream testing projects, fixing kernel configurations, improving full OS stack integration (firmware, kernel, userspace), and improving Arm device automation integration.

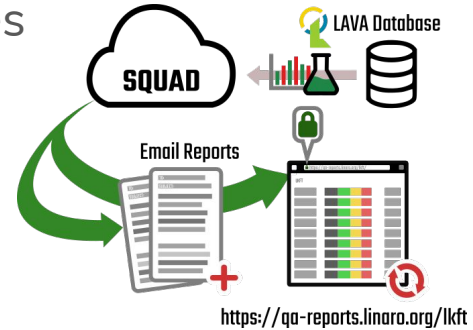


lkft.linaro.org and qa-reports.linaro.org



https://lkft.linaro.org is a website for kernel engineers, business partners, and managers to get up-to-date information on functional test results against the latest commits to a variety of Linux kernel source trees.

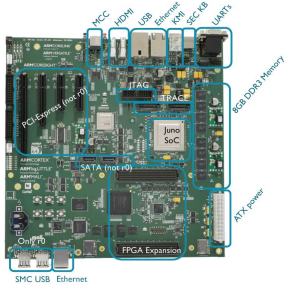
https://qa-reports.linaro.org/lkft is a website that provides full details of the latest and historical functional test results, as well as a variety of comparison and reporting tools. Its purpose is to aide kernel triage engineers in discovering the cause of functional test failures.



Qemu

Generic x86
X86-64 64 bit

TI Beagleboard X15
AM5728 32bit A15



ARM Juno
64 bit Axx/Axx/Mali

Lemaker HiKey
HiSilicon Octa 64 bit A53/Mali