KUnit - One Year Later

Brendan Higgins <brendanhiggins@google.com>

Who am I?

- Brendan Higgins <brendanhiggins@google.com>
- I am currently working on KUnit
- Previously I worked on
  - server bringup at Google
  - OpenBMC

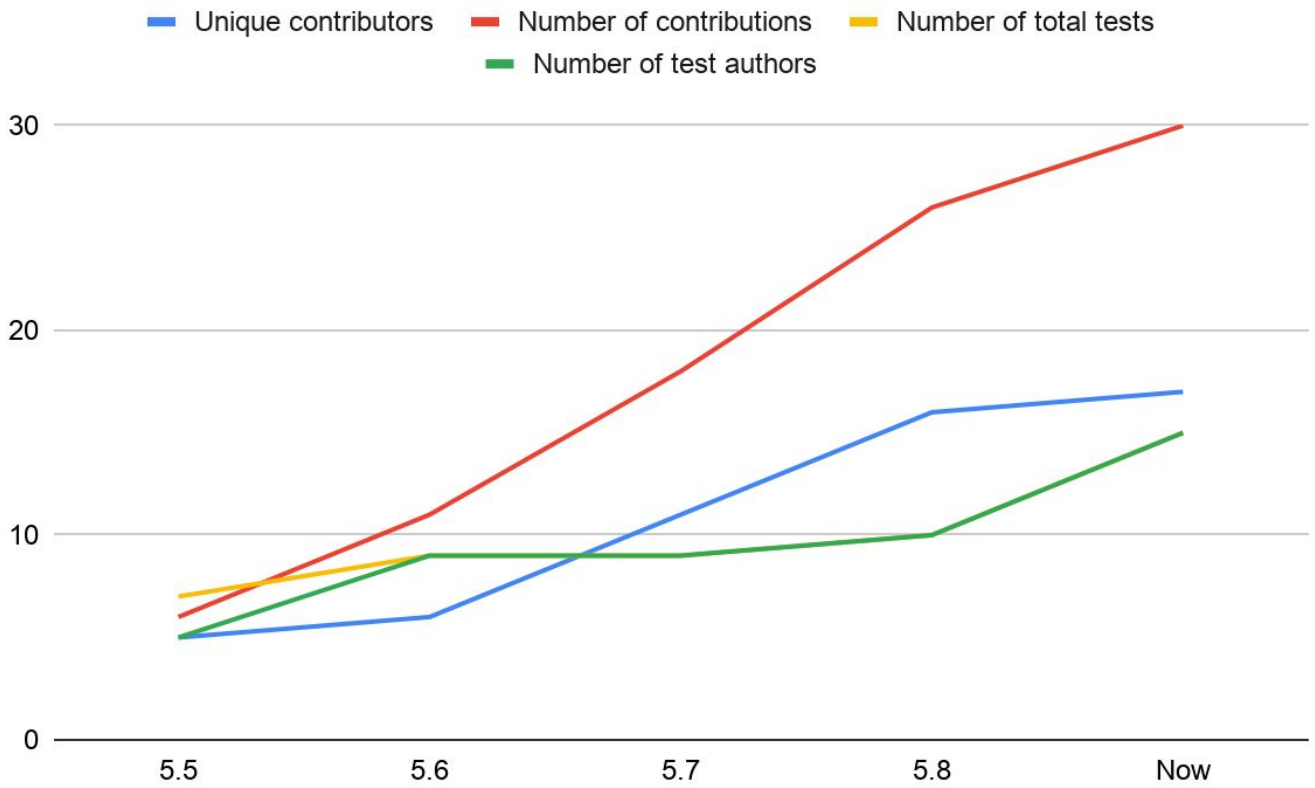
Context

What is KUnit?

- Unit testing for the Linux kernel.
- I have given a couple talks on it in the last year:
  - https://linuxplumbersconf.org/event/4/contributions/545/
  - https://linux.conf.au/schedule/presentation/97/
- Merged in torvalds/master in v5.5

Updates

Stats over the last year



Indicators for the next year

- Between 5 and 10 new tests currently under review
- A large number of conversions under way
- Still getting a lot of contributions

Takeaway

- Some people seem to find KUnit useful
- Adoption isn’t as fast as I would like
- Contributions vastly exceeded my expectations
- We still have a lot of room to grow
- I’m happy :-)

New Features

- Module support
- DebugFS support
- Module only/userspace tests*
- Multithread/multitask support
  - Access test data outside of test thread*
  - Named resources

KUnit: Linux Kernel Integration Testing?!

- None of these features were from Google
- I was originally against integration testing features
- Last year I said, “Integration testing - ???”
  - It seems the hive mind has spoken

KTAP: Unified Linux Kernel Test Output

- Converging on single test output implementation
- KUnit eating non-standard tests

Update on Old New Features: KernelCI

- Good progress in Q4-Q2
- Heidi moved on to a new project :-(
- I only picked up the work again in the last month
- Seems pretty close

Update on Old New Features: Mocking

- Not as much progress as we would have liked
- Less upstream interest than expected
  - Some, but less than expected
- My bosses are doubling down on this
  - Seems uncontroversial
  - Useful for Google stuff

Lessons Learned

kunit_tool: Love or Hate?

- We got a lot of feedback that kunit_tool was pointless
- Then we changed it, lots of complaints
- Clearly, some people like it, some don’t
- People usually don’t say anything when they are happy
  - “The squeaky wheel gets the grease.”

People care about names

- People care a lot more about naming consistency than I expected
- Everyone has an opinion
- Lately this has been a major delay

Don’t let your vision get in the way of the goal

- I started off KUnit with a lot of strong ideas/beliefs
  - How it should be
  - How it should be used
- The hivemind has different ideas
- The hivemind is like a river
  - Provide structure
  - Facilitate
  - Don’t get in the way

The Future

Plans

- KernelCI: We’re pretty close
- Mocking: The dragons are coming, I swear
- Parameterized testing: useful for data driven tests
- More test conversions
- Even more better documentation

Plans: Stuff you haven’t heard before (maybe)

- kunit_tool support for QEMU
  - I have an RFC out
  - Creating compatible toolchain, Kconfig, QEMU configs, etc is hard
  - Having a script do it for “all” architectures is useful?
- Device fakes for driver fuzzing

Predictions

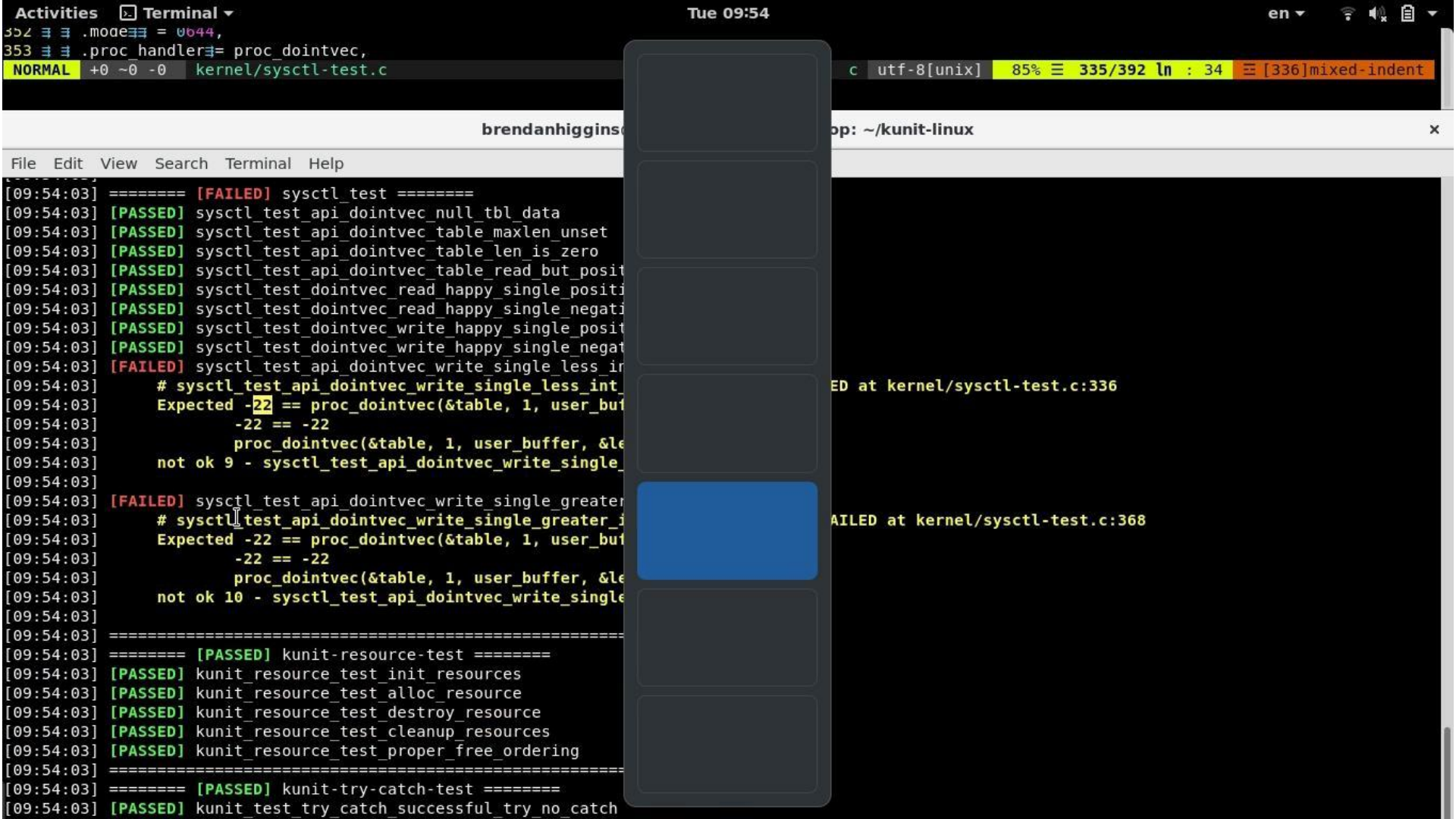
- KUnit will continue to develop integration test features
- All in kernel tests will report in KTAP
- KUnit coverage will continue to grow, faster, but still slowly

Talk to me!

- kunit-dev@googlegroups.com
- linux-kselftest@vger.kernel.org
- #kunit on oftc.net

Thanks!

Backup slides


Activities
Terminal
Tue 09:54
en
.mode = 0644,
.proc_handler= proc_dointvec,
NORMAL +0 ~0 -0 kernel/sysctl-test.c
c utf-8[unix] 85% 335/392 ln : 34 [336]mixed-indent
brendanhiggins
op: ~/kunit-linux
File Edit View Search Terminal Help
[09:54:03] ======== [FAILED] sysctl_test ========
[09:54:03] [PASSED] sysctl_test_api_dointvec_null_tbl_data
[09:54:03] [PASSED] sysctl_test_api_dointvec_table_maxlen_unset
[09:54:03] [PASSED] sysctl_test_api_dointvec_table_len_is_zero
[09:54:03] [PASSED] sysctl_test_api_dointvec_table_read_but_posit
[09:54:03] [PASSED] sysctl_test_dointvec_read_happy_single_positi
[09:54:03] [PASSED] sysctl_test_dointvec_read_happy_single_negati
[09:54:03] [PASSED] sysctl_test_dointvec_write_happy_single_posit
[09:54:03] [PASSED] sysctl_test_dointvec_write_happy_single_negat
[09:54:03] [FAILED] sysctl_test_api_dointvec_write_single_less_in
[09:54:03] # sysctl_test_api_dointvec_write_single_less_int_
ED at kernel/sysctl-test.c:336
[09:54:03] Expected -22 == proc_dointvec(&table, 1, user_buf
[09:54:03] -22 == -22
[09:54:03] proc_dointvec(&table, 1, user_buffer, &le
[09:54:03] not ok 9 - sysctl_test_api_dointvec_write_single_
[09:54:03]
[09:54:03] [FAILED] sysctl_test_api_dointvec_write_single_greater
[09:54:03] # sysctl_test_api_dointvec_write_single_greater_i
AILED at kernel/sysctl-test.c:368
[09:54:03] Expected -22 == proc_dointvec(&table, 1, user_buf
[09:54:03] -22 == -22
[09:54:03] proc_dointvec(&table, 1, user_buffer, &le
[09:54:03] not ok 10 - sysctl_test_api_dointvec_write_single
[09:54:03]
[09:54:03] ==========
[09:54:03] ======== [PASSED] kunit-resource-test ========
[09:54:03] [PASSED] kunit_resource_test_init_resources
[09:54:03] [PASSED] kunit_resource_test_alloc_resource
[09:54:03] [PASSED] kunit_resource_test_destroy_resource
[09:54:03] [PASSED] kunit_resource_test_cleanup_resources
[09:54:03] [PASSED] kunit_resource_test_proper_free_ordering
[09:54:03] ==========
[09:54:03] ======== [PASSED] kunit-try-catch-test ========
[09:54:03] [PASSED] kunit_test_try_catch_successful_try_no_catch

KUnit Example

```
static void list_del_init_test(struct test *test)
{
        struct list_head a, b;
        LIST_HEAD(list);

        list_add_tail(&a, &list);
        list_add_tail(&b, &list);

        /* before: [list] -> a -> b */
        list_del_init(&a);

        /* after: [list] -> b, a initialised */
        KUNIT_EXPECT_EQ(test, list.next, &b);
        KUNIT_EXPECT_EQ(test, b.prev, &list);
        KUNIT_EXPECT_TRUE(test, list_empty_careful(&a));
}
```

More on x-unit

- https://google.github.io/kunit-docs/third_party/kernel/docs/usage.html
- https://martinfowler.com/bliki/Xunit.html

Where does KUnit fit into the kernel's test paradigm?

- Lot's of unit tests (~80%)
  - This is where KUnit lives
- A moderate number of integration tests (~15%)
  - ???
- And some end-to-end tests (~5%)
  - We got this covered (kselftest, xfstest, etc)