The bugs are too fast
and why we can’t catch them.

Kevin Hilman, BayLibre

ATS 2019: Lyon, France
Introductions

BayLibre

- embedded Linux consultancy, engineering services
- based in Nice, France
- ~40 engineers
- open-source focus
  - top 20 Linux kernel contributor
  - top 5 AGL contributor
  - u-boot, Zephyr, ATF, OP-TEE, Yocto

Kevin

- co-founder, Sr. Engineer
- Linux kernel developer and maintainer
- based in Seattle
- co-founder KernelCI project
Agenda

● Kernel testing landscape
● Bugs
● Fragmentation
● KernelCI & Consolidation
Kernel testing landscape

- LTP, kselftest, syzbot, ...
- KUnit: unit testing and mocking [1]
  → arch agnostic, can use UML: fast!
  → just merged
- KTF: Kernel Test Framework [2]
  → RFC Aug 12, 2019

[1] https://google.github.io/kunit-docs/third_party/kernel/docs/
[2] https://lore.kernel.org/linux-kselftest/CAFd5g44-RMaH0kwbd+wE41HO_CgBZ3wK0vnryYvb_rE68JazWg@mail.gmail.com/
Kernel testing landscape

- Intel 0-Day and Linux Kernel Performance (LKP)[1]
  → Builds and static analysis for many arches, testing only on x86

  → In-depth testing; Only run tests on Linaro member platforms

- CKI: Continuous Kernel Integration[3]
  → Stable kernel focus: x86_64, arm64, ppc64le

- KernelCI
  → Broad hardware support; very basic test suites
Kernel testing landscape

- Developers, contributors to upstream, maintainers
  → Only run tests on their workstations / dev boards

- Users: distros, OEMs, SoC/CPU vendors
  → Only run tests on their own hardware
  → Don’t necessarily send fixes upstream
Total test coverage = On the beaten tracks
Bugs
Fixes: tags

- 2017: 7603/73873 (10.3%)
- 2018: 8947/75768 (11.8%)
- 2019: 8259/59959 (13.8%)
- <½ has Fixes tags (40% in linux-4.14.y)

Source: Dmitry Vyukov’s LPC2019 talk: 
https://linuxplumbersconf.org/event/4/contributions/554/
syzbot bugs

2 years:

- ~2300 bugs upstream (3/day)
- ~2500 bugs in Android/ChromeOS/stable/internal
  +1000 reported manually before syzbot (~40 bugs/mo for 2 years)

= 5800 bugs

- fuzzing is not supposed to find that many! (simple bugs, broken subsystems)
- only 7% coverage
- only "crashes" (fine with "does wrong thing", bad EINVAL)
- no KTSAN, no KUBSAN

Source: Dmitry Vyukov’s LPC2019 talk
"Stable" releases

+ not backported fixes (700+)
+ not fixed upstream bugs (500+)
+ not found/detectable bugs (???)

>20'000 bugs/release

Source: Dmitry Vyukov’s LPC2019 talk
Buried in bugs. Can we dig out?

Yes, BUT....
Fragmentation

- CI / CD pipelines
- test frameworks
- test suites
- results parsing
- pass / fail criteria
- log collection, aggregation
- results reporting, analysis
- results visualization
- bug tracking
- kernel developer processes for fixes

... and this is just in the open, community projects.
Conclusion

Fragmentation bad
Collaboration good
Work upstream
No upstream? create one!

… also for testing & CI
KernelCI status update
KernelCI: off-road testing

Goal: all CPU architectures

Today:
- \texttt{x86\_64, arm, arm64, mips, arc, riscv}

Goal: a wide range of hardware platforms

Today
- 35+ SoC vendors
- 250+ unique boards
KernelCI: multiple build dimensions

Multiple kernel trees
- mainline, next, stable, stable-rc
- subsystems: media, sound, clk, soc
- maintainers, developers
- android-common, chrome-platform

Multiple compilers
- gcc, clang
- multiple versions

Multiple config options
- all upstream defconfigs (220+)
- CONFIG_CPU_BIG_ENDIAN=y'
- CONFIG_SMP=n
- CONFIG_RANDOMIZE_BASE=y'
- and more...
Functional tests

Graphics: IGT (DRM/KMS)
   → Subset run on a handful of devices, gradually expanding

Media: v4l2-compliance
   → Full test suite run on hardware and QEMU (vivid driver)

Power: suspend / resume
   → Run on many boards, finding issues regularly

USB: smoke test
   → Check that the USB subsystem is initialised
Fragmentation

- CI / CD pipelines
- test frameworks
- test suites
- results parsing
- pass / fail criteria
- log collection, aggregation
- results reporting, analysis
- results visualization
- bug tracking
- kernel developer processes for fixes

… and this is just in the open, community projects.
Consolidation, Collaboration, Community

- Membership based, Sustainable funding
- Open testing philosophy
- KernelCI as open-source software
- KernelCI as a service: kernelci.org
- Founding members:
  - Collabora, BayLibre, Google, Microsoft, RedHat, CIP, Foundries.io
Challenge: data is growing

Matrix is expanding

Collecting lots of data, results, logs, artifacts

Storage, Analytics, Visualization, Reporting

Big Data?
What’s next?

Collaboration: LKFT, CKI, Fuego...

Improve reporting, analytics, visualization, reporting, etc.

More hardware, more compute

Other CI pipelines (gitlab CI,...)

More tests: fuzzing, KUnit?

Distro kernels, Yocto?

Join the project and help decide!
Open testing philosophy
We like open-source software
What about open-source testing?
github.com/kernelci/kcidb
Showing failures prompts developers to fix them.
Photo credits

agenda: https://pixabay.com/photos/agenda-appointment-calendar-coffee-2296195/
landscape: https://www.flickr.com/photos/hemlit/8212362709/
sand: https://www.flickr.com/photos/156754622@N02/23962149187/
thank you: https://flic.kr/p/bGhz
big data: https://flic.kr/p/deKzer
future: https://pxhere.com/en/photo/1449979
obvious: https://www.flickr.com/photos/emmajane/2580835224
ants: http://pngimg.com/download/19357
snow: https://www.flickr.com/photos/waynethume/4335043665
big data: https://flic.kr/p/deKzer
future: https://pxhere.com/en/photo/1449979