Kselftest running in test rings - Where are we?

Shuah Khan
Linux Kernel Fellow
The Linux Foundation
Kselftest is a developer test suite for

Kernel Developers

Kernel Users

@ShuahKhan
skhan@linuxfoundation.org
Kselftest is a collection of tests

- Code-Based (Open box)
- Behavioral (Closed box)
- Functional
- Feature
- Hardware and drivers
- Stress and performance

- https://git.kernel.org/pub/scm/linux/kernel/git/shuah/linux-kselftest.git/
- https://patchwork.kernel.org/project/linux-kselftest/list/
Kselftest is not for testing

- Workloads
- Applications

@ShuahKhan
skhan@linuxfoundation.org
Testing scope and focus

- Kernel Features & API
- Kernel Functionality
- Regressions in features and API
- Subsystem specific

Several new tests and test cases are added every single release.
Who are the authors?

Kernel Developers

Kernel Users

@ShuahKhan
skhan@linuxfoundation.org
Who are the users?

Kernel Developers

Kernel Users

@ShuahKhan
skhan@linuxfoundation.org
How do patches flow?

Other subsystem test patches follow similar flow in the interest of keeping features and tests grouped together in pull requests.

@ShuahKhan
skhan@linuxfoundation.org
Individual test view

Main test (Target)

sub-test

Test

Individual Test Cases
Kselftest goals & challenges

- Evolving common framework flexible for customizing tests
- Increase coverage (drivers, configs, and features)
- Add regression tests for fixed bugs
- Common interfaces for Pass/Fail/Skip reporting
- Reporting results in simple text based Test Anything Protocol 13
- Balance kselftest run-time and coverage
Kselftest goals & challenges

- Balance kernel developer and user use-cases
- Evolving common framework to support test ring use-cases
  - Framework is well suited for manual testing.
  - Needs changes to support auto-test environments.
Kselftest use-cases

- Native and cross-build use-cases
  - Individual tests
  - Subset of tests
  - All tests
- Relocating native and cross-build objects
Kselftest use-cases

- Running tests use-cases
  - Individual tests
  - Subset of tests
  - All tests
Kselftest use-cases

- Generating tests for install with run script use-cases
  - Individual tests
  - Subset of tests
  - All tests
- Support relocating install objects (native & cross-builds)
Kselftest and test rings

- Linux Kernel Functional Testing
- Runs Kselftests on:
  - linux-next
  - linux-mainline
  - Stable
  - Active kernel Releases

Kselftests from the same repo are used to rev match kernel. Once exception is kselftests from latest stable are run all stables.
Kselftest and test rings

- 0-Day Service
  - Runs Kselftests from mainline on several trees and kernel configs
Support relocating install objects (native & cross-builds)
  - Supported since Linux 5.6 (except bpf)
  - Relative path support is work in progress

Dependency checks for build/cross-build - kselftest_deps.sh
  - Supported since Linux 5.6.
  - Prints test targets that can be built. This output can be used in auto-test frameworks.

Build/cross-build tests for specific subsystems (supported with TARGETS var)
Kernel CI use-cases

• Build/cross-build tests for specific configs
  – Individual tests add config file with required dependencies
  tools/testing/selftests/*/config
  – “make kselftest-merge” generates kernel config to include
    individual test config files

• Build/cross-build tests for specific features (this is a bit tricky)
  – One single test could cover multiple features for a config or a
    subsystem.
Kernel CI use-cases

- Default builds/runs/install all TARGETS.
  - make kselftest-all
  - make kselftest
  - make kselftest-install

- Using TARGETS helps select a subset of tests to build.
  - make kselftest-install TARGETS="breakpoints timers"

- Install generates a script to run tests and report results.
Kselftest Kernel CI workflow

• Cross-compile kernel (relocatable):
  - make O=/arm64_build ARCH=arm64 HOSTCC=gcc CROSS_COMPILE=aarch64-linux-gnu-
    defconfig
  - make O=/arm64_build ARCH=arm64 HOSTCC=gcc CROSS_COMPILE=aarch64-linux-gnu-
    all

• Cross-compile kselftest-all (relocatable):
  - make kselftest-all ARCH=arm64 HOSTCC=gcc CROSS_COMPILE=arm-linux-gnueabi-
    O=/tmp/kselftest_arm > kselftest_all_arm.log 2>&1
  - make -C tools/testing/selftests ARCH=arm64 HOSTCC=gcc CROSS_COMPILE=aarch64-
    linux-gnu- CC="ccache aarch64-linux-gnu-gcc" O=build-arm64

• Cross-compile kselftest-install (relocatable):
  - make kselftest-install O=/arm64_build ARCH=arm64 HOSTCC=gcc
    CROSS_COMPILE=aarch64-linux-gnu- > kselftest_install 2>&1
Next steps

- Status - test breakages for cross-builds installs since 5.6?
- Add support for build/cross-build tests for specific configs
- Add support for build/cross-build tests for specific features (this is a bit tricky)
- One single test could cover multiple features for a config or a subsystem.
- Problems/issues/suggestions?