The seL4 GitHub Test Suite

Gerwin Klein | seL4 summit 2022

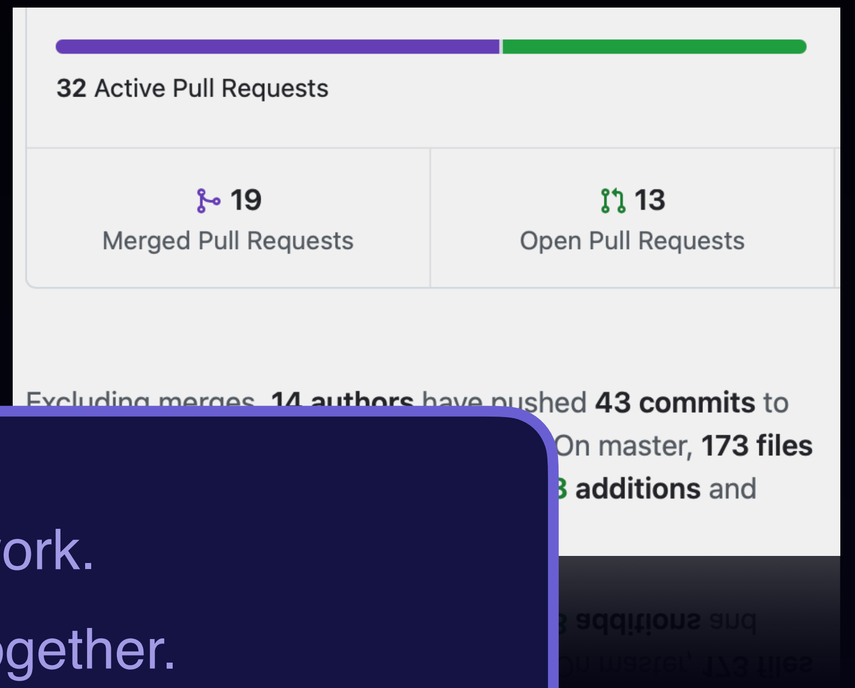
Why test?

Don’t we have formal verification?

More than just the kernel

- seL4 Foundation manages over 60 repositories
  - kernel and proofs just 2 of these

- Others include
  - sel4test
  - sel4bench
  - libraries
  - capDL/system initialiser
  - CAmkES framework, tools, components
  - virtual machine monitors
  - example system
  - manifest reposit
  - binary verificati
  - build tools, proc
  - websites, test in



All of these need to work.

And they need to work together.

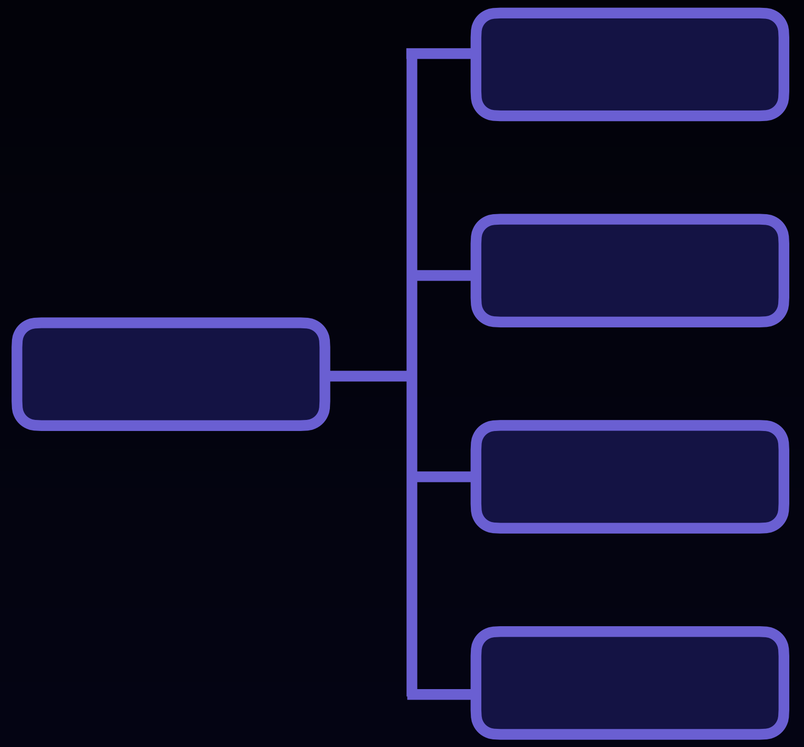
Main Questions

- Which repos do you need for a specific task?
- Which version of which repo works with what?
- How do we ensure this collection keeps working?

Which version of what?

Manifests

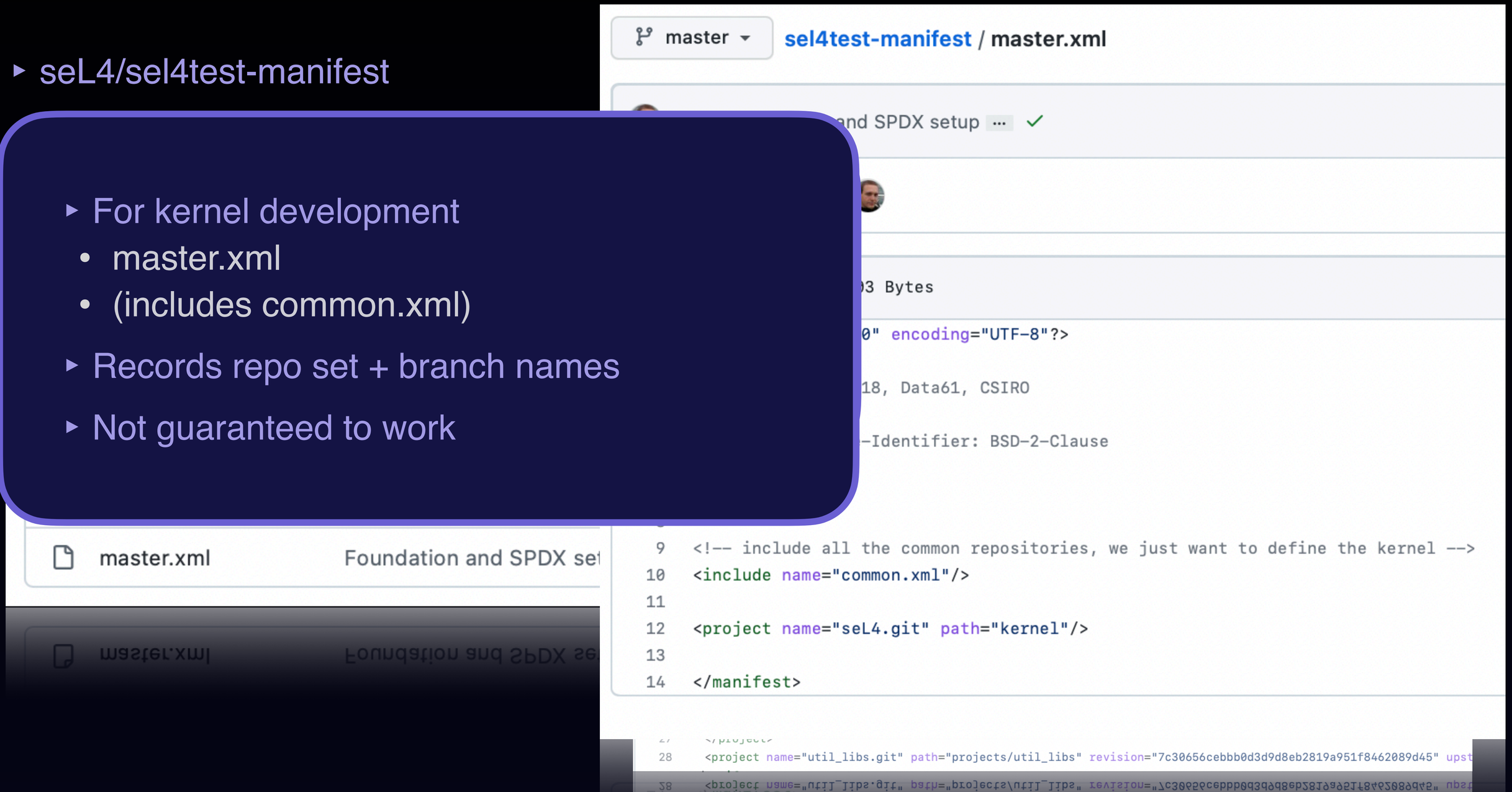
- Kernel is almost never used in isolation
- Proofs, tests, applications need sets of repos
- Requirements
  - record repo combinations
  - record which version combinations are known-good
  - ability to manipulate + version control config state



seL4test Manifest Example

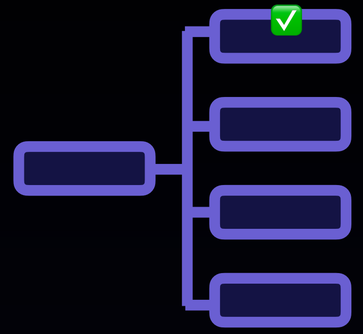
- seL4/sel4test-manifest

- For kernel development
  - master.xml
  - (includes common.xml)
- Records repo set + branch names
- Not guaranteed to work

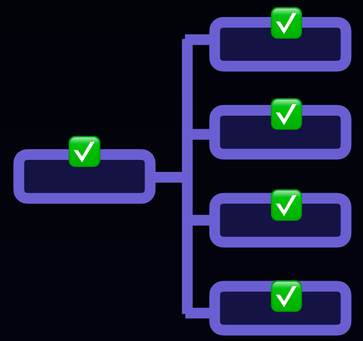


Local and Global Consistency

- Local Consistency:
  - code works for a specific repository in isolation
  - e.g.: code + git style, link check, license check, code compiles, local tests work



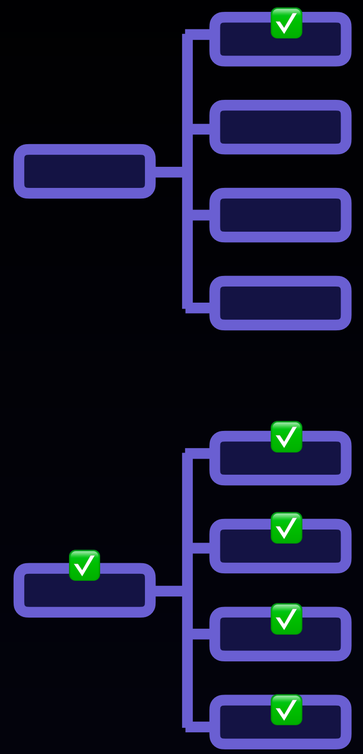
- Global Consistency:
  - all tests work for all repos in a manifest
  - e.g.: proofs, sel4test, benchmarks, user-level apps, VMs



- All changes must maintain local consistency
- May temporarily break global consistency:
  - might need multiple changes in multiple repos
  - too expensive to test for all local changes
  - if broken, must find out and fix ASAP

Implementation

- Local tests run immediately on pull request
- For each manifest one global test:
  - expensive, not triggered on pull request (some exceptions)
  - does not “live” in the manifest repo, but main content repo
    - e.g. in sel4bench repo for sel4bench-manifest
  - triggered when any repo in the manifest changes
  - updates manifest when test succeeded



- Implemented as GitHub Actions
  - code for all actions centralised in repo seL4/ci-actions
  - called in local .github/workflow directory in each repo
  - compute-expensive tests run on AWS
  - hardware board tests currently run at UNSW

What tests do we have?

Which tests exist?

- https://docs.sel4.systems/processes/test-status.html

seL4 Test Status

The following list shows the current status of all seL4 GitHub test workflows on the main branch.

Main Tests

seL4Test passing | Proofs passing | Proof Sync passing | seL4Bench passing | CAmkES passing | Camkes VM passing

Main repositories

seL4

CI passing | Compile passing | RefMan passing | XML passing | C Parser passing | Proof Sync passing | seL4Test passing | Trigger passing

l4v

Proofs passing | CI passing | Trigger passing | External passing | Weekly Clean passing | Proofs passing | Prepare binary verification failing

sel4bench

CI passing | seL4Bench passing

camkes-tool

Main Tests

- seL4Test:
  - basic test suite for seL4 on simulator + hardware, also exercises basic user-level libraries

- Proofs
  - the full formal proof suite for seL4

- ProofSync
  - green if the proof applies to the current head version of seL4

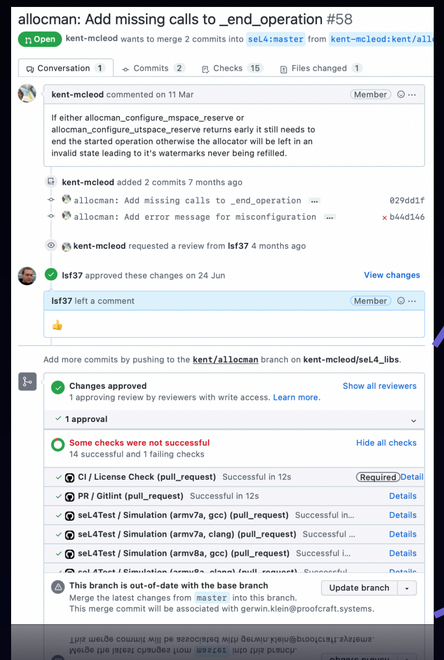
- seL4Bench
  - seL4 benchmarks, also exercises more user-level libraries

- CAmkES
  - component framework, system initialiser, capDL, user-level components + applications

- CAmkES VM
  - virtual machine monitors + virtual machines on ARM and x86

Mandatory Local Tests

- Minimal set of checks for any Foundat
  - code style
  - git lint
  - link check
  - SPDX license headers + copyright
  - DCO (developer certificate of origin)
  - local compile/build checks



Code Issues 6 Pull requests 6 Discussions Actions Security Insights

allocman: Add missing calls to _end_operation #58

Edit Code

Open kent-mcleod wants to merge 2 commits into seL4:master from kent-mcleod:kent/allocman

Conversation 1 Commits Checks 15 Files changed 1 +22 −14

allocman: Add error message for misconfiguration b44d146

DCO

seL4Test
on: pull_request

PR
on: pull_request

Gitlint
Trailing Whitespace
Portable Shell

CI
on: pull_request
1

License Check
Links
Style

DCO / DCO
succeeded on 11 Mar in 0s

DCO

All commits are signed off!

View more details on DCO

Two example repos

Kernel: seL4 repo

- Kernel repo has the strictest set of checks

- Standard tests + local:
  - Compile: check that kernel compiles for common configs (fast check)
  - RefMan: check that reference manual builds
  - XML: check that API xml files conform to their DTD
  - C-Parser: check if C code is in C verification subset

- Global consistency:
  - seL4test:
    - run full seL4test suite on simulators + hardware
    - deploy version update to sel4test-manifest if successful
  - ProofSync: fail if manual proof update is necessary
  - Trigger: if all other tests succeeded, trigger test run for all manifests that contain seL4
    - kicks off sel4bench, camkes, camkes-vm, tutorials, etc

Kernel: seL4 repo

- For pull reques
  - seL4test-sim:
  - Preprocess: c
    - if this succe
    - if this fails,
- For expensive
  - trigger manua
  - reviewer-role
  - add ***Test with***
  - seL4test-hw-b
  - seL4test-hw:
  - Proofs: run pr

Jobs

- HW Build (armv7a, gcc)
- HW Build (armv7a, clang)
- HW Build (armv8a, gcc)
- HW Build (armv8a, clang)
- HW Build (nehalem, gcc)
- HW Build (nehalem, clang)
- HW Build (rv64imac, gcc)
- Matrix
- HW Run (TX2, armv8a, gcc)
- **HW Run (TX2, armv8a, cla...**
- HW Run (ZYNQMP, armv8a, ...
- HW Run (ZYNQMP, armv8a, ...
- HW Run (TX1, armv8a, gcc)
- HW Run (TX1, armv8a, clang)

HW Run (TX2, armv8a, clang)

succeeded 13 days ago in 54m 50s

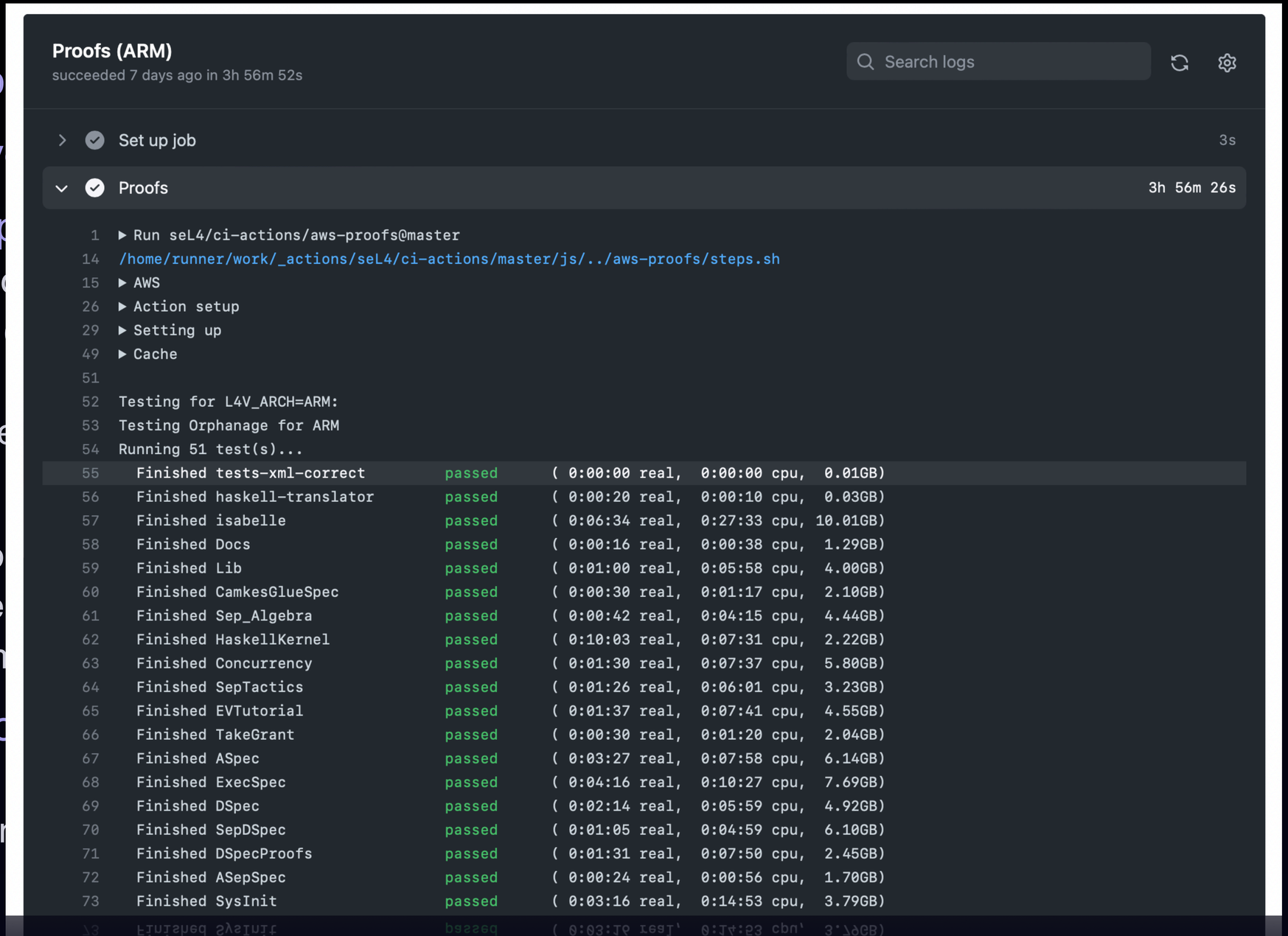
Search logs

- Get machine queue 2s
- Download image 22s
- Run 54m 13s

```
1    ▶ Run seL4/ci-actions/sel4test-hw-run@master
9    /home/runner/work/_actions/seL4/ci-actions/master/js/../sel4test-hw-run/steps.sh
10   ▶ Setting up
92
93   ▶ TX2_debug_clang_64
2297 TX2_debug_clang_64 succeeded
2298
2299 ▶ TX2_debug_MCS_clang_64
4711 TX2_debug_MCS_clang_64 succeeded
4712
4713 ▶ TX2_debug_hyp_clang_64
6919 TX2_debug_hyp_clang_64 succeeded
6920
6921 ▶ TX2_debug_SMP_clang_64
9148 TX2_debug_SMP_clang_64 succeeded
9149
9150 ▶ TX2_debug_SMP_MCS_clang_64
11593 TX2_debug_SMP_MCS_clang_64 succeeded
11594
11595 ▶ TX2_debug_SMP_hyp_clang_64
```

Proofs: l4v repo

- Runs standard tests + p
- Most compute-expensiv
- One AWS instance per p
  - currently 5 in parallel, ea
  - 3-4h each for a full build
  - caches previous proofs
  - detailed logs as build arte
- Triggers on
  - pull-request to l4v repo o
  - merge to main deployme
  - changes to verification-m
- On success on merge to
  - updates manifest
  - triggers binary verificatio



Logs and artefacts

What can you do with these tests?

▶ Run on your own repo fork

cmake: add a sanity check

RefMan #3150

Re-run all jobs

Summary

Jobs

Build PDF

Triggered via pull request 9 days ago

axel-h opened #923 `Hensoldt-Cyber:patch-axe…`

Status

Success

Total duration

2m 5s

Artifacts

1

manual.yml

on: pull_request

Build PDF 1m 42s

Artifacts

Produced during runtime

Name	Size
PDF	964 KB

cmake: add a sanity check

PR #3653

Re-run all jobs

Summary

int

iling Whitespace

table Shell

le

process (ARM)

process (ARM, MCS)

process (ARM_HYP)

process (AARCH64)

process (RISCV64)

process (RISCV64,...

process (X64)

Style

succeeded 9 days ago in 29s

Search logs

Set up job 1s

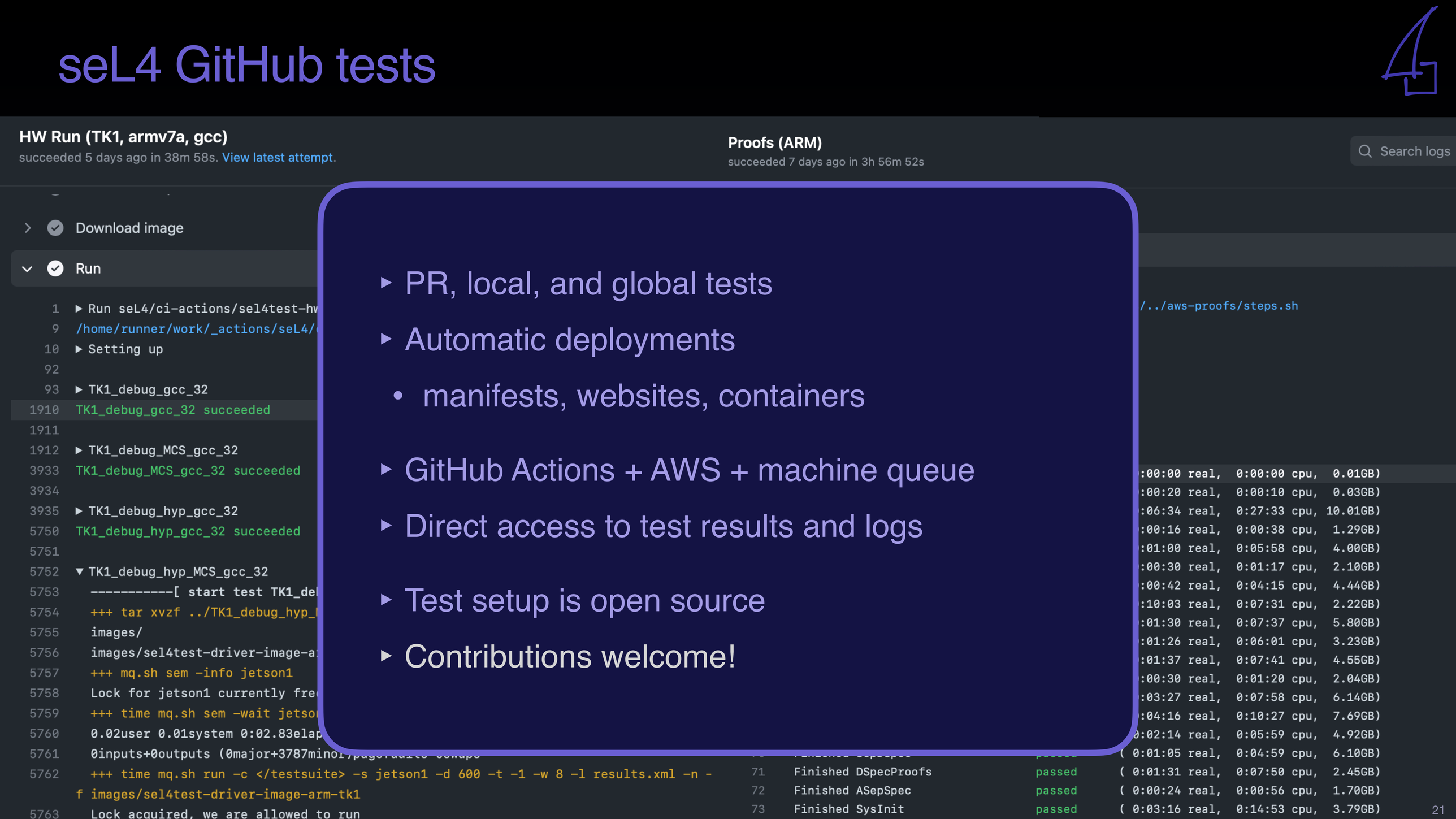
Run seL4/ci-actions/style@master 26s

```
1  ▶ Run seL4/ci-actions/style@master
4  /home/runner/work/_actions/seL4/ci-
   actions/master/js/../style/steps.sh
5  ▶ Setting up
15
16 Checking the following files:
17 configs/seL4Config.cmake
18
19 Check successful!
```

Complete job 0s

Summary

seL4 GitHub tests



- PR, local, and global tests
- Automatic deployments
  - manifests, websites, containers
- GitHub Actions + AWS + machine queue
- Direct access to test results and logs
- Test setup is open source
- Contributions welcome!

Thank You