



# The seL4 GitHub Test Suite



# Why test?

Don't we have formal verification?

# More than just the kernel

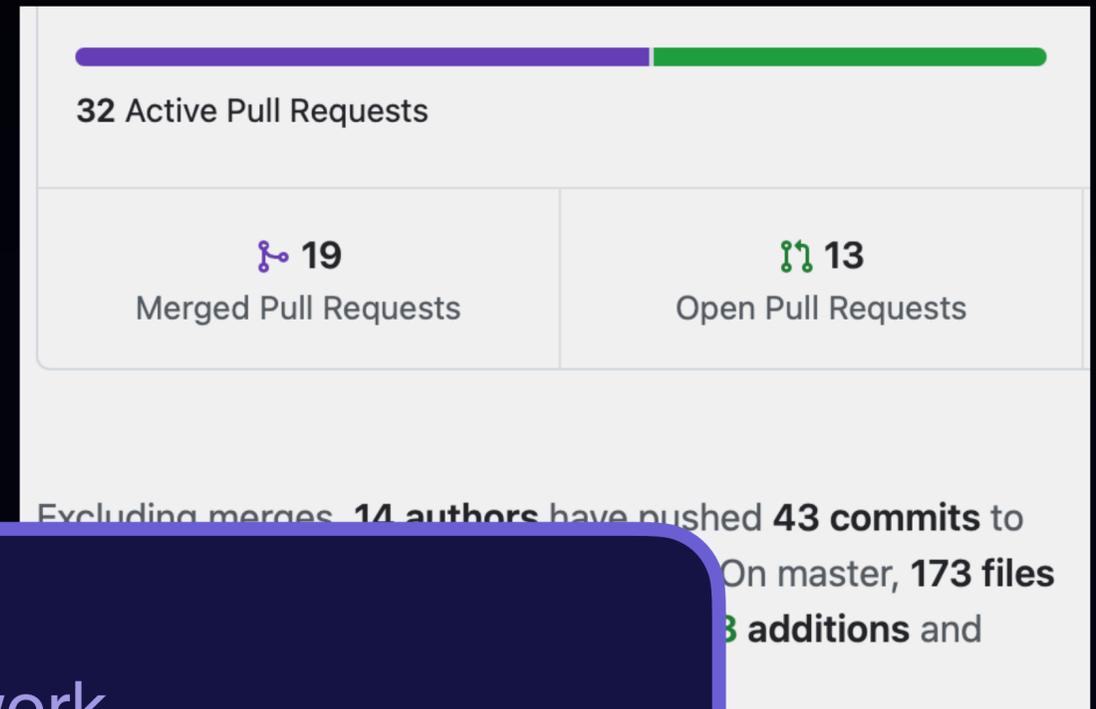


- ▶ seL4 Foundation manages over 60 repositories
  - kernel and proofs just 2 of these



- ▶ Others include

- sel4test
- sel4bench
- libraries
- capDL/system initialiser
- CAmkES framework, tools, components
- virtual machine monitors
- example systems
- manifest repositories
- binary verification
- build tools, procedures
- websites, test infrastructure



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?

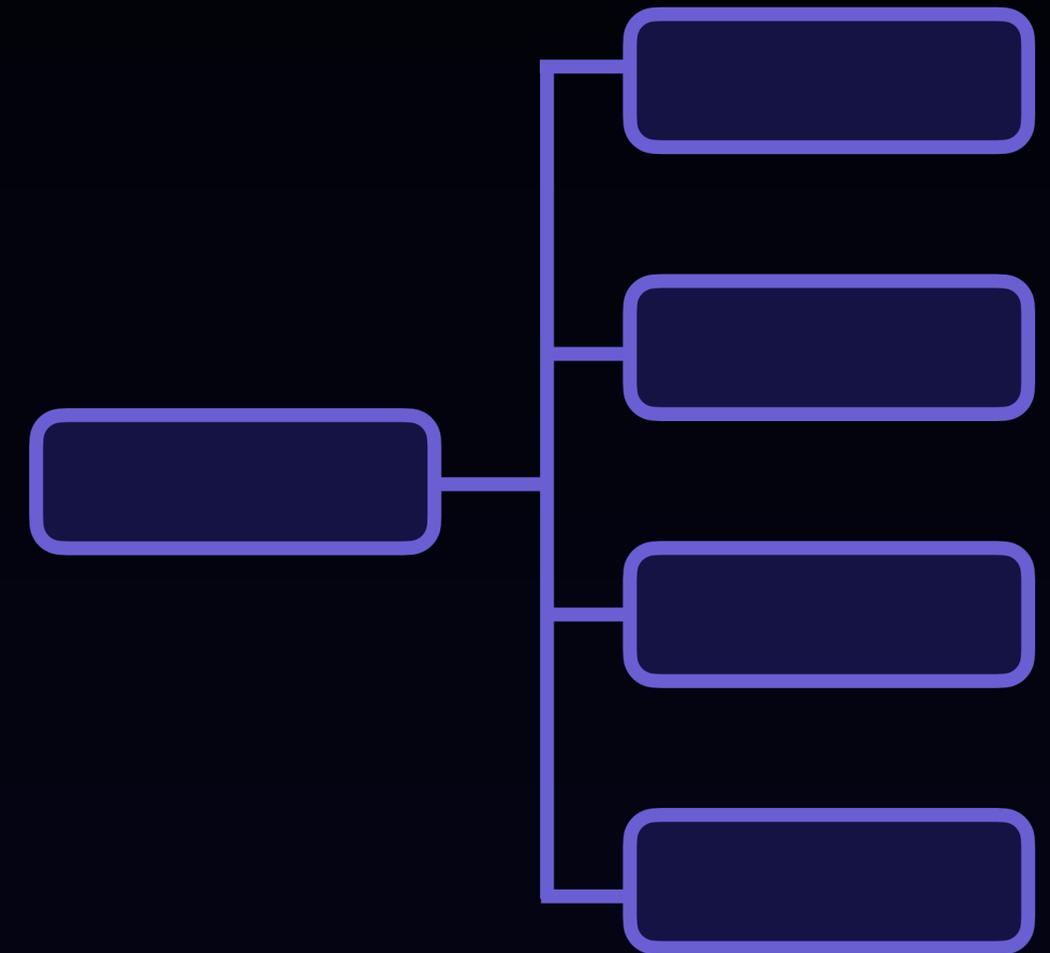


Which version of what?

# 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

```
master sel4test-manifest / master.xml  
and SPDX setup ... ✓  
03 Bytes  
" encoding="UTF-8"?>  
18, Data61, CSIRO  
-Identifier: BSD-2-Clause  
9 <!-- include all the common repositories, we just want to define the kernel -->  
10 <include name="common.xml"/>  
11  
12 <project name="seL4.git" path="kernel"/>  
13  
14 </manifest>  
27 </project>  
28 <project name="util_libs.git" path="projects/util_libs" revision="7c30656cebbb0d3d9d8eb2819a951f8462089d45" upst  
38 <project name="util_libs.git" path="projects/util_libs" revision="7c30656cebbb0d3d9d8eb2819a951f8462089d45" upst
```

master.xml

Foundation and SPDX set

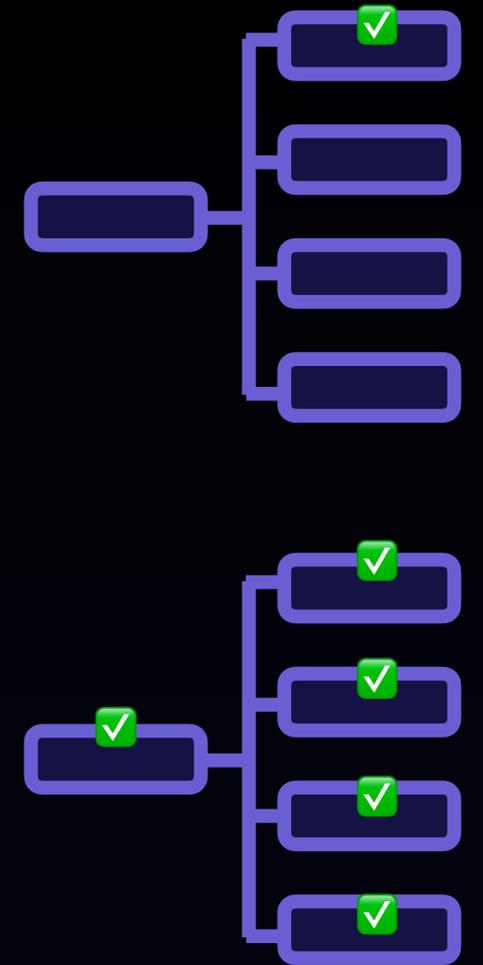
master.xml

Foundation and SPDX set

# Local and Global Consistency

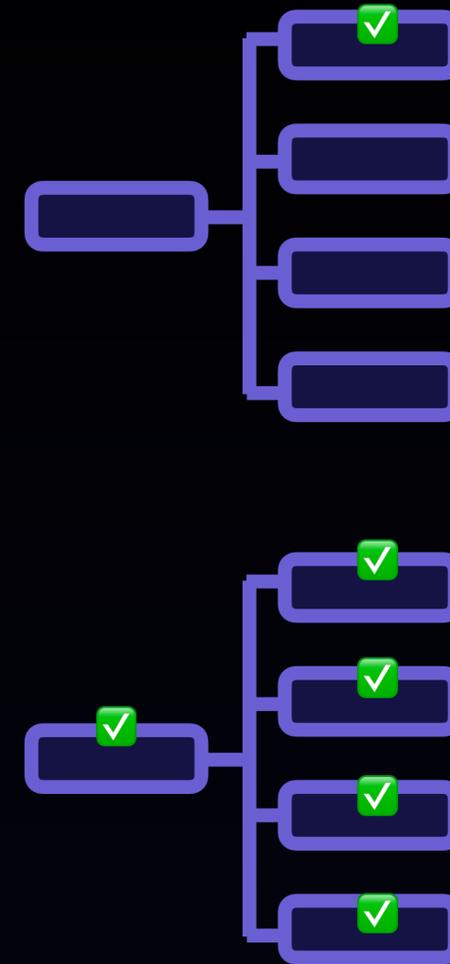


- ▶ Local Consistency:
  - code works for a specific repository in isolation
  - e.g.: code + git style, link check, license check, code compiles, local tests work
- ▶ Global Consistency:
  - all tests work for all repos in a manifest
  - e.g.: proofs, sel4test, benchmarks, user-level apps, VMs
- ▶ All 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

CAMkES **passing** CI **passing** Trigger **passing** Unit **passing**

# 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

 seL4Test **passing**

 Proofs **passing**

 Proof Sync **passing**

 seL4Bench **passing**

 CAmkES **passing**

 Camkes VM **passing**

# Mandatory Local Tests

## ► Minimal set of checks for any Foundation

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

allocman: Add missing calls to `_end_operation` #58

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

Conversation 1 Commits 2 Checks 15 Files changed 1

kent-mcleod commented on 11 Mar

If either `allocman_configure_mspace_reserve` or `allocman_configure_utspace_reserve` returns early it still needs to end the started operation otherwise the allocator will be left in an invalid state leading to it's watermarks never being refilled.

kent-mcleod added 2 commits 7 months ago

- allocman: Add missing calls to `_end_operation` ... 029dd1f
- allocman: Add error message for misconfiguration ... b44d146

kent-mcleod requested a review from lsf37 4 months ago

lsf37 approved these changes on 24 Jun

lsf37 left a comment

Add more commits by pushing to the `kent/allocman` branch on `kent-mcleod/seL4_libs`.

Changes approved  
1 approving review by reviewers with write access. [Learn more.](#) [Show all reviewers](#)

1 approval

Some checks were not successful  
14 successful and 1 failing checks

- CI / License Check (pull\_request) Successful in 12s (Required) [Details](#)
- PR / Gitlint (pull\_request) Successful in 12s [Details](#)
- seL4Test / Simulation (armv7a, gcc) (pull\_request) Successful in ... [Details](#)
- seL4Test / Simulation (armv7a, clang) (pull\_request) Successful ... [Details](#)
- seL4Test / Simulation (armv8a, gcc) (pull\_request) Successful in ... [Details](#)
- seL4Test / Simulation (armv8a, clang) (pull\_request) Successful ... [Details](#)

This branch is out-of-date with the base branch  
Merge the latest changes from `master` into this branch.  
This merge commit will be associated with `gerwin.klein@proofcraft.systems`.

<> 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 2 Checks 15 Files changed 1 +22 -14

allocman: Add error message for misconfiguration b44d146 ▾

> DCO	DCO / DCO succeeded on 11 Mar in 0s
> seL4Test on: pull_request	DCO All commits are signed off! <a href="#">View more details on DCO</a>
✓ PR on: pull_request	
✓ Gitlint	
✓ Trailing Whitespaces	
✓ Portable Shell	
✓ CI on: pull_request	1
✓ License Check	
✓ Links	
✗ Style	
✗ style	



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 requests

- **seL4test-sim:**
- **Preprocess:** C
  - if this succeeds
  - if this fails,

## ► For expensive

- trigger manual
- reviewer-role
- add **Test with**
- **seL4test-hw-b**
- **seL4test-hw:** r
- **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: I4v repo

- ▶ Runs standard tests + proofs
- ▶ Most compute-expensive
- ▶ One AWS instance per platform
  - currently 5 in parallel, each
  - 3-4h each for a full build
  - caches previous proofs
  - detailed logs as build artifacts
- ▶ Triggers on
  - pull-request to I4v repo on
  - merge to main deployment
  - changes to verification-manifest
- ▶ On success on merge to main
  - updates manifest
  - triggers binary verification

**Proofs (ARM)**  
succeeded 7 days ago in 3h 56m 52s

Search logs

>  Set up job 3s

▼  Proofs 3h 56m 26s

```
1 ▶ Run seL4/ci-actions/aws-proofs@master
14 /home/runner/work/_actions/seL4/ci-actions/master/js/./aws-proofs/steps.sh
15 ▶ AWS
26 ▶ Action setup
29 ▶ Setting up
49 ▶ Cache
51
52 Testing for L4V_ARCH=ARM:
53 Testing Orphanage for ARM
54 Running 51 test(s)...
55 Finished tests-xml-correct passed ( 0:00:00 real, 0:00:00 cpu, 0.01GB)
56 Finished haskell-translator passed ( 0:00:20 real, 0:00:10 cpu, 0.03GB)
57 Finished isabelle passed ( 0:06:34 real, 0:27:33 cpu, 10.01GB)
58 Finished Docs passed ( 0:00:16 real, 0:00:38 cpu, 1.29GB)
59 Finished Lib passed ( 0:01:00 real, 0:05:58 cpu, 4.00GB)
60 Finished CamkesGlueSpec passed ( 0:00:30 real, 0:01:17 cpu, 2.10GB)
61 Finished Sep_Algebra passed ( 0:00:42 real, 0:04:15 cpu, 4.44GB)
62 Finished HaskellKernel passed ( 0:10:03 real, 0:07:31 cpu, 2.22GB)
63 Finished Concurrency passed ( 0:01:30 real, 0:07:37 cpu, 5.80GB)
64 Finished SepTactics passed ( 0:01:26 real, 0:06:01 cpu, 3.23GB)
65 Finished EVTutorial passed ( 0:01:37 real, 0:07:41 cpu, 4.55GB)
66 Finished TakeGrant passed ( 0:00:30 real, 0:01:20 cpu, 2.04GB)
67 Finished ASpec passed ( 0:03:27 real, 0:07:58 cpu, 6.14GB)
68 Finished ExecSpec passed ( 0:04:16 real, 0:10:27 cpu, 7.69GB)
69 Finished DSpec passed ( 0:02:14 real, 0:05:59 cpu, 4.92GB)
70 Finished SepDSpec passed ( 0:01:05 real, 0:04:59 cpu, 6.10GB)
71 Finished DSpecProofs passed ( 0:01:31 real, 0:07:50 cpu, 2.45GB)
72 Finished ASepSpec passed ( 0:00:24 real, 0:00:56 cpu, 1.70GB)
73 Finished SysInit passed ( 0:03:16 real, 0:14:53 cpu, 3.79GB)
```





# Logs and artefacts

# What can you do with these tests?



Run on your own repo fork

cmake: add a sanity check  
PR #3653  
Re-run all jobs

Summary

cmake: add a sanity check  
RefMan #3150

Re-run all jobs

Summary

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

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



## HW Run (TK1, armv7a, gcc)

succeeded 5 days ago in 38m 58s. [View latest attempt.](#)

## Proofs (ARM)

succeeded 7 days ago in 3h 56m 52s

Search logs

> Download image

Run

```
1 ▶ Run seL4/ci-actions/seL4test-hw
9 /home/runner/work/_actions/seL4/
10 ▶ Setting up
92
93 ▶ TK1_debug_gcc_32
1910 TK1_debug_gcc_32 succeeded
1911
1912 ▶ TK1_debug_MCS_gcc_32
3933 TK1_debug_MCS_gcc_32 succeeded
3934
3935 ▶ TK1_debug_hyp_gcc_32
5750 TK1_debug_hyp_gcc_32 succeeded
5751
5752 ▼ TK1_debug_hyp_MCS_gcc_32
5753 -----[ start test TK1_de
5754 +++ tar xvzf ../TK1_debug_hyp_
5755 images/
5756 images/seL4test-driver-image-a
5757 +++ mq.sh sem -info jetson1
5758 Lock for jetson1 currently free
5759 +++ time mq.sh sem -wait jetson
5760 0.02user 0.01system 0:02.83elap
5761 0inputs+0outputs (0major+3787minor/pagesfaults 0swaps
5762 +++ time mq.sh run -c </testsuite> -s jetson1 -d 600 -t -1 -w 8 -l results.xml -n -
f images/seL4test-driver-image-arm-tk1
5763 Lock acquired, we are allowed to run
```

- ▶ 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!

```
../aws-proofs/steps.sh
:00:00 real, 0:00:00 cpu, 0.01GB)
:00:20 real, 0:00:10 cpu, 0.03GB)
:06:34 real, 0:27:33 cpu, 10.01GB)
:00:16 real, 0:00:38 cpu, 1.29GB)
:01:00 real, 0:05:58 cpu, 4.00GB)
:00:30 real, 0:01:17 cpu, 2.10GB)
:00:42 real, 0:04:15 cpu, 4.44GB)
:10:03 real, 0:07:31 cpu, 2.22GB)
:01:30 real, 0:07:37 cpu, 5.80GB)
:01:26 real, 0:06:01 cpu, 3.23GB)
:01:37 real, 0:07:41 cpu, 4.55GB)
:00:30 real, 0:01:20 cpu, 2.04GB)
:03:27 real, 0:07:58 cpu, 6.14GB)
:04:16 real, 0:10:27 cpu, 7.69GB)
:02:14 real, 0:05:59 cpu, 4.92GB)
:01:05 real, 0:04:59 cpu, 6.10GB)
:01:31 real, 0:07:50 cpu, 2.45GB)
:00:24 real, 0:00:56 cpu, 1.70GB)
:03:16 real, 0:14:53 cpu, 3.79GB)
```



Thank You