

To migrate or not to migrate.

K0xI

Rust for Linux

Storage Drivers

A new Rust driver candidate to be merged

02 / 14

An eighty-patch series replaces a C storage driver with a Rust one. **Should the maintainer take it?**

ARTIFACTS

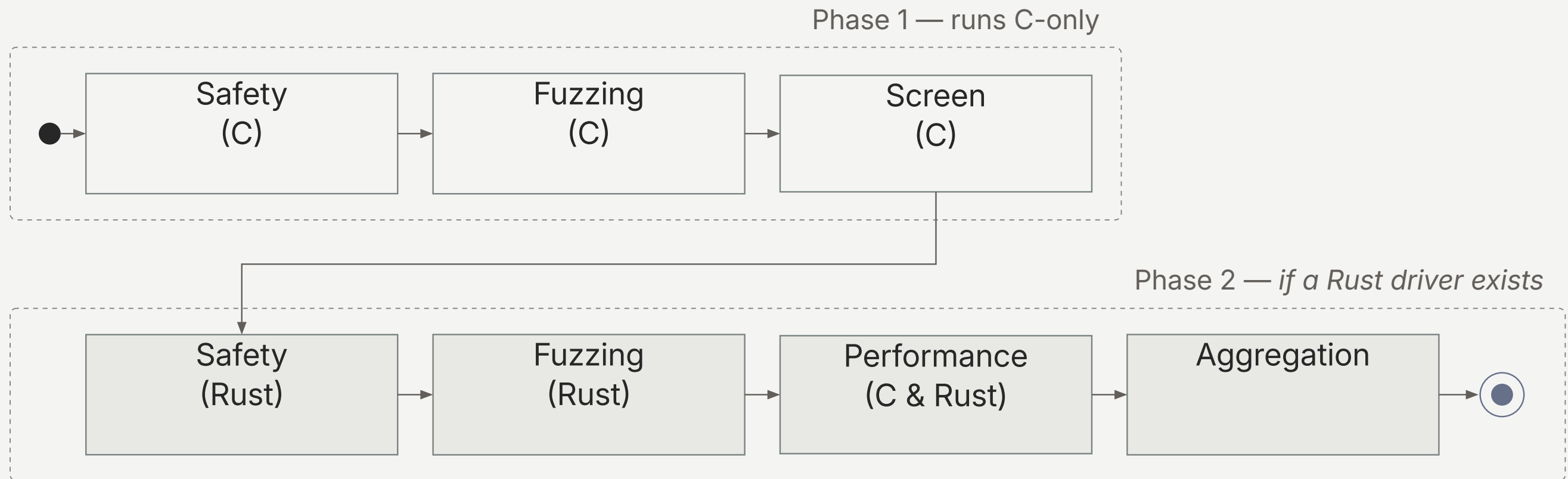
- Candidate's Code
- Rationale via the RFC
- Occasionally some shared tests

| More evidence needed for a grounded decision.

KOxl - Kernel Oxidation Instrument

03 / 14

- 3-Gate, 4-Lens evidence gated pipeline
- 2-Phase:
 1. Is this C driver a good candidate?
 2. Is this Rust counterpart a good fit?



`null_blk`

- “Multi queue aware block test driver... Written to facilitate testing of the blk-mq code.”
- First commit on 2013 `f2298c0`
- Any regression here blocks a patch series



`rnull`

- “The Rust null block driver rnull is an effort to implement a drop in replacement for null_blk in Rust.”
- First commit on 2024 `bc5b533`
- The first upstream Rust block driver



Fuzzing — do no harm

05 / 14

One driver crash in the whole study — on the C side.

CAMPAIGN	1	2	3	4	5	6	7	8	9	10	TOTAL
<code>null_blk</code> C	0	0	0	0	1	0	0	0	0	0	1
<code>rnull</code> Rust	0	0	0	0	0	0	0	0	0	0	0

Target-attributable crashes per campaign: a `null_blk` or `rnull` frame in the call stack. 10 × 24 h syzkaller campaigns per driver, KASAN guests.

PASS
VACUOUS

FAILS IF
 $p < 0.05$ and $A_{12} \geq 0.71$
Mann–Whitney U, Vargha–Delaney A_{12}

MEASURED
 $p = 0.37$, $A_{12} = 0.45$
nothing to compare

Safety — where the unsafe went

06 / 14

1,071

implicit unsafe operations
in null_blk's 2,491 lines

0

`unsafe` blocks in rnull's
driver body, 211 lines

Not eliminated: **concentrated**. rnull's unsafe sits in the shared `rust/kernel/block/` layer, reviewed once for every Rust block driver.

Li et al. measured 0.76–3.49% unsafe in the three flagship Rust drivers. Two units side by side, not a ratio.

FAIL
SOFT

PASSES IF
elimination rate $\geq 34.2\%$
Li et al.'s driver-CVE fleet average

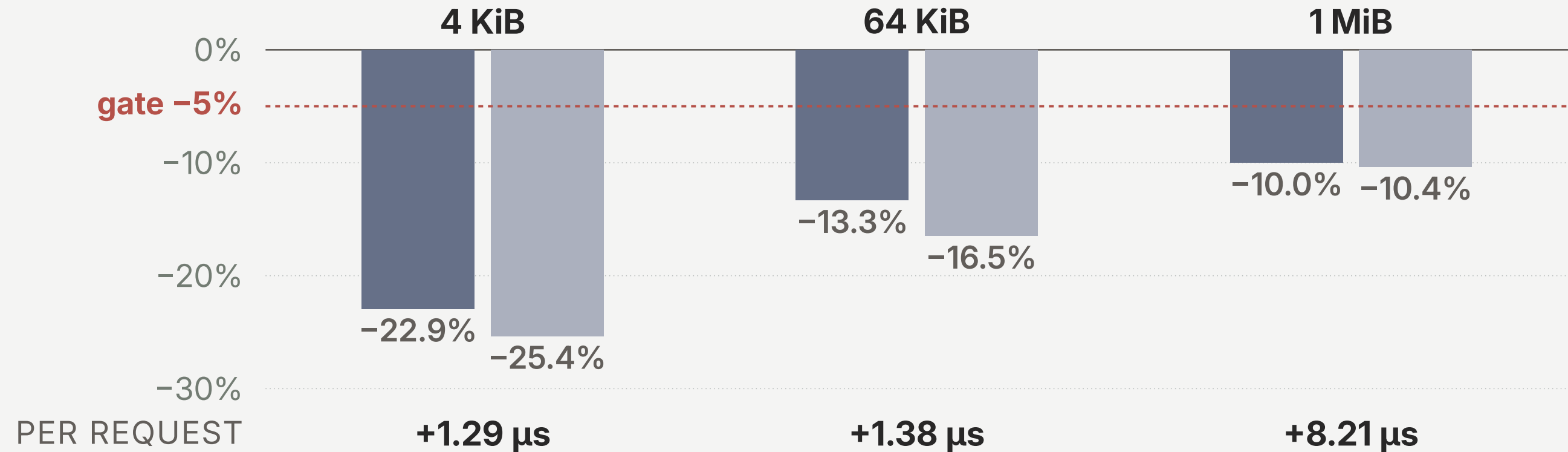
MEASURED
18.2%
2 of 11 classified fix commits

Performance — the shape, not the median

07 / 14

Median IOPS delta, rnull over null_blk

■ randread ■ randwrite



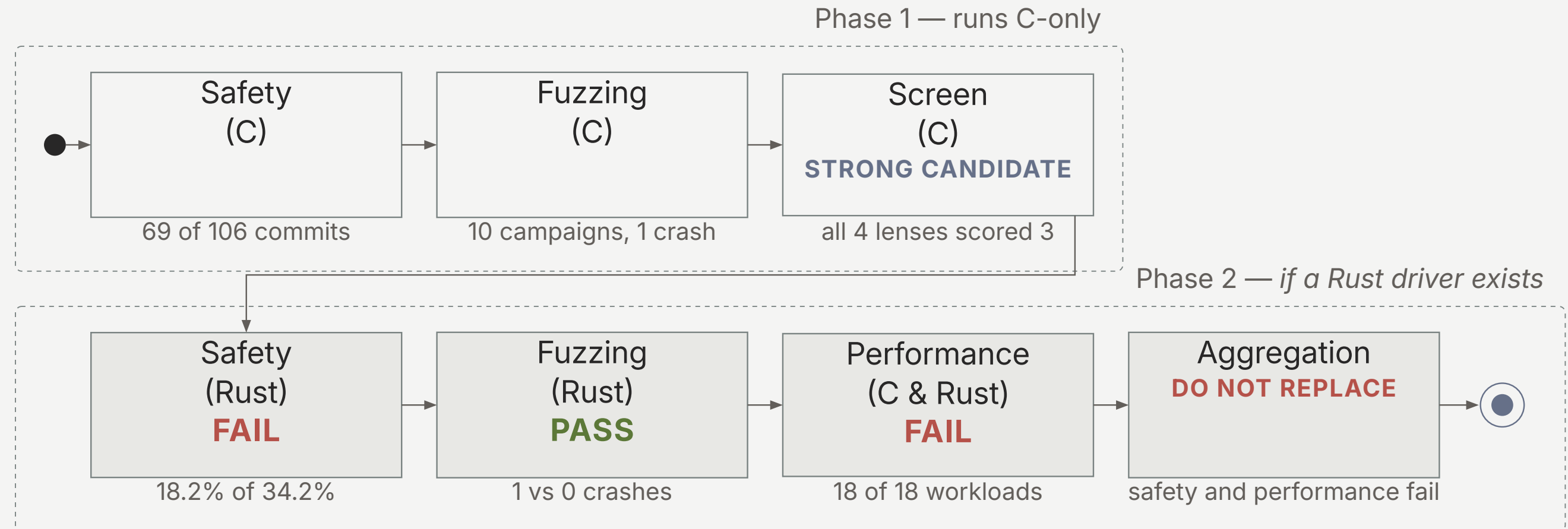
FAIL
UNIFORM

FAILS IF
any cell's CI below -5%
bootstrap CI on the median delta

MEASURED
18 / 18 cells
median delta -14.4%

The same pipeline, with the answers in

08 / 14



Phase 1 said `null_blk` is worth replacing. Phase 2 says *not by this driver, not yet* — and the three gates give three perspectives on it: the bugs Rust would have prevented, the crashes it did not add, the cost it charges to run.

Worst-of, not a weighted sum

09 / 14

SAFETY	FUZZING	PERF.	RECOMMENDATION
✓	✓	✓	Replace: Rust driver meets all criteria for production use
✓	✓	✗	Caution: safety and fuzzing pass but performance regresses
✓	✗	✓	Caution: fuzzing shows regression, investigate before replacing
✓	✗	✗	Do not replace: fuzzing and performance both regress
✗	✓	✓	Caution: safety elimination rate below threshold
✗	✓	✗	Do not replace: safety and performance fail ← this run
✗	✗	✓	Do not replace: safety and fuzzing both fail
✗	✗	✗	Do not replace: all dimensions fail

KOxI's verdict stage, transcribed: the (safety, fuzzing, performance) triple selects one of eight written recommendations. There is no weighted sum anywhere in the codebase.

What this does not show

10 / 14

- **One pair** — the pipeline transfers; the constants do not.
- **No memory backing** — the deltas bound abstraction cost, not device throughput.
- **Virtualised** — QEMU/KVM on one desktop: matched deltas, no absolute IOPS.

KOxl2 — the instrument, rewritten

11 / 14

KOxl v0.1 · the paper	KOxl2 v2.0.0-rc.1
shell and Python, 49 files	→ one Rust binary, pinned by a Nix flake
unpinned sources, rebuilt every run	→ locked sources, reproducible builds
gcc-built kernels only	→ any kernel you pin; gcc or clang/LLD
QEMU/KVM guests only	→ bare metal via kexec: <code>koxi metal</code>
each device at its own defaults	→ one geometry, read back from the guest
crash rate over budgeted hours	→ crash rate over hours actually run
rank tests: <i>is there a difference?</i>	→ non-inferiority: <i>is Rust not worse?</i>
no crashes on either side: pass	→ no crashes on either side: inconclusive
directory-name hashes, symlinks	→ identity-hashed manifests, resumable runs

Release candidate, 7 September 2026. No new numbers until 2.0.0, the publication-scale re-run.

- **KOxl2 2.0.0**: the publication-scale re-run, both devices matched.
- Gate rnull's **parity series**: memory backing, zones, discard.
- Repeat on Linux **7.2**, where mainline is today.
- Point Phase 1 at C drivers with **no Rust port yet** — which port pays off next?

To close

The language question is settled.
The per-driver question is not.

Thanks

Questions?

→ guilherme.lima@sou.inteli.edu.br