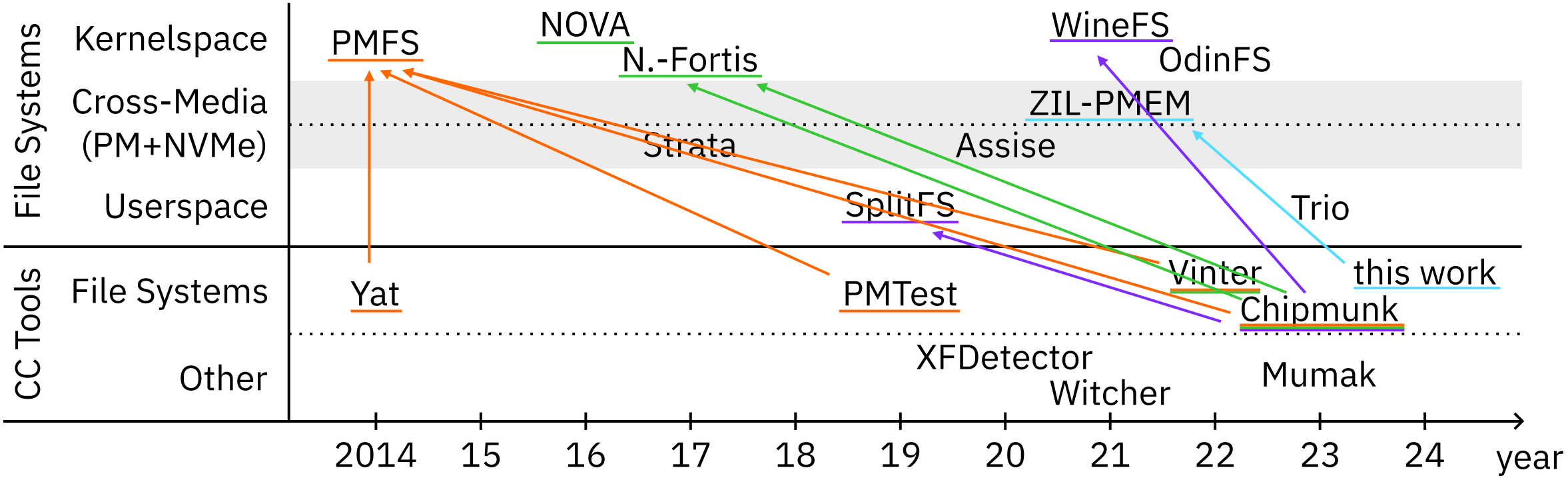
Karlsruhe Institute of Technology

Improvements in Crash Consistency Testing for Persistent Memory File Systems

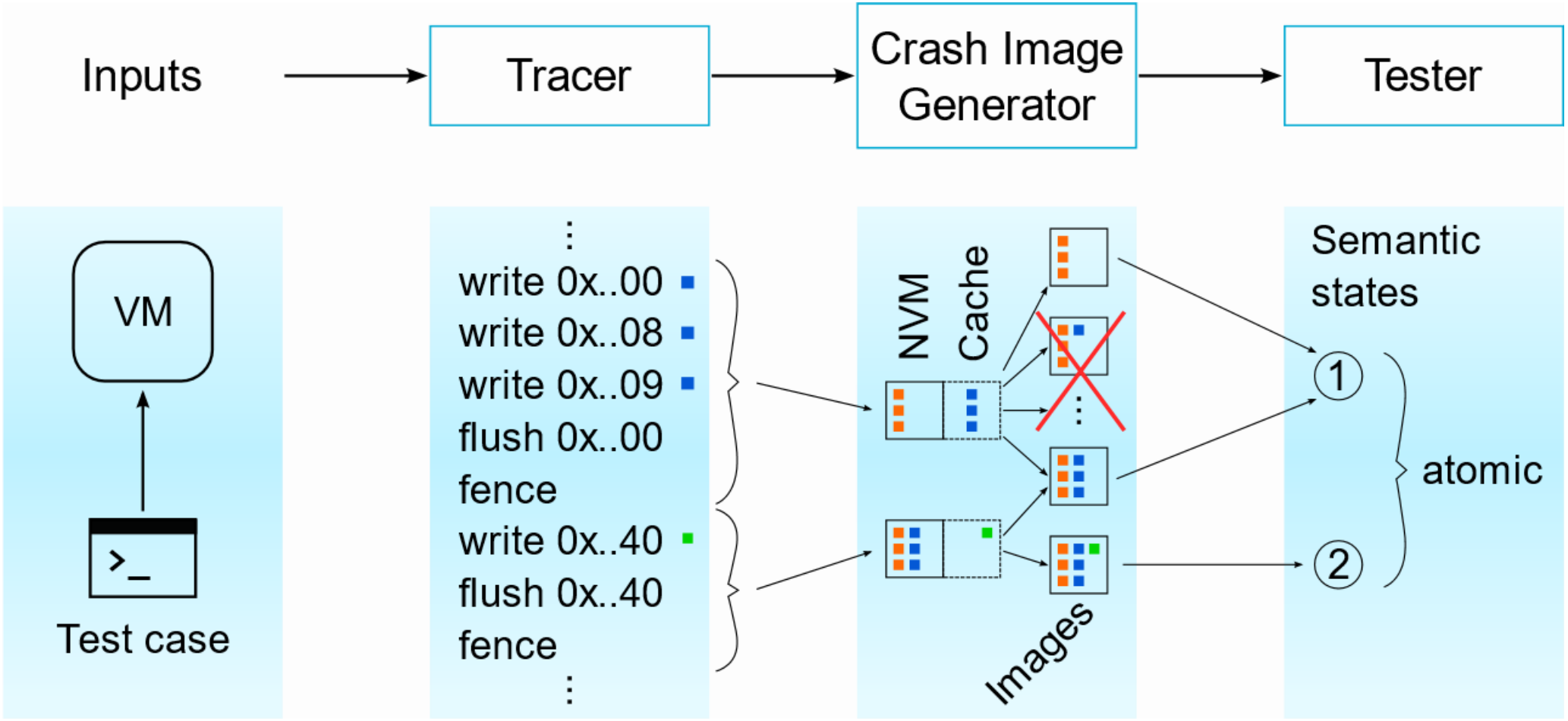
Lukas Werling, Thomas-Christian Oder, Lucas Wäldele, Daniel Ritz, Frank Bellosa

Timeline

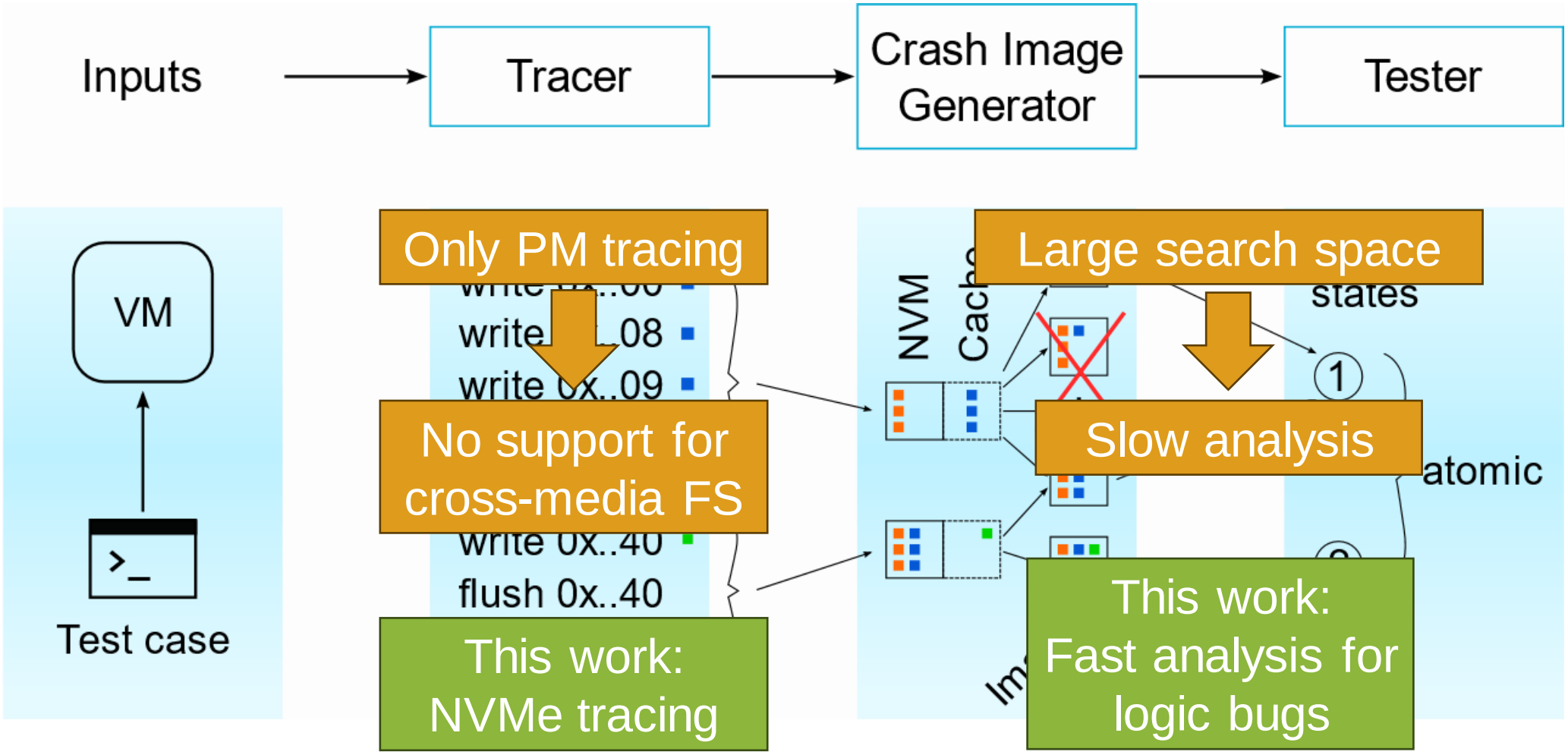


Crash consistency testing is not easy enough!

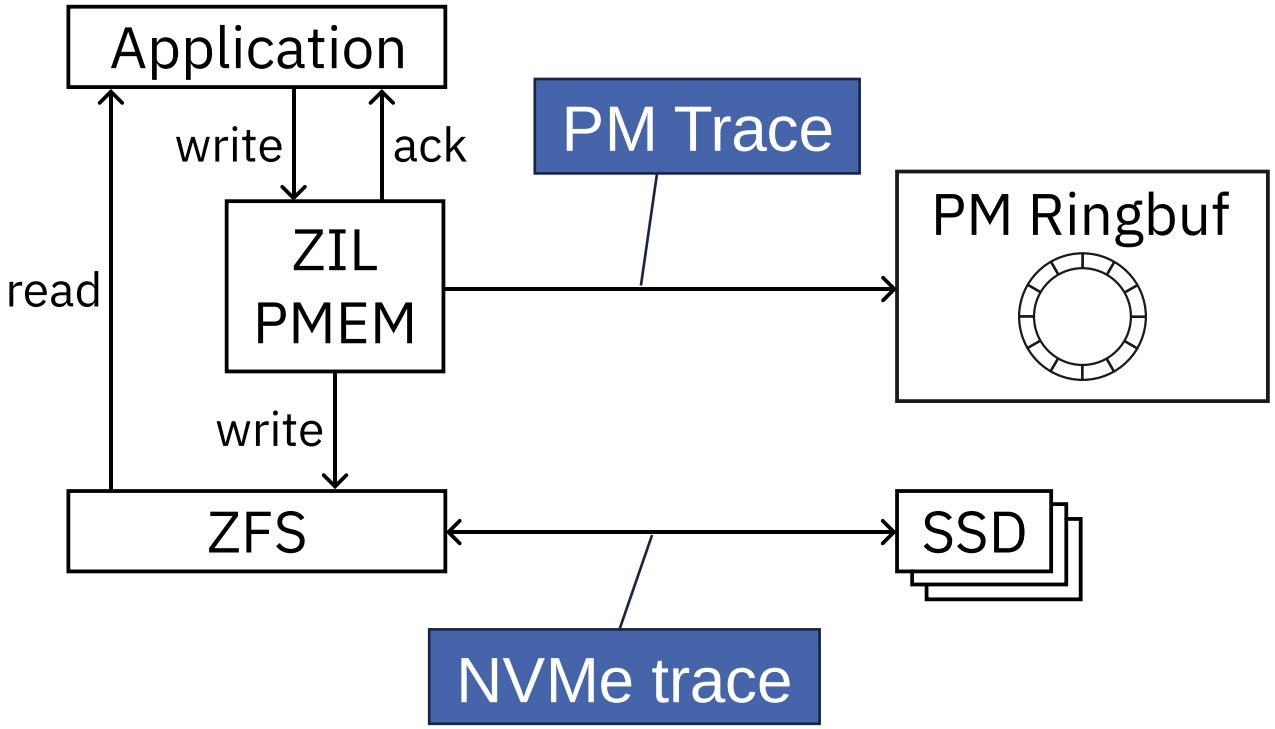
Background: Vinter (ATC'22)



Background: Vinter (ATC'22)

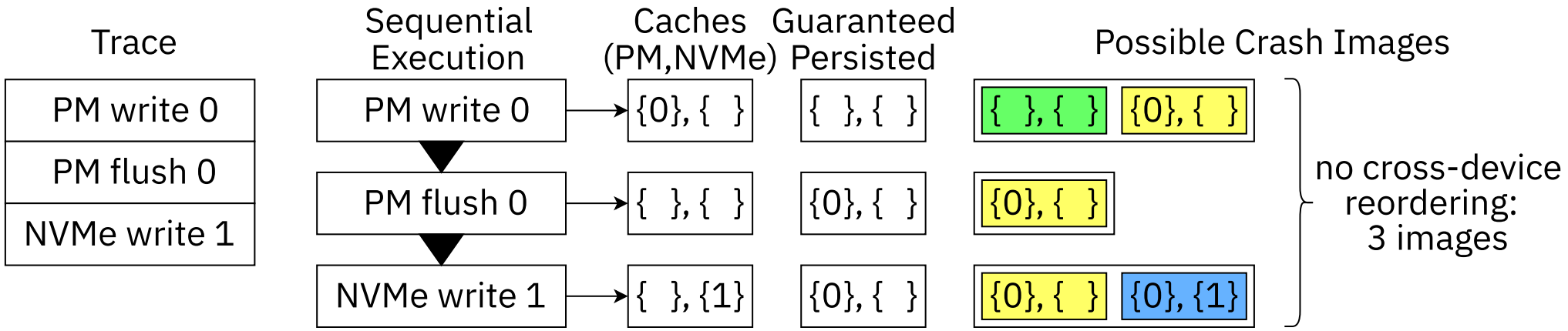


Cross-Media File System: ZIL-PMEM



- NVMe write
  - ≈ PM stores
  - Write one or more blocks
- NVMe flush
  - ≈ PM clflush + fence
  - Commit previous writes
- NVMe is asynchronous!
  - Submission
  - Completion

Cross-Media Crash Images



Is it possible to reorder PM and NVMe writes?

{ }, {1}

No!

- NVMe submission: UC write
- NVMe completion: interrupts
  → memory ordering

Cross-Media Analysis: Implementation

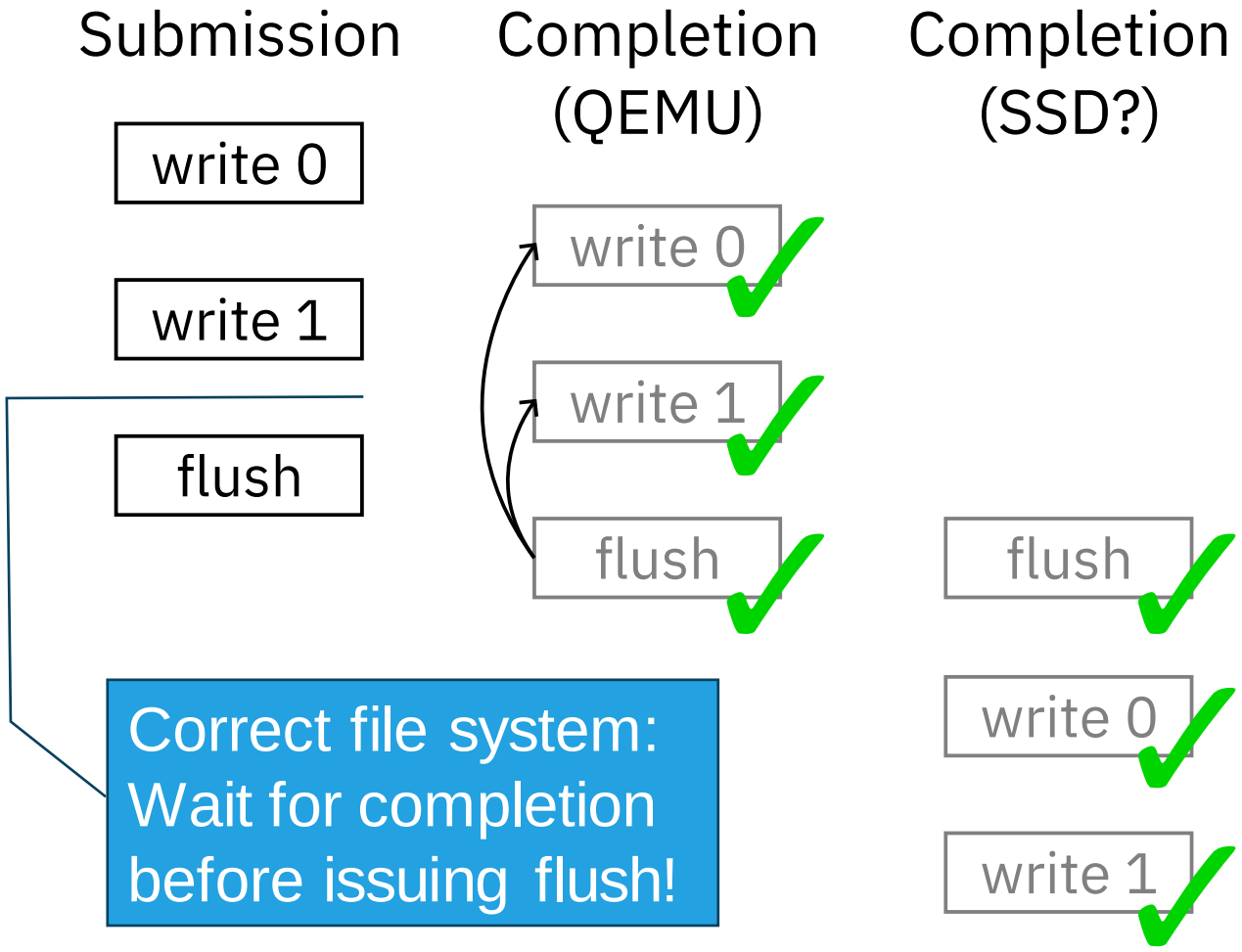
- Prototype called Permanent
- Tracer: Plain QEMU with binary translation, no PANDA
  - Hooks in virtual NVMe device
  - Hooks for memory accesses in code generation
- Crash Image Generator and Tester straightforward
- Lots of limitations
  - No heuristic, missing optimizations, …

Results and Discussion

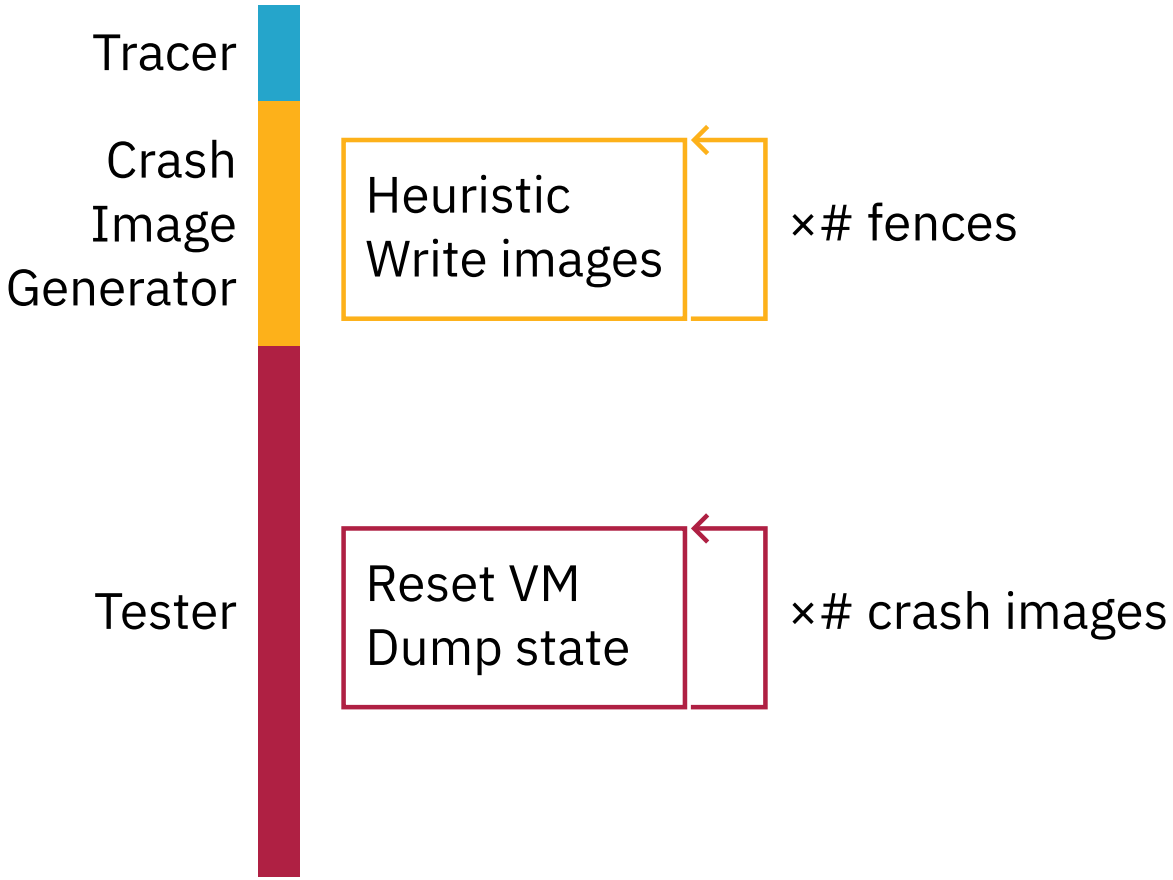
- Original Vinter file system tests
- No bugs in ZIL-PMEM!

NVMe spec is not strict
QEMU and real SSDs might differ

- Delayed completions
- Reordered completions



Vinter's Slow Analysis



Goal: fewer images, similar results

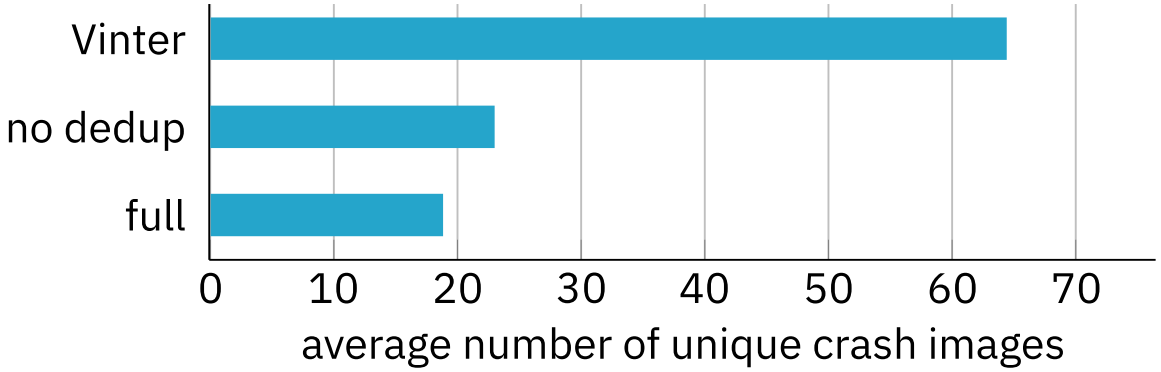
- Most FS bugs are logic bugs (Chipmunk, EuroSys'23)
- Mumak (EuroSys'23)
  - Consider fewer fences
  - Generate fewer images
  - Trace analysis

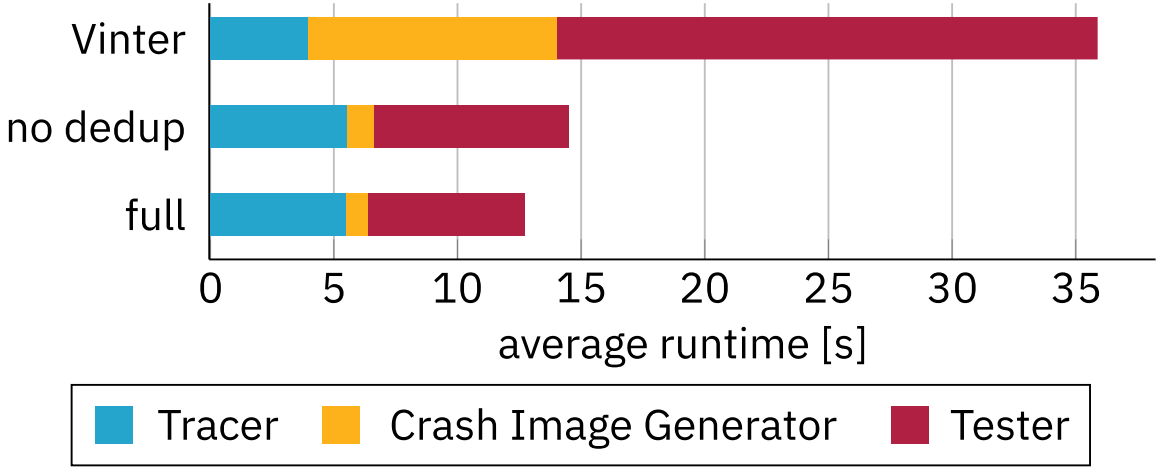
Vinter-FPT: Approach

Property	Vinter	Vinter-FPT / Mumak
Where?	Memory fences	Cache flushes, memory fences
When?	Always	Once per unique call stack trace
Contents?	1 no stores 1 all stores N subset of stores (heuristic)	1 no stores 1 all stores

- Tracer
  - Capture stack traces
  - Small overhead!
- Crash Image Generator
  - *Failure Point Tree* (FPT) for deduplication
  - No more heuristic

Vinter-FPT: Performance





70% fewer crash images

65% less overall runtime

Small tests suffer from slower tracer

Vinter-FPT: Results vs. Vinter

Observation	# tests
Identical states	25
Fewer states, similar result	17
Fewer states, wrongly assumed atomic	6

Trace Analysis:
3× unordered flushes
2× missing flush

Same results for most tests

Trace analysis helps find remaining bugs

Only weak hints from trace analysis

Future Work

Improve Tracing

- Binary translation costly
  - Slowdown for all instructions
  - Limited ISA support
- Idea: Selective trapping with MPK
  - Patch relevant instructions
  - Remaining code unchanged

CXL Crash Consistency

- CXL consistency model is similar
- Opportunity: Custom CXL device
- Idea: Tracing at the device
  - Observe real store ordering
  - Verify CPU consistency model

Conclusion

- Crash consistency testing is not as prevalent as it should be
  - Special requirements, slow analysis
- This paper: Improvements to Vinter
  - Crash consistency testing for cross-media file systems
  - Fast analysis for logic bugs

Consider crash consistency testing for your PM projects!

https://github.com/KIT-OSGroup