Towards High-Performance Application-Level Storage Management

Simon Peter, Jialin Li, Doug Woos, Irene Zhang,
Dan R. K. Ports, Arvind Krishnamurthy,
Thomas Anderson, Mark Zbikowski

University of Washington

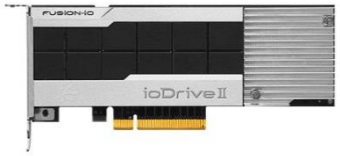
Funded in part by NetApp, Google

Talk Overview

- Dramatic increase in storage I/O performance
  - PCIe-attached flash
- **Problem:** OS storage stack becomes I/O bottleneck
  - Shared stack -> high overhead per I/O op
- **Solution:** Application-level storage architecture
  - Optimize storage stacks for each application
  - Assume storage devices remain on peripheral bus
- How to structure OS & hardware?

Problem: Storage I/O Performance

- Storage performance matters
  - Web servers
  - File servers
  - Key-value stores
  - Persistent lock managers
- Storage performance is improving 
  - PCIe attached flash
  - Flash-backed caching RAID controllers
- CPU frequencies have stalled
- **File system code expensive to run on CPU** 

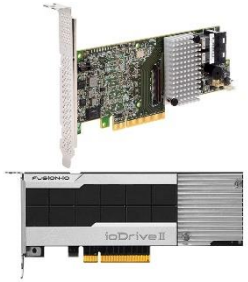


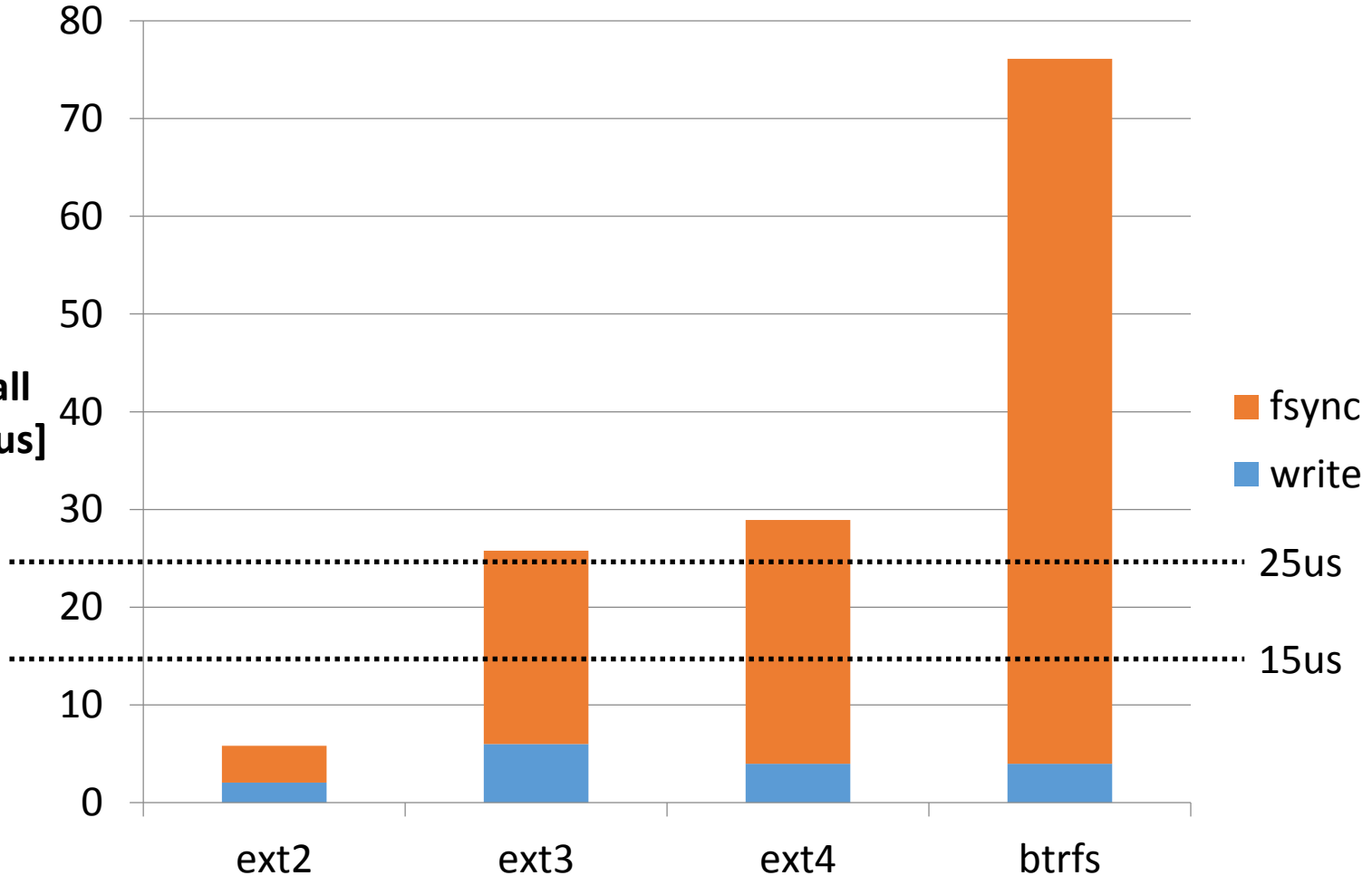
File system CPU overheads (writing 1K, then fsync)

= 25ms

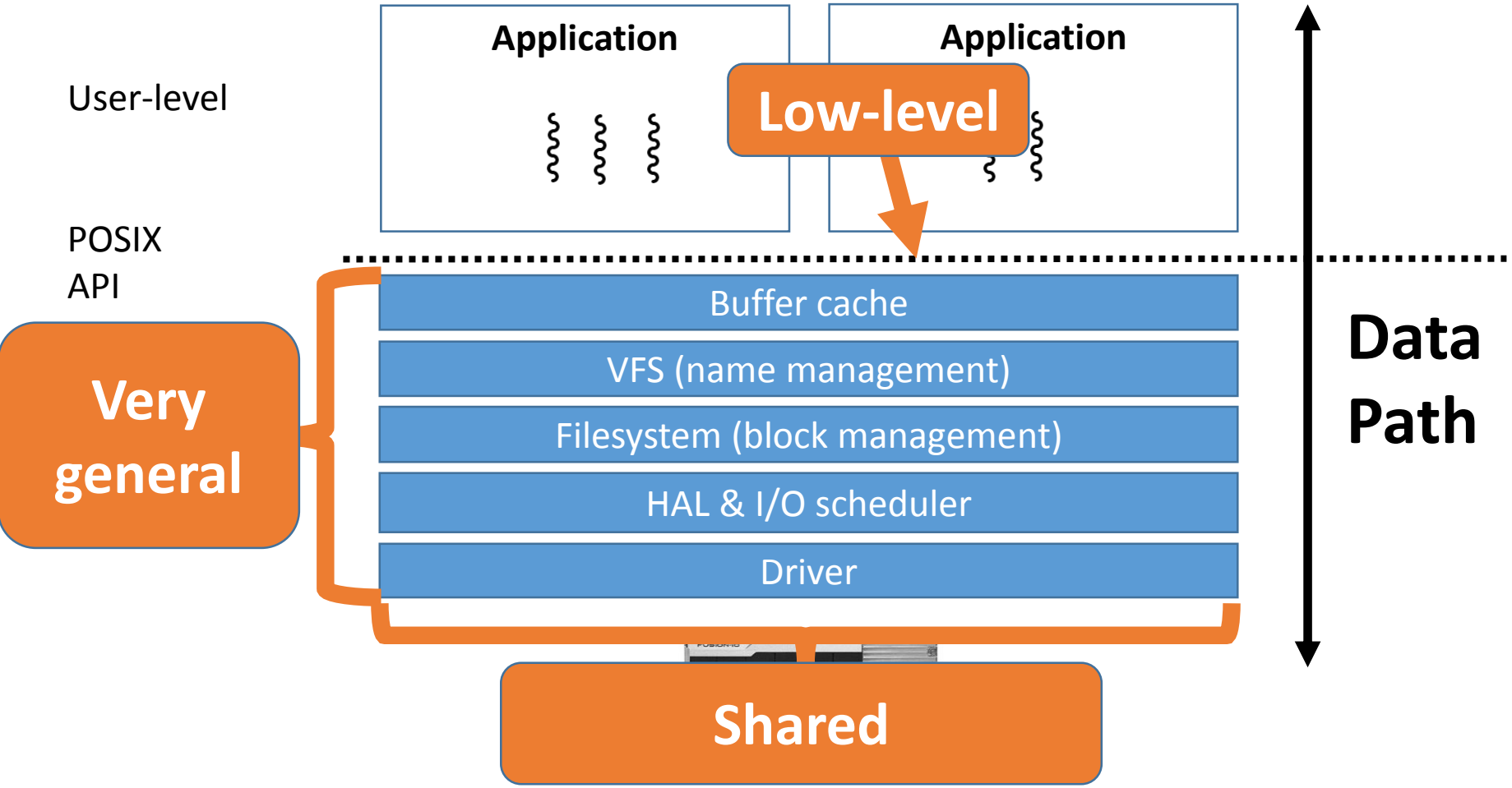
= 1ms

System call duration [us]

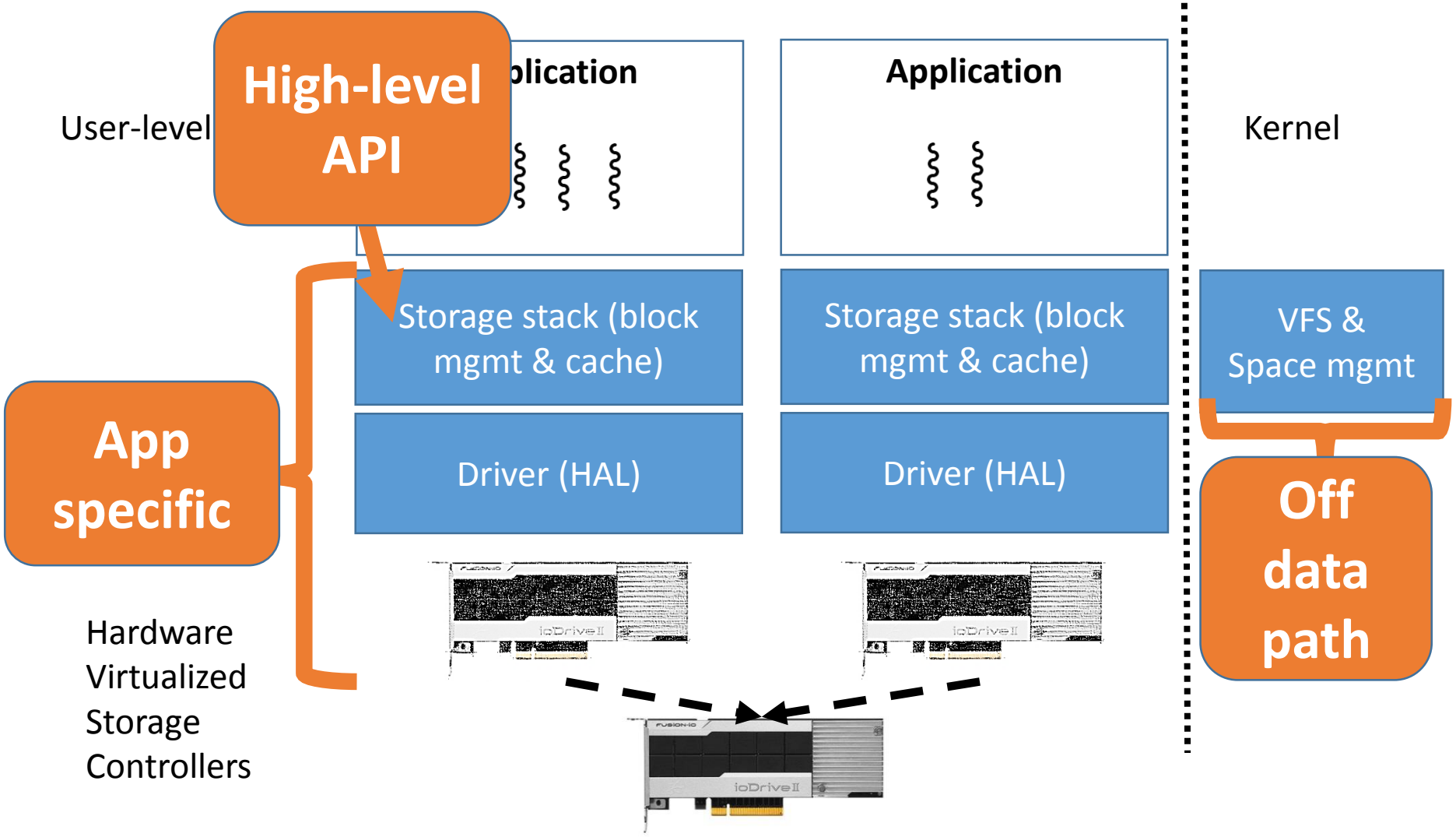




Today's Storage Stack



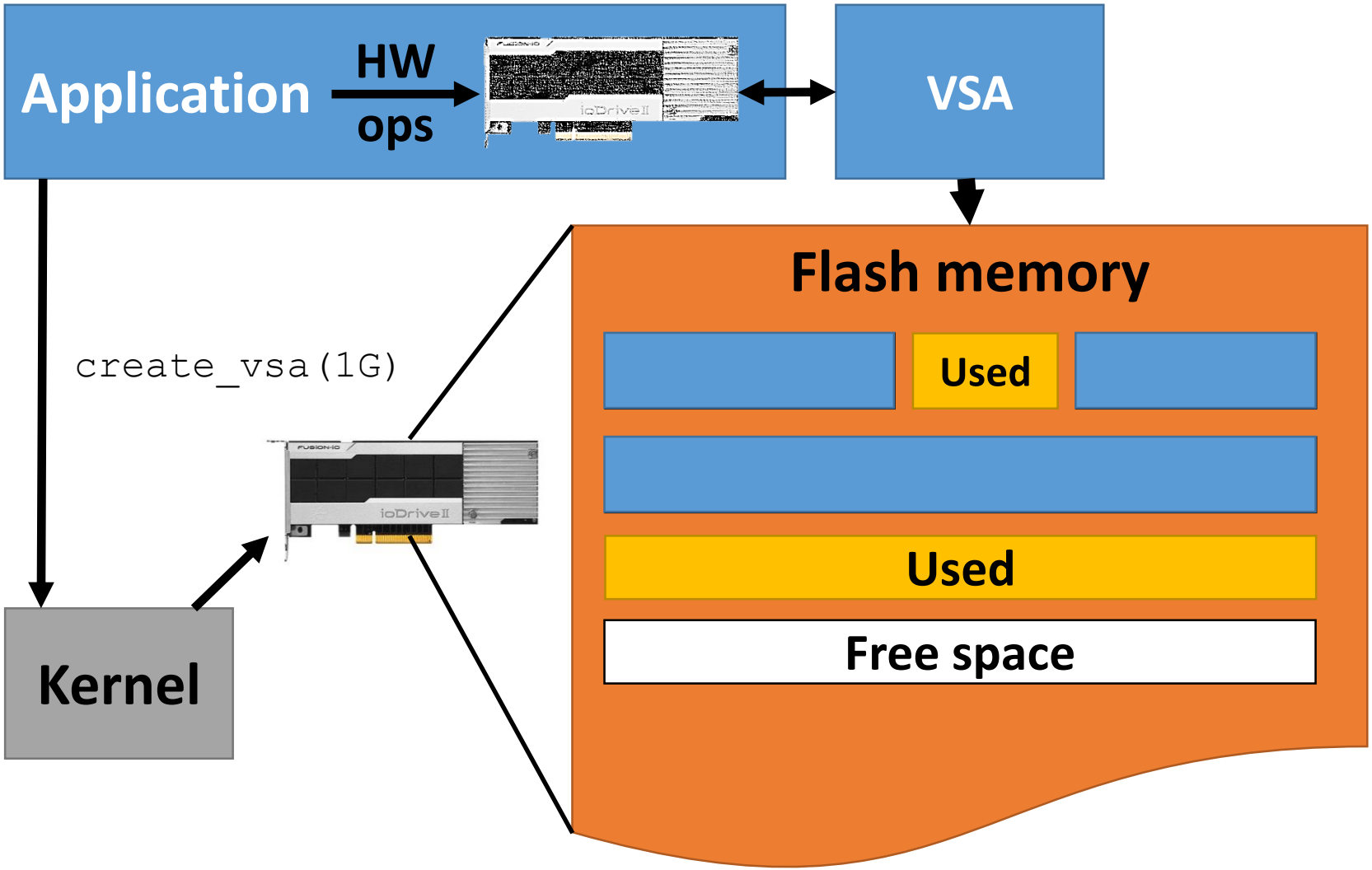
Our Storage Architecture



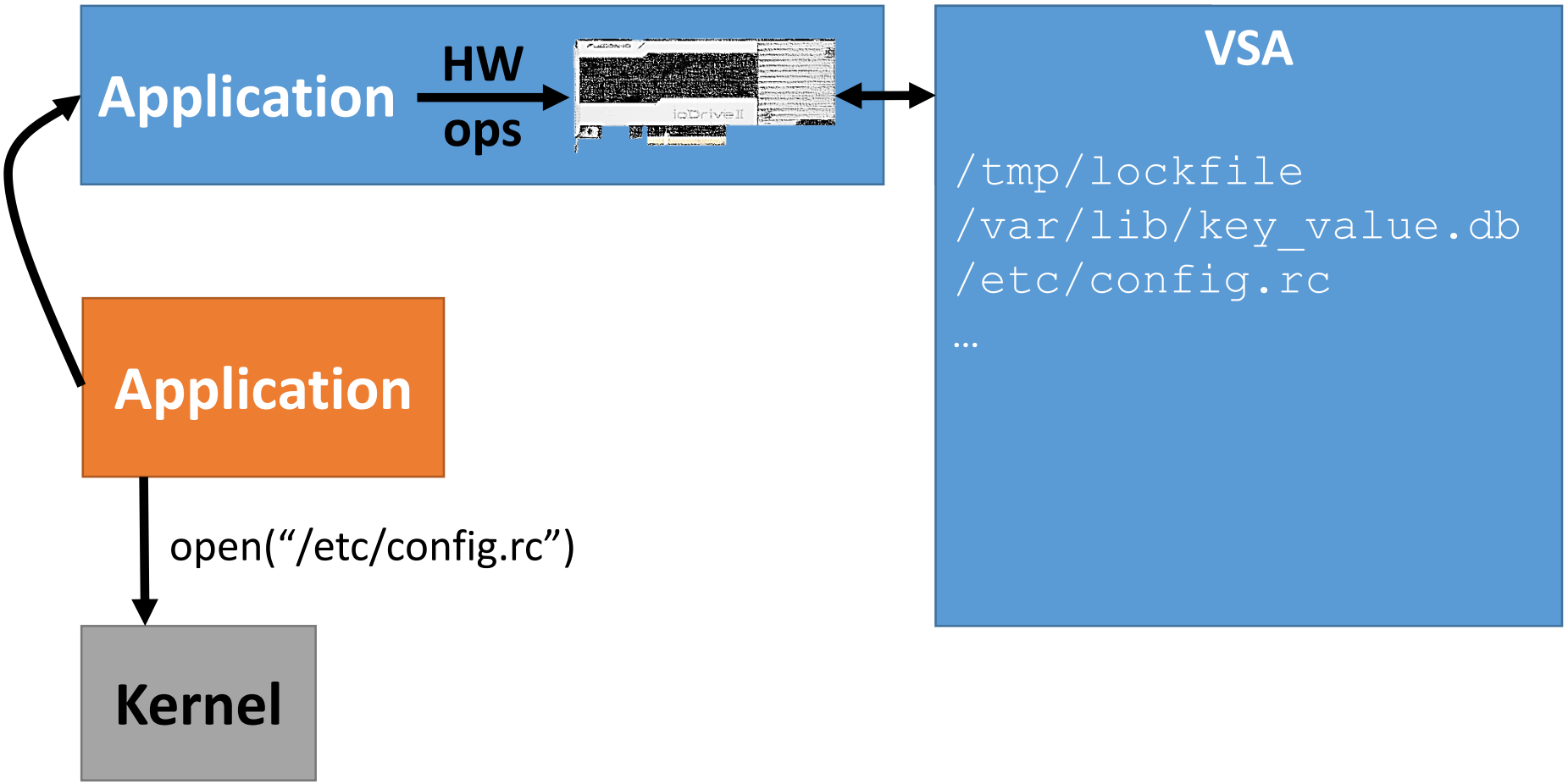
Storage Hardware Model

- Virtual storage devices
  - Protected command/DMA queues, interrupts
  - Can be provided by current technology
- Virtual storage areas (VSAs)
  - At least 1 per application
  - Map to physical storage extents
  - Protected in hardware
- Both managed by kernel

Storage Hardware Model: Example



Global File Name Resolution



High-Level API: Persistent Data Structures

- Examples: log, queue, hash table, tree
- Operations immediately persistent
  - To flash memory on storage peripheral

Goals:

- Scalability
- Robustness vs. crashes
- Low operation latency

Prototype Implementation

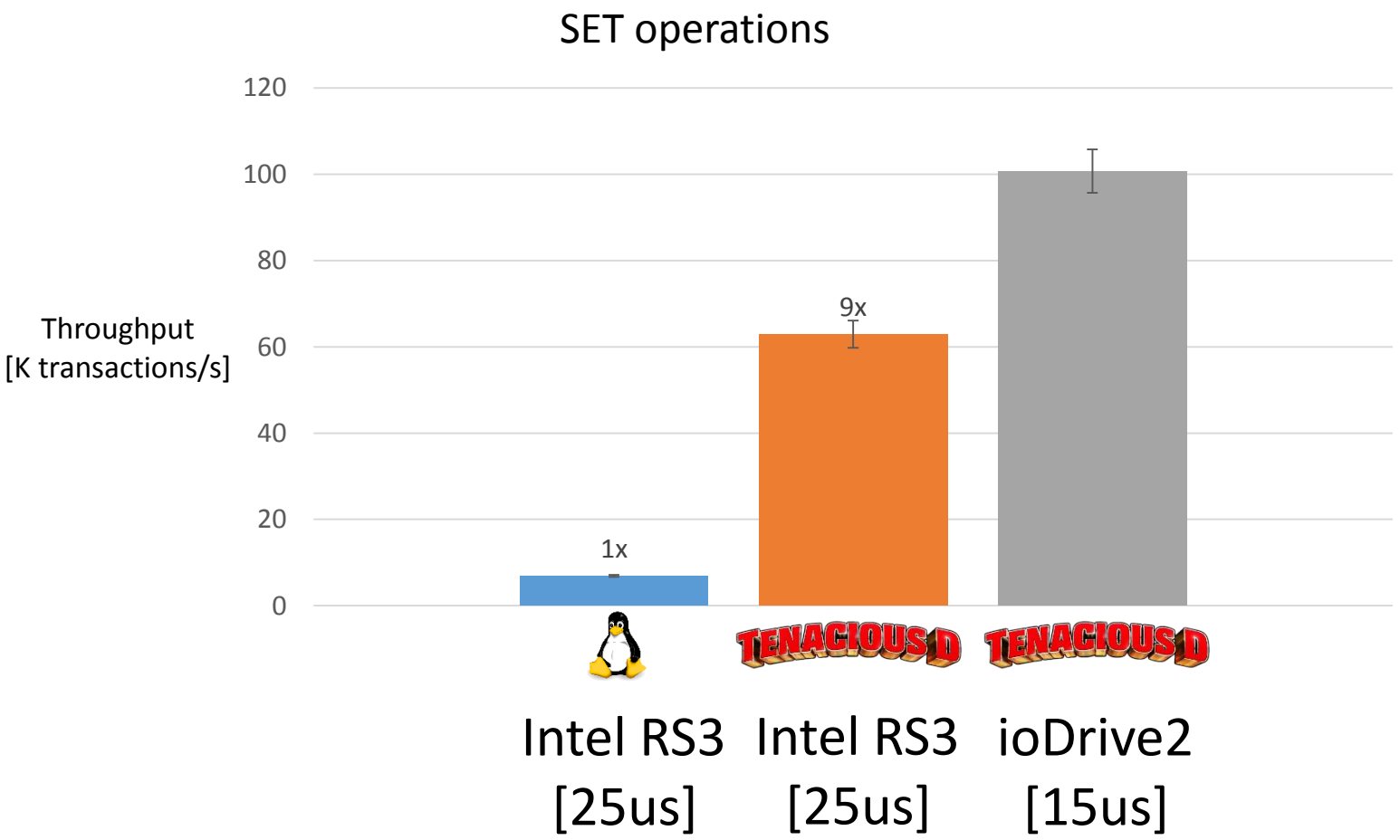
- In **Arrakis** OS
  - Has similar architecture for the network
- User-level device driver for Intel RS3 RAID controller
- Library of persistent data-structures
  - Persistent log

Evaluation: Redis NoSQL Store

- Redis logs operations to disk
- Modified to use **TenaciousD log**
  - 109 LOC changed
- Redis-benchmark
  - SET workload
  - 64K keys
  - 1K value size
- Compare to Linux on ext4
  - Server-sided performance

Evaluation: Redis NoSQL Store

- Cut SET latency by 81% (163 -> 31 us)



Summary

- New low-latency storage hardware
- **Problem:** Shared storage stack becomes bottleneck
- **Arrakis:** New OS storage architecture
  - HW/SW co-design
- Application-level storage eliminates I/O bottleneck
  - 9x speedup for Redis
  - Scales with CPUs & storage HW performance

http://arrakis.cs.washington.edu