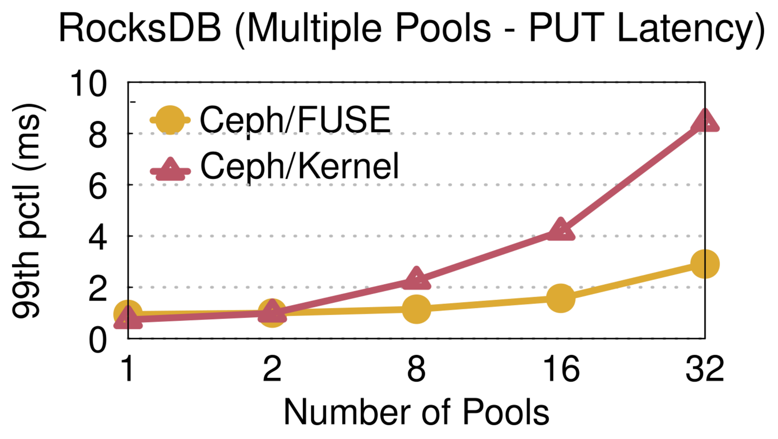
Diciclo: Flexible User-level Services for Efficient Multitenant Isolation

Giorgos Kappes, Stergios V. Anastasiadis
University of Ioannina, Ioannina 45110, Greece

Motivation

Problem

- Workload collocation causes severe performance variability & slowdown
- Put latency (99%ile)
  FUSE: up to 3.1x; kernel: up to 11.5x
  (longer for 32 pools vs. 1 pool)



Reasons

- Contention on shared kernel structures
- Kernel dirty page flushers running on arbitrary cores

Dynamic Storage Provisioning

Nodes (per tenant)

- Application or storage servers

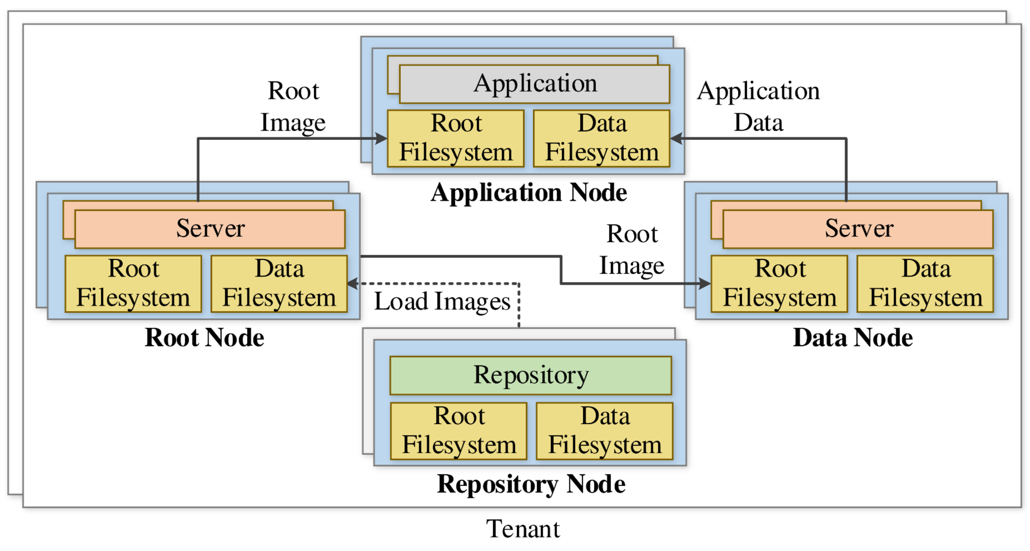
Machine

- Hosts multiple nodes possibly from different tenants

Container Storage System

Storage nodes of a tenant

- Root: uncompressed images
- Data: application data
- Repository: compressed images



Goals

1. Isolation

- Per-tenant I/O paths for improved performance isolation & fault containment

2. Compatibility

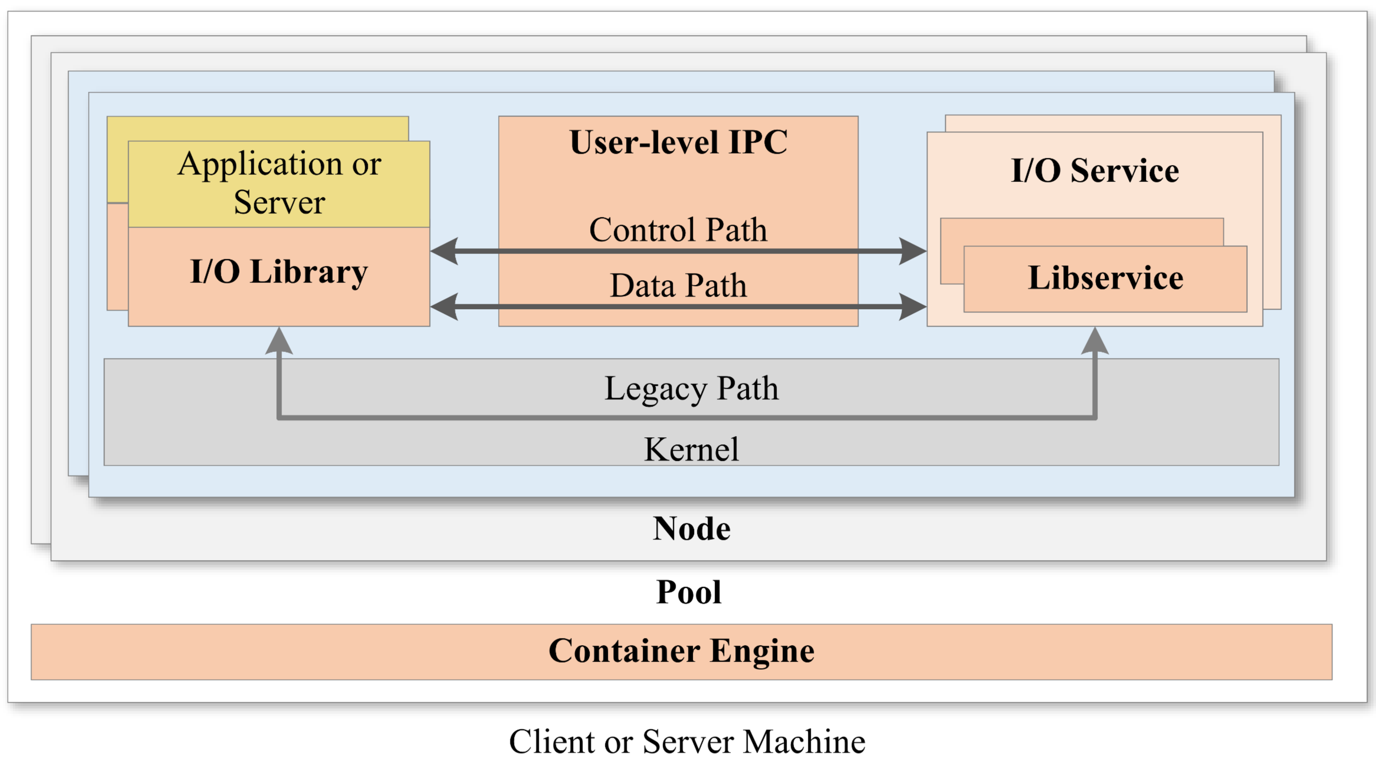
- POSIX-like interface for native multiprocess application access

3. Flexibility

- Per-tenant configuration of parameters & policies

4. Efficiency

- Lightweight on resources



Overview

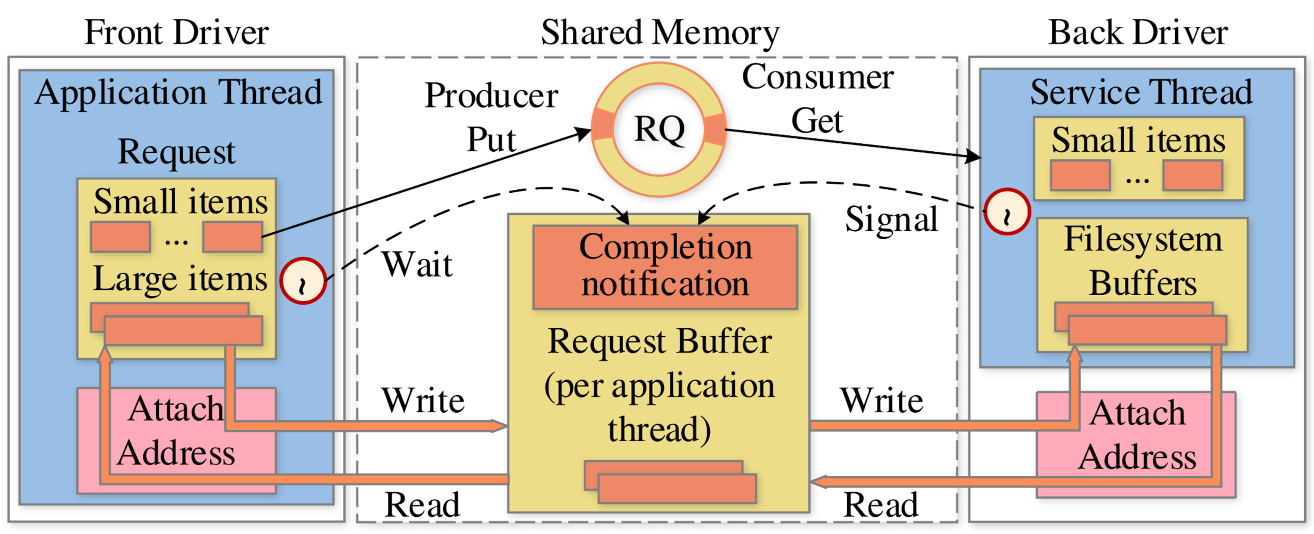
Per-tenant user-level I/O

- Move I/O services & I/O path from kernel to user level
- Separate instances of critical services per tenant
- Same design pattern at client & server

Polytropon toolkit

- Libservice: User-level storage function
- I/O Service: Provision container filesystems
- I/O Library: Provide fs access to processes
- Dual interface: Fast user-level IPC & legacy kernel-level path for compatibility

Inter-Process Communication



Request & data transfer at User-level through Shared Memory

- **Front driver** at I/O library, **Back driver** at I/O service

Request Queue (RQ)

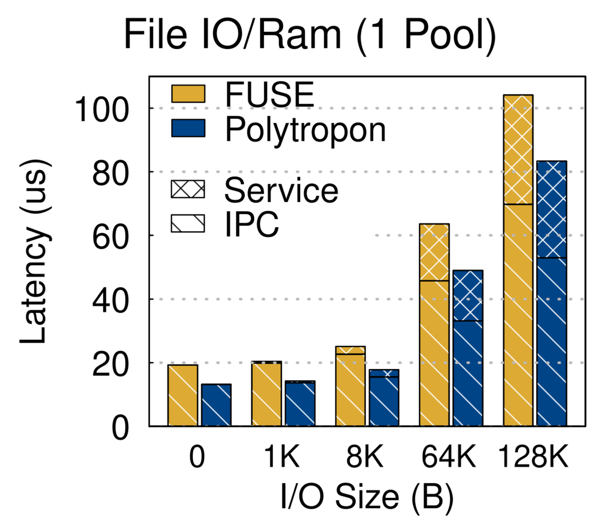
- I/O requests & small data
- Distinct queue per core group

Request Buffer

- Large data, completion notification
- Distinct per application thread

Results

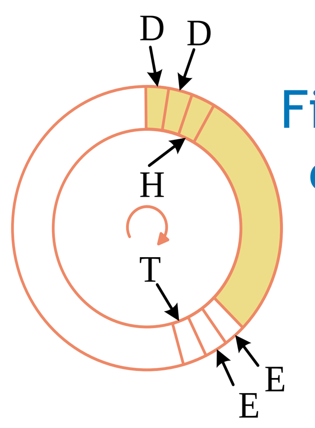
- Handling IPC at user-level makes Diciclo faster than FUSE
- FUSE: 32-44% longer to serve reads due to 25-46% higher IPC time



Relaxed Concurrent Queue Blocking (RCQB)

Goals

- High operation throughput (enqueue/dequeue)
- Low wait latency of the items in the queue



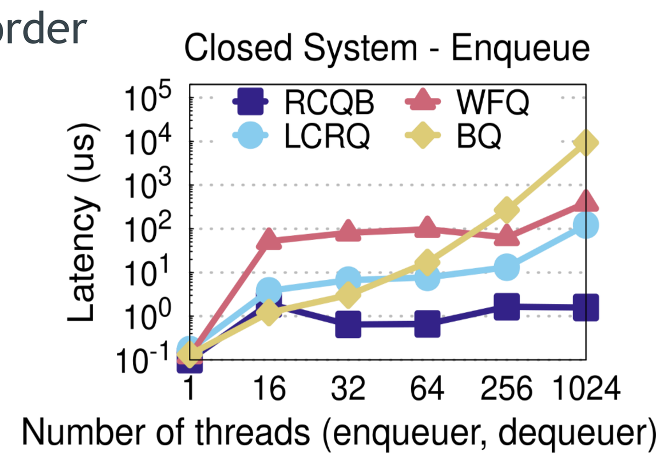
Fixed-size circular buffer

Two-stage relaxed ordering model

- 1st: Distribute operations sequentially
- 2nd: Complete concurrently potentially out of order

Results

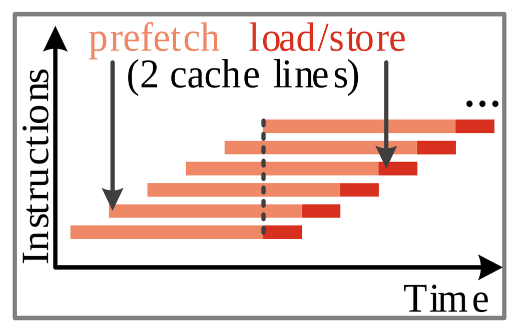
- RCQB achieves lower avg enqueue latency
- Avg enqueue latency (longer) LCRQ: up to 77x, WFQ: up to 246x, BQ: up to 5881x



Shared-memory Optimized (SMO) Copy

Optimized user-level data copy algorithm

- Copy: Source -> Shared Memory -> Destination
- 1st Stage: Non-temporal prefetch of 2 cache lines
- 2nd Stage: Non-temporal store of 2 cache lines

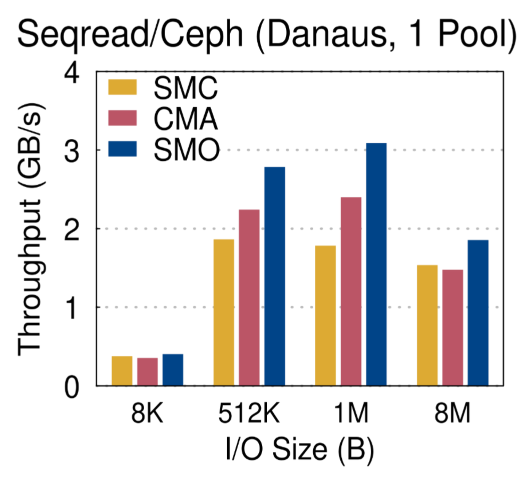


Other copy methods (using default memcpy)

- **Shared-memory Copy (SMC):** Source -> Shared Memory -> Destination
- **Cross-memory Attach (CMA)** with pvmreadv/pvmwritev syscalls: Source -> Destination

Results

- SMO pipelined copy makes data transfers faster
- SMO is 66% faster than SMC & 29% faster than CMA



Danaus

Ceph libservice

- Derived from libcephfs

Filesystem instance

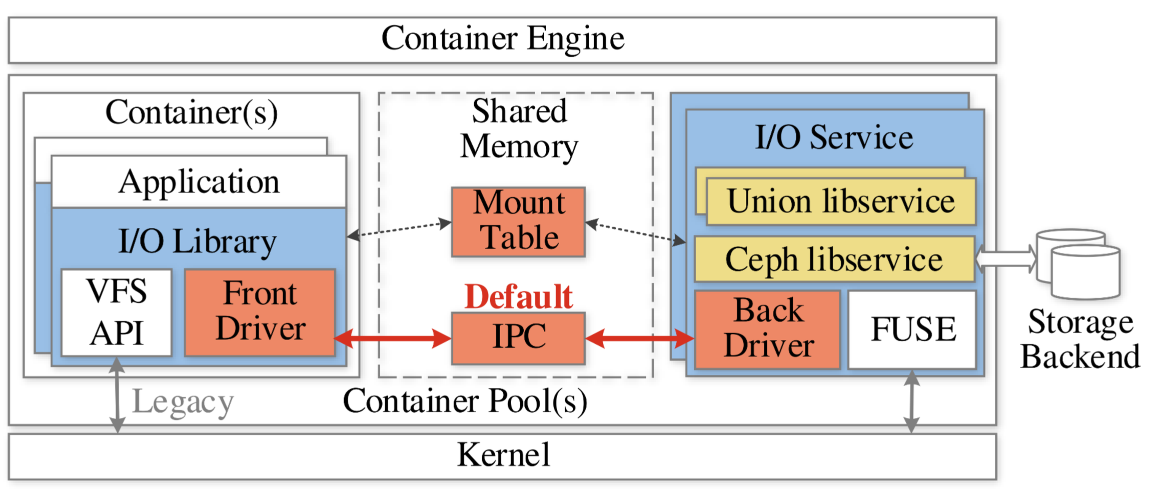
- Union libservice (optional)
- Ceph libservice

Union libservice

- Derived from unionfs-fuse

Container Engine

- Standalone process
- Manages container pools



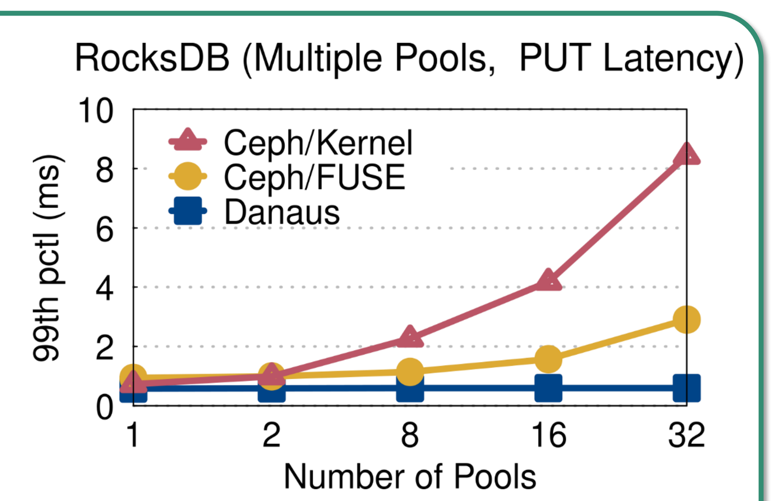
Results and Conclusions

Scaleout: serving multiple tenants with separate pools

- Danaus achieves faster I/O response
- Put latency (longer) FUSE: up to 4.8x, kernel: up to 14x

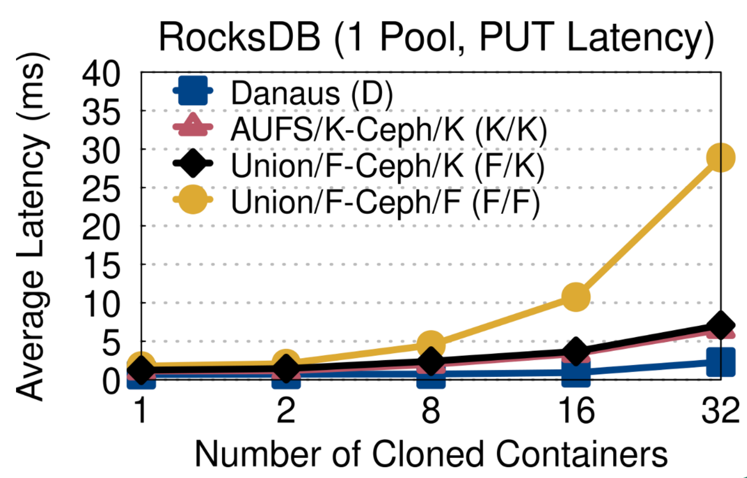
Scaleup: running multiple cloned containers in a single pool

- Danaus achieves lower put latency than Kernel & FUSE



Key lessons learned

1. Shared kernel causes performance interference on containers
2. Container images & data on shared filesystem beneficial
3. Functionality & execution separation improves isolation
4. Per tenant user-level client for decentralization/concurrency
5. Stable performance of user-level I/O access & handling



ACM Trans. Computer Systems, Vol.42, No.1-2, 47 pages, Feb 2024 (Invited at ACM SOSP 2025, Seoul, Republic of Korea)