Accelerated Disks and Flashes: LANL's Early Experience in Speeding Up Analytics Workloads Using Smart Devices

Qing Zheng, Scientist, Los Alamos National Laboratory (LANL)

5/23/23

LA-UR-23-25558

Overview

Problem

Scientific analytics increasingly bottlenecked on data operations

Trend

Smart devices offer opportunities for acceleration

LANL's Early Exploratory Work

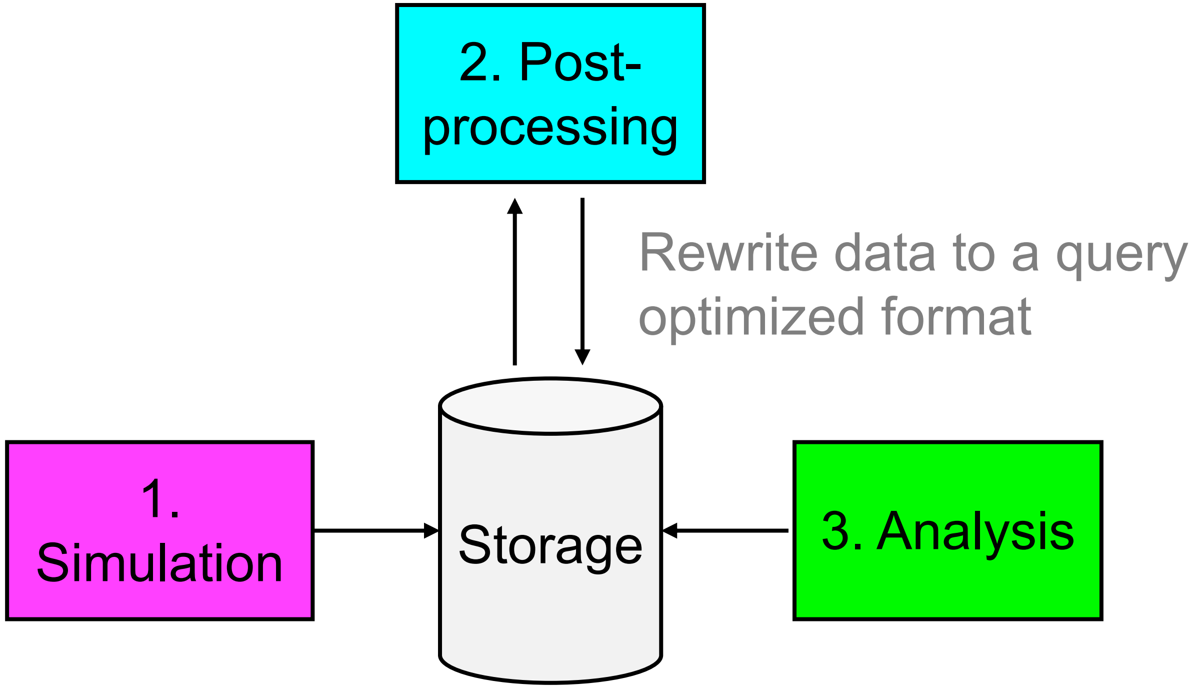
1. **Campaign Storage 2.0**: In-HDD SQL-like query processing (**Seagate**)
2. **KV-CSD**: a H/W accelerated KV store (**SK hynix**)
3. **ABOF**: H/W accelerated ZFS write pipeline (**Eideticom, Aeon, Nvidia, SK hynix**)

Background: HPC Simulation Workflow

A 3-step process: simulation, post-processing (may be skipped), and **analysis**

Perf. maximized when:

- Storage bandwidth fully utilized during data insertion
- Data transfer minimized during analysis (when query selectivity is high)
- Lowest possible data post-processing latency

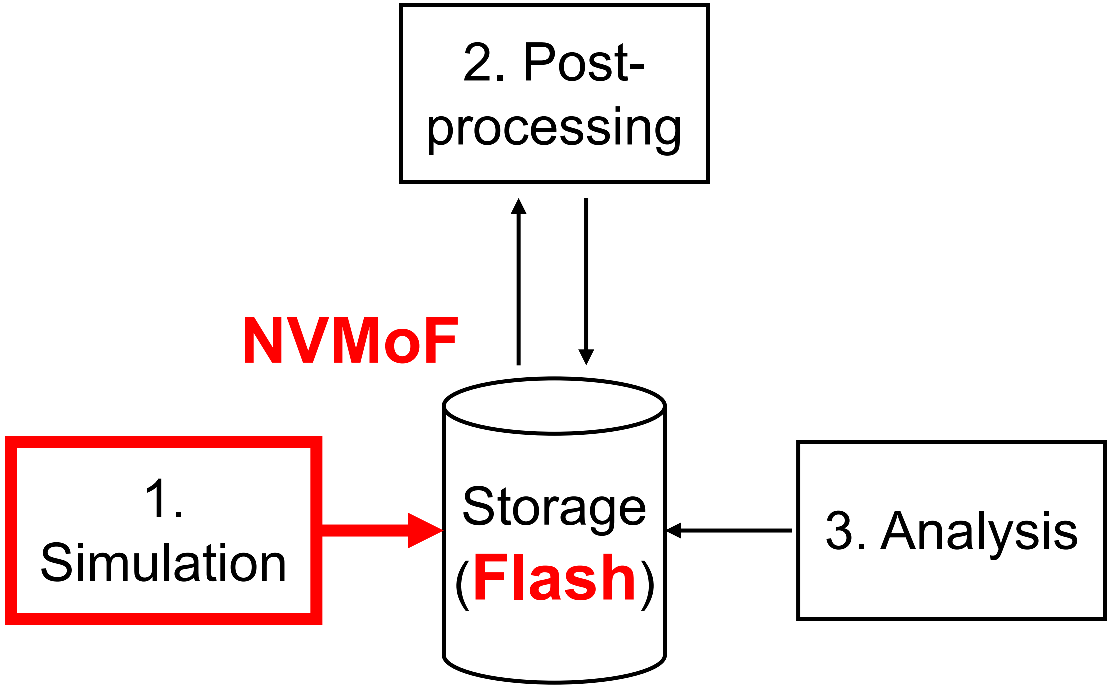


Today's HPC data centers are having problems achieving any of these

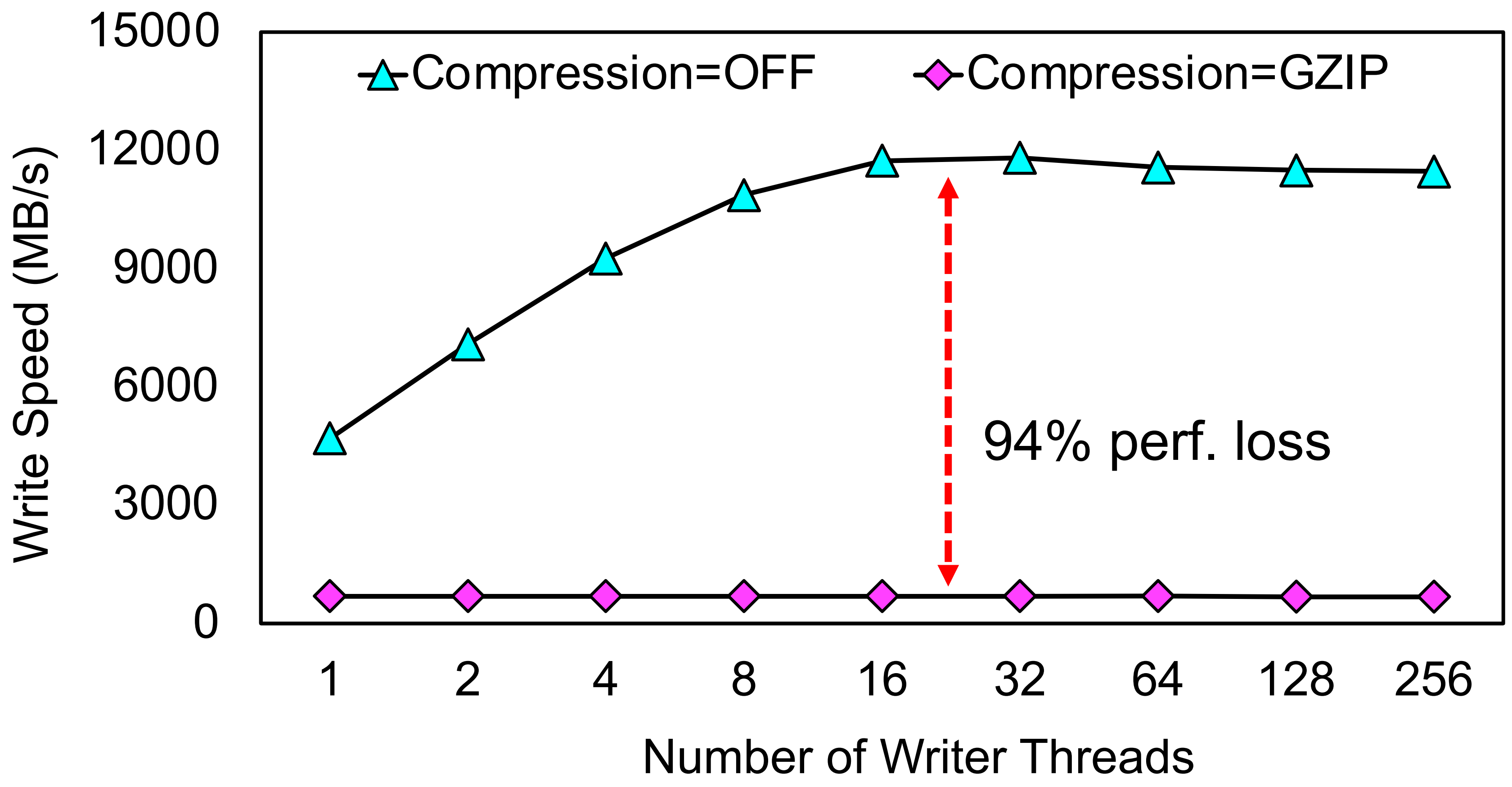
Part I – ABOF: Accelerated Box of Flashes

Problem: Today's host CPU fail to compress data as fast as storage can absorb it

- Compression necessary for frugal use of SSD storage space
- High-entropy scientific data requires heavy compression methods such as gzip
- Host bottlenecks prevent apps from fully utilizing available SSD bandwidth



Up to 94% Perf. Loss Due to Host Bottlenecks

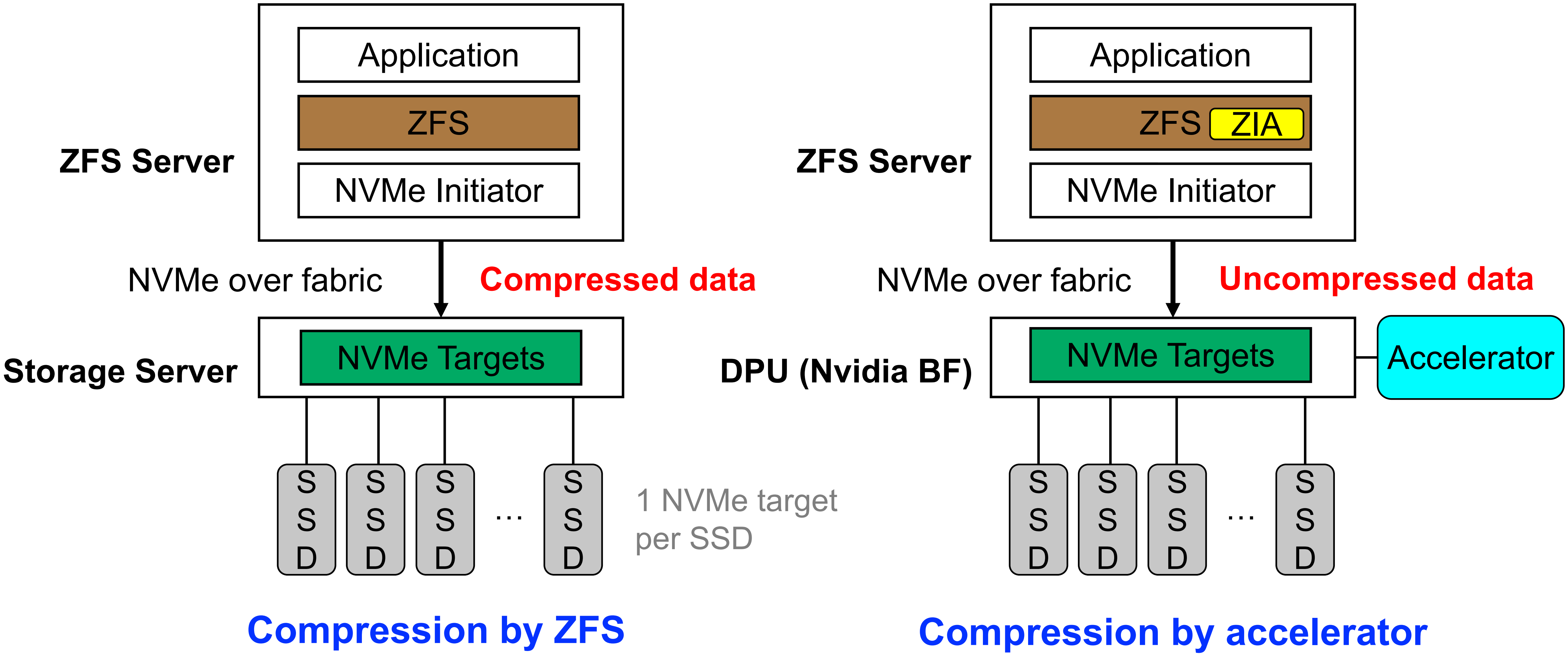


Streaming data into a 10+2 all-flash ZFS RAID pool

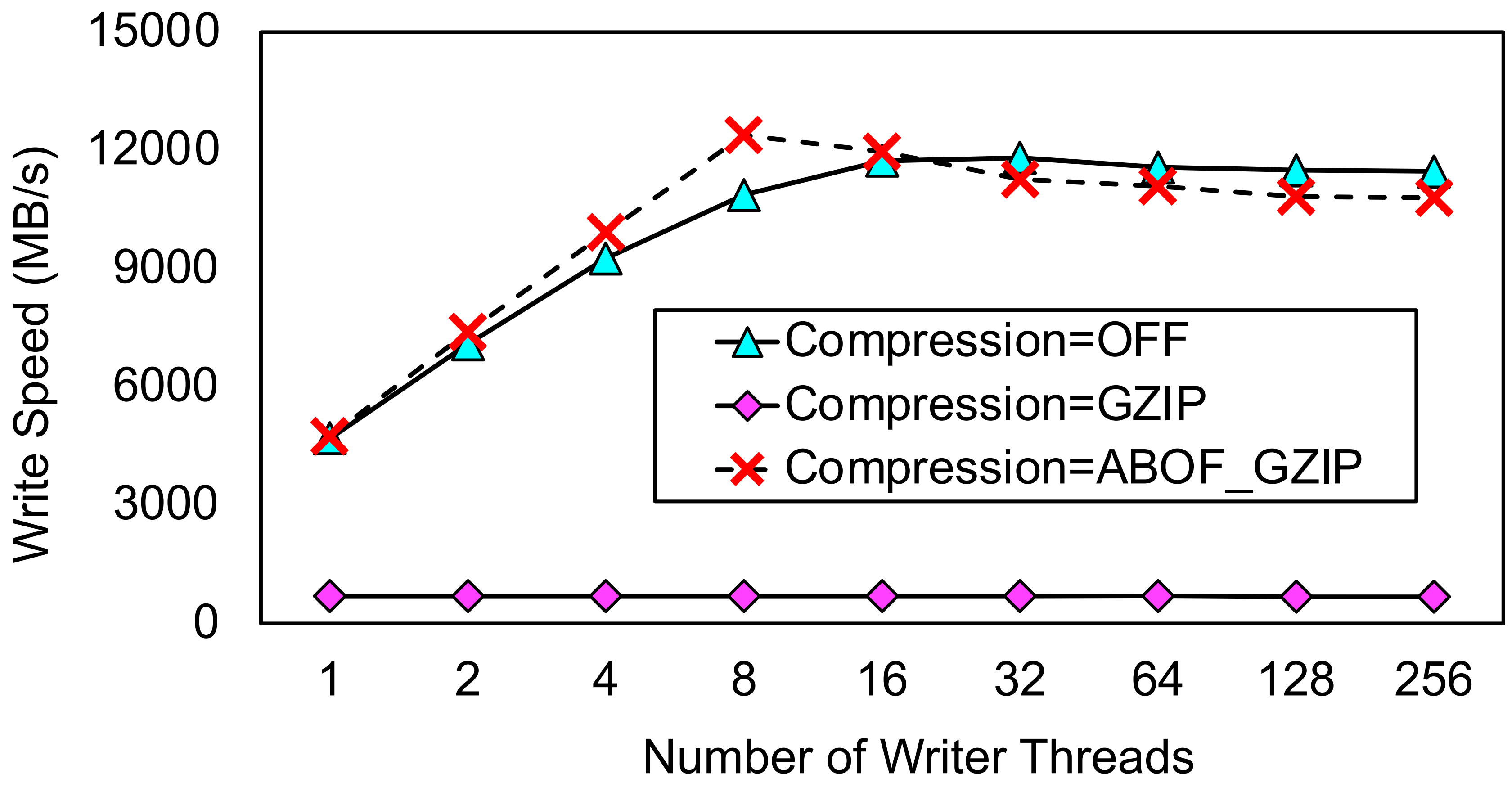
- Concurrent 1MB writes to a single file
- One ZFS host
- 12 NVMeOF flash SSDs

Can we offload compression to storage to bypass host bottlenecks?

ABOF: Towards Accelerated ZFS Writes



Result: 16x Faster Than Host Processing



Streaming data into a 10+2 all-flash ZFS RAID pool

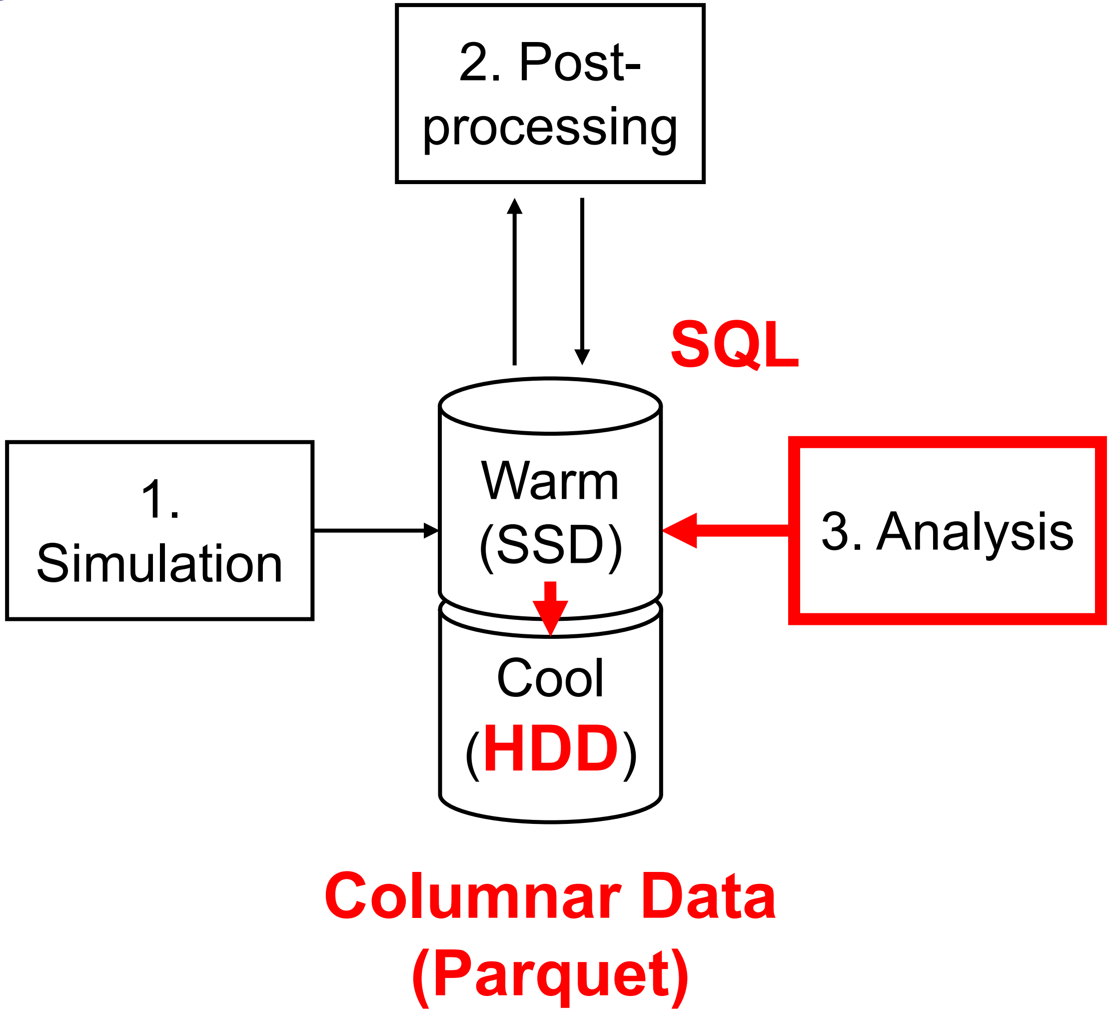
- Concurrent 1MB writes to a single file
- One ZFS host
- 12 NVMeOF flash SSDs

16x faster than host gzip, comparable with no gzip (ABOF gzip is “free”)

Part II – Campaign Storage 2.0

Problem: Scientific analytics increasingly bottlenecked on excessive data transfers

- A query may select only a tiny amount of data from a large dataset
- But a reader program may still have to read back an entire dataset from storage
- Excessive data movements cause long query latency



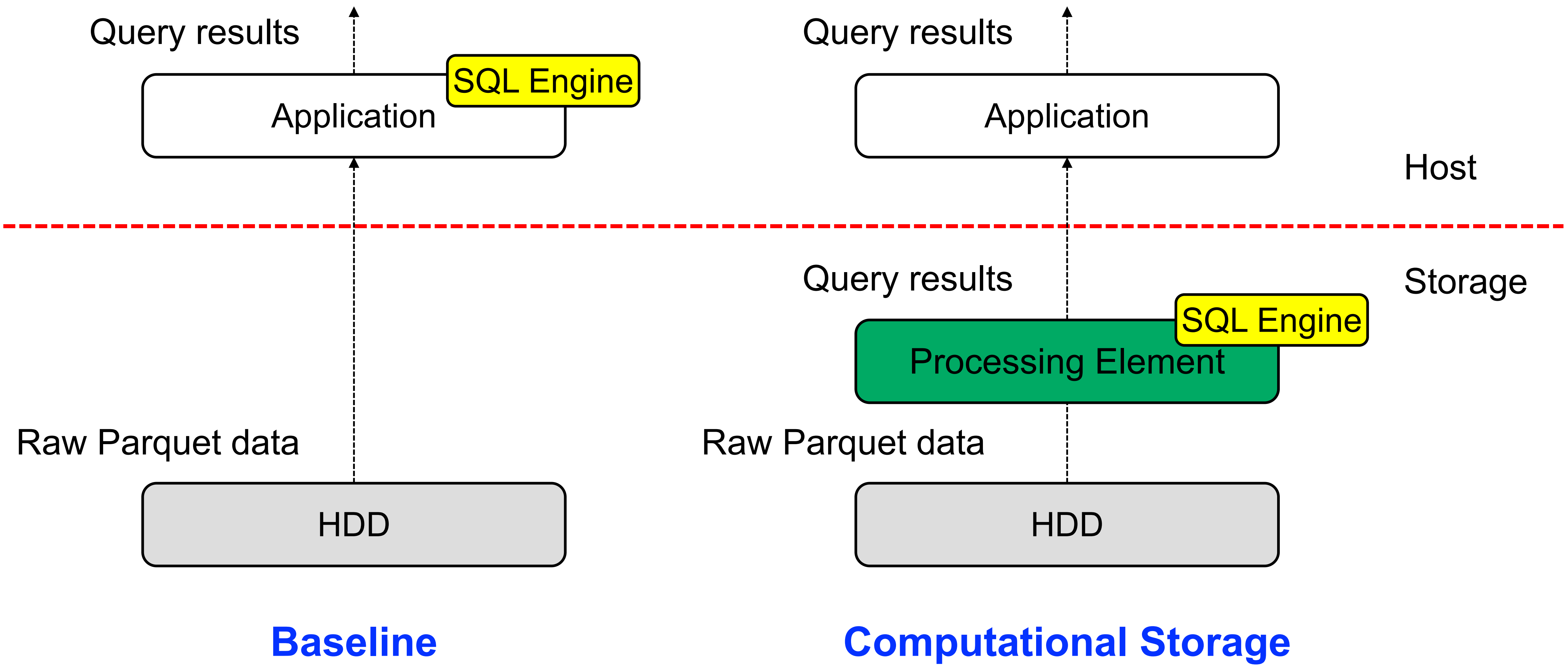
- **User sees even higher latency when data is read from a slow storage tier**

Time to Read 1PB of Data

Burst Buffer	Local Scratch	Campaign Storage	Tape Archive
3.2TB/s	1.2TB/s	100GB/s	10GB/s
312s	**14min**	**2.8hr**	**28hr**

← **Warmer Tiers** | **Cooler Tiers** →

Why Computational Storage Might Help

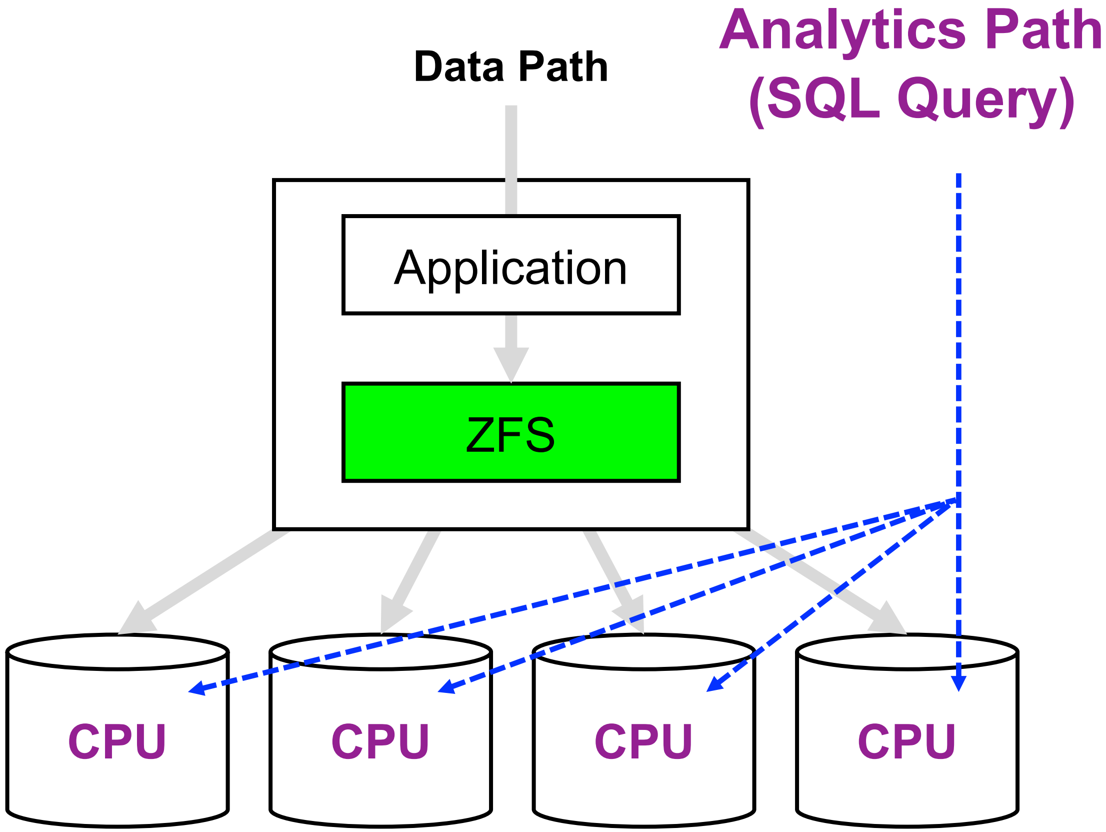


Current Prototype

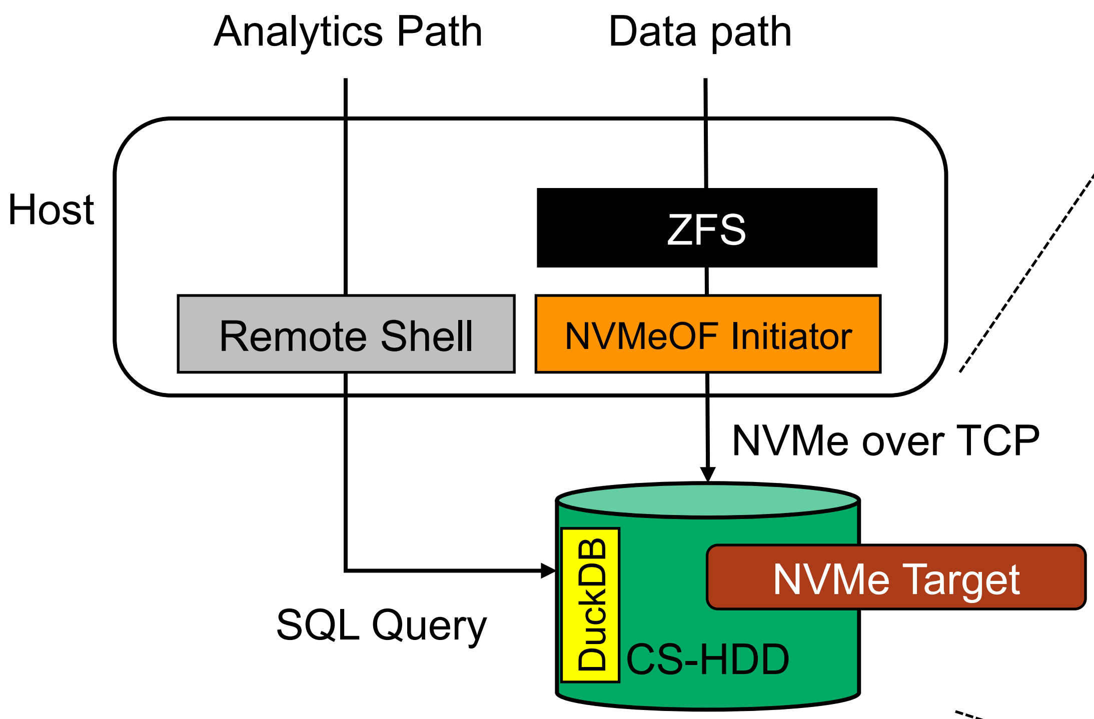
One ZFS host, many HDDs, 1 CPU per HDD

- **HDD:** Seagate Research's Kinetic CS-HDD
  - **CPU**: 2x ARM Cortex-A53 cores
  - **RAM**: 1GB
  - **OS**: Ubuntu Linux
  - **NIC**: 2.5GbE

- **Storage stack:** ZFS
  - **Data protection**: RAID (1, 2, or 3 parities)
- **Analytics software:** DuckDB



A Close Look



Analytics Path
Data path
Host
ZFS
Remote Shell
NVMeOF Initiator
NVMe over TCP
NVMe Target
DuckDB
CS-HDD
SQL Query

SEAGATE
EXOS
ENTERPRISE
16TB
Exos™ X16
ST16000NM001G
SEAGATE SECURE

Two Challenges

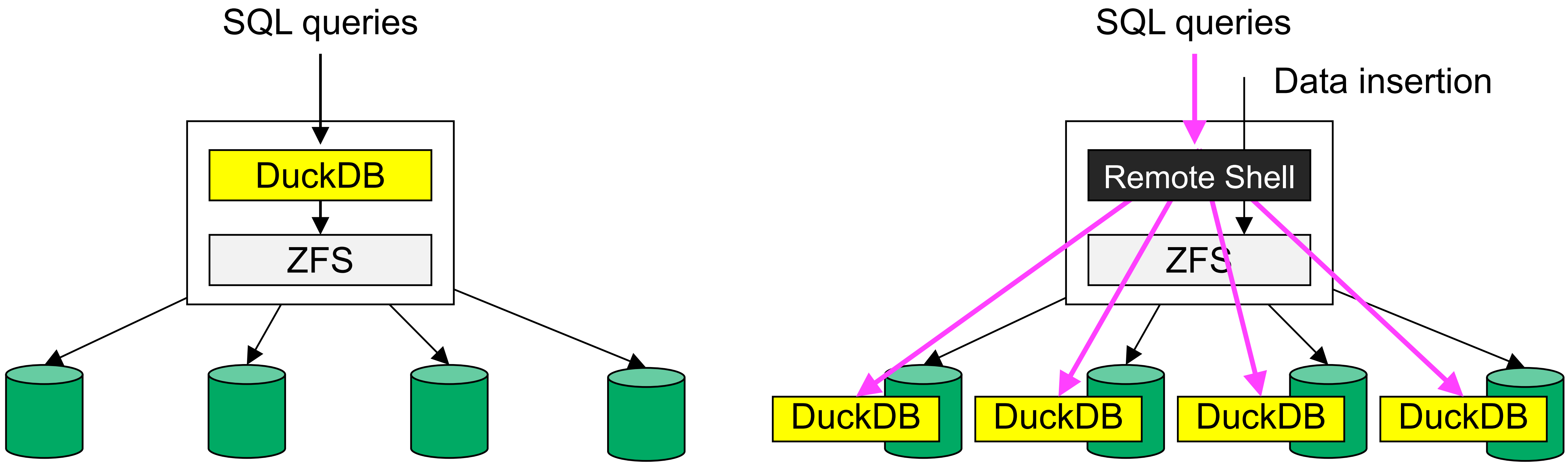
Drives have no knowledge of FS file-to-block mapping

- Solution: LibZDB (allow querying ZFS for mapping information)

A data row may be split over multiple drives

- Data alignment control

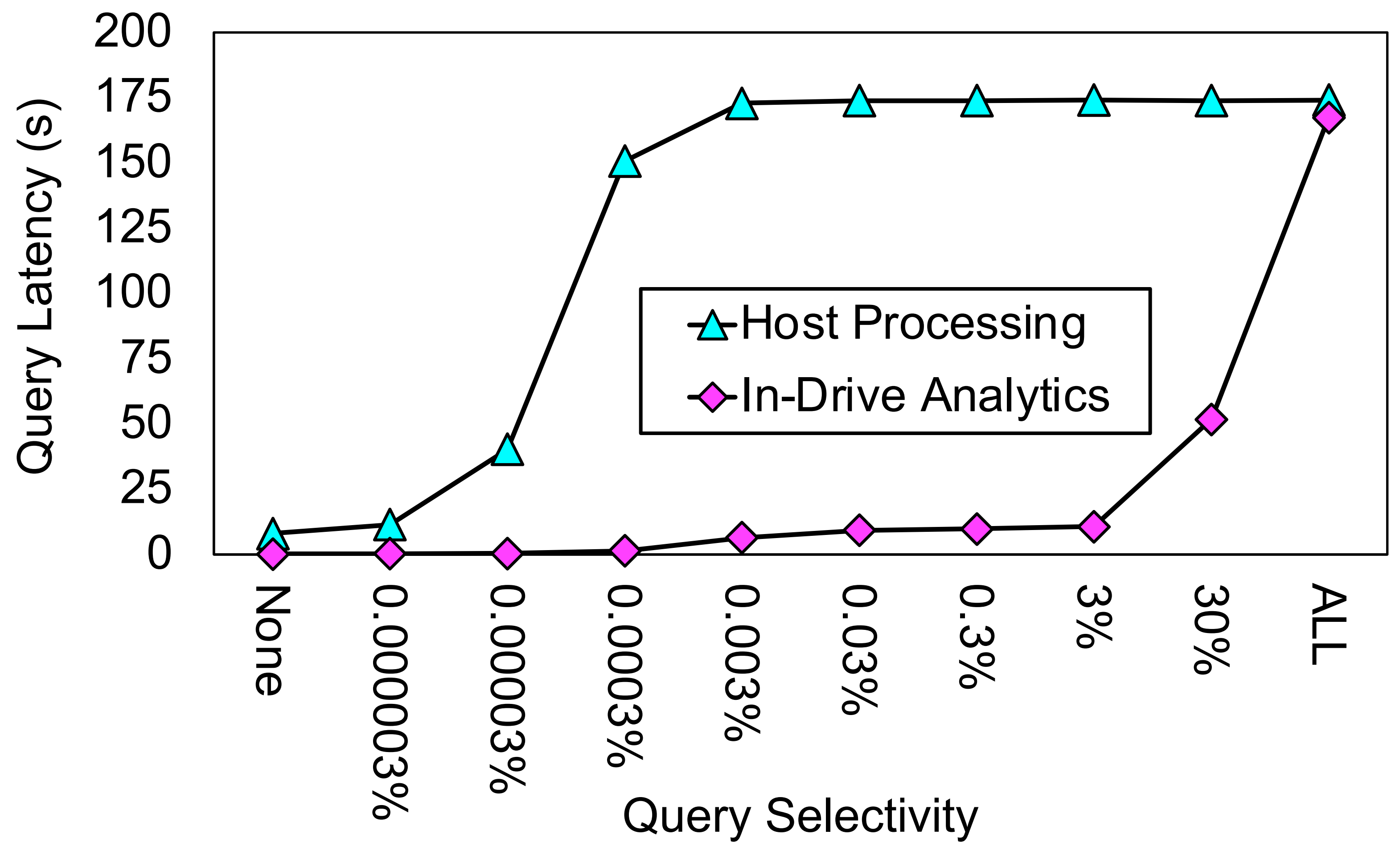
Unaccelerated vs. Accelerated Runs



Host Processing

In-Drive Analytics

Result: In-Drive Analytics Up To 106x Faster



- **1** ZFS host (32 AMD CPU cores)
- **38** CS-HDDs
  - 2x 16+3 RAID Pools
- **50GB** real scientific dataset
  - 2 billion rows
    - Columns: ID, x, y, z, **ke**
- **Query:**
  - SELECT * WHERE **ke>X**

In-drive analytics allows sending only query results over the network

General Scenarios For In-Drive Analytics

A) Host has **network** bottlenecks

Near data compute reduces data movement

B) Host has **CPU** bottlenecks

Near data compute enables parallel processing across smart devices

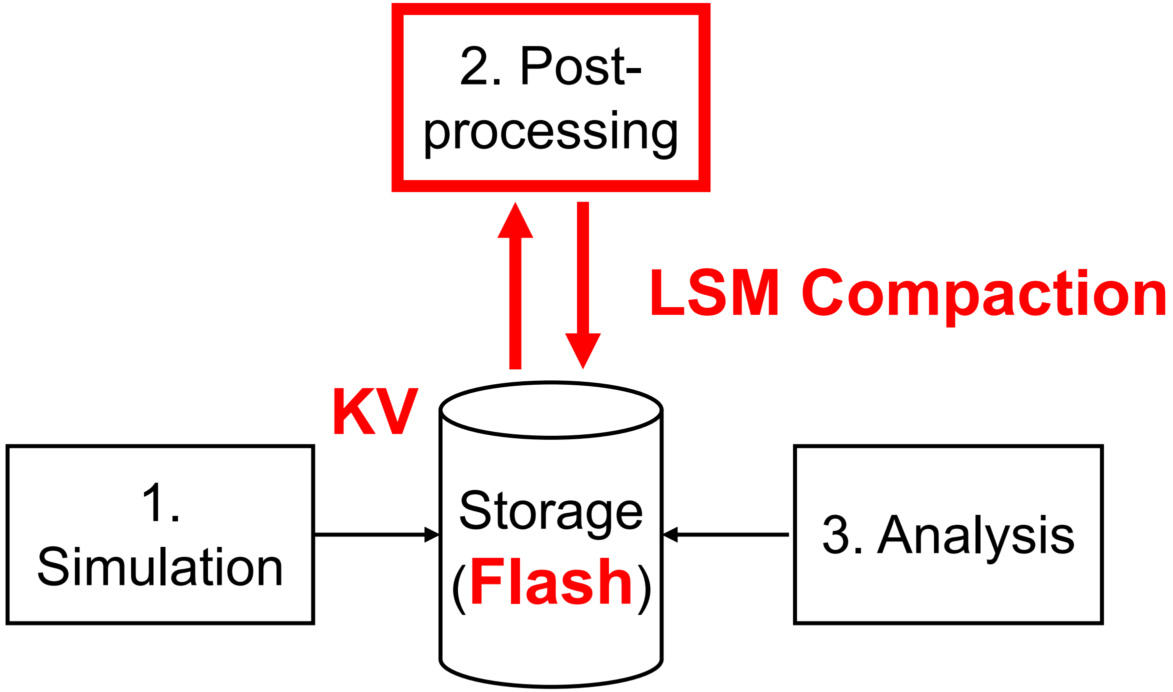
C) Host has **abundant network and CPU**

Near data compute allows for more fully utilizing storage media bandwidth

Part III – KV-CSD: KV Computational Storage Device

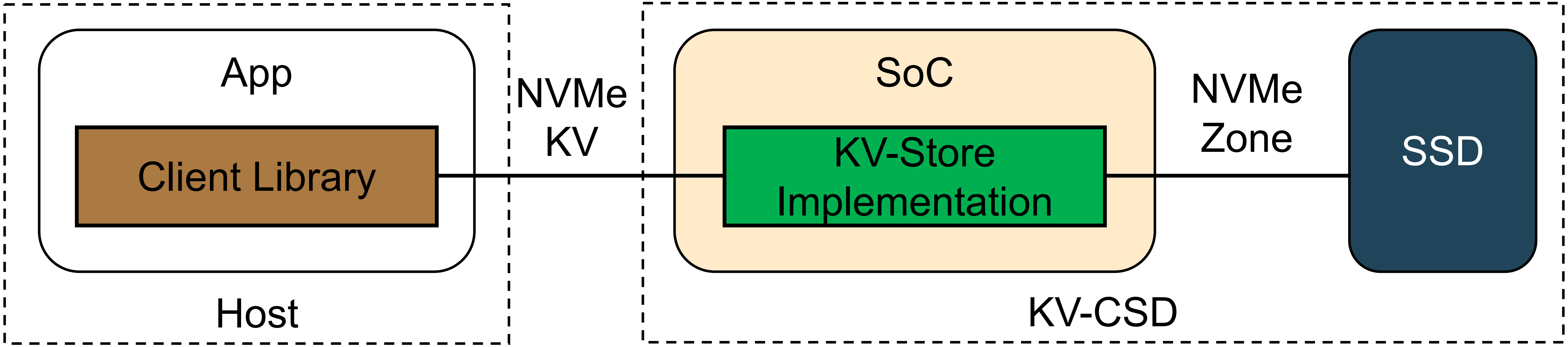
Problem: LSM-Tree based KV stores often experience write stalls due to background compaction ops

- LSM-Tree increasingly popular
- Fast point/range query perf. over primary/secondary index keys thanks to background compaction
- Writes may be blocked when background compaction cannot keep up with foreground insertion

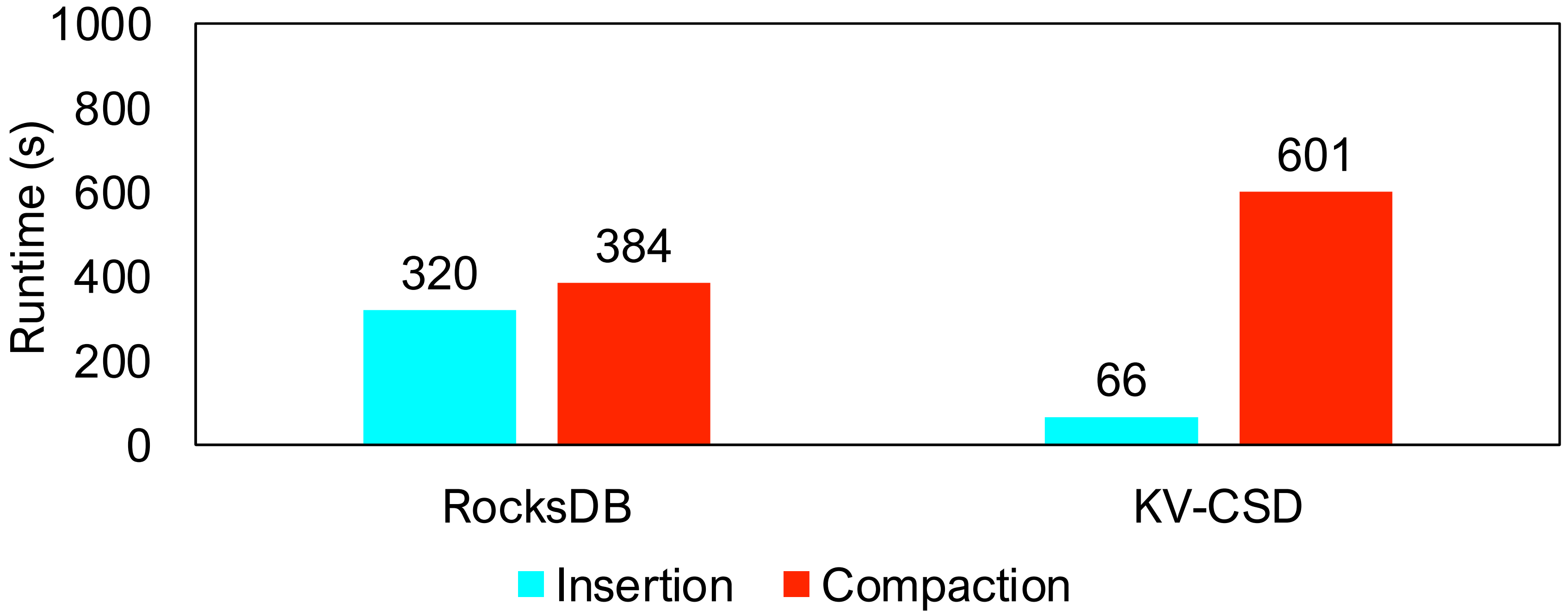


Can we offload compaction to storage?

KV-CSD


App
Client Library
Host
NVMe
KV
SoC
KV-Store
Implementation
NVMe
Zone
SSD
KV-CSD

KV-CSD Better Hides Compaction Latency

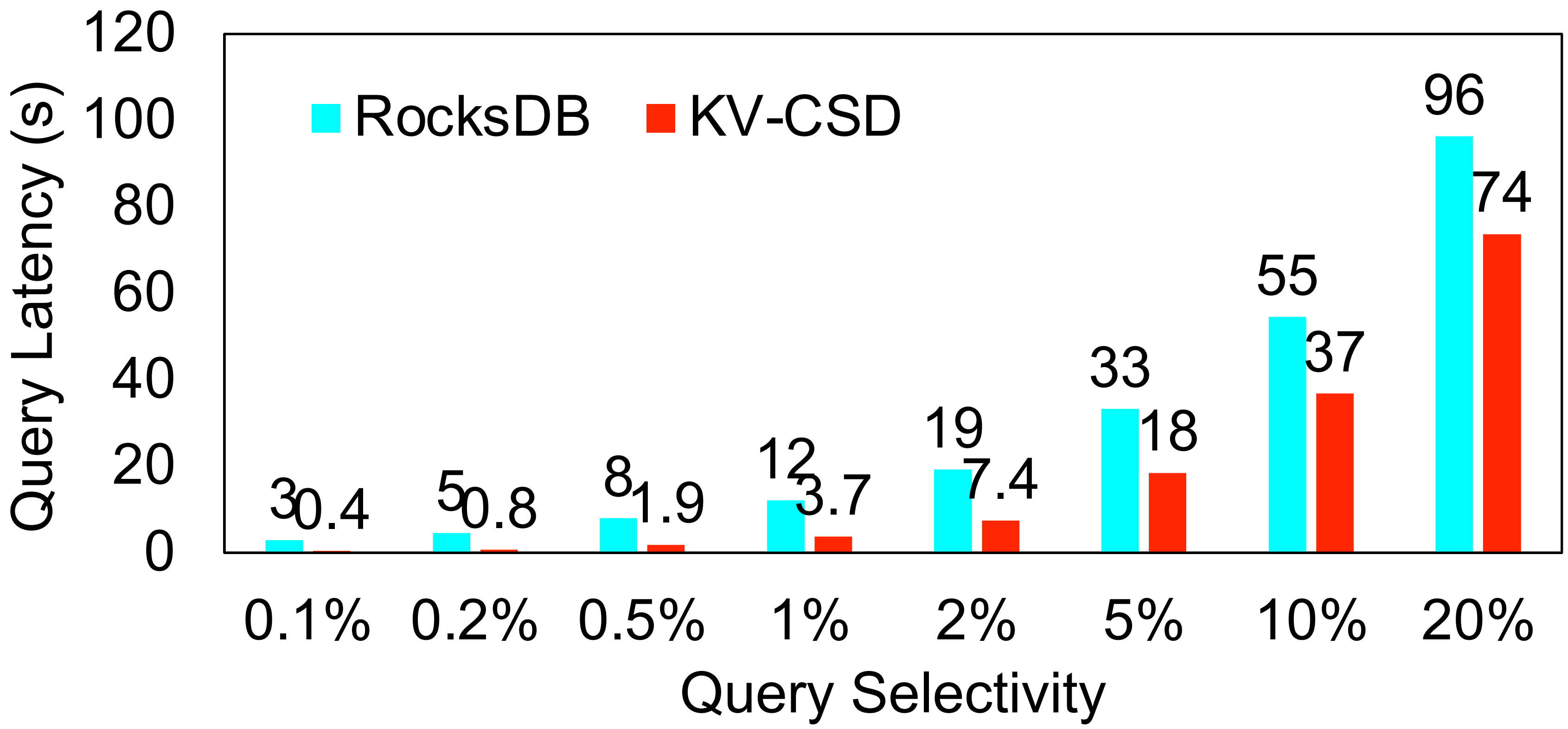


A 256M particle dataset stored as KV pairs

- **Key**: particle ID (16B)
- **Value**: particle payload (32B)

In-drive computation prevents compaction from blocking app writes

KV-CSD Allows Fast Queries to Run Faster



A 256M particle dataset stored as KV pairs

- **Key**: particle ID (16B)
- **Value**: particle payload (32B)
- **Range** query over a secondary index

In-drive KV search allows sending less data to host

Quick Recap

Campaign Storage 2.0	KV-CSD	ABOF
CSD	CSD	CSA
Cool tier	Hot tier	Hot tier
Format aware	Format aware	Format agonistic
Columnar datasets	Row-oriented datasets	Binary data
SQL	KV	FS
Multi-dimensional queries	Single-dimensional queries over primary/secondary indexes	Data capture, ABOF 2.0 will tackle the read path

Conclusion

Large-scale data analytics is a core element of scientific discovery

Computational storage provides new ways of accelerating data-intensive analytics workloads

Preliminary results are promising

More work/collaboration/integration is needed for production deployment