STORAGE DEVELOPER CONFERENCE

SDC 23

BY Developers FOR Developers

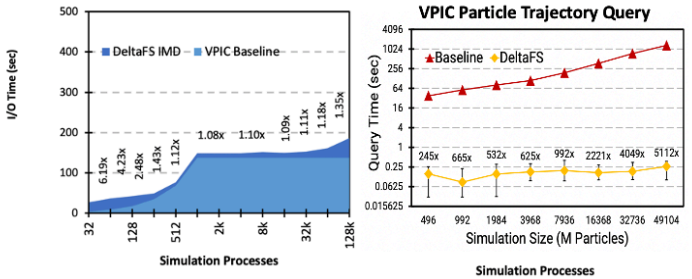
Leveraging Computational Storage for Simulation Science Storage System Design

Dominic Manno

LA-UR-23-29917

Computational Storage Why

- Need flexibility in where computation is done (host, network, device) as economics will change over time
- Its not just energy and time to insight, some of these analytics require the same size analytics footprint as the simulation footprint (petabyte of ram) making analytics not always as feasible
- Data Agnostic Offloads
  - Server memory BW does not allow many passes over streaming data
- Data Aware Offloads
  - Analytics is often multiple orders of magnitude less reading than writing
  - You just have a hard time finding what you are looking for (filter/index/histogram/etc.)
  - Can we add metadata/indexing/ordering to data as it is written with almost no overhead and reap huge wins on read (time, hdwr resources, energy)
  - For Science
    - Particle methods -> “Ordered” row-based analytics (KV)
    - Grid methods -> columnar-based analytics
    - Large Complex Grid methods -> THE KITCHEN SINK



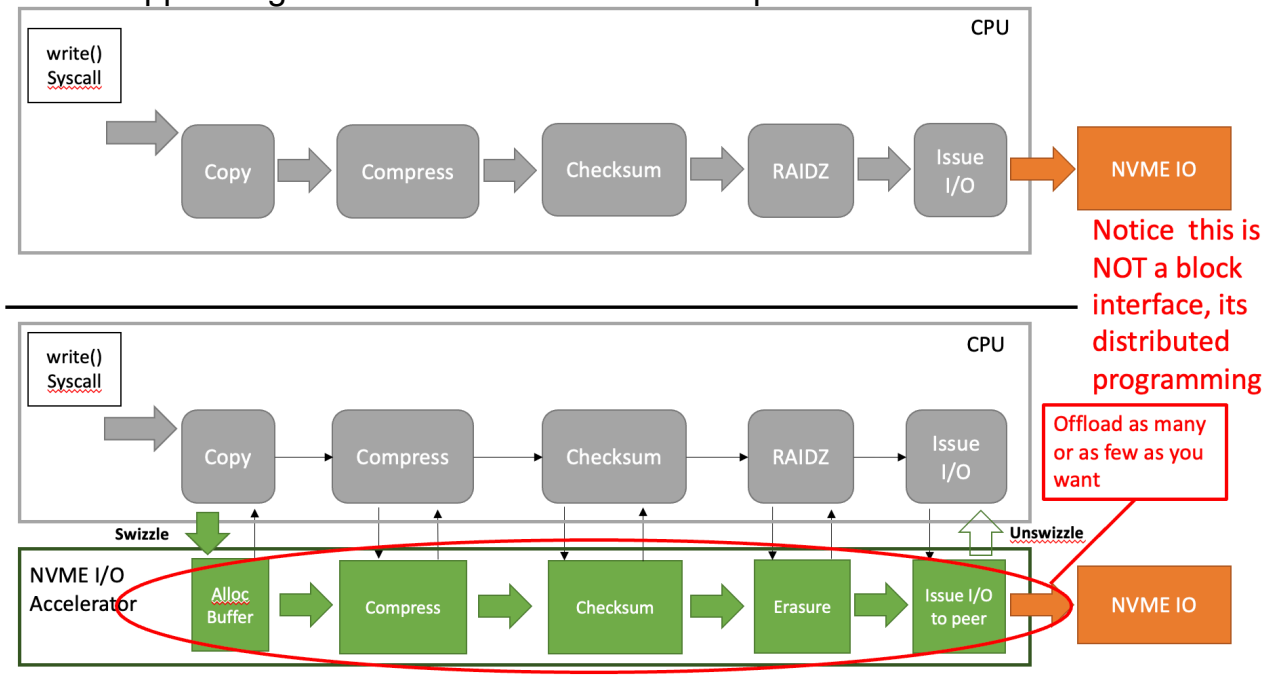
(Papers at PDSW 15, PDSW 17, SC19 (Best Student Paper)

Data Agnostic Offload (ABOF)

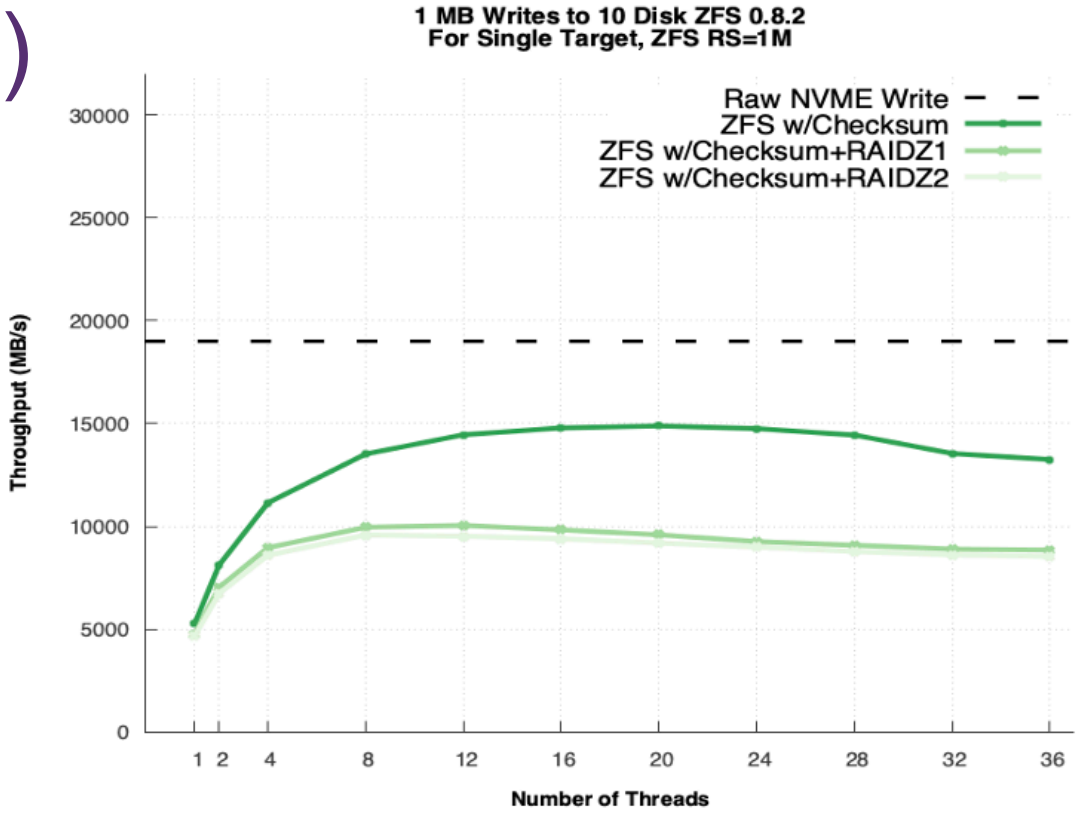
Offload erasure, encoding, compression and make it possible to run each anywhere you like (host, network/DPU, storage/CSA/CSP/CSD)

Consumable:

- Open ZFS
- Kernel module for offloads to register into
- ZFS mods unstreamed
- Tested under Lustre
- Data written is same, upon failure just fall back to host
- No app changes other than faster on less capable host



Notice this is NOT a block interface, its distributed programming



- Not a block interface, a programming interface
- Remote malloc, copy, operate on …

Great partnership

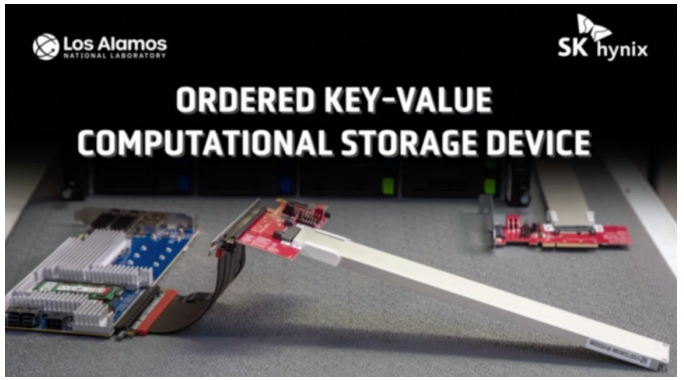
Data Aware Row And Col Based Analytics Offload Trials

Row (Point methods in Science)

- Early KV-CSD work was hashed based but many KV workloads require “order from chaos”
- Leverage LANL/CMU (DeltaFS (Best Student Paper SC19) trace 1K particles in 1T moving in 1 M cells based on LSM/Rocks
- WHY KV-CS: Performance/Energy win. Few sstables have filter matches but also very few records needed per sstable
- Consumable: (Pretty consumable)
  - User facing Rocks API
  - Extension to SNIA NVME-KV interface
  - Could be accelerator under Rocks

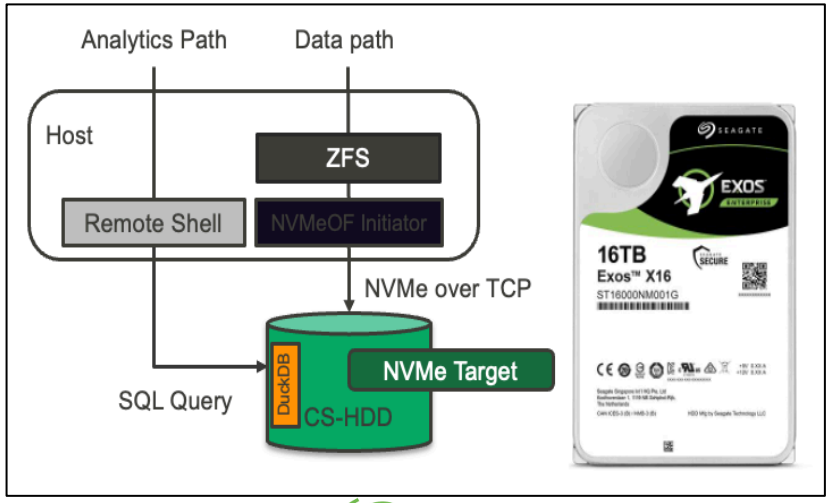
Columnar (Grid methods in Science)

- Leverage
  - Apache ecosystem columnar technology (Parquet/etc.)
  - LANL ZFS knowledge what blocks are part of a parquet file and DuckDB
- Why CSHDD: Performance/Energy win. Many row groups selected (due to AMR) but big variety of number of records needed from those row groups
- Consumable: (Not very consumable - due to inside filesystem implementation)
  - Special knowledge of ZFS file/erasure, something of a layer violation
- Tiny proc/mem performed simple reduction slightly faster than host
- With many drives behind host, scaling beats host by itself (frees host)



Carnegie Mellon University

SK hynix



Analytics Under Erasure CSHDD

SEAGATE

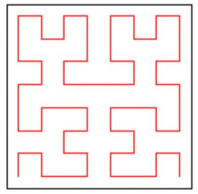
SDC 23

CS data agnostic/data aware learnings

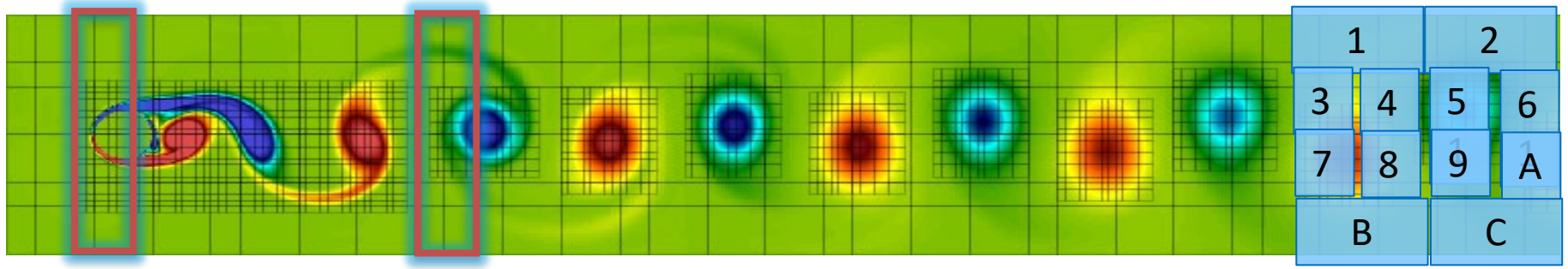
- CSA/CSP's: multi-device/big resource ops (erasure, compression, pipelined functions, etc.)
- CSD's: single-device ops, wins on reductions (requires analytics friendly stripe/erasure)
- Consumption models and broader use cases matter
  - Accelerate under popular server apps: Rocks (rows) / Object Servers (cols) (select on cols/rows)
  - Leveraging Parquet APACHE ARROW Substrait ICEBERG ecosystems seems wise
  - Consumers of large scale data live on one of two basic camps
    - Don't know what you are looking for, want a compact representation (ML training/AI)
    - Know what you are looking for, finding is difficult (due to data size and/or complexity)
      - this is the most obvious place where CS helps and can be implemented in mostly north-south communication patterns
- File systems can be a bit heavy for devices and blocks are useless, perhaps Object is the compromise , with erasure analytics chunks intact on single devices
- At rest compression/erasure seems doable but encryption will be interesting

Why Columnar and why Offload to near Storage?

Columnar? Multi-dimensional Unstructured Adaptive Meshes (grid methods) use distributed arrays/columns



Single process Hilbert order



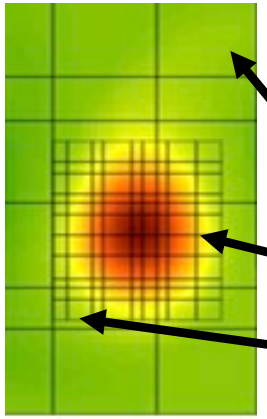
Time Steps 1 2 3 4 5 6

Processes have roughly same number of cells for comp/mem balance but must shuffle cells for AMR

Find outer edge of eddy's (light blue and yellow). Can light weight indexing near storage yield nnnX less data?

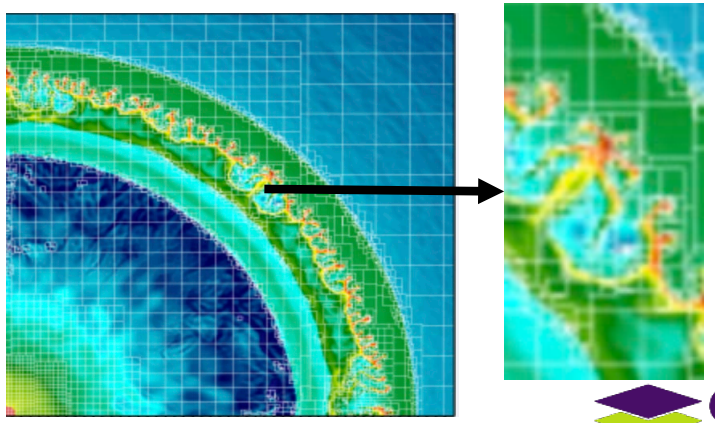
- 1 PB file per time step contains all the state (for restart) (think 1 PB) (and thousands of time step)
- Each cell has 10-100 state variables (64float) (temp, pressure, energy, momentum, differentials (for gradients)…)
- In 3D – applications view this as 10-100 distributed arrays (COLUMNS), serialize in Hilbert space filling curve order

Offload? Many selected row groups will yield only some rows and most queries will exclude many columns



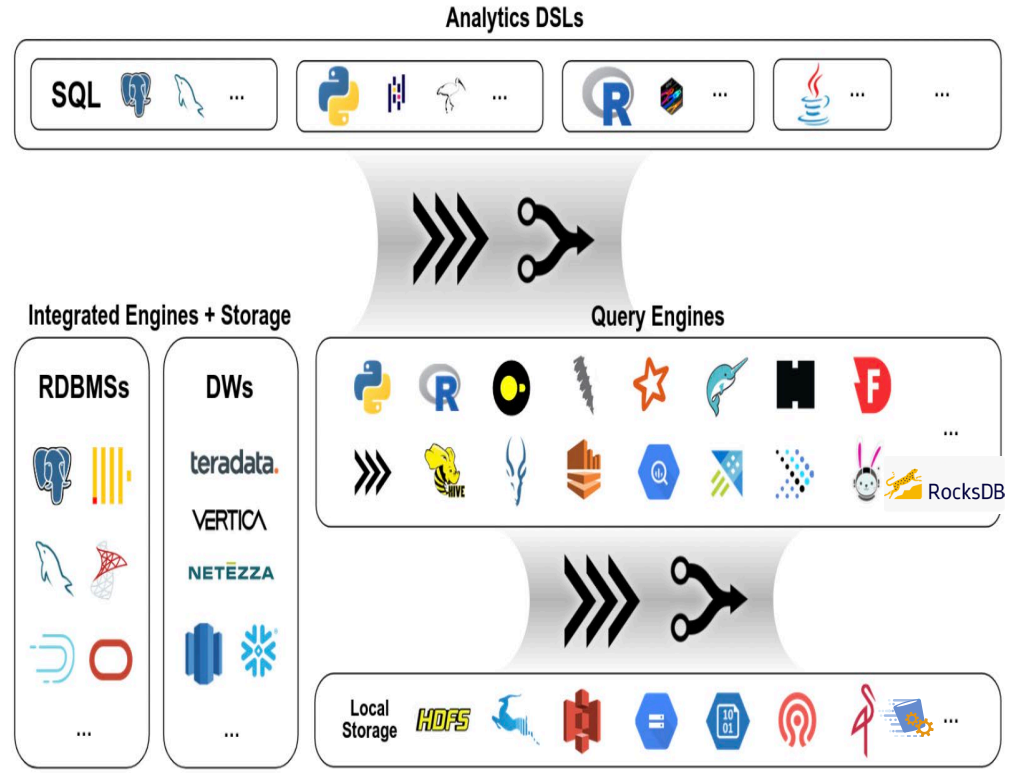
- If you need all row groups, or all rows of groups, or all columns of groups – why offload?

  We don't
  - Reduce trivially by excluding entire row groups
  - Need entire row group (all rows (subset of column)s
  - Need only some rows of the row group



Its never that simple

A Columnar end-to-end demo with Object CS Offloads



Courtesy Voltron Data

DEMO Stack

SQL ...

ADBC ODBC JDBC ... Custom Connector

presto Apache Spark

S3 plus RPC pushdown service (more complex SQL and Query over more than one object (bucket/objects list))

Open Src object server with S3 + RPC pushdown, use OCS backend

OCS api/OCS on NVME

OCSA | OCSD | OCSD

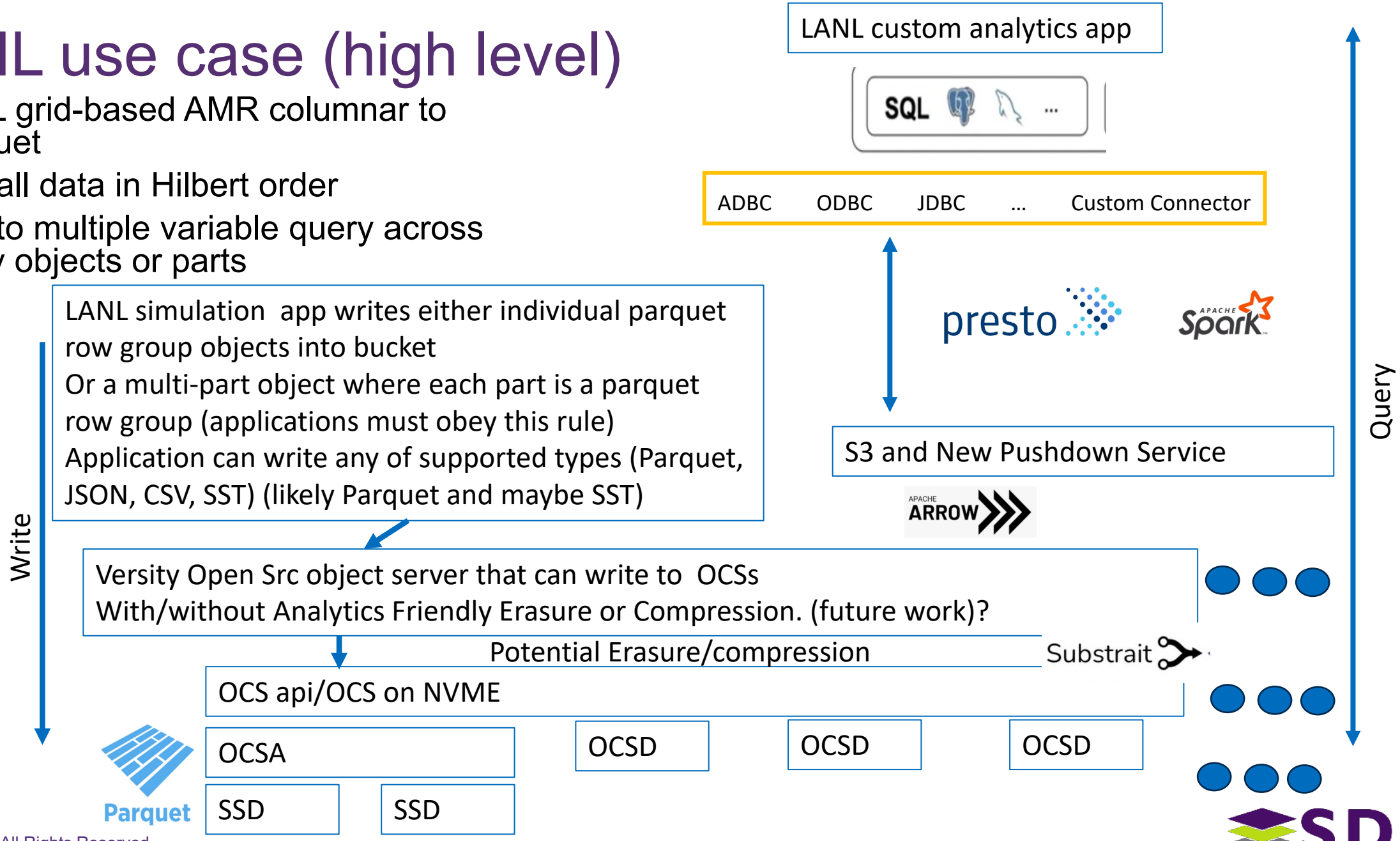
SSD | SSD

- Object enables same view of "analytic chunk" everywhere (App, obj server, CSA and CSD. (**object not block!**)
- Object Future proofs CS: if CXL becomes how to address smart storage, offload will be based on memory objects, and if file everywhere ever wins, files and objects are close cousins.
- Assists with reality that economics will insist on north/south bw devices

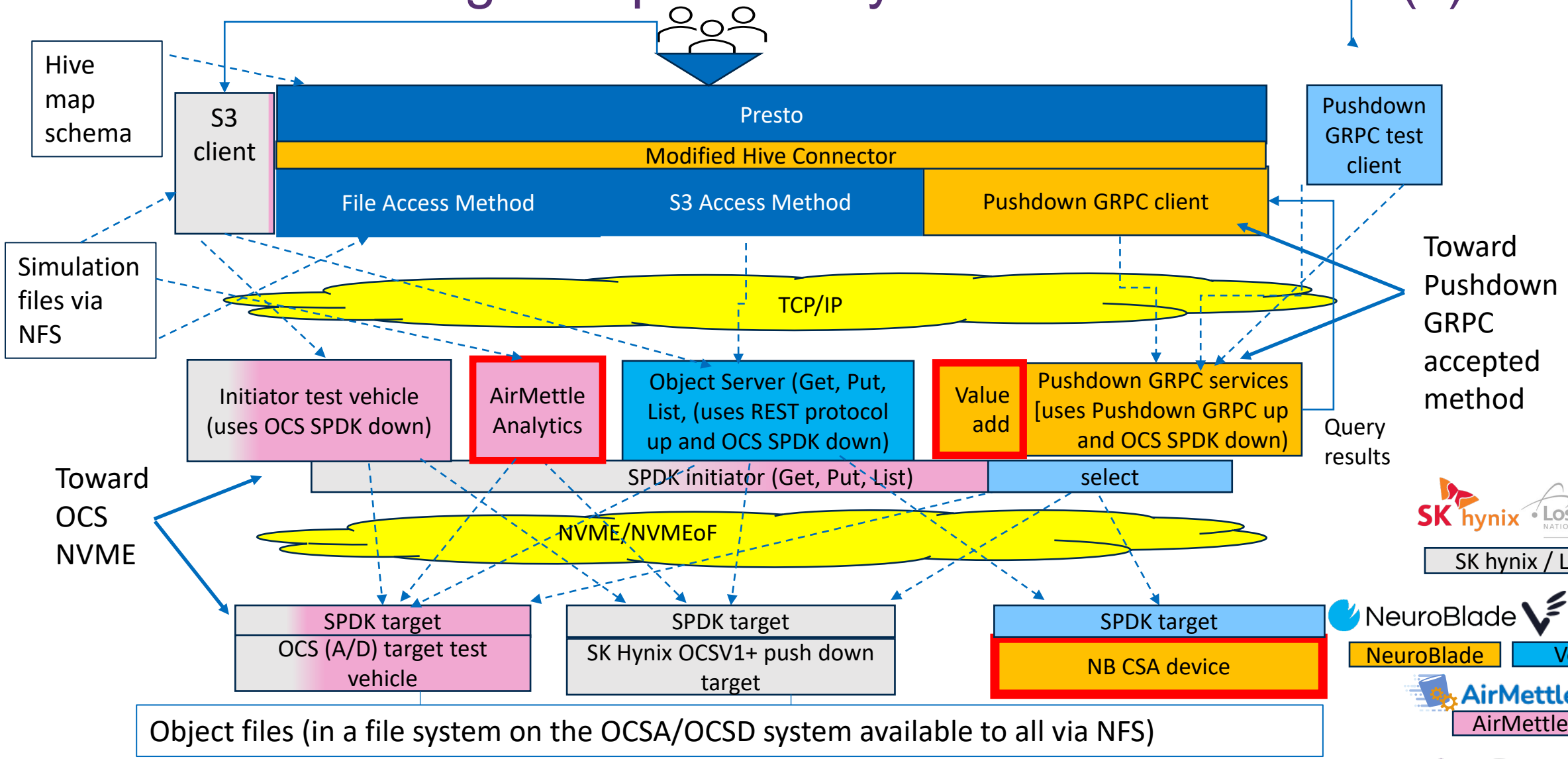
SDC 23

LANL use case (high level)

- LANL grid-based AMR columnar to Parquet
- Overall data in Hilbert order
- One to multiple variable query across many objects or parts



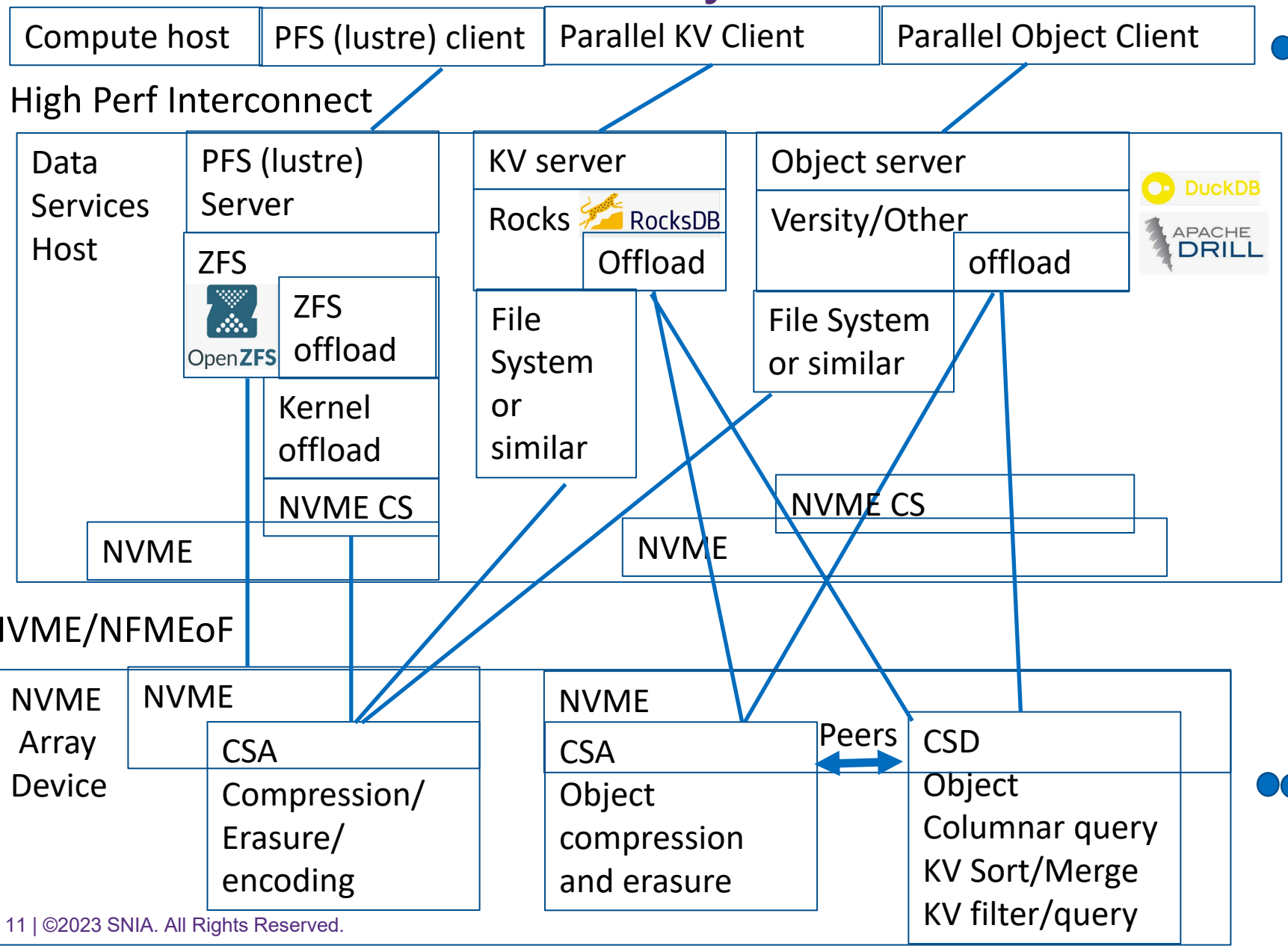
OCS Initiators/Targets Open Ecosystem Demonstration(s)



Partnering has been the key to this exploration!

SDC 23

What does this notionally look like?



- Support both row and columnar (multiple types of support (parquet, csv, json, sst, …)
- Leverage S3 for inspiration
- Leverage Apache Analytics Ecosystem
- Leverage as much of SNIA NVME KV interface as possible
- Open ecosystem for CS

Thanks for your time!

Ultra-Scale Systems Research Center

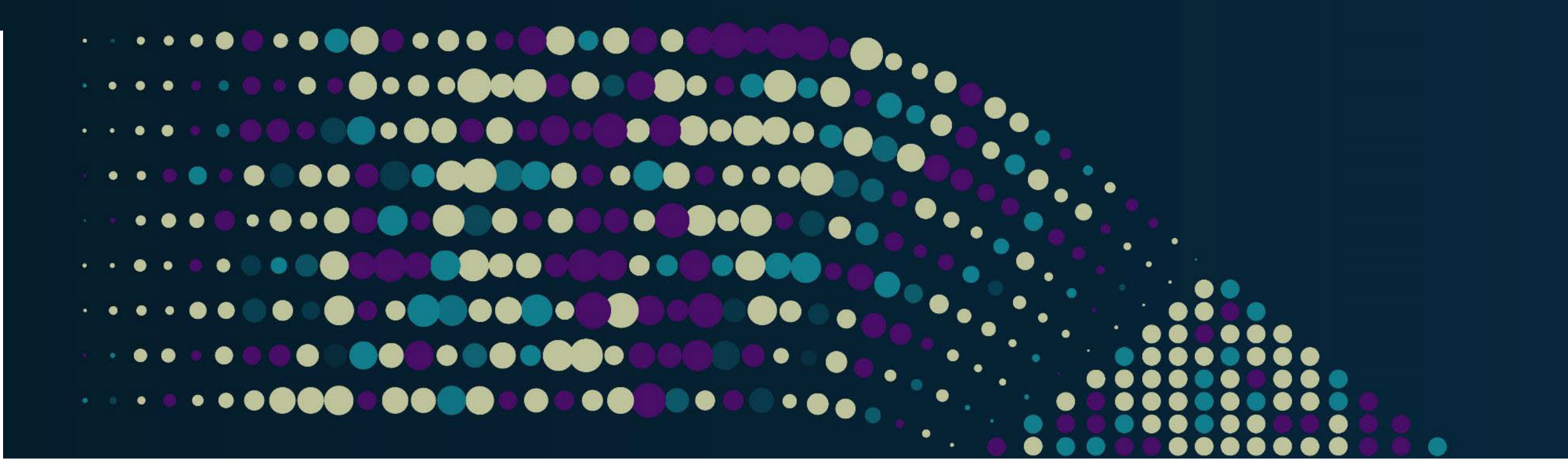
Los Alamos NATIONAL LABORATORY EST. 1943

NAS

HIGH PERFORMANCE COMPUTING

The Efficient Mission Centric Computing Consortium

SDC 23



Please take a moment to rate this session.

Your feedback is important to us.

