SNIA COMPUTE + MEMORY + STORAGE SUMMIT

Architectures, Solutions, and Community
VIRTUAL EVENT, APRIL 11-12, 2023

A NVMe Computational Storage Array for High Performance Data Analytics

Stephen Bates, PhD
VP and Chief Architect
Emerging Storage Systems, Huawei.

A NVMe CSA for High Performance Data Analytics

- The anatomy of an NVMe Computational Storage Array (CSA).
- Using the NVMe Key Value command set for Computational Storage
- High Performance Data Analytics example

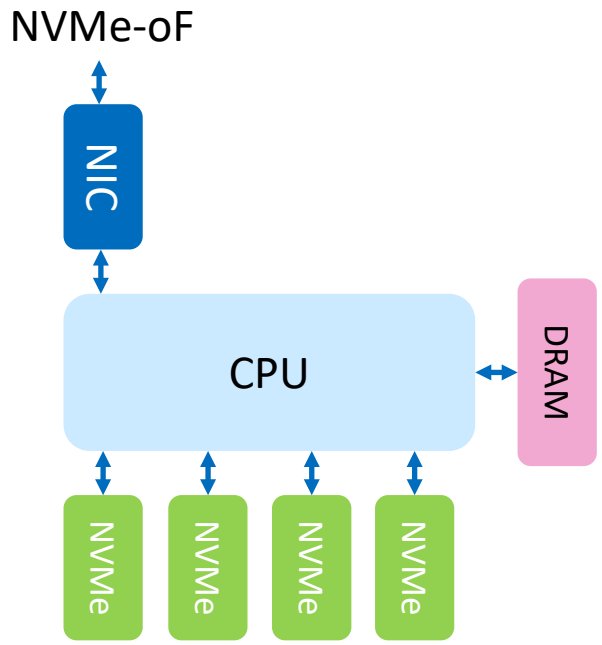
 © SNIA. All Rights Reserved.

- The Anatomy of a Computational Storage Array

 © SNIA. All Rights Reserved.

The anatomy of an NVMe Computational Storage Array

- An NVMe-based CSA has several key properties:
  - Exposes the resources of an NVMe subsystem (storage and compute) via NVMe-oF.
  - Can advertise existing, emerging and vendor-specific command sets and namespace types.
  - Allows many hosts to connect to the subsystem via controllers.

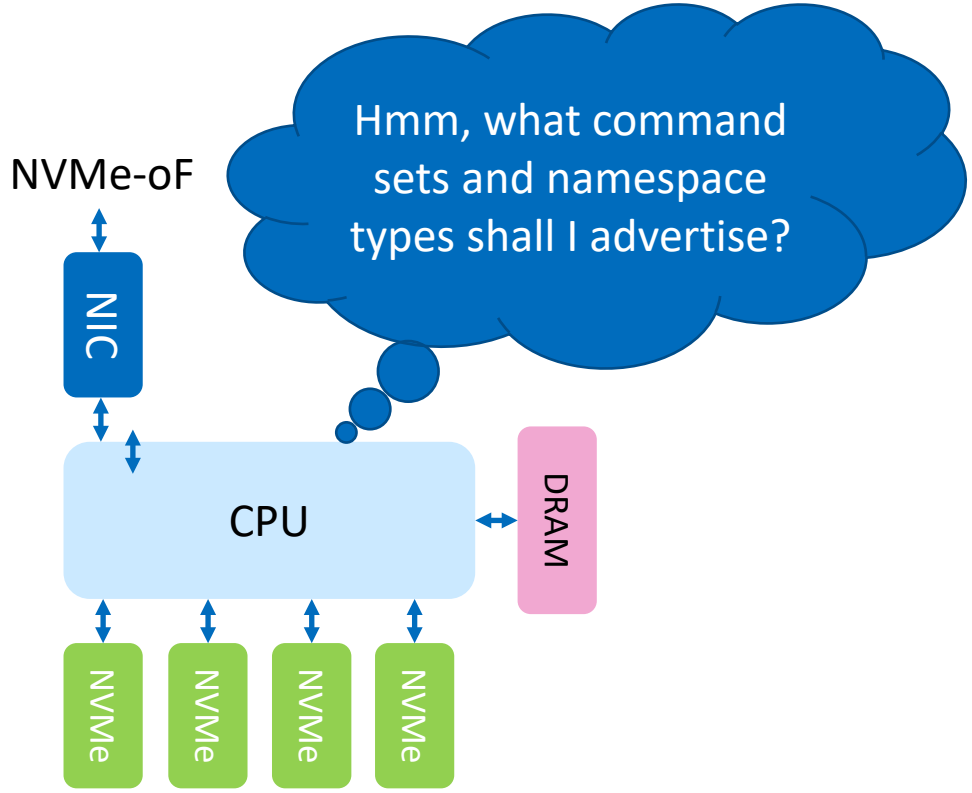


One example of hardware that can present to remote hosts as an NVMe-oF subsystem.

 © SNIA. All Rights Reserved.

The anatomy of an NVMe Computational Storage Array

- An NVMe-based CSA has several key properties:
  - Exposes the resources of an NVMe subsystem (storage and compute) via NVMe-oF.
  - Can advertise existing, emerging and vendor-specific command sets and namespace types.
  - Allows many hosts to connect to the controller via controllers.

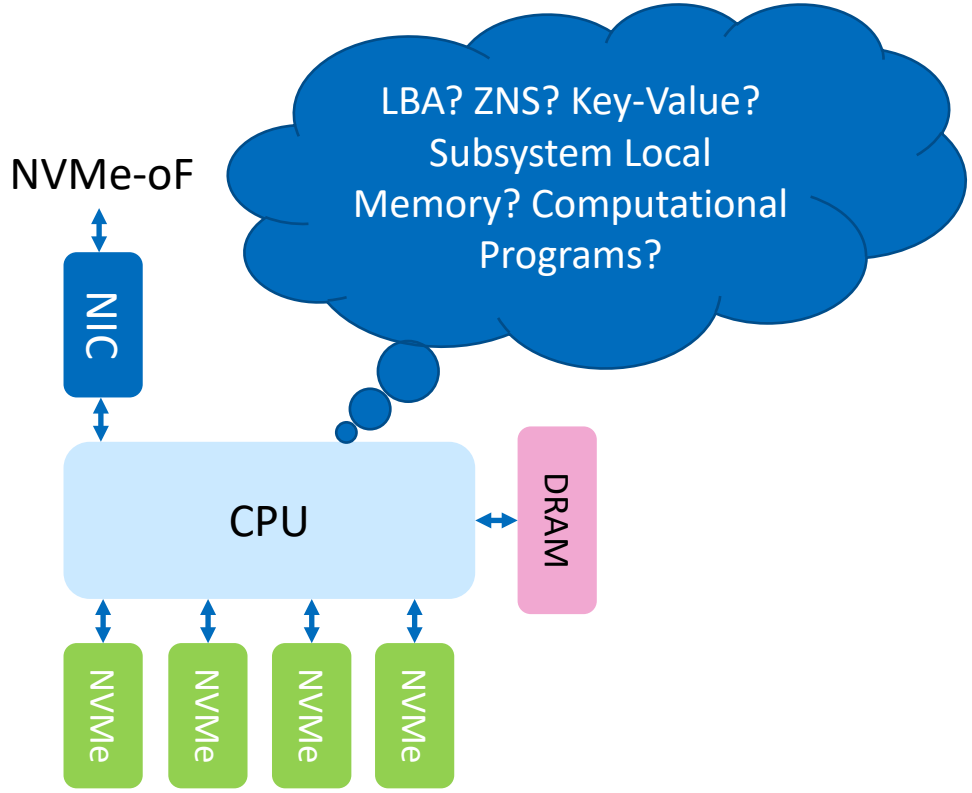


One example of hardware that can present to remote hosts as an NVMe-oF subsystem.

 © SNIA. All Rights Reserved.

The anatomy of an NVMe Computational Storage Array

- Most NVMe-oF arrays today only expose LBA-based namespaces.
- This will change in the future as new command sets emerge.
- One interesting command set we can expose over fabrics is NVMe Key Value. Turns our array into an object store!

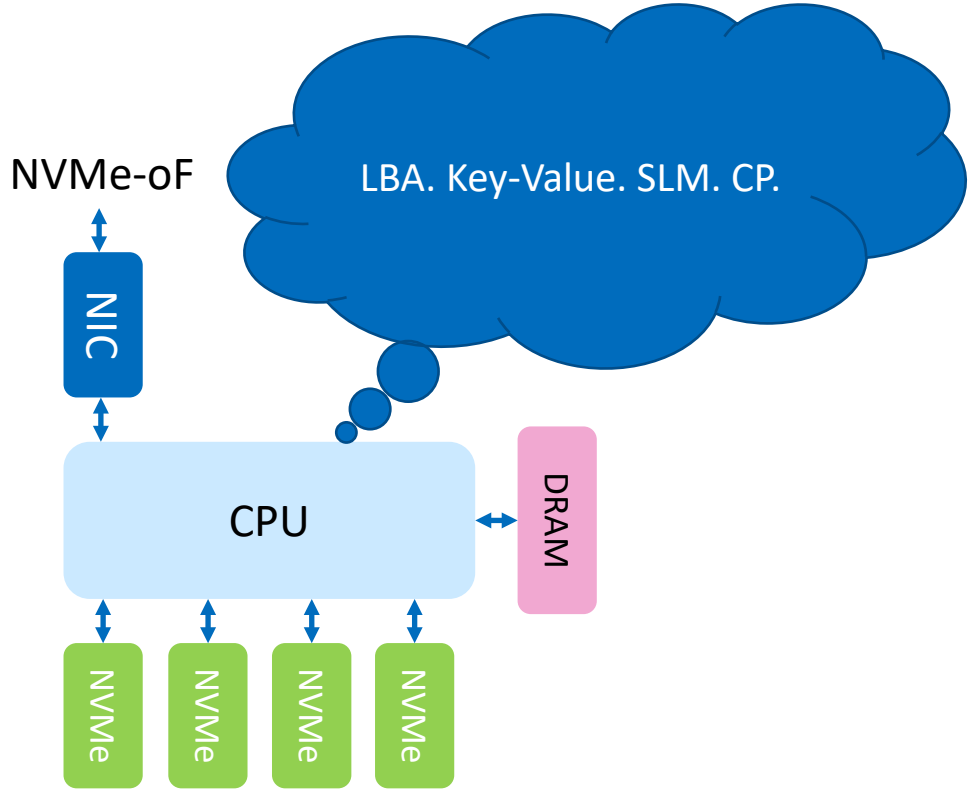


One example of hardware that can present to hosts as an NVMe-oF subsystem.

 © SNIA. All Rights Reserved.

The anatomy of an NVMe Computational Storage Array

- In the CSA we shall consider today we will expose three command sets/namespace types.
  - Logical Block Address (LBA)
  - Key-Value (KV)
  - Subsystem Local Memory (SLM)
  - Computational Programs (CP)
- LBA, KV and SLM namespaces will store data. CP will process data.



One example of hardware that can present to hosts as an NVMe-oF subsystem.

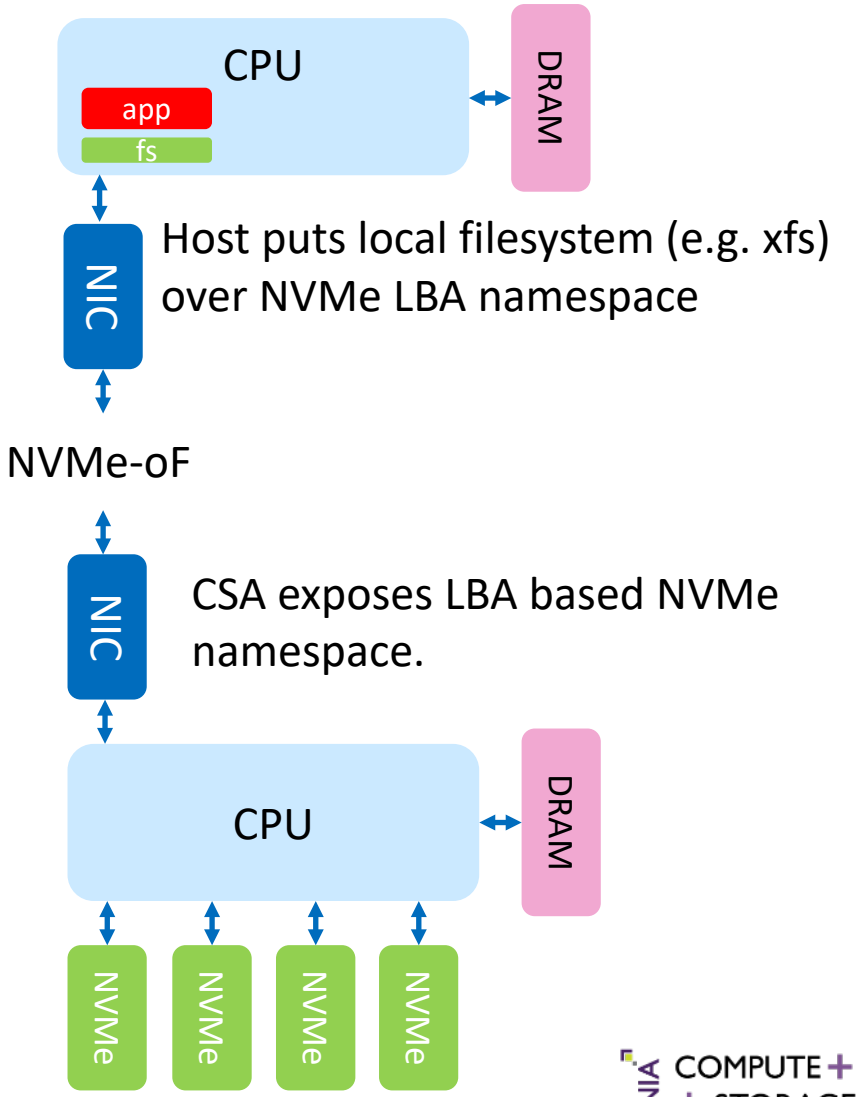
 © SNIA. All Rights Reserved.

- Using the NVMe Key Value command set for Computational Storage

 © SNIA. All Rights Reserved.

Using the NVMe Key Value command set for Computational Storage

- When we use NVMe LBA-based namespaces for computational storage we have one big challenge.
- The application on the host is (almost certainly) wanting to do compute on files but the CSA only understands LBAs!
- Often there is a host-local filesystem between the application and the CSA!

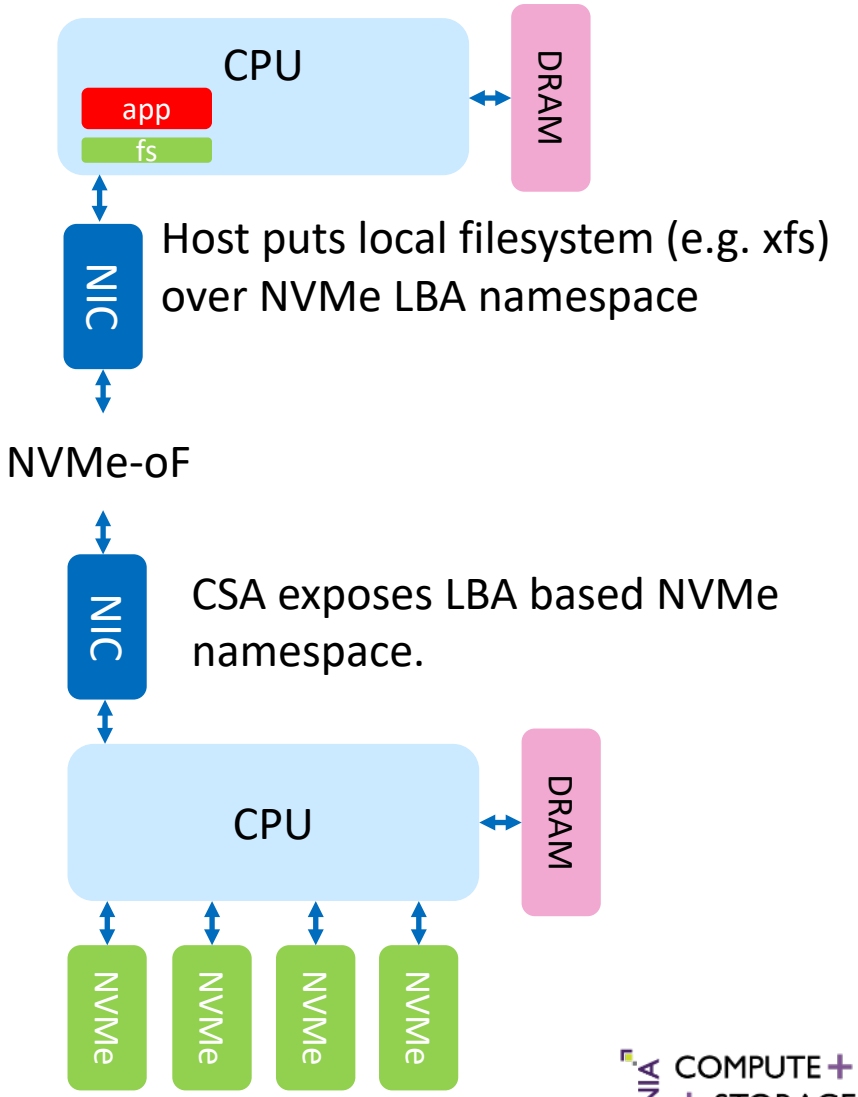


 © SNIA. All Rights Reserved.

Using the NVMe Key Value command set for Computational Storage

```
df = pd.read_parquet("data.parquet")
print(duckdb.query("SELECT * FROM df WHERE
high = (SELECT max(high) FROM df)").to_df())
```

- As an example this simple Python snippet reads a parquet file off the NVMe-oF namespace (with filesystem on top) and then finds (and prints) the row with the maximum in column "high".
- If we wanted to offload this snippet to the CSA how do we tell the array what LBAs the data.parquet file resides in? The CSA knows nothing about the local filesystem. This is a problem!
- By moving to a KV interface the mapping between objects is consistent for both the app and the CSA!

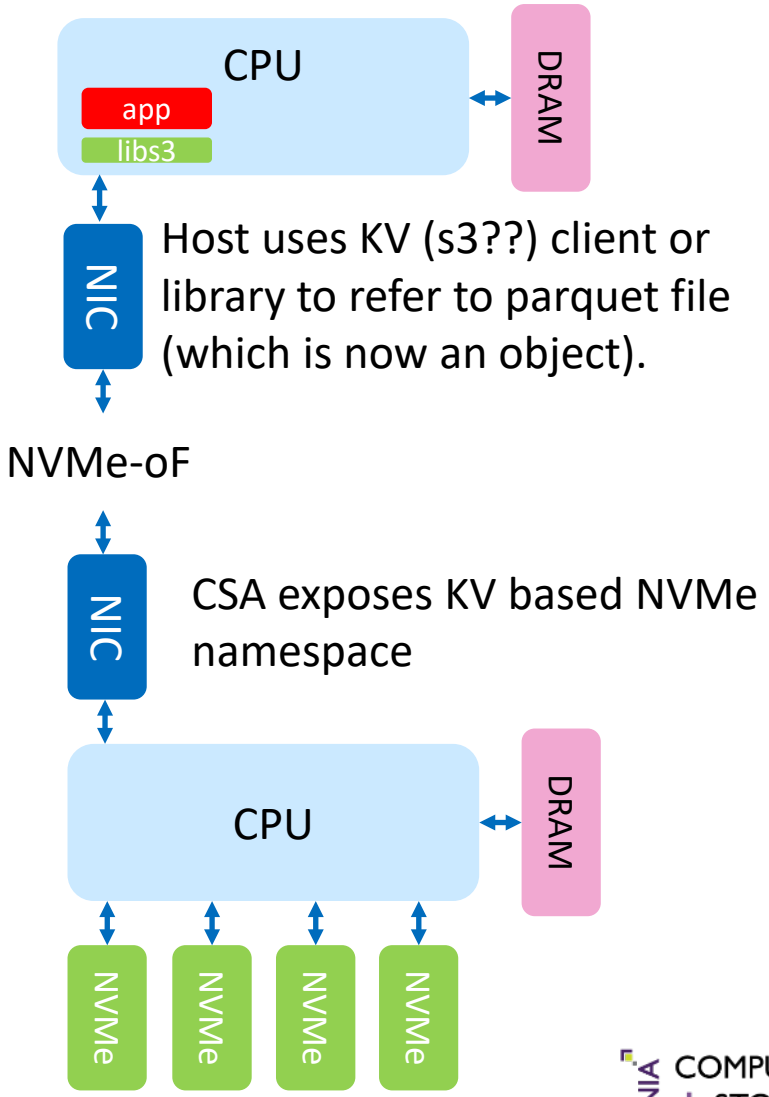


 © SNIA. All Rights Reserved.

Using the NVMe Key Value command set for Computational Storage

```
obj = s3.get(hash("data.parquet"))
df = pd.read_parquet("data.parquet)
print(duckdb.query("SELECT * FROM df WHERE high =
(SELECT max(high) FROM df)").to_df())
```

- KV interface can be used in both the app (via a library or client) and on the CSA.
- Let's assume (for this talk) the object keys are the same on both host and CSA.
- Now it becomes simple for the SQL query to be moved off the CPU and onto the CSA using NVMe commands!
- NVMe namespace copy hash("data.parquet") to SLM namespace.
- NVMe computational Program load vendor-specific program for SQL query.
- NVMe Computational Program execute SQL query against the parquet "file" which is now in SLM namespace. Results can be returned to host or written to SLM or KV namespace.



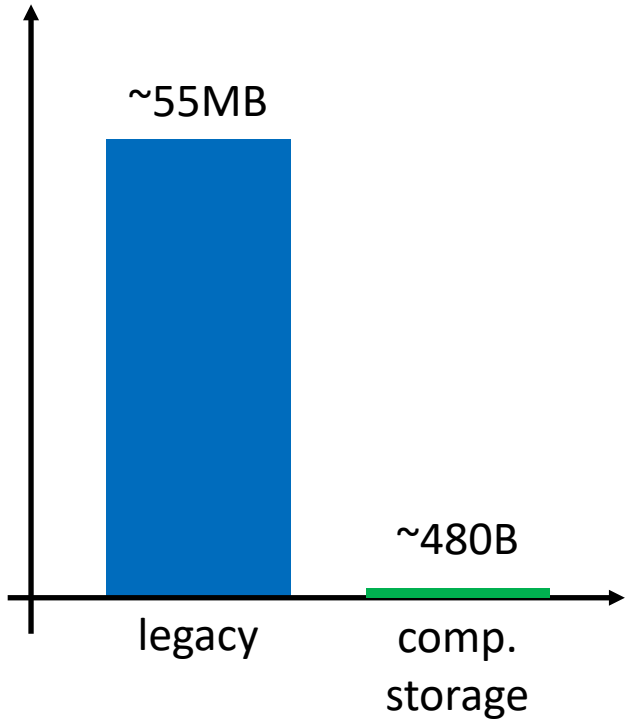
 © SNIA. All Rights Reserved.

- Conclusions

 © SNIA. All Rights Reserved.

Conclusions

- NVMe-based Computational Storage Arrays (CSAs) are coming!
- Using the NVMe-KV command set and namespaces can simplify certain scenarios compared to LBA-based command sets and namespaces.
- Computational Storage has several major customer benefits:
  - Massive reduction in data movement leading to better power consumption and lower networking costs.
  - Performing compute on the CSA reduces the computational load on the compute nodes. In our case we shifted about **7.8 million instructions** from the compute node to the storage node.
  - Performing compute on the CSA may reduce latency.



In our analytics example we reduced the storage network traffic from 55MB to 480B! That's about x100,000 less!!

 © SNIA. All Rights Reserved.

SNIA COMPUTE + MEMORY + STORAGE SUMMIT

Architectures, Solutions, and Community
VIRTUAL EVENT, APRIL 11-12, 2023

Please take a moment to rate this session.

Your feedback is important to us.

 © SNIA. All Rights Reserved.