Computational Storage A Standards Update

Scott Shadley

SNIA Executive BoD

SNIA CMSI GB

NVM Express Member

Solidigm Long-Term Strategy

The Evolution of Compute

Why Now for Compute Beyond the CPU

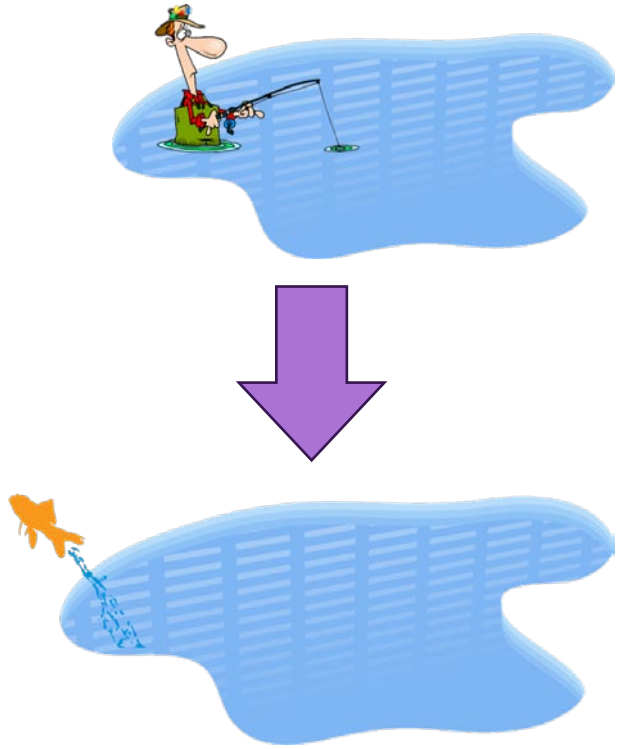
A Quick Recap of Why We Are Here

Why Now?

- Storage is no longer ‘SLOW’
- Memory is no longer ‘Gated’
- Data Gravity, Data Size, Data Locality
- Edge Data Explosion, Transport issue!
- SNIA, NVMe, CXL, OCP, Others are providing new guidance in new areas of implementation

Key Benefits?

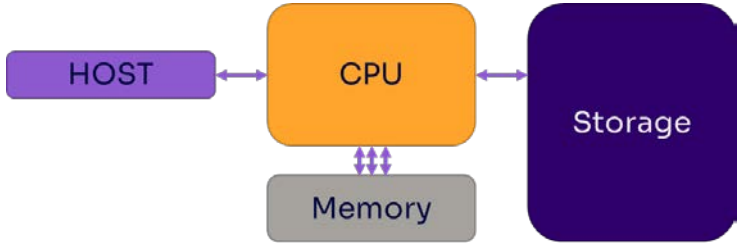
- Faster, Fewer, Easier I/O transfers
- Reducing DRAM/Network tax with new transports, solutions, products
- Redeploy Primary CPU to High Value Work, offer up new services to help
- Improved performance due to parallelism for certain workloads
- Better scheduling of data management and device functionality



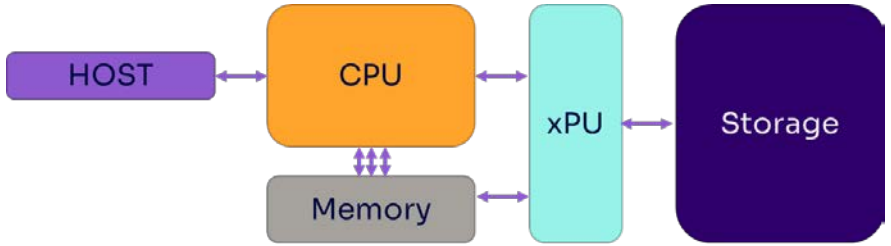
SNIA Architecture and API, NVM Express Command Infrastructure

John von Neumann – The Princeton Architecture

- The Ecosystem today, we have our friend J. v N. – CPU/Memory/Storage



- The world is evolving and there is a need for compute in more available locations, enter the world of "Accelerators" – SmartNIC, xPU, DPU, GPU, IPU



- These are great, but there is room for more! History tells Us this much…

Standardizing Computational Storage

On Behalf of the CS TWG Co-Chairs:

Bill Martin

Jason Molgaard

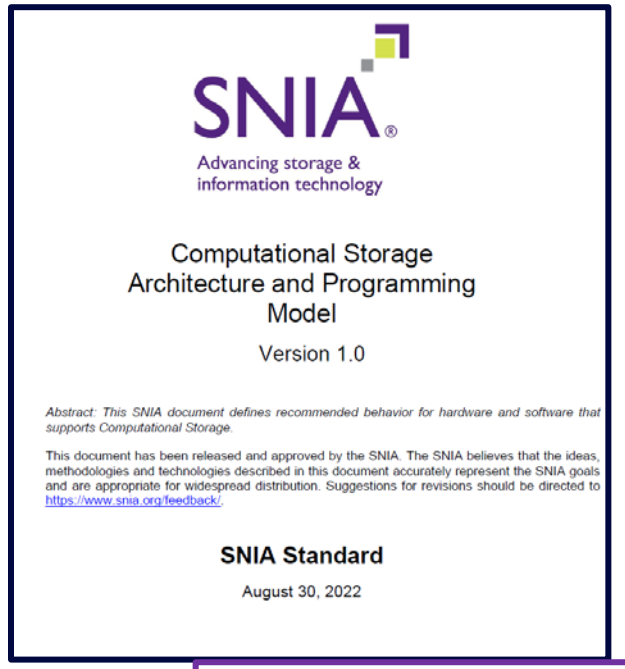
The Continued Growth of Experience

- CS TWG is continuing to see growth
  - **48** companies, **258** individual members
- Work within **SNIA** Efforts
  - **CS SIG** – Webinars, Blogs, Events
  - **SDXI** – Sub-Group Collaboration
  - **Security** TWG – Addressing Security
- Collaborating with External Groups
  - **NVM Express** – Computational Programs

Current Progress of TWG Output

- **Architectural Document v1.0 has been released**
- v1.1 under development
  - Security enhancements for multiple tenants
  - Chaining of Commands
  - Expansion of use cases

- **API v0.8 public review version also available**
- API v1.0 under development
  - Abort/reset handling
  - Device Memory
  - NVMe Computational Programs support
  - Other Miscellaneous Updates


SNIA
Advancing storage & information technology

Computational Storage Architecture and Programming Model

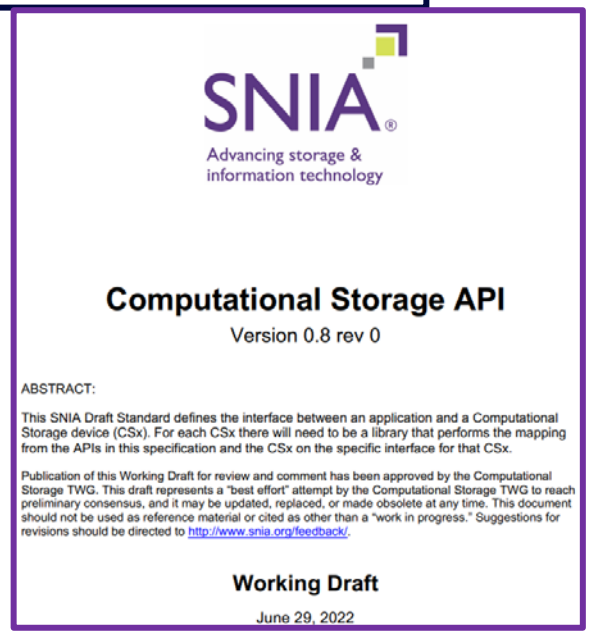
Version 1.0

Abstract: This SNIA document defines recommended behavior for hardware and software that supports Computational Storage.

This document has been released and approved by the SNIA. The SNIA believes that the ideas, methodologies and technologies described in this document accurately represent the SNIA goals and are appropriate for widespread distribution. Suggestions for revisions should be directed to https://www.snia.org/feedback/.

SNIA Standard

August 30, 2022


SNIA
Advancing storage & information technology

Computational Storage API

Version 0.8 rev 0

ABSTRACT:

This SNIA Draft Standard defines the interface between an application and a Computational Storage device (CSx). For each CSx there will need to be a library that performs the mapping from the APIs in this specification and the CSx on the specific interface for that CSx.

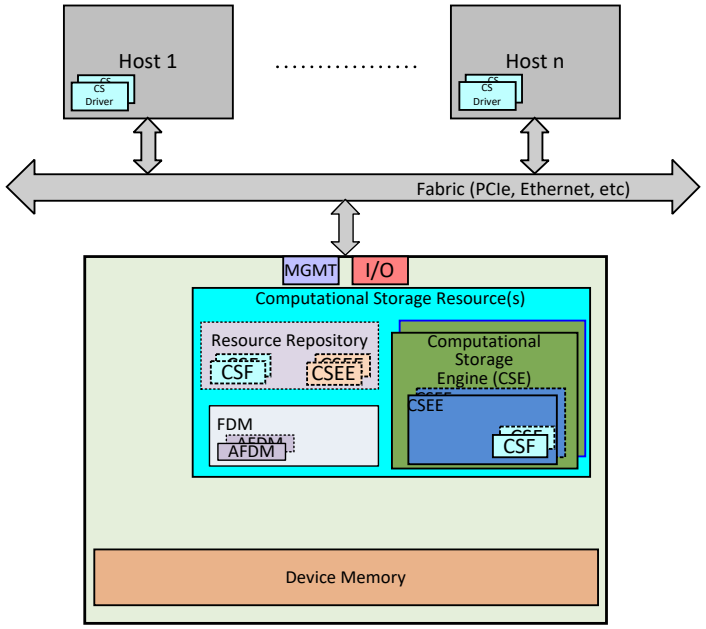
Publication of this Working Draft for review and comment has been approved by the Computational Storage TWG. This draft represents a "best effort" attempt by the Computational Storage TWG to reach preliminary consensus, and it may be updated, replaced, or made obsolete at any time. This document should not be used as reference material or cited as other than a "work in progress." Suggestions for revisions should be directed to http://www.snia.org/feedback/.

Working Draft

June 29, 2022

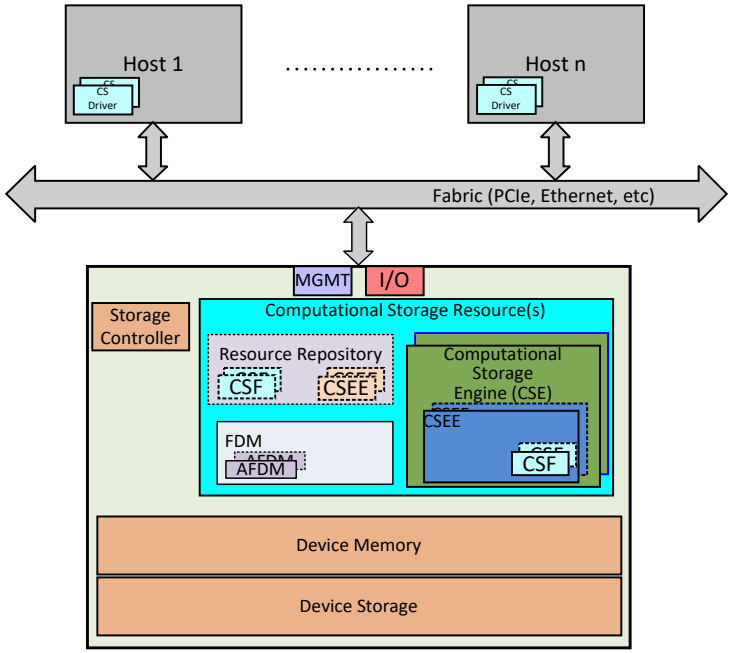
Computational Storage Architecture

Computational Storage Processor



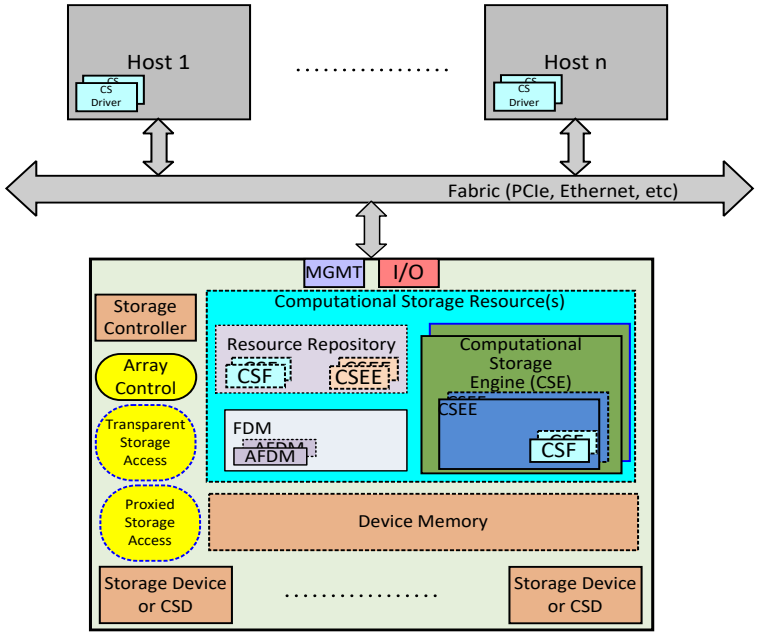
Computational Storage Processor (CSP)

Computational Storage Drive



Computational Storage Drive (CSD)

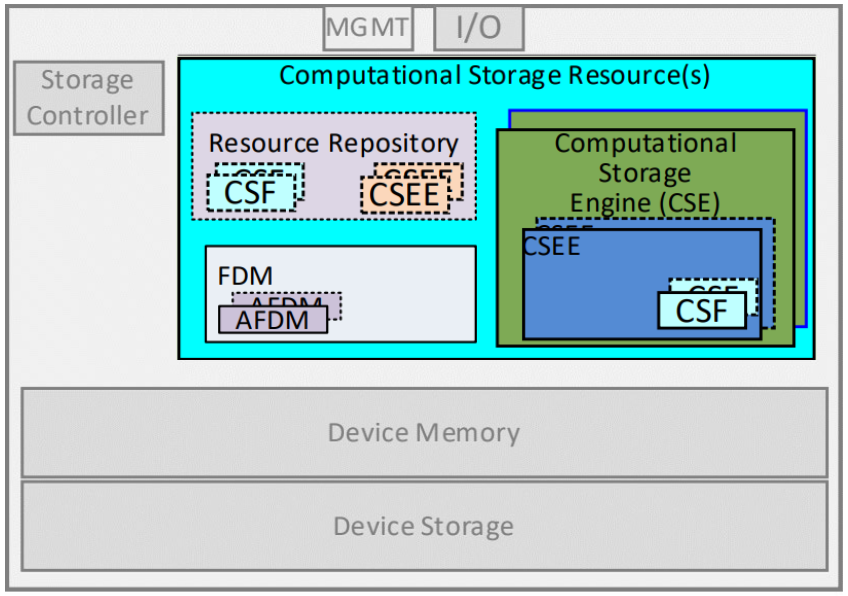
Computational Storage Array



Computational Storage Array (CSA)

CSx = Computational Storage **Device** – CSP or CSD or CSA

Deep Dive of the CSx Resources



Computational Storage Drive (CSD)

[SNIA Dictionary - CS Terms]

CSR - Computational Storage Resources are the resources available in a CSx necessary for that CSx to store and execute a CSF.

CSF - A Computational Storage Function is a set of specific operations that may be configured and executed by a CSE in a CSEE.

CSE - Computational Storage Engine is a CSR that is able to be programmed to provide one or more specific operation(s).

CSEE - A Computational Storage Engine Environment is an operating environment space for the CSE.

FDM - Function Data Memory is device memory that is available for CSFs to use for data that is used or generated as part of the operation of the CSF.

AFDM - Allocated Function Data Memory is a portion of FDM that is allocated for one or more specific instances of a CSF operation.

API Use In Computational Storage

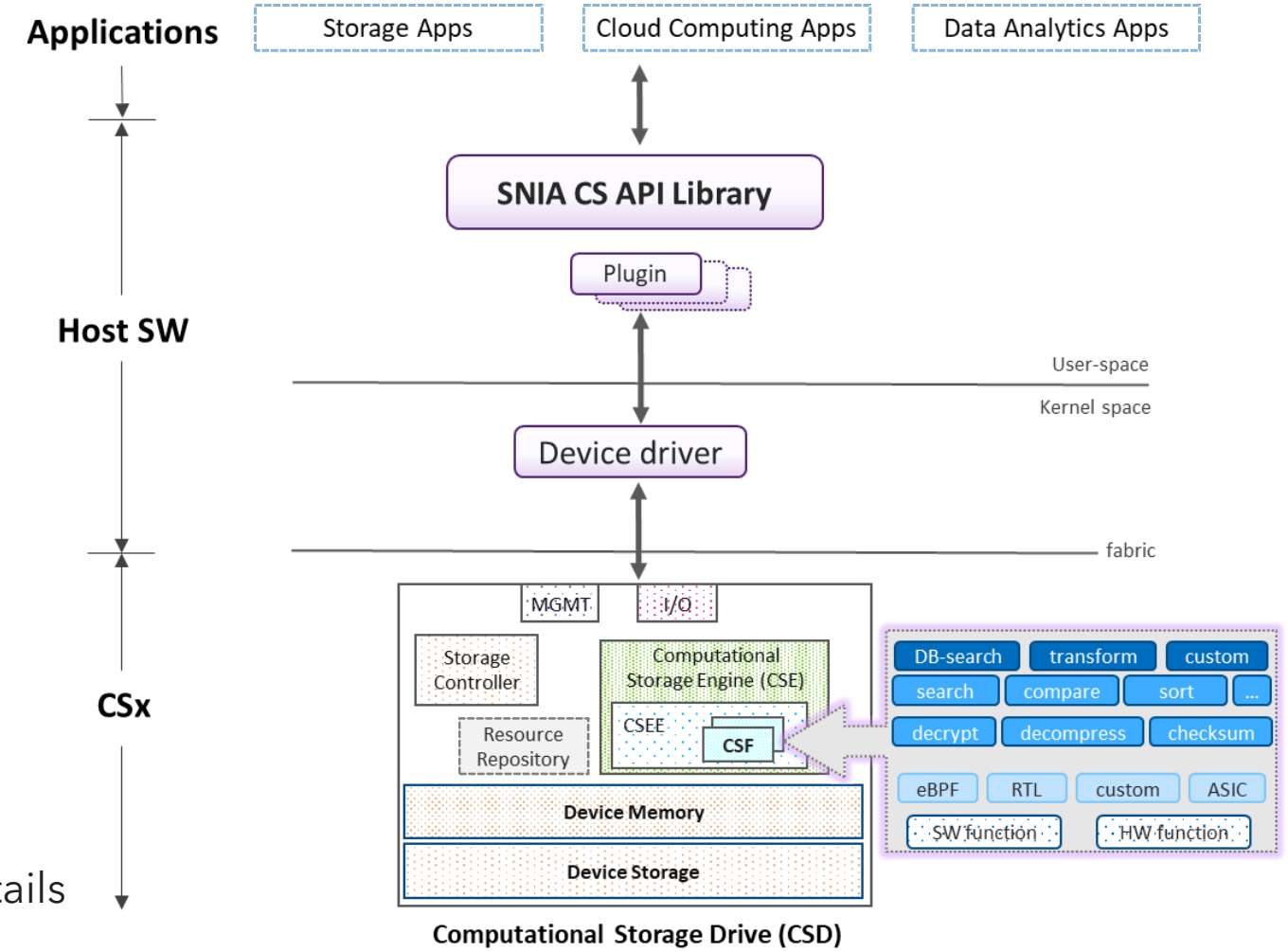
On Behalf of the CS API Lead Editors:

Oscar Pinto

Bill Martin

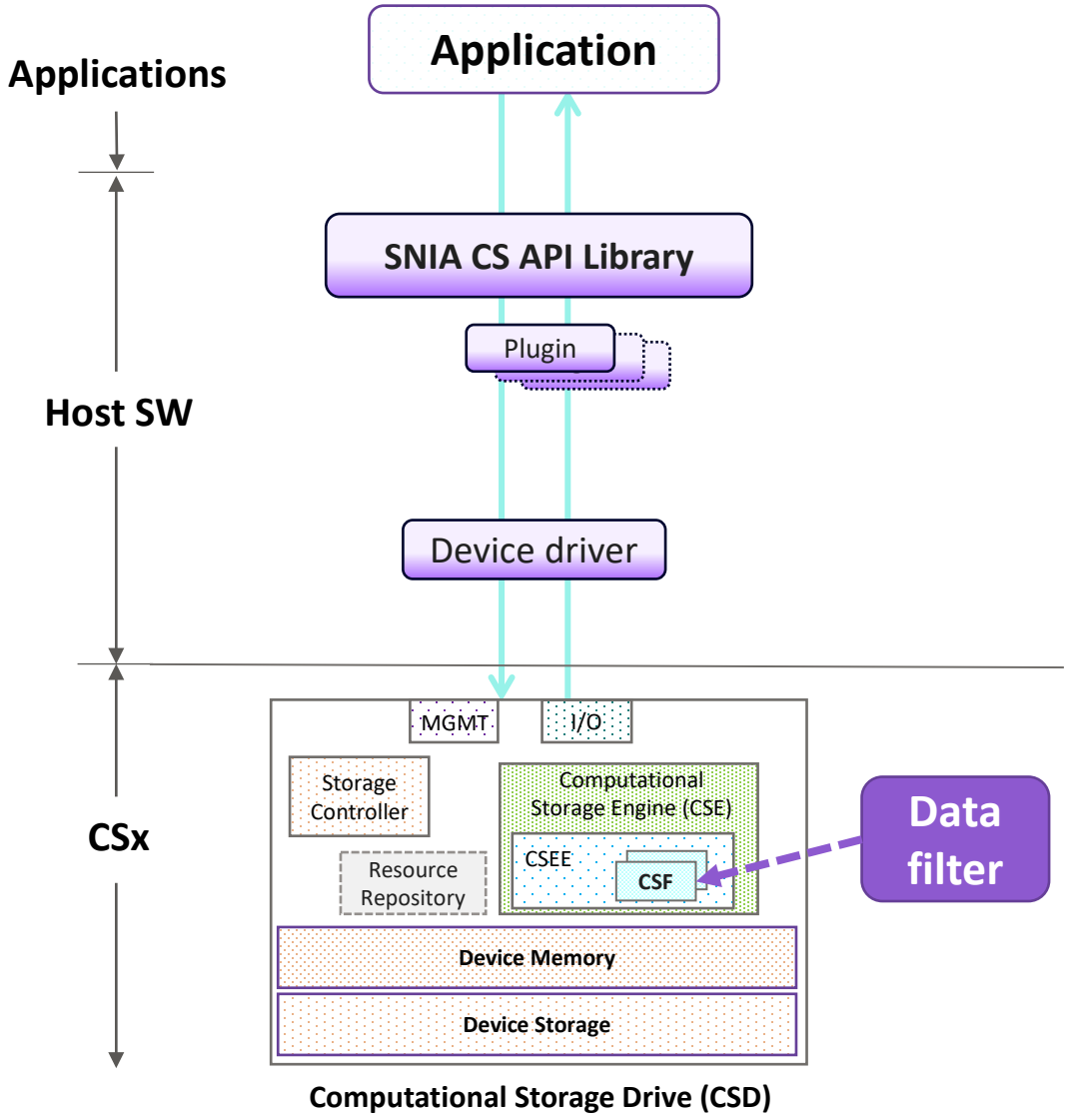
SNIA Computational Storage APIs

- One set of APIs for all CSx types
  - CSP, CSD, CSA
- APIs hide device details
  - Hardware, Connectivity (local/remote)
- Abstracts device details
  - Discovery
  - Access
  - Device Memory (mapped/unmapped)
  - Near Storage Access
  - Copy Device Memory
  - Download CSFs
  - Execute CSFs
  - Device Management
- Extensible Interface
  - Plugins connect CSx to abstracted APIs
  - Hides vendor specific implementation details
- APIs are OS agnostic

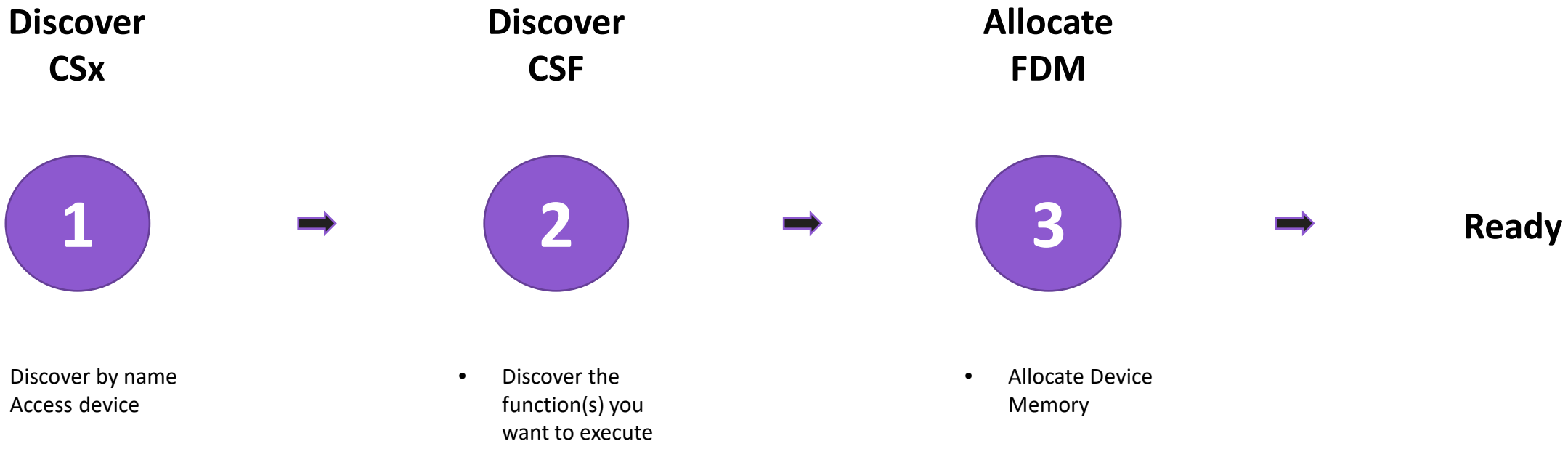


Example Use Case - Abbreviated

- Execute Data Filter CSF
  - Allocate Device Memory (FDM)
  - Load Data from Storage
  - Run Data Filter CSF on loaded Data
  - Copy Results to Host Memory



Prepare for Computational Storage (Setup)



Perform Computational Storage I/O (Run)

Load Storage Data

4

- Load Data near Compute

Execute CSF

5

- Run Compute Operation

Copy Results

6

- Copy from FDM into Host Memory

Done

Usage Summary

6 Steps

1
```
csGetCSxFromPath(“my_file_path", &length, &csxBuffer);
csOpenCSx(csxBuffer, &MyDevContext, &devHandle);
```

2
```
csGetCSFId(devHandle, “filter”, &infoLength, &count, &csfInfo);
```

3
```
csAllocMem(devHandle, CHUNK_SIZE, &f, &afdmHandle1, NULL);
```

4
```
csQueueStorageRequest(storReq, storReq, NULL, NULL, NULL, &compVal);
```

5
```
csQueueComputeRequest(compReq, compReq, NULL, NULL, NULL, &compVal);
```

6
```
csQueueCopyMemRequest(copyReq, copyReq, NULL, NULL, NULL, NULL);
```

Bridging the Gap
Architecture to Implementation

Correlation of SNIA/NVMe terms

SNIA Terms

- Computational Storage Engine (CSE)
- Computational Storage Engine Environment (CSEE)
- Resource Repository
  - Downloaded CSF and CSEE
  - Pre-loaded CSF and CSF
- Activation
- Function Data Memory (FDM)
- Allocated FDM (AFDM)
- Device Storage

NVMe Terms

- Compute Namespace
- Virtual (Not currently defined)
- Programs
  - Downloaded programs
  - Device-defined programs
- Activation
- Subsystem Local Memory (SLM)
- Memory Range
- NVM Namespaces

Differences between SNIA and NVMe

SNIA

- Defines CSEE
- CSF can directly access AFDM or Storage
- Supports an indirect model

NVMe

- CSEE is logical – no specific definition
- Program has access to Memory Range Set
- Specific Execute command only

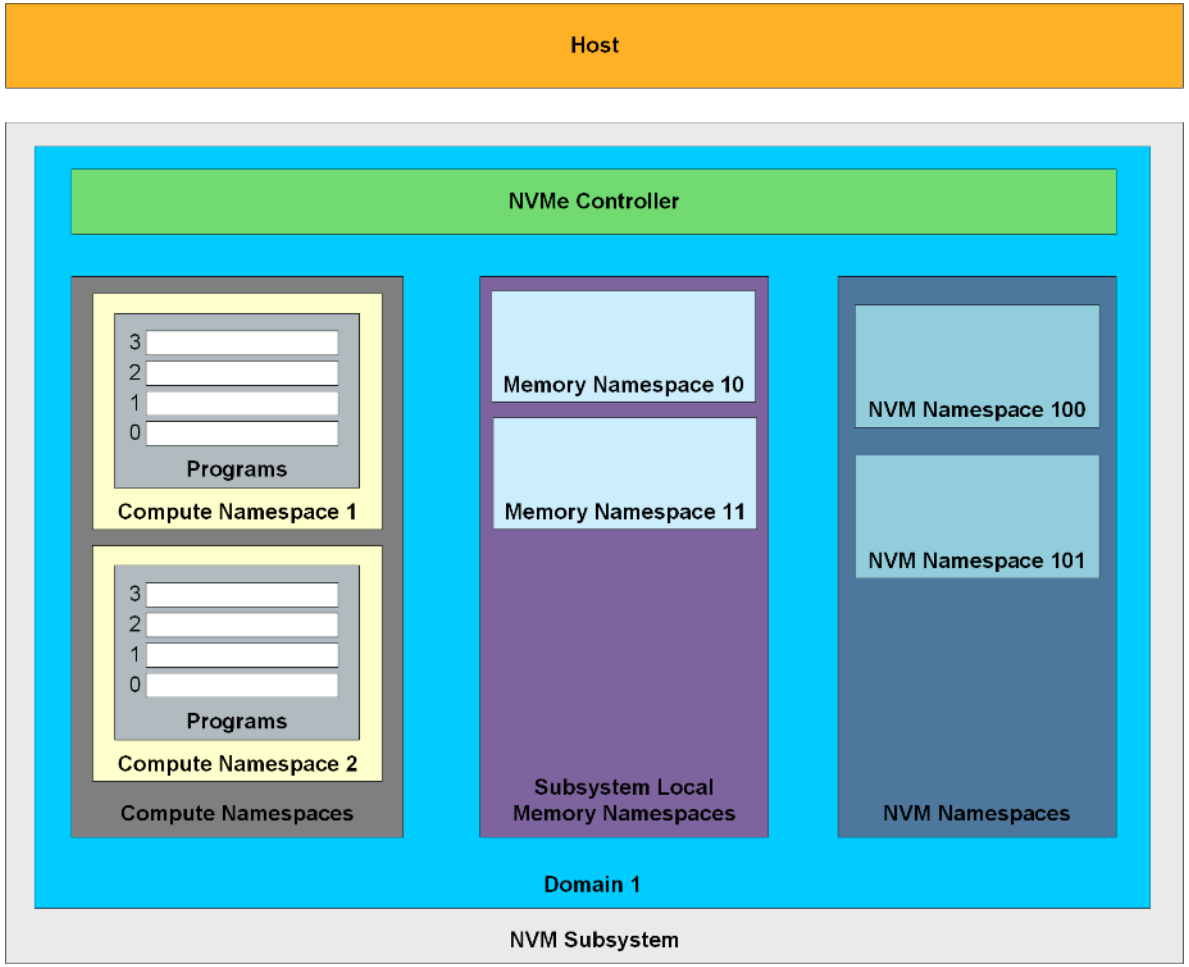
A Look at the NVMe Implementation

On Behalf of the NVM Express Computational Storage Task Group:

Bill Martin

Kim Malone

NVMe Major Architectural Components



The NVM Express® (NVMe®) computational storage architecture involves several types of namespaces:

- Compute namespaces (new)
- Memory namespaces (new)
- NVM namespaces
  - NVM, Zoned, and Key Value namespaces

Compute Namespaces

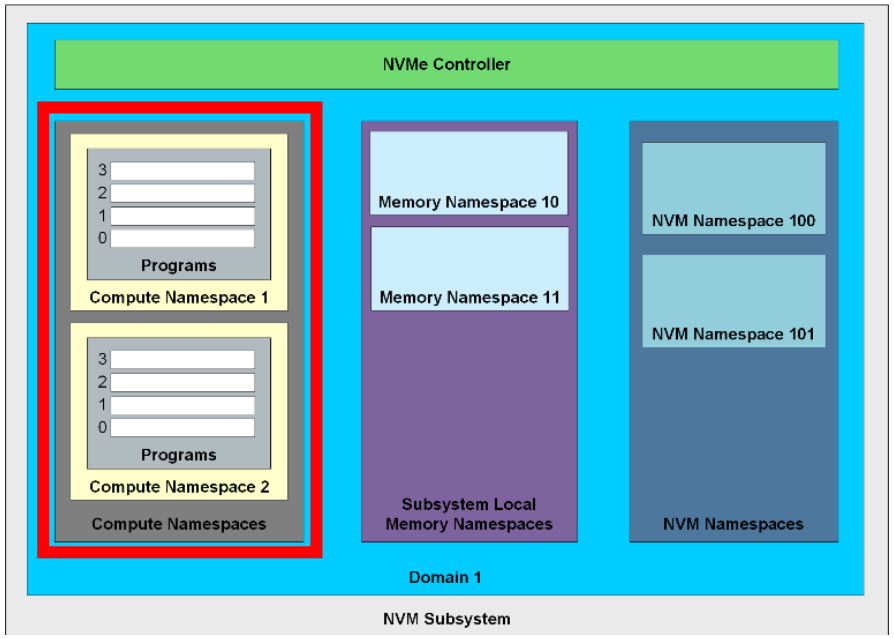
A compute namespace:

- Is a namespace in an NVMe technology subsystem that is able to execute one or more programs
- Is a namespace that is associated with the Computational Programs I/O command set
- Contains compute resources

TP4091: Computational Programs

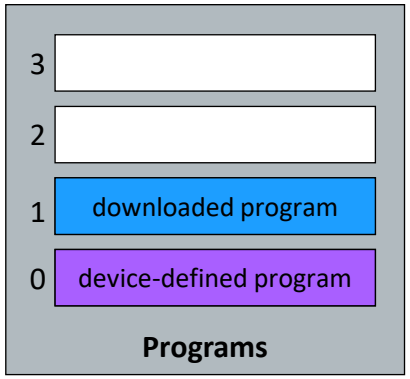
New Computational Programs I/O command set for compute namespaces

- New commands include:
  - Execute program
  - Load program
  - Activate program
  - Create/Delete Memory Range Set
- Provides log pages for program discovery



Computational Programs

- Conceptually similar to software functions
  - Called with parameters and run to completion
- Are addressed via a compute namespace program index
- May be identified by a globally unique program identifier
- Operate only on data in Subsystem Local Memory
- May be device-defined or downloadable
  - Device-defined programs
    - Programs provided at time of manufacture e.g., compression, encryption
  - Downloadable programs
    - Programs that are loaded to a Computational Programs namespace by the host
- A program may only be able to execute on a subset of the compute resources in an NVM subsystem
  - A program may be implemented in an ASIC
  - A program may be executed on a CPU core



Memory Namespaces

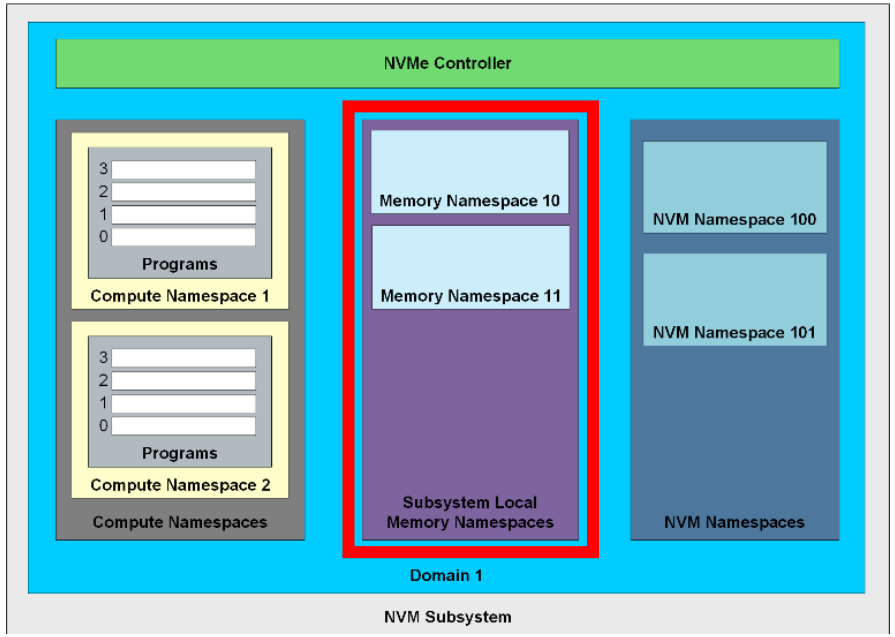
A memory namespace:

- Is a namespace in an NVMe technology subsystem that provides host command access to memory in the NVMe technology subsystem
- Is a namespace that is associated with the Subsystem Local Memory I/O command set
- Is used by the Computational Programs command set to provide access to SLM for program execution

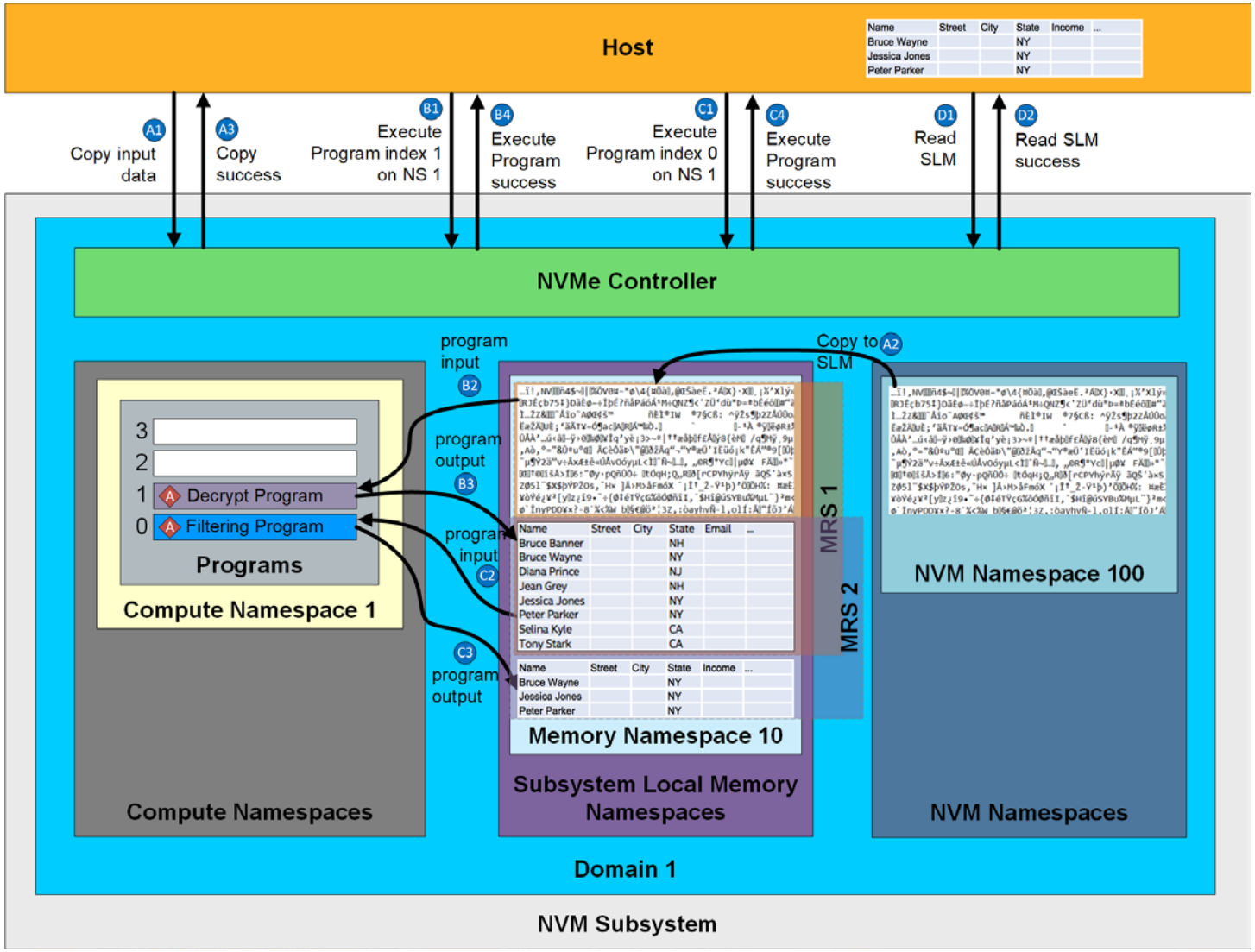
TP4131: Subsystem Local Memory (SLM)

New Subsystem Local Memory I/O command set for memory namespaces

- New commands include:
  - Memory read and memory write
    - Commands for transferring data between host memory and a memory namespace
  - Memory copy
    - Command for copying data between NVM and memory namespaces



Flow: Execute Program – Filter Encrypted Data



Precondition:

- Memory Range Sets MRS1 and MRS2 have been created

Flow steps

A. Copy encrypted data into SLM

B. Execute Program 1 on compute NS 1 using MRS1

C. Execute Program 0 on compute NS 1 using MRS2

D. Read filtered data from SLM to host

What's Next?

How Can You Help!

Help Progressing Computational Storage

- Join the SNIA Technical Work Group
  - Go to SNIA Member Portal
  - Select TWG: Computational Storage
  - Click on the "Join Group"
- CS TWG meetings
  - Wed 10-11am Pacific time

- Join the task group
  - Go to the NVMe workgroup portal
  - Select the CS Task Group
  - Click on the "Join Group" link
- Task group meetings
  - Thursdays 9–10am Pacific time

Thank you

Scott.Shadley@Solidigm.com