STORAGE DEVELOPER CONFERENCE

SDC 21

BY Developers FOR Developers

Virtual Conference
September 28-29, 2021

A SNIA Event

NVMe Computational Storage

Standardizing offload of computation via NVMe

Kim Malone, Intel and Stephen Bates, Eideticom

Agenda

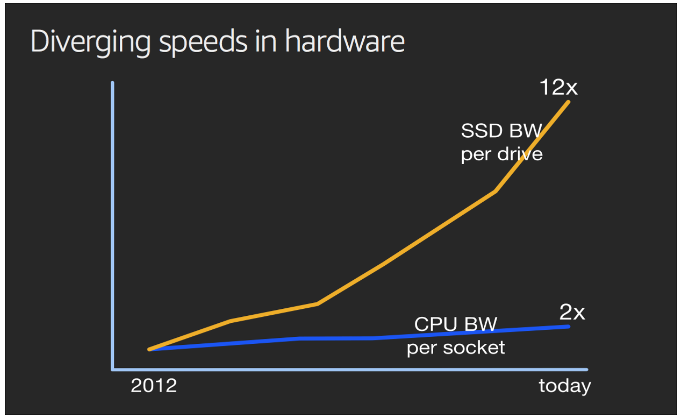
- Motivation and Background
- Why NVMe
- NVMe Computational Storage Architecture
- Example Flows
- NVMe changes for computational storage
- NVMe Computational Storage Task Group

Motivation

Why computational storage?

Motivation

- Explosion in stored data
- Diverging CPU and SSD bandwidth
  - Rise of the heterogeneous compute and distributed processing
- Desire for reduction of data movement
  - Save power, reduce TCO
  - Save networking bandwidth
  - Free up host CPU cycles
- Desire for standardized interfaces for offloading compute near storage



Background

What problem are we solving?

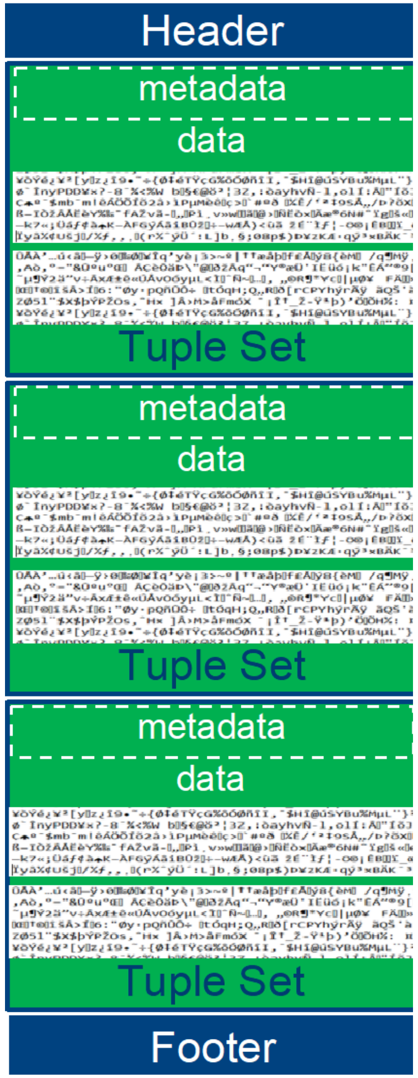
The Complex Data Universe

- Data Warehouses (Presto, SparkSQL, …) store LOTS of data
- Data stored in LOTS of (arbitrary) formats
- Data stored compressed and encrypted
- Formats and data constantly evolving

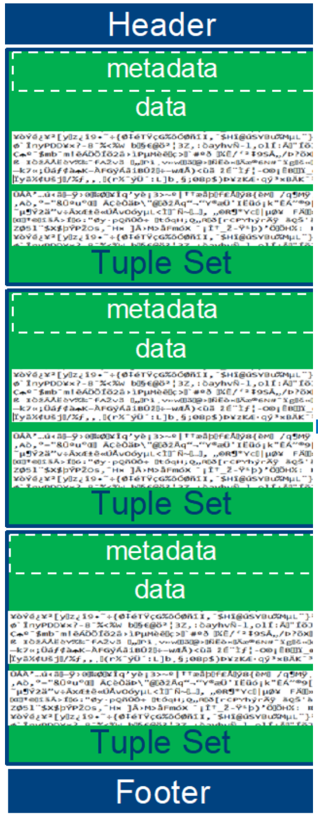
Data Table

NAME	STATE
Bruce Banner	NH
Bruce Wayne	NY
Diana Prince	NJ
Jean Grey	NH
Jessica Jones	NY
Peter Parker	NY
Selina Kyle	CA
Tony Stark	CA

Compressed, Encrypted, Arbitrary Format



Finding the Needle in the Haystack



Compressed,
Encrypted,
Arbitrary Format

FILTER Select Name where State="NY"

→ decrypt 'names' →

```
20 34 35 33 20 2F 54 20
35 37 20 2F 43 20 36 37
20 2F 46 69 6C 74 65 72
65 63 6F 64 65 20 2F 4C
35 3E 3E 0A 73 74 72 65
60 65 60 50 79 C7 C0 C6
04 80 62 EC 0C 2C 0C 1C
30 07 30 30 B0 94 73 A8
93 6A 60 60 11 30 68 74
AF E0 32 D1 A8 D5 5B DC
56 21 93 CE 60 F5 80 A5
36 EB 8D 36 2F 78 7A 79
52 6B 8F 94 45 25 93 4A
```

→ decompress 'names' →

Bruce Banner
Bruce Wayne
Diana Prince
Jean Grey
Jessica Jones
Peter Parker
Selina Kyle
Tony Stark

→ filter 'names' →

Bruce Wayne
Jessica Jones
Peter Parker

→ aggregate (count) →

3

Why NVMe?

Why not?

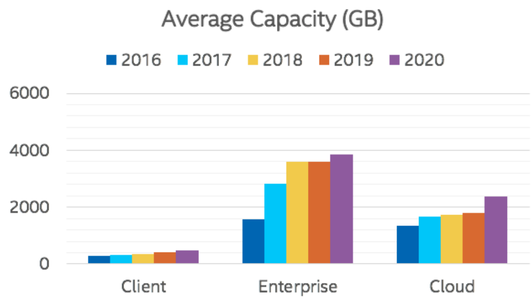
Why NVMe??

- NVMe is widely deployed
- Existing ecosystem
- Well supported in OSes
- Has the most active storage interface development community
- Extensible & efficient
- Support for both PCIe and networks
- Comprehends storage and data
- Why not?

NVMe® Technology Powers the Connected Universe

Units (Ku)	2016	2017	2018	2019	2020	2021*
Enterprise	364	749	1,069	2,045	4,910	7,290
Cloud	2,051	3,861	10,369	12,276	19,205	20,349
Client	33,128	48,951	82,587	143,236	226,221	350,253

* Projections provided by Forward Insights Q2'21



- NVMe shipped > 160K Petabytes in 2020! (Enterprise ~18K, Cloud ~43K, and Client ~99K)
- Excellent growth in units and incremental capacity growth across all three segments

NVMe Computational Storage Architecture

An extensible architecture

STORAGE DEVELOPER CONFERENCE SDC 21

Overview

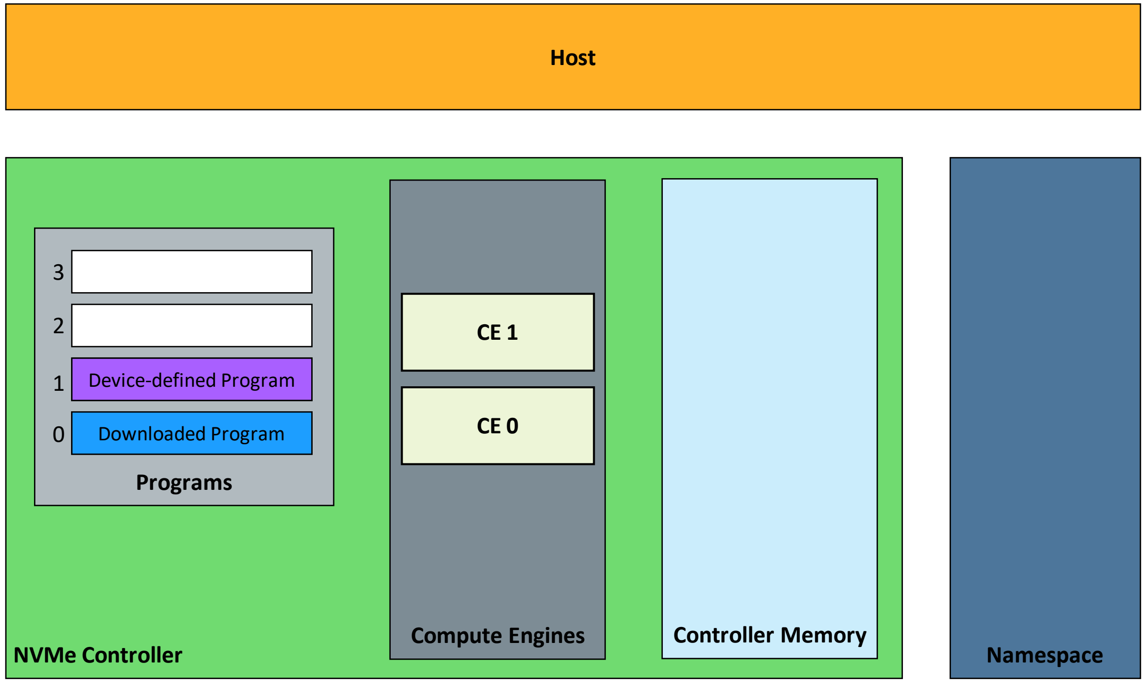
Memory-Centric

eBPF

Compute Engines

Programs (Downloadable or Device-Defined)

Major Architectural Components



- Programs operate only on data in Controller Memory
  - Includes program input, output
  - Data is moved between Controller Memory and host memory using new NVMe commands
- Existing I/O command sets are used to transfer data between namespaces and Controller Memory

This presentation discusses NVMe work in progress, which is subject to change without notice.

Programs as Computational Storage Offloads

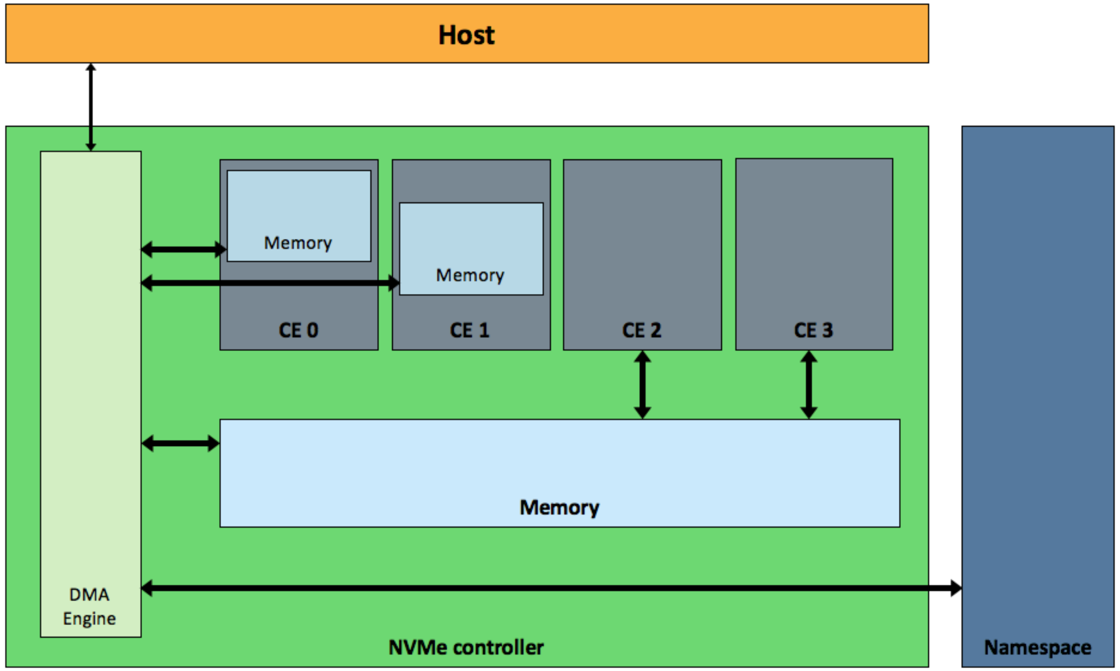
Programs:

- Invoked and used in a standard way
  - Conceptually similar to software functions
  - Called with parameters and run to completion
- Operate on data in on-device memory
- Run on a Compute Engine
- May be in hardware or software
  - Device may offer fixed function programs, or
  - Downloadable in hardware agnostic bytecode (eBPF) from host for later execution
- Managed on a per-NVMe controller basis

This presentation discusses NVMe work in progress, which is subject to change without notice.

Compute Engines

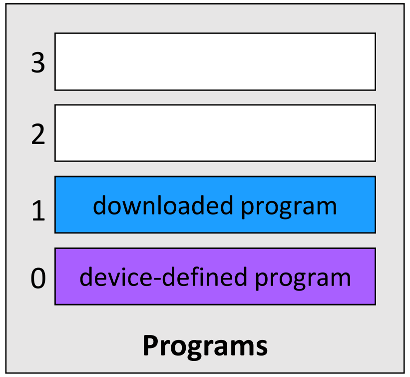
- A Compute Engine (CE) is an entity on an NVMe controller that can execute a computational program
  - Examples: CPU core, ASIC, FPGA
- CEs may have asymmetric access to controller memory
- A computational program may only be able to execute on a subset of a controller's CEs



This presentation discusses NVMe work in progress, which is subject to change without notice.

Downloadable and device-defined programs

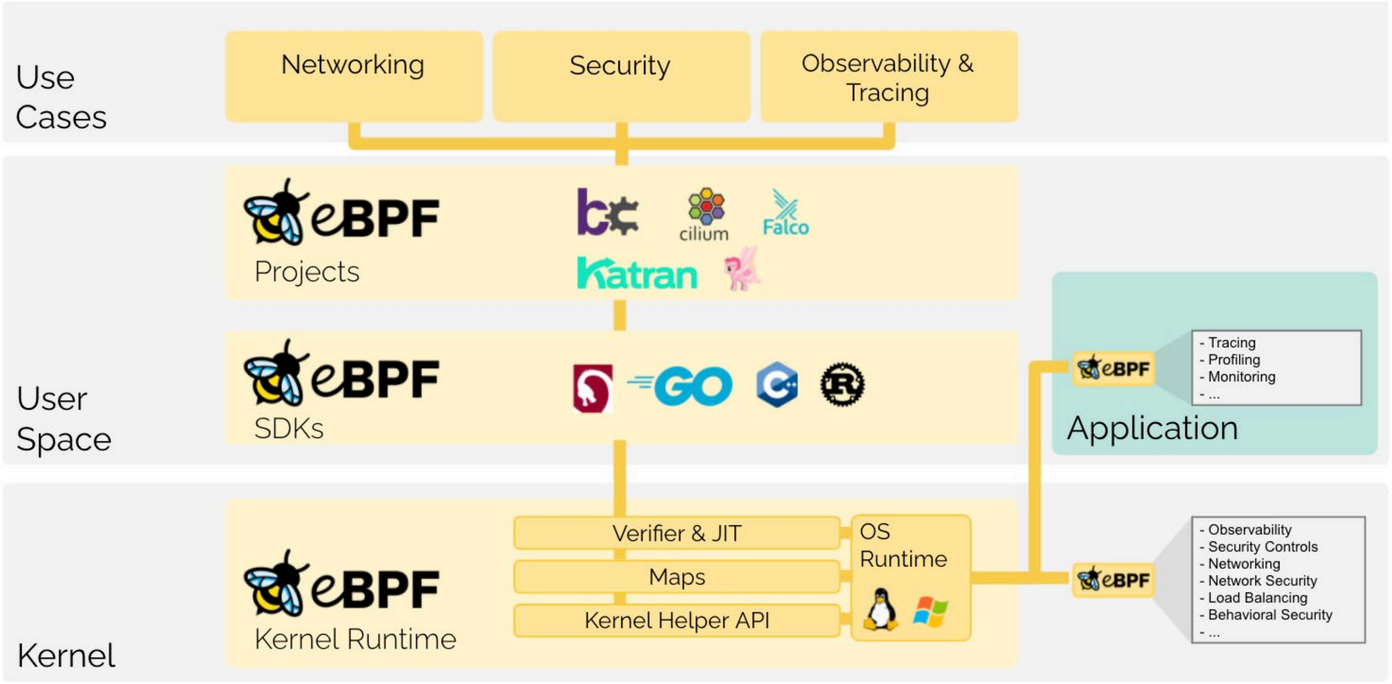
- Support for both device-defined and downloadable programs
- Device-defined programs
  - “Fixed” programs provided by the NVMe controller
  - Functionality implemented by the device that are callable as programs
  - e.g. compression, decryption
- Downloadable programs
  - Programs that are loaded onto the NVMe controller by the host



This presentation discusses NVMe work in progress, which is subject to change without notice.

eBPF for Downloadable Programs

- Why downloadable programs?
  - Flexibility
  - Process complex formats
  - Emerging applications
  - Portability from existing applications
- Why eBPF?
  - Vendor Agnostic
  - Well understood
  - Existing ecosystems
  - LLVM
  - Toolchains
  - Sits under Linux Foundation

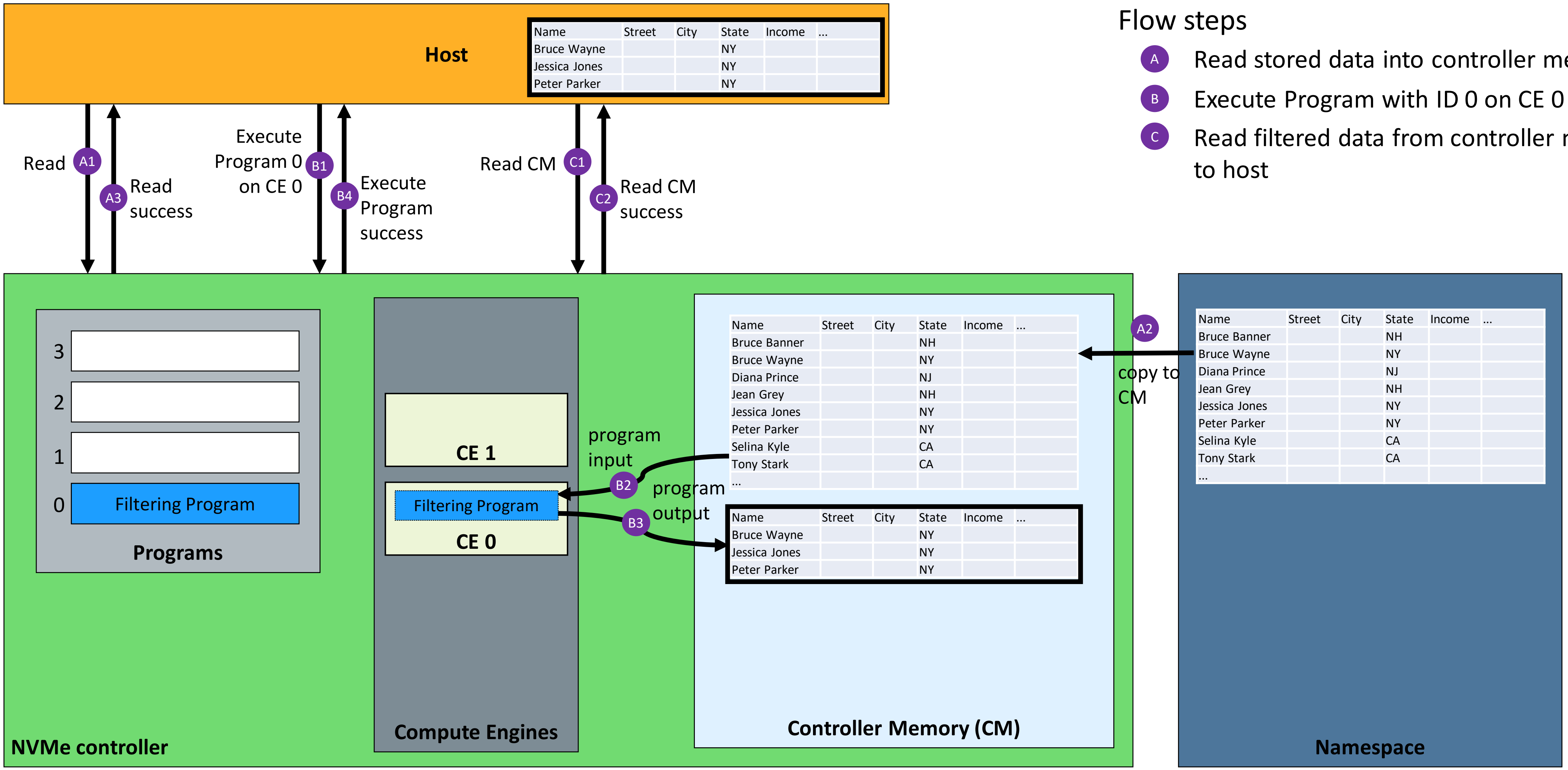


This presentation discusses NVMe work in progress, which is subject to change without notice.

Example Flows

How does it work?

Flow: Execute Program – Simple Data Filter

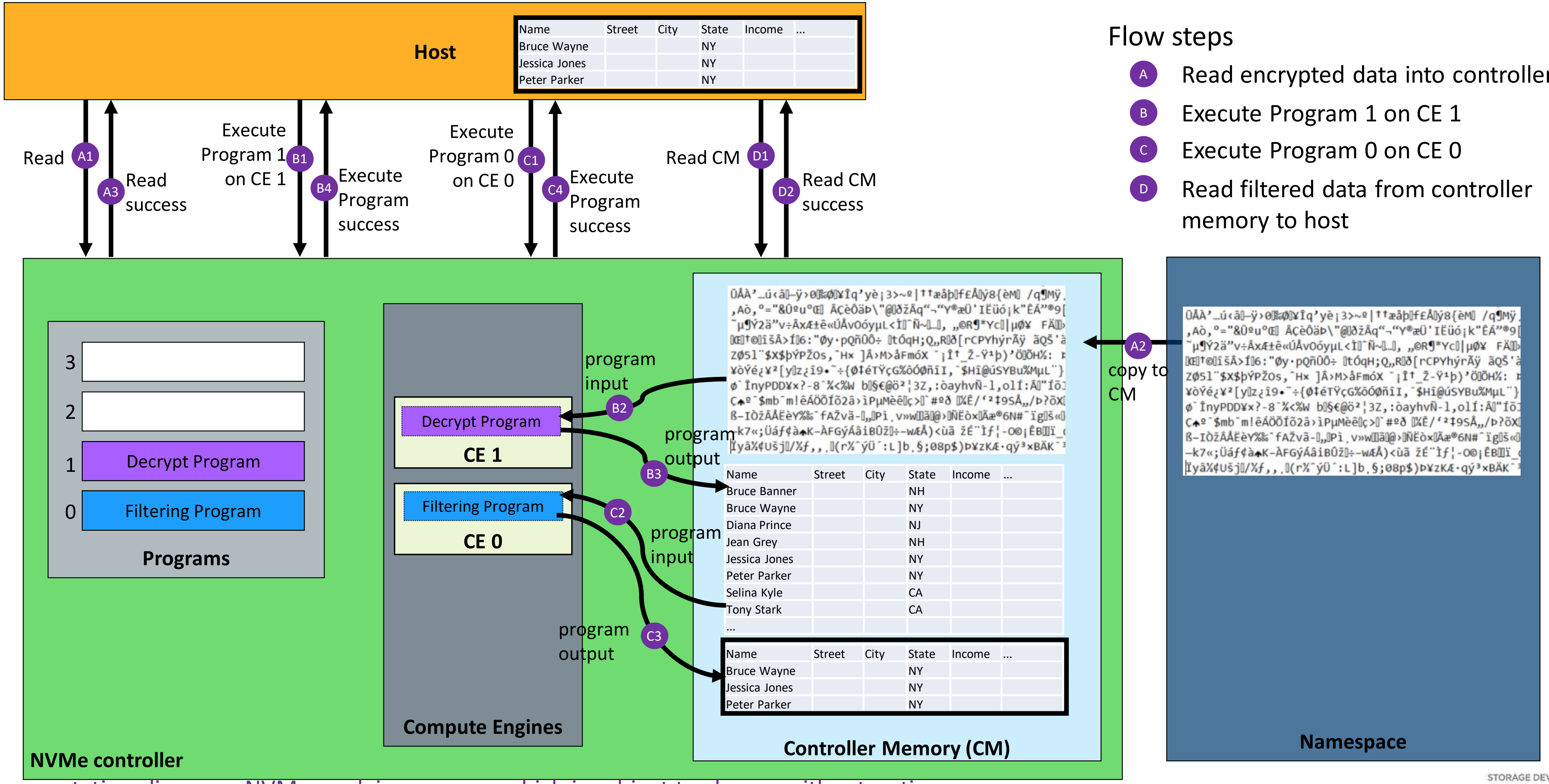


Flow steps

- A Read stored data into controller memory
- B Execute Program with ID 0 on CE 0
- C Read filtered data from controller memory to host

This presentation discusses NVMe work in progress, which is subject to change without notice.

Flow: Execute Program – Filter Encrypted Data



Flow steps

- A Read encrypted data into controller memory
- B Execute Program 1 on CE 1
- C Execute Program 0 on CE 0
- D Read filtered data from controller memory to host

This presentation discusses NVMe work in progress, which is subject to change without notice.

NVMe Changes for Computational Storage

Optional support

NVMe changes for Computational Storage

- TP4091 - Computational Programs I/O Command Set
  - Execute program
  - Load program
  - Activate program
- TP4131 – Controller Local Memory
  - Recent proposal that came out of this CS work
- Identify Controller command updates to indicate support/not
- New log pages to support Computational Programs
- Don't panic, this is all optional

This presentation discusses NVMe work in progress, which is subject to change without notice.

NVMe Computational Storage Task Group

Join us!

Computational Storage Task Group

- **Task group co-chairs**
  - Kim Malone (Intel)
  - Stephen Bates (Eideticom)
  - Bill Martin (Samsung)

- **Task Group Goals**
  - Define the architecture of TP4091
  - Take TP4091 through to ratification
  - Other CS TPs

- **Membership**
  - 167 members from 43 companies

- **Join the task group**
  - https://workspace.nvmexpress.org/apps/org/workgroup/portal/
  - Select the Computational Storage Task Group
  - Click on the “Join Group” link

- **Task group meetings**
  - Thursdays 9 – 10 am Pacific time

Please take a moment to rate this session.

Your feedback is important to us.

