Improving SW-HW processing pipeline for storage stack / service workflows with CXL

Presenters:

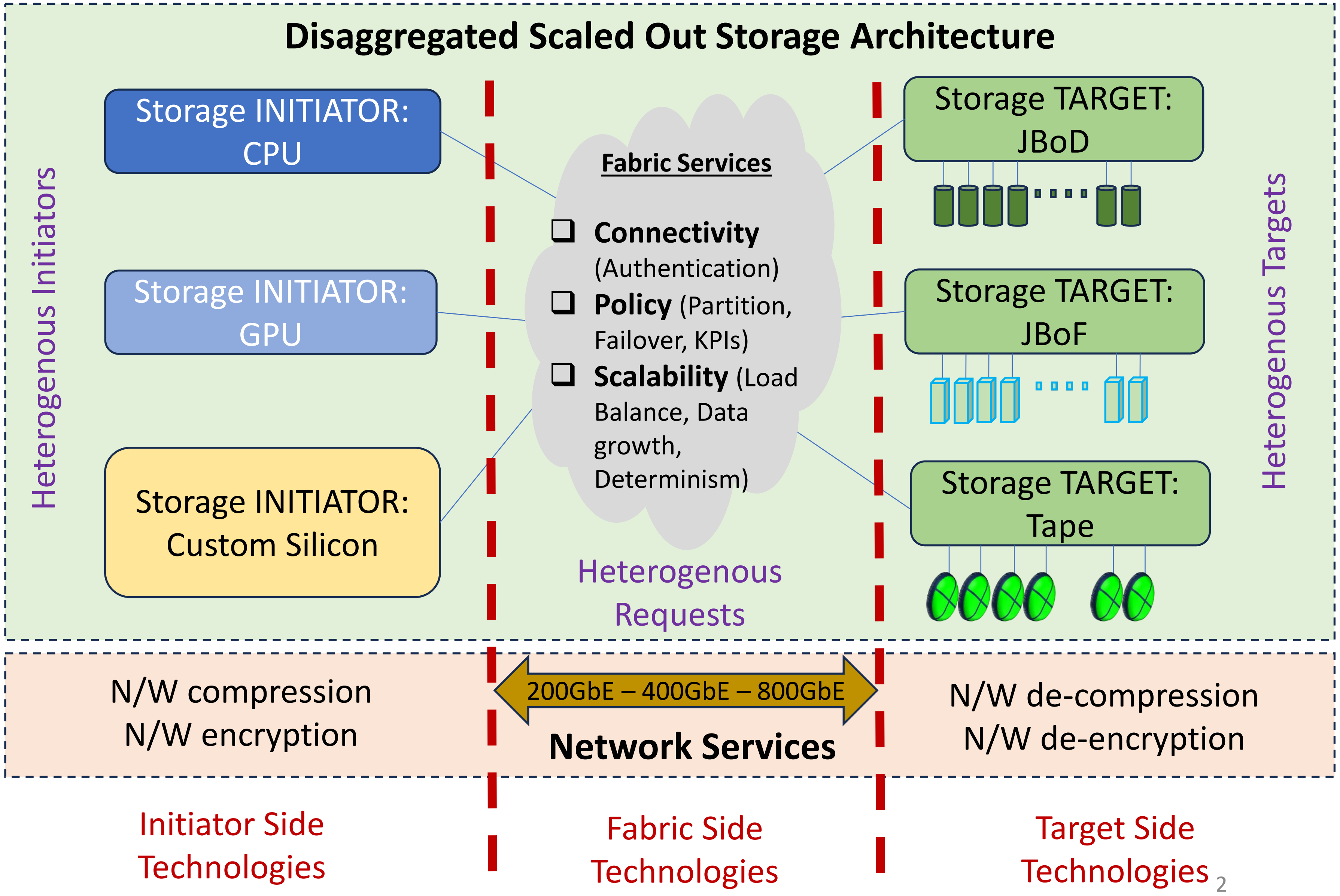
Navneet Rao, Solution Architect, DCAI / Altera, Intel

Bhushan Chitlur, Sr. Principal Engineer, DCAI / Altera, Intel

FMS the Future of Memory and Storage

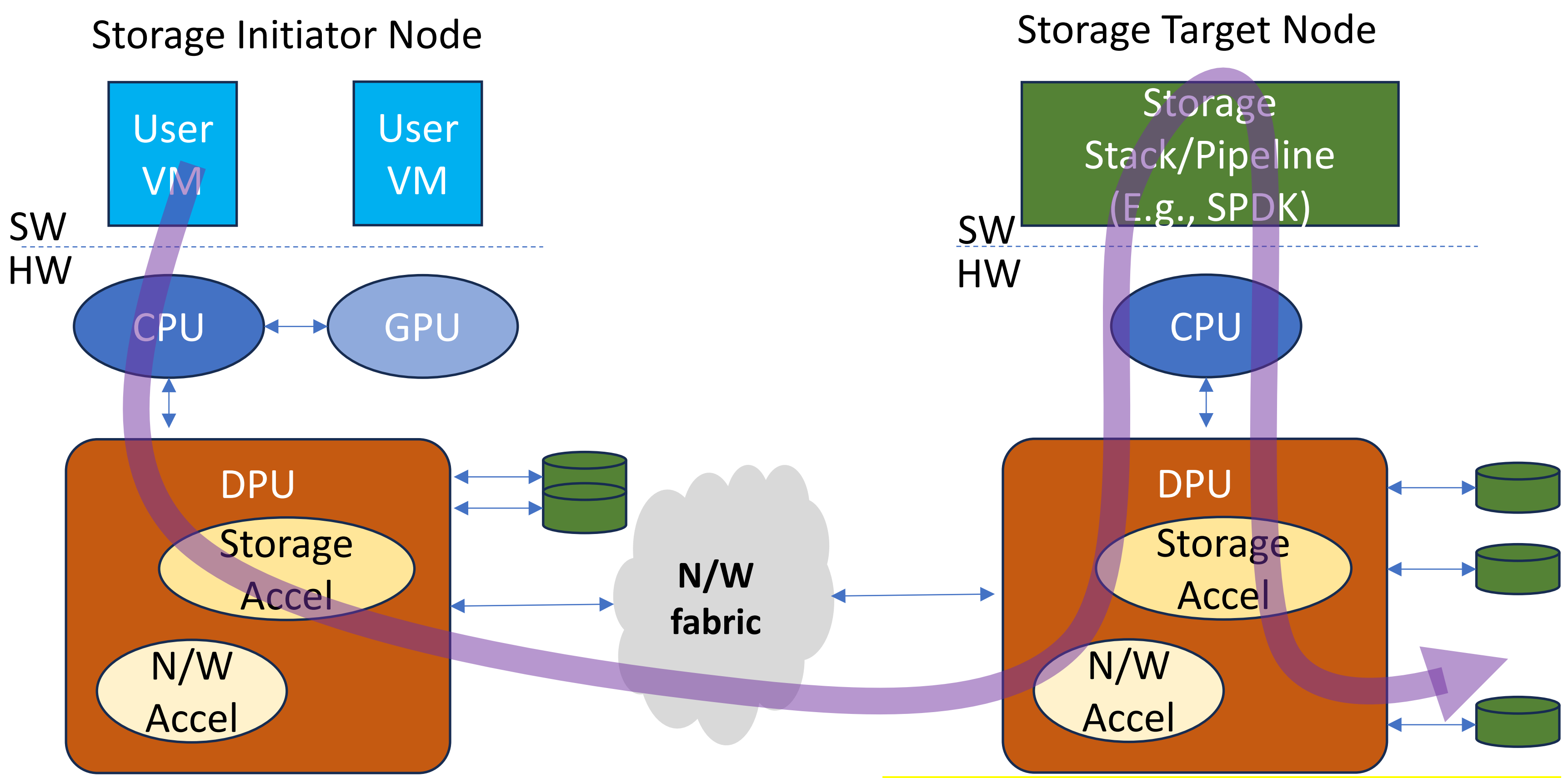
Building High Performance Storage Solutions

- Building scalable, disaggregated, secure, scaled-out datacenter storage infrastructure with reliability is extremely challenging
- Current accelerator offload techniques may not be sufficient to meet the increasing demand on high performance secure storage solutions



Rise of the DPU (aka IPU)

- DPU becomes the focal point for all infrastructure processing which includes networking and storage
- Storage target node requires significantly more storage specific computation (Focus of today's talk)



Initiator Storage Functions

- ❑ E.g., Virtio-blk, NVMe-oF

Target Storage Acceleration Functions

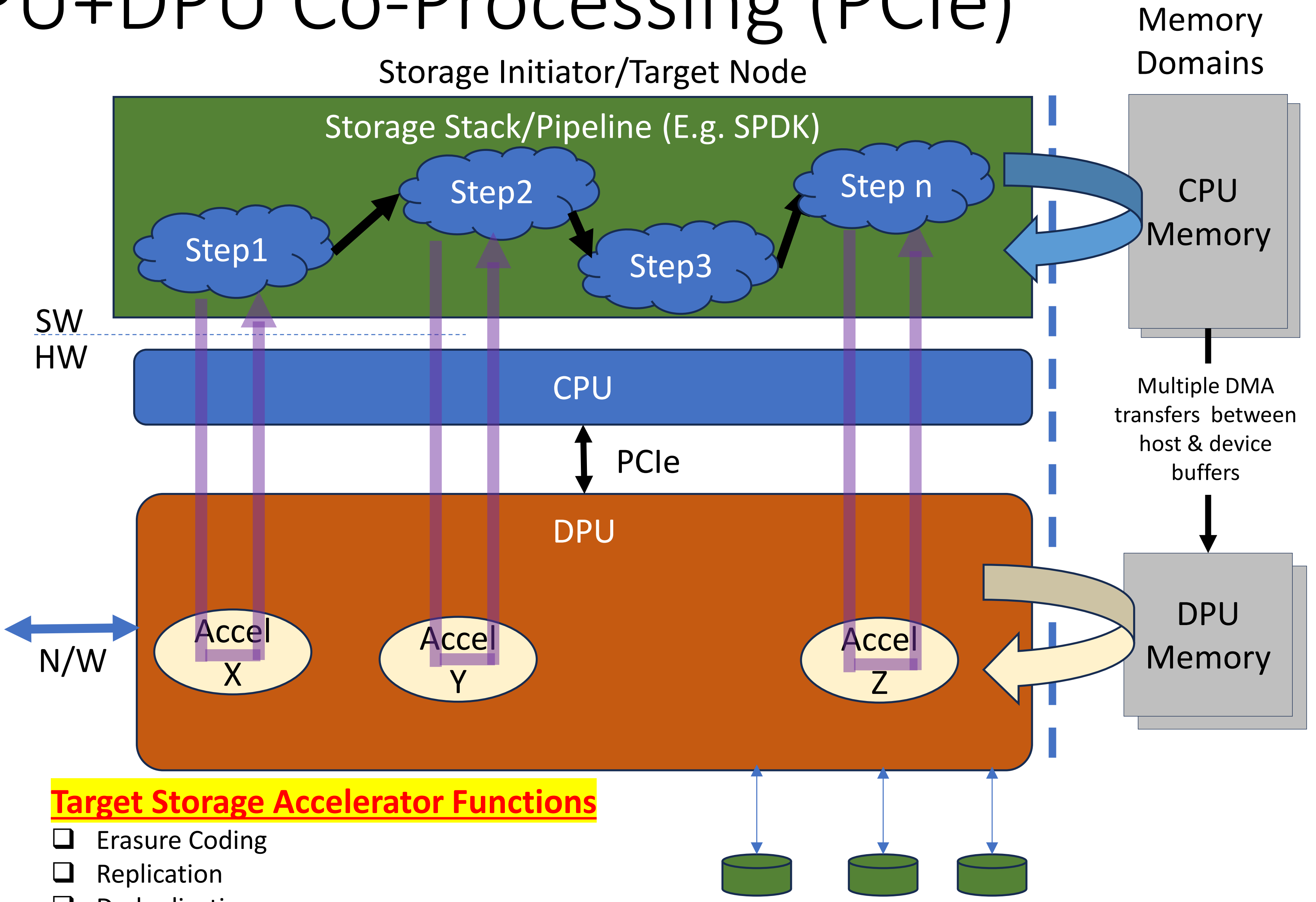
- ❑ Erasure Coding
- ❑ Replication
- ❑ Deduplication
- ❑ Storage Compression
- ❑ Storage Encryption

FMS

Challenges : CPU+DPU Co-Processing (PCIe)

- Storage pipeline control + dataplane processing requiring multistep compute intensive operations requires CPU+DPU co-processing

- CPU+DPU coprocessing using PCIe requires multiple data movements between CPU and DPU memory domains, resulting in significant loss in performance



Memory Domains
Storage Initiator/Target Node
Storage Stack/Pipeline (E.g. SPDK)
Step1
Step2
Step3
Step n
SW
HW
CPU
PCIe
DPU
N/W
Accel X
Accel Y
Accel Z
CPU Memory
Multiple DMA transfers between host & device buffers
DPU Memory

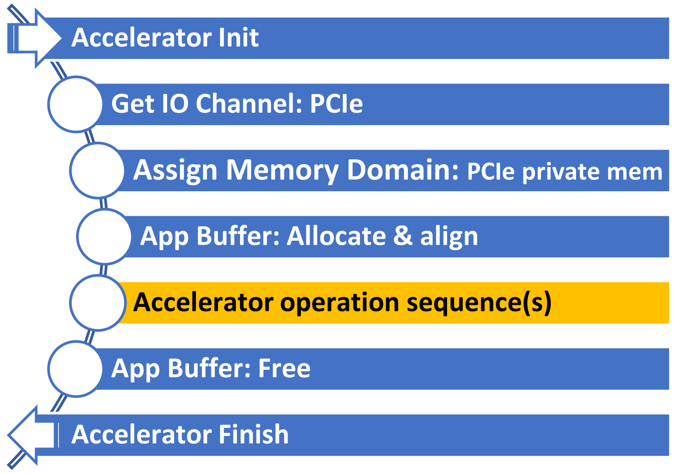
Target Storage Accelerator Functions

- ❑ Erasure Coding
- ❑ Replication
- ❑ Deduplication
- ❑ Storage Compression
- ❑ Storage Encryption

Storage Node:

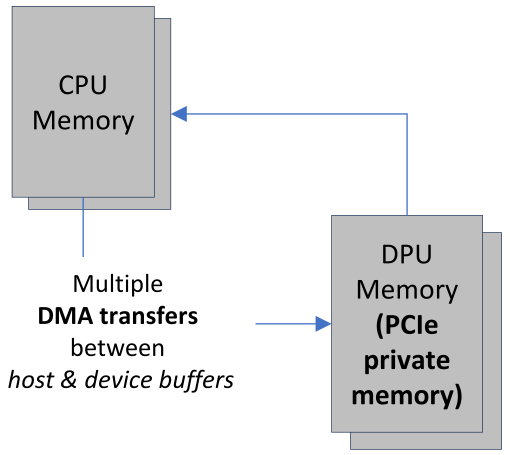
CPU+DPU Co-Processing (PCIe) using SPDK software stack / services

Application usage: Operations



Accelerator Init
Get IO Channel: PCIe
Assign Memory Domain: PCIe private mem
App Buffer: Allocate & align
Accelerator operation sequence(s)
App Buffer: Free
Accelerator Finish

Memory Domains



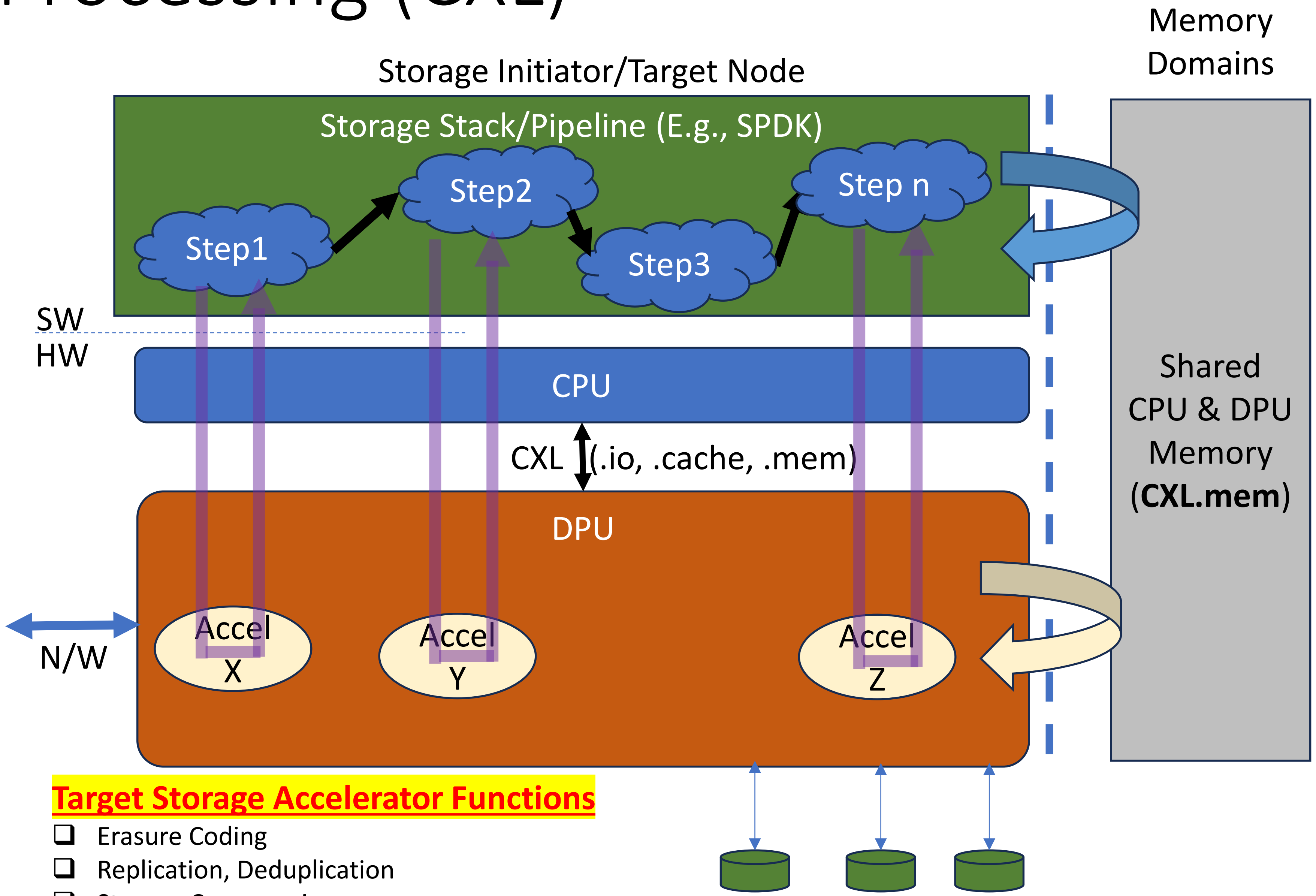
Workflow & Data Structures: **spdk_accel_***

- **initialize**
  - **get_io_channel**
    - **memory_domain**
      - **get_buf; get_buf_align**
        - **operation_exec_ctx; sequence_finish / reverse / abort**
          - **submit_dif_verify / encrypt / compress / xor**
          - **submit_dif_generate / decrypt / decompress**
          - **submit_crc32c / crc32cv**
          - **submit_compare / copy / dualcast**
      - **put_buf**
- **finish**

CPU+DPU Co-Processing (CXL)

Key paradigm shift

- Create single shared memory domain between CPU and DPU
- Use **CXL-attached device memory** (i.e., CXL.mem) as CPU+DPU shared memory
- Avoids explicit data movement between CPU and DPU
- Preserve, leverage existing software stack workflows & datastructure’s


Memory Domains
Storage Initiator/Target Node
Storage Stack/Pipeline (E.g., SPDK)
Step1
Step2
Step3
Step n
SW
HW
CPU
CXL
(.io, .cache, .mem)
DPU
Accel X
Accel Y
Accel Z
N/W
Shared CPU & DPU Memory (**CXL.mem**)

Target Storage Accelerator Functions

- ❑ Erasure Coding
- ❑ Replication, Deduplication
- ❑ Storage Compression
- ❑ Storage Encryption

*https://spdk.io/doc/accel_8h.html#details

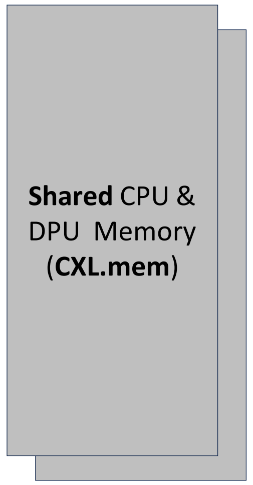
Storage Node:

CPU+DPU Co-Processing (**CXL**) using SPDK software stack / services

Application usage: Operations

- Accelerator Init
- ~~Get IO Channel: PCIe~~
- ~~Assign~~ Use Memory Domain: ~~PCIe private mem~~
- App Buffer: Allocate & align
- Accelerator operation sequence(s)
- App Buffer: Free
- Accelerator Finish

Memory Domains

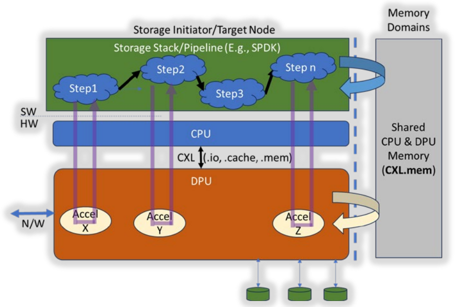


Workflow & Data Structures: **spdk_accel_***

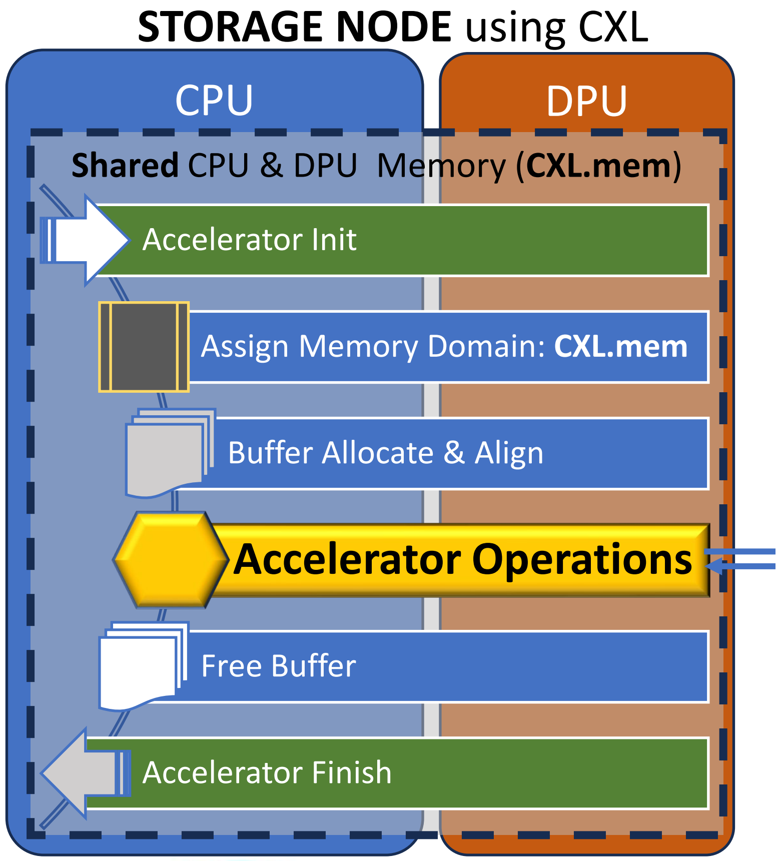
- **initialize**
  - ~~**get_io_channel**~~
    - **memory_domain**
      - **get_buf; get_buf_align**
        - **operation_exec_ctx; sequence_finish / reverse / abort**
          - **submit_dif_verify / encrypt / compress / xor**
          - **submit_dif_generate / decrypt / decompress**
          - **submit_crc32c / crc32cv**
          - **submit_compare / copy / dualcast**
      - **put_buf**
- **finish**

Storage Node:

CPU+DPU Co-Processing (**CXL**) using SPDK software stack

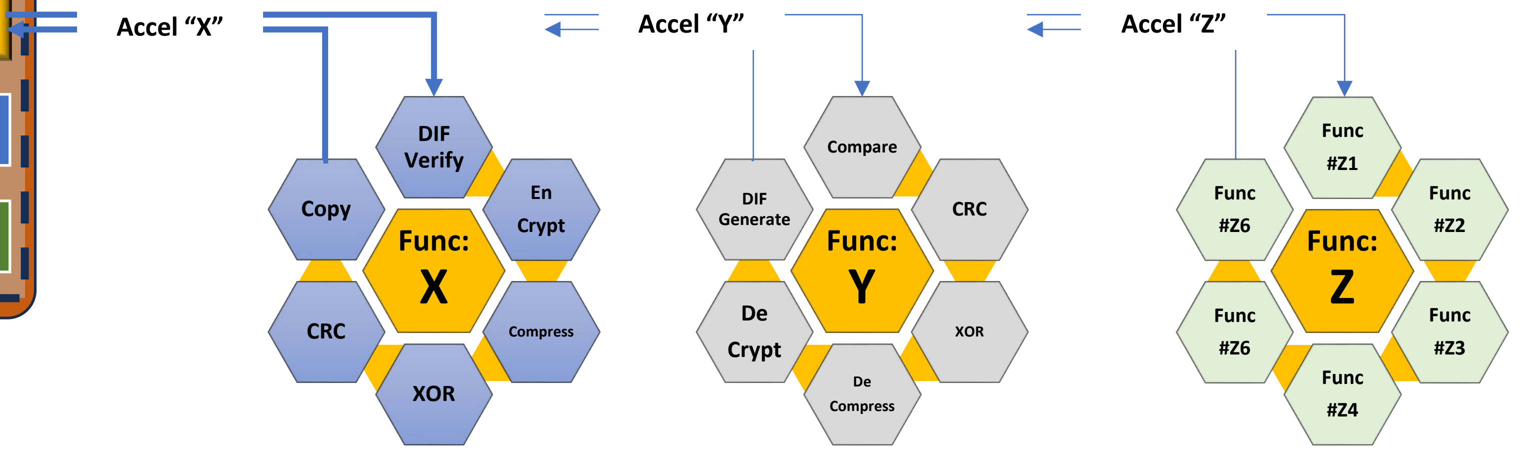


Memory Domains
Storage Initiator/Target Node
Storage Stack/Pipeline (E.g., SPDK)
Step1
Step2
Step3
Step n
SW
HW
CPU
CXL (.io, .cache, .mem)
DPU
N/W
Accel X
Accel Y
Accel Z
Shared CPU & DPU Memory (**CXL.mem**)



STORAGE NODE using CXL
CPU
DPU
Shared CPU & DPU Memory (**CXL.mem**)
Accelerator Init
Assign Memory Domain: **CXL.mem**
Buffer Allocate & Align
Accelerator Operations
Free Buffer
Accelerator Finish

- ✓ Higher IOPS due to simplified Storage data accesses & operations, e.g.,
  - bdev_write: sequence_encrypt + sequence_compress + Storage_write
  - bdev_read: Storage_read + sequence_decompress + sequence_decrypt
- ✓ Preserves Software stack / workflow investments
  - Existing CPU accelerators, newer DPU accelerators can both be leveraged
  - Accelerator operations vs [data segmentation & reassembly and storage transport]

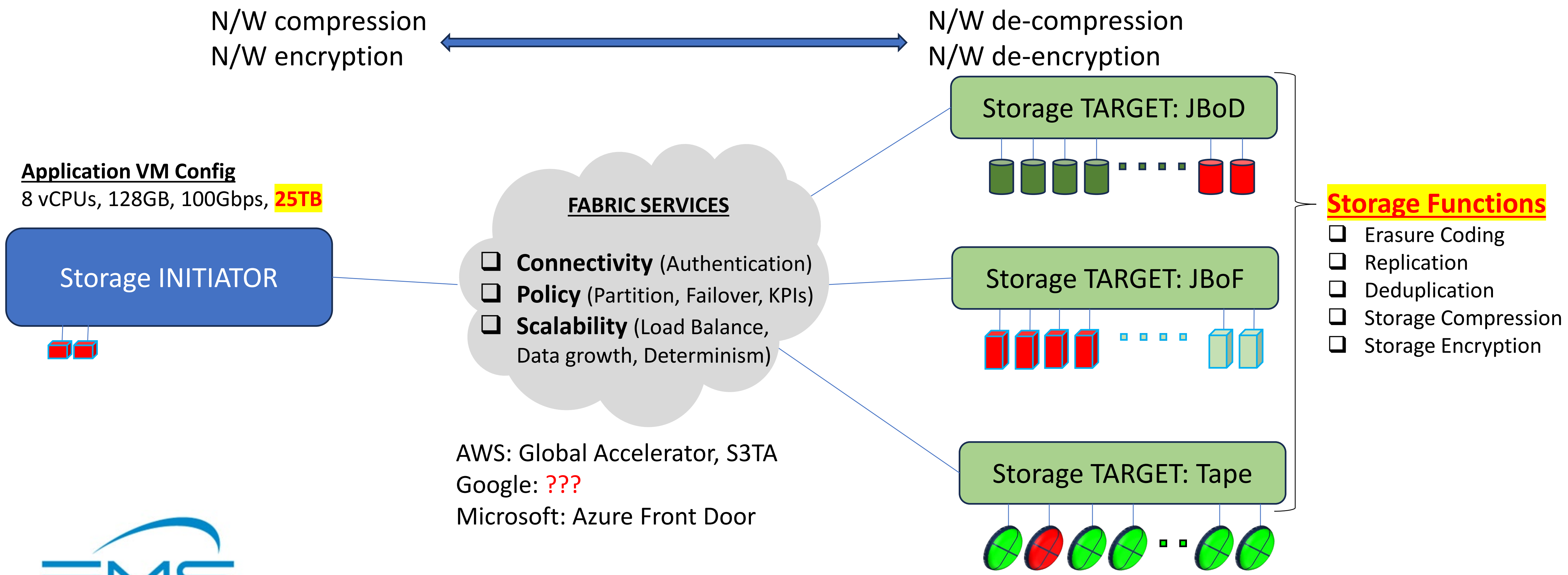


Thank you

- Q&A

Reference / Back up

Deployment Scenarios (e.g., 25TB)



N/W compression
N/W encryption

N/W de-compression
N/W de-encryption

Application VM Config
8 vCPUs, 128GB, 100Gbps, **25TB**

Storage INITIATOR

FABRIC SERVICES

- ❑ **Connectivity** (Authentication)
- ❑ **Policy** (Partition, Failover, KPIs)
- ❑ **Scalability** (Load Balance, Data growth, Determinism)

AWS: Global Accelerator, S3TA
Google: ???
Microsoft: Azure Front Door

Storage TARGET: JBoD

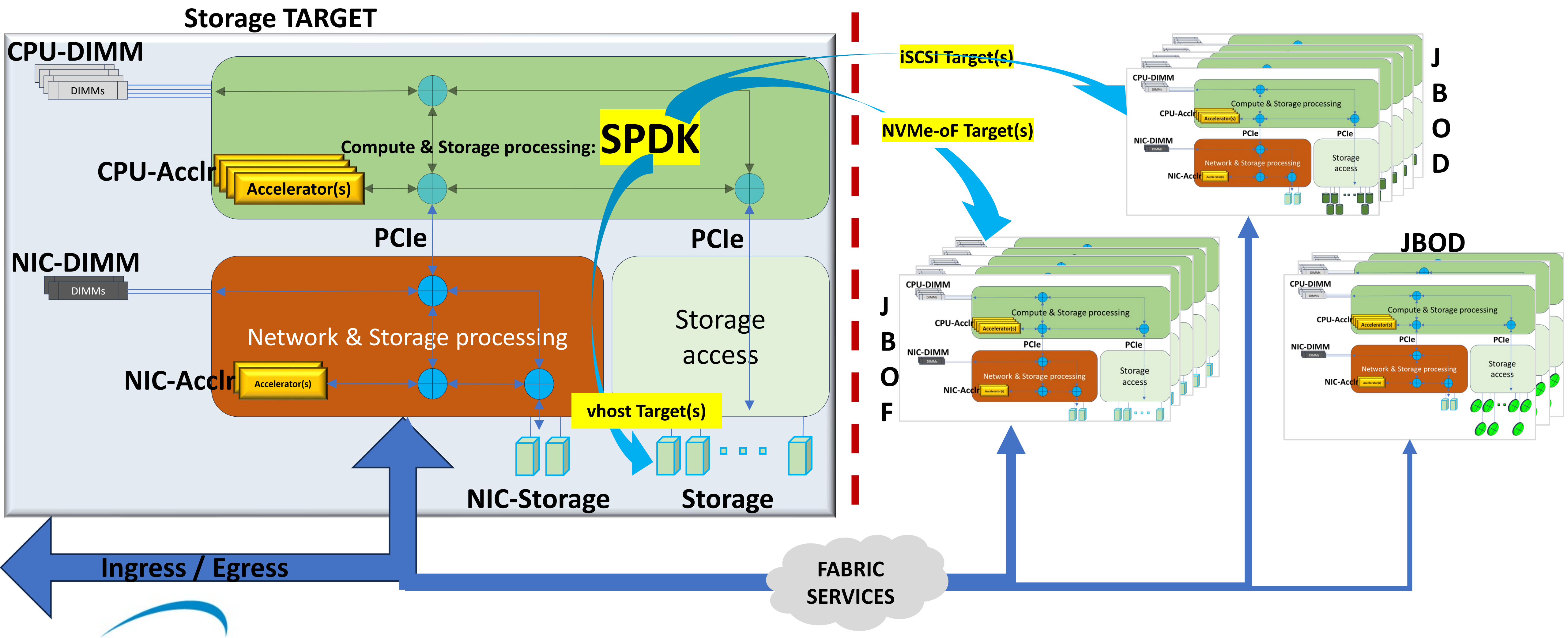
Storage TARGET: JBoF

Storage TARGET: Tape

Storage Functions

- ❑ Erasure Coding
- ❑ Replication
- ❑ Deduplication
- ❑ Storage Compression
- ❑ Storage Encryption

Implementation Scenario (e.g., SPDK)


Storage TARGET
CPU-DIMM
DIMMs
Compute & Storage processing: SPDK
CPU-Acclr
Accelerator(s)
PCIe
PCIe
NIC-DIMM
DIMMs
Network & Storage processing
NIC-Acclr
Accelerator(s)
Storage access
vhost Target(s)
NIC-Storage
Storage
iSCSI Target(s)
NVMe-oF Target(s)
JBOD
JBOF
JBOD
CPU-DIMM
Compute & Storage processing
CPU-Acclr
PCIe
NIC-DIMM
Network & Storage processing
NIC-Acclr
Storage access
Ingress / Egress
FABRIC SERVICES