SNIA DEVELOPER CONFERENCE
SDC 25
By Developers FOR Developers

Hyatt Regency Santa Clara, CA
September 15-17, 2025

SSD Virtualization: Enabling New Possibilities for SSD-Based Ecosystems

Dan Helmick, PhD
Principal Engineer, Samsung Semiconductor
Controller Architecture Team

www.sniadeveloper.org

Agenda

- Virtualization Background
- Highlighted New Virtualization Features and Example Use-Cases
- Potential Future Directions

What is Virtualization of Storage?

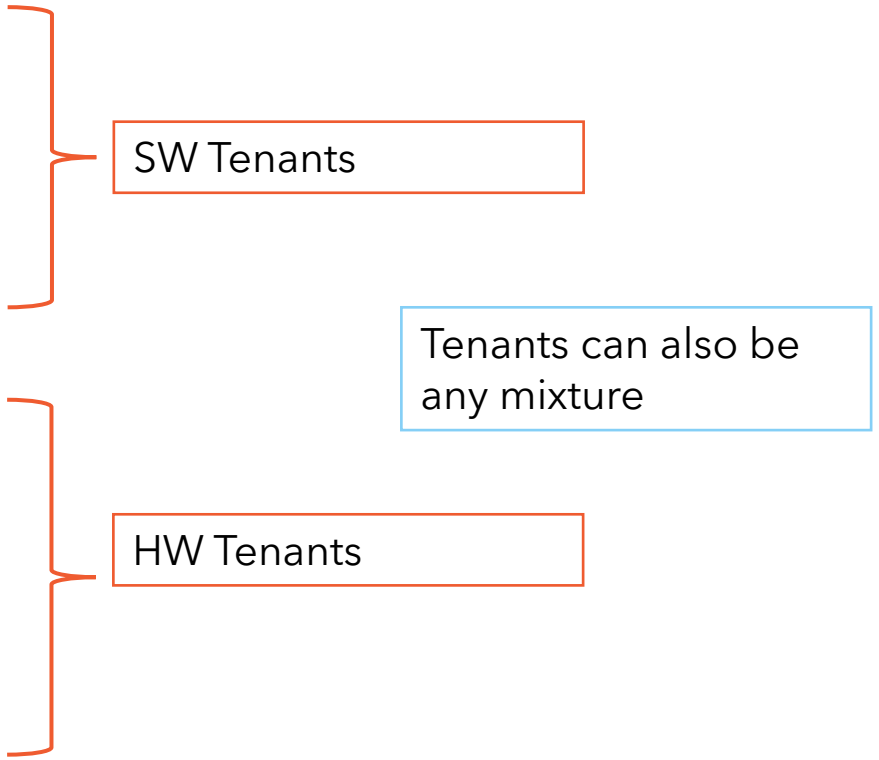
- Defining Storage Virtualization
  - Presenting access to Stored Data (Ex: on an NVMe™ SSD) with access attributes that differ from initial conventional access
- Example items that can be virtualized
  - PCIe® layer attributes
  - NVMe Controller and identifying data
  - Data separation
  - Storage administration and log pages
  - Namespace and LBAs
  - Security boundaries and key management
- Virtualization enables a limited or replaced view of storage access that may be provided to an end user.

Some Motivations

- Present a large capacity SSD as several smaller SSDs
- Limit SSD command availability to unprivileged users
- Isolate the performance impacts from several storage users
- Sandbox bad-actors to limit their data access
- Confidential Compute enablement
- Make a new SSD perform similar to the SSD it replaces

Defining a Tenant

- Tenant
  - Any host entity that can be identified as a separated source of traffic for the SSD
- Example Tenants
  - SW Process
    - Ex: File System (FS) can be separated into end user Reads, end user Writes, FS Reads, FS Writes, etc.
  - Application (App)
  - Container
  - Virtual Machine (VM)
  - CPU Processor
  - CPU Core(s)
  - CPU Thread(s)
  - FPGA
  - GPU
  - GPU partition



Some SSD Building Blocks for Virtualization and Scalability

- PCIe®
  - Dual port
  - TDISP Components and Protocols
    - PCIe IDE
    - SPDM
  - SR-IOV
- NVMe™
  - Resource Management of Primary and Secondary Controllers
  - NVMe-oF™ Scalable Resource Management
    - Creating Isolation
    - Breaking up large storage systems into manageable pieces

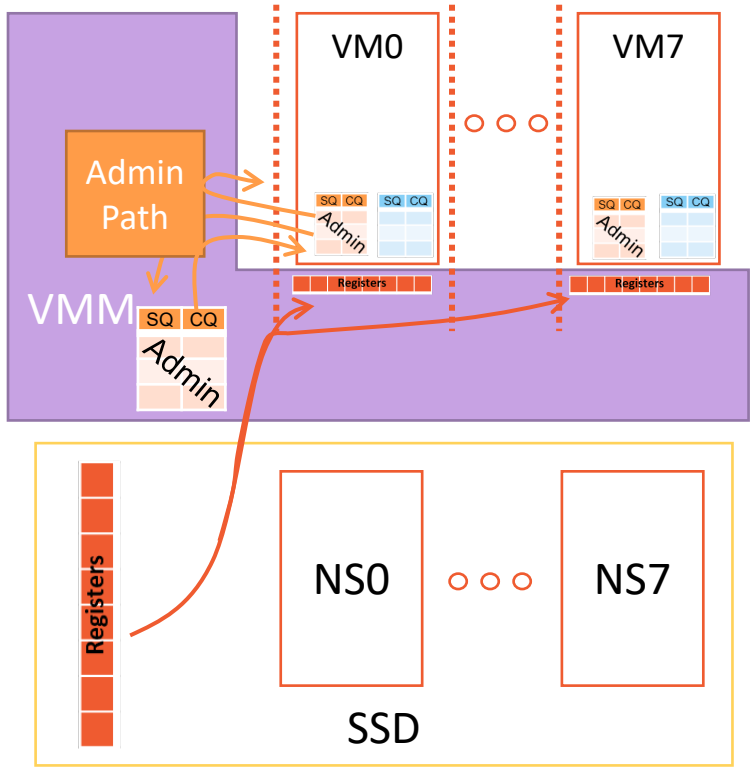
TDISP has many useful building blocks

These are some of the major features that were pre-existing or enabled concurrently by additional companies.

NVMe-oF has many useful building blocks

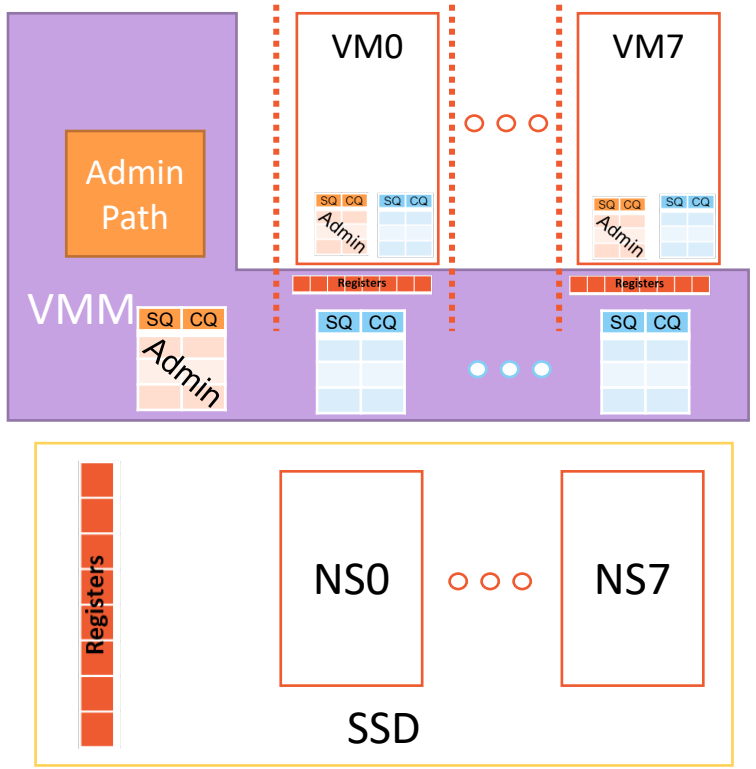
How has Virtualization been Done?

- SW Implementation!
  - Example: Virtual Machine Manager (VMM)
- Creates SW boundaries for VM
- Curated view of Registers presented to each VM
- Admin commands intercepted for
  - Local response from VMM only
    - Ex: Responding a capability is not supported
  - Pass through to SSD's Admin command
    - Ex: Create NS
  - Filtered pass through to SSD's Admin command
    - Ex: Query NUSE
- IO commands
  - Doorbells intercepted
  - Minimal SQE inspection and manipulation capabilities



SW Virtualization

- Advantages
  - Virtualization is enabled
  - VMM's virtualization can be updated
  - Exotic/Advanced features like per VM encryption could be enabled
- Challenges
  - SW layer consumes CPU cycles
  - Delays – Pass through commands must be copied, and Doorbells repeated.
  - Attack vector for bad actors through VMM
  - Generally incompatible with Confidential Computing
  - SSD has no knowledge it is being virtualized
    - Loss of differentiation between tenants within the SSD



Moving Virtualization to SSD

- Performance/Latency requirements by customers are tightening
- Virtualization needs have stabilized
- Security requirements are increasing (often moved into HW)

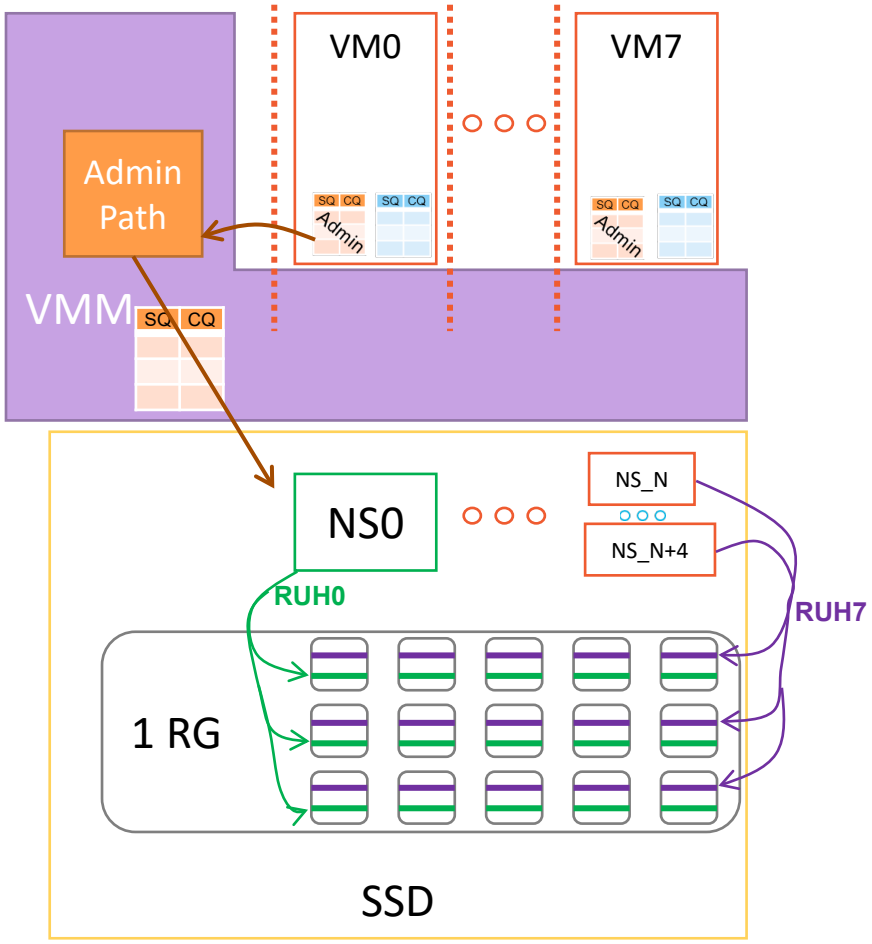
Highlighted New Virtualization Features

- Flexible Data Placement (FDP)
  - TP4146 Flexible Data Placement
- Tracking LBA Allocation
  - TP4165 Tracking LBA Allocation
- Live Migration (LM) of a Namespace (NS)
  - TP4159 PCIe Infrastructure for Live Migration
- LM of a Controller
  - TP4159 PCIe Infrastructure for Live Migration
  - TP4193 PCIe Exported NVM Subsystems Migration

Note: TP4193 has not been ratified. Information here is based off of Mike Allison's presentation at Future of Memory and Storage 2025.

Setting up FDP to Separate Tenant Data

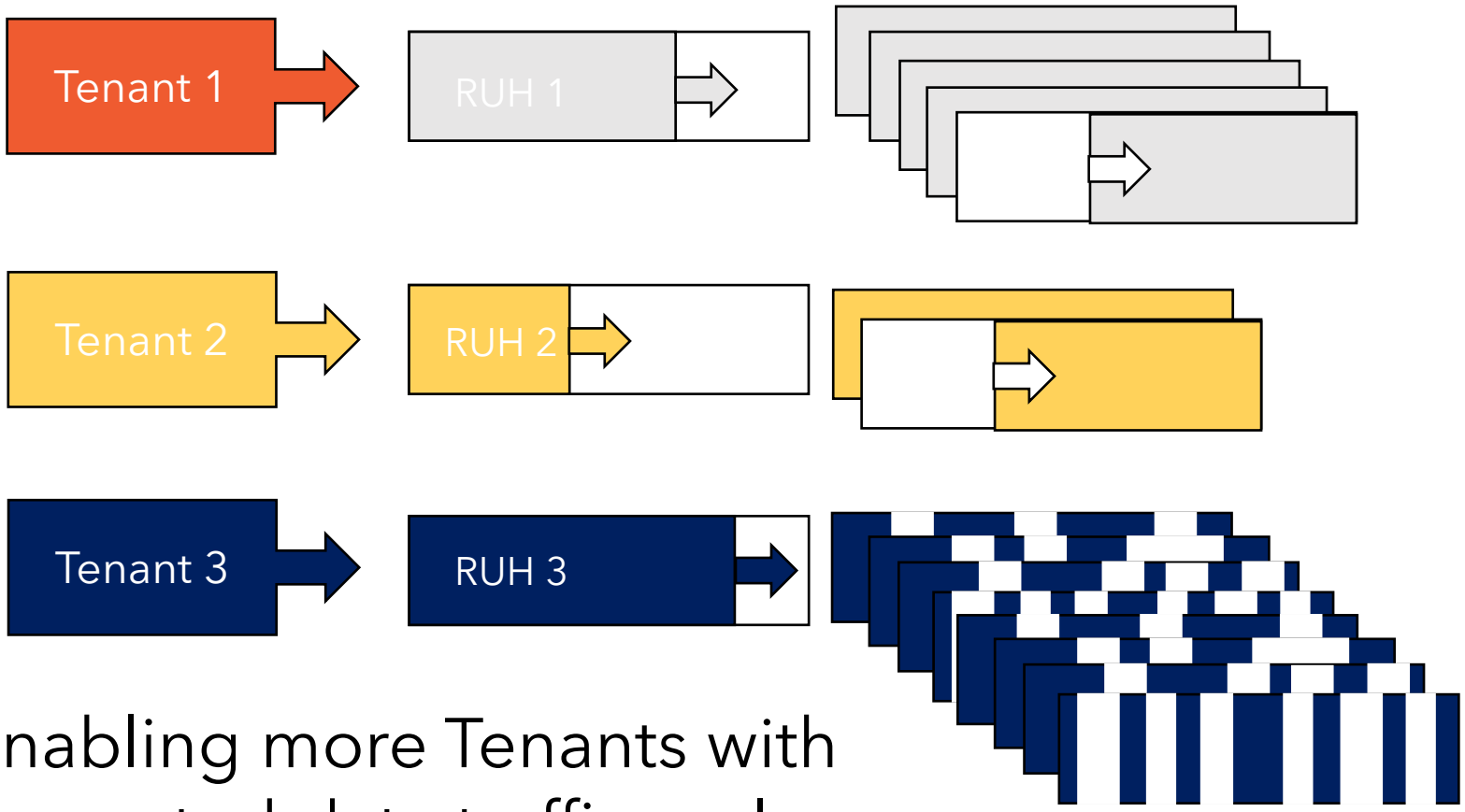
1. SSD is initialized with 1 Reclaim Group (RG)
2. VM sends NS Management "Create NS" command
3. VMM captures this command
4. VMM populates the default Placement Handle 0 with a different Reclaim Unit Handle (RUH)
5. VMM sends "Create NS" command to the SSD
   - ➢ Potentially each VM could be set up with a different RUH
   - ➢ Multiple NSes may share the same default RUH



- Separation of data separates the WAF of tenants.
  - Each tenant can optimize WAF for performance improvements
- Sequential Reads and Writes may be optimized per tenant
- Trims can be accelerated

Using Flexible Data Placement for Virtualization

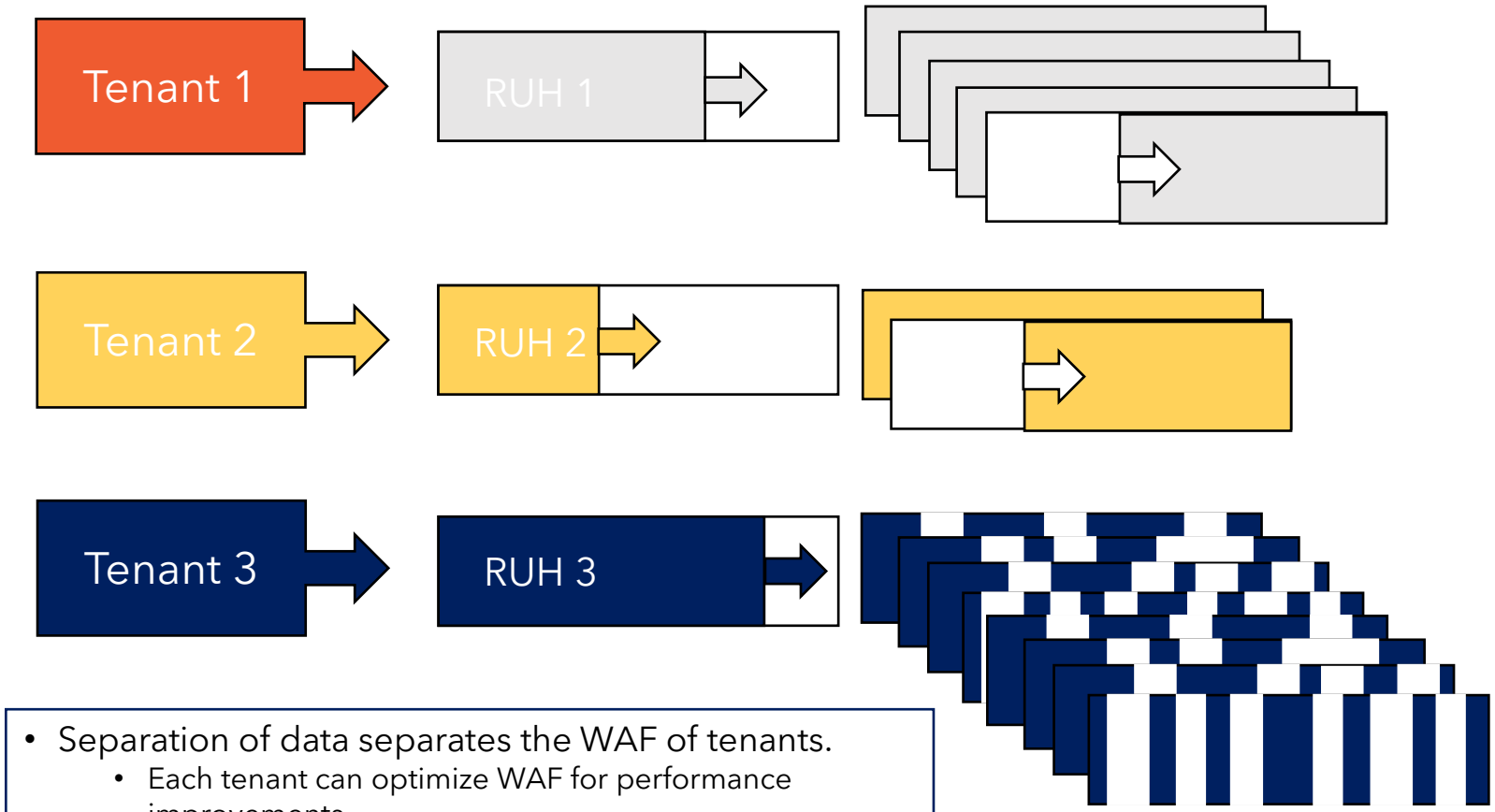
- Tenant 1 and 2
  - Writing sequential circular buffers
  - WAF=1 traffic
- Tenant 3
  - Random write traffic
- Reads
  - Prioritized traffic
  - Reads from all tenants will suspend programs/erases
- Writes
  - Max Performance*** achievable for each tenant is directly related to their WAF
  - *** Assuming other Tenants are idle



Enabling more Tenants with separated data traffic scales the usage of an SSD

Using Flexible Data Placement for Virtualization

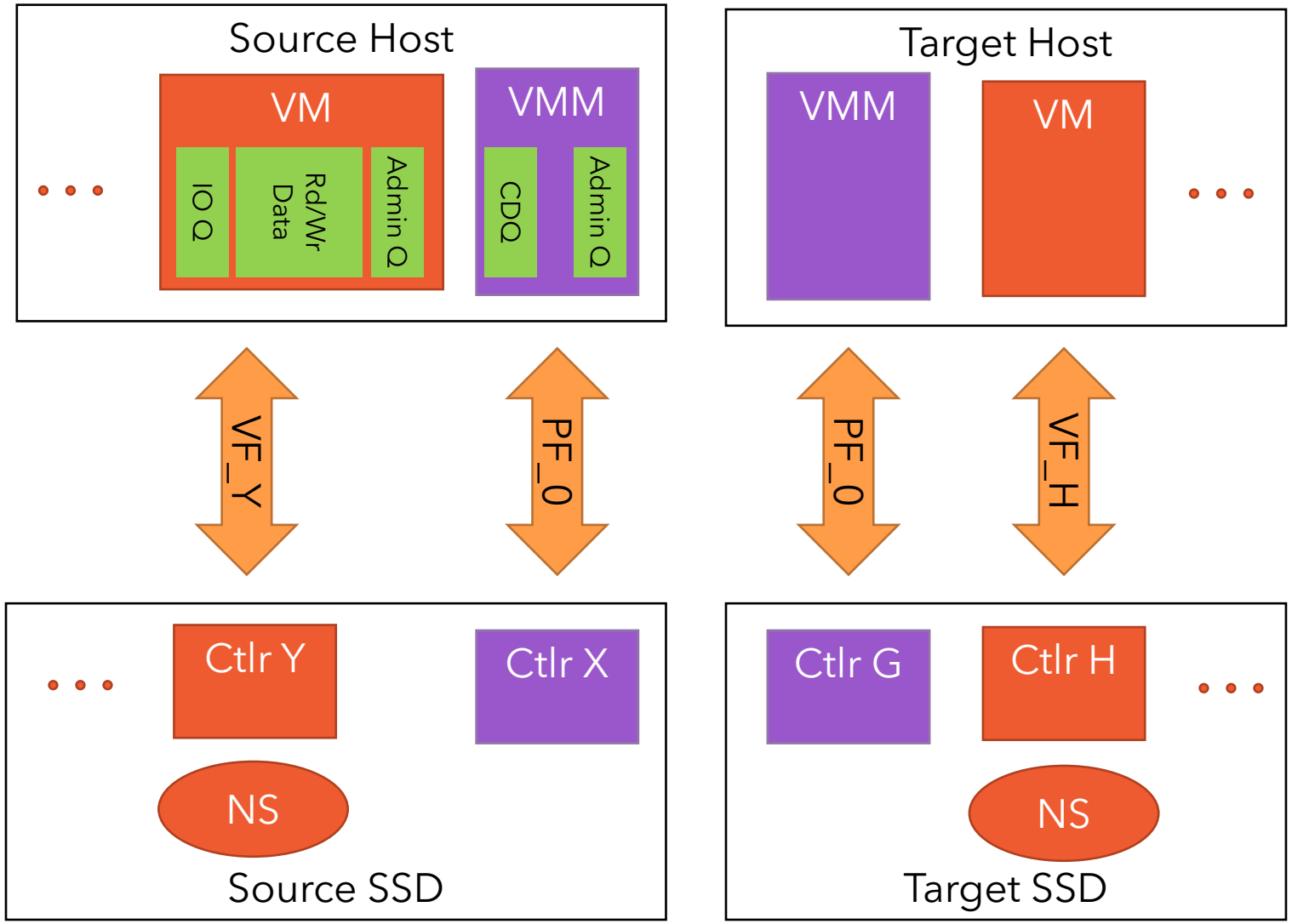
- Mixed Tenant Traffic
  - Data traffic time is interleaved between tenants
  - An over consuming Tenant can reduce the achievable performance of other Tenants
- Examples
  - Only Tenant 1 is writing
    - 100% Sequential Write Performance
  - Only Tenant 1 and 2 are writing
    - 50% Sequential Write Performance each
  - All Tenants are writing
    - Tenant 3 swamps the drive with GC
    - Tenant 1 and 2 will get 10-20% of Sequential Write Performance



- Separation of data separates the WAF of tenants.
  - Each tenant can optimize WAF for performance improvements
- But without limiting the media access time
  - Incomplete toolset for an SSD to protect 1 tenant from another's performance impacts

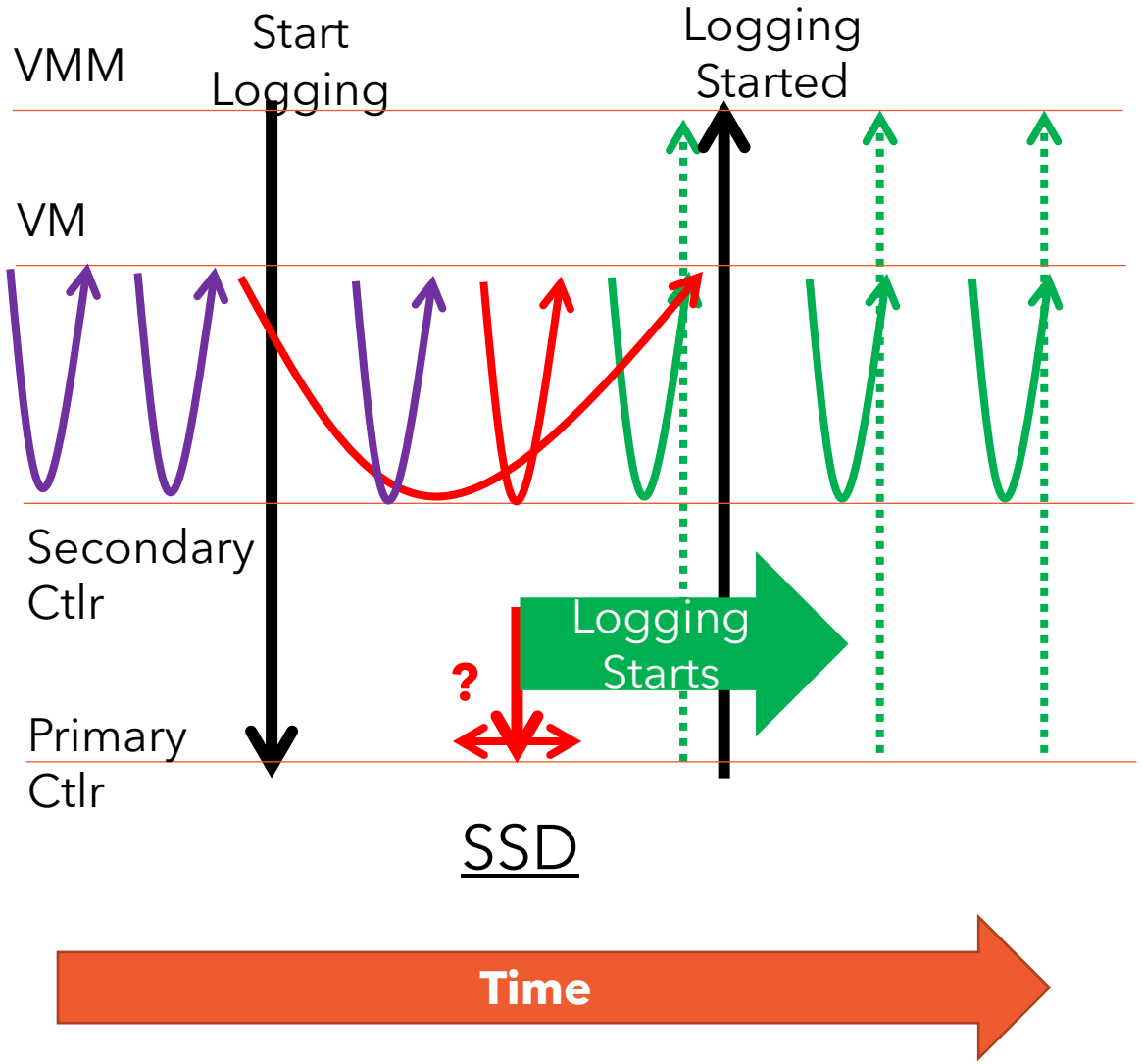
Data Center has Many VMs Deployed

- Enabling LM of a VM enables scalability for the Data Center
  - Redistributes system load, Route around failed resources, etc.
  - VM must be unaware LM is happening
- SSD example with SR-IOV
  - Primary Controller (Ctlr) per VMM on PF_0
  - Secondary Ctlr per VM on VF_Y and VF_H
- Virtual Machines (VMs) and VM Monitor (VMM)
  - 1 VMM to many VMs
  - VMM may decide a VM needs to move
- Target vs Source
  - Similar setups



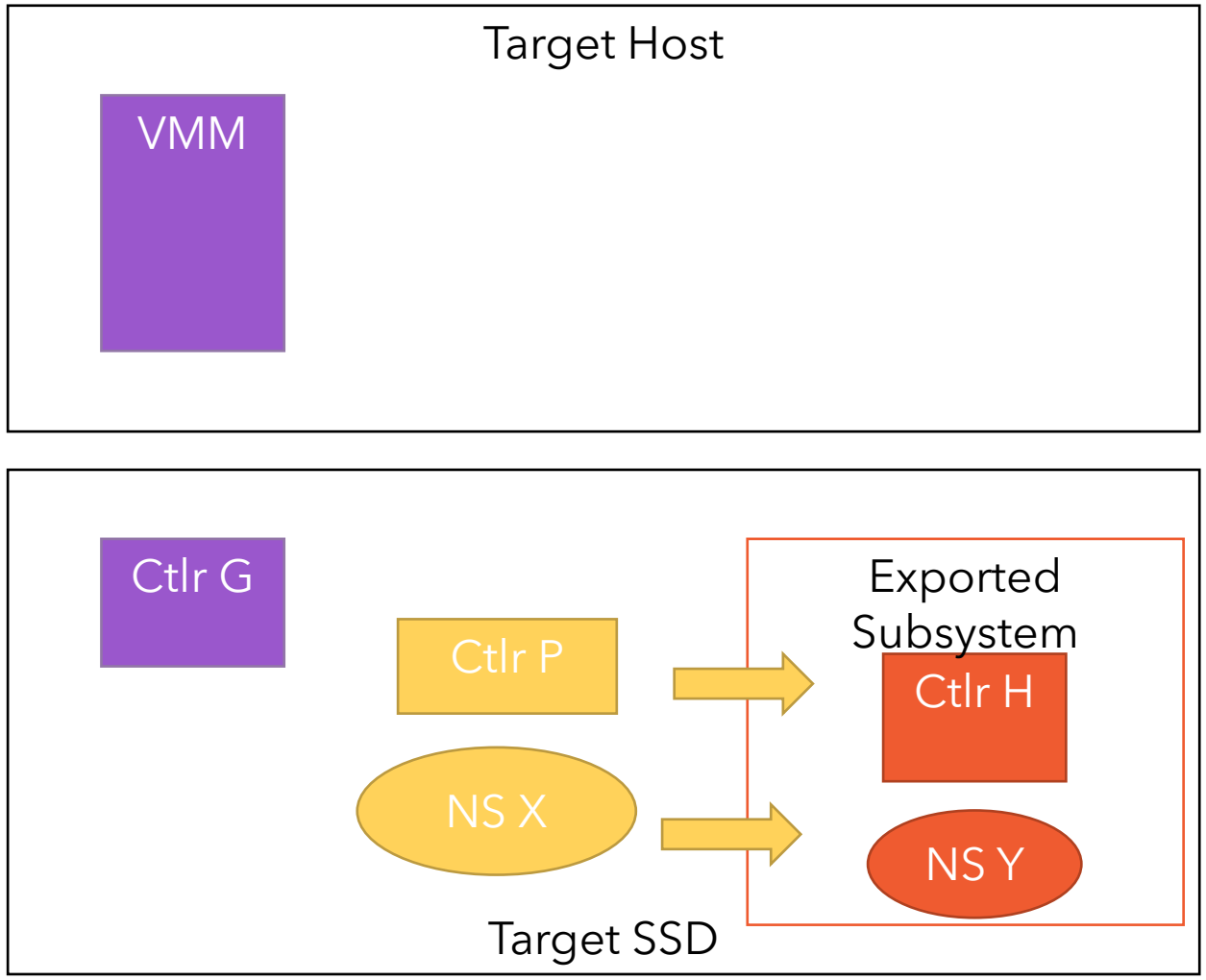
Pre-Copy Phase: Start Logging

- VM continues to interact with Secondary Ctlr on SSD (Rd/Wr)
  - **Race Conditions** are a concern
- "Start Logging" Command Flow
  - Ongoing VM IOs
  - VMM sends "Start Logging" Command
  - Primary Ctlr begins tracking all requested CDQ events occurring in VM's Ctlr (Secondary Ctlr)
    - Some commands in flight may be logged (excess logging is allowed)
    - Some commands in flight may not be logged
  - Primary Ctlr completes "Start Logging" Command
    - SSD Promise: All potentially log-able commands will now be logged
- VMM has successfully started logging in CDQ
  - Relationship of Logging Start and some commands is unknown
  - Unknown timing of where Logging Start occurred with respect to Completion of Start Logging command
  - "Logging Started" ensures
    - All prior commands in flight have finished
    - All future commands in flight will be logged



Pre-Copy Phase: Export an NVM Subsystem and Initialize

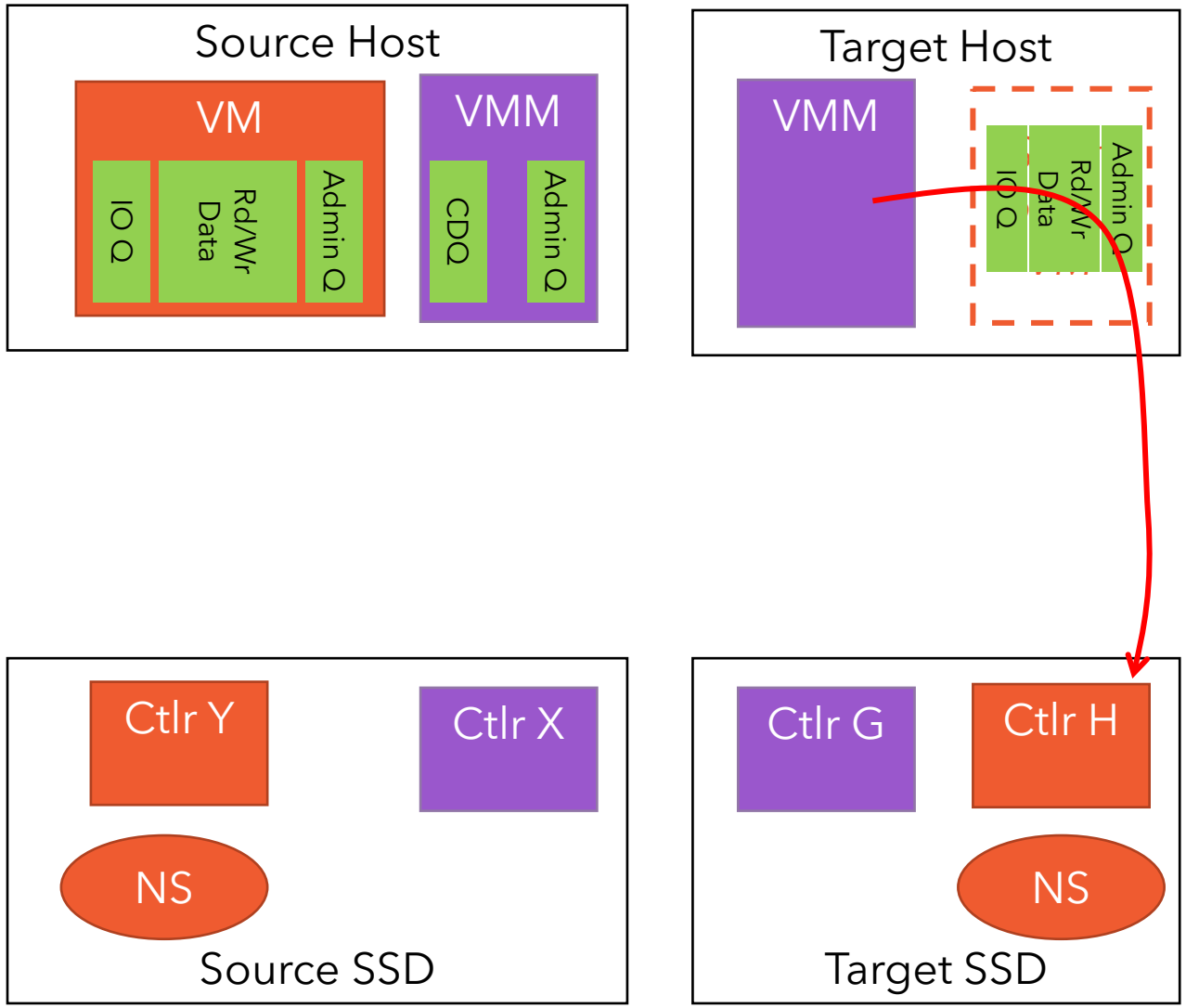
- TP4193 PCIe Exported NVM Subsystems Migration
  - Has not been ratified, but most recent information is here at SDC 2025!
  - "Host Management of NVM Express Exported NVM Subsystems in PCIe SSDs"
  - Presentation by Mike Allison and Lee Prewitt
  - Monday at 8:30-9:20AM
- Initial System
  - Primary Controller G managed by VMM on the Host
  - Underlying components (Ctlr P and NS X) are available
- Create an exported subsystem
- Associate underlying Controller and Namespace
- Initialize their virtualized reporting information
  - Exported NVM Subsystem could potentially be reporting any information
  - Ex: Use this feature to make the NS ID match on Source and Target system



Note: Later slides have insufficient space. Underlying Ctlr P and NS X are not shown later.

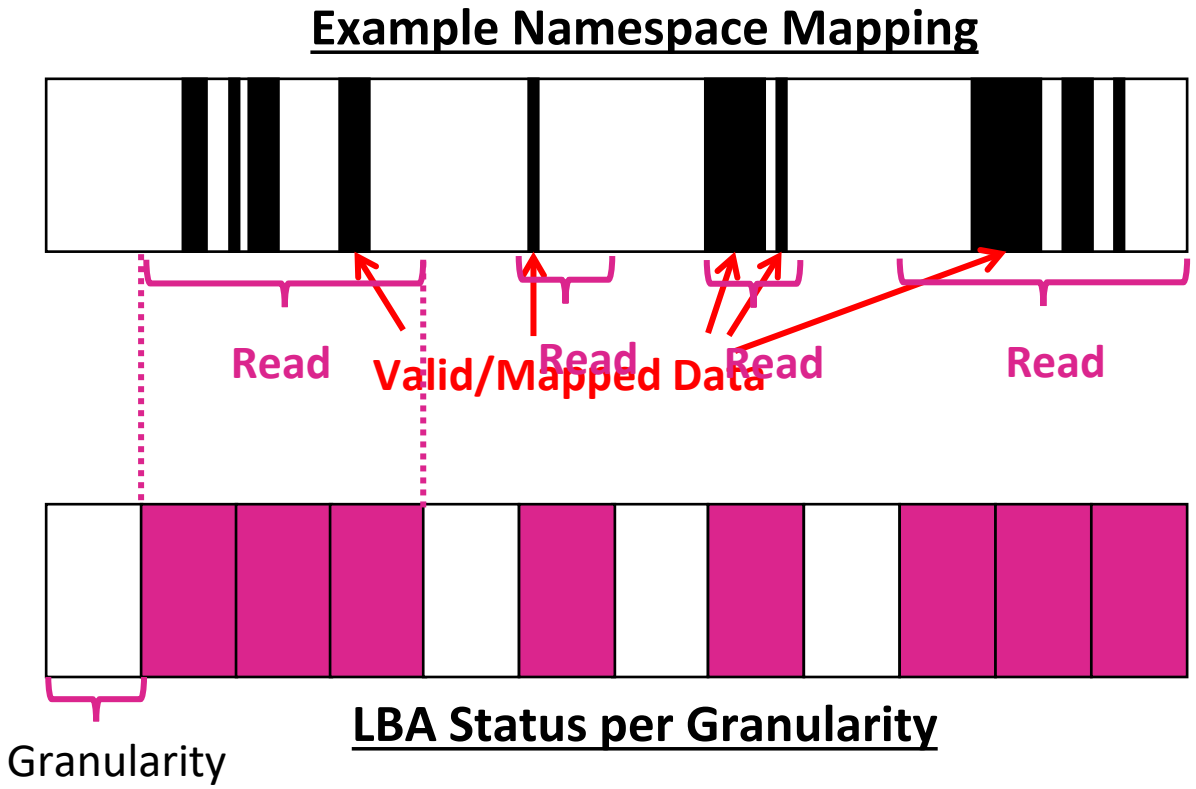
Pre-Copy Phase: Target Preparation

- Target Precondition
  - Available Secondary Controller and Namespace exported and initialized Available Host side VM resources
- Standard NVMe commands for initializing Target SSD
  - Initialize any Queue and IO command structures needed
  - Write data to NSes
- Above illustrates one potential flow, but other options exist
  - Ex: Shared NS created by VMM on Ctlr G



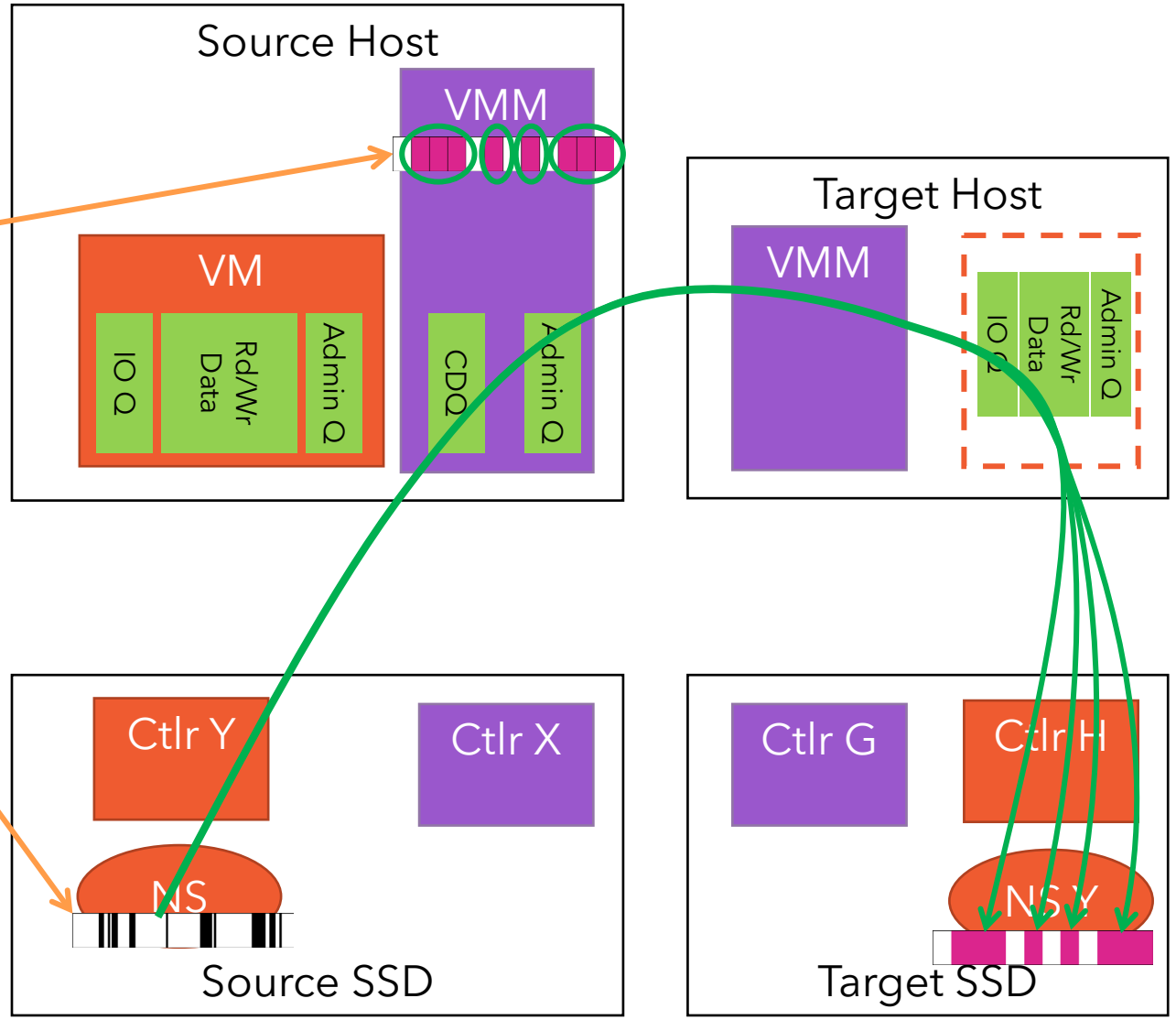
Pre-Copy Phase: Initial NS Migration

- Option 1:
  - VMM copies entire VM NS
    - Not optimal for sparsely written data
    - *<See example on right>*
- Option 2:
  - VMM sends Primary Ctlr: Get LBA Status
    - Granularity: Set by SSD
      - Customer requirements discussion
  - Primary Ctlr
    - Returns results with granularity restrictions
    - Any data state other than deallocated is returned as mapped
      - Ex: Read Uncorrectable
  - VMM
    - For each mapped LBA status
    - Submitted as Read of Child's NS



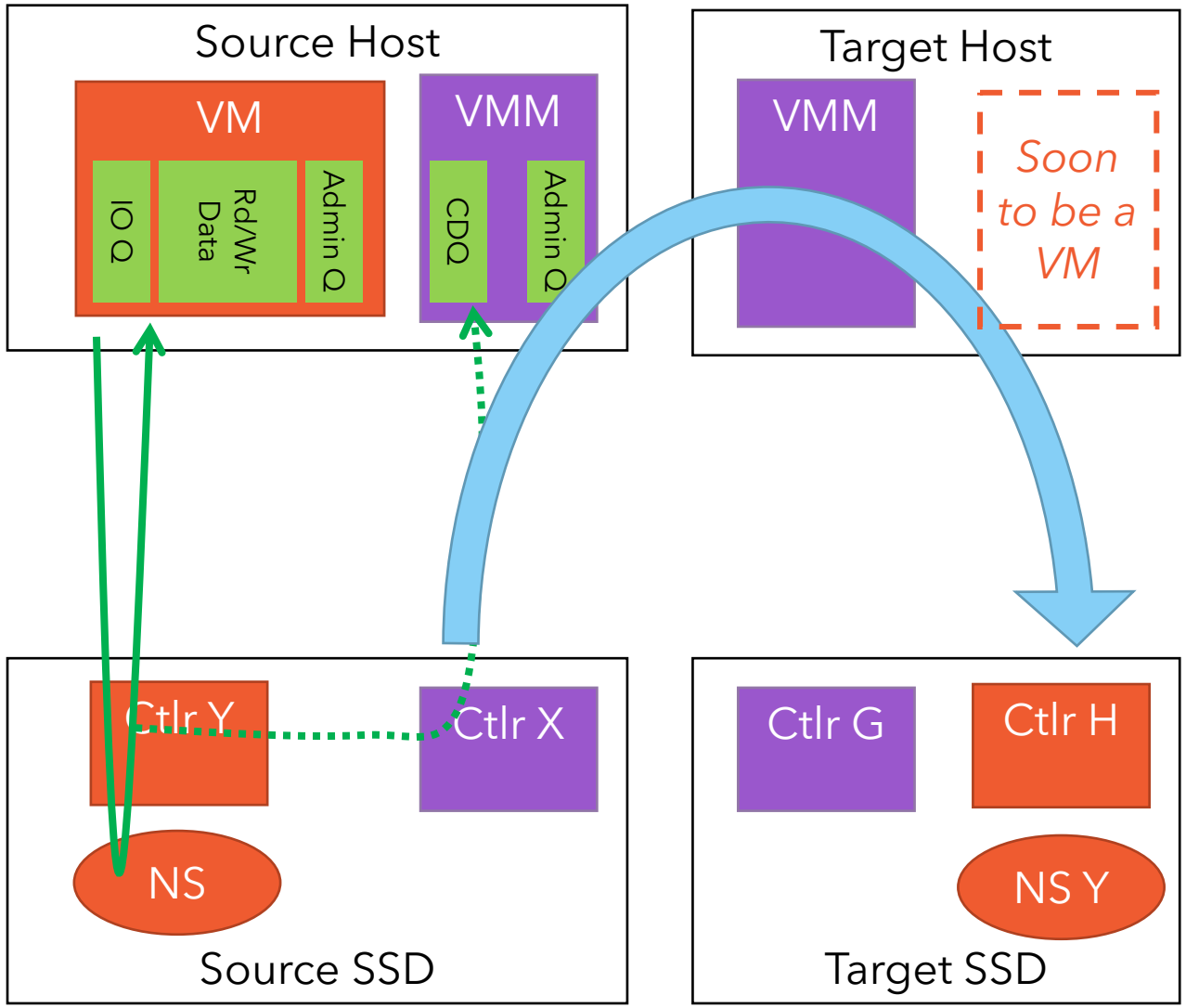
Pre-Copy Phase: Initial NS Migration to Target

- VM's NS Mapping
- Returned LBA Status per Granularity
- VMM submits Read to Child's NS for each contiguous mapped LBA range
- New NS is populated with no dependence/knowledge of Source SSD's granularities



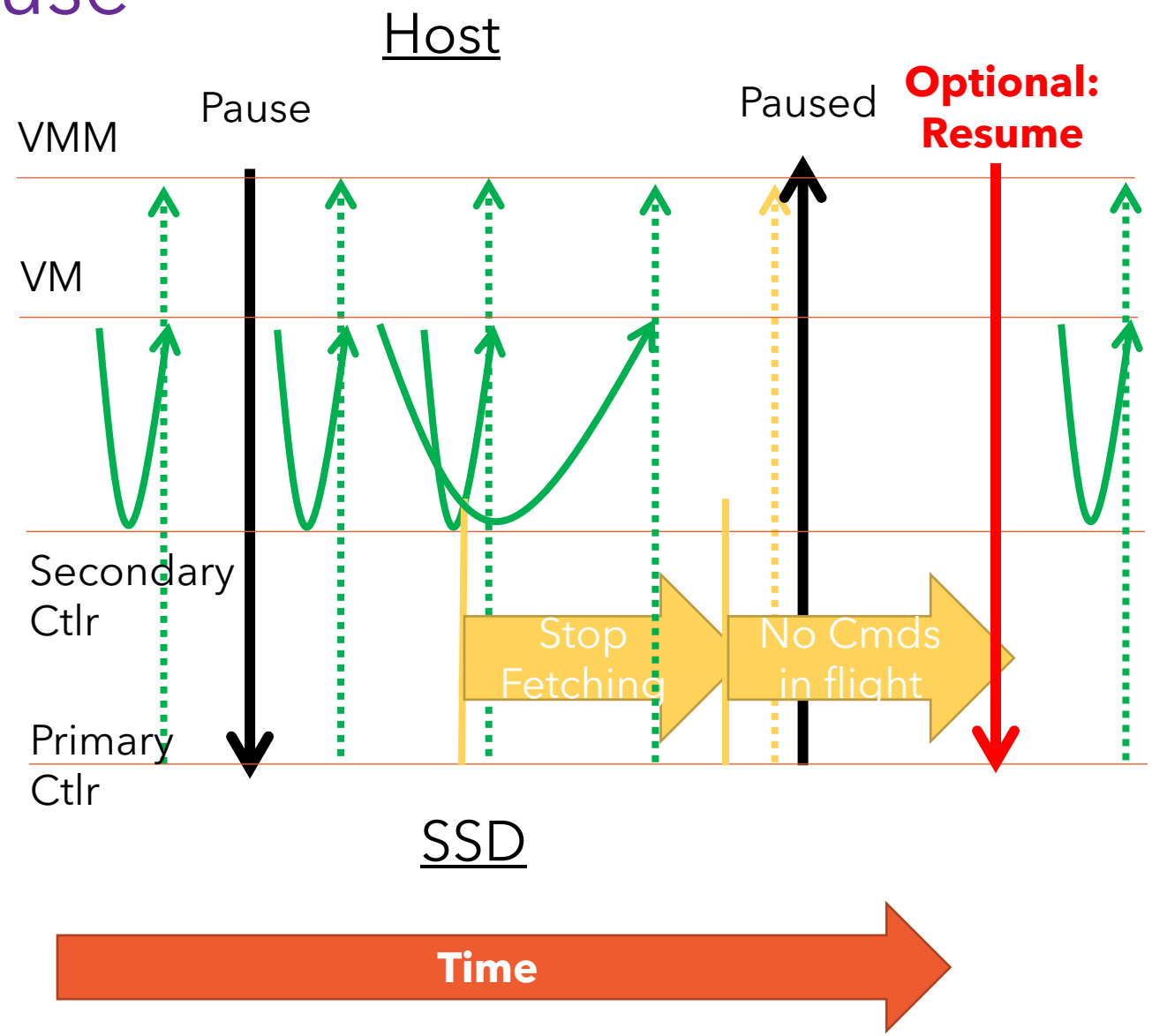
Pre-Copy Phase: Iterative Data Copy

- Ongoing
  - VM has continued to Rd/Wr to Source NS
  - Source Primary Ctlr X has continued to log all appropriate activities to VMM
  - Copying from Source SSD to Target SSD takes time
- Source Drive View
  - Has experienced Reads from initial copy of Source NS to Target NS
  - Continues to experience more Reads from VMM parsing CDQ logs
    - VMM is continuing to catch up to the VM's activity
  - Data is written to Target Child NS



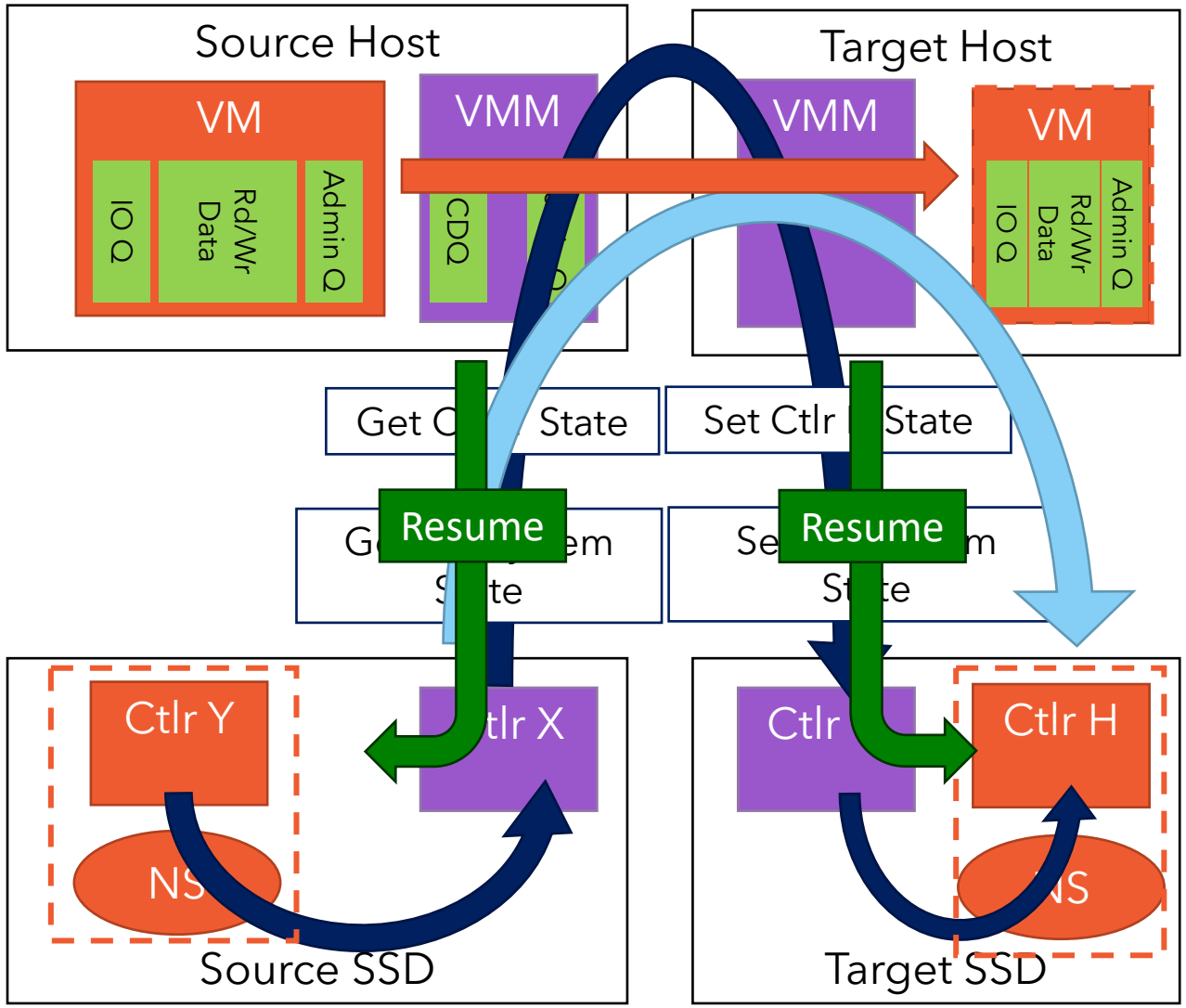
Stop and Copy Phase: Pause

- VMM decides to complete/execute the migration
  - VMM issues Pause Command to Primary Ctlr
- "Pause" Command Flow
  - Secondary controller stops fetching new commands
  - Secondary controller completes all commands in flight
    - Success vs Error are both acceptable
  - All CQEs are properly returned to VM
    - With any CDQ entries for logging
  - Primary Ctlr completes the Pause command to VMM
    - And may concurrently log this successful pause in the CDQ
- Stopped status Summary
  - SQE/CQEs may be on the SQ/CQs of the VM
- Source SSD
  - Must be prepared for potential Resume Command
    - Perhaps due to a system error
    - Conceptually Resume/Start should behave the same on both Source and Target
    - Except: Source SSD would continue logging
  - If not resumed, expect Secondary Ctlr to be reset.
- VMM will
  - Parse all remaining CDQ entries
  - Copy any remaining data to Target Child NS



Post-Copy Phase: Copy Final Data and Migrate Controller State

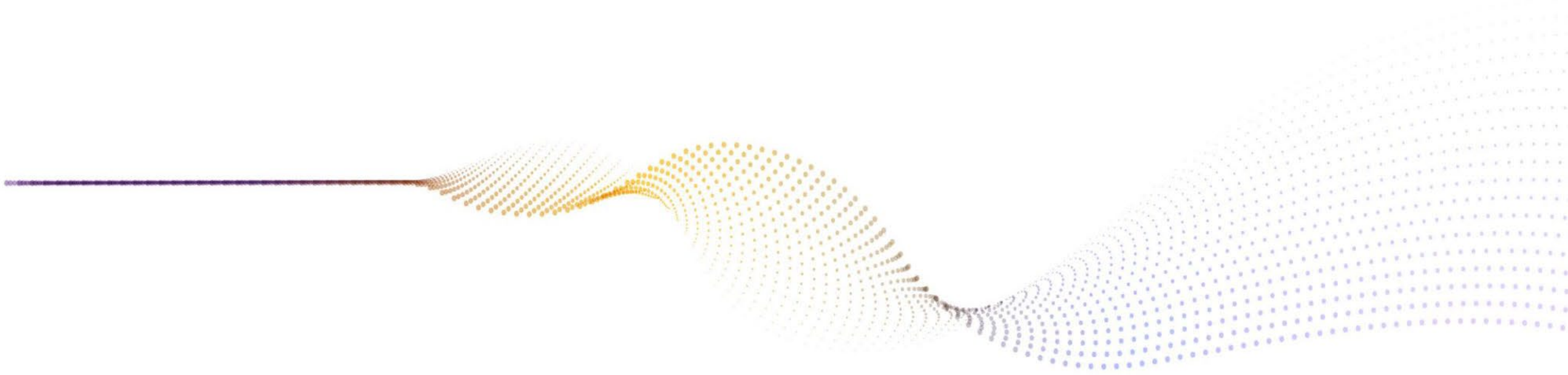
- Final Data Copy Iterations from CDQ Parsing
- Finalize Exported NVM Subsystem State
  - Reads Final Source NVM Subsystem State out to the VMM
  - VMM writes Final NVM Subsystem State to Target SSD
- Finalize Controller State
  - Gets Ctlr Y state out to the VMM
  - VMM Writes Ctlr H state into the Target SSD
- VMM will migrate the VM
- From SSD's view
  - Same behavior:
  - Resume Ctlr Y sent to Ctlr X
  - Resume Ctlr H sent to Ctlr G
  - One difference: unlikely Ctlr G has enabled logging on Ctlr H
- Nominal NVMe Flows
  - Source VMM will clean up and reset Source Child
    - Source Exported NVM Subsystem
    - Ctlr Y
    - NS



Potential Future Virtualization and Scalability Features

Where do we go from here?

- Performance
  - Quality of Service (QoS)
    - Rate Limiting Performance management provided by a Controller
  - Prioritization of 1 Tenant over another
    - Focused IO consistency
    - Impactful at low QD
- Security
  - KPIO
    - Keep data encrypted during LM
  - OCP's Caliptra and LOCK
    - Keep data encrypted during LM
  - TDISP extended into the SSD
    - Enable boundaries and confirmation of status
- Extensibility
  - Extended definition of a QoS Tenant
    - Enable Storage and NS differentiated use-cases
    - Ex: SQ Groups rather than a Controller
  - sIOV
    - Move from 10's-100's of Tenants to 1,000+ Tenants



Thank you for attending!

Please remember to rate this session. You get access the presentations at
http://sniadeveloper.org/conference

