Persistent Memory

Agenda

- Persistent Memory
  - Media vs. access/implementation (NVM)
  - Programing model
  - SNIA TWG Work
  - Security
  - Alliances/Use-cases

Persistent Memory

Persistent Memory (PM) Technology
is a type of Non-Volatile Memory (NVM)

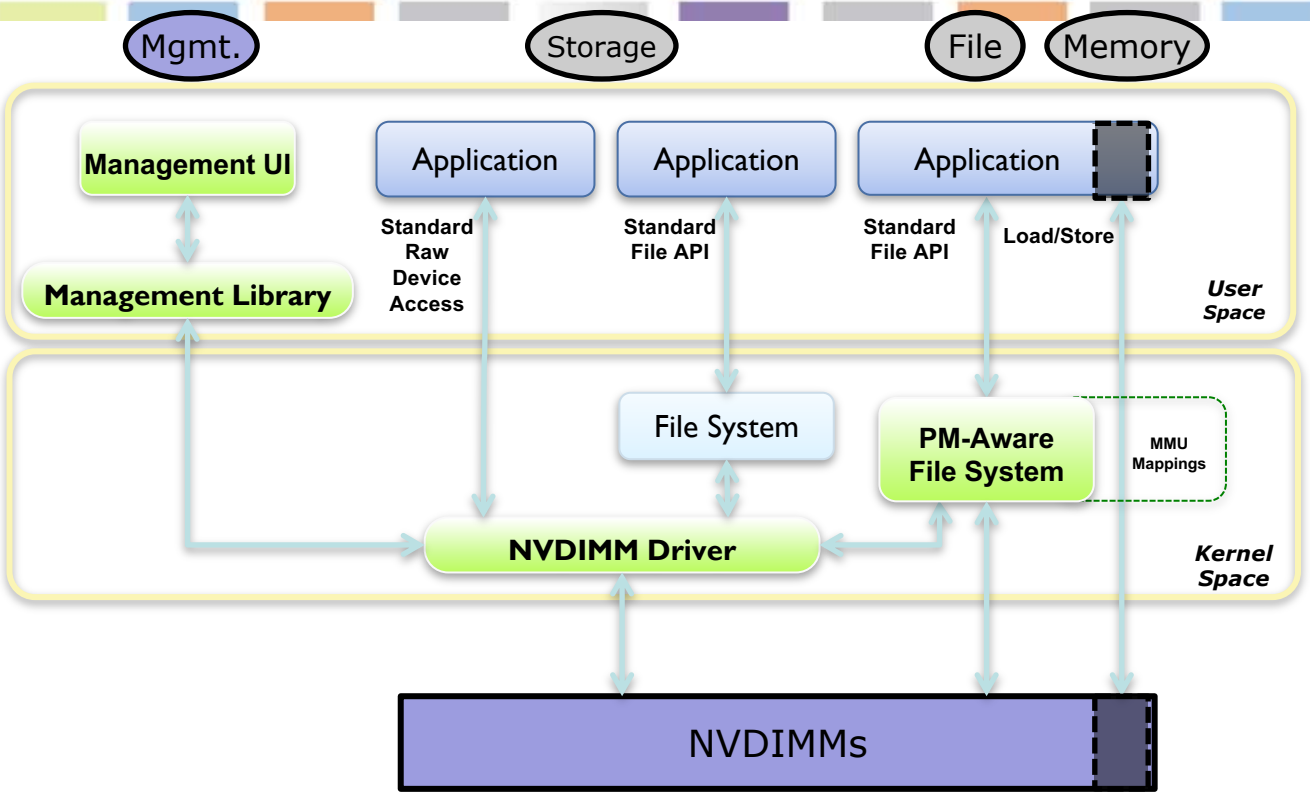
- Disk-like non-volatile memory
  - Persistent RAM disk
  - Appears as disk drives to applications
  - Accessed as traditional array of blocks
- Memory-like non-volatile memory (PM)
  - Appears as memory to applications
  - Applications store data directly in byte-addressable memory
  - No IO or even DMA is required

Persistent Memory Programming: The Current State and Future Direction

Ancient History

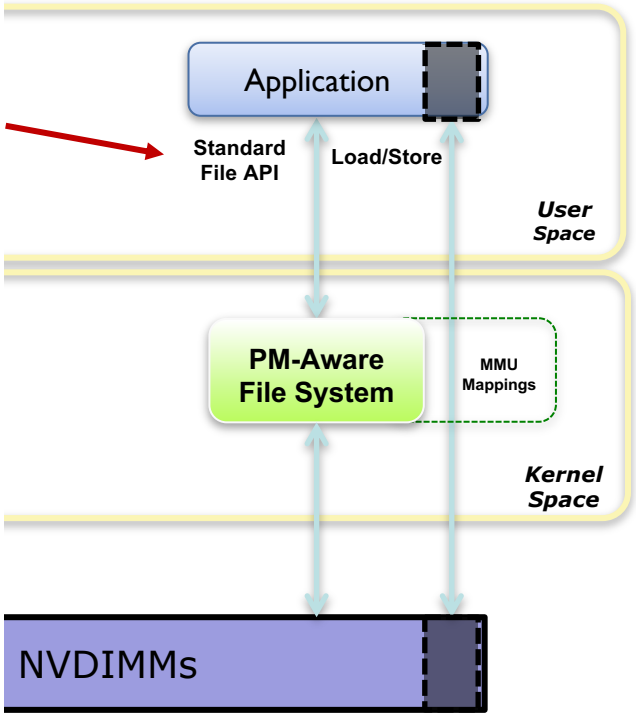
- June 2012
  - Formed the NVM Programming TWG
  - Immediate participation from key OSVs, ISVs, IHVs
- January 2013
  - Held the first PM Summit (actually called “NVM Summit”)
- January 2014
  - TWG published rev 1.0 of the NVM Programming Model

The Programming Model

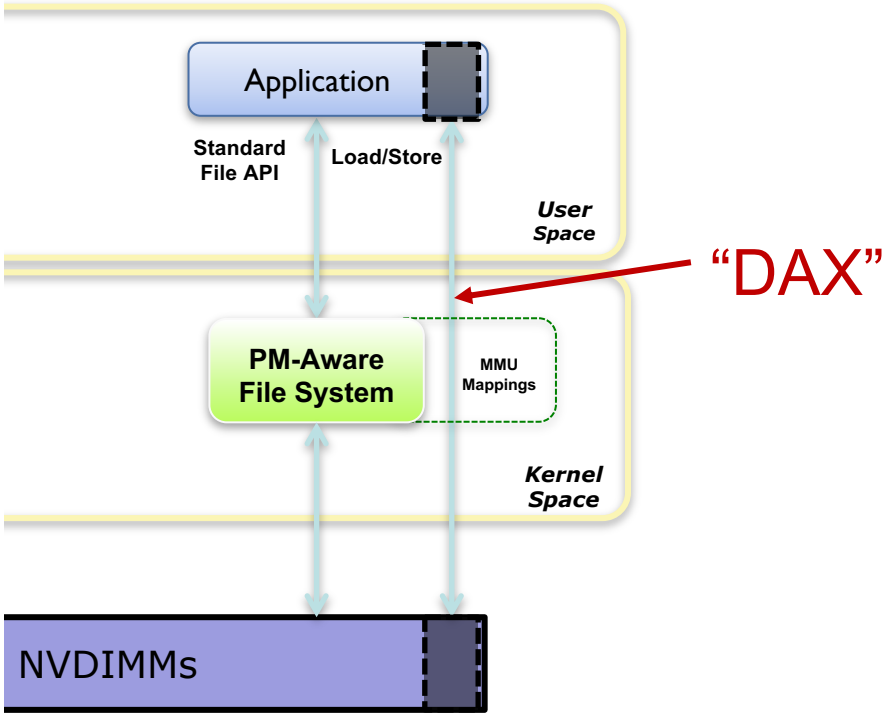


Must Open File Before Mapping

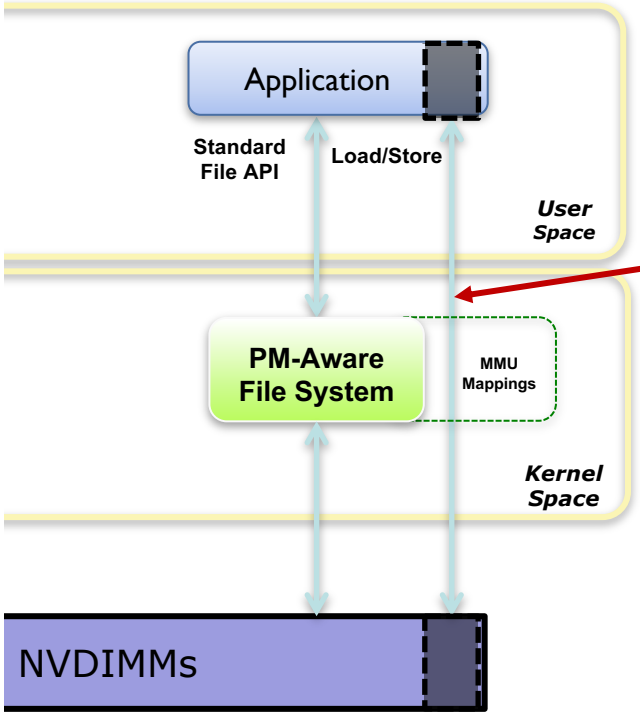
Standard Naming and Permission Model



Direct Access



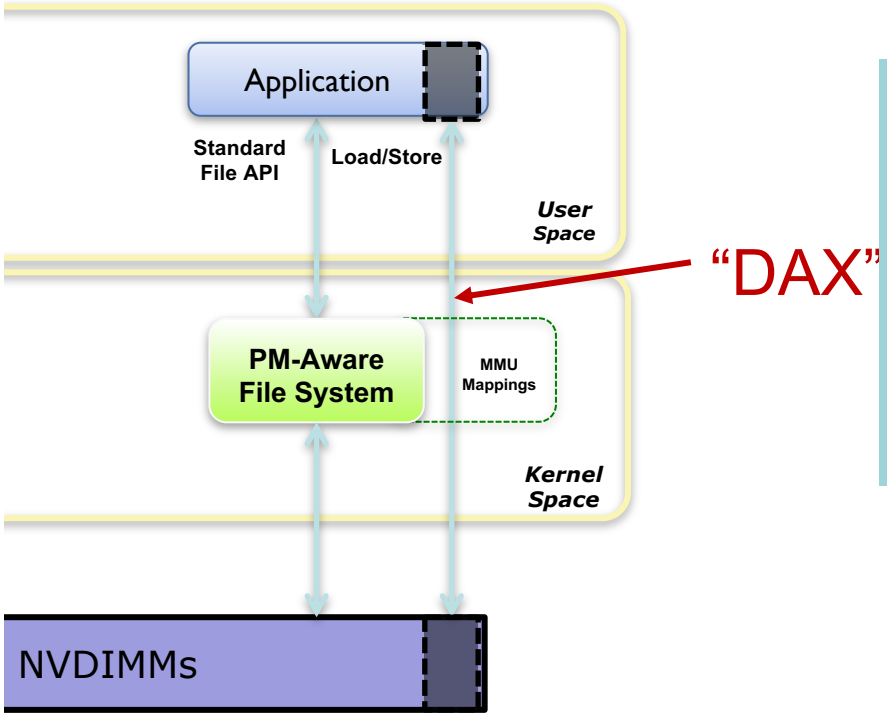
Direct Access



"DAX"

Windows:

- DAX Support is shipping
- NTFS is PM-Aware
- Some new APIs
- PMDK support

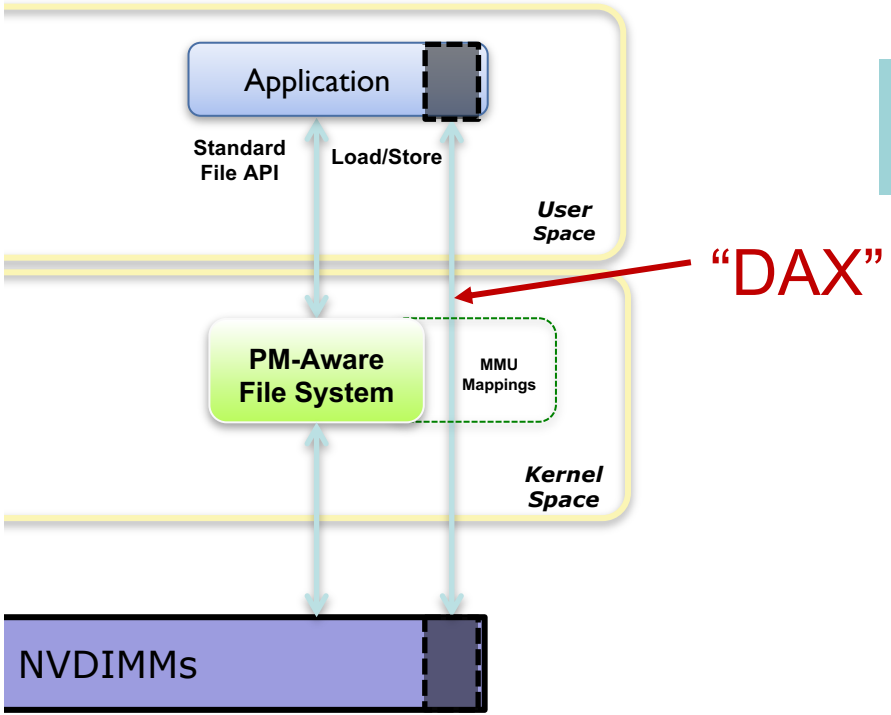


"DAX"

Linux:

- DAX Support is shipping
- ext4 is PM-Aware
- XFS is PM-Aware
- PMDK support

- More filesystems coming

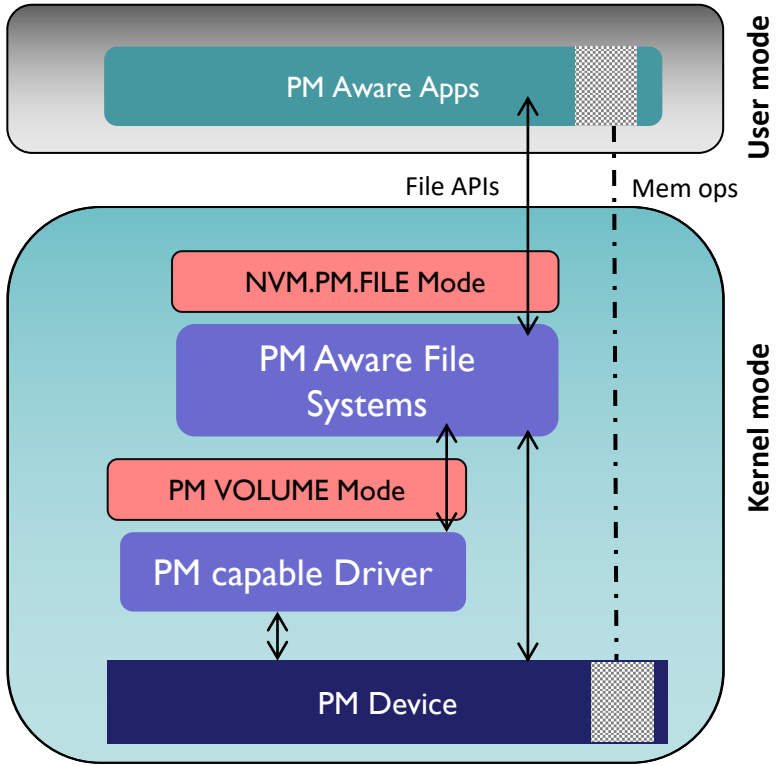


"DAX"

VMware:
Virtualization of PM

Persistent Memory (PM) Modes

- NVM.PM.VOLUME Mode
  - Software abstraction for persistent memory hardware
  - Address ranges
  - Thin provisioning management
- NVM.PM.FILE Mode
  - Application behavior for accessing PM
  - Mapping PM files to application address space
  - Syncing PM files



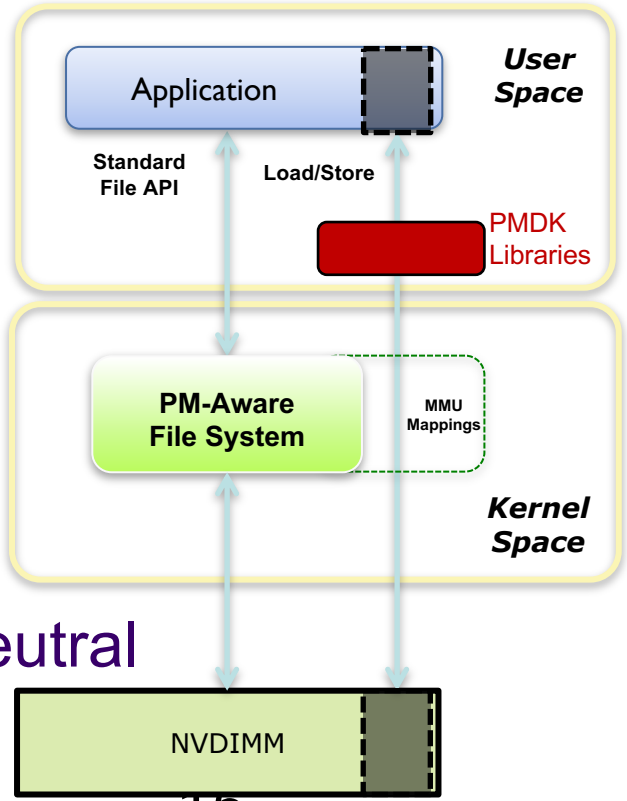
Applications: Public Demos

- 2017 was an interesting year for demos…
- SAP SAPPHIRE
- Oracle OpenWorld
- Built on the Persistent Memory programming model!

Persistent Memory Developer Kit

pmem.io

- PMDK Provides a Menu of Libraries
  - Developers pull in just what they need
    - Transaction APIs
    - Persistent memory allocators
  - Instead of re-inventing the wheel
    - PMDK libraries are fully validated
    - PMDK libraries are performance tuned
- PMDK Provides Tools for Developers
- PMDK is Open Source and Product-Neutral



PMDK in a Nutshell

- Complex transactions, allocation handled by libraries
  - No “flush” calls to manage in most cases
  - Each ISV doesn’t have to re-invent
  - Performance tuned (esp for future enhancements)
- Licensing is very liberal
  - Steal all the code you want!
- PMDK is a convenience, not a requirement
  - Build your own library if you like!

More Information

- http://snia.org/PM
  - Specs, workgroups, webcasts, videos, presentations
- http://pmem.io
  - PMDK and other persistent memory programming information
- http://pmem.io/documents
  - Links to publications, standards, Windows & Linux info

TWG Work

TWG Ongoing Work

- Security
  - PM Hardware Security Threat Model
- Remote persistent memory (via RDMA)
  - Ongoing – optimizations for RDMA worked in multiple forums
  - Remote asynchronous flush (under discussion)
- Higher-level Semantics
  - As we learn more..

Updating Original Work

- Error handling
  - Additions to V1.2 of the programming model specification
  - Refinements to error handling annex
- Atomicity
  - New white paper
  - Introduces PM data structure libraries with atomicity built in
  - Enables PM transactions

Persistent Memory Security

Purpose of SNIA PM Security work

- This work documents approaches for encryption of data on persistent memory (PM); particularly considering unique characteristics of PM.
  - Discover gaps in existing technologies related to PM security
  - Create a treat model and suggest requirements that could resolve these gaps
- The NVM Programming TWG has established an alliance with the Trusted Computing Group (TCG) outlining a collaboration between the SNIA NVMP TWG, TCG. The collaboration is structured as follows.
  - SNIA provides application/user level roles, behaviors and threat models
  - TCG provides security protocol definitions
- TCG, SNIA also approaching JEDEC
  - JEDEC provides NVDIMM specific specifications

PM Security

- Many aspects of security are unchanged by PM
  - Administrative security
  - Key management
  - Memory protection
- First order requirement: encryption of data at rest
  - Authentication/Re-authentication Triggers
  - Real time encryption mechanics
  - Continuity of principal identity

PM Security

- Protection granularity at the file and volume layers
  - Device, partition or volume protection of data at rest
  - Memory mapped file access authorization enforcement
- Achieving isolation analogous to external storage
  - Limiting access enablement windows
  - Rapid privilege transition

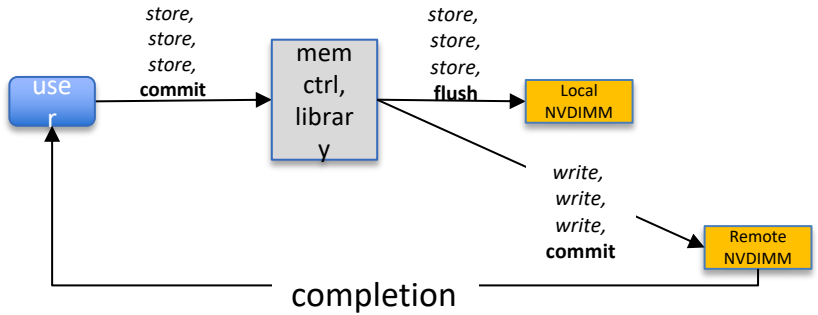
Public and private cloud requirements

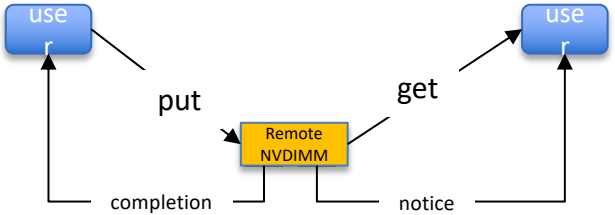
- Public speaks to how trust is established and isolation is assured in shared public cloud infrastructure
- Private speaks to multi-tenancy HW support
- Both – encryption at rest, issues from prior 2 slides

Alliances/Use Cases

REMOTE PERSISTENT MEMORY

High Availability Use Case





Remote Shared Memory Use Case

Collaborate to define solutions for multiple use cases

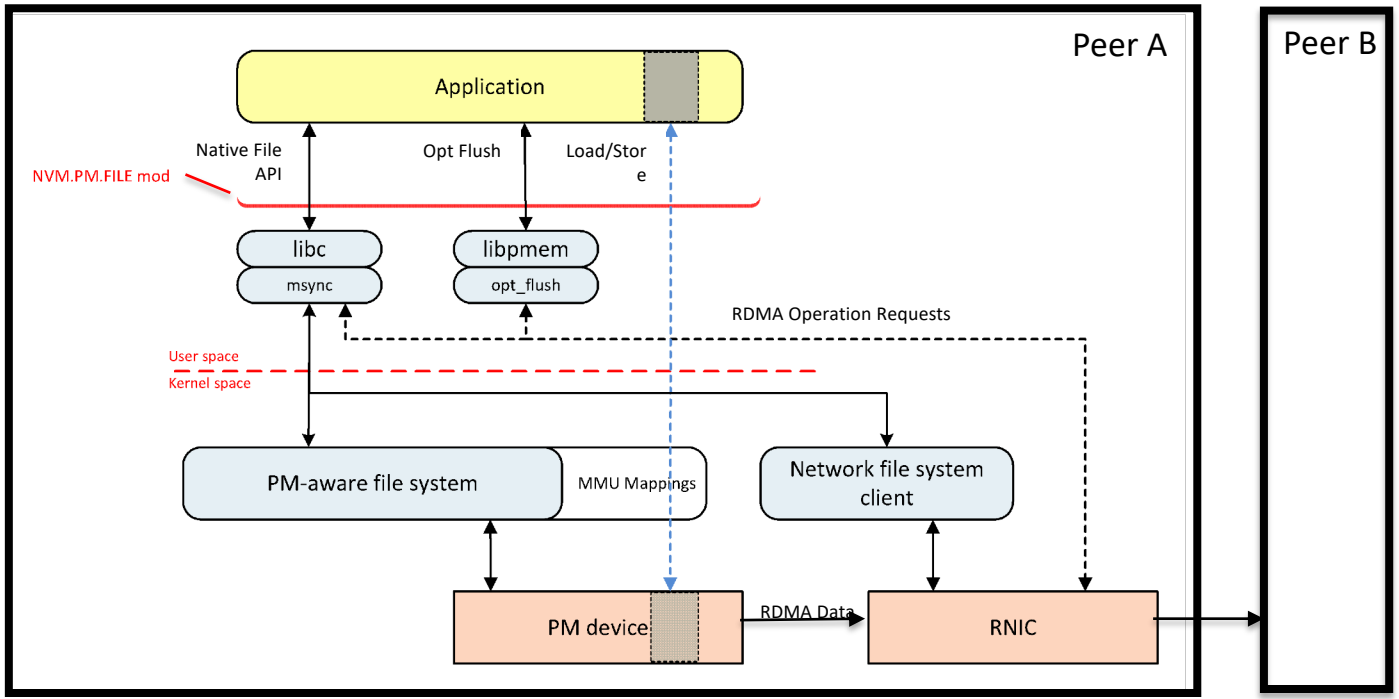
REMOTE ACCESS FOR HA SOFTWARE MODEL

RDMA for HA During msync or opt_flush

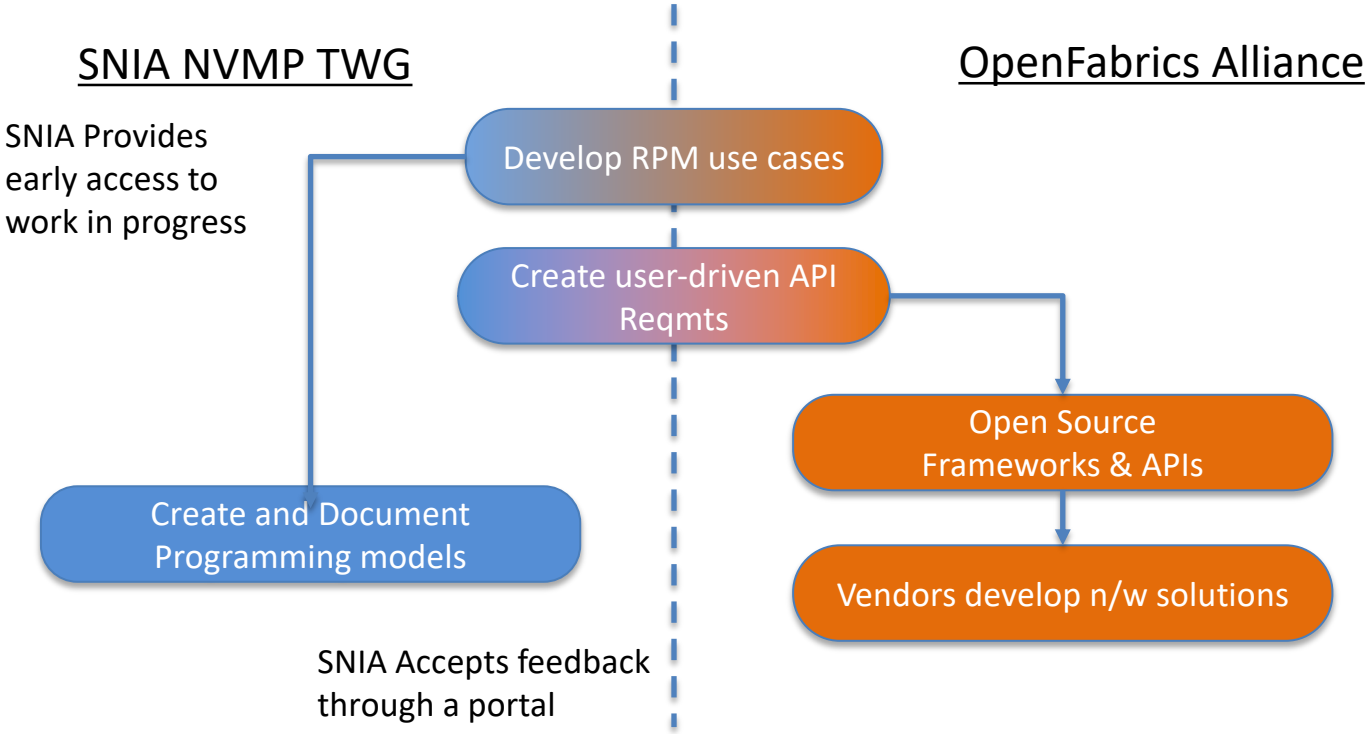
Remote Access for HA white paper released:

http://www.snia.org/sites/default/files/technical_work/final/NVM_PM_Remote_Access_for_High_Availability_v1.0.pdf

Requirements for consistent data recovery, for efficient remote optimized flush



SNIA & OPENFABRICS ALLIANCE



Backup

Persistent Memory Threat Model

Roles for Threat Model

- Customer – Security Principal/Data Owner Organization
- Developer – Storage/Application Developer, DevOps
- Security Officer – Security Rights Assigner
- Administrator – System configuration manager
- Deliver-er/Repair-er – Factory/Channel Support, Supply Chain

Secure Insecure

Threat Model

Attack		Attacker	Applicable existing approach	New issues with PM
Cross-Tenant	Privacy/ Confidentiality	Tenant, Administrator, Repair-er	Traditional authorization, authentication. Encryption at rest. Separation of roles. Memory protection.	None
	Integrity	Developer, tenant, administrator	Traditional authorization, authentication. Separation of roles. Memory protection.	Increased scope of damage due to mismanaged pointers, memory resources
	Availability – denial of service	Tenant, Developer	Per-tenant QoS	Potential for rapid disruption with limited detection window

Threat Model

Attack		Attacker	Applicable existing approach	New issues with PM
Cross-Tenant	Tenant, Administrator	Tenant, Administrator, Repair-er	Secure erasure (physical or cryptographic) during deletion	More rapid free space recycling in memory than disk.
Insider	Local HW attacks (e.g. DMA)	Tenant, Administrator, Developer	Memory Protection, Per-tenant QoS applied to IO	
	Remote access threats (e.g. RDMA)	Tenant, Administrator, Developer	RDMA security, s-tag, range access enforcement	

Threat Model

Attack		Attacker	Applicable existing approach	New issues with PM
Insider	Malware	Developer, deliver-er, repair-er, Administrator	Digital signing, virus protection	
	Access by admin/support	Administrator	Role separation, authentication/ Authorization	