



SDC¹⁸

September 24-27, 2018
Santa Clara, CA

www.storagedeveloper.org

Introduction to Persistent Memory Configuration and Analysis Tools

Usha Upadhyayula (Intel)
Steve Scargall (Intel)

Agenda

- ❑ Overview of Persistent Memory Tools
- ❑ Provisioning Utilities
- ❑ Persistent Memory Developer Kit (PMDK)
- ❑ Benchmarking Tools
- ❑ Analysis Tools
- ❑ Provisioning Persistent Memory Walkthrough
- ❑ Q&A

Introduction to Persistent Memory

What is Persistent Memory?

- ❑ Byte Addressable like DRAM
- ❑ Persistent Like Storage
- ❑ Cache Coherent
- ❑ Load/Store Accessible
- ❑ DIMM Form Factor

Why does it matter now?

- ❑ Larger Capacity, Higher Endurance, & Low Latency
- ❑ Adds a new tier between DDR & Block Storage (SSD/HDD)
- ❑ Ability to do in-place persistence
 - ❑ No Paging, No Context Switching, No Interrupts, No Kernel code execution
- ❑ Ability to do DMA and RDMA

Overview of Persistent Memory Tools

Persistent Memory Tools

Configuration

- ❑ Pre-boot
 - ❑ ipmctl
- ❑ Linux
 - ❑ ipmctl
 - ❑ ndctl
- ❑ Windows
 - ❑ ipmctl
 - ❑ New-StoragePool
 - ❑ New-Volume

Benchmark

- ❑ Intel® Memory Latency Checker (MLC)
- ❑ FIO (Flexible IO Tester)
- ❑ pmembench

Analysis

- ❑ Intel® VTune Amplifier
 - ❑ Memory Analyzer
 - ❑ Storage Analyzer
- ❑ Intel® Persistent Inspector
- ❑ Intel® VTune Platform Profiler
- ❑ pmempool
- ❑ pmemcheck
- ❑ Valgrind

Persistent Memory Enabling Standards

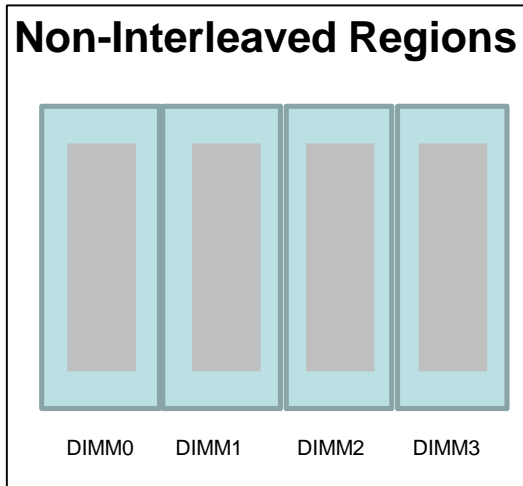
- ❑ ACPI 6.0 and above
 - ❑ NVDIMM Firmware Interface Table (NFIT)
 - ❑ Device Specific Methods (DSM)
- ❑ SNIA Programming Model
- ❑ DMTF SMBIOS 3.2.0 and above
- ❑ UEFI Spec
 - ❑ Namespace Label Protocol



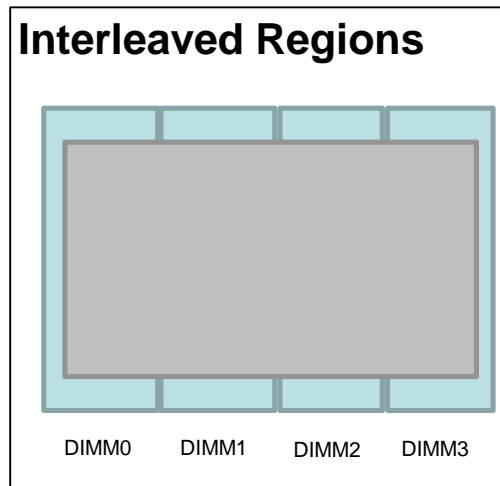
Provisioning Utilities



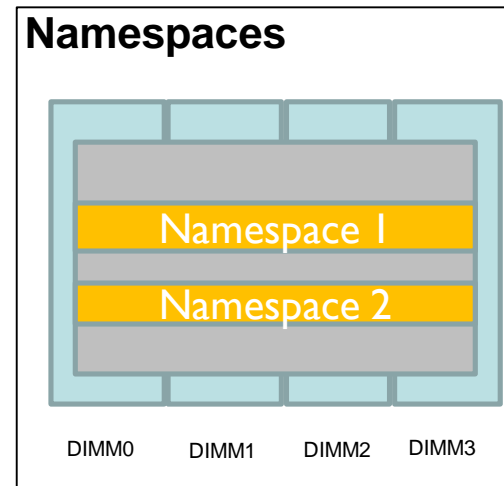
Provisioning Terms & Concepts



Regions are created within [non]interleaved sets. Interleaving can be 1 to n-way mapping.

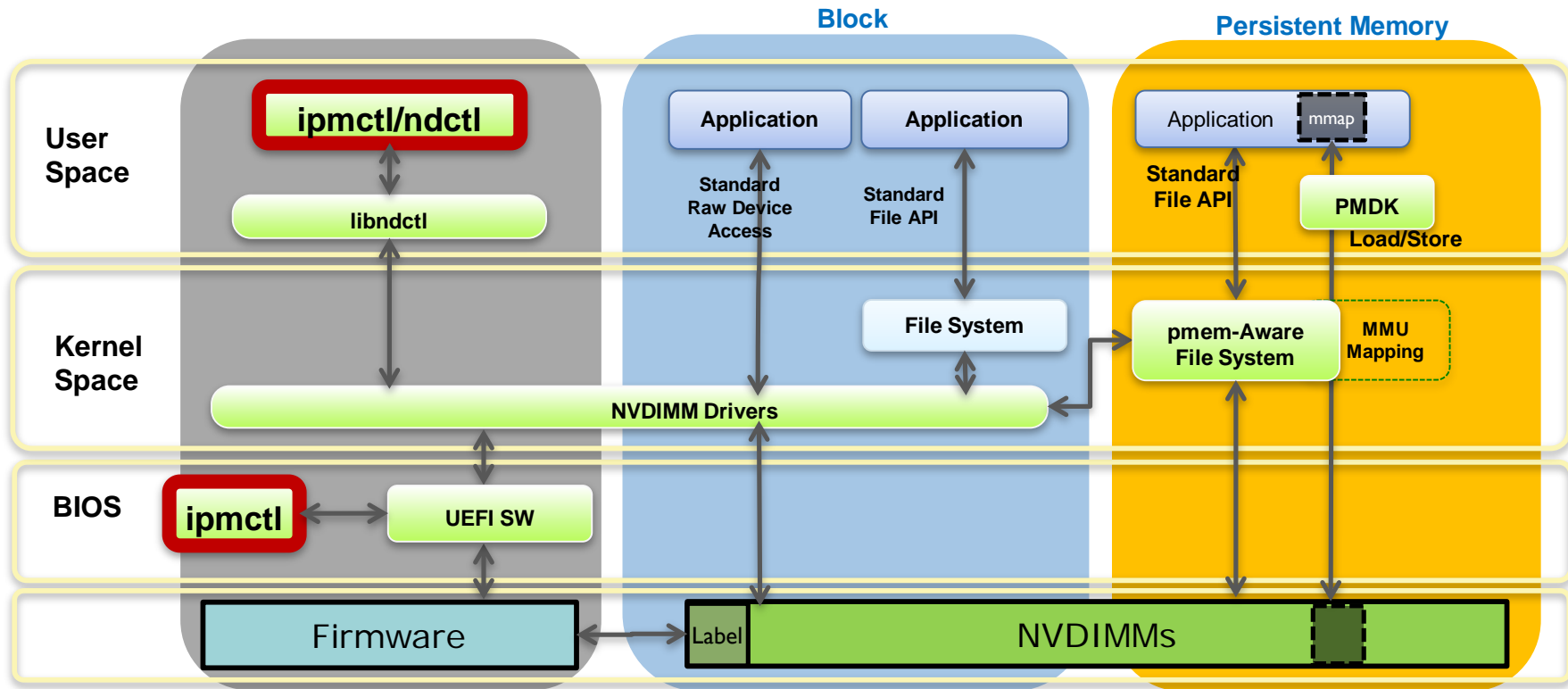


Creates contiguous physical address space and provides striped reads/writes for better throughput.



Similar to SSD, raw capacity of a region is partitioned into one or more logical devices called namespaces.

ipmctl & ndctl



ipmctl/ndctl – Supported Functions

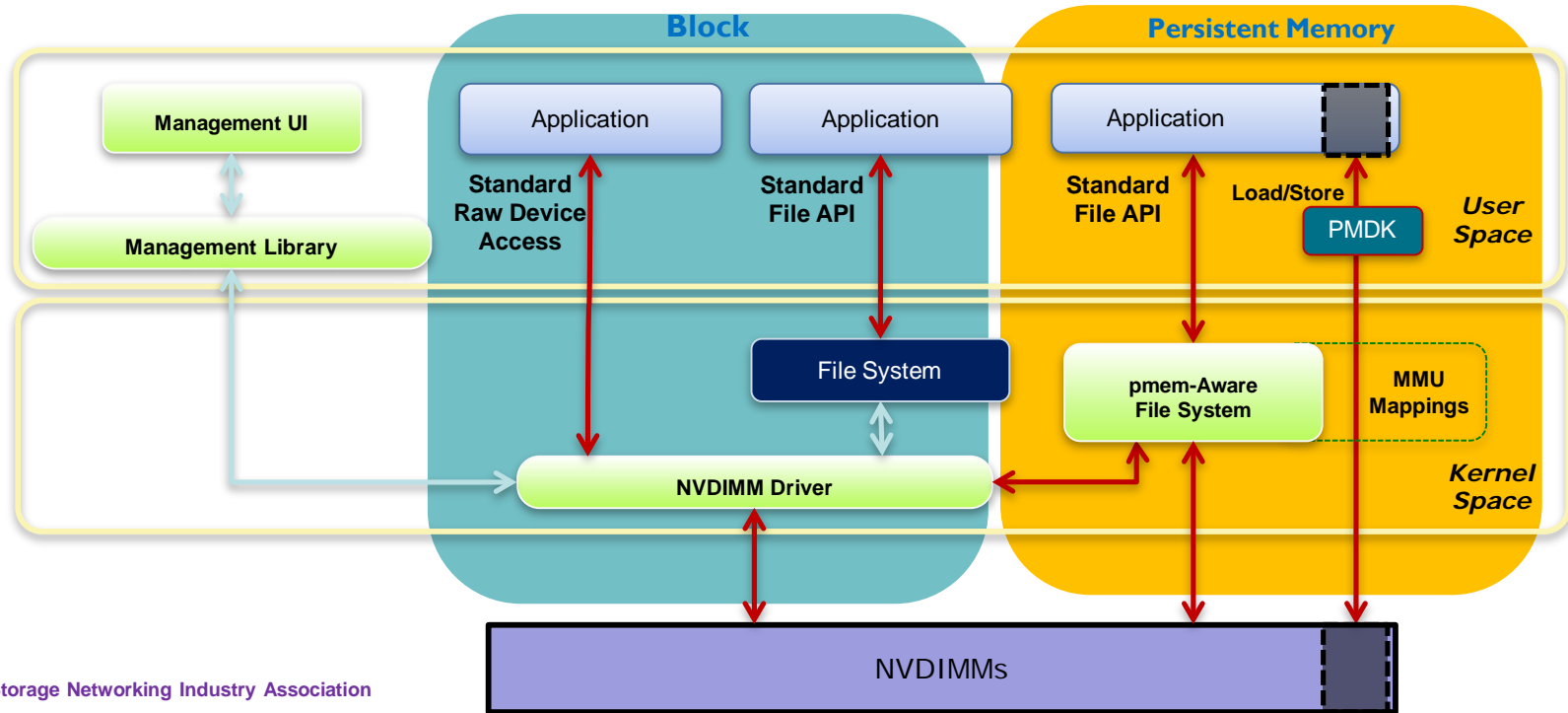
- ❑ Provisioning Regions and Namespaces
 - ❑ create, enable, disable, destroy, list
- ❑ Monitoring/Maintenance
 - ❑ DIMM topology, health, firmware, media-errors

Persistent Memory Developer Kit (PMDK)



The SNIA* NVM Programming Model

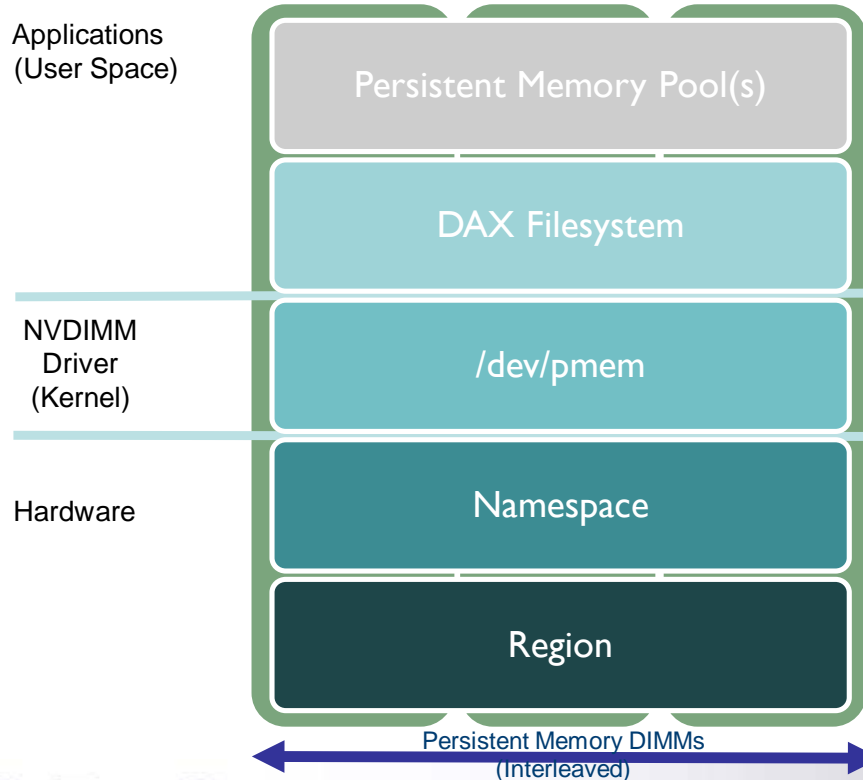
Exposing Persistent Memory to Applications



* - Storage Networking Industry Association

Exposing Persistent Memory to Applications

Filesystem DAX (FSDAX)



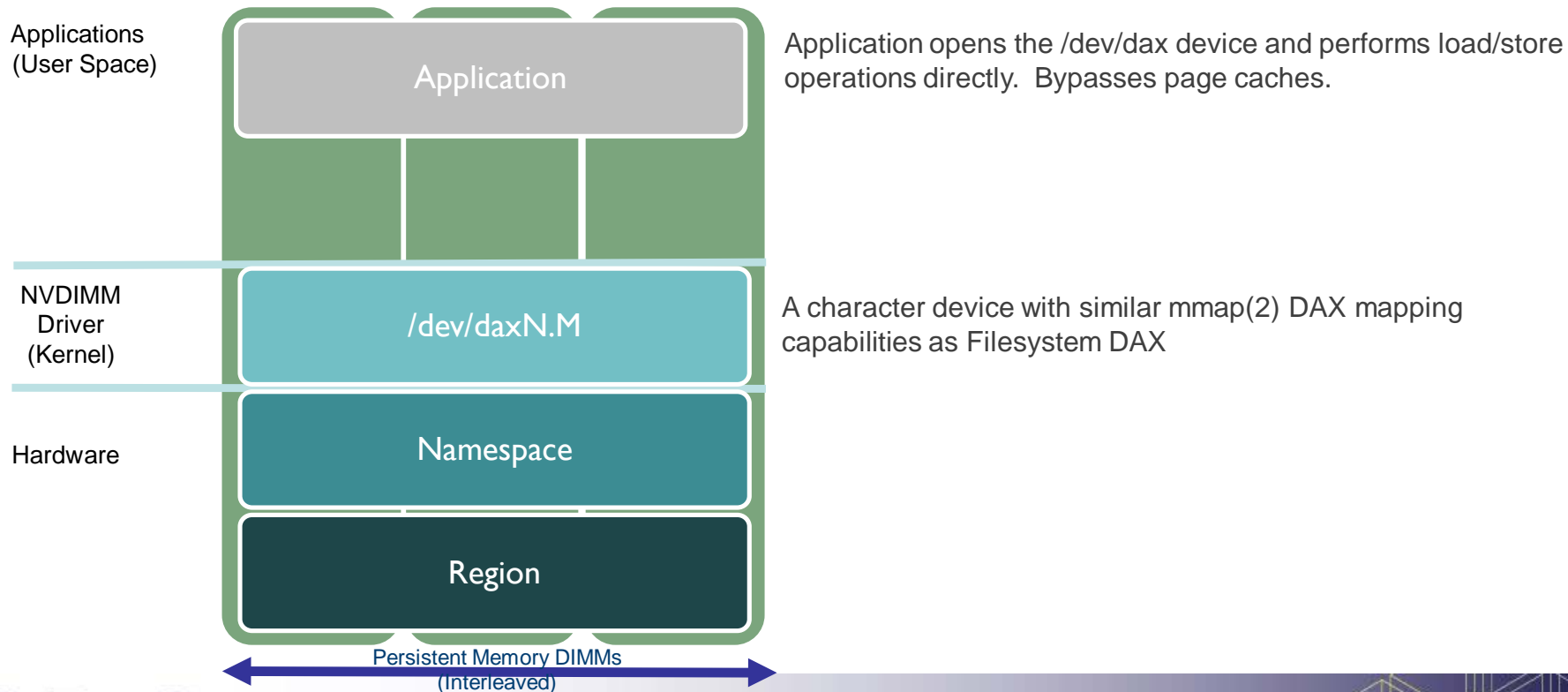
Persistent Memory Pool(s): persistent memory is exposed by the OS to the application as memory-mapped files when using PMDK.

Direct Access (DAX) Filesystem: For file mappings (mmap), the storage device is mapped directly into user space and bypasses page cache.

/dev/pmem: a device used to create a filesystem.

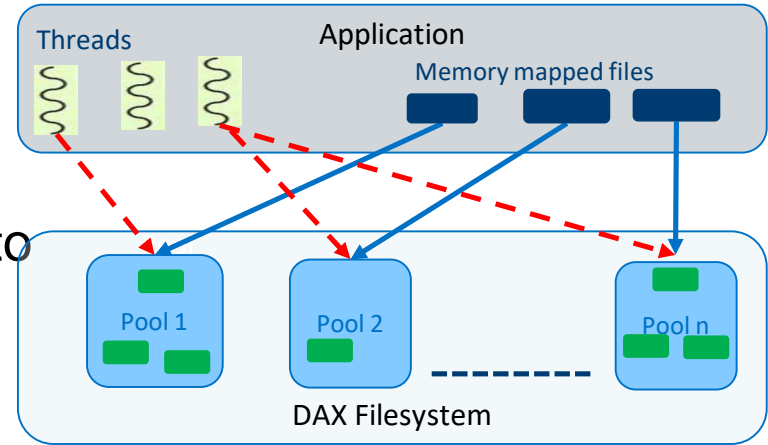
Exposing Persistent Memory to Applications

Device DAX (DevDAX)



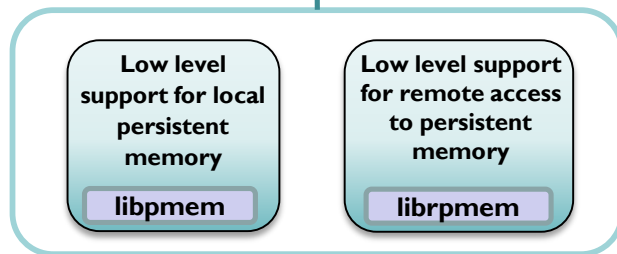
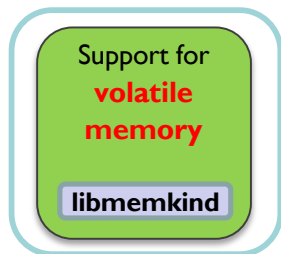
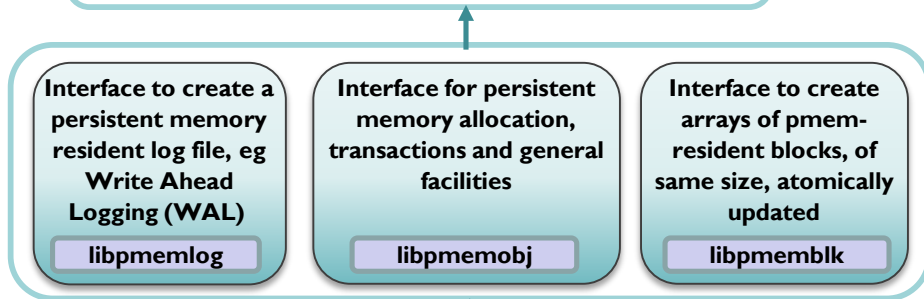
Persistent Memory Pools

- ❑ Intended for use on DAX File System
- ❑ Pools are tagged with a 'layout' name/string for identification
- ❑ Support for multiple pools per Application
- ❑ Pools can be aggregated into 'pool sets' to provide a larger address space and replication
- ❑ Easy backup/restore



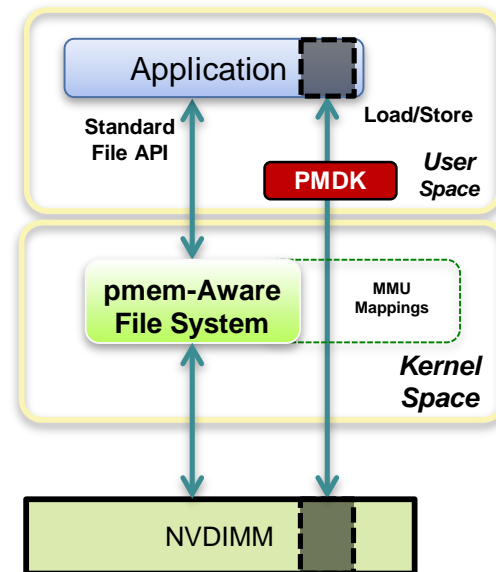
PMDK: A Suite of Open Source Libraries

Multiple Language Bindings



Low-level support

Transaction Support



In Development:
PCJ – Persistent Collection for Java
LLPL – Low-Level Persistence Java Library

Benchmarking Tools

Benchmarking Tools

- ❑ Memory Latency Checker (MLC)
 - ❑ Measures Memory Latencies and Bandwidth
 - ❑ Supports Intel® DCPMM
 - ❑ <http://www.intel.com/software/mlc>
 - ❑ Available on Windows & Linux
- ❑ FIO
 - ❑ Persistent Memory IOengines
 - ❑ libpmem, dev-dax, & libpmemblk
 - ❑ <http://pmem.io/2018/06/25/fio-tutorial.html>
- ❑ pmembench
 - ❑ Helps developers contributing to PMDK to evaluate performance improvements and catch regressions.

Analysis Tools

PMDK pmempool

Utility for Offline Analysis of Persistent Memory Pools

- ❑ Usage: pmempool <command> [<args>]
- ❑ Commands:
 - **info** : Prints information and statistics in human-readable format about specified pool
 - **create** : Creates a pool of specified type with additional properties specific for this type of pool
 - **check** : Checks pool's consistency and repairs pool if it is not consistent.
 - **dump** : Dumps usable data from pool in hexadecimal or binary format.
 - **rm** : Removes pool file or all pool files listed in poolset configuration file.
 - **convert** : Updates the pool to the latest available layout version.
- ❑ <http://pmem.io/pmdk/pmempool/>

PMDK pmemcheck

- ❑ Checks for non-persistent stores
 - ❑ Identified through the appropriate sequence of operations
 - ❑ Supports Valgrind, DRD, Helgrind, and Memcheck
 - ❑ PMDK delivers a modified Valgrind
- ❑ Logs persistent memory operations for post-processing
 - ❑ eg: Store re-ordering with fault injection
- ❑ Supports persistent memory transactions
 - ❑ Similar to Database transactions
 - ❑ Partially ACID (Atomicity, Consistency, Isolation, Durability)

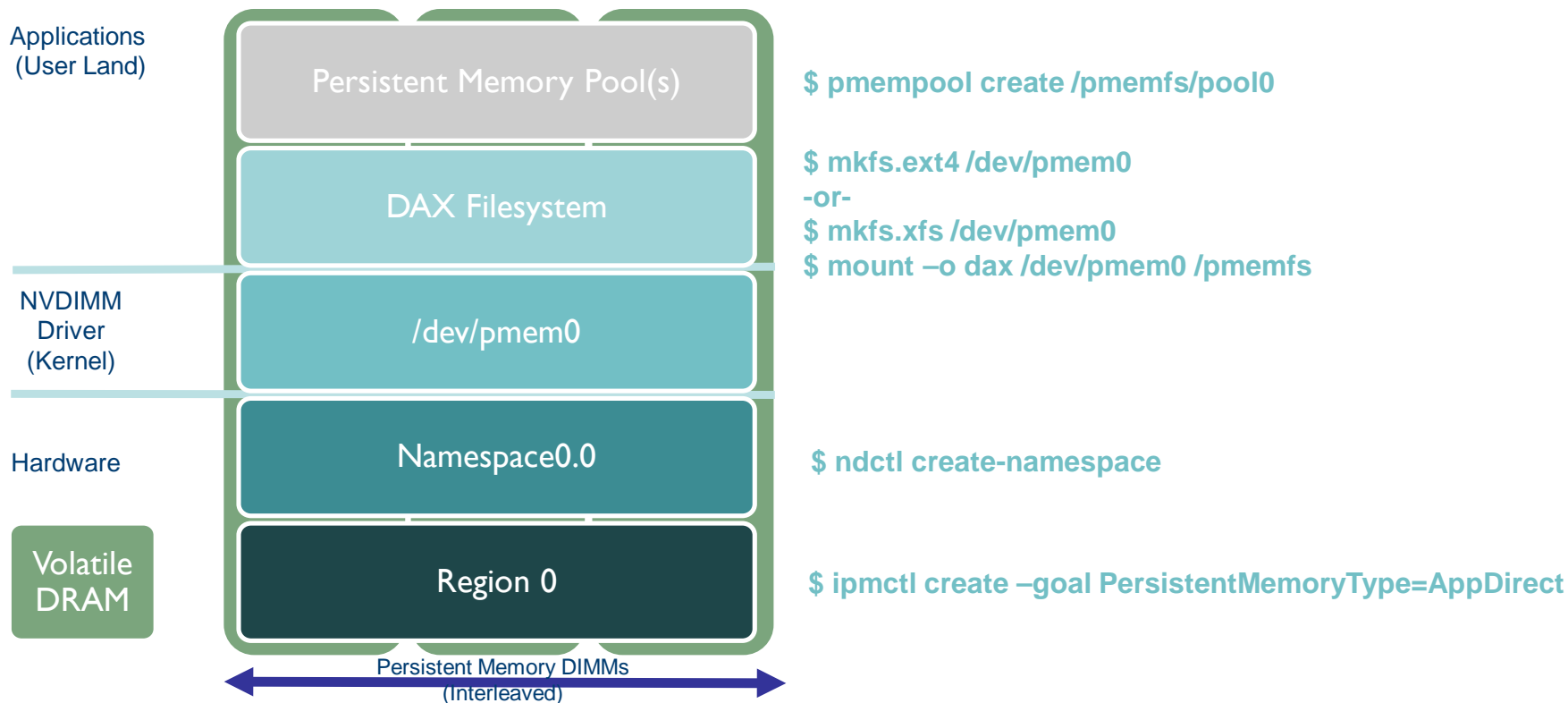
Software Tools For Intel® Optane™ DC Memory

Intel® Parallel Studio XE 2019

- ❑ Intel® VTune™ Amplifier
 - ❑ Detailed Application Analysis
 - ❑ Memory analysis – design data structures for hot/warm/cool memory
 - ❑ Storage analysis – are you CPU or I/O bound?
- ❑ Intel® VTune™ Platform Profiler
 - ❑ Uses hardware counters for system wide performance analysis and visualization
 - ❑ Find configuration issues and potential for larger memory
- ❑ Intel® Persistence Inspector
 - ❑ Finds missing/redundant cache flushes, PMDK logging errors, and more...
- ❑ Intel® Advisor
 - ❑ Memory Access Profiling – Identifies memory working set size and hot loops within the code
 - ❑ Cache Simulation feature - Allows you to get accurate memory footprints and miss information for your application
- ❑ Available for download
 - ❑ Visit <https://software.intel.com/en-us/persistent-memory>

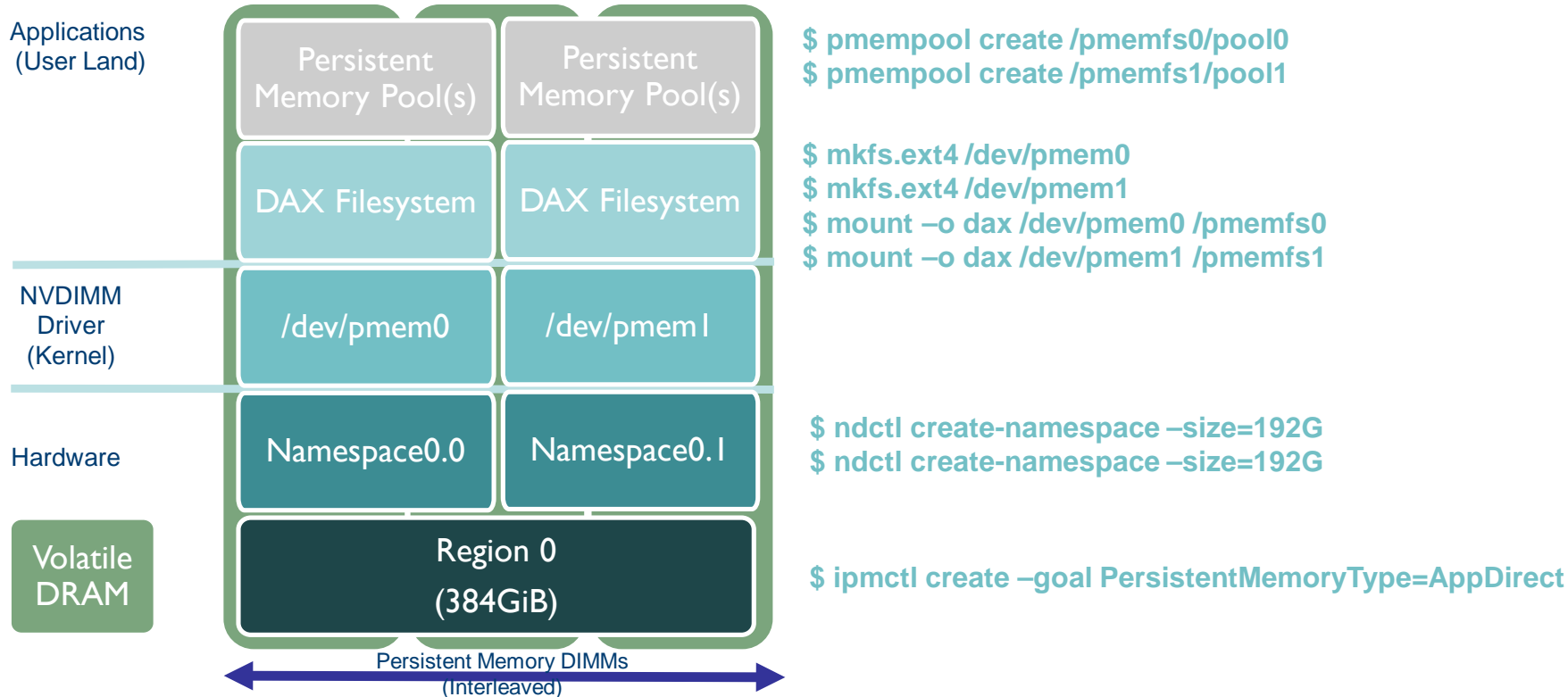
Provisioning Persistent Memory Walkthrough

Creating Persistent Memory Regions & Namespaces



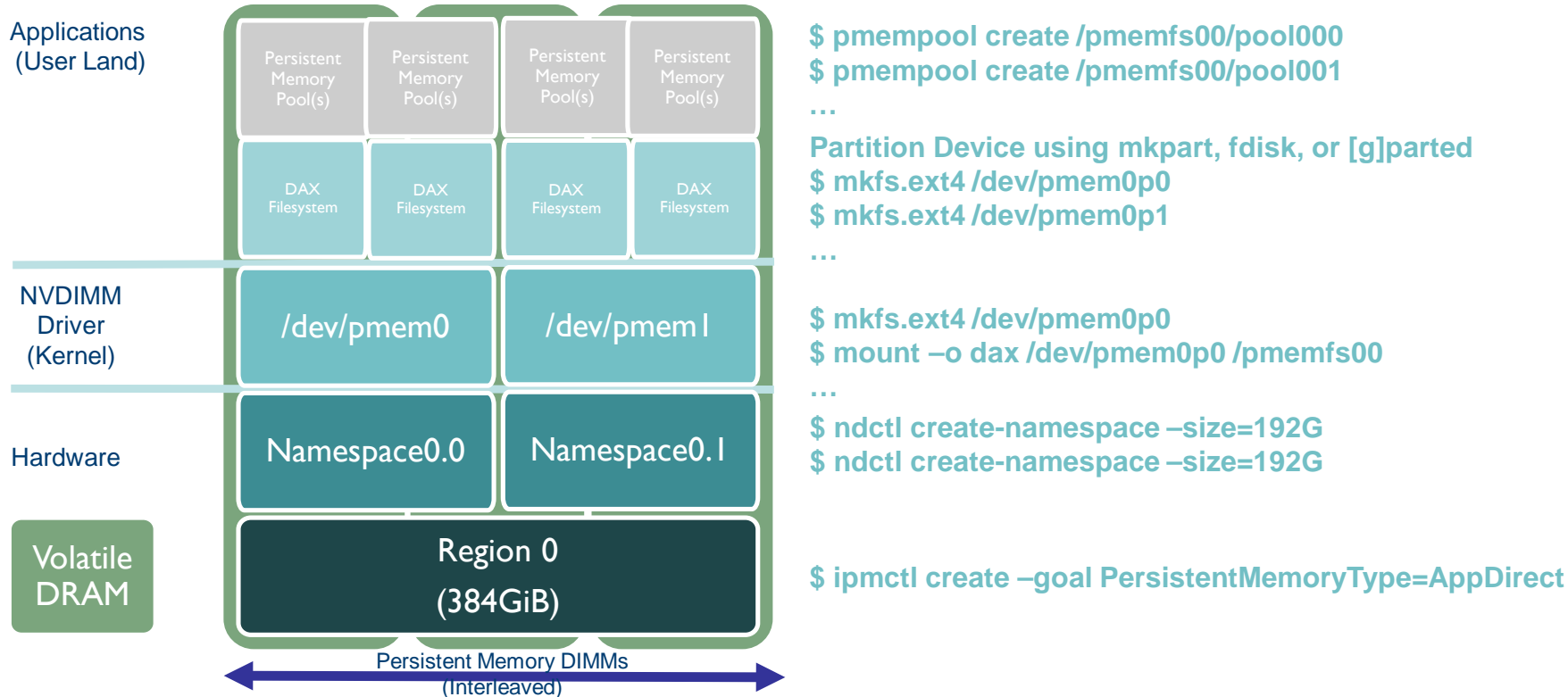
Creating Persistent Memory Regions & Namespaces

Multiple Namespaces per Region



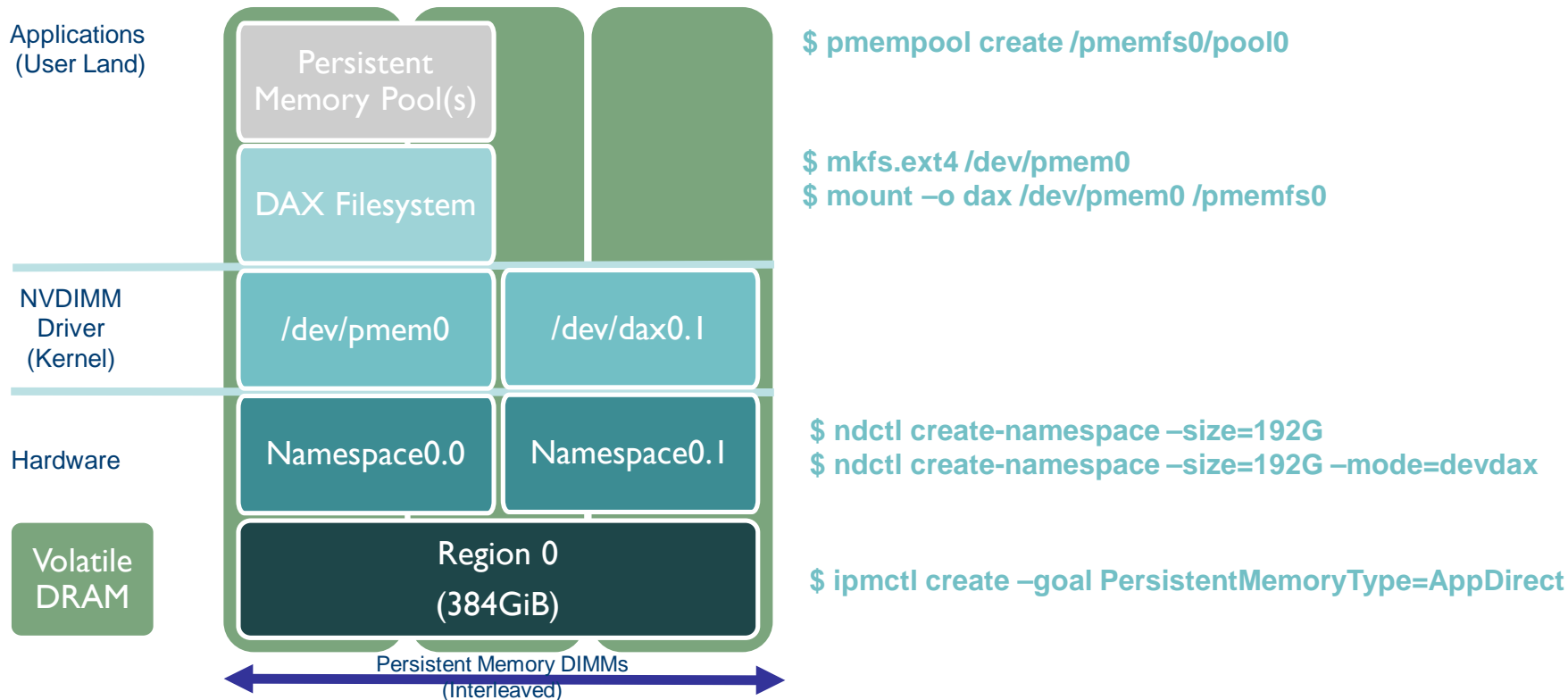
Creating Persistent Memory Regions & Namespaces

Device Partitioning



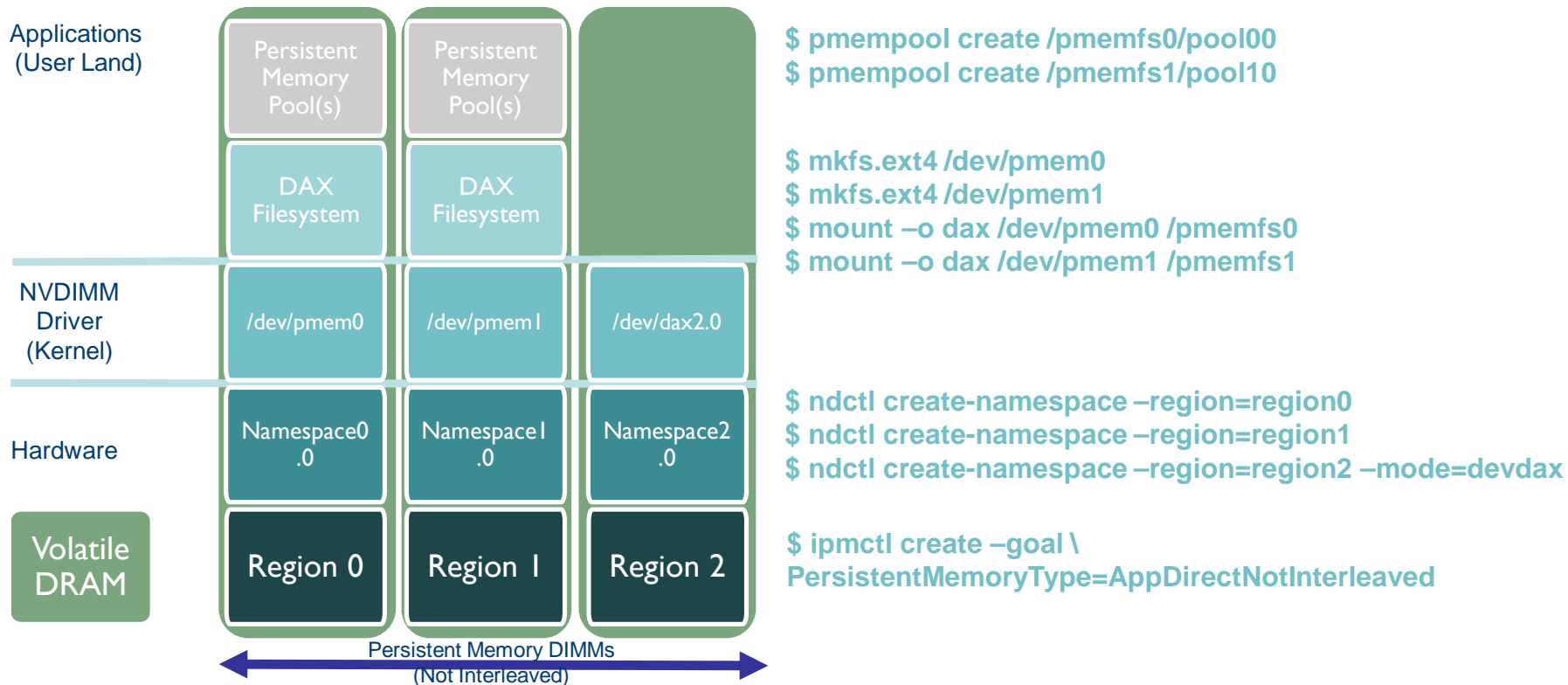
Creating Persistent Memory Regions & Namespaces

Mixing Namespaces Modes



Creating Persistent Memory Regions & Namespaces

Non-Interleaved NVDIMMs: Mixing Namespace Modes



pmempool info

- ❑ Displays pool header info and user data in human readable format
- ❑ Useful for debugging
- ❑ '-s' flag provides pool statistics
- ❑ Works with pools and poolsets

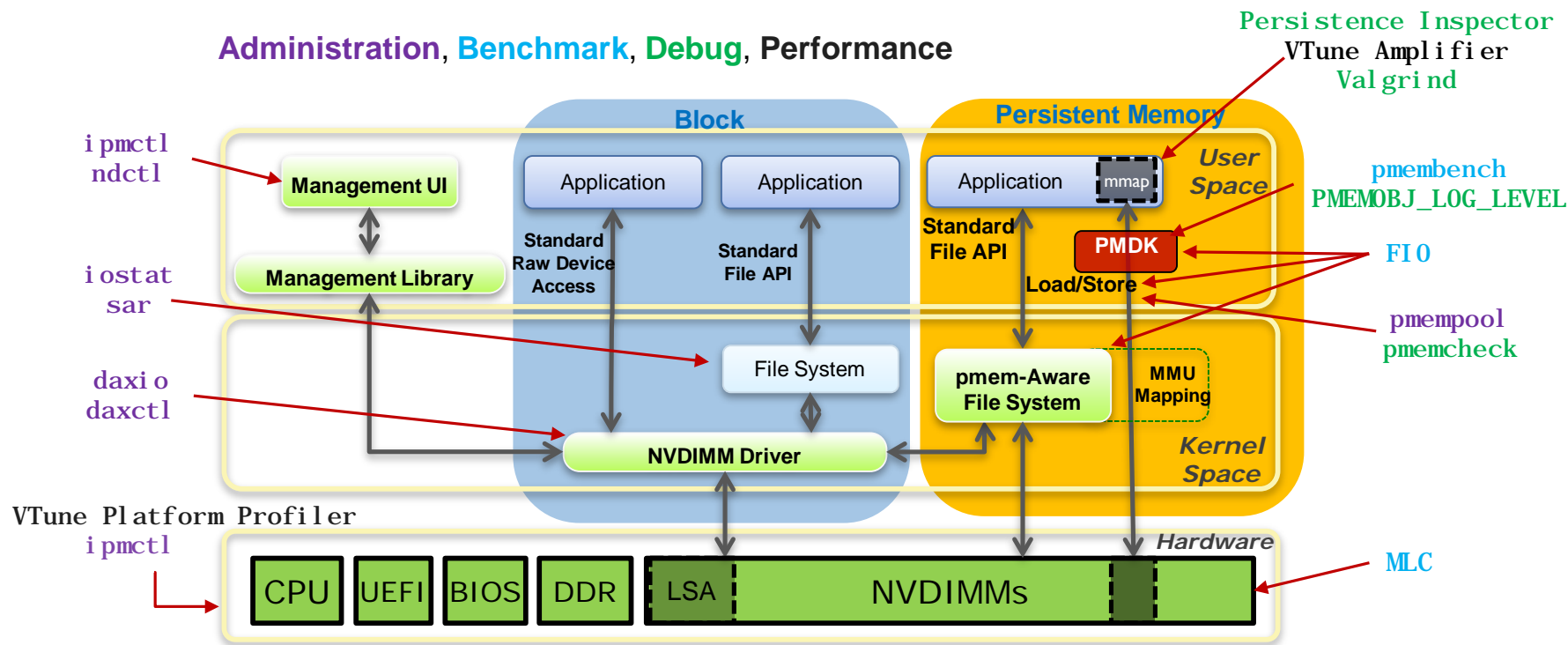
```
# pmempool create --size=4G --layout my_layout obj pool.obj
# pmempool info pool.obj
Part file:
path                : pool.obj
type                : regular file
size                : 4294967296

POOL Header:
Signature           : PMEMOBJ
Major              : 4
Mandatory features  : 0x0
Not mandatory features : 0x0
Forced RO           : 0x0
Pool set UUID       : da87ab10-b59c-4e66-90cc-076a70f791ba
UUID               : 44756b5f-91c7-4f63-89a3-ed08a1facd68
Creation Time       : Tue Sep 04 2018 13:56:42
Alignment Descriptor : 0x0000007f737777310[OK]
Class              : 64
Data               : 2's complement, little endian
Machine            : AMD X86-64
Checksum           : 0x36177b624a431c62 [OK]

PMEM OBJ Header:
Layout              : my_layout
Lanes offset        : 0x2000
Number of lanes     : 1024
Heap offset         : 0x302000
Heap size           : 4291813376
Checksum            : 0x57fa81b7e1ff3742 [OK]
Root offset         : 0x0
```



Persistent Memory Tools Recap



Resources

- PMDK Resources:
 - Home: <https://pmem.io>
 - PMDK: <https://pmem.io/pmdk>
 - PMDK Source Code : <https://github.com/pmem/PMDK>
 - Google Group: <https://groups.google.com/forum/#!forum/pmem>
 - Intel Developer Zone: <https://software.intel.com/persistent-memory>
- NDCTL: <https://pmem.io/ndctl>
- IPMCTL: <https://github.com/intel/ipmctl>
- MemKind: <https://memkind.github.io/memkind/>
- LLPL: <https://github.com/pmem/llpl>
- PCJ: <https://github.com/pmem/pcj>
- SNIA NVM Programming Model:
https://www.snia.org/tech_activities/standards/curr_standards/npm
- Getting Started Guides: <https://docs.pmem.io>

Takeaways

- ❑ Excitement for this disruptive technology
- ❑ Expand your toolbox
- ❑ Where to find help and information
- ❑ Persistent Memory Development
 - ❑ Enable existing applications
 - ❑ Build something new



SDC¹⁸

September 24-27, 2018
Santa Clara, CA

www.storagedeveloper.org

Q&A