SOLUTION BRIEF

Distributed Asynchronous Object Storage (DAOS)
Intel® Optane™ Technology

DAOS: Revolutionizing High-Performance Storage with Intel® Optane™ Technology

With the exponential growth of data, distributed storage systems have become not only the heart, but also the bottleneck of data centers. High-latency data access, poor scalability, difficulty managing large datasets, and lack of query capabilities are just a few examples of common hurdles. Traditional storage systems have been designed for rotating media and for POSIX* input/output (I/O). These storage systems represent a key performance bottleneck, and they cannot evolve to support new data models and next-generation workflows.

The Convergence of HPC, Big Data, and AI

Storage requirements have continued to evolve, with the need to manipulate ever-growing datasets driving a further need to remove barriers between data and compute. Storage is no longer driven by traditional workloads with large streaming writes like checkpoint/restart, but is increasingly driven by complex I/O patterns from new storage pillars. High-performance data-analytics workloads are generating vast quantities of random reads and writes. Artificial-intelligence (AI) workloads are reading far more than traditional high-performance computing (HPC) workloads. Data streaming from instruments into an HPC cluster require better quality of service (QoS) to avoid data loss. Data-access time is now becoming as critical as write bandwidth. New storage semantics are required to query, analyze, filter, and transform datasets. A single storage platform in which next-generation workflows combine HPC, big data, and AI to exchange data and communicate is essential.

DAOS Software Stack

Intel has been building an entirely open source software ecosystem for data-centric computing, fully optimized for Intel® architecture and non-volatile memory (NVM) technologies, including Intel® Optane™ DC persistent memory and Intel Optane DC SSDs. Distributed Asynchronous Object Storage (DAOS) is the foundation of the Intel exascale storage stack. DAOS is an open source software-defined scale-out object store that provides high bandwidth, low latency, and high I/O operations per second (IOPS) storage containers to HPC applications. It enables next-generation data-centric workflows that combine simulation, data analytics, and AI.

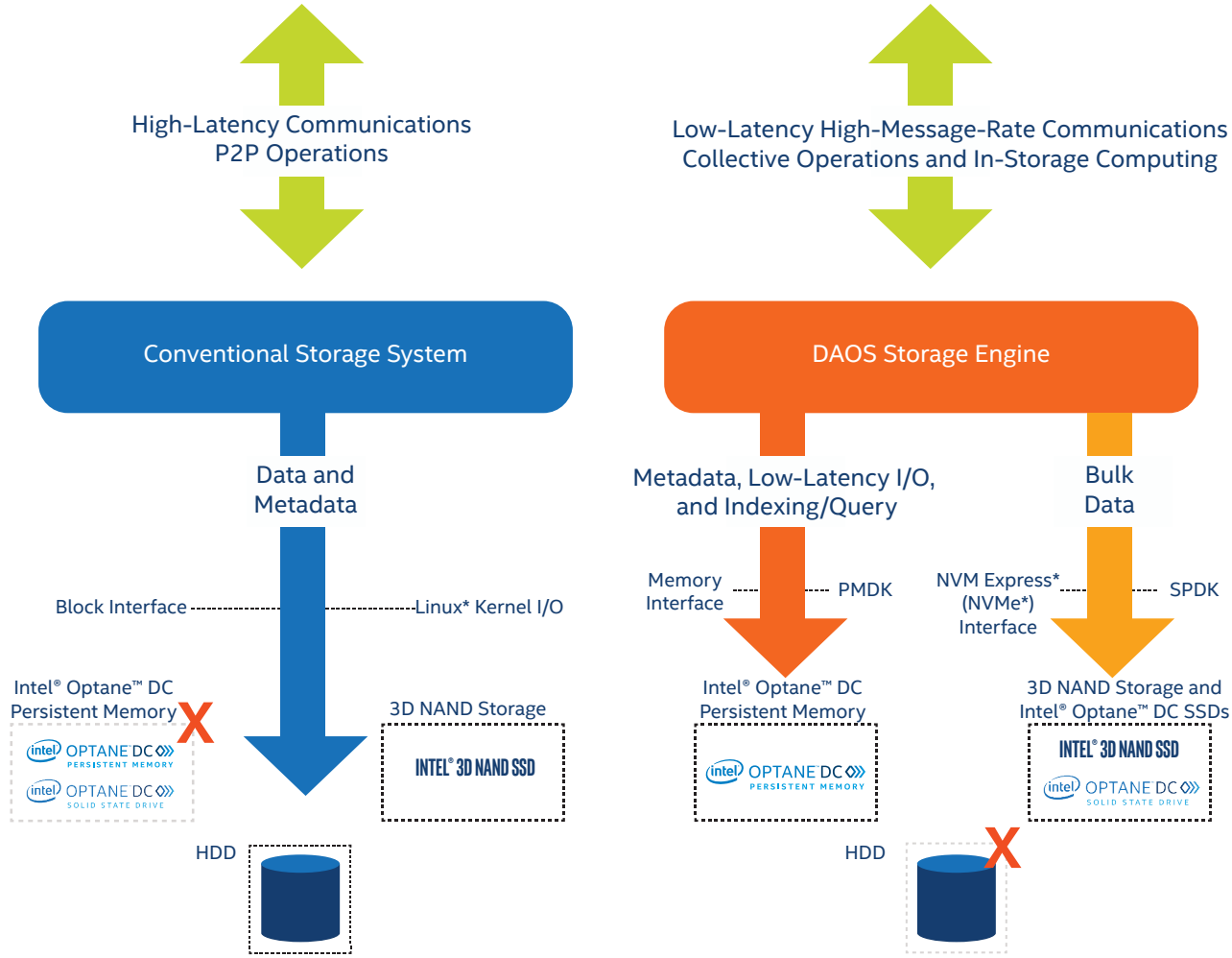

Figure 1. DAOS architecture versus conventional storage systems

Unlike traditional storage stacks that were primarily designed for rotating media, DAOS is architected from the ground up to make use of new NVM technologies, and it is extremely lightweight because it operates end-to-end in user space with full operating system bypass. DAOS offers a shift away from an I/O model designed for block-based, high-latency storage to one that inherently supports fine-grained data access and unlocks the performance of next-generation storage technologies. Figure 1 presents an overview of the DAOS architecture in comparison with existing storage systems.

Existing distributed storage systems use high-latency peer-to-peer communication, whereas DAOS is designed to use low-latency, high-message-rate user-space communications that bypass the operating system. Most storage systems today are designed for block I/O, where all I/O operations go through the Linux* kernel with a block interface. Much work has been done to optimize access to the block device (such as coalescing, buffering, and aggregation). But all those optimizations are not relevant for the next-generation storage devices that Intel is targeting, and they will incur unnecessary overhead if used. DAOS, on the other hand, is designed to optimize access to Intel Optane DC persistent memory and NVM Express* (NVMe*) solid state drives (SSDs), and it eschews this unnecessary overhead.

DAOS servers maintain their metadata on persistent memory, with bulk data going straight to NVMe SSDs. In addition, small I/O operations will be absorbed on the persistent memory before being aggregated and then migrated to the larger-capacity flash storage. DAOS uses the Persistent Memory Development Kit (PMDK) to provide transactional access to persistent memory and the Storage Performance Development Kit (SPDK) for user-space I/O to NVMe devices.[1,2] **This architecture allows for data-access times that can be several orders of magnitude faster than in existing storage systems (microseconds [µs] versus milliseconds [ms]).**

The DAOS software stack provides:

- Ultra-fine grained, low-latency, and true zero-copy I/O
- Non-blocking data and metadata operations to allow I/O and computation to overlap
- Advanced data placement to account for fault domains
- Software-managed redundancy supporting both replication and erasure code with online rebuild
- End-to-end (E2E) data integrity
- Scalable distributed transactions with guaranteed data consistency and automated recovery
- Dataset snapshot capability
- Security framework to manage access control to storage pools
- Software-defined storage management to provision, configure, modify, and monitor storage pools
- Native support for I/O middleware libraries like HDF5*, MPI-IO*, and POSIX over the DAOS data model and API, removing the need for applications to port their code to use DAOS APIs directly
- Apache Spark* integration
- Native producer/consumer workflows using publish/subscribe APIs
- Data indexing and query capabilities
- In-storage computing to reduce data movement between storage and compute nodes
- Tools for disaster recovery
- Seamless integration with the Lustre* parallel file system, with the ability to extend for other parallel file systems to provide a unified namespace for data access across multiple storage tiers
- Data mover agent to migrate datasets among DAOS pools, from parallel file systems to DAOS, and vice versa

As shown in Figure 2, the DAOS software stack relies on a client-server model. I/O operations are handled in the DAOS library linked directly with the application, and they are serviced by storage services running in user space on the DAOS server node (DN).

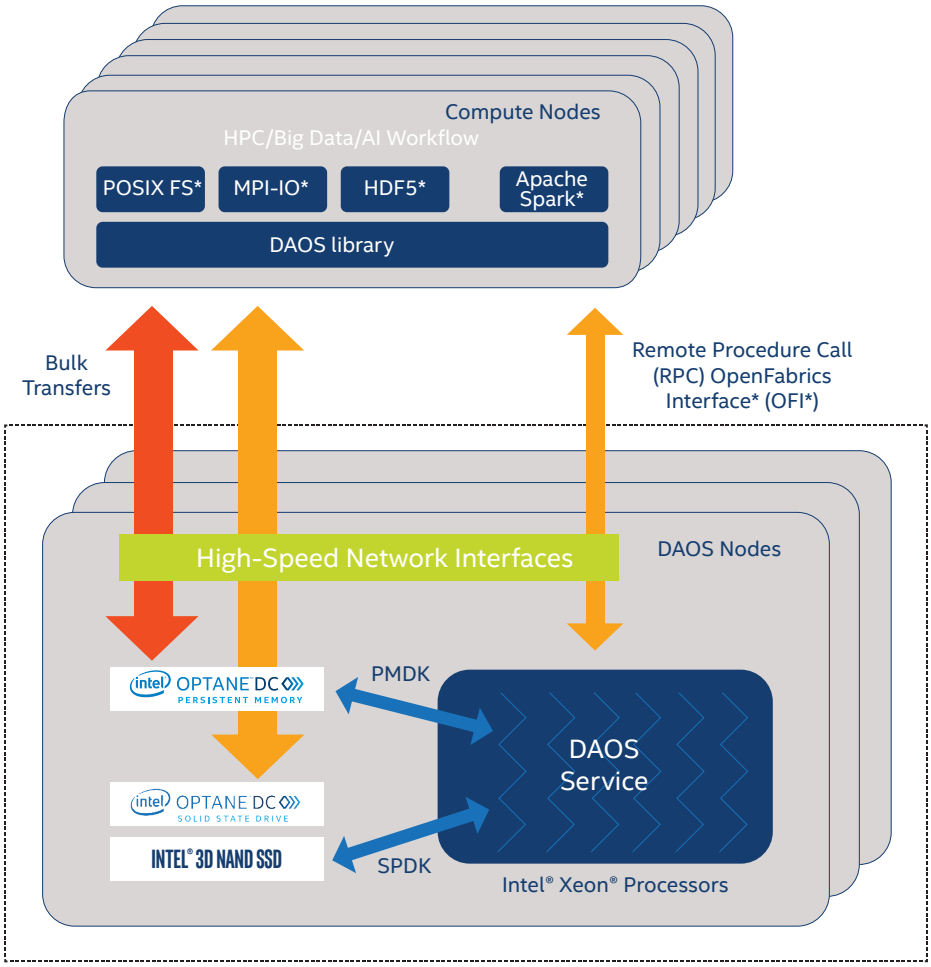

Figure 2. DAOS software stack

Application Interface and I/O Middleware Integration

The DAOS client library is designed to have a small footprint, to minimize noise on the compute nodes, and to support non-blocking operations with explicit progress. The DAOS operations are function-shipped to the DAOS storage servers using libfabrics* and OpenFabric Interface* (OFI*), taking advantage of any remote direct memory access (RDMA) capabilities in the fabric.

In this new storage paradigm, POSIX is no longer the foundation for new data models. Instead, the POSIX interface is built as a library on top of the DAOS back-end API, like any other I/O middleware. A POSIX namespace can be encapsulated in a container and mounted by an application into its file system tree. This application-private namespace will be accessible to any tasks of the application that successfully opened the container. Tools to parse the encapsulated namespace will be provided. Both the data and metadata of the encapsulated POSIX file system will be fully distributed across all the available storage with a progressive layout to help ensure both performance and resilience. In addition, the POSIX emulation features the following: scalable directory operations, scalable shared file I/O, scalable file-per-process I/O, and self-healing to recover from failed or corrupted storage.

While most HPC I/O middleware could run transparently over a DAOS backend via the POSIX emulation layer, migrating I/O middleware libraries to support the DAOS API natively will take advantage of DAOS's rich API and advanced features. Figure 3 represents the envisioned DAOS ecosystem.

DAOS containers are exposed to applications through several I/O middleware libraries, providing a smooth migration path with minimal (or sometimes no) application changes. Middleware I/O libraries that run on top of the DAOS library include:

- **POSIX FS:** DAOS offers two operating modes for POSIX support. The first is for well-behaved applications that generate conflict-free operations for which a high level of concurrency is supported. The second mode is for applications that require stricter consistency at the cost of performance.
- **MPI-I/O:** A ROMIO* driver supports MPI-I/O on top of DAOS. All applications or middleware I/O libraries that use MPI-I/O as their I/O backend can use that driver seamlessly on top of DAOS. The driver has been pushed upstream to the MPICH* repository. This driver is portable to other MPI implementations that use ROMIO as the I/O implementation for the MPI-IO standard. The DAOS MPI-IO driver is built directly over the DAOS API.
- **HDF5:** An HDF5 Virtual Object Layer (VOL) connector uses DAOS to implement the HDF5 data model. Applications that use HDF5 to represent and access their data can use the VOL plugin with minimal to no code changes with the existing HDF5 APIs to replace the traditional binary format in a POSIX file with a DAOS container. This connector implements the official HDF5 API with a native DAOS backend. Internally, the HDF5 library manages DAOS transactions and provides consistency from H5Fopen() to H5Fflush()/H5Fclose(). New features like asynchronous I/O, snapshot, and query/indexing are provided as API extensions.

Additional HPC I/O middleware like Silo*, MDHIM*, and Dataspaces* can benefit from a native port over the DAOS API. Intel is also collaborating with other organizations (for example, weather forecasting) and industry leaders (such as in the entertainment industry, cloud, and oil and gas) to support new data models over DAOS.

Finally, Intel is looking into enabling DAOS in big data and analytics frameworks, and, more specifically, having a DAOS back end for Apache Arrow*. The Apache Arrow standard defines the data to be stored in columnar vectors to support data-analytics use cases. The purpose of this standard is to define a standard for other data-analytics systems like Apache Spark, Apache Thrift*, and Apache Avro*. Right now, each of these systems has its own format, but by using the common Apache Arrow format, there would be no need for serialization/deserialization of data to be shared between those systems. Apache Arrow is meant as a component to tightly integrate other big data and analytics systems. Apache Arrow also provides I/O APIs to store files on disk. At this time, this works on the Apache Hadoop* Distributed File System (HDFS*) in an Apache Hadoop ecosystem. A DAOS plugin for Apache Arrow that converted the Apache Arrow format in-memory to a DAOS container would make more applications suitable for an HPC system.

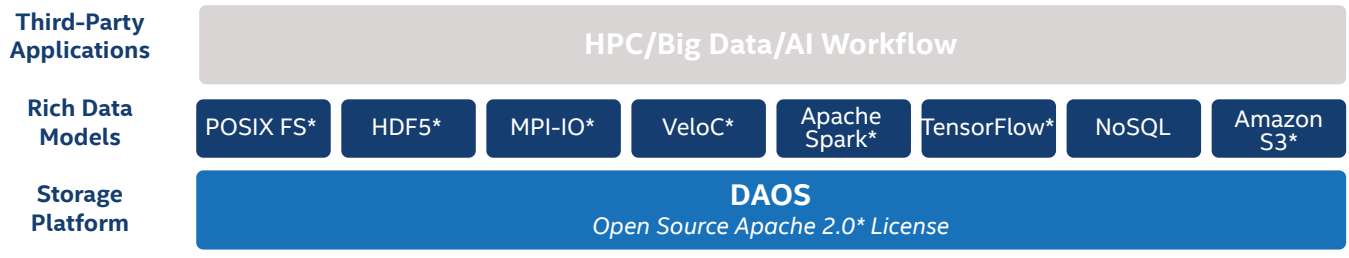

Figure 3. DAOS middleware ecosystem

DAOS Deployment and Roadmap

DAOS is available on GitHub* (https://github.com/daos-stack/) under the Apache 2.0* license. Instructions on how to install, configure, and administrate a DAOS installation are available in the DAOS admin guide (see http://daos.io/doc).
A new DAOS version is planned every six months; check the DAOS roadmap for more information (http://daos.io/roadmap). Issues should be reported via https://jira.hpdd.intel.com, with a reproducer whenever applicable.
A community mailing list is also available at https://daos.groups.io.

Conclusion

With ultra-low latency and fine-grained access to persistent storage, Intel Optane DC persistent memory represents a real opportunity to transform the industry and overcome the limitations of storage systems in data centers today. Intel Optane DC SSDs improve the solution further, bringing high IOPS and handling reads and writes concurrently without degradation. Existing distributed storage software, however, was not built for these new technologies, and it can mask the value the technologies provide. A complete rethink of the software storage stack is required to design a new solution from the ground up in order to throw off irrelevant optimizations designed for disk drives, embrace fine-grained and low-latency storage access with rich storage semantics, and unlock the potential of these revolutionary technologies for distributed storage.

[1] "Persistent Memory Programming." http://pmem.io/pmdk/.
[2] "Storage Performance Development Kit." spdk.io/.

Printed in USA 0619/DGS/PRW/PDF Please Recycle 340581-001US