# DAOS Version 1.0 Release Notes

We are pleased to announce the release of DAOS version 1.0, a key DAOS milestone focused on a newly created high-performance object store that is ultimately shifting the HPC and big data paradigm. Recently awarded the top spot in the [IO500 10-node Challenge](https://www.vi4io.org/io500/list/19-11/10node), DAOS is fully optimized for Intel(R) architecture and non-volatile memory (NVM).

This release is targeted towards benchmarking, partner integration, evaluation, and brings support for the following features:

* NVMe and Persistent Memory support
* Per-pool ACL
* Certificate Support
* Unified Namespace via dfuse
* MPI-IO ROMIO Driver
* HDF5 Support
* Basic POSIX I/O Support
* Replication and Self-Healing (Preview)

This release is not yet intended for production use.

DAOS is an open-source, software-defined storage (SDS) and storage-as-a-service (STaaS) platform. This technology was developed from the ground up for nextgen NVM technologies like Intel(R) 3D NAND Technology and Intel(R) Optane technology. DAOS operates end-to-end in user-space for ease of deployment and to maximize IOPS, bandwidth, and minimize latency. DAOS is a scale-out solution allowing small and medium deployments to grow as much as required (i.e., 100’s of PB with millions of client nodes and thousands of storage servers) to serve the most demanding scientific, Big Data and AI workloads.

The primary storage for the first US Exascale supercomputer (the Aurora system at Argonne National Laboratory - ANL) will be a 230PB DAOS tier composed to deliver an aggregated bandwidth of at least 25TB/s. Intel is collaborating closely with different partners to propose DAOS-based solutions for production deployments. DAOS is gaining traction among the enterprise and hyperscalers.

Designed to address evolving storage needs, DAOS is a new storage platform for the convergence of traditional modeling and simulation, data science analytics, and artificial intelligence.

Visit the [DAOS github.io](https://daos-stack.github.io/) page for more information. All DAOS project source code is maintained in the <https://github.com/daos-stack/daos> repository. Please visit this [link](https://wiki.hpdd.intel.com/display/DC/DAOS+v1.0+Licensing) for more information on the licenses.

## Software Dependencies

Reference the [Software Installation](https://daos-stack.github.io/admin/installation/) section of the [DAOS Administration Guide](https://daos-stack.github.io/admin/hardware/) for more details.

## Distribution Packages

DAOS RPMs are currently being developed for DAOS v1.0. Instructions for obtaining the RPMs will be provided in this document at a future date, and will be posted in the [DAOS mailing list](https://daos.groups.io/g/daos).

## Build Prerequisites

To build DAOS and its dependencies, several software packages must be installed on the system. This includes scons, libuuid, cmocka, ipmctl, and several other packages usually available on all the Linux distributions. Moreover, a Go version of at least 1.10 is required.

[CentOS](https://github.com/daos-stack/daos/blob/master/utils/docker/Dockerfile.centos.7#L53-L76) is currently the only supported Linux distribution. An exhaustive list of packages for each supported Linux distribution is maintained in the Docker files. Refer to the [Software Installation](https://daos-stack.github.io/admin/installation/) section of the [DAOS Administration Guide](https://daos-stack.github.io/admin/hardware/) for more details.

## Hardware Support

### Processor Requirements

DAOS requires a 64-bit processor architecture and is primarily developed on Intel 64 architecture. The DAOS software and the libraries it depends on (e.g., ISA-L, SPDK, PMDK, and DPDK) can take advantage of Intel SSE and AVX extensions.

### Network Requirements

The DAOS network layer relies on libfabrics and supports OFI providers for Ethernet/sockets, InfiniBand/verbs. An RDMA-capable fabric is preferred for better performance. DAOS can support multiple rails by binding different instances of the DAOS server to individual network cards.

The DAOS control plane provides methods for administering and managing the DAOS servers using a secure socket layer interface. An additional out-of-band network connecting the nodes in the DAOS service cluster is required for DAOS administration. Management traffic between clients and servers use IP over Fabric.

### Storage Requirements

DAOS requires each storage node to have direct access to storage-class memory (SCM). While DAOS is primarily tested and tuned for Intel(R) Optane DC Persistent Memory, the DAOS software stack is built over the Persistent Memory Development Kit (PMDK) and the DAX feature of the Linux operating systems as described in the [SNIA NVM Programming Model](https://www.snia.org/sites/default/files/technical_work/final/NVMProgrammingModel_v1.2.pdf). As a result, the open-source DAOS software stack should be able to run transparently over any storage-class memory supported by the PMDK.

The storage node should be equipped with NVMe (non-volatile memory express) SSDs to provide capacity. HDDs, as well as SATA and SAS SSDs, are not supported by DAOS. Both NVMe 3D-NAND and Optane SSDs are supported. NVMe-oF devices should theoretically work with the user-space storage stack but have not been tested.

A recommended ratio of 6% SCM to SSD capacity will guarantee that DAOS has enough space in SCM to store its internal metadata (e.g., pool metadata, SSD block allocation tracking).

For testing purposes, SCM can be emulated with DRAM by mounting a tmpfs filesystem, and NVMe SSDs can be also emulated with DRAM or a loopback file. Any data written using a tmpfs will not be persistent across a reboot.

## DAOS Testing

DAOS 1.0 validation efforts were focused on anticipated initial use cases including: system integration, application porting, benchmarking, initial evaluations, and the like. This release is not intended for large scale production use and testing to date has not focused on use cases or hardware typically found in HPC production environments. Testing has been completed in the following areas:

* Testing has been performed on Centos 7.7 and SLES 15 with Centos being used in the majority of the test cycles.
* Testing has been conducted using Intel Xeon processors, Intel 3D NAND and Optane based NVMe SSDs and Optane persistent memory modules although generally storage density was not tested at production levels.
* DAOS uses the libfabric network abstraction layer and testing has been performed on a number of network specific providers, including the IB verbs, OPA PSM2, socket and TCP providers. Network testing is on-going and none of the above providers should be considered fully tested at this time.
* All DAOS 1.0 supported functionality has been tested with an emphasis on use cases with positive outcomes error cases (e.g. DAOS server failure) have limited test cycles at this time.
* Maximum scale-out of DAOS servers during test runs was 128. Maximum scale-out of DAOS clients was 2048.
* Soak testing with an emphasis on I/O jobs in combination with basic administrative actions has been run and found to be error free for periods up to 48 hours. As with functional testing the focus has been on positive path testing with failure paths and fault injection coming in a future release.

## Version 1.0 major features

### NVMe and Persistent Memory Support

DAOS supports two tiers of storage: Storage Class Memory (SCM) and NVMe SSDs. Each DAOS server will be equipped with SCM (for byte-granular application data and metadata) along with NVMe SSDs (for DAOS application bulk data, ie >4KB). Similar to how PMDK is used to facilitate access to SCM, the Storage Performance Development Kit (SPDK) is used to provide seamless and efficient access to NVMe SSDs. DAOS utilizes the significant performance increase of the SPDK user space NVMe driver over the standard NVMe kernel driver.

As a part of the NVMe support, an extent-based block allocator was implemented and designed specifically for DAOS NVMe block device space management, including efficient management of smaller 4KB block allocations. A server module was also implemented for issuing I/O over NVMe SSDs. This involves internally managing a per-xstream DMA-safe buffer for SPDK DMA transfer over NVMe SSDs. The module also persistently tracks important server metadata. This per-server metadata includes the health state of each NVMe SSD, as well as the mapping between NVMe SSDs, DAOS server xstreams, DAOS pools, and allocated blocks IDs.

Two other key components include faulty device detection and device health monitoring. DAOS handles storing NVMe SSD health data, including raw NVMe device health as well as I/O error and checksum errors counters. If and when an I/O error occurs, an event notification will be sent to the console to notify the administrator. Management utility commands are also available for administrators to query all NVMe device health stats to gauge the general health of the system. DAOS currently only supports manual faulty device events, with future work including auto-detection upon configurable faulty criteria. This would involve an administrator manually setting the device state of an NVMe SSD to FAULTY using the management utility, which will trigger the appropriate rebuild of data and exclusion of the faulty device from the system. Reintegration and hotplug of NVMe SSDs are not currently supported in DAOS 1.0 but will be a part of a future release.

### Per-Pool ACL

Access to DAOS data is controlled through a simple identity based access control mechanism specified on a per pool basis. The access credentials for a DAOS client process are determined by the agent running on the compute node. This credential is validated against the Access Control List present on the DAOS server to determine the level of access granted to a specific user. The ACL mechanism support access permissions on an individual and group basis as well as a mechanism for supporting fall through users. As of DAOS 1.0, the set of permissions on a pool are very simple but will be extended when container-based ACLs are introduced in a future version.

### Certificate Support

Securing the transport of data and authorizing actions by components make up the backbone of the security model used in the DAOS control plane. All control plane components communicate with each other using mutual TLS authentication (mTLS) backed by certificates signed by a per cluster certificate authority. In addition to securing, the communications between components, certificates are also used to authenticate the various components and ensure that requests only come from components who are authorized to perform those actions. Certificates are also used to secure the administrative interface to the cluster, ensuring only admins with the proper credentials can perform administrative actions.

### Unified Namespace (UNS) in DAOS via dfuse

Unified NameSpace (UNS) is the ability to create relationships linking paths in the namespace tree to other DAOS containers, allowing greater flexibility of use and new workflows. UNS entry points can be created that behave in a similar way to hard links or submounts, where an attempt to traverse the entry point via dfuse or DAOS aware tools will automatically and seamlessly follow the link to the specified container. UNS entry points are created by providing a path to the DAOS container create command, and can exist either within existing containers or at locations within the regular POSIX filesystem.

### MPI-IO ROMIO Driver

The MPI standard includes specification for a low level IO interface for parallel access to files. The MPI-IO standard loosens some of the POSIX semantics that are not required for many applications and defines routines and methods for parallel access to files from processes in an MPI communicator. Several existing applications and other high level middleware IO libraries use the MPI-IO standard as a backend for data access. ROMIO is the de-facto MPI-IO implementation that is released with the MPICH library and is used by most MPI implementations to support the MPI-IO standard. ROMIO exposes an ADIO module interface for different backends. A DAOS backend was implemented as an alternative to the POSIX backend for direct user-space access to the DAOS storage stack through the MPI-IO interface. The DAOS ROMIO driver is distributed with the MPICH ROMIO source code.

### HDF5 Support

HDF5 is a data model, library, and file format for storing and managing data. It supports an unlimited variety of datatypes, and is designed for flexible and efficient I/O and for high volume and complex data. This DAOS release supports the HDF5 format, API, and tools through the HDF5 (sec2) POSIX backend or the HDF5 MPI-IO backend. HDF5 also provides a new DAOS VOL for mapping the HDF5 data model and API directly over the DAOS data model, bypassing the byte array serialization required over the POSIX and MPI-IO backends. The HDF5 DAOS VOL is not fully supported in this DAOS release and should be used as a prototype for basic testing of this new HDF5 feature.

### Basic POSIX I/O Support

POSIX IO is the main building block for all applications and IO libraries. The DAOS library provides POSIX support through the DFS (DAOS File System) library. The DFS API provides an encapsulated namespace with a POSIX like API directly on top of the DAOS API. Applications can link directly to the DFS library and use the DFS API for direct user-space access to the DAOS stack through the DFS API. A FUSE plugin over the DFS library (dfuse) is also provided to support direct access to POSIX calls through FUSE over the DAOS stack. The primary support provided in this release provides loose POSIX consistency in terms of metadata and assumes applications generate conflict-free operations, otherwise, the behavior is undefined. FUSE adds stricter consistency on top of the DFS layer, but that consistency is limited to a node local instance of the dfuse mount.

### Replication and self-healing (preview)

In DAOS, if the data is replicated with multiple copies on different targets, once one of the targets fail, the data on it will be rebuilt on the other targets. This reduces the data redundancy that would be impacted by the target failure. Self-healing in DAOS 1.0 is a preview feature, and it will not be enabled automatically when targets are failed; instead it can only be enabled manually by the dmg exclude command. In future versions, DAOS will support erasure coding to protect the data. At that time, the rebuild process will be updated accordingly.

## Document updates

All documents supporting the <https://daos-stack.github.io/> site have been refreshed and re-organized for this release. Design documents in the DAOS Source repository have also been refreshed for this release.

## Known Issues

### The dmg storage scan may report incorrect NUMA socket ID

The ndctl package maintainers have confirmed that this is a kernel regression that has been resolved in CentOS 8. Some DAOS nodes may not report the correct NUMA socket ID when running a “dmg storage scan.” This appears to be a regression in the CentOS7.7 kernel rather than an ndctl issue. This is due to different versions of ndctl provisioning different JSON namespace details (storage scan command reads the “numa\_node” field). The regression also affects the operation of the “dmg config generate” command which isn’t able to detect correct NUMA affinity for the PMem namespaces required for the config. The regression has been fixed in the CentOS7.9 kernel.

### PSM over OPA is not fully functional

If you must evaluate DAOS on an OPA fabric, we recommend using IP over fabric/sockets for evaluation. For benchmarking, PSM2 over OPA offers a better picture of DAOS performance, but the PSM2 provider is not fully functional. For a discussion of issues and options, contact us via the [mailing list](https://daos.groups.io/g/daos).

### Replication and Self-healing

While replication and self-healing features have been fully implemented, those features haven’t been thoroughly tested in 1.0. As a result, they are supported as preview features and will be fully supported in the next release. The storage node fault detection has thus been disabled by default and failed storage nodes have to be excluded manually to trigger automatic repair. Please refer to the [Pool Modifications] (https://daos-stack.github.io/admin/pool\_operations/#pool-modifications) section of the administration guide for more information.

### Limited bandwidth to SCM with some update/write operations

With a configuration of only Intel(R) Optane Pmem for storage in a DAOS pool (no SSDs), a known limitation currently exists on the write/update performance with bulk transfers to the SCM with all network providers which limits the BW per storage server.

### Aggregation Issue

The aggregation service is only activated after the system has idled for more than 5 seconds, so the aggregation is always suspended if there are sustained I/O requests from application. As a result, storage node will run out of SCM space even there is plenty of free space in SSD.

## Fixed Issues Details

First release, this section will be updated in subsequent releases

## Change Log

No changes for this first release.