Spectrum Scale and NVMe Storage

Michael Hennecke | SSUG::Digital at CIUK 2020, 04-Dec-2020

Agenda

- Optimizing NAND Flash usage with **TRIM Support** in Scale **5.0.4** (fs level 22.00)
  - Freeing unused NAND Flash space with the `mmreclaimspace` command
  - For classical NSDs only ... does **not** (yet) apply to Spectrum Scale RAID

- Spectrum Scale „SAN Mode“ with **NetApp EF600**
  - Using NVMe-over-Fabrics to eliminate the NSD server layer

- **DAOS** (Distributed Asynchronous Object Storage) Unified Namespace
  - Mounting DAOS POSIX containers into Spectrum Scale

TRIM Support

Optimizing NAND Flash Usage with `mmreclaimspace`

NAND Flash Media – The Problem Statement

- Two challenges with NAND Flash storage media:
  1. Cannot overwrite a sector in-place. NAND needs to be cleared in large „erasure blocks“.
  2. Endurance: NAND Flash cells wear out → Limited number of program/erase cycles.
- How these challenges are addressed:
  1. Overprovisioning. All NAND SSDs are overprovisioned; 3 DWPD more so than 1 DWPD.
  2. Background garbage collection. Causes „write amplification“. May not be able to keep up...
- What makes these challenges worse:
  1. Higher NAND density reduces cell endurance (SLC → MLC → TLC → QLC).
  2. More non-sequential workloads require more garbage collection.
  3. **SSD controllers do not know which sectors the file system uses, and which are free**
- The ATA **„TRIM“ command** (or SCSI „unmap“, or NVMe „deallocate“) allows the filesystem to communicate **unused LBAs** to the SSD controller
  - Helps with Write Amplification, Performance, and Garbage Collection
  - The devices have to support this ... Intel DC SSDs **do**, RAID controllers often do **not**.

Lenovo

TRIM Support in Spectrum Scale 5.0.4 (Step 1)

- The **%nsd** stanza input to **mmcrnsd** needs to specify TRIM support:

```
%nsd: device=DiskName nsd=NsdName servers=ServerList
        usage={dataOnly | metadataOnly | dataAndMetadata | descOnly | localCache}
        failureGroup=FailureGroup pool=StoragePool
        thinDiskType={no | nvme | scsi | auto}

      Specifies the space reclaim disk type:
        no     The disk device supports space reclaim.
               This value is the default.
        nvme   The disk is a TRIM capable NVMe device
               that supports the mmreclaimspace command.
        scsi   The disk is a thin provisioned SCSI disk
               that supports the mmreclaimspace command.
        auto   The type of the disk is either nvme or scsi.
               IBM Spectrum Scale will try to detect the actual disk type automatically.
```

Typo in man mmcrnsd:
„does not support space reclaim“
(man mmcrfs is correct.)

- **TODO**: Clarify if this setting can be changed after the NSD has been created...

TRIM Support in Spectrum Scale 5.0.4 (Step 2)

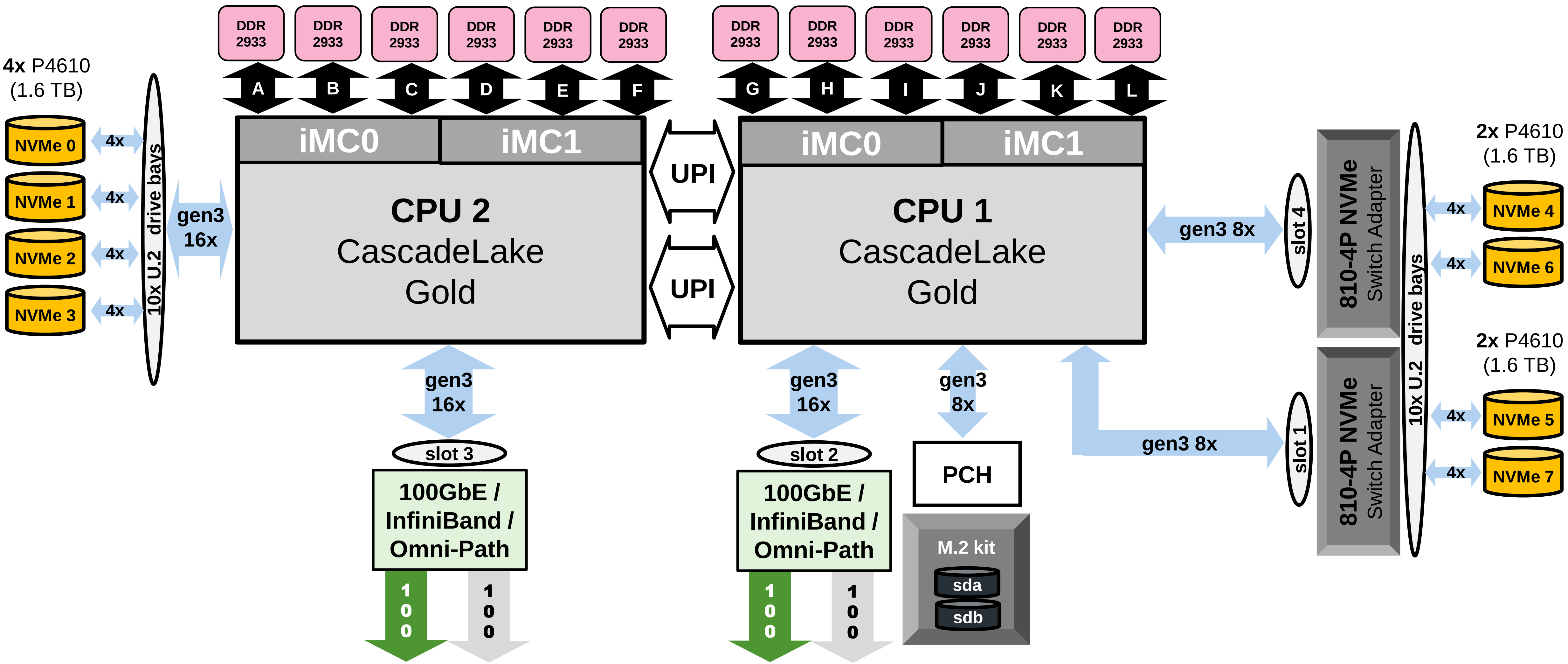
- No **implicit** / automatic space reclaim by Spectrum Scale at this time.
- Need to **explicitly** invoke the **`mmreclaimspace`** command:

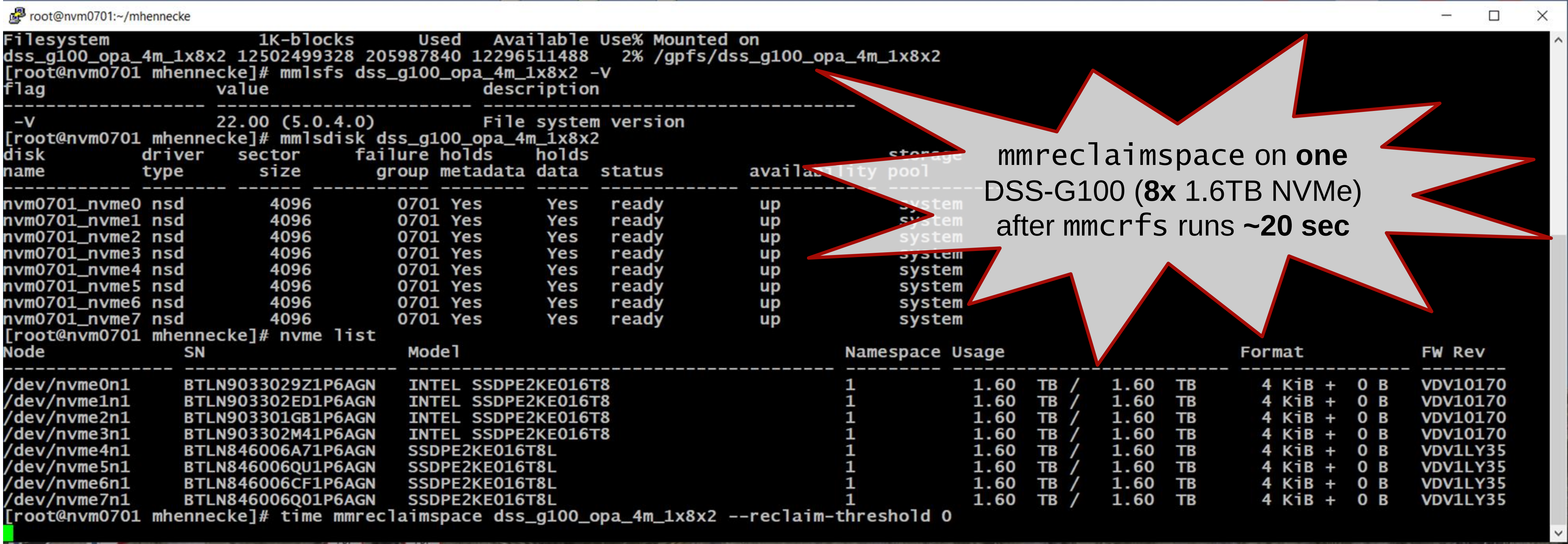
```
mmreclaimspace Device [-Y] [-P PoolName]
  [-qos {maintenance | other}]
  {--reclaim-threshold Percentage | --emergency-reclaim}
```

- Use Percentage=0 to reclaim all unused space. Use 90 for lighter load.
  - This command can be „I/O heavy“, as it sends all affected LBA ranges to the disks...

- Documentation: „*IBM Spectrum Scale with data reduction storage devices”* in: IBM Spectrum Scale: Concepts, Planning, and Installation Guide.
  - See also „*Chapter 18. File system format changes between versions of IBM Spectrum Scale*“ in: IBM Spectrum Scale: Administration Guide

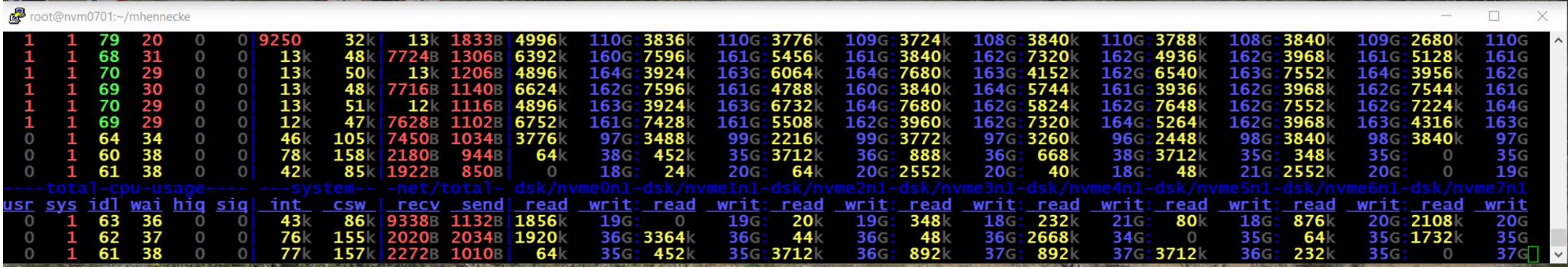
Lenovo DSS-G100 NVMe-rich Server: ThinkSystem SR630



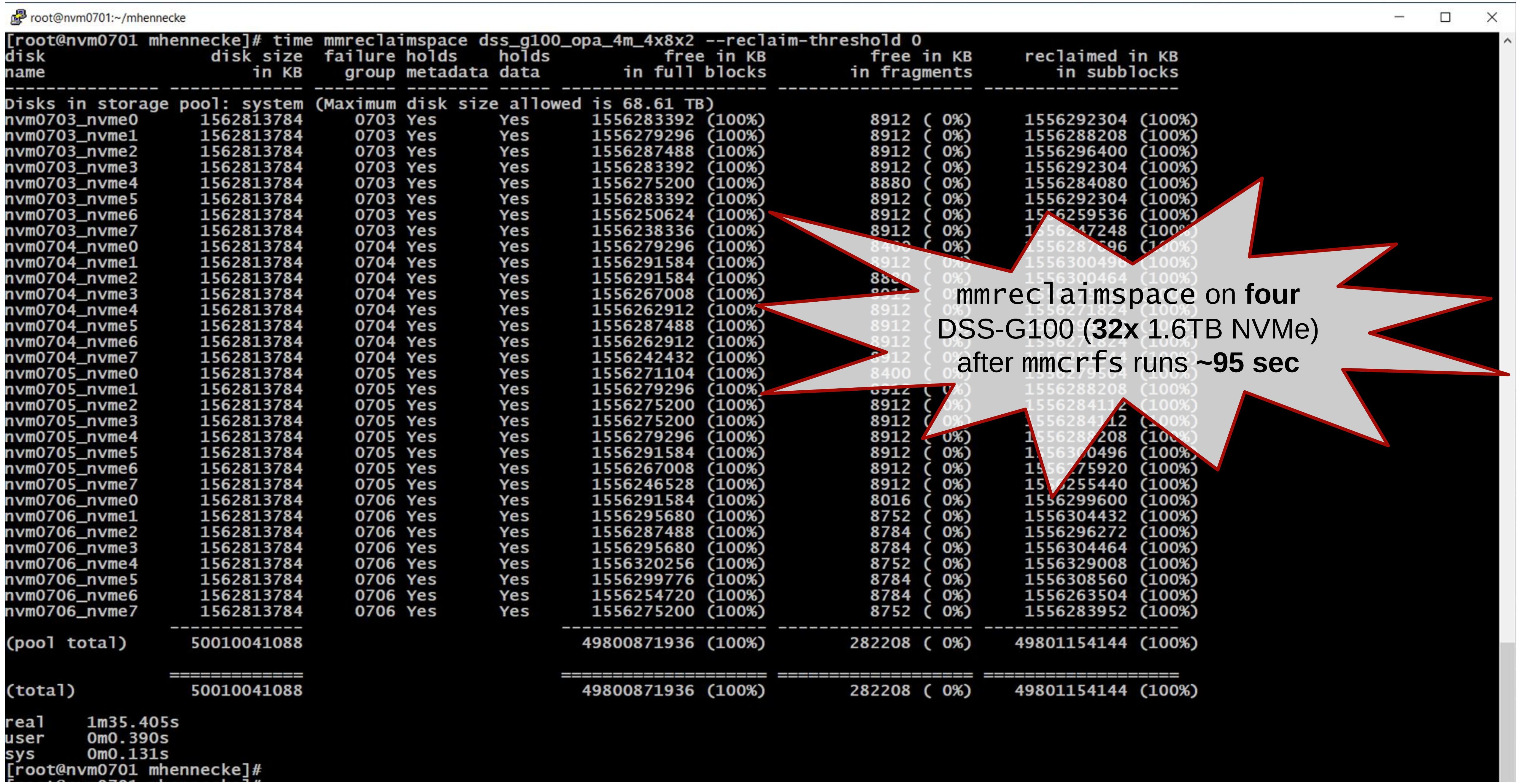


```
Filesystem          1K-blocks      Used   Available Use% Mounted on
dss_g100_opa_4m_1x8x2 12502499328 205987840 12296511488   2% /gpfs/dss_g100_opa_4m_1x8x2
[root@nvm0701 mhennecke]# mmlsfs dss_g100_opa_4m_1x8x2 -V
flag                value                    description
------------------- ------------------------ -----------------------------------
 -V                 22.00 (5.0.4.0)          File system version
[root@nvm0701 mhennecke]# mmlsdisk dss_g100_opa_4m_1x8x2
disk         driver   sector     failure holds    holds                            storage
name         type       size       group metadata data  status        availability pool
------------ -------- ------ ----------- -------- ----- ------------- ------------ ------------
nvm0701_nvme0 nsd         4096        0701 Yes      Yes   ready         up           system
nvm0701_nvme1 nsd         4096        0701 Yes      Yes   ready         up           system
nvm0701_nvme2 nsd         4096        0701 Yes      Yes   ready         up           system
nvm0701_nvme3 nsd         4096        0701 Yes      Yes   ready         up           system
nvm0701_nvme4 nsd         4096        0701 Yes      Yes   ready         up           system
nvm0701_nvme5 nsd         4096        0701 Yes      Yes   ready         up           system
nvm0701_nvme6 nsd         4096        0701 Yes      Yes   ready         up           system
nvm0701_nvme7 nsd         4096        0701 Yes      Yes   ready         up           system
[root@nvm0701 mhennecke]# nvme list
Node             SN                   Model                                    Namespace Usage                      Format           FW Rev
---------------- -------------------- ---------------------------------------- --------- -------------------------- ---------------- --------
/dev/nvme0n1     BTLN9033029Z1P6AGN   INTEL SSDPE2KE016T8                      1           1.60  TB /   1.60  TB      4 KiB +  0 B   VDV10170
/dev/nvme1n1     BTLN903302ED1P6AGN   INTEL SSDPE2KE016T8                      1           1.60  TB /   1.60  TB      4 KiB +  0 B   VDV10170
/dev/nvme2n1     BTLN903301GB1P6AGN   INTEL SSDPE2KE016T8                      1           1.60  TB /   1.60  TB      4 KiB +  0 B   VDV10170
/dev/nvme3n1     BTLN903302M41P6AGN   INTEL SSDPE2KE016T8                      1           1.60  TB /   1.60  TB      4 KiB +  0 B   VDV10170
/dev/nvme4n1     BTLN846006A71P6AGN   SSDPE2KE016T8L                           1           1.60  TB /   1.60  TB      4 KiB +  0 B   VDV1LY35
/dev/nvme5n1     BTLN846006QU1P6AGN   SSDPE2KE016T8L                           1           1.60  TB /   1.60  TB      4 KiB +  0 B   VDV1LY35
/dev/nvme6n1     BTLN846006CF1P6AGN   SSDPE2KE016T8L                           1           1.60  TB /   1.60  TB      4 KiB +  0 B   VDV1LY35
/dev/nvme7n1     BTLN846006Q01P6AGN   SSDPE2KE016T8L                           1           1.60  TB /   1.60  TB      4 KiB +  0 B   VDV1LY35
[root@nvm0701 mhennecke]# time mmreclaimspace dss_g100_opa_4m_1x8x2 --reclaim-threshold 0
```

mmreclaimspace on **one**
DSS-G100 (**8x** 1.6TB NVMe)
after mmcrfs runs **~20 sec**



```
  1   1  79  20   0   0|9250    32k|  13k 1833B|4996k  110G:3836k  110G:3776k  109G:3724k  108G:3840k  110G:3788k  108G:3840k  109G:2680k  110G
  1   1  68  31   0   0|  13k   48k|7724B 1306B|6392k  160G:7596k  161G:5456k  161G:3840k  162G:7320k  162G:4936k  162G:3968k  161G:5128k  161G
  1   1  70  29   0   0|  13k   50k|  13k 1206B|4896k  164G:3924k  163G:6064k  164G:7680k  163G:4152k  162G:6540k  163G:7552k  164G:3956k  162G
  1   1  69  30   0   0|  13k   48k|7716B 1140B|6624k  162G:7596k  161G:4788k  160G:3840k  164G:5744k  161G:3936k  162G:3968k  162G:7544k  161G
  1   1  70  29   0   0|  13k   51k|  12k 1116B|4896k  163G:3924k  163G:6732k  164G:7680k  162G:5824k  162G:7648k  162G:7552k  162G:7224k  164G
  1   1  69  29   0   0|  12k   47k|7628B 1102B|6752k  161G:7428k  161G:5508k  162G:3960k  162G:7320k  164G:5264k  162G:3968k  163G:4316k  163G
  0   1  64  34   0   0|  46k  105k|7450B 1034B|3776k   97G:3488k   99G:2216k   99G:3772k   97G:3260k   96G:2448k   98G:3840k   98G:3840k   97G
  0   1  60  38   0   0|  78k  158k|2180B  944B|  64k   38G: 452k   35G:3712k   36G: 888k   36G: 668k   38G:3712k   35G: 348k   35G:   0    35G
  0   1  61  38   0   0|  42k   85k|1922B  850B|   0    18G:  24k   20G:  64k   20G:2552k   20G:  40k   18G:  48k   21G:2552k   20G:   0    19G
----total-cpu-usage---- ---system-- -net/total- dsk/nvme0n1-dsk/nvme1n1-dsk/nvme2n1-dsk/nvme3n1-dsk/nvme4n1-dsk/nvme5n1-dsk/nvme6n1-dsk/nvme7n1
usr sys idl wai hiq siq| int   csw | recv  send| read  writ: read  writ: read  writ: read  writ: read  writ: read  writ: read  writ: read  writ
  0   1  63  36   0   0|  43k   86k|9338B 1132B|1856k   19G:   0    19G:  20k   19G: 348k   18G: 232k   21G:  80k   18G: 876k   20G:2108k   20G
  0   1  62  37   0   0|  76k  155k|2020B 2034B|1920k   36G:3364k   36G:  44k   36G:  48k   36G:2668k   34G:   0    35G:  64k   35G:1732k   35G
  0   1  61  38   0   0|  77k  157k|2272B 1010B|  64k   35G: 452k   35G:3712k   36G: 892k   37G: 892k   37G:3712k   36G: 232k   35G:   0    37G
```



root@nvm0701:~/mhennecke

```
[root@nvm0701 mhennecke]# time mmreclaimspace dss_g100_opa_4m_4x8x2 --reclaim-threshold 0
disk                disk size  failure holds    holds              free in KB            free in KB         reclaimed in KB
name                    in KB    group metadata data        in full blocks          in fragments           in subblocks
--------------- ------------- -------- -------- ----- -------------------- -------------------- --------------------
Disks in storage pool: system (Maximum disk size allowed is 68.61 TB)
nvm0703_nvme0      1562813784     0703 Yes      Yes         1556283392 (100%)          8912 ( 0%)     1556292304 (100%)
nvm0703_nvme1      1562813784     0703 Yes      Yes         1556279296 (100%)          8912 ( 0%)     1556288208 (100%)
nvm0703_nvme2      1562813784     0703 Yes      Yes         1556287488 (100%)          8912 ( 0%)     1556296400 (100%)
nvm0703_nvme3      1562813784     0703 Yes      Yes         1556283392 (100%)          8912 ( 0%)     1556292304 (100%)
nvm0703_nvme4      1562813784     0703 Yes      Yes         1556275200 (100%)          8880 ( 0%)     1556284080 (100%)
nvm0703_nvme5      1562813784     0703 Yes      Yes         1556283392 (100%)          8912 ( 0%)     1556292304 (100%)
nvm0703_nvme6      1562813784     0703 Yes      Yes         1556250624 (100%)          8912 ( 0%)     1556259536 (100%)
nvm0703_nvme7      1562813784     0703 Yes      Yes         1556238336 (100%)          8912 ( 0%)     1556247248 (100%)
nvm0704_nvme0      1562813784     0704 Yes      Yes         1556279296 (100%)          8400 ( 0%)     1556287696 (100%)
nvm0704_nvme1      1562813784     0704 Yes      Yes         1556291584 (100%)          8912 ( 0%)     1556300496 (100%)
nvm0704_nvme2      1562813784     0704 Yes      Yes         1556291584 (100%)          8880 ( 0%)     1556300464 (100%)
nvm0704_nvme3      1562813784     0704 Yes      Yes         1556267008 (100%)          8912 ( 0%)     1556275920 (100%)
nvm0704_nvme4      1562813784     0704 Yes      Yes         1556262912 (100%)          8912 ( 0%)     1556271824 (100%)
nvm0704_nvme5      1562813784     0704 Yes      Yes         1556287488 (100%)          8912 ( 0%)     1556296400 (100%)
nvm0704_nvme6      1562813784     0704 Yes      Yes         1556262912 (100%)          8912 ( 0%)     1556271824 (100%)
nvm0704_nvme7      1562813784     0704 Yes      Yes         1556242432 (100%)          8912 ( 0%)     1556251344 (100%)
nvm0705_nvme0      1562813784     0705 Yes      Yes         1556271104 (100%)          8400 ( 0%)     1556279504 (100%)
nvm0705_nvme1      1562813784     0705 Yes      Yes         1556279296 (100%)          8912 ( 0%)     1556288208 (100%)
nvm0705_nvme2      1562813784     0705 Yes      Yes         1556275200 (100%)          8912 ( 0%)     1556284112 (100%)
nvm0705_nvme3      1562813784     0705 Yes      Yes         1556275200 (100%)          8912 ( 0%)     1556284112 (100%)
nvm0705_nvme4      1562813784     0705 Yes      Yes         1556279296 (100%)          8912 ( 0%)     1556288208 (100%)
nvm0705_nvme5      1562813784     0705 Yes      Yes         1556291584 (100%)          8912 ( 0%)     1556300496 (100%)
nvm0705_nvme6      1562813784     0705 Yes      Yes         1556267008 (100%)          8912 ( 0%)     1556275920 (100%)
nvm0705_nvme7      1562813784     0705 Yes      Yes         1556246528 (100%)          8912 ( 0%)     1556255440 (100%)
nvm0706_nvme0      1562813784     0706 Yes      Yes         1556291584 (100%)          8016 ( 0%)     1556299600 (100%)
nvm0706_nvme1      1562813784     0706 Yes      Yes         1556295680 (100%)          8752 ( 0%)     1556304432 (100%)
nvm0706_nvme2      1562813784     0706 Yes      Yes         1556287488 (100%)          8784 ( 0%)     1556296272 (100%)
nvm0706_nvme3      1562813784     0706 Yes      Yes         1556295680 (100%)          8784 ( 0%)     1556304464 (100%)
nvm0706_nvme4      1562813784     0706 Yes      Yes         1556320256 (100%)          8752 ( 0%)     1556329008 (100%)
nvm0706_nvme5      1562813784     0706 Yes      Yes         1556299776 (100%)          8784 ( 0%)     1556308560 (100%)
nvm0706_nvme6      1562813784     0706 Yes      Yes         1556254720 (100%)          8784 ( 0%)     1556263504 (100%)
nvm0706_nvme7      1562813784     0706 Yes      Yes         1556275200 (100%)          8752 ( 0%)     1556283952 (100%)
                -------------                           -------------------- -------------------- --------------------
(pool total)      50010041088                            49800871936 (100%)        282208 ( 0%)    49801154144 (100%)

                =============                           ==================== ==================== ====================
(total)           50010041088                            49800871936 (100%)        282208 ( 0%)    49801154144 (100%)

real    1m35.405s
user    0m0.390s
sys     0m0.131s
[root@nvm0701 mhennecke]#
```

`mmreclaimspace` on **four** DSS-G100 (**32x** 1.6TB NVMe) after `mmcrfs` runs **~95 sec**

Feedback / Wish List (no, there‘s no RFE for it yet ☺)

- **Very useful feature, especially for more „random“ workloads**
- **`mmreclaimspace`** needs **performance scaling** improvements
  - Ensure that enough parallelism is used when reclaiming space on **many** devices
- Make space reclaim a **default** action at **`mmcrnsd`** and/or **`mmcrfs`** time
  - Because provisioning scratch filesystems „on the fly“ becomes more common...
  - If needed, can add an option to **not** do space reclaim, like XFS does:

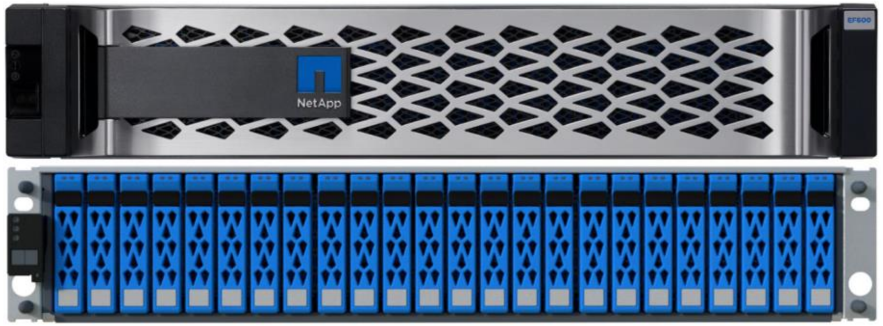
    ```
    mkfs.xfs  -K     Do not attempt to discard blocks at mkfs time.
    ```
- Provide **`mmchconfig` control** for more automatic reclaim. For example:
  - **`reclaimSpaceOnFileDelete {no|yes}`**
  - **`reclaimSpaceInterval {0|minutes}`**

    → Spectrum Scale 5.1 seems to contain more features for automatic space reclaim...
- **Need TRIM support for Spectrum Scale RAID ... coming soon ☺**

NetApp EF600

Scale „SAN Mode“ with NVMe over Fabrics

NetApp EF600 (2U24 NVMe)

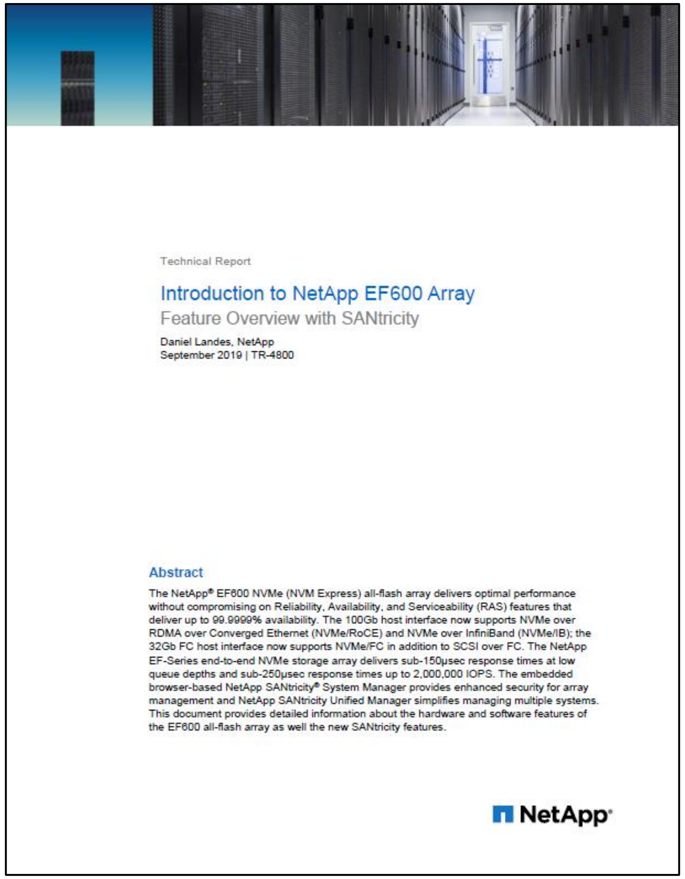


- **24x** Dual-Ported **NVMe** in 2U (1.92 to 15TB)
- **NVMe-o-F over**
  - FC32 (16 ports: two 4-port HICs per controller)
  - **EDR Infiniband** (8 ports: two 2-port HICs per controller)
  - 100GbE RoCE (8 ports: two 2-port HICs per controller)

- Peak Read Bandwidth: **44 GB/s**
- Peak Write Bandwidth:
  - 12.5 GB/s (CME)
  - **24 GB/s (FSWA)**

Have seen **48** GiB/s (sequential IOR read)

- See NetApp TR-4800 E-Series EF600 datasheet: https://www.netapp.com/us/media/ds-4002.pdf



Technical Report

Introduction to NetApp EF600 Array

Feature Overview with SANtricity

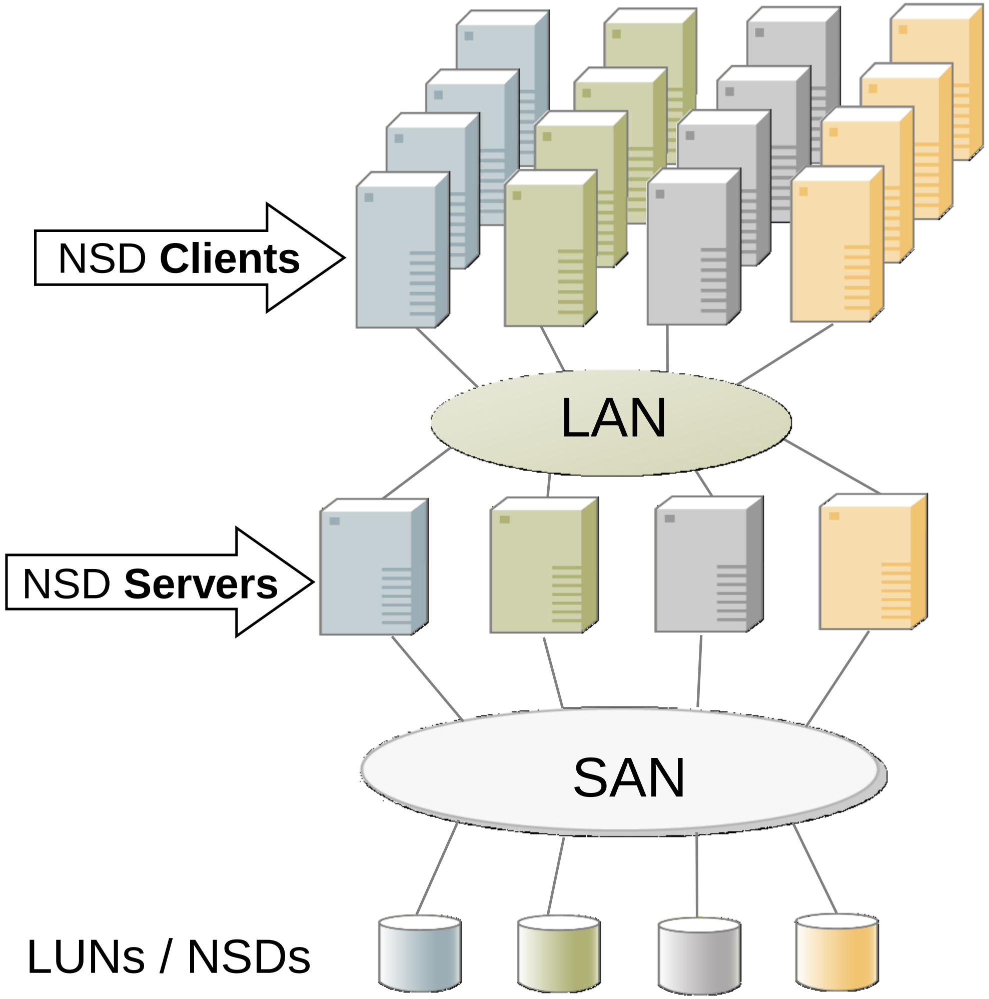
Daniel Landes, NetApp
September 2019 | TR-4800

Abstract

The NetApp® EF600 NVMe (NVM Express) all-flash array delivers optimal performance without compromising on Reliability, Availability, and Serviceability (RAS) features that deliver up to 99.9999% availability. The 100Gb host interface now supports NVMe over RDMA over Converged Ethernet (NVMe/RoCE) and NVMe over InfiniBand (NVMe/IB); the 32Gb FC host interface now supports NVMe/FC in addition to SCSI over FC. The NetApp EF-Series end-to-end NVMe storage array delivers sub-150µsec response times at low queue depths and sub-250µsec response times up to 2,000,000 IOPS. The embedded browser-based NetApp SANtricity® System Manager provides enhanced security for array management and NetApp SANtricity Unified Manager simplifies managing multiple systems. This document provides detailed information about the hardware and software features of the EF600 all-flash array as well the new SANtricity features.

NetApp®

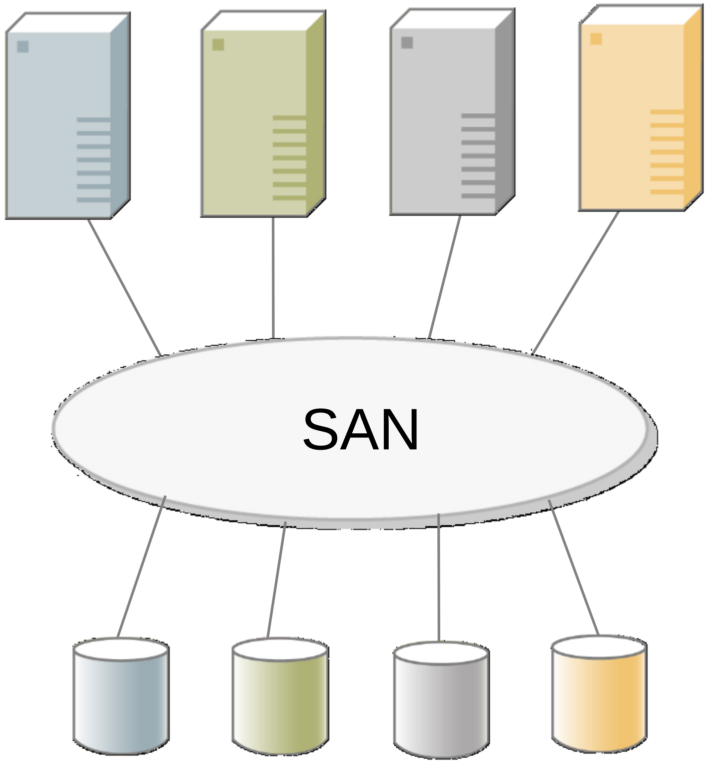
Spectrum Scale Architectures – NSD Client / Server Model



(NSD = Network Shared Disk)

Spectrum Scale Architectures – SAN Model

All NSDs visible on all application nodes through the SAN

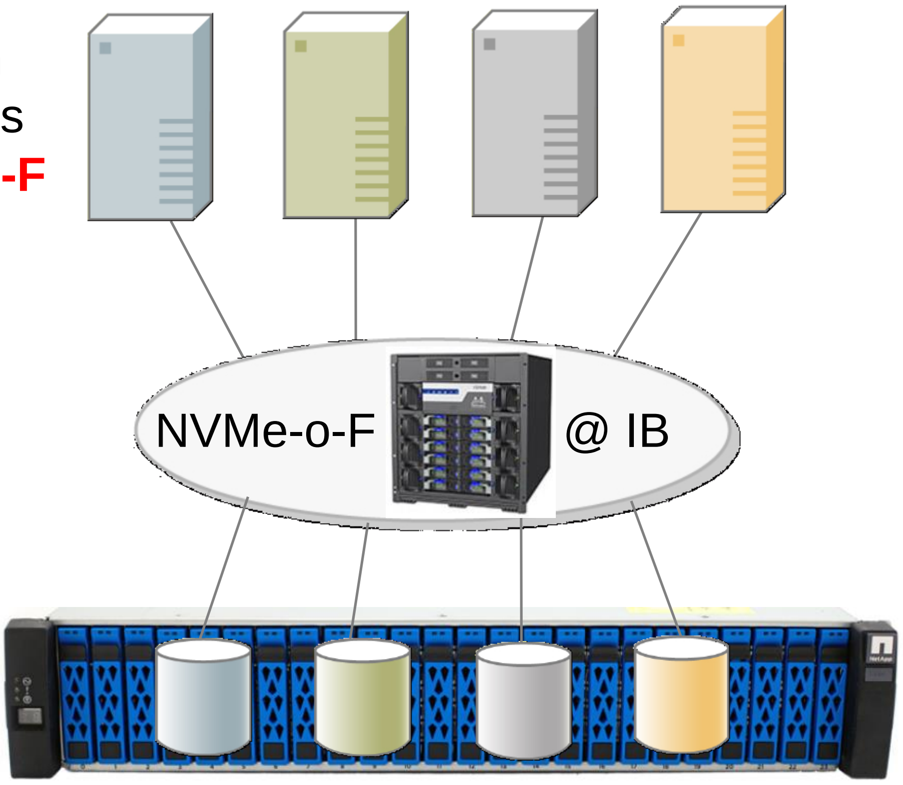


LUNs

(NSD = Network Shared Disk)

Spectrum Scale Architectures – SAN Model **with NVMe-o-F**

All NSDs visible on
all application nodes
through **NVMe-o-F**

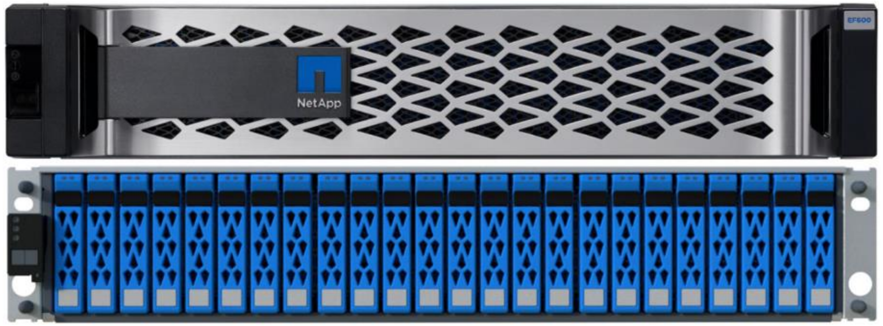


NVMe-o-F @ IB

LUNs

(NSD = Network Shared Disk)

NetApp EF600 (2U24 NVMe)



- Minimum SANtricity **11.60** software on EF600
- Host Software requirements
  - Latest RHEL 7.7 patches (or latest SLES12)
  - **MOFED** built with **NVMe-o-F support** (./mlnxofedinstall **--add-kernel-support --with-nvmf**)
  - **`modprobe nvme-rdma`** → this will create the **`/etc/nvme/hostnqn`** file (see next slide)
- 8x IB Host ports (4x on the A controller, and 4x B on the controller)

```
root@mgt2302:~# grep de0704 /etc/hosts
172.30.25.107     de0704a        de0704a.hpc.eu.lenovo.com
172.30.25.108     de0704b        de0704b.hpc.eu.lenovo.com
172.30.57.107     de0704a-ib0    de0704a-ib0.hpc.eu.lenovo.com
172.30.57.108     de0704a-ib1    de0704a-ib1.hpc.eu.lenovo.com
172.30.57.109     de0704a-ib2    de0704a-ib2.hpc.eu.lenovo.com
172.30.57.110     de0704a-ib3    de0704a-ib3.hpc.eu.lenovo.com
172.30.57.111     de0704b-ib0    de0704b-ib0.hpc.eu.lenovo.com
172.30.57.112     de0704b-ib1    de0704b-ib1.hpc.eu.lenovo.com
172.30.57.113     de0704b-ib2    de0704b-ib2.hpc.eu.lenovo.com
172.30.57.114     de0704b-ib3    de0704b-ib3.hpc.eu.lenovo.com
```

SANtricity **11.70**
now supports **TRIM**...

Creating the host initiators on the EF600

```
// SMcli commands to create hostGroup, host, and initiators:

create hostGroup
  userLabel="de0704_hg1";

create host
  userLabel="cmp2501" hostType=28 hostGroup="de0704_hg1";

create initiator
  identifier="nqn.2014-08.org.nvmexpress:uuid:783f3338-eda8-46d9-bb27-dd14cdcb4a1b"
  userLabel="cmp2501-ib0"
  host="cmp2501"
  interfaceType=nvmeof;
```

This **host NQN** UUID is stored on the nodes, in file **`/etc/nvme/hostnqn`**.
It is **not persistent** acrosss reboots !

Discover the EF600 IB Host Ports (repeat for all 8 ports)

```
[root@cmp2645 ~]# nvme discover -t rdma -a 172.30.57.107 # must be an IP address, not an IP name !

Discovery Log Number of Records 8, Generation counter 0
=====Discovery Log Entry 0======
trtype:  rdma
adrfam:  ipv4
subtype: nvme subsystem
treq:    not specified
portid:  0
trsvcid: 4420
subnqn:  nqn.1992-08.com.netapp:6000.6d039ea0003ef5100000000059729fff
traddr:  172.30.57.107
rdma_prtype: infiniband
rdma_qptype: connected
rdma_cms:    rdma-cm
rdma_pkey: 0x0000
```

This is the NQN UUID of the **EF600 storage subsystem.** Should see the same over all 8 ports...

Connect to the EF600 IB Host Ports (for all active EF600 ports)

```
ef600_sub_nqn_1=`nvme discover -t rdma -a 172.30.57.107
  | grep subnqn | sort -u | cut -d: -f2- | tr -d ' '`

echo "NQN=$ef600_sub_nqn_1"
NQN=nqn.1992-08.com.netapp:6000.6d039ea0003ef51000000000059729fff

queue_depth_setting=1024                # default is 128
controller_loss_timeout_period=3600     # default is 600

nvme connect -t rdma
 -n $ef600_sub_nqn_1 -a 172.30.57.107 \
 -Q $queue_depth_setting -l $controller_loss_timeout_period
```

- Both „nvme discover“ and „nvme connect“ are **not persistent** across reboots!

Listing the EF600 Volumes / Paths with nvme

- One NVMe **device #** per visible EF600 **host port** (A1,A2, ...B4)
- One NVMe **namespace ID** per mapped EF600 **volume (LUN)**

Example uses **four** EF600 host ports...

```
[root@cli0801 ~]# nvme netapp smdevices
/dev/nvme4n1, Array Name de0704-ef600, Volume Name vd0, NSID 1,
Volume ID 000009 7859b331b2d039ea00003ef510, Controller A, Access State unknown, 19.15TB # A1
/dev/nvme4n2, Array Name de0704-ef600, Volume Name vd1, NSID 2,
Volume ID 000009 4e59b33c26d039ea00003ef1fd, Controller A, Access State unknown, 19.15TB # A3
/dev/nvme5n1, Array Name de0704-ef600, Volume Name vd0, NSID 1,
Volume ID 000009 7859b331b2d039ea00003ef510, Controller A, Access State unknown, 19.15TB # A1
/dev/nvme5n2, Array Name de0704-ef600, Volume Name vd1, NSID 2,
Volume ID 000009 4e59b33c26d039ea00003ef1fd, Controller A, Access State unknown, 19.15TB # A3
/dev/nvme6n1, Array Name de0704-ef600, Volume Name vd0, NSID 1,
Volume ID 000009 7859b331b2d039ea00003ef510, Controller B, Access State unknown, 19.15TB # B1
/dev/nvme6n2, Array Name de0704-ef600, Volume Name vd1, NSID 2,
Volume ID 000009 4e59b33c26d039ea00003ef1fd, Controller B, Access State unknown, 19.15TB # B3
/dev/nvme7n1, Array Name de0704-ef600, Volume Name vd0, NSID 1,
Volume ID 000009 7859b331b2d039ea00003ef510, Controller B, Access State unknown, 19.15TB # B1
/dev/nvme7n2, Array Name de0704-ef600, Volume Name vd1, NSID 2,
Volume ID 000009 4e59b33c26d039ea00003ef1fd, Controller B, Access State unknown, 19.15TB # B3
```

DM-Multipathing for the EF600

Example uses **four** EF600 host ports...

```
# yum install -y device-mapper-multipath

# cat /etc/multipath.conf

# NetApp EF600 NVMe-o-F devices:
devices {
 device {
  vendor "NVME"
  product "NetApp E-Series*"
  path_grouping_policy group_by_prio
  failback immediate
  no_path_retry 30
 }
}
# exclude locally attached NVMe drives:
blacklist {
  wwid nvme.8086-*
}
```

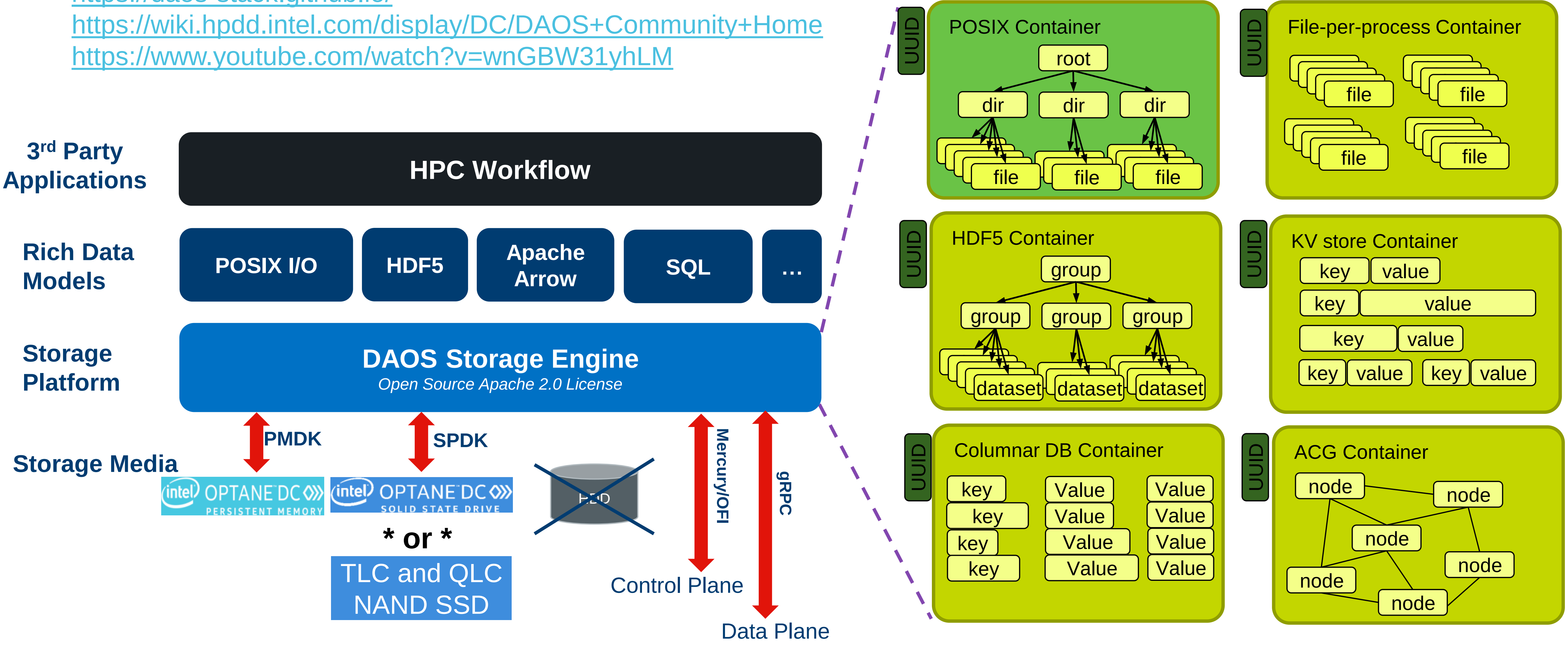
```
# multipath -ll
eui.0000097859b331b2d039ea00003ef510 dm-1 NVME,NetApp E-Series
size=17T features='1 queue_if_no_path' hwhandler='0' wp=rw
|-+- policy='service-time 0' prio=50 status=active
| |- 4:0:1:0 nvme4n1 259:4  active ready running
| `- 5:0:1:0 nvme5n1 259:6  active ready running
`-+- policy='service-time 0' prio=10 status=enabled
  |- 6:0:1:0 nvme6n1 259:8  active ready running
  `- 7:0:1:0 nvme7n1 259:10 active ready running
eui.0000094e59b33c26d039ea00003ef1fd dm-2 NVME,NetApp E-Series
size=17T features='1 queue_if_no_path' hwhandler='0' wp=rw
|-+- policy='service-time 0' prio=50 status=active
| |- 6:0:2:0 nvme6n2 259:9  active ready running
| `- 7:0:2:0 nvme7n2 259:11 active ready running
`-+- policy='service-time 0' prio=10 status=enabled
  |- 4:0:2:0 nvme4n2 259:5  active ready running
  `- 5:0:2:0 nvme5n2 259:7  active ready running
```

DAOS Unified Namespace

Distributed Asynchronous Object Storage

Intel Distributed Asynchronous Object Storage

https://daos-stack.github.io/
https://wiki.hpdd.intel.com/display/DC/DAOS+Community+Home
https://www.youtube.com/watch?v=wnGBW31yhLM



3rd Party Applications: HPC Workflow

Rich Data Models: POSIX I/O, HDF5, Apache Arrow, SQL, …

Storage Platform: **DAOS Storage Engine**
Open Source Apache 2.0 License

Storage Media

PMDK — intel OPTANE DC PERSISTENT MEMORY

SPDK — intel OPTANE DC SOLID STATE DRIVE

* or *

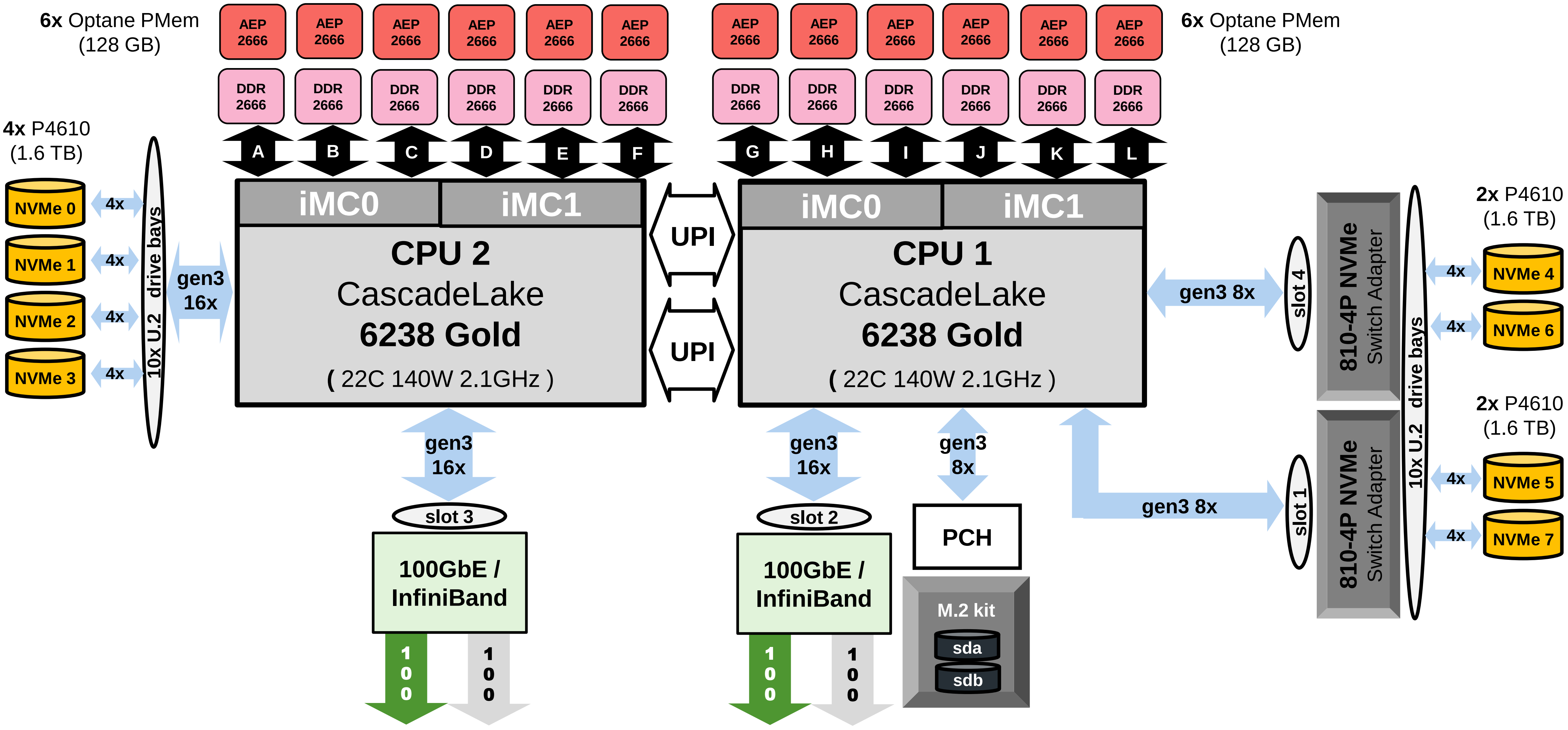
TLC and QLC NAND SSD

HDD

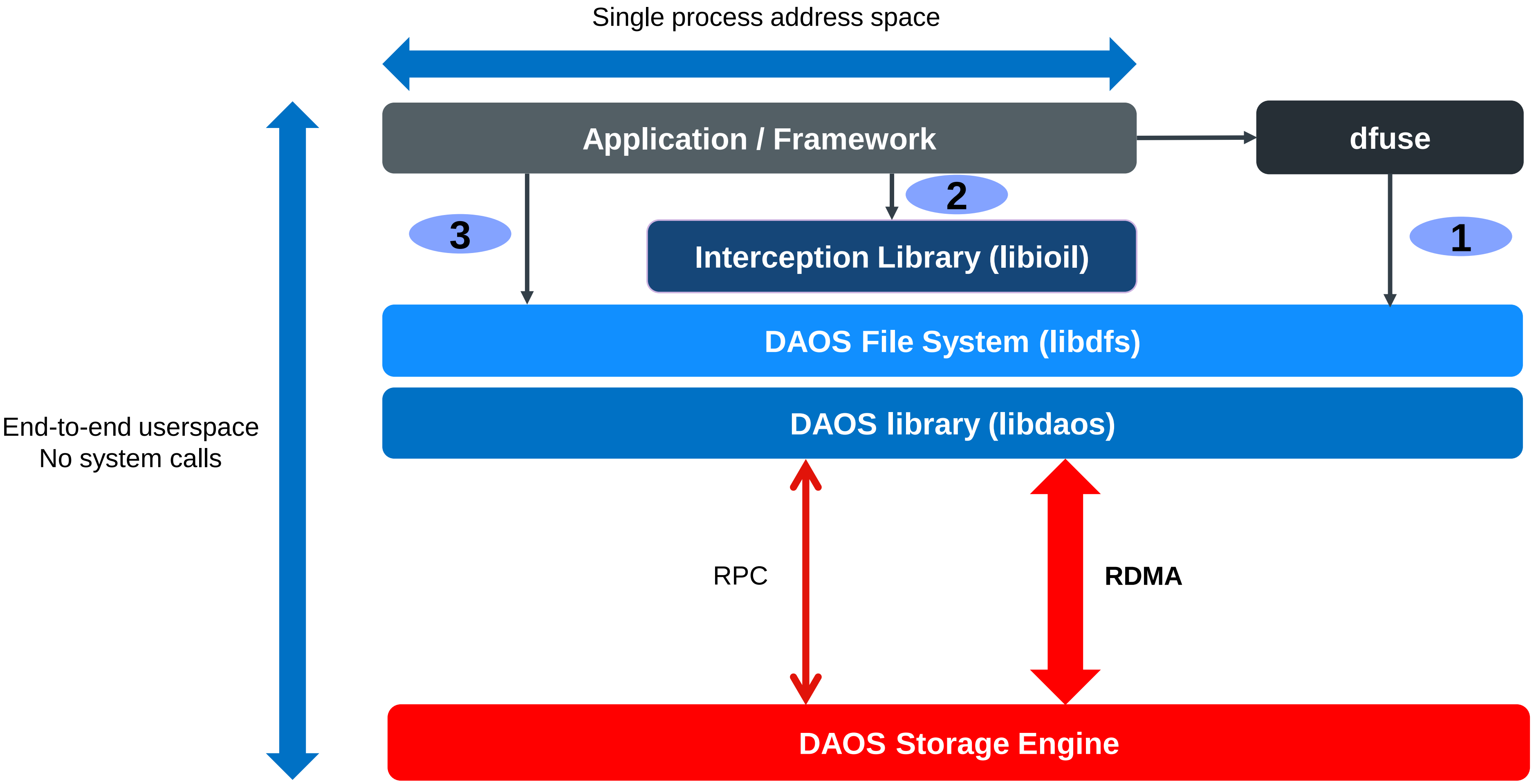
Mercury/OFI — Control Plane

gRPC — Data Plane

DAOS Server Architecture: Lenovo ThinkSystem SR630



Three Ways of POSIX Filesystem Support in DAOS



Single process address space

Application / Framework

dfuse

3

2

1

Interception Library (libioil)

DAOS File System (libdfs)

DAOS library (libdaos)

End-to-end userspace
No system calls

RPC

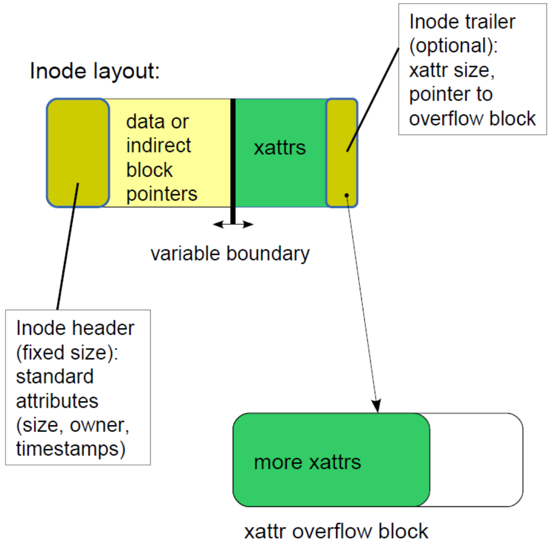
RDMA

DAOS Storage Engine

intel OPTANE DC PERSISTENT MEMORY

DAOS Unified Namespace with Spectrum Scale (1/2)

- DAOS „Unified Namespace“ Concept:
  1. Store DAOS ***pool UUID*** and ***container UUID*** as extended attributes (XATTR‘s) of the „mount point“ directory
  2. When this „mount point“ is in a global parallel filesystem, `dfuse` can use this instead of **--pool** and **--container**
- IBM Spectrum Scale supports Extended Attributes (XATTR‘s), both for internal features and for user metadata
  – Stored in inode if small, or in „overflow“ EA block (≤64 kiB)



Inode layout:

data or indirect block pointers

xattrs

Inode trailer (optional): xattr size, pointer to overflow block

variable boundary

Inode header (fixed size): standard attributes (size, owner, timestamps)

more xattrs

xattr overflow block

```
$ daos cont create --pool=$D_POOL --svc=$D_SVC --cont=$D_CONT \
    --type=POSIX --path /home/mhennecke/daos_tmp
```

```
$ mmlsattr --dump-attr /home/mhennecke/daos_tmp
file name: /home/mhennecke/daos_tmp
user.daos
$ mmlsattr --get-attr user.daos /home/mhennecke/daos_tmp
file name: /home/mhennecke/daos_tmp
user.daos: "DAOS.POSIX://c0c99a8c-5453-4950-9bbd-1d9d784b51c0/7b6ff2f2-b52d-4a25-8565-285006572c96?“
```

DAOS Unified Namespace with Spectrum Scale (2/2)

- **daos** can query the Spectrum Scale mountpoint directory‘s XATTR‘s on each node where the „containing“ Spectrum Scale filesystem is mounted:

```
$ daos cont query --path /home/mhennecke/daos_tmp --svc 0
Pool UUID:      c0c99a8c-5453-4950-9bbd-1d9d784b51c0
Container UUID: 7b6ff2f2-b52d-4a25-8565-285006572c96
Number of snapshots: 0
Latest Persistent Snapshot: 0
Highest Aggregated Epoch: 1605783539794720768
DAOS Unified Namespace Attributes on path /home/mhennecke/daos_tmp:
Container Type: POSIX
Object Class:   SX
Chunk Size:     1048576
```

- The **dfuse** mount command can use the path without `--pool` and `--cont`:

```
$ dfuse -m /home/mhennecke/daos_tmp --svc 0
$ df|grep daos
dfuse   13980468750   727   13980468024   1% /gpfs/gss1/home/mhennecke/daos_tmp
```

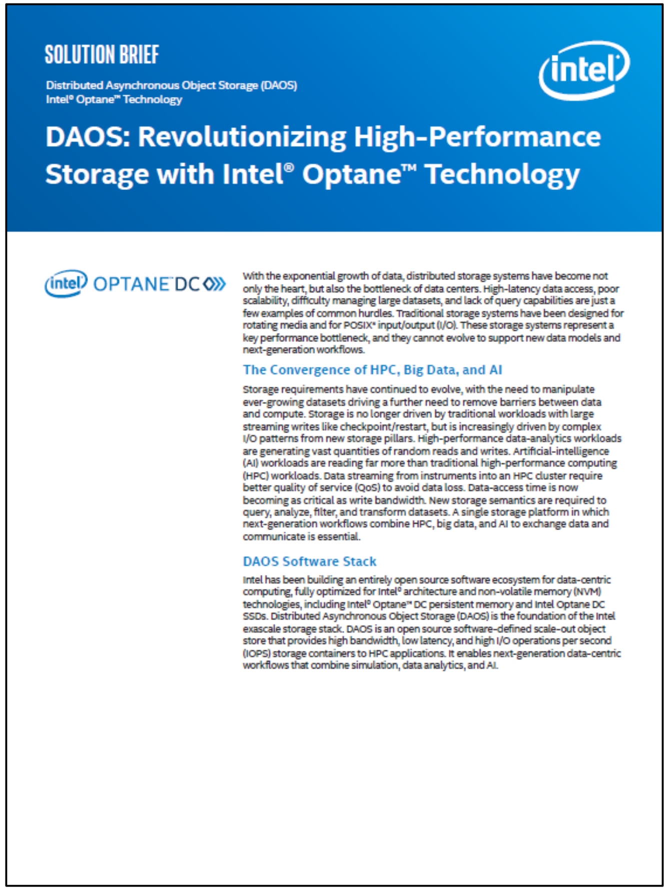
DAOS IO500 – 1 Server (P4610 3.2TB), 10 Clients

IO500 with API=DFS, and Intel‘s DFS-enabled `find` from `mpifileutils`:

```
IO500 version io500-sc20_v3
[RESULT]       ior-easy-write       20.754597 GiB/s : time 315.288 seconds
[RESULT]    mdtest-easy-write      586.050492 kIOPS : time 308.214 seconds
[RESULT]       ior-hard-write        8.015282 GiB/s : time 316.094 seconds
[RESULT]    mdtest-hard-write      120.679218 kIOPS : time 320.813 seconds
[RESULT]                 find      328.553089 kIOPS : time 657.437 seconds
[RESULT]        ior-easy-read       21.865060 GiB/s : time 298.510 seconds
[RESULT]     mdtest-easy-stat      919.974294 kIOPS : time 192.983 seconds
[RESULT]        ior-hard-read        9.767739 GiB/s : time 258.846 seconds
[RESULT]     mdtest-hard-stat      532.842517 kIOPS : time  73.066 seconds
[RESULT]   mdtest-easy-delete      389.111423 kIOPS : time 467.045 seconds
[RESULT]     mdtest-hard-read      186.604589 kIOPS : time 207.250 seconds
[RESULT]   mdtest-hard-delete      370.730738 kIOPS : time 192.598 seconds
[SCORE] Bandwidth 13.729175 GiB/s : IOPS 363.768691 kiops : TOTAL 70.669966
```

ior-hard used `--dfs.chunk_size=470080` (10x the transferSize)

For More Information on Lenovo‘s DAOS Solutions...



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.

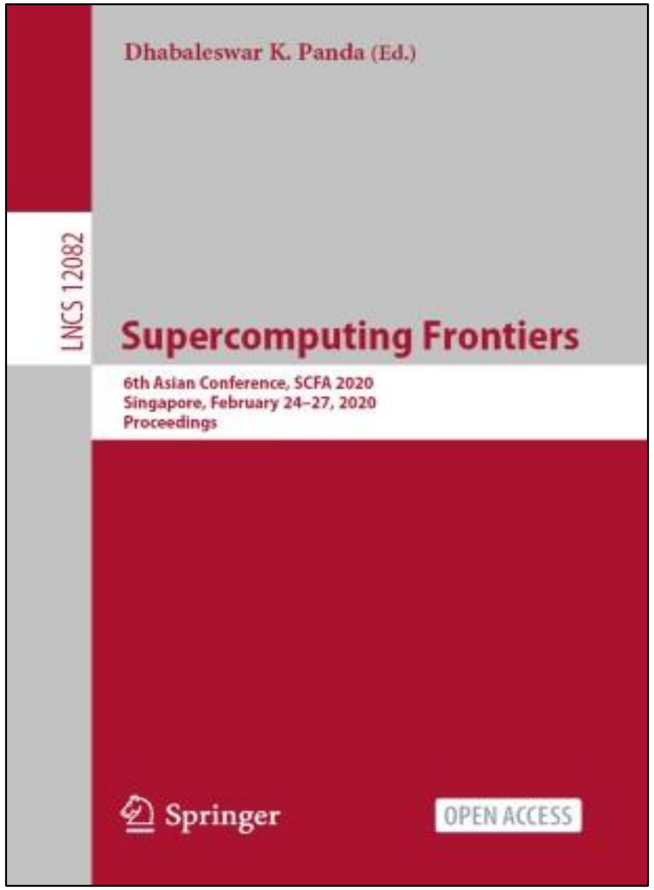
The Convergence of HPC, Big Data, and AI

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.

Intel‘s HPC **Solution Brief:**
DAOS with Optane Technology



Dhabaleswar K. Panda (Ed.)

LNCS 12082

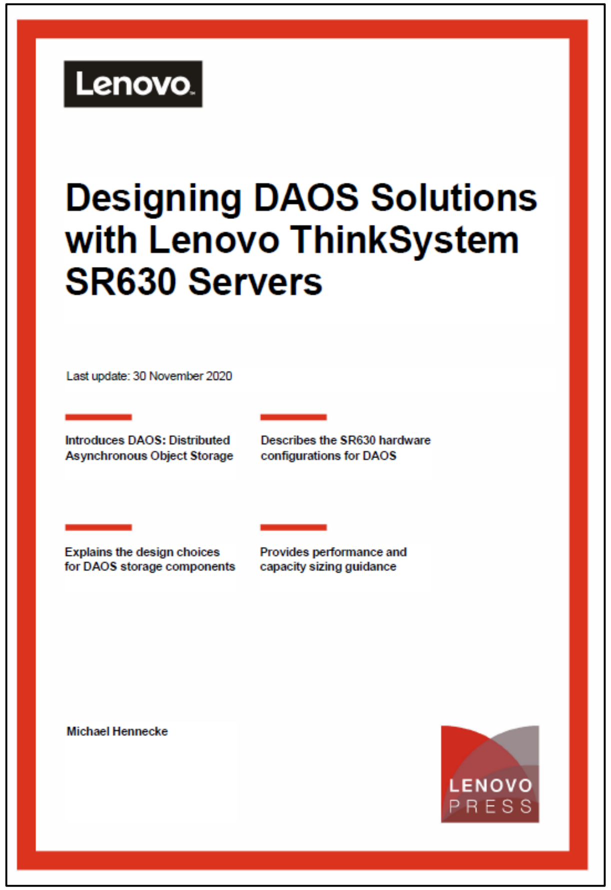
Supercomputing Frontiers

6th Asian Conference, SCFA 2020
Singapore, February 24–27, 2020
Proceedings

Springer

OPEN ACCESS

Intel / Lenovo **DAOS Article**
(SC-Asia, Springer LNCS 12082)



Lenovo

Designing DAOS Solutions with Lenovo ThinkSystem SR630 Servers

Last update: 30 November 2020

Introduces DAOS: Distributed Asynchronous Object Storage

Describes the SR630 hardware configurations for DAOS

Explains the design choices for DAOS storage components

Provides performance and capacity sizing guidance

Michael Hennecke

LENOVO PRESS

DAOS on Lenovo SR630
(LenovoPress LP1398)

thanks.

mhennecke @ lenovo.com