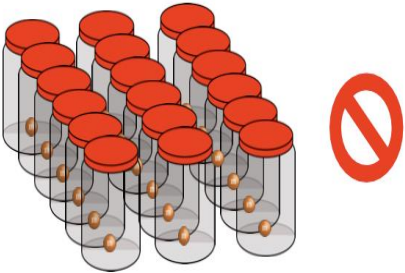
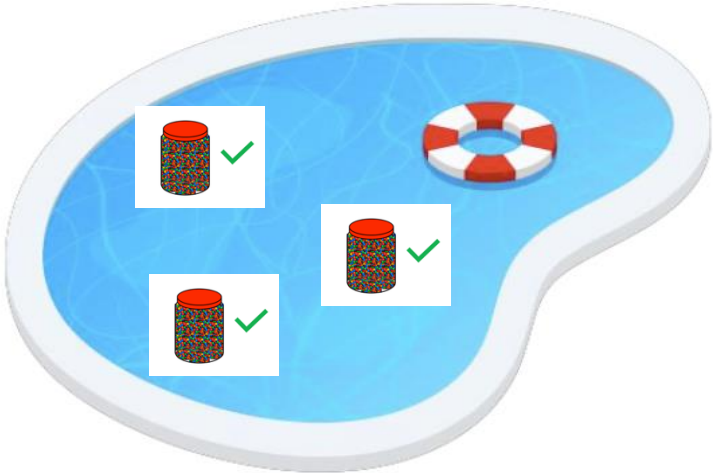


DAOS - Advanced

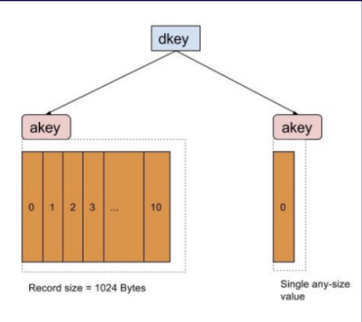
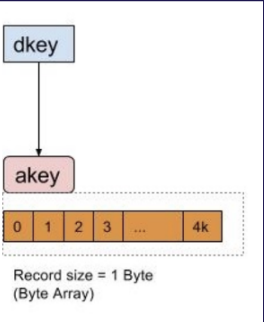
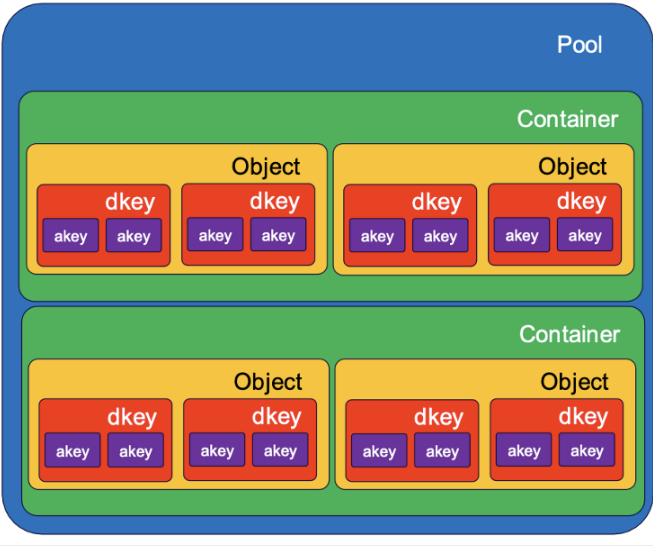
Kaushik Velusamy



Object



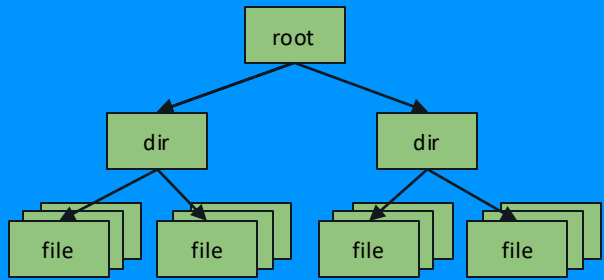
DAOS Data Model



Object Interface

Middleware/Framework View

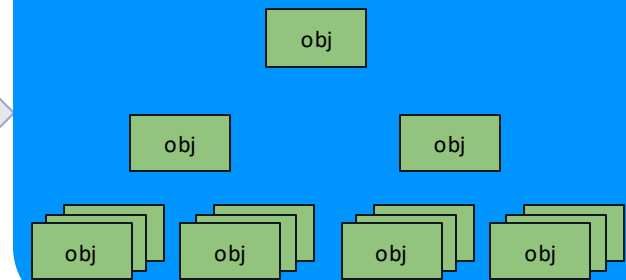
e.g. POSIX Dataset



Mapping

DAOS Layout View

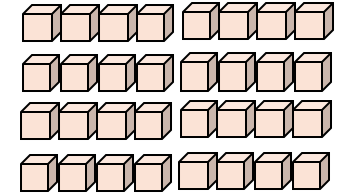
DAOS Container



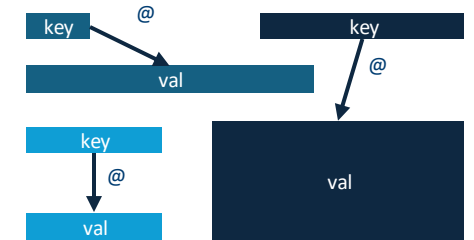
Object

128-bit
object Identifier

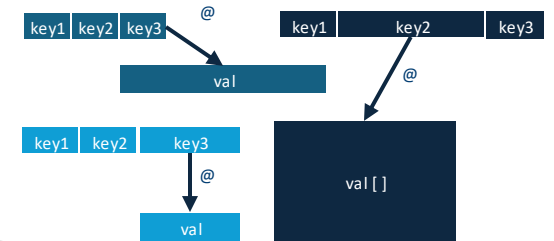
Array



Key-value Store



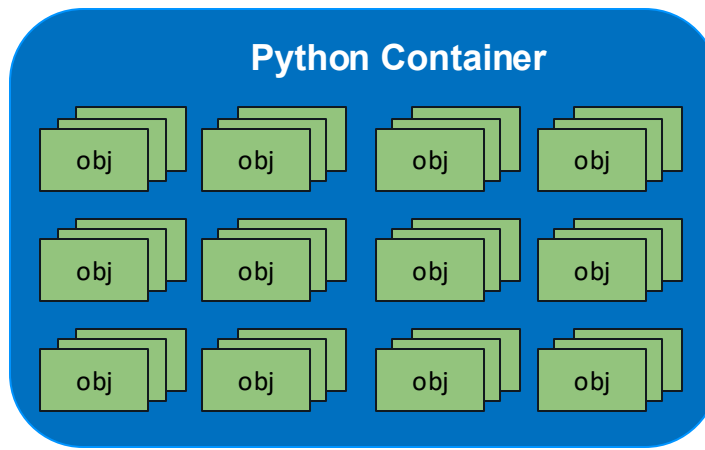
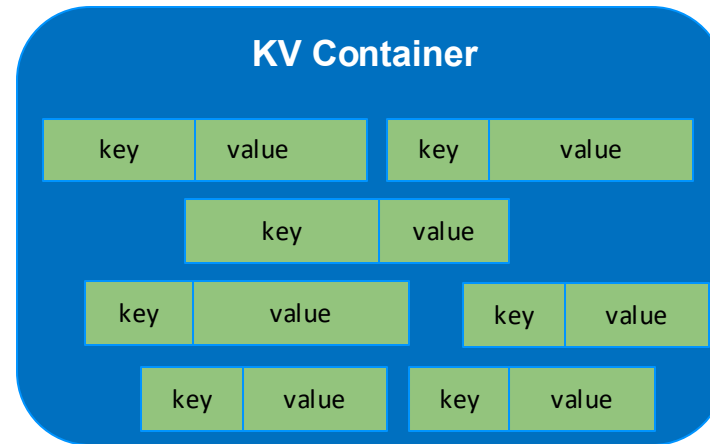
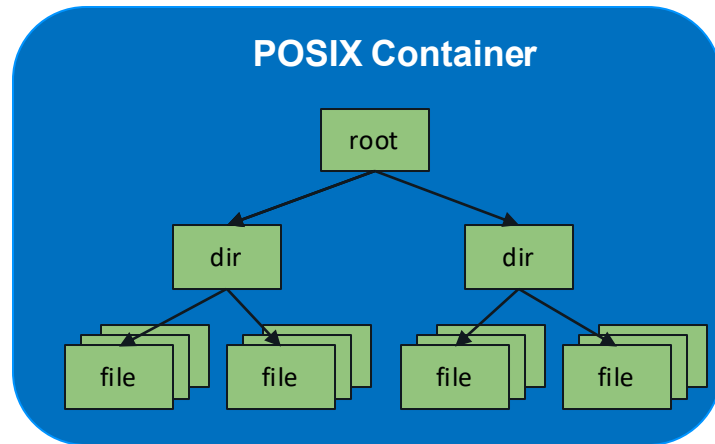
Multi-level
Key-value Store



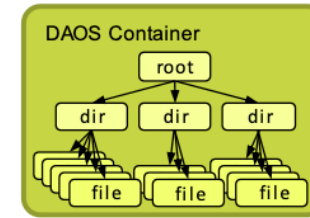
- No object create/destroy
- No size, permission/ACLs or attributes
- Sharded and erasure-coded/replicated
- Algorithmic object placement
- Very short Time To First Byte (TTFB)

Object

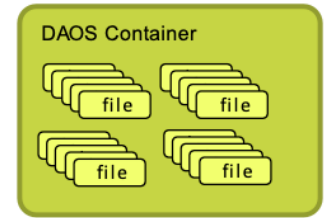
- Array or key-value store
- $O(1T)$ objects in a container
- e.g. files and directories in a POSIX container



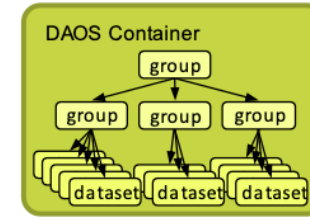
Examples



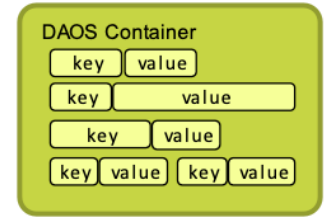
Encapsulated POSIX Namespace



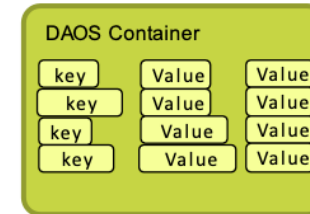
File-per-process



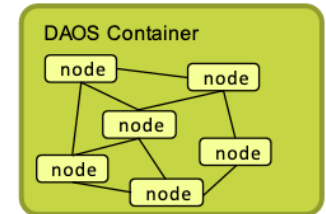
HDF5 « File »



Key-value store



Columnar Database



Graph

Retrieving information about your POSIX container (single process)

```
$ daos fs scan HPE_test new_cont
DFS scanner: Start (2025-05-10-13:59:22)
DFS scanner: Scanned 7236 files/directories (runtime: 30 sec)
DFS scanner: Scanned 14315 files/directories (runtime: 60 sec)
DFS scanner: Done! (runtime: 87 sec)
DFS scanner: 21461 scanned objects
DFS scanner: 19201 files
DFS scanner: 101 symlinks
DFS scanner: 2159 directories
DFS scanner: 15 max tree depth
DFS scanner: 809438484 bytes of total data
DFS scanner: 42156 bytes per file on average
DFS scanner: 170912509 bytes is largest file size
DFS scanner: 665 entries in the largest director
```



Where is my object?

Retrieving information about an object (advanced)

```
$ daos fs get-attr -H /daos/src $DAOS_POOL $DAOS_CONT
```

```
OID = 2533280060991858.0
```

```
Object Class = RP_3G1
```

```
Directory Creation Object Class = RP_3G1
```

```
File Creation Object Class = EC_16P2GX
```

```
File Creation Chunk Size = 4194304
```

```
$ daos obj query -i 2533280060991858.0 $DAOS_POOL $DAOS_CONT
```

```
oid: 2533280060991858.0 ver 0 grp_nr: 1
```

```
grp: 0
```

```
replica 0 204:10
```

```
replica 1 209:9
```

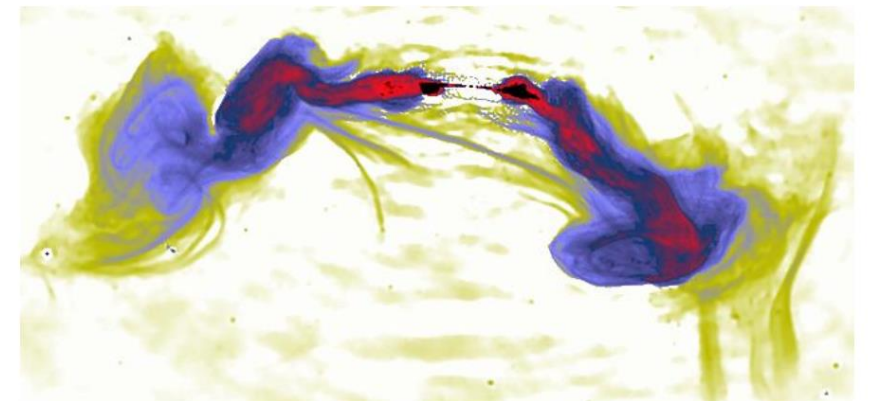
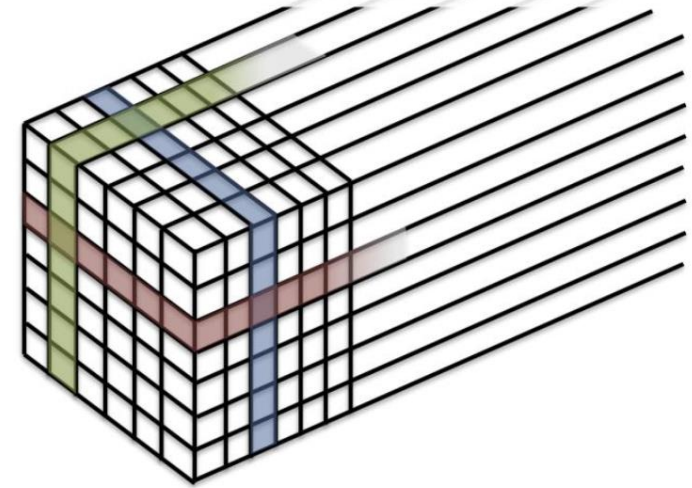
```
replica 2 56:5
```



Object Stores

- Object Stores can unlock previously expensive I/O patterns
- Supports different creation, querying, analysis, and use patterns
- Data retrieval leverages metadata - Build structure on the fly
- Weather/climate - Simulation (data generation) only one part Consumption workloads different layout/pattern from production.
- Radio astronomy- Data collected and stored by antenna (frequency and location) and capture time. Reconstruction of images done in time order.
- Evaluation of transients or other phenomenon undertaken across frequency and location.

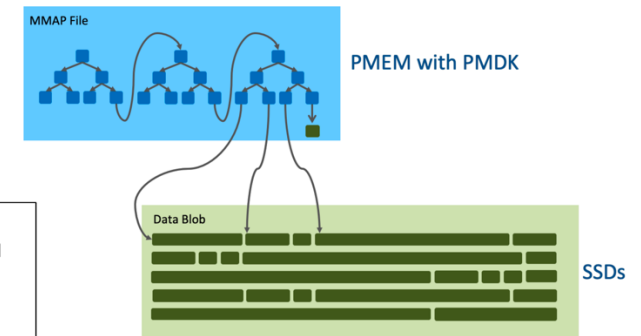
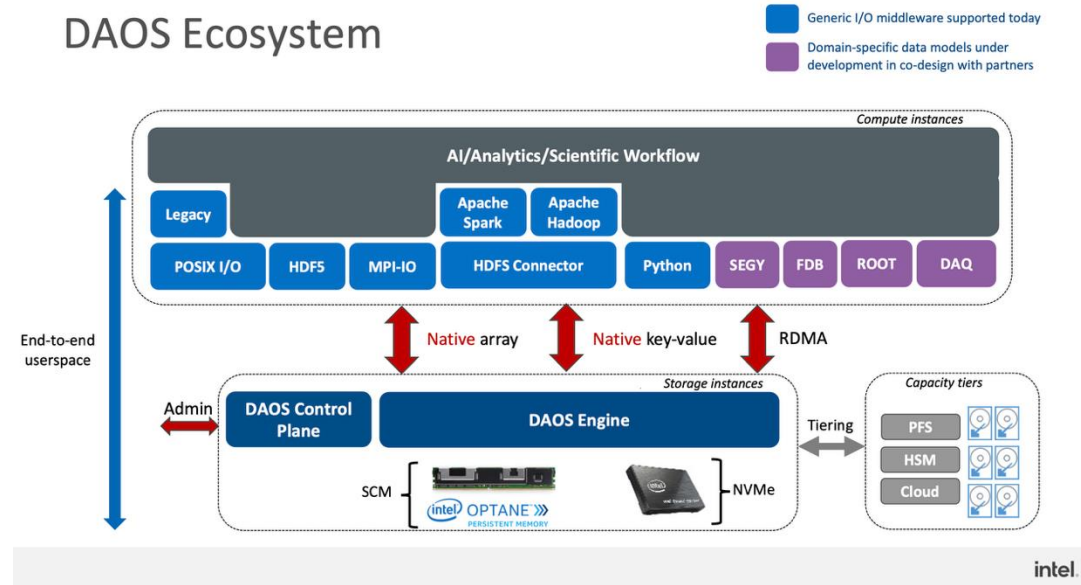
Clients want to do **different** analytics across **multiple** axis



DAOS Ecosystem and Design Fundamentals

- Efficient for unstructured data
- Efficient for accessing small data.
- High bandwidth, low latency, and IOPS
- No read-modify-write on I/O path (use versioning)
- No locking
- No client tracking or client recovery
- No centralized (meta)data server
- No global object table

DAOS Ecosystem



- Persistent metadata
- Require Intel Optane PMEM (or NVDIMM-N)
- App Direct mode
- Mode used on Aurora

Storage efficiency and greater fault tolerance against multiple failures

Replication duplicates entire data copies for simplicity and fast reads but is storage-inefficient

Erasure coding splits data into fragments with parity for high storage efficiency and greater fault tolerance against multiple failures, though it introduces performance overhead and complexity.

Erasure coding is better for large, infrequently accessed data.

Replication suits latency-sensitive, small-file workloads.



Replication

- **How it works:**
- Stores multiple, identical full copies of the same data across different locations.
- **Benefits:**
- **Simplicity:** Easy to implement and manage.
- **Fast Reads:** Direct access to full copies allows for rapid data retrieval.
- **Lower CPU Load:** Less computationally intensive than erasure coding.
- **Disadvantages:**
- **High Storage Overhead:** Requires substantially more storage space (e.g., triple for three copies).
- **Slower Writes:** Write performance can be impacted by the need to update all copies simultaneously.
- **Best For:**
- Latency-sensitive applications, small files, and workloads where read performance is critical and storage cost is less of a constraint.

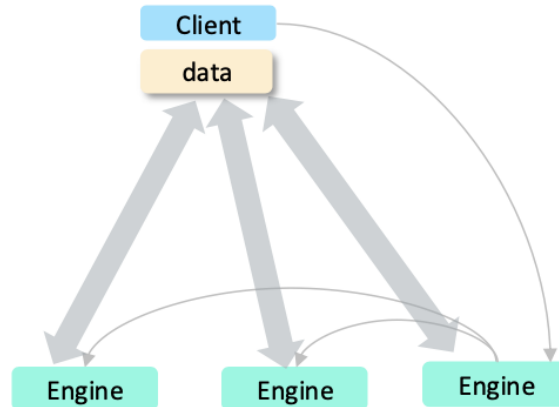
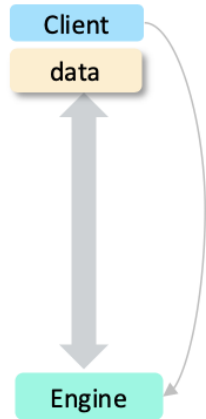
Erasure Coding

- **How it works:**
- Breaks data into smaller pieces and adds parity information, storing these fragments across multiple nodes or locations.
- **Benefits:**
- **Storage Efficiency:** Significantly less storage space is required compared to replication for the same level of protection.
- **Higher Fault Tolerance:** Can tolerate more simultaneous component failures than replication (e.g., losing multiple drives or nodes).
- **Greater Flexibility:** Offers more flexible and dynamic data protection configurations.
- **Disadvantages:**
- **Performance Overhead:** Encoding and decoding data introduce computational overhead, which can slow down performance.
- **Complexity:** More complex to implement and manage due to the algorithms involved.
- **Less Ideal for Small Files:** Not suitable for very small objects (under 200 KB) due to fragmentation overhead.
- **Best For:**
- Large-scale, cold storage, long-term archival, and scenarios where high durability, scalability, and storage efficiency are prioritized over read speed.



Erasure Coding and Replication

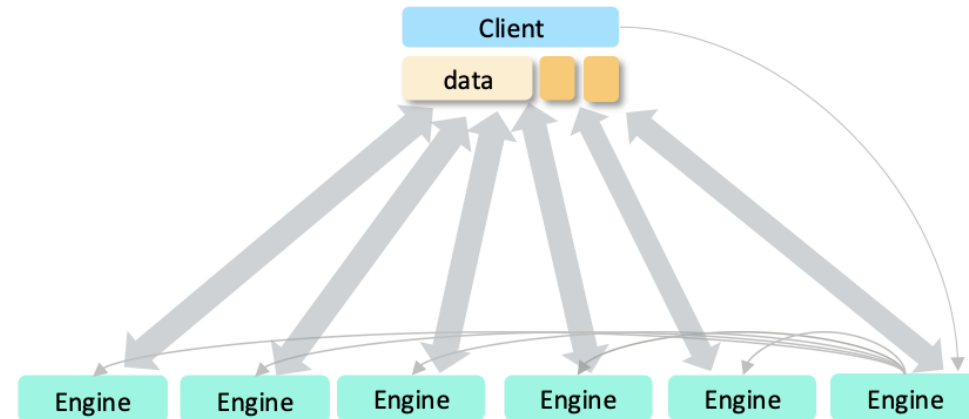
No Data Protection
Full bandwidth/IOPS on read
both read & write



3-way Replication
Full bandwidth/IOPS on read
66% loss on write

16+2 Erasure Code
Full bandwidth/IOPS on read
12% loss on full stripe write
66% loss on partial write

4+2 Erasure Code
Full bandwidth/IOPS on read
33% loss on full stripe write
66% loss on partial write
(replicated turned into EC in background)



Redundancy

What is a fault domain?

- Unit of failure: ssd, engine, server, rack, etc.
- On aurora the FD is set to the daos server node (2 engines, 32 targets/SSDs).
- Properties are set on container creation (most properties are immutable):

Container Properties (Redundancy)

Object class controls redundancy and striping/sharding of the object (file, directory).

daos cont create --properties=rd_fac:2

- Redundancy factor property describes the number of concurrent fault domains (servers) that containers are protected against.
- Redundancy factor of 0 is mostly used to measure system performance, but not for production workloads
- Data in container is non redundant; if a DAOS engines goes down, data loss / corruption will be observed by the user



DAOS Fault Handling

- Automatic online self-healing
- Storage nodes monitor each other (SWIM protocol)
- "Rebuild" is triggered to restore data redundancy/protection on surviving nodes when a node fails
- The rebuild process is done online and should complete quickly
- Failed storage node to be reintegrated by administrator once issue is fixed
- Checking rebuild status

```
$ daos pool query HPE_test
Pool 7d5b9907-1b31-4b65-b308-03947b0cbbc7, ntarget=4096, disabled=112, leader=21, version=634,
state=Degraded
Pool health info:
- Rebuild busy, 45 objs, 412318398 recs
Pool space info:
- Target(VOS) count:3984
- Storage tier 0 (SCM):
Total size: 146 GB
Free: 144 GB, min:36 MB, max:36 MB, mean:36 MB
- Storage tier 1 (NVMe):
Total size: 4.7 TB
Free: 4.6 TB, min:1.1 GB, max:1.1 GB, mean:1.1 GB
```



Objects can be created with different redundancy types and levels:

- No redundancy (non-production)
- Replication (good for metadata / small objects)
- Erasure Coding (good for files / large objects)



	RF0	RF1	RF2
File	SX	EC_16P1GX	EC_16P2GX
Dir	S1	RP_2G1	RP_3G1

	RF0	RF1	RF2
Large files, Single shared access, high BW required	SX	EC_16P1GX	EC_16P2GX
Tiny files, more IOPS required	S1	RP_2G1	RP_3G1
Something in between, and File per process to large files	S32	EC_16P1G32	EC_16P2G32



```
$ daos cont query HPE_test test_cont
```

```
Container UUID : e1870f74-609f-4ea0-9852-ef4a3de7b5af
```

```
Container Label : test_cont
```

```
Container Type : POSIX
```

```
Pool UUID : 7d5b9907-1b31-4b65-b308-03947b0cbbc7
```

```
Container redundancy factor : 2
```

```
Number of open handles : 1
```

```
Latest open time Latest close/modify time : 0x1e7ed8192c200000  
(2025-05-09 13:50:54.426841088 +0000 UTC)
```

```
: 0x1e7ed8193a240003 (2025-05-09 13:50:54.441537536 +0000 UTC)
```

```
Number of snapshots : 0
```

```
Object Class : UNKNOWN
```

```
Dir Object Class : RP_3G1
```

```
File Object Class : EC_16P2GX
```

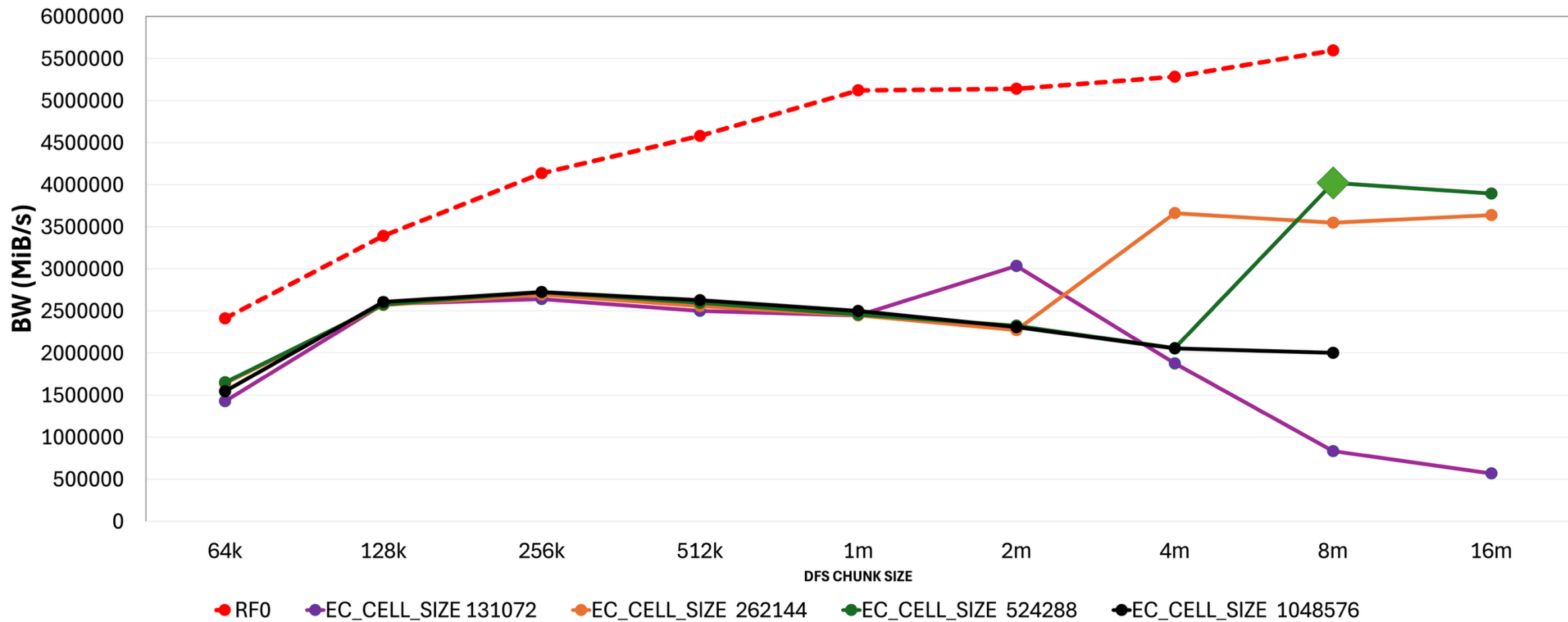
```
Chunk Size : 4.0 MiB
```




```
daos container create
  -type=POSIX ${DAOS_POOL_NAME}  ${DAOS_CONT_NAME}
  --chunk-size=2097152
  --file-oclass=EC_16P2G32
  --dir_oclass=RP3G1
  --properties=rd_fac:2,ec_cell_sz:131072,cksum:crc32,srv_cksum:on
```



HACC - GENERIC I/O WRITE WITH 240 BILLION PARTICLES
256 COMPUTE NODES 12 PPN AND 230 DAOS SERVERS - EC 16 P2 GX



```
daos fs set-attr --path=/mnt/dfuse/d1 --oclass=EC_16P1G32
```

–In this case, d1 must exist in the daos container mount at the dfuse mountpoint, and anything created under d1 will now have object class of EC_16P1G32.



MPI-IO Container Access

[↑ Back to top](#)

The MPICH MPI-IO layer on Aurora (ROMIO) provides multiple I/O backends including one for DAOS. ROMIO can be used with dFuse and the interception library utilizing the UFS backend, but the DAOS backend will provide optimal performance. By default ROMIO will auto-detect DFS and use the DAOS backend. MPI-IO itself is a common backend for many I/O libraries, including HDF5 and PNetCDF. Whether using collective I/O MPI-IO calls directly or indirectly via an I/O library, a process called collective buffering can be done where data from small non-contiguous chunks across many compute nodes in the collective is aggregated into larger contiguous buffers on a few compute nodes, referred to as aggregators, from which DFS API calls are made to write to or read from DAOS. Collective buffering can improve or degrade I/O performance depending on the I/O pattern, and in the case of DAOS, disabling it can lead to I/O failures in some cases, where the I/O traffic directly from all the compute nodes in the collective to DAOS is too stressful in the form of extreme numbers of small non-contiguous data reads and writes. In ROMIO there are hints that should be set to either optimally enable or disable collective buffering. At this time you should explicitly enable collective buffering in the most optimal fashion, as disabling it or allowing it to default to disabled could result in I/O failures. To optimally enable collective buffering, create a file with the following contents:

```
1 romio_cb_write enable
2 romio_cb_read enable
3 cb_buffer_size 16777216
4 cb_config_list *:8
5 striping_unit 2097152
```

Then simply set the following environment variable at run time to point to it:

```
1 export ROMIO_HINTS=<path to hints file>
```

If you want to verify the settings, additionally set:

```
1 export ROMIO_PRINT_HINTS=1
```

Which will print out all the ROMIO hints at run time.

MPI-I/O

- Part of MPI Standard
- Collective IO optimizations:
 - Aggregation of small IO across all your processes
 - Collective File Create/Open (if file system supports it)
- Supports multitude of access patterns
- Derived MPI datatype and File Views
- Building block for parallel IO libraries
- HDF5, PnetCDF , etc.
- Chances are if you are doing parallel IO on Aurora, your App has an MPIIO backend



- MPICH includes a DAOS ROMIO driver to avoid going through the kernel
- dfuse is still required to be mounted to access files through a path and query the pool and container information from that path.
- MPIIO uses a POSIX container, so the same advice applies in this case.
- Other libraries build on top of MPICH (HDF5, PnetCDF, ADIOS, etc.)
- Using those libraries also follow the same advise.

MPIIO recommendations:

- The DAOS adio driver is optimized for Single Shared File access (i.e. when the file is not opened with `MPI_COMM_SELF`).
 - Collective file open shares the underlying DAOS pool and container handles (expensive operations).
 - As long the file is open collectively, IO access whether independent or collective is OK.
 - Avoid File Per Process access with the driver. If not possible, set it to use the UFS driver instead: export `ROMIO_FSTYPE_FORCE=ufs` and use the Interception library.
- Enable ROMIO collective buffering when doing collective IO especially when using access patterns that are extremely non-contiguous in the file (thousands of small bytes offsets).



DFS Container Access

DFS is the user level API for DAOS. This API is very similar to POSIX but still has many differences that would require code changes to utilize DFS directly. The DFS API can provide the best overall performance for any scenario other than workloads which benefit from caching.

Reference code for using DAOS through DFS mode and DAOS APIs

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <mpi.h>
4  #include <daos.h>
5  #include <daos_fs.h>
6  int main(int argc, char **argv)
7  {
8      dfs_t *dfs;
9      d_iov_t global;
10     ret = MPI_Init(&argc, &argv);
11     ret = MPI_Comm_rank(MPI_COMM_WORLD, &rank);
12     ret = dfs_init();
13     ret = dfs_connect(getenv("DAOS_POOL"), NULL, getenv("DAOS_CONT"), O_RDWR, NULL, &dfs);
14     ret = dfs_open(dfs, NULL, filename, S_IFREG|S_IRUSR|S_IWUSR, O_CREAT|O_WRONLY, obj_class, chunk_size, NULL, &obj);
15     ret = dfs_write(dfs, obj, &sgl, off, NULL);
16     ret = dfs_read(dfs, obj, &sgl, off, &read, NULL);
17     ret = dfs_disconnect(dfs);
18     ret = daos_fini();
19     ret = MPI_Finalize();
20 }
```

The full code is available on the Aurora filesystem within `/soft/daos/examples/src/`

```

1  import torch as sys_torch
2  from pydaos.daos_torch import Dataset as DaosDataset
3  from pydaos.daos_torch import Checkpoint as DaosCheckpoint
4  from io import BytesIO
5  from mpi4py import MPI
6
7  comm = MPI.COMM_WORLD
8  pydaos_torch_ckpt = DaosCheckpoint("datascience", "my_container", "") #To connect to DFS container
9  a = sys_torch.ones(1048576)
10 data = dict()
11 data = { "a": a, }
12 name = f"/data-{comm.rank}-of-{comm.size}.pt"
13
14 # PyDAOS Torch dataloader example
15 def transform(data):
16     return np.load(BytesIO(data), allow_pickle=True)['x']
17 ds = DaosDataset(pool="datascience", cont="my-dataset", transform_fn=transform)
18
19 # PyDAOS Torch checkpoint save example
20 with pydaos_torch_ckpt.writer(name) as f:
21     sys_torch.save(data, f)
22 print(f"Torch save completed")
23
24 # PyDAOS Torch checkpoint load example
25 stream = pydaos_torch_ckpt.reader(name)
26 loaded_data = sys_torch.load(stream, weights_only=True)
27 print(f"Torch load completed")

```

- PyDAOS uses `dfs_write()` and read functions, which are faster than POSIX `dfuse_write()` and read functions.
- PyDAOS uses DFS containers and Python DAOS containers.
- The path to the dataset folders inside these containers does not include `/tmp` and just starts from `/dataset_dir1` which assumes a folder inside the `DAOS_POOL` and `DAOS_CONT`
- The above build path might be upgraded with newer builds without warning
- More examples can be found at [DAOS GitHub repo](#) > [pydaos.torch](#)

PyTorch - PyDAOS

- Support torch.save and torch.load function
- checkpoints directly to/from DAOS container
- could be same or different container than the one used for data loading
- pydaos.torch.Checkpoint class
- manages the DAOS connections
- provides reader and writer methods. Self-contained module
 - • pydaos.torch modules
 - • Link directly with libdfs (no need to dfuse or pil4dfs)
 - • Highly parallel and use multiple network contexts / even:

Support for both:

- Map-style dataset
 - torch.utils.data.Dataset
 - implement
 - __len__ / __getitem__ / __getitems__()
 - requires files scanning which is done in parallel thanks to dfuse
 - Iterable datasets
 - torch.utils.data.IterableDataset
 - iter__() over samples
- PyTorch multi-processing supported
- Provide worker_init() method
- Share pool/container handle across all the workers

```
import torch
from torch.utils.data import DataLoader
from pydaos.torch import Dataset

dataset = Dataset(pool='pool', container='container', path='/training/samples')
# That's it, when the Dataset is created, it will connect to DAOS, scan the namespace of the container
# and will be ready to load data from it.

for i, sample in enumerate(dataset):
    print(f"Sample {i} size: {len(sample)}")
```

```
dataloader = DataLoader(dataset,
                        batch_size=4,
                        shuffle=True,
                        num_workers=4,
                        worker_init_fn=dataset.worker_init)

# and use DataLoader as usual
for batch in dataloader:
    print(f"Batch size: {len(batch)}")
```





Monitoring DAOS

Kaushik Velusamy



Monitoring with DAOS interception library report

```
export D_IL_REPORT=1
```

```
mpiexec --env LD_PRELOAD=/usr/lib64/libpil4dfs.so  
-np ${NRANKS}..
```

```
11 Options:  
12 api : POSIX  
13 apiVersion :  
14 test filename : /tmp/datascience/1_fSX_dS1_rd_fac_0/ior_file_1.dat  
15 access : single-shared-file  
16 type : independent  
17 segments : 1  
18 ordering in a file : sequential  
19 ordering inter file : constant task offset  
20 task offset : 1  
21 nodes : 2  
22 tasks : 24  
23 clients per node : 12  
24 memoryBuffer : CPU  
25 dataAccess : CPU  
26 GPUDirect : 0  
27 repetitions : 1  
28 xfersize : 1 MiB  
29 blocksize : 10 MiB  
30 aggregate filesize : 240 MiB  
31 verbose : 1  
32 stonewallingTime : 100  
33 stoneWallingWearOut : 0  
34  
35 libpil4dfs intercepting summary for ops on DFS:  
36 [read ] 10  
37 [write ] 10  
38  
39 [open ] 2  
40 [stat ] 2  
41 [opendir] 0  
42 [readdir] 0  
43 [link ] 0  
44 [unlink ] 0  
45 [rdlink ] 0  
46 [seek ] 20  
47 [mkdir ] 0  
48 [rmdir ] 0  
49 [rename ] 0  
50 [mmap ] 0  
51  
52 [op_sum ] 44  
53 libpil4dfs intercepting summary for ops on DFS:  
54 [read ] 10  
55 [write ] 10  
56  
57 [open ] 2
```



Monitoring with DAOS DEBUG LOGs Client

```
export D_LOG_FILE="${PBS_O_WORKDIR}/${PBS_JOBID}-${NNODES}/daos_d_log_file.log"
export D_LOG_MASK=info
export D_LOG_FILE_APPEND_PID=1
export D_LOG_STDERR_IN_LOG=1
```

```
≡ daos_d_log_file.log ×
daos > darshan > darshan-daos > test-exps-2 > out1-posix-sx > ≡ daos_d_log_file.log
47 05/01-22:15:40.89 x4210c7s0b0n0 DAOS[45140/45140/0] daos INFO src/client/api/job.c:73 dc_job_init() Using JOBID ENV: DAOS_JOBID
48 05/01-22:15:40.89 x4210c7s0b0n0 DAOS[45139/45139/0] daos INFO src/client/api/job.c:73 dc_job_init() Using JOBID ENV: DAOS_JOBID
49 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] daos INFO src/client/api/job.c:74 dc_job_init() Using JOBID x4210c7s1b0n0-32939
50 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] mgmt INFO src/mgmt/cli_mgmt.c:618 dc_mgmt_net_cfg_init() Using server's value for FI_OFI_RXM_USE_SRX: 1
51 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] mgmt INFO src/mgmt/cli_mgmt.c:672 dc_mgmt_net_cfg_init() Network interface: hsn0, Domain: cxi0, Provider: ofi+cxi, Ranks count: 251
52 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:687 crt_init_opt() libcart (client) v4.9.0 initializing
53 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:76 dump_opt() options:
54 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:77 dump_opt() crt_timeout = 120
55 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:78 dump_opt() max_ctx_num = 1
56 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:79 dump_opt() swim_idx = 0
57 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:80 dump_opt() provider = ofi+cxi
58 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:81 dump_opt() interface = hsn0
59 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:82 dump_opt() domain = cxi0
60 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:83 dump_opt() port = (null)
61 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:85 dump_opt() Flags: fi: 0, use_credits: 0, use_esensors: 0
62 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:88 dump_opt() max_expected_size = 20480
63 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:90 dump_opt() max_unexpect_size = 20480
64 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_internal_types.h:332 crt_env_dump() --- ENV ---
65 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_internal_types.h:345 crt_env_dump() CRT_SECONDARY_PROVIDER = 0
66 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_internal_types.h:345 crt_env_dump() D_LOG_FILE = /lus/flare/projects/datascience/kaushik/daos/darshan/darshan-daos/te
67 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_internal_types.h:345 crt_env_dump() D_LOG_MASK = info
68 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_internal_types.h:345 crt_env_dump() FI_OFI_RXM_USE_SRX = 1
69 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:229 crt_gdata_dump() settings:
70 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:230 crt_gdata_dump() cg_post_init = 512
71 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:231 crt_gdata_dump() cg_post_incr = 512
72 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:232 crt_gdata_dump() cg_timeout = 120
73 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:233 crt_gdata_dump() cg_swim_crt_idx = 1
74 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:234 crt_gdata_dump() cg_credit_ep_ctx = 32
75 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:235 crt_gdata_dump() cg_iv_inline_limit = 19456
76 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:236 crt_gdata_dump() cg_auto_swim_disable = 0
77 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:237 crt_gdata_dump() cg_server = 0
78 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:238 crt_gdata_dump() cg_use_sensors = 0
79 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:239 crt_gdata_dump() cg_provider_is_primary = 1
80 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:240 crt_gdata_dump() cg_rpcid = 0x602724fc00000000
81 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:241 crt_gdata_dump() cg_num_cores = 208
82 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] crt INFO src/crt/crt_init.c:242 crt_gdata_dump() cg_rpc_quota = 64
83 05/01-22:15:40.86 x4210c7s1b0n0 DAOS[32939/32939/0] iv INFO src/crt/crt_iv.c:143 crt_iv_init() max inline buf size is 19456
84 05/01-22:15:41.00 x4210c7s1b0n0 DAOS[32939/32939/0] object INFO src/object/cli_mod.c:201 dc_obj_init() Set object collective operation threshold as 20
85 05/01-22:15:40.84 x4210c7s1b0n0 DAOS[32941/32941/0] daos INFO src/client/api/job.c:74 dc_job_init() Using JOBID x4210c7s1b0n0-32941
86 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32941/32941/0] mgmt INFO src/mgmt/cli_mgmt.c:618 dc_mgmt_net_cfg_init() Using server's value for FI_OFI_RXM_USE_SRX: 1
87 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32941/32941/0] mgmt INFO src/mgmt/cli_mgmt.c:672 dc_mgmt_net_cfg_init() Network interface: hsn3, Domain: cxi3, Provider: ofi+cxi, Ranks count: 251
```


Monitoring with DAOS DEBUG LOGs Server

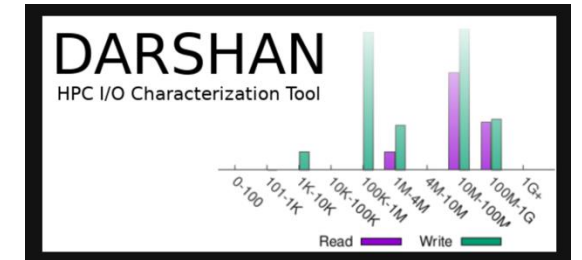
```
daos > darshan > darshan-daos > test-exps-2 > daos-server-logs > 20250501_daos_user_for_kaushik > aurora-daos-0162 > E daos_engine.log.9866.rank=231
0/940 05/01-22:01:48.41 aurora-daos-0162 DAOS(9866/11/33223) pool INFO src/pool/cli.c:553 dc_pool_map_update() 05/01-22:1
67943 05/01-22:01:48.41 aurora-daos-0162 DAOS(9866/11/33280) pool INFO src/pool/cli.c:553 dc_pool_map_update() 05/01-22:1
67942 05/01-22:01:48.41 aurora-daos-0162 DAOS(9866/13/33218) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67943 05/01-22:07:05.87 aurora-daos-0162 DAOS(9866/13/33224) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67944 05/01-22:07:05.87 aurora-daos-0162 DAOS(9866/6/33226) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67945 05/01-22:07:05.87 aurora-daos-0162 DAOS(9866/9/33225) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67946 05/01-22:07:05.87 aurora-daos-0162 DAOS(9866/10/33227) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67947 05/01-22:07:05.87 aurora-daos-0162 DAOS(9866/8/33228) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67948 05/01-22:07:05.87 aurora-daos-0162 DAOS(9866/6/33229) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67949 05/01-22:07:05.87 aurora-daos-0162 DAOS(9866/15/33238) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67950 05/01-22:07:05.87 aurora-daos-0162 DAOS(9866/7/33233) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67951 05/01-22:07:05.87 aurora-daos-0162 DAOS(9866/5/33234) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67952 05/01-22:07:05.87 aurora-daos-0162 DAOS(9866/14/33235) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67953 05/01-22:07:05.87 aurora-daos-0162 DAOS(9866/11/33231) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67954 05/01-22:07:05.87 aurora-daos-0162 DAOS(9866/12/33232) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67955 05/01-22:07:05.87 aurora-daos-0162 DAOS(9866/16/33237) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67956 05/01-22:07:05.87 aurora-daos-0162 DAOS(9866/17/33238) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67957 05/01-22:07:05.87 aurora-daos-0162 DAOS(9866/18/33239) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67958 05/01-22:07:05.87 aurora-daos-0162 DAOS(9866/4/33240) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67959 05/01-22:14:01.54 aurora-daos-0162 DAOS(9866/13/33248) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67960 05/01-22:14:01.54 aurora-daos-0162 DAOS(9866/14/33243) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67961 05/01-22:14:01.54 aurora-daos-0162 DAOS(9866/15/33245) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67962 05/01-22:14:01.54 aurora-daos-0162 DAOS(9866/16/33246) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67963 05/01-22:14:01.54 aurora-daos-0162 DAOS(9866/18/33247) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67964 05/01-22:14:01.54 aurora-daos-0162 DAOS(9866/4/33248) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67965 05/01-22:14:01.54 aurora-daos-0162 DAOS(9866/9/33249) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67966 05/01-22:14:01.54 aurora-daos-0162 DAOS(9866/9/33250) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67967 05/01-22:14:01.54 aurora-daos-0162 DAOS(9866/12/33251) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67968 05/01-22:14:01.54 aurora-daos-0162 DAOS(9866/9/33252) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67969 05/01-22:14:01.54 aurora-daos-0162 DAOS(9866/6/33242) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67970 05/01-22:14:01.54 aurora-daos-0162 DAOS(9866/16/33253) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67971 05/01-22:14:01.54 aurora-daos-0162 DAOS(9866/17/33254) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67972 05/01-22:14:01.54 aurora-daos-0162 DAOS(9866/18/33255) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67973 05/01-22:14:01.54 aurora-daos-0162 DAOS(9866/7/33244) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67974 05/01-22:14:01.54 aurora-daos-0162 DAOS(9866/7/33241) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67975 05/01-22:19:13.12 aurora-daos-0162 DAOS(9866/6/33256) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67976 05/01-22:19:13.12 aurora-daos-0162 DAOS(9866/7/33260) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67977 05/01-22:19:13.12 aurora-daos-0162 DAOS(9866/4/33263) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67978 05/01-22:19:13.12 aurora-daos-0162 DAOS(9866/8/33264) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67979 05/01-22:19:13.12 aurora-daos-0162 DAOS(9866/6/33265) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67980 05/01-22:19:13.12 aurora-daos-0162 DAOS(9866/5/33266) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67981 05/01-22:19:13.12 aurora-daos-0162 DAOS(9866/14/33267) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67982 05/01-22:19:13.12 aurora-daos-0162 DAOS(9866/11/33268) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67983 05/01-22:19:13.12 aurora-daos-0162 DAOS(9866/7/33269) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67984 05/01-22:19:13.12 aurora-daos-0162 DAOS(9866/18/33259) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67985 05/01-22:19:13.12 aurora-daos-0162 DAOS(9866/9/33261) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67986 05/01-22:19:13.12 aurora-daos-0162 DAOS(9866/12/33262) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67987 05/01-22:19:13.12 aurora-daos-0162 DAOS(9866/10/33258) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67988 05/01-22:19:13.12 aurora-daos-0162 DAOS(9866/13/33257) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67989 05/01-22:19:13.12 aurora-daos-0162 DAOS(9866/15/33270) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67990 05/01-22:19:13.12 aurora-daos-0162 DAOS(9866/17/33271) pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
67991 05/01-22:21:17.12 aurora-daos-0162 DAOS(9866/11/33272) vos WARN src/vos/vos_dtx.c:1268 vos_dtx_check_availability() DTX cd1f4a0e.1e59465b93c80000 (state:1, fla
```

```
daos_d_log_file.log
daos > darshan > darshan-daos > test-exps-2 > out1-posix-sx > E daos_d_log_file.log
110 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32941/32941/0] crt INFO src/crt/crt_init.c:234 crt_gdata_dump() cg_ 751f934f
111 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32941/32941/0] crt INFO src/crt/crt_init.c:235 crt_gdata_dump() cg_auto_swim_disable = 0
112 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32941/32941/0] crt INFO src/crt/crt_init.c:236 crt_gdata_dump() cg_server = 0
113 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32941/32941/0] crt INFO src/crt/crt_init.c:237 crt_gdata_dump() cg_use_sensors = 0
114 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32941/32941/0] crt INFO src/crt/crt_init.c:238 crt_gdata_dump() cg_provider_is_primary = 1
115 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32941/32941/0] crt INFO src/crt/crt_init.c:239 crt_gdata_dump() cg_rpcid = 0x1a7f60e300000000
116 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32941/32941/0] crt INFO src/crt/crt_init.c:240 crt_gdata_dump() cg_num_cores = 288
117 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32941/32941/0] crt INFO src/crt/crt_init.c:241 crt_gdata_dump() cg_rpc_quota = 64
118 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32941/32941/0] crt INFO src/crt/crt_init.c:242 crt_gdata_dump() cg_rpc_quota = 19456
119 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32941/32941/0] iv INFO src/crt/crt_iv.c:143 crt_iv_init() max inline buf size is 19456
120 05/01-22:15:40.92 x4210c7s1b0n0 DAOS[32941/32941/0] object INFO src/object/cli_mod.c:281 dc_obj_init() Set object collective operation threshold as 20
121 05/01-22:15:40.92 x4210c7s1b0n0 DAOS[32941/32941/0] object INFO src/object/cli_mod.c:286 dc_obj_init() Disable TX redundancy group verification
122 05/01-22:15:41.01 x4210c7s1b0n0 DAOS[32942/32942/0] pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
123 05/01-22:15:40.84 x4210c7s1b0n0 DAOS[32942/32942/0] daos INFO src/client/api/job.c:74 dc_job_init() Using JOBID x4210c7s1b0n0-32942
124 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] mgmt INFO src/mgmt/cli_mgmt.c:618 dc_mgmt_net_cfg_init() Using server's value for FI_OFI_RXM_USE_SRX: 1
125 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] mgmt INFO src/mgmt/cli_mgmt.c:672 dc_mgmt_net_cfg_init() Network interface: hsn1, Domain: cx11, Provider: ofi+cx11
126 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:687 crt_init_opt() Libcart (client) v4.9.0 initializing
127 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:76 dump_opt() options:
128 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:77 dump_opt() crt_timeout = 120
129 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:78 dump_opt() max_ctx_num = 1
130 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:79 dump_opt() swim_idx = 0
131 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:80 dump_opt() provider = ofi+cx11
132 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:81 dump_opt() interface = hsn1
133 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:82 dump_opt() domain = cx11
134 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:83 dump_opt() port = (null)
135 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:85 dump_opt() Flags: fi, 0, use_credits: 0, use_ensnors: 0
136 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:88 dump_opt() max_expected_size = 20480
137 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:90 dump_opt() max_unexpected_size = 20480
138 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_internal_types.h:332 crt_env_dump() --- ENV ---
139 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_internal_types.h:345 crt_env_dump() CRT_SECONDARY_PROVIDER = 0
140 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_internal_types.h:345 crt_env_dump() D_LOG_FILE = /lus/flare/projects/datascience/kaushik/daos
141 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_internal_types.h:345 crt_env_dump() D_LOG_MASK = info
142 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_internal_types.h:345 crt_env_dump() FI_OFI_RXM_USE_SRX = 1
143 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:229 crt_gdata_dump() settings:
144 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:230 crt_gdata_dump() cg_post_init = 512
145 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:231 crt_gdata_dump() cg_post_incr = 512
146 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:232 crt_gdata_dump() cg_timeout = 120
147 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:233 crt_gdata_dump() cg_swim_crt_idx = 1
148 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:234 crt_gdata_dump() cg_credit_ep_ctx = 32
149 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:235 crt_gdata_dump() cg_iv_inline_limit = 19456
150 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:236 crt_gdata_dump() cg_auto_swim_disable = 0
151 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:237 crt_gdata_dump() cg_server = 0
152 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:238 crt_gdata_dump() cg_use_sensors = 0
153 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:239 crt_gdata_dump() cg_provider_is_primary = 1
154 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:240 crt_gdata_dump() cg_rpcid = 0xc0b0880000000000
155 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:241 crt_gdata_dump() cg_num_cores = 288
156 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] crt INFO src/crt/crt_init.c:242 crt_gdata_dump() cg_rpc_quota = 64
157 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32942/32942/0] iv INFO src/crt/crt_iv.c:143 crt_iv_init() max inline buf size is 19456
158 05/01-22:15:40.92 x4210c7s1b0n0 DAOS[32942/32942/0] object INFO src/object/cli_mod.c:281 dc_obj_init() Set object collective operation threshold as 20
159 05/01-22:15:40.92 x4210c7s1b0n0 DAOS[32942/32942/0] object INFO src/object/cli_mod.c:286 dc_obj_init() Disable TX redundancy group verification
160 05/01-22:15:41.01 x4210c7s1b0n0 DAOS[32942/32942/0] pool INFO src/pool/cli.c:553 dc_pool_map_update() 751f934f: updated pool map: version=0->869
161 05/01-22:15:40.84 x4210c7s1b0n0 DAOS[32945/32945/0] daos INFO src/client/api/job.c:74 dc_job_init() Using JOBID x4210c7s1b0n0-32945
162 05/01-22:15:40.85 x4210c7s1b0n0 DAOS[32945/32945/0] mgmt INFO src/mgmt/cli_mgmt.c:618 dc_mgmt_net_cfg_init() Using server's value for FI_OFI_RXM_USE_SRX: 1
```

* Server logs can be collected on need basis.

Monitoring with Darshan instrumentation

- Darshan, a scalable HPC I/O characterization tool.
- Designed to capture an accurate picture of application I/O behavior.
- Properties such as patterns of access within files.
- Investigate and tune the I/O behavior of complex HPC applications.
- Minimum overhead lightweight design
- Can help uncover critical insights into applications' usage of this novel storage technology.



A shared library your application preloads at runtime, which generates a binary file at program termination, and a suite of utilities to analyze this file.

<https://github.com/darshan-hpc/darshan>

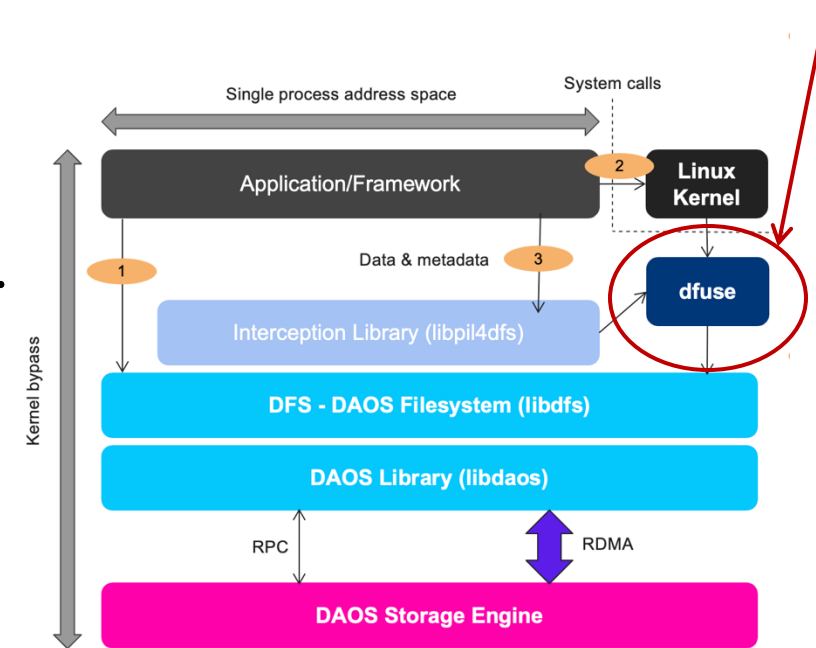
S. Snyder, P. Carns, K. Harms, R. Ross, G. K. Lockwood and N. J. Wright, "Modular HPC I/O Characterization with Darshan," 2016 5th Workshop on Extreme-Scale Programming Tools (ESPT), Salt Lake City, UT, USA, 2016, pp. 9-17, doi: 10.1109/ESPT.2016.006



Darshan + Legacy Posix access Via Fuse

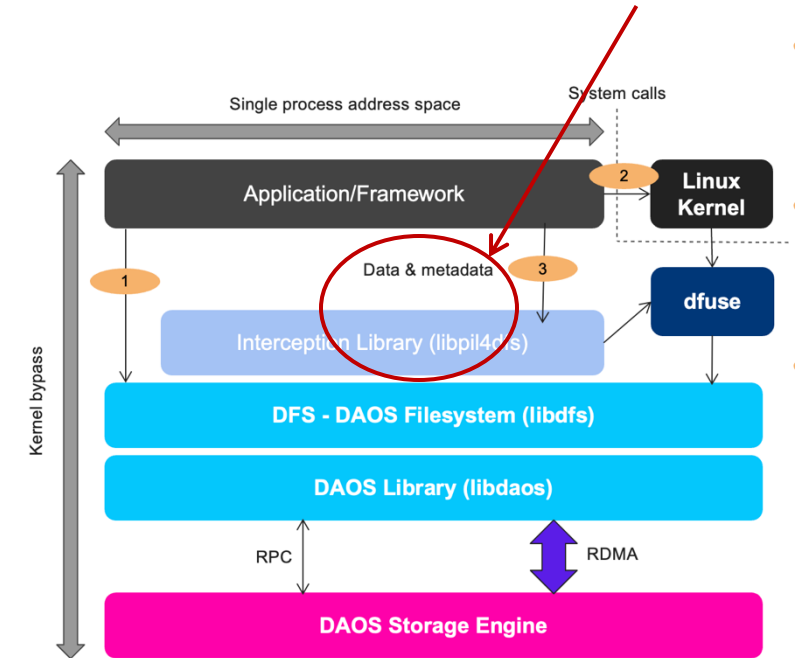
- Users have multiple options for accessing DAOS storage, each with varying performance, ease of adoption, etc.

- No app modifications required
- Performance limited by FUSE architecture, single process, etc.
- Darshan support:
 - Instrumentation via Darshan's POSIX module
 - No insight into DAOS usage by FUSE daemon



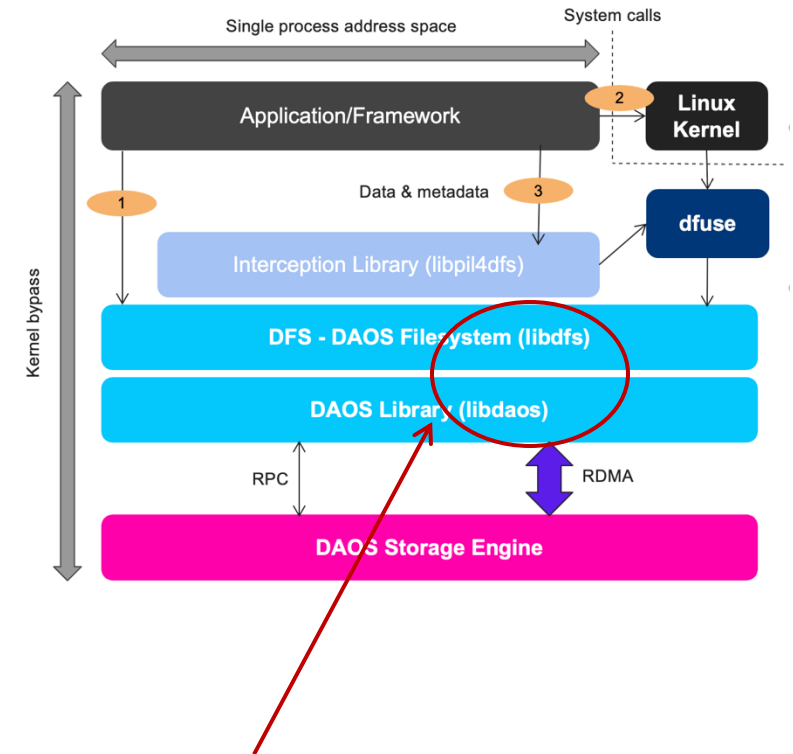
Darshan + Legacy POSIX access via FUSE + interception libs

- No app modifications required
- Performance improved by bypassing FUSE for some or all I/O operations
- Darshan support:
 - POSIX, DFS, and DAOS level instrumentation



Darshan + Direct access to DFS (file) and DAOS (object) APIs

- App or middleware modifications required
- Direct usage of DAOS APIs provides most control over performance, redundancy, etc.
- Darshan support:
 - Full instrumentation of DFS and DAOS APIs



To install your own setup of darshan profiler – on Aurora

```
module use /soft/modulefiles
module load daos
git clone https://github.com/darshan-hpc/darshan.git
./prepare.sh
```

```
./configure --enable-daos-mod \
            --enable-hdf5-mod --with-hdf5= /opt/aurora/24.347.0/spack/unified/0.9.0/install/linux-sles15-x86_64/oneapi-2025.0.5/hdf5-1.14.5-drswwe2
netcdf-1.12.3-jy2mhag \
            --enable-pnetcdf-mod --with-pnetcdf=/opt/aurora/24.347.0/spack/unified/0.9.0/install/linux-sles15-x86_64/oneapi-2025.0.5/parallel-
            --with-log-path=/lus/flare/projects/Aurora_deployment/pkcoff/darshanlog \
            --with-jobid-env=PBS_JOBID \
            --prefix=/lus/flare/projects/Aurora_deployment/pkcoff/lib/darshan \
            'CFLAGS=-O2 -g -Wall' CC=mpicc
make install
```

```
/lus/flare/projects/datascience/kaushik/daos/darshan/darshan-daos/install-daos-darshan/bin/darshan-config --log-path
```

```
/lus/flare/projects/datascience/kaushik/daos/darshan/darshan-daos/install-daos-darshan/bin/darshan-mk-log.pl
```

(should only need to do this one time) that script just sets up that **year/month/day** hierarchy within the log dir
This is where your final darshan logs would go

Then to save to a specific file location: export **DARSHAN_LOGFILE**=<fullpathtodarshanlogfile>

To test if darshan is working

- LD_PRELOAD=/path/to/libdarshan.so DARSHAN_ENABLE_NONMPI=1 cat /some/file
- LD_PRELOAD=/path/to/libdarshan.so mpiexec ...



To install pydarshan on Macbook

Python utilities to interact with Darshan log records

```
brew update
brew install --cask anaconda # restart terminal

conda create --name env-mac-darshan --clone base
conda activate env-mac-darshan

git clone https://github.com/daos-stack/daos.git
cd darshan
git submodule update --init
./prepare
cd darshan-util
../configure --disable-darshan-runtime --prefix=/Users/kvelusamy/Desktop/projects/tools/darshan-util-install \
             --enable-apmpi-mod --enable-apxc-mod CFLAGS='-g -O0 -Wall'

make & make install

cd pydarshan
pip install .

export LD_LIBRARY_PATH=/Users/kvelusamy/Desktop/projects/tools/darshan-util-install/lib/:$LD_LIBRARY_PATH
export PATH=/Users/kvelusamy/Desktop/projects/tools/darshan-util-install/:$PATH
export DYLD_FALLBACK_LIBRARY_PATH=/Users/kvelusamy/Desktop/projects/tools/darshan-util-install/lib/

$ LD_PRELOAD=/Users/kvelusamy/Desktop/projects/tools/darshan-util-install/lib/libdarshan-util.a
  python -m darshan summary ~/Desktop/pyDAOS/kaushikv_ior_id4576025-45130_5-1-80140-6343233643831684606_1.darshan

ls ~/Desktop/pyDAOS/kaushikv_ior_id4576025-45130_5-1-80140-6343233643831684606_1.html
```



```
module use /soft/perftools/darshan/darshan-3.4.7/share/craype-2.x/modulefiles  
module load darshan  
module load python  
./soft/daos/pydarshan_plots_venv/bin/activate  
pip show darshan  
module load darshan  
module load darshan-util  
module load py-darshan
```

```
LD_PRELOAD=/soft/perftools/darshan/darshan-3.4.7/lib/libdarshan-util.so
```

```
# For html file  
python -m darshan summary  
/lus/flare/logs/darshan/aurora/2025/5/21/mgarcia_fornax-GPU-3D_id4916309-158881_5-21-61181-  
16288946849860429419_1.darshan
```

```
# For text form  
/soft/perftools/darshan/darshan-3.4.7/bin/darshan-parser  
/lus/flare/logs/darshan/aurora/2025/5/21/mgarcia_fornax-GPU-3D_id4916309-158881_5-21-61181-  
16288946849860429419_1.darshan > out.txt.
```

```
deactivate
```



Darshan module on aurora and LD_PRELOAD ORDER

```
module use /soft/perftools/darshan/darshan-3.4.7/share/craype-2.x/modulefiles
module load darshan
```

```
LD_PRELOAD=/soft/perftools/darshan/darshan-3.4.7/lib/libdarshan.so: 1
/opt/aurora/24.347.0/spack/unified/0.9.2/install/linux-sles15-
x86_64/oneapi-2025.0.5/hdf5-1.14.5-zrlo32i/lib/libhdf5.so: 2
/opt/aurora/24.347.0/spack/unified/0.9.2/install/linux-sles15-
x86_64/oneapi-2025.0.5/parallel-netcdf-1.12.3-cszcp66/lib/libpnetcdf.so:
/usr/lib64/libpi14dfs.so 3
4
```

If your application is using gpu_tile_compact.sh then this whole LD_PRELOAD will go in your personal copy of the bash script via export.



Demo

```
mpiexec ${EXT_ENV1} -np 24 -ppn 12 --cpu-bind list:4:9:14:19:20:25:30:35:56:61:66:71:72:77:82:87 --no-vni -genvall \
../ior_mdtest_install_bin/bin/ior -a POSIX -i 1 -b 10m -t 1m -D 100 -k -w -r -C -e -v -o /tmp/datascience/1_fSX_dS1_rd_fac_0/ior_file_1.dat
```

```
1  kaushikvelusamy@x4210c7s0b0n0:/lus/flare/projects/datascience/kaushik/daos/darshan/dars
2  IOR-4.1.0+dev: MPI Coordinated Test of Parallel I/O
3  Began      : Thu May  1 22:15:41 2025
4  Command line : /lus/flare/projects/datascience/kaushik/daos/workloads/ior_md_tes
5  Machine     : Linux x4210c7s0b0n0
6  TestID      : 0
7  StartTime   : Thu May  1 22:15:41 2025
8  Path        : /tmp/datascience/1_fSX_dS1_rd_fac_0/ior_file_1.dat
9  FS          : 88.1 TiB   Used FS: 0.5%   Inodes: -0.0 Mi   Used Inodes: 0.0%
10
11  Options:
12  api      : POSIX
13  apiVersion :
14  test filename : /tmp/datascience/1_fSX_dS1_rd_fac_0/ior_file_1.dat
15  access     : single-shared-file
16  type       : independent
17  segments   : 1
18  ordering in a file : sequential
19  ordering inter file : constant task offset
20  task offset : 1
21  nodes       : 2
22  tasks       : 24
23  clients per node : 12
24  memoryBuffer : CPU
25  dataAccess   : CPU
26  GPUDirect    : 0
27  repetitions  : 1
28  xfersize     : 1 MiB
29  blocksize    : 10 MiB
30  aggregate filesize : 240 MiB
31  verbose      : 1
32  stonewallingTime : 100
33  stoneWallingWearOut : 0
34
```



With Darshan-Parser

install-daos-darshan/bin/darshan-parser

kaushikv_ior_id4576025-45130_5-1-80140-6343233643831684606_1.darshan



```
1104 # *****
1105 # DFS module data
1106 # *****
1107
1108 # description of DFS counters:
1109 #   DFS_*: DFS operation counts.
1110 #   OPENS, GLOBAL_OPENS, LOOKUPS, DUPS, READS, READXS, NB_READS, WRITES, WRITEXS, NB_WRITES, GET_SIZES, PUNCHES, REMOVES, STATS are types of operations.
1111 #   DFS_BYTES_*: total bytes read and written.
1112 #   DFS_RW_SWITCHES: number of times access alternated between read and write.
1113 #   DFS_MAX_*_TIME_SIZE: size of the slowest read and write operations.
1114 #   DFS_SIZE_*_*: histogram of read and write access sizes.
1115 #   DFS_ACCESS*_ACCESS: the four most common access sizes.
1116 #   DFS_ACCESS*_COUNT: count of the four most common access sizes.
1117 #   DFS_CHUNK_SIZE: DFS file chunk size.
1118 #   DFS_*_RANK: rank of the processes that were the fastest and slowest at I/O (for shared files).
1119 #   DFS_*_RANK_BYTES: bytes transferred by the fastest and slowest ranks (for shared files).
1120 #   DFS_F_*_START_TIMESTAMP: timestamp of first open/read/write/close.
1121 #   DFS_F_*_END_TIMESTAMP: timestamp of last open/read/write/close.
1122 #   DFS_F_READ/WRITE/META_TIME: cumulative time spent in read, write, or metadata operations.
1123 #   DFS_F_MAX_*_TIME: duration of the slowest read and write operations.
1124 #   DFS_F_*_RANK_TIME: fastest and slowest I/O time for a single rank (for shared files).
1125
1126 #<module> <rank> <record id> <counter> <value> <file name> <mount pt> <fs type>
1127 DFS -1 1327317385700483666 DFS_OPENS 24 ior_file_1.dat UNKNOWN N/A
1128 DFS -1 1327317385700483666 DFS_GLOBAL_OPENS 0 ior_file_1.dat UNKNOWN N/A
1129 DFS -1 1327317385700483666 DFS_LOOKUPS 72 ior_file_1.dat UNKNOWN N/A
1130 DFS -1 1327317385700483666 DFS_DUPS 0 ior_file_1.dat UNKNOWN N/A
1131 DFS -1 1327317385700483666 DFS_READS 240 ior_file_1.dat UNKNOWN N/A
1132 DFS -1 1327317385700483666 DFS_READXS 0 ior_file_1.dat UNKNOWN N/A
1133 DFS -1 1327317385700483666 DFS_WRITES 240 ior_file_1.dat UNKNOWN N/A
1134 DFS -1 1327317385700483666 DFS_WRITEXS 0 ior_file_1.dat UNKNOWN N/A
1135 DFS -1 1327317385700483666 DFS_NB_READS 240 ior_file_1.dat UNKNOWN N/A
1136 DFS -1 1327317385700483666 DFS_NB_WRITES 240 ior_file_1.dat UNKNOWN N/A
1137 DFS -1 1327317385700483666 DFS_GET_SIZES 0 ior_file_1.dat UNKNOWN N/A
1138 DFS -1 1327317385700483666 DFS_PUNCHES 0 ior_file_1.dat UNKNOWN N/A
1139 DFS -1 1327317385700483666 DFS_REMOVES 0 ior_file_1.dat UNKNOWN N/A
1140 DFS -1 1327317385700483666 DFS_STATS 0 ior_file_1.dat UNKNOWN N/A
1141 DFS -1 1327317385700483666 DFS_BYTES_READ 201351360 ior_file_1.dat UNKNOWN N/A
1142 DFS -1 1327317385700483666 DFS_BYTES_WRITTEN 251658240 ior_file_1.dat UNKNOWN N/A
1143 DFS -1 1327317385700483666 DFS_RW_SWITCHES 24 ior_file_1.dat UNKNOWN N/A
1144 DFS -1 1327317385700483666 DFS_MAX_READ_TIME_SIZE 1032 ior_file_1.dat UNKNOWN N/A
1145 DFS -1 1327317385700483666 DFS_MAX_WRITE_TIME_SIZE 1048576 ior_file_1.dat UNKNOWN N/A
1146 DFS -1 1327317385700483666 DFS_SIZE_READ_0_100 24 ior_file_1.dat UNKNOWN N/A
1147 DFS -1 1327317385700483666 DFS_SIZE_READ_100_1K 0 ior_file_1.dat UNKNOWN N/A
1148 DFS -1 1327317385700483666 DFS_SIZE_READ_1K_10K 24 ior_file_1.dat UNKNOWN N/A
1149 DFS -1 1327317385700483666 DFS_SIZE_READ_10K_100K 0 ior_file_1.dat UNKNOWN N/A
1150 DFS -1 1327317385700483666 DFS_SIZE_READ_100K_1M 192 ior_file_1.dat UNKNOWN N/A
1151 DFS -1 1327317385700483666 DFS_SIZE_READ_1M_4M 0 ior_file_1.dat UNKNOWN N/A
1152 DFS -1 1327317385700483666 DFS_SIZE_READ_4M_10M 0 ior_file_1.dat UNKNOWN N/A
1153 DFS -1 1327317385700483666 DFS_SIZE_READ_10M_100M 0 ior_file_1.dat UNKNOWN N/A
1154 DFS -1 1327317385700483666 DFS_SIZE_READ_100M_1G 0 ior_file_1.dat UNKNOWN N/A
1155 DFS -1 1327317385700483666 DFS_SIZE_READ_1G_PLUS 0 ior_file_1.dat UNKNOWN N/A
1156 DFS -1 1327317385700483666 DFS_SIZE_WRITE_0_100 0 ior_file_1.dat UNKNOWN N/A
1157 DFS -1 1327317385700483666 DFS_SIZE_WRITE_100_1K 0 ior_file_1.dat UNKNOWN N/A
1158 DFS -1 1327317385700483666 DFS_SIZE_WRITE_1K_10K 0 ior_file_1.dat UNKNOWN N/A
1159 DFS -1 1327317385700483666 DFS_SIZE_WRITE_10K_100K 0 ior_file_1.dat UNKNOWN N/A
1160 DFS -1 1327317385700483666 DFS_SIZE_WRITE_100K_1M 240 ior_file_1.dat UNKNOWN N/A
1161 DFS -1 1327317385700483666 DFS_SIZE_WRITE_1M_4M 0 ior_file_1.dat UNKNOWN N/A
1162 DFS -1 1327317385700483666 DFS_SIZE_WRITE_4M_10M 0 ior_file_1.dat UNKNOWN N/A
1163 DFS -1 1327317385700483666 DFS_SIZE_WRITE_10M_100M 0 ior_file_1.dat UNKNOWN N/A
1164 DFS -1 1327317385700483666 DFS_SIZE_WRITE_100M_1G 0 ior_file_1.dat UNKNOWN N/A
1165 DFS -1 1327317385700483666 DFS_SIZE_WRITE_1G_PLUS 0 ior_file_1.dat UNKNOWN N/A
1166 DFS -1 1327317385700483666 DFS_ACCESS1_ACCESS 1048576 ior_file_1.dat UNKNOWN N/A
1167 DFS -1 1327317385700483666 DFS_ACCESS2_ACCESS 1032 ior_file_1.dat UNKNOWN N/A
1168 DFS -1 1327317385700483666 DFS_ACCESS3_ACCESS 0 ior_file_1.dat UNKNOWN N/A
1169 DFS -1 1327317385700483666 DFS_ACCESS4_ACCESS 0 ior_file_1.dat UNKNOWN N/A
1170 DFS -1 1327317385700483666 DFS_ACCESS1_COUNT 432 ior_file_1.dat UNKNOWN N/A
1171 DFS -1 1327317385700483666 DFS_ACCESS2_COUNT 24 ior_file_1.dat UNKNOWN N/A
1172 DFS -1 1327317385700483666 DFS_ACCESS3_COUNT 0 ior_file_1.dat UNKNOWN N/A
1173 DFS -1 1327317385700483666 DFS_ACCESS4_COUNT 0 ior_file_1.dat UNKNOWN N/A
1174 DFS -1 1327317385700483666 DFS_CHUNK_SIZE 1048576 ior_file_1.dat UNKNOWN N/A
1175 DFS -1 1327317385700483666 DFS_FASTEST_RANK 19 ior_file_1.dat UNKNOWN N/A
1176 DFS -1 1327317385700483666 DFS_FASTEST_RANK_BYTES 18875400 ior_file_1.dat UNKNOWN N/A
1177 DFS -1 1327317385700483666 DFS_SLOWEST_RANK 17 ior_file_1.dat UNKNOWN N/A
1178 DFS -1 1327317385700483666 DFS_SLOWEST_RANK_BYTES 18875400 ior_file_1.dat UNKNOWN N/A
1179 DFS -1 1327317385700483666 DFS_F_OPEN_START_TIMESTAMP 0.495613 ior_file_1.dat UNKNOWN N/A
1180 DFS -1 1327317385700483666 DFS_F_READ_START_TIMESTAMP 0.538058 ior_file_1.dat UNKNOWN N/A
1181 DFS -1 1327317385700483666 DFS_F_WRITE_START_TIMESTAMP 0.499866 ior_file_1.dat UNKNOWN N/A
1182 DFS -1 1327317385700483666 DFS_F_CLOSE_START_TIMESTAMP 0.506562 ior_file_1.dat UNKNOWN N/A
1183 DFS -1 1327317385700483666 DFS_F_OPEN_END_TIMESTAMP 0.554160 ior_file_1.dat UNKNOWN N/A
1184 DFS -1 1327317385700483666 DFS_F_READ_END_TIMESTAMP 0.548369 ior_file_1.dat UNKNOWN N/A
1185 DFS -1 1327317385700483666 DFS_F_WRITE_END_TIMESTAMP 0.522338 ior_file_1.dat UNKNOWN N/A
1186 DFS -1 1327317385700483666 DFS_F_CLOSE_END_TIMESTAMP 0.554198 ior_file_1.dat UNKNOWN N/A
1187 DFS -1 1327317385700483666 DFS_F_READ_TIME 0.005253 ior_file_1.dat UNKNOWN N/A
1188 DFS -1 1327317385700483666 DFS_F_WRITE_TIME 0.007477 ior_file_1.dat UNKNOWN N/A
1189 DFS -1 1327317385700483666 DFS_F_META_TIME 0.743090 ior_file_1.dat UNKNOWN N/A
1190 DFS -1 1327317385700483666 DFS_F_MAX_READ_TIME 0.000057 ior_file_1.dat UNKNOWN N/A
1191 DFS -1 1327317385700483666 DFS_F_MAX_WRITE_TIME 0.000076 ior_file_1.dat UNKNOWN N/A
1192 DFS -1 1327317385700483666 DFS_F_FASTEST_RANK_TIME 0.023989 ior_file_1.dat UNKNOWN N/A
1193 DFS -1 1327317385700483666 DFS_F_SLOWEST_RANK_TIME 0.041173 ior_file_1.dat UNKNOWN N/A
1194
```

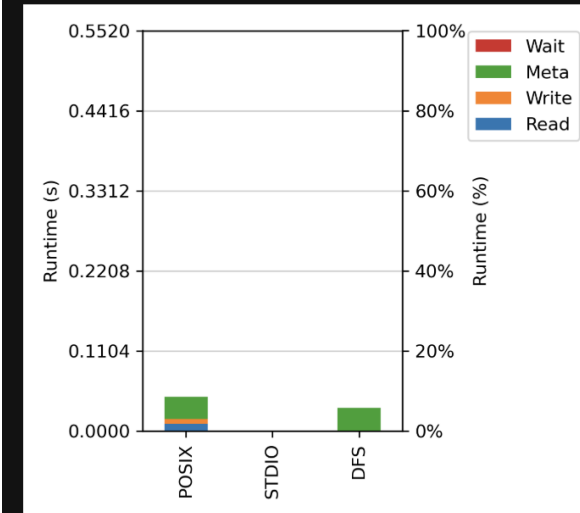


```
mpiexec ${EXT_ENV1} -np 24 -ppn 12 --cpu-bind list:4:9:14:19:20:25:30:35:56:61:66:71:72:77:82:87 --no-vni -genvall \
../ior_mdtest_install_bin/bin/ior -a POSIX -i 1 -b 10m -t 1m -D 100 -k -w -r -C -e -v -o /tmp/datascience/1_fSX_dS1_rd_fac_0/ior_file_1.dat
```

```
1 kaushikvelusamy@x4210c7s0b0n0:/lus/flare/projects/datascience/kaushik/daos/darshan/dars
2 IOR-4.1.0+dev: MPI Coordinated Test of Parallel I/O
3 Began : Thu May 1 22:15:41 2025
4 Command line : /lus/flare/projects/datascience/kaushik/daos/workloads/ior_md_tes
5 Machine : Linux x4210c7s0b0n0
6 TestID : 0
7 StartTime : Thu May 1 22:15:41 2025
8 Path : /tmp/datascience/1_fSX_dS1_rd_fac_0/ior_file_1.dat
9 FS : 88.1 TiB Used FS: 0.5% Inodes: -0.0 Mi Used Inodes: 0.0%
10
11 Options:
12 api : POSIX
13 apiVersion :
14 test filename : /tmp/datascience/1_fSX_dS1_rd_fac_0/ior_file_1.dat
15 access : single-shared-file
16 type : independent
17 segments : 1
18 ordering in a file : sequential
19 ordering inter file : constant task offset
20 task offset : 1
21 nodes : 2
22 tasks : 24
23 clients per node : 12
24 memoryBuffer : CPU
25 dataAccess : CPU
26 GPUDirect : 0
27 repetitions : 1
28 xfersize : 1 MiB
29 blocksize : 10 MiB
30 aggregate filesize : 240 MiB
31 verbose : 1
32 stonewallingTime : 100
33 stonewallingWearOut : 0
34
```

Cross-Module Comparisons

I/O Cost

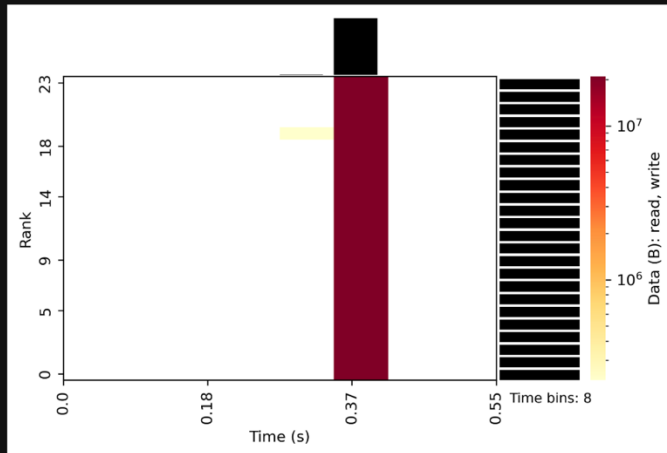


Average (across all ranks) amount of run time that each process spent performing I/O, broken down by access type. See the right edge bar graph on heat maps in preceding section to indicate if I/O activity was balanced across processes. The 'Wait' category is only meaningful for PNETCDF asynchronous I/O operations.

Heat Maps

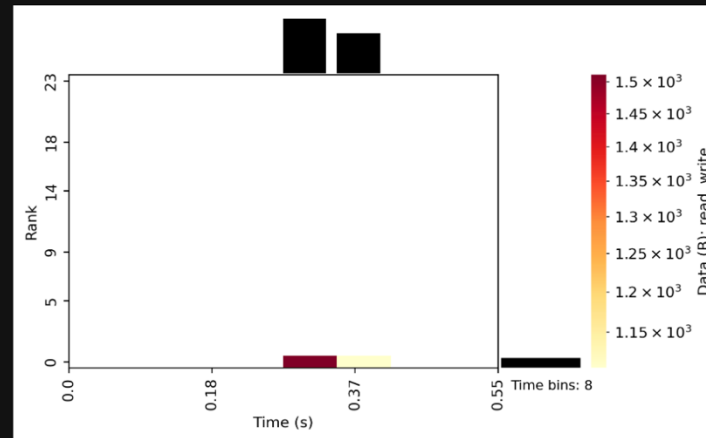
```
mpiexec ${EXT_ENV1} -np 24 -ppn 12 --cpu-bind list:4:9:14:19:20:25:30:35:56:61:66:71:72:77:82:87 --no-vni -genvall \
../ior_mdtest_install_bin/bin/ior -a POSIX -i 1 -b 10m -t 1m -D 100 -k -w -r -C -e -v -o /tmp/datascience/1_fSX_dS1_rd_fac_0/ior_file_1.dat
```

Heat Map: HEATMAP POSIX



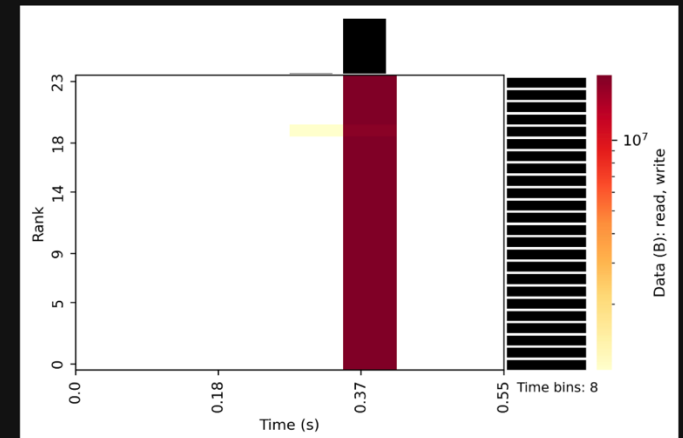
Heat map of I/O (in bytes) over time broken down by MPI rank. Bins are populated based on the number of bytes read/written in the given time interval. The top edge bar graph sums each time slice across ranks to show aggregate I/O volume over time, while the right edge bar graph sums each rank across time slices to show I/O distribution across ranks.

Heat Map: HEATMAP STUDIO



Heat map of I/O (in bytes) over time broken down by MPI rank. Bins are populated based on the number of bytes read/written in the given time interval. The top edge bar graph sums each time slice across ranks to show aggregate I/O volume over time, while the right edge bar graph sums each rank across time slices to show I/O distribution across ranks.

Heat Map: HEATMAP DFS



Heat map of I/O (in bytes) over time broken down by MPI rank. Bins are populated based on the number of bytes read/written in the given time interval. The top edge bar graph sums each time slice across ranks to show aggregate I/O volume over time, while the right edge bar graph sums each rank across time slices to show I/O distribution across ranks.



Per Module Statistics

```
mpiexec ${EXT_ENV1} -np 24 -ppn 12 --cpu-bind list:4:9:14:19:20:25:30:35:56:61:66:71:72:77:82:87 --no-vni -genvall \
../ior_mdtest_install_bin/bin/ior -a POSIX -i 1 -b 10m -t 1m -D 100 -k -w -r -C -e -v -o /tmp/datascience/1_fSX_dS1_rd_fac_0/ior_file_1.dat
```

Per-Module Statistics: POSIX

Overview

files accessed	1
bytes read	240.00 MiB
bytes written	240.00 MiB
I/O performance estimate	8567.13 MiB/s (average)

File Count Summary
(estimated by POSIX I/O access offsets)

	number of files	avg. size	max size
total files	1	240.00 MiB	240.00 MiB
read-only files	0	0	0
write-only files	0	0	0
read/write files	1	240.00 MiB	240.00 MiB

Per-Module Statistics: STDIO

Overview

files accessed	1
bytes read	0 Bytes
bytes written	2.56 KiB
I/O performance estimate	39.07 MiB/s (average)

File Count Summary
(estimated by STDIO I/O access offsets)

	number of files	avg. size	max size
total files	1	2.56 KiB	2.56 KiB
read-only files	0	0	0
write-only files	1	2.56 KiB	2.56 KiB
read/write files	0	0	0

Per-Module Statistics: DFS

Overview

files accessed	1
bytes read	192.02 MiB
bytes written	240.00 MiB
I/O performance estimate	10492.77 MiB/s (average)

File Count Summary
(estimated by DFS I/O access offsets)

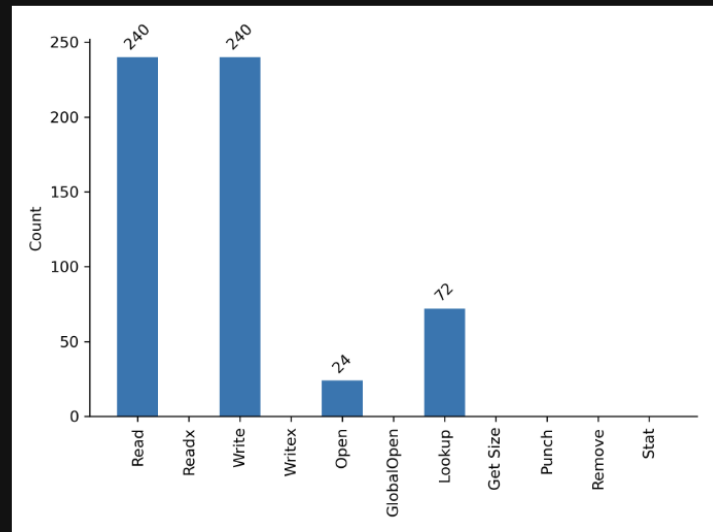
	number of files	avg. size	max size
total files	1	0	0
read-only files	0	0	0
write-only files	0	0	0
read/write files	1	0	0



Operation Counts

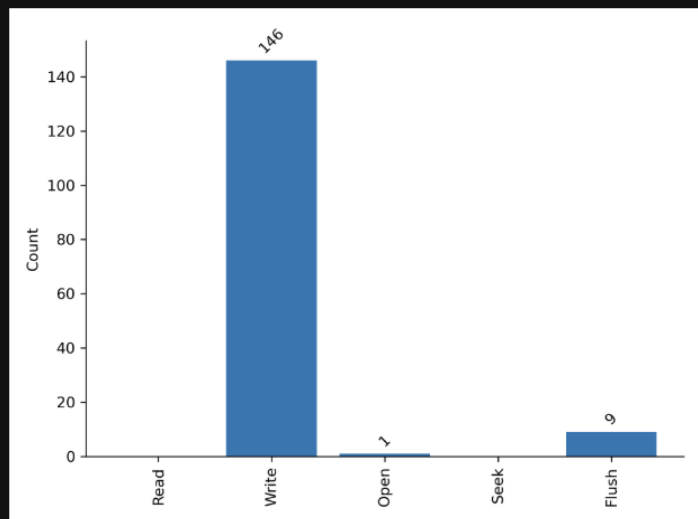
- POSIX
- STDIO
- DFS

Operation Counts



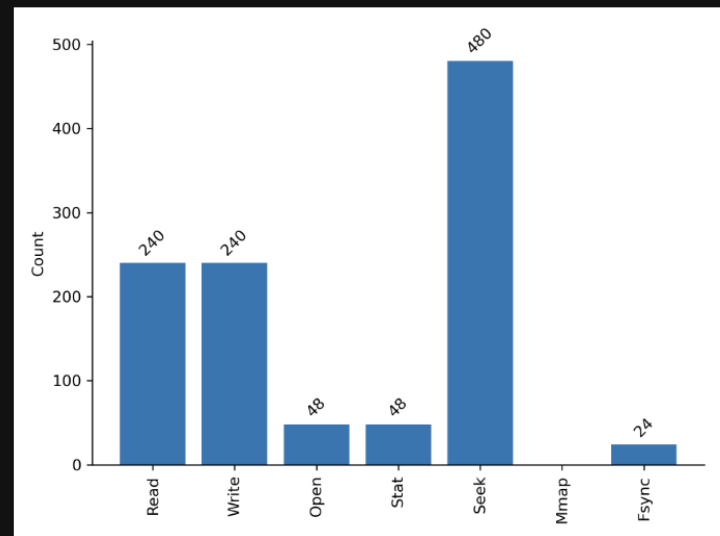
Histogram of I/O operation frequency.

Operation Counts



Histogram of I/O operation frequency.

Operation Counts



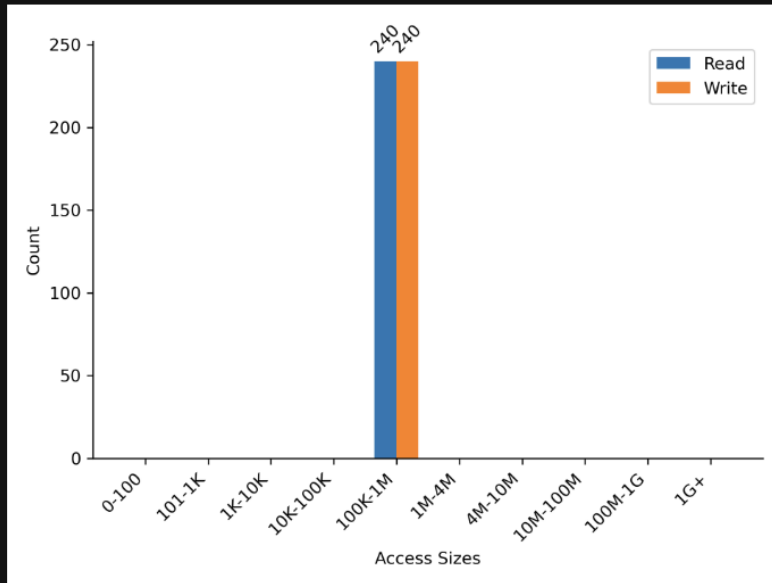
Histogram of I/O operation frequency.



Access Sizes

- POSIX

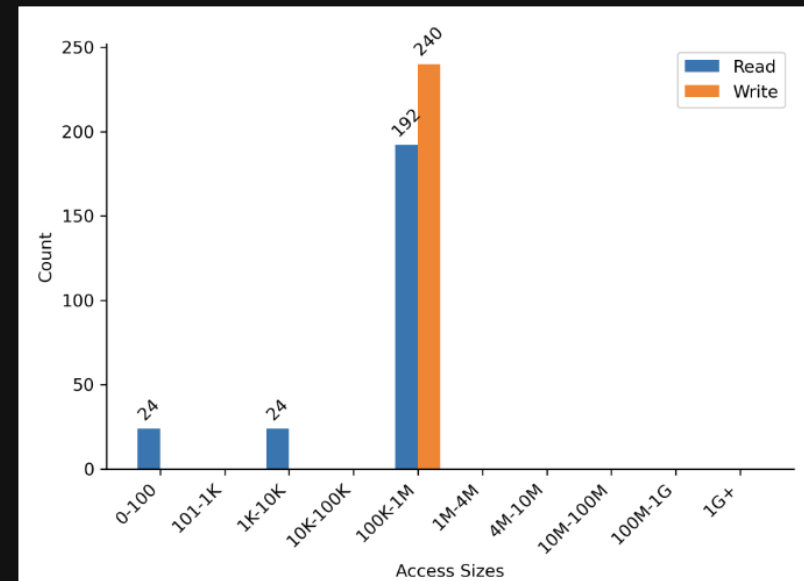
Access Sizes



Histogram of read and write access sizes. The specific values of the most frequently occurring access sizes can be found in the *Common Access Sizes* table.

- DFS

Access Sizes

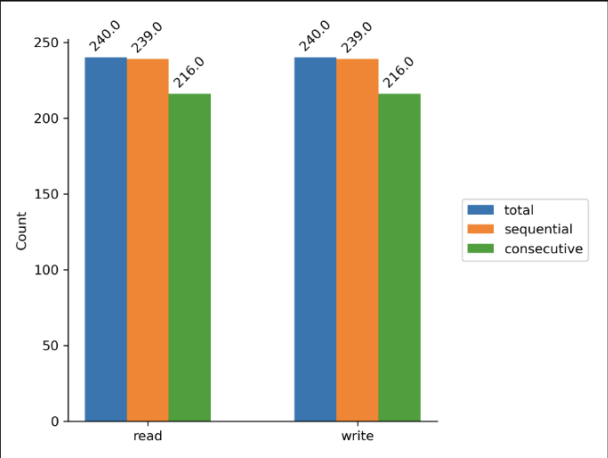


Histogram of read and write access sizes. The specific values of the most frequently occurring access sizes can be found in the *Common Access Sizes* table.

Common Access Sizes

- POSIX

Access Pattern



Sequential (offset greater than previous offset) vs. consecutive (offset immediately following previous offset) file operations. Note that, by definition, the sequential operations are inclusive of consecutive operations.

Common Access Sizes

Access Size	Count
1048576	480

- DFS

Common Access Sizes

Access Size	Count
1048576	432
1032	24

Data Access by Category



Thank You

