

HDF5 DAOS VOL Connector User's Guide

Jordan Henderson, Neil Fortner, Jerome Soumagne

This document aims to be a helpful guide on how to use the HDF5 DAOS VOL connector to leverage the capabilities of the DAOS object storage system within an HDF5 application.

Contents

| | |
|---|-----------|
| 1. Using the DAOS VOL connector with an HDF5 application | 3 |
| 1.1. Writing HDF5 DAOS VOL connector applications | 3 |
| 1.1.1. Skeleton Example | 4 |
| 1.2. Building HDF5 DAOS VOL connector applications | 5 |
| 1.3. Running HDF5 DAOS VOL connector applications | 6 |
| 1.3.1. Runtime Environment | 6 |
| 1.3.2. Example Applications | 6 |
| 2. HDF5 Feature support | 7 |
| 2.1. HDF5 API support | 7 |
| 2.1.1. H5A interface | 7 |
| 2.1.2. H5D interface | 8 |
| 2.1.3. H5F interface | 9 |
| 2.1.4. H5G interface | 11 |
| 2.1.5. H5L interface | 12 |
| 2.1.6. H5O interface | 13 |
| 2.1.7. H5R interface | 13 |
| 2.1.8. H5T interface | 14 |
| 3. Testing | 15 |
| 3.1. HDF5 and dynamically-loaded VOL connectors | 15 |
| 3.2. Testing the DAOS VOL connector | 16 |
| 3.2.1. Generic VOL connector test suite | 16 |
| 3.2.2. DAOS VOL connector-specific test suite | 16 |
| A. Reference Manual | 17 |
| A.1. H5daos_init | 17 |
| A.2. H5daos_term | 18 |
| A.3. H5Pset_fapl_daos | 19 |
| B. Native VOL connector API calls | 20 |
| B.1. H5A interface | 20 |
| B.2. H5D interface | 20 |
| B.3. H5F interface | 21 |

| | |
|------------------------------|----|
| B.4. H5G interface | 22 |
| B.5. H5L interface | 22 |
| B.6. H5O interface | 22 |
| B.7. H5R interface | 22 |
| B.8. H5T interface | 22 |

1. Using the DAOS VOL connector with an HDF5 application

This section outlines the unique aspects of writing, building and running HDF5 applications with the DAOS VOL connector.

1.1. Writing HDF5 DAOS VOL connector applications

Any HDF5 application using the DAOS VOL connector must:

1. Include `daos_vol_public.h`, found in the `include` directory of the DAOS VOL connector installation directory.
2. Link against `libhdf5_vol_daos.so` (or similar), found in the `lib` directory of the DAOS VOL connector installation directory.

An HDF5 DAOS VOL connector application also requires three new function calls in addition to those for an equivalent HDF5 application (see [Appendix A](#) for more details):

- `H5daos_init()` - Initializes the DAOS VOL connector
Called upon application startup, before any file is accessed.
- `H5Pset_fapl_daos()` - Set DAOS VOL connector access on File Access Property List.
Called to prepare a FAPL to open a file through the DAOS VOL connector. See [HDF5 File Access Property Lists](#) for more information about File Access Property Lists.
- `H5daos_term()` - Cleanly shutdown the DAOS VOL connector
Called on application shutdown, after all files have been closed.

1.1.1. Skeleton Example

Below is a no-op application that opens and closes a file using the DAOS VOL connector. For clarity, no error-checking is performed.

```
#include "hdf5.h"
#include "daos_vol_public.h"

int main(void)
{
    hid_t fapl_id;
    hid_t file_id;

    H5daos_init(MPI_COMM_WORLD, pool_uuid, NULL);

    fapl_id = H5Pcreate(H5P_FILE_ACCESS);
    H5Pset_fapl_daos(fapl_id, MPI_COMM_WORLD, MPI_INFO_NULL);
    file_id = H5Fopen("my/file.h5");

    /* operate on file */

    H5Pclose(fapl_id);
    H5Fclose(file_id);

    H5daos_term();

    return 0;
}
```

1.2. Building HDF5 DAOS VOL connector applications

Assuming an HDF5 application has been written following the instructions in the previous section, the application must be built prior to running. In general, the application should be built as normal for any other HDF5 application.

To link in the required libraries, the compiler will likely require the additional linker flags:

```
-lhdf5_vol_daos -luuid
```

However, these flags may vary depending on platform, compiler and installation location of the DAOS VOL connector. It is highly recommended that compilation of HDF5 DAOS VOL connector applications be done using the `h5cc/h5pcc` script, as it will manage linking with the HDF5 library. If HDF5 was built using Autotools, this script will be called `h5pcc` and may be found in the `bin` directory of the HDF5 installation. If HDF5 was built with CMake, this script will simply be called `h5cc` and can be found in the same location. The above notice about additional library linking applies to usage of `h5cc/h5pcc`. For example:

```
h5cc/h5pcc -lhdf5_vol_daos -luuid my_daosvol_application.c -o my_executable
```

1.3. Running HDF5 DAOS VOL connector applications

Running applications that use the HDF5 DAOS VOL connector requires access to a server which implements the [DAOS API](#). Refer to [DAOS Quick Start Guide](#) for more information on the setup process for this.

1.3.1. Runtime Environment

For the DAOS VOL connector to correctly interact with a DAOS API-aware server instance, there are two environment variables which should first be set. These are:

DAOS_POOL - The UUID of the DAOS pool to use

DAOS_SVCL - A comma-separated list of MPI ranks used for `daos_pool_connect()`.
Currently, for simple 1 MPI rank operations this should simply be specified as `DAOS_SVCL=0`

1.3.2. Example Applications

The file `test/daos_vol/test_daos_vol.c` contains several test functions that are examples of applications in miniature, focused on a particular behavior. These mini-applications test a moderate amount of HDF5's public API functionality with the DAOS VOL connector and should be a good indicator of whether the DAOS VOL connector is working correctly in conjunction with a running HDF5 DAOS API-aware instance.

In addition to this file, some of the example C applications included with HDF5 distributions have been adapted to work with the DAOS VOL connector and are included under the top-level `examples` directory in the DAOS VOL connector source root directory.

2. HDF5 Feature support

2.1. HDF5 API support

The following sections serve to illustrate the DAOS VOL connector's support for the HDF5 API, as well as to highlight any differences between the expected behavior of an HDF5 API call versus the actual behavior as implemented by the VOL connector. If a particular HDF5 API call does not appear among these tables, it is most likely a native HDF5-specific API call which cannot be implemented by non-native HDF5 VOL connectors. These types of API calls are listed among the tables in [Appendix B](#).

2.1.1. H5A interface

Supported API calls

| API call | Notes |
|--------------------|---|
| H5Acreate(1/2) | |
| H5Acreate_by_name | |
| H5Aopen(_by_name) | |
| H5Aopen_name | |
| H5Awrite | |
| H5Aread | |
| H5Aclose | |
| H5Aiterate(2) | <ul style="list-style-type: none"> ■ Restarting iteration from an index value is currently unsupported ■ Only H5_ITER_NATIVE is supported for the iteration order |
| H5Aiterate_by_name | <ul style="list-style-type: none"> ■ Restarting iteration from an index value is currently unsupported ■ Only H5_ITER_NATIVE is supported for the iteration order |
| H5Aget_name | |
| H5Aget_space | |
| H5Aget_type | |

Currently unsupported API calls

| API call | Notes |
|-------------------------------|---|
| H5Aopen_by_idx | Currently no support for attribute creation order |
| H5Aopen_idx | Currently no support for attribute creation order |
| H5Aexists(_by_name) | |
| H5Arename(_by_name) | |
| H5Adelete(_by_name/_by_idx) | |
| H5Aget_info(_by_name/_by_idx) | |
| H5Aget_name_by_idx | Currently no support for attribute creation order |
| H5Aget_create_plist | Default ACPL is currently returned |
| H5Aget_storage_size | |

2.1.2. H5D interface**Supported API calls**

| API call | Notes |
|---------------------|-------|
| H5Dcreate(1/2) | |
| H5Dcreate_anon | |
| H5Dopen(1/2) | |
| H5Dwrite | |
| H5Dread | |
| H5Dclose | |
| H5Diterate | |
| H5Dget_space | |
| H5Dget_type | |
| H5Dget_create_plist | |
| H5Dget_access_plist | |

Currently unsupported API calls

| API call | Notes |
|----------------------|--|
| H5Dflush | |
| H5Drefresh | |
| H5Dextend | |
| H5Dset_extent | |
| H5Dget_storage_size | |
| H5Dget_space_status | Space status is currently always set to H5D_SPACE_STATUS_NOT_ALLOCATED |
| H5Dget_offset | |
| H5Dvlen_reclaim | |
| H5Dvlen_get_buf_size | |
| H5Dscatter | |
| H5Dgather | |
| H5Dfill | |

2.1.3. H5F interface

Supported API calls

| API call | Notes |
|---------------------|--|
| H5Fcreate | <ul style="list-style-type: none"> ■ H5F_ACC_EXCL flag is currently not handled correctly |
| H5Fopen | |
| H5Freopen | |
| H5Fis_accessible | |
| H5Fget_create_plist | |
| H5Fget_access_plist | |
| H5Fget_intent | |
| H5Fget_name | |
| H5Fclose | |

Currently unsupported API calls

| API call | Notes |
|------------------|-------|
| H5Fflush | |
| H5Fmount | |
| H5Funmount | |
| H5Fget_obj_count | |
| H5Fget_obj_ids | |

2.1.4. H5G interface

Supported API calls

| API call | Notes |
|-------------------------------|--|
| H5Gcreate(1/2) | |
| H5Gcreate_anon | |
| H5Gopen(1/2) | |
| H5Gclose | |
| H5Gget_create_plist | |
| H5Gget_info(_by_name/_by_idx) | <p>Of the four fields in the H5G_info_t struct:</p> <ul style="list-style-type: none"> ■ storage_type is always set to H5G_STORAGE_TYPE_UNKNOWN ■ nlinks is set appropriately ■ max_corder is currently always set to 0 ■ mounted is currently always set to FALSE <p>H5Gget_info_by_idx is currently unsupported due to the lack of support for link creation order</p> |
| H5Gget_num_objs | |
| H5Glink(2) | Currently only soft link creation is supported |

Currently unsupported API calls

| API call | Notes |
|-----------------------|--|
| H5Gflush | |
| H5Grefresh | |
| H5Gmove(2) | Will be supported once H5Lmove is supported |
| H5Gunlink | Will be supported once H5Ldelete is supported |
| H5Gget_linkval | Will be supported once H5Lget_val is supported |
| H5Gget_objname_by_idx | Will be supported once H5Lget_name_by_idx is supported |

2.1.5. H5L interface

Supported API calls

| API call | Notes |
|----------------------|---|
| H5Lcreate_soft | |
| H5Lexists | |
| H5Literate(_by_name) | <ul style="list-style-type: none"> ■ Restarting iteration from an index value is currently unsupported ■ Only H5_ITER_NATIVE is supported for the iteration order |

Currently unsupported API calls

| API call | Notes |
|----------------------|--|
| H5Lcopy | |
| H5Lmove | |
| H5Lcreate_hard | |
| H5Lcreate_external | |
| H5Lcreate_ud | |
| H5Ldelete(_by_idx) | |
| H5Lget_info(_by_idx) | |
| H5Lget_name_by_idx | Currently no support for link creation order |
| H5Lget_val(_by_idx) | |
| H5Lvisit(_by_name) | |
| H5Lregister | |
| H5Lunregister | |
| H5Lis_registered | |
| H5Lunpack_elink_val | |

2.1.6. H5O interface**Supported API calls**

| API call | Notes |
|-----------------|-------|
| H5Oopen | |
| H5Oopen_by_addr | |
| H5Oclose | |

Currently unsupported API calls

| API call | Notes |
|-----------------------|-------|
| H5Oopen_by_idx | |
| H5Oexists_by_name | |
| H5Ovisit(1/2) | |
| H5Ovisit_by_name(1/2) | |
| H5Olink | |
| H5Ocopy | |
| H5Oflush | |
| H5Orefresh | |
| H5Oincr_refcount | |
| H5Odecr_refcount | |

2.1.7. H5R interface**Supported API calls**

| API call | Notes |
|----------|-------|
| | |

Currently unsupported API calls

| API call | Notes |
|----------------------|-----------------------------|
| H5Rcreate | |
| H5Rdereference(1/2) | Causes an assertion failure |
| H5Rget_name | |
| H5Rget_obj_type(1/2) | |
| H5Rget_region | |

2.1.8. H5T interface**Supported API calls**

| API call | Notes |
|---------------------|-------|
| H5Tcommit(1/2) | |
| H5Tcommit_anon | |
| H5Topen(1/2) | |
| H5Tclose | |
| H5Tget_create_plist | |

Currently unsupported API calls

| API call | Notes |
|------------|-------|
| H5Tflush | |
| H5Trefresh | |

3. Testing

3.1. HDF5 and dynamically-loaded VOL connectors

HDF5 has the capability to dynamically load and use a VOL connector for running tests with. While several HDF5 tests have been updated to take advantage of this capability, please be aware that many of these tests are likely to fail or crash due to their native HDF5-specific nature.

In order to choose a particular VOL connector to use for testing, two initial steps must be taken. First, one must help HDF5 locate the VOL connector by pointing to the directory which contains the built library. This can be accomplished by setting the environment variable `HDF5_PLUGIN_PATH` to this directory. Next, HDF5 needs to know the name of which library to use, which is configured by setting the environment variable `HDF5_VOL_CONNECTOR` to the name of the connector.

In order to use the DAOS VOL connector, the aforementioned environment variables should be set as:

```
HDF5_PLUGIN_PATH=/daos/vol/installation/directory/lib  
HDF5_VOL_CONNECTOR=daos
```

Having completed this step, HDF5 will be setup to load the DAOS VOL connector and use it for testing.

3.2. Testing the DAOS VOL connector

3.2.1. Generic VOL connector test suite

In order to test VOL connectors to make sure that they are functioning as expected, a suite of tests which only use the public HDF5 API has been written. This suite of tests is available under the path:

```
test/vol/vol_test.c
```

Currently, this test suite does not have the capability to query what kind of functionality a VOL connector supports and therefore a test will fail if it uses an HDF5 API call which is not implemented, or which is specifically unsupported, in a given VOL connector.

To test the DAOS VOL connector with this test suite, first refer to [Starting the DAOS Server](#) and [Runtime Environment](#) to make sure that the DAOS Server is up and running and your environment is setup correctly. Once that is done, tests can be run against the DAOS VOL connector and it should be as simple as running:

```
make check
```

for Autotools builds, or:

```
ctest .
```

for CMake builds¹.

3.2.2. DAOS VOL connector-specific test suite

There is no DAOS specific test as of this writing.

¹Automated testing is still under development for CMake builds.

A. Reference Manual

A.1. H5daos_init

```
herr_t H5daos_init(MPI_Comm pool_comm, uuid_t pool_uuid, char *pool_grp);
```

Purpose:

Initialize the DAOS VOL connector.

Description:

H5daos_init initializes the VOL connector by connecting to the pool and registering the connector with the library. pool_comm identifies the communicator used to connect to the DAOS pool. This should include all processes that will participate in I/O. This call is collective across pool_comm.

Parameters:

| | |
|--------------------|--|
| MPI_Comm pool_comm | IN: Communicator used for connecting to the pool |
| uuid_t pool_uuid | IN: UUID to identify a pool |
| char *pool_grp | IN: Process set name of the DAOS servers managing the pool |

Returns:

Returns a non-negative value if successful; otherwise returns a negative value.

A.2. H5daos_term

```
herr_t H5daos_term(void);
```

Purpose:

Terminate the DAOS VOL connector.

Description:

H5daos_term terminates the DAOS VOL connector.

Parameters:

None.

Returns:

Returns a non-negative value if successful; otherwise returns a negative value.

A.3. H5Pset_fapl_daos

```
herr_t H5Pset_fapl_daos(hid_t fapl_id, MPI_Comm comm, MPI_Info info);
```

Purpose:

Set the file access property list to use the DAOS VOL connector.

Description:

H5Pset_fapl_daos modifies the file access property list to use the DAOS VOL connector. file_comm and file_info identify the communicator and info object used to coordinate actions on file create, open, flush, and close.

Parameters:

| | |
|--------------------|----------------------------------|
| hid_t fapl_id | IN: File access property list ID |
| MPI_Comm file_comm | IN: MPI Communicator |
| MPI_Info file_info | IN: MPI Info |

Returns:

Returns a non-negative value if successful; otherwise returns a negative value.

B. Native VOL connector API calls

The following HDF5 API calls are specific to the native HDF5 VOL connector and thus are not able to be implemented by the DAOS VOL connector (or other VOL connectors):

B.1. H5A interface

| API call | Notes |
|------------------|------------------------------------|
| H5Aiterate1 | Deprecated in favor of H5Aiterate2 |
| H5Aget_num_attrs | Deprecated in favor of H5Oget_info |

B.2. H5D interface

| API call | Notes |
|---------------------------|-------|
| H5Dformat_convert | |
| H5Dget_chunk_index_type | |
| H5Dget_chunk_storage_size | |
| H5Dread_chunk | |
| H5Dwrite_chunk | |

B.3. H5F interface

| API call | Notes |
|---------------------------------|--|
| H5Fis_hdf5 | Uses a default FAPL so can only ever be routed through the native HDF5 VOL connector |
| H5Fget_vfd_handle | |
| H5Fget_freespace | |
| H5Fget_filesize | |
| H5Fget_file_image | |
| H5Fget_mdc_config | |
| H5Fset_mdc_config | |
| H5Fget_mdc_hit_rate | |
| H5Fget_mdc_size | |
| H5Freset_mdc_hit_rate_stats | |
| H5Fget_info(1/2) | |
| H5Fget_metadata_read_retry_info | |
| H5Fget_free_sections | |
| H5Fclear_elink_file_cache | |
| H5Fstart_swmr_write | |
| H5Fstart_mdc_logging | |
| H5Fstop_mdc_logging | |
| H5Fget_mdc_logging_status | |
| H5Fset_libver_bounds | |
| H5Fformat_convert | |
| H5Freset_page_buffering_stats | |
| H5Fget_page_buffering_stats | |
| H5Fget_mdc_image_info | |
| H5Fget_eoa | |
| H5Fincrement_filesize | |
| H5Fget_dset_no_attrs_hint | |
| H5Fset_dset_no_attrs_hint | |
| H5Fset_latest_format | |
| H5Fset_mpi_atomicity | |
| H5Fget_mpi_atomicity | |

B.4. H5G interface

| API call | Notes |
|-----------------------|--|
| H5Gset_comment | Deprecated in favor of H5Oset_comment/H5Oset_comment_by_name |
| H5Gget_comment | Deprecated in favor of H5Oget_comment/H5Oget_comment_by_name |
| H5Giterate | Deprecated in favor of H5Literate |
| H5Gget_objinfo | Deprecated in favor of H5Lget_info/H5Oget_info |
| H5Gget_objtype_by_idx | Deprecated in favor of H5Lget_info/H5Oget_info |

B.5. H5L interface

| API call | Notes |
|----------|-------|
| | |

B.6. H5O interface

| API call | Notes |
|-----------------------------|--|
| H5Oget_info(1/2) | |
| H5Oget_info_by_name(1/2) | |
| H5Oget_info_by_idx(1/2) | |
| H5Oset_comment(_by_name) | Deprecated in favor of using attributes on objects |
| H5Oget_comment(_by_name) | Deprecated in favor of using attributes on objects |
| H5Oare_mdc_flushes_disabled | |
| H5Oenable_mdc_flushes | |
| H5Odisable_mdc_flushes | |

B.7. H5R interface

| API call | Notes |
|----------|-------|
| | |

B.8. H5T interface

| API call | Notes |
|----------|-------|
| | |

Revision History

| | |
|----------------|-------------|
| March 29, 2019 | First Draft |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |