HDF5 Virtual Object Layer (VOL) User Guide

The HDF Group

11th November 2019

Contents

1 Introduction **1**

2 Quickstart **1**

3 Loading and Registering Connectors **2**

3.1 VOL Connector Search Path 2

3.2 Connector Versioning 2

3.3 Connector-Specific Registration Calls 2

3.4 `H5Pset_vol()` 2

3.5 Connector Compatibility 2

3.6 Connection Strings 2

3.7 Environment Variable 2

4 Adapting HDF5 Software to Use the VOL **3**

4.1 Specific API Call Substitutions 3

4.2 Protect Native-Only API Calls 4

5 Using VOL Connectors With The HDF5 Command-Line Tools **4**

6 Compatibility **4**

6.1 Feature Flags 4

6.2 List of HDF5 Native VOL API Calls 4

6.3 List of HDF5 VOL-Independent API Calls 5

6.4 List of HDF5 API Calls By Callback 6

1 Introduction

2 Quickstart

Read The Documentation For The New VOL Connector

Many VOL connectors will require specific setup and configuration of both the application and the storage. Specific permissions may have to be set, configuration files constructed, and connector-specific setup calls may need to be invoked in the application. In many cases, converting software to use a new VOL connector will be more than just a straightforward drop-in replacement done by specifying a name in the VOL plugin environment variable.

Use A VOL-Enabled HDF5 Library

The virtual object layer was introduced in HDF5 1.12.0, so you will need that version or later to use it. The particular configuration of the library (serial vs parallel, thread-safe, debug vs production/release) does not matter. The VOL is a fundamental part of the library and cannot be disabled, so any build will do.

On Windows, it's probably best to use the same debug vs release configuration for the application and all libraries in order to avoid C runtime (CRT) issues. On pre-2015 versions of Visual Studio, you'll probably also want to stick to the same Visual Studio version (and thus same CRT version) as well.

When working with a debug HDF5 library, it's probably also wise to build with the "memory sanity checking" feature disabled to avoid accidentally clobbering our memory tracking infrastructure when dealing with buffers obtained from the HDF5 library.

Make Sure The VOL Connector Is In The Search Path

The default location for all HDF5 plugins is set at configure time when building the HDF5 library. This is true for both CMake and the Autotools.

Default locations:

POSIX systems:	`/usr/local/hdf5/lib/plugin`
Windows:	`%ALLUSERSPROFILE%/hdf5/lib/plugin`

These default locations can be overridden by setting the `HDF5_PLUGIN_PATH` environment variable. There are also public `H5PL` API calls which can be used to add, modify, and remove search paths. The library will only look for plugins in the specified plugin paths. By default, it will NOT find plugins that are simply located in the same directory as the executable.

Update Your Code To Load And Use A VOL Connector

How this is done depends on the VOL connector, which may have special API calls for setup or require appropriate configuration files, but the most generic way to modify a program to use a specific VOL connector would be to `use H5Pset_vol()` to set the VOL connector in the file access property list (fapl) that will be used to open the file.

You will also need to protect any API calls which are only implemented in the native VOL connector as those calls will fail when using a non-native VOL connector. See the section entitled "Adapting HDF5 Software to Use the VOL", below. A list of native VOL API calls has been included in an appendix.

Optional: Set The VOL Connector Via The Environment Variable

3 Loading and Registering Connectors

3.1 VOL Connector Search Path

3.2 Connector Versioning

3.3 Connector-Specific Registration Calls

3.4 `H5Pset_vol()`

3.5 Connector Compatibility

PRELIMINARY

This feature is under active development and is either incomplete or may change in behavior before the final release.

3.6 Connection Strings

3.7 Environment Variable

The HDF5 library allows specifying a default VOL connector via an environment variable: `HDF5_VOL_CONNECTOR`. The value of this environment variable should be set to "*vol_connector_name <parameters>*".

This will perform the equivalent of:

1. `H5VLregister_connector_by_name()` using the specified connector name

2. `H5VLconnector_str_to_info()` using the specified parameters. This will go through the connector we got from the previous step and should return a VOL info struct from the parameter string in the environment variable.
3. `H5Pset_vol()` on the default fapl using the obtained ID and info.

The environment variable is parsed once, at library startup. Since the environment variable scheme just changes the default connector, it can be overridden by subsequent calls to `H5Pset_vol()`. The <*parameters*> is optional, so for connectors which do not require any special configuration parameters you can just set the environment variable to the name.

4 Adapting HDF5 Software to Use the VOL

4.1 Specific API Call Substitutions

`H5Fis_hdf5()` → `H5Fis_accessible()`

`H5Fis_hdf5()` does not take a file access property list (fapl). As this is where the VOL connector is specified, this call cannot be used with aribtrary connectors. As a VOL-enabled replacement, `H5Fis_accessible()` has been added to the library. It has the same semantics as `H5Fis_hdf5()`, but takes a fapl so it can work with any VOL connector.

Note that, at this time, `H5Fis_hdf5()` *always* uses the native VOL connector, regardless of the settings of environment variables, etc.

`H5Oget_info[1|2]()` → `H5Ofuture_function_goes_here()`

PRELIMINARY

This feature is under active development and is either incomplete or may change in behavior before the final release.

The `H5Oget_info1()` and `H5Oget_info2()` are often used by user code to obtain information about an object in the file, however these calls return a struct which contains native information and are thus currently handled via the native connector's *object optional* callback.

These calls will be separated into two functions: one for getting generic object info (which will go through the *object get* callback), and another for returning native file format information (which will go through the native connector's *object optional* callback).

4.2 Protect Native-Only API Calls

5 Using VOL Connectors With The HDF5 Command-Line Tools

6 Compatibility

6.1 Feature Flags

PRELIMINARY

This feature is under active development and is either incomplete or may change in behavior before the final release.

6.2 List of HDF5 Native VOL API Calls

These API calls will probably fail when used with terminal VOL connectors other than the native HDF5 file format connector. Their use should be protected in code that uses arbitrary VOL connectors.

H5Aget_num_attrs (deprecated) H5Aiterate1 (deprecated)
H5Ddebug H5Dformat_convert H5Dget_chunk_index_type H5Dget_chunk_info H5Dget_chunk_info_by_coord H5Dget_chunk_storage_size H5Dget_num_chunks H5Dread_chunk H5Dwrite_chunk
H5FD*
H5Fclear_elink_file_cache H5Fformat_convert H5Fget_dset_no_attrs_hint H5Fget_eoa H5Fget_file_image H5Fget_filesize H5Fget_free_sections H5Fget_freespace H5Fget_info1 (deprecated) H5Fget_info2 H5Fget_mdc_config H5Fget_mdc_hit_rate H5Fget_mdc_image_info H5Fget_mdc_logging_status H5Fget_mdc_size H5Fget_metadata_read_retry_info

H5Fget_mpi_atomicity H5Fget_page_buffering_stats H5Fget_vfd_handle H5Fincrement_filesize H5Fis_hdf5 (deprecated) H5Freset_mdc_hit_rate_stats H5Freset_page_buffering_stats H5Fset_dset_no_attrs_hint H5Fset_latest_format (deprecated) H5Fset_libver_bounds H5Fset_mdc_config H5Fset_mpi_atomicity H5Fstart_mdc_logging H5Fstart_swmr_write H5Fstop_mdc_logging
H5Gget_comment (deprecated) H5Giterate (deprecated) H5Gget_info H5Gget_info_by_name H5Gget_info_by_idx H5Gget_objinfo (deprecated) H5Gget_objname_by_idx (deprecated) H5Gget_objtype_by_idx (deprecated) H5Gset_comment (deprecated)
H5Oare_mdc_flushes_disabled H5Odisable_mdc_flushes H5Oenable_mdc_flushes H5Oget_comment H5Oget_comment_by_name H5Oget_info_by_idx1 (deprecated) H5Oget_info_by_idx2 H5Oget_info_by_name1 (deprecated) H5Oget_info_by_name2 H5Oget_info1 (deprecated) H5Oget_info2 H5Oset_comment H5Oset_comment_by_name

Table 1: Alphabetical list of HDF5 API calls specific to the native VOL connector

6.3 List of HDF5 VOL-Independent API Calls

These HDF5 API calls do not depend on a particular VOL connector being loaded.

H5*
H5Dfill H5Dgather

H5Diterate
H5Dscatter
H5Dvlen_reclaim (deprecated)
H5Dvlen_get_buf_size
H5E*
H5I*
H5Lis_registered
H5Lregister
H5Lunpack_elink_val
H5Lunregister
H5PL*
H5P*
H5S*
H5T* (non-committed)
H5VL*
H5Z*

Table 2: Alphabetical list of VOL-independent HDF5 API calls

6.4 List of HDF5 API Calls By Callback

VOL Callback	HDF5 API Call
FILE	
create	H5Fcreate
open	H5Fopen
get	H5Fget_access_plist
	H5Fget_create_plist
	H5Fget_fileno
	H5Fget_intent
	H5Fget_name
	H5Fget_obj_count
	H5Fget_obj_ids
specific	H5Fdelete
	H5Fflush
	H5Fis_accessible
	H5Fis_hdf5 (deprecated, hard-coded to use native connector)
	H5Fmount
	H5Freopen
	H5Funmount
close	H5Fclose
GROUP	
create	H5Gcreate1 (deprecated)
	H5Gcreate2
	H5Gcreate_anon
open	H5Gopen1 (deprecated)
	H5Gopen2
get	H5Gget_create_plist
	H5Gget_info

	H5Gget_info_by_idx
	H5Gget_info_by_name
	H5Gget_num_objs (deprecated)
specific	H5Gflush
	H5Grefresh
close	H5Gclose
DATASET	
create	H5Dcreate1 (deprecated)
	H5Dcreate2
open	H5Dopen1 (deprecated)
	H5Dopen2
read	H5Dread
write	H5Dwrite
get	H5Dget_access_plist
	H5Dget_create_plist
	H5Dextend
	H5Dget_offset
	H5Dget_space
	H5Dget_space_status
	H5Dget_storage_size
	H5Dget_type
specific	H5Dextend (deprecated)
	H5Dflush
	H5Drefresh
	H5Dset_extent
close	H5Dclose
OBJECT	
open	H5Oopen
	H5Oopen_by_addr
	H5Oopen_by_idx
	H5Oopen_by_name
copy	H5Ocopy
get	N/A
specific	H5Odecr_refcount
	H5Oexists_by_name
	H5Oflush
	H5O_incr_refcount
	H5Orefresh
	H5Ovisit_by_name1 (deprecated)
	H5Ovisit_by_name2
	H5Ovisit1 (deprecated)
	H5Ovisit2
close	H5Oclose
LINK	
create	H5Glink (deprecated)
	H5Glink2 (deprecated)
	H5Lcreate_hard
	H5Lcreate_soft
	H5Lcreate_ud

	H5Olink
copy	H5Lcopy
move	H5Gmove (deprecated) H5Gmove2 (deprecated) H5Lmove
get	H5Gget_linkval (deprecated) H5Lget_info H5Lget_info_by_idx H5Lget_name_by_idx H5Lget_val H5Lget_val_by_idx
specific	H5Gunlink (deprecated) H5Ldelete H5Ldelete_by_idx H5Lexists H5Literate H5Literate_by_name H5Lvisit H5Lvisit_by_name
DATATYPE	
commit	H5Tcommit1 (deprecated) H5Tcommit2 H5Tcommit_anon
open	H5Topen1 (deprecated) H5Topen2
get	H5Tget_create_plist
specific	H5Tflush H5Trefresh
close	H5Tclose
ATTRIBUTE	
create	H5Acreate1 (deprecated) H5Acreate2 H5Acreate_by_name
open	H5Aopen H5Aopen_by_idx H5Aopen_by_name H5Aopen_idx (deprecated) H5Aopen_name (deprecated)
read	H5Aread
write	H5Awrite
get	H5Aget_get_create_plist H5Aget_info H5Aget_info_by_idx H5Aget_info_by_name H5Aget_name H5Aget_name_by_idx H5Aget_space H5Aget_storage_size H5Aget_type

specific	H5Adelete H5Adelete_by_idx H5Adelete_by_name H5Aexists H5Aexists_by_name H5Aiterate1 (deprecated) H5Aiterate2 H5Aiterate_by_name H5Arename H5Arename_by_name
close	H5Aclose

Table 3: Breakdown of HDF5 API calls by VOL callback

References