# New Features in HDF5 Release 1.8

HDF5 Release 1.8.0 represents a major update to the HDF5 Library, utilities, and file format. The HDF5 development team has attempted to provide new capabilities and improve performance while retaining compatibility with previous releases.

The new features are briefly described below, but first a few words regarding the compatibility solutions.

## Compatibility Issues and Solutions

When new features and optimizations are introduced, as is certainly the case in this release, there is always the risk of creating compatibility problems. These problems can arise either with an application that must be ported to the new release (or cannot be ported, for any of a number of reasons), with applications based on a prior release that must read files created by the new release, or with files created by an older release that must work with an application based on the new release. The HDF5 team has made a concerted effort to provide a full range of compatibility solutions, hopefully addressing all of the situations a user or application is likely to encounter.

**Interface — Backward and Forward API Compatibility:**

This release contains many new features and related API routines, but at the same time attempts to provide stability for applications by continuing to make existing API routines available and by operating in a backwardly compatible manner, whenever possible.

*—API Compatibility Macros in HDF5* discusses the specifics of API compatibility and configuration options with respect to new features.

**Format — Backward and Forward Format Compatibility:**

The HDF5 Library Release 1.8.0 reads all existing HDF5 files, from this or any prior release. Although this release contains features that require additions and/or changes to the HDF5 file format, by default this release will write out files that conform to a “maximum compatibility” principle. That is, files are written with the earliest version of the file format that describes the information, rather than always using the latest version possible. This provides the best forward compatibility by allowing the maximum number of older versions of the library to read files produced with this release.

If library features are used that require new file format features, or if the application requests that the library write out only the latest version of the file format, the files produced with this version of the library may not be readable by older versions of the HDF5 library.

*—New Features in HDF5 Release 1.8.0 and Backward/Forward Format Compatibility Issues* discusses the new features in the release from the point of view of their impact on format comaptibility.

## New Features

New features are briefly described in this section. Further, instructional example codes for several of these features are provided here:

`http://www.hdfgroup.org/ftp/HDF5/examples/`

While all new APIs are documented in the *—HDF5 Reference Manual*, there has not been time yet to describe all of them in the *—HDF5 User’s Guide*.

### Object Format Control

- Select format limits by setting library version bounds
  Tunable properties enable the creation of files selectively compatible with older HDF5 applications and libraries. This feature enables the library, and thus an application, to create files that can be read by specific older HDF5 libraries and tools and by applications that those same use libraries.

  This is accomplished with the function H5Pset_libver_bounds, which sets the lower and upper bounds on allowable formats. The lower bound is determined by specifying the earliest library whose format may be used for an object; the upper bound is determined by specifying the latest library whose format may be used for an objects.

  The function H5Pget_libver_bounds can be used to retrieve the current settings.

### Groups and Links

- Configurable Compact-or-Indexed Link Storage
  Compact small groups and more scalable large groups

  For groups with only a few links, compact link storage allows groups containing only a few links to take up much less space in the file.

  On the other hand, an improved implementation of indexed link storage provides a faster and more scalable method for storing and working with large groups containing many links.

  The threshold for switching between the compact and indexed storage formats is configurable according an application's or a user community's expected use cases using the function H5Pset_link_phase_change.

  The function H5Pget_link_phase_change can be used to retrieve the current settings.
- External Links
  Links in a group that link to objects in a different HDF5 file

  External links allow a group to include objects in another HDF5 file and enable the library to access those objects as if they are in the current file. In this manner, a group may appear to directly contain datasets, named datatypes, and even groups that are actually in a different file. This feature is implemented via a suite of functions that create and manage the links, define and retrieve paths to external objects, and interpret link names:
  H5Lcreate_external
  H5Lget_info
  H5Lget_val
  H5Lunpack_elink_val
  H5Pset_elink_prefix
  H5Pget_elink_prefix
- User-defined Links
  Customized link types

  The user-defined link feature enables the definition of customized types of links that meet specific community or application needs. This feature is implemented via a suite of functions that define, create, register and unregister the link types:
  H5Lcreate_ud
  H5Lregister
  H5Lunregister
- Link Creation Order
  Tracking, indexing, and iterating over links in groups by creation order

  Links in a group can now be explicitly tracked and definitively indexed by the order in which they are created, enabling systematic iteration and lookup of links by creation order. This complements the already-existing alphanumeric-by-name capability.
  H5Pset_link_creation_order
  H5Pget_link_creation_order
  H5Literate
  H5Lvisit
- Dedicated Link Interface
  A Link API (H5L) for directly managing links

  New link APIs enables greater flexibility in the creation and management of links in an HDF5 file. The H5L routines allow links to be managed and manipulated more like objects in the HDF5 data model and provide detailed control of linking behavior.
  H5L: Link interface

## Attribute and Metadata Enhancements

- Enhanced Attribute Handling
  Faster access and more compact storage

  The Attribute interface (H5A) includes several new functions for attribute management. When large numbers of attributes are attached to a single object, new functionality enables faster access and allows those attributes to be stored in much less space in the file.
  For new attribute management functions:
  H5A API

  To configure the attribute storage format:
  H5Pset_attr_phase_change
  H5Pget_attr_phase_change
- Creation Order in Attributes
  Attributes can now be tracked and indexed on the order in which they are created, enabling iteration and lookup of attributes by creation order as well as alphanumeric order by name.
  H5Pset_attr_creation_order
  H5Pget_attr_creation_order
- Shared Object Header Messages (SOHM)
  To conserve space in an HDF5 file, large header messages that are used repeatedly in the file can be designated as *shared*.

  A shared object header message (SOHM) is written only once in a file then a pointer is inserted instead of the message itself on each object to which the header message would otherwise be attached. This can be particularly valuable when, for instance, an identical attribute is applied to tens of thousands of objects. (Note that there is will be no advantage if the attribute itself is smaller than the pointer

would be.)

This feature is implemented via a suite of functions that set up SOHM tracking and indexing and manage the thresholds for switching between shared and non-shared messages:
H5Pset_shared_mesg_nindexes
H5Pget_shared_mesg_nindexes
H5Pset_shared_mesg_index
H5Pget_shared_mesg_index
H5Pset_shared_mesg_phase_change
H5Pget_shared_mesg_phase_change

- UTF-8 Unicode Encoding
  UTF-8 Unicode encoding is supported for strings in datasets, the names of links, and the names of attributes.

  UTF-8 encoding is managed with H5Pset_char_encoding and H5Pget_char_encoding.

  See "UTF-8 Character Encoding in HDF5" and "Character Encoding for Links in HDF5 Files" (PDF) for further information.
- Metadata Caching
  Metadata caching enhancements boost performance with certain types of files and enable configurable metadata cache management and monitoring.

  A suite of functions is provided to set and review the metadata cache configurations, to review and reset hit rate statistics, and to retrieve the current cache size:
  H5Fget_mdc_config
  H5Fget_mdc_hit_rate
  H5Fget_mdc_size
  H5Freset_mdc_hit_rate_stats
  H5Fset_mdc_config
  H5Pset_mdc_config
  H5Pget_mdc_config

  See "Metadata Caching in HDF5" in the —*HDF5 User's Guide* for further information.

## Improved Object Handling

- Create Intermediate Groups
  Rather than having to step through a hierarchy creating groups one at a time, intermediate groups that do not yet exist can now be created when creating or copying an object in a file.

  The creation of missing groups is managed with H5Pset_create_intermediate_group and H5Pget_create_intermediate_group.

  See —*Creating Missing Groups* (PDF) for further information.
- Object Copying
  Copying an HDF5 object to a new location within a file or in a different file

  With this feature, an object in an HDF5 file can easily be copied to a new location within the current file or to a specified location in another HDF5 file. This is accomplised at a low-level in the HDF5 file, allowing entire group hierarchies to be copied quickly and compressed datasets to be copied without going through a decompression/compression cycle.

  A suite of functions is provided to manage copy properties and o perform the copying operation:
  H5Ocopy
  H5Gcreate_anon
  H5Pset_copy_object
  H5Pset_create_intermediate_group

  A command-line tool, h5copy is also provided to enable copying objects without having to create an application.
- Improved Object Information Retrieval
  Three new functions have been added to enhance the object information that can be retrieved.
  H5Lget_info retrieves information regarding a link.
  H5Oget_info retrieves information regarding an object.
  H5Gget_info retrieves information regarding a group.

  In each case, the function returns object information in a customized struct. For example, H5Lget_info returns the link type while H5Gget_info returns the number of links in the group.
- Anonymous Object Creation
  Anonymous object creation enables the creation and management of objects in a file independently of the links that integrate those objects into the file structure.
  H5Dcreate_anon
  H5Gcreate_anon
  H5Tcommit_anon

  The above routines are used in conjunction with the Link and Object interfaces discussed elsewhere (H5L and H5O, repectively).
  H5L: Link interface
  H5O: Object interface

- Dedicated Object Interface
  An Object API (H5O) for managing general objects

  A new object API enables greater flexibility in the creation and linking of objects in an HDF5 file.
  H5O:  Object interface

## Datatype Features

- User-defined Datatype Conversion Callback Functions
  - User-defined Datatype Conversion Callback Functions: Revised Datatype Conversion Exception Handling – It is now possible for an application to have greater control over exceptional circumstances (range errors, etc.) during datatype conversion.

  See "Revising Numeric Overflows in HDF5" and "Data Conversion Of Arithmetic Data Types."
- Integer-to-Floating-point Conversion Support
  - Integer-to-Floating-point Conversion Support – It is now possible for the HDF5 library to convert between integer and floating-point datatypes.

  See H5Tconvert in the *—HDF5 Reference Manual*.
- Datatype and Dataspace Serial Conversion
  - Datatype and Dataspace Serial Conversion – Routines have been implemented to serialize/deserialize HDF5 datatypes and dataspaces.  These routines allow datatype and dataspace information to be transmitted between processes or stored in non-HDF5 files.

  See "Encode and Decode HDF5 Objects," and the function entries in the *—HDF5 Reference Manual* for H5Tencode, H5Tdecode, H5Sencode, and H5Sdecode
- Two-way Conversion Between Datatype and Text Description of Datatype
  - Two-way Conversion Between Datatype and Text Description of Datatype – This feature enables the creation of a datatype from a text definition of that datatype and the creation of a formal text definition from a datatype.  The text definition is in DDL format; DDL definitions of HDF5 datatypes can be found in the "DDL in BNF for HDF5."

  H5LTtext_to_dtype creates an HDF5 data type based on the text description and returns the data type identifier.  Given a datatype identifier, H5LTdtype_to_text creates a DDL description of the datatype.

  Also see "Conversion Between Text and Datatype."

## Enhancements in the I/O Pipeline

- New Compression Filters
  - New Compression Filters – These new I/O filters allow better compression of certain types of data:

  o  N-Bit Filter – This filter compresses data which uses N-bit datatypes. See H5Pset_nbit in the *—HDF5 Reference Manual* and the section "Using Filters / N-bit" in the "Datasets" chapter of the *—HDF5 User's Guide*.

  o  Scale+Offset Filter – This filter compresses scalar (integer and floating-point) datatypes which stay within a range. See H5Pset_scaleoffset in the *—HDF5 Reference Manual* and the section "Using Filters / Scale-Offset" in the "Datasets" chapter of the *—HDF5 User's Guide*.
- Collective Chunk I/O in Parallel
  - Collective Chunk I/O in Parallel – The library now attempts to use MPI collective mode when performing I/O on chunked datasets when using the parallel I/O file driver.
- Arithmetic Data Transform on I/O
  - Arithmetic Data Transform on I/O – This feature allows arithmetic operations (add/subtract/multiply/divide) to be performed on data elements as they are being written to/read from a file. See H5Pset_data_transform in the *—HDF5 Reference Manual*.

## High-level Interfaces and Fortran and C++ Wrappers

- C++ Wrapper Improvements
  - C++ API Wrapper Improvements – Several improvements were made to the C++ build infrastructure, as well as adding support for previously missing and new API routines.
- Fortran Wrapper Improvements
  - FORTRAN API Wrapper Improvements – Several improvements were made to the FORTRAN build infrastructure, as well as adding support for previously missing and new API routines.

- New Packet Table and Dimension Scale High-Level APIs

    - New Packet Table and Dimension Scale High-Level APIs have been added to the high-level C interfaces.

  The Packet Table API (H5PT) is designed to allow variable-length records to be added to tables easily.

  The Dimension Scale API (H5DS) allows dimension scales to be created in HDF5 and attached to HDF5 datasets. Also see "HDF5 Dimension Scale Specification and Design Notes" (PDF).
- High-Level Fortran APIs

    - High-Level Fortran APIs – Fortran APIs have been added for the following High-Level HDF5 APIs:
      H5Lite (H5LT)
      H5Image (H5IM)
      H5Table (H5TB)

## New and Improved Tools

- h5mkgrp
- h5stat
- h5copy
- h5dump

    - Tool Improvements – Three new tools have been added, and existing tools were enhanced:

  o h5mkgrp is a new command-line tool that creates a new group in an HDF5 file.

  o h5stat (PDF) enables the analysis of an HDF5 file in various ways to determine useful statistics regarding the objects in the file, such as the numbers of objects per group, the sizes of datasets, the amount of free space in the file, etc.

  o h5copy makes a complete copy of an object in an HDF5 file as a new object in that HDF5 file or as a new object in a different HDF5 file.

  o Improved speed of h5dump – Performance improvements have been made to h5dump to speed it up when dealing with files that have large numbers of objects.

## Miscellaneous Features

- NULL Dataspace

    - "NULL" Dataspace – A new type of dataspace, which allows datasets and attributes without any elements to be described.

  See H5Screate in the —*HDF5 Reference Manual*.
- Extendible Identifier API

    - Extendible Identifier API – A new set of identifier management routines has been added, which allow an application to use the HDF5 identifier-to-object mapping routines.

  See the H5I APIs in the —*HDF5 Reference Manual* and "Allowing Users to Access HDF5's ID System."
- Enhanced Error Handling

    - Enhanced Error Handling – A new set of error API routines has been added, which allow an application to integrate its error reporting with the HDF5 library error stack.

  In the —*HDF5 Reference Manual*, see the error stack APIs. Also see the supporting document "Unified Error Reporting for HDF5 and Client Libraries."
- Better UNIX/Linux Portability

    - Better UNIX/Linux Portability – This release now uses the latest GNU "auto" tools (autoconf, automake, and libtool) to provide much better portability between many machine and OS configurations. Building the HDF5 distribution can now be performed in parallel (with the gmake "–j" flag), speeding up the process of building, testing and installing the HDF5 distribution. Many other improvements have gone into the build infrastructure as well.