HDF5 VFD Plugins

October 12, 2021

Jordan Henderson
Dana Robinson
The HDF Group

Quick Recap

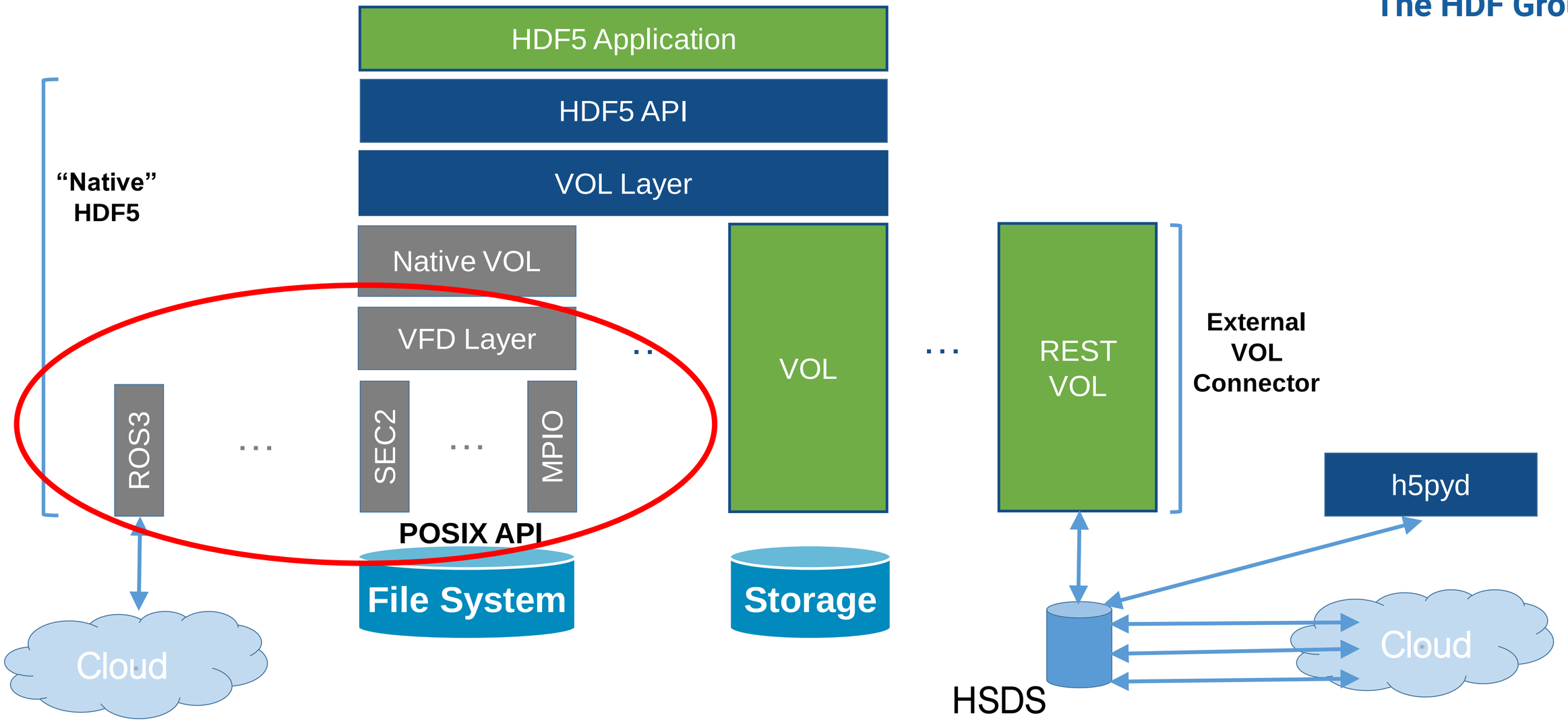
The HDF Group

- HDF5 Virtual File Drivers (VFDs) allow users to define mapping between HDF5 address space and underlying storage
  - Sec2 VFD – Uses POSIX I/O on a single file
  - Core VFD – I/O directly on memory
  - Multi/Family VFDs – Data/metadata written to separate files in a defined way
  - Etc.

- Set on an HDF5 File Access Property List by generic `H5Pset_driver` call, or by specialized driver-specific call

HDF5 1.13 Library Architecture

The HDF Group



VFD Plugin Feature – When and Why

The HDF Group

- Will be available in HDF5 1.13.0 release
- Connects the HDF5 native file format to a wider variety of data sources
- Integrates well with HDF5 by leveraging existing plugin scheme
  - Can easily try out VFD plugins with HDF5's tools, tests, etc.
- Welcomes new contributions to the HDF5 ecosystem
  - Plugins more easily hosted and built separately from HDF5
- More adaptable to a changing data landscape

Virtual File Driver (VFD) Changes

VFD 'values'

The HDF Group

- New VFD `H5FD_class_t` struct field

- A unique identifier for the VFD plugin
  - NOT library-managed `hid_t` IDs (hence 'value' to avoid confusion)
  - Implemented as an integer (typedef'd to `H5FD_class_value_t`)

- Ranges reserved for specific purposes
  - 0 - 255 for THG
  - 256 - 511 for testing (will never be used by internal or external plugins)
  - 512 - 65535 for new VFD plugins

- Register new VFD plugin values with THG (help@hdfgroup.org)

VFD Configuration Strings

The HDF Group

- Some VFDs will require a way to pass arbitrary configuration data to them
- This can be done by passing a string that will be interpreted by the VFD
  - THG imposes NO structure on this string
  - Use whatever you like - JSON, YAML, XML, roll-your-own, etc.
- Used in new `H5Pset_driver_*()` calls and HDF5 command-line tools (h5dump, etc.)
- VFD authors can call new `H5Pget_driver_config_str()` routine on a passed-in FAPL to get the string for processing

New API Calls

The HDF Group

```
herr_t
H5Pset_driver_by_name(hid_t fapl_id, const char *driver_name,
                      const char *driver_config);

herr_t
H5Pset_driver_by_value(hid_t fapl_id, H5FD_class_value_t driver_value,
                      const char *driver_config);

ssize_t
H5Pget_driver_config_str(hid_t fapl_id, char *config_buf, size_t buf_size);
```

New API Calls (continued)

The HDF Group

```
htri_t
H5FDis_driver_registered_by_name(const char *driver_name);

htri_t
H5FDis_driver_registered_by_value(H5FD_class_value_t driver_value);
```

VFD Plugins

Authoring a VFD

The HDF Group

The HDF5 library has two "example" VFDs:

- stdio - a single-file VFD
- multi - a multi-file VFD

Both use standard C and no internal HDF5 API calls so they can be copied to serve as templates for your own VFDs

No template project/repository at this time, but you could use the build files for the HDF5 GDS VFD or HDF5 Template VOL as a start.

Converting a VFD into a VFD plugin

The HDF Group

- Once VFD is written, simply add two functions for HDF5 to locate:

```
H5PL_type_t
H5PLget_plugin_type(void) {
    return H5PL_TYPE_VFD;
}


const void *
H5PLget_plugin_info(void) {
    return &H5FD_gds_g; /* Adjust accordingly */
}
```

- `H5PLget_plugin_type` always returns `H5PL_TYPE_VFD` to instruct HDF5 that this is a VFD plugin
- `H5PLget_plugin_info` should return a pointer to the VFD's `H5FD_class_t` structure (`H5FD_gds_g` is used as an example here; adjust accordingly for your VFD)

Loading a VFD plugin

The HDF Group

- Like HDF5 filter plugins and VOL connector plugins
  - HDF5 searches for plugins in default plugin paths
    - POSIX: `/usr/local/hdf5/lib/plugin`
    - Windows: `%ALLUSERSPROFILE%/hdf5/lib/plugin`
  - Ensure that `HDF5_PLUGIN_PATH` environment variable is set if plugin doesn't reside in one of these paths
- Call `H5Pset_driver_by_name/_by_value` on a FAPL to load a VFD by a given VFD name or VFD ID value and set it on FAPL

```
fapl_id = H5Pcreate(H5P_FILE_ACCESS);
H5Pset_driver_by_name(fapl_id, "my_vfd", my_vfd_config_str);
```

- Use the FAPL for calls to `H5Fcreate/H5Fopen`

OR...

Loading a VFD plugin (continued)

The HDF Group

- Set `HDF5_DRIVER` environment variable to a VFD name string
  - Replaces default file driver on FAPL
  - HDF5 application doesn't need to be modified; can quickly switch between file drivers

- `HDF5_DRIVER_CONFIG` environment variable can be used to pass configuration string to VFD plugin, rather than passing through `H5Pset_driver_*` API call

HDF5 Tools Support

The HDF Group

```
--vfd-value     Value (ID) of the VFD to use for
                opening the HDF5 file specified

--vfd-name      Name of the VFD to use for opening
                the HDF5 file specified

--vfd-info      VFD-specific configuration string
                to pass to the VFD
```

- Same way you specify a VOL connector in HDF5 1.12

VFD Testing

The HDF Group

- Ongoing work to modify HDF5's test suite for use with VFD plugins
  - Separation of library-VFD-specific VFD tests from generic VFD tests needed

- Simply set `HDF5_DRIVER` and `HDF5_DRIVER_CONFIG` and run 'make check' or CTest

More Information and Documentation

The HDF Group

- HDF5 VFD Plugins RFC
  - https://github.com/HDFGroup/hdf5doc/blob/master/RFCs/HDF5_Library/VFL_DriverPlugins/RFC__A_Plugin_Interface_for_HDF5_Virtual_File_Drivers.pdf
- Support portal page for registered plugins
  - https://portal.hdfgroup.org/display/support/Contributions
  - In process of creating a page for VFD plugins
- Support portal page for dynamic plugins of all types
  - https://portal.hdfgroup.org/display/HDF5/Dynamic+Plugins+in+HDF5
- Example VFD Plugin - HDF5 Nvidia GPUDirect Storage VFD
  - https://github.com/hpc-io/vfd-gds

Contact Info

Jordan Henderson

Jhenderson@hdfgroup.org

THANK YOU!

Questions & Comments?