Revisiting HDF5 SWMR and file versioning features

Elena Pourmal elena.pourmal@lifeboat.llc
Scot Breitenfeld brtnfld@hdfgroup.org

Lifeboat

The HDF Group

HDF5 Prototypes

- Single writer / multiple reader access to HDF5 files (VFD SWMR)
  - Available in the HDF5 repo branches “`feature/vfd_swmr_beta_2`” or “`feature/vfd_swmr`”
- HDF5 file provenance and revision/version control
  - Released in 1.14.0

We need your feedback on both features to finish productization!

Acknowledgement

- VFD SWMR work is supported by the U.S. Department of Energy, Office of Science under award number DE-SC0025856 for Phase I SBIR project "Toward Robust HDF5”

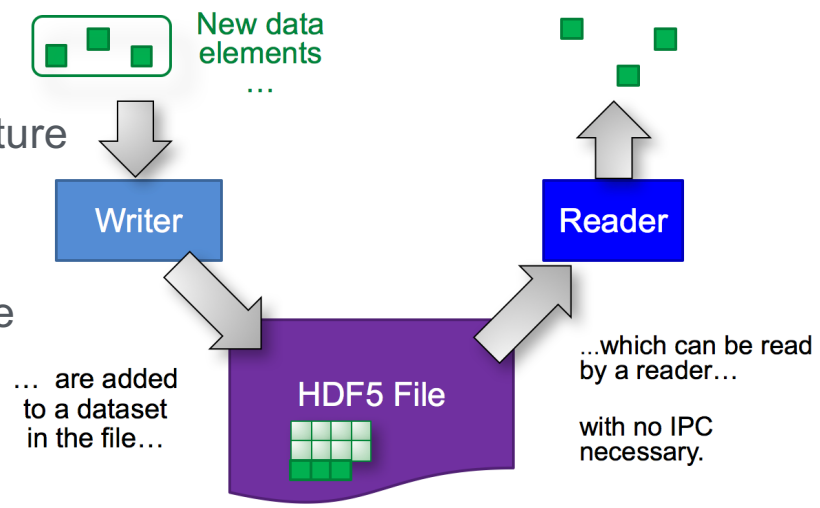
- The HDF Group

VFD SWMR

Full access to HDF5 file under construction

Current Single Writer/Multiple Reader (SWMR)

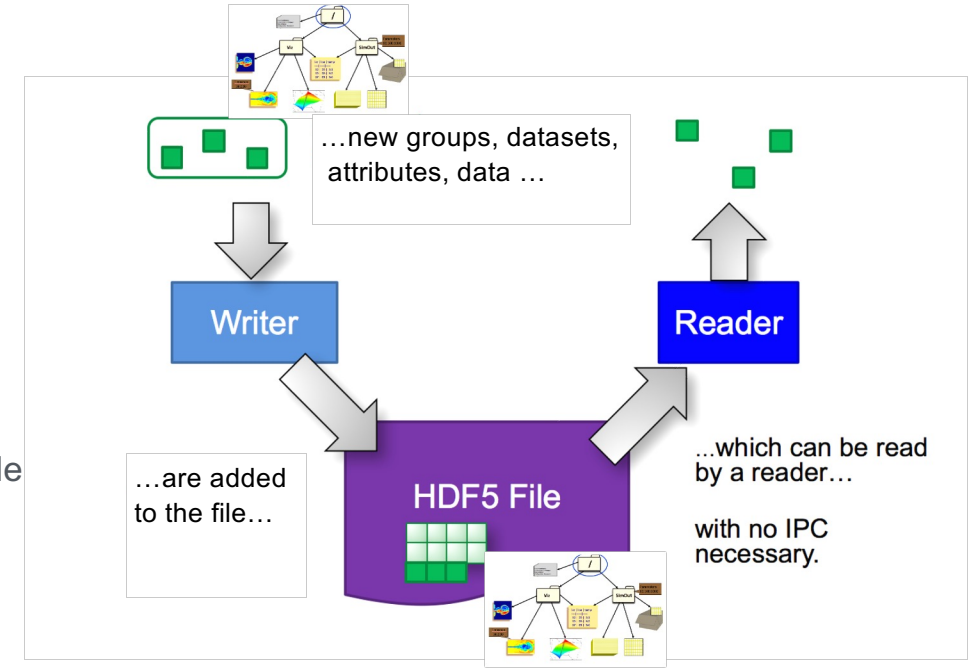
- Introduced in HDF5 1.10.0
- Implementation has many limitations
  - Based on POSIX semantics
    - No support for NSF
  - Doesn't allow modification of the file structure
    - Only dataset's raw data can be modified
    - Groups, datasets, attributes cannot be added
  - New data may not be visible for some time
  - No support for parallel applications
  - Order of writer and reader is critical



Full Single Writer/Multiple Reader (VFD SWMR)

Goals

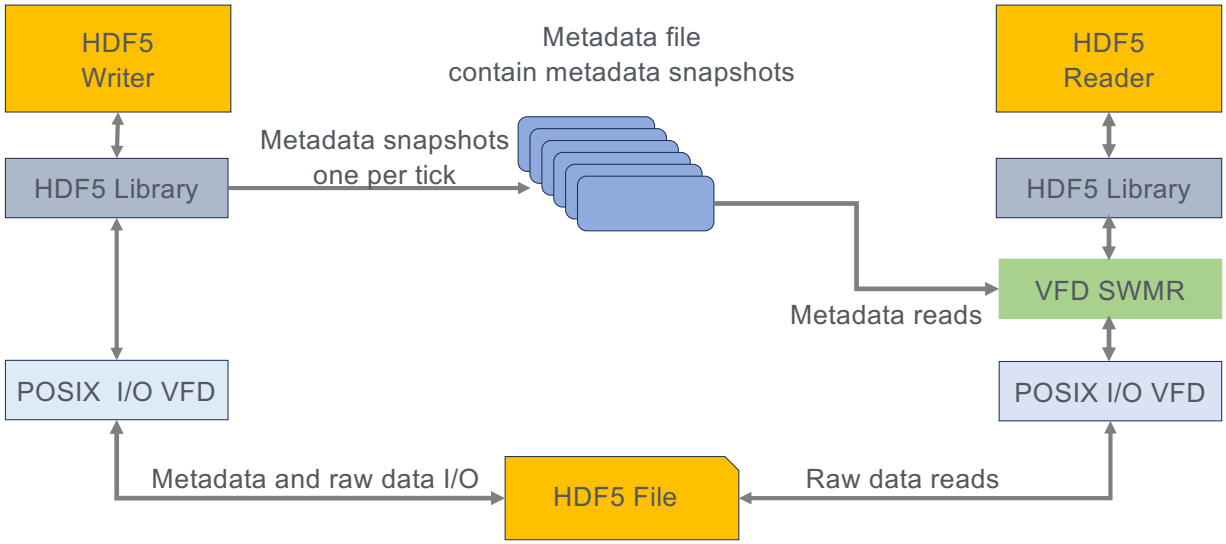
- SWMR feature
  - Allow modification of a file structure
  - Guarantee max time for new data visibility
  - *No POSIX semantics requirements*
    - Support NFS and Object Store
  - Lift requirement for writer/reader ordering
  - Support for VL types
  - Support for parallel applications
  - Create a tool to recover corrupted HDF5 file
- HDF5 Architecture improvement
  - Remove complex flush dependencies



Overview of VFD SWMR

- See “RFC: VFD SWMR” for design details.
- Major implementation concepts and requirements:
  - Tick – describes periods of operations for both writer and reader: if “t” is desired maximum latency (user defined parameter) from write to visibility on the reader side, tick is defined to be “`t/3`”; tick length is measured in tenths of a second.
  - VFD SWMR configuration parameters and HDF5 metadata file :
    - At the end of each tick HDF5 metadata cache is flushed to page buffer that keeps track of modified metadata pages
    - Modified metadata pages are written to storage to a metadata file
    - Written metadata pages are released in metadata file after `max_lag` ticks
    - Metadata file is shared between writer and readers
    - Readers satisfy metadata I/O requests from the metadata file
  - See requirements for file creation and access properties for writer and readers on the next slides.

VFD SWMR Architecture



VFD SWMR Programming Model - Writer

- Set creation property to use paged allocation:
  `H5Pset_file_space_strategy(fcpl, H5F_FSPACE_STRATEGY_PAGE, false, 1)`
- Set access properties to:
  - use latest (1.14.*) file format with `H5Pset_libver_bounds`
  - enable page buffering with `H5Pset_buffer_size`
  - set VFD SWMR configuration properties in the `H5F_vfd_swmr_config_t` structure; see `H5Fpublic.h` for description

```
config.tick_len = 4;
config.max_lag = 7;
config.writer = true;
config.md_file_path = “./”;
config.md_file_name = “my_md_file”;
H5Pset_vfd_swmr_config (fapl, &config);
```

- Create file with specified creation and access properties

VFD SWMR Programming Model - Reader

- Set access properties *with the same parameters* as for writer:
  - Use latest (1.14.*) file format with `H5Pset_libver_bounds`
  - Enable page buffering with `H5Pset_buffer_size`
  - Set VFD SWMR configuration properties

```
config.tick_len = 4;
config.max_lag = 7;
config.writer = false;
config.md_file_path = “./”;
config.md_file_name = “my_md_file”;
H5Pset_vfd_swmr_config (fapl, &config);
```

  - Open the file with specified access properties
- Reader needs to check if metadata file is available and then start read operations

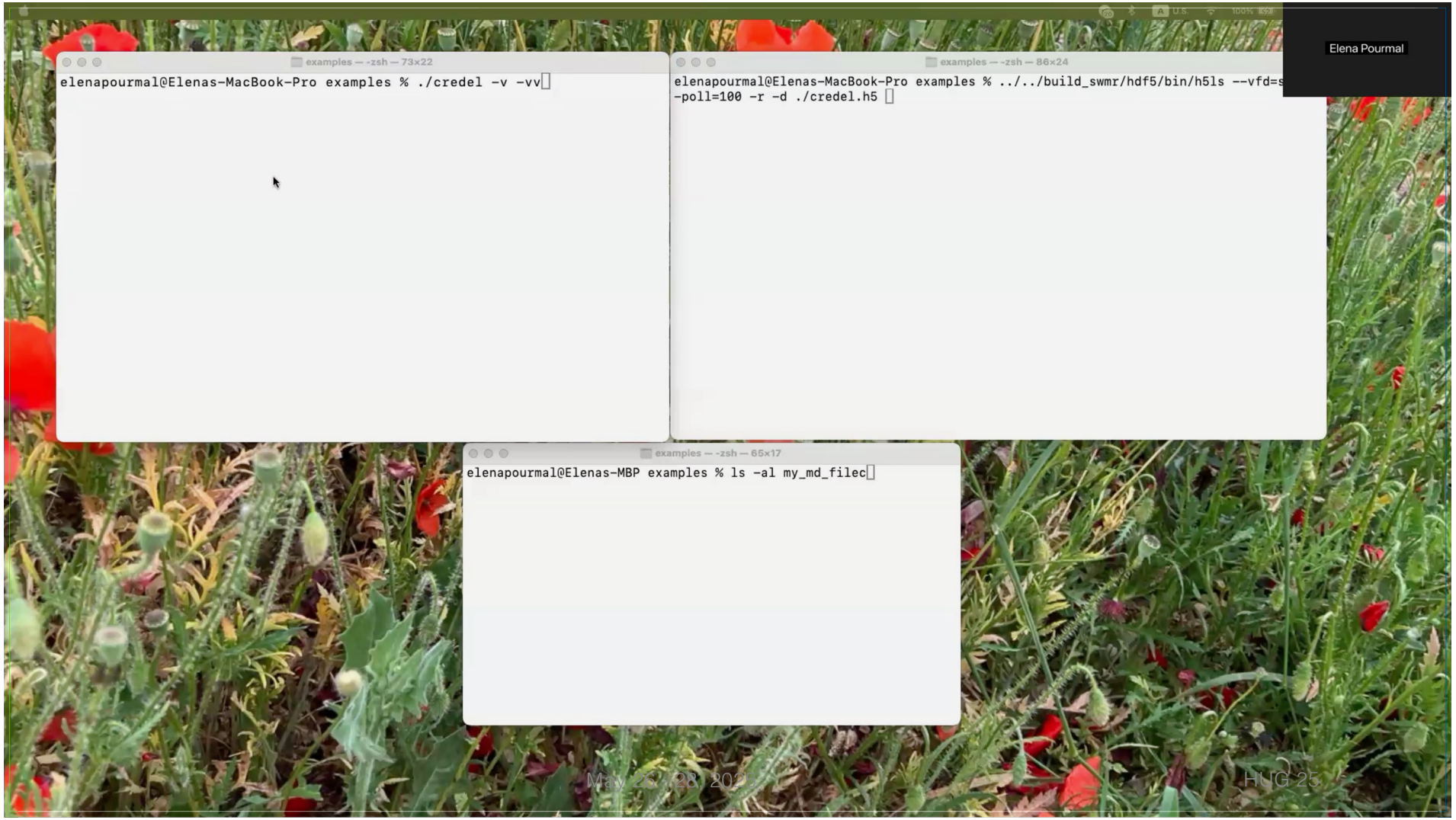
VFD SWMR Demo

- Code is available from the HDF5 GitHub repo https://github.com/HDFGroup/hdf5.git, "`feature/vfd_swmr_beta_2`" branch
  - See `doc` directory for VFD SWMR documentation
    - Check `vfd-swmr-user-guide.md` for building instructions and for the extensible datasets example; it compiles and runs as is.
- Demo
  - See README in the hdf5/examples directory
  - The files `examples/credel.c` and `tools/lib/h5tools.c` have been modified for the demo, but the updates have not been checked in yet.

VFD SWMR Demo

`credel.c` VFD SWMR settings

```
H5Pget_vfd_swmr_config(fapl, &config);
config.version  = H5F__CURR_VFD_SWMR_CONFIG_VERSION;
config.tick_len = 4;
config.max_lag  = 5;
config.flush_raw_data = true;
config.writer         = true;
config.md_pages_reserved      = 128;
config.pb_expansion_threshold = 50;
config.maintain_metadata_file = true;
strlcpy(config.md_file_path, "./", sizeof(config.md_file_path));
strlcpy(config.md_file_name, "my_md_file", sizeof(config.md_file_path));
```


examples — -zsh — 73×22
elenapourmal@Elenas-MacBook-Pro examples % ./credel -v -vv
examples — -zsh — 86×24
elenapourmal@Elenas-MacBook-Pro examples % ../../build_swmr/hdf5/bin/h5ls --vfd=s
-poll=100 -r -d ./credel.h5
examples — -zsh — 65×17
elenapourmal@Elenas-MBP examples % ls -al my_md_filec
Elena Pourmal
May 23 - 28, 2025
HUG 25

Onion VFD

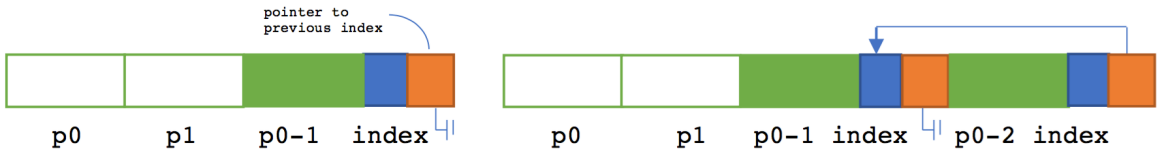
HDF5 file provenance and revision/version control

Why HDF5 File Provenance?

- HDF5 data provenance is a major concern in cases when
  - Original data must be preserved
  - Changes are tracked and attributed
- HDF5 “undo” capability
  - The feature was requested for HDFView (reads and writes data)
- AI/ML applications with HDF5 used for:
  - Model checkpointing and serialization (e.g., Keras, TensorFlow, PyTorch)
  - Storing large datasets for training
  - Intermediate data storage for preprocessed data, model state and optimizer states to resume training from specific point
- RFC: Onion VFD, May 9, 2021
  https://support.hdfgroup.org/releases/hdf5/documentation/rfc

HDF5 File Provenance Storage Options (conceptual view)

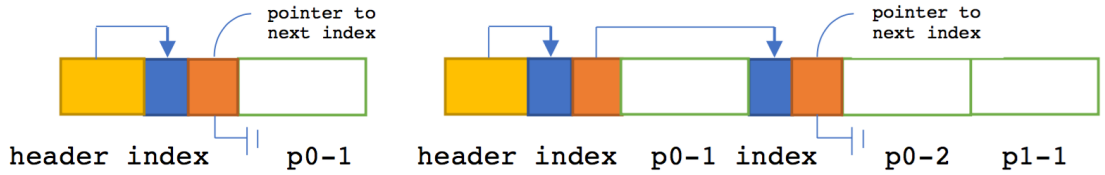
(1)



HDF5 file with version 1 info

HDF5 file with version 1 & 2 info

(2)



File with version 1 information

File with version 1 & 2 information

Storage options for change history:

1. Tracking in the same file
2. Additional file with tracked pages
3. Additional files for each version

(3)

Additional file for Version 1

Additional file for Version 2

HDF5 Onion VFD Implemented Features

- Changes are tracked per file open/close
  - No “undo” capability
- Modification history is stored in a separate file (option 2) (*filename*.onion)
- Amended data is stored as fixed-size verbatim pages (size is power of 2) (vs. amended runs, i.e., only modified data)
- Metadata (header, index) can be aligned on page boundary
- Only the latest version can be modified, there is no branching.
- I/O on the onion file is performed with the same I/O driver as on the HDF5 file.
- There is no support for parallel applications.
- Each revision can be annotated.
- `H5FDctl` function can be used to find onion info, revision history, to set or update revision annotation.

Onion VFD APIs

```
herr_t H5Pset(get)_fapl_onion (hid_t fapl_id, H5FD_onion_fapl_info_t  *fa)
herr_t H5FDonion_get_revision_count (const char *filename, hid_t fapl_id,
                                     uint64_t *revision_count);

typedef struct H5FD_onion_fapl_info_t {
    uint8_t                              version;
    hid_t                                backing_fapl_id;
    uint32_t                             page_size;
    H5FD_onion_target_file_constant_t    store_target;
    uint64_t                             revision_num;
    uint8_t                              force_write_open;
    uint8_t                              creation_flags;
    char                                 comment[256];
} H5FD_onion_fapl_info_t;
```

Enhancements for productized version of Onion VFD

- Implement support for
  - Different storage options for history data
  - Different I/O drivers for history data when using options 2 and 3
  - Parallel HDF5 applications
- Other considerations:
  - Storage size
    - Onion file may become huge when working with HDF5 files
  - Branching
    - Looks like a useful feature for AI applications
  - Performance
  - Tools support
  - User and developer documentation
- Tell us about your needs and requirements!

Thank you!

Questions?