HDFCloud

HDF5 in the Cloud
(and NetCDF4…)

The HDF Group

John Readey
The HDF Group
jreadey@hdfgroup.org

My Background

The HDF Group

Sr. Architect at The HDF Group

Started in 2014

Have been exploring remote interfaces to HDF

Previously: Dev Manager at Amazon/AWS

amazon web services

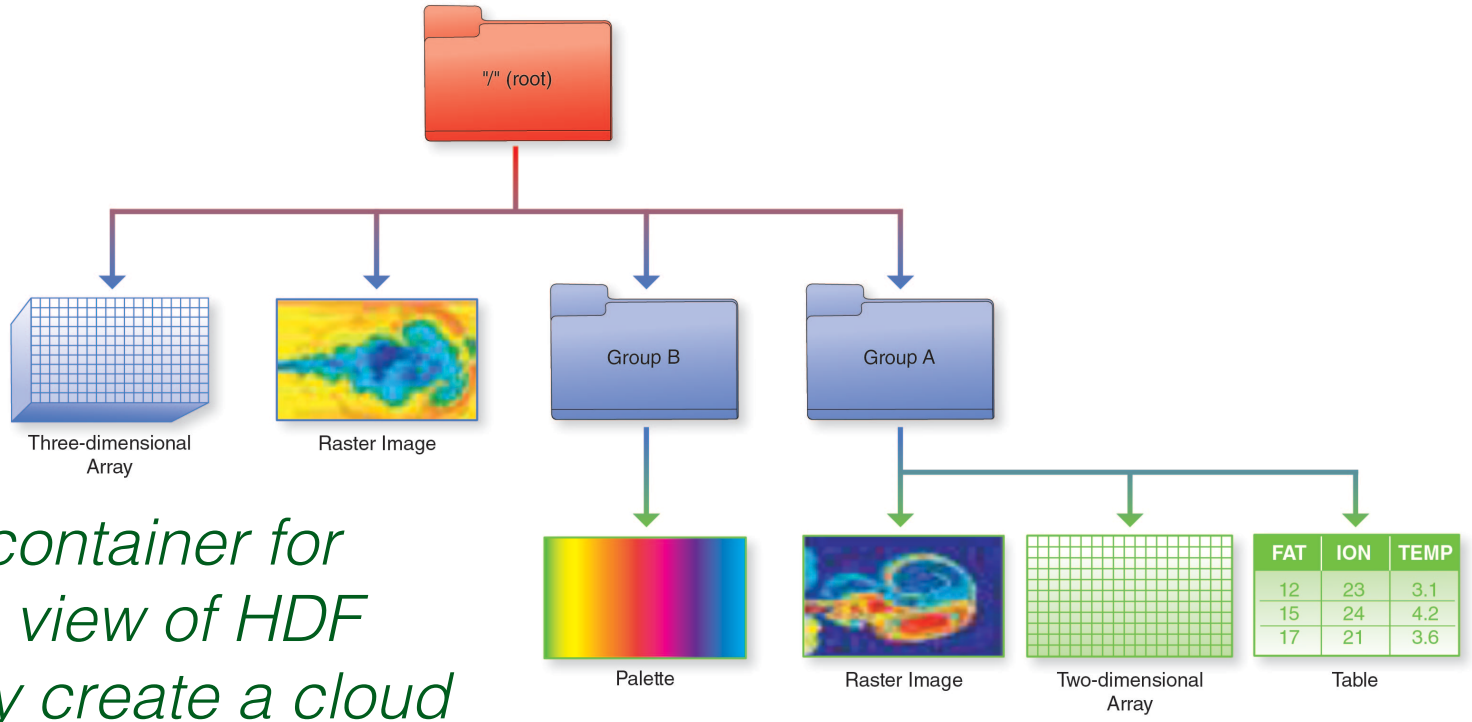
More previously: Used HDF5 while a developer at Intel

intel inside

What is HDF5?

Depends on your point of view:

- a C-API
- a File Format
- a data model



The File format is just a container for The data. Dropping this view of HDF allows us to more flexibly create a cloud version of HDF.

Why HDF in the Cloud

The HDF Group

- It can provide a cost-effective infrastructure
  - Pay for what you use vs pay for what you may need
  - Lower overhead: no hardware setup/network configuration, etc.
- Potentially can benefit from cloud-based technologies:
  - Elastic compute – scale compute resources dynamically
  - Object based storage – low cost/built in redundancy
- Community platform
  - Enables interested users to bring their applications to the data
  - Share data among many users

Cost Factors

- **Most public clouds bill per usage**
- **For HDF in the cloud, there are three big cost drivers:**
  - **Storage – What storage system will be used? (see next slide)**
  - **Compute – Elastic compute on demand better than fixed cost**
    - Scale compute to usage not size of data
  - **Egress charges**
    - Ingress is free but getting data out will cost you ($0.09/GB)
    - Enabling users to get just the data they need will tend to lower egress charges

Introducing Highly Scalable Data Service (HSDS)

- RESTful interface to HDF5 using object storage
- Storage using AWS S3 (portable to most other object storage systems)
  - Built in redundancy
  - Cost effective
  - Scalable throughput
- Runs as a cluster of Docker containers
  - Elastically scale compute with usage
- Feature compatible with HDF5 library
- Implemented in Python using asyncio
  - Task oriented parallelism

HSDS Features

- Clients can interact with service using REST API
- SDKs provide language specific interface (e.g. h5pyd for Python)
- Can read/write just the data they need (as opposed to transferring entire files)
- No limit to the amount of data that can be stored by the service
- Multiple clients can read/write to same data source
- Scalable performance:
  - Can cache recently accessed data in RAM
  - Can parallelize requests across multiple nodes
  - More nodes -> better performance

Object Storage Challenges for HDF

The HDF Group

- Not POSIX!
- High latency (>0.1s) per request
- Not write/read consistent
- High throughput needs some tricks
  - (use many async requests)
- Request charges can add up (public cloud)

For HDF5, using the HDF5 library directly on an object storage system is a non-starter. Will need an alternative solution…

HSDS S3 Schema

Big Idea: Map individual HDF5 objects (datasets, groups, chunks) as Object Storage Objects

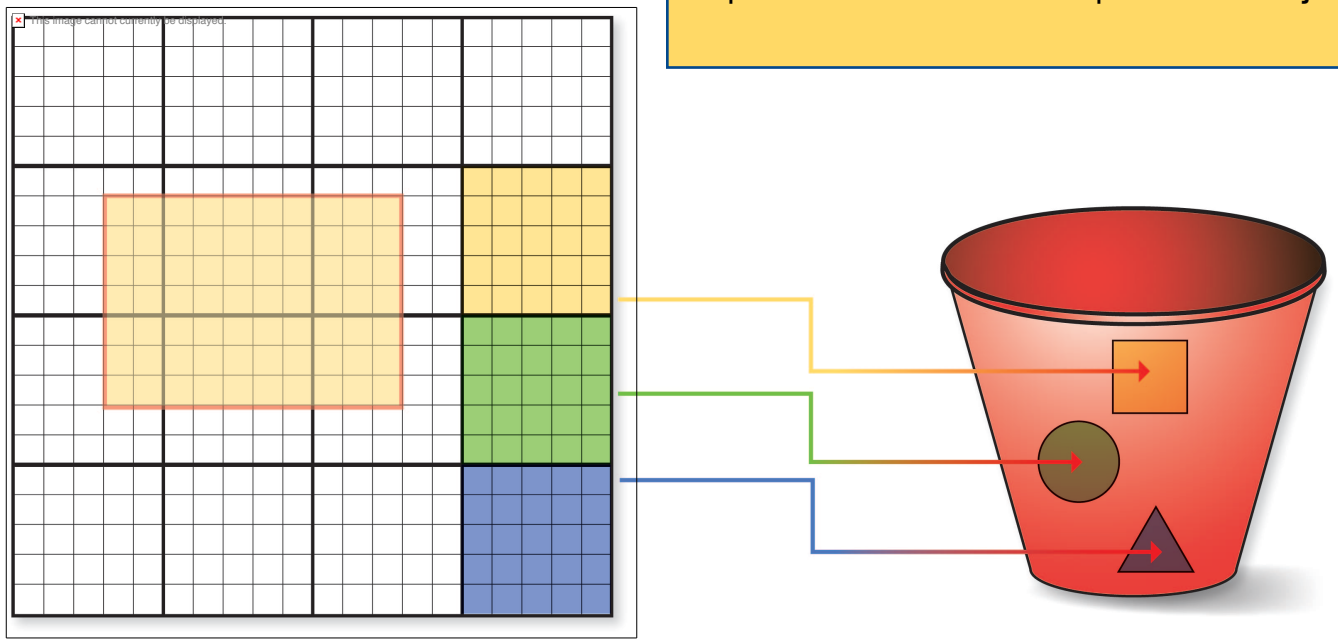
How to store HDF5 content in S3?

- Limit maximum storage object size
- Support parallelism for read/write
- Only data that is modified needs to be updated
- (Potentially) Multiple clients can be reading/updating the same “file”

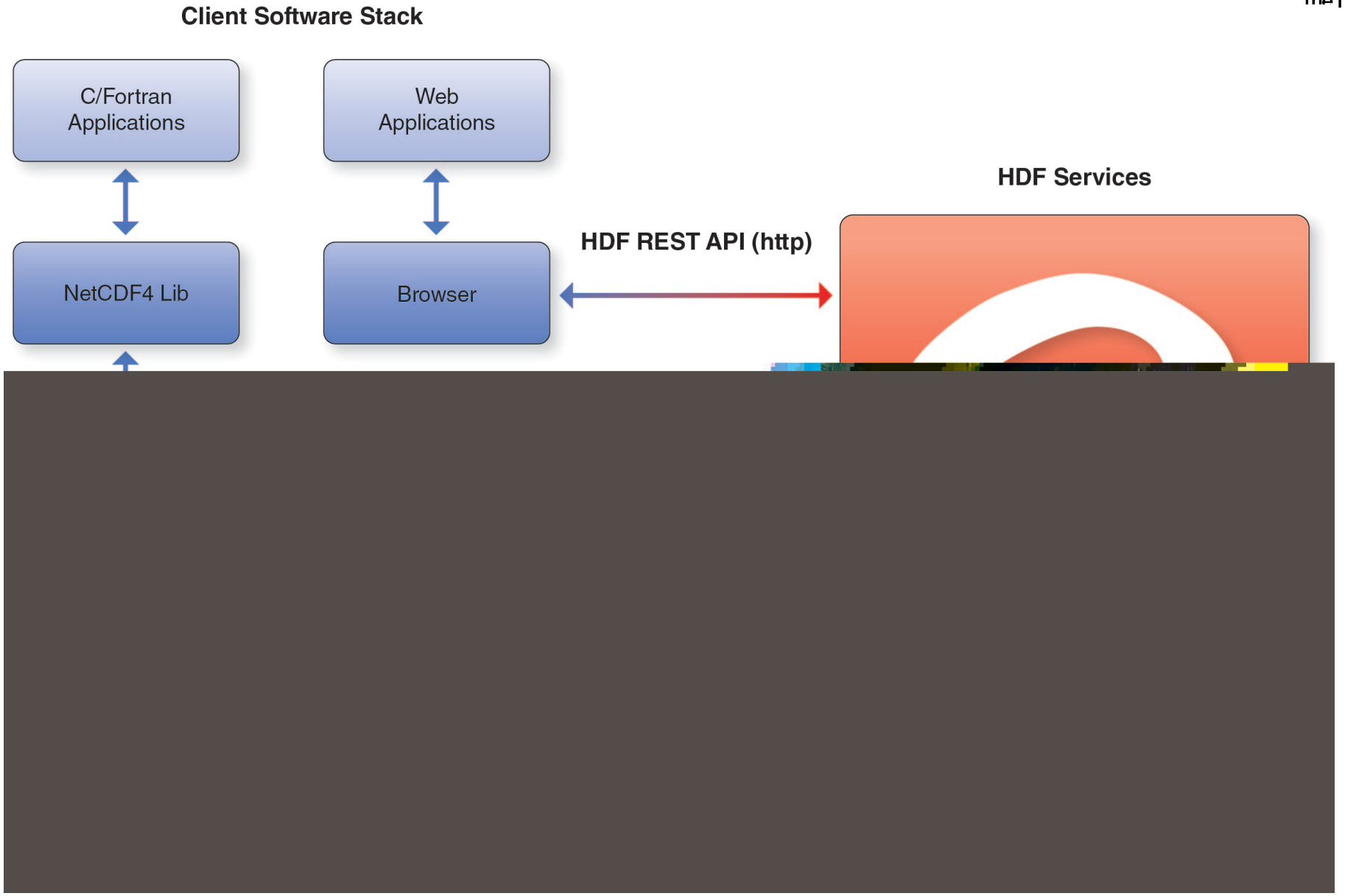
Each chunk (heavy outlines) get persisted as a separate object

Legend:

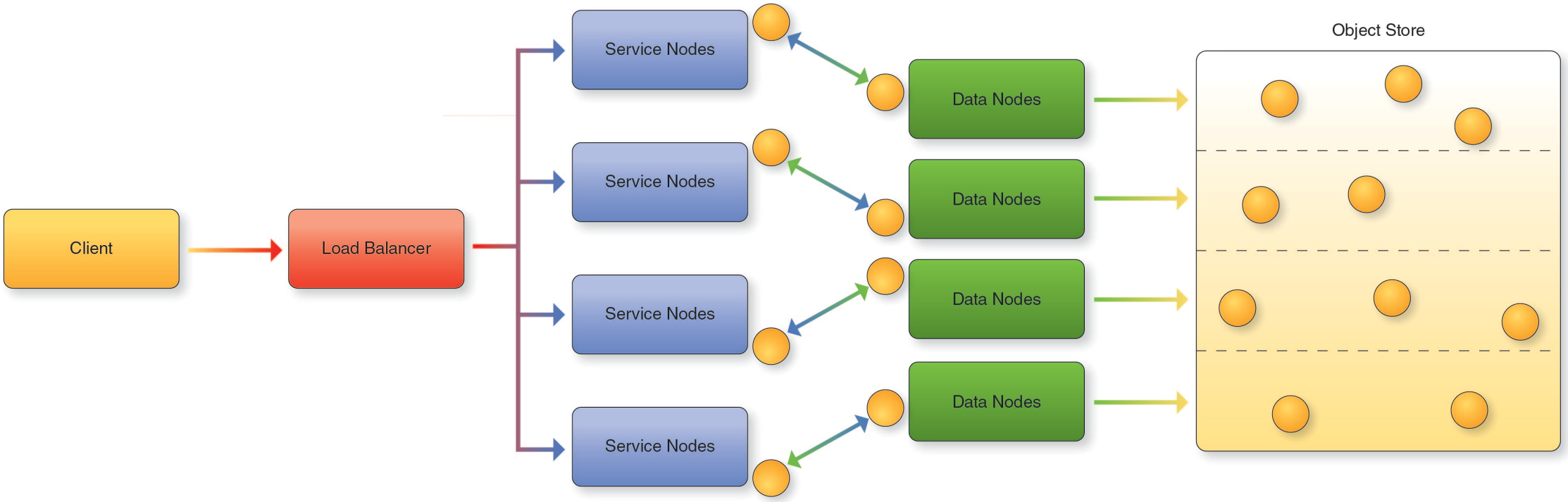
- *Dataset is partitioned into chunks*
- *Each chunk stored as an S3 object*
- *Dataset meta data (type, shape, attributes, etc.) stored in a separate object (as JSON text)*



Client/Server Architecture



Architecture for HSDS



Legend:

- Client: Any user of the service
- Load balancer – distributes requests to Service nodes
- Service Nodes – processes requests from clients (with help from Data Nodes)
- Data Nodes – responsible for partition of Object Store
- Object Store: Base storage service (e.g. AWS S3)

H5pyd – Python client for HDF Server

- H5py is a popular Python package that provide a Pythonic interface to the HDF5 library
- H5pyd (for h5py distributed) provides a h5py compatible h5py for accessing the server
- Pure Python – uses requests package to make http calls to server
- Compatible with h5serv (the reference implementation of the HDF REST API)
- Include several extensions to h5py:
  - List content in folders
  - Get/Set ACLs (access control list)
  - Pytables-like query interface

HDF REST VOL

- The HDF5 VOL architecture is a plugin layer for HDF5
- Public API stays the same, but different backends can be implemented
- REST VOL substitutes REST API requests for file i/o actions
- C/Fortran applications should be able to run as is
- Still in development – Beta expected this year

HSDS CLI (Command Line Interface)

- Accessing HDF via a service means one can't utilize usual shell commands: ls, rm, chmod, etc.
- Command line tools are a set of simple apps to use instead:
  - hsinfo: display server version, connect info
  - hsls: list content of folder or file
  - hstouch: create folder or file
  - hsdel: delete a file
  - hsload: upload an HDF5 file
  - hsget: download content from server to an HDF5 file
  - hsacl: create/list/update ACLs (Access Control Lists)
- Implemented in Python & uses h5pyd

Future Work

- Work planned for the next year
  - Compression
  - Variable length datatypes
  - NetCDF support
  - Auto Scaling
  - Scalability and performance testing
- Special thanks to NASA who is supporting this work under ACCESS grant 15-0031

To Find out More:

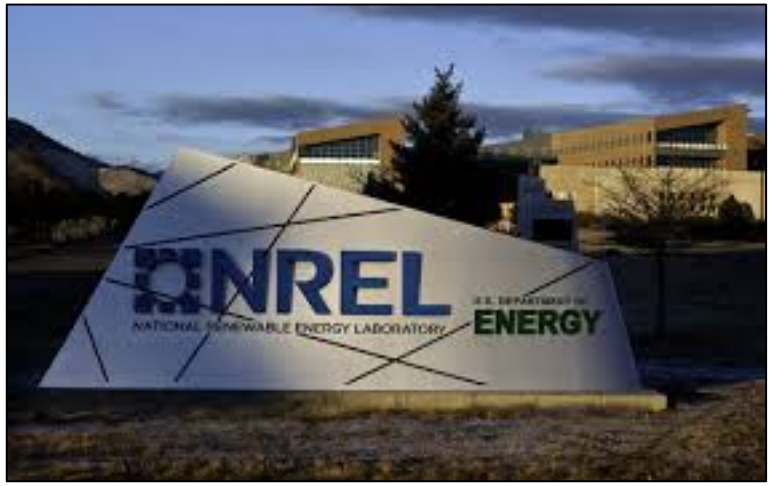
- H5serv: https://github.com/HDFGroup/h5serv
- Documentation: http://h5serv.readthedocs.io/
- H5pyd: https://github.com/HDFGroup/h5pyd
- RESTful HDF5 White Paper: https://www.hdfgroup.org/pubs/papers/RESTful_HDF5.pdf
- Blog articles:
  - https://hdfgroup.org/wp/2015/04/hdf5-for-the-web-hdf-server/
  - https://hdfgroup.org/wp/2015/12/serve-protect-web-security-hdf5/
  - https://www.hdfgroup.org/2017/04/the-gfed-analysis-tool-an-hdf-server-implementation/

HDF5 Community Support

- Documentation - https://support.hdfgroup.org/documentation/
  - Tutorials, FAQs, examples
- HDF-Forum – mailing list and archive
  - Great for specific questions
- Helpdesk Email – help@hdfgroup.org
  - Issues with software and documentation

https://support.hdfgroup.org/services/community_support.html

Demo Time!



NREL (National Renewable Energy Laboratory) is using HSDS to make 7TB of wind simulation data accessible to the public.

Datasets are three-dimensional covering the continental US:

- Time (one slice/hour)
- Lon (~2k resolution)
- Lat (~2k resolution)

Initial data covers one year (8760 slices), but will be soon be extended to 5 years (35 TBs).

Rather than downloading TB's of files, interested users can now use the HSDS client libraries to explore the datasets.

Questions? Comments?

The HDF Group

www.hdfgroup.org

Dave Pearah
CEO
David.Pearah@hdfgroup.org

Dax Rodriguez
Director of Commercial Services and Solutions
Dax.Rodriguez@hdfgroup.org