User-Friendly Data Management for Scientific Computing Users

R. Cheema [1], L. Gerhardt*[1], A. Greiner*[1], D. Hazen[1], R. Lee[1], K. Lozinskiy[1], K. Kallback-Rose[1]

***Abstract*— Wrangling data at a scientific computing center can be a major challenge for users, particularly when quotas may impact their ability to utilize resources. In such an environment, a task as simple as listing space usage for one's files can take hours. The National Energy Research Scientific Computing Center (NERSC) has roughly 50 PBs of shared storage utilizing more than 4.6B inodes, and a 150 PB high-performance tape archive, all accessible from two supercomputers. As data volumes increase exponentially, managing data is becoming a larger burden on scientists. To ease the pain, we have designed and built a Data Dashboard. Here, in a web-enabled visual application, our 7,000 users can easily review their usage against quotas, discover patterns, and identify candidate files for archiving or deletion. We describe this system, the framework supporting it, and the challenges for such a framework moving into the exascale age.**

I. Introduction

Scientists are inundated with data, and data volumes are expected to increase as new high-luminosity experiments turn on and exascale simulations are run. At the National Energy Research Scientific Computing Center (NERSC), the primary high-performance computing (HPC) center for the Office of Science in the U.S. Department of Energy [1], scientists are already encountering challenges with managing their data. NERSC supports more than 7,000 scientists from a broad range of scientific disciplines processing both experimental and simulated data in large volumes. NERSC offers 35 PB of Lustre [2] and 20 PB of Spectrum Scale (formerly GPFS) [3] disk storage, as well as more than 150 PB of tape storage in a High-Performance Storage System (HPSS) [4] archiving system, and 1.8 PB of NVMe Burst Buffer storage [5]. Long-term disk storage is typically filled to 90% capacity with scientific data. Future projections estimate that the volume will increase by a factor of 10 - 100 by 2025 [6].

While scientists have grown increasingly adept at producing data, there has been no commensurate increase in the sophistication of the tools available to manage data sets. As a result, scientists spend a large fraction of their time in manual data management. For instance, scientists often have only "du" and "ls" to examine tens to hundreds of terabytes of data spread across tens of millions of files at NERSC. They must manually move this data through a multi-level storage hierarchy (i.e., from high-speed scratch to a long-term tape archive) and manually verify that the transfer succeeded. The commands can often take hours to days to complete, making data management a time-consuming chore and management of petabyte-size data volumes untenable.

[1]NERSC, Lawrence Berkeley National Laboratory, Berkeley, CA, 94720, USA

*Corresponding Authors

These issues become more complex when considered with the idea of a Superfacility: a partnership that integrates experimental and observational instruments with computational and data facilities like those at NERSC, bringing the power of exascale systems to the analysis of real-time data from light sources, microscopes, telescopes, and other devices. Data from these experiments will stream to large computing facilities where it will be analyzed, archived, curated, combined with simulation data, and served to the community using powerful computing, storage, and networking systems. Scientist may interact with the computing facilites through their experimental facility, sometimes without direct access or membership at HPC centers.

NERSC is partnering with several different experimental facilities to build tools that can be used for the Superfacility project. Some of these partners that have a strong data management need are:

- **The Advanced Light Source** [7], a synchrotron facility used for scientific imaging, produces 30 - 60 TB of data from its beamlines. This number is expected to increase by approximately a factor of 10 in 2025. Data is copied from the experimental facility and processed, stored in the archive, and shared only with the specific beamline scientists. Data must move across many different tiers at the HPC center. The movement and processing is triggered by the beamline maintainer and scientists (who are not necessarily HPC facility members) consume finished products.
- **LSST Dark Energy Science Collaboration** [8] performs cosmological analyses on data from the Large Synoptic Survey Telescope and is expected to consume 10 TB of image data every night. Data must be processed, shared with collaborators, and archived for long term storage.
- **National Center for Electron Microscopy (NCEM)** [9] is developing a high frame rate (100KHz) 4D detector system to enable fast real-time data analysis of scanning diffraction experiments in scanning transmission electron microscopy. Data rates are expected to burst to 360 Gbps during operation. Resulting data must be analyzed and archived.

Scientists at these facilities will need the ability to curate, share, and publish data programmatically and on a mass scale. Managers of superfacilities will need the ability to discover and manage data on behalf of scientists, as well as shepherd the data through analysis pipelines. While the goals of these facilities may be wildly different, they face common challenges throughout the life cycle of their data.

We intend to ease some of the burden of managing data by creating a unified framework that provides basic functionality for all phases of data's life cycle. In this paper we will describe the initial deployment of this framework, the Data Dashboard, a web-enabled visual application that allows NERSC's 7,000 users to review their usage against quotas, discover patterns, and identify candidates for archiving or deletion. We will describe the design and support framework for the Data Dashboard, and discuss future plans for further deployment of data management tools.

II. NERSC Environment

NERSC is one of the largest facilities in the world devoted to providing computational resources and expertise for basic scientific research. It provides some of the most powerful computing and storage systems designed to accelerate scientific discovery.

A. Computing Resources

To support both data intensive and simulation workloads, NERSCs newest supercomputer, Cori, a Cray XC40, comprises a "data partition" made up of 2,004 dual-socket compute nodes with two 2.3-GHz, 16-core Haswell processors for a total of 64,128 cores and 128 GB of DRAM per node, and an "HPC partition" made up of 9,300 Intel Knights Landing (KNL) nodes. The KNL partition consists of more than 632,000 cores and has a combined 1 PB of memory. The innovative dual-architecture configuration within the same supercomputer is driven by two factors: the increasing demands of data analysis in HPC workflows, and the need to support both data intensive and simulation workloads at scale. Cori has a Cray Aries high speed "dragonfly" topology interconnect. Cori debuted as the world's fifth most powerful supercomputer in the 2016 TOP500 list [10] and is used for scientific computing at all scales. NERSC also has another supercomputer, Edison, a Cray XC30, with 5,586 dual-socket compute nodes, each with two 2.4-GHz, 12-core Ivy Bridge processors, for a total of 134,064 cores and 64 GB of DRAM per node.

B. Data Storage

A unique feature of the Cori system is the integration of a high-bandwidth SSD-based storage layer or Burst Buffer. For more than two decades, supercomputers have used large-capacity, high-performance file systems, which served both as temporary capacity storage and as a high-performance data sink for job I/O. Having a high-bandwidth storage tier improves machine productivity by letting defensive I/O, such as checkpointing, complete quickly. The Cori Burst Buffer file system [5] is based on Cray's DataWarp application IO accelerator technology [11]. It is made up of 288 DataWarp nodes, each with two Intel P3608 3.2-TB NAND flash SSD modules attached over two PCIe gen3 interfaces. These are packaged two to a blade and attached directly to the Cray Aries network interconnect of the Cori system. The Burst Buffer provides a total 1.8 PB of storage with an aggregate bandwidth of 1.7 TB/s. Requests for this storage go through the Slurm workload manager [11], and data are staged in and out from the Lustre file system via Cray's DataWarp software. The Burst Buffer is intended for extremely fast I/O during a job. Users can make reservations to keep data on the Burst Buffer between jobs. However, because of the limited size of the Burst Buffer, it is not intended for long-term storage.

Cori's Burst Buffer adds an additional layer to the traditional HPC storage hierarchy. The next layer closest to the compute nodes is a 28-PB Lustre file system that serves as a temporary home for data while it is being used for computation. Served by 248 OSTs and 5 MDS/DNEs, it delivers an aggregate bandwidth of 700 GB/s. This file system is mounted on both Cori and Edison, and serves as the primary scratch file system for Cori. Each user has their own scratch directory with a quota of 20 TB and 10M inodes. To manage the capacity growth of the scratch file system, files not accessed within 12 weeks are automatically purged.

Moving further down the hierarchy, NERSC also has a 20-PB Spectrum Scale file system called "Project" that is mounted on all systems at NERSC. The file system uses storage arrays built from enterprise-grade hard disk drives. Two copies of file system metadata are stored on high-performance SSDs. The file system fabric is a non-blocking Fat Tree built on FDR Infiniband. RDMA is used for bulk data transfers. To provide access to the Project file system from within the Cray compute clusters, the Cray Data Virtualization Service (DVS) [12] is used to forward I/O requests between the Project file system and compute nodes. DVS nodes are attached to both the Infiniband storage fabric and the Cray high-speed internal network.

Project is intended to be used as a shared data repository for science groups and for medium- to longer-term storage of large volumes of data. To facilitate data sharing, each science group is given a separate directory (a Spectrum Scale fileset) with a directory quota and group-readable Linux permissions. The default quota for each directory is 1 TB and 1M inodes, but this can be increased up to hundreds of TBs or even larger as needed. Quota enforcement is managed by Spectrum Scale software, which forbids further writes when the quota is exceeded. The aggregate bandwidth of Project is 400 GB/s.

An expansion to this tier is planned in at least two phases, with an initial deployment of approximately 75PB in 2019 and approximately 200PB in 2020. The tier will be disk-based with a focus on capacity over performance, allowing for larger quotas.

At the base of the NERSC storage hierarchy is a tape-based storage system. HPSS, or the High Performance Storage System, is a hierarchical storage management system used for long-term data retention. The system consists of a fast disk cache backed by a large tape layer. HPSS is connected to the compute systems at NERSC using a 400 Gb Ethernet link. Current HPSS deployment consists of two independent systems: one for scientific data archive and one for system backup, with a total of 150 PB stored. Data in the archive dates back nearly 40 years.

Each NERSC user is given a directory in HPSS for storing data. Group directories are also available by request. Usage is controlled by a yearly allocation of storage space. This is used primarily for record keeping. In practice, storage volumes under 1 PB/year are generally acceptable. Users can access HPSS via hsi, htar, ftp, and Globus [13].

C. Usage Patterns

At the beginning of this project, we recognized the need to thoroughly understand how users move and interact with their data on NERSC systems and at what points issues arise. Project members are frequently engaged with users as consultants, so we had a history of support experience from which to draw for our use cases. But we also recognized the importance of talking directly with users outside the context of a particular problem. We therefore undertook a series of interviews with researchers from a range of scientific domains, to identify the greatest pain points across the center. With this approach, we were able to identify the most desperate needs and the most pressing questions that our users have about their data. Besides qualitative information from interviews, we also examined quantitative data about file system usage to aid prioritization. Another means of determining user needs was to look at custom tools that some large user groups had already developed for themselves. We found that the most pressing needs were for managing growth against quotas, managing permissions, and more easily moving files from one storage system to another.

The capacity, performance, and administrative policies of the file systems greatly influence user behavior at NERSC. Given the choice, most users would prefer to store all their data on the Lustre file system, since it has the highest bandwidth to NERSC's computing resources. However, purchasing enough Lustre storage to enable every user to use their full 20-TB quota would be prohibitively expensive, so NERSC implements purges on the Lustre file system that automatically remove files not read in 12 weeks. This motivates users to move their data to the slower (but permanent) Project file system or HPSS. Scientists also need to publicize their data via web portals. These portals don't require high bandwidth, but do require long-term storage capacity which is well served by the Project file system. Storage allocations on Project are managed by directory quotas, and sometimes users are forced to archive "warm" data into HPSS because of space constraints. We anticipate the expansion of the project tier, and associated larger project quotas, will reduce some of this "warm" data migration. These pressures combine to create a pattern at NERSC where data typically starts on the Lustre scratch file system or Burst Buffer, gets migrated to Project, and finally is stored in HPSS. However, many exceptions to this pattern do exist. For example, climate simulations require very large datasets that are read for a few months while a new simulation is performed. In these situations, the O(10 TB) datasets may be moved directly from HPSS to scratch and returned to HPSS when the campaign is done. With the exception of the Burst Buffer, all data movement between the different storage systems is done manually.

Movement and dealing with separated file systems consume many hours of user effort. Several million inodes are deleted, created, or modified daily on the Project file system, and multiple terabytes of data are deleted or created. Cori's Lustre file system has tens of millions of files modified and nearly half a PB of data written per day. With such active data volumes, users often struggle to find their files across these many storage systems, principal investigators must manage quotas for groups of tens to hundreds of people, and ensuring data are migrated across tiers requires tedious checking.

III. Data Dashboard

The first user-level product of our data management framework is a tool for rapidly visualizing the state of a user's file storage, which we call the Data Dashboard. The goal is to enable a user to visualize their files across all file systems and to identify at a glance any storage area that is at risk of exceeding a quota, to identify users or groups who can be asked to free up space or inodes, and to identify the individual files and directories that contribute most to overages.

Ultimately the Data Dashboard will become the central hub for most routine data management tasks. For instance, when a users directory is near quota, it will provide a quick list of their largest directories. By clicking onto directories, users will be able to drill down and find files that havent been accessed in months. Selecting these files and dragging them to an HPSS icon will trigger backend invocations of the framework tool that will automatically archive them in HPSS. A rules interface will let them select directories and create workflows by using pull down menus with choices to indicate targets (files in this directory), conditions (new and larger than XXGB) and actions (copy to Lustre in YY directory).

A. Implementation

The data that feed the Data Dashboard are generated by daily Spectrum Scale scans. This scan outputs the full path, size, ctime, atime, owner uid, and owner gid for every file and directory in the Project file system. Initially the scans took about 12 hours. This was sped up by about a factor of five by moving filesystem metadata to SSD and by scaling out the number of nodes running the scan.

The scan generates a text file that contains $\sim$1B lines and is about 200 GB. This text file is parsed using the Spark framework [14] to produce aggregate usage for each user (in space and inodes), as well as the size and inode count of each directory on the file system. This process generally takes about 5 minutes on 16 Cori compute nodes. The entire process from scan to display takes between 4 and 10 hours in total, which allows us to refresh the data daily (Fig. 1).

The resulting data for each project directory is stored in a JSON-formatted file that can be read by PHP scripts invoked by the web server. Data from the scans is also loaded into a PostgreSQL database that can be queried to retrieve information to support additional views. Details of

how we use this data are given below in the discussion of the components of the user interface.

B. User Interface

In developing the user interface for the Data Dashboard, we followed a user-centered design approach. We began the design process by talking with NERSC research team data wranglers, individuals who perform a majority of a project's data movement and archiving work. Interviewees were selected to cover a range of scientific domains. We targeted large collaborations known to push the limits of their available storage. Some interviewees were principal investigators; others were not. Through these discussions, we developed a list of user needs and an understanding of how we might best address them with our tools.

When we began designing the system, we had planned to create two distinct interfaces, one for principal investigators and one for regular research group members. We soon discovered that the file system data from the Spectrum Scale scans did not support the level of permissions needed to differentiate content for the two views. As it turned out, our needs assessment showed that enough commonality existed between the principal investigators and others that we could create a single interface to meet most of the needs. Two of the biggest needs identified by our users were seeing growth against quotas, by owner or group, and finding files to archive, by age, size, and number. We chose to address these issues in our first release.

Based on the input from our interview group, we next created a prototype dashboard for testing. The prototype enabled us to refine the pipeline by which we move data from Spectrum Scale scans to the web server, and it also allowed us to conduct usability testing. The usability tests consisted of in-person, think-aloud task completions observed by a trained usability researcher. Before releasing the initial version to all users, we also solicited simple feedback from a group of test users given early access. After the initial release, we continued usability testing and iteration while adding new features.

Our dashboard interface has been integrated into the existing MyNERSC extranet web pages [15]. The MyNERSC site offers a suite of user tools for account and job management, so it was a natural fit for data management tools as well. It takes advantage of the existing NERSC Web Toolkit (NEWT) API [16] for authentication, so this provided a ready means of handling user logins. We have created custom API calls within NEWT to handle requests from the Data Dashboard. At page load, one custom call sends a request to a PHP script that obtains information about project membership for the logged-in user and then uses that information to grab the JSON-formatted data for each Project directory. The extra hop through the NEWT web server to the PHP script is needed because the request for user data must originate from a local machine. The MyNERSC web page, upon receiving the json data, has what it needs to construct the first set of visualizations in the user interface. We use the D3 Javascript library to render the visualizations.

On initial page load of the Data Dashboard, a user who is not yet logged in is prompted to do so. Once the user is logged in, the page is rendered as a series of small graphs representing each of the user's Project directories (Fig. 2). These show overall Project usage as a percentage of allocations, for both space and inodes. A toggle button allows the user to open up the display for each Project directory to reveal four additional bar graphs, showing the breakdown of usage, by users and groups, for both space and number of inodes (Fig. 3). The user can hover the cursor over any of the bars to obtain a small popup with usage details about the relevant group or user's usage of space or inodes. As our users requested the ability to copy and paste the detailed numbers into warning emails to collaborators, we enable the user to "pin" and "unpin" the popups in place with a click on the bar. For each Project directory, the user can also download the usage data in CSV format for use offline processing. For each project, we show the time at which the data was last refreshed.

We have continued to iterate the design and add features that address the needs of our users. In the first major revision of the Dashboard, we added an additional button within each Project directory panel that reveals a visualization of the user's "biggest" directories and files. That is, it shows the files and directories that use the most space and the greatest number of inodes. It is rendered as three graphs, one showing the top n largest individual files, one showing the n largest directories, and one showing the n directories with the most inodes. A drop-down menu allows the user to set n to 10, 20, 50, or 100. Coloring in this view is by atime or ctime, with the newest files appearing in a spring green and the oldest aged to a dark brown. The user can quickly switch between atime and ctime coloring with a set of radio buttons.

One requirement in representing the largest user directories was to elegantly handle the case where a single large directory sits in a parent directory that contains little else. Both parent and child would appear in the results, and the user would not know that the child was the directory on which to focus cleanup efforts. We therefore by default remove big parent directories from the list if the difference in size or number of inodes between them and a large child directory is not large enough to land in the top n. If they prefer, the user can also uncheck a checkbox to turn off the removal of parent directories.

We have also recently added a graphical file browser that lets users drill down into all of their directories in each project. The browser allows two views, a sunburst representation (Fig. 5) and an icicle plot, which the user can toggle between. The graphs show the hierarchy of directories, and a colorful strip at top shows the path to the directory most recently rolled over with the cursor. A button enables the user to capture this path to the clipboard.

The user-focused "Biggest Files and Dirs" view and the file browser are powered by a PostgreSQL database. The handling of data from Spectrum Scale scans now includes the loading of a database table with data for each inode that includes calculated directory size information as well

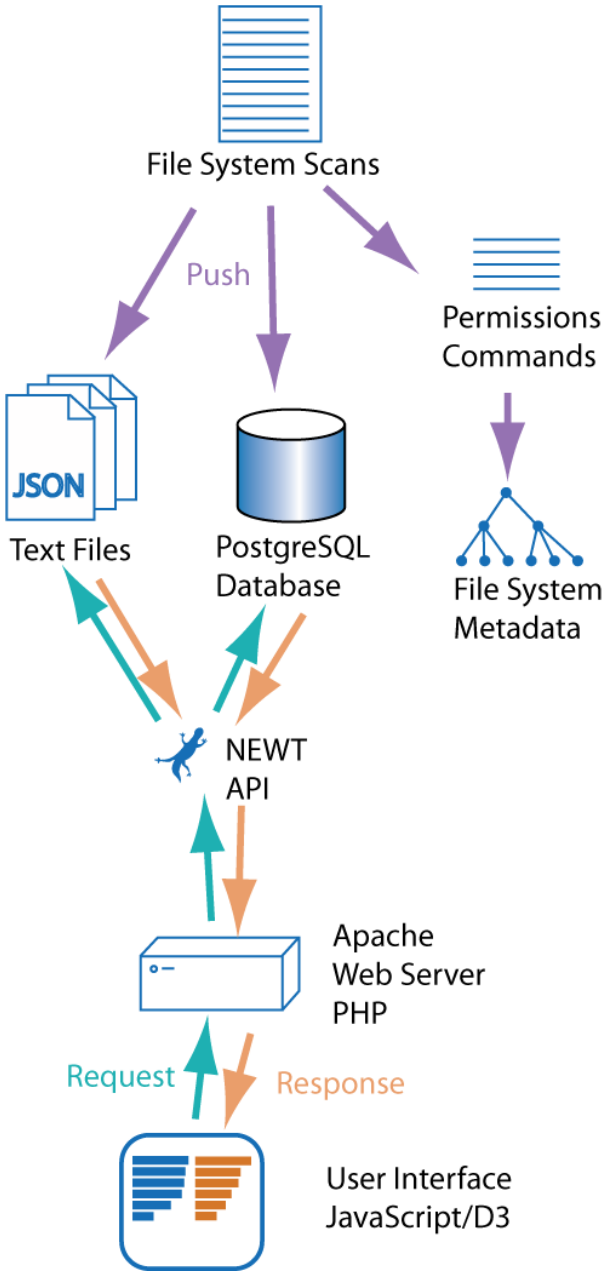

Fig. 1. A schematic diagram of the framework, showing data flows.

as the scan metadata. Data loading can be time intensive, taking between four and ten hours to load the data for all file systems in parallel. Once loaded, however, queries perform at acceptable speeds for a web application. Indexing helps speed up the queries, which must find all inodes of a given type owned by the logged-in user, sort, and return the top *n* hits.

C. Permissions Wrangler

To address the need of users to maintain proper group permissions throughout a project, we have developed a facility for correcting permissions on a daily basis. A Spark script mines the same scans used by the dashboard, looking for files and directories that have deviated from group readable permissions. It generates a list of permission-correcting commands that are then automatically run on a node with elevated permissions. At present, the permissions wrangler is being beta tested with a few project groups. We plan to eventually include an option for principal investigators to "opt in" to this service on the Data Dashboard.

D. Usage Statistics

Usage of the Data Dashboard is tracked a number of different ways. Google Analytics tracks unique page views over time. In the first quarter of 2019 we saw 59 unique and 168 non-unique page views of the Data Dashboard per week.

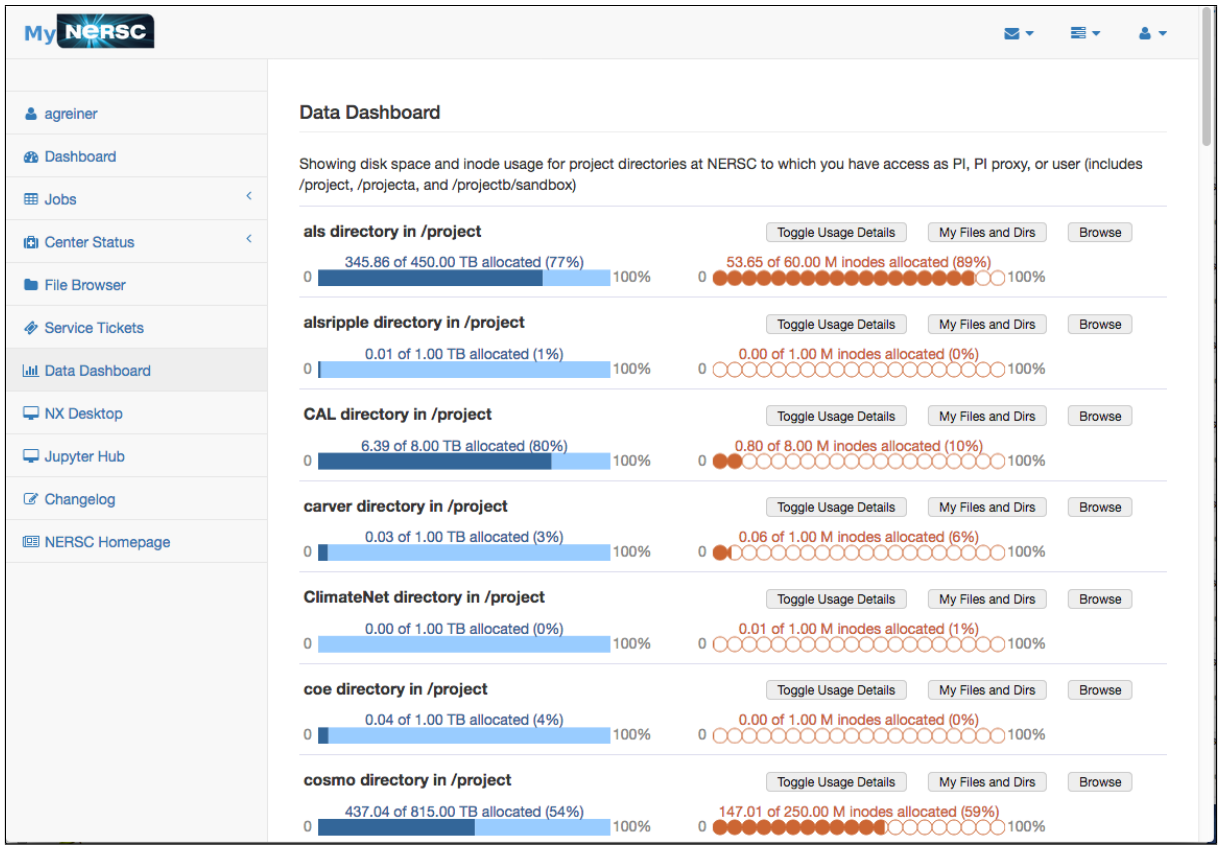

Fig. 2. The Data Dashboard user interface as it loads within MyNERSC. Each of the user's Project directories (here "als", "alsripple", "CAL", "carver", etc.) is represented by a panel, showing the percentages of Project space and inode allocations being used.

E. Challenges

The Lustre file system does not have any built-in policy functionality that could be used for data identification and tracking. Typically this is provided by the Robinhood Policy Engine [17], an open source tool that provides metadata tracking and policy engine functionality. While the functionality offered by Robinhood is good, in practice it often struggles to keep up with the load from a very large file system, and falling behind can lead to file system outages [18]. A quick scan is vital for presenting users with an accurate picture of their data and for enforcement of data management policies. NERSC uses Robinhood to parse Lustre Changelog information into a MariaDB database, but failure to keep up with changelogs or other Lustre failures and bugs frequently causes the database to get out of sync. Once Robinhood lags behind by a substantial number of entries, a full rescan of the file system to repopulate the database from scratch is required. On a file system with several PBs, this can take several days. Recent developments have been aimed at addressing some of these issues. However, while the Robinhood framework has been used for file system scans on Cori, it is not yet known if these new updates can keep up with the usage of a ∼30 PB Lustre file system.

For HPSS analytics, NERSC has created an in-house system for analyzing and storing data transfer and operations logs, as well as tape library statistics and health markers. Until recently, this database containing over 10 years of data has not been integrated into a centralized system. We are in the process of ingesting HPSS transfer logs into the web UI, with the goal of matching the level of detail provided by the Spectrum Scale scans.

IV. Future Plans

A. Data Dashboard

The Data Dashboard functionality will be extended to include full file system information for Sprectrum Scale, Lustre, and HPSS, with the ability for users to "walk" their directory trees and search for files across file systems. Eventually, the Data Dashboard will serve as the central control point for data movement in the center.

An API will also be available for users so that groups can incorporate the data management framework into their existing pipelines, replacing many lines of complicated code.

B. File System Evolution and Integration

NERSC has outlined a plan for the evolution of their storage systems for the next several years [19]. The Project and HPSS layers will be unified into a single "off-platform" tier and the high performance layers (Burst Buffer and Scratch) will be unified into a platform-integrated tier. Having fewer layers will greatly simplify data movement. In addition, NERSC is looking at deploying unification tools like GHI

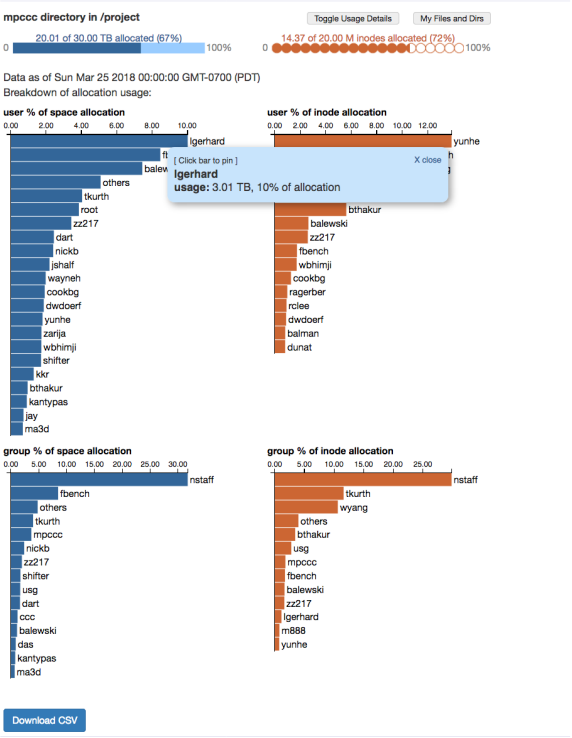

Fig. 3. One panel of the Data Dashboard user interface showing that Project directory's details expanded and a user space popup "pinned".

(Spectrum Scale / HPSS integration) [20] that allows policy-driven migration of files between the Spectrum Scale file system and HPSS. This coupling can be further leveraged by creating tools that allow users to define migration policies for their data. For instance, users could write pseudo code to detect when a new file has arrived in a particular directory, apply some criteria to it (like checking file size), then migrate this file to HPSS, and notify the user when it is finished.

C. *Superfacility Project*

The Data Dashboard will ultimately be the front end for the data management piece of the Superfacility Project, a suite of features designed to facilitate experimental and observational workflows across HPC centers and user facilities. In addition to data management tools, the Superfacility project will also include tools to assist in

- **Scheduling**: tools to make realtime and bursty scheduling easier across HPC centers, including reserving non-compute resources like network bandwidth or file system space for complex workflows
- **Centerwide Access**: An API for interacting with all aspects of an HPC center
- **Software Defined Networking**: Seamless streaming of data directly to compute nodes
- **Federated ID**: Leverage identity federation techniques to provide seamless access to data and services at HPC centers using credentials from a scientist's home institution.

The Data Dashboard will work in tandem with these tools

Fig. 4. The Data Dashboard view of one user's biggest files and directories, showing their largest files, largest directories, and directories with the most inodes. Coloring reflects the access time of each inode.

to facilitate experimental and observational workflows in the exascale era.

V. Related Work

A number of scanning frameworks for different file systems have been developed over the years. The most recent one, the Grand Unified File Index (GUFI), was developed by researchers at Los Alamos National Laboratory [21]. GUFI stores file metadata in a hierarchy of databases, which allows rapid searches across the entire scan. GUFI is still in development, but could be a potential solution to scanning a large Lustre file system.

iRODS is an open source data management system that is supported by a group at RENCI [22]. All data is accessed via the iRODS server. Data can be accessed directly from the storage pool without using iRODS, but the iRODS

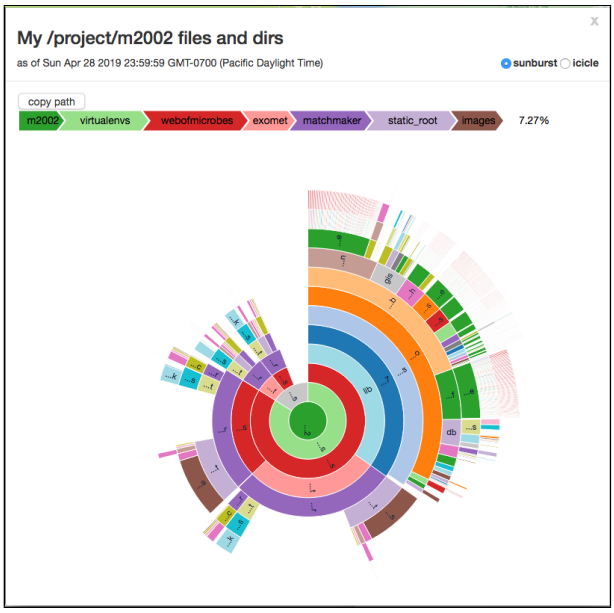

Fig. 5. Sunburst view of a user's files in the Data Dashboard. The path at top reflects the file most recently rolled over with the cursor.

directory structure is opaque, making this cumbersome. A rich suite of metadata for each file can be stored in the iRODS database. Programmable rules can be created to move storage between different layers of the storage pool while still presenting end users with a centralized namespace. While iRODS includes much of the functionality proposed here, it does it by enforcing a new framework. Many NERSC users operate on multiple platforms, and redesigning their workflows to accommodate something that is only used at a single center is enough of a burden to make the effort untenable. Maintaining a separate framework would also be quite challenging to scale alongside the file system as bandwidth and IOPS increase, representing an ongoing cost to NERSC in addition to the cost of the storage itself.

Researchers at Oak Ridge National Laboratory have developed TagIt, a scalable, distributed metadata indexing framework which allows users to "tag" their data with appropriate metadata information [23]. The system performs well at scale, but is tightly integrated to the GlusterFS file system (future plans may include CephFS, but not Spectrum Scale or Lustre).

Several vendors offer integrated metadata and search functionality and tier migration. For instance, Starfish [24] uses a software-only solution that doesn't require integrated hardware and works across a variety of file systems but their offering didn't cover all of the file systems at NERSC nor cover our needs for individual tools.

VI. Conclusion

The Data Dashboard described in this paper will remove much of the manual, redundant effort scientists spend managing data, freeing them up to do more science. The increasing complexity and size of storage systems mandate better tools for taming the data deluge and interfacing with HPC centers in the exascale age.

Acknowledgment

This research used resources of the National Energy Research Scientific Computing Center, a DOE Office of Science User Facility supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

References

[1] http://www.nersc.gov/about/.
[2] http://lustre.org/.
[3] http://www-03.ibm.com/systems/storage/spectrum/scale/.
[4] http://www.hpss-collaboration.org/.
[5] W. Bhimji *et al.*, "Accelerating science with the nersc burst buffer early user program," 2016. https://cug.org/proceedings/cug2016_proceedings/includes/files/pap162.pdf.
[6] S. Habib *et al.*, "Ascr/hep requirements review report," 2016. arXiv:1603.09302v2.
[7] https://als.lbl.gov/.
[8] http://lsst-desc.org/.
[9] http://foundry.lbl.gov/facilities/ncem/.
[10] https://www.top500.org/lists/2016/11/.
[11] http://www.cray.com/datawarp.
[12] "Cray DVS Installation and Configuration." http://docs.cray.com/books/S-0005-10/.
[13] https://www.globus.org/.
[14] http://spark.apache.org/.
[15] https://my.nersc.gov/.
[16] S.Cholia, D. Skinner and J. Boverhof, "Newt: A restful service for building high performance computing web applications," *Gateway Computing Environments Workshop (GCE), New Orleans, LA, 2010*, pp. 1–11.
[17] https://github.com/cea-hpc/robinhood/wiki.
[18] T. Declerck *et al.*, "Using robinhood to purge data from lustre file systems," *Cray Users Group Proceedings 2014*, 2014. https://cug.org/proceedings/cug2014_proceedings/includes/files/pap157.pdf.
[19] "Storage 2020: A vision for the future of hpc storage." https://escholarship.org/uc/item/744479dp.
[20] http://www.hpss-collaboration.org/ghi.shtml.
[21] https://www.hpcwire.com/off-the-wire/los-alamos-releases-file-index-product-to-software-communi
[22] http://irods.org/.
[23] H. Sim *et al.*, "Tagit: an integrated indexing and search service for file systems," No. 5, SC '17 Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, 2017.
[24] http://www.starfishstorage.com/.

Appendix

A. Abstract

The framework described in this paper makes use of technologies from a broad range of domains. We use high-performance storage technologies as well as tools for cluster computing, web engineering, and data visualization. The preparation of data begins with metadata scans of the Spectrum Scale file system generated with the mmapplypolicy command. A cronjob is used to trigger a check to determine when a new scan has become available in a designated directory. The new scan file is processed by a series of bash scripts that launch Spark 2.1.1 Python jobs on the Cori Supercomputer. The Spark jobs create a set of JSON files, one for each Project directory on NERSC shared storage. They also load data into a PostgreSQL 9.4.15 database and

build a list of bash shell permission-correction commands to be invoked later by another cronjob.

The Data Dashboard is made available to users via the MyNERSC extranet site (https://my.nersc.gov), which in turn makes use of the NERSC Web Toolkit (NEWT) [16], a web API for high-performance computing. We use some built-in functions of NEWT, such as authentication and authorization (including queries of the NERSC Information Management system and a local LDAP server), and add custom API calls to handle queries specific to the Data Dashboard, such as retrieval of JSON-formatted directory metadata and queries of the PostgreSQL database. Web pages and API endpoints are served by an Apache Web server running PHP 5.6. Data manipulation on the client side is performed with Javascript and the D3 visualization library, version 3.

This paper contains no computational results.