

Anatomy of the SEAGrid Science Gateway

Supun Nakandala
Indiana University
2709 E. 10th Street
Bloomington, IN 47408
snakanda@iu.edu

Sudhakar
Pamidighantam
Indiana University
2709 E. 10th Street
Bloomington, IN 47408
pamidigs@iu.edu

Shameera Yodage
Indiana University
2709 E. 10th Street
Bloomington, IN 47408
syodage@iu.edu

Nipurn Doshi
Indiana University
2709 E. 10th Street
Bloomington, IN 47408
nidoshi@iu.edu

Eroma Abeysinghe
Indiana University
2709 E. 10th Street
Bloomington, IN 47408
eabeysin@iu.edu

Chathuri Peli
Kankanamalage
Indiana University
2709 E. 10th Street
Bloomington, IN 47408
cpelikan@iu.edu

Suresh Marru
Indiana University
2709 E. 10th Street
Bloomington, IN 47408
smarru@iu.edu

Marlon Pierce
Indiana University
2709 E. 10th Street
Bloomington, IN 47408
marpierc@iu.edu

ABSTRACT

The SEAGrid science gateway provides scientists and educators with user interfaces and tools for conducting computational chemistry, material science, and engineering experiments online using XSEDE and campus computing resources. This paper describes the architecture of the recently completed technology refresh for the gateway, replacing its desktop user interface, adding a web browser user interface, using Apache Airavata middleware for job management, and providing enhanced data search and feature extraction capabilities. These introduce several challenges, particularly in providing unified authentication and authorization mechanisms to middleware services for the desktop and web clients, and in extending Apache Airavata middleware with new components. Access, authentication, and authorization problems were solved by using standard-based approaches (OAuth2, XACML) that were implemented by incorporating WSO2's Identity Server into both SEAGrid and Apache Airavata. SEAGrid-specific data extraction capabilities were added to Airavata middleware using a message-based component approach. This approach is generalizable to other advanced and gateway-specific capabilities and enables Airavata to add additional data analysis components without modifying its core functionality.

CCS Concepts

•Applied computing ~ Chemistry •Applied computing ~ Engineering •Software and its engineering ~ Software creation and management

Keywords

Science gateways; computational chemistry; distributed systems; material science

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org. XSEDE16, July 17-21, 2016, © 2016 ACM. ISBN 978-1-4503-4755-6/16/07...\$15.00 DOI: <http://dx.doi.org/10.1145/2949550.2949591>

1. INTRODUCTION

The Science and Engineering Applications Gateway (SEAGrid.org) [1] [2] provides access to computational chemistry, material science, and engineering applications. SEAGrid, formerly known as the computational chemistry grid (CCG/GridChem), has been in operation since 2005 and currently has 656 users organized into 329 projects. In 2014, the CCG/GridChem was rebranded as SEAGrid to convey its ability to support more than just chemistry applications. Since the beginning of XSEDE in 2011, SEAGrid has delivered over 45 million Service Units (SUs) to its user community and has executed over 71,700 jobs on XSEDE, placing it by both measures as the second most used XSEDE science gateway. A distribution analysis of jobs supported shows diverse sets of runs ranging from short to very large jobs that used 1000s of hours. ; All usage statistics are publicly available from XDMOD (<https://xdmod.ccr.buffalo.edu/>), XSEDE's metrics publication and analysis site. SEAGrid is used to teach classes and do research. SEAGrid services have resulted in more than 120 publications, 50 conference presentations, and at least 13 graduate MS and Ph.D. theses; see <https://seagrid.org/pages/publications> for detailed publication list.

This paper describes SEAGrid's recently completed end-to-end technology refresh. This was motivated by several factors. First, the previous SEAGrid user interface was based on Java Web Start and Java Swing; Oracle has announced that JavaFX is the replacement for Swing. Second, the ability to deliver purely web-based rich client experiences has grown dramatically since the GridChem user interface was developed. Third, mobile devices are an important access mechanism for many users, so SEAGrid needed a mechanism to support these users. Fourth, research data services have been lacking in the previous implementation, and this refactoring improves the data services to move toward a data-enabled model for discovery and computing. Lastly, technical and operational support needed to be more scalable in order to expand SEAGrid's support for a broad spectrum of users without losing the appeal of advanced consulting and collaboration with specific

research groups, as has been accomplished by the CIPRES science gateway [3] and Galaxy [4].

To accomplish this technology refresh, we re-implemented both the user interface and the middleware. We chose to retain the desktop user interface experience, which integrates well with other desktop tools, and added a new purely web-based user interface. The use of dual user interfaces, which could also be extended to support embedded clients in users' scripts, introduced technical challenges: we present here a solution that unifies the account creation, authentication, and authorization mechanisms between the two user environments and provides a seamless mechanism for both web and desktop clients to access the SEAGrid data store. Adopting Apache Airavata as middleware allows SEAGrid to have tiered support, with the gateway lead (Pamidighantam) providing scientific expertise but with operational issues and software upgrades handled by a team of operators and developers. Integration with Apache Airavata provides another advantage: as an extensible, component-based system, Airavata can be extended to add additional capabilities and components without disrupting its core capabilities. This paper shows how we extended Apache Airavata to provide SEAGrid-specific data post-processing and search services.

2. SYSTEM OVERVIEW

SEAGrid Gateway Capabilities: SEAGrid's goal is to provide both new and advanced users with tools to access computational science applications on advanced computing resources. SEAGrid uses XSEDE resources for computing and storage and has also integrated with advanced XSEDE services for resource scheduling, monitoring, and prediction [5] [6]. SEAGrid is also being integrated with the XSEDE-supplied Globus Transfer service. Many of these activities are supported by consultations with XSEDE's Extended Collaborative Support Services staff. SEAGrid has also powered XSEDE extended consultations, most notably virtual diffraction enhancements to LAMMPS [7].

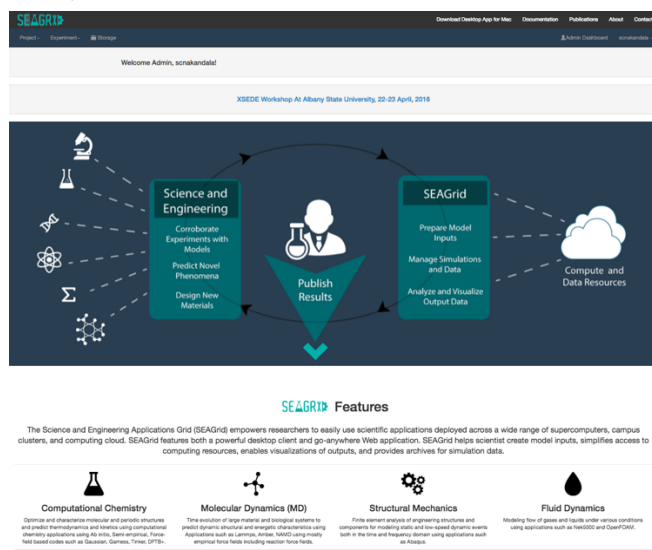


Figure 1 SEAGrid's web interface.

SEAGrid provides both web and desktop user interfaces; examples are shown in Figure 1 and 2. The desktop user interface is developed using JavaFX and integrates with external tools such as NanoCAD. This provides an upgrade path for SEAGrid's current user base, who access the gateway through a Java Swing-based client. SEAGrid's new web client is based on Apache Airavata's reference client implementation, the PHP Gateway for

Airavata (PGA). The PGA is developed in PHP with JavaScript and Bootstrap to enable interactivity and responsive design so that the user interface can scale to multiple screen resolutions.

Both the SEAGrid desktop and web interfaces allow authenticated and approved users to create, execute, monitor, and otherwise manage computational experiments. Users can also upload input data and download output data from experiments. The web interface supports administrative functions available to privileged users. These include the ability to add, modify, and delete computational resources and scientific applications that are available to regular users' experiments, to manage user accounts, and to view monitoring dashboards to get log information on failed and successful experiments for all users for given time periods. Computational experiments, based on Apache Airavata's data models, serve as SEAGrid's primary organizational structure for user interactions. Experiments are associated with one or more XSEDE-maintained scientific applications on backend computing resources.

Currently, SEAGrid supports over fifteen science and engineering applications. The most popular of these are Gaussian, Quantum Espresso, CHARMM, NWChem, LAMMPS, ABAQUS, and NEK5000. SEAGrid supports computations on XSEDE's Stampede, Comet, and Gordon resources; Jetstream and Bridges are currently being added. SEAGrid can also be integrated with campus resources; this has been done for Big Red 2 and Karst at Indiana University, and Raven at Cardiff University.

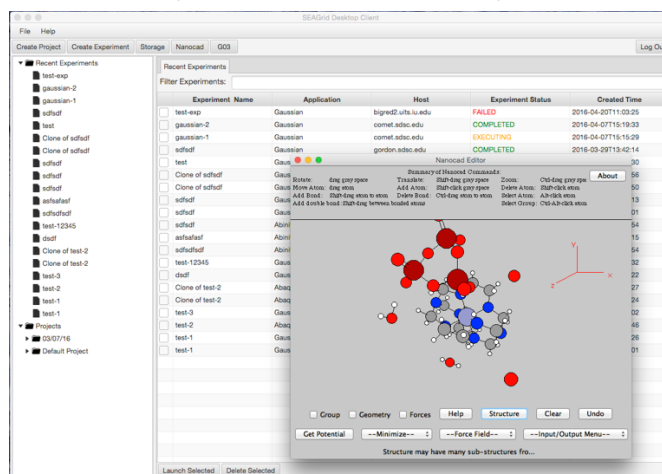


Figure 2 SEAGrid's desktop client

Apache Airavata and SciGaP: SEAGrid's web and desktop interfaces interact with XSEDE Quarry-hosted Apache Airavata middleware and associated third party services. We refer to the hosted versions of the third party systems (which also include a RabbitMQ message bus, a MySQL database, and Apache Zookeeper) needed to run Apache Airavata as a production service collectively as Science Gateway Platform as a service (SciGaP). Apache Airavata itself is a collection of integrated components accessed through a unified API and data model, both defined using Apache Thrift, that are used to manage computational experiments and data. Apache Airavata is middleware that acts as a client to a wide range of remote computing, storage, and other middleware services and as a server for end-user clients. Apache Airavata's API and components are described in more detail in [8][9].

Apache Airavata provides a mechanism to move data between SEAGrid's data store and a target computing resource, execute remote commands on the resource (typically by executing

generated SLURM or PBS submission scripts), and monitor applications. These operations are organized as “experiments”, which contain all the metadata about a particular invocation of a scientific application or workflow. The format of an experiment is defined using Apache Thrift and is part Apache Airavata’s data model. These operations collectively form Apache Airavata’s gateway user API methods. Apache Airavata also exposes administrator level API methods to its client gateways. These include mechanism for creating and modifying metadata about applications and computing resources, and for searching experiments. Apache Airavata does not provide functionality to explicitly manage users but instead depends upon the gateway and third party services to identify users and their groups and roles. Airavata also does not provide persistent storage; this is the gateway’s responsibility. Apache Airavata uses a messaging approach [10] for inter-component communication and to enable extensibility beyond its core components. This removes tight coupling and allows multiple components to subscribe to particular topics. As described below, we exploit this design to add SEAGrid-specific extensions.

From SEAGrid’s perspective, the most important third party service provided by SciGaP hosting is the Identity Service (IS), an open source software system developed by WSO2. This service is directly accessed by both SEAGrid clients and Apache Airavata. WSO2 IS acts as the gateway’s identity provider and supports several identity management capabilities including OAuth2, OpenID, SAML SSO, user/role/group management and XACML 2.0/3.0 based entitlement management. Airavata can be configured to work with WSO2 IS to provide API security using OAuth2 when API security is enabled in the Airavata server, as described below.

3. ACCOUNTS, AUTHENTICATION, AND AUTHORIZATION

As with many gateways, SEAGrid must provide authentication and authorization so that only gateway-approved users can access resources. For SEAGrid in particular, there are two challenges. First, its interactions with remote resources are brokered by the Apache Airavata services, which SEAGrid accesses through the Airavata API. Second, SEAGrid provides both desktop and web-based clients, which have different security issues.

Before describing the authentication and authorization, we note that SEAGrid users act in one of four roles with associated permissions. These are shown in Table 1. Assignment and enforcement of roles are described in subsequent sections.

Table 1. SEAGrid Roles and Permissions

Role	Permissions
USER-PENDING	Cannot use any Apache Airavata methods.

GATEWAY-USER	Can use Apache Airavata methods for creating, invoking, monitoring, and cloning experiments.
SEAGRID-ADMIN	Can create, delete, and modify metadata for accessing computing resources and applications; can approve, modify, and remove users; can access all users’ experiments and summary information.
SEAGRID-READ-ONLY-ADMIN	Has read access to experiments of all users, can view experiment summary information.

3.1 Account Creation and Management

SEAGrid provides access to a wide range of applications (including some that are licensed) as well as high-end resources on XSEDE and on campuses, so it requires a full account creation process: there is no anonymous access to resources, users must create accounts with strong passwords, users must verify by email that they have requested accounts, and gateway admins must approve the accounts and assign roles to the user. In particular, the account approval phase may be supplemented by additional information: a gateway admin may wish to personally contact the applicant or determine the researcher’s credibility through web searches. These may be used in deciding, for example, how much allocation access the user should have.

The capabilities described above are implemented using the features in WSO2 IS, which provides both its own web interface and SOAP service endpoints. Figure 3 illustrates the control flow. The SEAGrid web interface includes the SOAP client library along with IS administrator credentials and provides web form interfaces to the operations for creating pending accounts, approving accounts, and assigning roles. The administrator credential secures access between the gateway portal and the WSO2 IS. Using these credential removes the need for the gateway administrators to have administrator accounts on the Identity Server. Obviously, this credential can only be distributed to trusted clients since this enables the possessor to invoke arbitrary user creation, modification, and deletion operations for that particular WSO2 IS tenant. From the point of view of a SciGaP service, the SEAGrid web client can be trusted since access to the client code and deployment are limited to SEAGrid administrators; in general, vetting a provider of a web-based gateway is scalable. Desktop clients are different however, since the code is distributed to any interested user. SEAGrid thus requires all user accounts to be requested through the web portal, and all administrative functions are only provided through the web client.

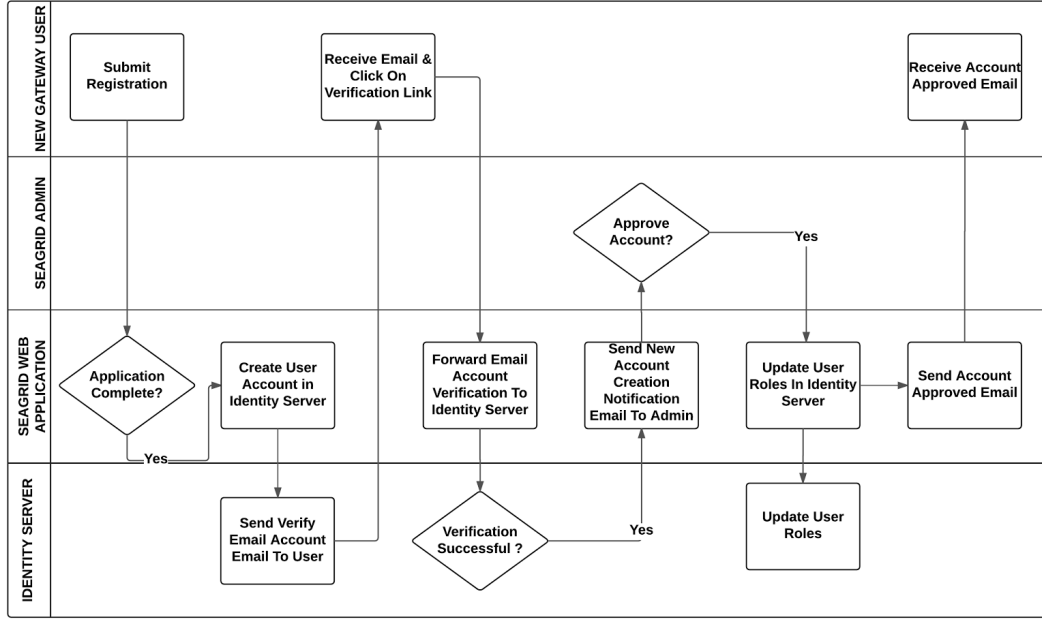


Figure 3 Control flow for user account creation.

3.2 Enforcing Access Control Using Roles

SEAGrid's desktop client currently provides the regular gateway user capabilities associated with the GATEWAY-USER role. These capabilities include functions for creating, launching, monitoring, and retrieving completed experiments. Unlike a browser-based client, which goes through a web server intermediary, desktop clients interact directly with Apache Airavata services through the API. In a web client scenario, we can limit access to Airavata services to only trusted gateway endpoints using known IP addresses, firewalls, and similar approaches. The gateway operator likewise acts as a gatekeeper to the API and can implement access control rules in the client. These approaches no longer work when providing direct access to Apache Airavata services to desktop clients, so it is essential to enforce access control at both the UI level (desktop client) and at the API level (Airavata). Airavata does not provide any access control functionality, but it does provide an extension point in the API through the security manager using WSO2 IS to define custom access control rules using XACML. XACML is the de-facto standard for fine grained, policy based access control. Figure 4 summarizes the communication interactions between the desktop client, identity server and Airavata server in order to achieve access control at the Airavata API. For a general discussion of these considerations, see [11].

In order to use the SEAGrid desktop, the user is first required to login with username and password. Upon providing these, the desktop client invokes the OAuth endpoint in WSO2 IS using a resource owner password credential grant [12]. In order to use this OAuth grant type, it is required to provide an API key and secret pair, which is pre-generated from the IS when invoking the OAuth

endpoint. SEAGrid desktop clients are shipped with a pre-configured API key and secret pair. Upon successful authentication, the OAuth endpoint returns an access token, access token expiration time and refresh token that can be used to get a new access token once the current one has expired. After retrieving an OAuth access token, the user can use the desktop client to invoke Airavata API methods. A valid OAuth access token is a mandatory parameter in every API request.

At the API level, every request is first passed to the security manager for security validation. The security manager then invokes the WSO2 IS administrator services to validate the user request. Every API request is checked whether it is coming from a valid user and then checked to ensure whether that user has required privileges to invoke that method based on the roles the user has. Capabilities of each role map to specific Airavata API methods. For example, a user with the GATEWAY-USER role can invoke only the methods that are relevant to managing experiments and will not be able to invoke administrator API functions. On the other hand, a user with the SEAGRID-ADMIN role can invoke all the administrator API calls in addition to the API methods available to the GATEWAY-USER role. After successful security validation, control is sent to the Airavata API from the security manager to continue the normal execution, or else it will throw an authorization validation failed exception. Currently for every API request an additional call is invoked from the security manager to the IS. There is a slight overhead associated with this communication, and it can be reduced by caching the validation results at the security manager.

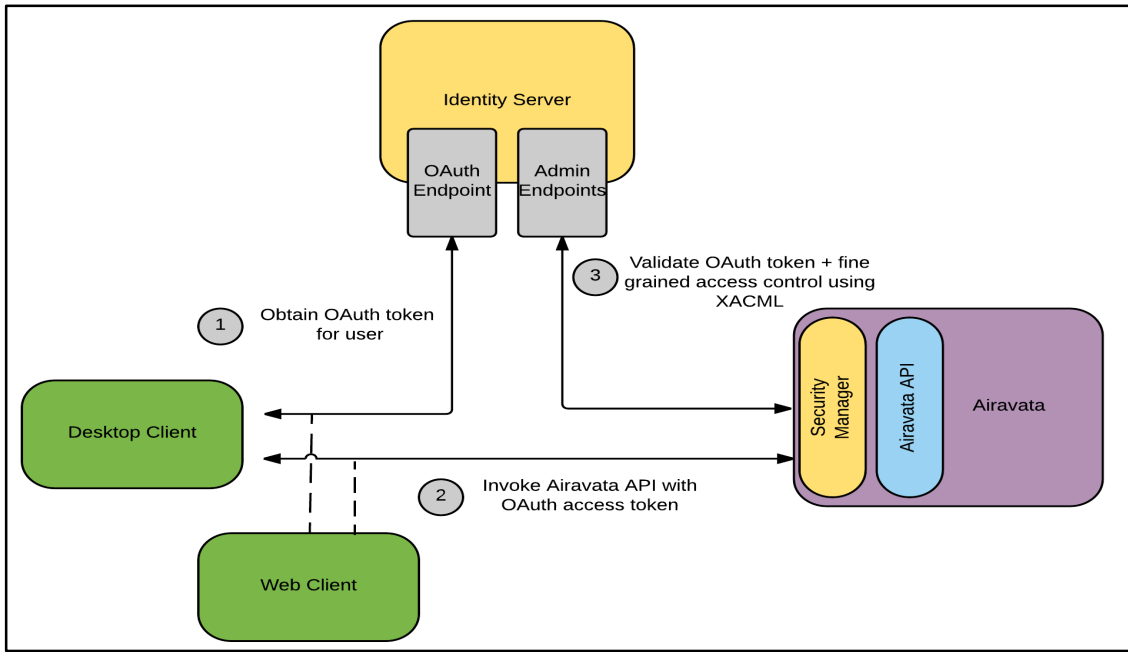


Figure 4 Communication flow between SEAGrid clients, Airavata and Identity Serve

4. MANAGING DATA

SEAGrid must enable both desktop and web client users to upload data to the gateway and download generated outputs. SEAGrid has the notion of a gateway data store, which is simply a storage resource that hosts all the user provided inputs and generated output data. The data movement between this gateway data store and the remote compute resources is handled by the Airavata. Here, we address the problem of getting the data from the user’s desktop to the gateway. The gateway data store is specific to the gateway and not an Apache Airavata component. Following the Apache Airavata data model structure [9], SEAGrid’s data store uses a hierarchical storage and naming scheme. The current implementation is to simply use the file system and standard file system operations. Data are arranged using */gateway-ds-root/user/project/experiment/* directory hierarchies. Each user has one or more projects, and each project is divided into multiple experiments. Data are stored in corresponding directories. We note that this file system is entirely owned by the account that runs the gateway service, so “user” in the hierarchy does not mean that we need for the user to have an account on the gateway host operating system. This is simply a convenient organization of the data.

Input data uploading occurs after the user has chosen a project and created or cloned an experiment within the project. Experiments can be associated with one or more applications. The user then executes the experiment, passing the pointers to the input files for Airavata to upload to the destination computing resource. When the experiment is complete, Airavata stages the output files back to the gateway data store in the appropriate experiment directory.

The challenge for SEAGrid is that the data may come from a desktop client as well as a browser upload. A purely web-based gateway does not need to expose the data store directly to the user since all file system operations are performed by the gateway server, and the authentication mechanisms discussed in the previous section are sufficient to secure the browser client.

Desktop clients however need to interact directly with the data store.

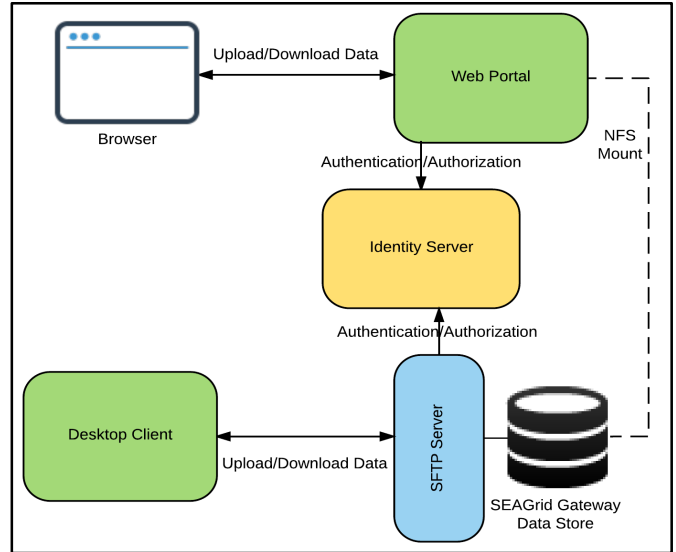


Figure 5 Data Management Infrastructure in SEAGrid

We thus needed to implement a service that integrates with WSO2 IS authentication from the desktop client that is geared toward efficient file transfer, and that restricts access for different users to different sections of the data store directory tree without assuming the users have accounts on the gateway server. Our solution is to expose the gateway data store using a secure file transfer protocol (SFTP) server. For this we use the Apache Mina library to develop a customized SFTP server. Apache Mina has a pluggable authentication mechanism that we extended to support WSO2 IS-based username/password authentication. Mina also allows us to define dynamic virtual file systems that restrict the user to specific areas of the data store. A high level architecture of the SEAGrid data management infrastructure is shown in Figure 5.

5. EXTENDING AIRAVATA WITH DATA EXTRACTORS

SEAGrid allows users to execute a wide range of scientific applications and download the resulting raw outputs. Additionally, we are building capabilities that allow users to manage their experimental results within the gateway and gain more insights into the data without having to download. We summarize here initial implementation efforts. We are applying these new capabilities to SEAGrid’s archives as well as to new experiments; SEAGrid has over 8TB of accumulated data since 2006. One desirable capability is to support domain-specific search over data using properties or attributes contained in the data. To provide this capability, we have added automated data

post processing pipelines to Apache Airavata for the SEAGrid gateway that parse the raw outputs generated to extract metadata and important attributes. These metadata are then indexed for searching and querying services. Airavata’s component-based architecture and message-based communications help us to integrate this new data cataloging system with other Airavata components. Our implementation uses Airavata’s message bus [10] to listen to events and the Airavata Registry to get metadata relating to experiments. The parsing is done using an extensible pluggable parsing framework that can be used to add new parsers. The current implementation uses parsers for Gaussian, GAMESS, NWChem and Molpro. Figure 6 shows the high level architecture of the data cataloging system.

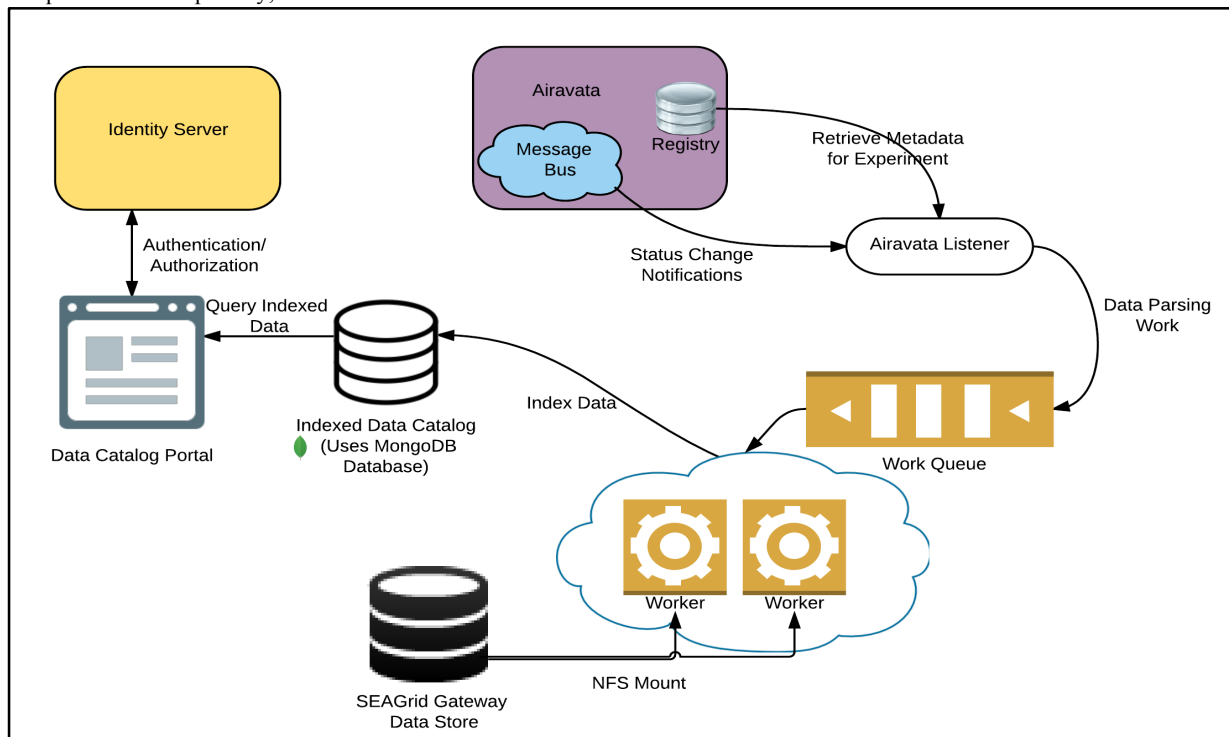


Figure 6 Architecture of the SEAGrid Data Cataloging System

Airavata uses messaging to notify its components of state changes, such as “inputs staged”, “job queued”, “job running”, “job complete”, and “output staged”. The data cataloging system relies on the Airavata message bus to get notifications on completion of experiments. Experiments are complete when the data has been staged to SEAGrid data store, which results in an “experiment completed” event posted to the message bus. Once an Airavata Listener component receives an “experiment completed” notification, it queries the Airavata Registry to retrieve metadata relating to that experiment such as the application and the location of the output data. After that, it creates a “data parsing” work message and puts this message into a work queue. Multiple workers are deployed; each worker takes work from the work queue and acts on it. These workers require access to the gateway data store, which we provide using a network file server (NFS) mount; other mechanisms such as transferring data from the store to a processing host are possible but add overhead. The workers do the actual parsing of the data and extract metadata and important attributes. The extracted data are formatted into key-value pairs in JSON format and are then

indexed in a MongoDB database. A browser interface is provided to enable users to search and query over this indexed data.

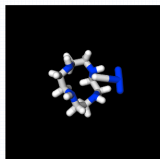
The parsing in workers is done using an extensible parsing framework that we developed in Java. New data parsers can be added to the system by providing jars and registering the parsers in a configuration file. Knowledge of how to parse the generated data will come from domain experts, who may prefer languages such as Perl and Python and will use natively built libraries in the parsing process. In order to unify these differences between parsers and to provide a unified architecture for easy deployment, we wrap the parsers using Docker containers [13] that are published to DockerHub (<https://hub.docker.com/>). These docker containers are equipped with all the natively built library requirements and the parsing scripts that will read the data and return the output in the key-value formatted JSON file format. These docker containers can be then invoked by the parser framework. For example, the current Gaussian data parser uses three different libraries (OpenBabel [14], CCLIB [15], and CCDBT [16]) to extract data. These libraries are built inside a

docker container and are wrapped using a python script for easy access from the parser framework.

In addition to indexing newly generated data, we have indexed the data archives that were generated from the CCG gateway. To reuse the same data cataloging infrastructure, we devised a file system scanner that can scan the data archive and populate the data cataloging work queue with work messages. It is likely that new data parsers for specific applications will be added to the parser framework over time, so archived data will need to be parsed again and reindexed. In such a scenario, we can add a new component that will scan through the registry and populate the work queue for the data products that needs to be reindexed.

SEAGrid Data Catalog Search Directory Browser ramasami

N.B: This data is automatically extracted using set of configured parser and may contain errors. Please report any issues in the [issue tracker](#)

Organization		 <p>Final Molecular Structure</p>
Experiment	5ca_2a_bp_e_2_comet.xsede.org.556613.150520	
Project	ramasami_pnj	
Owner	ramasami	
Indexed Time	2016-03-11 17:34:10	
Molecule		
Formula	C8H20CaN7	
Number of Atoms	36	
Electron Symmetry	1-A	
Multiplicity	1	
Charge	0	
Orbital Symmetry	Occupied A A A A A A A	
Identifiers		
InChI	InChI=1S/C8H20N4.Cs.N3/c1-2-10-5-6-12-8-7-11-4-3-9-1/-1-3-2/n9-12h,1-8h2/q+1;-1	
InChI Key	DHDIHPIAEPRCFT-UHFFFAOYSA-N	
SMILES	C1NCCNCCNCCN1.[N]([N])([N])C	
Canonical SMILES	N1CCHCCNCCNCC1.[N]([N])([N])	
Calculation		

Proceedings 00 Proceedings 00

Figure 7. Data Catalog - Extracted Metadata

Figure 7 shows a screen shot example of data extracted from a completed experiment. Fields include the molecular formula, number of atoms, electronic symmetry, and InChI and SMILES identifiers. The structure is also rendered.

6. RELATED WORK

Proceedings of the Gateway Computing Environments workshops, the International Workshop on Science Gateways series [17] [18] and the XSEDE Annual Conference provide snapshots of science gateway community activities. Science gateway development and hosting environments similar to the approach described here include CyVerse's Agave [19], HubZERO [20], and WS-PGRADE [21]. In addition to bioinformatics pipelines, Galaxy software has been used to support gateways in other fields [22].

7. CONCLUSIONS AND FUTURE WORK

This paper has described the technology refresh of the SEAGrid science gateway. This involved a rewrite of SEAGrid's desktop client, the development of a new web client, integration with Apache Airavata and WSO2 IS, and the extension of Apache Airavata to support custom data parsing and extraction tools specific to SEAGrid applications. Simultaneously supporting web and desktop clients introduced authentication and access control issues; in particular, we aimed to unify the login process so that users can move seamlessly between the two interfaces. We needed to ensure that regular users with full access to the Airavata API in the desktop client do not have access to administrator-restricted methods. These requirements were met by integrating WSO2 IS into the SEAGrid infrastructure; this provides an OAuth2 endpoint for password-based authentication and also XACML-based role definitions.

To streamline data uploads and downloads seamlessly between the web and desktop interfaces, Apache Mina was used. It provided a lightweight SFTP server with pluggable authentication and the ability to create dynamic virtual file systems. The SEAGrid desktop client can thus access the SEAGrid data store through a standard SFTP client using WSO2 IS-provided authentication. We are also able to use Apache Mina to restrict the user to specific areas of the file system-based SEAGrid data store. Finally, we demonstrated how SEAGrid benefits from Apache Airavata's extensibility by describing the development of custom data parsing and extraction services that allow users to search input and output files. This custom extension leverages Apache Airavata's AMQP-based message system: adding a new component is accomplished by adding a new subscriber to the message queue to receive "job completed" events; other Apache Airavata components do not need to be modified.

Future work includes the investigation of other data storage and transfer technologies. Currently, sharing of the data in the gateway datastore between users is not supported. Additional work is being done to integrate the SEAGrid desktop and web clients with the Globus Transfer service for large file transfers. At the time of writing, this is an active XSEDE Extended Collaborative Support effort between the SEAGrid and Globus teams. The Globus Transfer case is particularly useful for large data transfers and eliminates the need for two-stage file transfers. The implementation requires Globus Sharing to be enabled on the target XSEDE resource if transferring to community accounts. iRODS is also an appropriate tool for many of the problems discussed in this section and is being evaluated. The parsing and data extraction framework provide an important model for Apache Airavata to handle smaller executions that do not need to go through XSEDE supercomputers. Complementary investigations using the Apache Mesos framework to schedule single-node SEAGrid jobs on Jetstream and potentially other XSEDE resources is ongoing.

8. ACKNOWLEDGEMENTS

This work was partially supported by NSF ACI award #1339774, "Collaborative Research: SI2-SSI: Open Gateway Computing Environments Science Gateways Platform as a Service (OGCE SciGaP)". This work used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number ACI-1053575. We acknowledge Hasini Gunasinghe for initial integration of WSO2 services in Airavata and Dimuthu Upeksha for initial work on SEAGrid desktop client supported through a GSOC-2014.

9. REFERENCES

- [1] Dooley, Rion, Kent Milfeld, Chona Guiang, Sudhakar Pamidighantam, and Gabrielle Allen. "From proposal to production: Lessons learned developing the computational chemistry grid cyberinfrastructure." *Journal of Grid Computing* 4, no. 2 (2006): 195-208.
- [2] Shen, Ning, Ye Fan, and Sudhakar Pamidighantam. "E-science infrastructures for molecular modeling and parametrization." *Journal of Computational Science* 5, no. 4 (2014): 576-589.
- [3] Miller, Mark A., Wayne Pfeiffer, and Terri Schwartz. "The CIPRES science gateway: enabling high-impact science for

- phylogenetics researchers with limited resources." In Proceedings of the 1st Conference of the Extreme Science and Engineering Discovery Environment: Bridging from the eXtreme to the campus and beyond, p. 39. ACM, 2012.
- [4] Blankenberg, Daniel, Gregory Von Kuster, Nathaniel Coraor, Guruprasad Ananda, Ross Lazarus, Mary Mangan, Anton Nekrutenko, and James Taylor. "Galaxy: a web-based genome analysis tool for experimentalists." *Current protocols in molecular biology* (2010): 19-10.
 - [5] Smith, Warren, Sudhakar Pamidighantam, and John-Paul Navarro. "Publishing and consuming GLUE v2. 0 resource information in XSEDE." In Proceedings of the 2015 XSEDE Conference: Scientific Advancements Enabled by Enhanced Cyberinfrastructure, p. 25. ACM, 2015.
 - [6] Fan, Ye, Sudhakar Pamidighantam, and Warren Smith. "Incorporating Job Predictions into the SEAGrid Science Gateway." In Proceedings of the 2014 Annual Conference on Extreme Science and Engineering Discovery Environment, p. 57. ACM, 2014.
 - [7] Coleman, Shawn P., Sudhakar Pamidighantam, Mark Van Moer, Yang Wang, Lars Koesterke, and Douglas E. Spearot. "Performance improvement and workflow development of virtual diffraction calculations." In Proceedings of the 2014 Annual Conference on Extreme Science and Engineering Discovery Environment, p. 61. ACM, 2014.
 - [8] Pierce, Marlon E., Suresh Marru, Lahiru Gunathilake, Don Kushan Wijeratne, Raminder Singh, Chathuri Wimalasena, Shameera Ratnayaka, and Sudhakar Pamidighantam. "Apache Airavata: design and directions of a science gateway framework." *Concurrency and Computation: Practice and Experience* 27, no. 16 (2015): 4282-4291.
 - [9] Pierce, Marlon, Suresh Marru, Borries Demeler, Raminderjeet Singh, and Gary Gorbet. "The apache airavata application programming interface: overview and evaluation with the UltraScan science gateway." In Proceedings of the 9th Gateway Computing Environments Workshop, pp. 25-29. IEEE Press, 2014.
 - [10] Marru, Suresh, Marlon Pierce, Sudhakar Pamidighantam, and Chathuri Wimalasena. "Apache Airavata as a laboratory: architecture and case study for component-based gateway middleware." In Proceedings of the 1st Workshop on The Science of Cyberinfrastructure: Research, Experience, Applications and Models, pp. 19-26. ACM, 2015.
 - [11] Heiland, Randy, Scott Koranda, Suresh Marru, Marlon Pierce, and Von Welch. "Authentication and Authorization Considerations for a Multi-tenant Service." In Proceedings of the 1st Workshop on The Science of Cyberinfrastructure: Research, Experience, Applications and Models, pp. 29-35. ACM, 2015.
 - [12] Hardt, D. "RFC6749-The OAuth 2.0 Authorization Framework. Oct. 2012." URL: <https://tools.ietf.org/html/rfc6749> (visited on 04/24/2015).
 - [13] Merkel, Dirk. "Docker: lightweight linux containers for consistent development and deployment." *Linux Journal* 2014, no. 239 (2014): 2.
 - [14] O'Boyle, Noel M., Michael Banck, Craig A. James, Chris Morley, Tim Vandermeersch, and Geoffrey R. Hutchison. "Open Babel: An open chemical toolbox." *J Cheminf* 3 (2011): 33.
 - [15] O'Boyle, Noel M., Adam L. Tenderholt, and Karol M. Langner. "Cclib: a library for package-independent computational chemistry algorithms." *Journal of computational chemistry* 29, no. 5 (2008): 839-845.
 - [16] Chen, Mingyang, Amanda C. Stott, Shenggang Li, and David A. Dixon. "Construction of a robust, large-scale, collaborative database for raw data in computational chemistry: The Collaborative Chemistry Database Tool (CCDBT)." *Journal of Molecular Graphics and Modelling* 34 (2012): 67-75.
 - [17] Olabarriaga, Silvia Delgado, and Nancy Wilkins-Diehr. "GCE15 Special Issue Conference Publications." *Concurrency and Computation: Practice and Experience* (2015).
 - [18] Gesing, Sandra, and Nancy Wilkins-Diehr. "Science gateway workshops 2014 special issue conference publications." *Concurrency and Computation: Practice and Experience* 27, no. 16 (2015): 4247-4251.
 - [19] Dooley, Rion, Matthew Vaughn, Dan Stanzione, Steve Terry, and Edwin Skidmore. "Software-as-a-service: the iPlant foundation API." In 5th IEEE Workshop on Many-Task Computing on Grids and Supercomputers (MTAGS). 2012.
 - [20] McLennan, Michael, and Rick Kennell. "HUBzero: a platform for dissemination and collaboration in computational science and engineering." *Computing in Science & Engineering* 12, no. 2 (2010): 48-53.
 - [21] Kacsuk, Peter, Zoltan Farkas, Miklos Kozlovsky, Gabor Hermann, Akos Balasko, Krisztian Karoczka, and Istvan Marton. "WS-PGRADE/gUSE generic DCI gateway framework for a large variety of user communities." *Journal of Grid Computing* 10, no. 4 (2012): 601-630.
 - [22] Walker, Mark A., Ravi Madduri, Alex Rodriguez, Joseph L. Greenstein, and Raimond L. Winslow. "Models and Simulations as a Service: Exploring the Use of Galaxy for Delivering Computational Models." *Biophysical journal* 110, no. 5 (2016): 1038-10.