Security on the US Fusion Grid

J.R. Burruss[a], T.W. Fredian[b], M.R. Thompson[c]

[a]*General Atomics, P.O. Box 85608, San Diego, California 92186-5608*
[b]*Massachusetts Institute of Technology, Cambridge, Massachusetts*
[c]*Lawrence Berkeley National Laboratory, Berkeley, California*

Abstract

The National Fusion Collaboratory project is developing and deploying new distributed computing and remote collaboration technologies with the goal of advancing magnetic fusion energy research. This work has led to the development of the US Fusion Grid (FusionGrid), a computational grid composed of collaborative, compute, and data resources from the three large US fusion research facilities and with users both in the US and in Europe. Critical to the development of FusionGrid was the creation and deployment of technologies to ensure security in a heterogeneous environment. These solutions to the problems of authentication, authorization, data transfer, and secure data storage, as well as the lessons learned during the development of these solutions, may be applied outside of FusionGrid and scale to future computing infrastructures such as those for next-generation devices like ITER.

Keywords: security, FusionGrid, grid computing

1. Introduction

Critical to the success of any computational grid is security. Effective sharing of data and codes across the Internet requires a reliable means of identification, control over resources, and secure data transfer and storage. New security technologies are being deployed to improve security for the US Fusion Grid (FusionGrid) [1]. Collaboratory workers have adapted secure communication and authentication technologies from the Globus Security Infrastructure (GSI) [2] to the MDSplus scientific data management system [3] to create a GSI-enabled version of MDSplus. GSI is an extension to the Transport Layer Security (TLS) protocol in wide use for secure Web transactions; it provides secure and mutually authenticated communication using proven encryption protocols and X.509 public key certificates to globally identify individuals. GSI is one component of the larger Globus Toolkit™ suite of software used for building grids. A centralized credential manager based on the grid middleware MyProxy [4] server was deployed to provide the fusion scientists a convenient and secure way to manage their public key credentials. These components create a data management system capable of robust authentication and secure data transfer in a computational grid.

An authorization system appropriate for computational grids was developed to meet the security needs of computational resource stakeholders such as site security staff, systems administrators, and the scientists that develop, share, and maintain the codes and data made available through FusionGrid services. This system, known as the Resource Oriented Authorization Manager (ROAM) [5], consists of an authorization database and easy-to-use web interface. ROAM works with GSI-enabled MDSplus or any client capable of communicating via HTTPS, and is general enough to be applied to other distributed computational environments.

GSI-enabled MDSplus allows for secure data transfer and storage on FusionGrid. This more secure version of the popular fusion data acquisition and storage system is aware of X.509 identity credentials [6] and can be configured to provide increased security by encrypting data sent over the network.

2. Authentication

The most fundamental challenge to security in a computational grid is to identify the users of the grid. As Fig. 1 illustrates, the task of individually authenticating with the multiple geographically and administratively dispersed resources of a computational grid is time-consuming and confusing. Users often forget the different username and password combinations required for each resource. Administra-

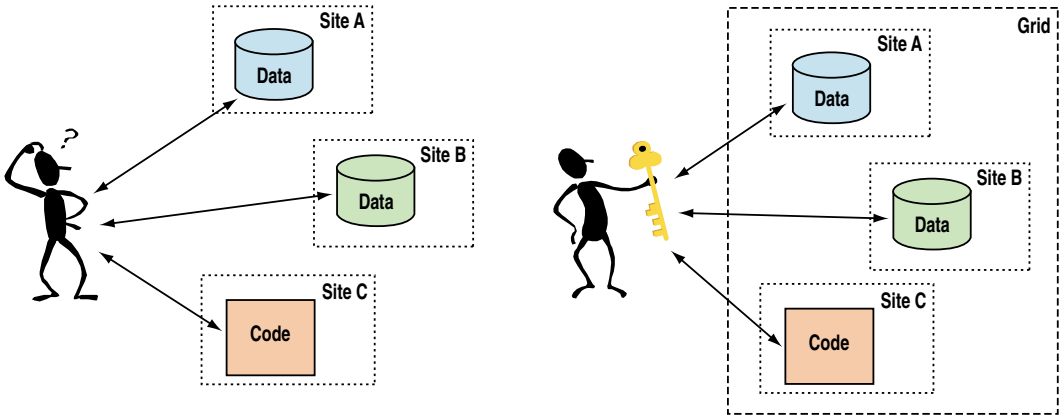

Fig. 1. (left) A scientist without X.509 credentials struggles to individually authenticate using a different username and password with different resources (right) A scientist uses a single X.509 credential to authenticate with each resource on a grid.

tors of different resources are limited to their own systems and, lacking an overall picture, have no way to link a user account at a remote site with an account for the same user at another site. FusionGrid developers elected to use X.509 credentials and GSI as mechanisms to uniquely identify users in a virtual organization. X.509 credentials have the advantage of identifying the user not by host-based means (e.g. *username@hostname*), but by a globally unique distinguished name. Not only is this more secure, but it allows users to identify themselves even when working from computers with dynamic or non-routable IP addresses, as is the case when working on a laptop in a coffee shop or an airport. FusionGrid scientists are each uniquely identified by their X.509 credentials, which are used to authenticate with every FusionGrid resource. Because X.509 credentials are unique, administrators can see when offsite connections are really coming from a known user working remotely.

GSI also brings to the table a means with which to implement single sign-on. Instead of requiring that a user individually authenticate with each resource, the user signs in to the grid a single time and creates a proxy which can be used to authenticate on the behalf of the user and includes a mechanism to further delegate to other processes. For example, a user can sign on and launch a code, which then retrieves the data it needs after authenticating using the delegated credential of that user (Fig. 2).

Early experiences with self-management of X.509 credentials demonstrated that, while the technology works well, it is difficult for most users to keep track of their certificate files. The process of exporting and importing credentials from their Web browsers, converting the credentials to the required formats, and installing their credentials in the appropriate place with the correct file permissions in their various home directories was a source of endless frustration to scientists. Ultimately this problem was ameliorated by removing the burden of credential management from the scientist altogether.

FusionGrid developers created a credential manager which consists of a MyProxy server, a Secure Sockets Layer (SSL)-enabled Apache Web server [7], and a custom Web interface, all of which run on a dedicated and secured host. The Web interface is used by scientists to register with FusionGrid to receive a credential, to change their password, or request their password hint in case they have forgotten their password. Users access their credentials using a username and password, a time-tested security mechanism understood by all users and requiring no training to use. Figure 3 illustrates the use of MyProxy — a scientist simply signs on with a username and password before using a code, which then can retrieve a delegated copy of user credentials and authenticate with other grid resources.

To use X.509 credentials it is necessary to run a Certificate Authority (CA) to create credentials. By definition, a certificate — the public half of a credential — is a public key and appropriate metadata digitally signed by a Certificate Authority. The FusionGrid CA [8] is used to issue new FusionGrid credentials. Rather than manage this CA on their own, FusionGrid developers outsourced the task to ESnet, an organization with a CA infrastructure already in place. FusionGrid hosts, the FusionGrid MDSplus install package used for data storage and transfer, and the Access Grid software package used for remote participation on FusionGrid were each updated to accept credentials from this new CA.

Taken together, the use of X.509 credentials and the FusionGrid credential manager have solved the difficult problem of identifying users on FusionGrid. However, complete access to computing resources requires more than just identification of users. Some system of access control to authorize usage of FusionGrid services is required for effective security.

3. Authorization

After authentication, the second most fundamental challenge to security in a computational grid is authorization. Once the identity of a user can be established, it is then necessary to determine the permission of that user to use a particular resource. Stakeholders of the various computing resources must be empowered to control access to their resources.

Managing access control can be a confusing task (Fig. 4). For example, to use a particular computational service, a user might need to get authorized by the author of the code, by the site security administrator of the site that hosts the code, by the database administrator for a database that holds data needed by the code, and by the site security administrator for the site that hosts the database. This is problematic not only for scientists, who must wade through the maze of authorizations, but for administrators who are not empowered to see the bigger security picture due to the lack of a coherent security model for the entire grid. The default Globus authorization mechanism is to use "grid mapfiles" — text files that map X.509 distinguished names to local accounts — which must be installed and maintained on each participating host. Thus, even after authorizations have been put in place, it is not easy for one administrator to quickly view authorizations for a single user over all resources because that information is distributed onto different hosts at different sites.

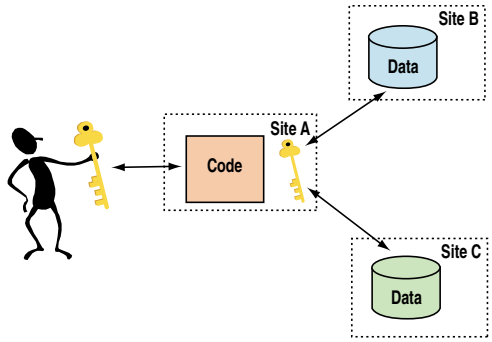

Fig. 2. Delegation allows a code, acting on behalf of the scientist, to authenticate using delegated credentials with each resource without user intervention.

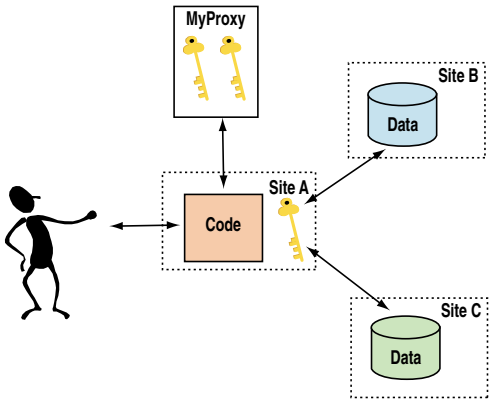

Fig. 3. A scientist with credentials stored in MyProxy signs on using a username and password; this retrieves delegated credentials which the code uses to authenticate with each resource without user intervention.

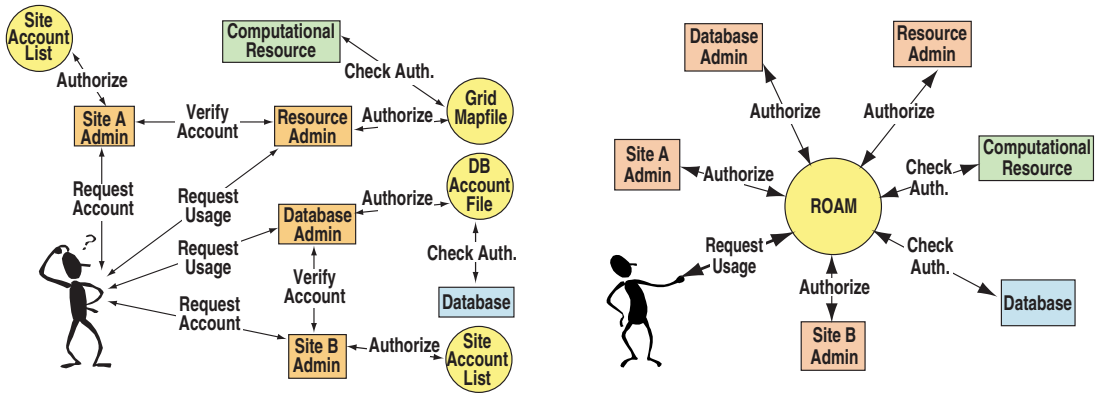

Fig. 4. (left) A scientist struggles to get authorized by the various stakeholders to use a resource (right) A scientists gets authorized to use a resource through ROAM.

FusionGrid architects set about to fix this problem by creating a coherent authorization model, by centralizing information and by simplifying the process of requesting and granting authorizations. The result is the Resource Oriented Authorization Manager (ROAM). Under ROAM, the process of requesting, viewing, and granting authorization is greatly simplified.

The ROAM information model consists of a framework of *resources*, *permissions*, *users*, and *authorizations*. Everything in the ROAM universe is one of these four types of things. A resource is typically a grid service, but it can also be an entire site, like the Alcator C-Mod or DIII-D fusion experiments. A user is any uniquely identified consumer of resources. When the FusionGrid credential manager loads a new credential into the MyProxy server, it enters the user name, the distinguished name, and other user information into ROAM. Thus, ROAM always has the list of all FusionGrid users.

Permission is a type of usage for a resource; in other words, a permission is a way in which a resource is used, for example "read" and "write" for a database, "access" for a site, or "execute" for a code. An authorization is a grant of a specific permission for a particular user on a specified resource. Authorizations may include contexts, which can be used to specify conditions or obligations that need to be met when exercising the permission. Presently, context is only used to specify the local user id and/or group id under which an action should be performed. The key to the ROAM model is this: clearly indicate resources, define stakeholders for those resources, and empower stakeholders to control access to their resources.

Users and administrators interact with ROAM through a Web interface. Programs consult ROAM through HTTP/HTTPS and make policy decisions based on the authorization information stored in ROAM. It is important to note that under the ROAM system, it is the resources that are empowered to make authorization decisions. ROAM holds the authorization information used by resources to make those decisions.

FusionGrid developers looked at several other technologies to improve upon the basic authorization capabilities of grid mapfiles before designing and implementing ROAM. Initial efforts to improve authorization in FusionGrid focused on the Akenti authorization system [9]. Akenti is an authorization system designed to handle distributed resources controlled by multiple stakeholders and allows for fine grain access control based on the code to be run and the job parameters the user requested. It supplemented the coarse grain admission control provided by the grid mapfile and allowed sites to closely control what grid users could do. Akenti was used with success in the very first FusionGrid computational service [10]. However, the Akenti implementation of authorization information as distributed, digitally signed documents made some of the desired access management operations difficult. FusionGrid developers also looked at the Community Authorization Server [11] and the Virtual Organization Membership Service [12]. However, both of these systems implement the

push model of authorization, where the user must first contact an authorization server to get credentials in addition to their X.509 credentials and then present them to the resource provider. The provider must then verify the additional credentials and often do additional local authorization checks. This was a problem because FusionGrid architects wanted to keep the authorization path as simple as possible, both for scientists and for developers. Furthermore, both of these designs focus on defining users and their attributes or permissions; they do not attempt to define the resources of a virtual organization, something that was desired for FusionGrid. Any system that would empower stakeholders to control access to their resources must first define those resources.

4. Data Transfer and Storage

FusionGrid developers chose to build upon the success of the MDSplus system and develop a secure version appropriate for use in computational grids. The new GSI-enabled MDSplus system uses X.509-based authentication and encryption to provide secure access to fusion data. Users with credentials or programs with delegated credentials acting on behalf of a user can now connect to MDSplus to read and write data. To enable access from Mac OS X, a lightweight version of GSI was ported by FusionGrid developers to that platform.

Basic MDSplus uses a simple protocol layered on TCP for data exchange. The security model for this data exchange is based on a simple user mapping similar to the original Berkeley rhosts mechanism. The client sends the name of the remote user and the server then uses the username and the IP address of the connecting client to look up a mapping to a local account. A text file is used by basic MDSplus for this mapping of connecting clients to local accounts, much like the grid mapfile in GSI except that MDSplus allows for wildcard account mapping.

The new GSI-enabled MDSplus uses X.509 credentials for mutual authentication: clients can confirm that the server is authentic, and the server can confirm the identity of connecting clients. For authorization MDSplus can be configured to use either ROAM or a text file. When using ROAM, GSI-enabled MDSplus uses the context feature to map clients to local accounts. Administrators configure authorization logic by editing functions written in the Tree Data Interface (TDI) expression language used by MDSplus. Since TDI is a structured language, administrators can code complex authorization logic into their functions. In practice, authorization policies and the logic required to implement those policies tend to be very simple, requiring only a few lines of TDI code. GSI-enabled MDSplus may also be configured to use a primary and secondary ROAM (analogous to primary and secondary DNS servers) to increase reliability. FusionGrid is currently configured to run both a primary ROAM server and a secondary ROAM server. The secondary server is run in a read-only mode and is kept in synch with the primary server on a nightly basis. The secondary server serves as a backup in case the primary server needs to be taken down for maintenance. If it were ever the case that the primary server would be down for a longer duration, the secondary server could be made writeable and designated as the new primary server.

GSI-enabled MDSplus is fully compatible with regular MDSplus. The data files are interchangeable; in fact, the same host could serve data files through both types of servers simultaneously, something that might be useful if for example it was determined that host-based authentication was sufficient for local access, but X.509 authentication was required for offsite access.

5. Outcome

The new credential manager and ROAM were deployed in Fall 2004. The credential manager has been used to create new FusionGrid credentials for 24 users. The password hint feature has also been used. ROAM is being used routinely and to date has processed over 13,000 authorization queries with a peak load of 854 authorization queries per hour. GSI-enabled MDSplus is in use at the Alcator C-Mod and DIII-D fusion facilities and is being used by both the GATO [13] and TRANSP [14] FusionGrid services.

ROAM is being used to implement the two-rule authorization policy for the FusionGrid GATO service. Users of the GATO service must sign one form to get permission from the author of GATO to use the code, and another form to get permission from DIII-D site security to use the computational resources (i.e. the network and computers) on which GATO runs. This two-rule policy was implemented by creating two resources — "GATO" and "D3D" — and building into the GATO service an authorization query to confirm that users have appropriate permissions for both resources. This has satisfied both stakeholders: the author of GATO can control access by granting or revoking the "execute" permission to the GATO resource, and the DIII-D site security staff can control

access to DIII-D resources by granting or revoking the "access" permission on the D3D resource.

6. Discussion

ROAM, the credential manager, and GSI-enabled MDSplus have been deployed for use with FusionGrid and meet the current security needs of scientists and administrators. The question remains as to how well this infrastructure, which works perfectly well for the current user base, would scale to ITER.

For authentication and authorization, there are four scaling dimensions that must be considered: credential creation, proxy delegation, authorization management, and authorization decisions. Since credential creation (or renewal) takes place once every two years for every user, system load is minimal provided that not all certificates are created or renewed on the same date. Proxy delegation occurs only about once a day for each user accessing the Grid; any congestion problems could be alleviated by using mirrored servers similar to what is now being used for robustness. Authorization management interactions increase in relation to the number of users and number of resources. Most authorization changes occur only when users and resources join or leave the system, so system load is also minimal. Authorization decisions are made each time a user requests access to a resource, and require database queries. This potential bottleneck will be tested by deploying ROAM for use with the popular TRANSP service, thus increasing by about an order of magnitude both the number of users in the ROAM database and the number of authorization queries sent to ROAM. It is expected that indexing of the database and replications of the ROAM server should allow the authorization decisions to scale.

The current implementation of GSI-enabled MDSplus should scale to the pulsed mode operational phase of ITER. Extensions to MDSplus that will make it capable of handling the long pulse operational phase of ITER by allowing for real-time access are presently being investigated. Specifically, the idea is to allow for access while the data is being written during the shot as opposed to waiting until the end of the shot, then writing data. This and other ideas for MDSplus are promising, but more work is needed to produce working prototypes for evaluation.

Acknowledgments

This work was funded by the SciDAC project, US Department of Energy under DE-FG02-01ER25455, DE-FC02-04ER54698, and by the Director, Office of Science, Office of Advanced Science, Mathematical, Information and Computation Sciences of the US Department of Energy under DE-AC03-76SF00098.

References

[1] D.P. Schissel, et al., "Building the US National Fusion Grid: Results from the National Fusion Collaboratory Project," Fusion Eng. and Design **71**, 245 (2004).

[2] I. Foster, et al., "A Security Architecture for Computational Grids," *Proc. 5th ACM Conf. on Computer and Communications Security Conf.* 83, (1998).

[3] T.W. Fredian and J.A. Stillerman, "MDSplus: Current Developments and Future Directions," Fusion Eng. and Design **60**, 229 (2002).

[4] J. Novotny, et al., "An Online Credential Repository for the Grid: MyProxy," *Proc. 10th Int. Symp. on High Performance Distributed Computing* (2001).

[5] ROAM, https://roam.fusiongrid.org/, accessed May 17, 2005.

[6] R. Housley, et al., "Internet X.509 Public Key Infrastructure Certificate and CRL Profile," RFC 2459, http://www.ietf.org/rfc/rfc3280.txt, (2002).

[7] Apache-SSL, http://www.apache-ssl.org/, accessed May 17, 2005.

[8] FusionGrid Credential Manager, https://cert.fusiongrid.org/, accessed May 17, 2005.

[9] M. Thompson, et al., "Certificate-based Authorization Policy in a PKI Environment," ACM Transactions on Information and System Security (TISSEC) **6**, 566 (2003).

[10] J.R. Burruss, et al., "Remote Computing Using the National Fusion Grid," Fusion Eng. and Design **71**, 251 (2004).

[11] L. Pearlman, et al., "A Community Authorization Service for Group Collaboration," *Proc. of the IEEE 3rd Int. Workshop on Policies for Distributed Systems and Networks* (2002).

[12] R. Alfieri, et al., "VOMS: an Authorization System for Virtual Organizations," (presented at the 1st European Across Grids Conf., Santiago de Compostela, 2003).

[13] L.C. Bernard, F.J. Helton, and R.W. Moore, "GATO: An MHD Stability Code for Axisymmetric Plasmas with Internal Separatrices." Computer Phys. Communications **24**, 377 (1981).

[14] TRANSP, http://w3.pppl.gov/transp, accessed May 17, 2005.