

# Building a unified grid, Part 3: Components for security, authentication, and authorization

End-to-end authentication and authorization

Level: Introductory

[Abel W. Lin \(awlin@ncmir.ucsd.edu\)](mailto:awlin@ncmir.ucsd.edu), Software Architect, National Center for Microscopy and Imaging Research  
[Kurt Mueller \(kurt@sdsc.edu\)](mailto:kurt@sdsc.edu), Software Engineer, San Diego Supercomputer Center (SDSC)

31 Jan 2006

What does it mean to "grid-enable" an end-to-end process? In this "[Building a unified grid](#)" series, grid architects in the Telescience Project at the National Center for Microscopy and Imaging Research show us how an infrastructure like theirs can guide developers to a more unified grid. [Part 1](#) described their grid-based system architecture. [Part 2](#) showed how their unified grid computes more data faster and unifies resources in an on-demand fashion to collect *better* data faster. At the core of this unification is the ability to securely connect to the various distributed resources, and that's what the authors discuss now in Part 3.

## Security, authentication, and authorization in a grid

Grid-based or not, security is critical to any organization. "Security," however, is often used as a catch-all phrase that includes several individual components: physical security, authentication, and authorization. All have important and defined requirements. *Security* is necessary for privacy, integrity, and general protection for grid-based communication (for example, preventing network sniffing and man-in-the-middle attacks). *Authentication* is necessary for verifying the identity of a user, as well as for single sign-on and delegation capabilities. *Authorization* is used to determine the allowed actions of an authenticated user.

In [Part 1](#), we briefly described the Telescience ATOMIC Authentication and Authorization (TeleAuth) bundle. Here we'll describe the Grid Accounts Management Architecture (GAMA), which provides the core functionality of the TeleAuth component of ATOMIC.

---

## Grid security -- More than just handing out certificates

The de-facto standard for grid security is the Grid Security Infrastructure (GSI). *GSI* is a public key-based X.509 conforming system that relies on trusted third parties for signing user and host certificates. Typical usage models require that each user is assigned a user credential consisting of a public and a private key. Users generate *delegated proxy* certificates with short life spans that get passed from one component to another and form the basis of authentication, access control, and logging. Users can typically manage the credentials (and proxy) manually or use a number of command line-based utilities to manage their credentials.

Despite the general acceptance of GSI and its use over the course of many years, GSI-based security systems are known to be difficult for administrators to deploy and for users to use. Ironically, while security is a fundamental and critical component of all grid systems, until recently, there were no complete end-to-end security systems that worked out of the box. GAMA was created to address this missing functionality for all classes of users: administrators, applications developers, and end users.

With GAMA, end users never have to know anything about grid security, credentials, proxies, or other technical matters. They simply request an account using a typical Web form interface, and after the account is created, they log in to the portal using a familiar username/password combination. All the grid activity happens in the background -- from creation of grid credentials to retrieval of these credentials for use by portlets.

For application developers, the GAMA server components are wrapped in grid services that can be accessed by clients, in addition to portals. For instance, a simple GAMA login function can easily be added to existing stand-alone applications that provide access to grid-enabled resources and, therefore, require a user's grid credentials. This has already been demonstrated using Python and XUL (JavaScript) rich-client applications.

For the grid administrator, the GAMA server is installed using the Rocks cluster packaging system. A bootable CD installs the operating system (Linux®) and all necessary grid middleware security software packages. Even for an experienced grid administrator, installing and configuring the OS and all grid-security packages by hand can take several hours (or even days), but with our Rocks installer, the GAMA server is ready to go within an hour.

## Grid Accounts Management Architecture (GAMA)

Along with the Geosciences Network (GEON), the Telescience Project served as a driving application environment for the creation of GAMA. GAMA unifies a number of grid security components into a single tool, making grid security as easy to use as any commercial Web site while maintaining the security and delegation capabilities of GSI. Like other tools in Application to Middleware Interaction Component (ATOMIC), GAMA provides an appropriate simplified interface to both end users, and portal and application developers.

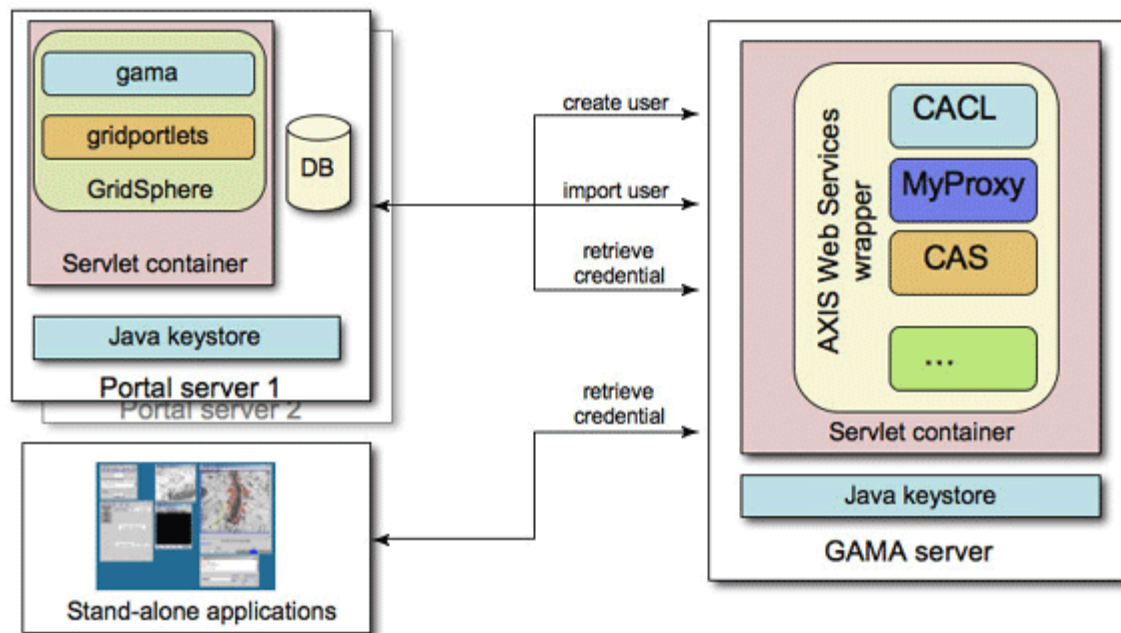
GAMA is a client-server system with a GridSphere portal that provides the GAMA client functionality to access a dedicated server machine running the GAMA server bundle. We will look at the server and client individually and see how they have unified previously disparate grid tools to interact and perform user management tasks. This article describes the GAMA V1.x architecture used in Telescience, as well as some changes under way for GAMA V2.0, which will increase robustness and scalability.

Figure 1 depicts the GAMA architecture; the GAMA server software stack is on the right. Currently, GAMA unifies a number of the most common grid components used for security (each of which were developed independently and in relative isolation):

- **CACL** -- Developed at the San Diego Supercomputer Center (SDSC), is a certificate authority system and is used to generate GSI credentials for users (and hosts).
- **MyProxy** -- Developed at National Center for Supercomputing Applications (NCSA), is a proxy repository used to store and generate limited-lifetime proxies derived from the user GSI credentials.
- **Community Authorization Service (CAS)** -- Developed by the Globus Project, CAS provides role-based authorization for communities of users through GSI mechanisms.

While these components are normally accessed individually and sequentially at the command line, GAMA unifies them as *Axis* Web services. This means that they can be accessed remotely through a well-defined and simple API by clients written in any language that supports Web services. For further security, Axis runs within the Apache Tomcat servlet container, which relies upon Java™ keystore mechanisms to negotiate communications with clients.

**Figure 1. GAMA architecture -- Clients on the left, GAMA server on the right**



On the client (portal) side, GAMA includes a set of GridSphere portlets and helper services integrated by portal developers into their portals (such as the Telescience Portal):

1. The account request portlet displays a form for new users to fill out and submit with their personal information and account preferences.
2. The account admin portlet is used by the portal administrator to view and manage the submitted user account requests.
3. The administrator chooses to approve or reject new account requests based on local site policies and can configure rules for auto-approval or auto-denial based on the domain name of the e-mail address. (For instance, the administrator may want all users from NCMIR to be approved automatically, so a rule would be added to auto-approve all users with "ncmir.ucsd.edu" in their e-mail addresses.)
4. Upon approval by the administrator (or triggering of an auto-approval rule), an e-mail message is sent to the user notifying him that he has been approved and providing an account activation Web link in the form of a URL.
5. When the user clicks on this link and proceeds with account activation on the portal, a sequence of steps creates a new user account:
  - The portal makes a call via Web services to the CACL service running on the GAMA server.
  - The CACL service generates a GSI certificate and key for the new user.
  - The portal instructs the GAMA server (again via Web services) to make the new certificate and key available through the MyProxy service on the GAMA server.
  - Then the portal creates a local GridSphere portal account for the user and configures the account to use the GAMA server for authentication during login.
6. At this point, the new user can log in to the portal with the username and password chosen during the account request and activation process.
7. When the user provides a username and password and clicks the Login button on the portal, the GridSphere framework uses a call to the MyProxy service on the GAMA server to authenticate the user.
8. If the supplied username and password are valid, the MyProxy service returns a limited-lifetime proxy credential derived from the user's credentials.
9. The proxy credential is activated in the GridSphere portal environment for the user, and the user is logged in to the portal.

Now, any grid-enabled portlet or service in the portal (such as a job submission portal or file transfer portal) can easily access the user's grid credentials and pass them on to grid resources and services.

CAS provides similar functionality to the MyProxy service in that it returns a GSI-based credential to the portal or

other client when presented with an appropriate username/password combination, but it enables more fine-grain authorization than the standard MyProxy credentials. A CAS credential includes authorization assertions that describe the user's role in a community and can be used by end resources to grant access accordingly. Currently, the GAMA server hosts a working CAS service, and we are actively developing portal-side tools for managing user roles through CAS.

---

## Conclusion

GAMA is a great solution for communities of users that are making the transition to grid computing and need a system that is relatively easy to install and maintain, and doesn't burden users with managing their own credentials. It simplifies and unifies processes that have typically been tedious, confusing, and error-prone.

GAMA V1.x, however, is currently tied to specific grid-based technologies (CAACL, MyProxy, and CAS) and provides limited mechanisms for extending or adding new GAMA services. We are working on GAMA V2, which will be built around a plug-in architecture to allow customization and expansion, and the chaining together of simple functions into more complex workflows (which may not be completely grid-based). For example, an implementer of GAMA may want to authenticate using the existing LDAP infrastructure. The new plug-in system will allow him to easily construct an LDAP GAMA module and combine it with other GAMA modules (such as MyProxy or CAS) to perform a multistep login sequence. In addition, GAMA V2 will better support multiple portals and will enable multiple local site administrators to act as GAMA administrators from their site -- all on the same GAMA server.

---

## Resources

### Learn

- Learn more about the [Telescience Project](#).
- Learn more about the [GEON project](#).
- Get more information about [Rocks Clusters](#).
- Learn more about research at the [National Center for Microscopy and Imaging Research \(NCMIR\)](#).
- Learn more about the [GridSphere framework](#).
- Learn more about the various grid software components at the [NSF Middleware Initiative](#).

### Get products and technologies

- Find out about downloading Telescience open source software used for deploying grids at the [Telescience Project](#).
  - Download a beta version of the [Telescience ATOMIC Toolkit](#).
  - [Download GAMA components](#).
- 

## About the authors

Abel W. Lin is the architect and technical lead for the Telescience Project at the National Center for Microscopy and Imaging Research. He has more than five years' experience applying grid and other computer science technologies to scientific processes. He designed and led the implementation of the first-generation proof-of-concept systems for the Telescience Portal and ATOMIC components of the Telescience Project, and is an interdisciplinary scientist with published research in biology and computer science. His interests include distributed systems architecture, software project management, and structural biology. Outside of the office, he is an avid reader, golfer, and bodysurfer.

Kurt Mueller is a researcher at the San Diego Supercomputer Center (SDSC) at the University of California, San Diego. He is the lead developer on the GAMA project. Kurt was involved in influential early Grid portal projects, including the GridPort toolkit and the NPACI and PACI HotPages, and he has collaborated with the Telescience team at NCMIR for many years. A cognitive scientist by training, Kurt's research interests include human-computer interaction, Grid security, and portal technologies.