Authorization Framework for Resource Sharing in Grid Environments

Jing Jin[1] and Gail-Joon Ahn[2]

[1] University of North Carolina at Charlotte
jjin@uncc.edu
[2] Arizona State University
gahn@asu.edu

Abstract. Grid data sharing services provide a unified platform for dynamic discovery, access and sharing of distributed data in Grid environments. A common authorization system is needed to provide access control for both Grid data sharing services as well as the data resources that are being shared through these services, accommodating different security requirements from the service providers and the data providers. In this paper, we present a flexible policy-driven authorization system, called RamarsAuthZ, for secure data sharing services in Grid environments. RamarsAuthZ adopts a flexible role-based approach with trust-aware feature to advocate originator control and provide unified access control both at the service level and at the data level.

1 Introduction

Data and resources in Grid environments are highly diverse in locations, types, structures, ownerships, naming conventions and access capabilities. The emergence of Grid data sharing services, such as the Globus Data Replication Service (DRS) [1], introduces a unified platform for data discovery, access and sharing transcending institutional boundaries. However, the data sharing facility provided to Grid clients requires more advanced access control mechanisms to accommodate various challenges ranging from the authorization model to the system architecture and deployment.

Firstly, Grid systems are usually composed of a number of dynamic and autonomous domains involving a large number of distributed users. An effective and manageable authorization scheme is necessary for the resource owners to control access and sharing of their resources. Attribute-based access control (ABAC), which makes access decisions relying on attributes of requesters and resources, has been widely adopted as a scalable and flexible authorization solution for highly distributed Grid environments [2]. In an attribute-based authorization system, the entity that manages user attributes is referred to as an Identity Provider (IdP). A user's attributes are normally collected by multiple IdPs in Grid environments. For example, a user is associated with a "home institution" which typically manages his employment status and affiliation attributes, while another IdP is associated with a Grid Virtual Organization (VO) that maintains attributes such as membership and role information within multiple domains. The authorization systems that support ABAC in Grid environments eventually need to be seamlessly integrated with all related IdPs and delivers users' attributes in a secure and trusted manner. Secondly, from the system architecture and deployment perspectives, there are a number of dimensions to be considered for an attribute-based authorization system. In terms of the attribute collection process, the "push" strategy requires the clients to obtain and push their attributes to the Grid service at the initial request. The "pull" strategy, on the other hand, does not require the clients to submit any attribute. Instead, the authorization system is responsible for acquiring attributes from the client's IdPs. While the clients have more options to select the attributes being released for authorization in the "push" mode, the "pull" mode simplifies the overall interception by the clients. It is impossible to determine which mode is more suitable for dynamic Grid environments. However, it is highly desirable for authorization systems be flexible enough to cope with both options. In terms of system deployment, the reliance on statically configured modules to render an authorization decision such as policy and attribute management should be minimized, as the authorization systems may serve for various Grid services running within the infrastructure. Finally, the data resources being shared through the Grid data sharing service normally belong to different institutions. These institutions, as the owners of the data resources, should directly participate in defining authorization policies for their data sets, and their authorizations need to be efficiently conveyed and effectively enforced within the Grid data sharing service. Meanwhile, the access and invocation of Grid data sharing service itself apparently need to be well protected to accommodate security requirements of its service provider. Therefore, it is required for authorization systems to be flexible enough to synthesize both service level and data level controls accommodating security policies from different stakeholders such as the data resource providers and the service providers.

There are continuous attempts to develop a common attribute-based authorization framework for Grids. Shibboleth [3] is an attribute authority service developed by the Internet2 community. A Shibboleth IdP asserts attributes about a user's home institution, and the relying parties can make access decisions based on these attributes. In VOMS [4], every Grid VO manages its own members, and a Grid user can access the resources by obtaining and presenting a credential that contains his VO membership to the resource. In [5], Grimm et al. proposed a "push" mode authorization system where Grid users are able to collect attributes both from his home institution as maintained in Shibboleth IdP and the Grid VO as maintained in VOMS, so that authorization decisions can be made based on the attributes from both sources. The Akenti system [6] represents the authorization policies for a resource as a set of certificates digitally signed by multiple distributed stakeholders. These certificates express the attributes a user must have in order to get specific rights to a resource. Akenti allows the certificates to be stored in distributed remote repositories and provides mechanisms based on the "pull" architecture to ensure that all applicable usage conditions are combined when making an access control decision. PERMIS [7] leverages the role-based access control and it has been recently integrated with Shibboleth as a "pull" mode system to retrieve the role attribute of a user. These authorization systems, however, rely on static configurations of their own policies and attribute providers, and cannot support dynamic policies and attribute discoveries to accommodate the above-mentioned requirements. In this paper, we present a flexible policy-driven authorization system, called RamarsAuthZ, for secure data sharing services in Grid environments. RamarsAuthZ adopts a flexible role-based approach with trust-aware feature to advocate originator control, delegation and dissemination control. A case study based on Globus DRS service is presented to provide effective access control both at the service level and at the data level. The rest of this paper is organized as follows. In Section 2, we introduce our proposed authorization framework and discuss the integrated RamarsAuthZ system design. Section 3 describes the performance evaluation of our RamarsAuthZ system. We conclude our paper with future research directions in Section 4.

2 Injecting Ramars Framework to Grid Environments

Role-based Access Management for Ad-hoc Resource Sharing framework (RAMARS) has been proposed as a policy-driven role-based access management solution for resource sharing in ad-hoc collaborative environments [8,9]. In RAMARS, the owner of the resource, also called an *originator*, has the ultimate authority over the resource that is being shared within the collaboration. An originator does not rely on any established security services to maintain membership and privileges. Instead, the originator defines its own sharing control domain by specifying a collection of collaborator roles and delegating fine-grained data sharing capabilities to these roles. Remote users are dynamically included in the originator's sharing control domain and authorized with certain access privileges by being assigned to the collaborator roles. Unlike the traditional RBAC that users are identified and assigned based on their identities, RAMARS introduces another layer of abstraction, where users are assigned to roles based on their attributes. In particular, an originator defines a set of attributes that a user must possess for the user to be assigned to a particular collaborator role. Remote users should present credentials to claim the possession of required attributes, yet it is up to the originator's discretion to validate and determine the trustworthiness of these credentials, thereafter to decide whether the claimants of certain attributes can be accepted for the role assignment. As there is no centralized trust base available in RAMARS, the delegation of authority is considered as an important mechanism for an originator to manage the degree of trust with different attribute authorities. For instance, an originator may place a higher level of trust for the "`citizenship`" attribute when the user presents his passport issued by US Department of State rather than a driver's license issued by a local DMV office.

As a policy-driven approach, RAMARS introduces a collection of policy components using standard XACML policy language [10] to realize the proposed authorization scheme. The policy set consists of the following components.

– Role-based Originator Authorization policy set (ROA) maintains the core authorization and trust management policies for an originator to govern

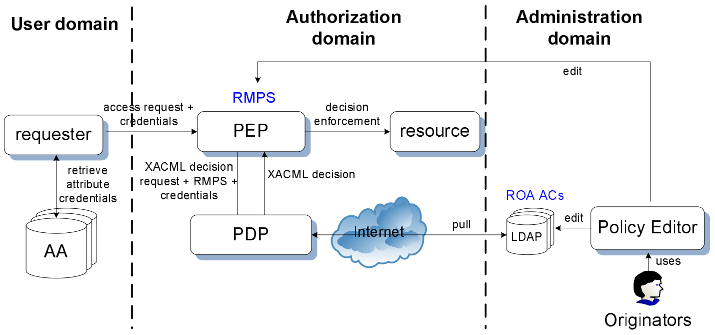

Fig. 1. RAMARS System Architecture

its control domain and delegate data sharing capabilities. The policy set contains the following subpolicies:

- Role Policy Set (RPS) defines a set of collaborator roles within an originator's control domain.
- Capability Policy Set (CPS) specifies the sharing capabilities assigned to each collaborator role.
- Role Assignment Policy Set (RAPS) defines the required attributes for a remote user to be assigned to a certain collaborator role.
- Trust Assessment Policy (TAP) defines internal policies to evaluate the trustworthiness of a user's attributes.

– Root Meta Policy Set (RMPS) is a light-weight top-level policy for an originator to declare the ownership of the data resource and specify the location of its ROA policy set. This enables the distributed policy deployment in RAMARS system as the authorization systems could dynamically locate and retrieve the ROA policies by utilizing the RMPS.

RAMARS system is designed to be deployed in distributed collaborative environments without assuming any centralized policy store. Different originators can edit and maintain their own ROA policies in their local administrative domains. The root RMPS for each respective data resource, however, should always be associated with the data resource for RAMARS system to locate its originator's ROA policies. Upon receiving a user's access request, RAMARS PEP invokes the PDP with a formulated XACML access decision request along with the root RMPS and the user's supportive credentials. RAMARS PDP then dynamically retrieves ROA policies from the originator's policy store based on the location reference specified in RMPS. The requester's credentials are evaluated against the originator's TAP policies where trusted attributes are derived. These trusted attributes are evaluated against RAPS policy to determine the user's roles. The access decision (e.g., *Permit* or *Deny*) is then determined based on the user's role. This access decision is sent back to PEP as an XACML response for decision enforcement. Figure 1 illustrates the architecture of RAMARS system.

The role-based authorization along with the trusted attribute-based role assignment method in RAMARS has provided great flexibility and manageability for an originator to authorize a large number of unknown users across domains. This feature could easily be adopted by the data resource providers and service providers in Grid environments to specify fine-grained access control policies. By implementing the unified policy scheme, the same RAMARS PDP engine could be used to evaluate the policies and make access control decisions to protect both the data resources and the Grid services in a unified manner. Meanwhile, as the user's credentials are pushed to the RAMARS PDP and the originator's ROA policies are dynamically pulled at runtime based on RMPS policy, a light-weight and portable RAMARS PDP could be implemented without having them configured with any centralized policy and credential stores to render an access decision. This increases the flexibility for RAMARS PDP to be coupled with existing Grid services (or loosely attached to the Grid infrastructure for authorization purposes).

2.1 Integrated RamarsAuthZ System

Figure 2 illustrates the system architecture for the integrated RamarsAuthZ system. Inside the Globus Container, a RAMARS Adaptor is introduced for the Grid service (e.g., DRS service) to communicate with the RamarsAuthZ service, which is essentially based on RAMARS PDP. The RamarsAuthZ service can be called out by RAMARS Adaptor through two mechanisms: a localized function call by API or a remote service invocation by standard OGSI SAML messages [11]. In terms of attribute acquisition, both "push" and "pull" modes are supported. A Grid client can retrieve his attribute assertions from various IdPs and "push" these assertions to the Grid service. Alternatively, the Grid client can embed the metadata information about his preferred IdPs in the credential so that RamarsAuthZ is able to dynamically locate and retrieve authorization attributes based on the metadata. We call this newly introduced credential with specialized extensions as a RamarsAuthZ proxy certificate [12]. By including these information in the legitimate extension fields of X.509 certificate, the RamarsAuthZ proxy certificate can be accepted and verified by the existing Grid GSI authentication module [13] without requiring any further changes. Since a proxy certificate is a self-issued certificate by the Grid user, all embedded attribute assertions and/or IdP's metadata must be signed by the issuing IdP and the metadata distributor, respectively. Any unsigned attribute assertions and IdP metadata without proper integrity protection are discarded and cannot be used by RamarsAuthZ for making authorization decisions.

The overall authorization flow for DRS service works as follows: the service provider of DRS first specifies and stores the ROA authorization policies in his administrative domain. The DRS service maintains the location reference of the ROA policies when the service is deployed in the Grid infrastructure. When a Grid client sends a RamarsAuthZ proxy certificate and his data replication request to the Grid DRS service, the client is authenticated through the Grid GSI module. Then RAMARS Adaptor is invoked to parse the extensions in the proxy certificate and prepare an authorization request for RamarsAuthZ service

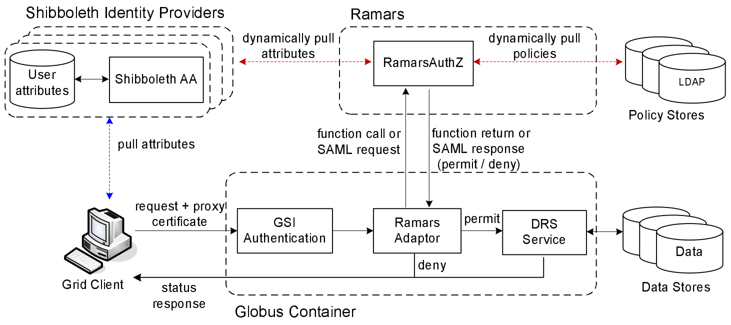

Fig. 2. Integrated RamarsAuthZ Authorization System

to check whether the Grid client is authorized to invoke the DRS service. The authorization request includes information on the requester's attributes and/or preferred IdPs passed by the proxy certificate, and the location reference of the service provider's ROA policies. Based on the authorization request, the RamarsAuthZ service can dynamically retrieve the service provider's ROA policies and the requester's attributes to make the authorization decision. The decision is sent back to RAMARS Adaptor and enforced accordingly by the DRS service.

2.2 Enhanced DRS for Access Control

As discussed in Section 1, data originators delegate the data sharing responsibilities to the DRS service, yet the DRS service is responsible to enforce the data originators' authorization policies on their behalf during each step of data sharing process. We demonstrate such capabilities by implementing an enhanced DRS service.

In the original DRS service, each data resource is identified by a unique *logical name*. A registry database is maintained where a *logical name–physical location* mapping is maintained for each data resource and its replicas. For instance, "*GeneSequence–gsiftp://abc.com/var/gseq.tar*" states an entry in the registry database. By querying the logical name "*GeneSequence*," a user could locate his desired data item at "*gsiftp://abc.com/var/gseq.tar*." After the physical location of the data item is successfully located, a file transfer component in DRS is invoked to copy the user's desired data item to the target location. We demonstrate how the invocation of DRS service can be protected by RamarsAuthZ. However, such configuration cannot further protect the actual data resources that are replicated through DRS. In order to enable the originator control for each individual data resource being shared through DRS, we introduce additional attributes with each registry entry to indicate the originator of the data item and its ROA policy information as *originator* and *roa_location*, respectively. With these two attributes being specified, the DRS service not only discovers the physical location of the data resource, but also collects the necessary ROA information for the RamarsAuthZ service to locate the data originator's policies. And these policies can be evaluated and enforced by RamarsAuthZ in the same manner as we discussed in Section 2.1.

3 Performance Evaluation

We conducted a series of experiments to evaluate how well the system scales along with the increased evaluation complexity and also analyzed the overhead of RamarsAuthZ authorization service over Globus DRS service. Figure 3 indicates the process time in miliseconds when we increase the number of user attributes involved in the authorization evaluation. The RamarsAuthZ policy evaluation time shows that with extreme complexity of evaluation when 80 attributes are involved, the overhead introduced by RamarsAuthZ authorization is 6.66% over the traditional GridMap authorization, which we believe is a promising outcome with respect to the performance of RamarsAuthZ authorization service. Same to our expectation, achieving a fine-grained authorization for stepwise data sharing involves considerable cost. However, considering the potential reduction of the administrative overhead against the practices of manually maintaining individual user accounts, RamarsAuthZ service still shows clear advantages both architecturally and technologically. Compared to the locally deployed authorization module, the overhead of SAML authorization messages cannot be neglected. Therefore, the usage of SAML for authorization in Grid envrionments needs to be limited for simple and optimized message assertion exchanges. Especially, instead of transferring a large number of attribute assertions as "push" mode, a reference to the Grid client's IdPs should be transferred within SAML message for RamarsAuthZ to operate under "pull" mode.

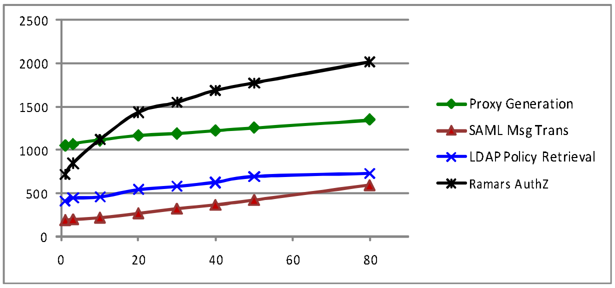

Fig. 3. RamarsAuthZ System Evaluation

4 Conclusion

In this paper, we have proposed an integrated framework that provides effective policy-driven role-based access control for Grid data sharing services. We also demonstrated that the RamarsAuthZ service does not rely on centralized policy stores and attribute authorities, which increases its scalability and portability, providing the authorization functionalities for various Grid services. As our future work, we currently attempt to explore policy cache and attribute negotiation mechanisms to further improve the performance of RamarsAuthZ system.

Acknowledgments

This work was partially supported by the grants from National Science Foundation (NSF-IIS-0242393) and Department of Energy Early Career Principal Investigator Award (DE-FG02-03ER25565).

References

1. Globus: GT 4.0: Data Replication Service (DRS), http://www.globus.org/toolkit/docs/4.0/techpreview/datarep/
2. Lang, B., Foster, I., Siebenlist, F., Ananthakrishnan, R., Freeman, T.: A Flexible Attribute Based Access Control Method for Grid Computing. Journal of Grid Computing 7(2) (2008)
3. Cantor, S.: Shibboleth Architecture: Protocols and Profiles (2005), http://shibboleth.internet2.edu/docs/internet2-mace-shibboleth-arch-protocols-200509.pdf
4. Alfieri, R., Cecchini, R., Ciaschini, V., dell'Agnello, L., Frohner, A., Gianoli, A., Lorentey, L., Spataro, F.: VOMS, an authorization system for virtual organizations. In: Proc. of 1st EuropeanAcross Grids Conferences (2003)
5. Groeper, R., Grimm, C., Piger, S., Wiebelitz, J.: An Architecture for Authorization in Grids using Shibboleth and VOMS. In: Proc. of 33rd EUROMICRO Conference on Software Engineering and Advanced Applications, pp. 367–374 (2007)
6. Thompson, M., Johnston, W., Mudumbai, S., Hoo, G., Jackson, K., Essiari, A.: Certificate-based Access Control for Widely Distributed Resources. In: Proc. of 8th Usenix Security Symposium (1999)
7. Chadwick, D.W., Otenko, A.: The PERMIS X.509 role based privilege management infrastructure. In: Proc.of the 7th ACM symposium on Access control models and technologies (SACMAT), pp. 135–140 (2002)
8. Jin, J., Ahn, G.J.: Role-based Access Management for Ad-hoc Collaborative Sharing. In: Proc. of 11th Symposium on Access Control Models and Technologies (SACMAT), pp. 200–209 (2006)
9. Jin, J., Ahn, G.J., Shehab, M., Hu, H.: Towards Trust-aware Access Management for Ad-hoc Collaborations. In: Proc. of 3rd IEEE International Conference on Collaborative Computing, pp. 41–48 (2007)
10. OASIS: XACML 2.0 core: extensible access control markup language (XACML) version 2.0 (2005),http://docs.oasisopen.org/xacml/2.0/access_control-xacml-2.0-core-spec-ospdf
11. Welch, V., Ananthakrishnan, R., Siebenlist, F., Chadwick, D., Meder, S., Pearlman, L.: Use of SAML for OGSI authorization (2005), https://forge.gridforum.org/projects/ogsa-authz/document/draft-ogsi-authz-saml-aug15-05.pdf/en/1
12. Tuecke, S., Welch, V., Engert, D., Pearlman, L., Thompson, M.: Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile (2004), http://rfc.net/rfc3820.html
13. Welch, V., Siebenlist, F., Foster, I., Bresnahan, J., Czajkowski, K., Gawor, J., Kesselman, C., Pearlman, S.M.L., Tuecke, S.: Security for Grid Services. In: Proc. of 12th IEEE International Symposium on High Performance Distributed Computing, pp. 48–57 (2003)