Supporting Secure Ad-hoc User Collaboration in Grid Environments

HPDC11 Paper Abstract

Markus Lorch, Dennis Kafura
Department of Computer Science
Virginia Tech
Contact e-mail: mlorch@vt.edu

Abstract

We envision that many usage scenarios involving computational grids will be based on small, dynamic working groups for which the ability to establish transient collaboration with little or no intervention from resource administrators is a key requirement. Current grid security mechanisms support individual users who are members of well-defined virtual organizations. Recent research seeks to provide manageable grid security services for self-regulating, stable communities. Our prior work with component-based systems for grid computation demonstrated a need to support spontaneous, limited, short-lived collaborations. The mechanisms we are developing focus on two key issues: binding and enforcement. Standard attribute certificates bind rights to users (or their surrogates); such rights may be freely delegated, traded, and combined. Enforcement is provided by operating systems extensions that regulate resource usage through access control lists. These extensions are available for common platforms and fully support legacy services. In combination, these mechanisms are compatible with and enable the usage of fine-grained rights, leverage other work in the grid computing and security communities, reduce administrative costs to resource providers, enable ad-hoc collaboration through incremental trust relationships and can be used to provide improved security service to long-lived communities.

1. Introduction

This paper describes preliminary work to support secure collaboration among ad-hoc, transient groups of grid users with minimal overhead imposed on resource administrators. Currently deployed grid security services support stable, well defined collaboratories [PEA01] and virtual organizations [FOS01] through single sign-on, rights delegation, the integration of local security solutions, and retention of final authority over the use of resources by the resource owners. Often only coarse grained access decisions are made by these grid authorization services (e.g. the mapping of authenticated global users to local user accounts as done by the Grid Security Infrastructure GSI [FOS98] and in UNICORE [ROM00]). The requirement for existing local user accounts for every grid entity poses a scalability problem. Current research aims at supporting long-lived communities in which the allocation of rights to the community as a whole (resource administration) is separated from the allocation of rights to individual members of the community (community administration). This separation reduces the administrative costs for resource owners and enables community-specific security policies. Our work seeks to further extend this line of research by supporting collaborations that are unanticipated (achieved by dynamic discovery or synchronous interaction), short-lived (perhaps only days or even a single session), and involving a small number (perhaps only two) of grid users.

The focus on ad-hoc groups arose from our earlier work with Symphony [LOR02], a component-based system for collaboration and sharing. Symphony enabled users to integrate

components representing grid computations or data made available by others. The integrated components could be accumulated through resource discovery (asynchronous sharing) or through real-time interaction (synchronous sharing). We envision that on computational grids many collaboration scenarios will be based on small, ad-hoc and dynamic working groups for which the ability to establish transient groups with little or no intervention from site administrators is a key requirement. While our work is focused on ad-hoc collaboration, the mechanisms we develop can also be used to provide improved security services to virtual organizations and other longer-lived communities as well.

2. Issues and Related Work

Our work and two related systems will be described in relation to three issues that are common among grid security systems. These issues are:

1. **Granularity:** the ability to specify fine grained security privileges in a platform independent manner. The scope of such privileges can range from single objects to multiple grid resources and entities of separate administrative domains.
2. **Binding:** the secure association and management of privileges with grid entities. This includes assignment, delegation, composition, and revocation of rights.
3. **Enforcement:** the limitation of operations performed on resources by a user to those permitted by an authoritative entity. Enforcement mechanisms need to be able to cope with expressiveness limitations of prevalent operating system security mechanisms. Portability and acceptable overhead are other key factors.

Two projects that have addressed these issues to varying extents and with different aims are the Community Authorization Service CAS [PEA01] and Akenti [THO99].

The granularity issue is addressed in both CAS and Akenti by way of flexible policy languages that allow arbitrarily fine grained specification of privileges. Akenti relies on its own policy language [THO01] to specify privileges, use-cases and attributes. While the current CAS implementation uses a simple proprietary policy language, the CAS architecture is designed independent of a specific language. A variety of policy languages exist which can be used to specify privileges, user attributes and the like ranging from basic privilege lists to sophisticated languages like the extensible access control markup language (XACML) [XACML] and the digital rights language for trusted content and services (XrML) [XRML].

The binding issue is handled differently in CAS and Akenti. CAS incorporates authorization information into restricted GSI proxy certificates (PCs). Policy information is carried in the PC to specify an entity's subset of rights from the set of rights assigned to its community. A PC is generated at the request of an authenticated user by a CAS server that manages the privileges for its community. Each resource owner assigns coarse grained rights to a single community group account; a user never authenticates directly to a resource but rather uses a restricted community identity known to the resource. Akenti binds attributes and privileges through attribute certificates (ACs) and thus separates authentication from authorization. Akenti uses X.509 identity certificates for authentication and a set of proprietary ASCII ACs that hold policy information for privilege management. Long-lived ACs convey group and role membership status. Short-lived ACs are used to cache and share authorization decisions among processes. So called "use-case" certificates are generated by resource owners to define usage policies for their resources.

For enforcement, both systems rely on the software providing the requested service to regulate system access. This approach solves the "expressiveness" limitations [PEA01] present in many traditional operating systems at the cost of limited support for legacy services that rely on the operating system to restrict their system access. The CAS team developed an implementation of the Generic Authorization and Access-Control API (GAA API) [RYU99] to provide a defined interface for the service to query authorization data from CAS independent of the used policy language. In CAS, the resource server that accepts a request enforces the policy resulting from the intersection of rights granted to the community account at the resource and the restrictions for this access specified in the PC by the CAS server. Akenti provides proprietary means by which a service can query information for authorization decisions.

Both CAS and Akenti provide solutions for long-lived collaborative environments that change in size and composition rather slowly and use code specifically developed for or ported to interact with their security mechanisms for enforcement. CAS reduces the administrative overhead that basic GSI mechanisms impose on large collaborative grid communities. Akenti allows a resource owner to specify usage policies in a flexible manner and provides support for role based access control. The use of policy languages in both projects provides for the necessary granularity in the specification of privileges for many collaborative scenarios.

To further promote the evolution of grid security mechanisms to support short-lived, ad-hoc collaborations additional mechanisms for binding and enforcement are needed. Our efforts are coordinated with the CAS development team at Argonne National Labs and other researchers from the Global Grid Forum and the IETF PKIX group to prevent redundant work and provide for the interoperability of our contribution.

3. Binding and Enforcement

We focus on two mechanisms: one, a binding scheme based on attribute certificates that allows for flexible assignment, delegation, composition and revocation of rights; and two, a portable, low-overhead enforcement scheme for fine-grained security policies that provides support for legacy applications. This modular approach combines the strengths of existing services with our components and also allows the two mechanisms to be used independently of each other. Our evaluation of the security mechanisms implemented by other projects, such as UNICORE and Legion [GRI99], suggests that our schemes can also be incorporated into those projects.

The proposed binding scheme employs standard attribute certificates to convey rights separately from identity certificates used for authentication. The separation of authentication from authorization allows for very flexible delegation and the combination of privileges from different sources and with variable lifetime. By delegating and trading such credentials users can form and dissolve groups directly through user-to-user trust relationships without administrator intervention. Due to the wide-spread use of the GSI, our approach complements the GSI and CAS services by adding a module that allows for the use of privilege management credentials (attribute certificates).

Our novel enforcement mechanism enacts fine-grained access policies through the use of commonly available, operating system extensions providing access control lists. This mechanism enables the secure execution of legacy codes, imposes significantly less overhead

than mechanisms which make access decisions on a call-by-call basis (e.g. sandboxing), and provides a simpler and thus more reliable means to protect system resources.

3.1 Binding

As noted above, binding refers to the means to assign, delegate, compose, and revoke rights. Our approach is to leverage existing public-key infrastructure (PKI), particularly the X.509 and IETF PKIX certificate structures, and provide a foundation for interoperating with other public-key based security mechanisms.

Rights are securely assigned and delegated to entities by embedding the rights in attribute certificates (AC). ACs bind attributes to a distinguished name (DN), the attribute holder. The grantor (issuer) of the capability will sign the AC. Given the identity certificate of the issuer a resource can check if the grantor is authoritative and determine whether the delegation is valid. The use of ASN.1 notation and structure as defined in [FAR01] allows us to create and process such certificates with standard tools.

The full separation of privilege credentials from identity credentials enables us to convey privileges without the limitations of impersonation. Impersonation schemes, like the single sign-on used in GSI proxy certificates, allow an entity to delegate all or a subset of privileges to an agent that then acts on the entity's behalf. However, these schemes are a subtractive security solution – the delegated rights are obtained by reduction from the set of all rights possessed by the grantor. This solution bears the danger of violating the least privilege principle [SAL75] as it is often problematic to clearly define the minimum subset of privileges needed by the agent.

With separate credentials for privileges and identities we are in the position to securely combine privileges from arbitrary sources and build a system that is based on the least privilege principle. This is a much needed feature which is not supported by impersonation schemes. An entity in a user-to-user collaboration scenario needs the ability to combine its own rights with rights received from collaborating entities. Collaboration may start with a low level of trust and minimal shared privileges. As the entities continue to work together the trust level increases and additional privileges are granted. The ability to create flexible and dynamic combinations of privileges allows us to model more accurately the actual relations between the collaborating entities.

Several mechanisms can be used for revoking rights. Certificate revocation lists are a common way to distribute information on certificate validity. The Online Certificate Status Protocol OCSP [MYE01] provides an alternative mechanism for revocation of attribute certificates.

3.2 Enforcement

Enforcement strategies can be characterized in one of three ways depending on how the identity credentials and the attribute information (if any) are combined to determine the effective privileges applicable to a request. The three strategies are:

- **Credential Mapping:** Basic or restricted identity credentials are mapped to local user accounts. This grants all permissions associated with the local user account less the optional restrictions to the request in restricted credentials. The GSI and CAS use this scheme.

- **Mixed Mode Authorization:** Credential mapping is used to define an initial set of privileges. Additional privileges based on presented capabilities are applied to the specific local user account before the request is served. After a resource access is completed the resource server revokes the additional privileges. Alternatively the privileges could remain assigned to the local user id for the lifetime of the capabilities as defined in the attribute certificates.

- **Capability Combination:** A user requests services from a resource where he has no permanent local account. The resource accumulates all the privileges from presented capabilities and applies them to a generic, initially very restricted user account; the request can then be served. After completion the privileges are removed and the account returned into its restricted state. It would also appear feasible to keep this temporary account assigned for subsequent requests based on the capability lifetimes. This type of dynamic user assignment and access authorization can be grouped with a demand-driven user allocation scheme for grid environments as suggested in [HAC01].

The enforcement of complex security policies which cannot be translated into a direct credential mapping is a challenging task. CAS and Akenti deal with this issue by requiring a grid service to implement self-policing mechanisms that perform access control based on provided policy information. The underlying operating system does not need to perform authorization for system access as these services run with a large set of OS privileges. Legacy services cannot be accommodated as they are unable to make the necessary authorization decisions.

We have investigated the use of file system access control lists (ACLs) on a number of UNIX operating systems, specifically Linux, Solaris, and IRIX, that were implemented following the POSIX.1E [POS88] recommendation. The application of shared access privileges in many operating systems traditionally involves the creation of user groups and the changing of group membership for users as well as resource objects (i.e. files). ACLs can be modified dynamically by the resource gatekeeper during the authorization of an incoming request. This provides the means necessary to assign transient access privileges to entities without the need for group constructs and enables dynamic, collaborative access to resource objects with a minimal set of rights assigned on demand. Many restrictions that limit the use of system wide resources can also be accomplished through portable and commonly available operating system tools, such as quota mechanisms.

The combined use of credential mapping mechanisms together with authorization mechanisms based on cumulative privileges conveyed through capabilities allows for very flexible and efficient authorization services that can operate within existing infrastructures. The particular strength of this approach lies in the low overhead due to enforcement within the operating system kernel and the ability to execute legacy code without modifications or wrappers. Unfortunately, the POSIX.1E recommendations have never been formally completed and standardized. Interfaces and implementations differ slightly. A portable and unified API is required to set and modify entries through grid gatekeepers. The linking of grid access control services to the security services provided by the underlying resource operating systems satisfies one of the requirements stated in the recent work on an Open Grid Services Architecture [FOS02].

Sandboxing of applications is another way for resources to enforce fine-grained security policies for legacy codes. Tools like Janus [GOL96] or the Virtual Execution Environment VXE [VXE02] provide a constrained environment. This environment monitors and evaluates all system access (e.g. through system call interception). Arbitrarily fine grained access policies can be enforced using such mechanisms. However, this flexibility comes at the price of considerable overhead which is often unacceptable for high-performance applications. In addition, it may be a significant effort to configure minimal access permissions for complex codes. Finally, sandboxing environments are often closely tied to vendor specific operating system mechanisms and not portable.

Redirection of library calls through dynamically injected shadow libraries is another mechanism that can be used to enforce security policies without application support. Bypass [THA01] uses such an approach as part of the Condor [BAS97] project. This approach is more promising than sandboxing as the shadow libraries are likely to be more portable. However, it requires a mechanism to ensure that the application is not using any system calls that are not "shadowed" and thus not controlled.

4. Status and research issues

We are currently working on a prototype that will enable grid users to specify and delegate access privileges using attribute certificates. We plan to incorporate these ACs into the GSI certificate transfer mechanisms in a modular way while remaining compatible to the standard GSI implementation. This will allow us to transmit and apply ACs together with PCs. Our user front end, Symphony, is written in Java and uses the Java CoG Kit [LAS01] to interface with Globus [FOS99] and the GSI.

We have had initial success with using mechanisms to apply and enforce file system privileges using file system access control lists. We are now working on a module that will allow us to parse an operating system independent policy and apply it using APIs of the mentioned operating systems. Next we will group this component with a grid resource gatekeeper to perform these functions on demand.

We have found that it may be necessary to create a protocol for entities to negotiate the necessary capabilities for a certain request. This will enable a user agent to intelligently select the minimum necessary capabilities to supply with a request. Another issue we encountered is the lack of unique serial numbers or unique names of proxy certificates. This prevents us from binding capabilities to a specific PC and thus to a specific request. This issue will be addressed in future releases of the GSI.

An auditing and logging server that issues unique certificate tracking (or serial) numbers to entities upon request could be used if it is necessary to audit the system and discover what capabilities an entity currently holds. Such a server could also aid in the revocation of capabilities.

File system ACLs and mechanisms to inject DLLs for security evaluation of system calls are also present in current versions of the Microsoft Windows operating system. We plan to investigate these possibilities more closely.

5. Conclusion

This paper reports our initial steps to provide enhanced grid security services that support a variety of collaborative scenarios. These services enable low-overhead transient collaboration and improve the abilities of existing systems that focus on more static collaborative structures. More flexible binding of rights is achieved through the use of standard attribute certificates. Low-overhead enforcement mechanisms for file access rights enable efficient realization of fine-grain access policies while supporting legacy software. The ability to securely run legacy codes with fine-grained access permissions using existing operating system extensions is a much-needed feature in grid environments. User-to-user delegation without administrative intervention may prove to be the enabling mechanism for transient ad-hoc collaboration. Our work complements that of GSI and CAS, can interoperate with these systems, and incorporates experiences from the Akenti project.

In our paper we will present a detailed description of our architecture, the protocols and mechanisms used. We will include a report of our experiences in building, deploying and using a working prototype on our local grid infrastructure.

References

[BAS97] J. Basney, M. Livny, T. Tannenbaum, "High Throughput Computing with Condor", HPCU news, Volume 1(2), June 1997.

[FAR01] S. Farrel, R. Housley „An Internet Attribute Certificate Profile for Authorization“ IETF PKIX Working Group draft, June 2001, http://www.ietf.org/internet-drafts/draft-ietf-pkix-ac509prof-09.txt

[FOS98] I. Foster et al, "A Security Architecture for Computational Grids", ACM Conference Proceedings, Computers and Security, ACM Press, NY, pp. 83-91, 1998

[FOS99] I. Foster, C. Kesselman, "Globus: A Toolkit-Based Grid Architecture" The Grid, Blueprint for a Future Computing Infrastructure, Morgan Kaufmann, San Francisco, 1999, pp. 259-278

[FOS01] I. Foster, C. Kesselman, and S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," International Journal of Supercomputer Applications, 2001.

[FOS02] I. Foster et al, "The Physiology of the Grid – An Open Grid Services Architecture for Distributed Systems Integration", Draft research paper presented at the Global Grid Forum 4, February 2002, http://www.globus.org/research/papers/ogsa.pdf

[GOL96] I. Goldberg et. al, "A secure environment for untrusted helper applications" Proceedings of the Sixth USENIX UNIX Security Symposium, July 1996

[GRI99] A. Grimshaw et al., "Legion: An Operating System for Wide-Area Computing", IEEE Computer, 32:5, May 1999: pp. 29-37.

[HAC01] T. Hacker, B. Athey, "A Methodology for Account Management in Grid Computing Environments", In Proc. Second Int. Workshop on Grid Computing, Denver, USA, November 2001

[LAS01] G. von Laszewski et al., "A Java Commodity Grid Kit", Concurrency and Computation: Practice and Experience, pages 643-662, Volume 13, Issue 8-9, 2001.

[LOR02] M. Lorch, D. Kafura, "Symphony – A Java-Based Composition and Manipulation Framework for Computational Grids", accepted for publication in the conference proceedings of ccgrid 2002, Berlin, Germany, May 2002

[MYE01] M Myers et al. „ Online Certificate Status Protocol, version 2 “
IETF PKIX Working Group draft, March 2001,
http://www.ietf.org/internet-drafts/draft-ietf-pkix-ocspv2-02.txt

[POS88] „IEEE standard portable operating system interface for computer environments”,
IEEE Std 1003.1-1988 , 30 Sept. 1988

[PEA02] L. Pearlman et al., “A Community Authorization Service for Group Collaboration”, submitted to the 2002 IEEE Workshop on Policies for Distributed Systems and Networks,
http://www.globus.org/Security/CAS/CAS_2002_Submitted.pdf

[SAL75] J. R. Salzer and M. D. Schroeder, "The Protection of Information in Computer Systems", Proceedings of the IEEE, Sept. 1975

[STE97] Stevens, R. et al., "From the I-WAY to the National Technology Grid"
Communications of the ACM, vol. 40, pp. 50-61, 1997.

[THA01] D. Thain, M. Livny, "Multiple Bypass: Interposition Agents for Distributed Computing", Journal of Cluster Computing, volume 4, pages 39-47, 2001.

[THO99] M. Thompson et al., “Certificate based Access Control for Widely Distributed Resources”,
Proceedings of the 8th Usenix Security Symposium, 1999

[THO01] M. Thompson, “Akenti Policy Language”, White paper,
http://www-itg.lbl.gov/Akenti/Papers/, July 2001

[ROM00] M. Romberg "UNICORE: Beyond Web-based Job-Submission"
Proceedings of the 42nd Cray User Group Conference, May 22-26,2000, Noordwijk

[RYU99] T.V. Ryutov, G. Gheorghiu and B.C. Neuman “
An Authorization Framework for Metacomputing Applications”,
Cluster Computing Journal, Vol. 2 Nr. 2, 1999, pp. 15-175

[VXE02] The Virtual Execution Environment
http://www.intes.odessa.ua/vxe, February 2002

[XACML] Organization for the Advancement of Structured Information Standards (OASIS) “extensible Access Control Markup Language”,
http://www.oasis-open.org/committees/xacml/index.shtml, February 21, 2002

[XRML] “XrML - The Digital Rights Language for Trusted Content and Services”, http://www.xrml.org,
February 25, 2002