Decentralized Trust Domain Management in Multiple Grid Environments

Vincent C. Hu and Karen Scarfone

National Institute of Standards and Technology,
Gaithersburg, MD 20899-8930, USA
{vhu, karen.scarfone}@nist.gov

Abstract. Trust domain management for the global access of a grid is managed under centralized schema for most of the current grid architectures, which are designed based on the concept that there is only one grid for every grid member, therefore requiring central management for authentication and authorization. This design not only has its own limitations, but it is also awkward when a member of a grid may also be a member of other intersecting grids. Schema for such multi-grid environments have not been well thought out. In this paper, we present a schema that enables trust domain management in a dynamic multi-grid environment

Keywords: trust domain, access control, grid

1 Introduction

Regardless of the readiness of software and hardware, grid systems have greater challenges with infrastructure security compared to non-grid computing environments. Among these challenges, access control of the grid is the most crucial and difficult, because the management of access control on a multi-organization grid and the grid-mapfile (map a global grid credential to a local user's login name) does not scale well, and it works only at the resource level, not the collective level [1, 2].

Grid resources such as software, data, and hardware components are managed by diverse organizations in widespread locations. The nodes, members, or computers of a grid are able to act independently without centralized control, but the Trust Domain (TD) (i.e., the coverage of the authentication and authorization for the global access of a grid) is managed under a centralized system for most of the current grid architectures [3, 4, 5]. The current grids have been designed based on the concept that there is only one grid for every grid member, therefore requiring central management in order to authenticate and authorize members. However, in reality, a member of grid A may also be a member of other grid systems B, C, D, etc. Consideration of such a multi-grid environment is elusive in most of the current grid architectures. Some [6, 7] suggest that a single grid architecture might be extended to integrate other grids by configuring them as hierarchical or sub-grids of a super grid, making the centralized mechanism today's main solution of TD management. However, this approach would inherit the limitations of centralized systems: 1) a potential single point of failure (i.e., when the root grid fails, its descendent [or sub] grids fail as well), 2) poor scalability - a member joining or leaving a grid requires administration effort from the center TD manager, and 3) a unified protocol and algorithm is required for TD management throughout the whole grid environment.

The limitations not only hinder the efficiency of TD management but also make it difficult to share resources with remote grids and to combine resources, users, and operations of a small grid community into a larger one, because of the complexity when TDs of multi-grid environments intersect with other TDs. Even though layers of security architecture for grid have been defined [8] that allow expansion for multi-grid environments, and techniques of trust negotiation for peer-to-peer access in an open system environment (not bounded by the same access control policy) have been proposed [9, 10], there are few schemas that address the limitations. In this paper, we discuss an extension of the TD management paradigm that remedies the issue by creating an algorithm and protocol on top of the current TD management architecture.

This paper contains seven sections. Section 1 introduces the motivation for the research. Section 2 defines terms that will be applied to this paper. Section 3 discusses the issues of TD management for multi-grid environments. Section 4 introduces an algorithm for multi-grid aware TD management followed by a supporting protocol. Section 5 reviews related projects and discusses how they can include the new TD management schema. Section 7 is the conclusion.

2 Terms

We define a set of terms named from the perspective of an individual grid member with the assumption that multiple grids are available for a grid member.

Local system sees other grid members as **remote systems**.

Resource is hardware and software accessible by any grid member; it is **local** if only available through the local system, and **global** otherwise.

Trust is the authentication and authorization from one system to another for the access of its resources.

Trust domain (TD) is a collection of subjects, operations, and resources of systems that trust each other (i.e., authentication and authorization managed under the same access control policy). A single host system's TD is managed by local access control policy, and various grid TDs are governed by the matching grid access control policies.

Combined policy is created by incorporating local and grid access control policies.

Capability is an operation and resource pair that a subject is allowed to access.

Sub-grid is a multi-grid environment where multiple unrelated grids are a subset of one super grid. The super grid might be a sub-grid of another, and so on, as illustrated in Figure 1.

Hierarchical grid is a multi-grid environment where a grid is a subset of an upper layer grid. The upper layer grid might be a subset of another upper layer grid, and so on, as illustrated in Figure 1.

3 Trust Domain Management

The key to sharing resources in a multi-grid environment is to ensure that the sharing relationship can be initiated among arbitrary local systems, allowing new participants to share dynamically across different platforms and TDs. Therefore, besides general security functions, including integrity, confidentiality, and nonrepudiation, multi-grid requires a standard TD mechanism for authentication and authorization. The main responsibility of a current security architecture layer for grid authentication and authorization is to translate the global entity (including subject and object) into the local entity. This layer contains mapping files for global IDs and local IDs, a unified access control list, a unified digital certificate (such as X.509), and an access control language (such as XACML). Although the layer maintains the independency, authentication, and combination requirements of grid [2], conceals the differences of local security solutions, and provides a unified platform for the upper layers [8], these mechanisms are tied into the data structure of the software or languages they are built from, so they are only available to the TD in hierarchical or sub grid environments, as illustrated in Figure 1. The TD management for such multi-grid environments is adding another layer of protocol to the existing single security architecture.

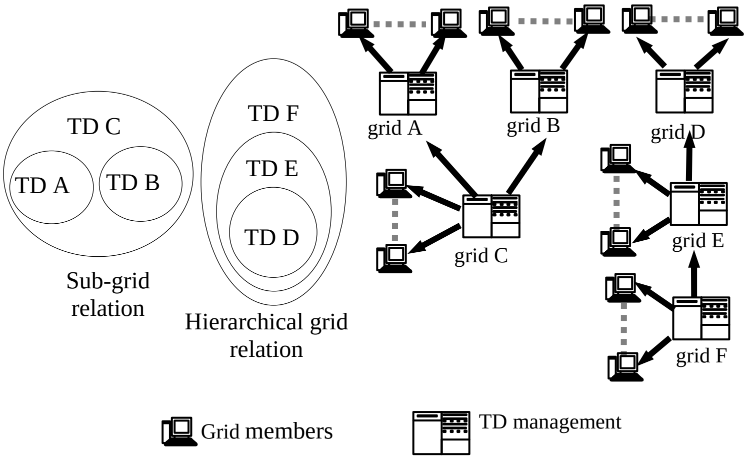

Fig. 1. TD management example of hierarchical and sub multi-grid environment

In reality, a local member of grid A may want to be a member of grid B – for example, to serve as an information provider for the research community, and at the same time access information from the engineering community, which is unrelated to the research community. Therefore, it is practical to allow grids or grid members to have their TDs intersect with other hierarchical and sub grids; in other words, the multi-grid environment does not have to be in a tree (hierarchical or sub) structure. Figure 2 shows an example of a non-tree type structure of grids, in which a local system might participate in more than one grid. To accomplish that, a schema is required that is neither based upon the embedded tree architecture nor reliant on the data structure of the access control languages.

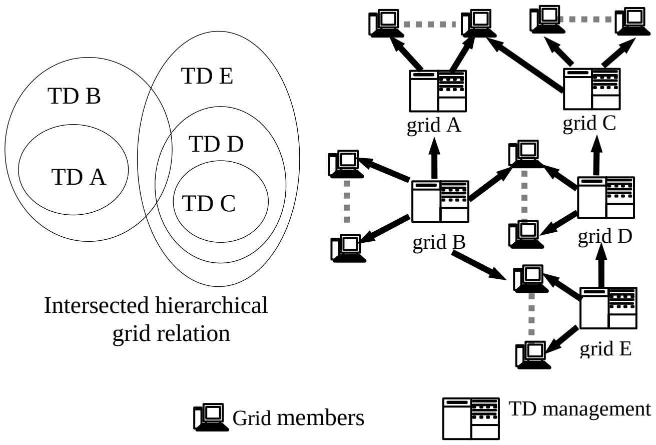

Fig. 2. TD management example of non-tree multi-grid environment

4 Multi-Trust Domain Schema

To devise a schema for a multi-grid environment, we need to generate an algorithm and a supporting protocol to handle the authentication and authorization across grids without relying on central TD management or services to avoid their limitations. Such schema should be able to accommodate any constellation of multi-grid environments as well as their dynamic changes. Sections 4.1 and 4.2 describe the algorithm and the protocol that support the schema.

4.1 Trust Domain Algorithm

If the dynamic TD relations of a multi-grid system are represented by a graph, with each node as a TD management system and each link from node *x* to node *y* as the hierarchical TD coverage of *y* by *x*, then the graph may not be a tree. To support a grid that might be in a non-tree (graph) environment, instead of consulting the central service, each grid needs to be able to identify the relations with other grids by referring to the locally maintained directory that maps the TD relations of its neighbors, and to calculate the access decision (authorization) according to the provided service requestor (subject) and capability (operation and resource pair).

The authorization process needs to include functions to find the intersection of TD coverage and determine which remote global access control policy to combine with the local access control policy of the requested resource [2], as required by grid security [11, 12] and for the authorization process, the local system of the resource of an access request needs to incorporate its local access control policy with a global access control policy so that correct global identification and delegation can be assigned.

To formally describe this algorithm we introduce the following symbols and functions:

- Symbol s_x denotes the subject s that is covered by TD x, o_x denotes the operation that is covered by TD x, r_x denotes the resource that is covered by TD x, S_x denotes a set of subjects that are covered by TD x, and C_x denotes a set of capability (operation and resource) pairs $\{(o1_x, r1_x)\ldots(on_x, rn_x)\}$ that are covered by TD x.
- Function $L(e) = x$ is the local TD where e could be a subject s_x or a capability c_x under the local TD x.
- Function $SC(x) = \{S_x, C_x\}$ lists the sets of subjects and capabilities that are covered under TD x.
- Function $TD(e) = \{p,......, q\}$ is the set of TDs that e is a member of, where e could be either a subject s_x or capability c_x.
- Function $CTD(e_1,...,e_n) = TD(e_1)\cap\ldots\cap TD(e_n) = \{p,......,q\}$ is the list of common (intersect) sets of trust domains that $e_1...e_n$ are covered under, where $e_i \in \{e_1,.., e_n\}$ could be a subject s_i or capability c_i under TD i.
- Function $LCTD(e_1,...,e_n) = x$ is the least common trust domain such that $|SC(x)| = \text{MIN}(|SC(p)|,\ldots,|SC(q)|)$, where $\{p,\ldots,q\}= CTD(e_1,....,e_n)$, and $e_i \in \{e_1,.., e_n\}$ could be a subject s_i or capability c_i under TD i.
- P_x denotes the access control policy of TD x.
- $Grant(P_q, P_r, <s, c>)$ returns the authorization result of access request $<s, c>$ that is evaluated by the combination of access control policies q and r.

For example, assume s_x and c_x are a subject and capability of S_x and C_x respectively, and x is 1 to 7. In Figure 3, then:

$L(s_1) = td1$, $SC(td9) = \{S_2, C_2, S_3, C_3, S_4, C_4, S_5\}$, $TD(c_5) = \{td5, td10, td11\}$, $CTD\ (s_5, c_7) = \varnothing$, $CTD(c_1, c_2) = \{td8, td10, td11\}$, $LCTD(c_1, c_2) = td8$, $CTD(s_6, c_1) = \{td10, td11\}$ and $LCTD(s_6, c_1) = td10$.

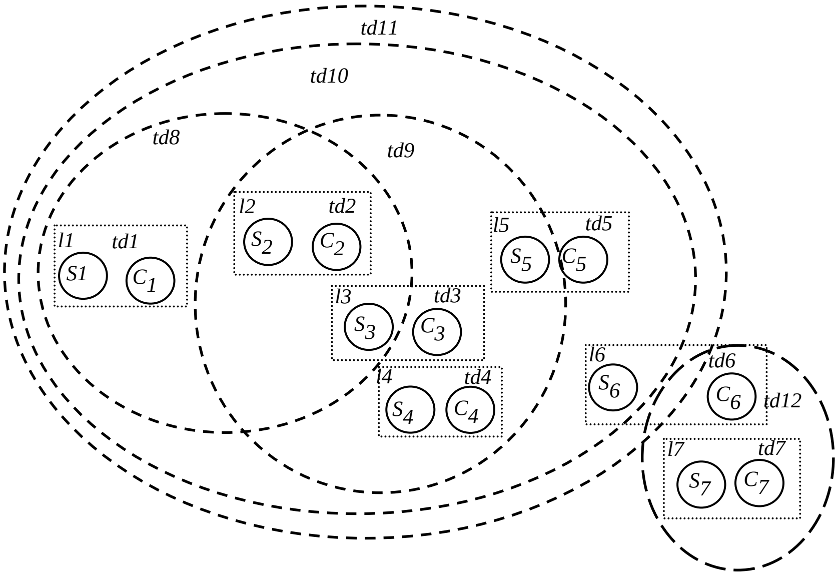

Fig. 3. Example of trust domains of multi-grid environment

Thus an access request <s_x, c_y> (i.e., s_x requests to perform c_y) is evaluated for access at local system; if there exists a common global TD for $L(c_y)$ to combine, then there is an access control policy that the access request can be managed under. However, there may be more than one common global policy that is qualified for the combination; although any one of them can be used, the **least common** one is best to choose because by least common we mean that it has the least number of local TDs involved in the policy integration, and therefore the minimum number of calculations required for searching the access permission. Formally, an access request <s_x, c_y> requires $L(c_y) = tdy$ to integrate with global trust domain $LCTD(s_x, c_y) = tdz$ if $\neg(L(s_x) = L(c_y))$ (no TD integration necessary for local access), and is evaluated by $Grant(P_{tdy}, P_{tdz}, <s_x, c_y>)$ if $CTD(s_x, c_y) \neq \emptyset$ (there is no common TD to satisfy the access request). The algorithm is:

```
Access($P_q$, $P_r$, < $s_x$, $c_y$>) {
  If {
        $q = L(c_y)$ ;
        $r = LCTD(s_x, c_y)$ ;
        $CTD(s_x, c_y) \neq \emptyset$ ;
        $\neg(L(s_x) = L(c_y))$
    } return Combine_Access($P_q$, $P_r$, $s_x$, $c_y$) /*apply combined policy of $P_q$ and $P_r$ for <$s_x$, $c_y$>*/
     else return DENIED
}
```

The *Combine_Access* (P_q, P_r, s_x, c_y) function evaluates the permission resulting from the combination of policies P_q, and P_r. Note that policy combination itself is another field of research, which mechanism is described in [2, 13, 14].

As shown in Figure 3, a request <s_1, c_5> can be evaluated by $Grant(P_{td5}, P_{LCTD(s_1,c_5)}, <s_1, c_5>) = Grant(P_{td5}, P_{td10}, <s_1, c_5>)$, and a request <$s_3$, c_7> cannot be evaluated (therefore, not granted for access) because $CTD(s_3, c_7)$ = Ø: no common TD coverage.

Assume $Cap(s_1 \ldots s_i, c_1 \ldots c_j) = \{c_1 \ldots c_j\}$ and $Sub(s_1 \ldots s_i, c_1 \ldots c_j) = \{s_1 \ldots s_i\}$ is a set of capabilities and a set of subjects respectively from a set of capabilities and subjects $\{s_1 \ldots s_i, c_1 \ldots c_j\}$. We can derive all the capabilities available from the multi-grid environment for a subject s_x by:

$Cap(s_x) = \{C \mid C = Cap(SC(TD(s_x)))\}$, and for each c_x, all permitted users are $Sub(c_x) = \{S \mid S = Sub(SC(TD(c_x)))\}$.

4.2 Scalability

The algorithm in Section 4.1 allows the creation of a new TD without broadcasting the new creation to all TDs in the environment. A TD of a local system or a grid can form a new TD by deciding which subset of *S* and *C* in its local system should participate and inviting subjects from other TDs to join, as long as the newly created grid ID is maintained in the local TD and in the TDs which are invited to join. As in Figure 4, a new *tdx* is formed by a local system *li* and invited remote system *lj* such that some of the remote subjects S_{j2} from S_j of TD *tdj* and local capabilities C_{i1} from C_i of TD *tdi* are available for the new trust domain *tdx*. An access request $<s_{j2}, c_{i1}>$ where $s_{j2} \in S_{j2}$, $c_{i1} \in C_{i1}$ is evaluated by $Grant(P_{tdi}, P_{LCTD(s_{j2},c_{i1})}, <s_{j2}, c_{i1}>) = Grant(P_{tdi}, P_{tdx}, <s_{j2}, c_{i1}>)$.

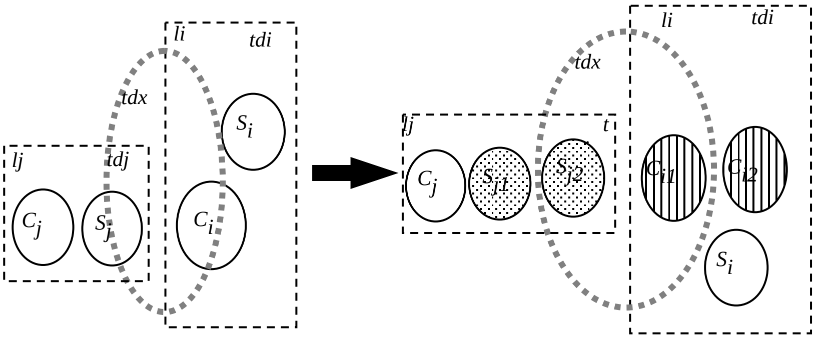

Fig. 4. Create new trust domain

As dynamic as creating a TD, deleting a TD requires the TD management system to remove the deleted TD from its database, and then broadcast the update to all *L*(*e*), where *e* ∈ *SC*(*tdx*) and *tdx* is the TD to be deleted, so that the receiving local systems can remove the *tdx* from their TD listings as well.

As shown, the scalability in terms of adding and removing grid members of TD management requires only information exchanges of involved grids, which can be easily retrieved from the locally maintained TD listing. The protocols of the maintaining processes are discussed in the next section.

4.3 Protocol

Assuming the protection and format of messages of the underlying security layer are provided, there are various ways to implement the algorithm in Section 4.1 as long as the following basic processes are enforced by all the local and TD management systems:

- The message (<s_i, c_j>, $TD(s_i)$) from a local system *li* is sent to the grid network for the access request, where <s_i, c_j> is the access request from subject s_i for resource c_j, and $TD(s_i)$ is stored in *li* and updated when *li* joins or leaves any grid.
- *lj* calculates $LCTD(s_i, c_j)$ and evaluates *Grant*($P_{L(c_j)}$, $P_{LCTD(s_i,c_j)}$, <s_i, c_j>) after receiving an access request.
- $L(s_x)$ and $L(c_x)$ of every local *lx* need to be sent to the TDs where they are covered when an update (joining or leaving of a grid member) occurs.
- Every TD should update its coverage list after receiving the updated subjects and capabilities list from the TDs it is managed by and is managing.
- A local system should be able to decide if it wants to remain in a TD when a $TD(s_x)$ or $TD(c_x)$ of local *lx* is changed, to avoid being in the same TD with untrusted subjects or capabilities.

The messages for the protocol can thus be defined based on the above functions. Note that we skipped the details of handshaking for lower level protocols because they are independent from the actual grid architecture, such as how proxy servers (for most current designs) are implemented in the architecture. The message format is:

<message id, message type, target system, action, target TD, target subjects, target capabilities>

- message id contains the message identifier (or sequence number) for the tracking of message sequence for both the sending and receiving ends.
- message type indicates if this message is for an access request, member's update (add/delete of subjects or capabilities), TD maintenance (add/delete of a TD), or ACKs of the former messages.
- target system is the identification of the TD or local system the message is sent from.
- action sets one of the following actions: add/delete TD, add/delete subject, or add/delete capabilities.
- target TD/subjects/capabilities are the TD, subjects and capabilitys the action is performed upon.

The finite states of the message exchange protocol are illustrated in Figure 5 below:

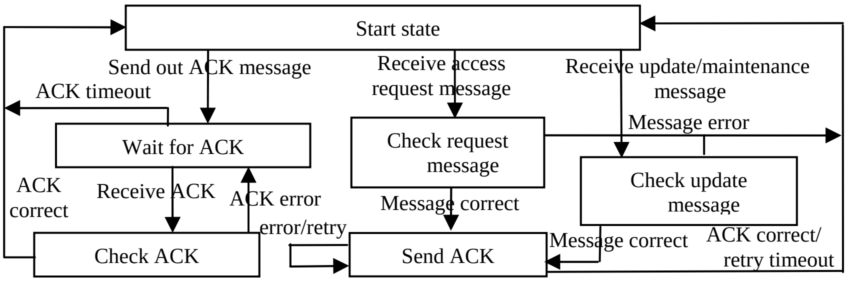

Fig. 5. Finite states of multi-TD message exchange protocol

The algorithm and protocol described allow the architecture of the multi-grid TD to be built in a graph instead of a tree structure, which permits multiple connections among TDs. This addresses the limitations of being constructed with a single centralized architecture:

1. No single point of failure for a hierarchical or sub grid, because TDs can communicate through alternative TD coverage.
2. When a grid member joins or leaves, the change only needs to propagate to the TDs that are related to that member.
3. There is no limitation on TD mechanisms as long as the global TD protocol is enforced.
4. Any local system can create/delete a TD freely without permission from the central TD manager.

5 Related Works

Grid Security Infrastructure (GSI) in Globus [4] (and other grid applications) contains a library and a few utilities that are used as a standard mechanism for bridging disparate security mechanisms. It not only understands identity credentials of all grid members but also supports delegation and policy distribution by translating between other mechanisms and GSI as needed and converting from a GSI identity to a local identity for authorization. Multi-grid TD management has not been articulated in SGI; however, the proposed schema can be included as an extension of its current security layer, such that the TD identification is added in the Authentication, Delegation, and Authorization layers (e.g., TD ID extension in X.509 End Entity and Proxy Certification, Attribute assertion in SAML). In contrast to the relatively homogenous approach of GSI, OGSA [3] security envisages translation and mapping of security parameters (e.g., credentials) between different domains [15]. To incorporate our schema in Section 4, the TD information in the protocol should be included in the Identity mapping services (i.e., Trust, Attribute and Bridge/Translation Services Service) such that the combination of the subject DN, issuer DN, and certificate’s serial number may be considered to carry not only the subject’s or service requestor’s identity [3] but also the TD information.

XACML [16] based authorization mechanisms such as Virtual Organization Membership System (VOMS), Shibboleth (with appropriate PDP implementation), PRIMA, and Privilege and Role Management Infrastructure Standards (PERMIS) [17], which may equip the capability but not yet include the multi-grid mechanism, can also consider incorporating our schema in their authorization functions.

6 Conclusion

In this paper, we developed a schema that includes an algorithm and general protocol that handle dynamic multi-grid TD management. The basic idea for the schema is to find the least common TD for the subject and resource of an access request to be combined with the local access control mechanism. Instead of the central management ideas of the existing architectures, the proposed schema relies on the exchange of TD information for each grid, which not only avoids the limitations of a centralized system but also provides the freedom to dynamically participate in grid membership. Although the detailed architecture and is not included in this paper (left for future research), we believe this schema could be used for the next generation of grid TD management design.

Reference

1. Chien A. A., “Globus Grid Security”, *CSE225 Lecture note #13*, University of California, San Diego, 2004.
2. Hu C. V., Ferraiolo D. F., and Scarfone K., “Access Control Policy Combinations for the Grid Using the Policy Machine”, Seventh IEEE International Symposium on Cluster Computing and the Grid — CCGrid 2007, Rio de Janeiro – Brazil, May 2007.
3. ”The Open Grid Services Architecture, Version 1.0”, http://www.gridforum.org/documents/GWD-I-E/GFD-I.030.pdf.
4. The Globus Security Team, “Globus Tookit Version 4 Grid Security Infrastructure: A Standards Perspective, www.globus.org/security/overview.html.
5. “The Open Source Architecture”, http://www.apac.redhat.com/software/architecture/forward.php3.
6. Neisse et al. “An Hierarchical Policy-Based Architecture for Integrated Management of Grids and Networks”. *Proceedings of The IEEE 5th International Workshop on Policies for Distributed Systems and Networks (POLICY2004)*, pages 103-106, Yorktown Heights, USA (short paper), July 2004.
7. In et al, “Policy Based Scheduling for Simple Quality of Service in Grid Computing”, *18th International Parallel and Distributed Processing Symposium (IPDPS'04)*, 2004.
8. Zhou Q., Yang G., Shen J., and Rong C., “A Scalable Security Architecture for Grid”, *Proceedings of the Sixth International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT’05)*, 2005.
9. Ravichandran A., Yoon J., “Trust Management With Delegation In Grouped Peer-to-Peer Communities”, *Proceedings of 11th ACM Symposium on Access Control Models and Technologies-SACMAT 2006.*
10. Lee A. J., Winslett M., Basney J., and Welch V.,“Traust: A Trust Negotiaton-Based Authoriztion Service for Open Systems”, *Proceedings of 11th ACM Symposium on Access Control Models and Technologies-SACMAT 2006.*
11. Nagaratnam et al, “The Security Architecture for Open Grid Services”, *Global Grid Forum, Open Grid Service Architecture Security Working Group (OGSA-SEC-WG)*, 2002.
12. Foster I., Kesselman C., Tsudik G., and Tuecke S., “A Security Architecture for Computational Grids”, *5th ACM Conference on Computer and Communication Security*, 1998.
13. Hu C. V., Frincke D. A., and Ferraiolo D. F., “The Policy Machine For Security Policy Management”, *Proceeding ICCS Conference*, San Francisco, 2001.
14. Ferraiolo D. F., Gavrila S., Hu C. V., and Kuhn D. R., “Composing and Combining Policies under the Policy Machine”, ACM SACMAT, 2005.
15. Robiette A., JISC Development group, “Grid Security: Present and Future”, JISC Grid Security Workshop, Dec. 2002.
16. OASIS, “Extensible Access Control Markup Language (XACML), TC”, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml.
17. Lang et al, “A Multipolicy Authorization Framework for Grid Security” *Proceedings 5th IEEE International Symposium on Network Computing and Applications, 2006.*