"1st International Symposium on Computing in Informatics and Mathematics (ISCIM 2011)"
in Collabaration between EPOKA University and "Aleksandër Moisiu" University of Durrës
on June 2-4 2011, Tirana-Durres, ALBANIA

Authorization Strategies for Grid Security: Attribute-Based Multipolicy Access Control (ABMAC) Model

Katerina RISTOSKA[1]

[1]*Faculty of Contemporary Sciences and Technologies, SEE University, Tetovo - R. Macedonia*
Email: kr11267@seeu.edu.mk

ABSTRACT

The emergence of Grid computing technology is being followed by three main security concerns: the independence of the domains where the resource providers (RPs) are situated; the need for supporting different security policies and the non-necessity of the science gateways for user authentication. Great effort has been involved in order to solve these concerns through the appearance of different access control models, like Identity-Based Authorization Control (IBAC) and Role-Based Authorization Control (RBAC), which based their access request decisions on user identity, that is, on user authentication. However, these models proved as inflexible, non-scalable and unmanageable in a distributed environment. Accordingly, a novel approach, known as Atrribute-Based Multipolicy Authorization Control (ABMAC) model has appeared. ABMAC, which is being described in this paper, uses the attributes of the Grid entities for user authorization, based on the concepts of service-oriented architecture (SOA) and the eXtensible Markup Language (XML) standards – eXtensible Access Control Markup Language (XACML) and Security Assertion Markup Language (SAML). Moreover, ABMAC has been partly implemented in the Globus Toolkit 4 (GT4) Authorization Framework, and consequently it is expected to be outstanding contributor to Grid security.

INTRODUCTION

Distributed computing has recently emerged into a new promising technology known as Grid computing, which involves Grid systems as Virtual organizations, composed of several independent autonomous domains in a service-oriented architecture. This means that the Grid architecture is based on Web services and their correlated technologies; hence, by having this kind or architecture, it is self-explanatory that the Grid should engage network security in its highest degree.

Typical network security infrastructure involves three main network security principles, widely familiar as the **CIA triad - confidentiality, integrity and availability** [1]. These principles are being preserved through three steps: (1) **identification**, which describes a method of ensuring that a subject (user, program, or process) is the entity it claims to be; (2) **authentication**, which is a process of confirming a claimed identity, that is, the mechanism whereby systems may securely identify their entities and (3) **authorization**, which is the mechanism of determining what level of access a particular authenticated entity should have to safe resources systematized by the system. Accordingly, these steps should be executed in a consecutive manner in order to provide the necessary level of network security.

However, the earlier access control models, namely IBAC and RBAC, were granting access to the users according to their identities, i.e., they have not even reached the authorization part of the assertion procedure. Therefore, it is more than obvious that the Grid security remained as challenging issue, which during the past two decades has motivated the researches to seek for appropriate security solutions. Hence, various new aspects are being brought along with the developing of the Grid: (a) its merging with the web services toward service-oriented architecture and (b) the definition of SAML and XACML as extraordinary contributors to Grid security. These have led to the appearance of the ABMAC model, which is the central point of discussion in this research paper [2].

The paper has been organized into several parts. The first section describes the Grid architecture and its security challenges, with its subsections covering Web services, XML, XACML and SAML. The second section briefly describes the disadvantages of the former access-control models and the reason why ABMAC is supposed to be the outstanding contributor to Grid security. Finally, the last section dives more deeply into review of the already proposed authorization models by various researchers and the likelihood of their eventual integration.

OUTLINE OF THE GRID ARCHITECTURE AND ITS SECURITY CHALLENGES

The Grid is an *open distributed, heterogeneous and loosely coupled network*, being composed of Grid systems in form of **Virtual Organizations (VOs)**. The VOs are based on SOA, where the users access the Grid through **science gateways -** web services that serve as portals between VO users and grid resources, *without obliging mandatory authentication to the grid site*. The VOs are consisted of several *independent autonomous domains*, where the RPs and the users do not reside in the same security domain and each VO has its own **security policy**. Hence, the resource-user relationship imposes three correlated security concerns:

- The independence of the domains where the RPs are situated;
- The need for supporting different security policies;
- The non-necessity of the science gateways for user authentication.

These issues are still being the focal point in many researches, thus the following subsections shortly introduce the basic Grid computing terms which depict their contribution in elimination of the major Grid security concerns.

Brief overview of Web Services and XML

According to the World Wide Web Consortium (W3C), a **Web Service (WS)** is defined as a software system designed to support interoperable machine-to-machine interaction over a network. Hence, the Grid's SOA is based on WSs standards as its underlying technology - **Simple Object Access Protocol (SOAP)** for invocation, **Web Services Description Language (WSDL)** for interface definition, and **Universal Description, Discovery and Integration (UDDI) registries** for service discovery. All of these standards use XML as the communication format and consequently they enable platform- and machine-independent communication via the network.

However, the widespread usage of WSs is still in delay because of the unmanageability in reaching consensus on how to secure the services. This occurs due to the distinct security standards available for each of the security aspects - encryption, message integrity, authentication, authorization, etc. Accordingly, two of the available standards stand out as possible significant contributors in solving these discrepancies - SAML and XACML, which are described in the next subsection.

Brief overview of XACML and SAML

XACML is an XML based WS standard for communicating access control policies between services, which provides: (1) standard XML schema for expressing policies, rules based on those policies and conditions, and (2) a request/response protocol for sending a request and having it approved, where the subjects and the resources are identified using URIs. Accordingly, the **XACML authorization model** contains:

- **PAP (Policy Administration Point)**, which provides security policies or policy sets;
- **PIP (Policy Information Point)**, which provides information regarding attributes of the subjects, the resource, and the environment obtained by querying the PIP;
- **PEP (Policy Enforcement Point)** that intercepts the access requests from users and sends the requests to the PDP;
- **PDP (Policy Decision Point)** that makes access decisions according to the PIP and PAP which are then sent back to the PEP; the PEP then fulfills the obligations (permits or denies the access request) according to the decision of PDP (Figure 1).

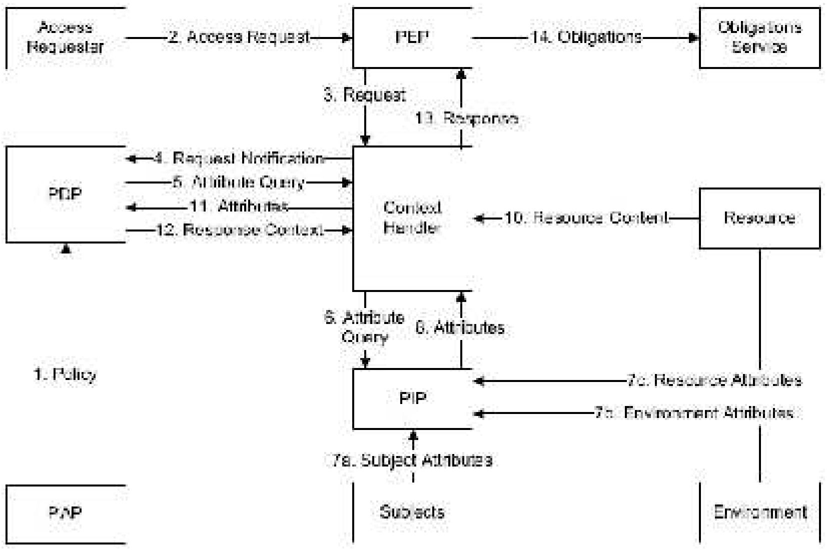

Figure 1: XACML authorization model

XACML also defines a policy language, where policies are organized hierarchically into PolicySets, Policies and Rules, combined using combining algorithms; a rule is composed of a target, an effect and a condition, while a Policy consists of a target, one or more rules, and an optional set of obligations [3].

SAML provides security assertion within a trust domain, based on authentication, attribute and authorization decision information in XML and several request/response protocols. SAML's **Single Sign-On (SSO)** mechanism allows a user to be able to sign on only once at one access point among a group of trusted sites. Accordingly, SAML involves two basic parties in its line of work: (1) Asserting or **Authority Party (AP)** also familiar as the **Identity Provider (IdP)**, that is, the *source site*, which does the assertion job to claim a user and (2) **Relying Party** or the *destination site*, which makes access control decision based on the information supplied to it by the Asserting Party. SAML also provides two profiles for transferring assertions from an Asserting Party to Relying Party and for achieving SSO, known as **push** and **pull mode**. The push mode requires the user to contact an attribute authority service to obtain attribute certificates and "push" them to the target service when submitting a request (Figure 2, left). This approach allows the user to select the specific roles he wants to be authorized with. The pull mode, on the other hand, does not require the users to present any attribute. The attributes are directly retrieved by the resource provider on behalf of the user (Figure 2, right). The pull mode makes attribute retrieval transparent to users [4].

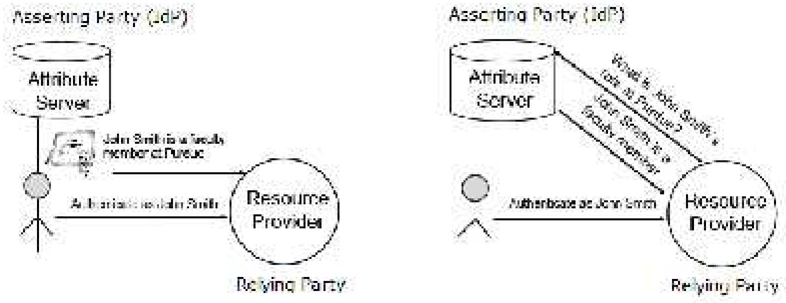
Asserting Party (IdP)
Attribute Server
John Smith is a faculty member at Purdue
Authenticate as John Smith
Resource Provider
Relying Party
Asserting Party (IdP)
Attribute Server
What is John Smith's role at Purdue?
John Smith is a faculty member
Authenticate as John Smith
Resource Provider
Relying Party

Figure 2: Push and Pull mode

FORMER ACCESS CONTROL MODELS AND THEIR DRAWBACKS

As it was mentioned earlier, the main reason for the delay of widespread usage of WSs and implementation of SOA is the unmanageability in reaching consensus on how to secure the services. During the researchers' efforts to explore the best solution about this concern, various access control models have appeared, like it was the case with **IBAC** and **RBAC**.

IBAC was initially developed with the intent of preventing user interference between the machines while committing their resource-sharing tasks. This was done by assigning permission to the users *according to their identities*. The IBAC approach was accomplished through the usage of **Access Control Lists (ACLs)** consisted of user's **Distinguished Name (DN)** mapped to a local grid site account. The account is set up to authorize access only to users whose DN appear on the ACL, according to their account privileges. The DNs are **Lightweight Directory Access Protocol's (LDAP)** referents to the directory server's collection of entries (objects). These are organized in hierarchical manner and have distinct attributes like name, address and type, which are expressed in 'attribute=value' pairs .

However, during many undertaken thorough analyses over time, IBAC showed a diversity of inevitable issues, like **non-manageable trust relationships, role explosions, information leakage, problems with delegation and revocation, misalignment of incentives, inflexibility for scalability, ambient authorities and difficulties with its distributed identity management** [5]. As a result, the administration of IBAC was becoming more untenable and error-prone with the constant increase of the number of user. This headed toward the emergence of new concepts - role, owner and group, which gave birth to the RBAC model.

The basic idea of the RBAC model was the assignment of users to roles, permissions to roles and users acquirement for permissions for being members of roles. A user was assigned to many roles and vice versa. The permission-role assignment was in similar concept. The RBAC model was organized in four levels: (1) **Flat (core)** RBAC level; (2) **Hierarchical -** added requirements for supporting role hierarchies; (3) **Constrained -** added constraints on the hierarchical level and (4) **Symmetric (consolidated) -** added a requirement for permission-role review.

The two main concepts associated with RBAC were: (a) **Role Enablement Authority (REA)**, which was responsible for assignment of subjects to roles and (b) **Role Assignment Policies (RAP)**, which was responsible for handling of role assignment requests for each candidate role. These concepts used different PolicySets defined by XACML in order to implement the role hierarchy.

However, as it was the case with IBAC, RBAC also revealed as flaw-full concept. Namely, it showed **non-correspondence to the constrained level of RBAC** and involved only **partial delegation of the roles** [6, 7]. RBAC’s flaws have led toward developing a new and highly improved access control model, familiar as the ABMAC model, which is discussed in the next section.

ATTRIBUTE-BASED MULTIPOLICY ACCESS CONTROL MODEL

Attribute-Based Multipolicy Access Control (ABMAC) model enables authorization decisions based on the attributes of the resource/service requesters and thus eliminating the flaws in the previous access control models. ABMAC achieves the scalability and flexibility necessities for open distributed networks, such as the Grid. ABMAC functions by replacing the subjects with a set of attributes and the objects with descriptions in an, so called, **Attribute Certificates (ACs)**. ACs specify access control information associated with the certificate holder (e.g., age, citizenship, credit status, group membership, role, security clearance). Thus, the decision to access a resource is based on the attributes in the requestor’s credentials, which must be protected in a similar way as any other certificate. Therefore these are digitally signed sets of attributes created by **Attribute Authorities**.

Formal Definition of ABMAC

The formal definition of ABMAC is composed of four parts: (1) **access control related entities**, **attributes of entities -** attributes define the identity and characteristics of the corresponding entity, (2) **policy representation -** a superset of independent policy units, and (3) **policy evaluation -** process of making an access decision based on the security policy [8]. In short terms, an example of an access request evaluation would be:

$$\begin{aligned}
&ABMAC\_adf(Attr(\mathrm{Re}q), Attr(Sev), Attr(\mathrm{Re}s), Attr(Act), Attr(Env)) \\
&\quad = combine\_f(p_1\_adf(Attr(\mathrm{Re}q), Attr(Sev), Attr(\mathrm{Re}s), Attr(Act), Attr(Env)), \\
&\quad\ p_2\_adf(Attr(\mathrm{Re}q), Attr(Sev), Attr(\mathrm{Re}s), Attr(Act), Attr(Env)), \\
&\quad\ \cdots \quad \cdots \quad , \\
&\quad\ p_m\_adf(Attr(\mathrm{Re}q), Attr(Sev), Attr(\mathrm{Re}s), Attr(Act), Attr(Env))) \\
&\quad = \begin{cases} permit \\ deny \end{cases}
\end{aligned}$$

(

Each shortcut in the definition is briefly explained by the following:

- **Requestor** (Req) **-** sends requests to and invokes actions on the Grid service;
- **Service** (Sev) - is invoked via standard protocols and data formats;
- **Resource** (Res) - always has a specific set of state data expressible as an XML document and a well-defined lifecycle;
- **Action** (Act) - Grid service operation invoked by clients;
- **Environment** (Env) - Grid service invocation, containing additional information;
- **Attributes -** represented as Attr;
- **ABMAC_adf() -** ABMAC decision function that implements a combining algorithm in combine_f(), which combines the decision results returned by the access control decision function **Pi_adf()** of each policy Pi and makes a final access decision.

ABMAC ADVANTAGES

ABMAC's preliminary purpose was to (1) reduce, and possibly – completely remove the inconsistencies, which appeared in the former access control models, (2) enable flexible and scalable support for multiple policies and (3) integration with the existing authorization systems. For that reason, various researchers have found a diversity of approaches to unravel the issues mentioned in Section 2; hence, the following subsections describe their proposed techniques and models, covering each security issue separately or in combination.

Solution to science gateways

Since the Grid is inherently a federated environment, every local site wants to retain its authority on determining who can use its resource. However, the science gateways, with their non-necessity for user authentication, implicated the question: *Without the user's identity, how can the grid site know who is using its resources and whether to grant access or not?* [9]. According to this, three possible access control models were proposed, involving three main entities **-** a **user**, a **service provider (SP)**, i.e. the science gateway, and a **RP**, i.e. the grid site:

- **Complete trust**, where the SP enforces both authentication/authorization and is trusted completely by the RP. The advantage is that a user does not need to obtain a grid account allocation while still being able to enjoy the computational power provided by the distributed infrastructure. The disadvantage is that the grid site cannot control its own access policy (e.g. differentiated service levels) based on users' credentials (Figure 3a);
- **Medium trust**, where the SP still performs authentication, but the final authorization decision is made by the RP. The advantage is that the RP can have its own policy to differentiate its service based on users' attributes and can have better control on how to allocate its computation resource (Figure 3b);

- **No trust**, where both authentication/authorization are executed by the RP. The advantage is that the RP can independently track individual users using their resources. The disadvantage is that only users with allocation on the RP can use the service provided by the SP, which may be too inflexible in practice and will require secure delegation of user credentials to the SP (Figure 3c)

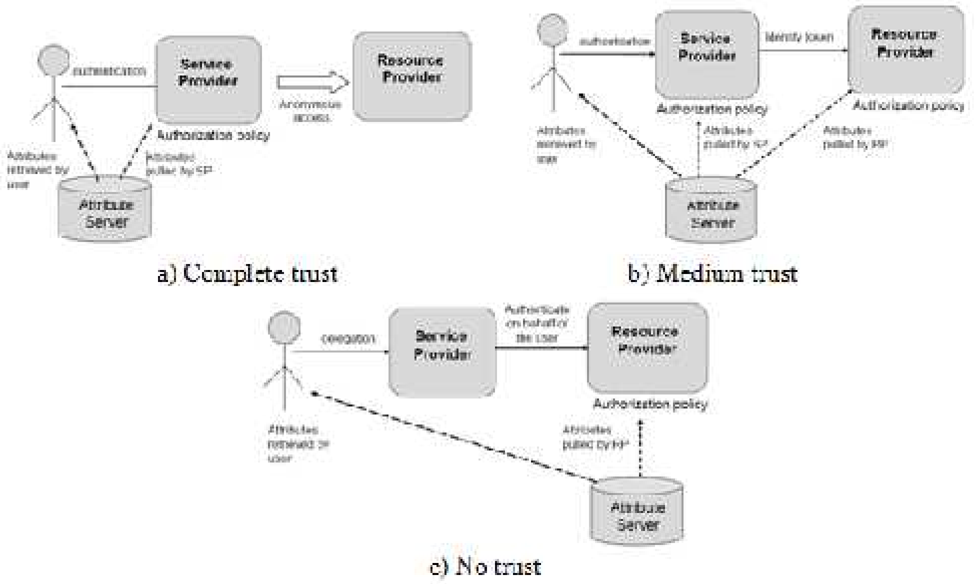

Figure 3: Access control models based on trust

Solution to independent domains

The Grid's open distributed architecture has brought the second security issue to the picture - the independency of the domains. The domains assumed pre-agreed trust between the SP and the user, based on the usage of **Public Key Infrastructure (PKI)** for authorization. This has proved as inflexible to some extent. Hence, another access control model was proposed (Figure 4), where trust is being assumed between SPs and IdPs within a VO. This approach does not rely on PKI user certificates; instead it combines XACML and SAML technologies, i.e. SSO for user authentication. Thus, the SP has full control over the overall authentication process, through the, so called, **AAProxy**. The **Attribute Releaser (AR)** as part of the AAProxy, handles user credentials, after the user has been authenticated, and passes them to the **Authorization Web Service (AuthZ WS)**. The AuthZ WS handles XACML bindings and assembles user attribute contents to XACML requests. The **Request Generator (RG)** generates a request, which is evaluated by the PDP and PEP and then sent as request decision to the AAProxy. The LDAP in this case is the attribute server, where the user attributes are being stored and from which they are being retrieved [10].

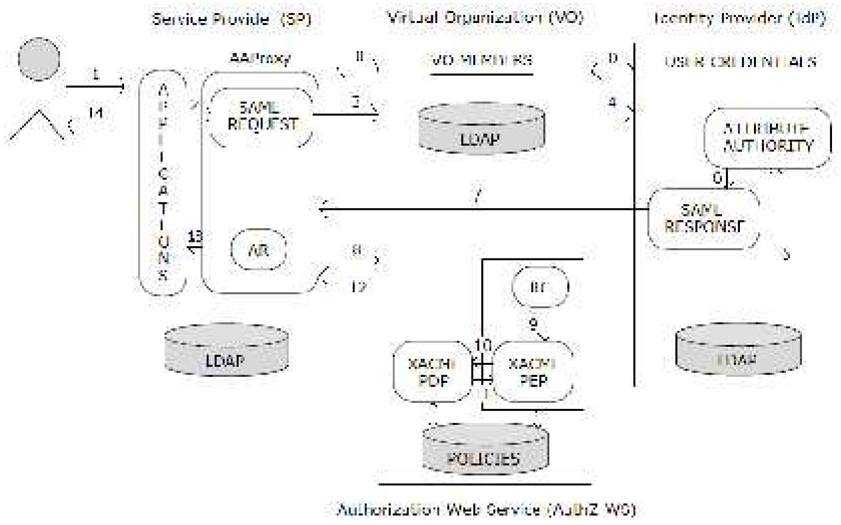

Figure 4: Authentication/Authorization Model Based on XACML and SAML

Solution to multiple policies

As it was previously portrayed, in grid environments, the VOs are consisted of several independent domains, where each VO has its own security policy that can be changed dynamically. Thus, these policies are encapsulated, i.e., they have their own definitions and decision-making algorithms. Having the XACML authorization model as in Figure 1 would lead to inflexibility of the Grid to support multiple policies. Primarily, it would force unified method for policy description. The scalability would be affected also, since it would not accept integration with the existing authorization systems. Hence, the ABMAC model through extension of the existing XACML authorization model, and its association with SAML has solved this issue. It is also practically implemented in the **Globus Toolkit 4 (GT4)**, the current de facto solution for Grid security. To be precise, in **GT4 Authorization Framework** (Figure 5), a **Master PDP** is created, with specific policies as its subclasses. These implement common interface "canAccess()", where the Master PDP combines a single access request decision from its set of diverse policies. One of the policies implemented under the Master PDP is the **SAMLAuthorizationCallout PDP**, which enables the needed integration with third-party authorization systems, like Shibboleth, VOMS and PERMIS [11].

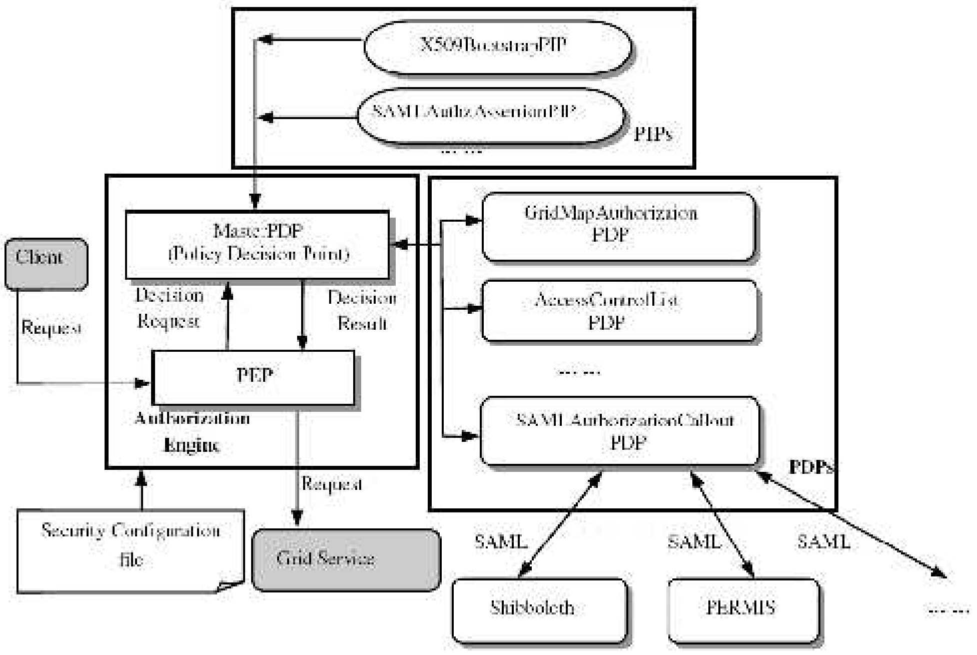

Figure 5: GT4 Authorization Framework

CONCLUSION

This paper has demonstrated how the three proposed attribute-based access control models have provided the flexibility and scalability necessary for large-scale distributed systems such as the Grid, by making access decisions based on the attributes of the involved entities. The proposed solutions have eliminated the misconceptions of the former access control models with the issues of science gateways, independent domains and the multipolicy support. Additionally, by being already practically implemented in the GT4 authorization framework, the concepts of cross-domain secure communication, support for multiple policies and integration of third-party attribute-based authorization systems have been enabled by now. Therefore, it is only a matter of time when these approaches will become mature enough to be widespread everywhere.

REFERENCES

[33] Harris S., (2005) "CISSP All-In-One Exam Guide", McGraw-Hill Osborne Media, 3rd edition, p.124-129

[34] Damiani E., De Capitani di Vimercati S. and Samarati P., (2005) "New Paradigms for Access Control in Open Environments", In Proc. 5th IEEE International Symposium on Signal Processing and Information, Athens, Greece

[35] Lang B., Foster I., Siebenlist F., Ananthakrishnan R. and Freeman T., (2006) "Attribute Based Access Control for Grid Computing"

[36] Ou X., Squicciarini A. C., Goasguen S and Bertino E., (2006) "Authorization Strategies for Virtualized Environements in Grid Computing Systems", IEEE Workshop on Web Services Security (WSSS), Berkeley, California, USA

[37] Karp A. H., (2006) "Authorization-Based Access Control for the Services Oriented Architecture", Fourth International Conference on Creating, Connecting, and Collaborating through Computing, HP Laboratories Palo Alto

[38] Haidar D. A., Boulahia N. C., Cuppens F. and Debar H., (2006) "An extended RBAC profile of XACML", Proceedings of the 3rd ACM Workshop on Secure Web Services SWS '06

[39] Karp A. H., Haury H. and Davis M. H., (2009) "From ABAC to ZBAC: The Evolution of Access Control Models", Technical report, HP Laboratories Palo Alto

[40] Lang B., Foster I., Siebenlist F., Ananthakrishnan R. and Freeman T., (2006) "A Multipolicy Authorization Framework for Grid Security", Proceedings of the Fifth IEEE International Symposium on Network Computing and Applications, p.269-272

[41] Squicciarini A. C., Bertino E. and Goasguen S., (2007) "Access Control Strategies for Virtualized Environments in Grid Computing Systems", Proceedings of the 11th IEEE International Workshop on Future Trends of Distributed Computing Systems, p.48-54

[42] Khider H., Osman T. and Sherkat N., (2010) "Attribute-Based Authorization for Grid Computing", International Conference on Intelligent Systems, Modelling and Simulation, p.71-74

[43] Singh D., Gupta B., Acharya B. M. and Hota S., (2011) "An authorization Framework for Grid Security using GT4", IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 1