The ISC Int'l Journal of Information Security
ISeCure

January 2015, Volume 7, Number 1 (pp. 1–19)
http://www.isecure-journal.org

# A Combination of Semantic and Attribute based Access Control Model for Virtual Organizations

Morteza Amini [1,*], and Majid Arasteh [2]
[1] Data & Network Security Lab (DNSL), Department of Computer Engineering, Sharif University of Technology, Tehran, Iran
[2] Department of Computer Engineering, Sharif University of Technology, Tehran, Iran

ARTICLE INFO.

*Article history:*
**Received:** ***
**Revised:** ***
**Accepted:** ***
**Published Online:** ***

*Keywords:*
Virtual Organization, Semantic Web, Access Control, ABAC, SBAC.

ABSTRACT

A Virtual Organization (VO) consists of some real organizations with common interests, which aims to provide inter organizational associations to reach some common goals by sharing their resources with each other. Providing security mechanisms, and especially a suitable access control mechanism, which enforces the defined security policy is a necessary requirement in VOs. Since VO is a complex environment with the huge number of users and resources, traditional access control models cannot satisfy VOs security requirements. Most of the current proposals are basically based on the attributes of users and resources. In this paper, we suggest to use a combination of the semantic based access control (SBAC) model, and the attribute based access control (ABAC) model with the shared ontology of subjects' attributes in VOs. In this model, each participating organization makes its access control decisions according to an enhanced model of the ABAC model. However, access decision in the VO is made in more abstract level through an enhanced model of the SBAC model. Using the ontology of users and resources in this model facilitates access control in large scale VOs with numerous organizations. By the combination of SBAC and ABAC, we attain their benefits and eliminate their shortcomings. In order to show the applicability of the proposed model, an access control system, based on the proposed model, has been implemented in Java using available APIs, including Sun's XACML API, Jena, Pellet, and Protégé.

## 1 Introduction

The term of Virtual Organization (VO) was first introduced by Mowshowitz in 1986 [1]. In literature, virtual organizations are also introduced as Virtual Corporations, Virtual Enterprises, and Virtual Companies [1]. Virtual organization is a technology that enables real organizations to confederate their resources to reach their common goals [2]. A VO consists of diverse resources, geographically distributed, temporal, and dynamic organizations [3, 4]. Resource sharing and collaboration among parties are the main goals of VOs; hence, providing security, especially through a suitable access control mechanism, is an important challenge in VOs.

In the life cycle of VOs there are four stages [2, 5–7]. At the initial stage, each organization forms a profile that contains its objectives, subjects, available

[*] Corresponding author.
Email addresses: amini@sharif.edu (M. Amini), arasteh@cert.sharif.edu (M. Arasteh).
ISSN: 2008-2045 © 2015 ISC. All rights reserved.

objects, and policies. At the formation stage, the VO manager selects suitable organizations according to the available profiles, and sends an invitation request to them for joining the VO. The first level of access control is done in this stage (i.e. over joining the members). At the operation stage, shared resources will be utilized by the members of the VO. The second and the most important level of access control (over the resources) is done in the operation stage. When a VO reaches its objectives, at the dissolution stage, the VO ends its activities. In this paper, we focus on the second level of access control specified above; i.e. the operation stage.

Three kinds of topologies are introduced for VOs in the literature [8, 9]. These topologies are as follows:

(1) Supply-chain VOs in manufacturing industries,
(2) Star VOs in construction industries,
(3) Peer to Peer VOs in creative and knowledge industries.

In the peer to peer topology, all nodes have direct relationships with each other without any hierarchy. This topology is the most common topology. To propose our access control model, we consider the VOs that are formed based on the peer to peer topology.

A VO has various kinds of resources such as processor, storage, database, and software. Access control is a security service, which is used to prevent unauthorized access to such resources. In this paper, we introduce a new combinatorial model for access control in VOs that makes decisions through the combination of the SBAC and ABAC models. In fact, the access control process is performed in two levels; real-organization-level (using users' attributes) and abstract inter-organization-level (using an ontology of subjects and resources existing in the environment). In inter-organization-level, the ontology is leveraged for interoperability of organizations participating in forming the VO. Abstraction of access control (and its required rules) in this level, makes the model more practical and easy to administrate in this level.

Although, performing access control in two levels is proposed in other related works, they are using inappropriate traditional access control models (especially DAC and RBAC models) for the two levels. The main focus of these works are on access control mechanism not access control model, which suitably satisfies the security requirements of VOs. In this paper, we enhance and leverage appropriate access control models for these two levels; i.e., SBAC for VO-level and ABAC for organization-level. As it is described more precisely in the rest of this paper, by combining the SBAC and ABAC models, we enhanced and improved these two models to be leveraged in a hierarchical structure for access control in VOs. Note that the enhanced SBAC model, which is introduced in this paper, is based on more powerful logic and semantics in comparison with the original SBAC, by modeling the permissible and prohibited actions as relations in the ontology and using SWRL for policy specification and inference. In fact, in original SBAC model, the policy inference rules do not have a formal semantics; however, in the enhanced model of SBAC, it is based on the semantics of SWRL rules. Also the ABAC model leveraged in this paper (for access control in organization-level) is enriched by adding policy inference to the model. It is the result of sharing the ontology employed in the SBAC model with the ABAC model.

In the rest of this paper, the related work is reviewed in Section 2. In Section 3 the necessity of proposing new access control model for VOs is explained, and narrative description of the proposed combinatorial model is described. Section 4, formally specifies the proposed access control model (named SABAC) for VOs. Section 5, describes a case study for clarifying the proposed model with some simple examples and scenarios. In Section 6, the implementation of a prototype of an access control system based on the SABAC model is described. Also the evaluation of the model is presented in this section. Finally, the last section concludes the paper.

## 2 Related Work

Discretionary Access Control (DAC), Mandatory Access Control (MAC), and Role Based Access Control (RBAC) are three kinds of traditional access control models, which are making access control decisions according to the users' identities [10]. Among the traditional models, RBAC is more flexible than DAC and MAC [11]. RBAC is simple and easy to use; but it still has some disadvantages, such as: not being scalable enough, and not being dynamic [12]. In order to increase the RBAC flexibilities some other features such as trust and context were used beside the plain RBAC by preserving its main specifications and functionalities such as separation of duty. For example, Extended RBAC (E-RBAC) was introduced by [13] to assign roles to the users by the consideration of their context. For instance, the location of users as well as trust could be used for this purpose. In [2], both trust and reputation were used to make dynamic and fair access control decisions in virtual organizations. Some other access control models such as credential based access control model (CBAC) were introduced for open systems with a great number of users. In these models users can receive their required permissions according to their certificates. In CBAC, users' certificates would be in the format of X.509 [14]. In this model we cannot make a significant discrepancy among authentication and authorization due to an abundant number of users; while, they may have similar certificates which are issued by a trusted CA. Park and Sandhu [15] introduced Usage CONtrol (UCON) which is doing authorization even after the granting of permissions which is called ongoing authorization. UCON consists of eight important components where conditions and obligations are more important than the others. Obligations denote the prerequisites that a user should have before sending his requests.

To address the problems mentioned for RBAC, especially in distributed environments, Attribute Based Access Control (ABAC) model was introduced [16]. ABAC is an evolved form of RBAC that enforces access control decisions according to the attributes of the users and resources. The attributes might be static like users' usernames, and might be dynamic like age. Since ABAC does not depend on users' identities, it is more suitable for distributed and collaborative systems where the number of users would be many and in some cases they are unknown [12]. In the ABAC model, access control decisions are associated with the subject, object, requested operations, and, in some cases, environment conditions against policy, rules, or relationships that describe the allowable operations for a given set of attributes [17]. Moreover, ABAC by combining the various attributes of authorization elements such as subjects, objects, actions, as well as context provides a fine-grained access control [18]. XACML policy specification language [19] defines an authorization structure, which is generic enough to implement the ABAC model. Some other access control models are also introduced which are mainly used in cloud.

The above models are not suitable for complicated environments such as virtual organizations [20], and also they are not abstract enough to be leveraged for inter-organization policy specification in virtual organizations. The semantic-based access control model (SBAC) [11, 17] was suggested to provide conceptual-level policy specification and reasoning to address such problems. SBAC makes access control decisions according to the credentials that requesters offer. SBAC uses ontology along the three domains of access control, namely subjects, objects, and actions [21, 22]. In SBAC, identification of users is not mandatory [11]; whereas, the verification of provided credentials is required. Moreover, this model facilitates access control management in distributed and large environments [11]. In literature, SBAC is also named as Semantic Aware Access Control (SAAC) [16] and Semantic Access Control (SAC) [11].

Different semantic powered models have been proposed during the last decade. Cirio *et al.* in [16] extend RBAC by semantic web technology. To this aim, an ontology for four classes including Role, Resource, Privilege, and Action is described. Durbeck *et al.* [23], extended XACML general access control scenario by adding semantic web concepts to its architecture. In their proposed architecture, PIP interacts with ontology and inference engines module for preparing precise information to PDP. In [24], XACML is also extended by adding a module to the general architecture of XACML. This module directly connects to the context handler and makes inference from the ontology.

In ontology-based access control model (OBAC) [25] for semantic web services, instead of PDP, "Ontology-based decision engine (OBDE)" is used. The OBDE uses the ontology and authorization policies for making decisions. Authorization decisions are made according to the attributes of users, resources, and environments. Shen in [26], provides a semantic aware attribute based access control model for web services (SABAC). Heuses ABAC for describing the access control policies and uses OWL for representing the ontology of users and resources. He also introduces a complementary model, named S-ABAC for web services [27]. S-ABAC is presented by combining the Attribute and Semantic based access control models. S-ABAC can realize semantic and attribute-based access control, by extending XACML architecture and representing the attributes of the resources and users semantically using an ontology.

Since yet, some access control models presented for VOs; however, they have some limitations in practice. Condor and Legion [28, 29] are two common grid computing middlewares that enable us to create VOs. Condor uses similar resource management mechanism such as UNIX. Beside the read and write privileges, Condor provides more access control modes for making decisions (e.g. reading, writing, administrating, configuring, and owning). Legion is an object oriented middleware for Grid environments where resources are considered as objects and access to them are done through the functions defined on objects. In Legion, each object is responsible for enforcing its own access control policies. Both above access control systems are based on the traditional models and depend on the identities of users and resources. Hence, the VO manager cannot define high level or coarse-grain policies. This makes access control management very difficult in these systems.

OGSA is a framework developed by Global Grid Forum (GGF) [29–31]. Globus toolkit is the reference implementation of OGSA. Globus uses Gridmap for mapping users' identities in the VO to their local identities [32]. Each resource provider in the VO should maintain an access control list (ACL), which contains users' identities and their permissions. Any update in the VO policies and the users should apply in all resource providers Gridmap file, otherwise conflicts would happen. To solve these problems, authorization systems like CAS, VOMS, PERMIS, Shibboleth, and Akenti were introduced [31, 32]. All of these access control systems are based on DAC models and thus they have similar problems to the ones mentioned for Condor and Legion.

In addition, in literature some other forms of access control are introduced that are making access control decisions in two levels which are also known as intro-domain and inter-domain decisions [29]. Each domain has its own access control policy and in order to make dynamic decision, intro-domain trust among the entities of each domain was introduced. If an entity from a domain wants to make a connection with another entity in the other domain, then mapping is done by another module through the consideration of inter-domain trust. Similarly in [33], intra-domain and inter-domain trust were introduced to provide a secure transaction with other nodes. By taking trust into account, each node can make a secure connection with other ones. The main focuses of the paper is on making a secure connection than proposing an access control model or mechanism.

In [2], we suggested an access control model which is employed in two levels of VO and resource provider. In order to provide a dynamic decision, both trust and reputation were used by resource providers and VO manager, respectively. It is important to note that using trust and reputation provides soft security not the hard one, which we approached to provide it through access control policies in our proposed model.

Push and pull models are two basic approaches for credential-based authorization in distributed systems, such as VOs [9, 31, 32, 34]. In the push model, a user sends his request to the authorization system. After authorization, the system issues and returns an access certificate to the user; then the user pushes the certificate to the resource provider. Some authorization systems, like CAS and VOMS, support the push model [35]. In the pull model, a user sends his request directly to the resource provider; then the resource provider forwards the request to the authorization system. After authorization, the system issues and pulls the users' permission as a certificate to the resource provider. Some authorization systems such as Akenti [35] support the pull model. In both the push and pull models, resource providers make the final decision on granting or denying the access requests of the users.

## 3 Proposed Access Control Model – Narrative Description

Virtual organizations have some important features that impose new security requirements. These new requirements motivated us to introduce a new access control model which can address them. Some of the important features and security requirements of VOs are as follows.

- A VO is a complex environment with numerous users whose identities might be unknown to the VO or to its resource providers. In fact, the identities of the users of one organization are unknown to the other organizations cooperating in the VO.
- There are many different and heterogeneous resources shared by the resource providers in the VO.
- In each VO, access control might be done in two levels. The first level of access control is done by the VO manager, and the second level is done by the resource providers. The VO manager is holistic and makes general access control decisions, whereas the resource providers are more precise than the VO manager, and make fine grain decisions.

We need an access control model and mechanism that can address the aforementioned requirements. Current access control mechanisms for VOs are mostly based on the ABAC model. ABAC is a suitable model for making precise and fine grain decisions, but it is not scalable enough for the great amount of users and does not consider semantic relationships (which are specified as an ontology in a distributed and heterogeneous environment with interoperability of its element) in its decision making process.

We suggest using a combination of ABAC and SBAC models for describing access control policies in VOs. In the rest of this section, we briefly introduce the ABAC and SBAC models and describe their advantages and disadvantages. A combination of these two models forms the basis of our proposed model for access control in VOs. SBAC is suitable for VO-level access control and ABAC is suitable for organization-level access control. Also the combination of these two models with the shared ontology of subjects (users' attributes) enables us to infer implicit policy rules from the explicit ones in both level of access control.

### 3.1 ABAC and XACML

Attribute based access control (ABAC) is a flexible access control model, providing access to resources based on the evaluation of attributes (of subjects and/or resources). XACML is an OASIS standard XML-based language, which is used to specify access control polices based on the ABAC model [18].

In fact, XACML is a general purpose policy system, designed to support the needs of most authorization systems. At its core, XACML defines the syntax for a policy language and the semantics for processing those policies. It supports both fine grain and complex policies [26, 36]. OpenXACML, enterprise-java-xacml, HERAS XACML, JBoss XACML, and Sun's XACML are current implementations of XACML specification [23].

Using XACML for ABAC policy specification in VOs has advantages that some of them are as follows:

- It is a standard language which is used in the organizations for describing their own access control policies. Standardization ensures interoperability between different communities.
- Some special tools like "UMU XACML" are available for writing access control policies in XACML easily.
- XACML is based on the ABAC model, which makes it flexible to specify and enforce fine grain authorization.
- It is economical for both organization managers and developers. Because XACML is a standard language, managers can reuse the policies defined in other organizations, and also developers can write new policies or reuse the current policies easily.
- XACML provides conflict resolution in its combining algorithms. The supported conflict resolution strategies are deny-override, permit-override, first-applicable, ordered-deny-override, ordered-permit-override, deny-unless-permit, and permit-unless-deny [18].

Besides the advantages of ABAC and XACML, they have some shortcomings, which are motivated us to complement it with a SBAC model to be appropriate for application in VOs.

- XACML does not have enough scalability for distributed systems. It is a suitable solution for organizations with a few numbers of subjects and objects.
- XACML does not consider semantic relationships (for inferring implicit policies from the explicit ones) and does not make decisions semantically.

For representing policies, the XACML language introduces the following main components [37]:

(1) `<PolicySet>` contains a set of access control policies or other policy sets.
(2) `<Policy>` represents an access control policy described through a set of rules.
(3) `<Rule>` represents an access rule or permission. The effect of a rule can be either permit or deny.

Figure 1 shows a simple example of XACML policy. Here we define a policy set (myPolicySet) which has a policy (myPolicy). The policy says a subject (User) has Read permission on Resource, in which Read is an action and Resource is an object.

In XACML, `<PolicySet>`, `<Policy>` and `<Rule>` may contain a `<Target>` element. The `<Target>` element specifies a set of subjects, resources, actions and environments to which the `<PolicySet>`, `<Policy>` and `<Rule>` are applied [38].

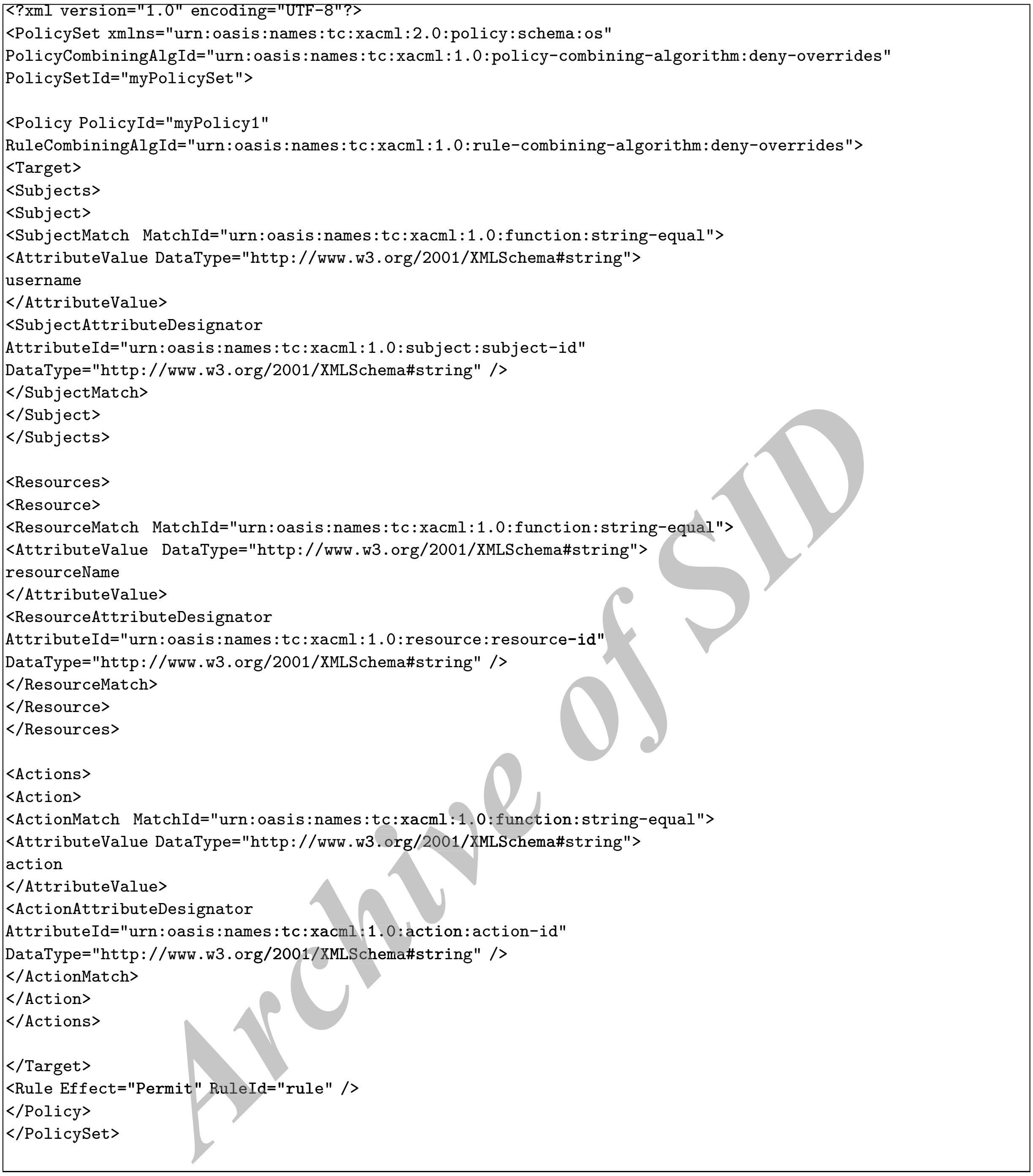
### 3.2 SBAC and SWRL

The term of semantic web was introduced by Tim Berners-Lee in 1998. Elevating human readable information on the web with meta data and precise semantics was his main purpose [10, 24]. Semantic web also tries to provide interoperation between heterogeneous applications. In the existing access control models, like ABAC, access control is done through the rules that are explicitly expressing which users are allowed to access which resources. By introducing an ontology on the elements engaging in the access control procedure,we just need to define the explicit policies; whereas, the implicit ones can be inferred through the semantic relationships defined in the ontology.

In the Semantic based Access Control (SBAC) model, access control decisions are made according to the semantic relationships defined among entities [21]. SBAC was developed to facilitate the access control management; by preserving simplicity, correction, and safety [11]. Ontology and Rule description language are the main parts of the SBAC. Ontology is used to prepare common perception among all organizations and enables us to describe the concepts of a certain domain and their relationships [37].

In a Virtual Organization, the VO manager can define its concepts; i.e., subjects, objects, actions, and conditions, and the hierarchy among the concepts. In order to specify the concepts and their relationships, Ontology Web Language (OWL) was introduced [25, 37]. OWL also can describe instances of concepts, and provide mechanisms for simple reasoning [26]. In this paper, OWL is used for specification of the concepts of subjects, objects, and actions and their relationships. It is used to describe the individuals and their attributes as well.

Semantic Web Rule Language (SWRL) is a Horn clause rule language, which is used to write rules in terms of OWL concepts and can reason about OWL individuals [25, 26]. In our proposed model, we enhance the SBAC model to leverage SWRL for policy rule specification to improve its expressiveness and



```
<?xml version="1.0" encoding="UTF-8"?>
<PolicySet xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
PolicyCombiningAlgId="urn:oasis:names:tc:xacml:1.0:policy-combining-algorithm:deny-overrides"
PolicySetId="myPolicySet">

<Policy PolicyId="myPolicy1"
RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:deny-overrides">
<Target>
<Subjects>
<Subject>
<SubjectMatch  MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
<AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
username
</AttributeValue>
<SubjectAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
DataType="http://www.w3.org/2001/XMLSchema#string" />
</SubjectMatch>
</Subject>
</Subjects>

<Resources>
<Resource>
<ResourceMatch  MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
<AttributeValue  DataType="http://www.w3.org/2001/XMLSchema#string">
resourceName
</AttributeValue>
<ResourceAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
DataType="http://www.w3.org/2001/XMLSchema#string" />
</ResourceMatch>
</Resource>
</Resources>

<Actions>
<Action>
<ActionMatch  MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
<AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
action
</AttributeValue>
<ActionAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
DataType="http://www.w3.org/2001/XMLSchema#string" />
</ActionMatch>
</Action>
</Actions>

</Target>
<Rule Effect="Permit" RuleId="rule" />
</Policy>
</PolicySet>
```

**Figure 1**. Simple example of XACML policy syntax.

inference ability. By SWRL, the VO manager can define which types of subjects can access which types of objects.

### 3.3 Proposed Model: A Combination of ABAC and SBAC

To enjoy the advantages of the ABAC model and eliminate its disadvantages, we use a combination of the ABAC and SBAC models named SABAC for access control in VOs. Figure 2 shows an access control framework based on our suggested access control model for VOs. Current access control models for VOs are based on either push or pull models. Although the framework shown in this figure follows pull model, it can be easily adopted for push model, too. Thus, the proposed model can be employed as both of push and pull models.

In the proposed model, we have two levels of access control; organization-level access control (based

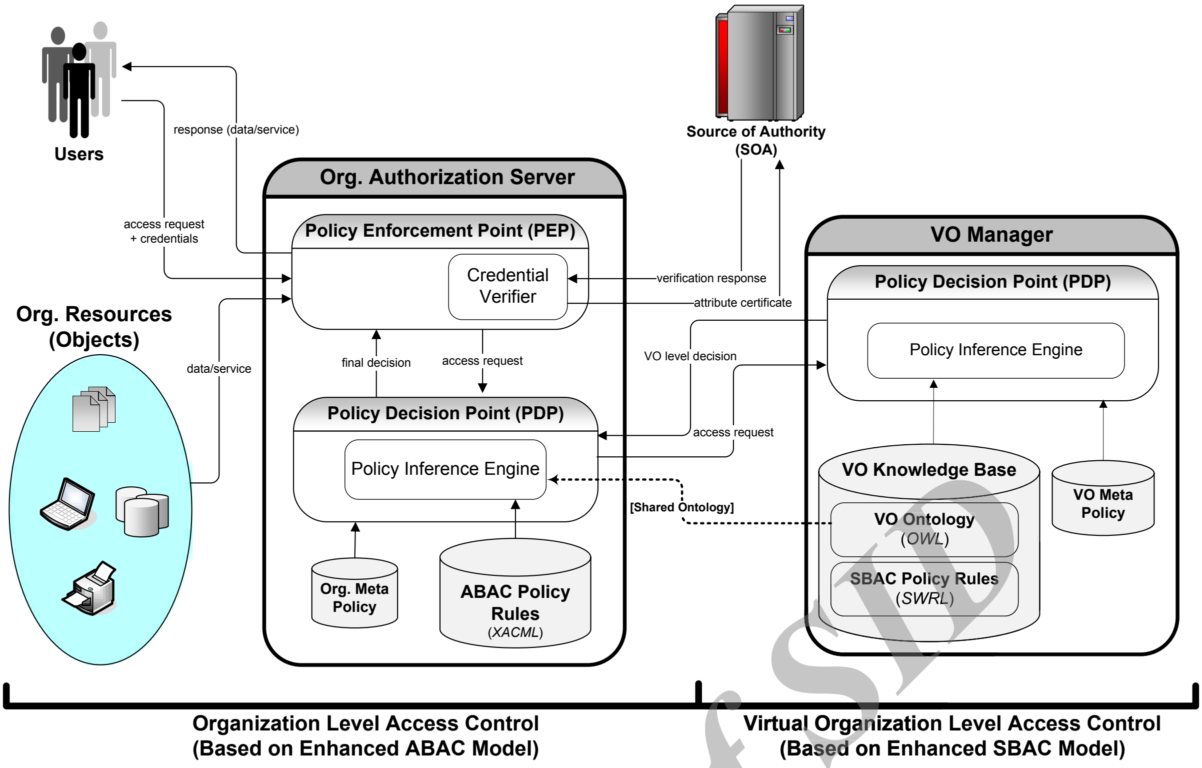


**Figure 2**. An access control framework based on the proposed SABAC model for VOs.

on the enhanced ABAC model) and VO-level access control (based on the enhanced SBAC model). Each participating organization has a limited number of resources and users. Thus, ABAC is a useful access control model for organizations. Each organization manager can easily define its organization's access control policies in XACML in organization's authorization server. A VO consists of many different kinds of organizations, users (maybe with unknown identities), and resources. Since ABAC does not make decisions semantically, and is not scalable enough for large environments, it is not solely an appropriate model for VOs. Due to this fact, we suggest using SBAC for VOs; while, ABAC is used for the participating organizations (resource providers). In this case, the VO manager can define access policies in the conceptual (and more abstract) level.

Regarding the SBAC model, first, we should define concepts, individuals, and taxonomies in an ontology. Then, by SWRL we can express access control policies easily. Note that in the proposed model, the ontology of subjects (which is used in the SBAC model in VO-level access control) and the taxonomy of attributes (which is used in the ABAC model for organization-level access control) are the same. In fact, in this model, each subject is considered as an attribute holder. Also it enjoys the relationships defined on different types of the actions in the ontology. This enables the access control system to have policy inference in the enhanced ABAC model in organization-level as well. More details of the proposed model are provided in the next section.

## 4 SABAC Model Formal Specification

Following the narrative description provided in the previous section, in the rest, the formal specification of the basic elements of the proposed model as well the policy inference and access control procedure are presented. As mentioned in previous section, the SABAC model employs the enhanced version of SBAC and ABAC models. Detailed specification of the proposed model clarifies its differences with the original SBAC and ABAC models.

### 4.1 Basic Elements of the Model

The basic elements of the SABAC model are presented in Figure 3. These elements are specified formally in the following definitions.

**Definition 1 (SABAC Model).** The SABAC model is defined as a binary tuple $\langle VO, Orgs\rangle$, where:

(1) $VO = \langle VOO, VOP\rangle$, denotes a Virtual Organization that consists the following elements.
- VOO specifies the ontology of subjects, objects (resources), and permissible and pro-

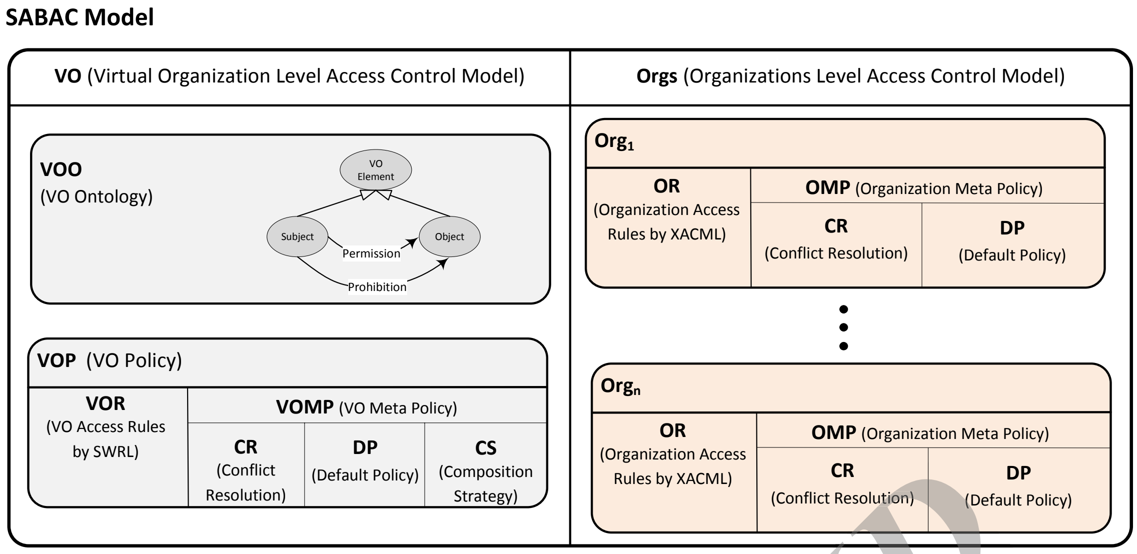


**Figure 3**. SABAC data model in one glance.

hibited actions in the VO (see Definition 4.2). In fact, the ontology provides a common conceptualization of elements engaging in access control for all organizations participating in the VO.

- VOP is the set of the VO's policy rules (in the form of the SWRL rules), which are defined by the VO manager according to the enhanced SBAC model (see Definition 3).

(2) $Orgs = \{Org_i | Org_i = \langle ID_i, OP_i \rangle\}$, represents a set of participating organizations. Each organization $Org_i$ is specified by the following elements.

- $ID_i$ is the identity of the joined organization.
- $OP_i$ is the set of the organization's policy rules (in the form of the XACML policy format),which are defined according to the ABAC model (see Definition 4).

The ontology which is used in the VO-level is in fact the abstract specification of the entities participating in access control in the VO. Using this ontology, the VO manager is capable of specifying high-level access policies based on the concepts specified in the ontology; regardless of the details of the resources existing in the organizations. For example, the VO manager can diminish access to the project management data by the remote users. Such data and such users are defined in the ontology in the conceptual level regardless of the identities and details of the individuals belong to them. Furthermore, the semantic relationships between the concepts in the abstract level are considered in inferring implicit policies from the explicit ones.

**Definition 2 (Virtual Organization Ontology–VOO).** The VOO that is leveraged for policy specification and inference in this model contains two basic concepts and two basic relations in top level; concepts *Subject* and *Object*, and relations *Permission* and *Prohibition* (see Figure 4).

- *Subject*: the concepts subsumed by the *Subject* concept in VOO are in fact the attributes (in terms of some credentials provided to show the eligibility) of the users who require to access the resources in the VO. Figure 3 shows a sample VOO. The concepts defined under the *Subject* concept (representing the attributes of users) are leveraged for attribute-based policy specification (following the ABAC model) in organization-level as well.
- *Object*: the concepts specified under the *Object* concept specify the shared resources of participating organizations in the VO. The VO manager's high-level policies (specified in VOP) are defined upon these concepts.
- *Permission*: the relation between subjects and objects to specify the permissible or authorized actions that the individuals of a specific subject can perform on the individuals of a specific object. Each permissible action (e.g., *CanRead*) should be defined as a relation subsumed by *Permission*.
- *Prohibition*: the relation between subjects and objects to specify the prohibitions or unauthorized actions that the individuals of a specific subject are not permitted to do on the individuals of a specific object. Each prohibited action (e.g., *CannotRead*) should be defined as a relation subsumed by *Prohibition*.

**Note 1:** It is important to note that although we should define permissions as complement of prohibitions logically (i.e., $Prohibition \equiv Permission$); we cannot define such a property in VOO explicitly. Because, if we define such a property in VOO, in case of defining conflicting access rules, the rule-base becomes inconsistent. This is resulted from the fact that the underlying logics (i.e., description logic and first-order logic) are monotonic (for more details refer to [39]). In order to mitigate such a problem, we should add some facts manually in the VOO for each type of permissions (or prohibitions) which is discussed in the policy inference section.

**Definition 3 (Virtual Organization-level Policy – VOP).** The VO policy, denoted by $VOP = \langle VOR, VOMP\rangle$, is a binary tuple containing the following elements.

(1) VOR as a set of SWRL rules in one of the following forms, where $Sub \sqsubseteq Subject, Obj \sqsubseteq Object$, $CanAction \sqsubseteq Permission$, $CannotAction \sqsubseteq Prohibition$, and $ContexConds$ is an expression specifying the contextual conditions using the properties defined in the ontology:

$$Sub(?s) \wedge Obj(?o)[\wedge ContexConds] \rightarrow CanAction(?s, ?o)$$

$$Sub(?s) \wedge Obj(?o)[\wedge ContexConds] \rightarrow CannotAction(?s, ?o)$$

Note that in the above format of access rules, specifying contextual conditions is optional. For the sake of simplicity, in the rest of this paper, we do not consider the contextual conditions.

(2) $VOMP = \langle CR, DP, CS\rangle$ contains three metapolicies; conflict resolution policy, default policy, and policy (decision) composition strategy.

$VOMP.CR \in \{DO, PO\}$ as a conflict resolution metapolicy is used for conflict resolution in VO-level access control. The strategy chosen for conflict resolution would be one of the following ones:

- $DO$ determines that deny rules override the other ones.
- $PO$ determines that permit rules override the other ones.

$VOMP.DP \in \{Permit, Deny\}$ as a default policy, determines the final decision in case of inferring neither permission nor prohibition for a requested action.

$$VOMP.CS \in [Union, Intersection, VO.Override, Org.Override]$$

as a policy composition strategy, determines how the policies of the VO-level and organization-level should be composed and final decision in access control should be made. In Union and Intersection strategies, union of permissions and intersection of permissions are considered, respectively. The other two strategies determine the more priority of VO-level and organization-level policies, respectively. Table 2 shows the effect of employing these composition strategies.

Note that in the above definition $Sub$ and $Obj$ could be a simple or complex concept. For example if we have $Student \sqsubseteq Subject$, it is trivial that the complex concept $\exists hasCourse.Student$ is subsumed by $Subject$. Thus, we can define the following access rule:

$$\exists\ hasCourse.Student(?s)\ \wedge ClassServer(?o) \rightarrow CanConnect(?s, ?o)$$

As defined above, each organization participated in a VO, has its own access policy rules. These access rules should be enforced in addition to enforcing the access policy rules defined in the VO-level, when accessing the resources provided by the organization.

**Definition 4 (Organization-level Policy – OP).** An organization policy $OP_i$ is defined as $OP_i = \langle OR, OMP\rangle$, where:

(1) $OR$ is a set of organization rules defining which subjects (with which attributes) in which situations can access which resources (with which attributes) following the XACML format described in the ABAC model.

```
<Policy PolicyId=" " RuleCombiningAlgId=" ">
<Target>
<Subjects></Subjects>
<Resources></Resources>
<Actions></Actions>
</Target>
<Rule Effect=" " RuleId=" " />
</Policy>
```

As we described earlier in this paper, the subject attributes which can be employed for policy specification in organization-level policy are defined in VOO under the Subject concept. Although an organization can add more attributes to its customized ontology of subject attributes.

(2) $OMP = \langle CR, DP\rangle$ contains two metapolicies; conflict resolution policy and default policy.

$OMP.CR \in \{DO, PO, FA\}$, is a metapolicy, which is used for conflict resolution in inter-organization-level access control. The strategy chosen for conflict resolution would be one of the following ones:

- $DO$ determines that deny overrides.
- $PO$ determines that permit overrides.
- $FA$ means that the first one is applicable. According to this strategy, all policies are ordered according to their importance or priority, and the first applicable rule is enforced.

$OMP.DP \in \{Permit, Deny\}$ as a default policy, determines the final decision in case of existing no rule for a requested action.

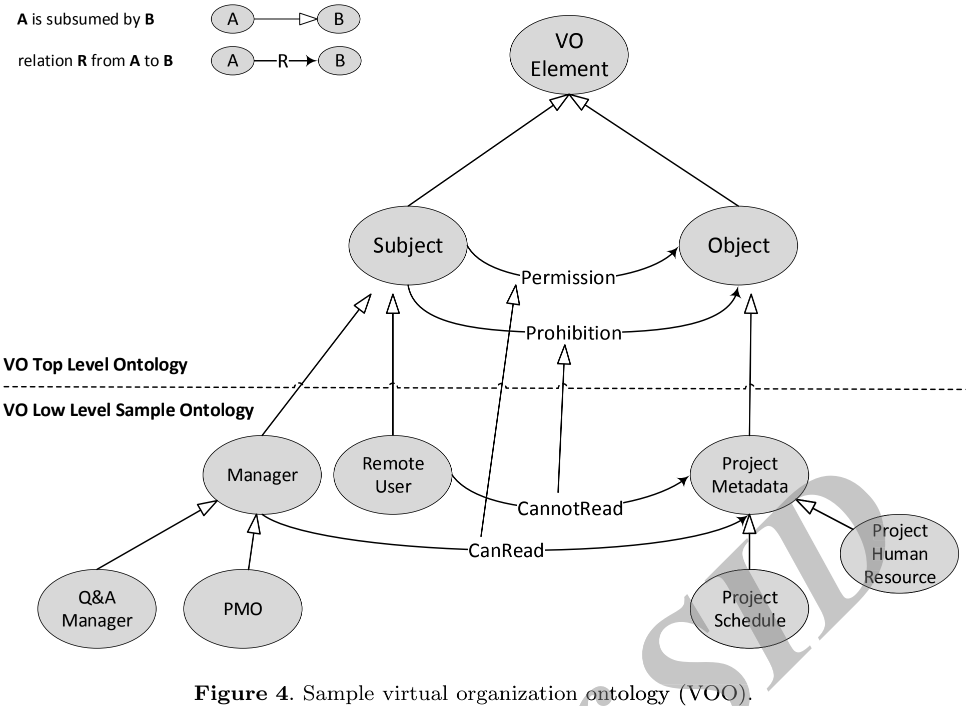


**Figure 4**. Sample virtual organization ontology (VOO).

## 4.2 Policy Inference

Inference of implicit security policy rules from the explicit ones are considered in both levels of access control in this model. This is the result of existing semantic relationships in the ontology leveraged in access control and also the combination of the enhanced SBAC and ABAC models.

### 4.2.1 Policy Inference in VO-Level Access Control

Different semantic relationships can be defined in the VO ontology (named VOO) between the concepts (or relations) defined in three domains of subjects, objects (resources), and (permissible or prohibited) actions. As we proposed in [21] for policy propagation in the SBAC model, permissions and prohibitions should be propagated from upper level concepts of subjects and objects, to lower level concepts of them. In this paper, as defined in Definition 4, we defined the access policy in form of the SWRL rules, and thus such an inference is considered inherently and we do not need to specify rules for policy propagation. In other words, if $S$ and $S'$ are subjects (i.e., subsumed by the *Subject* concept in VOO) where $S' \sqsubseteq S$ , $O$ is an object concept (i.e., subsumed by the *Object* concept) and there exists the following access rule (in SWRL) for a permission (or similarly a prohibition):

$$S(?s) \wedge O(?o) \rightarrow CanAction(?s, ?o)$$

[or similarly $S(?s) \wedge O(?o) \rightarrow CannotAction(?s, ?o)$]

The following access rule can be inferred, which results in propagation of a permission (or similarly a prohibition) from concept $S$ (the upper level concept) to $S'$ (the lower-level or subsumed concept):

$$S'(?s) \wedge O(?o) \longrightarrow CanAction(?s, ?o)$$

[or similarly $S'(?s) \wedge O(?o) \longrightarrow CannotAction(?s, ?o)$]

For objects, we have the similar situation. However, in SBAC, we have a different approach for actions, where positive authorizations (or permissions) are propagated from the upper level action concepts to the lower level ones, whereas for negative authorizations (or prohibitions) the propagation is done in the opposite direction. In this paper, we did not modeled actions explicitly and instead we modeled permissions and prohibitions as relations. In this approach, for example, if we define action Update as a type of action Read, we should define the corresponding permissions and prohibitions as follows.

$$CanUpdate \sqsubseteq CanRead$$
$$CannotRead \sqsubseteq CannotUpdate$$

As discussed in Note 1, since we cannot define prohibitions as complement of permissions (due to the possibility of defining conflicting access rules and monotonicity of the underlying logics), whenever we define a permission subsumed by another one (e.g., $CanUpdate \sqsubseteq CanRead$), we should add the opposite subsumption relationship between their corresponding prohibitions as well (e.g., $CannotRead \sqsubseteq CannotUpdate$).

Considering the above fact, if we have the following access rule:

$$S(?s) \wedge O(?o) \rightarrow CanActionX(?s, ?o)$$

By existing $CanActionX \sqsubseteq CanActionY$ in VOO, the following access rule can be inferred:

$$S(?s) \wedge O(?o) \rightarrow CanActionY(?s, ?o)$$

By the above assumptions, it is obvious that for prohibitions, the propagation direction is in the opposite way.

Hence in summary, for each permissible action subsumed by another, we should add the following axioms to VOO:

$$CanActionX \sqsubseteq CanActionY$$
$$CannotActionY \sqsubseteq CannotActionX$$

### 4.2.2 Policy Inference in Organization-Level Access Control

As described early in this paper, the ontology of subjects (which is a part of VOO) is in fact the ontology of subjects' attributes. In this ontology, if an attribute $X$ is subsumed by an attribute $Y$ (i.e., $X \sqsubseteq Y$), it means $X$ is a type of $Y$. Thus, when a user (subject) with some identity (e.g., $s_r$) holds (a credential of) attribute $X$, he holds attribute $Y$ as well. Hence, the permissions that are given to X-holders are given to Y-holders, too. For example in the sample ontology shown in Figure 4, each user who introduces himself as a PMO is a Manager as well. Hence, each permission given to the managers are given to the PMOs, too.

Also, we have the same inference rules for the permissions and prohibitions that have semantic relationships with each other according the defined ontology. For example, if X-holders have *CanUpdate* permission on special object, they have *CanRead* permission on the object, too.

## 4.3 Access Control Procedure

Each access request in form of $\langle s_r, SA, o_r, OC, Org, a_r \rangle$ is sent to the authorization server of the organization hosting the requested resource. In the request:

- $s_r$ is the identity of the subject or requester and $SA$ is a set of requester's credentials specifying his/her attributes.
- $o_r$ is the identity of the requested object or resource and $OC$ is the concept (in the ontology) that the requested resource is of its type and $Org$ is the identity of the organization which the resource belongs to.
- $a_r$ is the type of the operation or action that the subject requests to do on the resource.

Considering the access control framework presented in Figure 2, the following steps are taken for access control regarding the received access request.

(1) *Request Reception*: The PEP component of the authorization server in the organization of the resource provider (organization $Org$ in the request) receives the access request in the aforementioned format.

(2) *Validating the Credentials*: The attributes of the requester are provided by a set of credentials in $SA$. The validity of the attribute certificates (credentials) should be verified using an SOA– source of authority in privilege management infrastructure. The verified access request is sent to the PDP component of the VO manager through the PDP of the authorization server.

(3) *Updating Knowledge Base*: For each attribute $A$ whose credential is provided in $SA$, we should add $A(s_r)$ to the ABox of the VO knowledge base (to specify that $s_r$ is an individual of concept $A$ in the ontology). Also we should add $OC(o_r)$ for requested resource to the ABox.

(4) *VO-Level Decision Making*: Using the policy inference service– described in the previous section– the assertions added to the knowledge base (by the previous step) and the rules specified in Table 1, the VO manager makes its access control decision. The decision is "Deny" if $DenyAction(?s, ?o)$ is inferred. Otherwise, the decision is "Permit".

(5) *Organization-Level Decision Making*: In the $Org$ organization-level, the system should determine the list of the attributes that the user holds. It contains the attributes which their credentials are provided in the user's request (i.e., $SA$) and the attributes that the user holds (implicitly) due to the semantic relationships defined in the subjects' attributes in the ontology (as described in Section 4.1). Using the attributes of the user,the identity of the requested resource, and the type of the requested action:

   a) The agenda of applicable policy rules are extracted from the organization's access rules set (denoted by $OR$). Applicable policy rules are rules where their Subjects element is one of the attributes derived for the requester, the Resources element is equal to $o_r$, and the Actions element is of type $a_r$ in the request.

   b) If the retrieved rules in the agenda are consistent, the effect of the rule (that would be "Permit" or "Deny") is considered as the organization-level decision.

   c) If the retrieved rules in the agenda have conflict, the meta-policy, which is specified in $OMP.CR$, determines the organization-level decision.

d) If no rule exists in the agenda, the default policy, which is specified in $OMP.DP$ (in meta policy), determines the organization-level decision.

(6) *Decision Composition and Enforcement*: Following the decisions made in VO-level and organization-level, the PDP component of the authorization server makes the final decision based on the rules presented in Table 2 and policy composition strategy, which is determined and announced by the VO manager (i.e., $VOMP.CS$ in its meta-policy). If the request is permitted, the resource is provided by the organization to the requester. Otherwise, the rejection of the request is announced to the requester by the PEP component.

## 5 Case Study

In this section a case study is presented to clarify our proposed access control model. The proposed case study is a virtual digital library that formed by combination of four digital libraries. The Virtual Digital Library (VDL) and Digital Libraries (DL) relation is as follow:

$$VDL = \{DL_i|\ DL_i = \langle Identity_i, Policy_i\rangle\}$$

Each digital library has a unique identity and its own access control policy which is specified according to the ABAC model. Each DL shares a subset of its own resources and each DL has some members (subjects) that want to access the shared resources. In the case study the concepts of subjects and objects are considered as follow:

- $Subjects = \{Preteen, Teenager, Juvenile, Adult\}$
- $Objects = \{Wiki, Scientific\text{-}book, Story\text{-}book\}$

When a requester is authorized to access an object, then according to his allowed permissions can do some of these actions.

- $Actions = \{Read, Add, Edit, Delete\}$
- $Permissions = \{CanX|\ X \in Actions\}$
- $Prohibitions = \{CannotX|\ X \in Actions\}$

Each DL describe sits access control policies according to the ABAC model using XACML description language as depicted in Table 3.

The virtual digital library (VDL) specifies its access control policy according to the SBAC model. For which, SWRL is used to describe access control rules, and ontology is specified by OWL language. Figure 5 shows a simple ontology for our specified case study.

After description of the ontology, access control rules are specified by SWRL. Figure 6, shows an example of access control rules, which is described by SWRL.

### 5.1 Access Control Scenario

To clarify the proposed access control model, in this section the access control procedure of the model is elaborated by the consideration of the aforementioned case study.

- For example Tom is a Juvenile user who is a member of DL1 and wants to access the shared resources in DL2.
- By the rules of VDL, he receives the permissions $\{CanRead(Story\text{-}book), CanRead(Wiki), CanEdit(Wiki)\}$ directly or indirectly.
- By the rules of DL2, he gets the permission $\{CanRead(Scientific\text{-}book), CanRead(Story\text{-}book)\}$ .
- DL2 by consideration of its own granted permissions and the permissions which are granted to Tom by the VDL, decides that which permissions should be granted to Tom. To combine the access control policies of the $DL_i$ and the VDL many different types of algorithms such as union or intersection can be used. By the union algorithm all the permissions which are granted by the both of $DL_i$ and VDL will be granted to the requester, and by the intersection algorithm the only in common permissions of the $DL_i$ and VDL will be granted to the requester. For instance, if the union strategy is used by the VDL, Tom will get the $\{CanRead(Story\text{-}book), CanRead(Wiki), CanEdit(Wiki), CanRead(Scientific\text{-}book)\}$ permissions and if the intersection algorithm is used by the VDL, he will receive $CanRead(Story\text{-}book)$ permission.

## 6 Implementation and Evaluation

Access control decision in the proposed model is made in two steps. The first step of access control procedure is done by resource providers concerning the attributes of requests, and the second is done semantically by the VOM. The final decision about granting of permissions are made by providers according to the combination policy of the VO. The proposed model is evaluated in the rest of this section against the security requirements in VOs. Also a prototype of an access control system has been implemented based on the proposed model and framework to show the applicability of the proposed model and mechanism and evaluating the performance of the system in practice.

### 6.1 Evaluation of SABAC Model

In order to evaluate the proposed model, we compare it with the MAC, DAC, RBAC, E-RBAC, UCON, ABAC, and SBAC models. To this aim, the following criteria, according to the security requirements of virtual organizations are considered.

**Table 1**. Access decision making rules in VO-level access control system.

| |
|---|
| In case of inferring a conflict, the meta-policy (i.e., VOMP.CR) determines the VO-level decision. |
| $CanAction(?s, ?o) \wedge CannotAction(?s, ?o) \wedge VOMP.CR = PO \longrightarrow PermitAction(?s, ?o)$ |
| $CanAction(?s, ?o) \wedge CannotAction(?s, ?o) \wedge VOMP.CR = NO \longrightarrow DenyAction(?s, ?o)$ |
| The following rules, infer the final decision in case of existing no conflict. |
| $CanAction(?s, ?o) \wedge \mathbf{not} CannotAction(?s, ?o) \longrightarrow PermitAction(?s, ?o)$ |
| $CannotAction(?s, ?o) \wedge \mathbf{not} CanAction(?s, ?o) \longrightarrow DenyAction(?s, ?o)$ |
| The default policy (i.e., $VOMP.DP$) determines the VO-level decision in case of inferring neither permission nor prohibition for the requested action. |
| $\mathbf{not} CanAction(?s, ?o) \wedge \mathbf{not} CannotAction(?s, ?o) \wedge VOMP.DP = Permit \longrightarrow PermitAction(?s, ?o)$ |
| $\mathbf{not} CanAction(?s, ?o) \wedge \mathbf{not} CannotAction(?s, ?o) \wedge VOMP.DP = Deny \longrightarrow DenyAction(?s, ?o)$ |

**Table 2**. Decision composition based on different policy composition strategies.

| VO-level Decision | Org-level Decision | Decision Composition Strategy | | | |
|---|---|---|---|---|---|
| | | Union Str. | Intersection Str. | VO Override Str. | Org. Override Str. |
| Permit | Permit | Permit | Permit | Permit | Permit |
| Deny | Deny | Deny | Deny | Deny | Deny |
| Permit | Deny | Permit | Deny | Permit | Deny |
| Deny | Permit | Permit | Deny | Deny | Permit |

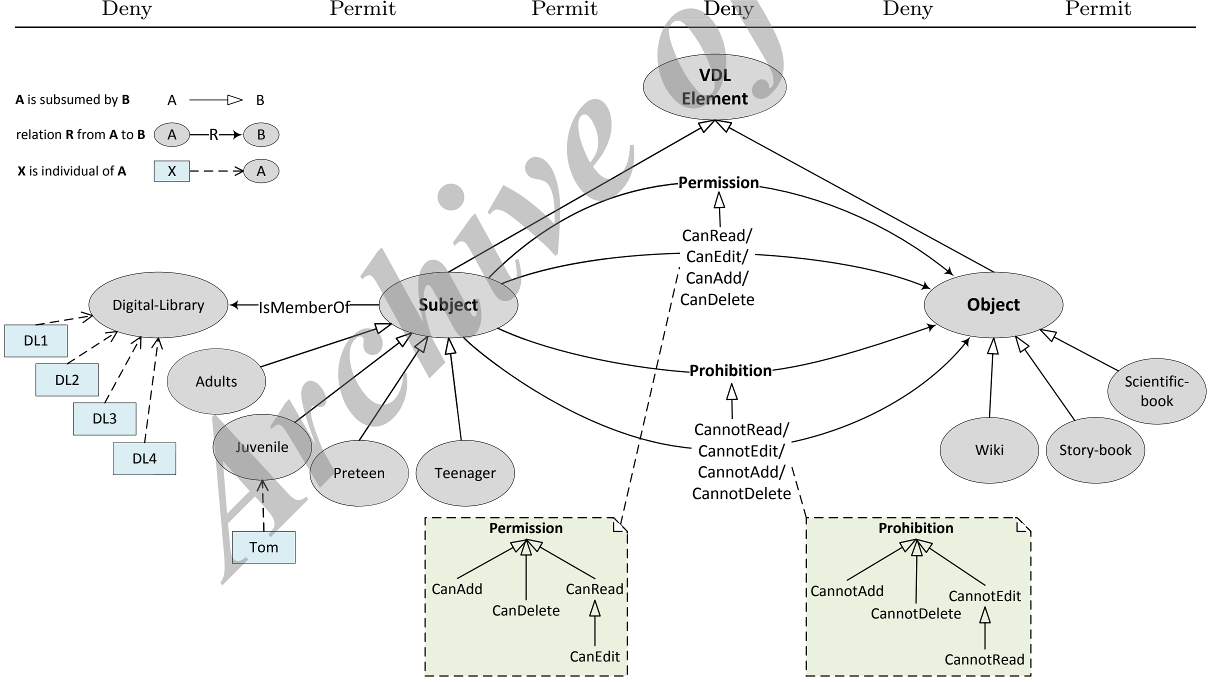


**Figure 5**. Simple ontology for virtual digital library (VOO).

- **Scalability:** since in VOs, by joining a new organization to a VO, the number of users and resources increases significantly, the access control model, which is used in these environment should be scalable enough from different aspects.
- **Management Complexity:** this criterion shows the complexity of management in access control models. In VOs with large numbers and variants of users and resources, the complexity of access policy management is one of the most important criteria in selecting or proposing an appropriate access control model for these environments. Performing access control in two levels and breaking management complexity is

**Table 3**. Access control policies of digital libraries.

| **Digital Library (1)** | **Access Control Policies** | **Digital Library (2)** | **Access Control Policies** |
|---|---|---|---|
| Meta Policy (MP) | **CR**=DO (Deny override)<br>**DP**=DN (Default is Deny) | Meta Policy (MP) | **CR**=DO (Deny override)<br>**DP**=DN (Default is Deny) |
| **Rule**:<br>1th_Access_Control_Rule | **Subjects**: Teenager;<br>**Object**: Scientific-book, Story-book;<br>**Action**: Read;<br>**Effect**: Permit; | **Rule**:<br>1th_Access_Control_Rule | **Subjects**: Adult;<br>**Object**: Wiki;<br>**Action**: Read, Edit;<br>**Effect**: Permit; |
| **Rule:**<br>2nd_Access_Control_Rule | **Subjects:** Juvenile, Adult;<br>**Object:** Wiki;<br>**Action:** Read, Add, Edit;<br>**Effect:** Permit; | **Rule:**<br>2nd_Access_Control_Rule | **Subjects:** Adult;<br>**Object:** Story-book, Scientific-book;<br>**Action:** Read;<br>**Effect:** Permit; |
| **Rule:**<br>3rd_Access_Control_Rule | **Subjects:** Teenager;<br>**Object:** Wiki, Story-book, Scientific-book;<br>**Action:** Edit;<br>**Effect:** Deny; | | |
| **Digital Library (3)** | **Access Control Policies** | **Digital Library (4)** | **Access Control Policies** |
| Meta Policy (MP) | **CR**=DO (Deny override)<br>**DP**=DN (Default is Deny) | Meta Policy (MP) | **CR**=PO (Permit override)<br>**DP**=PR (Default is Permit) |
| **Rule:**<br>1th_Access_Control_Rule | **Subjects:** Teenager, Juvenile, Adult;<br>**Object:** Scientific-book, Story-book, Wiki;<br>**Action:** Read;<br>**Effect:** Permit; | **Rule:**<br>1$th$_Access_Control_Rule | **Subjects:** Preteen, Teenager;<br>**Object:** Wiki;<br>**Action:** Edit, Delete, Add;<br>**Effect:** Deny; |
| **Rule:**<br>2nd_Access_Control_Rule | **Subjects:** Preteen;<br>**Object:** Story-book;<br>**Action:** Read;<br>**Effect:** Permit; | **Rule:**<br>2nd_Access_Control_Rule | **Subjects:** Adult;<br>**Object:** Scientific-book;<br>**Action:** Read;<br>**Effect:** Deny; |
| **Rule:**<br>3rd_Access_Control_Rule | **Subjects:** Preteen;<br>**Object:** Wiki;<br>**Action:** Delete, Add, Edit;<br>**Effect:** Deny; | **Rule:**<br>3rd_Access_Control_Rule | **Subjects:** Juvenile;<br>**Object:** Story-book, Wiki, Scientific-book;<br>**Action:** Read;<br>**Effect:** Permit; |

the main reason that makes the SABAC more suitable for VOs than other fine-grained models such as ABAC, RBAC, and E-RBAC which are enforcing access control policy in one level.

- **Semantic:** leveraging semantic technology for inter-organization interoperability and having more abstract access control policies (based on a shared ontology of subjects, resources, and actions) is an important requirement for VOs where different organizations (under different management and control) are cooperating with each other.
- **Policy Abstraction Level:** the abstraction level of policies is considered in this criterion.

Virtual Digital Library: VO-level access control rules by SWRL (VOP).

1) $Subject(?s) \wedge Story\_book(?sb) \longrightarrow CanRead(?s, ?sb)$

2) $Subject(?s) \wedge Wiki(?w) \longrightarrow CanRead(?s, ?w)$

3) $Juvenile(?j) \wedge Wiki(?w) \wedge IsMemberOf(?j, DL_1) \longrightarrow CanEdit(?j, ?w)$

4) $Adult(?a) \wedge Scientific\_book(?sb) \wedge IsMemberOf(?a, DL_1) \longrightarrow CanRead(?a, ?sb)$

5) $Subject(?s) \wedge Object(?o) \wedge IsMemberOf(?s, DL_4) \longrightarrow CanEdit(?s, ?o)$

6) $Subject(?s) \wedge Wiki(?w) \wedge IsMemberOf(?s, DL_3) \longrightarrow CanDelete(?s, ?w)$

7) $Subject(?s) \wedge Object(?o) \wedge IsMemberOf(?s, DL_2) \longrightarrow CanEdit(?s, ?o)$

**Figure 6**. Access control policy rules by SWRL.

In VOs, due to the hierarchical structure of management, we need to specify policies in both conceptual (abstract) and individual (concrete or ground) levels simultaneously.

- **User Identification:** users might be considered by their identities or attributes in an access control model. In large scale environments, we cannot specify access rules based on users' identities in practice and thus many traditional models become improper for large-scale environments such as VOs.
- **Policy Inference and Propagation:** due to the complexity of access policy rules in VOs with distributed resources and users, and also existing semantic relationships between the different elements engaging in access control, the access control model for VOs should be capable of inferring implicit access rules from the explicit ones. Policy inference and (permission/prohibition) propagation could be performed in different domains; i.e., subjects, objects (resources), and actions.
- **Suitable Level of using Model:** as discussed earlier in this paper, the VO manager requires specifying the policies of the VO in more abstract level; however, each participating real organization has its own access policies which should be specified in more concrete level. Whether a model is suitable to be leveraged for access policy specification and enforcement in a level or not is considered in this criterion.

**??** demonstrates the evaluation of the proposed model, i.e., the SABAC model, in comparison with the more famous access control models (which are employed for VOs or similar environments such as Cloud and Grid) based on the above introduced criteria. Note that most of the mechanisms or frameworks proposed for access control in VOs, such as Condor, Legion, CAS, VOMS, PERMIS, Shibboleth, and Akenti are based on the DAC or RBAC models (for more details refer to [2]). In **??** we compared the models and not the mechanisms. Note that in this table E-RBAC is a set of extended RBAC models which are proposed for access control in pervasive computing environments and surveyed in [13]. We categorized the extended RBAC models to the models considering contextual constraints (in permission assignment or role assignment) and the models considering trust to users in role assignment.

### 6.2 Implementation of Sample Access Control System

To show the applicability of the proposed model and framework for access control in VOs, a prototype of an access control system has been implemented.

**Implementation of Enhanced ABAC:** XACML is a language which supports ABAC model requirements. ABAC is used to implement access control mechanism for organizations. A simple text editor or some special tools such as “UMU XACML” editor can be used to write access control policies. The authorization system, which is enforcing policies specified in XACML, has been implemented in Java using sun's XACML API.

**Implementation of Enhanced SBAC:** semantic based access control involves the description of ontology and rules. For specification of the ontology we leveraged Protégé, which is an open-source software. The OWL consists of classes (concepts), individuals, and properties. In addition, the hierarchy (the taxonomy) of concepts can be represented by OWL. After description of ontology, access control rules could be specified with SWRL.

The required authorization system for VOs (based on the SBAC model) has been implemented by some java APIs such as “Pellet” and “Jena”. Pellet is a

**Table 4**. Detection rates for comparison between the EVS and the best of some other algorithms

| Criteria<br>Models | 1- Scalability | 2- Management Complexity | 3- Semantic | 4- Abstraction Level | 5- Subject Identification | 6- Propagation Domains | 7- Suitable Level |
|---|---|---|---|---|---|---|---|
| **DAC-MAC** | L | C | N | I | IB | - | - |
| **RBAC** | M | M | N | S | IB | S | O |
| **ABAC** [18] | H | M | N | C | AB | S | O |
| **E-RBAC** [13] (Context) | M | C | N | C | AB | - | O |
| **E-RBAC** [13] (Trust) | M | C | N | S | IB | - | O |
| **UCON** [15] | H | C | N | C | AB | - | O |
| **OBAC** [25] | M | C | Y | C | AB | S | O |
| **S_ABAC** [27] | VH | M | Y | C | AB | S,O | O |
| **SBAC** [21] | VH | E | Y | C,I | AB | S,O,A | V |
| **SABAC** (proposed model) | VH | M | Y | C,I | AB | S,O,A | V,O |

1- Scalability [VH: very highly scalable; H: highly scalable; M: moderately scalable; L: low scalable]
2- Management complexity [C: complex; M: moderate; E: easy]
3- Semantic relationship consideration [Y: yes; N: no]
4- Policy Abstraction level [C: conceptual level; S: semi-conceptual level; I: individual level]
5- Subject (user) identification in policy rule specification [IB: identity-based; AB: attribute-based]
6- Policy inference and propagation domains [S: subjects domain; O: objects domain; A: actions domain]
7- Suitable Level of using Model [O: organization level; V: virtual organization level]

reference engine that enables us to make inferences (of implicit policies from the explicit ones), check consistency of the ontology, and so on.

### 6.3 Experimental Results

By implementing a prototype of an access control system based on the proposed model, we can evaluate the performance (response time) of the access control procedure. To this aim, the previous case study has been considered and the response time of the access requests sent to the first organization (i.e., DL1) have been measured. The tests were run on a machine with 4GB of RAM and corei5 2.4GHZ processor.

Figure 7 shows the average response (decision making) time of different access requests submitted from concurrent users to the access control system. The access requests are generated randomly by different users on the existing resources and might be granted or denied based on the policies defined in the sample system. The response times are increased linearly by increasing the number of concurrent users sending their access requests to the system. The minimum response times and the maximum ones are represented in this figure, too.

## 7 Conclusion

Virtual Organization consists of some real organizations with common interests. Organizations federate their resources to achieve their goals. In this collaborative situation, provision of security and especially access control is necessary. In most of the existing access control systems for VOs, access decision is made in both levels of resource providers and the VO manager. However, in both level they are following the same model of access control (mostly DAC models). Such models are not appropriate for large scale environments like Vos, where users of each organization are unknown to each other and resource management is performed in different abstraction layers (i.e., VO-level is more abstract than real organizations level). In this paper, we introduced a combinatorial model for access control in VOs, where organizations make their access control decisions based on the enhanced ABAC model and VOs make their decisions based on the enhanced SBAC model. By ABAC, organizations can make fine-grain decisions and by SBAC, VOs can make their decisions in more abstract level by considering the semantic relationships of subjects and resources.

To show the applicability of the proposed model and measuring the response time of its access control procedure, an access control system based on the

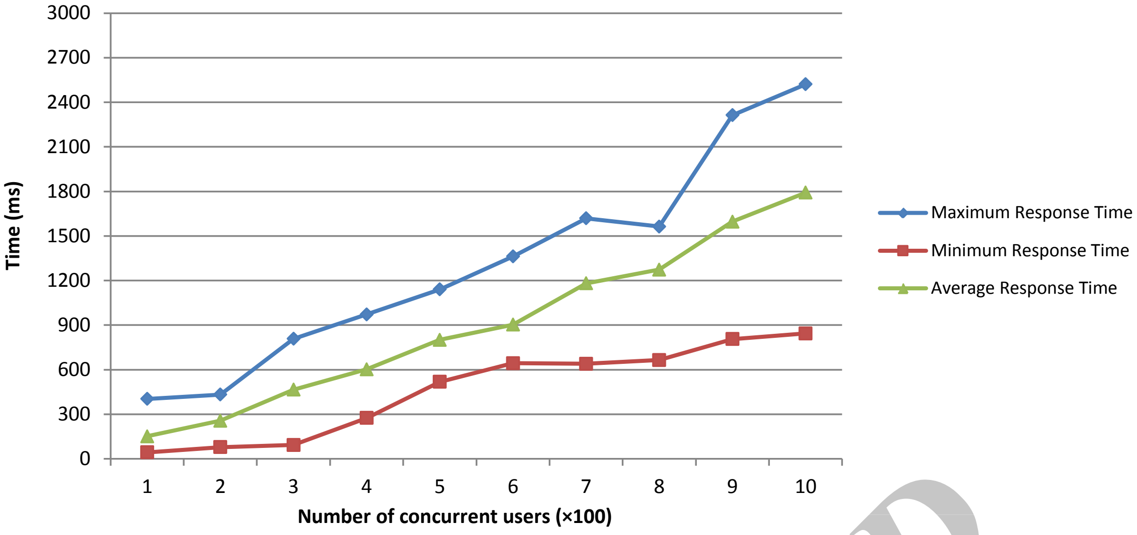


**Figure 7**. Response times of concurrent users' access requests.

proposed model was designed and implemented in Java using sun's XACML, Jena, and Pellet APIs. The implemented system enforces access control policies specified in both sides of resources providers and the VO manager. The final decisions are made in this system through the providers and according to the combination of their own policies and the VO's policies. The evaluation of our proposed access control model, in comparison with the famous access control models leveraged in these environments, shows that by using such an access control system, we obtain more scalable fine grained authorization system with lower complexity in management, and considering semantic relationships specified in the abstract level (i.e., the ontology defined in the VO).

## References

[1] K. Jacobsen. A Study of Virtual Organizations, 2004. Project Report, Norwegian University of Science and Technology, Department of Computer and Information Science, NTNU.

[2] M. Arasteh, M. Amini, and R. Jalili. A Trust and Reputation-based Access Control Model for Virtual Organizations. In *Proceedings of the 9th International IEEE Conference on Information Security and Privacy (ISCISC'12)*, pages 121–127, Tabriz, Iran, 2012.

[3] B. Nasser, R. Laborde, A. Benzekri, F. Barrére, and M. Kamel. Dynamic Creation of Inter-Organizational Grid Virtual Organizations. In *Proceedings of the First IEEE International Conference on e-Science and Grid Computing*, pages 405–412, Melbourne, Australia, 2005.

[4] T. Ryutov, C. Neuman, L. Zhou, and N. Foukia. Establishing Agreements in Dynamic Virtual Organizations. In *Proceedings of the First IEEE International Conference on Security and Privacy for Emerging Areas in Communication Networks*, pages 90–99, Athens, Greece, 2005.

[5] S. Crompton, M. Wilson, A. Arenas, L. Schubert, D. Cojocarasu, J. Hu, and P. Robinson. The TrustCoM General Virtual Organization Agreement Component. In *UK e-Science All Hands Meeting, Nottingham*, volume 135, Nottingham, UK, 2007.

[6] L. Huraj and H. H VO. Reiser. Intersection Trust In Ad hoc Grid Environment. In *Proceedings of the 5th IEEE International Conference on Networking and Services (ICNS'09)*, pages 456–461, Valencia, Spain, 2009.

[7] M. Ibrohimovna and S. Groot. A Framework for Access Control and Management in Dynamic Cooperative and Federated Environments. In *Proceedings of the Fifth Advanced IEEE International Conference on Telecommunications (AICT'09)*, pages 459–466, Venice, Italy, 2009.

[8] B. Katzy, C. Zhang, and H. Löh. Reference Models for Virtual Organizations Virtual Organizations. *Virtual Organizations*, pages 45–58, 2005.

[9] Benzakeri. Virtual Organization Security Policy: Specification & Deployment (V1). Technical report, IRIT, 2006.

[10] VT. Berners-Lee. A Roadmap to the Semantic Web. In *World Wide Web Consortium*, September 1998.

[11] MI. Yague, A. Mana, J. López, and JM. Troya. Applying the Semantic Web Layers to Access Control. In *Proceedings of 14th IEEE International Workshop on Database and Expert Systems Applications*, pages 622–626, 2003.

[12] S. Verma, S. Kumar, and M. Singh. Comparative Analysis of Role Base and Attribute Base Access Control Model in Semantic Web. *International*

*Journal of Computer Applications*, 46(18):1–6, 2012.

[13] A. Bakar and J. Jais. A Review on Extended Role Based Access Control (E-RBAC) Model in Pervasive Computing Environment. In *Proceedings of the First IEEE International Conference on Networked Digital Technologies in Ostrava*, pages 533–535, Ostrava, Czech Republic, 2009.

[14] N.Dagdee and R. Vijaywargiya. Credential Based Mediator Architecture for Access Control and Data Integration in Multiple Data Source Environment. *International Journal of Network Security & Its Applications (IJNSA)*, 3(3):42–56, May 2011.

[15] J. Park and R. Sandhu. The $\text{UCON}_{\text{ABC}}$ Usage Control Model. *ACM Transactions on Information and System Security*, 7(1):128?174, February 2004.

[16] L. Cirio, I. Cruz, and R. Tamassia. A Role and Attribute based Access Control System Using Semantic Web Technologies. In *Proceedings of the OTM Confederated International Conference on the Move to Meaningful Internet Systems*, pages 1256–1266. Springer, 2007.

[17] VC. Hu, D. Ferraiolo, R. Kuhn, AR. Friedman, AJ. Lang, MM. Cogdell, A. Schnitzer, K. Sandlin, R. Miller, and K. Scarfone. Guide to Attribute Based Access Control (ABAC) Definition and Considerations. Technical report, 2013. NIST Special Publication 800.

[18] D. Xu and Y. Zhang. Specification and Analysis of Attribute-Based Access Control Policies: An Overview. In *Proceedings of the 8th IEEE International Conference on Software Security and Reliability-Companion*, pages 41–49, San Francisco, USA, 2014.

[19] B. Lang, I. Foster, F. Siebenlist, R. Ananthakrishnan, and T. Freeman. A Flexible Attribute based Access Control Method for Grid Computing. *Journal of Grid Computing*, 7(2):169–180, 2009.

[20] X. Chen, Y. OUYang, M. Zhu, and Y. He. Semantic-Aware Access Control for Grid Application. In *Proceedings of the 9th IEEE International Conference for Young Computer Scientists*, pages 971–975, Hunan, China, 2008.

[21] S. Javanmardi, M. Amini, R. Jalili, and Y. GanjiSaffar. A Semantic based Access Control Model. In *Proceedings of the 11th Nordic Workshop on Secure IT-Systems (NordSec'06)*, pages 157–168, Linkping, Sweden, 2006.

[22] A.N. Ravari, M. Amini, R. Jalili, and J.H. Jafarian. History based Semantic Aware Access Control Model Using Logical Time. In *Proceedings of the 11th IEEE International Conference on Computer and Information Technology*, pages 43–50, Khulna, Bangladesh, 2008.

[23] S. Durbeck, C. Fritsch, G. Pernul, and R. Schillinger. A Semantic Security Architecture for Web Services the Access-eGov Solution. In *Proceedings of the 10th IEEE International Conference on Availability, Reliability, and Security*, pages 222–227, Krakow, Poland, 2010.

[24] T. Priebe, W. Dobmeier, and N. Kamprath. Supporting Attribute-based Access Control with Ontologies. In *Proceedings of the First IEEE International Conference on Availability, Reliability and Security*, 2006.

[25] A. Mohammad, G. Kanaan, T. Khdour, and S. Bani-Ahmad. Ontology-Based Access Control Model for Semantic Web Service. *Journal of Information and Computing Science*, 6(3):177–194, 2006.

[26] H. Shen. A Semantic-Aware Attribute-based Access Control Model for Web Services. *Algorithms and Architectures for Parallel Processing, LNCS 5574*, pages 693–703, 2009.

[27] S. Hai-Bo. A Semantic-and Attribute-based Framework for Web Services Access Control. In *Proceedings of the 2nd IEEE International Workshop on Intelligent Systems and Applications (ISA)*, pages 1–4, Wuhan, China, 2010.

[28] J. Luo, X. Wang, and A. Song. A Semantic Access Control Model for Grid Services. In *Proceedings of the 9th IEEE International Conference on Computer Supported Cooperative Work in Design*, pages 350–355, 2005.

[29] J. Luo, X. Ni, and J. Yong. A Trust Degree based Access Control in Grid Environments. *Information Sciences*, 179(15):2618–2628, 2009.

[30] I. Foster, H. Kishimoto, A. Savva, D. Berry, A. Djaoui, A. Grimshaw, B. Horn, F. Maciel, F. Siebenlist, and R. Subramaniam. The Open Grid Services Architecture. *Global Grid Forum*, GFD-I, 2006.

[31] AL. Pereira, V. Muppavarapu, and SM. Chung. Managing Role-based Access Control Policies for Grid Databases in OGSA-DAI Using CAS. *Journal of Grid Computing*, 5(1):65–81, 2007.

[32] L. Pearlman, V. Welch, I. Foster, C. Kesselman, and S. Tuecke. The Community Authorization Service: Status and Future. *arXiv preprint cs/0306082*, 2003.

[33] W. Meng, H. Xia, and H. Song. A Dynamic Trust Model Based on Recommendation Credibility in Grid Domain. In *Proceedings of the IEEE International Conference on Computational Intelligence and Software Engineering*, pages 1–4, 2009.

[34] W. Zhou and C. Meinel. Implement Role Based Access Control with Attribute Certificates. In *Proceedings of the 6th IEEE International Conference of ICACT on Advanced Communication*

*Technology*, pages 536–540, Phoenix Park, Korea, 2004.

[35] M. L. C. Hui NE and C. H. Yong. A Context-Aware based Authorization System for Pervasive Grid Computing. *World Academy of Science, Engineering and Technology*, 74, 2011.

[36] B. Stepien, S. Matwin, and A. Felty. Advantages of a non-Technical XACML Notation in Role-based Models. In *Proceedings of the 9th IEEE International Conference on Privacy, Security and Trust (PST)*, pages 193–200, Montreal, Canada, 2011.

[37] V. Muppavarapu and SM. Chung. Semantic-based Access Control for Grid Data Resources in Open Grid Services Architecture-Data Access and Integration (OGSA-DAI). In *Proceedings of the 20th IEEE International Conference on Tools with Artificial Intelligence*, pages 315–322, Dayton, USA, 2008.

[38] A. Anderson. Core and Hierarchical Role Based Access Control (RBAC) Profile of XACML v2.0. Online, May 2015. Available at http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-rbac-profile1-spec-os.pdf.

[39] S. A. Javadi and M. Amini. A Semantic-Aware Role-Based Access Control Model for Pervasive Computing Environments. *ISeCure- The ISC International Journal of Information Security*, 5(2):119–140, July 2013.

**Morteza Amini** received his Ph.D. in Software Engineering (Data Security field) from Sharif University of Technology. He is currently an assistant professor at Department of Computer Engineering, Sharif University of Technology, Tehran, Iran. His research interests include database security, access control, intrusion detection, and formal methods in information security.

**Majid Arasteh** received his B.S. degree in Information Technology Engineering from Islamic Azad University of Zanjan, Iran, in July 2009, and M.S. degree in Information Technology Engineering (Information Security) from Malek Ashtar University of Technology, Tehran, Iran, in February 2012. During his M.S, he joined as a member of Data and Network Security Lab (DNSL) at Department of Computer Engineering, Sharif University of Technology, Tehran, Iran. His research interests include network security, access control and privacy.