Access Control for Web Data: Models and Policy Languages

Barbara Carminati, Elena Ferrari
University of Insubria at Como, Italy
Email: {barbara.carminati,elena.ferrari}@uninsubria.it

Bhavani Thuraisingham
University of Texas at Dallas, USA
Email: bhavani.thuraisingham@utdallas.edu

Abstract

The web has made easily available an enormous amount of information in digital form and has facilitated the exchange of such information. In this environment, access control is a primary concern. The key issue is how to trade-off between maximizing the sharing of information and enforcing a controlled access to web data. In this paper we start by outlining which are the main access control requirements of web data. Then, we review researches carried on in the field, by mainly focusing on XML. Finally, we discuss policy languages for the semantic web, and outline which are the main research directions in this field.

I Introduction

The Web has made available a huge amount of information to an enormous user population and has greatly simplified the exchange of information in digital form. Although the benefits of the Web can be easily understood by everyone, it has also raised serious security concerns. Traditionally, securing data requires to deal with three main issues: authenticity, confidentiality, and integrity. Satisfying data authenticity means that the subject receiving a data is guaranteed that they actually come from the source they claim to be from. Ensuring data confidentiality means that data contents can only be disclosed to authorized users. By contrast, the term integrity implies two different security properties. The first refers to data protection from unauthorized modification operations, whereas the second means that data contents are not altered during their transmission over the net. In the last decades, several security mechanisms have been proposed to protect data stored in conventional DBMSs and OSs [31]. In general, these solutions enforce integrity through access control mechanisms and encryption-based techniques, confidentiality through access control mechanisms, whereas data authenticity requires the use of digital signature techniques. Therefore, a key component of the security infrastructure of any system is the access control mechanism, which mediates each access request by authorizing only those in accordance with the specified access control policies. Clearly, such paradigm should be revised for fitting in an open environment like the web, characterized by an highly dynamic population. Additionally, data on the web are unstructured or semistructured, being XML [37] the de-facto standard for their representation. As such, they have new security requirements with respect to traditional structured data, that must be taken into account in developing an access control solution suitable for the web environment.

Furthermore, the web has recently evolved into the semantic web. The semantic web [5] is a web that is intelligent with machine-readable web pages. The goal is to automate various activities

on the web without human intervention. The idea behind the semantic web is to make the web as intelligent as possible. Therefore, in addition to store and manage data and information, the web should enable people to carry out their daily activities. Each individual could have his or her own personal space on the web and carry out various activities from learning to teaching to providing services as well as obtaining services from the web. Clearly, as the web evolves into the semantic web, there are more and more possibilities for security breaches as we introduce new technologies. Therefore, it is critical that security be considered right from the beginning, in the semantic web development. To secure the semantic web, we need to secure all its components such as XML, RDF (Resource Description Framework, which is based on XML syntax and incorporates capability to specify semantics), Agents, Databases, Web Services, and Ontologies and ensure the secure interoperation of all these technologies. Security research for the semantic web is just at the beginning.

In this paper, we focus on access control for web data. We start by discussing which are the main access control requirements of web data, and outlining the main differences with respect to conventional structured data. Then, we review some of the most important proposals of access control models and policy languages for web data, by focusing on XML. We decide to cast our discussion on XML because it is today the standard for data representation and exchange over the web. We also provide a comparative analysis of the considered access control models with respect to the devised requirements.

Finally, the last part of the paper is devoted to the semantic web. First, we outline which are the main requirements of a policy language for the semantic web. Then, we review the research carried on in the fields of XML, RDF, and Web Services, and discuss what is still missing and which are the main research directions in the field. With respect to other survey papers (e.g., [15, 16]) on access control, the current paper has a particular focus on web data and the semantic web.

More precisely, the remainder of this paper is organized as follows. Next section introduces background on access control. Section III discusses access control requirements for web data, whereas Section IV surveys some of the most relevant proposals of access control models and policy languages for web data. Section V is devoted to semantic web, whereas Section VI concludes the paper.

II Background on access control

Access control deals with preventing unauthorized operations (e.g., read, write) on the managed data. Access control is usually performed according to a set of *access control policies*, stated by Security Administrators (SAs) or users. To make easier the task of policy evaluation, policies can be enforced through a set of *authorizations*, stating for each subject the accesses it can perform on the protected resources. How to represent and store authorizations/policies usually depends on the protected resources. For instance, in a relational DBMS authorizations/policies are modelled as tuples stored into system catalogs. By contrast, when resources to be protected are XML data, authorizations/policies are usually encoded using XML itself. However, whichever format is used to express authorizations/policies, we can distinguish three main components: a *subject* – the entity to which authorizations are granted (e.g., users, processes), an *object* – the resource to which access is granted (e.g., files, relational tables), and an *access mode*, specifying the action (e.g., read, write) that subjects can exercise on objects. Policies and authorizations are specified according to an *access control model*, which specifies the characteristics and relations among the authorization/policy basic components (e.g., subjects, objects, and access modes), the format of authorizations and/or policies, and states how access control should take place. For instance, some access control models support a hierarchical organization of subjects, objects, and privileges, other

consider only flat domains. Moreover, a policy language can be defined on support of an access control model, specifying the syntax of authorizations/policies. Finally, the last key component of the access control infrastructure is the *access control mechanism* (or reference monitor), which is a software module in charge of enforcing access control. It intercepts each access request submitted to the system (for instance, SQL statements in case of relational DBMSs) and, on the basis of the specified authorizations/policies, it determines whether the access should be partially or totally authorized, or it should be denied.

A basic distinction when dealing with access control is between *discretionary* and *mandatory* access control. Discretionary access control (DAC) governs the access of subjects to objects on the basis of subjects' identity and a set of authorizations. Authorizations state, for each subject, the privileges it can exercise on each object in the system. When an access request is submitted to the system, the access control mechanism verifies whether the access can be authorized according to the specified policies. If this is not the case, access is denied.

Discretionary access control has been widely investigated in the past decades [18]. Its success is mainly due to its flexibility, and adaptability to many application environments and security needs, in that, by properly configuring the set of authorizations, it is possible to specify a wide range of access control policies. A milestone in DAC is the research work carried on in the late 70s at IBM Almaden Research Center, in the framework of the System R project [21]. Such work is still the basis of discretionary access control mechanisms provided by commercial DBMSs. Since the developments of DAC in the 1970s, numerous other developments have been made to make DAC more suitable to the access control requirements of advanced applications, such as data warehouses, data mining systems, multimedia systems, sensor systems, workflow management systems, and collaborative systems. This has resulted in several extensions to the basic DAC models, by including the support for temporal constraints, derivation rules, positive and negative authorizations, strong and weak, and content-dependent authorizations [18].

By contrast, mandatory access control (MAC) specifies the accesses that subjects have to objects based on subjects and objects security classification [7]. Most mandatory access control models have been designed based on the Bell and LaPadula model [4], first specified for operating systems. In the Bell and LaPadula model, subjects are assigned clearance levels, whereas objects are assigned sensitivity levels. The clearance levels as well as the sensitivity levels are called security levels and form a partially ordered lattice. Access control is regulated by two axioms, referred to as *simple security property* and **-property*, which prevent subjects to read objects with an higher or not comparable security level and to write into objects with a lower or not comparable security level. While some work on mandatory security started in the late 1970s, it was not until the Air Force summer study in 1982 that many of the efforts in this area were initiated [3]. This resulted in the development of various secure database system prototypes and products (see [18] for an overview).

In addition to DAC and MAC, *role-based access control* (RBAC) has been more recently proposed [17]. RBAC is an alternative to discretionary and mandatory access control, mainly conceived for regulating accesses within companies and organizations. In RBAC, permissions are associated with roles, instead of with users, and users acquire permissions through their membership of roles. Roles are created to organize the various job functions in an organization and users are associated with roles according to their functions and/or to the tasks they must accomplish. The set of authorizations can be inferred by the sets of user-role and role-permission assignments. These components collectively state whether a specific user is allowed or denied to access a specific resource. Therefore, the authorization state is able to evolve with respect to the life of the system, by simply changing one of both of the two assignments. Authorization administration is greatly simplified. If a user moves to a new function within the organization, there is no need to revoke the authorizations he/she had in the previous function and grant the authorizations he/she needs in the new

one. The SA simply needs to revoke and grant the appropriate role memberships. Additionally, RBAC has been proven to be policy neutral [28] in that it is able to model both DAC and MAC.

Clearly, the advent of the World Wide Web in the 1990 has exacerbated the need for access control and has posed new research challenges, such as for instance issues related to access control for web services, semantic web, or issues related to the outsourcing of data over the web. All these issues call for new solutions, models, and mechanisms. In the following sections, we focus on one of the main issues related to access control for web data, that is, access control models and related policy languages. We start by discussing the access control requirements of web data.

III Access control requirements for web data

In what follows, we focus on access control requirements for web data. New requirements posed by web data are mainly due to two aspects: the first is the different nature of web data with respect to traditional ones (i.e., semi-structured or unstructured vs. structured data, graph-based data vs. flat data), the second is due to the open infrastructure supporting data dissemination on the web. These basic differences result in new access control requirements that we need to consider in specifying an access control model for web data. By recalling that the three basic building blocks of an access control model are subjects, objects, and access modes, in what follows we start by discussing the requirements along these three dimensions.

Subjects. Today the web is accessed by a large amount of people looking for any kind of data. This user population is characterized by different profiles and needs, which are not known a priori and can change very frequently over time. In this context, conventional identity-based access control scheme are no more appropriate, in that they can result in a huge number of authorizations/policies to be managed. A common adopted approach to cope with this issue is that of grouping subjects according to some criteria, and then specifying access control policies for subject groups instead of for single users. A relevant proposal in this context is represented by RBAC, that we have illustrated in the previous section. A further proposal, which is now widely accepted in the web environment is that of using *credentials* [36]. A credential consists of a set of characteristics of a user, which are considered relevant for access control purposes (e.g., age, nationality, membership to some associations). By using subject credentials the specification of access control policies is more direct and intuitive, since policies can be defined in general terms, by specifying conditions on credentials and credential properties. Therefore, a policy does not apply to a particular subject, but to a group of subjects having the same characteristics. For instance, it is possible to specify an access control policy that applies to all users which are affiliated to ACM by more than 10 years.

Objects. A further aspect that needs to be considered is the specification of *protection objects*, that is, the piece of information to which an authorization/policy applies. A first issue is related to the granularity of access control. Generally, granularity levels depend on the structure of the data to be protected. For instance, let us consider XML documents. Such documents have a nested or hierarchical structure, being defined in terms of components (i.e., elements) that can be themselves organized into subcomponents (i.e., sub-elements, attributes). Therefore, protection objects can be selected elements/attributes, a whole document, or an element with all (or some) of its subelements. Moreover, XML documents could be inter-linked through IDREFs/URI attributes, and the same happens for HTML pages. To support a differentiated and selective protection of web data, the access control model must be therefore flexible enough to support a wide spectrum of protection granularity levels, identified on the basis of the structure of data. Furthermore, data may have an associated intensional description of their structure (e.g., DTD or XMLSchema for XML documents). Thus, an access control model must be able to exploit this intensional description in

the specification of protection objects. For example, it must be possible to specify access control policies at the DTD/XML schema level, which apply to all valid documents conforming to that DTD/XML schema.

Besides granularity levels, a further requirement is the support for content-dependent access control. Content-dependent access control makes one able to state authorizations/policies that apply to a protection object on the basis of its content. This is an essential requirement on the web, since very often documents with the same type and structure have contents of different sensitivity degrees, and therefore need a differentiated protection. It is therefore necessary that the access control model allows one to include conditions against document content in authorizations/policies.

Access modes. In developing an access control model for web data, a further requirement is related to the definition of access modes, that is, privileges that subjects can exercise on protection objects. This step mainly depends on the nature of protection objects. For web data for instance, due to their hypertext-structures possible suitable privileges could be of two classes: *browsing privileges*, allowing subjects to access information in a document or to navigate through its links, and *authoring privileges*, enabling subjects to modify document contents and/or document structure. An access control model for web data must therefore support both these kinds of privileges.

Additional requirements. Besides the requirements related to subjects, objects and access modes specification, we have identified further requirements related to access control for web data, that we discuss in what follows.

Authorization propagation, exception management, and conflict resolution. To cope with the impressive number of authorizations that could be necessary to specify for web data, it is necessary to adopt a mechanism able to reduce as much as possible such number. A possible solution is to exploit a notion of propagation, which states whether and how an authorization/policy specified on a given protection object o and for a given subject s propagates to protection objects/subjects that are related to o/s by some sort of semantic relationships. For instance, if we consider XML documents propagation can be enforced on protection objects by exploiting their hierarchical structure.

Applying a propagation mechanism requires to address a further relevant issue, that is, the possibility of specifying exceptions to the by-default propagation. Consider, for instance, an access control policy that should apply to a whole XML document except for one of its node (i.e., an element/attribute). Without an exception mechanism, this requires to specify a high number of authorizations, that is, one for each document node to which the corresponding policy applies. By contrast, if we have a way to specify exceptions we need only to define a general policy applying to the whole document, plus an exception for the node to which the policy should not apply. A possible solution for exception management is to support both positive and negative authorizations/policies, where a negative authorization/policy expresses an explicit denial. This simple distinction allows one to easily specify exceptions. However, the flexibility provided by positive and negative authorizations/policies give rise to possible conflicts between policies/authorizations. For instance, it could be the case that two policies with opposite sign are specified for the same subject, protection object, and privilege. In this case a conflict resolution policy is needed stating which is the access control policy that should prevail, when evaluating an access request.

Constraint modeling. Depending on the scenario where access control takes place, the access control decision may depend on several additional factors, such as for instance the time of the access request, etc. Such additional conditions can be modeled by a set of *constraints*, expressing the additional checks that should be performed to decide whether an access request can be granted or not. For instance, an access can be granted or not, on the basis of the user location. This implies that the access control model should support the specification of policies/authorizations that hold only if the

user is in a particular position (e.g., inside a department). This feature is particularly relevant in the context of mobile computing, where the actions a user is authorized to do can be constrained by the location of mobile users. A further relevant class of constraints is related to the temporal dimension of access control. This implies that the access control model should support the specification of policies/authorizations that hold only for specific periods of time (such as for instance a particular day of the week). This is a useful feature, since there are several real situations in which subjects should have the right to access a datum only in specific periods of time. As an example, consider the case of a newspaper company offering several types of subscriptions, which differ for the subscription period (one year, six-month, only the week-end, and so on). The possibility of constraining the period of validity of an access control policy/authorization makes easier access control management for scenarios like the above one, thus simplifying SA's tasks. Finally, there are scenarios in which the access decision should be taken not only on the basis of the access request information (i.e., who is the requestor, which data requires, and under which access modes), but also by constraining the requiring user or the system to perform some actions (i.e., *provisional actions*) before accessing the data. For instance, this makes the SA able to specify access control policies allowing a user to access confidential information, only if its signature has been verified and the access has been logged by the system. Thus, if the access control mechanism supports this feature, the access is granted provided that certain actions are taken by the requesting user.

IV Access control models and policy languages for the web

In this section we review main results in the area of access control for web data. In doing that, we focus on XML. This is due to the fact that, since XML is today the new standard for data representation on the web, in the last years a great research effort has been done in developing access control solutions for the protection of XML data sources. More precisely, in this section we focus on two main issues. First, we discuss the main access control models for XML data proposed so far. Then, we discuss XML-based languages for access control policy specification.

IV.1 Access control models for XML data

In this section, we summarize the main features of the most relevant access control models proposed so far for the protection of XML data, by pointing out whether and how they satisfy the access control requirements introduced in Section III. In doing that, we organize models into two categories: discretionary models and role-based ones. For each group, we present a comparison table summarizing the main differences among the proposals with respect to the considered requirements.

Discretionary access control

Author-$\mathcal{X}$. The access control model proposed in [10] supports the specification of access control policies at varying granularity levels, as well as the specification of subject credentials, as a way to enforce access control based on subject qualifications and profiles. Author-$\mathcal{X}$ has been further extended in [8] to support the specification of temporal constraints. Author-$\mathcal{X}$ access control policies can be modelled as tuples: *(temp_cnstr, subject, object, privilege, propagation, sign)*, where *subject* is an XPath expression on the XML encoding of subject credentials, denoting subjects allowed to exercise *privilege* on the protection objects identified by *object*. The object specification is defined through an XPath expression, allowing thus a content-based access control at a fine granularity level (i.e., any element and attribute of the XML document). Moreover, the access control policy has a temporal constraint stating the period of time during which the policy holds. Exception management is obtained by supporting both positive (*sign* = +), and negative (*sign* = -) policies. Author-$\mathcal{X}$ supports *browsing privileges*, that allow a subject to read the information

in a protection object and to navigate through its links, and *authoring privileges*, that allow a subject to modify protection objects according to different modes (i.e., update, delete, insert). Author-$\mathcal{X}$ also supports policy propagation by providing two different options: *implicit* and *explicit propagation*. The first type is applied by default and exploits DTDs/XML Schemas (i.e., policies specified on a DTD/XML Schema propagate to all the DTD/XML Schema instances), or the node relationships (i.e., elements/subelements, or element/attributes relationships). By contrast, explicit propagation makes the SA able to state whether and how a policy specified on a given protection object propagates to lower level protection objects with respect to the document or DTD/XML Schema hierarchy. The explicit propagation options supported by Author-$\mathcal{X}$ are: *i)* no propagation; *ii)* the policy propagates to all the direct subelements of the elements in the specification; *iii)* the policy propagates to all the direct and indirect subelements. Finally, possible conflicts between positive and negative policies are solved based on the notion of *strongest access control policy*. According to this principle, access control policies specified at the document level prevail over policies specified at the DTD/XML Schema level, and policies specified at a certain level of a document or DTD/XML Schema hierarchy prevail over policies specified at higher levels. When conflicts cannot be solved by using these criteria, negative policies are considered as prevailing.

Damiani et al. The model proposed in [13, 14] allows the specification of policies based both on subject identity (i.e., IDs), and subject location (i.e., IP addresses). In both cases, the model supports the specification of groups: subject's groups and location patterns. A location pattern is an expression identifying a set of physical locations. Regarding the object specification, the model supports both instance and schema level specification, and exploits XPath to obtain a fine-grained and content-based access control. Access control policies are modelled through a tuple: (*subject, object, privilege, sign, propagation*). In particular, the propagation option can be of two different types: local propagation, that states that a policy applied to an element must be propagated to its attributes, and recursive propagation, specifying that a policy applied to an element must be applied also to all its descendants. Both types of propagation can be applied to XML documents as well as to DTDs (i.e., a policy applied to a DTD can be propagated to all its instances). Damiani et al. model also provides exception management based on the *sign* of the policy. Conflict resolution strategies exploit both the most specific subjects take precedence principle[1] and the denials take precedence principle. More precisely, when a conflict arises the access control mechanism first enforces the subjects take precedence principle, then, if the conflict is still not solved, it further applies the denials take precedence principle.

Gabillon et al. The authors propose a discretionary access control model for XML data [19], where access control policies are defined by a tuple: (*subject, object, sign, priority*), where the *subject* component specifies the user ID or groups to which the policy applies, and the *object* component is an XPath expression identifying a nodeset within an XML document. The only access mode considered by Gabillon at al. model is *read*. Policies can be either positive or negative, according to the value of the *sign* component (called *access* in [19]). The access control model supports explicit propagation in that if a subject is allowed to access a node n, automatically he/she is allowed to access also the sub-tree rooted at n. Exception management is based on sign. Additionally, access control policies could have an optional *priority* component, which states the importance of the policy in evaluating an access request. More precisely, the priority is exploited for conflict resolution, in that, whenever two conflicting policies apply to the same node, the policy with the highest priority is considered prevailing. In the case of two or more policies with highest priority, the last one in the policy base is selected.

In someway related to XML access control, there is another model proposed by Gabillon at. al

[1]Specificity is determined by the partial order defined over subject's groups.

in [20]. More precisely, the authors adapt the System R model for a native XML database obtaining thus grant-option-enabled access control policies.

Murata et al. Almost all the access control models proposed for the protection of XML data exploits XPath to denote the authorization protection objects. Therefore, the access control mechanism relies on some engine for XML query languages. An alternative approach has been presented by Murata et al. in [25], which propose a static analysis approach to access control. Given an access control policy, a query (i.e., the access request), and the schema of the requested document, the mechanism proposed by Murata et al. exploits a static analysis tool, which, by means of an automata, is able to verify whether the query expression accesses only authorized nodes with respect to the access control policy. The discretionary access control model underlying the proposed mechanism supports policies with the following format: (*subject, object, sign, privilege, propagation*), where subjects are identified by means of IDs, as well as groups, and roles. On the basis of the value of the *sign* component, an access control policy states the action (i.e., *privilege*) the denoted subjects are allowed (or not allowed) to perform on the denoted protection objects (i.e., *object* component), which are specified by means of XPath expressions. The model proposed in [25] supports also a propagation mechanism, in that a policy that applies to a node n could be automatically applied to all nodes belonging to the subtree rooted at n. Possible conflicts are solved according to the denials take precedence principle.

Kudo and Hada. The novelty of the model proposed by Kudo and Hada in [24] is that the authors incorporate provisional actions into a discretionary access control model for XML data. In the resulting model, a subject request is authorized provided that the requesting subject (and/or the system) performs certain security actions. An example of provisional authorization (taken from [24]) is: 'You are allowed to read sensitive data, but you must sign a terms and conditions statement first'. In the Kudo and Hada model, policies are modeled as tuples: *(subject, object, sign, privilege, context, provisional_action, propagation)*, where *privilege* states the access mode, *context* is any data related to the specific requested action (e.g., time, location), and *provisional_action* is a set of actions that the requesting subject has to perform. The model supports positive and negative policies, and three different types of *propagation*: *i*) no propagation; *ii*) propagation up, according to which an access control policy applied to a node n is propagated to all n's parent elements; *iii*) propagation down, according to which an access control policy applied to a node n is propagated to all its descendants. In order to solve possible conflicts among policies, the model allows one to select among three different conflict resolution policies: *i*) denials take precedence; *ii*) permissions take precedence; *iii*) nothing takes precedence. If the latter option is selected, the access control mechanism applies a default policy specified for each document by the SA.

A summary of the characteristics of the models described so far is reported in Table I. Table I shows that all the models use XPath to denote the objects to which a policy applies, and the majority of them support content-dependent access control, as well as DTD/XMLSchema level and document level policies. As far as access modes are concerned, only Author-$\mathcal{X}$ and Kudo and Hada model support both browsing and authoring privileges, whereas all the other models only consider read operations. Propagation is provided by all the models, whereas most of the models support both positive and negative policies as a means for exception management. The aspect on which no consensus among the models has been still reached is that of constraint modelling. Indeed, among the considered models, only two support constraints: Author-$\mathcal{X}$, which supports temporal constraints, and the Kudo and Hada model, which supports provisional actions. The development of a general constraint model is therefore still an open issue.

Role-based access control

Zhang et al. Zhang et al. [34] proposed a role-based access control model for XML data. The

Requirement	Author-$\mathcal{X}$	Damiani et al.	Gabillon et al.	Murata et al.	Kudo and Hada
Subject	Subject credentials	Subjects and IP groups	Subjects and groups	Subject Ids, roles and groups	Subject Ids, roles and groups
Object					
granularity	any element and attribute specified by XPath	any element and attribute specified by XPath	any element and attribute specified by XPath	any element and attribute specified by XPath	any element and attribute specified by XPath
content	yes	yes	yes	yes	yes
intensional	yes	yes	-	yes	yes
Access mode	browsing and authoring privileges	read	read	read, write	read, write create, delete
Propagation	implicit and explicit	local and recursive	automatically to subtrees to which policies apply	to subtrees to which policies apply	up and down
Exception management	based on sign	based on sign	based on sign	based on sign	based on sign
Conflict resolution	strongest access policies prevail	most specific subject/ deny takes precedence	based on priority	deny takes precedence	deny/permission takes precedence
Constraints	Temporal	-	-	-	provisional actions

Table I: Discretionary access control models comparison

authors exploit the flexibility of XML Schemas for representing permissions. More precisely, in [34], they extended the RBAC96 model [28] by modifying its mechanism for assigning permissions to roles. Indeed, in the model proposed by Zhang et al. permissions are directly specified on XML schemas. More precisely, with each schema component some atomic access types (i.e., read, create, update, delete) are associated, obtaining what they call *schema-based permissions*. Assignments of permissions to roles is carried out by associating roles directly to schema-based permissions. The model proposed in [34] supports the propagation of permissions, which can be specified by the SA in both directions, that is, permissions can propagate to sub-components (i.e., sub-elements, attributes) or to super-components (i.e., ancestor elements).

Wang et al. A further role-based access control model for XML data is the one proposed in [35], that shows how it is possible to combine role-based access control with methodologies designed for object-oriented databases, for protecting XML data. More precisely, the authors investigated how to exploit an object-oriented model for handling implications among privileges in the context of RBAC. The results have, then, been applied in the context of XML data. More precisely, the model presented in [35] considers roles for the subject specification, XPath expressions for object identification, and supports reading and authoring privileges (i.e., update, extend). The access control model also provides a notion of privileges propagation.

Table II presents a comparison of the role-based models previously discussed.

IV.2 Access control policy languages for web data

In this section we illustrate the most relevant languages devised for access control policy specification, by focusing on XML-based languages. In doing that we classify access control policy languages into two groups: general purpose access control languages that exploit an XML-based syntax, and access control policy languages designed to support access control models for XML data.

XML-based general purpose access control policy languages

Requirement	Zhang et al.	Wang et al.
Subject	Roles	Roles
Object		
granularity	any element specified by XPath	any element specified by XPath
content	yes	yes
intensional	yes	-
Access mode	read, create, update, delete	read, update, extend
Propagation	yes, in both directions	yes, prop of privileges
Conflict resolution	-	-
Exception management	-	-
Constraints	-	-

Table II: Role-based access control models comparison

These kinds of languages exploit an XML-based notation to specify policies that apply to any web resource (not necessarily XML data). In this area, the most relevant standardization effort has been carried out by the OASIS consortium, which proposed XACML [27]. XACML (extensible Access Control Markup Language) is an XML-based language that allows one to express, in a simple and flexible way, and enforce access control policies in several different environments, using a single language. The XACML specification has a twofold goal: it describes the vocabulary and syntax of the language devised for expressing the access control policies, and it states the framework to support an automated access control decision process, based on such a language. We start by analyzing the proposed language.

In order to allow a flexible access control policy specification, an XACML policy is specified by exploiting three distinct elements: *Rules*, *Policy*, and *PolicySet*. By defining who can access which information and under which mode, the Rule element represents a simple way for combining distinct rules, whereas the PolicySet element consists of a set of Policy elements. The main components of the Policy element are as follows.

- *Rule.* With each policy is associated one or more rules. A rule specifies an authorization (or a denial) for a subject to perform an action on a specific object. Each rule consists of a target, an effect (for instance, permit, deny), and a condition. By means of the target it is possible to specify the subject, object and action to which the rule is intended to apply; whereas the condition component allows further refinement of the subject/object specification (for instance, a predicate that must be evaluated on the object).

- *Target.* A policy element contains a unique Target element, which specifies the subject, resources and actions to which the policy applies. The target element of a policy may be either specified directly by the policy writer, or calculated from the target elements associated with the contained rules.

Figure 1: An example of XACML policy [27]

- *Obligations.* A policy could have attached an optional set of obligations, that is, a set of operations that must be performed during the authorization decision process.

- *Rule-combining algorithm.* According to the XACML specification, a policy can have multiple rules, which can also generate a conflict. In such a case, the rule-combining algorithm specified in the XACML policy states how the conflict should be solved. Examples of rule-combining algorithms supported by XACML are: deny-overrides; ordered-deny-overrides, which is similar to the previous one apart from the order in which rules are evaluated; permit-overrides; ordered-permit-overrides; first-applicable, where the authorization decision is taken by considering the first evaluated rule. Additionally, the XACML specification makes the SA able to define its own rule-combining algorithms.

Figure 1 shows an examples of XACML policy, taken from the XACML specification [27]. The policy states that any user having an e-mail address defined in the 'medico.com' domain is allowed to perform any action on any resources in that domain. In particular, the `Policy` element in Figure 1 states deny overrides as the rule-combining algorithm, whereas it does not specify a target or an obligation. The authorization is specified by means of a unique `Rule` element, having Permit as effect, and whose `Target` element states the conditions on subjects, objects and actions. The Rule's `Target` element states that no conditions are applied to objects and actions (i.e., AnyResource, AnyAction); whereas to subjects a condition matching the subject name with 'medico.com' is applied.

According to the XACML specification the authorization decision process follows the model illustrated in Figure 2. As reported in Figure 2, an access request arrives at the *Policy Enforcement Point* (PEP), that, in turn, sends it to the *Context Handler*. This last component is in charge of creating the corresponding XACML request, by translating the access request in a domain-specific

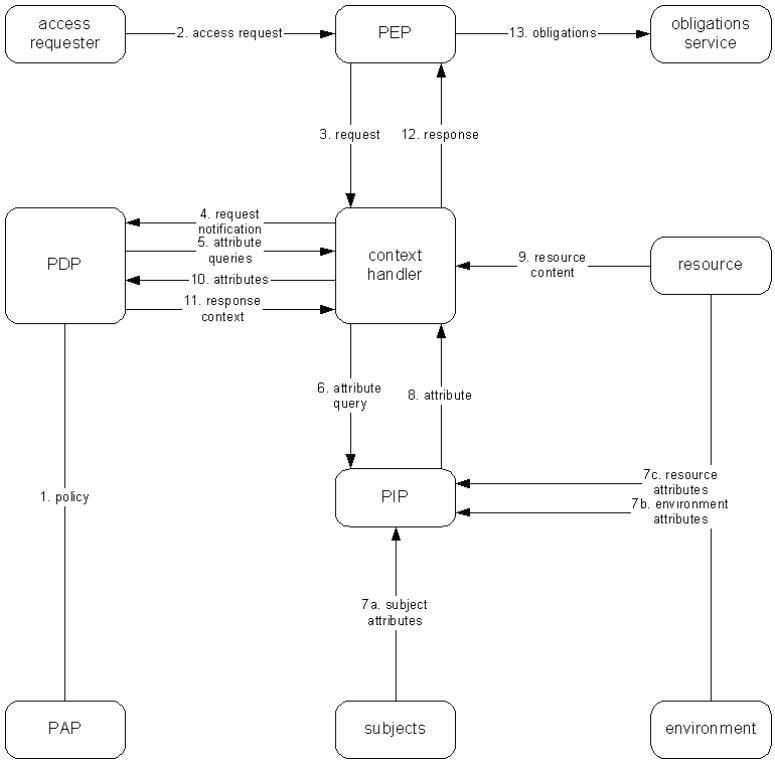

Figure 2: The data-flow model of XACML authorization decision process [27]

form into the XACML format. The resulting XACML request is then sent to the *Policy Decision Point* (PDP), that evaluates the access request and sends back a response. To make this decision, the PDP processes all and only those policies and/or rules that are relevant for the submitted access request. These policies are retrieved by the *Policy Access Point* (PAP), on the basis of the policy's target. Moreover, in order to evaluate the access request, the PDP needs attributes values about the subject, the resource, the action, or the environment corresponding to the request. This information are given to the PDP by the Context Handler, which exploits also the Policy Information Point (PIP) for retrieving the attribute values of the requesting subject. Finally, the authorization decision elaborated by the PDP is sent to the PEP, which, after performing the obligations actions, if any, permits or denies the access.

Access control policy languages for XML data

Besides the XACML standard proposal, several policy languages have been proposed for specific access control models for XML data. Examples of languages falling in this category are the *XACL* language, based on the provisional access control model proposed by Kudo and Hada in [24], and *X-Sec* [9], the policy language of Author-$\mathcal{X}$. There are also other XML-based languages, like, for instance, *Xupdate* [20], which allows the SA to specify access control policies for native XML databases. More precisely, this language borrows the functionalities of GRANT and REVOKE commands of the System R model, by adapting them to XML. In [20] the GRANT/REVOKE commands are extended to handle the propagation of an access control policy applied to a node n to all its descendants.

In order to show the main characteristics of these languages, in what follows, we focus on $\mathcal{X}$-Sec, since it allows the specification of both subject credentials and access control policies for XML data.

We start by describing how it models subject credentials, then we present the policy specification syntax.

In $\mathcal{X}$-Sec credentials with a similar structure are grouped into *credential-types*. A credential-type is a DTD and consists of a set of properties specifications, where each of them consists of the name and the domain of a property. $\mathcal{X}$-Sec allows the specification of both simple and composite properties. An $\mathcal{X}$-Sec credential is therefore an XML document, instance of the corresponding credential-type. $\mathcal{X}$-Sec credentials are issued by a certification authority, which is in charge of certifying properties asserted by credentials and certifying them by using standard digital signature techniques.

```
<!DOCTYPE carrier_employee[
<!ELEMENT carrier_employee (name,address,
        phone_number*,email?,company)>
<!ELEMENT name (fname,lname)>
<!ELEMENT address (#PCDATA)>
<!ELEMENT phone_number (#PCDATA)>
<!ELEMENT email (#PCDATA)>
<!ELEMENT company (#PCDATA)>
<!ELEMENT fname (#PCDATA)>
<!ELEMENT lname (#PCDATA)>
<!ATTLIST carrier_employee credID
                ID #REQUIRED>
<!ATTLIST carrier_employee CIssuer
                CDATA #REQUIRED > ]>
```

(a)

```
<carrier_employee credID=‘‘192" CIssuer = ‘‘CA72">
  <name>
    <fname> Alice </fname>
    <lname >Reds </lname>
  </name>
  <address> 16 Yellow Street </address>
  <phone_number> 0123456789 </phone_number>
  <email> a.reds@myorganization.com </email>
  <company> MyOrganization </company>
</carrier_employee>
```

(b)

Figure 3: An example of (a) $\mathcal{X}$-Sec credential-type and (b) a subject credential instance of it

An example of $\mathcal{X}$-Sec credential-type and credential is given in Figure 3. To facilitate subject credentials evaluation, all credentials associated with a specific subject are grouped into a unique XML document, called $\mathcal{X}$-*profile*.

We consider now access control policies. The basic concept underlining the proposed policy language is that of *policy base* that, according to $\mathcal{X}$-sec, is an XML document containing a distinct element, called `policy_spec`, for each access control policy defined for a given XML source.

Figure 4 reports the DTD modelling a policy base, whereas Figure 5 shows a policy base instance of the DTD in Figure 4. The policy base encodes a set of access control policies specified for an XML document, called purchase.xml, which stores purchase orders of an organization, grouped by sale areas.

```
<!DOCTYPE policy_base[
<!ELEMENT policy_spec EMPTY>
<!ATTLIST policy_spec cred_expr CDATA #REQUIRED>
<!ATTLIST policy_spec target CDATA #REQUIRED>
<!ATTLIST policy_spec path CDATA>
<!ATTLIST policy_spec priv CDATA #REQUIRED
<!ATTLIST policy_spec type CDATA #REQUIRED
<!ATTLIST policy_spec prop CDATA #REQUIRED
]>
```

Figure 4: $\mathcal{X}$-sec policy base template [9]

```
<policy_base>
  <policy_spec cred_expr="//Seller/[@level="top"]" target="purchase.xml" path=
    {"/Area/Profits" } priv="VIEW"type="GRANT"
    prop="CASCADE" / >
  <policy_spec cred_expr="//Seller/[@level="simple"] AND [@area="east"]" target="purchase" path=
    {"//Area[@Name='East']//node()" } priv="VIEW"type="GRANT"
    prop="NO_PROP"/ >
  <policy_spec cred_expr="//Seller/[@level="simple"] AND [@area="west"]" target="purchase.xml" path=
    {"//Area[@Name='West']//node()" } priv="VIEW"type="GRANT"
    prop="NO_PROP" / >
</policy_base>
```

Figure 5: An example of $\mathcal{X}$-Sec policy base

V Policy languages for the semantic web

The availability of data on the web is constantly expanding. Most of these data are semi-structured, consisting of text and multimedia. Overall, these data constitute the largest body of information ever accessible to any individual. The problem with such data is that they are not machine-understandable. Due to the volume of information the web contains it is not possible to manage it manually. A promising solution is that of using metadata to describe the data contained on the web and this concept has evolved into the semantic web, the vision of Tim Berners-Lee [5].

For the semantic web to be secure we need to ensure that all of the layers of the semantic web are secure. This includes secure XML, secure RDF, secure ontologies, and secure information integration. In the following, we consider all these aspects, by mainly focusing on the policy language component. That is, we will discuss a broad range of security issues for the semantic web.

V.1 Policy languages based on XML and RDF

While security has to be incorporated into the layers of the semantic web such as enforcing security policies on XML and RDF documents, one can also use XML and RDF [12] to specify security policies themselves. Work related to XML have been discussed in Section IV. The question is, are XML and RDF sufficient to specify all types of policies including role-based policies, where access is granted to users based on their roles, usage control policies, consisting of authorizations, obligations and conditions, obligations are actions that are required to be performed before or during the access process and conditions are environment restrictions that are required to be valid before or during the access process, and administration policies? Do we need a more expressive language such as rule languages or languages based on descriptive logics? Can ontology languages be used to specify policies? Some ideas are explored in the following sections. RDF specifies semantics. While XML is limited in providing machine understandable documents, RDF handles this limitation. As a result, RDF provides better support for interoperability as well as searching and cataloging. It also describes contents of documents as well as relationships between various entities in the document. While XML provides syntax and notations, RDF supplements this by providing semantic information in a standardized way.

For example, suppose we have the statement 'all engineers are also staff officers'. However we have only defined John and Mary as staff officers and Jim and Bill as engineers. If we query to retrieve all staff officers, we will get only John and Mary in an XML specification. In RDF we can specify subclass relationships that is ENGINEER is a subclass of STAFF-OFFICERS and therefore we will get John, Mary, Jim and Bill in the answer. The basic RDF model has three types: resources, properties and statements. Resource is anything described by RDF expressions. It could be a web page or a collection of pages. Property is a specific attribute used to describe a resource. RDF statements are resources together with a named property plus the value of the

property. Statement components are subject, predicate and object. So for example, if we have a sentence of the form 'John is the creator of document X', then document X is the subject or resource, property or predicate is 'Creator' and object or literal is 'John'. RDF diagrams are very much like say semantic nets or object diagrams and represent statements. There are various aspects specific to RDF syntax and for more details we refer to the various documents on RDF published by W3C [39, 40, 41]. Also, it is very important that the intended interpretation be used for RDF sentences. This is accomplished by RDF schemas. Schema is a sort of dictionary and has interpretations of various terms used in sentences. RDF and XML namespaces resolve conflicts in semantics. More advanced concepts in RDF include the container model and statements about statements. The container model has three types of container objects, namely Bag, Sequence, and Alternative. A bag is an unordered list of resources or literals. It is used to mean that a property has multiple values but the order is not important. A sequence is a list of ordered resources. Here the order is important. Alternative is a list of resources that represent alternatives for the value of a property. RDF also provides support for making statements about other statements. For example, with this facility one can make statements of the form 'The statement A is false' where A is the statement 'John is the creator of X'. Again one can use object-like diagrams to represent containers and statements about statements. RDF also has a formal model associated with it. This formal model has a formal grammar. As in the case of any language or model, RDF will continue to evolve. Now to make the semantic web secure, we need to ensure that RDF documents are secure. This would involve securing XML from a syntactic point of view. However with RDF we also need to ensure that security is preserved at the semantic level. The issues include the security implications of the concepts resource, properties and statements. That is, how is access control ensured? How can resources, properties and statements be protected? Other issues include how can one provide access control at a finer granularity? What are the security properties of the container model? How can bags, lists and alternatives be protected? How can we express security policies in RDF? How can we resolve semantic inconsistencies for the security policies?

V.2 Policy Languages based on web rule languages

Web rules language is being developed by W3C to specify various types of policy rules. We can use the rules to specify security policies also. Recently Berners Lee, Hendler and others are examining web rules to specify various access control rules. How expressive is the web rules language? Is it as powerful as XML, RDF and logic-based specifications? One advantage of using a rules language is to utilize the rules engine to reason about security. Some details on the web rules language can be found in [1, 11, 22].

Example of a web rules language is Rules ML [29] which is a rule markup language based on XML syntax. Rules ML utilizes both monotonic and non monotonic reasoning. In a monotonic system if the premise of a rule is satisfied then the consequences will follow. In a non monotonic system, even if the premises are satisfied, alternatives have to be considered before coming to a conclusion. For example if a student wants to select a course for his/her class, he/she may specify his/her requirements in Rule ML. The reasoning engineer based on the logic of the rules language then reasons about the rules and comes with appropriate choices for the student. The reasoning engine may be monotonic or non monotonic.

With the rule processing capability, one can determine whether security violations via inference can occur using the inference capabilities of the semantic web. Note that the semantic web has inferencing capabilities built into it together with the support for rules specification and rules manipulation. This would exacerbate the inference and privacy problems as one can use this built in capability to make all kinds of unauthorized inferences. Therefore, we need to examine the inference and privacy control techniques and determine their applicability for the semantic web

[32, 33].

V.3 Policy languages and ontologies

Ontology specification and management is a key aspect of the semantic web. An ontology is an explicit specification of a conceptualization [23]. Ontologies play an important role in providing an ability to model, represent and exchange formal conceptualizations as well as information of particular domains in a precise, machine-understandable form. An ontology can be constructed in two ways, i.e., domain dependent and generic. Generic ontologies attempt to provide a general framework for most categories. Generic ontologies are generally very large, difficult to build, and void of ample details. A domain-dependent ontology will contain a conceptual model about some domain such as say a medical domain. Now with respect to access control there are two major issues. One is how can access control be specified on ontologies? How can access control policies be specified? Can we use XML and RDF to specify policies or do we need ontologies to represent more powerful security policies. Can we use ontology languages such as OIL (Ontology Interface Language) to specify policies? We will discuss ontologies with a specific example. OWL-S [44] is the specific technology used to define the web ontology. To make use of a web service, a software agent needs a computer-interpretable description of the service, and the means by which it is accessed. An important goal is to establish a framework within which these descriptions are made and shared. Web sites should be able to employ a set of basic classes and properties for declaring and describing services, and the ontology structuring mechanisms of OWL provides the appropriate framework within which to do this. The challenge for security is whether OWL can be extended to provide a security framework within which one can reason about the service security properties. In summary, we need to protect ontologies from unauthorized accesses. We can also use ontologies to specify security policies.

V.4 Policy integration and information interoperability

While there is an urgent need for organizations to share data across the semantic web so that the big picture is formed there is also a need to protect the information within an organization. Essentially we have a conflict between data sharing and data security. The challenge is to enforce appropriate administration and security policies that facilitate data sharing as needed. The various data sources may have their own policies. When the information is integrated, the policies also will have to be integrated. Policies such as role-based access control and usage control policies need to be examined for policy integration across semantic web [26, 30]. Let us consider policy integration based on role-based access control [17]. RBAC is relevant to the protection of information in a local semantic web as well as in a global semantic web across a coalition. We need administration of roles and cross-organizational roles for the semantic web. With traditional approaches a human is in the loop for many administrative functions. Therefore a challenge for managing security in the semantic web environment is to make the administration as seamless as possible. The assignment of users and permissions to appropriate roles should occur transparently as part of the normal workflow of the organization. Different local organizations may use completely different roles or use same role names but with very different meaning. The hierarchical relationships between the roles may be different and possibly inconsistent. For example, in some semantic web environments, a Professor role may be senior to a Student role whereas in other semantic web environments within the same University the Student role may be senior to Professor. How to reconcile these differences effectively in building a consistent and useful global semantic web across organizations remains a challenge.

V.5 Policies, trust, privacy and logics

Policies are needed for trust and privacy. Semantic web is utilized by numerous users from multiple organizations. This means how can you trust the users and the data that are produced. If A trusts B and B trusts C, then does A trust C? How can trust be propagated? How can we trust the quality of the data? Does it mean that if data originates from source X and passes through source Y then we can trust the data only 50% of the time? We need mechanisms to specify such trust policies. Ideally, we need common policy languages to specify security as well as trust policies. The challenge is do we use XML, RDF, Ontology languages or the web Rules language? We need to conduct an evaluation of the various languages. Next, let us consider privacy. As stated in the specification [38], the Platform for Privacy Preferences Project (P3P), developed by W3C, is emerging as an industry standard providing users the support to maintain their privacy. Essentially P3P is a standardized set of multiple-choice questions, covering all the major aspects of a web site's privacy policies. It shows privacy policies enforced by a web site and users are given the choice to mark what they want to release and what they want to make private. P3P-enabled web sites make this information available in a standard, machine-readable format. P3P enabled browsers can read this snapshot automatically and compare it to the consumer's own set of privacy preferences. The challenge is to specify additional policies that a user wants and are not supported by P3P. Essentially, we need to examine P3P and examine extensions that would cover a broader set of policies. Finally, let us consider the various descriptive logics that have been developed for the semantic web. Descriptive logics evolved from classical logic and knowledge presentation languages. One can also specify access control policies as well as privacy and trust policies based on statements in descriptive logics. With such an approach one could use the theorem provers of the logics to test for consistency and completeness of the policies. However the logic may not be as expressive as say XML, RDF, and web rules language. This is an area that is worth exploring. For example, these logics have been used to specify ontologies in languages such as OWL. As stated in [2], descriptive logics permit efficient reasoning support. OWL-DL uses descriptive logic for specification, and the reasoning built into such a system can be used to reasoning about the application. Therefore, we need to explore reasoning about security, trust and policy properties using such logics. Further details on descriptive logics can be found in [43, 45].

VI Conclusions

The focus of this paper has been on access control for the web and the semantic web. We started by providing some background information on access control, then we discussed which are the main access control requirements of web data. The most important proposals of access control models and policy languages for web data have also been presented. Finally, the last part of the paper has been devoted to the semantic web and related policy languages.

References

[1] Antoniou (G.), Billington (G.D.), Governatori (G.) and Maher (M.), Representation Results for Defeasible Logic, *ACM Transactions on Computational Logic*, Volume 2, 2001.

[2] Antoniou (G.) and van Harmelen (F.), A Semantic Primer, *MIT Press*, 2003.

[3] Air Force Studies Board, Committee on Multilevel Data Management Security, Multilevel Data Management Security, National Academy Press, 1983.

[4] Bell (D.) and LaPadula (L.), Secure Computer Systems: Unified Exposition and Multics Interpretation. ESD-TR-75-306, Hanscom Air Force Base, Bedford, MA, 1975.

[5] Berners-Lee (T.), Hendler (J.), and Lassila (O.), The Semantic Web. *Scientific American,* May 2001.

[6] Bertino (E.), Carminati (B.), Ferrari (E.), Thuraisingham (B.), and Gupta (A.), Selective and Authentic Third-Party Distribution of XML Documents, *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 16(10):1263–1278, 2004.

[7] Bertino (E.) and Ferrari (E.), Development of Multilevel Secure Database Systems, *Database and Data Communication Network Systems*, Academic Press, 2002.

[8] Bertino (E.), Carminati (B.), and Ferrari (E.), A Temporal Key Management Scheme for Broadcasting XML Documents, In Proc. of the *9th ACM Conference on Computer and Communications Security (CCS'02)*, Washington, November, 2002, ACM Press.

[9] Bertino (E.), Castano (S.), and Ferrari (E.), On Specifying Security Policies for Web Documents with an XML-based Language, In Proc. of the *ACM Symposium on Access Control Models and Technologies (SACMAT'01)*, Fairfax, VA, May 2001.

[10] Bertino (E.) and Ferrari (E.), Secure and Selective Dissemination of XML Documents, *ACM Transactions on Information and System Security (TISSEC)*, 5(3): 290-331, 2002.

[11] Boley (H.), The Rule Markup Language: RDF-XML Data Model, XML Schema Hierarchy, and XSL Transformations, Available at: `www.dfki.uni-kl.de/ boley/ruleml-mht.pdf`.

[12] Carminati (B.), Ferrari (E.), and Thuraisingham (B.), Using RDF for Policy Specification and Enforcement, In Proc. of the *DEXA International Workshop on Web Semantics - WebS 2004*. Zaragoza, Spain, 2004.

[13] Damiani (E.), De Capitani di Vimercati (S.), Paraboschi (S.) and Samarati (P.), Securing XML Documents, In Proc. of the *International Conference on Extending Database Technology (EDBT2000)*, Konstanz, Germany, 2000.

[14] Damiani (E.), De Capitani di Vimercati (S.), Paraboschi (S.) and Samarati (P.), A Fine-grained Access Control System for XML documents, *ACM Transactions on Information and System Security, (TISSEC)*, 5(2):169–202, 2002.

[15] De Capitani di Vimercati (S.), Paraboschi (S.) and Samarati (P.), Access Control: Principles and Solutions, *Software – Practice and Experience*, 33(5):397-421, 2003.

[16] De Capitani di Vimercati (S.), Samarati (P.), and Jajodia (S.), Policies, Models, and Languages for Access Control, In Proc. of the *4th Databases in Networked Information Systems (DNIS'05)*, LCNS 3433, 225 -237, Aizu, Japan, 2005.

[17] Ferraiolo (D.), Sandhu (R.), Gavrila (S.), Kuhn (D.), and Chandramouli (R.), Proposed NIST Standard for Role-based Access Control. *ACM Transactions on Information and System Security (TISSEC)*, 4(3):224-274, 2001.

[18] Ferrari (E.) and Thuraisingham (B.), Secure Database Systems, In O. Diaz and M. Piattini editors, *Advanced Databases: Technology and Design*, Artech House, London, 2000.

[19] Gabillon (A.) and Bruno (E.), Regulating Access to XML Documents, In Proc. of the *Fifteenth Annual IFIP WG 11.3 Working Conference on Database Security*, Canada, 2001.

[20] Gabillon (A.), An Authorization Model for XML DataBases, In Proc. of the *Workshop on Secure Web Services*, Fairfax, VA, USA, 2004.

[21] Griffiths (P.P.) and Wade (B.W.), An Authorization Mechanism for a Relational Database System, *ACM Transactions on Database Systems*, 1(3):242–255, September 1976.

[22] Grosof (B.), Labrou (Y.), and Chan (H.), A Declarative Approach to Business Rules in Contracts: Courteous Logic Programs in XML, In Proc. of the *1st ACM Conference on Electronic Commerce*, Denver, Colorado, USA, 1999.

[23] Gruber (T.R.), A Translation Approach to Portable Ontology Specifications, Knowledge Acquisition, *International Journal of Knowledge Acquisition for Knowledge-based Systems*, 5:199-220, 1993.

[24] Kudo (M.) and Hada (S.), XML Document Security based on Provisional Authorization, In Proc. of the *7th ACM Conference on Computer and Communication Security*, Washington D.C., USA, 2000.

[25] Murata (M.), Tozawa (A.), Kudo (M.) and Hada (S.), XML Access Control using Static Analysis, In Proc. of the *10th ACM Conference on Computer and Communication Security*, Washington D.C., USA, 2003.

[26] Park (J.), and Sandhu (R.), The UCONABC Usage Control Model, *ACM Transactions on Information and System Security*, 7(1), February 2004.

[27] OASIS Consortium, eXtensible Access Control Markup Language (XACML), Version 1.1, Available at: `http://www.oasis-open.org/committees/xacml/`

[28] Sandhu (R.). Role Hierarchies and Constraints for Lattice-Based Access Controls, In Proc. of the *4th European Symposium on Research in Computer Security (ESORICS'96)*, Rome, Italy, 1996.

[29] RuleML Initiative, http://www.ruleml.org/

[30] Sandhu (R.), Coyne (E.), Feinstein (H.), and Youman (C.), Role-Based Access Control Models, *IEEE Computer*, 29(2):38-47, February 1996.

[31] Stallings (W.), Network Security Essentials: Applications and Standards, *Prentice Hall*, 2000.

[32] Thuraisingham (B.), and Ford (W.), Security Constraint Processing in a Multilevel Distributed Database Management System, *IEEE Transactions on Knowledge and Data Engineering*, 7(2), 1995.

[33] Thuraisingham (B.), Security Standards for the Semantic Web, *Computer Standards and Interface Journal*, 27(3): 257-268, March 2005.

[34] Zhang (X.), Park (J.), and Sandhu (R.), Schema Based XML Security: RBAC Approach, In Proc. of the *17th IFIP 11.3 Working Conference on Data and Application Security*, Estes Park, Colorado, USA, August, 2003.

[35] Wang (J.), and Osborn (S.), A Role-based Approach to Access Control for XML Databases, In Proc. of the *9th ACM Symposium on Access Control Models and Technologies*, New York, USA, June, 2004.

[36] Winslett (M.), Ching (N.), Jones (V.), and Slepchin (I.), Using Digital Credentials on the World Wide Web, *Journal of Computer Security*, 5(3):255-267, December 1997.

[37] World Wide Web Consortium, Extensible Markup Language (XML) 1.0, 1998, Available at: `http://www.w3.org/TR/REC-xml`

[38] World Wide Web Consortium, The Platform for Privacy Preferences 1.0 (P3P1.0) Specification, W3C Recommendation, April 2002. Available at: `http://www.w3.org/TR/P3P/`

[39] World Wide Web Consortium, Resource Description Framework (RDF) Concepts and Abstract Syntax. Available at: `www.w3c.org/TR/rdf-concepts`.

[40] World Wide Web Consortium, Resource Description Framework (RDF) Primer, 2003. Available at: `www.w3c.org/TR/rdf-primer`.

[41] World Wide Web Consortium, Resource Description Framework (RDF) Semantics, 2004. Available at: `www.w3c.org/TR/rdf-mt`.

[42] World Wide Web Consortium, XML Path Language (Xpath), 1.0, 1999. Available at: `http://www.w3.org/TR/xpath`

[43] World Wide Web Consortium, OWL Web Ontology Language Overview, 2003. Available at: `www.w3c.org/TR/owl-features`.

[44] World Wide Web Consortium, OWL Web Ontology Language Reference, 2004 Available at: `http://www.w3.org/TR/owl-ref/`

[45] World Wide Web Consortium, OWL Web Ontology Language Guide, 2004. Available at: `www.w3c.org/TR/owl-guide`.