XML Schema in XML Documents with Usage Control

Lili Sun and Yan Li

Department of Mathematics and Computing
University of Southern Queensland, Toowoomba, QLD 4350, Australia

Summary
With an increasing amount of semi-structured data, XML has become significant to humans and programs. XML promoted by the World Wide Web Consortium (W3C) is rapidly emerging as a new standard language for semi-structured data representation and exchange on the Internet. XML documents usually contain private information that cannot be shared by all user communities. So securing XML data is becoming increasingly important and several approaches have been designed to protect information in websites. Usage control has been considered as the next generation access control model with distinguishing properties of decision continuity. It has been proven to be efficient to improve security administration with flexible authorization management. Usage control enables finer-grained control over usage of digital objects than that of traditional access control policies and models. XML Schema defines format and contents of XML instance. Access control based on a schema will be transported to all its instances. Also, XML Schema provides a mechanism to build relationships between schemas and elements.

In this paper, we present a usage control model to protect information distributed on the web, which allows the access restrictions directly at Schema-level and XML document-level. Finally, comparisons with related works are analysed.

Key words:
XML documents, Document Type Definition(DTD), XML Schema, Usage control, Authorization

1. Introduction

The extensible markup language (XML) [3] is a standard for describing the structure of information and content on the Internet over the past several years. XML is a fundamental component in many XML web services and it is used to store and exchange data in the Internet environment that may include private messages of customers. It overcomes the complexity of Standard Generalized Markup Language (SGML) [7] and the user can define document structures, removing the limit of the fixed tags in Hypertext Markup Language (HTML) [14]. The following example displays library customer information in an XML document.

```
<?xml version= "1.0" encoding= "UTF8"?>
   <bookInfo xmlns=
      "http://www.library.com/BooksInfo">
     <library city= "Toowoomba">
       <books>
        <available>
          <categorize> children's book
          </categorize>
          <story book>
             <description>
                <English comedy story>
             </description>
             <author >John Mahanee</author>
          </story book >
        </available >
        <loan>
          <categorize>magazine
          </categorize>
          <title> girlfriend f</title>
          <reader>
             <name > Tony </name >
             <address> Jilan street, 5
             </address>
             <city> Toowoomba </city>
          </reader>
         </loan>
       </books>
      </library>
    </bookInfo>
```

Table 1: XML Document Example

XML documents not only show the contents of data but also the constraints and relationships between data in Table 1. The element *bookInfo* includes *library* element, and *books* element is a sub-element of *library*. While sub-element *books* is combined with their own sub-elements. Since an XML document can express complex relationship between data, it can satisfy with varying security requirements. XML is used to store and exchange data in the Internet with including private messages, some users may like to access some particular parts of an XML document. In the above example in Table 1, for the *available* objects everyone can read all information. However, some users' access to information such as *loan* and *reader* will be restricted. This is because when an internal or external user accesses this document, his/her access permission has to be limited according to security

Manuscript received October 5, 2007
Manuscript revised October 20, 2007

policies. This example shows that secure XML documents form a significant topic for research.

We identify two levels at which authorizations on XML documents can be defined, instance and the Document Type Definition (DTD) [11, 17, 16]. Instance level authorizations denote privileges that only to a specific document. DTD level authorizations specify the privileges of all documents following a given DTD. XML Schema is an XML-based alternative to DTD. It defines the contents and relationships of elements in an XML instance. It supports complex constraints for XML components, such as elements, attributes, datatypes and groups. A well-validated XML document must follow the format specified by one or several schemas. In our proposed access model, a user access permissions are defined on schema or schema element level and will be transported to all XML instances specified by these schemas or elements. In the access control model the central authority uses XML schemas to specify the format of information to be changed. With the features of XML Schema, a flexible and easy-customized access control model can be achieved.

Access control has been considered as a major issue in information security community since the beginning of the Information security discipline. Through access control, the system can restrict unauthorized users access to the resources in the system and guarantees the confidentiality and integrity of the resources [4, 5, 12]. Traditional access control models primarily consider static authorization decisions based on the subjects' permissions on target objects. It focuses on the protection of data in a closed environment. More recently research in authorization is about trust management. Trust management [2] relates authorization to a user's capability and properties. These access models have used on the control of access to server-side objects. Digital rights management (DRM) [6] is used for objects disseminated. Current DRM solutions are largely focused on payment-based dissemination controls. Because each of access control, the authorization decisions are generated at request time but do not consider ongoing controls for long access or for revocation. We need a comprehensive, systematic approach for controls on usage of digital objects [13, 15]. Recently proposed usage control is a new access control model extending traditional access control models in multiple aspects. The term ‘usage’ means usage of rights on digital objects. The main different properties of usage control with traditional access control models are continuity of access decision and mutability of subject attributes and object attributes. In this paper, we propose authorization models which adopt usage control to manage access both at the instance-level and at the schema-level. Traditional access control gives an access request and an algorithm which computes a view of the target XML document based on the user's requirement's right. It has analysed authorization decisions on a subject's access to target resources before access. However, usage access control authorization decisions are not only checked and made before access, but also are repeat checked during the access period. Meanwhile obligations and conditions become decision factors for the management of XML documents.

The remainder of this paper is organized as follows: Section 2 illustrates the background of XML and two XML validations: XML DTD and XML Schema. The usage control model and Continuity properties are introduced in this section. Section 3 shows our proposed authorization models for usage control using XML Schema. It includes *pre-Authorizations, ongoing-Authorizations,*
pre-Obligations, ongoing-Obligations, pre-Conditions and ongoing-Conditions. Section 4 reviews the differences between this works from others. Finally, Section 5 concludes the paper.

2. Related technologies

2.1 XML

XML [3] is a markup language for describing semi-structured information. An XML document consists of *elements*, *attributes* and *text node*, each delimited by a pair of start and end tags (e.g. <price> and </price>) or by an empty tag. The content of each element is a sequence of elements or text nodes. An element has a set of attributes, each of which has a name and a value. XML documents can be classified into two categories: well-formed and valid. A document is said to be well-formed if it follows the grammar rules of XML, such as there is exactly one element that completely contains all other elements, elements may nest but not overlapped, *etc*. A well-formed document is valid only if it contains a proper DTD in the source and if the document obeys the constraints of that declaration. Validation requires an XML instance to contain specified elements and attributes. On the other hand, since XML is a structural transformation, it can transform one structure to another structure.

2.2 XML Validation: DTD and Schema

Document Type Definition (DTD) and XML Schema [8, 9] are two main validation specification mechanisms. DTD is the first and earliest language to define the structures and contents of XML documents. A DTD is a file which contains a formal definition of a particular type of XML documents. It contains or points to markup declarations

that provide a grammar for a class of documents. Each DTD is also a schema and XML documents corresponding to that DTD are instances of that schema. But it has some limitations: Firstly, a DTD file is neither a well-formed nor a valid XML document. It is difficult to specify constraints on structures and contents of XML instances with DTDs. Secondly, it is hard to handle the name confliction in DTDs. Such as a student who is involved in two departments, the two course names in each department will conflict if a DTD is applied by both of them. Finally, a DTD can not define datatypes. As a result, it makes difficult to use by other DTDs. On the other hand, an XML Schema is an XML-based alternative to DTD. An XML schema describes the structure of an XML document. XML Schemas express shared vocabularies and allow machines to carry out rules made by people. They provide a means for defining the structures, contents and semantics of XML documents. XML Schemas are extensible to future additions. XML Schemas are richer and more powerful than DTDs. XML Schemas are written in XML. XML Schema is an XML document itself. It supports data *type* and *namespaces*. Complex user-defined datatypes can be created in XML Schemas. *Namespace* is supported in XML Schemas to solve name conflictions. With these reasons, XML schemas are a richer and more powerful of describing information than what is possible with DTDs. It will be very soon that XML Schemas will be used in most Web applications as a replacement for DTDs. Since DTD is not XML well-formed and valid-formed, the access control policy on XML instance documents and DTD have to be implemented separately. By using schemas, we can define and enforce the permissions on schema objects and instance objects with a uniform mechanism.

The example below in Table 2 displays an XML Schema for a corresponding valid XML instance in Table 1.

```
<?xml version= “1.0”  encoding= “UTF-8”? >
   <xs:schema

targetNamespace=“http://www.library.com/BooksInfo”>
          xmlns:xs= “http://www.w3.org/2001/XMLSchema”
          elementFormDefault=”qualified”>
      <xs:annotation>
           <xs:documentation>
              Book Information Instance
           </xs:documentation>
      </xs:annotation>
    <xs:element name= “bookInfo”>
      <xs:sequence>
         <xs:element name= “books”  type= “xs:string”/>
              <xs:element name= “available”/>
                 <xs:element name= “categorize”
                              type= “xs:string”/>
                 <xs:complexType>
                    <xs:sequence>
                       <xs:element name=”storybook”/>
                         <xs:sequence>
                           <xs:element name= “description”
                                        type= “xs:string”/>
                             <xs:element name= “auyhor”
                                         type= “xs:string”/>
                           </xs:sequence>
                         </xs:complexType>
                       </xs:sequence>
  <xs:element name= “loan”>
   <xs:sequence>
      <xs:element name=“categorize”  type= “xs:string”/>
      <xs:element name= “title”  type= “xs:string”/>
      <xs:element name= “reader”  type= “readerInfoType”/>
    </xs:sequence>
  </xs:element>
  <xs:complexType name= “readerInfoType”>
    <xs:sequence>
       <xs:element name=“name”  type= “xs:string”/>
       <xs:element name= “address”  type= “xs:string”/>
       <xs:element name= “city”  type= “xs:string”/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

Table 2: XML Schema Example

An XML Schema defines elements and attributes that can appear in a document. It also defines which elements are child elements, the order of child elements and the number of child elements. It gives data types for elements and attributes, default and fixed values for elements and attributes.

2.3 Usage control

In this section we briefly review the general ideas of usage control and its authorization models. The usage control is a generalization of access control. It enriches and refines the access control discipline in its definition and cover obligations, conditions, continuity (ongoing controls) and mutability. There are eight core components in the usage control model: subjects, subject attributes, objects, object attributes, rights, authorizations, obligations, and conditions [13, 18] (see Figure 1). The authorization, obligations and conditions are components of usage control decisions. In the usage control model, the authorization rule permits or denies the access of a subject to an object based on subject and object attributes. Obligations are performed by subjects or by the system. Conditions are not related to subject or object attributes. They are system environment restrictions.

In the usage control model, subjects and objects are familiar concepts with traditional access control. Subject and object attributes can be used during the access

decision process. Subject attributes are identities, group names, roles, memberships, security clearance, and so on.

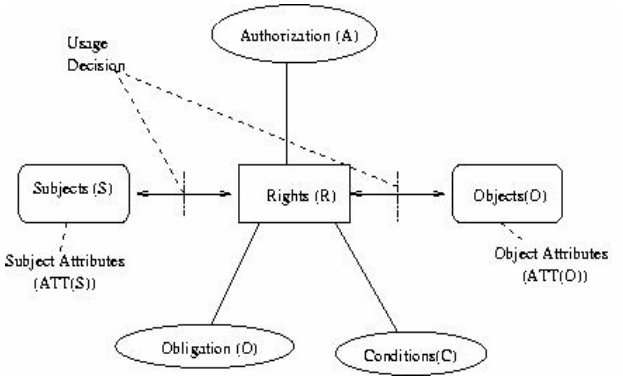

Fig.1 Components of Usage Control Model

Objects are entities that subjects hold rights on, whereby the subjects can access or use objects. For instance, in an on-line shopping store, a customer can be subject. A price could be an object attribute, the soybean machine is priced at $88 and with delivery is required at $98. Rights are privileges that subjects can hold on objects. The authorizations of rights require associations with subjects and objects. A right represents the access of a subject to an object, such as read or write.

Authorizations, obligations and conditions are decision factors used to check and determine whether a subject should be allowed to access an object. Obligations and conditions are new concepts that can resolve certain shortcomings that have been in traditional access controls. In general, the authorization of most traditional access controls are assumed to be done before access is allowed. However in the usage control model it extends this for continuous enforcement. Authorizations may require updates on subject and object attributes. The process of continuity properties in usage control model consists of three phrases, before usage, ongoing usage and after usage. To enforce control decisions, we have two different types: pre-decision and ongoing-decision. For mutability, there are three kinds of updates: pre-update, ongoing-update, and post-update. Therefore, Authorizations can be either pre-authorization (preA) or ongoing-authorization (onA). Pre-authorization is performed before authorization is required to the access. But ongoing authorization may be performed during the access, such as when a book stocking list in a bookstore is periodically checked while the access is in progress.

Obligations are requirements that a subject must perform before (pre) or during (ongoing) accesses. Conditions are decision factors that depend on environmental and system-oriented requirements. Subject and object attributes can be used to select which condition requirements have to be used for a request.

Based on the involvement of three decision factors: authorizations, obligations, and conditions, we focus on developing the usage control models for the XML Schemas and XML documents. We assume that a usage request exists on an XML target object. Decision-making can be done before (pre), during (ongoing) or after the exercise of the requested right. Based on the requirements we have six possible cases as a model for usage control: pre-Authorizations, ongoing-Authorizations, pre-Obligations, ongoing-Obligations,

3. Authorization models with XML Schema

In a real environment, the resource of XML objects based on different XML Schemas from various servers and organizations. Therefore there will be vast number of schema components. On the other hand, the vast number of users will make subjects complex. With all these features, the security administration will be very complex in both centralized and decentralized deployments.

An XML provides a uniform mechanism to solve problems in the environment. To apply the modularity, extensibility information flow in an XML format, all the components in our models, such as subjects, subject attributes, objects, object attributes, rights, authorizations, obligations, and conditions will be specified in an XML format. Through some XML operations, the content of a target XML document will be customized for a user to access with his permissions. Particular schemas will be identified security related information, and XML instances can be centrally stored or distributed among the organizations. Figure 2 shows the implementation layout based on our proposed architecture presented in Section 2. The entire messages transported among the services are identified in XML. XML requests and responses are XML messages, whose formats will be defined in schemas.

In this model, there are schema objects and instance objects. Schema objects (SO) are mapped to instance objects (IO) by an instance mapping (IM) function. In this authorization model, a Schema object is an XML Schema or schema component(s). An instance object is an XML instance or instance component(s). Instance Mapping is a mapping between SO and IO. In the following we explain the details of the model process.

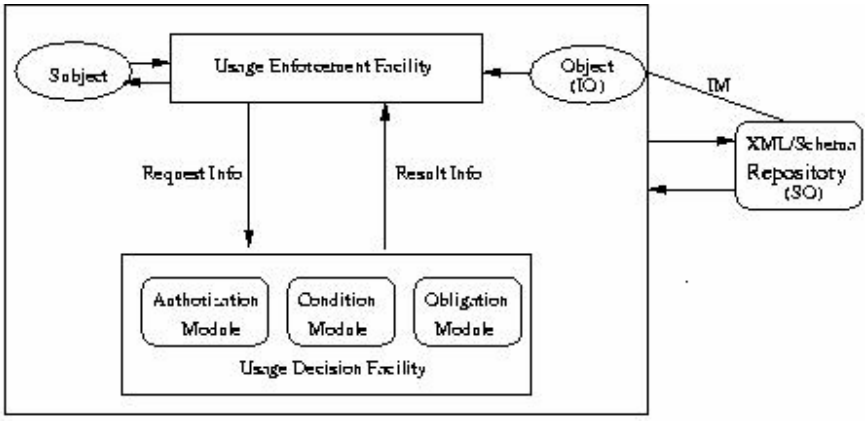

Fig.2 XML Architecture with usage control

A. Usage control for pre-Authorization Model (UCMpreA)

In a pre-Authorization usage control model, the decision process is performed before access is allowed. The following illustration of usage decision that can be expressed on the documents XML Schema and instance level are made in pre-authorizations.

The UCMpreA model consists of the following components: *S, XSch, XD, R, R1, ATT(S), ATT(XSch), ATT(XD)*, *IM* and usage decision Boolean functions *preA*, *preA1* on *XSch*, *XD*, respectively, where *S, XSch, XD, R, R1* represent Subject, XML Schema object, XML document object and Rights required on XML Schema level and XML document, respectively. *ATT(S), ATT(XSch), ATT(XD)* represent attributes of subjects, XML Schema object and XML document object, respectively. *IM* represent Instance Mapping between SO and IO. *preA* and *preA1* are predicates about authorization functions.

1. *allowed (s, xsch, r)* $\Rightarrow$ *preA(ATT(s), ATT(xsch), r),*
where A $\Rightarrow$ B means B is a necessary condition for A. In this example this predicate indicates that if subject *s* is allowed to access XML Schema level *xsch* with right *r,* then the indicated condition preA must be true.

2. *allowed (s, xd, r1)* $\Rightarrow$
preA1(ATT(s), ATT(xd), r1),
In this example the allowed (s, xd, r1) predicate indicates that if subject *s* is allowed to access XML document *xd* with right *r1,* then the decision function preA1 must be true.

3. *IM*: $SO \rightarrow 2^{IO}$.
Where *IM* is a one-to-many mapping relationship from *SO* to *IO*. In our model, we use *IM* to implicitly specify the authorization in the instance level. Specifically, the attributes defined on schema objects will be transported to all its instance objects.

The *UCMpreA* model provides an authorization method on whether a subject can access the XML Schema and Instance level document. The *allowed(s, xsch, r)* predicate shows that subject *s* can access the XML Schema object. The *allowed(s, xd, r1)* predicate shows that subject *s* can access the XML document object. At this process, private information in XML Schema and corresponding XML documents are restricted.

B. Usage control for ongoing Authorizations Model (UCMonA)

A usage control model for ongoing-Authorizations model is used to check ongoing authorizations during access processes. In this model, usage requests are allowed without any 'pre' decision making.

The *UCMonA* model has the following components:
S, XSch, XD, R, R1, ATT(S), ATT(XSch), ATT(XD) and *IM* as before, and ongoing usage decision functions *onA* on *XSch* (XML Schema level) and *onA1* on *XD* (XML document). *onA* and *onA1* are used to check whether *S* can continue to access or not.

1. *allowed (s, xsch, r)* $\Rightarrow$ *true*,
This is a prerequisite for ongoing authorization on XML Schema object *xsch*.

2. *allowed (s, xd, r1)* $\Rightarrow$ *true*,
This is a prerequisite for ongoing authorization on XML document object *xd*.

3. *stopped (s, xsch, r)* $\Leftarrow \neg$ *onA(ATT(s), ATT(xsch), r)*,
The access of subject *s* to *xsch* is terminated if the ongoing authorization onA is failed.

4. *stopped (s, xd, r1)* $\Leftarrow$
$\neg$ *onA1(ATT(s), ATT(xd), r1),*
The access of subject *s* to *xd* is terminated if the ongoing authorization onA1 is failed.

5. *IM*: $SO \rightarrow 2^{IO}$.
The mapping relationship is the same as those in pre-Authorization model.

In this model usage decision Boolean functions are *onA*, *onA1* instead of *preA, preA1*. During this process the requested access is always allowed as there is no pre-authorization all the time. Both *allowed (s, xsch, r)* and *allowed (s, xd, r1)* are required to be *true*, otherwise ongoing authorization should not be initiated. Ongoing authorizations are active throughout the usage of the requested right, and some requirements are repeatedly checked for a continued access. These checks are performed periodically based on time or event. In the process when attributes are changed and requirements are no longer satisfied, *stopped* procedures are performed. *Stopped (s, xsch, r)* and *stopped (s, xd, r1)* indicate that rights *r* and *r1* of subject *s* on object XML Schemas and XML documents are revoked and the ongoing access terminated. For example, a limited number of simultaneous usage, suppose only two administration staff can access the information about the price of sold books in an object XML Schema level simultaneously. If a third administration staff requests access and pass the pre-authorization, the staff with the earlier time access is terminated. While this is a case of ongoing authorizations, it is important that the certificate should be evaluated in a *pre* decision.

Due to the length of the paper, other authorization models are omitted. In practice, the six models pre Authorizations, ongoing-Authorizations, pre-Obligations, ongoing-Obligations, pre-Conditions and ongoing-Conditions may need to be combined for an access control. We obtain an authorization method for XML Schemas and XML documents and their elements by checking users' (subjects') authorizations, obligations and conditions with continuity properties. The following algorithm in Table 3 is based on these models and introduces how to manage an XML document access control when a user (subject) applies to access an XML Schema with right *r* and an XML document with right *r1*. The output of an access control decision is required to satisfy some expected schemas. The function *im(target'.xsd)* is to check if output *target'.xml* can be validated by *target'.xsd*. Since the authorization process can remove some parts of the input object, the output may not satisfy some particular schemas, which are required by most applications. In this case, the access will be denied.

XML-based Algorithm:
Input: Access request: (*u, r, target.xml*)
Schema of target: *target.xsd*
Schema of expected output: *target'.xsd*
Output: *target'.xml*
Method:
// Verify UCMpreA:
1) **if** *preA(ATT(s), ATT(xsch), r)* $\cup$ *preA(ATT(s), ATT(xd), r1)* = false
// The process in pre-Authorization is not successful
2) ACCESS denied;
3**) endif**;
4) **if** target'.xml $\in$im(target'.xsd)
5) ACCESS target.xml is permitted;
6) Output target'.xml;
7) **else** ACCESS denied;
8) **endif**;

// Verify UCMonA:
9) **if** *preA(ATT(s), ATT(xsch), r)* $\cup$ *preA(ATT(s), ATT(xd), r1)* = false
// The process in pre-Authorization is failed, don't need further verification.
10) Application denied;
11) **endif**;
12) **if** *onA(ATT(s), ATT(xsch), r)* $\cup$ *onA(ATT(s), ATT(xd), r1)* = false
13) ACCESS stopped;
14) **if** target'.xml $\in$im(target'.xsd)
15) ACCESS target.xml is permitted;
16) Output target'.xml;
17) **else** ACCESS denied;
18) **endif**;

Table 3: Algorithm of XML Access Control

4. Related Work

Xinwen, Jaehong and Ravi [19] introduced an extended role based access control (RBAC) model for XML

security. In their model, permissions are defined based on XML Schema components and will be transported to all instances. The permission reuse through these hierarchies provides the security administration. Several constraints are presented in the model. The proposed model can be modularly deployed and flexibly administrated in distributed environments. The model can be readily applied to no-schema based XML instances and instance level authorizations. However, our work substantially differs from that proposal. The main differences in our approach are in the following aspects. Firstly, their protocol is based on RBAC and hence it focuses on permissions-role assignment, objects hierarchies and constrains. Our approach is based on usage control, we have analysed the characteristics of various access authorizations and presented detailed models for different kinds of authorizations. Secondly, their approach does not mention how to update users' permissions on XML object when their conditions or obligations have changed. It is an important state for XML documents in the Internet since users always alter their conditions or obligations. By contrast, users in our scheme have to pass pre-Authorizations and ongoing-Authorizations as well as pre-Obligations, pre-Conditions and ongoing-Obligations and ongoing-Conditions. This indicates that our method is much more powerful in dynamic environments.

Elisa and Elena [1] presented an access control system supporting selective distribution of XML documents among possible large user communities by using a range of key distribution methods. In their papers, a formal model of access control policies for XML documents is given. It focuses on key distribution methods to protect XML documents. The approach consists of encrypting different portions of the same document according to different encryption keys, and selectively distributing these keys to the various users. By contrast, our work provides a rich variety of options that can deal with XML documents. Users can access XML documents with their keys at any time, even when their properties are updated. In our scheme, users have to satisfy pre-Authorizations, pre-Obligations and pre-Conditions as well as ongoing-Authorizations, ongoing-Obligations, ongoing-Conditions.

5. Conclusions and future work

In this paper we introduce XML Schema, XML, usage control and discuss access models for XML Schemas and XML documents by using usage control. Comparing with XML DTD, XML Schema has richer use than XML DTD. Usage control models provide an approach for the next generation of access control. In usage control we analyse not only decision factors, such as authorizations, obligations and conditions, but also the continuity properties. This paper also illustrates two different kinds of models built for XML Schemas and XML documents. The methods presented in this paper can be used to control XML documents in a dynamic environment. It also begins a new application with usage control.

Obviously, access control in Web services will be popular in the future since all messages and protocols in Web services are in XML format. Only using traditional access control is not adequate for modern application needs. The Schema based usage control model can be a solution of the Web service security. This paper represents only a first step for XML Schema level authorization in XML documents with usage control. Much work is still to be done before these models can be used in practice.

References

[1] Bertion E. and Ferrari E. Secure and selective dis-semination of xml documents. *ACM trans. Inf. Syst. Secur.*, 5(3):290-331, 2002.

[2] Bertion E., Castano S., Ferrari E. and Mesiti E. Controlled access and dissemination of xml documents. In *Processings of the second international workshop on Web information and data management*, pages 22-27. ACM Press, 1999.

[3] Bray T., Paoli J., Sperberg M and Maler E. *Extensible Markup Language (XML) 1.1 (Second Edition)*. World Wide Web Consortium (W3C), Cambridge, MA, USA, 2000.

[4] Damiani E., Capitani S. and Samarati P. Towards securing xml web services. In *Proc .of the 2002 ACM Workshop on XML Security,* Washington, DC, USA, November 2002.

[6] Damiani E., Paraboschi S. and Samarati P. A fine-grained access control system for aml documents. *ACM Trans. Inf. Syst. Secur.,* 5(2):169-202, 2002.

[7] Gabillon A. An authorization model for xml databases. In *Proceedings of the 11th ACM conference on Computer Security,* 2004.

[8] Igor T., Zachary G.I., Alon Y.H and Daniel S.W. Updating xml. 2001.

[9] Kudo M. and Hada S. Xml access control, http://www.tri.ibm.com/projects/xml/xacl/xmlac-proposal.html

[10] Kuper G., Massacci F. and Rassadko N. Generalized xml security views. In *Proceedings of the 10th ACM sysposium on Access control models and technologies*, pages 77-84. ACM Press, 2005.

[11] Lim C. H., Park S. and son S.H. Acess control of xml documents considering update operations. *ACM Workshop on XML Security*, 2003.

[12] Murata M., Tozawa A., Kudo M. and Hada S. Xml access control using static analysis. *In ACM Transactions on Information and System Security*, volumn9, pages 292-324. ACM Press, August 2006.

[13] Park J. and Sandhu R. Towards usage control models: beyond traditional access control. In *Proceedings of the seventh ACM symposium on Access control models and technologies*, page 57-64. ACM Press, 2002.

[14] Park J., Sandhu R., and Schifalacqua J. Security architectures for controlled digital information dissemination. *In Proceedings of 16th Annual Computer Security Application Conference,* December 2003.
[15] Sandhu R. and Park J. Usage control: A vision for next generation access control. In *MMM-ACNS 2003,* pages 17-31, Springer-Verlag Berlin Heideberg, 2003.
[16] Sun L. and Li Y. DTD level authorization in xml documents with usage control. In *International Journal of Science Network Security*, volumn 6, pages 244-250, November 2006.
[17] Wang Y. and Tan K. A scalable xml access control system. In *Proceedings of the 10th international WWW conference*. Poster, 2001.
[18] Zhang X., Park J. and Parisi-Presicce F. A logical specification for usage control. In *SACMAT'4*. ACM Press, 2004.
[19] Zhang X., Park J. and Sandhu R. Schema based xml security: Rbac approach. In *Proceedings of the IFIP WG*. ACM Press, 2003.