Semantic-Aware Data Protection in Web Services

Csilla Farkas
Amit Jain
Duminda Wijesekera
Anoop Singhal
Bhavani Thuraisingham

1 Center for Information Assurance Engineering, Dept of Computer Science and Eng., University of South Carolina, Columbia, SC-29208,
2 Center for Secure Information Systems, Dept of Info. and Software Eng., George Mason University, Fairfax, VA 22030–4444,
3 National Institute of Standards and Technology,
4 University of Texas at Dallas
email:{farkas|jain}@cse.sc.edu,dwijesek@gmu.edu,psinghal@nist.gov, bhavani.thuraisingham@utdallas.edu

Abstract

This paper presents a method to remove the dependency of XML access control models on the syntactic representation of the XML trees. We propose a semantics-based approach, expressing XML access control on ontologies that describe the XML documents. The semantics-based model is transformed to the syntactic representation on the actual XML instance. Our method supports the uniform enforcement of an authentication model on syntactically different but semantically equivalent XML documents.

Contact Author: Csilla Farkas, Dept. of Computer Science and Engineering, University of South Carolina, Columbia, SC 29208.
Telephone: 803-576-5762, Fax: 803-777-3767, email:farkas@cse.sc.edu
1 Introduction

Current trends of Web applications indicate that Web Services will become a fundamental technology for Web-based applications. Although Web Services (WS) are rapidly developing with a set of standards, like WSDL [3], UDDI [4], SOAP [7], and WS-Policy, current WS capabilities are limited to express application specific semantics. Further, a crucial aspect of WS is to achieve high-assurance security. Several standards, like SOAP message security [15], SAML, XACML have been developed along this goal. Specifications like [6, 14] are some other initiatives taken by organizations like IBM, Verisign, Microsoft to develop policy languages for WS security. XML is a fundamental technology used across the various components of WS, conveying application specific semantics. Several XML access control models have been developed [5], [2], [13] to provide controlled accesses to XML formatted data. However, XML access control models are based on XML syntactic constructs without incorporating data or application semantics.

Based on the use of XML to convey application specific semantics, we propose an approach to secure XML documents based on these semantics. Access control requirements are defined on ontologies, then propagated to the XML documents. Our approach supports not only semantics-aware access control but also protects against security violation via XML rewrite attacks. The syntax-based approach provides limited support for fully machine understandable processing and even simple manipulations of an XML file can bypass the security requirements. For example, the same XML file might be restructured when passed between two WSs. In the example shown in Figure 1, the data originating from a patient may be presented in a different syntactic form when it is shared between the two Web Services. The XML structure and syntax-based access control policy over the original data may not be correctly represented on the modified XML files.

![Figure 1: Web Service Data Sharing scenario](image-url)
One of the features of XML is that several XML trees can be developed to represent the same original data. For example changing the element nesting, cardinality constraints, or replacing the element names with abbreviated text or synonyms are user defined and different preferences lead to different XML syntax 1. Similarly a data item in XML can be designed either in the element or attribute form [16], based on the application requirements. Authorizations defined for an XML document would not work if the structure were changed syntactically even if it would correspond to the same original data. To ensure correct enforcement of the XML access control, it is necessary to provide transformation of policies over different syntactic forms. Performing these transformations by human expert is time consuming and may lead to errors in the case of complex policies. Automated tools that are capable of identifying restructured data and verify the satisfaction of original access control requirement on the restructured data are needed.

Our approach is based on the data semantics as defined by the mappings from XML documents to ontologies. None of the existing XML access control models utilize these mappings to express access control requirements in a syntax independent manner. Even simple modification of data structure would allow a malicious (or careless) user to violate the original access control requirements. Syntactic variations make access control in one XML file unusable or incorrect in a different XML structure over the same data. In [17] Qin and Atluri propose a concept-level access control for Web data, where access control is defined on ontological concepts and instances of these concepts inherit the access control of the concepts they belong to. They incorporate relationships supported by the ontology. However, they do not define how the instances are associated with the ontological concepts and how to enforce the access control requirements on these concepts.

In this paper we provide a framework to enable semantics-based access control model using the existing XML access control technology. The basis of our work is to ”enforce” uniform policy requirements on semantically equivalent but syntactically different XML trees. Semantic units are identified by the mappings from the XML trees to a common ontology.

Our approach is the generalization of the works of Kodali et al. [10], [8], [9], [11], [12] on SMIL formatted multimedia security. They propose the concept of SMIL Normal Form for the representation of SMIL equivalence classes. These classes constitute syntactically different SMIL fragments with the same operational semantics.

1Note that under XML syntax we consider syntax of the element and attribute names, and the structure of the XML tree
We incorporate data and application specific semantics, represented as an ontology that is associated with the XML tree, to define access control requirements. Consider patient information given in both XML documents in Figure 2 and Figure 3 from the previous example in Figure 1. Both trees were created from the same original data. Since XML access control policies are expressed on the XML tree syntax, different policies need to be developed for the two different representations. Further, no verification can be performed that semantically equivalent components have the same access control requirements.

For example, assume that the access control policy requires that a subject John is permitted to read the SSN of the patients. Expressing this requirement would use different XPath expressions in the two XML documents, because in XML format 1, SSN is represented as an attribute of the tag patients, while in XML format 2, it is represented as sub-element of the tag patient. Different examples are given in Figures 4 and 5. These trees have the same information data but they differ in their syntax. Similar to the previous example, two different access control policies need to be developed to secure the two XML trees.

We define access control policies on semantic data units, and develop automated transformation of these policies to the different syntactic forms.

The organization of the paper is as follows. The section 2 explains syntactic manipulation of XML trees and their mappings to ontologies. Section 3 describes the ontology-based access control framework for XML documents. We conclude in Section 4 and recommend future work direction.
2 Syntactic Manipulation of XML trees

In this section we provide definition of ontologies, XML trees, and mapping between them. We adopt the formalism of An et al. [1] defining the mappings between XML documents (schemas) and corresponding ontologies. We also propose an XML tree format, called element-only XML tree, that is equivalent to the general XML tree but will simplify our arguments. We provide transformation from an arbitrary XML document to its element only form. The aim of this transformation is to be able to define syntactic constructs that belong to the same equivalence class in a uniform manner. First, we define our XML trees.

Definition 2.1 (XML document tree) An XML document consisting of elements and attributes is defined recursively as an ordered labeled-tree as follows:

1. The Empty set \( \{ \} \) is a tree, called empty-tree.

2. A single node \( \{ n \} \) is a tree where \( n \) is a node or an attribute label, or an element or attribute value.

3. If \( t_1, t_2, \ldots, t_k \) are trees then \( \{ n \rightarrow \{ t_1, t_2, \ldots, t_k \} \} \) is a tree, where \( n \) is the label of the root node and has outgoing edges to subtrees \( t_1, t_2, \ldots, t_k \).

In general XML trees nodes may correspond to elements or attributes. For uniform representation, we propose that all attributes \( (a_1, \ldots, a_k) \) of an element \( n \) are treated as subelements of \( n \), with the cardinality restriction, that \( a_i \ (i = 1, \ldots, k) \) occurs 0 or 1 time. The transformation of a regular XML document to its element-only form is trivial and we omit it. Figure 6 shows the Element Only Form of the XML files that were given in Figures 2 and 3.

Ontologies provide the semantics XML documents convey. Our access control is based on this meaning to define the security requirements. For
this, we define a mapping from the XML trees to their corresponding ontologies. Since these mappings are based on the data and application semantics, rather than the original XML syntax, semantically equivalent XML documents will map to the same ontological components.

An ontology is a 6-tuple \((C, P, C_H, P_H, \text{dom}, \text{range})\) where \(C\) is a set of class names, \(P\) is a set of property names (including data type properties), \(C_H\) is the class hierarchy, \(P_H\) is the property hierarchy, \(\text{dom}\) is the domain of a property in \(P\), and \(\text{range}\) is the range of each property in \(P\).

We represent ontologies as directed, node and edge-labeled graphs. Each node corresponds to a class or a data type, and each edge corresponds to a property or the subclass hierarchy. We say that the domain of a property \(p\) is \(\text{dom}(p) = \bigcup c_i\) for all \(c_i\) such that 1) there is an edge labeled \(p\) in the ontology originating from \(c_i\) or 2) there is an edge originating from \(c_i\) and labeled \(p\) in the ontology, such that \(c_i \leq c'_i\). For each class \(c\) in \(O\) the properties of \(c\), denoted as \(p(c)\), are the union of all properties \(p\) such that
\[ c \in \text{dom}(p) \text{ or } c' \in \text{dom}(p), \text{ where } c \leq_{Ch} c'. \] Similarly, we say that the range of a property \( p \) is \( \text{range}(p) = \bigcup c_j \) for all \( c_j \) such that 1) there is an edge labeled \( p \) in the ontology pointing to \( c_j \) or 2) there is an edge pointing to \( c' \) and labeled \( p \) in the ontology, such that \( c_i \leq_{Ch} c' \). For simplicity, each graph is converted into directed, edge and node labeled tree [1].

**Definition 2.2 (Ontology)** Let \( C \) denote the set of classes, \( P \) the properties, \( \tau \) the data types, and \( X \) a set of variables. An ontology \( O \) is represented as the conjunction of literals, where literals are defined as follow:

1. \( c(x) \), where \( c \in C \) and \( x \in X \)
2. \( p(c_i, c_j) \), where \( p \in P \) and \( c_i, c_j \) are either classes in \( O \) or variables
3. data types \( f \) are represented as “artificial” nodes with the special property \( f \) labeling the edge to the node
4. \( \text{dom}(p) = \bigcup c_i | \exists p(c_i, c_j) \)
5. \( \text{range}(p) = \bigcup c_j | \exists p(c_i, c_j) \), where \( c_j \) may be a literal

To incorporate data semantics in our XML access control model, we need to define a mapping \( \nu \), from XML trees to partial ontologies. \( \nu \) maps each node of the XML tree to a class, a data type, or an edge of the ontology. Edges of the XML are mapped to paths of the ontology. We use this mapping to assign access control requirements to the syntactic XML trees, based on the access control specified over the corresponding ontology.

**Definition 2.3 (Mapping Correspondence)** Let \( \nu \) be a mapping from and XML tree \( T \) to ontology \( O \). Then, the correspondence between the elements and edges of \( T \) and the elements and edges of \( O \) is defined as

1. \( T.e \rightarrow_{\text{corr}} O.c: \) an element \( e \) of \( T \), identified by \( T.e \), where \( T \) is a path expression that leads to \( e \), corresponds to a class \( c \) in \( O \).
2. \( T.E \rightarrow_{\text{corr}} O.p(c, c_j): \) an element \( e \) of \( T \), identified by \( T.e \), where \( T \) is a path expression that leads to \( e \), corresponds to an edge labeled \( p \), originating from \( c_i \) to \( c_j \) in \( O \)
3. \((T.e_1,T.e_2) \rightarrow_{\text{corr}} \{O.p(c_1,c_2) \rightarrow \ldots \rightarrow O.p(c_n,c_{n+1})\}: \) an edge in \( T \) between \( T.e_1 \) and \( T.e_2 \), identified as a pair of nodes \((T.e_1,T.e_2)\), corresponds to a path originating at \( O.c_1 \) and terminating at \( O.c_{n+1} \) in \( O \).
We require that all nodes of an XML tree are mapped to either nodes or edges of the ontology, however, the ontology may have nodes, edges that do not participate in the mapping.

The mapping from an XML subtree maps to the classes and properties of the ontology, such that each element name is mapped to either a class name or to a property name. The access restrictions for a node in the XML tree are based on the access restrictions on the corresponding ontology components.

![XML to Ontology Mapping](image)

**Figure 7: XML to Ontology Mapping**

XML documents with different syntax and semantics may be mapped to the same partial ontology (see Figure 7).
3 Semantics-Based XML Access Control Architecture

We use the mapping between XML subtrees and partial ontologies to define access control for an XML document. Our approach allows the use of existing access control mechanisms, like [2], while supporting access control specification based on data and application semantics.

Figure 8 shows the proposed architecture to assign access control requirements to XML documents. The access control module accepts an XML document, an ontology and its security policy, and the mapping from the XML document to the ontology as the input. A security policy, applicable to the XML document is constructed using the mapping and the security policy for the ontology. Additional input, containing propagation policy, conflict resolution and XML restructuring requirements, may be used to aid complex security policies.

For the purpose of this paper, we show how our framework handles Discretionary Access Control. More advanced access control models can be similarly expressed by our model.

Definition 3.1 (Access Control Policy over Ontology) Access control is defined as a set of triples of the form $(s, o, \langle \text{sign} \rangle a)$, where $s$ is the subject, $o$ is the object, and $\langle \text{sign} \rangle a$ is a signed action type.

Definition 3.2 (Security Object) Let $O$ be an ontology. A security object is either a class $c \in C$ or a property $p$ of a class $c$.

Definition 3.3 (Security Assignment) Let $Pol$ define the security policy for an ontology $O$, and $\nu$ be the mapping from an XML tree $T$ to $O$. For all nodes $T.n$, if $\nu(T.n) = O.o$, where $O.o$ is a security object in $O$, then create a triple $(s, T.n, \langle \text{sign} \rangle a)$ for all triples $(s, O.o, \langle \text{sign} \rangle a) \in Pol$. 

Figure 8: Access Control Mapping Architecture
Clearly, because there is an association between all components of an XML and some of the components of an ontology, there is an access control rule for each XML component. However, the access control requirements for the components may not follow current XML access control requirements, like increasing restrictions traversing down the XML tree. Stoica et al. addressed this problem in [18].

4 Conclusion and Future Work

In this paper we presented a syntax independent method to define access control needs of XML trees. Our aim is to enforce the same authorization permissions for syntactic variations of the same XML document and to incorporate data and application specific semantics in XML access control. Our approach is based on defining access control on ontologies data and application semantics. Mappings from the XML trees to these ontologies enable to enforce access control on the syntactic representation. Our current approach aims to be built on top of XML access control mechanisms and to support presentation independence of XML developers.

Our future work includes formal definition of semantically equivalent XML documents based on specific applications, enhanced conflict resolution, policy verification, and addressing ontological inferences.

References


