Controlled Access and Dissemination of XML Documents*

Elisa Bertino *Silvana Castano* *Elena Ferrari*
Dip. di Scienze dell'Informazione
Università degli Studi di Milano
Via Comelico, 39/41 20135 Milano, Italy
{bertino,castano,ferrarie}@dsi.unimi.it

Marco Mesiti†
Dip. di Informatica e Scienze dell'Informazione
Università degli Studi di Genova
Via Dodecaneso, 35 16146 Genova, Italy
mesiti@disi.unige.it

Abstract

XML (*eXtensible Markup Language*) is becoming the most relevant standardization effort in the area of document representation through markup languages. Through XML, it is possible to define complex documents, containing information at different degrees of sensitivity. Moreover, the processes of document exchange and acquisition, which can be very frequent in Web-based information systems, are simplified and standardized. In this scenario, there is a strong need for policies to control and regulate the access and dissemination of XML documents. In the paper, we discuss main protection requirements posed by XML documents and we present a set of authorization and dissemination policies that enable both a controlled access to XML documents in a given source and the exchange of XML documents across different sources.

1 Introduction

XML (*eXtensible Markup Language*) [11] has recently emerged as the most relevant standardization effort in the area of document representation through markup languages in Web-based information systems. Main features of XML are related to the use of tags, for defining nested document structures, and to the introduction of document types (called DTDs -*Document Type Definitions*-), for describing the structure of documents in a given source. In this way, it is possible to define complex documents, containing information at different degrees of sensitivity. Moreover, document exchange and acquisition, which can be very frequent in Web-based systems, are simplified and standardized.

In this scenario, there is a strong need for policies ensuring a controlled access and exchange of XML documents. As for the document access, authorization policies are required to regulate access requests coming from different subjects to XML documents stored in a given source. As for document exchange, dissemination policies are required to regulate how to release documents stored in a source to different users. Moreover, for the application of both policies, the acquisition of a new document in a source has to be properly managed. The development of suitable authorization and dissemination policies for XML documents poses new protection requirements with respect to conventional object-oriented databases [5, 9] or HTML documents [10]. These new requirements arise because of the richer structure of XML documents with respect to HTML documents and the possibility of attaching a DTD to XML documents, which has similarity with the notion of type in the object-oriented context.

In the paper, we discuss XML protection requirements as well as a set of authorization and dissemination policies that cope with these requirements. The proposed policies support varying protection granularity levels and a view-based release of XML documents, to differentiate the access and dissemination based on user characteristics and needs. In this way, the security administrator can choose the most appropriate policy for a given document or a set of documents to be accessed or exchanged over the Web.

The remainder of the paper is organized as follows. Section 2 presents the basic concepts of XML and the new protection requirements for XML documents. Section 3 describes the authorization policies for XML document access, whereas Section 4 presents the dissemination policies for document exchange. Section 5 discusses the classification of new documents entering a

*The work reported in this paper has been partially supported by the Italian MURST under the Interdata project.

†Part of this work was carried out while on leave at Telcordia Technologies (formely BellCoRe) - Morristown (NJ), U.S.A.

```
<employee id="E123">
 <Ename name-ref="N-E123"/>
 <Eaddress address-ref="Ad-E123"/>
 <salary salary-refs="http://...salary.xml" employee="E123">
  level 2
 </salary>
 <resume>
  <name id="N-E123">
   <fname> Dario </fname> <lname> Bhor </lname>
  </name>
  <address id="Ad-E123">
   <street> 35, Lake Street, Morristown </street>
   <tel> 973 999 111 </tel> <tel> 973 111 999 </tel>
   <email mailto="dario@swan.com"> dario@swan.com </email>
  </address>
  <education> ...  </education>
 </resume>
 <medical-dossier medical-ref="http://../medical/bohr.xml"/>
</employee>
```

Figure 1: An example of XML Document

```
<!DOCTYPE employee[
<!ELEMENT employee(Ename,Eaddress,salary,resume,medical-dossier)>
<!ELEMENT Ename EMPTY>
<!ELEMENT Eaddress EMPTY>
<!ELEMENT salary (#PCDATA)>
<!ELEMENT resume (name,address,education,hobby?)>
<!ELEMENT medical-dossier EMPTY>
<!ELEMENT name (fname,lname)>
<!ELEMENT address (street tel*,email)>
<!ELEMENT street (#PCDATA)>
<!ELEMENT tel (#PCDATA)>
<!ELEMENT email (#PCDATA)>
<!ELEMENT education (#PCDATA)>
<!ELEMENT hobby (#PCDATA)>
<!ATTLIST employee id ID (#REQUIRED)>
<!ATTLIST name id ID (#REQUIRED)>
<!ATTLIST address id ID (#REQUIRED)>
<!ATTLIST email mailto CDATA>
<!ATTLIST salary salary-refs CDATA employee CDATA>
<!ATTLIST Ename name-ref IDREF>
<!ATTLIST Eaddress address-ref IDREF> ]>
```

Figure 2: An example of DTD

source for the application of the proposed policies. Finally, Section 6 concludes the paper and outlines future research.

2 Basic concepts and protection requirements of XML documents

Access and dissemination of XML documents pose new protection requirements which have to be taken into account when stating the authorization and dissemination policies for them. In the remainder of this section, we first survey XML concepts that are relevant for the subsequent discussion, and then we discuss those requirements.

2.1 Basic Concepts of XML

The basic concept of an XML document is the *element*. Elements can be nested at any depth and can contain other elements (*subelements*). An element contains a portion of the document delimited by two *tags*: the *start tag* at the beginning of the element, with the form <*tag-name*>, and the *end tag* at the end of the element, with the form </*tag-name*>, where *tag-name* indicates the type of the element (*markup*).

An example of XML document containing employee information is shown in Figure 1. For an employee, the document provides information on first name, last name, address, resume, education, and pointers to other XML documents containing the medical dossier and salary. In the figure, the `employee` element is an example of *document element*, that is, the outermost element containing all the elements of the document. `Resume`, `tel`, `education` are examples of elements at different depth in the hierarchical structure of the XML document. The `address` element is an example of element with subelements in that it contains `street`, `tel`, `email` elements. `street` is an example of element containing text (*data content*). `Ename` and `Eaddress` are examples of *empty elements*, that is, elements characterized only by a start tag and without data content and subelements.

The start tag of each element can specify a list of *attributes* providing additional information for the element (e.g., the start tag of the `salary` element). Attributes are of the form *name* = *attvalue*, where *name* is a label and *attvalue* is a string delimited by quotes. Attributes can have different types allowing one to specify the element identifier (attributes of type `ID` often called *id*), links to other elements of the document (attributes of type `IDREF` referring to a single target or `IDREFS` referring to multiple targets), or additional information about the element (e.g., attributes of type `CDATA` containing textual information).

A *document type declaration* can be attached to XML documents, specifying the *Document Type Definition* (DTD), that is, the set of rules the XML documents may follow. The document type declaration is composed of two parts: the *element declarations* and the *attribute list declarations*. The element declarations specify the structure of the elements contained in the document and their type. The attribute list declarations specify, for each element, the list of its attribute names, types and (possibly optional) default values. Figure 2 shows the DTD for employee documents of the source in Figure 1.

2.2 Protection requirements for XML documents

Main protection requirements for XML documents that influence the definition of the policies for their access and exchange are related to the following characteristics of XML and subjects accessing them.

Presence of valid and well-formed documents. Two kinds of documents can be found in an XML source, namely, *valid* and *well-formed* documents. A well-formed document follows the grammar rules of XML [11]. A valid document is a document conforming to a given

DTD. Therefore, valid documents can be considered instances of a corresponding DTD in the source. For example, the document in Figure 1 is a valid document, since it conforms to the DTD in Figure 2. The presence of DTDs in a source can be exploited in the definition of authorization and dissemination policies for valid documents. It should be possible to specify a policy at the DTD level, which applies to all valid documents that are instances of the DTD. In this way, the definition of policies for valid documents exploits a notion of schema or type, in analogy with conventional policies for relational and object-oriented databases.

The authorization and dissemination policies for well-formed documents must take into account the fact that a DTD is not available. It should be possible to define policies based on the classification of a well-formed document, by finding the "best matching" DTD in the source, capable of describing the structure of the well-formed document to be protected. If such a DTD is found, the protection of the well-formed document can be based on the policies defined for the selected DTD. Alternatively, explicit policies can be defined for well-formed documents, by treating each document separately.

Hierarchical, inter-linked structure of documents. An XML document is an element composed of attributes and other elements. Elements of a document can contain elements (subelements) in turn, originating a hierarchical structure. Moreover, XML documents are inter-linked since their attributes can be links to other related documents/elements in the source. The same considerations hold for DTDs. Very often, subelements of the same document have varying protection requirements. For example, information on `name` subelement of the employee document in Figure 1 could be made available to everyone, whereas information on the `address` subelement should be distributed only to selected subjects. As another example, the link from an employee document to its related medical dossier could be kept hidden from most subjects and made visible only to a restricted number of authorized subjects. This hierarchical, inter-linked document structure requires the definition of policies enforcing a protection granularity finer than the whole document, to ensure a selective and appropriate protection of the contents of a document, both in the access and dissemination process. Moreover, semantic relationships between various elements at different granularity levels in this structure suggest the definition of policies based on the notion of *propagation.* The notion of propagation introduces a "by default" principle in applying the policies to documents. According to the propagation principle, policies specified for a protection object at a given granularity level (e.g., a document or a DTD) propagate to all *protection objects*[1] that are semantically related to it according to a certain relationship in the structure (e.g., the set of its subelements or the set of its valid instances).

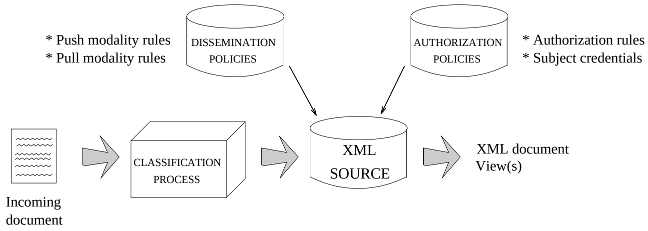

Figure 3: Controlled access and exchange for XML sources

Heterogeneity of subjects. The population accessing XML document sources is generally composed of heterogeneous subjects, characterized by different skills and needs. Moreover, the population is dynamic, in that the number and type of subjects is not known a priori and can very frequently change over time. This requires authorization and dissemination policies based on user characteristics and qualifications (in the following called *credentials*), rather than in terms of very specific and individual characteristics (e.g., user IDs).

In the following, we define authorization and dissemination policies that take into account the requirements outlined above. Figure 3 shows how the controlled access and exchange of XML documents takes place in presence of authorization and dissemination policies. In particular, authorization policies state how to control access requests coming from different subjects to XML documents stored in a given source, while dissemination policies state how to release documents from a source, given the authorization policy. Orthogonally to the kind of policy, a classification process can be required for new documents entering a source, in order to determine the most appropriate policies to apply to them.

3 Authorization policies for XML documents

Based on the requirements previously outlined, authorization policies must cope with a dynamic subject population, often making accesses from remote locations, and they must support a wide spectrum of protection granularities, ranging from a set of documents to specific elements within a document. Consequently, we base authorization policies on the notion of subject credentials and on authorization rules at different granularity levels, with different propagation options.

Subject credentials.
To deal with XML documents accessed by a dynamic

[1] In the following, the term protection object denotes the document(s) or the parts of document(s) to which a policy refers.

subject population, we support the specification of authorization policies in terms of *subject credentials* [1]. A credential is a set of attributes concerning a subject that are relevant for security purpose (e.g., age, nationality, position within the organization). Credentials with similar strucure are grouped into credential types (e.g., manager, programmer, employee).

The introduction of credentials allows one to directly express XML authorization policies in terms that are closer to the organizational structure of the enterprise. By using credentials one can simply formulate policies such as "*Only programmers that are permanent staff can access documents related to the internals of the system*", or "*A class of documents can be accessed only by users that are more than 18 years old*". Credentials are associated with each subject. Such credentials are then used to determine the valid authorizations of that subject based on *credential expressions* [1]. Credential expressions can specify both simple conditions on credentials and credential attributes, and composite conditions (obtained as a boolean combination of simple conditions). For instance, $Programmer(X)$ and $X.nationality = Italian$ are examples of simple credential expressions, denoting, respectively, programmers and subjects with an italian nationality. By contrast, $Manager(X) \wedge X.department = D_1$ is an example of composite credential expression denoting all the managers that work in department D_1. Credential expressions can be specified by means of a formal language like the one introduced in [1]. However, we plan to develop an XML-based language for specifying credentials, credential types, and credential expressions.

Authorization rules.
Authorization policies are implemented by a set of *authorization rules* establishing, for each subject, the accesses he/she can exercise on the documents in the source. Based on the previous discussion, authorization rules for XML documents have the following format:

$<$ `subj-spec, protection-objs, priv, prop-opt` $>$

`subj-spec` denotes the subjects to which the authorization applies. It can be either a credential expression or a subject identifier.

The component `protection-objs` denotes the document(s) (or parts of document(s)) to which the authorization rule applies. To cope with the fine-level granularity requirements, through `protection-objs` we want to identify a wide range of object, ranging from sets of XML documents to a specific portion of a document (like - for instance - a specific element or a specific attribute and/or link within an element). Thus, the specification of the protection objects in an authorization rule is realized in three steps. First, we identify a document (or a set of documents) to which the authorization rule applies. Documents can be specified at two different level of abstraction: *1)* at the DTD level, by specifying the identifier of a DTD (in such case the authorization applies by default to all the DTD instances) *2)* at the document (i.e., instance) level, by specifying the identifier of the document to which the authorization applies. Two special symbols are used to identify all the documents (* symbol) and all the DTDs (# symbol). Once the documents to which an authorization rule applies have been specified, we denote selected elements within the specified documents using their identifiers, or specifying paths from the root element, or using the symbol * to identify all the elements in the documents. Finally, the element components (i.e., links and/or attributes) are identified using their names. For example, with reference to Figure 1, `*.resume.address.city`, `#.resume.address.*`, and {`N-E123, Ad-E123`} are protection objects that can appear in an authorization rule.

The `priv` component of an authorization rule denotes the access modes that can be exercised on the protection objects specified by the authorization rule. We support two different kinds of privileges: *browsing* and *authoring* privileges. The two kinds of privileges have the same level of importance, thus no one subsumes the other. Browsing privileges allow subjects to read the information in an element (`read` privilege) or to navigate through its links (`navigate` privilege). Authoring privileges allow subjects to modify (or delete) the content of an element (`write` privilege) or to append new information in an element (`append` privilege). We assume that the `write` privilege subsumes the `append` privilege and also that if a subject has the `write` privilege on an element, then he/she can also delete the element.

Finally, the `prop-opt` component allows one to specify how authorizations specified at a given level propagate to lower level elements. The following options are provided: *i)* CASCADE: the authorization propagates to all the direct and indirect subelements of the element(s) specified in the authorization rule; *2)* FIRST_LEV: the authorization propagates to all the direct subelements of the element(s) specified in the authorization rule; *3)* NO_PROP: no propagation is performed.

Note that, the above three options allow us to concisely support a wide range of authorization policies. For instance, if all the information in a document (or in a set of documents) have the same protection requirements, then the CASCADE option can be used to concisely express all required authorizations. By contrast, the NO_PROP and the FIRST_LEV options can be used when different portions of a document need different authorization policies.

Example 1 *Authorization (*Secretary_Staff(X), `E123, read, cascade`*) allows the subjects with a credential of type* Secretary_Staff *to read all the attributes, links, and* *elements contained in the document identified by* E123. *However, they cannot navigate through the links, since in our model the read privilege does not imply the navigate one. By contrast, authorization (*Bob, E123, navigate, first_lev*) allows* Bob *to navigate through the* medical-ref, salary-refs, name-ref, *and* address-ref *links contained in the direct subelements of* E123. *Finally, Authorization (*Administrative_Staff(X), #.resume, write, cascade*), defined at DTD level, allows the subjects with a credential of type* Administrative_Staff *to write all subelements of the* resume *element of any employee document instance in the source.*

4 Dissemination policies for XML documents

Dissemination policies state how to release documents from a source. Release of XML documents can be done under two different modalities, called *information pull* and *information push* (see Figure 3).

Information pull.
Under this modality, the documents reside at one or more servers and the subjects ask them when needed, by issuing a proper access request. Access requests can refer to (a specific portion of) a document or to all instances (or portions of them) of a given DTD. When an access request is submitted, the access control mechanism checks which authorizations the subject issuing the request has on the requested document(s). Such authorizations can be either explicitly or implicitly given by the propagation modalities specified by the authorization policy. Based on such authorizations, the subject is returned a *view* of requested protection objects that contains only those portions for which he/she has a corresponding authorization. In the case of totally authorized requests, the view coincides with the whole protection objects. When, no authorizations are found, the access is denied.

Information push.
Under this modality, a server periodically broadcasts documents (or portion of them) to its clients (e.g., the case of a newsletter sent once a week to all users). Even in this case, different subjects may have privileges to see different, selected portions of the same document (or set of documents). Supporting an information push modality may entail generating different physical views of the same document (or sets of documents) and sending them to the proper subjects. The number of such views may become rather large and thus such an approach cannot be practically applied. Moreover, it is not possible to use a simple broadcast approach, since one has to send different copies to different subjects. To address this problem we plan to apply to XML documents an approach similar to the Cryptolope approach proposed in [6]. Under such an approach, the same (encrypted) copy of the document is sent to all the subjects. The idea basically consists of using different encryption keys for encrypting different portions of the same document. Each subject only receives the key(s) for the portion(s) he/she is enabled to access.

5 Classification of new documents entering a source

An important issue for the application of the authorization and dissemination policies in a given source is how to handle a new document entering the source. Because it is most likely the case that authorization/dissemination policies are specified in terms of document types, it is important to discover whether the document conforms to an existing DTD, and, consequently, if the document can be covered by the policies defined for this DTD. For this purpose, we propose to apply a classification process to new documents entering a source.

An XML document $\bar{d}$ entering a source undergoes a classification process to determine, among the available DTDs in the source, the one that best conforms to the new document (cfr. Figure 3). Such a process requires the analysis of the structure of the well-formed document and its comparison with the structure of existing DTDs in the source. Different algorithms have been proposed for the analysis of graph-based document structures, for their classification against a type hierarchy [3], and for the extraction of their representative structure [8]. The classification process returns one of the DTDs in the source with respect to which $\bar{d}$ conforms with a certain degree. For the definition of the policies for $\bar{d}$, we propose to proceed on the basis of the level of conformance between $\bar{d}$ and the selected DTD. In the following we present the different cases.

Conformance.
As the result of the classification process, a DTD is found that conforms $\bar{d}$, that is, for each element/attribute of $\bar{d}$ one corresponding matching element/attribute is found in the DTD. In this case, $\bar{d}$ can be classified as a valid instance of the selected DTD. Consequently, the policies specified for the DTD and/or its matching elements propagates to $\bar{d}$.

Partial conformance.
As the result of the classification process, a DTD is found that partially conforms $\bar{d}$, that is, only for some elements/attributes of $\bar{d}$ a corresponding matching element/attribute is found in the selected DTD. In this case, the policies defined for the DTD and/or its matching components propagate to $\bar{d}$ as in the previous case, while for each remaining non-matching element/attribute c of $\bar{d}$, different alternatives are possible:

i) Propagation-based policy: if c is a subelement of an element e for which a policy is defined in the DTD, then the policy for e propagates to c. For example, suppose that a new document $\bar{d}$ has a partial conformance with respect to the DTD in Figure 2, since the `resume` element of $\bar{d}$ contains also the `working-experience` subelement. If a subject can read the `resume` element of $\bar{d}$, then, according to the propagation-based policy, the `read` privilege can be propagated also to the `working-experience` subelement of $\bar{d}$.

ii) Affinity-based policy: an element/attribute c' in the DTD having affinity with c is exploited to derive the policy for c. The affinity criterion captures the fact that the labels of c and c' denote semantically related information, based on semantic content of the labels and on a domain ontology [4]. For example, suppose that a `medical-record` element is contained in the `resume` element of $\bar{d}$. Then, $\bar{d}$ has only a partial conformance with the DTD in Figure 2. According to the affinity-based policy, the same authorization policy defined for the `medical-dossier` element in the DTD is propagated to the `medical-record` element, since the term *medical-record* is synonym of the term *medical-dossier* in the Wordnet [7] ontology.

iii) Explicit policy: an explicit policy is defined from scratch for c. This alternative is always applicable, and is mandatory when no other alternative can be applied. For example, suppose that $\bar{d}$ has a partial conformance with the DTD presented in Figure 2 because the `employee` element contains also the `evaluation` subelement. For `evaluation`, neither the propagation-based nor the affinity-based policy can be applied. Consequently, explicit authorization rules have to be defined by the security administrator for `evaluation`.

The above three alternatives offer a different degree of automation and support to the security administrator in the definition of the authorization/dissemination policies for partially conforming documents. In particular, the first two alternatives exploit the propagation and affinity notions to limit the manual activity of the security administrator in the definition of the policies for a new component c, while the third one requires the manual definition of an "ad hoc" policy. Alternatives i) and ii) need suitable mechanisms to automatically derive all required authorization rules for all the involved subjects such as, for example, rule-based mechanisms [2]. In both i) and ii), the security administrator can interactively validate derived authorization rules for a new component c to check their suitability.

No conformance.
When no conforming DTD is found, the authorization/dissemination policy is explicitly specified for the new document. This approach can also be used when classification is not performed, or also when the new document has peculiar protection requirements, in addition to the ones of the DTD to which it conforms.

6 Concluding Remarks

In this paper, we have outlined a set of authorization and dissemination policies for a controlled XML document access and exchange. Novel features of the proposed policies are related to the capability of dealing with the inter-linked, hierarchical structure of XML documents, with documents that partially conform to the existing policies, and with a dynamic subject population to which different views of the same (set of) document(s) have to be released. Future research work includes the enrichment of the set of policies we have proposed (for example, by supporting policies which apply only to specific sets or collections of documents belonging to a given DTD), the development of a formal notation to specify the proposed policies, the implementation of access control and dissemination mechanisms enforcing the proposed policies, the development of tools supporting the security administrator. Other important research directions we plan to investigate are related to the dissemination of XML documents in the WWW environment. An important issue is how to support the Cryptolope approach with standard Web browsers.

References

[1] N. Adam, V. Atluri, E. Bertino, and E. Ferrari. A Content-based Authorization Model for Digital Libraries. Submitted for pubblication.

[2] E. Bertino, C. Bettini, E. Ferrari, and P. Samarati. A Temporal Access Control Mechanism for Database Systems. *IEEE Transactions on Knowledge and Data Engineering*, 8(1):67–80, February 1996.

[3] E. Bertino, G. Guerrini, I. Merlo, and M. Mesiti. An Approach to Classify Semi-Structured Objects. In *Proc. Thirteenth European Conference on Object-Oriented Programming*, number 1628 in Lecture Notes in Computer Science, pages 416–440, 1999.

[4] S. Castano and V. De Antonellis. A Discovery-Based Approach to Database Ontology Design. *Distributed and Parallel Databases - Special Issue on Ontologies and Databases*, 7(1), 1999.

[5] E. Fernandez, E. Gudes, and H. Song. A Model for Evaluation and Administration of Security in Object-Oriented Databases. *IEEE Transactions on Knowledge and Data Engineering*, 6:275–292, April 1994.

[6] H. Gladney and J. Lotspiech. Safeguarding Digital Library Contents and Users: Assuring Convenient Security and Data Quality. *D-lib magazine*, May 1997.

[7] A. Miller. WordNet: A Lexical Database for English. *Communications of the ACM*, 38(11):39–41, November 1995.

[8] S. Nestorov, S. Abiteboul, and R. Motwani. Extracting Schema from Semistructured Data. In *In Proc. of the ACM SIGMOD Int'l Conference*, pages 295–306, 1998.

[9] F. Rabitti, E. Bertino, W. Kim, and D. Woelk. A Model of Authorization for Next-Generation Database Systems. *ACM Trans. on Database Systems*, 16(1):88–131, March 1991.

[10] P. Samarati, E. Bertino, and S. Jajodia. An Authorization Model for a Distributed Hypertext System. *IEEE Transactions on Knowledge and Data Engineering*, 8(4):555–562, 1996.

[11] Word Wide Web Consortium. Extensible Markup Language (XML) 1.0, 1998.