INFORMATICA, 2008, Vol. 19, No. 4, 517–534

A Fine-Grained Access Control System Combining MAC and RBACK Models for XML

Mustafa M. KOCATÜRK, Taflan İ. GÜNDEM
Computer Engineering Dept., Boğaziçi University
34342 Bebek, İstanbul, Turkey
e-mail: mirackocaturk@yahoo.com, gundem@boun.edu.tr

Received: September 2006; accepted: February 2008

Abstract. In this paper, we present a novel fine-grained access control system for applications where the information flow is critical; the confidentiality of the data is essential and there are a huge number of users who access different portions of an XML document as in military applications. We combine MAC and RBACK models for XML for use in the mentioned type of applications. In accordance with the peculiarities of the target applications, the access control model is structured in such a way that the implementation can be done efficiently for large number of users. In the system presented, instead of using access control lists, we use a security labeling approach in defining the grant rules. By combining the advantages of role-based and mandatory access control schemes, the access control system presented provides a fine-grained, flexible and effective access for applications where the confidentiality of data is crucial. The system is implemented and tested for correctness. Performance analysis is also given.

Keywords: access control, MAC, RBAC, XML security, database systems.

1. Introduction

The security of data in XML (WWW Consortium, 2004) documents has become an important issue in the literature. Confidentiality, integrity and authentication (Bertino *et al.*, 2002; Sandhu, 2003) are important considerations in the security of an XML document. The confidentiality is achieved by access control models. The integrity is achieved both by access control models and by encryption techniques. Authentication may be achieved by the use of digital signature techniques. The activities of an access control system may be summarized as follows.

First a user is authenticated as a subject into the system. The authentication process gives us an opportunity to ensure that the source that wants to gain the access of a specific subject is the source that it claims to be. After the authentication phase, the computer resources (objects) are pruned to obtain the accessibility view of the data with the aid of access control policies. Accessibility view creation is essential in securing the data from unauthorized users. In a fully secure system, where the confidentiality is a critical issue, a user should not know the presence of objects that the user is not authorized to access. Thus the pruning of the resources is required in order to return only the objects that a user

is granted to access. The pruned data is called the accessibility view. The user is targeted to request from the accessibility view and a response is returned to the user. An overall access control strategy is summarized in Fig. 1.

In the literature, there are various studies (Bertino *et al.*, 2002; Kudo and Hada, 2000; Damiani *et al.*, 2000; Bertino *et al.*, 2004; Bertino *et al.*, 2001; Gabillon and Bruno, 2001) on access control models for XML documents that use Discretionary Access Control (DAC). Discretionary Access Control provides a secure and fine-grained access to the authorized subjects. However they do not prevent information flow from data objects to subjects and vice versa (please consult the appendix for more information on securing information flow). This is not acceptable in military applications.

In military applications, the main reason for using access control is the confidentiality of the information being shared among the authorized users who have different access rights with different security labels. In mandatory based access control (Sandhu, 2003), MAC, models are proposed to achieve confidentiality and integrity of data.

The novel access control system presented in this paper secures the information flow and provides confidentiality of the data. The proposed method can be effectively used in applications where confidentiality of the data is crucial and there are a huge number of users in a distributed environment such as in military applications.

In general, access control policies use access control lists, ACL, to specify the access rights of the subjects with respect to the objects. In the access control system that we present, security labeling approach is used instead of access control lists. Security labels associated with a document are in a hierarchical order. Furthermore to achieve a fine-grained access control model, access control rules are used to define the subjects who are not allowed to access specified objects.

If confidentiality is essential, the information flow between the objects and that between the subjects and the objects must be secured. In access control systems in the literature (Bertino *et al.*, 2002; Damiani *et al.*, 2000; Bertino *et al.*, 2004; Gabillon and Bruno, 2001) although the system administers the interaction between the subjects and the objects, the confidentiality of information among the objects is not taken into consideration. To secure information flow between the objects in an XML document, we use

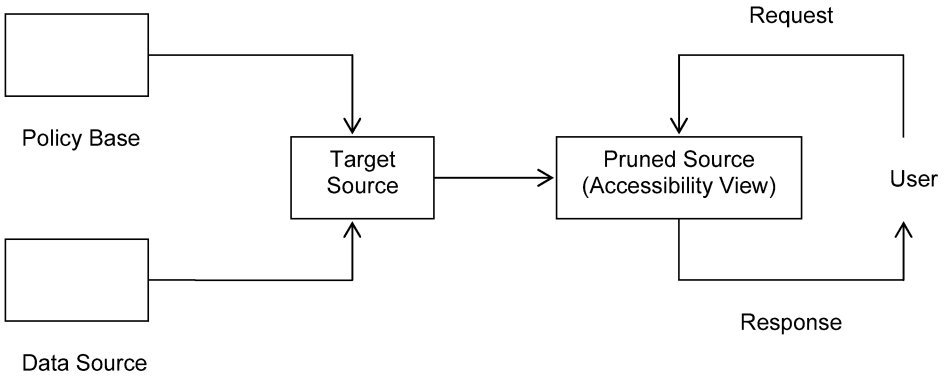

Fig. 1. Access control strategy.

mandatory access control (MAC) strategy and show that MAC can be applied to XML Access Control.

Another problem in access control is the storage and maintenance of access control lists, especially, if there are a huge number of users in the system. In the access control system that we present, we use a novel security labeling technique instead of access control lists to handle this problem.

Since XML documents are semi-structured, some XML documents may not have a predefined schema and the internal structure of the XML document may not be known. To overcome this difficulty, some techniques (Fan *et al.*, 2004; Zhuo, 2003; Harrison *et al.*, 1976) such as rewriting XPath expressions are proposed in the literature to satisfy secure query evaluation. These techniques have a performance drawback on the response time of the query to be executed. For the solution of this problem, we introduce some rules into our access control policies and use a 2-phase pruning scheme which enables us to query the database and construct the rules with XPath expressions.

The major contribution of the proposed method compared to work in the literature may be summarized as combining MAC and RBACK models for XML in a manner that allows efficient implementation. Thus the proposed method can be effectively used for large number of users. The proposed model is structured for applications where the information flow is critical and the confidentiality of the data is essential.

The rest of this paper is organized as follows. In Section 2, we have the related work in the literature. In Section 3, the proposed model is explained. In Section 4 we present and discuss the performance and the contributions of the proposed model. In Section 5, we have the conclusions.

2. Related Work

Recent work (in addition to ones already mentioned) on XML access control in the literature includes the following techniques that are based on DAC model. In (Verma, 2004) how to incorporate XACML into applications is presented. XACML is a language for specifying XML access control. It is created with the objective of providing a "portable and standard way of describing access control entities and their attributes and a mechanism that offers much finer granular access control than simply denying or granting access". In (Murata *et al.*, 2003) a technique to release the burden of XML query processors in runtime is presented. The technique uses static analysis to determine if an access request is to be granted or denied. In (Lim *et al.*, 2003) an XML access control model that supports not only read operations but also update operations is presented. In (Hada and Kudo, 2000) an access control language called XACL to specify security policies is presented. XACL supports both read and write operations and also provisional authorization. In provisional authorization a user denied access may be granted access, if the user takes certain actions requested by the system.

3. Proposed Access Control Model

The overall structure of the proposed model is shown in Fig. 2. First of all users are authenticated to the system and are defined as subjects in the system (Subsection 3.3). After the authentication process, the XML document is pruned (with the aid of grant rule policies) to specify the objects that can be accessed by the authorized user. XPath is used to specify the objects in the system (Subsection 3.2). Also grant rule policies are specified using pre-defined rules (Subsection 3.1). In these specifications, security label assignment to each individual object is determined in such a way to secure information flow. After completing the first pruning phase (Subsection 3.4), we obtain a view of the XML document that contains the objects that the user is granted to access. Next, the second pruning phase takes place (Subsection 3.6). This phase prunes objects of the XML document that the authorized subject is not allowed to access. For this process deny rules are used (Subsection 3.5). After the second pruning phase the view is ready to be accessed by the user.

In this paper, we consider only READ privileges. The READ privileges are generally expected to be the most common privileges where confidentiality is a critical issue. WRITE and UPDATE privileges are given to a small group of users whereas READ privileges are given to a large group of people. Thus most of the accesses require only READ privileges in applications where confidentiality is a critical issue. We assume the administration of the XML document, authorization of access rights to the users and the policy rules are in the responsibility of a security administrator. Therefore owner, write, delete and update privileges are not assigned to individual subjects of the system.

We use *denial takes precedence* conflict resolution method in our access control system. That is if an object is neither granted nor denied to a user than the object is denied access by default. When an access conflict arises on a subject then the conflict resolution policy operates in favor of denial rules.

3.1. *Grant Rule Policies*

In our proposed system, grant rule policies are constructed according to Mandatory Access Control model. Mandatory Access Control policies are based on the concept of as-

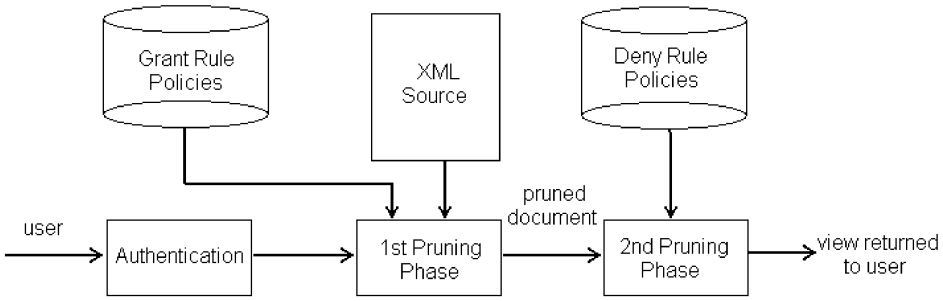

Fig. 2. The overall structure of the proposed access control scheme.

signing security labels for subjects and objects. Security labels which are assigned to the subjects are called *clearances* and the security labels which are assigned to the objects are called *classifications*.

XML documents have a tree structure as shown in the example in Fig. 3. Due to this hierarchical structure in XML Documents, we can state Rule 3.1 to secure information flow and get the highest confidentiality.

Rule 3.1. *Let us denote* $OS = \{O_{i1}, O_{i2}, O_{i3} \ldots O_{in}\}$ *as the topological sort of the object set of the XML Document (i.e., if* O_{ij} *is an ancestor of* O_{ik}*, then* O_{ij} *precedes* O_{ik} *in* OS*). Let* L_i *and* L_j *be the classifications of* O_i *and* O_j*, respectively, if* O_i *precedes* O_j*, then* $L_j \geqslant L_i$*, for any* O_i *and* O_j *in* OS*.*

Rule 3.1 states that in an XML tree, the root node has the minimum classification over all the nodes and every node has a classification that is equal to or higher than those of its ancestors. This makes information flow secure from root to leaves. An XML document should have a structure where the most sensitive information is stored in leaf nodes. An example of assigning classifications to the nodes of an XML document is given in Fig. 3. The security label of a node is given right next to the element name in a node.

By examining Fig. 3, we can make the following statements. The whole XML document's security label is 5, which means that if the security label set has a 10-degree security level, this document has at least mid-sensitive information. The most sensitive information is the position of the employee and has been assigned with the highest security label 9.

3.2. *Object Specifications*

In recent studies, object specification in access control of XML Documents are frequently implemented using path expressions since path expressions can identify elements and attributes in fine-granularity (Bertino *et al.*, 2002; Damiani *et al.*, 2000; Bertino *et al.*,

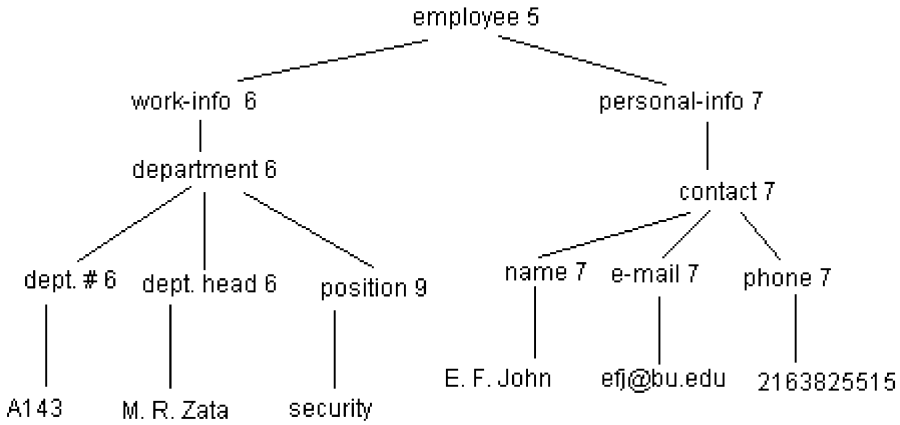

Fig. 3. An XML document with classifications assigned to nodes.

2004; Bertino *et al.*, 2001; Gabillon and Bruno, 2001). We use XPath (WWW Consortium, 1999) language to identify the elements and attributes of the XML Documents.

3.3. *Subject Specification*

Every individual user has to be uniquely defined and authorized in the system. We assume that a unique identifier such as an IP number, username, or an e-mail address is used to identify a user. Rule 3.2 states that each user is unique.

Rule 3.2. *If we denote* $U = \{U_1, U_2, U_3 \ldots, U_n\}$ *as the user set of the system, then* $U_i \neq U_j$ *where* $i \neq j$.

If the authorization is successful, then every individual user can be seen as the subject of the system. As in RBAC model (please consult the appendix for more information), the subject specification can be extended to the roles where individuals can be the members of the roles in the system. We make sure that the unique role identities (stated in Rule 3.3) are different from the unique user identities as stated in Rule 3.4

Rule 3.3. *If we denote* $R = \{R_1, R_2, R_3 \ldots, R_m\}$ *as the role set of the system, then* $R_i \neq R_j$ *where* $i \neq j$.

Rule 3.4. *If we denote* $S = U \cup R = \{S_1, S_2, S_3 \ldots S_p\}$ *as the subject set of the system, then for every* i *and* j $S_i \neq S_j$, *where* $i \neq j$.

The subject specifications will be used in authorizations and in access control policies. Therefore while specifying the grant rules, every subject must be assigned a security label. In this section, security labels and the rules for assigning these security labels to the subjects are specified. The terms security label (or security level) of the subjects and clearance are used interchangeably.

Rule 3.5 states that the number of security labels is equal to the number of objects at most. In the worst case, every object in the XML document has a different security level.

Rule 3.5. *If we denote* $L = \{L_1, L_2, L_3 \ldots L_n\}$ *as the security label set of the system and* N *as the number of the elements in the object set, then* $L_i \neq L_j$ *where* $i \neq j$ *and* $n <= N$.

Rule 3.6 states that every subject who is authorized to the system must be assigned a security label no matter whether the subject is an individual user or a role.

Rule 3.6. *Every* S_i *in set* S *is associated with a* L_i *in set* L.

Rule 3.7 states that every user of a given role must be assigned a security label greater or equal to the security label of the role that it belongs to.

Rule 3.7. (*user assignment rule*). *Let the security level of role* R_j *be* L_j. *For every user* U_i *with security level* L_i *associated with role* R_j, *it must be that* $L_i \geqslant L_j$.

If the system administrator associates a user with a role, then the clearance of the user becomes equal to that of the role if it is smaller. If it is greater than that of the role, then it is not changed. In effect, a person associated with a set of roles, has a clearance at least

as high as the clearance of the role that has the highest clearance among all the roles in the set. Roles make handling the clearances of a group of users easy. Instead of changing the clearance of users individually, we change the clearance of a role. Consequently the clearances of users associated with role x are changed if the users are associated with only role x. If a user is associated with roles other than x, obviously his/her clearance may or may not change depending on the clearances of the other roles the user is associated with.

Rule 3.8 states that a role's security level is determined by the maximum security label of the objects that are assigned to that role. In other words, a role's security level is defined by the object with the highest security label assigned to the role. The security labels of the roles are assigned directly from the objects which are defined by the permission assignment rule.

Rule 3.8. (*permission assignment rule*). *Let the security level of role R_j be L_j and that of an object O_i be L_i. Let OR_j be the set of objects associated with role R_j. $L_j = \max\{L_k | L_k$ is associated with O_k in $OR_j\}$.*

In this paper, every security level is assumed to be an integer to make the system easily understandable. The sensitivity of the security labels is defined by Rule 3.9.

Rule 3.9. *Let L denote the security label set of the system, $L = \{L_1, L_2, L_3 \ldots L_n\}$. For any L_i and L_j in L, if $L_i > L_j$, then L_i is a more confidential security label than L_j.*

The following theorem is crucial in simplifying the conflicts that rise between the RBAC and MAC models and also in securing the confidentiality.

Theorem 3.1 (confidentiality theorem). *Every user U_k associated with a role R_j can access all the objects assigned to that role* (*i.e., the security level of a given role is greater than or equal to the classifications of objects associated with that role*).

Proof. i) For every object O_i with a security level L_i associated with a role R_j (whose security level is L_j), $L_j \geqslant L_i$ (from Rule 3.8).

ii) Any user U_k (with a clearance of L_k) associated with role R_j (whose security level is L_j), $L_k \geqslant L_j$ (from rule 3.7).

i) and ii) together imply that $L_k \geqslant L_i$.

3.4. *First Pruning Process*

In our proposed system, after security labeling processes (when a subject authenticates itself to the system), the access control mechanism searches the entire document to specify the objects that can be accessed by the specific user. With the subject and object specification rules, this process becomes extremely easy to implement and operates in high performance. In our system, an object is considered to be accessible by a specific subject according to Rule 3.10.

Rule 3.10. *Let us denote L_s as the clearance of the subject S and L_o as the classification of the object O. O is considered to be accessible by S, if $L_s \geqslant L_o$ is satisfied.*

In Fig. 4, the algorithm to prune the objects which do not have a granted access to a specific user is given. In this algorithm, first of all, it is expected that a subject is authenticated to the system. After authorization process, clearance of the specific subject is obtained and starting from the root node, every object's classification is compared. If the clearance of the subject is equal to or greater than the classification of the object compared, then the object is added to the view of the pruned tree and recursively all the children nodes of the added node are compared. An example of a pruned document obtained after the first pruning phase is shown in Fig. 5. In the example, the clearance of the subject is equal to 6.

```
1. perform subject authentication
2. Let Ls be the clearance of the subject
3. Pt = {} // initialize pruned view.
4. Oi = root node // initialization root node
5. if Ls < Li then exit // subject have no permission to read the document
6. if Ls >= Li then add (Pt, Oi)
7. for all the children of the added node
8.     Oi = new child node
9.     if Ls >= Li then add (Pt, Oi) goto step 7.
10.   return Pt
```

Fig. 4. Pruning algorithm in the first phase.

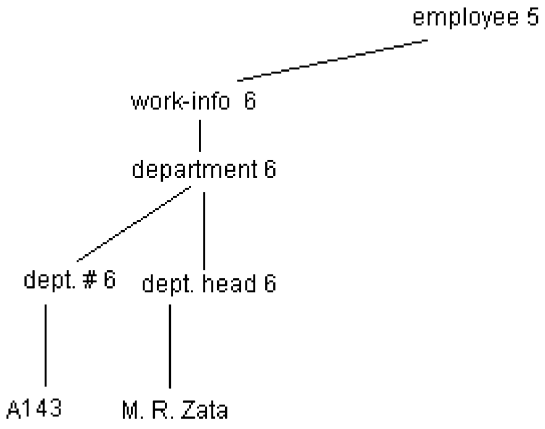

Fig. 5. The view obtained after the first pruning phase for the XML document in Fig. 3 for a subject with clearance $Ls = 6$.

3.5. *Deny Rule Policies*

In the system presented, one of our main objectives and consequently our contribution is to design a fine-grained access control model over XML documents. The model that we presented so far is inadequate because of the lack of denial rules for a given subject. For example, let us consider a scenario in which a couple of users are assigned to a role and their access specifications are assigned based on that role. However, for some reason, security administrator would like to deny access of an object to a specific subject in that role. The security administrator can not change the role's rules because that would affect all the subjects associated with the rule. Thus there should be a flexibility provided to the security administrator to achieve a fine-grained access control. The deny rules integrated in to our system and explained in the following provide the flexibility needed.

We specify a deny access control rule by a 2-tuple <s, o>, where s denotes the subject and o denotes the object. As an example let us assume that a user with an IP address 192.168.7.1 is not allowed an access on the object /profile/calendar/event/[location='Zabars']. The deny rule for this example can be constructed as <192.168.7.1, /profile/calendar/event/[location='Zabars'>.

In our system, we assume that all the deny rules are recursive rules. That is to say when we apply a rule, then the descendants of the object specified in the rule are also affected by the rule.

We store the deny rules in access control lists. Although access control lists have some drawbacks, as we mentioned earlier, they can be used to implement the deny rules which are only a small portion of the general access control rules. Since the deny rules are exception rules, we assume their number to be small.

Deny rules can also be defined for a role (i.e., the subject in rule definition is specified as a role). In such a situation the size of the ACL to store the deny rules is further reduced.

3.6. *Second Pruning Phase*

The algorithm to enforce the deny rules is given in the Fig. 6. The algorithm states that by using an XML query language processor such as an XQuery processor, the objects

```
1.   For every Oi in the deny rule policies do
2.      nodeSet = XQuery (Oi) // query the specific object
3.      if nodeSet = {} exit // if there is no object than exit
4.      if nodeSet ≠ {} then remove (Pt, node)
5.      for all the children of the removed node
6.         node = new child node
7.         if node ≠ {} then remove (Pt, node) goto Step 5
10.   return Pt
```

Fig. 6. Algorithm of the second pruning phase.

which are specified by the deny rules for a specific subject are found. If the query returns an empty result set, then we can conclude that there is no need to further pruning. Thus to get optimum efficiency from the access control mechanism, the deny rules should be defined according to Rule 3.11.

Rule 3.11. *Let L_i be the classification of object O_i and L_j to be the clearance of subject S_j. If $L_i > L_j$, then it is unnecessary to specify a deny rule $< S_j, O_i >$.*

After finding the object(s) (i.e., the **nodeSet** in the algorithm) to be pruned in the algorithm, each object (in the **nodeSet**) and all of its descendants are recursively pruned from the view of the XML document. The result of the algorithm is always an XML tree with a root node which is the same as that of the original XML Document. Thus we do not need to rewrite the original absolute XPath query to achieve a secure query evaluation. This is an important contribution of our proposed system.

In Fig. 7, we see the view obtained as a result of applying the second pruning phase to the view shown in Fig. 5 for the deny rule `<192.168.5.10, /employee/work-info/ department/[dept.# =”A143”]>`.

By the end of the two phase pruning algorithms we achieve a fine-grained access control over XML Documents. If a given subject is given a clearance that is adequate to access all the objects in an XML document, the view constructed after the first pruning algorithm will be the same as the original document. However due to the presence of deny rules for the given subject, the second pruning phase enables us to restrict the access of an object or a path of objects to the subject.

Also note that the accessibility view of an XML document is a sub-tree of the original XML document that has the same root as the original document. This enables us to use absolute path expressions in user requests and rule specifications.

The 2-phase pruning that we presented guarantees that conflict resolutions are handled for the safety of the confidentiality. That is a user is denied an access to an object (if a deny rule exists), although the user has adequate security level to access the object.

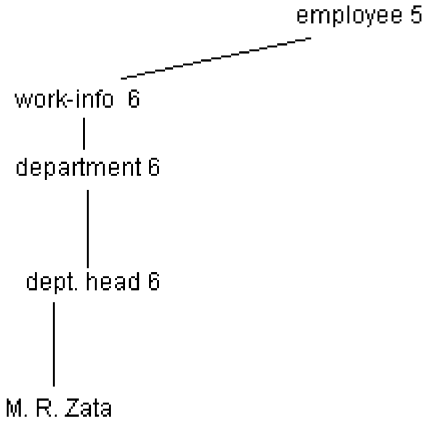

Fig. 7. The view in Fig. 5 after the second pruning phase.

4. Performance Analysis

In this paper our main contribution is designing a high-performance fine-grained access control system based on MAC and RBAC models for XML documents in applications where the information flow is a critical issue as in military applications. We show that MAC model can be appropriately used in a fine-grained access control of XML documents with the aid of RBAC model. By assigning security levels to the subjects and using the 2-phase pruning algorithm we just described, we obtain high performance.

In measuring performance, response time and memory requirements of access control algorithms are important considerations. In Fig. 8, we present the comparison of the worst case memory usage of our proposed system with that of access control matrix (ACM) based systems. In Fig. 8, we see that our proposed model uses less memory to store the access control policies than access control matrix based models.

Access control matrix is implemented as access control lists or capability lists. Access control lists are linked lists that link the subjects for a specific object. Capability lists are linked lists that link the objects together for a specific subject. The memory requirements of both implementations are equal to each other in the worst case (i.e., each user is associated with each object and vice versa).

Using security labeling technique instead of access control matrix reduces the number of memory units needed to store grant rule policies. The size of the access control matrix is calculated by multiplying the number of users with the number of objects. The access control matrix may become extremely sparse due to the fact that whether a subject is granted access to an object is not considered in allocating memory. In our proposed model, we assign security labels to each subject and object and consequently use fewer units in memory.

In most of the existing systems (Bertino *et al.*, 2002; Bertino *et al.*, 2004; Gabillon and Hudo, 2001) the access control models are based on access control lists (ACL's) which

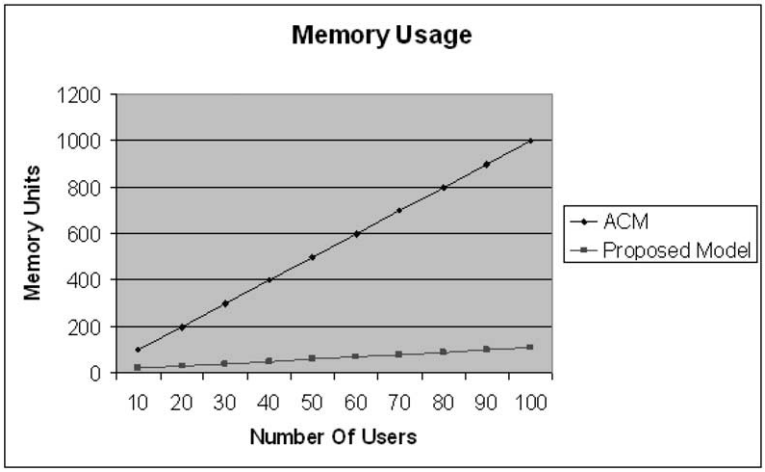

Fig. 8. Worst case memory usage comparison for 10 objects.

specify the subjects that have an access privilege on a specific object. If the number of the subjects increases, the size of the access control list increases also. This in turn causes a difficulty in the maintenance of the access control rules for the given document. Besides, access control matrix which is comprised of the ACL's is likely to be sparse. For example, in a system where there are 10000 users (denoted with U) who want to access 1000 objects (denoted with O) the matrix should have 10000000 (U*O) matrix entries. A big percentage of the whole matrix would be empty considering the fact that not all users want to access all the objects. Other drawbacks of ACL's are mentioned by Nagaraj (2001).

To deal with this efficiency problem, there are numerous proposals in the literature. In (Jagadish *et al.*, 2002) we see an important proposal that introduces the concept of compressed accessibility map (CAM). To deal with this problem, we propose a new model based on assigning security levels to both subjects (users) and objects (i.e., elements and attributes of a given XML document). In our model, we only have to specify 11000 (U+O) security entries for 10000 users (denoted with U) who want to access 1000 objects (denoted with O). The main idea is that a subject can have a granted access privilege on an object if and only if the subject has the same or higher security level than the one the object has. Therefore, there is no need to use access control lists to specify which subjects have a granted access on which object.

Our proposed model has better response time in the pruning algorithm then systems using access control lists. In systems using access control lists, for a specific subject's access control list every object is accessed (whether or not the subject has a granted access privilege on the specific object) to form a valid view of the XML accessibility view. Thus, if we specify the time metrics in the algorithm as the objects accessed, then the average response time is equal to the number of objects in the system. However in our proposed model, it may be unnecessary to access every object to compute the accessibility view of the original XML document. According to the rules for granting access (specified in Section 3), if an object is not accessible for a specific user, then we are guaranteed that every descendant of the node is not accessible to the user. Therefore there is no need to access every node in the XML tree structure. When the clearance of the subject is greater than that of every object in the XML document, the worst case time of our proposed method is equal to the average time of the systems using access control lists (Damiani *et al.*, 2000).

Three XML documents with different structures are used in comparing our proposed system with systems using access control lists, in the number of objects accessed, as shown in Fig. 9. One of the documents in Fig. 9 (XML Sample 1) has a single path structure. The other document (XML Sample 2) has a binary tree structure. The third document is the one in Fig. 3.

Let n be the number of nodes in an XML document with m levels and let n_i be the number of nodes at the ith level. If we assume that every sibling in a level of the XML tree is given the same permission, then the average case behavior in systems using ACL and our proposed system are given by $O_1(n)$ in (4.1) and $O_2(n)$ in (4.2), respectively. In our proposed model, if a subject is denied an access to an object, then all the descendants of

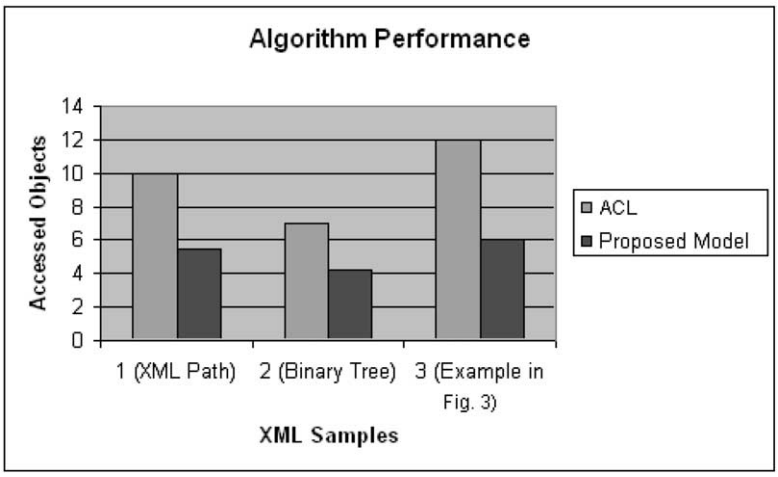

Fig. 9. Average time performance comparison for three XML documents.

that object are denied also. For any XML tree, it is always the case that $O_2(n) < O_1(n)$.

$$O_1(n) = n_1 + n_2 + n_3 + \ldots + n_m = n, \tag{4.1}$$

$$O_2(n) = \big(n_1 + (n_1 + n_2) + (n_1 + n_2 + n_3) + \cdots + (n_1 + n_2 + n_3 + \ldots + n_m)\big)/m. \tag{4.2}$$

Conflict resolution is easily handled in our model. The presence of grant access and deny access rules causes a need for conflict resolution. RBAC policy rules have dominance over the MAC policy rules (i.e., *deny overwrites*). If there is no rule defined for an object that deny the access to that role, then the object is defined as inaccessible (i.e., *closed policy*). These rules simplify the conflict resolution.

Assigning security labels to objects is done by the system administrator in the system presented. In applications for which this system is intended (such as in military applications), documents are classified and assigned security labels anyway. Thus assigning security labels to objects does not impose much burden.

The model is implemented and tested (Hünerkar, 2006). Apache Xindice 1.0 (an XML database management system) is used to store the XML data in a distributed environment. XML-DB API is used to access and manage XML data stored in Xindice. Java 2 platform, Enterprise Edition 1.4 SDK enabled accessing Xindice with XML_DB API. Netbeans IDE 5.0 is used as the integrated development environment for accessing Xindice, XML-DB API tools and J2EE SDK tools at the same time. The program was tested for 24 different XML documents related to inventory data with 5 different numbers of nodes. According to the results of the tests, it has been demonstrated that the model works correctly at least for all of the test data. The users were able to access only the parts of the document that they were allowed to. The first and second phase execution times are related to the number of nodes in the XML document. We tested the program for 5 different

XML documents with number of nodes ranging from 250 to 1351. For an XML document with 811 nodes in its DOM tree representation, the first phase takes 33.91 seconds. The first phase execution time increases nearly linearly between 250 and 1081 nodes with a difference of 51 seconds between the end points. Most of the time spent in the first phase is due to the construction of the tree representation of the XML document using DOM, in our implementation. Actually the DOM tree representation of the XML document is constructed once for an XML document and used for the access control of all users until the system shuts down. The second phase takes 2 milliseconds for an XML document with 1081 nodes in its DOM tree representation and 2 deny rules for each user. The second phase execution time changes from .5 to 2.5 milliseconds for XML documents with between 250 and 1351 nodes in their DOM tree representations.

5. Conclusions

In this paper we presented a novel fine-grained access control model, combining MAC and RBAC models, for XML Documents. The model is intended for applications where the information flow is critical and the confidentiality of the data is essential and there are a huge number of users who need to access the distributed data as in military applications. In the model each subject and object is assigned a security label. A 2-phase pruning scheme is developed to enforce the specified grant and deny rule policies. The contributions and important aspects of the model can be summarized as follows. MAC and RBAC models are combined. Conflict resolution is handled automatically. Coarse-grained access control is supported which helps in increasing the efficiency. The maintenance of the access control administration is eased by supporting RBAC model. A fine-grained access control is achieved due to deny rules which are implemented using ACLs to increase efficiency and availability. By eliminating the need for ACLs in granting access to objects memory usage performance is increased. In summary, the main contribution is combining MAC and RBACK models for XML (in a manner that allows efficient handling) for applications where the information flow is critical; the confidentiality of the data is essential and there are a huge number of users who access different portions of an XML document as in military applications.

Appendix A

Perliminary Information

Access control is generally referred to as the process of limiting accesses to system objects to authorized users. In general access control is defined by access control rules. An access control rule specifies which objects can be accessed by which subjects with which privileges. All the access control rules together for the subjects in the system can be represented by access control lists (ACLs) (Sandhu and Samarati, 1994). Each access control list defines the subjects that are given access to a specific object. For all the subjects in a

Table 1
An access control matrix

	Object 1	**Object 2**	**Object 3**	**Object 4**
Subject 1	R,W	W		
Subject 2		R	W	
Subject 3	W			R,W

system, access control lists form a general access control policy called the access control matrix (Sandhu and Samarati, 1994). A sample access control matrix is given in Table 1.

In Table 1, R and W stand for read and write privileges, respectively. For example, subject 1 can read access object 2 and write access object 3. If the default semantics is *deny* and the node is neither granted nor denied an access, then the node is not accessible. If the default semantics is *grant,* then the node is considered to be accessible. The default semantics is specified by taking the sensitivity of the information into consideration.

Access control is implemented using an access control mechanism which enforces access control policies over requests of users (Sandhu and Samarati, 1994). A reference monitor establishes the validity of a request and returns a decision either granting or denying a user to access the data. A reference monitor comprises of two parts: an access control decision function (ADF) which actually defines whether the request should be permitted and access control enforcement function (AEF) which uses ADF decisions to construct the object which will be the response to the user. To achieve confidentiality and security, the subject should not know whether there are objects that the subject is not authorized to access. Thus, AEF generally prunes the objects that the user is not authorized to access from the list of all objects before returning it to the user.

Access Control Models

Some of the fundamental access control models are briefly summarized in the following.

Discretionary Access Control Model (DAC)
DAC model enforces the access control on the specific objects using subject identities and authorizations (Sandhu and Samarati, 1994). Each request of a user to access an object is checked against specified authorizations. If there exists an authorization stating that the user can access the object in the specified mode, the access is granted. Otherwise it is denied. Access control matrices and access control lists can be used to model access control policies (Sandhu and Samarati, 1994). The main idea behind DAC model is that the decision that a subject can grant or revoke an access privilege on a subject is left to the discretion of users. A subject with sufficient access privilege on an object is capable of passing that privilege to another subject in the system. Thus, one of the drawbacks of the discretionary access control model is that it provides no real assurance on information flow (Sandhu and Samarati, 1994). In Fig. 10 we see how subject 3 bypasses authorization.

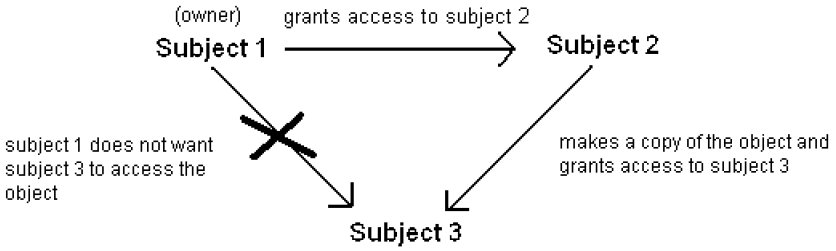

Fig. 10. Bypassing authorization.

A subject who is able to access an object can pass this authorization to other subjects (including those not authorized to read it) without the cognizance of the owner of the object. This is because access privilege of an object is left to the discretion of users (Sandhu and Samarati, 1994). That is dissemination of information is not controlled in DAC.

Also granting and denying access privileges introduce a problem of access conflicts that must be handled.

Mandatory Access Control Model (MAC)

This access control model is designed for securing the information flow between the objects in the system. *Information flow* is the flow of information from the object to the subject or vice versa. MAC model enforces the access control according to the sensitivity of the information and the trustworthiness of the subjects. In MAC policy, every subject and object is assigned a security level which is determined according to the sensitivity of the information for the object and according to the trustworthiness of the user for the subject. Security levels are elements of a partially ordered set. An example of the security level in military units may be defined as top secret (ts), secret (s) and unclassified (u), where the ts is the most secure level and u is the least. The information flow is from u to ts.

In this model a subject can have a READ access privilege on an object, if the security level of the subject is no less than the security level of the object. This model ensures that the information can only flow from a security level x to a security level y that is at least as secure as level x. This ensures that the information flow is secure and the information can not leak to lower security levels.

Role-Based Access Control Model (*RBAC*)

Role-Based Access Control (RBAC) may be a supplementary model for the discretionary access control and mandatory access control models (Sandhu and Coyne, 1996). The main idea behind RBAC is to assign the individual subjects with common access control privileges to a role. In most systems a subject can be assigned to more than one role and the roles can be in a hierarchical structure.

A role can be defined as a set of actions and responsibilities associated with a particular set of permissions. Instead of specifying all the accesses that each user is allowed to execute on objects, access authorizations on objects are specified for roles. Then the users of the system are given authorizations to adopt roles.

In DAC models, the administration of the access privileges among the subjects is not central whereas in RBAC policies the administration of the roles among the subjects is central and administrated by the security administrator. By grouping the access control privileges, RBAC policy enables reasonably efficient management of the access control of individual users.

In MAC models, the administration of RBAC requires special attention to secure information flow (Osborn, 1999). Security classifications and clearances must be taken into consideration while assigning users to the roles and assigning permissions to the roles.

Access Control in XML Documents

Due to the nature of XML documents, access control strategies give access privileges to users to access different portions of the XML data with fine granularity. A fine-grained access control model over an XML document should be able to specify that a user can have an access right for an individual element or attribute of the XML document. When doing this, the hierarchical relationship among the elements must be taken into consideration.

References

Bertino, E., S. Castano and E. Ferrari (2001). Securing XML documents with AuthorX. *IEEE Internet Computing*, **5**(3), 21–31.

Bertino, E., S. Castano, E. Ferrari and M. Mesiti (2002). Protection and administration of XML data sources. *Data & Knowledge Engineering*, **43**, 237–260.

Bertino, E., B. Carminati and E. Ferrari (2004). Access control for XML documents and data. *Information Security Technical Report*, **9**(3), 19–34.

Damiani, E., S. Vimercati, S. Paraboschi and P. Samarati (2000). Securing XML documents. In *Proceedings of 7th International Conference on Extending Database Technology*. Konstanz, Germany, March 27–31. pp. 121–135.

Fan, W., C. Chan and M. Garofalakis (2004). Secure XML querying with security views. *ACM SIGMOD*, June, 13–18.

Gabillon, A., and E. Bruno (2001). Regulating access to XML documents. In *Fifteenth Annual IFIP WG 11.3 Working Conference on Database Security*. Niagara on the Lake, Ontario, Canada, July 2001. pp. 16–28.

Hada, S., and M. Kudo (2000). *XML Access Control Language: Provisional Authorization for XML Documents*. `http://www.trl.ibm.com/projects/xml/xacl/xacl-spec.html`

Harrison, M., H.W.L. Ruzzo and J.D.Ullman (1976). Protection in operating systems. *Communications of ACM*, **19**(8), 461–471.

Hünerkar, G. (2006). *Implementation of a Fine-Grained Access Control System for XML Documents with MAC and RBAC Models*. Project report, Computer Engineering Department, Boğaziçi University.

Jagadish, H. V., D. Srivastava, L. Lakshmanan and T. Yu (2002). Compressed accessibility map: efficient access control for XML. In *Proceedings of the 28th VLDB Conference*. pp. 478–479.

Kocatürk, M.M. (2005). *Design of a Fine-Grained Access Control System for XML Documents with MAC and RBAC Models*. M.S. Thesis, Boğaziçi University.

Kudo, M., and S. Hada (2000). XML document security based on provisional authorization. In *Proceedings of the 7th ACM Conference on Computer and Communications Security*. ACM Press. pp. 87–96.

Lim, C.-H., S. Park and S.H. Son (2003). Access control of XML documents considering update operations. In *Proceedings of the 2003 ACM Workshop on XML Security*. ACM Press. pp. 49–59.

Murata, M., A. Tozawa, M. Kudo and S. Hada (2003). XML access control using static analysis. In *Proceedings of the 10th ACM Conference on Computer and Communications Security*. ACM Press. pp. 73–84.

Nagaraj, S.V. (2001). Access control in distributed object systems: problems with access control lists. In *Proceedings of the 10th IEEE International Workshops on Enabling Technologies; Infrastructure for Collaborative Enterprises*. IEEE Computer Society. pp. 163–164.

Osborn, S. (1999). Mandatory access control and role-based access control revisited. In *Proceedings of the Second ACM Workshop on Role-based Access Control*. pp. 22–27.

Sandhu, R.S., and P. Samarati (1994). Access control: principles and practice. *IEEE Communication Magazine*, September, 40–48.

Sandhu, R.S., and E.J. Coyne (1996). Role based access control models. *IEEE Computer*, February, 38–46.

Sandhu, R.S. (2003). Lattice based access control models. *Computer*, **26**, 9–19.

Verma, M. (2004). *XML Security: Control Information Access with XACML.*
`http://www.ibm.com/developerworks/xml/library/x-xacml/`

World Wide Web Consortium (1999). *XML Path Language (XPath)* Version 1.0.
`http://www.w3.org/TR/1999/REC-xpath-19991116`

World Wide Web Consortium (2004). Extensible Markup Language (XML) 1.1.
`http://www.w3.org/TR/2004/REC-xml11-20040204/`

Zhuo, D. (2003). *On Fine-grained Access Control for XML*. M.S. Thesis, University of Waterloo.

M.M. Kocatürk is a graduate student in computer engineering. He has worked on several software projects on security of government and military information systems as a designer and programmer.

T.İ. Gündem is a professor of computer engineering in Boğaziçi University. He has BS and MS degrees in electrical engineering from Boğaziçi University and PhD in computer science from Oregon State University. His research interests are in information systems and database systems.

Tikslios prieigos valdymo sistema, apjungianti XML modelius MAC ir RBACK

Mustafa M. KOCATÜRK, Taflan İ. GÜNDEM

Šiame straipsnyje pristatoma novatoriška tikslios (XML dokumento atributų lygio) prieigos valdymo sistema, tinkama naudoti ten, kur ypač svarbu užtikrinti informacijos srautų saugumą ir duomenų konfidencialumą, o vartotojų, nuolatos besikreipiančių į įvairias XML dokumentų sritis, yra labai daug. Tokio pobūdžio taikymuose autoriai siūlo apjungti XML modelius MAC ir RBACK. Atsižvelgiant į numatytos taikymo srities ypatybes, prieigos valdymo modelis yra struktūrizuotas taip, kad galėtų būti veiksmingai realizuotas dideliam kiekiui vartotojų. Aprašomoje sistemoje vietoj prieigos valdymo sąrašų autoriai teisių suteikimo taisyklėms aprašyti panaudoja saugumo žymėjimo požiūrį. Apjungus rolėmis grindžiamo ir privalomo prieigos valdymo schemas pasiekta, jog straipsnyje pristatoma sistema leidžia ypač detaliai, efektyviai ir lanksčiai valdyti prieigą srityse, kuriose būtina garantuoti duomenų saugumą. Sistema buvo realizuota ir ištestuota. Jos našumo analizė taip pat pateikta straipsnyje.