CORRELATED DATA INFERENCE

Ontology Guided XML Security Engine

Csilla Farkas
Information Security Laboratory
University of South Carolina *
farkas@cse.sc.edu

Andrei G. Stoica
Information Security Laboratory
University of South Carolina
stoica@cse.sc.edu

Abstract In this paper we examine undesired inferences in distributed XML documents. An undesired inference is a chain of reasoning that leads to protected data of an organization or an individual, using only intentionally disclosed information. We propose a framework, called Ontology guided XML Security Engine (Oxsegin), to detect and prevent undesired inference attacks. Oxsegin uses the Correlated Inference Algorithm to detect sensitive associations that may exist at a lower security levels. The system operates on the DTD's of XML documents to identify data associations and the corresponding security classifications. Oxsegin uses an ontological class-hierarchy to identify associations with two or more conflicting classifications. A Security Violation Pointer (SVP) is assigned to a set of tags that contribute to the conflicting classification. The likelihood of a detected security violation is measured by a confidence level coefficient attached to the SVPs.

Keywords: XML security, ontology based inference attack, data aggregation, correlated data inference, multi-level XML security

*This work was partially supported by the National Science Foundation under Grants IIS-0237782 and DUE-0112874

1. INTRODUCTION

Information systems have become a fundamental part of our everyday life. During the last few years the number of distributed applications using eXtensible Markup Language (XML) increased and the concept of Semantic Web emerged [15]. XML query languages [1, 17], supported by ontologies [11, 9, 8, 16, 2], enable semantic-based information processing without human assistance.

Unfortunately, techniques that support interoperation may also lead to unintended data disclosures. While individual data units are usually carefully analyzed not to disclose any confidential information, correlated data may allow unintended disclosure of confidential information. Current research in XML security follows two main trends: (i) Document Instance Security for digital signatures [5] and encryption [18] and (ii) Access Control Models for multi-level XML documents [3, 10, 4, 6, 14]. These techniques, however, do not consider the security implications of automated correlation of large amount of machine-understandable data that may lead to undesired inferences.

Intuitively, an undesired inference occurs when a user is able to infer non-permitted information from intentionally disclosed data and available metadata, such as ontologies. This inference threat is similar to the inference problem in traditional databases, where the ontology corresponds to the external domain knowledge. However, traditional inference control techniques are insufficient to provide protection against undesired inferences due to (i) the dynamic nature of the Web, (ii) the large amount of information to be processed, and (iii) the fact that the owner of the sensitive information does not have control over all publicly available data that may lead to undesired inferences. Up to date, small-scale data availability and the lack of automated data correlation tools limited the threat of unwanted inferences via external domain knowledge. The impact of automated XML document correlations from large distributed databases using ontologies has not been yet fully addressed from the information security point of view.

Our research targets the security impact of the ontology enhanced XML processing tools over large, distributed XML databases. We show that it is possible to use ontologies to mount specific data inference attacks on XML data. We develop techniques to detect and prevent attacks due to coexistence of sensitive association at a lower security level. To prevent these attacks, we propose the Ontology guided XML Security Engine (Oxsegin). Oxsegin is a probabilistic engine that computes security violation pointers over those tags that lead to low-level duplicates of sensitive associations.

The rest of the paper is organized as follows: Section 2 presents an example of ontology-guided attack using public domain data. Section 3 describes the architecture and functionality of Oxsegin. Section 4 gives the technical details for the correlated data inference process in the security engine. Finally we conclude and propose future research in Section 5.

2. ONTOLOGY-BASED ATTACKS IN XML DATABASES

Undesired inferences in multilevel secure databases have been studied extensively (see [12] for an overview). The inference problem is to detect and remove inference channels that lead to disclosure of unauthorized data by combining authorized data and metadata. In Web environment, where correlated data may come from several, independent sources, only a small portion of publicly available data is under the control of the owner of the sensitive information. This work focuses on detecting replicated data associations with different security requirements.

We assume that organizational data repositories contain both public (e.g., available from the Web) and confidential (e.g., available only to some of the users) data . To prevent undesired data disclosure, it is required that the security consequences of the release of new public data are evaluated before the release. That is, to determine whether unauthorized users will be able to combine the new information with other publicly available data to gain access to confidential data. Ontologies support semantic-based data integration, thus extend upon purely syntax-based data integration.

To illustrate a possible inference attack, consider the document fragment (Table 1.a) part of a database carrying information for upcoming air-shows. This document provides information such as the address and driving directions to military bases (Base_X) where an air-show is held. The second document fragment (Table 1.b), extracted from a local State Division for Health Administration, shows a map of drinking water basins within a given state. Finally, the third fragment (Table 1.c), is part of a sensitive document, containing data about the locations of the water sources for several military bases, including Base_X. The security requirement of the military is that the information about the water reservoirs of military bases should only be accessible by authorized users. The air-show information (fragment 1) is available on-line and the drinking water basins information (fragment 2) is outside of the military protection domain and publicly available. Indeed, our example

is based on data available on existing Web site but we replaced the real data with fictional values.

a. Air-show information	b. Drinking water basins	c. Critical Infrastructure
<?xml version="1.0"?>	<?xml version="1.0"?>	<?xml version="1.0"?>
<show> ... **P**	<waterMap> ... **P**	<infrastructure> ... **S**
<fort>Base_X	<district>District_Y	<base>Base_X
</fort> ... **P**	</district> ... **P**	</base> ... **S**
<address>District_Y	<basin>Basin_Z	<waterSource>Basin_Z
</address> ... **P**	</basin> ... **P**	</waterSource> ... **S**
</show>	</watermap>	</infrastructure>

Table 1. Undesired Inference from Public Data

A possible ontology for this attack unifies the <waterSource> with <basin>, <fort> with <base> and <address> with <district> tags. Using this correlation, the attackers gain access to secret information (association between the Base_X and its water source in Basin_Z), without any access to the critical infrastructure database. Note, that the complexity of this attack is reduced by the simplicity of the ontology and the uniform access to online resources.

3. ONTOLOGY GUIDED XML SECURITY ENGINE

The motivation for the design of Oxsegin was to assist security officers and database administrators to securely update XML databases by identifying possible security violations from illegal inferences. Oxsegin uses a probabilistic inference engine with varying precision levels. Oxsegin indicates the possibility of unwanted inferences where the correlated data from the test files (publicly available data) matches the reference file (protected, confidential data). If unwanted inference is detected appropriate countermeasures must be performed, e.g., withhold some of the test files or execute non-IT measures.

The security engine has four main components: the Probabilistic Inference Module - PIM, the User Defined Inference Parameters Module - UDIPM, the Ontology Module and the XML Database Access Module. The Input and Feedback Module - IFM is not incorporated in the Oxsegin architecture. The IFM functionality is to supply the reference and test XML set, the inference parameters and to decide the appropriate actions if a security violation is detected. The response policy on detected security violations is outside of the scope of this paper.

PIM computes possible security violation pointers between the reference document and the set of test documents. Intuitively, a security

violation pointer indicates tags from the corresponding reference and test DTD files that might constitute unwanted inferences. For each security violation pointer, PIM computes an associated confidence level coefficient that reflects the likelihood of security violation involving the set of tags. UDIPM allows the security officer to define different inference processing parameters that will control the complexity of inference analysis. The inference uses the semantic formalism and concept hierarchy supplied by the Ontology module.

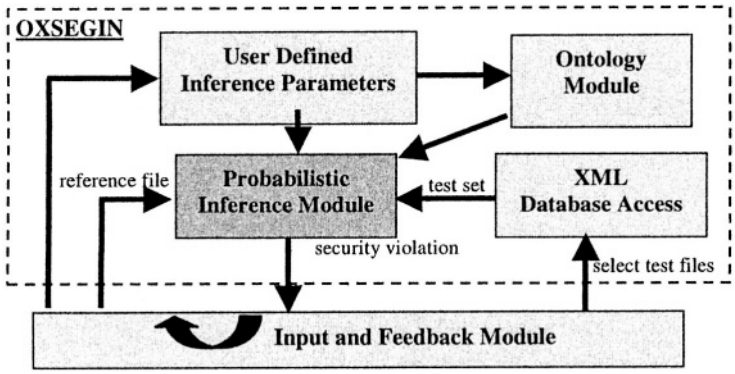

Figure 1. Replicated Information under Different Classification

The XML Database Access module represents a gateway to a collection of XML documents. The XML database can be the local, public document repository or files accessed via HTTP within a given web domain. As a result, Oxsegin can be used to securely publish documents over the web. In this case the reference DTD is the protected document and the test DTD is the set of all documents from the public domain.

3.1. Probabilistic Inference

PIM uses a set of procedures to identify security violations employing the ontology module to guide the inference process. Section 4 describes in full details the Correlated Inference Algorithm. The The input from the ontology module is used to abstract the concepts represented by the tags within the DTD files. A security violation pointer SVP is assigned to every unwanted inference. The confidence level coefficient CLC is computed for each SVP, based on the weights of the concepts in the ontology, the relative position of the tags in the DTD files, and the relative position of the concepts in the ontology class hierarchy.

Definition 1 *(Security Violation Pointer) Security Violation Pointer (SVP) is a set of tags* $T = \{t_1, \ldots, t_N\}$ *that represent a possible security violation via unwanted inference.*

Definition 2 *(Confidence Level Coefficient) The Confidence Level Coefficient (CLC) of an SVP is the likelihood of the inference involving the tags of the SVP.*

Within a DTD, we distinguish between syntactically identical tags at structurally different locations. We define all tags as a pairs, containing the tag's name and the tag's path information from the root node of the DTD. For clarity, in the following we omit the path information unless it is needed to differentiate between the tags.

To formalize ontologies we adapt the use of Frame Logic [13] as the conceptual modeling language. We assume that the security officer assigns a weight to each concept in the ontology class hierarchy to differentiate between less and more specific concepts from the perspective of the protected sensitive information. The more specific a concept is, the larger the weight assigned to it. The root of the ontology class-hierarchy has a minimal weight since it is the least specific concept. Concepts that are relevant to the given knowledge domain and the specific security requirements usually carry larger weights. After the security officer assigns the weights for each concept, the system computes the normalized weights for each concept. Normalized weights reflect the likelihood of the same syntactic forms to represent the same semantic concepts.

Definition 3 *(Ontological Abstraction Level) Given the concept C from ontology O, the Ontological Abstraction Level of C, denoted as OAL(C), is* n *if C is located at depth* n *in the corresponding ontology class hierarchy. The root concept* C_R *of the class-hierarchy has* $OAL(C_R)=0$.

Definition 4 *(Base Ontological Abstraction Level) The Base Ontological Abstraction Level of a tag* t, *denoted as* $BOAL(t)$, *is the OAL of the concept C contained within the tag* t.

Definition 5 *(Abstracting a concept N steps) A concept C from an ontology O is abstracted N steps when it is replaced N times by its immediate parents in the corresponding ontology class-hierarchy.*

Definition 6 *(Container and Data Tags) A container tag is an XML tag that holds only structural information in the form of other XML tags and has no tag attributes. A data tags is an XML tag that contains at least one unit of information. A data tag may contain data and container tags.*

4. CORRELATED INFERENCE

In this section we propose an inference procedure that detects undesired inference attacks within a particular knowledge or semantic domain. The Correlated Inference Algorithm detects ontology-based attacks similar to the one described in Section 2. The procedure checks a reference DTD structure (corresponding to the classified information) against a set of test DTD structures (corresponding to the publicly available information) by abstracting tags using the ontology.

The main data structure used by the Correlated Inference Algorithm is an Inference Association Graph (IAG). Intuitively, IAG represents the associations among tags of an XML DTD structure. The nodes of an IAG correspond to the XML data tags and the edges represent associations between the tags. Figure 2 represents the IAG corresponding to the XML files in Table 1.

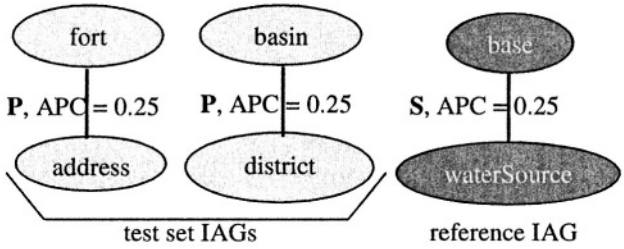

Figure 2. Inference Association Graphs IAGs

Each association has an attached Association Probability Coefficient (APC) that reflects the likelihood the corresponding nodes represent related concepts. In addition, associations can be classified according to the security policy of the organization. A security violation pointer identifies associations of different IAGs where two or more associations exist among the same tags but they have different security classifications. Such associations represent cases where users can derive information in one set of documents while they are disallowed to access the same information in a different set of documents.

Definition 7 *(XML Association) For any two nodes n_1 and n_2 in the DTD and their immediate parent P with security label L_P , P defines an XML association between n_1 and n_2. The association has a corresponding security label L_P and P represents the association source.*

Definition 8 *(Association Probability Coefficient) Association Probability Coefficient, denoted as APC, corresponding to an association be-*

tween tags n_1 and n_2 with an association source P, represents the probability that P is used to semantically correlate tags n_1 and n_2.

Definition 9 *(Inference Association Graph) The Inference Association Graph of an XML DTD structure, denoted by IAG=(V, E), is a graph with nodes V (data tags of the XML) and edges E (associations among the tags). Each edge is labeled with a pair (L_P, APC_P), representing the security label and probability coefficient of the association source tag.*

Definition 10 *(Document Structure Level DSL(t)) Given a tag T from a DTD tree D, the document structure level of T in D, denoted as DSL(T), is the maximum depth of the sub-tree rooted at T. All the leaves $l_1, l_2, \ldots, l_k$ in the DTD have $DSL(l_i) = 0$.*

Algorithm 1: Build IAG
Input: DTD
Output: IAG
BEGIN
FOR ALL data tags T_i in DTD DO
Create a corresponding node V_i
FOR ALL tags T_i in DTD DO
FOR ALL V_j, V_k such that T_i is the nearest parent of both tags T_j, T_k, corresponding to nodes V_j and V_k, respectively AND Depth(T_j)-Dept(T_i)<MaxDepth, Depth(T_k)-Dept(T_i)<MaxDepth DO
Create the edge e between (V_j, V_k)
Label e with (L_{T_i}, APC_{ijk})
END

Note, that it is always possible to find an XML association between any two tags in a DTD structure since the root tag is the parent for all tags in the DTD tree. However, this type of remote association is rarely relevant. In general, it is reasonable to assume that APCs decrease with the distance between the associated elements and the source. Algorithm 1. gives the procedure to build the IAG. To reduce the complexity of the inference process, the algorithm limits the number of tags considered for XML associations. Associations are considered only if the relative difference between the tags and the association source in the DTD tree is less than MaxDepth (set accordingly to the specifics of the XML data).

$$APC_{ijk} = \frac{1}{1 + Depth(T_j) - Depth(T_i)} * \frac{1}{1 + Depth(T_k) - Depth(T_i)} * \frac{1}{1 + |Depth(T_j) - Depth(T_k)|} * \frac{1}{DSL(T_j) + 1} * \frac{1}{DSL(T_k) + 1}$$

$\frac{1}{1+Depth(T_j)-Depth(T_i)}$ and $\frac{1}{1+Depth(T_k)-Depth(T_i)}$ coefficients in the definition of APC_{ijk} quantify the relative depth difference in the DTD tree between the associated tags and the source of the association. APC decreases with the distance between the tags and the association source. $\frac{1}{1+|Depth(T_j)-Depth(T_k)|}$ coefficient quantifies the relative depth difference between the associated tags. Tags at the same depth have a corresponding APC larger than tags at different depth in the DTD tree. $\frac{1}{DSL(T_j)+1}$ and $\frac{1}{DSL(T_k)+1}$ coefficients quantify the structural complexity of the associated tags. Tags that represent the root of larger sub-trees are more likely to be container tags, and this reduces the relevance of any association involving them.

Object[ ].	OAL=0 WGT=1	P=1/50
waterSource :: Object	OAL=1 WGT=15	P=15/50
basin :: waterSource	OAL=2 WGT=1	P=1/50
place :: Object	OAL=1 WGT=15	P=15/50
district :: place	OAL=2 WGT=1	P=1/50
address :: place	OAL=2 WGT=1	P=1/50
base :: Object	OAL=1 WGT=15	P=15/50
fort :: base	OAL=2 WGT=1	P=1/50

Table 2. Ontology represented with Frame Logic statements

After building the IAG for each XML DTD structure in the test set, the ontology is used to integrate them into a single structure - the test set IAG. The Frame Logic statements in Table 2. represent the ontology associated with the knowledge domain of the XML DTD structure in Table 1. Each concept is shown with the associated ontology abstraction level OAL, weight WGT, and normalized weight P. If the DTD structures in the test set belong to the same knowledge domain, abstracting the tag names may create pairs of duplicated nodes among different IAGs. Merging the duplicated nodes connects the test set IAGs. Each node in the IAG has an attached Concept Abstraction Level coefficient (CAL). Intuitively, CAL reflects the likelihood that the new concept is an abstract representation of the tag that is replaced. For the initial concepts in the DTD structure, CAL=1. Then for each abstraction, CAL is modified using the probability of the new concept in the ontology.

Definition 11 *(Concept Abstraction Level) Concept abstraction level (CAL) is the likelihood that the concept from the ontology hierarchy is*

an abstract representation of the initial XML tag name. For repeated replacements, CAL is the probability the present concept is an abstract representation of the original tag name.

Given the tree structure of the XML documents as well as the ontology hierarchy, all tags would eventually collapse into a single node if abstracted to the root of the ontology. To prevent this from happening, the Correlated Inference Algorithm has a set of restrictions on the abstraction process and the tags that it uses. The concepts are only abstracted within two predefined OAL limits: MaxOAL and MinOAL. MaxOAL is usually set to the depth of the ontology hierarchy tree while the MinOAL is set according to the specifics of the ontology. Usually, MinOAL is the average ontology depth of the concepts targeted by the inference attacks and is set by the security officer based on a particular knowledge domain. The second restriction on the abstraction process is based on the targeted tags. Tags located towards the root of the XML document are usually container tags, mostly used for structuring the document and rarely involved in semantic correlations. The security officer assigns a maximum level MaxDSL in the XML structure to consider tags in the abstraction process (DSL the document structure level).

Integrating the test set IAGs simulate the natural human brain inference process in three distinct stages. In the first stage the concepts associated with XML tags are abstracted, unifying same notions originally under different syntactic forms. In the second stage, by eliminating the duplicated nodes and collapsing the multi-structure IAGs, the system simulates the inference link between multiple files with related data. In the third stage the system performs a transitive correlation to simulate linking XML tags through similar abstract concepts. The transitive correlation relates two tags through an XML association (IAG edge) with a common third tag. Since the targeted inference is usually between multiple DTD structures, it follows naturally to perform the transitive correlation after duplicated node reduction. Algorithm 2. gives the formal description of the Correlated Inference Algorithm.

Each edge added in the transitive correlation of the test set IAG represents a possible illegal inference. The Correlated Inference Algorithm checks all these edges against the reference IAG to identify security violation pointers. The test for security violation pointers is performed on edges, since the edges represent valid XML associations. Each edge added to the test set IAG by the transitive correlation is compared to all edges in the reference IAG. The system places a security violation pointer (SVP) on pairs of edges between similar nodes if the reference edge security label dominates the test set edge security label. Intuitively this means that an association from the reference DTD structure

Algorithm 2: Correlated Inference
Input: Test set IAGs and Reference IAGs
Output: Security Violation Pointers (SVP)

MAIN
Check security violation for all existing edges of test and reference IAGs
FOR all edges use **Procedure 2** to identify SVPs
Abstract tags in test IAGs to create new associations
Let S be the set of all tag pairs (T_i, T_j)$\in$ test set IAGs
FOR ALL pairs (T_i, T_j) in S
Use **Procedure 1** to abstract T_i and T_j to the nearest concept c such that $c = T_i^l = T_j^k$
IF $T_i^l = T_j^k$ THEN
Merge nodes T_i and T_j:
Remove T_j; direct all edges to T_i, CAL_{T_i}=min[CAL_{T_i};CAL_{T_j}]
Create edges through transitive correlation:
FOR ALL tags T_h and T_k with an edge to T_i DO
Let security label L=max[$L_{e(T_i,T_h)}$;$L_{e'(T_i,T_k)}$]
Let APC=$APC_{e(T_i,T_h)}$*$APC_{e'(T_i,T_k)}$
Connect T_h and T_k by e_n with label (L,APC)
Check security violations due to newly created edges
FOR all new edges use **Procedure 2** to identify SVPs
Check data-level matches of SVPs
FOR $\forall$ SVP_i such that CLC_i>DSTcoef DO
Perform data search on associated tags
IF data-level match found THEN
CLC_i=1
END(MAIN)

Procedure 1: Abstract tag T
Input: *Ontology class hierarchy H, tag* T
Output: (Abtracted T)= T^n
Let $c, c_1, \ldots, c_n$ concepts$\in H$, $c = T$, c_i immediate parent of c_{i-1}
T^0=T, $n < MinOAL$
FOR i=1 TO n DO
$T^i = c_i$
CAL_{T^i}=$CAL_{T^{i-1}}$*P(c_i)
END(Procedure 1)

Procedure 2: Edge test for SVP
Input: edges $e = (n_1, n_2)$, $e' = (n_1', n_2')$ with security classifications L_e, $L_{e'}$
Output: SVP and CLC or nothing
Abstract tags of nodes n_1 and n_2 of e and n_1' and n_2' of e'
IF $e \equiv_H e'$ ($n_1^k = n_1'^l$ AND $n_2^x = n_2'^y$) and $L_e \neq L_{e'}$, say $L_e \succ L_{e'}$ THEN
CAL_{avrg},CAL_{max},CAL_{min}=(average,max,min) CAL for e, e' nodes
CLC=CAL_{avrg}*APC_e*$APC_{e'}$*(1-|CAL_{max}-CAL_{min}|)
Place SVP on e and e' nodes with CLC
END(Procedure 2)

is classified at a higher security level than an association among the test DTD structures discovered by the transitive correlation procedure. The edges are matched for a security violation employing again the ontology hierarchy to abstract concepts for each tested edge. Each SVP has a confidence level coefficient CLC computed based on the APC of the edge and the CAL of the nodes. The last coefficient in computing CLC, $1 - |CAL_{max} - CAL_{min}|$ quantifies the relative difference between the maximum and minimum level of abstraction for the concepts in the XML associations. Concepts on the same level of abstraction in the ontology hierarchy have a higher associated CLC.

Figure 3 shows the reference IAG and the integrated test set IAG corresponding to the IAGs in Figure 2 and the XML files in Table 1. The tag <fort> was abstracted to <base> and the tag <basin> was abstracted to <waterSource>. Both tags <address> and <district> were abstracted to <base> inducing a transitive correlation between <base> and <water Source>. The new XML transitive association between <base> and <waterSource> is classified public according to the Correlated Inference Algorithm. This triggers a security violation between the test set and the reference IAG where the same association is classified secret.

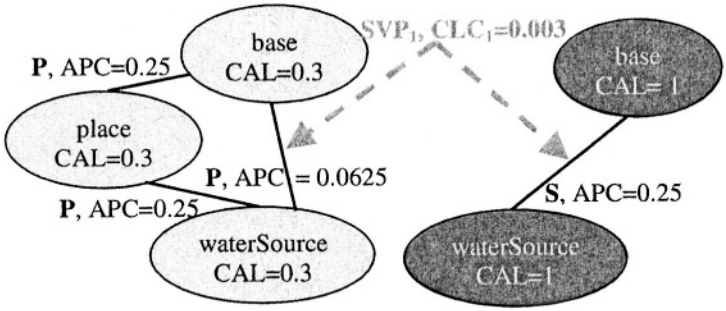

Figure 3. Unified Inference Association Graph

If the CLC corresponding to a particular SVP is above the Data Search Threshold coefficient (DSTcoef), the system provides low-level data granularity search. If data items associated with the reference and test set XML DTD structures match, the associated CLC is set to 1, the maximum confidence level. The low-level data search provides maximum security but also maximum processing complexity. High-level detection may produce false positive security violation pointers with high confidence coefficients. Data granularity search decreases the amount of false positives but does not guaranty to eliminate all of them. The

Correlated Inference Algorithm runs the analysis for security violation pointers on the DTD structure level. This represents an advantage for large XML documents databases where usually more than one document corresponds to any given DTD file. Operating at the DTD level is similar to high-level security detection with reasonable accuracy under reasonable computational complexity. For more accurate detection the procedure uses specialized data granularity search to identify security violations with maximum confidence level.

5. CONCLUSION

This paper presents a new method to prevent inference attacks in large XML databases. We show how ontologies can be used to implement automated attacks on large XML databases and develop methods and techniques to detect such attacks. Although ontological inferences have been studied from the perspective of providing interoperation, the security impacts of these new technologies have not been investigated and there are no tools to prevent these threats.

To the authors'best knowledge, Oxsegin is the first proposal to provide a semantically enhanced XML security framework. This paper adds a new component to the security engine to prevent inference attacks based on correlated data. The Correlated Inference Algorithm computes security violation pointers and their associated confidence level probability. The procedure can be tuned to run at different complexity levels to enhance the efficiency of the model. The main contribution of our model is to be able to handle large amount of semi-structured data that is infeasible by using human experts only.

References

[1] S. Abiteboul, D. Quass, J. McHugh, J.Widom, and J. Wiener. (1997). *The Lorel query language for semi-structured data.* Journal of Digital Libraries. Volume 1

[2] B. Amann, I. Fundulaki, and M.Scholl, et at. (2001). *Mapping XML Fragments to Community Web Ontologies.* Proceedings Fourth International Workshop on the Web and Databases

[3] E. Bertino, S. Castano, E. Ferrari, M.Mesiti. (2000). *Specifying and Enforcing Access Control Policies for XML Document Sources.* WWW Journal, Baltzer Science Publishers, Vol.3, N.3.

[4] E. Damiani, S. De Capitani di Vimercati, S. Paraboschi, and P. Samarati. (2000). *XML Access Control Systems: A Component-Based Approach.* In Proc. IFIP WG11.3 Working Conference on Database Security, The Netherlands.

[5] P. Devanbu, M. Gertz et al. (2001). *Flexible authentication of XML documents.* ACM Conference on Computer and Communications Security.

[6] F. Dridi and G. Neumann. (1998). *Towards access control for logical document structure.* In Proc. of the Ninth International Workshop of Database and Expert Systems Applications, pages 322–327, Vienna, Austria.

[7] M. Erdman, S.Decker. Ontology-aware XML Queries. http://www.aifb.uni-karlsruhe.de/ mer/Pubs/semantic-xql.webdb00.pdf

[8] M. Erdman and R. Studer. (to appear). *How to Structure and Access XML Documents with Ontologies.* Data and Knowledge Engineering, Special Issue on Intelligent Information Integration

[9] M. Erdman and R. Studer. (1999). *Ontologies as Conceptual Model for XML Documents.* Proc. of the 12-th Workshop for Knowledge, Acquisition, Modeling and Management. Banff, Canada.

[10] A. Gabillon and E. Bruno. (2001). *Regulating Access to XML Documents.* In Proc. IFIP WG11.3 Working Conference on Database Security.

[11] T.R. Gruber. (1993). *A Translation Approach to Portable Ontology Specifications.* Knowledge Acquisition. Vol.6, no.2, pp199-221

[12] S. Jajodia and C. Meadows. (1995). *Inference problems in multilevel secure database management systems.* In Information Security: An integrated collection of essays, pages 570-584, IEEE Computer Society Press, Los Alamitos, C.A.

[13] M. Kifer, Georg Lausen, James Wu. (1995). *Logical Foundations of Object Oriented and Frame Based Languages.* Journal of ACM, vol. 42, p. 741-843

[14] M. Kudo and S. Hada. (2000). *XML Document Security based on Provisional Authorizations.* In Proc. of the 7th ACM conference on Computer and Communications Security, Athens Greece, November.

[15] T. Lee and J. Hendler(2001). *The Semantic Web.* Scientific American.

[16] OIL. *Ontology Inference. Layer.* http://www.ontoknowledge.org/oil/

[17] J. Robie, J. Lapp, and D. Schach. (1998). *XML Query Language (XQL).* Proceedings of the W3C Query Language Workshop (QL-98), Boston.

[18] W3C. (2001). *XML Encryption Requirements.* W3C Working Draft, http://www.w3.org/TR/2001/WD-xml-encryption-req-20011018