International Journal of Engineering Research and Science & Technology

ISSN : 2319-5991
Vol. 3, No. 2
May 2014

www.ijerst.com

Email: editorijerst@gmail.com or editor@ijerst.com

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com
Vol. 3, No. 2, May 2014

Research Paper

ON INFERENCE PROOF VIEW PROCESSING OF XML DOCUMENT FOR HOSPITAL MANAGEMENT SYSTEM

E A Vimal[1], S Keerthana[1*], P M Suganthi[1] and S P Suganyasre[1]

Corresponding Author: **S Keerthana ✉ sekarkeerthu@gmail.com*

In this context of inference proof view processing of XML document in hospital database management aims at treating the inference problem in xml document that will provide some details to the client or other organization using XML schema for representing its data type. We are proposing an algorithm for weakening of the XML document by eliminating the confidential information, inference capabilities, modifying the schema of XML. The weakened XML document, modified schema conforms to the inference-proof viewed of the generated document to the client.

Keywords: XML document, XML schema, Inference control, Inference rule

INTRODUCTION

XML schema defines about the structure of the XML elements in such a way provides the secure communication in which sender can describe data in which receiver would understand. In this context, XML schema is used for XML document which is the earliest data type description for XML document. Using XML schema, the XML document will get its most efficient user defined element description. We have to control a client's options to infer information from observing XML documents, either directly by accessing them or parts of them or indirectly by receiving answers to queries. If a node is sensitive it must be eliminated from the answers to queries. XML document generated by a hospital to store the name, age, diseases, treatments, and the names of the doctors for every patient. The hospital wants to share this document with some organizations for some reasons. The client should never be able to obtain any information that would violate the confidentiality requirements. The process of inference-proof viewed is done carried out from the algorithm in the following steps as input: XML document, inference rules, schema of the XML document, potential secrets. Thus, the output of the algorithm should be the weakened XML document which the client or other organization

[1] Kumaraguru College of Technology, Coimbatore, India.

doesn't able to infer any of the other details from the hospital database. The aim of inference control is to prevent an unauthorized client from inferring sensitive information, whether directly or indirectly.

XML DOCUMENT

An XML document is a finite, unranked, unordered and labeled tree T = (N_E, A, n_r, child, descendant, f_l; f_s) such that the following conventions and properties hold:

1. NE is the finite set of elements and $n_r \in N_E$ is a distinguished root element. f_l: $N_E \rightarrow \Sigma$ is a function assigning a label to an element.
2. A is the finite set of functions from $N_E \rightarrow \gamma$, where each function a ∈ A has a distinct attribute name from Σ, by the injective function fs: A → Σ Given an element n ∈ NE and a function a ∈ A, if n is associated with with attribute f_s(a), then a(n) is the attribute value, otherwise ∈ I.
3. Child C (N_E x N_E) is a binary relation between a ("father") element and its "children" in the document.
4. Descendant is the transitive closure of child.

We choose to formalize XML documents as unordered trees only for the sake of simplicity of our discussion, and the value of an element is modeled as an attribute, with a particular name, e.g., "value," of the element. If (n,n^1) ∈ T: descendant, then we can construct a path instance in T from n to n^1.

XML DOCUMENT AND ITS DEFINITION

XML Schema

XML Schema specifies the structure to the XML document. It defines legal building blocks to the XML document .It provides allowable contents and checks for correctness of data. The use of XML schema over DTD is the use of namespace. Here in schema the namespace is specified in xmlns and can avoid naming conflicts.

Path Instance

Given an XML document T, a path instance ϕ of T is a sequence of T: child-connected elements in T, i.e., more formally, ϕ = $n_1/n_2/\ldots./n_m$ satisfies the following properties: for all i ∈ [1, m]: n_i ∈ T:N_E, and for all i ∈ [1, m-1] : (n_i,n_i+1) ∈ T.child.

In this definition, the functions P and Q serve to confine the possible sub elements and attributes, respectively, of an element in a conforming document.

Path

A path is a structure given by ψ = (PN_E, PN_A, child, descendant, pnr, pnl, fe, BackBone)such that the following properties hold:

1. PNE, PNA are finite and disjoint sets of path nodes, $pn_r \in PN_E$ is a distinguished root path node, $pn_l \in (PN_E \cup PN_A)$ is a path node.
2. Child PN_E x (PN_E U PN_A) is a binary relation between a path node and its children, representing a tree.
3. Descendant is the transitive closure of child (PN_E X PN_E).
4. f_e is a function returning the string expression of a path node and satisfying the following constraints:
   a. For each $p_n \in PN_E$, we have $f_e(p_n) = \sigma$, where σ ∈ Σ.
   b. For each p_n ~ PN_A, we have $f_e(p_n)$=@ σ # v ,where and σ ∈ Σ and v γ.

5. Backbone (PN_E U PN_A) describes the backbone of ψ, which satisfies the following constraints:

   a. pn_r, $pn_l \in$ BackBone;

   b. for each $p_n \in$ BackBone except pn_r, there is one and only one path node $pn^1 \in$ BackBone such that (pn^1, pn) $\in$ child;

   c. for each $p_n \in$ BackBone except pn_l, there is one and only one path node $pn^1 \in$ BackBone such that (pn, pn^1) $\in$ child.

EXAMPLE

Patient[@pname#Bob]/disease/@dname #Hypertension is a string expression representing the path such that

- Ψ.PNE = {pn1, pn3};Ψ.PNA ={pn2, pn4},
- Ψ.child ={(pn1,pn2); (pn1, pn3); (pn3, pn4)},
- Ψ.fe(pn1)=patient; Ψfe(pn2)=@pname#Bob; Ψ.fe(pn3)=disease; Ψ.fe(pn4)=@dname# Hypertension;
- Ψ.pnr = pn1 and Ψ.pnl = pn4,
- Ψ.BackBone ={pn1, pn3, pn4}.

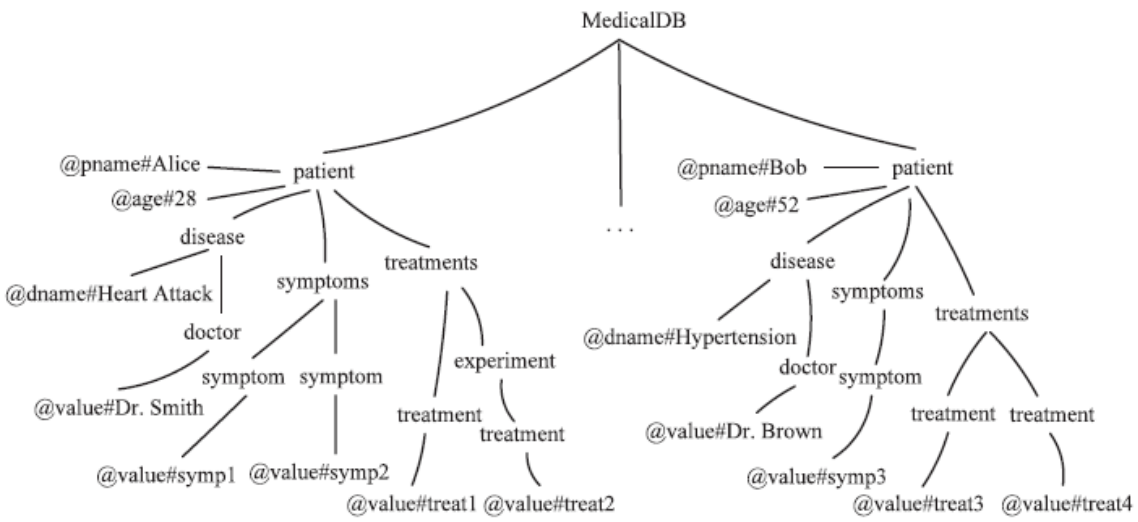
To increase the flexibility of information description, we also allow replacing a label or a string value in the string expression of a path node with a variable. Formally, there are two kinds of variables in paths. The domain of the first kind of variables is Σ, and we call them "label variables." The domain of the second kind of variables is γ, and we call them "string-value variables". In Figure 1, If variable x occurs in the string expression of a path like @ σ #x, then x must be a string-value variable; otherwise, x is a label variable. For example, Medical/patient [@pname# x] represents a path with the string-value variable x, and we can assign a string value, such as Alice or Bob, to x. Given a path with variables, a variable assignment of the path is a replacement of all the variables in the path with some specific labels and string values. As stated above, a path is a structure depicting certain relations between some labels and string values. If there exists a fragment (a group of elements along with their attributes and their relations) in an XML document such that the piece of information described by the fragment (the labels of these elements, the values of their attributes and the relations between these elements in the fragment) is just the information depicted by the path, then the path agrees with some information in the XML document, or the XML document satisfies the path.

RELATED WORK

The technique of sending xml document with incomplete information (Yang and Li, 2004): The secure parsing of XML documents without any information leakage. In this, they specified what information is sensitive and should be protected. The data inference using common knowledge can be represented as semantic constraints. Compute all possible inferable data. We show that if a partial document is published carelessly, users can use common knowledge to infer more data, which can cause leakage of sensitive information. The goal is to protect such information in the presence of data inference with common knowledge. Consider that the hospital plans to provide the data to another organization to conduct related research. Some data is sensitive and should not be released. In particular, the hospital does not want the department to know the disease of patient Alice for some reason. But if it is well known that through symptoms present in the document one can infer a disease and it should be avoided.

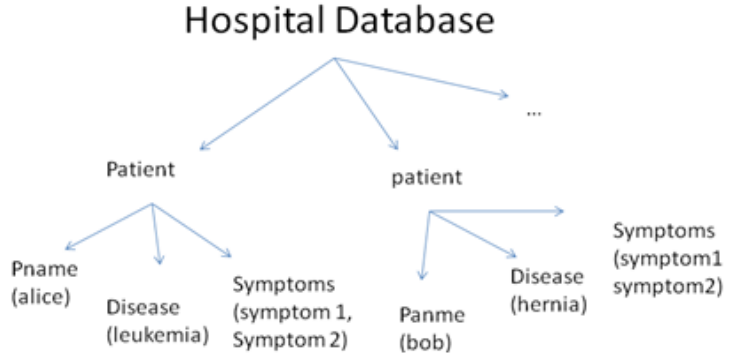
In Figure 2 the functional dependency in the

Figure 1: XML Document with a Tree Structure



XML document (Marcelo Arenas and Leonid Libkin) needs to be analyzed and eliminated. The introduction of the concept of functional dependency for XML and define its semantics relational representation of XML. We then dene an XML normal form, XNF that avoids update anomalies and redundancies. Functional dependency in XML document is avoided because of well designed XML schema for the xml document of the database to define its data type structure. Functional Dependency used to infer the other details from the document and it should be avoided by eliminating the paths when parsing over the xml document. Consider that the hospital database contains the pname, age, disease, symptoms, etc. If the document is sent to other organization in which they does not needs disease then we can eliminate the disease but they can infer that a patient that kind of disease through the symptom.symptom1, symptom2—

Figure 2: Inference Capabilities in XML Document



>{disease}. So this kind of functional dependency should be avoided since they lead to the inference of the other information from the document.

GENERATING XML SCHEMA

We propose a formal notion of an inference-proof view of an XML document, to meet the requirements of our goal of effective inference control under the inference capabilities of a client. We enhance the algorithm for generating an inference proof view by using the XML Schema over Document Type Definition. The inputs include an XML document and its schema, potential secrets, and the inference capabilities of a (group of) client(s). We also introduce variables into the inference capabilities to increase their expressiveness and analyze the high complexity of the algorithm because of the variables. Then, we discuss and use experiments to show in which situation the complexity of the algorithm would be decreased. This concept focuses on assigning a score that represents the sensitivity level of the data exposed to the user and by that predicts the ability of the user to maliciously exploit this data.

Advantages of using XML schema over DTD

1. The major advantage of schemas is their ability to more strongly type the data in XML documents. Schemas are described using XML instead of the archaic form used by DTDs. Schemas also provide a richer approach to describing complex XML types.
2. Document structure can be described more accurately with schemas as well, using features such as minOccurs and maxOccurs to specify the number of times an element can occur within a particular context.
3. The Introduction of Algorithm Complexity Analysis to increase their expressiveness and analyze the high complexity of the algorithm because of the variables. Then, we discuss and use experiments to show in which situation the complexity of the algorithm would be decreased.

Inference Control

The aim of inference control is to prevent an unauthorized client from inferring sensitive information, whether directly or indirectly. Previous research has shown that to achieve this goal for an information system, e.g., a relational database, it is necessary to control the inference channels in the information set of the system. As XML documents have more complicated structures than the relations, there may exist more intricate inference channels in XML documents. The intuitive purpose of inference control for XML documents then is the following: Given an XML document, some information assumed to be a priori known to a client and a declaration of pieces of information assumed to be sensitive and thus to be kept confidential, the client can never conclude that the document contains any sensitive information through any inference channel.

COMPLETENESS AND INCOMPLETENESS

The first feature distinguishes how a client relates the information set managed by an information system to the actual situation in (what he perceives to be) the “real world.” Assuming completeness, the client considers the information set to contain all the information that is true in the (relevant part of the) real world, and accordingly he holds those information not contained in the information set to be false. However, in many applications, a client will restrict himself to assume incompleteness: The

information set managed by the information system contains information considered to be true, while nothing is postulated about information that is not contained in the information set. For instance, the information set given by the XML documents maintained by a hospital, like the one e.g., the hospital may not know all the diseases of all patients .In this paper, we focus on inference control for XML documents with incomplete information, or incomplete XML documents, with the following properties:

- For some of its elements, an XML document may not include the attributes that are related to the information set given by the XML document but unknown to or hidden intentionally by the information system managing the XML document.
- For some of its elements, an XML document may not include the child or descendant relations that are related to the information set given by the XML document but unknown to or hidden intentionally by the information system managing the XML document.
- For some of its path instances, while keeping the descendant relations of the elements in the path instances, an XML document may not include some intermediate elements in some path instances that are related to the information set given by the XML document but unknown to or hidden intentionally by the information system managing the XML document. Therefore, any sub element of an element in an incomplete XML document might actually just be a descendant of the element in the "real world."

Inference Capabilities

Inference capabilities describe what a client already knows about an XML document as a means to infer further information. These capabilities may consist of integrity constraints, semantic constraints, and other a priori knowledge of the client on the information set. For inference control on an XML document, the inference capabilities of the client comprise the disclosed XML Schema (the one disclosed to the client), which defines the integrity constraints on the XML document, authorized XML data from the XML document, and the semantic constraints on the XML document. Therefore, we model the inference capabilities of a client by known information entailed by a XML Schema and the XML document, and the semantic implications between pieces of information contained in the XML document.

Confidentiality Policies

Confidentiality policies tell what is protected by inference control. In this paper, we protect the fact that an XML document contains some particular pieces of information, and we call those pieces potential secrets.

Interaction Sequences

The interactions considered in this paper contain the submissions of queries and the answers to these queries. The client could continue submitting queries over the XML documents, and the answers to these queries should be always controlled according to the pertinent confidentiality policy and authorization policy.

Alternative XML Document

Although the actual stored XML document may contain some potential secrets, the inference control should make the client believe in the possibility that there is an alternative XML document that does not contain any potential secret. In particular, the client should not be able to distinguish the actual document from the

alternative one according to his queries. Hence, inference control aims at ensuring the existence of an alternative XML document or even at constructing such an alternative XML document as an inference-proof XML view on the actual XML document.

Information Weakening

If the alternative XML document not containing any potential secret is an inference-proof view on the actual XML document, then the answers to the queries do not need to be distorted when these queries are evaluated over the alternative XML document. We use weakening to construct the alternative XML document, that is, all the information contained in the alternative XML document must be contained in the actual XML document, but some pieces of information of the actual XML document might not be part of the alternative XML document. The possible methods of weakening an incomplete XML document include:

- Delete an attribute from the XML document;
- Delete an element from the XML document and, if applicable, let its sub elements become sub elements of its parent.

Based on the features discussed above, the idea of inference control on XML documents presented in this paper is to construct both an inference-proof view by weakening the actual XML document and a new XML Schema to which this inference-proof view conforms. If only the new XML Schema instead of the declared Schema of the actual document is disclosed to the client and the inference proof view neither explicitly contains any potential secret nor implies any potential secret as implicit information under the inference capabilities of the client, then the information system can evaluate the client's queries over the inference-proof view according to an authorization policy.

Inference Rules

Inference capabilities are identified using functional dependency. Using this functional dependency (Barcelo *et al*., 2010) the inferring elements are identified and when sending the data to the client potential secret and the inferring elements are hidden. Thus the inference problem is resolved.

For example:

- LHS $\rightarrow$ RHS

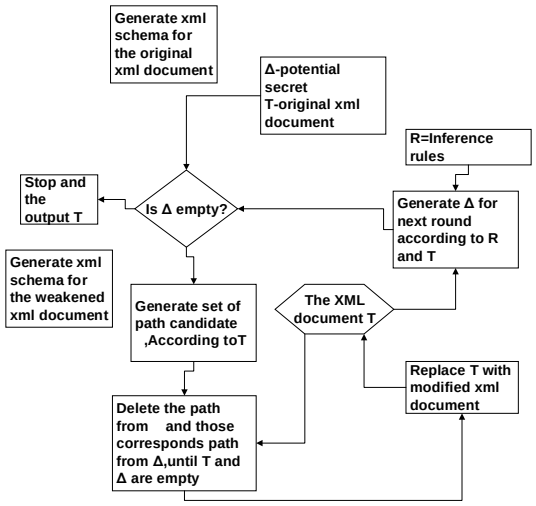
1. {@pid, treatment}®{@dname}
2. {@pid, symptom}®{@dname}
3. {@pid, doctor}®{@dname, treatment}
4. {@pid, address}®{@pname}

where 'dname' is disease name, 'pname' is patient name and 'pid' is patient id.LHS are inferring field and RHS are potential secret.

GENERATING AN INFERENCE-PROOF VIEW

In our algorithm for generating an inference-proof view, we will eliminate all the confidential information and some information that can be used to infer some confidential information from an XML document by weakening the document step by step. The inputs of our algorithm include an XML document, the XML schema of the XML document, a finite set of potential secrets, and a finite set of inference rules on the XML document. The outputs of our algorithm include an inference-proof view of the input XML document w.r.t. the two input sets and Xml schema to which both the inference-proof view and the input document conform. The returned XML schema might be different from the input XML schema.

Figure 3: Algorithm to Generate the Inference Proof View



ALGORITHM

Procedure inferenceviewgen()

Input: the set R of inference rules, the set S of potential secrets, the inference-closed XML document T w.r.t. R, the XML Schema.

Output: the weakened XML document, the modified XML Schema for the weakened XML document. Ω

1: $\Delta \leftarrow S$

2: loop

3: $X \leftarrow \phi$

4: for each path set $\delta \in \Delta$ do

5: p←func_obtaincandidate(δ){The generation of path candidates from δ}

6: for each path $p \in P$ do

7: proc_insertpath ($\in$,p) {inserting into fI}

8: end for

9: end for

10: for each path set $\gamma \in$ f do

11: proc_refinpathset (γ){ refine γ that should avoid containing more than one path candidate in path set Δ }

12: end for

13: loop

14: if $\Delta == \phi$ then

15: break

16: end if

17: Select a path set γ from f with largest cardinality in f I

18: proc_weakendocument (T,D,γ) {weaken T for all path in γ and modify D for weaken document}

19: f←f–{ γ }

20: for each path $p \in \gamma$ do

21: δ←func_findpath(Δ,p){Finding the path set from which p is generated in Δ }

22: $\Delta \leftarrow \Delta - \{\delta\}$

23: for each path candidate p' (from δ except p do

24: proc_delpath(f,p'){delete p' from I }

25: end for

26: end for

27: end loop

28: for each rule $r \in R$ do

29: R' ← fun_enumassignment(r){R' is the set of rules after grounding r }

30: for each $r' \in R'$ do

31: if func_violated(T,r')is true then

32: $\Delta \leftarrow \Delta$ U{r'.conditionpart}

33: end if

34: end for

35: end for

36: if $\Delta == \phi$ then

37: break

38: end if

39: end loop

MODIFYING XML SCHEMA FOR INFERENCE-PROOF XML VIEW

The idea of generating the new XML Schema is to modify some expressions of the original XML schema after every new kind of modification to the content of the XML document. Suppose the algorithm needs to weaken the XML document T for a path and the algorithm will modify some expressions of the original XML Schema D according to the type of ψ.pnl.

CONCLUSION

The dataset values are generated in to a XML document. The xml document of the hospital database is provided with its data type structure in XML schema. The potential secret of the medical database is to be identified. The inference rule based on the functional dependency of the XML document is identified in such a way the client does not able to infer any of the other information from the database provided to them. From the generated xml document, XML schema, potential secrets, inference rule the XML document needs to be parsed over by the node paths in which it needs to be eliminated. The result obtained from it should be in such a manner that it is a weakened XML document of the hospital database in which it similar to the modified schema. Thus, the result provided is not been able to infer any of the other details from the document which is provided to other organization.

REFERENCES

1. Arenas M and Libkin L (2004), "A Normal Form for XML Documents," *ACM Trans. Database Systems,* Vol. 29, pp. 195-232.
2. Barcelo P, Libkin L, Poggi A and Sirangelo C (2010), "XML with Incomplete Information," *J. ACM*, Vol. 58, No. 1, pp. 4.1-4.10.
3. Clark J and DeRose S (1999), "XML Path Language (XPath) Version 1.0.," W3C Recommendation, November.
4. Designing Functional Dependencies for XML" Mong Li Lee, Tok Wang Ling, and Wai Lup Low.
5. Joachim Biskup and Lan Li (xxxx), "On Inference-Proof View Processing of XML Documents", Dependable and Secure Computing, *IEEE Transactions*, Vol. 10, No. 2.
6. Yang X and Li C (2004), "Secure XML Publishing without Information Leakage in the Presence of Data Inference," Proc. 30th Int'l Conf. Very Large Data Bases (VLDB), M A Nascimento, M T O ¨ zsu, D Kossmann, R J Miller, J A Blakeley, and K B Schiefer (Eds.), pp. 96-107, 2004.

International Journal of Engineering Research and Science & Technology
Hyderabad, INDIA. Ph: +91-09441351700, 09059645577
E-mail: editorijlerst@gmail.com or editor@ijerst.com
Website: www.ijerst.com