International Journal of Internet, Broadcasting and Communication Vol.13 No.1 82-89 (2021)
http://dx.doi.org/10.7236/IJIBC.2021.13.1.82

IJIBC 21-1-10

Secure Healthcare Management: Protecting Sensitive Information from Unauthorized Users

Hye-Kyeong Ko

Assistant Professor, Department of Computer Engineering, Sungkyul University, Korea
ellefgt@sungkyul.ac.kr

Abstract

Recently, applications are increasing the importance of security for published documents. This paper deals with data-publishing where the publishers must state sensitive information that they need to protect. If a document containing such sensitive information is accidentally posted, users can use common-sense reasoning to infer unauthorized information. In recent studied of peer-to-peer databases, studies on the security of data of various unique groups are conducted. In this paper, we propose a security framework that fundamentally blocks user inference about sensitive information that may be leaked by XML constraints and prevents sensitive information from leaking from general user. The proposed framework protects sensitive information disclosed through encryption technology. Moreover, the proposed framework is query view security without any three types of XML constraints. As a result of the experiment, the proposed framework has mathematically proved a way to prevent leakage of user information through data inference more than the existing method.

Keywords: *Information Inference, XML Constraints, Privacy Preserving, Query-View Secure*

1. Introduction

The amount of data used in digital form continues to increase and the data can be used over the network. In most practical cases complex constraints of trust and secrecy exist between cooperation and competing groups. In many cases data can be transmitted when there are no security or confidentiality issues to potential recipients [1, 2]. Recently, applications are increasing the importance of security for published documents. In these applications, data owners publish data to other users on the web [3]. If the data owner publishes the data insecurely, the user can deduce information from the published document that is not authorized by reasoning. Therefore, a problem arises that it is possible to infer sensitive information such as patient information of a hospital from other users. The secure healthcare management framework proposed in this paper uses encryption to protect sensitive information from unauthorized users.

1.1 Data Inference by XML Constraints Figures

An XML document is defined by elements and attributes [4]. Element can contain other element and be nested at a depth, thus creating a hierarchical graph-based structure. An element are separated by two tags: the start

Manuscript Received: November. 22, 2020 / Revised: November. 27, 2020 / Accepted: December. 2, 2020
Corresponding Author: ellefgt@sungkyul.ac.kr
Tel: +82-31-467-8113
Assistant Professor, Department of Computer Engineering, Sungkyul University, Korea

tag, at the beginning of the element, and of the form <*tag-name*>, and the end tag, at the end of the element [4], and of the form </*tag-name*>; in both cases, "*tag-name*" indicates the type of element (*markup*). Attributes can have different types and be used to specify element identifiers, additional information about the element in the document [5]. An illustrative XML document for modeling hospital data is shown in Figure 1. The *hospital* data contains one or more *patients*, and one or more additional elements (*name*, *disease*, and *ward*)

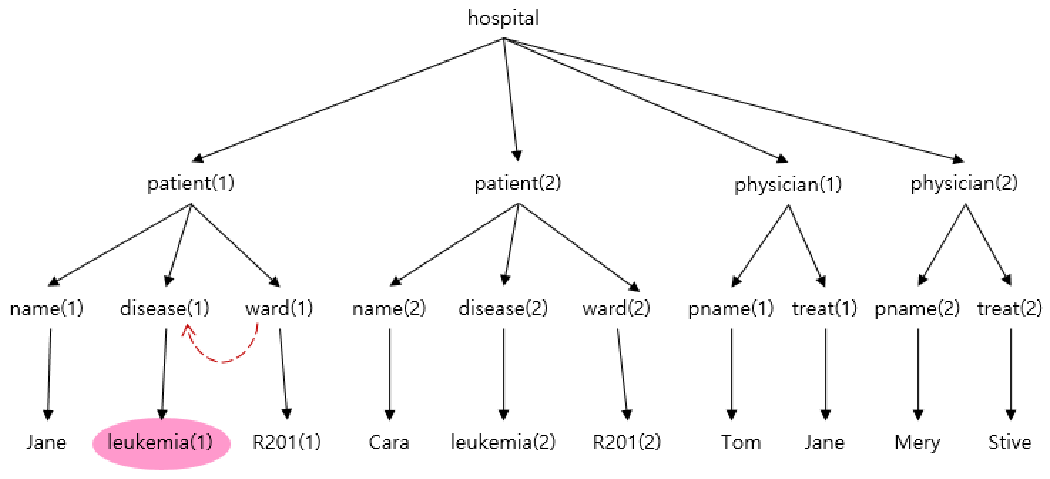

Figure 1. Original XML document of hospital. The shaded subtree is sensitive data [9]

Hospital data also contains *physicians*. Throughout the dissertation, a node uses a graphical representation of an XML document that represents an element and an edge represents a relationship between nodes. If it is known that *patients* in the same ward have the same *disease*, this information can be used by department users to infer that *Jane* has *leukemia*. Therefore, users can deduce information about *Cara* patients residing in *W201* Ward. Hiding the shaded *leukemia* branch cannot protect the sensitive information due to this common knowledge.

1.2 Three Types of XML Constraints

There are various ways in which users can infer information by exploiting common sense inference represented as XML Constraints [6]. XML Constraints specify the relationships that should be satisfied by the nodes in the document. XML DTD and XML Schema can define XML Constraints. XML Constraints are represented in the form "*conditions→facts*." If the conditions are satisfied in an XML document, the facts must also be true for the document. The three types of XML Constraints [6] are as follows.

Child Constraint: This is represented as *P→P/P*′ and indicates that every node of type *P* must have a child of type *P*′.

Descendant Constraint: This is represented as *P→P//P*′ and indicates that every node of type *P* must have a descendant of type *P*′.

Function Dependency Constraint: This is represented *P/P1→P/P2,* where *P, P1*, and *P2* are finite non-empty subtrees that conform to the document. This indicates whether the *P1* path has the same value for both subtrees *T1* and *T2* that match the *P/P1* path,

- Both are non-null equal subtrees that match *P/P2*; or
- Neither matches *P/P2*.

For example, the child constraint "*//patient→ //patient/ward*" indicates that each *patient* element must have a *ward* element as a child. Descendant Constraint "*//patient→ //patient//leukemia*" indicates that each patient element must have a *leukemia* element as a descendant. The function Dependency Constraint "*//patient/ward→ //patient/disease*" indicates that if two subtrees have the same "*//patient/ward*" value, they should have the same "*//patient/patient*" value. In other words, patients in the same *ward* have the same *disease*.
In the proposed framework, each sensitive information in a document is separated and encrypted, and the encrypted nodes are stored in a different structure. It is grouped into an encrypted structure index representing the structure information of the encrypted nodes. The structure of this paper is as follows. Section 2 describes the existing database security model as a related study, and Section 3 describes in detail the node encryption rules and the proposed framework proposed in this paper. Section 4 discusses the security of the proposed method through mathematical evaluation. Finally, the conclusions are described and the future studies of this study are described in Section 5.

2. Related Works

Database security has been studied extensively in the past. Several recent studies have suggested methods for data publishing [7, 8]. Miklau [9] studied an encryption-based approach to access control to published XML documents. This method uses the specification of access control policy, and designed a framework using extensions of XQuery to define sensitive data in published documents.

Yang [6] studied common knowledge by expressing semantic XML constraints. In this method, the user can infer the data as three types of general XML Constraints and developed a new algorithm to find partial documents of a given single XML document without information leakage. In this paper, unlike Yang, the user cannot detect the encrypted part in the document, and the encrypted node can only be accessed by authorized users. Lee [10] proposed the concept of Query-aware decryption to efficiently process queries for encrypted XML data. In Query-aware decryption, only the part that contributes to the query result can be decrypted.

3. Proposed Framework for Secure Data Management

In data publishing, once the data owner has published data, he/she loses control over the data [11]. Different users might have different viewing rights for different parts of the same document. The data owner maintains and manages keys and grants them to specific users. The main problem with data publishing is that a user might infer the unauthorized sensitive information using XML Constraints [6]. Therefore, we propose a framework for secure data publishing of XML documents that uses encryption techniques. In the proposed framework, each user is required to register during the registration phase. The data owner starts by annotating the XML document according to the access rights of the users. In this registration phase, the authorization record returns specific information, called keys, to the user; such keys, are necessary to decrypt the relevant parts of the XML source according to the user's access rights. Authorized users can access data, depending on the keys they possess. In our framework, users do not need to decrypt the entire document; they can selectively access those parts of the published document that are predetermined by the policy evaluator shown in Figure 2. In the following paragraphs, we introduce the basic components of the policy evaluator.

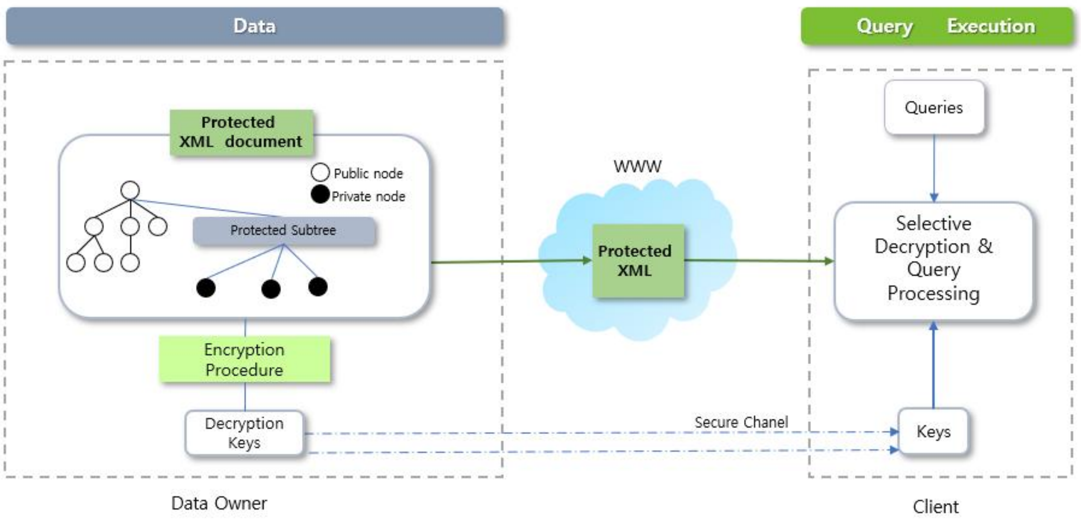

Figure 2. Example of proposed framework for secure healthcare management

3.1 Node Encryption Rules

In our framework, the document is partitioned into public and sensitive nodes by the following three encryption rules:

Child Encryption Rule: If a node satisfies the child constraint ($P \rightarrow P/P'$), every child of the node must be encrypted.
Descendant Encryption Rule: If Q and Q' satisfy the descendant constraint ($P \rightarrow P//P'$), Q and Q' must be Encrypted.
Function Dependency Encryption Rule: If Q, $Q1$, and $Q2$ satisfy the function dependency constraint ($P/P1 \rightarrow P/P2$) such that, Q, $Q1$, and $Q2$ must be encrypted.

3.2 Protected Subtree

In our framework, the system encrypts the sensitive parts of an XML document. Encrypting such sensitive parts means that the selected parts of the original document structure are hidden from unauthorized users. The key idea of our framework is to prune sensitive nodes from the original document tree and encrypt each sensitive node individually. Sensitive nodes are selected according to the authorization record; and moved to the protected subtree.

3.2.1 Protected Labeling Schemes

In our framework, the document is partitioned into public and sensitive nodes in order to support secure data publishing. The principal problem with query processing is to effectively find the positions of decrypted sensitive nodes. After encryption, all sensitive nodes are pruned from the original XML document and therefore the positions of these nodes in the document must be remembered. A secure and efficient labeling scheme is required for protecting the structure information of the sensitive nodes and representing their positions. In this paper, we extend IBSL approach [12], to protected IBSL, which takes advantage of the lexicographical order of binary strings [12]. When sensitive nodes are pruned from the original document, an unauthorized user should not be able to infer other structural information by using the labels of public nodes.

Protected IBSL approach can hide the label values, assigned to sensitive nodes, and its labeling effectively separates the public nodes by Definitions 4.1 and 4.2.

Definition 3.1. (*label for public nodes*) *For a non-root node P, the* i^{th} *child node of P is labeled with label(P).*1*i0, where label(P) is the label of P. The left side of delimiter (.) is called a prefix-label and the right side is called a self-label.*

In protected IBSL, as a breath-first traversal of the XML tree is performed, each node is assigned a label. Figure 3 shows an example of protected IBSL. The root node is labeled with *null*. For other nodes, the label is a concatenation of its *prefix-label* (i.e., the parent's label) and its *self-label*. For example, for the two nodes *10* and *10.110*, *10* is the parent of *10.110* because *10* is a prefix-label of *10.110*.

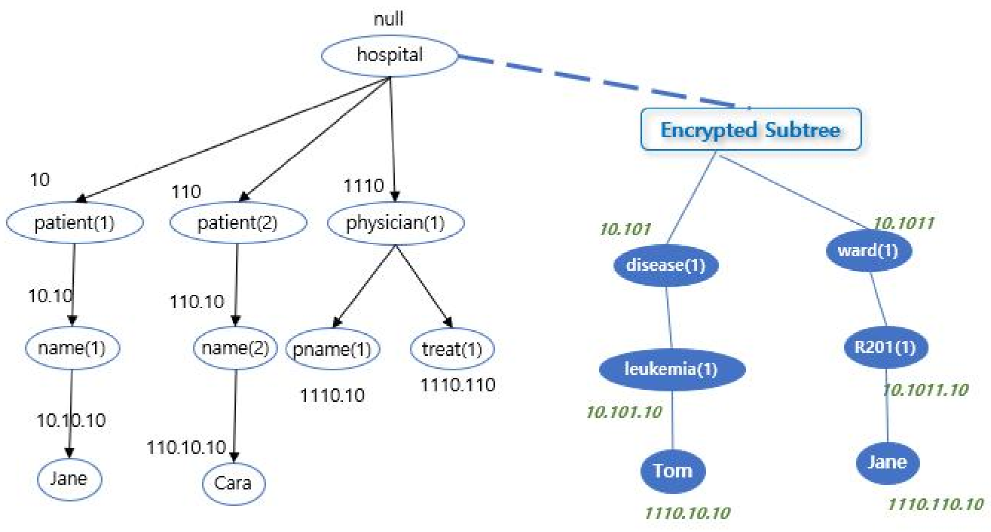

Figure 3. Public document obtained from after removing sensitive nodes

Definition 3.2 (*label for sensitive node*) *Sensitive node S is labeled using the following labeling rules, where len(P) denotes the bit length of P's label and* $\oplus$ *denotes concatenation. In the following, label(P) is assumed to be the parent of S.*

1. *Label sensitive node S before the leftmost sibling node*:
   S = label (*P*)*.Pfirst* $\oplus$ *sibling* $\oplus$ *0*
2. *Label sensitive node S between two public nodes*;
   If len(Pleft) $\leq$ *len(Pright), S = label(P).Pright* $\oplus$ *0*
   If len(Pleft) > len(Pright), S = label(P).Pleft $\oplus$ *1*
3. *Label sensitive node S after the rightmost sibling node*;
   S = label(P).Pleft $\oplus$ *1*
4. *Label sensitive node S without the sibling node*;
   S = label(P).1i0

Identifying the relationships among nodes is essential in finding the correct positions of the encrypted nodes in the protected information set. The labeling scheme can be utilized to identify the relationships among nodes.

To determine such relationship, the following rules are used [12]:

Ancestor-Descendant Relationship: If the label of node *x* is a prefix string of the prefix-label of node *y*, *x* is an ancestor of *y*.
• **Parent-Child Relationship**: If the label of node *x* is equal to the same as the prefix-label of node *y*, *x* is the parent of *y*.
• **Sibling Relationship**: Considering the adjacent binary strings S_{left} and S_{right}, S_{left} is said to be precompiled to be the same as S_{right} if the string is the same, otherwise the following procedure is performed to compare them [12, 16]:

4. Proof of Security

In this section, we prove that it is not possible for a user to infer information that the user is not authorized to know based on the information the user is authorized to know [13-15]. Because the proposed framework publishes the public document and protected information set, we need to ensure that the user cannot infer unauthorized information from the encrypted structure index. For the proof, we first assume that the Miklau's scheme [9] is Query-View Secure (QVS) and prove that our method is QVS.

Definition 4.1 (*Query-View Secure*) Suppose sensitive data in a database are represented by query *Q*, and a set of views *{Vi}* is published. If the difference between the probability of guessing the result of *Q* with *{Vi}* and that of guessing the result of *Q* without *{Vi}* is zero, a query *Q* is said to be QVS with respect to the set of views *{Vi}* [9].

Suppose $KEY_{GRANTED}$ is the set of the keys possessed by a user, and KEY_Q is the set of keys necessary for processing query *Q*. The elements encrypted using KEY_Q correspond to *Q* in Definition 4.1. The union of the encrypted elements that use the sets of key $KEY_{Vi} \subseteq KEY_{GRANTED}$ correspond to *{Vi}* in Definition 4.1.

Theorem 4.1 The proposed framework is QVS without the three types of XML Constraints.

Proof. Let *T* be the set of all elements in the encrypted structure index. Let $T_{Vi} \subseteq T$ be the set of elements encrypted using the set of keys $T_{Vi} \subseteq KEY_{GRANTED}$, and $T_Q \subseteq T$ be the set of elements encrypted using the set of keys $T_Q \subseteq T_{GRANTED}$. There are three cases: (case 1) $T_Q \cap T_{GRANTED} = \emptyset$, (case 2) $T_Q \cap T_{GRANTED} \neq \emptyset \wedge T_Q \subseteq T_{GRANTED}$, and (case 3) $T_Q \cap T_{GRANTED} \neq \emptyset \wedge T_Q \subseteq T_{GRANTED}$. By the elements encryption method of our framework, in case 1, there is no query result for the user and T_Q is QVS with respect to T_{Vi}. In case 2, the query results are all authorized for the user and T_Q is QVS with respect to T_{Vi}. In case 3, the user cannot access the unauthorized part information because the keys used for encrypting that part do not intersect with $T_{GRANTED}$, and T_Q is QVS with respect to T_{Vi}.

Theorem 4.2 The proposed framework is QVS without the three types of XML Constraints.

Proof. Let *P* be the public document and *{Si}* be the set of all sensitive nodes in the protected information set. *{Pi}* is not encrypted and *{Si}* is encrypted using the set of keys $KEY_{Si} \subseteq KEY_{GRANTED}$ and the node encryption rules against the three types of XML Constraints. The keys used for encrypting elements in the XML document are the same as those used for encrypting the encrypted structure index. Suppose that, *S*1 and *S*2 are two nodes under the child constraint. Sensitive elements *S*1 and *S*2 must then be encrypted according to the child encryption rule. The elements in *P* consist of the set of elements *{Pi}* accessible to unauthorized users, and those in *S* consist of elements *{Si}* that might not be accessible to unauthorized users. Thus, *S*1 and *S*2 are secure by the node encryption rules in Section 3.1. Similarly, for the other two types of XML Constraints, the relevant nodes are secure.

5. Conclusion

In this paper, we proposed the framework that protect sensitive information to be published. The proposed framework uses an encrypted subtree to allow the XML document to be published without inferring information by user inference. For safe publication, the structural information of the original XML document was summarized and saved. If the data owner publishes data insecurely, the user can infer unauthorized information by referring to other information in the published document. In the previous data posting work, we considered the specification of access control policy and efficient query processing for encrypted XML data. In this paper, we proposed a new framework to protect sensitive information disclosed through encryption technology. The proposed framework is query view security without any three types of XML constraints. Also, the proposed framework is safe when it comes to three types of XML constraints.

For future research, research on subtree-based secure group key management for multi-level access control and research on how to handle labels when access control policies are updated are needed.

References

[1] W. S. Ng, K. L, Tan, and A. Zhou, "A P2P-based System for Distributed Data Sharing," in Proc. 19th International Conference on Data Engineering, pp. 633-644, 2003.
DOI: https://doi.org/10.1109/icde.2003.1260827.

[2] M. Stonebraker, P. M. Aoki, W. Litwin, A. Pfeffer, A. Sah, J. Sidell, Staelin, and A. Yu, "Mariposa: A Wide-area Distributed Database System," VLDB Journal, Vol. 5, No. 1, pp. 48-63, 1996.
DOI: https://doi.org/10.3850/978-981-08-7300-4_1005.

[3] Z. G. Ives, A. Y. Lyer J. Madhavan, R. Pottinger. S. Saroiu, I. Tatarinov, S. Betzler, Q. Chen, E. Jaslikowska, J. Su, and W. Yeung, "Self-organizing Data Sharing Communities with SAGRES," in Proc. 2003 ACM SIGMOD International Conference on Management of Data, p. 582, June, 2000.
DOI: https://doi.org/10.1145/342009.335492.

[4] T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, and F. Yergeau, "Extensible Markup Language (XML) 1.0 (forth ed.)," W3C, 2006, http://www.w3.org/TR/REC-xml.

[5] N. R. Adam and J. C. Wortman, "Security-Control Methods for Statistical Databases," *ACM Computing Surveys*, Vol. 21, No. 4, pp. 515-556, 1989.
DOI: https://doi.org/10.1145/76894.76895.

[6] X. Yang and C. Li, "Secure XML Publishing without Information Leakage in the Presence of Data Inference," in Proc. 30th International Conference on Very Large Data Bases, pp. 96-107, 2004.
DOI: https://doi.org/10.1016/b978-012088469-8.50012-7.

[7] A. Brodskyand, C. Farkas, and S. Jajodia, "Secure Databases: Constraints, Inference Channels, and monitoring disclosures," IEEE Transactions on Knowledge and Data Engineering, Vol. 12, No. 6, pp. 900-919, 2000.
DOI: https://doi.org/10.1016/b978-012088469-8.50012-7.

[8] S. Castano M. G. Fugini and G. Martella, "Database Security," Addison-Wesley & ACM Press, 1995.

[9] G. Miklau and D. Suciu, "Controlling Access to Published Data using Cryptography," in Proc. 29th International Conference on Very Large Data Bases, pp. 898-909, 2003.
DOI: https://doi.org/10.1016/b978-012722442-8/50084-7.

[10] J. G. Lee and K. Y. Whang, "Secure Query Processing against Encrypted XML Data using Query-Aware Decryption," Information Sciences, Vol. 176, No. 13, pp. 1928-1947, 2006.
DOI: https://doi.org/10.1016/j.ins.2005.08.001.

[11] E. Bertino, S. Castano, E. Ferrari, and M. Mesiti, "Controlling Access and Dissemination of XML Document," in Proc. 2nd International Workshop on Web Information and Data Management, pp. 22-27, 1999.
DOI: https://doi.org/10.1145/319759.319770.

[12] H. Ko and S. Lee, "A Binary String Approach for Updates in Dynamic Ordered XML Data," IEEE Transactions on Knowledge and Data Engineering," Vol. 22, No. 4, pp. 602-607, 2010.
DOI: https://doi.org/10.1109/tkde.2009.87.

[13] D. Dobkin, A. K. Jones, and R. J. Lipton, "Secure Databases: Protection against User Influence," ACM Trans. Database System, Vol. 4, No. 1, pp. 97-106, 1979.
DOI: https://doi.org/10.1145/320064.320068.

[14] S. H. Yoon, K. S. Lee, C. J. Sang, T. Khudaybergenov, M. S. Kim, D. G. Woo, and J. U. Kim, "Building Control Box Attached Monitor based Color Grid Recognition Methods for User Access Authentication," The International Journal of Internet, Broadcasting and Communication (IJIBC), Vol. 12, No. 2, pp. 1-7, 2020.
DOI: https://doi.org/10.7236/JIJBC.2020.12.2.1.

[15] Y. Seo and Y. Chang, "A Study for Applicating and Introducing the Right to be Forgotten," The Journal of Convergence on Culture Technology (JCCT), Vol. 2, No. 3, pp. 23-28, 2016.
DOI: https://doi.org/10.17703/JCCT.2016.2.3.23.