Mathematical Biosciences and Engineering

MBE, 17(5): 4609–4630.
DOI: 10.3934/mbe.2020254
Received: 19 April 2020
Accepted: 23 June 2020
Published: 2 July 2020

http://www.aimspress.com/journal/MBE

Research article

XML security protection scheme based on Kerberos authentication and polynomials authorization

Lihong Guo[1,2,*], Jian Wang[1], Haitao Wu[2] and Najla Al-Nabhan[3]

[1] College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China
[2] Department of Information and Communications Engineering, Nanjing Institute of Technology, Nanjing 211167, China
[3] Department of Computer Science, King Saud University, Riyadh 11564, KSA

* **Correspondence:** Email: guolihongnj@163.com; Tel: +86 13952090181.

Abstract: With XML becoming a promising standard for data storage, describing, transferring and exchanging information on the Internet, data security and privacy protection of XML become the focus of research in recent years. In order to achieve the authorization of legitimate user and ensure the secure access to sensitive information, in this paper, in the context of cloud storage, with the purpose of sharing sensitive XML information, a polynomial authorization scheme with Kerberos authentication was proposed, which was based on the users' access purpose and privacy policy. In this scheme, first, Kerberos authentication was used to identify the user, and then the polynomial whose coefficients were from the leaf node address was used to complete the authorization of user. For the legitimate user, under the interaction of authorization polynomials and the global structure view, authorization matrix is generated dynamically, its temporary and dynamic characteristics greatly improves the security of the system. Finally, with the help of authorization matrix and auxiliary information tables, security queries were successfully completed. The experimental results show that the scheme not only effectively protects XML sensitive data, but also reduces the server's storage pressure, at the same time it is beneficial to the rapid search and information positioning.

Keywords: privacy; data security; authentication; authorization; polynomial

1. Introduction

With the development of information technology, the problem of network security is becoming more and more prominent, among which identity authentication and access authorization are particularly important. Since eXtensible Markup Language (XML) has been widely used in many enterprises and organizations for data representation and data exchange [1,2], and has been playing an important role in IT systems and applications, so XML's authentication and access authorization become more and more important [3,4]. In general, XML data used by the enterprises and organizations usually contains privacy information, such as transaction data in the electronic commerce and the document used to record the relationship of customer in company, and some are even the core commercial secrets of enterprises. Therefore, how to use these XML data effectively and safely is becoming the research focus of current database service [5,6].

At present, there are a number of literatures on XML security and data privacy protection, but most of them have some problems: Some studies only focus on structural protection, and ignore the importance of content; some studies only focus on content, and ignore the influence of structure; some spend too much time on nodes' authorization search, and some take up a lot of time to implement the storage for different access views. In fact, in a shared environment, we need a new privacy protection scheme, It should not only consider the structure, but also pay attention to the content, at the same time, it needs to improve the query efficiency and reduce the storage consumption.

In this paper, a novel scheme is proposed, which mainly focuses on XML's user authentication and authorization. The main contribution of this paper can briefly be summarized as follows:

(1) A new XML authorization scheme based on the polynomials is proposed for the first time, and Kerberos authentication scheme is used to identify the user.

(2) Under the interaction of authorization polynomials and the global structure view, authorization matrix is generated dynamically, its temporary and dynamic characteristics greatly improves the security of the system.

(3) Transforming the user's search requirement into a search matrix, which is in conjunction with the user's temporary authorization matrix and conditional privilege matrix to finish the user's security query.

(4) A serial of experiments result shows that the scheme not only saves the server's storage space but also efficiently speeds up the security search, which is convenient to find the information and positioning.

The rest of this paper is organized as follows. After introducing the main contribution in section 1, the existing access control models and privacy protection mechanisms are introduced for XML data in section 2. Then section 3 presents the idea and the design about the authentication and authorization. In section 4, the security search process is provided. In section 5, a series of experiments are carried out to measure the performance of the scheme. Section 6 summarizes this scheme and makes some suggestions for the next work.

2. Related works

In recent years, the demand of privacy protection warms up continuously, the domestic and foreign research institutions, enterprises and scholars have devoted to the research of privacy protection. At present, there are many privacy protection schemes for XML, most all of schemes

generally use data encryption, access control or digital signature technology et al. [7–18].

2.1. Data encryption

In the field of database service, the most direct solution to prevent the illegally access is to encrypt the data. Originally, World Wide Web Consortium (W3C) and the Internet Engineering Task Force Internet (IETF) provide a set of XML secure protocols and technical standards to meet XML security requirements [14,15]. In XML encryption standard, it defines encryption granularity from three aspects. From encryption granularity, we found that there were two problems in this standard: (1) It was impossible to encrypt an ancestor node while leaving the descendants of this node unchanged. (2) Information leakage existed between the users. The element xenc: Encrypted data exposed the initial location of the plaintext, and at same time the intruder could infer the size of plaintext from the size of cipher text. Therefore, there may be information leakage including the position disclosure and the size disclosure of plaintext information. The example of encryption scheme is shown in Figure 1a, the grey node represents the encrypted node. In order to hide the encrypted node's size and position, the literature [19] proposed XML pool encryption scheme. Its basic idea is to encrypt each node separately, and then move all the encrypted nodes into a pool from their original positions. In this scheme, before pushing the encrypted nodes into the pool, it need bind the original position information with the encrypted node. Because it need reconstruct the document once upon the user gets the nodes' decryption keys. In XML pool encryption scheme, the moving of encrypted nodes breaks down the structure of XML document, which makes its reconstruction difficult and time-consuming. The example of this scheme is shown in Figure 1b.

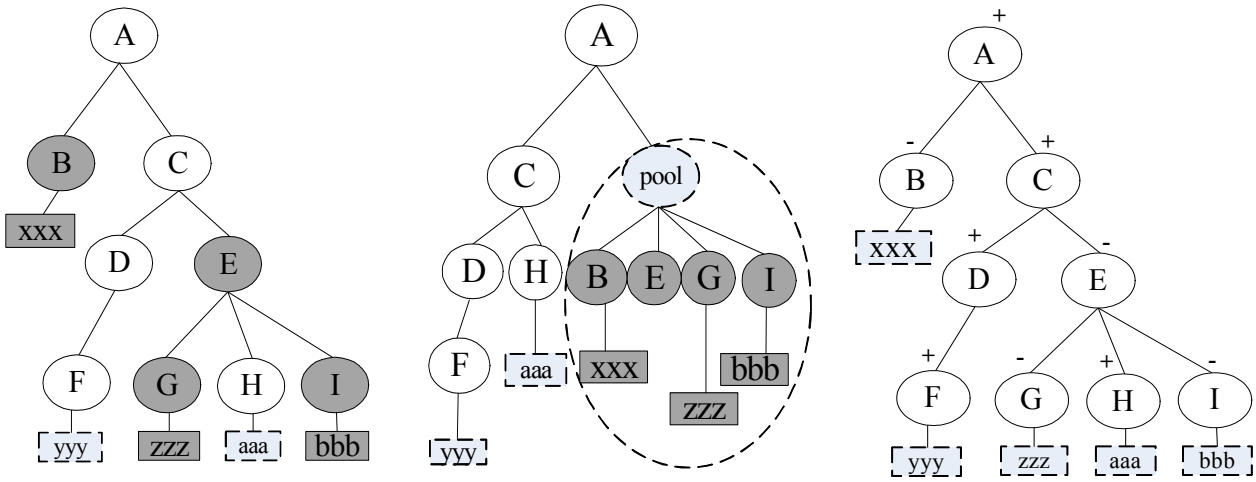
2.2. Access control

Access control is one of effective methods to protect the privacy. Several researches have been done to specify a fine-grained access control on XML data. The literature [20] first proposed a Hippocratic XML model based on the ancestor's authorization. In this model, it solved the problem of finding the nearest ancestor nodes' authorization and proposes an efficient access control mechanism that used the authorization index and the nearest neighbor search technique, the example is displayed in Figure 1c, here, '+' denotes permit and '−' denotes deny. The literature [21] proposed a view-based access control mechanism, which was named as ViewAC. This mechanism created and maintained a separate view for each user, which contained exactly the set of data elements authorized to access. So this mechanism suffered from high maintenance and storage costs especially for a large number of users. Angela et al. [22] put forward a privacy model called Privacy for All (P4A). In this model, a privacy policy considered two major elements: Data and the purpose of use. This model offered more flexibility than current approaches in that it allowed unconditional and conditional access. However, this model only considered the leaf nodes' authorization and ignored the protection of structure information. In other words, it only provided the authorization of leaf nodes and ignored the authorization of intermediate nodes. The literature [23] proposed a new protection mechanism based on the modified and enlarged theory of Shamir's secret. It can effectively protect the core data, at same time, this method can effectively authenticate user's identity. But the distributing of the secret is difficult and complex, which impacts on its efficiency. The literature [24] proposed a data access control model for individual users, through the semantic visual range of inverted XML structure is

realized. It extended the authority by the semantic dependency association, and obtained the XML view semantically dependent on inverted index. XML access control based on ontology semantic dependency is a difficult research direction, which need further consideration and research.

2.3. The existing problems

In summary, although above-mentioned methods can accomplish the protection of the underlying data and effectively prevent the external enemy to access the data, but there are still some problems in them. Like the relational database, encrypted XML document makes queries difficult, and spending too much time in encryption and decryption directly affects the system's efficiency. Although many access control schemes have been proposed, these schemes either ignore the protection of content or ignore the protection of structure. In fact, an XML document needs dual protection for the content and the structure. Facing with the non-absolute secure environment, we need to find the better data protection scheme, which brings great challenges to traditional XML data protection.



(a) W3C XML encryption. (b) XML pool encryption (c) XML access control

Figure 1. Examples of XML privacy protection scheme.

3. An XML user authentication and authorization scheme

In this section, we propose a new user authentication and authorization scheme based on the polynomial. Kerberos authentication is to ensure the legal identity of the user, and only legitimate user can obtain the corresponding authorization polynomial. Then the authorization polynomial combines with the global structure view to generate the temporary authorization matrix. Based on it, we can complete a series of security information search for XML document.

From the structural features of XML document, we know its tree structure determines the skeleton of information, and the content of leaf node is the core information that need to be protected. For example, in an XML document that stores electronic medical records, the structure information represents that this is the electronic medical records, but from the content of leaf node, we can get detailed personal medical records including the name, address, medicine data, etc. Therefore, the content is the core information, but the structural characteristics of XML also determine that its

structure hides some useful information, so both of them need to be focused on. In this scheme, inspired by the idea of separating structure and content in XML document, the encoding of structure and the content information are stored respectively in the server, and the auxiliary information is the foundation of information search, with the help of them, the structure and content can be combined into useful information by some algorithm.

3.1. The architecture

Figure 2 shows the architecture of new scheme. To better present the idea of the scheme, we take the electronic medical records as an XML example to describe patients' information. The system consists of three components: Kerberos authentication module, client module and the server module. Kerberos authentication module includes authentication server (AS) and ticket-granting server (TGS), its goal is to provide powerful authentication services for client and the server by the key system [25]. Client is the user who wants to get some useful information from the server. The server module is the core component; it is responsible for the user's authorization and feedback of security information.

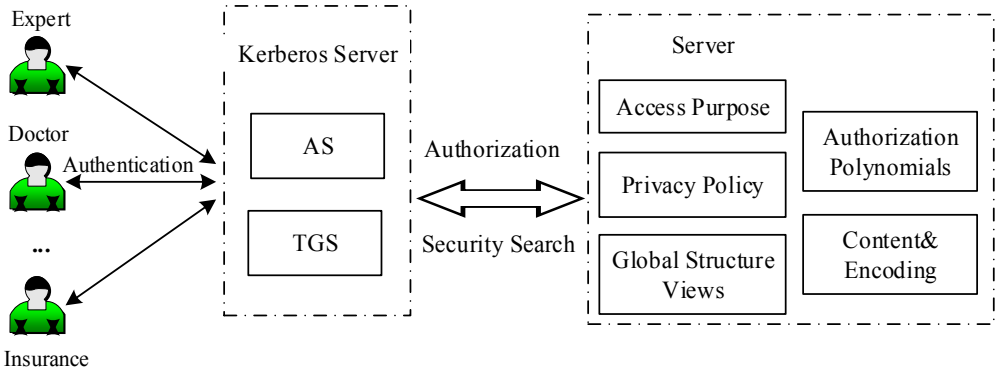


Figure 2. The architecture of new scheme.

3.2. Authentication module

Kerberos is an authentication protocol that is used to verify the identity of a user or host. It is widely used in many fields [26,27]. In the literature [28], it focuses on the improvement of mobile application security mechanism. This paper proposes a security model of mobile application which is based on Kerberos authentication. In this improved security model, every query request from the client will be authenticated by the Kerberos server, but the authentication will be different between users with different access purpose. In our system, Kerberos is used to identify different users. It provides a means of verifying the identities of principals in an open network environment [29]. The authentication process proceeds as follows:

(1) Get the tgs ticket

$$C \rightarrow AS : ID_c \,||\, ID_{tgs} \,||\, Times \,\|\, Nonces_1$$

$$AS \rightarrow C : ID_c \,||\, Ticket_{tgs} \,||\, E_{K_c}[K_{c,tgs} \,||\, Times \,\|\, Nounces_1 \,||\, ID_{tgs}]$$

$$Ticket_{tgs} = E_{K_{tgs}}[K_{c,tgs} \,\|\, ID_c \,\|\, AD_c \,\|\, Times]$$

A client sends a request to the AS requesting "Service Ticket" for a given server. The message contains client's identity ID_c, TGS server's identity ID_{tgs}, Times and random value $Nounces_1$. The AS responds with the request, the respond message contains a ticket-granting ticket (TGT) which can later be used with the TGS.

(2) Get the server's ticket

$$C \rightarrow TGS : ID_s \,||\, Times \,\|\, Nonces_2 \,\|\, Ticket_{tgs} \,\|\, Authenticator_c$$

$$TGS \rightarrow C : ID_c \,||\, Ticket_s \,||\, E_{K_{c,,tgs}}[K_{c,s}|| \, Times \,\|\, Nounces_2 \,||\, ID_s]$$

$$Ticket_s = E_{K_s}[K_{c,s} \,\|\, ID_c \,\|\, AD_c \,\|\, Times]$$

$$Authenticator_c = E_{K_{c,tgs}}[ID_c \,\|\, TS_1]$$

In the second step, the client sends a request to the TGS. The client sends TGT to the TGS in the same manner as if it were contacting any other server, which requires Kerberos credentials. The reply is encrypted in the session key from the TGT. Once obtained, credentials may be used to verify the identity of the principals in a transaction, to ensure the integrity of messages, or to preserve privacy of the messages. The application is free to choose whatever protection may be necessary.

(3) Get the server's service

$$C \rightarrow S : Ticket_s \,\|\, Authenticator_c$$

$$S \rightarrow C : E_{K_{c,s}}[TS_2 \,\|\, Subkey \,\|\, Seq\#]$$

$$Authenticator_c = E_{K_{c,s}}[ID_c \,\|\, TS_2 \,\|\, Subkey \,\|\, Seq\#]$$

The client transmits the ticket (which contains the client's identity and the TGS server's identity copy of the session key, all encrypted in the server's key) to the server. The session key (now shared by the client and server) is used to authenticate the client, and to optionally authenticate the server. At the same time, it can also be used to encrypt further communication between two parties or to exchange a separate sub-session key to be used to encrypt further communication. The server returns the session key to client, and then they can further exchange all kinds of information.

3.3. Authorization module

This is the core module. Only the legal users can obtain the corresponding authorization polynomial, generate the temporary access control matrix, and obtain the security data. Here we use the polynomial to represent the authorization of user. Temporary access control matrix will be created based on his authorization polynomial and the global structure view, due to its temporary characteristic, temporary access control matrix not only saves the storage space, but also ensures the safety of system.

3.3.1. Creation of global authorization polynomial and structure view

To create the global authorization polynomial and global structure view, first we adopt the start-end encoding to encode the whole XML document. For example, for the electronic medical records, the encoding example is shown in Figure 3, its detailed process is described in the literature [30,31].

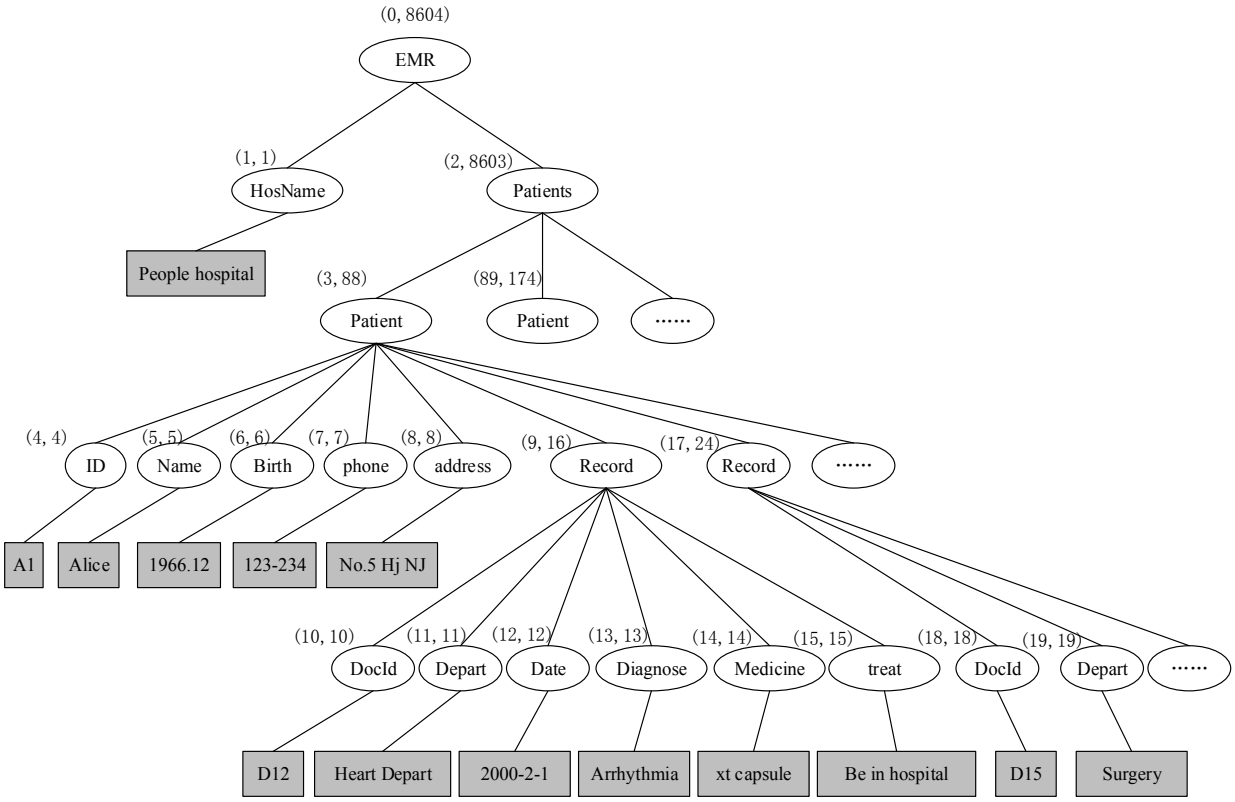


Figure 3. Encoding example of an XML document

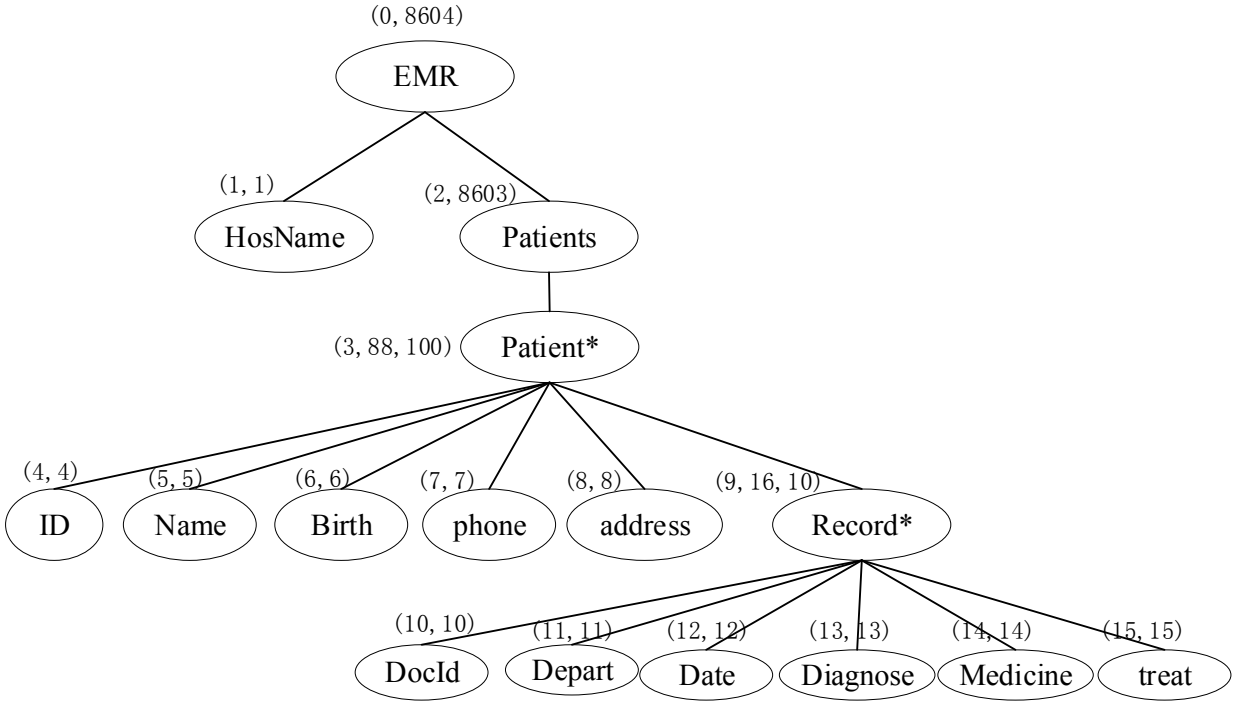


Figure 4. Global structure view corresponding Figure 3.

In order to establish a concise and accurate summary of XML structure, we extract the global structure view that adopts the method named as DataGuide [32]. The extracting rule specifies that XML's global structure view describe every unique label path exactly once, regardless of the number of times it appears in XML document. According to the DataGuide's extracting rule, a global structure view corresponding to Figure 3 is displayed in Figure 4. Especially, in this figure, the node 'Patient' and the node 'Record' are the repeated nodes, here labeled*.

According to the global structure view, we first establish the global polynomial equation f(x). In it, the degree of polynomial is decided by the number of leaf nodes, and at same time the coefficient of polynomial is the encoding (address) of leaf node content by the order of leaf node in global structure view. If a user knows the whole coefficient in the polynomial, which means he can get all the content in this XML document.

For example, in Figure 4, the number of leaf nodes is 12, from left to right and from top to bottom. 'HosName' is the first leaf node, whose content's address is the coefficient a_0 (encoded as 1). 'ID' is the second leaf node, whose content's address is the second coefficient a_1 (encoded as 4), and so on. So the global polynomial equation f(x) is Eq (1) corresponding to Figure 4. The real f(x) is Eq (2). (a_0 = 1, a_1 = 4, a_2 = 5, …., a_{11} = 15).

$$f(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + a_4x^4 + a_5x^5 + a_6x^6 + a_7x^7 + a_8x^8 + a_9x^9 + a_{10}x^{10} + a_{11}x^{11} \quad (1)$$

$$f(x) = 1 + 4x + 5x^2 + 6x^3 + 7x^4 + 8x^5 + 10x^6 + 11x^7 + 12x^8 + 13x^9 + 14x^{10} + 15x^{11} \quad (2)$$

Correspondingly, the system will create different tailored authorization polynomials for the users with different access purposes. The creation rule is: (1) If the leaf node is inaccessible according to the user's authorization rule, then corresponding coefficient is zero in his authorization polynomial. (2) If all children nodes of some node are inaccessible, we will delete this node. The deletion sequence is from the bottom to top and from the leaf node to this node.

Example 1: An industrial authorized user wants to know about the diagnose information, whose purpose is to analyze and research the reason of diseases. His access purpose displays that he wants to get the information about 'HosName', 'Birth', 'Depart', 'Diagnose', 'medicine', 'treat', which means that he wants to get the Eq (3). However, after filtered by the protection policy, the final authorization polynomial is displayed in Eq (4).

$$f(x) = 1 + 6x^3 + 11x^7 + 13x^9 + 14x^{10} + 15x^{11} \quad (3)$$

$$f(x) = 6x^3 + 11x^7 + 13x^9 + 14x^{10} + 15x^{11} \quad (4)$$

3.3.2. User authentication and generation of access control view

In this scheme, the system will create an authorization table to record the related authorization information. User authorization table is showed in Table 1.

In table 1, 'User ID' is system ID of the user; 'Key' is used to verify user's identity, which isn't the user's password, but it can be calculated from the user's name and password. 'Authorization Polynomial' describes the user's authorization; f(x) represents the global authorization polynomial, because the polynomial coefficient is the encoding of content, so their values are different from each other. In addition, f(x) (a_0, a_1,…a_n) represents the limited polynomial, a_0, a_1… a_n are the denied coefficient, whose corresponding items should be deleted from the f(x).

Table 1. User authorization table.

User name	User ID	Key	Authorization Polynomial
Alice	0	4532	f(x)
Mary	12	56322	f(x)(1,5,7,8)
Bob	34	562254	f(x)(2,6)
Smith	78	52362	f(x)(2,6,7,8)
…	…	…	…

For example:

If

$f(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + a_4x^4 + a_5x^5 + a_6x^6 + a_7x^7 + a_8x^8 + a_9x^9 + a_{10}x^{10} + a_{11}x^{11}$

then

$f(x)\ (a_0, a_2, a_5) = a_1x + a_3x^3 + a_4x^4 + a_6x^6 + a_7x^7 + a_8x^8 + a_9x^9 + a_{10}x^{10} + a_{11}x^{11}$

then

$f(x)\ (a_2, a_6, a_7, a_8) = a_0 + a_1x + a_3x^3 + a_4x^4 + a_5x^5 + a_9x^9 + a_{10}x^{10} + a_{11}x^{11}$

About the authorization procedure, it is displayed below:

(1) Get the authorization polynomial

Login in the system, if the user passes the Kerberos identity authentication, the server will return his corresponding authorization polynomial from the authorization table.

(2) Generation of temporary access control view

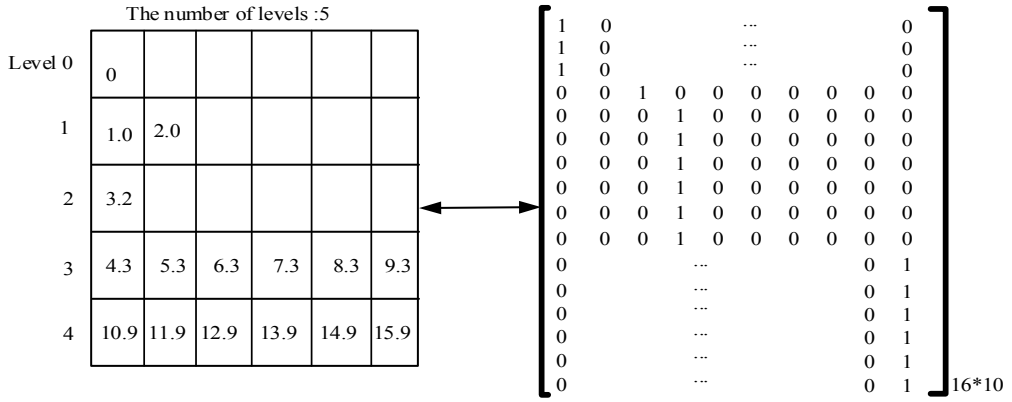
Based on the user's authorization polynomial and the global structure view, the system will create this user's temporary access control view. We provide some examples to explain the generation procedure of access control view.

In order to create the user's access control view, apart from the authorization polynomial, it also needs the global structure view. To be conveniently, we adopt the level structure to represent the global structure view, the level structure shows the levels and the tags in each level. Figure 5 shows the level structure corresponding with the global structure view in Figure 4.

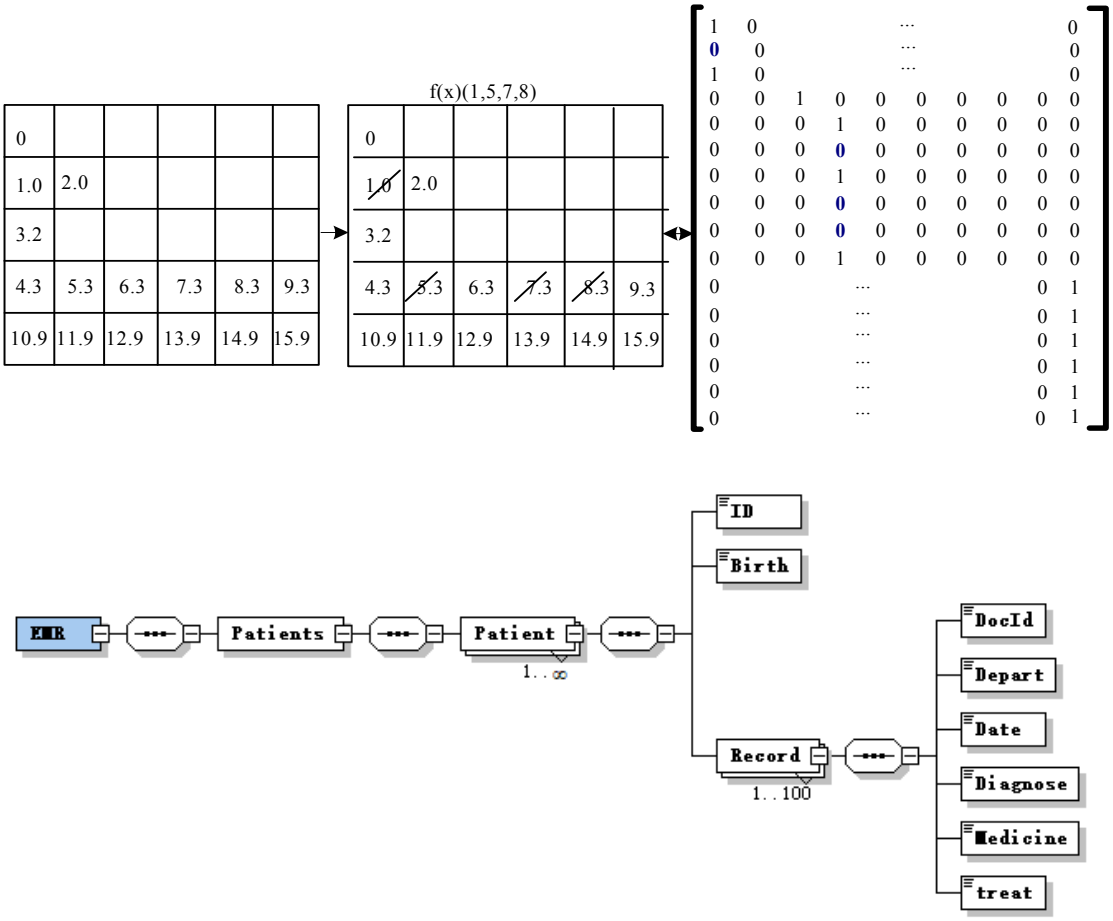
Figure 4 is the global structure view, in which every common node is labeled with (start, end), and the repeated node is labeled with (start, end, frequency). Figure 5a is the corresponding level structure of global structure view, in which every node is labeled with 'self-start value. Parent's-start value' according to global structure view. For example, Node 'HosName' is in the level 1 of the global structure view, it is labeled with (1, 1) and its parent node 'EMR' is encoded as (0,8604), so in the level structure, Node 'HosName' is labeled as '1.0', '1' represents itself start value, and '0' is its parent's start value. Node 'Patient' is in the level 2 of the global structure view, it is labeled with (3, 88, 100) and its parent node 'Patients' is encoded (2, 8603), so in the level structure, Node 'Patient' is labeled as '3.2'. In the same level, the encoding is sequential in arrangement. Root node is labeled as '0', because it hasn't parent node. The detailed level structure with label is displayed in the left figure of Figure 5a.

Furthermore, in order to search the information conveniently, we make the level structure change to be access control matrix. From the encoding character in the level structure, we can know the encoding value is different from each other, and is increasing gradually from top to down, from left to right. The encoding of final leaf node has the maximum start value and maximum parent's start value.

Therefore, access control matrix is created based on the encoding value of final leaf node. From the left figure of Figure 5a, we know that the final node's encoding is '15.9', because the encoding value begins from value '0', so we will create a 16 × 10 matrix. In matrix, if the leaf node exists in the level structure, then its value is 1, otherwise is 0.



(a) The global structure view and its corresponding level structure.



(b) Access control view corresponding to example 1.

Figure 5. Creation procedure of of access control view.

For example, in Figure 5a, left figure is the level structure, whose node has been encoded with (self-start, parent's-start). The root node '0' is saved in [0,0] in the matrix, its value is 1. Node 'HosName' (1.0) is saved in [1,0], Node 'Patients' (2.0) is saved in [2,0], …., 'Medicine' (14.9) is saved in [14, 9], and 'treat (15.9) is saved in [15, 9]. By this method, we complete the creation of access control matrix, transforming the level structure to be an access control matrix. The matrix stands for the user's access control view. The result matrix is the right figure in Figure 5a.

So far, we know the global structure view is corresponding with the level structure one to one. Now we know the global structure view is changed into an access control matrix. The following section, we will provide some examples to describe the process about how user's authorization polynomial to change to be an access control matrix.

Example 1: As an industrial registered user Mary, she wants to research the disease situation in different age based on all the electronic records. After the identity authentication, the system will provide her the global structure view and her authorization polynomial according to her user ID.

For example, Mary gets the authorization polynomial f(x) (1,5,7,8) and the global structure view (corresponding with the level structure). Then her authorization process is showed in Figure 5b.

In Figure 5b, left figure is the level structure corresponding to the global structure view. The middle figure is the tailored level structure according to Mary's authorization polynomial f(x) (1,5,7,8), in this figure, based on the f(x) (1,5,7,8), the system will delete corresponding node just as start value is 1,5,7,8. Right figure is her access control matrix corresponding to the middle tailored level structure. In Figure 5b, the below figure is Mary's real access control view corresponding to access control matrix.

3.3.3. The other auxiliary information tables

In order to complete the information search, the system must contain some core information tables, the main structure encoding table, the content table, etc.

The structure encoding table is displayed in Table 2 corresponding to Figure 3. The first column is the encoding of all the nodes. The leaf node content is displayed in Table 3 corresponding to Figure 3. In this table, the first column is the content encoding of all leaf nodes in whole XML document, and the second is the specified leaf node's content. Here, because the leaf node's start value is equal to its end value, so we take its start value to encode the leaf node's content, which is just like the address of content, by this encoding, we can obtain the content. For example, in Figure 4, the leaf node 'Name' is encoded as (5, 5), then we label this node's content is '5', through this digit 5, we can obtain the content 'Alice'.

In the use of XML document, it has many repeated nodes with same structure, although the repeated nodes occur only once in global structure view, the content of repeated nodes' is very important. So in order to protect the content and link all the structure and content information we use start-end encoding to encode all the nodes, and take the leaf node encoding as the content's address. Thus, in the processing of search, we can calculate other repeated leaf node content encoding based on the first leaf node encoding in global structure view. We assume that R [0] is the first repeated node, its encoding is (p, q). The repeated node R[i]'s encoding is labeled (m, n), then its calculation formula is present in Eq (5):

$$R[i](m,n) \quad \begin{cases} m = p + (q - p + 1) * i \\ n = m + (q - p) \end{cases} \tag{5}$$

In this formula, i = 0, 1, 2, 3…., it is the order of repeat node. About the detailed start-end encoding is shown in the literature [30].

Table 2. Structure information table.

Encoding (Structure view)	Node Name
(0, 8604)	EMR
(1, 1)	HosName
(2, 8603)	Patients
(3, 88, 100)	Patient
(4, 4)	ID
(5, 5)	Name
(6, 6)	Birth
…	…

Table 3. Leaf nodes' content table.

Encoding	Content
1	People hospital
4	A1
5	Alice
6	1966.12
7	123–234
……	……

In Table 2, it has two types of encoding, two value encoding is the ordinary node and the 3 value encoding is the repeated node. For example, ordinary node 'HosName' is encoded as (1, 1), and repeated node "Patient" is encoded as (3,88,100), digit 3 is the start value, digit 88 is the end value, and digit 100 is the number of repeat. With the help of encoding, the content is linked with its encoding (named address). In the server side, the content stored in disorder, even if the attacker gets the content, he can't link other information and get the meaningful information, so it is meaningless.

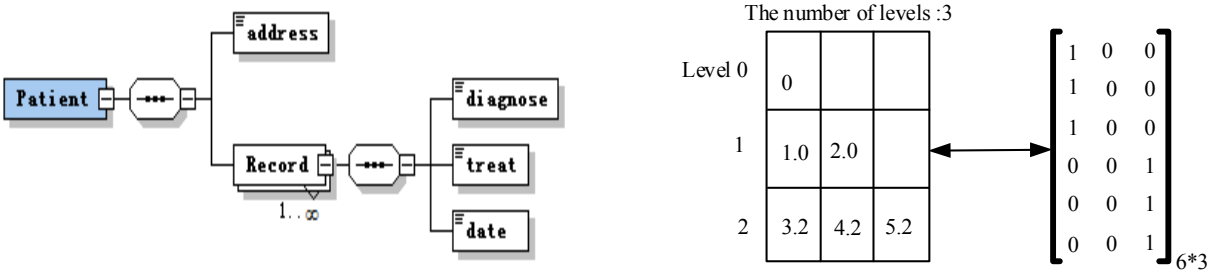
For safety considerations, the content table is saved in a confused form. The detailed search process about repeated structured nodes' content is described in literatures [31].

4. The search process of information

Depending on the global structure view, all the information tables, the authentication and authorization method, the system can deal with all kinds of user's search requirements. Because the access control view of user is changed into an access control matrix, so in the process of information search, the user's search requirement also can be representing as a search matrix, from this point we can conclude that the search process is a procedure of matrix calculation. The detailed search procedure is shown in Figure 6a.

```
Algorithm: Search Procedure
Input: ID, password, req        Output: Result
1.  variant Handle Search ( )    // Running in the server
2.  {   Matrix M, A, B, C, R;
3.      int fx[50][50];          //two-dimension array to save the global polynomial f(x)
4.      string req, condition;   //the user's search request, here it uses Xpath language
5.      variant result;          //the search result
6.      while listening ( )      //always listening the user's request
7.      {  if (Verify Authen (ID, password)= = true)          // verify the identity and get the authorization
8.          {      Parse_Request (req);                      //parsing the user's Xpath request
9.                 Prepare_ And_ Adjust (req);               //Preparing and adjust the Xpath request
10.                A = Trans_To_Matrix (req);                //Change request to be search matrix A
11.                B = Create_Access_Control_Matrix (f(x), M);   //Create access control matrix B
12.               If (the system has secure condition)
13.               {    C = Trans_To_Matrix (condition);     //Change condition to be matrix C
14.                    R = Calculate_Matrix (A,B,C);        //Calculate the result and save in matrix R
15.               }
16.           else   R = Calculate_Matrix Conditon (A,B); //calculate the result and save in matrix R
17.           result = Combine_Result (R);                  //Integrate the search result
18.           return result;
19.       }
20.   }
21. }
```

(a) Procedure of the search request.



(b) DTD view, access view, level structure and corresponding access matrix.

Figure 6. The procedure of information processing.

For example 1: In order to simplify the describing process, here considering to use DTD to express access view in an electronic record document, assume that DTD view is in below:

<!ELEMENT Patient (address, Record*)>

<!ELEMENT Record (diagnose, treat, date)>

Root node is 'Patient', it has two son nodes 'address' and 'Record', further node 'Record' has three son nodes 'diagnose', 'treat', and 'date', its DTD view, access view and corresponding access

control matrix are displayed in Figure 6b.

The DTD view's corresponding access control matrix is 'B' in Eq (6). Here we choose XPath [33] as the structured query language, if a user submitted the XPath request 'Patient[0]/address', his corresponding search matrix is A, assume that the conditional privilege matrix is C, and the search result is R, the calculate function is in Eq (6). Here, '*' represents 'and' operation.

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \mathrm{R = A * B * C} \quad R = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{6}$$

5. Experiments

In order to test this scheme's efficiency and performance, we have made a series of experiments. It uses Visual C++ 6.0 and Matlab mixed programming technology. In the process of parsing XML document, Document Object Model (DOM) [34] is used to gain the nodes and leaf nodes' content information. In the experiments, we choose four kinds of real test dataset [35]. In the test dataset, the maximum size is 2190 KB, the minimum size is 5 KB. The maximum node number is 48002, and the minimum node number is 127. The maximum depth of XML document tree is 5, and the minimum depth of XML document tree is 3. The number of leaf nodes ranges from 101 to 40001. The detailed dataset information is shown in Table 4.

Table 4. Test dataset information.

Number	Dataset name	Size	Node number	Depth	Leaf nodes number
1	Nation.xml	5 KB	127	3	101
2	Ubid.xml	20 KB	342	5	275
3	Customer.xml	504 KB	13502	3	12001
4	partsupp.xml	2190 KB	48002	3	40001

At present, in almost all XML access control models and privacy protection schemes, the storage space, the compression ratio, search time and security degree are often used to evaluate new model's performance. In them, the security degree is the primary goal, while query time is a secondary factor, and the final storage space is one of the factors to judge the advantages and disadvantages of the scheme. In order to evaluate the performance of this new scheme, ViewAC's approach [21] and Matrix storage method [30] are compared with ours.

5.1. The storage space

Theoretically, we can analyze that in ViewAC's scheme, it creates and maintains many separate different XML documents for different purpose's users, which contain exactly the set of data elements authorized to access, so as the number of different users increases, the demand for storage space becomes larger. Consequently, this scheme suffers from high maintenance and storage costs.

In Matrix storage scheme, it transforms access view into different storage matrix, so for different purpose's users, we only need to save the matrix message for different user, so as the number of different users increases, the need for storage will slowly increase, but not much, which has very low storage costs.

In our polynomial scheme, we use the polynomial to represent the authorization, and create the temporary access control view to help the user, so the temporary storage more effectively saves the space than other schemes.

In the experiments, we test and obtain the size of storage space in several different views. The comparison of storage space in several different views respectively are shown in Figure 7a–f, the horizontal abscissa digit stands for the test document number in Table 4, and the vertical abscissa digit stands for the storage space, whose basic unit is KB.

From the experiment data and the result, we know that in ViewAC's scheme, its space size is the size of entire XML document, which only saves one view for one type of user. For the test document Nation.xml, in one view, ViewAC's storage space is 5 KB, in two views, ViewAC's storage space is 9 KB, and it serves two different types of users, two different types of documents are cut from the original document. In three views, its storage space grows to 12 KB and it is 23 KB in four views. In ten views, its storage space grows to 45 KB and it is 85 KB in twenty views.

For the test document partsupp.xml, in one view, ViewAC's storage space is 2190 KB; in two views, ViewAC's storage space is 3604 KB. In three views, its storage space grows to 4328 KB and it is 5200 KB in four views. In ten views, its storage space grows to 17450 KB and it is 39004 KB in twenty views. The same analysis is in Matrix and our new polynomial scheme. In a word, from the storage space, it shows that ViewAC's storage space is growing greater than matrix and polynomial schemes', with the increase of different users, ViewAC's storage space follows the rapid growth, and our polynomial scheme has obviously low storage costs.

5.2. The compression ratio

The compression ratio is another factor to evaluate the scheme. It is similar to the storage space, the ratio value is from the calculation in storage space data, but they are slightly different, which reflects another size of the problem.

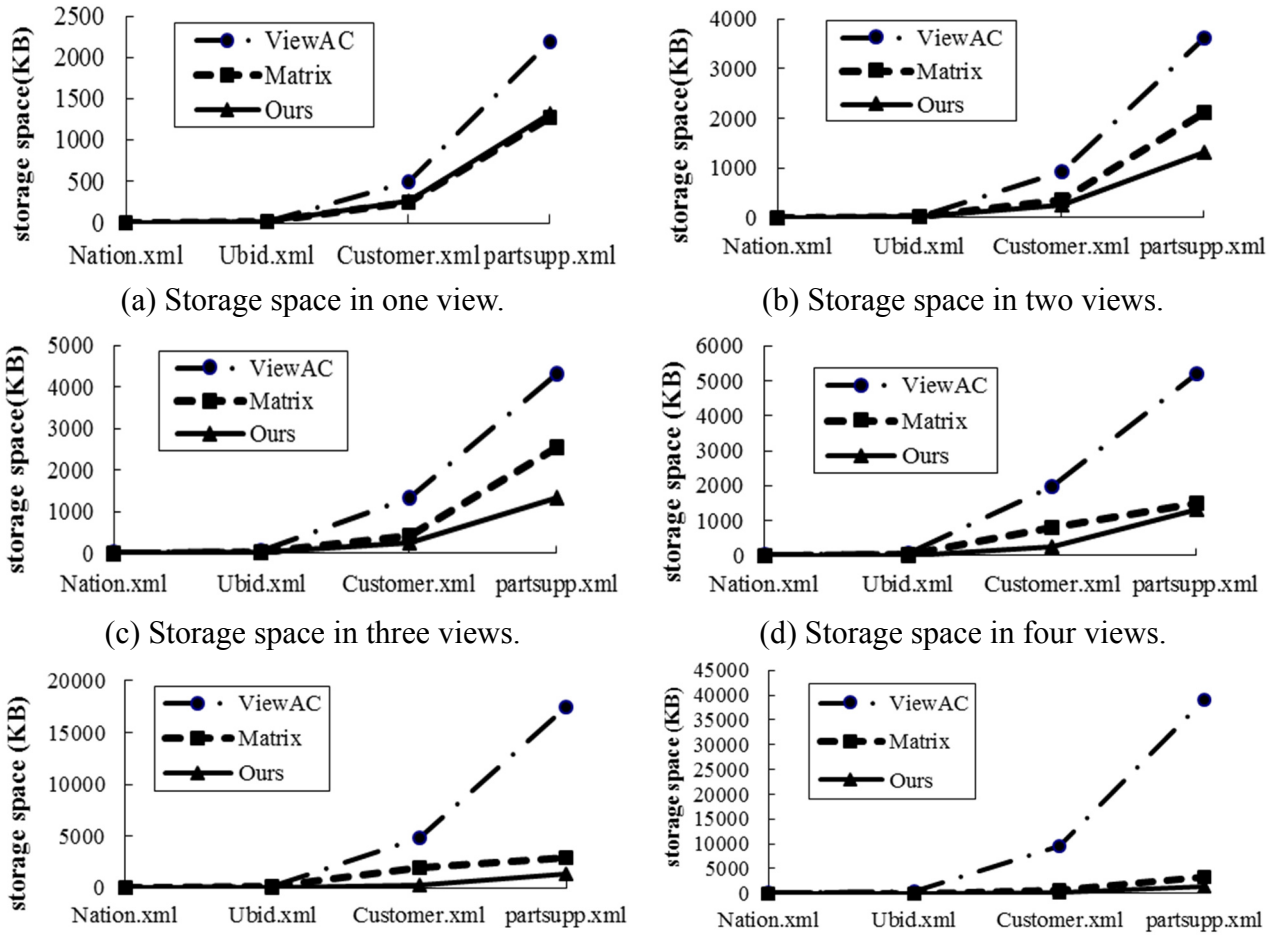
Compression ratio is the ratio of size after compression to size before compression. For example, a shared XML document, if its source size is 5 KB, and its size is 4 KB after compression, so compression ratio is: 4 KB/5 KB × 100% = 80%.

Here we take matrix scheme as an example, it uses different matrix to save different access control view. Therefore, with the increasing of different users, its space storage is also slowly increasing. With regard to compression ratio, we compare three schemes using same test dataset. The data source is from Figure 7 and the ratio value is displayed in Table 5.

From the data, we know that for Nation.xml its compression ratio is from 100 to 80% in ViewAC scheme, the ratio is from 65 to 10% in Matrix scheme, and the ratio is from 52 to 1% in our new scheme. For Ubid.xml its value is from 100 to 80% in ViewAC scheme, the value is from 68 to 9% in Matrix scheme, and the value is from 73 to 4% in our new scheme. The same analysis is for customer.xml and cartsupp.xml documents. Accordingly, can conclude that our new scheme has a good compression ratio.

In a word, for ViewAC's scheme, the number of different users has the greatest impact, and

others have less influence, with the increasing of different users' number, its space storage is growing quickly, so its compression ratio is very high. For matrix scheme, it uses different matrix to save different access control view. Consequently, with the increasing of different users, its space storage is also slowly increasing, and its compression ratio is slowly reducing. In our polynomial scheme, because only a global structure view is saved in the server, so in whatever conditions and in different views, the server can create different temporary access control view for different user, but these temporary access control views are not be saved, so its storage space is always constant. It means that our scheme has obviously low storage costs and low compression ratio especially with the increasing in the number of different users.



(a) Storage space in one view.

(b) Storage space in two views.

(c) Storage space in three views.

(d) Storage space in four views.

(e) Storage space in ten views.

(f) Storage space in twenty views.

Figure 7. Compare of storage space in different views.

5.3. The execution time

In order to judge the query efficiency of the scheme, we test the execution time from two aspects:

(1) The resolution time for XML document;

(2) the specific search time.

Parsing XML document is the first pre-processing step before searching the information. After parsing XML document, all the different nodes are encoded according to above labeling scheme, and

all the content is saved in the content table according to its encoding. Finally, the sever will save all kinds of structure encoding table, content table, etc. In the experiment, DOM technology is used to parse XML documents. The detailed resolution time about test XML documents is list in Table 6.

Table 5. Compare in space compression ratio to test dataset.

Number	Dataset Name	Scheme Name	1	2	3	4	10	20
1	Nation.XML	ViewAC	100%	90%	80%	95%	90%	85%
		Matrix	48%	36%	27%	65%	10%	12%
		Ours	52%	26%	2%	1%	5%	3%
2	Ubid.xml	ViewAC	100%	85%	83%	94%	93%	80%
		Matrix	68%	54%	43%	44%	39%	9%
		Ours	73%	36%	24%	18%	7%	4%
3	Customer.xml	ViewAC	100%	92%	89%	98%	97%	94%
		Matrix	48%	35%	27%	40%	38%	7%
		Ours	52%	26%	17%	13%	5%	3%
4	Partsupp.xml	ViewAC	100%	82%	66%	59%	80%	89%
		Matrix	60%	48%	39%	17%	13%	7%
		Ours	58%	30%	20%	15%	6%	3%

Table 6. Resolution time of XML documents.

Number	Dataset name	size	Depth	Resolution time (sec)
1	Nation.xml	5 KB	3	0.055664
2	Ubid.xml	20 KB	5	0.078755
3	Customer.xml	504 KB	3	1.414248
4	Partsupp.xml	2190 KB	3	96.765422

In this part, we also test and obtain the search time from the following aspects: (1) The query result tree is a single leaf node without any constraint condition. (2) The query result tree is the repeated sub tree without constraint condition. (3) The query result tree is a single leaf node with constraint condition. (4) The query result tree is the repeated sub tree with constraint condition.

In this experiment, we compare matrix scheme with our polynomial scheme from the search time. In Table 7, XPath is used as the structured query language. 'Condition' column refers to the special condition of the query, as mentioned before the system has some conditional privilege, which is represent as the privilege access. In the procedure of test it is from different aspect and different position to test the efficiency.

From the experiment result, we can conclude that our polynomial scheme speeds up the search efficiency for the large XML document, especially with repeated structure in most cases, and it is convenient to find the corresponding information. Combined with the test document and the document's features, we can conclude that the number of leaf nodes and the structure of XML document have the greatest influence in query time, the result tree's position in XML document has

greater influence, and the others' influence is relatively small. In some circumstance, the search condition also can cause a certain fluctuation for the execution efficiency.

Table 7. Search time of test XML documents.

Number	Dataset Name	XPath	Condition	Matrix Method Search time(sec)	Polynomial Method Search time(sec)
1	Nation.xml	T[10]	N	0.058062	0.041907
		T/N_COMMENT	N	0.029287	0.031675
		/table/T[5]/N_NAME	Y	0.046891	0.039087
		/table/T[24]/N_REGIONKEY	Y	0.062530	0.065283
2	Ubid.xml	memory	N	0.043862	0.066235
		/root/listing[10]	Y	0.091287	0.106812
		/root/listing/auction_info	Y	0.066891	0.075690
		/root/listing/item_info/description	N	0.072532	0.098834
3	Customer.xml	T[1500]	N	23.380162	20.120132
		T/C_COMMENT	Y	0.912387	0.665601
		/table/T[0,1000]/ C_ADDRESS	N	18.236891	12.236891
		/table/T[0,100]/ MKTSEGMENT	Y	13.132530	10.156790
4	partsupp.xml	T[4000]	N	75.238062	70.897015
		PS_COMMENT	N	21.491287	16.569018
		/table/T[0,1200]/ PS_AVAILQTY	Y	40.236891	35.345906
		/table/T[0,8000]/ S_SUPPLYCOST	Y	55.132530	50.098534

Table 8. Comparison about security degree.

Security Standpoint	ViewAC Scheme	Matrix Scheme	Our Scheme
Structure Security	No consider	Consider	Consider
Content Security	No consider	Consider	Consider
Security Authentication	No consider	No consier	Consider

5.4. The security

From the security standpoint, we compare and analyze the security degree from the structure security, content security and security authentication. In ViewAC scheme, it only considers to create different document for different purpose's user from the viewpoint of access security. It takes the document as a protection object, not considers the protection of structure and content, and not considers security authentication, so this scheme is coarse. In Matrix scheme, it separates the document's structure and content, and prevents the intruder's unauthorized access with a certain extent, but it lacks of security authentication. In polynomial scheme, it considers all the aspects including the structure security (creating the temporary access control view), content security (saving it in a confused table), and security authentication (adopting the polynomial to verify his identity). At the same time, the

scheme skillfully adopt the polynomial to represent the authorization, it also can help the user verify his identity. The analysis and comparison about security degree is shown in Table 8.

From the security standpoint, we know that our polynomial scheme takes into account all kinds of security factors, it includes the authentication, structure security and content security, and a good access structure is established to achieve a certain degree of security.

6. Conclusions

In this paper, we propose a new XML user authentication and authorization scheme. It uses the polynomial to indicate the user's authorization. On the other hand, this polynomial also is used to verify the user's identity to finish the authentication. Temporary access control matrix based on the global access control view and authorization polynomial can speed up the process of data access and ensure the access security. In the experiments, we test this scheme's storage costs, space compression ratio and execution efficiency, the experiment result shows that this scheme has low storage costs, low compression ratio and high execution performance.

At present, almost all the schemes are based on the hypothesis that the database is running in a trusted server. With the development of cloud services, database service provider is no longer a fully trusted entity, even itself may be a potential enemy. In the face of non-absolute secure database server, it brings great challenges to traditional database protection method. Therefore, we still need to find another good data protection scheme and good search algorithms for XML document [36–41].

Acknowledgments

This work was supported by the Innovation Funding of Nanjing Institute of Technology (No. CKJB201704), the Scientific Research Funding of Nanjing Institute of Technology (No. ZKJ201612), the National Natural Science Foundation of China (No. 61701221). The authors extend their appreciation to the Deanship of Scientific Research at King Saud University for funding this work through research group (No. RG-1441-331).

Conflict of interests

The authors declare no conflict of interest.

References

1. A. Moller, M. Schwartzbach, XML graphs in program analysis, *Sci. Comput. Program.*, **76** (2011), 492–515.
2. F. Zhang, Z. M. Ma, L. Yan, Construction of fuzzy ontologies from fuzzy XML models, *Knowl. Based Syst.*, **43** (2013), 20–39.
3. G. Sun, S. Su, *Formal analysis of the Kerberos authentication protocol with PVS*, in the Proceedings of 2013 AASRI Winter International Conference on Engineering and Technology, 2013, 202–206. Available from: https://www.atlantis-press.com/proceedings/aasri-wiet-13/10915.

4. P. Shen, X. Ding, W. Ren, *Research on Kerberos technology based on hadoop cluster security*, in Proceedings of the 2018 2nd International Conference on Advances in Energy, Environment and Chemical Science, 2018, 238–243. Available from: https://www.atlantis-press.com/proceedings/aeecs-18/25892275.
5. J. Sun, Z. Gao, *Improved mobile application security mechanism based on Kerberos*, in Proceedings of 2019 4th international workshop on materials engineering and computer sciences, (2019), 108–112. Available from: https://www.webofproceedings.org/proceedings_series/ESR/IWMECS%202019/IWMECS19017.pdf.
6. A. Ekelhart, S. Fenz, G. Goluch, M. Steinkellner, E. Weippl, XML security-a comparative literature review, *J. Syst. Software*, **81** (2008), 1715–1724.
7. H. Zhu, K. Lv, R. Jin, A practical mandatory access control model for XML databases, *Inf. Sci.*, **179** (2009), 1116–1133.
8. M. Smithamol, R. Sridhar, PECS: Privacy enhanced conjunctive search over encrypted data in the cloud supporting parallel search, *Comput. Commun.*, **126** (2018), 50–63.
9. W. Song, B. Wang, Q. Wang, Z. Peng, W. Lou, Y. Cui A privacy-preserved full-text retrieval algorithm over encrypted data for cloud storage applications, *J. Parallel Distrib. Comput.*, **99** (2017), 14–27.
10. S. Li, C. Xu, Y. Zhang, CSED: Client-side encrypted deduplication scheme based on proofs of ownership for cloud storage, *J. Inf. Secur. Appl.*, **46** (2019), 250–258.
11. G. Kalpana, P. V. Kumar, S. Aljawarneh, R. V. Krishnaiah, Shifted adaption homomorphism encryption for mobile and cloud learning, *Comput. Electr. Eng.*, **65** (2018), 178–195.
12. S. Ullah, X. Y. Li, M. T. Hussain, Z. Lan, Kernel homomorphic encryption protocol, *J. Inf. Secur. Appl.*, **48** (2019), 102366.
13. M. Alloghani, M. M. Alani, D. Al-Jumeily, T. Baker, J. Mustafina, A. Hussain, A systematic review on the status and progress of homomorphic encryption technologies, *J. Inf. Secur. Appl.*, **48** (2019), 102362.
14. T. Imamura, B. Dillaway, E. Simon, K. Yiu, M. Nyström, XML encryption syntax and processing, W3C Candidate recommendation, 2013. Available from: https://www.w3.org/TR/xmlenc-core1/.
15. M. Bartel, J. Boyer, B. Fox, B. LaMacchia, E. Simon, XML Signature Syntax and Processing, W3C Candidate Recommendation, 2013. Available from: https://www.w3.org/TR/xmldsig-core1/.
16. Z. Li, C. Chu, W. Yao, A semantic authorization model for pervasive healthcare, *J. Network Comput. Appl.*, **38** (2014), 76–87.
17. S. Shafeeq, M. Alam, A. Khan, Privacy aware decentralized access control system, *Future Gener. Comput. Syst.*, **101** (2019), 420–433.
18. L. Aliane, M. Adda, HoBAC: Toward a higher-order attribute-based access control model, *Procedia Comput. Sci.*, **155** (2019), 303–310.
19. C. Geuer-Pollmann, *XML pool encryption*, in Proceedings of the 2002 ACM Workshop on XML Security, 2003, 1–9. Available from: https://dl.acm.org/doi/abs/10.1145/764792.764794.
20. J. Lee, K. Y. Whang, W. S. Han, Y. Song, The dynamic predicate: Integrating access control with query processing in XML databases, *VLDB J.*, **16** (2007), 371–387.
21. E. Damiani, S. De Capitani, S. Paraboschi, P. Samarati, A fine-grained access control system for XML documents, *ACM Trans. Inf. Syst. Secur.*, **5** (2002), 169–202.

22. A. C. Duta, K. Barker, *P4A: A new privacy model for XML*, in Proceedings of the 22nd Annual IFIP WG 11.3 Working Conference on Data and Applications Security, 2008, 65–80. https://link.springer.com/chapter/10.1007/978-3-540-70567-3_6.
23. L. Guo, J. Wang, H. Wu, Application of Secret Sharing in XML Protection Mechanism, *Procedia Comput. Sci.*, **107** (2017), 21–26.
24. M. Wang, J. Wang, L. Guo, L. Harn, Inverted XML access control model based on ontology semantic dependency, *Comput. Mater. Continua*, **55** (2018), 465–482.
25. J. Sun, Z. Gao, *Improved mobile application security mechanism based on Kerberos,* Improved mobile application security mechanism based on Kerberos, 2019. Available from: https://www.webofproceedings.org/proceedings_series/ESR/IWMECS%202019/IWMECS19017.pdf.
26. H. Kaffel-Ben Ayed, B. Zaghdoudi, A generic Kerberos-based access control system for the cloud, *Ann. Telecommun.*, **71** (2016), 555–567.
27. R. Marin-Lopez, F. Pereñiguez-Garcia, Y. Ohba, F. Bernal-Hidalgo, A. F. Gomez, A Kerberized architecture for fast re-authentication in heterogeneous wireless networks, *Mobile Netw Appl*, **15** (2010), 392–412.
28. K. Juneja, An XML transformed method to improve effectiveness of graphical password authentication, *J. King Saud Univ. Comput. Inf. Sci.*, **32** (2020), 11–23.
29. F. Pereñíguez-García, R. Marín-López, G. Kambourakis, A. Ruiz-Martínez, S. Gritzalis, A. F. Skarmeta-Gómez, KAMU: Providing advanced user privacy in Kerberos multi-domain scenarios, *Int. J. Inf. Secur.*, **12** (2013), 505–525.
30. L. Guo, J. Wang, H. Wu, H. Du, eXtensible Markup Language access control model with filtering privacy based on matrix storage, *IET Commun.*, **8** (2014), 1919–1927.
31. L. Guo, J. Wang, H. Du, XML privacy protection model based on cloud storage, *Comput. Stand. Interfaces*, **36** (2014), 454–464.
32. R. Goldman, J. Widom, *DataGuides: Enabling query formulation and optimization in semi structured databases*, in Proceedings of the 23rd International Conference on Very Large Data Bases, San Francisco, 1997. Available from: https://www.researchgate.net/publication/2487373_DataGuides_Enabling_Query_Formulation_and_Optimization_in_Semistructured_Databases.
33. XML Path Language, 1998, Available from: http://en.wikipedia.org/wiki/XPath.
34. XML DOM Tutorial, W3C school, Available from: https://www.w3schools.com/XML/dom_intro.asp.
35. G. Miklau, XML data Repository, University of Washington, Available from: http://www.cs.washington.edu/research/xmldatasets/.
36. H. Rong, T. Ma, J. Cao, Y. Tian, A. Al-Dhelaan, M. Al-Rodhaan, Deep Rolling: A novel emotion prediction model for a multi-participant communication context, *Inf. Sci.*, **488** (2019), 158–180.
37. B. Al-Otibi, N. Al-Nabhan, Y. Tian, Privacy-preserving vehicular rogue node detection scheme for fog computing, *Sensors*, **19** (2019), 965–972.
38. Z. Pan, C. N. Yang, V. S. Sheng, N. Xiong, W. Meng, Machine learning for wireless multimedia data security, *Secur. Commun. Networks*, **2019** (2019), 7682306.
39. T. Ma, H. Rong, Y. Hao, J. Cao, Y. Tian, M. Al-Rodhaan, A novel sentiment polarity detection framework for Chinese, *IEEE Trans. Affect. Comput.,* **8** (2019), 61174–61182.

40. Y. Tian, M. M. Kaleemullah, M. A.Rodhaan, B. Song, A. Al-Dhelaan, T. Ma. A privacy preserving location service for cloud-of-Things system, *J. Parallel Distrib. Comput.*, **123** (2019), 215–222.
41. H. Yin, Z. Qin, J. Zhang, L. Ou, F. Li, K. Li, Secure conjunctive multi-keyword ranked search over encrypted cloud data for multiple data owners, *Future Gener. Comput. Syst.,* **100** (2019), 689–700.

AIMS Press