Journal of Science and Technology Volume 48, Issue 4, 2010 pp. 120-126

A COLUMN-LEVEL ACCESS CONTROL MECHANISM FOR DATABASE OUTSOURCING SERVICE

HUE T. B. PHAM, THUY T. B. DONG, AND THUC D. NGUYEN

ABSTRACT

Database outsourcing is emerging today as a successful paradigm allowing data owners to ship their data to the external service provider for the distribution of resources. An important problem to be addressed in this paradigm concerns the protection of outsourced data from unauthorized access even from the service provider’s server, which is not fully trusted. Several encryption schemes and access control mechanisms have been suggested to protect the outsourced data from unauthorized disclosure. However, by implementing these approaches, data owners are not capable of controlling and protecting the disclosure of the individual sensitive attributes of their data. Therefore, we propose a new column-level access control mechanism that is based on subkeys, which would allow a data owner to further control the access to his data at the column-level. We also propose a new mechanism to efficiently reduce the number of keys maintained by a data owner in cases when the users have different access privileges to different columns of the data being shared.

Keywords: Access control, column-level access control, database encryption.

1. INTRODUCTION

The amount of data held by organizations is increasing quickly and it often contains sensitive information. The management and protection of such data are expensive. An emerging solution to this problem is called *database as a service* (DAS), in which, an organization’s database is stored at an external service provider. The advantages of DAS are cost savings and service benefits. There are three main entities in the DAS scenario (Fig. 1):

- *Data owner:* individual or organization that is the subject of the data made available for controlled external use.
- *User*: individual or organization that requests data from the system.
- *Server*: organization that receives the data sent from the data owners and makes it available for distribution to clients.

DAS scenario can be classified into two main models [12]. The first one is SS model (Single user - Service provider) where a single data owner is also the unique client and their data is outsourced to the external database server. The second one is MMS model (Multiple data owner - Multiple clients Service provider) where multiple data owners outsource their databases to the service provider and multiple clients may also have access to the outsourced database on some agreement basis. The instances of MMS model are the two well-known electronic health record systems named Google Health and Microsoft HealthVault, in which patient’s health records are created, stored, used, exchanged, and retrieved over the Internet.

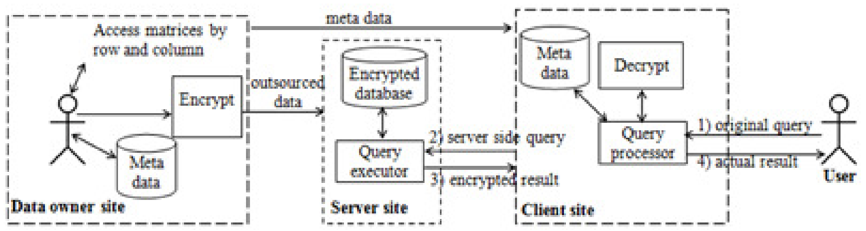

Figure 1. Block diagram of DAS scenario

In DAS scenario, however, the data owner store their data on an external server, which is typically not under their control, and it is assumed that this server cannot be trusted with the confidentiality of the database's content. It is important to protect data owner's data from unauthorized access by intruders and even the server's operators.

The MMS model is more complex than the SS model regarding to the needs of security requirements. Database encryption was regarded as a solution for protecting data from unauthorized disclosure. Existing encryption schemes assume that the client has complete access to the query results [6, 13, 14]. Such encryption schemes can only be applied for SS model. They do not fit MMS model in which the data owner often requires the enforcement of access restriction to different users. Some access control mechanisms have recently been proposed, but by using them, the data owners can only control access at the tuple-level and they are not able to restrict access to some of the more sensitive attributes of the data being shared [1, 3, 4, 11].

Despite the fact that health care providers such as Google Health and Microsoft Corp.'s HealthVault comply with the U.S Health Insurance Portability and Accountability Act (HIPAA), the privacy of patients is still at risk. Patients are required to trust the provider when using Microsoft HealthVault or Google Health. They cannot control whether the providers really adhere to their promises to protect their data and disclose personal data only to parties the patients have agreed to. They also cannot restrict access to their health records to some of the more sensitive columns.

The contributions of this study are: (a) a new access control mechanism called column-level access control, which would allow a data owner in MMS model to further control the access to his data at the column-level, (b) a new mechanism to efficiently reduce the number of keys maintained by the data owner when users have different access privileges to different columns of the data being shared.

2. RELATED WORK

Recent access control approaches combine cryptographic protection and authorization access control and enforce access control via selective encryption, which means users can decrypt only the data they are authorized to access [1]. They exploit a structure called the user tree hierarchy to represent the relationship between users and information items. The key for accessing lower-level nodes is based on the keys of their predecessors by using a family of one-way functions [10]. However, the complexity of the algorithm for building the tree hierarchy is exponential and it needs reconstruction after most of the modifications of the access control policies have been accomplished. Zych et al. [11] presented a key management scheme that was based on a partial order between the user groups and used the Diffie-Hellman key generation algorithm for the key derivation. The algorithm for constructing the hierarchy and the generation scheme are very complex and costly. El-khoury et al. [4] suggested a key management scheme that was based on the binary trie structure, using whichever keys ring for each group of users are generated. The key management complexity is reduced and most of the data access policy updates do not require significant changes to the structure, which reduces the number of key generations and data re-keyings. De Capitani di Vimercati et al. [3] proposed a two-layer access control mechanism that is based on the application of selective encryption. This solution has the benefits of being faster and less costly when the authorization policy is updated. However, all of the above-mentioned encryption schemes and access control mechanisms can only support tuple-level access control and fail to guarantee requirement 3. By using them, the data owners cannot restrict access to some of the more sensitive attributes of the shared data or have to partition the relation containing the shared data into fragments with the full share. This creates a lot of relations, and thus creates difficulties in effectively managing the database and query processing.

3. APPROACH FOR COLUMN-LEVEL ACCESS CONTROL

The database encryption with subkeys first proposed by Davida et al. [2] had the advantages of record orientation, security, and each field can be accessed under a different key. Using it, a user can read only some given fields depending on the readable subkeys he has without revealing all the attributes' values. We suggest using the encryption scheme proposed by Hacigümüs et al. [6] for MMS model, but with a modification of the encryption function. The encryption scheme with subkeys is used instead of the block cipher techniques, such as AES, RSA, or DES as in original scheme. For each relation $R(A_1, A_2, ..., A_n)$, the data owner stores encrypted relation $R^S($*etuple*$, A_1^S, A_2^S, ..., A_n^S)$ on the server, where the attribute *etuple* is the encryption form of a record using the encryption scheme with subkeys. The data owners follow these steps to protect their data:

Step 1: The data owners encrypt their data by tuples and thereby they grant access to users by tuples. The data owners first partition their data into disjoined row categories. Each row category consists of one or more rows. Users who have the same privileges to a row category are gathered into a single group. They describe the access policies to their data by using an access matrix, called an access matrix by row (**Error! Reference source not found.**).

Step 2: The data owners construct the binary trie for this row access matrix [4] to decide the number of keys that will be used to encrypt the data (Fig. 2). The data owner restricts the access to the sensitive columns by giving out only the subkeys corresponding to the columns that the user has access permission to. The key derivation method which is described in section 6, is used to derive the necessary keys (Table 2).

Step 3: If the users in a group have different access privileges to different columns of a row category, the data owner describes the access policies using other access matrices, and each is called an access matrix by column, and also manages the keys by using the binary trie. Users hold only some of the subkeys and derive the necessary ones to read the data with the granted permission. Appropriate keys are communicated to users by the data owner via a secure channel. Using our approach, the data owners can protect their data from unauthorized disclosure and they can restrict access to sensitive columns of the data being shared.

Table 1. Access matrix by row

	1	2	3	4	
1					
2					
3					
4					

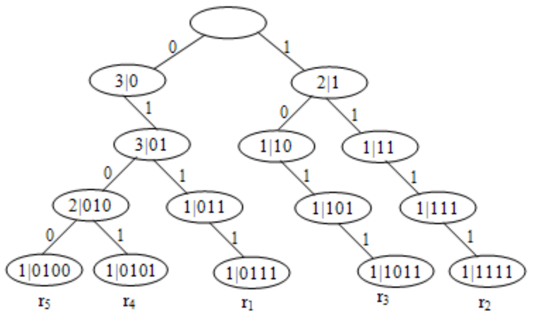

Figure 2. Binary trie structure for access matrix by row

Table 2. Keys held by each group of users and keys must be derivable

	Keys held	Keys derived
g_1	k_1	k_{1111}, k_{1011}
g_2	k_{01}, k_{11}	k_{0111}, k_{1111}, k_{0101}, k_{0100}
g_3	k_{011}, k_{101}, k_{111}	k_{0111}, k_{1111}, k_{1011}
g_4	k_{0111}, k_{1111}, k_{1011}, k_{0101}	

4. KEY MANAGEMENT FOR COLUMN-LEVEL ACCESS CONTROL

Database Encryption with Subkeys: There are variations when using the Chinese Remainder Theorem (CRT) for encrypting a database [2, 8, 7]. In DAS scenario, the convenience for a user to access data at a service provider is important, therefore an encryption scheme requiring less keys held by each valid user was chosen. We use an encryption with subkeys developed by Davida et al. [2] in DAS scenario. Let C_i be the ciphertext of an encrypted record, and let d_j be the reading subkey for field j. There are m records in a relation and n fields in each record. The encryption procedure is done by forming

$$C_i = \sum_{j=1}^{n} e_j \left(r_{ij} || x_{ij}\right) mod\ D. for\ i = 1, 2, \ldots,;$$

where $D = \prod_{j=1}^{n}$, x_{ij} is the value of field j of record i, || is the concatenation, r_{ij} is the random value for field j used for preventing attacks originated from using CRT, $r_{ij}\ ||x_{ij} < d_j$, e_j is the encryption key for field j, and b_j is the multiplicative inverse of D/d_j with moduli d_j. In addition, the decryption procedure is: $r_{ij}\ ||\ x_{ij} = C_i \bmod d_j$, $j = 1, \ldots, n$. By discarding the random bit r_{ij}, one can obtain the jth field data x_{ij} of record i.

Key Derivation Method: When using an encryption scheme with subkeys, the key of a security class SC_i consists of multiple different subkeys (d_{i1}, d_{i2}, ..., d_{in}), d_{ij} is a prime and also the decryption key used for decrypting the jth attribute. The key of security class SC_j, which is an immediate child of SC_i, is computed by $(f_1(d_{i1}), f_2(d_{i2}), ..., f_n(d_{in}))$, and each f_i is a one-way function [10]. The key of the security class SC_j also consists of multiple subkeys, each subkey must be a prime too (for properly using the CRT). So, each function f_i must be a one-way function that receives a prime number as the input and outputs a prime number. Please see the appendix for a method to construct this function.

Column-level Access Control: If a user group has full access authorization to all the columns of a row category, each user in this group holds all the subkeys of each key assigned to each group. By derivation, they obtain the actual keys to access all the columns of the row category with authorization. If a user group has access authorization to only some attributes of a row category, and the access authorizations are the same to all the members of the group, they are given appropriate subkeys to access the corresponding columns. In the cases when different users in a user group *G* have different access privileges to different columns of a row category, the data owner describes these privileges by using other access matrices, each is called access matrix by column. The number of rows of this matrix is equal to the number of subgroups of users in the group *G* and the number of columns it has equals to the number of groups of columns with different access authorizations. We construct the binary trie structure corresponding to this column access matrix. Each subgroup of users holds just some subkeys. These subkeys can then be used to derive other subkeys in order to access the other columns with granted permission.

Key Management: In our proposed access control mechanism, each user owns some subkeys (which are used to derive other subkeys) to access the fields he has access authorization for. Management of the number of subkeys is difficult for users. We propose using the key management method in [7] to handle this problem.

(1) Assigns each field a public prime number p_j, $j = 1, 2, \ldots, n$.

(2) Computes the secret master key MK_i by CRT for user i:

$MK_i = d_j \bmod p_j$ for some j $1 \leq j \leq (n+1)$ where d_{n+1} and p_{n+1} are random secret key and the prime number of a dummy field used to prevent users from colluding to disclose the secret master key of user i.

User i keeps only the secret master key MK_i.

(3) To read the field j, user i computes $d_j = MK_i \bmod p_j$ to get subkey d_j.

5. CONCLUSION AND FUTURE WORK

We have proposed a new column-level access control mechanism based on subkeys. Our proposal has the following characteristics:

(1) It allows a data owner in the complex model of database outsourcing service, MMS model, to further control the access of their data at the column-level.

(2) It efficiently reduces the number of keys maintained by a data owner when users have different access privileges to different columns of the data being shared. These subkeys are encrypted one time more (using CRT) so that each user holds only one secret master key, but they can derive the necessary subkeys to access data with authorization.

For future work, we will investigate the dynamic cases of access control in which the users and the privileges changed frequently. We will also analyze the system's performance when we implement our proposals.

REFERENCES

1. Damiani E., De Capitani di Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, and S. Samarati P. - Key Management for Multi-User Encrypted Databases. Proc. of the 2005 ACM Workshop on Storage Security and Survivability, 2005, pp.74-83.
2. Davida G. I., Wells D. L., and Kam J. B. - A Database Encryption System with Subkeys. ACM Transactions on Database Systems **6** (2) (1981) 312-328.
3. De Capitani di Vimercati S., Foresti S, Jajodia S., Paraboschi S., and Samarati P. - Over-encryption: Management of Access Control Evolution on Outsourced Data, VLDB, 2007, pp. 123-134.
4. El-khoury V., Bennani N., and Ouksel A. M. - Distributed Key Management in Dynamic Outsourced Databases: a Trie-based Approach, First Int. Conf. on Advances in Databases, Knowledge, and Data Applications, 2009, pp. 56-61.
5. Google, Health Privacy Policy, http://www.google.com/intl/en-US/health/privacy.html
6. Hacigümüs H., Iyer B. R., Li C., and Mehrotra S. - Executing SQL over encrypted data in the database-service-provider model, in SIGMOD, 2002, pp. 216-227.
7. Hwang M. S., and Yang W. P. - A Two-Phase Encryption Scheme for Enhancing Database Security, J. Systems Software, Elsevier Science (1995) 257-265.
8. Lin C. H., Chang C. C., and Lee C. T. - A record-oriented cryptosystem for database sharing, in Int. Computer Symposium, 1990, pp. 328-329.
9. Microsoft, HealthVault Privacy Policy,

   https://account.healthvault.com/help.aspx?topicid=PrivacyPolicy (2009)
10. Sandhu R. S. - Cryptographic implementation of a Tree Hierarchy for access control, Elsevier, 1988, pp. 95-98.
11. Zych A., Petkovic M., and Jonker W. - Efficient key management for cryptographically enforced access control, Elsevier Science, 2008, pp. 410-417.
12. Khanh, D. T. - Security Issues in Outsourced XML Databases. Open and Novel Issues in XML Database Applications: Future Directions and Advanced Technologies, IGI Global, ISBN: 978-1-60566-308-1, 2009, pp. 231-261.
13. Damiani E., Vimercati S. D. C., Jajodia S., Paraboschi S., and Samarati P. - Balancing confidentiality and efficiency in untrusted relational DBMSs. Proceedings of the 10th ACM Conference on Computer and Communication Security, Washington DC, USA, 2003, pp. 93-102.
14. Lin P., and Candan, K. S. - Hiding traversal of tree structured data from untrusted data stores. Proceedings of the 2nd International Workshop on Security in Information Systems, Porto, Portugal, 2004, pp. 314-323.

APPENDIX

We present a method for constructing a one-way function that receives a prime as an input and gives a prime as an output. Using this algorithm, the users with the same privileges will generate the same keys at different times for deriving keys to access the specific data.

Theorem. *Assume that* $n - 1 = F \times R = (\prod_{i=1,\ldots,s} p_i^{a_i}) \times$*, where* $R <$ *, and an integer b such that* $b^{n-1} \equiv 1 \pmod{n}$ *exists and* $\gcd(b^{(n-1)/p_i} - 1, n) =$*,* $i=1,\ldots, s$*. Then n is a prime.*

Algorithm G*:* Generating a prime of d digits with a given seed a.

(1) Generate a Q that has d-2 digits using the consecutive hash values of seed a: Q=h(a)||…||h(a), where || is the concatenation operator.

(2) Find the greatest R such that (R < Q) and $(1 + RQ \equiv 1 \pmod{6})$.

If p = RQ + 1 is a pseudo-prime with base 2 and 3,

(3) then return p.

(4)R = R+6.

(5)Goto (3).

Generating key function

Let h be a one-way hash function h: $Z_n \to H$, i.e. given $y \in H$, it is hard to find $x \in Z_n$ such that h(x)=y, where H is the hash value space. For example, MD5 and SHA are two popular one-way hash functions. Let g be a generating prime function g: $H \to P$, where P is a prime space. Let f: $Z_n \to P$ be defined by f =goh, then f is a one-way function. Indeed, given y=f(x)=g(h(x)), because h is a one-way hash function, it is hard to find $x \in Z_n$ such that y=f(x)=g(h(x)). To generate a prime, we use the combination of the above generating prime function and the hash function SHA.

Algorithm F: Generating key based on a given prime p.

(1) Compute a = SHA (p).

(2) Generate a prime q using the algorithm G with seed a.

Address:

Faculty of Information Technology,

University of Science, VNU – HCMC.

Received June 16, 2010