A Secure and Cost-effective Framework for Semi-trusted Database

Lwin Mar Thin, Nan Sai Moon Kham
University of Computer Studies, Yangon
lwinmarthin85@gmail.com,moonkhamucsy@gmail.com

Abstract

Database security has become a vital issue in modern Web applications. Critical business data in databases is an evident target for attack. Database encryption has become a critical technique in the security mechanisms of database. This paper proposed a secure framework for semi-trusted database in which data is encrypted in a mixed form using many keys owned by different parties. That can provide security and cost-effective by reducing the server tasks required to perform encrypt and decrypt operations. The framework aims to ensure the requirements of secure semi-trusted database include; Data Confidentiality and User Data Privacy and Data Integrity. The proposed framework is strengthening the protection of sensitive data even if the database server is attacked at multiple points from the inside or outside.

Keywords: semi-trusted database, database encryption, security, key, cryptography

1. Introduction

Nowadays, with the tremendous development of the Internet, new challenges have been posed in information security. In most organizations, databases hold a critical concentration of sensitive information, and as a result, databases are vulnerable. So database system should be protected from any attacks. Today, enhancing the security of database is becoming one of the most urgent tasks in database research and industry.

Recently, the number of reported data involving sensitive private information at governmental, organizational and company levels has grown at an alarming rate. In some extreme cases, sensitive information belonging to millions of individuals has been revealed. For example, in May 2008, researchers at security vendors uncovered a server containing the sensitive email and Web-based data of thousands of people, including healthcare information, credit card numbers and business personnel documents and other sensitive data (www.searchsecurity .techtarget.com/news).

Generally, database security methods could be divided into four layers [2]: physical security, operation system security, DBMS security and database encryption. These layers protect database in different aspects. But only first three layers are inadequate to protect the confidential data in database satisfactorily, because data is still stored in readable form. So without database encryption, it makes no sense to guarantee that the sensitive information in plaintext will be protected against a malicious user. Therefore two main issues both need to be considered: secure data storage and secure data transmission.

There are three approaches to database servers where encryption takes place: first, the trusted database server where the creator, or owner, of the data operates a database server, which processes queries and signs the results; second, the untrusted server where the owner’s database is stored at the service provider (Database as a Service). The third and final model called the semi-trusted server where the database is shared between many parties. Here, part of the data is stored as trusted while other parts are considered untrusted.

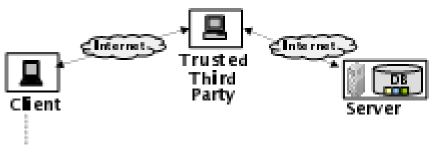


Figure. 1. A semi-trusted database system

In traditional client server based encryption, the data is encrypted using either a server key in a trusted database or a client key in an untrusted database. Simply, if the data is encrypted using server key(s) and the administrator has the authority to use this key(s), then the whole system becomes vulnerable. On the other hand, if each client encrypts his data using his key, then many problems might appear, such as how can organizations use the data? Using server-based encryption or client-based encryption is not sufficient to encrypt semi-trusted database where data is shared by many parties. The proposed system is ensuring that confidentiality, privacy and integrity are achieved for semi-trusted database where data is exchanged over an untrusted network such as the Internet.

The rest of the paper is organized as follows: section 2 describes the related work. Section 3 discusses the adversary model and determines the security threats in terms of security attributes. Section 4 presents database encryption scheme. Section 5 describes the proposed framework and gives more detail on its components. Section 6 concludes the paper.

2. Related Work

In the realm of cryptography, Diffie and Hellman [10] proposed the concept of public key cryptography in their landmark paper in 1976. This new concept brings revolutionary changes to cryptographic schemes, and establishes the foundations of a new paradigm and secure database systems. In 1978, Rivest, Shamir and Adleman [5, 6] published RSA key-pair scheme, the first public key system which is widely used in data encryption now.

In terms of encryption scheme research for relational database management system, many creative and efficient schemes have been proposed. Davida, Wells and Kam [3] in their 1981 paper presented a database encryption scheme with subkeys based on the Chinese Reminder theorem. This scheme is a record-oriented cryptosystem which enables encryption at the levels of rows and decryption at the level of cells. Using symmetric encryption, Ge and Zdonik [9] proposed a database encryption schema called FCE for column stores in data warehouses with a trusted server.

Recent years have been extensive database cryptography research conducted in the area of Database as a Service (DAS). The idea behind this research is to encrypt data so that it becomes accessible only to the client, because the data belongs to the client and is stored on an untrusted server. Bouganim and Pucheral [7] proposed chipped secured data access solution C-SDA, which enforces data confidentiality and controls personal privileges with a client based security component acting as a mediator between the client and an encrypted database. This component is embedded in a smartcard to prevent tampering.

The previous solutions aim to encrypt the organization data and information by using strong encryption algorithm that decreases the probability of data and information compromise during the transition of data and information between the client and server, even if the server is malicious [8]. Dong, Russello and Dulay [1] published Shared and sharable encrypted data for untrusted servers, its multi-users searchable data encryption, and each user will be able to encrypt and decrypt the inserted data by other users without need to know the users keys.

The database encryption operation in database cryptography research is done by using either by server or client key(s) using different algorithms. Kadhem, Amagasa, and Kitagawa [4] proposed a framework to realize database security in the semi-trusted server scenario where the data in databases is shared by many parties. MCDB has lower query performance compared with plaintext database, server-based, and client-based encrypting because of multi owner and mixed encryption. So, this paper proposed a secure and cost-effective framework for semi-trusted database which may provide a proper query performance by reducing the server tasks and encryption/decryption layer.

3. Security Issues

Data owners have to face the following potential internal and external attackers, especially when data management is entrusted to a trusted third party:

- **Internal Adversaries**: From the viewpoint of data management, internal adversaries are authorized parties who abuse their extended privileges to access confidential data.
- **External Adversaries**: An external adversary is a person who may access the database directly or indirectly with the objective of achieving personal gain or causing harm to data.

We categorize the threats emanating from these adversaries with the following security attributes:

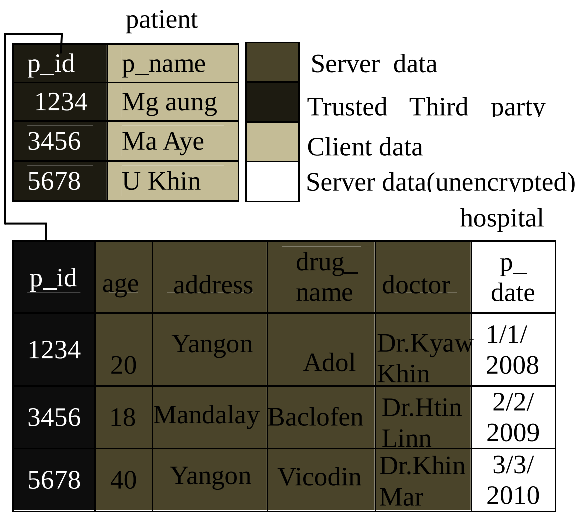
1. **Confidentiality**: Generally, data confidentiality is the protection of private information from surveillance or leaks when it is stored, or is transmitted across vulnerable networks such as the Internet.
2. **Privacy**: Data privacy is the prevention of confidential or personal information from being viewed by parties and the control over its collection, use and distribution.
3. **Integrity**: Integrity refers to the property of data records to be manipulated only by authorized users. Specific aspects of data integrity include data correctness (no unauthorized modification of data records) as well as data completeness (no unauthorized deletion or insertion of (fake) data records).

4. Database Encryption

In most databases used on the Web, data is stored in tables in the form they are loaded, mostly in plaintext, which does not satisfy high level security and privacy requirements. The proposed system, whose basic architecture is shown in Figure 1, adds encryption/decryption layers accumulatively while data moves from/to the database. The purpose of such design is to implement encrypted storage satisfying data confidentiality, privacy and integrity. Following the convention, E denotes the encryption function and D denotes that decryption function. Data is encrypted using symmetric and asymmetric algorithms.

Definition1: For each database schema S(R_1,….,R_n), where R_1 to R_n are relations created on the database server, R_i name is encrypted using the Trusted third party secret key SE_T.

Definition2: For each relation schema R(F_1,…,F_l, F_{l+1},….,F_m, F_{m+1},….,F_n) in a relational database, where F_i(1≤i<l) field is classified as trusted third party data, F_j (l≤j>m)field is classified as client data, and F_k (m≤k<n) field is classified as server data, we store an encrypted relation: R^E=(F_1^T,….,F_l^T, F_{l+1}^C,….,F_m^C, F_{m+1}^S,….,F_n^S) where, F_i^T(1≤i<l) in the encrypted relation R^E stores the encrypted value of F_i in relation R using trusted third party secret key SE_T, viz. $F_i^T = E^{SE_T}$ (F_i), F_j^C (l≤j<m) in the encrypted relation R^E stores the encrypted value of F_j in relation R using trusted client secret key SE_C , viz. $F_j^C = E^{SE_C}$ (F_j), F_k^S (m≤k<n) in the encrypted relation R^E stores the encrypted value of F_k in relation R using server secret key SE_S, viz. $F_K^S = E^{SE_S}$ (F_k). Field names are encrypted as follows: F_i and F_j names are encrypted using trusted third party secret key SE_T. With many clients, it is impossible to encrypt field names of client data using a client secret key. Figure 2 shows an unencrypted hospital database and Figure 3 shows an encrypted database.



patient

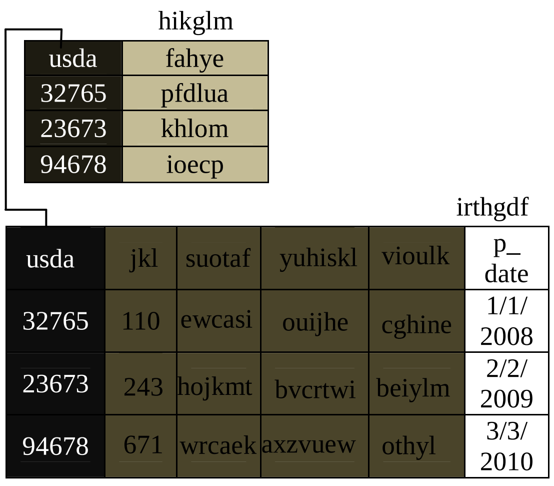
p_id	p_name
1234	Mg aung
3456	Ma Aye
5678	U Khin

▨	Server data
■	Trusted Third party
▧	Client data
□	Server data(unencrypted)

hospital

p_id	age	address	drug_ name	doctor	p_ date
1234	20	Yangon	Adol	Dr.Kyaw Khin	1/1/ 2008
3456	18	Mandalay	Baclofen	Dr.Htin Linn	2/2/ 2009
5678	40	Yangon	Vicodin	Dr.Khin Mar	3/3/ 2010

Figure. 2. Unencrypted database



hikglm

usda	fahye
32765	pfdlua
23673	khlom
94678	ioecp

irthgdf

usda	jkl	suotaf	yuhiskl	vioulk	p_ date
32765	110	ewcasi	ouijhe	cghine	1/1/ 2008
23673	243	hojkmt	bvcrtwi	beiylm	2/2/ 2009
94678	671	wrcaek	axzvuew	othyl	3/3/ 2010

Figure. 3. Encrypted database

5. The Proposed Framework

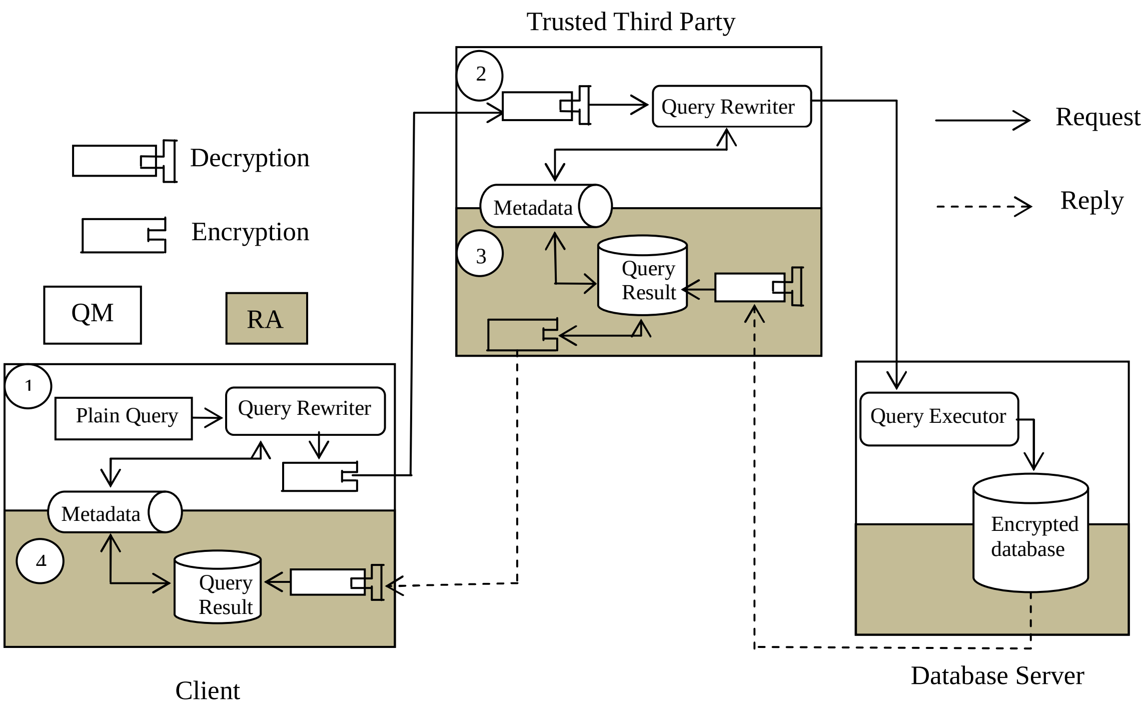


Figure. 4. The framework of the proposed System

The proposed system based on semi-trusted database, which means there is certain type of data that related of database server and it assume confidential data and the other is not. This system consists of three components: **Client, Trusted Third Party (TTP) and Server**. Each client and TTP party has responsible two parts. This is the Query Management and Result Analyzer. The main objective of QM is to encrypt, decrypt and rewrite the query request. The objective of RA is to decrypt the query result to make it read able for the client. The client side makes query request to retrieve, update, insert and delete data. The role of the trusted third party is to organize query requests and replies the results to the clients. The server contains the encrypted database and is responsible for replying to query requests on the encrypted database. Each of these parties owns his keys for encryption and decryption, the master keys able to be stored by Trusted Third Party and patient smart card.

This system could be used in many applications over the Web such as e-government, e-commerce and e-banking, where databases hold critical and sensitive information transferred over untrusted networks like the Internet. In this system discussed a hospital database as an illustrative example based on the proposed framework. An attacker trying to compromise patient records will be able to collect large amounts of data easily if these records are available electronically as plaintext. And if this information is encrypted using only sever keys, it is an easy target for inside attack. Therefore, protection of a patient's privacy is a basic requirement for ethical and legal use of information technology in health information systems; protecting confidential information related to hospital is also crucial.

The proposed framework is shown in Figure 4. This is a secure and cost-effective framework for semi-trusted database and provides protection for sensitive data even if the database server is attacked at multiple points by an inside or outside attacker.

5.1. Client

Database clients are actually the data users, who read, write and modify the database records according to their privileges. The clients may database-authorized employees in organization or a client such as patient. To ensure user privacy, the data such as name is encrypted by client key.

```
//1.Client rewrites the query
Input: plaintext query ($Q_p$)
Output: client query ($Q_c$)
Begin
    For each field ($F_i$ ) owned by client in ($Q_p$)
    Do
        If value ($V_i$) assigned to ($F_i$) then
            Replace ($V_i$) with $E^{SEc}$ (Vi)
        End if
    End for
End
```

Figure.5.Query Rewriter Algorithm for Client

Example for query rewriter in client

SELECT patient.p_name,hospital.age, hospital. address, hospital.drug_name, hospital.doctor, hospital.date
FROM patient, hospital
WHERE patient.p_id = hospital.p_id and patient.p_id=1234

```
//4. Client analyzes the query results
Input: dataset from TTP ($DS_t$)
Output: client dataset ($DS_c$) (plaintext)
Begin
    For each encrypted field ($F_i$) in ($DS_t$) Do
        decrypt field name $F_i = D^{PRc}$ ($F_i$)
    End for
    For each field ($F_i$) owned by client in ($DS_t$)
    Do
        decrypt field data $F_i = D^{SEc}$ ($F_i$)
    End for
    For each encrypted field ($F_i$) not owned by
    client in ($DS_t$) Do
        decrypt field data $F_i = D^{PRc}$ ($F_i$)
    End for //fields owned by server or TTP
End
```

Figure.6.Result Analyzer Algorithm for Client

5.2. Trusted Third Party (TTP)

Is the middle server owned by the organization (database owner), contains Metadata of the database and keys table. The data used to joins and coordinates between server and clients such as Id number is encrypted by TTP. This is enabling to prepare the complete query request form to retrieve results in database server.

//2. TTP rewrites the client query

```
Input: client query ($Q_c$)
Output: TTP query ($Q_t$)
Begin
    For each field ($F_i$) owned by client in ($Q_c$)
    or TTP Do
        replace ($F_i$) with $E^{SEt}$ ($F_i$)
    End for
    For each field ($F_i$) owned by server in ($Q_c$)
    Do
        replace ($F_i$) with $E^{SEs}$ ($F_i$)
        If value ($V_i$) assigned to ($F_i$) then
            replace ($V_i$) with $E^{SEs}$ ($V_i$)
        End if
    End for
    For each field ($F_i$) owned by TTP in ($Q_c$)
    Do
        If value ($V_i$) assigned to ($F_i$) then
            replace ($V_i$) with $E^{SEt}$ ($V_i$)
        End if
    End for
    For each relation ($R_i$) in ($Q_c$) Do
        replace ($F_i$) with $E^{SEt}$ (Fi)
    End for
End
```

Figure. 7. Query Rewriter Algorithm for TTP

Example for query rewriter in TTP

```
SELECT hikglm.fahye, irthgdf.jkl, irghtdf.suotaf, irthgdf.yuhjskl, irthgdf.yioulk, irthgdf.p_date
FROM hikgl, irghtdf
WHERE hikglm.usda = irthgdf.usda and hikglm.usda=76532
```

//3.TTP analyzes the query results

```
Input: dataset from server ($DS_s$)
Output: TTP dataset ($DS_t$)
Begin
    For each encrypted field ($F_i$) owned by
    server in ($DS_s$) Do
        decrypt field name $F_i = D^{SEs}$ ($F_i$)
        decrypt field data $F_i = D^{SEs}$ ($F_i$)
        encrypt field name and data = $E^{PUc}$ ($F_i$)
    End for
    For each encrypted field ($F_i$) owned by TTP
    in ($DS_s$) Do
        decrypt field name and data $F_i = D^{SEt}$ (Fi)
        encrypt field name and data $F_i = D^{PUc}$
        ($F_i$)
    End for
    For each encrypted field ($F_i$) owned by client
    in ($DS_s$) Do
        decrypt field name $F_i = D^{SEt}$ ($F_i$)
        encrypt field name $F_i = E^{PUc}$ ($F_i$)
    End for
End
```

Figure.8. Result Analyzer Algorithm for TTP

5.3. Database Server

The server belong the organization or company data, which is responsible for collecting and maintaining it. And the server host what is owned by several parties, such as clients and intermediate party, i.e., they contain data related to clients and intermediate party.

6. Experiment

For this section, we will conduct empirical study on how query processing performance differs among our proposed approach, client-based approach and server-based encryption approach. The experiments will conduct by implementing semi-trusted database over Microsoft SQL Server 2010. The algorithms will implement in ASP.NET as a web services.

7. Conclusion

Database Security is a major issue in any web-based application. No matter what degree of security is put in place, sensitive data in database are still vulnerable to attack. To avoid the risk posed by this threat, database encryption has been recommended. Encryption is a well established technology for protecting sensitive data. In this paper, we present a secure and cost-effective framework for semi-trusted database where database are encrypted in a mixed form by using many keys owned by different parties. This framework supports the three main security components in the database (Data Confidentiality and Data Privacy and Data Integrity).

References

[1] C. Dong, G. Russello, N. Dulay, “Shared and Searchable Encrypted Data for Untrusted Servers”, 22 edition, Springer, 2008.

[2] E. B. Fernandez, R . C. Summers and C. Wood, Database Security and Integrity, Addison Wesley, Massachusetts, 1980.

[2] G.I. Davida, D.L. Wells, and J.B. Kam, "A Database Encryption System with Subkeys", ACM Trans. Database Syst., 1981, pp. 312-328.

[4] H. Kadhem, T.Amagasa, and H.Kitagawa, “A novel framework for database security based on mixed cryptography”, International Conference on Internet and Web Applications and Services, 2009, 163–170.

[5] L. A. R. Rivest and A. Shamir, "A method for obtaining digit signatures and public-key cryptosystems", Communications of the ACM, Date, pp. 120-126.

[6] L. A. R. Rivest and A. Shamir, “On digit signatures and public key cryptosystems”, Technical Report MIT/LCS/TR-212, MIT Laboratory for Computer Science, Location, 1979.

[7] L. Bouganim and P. Pucheral. “Chip-secured data access: confidential data on untrusted servers”. In VLDB’02: Proceedings of the 28th international conference on Very Large DataBases, pages131–142.VLDB Endowment, 2002.

[8] M. Gertz, “Hand Book of database Security application and trends", Springer Science Business Media, LLC. 2008.

[9] T. Ge and S. Zdonik. “Fast, secure encryption for indexing in a column-oriented DBMS”, In Data Engineering, 200, ICDE 2007, IEEE 23rd International Conference on, pages 676–685, 2007.

[10] W. Diffie and M.E. Hellman, “New Directions in Cryptography”, IEEE Trans. Inform. Theory, 1976, IT-22:644-654.