A SECURE QUERY MODEL FOR ENCRYPTED DATABASE

NONGNUCH POOLSAWAD

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR
THE DEGREE OF
MASTER OF SCIENCE (COMPUTER SCIENCE)
FACULTY OF GRADUATE STUDIES
MAHIDOL UNIVERSITY
2006

COPYRIGHT OF MAHIDOL UNIVERSITY

Copyright by Mahidol University

Thesis
Entitled

A SECURE QUERY MODEL FOR ENCRYPTED DATABASE

Miss Nongnuch Poolsawad
Candidate

Lect. Sudsanguan Ngamsuriyaroj, Ph.D.
Major-Advisor

Assoc.Prof. Damras Wongsawang, Ph.D.
Co-Adviser

Lect. Songsri Tangsripairoj, Ph.D.
Co-Adviser

Prof. Dr. M.R. Jisnuson Svasti, Ph.D
Dean
Faculty of Graduate Studies

Assoc. Prof. Supachai Tangwongsarn, Ph.D.
Chair
Master of Science Programme
in Computer Science
Faculty of Science

Copyright by Mahidol University

Thesis
Entitled

A SECURE QUERY MODEL FOR ENCRYPTED DATABASE

was submitted to the Faculty of Graduate Studies, Mahidol University
For the degree of Master of Science (Computer Science)
on
November 3, 2006

Miss Nangnach Poolsawad
Candidate

Miss Thidima Srivatanakul,
Ph.D.
Chair

Lect. Sudsanguan Ngamsuriyaroj,
Ph.D.
Member

Lect. Songsri Tangsripairoj,
Ph.D.
Member

Assoc.Prof. Damras Wongsawang,
Ph.D.
Member

Prof. Dr. M.R. Jisnuson Svasti,
Ph.D.
Dean
Faculty of Graduate Studies
Mahidol University

Prof. Amaret Bhumiratana,
Ph.D.
Dean
Faculty of Science
Mahidol University

Copyright by Mahidol University

ACKNOWLEDGEMENT

First of all, I would like to express my gratitude and deep appreciation to my thesis advisor, Dr. Sudsanguan Ngamsuriyaroj, for her support by contributing her valuable time for guidance, invaluable advices and encouragement throughout this study, despite spending a lot of time to study for the purpose of giving accurate and helpful advice which greatly sustains me to achieve this thesis.

I also would like to thank Dr. Damras Wongsawang, Dr. Songsri Tangsripairoj and Dr. Thitima Srivatanakul for serving as the committee of my thesis.

In additional, I would like to thank all of Mahidol University Computing Center staff for their kindness and assistance, and for providing good facilities during my study at Mahidol University.

Lastly, I am very grateful to my family, especially for my parents for their financial support, encouragement and love. Without them, I may not reach the goal of my study life and also, I would not be able to come up to this point where pride and proud are found.

Nongnuch Poolsawad

Copyright by Mahidol University

A SECURE QUERY MODEL FOR ENCRYPTED DATABASE

NONGNUCH POOLSAWAD 4637276 SCCS/M
M.Sc.(COMPUTER SCIENCE)

THESIS ADVISORS: SUDSANGUAN NGAMSURIYAROJ, Ph.D., DAMRAS WONGSAWANG, Ph.D., SONGSRI TANGSRIPAIROJ, Ph.D.

ABSTRACT

With widespread use of databases in organizations and businesses, sensitive data in a database could be easily accessed by malicious or unauthorized users. Hence, security and privacy issues have become indispensable ingredients to the survival of any organization. However, treating security issues as an afterthought often result in inefficiency and non-secure design and implementation. In addition, high level security is likely to conflicts with high efficiency.

Encryption over sensitive data could provide a way to protect databases from unauthorized accesses including those from the database administrator and service providers. It also puts some limitations on data access since appropriate keys must be used for decrypting those data. But since most encryption techniques have high overheads and are complicated, the encryption usually degrades database performance. Generally, the cost of applying encryption to the database depends on data granularity, the encryption algorithm itself, and the number of encryptions performed.

In this thesis, the trusted query model for encrypted database where is proposed only sensitive attributes are encrypted and the role-based security policy is enforced via the key manager who maintains the authority table and the key table for allowing authorized accesses according to the security policy defined by the organization executives. The experimental results illustrate that the encryption plays a significant role in the query response time since the query response time of selective encrypted attributes in this model is much lower than that of the whole encrypted database. Thus, this model can be practically applied to real-world applications where few sensitive attributes are focused, and the database performance is not greatly reduced.

KEYWORDS: DATABASE ENCRYPTION / QUERY MODEL/ MULTIPLE ENCRYPTIONS
69 P.

Copyright by Mahidol University

ตัวแบบสอบถามที่ปลอดภัยของฐานข้อมูลเข้ารหัส (A SECURE QUERY MODEL FOR ENCRYPTED DATABASE)

นงนุช พูลสวัสดิ์ 4637276 SCCS/M
วท.บ. (วิทยาการคอมพิวเตอร์)

คณะกรรมการควบคุมวิทยานิพนธ์ : สุดสงวน งามสุริยโรจน์, ดำรัส วงศ์สว่าง, ทรงศรี ตั้งศรีไพโรจน์

บทคัดย่อ

เนื่องจากปัจจุบันมีการใช้ข้อมูลกันอย่างกว้างขวางทั้งในหน่วยงานและในภาคของธุรกิจทั้งข้อมูลที่สำคัญและเป็นความลับยังเป็นสิ่งที่ถูกเข้าถึงจากบุคคลที่ไม่มีสิทธิ์ในข้อมูลนั้นหรืออาจจะโดนคุกคามจากผู้ที่ไม่ประสงค์ดีที่ต้องการล้วงความลับภายในหน่วยงานต่างๆ จึงทำให้ประเด็นในเรื่องของ data privacy และ database security ถูกหยิบยกมาเป็นประเด็นสำคัญและน่าสนใจประเด็นหนึ่งในปัจจุบัน ซึ่งด้วยเหตุนี้ เรื่อง security จึงเป็นสิ่งที่ขาดไม่ได้ใน DBMS อย่างไรก็ตาม การที่จะรักษาความปลอดภัยของข้อมูลนั้น ต้องคำนึงถึงประสิทธิภาพ เนื่องจากระดับของความรักษาปลอดภัยของข้อมูลนั้นจะส่งผลตรงกันข้ามกับประสิทธิภาพของระบบด้วย

วิธีการรักษาความปลอดภัยของข้อมูลวิธีการหนึ่งคือ การ encryption ซึ่งวิธีการนี้จะป้องกันผู้จู่โจมได้ แม้ว่าจะเป็น DBA หรือ service provider เองก็ตาม แต่การทำ encryption นั้นมีข้อจำกัดในส่วนของการเข้าถึงข้อมูล เนื่องจากต้องสนใจในเรื่องของ key ที่ใช้ในการถอดรหัสข้อมูลแล้วยังมีขั้นตอนในการทำมีความซับซ้อนและทำให้เกิด overhead ค่อนข้างสูง ซึ่งโดยปกติแล้วการทำ encryption นั้นจะส่งผลให้ประสิทธิภาพของระบบลดต่ำลง โดย overhead ที่เกิดขึ้นส่วนใหญ่จะเกิดจาก ระดับของการเลือก encrypt ข้อมูล การเลือก encryption algorithm และจำนวน operation ในการทำ encryption เป็นต้น

ในงานวิจัยนี้ ได้นำเสนอ secure query model สำหรับการ encrypt เฉพาะ sensitive data และใช้ role-based security policy เพื่อการจัดการ key เพื่อใช้ในการ encrypt/decrypt ข้อมูล และตลอดจนการตรวจสอบสิทธิ์ในการเข้าถึงข้อมูล จากผลการทดลองได้ผลว่า response time ของแนวความคิดที่นำเสนอให้ผลที่ดีกว่าการ encrypt ทุก field ในฐานข้อมูล ดังนั้นในการที่จะตัดสินใจเลือกข้อมูลในเพื่อที่จะทำการ encrypt นั้น ควรเลือกเฉพาะข้อมูลที่มีความสำคัญและจำเป็นจริงๆเท่านั้น ดังนั้นใน model ที่นำเสนอจึงเหมาะสำหรับระบบที่เป็น real-world application ซึ่งมี sensitive data เพียงแค่บาง attribute และคำนึงถึงประสิทธิภาพของ database

69 หน้า.

Copyright by Mahidol University

CONTENTS

Page

ACKNOWLEDGEMENT……...iii
ABSTRACT……...iv
LIST OF TABLES…..…….viii
LIST OF FIGURES..…..……ix
CHAPTER

I INTRODUCTION ……..1
1. PROBLEM STATEMENT
2. OBJECTIVES OF THE THESIS
3. SCOPE OF THE THESIS
4. ORGANIZATION OF THE THESIS

II BACKGROUND..……...5
1. QUERY PROCESSOR
2. ENCRYPTION
3. LEVEL OF GRANULARITY
4. CERTIFICATE AUTHORITY
5. ACCESS CONTROL
6. ROLE-BASED ACCESS CONTROL (RBAC)

III RELATED WORK ……..…….15
1. EISA – AN ENTERPRISE APPLICATION SECURITY SOLUTION FOR DATABASES
2. EFFICIENT KEY UPDATES IN ENCRYPTED DATABASE SYSTEM
3. ROLE-BASED ACCESS CONTROL (RBAC) SYSTEM OF A EUROPEAN BANK

Copyright by Mahidol University

CONTENTS (CONT.)

Page

IV PROPOSED WORK…….26
1. OVERVIEW OF THE PROPOSED MODEL
2. PROPOSED QUERY MODEL
3. DATA ENCRYPTION
4. QUERY PROCESSING OVER ENCRRYPTED DATA
5. QUERY CLASSIFICATION
6. DESIGNING SAMPLE QUERIES
V IMPLEMENTATION AND EXPERIMENTS…….52
1. SYSTEM ARCHITECTURE
2. SYSTEM REQUIREMENTS
3. EXPERIMENT SETUP
4. EXPERIMENTS AND RESULTS
5. DISCUSSION OF EXPERIMENTAL RESULTS
VI CONCLUSIONS AND FUTURE WORK…..…….64
1. SUMMARY OF THE PROPOSED WORK
2. SUMMARY OF OUR CONTRIBUTION
3. SUGGESTIONS FOR FUTURE WORK
REFERENCES……..67
BIOGRAPHY ……..……..69

Copyright by Mahidol University

LIST OF TABLES

Page

Table 3.1: Sample Relation emp……...20
Table 3.2: Encryption Relation empS of emp ……..20
Table 3.3: Sample Key Registry..…..……..…….20
Table 4.1 Database Table in Plain and Encrypted Forms …..…….…….....29
Table 4.2 Role Hierarchy Table…….……..…….....30
Table 4.3 Authority Table ….……..…….....30
Table 4.4 Key Table ….……....32
Table 4.5 Role Functional Table…….……....32
Table 5.1 All 31 Query Types…….……....55
Table 5.2 Query Group Based on Sensitive Attributes……62

Copyright by Mahidol University

LIST OF FIGURES

Page

Figure 2.1: Encryption and Decryption ……..6
Figure 2.2: Encryption and Decryption with Key…….....7
Figure 2.3: Encryption and Decryption with Different Keys ….……7
Figure 2.4: Access Control ….…….....10
Figure 2.5: Role-based Access Control ….……...12
Figure 2.6: Role-based Access Control Model ……13
Figure 3.1 : EISA Architecture ….…….....16
Figure 3.2: Sample Relation and Its Attributes ….……...17
Figure 3.3: Sample Encrypted Relation ……..…….....17
Figure 3.4: Associated B-Tree……...…..……..17
Figure 3.5 Architecture of the DAS Model…… ….…….....19
Figure 3.6: FUB System Architecture ….…….....22
Figure 3.7: Role Based Access Control in FUB...…….....22
Figure 3.8 : Users, Roles, and Applications…..…….....23
Figure 3.9: Functions and Official Positions…….....25
Figure 3.10: Roles, Applications and Access Rights ….…….....25
Figure 4.1 The Proposed Model ….……..…….....27
Figure 4.2 Organization Chart ….…….....29
Figure 4.3 Database Transformation ….……..…….....34
Figure 4.4 Encrypted Data in Encrypted Database……..…….....36
Figure 4.5 Authority Table retrieved for a Query ……...…….....36
Figure 4.6 Encrypted Key Used ……....…….…….....36
Figure 4.7 Query Process over Encrypted Database ...…….....37
Figure 4.8 User MF-3 and Sample Query 1…….…….....39
Figure 4.9 Access Rights of Sample Query 1 ..…..…..…….....39
Figure 4.10 Encrypted Query Results of Sample Query 1 …….…….....40
Figure 4.11 Plain Results of Sample Query 1 …….…….....40

Copyright by Mahidol University

LIST OF FIGURES (CONT.)

Page

Figure 4.12 User VP and Sample Query 2 ..…….....41
Figure 4.13 Access Rights in Sample Query 2 ..……..…….....41
Figure 4.14 Encrypted Query Results in Sample Query …….....42
Figure 4.15 Plain Results of Sample Query 2 …….…….....43
Figure 4.16 User and Query for Sample Query 3…….....43
Figure 4.17 Access Rights in Sample Query 3 ...…….....43
Figure 4.18 Encrypted Query Results in Sample Query 3…...…….45
Figure 4.19 User and Query for Sample Query 4…….....45
Figure 4.20 Access Rights in Sample Query 4….…….....46
Figure 4.21 Encrypted Query Results in Sample Query 4.... ..…….....47
Figure 4.22 User and Query for Sample Query .....…….....47
Figure 4.23 Access Rights in Sample Query 5…..…….....48
Figure 4.24 Query Results in Sample Query 5 …..……..……..... 49
Figure 4.25 Query Results in Sample Query 5 ..…….....49
Figure 4.26 User and Query for Sample Query 6…….....49
Figure 4.27 Access Rights in Sample Query 6…..…….....50
Figure 4.28 Query Results in Sample Query6…..…….....51
Figure 4.29 Query Results in Sample Query 6…..…….51
Figure 5.1 System Architecture ….……..……..…….....52
Figure 5.2 Results of Query by Owners…….. …...…….....56
Figure 5.3 Results of Query by Supervisors……. ...…….....57
Figure 5.4 Results of Insertion Query ...……..…….....58
Figure 5.5 Results of Update Query ...…….....59
Figure 5.6 Results of Query by Roles ...……..…….....60
Figure 5.7 Results of Query by Unauthorized Persons ...…….....61
Figure 5.8 Results of Query by Groups ...…….....63

Copyright by Mahidol University

CHAPTER I
INTRODUCTION

Even though the advances in computer technology have been rapidly evolved, the most important asset in any organization and the heart of all processes is data. Some data should be kept privately and called *sensitive* information, whereas others could be gone to the public. For valuable data such as marketing and financial information, they are very essential for the survival of a company. Generally, data could be stored in any form and place; however, they must be kept safely and confidentially. In addition, they should be protected from unauthorized accesses or modifications.

Practically, the organization has set some security policy that determines who has which rights to access which parts of the database. As a result, the database has been under a certain level of protection from unauthorized accesses. However, the database administrator or DBA of the database usually has the full legal access to the whole database. Hence, both sensitive and non-sensitive information could not be hidden from the DBA. If the DBA is intended to steal sensitive information, nobody can stop and prevent such an incident, those data in an organization still have a high risk from unauthorized access.

Unauthorized accesses from an insider are hard to protect since we have a sufficient level of trust to those people. There are several methodologies proposed to protect the information, and encryption is one of the widely used techniques. It can help protect sensitive data, and keep them confidential by transforming data into unreadable format. Without knowing the proper key and the applied encryption algorithm, the DBA or any authority cannot understand the hidden information in the database even though he can get an access to the data.

Regarding encryption mechanism, key management plays a significant role in keeping the data secret. Typically, key management would cover the tasks of key assignments, and key maintenance such as key updates when a new employee joins the company, or when an old employee leaves. In some organizations, every personnel

Copyright by Mahidol University

would possess individual key so that it can be used to encrypt his or her private information.

1.1 Problem Statement

Even though the encryption could provide a certain level of confidentiality, it definitely comes with costs such as time and resources. Thus, it should be used when the system needs high security. In addition, instead of encrypting the whole database, only sensitive data such as salary or credit card number should be encrypted. Since the encryption is not a security cure-all, other mechanisms would be needed to keep the data secure as much as possible. In this work, four related issues will be addressed as follows.

- **Process Model**

Since encryption and decryption need to be performed onto an encrypted database, one option is to add the encryption during query pre-processing and the decryption during query post-processing. Authentication and authorization are also required before those processes.

- **Encryption Algorithm**

Individual encryption algorithm is suitable for some specific information. Thus, selecting one algorithm to use must take many issues into consideration. One factor is the cost of processing which depends on the nature of data and the results you require.

- **Data Granularity**

Data in a database has the level of granularity such as databases, relations, tuples and attributes. The level selected to be encrypted will affect the security and the performance of the database itself.

- **Query Process**

Another important issue to be concerned is the accuracy of the query results over the encrypted database. A query process typically consists of query translation and query

Copyright by Mahidol University

post processing. All of these steps are initiated when a user inputs a plain query until the final plain result is displayed.

1.2 Objectives of the Thesis

1 To keep sensitive information in the database confidential in such a way that even a DBA cannot interpret their hidden form;

2 To execute user queries without changing the query format;

3 To measure the query performance when sensitive attributes are encrypted;

4 To design and implement a secure query model for encrypted database;

5 To apply the role-based access control model to the encrypted database.

1.3 Scope of the Thesis

1 Implement a sample database in an organization that consists of both sensitive and non-sensitive data, and only the sensitive data will be encrypted;

2 Simulate a sample organization containing multiple departments and multiple levels of authority, and assign access rights according to roles of individual personnel;

3 Design a query model that accepts user queries requesting both sensitive and non-sensitive information from the encrypted database;

4 Have the authorization process enforced by the role based access control of the sample organization;

5 A query from an authorized person will return the results in plaintext, but the same query from unauthorized person will have the encrypted form as the results.

6 The performance of numerous queries to the sample encrypted database is measured and compared with the performance of the same queries to the plain database and the whole encrypted database.

1.4 Organization of the Thesis

The remainder of the thesis is organized as follows. We give background and related work in Chapter 2 and 3, respectively. Chapter 4 presents our proposed work. Chapter 5 describes the details of our implementation and experiments. We also

Copyright by Mahidol University

discuss the experimental results. Finally, Chapter 6 summarizes our contributions and suggests some future work.

Copyright by Mahidol University

CHAPTER II
BACKGROUND

This chapter gives some fundamental background used in our proposed work. They are described in turn as follows. First, the brief overview of a query processor is described. Second, the basic concepts of encryption are explained. Third, we discuss the issues of granularity in a database since they typically affect the query performance. Next, a brief overview of certificate authority or CA is presented. Finally, the general ideas of access control models, and the role based access control or well-known RBAC that will be applied to our work, are described.

2.1 Query Processor

A query processor is responsible for parsing and executing queries received from users. It consists of two components: query compiler and execution engine. The query compiler [3] translates a query into a query plan which is a sequence of actions derived from a query, and has three functions as follows:

- *Query parser* creates a parse tree structure from the received query;
- *Query preprocessor* examines the semantics of the query, and transforms the parse tree into the corresponding algebraic operation tree which is the initial query plan;
- *Query optimizer* transforms the initial query plan into a sequence of operations optimally.

The resulting optimal query plan is then sent to the execution engine which will execute each operation in the query plan, and the outcome is returned to the user. The execution engine must interact directly with other components of the DBMS.

Copyright by Mahidol University

2.2 Encryption

Encryption is the process of transforming a plaintext message into an encrypted form of the message that is not readable or understandable. The encrypted message is called *ciphertext*. Intuitively, encryption is the process to disguise the message in such a way as to hide its substance. The process of transforming a ciphertext into its corresponding plaintext is called *decryption*. The overview of these processes are given in Figure 2.1 below. Data to be encrypted could be a stream of bits, a text message, a stream of digitized voice, and even a digital video image. Encrypted data may have the same size as its corresponding plaintext. But, it may be larger or smaller, especially when combined with compression [18].

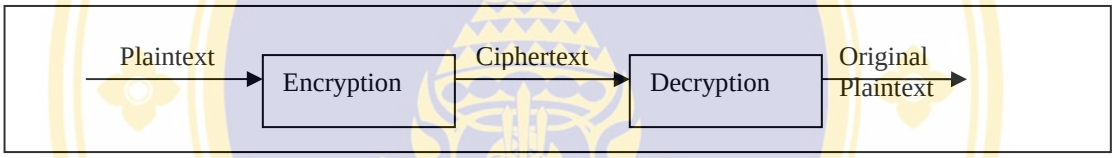

Figure 2.1: Encryption and Decryption

Mathematically, encryption and decryption functions can be expressed as follows.

Encryption: $E(P) = C;$

Decryption: $D(C) = P$

where P and C denotes plaintext and ciphertext, respectively; and E and D denotes an encryption and decryption functions, respectively. These two functions must be reversible. In other words, when decrypting an encrypted message, the original plaintext must be recovered. As a result, the following identity relationship, $D(E(P)) = P$, must be hold.

Some classical encryption algorithms may not use a key such as Caesar cipher [18], but most modern cryptographic methods use it. The notations for encryption and decryption functions with a key now become:

Encryption: $E_K(P) = C;$

Decryption: $D_K(C) = P$

Copyright by Mahidol University

Thus, the identity relationship is expressed as $D_K(E_K(P)) = P$, where K denotes a key. The size of a key could be small or large, and varied from 64 to 1024 bits. Figure 2.2 shows the encryption and decryption processes with a key.

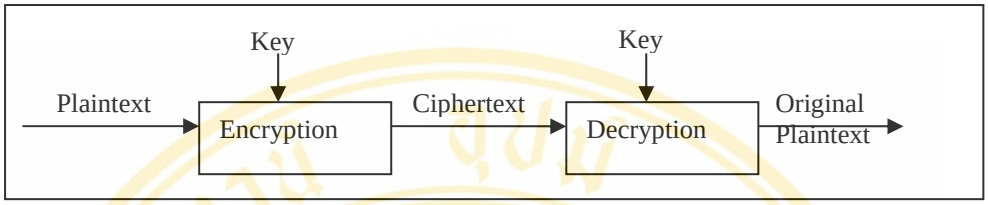

Figure 2.2: Encryption and Decryption with a Key

Some algorithms use different keys for encryption and decryption. For example, K_1 is an encryption key, and K_2 is its corresponding decryption key. Thus, the processes as shown in Figure 2.3 are modified as follows:

Encryption: $E_{K1}(P) = C;$

Decryption: $D_{K2}(C) = P$

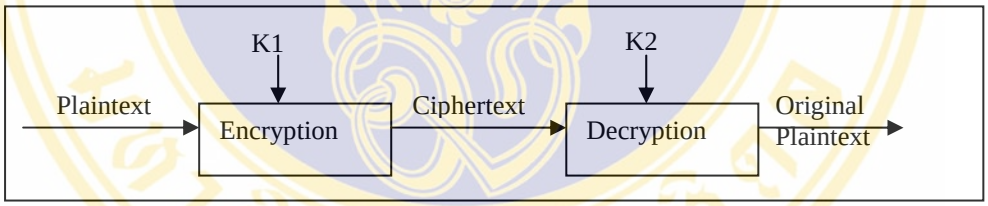

Figure 2.3: Encryption and Decryption with Different Keys

There are two main key-based cryptographic algorithms: secret key cryptography and public-key cryptography which are described in turn as follows.

- Secret Key Cryptography

In secret-key algorithms, the encryption key is the same as the decryption key. Thus, it is mandatory that the sender and the receiver must agree on a secret key to be used before they can securely communicate. They also have two operation categories: stream cipher and block cipher. A stream cipher performs an encryption of the plaintext by one bit at a time, whereas a block cipher groups a number of bits into a block and performs an encryption of the plaintext by one block at a time. The secret key size is usually ranged from 40 to 256 bits. The most widely used secret key algorithms are DES, triple-DES, and AES.

Copyright by Mahidol University

- Public Key Cryptography

Public key algorithms are designed in such a way that the key used for encryption is different from the key used for decryption. Typically, one key is called a private key and the other is called a public key. If we want to send a secret message to one person, we must use that person's public key to encrypt a message so that only the owner of the public key can use his private key to decrypt the encrypted message. Furthermore, if one public key is known, its corresponding private key cannot be directly inferred since both keys have a pre-defined mathematical relationship. The key lengths are normally between 512 to 2048 bits. The most popular public key algorithm used today is RSA.

2.3 Level of Granularity

Since the cost of performing an encryption is quite high, only crucial data should be encrypted. In addition, data in the database has multiple levels of granularity such as individual attributes or the whole record. Encrypting the whole record would not make sense. Thus, only sensitive data would need to be encrypted while non-sensitive data will be left intact. Motivations and effects of having data in a database encrypted at different levels of granularity are briefly described as follows.

- *Individual attribute*: An attribute is the smallest unit in a database. Encryption at this level has the highest flexibility because a specific row or column can be selected to encrypt.
- *Row*: Encryption is performed over individual and sensitive records such as the personal information of the company executives since the secret and privacy of individual records can be ensured.
- *Column*: Encryption is performed over sensitive attributes of a database table. Such sensitive information includes monthly salary or credit card numbers that would need high protection, and cannot be publicly disclosed.
- *Table*: In some cases, the whole table of the database needs to be encrypted. This encryption is easy to do, but usually we do not recommend it except the whole table is confidential.

Copyright by Mahidol University

2.4 Certificate Authority

A certificate authority (CA) is a trusted organization that provides a set of secure mechanisms for digital community. The main functions are (1) to issue a valid certificate to a requester, and (2) to verify the authenticity of a certificate. Thus, the CA is the third party who has been trusted by two parties that want to have secure communicate. Other issues related to CA are further given in detail here.

2.4.1 Body of Certificate

A certificate consists of three essential parts explained below.

a) *User Information*: This part contains the personal information of the certificate owner such as name, family name, and e-mail address;

b) User's Public Key: This part contains the public key that will be used with the corresponding private key of the certificate owner;

c) *CA's Digital Signature*: This part contains the digital signature of the CA who issues the certificate so that the originality of the certificate can be verified. In other words, it is the identification between the certificate owner and his public key.

2.4.2 Certification Authority Trust Model

The relationship between a service provider (CA) and a requester (a user) who asks for a service can be classified according to the following categories.

(a) *Relationship between users* is defined between users and affects only related users. It does not have any effect to other service providers. It has two types.

- *Direct Trust*: One person can authenticate another person who wants to contact with. Usually, they are friends and know or contact each other before.
- *Trusted Chain*: This relationship is defined as a chain of trusts. It is suitable for a working group and has a small size.

(b) Relationship between CA defines a relation among service providers and has two types.

- *Cross Certification:* Certifying relationships across service providers.

Copyright by Mahidol University

- *Hierarchy Trust*: The relationship is defined in a hierarchy structure like a tree. The CA at the top level is called “Root CA”.

2.5 Access Control

Access control is a security policy used to define the permissions or denials of accessing an object or data in a computer and communication system. It is used to define the rights of individuals or application programs to obtain data from computer resources. Another definition of access control is the process of limiting access to resources for authorized users, programs, processes, or other entities. It is typically performed by a resource controller that is responsible for allocating system resources in order to satisfy user requests.

Access rights of a user include the ability to store the data in, retrieve the data from, communicate with other components, and make use of any resource in a system. Access control mechanism is to allow an authorized user to access associated system resources of which he/she has permission to. On the other hand, an unauthorized user will be forbidden from accessing any associated system resource [14].

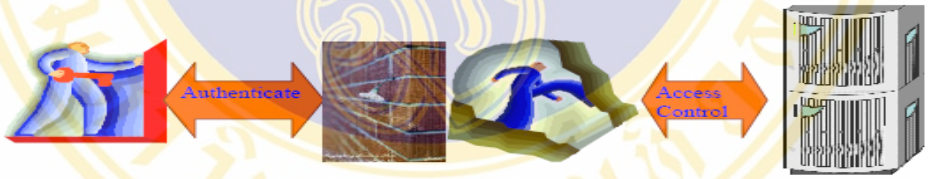

Figure 2.4: Access Control

Access control can be classified into two categories: discretionary access control (DAC) and mandatory access control (MAC). Each category is explained in turn below.

2.5.1 Discretionary access control (DAC)

DAC treats every user and permission independently. Permissions will be assigned to users according to their user id in a system. The system uses the access control list (ACL) table to define every user, user id and the corresponding permissions.

When a user asks for a permission to perform an operation on a specific resource of the system, the access control mechanism will examine the ACL table and return

Copyright by Mahidol University

the right permission before executing the operation. If the user has no permission to operate, the system will deny the request of the user. When a new user is added into the system, the security manager must examine what permissions are necessary and decide which specific permissions should be assigned to the new user.

DAC is the easy and flexible way in assigning permissions to a user, and it has been widely used in most computer systems nowadays.

2.5.2 Mandatory access control (MAC)

MAC was mainly used in the military system. It ensured that the security policy enforcement does not depend on users, but rely on the clearance of the users instead. MAC requires all those who create, access, and maintain information to follow rules set by administrators. The mechanism associates information with such labels as *TOP SECRET*, *SECRET*, and *CONFIDENTIAL* which is also called security classifications. When a user requests to perform an operation in the system, MAC will examine the label of the requested resource and the user. If the label of the user has a higher level than the label of the resource, then the user can read the resource. If the user has a lower level, the user can only write a new content to the resource, but cannot read it. These two rules are summarized as:

- *no-read-up property*: No process may read data at a higher level;
- *no-write-down or *-property*: No process may write data to a lower level.

The disadvantage of MAC is that it is very strict and not flexible enough for commercial use. All resources and users are formed as a partial order of nodes in the system. A user or a resource has been strictly labeled as lower as or higher than another user or another resource. For commercial uses, this policy is not applicable for real world applications.

2.6 Role-Based Access Control (RBAC)

In addition to DAC and MAC, an access control policy can be categorized according to time, task and role of users in a system. In this thesis, we will describe only role-based access control since it is related to our work.

Copyright by Mahidol University

Role-based access control or RBAC is another form of MAC which has evolved to meet the needs of commercial information systems. Rather than labeling information, it associates roles with each individual who might have a need to access information [6].On the other hand, if the label of user is the same as the label of the resource, RBAC allows the resource to be accessed because the operation is defined by a role; this is not possible in MAC. In RBAC, permissions are assigned to roles rather than to individual users. Users can acquire these permissions after they have been assigned to be a member of appropriate roles. This simple idea greatly eases the administration of authorizations. RBAC also provides some inherent flexibilities and a specific line of authority. Figure 2.5 shows many-to-many relationships of users, roles and permissions in RBAC.

Figure 2.5: Role-based Access Control

The major benefits of RBAC are the ability to express and enforce specific security policies and to simplify the process of security management. RBAC is a framework of policy rich mechanisms. Its configuration depends on the security policies of an organization which can be easily evolved according to the organization structure and security policy changes.

RBAC can statically and dynamically control user actions by establishing and defining roles, role hierarchies, role relationships and role constraints. After RBAC environment has been established, the main administrative tasks only deal with adding and deleting users to and from roles.

2.6.1 RBAC Model

The goal of creating RBAC model is to provide a concise and usable notation so that the security relevant properties of the model are not obscured by excessive

Copyright by Mahidol University

notational details. RBAC model supports a variety of security policies. In particular, one argument is made for least privilege and separation of duty.

Least privilege is the administrative practice of selectively assigning privileges to users such that the user is given no more privileges than necessary to perform their jobs. Separation of duty refers to the partitioning of tasks and privileges among different roles that are associated with a single user such that colluding among users can be prevented. The separation of duty concept includes multiplexing of shared resources, assigning distinctive sets of permissions for functional decomposition, categorical classification, and hierarchical decomposition. Within RBAC, the separation of duty concepts is supported by the establishment of the principle of least privilege [6].

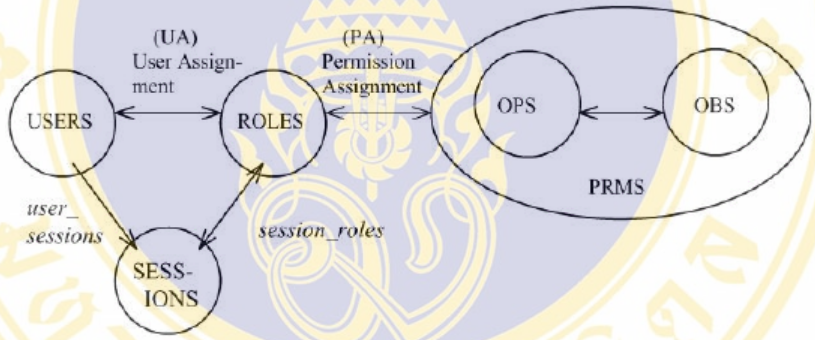

Figure 2.6: Role-based Access Control Model

Sample RBAC relationship is shown in Figure 2.6. In the example, a role is used to associate with users and permissions. Permission is a pair of objects and operations. If a user in this model is a human being, a role is a job function or a job title within an organization and has been associated with the authority and responsibility. A session is a mapping of one user to his/her possible roles.

The RBAC security model consists of two components namely MC_0 and MC_1. MC_0 is the RBAC authorization database model which defines RBAC security properties for authorization in static roles, whereas MC_1 is the RBAC activation model which defines RBAC security properties for dynamic activation of roles. In addition, RBAC model supports the following specifications:

- *User/role associations*: the constraints specifying user authorizations to perform roles.

Copyright by Mahidol University

- *Role hierarchies*: the constraints specifying which role may inherit all or part of permissions of another role.
- *Duty separation* constraints: they are role/role associations indicating conflict of interest.
- *Static separated duty*: the constraints specifying that a user cannot be authorized for two different roles.
- *Dynamic separated duty*: the constraints specifying that a user can be authorized for two different roles but cannot act both roles simultaneously.
- *Cardinality*: the maximum number of users allowed for a particular role. For example, there will be only one manager per department in a company.

Copyright by Mahidol University

CHAPTER III

RELATED WORK

This chapter presents related work that addressed numerous encryption issues in databases and also proposed the concept of enforcing security in the database. They include the data organization and the query model for encrypted database, and RBAC security policy applied to the database. We will present the following related work in turn below.

The first paper proposes a model for querying over encrypted database, but only row encryption is considered whereas sensitive attributes are encrypted in our model. The second paper addresses the issues about key generation, key management and key updates for encrypted database since we also need to handle those issues in our proposed model. The last paper illustrates the application of role-based access control in an organization which focuses on the hierarchy of official positions and functions.

3.1 EISA: An Enterprise Application Security Solution for Databases [16]

Three primary challenges in encrypting databases are (1) locating which part of the database should be encrypted, (2) performing the efficient query execution over the encrypted database, and (3) maintaining data integrity with minimal overhead. This work focuses on data integrity and having efficient query execution over encrypted database. In other words, all queries are performed at the application server and only related necessary records are retrieved from the server. Hence, there is a secure wrapper presented between encrypted database server and applications in order to ensure both data privacy and integrity.

The proposed architecture shown in Figure 3.1 consists of three fundamental components: Encrypted Database Server, Security Wrapper or EISA: Enterprise

Copyright by Mahidol University

Information Security Adaptor, and Client Application. The server has only encrypted tuples. A query submitted at the client application will connect to the EISA instead of directly connecting to the server. EISA is responsible for maintaining the metadata, indexing and keys used for encrypting and decrypting database tuples.

Each query submitted is processed as follows.

1.1 The client application sends a query to the EISA wrapper.

1.2 The query will be optimized and sent to the query translator which is part of the query engine.

1.3 The query translator evaluates the query at the wrapper using indexes stored, and disintegrates the original query into the server query and the wrapper query.

1.4 The server query is pushed onto the encrypted database server. The results are retrieved and then sent to the integrity checker of EISA which examines the integrity of each record retrieved.

1.5 The integrity checker calculates a hash value of each record and compares it with already stored hash. If they do not match, a tampered message alert will be returned. Otherwise, the results are decrypted and stored temporarily at the wrapper before returning them to the client application.

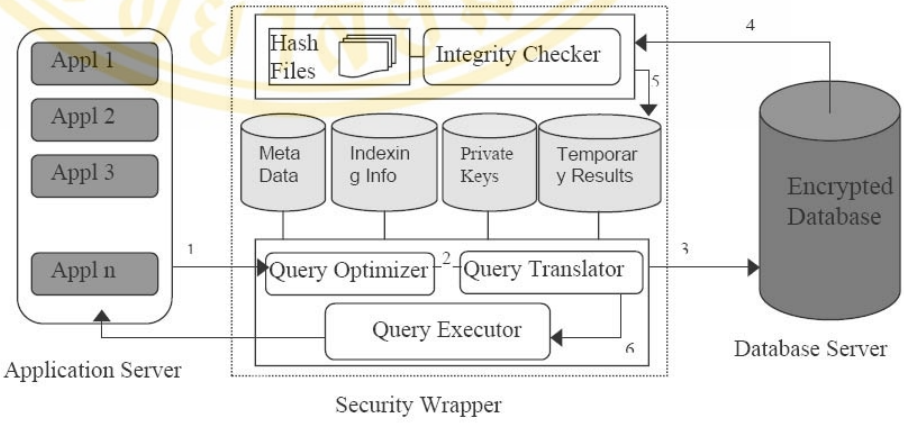

Figure 3.1 : EISA Architecture

Another essential issue mentioned in this paper is the data organization. At the server side, each relation R (A1, A2,…, An) is stored at the encrypted server in the form of {id, etuple} where ‘id’ represents an incremental variable maintained by the

Copyright by Mahidol University

wrapper, and ‘etuple’ is the encryption of all the attributes of the tuple. An example of a database relation and its attributes are shown in Figure 3.2. Figure 3.3 shows the encrypted version of the relation stored at the server.

ename	ssn	bdate	addr	dno
Smith	123456789	1982-01-09	Houston	5
Wong	333445555	1979-12-08	Spring	4
Zelaya	999887777	1981-07-19	Houston	1
English	453453453	1980-07-31	Humble	4
Joyce	334555677	1976-08-09	Spring	6

Figure 3.2: Sample Relation and Its Attributes

id	etuple
1	1000101010100000010..
2	1110111000111000001..
3	0100011111110101010..
4	0000010101011101010..
5	1111111000101010101..

Figure 3.3: Sample Encrypted Relation

Meanwhile, the wrapper maintains the indexing information in the form of associated B-trees as shown in Figure 3.4 which associates the attribute ‘dno’ of the sample relation in Figure 3.2 with the record id of encrypted tuples.

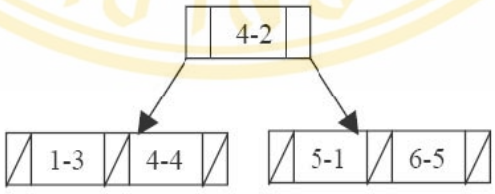

Figure 3.4: Associated B-Tree

In other words, for each index attribute, the wrapper will create an associated B-tree over the plaintext values and store the attribute value as a search key and the record id of the encrypted tuple as the associated information. Similarly, the wrapper creates another associated B-tree for each table that contains the id and the hash of each record being inserted into the database. These B-trees are used for integrity checking of records when retrieving them from the encrypted database.

Copyright by Mahidol University

As previously mentioned, a user query is disintegrated into a server query and a wrapper query. The server query returns the encrypted tuples to the wrapper which will decrypt the tuples and store them temporarily for the execution of the wrapper query. For selection operation, all mapping conditions are done at the wrapper itself, and the results are a set of id's that are part of the server query. Other operations like projection and aggregation are evaluated at the wrapper.

The work has been evaluated using TPC-H benchmark [x] and running on Microsoft SQL server [x]. The results show substantial costs incurred by data integrity checker.

3.2 Efficient Key Updates in Encrypted Database Systems [9]

This paper presents an efficient implementation of the key updates and key management in the database as a service (DAS) model using specialized techniques and data structure in order to improve the concurrency performance in the DAS model.

In the DAS model, the client's database is stored at the service provider. The provider is responsible for provisioning processing power, storage and network resources required to do database operations. One significant challenge in the DAS model is the database security since clients view their information on the database as an asset. Thus, they need to be assured that the data will be well protected against malicious attacks and unauthorized accesses from outsiders. In addition, recent studies indicate that 40% of those attacks are originated by insiders. Another challenge is the privacy of the data since the owner cannot trust even the service provider itself. Hence, data encryption is proposed as a solution to ensure the privacy of users' data.

The architecture of the DAS model and its control flow are shown in Figure 3.5 below, and is comprised of three basic entities: *user*, *server* and *client*.

A user poses a query to the client who stores the data at the server and the server is hosted by the service provider that stores the encrypted database. The query is processed at the server without jeopardizing the user's data privacy since all data are stored in the encrypted form.

Copyright by Mahidol University

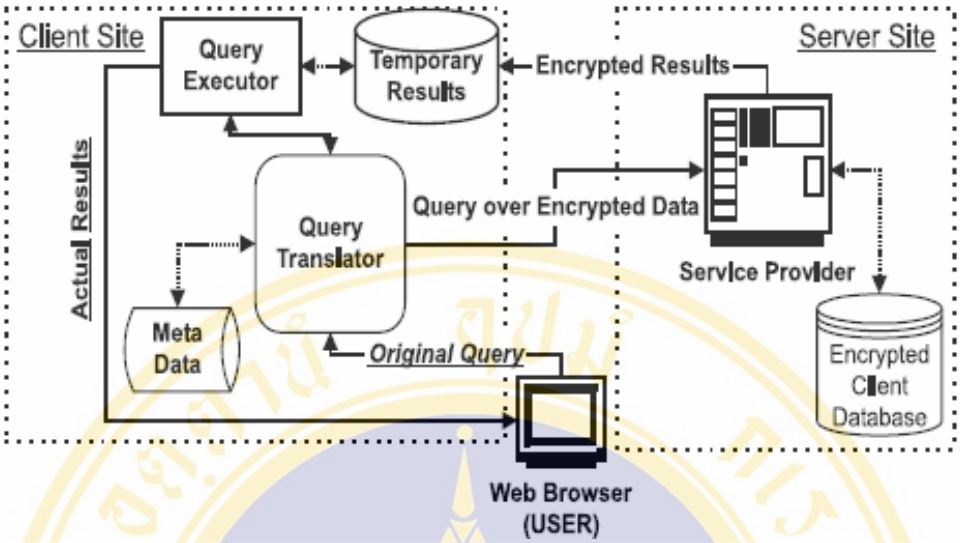

Figure 3.5 Architecture of DAS Model

Specifically, each relation, R, with attributes, A_1, A_2,…,A_n, is stored on the server as an encrypted relation, R^S , having attributes (RID, KID, *etuple*, P_1^{id}, P_2^{id},…, P_i^{id}). The *etuple* is the encrypted string corresponding to a tuple in the relation R, and each attribute, P_i^{id} , keeps the partition index for the attribute A_i used for query processing at the server. In addition, RID represents the unique record identifier created by the client for each tuple while KID represents the key identifier also created and assigned by the client. KID is used to identify which key is applied to encrypt the *etuple*. However, the RIDs are not the same unique identifiers used as references to the database records assigned by the database manager. The example in Table 3.1 and 3.2 below shows the relation *emp* and its corresponding encrypted form, emp^S, as emp^S (RID, KID, etuple, eid^{id}, $ename^{id}$, $esalary^{id}$), respectively.

Another issue addresses in this paper is key management which is a group of policies and procedures that regulate the maintenance of encryption keys within the system. The key-related operations include: key generation, key installation, key distribution, and key update. In addition, each key can be used to encrypt different database objects such as a table, a row or an attribute. Thus, the selection of key granularity would depend on the nature of the database security required, and there are three categories of key granularity: database-level (one key per database), table-level (one key per group of tables), and row-level (one key per group of rows).

Copyright by Mahidol University

eid	ename	salary	addr	did
23	Tom	70K	Maple	40
860	Mary	60K	Main	80
320	John	23K	River	35
200	Sarah	55K	River	10

Table 3.1: Sample Relation *emp*

RID	KID	*etuple*	eid^{id}	$ename^{id}$	$salary^{id}$
1	45	=*?Ew@R*((¡¡=+,-...	2	19	81
2	78	b*((¡¡(*?Ew@=l,r...	4	31	59
3	65	w@=W*((¡¡(*?E:,j...	7	59	22
4	52	ffTi* @=U(¡?G+,a...	8	49	59

Table 3.2: Encryption Relation emp^S of *emp*

The paper also presents two key generation schemes: pre-computation scheme where encryption keys are pre-computed and stored in a ready-to-use format, and re-computation scheme where encryption keys are re-computed whenever they are needed. All generated keys and their related information are kept in the *key registry* and must be accessible by authorized users only. The key registry consists of five attributes shown in Table 3.3 below. The attribute *Correspondence* indicates the table name and the range of record ids of the table that the key will be applied, and the attribute *Cardinality* tells the number of records that have been encrypted with the key.

KID	Correspondence	Expiration	Cardinality	Key
45	mgr.RID:[1,200]	06/12/05	120	*the key*
92	mgr.RID:[1,200]	06/12/05	30	*the key*
52	proj.RID:[1,500]	06/09/05	275	*the key*
77	emp.RID:[1,300]	07/01/05	130	*the key*
23	mgr.RID:[201,500]	06/08/05	40	*the key*

Table 3.3: Sample Key Registry

In addition, the key registry may be encrypted with a master key kept securely at the server. The key can be downloaded from the server and decrypted when the client requests it. If a third party is chosen for key distribution, the public key infrastructure or PKI may be used for both authentication and authorization of users. Furthermore, from the security perspective, the lifetime of an encryption key should be limited, and

Copyright by Mahidol University

rekeying is recommended, and sometimes required in order to prevent a key compromise. In practice, periodic rekeying scheme has been mostly used (compromise availability).

3.3 Role-Based Access Control (RBAC) System of a European Bank [17]

This paper describes the access control system of a major European bank called Dresdner Bank having over 50,000 employees and more than 1,400 branches worldwide. The bank uses a variety of different computing applications to support its business, and has different levels and types of users. For security reasons, the bank has successfully deployed access control systems based on the concept of RBAC and can provide access control services at the enterprise-wide level to a variety of applications. The access control system for those applications is called FUB (Funktionals Berechtigung).

In the system, access rights are given to the individual users according to the combinations of their job functions and the official position within the organization. An application cannot make access control decisions on its own. It grants an access to the data based on a centrally provided security profile. An application is launched by a user who has been identified and authenticated by the system. The system defines roles for each user, and delivers access rights for usage within other applications that run under various environments.

Role information and related organizational units are used as the access control criteria for applications. These data are defined and maintained in the human resources database as shown in Figure 3.6. When a user starts an application, the FUB will deliver the corresponding security profile giving the application which individual access rights the user possesses. Then, the access rights are assigned to the application accordingly.

Several applications are accessed through a role, and each application has a set of access rights assigned to it. A simplified underlying data model is shown in Figure 3.7 below. Specifically, a role in the FUB system is defined using the official position within the organizational hierarchy and the description of the job function. These data

Copyright by Mahidol University

are delivered by the human resources system. Theoretically, the total number of roles would be the product of official positions and functions. The bank has 65 official positions combined together with 368 different job functions. Thus, there should have 23,920 roles in the bank, but only 1,300 roles are currently in use for about 40,000 users in the FUB. The role/user ratio would be approximately only 3.2% due to the hierarchical structure of official positions in the bank.

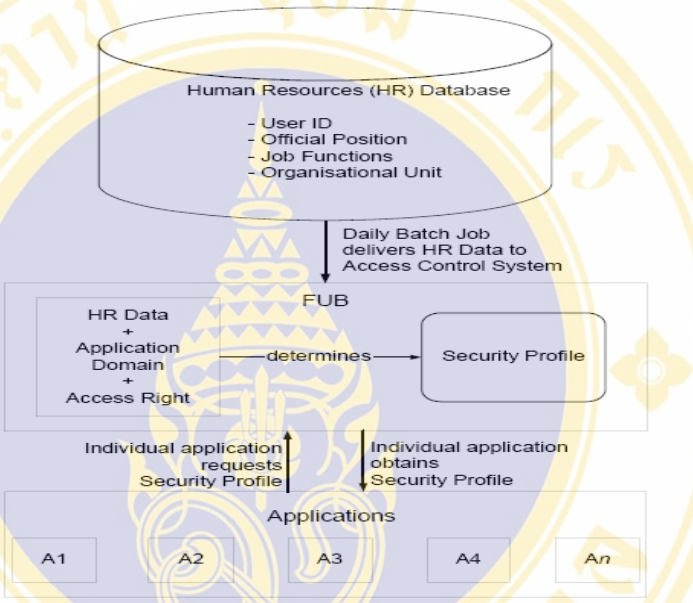

Figure 3.6: FUB System Architecture

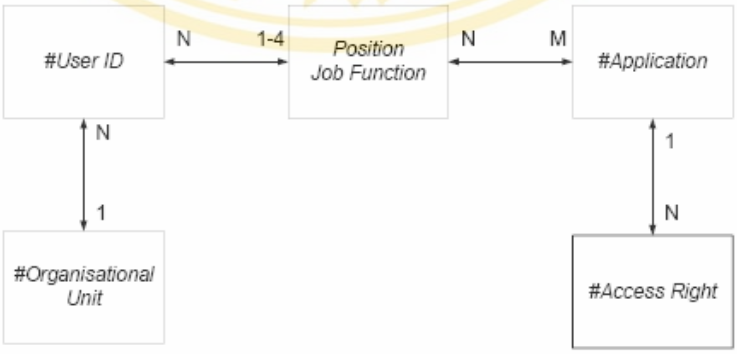

Figure 3.7: Role Based Access Control in FUB

From the administrative perspective, the system also provides access control services to non-permanent staffs such as third party consultants, temporary staffs and

Copyright by Mahidol University

freelancers. Hence, these users should always get new accounts when starting to work and those accounts should be deleted when they leave the project.

In addition, assignments of roles to users and of access rights to roles occur at three different levels of authority within the organization as shown in Figure 3.8 and described as follows.

(1) Human Resources Department

Users and roles are initially created in the human resources department. A user is given a unique identification number, and associated with functions and official positions before assigning a role.

(2) Application Administration

For each application identified by a unique id, access rights represented as numbers are assigned to a role through the application administrator.

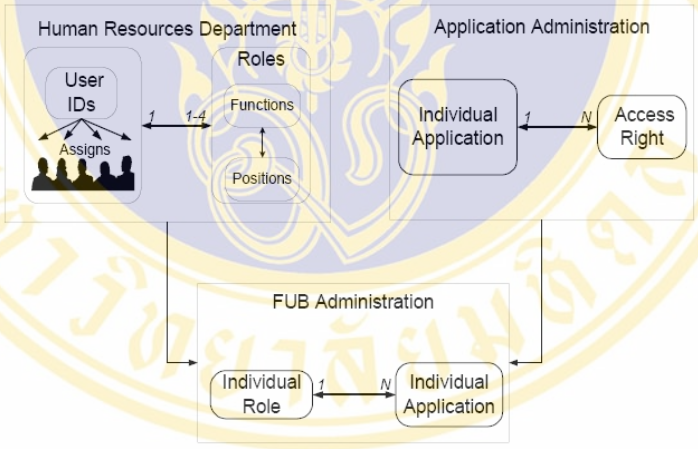

Figure 3.8 : Users, Roles, and Applications

(3) The FUB administrator

The FUB administrator, who knows about the identity of a user, can only do role to application assignments. A user identifier is given to the user by the administrators and applications are assigned as needed when a consultant works on a project. Also, any user/role, role/application assignment becomes invalid when the organizational structure changes.

Further improvements of the access control system are suggested as follows.

Copyright by Mahidol University

- Access Right Allocation: Assign certain access rights directly to a user. For example, an attorney who represents a user should have the same access rights as the user.
- Grouping Mechanism: (1) Only a group of employees needs to be assigned with a role, not individual employee; (2) Two access rights that can be grouped together such as creating and deleting an account will provide an easy management of access rights to applications.
- Covering/Standing-in regulations: In practice, one employee may have to cover duty for another employee in case of emergency or holidays. In another situation, one person may have to stand in for two employees at the same time. However, these may violate the separation of duty policy.
- Competences and Constrains: The specification of competences and constraints in the security profile should be allowed. For example, the authorization to sign a contract up to a certain amount of money.
- Mapping Access Control System to Organizational Structure: An organization is subject to continuously changes either due to the market needs, unforeseen acquisitions or mergers. Also, the system must be able to have flexible support of business strategies as needed. The RBAC Model in this system considers two issues: access right inheritance through a role hierarchy, and groups of users. A role in FUB consists of both function and position as shown in Figure 3.9. The partial ordering of the FUB roles can be defined either or both of function and position as described as follows.
- Hierarchy of official positions: It is a strict partial order in the organization of official positions (denoted by the > symbol). For example,

Head of Division > Group Manager > Clerk

A role hierarchy could be defined in which one role is superior to another if its position is superior and their functions are identical. Figure 3.10 show that both roles, A and B, have many access rights in each application.

Copyright by Mahidol University

Role	Function	Official Position
A	financial analyst	Clerk
B	financial analyst	Group Manager
C	financial analyst	Head of Division
D	financial analyst	Junior
E	financial analyst	Senior
F	financial analyst	Specialist
G	financial analyst	Assistant
…	…	…
X	share technician	Clerk
Y	support e-commerce	Junior
Z	office banking	Head of Division

Figure 3.9: Functions and Official Positions

Role	Application	Access Right
A	Money Market Instruments	1,2,3,4
	Derivatives Trading	1,2,3,7,10,12
	Interest Instruments	1,4,8,12,14,16
B	Money Market Instruments	1,2,3,4,7
	Derivatives Trading	1,2,3,7,10,12,14
	Interest Instruments	1,4,8,12,14,16
	Private Customer Instruments	1,2,4,7
…	…	…

Figure 3.10: Roles, Applications and Access Rights

- Hierarchy of Functions: A partial ordering of functions could exist. For example, the ordering of the two functions: inspector and finance accountant could form an ‘isa’ hierarchy which would be:

  inspector > finance accountant

  Alternatively, a role hierarchy would be defined as having one role superior to another if its function is superior, regardless of position. It can be formally defined as:

  $$Role(x) > Role\ (y) \Leftrightarrow Role(x).Function > Role\ (y).Function$$

  Finally, grouping mechanism will provide ease of administration as only the group is assigned to a role and not each individual. However, currently the model does not allow for the assignment of groups to roles. It explicitly states that the assignment of users to roles is a relationship between users and roles.

Copyright by Mahidol University

CHAPTER IV
PROPOSED WORK

In this chapter, we present our proposed model for querying over encrypted database. The model modifies a general query model by adding the key management, query translator and query post-processor for encrypting input query and decrypting query outputs, respectively. This work focuses on encrypting of sensitive data only. In addition, role-based access control or RBAC has been related with how keys are assigned to roles and positions of individual members of the organization.

4.1 Overview of the Proposed Model

The proposed model aims to provide secure query processing over the encrypted database system. When a user submits a plain query to the encrypted database, both user authentication and query authorization will be examined against the authority table for the appropriate access rights assigned to the query. The plain query will then be sent to the query translator for encrypting the query which is forwarded to the trusted query manager. Based on the access rights received from the authorization module, the trusted query manager will either process the query by retrieving data from the encrypted database and return the encrypted results to the query post-processor, or reject the query and return nothing to the user. The query post-processor is responsible for translating encrypted results to the corresponding plain results. The proposed model would be able to apply to real world situations in an organization.

Figure 4.1 below shows the model of querying data from encrypted database. Steps of processing each query submitted are given as follows.

(1) A user logs into the system by submitting his username and password which will be authenticated by the authentication module.

(2) The authentication module verifies the user via username and password, and returns an acknowledgement to the user.

(3) If the user is authenticated, the authentication module will send the user information to the authorization module.

Copyright by Mahidol University

(4) The authorization module will retrieve the user's access rights from the authority table based on the user's information.

(5) Given access rights of the user is returned to the authorization module.

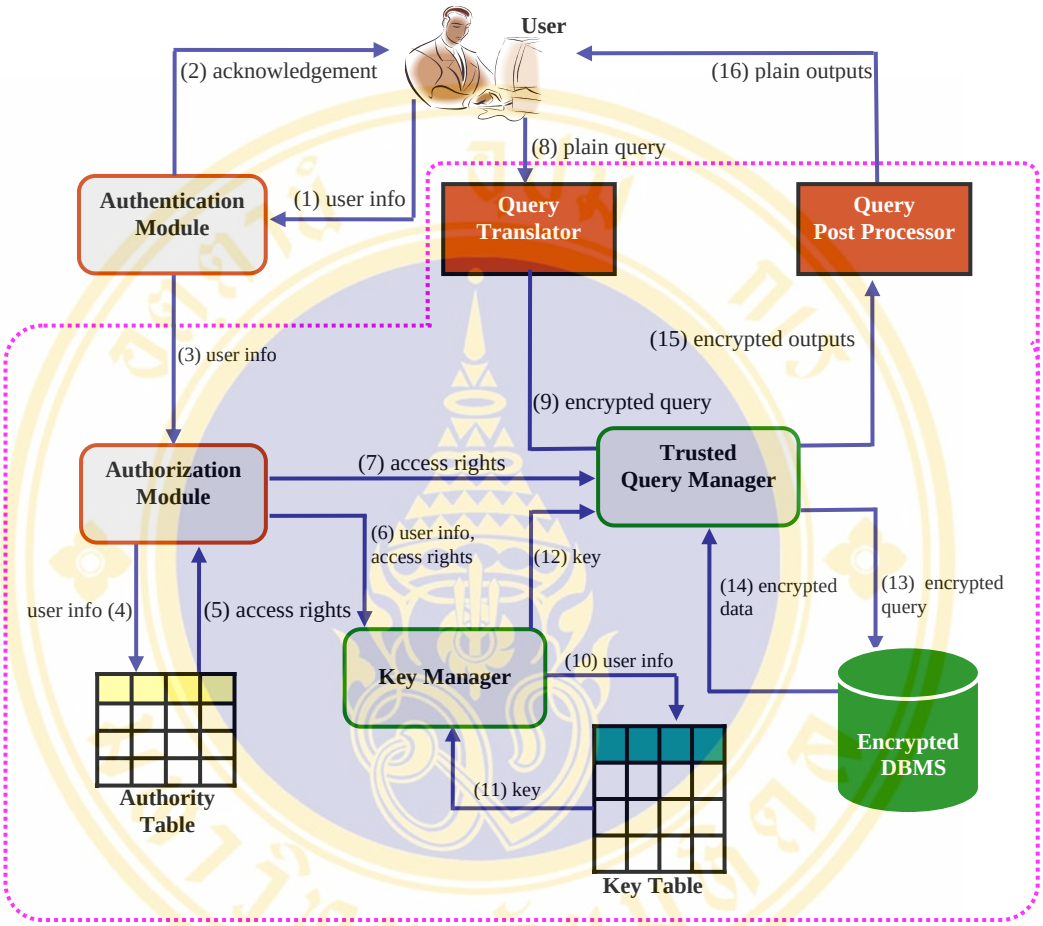

Figure 4.1 The Proposed Model

(6) The authorization module sends the user information and the user's access rights to the key manager for retrieving the corresponding encryption keys.

(7) At the same time, the authorization module also sends the user's access rights to the trusted query manager.

(8) After the user is authenticated, the user submits a query to the query translator.

(9) The query translator translates the user's query into an encrypted query form.

(10) If the user has no encryption key, the key manager will generate the key and keep the key in the key table.

Copyright by Mahidol University

(11) If the user has the encryption key, the key manager will retrieve the user's key from the key table.

(12) The key manager sends the user's key to the trusted query manager for decrypting the submitted query.

(13) The trusted query manager processes and retrieves the encrypted data from the encrypted database.

(14) The encrypted DBMS returns the outcomes in an encrypted form to the trusted query manager.

(15) The trusted query manager sends the encrypted outcomes to the query post processor.

Finally, the query post processor transforms the encrypted outcomes to the corresponding plain results which will be given to the user.

4.2 Proposed Query Model

The proposed query model consists of three components: table structures, the query processing and the trusted query manager. They are explained in details below.

4.2.1 Table Structures

There are three essential tables in our proposed model. First, it is the database table used as a sample database throughout the work. Second, it is the authority table containing the authority of all positions in an organization. Finally, the key table keeps all keys used to encrypt and decrypt objects in the database. We describe each table in detail as follows.

- **Database Table**

A sample database table used in our proposed model is shown in Table 4.1, and defined to contain both sensitive and non-sensitive attributes. There are two sensitive attributes: *emp_sal* and *emp_diag* representing the salary and illness of individual employee. Both attributes are displayed in plain and encrypted forms, respectively, whereas other personal information attributes such as name, address, age and position are displayed in a plain form. Hence, each tuple contains both plain and encrypted

Copyright by Mahidol University

attributes. Because our work focuses on column granularity, only interesting attributes will be encrypted.

emp_id	emp_dept	emp_name	emp_add	emp_sal	emp_diag	emp_age
e010	Engineer	Jerry	Ocean	12000	tuberculosis	39
e021	FI	Tom	Maple	15000	cancer	28
e030	IT	Mary	Main	22000	cancer	35
e041	HR	John	River	15000	heart disease	55
e053	Manufacture	Hunt	Main	20000	allergy	26

Table 4.1 Database Table in Plain and Encrypted Forms

emp_id	emp_dept	emp_name	emp_add	emp_sal	emp_diag	emp_age
e010	Engineer	Jerry	Ocean	00001100…	10001110…	39
e021	FI	Tom	Maple	11110001…	00001110…	28
e030	IT	Mary	Main	11100011…	11100000…	35
e041	HR	John	River	11110000…	11100011…	55
e053	Manufacture	Hunt	Main	00111010…	11010100…	26

- **Authority Table**

The authority table contains the role hierarchy defined as relationships among positions within an organization. The role hierarchy table is shown in Table 4.2. It is used to define the level of positions that have an authority to access the data. A sample organization used in our proposed model can be represented as an organization chart shown in Figure 4.2. When a user wants to be authenticated and authorized for accessing the database, the processes must be performed thru the authority table given in Table 4.3.

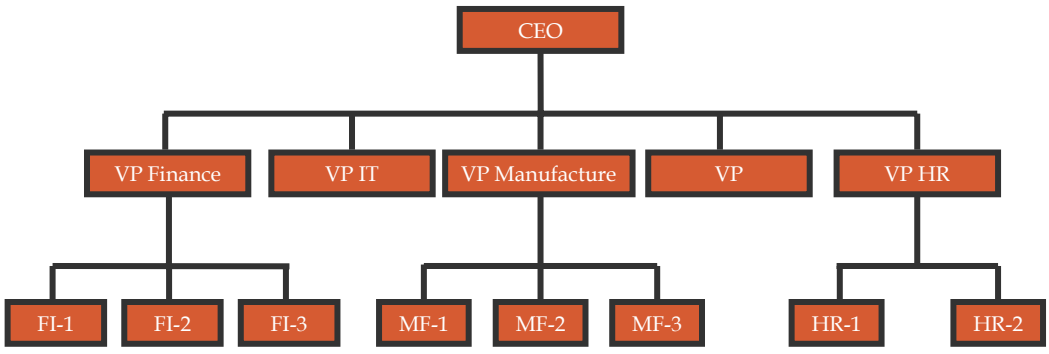

Figure 4.2 Organization Chart

Copyright by Mahidol University

Table 4.2 Role Hierarchy Table

Employee	Immediate supervisor	Top-level supervisor
CEO	Null	Null
VP Engineering	CEO	Null
VP Finance	CEO	Null
VP IT	CEO	Null
VP HR	CEO	Null
VP Manufacturing	CEO	Null
FI-1	VP Finance	CEO
FI-2	VP Finance	CEO
HR-1	VP HR	CEO
HR-2	VP HR	CEO
HR-3	VP HR	CEO
MF-1	VP Manufacturing	CEO
MF-2	VP Manufacturing	CEO
MF-3	VP Manufacturing	CEO

Table 4.3 Authority Table

emp_id	immediate_id	top-level_id
e000	Null	Null
e010	e000	Null
e020	e000	Null
e030	e000	Null
e040	e000	Null
e050	e000	Null
e021	e020	e000
e022	e020	e000
e041	e040	e000
e042	e040	e000
e043	e040	e000
e051	e050	e000
e052	e050	e000
e053	e050	e000

Copyright by Mahidol University

The role hierarchy table given in Table 4.2 can be used to set the authority table which defines what rights a person has for accessing a specific object. In addition, a role relationship table is used to define which employee belongs to which department in the organization, and also to define which table can be accessed by which department. In practice, the authority table may be different from one system to another.

Furthermore, the relationship between function and sensitive data is defined by the role policy of the organization. Normally, sensitive data should be exclusively accessed by particular personnel responsible for those data. Therefore, each attribute in the database will have a specific access policy defined via roles. For example, HR personnel can access the attribute *emp_diag* while Finance personnel can access the attribute *emp_sal*. Since the organization usually has people working on some specific role, the security policy should be defined which part of the database that particular role has specific rights to access.

- **Key Table**

The key table keeps all the keys generated for every employee in the organization. It is used to provide a proper key for submitted queries of a user. Whenever a new employee joins the organization, a new key will be generated and keep in the key table for use in data encryption and decryption of the employee. The key table used in our work is shown in Table 4.4 below.

4.2.2 Process Components

The proposed work consists of four process components: authentication module, authorization module, query translator module and query post-processor module. They are described in turn as follows.

- **Authentication module**

This module will be responsible for authenticating a user who wants to access to the database. The authentication is performed via a user name and a password. After a user is properly authenticated, the user's information will be passed to the authorization module.

Copyright by Mahidol University

Table 4.4 Key Table

emp_id	key
e000	11100011…
e010	00111010…
e020	10101011…
e030	11100010…
e040	11100111…
e050	01100011…
e021	00011010…
e022	00101011…
e041	01100010…
e042	01100111…
e043	01100001…
e051	00000010…
e052	10111111…
e053	01100000…

- **Authorization module**

This module is responsible for authorizing a user who wants to access the system. After the user has been authenticated, the user information is mapped to the authority table so that appropriate access rights will be assigned to the user query. However, if a user is a member of specific functional roles, the user will be authorized via the role functional table shown in Table 4.5. The access rights obtained will be encrypted and sent to the trusted query manager.

- **Query Translator module**

After the user has been authorized, the user will submit a plain query to the query translator that will transform the plain query into the corresponding encrypted query before sending it to the trusted query manager.

Copyright by Mahidol University

- **Query Post-Processor module**

The query post-processor module will receive the encrypted outcomes from the trusted query manager, and transform or decrypt the encrypted results to the plain results before sending them back to the user.

Table 4.5 Role Functional Table

emp_id	emp_dept	emp_sal		emp_diag	
		Read	Write	Read	Write
e020	FI	●	●	✖	✖
e021	FI	●	●	✖	✖
e022	FI	●	●	✖	✖
e040	HR	✖	✖	●	●
e041	HR	✖	✖	●	●
e042	HR	✖	✖	●	●
e043	HR	✖	✖	●	●

✖ means *deny*; ● means *accept*;

4.2.3 Trusted Query Manager

The trusted query manager is responsible for processing encrypted query received from the query translator module. The encrypted query is then processed on the secure DBMS, and the encrypted results will be obtained. Next, the trusted query manager will send the encrypted results to the query post processor module which decrypts the results to the plain form and sends them to the user. The trusted query manager consists of two significant components described as follows.

- **Key Manager**

Key manager is responsible for managing and maintaining all keys in the key table shown in Table 4.4. It also generates a new key when needed. Most keys generated are pre-computed at the time of creating the database. To use the key for encrypting and decrypting an attribute, the key manager will examine the authority table whether the attribute needs to be encrypted or whether the attribute is accessible by which employee. One attribute may have multiple encryptions. For example, the “salary” attribute may be encrypted by three keys: the owner’s key, a supervisor’s key and HR staff’s key.

Copyright by Mahidol University

- **Encrypted Database**

The encrypted database contains both sensitive and non-sensitive data. Only sensitive data will be encrypted. Which keys used to encrypt which attribute rely on the key table and the authority table which is defined by the security policy of the organization.

4.3 Data Encryption

In this work, multiple different keys will be applied for an individual record, and they do not belong to a particular owner of the record. The situation is similar to having two keys for a safe kept at a bank. To open a safe securely, one key from the owner and the other key from a responsible bank officer must be used simultaneously. If the data belongs to an employee who has two supervisors, then after being authorized, three keys (one from the owner and the other two from both supervisors) will be needed to encrypt and decrypt the data. A sample database in Figure 4.3 below shows the two attributes, *emp_sal* and *emp_diag*, encrypted with three keys.

emp_id	emp_dept	emp_name	emp_add	emp_sal	emp_diag	emp_age
e010	Engineer	Jerry	Ocean	12000	tuberculosis	39
e021	FI	Tom	Maple	15000	cancer	28
e030	IT	Mary	Main	22000	cancer	35
e041	HR	John	River	15000	heart disease	55
e053	Manufacture	Hunt	Main	20000	allergy	26

$E_{Kt}(E_{Ki}(E_{Ke}(emp_sal)))$ $E_{Kt}(E_{Ki}(E_{Ke}(emp_ diag)))$

emp_id	emp_dept	emp_name	emp_add	emp_sal	emp_diag	emp_age
e010	Engineer	Jerry	Ocean	00001100…	10001110…	39
e021	FI	Tom	Maple	11110001…	00001110…	28
e030	IT	Mary	Main	11100011…	11100000…	35
e041	HR	John	River	11110000…	11100011…	55
e053	Manufacture	Hunt	Main	00111010…	11010100…	26

Figure 4.3 Database Transformation

Copyright by Mahidol University

where: E_{Ke} denotes the encryption using the owner key;

E_{Ki} denotes the encryption using the key of the supervisor at the immediate position;

E_{Kt} : denotes the encryption using the key of the supervisor at the top position.

Since multiple keys are used to encrypt attributes in the database, we need to handle those keys properly in the proposed model. After a user has been authorized, the keys of the data owner and of owner's supervisors will be selected. If the user is one of the supervisors within the organizational structure, he or she can access all data under the supervision, and get the query results in plaintext. On the other hand, if the user wants to access sensitive data that he or she does not have any authorization, but has been assigned to a role possessing a proper authorization and responsibility, he or she can access the data and would get the query results in plaintext as well.

4.4 Query Processing over Encrypted Data

Performing a query processing over the encrypted database consists of two main functions: query translation and query post processor. When a user submits an original plaintext query, the system will transform that query into a corresponding encrypted form and send to the trusted query manager to process the query. The encrypted query results are obtained from the trusted query manager and sent to the query postprocessor for decrypting the encrypted results to the plain results and give them to the user. An example shown in Figure 4.4 below shows a query to retrieve the information of the employee id "e053" is processed, and its corresponding outputs.

Select * from employee where emp_id = e053

Query Translation

Select * from emp^s where emp^s.emp_id = e053

Copyright by Mahidol University

emp_id	emp_dept	emp_name	emp_add	emp_sal	emp_diag	emp_age
e010	Engineer	Jerry	Ocean	00001100…	10001110…	39
e021	Finance	Tom	Maple	11110001…	00001110…	28
e030	IT	Mary	Main	11100011…	11100000…	35
e041	HR	John	River	11110000…	11100011…	55
e053	Manufacturer	Hunt	Main	00111010…	11010100…	26

Figure 4.4 Encrypted Data in Encrypted Database

After examining the authority table to retrieve all levels related to the employee in the role hierarchy, part of the authority table relevant to the emp_id ‘e053’ is shown in Figure 4.5 below.

emp_id	immediate_id	top-level_id
e000	Null	Null
e050	e000	Null
e053	e050	e000

Figure 4.5 Authority Table retrieved for a Query

The results above show that the employee id ‘e053’ has both immediate supervisor id ‘e050’ and the top-level id ‘e000’. Thus, the keys of the owner and of both ids will be retrieved from the key table as shown in Figure 4.6 below. If the employee id is not found in the role hierarchy, we need to check whether the user has a role in the department that is responsible for that sensitive data or not. In this example, the departments will be HR and Finance. If the user id cannot be found in the role hierarchy and in the functional departments, the query will be rejected.

emp_id	key
e000	11100011…
e050	01100011…
e053	01100000…

Figure 4.6 Encrypted Key Used

Copyright by Mahidol University

Thus, K_e will be 01100000, K_i be 01100011, and K_t be 11100011. These keys will be used for decrypting the encrypted sensitive attribute of the employee to the plain results as shown below.

emp_diag : *11010100…* → $D_{Ke}(D_{Ki}(D_{Kt}(11010100…)))$ → allergy

emp_sal : *00111010…* → $D_{Ke}(D_{Ki}(D_{Kt}(00111010…)))$ → 20000

The plain outputs are illustrated in Figure 4.7 below.

emp_id	emp_dept	emp_name	emp_add	emp_sal	emp_diag	emp_age
e053	Manufacture	Hunt	Main	20000	allergy	26

Figure 4.7 Query Process over Encrypted Database

4.5 Query Classification

In our model, a query type is categorized into three classifications: hierarchical, functional, and unauthorized. In addition, each query is classified according to the position of the user, the function of the user and the data that the user wants to query.

4.5.1 Hierarchical Classification

In this case, a query is classified according the position of the user in the organization. In other words, the user who submits the query has the authorization to access the data because the user is the owner or the supervisor of the owner.

4.5.2 Functional Classification

In this case, a query is classified according to the role of the person in the organization. The role allows the person to access specific sensitive data even though the person does not have direct authorization on the data at all. For example, the finance personnel is allowed to access the ‘salary’ attribute of every employee in the organization.

4.5.3 Unauthorized Classification

A query in this case is classified according to the authorization of the person who submits the query. Thus, the person with no proper authorization can access only non-sensitive data. If the person tried to access sensitive data, the data will be displayed in

Copyright by Mahidol University

encrypted form which is not understandable since there is no proper key to decrypt the sensitive data.

4.6 Designing Sample Queries

A set of queries is designed to evaluate the performance of the proposed model based on the three query classifications described in the previous section. We have seven sample queries in total. They are briefly explained as follows.

(1) A user submits a query to read his/her own data.

(2) A supervisor submits a query to read the data owned by his/her subordinates.

(3) A user in the HR department adds the information of a new employee in the Manufacturer department.

(4) A user in the Finance department updates the salary of an employee in the Manufacturer department.

(5) A user in the Finance department wants to read the salary of an employee in the Manufacturer department.

(6) An employee in the Manufacturer department submits a query to read the information of another employee in the same department.

(7) All users submit queries either with or without proper authorization.

The followings give the details of each sample query showing in a box and the processing of the query.

4.6.1 Sample Query 1

Sample Query 1: A user submits a query to read his/her own data.

- **User:** MF-3
- **Query:** Select * from employee where emp_id=’e053’

Copyright by Mahidol University

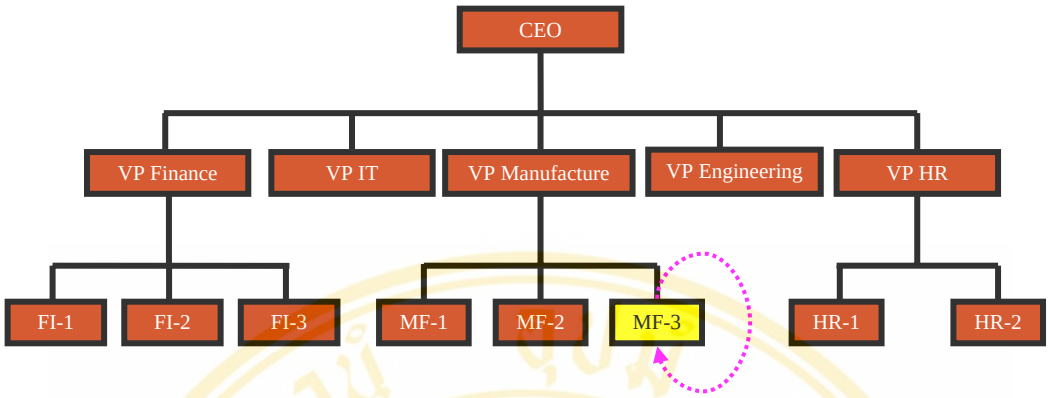

Figure 4.8 User MF-3 and Sample Query 1

Step 1: Authentication and Authorization

(1) The user logons with the user id of MF-3;

The system identifies the user as one of the manufacture staff and MF-3 is the owner of the data.

(2) The user, MF-3, submits the following query;

*Select * from employee where emp_id='e053'*

(3) The system returns the access rights of the user, MF-3, shown below:

emp_id	R-emp_sal	W-emp_sal	R-emp_diag	W-emp_diag
e053	●	●	●	●

● means *accept*;

Figure 4.9 Access Rights of Sample Query 1

(4) The system sends the plain query to the query translator and sends the access rights to the trusted query manager.

Step 2: Query Processing

(5) The system identifies a set of sensitive attributes from the submitted query, and sends them and the user id to the key retrieval process.

Sensitive field : emp_sal, emp_diag

userid : MF-3

Copyright by Mahidol University

Step 3: Key Retrieval

(6) Retrieve the corresponding keys from the key table.

emp_id = 'e053' and field_name = 'emp_sal'

emp_id = 'e053' and field_name = 'emp_diag'

(7) Send the MF-3's key for encrypting both attributes: *emp_sal* and *emp_diag* to the trusted query manager.

Step 4: Query Translator

(8) Transform the plain query to the encrypted query as shown below.

Select * from employee where emp_id='e053'

Query Translation

Select * from emp^S where emp_id='e053'

(9) The encrypted query is submitted to the trusted query manager, and the encrypted form of outputs is obtained as follows.

emp_id	emp_dept	emp_name	emp_add	emp_sal	emp_diag	emp_age
e053	Manufacture	Hunt	Main	00111010…	11010100…	26

Figure 4.10 Encrypted Query Results of Sample Query 1

Step 5: Query Post Processor

(10) The plain results are displayed to the user, MF-3, as shown in Figure 4.11 after going thru the decryption process.

emp_diag : 11010100… → $D_{Ke}(D_{Ki}(D_{Kt}(11010100…)))$ → *allergy*

emp_sal : 00111010... → $D_{Ke}(D_{Ki}(D_{Kt}(00111010...)))$ → *20000*

emp_id	emp_dept	emp_name	emp_add	emp_sal	emp_diag	emp_age
e053	Manufacture	Hunt	Main	20000	allergy	26

Figure 4.11 Plain Results of Sample Query 1

Copyright by Mahidol University

4.6.2 Sample Query 2

Sample Query 2 : A supervisor submits a query to read the data owned by his/her subordinates.

- **User:** VP Manufacture
- **Query:** Select * from employee where emp_id='e053'

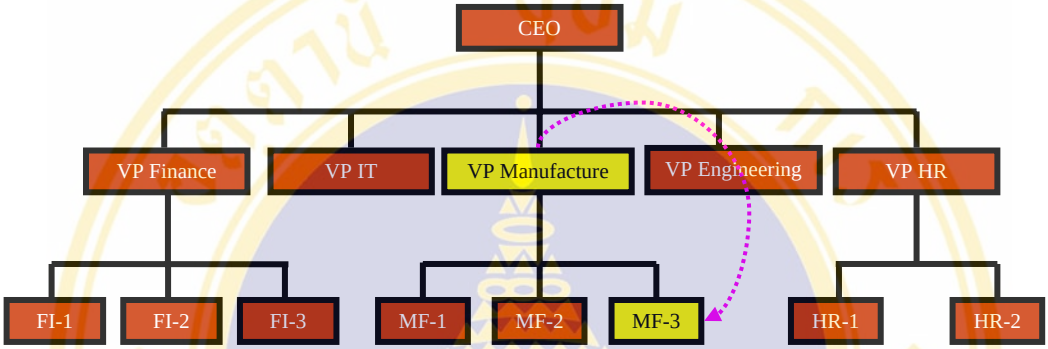

Figure 4.12 User VP and Sample Query 2

Step 1: Authentication and Authorization

(1) The user logons with the user id of VP manufacture.

(2) The user, VP manufacture, submits the following query:

*Select * from employee where emp_id='e053'*

(3) The system returns the access rights of the user, VP manufacture, as shown below:

emp_id	R-emp_sal	W-emp_sal	R-emp_diag	W-emp_diag
e053	●	●	●	●

✖ means *deny*; ● means *accept*;

Figure 4.13 Access Rights for Sample Query 2

(4) The system sends the plain query to the query translator and sends the access rights to the trusted query manager.

Step 2: Query Processing

(5) The system identifies a set of sensitive attributes from the submitted query and sends them to the key retrieval and the user id.

Copyright by Mahidol University

*Select * from employee where emp_id='e053'*

Sensitive field : emp_sal, emp_diag

User id: VP Manufacture

Step 3: Key Retrieval

(6) Retrieve the corresponding keys from the key table.

emp_id = 'e053' and field_name = 'emp_sal'

emp_id = 'e053' and field_name = 'emp_diag'

(7) Send the MF-3's key for encrypting attributes: emp_sal and emp_diag to the trusted query manager.

Step 4: Query Translator

(8) Transform the plain query to the encrypted query.

Select * from employee where emp_id='e053'

Query Translation

Select * from emp^S where emp_id='e053

(9) The query is submitted to the trusted query manager and the encrypted form of results is obtained as follows.

emp_id	emp_dept	emp_name	emp_add	emp_sal	emp_diag	emp_age
e053	Manufacture	Hunt	Main	00111010…	11010100…	26

Figure 4.14 Encrypted Query Results in Sample Query 2

Step 5: Query Post Processor

(10) The plain results are displayed to the user, VP manufacture, as shown in Figure 4.15 after going thru the decryption process below.

emp_diag : 11010100… → $D_{Ke}(D_{Ki}(D_{Kt}(11010100…)))$ → *allergy*

emp_sal : 00111010... → $D_{Ke}(D_{Ki}(D_{Kt}(00111010...)))$ → *20000*

emp_id	emp_dept	emp_name	emp_add	emp_sal	emp_diag	emp_age
e053	Manufacture	Hunt	Main	20000	allergy	26

Figure 4.15 Plain Results of Sample Query 2

Copyright by Mahidol University

4.6.3 Sample Query 3

Sample Query 3 : A user in the HR department adds the information of a new employee in the Manufacturer department.

- **User:** HR-1
- **Query:** Insert into employee (emp_id, emp_name, emp_add, emp_diag, emp_age) values ('e053', 'Hunt', 'Main', 'allergy', '26')

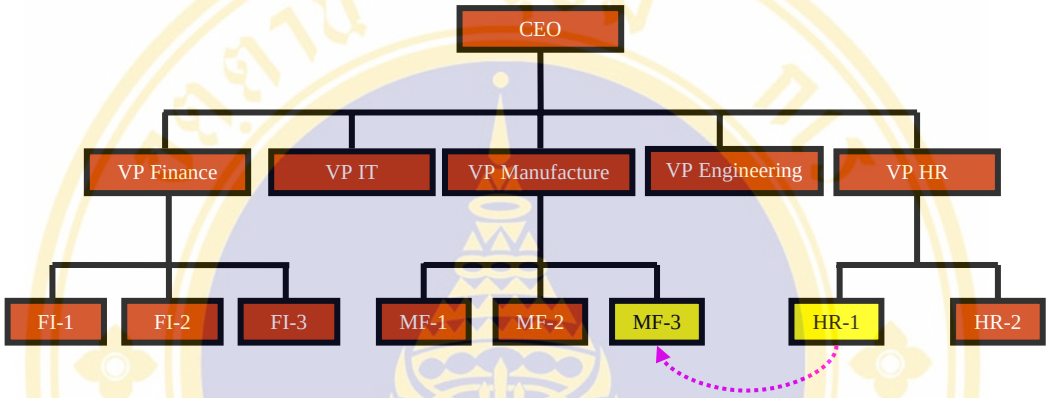

Figure 4.16 User and Query for Sample Query 3

Step 1 Authentication and Authorization

(1) The user logons with the userid of HR-1;

The system identifies the user as one of the staff HR and MF-3 is the owner of the data.

(2) The user, HR-1, submits the following query;

Insert into employee (emp_id, emp_name, emp_add, emp_diag, emp_age) values ('e053', 'Hunt', 'Main', 'allergy', '26')

(3) The system returns the access rights of the user, HR-1, shown below:

emp_id	R-emp_sal	W-emp_sal	R-emp_diag	W-emp_diag
e010	✖	✖	●	●
e021	✖	✖	●	●
e030	✖	✖	●	●
e041	✖	✖	●	●
e053	✖	✖	●	●

✖ means *deny*; ● means *accept*;

Figure 4.17 Access Rights in Sample Query 3

Copyright by Mahidol University

(4) The system sends the plain query to the query translator and sends the access rights to the trusted query manager.

Step 2 Query Processing

(5) The system identifies a set of sensitive attributes from the submitted query and sends them to the key retrieval and the user id.

Insert into employee (emp_id, emp_name, emp_add, emp_diag, emp_age) values ('e053', 'Hunt', 'Main', 'allergy', '26')

Sensitive field : emp_diag

userid : HR-1

Step 3 Key Data Process

(6) Retrieve the corresponding keys from the key table.

emp_id = 'e053' and field_name = 'emp_diag'

(7) Update the new key into the key table.

(8) Send the MF-3's key for encrypting attributes: emp_diag to the trusted query manager.

Step 4 Query Translator

(9) Transform the plain query to the encrypted query;

- Encrypt/Decrypt process

  emp_diag : allergy → $E_{Ke}(E_{Ki}(E_{Ke}(allergy)))$ → *11010100…*

- Query translator

  Insert into employee (emp_id, emp_name, emp_add, emp_diag, emp_age) values ('e053', 'Hunt', 'Main', 'allergy', '26')

  Query Translation

  Insert into emp^S (emp_id, emp_name, emp_add, emp_diag, emp_age) values ('e053', 'Hunt', 'Main', '11010100…', '26')

(10) The query is submitted to the trusted query manager and the encrypted form of results is obtained as follows.

Copyright by Mahidol University

emp_id	emp_dept	emp_name	emp_add	emp_sal	emp_diag	emp_age
e053	Manufacture	Hunt	Main	*Null*	11010100…	26

Figure 4.18 Encrypted Query Results in Sample Query 3

4.6.4 Sample Query 4

Sample Query 4 : A user in the Finance department updates the salary of an employee in the Manufacturer department.

- **User:** FI-1
- **Query:** Update employee set emp_sal = ‘20000’ where emp_id=‘e053’

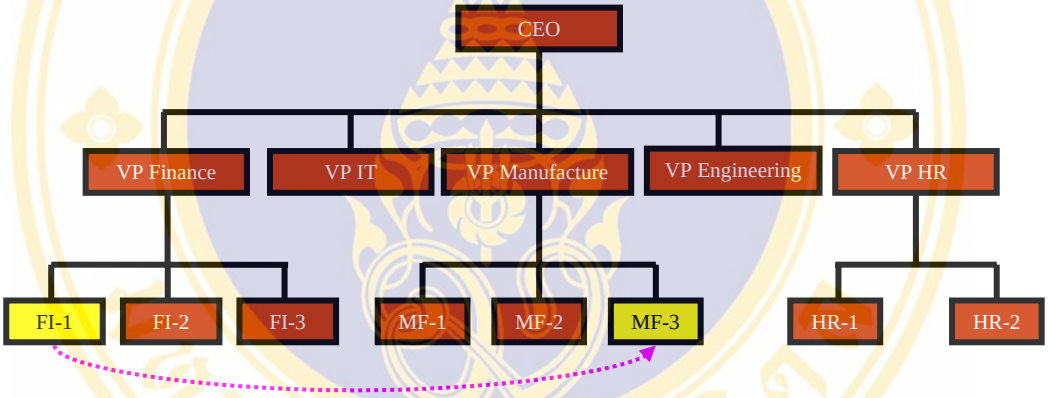

Figure 4.19 User and Query for Sample Query 4

Step 1 Authentication and Authorization

(1) The user logons with the userid of FI-1;

The system identifies the user as one of the staff Finance and MF-3 is the owner of the data.

(2) The user, FI-1, submits the following query;

Update employee set emp_sal = ‘20000’ where emp_id=‘e053’

(3) The system returns the access rights of the user, FI-1, shown below:

Copyright by Mahidol University

emp_id	R-emp_sal	W-emp_sal	R-emp_diag	W-emp_diag
e010	●	●	✖	✖
e021	●	●	✖	✖
e030	●	●	✖	✖
e041	●	●	✖	✖
e053	●	●	✖	✖

✖ means *deny*; ● means *accept*;

Figure 4.20 Access Rights in Sample Query 4

(4) The system sends the plain query to the query translator and sends the access rights to the trusted query manager.

Step 2 Query Processing

(5) The system identifies a set of sensitive attributes from the submitted query and sends them to the key retrieval and the user id.

Update employee set emp_sal = '20000' where emp_id='e053'

Sensitive field : emp_sal

userid : FI-1

Step 3 Key Data Process

(6) Retrieve the corresponding keys from the key table.

emp_id = 'e053' and field_name = 'emp_sal'

(7) Send the MF-3's key for encrypting attributes: emp_sal and emp_diag to the trusted query manager.

Step 4 Query Translator

(8) Transform the plain query to the encrypted query;

- Encrypt/Decrypt process

  emp_sal : 20000 → $E_{Kt}(E_{Ki}(E_{Ke}(20000)))$ → *00111010...*

- Query translator

  Update employee set emp_sal = '20000' where emp_id='e053'

Query Translation

Copyright by Mahidol University

Update emp^{S} set emp_sal = '00111010...' where emp_id='e053'

(9) The query is submitted to the trusted query manager and the encrypted form of results is obtained as follows.

emp_id	emp_dept	emp_name	emp_add	emp_sal	emp_diag	emp_age
e053	Manufacture	Hunt	Main	00111010…	11010100…	26

Figure 4.21 Encrypted Query Results in Sample Query 4

4.6.5 Sample Query 5

Sample Query 5 : A user in the Finance department wants to read the salary of an employee in the Manufacturer department.

- **User:** FI-1
- **Query:** Select * from employee where emp_id='e053'

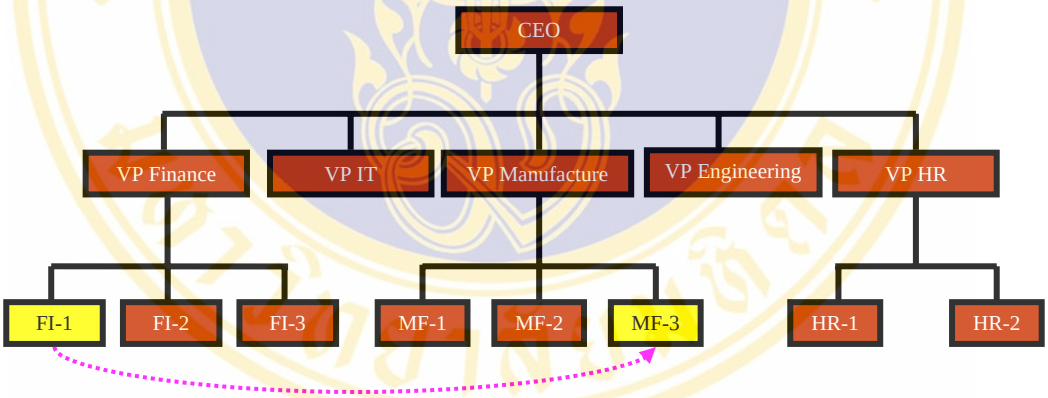

Figure 4.22 User and Query for Sample Query 5

Step 1 Authentication and Authorization

(1) The user logons with the userid of FI-1;

The system identifies the user as one of the staff Finance and MF-3 is the owner of the data.

(2) The user, FI-1, submits the following query;

*Select * from employee where emp_id='e053'*

(3) The system returns the access rights of the user, FI-1, shown below:

Copyright by Mahidol University

emp_id	R-emp_sal	W-emp_sal	R-emp_diag	W-emp_diag
e010	●	●	✖	✖
e021	●	●	✖	✖
e030	●	●	✖	✖
e041	●	●	✖	✖
e053	●	●	✖	✖

✖ means *deny*; ● means *accept*;

Figure 4.23 Access Rights in Sample Query 5

(4) The system sends the plain query to the query translator and sends the access rights to the trusted query manager.

Step 2 Query Processing

(5) The system identifies a set of sensitive attributes from the submitted query and sends them to the key retrieval and the user id.

*Select * from employee where emp_id='e053'*

Sensitive field : emp_diag, emp_sal

userid : FI-1

Step 3 Key Data Process

(6) Retrieve the corresponding keys from the key table.

emp_id = 'e053' and field_name = 'emp_sal'

emp_id = 'e053' and field_name = 'emp_diag'

(7) Send the MF-3's key for encrypting attributes: emp_sal and emp_diag to the trusted query manager.

Step 4 Query Translator

(8) Transform the plain query to the encrypted query;

- Query translator

Select * from employee where emp_id='e053'

Query Translation

Select * from emp^S where emp_id='e053'

Copyright by Mahidol University

(9) The query is submitted to the trusted query manager and the encrypted form of results is obtained as follows.

emp_id	emp_dept	emp_name	emp_add	emp_sal	emp_diag	emp_age
e053	Manufacture	Hunt	Main	00111010…	11010100…	26

Figure 4.24 Encrypted Query Results in Sample Query 5

Step 5 Query Post Processor

(10) The plain results are displayed to the user, FI-1, as shown in Figure 4.25 after going thru the decryption process.

emp_diag : not authorized

emp_sal : 00111010... $\rightarrow D_{Ke}(D_{Ki}(D_{Kt}(00111010...)))$ $\rightarrow$ *20000*

emp_id	emp_dept	emp_name	emp_add	emp_sal	emp_diag	emp_age
e053	Manufacture	Hunt	Main	20000	11010100…	26

Figure 4.25 Plain Query Results in Sample Query 5

4.6.6 Sample Query 6

Sample Query 6 : An employee in the Manufacturer department submits a query to read information of another employee in the same department.

- **User:** MF-1
- **Query:** Select * from employee where emp_id='e053'

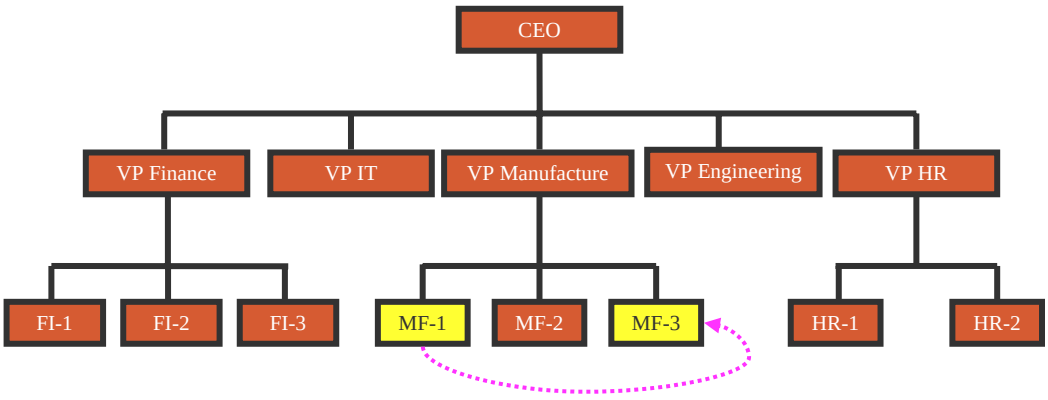

Figure 4.26 User and Query for Sample Query 6

Copyright by Mahidol University

Step 1 Authentication and Authorization

(1) The user logons with the userid of MF-1;

The system identifies the user as one of the staff Manufacture and MF-3 is the owner of the data.

(2) The user, MF-1, submits the following query;

*Select * from employee where emp_id='e053'*

(3) The system returns the access rights of the user, MF-1, shown below:

emp_id	R-emp_sal	W-emp_sal	R-emp_diag	W-emp_diag
e010	✖	✖	✖	✖
e021	✖	✖	✖	✖
e030	✖	✖	✖	✖
e041	✖	✖	✖	✖
e053	✖	✖	✖	✖

✖ means *deny*; ● means *accept*;

Figure 4.27 Access Rights in Sample Query 6

(4) The system sends the plain query to the query translator and sends the access rights to the trusted query manager.

Step 2 Query Processing

(5) The system identifies a set of sensitive attributes from the submitted query and sends them to the key retrieval and the user id.

*Select * from employee where emp_id='e053'*

Sensitive field :emp_diag, emp_sal

userid : MF-1

Step 3 Key Data Process

(6) Retrieve the corresponding keys from the key table.

emp_id = 'e053' and field_name = 'emp_sal'

emp_id = 'e053' and field_name = 'emp_diag'

(7) Send the MF-3's key for encrypting attributes: emp_sal and emp_diag to the trusted query manager.

Copyright by Mahidol University

Step 4 Query Translator

(8) Transform the plain query to the encrypted query;

- Query translator

Select * from employee where emp_id='e053'

Query Translation

Select * from emp^S where emp_id='e053'

(9) The query is submitted to the trusted query manager and the encrypted form of results is obtained as follows.

emp_id	emp_dept	emp_name	emp_add	emp_sal	emp_diag	emp_age
e053	Manufacture	Hunt	Main	00111010…	11010100…	26

Figure 4.28 Encrypted Query Results in Sample Query 6

Step 5 Query Post Processor

(10) The plain results are displayed to the user, MF-1, as shown in Figure 4.29 after going thru the decryption process.

emp_diag : not authorized

emp_sal : not authorized

emp_id	emp_dept	emp_name	emp_add	emp_sal	emp_diag	emp_age
e053	Manufacture	Hunt	Main	00111010…	11010100…	26

Figure 4.29 Plain Query Results in Sample Query 6

Copyright by Mahidol University

CHAPTER V
IMPLEMENTATION AND EXPERIMENTS

This chapter presents the implementation of our proposed work. We describe the system architecture, hardware and software requirements, and database generation. Next, we explain all of the six experiments conducted based on the queries presented in Chapter 4. Finally, the experimental results are presented and discussed.

5.1 System Architecture

Figure 5.1 presents the system architecture implemented based on our proposed model. The system consists of the database server, the web server, and the client. They all implemented on one PC machine in order to avoid the communication cost when the response time is measured for a query request.

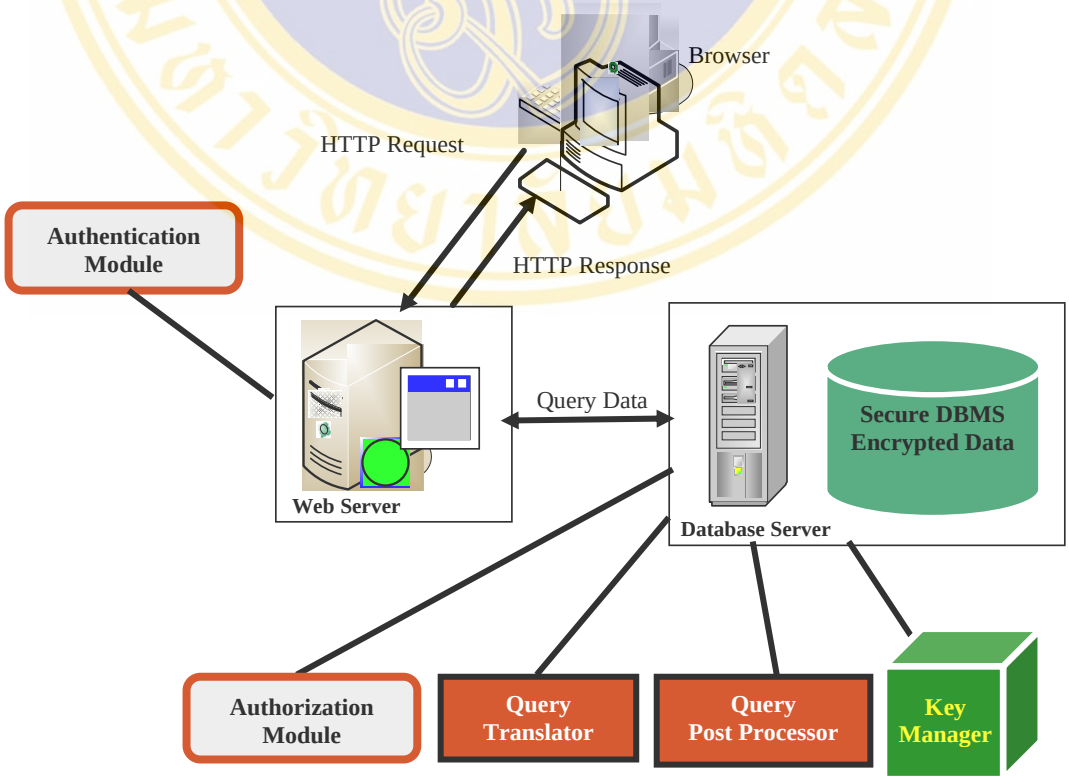

Figure 5.1 System Architecture

Copyright by Mahidol University

5.2 System Requirements

- The hardware used for implementing our proposed model has the Intel Pentium processor of 1.6 GHz speed and 768 MB of RAM.
- The software used in our implementation includes the following:
  (1) Microsoft Windows XP Professional as the operating system;
  (2) Microsoft SQL Server 2000 as the database server;
  (3) Internet Information Services (IIS) as the web server;
  (4) EditPlus text editor v2.11 as the editor tool;
  (5) aspEasyCrypt v2.10 as the cryptographic tool used.

5.3 Experiment Setup

In order to evaluate the effects of having some attributes encrypted in the database, we set up three environments based on the nature of the database used. In other words, we have three databases generated. They are:

(1) Plain database with no encryption over all attributes and having the following schema:

Employee(ID, EmpName, EmpAdd, EmpDiag, EmpSal, EmpAge)

(2) Mixed database where only sensitive data are encrypted. In our experiments, only diagnosis and salary are encrypted in the schema below:

Employee(ID, EmpName, EmpAdd, EmpDiag, EmpSal, EmpAge)

(3) Pure encrypted database where all attributes except the attribute “ID” are encrypted into “EmpEnc” as shown in the following schema:

Employee(ID, EmpEnc)

In addition, the number of tuples generated in each database is approximately 100,000 tuples. The encryption algorithm used in our implementation is DES that both the data block size and the key size are 64 bits. Also, the encryption mode applied is cmCTS (Cipher Text Stealing), a variant of cmCBC, which gives relaxed restriction and fast.

Copyright by Mahidol University

5.4 Experiments and Results

In this work, we carry out six experiments for each database environment setup. In each experiment, the number of queries is generated and ranged from 50 to 1000. The average response time is measured for all queries submitted.

5.4.1 Query Generation

Our database consists of five attributes. Thus, the experiments will generate 31 (2^5-1) combinations of query types, and they are shown in Table 5.1. All 31 query types can be classified into three groups. The first query group, Q: 1, 2, 5, 6, 9, 12 and 18, accesses non-sensitive data only. The second query group, Q: 3, 4, 7, 8, 10, 11, 14, 15, 16, 17, 20, 21, 23, 24, 27 and 28, simultaneously accesses both sensitive and non-sensitive data, but only one sensitive attribute will be retrieved. The last query group, Q: 13, 19, 22, 25, 26, 29, 30 and 31, is similar to the second query group except that eventually, both sensitive attributes will be retrieved.

The objectives of classifying those queries into three groups are to differentiate the characteristics of accessing sensitive and non-sensitive data with respect to the query response time as it will be contributed from the encryption and decryption processes.

5.4.2 Query Experiment 1

In this experiment, a query is submitted by the data owner who has full authority to access the data. In other words, each query will have only one attribute retrieved and decrypted. For all 1000 transactions, different user ID is randomly generated for accessing the data of their own. We run this experiment for all three database environments, and the results are shown in Figure 5.2 below.

Copyright by Mahidol University

Table 5.1 All 31 Query Types

Query no.	emp_name	emp_add	emp_sal	emp_diag	emp_age
1	●	-	-	-	-
2	-	●	-	-	-
3	-	-	●	-	-
4	-	-	-	●	-
5	-	-	-	-	●
6	●	●	-	-	-
7	●	-	●	-	-
8	●	-	-	●	-
9	●	-	-	-	●
10	-	●	●	-	-
11	-	●	-	●	-
12	-	●	-	-	●
13	-	-	●	●	-
14	-	-	●	-	●
15	-	-	-	●	●
16	●	●	●	-	-
17	●	●	-	●	-
18	●	●	-	-	●
19	●	-	●	●	-
20	●	-	●	-	●
21	●	-	-	●	●
22	-	●	●	●	-
23	-	●	●	-	●
24	-	●	-	●	●
25	-	-	●	●	●
26	●	●	●	●	-
27	●	●	●	-	●
28	●	●	-	●	●
29	●	-	●	●	●
30	-	●	●	●	●
31	●	●	●	●	●

●Attribute to be accessed in a query.

Copyright by Mahidol University

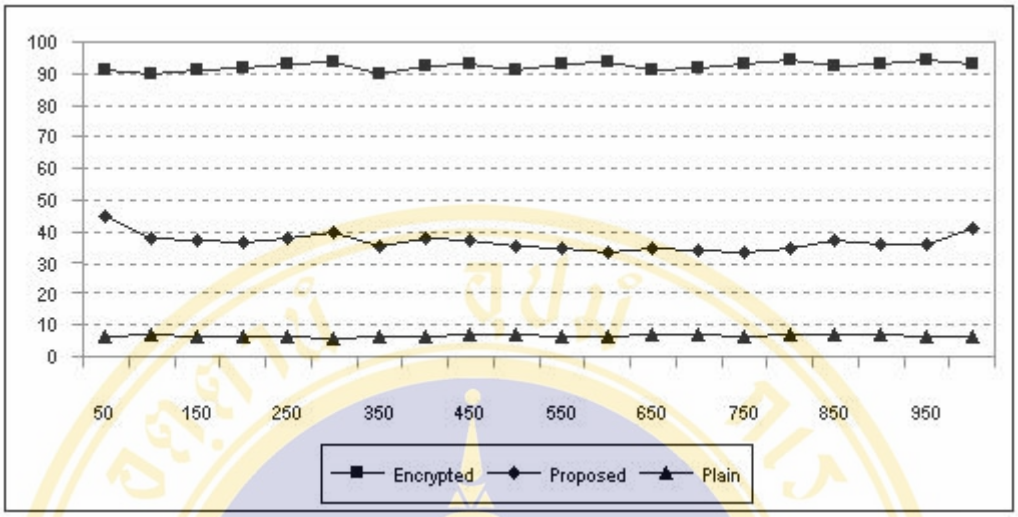

Figure 5.2 Results of Query by Owners

Obviously, the experimental results show the significant effects of encryption to the query response time. In other words, the query response time of the plain database is much lower than those of both the proposed and encrypted databases. The explanation is that there is no encryption cost on the plain database. The results also show that the query response time of the proposed model is higher than that of the plain query, and much lower than that of the query response time of the encrypted database. The main reason for these results is the cost of encryption and decryption always performed by the pure encrypted model whereas the proposed model will perform the encryption or decryption solely on the attributes accessed by a query.

5.4.3 Experiment 2

In this experiment, a query is submitted by the data owner's supervisor who also has the right to access the data owned by an employee under his/her supervision. For each transaction, different user ID of a supervisor is randomly generated for accessing any attributes in the database. This experiment is run for all three database environments, and the results are shown in Figure 5.3 below.

Copyright by Mahidol University

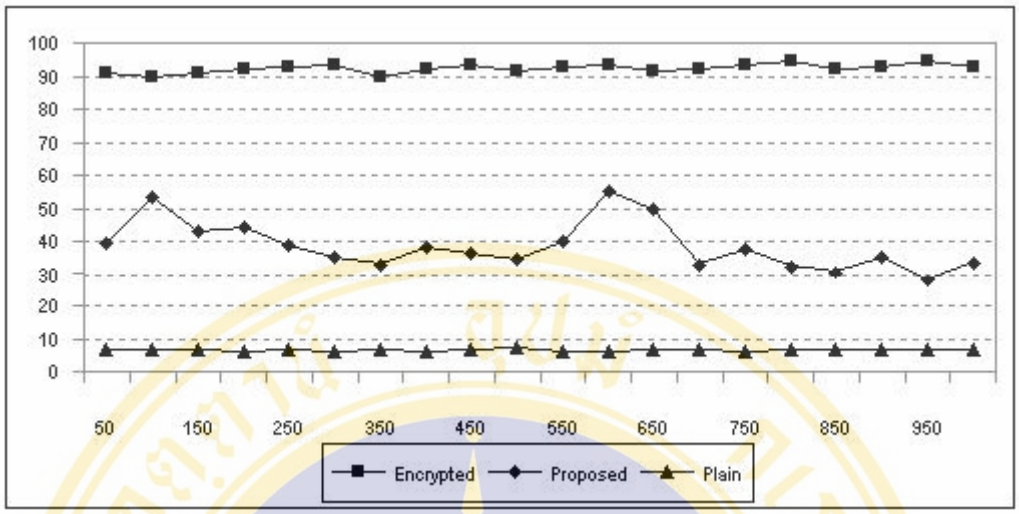

Figure 5.3 Results of Query by Supervisors

The results of this experiment show a similar trend with those of Experiment 1 in such a way that the query response time of the plain database is still much lower than those of the proposed model and the encrypted database. However, the query response time in the proposed model is mostly a little higher than that in Experiment 1 since the supervisor has to access and decrypt the data owned by his/her subordinates. In addition, the query response times of the proposed model are still much lower than the encrypted database since the cost of encryption of the encrypted database is still higher.

5.4.4 Query Experiment 3

In this experiment, a query is to perform a data insertion. Each query will randomly generate a new employee and his or her data will be inserted to the HR department. According to our proposed model, the HR department is able to access individual employee's diagnosis. Thus, whenever a new employee data is inserted, the attribute 'diagnosis' will be encrypted as well. For the plain database, the attribute is not encrypted, but for the encrypted database, the whole tuple is encrypted. The results are shown in Figure 5.4 below.

Copyright by Mahidol University

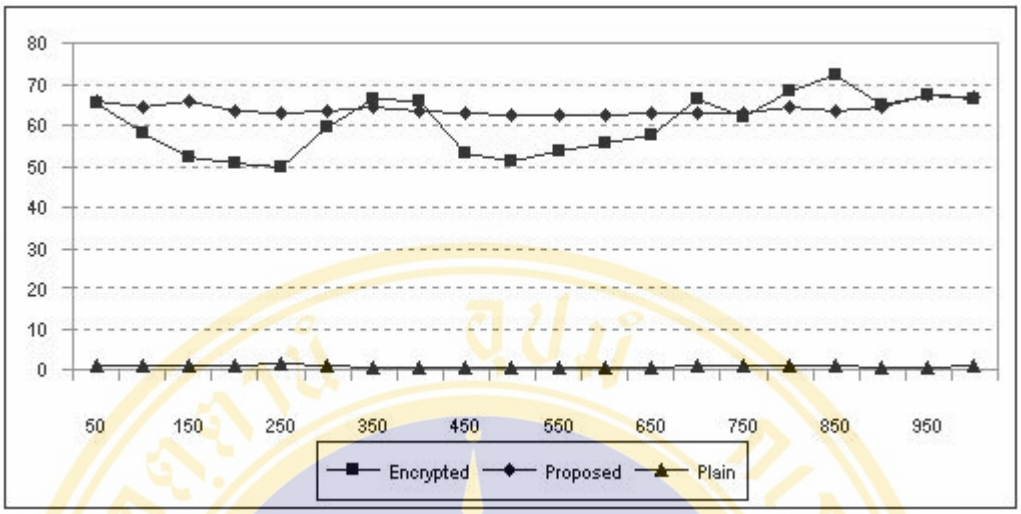

Figure 5.4 Results of Insertion Query

The results illustrate that the query response time for both the proposed model and the encrypted database is higher than that of the plain database because both the proposed model and the encrypted database have to encrypt data, and the process makes high response time for both. On the other hand, the plain database gives a low response time because there is no need to do encryption since the cost of inserting new tuples to the database is very small whereas the cost of encryption incurred is quite high.

5.4.5 Query Experiment 4

This experiment generates a set of queries to perform updates to the data in the database. However, only a certain group of employee, who has the responsibility for updating a particular data, can update the data. For the plain database, the updates are straightforward since no encryption is required, while for the proposed model, only the sensitive data are updated or re-encrypted, and for the encrypted database, the whole tuple must be decrypted and re-encrypted.

Copyright by Mahidol University

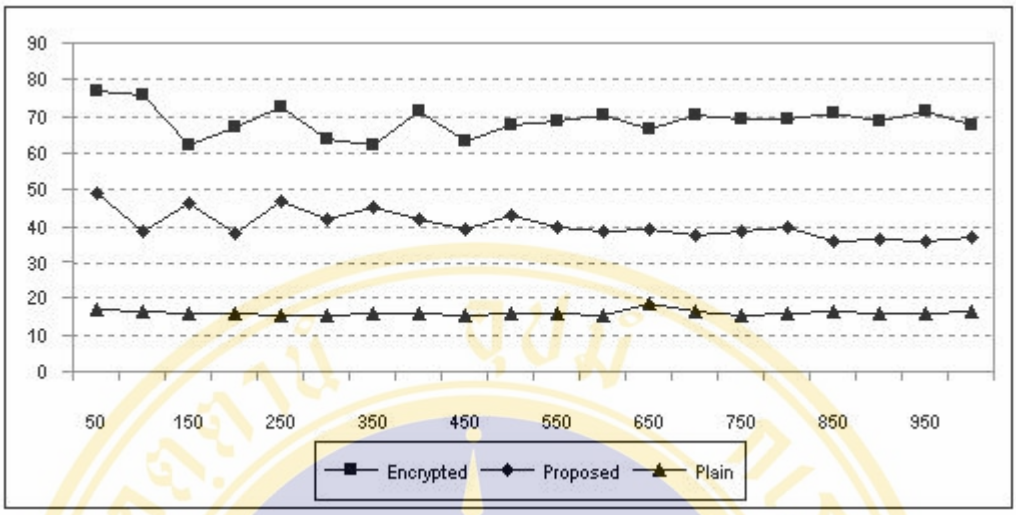

Figure 5.5 Results of Update Query

The results shown in Figure 5.5 illustrate that the plain database has almost the constant query response time when a tuple is updated due to no encryption cost even though the query response time in this case is higher than those of both the selection and insertion. The query response time of the encrypted database is higher since the cost of decryption and re-encryption the whole updated tuple would be high when more tuples are updated. However, the query response time in the proposed model is lower than that of the encrypted database since only sensitive attributes need to be updated and re-encrypted.

5.4.6 Query Experiment 5

In this experiment, a query is submitted by a user who has the authority over some sensitive data since he or she is a member of a particular role. For example, a staff in the finance department can access the ‘salary’ attribute of an employee working in the manufacture department. Our experiment will randomly generate both a designated role selected from all departments, and a user who will access sensitive attributes of an employee in other departments. Thus, a generated user may be able to access every attribute of one employee, and may be able to access only the sensitive attributes of another employee. The results of this experiment are shown in Figure 5.6 below.

Copyright by Mahidol University

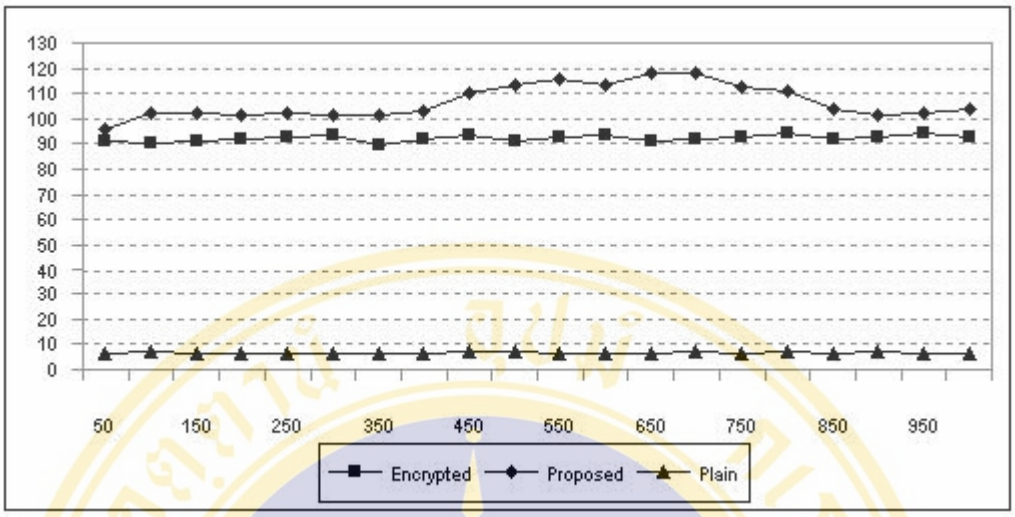

Figure 5.6 Results of Query by Roles

The results in this experiment illustrate that the query response time of the plain databases is almost constant and gives a low response time. On the other hand, the query response time of the proposed model is little higher than that of the encrypted database since users, who have the rights to access data, have no authorization in the hierarchy level, and thus, the proposed model has to examine the user role for every query. As a result, the proposed model has to spend time to verify the access rights of users as well as to perform encryption and decryption.

5.4.7 Query Experiment 6

In this experiment, unauthorized persons submitted generated queries to access sensitive attributes. The outcome would be a rejection since the users have no rights and the data cannot be read since they are in an encrypted form. For example, an employee submits a query to access the salary of another employee in the same department. The results of this experiment are shown in Figure 5.7 below.

Copyright by Mahidol University

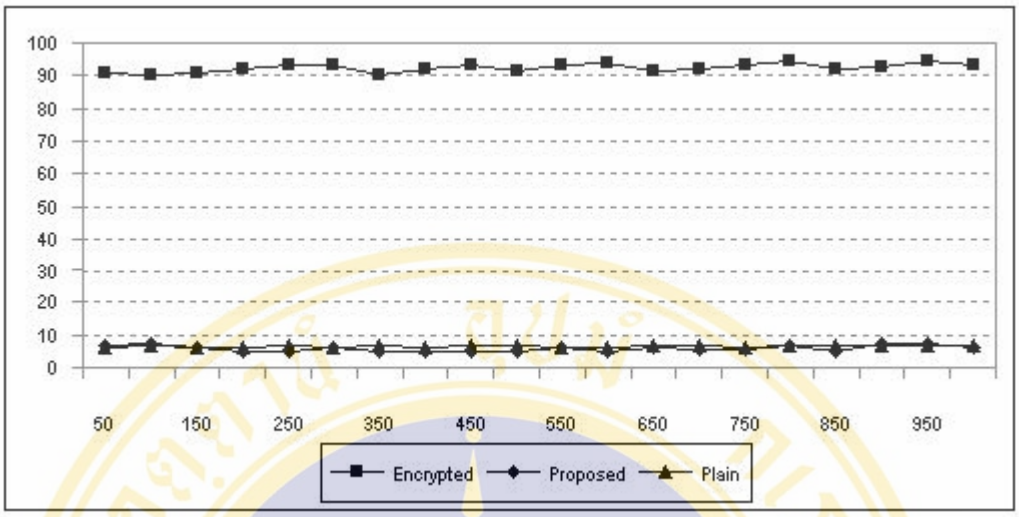

Figure 5.7 Results of Query by Unauthorized Persons

The results illustrated that the query response time of all databases are mostly constant. In addition, the encrypted database gives higher query response time than that of the proposed and plain database. For the plain database, the results show that all tuples will be accessed and retrieved regardless of the authority of the users who submitted a query. The proposed model demonstrates the same low query response time as the plain database since all tuples are accessed, but not decrypted and retrieved since users are unauthorized.

5.5 Discussion of Experimental Results

Experiments 1 and 2 give similar results because both the owner and his/her supervisor have the same authorization. Moreover, Experiment 3 presents the outcomes of inserting new data and the results are comparable between the proposed model and the encrypted database due to similar encryption costs. The results in Experiment 4 demonstrate that the encryption cost of updating only sensitive attributes would be less than that of performing the whole tuples. The results from Experiment 5 are the only outcome that our proposed model is outperformed by the encrypted database. It is the case of applying RBAC to the model and the results confirm that the cost of examining access rights in RBAC is quite high. The results in Experiment 6 also confirm this reason as the results of the proposed model are almost the same as

Copyright by Mahidol University

the plain database since all queries are rejected, and no need to examine access rights in RBAC.

In summary, our proposed model indicates that, in most cases, encrypting only sensitive attributes gives the better query response time than encrypting the whole tuple in the database. In addition, the encryption cost is not as high as we expected, and can be varied according to the number of attributes to be encrypted as the results are shown in Figure 5.8 below.

In Figure 5.8, we put the experiment results in another perspective. All queries in Table 5.1 are classified into three groups according to the number of sensitive attributes encrypted. The three groups are shown in Table 5.2 below.

Table 5.2 Query Group Based on Sensitive Attributes

Group 1: No sensitive attributes	
1	*One non-sensitive attribute*
2	*Two non-sensitive attribute*
3	*Three non-sensitive attribute*
Group 2: One sensitive attributes	
4	*One sensitive attribute*
5	*One sensitive attribute & one non-sensitive attribute*
6	*One sensitive attribute & two non-sensitive attributes*
7	*One sensitive attribute & three non-sensitive attributes*
Group 3: Two sensitive attributes	
8	*Two sensitive attributes*
9	*Two sensitive attributes & one non-sensitive attribute*
10	*Two sensitive attributes & two non-sensitive attributes*
11	*Two sensitive attributes & three non-sensitive attributes*

Figure 5.8 illustrates the query response time of the three databases with respect to the number of sensitive attributes to be encrypted. As can be seen that the query response time of the plain database gives the lowest response time, and the query response time of the encrypted database gives the highest response time. In addition, both query response times are comparable in all three groups. On the other hand, the query response time of the proposed model is increased if the number of encrypted attributes is increased for each group.

Copyright by Mahidol University

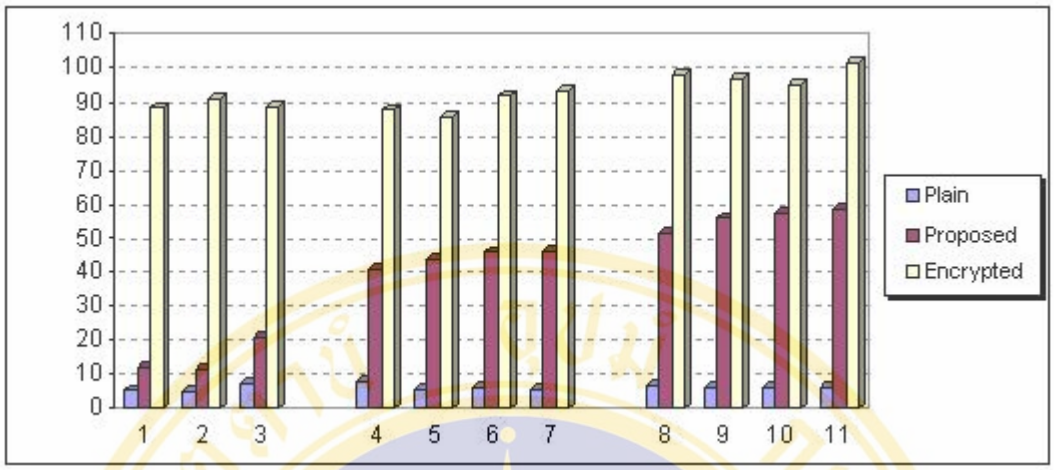

Figure 5.8 Results of Query by Groups

Copyright by Mahidol University

CHAPTER VI
DISCUSSION AND CONCLUSIONS

This chapter gives the conclusions and suggests our future work. We also summarize the proposed work, discuss the experimental results, as well as conclude our contributions.

6.1 Summary of the Proposed Work

The propose work designs and implements an encrypted database. The main motivation of our work is that even the most sensitive data in an organization have been accessed by people who do not have any authorization regardless of both the protection techniques put on the database and even on the server that it resides on, and the rules used to classify who can access which items on the database. Currently, most data in an organization are stored in the database or even in a service provider database. Those data have a high degree of risks of being accessed by unauthorized persons, attackers, or even database administrators who possess high privileges, technical skills, and full opportunity. In addition, recent research concludes that most attacks or illegal accesses to the data often happened inside the organization and mostly by insiders, not outsiders.

To hide those sensitive data from unauthorized accesses, the data needs to be encrypted with some keys hold only by authorized persons. But, adding the encryption to the database would cost the degradation in the database performance due to several influences such as high processing time required for encrypting and decrypting data, the complexity of applying encryption to specific attributes or the whole tuple, and the cost of key management. Moreover, we must ensure that the database has the high degree of security.

The main purpose of our research is to keep specific sensitive data in a database confidential at a certain degree and even the database's DBA cannot see them. The process of encryption is also transparent to users. In other words, users can submit

Copyright by Mahidol University

queries using the common formats as they used to do. Furthermore, the cost of encryption should not be so high so that the database performance can be kept at an acceptable level even though the cost of encryption and decryption processes would not be negligible.

Our work adds two essential components to a query model used in a normal database. The first component is the pre-query and post-query translators which translates a plain query to its corresponding encrypted query and vice versa. The second component is the key manager which not only maintains the key table and the authority table, but also assigns the right key for a person working at a particular role in the organization. The query translators hide sensitive data whenever it will be exposed outside the database, while the key manager is responsible for enforcing the role-based security policy defined by the organization executives.

The performance of our proposed work is evaluated by implementing a real prototype using Microsoft SQL Server as the database tool. In our experiments, three types of databases are built for performance evaluation. They are the plain database (no encryption at all), the mixed database (some attributes are encrypted), and the encrypted database (the whole tuple is encrypted). A number of queries is generated and submitted to the database server. Some queries may access the encrypted attributes and others may not. The experimental results show that the encryption in the mixed database slightly degrades the database performance, but not as much as in the encrypted database except when the number of queries is very high, and we doubt that our system may be over the limit at that point. In summary, the performance results ensure that the cost of encryption is fair and encrypting particular sensitive data in the database would be worth in the real implementation.

6.2 Summary of Our Contribution

The contribution of the proposed work is summarized as follows.

1. Our work shows that implementing an encrypted database does not significantly degrade the performance of the database. In addition, having encrypted sensitive data in the database would prevent an unauthorized access and the database is secure on a certain degree.

Copyright by Mahidol University

2. Our work demonstrates the implementation of the role-based access control policy to the database. The policy is defined according to the hierarchical structure of the organization. We use two related tables: authority table and key table. The authority table defines the access rights of a specific role while the key table defines users' individual key.
3. Our work gives the flexibility in performing encryption granularity at the column level. Thus, individual application can independently select which attributes to be encrypted.
4. Our experimental results give the better query response time than that of the whole encrypted database due to the fact that only sensitive attributes are encrypted. As a result, querying non-sensitive data does not have to go thru the encryption and decryption processes.

6.3 Suggestions for Future Work

The proposed work could be extended in several directions such as detailed analysis of encryption performance, the issues of key management and data privacy issues. The list of future work is suggested below.

1. Our model focuses on selection query only. The performance of our model should be extended to include other types of queries such as range query, and join operations.
2. Additional database parameters such as the number of tables in the database, the number of encrypted attributes, and having specific application workload should be studied further.
3. Implementing this proposed model to a real world application to see whether other security issues such as privacy and inference exposure are needed to be addressed in the future.

Copyright by Mahidol University

REFERENCES

1. Damiani, E., De Capitani diVimercati, S., Jajodia, S., Paraboschi, S. and Samarati, P. *Balancing Confidentiality and Efficiency in Untrusted Relational DBMSs*. CCS'03, Washington, 27–31, 2003.
2. The Economic Impact of Role Based Access Control. Research Triangle Institute. NIST Planning Report 02-01. 2002
3. Elmasri, R., and Navathe, B. S. Foundamentals Of Database Systems. Addison-Wesley., Third edition, 2000.
4. Elovici, Y., Gudes, E., Shmueli, E., Waisenberg, R. A Structure Preserving Database Encryption Scheme. SDM'04, Toronto, 2004.
5. Ferraiolo, D.F., Barkley , J.,and Kuhn, D.R. A Role Based Access Control Model and Reference Implementation within a Corporate Intranet. ACM Transactions on Information Systems Security, Volume 1, Number 2, February 1999.
6. Ferraiolo, D. F., Sandhu, R., Gavrila S., Kuhn, D. R. and R. Chandramouli. Proposed NIST Standard for Role-Based Access Control. ACM Transactions on Information and Systems Security (TISSEC), Volume 4, Number 3, 2001.
7. Hacigümüs, H., Iyer, B., Li, C., and Mehrotra, S. Executing SQL over encrypted data in the database-service-provider model. In Proc. of the ACM SIGMOD'2002, Madison, Wisconsin, USA, 2002.
8. Hacigümüs, H., and Mehrotra, S. Performance-Conscious Key Management In Encrypted Databases., IFIP WG 11.3 Working Conference on Data and Application Security, Sitges, Spain , 95-104, 2004.
9. Hacigümüs, H., and Mehrotra, S. Efficient Key Updates in Encrypted Database Systems. Secure Data Management, 1-15, 2005.
10. He, J., Wang, M. Cryptography and Relational Database Management Systems. Proceedings of the 5th International Database Engineering and Applications Symposium, 273-284, 2001.

Copyright by Mahidol University

11. Hore, B., Mehrotra, S., Tsudik, G. A Privacy Preserving Index for Range Queries. In Proc. Of the 30th VLDB Conference, Toronto, Canada, 2004.
12. Iyer, B., Mehrontra, S., Mykletun, E., Tsudik, G., Wu, Y. A Framework for Efficient Storage Security in RDBMS. Springer-Verlag Berlin Heidelberg, 147-164, 2004.
13. Kantarcioglu, M., and Clifton, C. Security Issues in Querying Encrypted Data. National Science Foundation.
14. Li, Mengke. Role-Based Access Secure System. Project Report of CSE877, Fall 2002.
15. Ozsoyoglu, G., Singer, D., Chung, S. Anti-Tamper Database: Querying Encrypted Databases. In the process of Book, Kluwer Academic Publishers, Prague, Boston, 2004.
16. Radha, V., and Kumar, H. EISA – An Enterprise Application Security Solution for Databases. Springer-Verlag Berlin Heidelberg, 164-176, 2005.
17. Schaad, A., Moffett, J., and Jacob, J. The RoleBased Access Control System of a European Bank: A Case Study and Discussion. in Proc. of the 6th ACM Symposium on Access Control Models and Technologies, LittonTASC, Chantilly, Virginia, USA, May, 3–9, 2001.
18. Schneier, B. Applied Cryptography: Protocols, Algorithms, and Source Code in C. John Wiley & Sons, Inc., Second edition, 1996.
19. Symantic Encryption http://www.arl.org/access/naildd/ipig/res/encrypt/sld001.htm
20. Security Tips: Encryption made simple http://searchsecurity.techtarget.com/tip/1,289483,sid14_gci526718,00.html
21. NIST: Role Based Access Control http://csrc.nist.gov/rbac/

Copyright by Mahidol University

BIOGRAPHY

NAME	Miss Nongnuch Poolsawad
DATE OF BIRTH	5 November 1978
PLACE OF BIRTH	Bangkok, Thailand
INSTITUTIONS ATTENDED	The University of the Thai Chamber of Commerce, 2000 : Bachelor of Science (Computer Science) Mahidol University, 2006 : Master of Science (Computer Science)
POSITION&OFFICE	National Metal and Materials Technology Center, National Science and Technology Development Agency, Ministry of Science and Technology, 114 Thailand Science Park, Paholyothin Rd., Klong 1, Klong Luang, Pathumthani 12120 Thailand Position : Computer System Analyst E-mail: nongnucp@mtec.or.th
HOME ADDRESS	12/754 Rangsit-Nakhonnayok Rd., Bungyeetho, Thanyaburi, Pathumthani 12130 Thailand

Copyright by Mahidol University