Scientific Journal Impact Factor (SJIF): 1.711

e-ISSN: 2349-9745
p-ISSN: 2393-8161

International Journal of Modern Trends in Engineering and Research
www.ijmter.com

Concurrent And Independent Access To Encrypted Cloud Databases

Rashmi A J Sheikh[1]
[1]*Computer Department, SRES College of Engineering, Kopargaon,*
rashmisheikh7@gmail.com

Abstract— Since data in cloud will be placed anywhere, because of the critical nature of the applications, it is important that clouds be secure. The major security challenge with clouds is that the owner of the data may not have control of where the data is placed. This is because if one wants to exploit the benefits of using cloud computing. This requirement imposes clear data management choices: original plain data must be accessible only by trusted parties that do not include cloud providers, intermediaries, and Internet; in any untrusted context, data must be encrypted. Satisfying these goals has different levels of complexity depending on the type of cloud service.

We propose SecureDBaaS as the first solution that allows cloud tenants to take full advantage of DBaaS qualities, such as availability, reliability, and elastic scalability, without exposing unencrypted data to the cloud provider. The architecture design was motivated by goal: to allow multiple, independent, and geographically distributed clients to execute concurrent operations on encrypted data, including SQL statements that modify the database structure.

Keywords- Cloud, security, confidentiality, SecureDBaaS, database.

I. INTRODUCTION

The Aim of our system is, to integrate cloud database services with data confidentiality and the possibility of executing concurrent operations on encrypted data. We use cloud for uploading owner's data. Data Owner who has uploaded his data on cloud he is not ensure about his data, so we have to store his data on the cloud by encrypting his data. This encryption of data takes place at client side and metadata of that data also created i.e. secureDBaaS concept. This encrypted data is stored at the cloud along with its encrypted metadata. Then the authorized clients can access the data by using only metadata.

This is the first solution supporting geographically distributed clients to connect directly to an encrypted cloud database, and to execute concurrent and independent operations including those modifying the database structure. The proposed system has the further advantage of eliminating intermediate proxies that limit the elasticity, availability, and scalability properties that are intrinsic in cloud-based solutions. SecureDBaaS provides several original features that differentiate it from previous work in the field of security for remote database services.

II. SYSTEM OVERVIEW

The system mainly focuses on following-

- Cloud database
- Metadata Management
- Encryption algorithm

Cloud database: We assume that tenant data are saved in a relational database. We have to preserve the confidentiality of the stored data and even of the database structure because table and column names may yield information about saved data. We distinguish the strategies for encrypting the database structures and the tenant data.

Metadata Management: Metadata generated by SecureDBaaS contain all the information that is necessary to manage SQL statements over the encrypted database in a way transparent to the user. Metadata management strategies represent an original idea because SecureDBaaS is the first architecture storing all metadata in the untrusted cloud database together with the encrypted tenant data.

Encryption algorithm: Choosing the encryption algorithms used to encrypt and decrypt all the data stored in the database table.

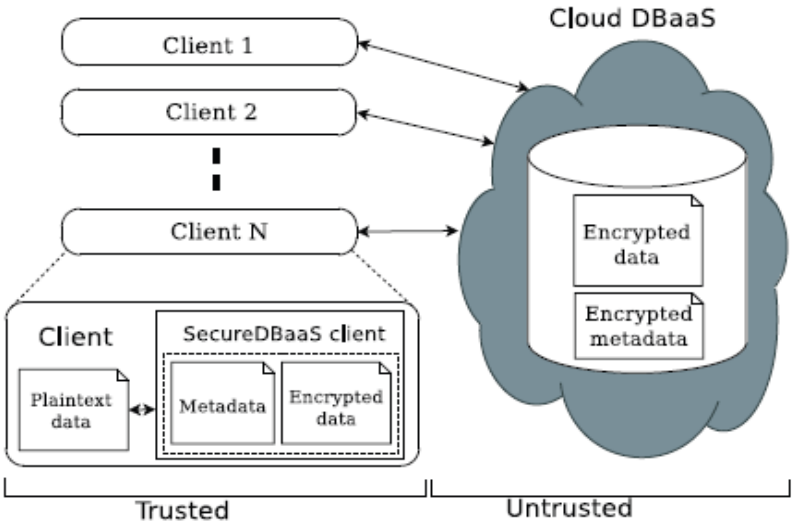

Fig. 1. SecureDBaaS architecture.

Fig. 1 describes the overall architecture. We assume that a tenant organization acquires a cloud database service from an untrusted DBaaS provider. The tenant then deploys one or more machines (Client 1 through N) and installs a SecureDBaaS client on each of them. This client allows a user to connect to the cloud DBaaS to administer it, to read and write data, and even to create and modify the database tables after creation. SecureDBaaS is designed to allow multiple and independent clients to connect directly to the untrusted cloud DBaaS without any intermediate server

III. SYSTEM DESIGN

3.1 Cloud database: We assume that tenant data are saved in a relational database. We have to preserve the confidentiality of the stored data and even of the database structure because table and column names may yield information about saved data. We distinguish the strategies for encrypting the database structures and the tenant data.

3.2 Metadata Management: Metadata generated by SecureDBaaS contain all the information that is necessary to manage SQL statements over the encrypted database in a way transparent to the user. Metadata management strategies represent an original idea because SecureDBaaS is the first architecture storing all metadata in the untrusted cloud database together with the encrypted tenant data.

3.3 Encryption algorithm: Choosing the encryption algorithms used to encrypt and decrypt all the data stored in the database table.

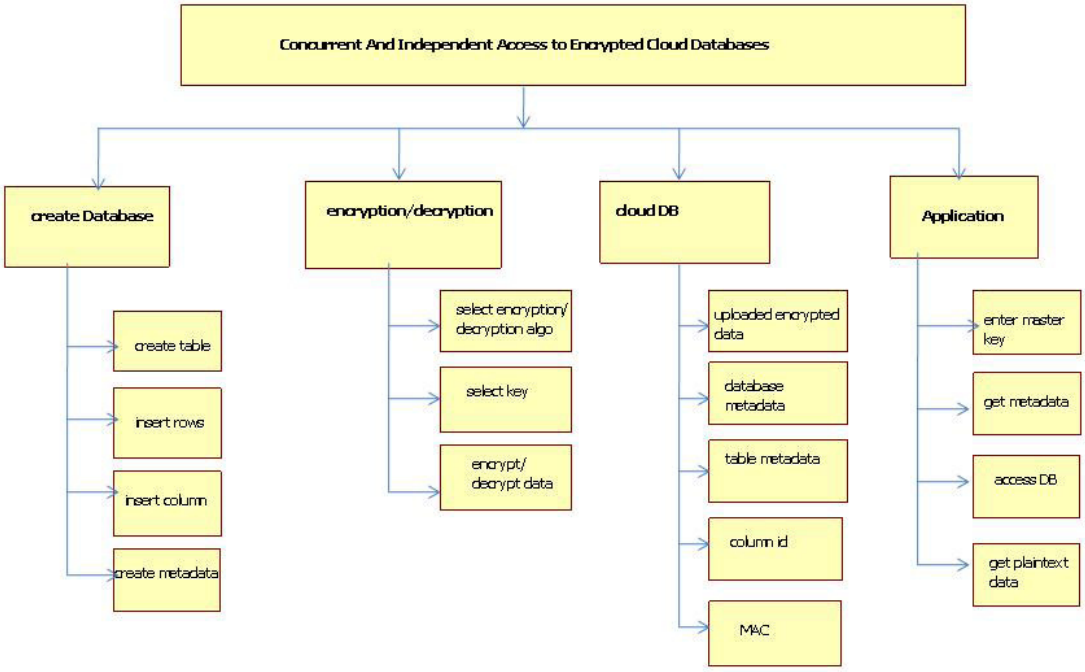

Fig.2. System Design

Fig 2 describes the system design having modules and its components.

1. Creation of database-

In this module client creates its database and store data in the form or columns and rows. After creation of Database the client also creates its metadata which will help for later communication instead of whole database.

2. Selection of encryption and decryption algorithm -

In this module we select the encryption algorithm to encrypt and decrypt the created database and its metadata. It will provide security to whole data of client which is to be uploaded on the cloud.

3. Cloud Database-

Cloud Database is the service provider, which provides services to the tenants. All the encrypted data from data owner is uploaded on cloud which provides concurrent access to cloud DB to the geographically deployed clients. Cloud DB contains encrypted database and its encrypted metadata.

4. Application-

This module contains the application of system to the cloud. How we will
Apply these all on cloud this module explains it. We use master key to access cloud data after data is uploaded on data. First we will get encrypted data if our key is correct then by using random decryption keys we will get the final output in the form of plaintext data. Input is taken from user in the form of sql querry. Firstly client will create Database then, will enter rows into the database. After that the metadata of database is created. Then selected encryption algorithm is applied to the database and its metadata. final output gives the encrypted data with all its information and key used.

IV.IMPLEMENTATION

4.1 Data Management:

Cloud database acts as service provider for tenants. The cloud is created first for the system. All information or data store in the relational database. So for creating tables and column we have to access it with SQL query only.

4.2 Metadata Management:

Metadata generated by SecureDBaaS contain all the information that is necessary to manage SQL statements over the encrypted database in a way transparent to the user. Metadata management strategies represent an original idea because SecureDBaaS is the first architecture storing all metadata in the untrusted cloud database together with the encrypted tenant data.

SecureDBaaS uses two types of metadata.

- Database metadata are related to the whole database. There is only one instance of this metadata type for each database.
- Table metadata are associated with one secure table. Each table metadata contains all information that is necessary to encrypt and decrypt data of the associated secure table.

This design choice makes it possible to identify which metadata type is required to execute any SQL statement so that a SecureDBaaS client needs to fetch only the metadata related to the secure table/s that is/are involved in the SQL statement.

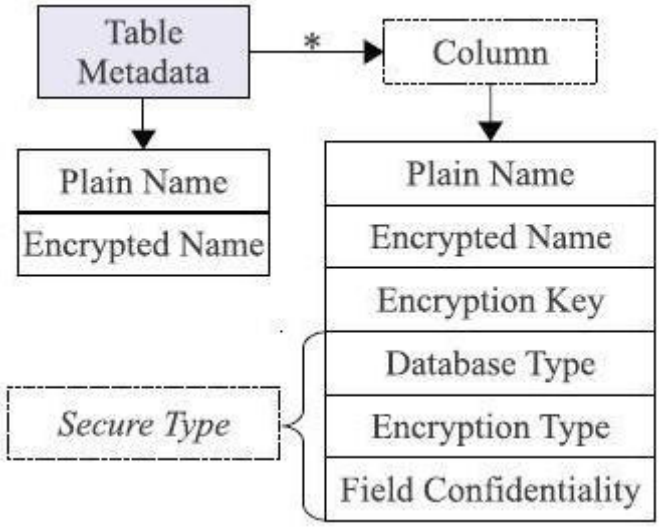

Fig.3. Structure of table metadata.

This design choice minimizes the amount of metadata that each SecureDBaaS client has to fetch from the untrusted cloud database, thus reducing bandwidth consumption and processing time. Moreover, it allows multiple clients to access independently metadata related to different secure tables. Database metadata contain the encryption keys that are used for the secure types. A different encryption key is associated with all the possible combinations of data type and encryption type. Hence, the database metadata represent a key ring and do not contain any information about tenant data.

The structure of a table metadata is represented in Fig. 3. Table metadata contain the name of the related secure table and the unencrypted name of the related plaintext table. Moreover, table metadata

include column metadata for each column of the related secure table. Each column metadata contain the following information.

- Plain name: the name of the corresponding column of the plaintext table.
- Coded name: the name of the column of the secure table. This is the only information that links a column to the corresponding plaintext column because column names of secure tables are randomly generated.
- Secure type: the secure type of the column. This allows a SecureDBaaS client to be informed about the data type and the encryption policies associated with a column.
- Encryption key: the key used to encrypt and decrypt all the data stored in the column.

SecureDBaaS stores metadata in the metadata storage table that is located in the untrusted cloud as the database. This is an original choice that augmnts flexibility, but opens two novel issues in terms of efficient data retrieval and data confidentiality. To allow SecureDBaaS clients to manipulate metadata through SQL statements, we save database and table metadata in a tabular form. Even metadata confidentiality is guaranteed through encryption. The structure of the metadata storage table is shown in Fig. 4 This table uses one row for the database metadata, and one row for each table metadata.

Database and table metadata are encrypted through the same encryption key before being saved. This encryption key is called a master key. Only trusted clients that already know the master key can decrypt the metadata and acquire information that is necessary to encrypt and decrypt tenant data. Each metadata can be retrieved by clients through an associated ID, which is the primary key of the metadata storage table. This ID is computed by applying a Message Authentication Code (MAC) function to the name of the object (database or table) described by the corresponding row. The use of a deterministic MAC function allows clients to retrieve the metadata of a given table by knowing its plaintext name. This mechanism has the further benefit of allowing clients to access each metadata independently, which is an important feature in concurrent environments. In addition, SecureDBaaS clients can use caching policies to reduce the bandwidth overhead.

Metadata Storage Table

ID	*Encrypted Metadata*	*Control Structure*
MAC('.'+Db)	Enc(Db metadata)	MAC(Db metadata)
MAC(T1)	Enc(T1 metadata)	MAC(T1 metadata)
MAC(T2)	Enc(T2 metadata)	MAC(T2 metadata)

Fig.4. Organization of database metadata and table metadata in the metadata storage table.

4.3 Algorithms:

Encryption algorithms are applied to encrypt the database. There are various encryption algorithms symmetric and asymmetric, but we will apply symmetric algorithm which proved key distribution only once to all tenants there will be no different private key related to every user.

V. CONCLUSION

In this paper, we have discussed concurrent and independent access to encrypted cloud databases, proposes an innovative architecture that guarantees confidentiality of data stored in public cloud databases. The proposed system will not require modifications to the cloud database, and it will be immediately

applicable to existing cloudDBaaS. Resolve problem of single point failure and a bottleneck limiting availability and scalability of cloud database services.

REFERENCES

[1] Luca Ferretti, Michele Colajanni, and Mirco Marchetti, "Distributed, Concurrent, and Independent Access to Encrypted Cloud Databases", IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 25, NO. 2, FEBRUARY 2014.

[2] Kevin Hamlen, Murat Kantarcioglu, Latifur Khan, Bhavani Thuraisingham, "Security Issues for Cloud Computing", International Journal of Information Security and Privacy, 4(2), 39-51, April-June 2010.

[3] Auditor Bhavna Makhija,VinitKumar Gupta,Indrajit Rajput, "Enhanced Data Security in Cloud Computing with Third Party", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 2, February 2013.

[4] M. Armbrust et al., "A View of Cloud Computing", Comm. of the ACM, vol. 53, no. 4, pp. 50-58, 2010.

[5] L. Ferretti, M. Colajanni, and M. Marchetti, "Supporting Security and Consistency for Cloud Database", Proc. Fourth Intl Symp. Cyberspace Safety and Security, Dec. 2012.

[6] H. Hacigumu s, B. Iyer, and S. Mehrotra, Providing Database as a Service, Proc. 18th IEEE Intl Conf. Data Eng., Feb. 2002.

[7] C. Gentry, Fully Homomorphic Encryption Using Ideal Lattices, Proc.41st Ann. ACM Symp. Theory of Computing May 2009.

