SQL Aware S-DBaaS Model for Separated and Parallel Access to Encrypted Cloud Database Using Froward Security

[1]Cholke Satish .C, [2]Prof. S. B. Natikar

[1]PG Student, [2]Asst. Professor, [1,2]Dept. of Computer Engg. Vishwabharati Academy College of Engineering, Ahmednagar, Maharashtra, India

[1]*cholkesatishchangdeo@gmail.com*, [2]*sbnatikar5@gmail.com*

Abstract : The concept of "cloud" is not new for us. We have been used cloud computing from many years in one or other form. Cloud computing is a way of using the computing resources that are available and accessing over the network. Cloud storage is used to store large amount of data as in the form of pay-per-use. We are moving to future where some or even most of our data will exist in cloud, because cloud provides stored data to be accessed at anywhere through networks. 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; it will create other problems also like data confidentiality and bottleneck. This paper presents a new technique that provides data confidentiality and concurrent access of encrypted cloud database with resolving the bottleneck problem.

Keywords: S-DBaaS, Forward Security, Confidentiality, Cloud Storage.

I. INTRODUCTION

The goal of this technology to stored data on cloud in an encrypted form and only authorized person can access data with the help of key which is also called a master key and the possibility of executing concurrent operations on encrypted data. We store the data of owner on cloud. Data Owner is not ensure about his data, so we store his data on cloud by encrypting data. This encryption of data takes place at client side and as secure DBaaS concept, metadata of that data also created. This encrypted data is stored at the cloud along with its encrypted metadata. Privileged user and multifactor access control, data classification and discovery, transparent data encryption, secure configuration management, and data masking is the key of security. In Cloud Databases customers can deploy reliable data security solutions that require no changes to existing applications, it saves time and money. Cloud Databases provide powerful preventive and detective security controls include database activity monitoring and blocking. This project proposes SecureDBaaS. Here all databases are encrypted and stored in the cloud. It allows multiple and only authorized users can access their own databases concurrently and alone. Each user use a privet key to encrypt a data and same key is use to decrypt data, So by that it make more secure user data on cloud. There is RSA algorithm plays an important role because to encrypt and decrypt plan text (i.e. user data) by using RSA algorithm and information square measure encrypted exploitation AES technique so overhead on the network will be reduced. SecureDBaaS discard any type of intermediate proxy server, so a user can achieve availability, scalability and elasticity of DBaaS. SecureDBaaS maintain the concurrency as well as confidentiality. Database-as-a-service (DBaaS) is very impressive because of two reasons. First, due to it the cost (i.e. economical, energy) incurred by users are much lower when they are paying for a share of a service compare to

running everything themselves. Second, the costs for both software licensing and administrative costs of a well-designed DBaaS will be proportional to actual usage. DBaaS can largely reduce operational costs and perform well.

II. LITRATURE SURVEY

This Paper proposes the work that has done the adequate literature survey, analysis and comparison of various journals, conference papers with real work as follows.

We have studied the paper published by authors 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 2015. From this paper we have learn the technique of Fine grain encryption policies[1] and its types in order to encrypt the plain text data tables with separate encryption keys[1]. We learn the idea of encrypting each columns of clients data i.e plaintext data tables with randomly generated separate secrete keys[1]. We also taken the idea of maintaining the secure meta data table which includes all the information's of plain text data tables and encrypted data tables, means we storing all the attributes of data tables in secure metadata table, like plain text table name, secure table name, all keys which used for encryption of columns etc. Then we generates one unique master key to encrypt the secure plain text meta data table[1], then that master key will be sent to the DBA.

Then we have taken the idea of forward security[2], which is responsible for changing the unique master key with specific time span[2]. We studied the forward security techniques from the paper published by the authors Xinyi Huang, Joseph K. Liu, Shaohua Tang, Yang Xiang, Kaitai Liang, Li Xu, Jianying Zhou, "Cost-Effective Authentic and Anonymous Data Sharing with Forward Security",IEEE transactions on computers vol: 64 no: 6 year 2015[2].

In order to access the encrypted cloud data base separately as well as concurrently and distributed. We have studied the idea of maintaining the policies of access control matrix[3], which will be maintaining by separate trusted entity named as DBA[3]. The idea of access control matrix are published by authors Luca Ferretti, Fabio Pierazzi, Michele Colajanni, and Mirco Marchetti, "Scalable Architecture for Multi-User Encrypted SQL Operations on Cloud Database Services", IEEE transactions on cloud computing, vol. 2, no. 4, october-december 2014 given us the unique solution for maintaining access control matrix policies[3].

III. ARCHITECTURAL DESIGN OF SYSTEM

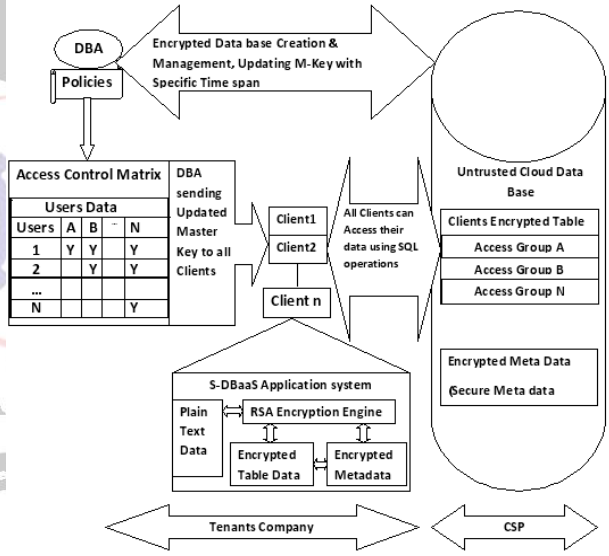

Fig. 1 Proposed System Architecture.

SecureDBaaS[1] is designed to allow multiple and independent clients to connect directly to the untrusted cloud DBaaS without any intermediate server. Figure 1 describes the overall architecture. We assume that a tenant organization acquires a cloud database service from an untrusted DBaaS provider[1]. 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.

A Secure DBaaS allow multiple and independent clients to connect directly to untrusted cloud. Assume an organization obtain database as a service from untrusted cloud provider. A secure DBaaS manage database related information, such as encrypted database and encrypted metadata. An encryption of database in cloud database prevents the violation of confidentiality by untrusted cloud provider.

Secure DBaaS stores a metadata in cloud and allow Secure DBaaS client to retrieve necessary metadata, which is required to extract data from cloud database. Assume that data stored in cloud database is relational database[1]. Encrypted data is stored through secure table in cloud database. Encryption operation is done at Secure DBaas client.

IV. IMPLEMENTATION STRATEGY

This S-DBaas System is divided into various modules as follows:

A. Module 1: Metadata Storage Table

This module of SecureDBaaS generates a metadata which include all the information need to access the data from encrypted database[1]. Secure DBaaS stores metadata in metadata storage table that is placed in cloud database[4]. This is flexible approach but come with two issues efficiency of data access and confidentiality[3]. To provide efficiency of data access Secure DBaaS use two metadata.

B. Module 2: Database Metadata

This module generally responsible to maintain the database Meta data[1]. This metadata associated to entire database. This metadata has a only one example for each database in a cloud.

C. Module 3: Table metadata

This is module related with secure table. That is this meta storage table include all the information about encryption and decryption of secure table. Database and table metadata are encrypted by using the same encryption key before it has been stored at cloud database. This encryption key is called a master key. Only trusted clients know this key. If you want to decrypt the metadata, it requires that same key (i.e. master key at cloud database. Each client can recover metadata through an associated ID[1].The ID is work as primary key of metadata table. Through this mechanism each clients are allowed to access metadata independently, which is an important feature in concurrent environments. In addition, SecureDBaaS clients can use caching policies to reduce the bandwidth overhead.

D. Module 4: Secure DBaaS Client

Suppose that a connection is a resident of cloud and gets cloud database administration from an untrusted DBaaS cloud administration supplier. The resident then arranges one or more machines and introduces a SecureDBaaS customer on each of them. Suppose this customer is a client to extract with the cloud database to get an administration. The data oversaw by SecureDBaaS customer incorporates plaintext information, encoded information, metadata, and scrambled metadata. Plaintext information is the data about information place absent and handle remotely in cloud database. A confined DBaaS customer encoded the information before it has been place absent in cloud remotely. It delivers an arrangement of metadata that include data needed to encode and decoded the information. After store of information it will repair the metadata also in cloud metadata stockpiling table. It recovers the grateful metadata from metadata stockpiling table to get to the cloud database.

E. Module 5: Setup Phase

So in this module we explain that how to initialize a secure DBaaS architecture from a cloud database service acquired by a tenant from a cloud provider. Here suppose that the DBA creates the metadata storage table that at the beginning contains immediately the database metadata, and not the table metadata.

F. Module 6: Sequential SQL Operations

The first connection between the client and the cloud DBaaS is for authentication purposes. Secure DBaaS relies on ordinary authentication and authorization mechanisms from the original DBMS server. After the authentication, a user interacts with the cloud database through the Secure DBaaS client.

G. Module7:Concurrent SQL Operations

The support to parallel execution of SQL statements issued by multiple independent (and possibly geographically distributed) clients is one of the most important benefits of Secure DBaaS with respect to state-of-the-art solutions.

V. ALGORITHMS

A. Fine Grain Encryption Approach Using RSA

1. S-DBaaS client will select any large prime numbers, suppose **P** and **Q**.
2. Then calculating **n = P x Q**.
3. Then S-DBaaS will take the encryption keys (Public keys) i.e.**E1,E2,E3**,…..En according to the requirement such that it must not be the factors of (**P-1**) and (**Q-1**).
4. **Encryption of PTN**

   Encryption for the plain text table names with the same key. S-DBaaS will take the Plain Text Table Name (PTN) for the encryption with key E and producing cipher text table name i.e. secure table name (STN) as follows.

   $$STN = PTN^{E} \bmod n.$$

   Same encryption key with same procedure will be applied for encryption of all plain text table names.

5. **Encryption of PTCN**

   For the encryption of plain text table column name (PTCN) separate encryption keys will be used for each column name. Encryption keys are randomly generated by S-DBaaS client.

   $$STCNo…STCNn = [PTCNo…PTCNn]^{E0….En} \bmod n$$

6. **Encryption of metadata (database and table metadata)**

   In this step same encryption key will be used for encryption of both i.e. database metadata as well as table metadata, such that:

a) Encrypt database metadata which contains encryption keys that are used for a secure type which having the field confidentiality set to database.

b) The database metadata and table metadata will be encrypted by same encryption key known as Master Key (M) and after the encryption of metadata the master key will be same to all trusted S-DBaaS clients and maintain Secure Metadata Storage Table (SMST).

c) Then by taking MAC code which will be derived from the name of database and tables by using MAC function and assign the unique ID code (Primary Key) to all the rows of maintain Secure Metadata Storage Table (SMST).

7. **Decryption Process**

   All trusted S-DBaaS Clients have a Master key for decryption SMST. S-DBaaS client can decrypt the Tenants data as follows:

a. S-DBaaS client will select separate decryption key (D) for the decryption of Secure Metadata Storage Table (SMST). Such that following equation becomes true:

$$[D \times E] \bmod [P - 1] \times [Q - 1] = 1$$

b. The S-DBaaS Clients machine will be decrypt the SMST table as follows:

Plain Text Metadata Table = $SMST^{D}$ mod n.

8. Then the Plain Text Metadata Table has all the information like, Encryption keys of columns as well as secure tables, secure types with field confidentiality.

9. With the help of information contains in decrypted SMST Table the S-DBaaS Clients can easily decrypt the tenants data concurrently or independently by repeating steps 6.

VI. COMPARISON OF EXISTING AND PROPOSED SYSTEM

Existing system architecture can store only clients data in cloud databases and it just save metadata information in the client machine or it used to split a metadata between trusted proxy server and the cloud database[1] but this existing system scenarios are quite inefficient for simultaneous access to the same database by multiple clients. So, the existing system totally based on trusted proxy servers (e.g., [1], [4], [7]), that are more feasible but the proxy introduce a system bottleneck. Hence it reduces availability, elasticity and scalability of cloud database [1]. The S-DBaaS Model proposes a different approach where all data and metadata are stored in cloud database and eliminating all intermediate proxy servers [1]. The S-DBaaS can retrieve required metadata from the entrusted databases by using SQL statement in order to the S-DBaaS client can access tothe entrusted cloud database independently with guarantee of availability and scalability[2].

VII. EXPERIMENT ANALYSIS AND RESULTS

In the experimental result analysis we test and analyze the SQL query response time For various SQL operation such as update, insert etc. while performing the update as well as insert query operations the proposed S-DBaaS systems response time is very less than other existing system. In figure 7.1 we have shown the Comparison for query response time among existing vs. proposed system.

Table 1: Performance measure for SQL query response time with respect to update and insert operations.

Query Type	Response Time Plain	Response Time by Existing system	Response Time by Proposed system
update	0.8	1.7	1.3
Insert	0.9	2.8	1.9

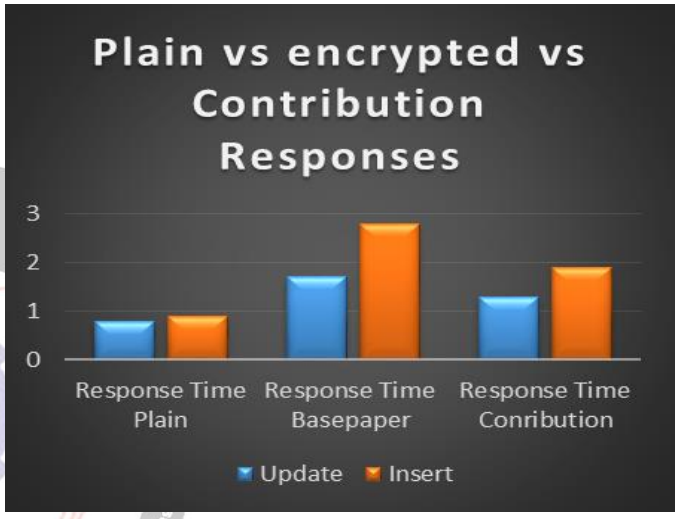

Fig. 2 Comparison for query response time among existing vs. proposed system.

VIII. CONCLUSION

We proposed new unique system architecture with having a guarantee of scalability, security and data confidentiality of clients data placed in public cloud database. Unlike existing system our system does not depends on an intermediate proxy servers that as consider as a single point of failure and also responsible for a bottleneck conditions which leads to limiting availability and scalability of cloud database services. The important part of our system is that it designed to allow any separate client or geographically scattered multiple clients to access the data from encrypted cloud databases.

IX. ACKNOWLEDGMENT

I wish to express my sincere gratitude to the Head of Department Prof. Dattatray Zine of M.E. Computer Engineering Department for providing me an opportunity for presenting a project on the topic "Separated and Parallel access to Encrypted cloud databases using S-DBaaS Model". I sincerely thank to my project guide Prof. S .B. Natikar and our P.G Coordinator Prof. Dr. Vrushali Ranmalkar for their guidance and encouragement in the partial stage completion of my project work. I also wish to express my gratitude to the officials and other staff members who rendered their help during the period of my project work. Last but not least I wish to avail myself of this opportunity, to express a sense of gratitude and love to my friends and my parents for their manual support, strength, and help and for everything.

REFERENCES

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

[2] Xinyi Huang, Joseph k Liu,Shaohua Tang, Yang Xiang. *Cost-Effective Authentic and Anonymous Data Sharing with Forward Security*., IEEE transactions on computers, vol. 64, no. 6. Year 2015.

[3] Ferretti, Fabio Pierazzi, Michele Colajanni, and Mirco Marchetti, "*Scalable Architecture for Multi-User Encrypted SQL Operations on Cloud Database Services*", IEEE transactions on cloud computing, vol. 2, no. 4, october-december 2014

[4] P. Mahajan, S. Setty, S. Lee, A. Clement, L. Alvisi, M. Dahlin, and M. Walfish, "*Depot: Cloud Storage with Minimal Trust*," ACM Trans. Computer Systems, vol. 29, no. 4, article 12, 2011.

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

[6] E. Mykletun and G. Tsudik, "Aggregation Queries in the Database-as-a-Service Model," Proc. 20th Ann. IFIP WG 11.3 Working Conf. Data and Applications Security, July/Aug. 2006.

[7] D. Agrawal, A.E. Abbadi, F. Emekci, and A. Metwally, "*Database Management as a Service: Challenges and Opportunities*," Proc. 25th IEEE Int'l Conf. Data Eng., Mar.-Apr. 2009.

[8] "*Oracle Advanced Security," Oracle Corporation*, http://www.oracle.com/technetwork/database/options/ advanced security,Apr. 2013.

[9] L. Ferretti, M. Colajanni, and M. Marchetti, "*Supporting Security and Consistency for Cloud Database*," Proc. Fourth Int'l Symp. Cyberspace Safety and Security, Dec. 2012.

[10] "*Transaction Processing Performance Council*," TPC-C, http:// www.tpc.org, Apr. 2013.

[11] H. Berenson, P. Bernstein, J. Gray, J. Melton, E. O'Neil, and P. O'Neil, "*A Critique of Ansi Sql Isolation Levels*," Proc. ACM SIGMOD, June 1995.

[12] "*Xeround: The Cloud Database*," Xeround, http://xeround.com, Apr. 2013.

