Secure Mute-DB Architecture for Separated and Parallel Access to Encrypted Cloud Database Using SQL Operations

[1]Gauri G. Bhagwat, [2]Pooja N. Bapte, [3]Swapnili K. Dahiwalkar, [4]Anjali S. Palshikar, [5]Prof. S. Y. Raut

[1,2,3,4]UG Student, [5]HOD, [1,2,3,4,5]Department of I.T Engineering, PREC, Loni, Savitribai Phule Pune University, Maharashtra, India.

[1]gaubhagwat7@gmail.com, [2]poojabapte9gmail.com, [3]swapnali29.sd@gmail.com, [4]palshikaranjali@gmail.com, [5]it.hod@pravraengg.org.in

Abstract: Cloud computing refers to the delivery of computing resources over the web rather than keeping information on your own drive or change applications for your desires, you use a service over the internet at other location to store your text or use its applications [1]. Doing thus could give rise to sure privacy implications. In a cloud paradigm, wherever important information is placed in infrastructures of untrusted third parties [1], guaranteeing information confidentiality is of overriding importance. This requirement imposes clear information management choices original plain information should be accessible solely by trusty parties that don't embrace cloud providers, intermediates and Internet in any untrusted context, information should be encrypted. Satisfying the goals has completely different levels of complexity looking on the kind of cloud service. There are many solutions for the storage as a service area, whereas guaranteeing confidentiality within the information as a service (DBaaS) [3] is still open research area remains associate open analysis space. During this context, we propose SecureDBaaS because the resolution that permits cloud tenants to take full advantage of DBaaS qualities such as availability, security, reliability, responsibility and elasticity, measurability, without exposing unencrypted information to the cloud provider. It is a unique S-MuteDB Architecture that is made up of various cloud database services with data confidentiality parameter, and feasible to executing parallel as well as separated operations on encrypted cloud databases with forward security secret data sharing scheme which having changing master secret key with specific time span. This is unique solution that can be able to directly connect geographically scattered users to an encrypted cloud databases and to allow performing independent and parallel SQL operations with modifying and updating database system structure. In order to achieve concurrent as well as distributed access to encrypted cloud database we proposes the Secure MuteDB system with access control matrix which maintained by one separate entity known as DBA. In the S-MuteDB system the responsibilities of Database Administrator is to monitor system operations with maintaining and updating Master key according to the Access control matrix policies.

Keywords: S-MuteDB, Cloud Storage, Forward Security, Confidentiality, DBA, SQL queries.

I.INTRODUCTION

The primary aim of this S-MuteDB Architecture is to allow more security to complex and risky data storage on the database of the untrusted cloud service provider that the third parties or intermediate severs or proxies may manage easily and that leads to data leakages. Even the untrusted cloud service providers may have curiosity about the clients data they can perform passive attacks on clients data, so for this problem we proposes one unique solution by using S-MuteDB architecture that allows the multiple as well as independent clients to perform various SQL operations parallel or separately on encrypted cloud data bases with forward security secret data sharing scheme which having changing master secret key with specific time span. 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-MuteDB 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 Secure-MuteDB. 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. SecureMuteDB discard any type of intermediate proxy server, so a user can achieve availability, scalability and elasticity of DBaaS. Secure-MuteDB maintains 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. LITERATURE 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 column of client's 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 generate 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 distributedly 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 solusion for maintaining access control matrix policies[3]. Authors: P. Mahajan, S. Setty, S. Lee, A. Clement, L. Alvisi, M. Dahlin, and M. Walfish [4]: The have discussed the various methods of use of encrypted untrusted cloud storage. Authors: C. Gentry [5]:

Hi has discussed about fully homomorphic (fine grain) encryption technique which can be useful for the encryption of secure table contents. Authors: Jin Li, Xiaofeng Chen, Xinyi Huang, Shaohua Tang and Yang Xiang and Mohammad Mehedi Hassan and Abdulhameed Alelaiwi [6]: The have discussed the various techniques of deduplication techniques which can be useful to wastage of cloud memory space. Authors: E. Mykletun and G. Tsudik [7]: They have given the idea of use of Data base as a service in cloud computing, so we have learn the idea of how to use Database as a service with encrypted cloud database using SQL operations.

III. STUDY OF EXISTING SYSTEMS

In existing system there are more untrusted intermediate systems are used like intermediate proxies or servers [1], that may leads to data leakages And highly responsible for bottle neck conditions [1]. In this context there is possibility that Cloud service Providers can curious about users data [11], and due to the coarse grain encryption it is possible to perform a Passive attacks on tenant's data.

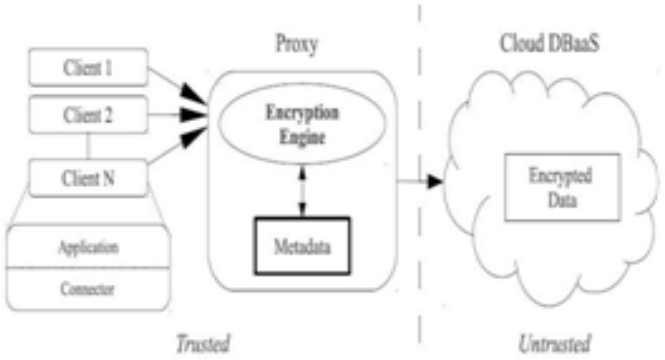

Fig. 1 A proxy Based Architecture of Existing System.

IV. PROPOSED S-MuteDB SYSTEM

We propose the idea to develop an efficient approach to store the users or clouds tenant's data to untrusted cloud database in encrypted form that can be access by geographically scattered users by executing various common SQL operations on encrypted data independently and concurrently.

By removing intermediate trusted proxy servers we try to achieve various security parameters like data availability, scalability and confidentiality of encrypted data.

A. Architectural Design of S-MuteDB System

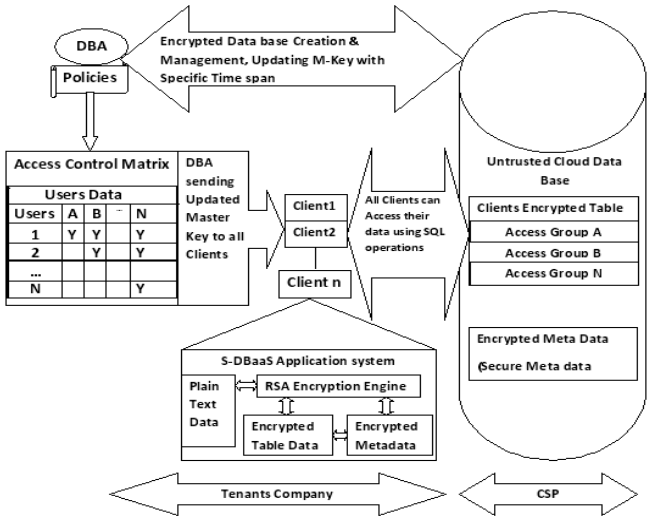

Fig. 2 Proposed System Architecture.

Secure MuteDB:

An S-MuteDB allow multiple and independent clients to connect directly to untrusted cloud. Assume an organization obtain database as a service from untrusted cloud provider [1]. A secure MuteDB manage database related information, such as encrypted database and encrypted metadata [1]. An encryption of database in cloud database prevents the violation of confidentiality by untrusted cloud provider [1]. S-MuteDB stores a metadata in cloud and allow S-MuteDB 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 S-MuteDB client [3].

In S-MuteDB system there are unique trusted entity named as DBA will manage and maintain whole scenario of the system [3]. It is the first responsibility of DBA that DBA will create and manage the access control matrix policies [3] in which it is decided that which client has access right to retrieve how many no. of data base tables from encrypted cloud database and which client has access deny for which data tables. Then according to access control matrix policies DBA maintain no. of access groups for those particular respective clients. The main aim of access control matrix policy with access groups is to allow multiple distributed clients to access encrypted

cloud database concurrently as well as independently. Then its second responsibility of DBA to create and initialize the cloud database, then under the control of DBA each client will encrypt his data tables using S-MuteDB system and store it to the untrusted cloud database in encrypted form, then S-MuteDB will maintain the database metadata as well as table metadata of all encrypted tables and maintaining one separate Secure metadata table which having all the secrete keys of encrypted database tables. After that S-MuteDB system will encrypt this metadata table using one unique randomly generated secrete key known as Master key (Mkey) [1],[3] and converting plaintext metadata table in to the encrypted metadata table which also stored on same cloud database. After that S-MuteDB will allow DBA to update the master key with specific time period regularly and sending that updated Mkey to all clients in order to perform various SQL operations like create, select insert update and delete etc. on encrypted cloud database.

V. IMPLIMENTATION STRATEGY

This S-MuteDB System is divided into various modules as follows

A. Module 1: Secure Encrypted Metadata Storage Table:

This module of Secure-MuteDB generates a metadata which include all the information need to access the data from encrypted database [1]. Secure MuteDB 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 MuteDB use two metadata.

B. Module 2: Encrypted 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: Encrypted Table metadata

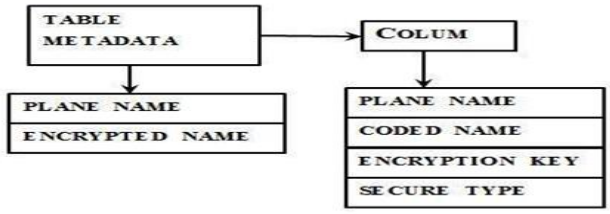

Fig. 3 Table Metadata Structure

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, Secure-MuteDB clients can use caching policies to reduce the bandwidth overhead.

D. Module 4: Secure -MuteDB 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 Secure-MuteDB 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 Secure-MuteDB 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: System Setup Phase

So in this module we explain that how to initialize a secure MuteDB 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 MuteDB 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 MuteDB 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 MuteDB with respect to state-of-the-art solutions.

VI. MATHEMATICAL MODEL OF SYSTEM

Basic steps in mathematical model:

The system can be defined as:

S = {I, E, D, STN, STCN, SMST, Mkey, F, O}

Such that,

I = Set of input parameters i.e. {PTN, PTCN}

Where,

PTN =Plain Text Table Name.

PTCN =Plain Text Table Column Name.

E = Set of Encryption Keys i.e. {EO, E1… En}

D = Set of Decryption Keys i.e. {D0, D1… Dn}

Mkey =Master Key which is known to all Trusted S-DBaaS Clients.

STN =Set of all Secure Table Names.

STCN =Set of all Secure Table Column Name.

SMST =Secure Metadata Storage Table.

O =Set of Output Parameters i.e. Decrypted Tenants data.

F=Set of Functional Parameters.

i.e. **F = {F1, F2, F3, F4, F5 & F6}**

Where,

Function F1:

Check for the security parameters for authenticates login users.

Function F2:

This function is responsible for encryption of Plain Text Table Names using RSA encryption technique and produces a secure table name that uses same encryption key for all table names.

i.e. **$STN = PTN^{E1} \bmod n$.**

Function F3:

This function is responsible for encryption of all column names and contains of secure table. It uses separate keys for each column where keys are randomly generated by S-MuteDB Clients such that:

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

Function F4:

This function is responsible for encryption of metadata such that same encryption key will be used for encryption of both i.e. database metadata as well as table metadata by following ways:

- Encrypt database metadata which contains encryption keys that are used for a secure type which having the field confidentiality set to database.
- 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-MuteDB clients and maintain Secure Metadata Storage Table.
- 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.

Function F5:

This function is responsible for decryption.

Consider,

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

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

[D x E] Mod [P - 1] x [Q - 1] = 1

- The S-MuteDB Clients machine will be decrypt the SMST table as follows:

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

Function F6:

This function is responsible for the decryption of Tenants data tables and information with the help of the information contains in decrypted SMST table.

VII. 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], [2], [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-MuteDB Model proposes a different approach where all data and metadata are stored in cloud database and eliminating all intermediate proxy servers [1], [3]. The S-MuteDB can retrieve required metadata from the entrusted databases by using SQL statement in order to the S-MuteDB client can access to the entrusted cloud database independently with guarantee of availability and scalability [1].

VIII. ADVANTAGES OF PROPOSED S-MuteDB SYSTEM

In this proposed S-MuteDB system there is no need of any modification in the existing cloud databases. The S-MuteDB architecture is immediately applicable to any kind of current existing cloud databases. There are no any practical as well as theoretical limits to expand this S-MuteDB system to the platforms. It supports to any encryption algorithm. It has a definite guarantee to data confidentiality by allowing a cloud database servers to execute parallel SQL operations (not only read or Wright but also modified and updates to database structures) over encrypted data.

It provides scalability, availability, security, elasticity and data confidentiality by eliminating untrusted intermediate servers.

IX. 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-MuteDB systems response time is very less than other existing system. In fig.4 we have shown the Comparison for query response time among existing vs. proposed system.

We also test and analyze Performance measure for SQL query response time for encryption in proposed system for various operations such as Database creation, Table creation, update, insert and delete etc. while performing the update as well as insert query operations the proposed S-MuteDB systems response time is very less than other existing system. In fig.5 we have shown the Comparison for query response time among various operations in proposed system. In fig.6 we have shown Performance measure of Encryption and Decryption response time for Metadata, Database, MAC and able operations.

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

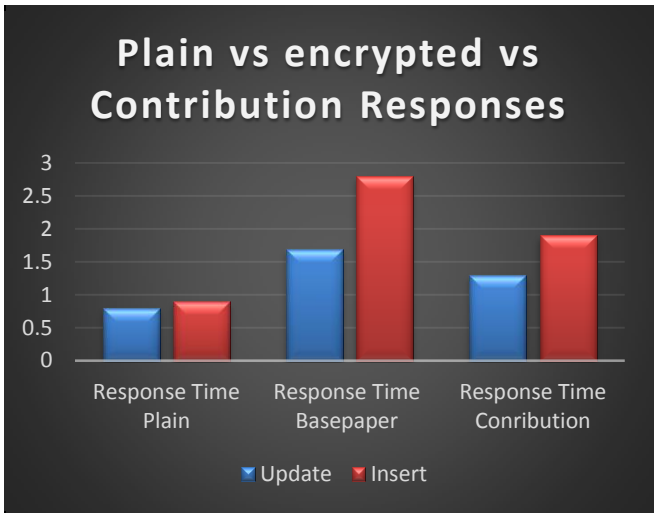

Fig. 4 Comparison for query response time among existing vs. proposed system.

Table 2: Performance measure for SQL query response time of proposed system for various operations.

Database Creation	Table Creation	Insert	Update	Delete
1.8	2.2	1.9	1.3	1.7

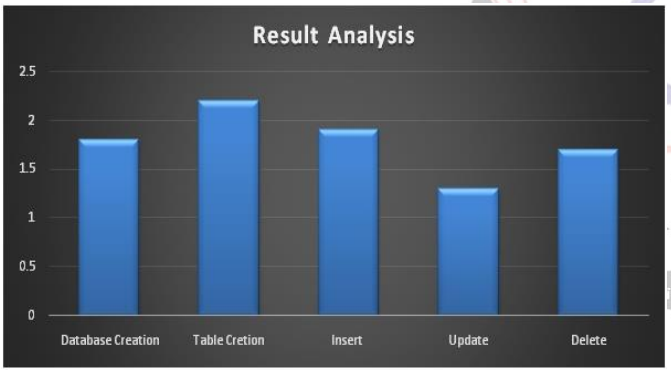

Fig.5 Performance measure for SQL query response time of proposed system for various operations.

Table 3: Performance measure of Encryption and Decryption response time for Metadata, Database, MAC and able operations.

	Metadata	**DB**	**MAC**	**Table**
Encryption	449 ms	10 ms	10 ms	285 ms
Decryption	378 ms	8 ms	9 ms	249 ms

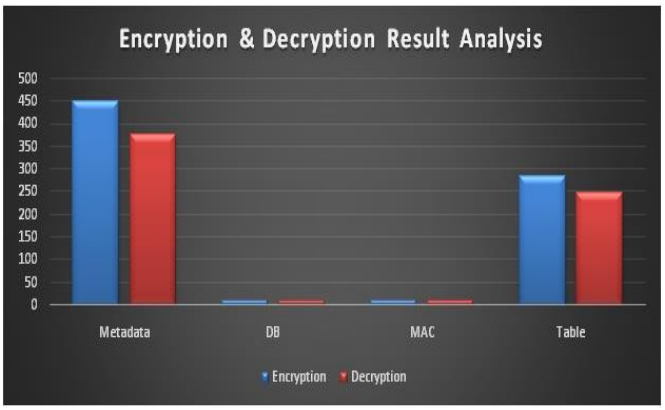

Fig. 6 Performance measure of Encryption and Decryption response time for Metadata, Database, MAC and able operations.

X. 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 [1]. 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 [1].

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 [1], [10]. We propose the S-MuteDB system by using asymmetric key cryptographic method i.e. RSA encryption and decryption method in which both the operations i.e. encryption and decryption will be performed with two different secrete keys. But in existing system symmetric key algorithms i.e. AES are used. So, the S-MuteDB shows more strength of data confidentiality using Asymmetric key algorithm. As shown in fig.4, fig.5 and fig.6 We test the performance of S-MuteDB system for various SQL operations against old system, the S-MuteDB shows minimum response time than existing system to perform SQL operations like insert, update, select and delete etc. By using forward security for master key S-MuteDB strengthen the confidentiality and security of master secret key against the chance of unauthorized access [2]. By implementing access control matrix policies the S-MuteDB allows to any no. of

authorized clients to access encrypted cloud database independently as well as concurrently [3].

XI. FUTURE SCOPE

In future data filtration operation can be implemented for sharing of data in cloud storage. It means the unwanted data files can be detected and sharing of those files on cloud can be restricted. Now days there are various techniques currently used to secure the database using replication (duplication) of data base files on different cloud servers, it leads to wastage of storage space [6]. So, we can extend the power of S-MuteDB system by implementing secure deduplication techniques to reduce wastage of cloud storage space [6]. We can further modify our S-MuteDB system with both the deduplication techniques [6] i.e. File level deduplication as well as block level deduplication in order to delete duplicate redundant data files from cloud storage space to save more storage space of cloud database [6].

ACKNOWLEDGEMENT

We wish to express our sincere gratitude to our project guide Prof. Mrs. S. Y. Raut, Head of I.T. Engineering Department for providing us an opportunity for presenting a project on the topic "Secure Mute-DB Architecture for Separated and Parallel Access to Encrypted Cloud Database Using SQL Operations". We sincerely thanks to other Staff of I.T. Department for their guidance and encouragement in the partial stage completion of our project work. We also wish to express our gratitude to the official's staff members who rendered their help during the period of our project work. Last but not least we wish to avail our self of this opportunity, to express a sense of gratitude and love to our friends and our parents for their manual support, strength, and help 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, February 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. 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] Jin Li, Xiaofeng Chen, Xinyi Huang, Shaohua Tang and Yang Xiang Senior Member, IEEE and Mohammad Mehedi Hassan Member, IEEE and Abdulhameed Alelaiwi Member, IEEE, " *Secure Deduplication Systems with Improved Reliability*", DOI 10.1109/TC.2015.2401017, IEEE Transactions on Computers

[7] 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.

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

[9] "*Oracle Advanced Security*," Oracle Corporation, http://www.oracle.com/technetwork/database/options/advanced-security, Apr. 2010

[10] 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.

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

[12] 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.

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

