An Optimized SecureDBaaS Architecture for Accessing Encrypted Cloud Databases

Nikhil. Deshpande

SCOE, Pune, Maharashtra, India

Deepa Kulkarni

SKNCOE, Pune, Maharashtra, India

Abstract: **Cloud storage provides users to easily store their data and enjoy the good quality cloud applications which need not install in local hardware and software system. So benefits are clear, such a service is also gives users' physical control of their outsourced data, which provides control over security problems towards the correctness of the storage data in the cloud. So, in order to do this new problem and further achieve secure and dependable cloud storage services. Setting discriminating information in the hands of a cloud supplier ought to accompany the insurance of security and accessibility for information at rest, in movement, and being used. We propose a novel structural engineering that coordinates cloud database administrations with information secrecy furthermore the likelihood of executing simultaneous operations on encoded information. This is the first arrangement supporting geologically circulated customers to join straightforwardly to an encoded cloud database, and to execute simultaneous and autonomous operations including those adjusting the database structure. The proposed structural planning has the further focal point of taking out halfway intermediaries that utmost the flexibility, accessibility, and versatility properties that are characteristic in cloud-based arrangements. The adequacy of the proposed construction modeling is assessed through hypothetical investigations and far reaching test results focused around a model execution subject to the TPC-C standard benchmark for diverse quantities of customers and system latencies**

Key Words: ***Access control, authentication, attribute-based signatures, attribute-based encryption, cloud storageSecureDBaas.***

I. INTRODUCTION

Quick growth in communications, storage and processing information allow us to move all data into some remote place. Data management systems activated by automated traditional tasks such as keeping record of business transactions. This data involved mainly of numeric and characteristics. Cloud is not a particular product; it is a simple technique of delivering IT services based on demand, adaptable to rescale as required based pay-as-use model. Cloud computing becomes popular because of its delivery of flexible access to the cloud data storage. The cloud data storage capacity allows users to access and extract the sensitive information whenever they require.

Cloud Computing, provides Internet based service and use of computer technology. This is cheaper and has more strong processors, together with the software as a service (SaaS) computing architecture, is transforming data into data centers on huge scale. The flexible network connections and increasing network make it even possible that users now use high quality services from data and provides remote on data centers. Storing data in (removed) the cloud offers great help to users since they don't have to care about the problems of hardware problems.

Cloud computing has ended up financially famous, as it guarantees to ensure versatility, flexibility, and high accessibility effortlessly [1], [2]. Guided by the pattern of the everythingas an issue (Xaas) model, information stockpiles, virtualized foundation, virtualized stages, and additionally programming and applications are generally given and expended as benefits in the cloud. Distributed storage administrations can be respected as an issue benefit in distributed computing, which includes the conveyance of information stockpiling as an issue, including database-like administrations and system joined capacity, regularly charged on an utility registering premise, e.g., every gigabyte every month. Cases incorporate Amazon Simpledb1, Microsoft Azure storage2, and so on. By utilizing the distributed storage benefits, the clients can get to information put away in a cloud whenever and anyplace utilizing any gadget, without thinking about a lot of capital venture when sending the fundamental equipment frameworks.

SecureDBaaS that backings the execution of simultaneous also autonomous operations to the remote encoded database from numerous topographically conveyed customers as in any decoded DBaaS setup. To attain to these objectives, SecureDBaaS incorporates existing cryptographic plans, seclusion instruments, and novel methodologies for administration of scrambled metadata on the untrusted cloud database. This paper contains a hypothetical talk about answers for information consistency issues because of simultaneous and autonomous customer gets to scrambled information. In this connection, we can't apply completely holomorphic encryption plans [7] due to their inordinate computational many-sided quality.

The SecureDBaaS building design is customized to cloud stages and does not present any mediator intermediary alternately representative server between the customer and the cloud supplier. Taking out any trusted middle of the road server permits SecureDBaaS to attain to the same accessibility, dependability, and flexibility levels of a cloud DBaaS. Other recommendations (e.g., [8], [9], [10], [11])

based on middle of the road server(s) were viewed as impracticable for a cloud-based arrangement on the grounds that any intermediary speaks to a solitary purpose of disappointment and a framework bottleneck that restricts the principle profits (e.g., adaptability, accessibility, and flexibility) of a database administration conveyed on a cloud stage. Not at all like

SecureDBaaS, architectures depending on a trusted middle of the road intermediary don't help the most commonplace cloud situation where topographically scattered customers can simultaneously issue read/compose operations and information structure changes to a cloud database. A substantial set of examinations focused around genuine cloud stages show that SecureDBaaS is promptly relevant to any DBMS on the grounds that it obliges no change to the cloud database administrations. Different studies where the proposed building design is liable to the TPC-C standard benchmark for distinctive quantities of customers and system latencies demonstrate that the execution of simultaneous read and compose operations not altering the SecureDBaaS database structure is practically identical to that of decoded cloud database. Workloads including changes to the database structure are likewise upheld by SecureDBaaS, yet at the cost of overheads that appear satisfactory to accomplish the fancied level of information privacy. The inspiration of these results is that system latencies, which are commonplace of cloud situations, have a tendency to veil the execution expenses of information encryption on reaction time. The general conclusions of this paper are imperative in light of the fact that surprisingly they exhibit the materialness of encryption to cloud database benefits as far as practicality and execution.

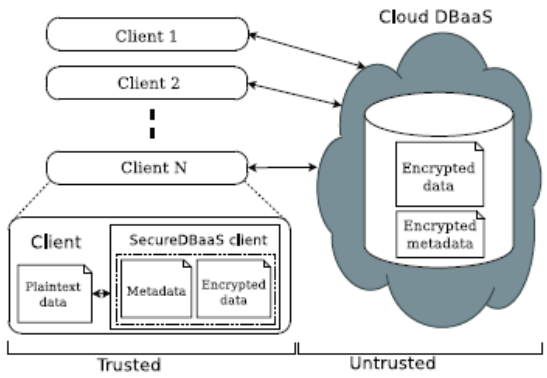

Fig. 1. SecureDBaaS architecture.

II. RELATED WORK

SecureDBaaS relates all the more nearly to works utilizing encryption to ensure information oversaw by untrusted databases. In such a case, a fundamental issue to address is that cryptographic systems can't be naı¨vely connected to standard DBaaS since DBMS can just execute SQL operations over plaintext information. A few DBMS motors offer the likelihood of encoding information at the filesystem level through the purported Transparent Information Encryption characteristic [16], [17]. This gimmick makes it conceivable to construct a trusted DBMS over untrusted stockpiling. Then again, the DBMS is trusted and decodes information some time recently their utilization. Henceforth, this methodology is not pertinent to the DBaaS connection considered by SecureDBaas, on the grounds that we accept that the cloud supplier is untrusted.

A cloud is basically a huge scale circulated framework where each one bit of information is reproduced on various geographically distributed servers to attain high accessibility and elite. In this manner, we first survey the consistency models in appropriated frameworks. Ref. [10], As an issue course reading, proposed two classes of consistency models: information driven consistency also customer driven consistency. Information driven consistency model considers the inner condition of a stockpiling framework, i.e., how redesigns course through the framework and what ensures the framework can give admiration to redesigns. Then again, to a client, it truly does not make a difference whether a stockpiling framework inside contains any stale duplicates. The length of no stale information is seen from the customer's perspective, the client is fulfilled. Along these lines, customer driven consistency model focuses on what particular clients need, i.e., how the clients watch information overhauls. Their work likewise depicts diverse levels of consistency in circulated frameworks, from strict consistency to frail consistency. High consistency intimates high cost and lessened accessibility. Ref. [11] states that strict consistency is never required in practice, and is indeed considered destructive. Actually, commanded by the CAP convention [3], [4], numerous conveyed frameworks present strict consistency for high accessibility.

At that point, we survey the work on attaining diverse levels of consistency in a cloud. Ref. [12] explored the consistency properties gave by business mists and made a few helpful perceptions. Existing business mists normally confine solid consistency certifications to little datasets (Google's Megastore and Microsoft's SQL Data Services), or give just consequent consistency (Amazon's simpledb and Google's Bigtable). Ref. [13] portrayed a few answers for accomplish distinctive levels of consistency while sending database applications on Amanzon S3. In Ref. [14], the consistency necessities fluctuate about whether relying upon genuine accessibility of the information, and the creators give strategies that make the framework powerfully adjust to the consistency level by checking the condition of the information. Ref. [15] (from here)proposed a novel consistency show that permits it to naturally conform the consistency levels for distinctive semantic information. At last, we survey the work on checking the levels of consistency gave by the Csps from the clients' purpose of view. Existing arrangements can be arranged into follow based checks [7], [9] and benchmark-based confirmations [16]– [19]. Follow built checks concentrate in light of three consistency semantics: security, consistency, and atomicity, which are proposed by Lamport [20], and reached out by Aiyer et al. [21]. A register is sheltered if a read that is not simultaneous with any compose returns the estimation of the latest compose, and a read that is simultaneous with a compose can give back any quality. A register is standard if a read that is not simultaneous with any compose furnishes

a proportional payback of the latest compose, and a read that is simultaneous with a compose returns either the estimation of the latest compose, or the estimation of the simultaneous compose. A register is nuclear if each perused gives back where its due of the latest compose. Misra [22] is the first and foremost to present a calculation for checking whether the follow on a read/compose register is nuclear. Taking after his work, Ref. [7] proposed logged off calculations for checking whether a key-esteem capacity framework has security, normality, and atomicity properties by developing a steered diagram. Ref. [9] proposed an online check calculation by utilizing the GK calculation [23], and utilized diverse measurements to evaluate the seriousness of infringement.

The fundamental shortcoming of the current follow based confirmations is that a worldwide clock is needed among all clients. Our answer has a place with follow based confirmations. Nonetheless, we concentrate on diverse consistency semantics in business cloud frameworks, where an inexactly synchronized clock is suitable for our answer. Benchmark-built confirmations concentrate in light of benchmarking staleness in a stockpiling framework. Both [16] and [17] assessed consistency in Amazon's S3, however indicated diverse results. Ref. [16] utilized one and only client to peruse information in the trials, and demonstrated that few inconsistencies exist in S3. Ref. [17] utilized different topographically dispersed clients to peruse information, and discovered that S3 often abuses monotonic-read consistency. The consequences of [17] legitimize our two-level reviewing structure. Ref. [18] presents a customer driven benchmarking procedure for comprehension inevitable consistency in dispersed keyvalue capacity frameworks. Ref. [19] surveyed

III. ARCHITECTURE DESIGN

SecureDBaaS is intended to permit various and autonomous customers to interface specifically to the untrusted cloud DBaaS with no halfway server. Fig. 1 depicts the general construction modeling. We accept that an occupant association gains a cloud database administration from an untrusted DBaaS supplier. The inhabitant then sends one or more machines (Customer 1 through N) and introduces a SecureDBaaS customer on each of them.

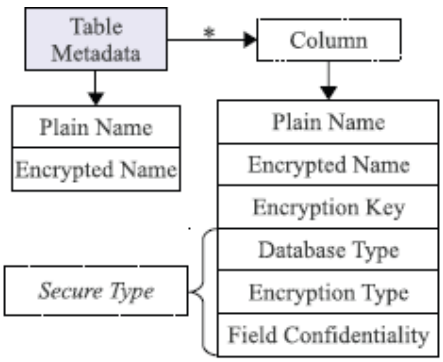

Fig. 2. Structure of table metadata.

Cloud database:

Let us assume that tenant data are saved in a relational database and 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 as we distinguish the strategies for encrypting the database structures and the tenant data.

Metadata Management:

Metadata that is generated by the SecureDBaaS contains all the information that is necessary to manage all the available SQL statements over the encrypted database in a way transparent to the user and the metadata management strategies represent an original idea because Se- cureDBaaS is the rst architecture storing all the meta- data in the un- trusted cloud database together with the encrypted tenant data in the system.

Encryption algorithm:

In order to choose the encryption algorithms used to encrypt and decrypt all the available data stored in the database table.

MODULES

Creation of database:

In this module a client creates a specific database and stores the data in the form of columns and rows then later creates the database along with the metadata which will help for later communication instead of providing the whole database to a end user.

Selection of Encryption and Decryption algorithm:

In this module we select the encryption algorithm to en- crypt and decrypt the created database and its metadata by providing the security to whole data of client which is to be uploaded on the cloud.

Cloud Database:

A cloud database is the service provider which provides services to the tenants as all the encrypted data from data owner is uploaded on cloud which provides the concur- raccess to cloud DB to the geographically deployed cli- ents and the cloud database contains encrypted database and its encrypted metadata.

Application

The data oversaw by SecureDBaaS incorporates plaintext information, encoded information, metadata, and scrambled metadata. Plaintext information comprise of data that an occupant needs to store and process remotely in the cloud DBaaS. To keep an untrusted cloud supplier from abusing classifiedness of inhabitant information put away in plain structure, SecureDBaaS receives various cryptographic procedures to change plaintext information into encoded inhabitant information and scrambled occupant information structures in light of the fact that even the names of the tables and of their segments must be scrambled.

SecureDBaaS customers deliver additionally a set of metadata comprising of data needed to encode and unscramble information and also other organization data. Indeed metadata are encoded furthermore put away in the cloud DBaaS.

SecureDBaaS moves far from existing architectures that store only occupant information in the cloud database, and recovery metadata in the customer machine [9] or part metadata between the cloud database and a trusted intermediary [8]. At the point when considering situations where various customers can get to the same database simultaneously, these past arrangements are very wasteful. For instance, sparing metadata on the customers would require cumbersome systems for metadata synchronization, and the useful incomprehensibility of permitting various customers to get to cloud database benefits freely.

Arrangements focused around a trusted intermediary are more doable, yet they present a framework bottleneck that decreases accessibility, versatility, and adaptability of cloud database administrations. SecureDBaaS proposes an alternate methodology where all information and metadata are put away in the cloud database.

SecureDBaaS customers can recover the fundamental metadata from the untrusted database through SQL articulations, so that various occurrences of the SecureDBaaS customer can get to to the untrusted cloud database autonomously with the assurance of the same accessibility and adaptability properties of average cloud DBaaS. Encryption techniques for inhabitant information and creative answers for metadata administration also capacity are portrayed in the accompanying two segments.

SecureDBaaS stores metadata in the metadata stockpiling table that is spotted in the untrusted cloud as the database. This is an unique decision that increases adaptability, however opens two novel issues regarding productive information recovery and information secrecy. To permit SecureDBaaS customers to control metadata through SQL articulations, we spare database and table metadata in a plain structure. Indeed metadata classifiedness is ensured through encryption.

Database and table metadata are scrambled through the same encryption key before being spared. This encryption key is known as an expert key. Just trusted customers that as of now know the expert key can decode the metadata and gain data that is important to encode and unscramble inhabitant information. Every metadata can be recovered by customers through an related ID, which is the essential key of the metadata capacity table. This ID is processed by applying a Message Confirmation Code (MAC) capacity to the name of the object (database or table) depicted by the relating column. The utilization of a deterministic MAC capacity permits customers to recover the metadata of a given table by knowing its plaintext name.

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. 3. Organization of database metadata and table metadata in the metadata storage table.

IV. OPERATIONS

We portray how to introduce a SecureDBaaS structural planning from a cloud database administration gained by an inhabitant from a cloud supplier. We accept that the DBA makes the metadata stockpiling table that toward the starting contains just the database metadata, and not the table metadata. The DBA populates the database metadata through the SecureDBaaS customer by utilizing arbitrarily produced encryption keys for any blends of information sorts and encryption sorts, and stores them in the metadata stockpiling table after encryption through the expert key. At that point, the DBA conveys the expert key to the genuine clients. Client access control strategies are administrated by the DBA through some standard information control dialect as in any decoded database.

In the accompanying steps, the DBA makes the tables of the encoded database. It must consider the three field privacy traits (COL, MCOL, and DBC) presented toward the end of the Section 3. Given us a chance to portray this stage by alluding to a basic yet illustrative case indicated in Fig. 4, where we have three safe tables named ST1, ST2, and ST3. Each one table STi (i ¼ 1; 2; 3) incorporates an encoded table Ti that contains scrambled inhabitant information, and a table metadata Mi. (Albeit, as a general rule, the names of the segments of the protected tables are haphazardly created; for the purpose of straightforwardness, this figure alludes to them through C1-CN.

Mathematical Model

Let consider Tables T= $\{T_1, T_{2,} \ldots . T_n\}$ with columns C = $\{C_1, C_{2,} \ldots . C_n\}$

Represent Column name of corresponding plain text tables T

T= $\sum_{i=0}^{n-1} Cname$

Create a secure table and generate random column names

$ST_i = ST_1, ST_{2,} \ldots . ST_n$ where ST_i includes an encrypted table T_i it contains encrypted tenat data and table meta data M_i

Assign coded names to the columns to secure table with corresponding plain text table

$$ST_i = C_n = Codedname$$

Define secure type of the column by assigning database type, encryption type and Field Confidentiality

Generate Encryption key and assign

Secure DBaaS client by using randomly generated encryption keys for any combinations of data types and encryption
types, and stores them in the metadata storage table after encryption through the master key.

Database and table metadata are encrypted through the same encryption key before being saved
D= $\{Db_1, Db_2, \ldots . Db_n\}$ and T= $\{T_1, T_2, \ldots . T_n\}$

Compute an association ID for each metadata by message authentication code

$$MAC(Db), MAC\ (T_1, T_2, \ldots . T_n)$$

Use the encryption key associated with their data and encryption types

Sequential SQL Operations

The first connection of the client with the cloud DBaaS is for authentication purposes. Secure DBaaS relies on standard authentication and authorization mechanisms pro-vided by the original DBMS server. After the authentication, a user interacts with the cloud database through the Secure DBaaS client. Secure DBaaS analyzes the original operation to identify which tables are involved and to retrieve their metadata from the cloud database. The metadata are decrypted through the master key and their information is used to translate the original plain SQL into a query that operates on the encrypted database. Translated operations contain neither plaintext database (table and column names) nor plaintext tenant data. Nevertheless, they are valid SQL operations that the Secure DBaaS client can issue to the cloud database. Translated operations are then executed by the cloud database over the encrypted tenant data. As there is a one-to-one correspondence between plaintext tables and encrypted tables, it is possible to prevent a trusted database user from accessing or modifying some tenant data by granting limited privileges on some tables. User privileges can be managed directly by the untrusted and encrypted cloud database. The results of the translated query that includes encrypted tenant data and metadata are received by the Secure DBaaS client, decrypted, and delivered to the user. The complexity of the translation process depends on the type of SQL statement.

Cashing of SELECT Query

Heredesigned query cache model (QCM) stores the text of a SELECT statement together with the corresponding result that was store at client side. If an identical statement is received later, the server retrieves the results from the QCM rather than sending the same statement to the server.

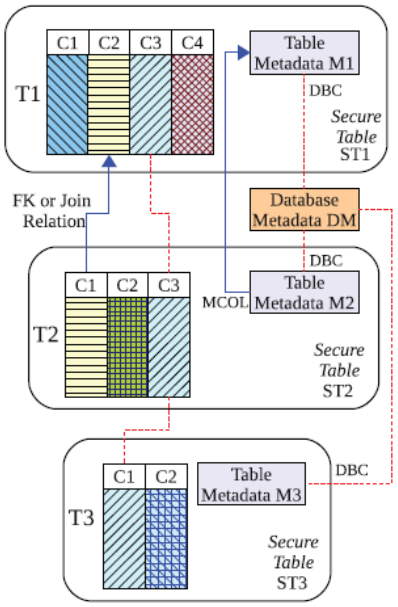

Fig. 4. Management of the encryption keys according to the field confidentiality parameter.

Concurrent SQL Operations:

The support to concurrent 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. Our architecture must guarantee consistency among encrypted tenant data and encrypted metadata because corrupted or out-of-date metadata would prevent clients from decoding encrypted tenant data resulting in permanent data losses. A thorough analysis of the possible issues and solutions related to concurrent SQL operations on encrypted tenant data. Here, we remark the importance of distinguishing two classes of statements that are supported by Secure DBaaS: SQL operations not causing modifications to the database structure, such as read, write, and update; operations involving alterations of the database structure through creation, removal, and modification of database tables (data definition layer operators).

V. IMPLEMENTATION

A. Data Management:

Whenever a cloud database acts as service provider for tenants then the cloud is created first for the system and all the information or data stored in the relational database for creating tables and column we have to access it with SQL query only is available to the end user.

B. Metadata Management:

Metadata generated by SecureDBaaS contains all the in-formation that is necessary to manage SQL statements over the encrypted database in a way which is transparent to the user and the metadata management strategies rep- resent an original idea because SecureDBaaS is the rst architecture

storing all metadata in the un-trusted cloud database together with the encrypted tenant data and the SecureDBaaS uses two types of metadata.

In a database metadata are related to the whole database and there is only one instance of this metadata type for each database.

In a table metadata are associated with one secure table where each table metadata contains all information that is necessary to encrypt and decrypt data of the associated secure table.

Both the database and the table metadata are encrypted through the same encryption key before being saved where the encryption key is called a master key and 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. Where each of the metadata can be retrieved by clients through an associated ID which is the primary key of the metadata storage table and the ID is computed by applying a Message Authen- tication Code (MAC) function to the name of the object database or table that describes the corresponding row. The deterministic MAC function allows clients to retrieve the metadata of a given table by the knowing based on its plaintext name where this mechanism has the further benet of allowing clients to access each metadata independently which is an important feature in concurrent environments and in addition to SecureDBaaS clients can use caching policies to reduce the bandwidth overhead.

VI. EXPERIMENTAL RESULTS

We show the relevance of SecureDBaaS to distinctive cloud DBaaS arrangements by actualizing and taking care of scrambled database operations on imitated and genuine cloud foundations. The present form of the SecureDBaaS model backings PostgreSQL, MySql, and SQL Server social databases. As a first result, we can watch that porting SecureDBaaS to diverse DBMS obliged minor changes identified with the database connector, also negligible changes of the codebase. To evaluate the performance overhead of encryptedSQL operations, we focus on the most frequently executedSELECT, INSERT, UPDATE, and DELETE commands ofthe TPC-C benchmark. We compare theresponse times of SELECT and DELETE, and UPDATE and INSERT operations, respectively. The second set of the tests is arranged to assess the effect of system dormancy and concurrency on the utilization of a cloud database from topographically far off customers.

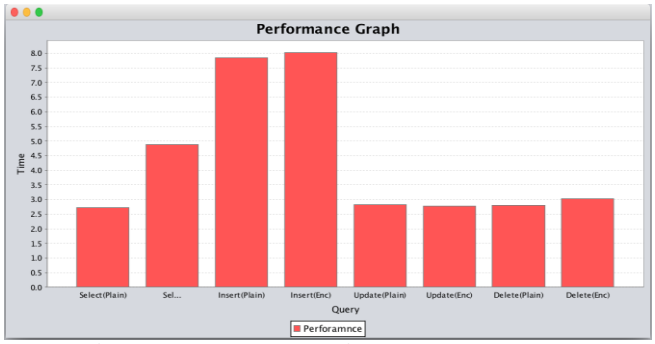

Fig 5: Performance comparison chart of plain text vs encrypted SELECT, INSERT, UPDATE and DELETE queries

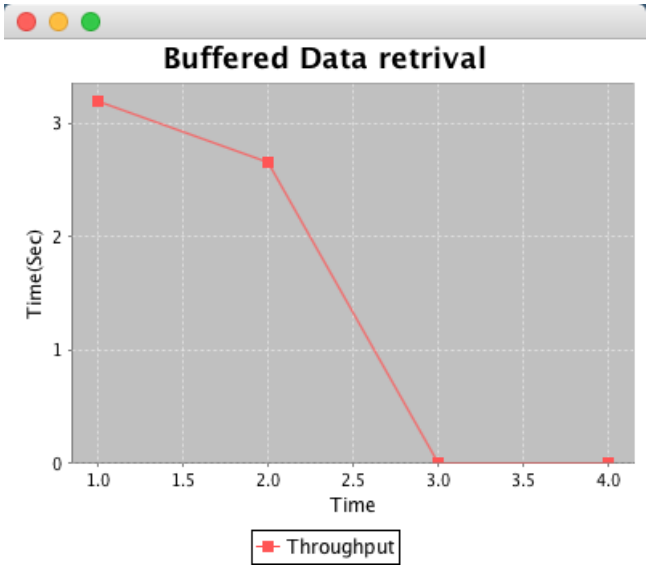

Fig 6: Performance when SELECT query is used using cache query model

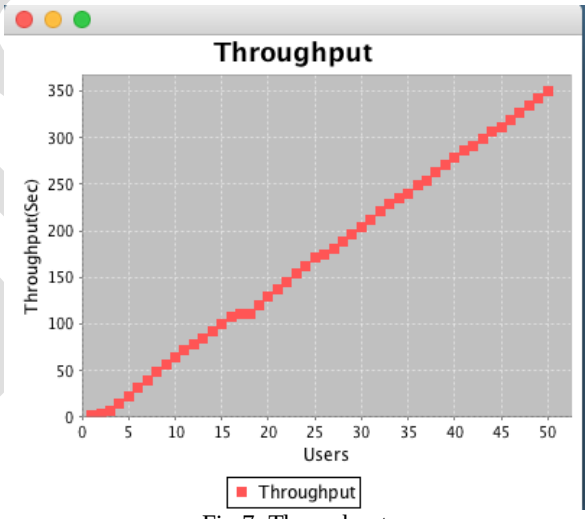

Fig 7: Throughput

VII. CONCLUSION

In this paper we propose an structural planning that ensures privacy of information put away in broad daylight cloud databases.

We propose a resourceful construction modeling that ensures classifiedness of data place away move into the open cloud databases. Not in the least like leading edge approaches, our answer doesn't rely upon a middle of the road negotiator that we have a tendency to think about a solitary purpose of disappointment and a bottleneck limiting accessibility and flexibility of run of the mill cloud info administrations

The planned structural engineering doesn't oblige changes to the cloud database , adit is immediately applicable to existing cloud DBaaS , like the xperimented PostgreSQL and Cloud info, Windows Azure, and Xeround.

An expansive part of the examination incorporates answers for backing simultaneous SQL operations (counting explanations adjusting the database structure) on encoded

information issued by heterogenous furthermore conceivably geologically scattered customers. It is worth observing that experimental results based on the TPC-C standard benchmark show that the performance impact of data encryption on response time becomes negligible because it is masked by network latencies that are typical of cloud scenarios. In particular, concurrent read and write operations that do not modify the structure of the encrypted database cause negligible overhead.

The proposed structural planning does not oblige alterations to the cloud database, and it is quickly relevant to existing cloud DbaaS, and by using the casching for SELECT query we can reduce the time required for the same queries executed again and again which also improves the response time.The query cache can be useful in an environment where you have tables that do not change very often and for which the server receives many identical queries. This is a typical situation for many Web servers that generate many dynamic pages based on database content.

REFERENCES

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

[2]. W. Jansen and T. Grance, "Guidelines on Security and Privacy in Public Cloud Computing," Technical Report Special Publication 00-144, NIST, 2011.

[3]. A.J. Feldman, W.P. Zeller, M.J. Freedman, and E.W. Felten, "SPORC: Group Collaboration Using Untrusted Cloud Resources," Proc. Ninth USENIX Conf. Operating Systems Design and Implementation, Oct. 2010.

[4]. J. Li, M. Krohn, D. Mazie`res, and D. Shasha, "Secure Untrusted Data Repository (SUNDR)," Proc. Sixth USENIX Conf. Opearting Systems Design and Implementation, Oct. 2004.

[5]. 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.

[6]. H. Hacigu¨mu¨ s¸, B. Iyer, and S. Mehrotra, "Providing Database as a Service," Proc. 18th IEEE Int'l Conf. Data Eng., Feb. 2002.

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

[8]. R.A. Popa, C.M.S. Redfield, N. Zeldovich, and H. Balakrishnan, "CryptDB: Protecting Confidentiality with Encrypted Query Processing," Proc. 23rd ACM Symp. Operating Systems Principles, Oct. 2011.

[9]. H. Hacigu¨mu¨ s¸, B. Iyer, C. Li, and S. Mehrotra, "Executing SQL over Encrypted Data in the Database-Service-Provider Model," Proc. ACM SIGMOD Int'l Conf. Management Data, une 2002.

[10]. J. Li and E. Omiecinski, "Efficiency and Security Trade-Off in Supporting Range Queries on Encrypted Databases," Proc. 19th Ann. IFIP WG 11.3 Working Conf. Data and Applications Security, Aug. 2005.

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

[12]. 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.

[13]. V. Ganapathy, D. Thomas, T. Feder, H. Garcia-Molina, and R. Motwani, "Distributing Data for Secure Database Services," Proc.Fourth ACM Int'l Workshop Privacy and Anonymity in the Information Soc., Mar. 2011.