International Journal of Science Engineering and Advance Technology

Scattered simultaneous and autonomous access to scrambled cloud databases

B.Mahalakshmi Rao[1],K.Ravi Kumar[2]
M.tech(Research scholar)[1],Associate. Professor &HOD[2] in Dept computer science & engineering
[1]kakinada institute of engineering & technology, korangi,AP,India
[2]kakinada institute of engineering & technology for women,korangi,AP,India

Abstract:
Cloud data environments square measure unpleasantly tempting for the preparing of tremendous scale applications owing to their amazingly ascendible and offered foundation. Data as a Service (DBaaS) model is utilized to oversee databases in cloud setting. Secure DBaaS standard gives data classification to distributed storage. Secure DBaaS is expected to allow various and independent buyers to append to the cloud while not middle server. Documents, data structures and data square measure encoded before exchange to the cloud. Different cryptography methods square measure won't to change over plain content into scrambled data. Table names and their section names likewise are scrambled inside of the cloud data wellbeing subject. The framework underpins topographically conveyed buyers to append on to A scrambled cloud data. Amid this paper we tend to quadrangular measure proposing new plan that coordinate distributed storage administration with data protection and have a component of flogging co-happening operations on scrambled data and together with the topographically dispersed buyers to connect on to these cloud data that is encoded and that they conjointly given to execute their operations over the cloud data. This configuration disposes of the dealers (Intermediate intermediaries) it constrains the quantifiability, flexibility, availability. High touchy data square measure encoded by RSA cryptography and standard data square measure scrambled abuse AES method so overhead on the system will be decreased.

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

I. Introduction

Distributed computing innovation is an administration based, Internet driven, protected, helpful information stockpiling and system figuring administration. It is a web based model for empowering an advantageous and on-interest system access to a mutual pool of configurable registering assets. One of the vital administration of distributed computing is database as an administration. Database-as-a-Service (DBaaS) is an administration that is overseen by a cloud administrator that backings applications, without the application group accepting obligation regarding conventional database organization capacities. With a DBaaS, the application engineers will not have to be database specialists, nor if they need to contract a database overseer (DBA) to keep up the database.

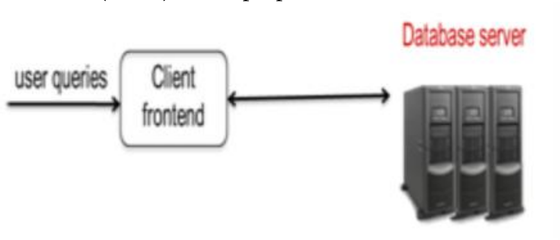

Figure 1.1: Database-as-a service

Genuine DBaaS will be accomplished when application designers can basically call a database administration and it works without considering the database. A definitive objective of a DBaaS is that the client doesn't need to consider the database. Today, cloud clients don't need to consider server occasions, stockpiling and systems administration, they simply work. Virtualization empowers mists to give these administrations to clients while computerizing a great part of the customary torment of purchasing, introducing, arranging and dealing with these abilities. Presently database virtualization is doing likewise for the cloud database and it is being given as Database as a Service (DBaaS). This venture proposes secure DBaaS. Here all databases are encoded and put away in the cloud. It permits various and disseminated clients can get to their own particular databases simultaneously and autonomously. Every client utilizes versatile encryption plan for encoding databases. Secure DBaaS ensures classification of data very still, in movement and being used when information are overseen through cloud database administrations. SecureDBaaS dispose of any middle of the road intermediary server, so a client can accomplish accessibility, versatility and flexibility of DBaaS. Same as privacy secure DBaaS keep up the simultaneousness. The customers get to the encoded database through sql inquiries and decode the database through relating calculations. In the present time of PCs our fundamental point of composing this paper for actualizing the framework is to coordinate the cloud database administrations with information secrecy and the likelihood of executing simultaneous operations on scrambled information so we utilize cloud for transferring proprietor's information where an information proprietor is the person who has transferred his information on cloud and he or she is not guaranteed about his or her information so we need to store the information on the cloud by encoding it. For the most part this encryption of information happens at customer side and metadata of that information likewise made that is secureDBaaS idea where the scrambled information is put away at the cloud alongside its encoded metadata and after that the approved customers can get to the information by utilizing just metadata. We consider this to be the first arrangement that backings geologically conveyed customers to associate straightforwardly to a scrambled cloud database and to execute simultaneous and autonomous operations including those altering the database structure. Our proposed framework incorporates the upside of wiping out halfway intermediaries that breaking point the flexibility; accessibility and adaptability properties that are characteristic in the cloud-based arrangements where a secureDBaaS gives a few unique elements that separate it from past work in the field of security for remote database administrations.

II. RELATED WORK

We tried our strategy on two certifiable information sets – the spine systems of Stanford University and Internet2, speaking to an undertaking system and an across the country ISP. The outcomes are empowering: on account of the structure of certifiable principle sets, the quantity of test parcels required is shockingly little. For the Stanford system with more than 757,000rules and more than 100 VLANs, we just need 4,000 parcels to practice all sending principles and ACLs. On Internet2, 35,000 parcels suffice to practice all IPv4 sending guidelines. Put another way, we can check each tenet in each switch on the Stanford spine ten times each second, by sending test parcels that expend under 1% of system data transfer capacity. The connection spread for Stanford is considerably littler, around50 bundles which permits proactive liveness testing each securing 1% of system data transfer capacity. The principle commitments of our work are: (1) A review of system administrators uncovering normal system disappointments and their main driver (2) the test parcel era calculation (Section 4.1), (3) A flaw restriction calculation to break down results and disengage the defective gadget and tenet (Section 4.2), (4) use instances of ATPG structure for utilitarian and execution Resources" A.J. Feldman, W.P. Zeller, M.J. Freedman, and E.W. Felten [2], have depicted Cloud-based administrations are an appealing organization model for client confronting applications like word handling and calendaring. In SPORC, a server watches just scrambled information and can't digress from right execution without being identified. SPORC permits simultaneous, low-idleness altering of shared state, grants disengaged operation, and backings element access control even in the vicinity of simultaneousness Acknowledgments. "Secure Untrusted Data Repository (SUNDR)" J. Li, M. Krohn, D. Mazie` res, and D. Shasha, [3] have proposed SUNDR is a system document framework intended to store information safely on untrusted servers. SUNDR's convention accomplishes a property called fork consistency, which ensures that customers can identify any respectability or consistency disappointments the length of they see one another's record changes. Estimations of our execution show execution that is normally near and now and again superior to the well-known NFS record framework"Secure Untrusted Data Repository (SUNDR)" J. Li, M. Krohn, D. Mazie' res, and D. Shasha, [3] have proposed SUNDR is a network file system designed to store data securely on untrusted servers. SUNDR's protocol achieves a property called fork consistency, which guarantees that clients can detect any integrity or consistency failures as long as they see each other's file modifications. Measurements of our implementation show performance that is usually close to and sometimes better than the popular NFS file system. "Depot: Cloud Storage with Minimal Trust" P. Mahajan, S. Setty, S. Lee, A. Clement, L. Alvisi, M. Dahlin, and M. Walfish [4] have described the design, implementation, and evaluation of Depot, a cloud storage system that minimizes trust assumptions. Depot began with an attempt to explore a radical point in the design space for cloud storage: trust no one. "Providing Database as a Service" H. Hacigu¨ mu¨ s, B. Iyer, and S. Mehrotra [5], have proposed a new paradigm for data management in which a third party service provider hosts "database as a service" providing its customers seamless mechanisms to create, store, and access their databases at the host site. The authors introduced NetDB2, an internet-based database service built on top of DB2 that provides users with tools for application development, creating and loading tables, and performing queries and transactions. "Fully

Homomorphic Encryption Using Ideal Lattices" C. Gentry [6], has proposed a fully homomorphism encryption scheme – i.e., a scheme that allows one to evaluate circuits over encrypted data without being able to decrypt. The circuit privacy of E2 immediately implies the (leveled) circuit privacy of our (leveled) fully homomorphism encryption scheme. "Crypt: Protecting Confidentiality with Encrypted Query Processing" R.A. Popa, C.M.S. Redfield, N. Zeldovich, and H. Balakrishnan [7], have described, Crypt is a system that provides practical and provable confidentiality in the face of these attacks for applications backed by SQL databases. It works by executing SQL queries over encrypted data using a collection of efficient SQL-aware encryption schemes. "Efficiency and Security Trade-Off in Supporting Range Queries on Encrypted Databases" J. Li and E. Omiecinski [8], had discussed concerns about protecting sensitive information of data and queries from adversaries in the DAS model. Data and queries need to been cryptic, while the database service provider should be able to efficiently answer queries based on encrypted data and queries. "Distributing Data for Secure Database Services," V.Ganapathy, D. Thomas, T. Feder, H. Garcia-Molina, and R.Motwani [9] have proposed, the advent of database services has resulted in privacy concerns on the part of the client storing data with third party database service providers. This paper provide algorithms for (1) distributing data: our results include hardness of approximation results and hence a heuristic greedy algorithm for the distribution problem (2) partitioning the query at the client to queries for the servers is done by a bottom up state based algorithm. Finally the results at the servers are integrated to obtain the answer at the client. "How to Share a Secret," A. Shamir[10], has described how to divide data D into n pieces in such a way that D is easily reconstruct able from any k pieces, but even complete knowledge of k - 1 pieces reveals absolutely no information about D. This technique enables the construction of robust key management schemes for cryptographic systems that can function securely and reliably even when misfortunes destroy half the pieces and security breaches expose all but one of the remaining pieces.

III. Problem Definition

Database-as-a-service is a secondary cloud computing service model. Users can access the database from cloud without need to installation of software and configuring hardware. The DBAAS reduces the maintenance cost in organization. The main drawback is lack of control over the database. Cloud providers are untrusted, whatever data storing in database, there is no security for data, and cloud providers can hack the data. This paper considers databaseas-a-service and providing security for stored databases. Paper proposes secure database-as-a-service (SDBAAS).

IV. CONCURRENCY CONTROL PROTOCOLS

In what follows, we briefly present the most prominent concurrency control protocols that can be used in cloud database.

Self-optimizing One Copy Serializability (SO- 1SR)

1SR is the strongest and well known correctness criterion for applications that are newly deployed in the cloud. It assures the serializable execution of concurrent transactions and a one copy view of the data. The most commonly used approaches to implement 1SR is to use lock based protocols such as strict two-phase locking (S2PL) for providing serializable transaction execution and two-phase commit (2PC) for synchronous updating all replicas.

Transaction model:

In a system providing 1SR, each transaction which writes to a data object must update all copies of the data object. In case of update transactions the replicated data increases the response time and thus decreases the overall scalability of the system. In order to exploit the merits of the cloud, it is essential to provide scalability, availability, low cost and strongly consistent data management. Under distributed systems, it is not possible to provide consistency and availability. The stronger consistency level decreases the availability and scalability. In cloud environments, the cost of guaranteeing a certain consistency level on top of replicated data is to be considered. Strong consistency is costly; on the other hand, weak consistency is cheaper, but may lead to high operational costs of compensating the effects of anomalies and access to stale data. The first generation cloud DBMS's provide on the weak consistency in order to provide maximum scalability and availability. It is sufficient for satisfying requirements related to consistency of simple cloud applications. However, more sophisticated like web shops, online stores and credit card services requires strong consistency levels. The advantages of cloud such as availability and scalability are not yet exploited by existing commercial and open source DBMS's which provide strong consistency [12]. SO-1SR (self-optimizing 1SR) is a novel protocol for replicated data in a cloud that dynamically optimize all phases of transaction executions. System model of SO-1SR assumes that applications are built on the top of a cloud data environment.

Implementation:

The SO-1SR middleware should be present at each replica node. The transactions that are submitted by the client to the application servers are forwarded to the SO-1SR middleware for optimal execution. The

SO-1SR is based on a fully replicated system and flat transaction model. Protocols like 2PC or Paxos are needed to provide strong consistency guarantees. The main goal of SO-1SR is to decrease latency by using dynamic optimization technique at different phases of transaction life cycle, not to replace protocols like 2PC or Paxos. .

Snapshot Isolation

The transactional guarantees of SI are weaker than 1SR, such that the database system can achieve increased concurrency by relaxing isolation requirements on transaction. In SI, the transaction attempting read is never blocked. The tradeoff between transaction isolation and performance is that higher degrees of transaction isolation assure fewer anomalies. Anomalies avoided by 1SR are also avoided in SI. Under SI, write skew anomaly is possible if two transactions concurrently update one or more common data item. For example, consider two transactions Tm and Tn. Transaction Tm reads data items p and q and then updates concurrently with other transaction Tn that reads data item p and q and then updates q. Here transaction Tm and Tn do not have a write-write conflict because none of the transaction updates a common data item. Different variations of SI exist for replicated systems like cloud which provide different consistency guarantees. In a lazily synchronized replicated database system; if two transactions Ts and Tv do not have a write–write conflict under SI, then their updates may be committed in the order Ts followed by Tv at a site S1 but in reverse order at another site S2 in which each site individually guarantees SI. In this case, consider a transaction Tk that reads x and y at site S1 and view database state from the commit of Ts will not view this same database state if it were to be executed on the database replica at site S2.But this kind of replica in consistency will not occur in a centralized database system that guarantees SI. SI was introduced by Berenson et al [13]. SI is defined as; it does not allow dirty reads, dirty writes, non-repeatable reads, phantoms or lost updates. Write skew anomalies are possible in SI. By the definition of SI, when the transaction starts the system assigns a transaction Ta start timestamp called start (T). The database state seen by T is determined by start (T). The system can choose any time less than or equal to the actual start time of T to start (T). The update transactions made by Tl that commit after start (T) will not be visible to T. Only update transaction that commits before start (T) will be visible to T. Each transaction T is able to see its own updates are also a requirement in SI. Thus, if T updates a database item and reads that item, then T will see the updating even though the update occurred after the start (T). International Journal of Research in Advent Technology, Vol.2, No.10, October 2014 E-ISSN: 2321-9637 47

Transaction model:

Commit timestamp, commit (T) is assigned to a transaction when a transaction is to commit. The time commit (T) is more recent than any other start or commit timestamp assigned to any transaction. If no other committed transaction Tk with lifespan [start (Tk), commit (Tk)] that overlaps with a T's lifespan of [start (T), commit (T)] write data that T has also written then only T commits. Otherwise, to prevent lost updates T is getting aborted. This technique of preventing lost updates is called the first-committerwins (FCW) rule. Transaction inversions are possible in SI, i.e. for every pair of transactions T1 and T2, if T2 executes after T1 then T1 will view T1's updates. This is because the actual start time of T2 can be larger than that of a start (T2). In particular, if T2 starts after T1 has finished, then T2 will see a database state that does not contain the effects of T1. In order to prevent these kinds of transaction inversions, strong SI is introduced. In the definition of strong SI (SSI), if for every pair of committed transactions Tp and Tq in transaction history TH such that Tp's commit precedes the first operation of Tq, start (Tq) > commit (Tp) and it is SI then we can say that the transaction execution history TH is strong SI. 3.2.2. Implementation: The decentralized model of SI based transactions consists of some mechanisms such as: (a) Keeping a consistent, committed snapshot for reading (b) a global sequencer is used for arranging the transactions by allocating commit timestamps (c) detection of write-write anomalies in concurrent transactions and (d) atomically commit the updates and make them durable. In the model, each transaction goes through a sequence of phases during execution. The main phase is the active phase in which all read/write on data item is performed in this phase. The remaining phases are part of the commit of the transaction. Validation phase is required for detecting the conflicts among transactions that are executed concurrently.

Session Consistency Session

Consistency is considered to be the minimum consistency level in a distributed environment that does not result in complexities for application developers. Under Session Consistency, the application will not see its own updates and may get inconsistent data from successive accesses. The key idea is that, all data does not need the same level of consistency. There is a term called consistency rationing i.e. the data is divided into three categories A, B, C and each type of data is treated differently depending on the consistency level provided. The category A contains data in which consistency violations may result in large penalty costs. The category B includes data where the consistency requirements change over time. Category C comprises data in which inconsistency is acceptable. Session consistency considers data under category C. C category is always a preferred category for placing data in the cloud database [14]. By considering a transaction cost and response time the session consistency is very cheap; because only few messages are needed as compared to strong consistency guarantees. The performance level can be increased by providing extensive caching mechanisms which in turn lowers the cost.

Cost-Based Adaptive Concurrency Control (C3)

Cost plays an important role in the cloud environment along with the performance [15]. Consistency leads to high cost, whereas weak consistency leads to high operational costs [16]. In C3 approach, a consistency rationing model is used which categorized the data into three: the first category contains data which require ISR, the second category data require SC and the third category data handled with adaptive consistency. At the data level, specific policy will be defined based on that policy consistency level is selected between 1SR and SC at the time of running. Moreover, C3 is implemented on the top of 1SR, SC and SSI concurrency protocols by utilizing the resources provided by the cloud providers. The update anywhere and full replication procedure are the basis for the C3 system model. The updating of all replicas will be carried out in ISR and SSI transactions using 2PC, while SC transactions only commits at the remote local replicas. The C3 model does not introduce any hindrance for the replication strategy. Each and every replica in the system is known to all other replicas. The C3 procedure uses an adaptive layer, which allows the dynamic switching between the different CCPs at runtime. Thus the reduction of operational costs and transaction response time is possible [17].

V. Methodology

This paper propose a novel architecture that integrates cloud database services with data confidentiality and the possibility of executing concurrent operations on encrypted data. This is the novel 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 architecture has the further advantage of eliminating intermediate proxies that limit the elasticity, availability, and scalability properties that are intrinsic in cloud-based solutions. Secure DBaaS provides several original features that differentiate it from previous work in the field of security for remote database services.

Proposed System

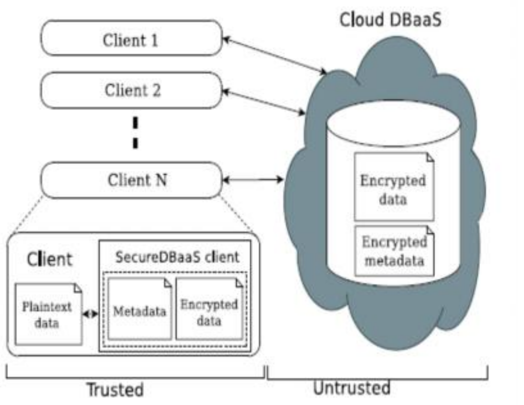

Figure: Architecture of Secure DBaaS

Data Management

It assumes that tenant data are saved in a relational database. It has 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. This paper distinguishes the strategies for encrypting the database structures and the tenant data. Encrypted tenant data are stored through secure tables into the cloud database. To allow transparent execution of SQL statements, each Plaintext table is transformed into a secure table because the cloud database is untrusted. Secure DBaaS offers three field confidentiality attributes: 1) Column (COL) is the default confidentiality level that should be used when SQL statements operate on one column; the values of this column are encrypted througha randomly generated encryption key that is not used by any other column. 2) Multicolumn (MCOL) should be used for columns referenced by join operators, foreign keys, and other operations involving two columns; the two columns are encrypted through the same key. 3) Database (DBC) is recommended when operations involve multiple columns; in this instance, it is convenient to use the special encryption key that is generated and implicitly shared among all the columns of the database characterized by the same secure type. The choice of the field confidentiality levels makes it possible to execute SQL statements over encrypted data while allowing a tenant to minimize key sharing.

Metadata Management

Metadata generated by Secure DBaaS 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 Secure DBaaS is the first architecture storing all metadata in the untrusted cloud database together with the encrypted tenant data. Secure DBaaS uses two types of metadata. 1) Database metadata are related to the whole database. There is only one instance of this metadata type for each database. 2) 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.

VII. Conclusion

DBaaS is one of the main service of cloud computing. This project proposed secure DBaaS, which allow multiple cloud users can access database concurrently and independently over the encrypted data. It also maintains the confidentiality of data because it does not rely on proxy server. It provides same availability, scalability and elasticity of traditional DBaaS. It provides confidentiality to the cloud database using adaptive encryption architecture. This project also compares the performance of sql aware encryption scheme and adaptive encryption scheme.

Future Enhancement:

In future, the different concurrency controls in the encrypted cloud database such as SO-ISR, SI, SC and C 3 are going to be used. These protocols provide different data consistency levels at different costs. The concurrency and performance varies according to the concurrency protocols used in the cloud environment. The architecture which supports the distributed, concurrent and independent access to the encrypted cloud database is SecureDBaaS. SecureDBaaS uses the isolation mechanisms for providing concurrent access to its geographically distributed clients.

References:

[1] Homomorphic Encryption Using Ideal Lattices,‖ Proc. 41st Ann. ACM Symp. Theory of Computing, May 2009.

[2] 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.

[3] 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, June 2002.

[4] 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.

[5] 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.

[6] 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.

[7] V. Ganapathy, D. Thomas, T. Feder, H. GarciaMolina, and R. Motwani, ―Distributing Data for Secure Database Services,‖ Proc. Fourth ACM Int'l Workshop Privacy and Anonymity in the Information.

[8] A. Shamir, ―How to Share a Secret,‖ Comm. of the ACM, vol. 22, no. 11, pp. 612-613, 1979.

Authors:

Mr.**B.Mahalakshmi Rao** is a student of KIET College of Engineering &Technology, korangi, Presently Iam pursuing my M.Tech[CS] from this college. I received my B.Tech from Kakinada Institute of Engineering and Technology affiliated to JNT University Kakinada in the year 2013. My area of interest includes Computer Networks and Data Base Management System.

Mr. **K. Ravi Kumar**, Associate Professor &HOD has completed his graduation and post graduation in Pragati Engineering College, Surampalem in 2010.He has 9 Years of experience in Teaching .He published more than 8 papers in national and international journals. He has been working on Big data analytics. Data miming, Bioinformatics and cloud computing are the interested areas of research.