International Journal of Science Engineering and Advance Technology

Concurrent and independent operations to the remote encrypted cloud database from many geographically distributed clients

PalikaSurendranath [1], S.Madhri [2]
[1]PG Scholar, Pydah College of Engineering, Kakinada, AP, India, E-mail: surendra.kakinada@gmail.com.
[2]Assistant Professor, Pydah College of Engineering, Kakinada, AP, India.

Abstract— since data in cloud will be placed anywhere, because of the critical nature of the applications, it is important that clouds be secure. The major security challenge with clouds is that the owner of the data may not have control of where the data is placed. This is because if one wants to exploit the benefits of using cloud computing. This requirement imposes clear data management choices: original plain data must be accessible only by trusted parties that do not include cloud providers, intermediaries, and Internet; in any untrusted context, data must be encrypted. Satisfying these goals has different levels of complexity depending on the type of cloud service. We propose Secure DBaaS as the first solution that allows cloud tenants to take full advantage of DBaaS qualities, such as availability, reliability, and elastic scalability, without exposing unencrypted data to the cloud provider. The architecture design was motivated by goal: to allow multiple, independent, and geographically distributed clients to execute concurrent operations on encrypted data, including SQL statements that modify the database structure.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. The efficacy of the proposed architecture is evaluated through theoretical analyses and extensive experimental results based on a prototype implementation subject to the TPC-C standard benchmark for different numbers of clients and network latencies.

Key words: Cloud, security, confidentiality, Secure DBaaS, database.

I. INTRODUCTION

The Aim of our system is, to integrate cloud database services with data confidentiality and thepossibility of executing concurrent operations on encrypted data. We use cloud for uploading owner's data.Data Owner who has uploaded his data on cloud he is not ensure about his data, so we have to store his dataon the cloud by encrypting his data. This encryption of data takes place at client side and metadata of thatdata also created i.e. secure DBaaS concept. This encrypted data is stored at the cloud along with itsencrypted metadata. Then the authorized clients can access the data by using only metadata.This is the first solution supporting geographically distributed clients to connect directly to anencrypted cloud database, and to execute concurrent and independent operations including those modifyingthe database structure. The proposed system has the further advantage of eliminating intermediate proxiesthat limit the elasticity, availability, and scalability properties that are intrinsic in cloud-based solutions.SecureDBaaS provides several original features that differentiate it from previous work in the field ofsecurity for remote database services.The architecture design was motivated by a threefoldgoal: to allow multiple, independent, and geographicallydistributed clients to execute concurrent operations onencrypted data, including SQL statements that modify thedatabase structure; to preserve data confidentiality andconsistency at the client and cloud level; to eliminate anyintermediate server between the cloud client and the cloudprovider. The possibility of combining availability, elasticity,and scalability of a typical cloud DBaaS with dataconfidentiality is demonstrated through a prototype ofSecureDBaaS that supports the execution of concurrent and independent operations to the remote encrypted database from many geographically distributed clients asin any unencrypted DBaaS setup. To achieve these goals,SecureDBaaS integrates existing cryptographic schemes,isolation mechanisms, and novel strategies for managementof encrypted metadata on the untrusted cloud database. Thispaper contains a theoretical discussion about solutions fordata consistency issues due to concurrent and independentclient accesses to encrypted data. In this context, we cannotapply fully homomorphic encryption schemes [7] because oftheir excessive computational complexity.The SecureDBaaS architecture is tailored to cloudplatforms and does not introduce any intermediary proxyor broker server between the client and the cloudprovider. Eliminating any trusted intermediate serverallows SecureDBaaS to achieve the same availability,reliability, and elasticity levels of a cloud DBaaS. Otherproposals (e.g., [8], [9], [10], [11]) based on intermediateserver(s) were considered impracticable for a cloud-basedsolution because any proxy represents a single point offailure and a system bottleneck that limits the mainbenefits (e.g., scalability, availability, and elasticity) of adatabase service deployed on a cloud platform. UnlikeSecureDBaaS, architectures relying on a trusted intermediateproxy do not support the most typical cloud scenariowhere geographically dispersed clients can concurrentlyissue read/write operations and data structure modificationsto a cloud database.A large set of experiments based on real cloud platformsdemonstrate that SecureDBaaS is immediately applicable toany DBMS because it requires no modification to the clouddatabase services. Other studies where the proposedarchitecture is subject to the TPC-C standard benchmarkfor different numbers of clients and network latenciesshow that the performance of concurrent read and writeoperations not modifying the SecureDBaaS databasestructure is comparable to that of unencrypted

clouddatabase. Workloads including modifications to the databasestructure are also supported by SecureDBaaS, but atthe price of overheads that seem acceptable to achieve thedesired level of data confidentiality.

II. SYSTEM OVERVIEW

The system mainly focuses on following-

- Cloud database
- Metadata Management
- Encryption algorithm

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

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

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

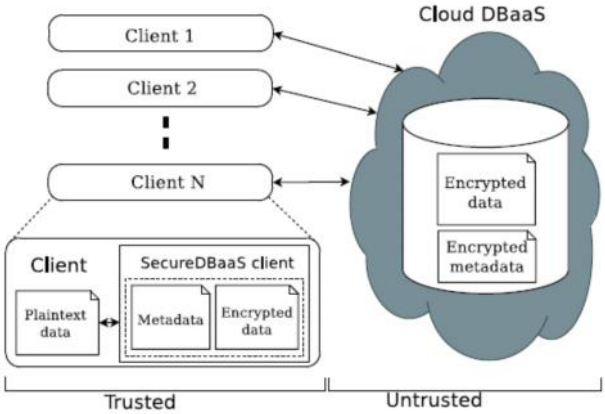

Fig 1: Secure DBaaS architecture.

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

III. SYSTEM DESIGN

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

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

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

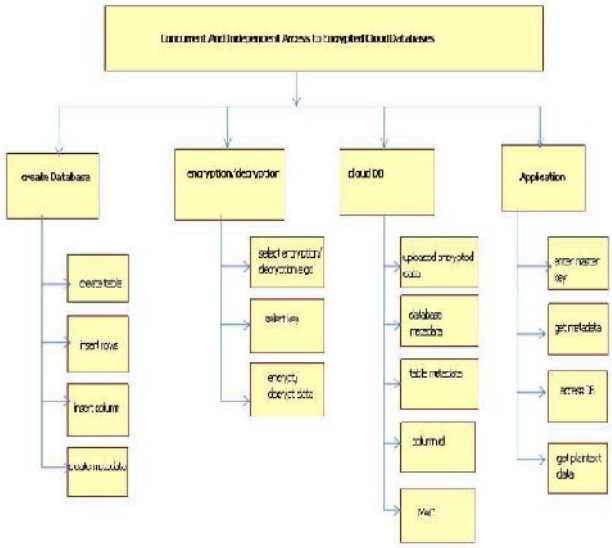

Fig 2 describes the system design having modules and its components.

1. Creation of database-

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

2. Selection of encryption and decryption algorithm -

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

3. Cloud Database-

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

4. Application-

This module contains the application of system to the cloud. How we willApply these all on cloud this module explains it. We use master key to access cloud data after data isuploaded on data. First we will get encrypted data if our key is correct then by using random decryption keyswe will get the final output in the form of plaintext data. Input is

taken from user in the form of sql query.Firstly client will create Database then, will enter rows into the database. After that the metadata of databaseis created. Then selected encryption algorithm is applied to the database and its metadata. Final output givesthe encrypted data with all its information and key used.

IV.IMPLEMENTATION

Data Management:

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

Metadata Management:

Metadata generated by SecureDBaaS contain all the information that is necessary to manage SQLstatements over the encrypted database in a way transparent to the user. Metadata management strategiesrepresent an original idea because SecureDBaaS is the first architecture storing all metadata in the untrustedcloud database together with the encrypted tenant data.SecureDBaaS uses two types of metadata.

• Database metadata are related to the whole database. There is onlyone instance of this metadata type for each database.

• Table metadata are associated with one secure table. Each table metadatacontains all information that is necessary to encrypt and decrypt data of the associated secure table.This design choice makes it possible to identify which metadata type is required to execute any SQLstatement so that a SecureDBaaS client needs to fetch only the metadata related to the secure table/s thatis/are involved in the SQL statement.This design choice minimizes the amount of metadata that each Secure DBaaS client has to fetch from the untrusted cloud database, thus reducing bandwidth consumption and processing time. Moreover, it allows multiple clients to access independently metadata related to different secure tables. Databasemetadata contain the encryption keys that are used for the secure types. A different encryption key is associated with all the possible combinations of data type and encryption type. Hence, the database metadata represent a key ring and do not contain any information about tenant data.

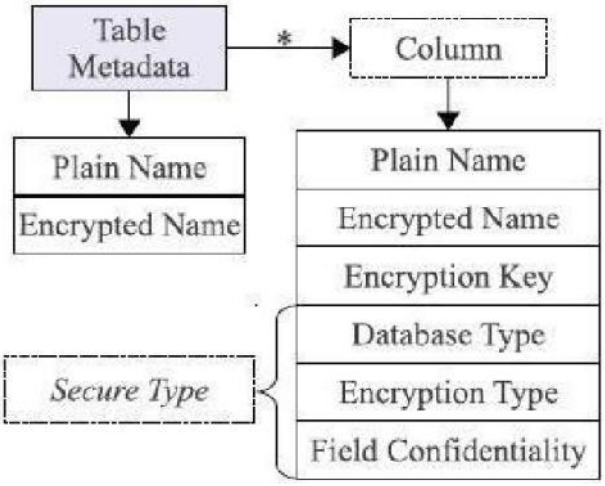

Fig.3. Structure of table metadata.

The structure of a table metadata is represented in Fig. 3. Table metadata contain the name of therelated secure table and the unencrypted name of the related plaintext table. Moreover, table metadatainclude column metadata for each column of the related secure table. Each column metadata contain thefollowing information.

• Plain name: the name of the corresponding column of the plaintext table.

• Coded name: the name of the column of the secure table. This is the only information that links acolumn to the corresponding plaintext column because column names of secure tables are randomlygenerated.

• Secure type: the secure type of the column. This allows a SecureDBaaS client to be informed aboutthe data type and the encryption policies associated with a column.

• Encryption key: the key used to encrypt and decrypt all the data stored in the column.SecureDBaaS stores metadata in the metadata storage table that is located in the untrusted cloud asthe database. This is an original choice that augmnts flexibility, but opens two novel issues in terms ofefficient data retrieval and data confidentiality. To allow SecureDBaaS clients to manipulate metadatathrough SQL statements, we save database and table metadata in a tabular form. Even metadataconfidentiality is guaranteed through encryption. The structure of the metadata storage table is shown in Fig.4 this table uses one row for the database metadata, and one row for each table metadata.Database and table metadata are encrypted through the same encryption key before being saved. Thisencryption key is called a master key. Only trusted clients that already know the master key can decrypt themetadata and acquire information that is necessary to encrypt and decrypt tenant data. Each metadata can beretrieved by clients through an associated ID, which is the primary key of the metadata storage table. ThisID is computed by applying a Message Authentication Code (MAC) function to the name of the object(database or table) described by the corresponding row. The use of a deterministic MAC function allowsclients to retrieve the metadata of a given table by knowing its plaintext name. This mechanism has thefurther benefit of allowing clients to access each metadata independently, which is an important feature inconcurrent environments. In addition, SecureDBaaS clients can use caching policies to reduce the bandwidthoverhead.

Metadata Storage Table

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

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

Algorithms:

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

V. CONCLUSION

In this paper, we have discussed concurrent and independent access to encrypted cloud databases,proposes an innovative architecture that guarantees confidentiality of data stored in public cloud databases.The proposed system will not require modifications to the cloud database, and it will be immediatelyapplicable to existing cloudDBaaS. Resolve problem of single point failure and a bottleneck limitingavailability and scalability of cloud database services.A large partof the research includes solutions to support concurrentSQL operations (including statements modifying thedatabase structure) on encrypted data issued by heterogenousand possibly geographically dispersed clients. Theproposed architecture does not require modifications tothe cloud database, and it is immediately applicableto existing cloud DBaaS, such as the experimentedPostgreSQL Plus Cloud Database [23], Windows Azure [24], and Xeround [22]. There are no theoretical andpractical limits to extend our solution to other platformsand to include new encryption algorithms.It is worth observing that experimental results based onthe TPC-C standard benchmark show that the performanceimpact of data encryption on response time becomesnegligible because it is masked by network latencies thatare typical of cloud scenarios. In particular, concurrentread and write operations that do not modify the structureof the encrypted database cause negligible overhead.Dynamic scenarios characterized by (possibly) concurrentmodifications of the database structure are supported, butat the price of high computational costs. These performanceresults open the space to future improvements thatwe are investigating.

REFERENCES

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

[2] W. Jansen and T. Grance, “Guidelines on Security and Privacy inPublic Cloud Computing,” Technical Report Special Publication800-144, 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 andImplementation, Oct. 2010.

[4] J. Li, M. Krohn, D. Mazie‘res, and D. Shasha, “Secure UntrustedData Repository (SUNDR),” Proc. Sixth USENIX Conf. OpeartingSystems Design and Implementation, Oct. 2004.

[5] P. Mahajan, S. Setty, S. Lee, A. Clement, L. Alvisi, M. Dahlin, andM. Walfish, “Depot: Cloud Storage with Minimal Trust,” ACMTrans. Computer Systems, vol. 29, no. 4, article 12, 2011.

[6] H. Hacigu¨mu¨ s¸, B. Iyer, and S. Mehrotra, “Providing Database as aService,” 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 QueryProcessing,” Proc. 23rd ACM Symp. Operating Systems Principles,Oct. 2011.

[9] H. Hacigu¨mu¨ s¸, B. Iyer, C. Li, and S. Mehrotra, “ExecutingSQL over Encrypted Data in the Database-Service-ProviderModel,” Proc. ACM SIGMOD Int’l Conf. Management Data, June2002.

[10] J. Li and E. Omiecinski, “Efficiency and Security Trade-Off inSupporting Range Queries on Encrypted Databases,” Proc. 19thAnn. 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 11.3Working Conf. Data and Applications Security, July/Aug. 2006.

[12] D. Agrawal, A.E. Abbadi, F. Emekci, and A. Metwally, “DatabaseManagement as a Service: Challenges and Opportunities,” Proc.25th IEEE Int’l Conf. Data Eng., Mar.-Apr. 2009.