INTERNATIONAL JOURNAL FOR TRENDS IN ENGINEERING & TECHNOLOGY,
VOLUME 3 ISSUE 2 – FEBRUARY 2015 – ISSN: 2349 – 9303

A Security and Privacy Measure for Encrypted Cloud Database

Krithika.s.s, Neeraja.T, and Nivedha.P,
Information Technology,
Panimalar Institute of Technology, Chennai,
krithi.s07@gmail.com,neeraja15894@gmail.com,nivedhapan@gmail.com.

Abstract-- Cloud security is an evolving domain in computer security. It refers to a set of policies, technologies and controls deployed to protect the data, applications, and the associated infrastructure of cloud computing. Existing system does not allow multiple clients to perform concurrent operation. In our proposed architecture there is threefold goal: to allow geographically distributed clients to execute concurrent operations on encrypted data independently including those modifying the DataBase structures. It also offers enhancement of file storage. Multiple clients can directly connect to the cloud server by eliminating the intermediate proxies between the cloud client and cloud server that limits the pliability, readiness and ductility properties. To provide security and privacy for client's data, the data are stored on cloud in an encrypted form. In storage as a service paradigm confidentiality has been guaranteed with several solutions, while in DataBase as a service paradigm (DBaaS) guaranteeing confidentiality is still an open research area.

Keywords- Encrypted data, Cloud DataBase, DataBase as a Service (DBaaS), Data Security.

1. INTRODUCTION

In this concept, the data given by the client that are placed in cloud storage must be in encrypted form in order to ensure the confidentiality. Only trusted parties can access the original plain data, whereas the data must be shown in encrypted format for other cloud provider, intermediaries and internet. Cloud provides different types of services. In storage as a service paradigm, confidentiality can be ensured by several solutions, while in database as a service paradigm, guaranteeing confidentiality is still an open research area. Here SecureDBaaS is used; it does not expose the unencrypted data to the cloud provider and take full advantage of the DBaaS qualities such as pliability, ductility and accessibility. It motivates threefold goal: It allows distributed clients to execute their operations to the remote encrypted database concurrently and independently including those modifying the database structure. It provides data confidentiality by eliminating intermediate server between the cloud client and cloud server. These can be achieved by integrating the existing isolation mechanism, cryptographic schemes and novel strategies for management of encrypted metadata stored in the untrusted cloud database. Unlike Secure DBaaS services, other services can rely on trusted intermediate proxy but it does not support in this context where multiple clients can concurrently issue read/write operations and including those modifying the database structure to a cloud database. The main advantage is that SecureDBaaS can applicable to any DBMS but it requires no modification to the cloud database services. In the proposed architecture TPC-C standard benchmark are subjected to show the performance of concurrent read and write operations, network latencies and the number of clients. The motivation of these contexts is that network latencies, it tends to mask the performance costs of data encryption on response time.

EXISTING SYSTEM:

In the existing system some confidentiality has been guaranteed by different approaches by distributing data among different cloud providers and by using secret sharing. In this way, it prevents on cloud provider to read its portion of data, but that information can be reconstructed by colluding cloud providers. Here it is possible to execute range queries on data that represents only one client can execute their operations at that time, others have to wait until the operations to be finished. Transparent Data Encryption feature is used to encrypting the data at the file system. It can be offered by some database engine. This feature makes it possible to build a trusted DBMS over untrusted storage. This approach is not applicable to the DBaaS context considered by SecureDBaaS, because we come to a conclusion that the cloud provider is untrusted.

PROPOSED SYSTEM:

In our proposed architecture, the data which are stored in the cloud comes with a guarantee of providing security and privacy that are given by the client. There are threefold goal: it allows geographically distributed client to execute their operations on encrypted cloud database in any unencrypted DBaaS concurrently and independently including those modifying the database structure. Then the cloud client can directly connect to the cloud server without including any intermediate server between them, which limits the pliability, ductility and accessibility. Multiple client can execute the concurrent operation on encrypted cloud database and the original plain data has been shown by the trusted parties. These goals have been achieved by integrating the existing cryptographic methods, isolation mechanisms and novel strategies for management of encrypted metadata on the untrusted cloud database.

2 RELATED WORKS

SecureDBaaS provides several original features that differentiate it from previous work in the field of security for remote database services. It guarantees data confidentiality by allowing a cloud database server to execute concurrent SQL operations on the encrypted data. It does not require any intermediate server. Response times are affected through cryptographic overheads that for most SQL operations that are masked by network latencies. Multiple client scan access concurrently and independently a cloud database service. It does not require a trusted proxy because tenant data and metadata stored by the cloud database are always encrypted. It is compatible with the most popular relational database servers. Cryptographic file systems and secure storage solutions represent the earliest works in this field. Different approaches guarantee some confidentiality. They prevent one cloud provider to read itsportion of data, but information can be reconstructed only bythe cloud providers. It is possible to execute range queries on data. SecureDBaaS differs from these solutions as it does not require the use of multiple cloud providers in execution of most common SQL operations on encrypted data. SecureDBaaS relates more closely to works using encryption to protect the data that are managed by untrusted databases. DBMS can only execute SQL operations over plain text data. Some DBMS engines offer the possibility of encrypting data at the file system level through the so-called Transparent Data Encryption feature. This makes it possible in building a trusted DBMS over untrusted storage. It is not applicable to the DBaaS context considered by SecureDBaas, because we think that the cloud provider is untrusted. As on this the trusted solution is in the encrypted format. However, the architecture of these solutions is based on an intermediate and trusted proxy that mediates any interaction between each client and the untrusted DBMS server. The approach proposed in by the authors of the DBaaS model works by encrypting blocks of data. Whenever the data is required the trusted proxy needs to retrieve the whole block, to decrypt it, and to filter out unnecessary data that belong to the same block. They share the same proxy-based architecture and its intrinsic issues. On the other hand, SecureDBaaS allows the execution of operations over encrypted data through SQL-aware encryption algorithms. Through this technique, initially proposed in Crypt DB, but makes it possible to execute operations over encrypted data that are similar to operations over plain text data.

3. ARCHITECTURE DESIGN

SecureDBaaS is created to allow multiple and independent Clients to connect directly to the untrusted cloud DBaaS without any intermediate server. This client allows a user to connect to the cloud DBaaS for administering it, that is to read and write data, modify the database tables once it is created. The data that are been presented are in the plain text .the data and metadata must be encrypted before exiting them. The data obtained from client is managed by SecureDBaaS includes plaintext data, encrypted data, metadata, and encrypted metadata. Plaintext data are those that consist of information that a tenant wants to store and process remotely in the cloud DBaaS. On Encryption the data and the meta data must be encrypted so that SecureDBaaS clients produce also a set of metadata consisting information required to encrypt and decrypt data.Even metadata are encrypted and stored in the cloud DBaaS.

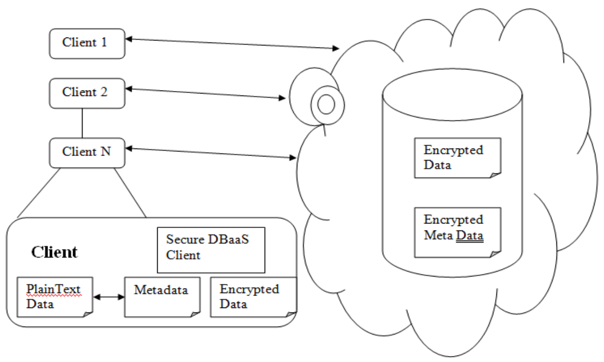


Fig. 1. SecureDBaaS architecture

The information that are managed by SecureDBaaS includes plain text data, metadata, are encrypted to data and metadata. Plaintext data consist of information that a tenant wants to store and process remotely in the cloud DBaaS. Top r event the data in plain text format we need to encrypt the data .Even metadata are encrypted and stored in the cloud DBaaS. The information managed by SecureDBaaS that plaintext data and encrypted metadata. Plaintext data consist of data that a tenant wants to store and process in the cloud DBaaS.

3.1 Data Management

Tenant data are stored in relational database. In order to provide data confidentiality both the data and their Meta need to be encrypted. Encrypted data are stored through secure tables into the cloud database. Transformation of each plaintext to secure table is done by encrypting the data using Encryption algorithm and key that are known to all the SecureDBaaS clients. Since the column names are generated randomly by the Secure DBaaS. Secure tables have unique column name even though different Plaintext tables have similar column name. The main purpose of this data management is that secureDBaaS extends the concept of secure type for each of the data type in the table. A secure type is of three fields. They are:

Data type, encryption type, field confidentiality. **Data type** is nothing but the data in the plain text. **Encryption type** is nothing but the encryption algorithm that are use to change the data to cipher text. **Field confidentiality** is that used to define the columns of the secure table.

3.2 Metadata Management

It is generated by SecureDBaaS that contains the information that are necessary to manage SQL statements over Encrypted database. Metadata management represents storing all metadata in the untrusted cloud database together with the encrypted tenant data. Meta data is nothing but data about data. SecureDBaaS uses two types of metadata:

Database Metadata, Table Metadata**. Database Metadata** is that represent the whole database. **Table Metadata** is that which has table that are necessary to encrypt and decrypt the data of a secure table.

Plain name: The name of the column in the plaintext table.

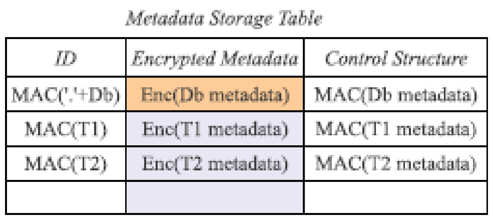
Coded name: the name of the column in the secure table.

Secure type: it s that SecureDBaaS client needs to be

Informed on the data type and the encryption type.

Encryption key: it is the key used to encrypt and decrypt

the data.



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.

4 OPERATIONS

In this section, we outline the setup setting operations carried out by a database administrator (DBA), and we describe the execution of SQL operations on encrypted data

4.1 Setup Phase

It stores them in the metadata storage table after encryption through the master key. The DBAdistributes the master key to the legitimate users. User access control policies are administrated by the DBA.In the following steps, the DBA create the tables of the encrypted database. It must consider the three field confidentiality attributes (COL, MCOL, and DBC) .Let us describe this phase by referring to a simple but representative example shown in Fig. 4, where we have three secure tables named ST1,ST2, and ST3. Each table STi (i ¼ 1; 2; 3) includes an encrypted table Ti that contains encrypted tenant data, and a table metadata Mi. (Although, in reality, the names of the columns of the secure tables are randomly generated; forthe sake of simplicity, this figure refers to them throughC1-CN.)

For example, if the database has to support a join statement among the values of T1.C2 and T2.C1, the DBA must use the MCOL field confidentiality for T2.C1 that references T1.C2 (solid arrow). In such a way, SecureDBaaS can retrieve the encryption key specified in the column data of T1.C2 from the metadata table M1 and can use the same key for T2.C1. The solid arrow from M2 to M1denotes that they explicitly share the encryption algorithm and the key.

ALGORITHM

DES with ECB Algorithm

Data Encryption Standard	-	DES
Electronic Codebook mode	-	ECB

DES:

Input	:	64bit PlainText
Output	:	64 bit CipherText

DES expects two inputs - the plaintext to be encrypted and the secret key.

DES performs an initial permutation on the entire 64 bit block of data. It is then split into 2, 32 bit sub-blocks, Li and Ri which are then passed into what is known as a of which there are 16 (the subscript i in Li and Ri indicates the current round). Each of the rounds are identical and the effects of increasing their number is twofold - the algorithms security is increased and its temporal efficiency decreased. Clearly these are two conflicting outcomes and a compromise must be made. For DES the number chosen was 16, probably to guarantee the elimination of any correlation between the cipher text and either the plaintext or key6.

At the end of the16th round, the 32 bit Li and Ri output quantities are swapped to create what is known as the pre-output. This [R16, L16] concatenation is permuted using a function which is the exact inverse of the initial permutation. The output of this final permutation is the 64 bit cipher text.

Electronic Codebook mode

Electronic codebook mode is the most obvious way to use a block cipher.

Enciphering.

Input:

k-bit key K

n-bit plaintext blocks M = M1M2 . . . Mt

Algorithm:

Cj = EK(Mj).

Output:

n-bitciphertext blocks C = C1C2 . . . Ct

Deciphering.

Input:

k-bit key K

n-bitciphertext blocks C = C1C2 . . . Ct

Algorithm:

Mj = DK(Cj).

Output:

n-bit plaintext blocks M = M1M2 . . . Mt

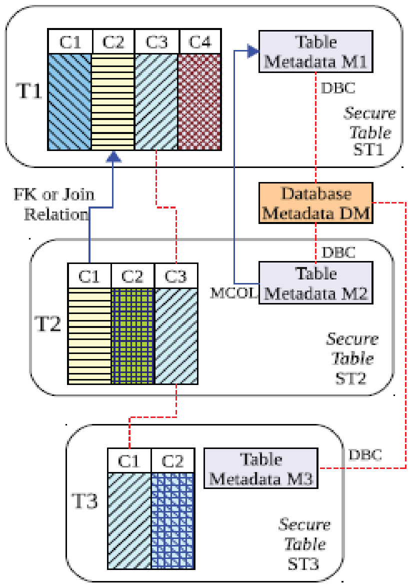


Fig. 4. Management of the encryption keys according to the field confidentiality parameter.

4.2 Concurrent SQL Operations

In order to support the concurrent SQL statements issued by multiple clients. in our architecture we guarantee consistency in encrypted data and metadata. SQL operations does not cause any modifications in the database structure. In an encrypted cloud database the clients can operate through SQL commands such that meta data and the data can be read once and they can be retrieved many times from the cache.

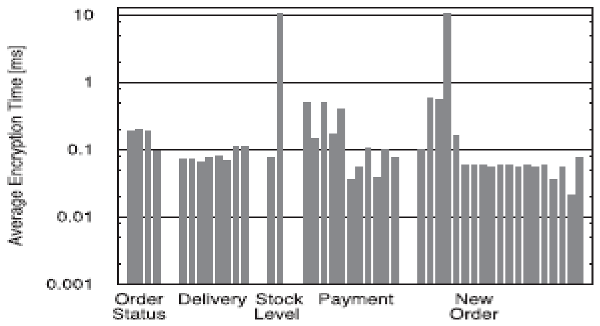


Fig. 5. Encryption times of TPC-C benchmark operations grouped by the transaction class.

5 EXPERIMENTAL RESULTS

We demonstrate the applicability of SecureDBaaS to different cloud DBaaS solutions by implementing and handling encrypted database operations on emulated and real cloud infrastructures. The present version of the SecureDBaaS prototype supports PostgreSQL, MySql, andSQL Server relational databases. As a first result, we canobserve that porting SecureDBaaS to different DBMSrequired minor changes related to the database connector, and minimal modifications As the workload model for the database, we refer to the TPC-C benchmark. The DBMS server is PostgreSQL9.1deployed on a quad-core Xeon having 12 GB of RAM. Clients are connected to the server through a LAN, where we can introduce arbitrary network latencies to emulate WAN connections that are typical of cloud services. The experiments evaluate the overhead of encryption, compare the response times of plain versus encrypted database operations, and analyze the impact of network latency. We consider two TPC-C compliant databases with10 warehouses that contain the same number of tuples:plain tuples consist of 1,046 MB data, while SecureDBaaStuples have size equal to 2,615 MB because of encryption over head. Both databases use repeatable read (snapshot) isolation level

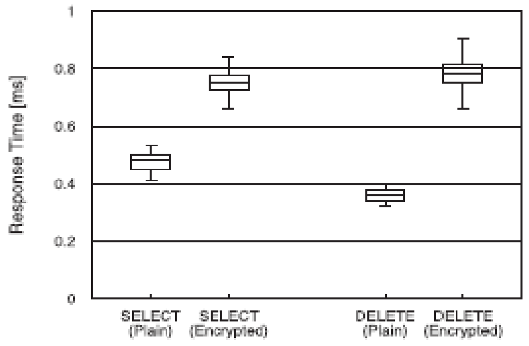


Fig. 6. Plain versus encrypted SELECT and DELETE operations.

To evaluate encryption costs, the client measures theexecution time of the 44 SQL commands of the TPC-Cbenchmark. Encryption times are reported in the histogram
of the Fig. 5 that has a logarithmic Y -axis. TPC-Cperationsare grouped on the basis of the class of transaction: OrderStatus, Delivery, Stock Level, Payment, and New Order. From this figure, we can appreciate that the encryption time is below 0.1 ms for the majority of operations, and below1 ms for almost all operations but two. The exceptions arerepresented by two operations of the Stock Level andPayment transactions where the encryption time is twoorders of magnitude higher. This high overhead is caused by the use of the order preserving encryption that is necessary for range queries.The last set of experiments assess the performance of SecureDBaaS in realistic cloud database scenarios, as well as its ability to support multiple, distributed, and independent clients. The test bed is similar to that described previously, but now the runs are repeated by varying the number of concurrent clients (from 1 to 40) and the network latencies (from plain LAN to delays reaching 150 ms). All clients execute concurrently the benchmark for 300 seconds. The results in terms of throughput refer to three types of database operations:

. Original TPC-C: the standard TPC-C benchmark; Plain-SecureDBaaS: SecureDBaaS that use plain encryption,
that is, all SecureDBaaS functions and data structures with no encryption; it allows us to evaluate the overhead of SecureDBaaS without the cost of cryptographic operations;

. SecureDBaaS: SecureDBaaS referring to the highest confidentiality level. Fig. 8 shows the system throughput referring to 20 clients issuing requests to SecureDBaaS as a function of the network latency. The Y -axis reports the number of committed transactions per minute during the entire experiment. This figure shows two important results: .if we exclude the cryptographic costs, SecureDBaaS does not introduce significant overheads. This can be appreciated by verifying that the throughput of plain SecureDBaaS and original TPC-C overlies for any realistic Internet delay (>20 ms); . as expected, the number of transactions per minute executed by SecureDBaaS is lower than those referring to original TPC-C and plain-SecureDBaaS, but the difference rapidly decreases as the network latency increases to the extent that is almost nullified in any network scenario that can be realisticallyreferred to a cloud database context

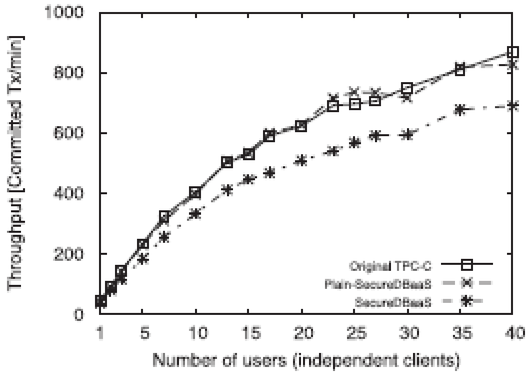


Fig. 9. TPC-C performance (latency equal to 40 ms).

40 ms and 80 ms network latencies, respectively. These measures are optimistic representations of continental and intercontinental delays. The Y -axis represents the numberof committed TPC-C transactions per minute executed bythe clients. The trends of the SecureDBaaS lines are close to those of the original TPC-C database, thus demonstrating that SecureDBaaS encrypted database does not affect scalability with respect to the plain database. Even more important, the network latencies tend to mask cryptographic overheads for any number of clients. For example, the overheads of SecureDBaaS with 40 concurrent clients decreases from 20 percent in a 40-ms scenario to 13 percent in a realistic scenario, where the client-server latency is equal to 80 ms. This result is important because it confirms that SecureDBaaS is a valid and practical solution for guaranteeing.

6 CONCLUSIONS

We propose an innovative architecture that guarantees confidentiality of data stored in public cloud databases. our solution does no trely on an intermediate proxy that we consider a single point of failure and a bottleneck limiting availability and scalability of typical cloud database services. It includes solutions to support concurrentSQL operations on encrypted data issued by geographically distributed clients. The proposed architecture does not require modifications to the cloud database, and it is applicable to existing cloud DBaaS, it includes new encryption algorithms. On observing that experimental results based onthe TPC-C standard benchmark show that the performanceimpact of data encryption on response time becomes negligible because it is masked by network latencies that are typical of cloud scenarios. In this, concurrent read and write operations that do not modify the structure of the encrypted database characterized by concurrent modifications of the database structure are supported, but the price is of high computational costs.

ACKNOWLEDGMENTS

We would like to thank **Ms.R.Dharani**, Assistant Professor-1 of the Panimlar Institute of Technology for her constructive comments on preliminary versions of this paper.

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
800-144, 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, June
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 the
Database-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 InformationSoc., Mar. 2011.

AUTHOR PROFILE:

KRITHIKA.S.S is a final year student ofInformation technology; Panimalar institute of technology. She has attended several national level symposiums. Area of interest are cloud computing, Real time systems, Web designing. An aspiring Master'sCandidate, she is currently working on RealTime software projects.

NEERAJA.T is a final year student of Information technology; Panimalar institute of technology. She has presented 1 paper in national conference and attended several national level symposiums. Area of interest are cloud computing, Web designing. An aspiring Master's Candidate, she is currently working on Real Time software projects.

NIVEDHA.P is a final year student of Information technology; Panimalar institute of technology. She has attended several national level symposiums. Area of interest are cloud computing, Real time systems, Web designing. An aspiring Master's Candidate, she is currently working on Real Time software projects