International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

Distributed, Concurrent, and Independent Access to Encrypted Cloud Databases

Praveenkumar[1], Jyoti Patil[2]

[1]M.Tech Department of Computer Science and Engineering, Poojya Doddappa Appa College of Engineering Gulbarga, Karnataka, India

[2]Associate Professor Department of Computer Science and Engineering, Poojya Doddappa Appa College of Engineering Gulbarga, Karnataka, India

Abstract: ***Placing critical data in the hands of a cloud provider should come with the guarantee of security and availability for data at rest, in motion, and in use. Several alternatives exist for storage services, while data confidentiality solutions for the database as a service paradigm are still immature. The proposed novel architecture 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. The efficacy of the proposed architecture is evaluated through theoretical analyses and extensive experimental results based on a prototype implementation subject to the TPC-Standard benchmark for different numbers of clients and network latencies.***

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

1. Introduction

In a cloud context, where critical information is placed on infrastructures of untrusted third parties ensuring data confidentiality is of paramount importance. This requirement imposes clear data management choices: original plain data must be accessible only by trust expertise that does 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. There are several solutions ensuring confidentiality for the storage as a service paradigm while guaranteeing confidentiality in the database as a service (DBaaS) paradigm is still an open research area. This context proposes Secure DBaaS as the novel 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 a threefold goal: to allow multiple, independent, and geographically distributed clients to execute concurrent operations on encrypted data, including SQL statements that modify the database structure; to preserve data confidentiality and consistency at the client and cloud level; to eliminate any intermediate server between the cloud client and the cloud provider. The possibility of combining availability, elasticity, and scalability of a typical cloud DBaaS with data confidentiality is demonstrated through a prototype of Secure DBaaS that supports the execution of concurrent and independent operations to the remote encrypted database from many geographically distributed clients as in any unencrypted DBaaS setup. To achieve these goals, Secure DBaaS integrates existing cryptographic schemes, isolation mechanisms, and novel strategies for management of encrypted metadata on the untrusted cloud database.

This paper contains a theoretical discussion about solutions for data consistency issues due to concurrent and independent client accesses to encrypted data. In this context, it does not apply fully homomorphic encryption schemes because of their excessive computational complexity. The SecureDBaaS architecture is tailored to cloud platforms and does not introduce any intermediary proxy or broker server between the client and the cloud provider Workloads including modifications to the database structure are also supported by Secure DBaaS, but at the price of overheads that seem acceptable to achieve the desired level of data confidentiality. The motivation of these results is that network latencies, which are typical of cloud scenarios, tend to mask the performance costs of data encryption on response time.

This paper is organized as follows, section 1 discusses the introduction, and section 2 describes related work. Section 3 details the system design and implementation. Section 4 presents the performance evaluations of our system design. Finally, section 5 presents some concluding remark.

2. Related Work

" A View of Cloud Computing" M. Armbrust [1], has developed with innovative ideas for new Internet services no longer require the large capital outlays in hardware to deploy their service or the human expense to operate it. Cloud Computing will grow, so developers should take it into account. Moreover: 1. Applications Software needs to both scale down rapidly as well as scale up, which is a new requirement. Such software also needs a pay-for-use licensing model to match needs of Cloud Computing.

2. Infrastructure Software needs to be aware that it is no longer running on bare metal but on VMs. Moreover, billing needs to build in from the start. 3. Hardware Systems should be designed at the scale of a container (at least a dozen racks), which will be is the minimum purchase size. **"SPORC: Group Collaboration Using Untrusted Cloud**

Resources" A.J. Feldman, W.P. Zeller, M.J. Freedman, and E.W. Felten [2], have described Cloud-based services are an attractive deployment model for user-facing applications like word processing and calendaring. In SPORC, a server observes only encrypted data and cannot deviate from correct execution without being detected. SPORC allows concurrent, low-latency editing of shared state, permits disconnected operation, and supports dynamic access control even in the presence of concurrency Acknowledgments. **"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.

3. 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.

3.1 Proposed System

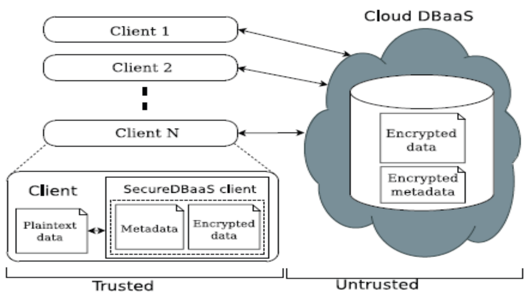


Figure 1: Architecture of Secure DBaaS

3.2 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 through

a 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.

3.3 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.

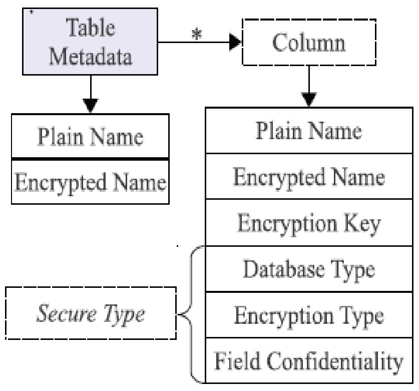


Figure 2: Structure of table metadata

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

1) **Coded name**: the name of the column of the secure table. This is the only information that links a column to the corresponding plaintext column because column names of secure tables are randomly generated.
2) **Secure type**: This allows a SecureDBaaS client to be informed about the data type and the encryption policies associated with a column.
3) **Encryption key**: the key used to encrypt and decrypt all the data stored in the column.

3.4 Data Flow Diagram

Data flow diagrams illustrate how data is processed by a system in terms of inputs and outputs. Data flow diagrams can be used to provide a clear representation of any business Function. The technique starts with an overall picture of the business and continues by analyzing each of the functional areas of interest. This analysis can be carried out in precisely the level of detail required. The technique exploits a method called top-down expansion to conduct the analysis in a targeted way. As the name suggests, Data Flow Diagram (DFD) is an illustration that explicates the passage of information in a process. A DFD can be easily drawn using simple symbols. Additionally, complicated processes can be easily automated by creating DFDs using easy-to-use, free downloadable diagramming tools. A DFD is a model for constructing and analyzing information processes. DFD illustrates the flow of information in a process depending upon the inputs and outputs. A DFD can also be referred to as a Process Model. A DFD demonstrates business or technical process with the support of the outside data saved, plus the data flowing from the process to another and the end results.

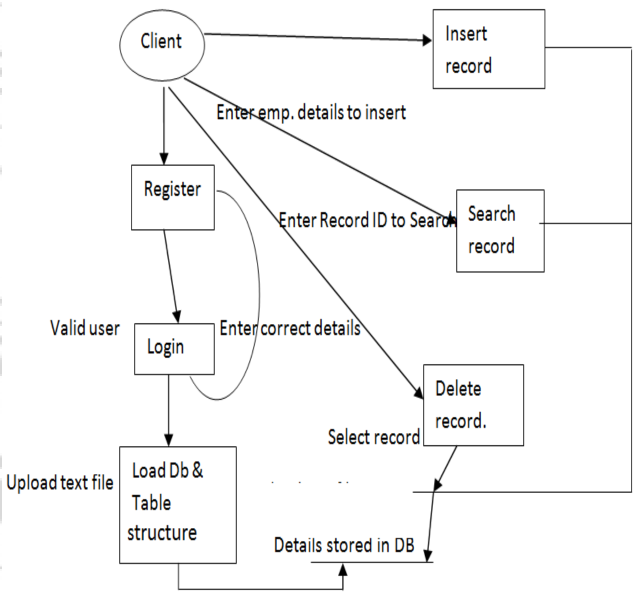
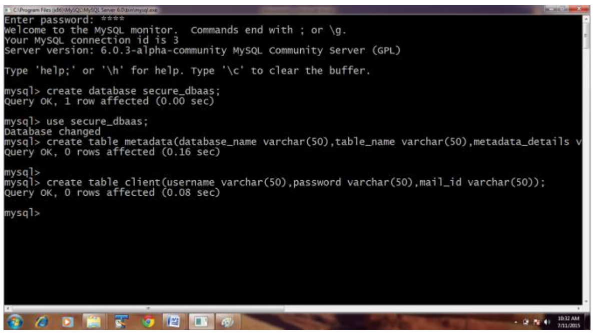


Figure 3: Data flow diagram.

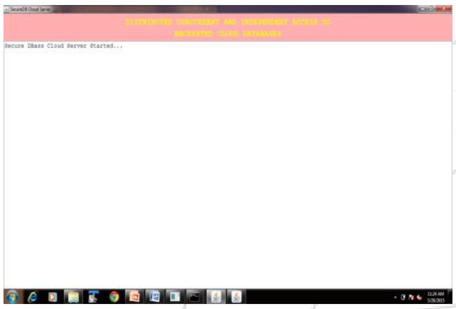
4. Results

This paper demonstrates the applicability of SecureDBaaS to different cloud DBaaS solutions by implementing and handling encrypted database operations on emulated real cloud infrastructures. The present version of the SecureDBaaS prototype supports Postgre SQL, MySQL, and SQL Server relational databases. As a first result, observe that porting Secure DBaaS to different DBMS required minor changes related to the database connector, and minimal modifications of the codebase.



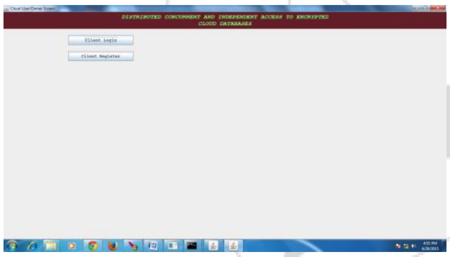
```
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3
Server version: 6.0.3-alpha-community MySQL Community Server (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> create database secure_dbaas;
Query OK, 1 row affected (0.00 sec)

mysql> use secure_dbaas;
Database changed
mysql> create table metadata(database_name varchar(50),table_name varchar(50),metadata_details v
Query OK, 0 rows affected (0.16 sec)

mysql>
mysql> create table client(username varchar(50),password varchar(50),mail_id varchar(50));
Query OK, 0 rows affected (0.08 sec)

mysql>
```

Initially create database using command "create *database database_name;"* and connect to this database using command "*use database_ name;"* and create metadata to store the information about tables. Run the cloud server which pops up the following window indicating server is started.

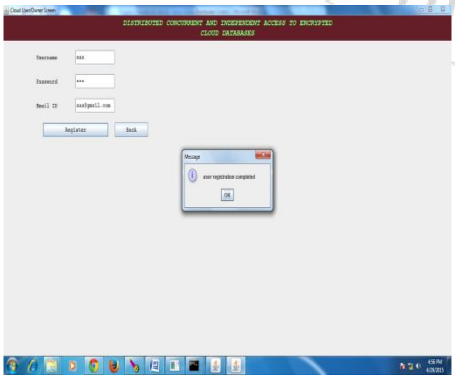

DISTRIBUTED CONCURRENT AND INDEPENDENT ACCESS TO ENCRYPTED CLOUD DATABASES

Secure DBaas Cloud Server Started...

After running the cloud, start the client-side which pops up the following window


DISTRIBUTED CONCURRENT AND INDEPENDENT ACCESS TO ENCRYPTED CLOUD DATABASES
Client Login
Client Register

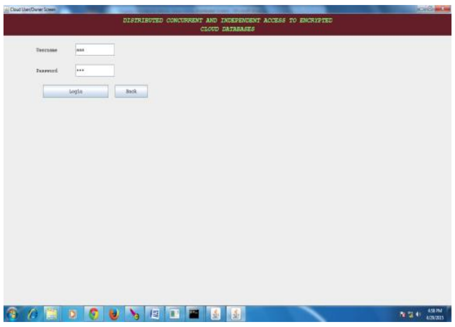
Before using the cloud service you need to register by providing user name, password and email id.


DISTRIBUTED CONCURRENT AND INDEPENDENT ACCESS TO ENCRYPTED CLOUD DATABASES
Register
Back
OK

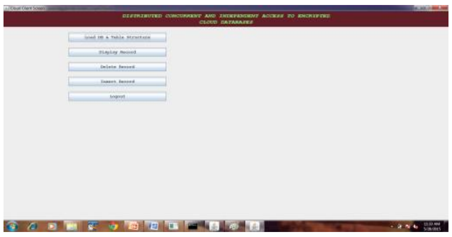
Username	xxx
Password	***
Email ID	xxx@gmail.com

user registration completed

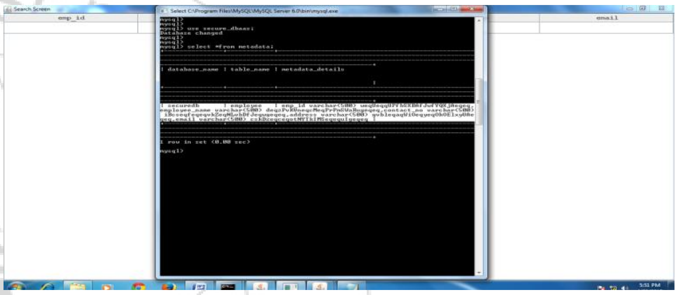
After successful registration you can login using valid email_id and password to access the cloud services.


DISTRIBUTED CONCURRENT AND INDEPENDENT ACCESS TO ENCRYPTED CLOUD DATABASES
Login
Back

Username	xxx
Password	***


DISTRIBUTED CONCURRENT AND INDEPENDENT ACCESS TO ENCRYPTED CLOUD DATABASES
Load DB & Table Structure
Display Record
Delete Record
Insert Record
Logout

After successful log-in, cloud provides the above services.



```
mysql>
mysql>
mysql> use secure_dbaas;
Database changed
mysql>
mysql>
mysql> select *from metadata;
| database_name | table_name | metadata_details |
| securedb      | employee   | emp_id varchar(500) [illegible],
employee_name varchar(500) [illegible],contact_no varchar(500)
[illegible],address varchar(500) [illegible]
[illegible],email varchar(500) [illegible] |
1 row in set (0.00 sec)

mysql>
```

The user data stored in encrypted format to protect the critical data and it is decrypted and given to authorized user.

5. Conclusion

This paper proposes an innovative architecture that guarantees confidentiality of data stored in public cloud databases. Proposed solution does not rely on an intermediate proxy that is considered a single point of failure and a bottleneck limiting availability and scalability of typical cloud database services. A large part of the research includes solutions to support concurrent SQL operations (including statements modifying the database structure) on encrypted data issued by heterogeneous and possibly geographically dispersed clients. The proposed architecture does not require modifications to the cloud database, and it is immediately applicable to existing cloud DBaaS such as the experimented PostgreSQL plus Cloud Database, Windows Azure, and Expound. There are no theoretical and practical limits to extend our solution to other platforms and to include new encryption algorithms. It is worth serving 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 overhead. Dynamic scenarios characterized by (possibly) concurrent modifications of the database structure are supported, but at the price of high computational costs.

These performance results open the space to future improvements that we are investigating.

References

[1] M. Armbrust et al., “A View of Cloud Computing,” Comm. of the ACM, vol. 53, no. 4, pp. 50-58, 2010.
[2] 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.
[3] J. Li, M. Krohn, D. Mazie` res, and D. Shasha, “Secure Untrusted Data Repository (SUNDR),” Proc. Sixth USENIX Conf. Operating Systems Design and Implementation, Oct. 2004.
[4] P. Mahajan, S. Setty, S. Lee, A. Clement, L. Alvisi, M. Dahlin, and M. Walfish, “Depot: Cloud Storage with Minimal Trust”, ACMTrans. Computer Systems, vol. 29, no. 4, article12, 2011.
[5] H. Hacigu¨ mu¨ s¸, B. Iyer, and S. Mehrotra, “Providing Database as a Service”, Proc. 18th IEEE Int’l Conf. Data Eng.Feb.2002.
[6] C. Gentry, “Fully Homomorphic Encryption Using Ideal Lattices”, Proc. 41st Ann. ACM Symp. Theory of Computing May 2009.
[7] 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.
[8] 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.
[9] 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.
[10] A. Shamir, “How to Share a Secret”, Comm. of the ACM, vol. 22, no. 11, pp. 612-613, 1979.