A NEW APPROACH FOR SECURELY SHARING DATA BETWEEN CLOUD USERS WITH DUAL KEYS

Varsha K N[1] and J. Bhuvana[2]

[1]Research Scholar, School of Computer Science and Information Technology, JAIN (Deemed to be University), Bangalore, India

[2]Associate Professor, School of Computer Science and Information Technology, JAIN (Deemed to be University), Bangalore, India

ABSTRACT

Cloud storage is an application of clouds that liberates organizations from establishing in-house data storage systems. However, cloud storage gives rise to security concerns. In case of group-shared data, the data face both cloud-specific and conventional insider threats. Secure data sharing among a group that counters insider threats of legitimate, yet malicious users is an important research issue. In this paper, we propose the Secure Data Sharing in Clouds (SeDaSC) methodology that provides: 1) data confidentiality and integrity; 2) access control; 3) data sharing (forwarding) without using compute-intensive re-encryption; 4) insider threat security; and 5) forward and backward access control.

1. INTRODUCTION

Cloud computing is rapidly emerging due to the provisioning of elastic, flexible, and on demand storage and computing services for customers. The data are usually encrypted before storing to the cloud. The access control, key management, encryption, and decryption processes are handled by the customers to ensure data security. A single key shared between all group members will result in the access of past data to a newly joining member. The aforesaid situation violates the confidentiality and the principle of least privilege. A separate key for every user is a cumbersome solution. The data must be separately encrypted for every user in such a scenario. The changes in the data require the decryption of all the copies of the users and encryption again with the modified contents. a methodology named Secure Data Sharing in Clouds (SeDaSC) that deals with the above-mentioned security requirements of shared group data within the cloud. The SeDaSC methodology works with three entities as follows: 1) users; 2) a cryptographic server (CS); and 3) the cloud. The data are decrypted and sent back to the user. For a newly joining member, the two portions of the key are generated, and the user is added to the ACL. For a departing member, the record is deleted from the ACL

2. LITERATURE SURVEY

All the existing fully homomorphic encryption schemes are based on three different problems, namely bounded distance decoding problem over ideal lattice, approximate greatest common divisor problem over integers and learning with error problem. In this paper, we unify the first two families of problems by introducing a new class of problems, which can be reduced from both problems. Based on this new problem, namely the bounded distance decoding over hidden ideal lattice, we present a new fully homomorphic encryption scheme. Since it is a combination of the two problems to some extent, the performance of our scheme lies between the ideal lattice-based schemes and the integer-based schemes. Furthermore, we also show a lower and upper bound of the problem our scheme is based on. As a result, we present a security conjecture. Assuming this security conjecture holds, we can incorporate smaller parameters, which will result in a scheme that is more efficient than both lattices based and integer-based schemes.[1]

In this paper, we explore 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. Such a model alleviates the need for organizations to purchase expensive hardware and software, deal with software upgrades, and hire professionals for administrative and maintenance tasks which are taken over by the service provider. We have developed and deployed a database service on the Internet, called NetDB2, which is in constant use. In a sense, data management model supported by NetDB2 provides an effective mechanism for organizations to purchase data management as a service, thereby freeing them to concentrate on their core businesses. Among the primary challenges introduced by "database as a service" are additional overhead of remote access to data, an infrastructure to guarantee data privacy, and user interface design for such a service. [2]

Online applications are vulnerable to theft of sensitive information because adversaries can exploit software bugs to gain access to private data, and because curious or malicious administrators may capture and leak data. Crypt DB 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. Crypt DB can also chain encryption keys to user passwords, so that a data item can be decrypted only by using the password of one of the users with access to that data. As a result, a database administrator never gets access to decrypted data, and even if all servers are compromised, an adversary cannot decrypt the data of any user who is not logged in. [3]

3. EXISTING SYSTEM

The Secured BaaS architecture is tailored to cloud platforms and does not introduce any intermediary proxy or broker server between the client and the cloud provider. Secured BaaS relates more closely to works using encryption to protect data managed by UN trusted databases. In such a case, a main issue to address is that cryptographic techniques cannot be natively applied to standard DBaaS. As expected, the number of transactions per minute executed by Secured BaaS is lower than those referring to original TPC-C and plain-Secured BaaS.Secured BaaS moves away from existing architectures that store just tenant data in the cloud database and save metadata in the client machine or split metadata between the cloud database and a trusted proxy. When considering scenarios where multiple clients can access the same database concurrently.

3.1 Disadvantages of the existing system

- Even though they are using secure DBaaS means Distributing data among different providers and it give more secure, but its functions cannot be taking advantage of secret sharing outsourced to an un trusted cloud provider
- It Cannot Store them in encrypted format
- When considering scenarios where multiple clients can access the same database concurrently

4. PROPOSED SYSTEM

The proposed architecture is subject to the TPC-C standard benchmark for different numbers of clients and network latencies show that the performance of concurrent read and write operations not modifying the Secured BaaS database structure are comparable to that of unencrypted cloud Database. Even metadata confidentiality is guaranteed through encryption. This table uses one row for the database metadata, and one row for each table metadata. This encryption key is called a master key.
Only trusted clients that already know the master key can decrypt the metadata and acquire information that is necessary to encrypt and decrypt tenant data. Each metadata can be retrieved by clients through an associated Id. This Id is computed by applying a Message Authentication Code (MAC) function to the name of the object described by the corresponding row. whatever user send the data our DB (Data Base) to cloud server in this process first our data base meta data split the data is data type and encrypted type, field confidentiality (size, time, path) these things are trusted proxy. After these data send to cloud server and it is un-trusted DB (data base).

4.1 Advantage of Proposed System

- To improve good Quality of Service (QoS)
- Distributing data among different providers and taking advantage of secret sharing
- Every user having the own master key

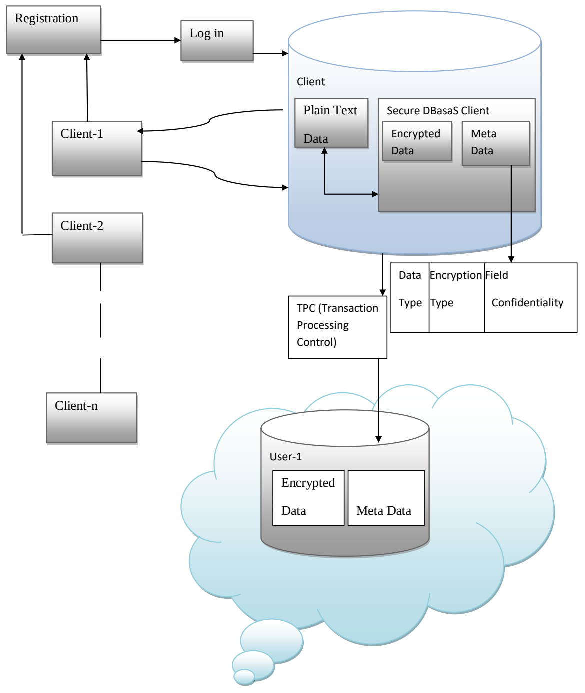

Fig 4: System Architecture

The proposed architecture is subject to the TPC-C standard benchmark for different numbers of clients and network latencies show that the performance of concurrent read and write operations not modifying the Secured BaaS database structure are comparable to that of unencrypted cloud Database. Even metadata confidentiality is guaranteed through encryption. This table uses one row for the database metadata, and one row for each table metadata. This encryption key is called a master key. Only trusted clients that already know the master key can decrypt the metadata and acquire information that is necessary to encrypt and decrypt tenant data. Each metadata can be retrieved by clients through an associated I'd. This ID is computed by applying a Message Authentication Code (MAC) function to the name of the object described by the corresponding row. whatever user send the data our DB (Data Base) to cloud server in this process first our data base meta data split the data is data type and encrypted type, field confidentiality (size, time, path) these things are trusted proxy. After these data send to cloud server and it is un-trusted DB (data base).

5. ACKNOWLEDGEMENT

I should convey my real tendency and obligation to Dr M N Nachappa and Dr. J. Bhuvana undertaking facilitators for their effective steerage and consistent inspirations all through my assessment work. Their ideal bearing, absolute co-action and second discernment have made my work gainful.

7. REFERENCES

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

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

[3]. 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.

[4]. W. Jansen and T. Grance, "Guidelines on Security and Privacy in Public Cloud Computing," Technical Report Special Publication 800-144, NIST, 2011.

[5]. 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.

[6]. 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.