Improving Database Security in Cloud Computing

Mrunali Sangar[1], Pallavi Patil [2], Priyanka Soude[3], Shambala Mali [4] ,Swati Shelake [5] B.A.Kelkar [6]

[1,2,3,4,5,6]**Computer Science and Engineering, Sanjay Ghodawat Group of Institutions,Atigre.(India)**

***Abstract*— Cloud computing is a technology that promotes various configurable resources in which the data is stored and managed in a decentralized manner. However, since the data is out of the owner's control, concerns appear regarding data confidentiality. Encryption techniques have previously been proposed to provide users with confidentiality in terms of outsourced storage; however, many of these encryption algorithms are weak, enabling data security to be imposed simply by an algorithm. We propose a combination of encryption algorithms and a distribution system to increase database confidentiality. This scheme divides a database across the cloud based on the level of security provided by the encryption algorithm utilized. We compare the advantages of our scheme with other existing algorithms and claim that our scheme offers a highly secure approach that provides users with data confidentiality and adequate performance.**

***Keywords*— Cloud Computing, encryption, data privacy, decryption algorithm.**

I. INTRODUCTION

In cloud context, the confidential and critical information is stored across the cloud. A recent report by the Cloud Security Alliance lists data loss and leakage as one of top security concerns in the cloud. Recent laws, regulations and compliance frameworks compound the risks; offending companies can be held responsible for the loss of sensitive data and may face heavy fines over data breaches. To lose data security practices also harm on a personal level. Lost or stolen medical records, credit card numbers or bank information may cause emotional and financial ruin. Sensitive data stored within cloud environments must be safeguarded to protect its owners. Ensuring the confidentiality of information requires the best data management choices like only the authorized party can access the original plain text. We propose the solution for these choices that removes the third-party involvement. The previously designed architecture for this is Proxy Less architecture (PLAC), in which the proxy server between the client and database is removed. The PLAC architecture has two problems. First is single point failure (means that data will be lost if the system failure occurs due to some problems) and second is bottleneck (too many request coming for same operation). Due to these drawbacks the performance of cloud is decreased.

To overcome these issues the DD-PLAC (Distributed Database-Proxy Less Architecture) is proposed. This architecture does not introduce any intermediary proxy server between the client and cloud database. In this distributed cloud, data is distributed across the cloud. The benefit of storing data on distributed database is it decreases the load coming on single database. The access permission is given to each authorized user according to their role. The DD-PLAC architecture is implementing with the help of vertical fragmentation of data. Fragmentation overcomes the problem of bottleneck. In this architecture AES algorithm is used with help of hash function so data is stored in encrypted format using the symmetric key. This DD-PLAC architecture supports the concurrent execution of the operations on the encrypted data. It gives guarantee about data security, confidentiality, Consistency, Integrity.

II.LITERATURE SURVEY

A. Existing System

The following are three types of architectures are defined to preserve the privacy.

1. Proxy-based architectures.
2. Proxy-less architectures that store metadata in the clients.
3. Proxy-less architectures that store metadata in the cloud database.

1. Proxy-based architectures (PBA):-

The proxy-based architectures do not satisfy our design requirements because the proxy is a bottleneck and a single-point-of-failure that limits availability, scalability and elasticity of the cloud DBaaS. Since the proxy must be trusted, it cannot be out sourced to the cloud and has to be deployed and maintained locally. Moreover, proxy-based architectures cannot scale trivially by increasing the number of proxies. Such a naive solution would imply the replication of metadata among all the proxies, but this would require synchronization algorithms and protocols to guarantee consistency among all the proxies.

2. Proxy-less architectures that store metadata in the clients (PLA):-

The Proxy-less architecture that store metadata in the clients does not use an intermediate proxy and metadata are stored at the client side. So the clients can connect directly to the cloud database, this architecture provides availability, scalability and elasticity. So, each client has its own encryption engine and manages a local copy of metadata. So, this solution can represent a sub-case of the proxy-based architecture, in which a different proxy is deployed within each client. A similar architecture for cloud accesses would suffer from the same consistency issues of proxy-based architectures.

3. Proxy-less architectures that store metadata in the cloud database (PLAC):-

The third architecture is proxy-less architectures that store metadata in the cloud database. In this the metadata is stored to the cloud database, but the encryption engine is executed by each client. As metadata are not shared among all the clients there is no need of synchronization. Client machines execute a client software component that allows a user to connect and issue queries directly to the cloud DBaaS. This software component retrieves the necessary metadata from the un trusted database through SQL statements and makes them available to the encryption engine at the client. Multiple clients can access the un trusted cloud database independently, with high availability, scalability and elasticity. The drawback of this architecture is bottleneck and the single point of failure.

B. Proposed System

The proposed architecture is based on PLAC architecture. The PLAC architecture has two drawbacks. First is the single point of failure and the second is the bottleneck. PLAC solves client concurrency management problems for write/read accesses to encrypted data in the cloud, but it does not guarantee data isolation and confidentiality against the collusion risk. All occupant users are provided with the same master key, and access control policies are implemented by leveraging the standard database access control mechanisms at the cloud provider side. To overcome these issues the DD-PLAC architecture is proposed. The Proxy less architecture with Encrypted Metadata in the cloud and Distributed database (DD-PLAC) mechanism is developed remove the problems of privacy, availability and bottleneck.

Architecture Design

The Proxy less architecture with Encrypted Metadata in the cloud and Distributed database (DD-PLAC) mechanism is developed to address the problems of privacy, availability and bottleneck. The distributed cloud database is used where the data is distributed over the cloud, which will allow the databases to totally support the flexible requirements of cloud computing applications. Databases have been distributed in terms of instances running on servers that have access to a high-speed network for a while. Access permissions are given for each user based on their role. Security algorithms are used to improve the privacy of the data stored in cloud.

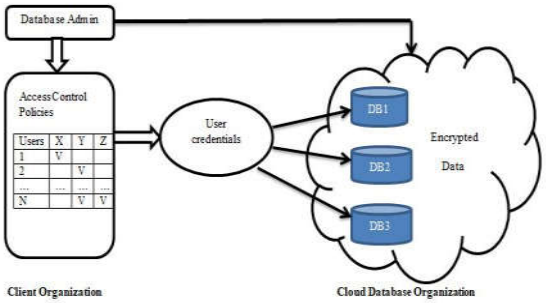

Figure 1: DD-PLAC Architecture

The DD-PLAC architecture shown in Fig.1.A client organization in which a trusted Database Admin machine hosts the DD-PLAC client, which is the responsible for the creation and management of the encrypted database. All database users can issue SQL operations directly to the cloud database even from geographically distributed locations by executing a DD-PLAC client on their machines. The entire set of data are stored in an encrypted form in the cloud database. Due to the SQL-aware encryption strategies, the cloud database engine can execute queries on encrypted data without accessing any decryption keys. Even metadata that are necessary to manage encryption strategies are considered critical information.

Proposed system includes following parameters for data security and data availability:-

1. AES Algorithm

Advanced Encryption standard it is based on substitution-permutation network. It uses 128,192 or 256-key size. AES is a Symmetric-key algorithm, meaning the same key is used for both encrypting and decrypting the data.

2. Hash Function

Hash function is used with AES algorithm to generate a symmetric key for encryption and decryption. For this hash function can randomly select any character from input string. The resultant key will provide very strong security to data.

3. Fragmentation Techniques

Fragmentation ensures for data availability. In the proposed architecture we use the vertical fragmentation technique in which data is fragmented using database columns. Fragmentation improves system performance it helps to fast access of data.

4. Metadata Management

Metadata management strategies represent an original idea because DD-PLAC architecture stores all metadata in the cloud database together with the encrypted tenant data.

DD-PLAC 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 is associated with one secure table. Each metadata table contains secure information that is necessary to encrypt and decrypt data of associated secure table.

III.COMPARATIVE RESULT

There are many ways to store the data on cloud such as using Single Database, Distributed Database. Using Single database client can store only data on single server, and if multiple client can access the data at the same time may be there is chances to data consistency, may be the data can modified during the concurrent access. It is easy to handle the data on single database but there was some problem regarding to data confidentiality, data consistency, data availability, bottleneck and single point of failure.

To overcome these problem, today we used Distributed Database, data can store on multiple server, so client can access the data concurrently. In this mechanism data can distribute using fragmentation techniques such as Vertical Fragmentation and Horizontal Fragmentation. The main advantage of distributed database is the data can available anytime and anywhere. Also the multiple replicas of same data are stored on multiple servers so if the data can be lost during the concurrent access, the replica of same data is available, so data can easily available to client. Security of data is maintained using various encryption algorithm such as AES (Advanced Encryption Standard), DES (Data Encryption Standard) etc. Distributed data can access using proxy server or without proxy server. In proxy based server there is another server in between client and server. The main issue of this architecture was data security, decrement of system performance, single point of failure etc. So the earlier architecture of distributed database refers proxy less architecture. In this architecture there is no any server in between client and server. The main advantages of this architecture are Data Security, Data availability, Data consistency etc.

IV.CONCLUSION

The DD-PLAC architecture provides a strong level of security and privacy. All the data which is stored on the cloud provider are encrypted through cryptographic algorithms which allow the execution of standard SQL queries on encrypted data. This architecture is also provides direct, independent and concurrent access to the cloud database. It does not rely on a trusted proxy that represents and also avoids the single point of failure and a system bottleneck, which in turn increases the availability and scalability of cloud database services.

REFERENCES

[1] Luca Ferretti, Michele Colajaani, and Micro Marchetti, " Distributed, Concurrent, and Independent Access to Distributed Database," 2015.

[2] Amjad Alsirhani, Peter Bodorik, Srinivas Sampalli, "Improving Database Security in Cloud Computing by Fragmentation of Data," 2017.

[3] Sajjan R.S., Vijay Ghorpade, Vishvajit Dalimbkar, "A Survey Paper on Data security in Cloud Computing," 2016.

[4] Parneet Kaur and Sachin Majithia, "Various Aspects for Data Migration in Cloud Computing and Related Reviews," 2014.

[5] L. M. Kaufman, "Data Security in the World of Cloud Computing," 2009.

[6] A. Hudic, S. Islam, P. Kieseberg, S. Rennert, and E. R. Weippl, "Data Confidentiality using Fragmentation in Cloud Computing," 2013.

[7] J. Daemen, "The design of Rijndael : AES - the Advanced Encryption Standard with 17 Tables," 2002.

[8] V. Ganapathy, D. Thomas, T. Feder, H. Garcia-Molina, and R. Motawani, "Distributing Data for Secure Database Services," 2011.

[9] Danie J. Abadi, "Data Management in the Cloud:Limitations and Opportunities," 2009.

[10] "Addressing Data Security Challenges in the Cloud A Trend Micro White Paper," 2010.

[11] James Broberg, Rajkumar Buyya, Zahir Tari, "MetaCDN:Harnessing Storage Clouds or high performance content delivery," 2009.