International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 7 (2018) pp. 4791-4795

Secure Data Access in Cloud with Multi-User Encrypted SQL Operations

P. Satyanarayana
Department of Mathematics & Computer Science, Osmania, University, Hyderabad, India.

M.V. Ramanamurthy
Professor & Head, Department of M&H, MGIT, Gandipet, Hyderabad-78, India.

S. China Ramu
Associate Professor, Department of Computer Science and Engineering, CBIT, Gandipet, Hyderabad-78, India.

Mustafa Abdu Lrazzaq Radhi Almusawi
Department of Mathematics & Computer Science, Osmania University, Hyderabad, India.

Abstract

It is a rare requirement to provide security at SQL level of cloud environment. When we use cloud for our business transactions or software development an deployment operations, we need to take care in all aspects of operations performing in cloud. As the cloud using internet as its backbone, it is essential to provide security to the data request and data providing. This paper provide an architecture to encrypt and decrypt the SQL request and response from the centralized cloud database.

Keywords: RSA algorithm, Cloud computing, **Eucalytus, Infrastructure as Service (IaaS), Cipher Text, Decrypting,** Diffie Hellman Algorithm

INTRODUCTION

Distributed computing is a progressive figuring strategy, by which processing assets are given powerfully through Internet and the information stockpiling and calculation are outsourced to somebody or some gathering in a 'cloud'. It extraordinarily pulls in consideration and enthusiasm from both the scholarly world and industry because of the benefit, yet it additionally has no less than three difficulties that should be taken care before going to our genuine to the best of our insight. Above all else, information privacy ought to be ensured.

There are three main related issues behind the data related problems: execution of SQL operators over encrypted data; enforcement of access control mechanisms through selective encryption strategies; design of architectures not penalizing the performance and scalability that are typical of cloud-based services[1][4] . Existing proposals offer partial and separate solutions to data confidentiality and isolation.

For example, architectures supporting SQL operations on encrypted data leave access control to the cloud provider or enforce it through an intermediate trusted server. Other proposed architectures solve the problem of access control without the intervention of the cloud provider, but they do not allow execution of SQL operations on encrypted data. We propose the first architecture, called Multi-User relational Encrypted DataBase (MuteDB) [2][6], that guarantees data confidentiality by executing SQL operations on encrypted data and by enforcing access control policies through selective encryption methods. By combining these two approaches MuteDB is the only solution ensuring confidentiality of data stored in the cloud even in the worst threat scenario where legitimate database users collude with cloud provider employees. This result is achieved through an innovative model that translates access control policies related to a plaintext database into selective encryption strategies that are applied to the corresponding encrypted database. Our solution works even in dynamic scenarios, in which users and access control policies change over time, without the need to renew and redistribute user credentials. The proposed architecture is specifically designed for cloud database scenarios where multiple users can access the cloud database through the Internet possibly from different geographical areas.

The performance and scalability of MuteDB are evaluated through a prototype that is subject to different query workloads based on standard (TPC-C) and recently proposed (YCSB) database benchmarks. We highlight that, as a further contribution. T his paper reports the first performance evaluation studies related to encrypted cloud database services in real distributed environments where the clients are geographically distributed over the Planet Lab platform. Experimental results shows that MuteDB does not affect the scalability of the original cloud service, and its performance for geographically distributed clients are comparable to those of encrypted cloud database services.

In this paper, we exhibit **Eucalyptus** [5][10]- an open-source programming structure for distributed computing that actualizes what is usually alluded to as framework as an administration of Infrastructure As Service (IaaS) frameworks that give clients the capacity to run and control whole virtual machine occurrences conveyed over an assortment physical assets.

We diagram the essential standards of the Eucalyptus outline, point of interest imperative operational parts of the framework, and talk about engineering exchange offs that we have made with a specific end goal to permit **EUCALYPTUS**

to be versatile, particular and easy to use on foundation ordinarily found inside scholarly settings. At long last, we give confirm that **EUCALYPTUS** empowers clients acquainted with existing matrix and HPC frameworks to investigate new distributed computing usefulness while keeping up access to existing, well known application advancement programming and network middleware[11].

PROBLEM STATEMENT

The research problem statement is as follows:

1. To propose solutions for enforcing access control and for guaranteeing confidentiality of data and metadata.
2. To analyze and present the program in Java.
3. Any cloud provider assures the security and availability of its platform, while the implementation of scalable solutions to guarantee confidentiality of the information stored in cloud databases is an open problem left to the tenant.
4. To study the three main related issues behind these two problems: execution of SQL operators over encrypted data; enforcement of access control mechanisms through selective encryption strategies; design of architectures not penalizing the performance and scalability that are typical of cloud-based services .

PROBLEM ANALYSIS

NEED OF THE STUDY

1. To propose the first complete architecture that combines data encryption, key management, authentication and authorization solutions, and it addresses the issues related to typical threat scenarios for cloud database services.
2. To study the address for some preliminary issues through SQL operations on encrypted data.
3. To analyze the architecture and present the output results.

LIMITATIONS:

The research has some limitation as follows:

- The research limits in analyzing the cloud computing in networking.
- The study mainly focuses on implementation of public auditing using Multi-user encryption services in cloud technology.
- The data collection for the research and practical programming may not be accurate.

Property based encryption module is utilizing for every last hub encode information stored. After scrambled information and again the re-encoded the same information is utilizing for fine-grain idea utilizing client information transferred the characteristic based encryption have been proposed to secure the distributed storage. Quality Based Encryption (QBE). In such encryption conspire, a character is seen as an arrangement of clear properties, and unscrambling is conceivable if a decrypter's personality has a few covers with the one determined in the cipher text.

SYSTEM DESIGN

1- Symmetric Key Cryptography

Symmetric-key
algorithms are algorithms for cryptography that use the same cryptographic keys for both encryption of plaintext and decryption of cipher text. The keys may be identical or there may be a simple transformation to go between the two keys. The keys, in practice, represent a shared secret between two or more parties that can be used to maintain a private information link. This requirement that both parties have access to the secret key is one of the main drawbacks of symmetric key encryption, in comparison to public-key encryption (also known as asymmetric key encryption).

Security of symmetric ciphers

Symmetric ciphers have historically been susceptible to known-plaintext attacks, chosen-plaintext attacks, differential cryptanalysis and linear cryptanalysis. Careful construction of the functions for each round can greatly reduce the chances of a successful attack.

Security of RSA

As discussed, the security of RSA relies on the computational difficulty of factoring large integers. As computing power increases and more efficient factoring algorithms are discovered, the ability to factor larger and larger numbers also increases. Encryption strength is directly tied to key size, and doubling key length delivers an exponential increase in strength, although it does impair performance. RSA keys are typically 1024- or 2048-bits long, but experts believe that 1024-bit keys could be broken in the near future, which is why government and industry are moving to a minimum key length of 2048-bits. Barring an unforeseen breakthrough in quantum computing, it should be many years before longer keys are required, but elliptic curve cryptography is gaining favor with many security experts as an alternative to RSA for implementing public-key cryptography. It can create faster, smaller and more efficient cryptographic keys. Much of today's hardware and software is ECC-ready and its popularity is likely to grow as it can deliver equivalent security with lower computing power and battery resource usage, making it more suitable for mobile apps than RSA. Finally, a team of researchers which included Adi Shamir, a co-inventor of RSA, has successfully determined a 4096-bit RSA key using acoustic cryptanalysis, however any encryption algorithm is vulnerable to this type of attack.

Diffie Hellman Algorithm

Diffie-Hellman key exchange, also called exponential key exchange, is a method of digital encryption that uses numbers

raised to specific powers to produce decryption keys on the basis of components that are never directly transmitted, making the task of a would-be code breaker mathematically overwhelming.

To implement Diffie-Hellman, the two end users Alice and Bob, while communicating over a channel they know to be private, mutually agree on positive whole numbers p and q, such that p is a prime number and q is a generator of p. The generator q is a number that, when raised to positive whole-number powers less than p, never produces the same result for any two such whole numbers. The value of p may be large but the value of q is usually small.

Once Alice and Bob have agreed on p and q in private, they choose positive whole-number personal keys a and b, both less than the prime-number modulus p. Neither user divulges their personal key to anyone; ideally they memorize these numbers and do not write them down or store them anywhere. Next, Alice and Bob compute public keys a^* and b^* based on their personal keys according to the formulas

$$a^* = q^a \bmod p \quad \text{AND} \quad b^* = q^b \bmod p$$

The two users can share their public keys a^* and b^* over a communications medium assumed to be insecure, such as the Internet or a corporate wide area network (WAN). From these public keys, a number x can be generated by either user on the basis of their own personal keys. Alice computes x using the formula

$$x = (b^*)^a \bmod p$$

Bob computes x using the formula

$$x = (a^*)^b \bmod p$$

The value of x turns out to be the same according to either of the above two formulas. However, the personal keys a and b, which are critical in the calculation of x, have not been transmitted over a public medium. Because it is a large and apparently random number, a potential hacker has almost no chance of correctly guessing x, even with the help of a powerful computer to conduct millions of trials. The two users can therefore, in theory, communicate privately over a public medium with an encryption method of their choice using the decryption key x.

The most serious limitation of Diffie-Hellman in its basic or "pure" form is the lack of authentication. Communications using Diffie-Hellman all by itself are vulnerable to man in the middle attacks. Ideally, Diffie-Hellman should be used in conjunction with a recognized authentication method such as digital signatures to verify the identities of the users over the public communications medium. Diffie-Hellman is well suited for use in data communication but is less often used for data stored or archived over long periods of time.

IMPLEMENTATION

Execution is the phase of the venture when the hypothetical outline is transformed out into a working framework. Hence it can be thought to be the most basic stage in accomplishing a fruitful new framework and in giving the client, certainty that the new framework will work and be compelling. The usage stage includes watchful arranging, examination of the current framework and it's requirements on execution, planning of systems to accomplish changeover and assessment of changeover techniques.

Encryption and Decryption implementation can be done using any platform. The sample APIs are developed using Java platform. The interfaces developed are depicted in Fig. 5.1, 5.2, 5.3, 5.4, 5.5, 5.6 right from user registration, login, query encryption with algorithm options and decryption of the cloud result of database tables. The algorithm can be implemented for encryption and decryption algorithms as follows.

Encryption algorithm

The following steps should be followed to develop an encrypted text:

- 1.) Generate the ASCII value of the letter
- 2.)Generate the corresponding binary value of it (Binary value should be 8 digits e.g. for decimal 32 binary number should be 00100000)
- 3.) Reverse the 8 digit's binary number
- 4.) Take a 4 digits divisor (>=1000) as the Key
- 5.)Divide the reversed number with the divisor
- 6.)Store the remainder in first 3 digits & quotient in next 5 digits (remainder and quotient wouldn't be more than 3 digits and 5 digits long respectively. If any of these are less than 3 and 5 digits respectively we need to add required number of 0s (zeros) in the left hand side. So, this would be the cipertext i.e. encrypted text.

Now store the remainder in first 3 digits & quotient in next 5 digits.

Let us see an example to apply the above mentioned steps:

Encryption

Let, the character is "T". Now according to the steps we will get the following:

- Step 1: ASCII of "T" is 84 in decimal.
- Step 2: The Binary value of 84 is 1010100. Since it is not an 8 bit binary number we need to make it 8 bit number as per the encryption algorithm. So it would be 01010100
- Step 3: Reverse of this binary number would be 00101010
- Step 4: Let 1000 as divisor i.e. Key
- Step 5: Divide 00101010 (dividend) by 1000(divisor)
- Step 6: The remainder would be 10 and the quotient would be 101. So as per the algorithm the ciphertext would be 01000101 which is ASCII 69 in decimal i.e. "E"

Decryption algorithm

- Step 1: Multiply last 5 digits of the cipher text by the Key
- Step 2: Add first 3 digits of the cipher text with the result produced in the previous step
- Step 3: If the result produced in the previous step i.e. step 2 is not an 8-bit number we need to make it an 8- bit number
- Step 4: Reverse the number to get the original text i.e. the plain text

Decryption

- Step 1: Multiply last 5 digits of the cipher text by the Key
- Step 2: Add first 3 digits of the cipher text with the result produced in the previous step
- Step 3: If the result produced in the previous step i.e. step 2 is not an 8-bit number we need to make it an 8- bit number
- Step 4: Reverse the number to get the original text i.e. the plain text

SAMPLE OUTPUT SCREEN SHOTS

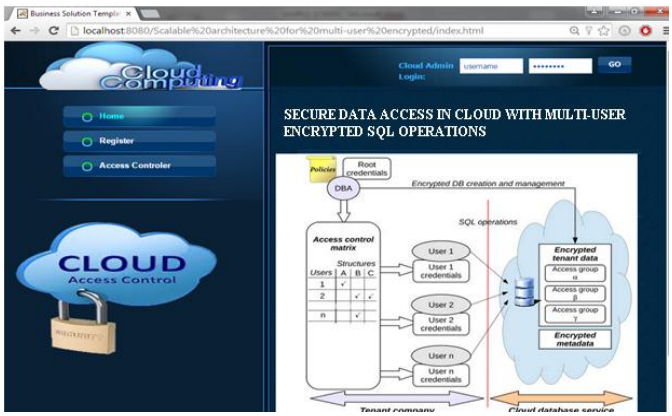

Figure 5.1 Home Page

Figure 5.2 User Login

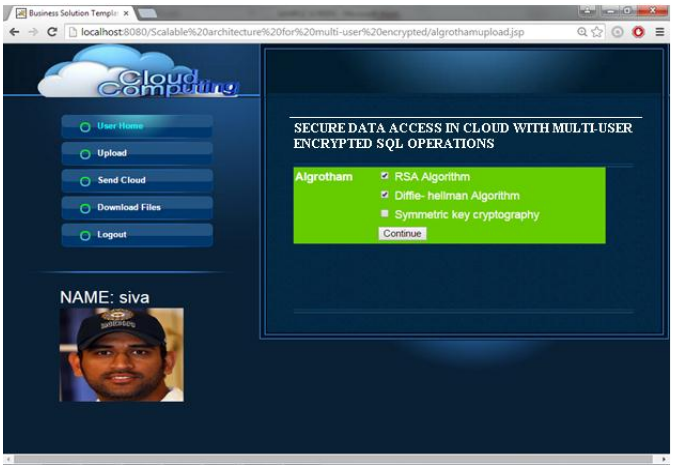

Figure 5.3 Encryption message

Figure 5.4 Encrypted message

Figure 5.5 Security Key

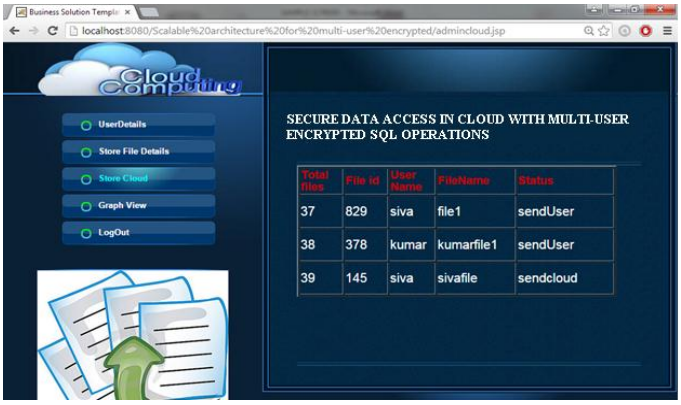

Figure 5.6 Stored Cloud

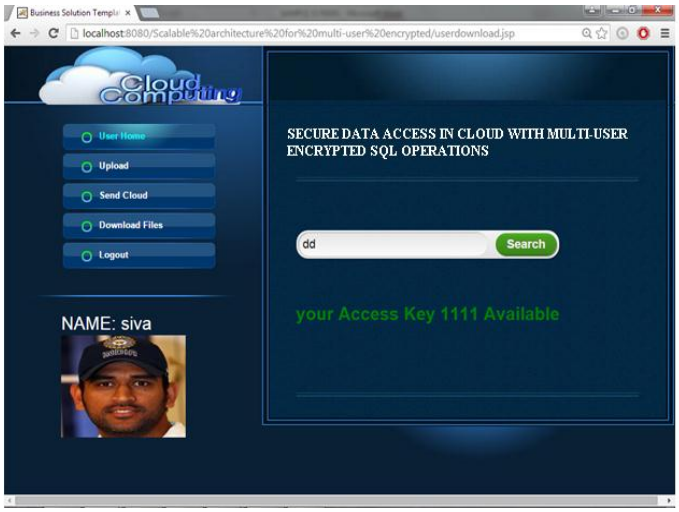

Figure 5.7 Message decrypting

CONCLUSION

In this paper we propose Mute DB, a novel architecture for cloud database services that guarantees for the first time data confidentiality through SQL-aware encryption algorithms and data isolation through access control enforcement based on encryption and key derivation techniques. These solutions allow Mute DB to address threat issues that are relevant for cloud services including risks of information leakage due to collusions between cloud provider employees and tenant users. The most important solutions are described through formal models, while the feasibility, performance and scalability of the proposed architecture are demonstrated through a large set of experiments carried out through a prototype deployed in a real Internet-based environment where cloud database services are accessed concurrently by geographically distributed clients.

All results confirm that for realistic workloads, the Mute DB architecture achieves performance and scalability comparable to those of unencrypted cloud database services. Ongoing work is focused on integrating private information retrieval solutions in Mute DB with the goal of preventing information leakage caused by access pattern analyses, and no vel. architectural solutions for hybrid cloud environments.

REFERENCES

[1] Bitar, N., Gringeri, S., & Xia, T. J. (2013). Technologies and protocols for data center and cloud networking. Communications Magazine, IEEE, 51(9), 24-31.

[2] Ramgovind, S., Eloff, M. M., & Smith, E. (2010, August). The management of security in cloud computing. In Information Security for South Africa (ISSA), 2010 (pp. 1-7). IEEE.

[3] Oberheide, J., Veeraraghavan, K., Cooke, E., Flinn, J., & Jahanian, F. (2008, June). Virtualized in-cloud security services for mobile devices. In Proceedings of the First Workshop on Virtualization in Mobile Computing (pp. 31-35). ACM.

[4] Schoo, P., Fusenig, V., Souza, V., Melo, M., Murray, P., Debar, H., ... & Zeghlache, D. (2011). Challenges for cloud networking security. In Mobile Networks and Management (pp. 298-313). Springer Berlin Heidelberg.

[5] Nurmi, D., Wolski, R., Grzegorczyk, C., Obertelli, G., Soman, S., Youseff, L., & Zagorodnov, D. (2009, May). The eucalyptus open-source cloud-computing system. In Cluster Computing and the Grid, 2009. CCGRID'09. 9th IEEE/ACM International Symposium on (pp. 124-131). IEEE.

[6] Wodczak, M. (2011, November). Resilience aspects of autonomic cooperative communications in context of cloud networking. In Network Cloud Computing and Applications (NCCA), 2011 First International Symposium on (pp. 107-113). IEEE.

[7] Zissis, D., & Lekkas, D. (2012). Addressing cloud computing security issues. Future Generation computer systems, 28(3), 583-592.

[8] Ramgovind, S., Eloff, M. M., & Smith, E. (2010, August). The management of security in cloud computing. In Information Security for South Africa (ISSA), 2010 (pp. 1-7). IEEE.

[9] Okuhara, M., Shiozaki, T., & Suzuki, T. (2010). Security architecture for cloud computing. Fujitsu Sci. Tech. J, 46(4), 397-402.

[10] Houidi, I., Mechtri, M., Louati, W., & Zeghlache, D. (2011, July). Cloud service delivery across multiple cloud platforms. In Services Computing (SCC), 2011 IEEE International Conference on (pp. 741-742). IEEE.

[11] Wang, Q., Wang, C., Li, J., Ren, K., & Lou, W. (2009). Enabling public verifiability and data dynamics for storage security in cloud computing. In Computer Security–ESORICS 2009 (pp. 355-370). Springer Berlin Heidelberg.