Efficient and Secure Access to Cloud Database

[1]Sanket Rameshwarrao Milke
[1]PG Scholar
[1]Computer Science & Engineering Department
[1]Government Engineering College, Aurangabad, India

***Abstract* - Availability and scalability can be achieved by usual existing Cloud database services, but there are issues to provide guarantee about data confidentiality. Adding encryption at the time of deployment of data on the cloud is a good way. Existing applications based on some trusted intermediate server adds limitations to availability and scalability of original cloud database services.The architecture intended in this paper that avoids any intermediary component, thus achieving availability and scalability comparable to that of unencrypted cloud database services. Also this architecture promises data consistency in situations in which independent clients concurrently execute SQL queries, and the structure of the database can be modified. Several alternatives exist for storage services, while data confidentiality solutions for the database as a service model are still have scope to work. The intended architecture integrates cloud database services with data confidentiality and the possibility of executing concurrent operations on encrypted data. This is the solution which supports geographically distributed clients to connect directly to an encrypted cloud database, and to execute concurrent and independent operations. The proposed architecture has the additional advantage of eliminating intermediate proxies that limit the elasticity, availability, and scalability properties that are basic in cloud-based solutions.**

***IndexTerms* - Cloud, security, confidentiality, SecureDBaaS, database**

I. INTRODUCTION

Cloud Computing refers to both the applications delivered as services over the Internet and the hardware and systems software in the datacenters that provide those services. The services themselves have long been referred to as Software as a Service (SaaS). We can call cloud the datacenter hardware and software together. When a cloud is made available in a pay-as-you-go manner to the general public, we call it Public Cloud and the service being sold is Utility Computing. Private Cloud refers to the internal datacenters of a business or other organization, not made available to the general public. So we can call Cloud Computing to the Software as a Service (SaaS) and Utility Computing together but excluding Private Clouds.[1]

In the cloud context critical information is placed in environment of untrusted third parties assuming data confidentiality is very important. [2] In this environment access to original plain data must limited to trusted parties excluding cloud provider, intermediate proxies if any and internet also. The SecureDBaas architecture can be used specially for cloud platforms and does not include any intermediate proxy server between the client and cloud provider. Eliminating any trusted intermediate server allows SecureDBaaS to achieve the same availability,reliability, and elasticity levels of a cloud DBaaS. There are some other architectures with intermediate servers are not considered as efficient in use for cloud based solutions because due to any proxy server single point failure and overall system bottleneck problems may increase which may result into limitation of main benefits i.e. availability, scalability and elasticity of database service deployed on a cloud platform.

The set of database operations are performed on real cloud platform such as Jelastic Cloud[3]. This shows that SecureDBaas architecture is ready to immediate implementation in any database services.

II. RELATED WORK

This paper intends an architecture that is different from previous work in the field of secure cloud database services. Cryptographic file systems and secure storage solutions represent the earliest works to provide confidentiality and integrity of data outsourced to untrusted cloud storage services. There are many existing solutions for the storage as a service such as SPORC[4], SUNDR[5], and DEPOT[6] but not for database as a service.We do not give detail explanation of the several papers and products in this field [7] because they do not allow any computation on encrypted data. Hence they cannot be applied to the context of cloud DBaaS. Some DBMS engines offer the possibility of encrypting data at the file system level through the so called Transparent Data Encryption (TDE) feature [8]. This feature makes it possible to build a trusted DBMS over untrusted storage. However, in the DBaaS context the DBMS engine is not trusted because it is controlled by the cloud provider, hence the TDE approach is not applicable to cloud database services.

An approach to preserve data confidentiality in scenarios where the DBMS is not trusted is proposed in [9]. However it requires a modified DBMS engine that is not compatible with commercial and open source DBMS software adopted by cloud providers. On the other hand, the architecture we propose is compatible with standard DBMS engines, and allows customers to build a secure cloud database by leveraging cloud DBaaS readily available. Supporting Security and Consistency for Cloud Database . The proposal in [10] uses encryption to control accesses to encrypted data stored in a cloud database. This solution is not applicable to usage contexts in which the structure of the database changes, and does not support concurrent accesses from multiple clients possibly distributed on a geographical scale.

Data outsourcing or database as a service is a new model for data management in which a third party service provider hosts a database as a service. By using Database as a service model customers can manage their data without purchasing costly hardware

and software. Also it reduces the need of hiring technical professionals for administrative and maintenance tasks. So using an external database service like DBaas guarantee reliable data storage at a low cost so that it attracts companies and individual customers. Also it provides universal access through internet URL.
Most of the existing systems providing storage as a service in the cloud context or intermediate proxy servers used for security of the database used AES encryption algorithm for securing data stored over cloud platforms.

III. PROPOSED SYSTEM

The previous architectures are based on an intermediate proxy server that plays a mediator role for security purpose between clients and the untrusted DBMS server. The reliance on a trusted proxy has several drawbacks like less security, possible access of security providers administrator and more important that it takes more time and violates the main benefits of the cloud database service i.e. Availability, Confidentiality and Scalability. The architecture proposed in this system has several benefits from existing solutions because it allows multiple and independent clients to connect directly to the untrusted cloud DBaaS without any intermediate server. The main goal of the proposed system is to reduce the time required for accessing the database stored at the cloud.

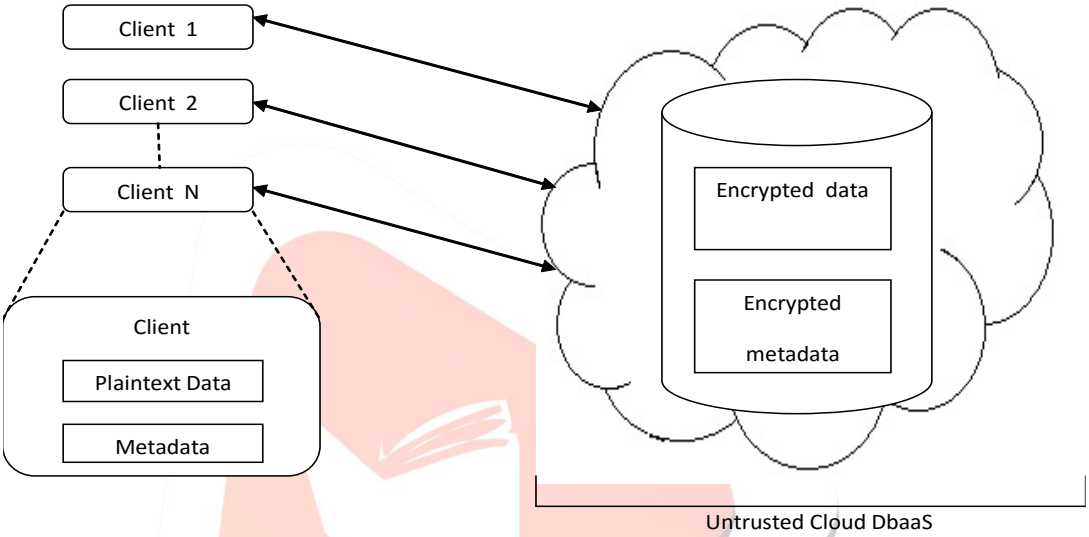

Figure:. System Architecture of the Proposed System

Only for encryption purpose custom Blowfish encryption algorithm is used instead of AES algorithm in this system architecture. Custom Blowfish algorithm differs from standard existing Blowfish algorithm.

Blowfish is a variable-length key, 64-bit block cipher encryption algorithm. The algorithm consists of two parts:

1) A key-expansion part

Key expansion converts a key of at most 448 bits into several subkey arrays totaling 4168 bytes.

Blowfish uses a large number of subkeys. These keys must be precomputed before any data encryption or decryption.

The P-array consists of 18 subkeys of 32-bit each:

P1, P2,..., P18.

There are four 32-bit S-boxes with 256 entries each:

S1,0, S1,1,..., S1,255;
S2,0, S2,1,..., S2,255;
S3,0, S3,1,..., S3,255;
S4,0, S4,1,..., S4,255.

The P array and S box entries are taken in long Integer in existing Blowfish encryption algorithm which are converted into hexadecimal code and finally into bytes for encryption purpose. In order to reduce time duration required for encryption process proposed system takes the P array entries and S box values directly in hexadecimal code which eliminates the conversion time from long Integer to hexadecimal code.

2) A data- encryption part.

Data encryption occurs through a 16-round Feistel process. Each round consists of a key dependent permutation, and a key- and data-dependent substitution. All operations are XORs and additions on 32-bit words. The only additional operations are four indexed array data lookups per round.The input is a 64-bit data element, x. Divide x into two 32-bit halves: xL representing Left Part, and xR representing Right Part.

Then, for i = 1 to 16:
xL = xL XOR Pi
xR = F(xL) XOR xR
Swap xL and xR
After the sixteenth round, swap xL and xR again to undo the last swap.
Then, xR = xR XOR P17 and xL = xL XOR P18.

Finally, recombine xL and xR to get the encrypted text as a result.

To minimize the time duration required for encryption process the number of rounds are reduced in the proposed system.

Proposed architecture also provide security to the metadata in all conditions whether it is in rest, in motion or in use by the stored encrypted metadata at the cloud. Only clients knowing encryption key can decrypt metadata. In the intended architecture the plaintext database is converted into encrypted database by translating each plaintext table into corresponding encrypted table.

IV. SYSTEM DEVELOPMENT

For implementation of the proposed system an architecture is built where the SQL operations are executed on the database deployed at the cloud . For now the Insert query is executed for uploading data on the cloud platform. Then the Update and Delete SQL operations are performed on the same database.

For the ease of implementation of Batch SQL operations and simulation of multiple operations at a time on the same database we have performed each SQL operation through XML file. At first user or customer can insert the records into XML file through GUI. The same procedure is performed for all the three operations. The all inserted records till now are get written in XML files. The actual change in the database is done after the successful execution of these XML files.

Metadata driven Configurability is one of the features of the SaaS applications in which customers can configure their application through metadata. So it may be a possible threat.

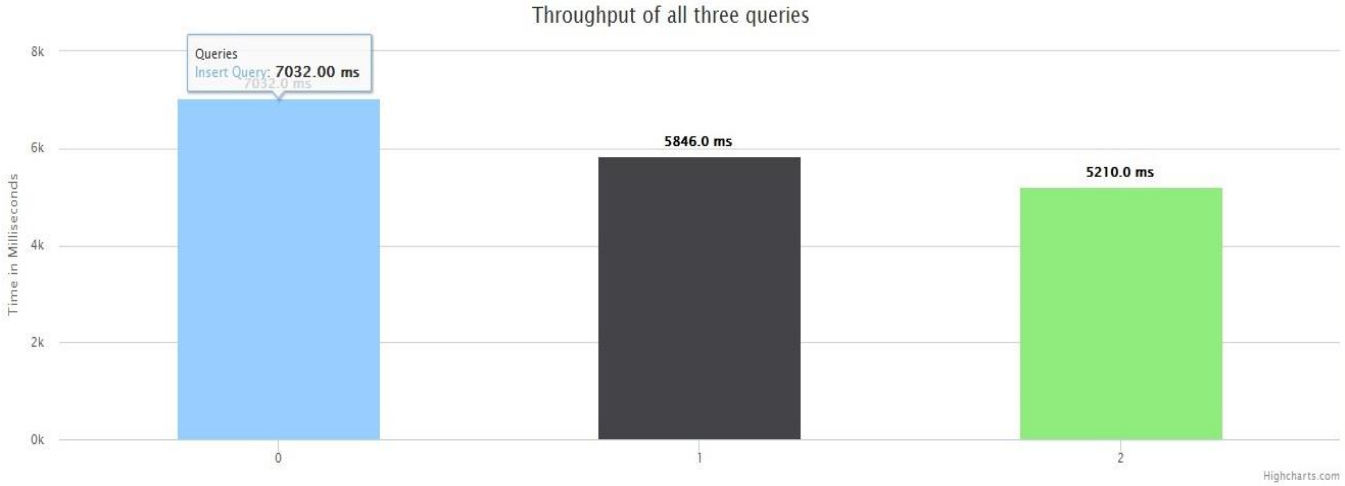

Figure: Graphical representation of all three queries executing on 100 records in existing AES encryption

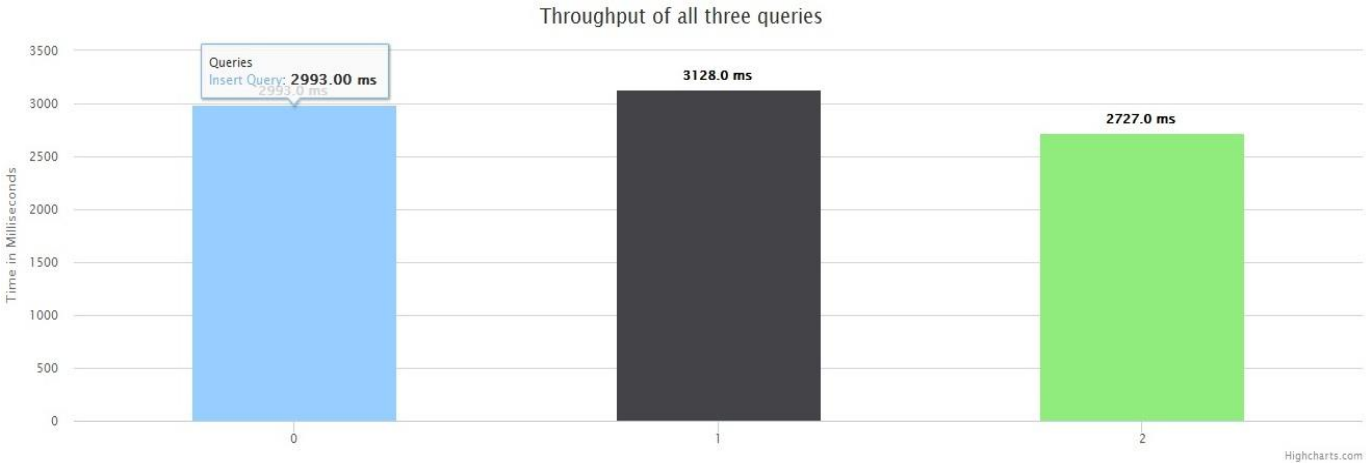

Figure: Graphical representation of all three queries executing on 100 records in Custom Blowfish encryption

to security of the database present at the cloud. To avoid this threat we secure metadata also by encrypting it with the same encryption algorithm mentioned above.

V. PERFORMANCE ANALYSIS

The applicability of SecureDBaaS to different cloud DBaaS solutions is demonstrated by implementing and handling encrypted database operations on emulated and real cloud infrastructures. The present version of the SecureDBaaS prototype carried SQL operations on Jelastic Cloud platform. At first hundred records are inserted in the database detailing doctors information by successful execution of Insert record XML file. Likewise the Insert operation took **7032ms**. For Update query execution hundred records are inserted in XML file updating a single field in the database, but for each record. The time taken for Update query is also noted down as **5846ms**. In the same pattern again for Delete operation also hundred records are inserted in XML file which after execution deletes hundred records in the database. The Delete operation took **5210ms** for deleting hundred records from the

database.The time taken to execute query is shown and at the same time graph is also generated. We are showing the graph generated at the very first reading of executions of three SQL operations. The readings of time taken to execute the operations are different at each time we execute the XML files as it is affected by the network latencies, Cloud performance and client machine also.

VI. CONCLUSION

The architecture implemented provides confidentiality of data stored in public cloud databases. The solution given by this architecture does not depend on an intermediate proxy which can be a single point of failure and a bottleneck limiting availability and scalability of typical cloud database services. This architecture does not require modifications to the cloud database, and it is immediately applicable to existing cloud DBaaS, such as the experimented Jelastic Cloud. There are no theoretical and practical limits to extend this solution to other platforms and to include new encryption algorithms.

VII. ACKNOWLEDGMENT

I wish to express gratitude to my guide Prof. S.G. Shikalpure for his valuable and firm suggestions, guidance, encouragement and constant support throughout this work without which it would not have been possible to complete the paper. He took deep interest in checking the minute details of the paper.

I am sincerely thankful to Prof. V.P. Kshirsagar ,Head of the Department, Department of Computer Science and Engineering, and Dr. P. S. Adwani, Principal, Government College of Engineering, Aurangabad for being a source of constant motivation.

I am very thankful to my parents, family and friends for always being there with me. With all due respect, I would like to thank all the people who have helped me directly or indirectly. Without their silent support and encouragement this paper would never have been possible.

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] https://jelastic.com/wp-content/uploads/2014/06/Jelastic_Hosting_fnl.pdf

[4] 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.

[5] 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.

[6] 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.

[7] R.A. Popa, C.M.S. Redfield, N. Zeldovich and H. Balkrishnan, "CryptDB: Protected Confidentiality With Encrypted Query Processing," Proc. ACM 23rd Symp Operating System Principles, Oct 2011.

[8] H. Hacigumus, 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.

[9] E. Damiani, S.D.C. Vimercati, S. Jajodiya, S. Paraboschi, P. Samarati, "Balancing Confidentiality and Efficiency in Untrusted Relational DBMS," Proc Tenth ACM Conf. Computer and Comm. Security, Oct. 2003.

[10] L. Ferretti, M. Colajanni, and M. Marchetti, "Supporting security and Consistency for Cloud Database, " Proc. Fourth Int'l Symp. Cyberspace Safety and Security, Dec.2012