© 2015 IJSRSET | Volume 1 | Issue 2 | Print ISSN : 2395-1990 | Online ISSN : 2394-4099
Themed Section: Engineering and Technology

Secure Storage using TPC-C in Cloud

A.Azarudeen, N.Ganesh, R.Dinesh, R.Ramakrishnan
Information Technology, Dhanalakshmi College of Engineering, Chennai, Tamilnadu, India

ABSTRACT

We propose a architecture that integrates cloud database services with data confidentiality and the possibility of executing concurrent operations on encrypted data. The proposed architecture is based on TPC-C standard benchmark for different number of clients and network latencies. Data and metadata are encrypted through the same encryption before being saved. Only trusted clients that already know the master key can decrypt the metadata and acquire information that is necessary. Each metadata can be retrieved by clients through an associated ID, which is the primary key of the metadata storage table. This ID is computed by applying a Message Authentication Code (MAC) function to the name of the object (database or table) described by the corresponding row.

Keywords: Cloud, Security, Confidentiality, Database.

I. INTRODUCTION

In a cloud context, where important data is placed in storage area of untrusted third parties, attaining data confidentiality is the main goal. This ensures clear data management: original plain data must be accessible only by authenticated users that do not include cloud providers, intermediaries, and Internet; in those untrusted areas, data must be encrypted and decrypted. 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. In this context, we propose TPC-C standard benchmark for different numbers of clients and network latencies for the performance of concurrent read and write operations in distributed environment. This is the first 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 architecture design was mainly based by on the following: to allow multiple, independent, and geographically distributed clients to execute concurrent operations on encrypted data and to preserve data confidentiality and consistency at the client and cloud level. Here there is no intermediary server used between user and cloud, thus assures more security.

About TPC-C

TPC Benchmark C is an Online Transaction Processing (OLTP) workload.TPC-C benchmark has been the industry benchmark to measure the performance of complex OLTP systems. It supports environments which are characterized by several conditions:

- Simultaneous Execution of Multiple transaction types that span a breath of complexity.
- Multiple on-line terminal sessions.
- ACID properties.
- Non-uniform distribution of data access through primary and secondary keys.
- Databases consisting of many tables with a wide variety of sizes, attributes and relationships.

II. METHODS AND MATERIAL

A.Related Work

This architecture provides several original features that differentiate TPC-C from previous work in the field of security for remote database services.

- It guarantees data confidentiality by allowing a cloud database server to execute concurrent operations over encrypted data.
- It supports availability, elasticity, and scalability of the original cloud DBaaS because it does not require any intermediate server.
- Multiple clients who were possibly distributed, can access concurrently and independently a cloud database service.

- It does not require a trusted broker or a trusted proxy because original data and metadata stored by the cloud database are always encrypted.

Cryptographic algorithms and various key generation algorithms already used can be exercised with little changes. Different approaches guarantee some confidentiality by distributing data among different providers and by taking advantage of secret sharing.

In such a way, they prevent one cloud provider to read its portion of data, but information can be reconstructed by colluding cloud providers. A step forward is proposed in, that makes it possible to execute range queries on data and to be robust against collusive providers. TPC-C differs from these solutions as it does not require the use of multiple cloud providers, and makes use of encryption algorithms to support the user.

TPC-C relates more closely to works using encryption to protect data managed by untrusted databases by providing automatic key generation. In such a case, a main issue to address is that users must make note of key details during uploading the file and secretly maintaining it

SecureDBaaS provides similar solution but it does not store data in encrypted format. SecureDBaaS is more related to and that preserves data confidentiality in untrusted DBMSs through encryption techniques, allows the execution of SQL operations over encrypted data, and is compatible with common DBMS engines. However, the architecture of these solutions is based on an intermediate and trusted proxy that mediates any interaction between each client and the untrusted DBMS server.

As TPC-C does not have intermediate proxy, the above issues can be resolved efficiently.

“Executing SQL over Encrypted Data in the Database-Service-Provider Model,”

The above approach proposed works by encrypting blocks of data instead of each data item. Whenever a data item that belongs to a block is required, the trusted proxy needs to retrieve the whole block, to decrypt it, and to filter out unnecessary data that belong to the same block. As a consequence, this design choice requires heavy modifica-tions of the original SQL operations produced by each client, thus causing significant overheads on both the DBMS server and the trusted proxy. On the other hand, SecureDBaaS allows the execution of operations over encrypted data through SQL-aware en-cryption algorithms. This technique, initially proposed in CryptDB, makes it possible to execute operations over encrypted data that are similar to operations over plaintext data.

B. Architecture Design

Our approach is designed to allow multiple and independent clients to connect directly to the untrusted cloud DBaaS without any intermediate server. Fig. 1 describes the overall architecture. We assume that client logins in his previously created account in our cloud service provider. Then if new client registers and then will able to login to his account to use his storage as a service feature through this confidentially in cloud. Tenant then deploys one or more machines (Client 1 through N). Thus administrator of cloud will be only able to provide storage area and user will have ultimate authority over the data. Also both the plain text data and its Meta data information will be in the encrypted format, it ensures more confidentiality to the registered users.

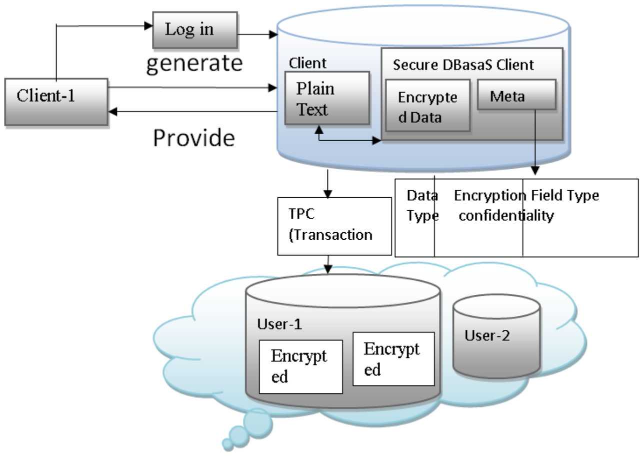

Figure 1: System Architecture

We assume the same security model that is commonly adopted by the literature in this field, where tenant users are trusted, and the network is untrusted, and the cloud provider is honest-but-curious, that is, cloud service operations are executed correctly, but tenant information confidentiality is at risk. Thus, in this case our architecture based on TPC-C all tenant data, data structures, and metadata must be encrypted before exiting from the client and there is no “intermediate proxy”.

Plaintext data consist of information that a tenant wants to store and process remotely in the cloud DBaaS. To prevent an untrusted cloud provider from violating confidentiality of tenant data stored in plain form, this technique adopts multiple cryptographic techniques to transform plaintext data into encrypted tenant data and encrypted tenant data structures because even the names of the tables and of their columns must be encrypted. Cloud users produce also a set of metadata consisting of information required to encrypt and decrypt data as well as other administration information. Even metadata are encrypted and stored in the cloud DB.

TPC-C architecture differs 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, these previous solutions are quite inefficient. For example, saving metadata on the clients would require onerous mechanisms for metadata synchronization, and the practical impossibility of allowing multiple clients to access cloud database services independently. Solutions based on a trusted proxy are more feasible, but they introduce a system bottleneck that reduces availability, elasticity, and scalability of cloud database services.

TPC-C proposes a different approach where all data and metadata are stored in the cloud database in an encrypted format. Encryption strategies for tenant data and innovative solutions for metadata management and storage are described in the following two sections

C. Data Management

Database tables consist of columns and rows. Each column contains a different type of attribute and each row corresponds to a single record. From that, we can retrieve all records that match certain criteria and also can Cross-reference records in different tables. Cloud resource management requires complex policies and decisions for multi-objective optimization. The cloud service provider is responsible for maintaining an agreed-on level of service and provisions resources accordingly.

D.Meta-Data Management

In the Proposed Architecture the Meta data must also be in the encrypted format, so that unauthorized users can't know what type of data they are going to access and hence it supports security.

- The metadata can give information about the characteristics of data.
- Here the metadata gets split into Data Type, size and Field Confidentiality. Since the clients who are accessing cloud database are distributed globally, we can adopt proportional share allocation concept.
- Proportional share allocation means authorized users only accessing our own data other Proxy don't access.
- To avoid the third proxy any registered users communicate to cloud server based upon requested names.
- Cloud response to the metadata (M.D) after corresponding questions properly respond means original data displayed otherwise not displayed.

III. RESULTS AND DISCUSSION

Operation

We can adopt different encryption algorithms like "Attribute Based Encryption Schema Algorithm" (ABS) in which clients files when uploaded in cloud DB are sorted according to their types. In our model automatic generation of key for files can be created uniquely based on the following sample.

```
function keyfunction()
 {
          var totalchar="123456789";
           // Random  random=new Random();
           var key="";
  for(var i=1;i<=4;i++)
          {
          var        keygen=Math.floor(Math.random()*
          totalchar.length);
          key=key+totalchar.charAt(keygen);
         }
   document.getElementById("key").value=key;
}
```

Encryption and key makes the files stored in the cloud DB is more secured by this TPC-C architecture

IV. CONCLUSION

Thus our proposed concept based on TPC-C benchmark can me effective and it improves efficiency by the concurrent read and write operations on the cloud database.

Also we propose an innovative architecture that guarantees confidentiality of data stored in public cloud databases. Unlike state-of-the-art approaches, our solution does not rely on an intermediate proxy that we consider 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. It is worth observing 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.

Our future work we propose an innovative architecture that guarantees confidentiality of data stored in public cloud databases. Unlike state-of-the-art approaches, for example if your stored 100 files means now you needed only 50 files remaining no need we deleted remaining 50 files. In extra future enhancement, we are going to show over all cloud monitoring with the pie chart. The pie chart shows the number of user in cloud database and corresponding cloud space memory registered by each and every user. This result is important and practical solution for guaranteeing data confidentiality in real cloud database services. Furthermore, the network latencies tend to mask cryptographic overheads for any number of clients.

V. REFERENCES

[1] Distributed, Concurrent, and Independent Access to Encrypted Cloud Databases; Luca Ferretti, Michele Colajanni, and Mirco Marchetti,,february-2014

[2] Energy Cost,The Key Challenge of Today's Data Centers:A Power Consumption Analysis on TPC-C Results,Meikel Poess Raghmath Othayoth Nambiar,2008

[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] 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.

[5] D. Agrawal, A.E. Abbadi, F. Emekci, and A. Metwally, "Database Management as a Service: Challenges and Opportunities," Proc. 25th IEEE Int'l Conf. Data Eng., Mar.-Apr. 2009.

[6] 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.

[7] A. Shamir, "How to Share a Secret," Comm. of the ACM, vol. 22, no. 11, pp. 612-613, 1979.

[8] M. Hadavi, E. Damiani, R. Jalili, S. Cimato, and Z. Ganjei, "AS5: A Secure Searchable Secret Sharing Scheme for Privacy Preserving Database Outsourcing," Proc. Fifth Int'l Workshop Autonomous and Spontaneous Security, Sept. 2013.

[9] "Transaction Processing Performance Council," TPC-C, http:// www.tpc.org, Apr. 2013.