ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Distributed Concurrent and Independent Access to Cloud Database

Amanpreetsaini
Dept of IT,
MLR Institute of Technology,
Hyderabad.

B.Divyalakshmi
Dept of IT,
MLR Institute of Technology,
Hyderabad.

Mr.Ram Mohan Rao
Associate Professor,
Dept of CSE,
MLR Institute of Technology,
Hyderabad.

Abstract:

Allocatingcritical data in the hands of a cloud provider should come with the assurance of security and availability for data at which is under usage and while using the data. There are many alternatives existing for storage services while data confidentiality solutionsfor the database as a service prototype are still immature. We propose a architecture that consolidate cloud database services with data confidentiality and the possibility of executing concurrent operations on encrypted data The proposed architecture has the further advantage of eliminating intermediate proxies that limit the availability, properties that are intrinsic incloud-based solutions.Multiple customers' data is getting stored in a single data centres or repository. Admin will have access to all the data. But customer can have access to only their data. At the same time, multiple customers will access the data. Need to setup a multi-tenant repository over cloud. Build an UI interface to upload or download document from repository but through proper authorization or authentication.

I.INTRODUCTION:

Cloud computing is the use of computing resources (hardware and software) that are delivered as a service over a network (typically the Internet). The name comes from the common use of a cloud-shaped symbol as an abstraction for the complex infrastructure it contains in system diagrams. Cloud computing entrusts remote services with a user's data, software and computation. Cloud computing consists of hardware and software resources made available on the Internet as managed third-party services. These services typically provide access to advanced software applications and high-end networks of server computers. On-demand self-service is consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service's provider.

Broad network access capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms.

II.ARCHITECTURE DESIGN:

SecureDBaaS is designed to allow multiple and independent clients to connect directly to the untrusted cloud DBaaS without any intermediate server. It is the overall architecture. We assume that a tenant organization acquires a cloud database service from an untrusted DBaaS provider. The tenant then deploys one or more machines (Client 1 through N) and installs a SecureDBaaS client.

Data Management:

We assume that tenant data are saved in a relational database. We have to preserve the confidentiality of the stored data and even of the database structure because table and column names may yield information about saved data. We distinguish the strategies for encrypting the database structures and the tenant data. Encrypted tenant data are stored through secure tables into the cloud database.

Metadata Management:

Metadata generated by SecureDBaaS contain all the information that is necessary to manage SQL statements over the encrypted database in a way transparent to the user. Metadata management strategies represent an original idea because SecureDBaaS is the first architecture storing all metadata in the untrusted cloud database together with the encrypted tenant data. SecureDBaaS uses two types metadata. Database metadata are related to the whole database. There is only one instance of this metadata type for each database. Metadata are associated with one secure table. Each table metadata contains all information that is necessary to encrypt and decrypt data of the associated secure table.

Concurrent SQL Operations:

The support to concurrent execution of SQL statements issued by multiple independent (and possibly geographically distributed) clients is one of the most important benefits of SecureDBaaS with respect to state-of-the-art solutions. Our architecture must guarantee consistency among encrypted tenant data and encrypted metadata because corrupted or out-of-date metadata would prevent clients from decoding encrypted tenant data resulting in permanent data losses. A thorough analysis of the possible issues and solutions related to concurrent.

SQL operations on encrypted:

Tenant data and metadata is contained available in the online supplemental material. Here, weremark the importance of distinguishing two classes ofstatements that are supported by SecureDBaaS: SQL operationsnot causing modifications to the database structure,such as read, write, and update; operations involvingalterations of the database structure through creation,removal, and modification of database tables.

III.SECURITY ISSUES IN CLOUD:

The security will be analysed in terms of two aspects, that is, the confidentiality of data and the authorization of duplicate check. We suppose that all the files are sensitive and needed to be fully protected against both public cloud and private cloud. Under this assumption, two kinds of adversaries are considered, that is, adversaries who aim to extract secret information as much as possible from both public cloud and private cloud, and internal adversaries who aim to obtain more information on the file from the public cloud and duplicate-check token information from the private cloud outside of their scopes. The data will be encrypted in our deduplication system before outsourcing to the storage cloud to maintain the confidentiality of data. The data is encrypted with the traditional encryption scheme and the data encrypted with such encryption method which guarantees the security of data. System address the problem of privacy preserving deduplication in cloud computing and propose a new deduplication system supporting for Differential Authorization and Authorized Duplicate Check. Each authorized user is able to get his/her individual token of his file to perform duplicate check based on his privileges. Under this assumption, any unauthorised user cannot generate a token for duplicate check out of his privileges or without the aid from the private cloud server. Authorized user is able to use his/her individual private keys to generate query for certain file and the privileges he/she owned with the help of private cloud, while the public cloud performs the duplicate check directly and tells the user if there is any duplicate. The security requirements considered in two folds, including the security of data files and security of file token. Unauthorized users without appropriate privileges or file prevented from getting or generating the file tokens for duplicate check of any file stored at the Storage cloud. The users are not allowed to collude with the public cloud server. It requires that any user without querying the private cloud server for some file token, he cannot able to get any useful information from the token, which includes the privilege or the file information and to maintain the data confidentiality unauthorized users without appropriate privileges or files, prevented from access to the underlying plaintext stored at Storage cloud. We describe how to initialize SecureDBaaS architecture from a cloud database service acquired by a tenant from a cloud provider. We assume that the DBA creates the metadata storage table that at the beginning contains just the database metadata, and not the table metadata. The DBA populates the database metadata through the Secure DBaaS client by using randomly generated encryption keys for any combinations of data types and encryption types, and stores them in the metadata storage table after encryption through the master key.

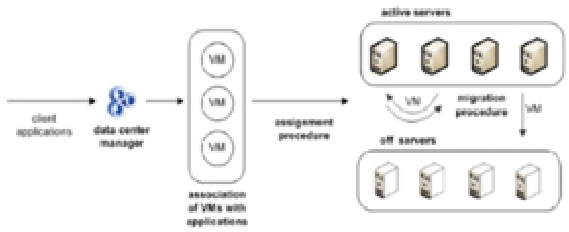

Fig 1: Process Flow Diagrams for Existing &Proposed System

PTC Integrity Platform:

PTC Integrity is a unique process-based change and configuration management software development platform, supporting enterprise solutions for engineering change management and application lifecycle management (ALM). Integrity enables organizations to solve the complexity involved in developing today's products and applications. With seamless, collaborative management of all activities and assets, Integrity delivers greater transparency, better productivity and shorter cycle times across the entire development lifecycle.

The Spiral Model :

The spiral model, originally proposed by Boehm, is an evolutionary software process model that couples the iterative nature of prototyping with the controlled and systematic aspects of the linear sequential model. It provides the potential for rapid development of incremental versions of the software. Using the spiral model, software is developed in a series of incremental releases. During early iterations, the incremental release might be a paper model or prototype. During later iterations, increasingly more complete versions of the engineered system are produced. A spiral model is divided into a number of framework activities, also called taskregions.Typically, there are between three and six task regions.

Planning—tasks required to define resources, timelines, and other projectrelate information.

Risk analysis—tasks required to assess both technical and managementrisks.

Engineering—tasks required to build one or more representations of theapplication.

Construction and release—tasks required to construct, test, install, andprovide user support (e.g., documentation and training).

Customer evaluation—tasks required to obtain customer feedback basedon evaluation of the software representations created during the engineeringstage and implemented during the installation stage.Each of the regions is populated by a set of work tasks, called a task set, that areadapted to the characteristics of the project to be undertaken. For small projects, thenumber of work tasks and their formality is low. For larger, more critical projects,each task region contains more work tasks that are defined to achieve a higher levelof formality.

Customer communication—tasks required to establish effective communicationbetween developer and customer.

Setup Phase:

We describe how to initialize SecureDBaaS architecture from a cloud database service acquired by a tenant from a cloud provider. We assume that the DBA creates the metadata storage table that at the beginning contains just the database metadata, and not the table metadata. The DBA populates the database metadata through the Secure DBaaS client by using randomly generated encryption keys for any combinations of data types and encryption types, and stores them in the metadata storage table after encryption through the master key. Then, the DBA distributes the master key to the legitimate users. User access control policies are administrated by the DBA through some standard data control language as in any unencrypted database.

Plain name: The name of the corresponding column ofthe plaintext table.

Coded name: The name of the column of the secure. This is the only information that links acolumn to the corresponding plaintext columnbecause column names of secure tables are randomlygenerated.

Secure type: The secure type of the column, as defined SecureDBaaS client to beinformed about the data type and the encryption policies associated with a column.

Encryption key: The key used to encrypt and decryptall the data stored in the column.

IV.EXPERIMENTAL RESULTS:

We demonstrate the applicability of SecureDBaaS todifferent cloud DBaaS solutions by implementing andhandling encrypted database operations on emulated andreal cloud infrastructures. The present version of theSecureDBaaS prototype supports PostgreSQL, MySQL, and SQL Server relational databases.

As a first result, we can observe that porting SecureDBaaS to different DBMS required minor changes related to the database connector,and minimal modifications of the codebase. We refer to Appendix C, available in the online supplemental material,for an in-depth description of the prototype implementation.

REFERENCES:

1.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 andImplementation, Oct. 2010.

2.J. Li, M. Krohn, D. Mazie`res, and D. Shasha, "Secure UntrustedData Repository (SUNDR)," Proc. Sixth USENIX Conf. Operating Systems Design and Implementation, Oct. 2004.

3.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.

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

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

6.R.A. Popa, C.M.S. Redfield, N. Zeldovich, and H. Balakrishnan,"CryptDB: Protecting Confidentiality with Encrypted QueryProcessing," Proc. 23rd ACM Symp. Operating Systems Principles,Oct. 2011.

7.H. Hacigu¨mu¨ s¸, B. Iyer, C. Li, and S. Mehrotra, "ExecutingSQL over Encrypted Data in the Database-Service-ProviderModel," Proc. ACM SIGMOD Int'l Conf. Management Data, June2002.

8.J. Li and E. Omiecinski, "Efficiency and Security Trade-Off inSupporting Range Queries on Encrypted Databases," Proc. 19thAnn. IFIP WGWorking Conf. Data and Applications Security.

Author Details:

Amanpreetsaini
Dept of IT, MLR Institute of Technology,
Hyderabad.

B.Divyalakshmi
Dept of IT, MLR Institute of Technology,
Hyderabad.

Mr.Ram Mohan Rao
Associate Professor, Dept of CSE,
MLR Institute of Technology,
Hyderabad.