Volume 2, Issue 12, December-2015, pp. 1257-1262 ISSN (O): 2349-7084

International Journal of Computer Engineering In Research Trends

Available online at: www.ijcert.org

A Survey: Query Processing Techniques for Secure Cloud Databases

[1]N.Swetha, [2]Prof. S Ramachandram

[1]Research Scholar, Dept. of CSE, Osmania University, Hyderabad

[2]Principal, Dept. of CSE, Osmania University, Hyderabad.

Abstract: The Major challenge in cloud computing is to manage distributed cloud. Traditional relational database management systems (RDBMS) are a choice but they are not compatible to measure crossways large clusters of distributed servers. Therefore changeovers to RDBMS have been developed. The development of new database management systems (DBMS) for the cloud computing environment or flexibility of the existing systems to the cloud computing environment is a critical component of cloud computing research. This paper focuses on the study of a relational database system based on holomorphic encryption schemes to preserve the integrity and confidentiality of the data. And also examines various SQL queries over encrypted data. And this paper helps in focusing where research is needed.

***Keywords*:** database management systems, relational database management systems, Homomorphic encryption, Cloud Data bases.

1. INTRODUCTION

Cloud computing is an attractive solution that can provide low cost storage and processing capabilities for government agencies, hospitals, and small and medium enterprises. It has the advantage of reducing the IT costs and providing more services for the requesting parties through making specialized software and computing resources available. However, there are major concerns that should be considered by any organization migrating to cloud computing. The confidentiality of information as well as the liability for incidents affecting the infrastructure arises as two important examples in this context. Indeed, cloud computing poses several data protection risks for the cloud's clients and providers. For example, the cloud's client may not be aware of the practices according to which the cloud's provider processes the stored data. Therefore, the cloud's client cannot guarantee that the data are processed (for example, altered or deleted) in a legal and accepted manner. All of the above mentioned issues can be resolved if the data in the cloud are stored and processed in encrypted form. The latter is possible if the encryption scheme can support addition and multiplication of the encrypted data. Many encryption schemes support one of these operations, like the encryption schemes in .A cryptosystem which supports both addition and multiplication (referred to as the holomorphic encryption scheme) can be effective data protection, and enables the construction of programs that receive encrypted input and produce encrypted output. Since such programs do not decrypt the input, they can be run by an un-trusted party without revealing their data and internal states. Such programs will have great practical implications in the outsourcing of private computations, especially in the

 DOI: 10.22362/ijcert/2015/v2/i12/1257

context of cloud computing. Homomorphic cryptosystems have received valuable attention in the literature, see [6][7]. In theory, the data can be encrypted by the client, and then sent to the cloud's provider for storage or processing. Only the client holds the decryption keys necessary to read the data. Despite the fact that this type of processing may increase the amount of computing time, the benefits associated with it are worth the processing overhead. Indeed, this model of computing can preserve the confidentiality and integrity of the data while delegating the storage and processing to an un-trusted third party.

A. Database Management System

The relational model was first introduced by Edgar Codd in 1970. It uses a collection of tables to represent data items and their relationships. It is wellsuited for Online Transaction Processing Systems. These applications are based on the ACID properties i.e. Atomicity, Consistency, Isolation, and Durability (ACID). These traditional DBMS are also known as row-oriented DBMS, as they store the data row-byrow. They keep all the information about an entity together and are preferred when queries access the data regarding an entity. These row-oriented DBMS includes Oracle, DB2, SQLServer, Teradata

B. Cloud Databases

The data is distributed across several machines in network, so efficient management of data is a big worry for organizations using services of cloud. Here, the most important requirement of database management system is of scalability. Though RDBMS are robust and a vast majority of current database systems are based on the relational model yet they are not well-suited to scaling across large clusters of servers as they are not designed to be distributed. So it is difficult to ensure consistency, referential integrity and query performance in a distributed relational environment. Structured Query Language (SQL) is the dominant language of RDBMS. But SQL databases don't scale .

Hence alternatives to relational database management systems have been developed. Some of them are:

1. Column-oriented DBMS is a database management system that stores the data by column. It is faster to read and allows a more efficient compression of the data. Hence query processing time is decreased. These are wellsuited for online analytical processing systems (OLAP) and data warehouses. Some of the database management systems which are already in the market are Vertica (a commercial version of C-store), SADAS as well as open-source DBMS like LucidDB and MonetDB.

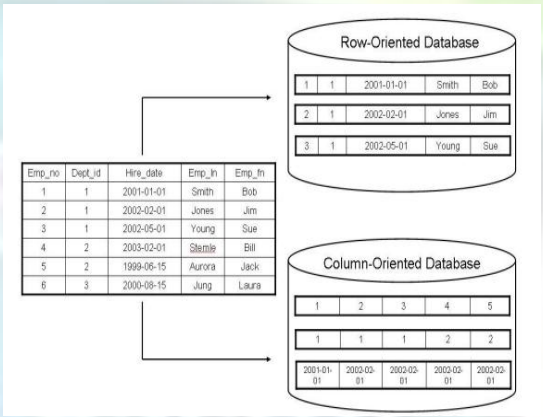

Fig 1. Row and Column Oriented database[3]

2. Key-value Stores (non-relational DBMS or NoSQL datastores) are used for storing large scale data & provide easy access. Its data models are schema-free, join-less and horizontally scaled. Here, domain is just like a table, but there is not a predefined schema. It is like a bucket where we can put the items. Items are identified by keys and a given key can have dynamic set of attributes attached to it. Data is created, updated, deleted and retrieved using API method calls. Some of its projects include Dynamo[4] used by Amazon.com, Google's Bigtable used in the Google's application, Cassandra used by Facebook for inbox search and project Voldemort used by LinkedIn.

3. Mapreduce[5] is a distributed framework that allows us to process large amount of data in parallel approach.

 DOI: 10.22362/ijcert/2015/v2/i12/1257

Programmers create different map and reduce functions on the basis of user queries. The data files are stored in distributed file system (DFS). This approach is being used in Google's web search service, for the generation of data stored in Bigtable.

4. Streaming Databases: As the name implies, streaming databases treat data as a single stream passing under the "head" of the database engine, which must make an immediate decision whether to store it, process it, use it to generate an alert and/or re-route it to some other appropriate data source. Thus, the streaming database is often used as a "rapid-response" approach, unable to bring to bear the full context of a data warehouse in analyzing the next bit of information but far quicker to note obvious important and time-critical data.

2. LITERATURE SURVEY

Query efficiency is achieved by employing a pure key-value data model where, both key and value are arbitrary byte strings (e.g. Dynamo), or its variant (as in Bigtable), where key is an arbitrary byte string and value is a structured record consisting of a number of named columns. But these solutions lack support for secondary indexes, range queries and multidimensional queries .

Mapreduce [5], which is a programming model and a framework for processing large sets of raw data. A map-reduce program consists of two functions: Map and Reduce. The Map function processes the input data by distributing them to worker nodes for parallel computation and produces a set of intermediate results as key-value pairs, while the reduce function aggregates all the intermediate results with the same key from each node to produce the result. It can be used for structured data analysis of large sets. The limitations of Mapreduce as given by are:

- It produces the necessary secondary indices in an offline batch manner. Hence, secondary indexes are not up-to-date. So newly inserted rows cannot be queried until they are indexed
- It does not provide data schema support, declarative query language and cost-based query optimizations.

Raykova et al. [8] extended the PIR approach by proposing a secure anonymous search system. The system employs keyword search such that only authorized clients have access to their blocks. This system is capable of mapping the database content to the appropriate client, thus guaranteeing the privacy of the data and the query. The ultimate target of Raykova's system is to ignore the identity of the client while protecting the database from malicious queriers.

Shang et al. [9] tackled the problem of protecting the database itself. The problem is studied through monitoring the amount of data disclosed by a PIR protocol during a single run. The information attained from the monitoring process is used to understand how a malicious querier can conduct attacks to retrieve excessive amount of data from the server.

PIR has also been used to develop authentication systems. Nakamura et al. [10] constructed a system with three components, a querier that initiates requests, an authentication-server that processes these requests, and a database that returns the appropriate data in response to the request. This system ensures the security of data and the anonymous communication between the querier and the database. Yinan and Cao [11] used the PIR approach to propose a system that controls the access to the database. According to this system, the privacy of data is enforced by enabling each authorizer to give or deny access to his/her own data with a hierarchical authorization access right scheme.

Among the most important criteria in PIR protocol are the communication cost and the amount of data sent back to the querier. The trivial solution of the PIR protocol is to send back the entire database to the client. However, this solution is expensive, even for a simple request that results in retrieving two matching records. Other approaches proposed to retrieve only the requested data, by using replicated databases that are stored at multiple servers. In this case, the request is forwarded to all servers. With this approach, although we deal with multiple replicated databases, the privacy is better protected. However, this approach is still complicated and may result in extended processing and communication times. Gentry et al. [12] proposed a scheme to retrieve a bit or a block from a database with a constant communication rate. Melchor et al. [13] proposed a scheme that reaches the available

 DOI: 10.22362/ijcert/2015/v2/i12/1257

data with a reasonable communication cost while achieving lower computational cost compared to other PIR protocols.

3. SECURED SQL OPERATIONS

A secure database system that processes SQL queries over encrypted data. Parameters of the queries are encrypted by the client and sent to the server for processing. The latter performs the requested operation and returns encrypted results to the client.

Fig 2. Secure data retrival

Source from:[14]

4. PERFORMANCE OF DATABASE MANAGEMENT SYSTEMS:

The objective of performance enhancement is to minimize the response time for each query and to maximize the throughput of the database server. The performance of the database management system can be determined by the following factors:

1. **System Level Issues:** These issues can perform serious performance degradation. It occurs if Utilization of CPU is high , Loads of I/O changes frequently , Hardware and software is not configured properly ,Operating System of virtual machines and hypervisor
2. **Database design:** The database design is one of the most important decisions which have to be made carefully as the performance of the queries depends on: File organization techniques, Constraints on the attributes, Normalization and De-normalization of relations. , Indexing.
3. **Query Processing and Optimization techniques:** Query processing and optimization are the main components of the database management system. The function of query processor [2] is to transform the query written in high-level language into a correct and efficient execution plan expressed in low-level language. As there are many ways to execute the same query, the aim of query optimization is to choose an efficient execution plan for processing a query. It chooses the one that minimizes the resources.. It takes information from the system catalog. The optimizer are required to consider factors such as the order in which to join the tables, the number of rows for each join when calculating an optimal access path, the algorithm to be used for performing the joins. In a distributed environment like cloud, data is distributed to a number of sites, stored in its entirety on all sites or spilt on many sites. Here the query is processed and optimized in a different way. There are various issues which needs to be considered like, which copy of the data is to be used i.e. site selection, amount of data that needs to be transmitted from its location to the execution site, relative processing speed at each site and transmitting the final result to the site where the query is issued. The cost of the plan depends on these issues.

5. HOMOMORPHIC ENCRYPTION SCHEME

In [6], Gentry proposed a fully holomorphic encryption scheme that enables to perform an arbitrary number of arithmetic operations (i.e. addition and multiplication) on encrypted data. The components of the encryption scheme are described below.

Arithmetic Operations: Addition and multiplication can be performed on clear text by simply adding and multiplying the ciphertext, respectively.

$$\varepsilon(m1 * m2) = \varepsilon pk(m1) * \varepsilon pk(m2)$$

$$\varepsilon(m1 + m2) = \varepsilon pk(m1) + \varepsilon pk(m2)$$

The output ciphertext c* consists of c together with the result of post-processing the resulting ciphertext .

1. Additive Homomorphic Encryption:

A Homomorphic encryption is additive, if:

 DOI: 10.22362/ijcert/2015/v2/i12/1257

$$\text{Enc}(x \oplus y) = \text{Enc}(x) \otimes \text{Enc}(y)$$

$$\text{Enc}\left(\sum_{i=1} m_i\right) = \prod_{i=1} \text{Enc}(m_i)$$

Suppose we have two ciphers C1 et C2 such that:

C1 = gm1. r1n mod n2
C2 = gm2. r2 n mod n2
C1.C 2 = gm1. r1n. gm2. r2 nmod n2 = gm1+ m2 (r1r2) n mod n2

6. SQL QUERIES OVER AN ENCRYPTED DATABASE.

The user can specify a search criterion through a database. Then, the client software encrypts the parameters of the query, corresponding to the search criterion, and sends it to the appropriate server. The server retrieves the requested record (blind processing) from the database and returns it to the client. The client software decrypts the record and displays it to the user. We built a simple medical application containing 10 patients' records. In Figure 3, we show the result of the SELECT query. This is how the result appears in a screenshot of the client side of our built application. The application supports the following SQL operations:

- SELECT with wildcard characters (*, ?) and relational operators (< >).
- UPDATE with wildcard characters (*, ?) and relational operators (< >).
- DELETE with wildcard characters (*, ?) and relational operators (< >).
- Statistical operations like COUNT and AVG.

The execution of SQL statements over encrypted data is feasible. However, the time required to execute these statements is very high and therefore is not suitable for real-time transactions that involve a large database (i.e. several terabytes database). This drawback is mainly due to the homomorphic encryption scheme. In fact, there might be more efficient techniques to optimize the implementation, that is, one could perform encryption only when it is necessary, since the noise value can be bounded; however, we do believe that a more practical Homomorphic cryptosystem is yet to be developed.

7. Conclusion:

In this paper we observe various alternatives to relational database management systems and also understand a secure database system that processes SQL queries over encrypted data. We have been review on Performance of Database Management Systems.

Future scope of this survey is to focus on improve the performance by Processing can be parallelized in order to take advantage of multiple processors executing the encrypted requests

REFERENCES:

[1] Wiggins, A. (2009), SQL Databases Don't Scale. http://adam.heroku.com/past/2009/7/6/sql_databases_dont_scale/

[2] Abadi, D. J. (2009), Data Management in the Cloud: Limitations and Opportunities. IEEE Data Engineering Bulletin, Volume 32, Number 1, March 2009, pp. 3-12

[3] Garret (2010), What is a Column Oriented Database? http://www.columnorienteddatabase.com/

[4] DeCandia, G., D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels(2007) Dynamo: Amazon's highly available key-value store, In SOSP, pp. 205–220

[5] Dean, J. and S. Ghemawat (2004) Mapreduce: Simplified data processing on large clusters. Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation ACM pp. 137–150

[6] C. Gentry, Computing arbitrary functions of encrypted data, Commun. ACM, Vol. 53, No. 3., pp. 97-105, March 2010.

[7] C. Gentry, A fully homomorphic encryption scheme. PhD thesis, Stanford University, 2009.

[8] M. Raykova, B. Vo, and S. Bellovin, Secure Anonymous Database Search, CCSW'09, pp. 115-126, Chicago, Illinois, USA, November 13, 2009.

 DOI: 10.22362/ijcert/2015/v2/i12/1257

[9] N. Shang, G. Ghinita, Y. Zhou, and E. Bertino, Controlling Data Disclosure in Computational PIR Protocols. ASIACCS'10, pp. 310- 313, Beijing, China, April 13–16, 2010.

[10] T. Nakamura, S. Inenaga, D. Ikeda, K. Baba, H. Yasuura, Anonymous Authentication Systems Based on Private Information Retrieval. Networked Digital Technologies. NDT '09, pp.53-58, 28- 31 July 2009.

[11] S. Yinan and Z. Cao, Extended Attribute Based Encryption for Private Information Retrieval. Mobile Adhoc and Sensor Systems, 2009. MASS '09, pp. 702-707, 12-15 Oct. 2009.

[12] C. Gentry and Z. Ramzan, Single-Database Private Information Retrieval with Constant Communication Rate. ICALP 2005, LNCS 3580, pp. 803–815, 2005.

[13] C. A. Melchor and P. Gaborit, A Fast Private Information Retrieval Protocol. ISIT 2008, pp. 1848-1852, Toronto, Canada, July 6 - 11, 2008.

 DOI: 10.22362/ijcert/2015/v2/i12/1257