MATRIX Academic International Online Journal of Engineering and Technology
ISSN: 2348–3326 © MAIOJET
Volume III Issue I, April 2015

Efficient Query Integrity Management in Cloud backing with Grid Computing: An appraisal

Ms. Ashvini A. Todkar[1], Mr S. G. Sutar[2]

[1]*Research Scholar, Computer Science and Engineering, ADCET, Ashta, Maharashtra, India.*

todkaraa@sbgimiraj.org

[2]*Assistant Professor, Computer Science and Engineering, ADCET, Ashta, Maharashtra, India.*

sutarsandeep07@gmail.com.

***Abstract*— In cloud environment, queries are processed by the computational servers over the externally stored databases. There might be possibility of problem of data integrity. To resolve this problem different protection techniques are used (like salts, buckets, markers etc). But these protection techniques are designed for single computational server. Existing system is designed only for simple join query. It could not be applicable for parallel joins or chain of joins. To reduce the overhead of single computational server, system presented here gives distributed environment at computational server using Grid computing. It increases the performance of system.**

In this system, client request is divided into subtasks at the computational server who acts as manager. Then subtasks are forwarded to number of workers among the grid environment. The worker independently executes their tasks & result is given back to manager. The system presented here, provides enhanced security to data, as data is in encrypted form at computational server.

***Keywords*—** Cloud Server, Infrastructure as a Service, Map Reduce Query Language, Platform as a Service, Resource Description Framework, Rainbow Table, Software as a Service,SPRAQL Protocol and RDF Query Language, Computational Server, Storage Server.

I. INTRODUCTION

Now a days, the use of cloud computing is rapidly increasing due to on-demand services. Cloud Computing is a technology that uses the internet and central remote servers to maintain data and applications. Cloud Computing allows consumers and businesses to use applications without installation and access their personal files at any computer with internet access.

In cloud computing, there are different providers of storage services which offer continuous availability of stored data, with high bandwidth and reliability guarantees. A storage service offered by storage-as-a-service provider fetches issues related to data integrity, privacy and security. As cloud uses distributed environment, correctness of data may be get affected.

The data present at different storage servers is retrieved through query evaluation. Here, it uses different queries like, Join, Aggregate, etc. These queries are executed by computational server present into cloud environment

II. RELATED WORK

Sabrina de Capitani di Vimercati, Sara Foresti, Sushil Jajodia have suggested the innovative and effective integrity protections techniques at limited cost. It includes techniques like salts, markers, twins, buckets to improve the correctness of data. The proposed system uses the techniques mentioned here [1].

Ling Hu,Wei-Shinn Ku, SpiridonBakiras, and Cyrus Shahabi have proposed an Outsourced Spatial Database model in which neighborhood information derived from the Voronoi diagram of the underlying spatial data set and can handle fundamental spatial query types. They have suggested method which separates the authentication information from the spatial index, thus allowing efficient query processing at the service provider [2].

Leonidas Fegaras, Chengkai Li, Upa Gupta have proposed an optimization framework for MapReduce queries. This framework [3] support to MapReduce Query Language.Tasks involved in MapReduce is performed on the clusters.

Dinesh.C, has proposed an innovative method for data integrity and dynamic storage in cloud. He has suggested read protocol algorithm and multi-server data comparison algorithm for every data upload which is useful for data recovery management. The major advantage is to avoid server failure and any unexpected error. This [6] system support to put one server restore point in cloud server database for efficient data back up or restore using multi server data comparison method.

III. NEED OF PRESENT WORK

Following are some issues identified in cloud computing:

A. Integrity

"Data Integrity" as the word in itself explains the "completeness" or "wholeness" of the data which is the basic requirement of the information technology. As Data Integrity is an important in databases similarly integrity of Data Storage

is an essential in the cloud, it is a key aspect that affects on the performance of the cloud. The data integrity provides the legality of the data, assuring the reliability or regularity of the data. It is the complete mechanism of writing of the data in a reliable manner to the persistent data storage which can be retrieved in the same format without any changes later.

B. *Security and Privacy*

In cloud computing, major security issues are Loss of control, Lack of trust (mechanisms) and Multi-tenancy. Here, client loses his control because everything i.e. data, applications and resources are located at the cloud service provider. User identity management, user access control rules and security policies are managed by cloud service provider.
Client relies on cloud service provider to ensure data security and privacy, resource availability, monitoring and repairing of services/resources.

C. *Problem Statement*

Our ultimate objective is to provide solution to the issues identified here. This objective can be achieved by our proposed "Efficient Query Integrity Management in Cloud backing with Grid Computing" system. The system presented here, solves integrity problem by making use of protection techniques named as markers, twins, salts and buckets. The issues related to security and privacy can be resolved by making use of encryption technique named as 'encryption-on-fly'. Also, it will use grid computing at computational server side which will improve the performance in terms of memory and time.

This system implements a mechanism to process data in encrypted form. Also, it supports for parallel joins and chain of joins through the Grid Computing.

IV. APPROACHES

In this system, client requests to cloud server for query. Only authorized client can forward his query to cloud server. Cloud server divides that query into sub queries, encrypts them and sends to computational server for further processing. At computational server, grid manager distributes the tasks among workers. Workers independently execute given task by fetching data from storage server. Then workers send result back to manager. Manager combines all the results and final result is sent to cloud server. Finally client gets result from cloud server.

Our system can be divided into following modules: Login Module, Cloud Server Module, Implementation of Grid Computing, and Storage Server Module.

A. *Login Module*

This module implements a mechanism for authentication and authorization. Only authorized clients can send request for services to the cloud server. After, successful execution of request result is sent to the client. As shown in fig. 1, step 1 indicates request and step 8 indicates result for request.

B. *Cloud Server Module*

In this module, cloud server takes authenticated query as input. It divides the user query into sub queries. Also, it encrypts the sub query with markers, buckets, salts [9], twins. The encryption on data is done at storage server. After getting result from computational server, it decrypts the result of queries. Finally, result is given to user.

C. *Implementation of Grid Computing*

This module attempt to create a grid environment with following steps:
Implementation of Grid Manager.
Implementation of Grid Sub Manager or Worker.
Forwarding of queries to the manager.
Collection of results from Grid manager.

D. *Storage Server Module*

This module evaluates requests from sub manager. The storage server maintains data in its original form. To maintain the correctness of data, cloud server provides markers, salts, buckets, twinning condition as encryption parameters. Storage server creates the group of tuples which satisfies condition mentioned in the query. The resultant group is forwarded to grid.

E. *System Design*

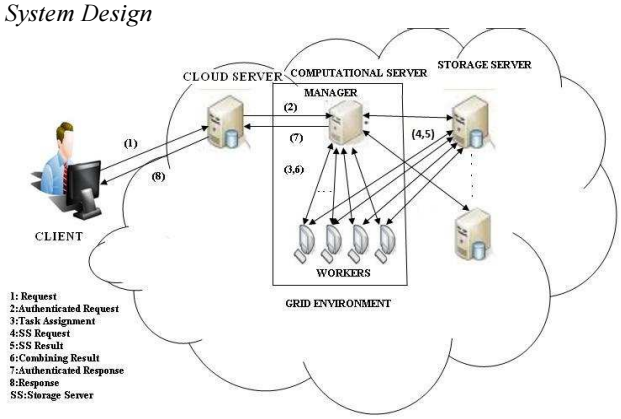

Fig. 1 The System Architecture

Fig. 1 shows the system architecture of our proposed system. Here, client sends request to cloud server. If that user is valid user, then cloud server forwards query with string containing sub-query, number of salts, buckets, markers and twining condition to the computational server. Computational server, here we call it as manager in grid environment partition that string into number of tasks. Manager assigns tasks to the workers using map () function. In map () function (key, value) pair is used for table name and attributes.

Worker sends request to desired storage server for retrieving the group of tuples which satisfies the condition

mentioned in the task. After, getting results from storage server worker forwards result to manager.

Manager combines the results from workers using reduce () function. Data presented at the computational servers is in the encrypted form only. Final result is sent to cloud server. Cloud server decrypts final result for getting original result. Result is given to client. In this system use of encryption at every stage keeps track on data integrity and confidentiality.

Following algorithm is used to implement cloud server module:

Step 1: Divide query into sub queries.
Step 2: Add encryption - decryption functions [10], corresponding keys and salts to the sub queries.
Step 3: Wrap above contents into the strings.
Step 4: Distribute these strings over the workers present in the grid.
Step 5: Collect the results from each worker.
Step 6: Combine the final result & send to cloud server.

Following algorithm is used to implement storage server module:

Step 1: Get string which contains sub query, encryption-decryption function [10] & keys.
Step 2: Decrypt sub query & execute it.
Step 3: Add markers to result.
Step 4: Encrypt the result & send it to worker.

V. Experimental Setup and Results

The experiment is carried out using jdk1.6.0_23 on single machine with windows 8 operating system. The modules are implemented using JAVA. Cloud environment setup is done with three nodes i.e. controller, compute and network using OpenStack [7]. GRID environment setup is done with alchemy [8].

VI. Result Analysis

Following table-I shows sample join query with presented system and without system. Execution time is given hh:mm:ss format.

TABLE I
QUERY EXECUTION TIME WITH GRID & WITHOUT GRID

Join Query	Environment	Time(hh:mm:ss)
SELECT p1.PID, f1.PNAME, p2.PID, f2.PNAME FROM PLAYER f1, PLAYER f2, PLAYS p1 FULL OUTER JOIN PLAYS p2 ON p1.PID < p2.PID AND p1.TID = p2.TID GROUP BY p1.PID, f1.PID, p2.PID, f2.PID HAVING Count(p1.PID) = Count(*) AND Count(p2.PID) = Count(*) AND p1.PLRID = f1.PID AND p2.PID = f2.PID;	Without GRID	00:00:0.31
	USING GRID	00:00:0.006

TABLE II
SAMPLE QUERY & SUB QUERIES

Query	Sub Queries
SELECT p1.PID, f1.PNAME, p2.PID, f2.PNAME FROM PLAYER f1, PLAYER f2, PLAYS p1 FULL OUTER JOIN PLAYS p2 ON p1.PID < p2.PID AND p1.TID = p2.TID GROUP BY p1.PID, f1.PID, p2.PID, f2.PID HAVING Count(p1.PID) = Count(*) AND Count(p2.PID) = Count(*) AND p1.PLRID = f1.PID AND p2.PID = f2.PID;	1.Select count (p1.pid) =count(*) from plays p1 , player f1 where p1.pid=f1.pid ; 2. Select count (p2.pid) =count(*) from plays p2, player f2 where p2.pid =f2.pid; 3. Select * from player f1,player f2, plays p1, plays p2 group by p1.pid, p2.pid, f1.pid, f2.pid; 4. Select pid, pname from player f1, player f2, plays p1, plays p2 where p1.pid < p2.pid AND p1.tid=p2.tid;

Table-II shows sample query which is divided into sub queries. Sub queries mentioned in table are distributed among the workers. Workers execute them independently with the help of storage server.

VII. Conclusion & Future work

In this paper, we have provided solution to the problem of query integrity through "Efficient Query Integrity Management in Cloud backing with Grid Computing". The system presented here is applicable for grid environment. It is also used for parallel joins or chain of joins. Powerful encryption technique i.e. salt encryption is used in this system.

This paper only present abstract work. In future, we are applying these things to real time applications.

REFERENCES

[1] Sabrina de Capitani di Vimercati,SaraForesti, SushilJajodia , "Integrity for join Queries inthe Cloud IEEE TRANSACTIONS ON CLOUD COMPUTING, vol. 1, no. 2, July-December2013.

[2] Ling Hu,Wei-Shinn Ku,SpiridonBakiras, and Cyrus Shahabi "Spatial Query Integrity withVoronoi Neighbors," IEEE Trans.KNOWLEDGE AND DATA ENGINEERING,vol.25, no.4, April 2013.

[3] Leonidas Fegaras, Chengkai Li,UpaGupta, "An Optimization Framework for Map-ReduceQueries," University of Texas at Arlington.

[4] Li Lin, Li Qingzhong, Kong Lanju and Shi Yuliang, " Efficient Query Integrity Protectionfor Multi-tenant Database," International Journal of Database Theory and Application,Vol.7, No.3 , 2014.

[5] Mohammad Farhan Husain, Latifur Khany, Murat Kantarciogluz and Bhavani Thuraisingham,"Data Intensive Query Processing for Large RDF Graphs Using Cloud Computing.

[6] Dinesh.C, "Data Integrity and Dynamic Storage Way in Cloud Computing," International Journal of Computer Applications (0975 8887), in Cloud Computing.,vol. 31 No.6, October 2011.

[7] OpenStack setup guide homepage [online]. Available: http://docs. openstack.org/havana/install-guide/install/apt/content/ch_overview.html (accessed on 11-12-2014)

[8] Grid setup steps [online]. Available: http://www.cloudbus.org/~ alchemi/doc/0_6_1/index.html (accessed on 05-01-2015)

[9] Encryption techniques (salts) in JAVA [online]. Available: http://www. quicklyjava.com/salt-in-java/ (accessed on 20-12-2014)

[10] Java Cryptography by Jonathan B. Knudsen