CHAPTER - IV
AN INTELLIGENT BROKER BASED CREDENTIAL SERVICE FOR ENSURING CLOUD SECURITY

4.1 INTRODUCTION

In the previous chapter we have discussed proposed cloud service security model on the basis of brokerage functionalities also some of the promising areas were the trusted model could be applied is also highlighted. In this chapter, an improved security model on the basis of the cloud service Brokerage (CSB) is further explained with inclusion of well known data displacement strategy concept, namely “Data Partitioning”. By adhering vertical partitioning concept, an algorithm namely “vertical portioning” is being proposed and the proposal has been successfully implemented using a Cloud Simulator CloudSim and the results are discussed for validating the claim.

4.2 BASICS OF PARTITIONING

Partitioning allows a table, index, or index-organized table to be subdivided into smaller pieces, where each piece of such a database object is called a partition. Each partition has its own name, and may optionally have its own storage characteristics.

From the perspective of a database administrator [18], a partitioned object has multiple pieces that can be managed either collectively or individually. This gives the administrator considerable flexibility in managing partitioned objects. However, from the perspective of the application, a partitioned table is identical to a non-partitioned table; no modifications are necessary when accessing a partitioned table using SQL queries and DML statements. Partitioning can provide tremendous benefit to a wide variety of applications by improving performance, manageability, and availability. It is not unusual for partitioning to improve the performance of certain queries or maintenance operations by an order of magnitude. Moreover, partitioning can greatly simplify common administration tasks.
Partitioning also enables database designers and administrators to tackle some of the toughest problems posed by cutting-edge applications. Partitioning is a key tool for building multi-terabyte systems or systems with extremely high availability requirements.

There are basically two approaches for partitioning, they are as follows [19]

- Vertical partitioning
- Horizontal partitioning

4.2.1 Horizontal partitioning

It involves putting different rows into different tables. Horizontal portioning is like splitting up a table by rows: one set of rows goes into one data store, and another set of rows goes into a different data store.

4.2.2 Vertical partitioning

It involves creating tables with fewer columns and using additional tables to store the remaining columns. Vertical partitioning is like splitting up a table by columns: one set of columns goes into one data store, and another set of columns goes into a different data store. It involves creating tables with fewer columns and using additional tables to store the remaining columns. Normalization also involves this splitting of columns across tables, but vertical partitioning goes beyond that and partitions columns even when already normalized. Different physical storage might be used to realize vertical partitioning as well; storing infrequently used or very wide columns on a different device, for example, is a method of vertical partitioning. Done explicitly or implicitly, this type of partitioning is called "row splitting" (the row is split by its columns). A common form of vertical partitioning is to split dynamic data (slow to find) from static data (fast to find) in a table where the dynamic data is not used as often as the static. Creating a view across the two newly created tables restores the original table with a performance penalty, however performance will increase when accessing the static data e.g. for statistical analysis.
4.3 ADVANTAGES OF PARTITIONING

4.3.1 Performance Advantages

A table is partitioned based on a criterion such as the value for a particular column. If a query requests data with a particular selection condition that would eliminate a complete partition, Oracle automatically ignores that partition in executing the query. In this way, you can partition a large table to get the advantages of a smaller table. For instance, you may have a very large table with all of the orders for a year. Most queries, though, only request data for a single month. You could partition the table by month, so that these common queries would, in effect, be accessing a smaller table, while the complete set of data still would be available. The Oracle cost-based optimizer also takes this into consideration when deciding how to execute an SQL statement.

4.3.2 Maintenance Advantages

Most maintenance operations can be performed on a single partition. You can backup or recover a partition rather than the entire table. In this way, you can significantly reduce the time required to perform maintenance operations. In the above example of a table partitioned by month, you could perform daily backups on the current month's partition and less frequent backups on previous months. In an extremely large database, this can make the difference between having enough time to complete a backup in off-business hours or not.

4.3.3 Availability

You can also use partitioning to place partitions into different table spaces to improve availability. One table space can go down without affecting the other table spaces. If a table space becomes unavailable, the other table spaces and their partitions are still available. Returning to the table partitioned by months, if the table space that contains the partition for January's data goes down, all the other partitions are still available. Only queries that require January data will fail to execute. Splitting data into different partitions is also a time saver. When you reduce the amount of data in each partition, you also reduce the amount of time required to recover that partition.
4.4 PROPOSED SYSTEM ARCHITECTURE USING VERTICAL PARTITIONING

This section introduces the extended security model on the basis of vertical partitioning paradigm with the inclusion of data displacement strategies.

Figure 4.1 Proposed System Architecture using Vertical Partitioning

In our proposed system, as shown in Figure 4.1, we are introducing a new algorithm which will give the high end security for the cloud user’s data. In the proposed architecture there are different components like, Trusted Third Party Registration, Secured login in mechanism, Encryption of input data and Vertical Partitioning Algorithm. In this cloud architecture, the user registers his profile with the trusted third party. The trusted third party verifies the users profile and allowing entering into the service provider. In this service provider by giving the
login details the user can able to upload the data to the cloud as well as retrieve the data from the cloud. The end result of algorithm will be stored in cloud provider. In this proposed architecture high end security will be achieved when upload the user’s input to the cloud provider. If the user wants to retrieve the data, then the data available in the different databases are integrated, decrypted and shown to the cloud users. In this architecture, we have shown only the storage of data in the cloud. This architecture assures that no third party will access cloud users data.

4.5 EXPERIMENTAL SET UP – ALGORITHMIC IMPLEMENTATION OF PROPOSED MODEL

The experimental setup has been made with the aid of JDK 1.6 with the Net Beans IDE for making vertical partitioning and data displacement, Microsoft Access has been chosen as the designated backend. To simulate cloud infrastructure, a well known cloud simulator namely CLOUDSIM [55][58] has been used for the proposed experimental setup.

4.5.1 About Cloud - Sim

IT companies who are willing to offer some services in the Cloud can use a simulation-based approach to perform some benchmarking experiments with the services to be offered in dependable, scalable, repeatable, and controllable environments before real deployment in the Cloud. Therefore, they can test their services in a controlled environment free of cost, and through a number of iterations, with less effort and time. Also, by using simulation, they can carry out different experiments and scenarios to identify the performance bottlenecks of resources and develop provisioning techniques before real deployment in commercial Clouds. Therefore, CloudSim has been developed to fulfill these requirements by simulating and extensible Clouds.

CloudSim can be defined as “a new and extensible simulation framework that allows seamless modeling, simulation, and experimentation of emerging Cloud Computing infrastructure and application services”. Initially, the framework of CloudSim consists of multiple layers starting from the lowest layer of SimJava up
to the top layer of User Code. At the lowest layer, SimJava provides the base engine of the simulation that supports the implementation of core functionalities essential for the higher level frameworks of the simulation, like queuing and processing of events; formation of system components (services, hosts, brokers, VMs); interaction between these components, and administration of the simulation clock. On top of that layer is the GridSim layer which supports high-level and fundamental Grid components, such as networks, resources, data sets, and information services. Then, the CloudSim layer forms the next level of the architecture that extends the core functionalities of the GridSim layer.

This layer supports Cloud-based data center environments, including VMs, memory, storage and bandwidth. Also, this layer can manage instantiating and simultaneously implementing a large scale Cloud infrastructure composed of thousands of system components (VMs, hosts, data centers, and application). Finally, User Code is the top-most layer of the simulation toolkit, which reveals the configuration of functionality for the system components, such as the number and specification of hosts and the scheduling policies of the broker. At this layer, a developer can model and perform robust experiments and scenarios of Cloud environments based on custom policies and configurations already supported by the CloudSim, in order to evaluate and tackle some Cloud issues like the complexities of Cloud infrastructure and application.

4.5.2 Implementation of the proposal

The username and password is validated through login screen. Next, the user has to select the input data for the encryption using RSA and Elliptic Cryptography. The Encrypted data is splitted using vertical partitioning and placed in the different tables. When retrieving the data, the encrypted data is collected from the different tables and given to decryption mechanism. The following algorithm is implemented and tested in java platform.

A user of RSA creates and then publishes a public key based on the two large prime numbers, all along with a supplementary value. The prime numbers must be kept covert. Anyone can use the public key to encrypt a message, but with presently published methods, if the public key is large adequate, only someone with information of the prime factors can possibly decode the communication [51].
Choose $p = 3$ and $q = 11$

Compute $n = p \times q = 3 \times 11 = 33$

Compute $\varphi(n) = (p - 1) \times (q - 1) = 2 \times 10 = 20$

Choose $e$ such that $1 < e < \varphi(n)$ and $e$ and $n$ are co prime. Let $e = 7$

Compute a value for $d$ such that

$$(d \times e) \mod \varphi(n) = 1.$$ One solution is $d = 3 \mod (3 \times 7) = 20 = 1$

Public key is $(e, n) \Rightarrow (7, 33)$

Private key is $(d, n) \Rightarrow (3, 33)$

The encryption of $m = 2$ is $c = 2^e \mod 33 = 29$

The decryption of $c = 29$ is $m = 29^d \mod 33 = 2$

Elliptic curve cryptography (ECC) [73] is an advance to public-key cryptography based on the arithmetical structure of elliptic curves over finite fields. NIST suggested fifteen elliptic curves. Specifically, FIPS 186-3 has ten recommended finite fields:

- Five prime fields $\mathbb{F}_p$ for definite primes $p$ of sizes 192, 224, 256, 384, and 521 bits. For each of the prime fields, one elliptic curve is suggested.

- Five binary fields $\mathbb{F}_{2^m}$ for $m$ equal 163, 233, 283, 409, and 571. For each of the binary fields, one elliptic curve and one Koblitz curve was preferred.

The NIST [52] recommendation thus contains a total of five prime curves and ten binary curves. The curves were apparently selected for optimal security and accomplishment effectiveness.
4.5.3 Algorithm

Step 1: Creation of validation

TTP Registration → Service provider Login

Step 2: Read the input file

\[ r = \{a_1, a_2, a_3 \ldots a_n\} \]

Step 3: Encryption Algorithm

\[ r = \{\text{cipher} (a_1, a_2, a_3 \ldots a_n)\} \text{ using RSA or} \]

\[ r = \{\text{cipher} (a_1, a_2, a_3 \ldots a_n)\} \text{ using ECC} \]

Step 4: Vertical Partitioning Algorithm

\[ r = r_1 + r_2 + r_3 \ldots + r_n \]

Step 5: Decryption Algorithm

\[ r = \{\text{plain} (r_1, r_2, r_2 \ldots r_n)\} \text{ using RSA} \]

Or

\[ r = \{\text{cipher} (a_1, a_2, a_3 \ldots a_n)\} \text{ using ECC} \]

In our proposed system, the user will register his details with the trusted third party (TTP) for secured login in the service provider. After successful registration he has keys through RSA or ECC algorithm for his data storage or retrieval. Through these keys, he has sent the request to the service provider. The service provider verifying the details and ask for the options like storage or retrieval. If the user wants to store the input data is divided into n number of attributes and the attributes will be stored in the different databases by using Vertical Partitioning algorithm. In other case, for retrieval the data from the various databases are integrated through vertical partitioning and decrypted either by RSA or ECC. The Vertical Partitioning algorithm is explained as follows.
Consider a relation $r = \{a_1, a_2, a_3...a_n\}$, which is going give as input data given by the cloud user. The given table has attributes $a_1, a_2, a_3...$ etc. These attributes are divided and perform the vertical Partitioning. For each and every vertical partitioning is done using randomized model. It can be divided into required number of Partitioning. Each splitting is stored in different cloud servers. Before storing it into a cloud server it checks for the already stored data on the servers, if the fields are same (already existing and new one) then the splitted column will be moved to another cloud database. Then the splitted file will be uploaded into the different cloud servers. If a user wants to down the load the file, then he has to get the two types of keys. One is the trusted third party key, and another one is one time password. Then the user has to tell the required fields to the third party auditor. It will pass the field values to each and every cloud server. Then data will be given to the requested server. There will be a stored accountability will be maintained in the trusted third party. And the accountability will be intimated to the owner of the data. Stop the process.

**4.6 RESULTS AND DISCUSSION**

The above mechanism implemented successfully and the results are verified with existing systems. In this mechanism, one input file in the size of 100 kb given to the systems and uploaded by using the RSA and ECC Algorithms. The encrypted data is stored in the different databases by using the Vertical Partitioning Algorithm. While retrieving the uploaded data by these two different algorithms, the execution time is calculated.

Here, the size, time and security level are taken as parameters to conduct the testing. Comparing RSA and ECC algorithm, the ECC algorithm is taking less time for the given input data while upload the file and retrieve the file. Here, 100kb comma separated value file is tested by both algorithms. For retrieving the uploaded file, RSA taking 78 seconds to download the data, whereas ECC is taking 16 seconds.
The prototypes of various working phases are presented along with their screen shots.

Fig. 4.2 CredentialRegistration
Fig. 4.3 Login Window
Fig. 4.4 Home page of the proposed system
Fig. 4.5 User Registration Window
Fig 4.6 Prototype for inclusion of the credential services
Fig 4.7 Encryption of Request
Fig 4.8 Request Testing window after Decryption
Fig 4.9 Request send to the Service Provider
Fig 4.10 Loading of Files
Fig 4.11 Selection of Input File
Fig 4.12 Down loading the encrypted file from Service Provider
Fig 4.13 Data Partitioning using vertical partitioning algorithm
Fig 4.14 Data Base Selection
Fig 4.15 Data Retrieval Form
Fig 4.16 Splitting of Data using Vertical partitioning in Netbean IDE
Fig 4.17 Message Box after File Down Loading

The results are compared by using line chart as given below in figures Fig.4.18, Fig.4.19 and Fig.4.20.

Fig. 4.18 Time Comparison of RSA and ECC while input upload
Fig. 4.19 Comparison of time taken to execute the RSA and ECC algorithm with number of data centers.
Fig 4.20 Comparison of time taken to handle the number of transaction for different algorithms
Fig. 4.21 Reliability of the software between Existing and Proposed
4.7 CONCLUSION

The benefits of the cloud computing are to achieve the economics of scale, reduce the spending on technology infrastructure which is globalize the workforce as very cheap, streamline process, reduces capital cost, improves accessibility and monitoring the projects more effectively. Another focus, as a cloud provider, they have to ensure the security of user’s data. The cloud computing security issues are discussed and new algorithm for protecting the data is developed. The test result based on simulator shows that the new algorithm is used to protect the data more efficiently. The above system tested with various users and collected the feedback about the reliability of the software is compared with the previous one.