International Journal of Cloud Computing and Services Science (IJ-CLOSER)
Vol.3, No.3, June 2014, pp. 144~155
ISSN: 2089-3337

TMR-MCDB: Enhancing Security in a Multi-cloud Model through Improvement of Service Dependability

Mohammed A. AlZain *, Ben Soh, **, Eric Pardede **
** College of Computers and Information Technology,*
Taif University, P.O. Box 888, Al-Hawiya-Taif, 21974, Saudi Arabia
***Department of Computer Science and Computer Engineering,*
La Trobe University, Bundoora 3086, Australia.

Article Info

Article history:

Received Mar 15th, 2014
Revised Apr 25th, 2014
Accepted June 12th, 2014

Keyword:
Cloud Computing,
Multi-Clouds,
Data Security,
Service Dependability,
Data Trustworthiness

ABSTRACT

In IT enterprises, different computing needs are provided as a service. The service providers take care of the customers' needs by, for example, maintaining software or purchasing expensive hardware. In addition, there are many benefits of using the technology available from cloud service providers, such as access to large-scale, on-demand, flexible computing infrastructures. However, increasing the dependability of cloud computing is important in order for its potential to be realized. Data security is one of the most critical aspects in a cloud computing environment due to the sensitivity and importance of the information stored in the cloud, as is the trustworthiness of the cloud service provider. The risk of malicious insiders in the cloud and the failure of cloud services have received intense attention by cloud users.This paper focuses on issues related to service dependability in order to enhance the data security of multi-cloud computing. Service dependability, which encompasses data output trustworthiness, is one of the important factors in enhancing data security in a multi-cloud computing environment. We apply triple modular redundancy (TMR) techniques with the sequential method into our previously proposed Multi-Cloud Database (MCDB) model to improve the data output trustworthiness of our newly proposed TMR-MCDB model. In addition, the improvement in data trustworthiness enhances data security in our TMR-MCDB model. This paper analyzes the impact of data trustworthiness implementation using the voting technique to evaluate the model performance.

Corresponding Author:

Mohammed A. AlZain
Department of Computer Science and Computer Engineering,
La Trobe University, Bundoora 3086, Australia.
Email: maalzain@students.latrobe.edu.au,
College of Computers and Information Technology,
Taif University, P.O. Box 888, Al-Hawiya-Taif, 21974, Saudi Arabia
Email: alzain50@gmail.com

1. INTRODUCTION

As a result of the importance of data security in cloud computing, this paper focuses on issues relating to service dependability and data security in cloud computing. Service dependability encompasses many attributes such as data trustworthiness which will be addressed in this work. Data security raises concerns in relation to service availability, data integrity and data confidentiality. The characteristics of security complement dependability [2].

This work applies the triple modular redundancy (TMR) technique [14] with the sequential method [19] to improve data output trustworthiness of the multi-cloud model. This technique aims to improve service dependability in the cloud computing environment and as a result, the level of data security in the cloud computing environment will be increased.

Currently, there is not much research on using service dependability to enhance the data security of the cloud environment by using the TMR technique with Shamir's secret sharing approach, similar to the approach proposed in this paper. Our contributions can be summarized as follows: in addition to the multi-shares and Shamir's secret sharing approach [18] which have been used previously in our existing MCDB model [7, 9, 8], there is a benefit from adapting the TMR technique and the sequential method to be used in the newly proposed TMR-MCDB model to improve the data trustworthiness of the system. We incorporate service dependability as a new feature into our newly proposed TMR-MCDB model, in that we examine the data trustworthiness of the system, and also analyze how this new feature can enhance data security in our newly proposed model. We provided experiments [10] to evaluate and compare TMR-MCDB model with other cloud cryptographic based model while the evaluation and the comparison of the types of MCDB model discussed in [6].

The remainder of this paper is organized as follows. Section 2 presents the relationship between service dependability and data security in the cloud computing environment. Section 3 overviews Shamir's secret sharing approach and the TMR techniques used in our newly proposed model. Section 4 and 5 proposes the improved TMR-MCDB model, with a thorough explanation of the data flow. Section 6 discusses the analysis and implementation of the newly proposed model. Section 7 concludes the paper.

2. RELATIONSHIP BETWEEN SERVICE DEPENDABILITY AND DATA SECURITY

Avizienis et al. [13] define dependability as "the ability of a system to deliver the required specific services that can justifiably be trusted". Al-Kuwaiti et al. [2] define dependability as "the system property that prevents a system from failing in an unexpected or catastrophic way". Dependability attributes consist of reliability, availability, integrity, safety and maintainability [13]. This paper will not focus on the reliability of the whole system, rather, it focuses on Shamir's data dependability of the system (in this case, data trustworthiness). Shamir's data output trustworthiness has the characteristics of data security attributes. Data security raises concerns about data availability, as well as data integrity and data confidentiality. For example, the trusted outputs of Shamir's data in our TMR-MCDB model are related to data integrity and data confidentiality because it is difficult for a malicious insider to modify Shamir's data in the clouds as the data are hidden by using Shamir's secret sharing algorithm. In addition, Shamir's data are related to service availability because of the replication of Shamir's data into multi-clouds in our TMR-MCDB.

Security is a complement of dependability [2]. Avizienis et al. [13] provide definitions for these attributes. Features such as a high degree of data output trustworthiness, service availability, and protection are the result of a high degree of service dependability which is a very important property in the cloud computing environment. Since cloud computing is large-scale and its applications are accessible anywhere at any time, service dependability in the cloud environment becomes more significant and difficult to achieve [20]. In other words, increasing service dependability of cloud computing is important in order for its potential to be realized [12]. The relationship between service dependability and data security in cloud computing is shown in Figure 1 [13].

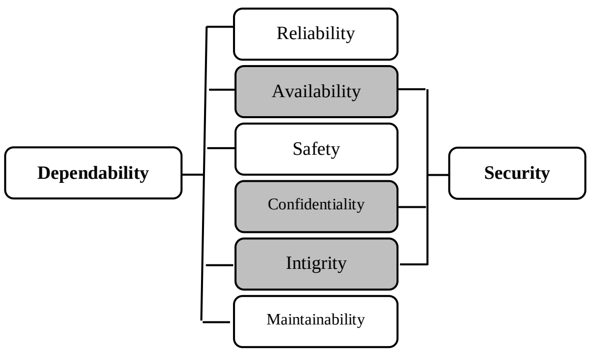


Figure 1.Relationship between Dependability and Security.

As shown in Figure 1, security is a complement of dependability in cloud computing. According to Avizienis et al. [13], security has not been classified as an attribute of dependability. Our work aims to increase the level of Shamir's data output trustworthiness in the proposed model to improve service dependability and thereby enhance security.

3. BACKGROUND

This section explains Shamir’s secret sharing algorithm which will be used in the proposed model. In addition, it covers the technique of redundancy which will be applied in the proposed TMR-MCDB model.

3.1. Secret sharing approach

Agrawal et al. [1] discussed the issue of information distribution with the aim of showing that there is an orthogonal approach which is based on information distribution instead of encryption in the area of data and computer security. The need to communicate important or private information from one party to another motivated most of the work on data security.

Agrawal et al. [1] introduced Shamir’s secret sharing algorithm [18] as a solution to the privacy issue. The algorithm proposed dividing the data D into (n) pieces ($D_1 .... D_n$) in such a way that knowledge of any k or more of D_i pieces makes the value of D known. Therefore, a complete knowledge of $(k – 1)$ pieces reveals no information about D and k should be less than n to keep the value of shares unconstructible and ensure that the adversary cannot access k data pieces. Shamir’s method theoretically secures information.

In addition, by using a (k,n) threshold scheme with $n = 2k – 1$, Agrawal et al. [1] show that a strong key management scheme can be achieved. The goal is to take a distributed approach to secure DaaS, the reason being that they want to explore the use of a secret-sharing approach and multiple service providers. The advantage of this approach is that it addresses both privacy-preserving querying and the data security of outsourced data [7]. We used Shamir’s secret sharing approach in our previous research [5].

3.2. Redundancy Techniques

This section discusses redundancy techniques and also explains TMR techniques with majority voting which is one type of redundancy technique. In addition, it describes the sequential method. The advantage of exploring these techniques in relation to our newly proposed TMR-MCDB model is to improve system dependability which encompasses data output trustworthiness. In our case, we focus on Shamir’s data output trustworthiness.

Johnson [14] defines redundancy techniques as “the addition of information, resources, or time beyond what is needed for normal system operation”. Although the system will have enhanced capabilities by using the redundancy technique, there will be a significant effect on the system in relation to performance, size, power consumption, etc. [14]. This research will provide in depth detail regarding the application of hardware redundancy techniques in our multi-cloud model.

3.2.1. Triple Modular Redundancy

Triple modular redundancy (TMR) is the main type of hardware redundancy which is identified by the triplication of hardware or modules. In this conventional TMR technique, three identical modules execute the same task in parallel. The output of the three identical modules is determined by majority voting inside the voter in the TMR technique. When majority voting of the three modules is undertaken, this means the voter identifies the output from two of the modules or hardware which is fault free. If one of the three models is faulty, the other two models will mask and hide the result of the faulty module and prevent errors being generated Johnson [15]. Figure 2 provides a general overview of the TMR technique [14]. In addition, Lyons and Vanderkulk [16] explain the concept of TMR as originally envisaged by Neumann [21], where the three identical modules in Figure 2 may be computers or less complex units.

The circle labeled ‘voter’ in Figure 2 is called the majority organ by Von Neumann, whereas Lyons and Vanderkulk [16] called it the *voting* circuit because it delivers the *majority opinion* as an output after accepting the input from the three modules. According to Shinohara and Watanabe [19], the output result will be received after majority voting has taken place. The voting circuit is responsible for disregarding the module that receives the single event upset and takes the correct value from the other two modules. The TMR technique with majority voting is a common method that has positive effects on single-event upsets [19]. The purpose of using the TMR technique is to address computer reliability requirements [14, 16, 19]. Our model will not focus on the reliability of the whole system, rather, it focuses on Shamir’s data output trustworthiness (see section 2.3).

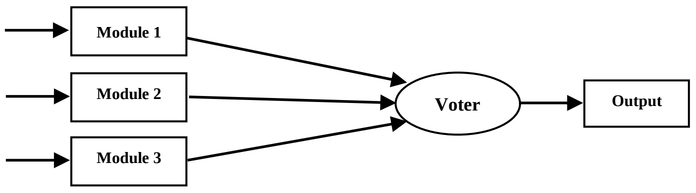


Figure 2. General overview of TMR technique [14].

Figure 2 illustrates the data output trustworthiness of the redundancy system. Since there is an odd number of modules providing input, majority voting is used [19]. The redundancy system will not fail if two of the three modules have not failed, or if none of the three modules fails. In other words, the failure of a single module will not cause the whole system to fail. Hence, the TMR system uses two-of-three majority voting to ensure the data output trustworthiness of the system. The failure of the three modules is independent [19].

3.2.2. Sequential Method

Shinohara and Watanabe [19] discussed the sequential method which is the simplest method and is similar to the TMR technique. In terms of modules, there is no replicating for three modules similar to the TMR method, but a single module is implemented as hardware, and the software execution is executed three times on the hardware. The output results will be taken after majority voting occurs on the three executions on the hardware. The sequential voting method will eliminate the single event upsets in the cycle, so that if there is a single event upset, the system will work perfectly [19].

The number of execution cycles will be decreased by the sequential voting method. For example, if the result of the first and the second cycle is the same, then there will be no need to execute the third cycle because the voting result will not be affected by the third execution result. Therefore, if a single event upset occurs, a triple times execution cycle must be executed. However, Shinohara and Watanabe [19] assumed that if an error occurs on the hardware, the system will not recover and hence will not be able to function properly. Hence, the reliability of the sequential method is not high [19].

4. SYSTEM MODEL

In this section, we overview the new features in the newly proposed TMR-MCDB model. The purpose of the proposed model is to avoid the risk of malicious insiders in the cloud and to prevent the failure of cloud services. As mentioned earlier, the use of TMR techniques improves Shamir’s data output trustworthiness of the system. It is assumed that there is a benefit from adapting the TMR technique and the sequential method to our proposed TMR-MCDB model to improve data output trustworthiness, additional to the previously used Shamir’s secret sharing algorithm and the multi-share technique to improve the level of security in the proposed model. Security risks [4], [11], such as a loss of data integrity, data confidentiality, and service availability will be examined in the model.

TMR-MCDB provides database storage in multi-clouds. The TMR-MCDB model (see Figure 3) does not preserve security by a single cloud; rather, security and privacy of data will be preserved by applying a multi-share technique [7] on multi-clouds. By doing so, it avoids the negative effects of a single cloud, reduces the security risks from malicious insiders in the cloud computing environment, and reduces the negative impact of encryption techniques [3].

TMR-MCDB preserves the security and privacy of the user’s data by replicating data among several clouds, using the secret sharing approach and the TMR technique with the sequential method. It deals with the cloud manager to manage and control the operations between the clients and the multi-clouds inside a super cloud service provider (CSP).

5. SYSTEM DESIGHN

This section will discuss the architecture for the proposed TMR-MCDB model and shows the communication protocol of this model. In addition, it explains how the users can run queries through the model in a secure and private way. Furthermore, it describes how the cloud manager manages the data and divides them into shares and distributes these shares into different clouds (C) inside a super cloud service provider. Voting takes place inside the cloud manager component which is responsible for voting the retrieved result from the clouds before sending it to the user.

5.1. Architecture Overview

The three main components of the TMR-MCDB model are: the cloud manager, the communication protocol, and the cloud-side (see Figure 3). First, the cloud manager is responsible for submitting queries from clients to the clouds and applying Shamir’s secret sharing algorithm on the confidential data. In addition, it is responsible for voting the retrieved result from the clouds before it has been sent to the client. Second, the communication protocol ensures the security and privacy of requests between clients and clouds. Third, the cloud-side is responsible for performing the client’s queries on the converted data before sending responses to the cloud manager.

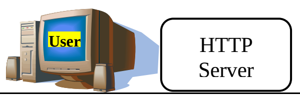
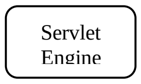
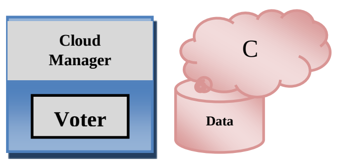
TMR-MCDB contains three layers (Table 1): the presentation layer, the application layer, and the data management layer. The presentation layer contains the end user's browser and HTTP server, the application layer contains the servlet engine and the management layer consists of the cloud manager and the cloud service provider.

5.2. Communication Protocol

As shown in Figure 3, a user sends a query using a user interface and a web browser through an HTTP request. The HTTP server plays a major role in communication between the web browser and the application. The user's query will be sent from the HTTP server to a Servlet Engine by an application request. Hereafter, the communication between the Servlet Engine and the cloud manager is done by a JDBC protocol. When the query arrives at the data source, the cloud manager will manage the query and send it to the C. After the result of the query is returned to the cloud manager, the cloud manager returns the query result to the Servlet Engine and then the HTTP server returns the result of the query to the user interface again. The benefit of the HTTP server is it facilitates communication between the two components: the user browser and the Servlet Engine.

As shown in Figure 3, the process of polynomial functions and voting technique operate within the cloud manager component. Data is sent and stored directly into the clouds and there is no need to store a copy of the user's data in the cloud manager. The proposed TMR-MCDB model will benefit from the voting technique as it will improve data output trustworthiness.

Table 1: TMR-MCDB Layers.

Layer Name	Component
Presentation Layer	
Application Layer	
Management Layer	

5.3. Cloud Manager and Clouds with TMR

As mentioned previously, the TMR-MCDB model (see Figure 3) contains a cloud manager component which is responsible for generating and computing polynomial functions. It is also responsible for conducting majority voting in relation to the retrieved result from the clouds and in doing so, detects the faulty cloud inside the super cloud. The cloud manager component is placed in the client-side and outside the public cloud (within the company network or in a private cloud).

Shamir's secret sharing approach has been extended to suit the proposed TMR-MCDB model. In this section, we describe the data flow from the cloud manager to the multi-clouds inside a super cloud service provider in our proposed model. The cloud manager divides the data into n shares and stores each share in different Cs (see Figure 4). After this, the cloud manager generates a random polynomials function in the same degree for each value of the important attribute that the client wants to hide from the untrusted cloud. The polynomials are not stored at the data source inside the cloud manager but are generated at the front (when the query is received from the user at the cloud manager) and the end of the query processing (when the value is retrieved from C) at the data source. When a user's query arrives at the cloud manager, the cloud manager rewrites n queries, one query for each C and there are n Cs. After this, the relevant share will be retrieved from the C.

For example, the rewritten query for C_1 retrieves all workers whose salary is *share(2000,1)* where the secret value is the salary 2000 and the cloud order is C_1. To find *share(2000,1)*, the data source D first generates polynomials for the secret value salary 2000 and the position for the value in the share $q_{2000}^{(xi).}$ After retrieving the relevant tuple from C, D computes the secret value to send to the client through a secured and private network. The secret sharing method can be applied to execute different types of queries such as exact match, range, and aggregation queries. More details regarding the procedures of these types of queries in TMR-MCDB model can be found in [17]

As previously discussed, the most critical part in the TMR technique is the voter that utilizes the majority voting on the output results of the three modules (in our case, three clouds).

The voter is located inside the cloud manager in the TMR-MCDB model. Regarding the retrieved shares from Cs, the shares of the three clouds will go into the voter inside the cloud manager. If the voter receives the output from C_1 and C_2 after the cloud manager has constructed the polynomial functions of the results, and the results are similar, there is no need to retrieve the result of C_3. In other words, if two of the three C's results are the same, then there is no need to execute the third C, because the voting result will not be affected by the execution of the third cloud result. This is dependent on the combination between the TMR technique and the sequential method as described in section 3.2.1.

There are many benefits from adding a voter in the proposed TMR-MCDB model, for example, error detection in Cs. In addition, it increases data output trustworthiness in the proposed model. Furthermore, the number of execution clouds will be decreased by the sequential voting method.

The final action after majority voting has been applied on the output from the clouds is that the voter will send the voted result to the concerned requester.

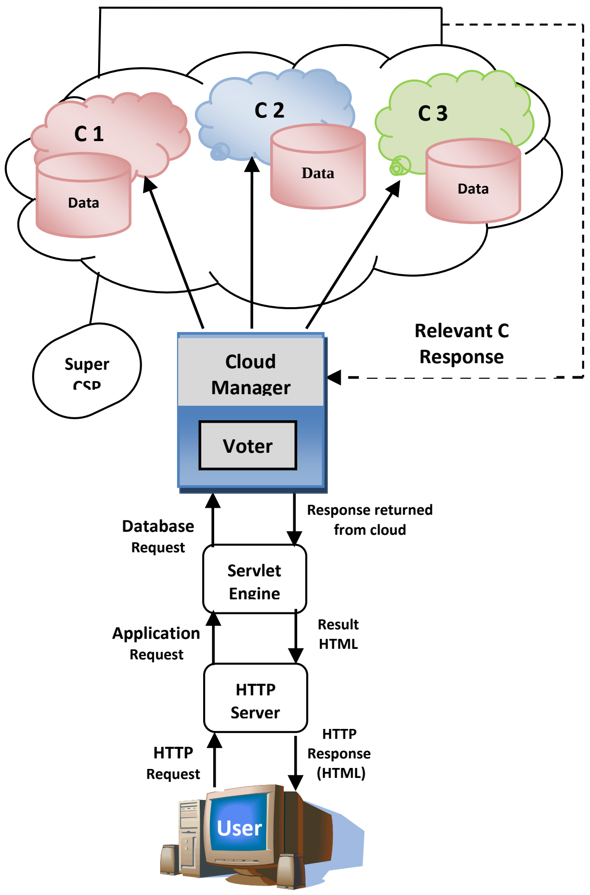


Figure 3.General Architecture of TMR-MCDB Model.

6. Analysis and Evaluation

This section discusses four issues. Firstly, it describes the TMR-MCDB data flow and the architecture of cloud manager components with the voting technique. Secondly, it explains data storing and data retrieval procedures in our newly proposed model. Finally, it analyzes and compares the proposed TMR-MCDB model and the single cloud service model in terms of data integrity, data confidentiality and service availability.

6.1. TMR-MCDB Data Flow

In our proposed model (see Figure 4), the scenario starts by the cloud manager receiving data from a client or from a data provider to be stored in the clouds. The cloud manager divides the received data that the user wants to hide from the untrusted cloud provider into n shares or clusters. After dividing the data (assuming the data is a numeric value, for example, worker's salary) into three shares and storing them in different Cs, the cloud manager generates random polynomial functions with degrees at the same level, one for each C to be stored in different Cs.

For data retrieval, the user's query should arrive at the cloud manager and the cloud manager should rewrite the three queries to retrieve the result from the Cs. Then, the cloud manager computes the secret value of the output result from the Cs to send them to the voter. The voter applies majority voting on the result which has been computed by the cloud manager. After the voting procedure has taken place, the cloud manager will send the results of the vote to the concerned requester.

We provided experiments [10] to evaluate and compare TMR-MCDB model with other cloud cryptographic based model while the evaluation and the comparison of the types of TMR-MCDB model discussed in [6].

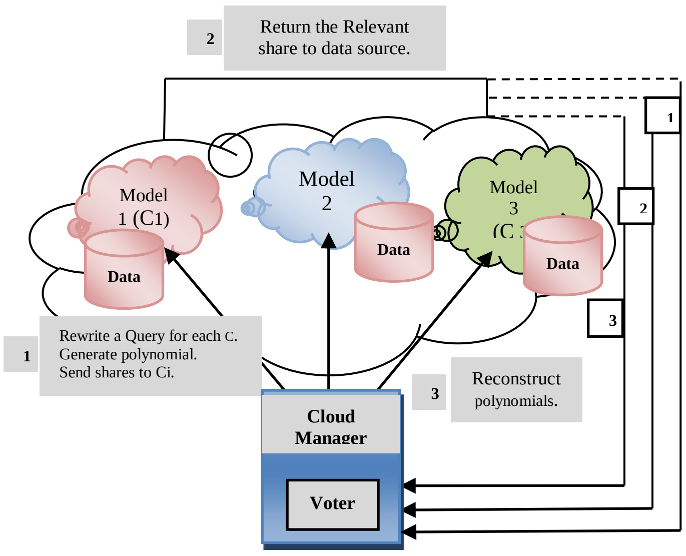

2
Return the Relevant share to data source.
1
2
3
Model 1 (C1)
Data
Model 2
Data
Model 3 (C
Data
1
Rewrite a Query for each C. Generate polynomial. Send shares to Ci.
3
Reconstruct polynomials.
Cloud Manager
Voter

Figure 4. Procedure between Cloud Manager and Cs.

6.2. Architecture of the Cloud Manager with the Voting Technique

This section describes, in more detail, the cloud manager components and the procedures between them. The cloud manager is considered the most critical part in the proposed TMR-MCDB model as it is the control centre of the model. As mentioned before, the cloud manager component is located in the client-side and outside the public cloud (within the company network or in a private cloud). We are aware that the decision to run the cloud manager in a trusted platform and outside the cloud storage platform incurs a communication cost, however it is reasonable to keep the secret keys of Shamir's data and the polynomial functions away from untrusted cloud. The two most significant procedures in the cloud manager are the storing and retrieving procedures. The cloud manager consists of five components: receiver unit, interpreter unit, data source unit, sender unit and voter unit, as shown in Figure 5.

In relation to the storing procedure in the TMR-MCDB model, the database provider sends data to the cloud manager to be received by the receiver unit inside the cloud manager. The receiver unit divides the received data that the user wants to hide from the unsafe clouds into n shares or clusters (in our case, there are three shares because we are dealing with three clouds). After the receiver unit has divided the data (assuming the data is a numeric value, for example, a worker's salary) into three shares to store them in different Cs, the receiver unit sends the shares to the interpreter unit inside the cloud manager. The interpreter unit (see Figure 6) generates random polynomial functions with degrees at the same level, one for each worker's salary in the WORKER table with the actual salary as the constant part of the function. These values will then be stored in different Cs.

For this scenario, the value of n = 3 and k= 2. Note: n=2k-1(see section 3.1). In addition, the interpreter unit uses the secret information *X* values (x_1=3, x_2=1, x_3=2) to create the secret values V_s. The secret information with polynomial functions is stored in array lists inside the data source in the cloud manager.

The sender unit sends the hidden values to be stored in the clouds. Figure 6 illustrates the data distribution procedures between the cloud manager components and the clouds.

Regarding data retrieval, the requester sends a query to the cloud manager as, by this stage, the user's query should have arrived at the receiver unit inside the cloud manager. The receiver unit sends the requester’s query to the interpreter unit to rewrite the three queries, one for each cloud, to retrieve the converted information from the clouds. After the receiver unit has received the results from the clouds, it sends the results to the interpreter unit to re-execute the polynomial functions for each result. After computing the secret values of the results from the Cs, the sender unit sends the three executed results to the voter unit. Consequently, the voter unit applies majority voting on the output results from the sender unit inside the cloud manager. As a result of majority voting, if the voter unit receives the output from C_1 and C_2 after the interpreter unit has constructed the polynomial functions of the output result, and the output results are the same values, then re-execution of the result of C_3 is not required because the voting result will not be affected by the execution of the third cloud output result. As discussed earlier, applying redundancy techniques is beneficial to our multi-clouds model. Finally, the sender unit inside the cloud manager receives the voting results from the voter unit and then sends the results to the concerned requester.

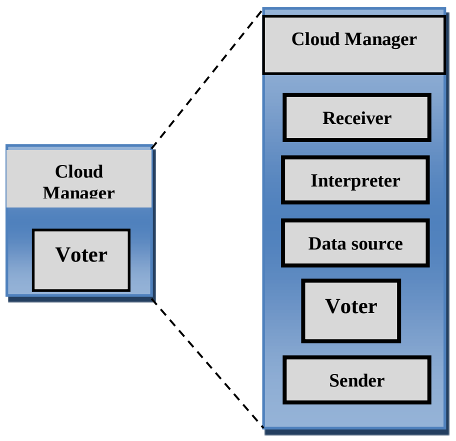


Figure 5. Cloud Manager Components.

There are many benefits of using voting techniques in the proposed TMR-MCDB model, for example, error detection and data recovery in faulty Cs, improved data output trustworthiness in the TMR-MCDB model, and a decreased number of cloud executions as a result of the sequential voting method.

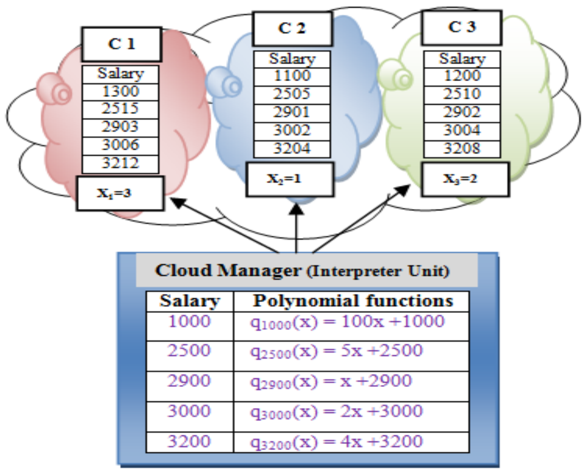


Figure 6. Data Distribution between Cloud Manager and Multi Cs.

The following sections will explain these benefits in detail.

Assumption 1. Error detection and data recovery are one of the benefits of using the voting technique in TMR-MCDB model.

Rationale. Error detection is one of benefits of voting in redundancy techniques. The redundancy of clouds in the TMR-MCDB model and the use of the voting technique have a positive impact on error detection in faulty clouds. The cloud manager can detect a faulty cloud if there are differences between the results of clouds. The faulty cloud will be identified and its result will be disregarded after it has been detected by the voter unit in the cloud manager. In addition, the voter helps to recover data which is determined to be erroneous. For example, if the voter indicates that C_1 is a faulty cloud, the output result of C_2 should not be the same as the output result of C_1, after the execution of the polynomial function of both retrieved shares from both clouds by the interpreter unit. This depends on the procedure of data retrieval in the TMR-MCDB model. In other words, the voter unit simply compares the outputs from the clouds with the first received output result; if the values do not match, an error has occurred at some point during the transmission. An error could occur randomly from one of the three clouds and consequently the presence of the voter unit is vitally important to detect errors in clouds.

There are many benefits of data replication among multi-clouds, since the voter does not have to ask the faulty cloud to retransmit the data, rather the voter can obtain the result from a non-faulty cloud. Thus, the use of the voter unit inside the cloud manager in the proposed TMR-MCDB model is helpful in the area of error detection and to identify the faulty cloud, in addition to data recovery in the faulty cloud.

Assumption 2. Another benefit of using the voting technique in the TMR-MCDB model is the improvement of Shamir's data output trustworthiness.

Rationale. Data trustworthiness is one of the most important aspects of any system. Weakness in data output trustworthiness causes weakness in the system's results which affects the user's queries or causes the system to fail. Component failure may occur in any system if there is no technique to measure or to addresses data trustworthiness in the system. Ensuring a system has a trusted management life cycle and that the system's components perform in the manner in which they have been designed is extremely important. A cloud service provider should ensure that stored data is not modified or changed in any way by an untrusted cloud.The redundancy technique is one of the techniques which help in measuring the data trustworthiness in the system. In other words, if one system component fails, an alternate component can be used and the system can still operate, meaning that data is backed up. Hence, using redundancy techniques together with system failure detection can lead to a high level of system data trustworthiness.

As mentioned previously, the advantage of employing TMR techniques is to improve Shamir's data output trustworthiness in the proposed TMR-MCDB model. Majority voting is considered to be the key factor in TMR techniques because it increases the data trustworthiness of the redundancy system. Since there are an odd number of results coming from the clouds, majority voting is undertaken by the voter unit in the TMR-MCDB model. The redundancy system in the clouds will not fail if two of the three clouds do not fail, or if none of the three clouds fails. In other words, the failure of a single cloud will not damage the operation of the whole system. Therefore, the result from the voter unit is more trusted because it is generated after comparing the results from at least two clouds. Hence, the TMR system uses two-out-of-three majority voting to ensure the data trustworthiness of the system.

Assumption 3. The number of **c**loud executions should be decreased in the TMR-MCDB model as a result of the sequential voting method.

Rationale. It is clear that the use of sequential majority voting techniques have had a positive impact on cloud execution in the TMR-MCDB model. The output results will be taken from a cloud manager after majority voting occurs on the three execution results from the clouds. The sequential voting method will eliminate faulty cloud event upsets in the cycle if there is a single cloud event upset. The number of cloud execution cycles will be reduced by the sequential voting method. For instance, if the output of the first cloud and the second cloud cycles are similar, it is unnecessary to execute the third cloud execution cycle because the voting result will not be affected by the third cloud execution cycle. Simply, if any of the clouds fail, then a triple cloud execution cycle must be applied. In other words, as shown in Figure 7, in cycle 1, the voter unit will execute cloud 1 whereas cloud 2 will be executed in cycle 2 by the voter unit in the TMR-MCDB model. If the output of the executed results is the same, then it is unnecessary to execute cycle 3. However, if the results of the two clouds are different, the third cycle should be executed. Thus, the faulty cloud will be identified. Although the time cost is increased with the increased number of cycles in the TMR-MCDB model, majority voting techniques may reduce the execution cycle of clouds which decreases the time cost of cloud executions.

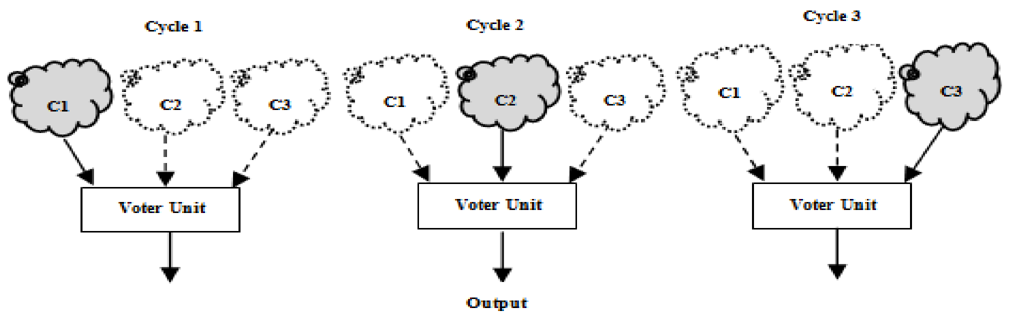


Figure 7.Cloud Life Cycle execution, adopted from [19].

6.3. Data Storing Procedure

The data storing procedure in the TMR-MCDB model involves data distribution from the data provider to multi-clouds inside a super CSP. This is done after executing the polynomial functions on the data. Figure 8 shows the TMR-MCDB-Store algorithm (*Algorithm 1*). The key idea of the TMR-MCDB-Store algorithm is to first (line 4) receive the data from the client or data provider to be stored in the clouds. Subsequently, the receiver unit inside the cloud manager divides the data into *n* shares (line 5). Then, (line 7) the interpreter unit inside the cloud manager generates random polynomial functions with the degree at the same level, one for each C to be stored in different Cs. The following acronyms will be used in the algorithms: Data provider (DP), Receiver Unit (RU), Interpreter Unit (IU), Sender Unit (SU), Data storage (DS), Voter Unit (VU), User (U), Data (D), and Cloud (C).

6.4. Data Retrieval Procedure

The data retrieval process in the TMR-MCDB model starts with rewriting the received query from the user in the cloud manager (*n* number of queries) and then sends these queries, one for each C, after constructing the polynomial functions and the order for the secret values. The shares will be returned to the cloud manager in parallel to compute the polynomial function on the returned values. Consequently, the sender unit sends the three executed results to the voter unit in parallel. As a result, the voter unit applies majority voting on the output from the sender unit inside the cloud manager to determine whether the result is trusted and if so, sends this to the user in a secure way.

Figure 9 shows the TMR-MCDB-Retrieve algorithm (*Algorithm 2*). The purpose of the TMR-MCDB-Retrieve algorithm is to first (line 4) receive from the client the query to be sent to the clouds through the receiver unit in the cloud manager. Consequently, the interpreter unit inside the cloud manager rewrites the three queries (line 5) to retrieve the result from the Cs. After this, (line 7) the interpreter unit inside the cloud manager computes the secret value of the output from the Cs to send them to the voter. Then, the voter unit applies majority voting (line 12) on the computed result before it is sent to the user.

Algorithm 1: TMR-MCDB-Store

```
1   **Procedure** TMR-MCDBStore($D$)
2   **Begin**
3   DP sends $D$
4   RU ←$D$
5   RU divide  $D$ into ($n$) shares ($D_1$....$D_n$)
6             Where   $n = 2k – 1$ AND $k<n$
7   IU Generates $q_{(x)}$ for each  $v_s$ Where
8       $\begin{bmatrix} Shares(v_s,1) = q(x_1) = ax_1^{k-1} + bx_1^{k-1} ... + v_s \\ ... \\ Shares(v_s,n) = q(x_n) = ax_n^{k-1} + bx_n^{k-1} ... + v_s \end{bmatrix}$
9   DS=[$x_1$=3, $x_2$=1, $x_3$=2, ...]
10  SU Sends ($D_1$....$D_n$)  into Cs
11  Cs.DS ←($D_1$....$D_n$)
12  **END**
```

Figure 8.TMR-MCDB-Store Algorithm.

Algorithm 2: TMR-MCDB-Retrieve

```
1   **Function** TMR-MCDBRetrieve(*query*)
2   **Begin**
3   U sends *query*
4   RU ←*query*
5   IU write ($n$) *query* for each C
6   Cs ←*queries*
7   Cs sends *results*
8   RU ←*results*
9   IU Re-execute $q_{(x)}$ for each $V_s$
10  SU sends executed $V_s$ to VU
11  VU ←$V_s$
12  VU majority voting  results
13  SU ←$V_s$
14  U ←$V_s$
15  **END**
```

Figure 9.TMR-MCDB- Retrieve Algorithm

7. CONCLUSION

It is clear that although the use of cloud computing has increased rapidly, cloud computing security is a major issue in the cloud computing environment. Customers do not want to lose their private information as a result of malicious insiders in the cloud. In addition, data integrity and data confidentiality leads to many problems for the users of cloud computing. Furthermore, recently, the loss of service availability has caused many problems for a large number of customers. This paper focuses on the issues relating to service dependability and data security aspects in cloud computing. Service dependability is one of the most important factors in enhancing data security in the cloud computing environment. The purpose of this work is to propose a new model called TMR-MCDB which uses Shamir’s secret sharing algorithm with multi-clouds instead of a single cloud. In addition, the TMR-MCDB model adopted TMR techniques with a sequential method to improve Shamir’s data output trustworthiness of our model which enhances service dependability. This paper discussed the architecture of the model, as well as the model’s components and layers.

The aim of the proposed model is to reduce data security risks that occur in cloud computing and improve Shamir’s data output trustworthiness. At this stage, we have compared our proposed multi-clouds model with the Amazon cloud service as a single cloud model. As a result of this comparison, it has been shown that the multi-cloud model is superior to the single cloud model in addressing security issues in cloud computing. For future work, we plan to compare our model with other multi-cloud models, and propose an improved model. In addition, further analysis of service dependability and security in the context of the TMR-MCDB model will be undertaken. TMR-MCDB will be deployed and systematically tested in the private cloud computing environment to prove the finding on a real world application.

REFERENCES

[1] D. Agrawal, A. El Abbadi, F. Emekci and A. Metwally, *Database Management as a Service: Challenges and Opportunities, Proceedings of The 2009 25th International Conference on Data Engineering*, IEEE 2009, pp. 1709-1716.

[2] M. Al-Kuwaiti, N. Kyriakopoulos and S. Hussein, *A comparative analysis of network dependability, fault-tolerance, reliability, security, and survivability*, Communications Surveys & Tutorials, IEEE, 11 (2009), pp. 106-124.

[3] M. A. AlZain and E. Pardede, *Using Multi Shares for Ensuring Privacy in Database-as-a-Service, Proceedings of The 2011 44th Hawaii International Conference on System Sciences (HICSS)*, IEEE, Kauai, USA, 2011, pp. 1-9.

[4] M. A. AlZain, E. Pardede, B. Soh and J. A. Thom, *Cloud Computing Security: From Single to Multi-clouds, Proceedings of The 2012 45th Hawaii International Conference on System Science (HICSS)*, IEEE, Maui, USA, 2012, pp. 5490-5499.

[5] M. A. AlZain, B. Soh and E. Pardede, *A Byzantine Fault Tolerance Model for a Multi-cloud Computing, Proceeding of The 2013 16th International Conference on Computational Science and Engineering CSE*, IEEE, Sydney, Australia, 2013, pp. 130-137.

[6] M. A. AlZain, B. Soh and E. Pardede, *Evaluation of Multi-Cloud Computing TMR-Based Model Using a Cloud Simulator, Proceedings of The 2014 11th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2014)*, Co-sponsered by IEEE, China, Aug-2014.

[7] M. A. AlZain, B. Soh and E. Pardede, *MCDB: Using Multi-clouds to Ensure Security in Cloud Computing, Proceedings of The 2011 Ninth International Conference on Dependable, Autonomic and Secure Computing (DASC)*, IEEE, Sydney, Australia, 2011, pp. 784-791.

[8] M. A. AlZain, B. Soh and E. Pardede, *A New Approach Using Redundancy Technique to Improve Security in Cloud Computing, Proceedings of The 2012 International Conference on Cyber Security, Cyber Warfare and Digital Forensic (CyberSec12)*, IEEE, Kuala Lumpur, Malaysia, 2012, pp. 230-235.

[9] M. A. AlZain, B. Soh and E. Pardede, *A new model to ensure security in cloud computing services*, Journal of Service Science Research, 4 (2012), pp. 49-70.

[10] M. A. AlZain, B. Soh and E. Pardede, *Performance Overhead Evaluation of Multi-cloud Computing with Secret Sharing approach Based Model,*, International Journal of Cloud Computing and Services Science (IJ-CLOSER), 3(2) (2014).

[11] M. A. AlZain, B. Soh and E. Pardede, *A Survey on Data Security Issues in Cloud Computing: From Single to Multi-Clouds*, Journal of Software, 8 (2013), pp. 1068-1078.

[12] J. Arshad, P. Townend and J. Xu, *Quantification of security for compute intensive workloads in clouds, Proceedings of The 2009 15th International Conference on Parallel and Distributed Systems (ICPADS)*, IEEE, 2009, pp. 479-486.

[13] A. Avizienis, J. C. Laprie, B. Randell and C. Landwehr, *Basic concepts and taxonomy of dependable and secure computing*, IEEE Transactions on Dependable and Secure Computing, 1 (2004), pp. 11-33.

[14] B. W. Johnson, *Design & analysis of fault tolerant digital systems*, Addison-Wesley Longman Publishing Co., Inc., 1988.

[15] B. W. Johnson, *An introduction to the design and analysis of fault-tolerant systems*, Upper Saddle River, New Jersey: Prentice Hall, 1995, pp. 1-84.

[16] R. Lyons and W. Vanderkulk, *The use of triple-modular redundancy to improve computer reliability*, IBM Journal of Research and Development, 6 (1962), pp. 200-209.
[17] M. A. AlZain, B. Soh and E. Pardede, *PPQ Privacy Preserving Query Service over Shamir's Data in a Multi-Cloud Computing Environment* Journal of Parallel and Distributed Computing, (in review) (2013).
[18] A. Shamir, *How to share a secret*, Commun. ACM, 22 (1979), pp. 612-613.
[19] K. Shinohara and M. Watanabe, *A double or triple module redundancy model exploiting dynamic reconfigurations*, *The 2008 IEEE Conference on Adaptive Hardware and Systems, NASA/ESA*, IEEE, 2008, pp. 114-121.
[20] D. Sun, G. Chang, Q. Guo, C. Wang and X. Wang, *A Dependability Model to Enhance Security of Cloud Environment Using System-Level Virtualization Techniques*, *Proceeding of The 2010 First International Conference on Pervasive Computing Signal Processing and Applications (PCSPA)*, IEEE, 2010, pp. 305-310.
[21] J. Von Neumann, *Probabilistic logics and the synthesis of reliable organisms from unreliable components*, Automata studies, 34 (1956), pp. 43-98.

BIOGRAPHY OF AUTHORS

Mohammed A. AlZain is a PhD candidate in the Department of Computer Science and Computer Engineering at La Trobe University, Melbourne, Australia since Oct-2010. Currently, his PhD research is in Cloud Computing Security under Assoc/Prof. Ben Soh and Dr. Eric Pardede. He has achieved his Bachelor degree in Computer Science from King Abdulaziz University, Saudi Arabia in 2004, and then achieved his Master's degree in Information Technology from La Trobe University in 2010. He is a lecturer in the faculty of Computer Science and Information Technology at Taif University in Saudi Arabia. His area of interest: Cloud Computing security, Database As Services. Mohammed is an IEEE student member.

Ben Soh is an Associate Professor in the Department of Computer Science and Computer Engineering at La Trobe University, Melbourne, Australia and a Senior Member of IEEE. He in 1995 obtained his PhD in Computer Science and Engineering at La Trobe. Since then, he has had numerous successful PhD graduates and published more than 150 peer-reviewed research papers. He has made significant contributions in various research areas, including fault-tolerant and secure computing, and web services.

Eric Pardede received the Master of Information Technology degree and Ph.D. degree in computer science from La Trobe University, Melbourne, Australia, in 2002 and 2006, respectively. He is currently a Lecturer with the Department of Computer Science and Computer Engineering, La Trobe University. He has wide range of teaching and research experience including in the area of databases, software engineering, information systems, entrepreneurship, and professional communication.