ISSN: 2633-4828 Vol. 5 No.4, December, 2023

International Journal of Applied Engineering & Technology

D1FTBC FRAMEWORK FOR CLOUD DATA TRANSACTIONS

Dr. J. Antony John Prabu[1] and Dr. V. A. Jane[2*]
[1]Assistant Professors, Department of Computer Science, St. Joseph's College (Autonomous), Affiliated to Bharathidasan University Tiruchirappalli
[2]Assistant Professors, Department of B. Voc (SD & SA), St. Joseph's College (Autonomous), Affiliated to Bharathidasan University Tiruchirappalli
[1]ajp.trichy@gmail.com and [2]withdifferentjane07@gmail.com

ABSTRACT

Transactional database systems in cloud environments involve individuals and organizations granting permissions to the cloud server. Information provided by end-users during a transaction, validated by a transaction manager for permission, serves as evidence. The flexibility of the cloud introduces a less stringent integrity model. In distributed settings using transaction databases, consistency issues may arise during transactions. System updates may encounter mismatches in policies, and external factors can disrupt user credentials during exchanges. To address these challenges in cloud transactions, we propose the Depth 1 Fixed Tree Based Consistency (D1FTBC) framework. This transactional approach ensures rule adherence without causing identity inconsistencies. The term "safe transactions" is introduced, encompassing trusted transactions adhering to the basic rules of distributed storage systems. This study presents a method to enhance the safety and reliability of activities in the cloud environment.

Index Terms: Cloud, data transactions, DIFTBC framework

INTRODUCTION

The internet has witnessed an unprecedented surge in users, escalating the demand for data. However, the accessibility of data at any time and from any location has become impractical due to the high cost and administrative complexity of existing computer infrastructures [1]. Cloud Computing (CC) addresses this challenge by offering on-demand access to networked, customizable computing resources, simplifying resource supply and release for management and service providers [2].

CC encompasses the sharing of data, connections, processing power, and programs across computers [3]. Its applications have proliferated across diverse sectors such as research, manufacturing, education, consumption, and entertainment [4]. With the increasing adoption of cloud computing (CC) for data storage, users and organizations are relieved of the burden of maintaining basic storage infrastructures, leading to a substantial growth in the volume of data stored on cloud servers [5].

The key benefits of CC lie in its cost-effectiveness and user-friendly nature. Users can leverage pay-as-you-go services, paying only for the utilized computer resources. The cloud's massive capacity accommodates extensive user use, involving three main players: the user, the cloud service provider (CSP), and the data owner (DO). CC plays a pivotal role in the evolution of business IT systems, allowing tech giants like Google, Amazon, IBM, and Microsoft to serve their customers on a vast scale. Target users of CC include organizations in military, healthcare, service computing, and other sectors relying on mega or high-performance computing for efficient data processing and distribution.

The common cloud deployment options are public, private, and hybrid clouds, with community clouds being a less popular concept. Public clouds share resources among all registered users of a cloud service provider's platform, offering users the flexibility to store and exchange data. Despite the benefits, cloud computing introduces security challenges, necessitating the protection of users' personal information and secure transfer of sensitive data [6]. Service providers bear a significant responsibility to ensure the security of their storage systems, addressing data loss or theft concerns [7].

In response to the security challenges, this work proposes a novel Depth 1 Fixed Tree-Based Consistency (D1FTBC) framework. The subsequent sections of the paper are organized as follows: Section II provides a clear definition of related works, followed by Section III detailing the proposed methodology. Section IV elaborates on the results and discussions, and finally, Section V concludes the paper.

RELATED WORKS

In the future, numerous enhancements can be implemented to this concept in accordance with the relaxed consistency model of the cloud environment. A primary consideration in the dependable outsourcing of technologies to the cloud is the security of its services and its users. A critical procedure by which cloud service providers preserve duplicates of the services they offer their customers is known as data replication. In other relevant literature, the integration of data replication with proofs of retrievability has been proposed as a means to ensure users are provided with consistent policies, credentials, and data integrity. In order to safeguard user access patterns from a cloud data repository, [8] proposes a mechanism that enables untrusted service providers to facilitate transaction serialisation, backup, and recovery while ensuring complete data correctness and confidentiality for encrypted reads, writes, and inserts from cloud storage. The focus of this study is not on the issues of consistency that arise in policy-based database transactions. An assault detection model that utilises certified properties in a multi-tenant cloud configuration is detailed in Paper [9]. Recent research has also concentrated on establishing a degree of assurance regarding the correlation between policies and data [10]. It ensures that the server-side policies accurately correspond to the data that is already stored on the server. In accordance with the revised policy, the server will evaluate the information and remove any malevolent content whenever the policy is modified. Probably, research pertaining to the security of virtual cloud computing can be divided into two distinct categories [11]. It entails the procedure for assessing the security of cloud storage and cloud computation. The authors of [12] have put forth a protocol known as the Orthogonal Handshaking Authentication Protocol with the purpose of facilitating transactions in the cloud. The algorithm proposed in this paper is an orthogonal encryption method for server message storage.

PROPOSED WORK

Utilizing the proposed D1FTBC framework, the transaction is carried out in a cloud environment. It is designed to decrease execution time and increase consistency among cloud-based replica servers. The initial segment of the framework performs query analysis, followed by transaction tree adjustment in the second segment, transaction execution in the third, and data update in the fourth. This chapter describes in detail the proposed Depth 1 Fixed Tree Based Consistency framework's preliminary configuration and functionality, as well as its numerous

The elements. Additionally, the execution state of the proposed three-phase secure tree-based commit (3PSTBC) protocol is delineated in order to attain an elevated level of security.

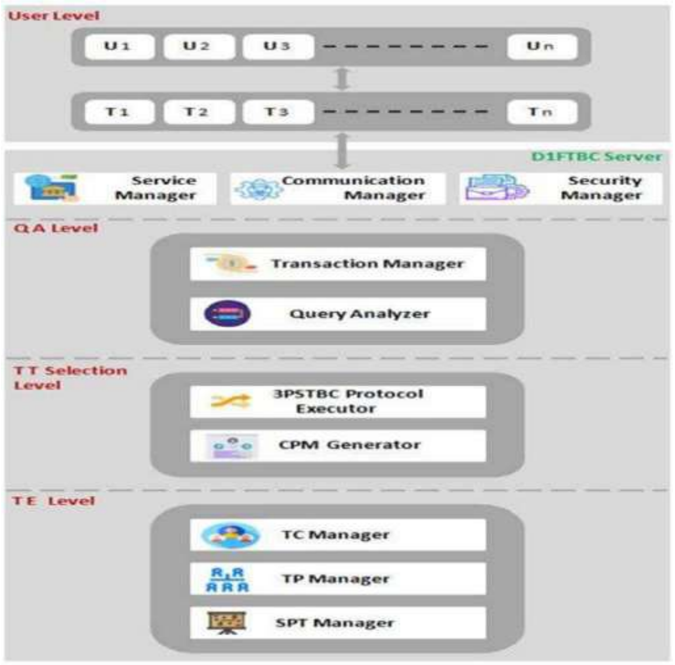

Figure 1: Schematic representation of the suggested methodology

Figure 1 laid out the various parts of the D1FTBC server, which work together to improve the consistency state of cloud data transactions. These parts include the following: Query Analyzer (QA), 3PSTBC Protocol Executor (3PSTBC), CPM Generator (CPMG), Transaction Tree (TT), Transaction Coordinator Manager (TCM), Transaction Participant Manager (TPM), and Shortest Path Tree Manager (SPTM).

User

To send in their submissions, users may talk to the communication manager, the request for the transaction and get a reply while keeping all the data.

Communication Manager (CM)

For transaction execution, the communication manager (CM) connects the Cloud Data Locker server (CDL) and the Cloud Services Controller (CSC). It then accesses the Security Manager (SEM), the Service Manager (SM), and the Transaction Manager (TM).

Security Manager (SEM)

In order to monitor, evaluate, and manage cloud data transactions, the Service Manager (SM) verifies the specifics of the service verification log provided by the Cloud Service Controller (CSC).

Transaction Manager(TM)

To guarantee the safety of every transaction, the security manager keeps track of the log information from the CSC server and the CDL server.

All cloud-based transactions must be overseen by the transaction manager (TM). In order to determine whether a transaction is an update or search, it manages the Query Analyzer (QA). whether the transaction is an upgrade, it will assign the participants and coordinator the fixed tree.

Query Analyzer (QA)

In order to determine whether a query is a read or write query, the Query Analyzer (QA) looks at the submission history. The 3PSTBC protocol is sent to execute in the case of a write query; otherwise, any node may be accessed.

3PSTBC Protocol Executer
The work delves into the features and characteristics of a three-phase secured tree-based commit procedure.

CPM Generator (CPMG)
Based on the valuation metric formula, the CPM Generator keeps track of the CPM value for every depth 1 tree. A different tree is chosen for every transaction based on the CPM value. The driving of tree transactions is more important for low CPM values.

Consistency Performance Metric (CPM)
Each depth 1 tree's CPM value is assessed using the valuation metric formula. For every transaction, a static tree is chosen based on the CPM value. The driving of tree transactions is more important for low CPM values.

Transaction Coordinator Manager (TCM)
The TC logs, which record each transaction, are maintained by the transaction coordinator manager. In it, you'll find information on the transactions that TC processes.

Transaction Coordinator (TC)
In order for a transaction to commit, the coordinator is in charge of keeping track of everyone involved in a predetermined tree. Those processes for committing are run by the transaction coordinator.

Transaction Participant Manager (TPM)
Each transaction's transaction participant (TP) log is maintained by the transaction participant manager. The specifics of the deal, including the TP involved, are included in it.

Transaction Participants
A limited number of processes are used to classify the transaction and communicate data to the participants in a fixed tree. While a transaction is underway, the Transaction Coordinator keeps tabs on everyone involved.

Fixed Tree/ Transaction Tree (TT)
These possible fixed-tree topologies are solid for implementing the 3PSTBC protocol, which uses a single coordinator for all transactions and a large number of participants to update them. Individuals taking part in the activity should maintain their distance from one another and speak only with the facilitator. Based on the CPM value, the transaction coordinator may be chosen by the transaction manager, and the linked nodes can be chosen as participants in the transaction.

Shortest Path Tree Manager (SPTM)
So that data replication may take place, SPTM establishes the shortest route between each node. Following the completion of each transaction, the SPTM modifies the trustworthy SPT to generate the necessary data for committing the transaction.

Shortest path Trees
Every transaction in the cloud duplicates data across a wide geographic area, with the possibility of data loss throughout the replication process. Finding a shortcut requires combining all virtual machines. This ought to prevent using the wrong database and shorten the cloud environment's reflection time.

Preliminary setup
The first step in the D1FTBC technique is to create transaction trees within the virtual machines that are participating. It guarantees virtual machine scalability for cloud data transfers. Virtual computers, or nodes, may be added or removed based on the volume of work resulting from transaction requests.In the following phases, the D1FTBC initial configuration consists of portraits.

Step 1: For every accessible node, create a joint graph.

Step 2: Generate Transaction Trees or Potential Depth 1 Fixed Trees. With a linked graph, the Adjacency List is used to identify potential depth 1 fixed trees that might act as cloud replica servers.

Step 3: Calculate the separation between every node. The adjacency matrix is used to ascertain the total distance between each node in the tree as well as the number of edges for every tree.

Step 4: Determine the quickest route. The Dijkstra method is used to determine the shortest route from the given edge to all other frequency nodes. The shortest route tree is used by the transaction manager to update the data after the completion of each transaction.

Consistency Performance Metric (CPM)

The measure used to evaluate the performance of all available standard trees and nodes is the consistency performance parameter. The optimum sequence for sustainable trees, as determined by the CPM value, aids in speeding up data transaction processing.

The following are CPM performance factors:

Table: 1. Performance factor of CPM

“N	Number of Nodes
D	Total Distance between Nodes
PR	Path Reliability
TD	Time Delay
E	Total Time of Execution
CPM	Consistency Performance Metric”

3PSTBC Protocol

Among the most effective methods for implementing data transactions in a distributed system is the 3-Phase Secured Tree Based Commit (3PSTBC) protocol. It will be useful for conducting transactions with ACID guarantees in a cloud setting. The suggested D1FTBC structure benefits from its increased reliability. In the first of the three stages that make up the 3PSTBC Protocol, the Consistency Performance Metric Manager (CPM) is used to distribute the Transaction Tree (TT). In the second step, the TT's Transaction Coordinator (TC) and Transaction Participants (TP) verify each other's transactions to carry out the transaction. The last stage involves updating or replicating the Cloud data by analysing and completing the Shortest Path Tree (SPT) with the aid of the Short Path Tree Manager (SPTM).

Phase 1

A request to generate a Transaction Performance Metric (CPM) is required for the preparation of a Transaction Tree (TT) by the Transaction Manager (TM). A notification is produced and delivered to TM once the CPM manager fixes the Transaction Tree. If the request does not proceed to the second phase after receiving the prepared message from CPM, the transaction is terminated.

Phase 2

After waiting for all Transaction Participants (TP) engaged in the Transaction Tree (TT) to get the "request-to-prepare" message, the Transaction Coordinator (TC) may cast their vote. If a TP is prepared to commit, it may vote with a Prepared message; otherwise, it can send a No message; or it can postpone voting forever.

After receiving a "Prepared message" from each TP, the request moves on to the third phase if the TC determines that everything is ready to commit. In all other cases, the abort state is reached.

Phase 3

Request a "prepare" message from the Shortest Path Tree Manager (SPTM) if the Transaction Coordinator (TC) has received "ready to commit" from all participating Transaction Participants (TP). After that, in order to commit the transaction, the SPTM's Shortest Path Tree (SPT) updates or replicates the data and sends a "Prepared" message to the TC. In such case, the deal will not go through. By responding with a status code like "Done," TPs let the TC know whether they want to commit or cancel the transaction. The choice to commit or abort is thrown to all TP by the Transaction Coordinator.

Pseudocode for D1FTBC Approach

```
"Algorithm for Depth 1 Fixed Tree Consistency (D1FTBC) Framework: Step 1 : User sends request to transaction manager (TM)
Step 2 : TM access Query Analyzer (QA)
Step 2.1 : If (Read operation)
{
Access any node and get updated data.
}
Else
Step 2.2 : Refer 3PSTBC Protocol Executer (3PE) Step 3 : Refer CPM Generator to select Transaction Tree
Step 4: Select the Fixed Tree [According to the transaction querry,]
Step 4.1: Fix the root node as Transaction Coordinator(TC) and Access Transaction Coordinator through TC Manager(TCM)
Step 4.2: Fix the child nodes are Transaction Participants (TP) through TP Manager (TPM)
Step 5: Implement 3PSTBC protocol in the TC.
Step 5.1: TC divides the transaction and sends it to Transaction Participants (TP) and executes the transaction
Step 6 : Refer Shortest Path Tree Manager (SPTM )
Step 7 : Select shortest path tree and Replicate/update data.
 Step 8 : Successfully commit the transaction"
```

PERFORMANCE ANALYSIS

When it comes to improving the efficiency of cloud data transfers, consistency measures are crucial. In these chapters, we take a look at how various consistency approaches—like the Classic, Quorum, TBC, and D1FTBC approaches—are put into action.

Execution of Classic Approach

Write and read transactions are executed using the conventional manner, as shown in Figure 2. Presuming the five copies are distributed across six distinct nodes in the network. Despite having several secondary nodes, it only has one main node. The main node is the appropriate place to implement the lock management. Before a lock manager execution can begin, all write transactions must meet the main node. An execution in the traditional method would wait for acknowledgement from every replica node. In a write transaction, all nodes are asked to acknowledge the request.

One node is all that's needed to return data for a read transaction.

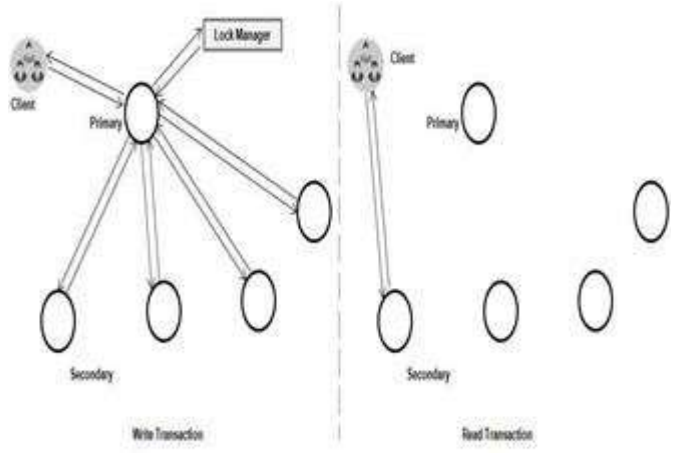

Figure 2: Execution of Classic Approach

Therefore, the old method is faster for numerous reads but slower for few writes. Any replica node may return the value for read requests, which is a benefit of this strategy. However, the execution wait for acknowledge from all replica nodes.

Execution of Quorum Approach

The procedure for executing write and read transactions using the quorum technique is shown in Figure 3. Presuming the five copies are distributed across six distinct nodes in the network. Nodes in the system may operate as either a coordinator or a participant in a transaction. The Transaction Coordinator node is a suitable location to implement the lock manager.

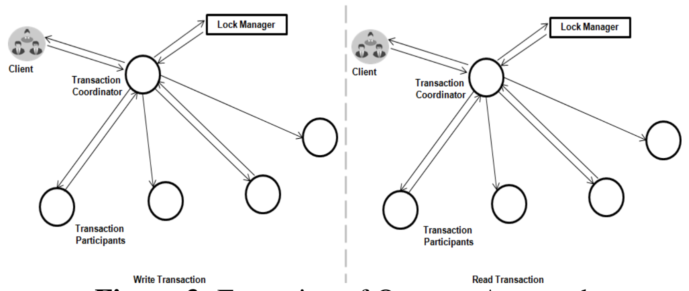

Figure 3: Execution of Quorum Approach

Execution of TBC Approach

The majority of nodes must agree to conduct a read/write transaction for the quorum strategy to work. To ensure uniformity, a voting system based on a simple majority is used. Any server may take the role of coordinator, with other servers playing the role of participants, since any node can function as either a main or secondary replica. No particular main or secondary copies are available for purchase. All the participants do is comply with the demands made by the coordinator. The response time for numerous writes and few read requests is improved. The benefit of this method is that any replica node may take on the role of a transaction coordinator, with the other nodes just participating in the transaction. But in a Read Transaction, the coordinator doesn't act until there's a unanimous response from most of the participants. Read activities result in a poor reaction time.

In Figure 4, we can see how the TBC method for write and read transactions is executed. Here we're assuming that the six copies are distributed across six separate nodes in the network. It builds a transaction consistency tree. There is a hierarchical structure with one parent and several children and subchildren.

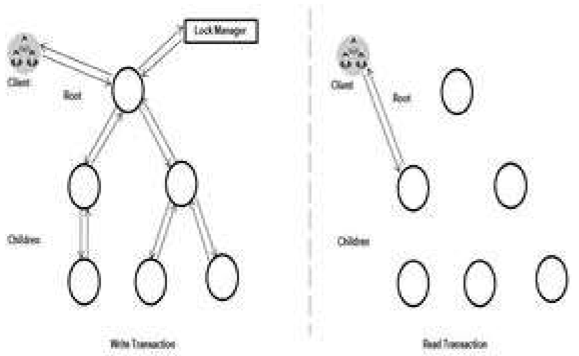

Figure 4: Execution of TBC Approach

Data updates and communication between nodes are defined by the created tree. A replica node in a tree can only connect with nodes in its subset when update requests come. Afterwards, it can read data and write data. While one replica keeps the data up-to-date, the two replicas at the very top of the consistency tree process read/write requests. Both the density of the consistency tree and the number of children in the tree determine the reaction time. The read operation reaction time is quicker, while the write operation response time could be slower when dealing with thick trees. Defines a way to communicate nodes across consistency tree, which is a benefit of this technique. Choosing the root node does not end the process of creating a consistency tree, however. To get a short reaction time for a thick tree, it could take longer time from conception to lead.

Execution of D1FTBC Approach
In Figure 5, we can see how the D1FTBC method for write and read transactions is executed. Presuming the five copies are distributed across six distinct nodes in the network.

When it comes to write transactions, the D1FTBC method has two states: execution and commit. In the initial configuration, the network's linked nodes are arranged in a depth 1 fixed tree. In order to choose which transaction tree to prioritise for execution, the Consistency Performance Metric (CPM) is used. The CPM formula determines the value of the number of nodes and the distance between them in each transaction tree. Among the various Depth 1 Fixed Trees, CPM is used to fix the Transaction Tree. The root node takes on the role of transaction coordinator when the transaction tree is chosen, while the nodes below it in the depth 1 level take on the role of transaction participants. The suggested 3PSTBC protocol works well with this Depth 1 fixed tree layout. It is possible to use several transaction trees to process many transactions in parallel. Following the last countdown confirm that the shortest route tree was used to reproduce the data. When doing an update or replica, the Shortest Path Tree Manager (SPTM) is the tool of choice. As a result, it has a long lifespan.

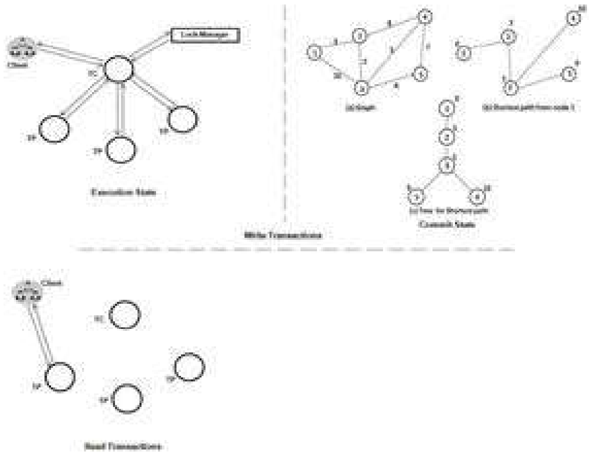

Figure 5: Execution of D1FTBC Approach

For a read request, the value may be returned by a single node. It is free to choose any branch in the transaction tree. In this method, there aren't any designated main or secondary copies. Throughout the whole transaction process, the participants do nothing more than react to the coordinator's commands. Requests for reads, writes, and combined transactions are processed more quickly. Because of this, the D1FTBC method outperforms its competitors in terms of consistency.

Success Rate of D1FTBC Approach

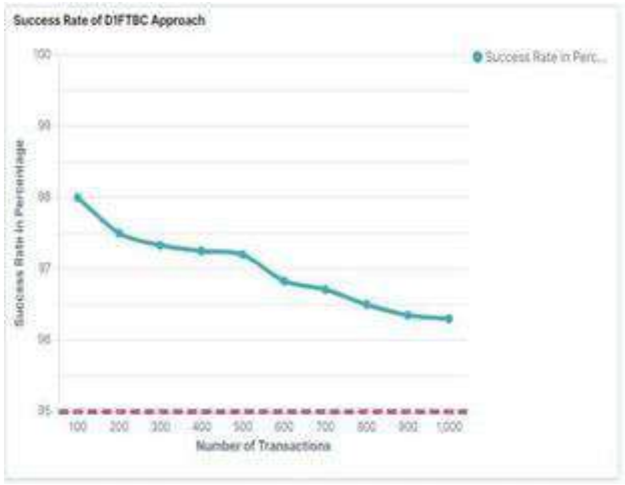

Figure 6: Performance Analysis of Success Rate for Increasing Service Request

The relationship between the success rate and the arrival rate is seen in Figure 6. The connection between the amount of transactions and the percentage of success is shown by the plot. The system seems to provide steady declines of success rate % while observing benchmarks from 100 to 1000 service requests. It has been noted that the success rate of the D1FTBC technique decreases as the number of service requests increases. With a rise in the number of transaction requests, the success rate naturally drops to a minimum, as is the case with any system that reaches its load.

Success Rate of Consistency Approaches

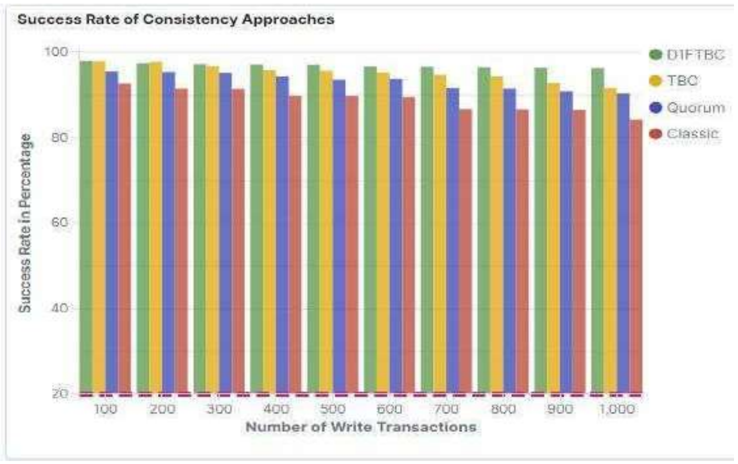

Figure 7: Performance Analysis of Success Rate for Consistency Approaches

The D1FTBC method outperforms other consistency algorithms when it comes to successfully executing Write transactions. A high success rate indicates that the suggested D1FTBC method outperforms the current consistency techniques, which is a direct outcome of the increased arrival rate's impact on write transactions.

Table: 2 Average Transaction Execution of Consistency Approaches

“Consistency Approach	Average Transaction Execution Time in (Ms)
D1FTBC	43.30
TBC	49.07
QUORUM	54.79
CLASSIC	52.08”

Table: 3 Average Transaction Success Rate of Consistency Approaches

Consistency Approach	Average Transaction Success Rate in Percentage %
D1FTBC	97
TBC	95
QUORUM	93
CLASSIC	89

Two metrics, a low execution time and a high success rate, guarantee the integrity of data transactions. Based on what we can see in Tables 2 and 3.

CONCLUSION

The proposed system has undergone rigorous experimental validation. Demonstrating exceptional performance, the suggested Depth 1 Fixed Tree-Based Consistency (D1FTBC) method exhibits a minimum execution time of 43.30 ms, surpassing alternative methods. Particularly, the Transaction-Based Consistency (TBC) method excels with an even superior performance at 49.07 ms. In terms of success rates, the TBC method achieves 95%, while the suggested D1FTBC method outperforms alternatives with an impressive success rate of 97%. This substantial improvement in performance and success rates underscores the efficacy of the suggested D1FTBC method in ensuring cloud data transactions attain a superior consistency state.

REFERENCES

1. J.Antony John Prabu, Dr.S Britto Ramesh Kumar, “An Integrated Architecture For Secured Cloud Data Transactions With Consistency Enhancements”, Annals of R.S.C.B, ISSN:1583-6258, Volume - 25, Issue - 4, 2021, pp: 18328- 18339, April 2021. (Scopus Indexed)

2. J.Antony John Prabu, Dr.S Britto Ramesh Kumar, “Performance Analysis Of Proposed D1FTBC Approach For Improving Consistency In Cloud Data Transactions”, International Journal of Scientific & Technology Research, ISSN: 2277-8616, Volume-8, Issue-08, pp: 1803-1807, August 2019. (Scopus Indexed)

3. J.Antony John Prabu, Dr.S Britto Ramesh Kumar, “Performance of Proposed Architecture for Data Transactions in Cloud using D1FTBC”, International Journal of Recent Technology and Engineering (IJRTE), ISSN: 2277-3878, Volume-8, Issue-2, pp: 5390 - 5395, July 2019. (Scopus Indexed)

4. J.Antony John Prabu, Dr.S Britto Ramesh Kumar, “Proposed Architecture for Improving Security and Consistency of Data Transactions in Cloud Database using Tree-Based Consistency Approach” International Journal of Computer Science and Information Security (IJCSIS), ISSN: 1947-5500, Volume-15, Issue-12, pp 118-126, December 2017. (UGC-CARE Journal)

5. J.Antony John Prabu, Dr.S Britto Ramesh Kumar, “An Efficient Depth 1 Fixed Tree Consistency (D1FTC) Method for Distributed Data Transactions in Cloud Environment”, International Journal of Advanced Research in Computer Science, ISSN: 0976-5697, Volume-8, Issue-7, pp: 936 – 942, July – August 2017. (UGC-CARE Journal)

6. J.Antony John Prabu, Dr.S Britto Ramesh Kumar, “Proposed Architecture for Secure Distributed Data Transaction Management in Cloud Environment”, International Journal of Applied Engineering Research - (IJAER), ISSN: 0973-4562, Volume-10, Issue -82, pp: 489 – 497, 2015. (Scopus Indexed)

7. J.Antony John Prabu, Dr.S Britto Ramesh Kumar, "Issues and Challenges of Data Transaction Management in Cloud Environment", International Research Journal of Engineering and Technology (IRJET), e-ISSN: 2395 -0056, p-ISSN: 2395-0072, Volume-02 Issue – 04, PP.: 123-128, July-2015. (UGC-CARE Journal)

8. Williams, Peter, Radu Sion, and Dennis Shasha. "The Blind Stone Tablet: Outsourcing Durability to Un trusted Parties." NDSS. 2009.

9. Varadharajan, Vijay, and Udaya Tupakula. "Counteracting security attacks in virtual machines in the cloud using property based attestation", Journal of Network and Computer Applications 40 (2014): 31-45.

10. Hsiao, Hung-Chang, et al. "Load rebalancing for distributed file systems in clouds" Parallel and Distributed Systems, IEEE Transactions on 24.5 (2013): 951-962, 2013.

11. Wei, Lifei, et al. "Security and privacy for storage and computation in cloud computing", Information Sciences 258 (2014): 371-386.

12. Mohammed M, Subramanian,"Enhancement of the Private Cloud Data Transaction by using an Orthogonal Handshaking Authentication Protocol (OHSAP)", International Journal of Computer Applications, Vol.96, No.23,2014.

13. V A Jane, BJ Hubert Shanthan, L Arockiam, "A Survey of Algorithms for Scheduling in the Cloud: In a metric Perspective", International Journal of Computer Sciences and Engineering, Volume 6,Issue 2, Pages 66-70, 2018.