www.ijarp.com ISSN 2456-9992 Page: 01-14

International Journal Advanced Research Publications

Case Study

Volume: 02
Issue: 07

EFFICIENT MODEL FOR DATA SECURITY IN DISTRIBUTED COMPUTING ENVIRONMENT: A COMPARATIVE STUDY

Jumbo Promise*[1], Okparaji Helen Onungwe [2], Umejuru Daniel[3]

[1]Department of Information Technology, University of Port Harcourt Choba, Nigeria.

[2,3]Department of Computer Science, University of Port Harcourt Choba, Nigeria.

Article Received: 12 June 2026, Article Revised: 01 July 2026, Published on: 21 July 2026

***Corresponding Author: Jumbo Promise**

Department of Information Technology, University of Port Harcourt Choba, Nigeria.

Doi: https://doi-doi.org/101555/ijarp.9544

ABSTRACT

In the Computing driven world which we are currently, and everything now digital, it shows coherent technological evolution and scientific innovations at its peak. However, these advantages also come with its own unique challenges as every business has some sensitive, crucial data and processes that need to be secured. The existing model falls short in addressing the issue of unauthorized user gaining access to distributed server without the admin approval. This study addressed the problem by developing an efficient approach for data security in a distributed computing environment. This is done by training the model with the agent parameters through Rivest, Shamir, and Adelman (RSA) algorithm leading to a more precise and optimal value. The new model provides availability of database and creates large space memory location that can handle bulk unstructured information in the distributed computing environment. The proposed system is built using Object Oriented Design Approach (OODA), JavaScript and Python Programming Language was employed at the backend while HTML and CSS was used as the frontend and MYSQL for relational database. Different sizes of files were uploaded to the distributed sever and time for execution (encryption and decryption) of these files, were determined to indicate the amount of time taken to encrypt each file depending on its size. The times taken in millisecond (ms) for existing and proposed system are: 12013ms, 28409ms, 59720ms, 63003ms, 41749ms, and 246ms, 158ms, 362ms, 191ms, 783ms, respectively. Some of the file sizes in megabyte used in the existing and proposed system are: 3.73mb, 16.2mb, 17.8mb, 18mb, 25.3mb and 46.19mb, 116.56mb, 188.83mb, 302.25mb, 6858.25mb respectively. Comparing the results, it is observed that the proposed model minimal time to execute files than the existing system.

Summarily, the result of the new model when tested shows that it can be used efficiently to achieve the principle of data security in a distributed computing environment.

KEYWORDS: Security, Data, Distributed Computing, Model, Encryption, RSA algorithm.

1.0 INTRODUCTION

There are so many distributed systems in operation nowadays which includes cluster computing, grid computing distributed storage systems and distributed databases. For some many organizations, it is absolute that they have a breach-free database [2]. While the rise of database computer technology used in the workplace has been exponential, business security methods and products have also been very minimal until recent years. More and more people are recognizing that their database system are extremely a valuable but also vulnerable resource, and need to apply the appropriate safety measures. [4] As with the protection of airports due to the recent terrorist events, the computer database security industry is just beginning to catch up with its need for better protection of hardware, software, data, and networks. Ensuring the security of data in large database system can be a difficult and expensive task. There are so many vendors and products that provide good security to Distributed Database Management System (DDBMSs).[10] Between database system security measures, security kernels, access control, encryption, cryptography, firewalls, intrusion detection systems, auditing mechanisms. Practically all large computer database systems are distributed. Huge business systems must support various users running common applications from across different locations. [3] A Distributed Database Management System (DDBMS) encompasses these applications, their underlying support software, the hardware they are run on, and more the communication links that connect the nodes. The largest and best-known DDBMS is the set of computers, software, and services comprising the Internet, which is so pervasive that it connects with almost every existing DDBMS. Security embraces many aspects of a distributed database management system (DDBMS), including hardware design, operating systems, networks, compilers, user applications programs, and then the users themselves.[8] Therefore, in this proposed system, knowledge has been derived from the limitations found in the previous technique in solving problems of data security in a distributed system. The idea in the proposed system is ensure and design a database system that can afford the amount of big data in the distributed system and to design several interface for each user with a security platform and encryption. Also, design a system for concurrency limitation in the existing method. Many textbooks and periodicals reads were used as well as

many subjects, including database security experts and former hackers, who were interviewed for this research. Also, trends of computer crimes and statistics were analyzed too. The conclusion is somewhat surprising though as Distributed Database Management System (DDBMS) security is not improving [6].

2.0 Concept of Data Security in Distributed Computing

The integration of diverse distributed components leads to emergence of new challenges from a security perspective. Thus, security has been identified as one of the major considerations when designing distributed systems [5]. The web, clusters, grids, and clouds are the pillars on which distributed systems are built. The security in distributed systems offers an overall perspective of the security considerations prevailing in today's world. [12]. The distributed computing system is the architectural style which enables a number of heterogeneous computers/workstations to behave and function as one single computing system. In the computing environment provided by the distributed computing system, users are able to have a uniform access to local/remote resources and be able to initiate processes from any part of the system [1]. The Distributed system is a group of (homo/hetero) genous automata connected via a network. Thus, the system can be viewed as a single machine. In a distributed system, each node exchanges information with other nodes through the exchange of messages over the network.[11] Different distributed resources (for example, files and printers) can be shared among all the nodes through different network services offered by various servers. Processes/clients directly request a specific server for accessing different resources. Therefore, the three main features of a distributed system include nodes, interconnection, and shared state. In this section, we discuss what distributed systems are and how to describe them. [13] "A distributed system is one in which the computer components appear to the user as a single coherent system, even though the physical structure consists of multiple connected computers." "A distributed system is a system in which the computer hardware and software components are physically disconnected from each other." From these two definitions it can therefore be noted that the distributed system can be termed as an application that interacts with various hardware and software that are scattered throughout in order to control the functioning of various processes carried out by various independent computers through the communication network such that all these components work together in order to achieve a particular aim.

3.0 METHODOLOGY

The research adopted Object-Oriented System Analysis and Design Methodology (OOADM).

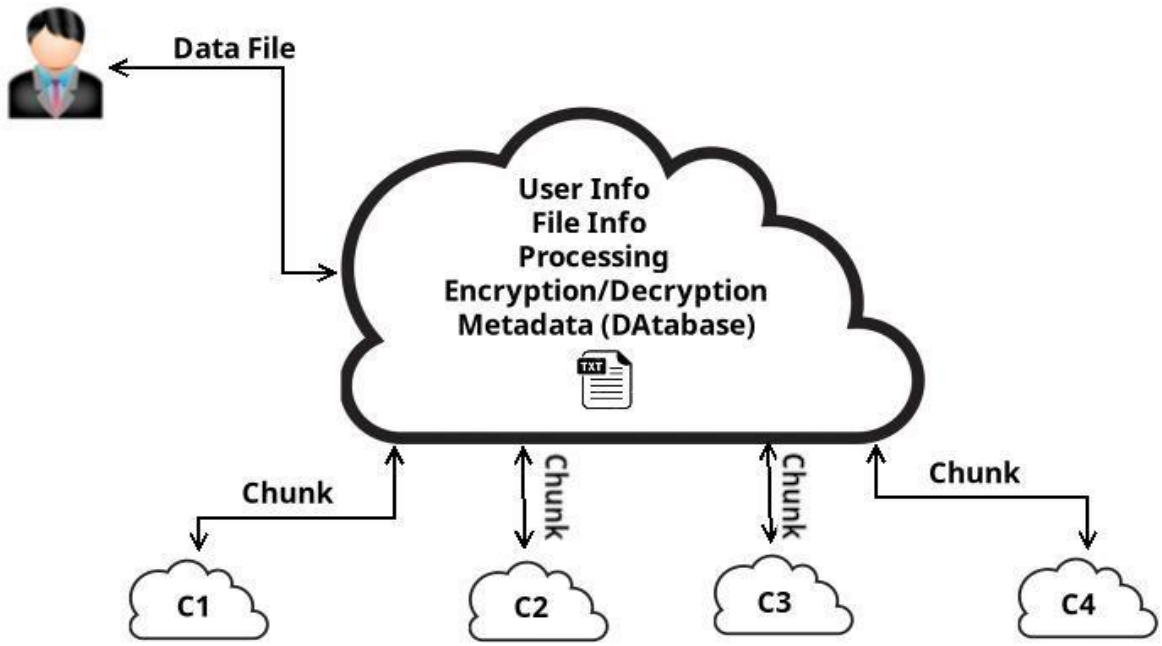


Figure 3.1: Architecture of the Existing System (Source: Munwar et al., 2021)

3.2.1 Explanation of the Existing System Components

The following Existing System components include:

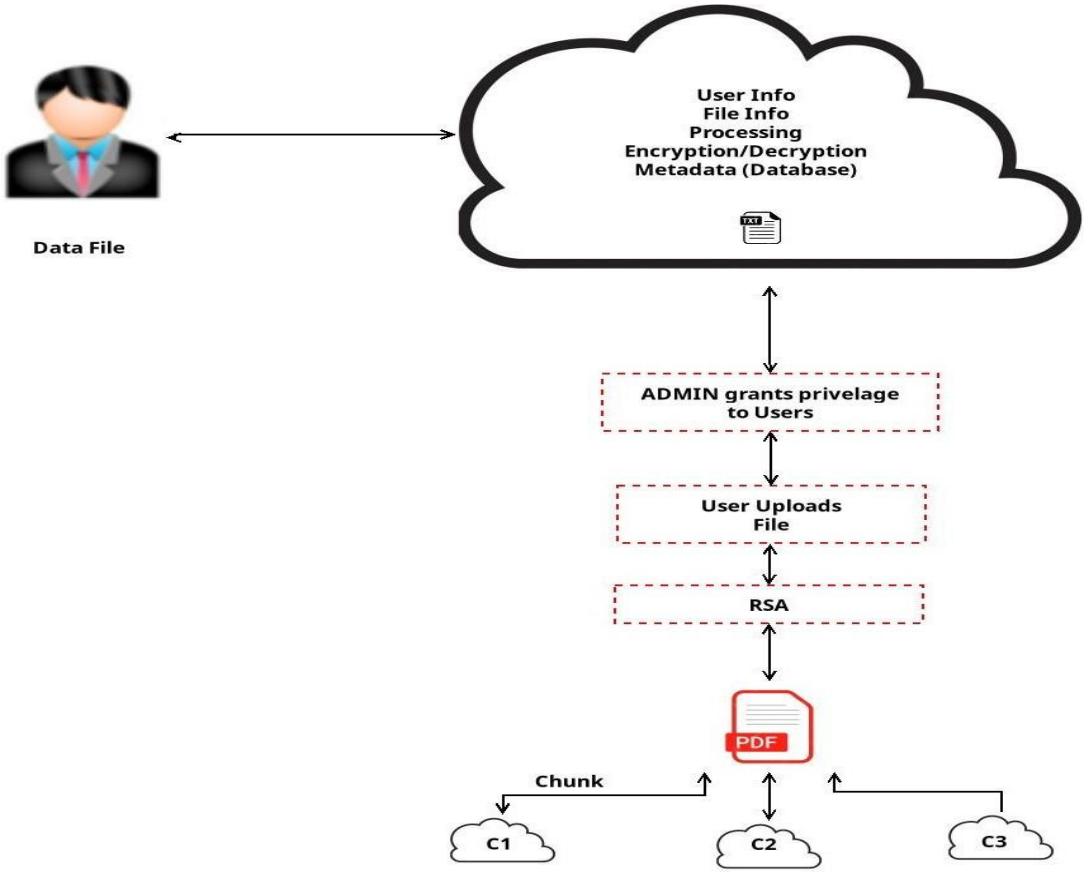
Data File: Users firstly upload data to servers via an internet connection, where it is then saved on a virtual machine on the physical server. To maintain availability and provide redundancy, cloud providers will often spread data to multiple virtual machines in data centers located across the world. Users can further access data in Cloud Storage through an internet connection and software such as web portal, browser, or mobile app via an application programming interface (API) [9].

File Information: The file information is the collection of the data stored in the distributed system. Basically, these files are used for storing information about users, software applications, or other data that are needed for running of the programs.

Processing: Processing system enables to read, write, modify, and store data too. This involves different transactions performed by the user. Some of the transaction includes definition of a new transaction, deletion of existing transaction, updating of the transaction details, receiving the list of transactions authority and then receiving authority for transaction.

Encryption/Decryption: Encryption is implemented to convert readable message into an unreadable form to in other prevent unauthorized parties from reading it while decryption converts an encrypted message back to its original (readable) format.

Database Base: This is exactly where all individual files are stored. Encryption of transaction is applied to each user file and transactions performed.



3.3.2. Modules Description

The Proposed System components include:

Data File: Users upload data to servers via an internet connection, where it is then saved on a virtual machine on the physical server. To maintain availability and provide redundancy, cloud providers will often spread the data to multiple virtual machines in data centers located across the world. Users can then access data in cloud storage using an internet connection and software such as web portal, browser, or mobile app through an application programming interface (API).

File Information: File information is the collection of data stored in the distributed system. Basically, these files are used for storing information about users, software applications, or other data that are needed for running of the programs.

Processing: Processing system enables it to read, write, modify, and store data. This involves different transactions performed by the user. Some of the transactions include: definition of a new transaction, deletion of existing transaction, updating of the transaction details, receiving the list of transactions authority and receiving authority for transaction.

Encryption/Decryption: Encryption is implemented to convert readable message into an unreadable form in other to prevent unauthorized parties from reading it while decryption converts an encrypted message back to its original (readable) format.

Database Base: This is where all individual files are stored. Encryption of transaction is applied to each user file and transactions performed.

Role-Based Access Control: The roles of the users are grouped based on their common responsibilities. Each user is assigned one or more roles and needs one or more permissions to each role. The user role and role permissions relationship makes it simple enough to perform user assignments since user no longer needs to be managed individually, but rather have privileges that conform to the permissions assigned to their role

3.3.3 Algorithm for the Proposed System

In developing an efficient model for data security in distributed environment, Rivest, Shamir, and Adelman (RSA) algorithm was applied. The algorithm is based on asymmetric encryption and decryption principle .In this algorithm public key is distributed to all, through which anyone can encrypt the message and a private key which is used for decryption and kept secret and is not shared to everyone. Set of Functions in the Algorithm are:

NumberOfBlock(F) : It returns the number of block in the file F. ENC_AES (B,K) : It encrypts the block B with key K.

send_to_cloud(F') : It permits to send the encrypted file F in Cloud storage ENC_RSA(k) : It encrypts k utilizing RSA Algorithm.

Save_in_server(K') : It permits to save K' in the server NumberOfBlock(F) : It returns the number of block in the file F. DEC_RSA(k') : It decrypts k' using the RSA Algorithm.

DEC_AES(B',K) : It decrypts the block B with key K.

Algorithm 1: Algorithm for File_Upload

Step 1: Encrypt_file (F)

Step: 2 Obtain a public key of recipient Pu=$\{n, e\}$ to calculate the cipher: C=M^emod(n), where 0≤M<n.

Step 3: To transform Clair text in file F into Cipher text in file F'. Set $a^{\emptyset(n)}$mod(n) = 1

where gcd

(a,n)=1 and calculate n=p.q such that $\emptyset(n)$=(p-1)(q-1) then chose e and d to be inverses mod ø(n).

Step 4: For B←1 to numberOfBlock(F) do Step 5: Else

Step 6: B'=ENC_AES(B,K)

Step 7: else

Step 8: send_to_cloud(F')

Step 9: For k←1 to SizeOf(K) do Step 10: else

Step 11: k'=ENC_RSA(k)

Step 12: decryption of the recipient using their private key Pr={n, d} and computes: M=C^dmod(n).

Step 13: end for statement Step 14: Save_in_server(K')

Step 15: Exist

Algorithm 2: Algorithm for File_Download

Step 1: Decrypt_file (F')

Step 2: obtained using private key Pr={n, d} and computes: M=C^dmod(n). Step 3: transform the Cipher text in file F' into Clair text in file F

Step 4: For k'←1 to SizeOf(K') do Step 5: Else

Step 6: k=DEC_RSA(k')

Step 7: else

Step 8: return(K)

Step 9: apply Phase 2: Decrypt Cipher text Step 10: For B'←1 to numberOfBlock(F') do Step 11: else

Step 12: B=DEC_AES(B',K)

Step 13: end of statement Step 14: return(F)

Step 15: Exist

4.0 RESULTS

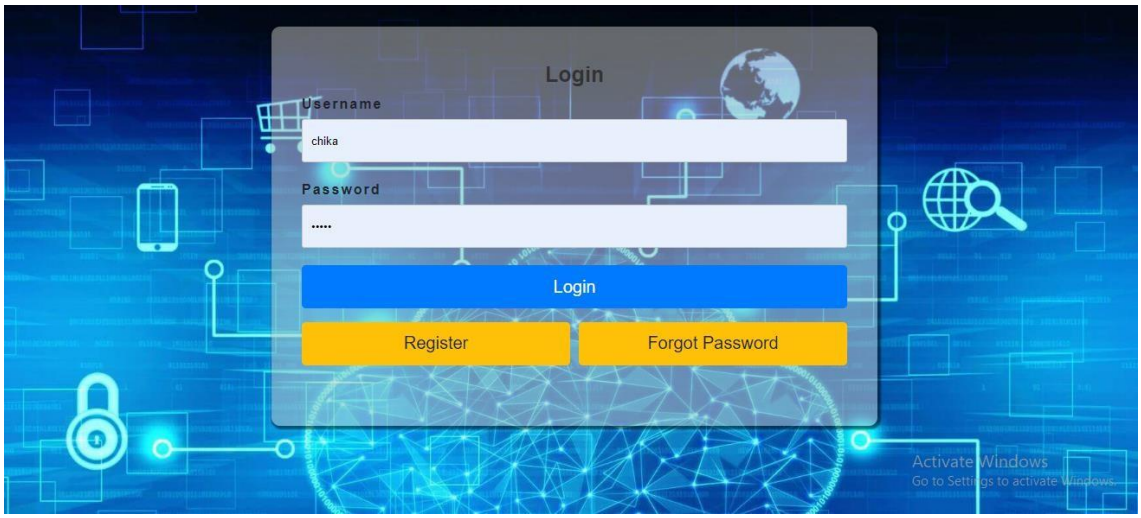


Figure 4.1: Data Security Interface.

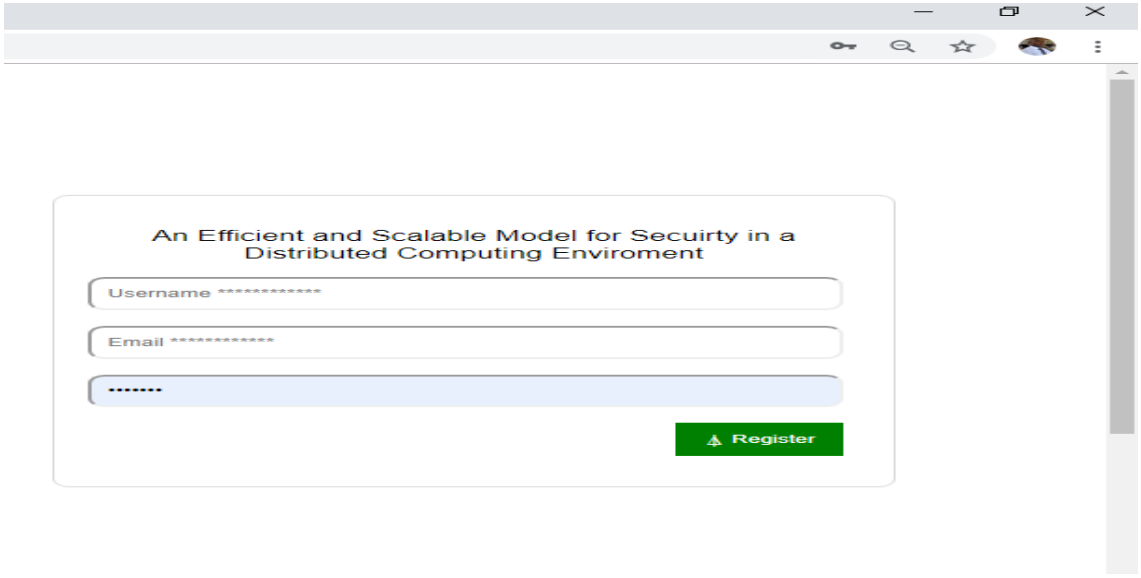


Figure 4.2: Access Granted Page.

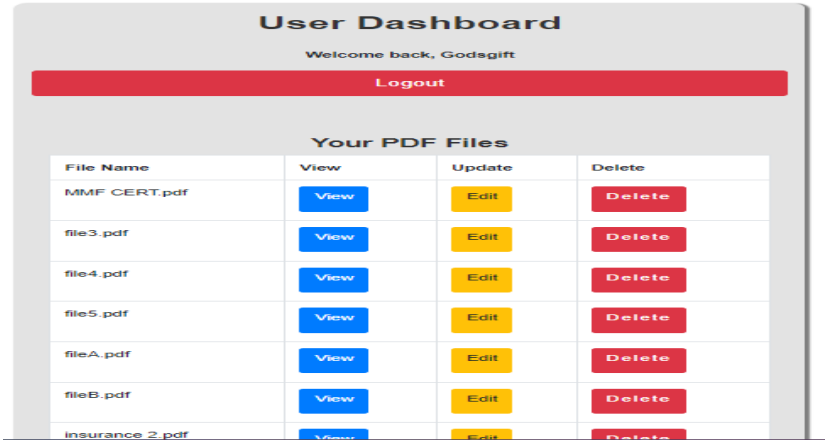


Figure 4.3: User Dashboard.

Figure 4.4: List of Encrypted Files.

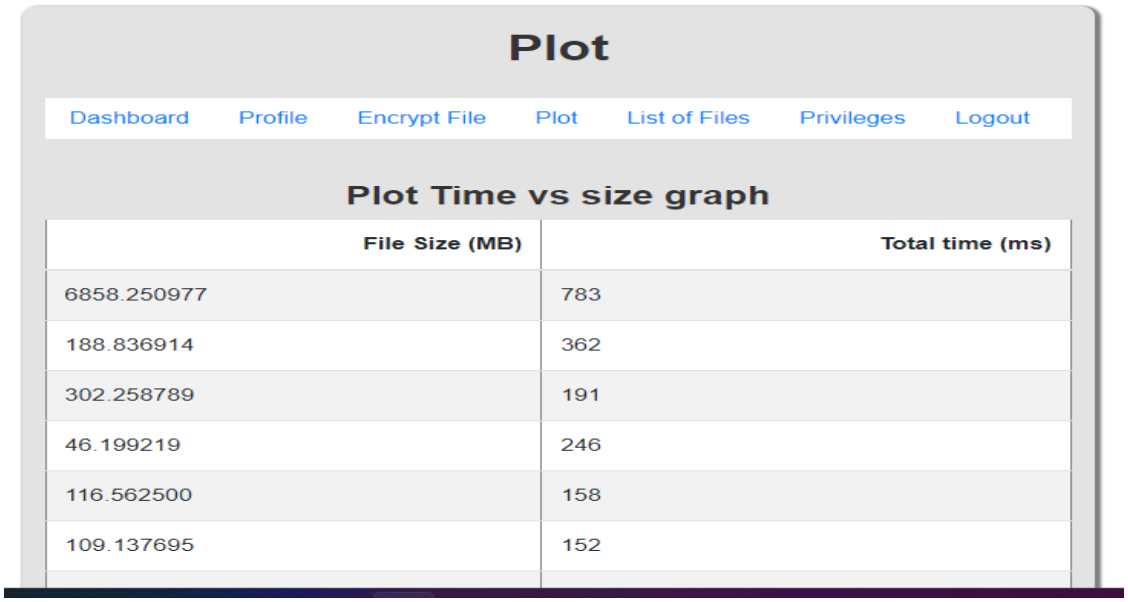


Figure 4.5: Encrypted Files with their sizes and Total Time.

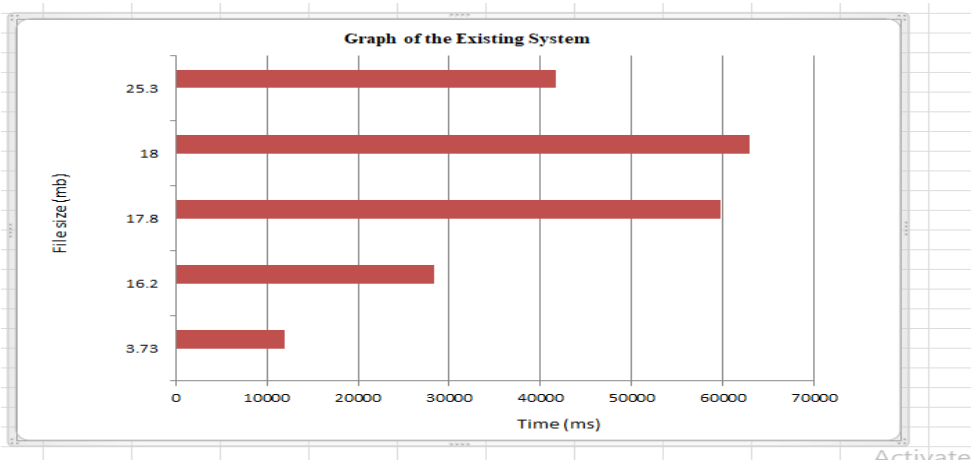


Figure 4.6: Existing System Graph Evaluation.

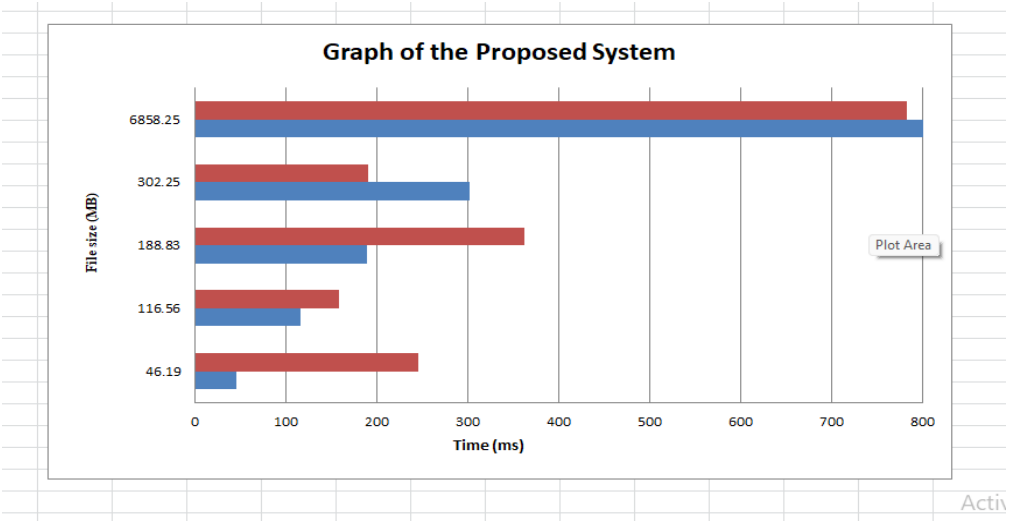


Figure 4.7: Proposed System Graph Evaluation.

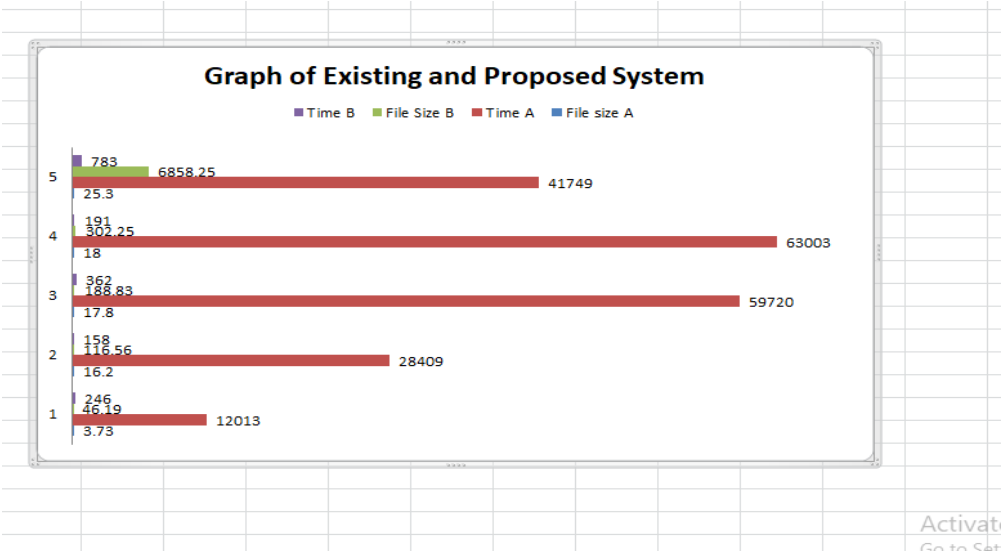


Figure 4.8: Result Performance Evaluation Graph (Existing System vs. Proposed Systems)

Table 4.1: Result of the Existing System Evaluation Performance.

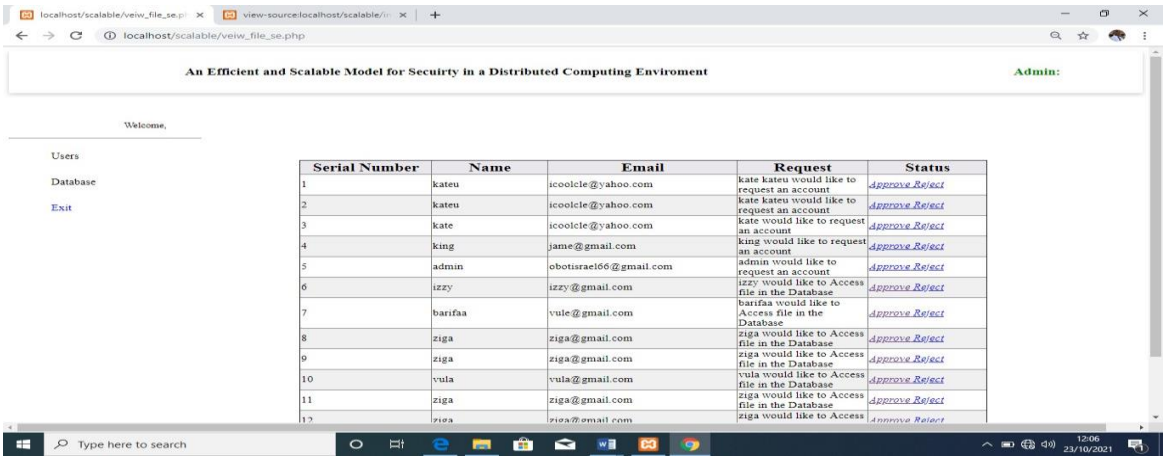
S/N	File Size (MB)	Total Time (ms)
1	3.73	12013
2	16.2	28409
3	17.8	59720
4	18	63003
5	25.3	41749

Table 4.2: Result Performance Evaluation of Proposed and Existing System.

Existing system			Proposed system		
S/N	File Size(MB)	Total Time(ms)	S/N	File Size (MB)	Total Time(ms)
1	3.73	12013	1	46.19	246
2	16.2	28409	2	116.56	158
3	17.8	59720	3	188.83	362
4	18	63003	4	302.25	191
5	25.3	41749	5	6858.25	783

Table 4.3: Test Case 1.0. (Login)

Test Case Number	**1.0**
Case Name	Login
Precondition	The application has been lunched and the login form is loaded.
Case Input	Enter username and password
Case Expected Output	The dashboard (Home Page) is successfully loaded
Case Steps/Description	**Input**: Type in username and password in their respective textboxes and click on the login button **Output**: The Dashboard (Home Page) is loaded Successfully if the username and password is Valid. Else, a failure message is displayed for Invalid username or password.



4.7: Admin Database Page

4.1 Comparative Analysis between Existing and Proposed System

A comparative analysis is conducted between the existing system and the proposed system in

terms of user's password authentication. In the proposed system; a password and admin approval are required for file access. In the event of a password breach, files cannot be easily accessed; instead, the admin must verify the user before granting access to the files. The proposed system uses the Rivest, Shamir, and Adelman (RSA) algorithm. The algorithm is based on an asymmetric encryption and decryption principle. In this algorithm, public key is distributed through which one can now encrypt the message and a private key which is used for decryption, is given by the admin and is kept secret and is not shared to everyone. This algorithm further suggests the encryption of the files to be uploaded on the cloud. The integrity and confidentiality of the data uploaded by the user, is now ensured by not only encrypting it but also providing access to the data only based on successful authentication. The authorized user can now also download any of the uploaded encrypted files, and also read it on the system. In the proposed model, the record of different individual users that attempted to gain access in the encrypted files in the cloud system that was denied access were recorded as shown in study. The existing system uses Blowfish Algorithm, which has symmetric-key principle, which is a known class of algorithms for cryptography that makes use of the same cryptographic keys for both encryption of plaintext and also decryption of the cipher text. These keys may be identical, or there may even be a simple transformation to go between the two keys. The keys, in practice, only represent a shared secret between two or more parties that can be used to also maintain a private information link.

4.2 Discussion of Result

The intent of this research is basically just to design and implement an Efficient Model for Data Security in a Distributed Computing Environment with improvements on confidentiality and data integrity on the existing system. The proposed system also provides security in the distribution of various files or information uploaded by different clients or users into a computing environment or distributed system. Different sizes of files or documents were uploaded to the distributed server and the time for execution of these files were determined to indicate the amount of time taken to encrypt each file depending on its size. The times taken in millisecond (ms) for existing and proposed system are: 12013ms, 28409ms, 59720ms, 63003ms, 41749ms and 246ms, 158ms, 362ms, 191ms, 783ms respectively. Some of the file sizes in megabyte used in the existing and proposed system are; 3.73mb, 16.2mb, 17.8mb, 18mb, 25.3mb and 46.19mb, 116.56mb, 188.83mb, 302.25mb, 6858.25mb respectively. Comparing the result, you will observe that the proposed model takes very minimal time to execute a file than the existing system. Hence, the result of the new model

when tested showed it can be used to achieve security principle in a distributed computing environment.

5.0 CONCLUSION

The various challenges associated with distributed cloud database system are security principle in terms of confidentiality, data integrity and availability of memory space. Ensuring that system, and important data transmission between parties or different uses in a distributed computing environment is kept fully undisclosed to unauthorized entities or owner. For the newer distributed architectures, such as wireless and peer-to-peer, this challenge becomes more interesting due to factors such as lack of structure and lack of trust in the network. This has increased the risk operational disruption, financial losses, legal issues, compliance penalties and reputational damages as time evolved. Maintaining this gained accuracy and consistency of data over its entire life-cycle is very paramount and important. Hence, the need for a more scalable model for data security in distributed database system is developed. The developed scalable Model for Data Security in a Distributed Computing Environment, will proffer lasting solution to the security challenges in a distributed database system. The following are the summary of the new model: The system has been able to reduce intrusion in the cloud database system where by users cannot go and access files without admin approval. Secondly, an improved security confidentiality, data integrity and availability web-based application, which can be accessed globally at any given time as long as there is internet access has been borne. Also, an improved large storage availability system that will accommodate unstructured data and finally, openness and extensibility of different user interfaces was separately design and publicly available to enable easy extensions.

REFERENCES

1. Ali, W. (2022). Comparative analysis on different parameters of encryption algorithms for Information Security, International Journal of Machine Learning and Computing (IJMLC), 8(4), 319 – 323
2. Belapurkar (2021). Security shapes future directions in the context of todays distributed systems. Journal of Parallel and Distributed Computing, 67(10), 1067- 1081.
3. Chang (2021). Computing methods for efficient analysis of PEPA models of non-repudiation protocols, 15th International Conference on Parallel and Distributed Systems
4. (ICPADS), 821- 827.

5. Coulouris R. (2023). Enhanced information security in distributed mobile system based on delegate object model, Procedeia Engineering, 34(30), 774-781.
6. Fernandez A. (2022). A policy driven trust management framework in iTrust. This translates into more reliability as in case of a single machine, International Journal of Software and Development Research (IJSD) 13(54): 301-314.
7. Firdhous, T. (2021). Performance evaluation of a new scheduling algorithm, Journal of Parallel and Distributed Computing, 6(20), 167- 181
8. Guynes (2021). Database security in a client/server environment. an access security system operating only in a local area network environment (LAN) with many stations and users. Journal of Distributed and Parallel Databases. 21(32): 124-541.
9. Mingyao, Lixongtei. (2020). Embedded database query optimization algorithm based on particle swarm. Seventh international conference on measuring Technology 1-14
10. Munwar et al., (2021) distributed file sharing and retrieval model for cloud virtual environment, International Journal of Engineering Technology (IJET), 9(64), 196-202.
11. Tanenbaum & Steen (2024). Secure password authentication for distributed computing,International Conference on Computational Intelligence and Security, 2(21): 1345-1350.
12. Wald, E. (2010). A distributed system over the centralized system in its modular extendibility. IEEE Transactions on Computers, 60(7): 1017-1029.
13. Yi, C. (2015). An efficient xpath query processor for xml streams. International Journal of Computer Applications (IJCA), 10(7), 33 – 39.
14. Yong, Y. (2016). The cougar approach to in-network query processing in sensor networks. International Journal of Computer Applications (IJCA), 10(9), 44 – 4