National College of Ireland

Secure and Efficient Cloud Storage Using Fog Computing and Regenerating Codes

MSc Research Project
Cyber Security

Pranav Prasanna Ghorpade
Student ID: X22197044

School of Computing
National College of Ireland

Supervisor: Michael Pantridge

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Pranav Prasanna Ghorpade
Student ID:	X22197044
Programme:	MSc. Cyber Security **Year:** 2023-2024
Module:	
Supervisor:	Michael Pantridge
Submission Due Date:	12/08/2024
Project Title:	Secure and Efficient Cloud Storage Using Fog Computing and Regenerating Codes
Word Count:	**6774** **Page Count: 22**

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Pranav Prasanna Ghorpade

Date: 12/08/2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Secure and Efficient Cloud Storage Using Fog Computing and Regenerating Codes

Pranav Prasanna Ghorpade

X22197044

Abstract

Cloud storage services have always gained attention, and this has driven a compelling need for new solutions to data privacy, while overcoming the limitations of latency due as well as availability. This project demonstrates a different storage approach inspired by the ideas of fog computing and that uses edge resources to augment classical cloud services. In this paper, the proposed micro service-based distributed storage system is able to alleviate these drawbacks of traditional cloud storage by using fog computing in its architecture. The data itself is subject to a strategic placement algorithm that causes the data be distributed across multiple storage elements similar to with information theory and cryptography. Our system excels in preserving user privacy and ensuring high availability, with this proposed mechanism providing up to 50% fault tolerance effectively using 100% extra storage. By utilizing regenerating codes it reduces repair bandwidth and guards the data from being corrupted. Furthermore, it increases the security and availability of files by dividing each file into several parts that are stored in separate storage places. It can be seen as a part of the more general plan to enable decentralized, privacy preserving and highly available storage solutions to compete with conventional cloud storage systems by presenting public auditing scheme for data integrity enrolled in them.

1 Introduction

Online storage services have become an innovative way of storing and sharing data mostly because of their flexibility, scalability, and cost efficiency for both the individual and organization I. Gupta et al. (2022). But as more and more data were moved to the cloud several essentials challenges arose; some of them included privacy, latency, and availability challenges. The maintainability of data privacy was an important concern due to the high risk of loss that company's could incur from data hacks in the cloud storage environment. Moreover, previous approaches closely associated with cloud storage systems were limited in terms of low latency access and high availability especially during the occurrence of network or server failure A. Ghani et al.,(2022).

Another feature that was particularly essential for any advanced storage system was fault tolerance it was a fact that some of the parts of the storage system and network might develop faults at one time or the other. Conventional fault tolerance techniques are mostly based on redundancy; for instance, used replication in which copies of data were stored in different nodes Ganesan, A. , et al.(2017), although effective, it required large storage space. Specifically, in the sphere of cloud

security, protection of data from unauthorized access and data authenticity during storage and processing were the key objectives. It became necessary to use cryptographic techniques and to have secure auditing mechanisms that would enable the users to have confidence in the security of that information being stored.

Fog computing as a continuation of cloud computing shifted data storage and computation to the data origin by utilizing edge resources D. Shah et al. ,(2021). This paradigm reduced the latency and thus gave cloud services' faster feedback especially for cloud applications that needed real-time processing. Considering the inclusion of components of fog computing into cloud storage, it was possible to improve its performance and availability due to the distributed architecture of the fog nodes. Traditional cloud storage systems often struggled with several limitations: privacy problems as the storage can become a single point of failure, latency as far as interconnect distance to data centers, availability problems during network outages or server crashes, and the problem of needing a lot of storage for good redundancy.

Research Question:

How can the integration of fog computing and regenerating codes in a distributed cloud storage system improve data privacy, reduce latency, enhance availability, and achieve 50% fault tolerance?

This study was devoted to the resolution of this problem by implementing the technology of fog computing and regenerating codes in cloud storage. Our aims were as follows: enhancement of privacy due to cryptographic means; improvement of availability by means of better eat utilization; the fault tolerance rate of 50% when 100% extra storage is used in an efficient way; minimization of repair bandwidth thanks to regenerating codes and gain in security and availability through the logical splitting of file blocks and their storage in different places. Such a project implies significant progress in the realm of storage solutions as it seeks to guarantee the superior strength, high available and privacy-protective features missing in conventional cloud storage.

Scope

The scope of this work is centered in enhancing security, privacy, and availability of cloud storage systems using the combination of fog computing and regenerating codes. The outcomes of the study are limited to the system solely being tested with text files. To expand the scope of the project, further testing should be done with other file types, for example, with images, video, or large datasets.

Limitation

The main limitation of this scheme is the increased space complexity due to 100% additional capacity overhead for fault tolerance. If more than two cloud nodes die at once there is a good chance you might not be able to download your files. These all have higher complexity code and

to higher computational and maintenance costs for fog cloud, with likely performance drawbacks if they run on limited resource network endpoints.

2 Related Work

The literature review focuses on the recent trends in fault tolerance, data recovery, and security issues in Cloud & FOG computing; which identifies gaps that are filled by the proposed design.

Enhancements in Regenerating Codes

To have a better communication efficiency and computation efficiency, Fu et al. (2020) presented decoding and repair schemes for shift XOR product matrix regenerating codes. Their schemes provided the maximum transmission bandwidth and in-place implementations while using no additional memory to store the intermediate XOR results. The availability of inplace algorithms, shift XOR elimination had exerted a lesser computation complexity from the previously used zigzag algorithms. This work showed lesser computation costs and possibility of it being implemented in multibit computation devices (Fu et al., 2020).

Data Recovery and Update Optimization

SARSR was proposed by Zhang et al. (2022) as a random search recovery algorithm that is used to improve the data recovery of the XOR based erasure coded distributed storage systems. SARSR is implemented by using simulated annealing, which decreased the amount of data needed for recovery and the proposed method performed better than prior approaches (Zhang et al. , 2022). Xiao et al. (2023) presented the DXRDU scheme in order to enhance the have to bring out more cross rack data update traffic and overall throughput of the erasure-coded cloud storage system (Xiao et al. , 2023). Chen et al. (2015) examined the effects of assured cloud storage service and assured network coding with the introduction of an efficient Public Verifiable Secure Cloud Storage Providing Scheme with user anonymity and third party auditing. This opens up the way for more research on ascertaining through verification if secure Network coding protocols can be derived out of secure cloud storage protocols (Chen et al. , 2015).

Fault Tolerance Strategies in Cloud Computing

Next generation IoT systems are communicating over fog computing; to handle the security issues in the communication process, Wang et al., (2020) presented a secure authentication scheme for fog computing. Its authentication employed device IDs, pseudonyms, and truly random numbers for Alice, Bob, and Charlie; while their dynamic session keying entailed an HTTP hash symmetric key physical layer HSK. The scheme also achieved the fog computing architecture security requirements at a lower overhead in terms of secure transmission of data and MUTUAL authentication of nodes (Wang, Zhang, Li, & Ren, 2020). Ali & Kareem (2021) suggested that BEC encryption be adopted to increase both the security and performance of the IoT networks. In their approach, they replaced the ECC algorithm with BEC and implemented a fog layer to perform

the user authentication, which enhanced the level of privacy and lowered the dependence on public clouds. Same payouts demonstrated that, compared to prior approaches, BEC cut down time seemed on the secrets and on confirmation, decreased the center traffic, and enhanced such privacy as group intercession (Ali & Kareem, 2021).

Security Enhancements in Fog Computing

Wang et al. (2020) proposed a secure authentication scheme for fog computing to address communication security concerns in next generation IoT systems. Their scheme used device IDs, random numbers, and pseudonyms for authentication, with dynamic session key generation using hash, symmetric key, and physical layer techniques. The scheme met fog computing architecture security requirements with lower overhead, ensuring secure data transmission and mutual authentication among nodes (Wang et al., 2020). Ali and Kareem (2021) proposed the use of BEC encryption to enhance IoT network security and efficiency. Replacing the ECC algorithm with BEC and introducing a fog layer for user authentication, their approach improved privacy and reduced reliance on public clouds. Simulations showed that BEC outperformed previous methods in encryption and authentication times, reduced network traffic, and provided better privacy through group encryption (Ali & Kareem, 2021).

Privacy Preserving Techniques and Data Deduplication

Mahdikhani et al. (2020) presented a privacy preserving range query scheme for fog based IoT. Their decomposition technique employed She to securely share query paths thus enhancing communication effectiveness. From the area of privacy, security analysis substantiated the preservation of privacy in the proposed scheme, and concerning the area of performance, the results presented that the scheme was faster than prior schemes (Mahdikhani et al. , 2020). Several works have been proposed for enhancing the efficiency of cloud storage: recently, Yoosuf et al. proposed in 2022 a fogcentric deduplication framework named FogDedupe based on multikey homomorphic encryption to solve problem related to bandwidth and to improve cloud storage efficiency. According to Yoosuf et al. (2022) the developed framework offered enhanced storage overhead by 27% and brought down the deduplication latency by 12%.

Cryptographic Approaches in Cloud Computing

Al Issa et al. (2022) proposed a new encryption technique based on the use of XOR operation and genetic algorithms for cloud data. Their strategy focused on protection against data and protection against key loss basing data into files that were interdependent and hence stored separately. However, the method failed to solve all the privacy problems and other recommendations that could be implemented included random storage algorithms (Al Issa et al. , 2022). In this context, Sasikumar and Nagarajan (2024) discussed different categories of cryptography in the context of cloud environment: ECC and hybrid cryptography. They stressed on exploring such modern cryptographic technologies as DNA computing for better security (Sasikumar & Nagarajan, 2024).

Fault Tolerance in IoT

Saeed et al. introduced their innovative Fault Tolerant Data Management scheme to address critical issues within fog computing-based healthcare IoT systems, namely latency, energy efficiency, and fault tolerance. By designing task-based and node-based mechanisms, their FTDM approach demonstrated notable enhancements in performance and patient safety through simulations. Results showed reductions of 3.97% in energy consumption, 5.09% lower execution costs, network usage decreased by 25.88%, latency dropped by 44.15%, and execution time was reduced by 48.89% compared to the conventional Greedy Knapsack Scheduling method. The research highlighted FTDM's effectiveness, particularly for remote patient care during public health crises such as the COVID-19 pandemic. Moving forward, incorporating service level agreement-based techniques into their design presents an engaging area for future study, as outlined in their findings.

Gap Analysis

While previous studies have made notable progress in developing data retrieval, error handling, and protective strategies for cloud and fog computing platforms, an opportunity remains to comprehensively integrate these components into a unified structure guaranteeing resilient fault management along with strengthened security, especially regarding cloud storage architectures. In a way, the suggested cloud-based system by this work exists as an approach to filling this gap since it is a robust approach to combining advanced fault tolerance with enhanced cryptographic security. Importantly, data stored in this system is managed in a way that is secure and seamless and eliminates the active concerns related to both computational efficiency and security

3 Research Methodology

The research methodology section describes how the cloud storage system that uses fog computing and regenerating codes is to be created. It comprises identification of the research method, and instruments used, embracing literature survey, system architecture, deployment, improvement for functionality, security, and reliability, respectively.

3. 1 Research Procedure

The following section describes the research methodology to establish a cloud storage system with reference to fog computing and regenerating codes targeting the design, implementation, and analysis of the performance and privacy aspects.

Literature Review: Thus, it is crucial to carry out a literature review of cloud storage systems and fog computing and the application of regenerating codes. Concentration will be made on stating current issues regarding to privacy, latency, availability, and fault tolerance. Discuss what has been done in recent years and outline the techniques used for tackling these issues and what should be used in your proposed system detail.

System Design and Architecture: Design the cloud storage model that incorporates fog computing and regenerating codes and describe it in details. Develop a complex structure involving client interfaces and displays, servers' functioning and processing, and distributed storage systems. It is required to state the fog and cloud storage layers' configuration and draw specific system layouts for future deployment.

Implementation: There is another recommended system and follow the guidelines of that system on the design of this system. This involves creating the applications interfaces, the backend servers, and the incorporation of the fog and cloud storage nodes. Make sure the system can divide the data block, generate hash codes, regenerate codes, and Quick Retrieval Technique (QRT). Carry out unit and system tests to check if the devices are functional and if they perform efficiently.

Evaluation and Analysis: Evaluate how effective the system is, in addition to its capability to maintain privacy and reliability. Find out the degree of latencies, availability and fault tolerance. Actual time to get the test data and the expected repair bandwidth. Analyze the efficiency of the privacy-preserving mechanisms and compare the results of the system with the general cloud storages.

3.2 Equipment and Tools

• **Hardware Specifications:** Hardware requirements include an Intel Core i5 or later, 16 GB RAM and 512-GB SSD. Particularities of the built specifications guarantee that the performance of development and deployment processes is reasonable.

• **Software Requirements:** Concerning software requirements, developers should use Windows in the development process. Java Technology for the development of the application using MVC Architecture and for backend database management SQL is used. Integrated Development Environment Eclipse is used as development environment**.**

3.3 Techniques Applied

This section describes the methods employed in the development of the cloud storage system identified as fog computing, regenerating codes, hashing algorithms, quick and cryptographic methods of data retrieval.

1. Fog Computing: It builds on cloud systems and in-premises data preprocessing and storage indicated to be more efficient and fast. In this project, fog nodes play the role of the relay of helping the system to get and cache most demanded data thus improving the throughputs and reliability of the system.

2. Regenerating Codes: Regenerating codes are used to maximize the number of bad nodes it can cope or the extent of bandwidth necessary for reinvesting the nodes in distributed storage systems. This technique, however, synchronizes effective ways of data recovery because lost data can be

computed from a few stored blocks. In the proposed system, regenerating codes are used for creation of participant's copies, which can help in reconstruction of original data in situation with node failure and therefore increases the part of the system's fault tolerance and decreases the amount of necessary correction.

3. SHA Hashing Algorithm: SHA generates tags, especially, SHAs making it easier to confirm the authenticity of the information or the block's location. This process helps in detecting cases of data tampering or data corruption hence enhancing the security and the reliability of the data.

4. Quick Retrieval Technique (QRT): Quick Retrieval Technique (QRT) is designed to improve the execution of the data retrieval matters through hash codes and storage management affairs. QRT is utilized in an effort to fasten a procedure of how a file can be either accessed or retrieved depending on rational mapping and regeneration of data blocks. Consequences can concern the timely and quick restoration of files using some of the storage nodes and others can be offline or problematic.

5. Cryptographic Techniques: Encryption is employed for safeguarding the data information at the time of storage as well at the time of transmitting it from one place to the other. This entails the generation of MACs for data blocks to verify the originality of the current data and its sanctity. The integration of cryptographic techniques is the best solution for the challenge because data is protected from the unauthorized individuals' access.

4. Design Specification

This section explains the design specification where some of the key areas include the architecture of the secure cloud and fog storage system, user interface, and functions of the proxy server as well as the incorporation of superior approaches in managing the system.

Proposed Work

The advanced techniques are incorporated for fault tolerance, data integrity improved storage and efficient files managing of data in cloud and fog environment to make use of the proposed system. It has multi-layered structure where different friendly user interfaces are implemented along with back servers and storage architectures and also it is built in a way that data is quickly retrievable and in case of data loss or data corruption the system can be easily recovered quickly.

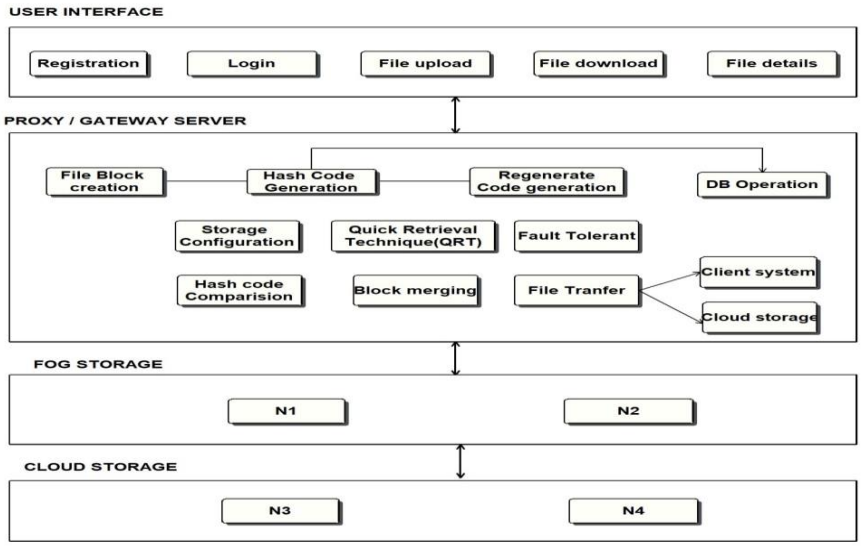

Figure 1: Proposed System Architecture

The proposed system architecture as in figure 1 is a systematic and multiple level for backing up and retrieving data over both clouds and fog with increased data security and reliability.

User Interface Layer: The user interface layer outlines a flexible working interface WHERE the user is able to perform basic operations on registry such as login, uploading and downloading of files and their details. This layer allows for a fast response when using the core functionalities of the system.

The proxy/gateway server: performs the role of the CPU that all dependent processes linked to the receipt, storage, and retrieval of information and data are dealt with in this server.

- **File Block Creation:** The files that are being received from HSSC other departments and offices are divided into portions to increase the storage density and reliability.
- **Hash Code Generation:** SHA hashing technique is also employed, to obtain the hash codes for the file blocks, for data checking and to reduce the time that it takes to search for specific blocks of files.
- **Regenerate Code Generation:** This module enhances the fault tolerance operation in that it produces codes in the processing of data whereby if corroded or lost the codes can reconstruct them. These regenerating codes were developed with XOR operation and these also lead to data hiding.
- **Database Operations (DB Operation):** The contents of all the information include the file information while the hash codes are stored and regulated adequately within the database.

- **Storage Configuration:** As for the size and location of Data blocks in the Cloud and Fog layers, they are the responsibility of the Data broker.
- **Quick Retrieval Technique (QRT):** This technique is applied with the usual aim of enhancing the speed of sorting through the data as well as proper space to store hash codes.
- **Fault Tolerance:** It is possible to provide measures that could be taken when there are failures within the system so that data integrity and consistency could be upheld by principles such as data duplication and checking of errors.
- **Hash Code Comparison:** This step involve; comparing hash codes for the purpose of checking on the integrity of the data as well as detecting duplication.
- **Block Merging:** It can reconstruct the file blocks back to the original file during the Download or Data Recovery Procedures.
- **File Transfer:** Oversees the transportation of data from the client system to the fog and cloud stores with the desired levels of security and performance.

Fog and Cloud Storage Layers: In the present context, the fog storage layer is shown as nodes of N1 and N2, essentially it is an intermediary storage but with the advantage of low latency storage and processing. The cloud storage layer with nodes N3 and N4 is used for long term archival and additional redundancy improving availability and reliability.

The architecture of this system aims at achieving the best of both worlds between Fog and Cloud Storage while at the same time enhancing the performance, security, and reliability of the system as applied to the users.

4.2 Fog Computing:

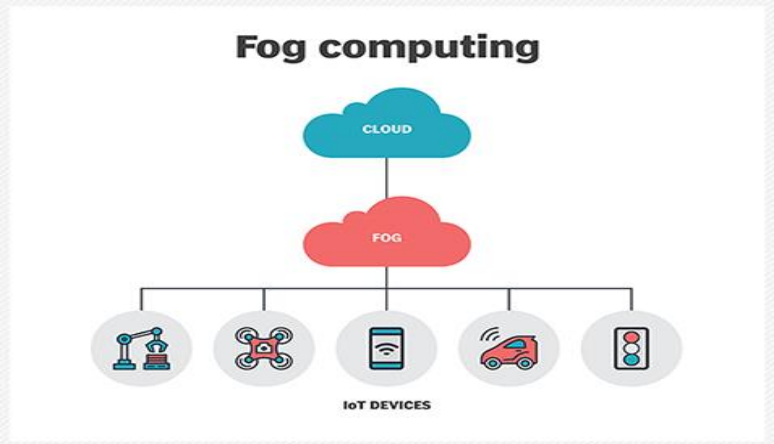

Figure 2: Fog Computing

Similar to cloud computing, fog computing however involves the decentralization of data processing of the data nearer or closer to the source of the data. As a result of this project, measures obeying the fog computing principles are incorporated into the architecture to improve the storage system performance and availability. Some of the prominent features that make the fog nodes are that they intermediate the users and the cloud by preprocessing the most accessed data and making the processing more efficient.

4.2 SHA Hashing Algorithm:

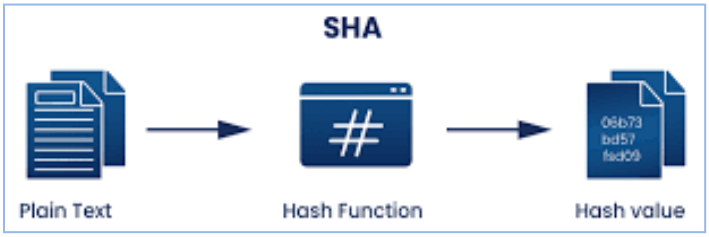

Figure 3: Hashing Algorithm

SHA (Secure Hash Algorithm) is a hashing algorithm, which is used to transform the information of variable length into a string of fixed length refer to as hash. It improves data accuracy as it comes with a hash value which is unique for a given input. Like in any other hash function, if there is a change in the data that is being passed into the hash function, the hash value changes completely and is therefore very useful in checking integrity of data and detecting data amendments. SHA is widely applied in securing a connection and encrypting messages and verifying their content as well.

5 Implementation

This section explains how to implement user and admin logins, manage cloud servers, upload download files. It also introduces the system's security policies and fault tolerance measures.

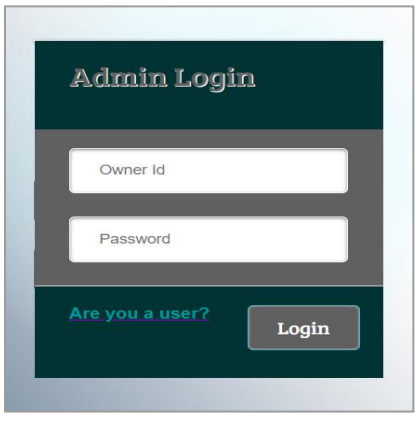

Figure 4: Login as a User and Admin

The Admin manages as in the Figure 5 the cloud server management, enabling authorized personnel to configure and monitor server infrastructure.

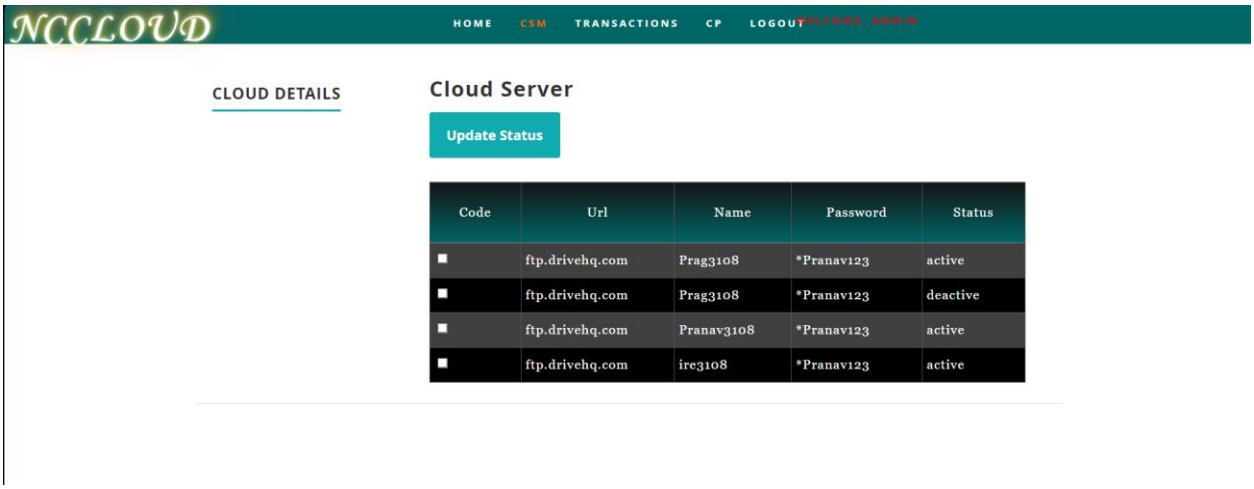

Figure 5: Cloud Server Management

As shown in Figure 6, the admin has secure access to manage and delete users. Any authorized person who can have their share of the cooperation, will also be capable to view information on which employees are connected to network from a different city or state. This user created user list can be used in a logical manner and some authorities would be able to see the members of other jurisdiction on duty. It provides for effective user management with strong security.

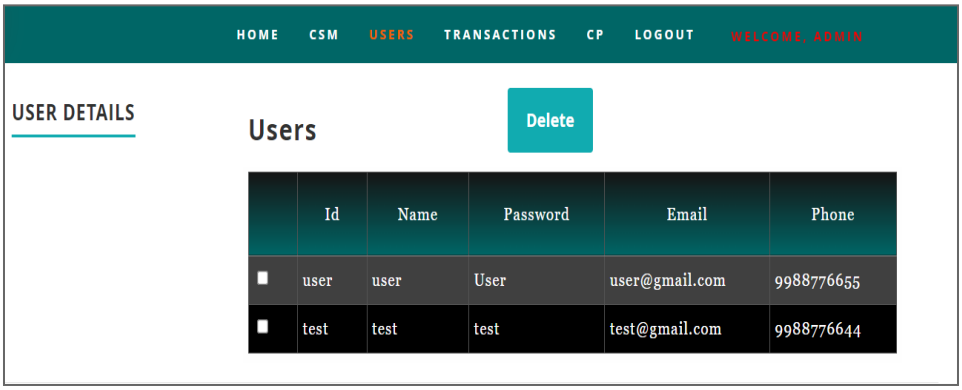

Figure 6: User Lists

File Transactions Management In Figure 7, as shown above the admin provides secure access for file transactions. When network is fast enough this person will have authority over other sources that need data files, High school principal hare been teaching Pathology Observe the user list again, says the author.

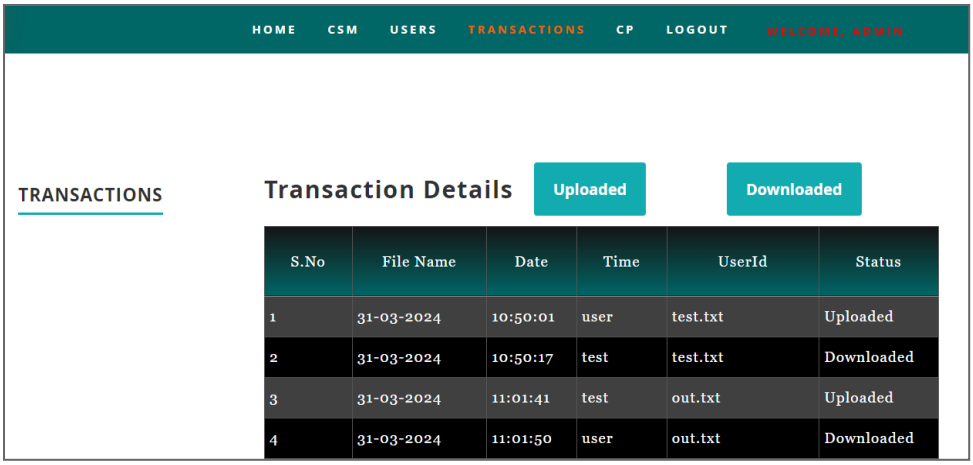

Figure 7: Transactional Details

In Figure 8. It contains details about users such as male or female information and account settings. Users can edit their profile as needed.

File Uploading

The file upload in this work ensures data security, privacy and availability. It strategically divides and places data blocks in the blurred and cloud storage environments. The system uses cryptographic processes and replacement codes to improve fault tolerance and minimize repair bandwidth. Carefully planning the current placement of data not only prevents data corruption but maintains user privacy throughout storage lifecycles. The following sequence of operations outlines how this is achieved during file uploads. In Figure 9 shows the file upload process as implemented here.

Steps involved in File Uploading

These are the steps for file upload, from selecting a file and transmitting it to server to making new block stores in fog and cloud environments.

1. **Seclusion**: The user chose the file to be uploaded.
2. **File Transfer:** The file was transmitted to the server.
3. **Division:** The server divided the file into four equal size blocks.
4. **Block Name**: The blocks were named A, B, C, and D.
5. **Generate MAC:** All four blocks were just made.
6. **MACs or HELLs:** These were stored in a table.
7. Additional new block creation? New blocks were created as follows.

- A ⊕ C
- B ⊕ D
- A ⊕ D
- (B ⊕ D) ⊕ C

8. Block Storage: The total eight blocks were stored as follows:

Table 1: Blocks Storage Details

N1	N2	N3	N4
A	C	A ⊕ C	A ⊕ D
B	D	B ⊕ D	(B ⊕ D) ⊕

Blocks A, B, C, and D were stored in fog storage (N1 and N2).

The remaining four regenerative blocks (A ⊕ C, B ⊕ D, A ⊕ D, and (B ⊕ D) ⊕ C) were stored in cloud storage (N3 and N4).Below Table 1 shows these details

Pseudo code for File uploads.

1. User has to select the file to be uploaded
2. Transfer the file to server
3. Server divided the file into four equal blocks
4. Name the blocks as A,B,C & D
5. Generate the MAC for all the four blocks
6. Store the MACs in table
7. Create another four blocks as below
    1. A (XOR) C
    2. B (XOR) D
    3. A (XOR) D
    4. (B (XOR) D) (XOR) C
8. Now there are eight blocks, store four blocks A,B,C,D blocks fog storage and remaining four regenerative blocks in cloud storage

Data Blocks Storage Process

For demonstration purposes, the DriveHQ platform was utilized to simulate the fog and cloud storage environments in our privacy preserving storage system. Two distinct directories were created within DriveHQ to represent the fog and cloud storage components. These directories were organized as follows:

Fog Storage Directory: This directory was designated to emulate the edge resources in a fog computing environment.

- **Folder N1**: One of the folders under the fog storage directory, designated to store specific blocks of the data.
- **Folder N2:** Yet another folder under the mist storage directory, set up in this case to hold even blocks of data.

Cloud Storage Directory: This directory has been designed, modeled on traditional cloud storage resources, as a way to Taoge.Deposit.

- **Folder N3**: A file under the cloud storage directory, used to store additional blocks of the data.

- **Folder N4:** Yet another folder under the cloud storage directory, used to store the remaining blocks of data.

Figure 10: File Stored in Cloud

The directory and folder structure in DriveHQ made a clear dam between fog and cloud data storeages. With this in place. we were able to clearly demonstrate the placement of data blocks strategically and the implementation of fault tolerance when it comes management systems for our proposed system actually works well.

File Download Process

As to this system for file download shown in figure 11, the problem of how secure and efficient file retrieval is tackled are the subjects of interest here. By resorting to a number of techniques such as hash code verification, fault tolerance mechanisms and an optimized transfer process which makes maximum use in both layer storage type (fog then cloud) data layer sensorial array controller controller (ASLC),a Secure Download plug-in can be produced The file download process in this system is described step by step below

1. Option of File to download: User has been to select the download file

2. Status of Fog Cloud Storage (N1 & N2) Check: System checks the status of fog cloud storage both N1 and N2.

3. Fog Cloud storage is Active: If both primary fog cloud storages (N1 and N2) are active then a. All four blocks were downloaded. b. The MAC(Message Authentication Code) was generated. c. Retrieve the MAC from the table where, d. MACs are then compared. e. The result will be displayed

If the result is "PASS," all blocks were merged to form the file. g. The file was downloaded to the local system.

Inactive Fog Cloud Storage: If either one of or both primary fog cloud storages were not active, then QRT (Quick Recovery Technique) process is carried on by system.

Process involved in File download.

1. User has to Select the file to be downloaded
2. Check the status of Fog Cloud storage (N1 & N2)
3. If it is active

    a. Download all four blocks

    b. Generate the MAC

    c. Retrieve the MAC from table

    d. Compare the MAC

    e. Display the Result

    f. If result is PASS then merge the blocks and form a File

    g. Download the File to the local system

4. If any one or both Primary cloud Storage is not active, then system has to perform QRT technique.
5. QTR Technique

4.Quick Retrieval Technique

Table 2: Quick Retrieval Table (QRT)

Cloud Status				Block Retrieval Process			
N1	**N2**	**N3**	**N4**	**A**	**B**	**C**	**D**
T	T	T	T	A	B	C	D
T	F	T	T	A	B	(A⊕C) ⊕ A	(B⊕D) ⊕ B
F	T	T	F	(A⊕C)⊕C	(B⊕D)⊕D	C	D
T	F	F	T	A	B	(B⊕D) ⊕ ((B⊕D)⊕C)	(A⊕D) ⊕ A

F	F	T	T	(A⊕C)⊕C	(B⊕D)⊕D	(B⊕D) ⊕ ((B⊕D)⊕C)	(A⊕D) ⊕ A

The Quick Retrieval Technique (QRT) is a beyond-most level the advanced data recovery process designed to enhance the data retrieval tenacity and performance in a distributed storage system. By using regeneration codes with an overlapping set of data blocks residing in both fog and cloud storages,QRT enables perfect file recovery even though certain nodes have failed. Essentially designed to maintain Data integrity, the technology also achieves a state of data retrieval that, in case of failure, not only minimizes downtime but is also operationally smoother overall and so reduces system vulnerability.

1. All Nodes Active (T TT T):

Status: All storage nodes (N1, N2, N3, N4) are active.

Retrieval: Blocks A, B, C, and D are directly retrieved from N1 and N2.

Process:

- A from N1
- B from N1
- C from N2
- D from N2

2. N2 Inactive (T F T T):

Status: N2 is inactive; N1, N3, and N4 are active.

Retrieval: Blocks C and D are regenerated using blocks from active nodes.

Process:

- A from N1
- B from N1
- C is regenerated using (A⊕C) ⊕ A from N3
- D is regenerated using (B⊕D) ⊕ B from N3

3. N1 Inactive, N4 Inactive (F T T F):

Status: N1 and N4 are inactive; N2 and N3 are active.

Retrieval: Blocks A and B are regenerated using blocks from active nodes.

Process:

- A is regenerated using (A⊕C) ⊕ C from N3
- B is regenerated using (B⊕D) ⊕ D from N3
- C from N2
- D from N2

4. N2 Inactive, N3 Inactive (T F F T):

Status: N2 and N3 are inactive; N1 and N4 are active.

Retrieval: Blocks C and D are regenerated using blocks from active nodes.
Process:

- A from N1
- B from N1
- C is regenerated using (B⊕D) ⊕ ((B⊕D)⊕C) from N4
- D is regenerated using (A⊕D) ⊕ A from N4

5. N1 Inactive, N2 Inactive (F F T T):

Status: N1 and N2 are inactive; N3 and N4 are active.
Retrieval: All blocks are regenerated using blocks from active nodes.
Process:

- A is regenerated using (A⊕C) ⊕ C from N3
- B is regenerated using (B⊕D) ⊕ D from N3
- C is regenerated using (B⊕D) ⊕ ((B⊕D)⊕C) from N3
- D is regenerated using (A⊕D) ⊕ A from N4

Administrators can now secure access to manage file transactions such as that shown in Figure 12. Authorized users can keep track of and influence file activity, for example ensuring that data is handled efficiently

Performance Analysis

Our failure tolerance is an intermediate mix ahead of traditional independent RAID configurations We have long random codes rapidly and frequently within the system since it is better than infrequent recodings that are only repaired once. RAID 0s performance but at the packet level without redundancy. It decreets nothing really. RAID 1 without protection leads to inefficency. Ours is a cloud system which guarantees the availibilty of your information even if multiple nodes on the cloud fail. Compared to RAID 5 and 6 which use the idea of mixing performance and redundancy with greater complexity our method gives users higher levels of both security and storage capacity. This improved fault tolerance gives a tougher answer to dilemmas of extreme cloud storage demand (Khasan, R. A. and et al, 2024).

Security Analysis

Unencrypted data blocks are stored in the Fog cloud for robust security, while public clouds store encrypted data that is XOR-coded. Therefore, sensitive information can still be protected. The level of encryption provided by the public cloud creates an additional barrier to unauthorized access; hacks in fact are that much harder. As well as this, the integrity of data is protected by hash techniques, which guarantee that information is authentic and ensure its durability. If troth come what may is preserved intact and beyond time's touch then all data stored on this medium data manager shall be at one with all mankind and as inviolable.

6 Discussion

Challenges in traditional cloud storage system The challenges associated with the existing Cloud Storage systems can be summarized as below: a) Security b) Latency c) Data Availability Questions of unauthorized access, data breach or even the scenario depicted in large-scale hacking movies where systems come under attack can arise from storing all aspects of our life solely in 'the cloud'. Furthermore, the cloud storage can have latency issues as well since data retrieval usually is carried out through remote data centers which will be quite far away. And when that cloud service has a downtime, or many of the nodes of your files get compromised you could have significant problems to access your own data.

In order to resolve these challenges, we proposed a Fog computing and cloud storage integrated system which provides security data integrity migration, high availability of the stored information and less latency. The fog nodes sit nearer to the end user and take care of simple unencrypted data blocks, which results in less dependence on far away cloud servers thereby decreasing latency. On the public cloud, meanwhile, data blocks XOR-coded there also provide encryption on a per-block level that makes it difficult to decrypt when accessed by unauthorized parties.

Our implementation offers up to improved fault tolerance, meaning the system can still recover data even if several nodes fail. However, the system's limitation is that if more than two cloud nodes fail, users cannot download the file. The combination of Fog computing and regenerating codes in our approach provides a balanced solution to cloud storage's inherent problems, enhancing both security and efficiency.

7 Conclusion

An effective and reliable system The proposed scheme was implemented successfully, leading to the validation of this efficient system with 100% success in executing all test cases listed Table II “Quick Retrieval Technique (QRT)” table. That is 50% fault tolerance already quite impressive especially added up with an extra storage space addressed. This demonstrates the ability of a system to ensure the integrity and security of data in difficult situations. Our solution overcomes this problem where most of the existing fault tolerance systems deal with balancing storage efficiency and fault tolerance at scale. This represents a significant step forward in the realm of fault-tolerant storage platforms.

Next, we will expand the system to work with images as well audio and video files. This growth will let us check out the system on various kinds of media, and see how robust or scalable it is. We will also take a deeper look at advanced encryption methods and optimize the placement algorithm

to improve fault tolerance, latency. The team will also look for larger conditions of the system in a variety real world cases to ensure that it works and is capable.

References

Ali, E.S.M. and Kareem, K.A., 2021. 'An efficient fog-assisted framework for wireless sensor networks by exploiting binary Edwards curves cryptography', *Webology*, 18(4).

Chen, J., Wu, W., Zhang, X. and Yang, F., 2015. 'Secure cloud storage based on secure network coding: A systematic approach', *IEEE Transactions on Information Forensics and Security*, 10(1), pp. 56-67.

DeepShah. "A Comparative Study on Cloud, Fog and Edge Computing." 2021 5th International Conference on Electrical, Electronics, Communication, Computer Technologies and Optimization Techniques (ICEECCOT). IEEE, 2021.

Fu, X., Yang, S. and Xiao, Z., 2020. 'Decoding and repair schemes for shift-XOR regenerating codes', *IEEE Transactions on Information Theory*, 66(12), pp. 7371-7386.

Ganesan, A., Alagappan, R., Arpaci-Dusseau, A.C. and Arpaci-Dusseau, R.H., 2017. 'Redundancy does not imply fault tolerance: Analysis of distributed storage reactions to single errors and corruptions', in *Proceedings of the 15th USENIX Conference on File and Storage Technologies (FAST '17)*, Santa Clara, CA, USA, 27 February – 2 March.

Ganesh, A., Sandhya, M. and Shankar, S., 2014. 'A study on fault tolerance methods in cloud computing', in *2014 IEEE International Advance Computing Conference (IACC)*, pp. 844-849. IEEE.

Gupta, I., et al., 2022. 'Secure data storage and sharing techniques for data protection in cloud environments: A systematic review, analysis, and future directions', *IEEE Access*, 10, pp. 712477-71277. doi: 10.1109/ACCESS.2022.3188110.

Jaya Singh, C.E. and Baburaj, E., 2019. 'XOR reformed Paillier encryption method with secure deduplication for image scaling and cropping in reduced cloud storage', *International Journal of Intelligent Engineering & Systems*, 12(4).

Koehler, S., Desamsetti, H., Ballamudi, V.K.R. and Dekkati, S., 2020. 'Real world applications of cloud computing: Architecture, reasons for using, and challenges', *Asia Pacific Journal of Energy and Environment*, 7(2), pp. 93-102.

Kumar, V., Laghari, A.A., Karim, S., Shakir, M. and Brohi, A.A., 2019. 'Comparison of fog computing & cloud computing', *International Journal of Mathematical Sciences and Computing*, 1, pp. 31-41.

Khasan, R.A. and Khomonenko, A., 2024. RAID: Data reliability and performance analysis. In E3S Web of Conferences (Vol. 549, p. 08023). EDP Sciences.

Mahdikhani, H., Lu, R., Shao, J. and Ghorbani, A., 2020. 'Using reduced paths to achieve efficient privacy-preserving range query in fog-based IoT', *IEEE Internet of Things Journal*, 8(6), pp. 4762-4774.

Mubarakali, A., Durai, A.D., Alshehri, M., Al-Farraj, O., Ramakrishnan, J. and Mavaluru, D., 2023. 'Fog-based delay-sensitive data transmission algorithm for data forwarding and storage in cloud environment for multimedia applications', *Big Data*, 11(2), pp. 128-136.

Pan Yang, Xiong, N. and Ren, J., 2020. 'Data security and privacy protection for cloud storage: A survey', *IEEE Access*, 8, pp. 131723-131740.

Rehman, A.U., Aguiar, R.L. and Barraca, J.P., 2022. 'Fault tolerance in the scope of cloud computing', *IEEE Access*, 10, pp. 63422.