Effective Access Sharing Mechanism for Data Stored in Cloud by Leveraging Permissioned Blockchain

MSc Research Project
Cloud Computing

Alby Sabu
Student ID: 21240906

School of Computing
National College of Ireland

Supervisor: Dr. Shivani Jaswal

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Alby Sabu
Student ID:	21240906
Programme:	Cloud Computing
Year:	2023
Module:	MSc Research Project
Supervisor:	Dr. Shivani Jaswal
Submission Due Date:	14/12/2023
Project Title:	Effective Access Sharing Mechanism for Data Stored in Cloud by Leveraging Permissioned Blockchain
Word Count:	8531
Page Count:	21

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:	Alby Sabu
Date:	13th December 2023

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):	

Effective Access Sharing Mechanism for Data Stored in Cloud by Leveraging Permissioned Blockchain

Alby Sabu
21240906

Abstract

The rise in data volume that is generated every day has influenced the end users to adopt cloud storage systems to meet their requirements. The user is concerned about the integrity and privacy of the data that is stored online even though the cloud service providers ensure maximum confidentiality, integrity, and availability. The users also have to provide access to the data to other data users in certain cases. To tackle these challenges, utilizing a blockchain-based solution can offer transparency and enable data access sharing without the need for a trusted third-party verifier. In this research project, a data access-sharing system is developed by utilizing the permissioned nature of Hyperledger Fabric that will create a network of verified participants and help to maintain trust with each other while sharing data. Hyperledger Fabric offers an added advantage in designing solutions for organizational use cases by providing higher scalability and privacy than using public blockchain systems. A test prototype of the system was developed for creating a data access sharing system between data owners and data users and evaluated to verify the research study. The evaluations based on the testing and benchmarking performed show that the system implemented offers a significant performance such as high throughput and less latency.

Keywords— Confidentiality, Hyperledger Fabric, Scalability, Access Sharing

1 Introduction

1.1 Background

The technological shift from on-premise solutions to cloud-based services has provided significant advantages to organizations, businesses, and end users in many aspects. This shift has led to the high volume of data that is stored online using the various services provided by cloud service providers. Storing, managing, and accessing data from anywhere at any time has been enabled by the cloud services. Along with these advantages, data security and privacy must be a top priority for cloud service providers. Improved and innovative techniques are utilized and are evolving day by day to maintain Confidentiality, Integrity, and Availability of the data and resources.

Securing data can be challenging but can be achieved by adopting stringent measures such as restricting access to personal data, encryption mechanisms, auditing integrity to check data manipulations, and reducing data breaches. Cloud service providers should be able to guarantee the safety of the data stored to be trusted by the end users. In this scenario, Blockchain technology raises its relevance as it can be also said as a record of trust. It serves as a public record book wherein every participant has the right to view records, record data and maintain the record book. The records are appended with a timestamp while writing to the blockchain and cannot be modified by an individual entity. Blockchain has replaced the characteristics of the traditional internet from a centralized system to a decentralized system by avoiding the need for a trusted third party. But if in case the trusted third party itself can act as a malicious entity, then it can lead to a huge data breach and loss of privacy. Blockchain solves the issues of data authentication as well as can handle the sharing of data Wei et al. (2020). Initially, blockchain technology was introduced as a system to create virtual currencies like Bitcoin which enabled a decentralized payment system using cryptographic functions that do not need a controlling authority like the way banks work with fiat currencies like cash.

Cloud storage systems can benefit from integrating blockchain-based techniques to create decentralized solutions that will enable users to perform data-sharing in a secure way without having a mediator. This will enhance the transparency and integrity of the cloud system as well as improve the trustability of users. Additionally, these systems offer a more reliable and secure means of storing data and sharing access to those data with other users by utilizing decentralized authentication and also reduce the chance of attacks by malicious participants. Enterprises from various sectors such as finance or healthcare can heavily benefit from utilizing the advantages of blockchain-powered cloud solutions to protect sensitive and confidential data since these systems can restrict unauthenticated access and avoid tampering of data Tarannum and Abidin (2023). Banking systems can take advantage of the increased auditability and security offered by blockchain as well can also utilize permission-based techniques to enable an efficient authentication scheme. Also storing and managing the identity data of people and enabling digital identities securely can be achieved by the Governments using blockchain-based systems. It also offers high network resilience by preventing unauthorized access Murthy et al. (2020).

Previous works and research in this area have tried to address these challenges and have tried to achieve the objectives of developing a system that utilizes blockchain technology to improve the security and efficiency of cloud-stored data. Along with that multiple subdomains in this area of integrating cloud with blockchain have also been explored in many of the past works. This includes Blockchain-as-a-Service(BaaS) offerings where users can deploy decentralized applications using the service provided, data provenance solutions that enable recording the activities related to data and its lifecycle, access control mechanisms, etc Gai et al. (2020). Each of the works proposed is relevant when considering this as a growing domain, but still has a wide scope for development. A common approach in many of the works was found to be using the smart contract functionality that provides the ability to write programmable functions that execute automatically in the blockchain nodes when certain conditions are fulfilled. Along with that, the Ethereum blockchain network is selected for deploying and testing those smart contract-based techniques. When comparing the traditional models, this Ethereum-based solution becomes significant by offering a decentralized system that eliminates the need for a third party to control the system. But while implementing solutions that can be used by authorized users or enterprise systems where data about the organizations should be private, the issue with public blockchain networks like Ethereum arises since anyone from the public can join the network and participate in the transactions. Along with that, the consensus algorithms that help these blockchain networks maintain the ledger with immutability are highly resource-intensive and also require transaction costs such as Gas fees in the Ethereum network for writing data to the blockchain and executing smart contracts. This will add additional overhead in terms of communication and computation costs in the systems based on public blockchain. This is where permissioned blockchain systems like Hyperledger Fabric have their advantage.

1.2 Research Question

How can the resources and data that are stored in the cloud storage service leverage the trust and immutability of a permissioned blockchain like Hyperledger Fabric to enhance security and transparency while sharing access with data users?.

In this research, the objective is to provide a solution that effectively solves the research question mentioned. Utilizing a permissioned blockchain like Hyperledger Fabric can offer effective mechanisms for improving the transparency of transactions that are part of organizational systems with multiple users. The main motivation for the project is to create a data access-sharing system that benefits data owners and data users by upholding trust and verifiability.

1.3 Ethics Consideration

The ethical consideration of this research project is represented in table 1.

Table 1: Human Participants Involvement

This project involves human participants	Yes/No
The project makes use of secondary dataset(s) created by the researcher	Yes/No
The project makes use of public secondary dataset(s)	Yes/No
The project makes use of non-public secondary dataset(s)	Yes/No
Approval letter from non-public secondary dataset(s) owner received	Yes/No

1.4 Report Structure

The research report is divided into the following sections. The related works that have been conducted in the research domain were reviewed and discussed in section 2. The methodology adopted for conducting the research activities is elaborated in section 3. Section 4 gives the overall design specification of the work. The implementation part is discussed in section 5 in detail. The evaluation and results obtained in this research are explained in section 6. Finally, the conclusion and the possible future works are discussed in section 7.

2 Related Work

In the process of understanding the evolution of existing works and studies in this proposed area of utilizing a blockchain-based approach to offer a secure data-sharing system for cloud storage, the author has reviewed several papers. An in-depth analysis of the papers was conducted to identify the advantages offered and the challenges faced by them. In the following section, some of the past works from specific areas that are related to the research domain are discussed in detail.

2.1 Blockchain-based integrity auditing of Cloud stored data

Preserving the integrity of the data that is uploaded by the users into a cloud storage or service is highly important. Cloud storage services offer the flexibility for the applications to handle and manage data remotely but they always have the risk of tampering the data. To verify the integrity and availability of the data, the data owners can perform auditing in the cloud storage. A request for integrity auditing is submitted to the cloud service provider by the data owner and the cloud service provider should respond with proof of how data is stored securely. The data owner can verify the proof and thereby validate whether the data is corrupt or not. In the past, many works discussed the techniques for auditing the integrity of cloud-stored data. Even though some have suggested the use of a third-party auditor as a trusted entity, it cannot be trusted fully.

The paper presented by Li et al. (2020) has suggested the use of Merkle Hash Tree as a solution for auditing data integrity. The authors propose the use of blockchain instead of using a third-party auditing scheme. In this proposed model, there are only two entities, the data owner and the cloud service provider. The file that is to be uploaded to the cloud will be initially divided into smaller blocks and each block is then encrypted by the data owner. Then a tag is generated by performing a hashing function on each block and then uses these hashtags to construct the Merkle hash tree. The root of the Merkle hash tree is stored locally by the data owner to quickly verify the integrity of the data. Along with that, these hashtags are broadcasted to other data owners who are in the network and also stored in the blockchain. The encrypted blocks are uploaded into the cloud. In the auditing phase, a request is sent by the data owner to another random data owner to calculate the root of the Merkle hash tree using the available hashtags that are stored in the blockchain. Also, the same challenge is sent to the cloud service provider which will also construct the Merkle hash tree. If the root returned by the cloud service provider matches the root calculated by the random data owner, then the data is said to be secure without any tampering. This method reduces overhead expenses like communication costs and increasing data security by eliminating the third party auditor.

Another work proposed by Liu et al. (2017) has put forward a framework for Data Integrity in the context of the Internet of Things applications and the data handled by them in a cloud environment. The authors have tried to solve the reliability issues of using third-party auditor-based frameworks for checking the integrity of data by making use of the decentralized characteristics of blockchain. Even though the cloud service providers offer maximum assurance to safeguard the CIA of user data, there is always a risk associated with it since the end users mostly have limited control of the data stored. The framework mainly has four elements including the user application(both data owner and consumer), the blockchain network, and the cloud storage service. In the system, a data integrity service(DIS) is developed using a smart contract and is deployed in the blockchain network. For the data owner or consumer applications to use the DIS, the blockchain client should be started in the nodes. The paper also discusses the data integrity verification protocols that are essential for the data owner and consumer in verifying the integrity of the data stored in the cloud storage service. This work has overcome the limitations of using cloud-based identity management systems as they are less flexible in handling IoT data. Additionally, since the blockchain network cannot be stopped by a single node the reliability has also increased. However, by using an Ethereum-based blockchain the transactions will require a 'Gas' fee for writing the data into the blockchain. This may cause other overhead expenses in implementing this model.

A data integrity verification framework for peer-to-peer cloud storage was put forward by Yue et al. (2018) using the Merkle tree. Every user in a peer-to-peer cloud can serve either as a user or storage provider. The framework consists of three main components including Clients, Cloud storage, and the Blockchain. The system flow is designed into two stages, preparation and verification. During the preparation phase, a five-step process is performed. Initially, the data will be divided into smaller chunks called shards and a Merkle hash tree is constructed using those shards. Both the client and the cloud storage service should agree on the constructed Merkle hash tree. The client will then store the root of the Merkle hash tree into the blockchain and the data and Merkle tree will be uploaded to the cloud storage. The cloud storage service will respond to the client with the address in which the data is stored. During the verification phase, the client will request the cloud storage with a challenge that uses a shard from the original data. Then the Cloud storage solves the challenge by computing a hash digest using the provided shard. this hash digest along with some additional details is sent to the blockchain which will execute the smart contract to calculate a new root for the Merkle hash tree. This can be compared with the root hash calculated during the preparation phase in order to ensure that data integrity remains valid or not. The verification result is thus sent to the client. This approach utilizes the immutable property of blockchain to store the root of the Merkle hash tree when each data is uploaded to the cloud and thus makes it impossible for users or the cloud storage to modify its value thereby ensuring trust and traceability. Since the system is decentralized, the system is also highly reliable. The proposed system uses the Ethereum blockchain for implementation and IPFS as the peer-to-peer cloud storage.

2.2 Using Blockchain as a data storage mechanism in cloud

In order to solve the complexities caused when the data is stored in blockchain-based cloud storage needs to be modified frequently, Pan et al. (2021) has proposed a flexible scheme for implementing a cloud storage system based on blockchain. The proposed system has multiple phases that include Setup, Chameleon Hash, Merkle Tree, and Modification. The chameleon hash function basically transforms inputs having any length to an output having a fixed length. It is computed utilizing an algorithm with Key generation, Hash computation and verification, and collision calculation. For tasks such as identity-based encryption and digital signature creations, bilinear pairing is used. Merkle tree can be used to verify data integrity. In order to avoid the issues that arise when computing the Merkle tree root hash each time the data is modified, this model adopts a pairing-based Chameleon hash technique in the Merkle tree creation. This will avoid the computation of root hash every time since it remains unaltered. The proposed scheme offers improved collision resistance and key exposure freeness thereby providing a privacy-based data-sharing method. Similarly, the paper by Sandeepkumar and Suresh (2023) discusses the need for a secure method for storing healthcare information in cloud data storage systems. The author raises the concern that there is a lack of interoperability and privacy issues in the existing systems. The challenge is to provide a system that helps to store different types of medical data in off-chain storage and using on-chain techniques for sharing access to those data. Context-based access control scheme is utilized to allow or deny access requests depending on parameters like the identity of the users, time stamp, and context of data request. The user will be required to provide the Temporal hash Signature in order to verify his identity and access privileges. This method has also utilized smart contracts to design the method and it is deployed in the Ethereum blockchain network. IPFS is used to store the data in the of-chain system.

2.3 Blockchain-based security mechanisms for cloud stored data

Utilizing the Advance Encryption Standard(AES) algorithm to design a decentralized cloud storage system to enhance the security of the data stored can be found in the work by Pise and Patil (2021). The user utilizes the cloud storage to store the data and then shares the private key with other users to whom access should be provided. A key manager is utilized to provide access to the user who requests to obtain the key. While uploading the data, initially it is divided into multiple chunks and then it is encrypted using the AES algorithm to transform the data into an unreadable format. Also by using the SHA-512 hashing algorithm, the data blocks are linked to each other. This makes it possible to identify in case of data modification or deletion by an attacker. Geetha et al. (2023) has also adopted a similar approach to improve the security of data stored using encryption techniques and make use of the Ethereum blockchain that enables the users to identify whether data has been compromised or not. Both the cloud service provider and blockchain are the main components of the system. Whenever the client user's data is stored in the cloud storage system, the cloud storage performs some computations to obtain a master hash value. This is written to the blockchain as a transaction and the block header is returned back which is needed for verifying the integrity of the data. To enable improved data confidentiality, the data is encrypted using the AES algorithm, and the HE-RSA algorithm is used to encrypt the secret key. The authors suggest that even though this method has improved security, it can be further developed using other possible blockchain methods instead of an Ethereum-based network as it can reduce the network communication costs and provide a cost-effective system.

2.4 Using Blockchain for enabling access control mechanism in cloud

Access control mechanisms are primarily responsible for preventing unauthorized and unrestricted access to the resources stored in the cloud. It enables cloud service providers to safeguard and secure the confidential and private data of organizations and individuals. The traditional techniques of relying on centralized schemes for access control in the cloud has always a chance of risk associated with it. This can be caused due to external malicious attacks or even some internal data breaches. The data owners always have a significant concern about the data stored in the cloud since they have limited control after storing the data.

A framework named 'AuthPrivacyChain' has been put forward by Yang et al. (2020) for protecting data privacy and implementing an access control system. The proposed system utilizes blockchain to provide a decentralized architecture and makes use of the tamper-proof characteristics of the blockchain to define and store the access control policies so that they cannot be altered. Here the wallet address associated with the participant nodes is taken into consideration as the identity of the node. The authors mainly suggest this framework as a solution to two main problems. Firstly, attacks from external malicious entities can cause the Access Control Policy Database(ACPD) to be altered which can cause potential vulnerabilities such as impersonation attacks, modification of permissions for unrestricted access, etc. Additionally, there are also chances that the system administrator is malicious which can also lead to potential threats. In this system, the cloud, data owner, and data user should be registered. For this, a smart contract is defined to initialize and add the verified nodes to the blockchain. After that when a data owner uploads the data into the cloud, the metadata of the resources are also written to the blockchain using a defined function. It contains parameters like the node address, index, content, and timestamp. For writing the access control permissions to the blockchain, a function named 'VerifyCap' is defined. When the user sends a request to access a particular resource, the cloud fetches the access permissions from the blockchain that are associated with the user address and decides whether access can be granted or not. Along with that, all these transactions are logged into the blockchain by the cloud. Additional functions are defined to authorize the other data users by the data owner and to revoke those permissions. The authors have used the EOS blockchain to implement the proposed solution. The authors stated that the performance of the solution depends on the blockchain network selected and can increase the performance by tuning the configuration of the nodes.

Another work on an access control system using the Ethereum blockchain is presented by Wang et al. (2019) and the proposed work aims to preserve the privacy and security of the data stored in the cloud. A ciphertext-policy attribute-based encryption (CP-ABE) scheme is utilized in this model to enable security and privacy. By using the smart contract, the data owners can store data as ciphertext in the blockchain network. Also, time periods can be defined by the data owner to decide how much time the resources should be made available. The proposed system comprises of four main participants including the cloud, blockchain, data owner, and data user. The role of the data owner is to create and deploy the smart contracts, set the access control policy, upload the files to the cloud, etc. For the system to start, the data owner is required to deploy the smart contract named 'StorageSC' to the Ethereum network. The address of the contract will be returned after the successful deployment of the smart contract. Whenever a data owner needs to upload a file to the cloud, first the file ID is passed through a hashing function and this value is stored in the blockchain. Then the file is encrypted using the ID by utilizing the AES algorithm. The encrypted file along with the ID and contract address is uploaded to the cloud. When the data user needs to access the file, a request should be made to the data owner. The data owner decides upon the timeframe in which the resource should be available and is written into the blockchain using the smart contract. The data user can only decrypt the file if there is a valid time period remaining and the access policy defined by the data owner is fulfilled. In this work, decentralization is ensured by providing a distributed mechanism for access control and by using blockchain to enable the transactions between the data owner and the data user. The method was also successful in reducing the cost associated with data sharing.

A permissioned blockchain-based system for access control of medical data stored in the cloud and its challenges are discussed in the paper by Xia et al. (2017). Since the participants in this system are verified there is no chance of unauthenticated nodes present in the network. The identity of the network participants is verified by the use of cryptographic keys. The system design proposed contains three layers: the user layer with multiple types of users, the system management layer that includes the issuer and verifier modules, the consensus nodes, and finally the storage layer. The users who want to join the authenticated by the issuer. The keys for membership issuing and verification are created by the issuer. It also shares the verification key with the verifier. When a user wants to join the network, the process starts by sending a request to the issuer. The issuer has the right to accept or reject the request. if the user is authenticated successfully, a membership issuing key is returned to the user. The membership key along with a tag is sent to the issuer for confirmation and after verification, it is sent back along with a few parameters. This parameter is used by the user to create the transaction keys. Similarly, the user requests for membership verification and is returned with the membership private key. Finally, the user sends the transaction public key and the verifier stores it privately in a database. After the user is verified, they can request to access a file or upload a file. For this, they need to use the membership and the transaction keys set up during the system initialization phase. Every time when data request or data access is made, it is recorded into blocks and broadcasted into the blockchain. This is done by the consensus node. The authors have tried to utilize blockchain to provide a scalable system through which privacy can be maintained while data-sharing.

3 Methodology

In this section, the methodology that was adopted in the process of attaining the research objectives is discussed. This research has tried to design and implement a solution for providing an effective system that can be used for sharing and providing access to data users for data that is stored in cloud storage by utilizing a permissioned blockchain like Hyperledger Fabric. By adopting this method, the privacy and traceability of the data can be ensured and the trust between the users of the system can be maintained.

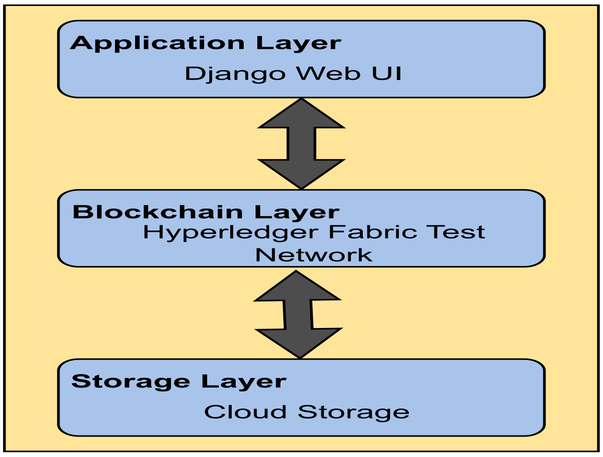


Figure 1: Three layered structure of the designed system

In order to demonstrate the underlying concept of the research, a prototype was developed that utilizes the Hyperledger Fabric test network, and the developed chaincode was deployed into the test network. Along with that, a web application that the end user can utilize for interacting with the test network was also developed using the Django web framework with Python. As represented in the figure 1, a three-layered structure was adopted in designing the specified system. As shown in figure 2, an iterative development approach was taken in the process of implementing the prototype of the system through a set of intermediate stages and then evaluating the system's performance by comparing it with the results of this system and existing systems. The main phases that this research has carried out are discussed below:

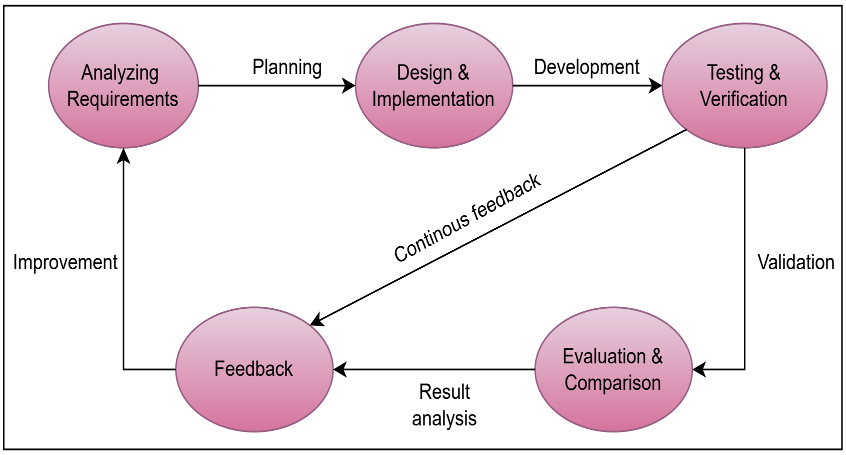


Figure 2: Phases of the research project

- **Requirement Analysis:** In this stage, initially the existing works that had tried to address the domain have been reviewed and their outcomes were evaluated. The possibility of using a permissioned blockchain, Hyperledger fabric-based approach that can outperform the existing techniques using the public blockchains was evaluated. The functional requirements for implementing the prototype with the Hyperledger fabric network and required technologies was finalized in this stage.

- **Design and Implementation:** The prototype development was carried out in this stage. Initially, the configurations for creating and setting up a Hyperledger Fabric test network were performed and the chaincode contract with the defined functions was deployed into the test network. Along with that, the web application through which the end users can interact was developed using Django framework and Python. The required backend APIs that enable the communication between the client application and the blockchain network were also developed.

- **Testing and Verification:** In this stage, the developed prototype was tested with the functionalities of the end user as Data owner as well as data user. The updates made to the Hyperledger fabric ledger were tested by querying the peers of the network. Additionally, the performance metrics of the Hyperledger fabric system such as throughput, latency, and resource utilization are measured through benchmark tests using Hyperledger Caliper.

- **Evaluate, Compare and Feedback:** The performance of the prototype developed was evaluated by reviewing the results obtained during the testing and verification stage, and it was compared with the performance of the existing techniques. Based on the results the feedback was utilized for improving the overall effectiveness of the system and thereby achieving the research objectives.

4 Design Specification

4.1 Components of the designed system

4.1.1 Data User

In the proposed system design, an end user can utilize the system in two ways. Either they can be a data owner who uploads the data or they can be a data requestor who wants to access the resources and data. Data owners can utilize the client application to upload their data and files into the system. The uploaded data will be stored in the cloud storage and can be any type of data that is supported by the cloud storage including images, audio, or documents. Whereas, the data requestor needs to request access to specific resources through the client application and the request will be available in the data owner application. The data owner can decide on whether the request should be accepted or not and only if it is accepted the data get available to the data requestor. Since the proposed system makes use of a permissioned blockchain-based approach, only the verified participants will be part of the network, and all the data operations performed such as creating new data, requesting access, and access approval will be logged into the blockchain for ensuring data integrity.

4.1.2 Cloud Storage Service

Cloud storage is an essential component in this proposed system and provides the necessary infrastructure for efficient storage of the data. The data that is being created by the data owners using the client application is uploaded into the cloud storage. Any suitable cloud storage services can be configured as per the requirements of the solution.

4.1.3 Blockchain

The proposed system design utilizes blockchain as a core component to improve data security and transparency. Blockchain can be referred to as a distributed system that enables the participants in the network to perform transactions in a decentralized method. The transactions that are performed will be written to a shared ledger that is available to all the participating entities using a consensus mechanism. Since the records are maintained by every node, no single entity can make modifications to the ledger thereby ensuring immutability. Data is considered as assets while defining a blockchain network that is developed for an organizational environment. These assets can be tangible as well as intangible and can represent anything that has value. To achieve the objectives of this research project, a permissioned blockchain service, Hyperledger Fabric is utilized in designing the system. Compared with the existing schemes that use public blockchain like Ethereum to offer data security and integrity, this scheme utilizes the permissioned nature of Hyperledger fabric to make the system available to only verified participants thereby adding an extra layer of authentication. Additionally, permissioned blockchains are very effective solutions in case of a network having transactions between known participants but they do not trust fully and also do not need a third-party verification system. In this case, since data is shared within the system with different users, the use of Hyperledger Fabric can increase the trust of users and maintain the traceability of data.

Hyperledger Fabric

Hyperledger Fabric is an open-source project created by the Linux Foundation that aims to provide a permissioned blockchain framework that can incorporate business use cases. It provides a modular architecture that is resilient, maintains the confidentiality of data, and is highly scalable. The fabric also provides the ability to deploy self-executing programs called Chaincode that resemble the smart contract functionality offered by blockchain networks like Ethereum. It allows the use of programming languages like go, java, node.js, etc for developing the business logic required.

The architecture of Hyperledger Fabric adopts an execute-order-validate structure for the transaction flow. In this three-stage process, the initial step starts by endorsing a particular transaction by executing it and verifying its correctness and validity. Following that ordering happens when the endorsed transactions are arranged and an agreement is reached between the nodes on what order the transactions are written to the blockchain. This agreement is achieved with the help of a consensus protocol such as Byzantine Fault-tolerant or Crash Fault-tolerant. The final stage is the validation phase where the transactions are validated according to the respective conditions required by the application. This eliminates invalid transactions and avoids the occurrence of race conditions that happen when multiple transactions are processed concurrently thereby maintaining the consistency of the ledger Androulaki et al. (2018).

Hyperledger Fabric utilizes the gRPC framework for enabling the communication between the underlying components such as peers, clients and orderers. The nodes in which the ledger and the chaincode is hosted and executed are called as Peers. The Membership Service Provider(MSP) authenticates the nodes and provides permission to be part of the network. The Certificate Authority of the MSP will provide an enrollment certificate and the transaction certificate to the peers for joining the network and submitting the transactions.

In Hyperledger Fabric, the transaction flow begins when a transaction proposal is made by the client application that contains the details of the proposed ledger update and the chaincode to be executed to the peers and then endorses it. The endorsing peers will then send a response to the client application proposal. In the next step, these endorsed transactions are submitted to the ordering service by the client application. The ordering service upon receiving the transactions from the clients will sort them into batches and then package it into blocks. These blocks are then distributed to the peers that are part of the channel created for validating the blocks and committing it to the blockchain. In the final stage, the peers will verify each of the transactions and validate whether it has received the required endorsement from the endorsing peers. Finally, these validated blocks are added to the blockchain, and the ledger state is updated.

4.2 System Architecture

The high-level architecture of the designed solution that utilizes Hyperledger Fabric to provide a decentralized system that enables the sharing of data stored in the cloud storage while ensuring privacy and transparency is represented in figure 3. The client application in the system enables the users to interact with the system and perform the required operations. Here, the users can be either data owners or data users according to the user's requirements such as uploading data or accessing an already existing data uploaded by other users. As this is a permissioned system that enables only verified participants, the users need to register in the system using the web application. Once they have logged

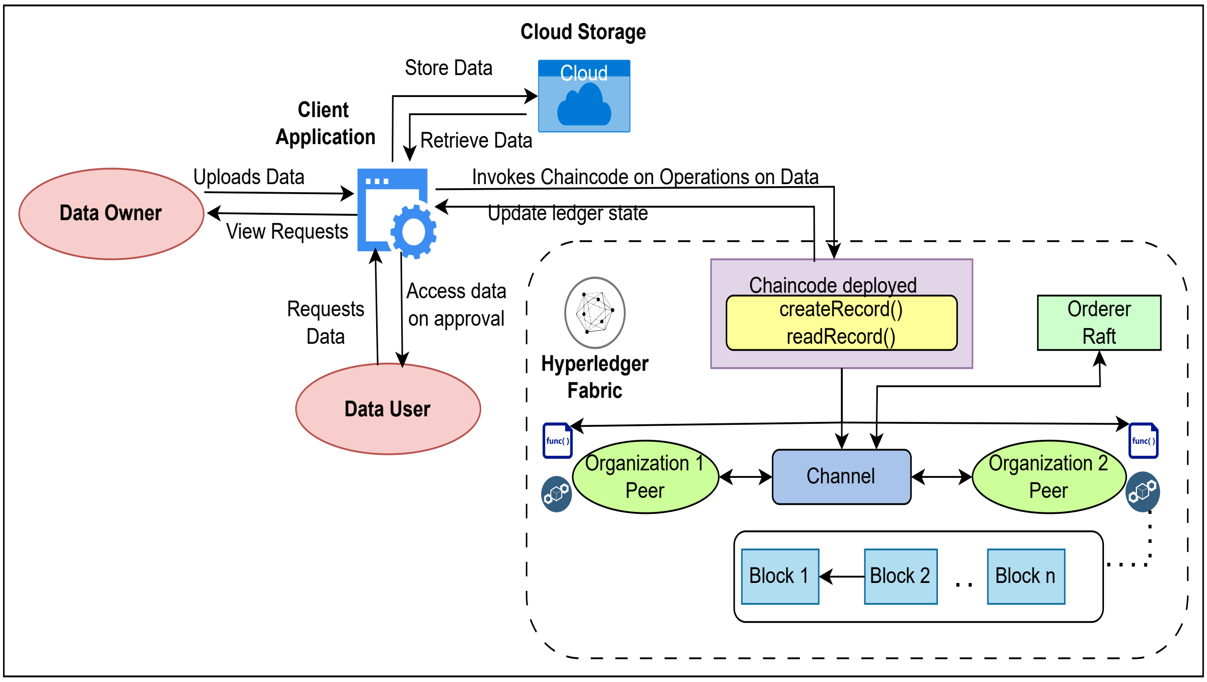


Figure 3: Hyperledger Fabric-based architecture of the system

in using the credentials, they can access the dashboard of the application which provides the feature to upload their own data which they wish to share access to other users or can see the files that are uploaded by other users in the network.

In the first case, if the user is the Data owner, he can upload the data files by specifying its related information like filename and description. On uploading the file to the cloud storage, the chaincode function is invoked using the backend API call which will create a transaction in the blockchain that indicates the creation of the asset along with the data owner's identity, the file identifier and timestamp. Along with that, the data owner can view the requests raised by other users who want to access the data that is uploaded by him. In the second case, if a user acts as a data user, then they can view the file details like name and description that are uploaded by other users. Here the data users have only the provision of viewing the file identifiers and not the actual data. In order to access a required file, the data user can make a request using the client application. This will also invoke the chaincode function and add an entry to the ledger with the requesting user's identity, the required file ID, the timestamp, and the identity of the original owner of the requested data. Whenever the ledger state is updated after a transaction is added to the blockchain it also updates the client application. The user will have to wait until the data owner approves the request to access the data. As mentioned earlier, the data owner application will fetch these user requests from the blockchain and display the list of requests made by other users who want access to that particular data. The data owner can decide whether he wants to approve or not for each of those users' requests. Whenever the data owner accepts a specific data user request, the chaincode function is invoked which will add a transaction to the ledger with the arguments that indicate the access provided by the data owner to the data user. The client application also receives the updated ledger state and the data user can successfully access and view the data that he has requested. Since each of the operations performed by the users are added as transactions to the Hyperledger Fabric blockchain, and cannot be modified or tampered by any of the users in the network, the traceability of the data can be guaranteed as well as provide a secure and reliable method for providing access to the data.

5 Implementation

In this research activity, the implementation of the prototype system with Hyperledger Fabric that will help users share access to cloud-stored data with other users while maintaining privacy and transparency was performed. It mainly consisted of two phases, setting up the Hyperledger Fabric test network and developing the client application.

5.1 Implementation of Hyperledger Fabric Blockchain Network

To utilize the advantages of a permissioned blockchain as mentioned in the research objective, the prototype developed uses Hyperledger Fabric to create a blockchain network in which the participants of this system can perform transactions. This project has utilized the Hyperledger Fabric version 2.5.4 test network for creating the network since this is a prototype for evaluating the study Fabric (2023). The complete environment was set up on an Ubuntu 20.04 operating system on a virtual machine. In this project, the Hyperledger fabric test network was configured with two organizations with one peer node each which will indicate the users in the network. Each peer node will have a copy of the ledger and the developed chaincode installed on them. An ordering node with a Raft consensus mechanism which will help in reaching an agreement on how the transactions are ordered is also configured. The fabric test network uses Docker Compose for deployment to containerize each node in the network. After the required prerequisites are met, the fabric network is started which will enable the components including peers and ordering nodes.

Once the network is up and all the peer nodes and ordering nodes are running a channel is created that will enable the two user organizations that were created to communicate with each other. In this prototype, only one channel is created for the project's purpose. The peer nodes of each user organization will be joining the channel. In fabric, the distributed ledger consists of two components, the world state as well as the blockchain. The world state uses a database for storing the current values of the parameters that represent the data assets in the network. In this project setup, LevelDB is utilized for configuring the world state and it holds the parameters as key-value pairs.

For developing the required chaincode that contains the required functions to interact with the blockchain Go programming language was utilized in this project. The users who are participants in the network can invoke the chaincode using the client application functionalities which will then record those transactions to the blockchain. Some key functions that are developed for this project are:

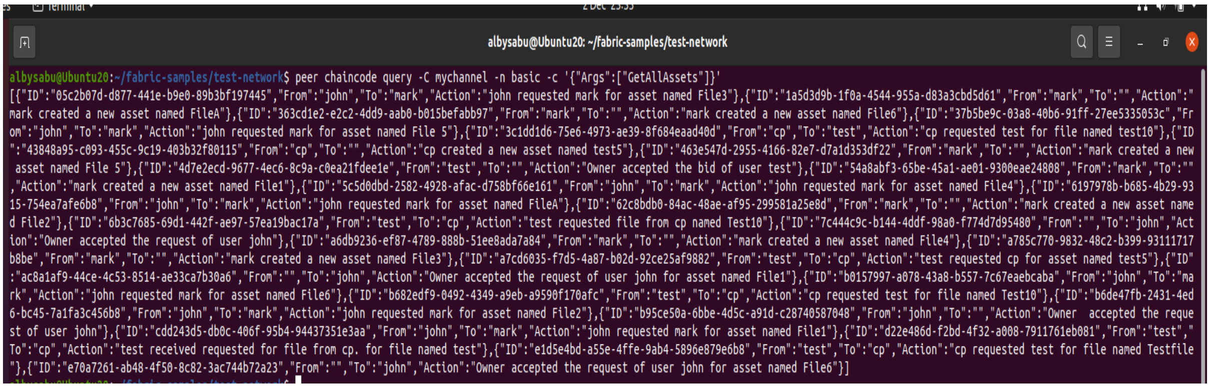
- When a data owner creates new data and uploads it to the cloud, create a transaction in the blockchain to represent asset creation

- When a data user requests access to specific data from a particular data owner, add the transaction to the blockchain for adding asset request log

- A function that queries the blockchain when the data owner needs to fetch the requests received for each data stored

- If the data owner approves the request of a data user, create a transaction in the blockchain to log that access was provided to that user for that particular data
- A function that queries the blockchain for reading permissions of data users to provide them access to the approved data

The developed chaincode was then deployed to the Hyperledger fabric test network that was configured during the initial setup. While deploying the chaincode smart contract, it will be initially installed in the peer nodes of the two user organizations that were created and then deployed to the channel that was created to enable communication between the two organizations. The process of deploying chaincode to the network involves processes such as packaging the chaincode by one or more member organizations in the network, installing the packaged chaincode on its peers by each of the organizations, approving the chaincode definition for the created channels endorsement policy to be satisfied and finally the chaincode definition is committed to the channel. After the deployment of the chaincode is done successfully, the chaincode will be started on the peer nodes which are part of the two user organizations created. Now, the users can interact with the Hyperledger fabric test network created using the web application. The transactions that are added to the blockchain can be queried also by using the peer CLI as shown in figure 4.



```
albysabu@Ubuntu20:~/fabric-samples/test-network$ peer chaincode query -C mychannel -n basic -c '{"Args":["GetAllAssets"]}'
[{"ID":"05c2b07d-d877-441e-b9e0-89b3bf197445","From":"john","To":"mark","Action":"john requested mark for asset named File3"},{"ID":"1a5d3d9b-1f0a-4544-955a-d83a3cbd5d61","From":"mark","To":"","Action":"
mark created a new asset named FileA"},{"ID":"363cd1e2-e2c2-4dd9-aab0-b015befabb97","From":"mark","To":"","Action":"mark created a new asset named File6"},{"ID":"37b5be9c-03a8-40b6-91ff-27ee5335053c","Fr
om":"john","To":"mark","Action":"john requested mark for asset named File 5"},{"ID":"3c1dd1d6-75e6-4973-ae39-8f684eaad40d","From":"cp","To":"test","Action":"cp requested test for file named test10"},{"ID
":"43848a95-c093-455c-9c19-403b32f80115","From":"cp","To":"","Action":"cp created a new asset named test5"},{"ID":"463e547d-2955-4166-82e7-d7a1d353df22","From":"mark","To":"","Action":"mark created a new
 asset named File 5"},{"ID":"4d7e2ecd-9677-4ec6-8c9a-c0ea21fdee1e","From":"test","To":"","Action":"Owner accepted the bid of user test"},{"ID":"54a8abf3-65be-45a1-ae01-9300eae24808","From":"mark","To":""
,"Action":"mark created a new asset named File1"},{"ID":"5c5d0dbd-2582-4928-afac-d758bf66e161","From":"john","To":"mark","Action":"john requested mark for asset named File4"},{"ID":"6197978b-b685-4b29-93
15-754ea7afe6b8","From":"john","To":"mark","Action":"john requested mark for asset named FileA"},{"ID":"62c8bdb0-84ac-48ae-af95-299581a25e8d","From":"mark","To":"","Action":"mark created a new asset name
d File2"},{"ID":"6b3c7685-69d1-442f-ae97-57ea19bac17a","From":"test","To":"cp","Action":"test requested file from cp named Test10"},{"ID":"7c444c9c-b144-4ddf-98a0-f774d7d95480","From":"","To":"john","Act
ion":"Owner accepted the request of user john"},{"ID":"a6db9236-ef87-4789-888b-51ee8ada7a84","From":"mark","To":"","Action":"mark created a new asset named File4"},{"ID":"a785c770-9832-48c2-b399-93111717
b8be","From":"mark","To":"","Action":"mark created a new asset named File3"},{"ID":"a7cd6035-f7d5-4a87-b02d-92ce25af9882","From":"test","To":"cp","Action":"test requested cp for asset named test5"},{"ID"
:"ac8a1af9-44ce-4c53-8514-ae33ca7b30a6","From":"","To":"john","Action":"Owner accepted the request of user john for asset named File1"},{"ID":"b0157997-a078-43a8-b557-7c67eaebcaba","From":"john","To":"ma
rk","Action":"john requested mark for asset named File6"},{"ID":"b682edf9-0492-4349-a9eb-a9590f170afc","From":"test","To":"cp","Action":"cp requested test for file named Test10"},{"ID":"b6de47fb-2431-4ed
6-bc45-7a1fa3c456b8","From":"john","To":"mark","Action":"john requested mark for asset named File2"},{"ID":"b95ce50a-6bbe-4d5c-a91d-c28740587048","From":"john","To":"","Action":"Owner  accepted the reque
st of user john"},{"ID":"cdd243d5-db0c-406f-95b4-94437351e3aa","From":"john","To":"mark","Action":"john requested mark for asset named File1"},{"ID":"d22e486d-f2bd-4f32-a008-7911761eb081","From":"test","
To":"cp","Action":"test received requested for file from cp. for file named test"},{"ID":"e1d5e4bd-a55e-4ffe-9ab4-5896e879e6b8","From":"test","To":"cp","Action":"cp requested test for file named Testfile
"},{"ID":"e70a7261-ab48-4f50-8c82-3ac744b72a23","From":"","To":"john","Action":"Owner accepted the request of user john for asset named File6"}]
```

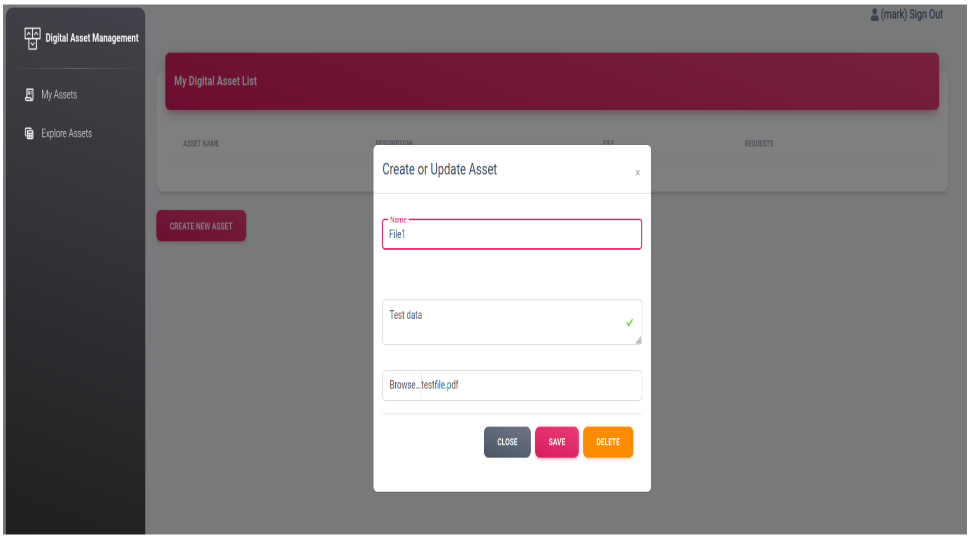
Figure 4: Querying ledger utilizing peer CLI

5.2 Implementation of Web Application

The users of the system need a client application to interact with the Hyperledger Fabric environment. For that, a web application user interface is developed in this project. The application frontend was developed using Django 4.1.7 framework and Python 3.8.10. The web application can be used by the Data Owners as well as the Data users according to the needs of the users. The stored data can only be accessed by the users with access permissions. The developed web application uses backend API calls to invoke the chaincode functions on user activities and then submits the transaction to the network to record the activities performed. The user flow through the web application is discussed below using the following cases:

- **User not logged in:** The users who are participants of the system can use their login credentials to log in to the system which will allow them to view the user dashboard. The dashboard is the same for the users since they can take the role of data owner or data user as per their requirements.

- **The user is a Data Owner:** As shown in figure 5, the data owners can create an asset by uploading a new data file that they need to store in the cloud and share with other users. They also can see the already uploaded data if any exist in the system and the requests raised by other users who need access to each of those files. The data owner can review the requests received for each file he owns and can accept the request of the data users with whom he decides to share access to that file as shown in figure 6.

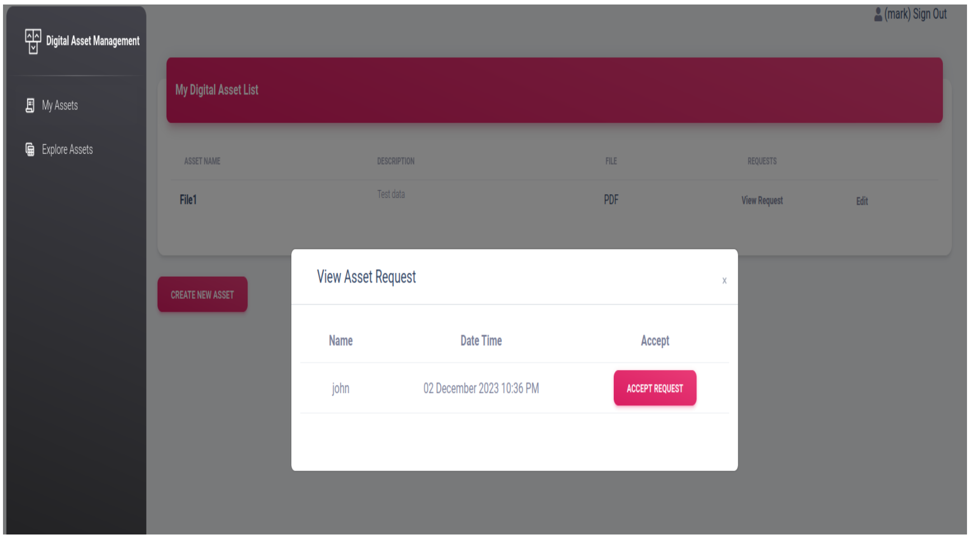


Name: File1

Test data

Browse... testfile.pdf

Figure 5: Data owner creating a new data asset

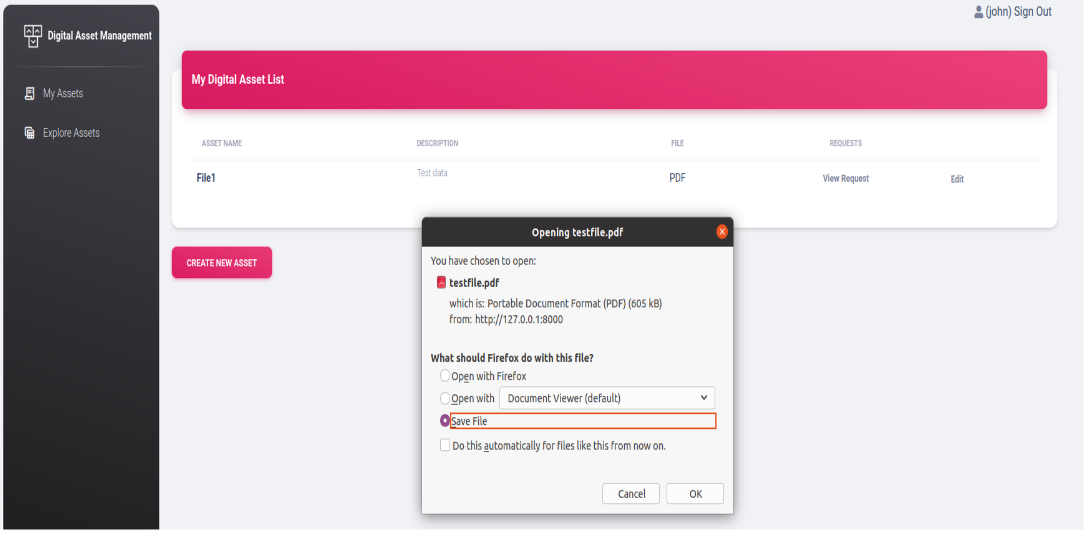


ASSET NAME	DESCRIPTION	FILE	REQUESTS	
File1	Test data	PDF	View Request	Edit

Name	Date Time	Accept
john	02 December 2023 10:36 PM	ACCEPT REQUEST

Figure 6: Data owner accepting request of a data user

- **The user is a Data User:** The data users can utilize the explore assets section of the application to view the data that is uploaded by other data owners. If they need access to download or view the data they should submit a request to the data owner of that file using the request option provided. Once the data owner has accepted the request, the access will be shared with the data user and can be viewed or downloaded as shown in figure 7.



ASSET NAME	DESCRIPTION	FILE	REQUESTS	
File1	Test data	PDF	View Request	Edit

Opening testfile.pdf

You have chosen to open:

testfile.pdf

which is: Portable Document Format (PDF) (605 kB)
from: http://127.0.0.1:8000

What should Firefox do with this file?

○ Open with Firefox

○ Open with Document Viewer (default)

● Save File

□ Do this automatically for files like this from now on.

Figure 7: Data user able to access file after data owner approves request

6 Evaluation

The system prototype developed during the implementation phase was tested and verified to ensure the effectiveness of the solution. During the testing, the user's functionality to interact with the Hyperledger Fabric network through the web application as well as the performance of the deployed chaincode in the Hyperledger Fabric network was verified by performing several test activities. The results obtained were evaluated to understand the advantages offered by the developed system in providing data access sharing with privacy and transparency. The performance evaluation is done by evaluating the results of the benchmarks performed.

6.1 Evaluating the data access sharing mechanism and data privacy

The developed system was tested with different cases when a user who is a participant in the system will utilize the functionalities as a Data Owner or Data User. The feature to upload data and create a new asset in the system as a data owner was successfully tested. Also, the data owner's ability to grant access to the requesting data users was found to be successful. Similarly, the data users' functionalities were tested to verify that the data users can request a particular asset and access it after receiving the access rights from the data owner. The privacy offered by the system is also verified by validating the condition when a data user without having access is not allowed to view or download the required data.

6.2 Evaluating the transparency offered by Hyperledger Fabric

Since the developed system uses the permissioned characteristics of Hyperledger Fabric, only the verified participants can perform transactions to the ledger. To provide transparency between the participants all of them have a shared copy of the ledger in which the complete history of the transactions performed are available. In our system, by utilizing the Raft consensus mechanism the operations performed on the data by the data owners and data users are written to the blockchain. by utilizing the immutability property of blockchain, it can be guaranteed that these recorded transactions cannot be altered to deleted by any single node. During the testing of the system, the transactions updated in the ledger were queried using the peer node CLI to ensure that the data operations were recorded and found to be successful.

6.3 Performance evaluation of the Hyperledger Fabric test network

The performance metrics of the configured Hyperledger Fabric test network and the deployed chaincode were tested using the Hyperledger Caliper benchmark tool Caliper (2023). The benchmarking tool evaluates the Hyperledger fabric network through custom use cases and provides information on the metrics which include Latency, Throughput, and Resource consumption. Caliper makes use of two configuration files and a defined workload module for running the benchmark and providing results. The network configurations of the configured Hyperledger fabric test network that include the information about the created network channel, deployed chaincode, and organizations are defined in a YAML file. The workload module consists of the files that contain the required functions to interact with the chaincode smart contract deployed to the network while running the benchmark. In this project, for testing the read as well as the write operations performance of the test network, two javascript files for read and write workloads are defined in the workloads module. The benchmark configuration file is also a YAML file that will define the information that includes the number of test workers, details on test rounds and workloads in each round, duration, and transaction load. It also includes parameters defined to measure the resource consumption by monitoring the node's utilization of CPU, Memory, network traffic, etc.

Table 2: Summary of the performance evaluation benchmarking

Operation Type	**Transaction Load**	**Send Rate (TPS)**	**Max Latency (s)**	**Min Latency (s)**	**Avg Latency (s)**	**Throughput (s)**
Read	5	95.7	0.35	0.01	0.04	95.5
	10	203.6	0.20	0.01	0.03	203.5
	25	236.2	0.31	0.01	0.05	236.0
Write	5	2.6	2.25	0.32	1.80	2.4
	10	9.4	2.23	0.18	0.78	9.2
	25	16.8	2.60	0.34	0.96	16.5

The benchmarking was performed for the Read as well as Write operations on the configured Hyperledger Fabric test network. The transaction load parameter is used to define the rate of transactions that is to be maintained constantly by the rate controller. Several tests were performed by varying the value of transaction load for both operations to evaluate the variations in the performance metrics. The overall summary of the results obtained from the test reports is provided in table 2. Transaction Latency can be defined as the time taken to get a response after a request was sent. The maximum latency observed for the Read operation was much less when compared to Write operations and showed slight variations for different transaction loads. Throughput can be defined the the amount of data that is sent successfully. While varying the transaction load, the throughput achieved showed a rise in the case of both Read and Write operations. It was also observed that the Read operation had a higher throughput value than the Write. Figure 8 shows the graph representing the variations in maximum latency and throughput on different transaction loads. The maximum CPU utilization of the nodes on varying transaction loads is represented in the figure 9.

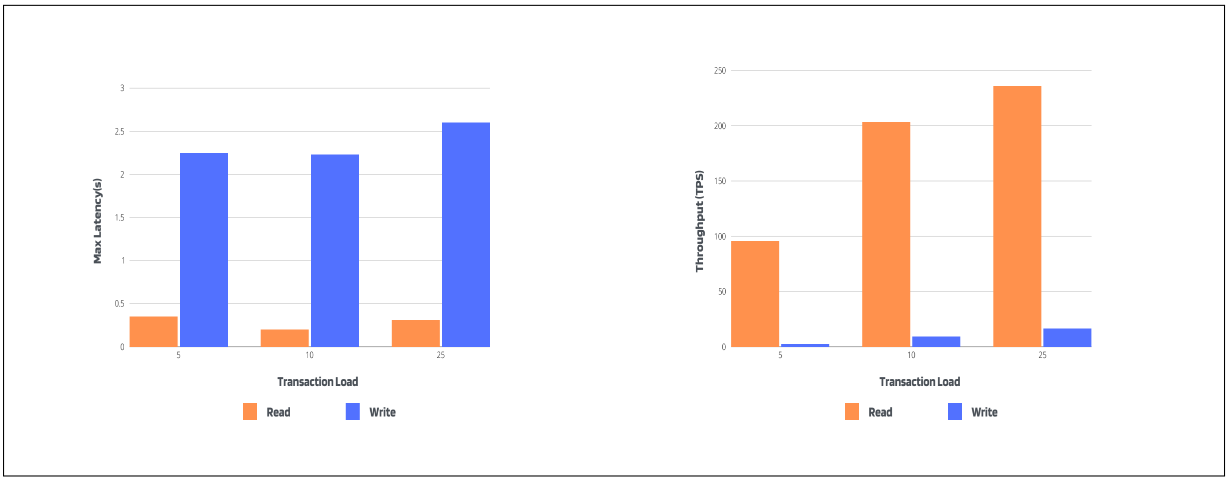


Figure 8: Maximum Latency & Throughput vs Transaction Load

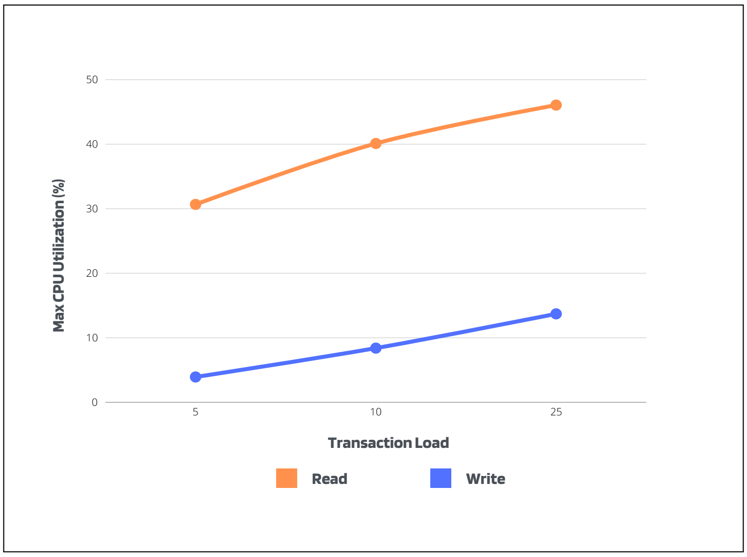


Figure 9: Maximum CPU Utilization vs Transaction Load

6.4 Comparison with existing Ethereum based systems

Many of the existing methods that aim to integrate blockchain to offer data integrity, privacy, and confidentiality for the cloud have utilized smart contracts deployed to Ethereum-based public blockchains. Some of them were also discussed in the related works section of this research. The main limitation of utilizing the public blockchain systems is having access to any unknown person in the public. This can be overcome in our work by utilizing the permissioned approach. Additionally, the transaction throughput in public blockchains like Ethereum is very low when compared with the throughput that Hyperledger fabric offers. Also, the consensus mechanism of Ethereum was very high resource-consuming. In the work by Jyothilakshmi et al. (2022), a detailed comparison was performed in analyzing Hyperledger fabric and Ethereum for real-world use cases and found out that Hyperledger Fabric outperforms Ethereum-based implementations by providing a highly scalable, confidential, reduced latency and offers maximum throughput.

6.5 Discussion

As discussed in the previous subsections, the prototype of the system that was developed to create a privacy and transparency-oriented data access-sharing mechanism through a permissioned blockchain approach was evaluated and found to be successful in achieving the objectives of the study. The system offered advantages to the data owners as well as data users to maintain trust without having to rely on a third-party system by implementing a decentralized network of participants. Additionally, the performance of this solution was found to be better than existing systems using public blockchains after considering the observations made during the benchmark tests. Since only the verified parties can become members of the network, this method can be greatly beneficial in creating organizational solutions. Also, unlike Ethereum which spends Ether as transaction charges to maintain the network, this solution does not require transaction fees which can significantly reduce the communication overheads.

7 Conclusion and Future Work

Preserving the confidentiality, integrity, and availability of the data that is stored online is a challenging task and requires the designing and development of efficient solutions. The primary goal of this research was to utilize the permissioned nature of the Hyperledger Fabric blockchain to develop a system that will facilitate data access sharing effectively while upholding privacy and trust. The research project has undergone many activities during the phases from understanding the existing works, implementing the solution, testing and verifying, and then evaluating the outcomes. The prototype for demonstrating the study was created using the fabric test network and by developing a client application through which the users interact with the network. By utilizing the system the Data owners can specifically provide access to the data they store in the cloud to those data users who are requesting that data effectively and by maintaining a transparent system. The benchmarking performed proved that the system offers increased throughput and reduced latency thereby guaranteeing scalability and usability for enterprise use cases.

As mentioned, the prototype developed used a test environment of the Hyperledger Fabric which can be implemented into a production environment by making some necessary configurational changes. Also, the system that was developed focussed on a generic data access sharing technique while it can be further extended to real-life application-specific systems such as KYC data sharing by customers in the banking industry or user certificate sharing to organizations for verification purposes. The system can also make use of private data collection for communicating sensitive information through the channel created between the network participant nodes. As the current system utilizes peer CLI to query the blockchain to see the transaction records, the feature for data owners to verify the logs of the data operations performed can be integrated into the web application to improve the user experience and easy verification. Additionally, tools like Hyperledger Fabric Explorer can be integrated into the system to monitor the network's high-level structure and associated metrics.

References

Androulaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., De Caro, A., Enyeart, D., Ferris, C., Laventman, G., Manevich, Y. et al. (2018). Hyperledger fabric: a distributed operating system for permissioned blockchains, *Proceedings of the thirteenth EuroSys conference*, pp. 1–15.

Caliper, H. (2023). Hyperledger caliper - getting started.
URL: *https://hyperledger.github.io/caliper/v0.5.0/getting-started/*

Fabric, H. (2023). Hyperledger fabric documentation - getting started. Accessed: November 13, 2023.
URL: *https://hyperledger-fabric.readthedocs.io/en/release-2.2/getting$_s$tarted.html*

Gai, K., Guo, J., Zhu, L. and Yu, S. (2020). Blockchain meets cloud computing: A survey, *IEEE Communications Surveys Tutorials* **22**(3): 2009–2030.

Geetha, S., Naveenkumaran, R., Selvaraju, K., Kishore, C. and Rathish, A. N. (2023). Blockchain based mechanism for cloud security, *2023 International Conference on Sustainable Computing and Data Communication Systems (ICSCDS)*, IEEE, pp. 1287–1295.

Jyothilakshmi, K., Robins, V. and Mahesh, A. (2022). A comparative analysis between hyperledger fabric and ethereum in medical sector: A systematic review, *Sustainable Communication Networks and Application: Proceedings of ICSCN 2021* pp. 67–86.

Li, J., Wu, J., Jiang, G. and Srikanthan, T. (2020). Blockchain-based public auditing for big data in cloud storage, *Information Processing & Management* **57**(6): 102382.

Liu, B., Yu, X. L., Chen, S., Xu, X. and Zhu, L. (2017). Blockchain based data integrity service framework for iot data, *2017 IEEE International Conference on Web Services (ICWS)*, IEEE, pp. 468–475.

Murthy, C. V. B., Shri, M. L., Kadry, S. and Lim, S. (2020). Blockchain based cloud computing: Architecture and research challenges, *IEEE access* **8**: 205190–205205.

Pan, Y.-y., Li, Y., Gao, C.-y., Fang, L. and Chen, P. (2021). Flexible and efficient blockchain-based cloud storage, *2021 IEEE 14th International Conference on Cloud Computing (CLOUD)*, IEEE, pp. 304–312.

Pise, R. and Patil, S. (2021). Enhancing security of data in cloud storage using decentralized blockchain, *2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)*, IEEE, pp. 161–167.

Sandeepkumar, E. and Suresh, A. (2023). Blockchain assisted cloud storage for electronic health records, *2023 International Conference on Computer Communication and Informatics (ICCCI)*, IEEE, pp. 1–6.

Tarannum, W. and Abidin, S. (2023). Integration of blockchain and cloud computing: A review, *2023 10th International Conference on Computing for Sustainable Global Development (INDIACom)*, pp. 1623–1628.

Wang, S., Wang, X. and Zhang, Y. (2019). A secure cloud storage framework with access control based on blockchain, *IEEE Access* **7**: 112713–112725.

Wei, P., Wang, D., Zhao, Y., Tyagi, S. K. S. and Kumar, N. (2020). Blockchain data-based cloud data integrity protection mechanism, *Future Generation Computer Systems* **102**: 902–911.

Xia, Q., Sifah, E. B., Smahi, A., Amofa, S. and Zhang, X. (2017). Bbds: Blockchain-based data sharing for electronic medical records in cloud environments, *Information* **8**(2): 44.

Yang, C., Tan, L., Shi, N., Xu, B., Cao, Y. and Yu, K. (2020). Authprivacychain: A blockchain-based access control framework with privacy protection in cloud, *IEEE Access* **8**: 70604–70615.

Yue, D., Li, R., Zhang, Y., Tian, W. and Peng, C. (2018). Blockchain based data integrity verification in p2p cloud storage, *2018 IEEE 24th international conference on parallel and distributed systems (ICPADS)*, IEEE, pp. 561–568.