Student's Declaration

I hereby declare that the work presented in the report entitled **"Blockchain Based Healthcare system using Offchain storage"** submitted by me for the partial fulfillment of the requirements for the degree of *Bachelor of Technology* in *Computer Science & Engineering* at Indian Institute of Technology, Jammu, is an authentic record of my work carried out under guidance of **Dr. Yamuna Prasad Shukla**. Due acknowledgements have been given in the report to all material used. This work has not been submitted anywhere else for the reward of any other degree.

Aman Pawar 2016UCS0016 **Place & Date: Jammu 07-06-2020**
Rahul Nirania 2016UCS0015

Certificate

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Dr. Yamuna Prasad Shukla **Place & Date: Jammu 07-06-2020**

Abstract

Today when we go to any hospital for any treatment we get a big bundle of our medical records like prescription, xrays, blood reports, bills etc. Its hard keep it safe for long and the bigger problem is how to carry all these when we go the hospital once again or to another hospital. Also if we keep it in a centralised location there are many problems like to find a proper method to share this data to any doctors or insurance company etc. What if any hospital keep our data? It seems a good option but what if we want to go to any other hospital for treatment or if its server go down and we are in an emergency. We can think of that we take the copy of it and go to another hospital but its a hectic job. What if all the hospital can share the date through any method? But the security and privacy is a big issue. So we need a good option for that which is secure and decentralised.

Keywords: ..(e.g.Healthcare system, Blockchain, Security, Decentralisation, Hyperledger Composer, Hyperledger Fabric, offchain, IPFS)...

Acknowledgments

I have taken efforts in this project. However, it would not have been possible without the kind support and help of our mentor Dr. Yamuna Prasad Shukla and Dr. Chaggan lal. I would like to extend my sincere thanks to both of them.
I am highly indebted to both of them for their guidance and constant supervision as well as for providing necessary information regarding the project and also for their support in completing the project.
I would like to express my gratitude towards my parents and member of IIT Jammu for their kind co-operation and encouragement which help me in completion of this project. My thanks and appreciations also go to my colleague in developing the project and people who have willingly helped me out with their abilities.

Work Distribution

Most of the part in our project is theoretical. The basic of this project i.e. Blockchain is read by both. The conceptual in depth part of Hyperledger Fabric and composer is done by Rahul Nirania and Implementation part (mostly) is done by Aman Pawar. For the off-chain part two out of three options i.e. IPFS and StorJ are read by Rahul Nirania and the Swarm part is read by Aman Pawar.

Contents

Introduction

HEALTHCARE is a data-intensive discipline in which large- scale data is generated, disseminated, stored, and accessed daily. In 2017, 16.5 million patients globally exploited remote health monitoring (a 41 percent growth from 2016), and this panorama has the potential to reach 50.2 million by 2021 . Also, since January 1st, 2018, the U.S. Centers for Medicare and Medicaid Services1 developed new reimbursement incentives to promote the adoption of "active feedback loop" devices to provide real-time monitoring. Data created when a patient is monitored or undergoes some tests, need to be stored in order to be accessible at a later time by a healthcare provider within the same or even a different network or context. As this realm expands, concerns about secure and efficient transmission of the medical data increase. Despite these concerns, polls show that 90 percent of Americans still value online access to their health records. It is easy to perceive that technology can contribute to enhancing the quality of caregiving for patients at a reduced cost. For example, in 2014, 15 percent of U.S. patients who visited a healthcare provider reported having to bring their medical test results to their appointment, and 5 percent required to have a procedure or a test replicated due to the lack of access to a prior test result. The technology of blockchain attracted considerable attention due to the possibility of recording all financial transactions in a secure and verifiable decentralized (peer-to-peer) fashion, without the rule from a third party to process transactions, which are then combined into blocks where each block contains a timestamp and is linked to its precedent. Once recorded, data cannot be altered, and the transactions history is combined into a chain structure without the possibility of additional branches of alternative transactions emerging or wedging into the middle of a chain

Chapter 1

Healthcare Data of Patients:

Healthcare is a data-intensive domain with a considerable volume of data is generated daily from monitoring patients, managing clinical research, producing medical records, and processing medical insurance claims. This industry is ever growing and will keep on growing for ever. So to manage the large amounts of data generated from the healthcare systems need to monitored correctly or it may impose some very serious problems.

Present Day Scenario and related Problems:

Currently most of the medical data of patients resides on a hard copy in India especially the images(like X-Ray, Sonography etc) and reports of different test.All these reports of the patients need to stored safely by the patient for their record as their medical history to present whenever needed by involved hospital as any new medication for any disease needs to be given on the basis of the previous medical history. Some good hospitals manage the data on their systems but it is centralised on one of the servers which cannot be shared directly to other hospitals at any time. For this patient has to take a copy of his records from the hospital and then present it to the next hospital who would then save it on their systems. There are two major flaws to this system. Firstly, even if we want that some hospital that previously had our data should not have access anymore then it is almost impossible for the patient to do so. Secondly, if in some unfortunate sequence of events someone needs to be immediately admitted to the hospital then they have to get their data from previous hospital and then give it to the current hospital which in their current situation would be very difficult and cumbersome. Other problems that are prevalent are security and server issues concerns of centralised data storage which may leak to either unwanted leakage of data or complete or partial loss of it and also this cannot be tamper proof as the hospital has complete access over the data without no metadata of how frequently the data was changed.

Using Blockchain in Healthcare systems

Since we wnat to create a more secure file sharing platform with a clear audit trail to improve the trustworthiness of the data, as well as of authorship protection, we intended to use Blockchain as the residing technology. While the focus of applications of blockchain in practice has been to build distributed ledgers involving virtual tokens, the impetus of this emerging technology has now extended to the medical domain. With the increased popularity, it is crucial to study how this technology accompanied with a system for smart contracts can support and challenge the healthcare domain for all interrelated actors (patients, physicians, insurance companies, regulators) and involved assets (e.g. patients' data, physician's data, medical history etc.)
The introduction of Blockchain into the Healthcare system solves most of our problems we mentioned above. Primarily the blockchain data is immutable which makes it tamper proof and so it can also be used as an evidence. Also for every change any permissioned member make sit gets recorded on the blockchain and so the the person making change can never back off from the fact that the data entered is not made by him whcih solves the security concerns. Now for the problem of access control, every member of the blockchain has full access to allow or revoke access from any other member at any time. Lastly since it is decentralised system, same data is stored on multiple peers which makes its almost completely fault tolerant.

Chapter 2

Blockchain

A blockchain is a technology of distributed datasets stored in blocks. Together these blocks form a record in which the blocks are connected and secured against mutation by using cryptographic procedures. The users are creating transactions for writing data on the blockchain .These transactions get saved in blocks generated by miners utilizing a consensus finding algorithm in case of permissionless blockchain and using smart contracts in case of permissioned blockchain . If the same block is generated multiple times with different content, various branches of the chain are resulting. In this case, the longest chain is valid because there is the most effort done for calculating this chain.
Blockchains are a combination of different computing and economics concepts,predominantly including peer-to-peer networks, asymmetric cryptography, consensus protocols, decentralized storage, decentralized computing and smart contracts, and incentive mechanisms. Blockchains introduce unique properties including immutability and transparency of cryptographically-secured and peer-recorded transactions, which have been prescribed by network consensus. As such, the potential associated with blockchain to fundamentally trans-form how organizations produce and capture value is huge and very real.

Types of Blockchain:

A **Permissionless Blockchain**(e.g., Ethereum, Bitcoin) is open to the public, and every transaction is to be validated by every or majority of participants. In this type of blockchain a block is usually added into the pre-existing chain only after the miners have verified the transaction using the consensus algorithm.
While only the authenticated users can join a **Permissioned Blockchain** (e.g., Hyperledger Fabric), validation in this type of blockchain is performed by only pre-selected nodes. Thus, it usually achieves higher performance than public blockchains. This usually does not include any miners and the transaction are added by the verification from various types of peer in the blockchain. Along with this the identity of every member is known so it makes the blockchain fraud tolerant.

Choosing the right Blockchain for us:

In current times all the industry who are moving to blockchain are choosing Permissioned Blockchain over Permissionless Blockchain. This is primarily because they want full control over the system that has access to join the blockchain and what member is allowed to access what data and also what type of peer function a member should be allowed to perform depending on their work profile in the organisation. Also Permissioned Blockchian do not require any type of mining, avoiding any extra cost incurred in case of Permissionless Blockchain due to mining. Moreover Permissioned Blockchain besides being economical, it is also highly efficient giving higher performance output in comparison to the latter. Most famous Permissioned Blockchain is currently the **Hyperledger Fabric** created by the Linux Foundation. Besides having many other features one of the main points of choosing it over other is that it is open source.

Hyperledger Fabric:

The Hyperledger Fabric (www.hyperledger.org) is part of a larger project known as the Hyperledger. Hyperledger Fabric is a permissioned blockchain technology unlike permissionless blockchains such as Ethereum and Bitcoin. Hyperledger Fabric is an open source enterprise-grade permissioned distributed ledger technology (DLT) platform, designed for use in enterprise contexts, that delivers some key differentiating capabilities over other popular distributed ledger or blockchain platforms. It is a private blockchain technology in that members of the blockchain network are known to each other and members are permissioned to join. On permissionless blockchains, membership is open to the public and any one can join and perform transactions on the network. Hyperledger Fabric is a modular blockchain framework which acts as a foundation for developing blockchain-based products, solutions, and applications using plug-and-play components that are aimed for use within a private enterprise environment.It has predominating features of highly modular and configurable architecture, enabling innovation, versatility and optimization for a broad range of industry use cases including banking, finance, insurance, healthcare, human resources, supply chain and even digital music delivery.

Hyperledger Fabric in Detail:

Broad Overview:

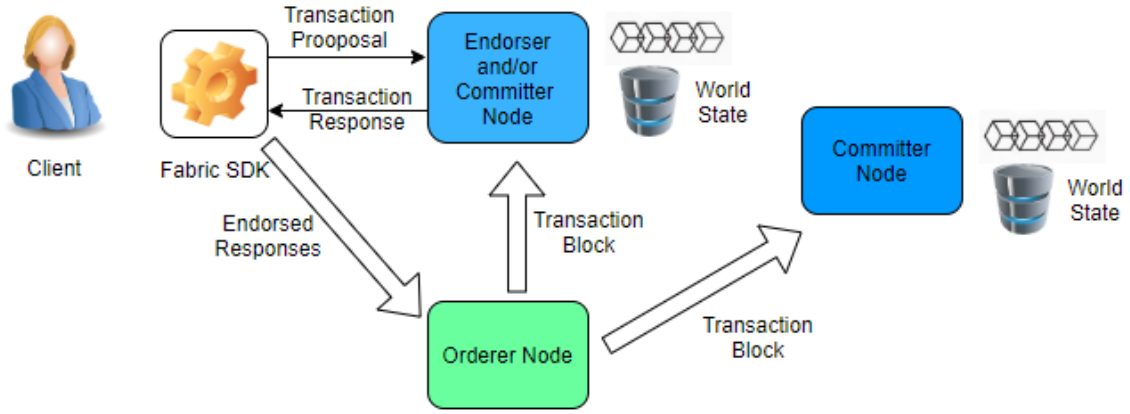

Figure 2.1: HyperLedger Fabric Architecture

In the image above when clients submit the transaction proposal through the Fabric SDK, this proposal is sent to all Endorsing Peers. These endorsing peers check the transaction verifies and executes and generate the Read and Write set as output. Now, this response is again sent to the client. The client collects all responses from all endorsing peers, and send them to Orderer. Now, Orderer sees all transactions and orders them in ascending order and form a block. Now, this block is sent to all committers which checks the transaction and add a new block in their own copy of the ledger.

Components of Fabric:

Membership Service Provider: The membership service provider (MSP), is a component that defines the rules in which, identities are validated, authenticated, and allowed access to a network. The MSP manages user IDs and authenticates clients who want to join the network. This includes providing credentials for these clients to propose transactions. The MSP makes use of a Certificate Authority, which may be a pluggable interface that verifies and revokes user certificates upon confirmed identity. The default interface used for the MSP is that the Fabric-CA API. However, organizations can implement an External Certificate Authority of their choice. As a result, one Hyperledger Fabric network are often controlled by multiple MSPs, where each organization brings its own favorite.
There are two types of MSPs.
Local MSP: It defines users(Clients) and nodes(peers, orderers). It defines who has administrative or participatory rights at that level.
Channel MSP: It defines administrative and participatory rights at the channel level.

ClientThese are the applications which work on behalf of any person who wants to access the network and do any transaction. The client uses Fabric SDK to communicate with the network. He can read or write the data to the network with the help of SDK. Every client is issued with the certificate of his identity by the CA i.e. Certification authority so that each

transaction must be executed by a valid client.

PeerA node that commits transactions and maintains the state and a copy of the ledger. A peer receives ordered state updates in the form of blocks from the ordering service and maintains the state and the ledger. Besides, peers can have a special endorser role. The special function of an endorsing peer occurs with respect to a particular chaincode and consists in endorsing a transaction before it is committed. Their types are Endorsing, Commiting, Anchor and Leading.
Endorsing Peer: Endorsing peers is a special type of committing peers who have an additional role to endorse a transaction. They endorse the transaction request which comes from the client. Each endorsing peer possesses the copy of smart contract installed and a ledger. The main function of Endorser is to simulate the transaction. It is executed based on the smart contract on the personal copy of the ledger, and generates the Read/Write sets which are sent to Client. Though during simulation, the transaction is not committed to the ledger.
Commiting Peer: Peers who commit the block which is received from the Ordering service, in their own copy of the blockchain. This block contains the list of transactions where committing peer to validate each transaction and mark it as either valid or invalid and commits to the block. All transaction either valid or invalid are all committed to blockchain for future audit purpose.
Anchor Peer: As Fabric network can extend across multiple organization, we need some peers to have communication across an organization. Not all peers can do this, but these are special peers who are only authorized to do so which are nothing but Anchor peer. The anchor peers are defined in Channel configuration.
Leading peer: Leader peers are those who communicate or disseminate messages from Ordering service to other peers in the same organization. These peers use Gossip protocol to make sure that every peer receives the message. Leading peers cannot communicate across an organization. If any Leading peer is not responding or is out of network, then we can select a leading peer from available peer based on voting or randomly choose one.

Ordererhe order of transactions has to be established to ensure that the updates to the world state are valid when they are committed to the network. In a Blockchain network, transactions have to be written to the shared ledger in a consistent order. Hyperledger Fabric allows the organizations running the network to choose the ordering mechanism that best suits that network. This modularity and flexibility make Hyperledger Fabric incredibly advantageous for enterprise applications.
Hyperledger Fabric provides three ordering mechanisms: SOLO, Kafka, and Simplified Byzantine Fault Tolerance (SBFT).

Channels A fabric network can have multiple channels. Channels allow organizations to utilize the same network while maintaining separation between multiple blockchains. Only members(peers) of the channels are allowed to see the transaction created by any member in a channel. In other words, channels partition the network in order to allow transaction visibility for stakeholders only. Only the members of the channel are involved in consensus, while other members of the network do not see the transactions on the channel. The peer can maintain multiple ledgers. And peer can be connected to multiple channels.The configuration of the channel is maintained by configtx.yaml file. Using this file we generate channel.tx file and then create a channel using it. Chaincode is installed on all participating peers in a channel, where as chaincode is instantiated on a channel. A channel contains all the configurations of communication between peers. It holds the list of peers along with who are endorsing, anchor, leader peers. When a client communicates with the network using SDK, the SDK first gets a list of

all endorsing peers to which the transaction request needs to send. Using this list the SDK sends transaction requests to peers. Peers that do participate in multiple channels simulate and commit transactions to different ledgers. Orderers are also a part of channels.

IdentityEach actors in a network peer, orderer, client, admin have some digital identity in the form of certificate X.509. This identity is used to verify at each and every step of a transaction, in order to check if the source of the transaction is from a valid source. In addition to the multitude of endorsement, validity, and versioning checks that take place, there are also ongoing identity verification happening during each step of the transaction flow.

Policies A policy is a function which accepts as input a set of signed data and evaluates successfully or returns an error because some aspect of the signed data did not satisfy the policy. Policies reside in the Channel configuration, but in some cases, it resides in chaincode too.More concretely, policies test whether the signer or signers of some data meet some condition required for those signatures to be considered 'valid'.

Ledger It is a current state of the business as a journal of transaction. A ledger consists of two different parts, a world state, and a blockchain. A ledger is kept at all peers and, optionally, at a subset of orderers. In the context of an orderer, we refer to the Ledger as to OrdererLedger, whereas in the context of a peer we refer to the ledger as to PeerLedger. PeerLedger differs from the OrdererLedger in that peers locally maintain a bitmask that tells apart valid transactions from invalid ones.The two different part of Ledger are World State(It is a DB that holds the current state of the ledger state.) and Blockchain(A transaction log that records all the changes that have resulted in the current world state).

Chaincode It is a package that is deployed to the blockchain network or a specific channel which consists of smart contracts, endorsement policy and other useful information.A smart contract defines the transaction logic that controls the lifecycle of a business object contained in the world state. Multiple smart contracts can be defined within a single chaincode. Smart contract is a domain specific program which relates to specific business processes, whereas a chaincode is a technical container of a group of related smart contracts for installation and instantiation. Every chaincode has endorsement policy attached to it, which applies to every smart contract defined within it. This identifies which organization must sign a transaction generated by Smart contract, in order to consider it valid.

Chapter 3

Off-Chain Storage

An off-chain storage solution unburdens the blockchain from part of the load of saving large datasets onto the chain. The challenge is to swap out the data but retain as many ad- vantages, like immutability and traceability, of the blockchain as possible. In the following section four approaches for off-chain storage solutions are presented. While on-chain storage is neither financially nor techni- cally practical, the advantages that the blockchain provide can be applied to off-chain storage methods. As a general example, the hash of a piece of data, which is quite small, can be stored in the chain. Due to the relatively small size of a hash, the corresponding cost for storage will also be low. The challenge, then, is to provide a link between the hash in the chain and the physical storage location. To that end, the two methods considered are smart contracts and distributed hash tables (DHT). The two are dif- ferentiated, because a smart contract defines what, who, and how, while a DHT more specifically handles how data is distributed and accessed.
Speaking with respect to our motive of implementing it with the Healthcare system, there is large amount of data that needs to be stored for a patient. Storing Large amounts of data on the Blockchain ledger will make it slow as every peer will receive large amount of same data. Not only this, the medical data is not all text information which can directly be stored onto the blockchain. It also includes X-ray images, Sonography files, lab reports, MRI, CT-Scan and many other similar files which are impossible to store on the blockchain. For storing these type of files, we firstly need to store them on some external database and then mention the corresponding address of the data in the database, onto the ledger so that anyone who wants access to the data can go to the database and access the files. As we already mentioned in our problems that a centralised database can create many problem including loss of complete or partial data of patients which would be problematic so for these reasons we wish to use a decentralised data storage. Decentralisation of data will not only prevent loss of data it will also improve access times as most of the current decentralised databases store same data on multiple peers which helps reduce access times fro database in comparison to a centralised database.

Types of Off-Chain Databases:

For the case of decentralised database we are presented with a variety of them namely,

- IPFS(Inter Planetary File System)
- StorJ
- Swarm
- Filecoin
- Siacoin
- etc

Each one of the above having its pros and cons. Below we will be discussing some of these which suit our need the most and then ultimately choose one of them for our Use case scenario for Healthcare Systems.

Swarm:

Swarm is a distributed storage platform and content distribution service, a native base layer service of the ethereum web 3 stack. It stores the data redundantly and distributed over multiple nodes. This approach allows Swarm to be DDoS-resistant, fault-tolerant, and censorship-resistant during large-scale operation. Swarm is primarily designed to store the Ethereum's public records but also provides the opportunity to store other files. For motivating the peers to offer their resources, an incentive system is integrated which can even penalize a peer for losing the hosted data of another party.Store and distribute Dapp code and data as well as provides blockchain data Immutability, Hash based content storage addressed through URL. Swarm is also said to be BitTorrent on Steroids.It provides schemes that make storers individually accountable for particular content. Incentives for Long term storage of various content.This storage system is quite less Popular and content can get deleted as there is no storage insurance currently, also it does not provide easy editing capabilities.

StorJ:

StorJ is a network of distributed and encrypted data storage. A file gets split up into different so-called shards, and the shards get encrypted on the client site for uploading to different storage nodes. As the identifier for the file, the hash value of the original file is used and supplemented with ordering information of the blocks. As data storage personal computer or servers can be used and the shards are stored redundantly on multiple machines. The storage owner gets an incentive payment after checking the avail- ability and integrity of the data from time to time. Splitting, hashing and encrypting has to be done by the publisher of the file. This is computation intensive for the client. The data stored on this is secured by Encryption using Owner's Private key, mapped using Hash table. It provides benefits of Encryption, Redundancy solution using parity shards, Erasure coding rules for too much redundancy, Regular Audits and verification in the network to ensure that no data is lost. One of its problem id that it was not developed for File sharing.

IPFS- Inter Planetary File System:

The Interplanetary File System (IPFS) is a peer-to-peer hyper media protocol designed to make the internet faster, safer, and more open. In a P2P network such as IPFS, if one node is down, other nodes in the network can serve needed files. Files are content addressed rather than location addressed on IPFS. When a file is added to the network, IPFS creates a multi-hash address of the file based on its content and the node ID ensuring that no two files share the same multi-hash. The node ID is the cryptographic hash of the node's public key. To allow other nodes to access the file, the multi-hash can be published on the network or shared among nodes. IPFS looks for files through the Content Identifier (CID) instead of searching by the location of the file. This means that any node with the hash of the content can search for the specific file on the network and any node that has a matching file will serve it. As a result, content on the network can be served so long as there is a node that can serve it. The P2P network is therefore robust and can withstand server attacks. Since content can be downloaded from the network, nodes can access local content offline.

IPFS eliminates content duplication by ensuring that files with the same content are stored only once. This is in respect to files uploaded by a specific node. For instance, if node A adds a file *myfile.txt* that contains "Hello world", IPFS creates only one instance of this file on the network even if the user of node A adds *myfile.txt* multiple times.IPFS can be used to share files on a private network or on the internet through the IPFS gateway. Most users of IPFS use it to share large files since IPFS uses local hosting which reduces the need for bandwidth which is usually associated with sharing large files on the internet. Others have started using IPFS to host websites. When a file is uploaded to IPFS, all peers with the hash address can access the content, download and view content locally or they can view content from the author's node.The greatest advantage of using IPFS in this way is that nodes become individual servers that can serve the content to others.

The main disadvantages for this include security and access control concerns which means anyone having access to the file's hash can access the file with any control of the uploading peer and also no information of the peer accessing the file Secondly the authorship protection means a person can get a file from IPFS and change it a bit and upload with a different name causing plagiarism and no information of the culprit is recorded on the system and the original author will never be able to prove its authorship and originality.

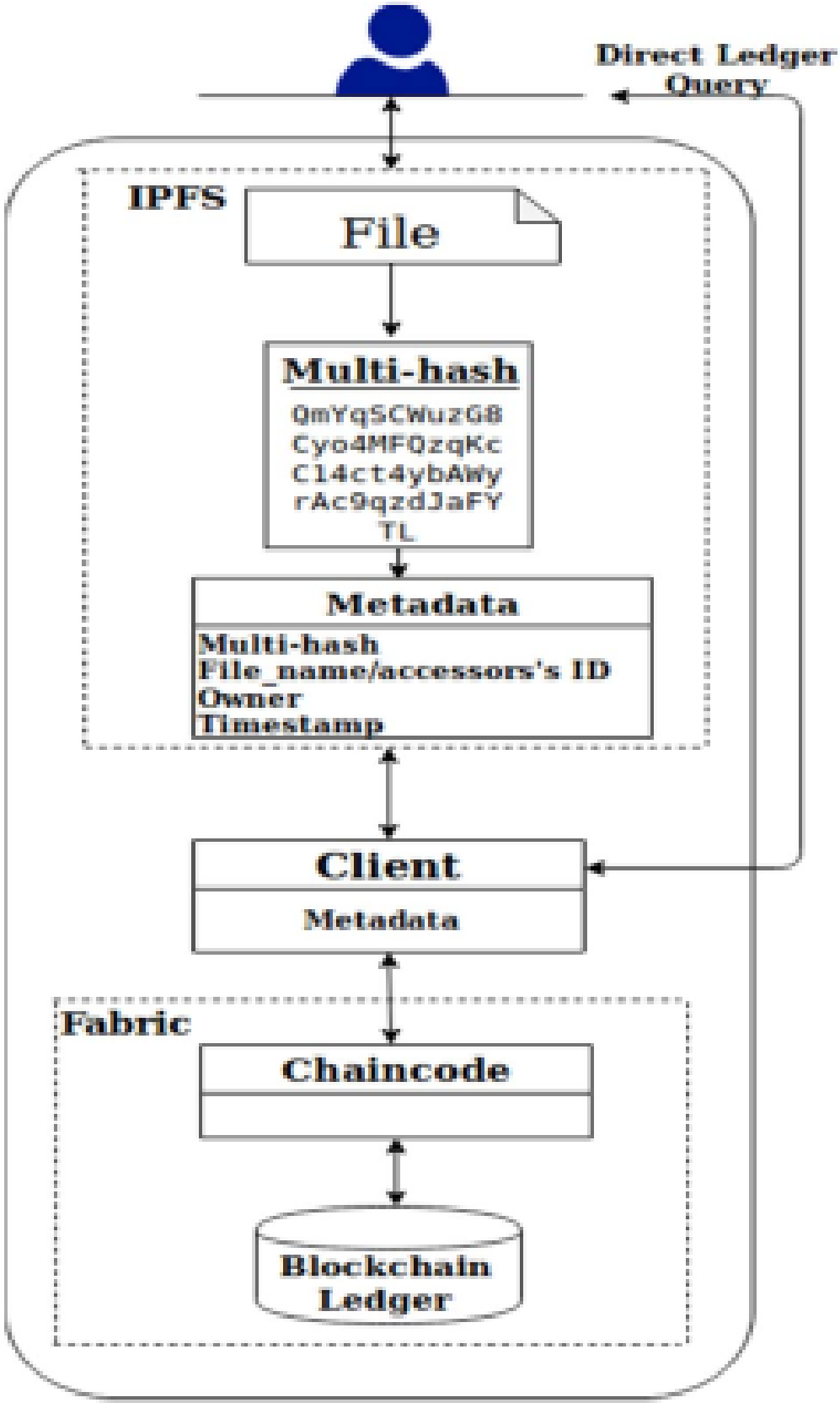

Figure 3.1: Fabric-IPFS Architecture

Using IPFS with Hyperledger Fabric:

Of all the above mentioned Decentralised databases, IPFS is least disadvantageous. Since all its disadvantages can be easily overcome by using Hyperledger Fabric in conjunction with IPFS. Since the hash of stored files will only be shared between the member of the blockchain whom the access to the data has already been given. Also since for every access and upload of file there will be a transaction submitted so anyone who access the file or changes it will have their identity and changes recorded onto the ledger, preventing any plagiarism and unsecured access. Te IPFS will work underneath the Hyperledger Fabric so every access to the IPFS can be made only through the Blockchain by following all its accesss control rules. Following are the specific reasons why Hyperledger Fabric is suited for solving the stated problems with IPFS in our case:
Performance: This is probably the most important feature of Hyperledger Fabric that makes it suitable for healthcare data in compared to open blockchains. As Fabric is permissioned, all nodes that participate in the network have an identity, as provided by a modular membership provider (MSP). Therefore, no mining process can be both resource and time-consuming in the transaction flow. This is critical to achieving minimal performance overhead when combining IPFS with blockchain.
Dynamic Channel Management: Hyperledger Fabric introduces the concept of channels, which can be dynamically managed within a single blockchain network. Participants in the same channel share the same set of ledger data, which cannot be accessed from outside of this channel. Therefore, by using Fabric underneath, it allows IPFS users to flexibly form groups in form of Hospitals and securely share files within a group.
Modularity and Extendability: One of the main characteristics of Hyperledger Fabric that differentiates it from other blockchain technologies such as Ethereum, R3 Corda, is its modularity. Fabric intends to provide a modular and extendable architecture that can be employed in various environments. The modularity of Hyperledger Fabric makes it suitable to be integrated with other technologies such as IPFS and enables us to integrate the needed modules with IPFS to extend the capabilities of IPFS.
Plug and Play API: Hyperledger Fabric provides SDKs that can be used to interact with the blockchain. The Fabric has SDKs in Node.js and Go. This capability enables IPFS to interact efficiently with Fabric. Through the capabilities provided by the SDK, we can efficiently develop a client to query the blockchain and provide the response or audit data needed.
Node Heterogeneity : Within Hyperledger Fabric, nodes are differentiated based on whether they are clients, peers, orderers, or certificate authorities . While IFPS also consists of multiple nodes, there is no strict mapping between the nodes in Hyperledger Fabric and that in IPFS.

Chapter 4

Use Case Implentation

The way we have created our Healthcare is mainly as a Proof of Concept to show that IPFS can be used as an Off-Chain storage solution in conjunction with Hyperledger Fabric. Our Healthcare system primarily consists of three participants which include *Patient*, *Doctor*,*Insurer*.There are avrious attributes of each of them nut the two most important with respect to access controls and data managed are *Medical Records* and *Insurance Details*. For the access control mechanisms we have created attributes named *authorised* both for doctors and insurers separately. The *cto* file contains the model or the structure of particiapnts, assets and transact For the implementation of the Healthcare System onto the Hyperledger Fabric Blockchain we need to begin by installing the Hyperledger Composer.
Hyperledger Composer is a set of tools that allows a way to create blockchain applications and smart contracts aimed at solving business problems and/or improving operational efficiencies. Hyperledger Composer simplifies application development on top of the Hyperledger Fabric blockchain infrastructure.It is written in javascript. It uses the installed docker images in the background for creating creating peers according to the user specification of the use case.As making each of the component separately is a very hectic so the tool "Hyperledger Composer" was developed which incredibly reduces the effort to make and deploy the blockchain network. It reduces the work of months to mere some weeks.

4.1 Hyperledger Composer installation:

For new setup, it is better to always set the locale and do apt update/upgrade :

```
$sudo dpkg-reconfigure locales
$sudo apt-get update && sudo apt-get upgrade
```

Setup a new user, because installing docker images in the administrator or *sudo* may create problems as it will interfere :

```
$sudo adduser blockckain
$sudo usermod -aG sudo blockchain
```

Switch to the newly created user :

```
$su - blockchain
```

Now we need to firstly install the composer tools for its use with the Hyperledger Fabric in the background. Firstly we need to install essential CLI(command line interface) tools for composer:

```
$npm install -g composer-cli
```

:Now we need to install utility for running a REST Server on your machine to expose your business networks as RESTful APIs:

```
$npm install -g composer-rest-server
```

After this we need useful utility for generating application assets:

```
$npm install -g generator-hyperledger-composer
```

Also install Yeoman. It is a tool for generating applications, which utilises generator-hyperledger-composer:

```
$npm install -g yo
```

We can also install composer playground locally on our machine as gives a better viewing experience and demonstrating our business networks with a great web app UI. So now we install the composer browser app for simple editing and testing Business Networks:

```
$npm install -g composer-playground
```

Now comes the most important part of installing the **Hyperledger Fabric** on which the Hyperledger composer will run. Firstly we will make its directory and then download the scripts:

```
$mkdir ~/fabric-dev-servers && cd ~/fabric-dev-servers
$curl -O https://raw.githubusercontent.com/hyperledger/composer-
    tools/master/packages/fabric-dev-servers/fabric-dev-servers.tar.
    gz
$tar -xvf fabric-dev-servers.tar.gz
```

Now we need to run the downloaded script for installation of Fabric binaries and images:

```
$cd ~/fabric-dev-servers ./downloadFabric.sh
```

Project Creation and Deployment :

Now we will create a skeleton business network using Yeoman. This command will require a business network name, description, author name, author email address, license selection and namespace. Run the below command and enter the required Details:

```
$yo hyperledger-composer:businessnetwork
```

If we select No when asked whether to generate an empty network or not, then we will get a number of pre-created files. Now we need to define our Business Description as per our need and accordingly change the various files such as the *permissions.acl*, *models/¡namespace¿.cto*, *lib/logic.js* and also create some new files like *queries.qry*.Also we need to copy some scripts from main folder to our project folder. These file include *startFabric.sh*, *stopFabric.sh*, *loader.sh*, *createPeerAdminCard.sh* and the *fabric-scripts*.
Now we need to generate a business network archive for our business network.

In the project folder we need to to run the following command:

```
$composer archive create -t dir -n .
```

Next step would be to Deploy the business network. After creating the .bna file, the business network can be deployed to the instance of Hyperledger Fabric. Normally, information from the Fabric administrator is required to create a PeerAdmin identity, with privileges to install chaincode to the peer as well as start chaincode on the composerchannel channel.

To start the fabric and create peer Admin card we need to run the following commands:

```
$./startFabric.sh
$./createPeerAdminCard.sh
```

Next step is to deploy the business network onto the amdin card. Deploying a business network to the Hyperledger Fabric requires the Hyperledger Composer business network to be installed on the peer, then the business network can be started, and a new participant, identity, and associated card must be created to be the network administrator. Finally, the network administrator business network card must be imported for use, and the network can then be pinged to check it is responding.

To install the business network, from the project directory, run the following command:

```
$composer network install --card PeerAdmin@hlfv1 --archiveFile <
    project-folder-name>@0.0.1.bna
```

To start the business network, run the following command:

```
$composer network start --networkName <project-folder-name> --
    networkVersion 0.0.1 --networkAdmin admin --
    networkAdminEnrollSecret adminpw --card PeerAdmin@hlfv1 --file
    networkadmin.card
```

To import the network administrator identity as a usable business network card, run the following command:

```
$composer card import --file networkadmin.card
```

Finally to check that the business network has been deployed successfully, run the following command to ping the network:

```
$composer network ping --card admin@<project-folder-name>
```

To run and test with playround, run the following command :

```
$composer-playground
```

To create the REST API, navigate to the tutorial-network directory and run the following command:

```
$composer-rest-server
```

After running the above command enter the required details as needed.

Conclusions and Future Work

[?]

Bibliography

Papers

Exploring Research in Blockchain for Healthcare and a Roadmap for the Future Mohamad Kassab, Joanna DeFranco,,Tarek Malas, Phillip Laplante, Giuseppe Destefanis, and Valdemar Vicente Graciano Neto

A performance study of Hyperledger Fabric in a Smart Home and IoT Environment by Salman Ahmed

Performance Benchmarking and Optimizing Hyperledger Fabric Blockchain Platform Parth Thakkar,Senthil Nathan N, Balaji Viswanathan

IBM's Why new off-chain storage is required for blockchains, Document version 4.1

BlockIPFS - Blockchain-enabled Interplanetary File System for Forensic and Trusted Data Traceability

Websites

https://hyperledger.github.io/composer/

https://docs.ipfs.io/

https://www.hyperledger.org/use/fabric

https://swarm-guide.readthedocs.io/en/latest/resources.html