Addressing Scalability Issues in Blockchain: A Use case from Healthcare

Lipsa Sadath (lsadath@amityuniversity.ae)
Amity University
Deepti Mehrotra
Amity University Uttar Pradesh
Anand Kumar
Amity University

Research Article

Keywords: Blockchain, BCT, Scalability, Smart Contract, Hyperledger Fabric, Hierarchical Model, Healthcare Sector, Patient Data, Privacy, Data Security

Posted Date: August 3rd, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1903767/v1

Abstract

The opportunities with blockchain technology has grown immensely especially with its unique features such as immutability, transparency, decentralization etc. It is also important that the basic features such as privacy, security and scalability co-exists. Our novel study proposes a framework named ERSB (Enhanced Reliability of a Scalable Blockchain) implemented with Hyperledger fabric as a solution to address scalability issues. Multiple records are generated and processed during a patient's hospital visits/stays, laboratory tests and insurance claims. When there are networks of hospitals under one umbrella (a group of hospitals), it is critical to address maintenance of this complex system in context of the afore-mentioned unique features. Our proposed framework uses a hierarchical model at hospital networks, laboratory and insurance company that has multiple levels. Stages are evaluated with nodes added along with smart contracts. Smart contracts are also amended with more changes to observe performance, reliability and latency of the model at the first level of hierarchy.

1. Introduction

Technology took a real turn with Blockchain as a medium that enabled data transfer between peers or participants as that established transparency without intermediaries in the system [1], [2]. This was a strong discovery. When there was a stock exchange collapse, Satoshi Nakamoto first established the concept in 2008 through the Bitcoins [3] - the first cryptocurrency. With smart contracts [4] in ethereum [5] and their popularity, the concept was much used in developing most of the blockchain built systems in other entities. These included finance, healthcare, education, government organizations, supply chain [6] and now extended to artificial intelligence[7] etc. The technological breakthrough that had the potential to revolutionize the way society did trade and interacted. It gained reputable label for its ability of allowing mutual untrusted nodes to exchange financial value and engage without the involvement for a trust worthy third party member.

Bitcoin was one of the first to successfully implement Blockchain technology and attract the public's attention. After which, many crypto companies and currencies emerged thereby making Blockchain a highly discussed and popular topic. Blockchain combines technology features like cryptography, P2P networking, distributed system, transparency, access identity permissions, open source, autonomy, and immutability. This makes the technology very secure as the transactions cannot be tweaked or tampered with and all the entities/parties, which make the transactions, are kept anonymous [8].

Thus, Blockchain offer integrity protected distributed ledgers, provides transparency of the entire process by upholding a set of global states. The participating nodes agree upon the existence, value and histories of all states. Each state contains multiple transactions. Hence, the Blockchain is a way to manage distributed transaction. The concerned entities preserve replicas of the data and jointly agree on a transaction execution order. This high performance blockchain system is a means for completely monitoring a system as if under the surveillance of a legal third party. However, then as more users

entered the blockchain systems, there appeared the issue of scalability [9]. Our study addresses the issue of scalability in blockchain technology with the help of a use case in the healthcare sector.

At the time of this experiment, the system is evaluated at level-1. That is joining the basic 3 organizations. To test and understand the scalability issues, we have presently used one channel through the three organizations adding peers through all three. Transactions were increased from 10 to 1000 to know the performance of the system. The paper discusses scalability issues, structure of blockchain, different consensus models, relevance of hyperledger fabric, hierarchical model and the proposed ERSB model.

2. Scalability Issues

With more users in the public chains such as Bitcoins and Ethereum, the scalability issues have arisen from sometime [10] with blockchains. To an extent, this has affected the technology in terms of development that can be implemented in other domains also. Much research has taken place in this area by researchers as well as software development companies. Though many of these solutions act to the needs somewhat, many other important aspects of blockchain systems such as security and decentralization were the biggest challenge to maintain [9]. Thus as more nodes increases, validating every node and every transaction was an issue due scalability. At the same time, it is important that all transactions validate themselves for the next transactions. This is important for appropriately further validating the entire system.

One of the major techniques that could be used was to redesign the blocks or to do a storage optimization. Even as the blockchain challenges exists, the technology still thrives to excel in its implementations. Many frameworks are in discussion on these grounds. Few of them were on-chain and other few off-chain. Especially with smart contracts in picture.

3 Blockchain Structure And Smart Contracts

A complete block structure [56] will have the following data structure that extends from the genesis (first block) to the block “n”. The application can be in any area like cryptocurrency, asset management or any security systems. Figure: [1] explains this concept. There are blocks created and added to the chain that starts from the genesis block. Every node added runs the chaincode or smart contract to form the block and keep the ledger same for all the peers. Every block will have transaction and hash details including that of the previous block.

A legal agreement or contract that is in the form of digital set of codes between two parties during a deal is how a smart contract works in the blockchain environment. This ethereum concept in cryptocurrencies verify the nodes in a decentralized network. Thus, smart contracts initiate a deal as if it is a trusted third party in action [4], [11]. The integrity of smart contracts is also a concern during implementation. This is even more essential in a public network where unknown parties get involved in transactions. Mainly because public networks use smart contracts where nodes are unknown to each other. This may not be of

large concern in private organizations as the verifications take place at organization level itself and controlled more by a central authority that is the organization itself.

When blockchain technologies (BCT) have the cryptographic models to maintain security of the systems, consensus protocols [11] add complexities to these models for quality and trustworthy BCT transactions. It is important that the nodes separately at their ends run these consensus protocols in order to ensure these nodes reach the required consensus about their previous transactions. This will help new block entries in the correct order needed. The nodes get ready to participate in any transaction with the hashes of the previous blocks. Typically, validation is established to improve the trust among all nodes in the system because all nodes produce identical outputs. This is in accordance with the consensus rules.

4 Consensus Protocols

Nevertheless, new challenges and security issues keep on emerging every day due to which the design of a consensus protocol remains a major issue. The consensus of Blockchain is that every entity/node maintains the same distributed ledger. In the blockchain, the entity/party is both the server and the host. It requires exchanging information with other similar entities/parties to arrive at a consensus. It is important that the Consensus Protocol implemented must also be suitable for certain blockchain type that has the framework and purpose for which the blockchain is used [9],[12]. Some of the major Consensus Protocols are discussed below:

A. Proof of Work (PoW)

Bitcoin, the popular Cryptocurrency, uses the Proof of Work consensus. All participants or miners compete to resolve a computational puzzle in order to generate a new block [3]. Here, on resolving the puzzle, the miner broadcasts the message to other participants for verification and gets a chance to create a new block. This protocol provides security in the form of block mining. But this protocol achieves its consensus by placing blocks in a distributed system even though the notes may show faults such as them being unreliable, unavailable, malicious, and more. The security of these kinds of the network is supported by specialized hardware and electricity, which makes them overpriced and inefficient from a resource and environmental standpoint [13].

One of the main challenges of this protocol is Selfish mining. It was an attack discovered by Eyal and Sirer [14]. Selfish mining is when a miner keeps his/her discovered blocks private and continues to mine over them. This attack enables Selfish miners to avail unfair rewards. This scenario destroys the decentralized structure of the system and raises the success rate of various similar attacks. Another kind of attack is Double Spending [21]. Double Spending is when spending happens again on the same asset that is, an attack mode when a currency is stolen and the attacker transmits a copy of the transaction as though it is original. Earlier, double spending was thought to be difficult and impossible due to the mining power. But, in 2016, a paper published by Sompolinsky and Zohar [15] exposed that an attacker could easily launch this attack on low mining power with the combination of Selfish Mining.

Another kind of attack is Feather Forking [16],[17] proposed by Miller. In this attack, the attacker forks the blockchain to render the entire blocks invalid confirming the target. The attacker declares this publicly. The attacker continues the mining until the main chain is “k” blocks forwards. Luckily, the attack is considered unprofitable and the success rate is low with the mining power. Scientists argue that the latency is less when miners follow the consensus and compliant strategies [18–20] and this improves scalability to an extent.

B. Proof of Stake (PoS)

In Proof of Stake [8], [9] a method selecting which node can create, a new block relies on the stake rather than computational power. That is, participants get a chance to vote and elect leaders. These leaders are elected by their investments. Hence the name Proof of Stake. The challenges of verification here still exists because of the computational elimination. Etheruem uses the PoS consensus [22]. What makes this different from Proof of Work is that the nodes do not need to adjust their nonce many times instead; the nodes can solve this with the amount of stake. This makes proof of stake an energy-saving consensus because it does not use lots of computational power to arrive at a consensus [23]. For the block to be valid, it needs to have an address and a timestamp. Along with that, the user also needs to provide proof of ownership of the address.

Also, due to one block being created per round of Proof of Stake [24], the generation and transaction speeds of blocks are much quicker than Proof of work. This has led to the recent popularity of this protocol. The security of Proof of Stake relies on many factors. A few of the Proof of Stake protocols are considered secure as long as the messages sent by network reaches their destinations within a certain time limit and handles scalability to extent.

C. Delegated Proof of Stake (DPoS)

This model has a small group of delegates elected by the stakeholders. These delegates produce and validate the blocks. This algorithm was used for addressing scalability issues in Bitshare and EOS. The model reduce the time taken for the block generation and considered as just 3 secs. This divides the system into 3 parts. The first part is the witnesses that are elected through the voting system and play the main role by generating the blocks. The second is the delegates and the third is the workers. Security concerns arise for the fact that witnesses usually hold the block generation rights for a long period unless for a specific reason. This is considered as a risky situation [25].

In Delegated Proof of Stake, the node that create the block is also responsible for validating the transactions. In this protocol, a token holder can vote for their preferred block producer and their votes will be prioritized based on their stake. In addition, Token Holders may give their stake to another voter and cast a vote on their behalf in the block producer election [26]. Once they are elected, they may create blocks after verification of transactions that were created for the last block time in the order in which they were selected. If all blocks in the transactions are verified and signed, they receive rewards. Also, the rewards are shared equally with the users who voted for them. However, the rewards will not be given if

they do not do their task in the allowed time. After which if the block is missed, and transactions remain unverified. Users could also vote out any offending party/delegate if they believe that there is malicious intent or activity involved thereby keeping real-time voting secured. Another advantage is its efficiency for Generating Blocks [25]. The number of blocks created by this protocol increases as time increases. The reason for this is that the time taken to select the node remains unchanged, which is why the protocol's time increase. This leads to a slight increase in the creation of blocks. Compared to traditional protocols, this one has improved security, efficiency and better decentralization of block generation [27]. The model was further taken for other improvements using the lattice models as well [28].

D. Practical Byzantine Fault Tolerance (PBFT)

In current blockchains, Bitcoin uses Proof of Work (PoW), Ethereum uses Proof of Stake (PoS), other consensus were also discussed. The PBFT algorithm was created from Byzantine Generals Problem. It remains unharmed by malicious attacks and is implemented in many distributed computing environments.

A Byzantine fault occurs when in a distributed system, the nodes fail and the information about the failure is inconsistent and unclear. The community has discussed quite a bit about this problem. It is still unclear as to what vulnerability brings this condition to a system. These discussions have also proposed various architectural solutions [29–31]. A solution to this was with the proposal of PBFT (Practical Byzantine Fault Tolerance) [32], [33].

In PBFT, there is a group that requires a leader, who is then elected using an 'Election Algorithm'. Hence, the other nodes are known as Replicas. If a primary node fails, a new leader maybe selected with the creation of a new group. However, the nodes do not change their state and the requests remain organized. This is what makes this algorithm safe [34]. The system has a protocol with a 3 phase. The first phase is pre-prepare, the second is prepare and the third is the commit phase. The entire system has one message orderer server that begins the working of the phases on receiving a client request. This primary server broadcasts a message to pre-prepare to all other replica servers. During the prepare phase, if the any server makes a decision to accept the broadcasted pre-prepare message, then the server broadcasts accepted messages to all other replicas again. While "f" is the number of faulty nodes indicator, the server starts the commit phase, when it receives a 2f + 1 messages as feedback. As in the prepare phase, the commit phase also has servers sending broadcast messages to others and wait for 2f + 1 messages as feedback. This is a kind of voting to indicate that majority of the servers and clients agree on the messages sent [8].

IBM utilized PBFT for developing 'Hyperledger fabric' that provides member registration, identity management, auditability and more which helps to achieve high performance in certain environments [35]. While scalability is limited and needs improvement, PBFT has shown excellent performance with latency and throughput. Due to this protocol's fault tolerance, many experts add this to their blockchain systems [36]. PBFT works in a synchronized system and needs many voting rounds, which makes it impractical. PBFT has inspired many BFT protocols with better security and performance. For

synchronous networks, PBFT may exploit the timeout algorithm and detect of any issues or anomaly of the primary node [37]. Though PBFT works without computational tasks, it might cause some communication overhead. The utilization of PBFT in the Hyperledger fabric is an advantage in private organizations in terms of security, latency and throughput where the system is in a controlled permissioned environment.

E. Hybrid and Other Consensus

When protocols are combined, they form to give some classic models like the hybrid ones. PoW and PBFT were combined and used in Byzcoin [38]. Scalability was an advantage here when PBFT was able to reduce the cost of the prepare and commit phases by using a protocol called Cosi [39]. Mainly PoW was seen combined with many other consensus to form hybrid consensus [40]. Other consensus models that were proposed to improve the scalability was Proof of Authority (PoA) [41] relies on a very limited number of block validators that undergo rigorous security and identity scrutiny, thereby protecting the security and enhancing scalability. Proof of Participation (PoP) [42] was built on PoS and PoW combined. Unlike PoW the computational task is simpler and the transactional fee is distributed among participants which encourages them. Proof of Capacity (PoC) depends on miner's disk space capacity to generate next block and get a reward. This is much similar to PoW but with less complex computational tasks.

5 Types Of Frameworks In Bct

There are various blockchain frameworks such as Hyperledger, Openchain, Graphene, Corda, Multichain and various other blockchain frameworks. Since Blockchain technology can be used anywhere, major interest from various organizations has increased and it has caused more and better technology/frameworks. Hence, the technology is used in finance, commerce, healthcare, government, agriculture, and various other fields you can think of. For this to be implemented, organizations are required to choose a certain type of framework to suit their demands [44].

Openchain is a framework released by Coinprism. A framework that is designed to issue and manage digital assets. This framework is mainly used for finance and banking organizations. It also allows us to manage various kinds of financial transactions. It has enhanced anti-fraud and anti-theft systems, which is optimized and adaptive to different systems or needs. The architecture is mainly based around a client-server framework, which is more reliable than peer-peer framework. It is also a private network, and various users on GitHub actively update it. It is also open source and runs on JavaScript [45].

Graphene is a framework written in C + + language and it is the most adaptable one. It was created as an exchange medium for various cryptocurrencies, since then it has been used for various other fields. It is the most transparent and reliable framework it can process around 5,000 to 100,000 + transactions per second which is most required by large financial organizations. It has the fastest block creation time as compared to other frameworks. It is an open source, and it is a private/public network [46].

Corda is a framework written in Kotlin and Java, and it works as a mediator for businesses to cut down on transactional expenses. It has also helped with reducing the cost of record keeping and help business make their transactions mobile. The framework is open source, and it can be used to store and synchronize data between various financial institutes. It helps with keeping an open network to transfer various valuable assets with privacy. Its network is private, and permission based, therefore those with access can view or use the assets on the given server/network [47].

Multichain is written in C + + language and it can help you create a private block that can be used by your company or by various other companies. It was initially created to beat major problems affected by blockchains in financial areas, it even offers privacy packages and access levels for compartmentalizing data for protect. It even allows user permissions for the organization to set which user has which type of access. Members of the framework can manage the size of the block but can only view the data if they have their own private key. All blockchains created in the framework are regionalized which increase the security level. Multichain is easily configurable and modifiable [48].

One of the frameworks used by Hyperledger known as Hyperledger Fabric, which was created by various Japanese companies. This is a collaborative project to develop an open-source blockchain framework for enterprise usage. Its goal is develop blockchain technology by developing and implementing a cross-industry method for distributed ledgers that has the potential to revolutionize the way businesses transact globally in an open –standard platform. The system gained more popularity because of the PBFT used in it and ease of implementation and community that supported the framework. Hence, Hyperledger fabric framework is used to implement our use case.

6 Hyperledger Fabric

Hyperledger is an integrated blockchain framework that acts as a base for developing various blockchain-based products and applications using components that can be used by private entities. It is open source. It is designed in an environment in which multiple users can work within, which makes it suitable for finance, healthcare, and various domains. The community is quite huge and are its members [49] are currently funding it.

In the year 2016, the Linux Foundation launched the project. The framework was then created to host a mobile-focused framework that is easy to use and execute. Hyperledger Fabric is an open, proven, enterprise-grade, distributed ledger platform. It has advanced privacy controls so only the data you want shared gets shared among the “permissioned” (known) network participants [50]. Some of the main benefits [51], features and applications of the framework are discussed below:

A. Benefits

i. Permissioned Network: Decentralized trust in a network of participants.

ii. Confidential Transactions: Only data you want to share will be sharable.

iii. Pluggable architecture: It is easily usable regardless of usage size/environment.
iv. Ease of use: Custom languages and custom architectures.

B. Features and Applications

The core purpose of Fabric is Blockchain services; this category includes components such as a consensus manager, distributed ledger, peer-to-peer protocol and ledger storage [52]. The consensus manager is in charge of providing an interface to the consensus algorithm as well as receiving transactions from other Hyperledger networks and executing them according to the desired consensus algorithm type [52]. Certain use cases and specific trust models have adapted this for modular consensus protocols. The platform includes a native implementation of Solo, a consensus protocol in which the endorsing nodes execute and validate transactions in accordance with endorsement policies. Nevertheless, it is easy to adapt as other consensus protocols like the PBFT are used to be able to allow several ordering nodes, can be plugged in [53] [54].

Smart contracts use distributed ledger to record relevant state information during transactions. These transactions include chaincode. Which executes operations that may result in the global state being updated. The state is saved on a disk by each node. Fabrics block structure includes data like version, timestamps, transaction hash, state hash and prior hash. The other component is the peer-to-peer protocol, built using Google's googleRPC platform. Protocol buffer defines the message structure in Fabric. The network discovers peers and executes confidential and public transactions using various messages. The final component is the ledger storage; RockDB is used to save state [52]. Major features are:

i. Membership Service:

Fabric implements a private permissioned model by utilizing a portable idea of membership coupled with industry standard identity management systems. A membership service provider or the MSP determines a node's identity in a Fabric network. These nodes may include client nodes those who may propose transactions for or peers who maintain the ledger and its state; or ordering service nodes who determine the order of all transactions [53][54]. These features work together to provide fabric users with access control. It integrates public key infrastructure to facilitate authorization and identity management operations.

There are three certificate authorities; Certificate Authority is one that will provide a long-term certificate to enrolled participants in order to provide to verify their identity. The transaction certificate authority issues transaction certificates to participants in order for them to send transaction via the network. TLS certificates are issued by a TLS to secure network level communication between fabric nodes. Fabric offers an entirely new blockchain design to accommodate flexibility as well as a revamped approach to dealing with non-determinism resource exhaustion, and performance threats [52].

ii. Chaincode Services:

Fabric implements application logic with smart contracts known as chaincodes [53]. Chaincodes execute inside a secure container that this service creates. It is comprised up of two parts: a secure container and a secure registry. Deploy Transaction – take the installed chaincode, written in the Go programming language and is ready to use [52].

iii. Invoke Transaction:

Invoke the transactions of a specific chaincode previously installed. Chaincode runs transactions and updates the state, then reports if the transaction was successful or unsuccessful [52].

iv. Nodes:

The fabric has three types of nodes namely the client, peer and ordered. The clients are the application users that initiate a transaction, a peer maintains a transaction by committing them and maintaining the ledger state by also letting other peers know the status, orderer nodes ensure the transaction communication and services by creating the blocks.

v. Peers:

The peers are owned by organizations. It is a must that organizations own at least one peer. They responsibly notify the clients about the application submission or transaction update. The diagram shown below shows peer communications. The peers can be from different organizations communicating and doing transactions. Each peer can be independently running in their own environment. And can get connected to one or more channels. Thus peers are assigned a specific role in an organization and given blockchain access by a Certificate Authority [66]. The peer waits for other peers for the ledger approval. This process is the consensus.

vi. Consensus:

The consensus in a hyperledger fabric has three steps of endorsing, ordering and validating the transaction. These are steps in a transaction flow. First the transaction is run and checked if it's a correct transaction. Then the ordering takes place. This step does not have any grammar check or semantic check. Finally its validated.

7 Blockchain In Healthcare –use Case

Because of data replication, healthcare suffers the largest issues at all times applications due to scalability. This is because patient data are updated all the time based on regular checkups. A patient undergoes many tests from different laboratories or departments in a hospital (Figure: 3) which may include tests like X-Ray, MRI scan or blood tests. These hospitals may belong to a network of hospitals. Data may be replicated in such scenario. Majority of these data is used for Research and Development. The data may also include many patient personal data. In a typical blockchain scenario, the data is

encrypted and stored in a data lake. Hence, main challenges areas obviously in the medical field itself [55].

8 Hierarchical Model–a Solution To Scalability

Scalability enhancement was proposed by Sahoo et.al [57] in their hierarchical model [Figure: 4] as a solution to scalability. The model uses the lattice property of set of intervals where the access control and details are given at abstract levels instead of all communications within an entire block. This abstract mathematical interpretation [58] is used in security, programming and databases [59–61]. Access control mechanism is a difficult implementation in organizations with hierarchy. A better and reliable solution is to have smart contracts in the system.

Every next hierarchy level has the BCi levels, verifications and validations that is at the private networks of the organizations. Private Blockchains are the highlight in the system at local levels in the organization. Then these are branched out to public blockchains with hierarchies. This is where the level of abstraction is proposed from a lower level.

Hierarchical Model using Hyperledger Fabric for ERSB System

There are many stakeholders for the data stored in a healthcare sector. This ranges from patients, doctors, hospitals, hospital networks to insurance companies. There can be a network of hospitals that record the patient data that are needed when there is an insurance claim. Whenever there is a reference to the data stored, we assume the patients also have an access and are aware where their data is used unlike the present scenario. This is where the hyperledger fabric and the inbuilt chaincode or smart contract is essential. Our healthcare use case assume that the insurance company access the patient data in an abstraction model and access as in a branch. Hierarchical model in clinical data study has been attempted before [62].

The model below is the hierarchical representation [Figure: 5]. It represents Insurance company on top in the hierarchy. A group of hospitals that belong to a particular network will report to the hospital network unit. Each hospital can be under the hospital network that are owned by the same group. And then there are multiple departments under each hospital. The enormous amount of patient data is a concern in such scenario. And only those channels that need the data should access them at that time through their peers.

9 Implementation Of Ersb Model

The blockchain was implemented using the IBM’s hyperledger fabric model [63] with the hierarchical model [57]. The implementation [Figure: 5] shows how the first level of BCT is established between three organizations. That is one single channel that holds the same data that is accessible by all the organizations in the chain. The organizations that are used in the use case are Insurance company that gets all the data from a hospital network, The hospital network that holds data from all the concerned

hospitals that come under them, and a set of ‘n’ hospitals that are linked to the same network, For example; a hospital branch that is in Texas (United States of America) and another one in Edinburgh (United Kingdom). A client who may take treatment from both the places should be able to access his data safely and know and discuss his rights with the Insurance company in whichever part of the world he goes. All the ledgers at all the hospitals are expected to update themselves after every transaction.

Figure 6: Implementation of Hyperledger fabric in the ERSB Model

The experiment uses Java script on the chaincode [Figure: 6]. The chaincode consists of the following data in the script that the hospitals have and is accessible by the Insurance Company:

1. 1. Transaction ID
2. 2. Patient ID
3. 3. Type of service, for example: consulting a doctor, lab test, medicine purchase etc.
4. 4. Doctor ID
5. 5. Cost of the Service
6. 6. Insurance ID

Since the consensus depend on the smart contract, the transaction is validated only when the policy is met according to the rules of the insurance company. These smart contracts are otherwise called the chaincode. The chaincode is deployed in the channel and once the transaction is ordered and validated, organizations are able to access the chaincode. Each peer (p) that is hosted by the hospital, hospital network and the insurance company act as an endorser or committer who commits a transaction and keeps the ledger state updated or the world state database. Before any transaction is committed, the endorsing peer endorses the transaction. Not all peers are endorsers in a transaction but all peers are committers to store the world state database for that particular channel with the updated asset information.

The client applications (c) need to access the ledgers and for this, they get always connected to the peers. Only endorsed transactions are committed and stored and are accessible by the clients. Typically, whenever the user makes any transaction, the endorsed transaction is submitted and then committed. The user application also receives a notification about this.

Whenever the client or a patient application requires an endorsed transaction to be committed, the ordering service (O) takes this up. That is, the ordering service orders the transaction and groups them as a block, sends them to the peers (hospital, hospital networks and the insurance company). This is important because this determines the order of all the services and the transactions until then.

The MSP plays an important role ensuring all the protocols and mechanism for cryptography are maintained in the system. This include rules to check and maintain authenticity and authority of peers in the chain. Either from the fabric or an external Certificate Authority (CA) is used to issue keys and generate the MSPs in each peer (for hospital, networks and the Insurance).

The experiment has recorded the time in accessing existing records from 100 to 1000 transactions-between the 3 organizations -Insurance Company, Hospital Network and Hospital1.

This experiment currently works under the basic hierarchical model with first level of abstraction in hierarchy tested. This is much similar to the linear model at level1, as they all run through the same channel. The time to fetch the transactions are recorded with 100 transactions and then raised to 1000. The time is recorded with 118 peers or nodes added. The aim of the experiment is to replicate more number of transactions to check the throughput and latency with the same chaincode. Each time a record of complete transaction access is noted along with a search needed for a particular transaction. We assume 5 insurance companies, 100 hospital networks and each hospital to have 10 departments at least.

10 Results And Discussions

More transactions are accessible. The experiment is conducted at different network connections. Time is recorded as before adding nodes and after adding nodes. The two operations performed separately are complete transactions accessed and search one particular transaction. Chaincode is found immutable because after every change in the chaincode the network had to be restarted. Transaction fetch and search is seen affected at stage one of implementation with more nodes added to the same channel. TPS might have been affected by the network connection.

The user time is how long your program was running on the CPU, and the sys time was how long your program was waiting for the operating system to perform tasks for it. So basic benchmarking can be with user time + sys time. Real can be affected by other running processes and is considered more inconsistent. Hence we have considered user + sys time to check the results.

Observation 1:

The test results in figure: 9 shows difference in time is not very evident with our experiment readings before and after adding the nodes. The first experiment is fetching the complete transaction dataset every time with an increment in 100 transaction added at each time. Hence scalability is increased at 100 transactions.

Observation 2:

The test results in figure: 10 shows difference in time is not very evident with our experiment readings before adding nodes and after adding the nodes. The second experiment is searching for a particular transaction at a time. 100 transactions were added each time to the dataset. Hence the scalability is increased at 100.

Observation 3:

When the first part of the implementation is done, there is some significant delay that is experienced while adding the peers. Once the peers are added then more or less the fetch time is the same. When more transactions are added, the response time with or without peers are the same. Hence there is a steady increase linearly.

Observation 4:

The experiment is confirmed with a t-test performed using the time recorded during the entire transaction fetch and individual search operation. In both the above scenario, the Null Hypothesis and Alternate Hypothesis is as follows:

Null Hypothesis: There is no significant difference in the means of the sample time between the two methods (Before adding the nodes).

Alternate Hypothesis: There is significant difference between the means of sample time between the two methods (After adding the nodes).

The Null Hypothesis is accepted in both above experiments in Figure: 9 and Figure: 10 since the nodes addition has not affected the framework.

11 Conclusion And Future Work

The next level of implementation will be as shown below to test the scalability. The models will be compared with same and different chaincodes, with more channels and nodes added.

The main feature of a Blockchain system is that it believes entities to behave in an arbitrary manner, also known as Byzantine. Blockchain is more secure than traditional database systems, as it is built to tolerate Byzantine failure [64][65]. Blockchain technology introduced in a variety of industries including finance and banking, healthcare, government, security and many more. Every blockchain framework has a distinct goal and use case. When choosing a blockchain framework, there are a number of factors to consider as each framework comes with its own set of features. Ever since the conception of Blockchain implemented by Bitcoin, the world of crypto has not only popularized but also improved in terms of speed, scalability and so on. What we have discussed in this paper is just four of many consensus protocols. We have been able to analyze how they work, what is their fault tolerance etc. We have even been able describe their performance in terms of each of them and have identified the current problem / security issue. However, these all are at the time of writing this paper. New Protocols are introduced, and current Protocols upgrade regularly with better security, scalability, and speed. We hope that you were able to grasp the gist of few of the consensus protocols and their functions. The paper has further discussed different framework in BCT and why hyperledger fabric is a chosen one.

The case study focuses on the healthcare sector with hierarchical model and abstract interpretation using smart contracts at each level. At the time of writing this paper, the experiment conducted is until the first abstract level, which creates, connect between hospital network and the insurance company. As we go

down the levels in the model, scalability can be dealt by using the data only at the hospital or local level and not replicating all the data at all levels. The experiment holds three organizations in the chain where Hospital1 and Hospital 2 belong to the Hospital network presently. In addition, the network is connected to the Insurance Company.

Our future work will be an expansion from here with multiple departments in the hospitals hosting their data where each hospital will maintain their own blockchain in the private mode, which according to the model will be BC1, and the network of hospitals will be BC2. This would further extent to BC3 to the Insurance company. As more channels will be added the hierarchical levels should also increase. This level of abstraction should reduce data duplicity because data should be accessible only where required within those particular channels.

References

1. Butijn, B.J., Tamburri, D.A., Heuvel, W.J.V.D.: Blockchains: a Systematic Multivocal Literature Review. (2019). arXiv preprint arXiv:1911.11770.
2. Crosby, M., Pattanayak, P., Verma, S., Kalyanaraman, V.: Blockchain technology: Beyond bitcoin. Appl. Innov. **2**(6–10), 71 (2016)
3. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system.(2008). (2008)
4. Macrinici, D., Cartofeanu, C., Gao, S.: Smart contract applications within blockchain technology: A systematic mapping study. Telematics Inform. **35**(8), 2337–2354 (2018)
5. Buterin, V.: What is Ethereum?. Ethereum Official webpage. Available: (2016). http://www.ethdocs.org/en/latest/introduction/what-is-ethereum. html [Accessed on 07-06-2020]
6. Chang, S.E., Chen, Y.: When blockchain meets supply chain: A systematic literature review on current development and potential applications. IEEE Access. **8**, 62478–62494 (2020)
7. Salah, K., Rehman, M.H.U., Nizamuddin, N., Al-Fuqaha, A.: Blockchain for AI: Review and open research challenges. IEEE Access. **7**, 10127–10149 (2019)
8. Zhang, S., Lee, J.H.: Analysis of the main consensus protocols of blockchain. ICT express. **6**(2), 93–97 (2020)
9. Zhou, Q., Huang, H., Zheng, Z., Bian, J.: Solutions to scalability of blockchain: A survey. IEEE Access. **8**, 16440–16455 (2020)
10. Wang, H., Zheng, Z., Xie, S., Dai, H.N., Chen, X.: Blockchain challenges and opportunities: A survey," Int. J. Web Grid Services. **14**(4), 352 (2018)
11. Hanada, Y., Hsiao, L., Levis, P.: November. Smart contracts for machine-to-machine communication: Possibilities and limitations. In 2018 IEEE International Conference on Internet of Things and Intelligence System (IOTAIS) (pp. 130–136). IEEE. (2018)
12. Evermann, J., Kim, H.: Workflow Management on Proof-of-Work Blockchains: Implications and Recommendations. SN Comput. Sci. **2**(1), 1–22 (2021)

13. Zhang, R., Preneel, B.: May. Lay down the common metrics: Evaluating proof-of-work consensus protocols' security. In 2019 IEEE Symposium on Security and Privacy (SP) (pp. 175–192). IEEE. (2019)
14. Heilman, E.: March. One weird trick to stop selfish miners: Fresh bitcoins, a solution for the honest miner. In International Conference on Financial Cryptography and Data Security (pp. 161–162). Springer, Berlin, Heidelberg. (2014)
15. Sapirshtein, A., Sompolinsky, Y., Zohar, A.: February. Optimal selfish mining strategies in bitcoin. In International Conference on Financial Cryptography and Data Security (pp. 515–532). Springer, Berlin, Heidelberg. (2016)
16. Miller, A., Juels, A., Shi, E., Parno, B., Katz, J.: Permacoin: Repurposing Bitcoin Work for Data Preservation. In IEEE Symposium on Security and Privacy, (2014)
17. Miller, A.: Feather-forks: enforcing a blacklist with sub-50% hash power. bitcointalk.org, [Accessed on 03-11-2021]
18. Bonneau, J., Miller, A., Clark, J., Narayanan, A., Kroll, J.A., Felten, E.W.: May. Sok: Research perspectives and challenges for bitcoin and cryptocurrencies. In 2015 IEEE symposium on security and privacy (pp. 104–121). IEEE. (2015)
19. Miller, A., LaViola, J.J.: Jr. Anonymous Byzantine Consensus from Moderately-Hard Puzzles. A Model for Bitcoin (2014)
20. Garay, J., Kiayias, A., Leonardos, N.: The Bitcoin Backbone Protocol: Analysis and Applications. Cryptology ePrint Archive, Report 2014/765, (2014)
21. Karame, G.O., Androulaki, E., Capkun, S.: October. Double-spending fast payments in bitcoin. In Proceedings of the 2012 ACM conference on Computer and communications security (pp. 906–917). (2012)
22. Casper-Proof-of-Stake-Compendium, Accessed: Nov 4, 2021. [Online]. Available: https://eth.wiki/en/concepts/casper-proof-of-stake-compendium
23. Vashchuk, O., Shuwar, R.: Pros and cons of consensus algorithm proof of stake. Difference in the network safety in proof of work and proof of stake. Electron. Inform. Technol. **9**(9), 106–112 (2018)
24. Nguyen, C.T., Hoang, D.T., Nguyen, D.N., Niyato, D., Nguyen, H.T., Dutkiewicz, E.: Proof-of-stake consensus mechanisms for future blockchain networks: fundamentals, applications and opportunities. IEEE Access. **7**, 85727–85745 (2019)
25. Yang, F., Zhou, W., Wu, Q., Long, R., Xiong, N.N., Zhou, M.: Delegated proof of stake with downgrade: A secure and efficient blockchain consensus algorithm with downgrade mechanism. IEEE Access. **7**, 118541–118555 (2019)
26. Dantheman:. DPOS Consensus Algorithm-The Missing White Paper. [Online]. Available: (2021). https://steemit.com/DPoS/@dantheman/ DPoS-consensus-algorithm-this-missing-white-paper
27. Kaur, S., Chaturvedi, S., Sharma, A., Kar, J., A Research Survey on Applications of Consensus Protocols in Blockchain. Security and Communication Networks, 2021. (2021)

28. Zhou, T., Li, X., Zhao, H.: DLattice: A permission-less blockchain based on DPoS-BA-DAG consensus for data tokenization. IEEE Access. **7**, 39273–39287 (2019)
29. Driscoll, K., Hall, B., Sivencrona, H., Zumsteg, P., September. Byzantine fault tolerance, from theory to reality. In International Conference on Computer Safety, Reliability, and Security (pp. 235–248). Springer, Berlin, Heidelberg. (2003)
30. Veronese, G.S., Correia, M., Bessani, A.N., Lung, L.C., Verissimo, P.: Efficient byzantine fault-tolerance. IEEE Trans. Comput. **62**(1), 16–30 (2011)
31. Kotla, R., Alvisi, L., Dahlin, M., Clement, A., Wong, E., October. Zyzzyva: speculative byzantine fault tolerance. In Proceedings of twenty-first ACM SIGOPS symposium on Operating systems principles (pp. 45–58). (2007)
32. Castro, M., Liskov, B.: Practical Byzantine fault tolerance and proactive recovery. ACM Trans. Comput. Syst. (TOCS). **20**(4), 398–461 (2002)
33. Canetti, R., Rabin, T., Fast asynchronous Byzantine agreement with optimal resilience," in Proc. 25th Annu. ACM Symp. Theory Comput.-STOC, pp. 42–51. (1993)
34. Zoican, S., Vochin, M., Zoican, R., Galatchi, D., November. Blockchain and consensus algorithms in Internet of Things. In 2018 International Symposium on Electronics and Telecommunications (ISETC) (pp. 1–4). IEEE. (2018)
35. Vukoli´c, M.. The quest for scalable blockchain fabric: Proof-of-work vs. BFT replication. In Proceedings of the International Workshop on Open Problems in Network Security, Zurich, Switzerland, 29 ; pp. 112–125. (2015)
36. Feng, L., Zhang, H., Chen, Y., Lou, L., Scalable dynamic multi-agent practical byzantine fault-tolerant consensus in permissioned blockchain. Applied Sciences, 8(10), p.1919. (2018)
37. Xiao, Y., Zhang, N., Lou, W., Hou, Y.T.: A survey of distributed consensus protocols for blockchain networks. IEEE Commun. Surv. Tutorials. **22**(2), 1432–1465 (2020)
38. Kogias, E.K., Jovanovic, P., Gailly, N., Khoffi, I., Gasser, L., Ford, B., Enhancing bitcoin security and performance with strong consistency via collective signing," in Proc. 25th USENIX Security Symp. USENIX Secur., pp. 279–296. (2016)
39. Syta, E., Tamas, I., Visher, D., Wolinsky, D.I., Jovanovic, P., Gasser, L., Gailly, N., Khoffi, I., Ford, B., Keeping authorities `honest or bust' with decentralized witness cosigning," in Proc. IEEE Symp. Secur. Privacy (SP), May pp. 526–545. (2016)
40. Pass, R., Shi, E., Hybrid consensus: Efficient consensus in the permissionless model," in Proc. 31st Int. Symp. Distrib. Comput. (DISC), Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, (2017)
41. Chikezie, U., Karacolak, T., Do Prado, J.C., August. Examining the Applicability of Blockchain to the Smart Grid Using Proof-of-Authority Consensus. In 2021 IEEE 9th International Conference on Smart Energy Grid Engineering (SEGE) (pp. 19–25). IEEE. (2021)
42. Nandwani, A., Gupta, M., Thakur, N., Proof-of-participation: Implementation of proof-of-stake through proof-of-work," in Proc. Int. Conf. Innov. Comput. Commun. New Delhi, India: Springer, pp. 17–24. (2019)

43. Bano, S., Sonnino, A., Al-Bassam, M., Azouvi, S., McCorry, P., Meiklejohn, S., Danezis, G., Consensus in the age of blockchains. (2017). arXiv preprint arXiv:1711.03936.
44. Yuan, Y., Fei-Yue, W.: "Blockchain and cryptocurrencies: Model, techniques, and applications.". IEEE Trans. Syst. Man Cybernetics: Syst. **48**(9), 1421–1428 (2018)
45. Baliga, A.: "The blockchain landscape." Persistent Systems. **3**, 5 (2016)
46. Ozisik, A., Pinar, et al.. "Graphene: efficient interactive set reconciliation applied to blockchain propagation." Proceedings of the ACM Special Interest Group on Data Communication. 303–317. (2019)
47. Benji, M., Sindhu, M.: "A study on the Corda and Ripple blockchain platforms." Advances in Big Data and Cloud Computing, pp. 179–187. Springer, Singapore (2019)
48. [10] Ismailisufi, A., et al.. "A Private Blockchain Implementation Using Multichain Open-Source Platform." 2020 24th International Conference on Information Technology (IT). IEEE, (2020)
49. Thakkar, P., Nathan, S., Viswanathan, B., "Performance Benchmarking and Optimizing Hyperledger Fabric Blockchain Platform," IEEE 26th International Symposium on Modelling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS), 2018, pp. 264–276, doi: (2018). 10.1109/MASCOTS.2018.00034
50. Cachin, C.. "Architecture of the Hyperledger blockchain fabric." Workshop on distributed cryptocurrencies and consensus ledgers. Vol. 310. No. 4. (2016)
51. Baliga, A., et al.. "Performance characterization of Hyperledger fabric." 2018 Crypto Valley conference on blockchain technology (CVCBT). IEEE, (2018)
52. Sajana, P., Sindhu, M., Sethumadhavan, M.: On blockchain applications: hyperledger fabric and ethereum. Int. J. Pure Appl. Math. **118**(18), 2965–2970 (2018)
53. Polge, J., Robert, J., Le Traon, Y.: Permissioned blockchain frameworks in the industry: A comparison. ICT Express (2020)
54. Androulaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., De Caro, A., Enyeart, D., Ferris, C., Laventman, G., Manevich, Y., Muralidharan, S., April. Hyperledger fabric: a distributed operating system for permissioned blockchains. In Proceedings of the thirteenth EuroSys conference (pp. 1–15). (2018)
55. Zheng, Z., Xie, S., Dai, H.N., Chen, X., Wang, H.: Blockchain challenges and opportunities: A survey. Int. J. Web Grid Serv. **14**(4), 352–375 (2018)
56. Dinh, T.T.A., Wang, J., Chen, G., Liu, R., Ooi, B.C., Tan, K.L., May. Blockbench: A framework for analyzing private blockchains. In Proceedings of the 2017 ACM International Conference on Management of Data (pp. 1085–1100). (2017)
57. Sahoo, S., Fajge, A.M., Halder, R., Cortesi, A., A Hierarchical and Abstraction-Based Blockchain Model. Applied Sciences, 9(11), p.2343. (2019)
58. Cousot, P.: Abstract interpretation. ACM Comput. Surv. (CSUR). **28**(2), 324–328 (1996)

59. Blanchet, B., March. Security protocol verification: Symbolic and computational models. In International Conference on Principles of Security and Trust (pp. 3–29). Springer, Berlin, Heidelberg. (2012)
60. Jana, A., Halder, R., Kalahasti, A., Ganni, S., Cortesi, A.: Extending Abstract Interpretation to Dependency Analysis of Database Applications. IEEE Transactions on Software Engineering (2018)
61. Halder, R., Cortesi, A., Abstract interpretation Hassani, H., Huang, X., Silva, E., 2018. Big-Crypto: Big data, blockchain and cryptocurrency. Big Data and Cognitive Computing, 2(4), p.34. (2012)
62. Kuo, T.T., Ohno-Machado, L., Modelchain: Decentralized privacy-preserving healthcare predictive modeling framework on private blockchain networks. (2018). arXiv preprint arXiv:1802.01746.
63. Benedict, A. et.al, Course Guide, Blockchain Developer 2019, IBM MEA Skills Academy, Course code SABCD ERC 2.0
64. Wüst, K., Gervais, A., June. Do you need a blockchain?. In 2018 Crypto Valley Conference on Blockchain Technology (CVCBT) (pp. 45–54). IEEE. (2018)
65. Dinh, T.T.A., Liu, R., Zhang, M., Chen, G., Ooi, B.C., Wang, J.: Untangling blockchain: A data processing view of blockchain systems. IEEE Trans. Knowl. Data Eng. **30**(7), 1366–1385 (2018)
66. Blockchain, Overview, IBM material @IBM Corporation 2018, (2019)

Figures

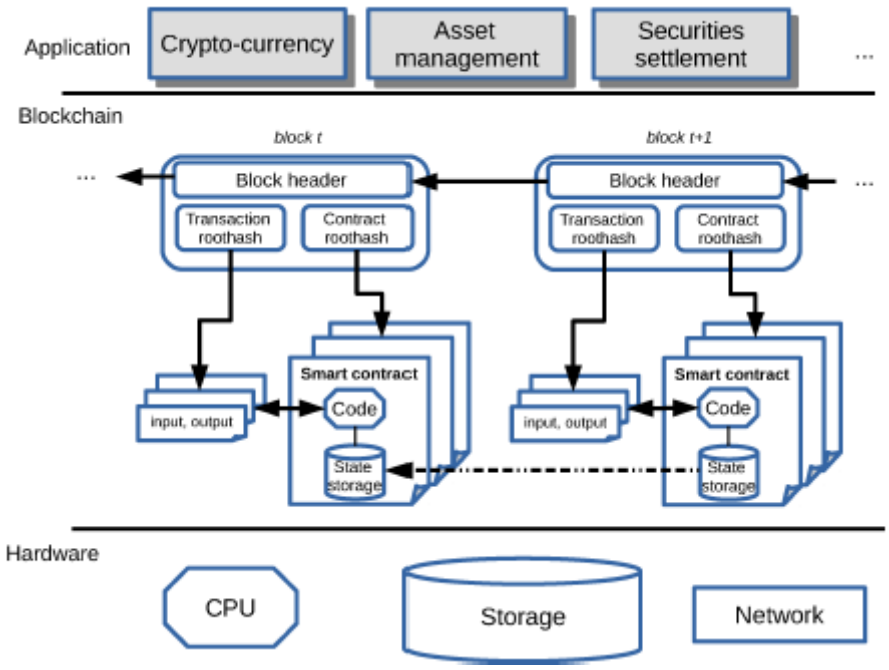

Figure 1

Blockchain Data Structure on a fully validating node

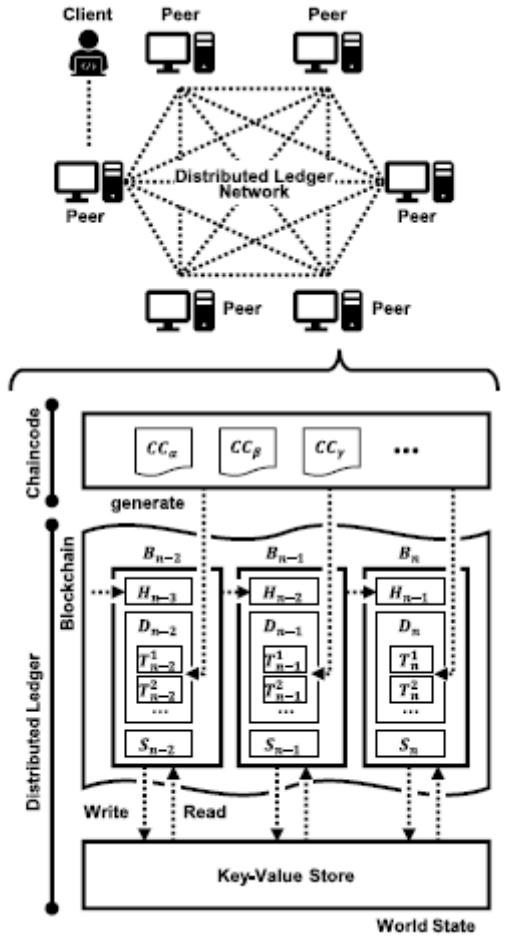

Figure 2

Hyperledger Fabric

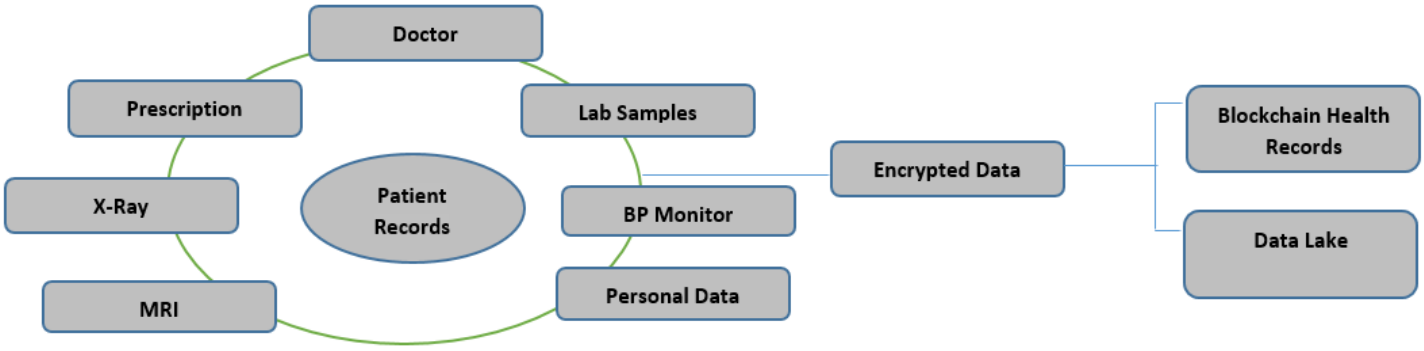

Figure 3

Representation of Blockchain data in Healthcare Use case

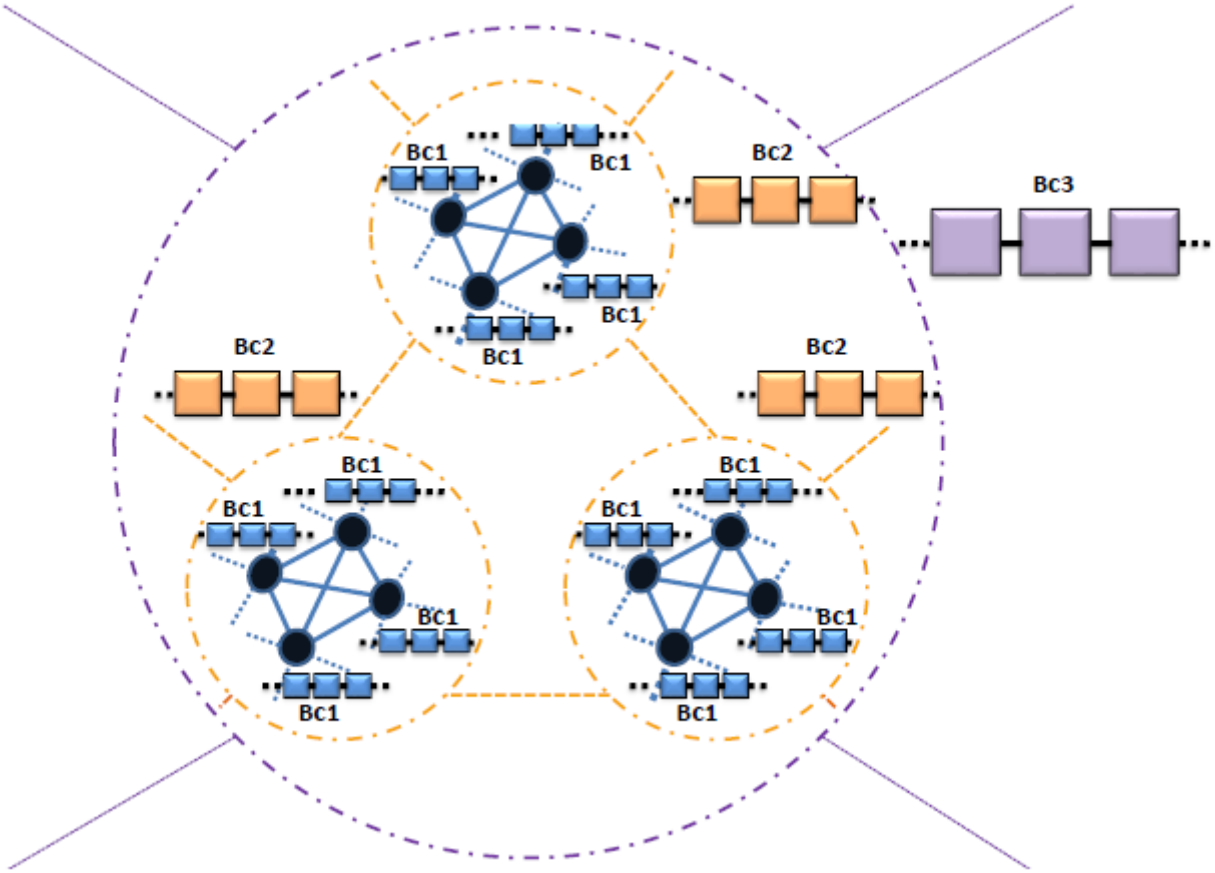

Figure 4

Diagrammatic Representation of Hierarchical Blockchains

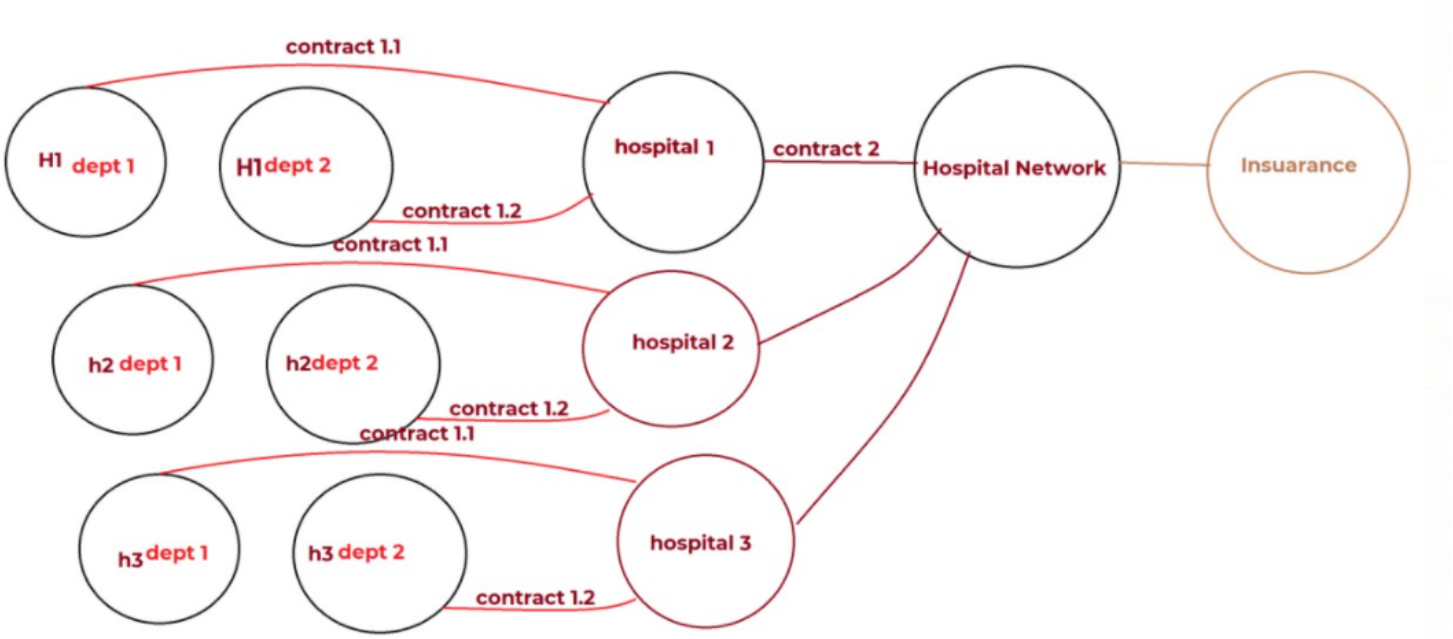

Figure 5

Enhance Reliability of Scalable Blockchain Solution (ERSB) Model

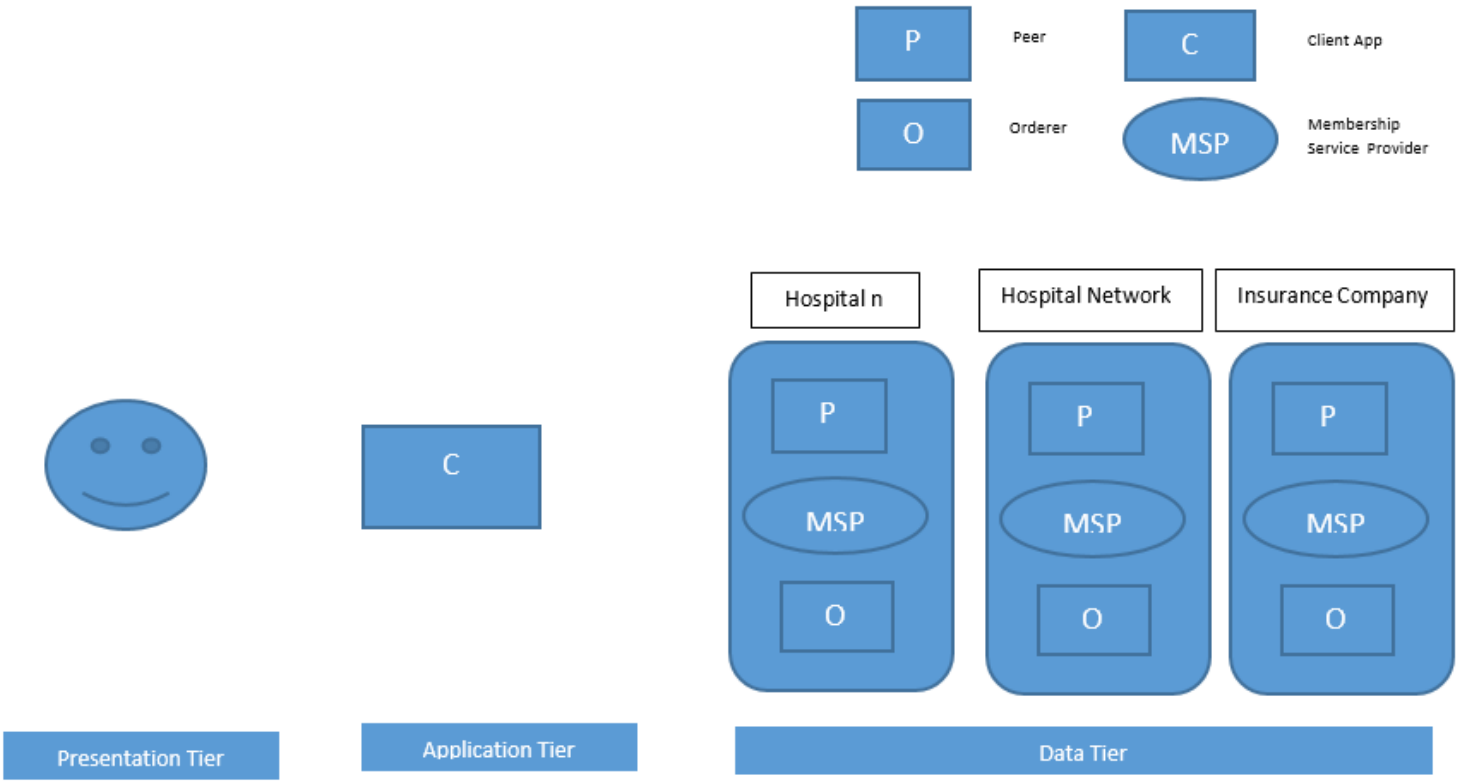

Figure 6

Implementation of Hyperledger fabric in the ERSB Model

```
{
    ID: 'transaction1',
    PatientId: '1',
    Service: 'CFee',
    DoctorId: '1',
    Cost: 300,
   InsuranceId:'1',
},
{
    ID: 'transaction2',
    PatientId: '2',
    Service: 'CFee',
    DoctorId: '2',
    Cost: 300,
   InsuranceId:'2',
},
{
    ID: 'transaction3',
    PatientId: '1',
    Service: 'MedPurchase',
    DoctorId: '1',
    Cost: 500,
   InsuranceId:'1',
},
```

Figure 7

Data on chaincode

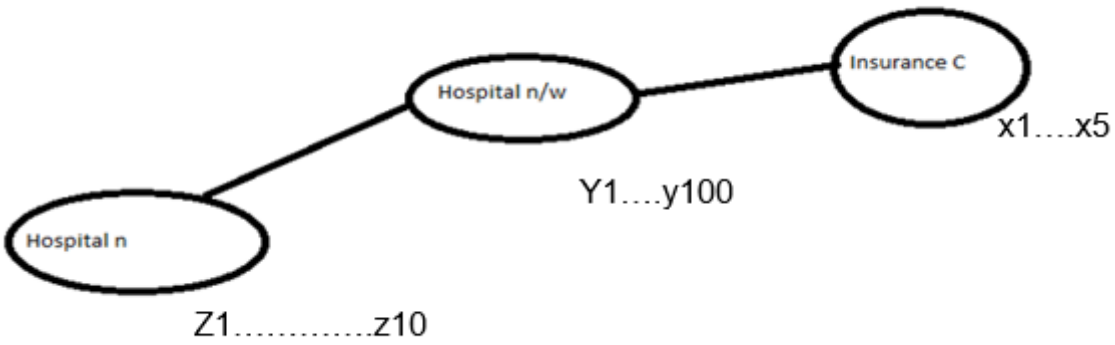

Figure 8

Example data on chaincode –First level

```
sadath@ubuntu:~/Documents/Med/fabric-samples/test-network$ time peer chaincode query -C channel1 -n basic -c '{"Args":["GetAllServices"]}'
[{"Key":"transaction1","Record":{"ID":"transaction1","PatientId":"1","Service":"CFee","DoctorId":"1","Cost":300,"InsuranceId":"1","docType":"s
ervice"}},{"Key":"transaction10","Record":{"ID":"transaction10","PatientId":"4","Service":"RRent","DoctorId":"1","Cost":600,"InsuranceId":"3",
"docType":"service"}},{"Key":"transaction2","Record":{"ID":"transaction2","PatientId":"2","Service":"CFee","DoctorId":"2","Cost":300,"Insuranc
eId":"2","docType":"service"}},{"Key":"transaction3","Record":{"ID":"transaction3","PatientId":"1","Service":"MedPurchase","DoctorId":"1","Cos
t":500,"InsuranceId":"1","docType":"service"}},{"Key":"transaction4","Record":{"ID":"transaction4","PatientId":"3","Service":"CFee","DoctorId"
:"1","Cost":300,"InsuranceId":"3","docType":"service"}},{"Key":"transaction5","Record":{"ID":"transaction5","PatientId":"2","Service":"XRay","
DoctorId":"2","Cost":1300,"InsuranceId":"2","docType":"service"}},{"Key":"transaction6","Record":{"ID":"transaction6","PatientId":"4","Service
":"RRent","DoctorId":"1","Cost":600,"InsuranceId":"3","docType":"service"}},{"Key":"transaction7","Record":{"ID":"transaction7","PatientId":"4
","Service":"RRent","DoctorId":"1","Cost":600,"InsuranceId":"3","docType":"service"}},{"Key":"transaction8","Record":{"ID":"transaction8","Pat
ientId":"4","Service":"RRent","DoctorId":"1","Cost":600,"InsuranceId":"3","docType":"service"}},{"Key":"transaction9","Record":{"ID":"transact
ion9","PatientId":"4","Service":"RRent","DoctorId":"1","Cost":600,"InsuranceId":"3","docType":"service"}}]

real    0m0.090s
user    0m0.045s
sys     0m0.027s
sadath@ubuntu:~/Documents/Med/fabric-samples/test-network$ time peer chaincode query -C channel1 -n basic -c '{"Args":["GetAllServices"]}'
[{"Key":"transaction1","Record":{"ID":"transaction1","PatientId":"1","Service":"CFee","DoctorId":"1","Cost":300,"InsuranceId":"1","docType":"s
ervice"}},{"Key":"transaction10","Record":{"ID":"transaction10","PatientId":"4","Service":"RRent","DoctorId":"1","Cost":600,"InsuranceId":"3",
"docType":"service"}},{"Key":"transaction2","Record":{"ID":"transaction2","PatientId":"2","Service":"CFee","DoctorId":"2","Cost":300,"Insuranc
eId":"2","docType":"service"}},{"Key":"transaction3","Record":{"ID":"transaction3","PatientId":"1","Service":"MedPurchase","DoctorId":"1","Cos
t":500,"InsuranceId":"1","docType":"service"}},{"Key":"transaction4","Record":{"ID":"transaction4","PatientId":"3","Service":"CFee","DoctorId"
:"1","Cost":300,"InsuranceId":"3","docType":"service"}},{"Key":"transaction5","Record":{"ID":"transaction5","PatientId":"2","Service":"XRay","
DoctorId":"2","Cost":1300,"InsuranceId":"2","docType":"service"}},{"Key":"transaction6","Record":{"ID":"transaction6","PatientId":"4","Service
":"RRent","DoctorId":"1","Cost":600,"InsuranceId":"3","docType":"service"}},{"Key":"transaction7","Record":{"ID":"transaction7","PatientId":"4
","Service":"RRent","DoctorId":"1","Cost":600,"InsuranceId":"3","docType":"service"}},{"Key":"transaction8","Record":{"ID":"transaction8","Pat
ientId":"4","Service":"RRent","DoctorId":"1","Cost":600,"InsuranceId":"3","docType":"service"}},{"Key":"transaction9","Record":{"ID":"transact
ion9","PatientId":"4","Service":"RRent","DoctorId":"1","Cost":600,"InsuranceId":"3","docType":"service"}}]

real    0m0.075s
user    0m0.038s
sys     0m0.027s
```

Figure 9

Example data on chaincode –First level installations and time recorded for transactions

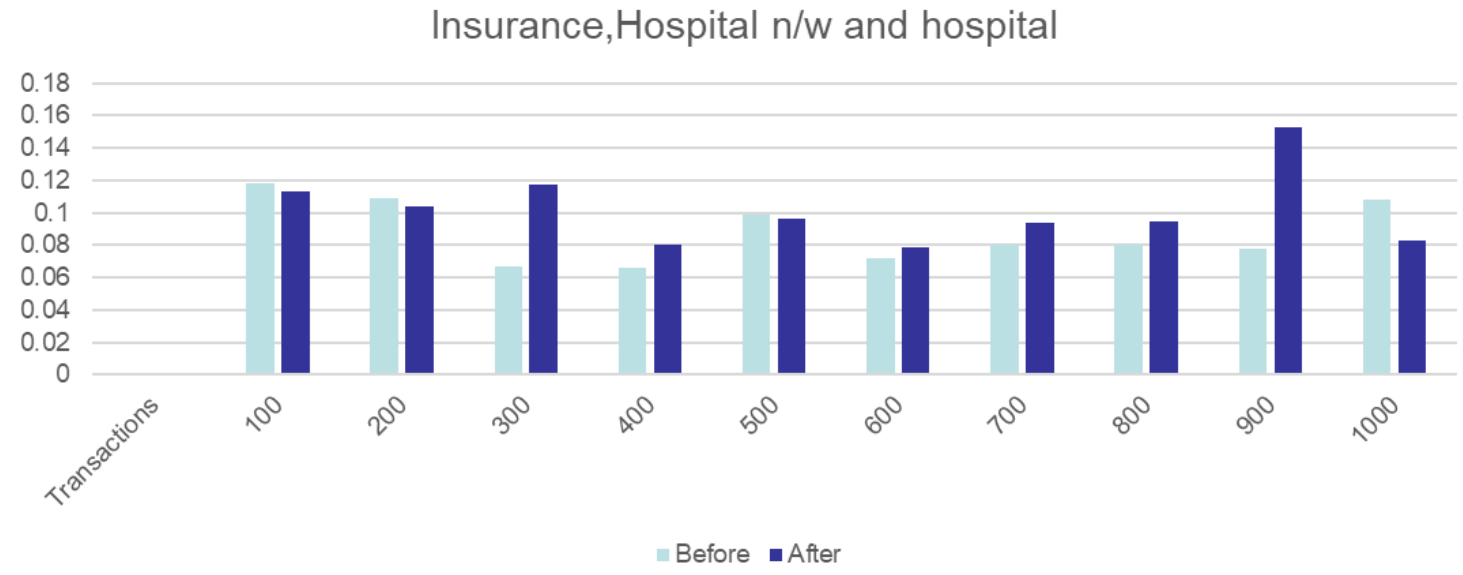

Figure 10

Sample time recorded to access complete transactions before and adding nodes

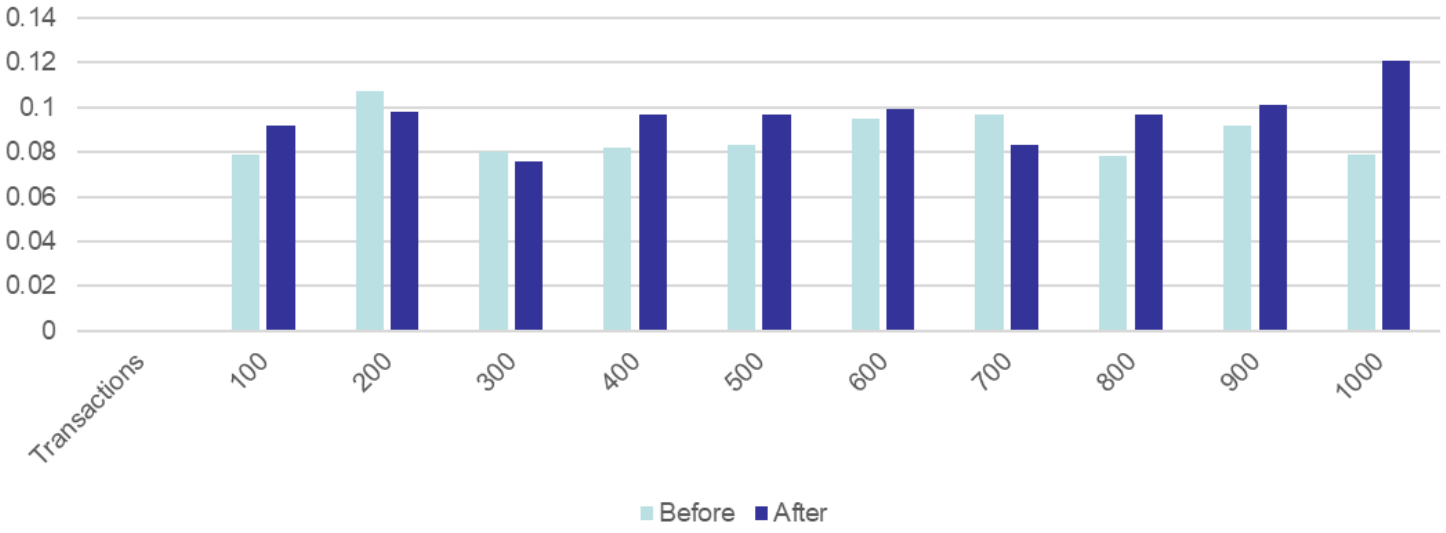

Figure 11

Results with Transactions and User Time Recorded for a particular search operation

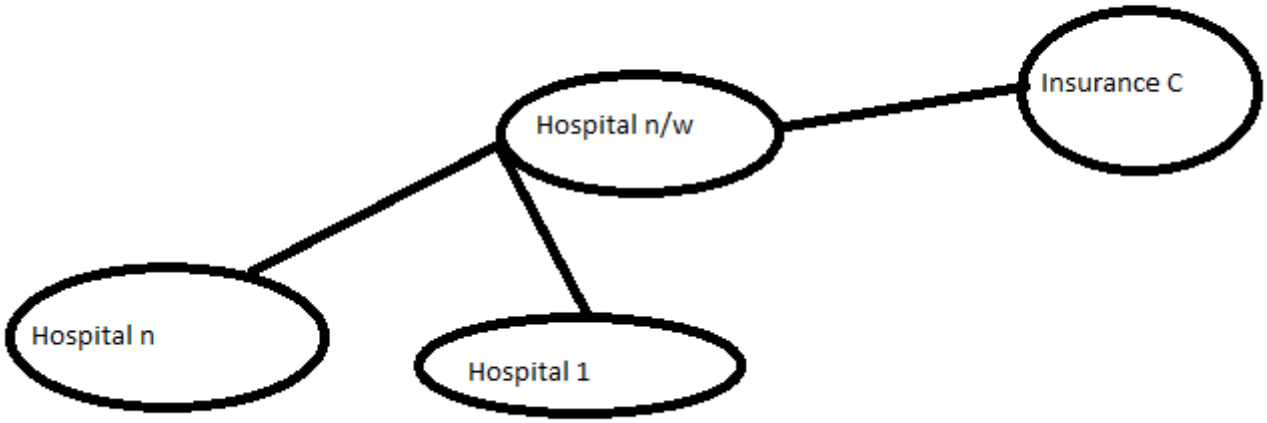

Figure 12

Example data on chaincode –More nodes added with hierarchies