Blockchain-based Data Storage Security Architecture for e-Health Care Systems: A Case of Government of Tanzania Hospital Management Information System

Richard Mnyawi[1†], Cleverence Kombe[2††], Anael Sam[3†††], and Devotha Nyambo[4††††],

[1†,3†††,4††††]The School of Computational and Communication Sciences and Engineering, The Nelson Mandela African Institution of Science and Technology, Arusha, Tanzania
[2††]Telecommunication Development Bureau (BDT), International Telecommunication Union (ITU), Dodoma, Tanzania

Summary
Health information systems (HIS) are facing security challenges on data privacy and confidentiality. These challenges are based on centralized system architecture creating a target for malicious attacks. Blockchain technology has emerged as a trending technology with the potential to improve data security. Despite the effectiveness of this technology, still HIS are suffering from a lack of data privacy and confidentiality. This paper presents a blockchain-based data storage security architecture integrated with an e-Health care system to improve its security. The study employed a qualitative research method where data were collected using interviews and document analysis. Execute-order-validate Fabric's storage security architecture was implemented through private data collection, which is the combination of the actual private data stored in a private state, and a hash of that private data to guarantee data privacy. The key findings of this research show that data privacy and confidentiality are attained through a private data policy. Network peers are decentralized with blockchain only for hash storage to avoid storage challenges. Cost-effectiveness is achieved through data storage within a database of a Hyperledger Fabric. The overall performance of Fabric is higher than Ethereum. Ethereum's low performance is due to its execute-validate architecture which has high computation power with transaction inconsistencies. E-Health care system administrators should be trained and engaged with blockchain architectural designs for health data storage security. Health policymakers should be aware of blockchain technology and make use of the findings. The scientific contribution of this study is based on; cost-effectiveness of secured data storage, the use of hashes of network data stored in each node, and low energy consumption of Fabric leading to high performance.

Keywords:
Blockchain; Hyperledger Fabric; data storage security architecture; confidentiality; integrity.

1. Introduction

Health information system (HIS) is a digital technology used in health care data management for the improvement of health services provision [1]. HIS contains sensitive, critical and confidential medical data used in daily operation. Compromised security of these data leads to loss of data privacy and confidentiality due to centralized storage architecture. Blockchain technology paradigm to e-Health care systems has provided improved digital service delivery with addressed security needs of privacy in data storage and sharing [2, 3].

Blockchain is an impenetrable technology that protects data from cyber attacks. Its key characteristic features are anonymity, immutability, transparency, privacy, and decentralization. These features have shown maximum security levels due to its unique architectural design [4-6]. The key pillars of the technology are data confidentiality, integrity, and availability (CIA-triad) based on principles of cryptography, decentralization, and consensus algorithms [7].

Several studies were conducted to the solve problems of centralized health systems using blockchain technology. The study done by [8, 9], used Ethereum blockchain architecture to address data sharing challenges resulting from health system scalability, interoperability, information asymmetry, and data security risks. The main focus was on patients to securely own, control, and share health data. Cloud storage through off-chain storage was implemented due to big data sets and on-chain facilitated secured data storage and sharing. The same study approach by [10] used Ethereum architecture for the security and privacy of health data sharing. Both off-chain and on-chain storage methods were implemented for security purposes.

Secured information-sharing problems were also described by [11, 12] resulting from a lack of trust and access to patients' data. Centralized health system's architecture created system vulnerability to cyber-attacks. Ethereum blockchain architecture was integrated with the health system for data security and sharing. Off-chain storage was within controlled access of patient-centered channels and on-chain ensured data security. This was the same approach that was used by [8]. Security challenges of centralized architecture were analyzed and Ethereum architectural framework was designed for individual data sharing aided by cloud storage.

The literature indicates that more has been done on data sharing and storage and little on the security of stored

Manuscript received March 5, 2022
Manuscript revised March 20, 2022
https://doi.org/10.22937/IJCSNS.2022.22.3.46

data. These studies focused on the combination of traditional database storage with distributed Ethereum blockchain architecture, while others used cloud storage and blockchain for data security. Cloud storage has associated security risks due to its centralized storage which acts as a single point of failure. On-chain storage deploys all nodes for storage purpose which lead to high consumption of storage space [13, 14].

This paper aims to show how data storage security can be improved for the case of health information systems using Hyperledger blockchain architecture, to meet security goals of confidentiality, integrity and availability. The designed architecture was implemented by developing a blockchain system that was virtually integrated with GoT-HOMIS for effective data protection from cyber-security attacks and system hacks. Data storage is implemented through private data collection to guarantee data privacy. Data is held within a database of a Hyperledger Fabric platform and managed with a private data policy. Nodes in a network are decentralized with blockchain only to avoid storage complications. This approach addressed other related security challenges which were centralized in nature. Private blockchain HyperLedger Fabric v2.3.2 platform was deployed for smart contracts embedded in chaincodes.

The paper sections are; section 2 with literature review, section 3 describes methodology of the study, section 4 describes results of the study, section 5 presents discussion of the results, and section 6 presents conclusion and recommendations.

2. Literature Review

The study on the implementation of blockchain technology in electronic health records by [9] suggested Ethereum storage security architecture to address technical issues related to; scalability, interoperability, information asymmetry, security, and data integrity of patients' records. The challenge of big data sets was solved through off-chain storage. The same technological approach for handling privacy and security of patients' data was used by [10]. Ethereum platform with on-chain storage was selected for solidity smart contracts, both off-chain and on-chain storage methods were implemented for security purposes. However, the approach of the solution to these challenges did not consider the risk of cloud storage though it solved the challenge of big data through off-chain storage. The approach did not also consider the complications caused by on-chain storage despite of the fact that data security and privacy is maintained.

The study conducted by [11] used the Ethereum platform to solve challenges on trust and access to patients' data due to centralization which creates system vulnerability. Personal health data was integrated with blockchain for data security and sharing for accessing the Personal Health Record (PHR) system. Off-chain and on-chain data storage mechanisms were deployed, and Ethereum architecture was used for smart contracts development written in solidity language. Challenges due to the large volume of data and design were also discovered by [12] due to the traditional system of data storage. The storage system was migrated to a blockchain platform which facilitated the smooth process of secure data exchange among entities. This resulted in simplified controlled data access to the individual patient through off-chain storage. Likewise, these approaches did not consider how to address the challenges of cloud storage for big data, and on-chain storage which leads to consumption of much storage space.

3. Methodology

3.1 Case Study

GoT-HoMIS is the Electronic Medical Record system running in client-server network architecture. The system is web-based on centralized database storage using Windows Server 2012 hosted in a Local Area Network. The operational environment is on PHP installed with Xampp Server and MySQL database. The study took place at the Mount Meru Referral Hospital located in Arusha region, the northern part of The United Republic of Tanzania.

3.2 Data Collection

The study gathered information on the existing system through interviews, document analysis, and public documents which gave a clear picture of the expected system to be developed. The interview was directed to the technical person, the system administrator. Systematic investigation of the current system was carried out to come up with a blockchain-based system to solve its security challenges. An investigation focused on the core functionalities and operations of the existing system features. Research materials published in peer-reviewed journals, books, technical reports and websites, were used to come up with the detailed knowledge of the system to be developed.

3.3 Data Analysis

The study used the requirement engineering process since only one participant was used during the data collection process. The data analysis procedure involved four processes namely; system requirement gathering, system requirement analysis, system requirement validation, and system requirement documentation. These processes

enabled the study to come up with a list of required system requirements.

3.4 System Development

The study used a prototype system development methodology, which has an iterative process allowing refinement of the prototype to reflect user requirements. A decentralized peer-to-peer blockchain system was developed based on a final working prototype that has users' satisfaction. The system was virtually integrated with the existing system for sharing health data using the Hyperledger Fabric framework (Fig. 5). Virtualization aims at the creation of a virtual blockchain network to avoid the cost of buying several computers that were to be configured in a real physical network. The system used Ubuntu operating system 20.04.2.0, 3.40GHz CPU, 12GB of RAM and secondary storage of 1TB installed in VirtualBox. Functional operation of Hyperledger Fabric v2.3.2 was facilitated through the installation of several tools such as; Curl version 7.68.0, Docker version 20.10.2, Docker-compose 1.29.1, node.js V10.19.0, npm 6.14.4 and python 2.7.18. JavaScript was used for the development of the smart contract. The study used Visual Studio Code version 1.55.2 for writing and editing codes.

3.5 System Testing

The system was tested based on the requirement specifications. This was done through experimentation, simulation, and scenarios to the proposed system to detect any defects with the system. The study used V-Model testing for system quality verification.

4. Results

4.1. System Requirements

Table 1 demonstrates blockchain-based data storage security system requirements.

Table 1: Proposed system requirements

S/N	Item	System requirement
1	Identity management	System components should be identified with their credentials for authentication to the network to avoid injection of malicious code from an unknown malicious entities.
2	Data integrity	The system should be able to preserve data integrity to avoid data modification which leads to loss of data integrity.
3	Data privacy	The system should be able to protect data privacy. Data should remain confidential and prevented from unauthorized disclosure.
4	Data verification	The system should be able to do data verification to ensure the addition of the right data to the database and false data is not added to the system.
5	Data validation	The system should be able to detect malicious and non-malicious data. Validated data should be shared among system users.
6	Non-repudiation	The system should be auditable for the identification of malicious actions to the system to hold accountable the responsible entity.
7	System availability	The system should be able to do an automatic backup, and authorized system users should be able to access information, resources, and services when needed.

4.2 Blockchain Data Storage Security Architecture

A blockchain is a linked list of blocks with pointers formed by transactions bundled together in a specified period (Fig. 1). A block created after the first block contains the hash of the previous block's data. Blocks store information validated by cryptographically secured nodes. The linked lists (blocks) are encrypted using hashes and digital signatures based on public/private key encryption algorithms. The hash of the previous block creates a chain of blocks making a blockchain secure. Fig. 1 illustrates a chain of five blocks and each block with its hash and the hash of the previous block. The fifth block points to the fourth block, the fourth block points to the third block, the third block points to the second, the second one points to the first block respectively. Because it is the first to be created, the first block is also known as the genesis block, and it does not point to any previous blocks.

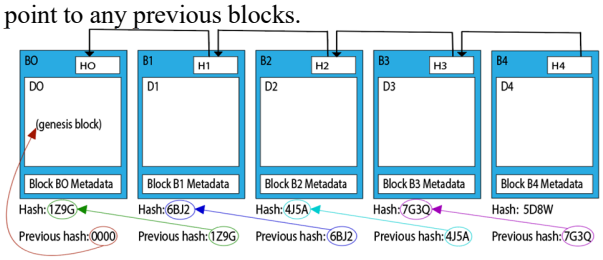


Fig 1. Demonstration of data integrity of blockchain linked lists.

A Fabric ledger consists of a blockchain and a world state (Fig. 2). A world state is a database with the values of ledger states which are in key-value pairs. Blockchain is a log of transactions that has a record of all changes from the existing world state using the metadata [15, 16].

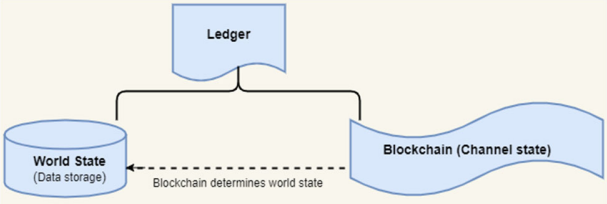


Fig 2. A Ledger comprises blockchain and world state.

4.3 Architectural Security View of Hyperledger Fabric

Hyperledger Fabric is a permissioned modular architecture blockchain with a flow of transactions that follows execute-order-validate model (Fig. 3). Its architecture consists of different types of nodes such as peers, orderers, clients with identities provided by Fabric Certificate Authority (CA). Processing of a smart contract starts with the generation of a transaction proposal from a client to endorsing peer. The proposal is endorsed and submitted back to the client. The client gathers all information of the endorsed transaction and submits them to the ordering node service. The ordering service receives a batch of transactions for ordering and submits them to committing peer for execution. A block is generated from the ordered batch of transactions, validated and committed to the ledger [15, 17, 18].

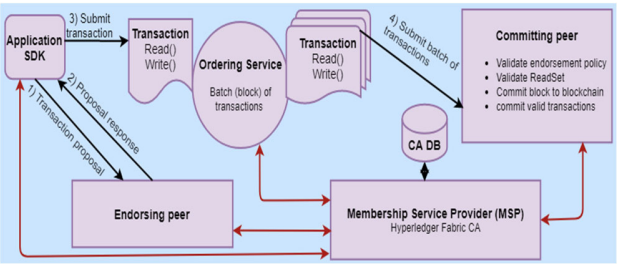


Fig 3. Security view of the execute-order-validate Fabric architecture.

4.3.1 Transaction Endorsement Policy

Fig. 3 illustrates the endorsement process of a transaction during chaincode execution. Hyperledger Fabric architecture ensures that transactions are not compromised through endorsement policies [15]. Endorsement policy ensures transaction integrity hence preventing inconsistent transactions. Transactions will be created and stored in a way that will be prevented from tampering and make it easy to detect any change in a smart contract execution [19]. A transaction proposal will be endorsed if endorsement responses listed in the policy matches to avoid unexpected results [17]. Endorsing peers can not be suspended because transaction proposals that need their approval can not proceed as well. Likewise, no new transactions will be committed if the endorsement is suspended. Endorsement is one of the deployed trust mechanisms to stop malicious peers in the system (Fig 3).

4.3.2 Transaction Verification by the Ordering Node

The main function of the ordering service is to approve the addition of transaction blocks into the ledger [20]. Transaction verification is done through communication between the endorsing and committing peers [21]. The orderer verifies all the cryptographic pieces of information of the policy and other aspects of the chaincode execution on a channel [22]. If the results of endorsement responses mismatch, invocation request will not be granted and the ledger will not be updated although data will be stored for audit purposes. This mechanism is implemented to avoid the injection of malicious code. If the chaincode policy is correct, the ordering node will send the data to all peers in the channel. All peers in the network will confirm that they have a valid transaction to be appended to the ledger. Every peer will append the read/write set to its ledger to have synchronized results.

4.3.3 Cryptographic Identification

Cryptographic identification provides security trust through the authentication of entities to the network [15, 21]. Their identities are secured by a private key and a public certificate [22, 23]. This mitigates spoofing attacks which uses the impersonation technique to tamper with trusted source credentials. Spoof attacks compromise the communication identity of an authorized user in a network and redirect to a malicious source. Cybercriminals use this attack type in combination with other attacks such as IP address spoofing, in combination with SYN flood attacks. This exposes the network to attacks through opened connections. Permissioned Hyperledger Fabric mitigates this risk by generating unique X.509 digital certificates to all its network members, revoked certificates will be denied system access.

4.3.4 Mechanisms of Digital Signatures

Digital signatures play a vital role during the endorsement process of a transaction. An endorsement request is signed by the sending client application and validated by the receiving peer. Valid transactions with the same endorsement responses will be executed and committed. Non-repudiation is attained through

mechanisms of digital signatures. There is no way that an entity or any system user can deny its actions including malicious activity. Entities can be held accountable because transactions created cannot be impersonated or forged. Membership service management grants auditable mechanisms that lead to accountability of individual Fabric components. HyperLedger Fabric screens the events using digital signatures to track who did what during ledger creation [22].

4.3.5 Contract Confidentiality

Contract confidentiality is attained through encryption algorithms during the endorsement process. Created transactions and smart contracts are concealed to unauthorized entities at the same time ensuring their correctness. Transactions can be verified if they are legal to be invoked by the respective entity. Every entity has control over its transaction sharing hence creating user participation privacy [23].

4.3.6 Transaction Validation

During validation of read/write sets to the ledger, the ordering peer verifies the chaincode to be executed on a channel [20]. If happens endorsement responses of all peers are not the same, then invocation requests will not be permitted due to data mismatch. The ledger state will not be updated due to suspiciousness on transaction differences possessing suspicious data which might be replay attacks [23]. Every node in a network is responsible for data sharing verification to make sure false data is not added and existing data is not deleted. Member nodes have to come to an agreement on whether the new block of data is valid and eligible to the shared ledger.

Replay attacks are also compared to man-in-the-middle, where the hacker interferes with the network communication between two hosts. The attacker eavesdrops on a network and intercepts it fraudulently. The hacker gains access to data during transmission and retransmits them as if it is from an authentic source. Network resources that are subjected to this attack visualize the attack as a legitimate message. Data transmitted is delayed and may even be tampered with and then resent to the receiver with malicious information. Hyperledger Fabric mitigates this attack by using read/write sets for transaction validation [24, 25]. Transaction validation is also used to address double-spending problems. It ensures ordered execution and committing of transactions are followed and no transaction will be skipped.

4.3.7 Blockchain Linked Lists

The chaining of blocks creates layered protection against cyber-security threats through encryption algorithms to maintain data integrity. The blocks are encrypted using hashes and digital signatures based on public/private key encryption algorithms. The hash of the previous block creates a chain of blocks making a blockchain secure. The chaining process of blockchain blocks hardens hacking attempts of penetrating the system, unless the hacker attacks the whole network at once which is not possible [19, 26]. More members on a network increase security hence reducing the possibility of hackers attacking the system. System attack is lowered due to the complexity created by several nodes in the network. Suppose a system hacker wants to tamper with the third block (Fig. 1), this will lead to hash changes of the block making block three and other following blocks invalid (Fig. 4). The reason for invalidity is because block three does not contain the correct hash of the previous block. Therefore, changes made to a hash of a single block will lead to the invalidation of all other subsequent blocks.

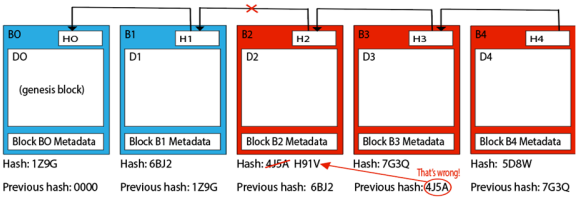


Fig 4. Demonstration of data modification detection (loss of integrity).

4.3.8 Transportation Layer Security (TLS)

Fabric architecture uses TLS 1.3 for data transit encryption to avoid accidental, and intentional data exposure. TLS is a security protocol with cryptographic algorithms for privacy and data security. The protocol provides end-to-end secure communications between Fabric components. Authentication is part of TLS using credentials created from Fabric CA to ensure authentic communications between the hosts [22]. It is also the operator's responsibility to prevent this security breach to occur by following the best information security practices of the Fabric.

4.4 Integration of GoT-HoMIS with the Blockchain System

The current system (GoT-HOMIS) was virtually integrated with the developed system, through API to enable system functionality for data storage security (Fig. 5). GoT-HOMIS records are submitted to API for data translation. Data conversion can either be from MySQL relational database to CouchDB key-value database system and vice versa (Fig. 6). Converted records in the key-value database are sent to Fabric SDK for chaincode execution. After chaincode processing, data is stored in private state

data collection in the ledger. The hashes of a private state are stored in a channel state for data integrity verification.

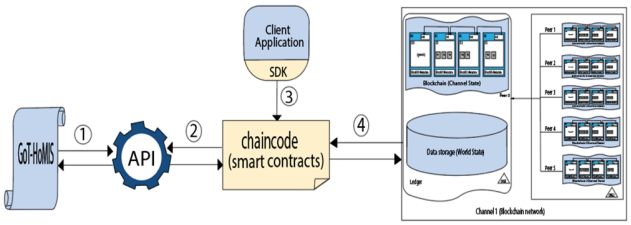


Fig. 5 System components interaction and workflow.

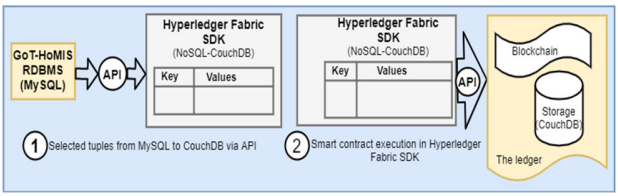


Fig. 6 No.1-selection of tuples from MySQL through API. No.2-Fabric SDK smart contract records and transactions execution for ledger storage.

4.5 System Validation

System validation was carried out based on system requirements to ensure developed system functions and operates to address security challenges. The validation procedure followed a defined order of transaction consensus lifecycle from endorsing peer to committing peer. The process involved system execution, and it was carefully monitored so that it consistently conform to expected outputs of system security. The following validation metrics were used; identity management, data integrity, data privacy, data verification, data validation, non-repudiation, and system availability.

4.5.1 Identity Management

Every unit in a channel is cryptographically identified. All identities were secured by a private key and a public certificate. Each organization in a channel has its own Certificate Authority (CA), proof of trust for its members' identities. Network member identities were created through unique X.509 digital certificates (Fig. 3). This provided security trust by proving the authenticity of the entity in a network. Network members' identity is certified to join the network while denying network access to revoked certificates.

4.5.2 Data Integrity

The chaining of blocks preserves data integrity. Even if a node is tampered other nodes will remain secure. Secured nodes will continue with data verification, keeping a record of the entire network. Any data alteration in the network is analyzed and compared to the whole chain metadata excluding those not matching. If a block in a chain is tampered, hash changes of the block tampered and other subsequent blocks will be invalidated. The reason for invalidity is due to the fact tampered block does not contain the correct hash of the previous block. Tampering with the data will need to attack every single node on the network and alter all of their data simultaneously, which is not possible [27].

4.5.3 Data Privacy

System transactions and smart contracts are hidden to unauthorized nodes at the same time ensuring their correctness. Endorsement policy ensures the confidentiality of a contract by concealing it to unauthorized entities (Fig. 3). Every entity has control over its transaction hence creating privacy of user participation. Mechanisms of contract confidentiality are implemented through encryption algorithms. System confidentiality is also attained through the chaining process where encryption algorithms are implemented. The linked lists and blocks are encrypted using hashes and digital signatures based on public/private key encryption algorithms.

4.5.4 Data Verification

Transaction verification is carried out through communication between the endorsing and committing peers. The orderer verifies all the cryptographic pieces of information of the endorsement policy and other aspects of the chaincode execution on a channel (Fig. 3). If the results of endorsement responses mismatch, invocation request will not be granted and the ledger will not be updated although data will be stored for audit purposes. If the chaincode policy is correct, then the ordering node will send the data to all peers in the channel. All peers in the network will confirm that they have a valid transaction to be appended to the ledger. Every peer will also append the read/write set to its ledger to have synchronized results.

4.5.5 Data Validation

Every node in a network is responsible for data sharing verification to make sure false data is not added and existing data is not deleted. Member nodes come into a consensus on whether the new block of data is valid and eligible to be shared in the ledger. Data validation is carried out through the endorsement process of transactions during chaincode execution. Endorsement policy ensures transaction proposal is endorsed if it matches endorsement responses that contain valid data. An endorsement request is signed by the sending application and validated by the receiving peer. New

transactions will not be committed if the endorsement is suspended.

4.5.6 Non-repudiation

HyperLedger Fabric screens events using mechanisms of digital signatures to track who did what during ledger creation. Non-repudiation is implemented during transaction endorsement and processing [28]. Identity management plays a big role in accountability to system users through auditability of user behaviour. Membership service management grants auditable mechanisms to users which leads to accountability to individual Fabric components (Fig. 3). There is no way that an entity or any system users can deny its actions. Entities can be held accountable for their transactions because transactions created cannot be impersonated or forged.

4.5.7 System Availability

Decentralized peer to peer network architecture removed a single point of system failure of centralized data storage management. This created system availability, fault tolerance, and automated data backup management where nodes in the network store the same copy of data, and information is exchanged without a central authority.

4.6 Performance Evaluation of Hyperledger Fabric Architecture

The performance of the developed system was evaluated against the Ethereum blockchain system identified in a literature review. Performance metrics focused on overall performance and detailed performance of the system. Overall performance evaluated the system's throughput and latency. The detailed performance provided detailed information on the whole process of system performance.

4.6.1 Detailed Performance Evaluation of the Architecture

4.6.1.1 Design Architecture

Hyperledger Fabric is a permissioned modular architecture allowing system customization to specific security requirements [29]. It is a framework where applications are written using standard programming languages which are platform-independent [16]. Fabric architecture follows the execute-order-validate model of transactions processing. The model separates the flow of transactions into three steps; execution phase of transactions through endorsement, ordering phase of transactions, and validation phase. The architectural design of Fabric addresses challenges of security, resiliency, flexibility, and scalability faced by Ethereum. Ethereum is a permissionless order-execute architecture blockchain with consensus protocol based on proof of work. Ethereum requires applications to be written in a specific domain of languages.

4.6.1.2 Built-in Support Architecture for Data Privacy

Fabric architecture deploys a private data policy that uses private data collection [30]. The actual private data is stored in a private state database that holds the current values of ledger states. Private data is accessed through chaincodes. A hash of the data stored is written to each node with the access rights to the private data leading to data privacy. Private data is also referred to be off-chain data or off-chain transactions. The off-chain data storage approach is used by Ethereum to address data storage challenges while Fabric network uses a private data approach for addressing data privacy challenges. The storage of Ethereum's private data is outside the platform while with Fabric, the storage is held within the framework.

4.6.1.3 Data Storage Capacity

Data storage implementation of the developed system is through private data collection to guarantee privacy as well as minimize storage capacity. Data is held within a database of a Hyperledger Fabric platform and managed with a private data policy [30]. This led to a single storage device compared to deployment of all nodes for data storage which has much consumption of storage space.

4.6.1.4 Cost Implications

Cloud storage is one of the approaches to avoid large data set challenges. Using only one service provider for cloud storage service behaves like the centralized system architecture. This creates a single point of system attack and failure in case of any network vandalism to the service provider. Cloud storage requires renting to several service providers to avoid the risk of central storage. This approach is used for maintaining data availability but it has cost implications as compared to the developed system approach which used a single storage device within the Hyperedger Fabric.

4.6.1.5 System Confidentiality

Hyperledger Fabric is a private permissioned blockchain system requiring its users to be granted permission to join and connect to the network. The system can be designed into sub-channels allowing the same nodes to participate in other multiple channels at the same time guaranteeing data storage confidentiality. Ethereum is a permissionless blockchain system where anyone can join the network, interact with the system ledger and access stored data. No data privacy since whatever is stored in the

blockchain system is visible to all network members [16, 22].

4.6.1.6 Computational Power

The process of transaction validation in Hyperledger Fabric is relatively quick compared to Ethereum. This is due to Fabric's low computation power resulting from low cost and low latency of transactions. Hyperledger Fabric transactions are signature-based, while Ethereum transactions are not. Ethereum validation process has much high computation power, due to its protocol which involves computation of complex mathematical calculations for the addition of a block to the chain [18, 22, 26]. Transaction validation can be reversed, becoming wasteful consumption of resources if the miners fail to successful add a block in a chain. This is contrary to Fabric where transactions are irreversible.

The cost of computational power is lowered in the Fabric network due to sharing of validation processes across the network rather than leaving the whole task to specific organizations. Transaction proposal is endorsed by endorsing peers and sent to ordering nodes for ordering service. After ordering the service, transactions will be validated by all peers involved in a respective transaction. The suffering of a specific single set of computation processes is removed. This causes a reduced computational or latency burden during smart contract processing [31]. It is a very different process in the Ethereum network where only miners bear the whole cost of transaction processing leading to high computational power.

4.6.1.7 Transaction Ordering and Validation

The fabric has an ordering service for ordering a batch of transactions and distributes them to the validating peers on the network. The ordering service does not either access ledger transactions or validate transactions. Its main task is to order transactions to be validated and committed to the ledger by responsible peers. Each organization runs an ordering service to avoid the responsibility of a single organization to create and distribute blocks in a chain. This approach addresses several security challenges faced by Ethereum including computation, this is because the creation and distribution of blocks in Ethereum is the responsibility of a single organization leading to high computation power and inconsistency [16].

4.6.1.8 Modularity, Plug and Play Components

Fabric's designs architecture enables configuration in multiple ways which lead to innovation and optimization that satisfies solution requirements. Fabric supports the use of smart contracts for general-purpose programming languages without constraining to a specific language. Pluggable consensus protocol made it to be effective for customization to specific user requirement models. Its modularity led to the high performance of consensus services and support of various database management systems. This addressed challenges such as confidentiality, performance, scalability, flexibility and resiliency faced by Ethereum [16, 26].

4.6.2 Overall Performance of Hyperledger Fabric Architecture

The performance measurement process of blockchain architecture was carried out through experimentation of a performance monitoring framework for blockchain systems. The process was facilitated through a combination of blockchain data and the consumption of computing recourses. Performance metrics used were; transactions per second (TPS), transactions per CPU (TPC), transactions per memory speed (TPMS), transactions per disk input/output (TPDIO), and transactions per network data (TPND). These metrics enabled the discovery of different blockchain systems' throughput and latency [32].

4.6.2.1 Transaction Per Second (TPS)

This metric measures the throughput of the transaction during its execution. The metric shows the number of transactions (Txs) executed in a given period from time t_p to t_q. TPS of the peer (p) can be calculated with the formula:

$$\mathrm{TPS_p} = \frac{\mathrm{Count(Txs\ from\ (tp,tq))}}{\mathrm{tq-tp}}\left(\frac{\mathrm{Txs}}{\mathrm{s}}\right) \quad (1)$$

from this formula, the average TPS for (P) peers in a network is:

$$\overline{\mathrm{TPS}} = \frac{\sum_{\mathrm{p}} \mathrm{TPS_p}}{\mathrm{P}} = \left(\frac{\mathrm{Txs}}{\mathrm{s}}\right) \quad (2)$$

4.6.2.2 Transaction Per CPU (CPU)

This is the measurement of CPU resources consumption during smart contract execution. The level of CPU consumption depends on the smart contract's business logic. Smart contracts with encryption and looping series consume much CPU resources. Utilization of CPU is highly noticed during transaction consensus life cycle while validation and committing of blocks. From t_p to t_q, the TPC of the peer (p) is computed as follows:

$$\mathrm{TPC_p} = \frac{\mathrm{Count\ (Txs\ from\ (t_p,t_q))}}{\int_{t_p}^{t_q} \mathrm{F*CPU(t)}}\ (\mathrm{txs/(GHz.s)}), \quad (3)$$

F stands for a CPU core, and CPU(t) denotes blockchain CPU usage from t_p to t_q . The average utilization of CPUs in the network is computed by:

$$\overline{\mathrm{TPC}} = \frac{\sum_{\mathrm{p}} \mathrm{TPC_p}}{\mathrm{P}} = (\mathrm{txs/(GHz.s)}), \quad (4)$$

4.6.2.3 Transaction Per Memory Second (TPMS)

This is the measurement of memory consumption during smart contract execution. Memory utilization during the execution of transactions (Txs) from t_p to t_q is computed using this formula:

$$\mathrm{TPMS_p} = \frac{\text{Count (Txs from } (t_p, t_q))}{\int_{t_p}^{t_q} \mathrm{RMEM(t)+VMEM(t)}} (\mathrm{txs/(MB.s)}), \quad (5)$$

RMEM(t) is the physical memory utilized by the blockchain system at time t_p to t_q, and VMEM is the virtual memory. The average memory utilization can be calculated with the following formula: $\overline{\mathrm{TPMS}} = \frac{\sum_p \mathrm{TPMS_p}}{\mathrm{P}} (\mathrm{txs/(MB.s)}), \quad (6)$

4.6.2.4 Transactions Per Disk I/O (TPDIO)

The measurement represents the utilization of blockchain I/O resources. The processes such as block committing and contract execution consume I/O resources during ledger state maintenance.

$$\mathrm{TPDIO_p} = \frac{\text{Count (Txs from } (t_p, t_q))}{\int_{t_p}^{t_q} \mathrm{DISKR(t)+DISKW(t)}} (\mathrm{txs/kilobytes}), \quad (7)$$

DISKR(t) shows the amount of data read from the storage disk and DISKW(t) shows the amount of data written to the storage disk from time t_p to t_q. The consumption of disk resources by all peers (P) in the network can be computed as follows: $\overline{\mathrm{TPDIO}} = \frac{\sum_p \mathrm{TPDIO_p}}{\mathrm{P}} (\mathrm{txs/kilobytes}), \quad (8)$

4.6.2.5 Transactions Per Network Data (TPDN)

This is the measurement of blockchain system network flow utilization from time t_p to t_q. The metric measures network data flow for transactions (Txs) in kb. The TPND of the peer (p) in a network is computed as folows:

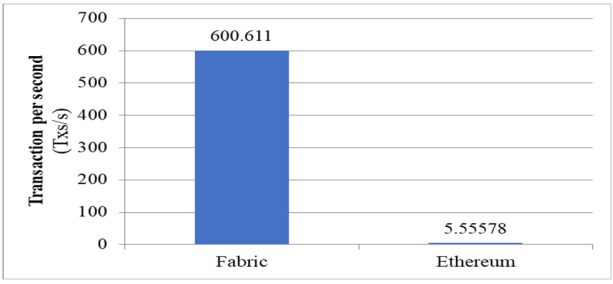
$$\mathrm{TPND_p} = \frac{\text{Count } (\text{Txs from } (t_p, t_q))}{\int_{t_p}^{t_q} \mathrm{UPLOAD(t)+DOWNLOAD(t)}} \left(\frac{\mathrm{txs}}{\mathrm{kilobytes}}\right), \quad (9)$$

UPLOAD(t) denotes the upstream size of the network and DOWNLOAD(t) denotes downstream size from time t_p to t_q. Average computation of the whole network flow can be obtained by: $\overline{\mathrm{TPDN}} = \frac{\sum_p \mathrm{TPDN_p}}{\mathrm{P}} \left(\frac{\mathrm{txs}}{\mathrm{kilobytes}}\right), \quad (10)$

4.6.3 Assessment of Overall Performance Metrics Results

Fig. 7 Computed average transactions per second.

This section assesses metric results of overall architectural performance based on experimentation for comparison between Fabric and Ethereum blockchain platforms. Nodes



with Intel Core i7-4790 3.60GHz CPU and 8GB RAM were used with 1000 smart contracts [32].

Fig. 7 illustrates the assessments transactions of Fabric and Ethereum in a second. The results show that the average throughput of Fabric is higher than Ethereum. This is an indication that Fabric architecture has a higher transactions rate per second compared to Ethereum.

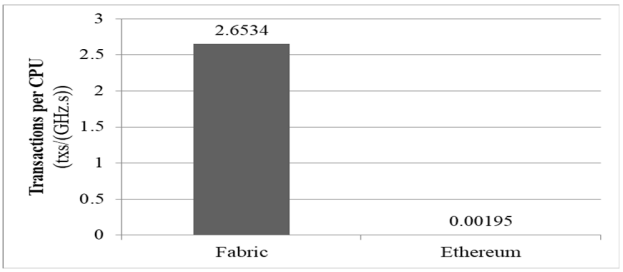


Fig. 8 Computed average transactions per CPU.

Fig. 8 shows Fabric and Ethereum utilization of one gigahertz CPU core for each node in a second. The results show that Ethereum's utilization of processors is very low compared to Fabric.

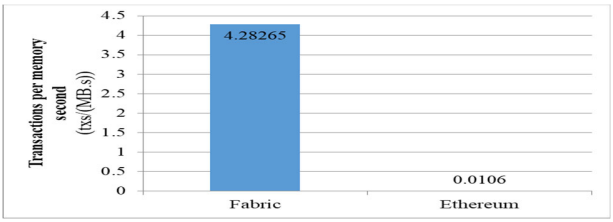


Fig. 9 Computed average transactions per memory second.

Fig. 9 illustrates transactions of Fabric and Ethereum consumption of computer memory per unit time. The results show that the Fabric system consumes more than four transactions per 1 megabyte per second of a peer's memory.

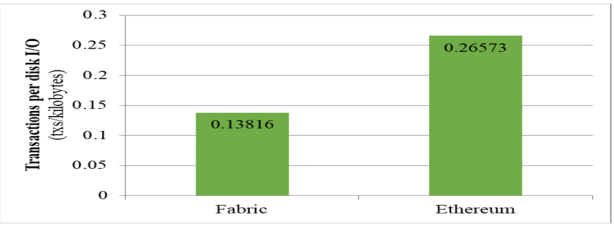


Fig. 10 Computed average transactions per disk read/write.

Fig. 10 illustrates transactions of Fabric and Ethereum applications for reading and writing of 1 megabyte of a peer per unit time to/from a peer's disk storage. The results show that Ethereum has higher reads and writes transactions for 1 Mb per second compared to Fabric.

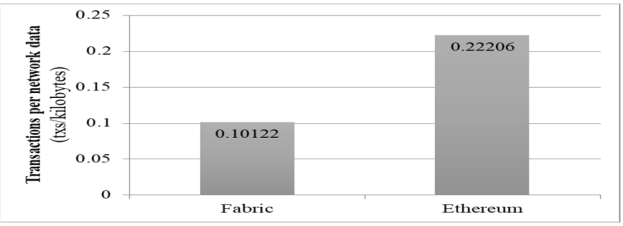


Fig. 11 Computed average of transactions network data.

Fig. 11 displays computed transactions of a network data flow in a second. The results shows that Fabric's transactions per network data are lower than Ethereum. Ethereum consumes half the bandwidth of the Fabric system.

5. Discussion

The study aimed to develop a blockchain-based system with secured data storage architecture to address cybersecurity storage challenges. The study developed the system that was integrated with the existing system to address storage security issues. In addition to security issues addressed by Fabric architecture, the overall architectural performance of the execute-order-validate model shows that Fabric has higher throughput than Ethereum. This is because the architectural design of Fabric is purposely designed to be permissioned blockchain. The architecture can also be used for private data storage and its consensus protocol is much faster. Ethereum's throughput is very low due to its architectural consensus protocol which has a high consumption of computing resources while computing hashes. Therefore, Hyperledger Fabric blockchain architecture shows higher overall performance compared to Ethereum.

6. Conclusion and Recommendations

This study focused on data storage security architecture. The findings of the study show that, the Execute-Order-Validate architectural design of Hyperledger Fabric blockchain architecture provides secured data storage, with higher overall performance compared to Execute-Validate Ethereum blockchain architecture. Fabric's modular architecture provided separation of transaction flow from execution phase to validation phase leading to secured transactions with high throughput. Challenges of big data sets and data privacy identified in the literature review were addressed through private data collection. Data storage is maintained within the Hyperledger Fabric framework with private data policy, and the nodes were decentralized with blockchain only.

It is advised that decision-makers, the health care industry, and other researchers make use of these findings. Frequent training should be given to system administrators for increasing their knowledge and awareness of cyber-security issues. This will help them to be updated on new emerging security technologies and stay current on all issues related to information system security. Further studies will be on the assessment of Hyperledger Fabric consensus protocols.

References

[1] S. Biswas, K. Sharif, F. Li, and S. Mohanty, "Blockchain for e-health-care systems: Easier said than done," *Computer,* vol. 53, pp. 57-67, 2020.

[2] S. Shamshad, K. Mahmood, S. Kumari, and C.-M. Chen, "A secure blockchain-based e-health records storage and sharing scheme," *Journal of Information Security and Applications,* vol. 55, p. 102590, 2020.

[3] W. Liu, S. Zhu, T. Mundie, and U. Krieger, "Advanced block-chain architecture for e-health systems," in *2017 IEEE 19th International Conference on e-Health Networking, Applications and Services (Healthcom)*, 2017, pp. 1-6.

[4] H. Wang and J. Zhang, "Blockchain Based Data Integrity Verification for Large-Scale IoT Data," *IEEE Access,* vol. 7, pp. 164996-165006, 2019.

[5] K. Tsoulias, G. Palaiokrassas, G. Fragkos, A. Litke, and T. A. Varvarigou, "A Graph Model Based Blockchain Implementation for Increasing Performance and Security in Decentralized Ledger Systems," *IEEE Access,* vol. 8, pp. 130952-130965, 2020.

[6] Y. Sun, L. Zhang, G. Feng, B. Yang, B. Cao, and M. A. Imran, "Blockchain-enabled wireless Internet of Things: Performance analysis and optimal communication node deployment," *IEEE Internet of Things Journal,* vol. 6, pp. 5791-5802, 2019.

[7] C. Kombe, A. Sam, M. Ally, and A. Finne, "Blockchain Technology in Sub-Saharan Africa: Where does it fit in Healthcare Systems: A case of Tanzania," *Journal of Health Informatics in Developing Countries,* vol. 13, 2019.

[8] X. Zheng, R. R. Mukkamala, R. Vatrapu, and J. Ordieres-Mere, "Blockchain-based personal health data sharing system using cloud storage," in *2018 IEEE 20th International Conference on e-Health Networking, Applications and Services (Healthcom)*, 2018, pp. 1-6.

[9] A. Shahnaz, U. Qamar, and A. Khalid, "Using blockchain for electronic health records," *IEEE Access,* vol. 7, pp. 147782-147795, 2019.

[10] R. Akkaoui, X. Hei, and W. Cheng, "EdgeMediChain: A Hybrid Edge Blockchain-Based Framework for Health Data Exchange," *IEEE Access,* vol. 8, pp. 113467-113486, 2020.

[11] M. Madine, K. Salah, R. Jayaraman, I. Yaqoob, Y. Al-Hammadi, S. Ellahham*, et al.*, "Fully Decentralized Multi-Party Consent Management for Secure Sharing of Patient Health Records," *IEEE Access,* 2020.

[12] S. Biswas, K. Sharif, F. Li, I. Alam, and S. Mohanty, "DAAC: Digital Asset Access Control in a Unified Blockchain Based E-Health System," *IEEE Transactions on Big Data,* 2020.

[13] R. Kumar, N. Marchang, and R. Tripathi, "Distributed off-chain storage of patient diagnostic reports in healthcare system using IPFS and blockchain," in *2020 International Conference on COMmunication Systems & NETworkS (COMSNETS)*, 2020, pp. 1-5.

[14] T. Hepp, M. Sharinghousen, P. Ehret, A. Schoenhals, and B. Gipp, "On-chain vs. off-chain storage for supply-and blockchain integration," *it-Information Technology,* vol. 60, pp. 283-291, 2018.

[15] T. S. L. Nguyen, G. Jourjon, M. Potop-Butucaru, and K. L. Thai, "Impact of network delays on Hyperledger Fabric," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2019, pp. 222-227.

[16] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro*, et al.*, "Hyperledger fabric: a distributed operating system for permissioned blockchains," in *Proceedings of the thirteenth EuroSys conference*, 2018, pp. 1-15.

[17] H. Javaid, C. Hu, and G. Brebner, "Optimizing validation phase of hyperledger fabric," in *2019 IEEE 27th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*, 2019, pp. 269-275.

[18] Y. Manevich, A. Barger, and Y. Tock, "Service discovery for hyperledger fabric," in *Proceedings of the 12th ACM International Conference on Distributed and Event-Based Systems*, 2018, pp. 226-229.

[19] S. ah Khan, A. Jadhav, I. et Bharadwaj, M. Rooj, and S. Shiravale, "Blockchain and the Identity based Encryption Scheme for High Data Security," in *2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC)*, 2020, pp. 1005-1008.

[20] C. Wang and X. Chu, "Performance characterization and bottleneck analysis of hyperledger fabric," in *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*, 2020, pp. 1281-1286.

[21] L. Ismail and H. Materwala, "A review of blockchain architecture and consensus protocols: Use cases, challenges, and solutions," *Symmetry,* vol. 11, p. 1198, 2019.

[22] M. Saad, J. Spaulding, L. Njilla, C. Kamhoua, S. Shetty, D. Nyang*, et al.*, "Exploring the attack surface of blockchain: A systematic overview," *arXiv preprint arXiv:1904.03487,* 2019.

[23] L. S. Sankar, M. Sindhu, and M. Sethumadhavan, "Survey of consensus protocols on blockchain applications," in *2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS)*, 2017, pp. 1-5.

[24] M. K. Mustafa and S. Waheed, "An E-Voting Framework with Enterprise Blockchain," in *Advances in Distributed Computing and Machine Learning*, ed: Springer, 2021, pp. 135-145.

[25] H. Honar Pajooh, M. Rashid, F. Alam, and S. Demidenko, "Hyperledger Fabric Blockchain for Securing the Edge Internet of Things," *Sensors,* vol. 21, p. 359, 2021.

[26] J. Sousa, A. Bessani, and M. Vukolic, "A byzantine fault-tolerant ordering service for the hyperledger fabric blockchain platform," in *2018 48th annual IEEE/IFIP international conference on dependable systems and networks (DSN)*, 2018, pp. 51-58.

[27] H.-Y. Paik, X. Xu, H. D. Bandara, S. U. Lee, and S. K. Lo, "Analysis of Data Management in Blockchain-Based Systems: From Architecture to Governance," *IEEE Access,* vol. 7, pp. 186091-186107, 2019.

[28] V. Ribeiro, R. Holanda, A. Ramos, and J. J. Rodrigues, "Enhancing key management in LoRaWAN with permissioned blockchain," *Sensors,* vol. 20, p. 3068, 2020.

[29] P. Sajana, M. Sindhu, and M. Sethumadhavan, "On blockchain applications: hyperledger fabric and ethereum," *International Journal of Pure and Applied Mathematics,* vol. 118, pp. 2965-2970, 2018.

[30] S. Brotsis, N. Kolokotronis, K. Limniotis, G. Bendiab, and S. Shiaeles, "On the security and privacy of hyperledger fabric: Challenges and open issues," in *2020 IEEE World Congress on Services (SERVICES)*, 2020, pp. 197-204.

[31] A. A. Monrat, O. Schelén, and K. Andersson, "A survey of blockchain from the perspectives of applications, challenges, and opportunities," *IEEE Access,* vol. 7, pp. 117134-117151, 2019.

[32] P. Zheng, Z. Zheng, X. Luo, X. Chen, and X. Liu, "A detailed and real-time performance monitoring framework for blockchain systems," in *2018 IEEE/ACM 40th International Conference on Software Engineering: Software Engineering in Practice Track (ICSE-SEIP)*, 2018, pp. 134-143.