UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

LAURA RODRIGUES SOARES

Design of a Blockchain-based Secure Storage Architecture for Resource-Constrained Healthcare

Thesis presented in partial fulfillment of the requirements for the degree of Master of Computer Science

Advisor: Prof. Dr. Jéferson Campos Nobre

Porto Alegre
June 2024

CIP — CATALOGING-IN-PUBLICATION

Soares, Laura Rodrigues

Design of a Blockchain-based Secure Storage Architecture for Resource-Constrained Healthcare / Laura Rodrigues Soares. – Porto Alegre: PPGC da UFRGS, 2024.

61 f.: il.

Thesis (Master) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2024. Advisor: Jéferson Campos Nobre.

1. Blockchain. 2. Internet of Things. 3. Decentralized storage. 4. IPFS. 5. Healthcare. 6. Security. I. Campos Nobre, Jéferson. II. Título.

ACKNOWLEDGEMENTS

Essa seção de agradecimentos começa com a minha família, principalmente meus pais, Isabel e Marcos. Durante esse período de mestrado, eu tive o privilégio de contar com o apoio deles em vários aspectos – e esse apoio foi fundamental tanto para minha permanência no curso como para sua conclusão. Em particular eu gostaria de agradecer o meu avô, Antônio Rodrigues, que faleceu durante esse período. Meu avô não terminou o ensino básico, mas viu todos os três filhos e três das netas se formando no ensino superior – alguns no pós-superior, inclusive. Foi ele quem me ajudou a decidir qual caminho seguir naquele período de final de graduação complicado, com direito a pandemia e muita incerteza. Fico triste que ele não tenha me visto concluir o mestrado, mas ao mesmo tempo sei que ele não tinha dúvidas de que daria tudo certo. Preciso agradecer também aos meus amigos, principalmente o Leo, a Mari, a Catarina, o Marcelo e a Bri. Nos períodos de caos em que me senti mal por não ter tudo sob controle eles foram as pessoas em quem pude me apoiar – e também me ajudaram a perceber que não tá fácil pra ninguém. Por último, gostaria de agradecer a todo o pessoal do Grupo de Redes do Instituto de Informática da UFRGS, em particular ao meu orientador, o prof. Jéferson Nobre. Eu e o prof. Jéferson trabalhamos juntos há algum tempo, desde 2019, e nós sempre nos comunicamos com muita facilidade – o que pra mim significa muito. Caso meu orientador fosse outro, minha gastrite nervosa certamente não teria aguentado dois anos de mestrado. Muito obrigada por tudo, pessoal, e vamos em frente.

ABSTRACT

The Internet of Things (IoT) paradigm can improve a broad range of applications, such as medical sensors, smart cities, industrial monitoring, and so on. In healthcare specifically, these devices can aid in management tasks and improve the quality of life of patients in intensive care. However, securing medical IoT devices and its data is a crucial task. Not only do they handle Electronic Medical Records (EMR) and physiological information, but in some cases, disruption can affect a patient's treatment. Blockchain technologies applied in the healthcare context can provide privacy, immutability, decentralization, and easier access and sharing of medical data. Despite the emergence of applications aiming to solve security issues in the Healthcare IoT (HIoT) scenario using blockchain, there is still much to be addressed, mainly regarding throughput, storage of data, and efficient use of resources. This work proposes a blockchain-based storage architecture for HIoT, using a private blockchain network and distributed data storage to achieve integrity, accountability, and availability of medical data. We introduce a vault component to the off-chain and distributed storage model to avoid storing the address of medical files directly on-chain, adding an extra layer of security and privacy to the system. We evaluate the performance of each system component and reach feasible latency and throughput results for the desired use-case scenario.

Keywords: Blockchain. Internet of Things. Decentralized storage. IPFS. Healthcare. Security.

Implementação de uma Arquitetura de Armazenamento Seguro Baseada em Blockchain para Ambientes Restritos em Saúde

RESUMO

Sistemas IoT (do inglês Internet of Things, ou Internet das Coisas) trazem melhorias para diversas aplicações como sensores médicos, cidades inteligentes, monitoramento industrial, entre outras. Na saúde especificamente, esses dispositivos podem auxiliar em tarefas administrativas e melhorar a qualidade de vida de pacientes em cuidado intensivo. Porém, a segurança de aparelhos médicos de IoT e de seus dados é fundamental. Não apenas eles lidam com registros médicos como também, em alguns casos, a interrupção do serviço pode ser prejudicial para o tratamento dos pacientes. Redes blockchain aplicadas em contextos médicos podem fornecer privacidade, imutabilidade, e decentralização, assim como facilitar o acesso e compartilhamento de dados médicos. Apesar da emergência de aplicações buscando resolver questões de segurança em sistemas de IoT em saúde usando blockchain, ainda há muito para ser estudado considerando taxa de rendimento, armazenamento de dados, e uso eficiente de recursos. Esse trabalho propõe uma arquitetura de armazenamento para IoT em saúde baseada em blockchain, usando redes privadas e armazenamento de dados distribuído para oferecer integridade, responsabilização, e disponibilidade de dados médicos. Nós introduzimos um arquivo-cofre na arquitetura para evitar armazenar o endereço de arquivos médicos diretamente na blockchain, adicionando uma camada extra de segurança e privacidade ao sistema. Nós avaliamos a performance de cada componente e alcançamos resultados de taxa de rendimento e latência convenientes para o uso de caso desejado.

Palavras-chave: Blockchain, Internet das Coisas (IoT), Armazenamento distribuído, IPFS, Saúde, Segurança.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

CA	Certificate Authority
CID	Content Identifier
DAG	Direct Acyclic Graph
DHT	Distributed Hash Table
EMR	Electronic Medical Record
GDPR	General Data Protection Regulation
IoT	Internet of Things
IPFS	Interplanetary File System
HIoT	Helthcare Internet of Things
HTTP	Hypertext Transfer Protocol
LGPD	Lei Geral de Proteção de Dados
MSP	Membership Service Provider
PBFT	Practical Bizantine Fault Tolerance
PHI	Personal Health Information
PKI	Public Key Infrastructure
PoS	Proof-of-Stake
Pow	Proof-of-Work
RPC	Remote Procedure Call
TPS	Transactions Per Second

CONTENTS

1 INTRODUCTION

The Internet of Things (IoT) paradigm has been successfully applied to numerous applications in the past few decades, such as industrial monitoring, agriculture, smart cities, and others. Healthcare IoT (HIoT) sensors (e.g., ECG and blood pressure monitors, oximeters, and so on) can provide rapid feedback on patients in medical emergencies (GHUBAISH et al., 2020) and facilitate the monitoring and treatment of patients outside conventional healthcare settings (RAY et al., 2021). They can also aid the medical team in management tasks, such as keeping track of patients, equipment, and medication. Most of the physiological data these sensors acquire can be reused as biometric signatures in authentication tasks, providing health diagnosis and security improvement at the same time (HUANG et al., 2019). However, HIoT systems come with specific challenges. To name a few, the applications must be able to deal with hundreds of devices of varying capacities from diverse manufacturers requiring different types of management. Also, the volume of data they provide demands data aggregation techniques to achieve scalability in institution-wide deployments. Finally, from a security standpoint, special attention is due to health-related applications dealing with Electronic Medical Records (EMRs, e.g., medical examination and diagnosis data) and patient physiological information. Compliance legislation and significant privacy concerns demand a robust framework capable of providing the necessary security properties.

There is a growing interest in using decentralized health monitoring systems to solve traditional security issues of the HIoT industry. Public blockchain networks have brought distributed ledger technologies into attention after extensive use in cryptocurrency systems, and have recently expanded to new areas after a wave of interest on the topic (JAOUDE; SAADE, 2019). IoT and healthcare, in particular, can benefit from decentralization, transparency, and immutability, all of which are inherently provided by the blockchain. However, the integration between blockchain and HIoT must be handled carefully. In addition to the standard challenges of health applications, IoT blockchain-based applications have to deal with storage limits, insufficient computational power to constantly validate transactions, and high throughput demands (RAY et al., 2020). Compliance and data ownership issues demand special attention since all transaction data written on the blockchain is permanent and cannot be erased. Therefore, several solutions aim for a hybrid of "on-chain" and "off-chain" storage to handle medical and personal data. The blockchain provides indexing, authentication, and record-keeping while other tech-

nologies can provide data transfer and storage (ALI et al., 2020). Any security solution for HIoT systems must carefully address these concerns to employ blockchain networks advantageously.

Blockchain-based solutions for HIoT experienced a rise over the last few years. Zhang et al. (2022), for instance, proposed a blockchain-based medical data sharing framework that efficiently handles the decryption key infrastructure (tasks such as the generation, verification, and distribution of keys, key-based search, and so on) while dealing with untrusted cloud storage providers. Chen et al. (2021) developed a blockchain system for medical data collection, storage, and sharing, designed to work across multiple hospitals and third-party organizations. Akkaoui, Hei and Cheng (2020) used a combination of public and private blockchain networks for indexing medical files while using a distributed off-chain system for storage. However, despite the surge of works on the topic, there is still room for improvement in several topics related to blockchain usage and the integration with the storage system. While promising, the integration between the blockchain-as-an-index and the storage system can affect the system's privacy, specially if any network participant can have access to the on-chain address of EMRs. In addition, the amount of computational resources demanded by the blockchain, both in terms of processing capabilities and energy consumption, is far from negligible and should be optimized as much as possible to meet the target HIoT application requirements of resource constraints. Finally, assessing the performance the existing algorithms is necessary to justify the use of blockchain in favor of traditional alternatives that would achieve the same security properties demanding far less use of resources.

Considering the current challenges in integrating blockchain into HIoT systems, this work aims to propose a secure and decentralized storage architecture for healthcare applications. The data is stored off-chain in a decentralized, peer-to-peer storage system. The architecture employs blockchain for indexing data from patients, but includes an extra step between the index and the distributed storage. Instead of keeping the address of every EMR directly on the ledger, the system introduces a vault component to keep a list of all the addresses of medical files associated with a patient. This vault file is encrypted and its address is kept on-chain. The use-cases of the proposed architecture includes handling both patient EMRs and data provided by IoT sensors. It provides confidentiality, integrity of records, and access control in a decentralized, multi-authority environment. In addition, this work presents a brief background of the related areas of HIoT and blockchain-based systems, a detailed overview of the related works on the topic, and the key concepts

behind the important architectural components of the system.

The remaining of this work uses the following structure. Chapter 2 presents key concepts of IoT systems and blockchain. Chapter 3 has an overview of related state-of-art systems and their architectural choices. Chapter 4 outlines our solution's requirements and desired security properties, and introduces the proposed architecture. Chapter 5 presents the frameworks used for the proof-of-concept implementation, and the results of experiments and performance evaluation are in Chapter 6. Finally, Chapter 7 presents the next steps in the development of this system and concludes this dissertation.

2 BLOCKCHAIN AND IOT FOR HEALTH APPLICATIONS

This chapter presents the key concepts of all the areas concerned in this dissertation. First, we discuss the architecture commonly found in IoT systems. Then, we present the blockchain concepts necessary for the understanding of this work. We then cover how blockchain networks can improve traditional healthcare systems, and finally, we combine all of the previous to discuss common requirements of blockchain-based IoT systems for healthcare.

2.1 IoT Architecture

Nowadays, IoT is a pervasive area with a broad range of use cases. It is widely used in the agricultural industry, Smart City appliances, industrial monitoring, and several others. The applicability of IoT in the healthcare environment is also extensive. Wearable sensors can be small and unobtrusive, offering data such as body temperature, heart rate, blood oxygen levels, etc. More robust sensors can offer more detailed information like the electrocardiogram (ECG), electroencephalogram (EEG), and others (PINTO; CARDOSO; LOURENÇO, 2018). This data can aid, for example, in the remote care for the elderly, long-term monitoring of cardiac patients, sleep apnoea studies, and so on. A functional, secure HIoT framework means safeguards to patients in case of emergencies, more independence, and general quality-of-life improvement. HIoT can also be employed in administrative tasks, such as location tracking of medical equipment and medication monitoring, offering overall convenience to the medical team (HE; ZEADALLY, 2014).

Most IoT systems follow similar architectural patterns. The key components are the sensor devices, the gateway nodes, and the Application Provider. The sensor devices in a single network usually have a broad range of constraints for power, memory, and processing capabilities. Memory size can be used as a rough estimator of device capabilities. A device is considered constrained if it ranges around 250 KiB of code size, following the estimation in RFC 7228 (BORMANN; ERSUE; KERANEN, 2014). The gateway nodes, in turn, handle the communication with the sensor nodes. These nodes can also range in capacity, from a Raspberry Pi to the user's smartphone to an actual desktop computer, but they are usually more computationally capable than the sensors. Finally, the application provider can be, e.g., a health service available to patients and medical staff on a remote server. This application is often also in charge of storing the acquired data using either

traditional centralized data centers, cloud services, or decentralized storage solutions.

Security solutions for HIoT have complex requirements. Most sensor devices have low-capacity hardware and no computational power to spare for expensive cryptographic procedures. In addition, a single HIoT network usually is comprised of several different sensors from different vendors. This heterogeneity of devices and protocols is a substantial challenge to be handled by a single security solution. Furthermore, sensors transmit measured data at small intervals and generate a considerable amount of data, which demands a scalable solution capable of handling it. Availability is also critical in some scenarios since service disruption may harm patients in continuous care. Finally, handling physiological EMRs from patients demands strong privacy requirements. In Europe, the use of any personal data from users must respect the European Union's General Data Protection Regulation (GDPR). In Brazil, this use is regulated by the *Lei Geral de Proteção de Dados* (LGPD).

HIoT systems often struggle to meet all the desired security properties and provide scalability. Blockchain is one of the potential solutions to traditional HIoT systems, aiming to improve security and facilitate access to medical data.

2.2 Key Blockchain Concepts

Blockchain networks came gradually into attention after the proposal of the first public blockchain with the Bitcoin cryptocurrency in 2008 (NAKAMOTO, 2008). The main appeal of a public blockchain for use in cryptocurrency is to provide an anonymous, decentralized environment where no third party is required to oversee transactions between two concerned parties (JAOUDE; SAADE, 2019). In a traditional financial system the central authority is responsible not only for the authentication of all parties but also for ordering the transactions, ensuring there is no risk of double spending in case of repeating the same operation. In order to replicate this functionality, the public blockchain keeps a distributed shared ledger updated by a community of peer nodes. The use of cryptography and asymmetric keys provides anonymity to the parties.

There are three types of blockchain networks more commonly used in the literature. They are public, private, and consortium. The appropriate type will depend on each application scenario. In public blockchains, any user in the network can issue transactions or participate as a peer node in the validation process. This model usually offers monetary rewards to the nodes as an incentive in the mining process to reach consensus.

The Bitcoin cryptocurrency is an example of a public blockchain. Private (or permissioned) blockchains require the users to be previously enrolled in the system before they can propose transactions and act as validators, which makes this model more suitable for applications handling sensitive data (RAY et al., 2020). The Ethereum protocol, for example, can be used to set up both public and private blockchain networks. Finally, the consortium blockchain is a type of permissioned blockchain in which the network is operated by a group of organizations instead of a single centralized entity. The organizations manage the infrastructure and enroll the trusted nodes capable of validating transactions. The Fabric network from the Hyperledger Foundation is an example of a consortium blockchain widely mentioned in the literature. Some components of blockchain networks are detailed as follows:

- **Distributed ledger:** the ledger is the public record that stores all the transactions already executed. Each transaction and all related information is stored in a block, together with a cryptographic hash of the previous block. Every node in the network has a local copy of the chain and uses the previous blocks to validate new blocks.
- **Smart contract:** is a computerised transaction protocol very popular in blockchain environments (ZHENG et al., 2018). In a blockchain network smart contracts can be used as the set of rules based on which the peer nodes will operate. It can define standard structures of data, operations to handle it, rules regarding proper authorization, and so on. The participants must previously agree on a common contract before it can be deployed in the network.
- **Nodes:** the peer nodes are the participants of the network capable of using a particular consensus algorithm to verify and validate the new transactions, store the results in a block, and append it to the chain (RAY et al., 2020). In some blockchain deployments the validating process is decoupled in small intermediate steps, which allows for groups of nodes of varying capacities and different levels of permission to operate separate roles in the network. Once all the nodes participating in validation tasks confirm the new transactions, it is said the ledger has reached consensus.
- **Consensus algorithm:** there are several algorithms the network can use to reach consensus, each applicable for different blockchain types and use cases. Some consensus algorithms designed for public chains reward the nodes for checking the correctness of the new blocks in a process called mining. The Proof-of-Work (PoW) algorithm from the original bitcoin, for example, has a mining process consisting in the nodes trying to be the first to solve a mathematical puzzle in exchange for

monetary incentive. It is widely known for the vast amounts of wasted energy (YLI-HUUMO et al., 2016). Proof-of-Stake (PoS) surged as a more energy-friendly solution, in which the node to mine the next block is chosen by a lottery system, with no monetary reward system (RAY et al., 2020). PoS is currently used by the Ethereum cryptocurrency (ETHEREUM.ORG, 2023). As for algorithms suitable for private chains, the Practical Byzantine Fault Tolerance (PBFT) is capable of handling byzantine faults and the presence of up to 1/3 malicious nodes (ZHENG et al., 2018). Since the network knows every participating node there is no mining process, and the algorithm saves energy greatly. The Fabric network is currently working in their own implementation of PBFT (HYPERLEDGER FOUNDATION, 2022).

After the spike of works on public blockchains inspired by its use on cryptocurrency, blockchain applications went through a generalization process to be applied in several other areas (JAOUDE; SAADE, 2019). The main features of immutability, authentication, and transparency benefit both IoT applications and the healthcare field. For that reason, these are two of the most popular emerging topics to be the target of blockchain integration.

2.3 Blockchain-based Healthcare and IoT

Interoperability problems, difficulties in properly auditing systems, and data leakage of sensitive information are some of the most common challenges to traditional health systems (SANTOS; INACIO; SILVA, 2021). Employing blockchain networks in the security system of a health facility can help in several ways. The distributed ledger can facilitate the access and sharing of medical records, as well as improve standardization efforts across different institutions (JAOUDE; SAADE, 2019). Blockchain can provide decentralization through the network of nodes, which reduces server costs and mitigate central server performance bottlenecks (ZHENG et al., 2018). Since transactions are time-stamped and broadcast to all the nodes participating in the system, it provides transparency and auditability. The network also guarantees immutability since the information stored in a block is linked with the previous block in a way that would be necessary to alter the entire chain to tamper with a block.

In particular, the immutability feature raises further discussions in a health-related

scenario. Several compliance legislations contemplate the user's right to data erasure, also known as the right to be forgotten. Article 17 of the GDPR states that users have the right to personal data erasure if the data are no longer necessary to the purpose it was collected, or upon withdrawing of consent on their usage at any time (EUROPEAN PARLIAMENT AND COUNCIL OF THE EUROPEAN UNION, 2016). The right to data erasure is also present in LGPD. Section 5(XIV) states that the data owner has the right to request the deletion of their personal information from the agent's database at any given time (NATIONAL CONGRESS OF BRAZIL, 2018). Since information stored in a blockchain's ledger cannot be erased, no Personal Health Information (PHI) should be stored directly on the ledger. A method for handling the right to be forgotten consists in using the blockchain as an index for storing exclusively non-identifiable information (such as encryption keys and logs of data access), and having the EMRs and PHI stored elsewhere.

In a HIoT system the standard architecture mentioned in Section 2.1 only needs minor modifications to integrate blockchain networks, mainly the addition of extra nodes to act as peers. These nodes can range in capacity, despite demanding considerably more computational power than the average IoT gateway. The more lightweight nodes can store only a copy of the current ledger state and issue transactions. Then, the nodes of increased computational capacity are tasked with validating new transactions and achieving consensus in the network. Depending on the system architecture, the blockchain nodes can be located at the edge of the network closer to the sensor devices. Finally, the gateway nodes communicating with sensor devices will need increased computational capability to interact with the blockchain nodes as well, for tasks such as registration, authentication, data storage, and so on. Figure 2.1 shows a generic blockchain-based HIoT architecture.

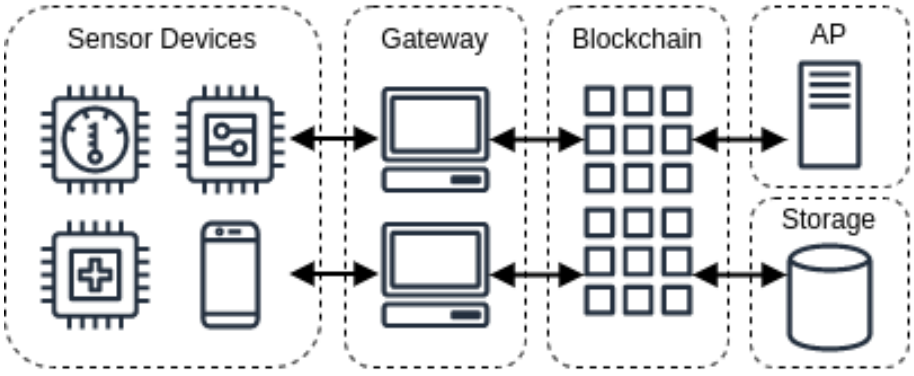


Figure 2.1 – Example of a generic architecture using both IoT and blockchain.

3 RELATED WORKS

This chapter presents a selection of works from the related literature presenting systems or schemes similar in scope to the proposed architecture. Then, we discuss the current state-of-art and present the motivation behind this dissertation.

3.1 State-of-art HIoT blockchain-based schemes

The works presented in this section are considered highly relevant for the current research on the topic of blockchain for HIoT systems. They are not meant as an exhaustive review of the literature. Each of the selected works employs blockchain networks to enhance the security of HIoT systems and offer a practical analysis of the proposed scheme. We first briefly explain the functionality of each system and the architectural components they used, and then present the performance metrics of their evaluation.

3.1.1 Xu et al. (2019)

Xu et al. (2019) opted for two separate layers of blockchain ledgers and to avoid centralized third-party cloud storage. Their goal is to support large-scale HIoT devices with satisfactory efficiency while providing user privacy, data accountability, and on-demand data access revocation. Patient data is encrypted and stored in a decentralized distributed storage system, while the hash address is indexed in the *userchain*. The *user-chain* also stores key transaction information besides health data. On the other hand, the *doctor chain* indexes diagnostic data provided by doctors. The participants of the *user-chain* can query the *doctor chain* but cannot issue transactions to it. The *doctor chain* can read from the user's, but only special authorized nodes can issue diagnostic transactions.

The authors opted for a consortium blockchain with the PBFT consensus algorithm for the doctor's ledger, and a public chain running PoW for the user's ledger. Another point is the creation of the Healthcoin as an incentive for the public mining nodes, as well as the operational decision of incorporating the Healthcoin as a global currency for the system - used to hire health insurance, pay doctors, and so on. The IPFS does the decentralized storage of patient data and diagnostics. Xu et al. (2019) evaluate the throughput of the system based on the number of transactions that could be stored in a

block, achieving 89 patient data transactions per second and 76 diagnostic transactions per second. They also measure the time it takes to generate each transaction (encryption, decryption, and hashing of the files) and compare the computation and communication costs with those of traditional systems.

3.1.2 Akkaoui, Hei and Cheng (2020)

Akkaoui, Hei and Cheng (2020) proposal incorporates edge pools to the standard blockchain-based health framework outlined in Section 2.3, bringing data processing close to the source to reduce latency and increase throughput. They store the patient's full encrypted EMR off-chain while keeping the metadata in a global blockchain. Meanwhile, physiological data being monitored by sensors is processed at a group of edge devices. These devices operate an intermediate-level blockchain separated from the global chain. Each edge device will manage tasks for geographically close IoT devices, such as authentication, accounting, and data storage. The edge pool then stores the necessary HIoT data off-chain and the corresponding metadata in the global blockchain. The metadata includes the hashed address of the content in the off-chain storage. This enables the layered blockchain system to work as an index, offering integrity of records, manageability of access rights, and ownership of health data.

The practical implementation uses Ethereum both for the global and local edge blockchains. The global public blockchain uses the PoW consensus mechanism, while the local edge chain is private and uses PoA due to the different latency and security requirements. Inter-Planetary File System (IPFS) is used for the off-chain storage. The performance evaluation assessed the execution time of each function under variating loads of concurrent transactions, as well as the average execution time and throughput of the system (several pools of edge nodes operating PoA in parallel) if compared to the standard execution of a single mining pool running the PoA consensus algorithm. The proposed system can achieve a throughput of 2.6 transactions per ms at a load of 400 concurrent transactions, which is about six times better than the comparison baseline. They also evaluated the effect of parallel processing concurrent transactions on the scalability of the ledger. They assessed that by dividing the load among the different edge pools, the system is capable of reducing the growth rate of the records.

3.1.3 Chen et al. (2021)

Chen et al. (2021) has the main goal of safe storage in semi-trusted cloud providers. Their addition to the standard blockchain-based health system architecture, as outlined in Section 2.3, is the proxy re-encryption algorithm that allows medical data to be accessed by authorized third parties and to be stored in the cloud. The system works by separating patient's data and logs of data usage in two different ledgers. The Data Usage ledger provides access control by allowing authorized data consumers to access patient's data in the first ledger, which generates a data usage record to be stored in-chain. The registry in the Patient's Data ledger includes the address for the encrypted data stored on the cloud, authorized signatures, and other metadata.

Both ledgers in Chen et al. (2021)'s system use Hyperledger Fabric, a consortium blockchain framework. It comes with its own crash fault-tolerant consensus algorithm based on Kafka and Zookeeper, introduced in Fabric v1.2. The authors designed their own HIoT sensor data collector device for evaluation. The performance analysis measured the latency for each operation of the system. They noticed that the operations related to the client application (such as encryption and decryption of files using RSA, obtaining digital signatures, and so on) are directly influenced by the size of the unencrypted medical file. However, the file size has no relevant impact on the blockchain operations, which take considerably longer - a single transaction can take up to 2.15 seconds. The evaluation of the blockchain network measured the throughput under varing request rates for different block sizes/messages per block and achieved an optimal load of 140 requests per second.

3.1.4 Ali et al. (2020)

Ali et al. (2020) main concern is to avoid sharing medical data with a third-party entity. Their solution is akin to a messaging application, in which patients are put in direct contact with doctors, and logging of the interaction is stored in the blockchain. The metadata includes timestamps and the overall size of the file exchange. Since the medical records are sent directly from one party to the other through a secure channel, their system doesn't include a storage component, and the health records are assumed to be kept individually by each patient. The on-chain indexing of the data transfer logs provides identity management, transparency, and accountability.

The system uses an Ethereum public blockchain and a Tor-based network for the

instant messaging data transfer between patient and doctor. Their evaluation used the Rinkeby Ethereum test network, meaning that no infrastructure deployment were performed by the authors. PoA is the consensus mechanism of the system, but since the application itself makes no validation, there is no measurable computational overhead. The performance analysis compared the data propagation time of their solution with existing centralized and blockchain-based messaging services. Regarding the blockchain evaluation, they issued a fixed transaction rate and measured the time it would take to complete the transaction, achieving about 41 seconds under 15 transactions per minute.

3.1.5 Mayer et al. (2021)

Mayer et al. (2021) has a similar proposal to Akkaoui, Hei and Cheng (2020), to incorporate fog computing and bring data processing closer to the data source to improve latency. The authors define fog as a layered service infrastructure similar to the cloud, with low-energy computing nodes with limited hardware. They aim to avoid using centralized cloud services for storage and use the blockchain for this goal instead. Though their model does have an option to store large files off-chain, this option is not addressed in the evaluation, so it's assumed the bulk of small-sized files generated by HIoT sensors is indeed stored on-chain.

The implementation uses the Hyperledger Fabric consortium blockchain and PBFT consensus mechanism. Their performance testing first evaluated the ideal batch size to add an array of several transactions to the blockchain. They noticed performance degradation as the size of the batch increased and were capable of achieving about 579 transactions per second under a light transaction load. The analysis focused on comparing the blockchain performance with cloud services, which they did by running an AWS virtual machine. The authors claim the increased latency of the cloud server is due to the public internet's different data traffic paths, while the blockchain sits closer to the data source. Blockchain and IoT scalability inside a constricted fog environment is not addressed.

3.1.6 Zhang et al. (2022)

While their work concerns mainly tasks related to encryption keys, such as issuing, distributing, and storing, Zhang et al. (2022) system also shares the goal of blockchain-

based safe storage of HIoT data in untrusted cloud providers. They address limitations such as key generation and verification burden, key leakage, safe storage, search, and so on. They employ Cyphertext-policy Attribute-Based Encryption (CP-ABE) to provide key search but with reduced burden in the cloud and bandwidth usage on the IoT devices. The system provides trustworthiness of PHR data, efficient authentication, and decentralized integrity checks, and the blockchain is employed for immutability of cyphertext codes to avoid tampering. They use a permissioned implementation of the Ethereum blockchain with the Proof-of-Stake consensus algorithm. Their performance evaluation concerns mostly operations related to the encryption keys, such as generation latency, key size, size of ciphertext, and so on.

3.2 Discussion on the Related Works

Across the state-of-art of blockchain-based security systems for HIoT, the more popular motivations behind blockchain usage is access control and management, accounting in the form of record-keeping, and logging of data access. The majority of the related works use blockchain for indexing while the actual medical file is stored elsewhere. Whether using a cloud provider or decentralized storage, usually only the address to the file is stored on-chain. This not only solves compliance issues but also improves the utilization of resources such as storage space, and adds an extra layer of encryption and access control between systems. Table 3.1 summarizes the key architectural components of the related works mentioned in this chapter.

While the topic of blockchain networks has experienced a surge of works, there is still much room for improvements regarding the efficient use of resources. These improvements have increased importance in the integration with the HIoT paradigm. It is pressing for researchers to opt for resource-aware implementation choices, such as avoiding PoW and similar algorithms that rely heavily on mining processes. Implementation choices capable of optimizing ledger scalability while mainly focusing on performance issues can ultimately save memory resources. The use of decentralized storage can substantially improve the performance and security of HIoT systems, however, it has a few drawbacks widely overlooked in the existing literature. For example, a large-scale health facility will have thousands of files stored in its database. Using a blockchain transaction to store the address and add a signature to every single file might not escalate well and become too costly in the long run. In addition, the blockchain-as-index method commonly

Table 3.1 – Overview of related works.

Author	Blockchain	Consensus	Storage
Xu et al. (2019)	Both public and consortium	PoW for the public chain and PBFT for the consortium one	Decentralized file system (IPFS)
Akkaoui, Hei and Cheng (2020)	Both public and private (Ethereum)	PoW for the public chain and PoA for the private one	Decentralized file system (IPFS)
Ali et al. (2020)	Public (Ethereum)	PoA	Storage is local on user device (Tor as a data delivery system)
Chen et al. (2021)	Consortium (Hyperledger Fabric)	Kafka	Semi-trusted cloud provider
Mayer et al. (2021)	Consortium (Hyperledger Fabric)	PBFT	On-chain
Zhang et al. (2022)	Private (Ethereum)	PoS	Untrusted cloud provider

used in the related literature makes the address of EMRs available to every blockchain participant. This strategy relies heavily on a single layer of cryptography and delegates access control to the storage system. Further research is necessary on how to best employ decentralized storage systems in combination with blockchain networks, while still considering the resource-aware environment of HIoT systems in favor of lightweight state-of-art algorithms.

4 PROPOSED BLOCKCHAIN-BASED HIOT ARCHITECTURE

The previous chapter introduced several blockchain-based security systems for HIoT, each with its own way of combining the architectural components. Each arrangement offers security properties and functionality suitable its own application requirements. However, we believe the model of hybrid on-chain off-chain storage with the blockchain acting as index has room for improvements, specially if used in combination with distributed storage systems. For instance, a system with different levels of access control might wish to avoid sharing the address of a patient's EMR with all the peers of the blockchain network. To explore this issue, this work employs a vault file to act as a middle layer between the blockchain network and the storage system. This chapter introduces our architectural requirements, and then outlines the solution using the vault component.

4.1 Requirements

This section classifies the functional requirements expected from a blockchain-based solution comprehending IoT and medical records into four categories: resource consumption, data ownership, compliance, and security properties.

Resource Consumption. The employment of blockchain in any system must be carefully evaluated, or else it can harm the performance of the application before it can bring substantial benefits. The consumption of energy and resources should be lower or at least the same as traditional techniques capable of providing the same functionality. Some architectural choices that can help attain resource economy are lightweight consensus algorithms and optimizing the total number of transactions that can fit in a block (ZHENG et al., 2018). Another performance factor that must be investigated is the cost of the minimum transaction throughput required by the application. Finally, not all of the performance requirements are related to the blockchain. Medical files must be encrypted before storage and decrypted back into a human-readable format before being handled back to the user, which adds to the system's overhead.

Data Ownership. The related literature constantly discusses the distinction between data owner and data custodian. Some use cases demand that other users apart from the patients might have access to the medical data, either anonymized (e.g., for medical

research) or not (such as the patient's physician). This issue directly impacts implementation choices regarding encryption and access to data since multiple access keys are needed. Besides handling authorization, other possible requirements regarding the access keys are the addition of multiple authentication methods and the possibility of key loss. Finally, some application scenarios must account for providing access when the patient is unconscious or unreachable. In the scenario of this work, the main requirement is that the patient, instead of the organization, should be the owner of the data stored in the system. This would allow the patient to pursue treatment across different medical facilities enrolled in the federation and request the deletion of the data in case they no longer require medical assistance.

Compliance. Any architecture handling personal information must follow the applying compliance legislation. As mentioned in Chapter 2, GDPR and LGPD, for instance, both demand that all the personal data stored must be capable of deletion upon request by the user. In the case of blockchain systems, it is worth noting that data stored directly in the ledger as part of a transaction can be accessed indefinitely since the ledger blocks part of the chain are immutable. The system's design must take this into account and ensure that no version of the patient's private data can be accessed from a previous version of the ledger. Consequently, if the system employs other methods for off-chain storage, it must also enable the removal of personal health records and any other private information per user request.

Security Properties. It is crucial in any health system that only authorized people have access to the patient's medical data and that no other party apart from the patient or authorized by the patient (e.g., the patient's doctor at an appointment) can access it. Encryption schemes and strong authentication policies are recommended to ensure the confidentiality of the data and to guarantee that the patient's data stays private. In the case of blockchain systems, the data should also be protected from other people within the same network with access to the ledger. The system should also guarantee the integrity of data, and that the data a user receives is not tampered with or altered in any way. Finally, the system must provide availability by ensuring that the authorized users can access their data on demand. In the proposed architecture, this availability is applicable to all the participating organizations in case the patient wishes to pursue treatment in another facility part of the consortium.

4.2 System Architecture Design

This section presents the desired functionality based on the requirements discussed in the previous section. First, a public blockchain network would be inappropriate for handling HIoT data. This type of network demands far more expensive consensus algorithms. PoW, for instance, has high network bandwidth requirements (RAY et al., 2020) that are unsuitable for a structure already under the considerable traffic load of numerous IoT devices. The mining process of public blockchains also has an enormous energy cost (ZHENG et al., 2018), while consensus algorithms for private chains use different processes that save energy greatly in comparison. In addition, the data stored in a medical blockchain should not be publicly accessible to everyone. The optimal alternative is a consortium blockchain network, where multiple health organizations agree beforehand on the contracts, operations, and the capabilities of each peer. The organization that first admits a patient is responsible for authenticating the enrollment and generating the credentials. The credentials can later be used in tasks such as authentication and encryption of medical files in any participating organization.

Ideally, the patients should be able to pursue treatment in any participating health center and access their medical data using the same credentials due to the distributed nature of the federated consortium network. Cloud-based HIoT systems are vulnerable to many security attacks and data breaches (ZHANG et al., 2022), and using a centralized environment would lead to further issues such as the ownership debate and unequal cost distribution among members. A distributed file storage system instead would meet every requirement of data ownership and decentralization, more so when combined with the consortium blockchain. When the patient no longer requires medical assistance and decides to remove their data from the system, the participating organizations can orchestrate a deletion request, and the records with the associated identity are removed from storage.

In the proposed architecture the blockchain acts as a data index for the medical data of a patient, which is stored off-chain in the distributed storage system. To add an extra layer of privacy and access control, the system uses a intermediate file acting as a vault of file addresses instead of storing the addresses directly on the blockchain. There are two types of use cases to handle. The first occurs in punctual occasions, such as doctor appointments and carrying out medical examinations. The medical staff requires the patient's credentials, and the system fetches the vault file in the blockchain, containing the location of all the available medical data. The health worker can then select the necessary

files from the index, and the system will get them from the distributed storage network. Similarly, if there are new EMRs to be stored in the system, they are encrypted using the patient's credentials, and the storage location is placed with the other EMR in the vault file. A blockchain transaction is then necessary to update the vault file stored in the ledger. The second use case regards continuous sensor data from devices attached to patients in short-term examinations such as sleep apnoea tests and cardiac monitoring. While the workflow remains the same (get credentials from the patient, fetch the medical vault, and so on), the volume of data produced is considerably higher. The gateways handling the sensor devices may aggregate and keep the data locally for a set interval before uploading the sensed data to the system. This interval will rely heavily on the system performance regarding throughput and scalability since it takes a considerably higher number of transactions than the previous scenario.

Figure 4.1 outlines the proposed scheme. Besides patients, the target users include doctors and medical facilities such as hospitals and health providers. These will be affiliated with the different medical organizations part of the consortium. The sensor devices and end-point hosts from patients or doctors make up the device layer, providing the users with access to the medical files stored in the system. The communication between devices and all the different applications providing medical service to users goes through an intermediate layer, encompassing the IoT gateway nodes for the sensor devices, the blockchain network peers, and distributed storage nodes. At last, authenticated applications from the participating medical organizations are located in the application layer. These applications can access the network to provide users with medical and operational services.

The blockchain inherently provides integrity, auditability, and anonymity (ZHENG et al., 2018). In addition, a private blockchain also provides authentication to the participating users. The system also encrypts any medical data before storage in order to provide confidentiality. However, it is necessary to evaluate the performance of the proposed architecture to investigate its capability to keep up with real-world applications, especially regarding the second use case scenario of keeping a reasonably up-to-date sensed value in the distributed system. The performance analysis will also dictate whether the components of the intermediate layer (gateway, blockchain, and storage nodes) can be placed in a single high-capacity device or in small resource-constrained ones. The next chapter presents implementation details and performance results under the specified conditions.

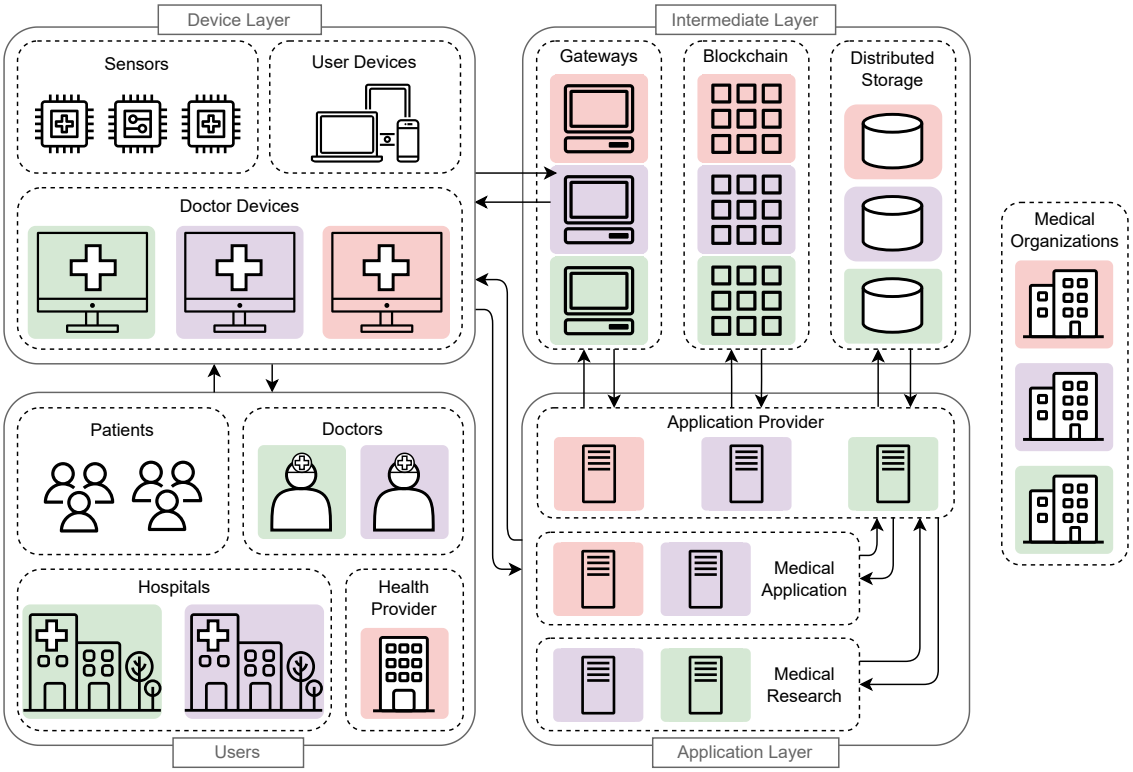


Figure 4.1 – Proposed architecture integrating blockchain, IoT and distributed storage.

5 EXPERIMENTAL SETUP

This chapter provides background concepts about the frameworks selected for the proof-of-concept implementation, as described in Section 4.2. Mainly, we provide details on the Hyperledger Fabric platform and describe the functionality of the IPFS distributed storage. Then, this chapter presents the step-by-step operation of the proof-of-concept system.

5.1 Implementation Settings

This section presents the necessary concepts of the frameworks used in the proof-of-concept system. As shown in the state-of-art literature discussed in Section 3.2, Hyperledger Fabric is one of the most widely adopted frameworks in permissioned blockchain architectures. Fabric default consensus algorithm is based on Raft (ONGARO; OUSTERHOUT, 2014), which substituted the Kafka-based algorithm after Fabric v2.x. Finally, a private network in the IPFS system was selected for distributed storage in the system evaluation.

5.1.1 Hyperledger Fabric

Hyperledger is the Linux Foundation's blockchain development project, launched in 2015. Open-source under the Apache-2.0 license, Fabric[1] is one of Hyperledger's graduated projects in production today. The main difference between Fabric and other projects is the modular architecture. Developers have "plug and play" options for components such as the consensus algorithm, membership services, storage systems, and so on. Fabric is permissioned, and network participants must be previously enrolled in the system before they can issue and validate transactions.

In Fabric, the network participants are primarily organizations, making it a consortium blockchain. Each organization first enrolls through its own Membership Service Provider (MSP). The MSP handles the organization's CA server, which uses a Public Key Infrastructure (PKI) to sign its peers' transactions and assets. In this way, the MSP provides authentication to all the peer nodes, ordering nodes, smart contract code, and ap-

[1] https://www.hyperledger.org/projects/fabric

plications related to that organization. The peers properly authenticated through an MSP are capable of proposing transactions to all the other network participants, regardless of the organization they belong. The communication mechanism by which the members of the network exchange messages is called a channel. The communication can be restricted to subsets of peers and organizations by using different private channels. Finally, the chaincode is the Fabric equivalent of a smart contract.

In addition to regular peers, organizations also own ordering nodes. These nodes make up the ordering service, the key component of the consensus mechanism of Fabric, tasked with ordering transactions and achieving consensus. This design separates the proposal and the execution of transactions to different set of nodes. The primary role of the ordering service is to guarantee ledger consistency, which prevents forks of divergent ledger states. According to the Fabric documentation, this decoupling of ordering and execution benefits performance and scalability (HYPERLEDGER FOUNDATION, 2022). The ordering nodes also keep track of the authorized organizations, the read and write permissions of each channel, and overall access control. The transaction flow of the Fabric ordering service works as follows. Firstly, upon request from the application, the peers produce a proposal for ledger update based on a smart contract previously agreed on and submit it to an ordering node. The ordering service then arranges all the submitted transactions into a sequence of blocks that are distributed back to the peers for validation. Each peer checks for the necessary endorsements and possible conflicts and, finally, commits the blocks to the current state of the ledger.

The current default ordering service used on Fabric is based on the Raft protocol. Raft is crash-fault tolerant, which means that a collection of several replicated state nodes that can withstand consensus and continue operating even if some of them are down (ONGARO; OUSTERHOUT, 2014). Raft was developed primarily with understandability in mind. In order to improve clarity, Raft decomposes the consensus problem into smaller subproblems. The algorithm first elects a leader, who becomes responsible for accepting client requests and replicating them across the cluster of nodes. If the leader fails, a new leader is elected among the nodes who remain online. The log replication and safety protocols are also decoupled from the consensus process, streamlining the process and making it easier to understand. Raft substituted Kafka as standard since Fabric v2.x, mainly because Kafka and the accompanying Zookeper infrastructure are complicated to set up and have way more components. Despite that, the operation of the two algorithms is very similar. Both are Crash-Fault Tolerant and use the same model of "leader and

follower" design.

5.1.2 Inter-Planetary File System

IPFS[2] is the distributed storage protocol most widely used in the related literature. It employs content addressing instead of the location addressing commonly used in the internet. This section will cover some concepts for understanding how IPFS works, as well as some consequences of this design to the system architecture.

Content addressing in IPFS works by dividing each file into blocks of same size, and then hashing the content of each block to generate a unique Content Identifier (CID). All the blocks that make up a file are indexed through the root CID of that file, which is possible by structuring it using a Merkle Directed Acyclic Graph (Merkle DAG). Only this root CID matters to the system architecture since it will intrinsically lead to all the remaining file content. It is worth noting there is additional data besides the block hash embedded into a CID (version information, codecs, the hashing algorithm used, etc.), so simply applying a hashing algorithm such as SHA-256 will not make up a CID number. For the peers to discover which nodes host which CIDs, the IPFS network employs a Distributed Hash Table (DHT). The DHT of IPFS implements a variant of the Kademila algorithm (MAYMOUNKOV; MAZIERES, 2002). It provides key-value lookup in physically separated systems and stores both content and routing information about how a peer can be reached. The nature of the described content addressing and CIDs has several implications. First, since the address is based on the file content, two identical files will generate the same CID. This means that every file is only stored in the system once, eliminating the need for double storage. In addition, two similar files can share parts of the Merkle DAG - that is, a subset of their compounding blocks are identical and can be referenced by both files. This is useful while optimizing the storage of large datasets or new versions of the same file with small modifications. Finally, since the address of a file relies on its content, a new root CID must be generated every time a file is modified. The system workflow to deal with these implications is detailed in the next session.

The data lifecycle in IPFS has several practical points worth noticing. The first one concerns the addition of new files and *pinning*. To say a node has a file *pinned* means the node is announcing it hosts that file, and the pair of CID and node's IP is available on the public DHT. A node will automatically pin all the files added by itself. Other nodes

[2]https://ipfs.tech/

will usually only store and pin a file if they request it at some point. The requested files stay pinned for an indeterminate amount of time unless garbage collection is enabled. Garbage collection is the removal of old pinned files to save up storage space, however, it is not enabled by default in a new installation of the IPFS daemon. Garbage collection can be configured to run either on a set time interval or whenever the storage capacity of that node reaches a certain threshold. If garbage collection is not enabled, there is no limit to the number of pinned files or how long they stay pinned. It is crucial to notice that if only one node pins a particular file, that file will no longer be accessible if the node goes offline. There are strategies for replication and persistence of data, e.g., IPFS Cluster[3], that can be used to provide fault-proof availability of critical files. Finally, data deletion in the IPFS network is a complicated task. In order to delete a file, it must be unpinned (and the garbage collection must run afterwards) in every node that hosts it. Unpinning (and garbage collecting) in only one node means the network can still reprovide the file, provided that another node still has it. All this process can be tricky in the public IPFS network, but it is feasible inside a private network since all nodes belong to cooperating organizations. In addition, there is research in delegated content deletion protocols (POLITOU et al., 2020) that can be used to distribute and orchestrate a deletion request through the IPFS network.

Regarding privacy, IPFS requires some adjustments before it can be safely applied to healthcare settings. The core functionality of the IPFS network requires the CIDs, the participating nodes, and DHTs to be public. This means that even though data in transit between nodes is secure, a third party monitoring the traffic to the DHT can determine who requests which CID. In addition, the public IPFS network is unsuitable for sensitive applications since anyone can upload and request any file simply by having the CID. Instead, access to the network should be restricted to a selection of trusted nodes by employing a private IPFS network, wherein the nodes must have a swarm key to join. Privacy-critical files such as EMRs must be encrypted before storage in the system since anyone who holds the CID might request the corresponding file. IPFS lacks authentication and the ability to track access (HANAFI; PRAYUDI; LUTHFI, 2021), a common reason why blockchain is often employed in IPFS applications. However, the usual practice of simply storing the file address in a distributed ledger presents a few drawbacks. It is a rather computationally expensive solution for adding a signature to data and has the substantial disadvantage of making the file, its owner, and its location known to all

[3] https://ipfscluster.io/

participants of the ledger network. To address this issue, the proposed solution employs a vault component outlined in the following section.

5.2 System Workflow

The system workflow is described schematically in Figure 5.1. It has three main tasks: initialization, storage of a patient's new medical files, and access of files by an authorized party. In the initialization step the patient's Vault file is created, encrypted, and stored in the IPFS storage, and the root CID address is used in the creation of a Medical Asset in the Fabric Ledger. For the addition of medical records, the same asset is read from the Ledger. The system gets the Vault from IPFS based on the address stored in the Medical Asset, decipher it, and appends the CID address of the medical file previously added to IPFS. The Vault then is encrypted and returned to IPFS, and a blockchain transaction updates the Vault address in the Medical Asset. As for the read-only operation there's no need to modify the Medical Asset, simply read it from the Ledger, extract the address and fetch the Vault, and extract the IPFS address of the desired medical files from the index. This section details further these three operations.

The initialization step described in this section will not be part of the performance analysis since it happens only once when a patient is enrolled in the system. The addition and access of medical files, in turn, takes several requests to the Fabric network and IPFS and will be explored in further detail in the next chapter.

5.2.1 Initialization

Upon admitting a patient for the first time in the network, the health facility must generate a Patient ID for indexing in the ledger. Ideally, any participating organization can enroll a new patient through its own MSP, and the patient's data will still be available in the shared ledger if they decide to pursue treatment in any other organization in the consortium. Following the enrollment process, the patient must generate a Personal Key using either traditional methods (password, passphrase) or biometric signals (such as fingerprint, electrocardiogram, and so on). This key can be used both in authentication tasks and in the symmetric encryption/decryption of files.

In the next step, the system creates a Vault file. This is the first step shown in

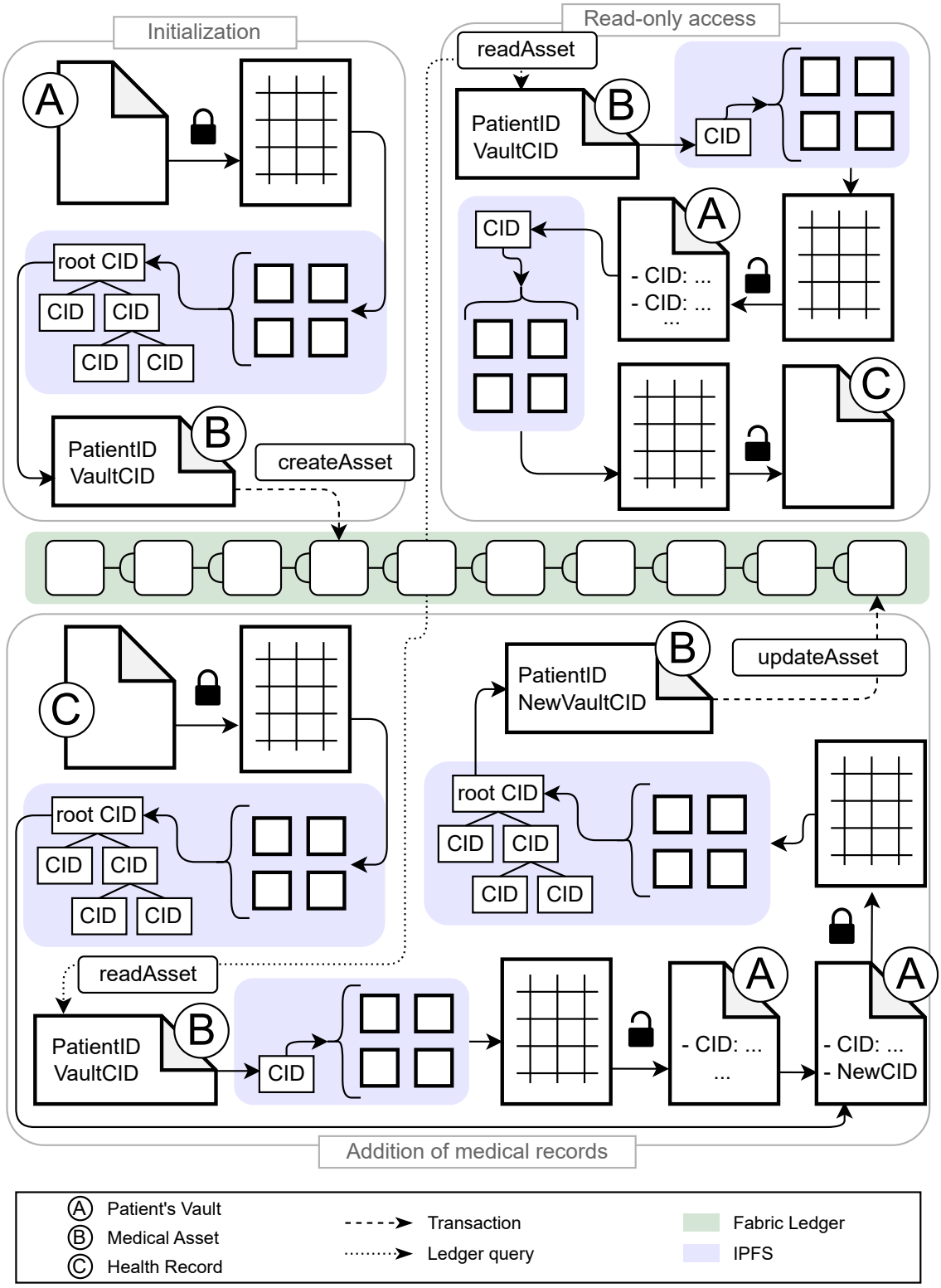


Figure 5.1 – Detailed workflow of the system operation.

the Initialization block in Figure 5.1. The Vault file is one of the key components of the proposed architecture. Instead of storing the CID of the medical files directly in the ledger, the Vault serves as the index of all information regarding that patient. This

alternative reduces the number of medical assets in the ledger and facilitates the deletion of files since the permanent address will not be stored in the immutable ledger. Also, it adds an extra layer of privacy. Not storing CIDs of medical files directly in the shared ledger means the file address will not be immediately visible to every network participant, attached to the patient's identification.

Following the workflow in Figure 5.1, the system then encrypts the Vault using the Personal Key and adds it to the IPFS system. With the Vault CID in hands, the Patient's Medical Asset[4] is created using the pair PatientID, VaultCID and added to the ledger. By the end of the initialization step, the patient is successfully enrolled in the system, and the address of the encrypted Vault is publicly accessible within the network.

5.2.2 Addition of medical records

The operational flow is depicted in the "Addition of medical records" block in Figure 5.1. It requires several interactions with the Fabric network and the IPFS system and is roughly divided into three steps: the vault retrieval, the encryption step, and indexing and update of the Medical Asset. Figure 5.2 have each operation described using a sequence diagram.

Vault retrieval. The flow starts when the patient requests the *addFile* operation to the system and provides their PatientID. The system then performs the following operations to fetch the patient's Vault and provide its content to the user.

- *readAsset* (rA). Using the PatientID, the system queries the ledger about the Patient's Medical Asset to get the Vault CID.
- *ipfsGet(Vault)* (iG_V). The Vault CID from the Medical Asset is used to retrieve the encrypted Vault from IPFS. If the Vault CID in the ledger no longer matches the CID of the stored Vault file, it is a good indicator that the file is corrupted.
- *decrypt(Vault)* (d_V). The patient provides the Patient Key and the system decrypts the Vault file, providing access to the medical registry of that patient in the system.

Encryption step. The next two operations in Figure 5.2 regards the pre-processing

[4]The use of "asset" as a word choice is due to a common terminology in blockchain spaces and is maintained for clarity even though it has no monetary value in this context.

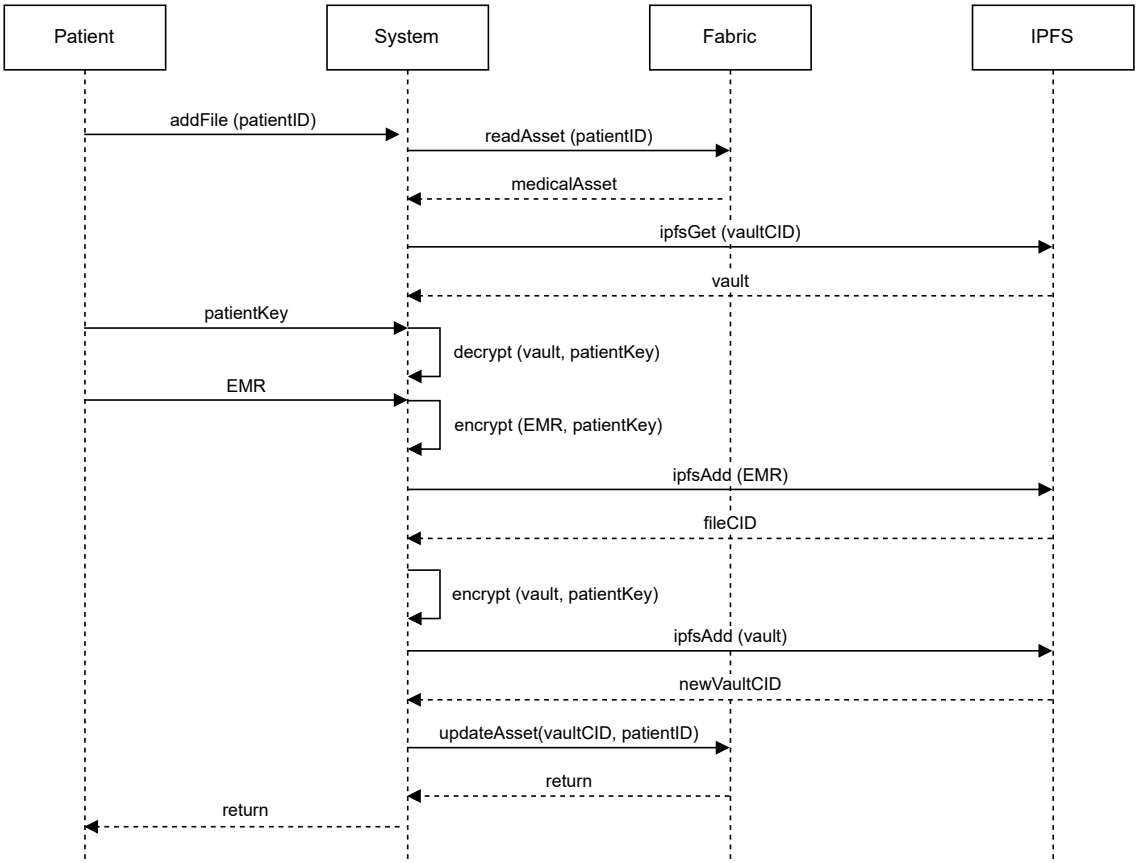


Figure 5.2 – Sequence diagram of the add operation.

of the medical file to be added to the patient's registry.

- *encrypt(File)* (e_F). Before it can be uploaded to the storage system, the medical file must be encrypted using that patient's Personal Key.
- *ipfsAdd(File)* (iA_F). The encrypted file is then added to the IPFS storage system, obtaining the file CID.

Indexing the file and updating the Medical Asset. The Vault file acts like an index of all the medical records the patient has stored in the system. Before the following steps, the CID of the encrypted medical file is added to the patient's registry in the Vault. This interaction with the Vault file is not depicted in Figure 5.2.

- *encrypt(Vault)* (e_V). After the addition, the Vault file is encrypted back using the same Personal Key from the patient.
- *ipfsAdd(Vault)* (iA_V). Since the Vault file has been altered, a different Vault CID will be generated upon returning it to the IPFS system.
- *updateAsset* (uA). The system must then perform a blockchain transaction to update the Patient's Medical Asset to PatientID, NewVaultCID. This step

ensures that all parties are appropriately authorized and that the modification is legitimate.

In addition of the mentioned operations, it is necessary to remove the previous version of the Vault file from IPFS. Otherwise, it will still be available in the network to any party holding the previous CID. As mentioned in Subsection 5.1.2, the orchestration of data erasure in IPFS is a delicate procedure and will not be covered by the performance analysis in the next section.

5.2.3 Read-only access

The reading process, depicted in the "Read-only access" block in Figure 5.1, is considerably more straightforward. Figure 5.3 presents the sequence diagram detailing the read-only operation. The workflow starts when the patient requests the *readFile* operation from the system and provides their Patient ID.

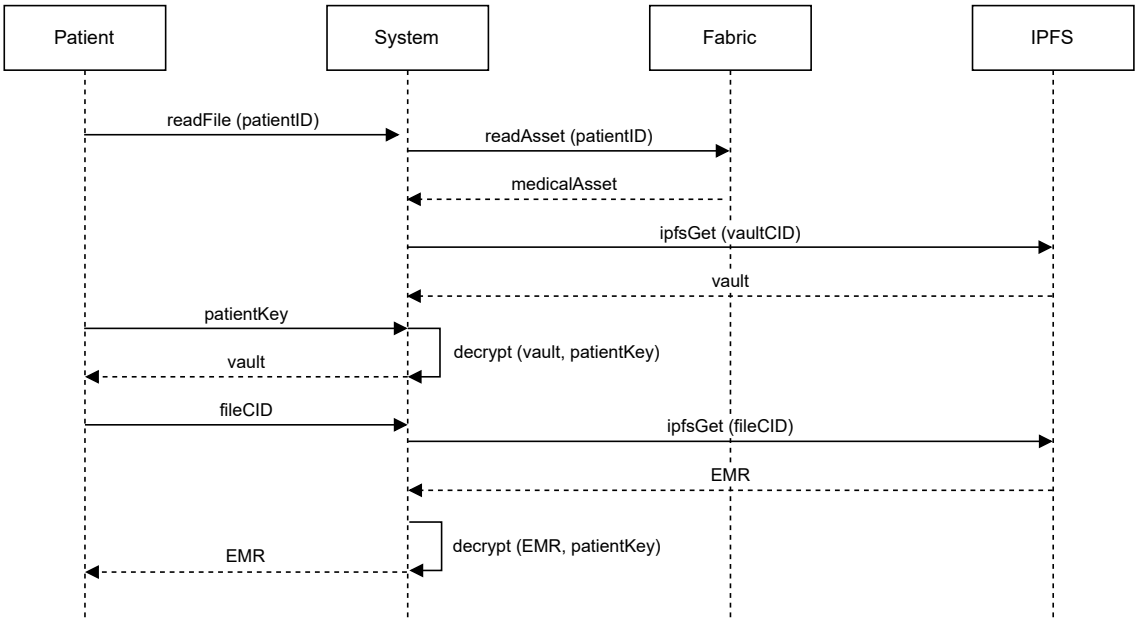


Figure 5.3 – Sequence diagram of the $read$ operation.

- *readAsset* (rA). The process begins in the same manner as in the add operation, obtaining the Vault CID from the Patient's Medical Asset on the ledger.
- *ipfsGet(Vault)* (iG_V). The system then fetches the encrypted Vault file from IPFS.
- *decrypt(Vault)* (d_V). The vault must be decrypted using the Personal Key to access the address of the patient's medical files.

- *ipfsGet(File)* (iG_F). At this point, the system can show the contents of the Vault to the patient for a specific file to be selected. The performance evaluation will not take this intermediate step of prompting the patient into account so that the test flow will not be disrupted. After obtaining the desired medical file CIDs, they must be fetched from storage.
- *decrypt(File)* (d_F). The same key is then used to decrypt the medical file obtained from IPFS.

These steps must be performed in the presence (or with the authorization) of the patient, since the Personal Key is necessary. A read-only query does not alter the blockchain ledger, so the final step of updating the Patient's Medical Asset is unnecessary. If any unauthorized modification is made, the corrupted file CID will no longer match the official one stored in the Vault.

Some use cases will require constant updating of medical files (e.g., keeping a file with a fresh sensed value stored in the system), which will, in turn, require several transactions to be performed in a time frame. The following chapter brings the performance results of the proof-of-concept system.

6 PERFORMANCE EVALUATION

The performance analysis of the proof-of-concept system is divided into three complementary parts. First, we evaluated the performance of the Fabric network on the chaincode operations specified in Section 5.2, specifically the *updateAsset* transaction and the *readAsset* query. Then, we assessed the capability of the IPFS distributed storage to keep up with the throughput attained by the blockchain. Finally, all the parts are joined in the Node.js application implementing the functionality described in Section 4.2 and outlined in Section 5.2. The performance of both the blockchain and the storage system is put in perspective with other crucial system operations, such as the decryption and encryption of files. The evaluation aims not only to assess the system's capability but also to identify current bottlenecks in need of improvement.

The implementation environment consists of a Ubuntu 20.04 LTS system with 15,5 GiB of RAM and a quad-core, 2.70GHz CPU. The IPFS nodes run in three virtual machines, each with Linux Mint 21.2, 2GB of RAM, and one single core allocated. They use Kubo v0.23.0, the IPFS implementation written in Go. The Fabric v2.2 network runs directly in the host machine.

6.1 Chaincode Performance

Hyperledger Caliper v0.5.0[1] was the blockchain benchmarking tool used to evaluate the Fabric Network. Caliper executes one or more testing rounds specified under a benchmark configuration file. Each round has a pre-defined duration, rate control, and the workload to be executed as defined in the test workload module. The rate controller options have different profiles for each type of testing, such as keeping a fixed rate of transactions per second, increasing the rate at a fixed load, and so on. After all the rounds, Caliper produces a report with performance indicators such as throughput, latency, and success rate.

The performance test of the proof-of-concept blockchain network consisted of gradually increasing the transaction load of each round for the two chaincode operation, *readAsset* and *updateAsset*. The type of rate controller used aims to maintain the specified transaction load while increasing throughput as much as possible. The result is the maximum possible TPS (transactions per second) for the system under testing. It was nec-

[1] https://hyperledger.github.io/caliper/

essary to watch the fail rate closely while benchmarking the *updateAsset* operation, since failed transactions happen mostly because of reading conflicts. The test module specifies randomly which asset will be updated by the worker client; each transaction takes about half a second to complete, and each worker client performs several transactions per second. Therefore, the benchmark will sometimes randomly select an asset currently undergoing edition. The benchmark increased the number of assets available for operation as the load of each round got heavier in order to keep the fail rate under 10%. The generation of assets for testing happens at the initialization module of the benchmarking process and has no impact on the performance testing rounds. Since the *readAsset* operation is read-only, it generates no reading conflicts.

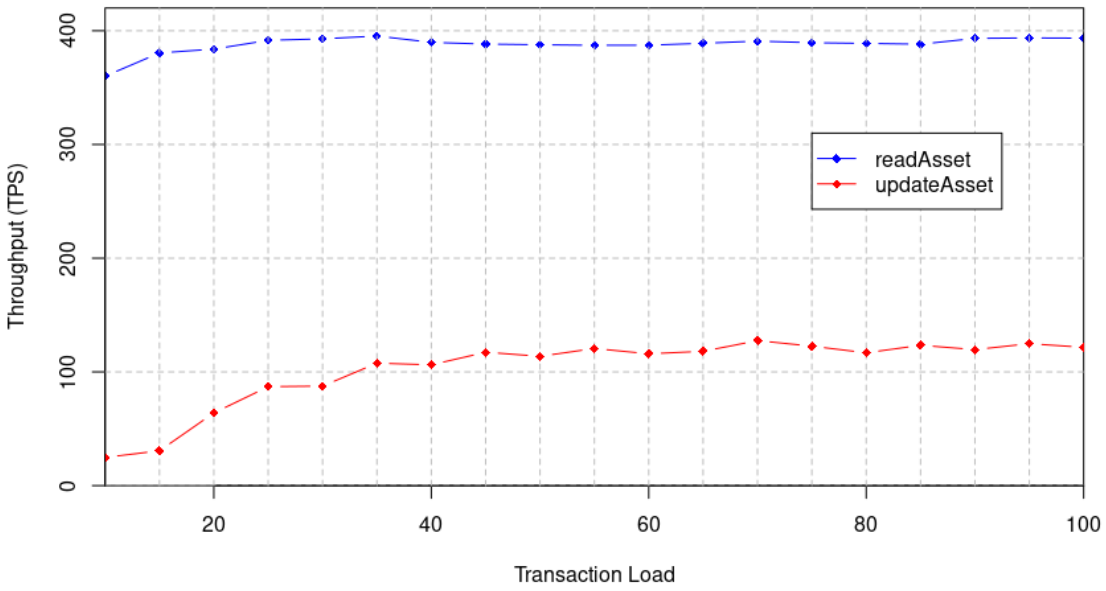


Figure 6.1 – Throughput of the $readAsset$ query and $updateAsset$ transaction, per transaction load.

Figure 6.1 presents the throughput of both the *readAsset* query and the *updateAsset* transaction. After reaching the load of 50 concurrent transactions, the *updateAsset* method stabilized between 117 and 127 transactions per second. Because it is a read-only query, *readAsset* could achieve a greater throughput and stabilize around 390 transactions per second. Figure 6.2 shows the average latency of *updateAsset* increasing steadily with the number of concurrent transactions as expected. Therefore, the system's optimal load would be between 50 and 70 concurrent transactions, since it reached the maximum throughput under shorter latency. The average latency of the *readAsset* query is omit-

ted in Figure 6.2 since it is mostly negligible, taking 0.03 seconds under the load of 100 concurrent transactions.

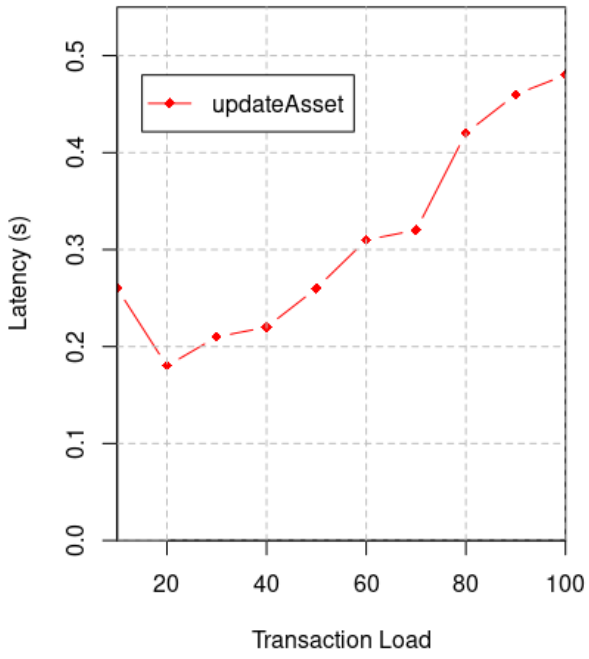


Figure 6.2 – Total latency of the $updateAsset$ transaction, per transaction load.

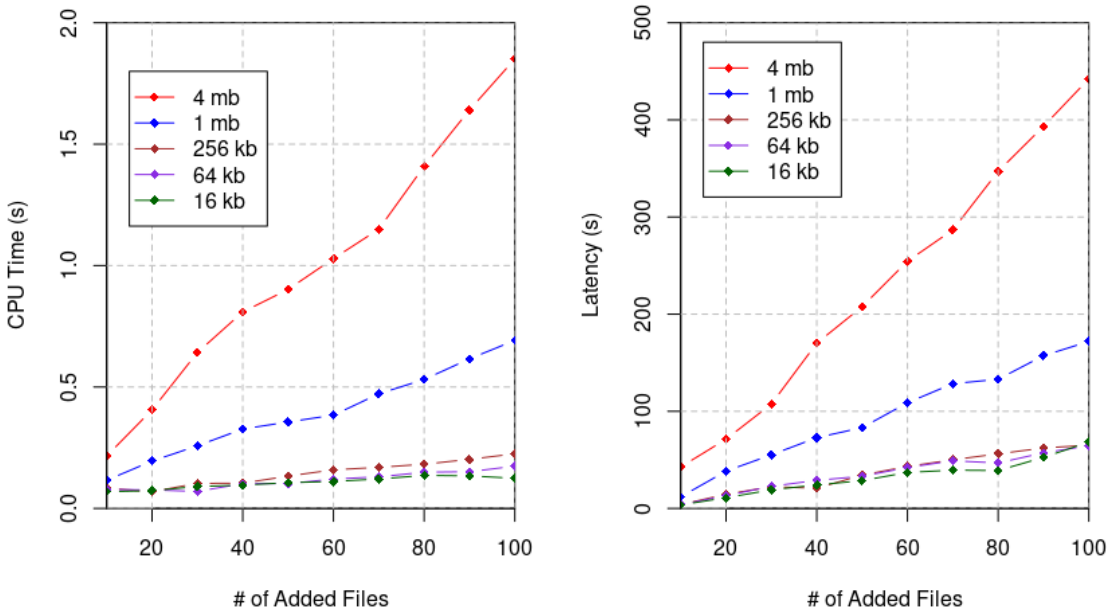
While the overall throughput of the Fabric network in our tests did not reach the same performance as other works using the same platform, much of this gap can be attributed to the hardware that runs the experiments. The goal of the proposed architecture is to run in health facilities where there won't always be specialized high-performance hardware available for the blockchain network, so our test environment is a more faithful representation of the type of equipment expected to run the system in its final version. This makes the performance tests an important assessment of the Fabric network and of the complete blockchain-based architecture itself. More so, as we will see later in this chapter, other crucial system operations are the current bottleneck of the proposed architecture. The following section evaluates the IPFS network and assesses its ability to keep up with the blockchain throughput.

6.2 IPFS Performance

The performance test of the IPFS network is intended to measure how long it would take for the storage system, specifically the *IPFSadd* and the *IPFSget* methods described in Section 5.2, to keep up with the throughput of the Fabric network shown in the previous subsection. The *IPFSadd* test consisted of adding several batches of files to the IPFS network and measuring latency and CPU time. For this operation we used a bash script and the *time* command. The batches contain an increasing amount of files, from 10 to a hundred. The test was repeated with varying file sizes, influenced by the work of Lajam and Helmy (2021), with the goal of comparing the processing time for each of them. The block size was kept at the standard 256 Kb in all tests. Between each round, the pin list from the previous round was removed and garbage collected to ensure that cached content did not impact the performance of the current round. To measure the *IPFSget* operation, in turn, the file batches were previously uploaded by a different node in the network and then requested using the same tools as the *IPFSadd* tests.

For the *IPFSadd* operation, Figure 6.3a first shows the CPU processing time of the IPFS node - that is, the amount of CPU time used by the process both inside and outside the kernel. It does not take into account the time the process was blocked, for example, waiting for I/O operations. Figure 6.3b then shows the total latency experienced by the system taking I/O operations into account. The $ipfs\ add$ operation doesn't make any requests to the IPFS network. It mostly processes the files and metadata inside the node by breaking the file into blocks, hashing the blocks, and storing the address in the corresponding IPFS directories where they can be fetched the next time some other node requests it. Reading and writing operations take the vast majority of the total latency, while the time spent processing the metadata is short in contrast. We can also observe in Figure 6.3b that processing files up to the block size of 256 Kb takes roughly the same total latency. A node handling bigger files would benefit from a larger block size since it would need fewer blocks and, consequently, fewer I/O operations (LAJAM; HELMY, 2021). Still, it would ultimately degrade the performance of all file sizes smaller than the current block size. While the block size parameter can be modified in the configuration of the IPFS daemon, it is static once the daemon is running and currently cannot be altered dynamically.

The same gap between CPU time and total latency can be observed in the *IPFSget* operation to a greater degree, as shown in Figure 6.4. In the existing literature, Shen et



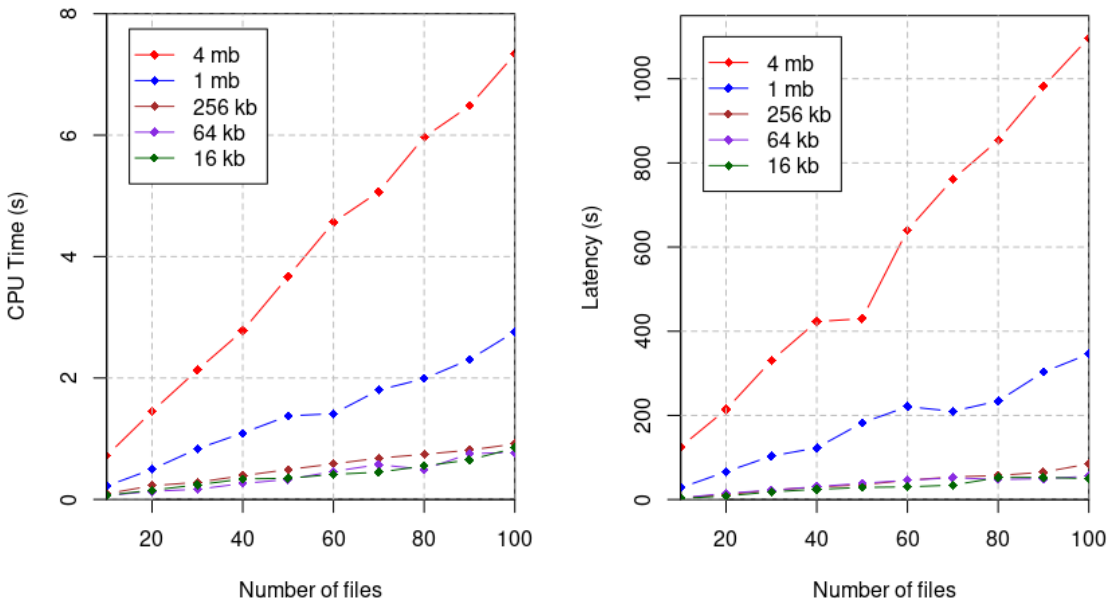
(a) CPU time of the *IPFSadd* operation per number of files.

(b) Total latency of the *IPFSadd* operation per number of files.

Figure 6.3 – Performance results of the *IPFSadd* operation.

al. (2019) shows that the system performance gets worse as the size of the data increases (SHEN et al., 2019). Lajam and Helmy (2021) then observed the same in an isolated private network (LAJAM; HELMY, 2021). Since our test environment consists in a small private network, the DHT lookup time to locate the desired files and their addresses is mostly negligible. The situation is very different in the wide public network, where the resolving operation can be one of the bottlenecks of IPFS. In opposition to the $ipfs\ add$ operation, $ipfs\ get$ and downloading blocks rely on the node's connection bandwidth. However, even though the LAN network running the experiments had a 2 Gb/s bandwidth between virtual and host machines, the system did not use all the available bandwidth. This is explained again by the set block size, 256 Kb, which prevents them from using the whole link width - another instance in which bigger blocks might optimize performance for large files.

It is not within the scope of this work to provide a complete performance analysis of the IPFS network, but to assess the current bottlenecks of the proposed architecture. The previous section shows that the Fabric network can perform 390 *readAsset* queries per second and around 100 *updateAsset* transactions per second (rounded down for convenience). This means that it would take, for instance, 170 seconds for the IPFS system



(a) CPU time of the *IPFSget* operation per number of downloaded files.

(b) Total latency of the *IPFSget* operation per number of downloaded files.

Figure 6.4 – Performance results of the *IPFSget* operation.

to keep up with the optimal blockchain throughput when handling files with 1 Mb, and around 350 seconds to download the same files. Besides, the *readFile* operation described in Section 5.2 takes two *IPFSget* requests for every *readAsset* query, and for the *addFile* operation, it takes one *IPFSget* and two *IPFSadd* requests for every pair of *readAsset*, *updateAsset*. It is safe to say that the IPFS storage is the system's bottleneck in the current development state, and that I/O disk operations are the main factor behind the processing degradation. However, as we discuss further in the next chapter, the IPFS storage is not without benefits. To name one, the intrinsic address permanency is a novel feature that has no equivalent in traditional storage systems. The next session covers the remaining system operations, such as encryption and decryption of files, and compares each step of the complete application.

6.3 The Complete Application

The proof-of-concept implementation was written in Node.js, v20.8.0, using *npm* v10.1.0. Besides the official Fabric modules, the application used the *js-kubo-rpc-client*

module v3.0.1 to access the HTTP RPC API of the IPFS daemon and the *crypto* module for cryptographic operations. A simple bash script passes commands for the performance analysis tests through the command line interface.

For the cryptographic operations, we opted for a symmetric key encryption algorithm. The proof-of-concept employs AES-256 with cipher block chaining as the mode of operation. AES-256 is HIPAA compliant and a known industry standard, used here to compare the cryptographic process with the other system operations. It is worth noting that the encryption and decryption steps also altered the size of the files when executing the complete proof-of-concept application. Previously, we evaluated how IPFS handled files up to 4 Mb. In this evaluation, the original plaintext file had 1 Mb but the encrypted version processed by the IPFS storage had approximately 2 Mb due to encoding differences.

Table 6.1 – Execution time of each system operation, in milliseconds, for files with 1 Mb.

	rA	iG_V	d_V	iG_F	d_F	e_F	iA_F	e_V	iA_V	uA	*total*
read (ms)	16.844	6.645	0.134	5586.85	8.235	-	-	-	-	-	**5618.70**
add (ms)	34.132	9.009	0.175	-	-	4.796	2318.3	0.155	832.13	2097.3	**5295.99**

Table 6.1 shows the side-by-side comparison of all system operations when handling a medical file of 1 Mb unencrypted. Surprisingly, *read* surpassed *add* due to the time increase of *ipfs get* over *ipfs add*. We can see that the IPFS storage consistently remains the most costly step regarding execution time. The *ipfsGet(File)* and *ipfsAdd(File)* times are consistent with the results obtained previously if considering encrypted files of 2 Mb. As for the blockchain operations, they appeared way above the latency obtained in Section 6.1. This can be explained by the lack of load pressure by the application, as we noted that Fabric lacks the incentive to increase throughput under very small loads. This behavior also appeared in the benchmarks when executing the chaincode under a small transaction load (up to 5 concurrent transactions). The cryptographic operations are negligible if compared with the total application time.

7 CONCLUDING REMARKS

This dissertation proposes and evaluates a blockchain-based architecture for improving the security of medical systems, particularly considering HIoT use cases. This chapter concludes the work with an overview of the contributions and their relevance if compared with the related literature. Then, we present the next steps for this research.

7.1 Summary of Contributions

The proposed system employs a consortium blockchain and distributed content-addressing storage to meet the necessary functional requirements of HIoT applications. Regarding resource consumption, the proof-of-concept implementation opts for resource-aware algorithms and architectural choices due to the limitations of HIoT networks. In order to avoid storing the permanent address of medical files directly into the blockchain ledger, the architecture uses a vault component working like an index. The vault keeps the location of the EMRs of a patient hosted on the distributed storage system and allows the deletion of personal files upon request in compliance with most data protection laws. Finally, regarding the security properties of confidentiality, integrity, and availability, the system employs cryptographic techniques in addition to the mechanisms that private blockchains intrinsically provide.

Table 7.1 compares the proposed architecture with the related literature discussed in Chapter 3. We can see that storing the address of the medical files in the ledger, which we aim to avoid, is a common practice among works using decentralized storage. In addition, even works with similar concerns regarding decentralized storage use public blockchains to handle the issue of data ownership in their architecture. While combining private and public blockchains is a decent short-term solution for managing different levels of access authorization to private data, this work wishes to avoid using public blockchains at all, mainly due to the energy costs of the mining process. When properly anonymized, we believe that authorized third-party access to patient data can be accomplished through other energy-aware methods, as we discuss further in the next section.

We evaluated the proposed architecture by means of throughput and latency. We first benchmarked separately the blockchain network and the distributed storage system. Then, we combined all the components into a Node.js application. The blockchain performance attained suitable results for the specified use cases. The distributed storage system

Table 7.1 – Comparison of this system with related systems or schemes.

	Xu et al.	Akkaoui et al.	Ali et al.	Chen et al.	Mayer et al.	Zhang et al.	This work
Platform	-	Ethereum	Ethereum	Fabric	Fabric	Ethereum	Fabric
Consensus	Pow/PBFT	Pow/PoA	PoA	Kafka	PBFT	PoS	Raft
Avoid using public chains	×	×	×	√	√	√	√
Off-chain storage	√	√	√	√	×	√	√
Decentralized storage	√	√	×	×	×	×	√
No CID on ledger	×	×	×	×	×	×	√

is the current bottleneck in the present development state. However, it was demonstrated that the performance is improved when handling small files, and the obtained latency is still feasible for some of the use cases. Besides, the IPFS network offers a different set of features from the ones commonly found in traditional storage systems with higher performance results. In addition to the benefits inherent to distributed, p2p storage (such as scalability, cost-effectiveness, and elimination of a central server as single point of failure), IPFS content addressing and permanency of files can work in advantage to the underlying architecture. In systems that handle large datasets of similar data, for instance, it can compensate by saving storage space. Apart from minor adjustments to the current configuration, the next steps of this research are outlined in the next section.

7.2 Future Work

Possible future research should investigate the use of other consensus algorithms and their impact on the overall performance. Fabric is currently developing their own Byzantine Fault-Tolerant ordering service, but there is also independent research on the topic. One of the driving motivations behind the use of Fabric, compared with other Hyperledger projects, is the adoption of modular components allowing users to add their custom algorithms to the infrastructure. In addition, we believe that evaluating different encryption methods can substantially improve the architecture. Public-key Attribute-Based Encryption (ABE), for instance, enables fine-grained access control based on matching attributes between the key and the cyphertext. At the same time, Multi-Authority ABE schemes allow more than one organization to issue the attributes, distributing the authority behind access credentials to independent parties. We believe that further research into aggregating a suitable data encryption model to the present architecture could even advance the issue of providing data access to authorized third parties (e.g., emergency contacts, medical research, and so on). This might be a suitable alternative to the more costly method of employing additional public blockchain networks often found in the related

literature.

To expand the present architecture, the next steps could include adding further options to each storage node configuration in the setup, e.g., the block size. Also, the addition of a web interface would improve usability. Finally, including the network scope in the performance assessment would be valuable. Simulating the traffic from IoT devices or using real sensor devices to generate traffic would allow us to measure the bandwidth cost in real-world scenarios.

REFERENCES

AKKAOUI, R.; HEI, X.; CHENG, W. Edgemedichain: a hybrid edge blockchain-based framework for health data exchange. **IEEE access**, IEEE, v. 8, p. 113467–113486, 2020.

ALI, M. S. et al. A decentralized peer-to-peer remote health monitoring system. **Sensors**, MDPI, v. 20, n. 6, p. 1656, 2020.

BORMANN, C.; ERSUE, M.; KERANEN, A. Terminology for constrained-node networks. **Internet Engineering Task Force (IETF): Fremont, CA, USA**, p. 2070–1721, 2014.

CHEN, Z. et al. A blockchain-based preserving and sharing system for medical data privacy. **Future Generation Computer Systems**, Elsevier, v. 124, p. 338–350, 2021.

ETHEREUM.ORG. **Proof-of-Stake (PoS)**. 2023. Avaliable at <https://web.archive.org/web/20240128220433/https://ethereum.org/en/developers/docs/consensus-mechanisms/pos>. Accessed on 2024/01/28.

EUROPEAN PARLIAMENT AND COUNCIL OF THE EUROPEAN UNION. **Regulation on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation)**. 2016. Available at <https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679>. Accessed on 2023/10/14.

GHUBAISH, A. et al. Recent advances in the internet-of-medical-things (iomt) systems security. **IEEE Internet of Things Journal**, IEEE, v. 8, n. 11, p. 8707–8718, 2020.

HANAFI, J.; PRAYUDI, Y.; LUTHFI, A. Ipfschain: Interplanetary file system and hyperledger fabric collaboration for chain of custody and digital evidence management. **International Journal of Computer Applications**, v. 183, n. 41, p. 24–32, 2021.

HE, D.; ZEADALLY, S. An analysis of rfid authentication schemes for internet of things in healthcare environment using elliptic curve cryptography. **IEEE internet of things journal**, IEEE, v. 2, n. 1, p. 72–83, 2014.

HUANG, P. et al. Practical privacy-preserving ecg-based authentication for iot-based healthcare. **IEEE Internet of Things Journal**, IEEE, v. 6, n. 5, p. 9200–9210, 2019.

HYPERLEDGER FOUNDATION. **The Ordering Service**. 2022. Avaliable at <https://web.archive.org/web/20240126204058/https://hyperledger-fabric.readthedocs.io/en/release-2.2/orderer/ordering_service.html>. Accessed on 2024/01/26.

JAOUDE, J. A.; SAADE, R. G. Blockchain applications–usage in different domains. **IEEE Access**, IEEE, v. 7, p. 45360–45381, 2019.

LAJAM, O. A.; HELMY, T. A. Performance evaluation of ipfs in private networks. In: **2021 4th International Conference on Data Storage and Data Engineering**. [S.l.: s.n.], 2021. p. 77–84.

MAYER, A. H. et al. Fogchain: a fog computing architecture integrating blockchain and internet of things for personal health records. **IEEE Access**, IEEE, v. 9, p. 122723–122737, 2021.

MAYMOUNKOV, P.; MAZIERES, D. Kademlia: A peer-to-peer information system based on the xor metric. In: SPRINGER. **International Workshop on Peer-to-Peer Systems**. [S.l.], 2002. p. 53–65.

NAKAMOTO, S. Bitcoin: A peer-to-peer electronic cash system. **Decentralized Business Review**, p. 21260, 2008.

NATIONAL CONGRESS OF BRAZIL. **Lei Geral de Proteção de Dados Pessoais (LGPD)**. 2018. Available at <http://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/Lei/L13709.htm>. Accessed on 2023/10/14.

ONGARO, D.; OUSTERHOUT, J. In search of an understandable consensus algorithm. In: **2014 USENIX annual technical conference (USENIX ATC 14)**. [S.l.: s.n.], 2014. p. 305–319.

PINTO, J. R.; CARDOSO, J. S.; LOURENÇO, A. Evolution, current challenges, and future possibilities in ecg biometrics. **IEEE Access**, IEEE, v. 6, p. 34746–34776, 2018.

POLITOU, E. et al. Delegated content erasure in ipfs. **Future Generation Computer Systems**, Elsevier, v. 112, p. 956–964, 2020.

RAY, P. P. et al. Biothr: Electronic health record servicing scheme in iot-blockchain ecosystem. **IEEE Internet of Things Journal**, IEEE, v. 8, n. 13, p. 10857–10872, 2021.

RAY, P. P. et al. Blockchain for iot-based healthcare: background, consensus, platforms, and use cases. **IEEE Systems Journal**, IEEE, v. 15, n. 1, p. 85–94, 2020.

SANTOS, J. A.; INACIO, P. R.; SILVA, B. Towards the use of blockchain in mobile health services and applications. **Journal of Medical Systems**, Springer, v. 45, n. 2, p. 1–10, 2021.

SHEN, J. et al. Understanding i/o performance of ipfs storage: a client's perspective. In: **Proceedings of the international symposium on quality of service**. [S.l.: s.n.], 2019. p. 1–10.

XU, J. et al. Healthchain: A blockchain-based privacy preserving scheme for large-scale health data. **IEEE Internet of Things Journal**, IEEE, v. 6, n. 5, p. 8770–8781, 2019.

YLI-HUUMO, J. et al. Where is current research on blockchain technology?—a systematic review. **PloS one**, Public Library of Science San Francisco, CA USA, v. 11, n. 10, p. e0163477, 2016.

ZHANG, J. et al. An efficient blockchain-based hierarchical data sharing for healthcare internet of things. **IEEE Transactions on Industrial Informatics**, IEEE, v. 18, n. 10, p. 7139–7150, 2022.

ZHENG, Z. et al. Blockchain challenges and opportunities: A survey. **International journal of web and grid services**, Inderscience Publishers (IEL), v. 14, n. 4, p. 352–375, 2018.

APPENDIX A — PUBLISHED PAPER - ISCC 2023

Laura Rodrigues Soares, Jéferson Campos Nobre, and Gabriel Kershcner. **Design of a Blockchain-based Secure Storage Architecture for Resource-Constrained Healthcare.** 28th IEEE Symposium on Computers and Communications (ISCC), Tunisia, 2023, pp. 1-6. DOI: 10.1109/ISCC58397.2023.10218178

- **Title:** *Design of a Blockchain-based Secure Storage Architecture for Resource-Constrained Healthcare.*
- **Contribution:** The architectural model for a security system combining blockchain, decentralized storage, and HIoT.
- **Abstract:** The Internet of Things (IoT) paradigm can improve a broad range of applications, such as medical sensors, smart cities, industrial monitoring, and so on. In healthcare specifically, these devices can aid in management tasks and improve the quality of life of patients in intensive care. However, securing medical IoT devices and its data is a crucial task. Not only do they handle Electronic Medical Records (EMR) and physiological information, but in some cases, disruption can affect a patient's treatment. Blockchain technologies applied in the healthcare context can provide privacy, immutability, decentralization, and easier access and sharing of medical data. Despite the emergence of applications aiming to solve security issues in the Healthcare IoT (HIoT) scenario using blockchain, there is still much to be addressed, mainly regarding throughput, storage of data, and efficient use of resources. This work proposes a blockchain-based storage architecture for HIoT, using a private network and distributed data storage to achieve integrity, accountability, and availability of medical data.
- **Status:** Published.
- **Qualis:** A2.
- **Conference:** 28th IEEE Symposium on Computers and Communications (ISCC).
- **Date:** 9 - 12 July, 2023.
- **Local:** Gammarth, Tunisia.
- **URL:** <https://2023.ieee-iscc.org/>
- **Digital Object Identifier (DOI):** <https://doi.org/10.1109/ISCC58397.2023.10218178>.

Design of a Blockchain-based Secure Storage Architecture for Resource-Constrained Healthcare

Laura Rodrigues Soares
Institute of Informatics
UFRGS
Porto Alegre, Brazil
lrsoares@inf.ufrgs.br

Jéferson Campos Nobre
Institute of Informatics
UFRGS
Porto Alegre, Brazil
jcnobre@inf.ufrgs.br

Gabriel Kerschner
Institute of Informatics
UFRGS
Porto Alegre, Brazil
gabriel.kerschner@inf.ufrgs.br

***Abstract*—The Internet of Things (IoT) paradigm can improve a broad range of applications, such as medical sensors, smart cities, industrial monitoring, and so on. In healthcare specifically, these devices can aid in management tasks and improve the quality of life of patients in intensive care. However, securing medical IoT devices and its data is a crucial task. Not only do they handle Electronic Medical Records (EMR) and physiological information, but in some cases, disruption can affect a patient's treatment. Blockchain technologies applied in the healthcare context can provide privacy, immutability, decentralization, and easier access and sharing of medical data. Despite the emergence of applications aiming to solve security issues in the Healthcare IoT (HIoT) scenario using blockchain, there is still much to be addressed, mainly regarding throughput, storage of data, and efficient use of resources. This work proposes a blockchain-based storage architecture for HIoT, using a private network and distributed data storage to achieve integrity, accountability, and availability of medical data.**

***Index Terms*—Blockchain, Internet of Things, Healthcare, Decentralized storage, Security.**

I. Introduction

The Internet of Things (IoT) paradigm has been successfully applied to numerous applications in the past few decades, such as industrial monitoring, agriculture, smart cities, and others. HIoT sensors (e.g., ECG and blood pressure monitors) can provide rapid feedback on patients in medical emergencies [1]. Most of the physiological data these sensors acquire can be reused as biometric signatures in authentication tasks, providing health diagnosis and security improvement at the same time [2]. Still, the benefits come with several challenges. HIoT applications must be able to deal with hundreds of varying capacity devices from diverse manufacturers requiring different types of management Gope and Hwang [3]. The volume of data they provide demands some level of data aggregation to achieve scalability in institution-wide deployments. From a security standpoint, special attention is due to health-related applications dealing with EMRs (e.g., medical examination and diagnosis data) and patient physiological information. Compliance legislations and significant privacy concerns demand a robust framework capable of providing the necessary security properties.

Blockchain has expanded to new areas after initially proposed with the Bitcoin cryptocurrency in 2008 [4]. IoT and healthcare, in particular, can benefit from decentralization, transparency, and immutability. However, the integration between blockchain and HIoT must be handled carefully. In addition to the standard challenges of health applications (data safety, privacy, compliance, and so on), IoT blockchain-based applications have to deal with storage limits, insufficient computational power to constantly validate transactions, and high throughput demands [5]. Any solution for HIoT must adequately address these concerns to employ blockchain usage advantageously.

Blockchain-based solutions for IoT and healthcare experienced a upraise over the last years [6]. Ray et al. [5] proposed an architecture to integrate these elements and offer secure access and management of IoT sensor data to healthcare service providers. Chen et al. [7] developed a system for medical data collection, storage, and sharing, designed to work across multiple hospitals and third-party organizations. Nevertheless, in the literature, there is still room for improvement in several topics related to blockchain integration. The amount of computational resources demanded by it, both in terms of processing capabilities and energy consumption, is far from negligible and should be optimized as much as possible to meet the target HIoT application requirements of resource constraints. Additionally, improvement of the current methods is also necessary in order to justify the use of the blockchain in favor of traditional alternatives that would achieve the same security properties demanding far less use of resources.

Considering the current challenges in integrating blockchain technologies into HIoT systems, this work aims to present a brief overview of the architectural components of related works on the topic, as well as to propose a secure and decentralized storage architecture for healthcare applications. This architecture is designed to handle patient EMRs and data provided by IoT sensors. It employs blockchain technology as means of indexing patients' data, providing decentralization, access control, accountability, and other key security properties. The data is stored off-chain in a decentralized, peer-to-peer storage system.

The remaining of this work uses the following structure. Section II presents key concepts, Section III has an overview of related state-of-art systems and their architectural choices. Section IV briefly outlines our solution's requirements and desired security properties before introducing the proposed

architecture. The results of experiments and performance evaluation are in Section V. Finally, Section VI presents the next steps in the development of this system and concludes the paper.

II. Background

This section divides the key concepts into two topics: the use of IoT in healthcare systems and the more recent employment of blockchain in the same scenario.

A. IoT Paradigm in Healthcare Systems

IoT has a broad range of use cases. In the healthcare environment, HIoT can improve patients' mobility, help the management of equipment and medication, and bring convenience to the medical team.

Most IoT systems follow similar architectural patterns. The key components are the sensor devices, the gateway nodes, and the Application Provider. Sensor devices in the same network can have different classes of processing resource constraints. Memory size can be used as a rough estimator of device capabilities. A device is considered constrained if ranging around 250 KiB of code size [8]. The communication with the sensor is done through gateway nodes. These nodes can also vary in capacity, from a Raspberry Pi to the user's smartphone to an actual computer, but are usually more computationally capable than the sensors. Finally, the application provider can be, e.g., a health service available to patients and medical staff on a remote server. It is often also in charge of storing the acquired data using traditional centralized data centers, cloud services, or others.

Solutions for HIoT often struggle to meet all the desired security properties and also provide scalability. Blockchain technologies can offer improved security and easier access to medical data.

B. Blockchain-based Healthcare

Blockchain has had deep mediatic coverage in the past few years, unfortunately not always associated with its redeeming qualities of providing privacy, anonymity, decentralization, and immutability. As a data management technology, it allows a decentralized environment where no third party is required to oversee transactions between two concerned parties [6]. Transactions are time-stamped and broadcast to all the nodes participating in the system, making it publicly auditable. The information is stored in a block linked with the previous block in a way that guarantees its immutability since it would be necessary to alter the entire chain to tamper with a block. In the initial research initiatives until 2016, about 80% of the research on the topic was related to the Bitcoin system, and not much was dedicated to other industries [9]. Publications about blockchain applications peaked in 2017 and 2018, moving from cryptocurrency to other emergent fields [6], such as IoT, energy, finances, and healthcare.

Each application does some modifications to the blockchain protocol in order to attend to its unique features. Traditional health system struggles with interoperability problems, poor auditability, data leakage of sensitive information, and the data ownership debate [10]. Blockchain can provide easier access to medical data, facilitate the sharing of medical records, and aid in standardization efforts across different institutions [6].

While blockchain technology has experienced a boom of works on the topic, there is still much room for improvements regarding the efficient use of resources. This research is fundamental for integration with the IoT paradigm.

III. Related Works

Integrating a blockchain-based architecture and the HIoT paradigm has great potential for addressing long-lasting security issues in the industry. This section presents some of the works on this topic.

Akkaoui et al. [11] goal was to divide the mining process into two layers of consortium private blockchains. The first runs on the Edge, near the sensor devices, and acts as a personal node performing tasks of registration, authentication, and reception of data. The second layer is a global blockchain operating as a database of hash pointers for medical data, which are stored off-chain using the Interplanetary File System (IPFS), a distributed P2P file system. Ali et al. [12] employs blockchain technology aiming to avoid sharing data with a trusted third party. They use a public blockchain for accountability purposes and the Tor distributed file system to relay patient data directly to doctors. Chen et al. [7] designed their system to include a central system administrator as a trusted third party in order to meet regulatory and compliance requirements for medical institutions. The private blockchain acts as an index record of data and digital signatures located in cloud servers. Mayer et al. [13] had a proposal of integrating a fog computing layer closer to the sensor devices of an IoT network. Their work aimed to avoid centralized cloud storage and to use the blockchain for this goal instead. Xu et al. [14] created their own implementation of a blockchain network and divided it into two layers. Patient data is hashed and indexed in the public *userchain*, while stored in IPFS nodes. A separate consortium blockchain for diagnostic data is used by doctors, in which only authorized nodes can add a transaction to the chain.

Table I summarizes the key architectural components of the related works mentioned in this section. However, not every solution aims specifically for resource-aware implementation choices. This work aims to develop a secure storage system for the HIoT that employs state-of-art, resource-aware methods in a blockchain-based architecture.

IV. Proposed Solution

It is not uncommon to combine blockchain and IoT in medical settings. Figure 1 presents the key component generally found in blockchain-based IoT architectures. The IoT devices exchange sensor data with gateway nodes of considerable computational capability. The gateway then interacts with one or several blockchain networks for tasks such as registration, authentication, data storage, and so on. Since the blockchain itself is unsuitable for storing large files, most systems employ

TABLE I
OVERVIEW OF RELATED WORKS.

Author	Platform	Privacy	Consensus	Storage
Akkaoui et al. [11]	Ethereum	Consortium	PoA, PoW	P2P file system (IPFS)
Ali et al. [12]	Ethereum	Public	PoA	P2P file system (Tor)
Chen et al. [7]	Hyperledger Fabric	Private	Kafka	Cloud
Mayer et al. [13]	Hyperledger Fabric	Consortium	PBFT	On-chain
Xu et al. [14]	-	Public and Consortium	PoW, PBFT	P2P file system (IPFS)

a separate storage unit that can either be cloud, decentralized, or a traditional central data server. An application then accesses the data to provide a variety of medical services.

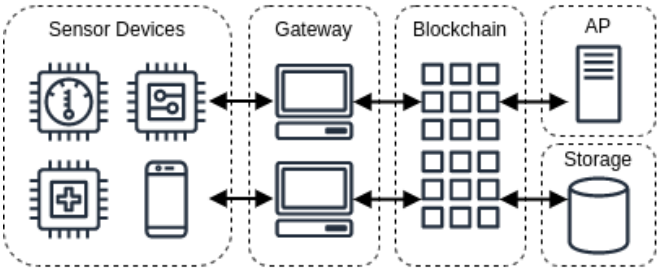


Fig. 1. Example of a generic architecture using both IoT and blockchain.

There are several ways of combining the components of a blockchain-based IoT architecture. Each arrangement can offer different security properties and functionality suitable for distinct application requirements. This section discusses some of these requirements and outlines a scheme to afford them.

A. Requirements

This section classifies the functionality requirements expected from a blockchain-based solution comprehending IoT and medical records into four categories.

1) Resource Consumption: The employment of blockchain in any system must be carefully evaluated, lest it can harm the performance of the application before it can bring substantial benefits. The consumption of energy and resources should be lower or at least the same as traditional techniques capable of providing the same functionality. Some architectural choices that can help attain resource economy are lightweight consensus algorithms and small-sized ledger files (also called assets). Another performance factor that must be investigated is the cost of the minimum transaction throughput required by the application. Finally, not all of the performance requirements are related to the blockchain. Medical files must be encrypted before storage and decrypted back into a human-readable format before being handled back to the user, which adds to the system's overhead.

2) Data Ownership: In the scenario of this work, the patient, not the organization, should be the owner of the data stored in the system. This would allow them to pursue treatment across different medical facilities enrolled in the network and request the deletion of the data in the case they no longer require medical assistance.

3) Compliance: An architecture must follow the applying compliance legislation. All the personal data stored must be capable of deletion, and the system should guarantee that no past version of this data can be accessed from a previous version of the ledger. Storing medical data directly on-chain would violate such codes since ledger blocks are immutable. Off-chain storage must enable the removal of personal health records per user request.

4) Security Properties: Privacy and confidentiality are essential in any medical system. No user, apart from the patient or authorized by the patient (e.g., the patient's doctor at an appointment), should be allowed to access the stored medical data. The data should also be protected from other people within the same network with access to the ledger, and from eavesdroppers/adversaries. Finally, the system should guarantee that the data the user receives has not been tampered with.

B. Proposed Blockchain-based Secure Architecture for HIoT

This section presents the system's functionality based on the requirements discussed in Subsection IV-A.

Firstly, a public blockchain network would be inappropriate for handling HIoT data since these networks demand more expensive consensus algorithms, and the data should not be publicly accessible. Instead, multiple health facilities join a federation blockchain network, in which the participants agree beforehand on the contracts, operations, and capabilities of the peers. The organization that first admits a patient is responsible for authenticating its enrollment on the system and generating the patient's personal key. This key can later be used in tasks such as patient authentication and the encryption of medical files.

Due to the distributed nature of the network, the patient can pursue treatment in different health centers and access medical data using the same personal key. This would be possible through a decentralized file storage system in combination with the distributed ledger framework.

The basic functionality of the HIoT system is to act as a data index for all the Electronic Medical Records (EMR) of a patient, with extra steps for authentication, integrity checks, and confidentiality. The medical staff requests the patient's personal key, e.g., during an appointment, to access a health report from a previous exam stored in the system. The same process can add new reports to the patient's record. Similarly, a sensor device attached to the patient is able to keep an updated sensed value stored in the system. This functionality in particular will heavily rely on the blockchain performance in

means of throughput and ledger size scalability, since it will need frequent transactions. If the patient no longer requires medical assistance and decides to remove their data from the system, then the records with the associated identity are removed from storage.

Figure 2 presents the outline of the proposed scheme. The sensor devices and end-point hosts from either users or doctors belong in the device layer. The communication with the system goes through an intermediate layer encompassing the IoT gateway nodes, the blockchain network peers, and distributed storage nodes. The performance evaluation of the consensus algorithm and additional cryptography overhead dictates whether these components are located in a single high-capacity device or small resource-constrained ones. At last, one or several authenticated applications located in the application layer can be provided access to the network to provide medical and operational services.

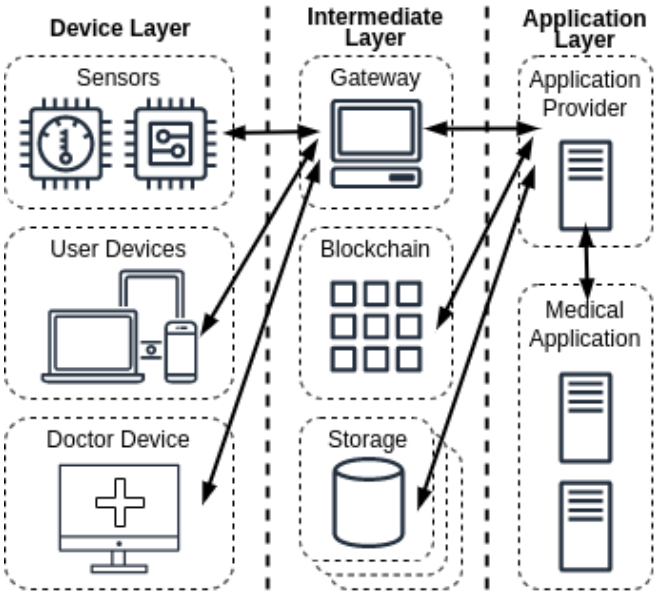


Fig. 2. Proposed architecture integrating blockchain, IoT and distributed storage.

V. Evaluation

This section presents the frameworks used in the proof-of-concept design. Then, it details the implementation and shows the performance experiments of the architecture.

A. Experimental Setting

This work uses Hyperpedger Fabric, the blockchain environment standing out the most in the state-of-the-art. It employs modular components and is considered lightweight [7]. The consensus mechanism used is the Raft ordering service that comes by default with Fabric 2.x. A private network in the IPFS system is used for distributed storage.

1) Hyperledger Fabric: Fabric is a consortium permissioned blockchain that allows multiple organizations, each with enrolled peers capable of proposing transactions to the network participants. Each organization has its own Certificate Authority that employs a Public Key Infrastructure (PKI) to sign its peers' transactions and assets. The communication mechanism by which the members exchange messages is called a channel. There can be multiple channels in a network, providing private communication between members. Organizations also own ordering nodes besides regular peers. The main role of an orderer is to guarantee ledger consistency, which prevents forks of divergent ledger states. Finally, the chaincode is the rough equivalent of a smart contract - the definition of rules concerning the transaction logic of the network.

The transaction flow of the Fabric ordering service works as follows. Firstly, upon request from the application, the peers produce a proposal for ledger update based on a smart contract previously agreed on and submit it to an orderer node. The ordering service then arranges all the submitted transactions into a sequence of blocks that are distributed back to the peers for validation. Each peer checks for the necessary endorsements and possible conflicts and, finally, commits the blocks to the current state of the ledger.

2) IPFS: The IPFS system employs content addressing instead of the location addressing commonly used on the internet. Each file is divided into blocks, and the content of each block is hashed to generate a unique Content Identifier (CID). This means that two identical files will generate the same CID, eliminating the need for double storage. A consequence of this design is that a new CID must be generated every time a file is modified. Each node in the system keeps a Distributed Hash Table (DHT), mapping a selection of neighbor nodes, their addresses, and the CIDs they host.

IPFS requires some adjustments before it can be applied to healthcare settings. Its core functionality requires the CIDs, the participating nodes, and DHTs to be public. A third party monitoring the traffic to the DHT can determine who requests which CID. Access to the network can be restricted to a selection of trusted nodes through a private IPFS network, wherein the nodes must have a swarm key to join. In addition, privacy-critical files must be encrypted before storage in the system as, while the data transfer between nodes is secure, anyone who holds the CID might request the corresponding file. IPFS lacks authentication and the ability to track access[15], a common reason why blockchain is often employed in IPFS applications. However, the usual practice of simply storing the CID in a distributed ledger presents a few drawbacks. It is a rather computationally expensive solution for adding a signature to data and has the substantial disadvantage of making the file, its owner, and its location known to all participants of the ledger network. To address this issue, the proposed solution employs a vault component outlined in the following section.

B. Simulation Experiments

1) System Workflow: The workflow has three main tasks: initialization, storage, and access of files. The step-by-step is detailed as follows.

Initialization. Upon admitting a patient for the first time in the network, the health facility must generate a Patient ID for indexing in the ledger. Then, the patient must generate a

Personal Key using traditional methods (password, passphrase) or biometric signals (fingerprint, electrocardiogram). The system then creates a Vault file, encrypts it using the Personal Key, and adds it to the IPFS system, which returns the Vault CID. Finally, the Patient's Medical Asset is created using the pair PatientID, VaultCID and added to the ledger. The use of "asset" as a word choice is due to a common terminology in blockchain spaces and is maintained for clarity even though it has no monetary value in this context. By the end of the initialization step, the patient is successfully enrolled in the system, and the address of their vault is publicly accessible within the network.

Storage of medical records. The storage of new files is a four-step process.

i. Encryption step. First, the report must be encrypted using the Personal Key of the patient. The encrypted file is then added to the IPFS storage system, obtaining the file CID.

ii. Vault retrieval. The system queries the ledger about the Patient's Medical Asset to get the Vault CID, which is retrieved from the IPFS system. If the Vault CID in the ledger no longer matches the CID of the stored Vault file, it's a good indicator that the file is corrupted. After obtaining the correct file, the Vault must be decrypted using the same Personal Key.

iii. Indexing. The CID of the encrypted report is then added to the Vault file, which acts like an index of all the medical records the patient has stored in the system. The Vault file is encrypted back using the patient's key.

iv. Updating the Medical Asset. Since the Vault file has been altered, a different Vault CID will be generated upon returning it to the IPFS system. The system must then perform a blockchain transaction to update the Patient's Medical Asset to PatientID, NewVaultCID. This step ensures that all parties are properly authorized, and the modification is legitimate.

Read-only access. The reading process is considerably simple: obtaining the Vault CID from the ledger, fetching the file from the IPFS system, decrypting it, getting the desired medical file CIDs, and retrieving and decrypting it. These steps must be performed in the presence (or with the authorization) of the patient, since the Personal Key is necessary. A read-only query does not alter the blockchain ledger, so the final step of updating the Patient's Medical Asset unnecessary. If any unauthorized modification is made, the corrupted file CID will no longer match the official one stored in the vault.

The described system workflow is detailed schematically in Figure 3. It inherently provides some of the desired security properties mentioned in Section IV-A, such as integrity and authenticity, due to the nature of content addressing. In addition, by not updating the CID of every file directly into the ledger, we reduce the size of the Patient's Medical Asset to be stored in-chain. We can also guarantee that the actual medical records can be removed from storage upon request.

Some use cases will require constant updating of medical files (e.g., keeping a file with a fresh sensed value stored in the system), which will, in turn, require several transactions to be performed in a time unit. The following subsection brings the performance results of the described transactions.

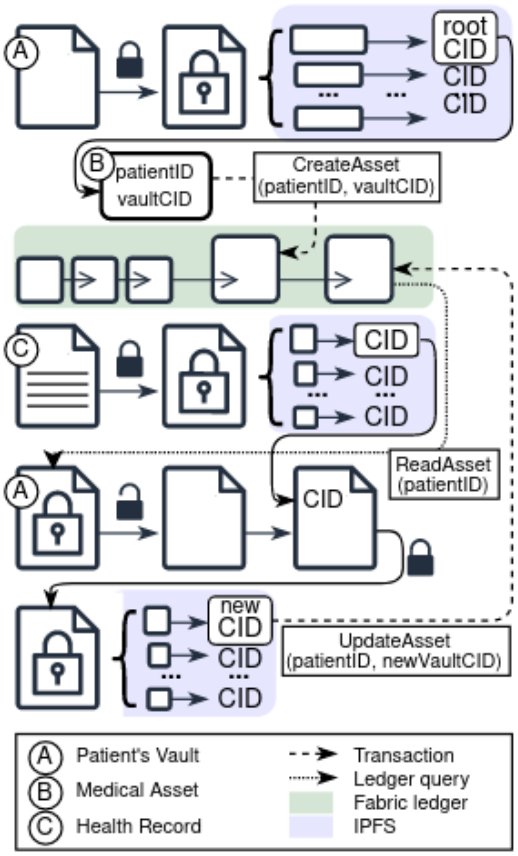


Fig. 3. The operational flow of the proof-of-concept system.

2) Results: The use case of a sensor device keeping a reasonably updated measured value stored in the system will demand a considerable amount of updating transaction requests. This performance analysis intends not to provide real-time information but to specify if the sensed value can be as recent as the last hour or as the last two minutes.

Figures 4 and 5 present a preliminary evaluation of the chaincode implementing the operations described in Subsection V-B1. The tests were performed using Hyperledger Caliper, a blockchain benchmark tool. The transaction load parameter means the fixed number of transactions being requested and processed by the system. Figure 4 shows the achieved throughput under the variating load. Figure 5 presents the latency in seconds under the same circumstances. The performance tests show that even the most extreme use case of keeping an updated sensed value is still feasible with the current architecture.

VI. Conclusion and Ongoing Work

This paper proposes and evaluates a blockchain-based solution for improving the security of medical systems, particularly considering IoT use cases. The proposed system employs a consortium blockchain and distributed content-addressing storage to provide the requirements of this scope. In order to avoid storing the address of medical files directly into the

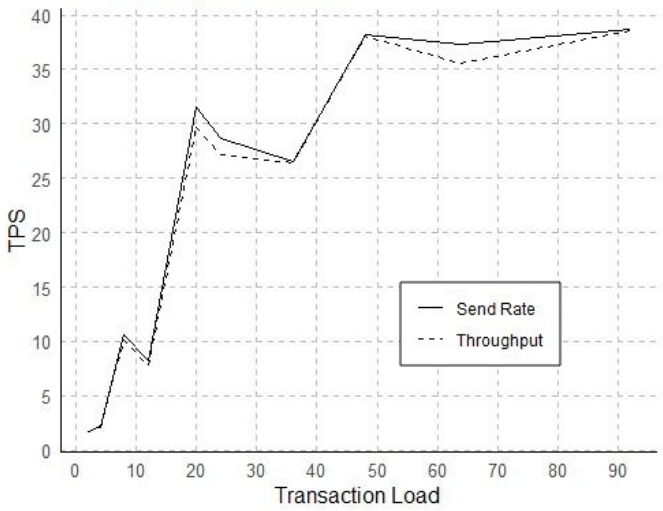


Fig. 4. Transactions Per Second (TPS) obtained under varying load.

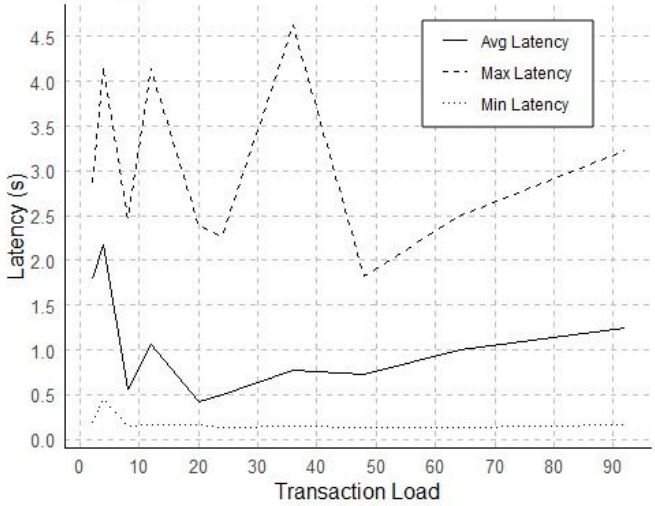


Fig. 5. Latency of the transactions in seconds.

blockchain ledger, the architecture uses a vault component that stores the location of the EHRs and allows the deletion of personal files in compliance with most data protection laws. We evaluate the proposed architecture through a benchmark tool, and the performance is feasible for the proposed use case. The future research is to simulate the complete scope of the system - IPFS network, sensor devices, and health application, as well as to further analyze the performance of each component.

REFERENCES

[1] A. Ghubaish, T. Salman, M. Zolanvari, D. Unal, A. Al-Ali, and R. Jain, "Recent advances in the internet-of-medical-things (iomt) systems security," *IEEE Internet of Things Journal*, vol. 8, no. 11, pp. 8707–8718, 2020.

[2] P. Huang, L. Guo, M. Li, and Y. Fang, "Practical privacy-preserving ecg-based authentication for iot-based healthcare," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 9200–9210, 2019.

[3] P. Gope and T. Hwang, "Bsn-care: A secure iot-based modern healthcare system using body sensor network," *IEEE sensors journal*, vol. 16, no. 5, pp. 1368–1376, 2015.

[4] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Business Review*, p. 21260, 2008.

[5] P. P. Ray, D. Dash, K. Salah, and N. Kumar, "Blockchain for iot-based healthcare: background, consensus, platforms, and use cases," *IEEE Systems Journal*, vol. 15, no. 1, pp. 85–94, 2020.

[6] J. Abou Jaoude and R. G. Saade, "Blockchain applications–usage in different domains," *IEEE Access*, vol. 7, pp. 45 360–45 381, 2019.

[7] Z. Chen, W. Xu, B. Wang, and H. Yu, "A blockchain-based preserving and sharing system for medical data privacy," *Future Generation Computer Systems*, vol. 124, pp. 338–350, 2021.

[8] C. Bormann, M. Ersue, and A. Keranen, "Terminology for constrained-node networks," *Internet Engineering Task Force (IETF): Fremont, CA, USA*, pp. 2070–1721, 2014.

[9] J. Yli-Huumo, D. Ko, S. Choi, S. Park, and K. Smolander, "Where is current research on blockchain technology?—a systematic review," *PloS one*, vol. 11, no. 10, p. e0163477, 2016.

[10] J. A. Santos, P. R. Inacio, and B. Silva, "Towards the use of blockchain in mobile health services and applications," *Journal of Medical Systems*, vol. 45, no. 2, pp. 1–10, 2021.

[11] R. Akkaoui, X. Hei, and W. Cheng, "Edgemedichain: a hybrid edge blockchain-based framework for health data exchange," *IEEE access*, vol. 8, pp. 113 467–113 486, 2020.

[12] M. S. Ali, M. Vecchio, G. D. Putra, S. S. Kanhere, and F. Antonelli, "A decentralized peer-to-peer remote health monitoring system," *Sensors*, vol. 20, no. 6, p. 1656, 2020.

[13] A. H. Mayer, V. F. Rodrigues, C. A. da Costa, R. da Rosa Righi, A. Roehrs, and R. S. Antunes, "Fogchain: a fog computing architecture integrating blockchain and internet of things for personal health records," *IEEE Access*, vol. 9, pp. 122 723–122 737, 2021.

[14] J. Xu, K. Xue, S. Li, H. Tian, J. Hong, P. Hong, and N. Yu, "Healthchain: A blockchain-based privacy preserving scheme for large-scale health data," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8770–8781, 2019.

[15] J. Hanafi, Y. Prayudi, and A. Luthfi, "Ipfschain: Interplanetary file system and hyperledger fabric collaboration for chain of custody and digital evidence management," *International Journal of Computer Applications*, vol. 183, no. 41, pp. 24–32, 2021.

APPENDIX B — RESUMO EXPANDIDO

Sistemas IoT (do inglês Internet of Things, ou Internet das Coisas) são usados com sucesso em diversas aplicações como monitoramento industrial, agricultura, cidades inteligentes, e outras. Na área da saúde, esses dispositivos são chamados de sensores HIoT (do inglês, Healthcare IoT, ou IoT em Saúde). São exemplos sensores de monitoramento cardíaco, do sono, de oxigenação, e vários outros. Eles oferecem respostas rápidas durante o tratamento de pacientes em situações como prática de exercícios e emergências médicas, e trazem benefícios para a equipe médica responsável. Porém, soluções de segurança para sistemas HIoT tem requisitos complexos. A maioria dos sensores tem baixa capacidade computacional, geram um grande volume de dados, e precisam de disponibilidade constante. Outros desafios incluem a heterogeneidade dos dispositivos e protocolos vindos de fornecedores variados, e, principalmente, a necessidade de atenção ao lidar com dados biológicos e de saúde. Blockchain, por sua vez, é uma tecnologia de popularidade emergente e cuja aplicabilidade pode fornecer propriedades de segurança valiosas para sistemas HIoT, como por exemplo, decentralização, transparência, e imutabilidade. Alguns desafios da aplicação de redes blockchain em sistemas HIoT são os limites de armazenamento e poder computacional dos dispositivos e a demanda por uma taxa de rendimento elevada.

No estado da arte, a motivação mais popular no uso de blockchain em sistemas HIoT é para controle e gerenciamento de acesso, mas também para responsabilização e para manter registros. Apesar de haver vários trabalhos no tópico, é necessário que se expanda a pesquisa existente com foco no uso eficiente e econômico de recursos. Blockchain é uma tecnologia que demanda vastas quantidades de energia e poder computacional, e que deve ser otimizada o melhor possível para obedecer as demandas de sistemas HIoT. Uma melhoria dos algoritmos atuais é necessária para justificar o uso de blockchain no lugar de sistemas tradicionais que oferecem a mesma segurança em troca de bem menos gasto energético. Esse trabalho busca desenvolver um sistema de armazenamento seguro para HIoT que use uma arquitetura baseada em blockchain e métodos que façam uso consciente de recursos.

A arquitetura proposta busca seguir requisitos de desenvolvimento divididos nas seguintes quatro categorias. Primeiro, o consumo de recursos idealmente deveria ser menor do que soluções tradicionais que oferecem a mesma funcionalidade. Algoritmo de consenso, tamanho das transações, e as técnicas de criptografia utilizadas devem ser

econômicos dentro do possível. Sobre a propriedade dos dados, o requisito principal é que esteja a cargo do paciente ao invés da organização. Isso permite que se busque tratamento em qualquer unidade de saúde parte da federação. Outro requisito é a observação das regulações de conformidade. Qualquer sistema lidando com dados de saúde está sujeito à legislação de proteção de dados vigente, e a LGPD exige que dados pessoais possam ser deletados caso solicitado pelo usuário. Por último, os requisitos de segurança principais são confidencialidade (ou seja, que apenas as pessoas autorizadas possam acessar os dados médicos do paciente), integridade (garantir que os dados entregues pelo sistema não sofreram alterações), e disponibilidade (em que as demais medidas de segurança não impeçam que os dados sejam entregues ao solicitante em tempo hábil).

Para observar os requisitos de desenvolvimento o sistema proposto utiliza uma rede blockchain de consórcio. Isto garante que os participantes estão previamente autenticados por suas organizações e permite o uso de algoritmos de consenso menos custosos. Os dados médicos são criptografados antes do armazenamento que é feito em um sistema distribuído, já que ambientes em nuvem estão sujeitos a uma variedade de vulnerabilidades de segurança e são complicados de se gerenciar em ambientes federados. Nesse contexto, a blockchain funciona como um índice dos arquivos de um determinado paciente que estão armazenados no sistema. Além disso, para evitar que todos os endereços dos arquivos médicos do paciente fiquem salvos diretamente na blockchain, o sistema utiliza um arquivo "cofre" que fica criptografado e armazenado fora dela. Isso evita o armazenamento de dados pessoais imutáveis, adiciona mais uma camada de segurança e privacidade ao sistema, e permite transações menos custosas. Um esquema da arquitetura proposta está detalhado na Figura B.1. Além dos pacientes, outros usuários são médicos e entidades de saúde afiliados às organizações parte do consórcio. Esses usuários usam seus dispositivos para acessar registros médicos através de uma camada intermediária, composta por gateways, nodos da blockchain e de armazenamento. Por último, aplicações médicas autorizadas acessam a rede para forneces serviços operacionais e de saúde.

A implementação da arquitetura proposta como prova de conceito foi feita usando a plataforma de código aberto Hyperledger Fabric para a blockchain. O sistema de consenso usado por padrão no Fabric é baseado em Raft e usa nodos ordenadores em adição aos nodos regulares. Isso delega a proposta e a validação de transações para conjuntos diferentes de participantes. Para o sistema de armazenamento distribuído, o IPFS (do inglês Interplanetary File System, ou Sistema de Arquivos Interplanetário) foi o protocolo selecionado devido a sua crescente proeminência na literatura. O IPFS usa endereçamento

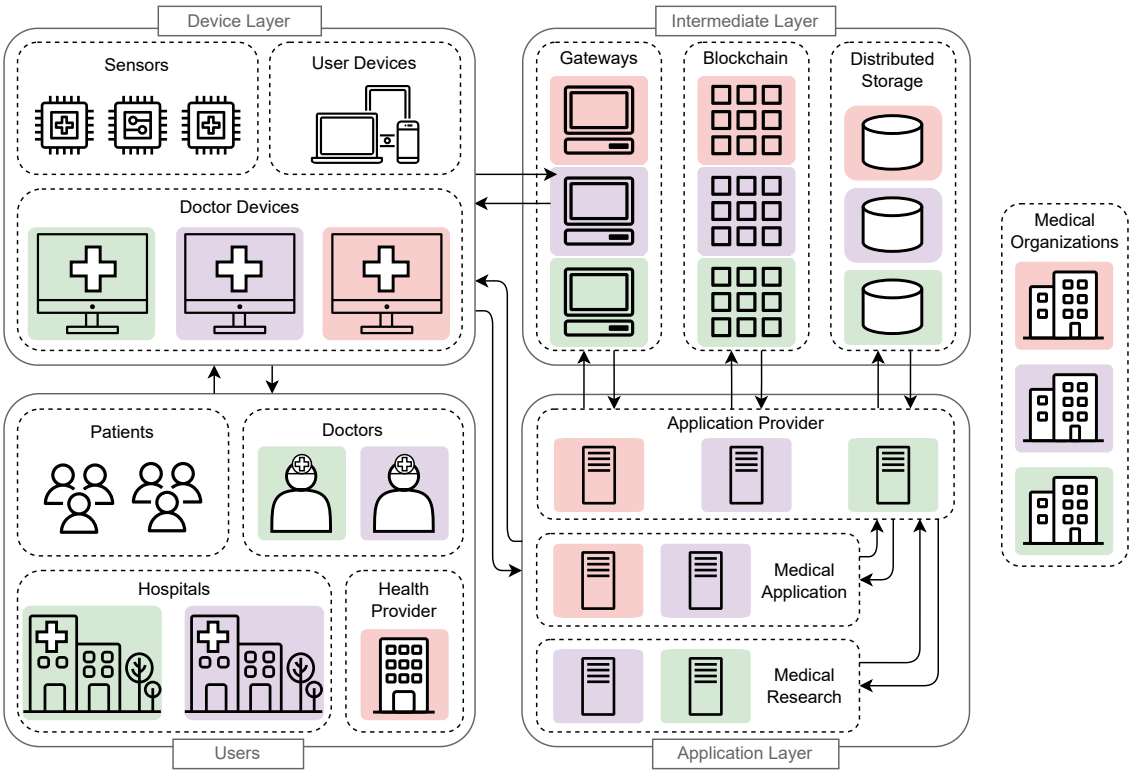


Figure B.1 – Arquitetura proposta integrando blockchain, HIoT, e armazenamento distribuído.

baseado em conteúdo, e um endereço é chamado de CID (do inglês, Content Identifier). Uma consequência disso é que arquivos idênticos geram endereços idênticos, e um novo endereço precisa ser gerado caso um arquivo seja modificado. O IPFS divide os arquivos em blocos de mesmo tamanho antes de adicioná-los ao sistema, semelhante ao protocolo BitTorrent, com a diferença que arquivos semelhantes podem compartilhar blocos. Além disso, o funcionamento do IPFS demanda que os CIDs e suas tabelas de endereçamento sejam públicas, portanto, é necessário criptografar arquivos sensíveis e utilizar uma rede privada no contexto desse trabalho.

O aplicativo desenvolvido para o reste de performance implementa as funções de leitura de arquivos e adição de novos arquivos ao sistema. As operações demandam que seja obtido o registro médico do paciente na blockchain, que vai fornecer o endereço do arquivo-cofre armazenado no IPFS. Esse arquivo precisa ser descriptografado com as credencias do paciente para que os endereços dos registros médicos sejam acessados. Para a leitura de arquivos, basta buscar no IPFS pelo endereço obtido e, em seguida, descriptografar o arquivo médico. Para a adição de novos arquivos, é necessário criptografá-los, adicioná-los ao IPFS, e armazenar o CID de volta no arquivo cofre. Ao ser criptografado e devolvido ao IPFS, o cofre irá gerar um CID diferente devido à alteração. Uma transação na rede blockchain será necessária para atualizar o registro médico do paciente com o

novo endereço do cofre. O teste de performance da aplicação levou em consideração a taxa de rendimento da blockchain sob um numero variável de transações concorrentes, e a latência que o IPFS demanda para acompanhar essa taxa. Apesar de os resultados de latência e rendimento da blockchain serem satisfatórios, o sistema de armazenamento se mostrou um gargalo para o desempenho do sistema devido aos tempos de upload e download dos blocos de arquivos. A Tabela B.1 apresenta uma comparação do tempo médio de execução de cada uma das operações do sistema. É possível perceber que mesmo com a latência do IPFS o sistema ainda apresenta uma latência dentro do admissível para o caso de uso em questão.

Table B.1 – Tempo de execução de cada operação do sistema, em milisegundos, para arquivos de 1 Mb.

	rA	iG_V	d_V	iG_F	d_F	e_F	iA_F	e_V	iA_V	uA	*total*
read (ms)	16.844	6.645	0.134	5586.85	8.235	-	-	-	-	-	**5618.70**
add (ms)	34.132	9.009	0.175	-	-	4.796	2318.3	0.155	832.13	2097.3	**5295.99**

Os próximos passos dessa pesquisa são o teste de outros algoritmos de consenso e seu impacto no desempenho da blockchain, e o uso de técnicas de criptografia que permitam o acesso granular aos dados em circunstancias além da presença do paciente. Além disso, existe a possibilidade de aprimorar a implementação teste atual com uma interface gráfica, e de analisar o impacto de diferentes tamanhos dos blocos de arquivo no IPFS na performance do sistema.