International Journal of Business Data Communications and Networking
Volume 19 • Issue 1

Blockchain-Based Data Storage for Internet of Things Networks

Wassim Jerbi, CES Lab, National School of Engineers of Sfax, University of Sfax, Sfax, Tunisia & Higher Institute of Technological Studies of Sfax, Tunisia*

https://orcid.org/0000-0002-1618-7469

Omar Cheikhrouhou, Higher Institute of Computer Science of Mahdia, University of Monastir, Mahdia, Tunisia & CES Lab, National School of Engineers of Sfax, University of Sfax, Sfax, Tunisia

Abderrahmen Guermazi, Higher Institute of Technological Studies of Sfax, Tunisia

Hafedh Trabelsi, CES Lab, National School of Engineers of Sfax, University of Sfax, Sfax, Tunisia

ABSTRACT

Consensus mechanisms are fundamental to blockchain technology, ensuring network integrity and the orderly progression of transactions. However, the predominant proof of work method, while widely adopted, has raised concerns regarding its energy-intensive nature and lack of scalability. In response, this research explores alternative consensus methods tailored to participants' engagement in blockchain storage activities. We conducted a comprehensive review of existing approaches, examining their reusability and efficacy. Concurrently, we observed a growing demand for distributed storage solutions driven by escalating data volumes. Our investigation identified areas for improvement in existing storage blockchains, motivating the development of our own system, BlockStock. This protocol, meticulously designed, aims to enhance intelligent data storage management reliably and robustly. Through rigorous performance evaluations, including assessments of power consumption, throughput, and data transfer times, BlockStock demonstrates superior efficacy and cost-effectiveness. These findings underscore its significance as a notable advancement in blockchain-based storage solutions, offering a promising avenue for future research and application.

KEYWORDS

Blockchain, Blockstock, Consensus, Decentralized, Ethereum, Secure, Smart Contract, Storage Data

1. INTRODUCTION

A blockchain serves as a distributed and decentralized database, enabling secure data storage and transmission without reliance on a central authority. Each participant autonomously maintains a record of all network activities, ensuring decentralization as a fundamental characteristic (Zhang et al., 2024). This distributed nature safeguards against malicious nodes attempting to manipulate data or alter the blockchain's history. Over recent years, blockchains have gained significant popularity. As the user base and transaction volume continue to grow, active participation in the system becomes

DOI: 10.4018/IJBDCN.341590 *Corresponding Author

increasingly challenging for users with limited resources (Khallel et al., 2023). In a conventional blockchain setup, nodes contribute to the network by validating new blocks and securely storing the entire blockchain. While this distributed storage mechanism effectively protects against data tampering, the widespread adoption of blockchains has raised barriers for users with constrained capabilities to actively engage in the system (Wang et al., 2023).

We are introducing a groundbreaking client known as "low storage," aimed at overcoming critical challenges in scalability. This innovative approach involves storing only coded fragments of the blockchain, rather than the entire chain. The process involves disassembling original blocks into fixed-size pieces, which are then encoded using linear fragment combinations (Jerbi et al., 2022). This methodology effectively addresses two out of the three major scalability issues: the limited storage space required for the complete blockchain and network congestion caused by an insufficient number of nodes. Upon conducting extensive research into existing storage blockchains, we identified areas for improvement based on a thorough examination of their characteristics and differences. Motivated by these findings, we have developed our own system to enhance the current state of blockchain technology (Zhang et al., 2023).

In a subsequent phase, we present BlockStock—a cutting-edge, smart contract-based solution designed for leasing storage space among blockchain nodes. A distinctive feature of BlockStock is its ability to record all transactions on the blockchain, ensuring transparency. Furthermore, the system incorporates regular and automated audits performed by the entire network, facilitated by recoverability proofs (Pourmajidi et al., 2023). This dual approach not only optimizes storage efficiency but also enhances security and accountability within the blockchain ecosystem (Jerbi et al., 2020).

The paper presents several notable contributions, including:

1. **Smart Contract Implementation:** Introducing a robust smart contract system wherein each new transaction undergoes validation and execution by the nodes representing the involved actors.
2. **Storage Capacity Management:** Addressing the challenge of blockchain size expansion by acknowledging the escalating workload associated with storing larger blockchains.
3. **Network Load Management:** Recognizing the impact of a peer-to-peer network structure on nodes, emphasizing that as the blockchain distribution increases, both the network load and associated fees become more distributed. However, the trade-off is a reduction in the number of nodes holding the complete blockchain, leading to increased network fees for those retaining the entirety of the blockchain.
4. **Optimizing Storage Costs:** Emphasizing the importance of minimizing bandwidth and computation costs in responding to challenges, with a goal to outperform the expense of downloading the entire file. This approach aims to ensure a high level of certainty regarding the presence of the file on the server.

In summary, the paper offers a comprehensive framework that addresses key aspects such as smart contracts, storage capacity, network load, and storage cost optimization in the context of blockchain technology.

This paper is structured as follows: In Section Two, we delve into related work. Following that, Section Three introduces our BlockStock protocol. Section Four is dedicated to the evaluation and performance analysis of the proposed protocol. Finally, Section Five encapsulates the conclusion and future work.

2. RELATED WORK

Each block within a Blockchain consists of two integral components: a block header and the block itself. Blocks serve as repositories for transaction records or transactions, encompassing a wide array

of data such as health records, financial transactions, traffic data, system logs, and more (Buterin, 2013). Block headers incorporate two sets of metadata: one for mining, comprising timestamps, Nonce values, and difficulty objectives; and the other for the block itself, including Merkle tree roots (Wood, 2014) and fields linking parent blocks and version numbers. The Merkle tree's leaf node is derived by hashing all transaction records in the block twice, and recursively obtaining hash values of adjacent nodes until the final hash result, known as the Merkle root, is achieved. Transactions are bundled and transmitted to blockchains in block format, with cryptographic techniques linking all blocks in a predefined sequence, forming an organized chain structure.

Several innovators have proposed solutions leveraging blockchain and existing storage systems, and we will explore some of the most compelling projects. Sia coin (Vorick & Champine, 2014), is both a cryptocurrency and a toolkit built upon the Bitcoin network. Its primary objective is to enable individuals to store or have their data stored in exchange for tokens. While only the lease agreement is recorded on the blockchain (offchain), the remaining data is not. Consequently, there is no concrete process for validating the integrity of servers, and the system heavily relies on a reputation system. This system, among other factors, considers a financial commitment made by each node to demonstrate their dedication to the system's proper functioning. To enhance the likelihood of successfully recovering data at the end of the contract, the data is stored on multiple servers using encryption codes.

Storj (Wilkinson et al., 2014), stands as a decentralized cloud system built on the Ethereum blockchain. The storage process involves breaking the file into 20 pieces, numbering and coding them, and subsequently distributing them across 40 different servers (the default choice). Satellites manage metadata, allocate data to servers, and conduct regular audits to ensure continuous data storage. In the event of servers failing to store data, replication occurs to maintain redundancy (Jerbi et al., 2021 & Jerbi et al., 2020). BlockStore (Ruj et al., 2018) a pioneer in outlining the file storage process, employs Space Wallets—a specialized structure tracking storage space across nodes. To mitigate the impact on chain size, it primarily monitors proof failures, leaving other functions such as audits and transfers to be handled off-chain. Notably, while this system efficiently retrieves files at the end of storage, it does not automatically address potential legal issues.

Spacemint (Park et al., 2018) represents a blockchain system where mining involves generating storage proofs through the allocation of a sizable storage space. This process is akin to the storage proof outlined in the preceding section. However, a notable drawback is that the data utilized in creating the proof is static, potentially leading to the perception of wasted memory (Jerbi et al., 2021). In response to this challenge, the authors (Xu 2018) introduce Section-Blockchain, a pioneering blockchain protocol designed to address the issue of oversized storage without compromising the security of the blockchain. Notably, the Section-Blockchain network operates without distinct full or lightweight nodes; all nodes are considered equal contributors to the network. Experimental findings indicate that Section-Blockchain is efficient, significantly reduces storage requirements, and is resilient to catastrophic node losses. Furthermore, Section-Blockchain expands the capabilities of blockchain by establishing a self-sufficient, tamper-resistant decentralized storage system. This system enables automatic global distribution of data without relying on a centralized dispatcher for storage assignments. Nodes are incentivized to modify their local storage to receive higher compensation. The global storage distribution is continuously optimized to accommodate new nodes or compensate for lost ones. Despite its merits, a drawback of Section-Blockchain is the simplicity of authenticating the data used for storage (Jabbar et al., 2020).

The proposed architecture by the authors (Gang Wang, et al., 2019) introduces a hierarchical storage system for blockchain, wherein the majority of the blockchain is stored in cloud infrastructure, and the most recent blocks are stored in specific Industrial Internet of Things (IIoT) networks' overlay networks. This architecture establishes a hierarchical blockchain storage framework by seamlessly connecting local IIoT networks, the blockchain overlay network, and cloud infrastructure using two connectors: the blockchain connector and the cloud connector. The blockchain connector within the overlay network generates blockchain blocks from IIoT data, while the cloud connector resolves

synchronization issues between the overlay network and the clouds. A notable challenge in this architecture is the transaction load cost and the volume of data transferred in the network (Jamil et al., 2021 & Jerbi et al., 2022).

Another innovative approach suggested by the authors (Vijay & Kanade, 2021) involves a blockchain-based data storage paradigm where users contribute their electronic device storage space to meet growing data storage needs. The study explores a decentralized system architecture and proposes a fair compensation scheme for users providing storage space. Although this project doesn't directly implement a blockchain, it establishes a direct link to several blockchains. Additionally, the authors (Baza et al., 2019) present a solution utilizing smart contracts and blockchain to create a decentralized charging coordination system, eliminating the need for a single charge coordinator. Energy Storage Units (ESUs) leverage tokens for anonymous authentication on the blockchain. Each ESU submits a charging request to the smart contract address, including State-of-Charge (SoC), Time-to-Complete-Charge (TCC), and the required charging amount. The smart contract autonomously executes the charging coordination mechanism, prioritizing the highest-priority ESUs for charging within the current time window. Challenges in this scenario involve the energy consumption cost during data transmission and the network load (Aljumaili et al., 2023 & Jerbi et al., 2024 & Zaabar et al., 2021).

3. PROPOSED PROTOCOL

3.1 Network model

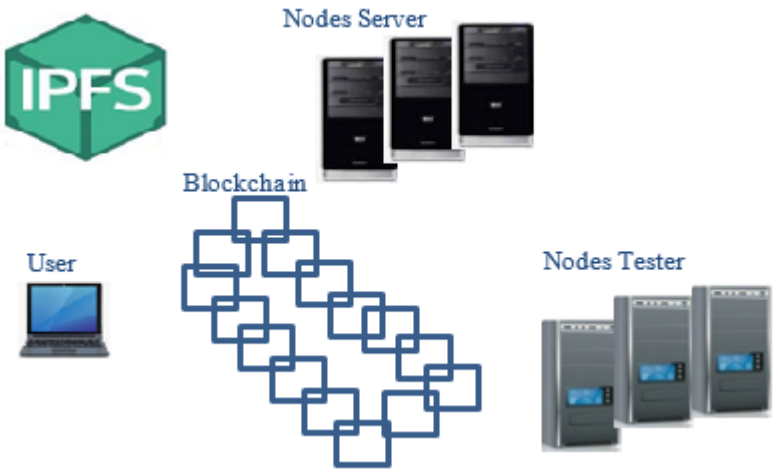
Each node actively participating in the blockchain network can assume one or more roles, as illustrated in Figure 1:

1. User Node: A user node refers to an individual seeking to store data on a server with available space, in return for a fee.
2. Blockchain: The blockchain represents a distributed and decentralized ledger, facilitating secure storage and exchange of information without reliance on a trusted third party. Transactions are aggregated into blocks, which are then cryptographically linked together, ensuring immutability. The ledger functions as a comprehensive record of all network participants' actions since its inception .
3. Server Node: This node is responsible for storing information, providing verifiable proofs, and returning the data at the conclusion of the contract. It is an integral component of the system, possessing storage capacity, and maintains a continuous connection to the network.
4. Tester Nodes: Tester nodes are volunteers selected randomly from a list, assigned the task of resolving potential contract disputes between users and servers. These individuals play a crucial role in ensuring the integrity of the system and are compensated for their contributions.
5. Data storage using Inter Planetary File System (IPFS): To assess file transmission or download within the proposed system, we gauge the time taken for a chain code to access data stored in OrbitDB using IPFS.

3.2 Fragmentation of a Block

The BlockStock protocol operates on the fundamental principle of storing coded fragments instead of entire blocks. This is achieved by breaking down initial blocks into fixed-size components, which are then encoded through linear fragment combinations. The specific linear combinations are determined by the chosen coding method. In this context, let's define i as a unique identifier for identifying a network node, and $N^{(i)}$ as the node in question. Additionally, $B^{(j)}$ represents the j^{th} block in the blockchain, starting with $B^{(1)}$. The maximum size of a block is denoted by S_B. To further elaborate, we introduce two integers, k and r. These values correspond to the number of fragments obtained by decomposing a block and the number of coded fragments generated and stored by a node, respectively.

Figure 1. Network model of system



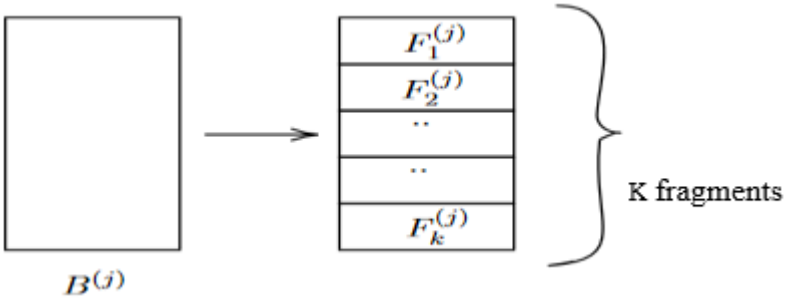
The initial step involves the fragmentation of blocks, with each block undergoing independent processing. It is assumed that the maximum size of a block, denoted as S_B, is predetermined. The block is then divided into k fragments, where k is a parameter shared across all network nodes. The u^{th} individual fragments of block $B^{(j)}$ are represented as $F^{(j)}u$. If the block's size is less than S_B, it is padded with zeros to attain the size S_B. This process ensures the generation of k fragments, each of identical and fixed size, denoted as $S_F = S_B/k$. Refer to Figure 2 for a visual representation of the partitioning of block $B^{(j)}$ into k fragments of equal size.

3.3 Protocol Proposed: BlockStock

An increasing number of users are opting to transfer their data to alternative servers, such as cloud-based platforms. However, the available options are often limited, and the prevailing system heavily relies on the trust users place in major corporations managing these servers. Recognizing this, various research and business organizations have been actively developing blockchain solutions for several years. These solutions aim to facilitate the exchange of free storage space among individuals. Upon thorough assessments, numerous recurring issues have been identified, particularly those associated with activities occurring outside the blockchain, making them unverifiable by all network nodes.

In response to these challenges, we present BlockStock, our innovative solution. BlockStock is a blockchain-based system designed to enable users to securely store their data or have it stored. Security

Figure 2. Fragmentation of a block



is ensured by requiring the storage server to regularly provide proof of storage to receive payment for its services. This proof is generated automatically through a challenge, offering a high level of certainty that the data is stored intact and unaltered. The payment process is automatic, contingent upon the server maintaining accurate information. This is achieved by leveraging evidence of recoverability.

In our network, every node meticulously verifies all contracts, transactions, and proofs, aligning with the principles of a typical blockchain. In the event of a dispute, the blockchain acts as an impartial adjudicator, determining whether the client has been wronged and whether compensation is warranted. The BlockStock protocol, outlined comprehensively in Figure 3, consists of three key phases: initialization, establishing a contract between a user and a storage server; audit and storage, conducting regular checks to ensure seamless operations; and contract termination, which involves final payment and file delivery to the user. BlockStock's approach integrates as many system management operations as possible into the blockchain, addressing the challenges associated with off-chain activities and enhancing transparency and trust in data storage transactions.

a) Initialization step

1. The user initiates a specialized transaction detailing storage requirements, encompassing data size, desired rental duration, proof of retrievability frequency, location price, and requisite proof-test parameters. This transaction, illustrated in step 1 of figure 4, is broadcasted over the network, encapsulated within a block accessible to all blockchain participants for examination and potential responses.
2. Upon fulfillment of the specified criteria by one or more nodes interested in the user's request, these nodes initiate a transaction to propose an agreement. In step 2 of figure 4, the server reserves a sum corresponding to the requested sequestrations in step 1.
3. In the subsequent step 3, the user selects one or more servers based on personal criteria, such as reputation. For each chosen server, the user conducts a transaction specifying the server identifier and locks the total payment for the rental and tester sequestration on a smart contract. Candidate servers can verify their selection status by inspecting these transactions.
4. During step 4, the user transfers the file to be stored outside the blockchain to the selected servers.
5. In step 5, the user generates and stores metadata on the blockchain, serving as proofs of retrievability. This metadata includes the file's fingerprint among other details.
6. In the final step 6, the server nodes generate and store their initial proofs of retrievability on the blockchain.

b) Storage phase and end of contract

1. *Enhanced Proof Transmission:* The server constructs and sends proofs using a retrievability mechanism. Employing the BlockStock protocol, the server receives a randomly generated challenge from the blockchain, reducing the need for additional data storage on nodes. This streamlined proof process results in space savings across the network, as depicted in Figure 5.
2. *Decentralized Verification Process:* Blockchain nodes independently verify proofs in each new block. At time 't,' nodes become aware of a random challenge, validate the correctness of the proof using metadata from phase 5 of the initialization step, and trigger an automatic payment from the client to the server. Smart contracts facilitate this process, ensuring not only proof accuracy but also the timely request based on predefined contract intervals.
3. *Contract Conclusion and File Retrieval:* As the contract nears completion, the client may need to download their file.

Figure 3. Diagram of the proposed protocol BlockStock

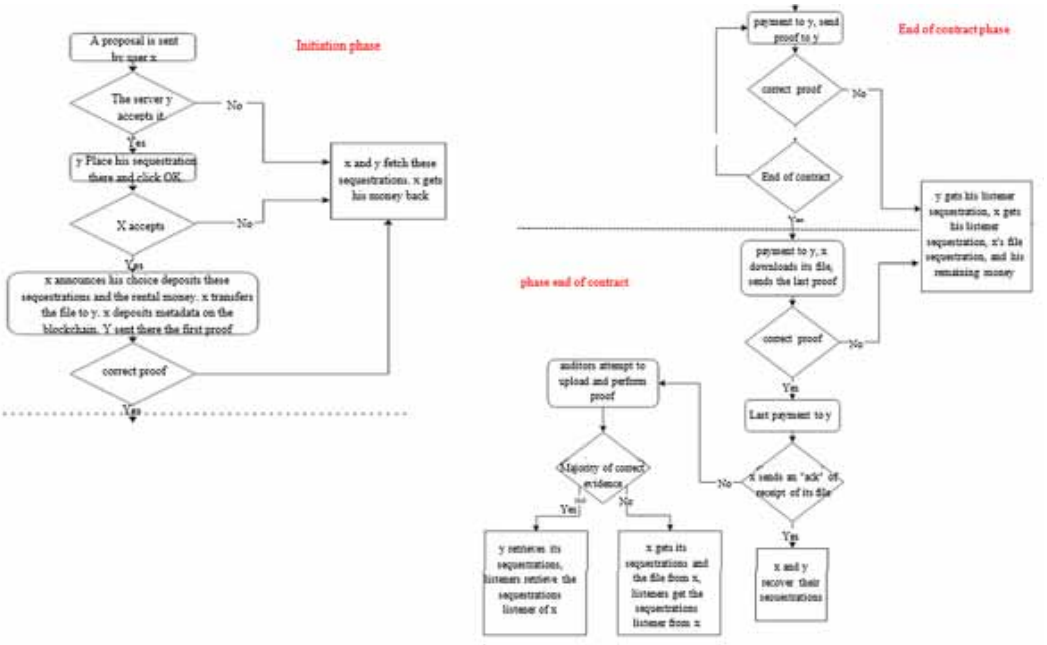


Figure 4. Initialization step for BlockStock

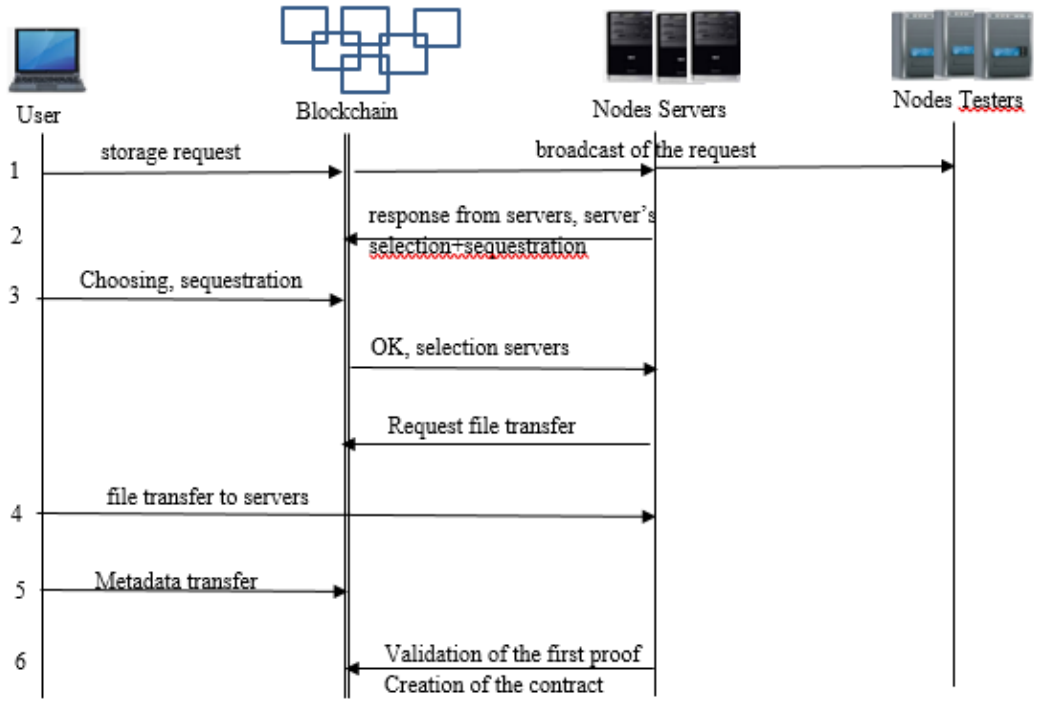
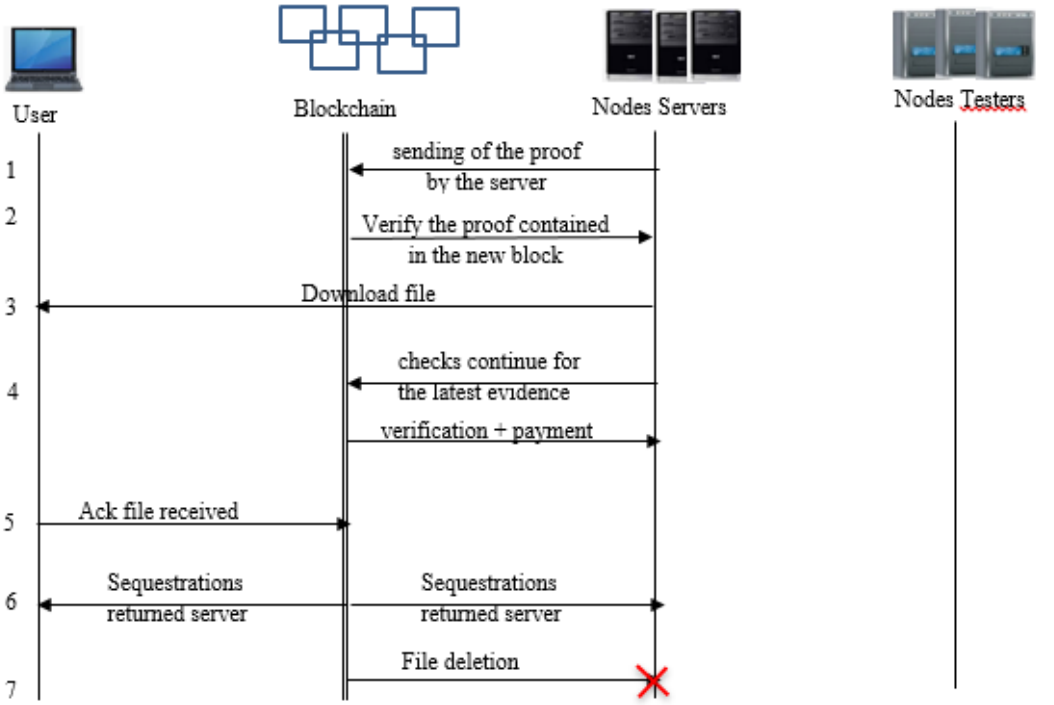


Figure 5. Storage phase and end of contract



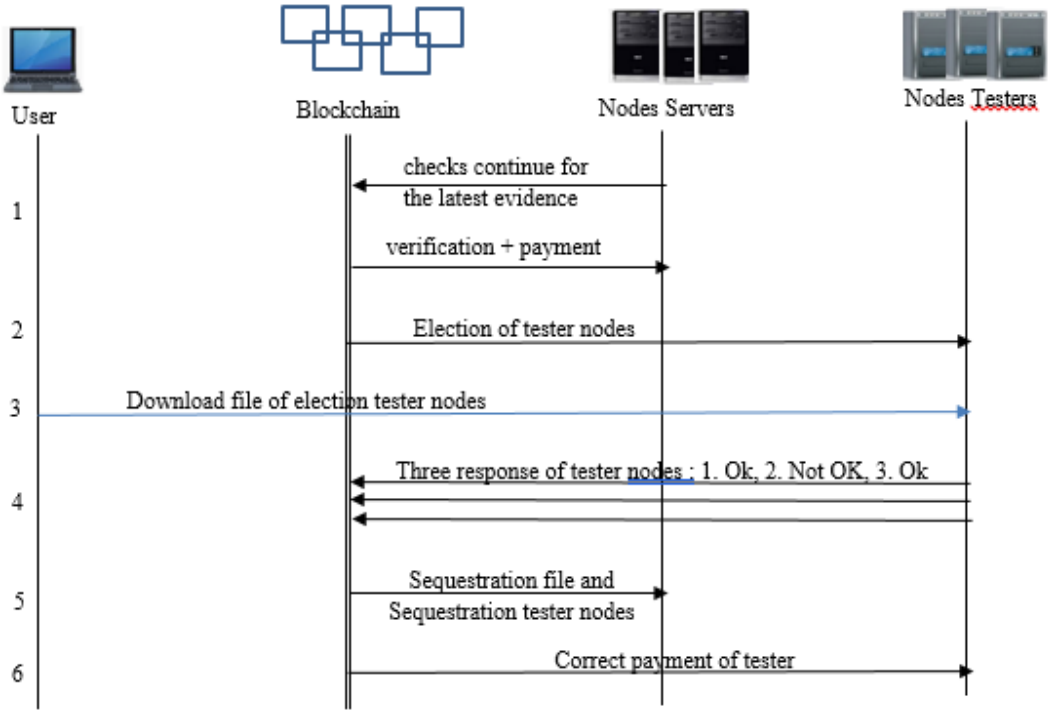
4. *Ongoing Verification Process:* Simultaneously, continuous checks are conducted on the latest evidence.
5. *File Confirmation Transaction:* Upon successful and complete file download, the client initiates a special transaction on the blockchain, informing other nodes of the file's retrieval.
6. *Full Sequestration Return:* With all processes executed correctly, sequestrations are entirely returned to the server nodes.
7. *Final File Deletion:* Ultimately (end of the contract), once the client confirms successful recovery, the server proceeds to delete the file securely, completing the final phase of the process.

3.4 Tester Nodes

The process of downloading a stored file occurs outside the blockchain. In the event of a dispute at the conclusion of a contract, identifying the party at fault and determining the recipients of various sequestrations becomes challenging for other network members. To address this, we introduce the role of tester nodes, tasked with resolving such conflicts. In the event of a disagreement, either the user or the server is inherently dishonest. To ascertain whether the client is providing accurate information (indicating the file's availability and correctness) or if the server is at fault (resulting in an unavailable or incorrect file), a designated set of nodes assumes the role of tester nodes.

Tester nodes are responsible for downloading the data and verifying its integrity using the fingerprint initially stored in the blockchain during phase 5 of the initialization step. Subsequently, they submit a response in the form of a transaction, disclosing the party at fault. This transaction signals either the unavailability or correctness of the file. In cases where the file is claimed to be available, tester nodes perform a proof of retrievability to substantiate this claim. Once a sufficient number of tester nodes have submitted their responses, a smart contract analyzes the results and determines

Figure 6. Checking transactions between users and servers



the course of action based on the majority. The role of tester nodes necessitates the allocation of resources to the system. To incentivize nodes to actively participate as tester nodes and act honestly, a remuneration system is implemented. Correct responses, in this context, are those aligning with the majority of nodes. This reward is integrated into the compensation received by the tester nodes identifying the erroneous entity.

In Figure 6, during the initial step 1, the server provides its latest proof, but the client fails to send an acknowledgment confirming the successful file download (occurs in step 5 during the storage phase and contract termination, labeled "Ack file received"). Subsequently, in step 2, the network selects several testing nodes tasked with attempting to download the file to verify its integrity, leading to the subsequent step 3. These nodes then provide responses, with the outcome varying based on the circumstances.

Figure 6 illustrates a scenario where the server is deemed honest and possesses the complete file. Once all the testing nodes have been chosen and have verified the file's availability, they collectively communicate their findings in a transaction. In this specific instance, during step 4, the majority of testing nodes, namely 1 and 3, confirm that the file is consistently accessible and correct. Consequently, it is determined that the server is honest, whereas the client is considered dishonest. To conclude the process, in step 5, the server recovers its two sequestrations, and testing nodes 1 and 3, which provided valid responses, are duly rewarded with the sequestrations initially held by the user's testing node.

4. PERFORMANCE AND EVALUATION

We conduct a comprehensive set of tests in this section to ensure the optimal performance of the proposed BlockStock system concerning block size, transaction speed, data volume, and storage duration. Leveraging Hyperledger Caliper (Hyperledger-Fabric, 2021), an open-source benchmarking

tool maintained by the Linux Foundation, we enable users to analyze the performance of blockchain applications. Our experiment analysis incorporates specific network factors, including users, server nodes, and tester nodes. To assess file transmission or download within the proposed system, we gauge the time taken for a chain code to access data stored in OrbitDB using IPFS (Inter Planetary File System). Throughput is quantified by the number of verified transactions per second. Conducting 100 iterations, we scrutinize memory consumption (average, maximum) and central processing unit (CPU) usage (average, max). The average and highest CPU utilization for the peer are 6.54 percent and 17.09 percent, respectively. Similarly, the peer's highest memory usage is 806.7 MB (minimum) and 798.5 MB (average). Our findings underscore that the suggested permissioned blockchains outperform across all performance and user experience metrics.

To assess the likelihood of a majority of honest nodes among tester nodes and identify potentially deceptive nodes, we denote 'n' as the number of nodes designated as tester nodes. Let 'p' be the probability that a node responds correctly, indicating honesty, and 'Xn' represent the total number of correct answers. Assuming 'Xn' follows a binomial probability distribution with probability 'p', we treat it as realizations of independent discrete random variables. Recognizing the need for a minimum number of nodes in the network for proper blockchain functioning and valid consensus, we assume that this quantity is adequate to approximate the binomial distribution using a normal distribution.

$$f\left(x\right)=\frac{e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^{2}}}{\sigma\sqrt{2\pi}}=\frac{e^{-\frac{1}{2}\left(\frac{x-np}{\sqrt{np\left(1-p\right)}}\right)^{2}}}{\sqrt{np\left(1-p\right)}\sqrt{2\pi}} \tag{1}$$

The probability of having a dishonest majority is as follows:

$$\overline{P_{m}}=p\left(0\le X_{n}\le\frac{n}{2}\right)=\int_{0}^{\frac{n}{2}}\int\left(x\right)dx=\int_{0}^{\frac{n}{2}}\frac{e^{-\frac{1}{2}\left(\frac{x\quad np}{\sqrt{np\left(1-p\right)}}\right)^{2}}}{\sqrt{np\left(1-p\right)}\sqrt{2\pi}} \tag{2}$$

In Figure 7, we represent the probability of having the maximum percentage of tester nodes trust according to the number of tester nodes. The percentage is greater than 98% with 100 test nodes for a probability p = 6/7 and p = 5/6 and greater than 96% for a probability p = 4/5. with 200 test nodes the percentage is greater than 97% for a probability p = 6/7 and p = 5/6 and greater than 95% for a probability p = 4/5. but from 300 and 400 nodes the percentage of tester nodes trust remains stable, the percentage is greater than 96% for a probability p = 6/7 and p = 5/6 and greater than 94% for a probability p = 4/ 5. It is necessary to have a large number of trusted test nodes to intervene in the event of a problem between the actors.

In Figure 8, we represent the probability of having the maximum of server's trust percentage according to the number of servers. The percentage is greater than 99% with 100 servers for a probability p = 6/7, p = 5/6 the percentage is greater than 98% and for a probability p = 4/5 the percentage is greater than 97% with 400 servers. The percentage is greater than 98% with 400 servers for a probability p = 6/7, p = 5/6 the percentage is greater than 97% and for a probability p = 4/5 the percentage is greater than 96% with 400 servers. Most servers are trusted by blockchain. According to this analysis, increasing the number of servers leads to a decrease in the maximum server trust

Figure 7. Percentage of tester nodes trust

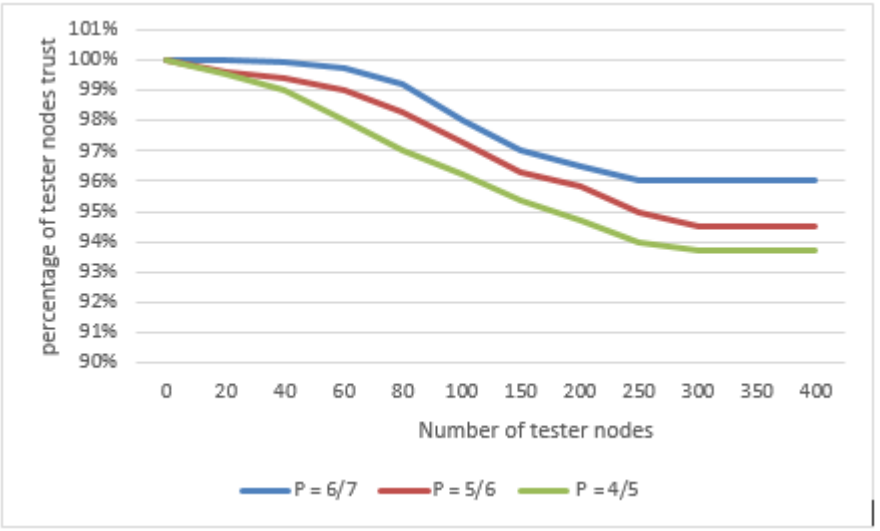
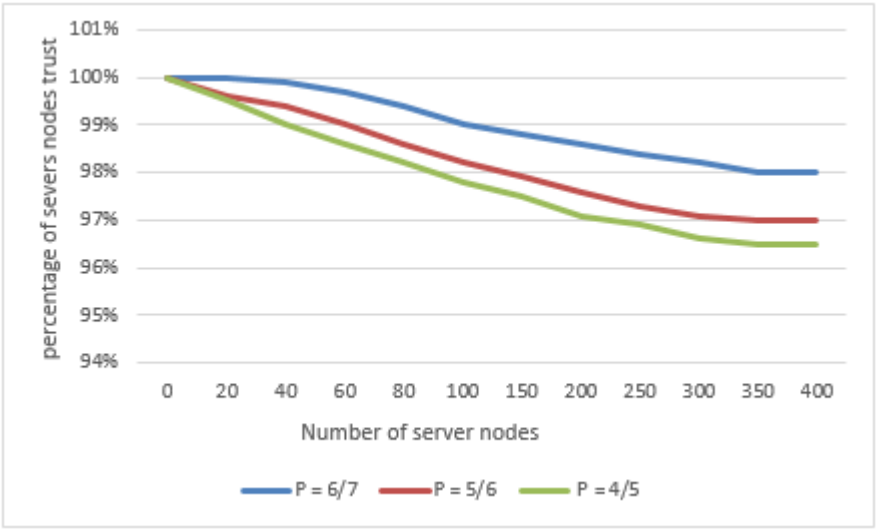


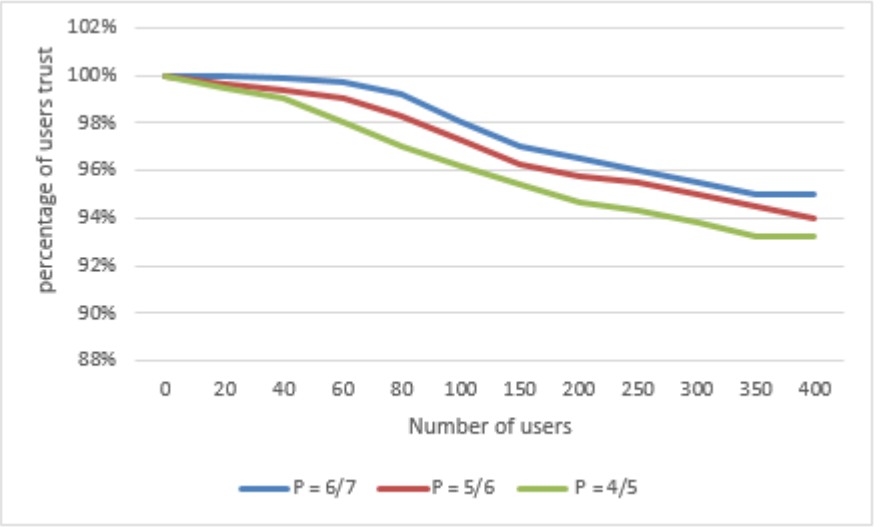
Figure 8. Percentage of server nodes trust



percentage by 1%. This suggests that increasing the number of servers can improve the overall server trust percentage.

In Figure 9, we represent the probability of having the maximum of users trust percentage according to the number of users. the percentage is greater than 99% with 100 servers for a probability p = 6/7, p = 5/6 the percentage is greater than 98% and for a probability p = 4/5 the percentage is greater than 97%. With 400 servers, the percentage is greater than 95% for a probability p = 6/7, p = 5/6 the percentage is greater than 94% and for a probability p = 4/5 the percentage is greater than 93%. This drop is due to new users. the latter must confirm their trust to be registered in the blockchain.

Figure 9. Percentage of user trust



Throughput is subdivided into two steps: Transaction throughput and read throughput. The transaction throughput refers to the number of successful transactions completed on the blockchain network in a certain time period. The read throughput, on the other hand, is defined as the total number of reading operations done over the blockchain network in a particular time frame. As shown in figure 10, the read throughput of the developed system is evaluated by varying the send rate from 500 to 3000 transactions per send, with an arbitrary configuration of machine utilization. The graph shows that under the ideal case, read transaction throughput grows dramatically when it reaches the peak, then decreases somewhat after transmit rate 2500, which is considered the optimal scenario.

Figure 10. Read transaction throughput for blockStock

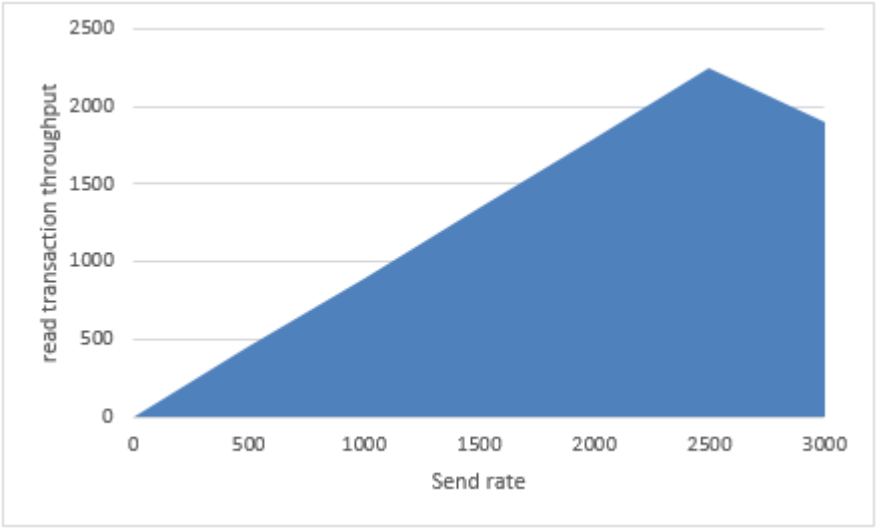
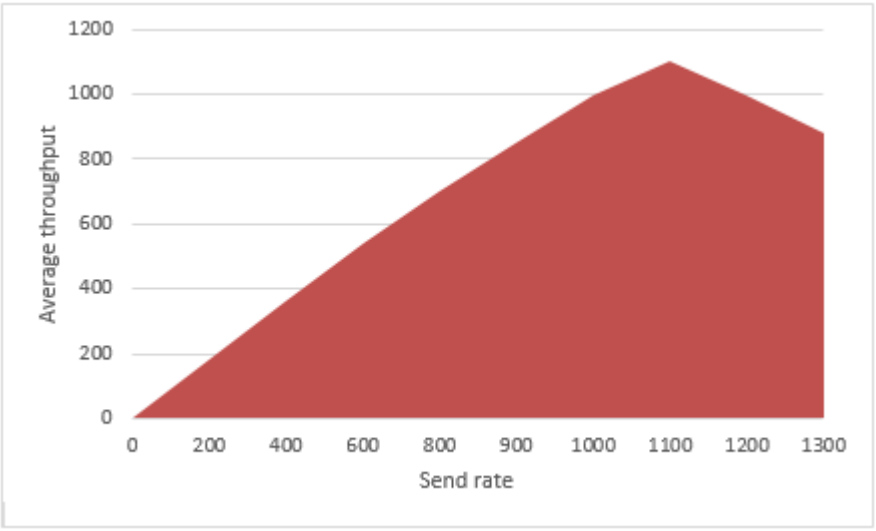


Figure 11. Transaction throughput of BlockStock



As illustrated in Figure 11, the transaction throughput is measured by adjusting the transmit rate from 200 to 1400 transactions per second.

According to Figure 8 and 9, the transactions are carried out in an optimal way to lighten the load occupied by the network. likewise, the actor nodes (users, test nodes and servers) in our architecture make it possible to offer the data stored in the blockchain in complete confidence to all requesters.

The graph shows that as the send rate increases, the average transaction throughput grows until the send rate hits a high of 1200 transactions per send. Furthermore, when the average transaction throughput is 1200 transactions per second, the throughput begins to decline as the send rate increases.

Figure 12 presents the number of tests performed for protocol BlockStock according to of time in ms. The time required to execute a transaction: the average time for a block mining is around 20 ms. This time represents the duration of the exchange of several messages relative to the number of the transaction between different actors. Plus, the duration of validation of the smart contract transaction by the blockchain and the file download.

In figure 13 depicts the overall energy spent by the actors during its transaction period, which is the sum of the energy consumed during the communication. The energy consumption of the actors is average 0.9 mj. This consumption is very reduced, allows an extension of the life of the wireless devices of the actors.

5. CONCLUSION AND FUTURE WORK

The advantage of BlockStock is that it can fit into existing blockchain frameworks without the need to modify their architecture. It permits reducing the storage space. With our solution BlockStock, we can expect a significant increase in the number of participating nodes, which would lead to network decongestion by offering more nodes from which to download the channel.

Moreover, we propose to store the blockchain in an encrypted form. Each block is divided into fragments, of fixed sizes, linear combinations are performed of these fragments in order to create encoded fragments. Only these coded fragments are kept, as well as the block header. The original block is deleted. To have access to the initial data, a node must download enough coded fragments to perform a decode operation and reconstruct the block. It can be checked thanks to its fingerprint,

Figure 12. Time transaction in protocol BlockStock

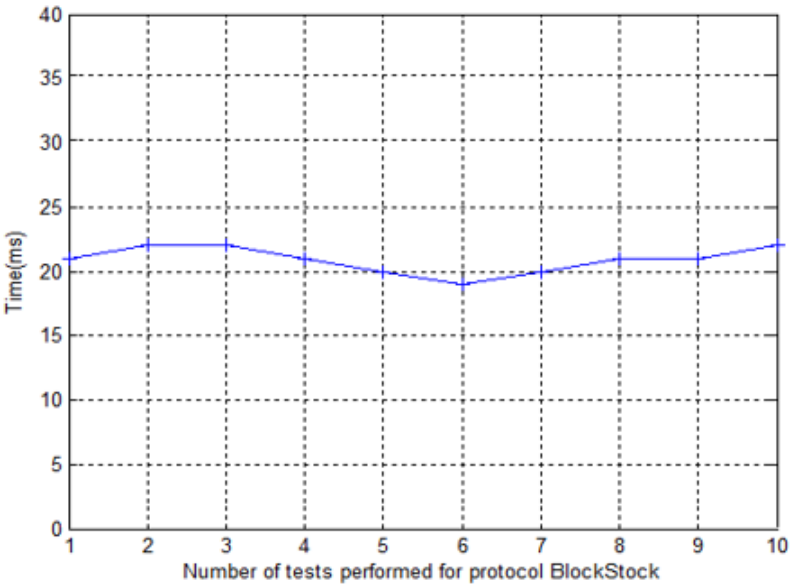
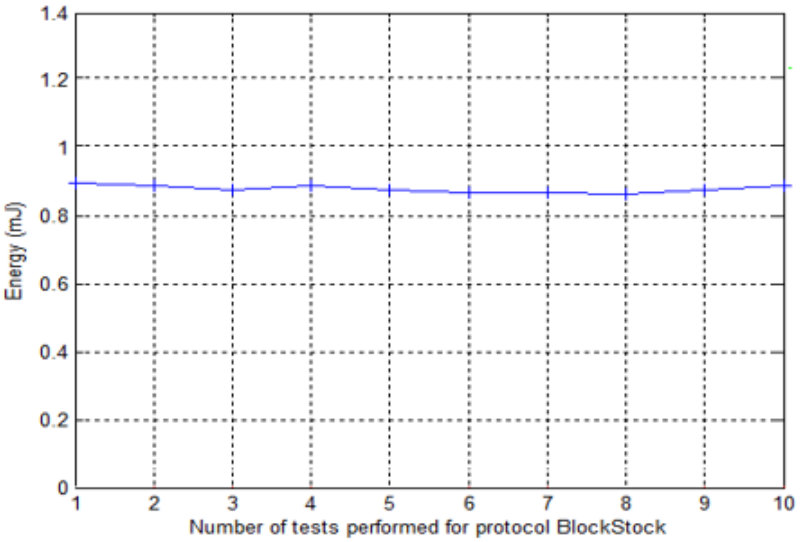


Figure 13. Energy consumption in protocol BlockStock



stored in the header. Blockchain ensures the confidentiality, integrity, traceability and availability of data. However, storing large data by the BlockStock protocol avoids the scalability problem. As a future work we are planning to use the proposed solution in IoT logistics to strengthen IoT logistics systems' security.

CONFLICTS OF INTEREST

We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

FUNDING STATEMENT

No funding was received for this work.

Correspondence should be addressed to Wassim Jerbi; wassim.jerbi@isetn.rnu.tn

REFERENCES

Aljumaili, A., Trabelsi, H., & Jerbi, W. (2023) A Review on Secure Authentication Protocols in IOV: Algorithms, Protocols, and Comparisons. *7th International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT),* (pp. 1–11). IEEE. doi:10.1109/ISMSIT58785.2023.10304917

Baza, M., Muhammad, M. I., Mahmoud, M., Serpedin, E., & Ashiqur, M. R. (2019). *Blockchain-Based Charging Coordination Mechanism for Smart Grid Energy Storage Units*. *IEEE International Conference on Blockchain (Blockchain)*, Atlanta. GA. USA. doi:10.1109/Blockchain.2019.00076

Buterin, V. (2013). *Ethereum: A next-generation smart contract and decentralized application platform*. GitHub. https://github.com/ethereum/wiki/wiki/White-Paper.

Hyperledger-Fabric. (2021). *A blockchain platform for the enterprise*. Hyperledger Fabric. https://hyperledgerfabric.readthedocs.io/en/release-2.2/,

Jabbar, R., Fetais, N., Krichen, M., & Barkaoui, K. (2020). Blockchain technology for health- care: Enhancing shared electronic health record interoperability and integrity. *IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT),* IEEE.

Jamil, F., Kahng, H. K., Kim, S., & Kim, D. H. (2020). Towards secure fitness framework based on IoT-enabled blockchain network integrated with machine learning algorithms. *Sensors (Basel)*, *21*(5), 1640. doi:10.3390/s21051640 PMID:33652773

Jerbi, W., Cheikhrouhou, O., Guermazi, A., & Trabelsi. (2023). MSU-TSCH: A Mobile scheduling updated algorithm for TSCH in the internet of things. *IEEE Transactions on Industrial Informatics*. IEEE. .10.1109/TII.2022.3215990

Jerbi, W., Cheikhrouhou, O., Guermazi, A., Baz, M., & Trabelsi, H. (2021) BSI: Blockchain to secure routing protocol in Internet of Things. Concurrency and Computation, e6794. doi:10.1002/cpe.6794 doi:10.1002/cpe.6794

Jerbi, W., Cheikhrouhou, O., Guermazi, A., Boubaker, A., & Trabelsi, H. (2021). *A Novel Blockchain Secure to Routing Protocol in WSN*. *International Conference on High Performance Switching and Routing (HPSR)*, Paris, France. doi:10.1109/HPSR52026.2021.9481805

Jerbi, W., Cheikhrouhou, O., Guermazi, A., Hamam, H., & Trabelsi, H. (2021). A Blockchain based Authentication Scheme for Mobile Data Collector in IoT. 17th International Wireless Communications & Mobile Computing Conference. IWCMC. doi:10.1109/IWCMC51323.2021.9498656

Jerbi, W. Cheikhrouhou, O., Guermazi, A., & Trabelsi., H. (2024). An enhanced MSU-TSCH scheduling algorithms for industrial wireless sensor networks. *Concurrency Computat Pract Expert*. Wiley. .10.1002/cpe.7938

Jerbi, W., Cheikhrouhou, O., Hamam, H., Trabelsi, H., & Guermazi, A. (2022). A blockchain-based storage intelligent. *International Wireless Communications and Mobile Computing Conference (IWCMC)*, Dubrovnik, Croatia. doi:10.1109/IWCMC55113.2022.9824790

Jerbi, W., Guermazi, A., & Trabelsi, H. (2020). A secure routing protocol in heterogeneous networks for internet of things. *International Wireless Communications and Mobile Computing Conference (IWCMC)*, Limassol, Cyprus. doi:10.1109/IWCMC48107.2020.9148502

Khallel, A., Iqbal, J., Hussain, S., Ullah, S. & Nayab. (2023). A Comprehensive Survey on Blockchain-Based Decentralized Storage Networks. *IEEE Access*. IEEE. .10.1109/ACCESS.2023.3240237

Park, S., Kwon, A., Fuchsbauer, G., Gaži, P., Alwen, J., & Pietrzak, K. (2018). Spacemint: A cryptocurrency based on proofs of space. *International Conference on Financial Cryptography and Data Security*. Springer. doi:10.1007/978-3-662-58387-6_26

Pourmajidi, W., Zhang, L., Steinbacher, J., Erwin, T., & Miranskyy, A. (2023). Immutable Log Storage as a Service on Private and Public Blockchains. *IEEE Transactions on Services Computing*, *16*(1), 356–369. doi:10.1109/TSC.2021.3120690

Ruj, S., Rahman, M., Basu, A., & Kiyomoto, S. Blockstore (2018): A secure decentralized storage framework on blockchain. *International Conference on Advanced Information Networking and Applications (AINA),* (pp. 1096–1103). IEEE. doi:10.1109/AINA.2018.00157

Vijay & Kanade, A. (2021). A Blockchain-Based Distributed Storage Network to Manage Growing Data Storage Needs. *3rd International Conference on Signal Processing and Communication (ICPSC).* IEEE.

Vorick, D. & Champine, L. (2014). *Sia: Simple decentralized storage*. Nebulous Inc.

Wang, G., Shi, Z., Nixon, M., & Han, S. (2019). ChainSplitter: Towards Blockchain-Based Industrial IoT Architecture for Supporting Hierarchical Storage. *IEEE International Conference on Blockchain (Blockchain)*, (pp. 166-175). IEEE. doi:10.1109/Blockchain.2019.00030

Wang, Y., Liao, J., Yang, J., Li, Z., Ma, C., & Mao, R. (2023). *Meta-Block: Exploiting Cross-Layer and Direct Storage Access for Decentralized Blockchain Storage Systems*. IEEE Transactions on Computers. doi:10.1109/TC.2022.3226305

Wilkinson, S., Boshevski, T., Brandoff, J., & Buterin, V. (2014). *Storj a peer-to-peer cloud storage network*.

Wood, G. (2014). Ethereum*: A secure decentralised generalized transaction ledger.* Ethereum. https://ethereum.github.io/ yellowpaper/paper.pdf

Yibin, X. (2018). Section-Blockchain: A Storage Reduced Blockchain Protocol, the Foundation of an Autotrophic Decentralized Storage Architecture. *International Conference on Engineering of Complex Computer Systems (ICECCS).* (pp 115-125). IEEE. doi:10.1109/ICECCS2018.2018.00020

Zaabar, B., Cheikhrouhou, O., Jamil, F., Ammi, M., & Abid, M. (2021). HealthBlock: A secure blockchain-based healthcare data management system. Computer Networks. https://doi.org/10.1016/j.comnet.2021.108500

Zhang, C., Xuan, H., Wu, T., Liu, X., Yang, G., & Zhu, L. (2024). *Blockchain-Based Dynamic Time-Encapsulated Data Auditing for Outsourcing Storage*. IEEE Transactions on Information Forensics and Security. IEEE. doi:10.1109/TIFS.2023.3338485

Zhang, P., Liu, Z., Zhou, M. & Huang. (2023). A Group-Based Block Storage Model With Block Splitting and Unit Encoding for Consortium Blockchains. *IEEE Transactions on Network and Service Management*, (pp 2886 – 2897). IEEE. .10.1109/TNSM.2023.3237847

Wassim Jerbi is currently a Teacher at the Higher Institute of Technological Studies of Sfax-Tunisia (ISET Sfax). He obtained the master's degree in Computer Systems Engineering from the National School of Engineers of Sfax in 2010. Since 2010, He has been becoming an active member of the Computer and Embedded System Laboratory at the National School of Engineers of Sfax-Tunisia (ENIS). He received the PhD degree in Computer Systems Engineering from the National School of Engineers of Sfax in 2017. He contributed in the organization of several workshops and conferences. His research and teaching interests focus on Routing and Security in Wireless Sensor Networks, Blockchain, simulation and performance evaluation. He has several publications in refereed international conferences and peer reviewed journals.

Omar Cheikhrouhou is currently an Assistant Professor at Higher Institute of Computer Science of Mahdia, University of Monastir, Tunisia. He was an Assistant Professor at College of Computer and Information Technology, Taif, KSA. He is also a member of CES Lab (Computer and Embedded System), University of Sfax, National School of Engineers. Dr. Omar Cheikhrouhou has received his B.S., M.S., and Ph.D. degrees in Computer Science from the National School of Engineers of Sfax in March 2012. His Ph.D. deals with security in Wireless Sensor Networks and more precisely in "Secure Group Communication in Wireless Sensor Net- works". Currently, his research interests span over several areas related to Wireless Sensor Networks, CyberSecurity, Edge Computing, Blockchain, Multi-Robot System Coordi- nation, etc. Dr. Omar has several publications in several high-quality international journals and conferences. He has received some awards, including the "Governor Prize" from the Governor of Sfax in 2000.

Abderrahmen Guermazi phd since 2017 in National Engineering School of Sfax, CES LAB, Tunisia. is currently a Technologist Professor in computer science at the Higher Institute of Technological Studies of Sfax –Tunisia. He is preparing his PhD in computer systems engineering at the National School of Engineers of Sfax – Tunisia where he is member of Computer and Embedded System Laboratory. He received the National Aggregation degree in computer science at 1998. His research and teaching interests focus on Wireless Sensor Networks, Routing and Security. He has several publications in international conferences of high quality.

Hafedh Trabelsi received the B.S. degree from National School of Engineers of Sfax (ENIS), university of Sfax, Tunisia, in 1989, the M.S. degree in the Central School of Lyon, France, in 1990, the Ph.D degree from the University of Paris XI Orsay, France, in 1994 and the habilitation degree from the University of Sfax, in 2008, all in Electrical Engineering, . He is currently a full professor of National School of Engineers of Sfax (ENIS). He is a member of the research laboratory computer Embedded system (CES) dealing will smart system in device.