Journal of Theoretical and Applied Information Technology

31st October 2023. Vol.101. No 20
© 2023 Little Lion Scientific

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

ENHANCING TRANSACTION VERIFICATION THROUGH PRUNED MERKLE TREE IN BLOCKCHAIN

M.GRACY[1], B. REBECCA JEYAVADHANAM[2] , S. ALBERT ANTONY RAJ[3]

[1]Department of Computer Applications, College of Science and Humanities, SRM Institute of Science and Technology, Kattankulathur, India

[2]Department of Computer Science, York St John University London, United Kingdom

[3]Department of Computer Applications, College of Science and Humanities, SRM Institute of Science and Technology, Kattankulathur, India

E-mail: [1]gm3064@srmist.edu.in, [2]r.balasundaram@yorksj.ac.uk , [3]alberts@srmist.edu.in

ABSTRACT

A Merkle tree is a data structure employed within Blockchain technology to securely verify information or transactions within a vast data collection. This paper proposes a new and improved verification method, Pruned Merkle Tree (PMT), for hash nodes marching to the Merkle Root in a Minimal duration. PMT is a unique mechanism for verifying unpaired transactions in a block. The future influence of cryptocurrency will be immense, and PMT showcases its effectiveness in terms of transaction speed and node repetition. Our method allows any block to validate the full availability of transactions without repeating hash nodes and focuses on improving the transaction process through the Pruned Merkle Tree and achieving remarkable results. To assess the performance of the proposed system, we used Hyperledger Caliper, a benchmarking tool specifically designed for measuring the performance of Hyperledger-based blockchain solutions. The evaluation results show a significant improvement in throughput, with a value of 30450kbps recorded. The processing time has also increased noticeably, reaching 1660ms. Security measures have also been strengthened, yielding an impressive 99.60%. The energy consumption factor plays a crucial role, and the PMT exhibits the lowest value at 235 joules.

Keywords: *Blockchain, Merkle Tree, Pruned Merkle Tree, Security, Transaction Verification*

1. INTRODUCTION

According to the advancement of information technology, the Internet has become an integral part of data transmission in our daily lives. Blockchain technology was first presented as a public, decentralized, and trust-free digital currency ledger [1], and it has since gained widespread acceptance in various fields. Blockchain [2] (for example, Bitcoin and Ethereum) keep a chronological record of transactions organized into a chain of blocks. Starting with the Genesis block, network nodes add to the ledger by creating and adding new blocks. The transactions in the received blocks are validated by full nodes that download the entire block tree. On the other hand, a blockchain must include light nodes [3], which may be interested in confirming a few specific transactions to improve scalability.

Pairing nodes in a Merkle tree can improve the verification process's efficiency and the data's security[4]. The choice of pairing scheme in a Merkle tree depends on the specific use case and the requirements for efficiency and security. While different pairing schemes can affect the performance and security of the Merkle tree, they all serve the same basic purpose of ensuring the integrity and authenticity of the data stored in the blockchain. The potential drawback is that Merkle trees are vulnerable to attacks if the hash function used to generate the tree is compromised. Suppose an attacker can generate hash collisions (i.e., two pieces of data producing the same hash value). In that case, they may be able to construct a fraudulent Merkle tree that appears valid but contains incorrect data. When a block has transactions with a node unpaired, the Merkle tree may need to be balanced, which may cause a problem during verification. In a balanced Merkle tree, each level has nodes paired legibly, except the leaf nodes at the bottom of the tree. This enables efficient verification by requiring only a logarithmic amount of nodes in the tree to be accessed. If a block contains transactions with a node unpaired, the Merkle tree will be unbalanced, requiring the last transaction to be duplicated to construct a complete tree. This duplication may result in inefficiencies during verification because traversing the unbalanced branches of the tree may necessitate more computational resources. To address this shortcoming, we proposed Pruned Merkle Tree to make the verification process faster and easier with no node repetition. This conception simplifies transaction verification, especially when a block contains transactions with a node unpaired in the tree formation. Our proposed idea can lead to more efficient verification by avoiding node repetition and the construction of an unbalanced Merkle tree. An attacker could, for example, create a valid hash for a modified transaction block that contains a different transaction at the location of the duplicated transaction. The attacker may be able to insert fraudulent transactions into the block without being detected by the Merkle tree verification process [5] [6]. The proposed work makes several key contributions:

1. Introducing Pruned Merkle Tree: Unlike traditional Merkle tree implementations that duplicate an odd leaf to create a "dummy leaf," our proposed method, PMT, presents a novel form of the Merkle tree. It specifically addresses the challenge of blocks accumulating transactions with a node unpaired, offering an alternative approach to ensure the integrity and structure of the tree.
2. Consistent Throughput Time: PMT has been extensively evaluated and demonstrated consistent throughput time. The proposed method optimizes the verification and computation processes, resulting in an efficient and predictable performance for transaction processing.
3. Efficiency Improvements: PMT offers notable advantages over existing methods regarding energy consumption, storage requirements, and authentication time. The proposed approach reduces resource demands by leveraging the Pruned Merkle Tree formation, making it more resource-efficient than alternative techniques.
4. Enhanced Data Security and Processing Time: PMT has exhibited promising results in data security and processing time. The novel formation of the Merkle tree maintains a high level of data integrity, and the optimized verification process contributes to faster and more reliable transaction processing.

These contributions collectively highlight the significance of the proposed PMT method in improving efficiency, resource utilization, data security, and processing time within blockchain systems.

The paper is divided into sections to allow for a thorough examination of the subject. Section 2 provides an overview of blockchain technology and the basic concepts of Merkle trees. Section 3 explores related work. Section 4 describes the proposed method of PMT. Section 5 contains a security analysis. Section 6 presents results and performance metrics to validate its efficacy. Finally, Section 7 concludes the paper by highlighting potential future research and development directions.

2. UNDERSTANDING BLOCKCHAIN TECHNOLOGY AND THE MERKLE TREE

This section will present a comprehensive introduction to blockchain technology, specifically emphasizing the Merkle Tree, a fundamental element of this innovative technology.

2.1 Blockchain

Blockchain is a decentralized and distributed digital ledger that enables multiple parties to securely and transparently maintain a shared record of transactions or information [7] [8]. A blockchain comprises a series of blocks containing a collection of transactions or data. [9] The typical blockchain is shown in Figure 1. Each block typically references the previous block, creating a chain-like structure. The components of the blockchain are as follows:

1. Block Header: The block header contains data about the block, such as the block's unique identifier (hash), the time the association was formed, and a reference to the previous block's hash.
2. Transactions: Transactions are records of activities or data stored in a block. They can represent various data types, including cryptocurrency payment transfers, smart contract interactions, and other relevant data stored on the blockchain.
3. Nonce: The nonce (once-used number) is a random or incremental value added to the block header during mining. Miners change the nonce repeatedly until they find a hash that meets the consensus mechanism's predetermined criteria, such as having a certain number of leading zeros.
4. Hash: The hash is a unique digital fingerprint generated from the data within a block. It ensures

its integrity and references subsequent blocks, establishing an immutable and secure information chain.

5. Previous Hash: The previous hash represents the block's hash before it is in the blockchain. It establishes the chronological order and creates a chain-like structure. Including the previous hash in a block's header connects each block to the previous block, forming a continuous sequence. [10].
6. Merkle root: The Merkle root in a blockchain is a single hash that represents the entire set of transactions within a block, providing a concise and efficient way to verify the integrity of the data without revealing the individual transactions.

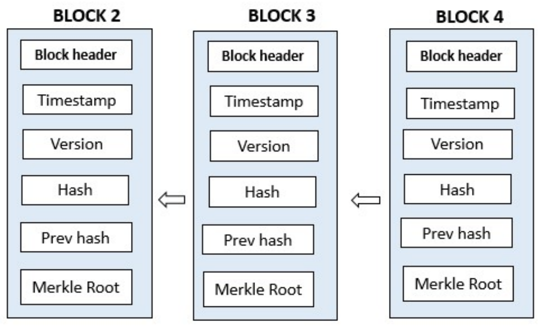

Figure 1. Blockchain structure

2.2 Merkle Tree

In a traditional Merkle tree, the data is organized into a binary tree structure, where each leaf node represents a data chunk or a transaction, and each non-leaf node represents the hash value of its child nodes. The tree is built by recursively hashing pairs of child nodes until a single root hash value is obtained.

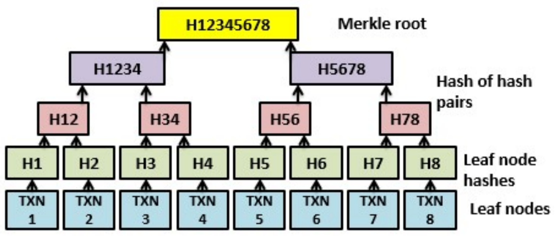

Figure 2. Existing Merkle Tree structure

Merkle trees are information structures that are used to securely verify data in a large content pool [11] [12]. Figure 2 illustrates the traditional Merkle tree construction. The construction of the Merkle root with numerous transactions inside block 3 is shown in Figure 3.

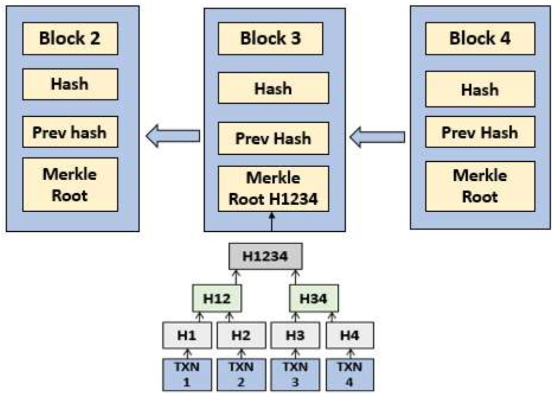

Figure 3. Formation of Merle root

Figure 4 shows transactions with five nodes and the Merkle root is formed from the five nodes. Nodes with green color represent the verification of the transaction T3. The hash nodes involved in verifying T3 = H4, H7, H12.

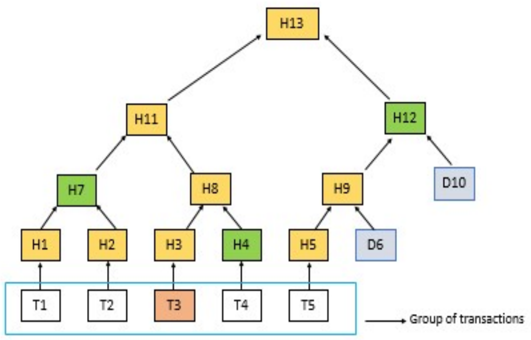

Figure 4. Transaction verification in Merkle tree

3. RELATED WORK

This section provides an in-depth review of various papers addressing different aspects of Merkle Trees and their applications in blockchain transactions. The surveyed literature encompasses innovative techniques for data authentication, consistency verification, synchronization, operational online authentication, classification, traversal algorithms, outsourcing data security, efficient payments, energy-saving calculations, bandwidth reduction, and web document content justification.

Usharani et al. [13] introduced a data structure called the Modified Merkle Tree, which facilitates data authentication, consistency verification, and data synchronization. Dong et al. [14] proposed a novel approach called Merkle tree-based operational online authentication, which involves incorporating random inputs between two distinct entities: the prover and the approver. A technique known as the Merkle Quad-Tree [15] has shown consistent advantages with various modifications and offers a computational time reduction of approximately 1% compared to traditional Merkle Trees. Yuji et al. [16] presented an authentication method inspired by Lamport's approach, where authentication is performed by revealing hash values of nodes greater in the chain than the leaf node's hash. Adhikari et al. [17] focused on utilizing Merkle hash trees to ensure the integrity of large and dynamic spatial data. The researchers conducted a comparative analysis between Merkle Tree data structure and conventional data structures to assess their effectiveness in storing and retrieving geospatial data. In the context of Merkle multi-proof, Ramabaja et al. [18] have introduced a standard sparse Merkle multi-proof that necessitates index storage for every non-leaf hash present in the multi-proof. Ripon et al. [19] introduced a substitute model called Hex-Bloom, aimed at replacing Merkle Trees to mitigate network intervals. Kan et al. [20] presented MTFS, a solution for private file storage in blockchain. Markus et al. [21] proposed an efficient algorithm for traversing Merkle Trees and a technique for classifying leaves and their associated authentication paths. Dong et al. [22] introduced a novel technique for ensuring the security of outsourced data using Merkle tree authentication. Wang et al. [23] proposed a modified Merkle tree structure to facilitate efficient payments in blockchain-based Industrial Internet of Things (IIoT) systems. Castellon et al. [24] proposed an energy-saving algorithm for Merkle Tree root calculations, a crucial component of blockchain models, which maintains the preferred level of security while sacrificing certain system availability. Kuszmaul [25] proposed a novel data structure called Verkle Trees, which reduces the bandwidth requirements of Merkle Trees. It is done at the cost of computational power. Francesco Bruschi et al. [26] presented a system based on a Merkle tree demonstration, which generates a concise explanation of web document content. Roberto et al. [27] proposed Merkle Trees to enable authentication with just a single cryptographic hash per storage system. Ecsobar et al. proposed energy reducing algorithmic technique and conducted experiments involving both the standard and energy-efficient implementations across various input sizes. Their findings indicate that substantial reductions in energy consumption are achievable [28]

In our investigation, we explored five existing methods that use Merkle trees in various dimensions to deal with transactions, data, and image authentication and acknowledged their contributions. We must emphasize that we intend to uphold the significance of the papers we have compared against. On the contrary, we have thoroughly analyzed their key points and findings, leveraging them as foundational knowledge to strengthen our arguments and contributions. However, we have identified certain limitations in these papers concerning a specific parameter that we aim to highlight in our study. We have strategically positioned our research to demonstrate superiority across all relevant parameters by leveraging these identified gaps.

1) The first method, MTDALR, Dongyoung Koo et al. [14], addresses continuous information leakage that can occur due to repetitions of the authentication process. The consumption of energy and memory is high to do their proposed task.
2) The second method, EKAMT, Zajac [29], combines ephemeral public keys with a simple Merkle tree to obtain a server-authenticated key encapsulation suitable for Transport Layer Security like handshake protocols. Due to the operating time and memory requirements, it can be used only in controlled devices
3) The third method, IASMTM, Yi-Cheng Chen et al. [30], achieved image authentication and tampered area restoration by utilizing a Merkle tree. However, it takes longer to authenticate, has lower throughput, and has a longer end-to-end delay.
4) The fourth method, SELCOM, Kim, et al. [31], proposes a selective compression scheme. This scheme helps prevent accumulated compression results using a unique checkpoint chain. Here security is a concern, and processing time could be higher.
5) Finally, in CMT, Mingchao Yu et al. [32] proposed a Coded Merkle tree, which enables any node to detect tree manipulation by downloading only (log b) bytes with the assistance of a single honest node. However, this approach exhibits a slower throughput and consumes more memory space.

By capitalizing on the identified gaps and utilizing them as an advantage, we have meticulously addressed the shortcomings through our novel research approach. Our objective is to present a comprehensive and compelling argument that establishes the superiority of our proposed PMT method. Our approach uses a new formation of the Merkle tree called Pruned Merkle Tree to overcome potential weaknesses, energy lessening, storage, and authentication requirements. We then compared our method's performance with the existing methods regarding throughput time, data security, and processing time and showed promising results.

4. PROPOSED PMT METHOD

This section presents a comprehensive exploration of the construction of the proposed system, offering a detailed overview of its architecture and key components.

4.1 The Proposed PMT – Construction

Merkle trees are time-consuming data structures and need many computational resources to reach the root. Figure 5 illustrates the PMT development, which eliminates node repetition. After successfully clearing all the consensus mechanisms, a data block is joined in the chain with Merkle root and for verification and integrity, accepts a few hash values, and refrains from imposing the Merkle tree on the entire data block. A block can hold 1MB of transactions; each transaction will have changed as hash values and are called nodes. The nodes have to pair it up until it reaches a root. If pairing is done without any shortage of nodes, reaching the Merkle root is easy, and the tree formation is called a Merkle tree. Our proposed PMT scheme plays a role in pairing nodes if a node is left unpaired at any level. Due to the high level of data in the Blockchain, any third party cannot manipulate the integrity and transactions accommodated in a block. The construction steps of the Pruned Merkle Tree (PMT) for a block are outlined as follows:

1) Allocate key (1) to the first node, denoted as n1, with the corresponding hash value of 1 in the PMT for the block.
2) In succession, join left and right nodes in pairs, not including the first block with key (1), for a total of ‘ts’ transactions.
3) Apply the hash function to each divided node from key 1 to ‘ts’, generating their respective hash values.
4) Proceed to construct the root of the Merkle Tree by pairing the generated hash values until reaching the root.

The PMT for the block can be effectively constructed by following these steps.

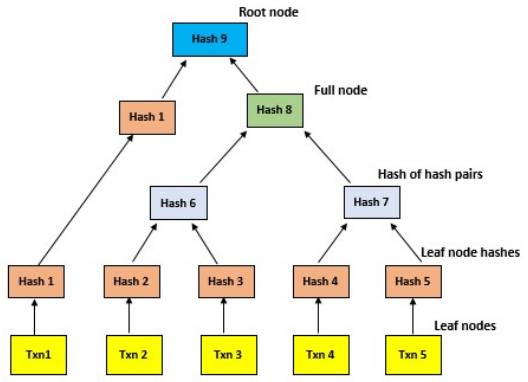

Figure 5. Pruned Merkle Tree Verification Method

In addition, the existing Merkle tree requires massive memory to store all of the hash values. But the proposed PMT uses comparatively less memory to store the transactions. This has been shown in Figures 6a and 6b, comparing transactions with the existing Merkle tree (EMT) and Pruned Merkle Tree (PMT) to form the root for three and five transactions, respectively. The number of nodes 'n' clearly shows the scheme’s difference.”

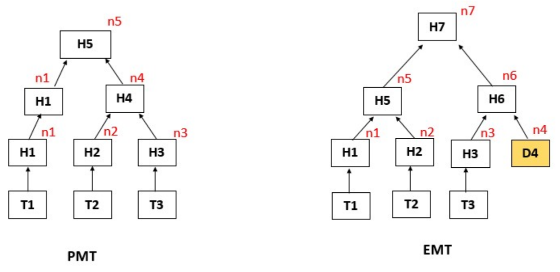

Figure 6a. Transactions Comparison With The Existing Merkle Tree (EMT) And Pruned Merkle Tree (PMT) For Three Transactions

PMT stands tall as it can eliminate repetition and the number of nodes it uses for forming the tree transactions. Compared to the existing Merkle Tree, the Pruned Merkle Tree removes node repetition and moderates the number of nodes.

The reduction of usage in the nodes can be shown in an equation

Number of nodes ts=n+(n-2) +1

ts=2n-2+1

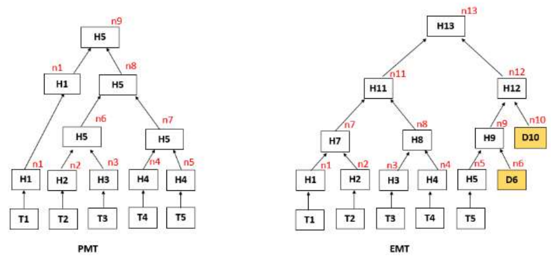

Figure 6b. Transactions Comparison With The Existing Merkle Tree (EMT) And Pruned Merkle Tree (PMT) For Five Transactions.

The primary advantage is the removal of node repetition. When there are fewer nodes, traversal becomes faster. The goal of the PMT is to remove the repetition of nodes when a block contains an unpaired node and must be set right at every tree level. When nodes are verified, PMT has incredible security too.

The Pruned Merkle Tree structure operates as follows: the first node remains unchanged, and the pairing process commences from the second node onwards. When the last node in the block lacks a partner node for pairing, the first node ascends towards the top of the tree without disturbing other nodes. Eventually, it combines with the remaining node to create the root node. This scenario occurs when the transaction node is unpaired in the block.

4.2 The Proposed PMT Scheme – Overview

The overview of the proposed PMT has been shown with the neat arrangement and the different cases.

Arrangement: Blockchain is made up of systematically ordered blocks. Every block will include a block header, a nonce, a timestamp, a hash, a previous hash, and transactions. To validate a blockchain and add new blocks, consensus mechanisms are used. From the pool of transactions, transactions will be chosen to join the block. Figure 7 shows the overview of the proposed system.

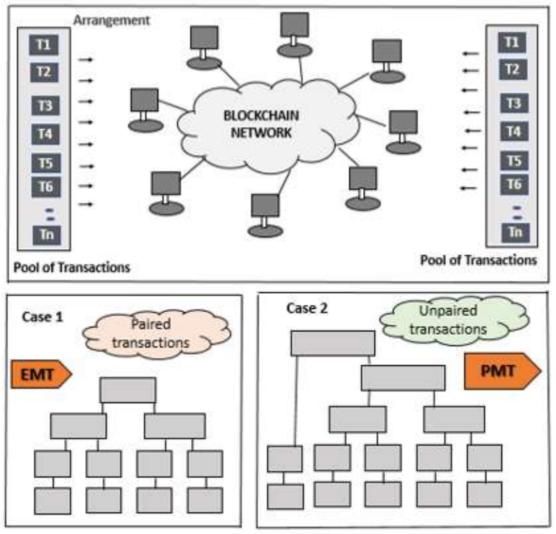

Figure 7. The Proposed System

Case 1: Block with paired nodes (transactions), $N=2^n$ transactions, where n=1, 2, 3…∞

Nodes $N = 2^1, 2^2, 2^3, \ldots 2^n$. [N= 2,4,8,16,32,64,128,…] No repetition is needed.

Case 2: Block with unpaired nodes (transactions), $N=2^n$, and N-1 transactions**.** where n=1, 2, 3…∞

According to case 2, $N=2^n$ and N-1, if n=4, then $2^4 = 16$ transactions and N-1=15 transactions, needs repetition.

4.3 The Proposed PMT Scheme – Algorithm

Here is the algorithm for constructing the PMT, considering both paired and unpaired transactions.

Define the sets:
N: Paired nodes
N-1: Unpaired nodes

1. Initialize variables:

   ts: Total no. of transactions (initially set to 0)
   H: Hash of the node (initially set to null)
   Tree: List to store intermediate hashes and root nodes

```
2. Repeat the following steps while ts < N:
   a. Increment n by 1 (n++)
   b. Append transaction t_n with its
      corresponding hash H_n to Tree.
      Tree.append(H_n)
   c. Increment tn by 1 (ts++)

3. If ts = N-1, perform the following steps:
   a. Skip appending t1(H1) to Tree.
4. For any n ≥ 1 and 1 ∈ N-1, perform the
   following steps:
   a. Increment n by 1 (n++)
   b. Append transaction ts with its corresponding
      hash H_n to Tree. Tree. append(H_n)
5. Compute the Merkle root:
   a. Repeat until the length of the Tree is 1:
      i.  Initialize an empty list of Temp
      ii. Iterate over Tree in pairs, taking two
          consecutive elements:
          (1) Concatenate the two elements to form
              a new hash H_new
          (2) Append H_new to Temp
      iii. If the length of the Tree is odd, append
          the last element of the Tree to Temp
      iv. Set Tree as Temp
   b. The last remaining element in Tree is the
      Merkle root (Hn)
6. Return Hn as the Merkle root of the transaction
   set.
```

4.4 Addressing the apprehensions in our proposed method

In our proposed method, leaving the hash of the first transaction to compute the Merkle root may initially raise concerns about the security of the stored transactions. We accept the potential vulnerability the researchers can dubious, wherein an attacker could tamper with the first transaction and compute a new Merkle root and the second top layer's hash value. However, we believe that our proposed method remains highly secure due to the following reasons:

1. Enhanced Efficiency: The computational cost required to tamper with the first transaction in our method is significantly lower than in existing Merkle trees. This trade-off allows for faster computation and verification processes, making our method more practical for real-time applications.
2. Overall Blockchain Security: While the proposed method may present a potential weakness at the first transaction, it is crucial to consider the holistic security provided by the blockchain system. Blockchain security relies on the Merkle tree structure and other critical components, such as consensus mechanisms and distributed network validation. These elements collectively reinforce the overall security of the blockchain, mitigating the risks associated with isolated vulnerabilities.
3. Additional Security Measures: Our proposed method can be combined with other security measures to mitigate the potential risk of the first transaction. For instance, incorporating digital signatures or salt values can provide an additional layer of security, making it more challenging for an attacker to tamper with transactions without detection.
4. Context-Specific Use Cases: Our proposed method may suit specific use cases where security and efficiency trade-offs align with the desired objectives. In scenarios where real-time transaction processing or high-speed data verification is of utmost importance, the proposed method offers a practical solution without compromising the overall security of the blockchain system.

5. SECURITY ANALYSIS OF THE PROPOSED WORK

The primary goal of blockchain is creating, validating, and storing data blocks in a decentralized, trustworthy behaviour while maintaining consistency. Security analysis is an iterative process that encourages ongoing monitoring, testing, and improvement of security measures. PMT security analysis aims to ensure the confidentiality, integrity, and availability [33] of systems, algorithms, or applications and to reduce the likelihood and impact of security incidents. It is crucial to note that our proposed method aims to address certain applications' specific challenges and requirements. Through rigorous analysis and comparison with traditional methods, we have demonstrated that our proposed PMT approach maintains high security while offering valuable efficiency improvements in specific use cases.

Pruned Merkle tree structure provides a strong cryptographic design for transaction processing; preimage attacks are one of the potential vulnerabilities that can impact their security. The underlying hash function used in the proposed algorithm possesses preimage resistance [5]. Consider PMT with five transactions (tn = 5).

1. Assigning the index and hash values:
   - n1: Index 1, Hash Value = 1
   - n2: Index 2, Hash Value = H(n1)
   - n3: Index 3, Hash Value = H(n2)
   - n4: Index 4, Hash Value = H(n3)
   - n5: Index 5, Hash Value = H(n4)
2. Construction of the Merkle Tree:
   - Level 1: n1 | n2 | n3 | n4 | n5
   - Level 2: H(n1) | H(n2, n3) | H(n4, n5)
   - Level 3: H(n1), H(H(n2, n3), H(n4, n5)) (Merkle Root)

   Suppose an attacker wants to find a preimage for a specific hash value in the Pruned Merkle tree, such as H(n3). The attacker would need to find an input that, when hashed, produces the hash value H(n3).
3. Preimage Resistance:
   - Definition: A hash function is considered preimage-resistant if it is computationally infeasible to find an input M given its hash value H(M) within a reasonable timeframe.
   - Assumption: Let's assume we have a cryptographic hash function, denoted as H, that is preimage-resistant.
   - Proof by contradiction: Suppose an attacker can find an input M' such that H(M') = given hash value H(M).
   - This implies finding a preimage, which contradicts the assumption that H is preimage-resistant.
   - Therefore, the hash function H is considered preimage-resistant.

The attacker would have to exhaustively search all possible inputs and compute their hashes until they find one that matches H(n3). The process of attacking would require a significant amount of time and computational resources. The security of the algorithm's preimage resistance heavily relies on the strength of the chosen hash function. A widely recognized and robust hash function, such as SHA-256, helps ensure the desired preimage resistance property.

Considering the proposed algorithm for PMT, preimage attacks aimed at tampering with the first transaction are highly improbable. Constructing a new transaction with the same hash value as an existing transaction is incredibly challenging. The properties of the chosen cryptographic hash function, such as preimage resistance, make it extremely unlikely for an attacker to find an input that produces the same hash value as an existing transaction. Therefore, the security of the first transaction against preimage attacks is significantly enhanced within the proposed algorithm.

6. IMPLEMENTATION OF PMT

This section presents the implementation of Pruned Merkle tree. The parameters we took are Throughput, Average Processing Time, Average Energy, Average Authentication Time, Security, and Memory.

6.1 Hyperledger Caliper

Hyperledger Caliper serves as a dedicated blockchain benchmark tool that enables users to evaluate the performance of a particular blockchain implementation using a predefined set of use cases. It is specifically designed to assess the performance of blockchain frameworks [34]. It's important to note that Hyperledger Caliper requires a functioning blockchain implementation as the benchmarking target. To enhance the efficiency of the benchmarking process, it is ideal to collaborate with tools that facilitate the quick setup of a blockchain network. By utilizing Hyperledger Caliper in conjunction with devices that expedite the creation of a blockchain network, we can leverage the benchmarking capabilities of Caliper to accurately measure and compare the performance of different blockchain frameworks, including our proposed system. This integrated approach ensures a comprehensive evaluation and enables us to demonstrate our system's superiority in metrics such as throughput and latency. [35] [36]. The algorithms were implemented as Visual Studio 16.0 programs using the Hyper Caliper, a cryptographic analysis tool. Pruned Merkle tree was constructed by dividing the data into 256-byte blocks to ensure consistent comparisons. Multiple iterations of each parameter were executed within the same environment to generate an optimized report, which compared the proposed approach with the existing method. The parameters were then computed and documented. The computation time for each experiment was measured in terms of CPU time. The performance of each algorithm was analyzed across different data sizes, considering metrics such as throughput, end-to-end delay, average processing time, average authentication time, average energy consumption, packet delivery ratio, security, and memory utilization.

6.2 Throughput

The consistency of PMT's throughput time has been substantiated, making it a valuable baseline efficiency metric in our experiment. To provide empirical evidence of PMT's superiority, we have included graphical representations in Figure 8. The figure demonstrates that PMT outperforms other existing methods in terms of throughput. This visual representation of the data supports our claim of PMT's enhanced efficiency and reinforces its competitive advantage over alternative approaches.

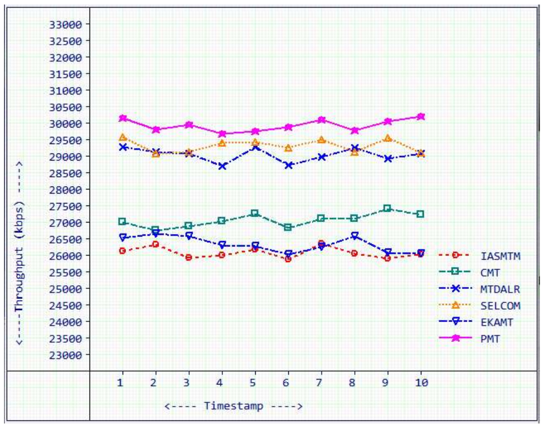

Figure 8. Throughput Time

6.3 Average Processing Time

The processing time for a job holds significant importance as it defines the efficacy of a work methodology. In the case of PMT, it accomplishes the given task within a remarkable 1660 milliseconds. This fact can be verified through multiple timestamps, ensuring the consistency and reliability of the results. For a more comprehensive understanding, Figure 9 provides a detailed graphical representation that visually depicts the processing time achieved by PMT. This graph is compelling evidence of PMT's efficiency and demonstrates its ability to deliver swift and reliable performance in completing tasks.

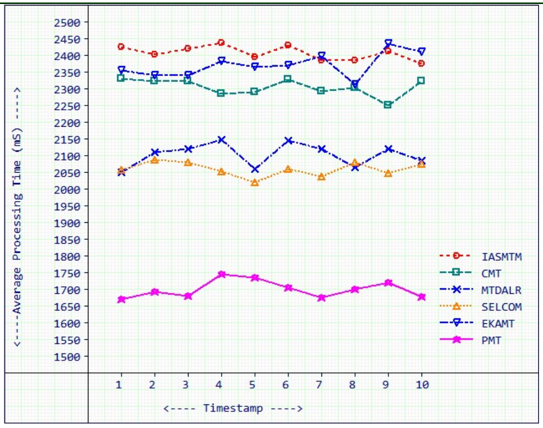

Figure 9. The Average Processing Time

6.4 Average Energy

Energy consumption is a crucial factor to consider in the context of blockchain systems [37] [38]. Different methods exhibit varying energy usage levels depending on the process's duration. Cesar et al. [24] implement an algorithmic engineering technique that reduces energy consumption for calculating Merkle Tree roots, which are essential for blockchain computations. Escobar et al. [28] shows substantial reductions in energy consumption are achievable. Figure 10 presents compelling evidence of PMT's low energy consumption, with the lowest recorded value of 235 joules. This graph provides a clear visual representation and empirical support for the energy efficiency of PMT compared to other methods. The significant reduction in energy consumption showcased by PMT underscores its environmentally friendly nature and potential to contribute to sustainable blockchain solutions.

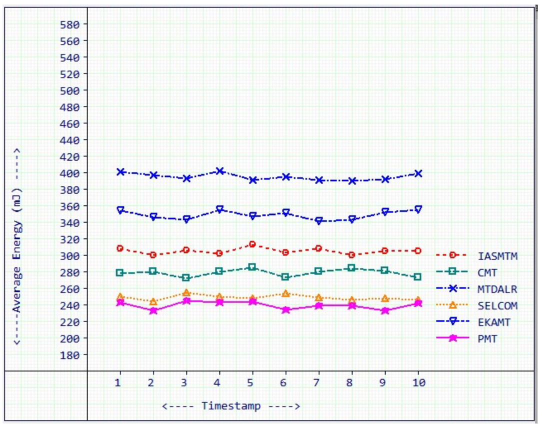

Figure 10. Energy Consumption

6.5 Average Authentication Time

When evaluating the proposed PMT method, it is crucial to consider the average authentication time, which serves as an important performance metric. Compared to IASMTM, which exhibits the maximum authentication time, and MTDALR, which has a medium authentication time, PMT demonstrates superior efficiency with an average authentication time of fewer than 105 milliseconds. These comparisons are visually presented in Figure 11, providing clear evidence of PMT's faster authentication process. The graph highlights the significant performance advantage of PMT over the alternative methods, affirming its efficacy in achieving swift and reliable authentication in blockchain systems.

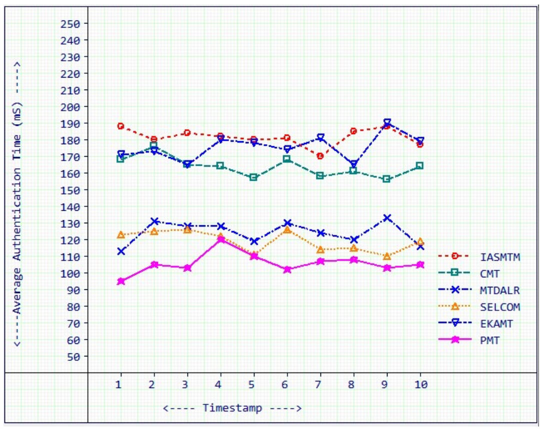

Figure 11. Average Authentication Time

6.6 Security

Security is paramount in decentralized systems, particularly in applications like blockchain. Ensuring data security is critical to maintaining the integrity and trustworthiness of the system. In Figure 12, we present compelling evidence of PMT's robust security, with an impressive rating of 99.6%. This graph is a testament to PMT's unwavering commitment to security, highlighting its ability to maintain high data protection without compromising other performance metrics. With PMT, security remains a top priority, ensuring the system maintains its integrity and safeguards against potential threats or vulnerabilities.

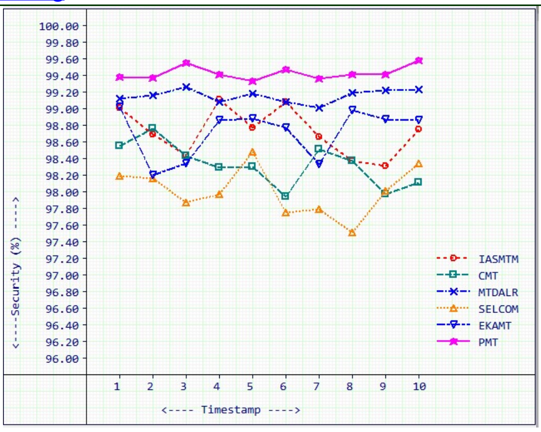

Figure 12. Security

6.7 Memory

Memory consumption is an ongoing issue when it comes to developing new applications, tools, or methods. Due to the inclusion of hash values for every data entry within a block, scalability becomes a significant concern in the case of blockchain technology, resulting in duplicated entries in the ledger. Memory consumption may increase as a result of this duplication. However, our proposed PMT method effectively addresses this issue.

In Figure 13, we present the results of our experiment, demonstrating that PMT consumes significantly less memory compared to existing methods. For a ledger size of 1000 blocks, PMT utilizes only 145 MB of memory. This impressive reduction in memory consumption is achieved by eliminating duplicate nodes in the existing Merkle Tree structure, thereby optimizing the storage requirements of the blockchain.

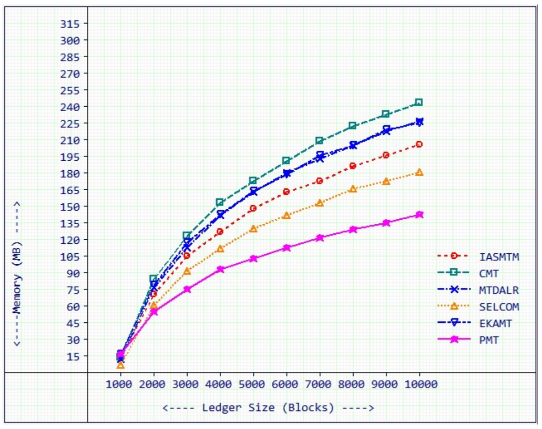

Figure 13. Memory

Journal of Theoretical and Applied Information Technology
31st October 2023. Vol.101. No 20
© 2023 Little Lion Scientific

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

The findings presented in the text effectively demonstrate PMT's superior performance in terms of throughput, processing time, energy efficiency, and authentication speed, backed by compelling graphical evidence. The presentation of PMT's low energy consumption in Figure 10 aligns with the growing concern for energy efficiency in blockchain, emphasizing PMT's environmental friendliness and potential for sustainable solutions. The security rating of 99.6% and the memory efficiency results further reinforce PMT's strength as a robust and resource-efficient solution for blockchain applications. We maintain a respectful and objective stance toward the existing papers throughout our research paper, recognizing their value within the academic and scientific community. However, by building upon their insights and incorporating our unique perspective, we have demonstrated how our research surpasses previous work regarding the identified parameters.

7. CONCLUSION AND FUTURE WORK

The research findings unequivocally demonstrate the effectiveness of the Pruned Merkle Tree construction in eliminating hash duplication. By leveraging this innovative structure, PMT successfully addresses the repetition issue, particularly when dealing with the block with unpaired nodes. The ingenious data-adding technique employed by PMT results in a consistent and compact data size, further enhancing its performance. PMT's unique approach ensures energy consumption remains well-controlled, enabling it to outperform other methods in transaction verification. The proposed data-adding technique in PMT contributes to a limited and constant size, significantly enhancing performance. PMT is pivotal in driving sustainable energy consumption, efficient resource utilization, and heightened security within blockchain technology. Its impact extends beyond mere transaction processing, making it a key enabler for the future of secure and environmentally conscious blockchain systems. These findings collectively reaffirm PMT as a ground-breaking and impactful innovation, poised to reshape the landscape of blockchain technology, offering enhanced efficiency and security for decentralized systems. In future research, we intend to explore integrating the Pruned Merkle Tree with the Bloom-filter algorithm. This PMT implementation will prioritize data integrity, offering heightened assurance. By merging these two powerful techniques, we aim to enhance the performance and security of PMT further, paving the way for even more robust and reliable blockchain systems.

REFERENCES:

[1] S. Dhumwad, M. Sukhadeve, C. Naik, M. K.N., and S. Prabhu, “A Peer to Peer Money Transfer Using SHA256 and Merkle Tree,” in *2017 23RD Annual International Conference in Advanced Computing and Communications (ADCOM)*, Bangalore, India: IEEE, Sep. 2017, pp. 40–43. doi: 10.1109/ADCOM.2017.00013.

[2] Satoshi Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System,” p. 9.

[3] R. Nagayama, R. Banno, and K. Shudo, “Trail: A Blockchain Architecture for Light Nodes,” in *2020 IEEE Symposium on Computers and Communications (ISCC)*, Rennes, France: IEEE, Jul. 2020, pp. 1–7. doi: 10.1109/ISCC50000.2020.9219673.

[4] H. Liu, X. Luo, H. Liu, and X. Xia, “Merkle Tree: A Fundamental Component of Blockchains,” in *2021 International Conference on Electronic Information Engineering and Computer Science (EIECS)*, Changchun, China: IEEE, Sep. 2021, pp. 556–561. doi: 10.1109/EIECS53707.2021.9588047.

[5] E. Andreeva *et al.*, “New Second-Preimage Attacks on Hash Functions,” *J. Cryptol.*, vol. 29, no. 4, pp. 657–696, Oct. 2016, doi: 10.1007/s00145-015-9206-4.

[6] Z. Wang, J. Lin, Q. Cai, Q. Wang, D. Zha, and J. Jing, “Blockchain-Based Certificate Transparency and Revocation Transparency,” *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 1, pp. 681–697, Jan. 2022, doi: 10.1109/TDSC.2020.2983022.

[7] H. Guo and X. Yu, “A survey on blockchain technology and its security,” *Blockchain Res. Appl.*, vol. 3, no. 2, p. 100067, Jun. 2022, doi: 10.1016/j.bcra.2022.100067.

[8] B. Rebecca Jeyavadhanam, M. Gracy, V. V. Ramalingam, and Christopher Xavier, “Fundamental Concepts and Applications of Blockchain Technology,” [Online]. Available: https://www.taylorfrancis.com/chapters/edit/10.1201/9781003032328-8/fundamental-concepts-applications-blockchain-technology-rebecca-jeyavadhanam-gracy-ramalingam-christopher-xavier?context=ubx&refId=4ce1cdef-18de-4085-b4c2-297e4e683949

[9] Y. Maleh, M. Shojafar, M. Alazab, and I. Romdhani, Eds., *Blockchain for cybersecurity and privacy: architectures, challenges, and*

applications, First edition. in Internal audit and it audit. Boca Raton London New York: CRC Press, Taylor & Francis Group, 2020.

[10] M. Gracy and B. Rebecca Jeyavadhanam, “A Systematic Review of Blockchain-Based System: Transaction Throughput Latency and Challenges,” in *2021 International Conference on Computational Intelligence and Computing Applications (ICCICA)*, Nagpur, India: IEEE, Nov. 2021, pp. 1–6. doi: 10.1109/ICCICA52458.2021.9697142.

[11] R. C. Merkle, “A Digital Signature Based on a Conventional Encryption Function,” in *Advances in Cryptology — CRYPTO ’87*, C. Pomerance, Ed., in Lecture Notes in Computer Science, vol. 293. Berlin, Heidelberg: Springer Berlin Heidelberg, 1988, pp. 369–378. doi: 10.1007/3-540-48184-2_32.

[12] Haitz Sáez de Ocáriz Borde, “An Overview of Trees in Blockchain Technology: Merkle Trees and Merkle Patricia Tries”, [Online]. Available: https://www.researchgate.net/publication/358740207_An_Overview_of_Trees_in_Blockchain_Technology_Merkle_Trees_and_Merkle_Patricia_Tries

[13] U. Chelladurai and S. Pandian, “HARE: A new Hash-based Authenticated Reliable and Efficient Modified Merkle Tree Data Structure to Ensure Integrity of Data in the Healthcare Systems,” *J. Ambient Intell. Humaniz. Comput.*, Apr. 2021, doi: 10.1007/s12652-021-03085-0.

[14] D. Koo, Y. Shin, J. Yun, and J. Hur, “Improving Security and Reliability in Merkle Tree-Based Online Data Authentication with Leakage Resilience,” *Appl. Sci.*, vol. 8, no. 12, p. 2532, Dec. 2018, doi: 10.3390/app8122532.

[15] W. Zhai, K. Qi, J. Duan, and C. Cheng, “Merkle quad-tree based remote sensing image analysis,” in *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, Fort Worth, TX: IEEE, Jul. 2017, pp. 6193–6196. doi: 10.1109/IGARSS.2017.8128423.

[16] Y. Suga, “Formulation of Information Hiding Model for One-Time Authentication Methods Using the Merkle Tree,” in *Advances in Internet, Data and Web Technologies*, L. Barolli, F. Xhafa, Z. A. Khan, and H. Odhabi, Eds., in Lecture Notes on Data Engineering and Communications Technologies, vol. 29. Cham: Springer International Publishing, 2019, pp. 363–373. doi: 10.1007/978-3-030-12839-5_33.

[17] N. Adhikari, N. Bushra, and M. Ramkumar, “Complete Merkle Hash Trees for Large Dynamic Spatial Data,” in *2019 International Conference on Computational Science and Computational Intelligence (CSCI)*, Las Vegas, NV, USA: IEEE, Dec. 2019, pp. 1318–1323. doi: 10.1109/CSCI49370.2019.00246.

[18] L. Ramabaja and A. Avdullahu, “Compact Merkle Multiproofs,” 2020, doi: 10.48550/ARXIV.2002.07648.

[19] Ripon Patgiri, Malaya Dutta Borah, “HEX-BLOOM: An Efficient Method for Authenticity and Integrity Verification in Privacy-preserving Computing”, [Online]. Available: https://eprint.iacr.org/2021/773.pdf

[20] J. Kan and K. S. Kim, “MTFS: Merkle-Tree-Based File System,” in *2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, Seoul, Korea (South): IEEE, May 2019, pp. 43–47. doi: 10.1109/BLOC.2019.8751389.

[21] M. Jakobsson, T. Leighton, S. Micali, and M. Szydlo, “Fractal Merkle Tree Representation and Traversal,” in *Topics in Cryptology — CT-RSA 2003*, M. Joye, Ed., in Lecture Notes in Computer Science, vol. 2612. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 314–326. doi: 10.1007/3-540-36563-X_21.

[22] D. Koo, Y. Shin, J. Yun, and J. Hur, “An Online Data-Oriented Authentication Based on Merkle Tree with Improved Reliability,” in *2017 IEEE International Conference on Web Services (ICWS)*, Honolulu, HI, USA: IEEE, Jun. 2017, pp. 840–843. doi: 10.1109/ICWS.2017.102.

[23] J. Wang, B. Wei, J. Zhang, X. Yu, and P. K. Sharma, “An optimized transaction verification method for trustworthy blockchain-enabled IIoT,” *Ad Hoc Netw.*, vol. 119, p. 102526, Aug. 2021, doi: 10.1016/j.adhoc.2021.102526.

[24] C. Castellon, S. Roy, P. Kreidl, A. Dutta, and L. Boloni, “Energy Efficient Merkle Trees for Blockchains,” in *2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, Shenyang, China: IEEE, Oct. 2021, pp. 1093–1099. doi: 10.1109/TrustCom53373.2021.00149.

[25] John Kuszmaul, “Verkle Trees”, [Online]. Available: https://math.mit.edu/research/highschool/primes/materials/2018/Kuszmaul.pdf

[26] F. Bruschi, V. Rana, A. Pagani, and D. Sciuto, “Tunneling Trust Into the Blockchain: A Merkle Based Proof System for Structured

Documents," *IEEE Access*, vol. 9, pp. 103758–103771, 2021, doi: 10.1109/ACCESS.2020.3028498.

[27] R. J. Bayardo and J. Sorensen, "Merkle tree authentication of HTTP responses," in *Special interest tracks and posters of the 14th international conference on World Wide Web - WWW '05*, Chiba, Japan: ACM Press, 2005, p. 1182. doi: 10.1145/1062745.1062929.

[28] C. C. Escobar, S. Roy, O. P. Kreidl, A. Dutta, and L. Boloni, "Toward a Green Blockchain: Engineering Merkle Tree and Proof of Work for Energy Optimization," *IEEE Trans. Netw. Serv. Manag.*, vol. 19, no. 4, pp. 3847–3857, Dec. 2022, doi: 10.1109/TNSM.2022.3219494.

[29] P. Zajac, "Ephemeral Keys Authenticated with Merkle Trees and Their Use in IoT Applications," *Sensors*, vol. 21, no. 6, p. 2036, Mar. 2021, doi: 10.3390/s21062036.

[30] Y.-C. Chen, Y.-P. Chou, and Y.-C. Chou, "An Image Authentication Scheme Using Merkle Tree Mechanisms," *Future Internet*, vol. 11, no. 7, p. 149, Jul. 2019, doi: 10.3390/fi11070149.

[31] T. Kim, S. Lee, Y. Kwon, J. Noh, S. Kim, and S. Cho, "SELCOM: Selective Compression Scheme for Lightweight Nodes in Blockchain System," *IEEE Access*, vol. 8, pp. 225613–225626, 2020, doi: 10.1109/ACCESS.2020.3044991.

[32] M. Yu, S. Sahraei, S. Li, S. Avestimehr, S. Kannan, and P. Viswanath, "Coded Merkle Tree: Solving Data Availability Attacks in Blockchains," 2019, doi: 10.48550/ARXIV.1910.01247.

[33] F. Thabit, S. Alhomdy, and S. Jagtap, "Security analysis and performance evaluation of a new lightweight cryptographic algorithm for cloud computing," *Glob. Transit. Proc.*, vol. 2, no. 1, pp. 100–110, Jun. 2021, doi: 10.1016/j.gltp.2021.01.014.

[34] "Measuring Blockchain Performance with Hyperledger Caliper." [Online]. Available: https://www.hyperledger.org/blog/2018/03/19/measuring-blockchain-performance-with-hyperledger-caliper

[35] M. Sreenu, N. Gupta, C. Jatoth, A. Saad, A. Alharbi, and L. Nkenyereye, "Blockchain based secure and reliable Cyber Physical ecosystem for vaccine supply chain," *Comput. Commun.*, vol. 191, pp. 173–183, Jul. 2022, doi: 10.1016/j.comcom.2022.04.031.

[36] T. Guimarães, R. Duarte, B. Pinheiro, D. Faria, P. Gomes, and M. F. Santos, "Blockchain Analytics - Real-time Log Management in Healthcare," *Procedia Comput. Sci.*, vol. 201, pp. 702–707, 2022, doi: 10.1016/j.procs.2022.03.094.

[37] J. Sedlmeir, H. U. Buhl, G. Fridgen, and R. Keller, "The Energy Consumption of Blockchain Technology: Beyond Myth," *Bus. Inf. Syst. Eng.*, vol. 62, no. 6, pp. 599–608, Dec. 2020, doi: 10.1007/s12599-020-00656-x.

[38] J. Truby, "Decarbonizing Bitcoin: Law and policy choices for reducing the energy consumption of Blockchain technologies and digital currencies," *Energy Res. Soc. Sci.*, vol. 44, pp. 399–410, Oct. 2018, doi: 10.1016/j.erss.2018.06.009.