





Please cite the Published Version

Jabbar, S , Abideen, ZU, Khalid, S , Ahmad, A , Raza, U  and Akram, S (2023) Enhancing computational scalability in Blockchain by leveraging improvement in consensus algorithm. *Frontiers in Computer Science*, 5. 1304590

DOI: <https://doi.org/10.3389/fcomp.2023.1304590>

Publisher: Frontiers Media SA

Version: Published Version

Downloaded from: <https://e-space.mmu.ac.uk/633841/>

Usage rights:  [Creative Commons: Attribution 4.0](https://creativecommons.org/licenses/by/4.0/)

Additional Information: This is an Open Access article published in *Frontiers in Computer Science*, by Frontiers Media SA.

Data Access Statement: The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

Enquiries:

If you have questions about this document, contact openresearch@mmu.ac.uk. Please include the URL of the record in e-space. If you believe that your, or a third party's rights have been compromised through this document please see our Take Down policy (available from <https://www.mmu.ac.uk/library/using-the-library/policies-and-guidelines>)



OPEN ACCESS

EDITED BY

Muhammad Usman Tariq,
Abu Dhabi University, United Arab Emirates

REVIEWED BY

Babar Khan,
Princeton University, United States
Miteen Ali,
North Dakota State University, United States

*CORRESPONDENCE

Sohail Jabbar

✉ sjjabar@imamu.edu.sa

Zain Ul Abideen

✉ bscs-fa15-154@tuf.edu.pk

RECEIVED 29 September 2023

ACCEPTED 10 November 2023

PUBLISHED 07 December 2023

CITATION

Jabbar S, Abideen ZU, Khalid S, Ahmad A,
Raza U and Akram S (2023) Enhancing
computational scalability in Blockchain by
leveraging improvement in consensus
algorithm. *Front. Comput. Sci.* 5:1304590.
doi: 10.3389/fcomp.2023.1304590

COPYRIGHT

© 2023 Jabbar, Abideen, Khalid, Ahmad, Raza
and Akram. This is an open-access article
distributed under the terms of the [Creative
Commons Attribution License \(CC BY\)](#). The use,
distribution or reproduction in other forums is
permitted, provided the original author(s) and
the copyright owner(s) are credited and that
the original publication in this journal is cited, in
accordance with accepted academic practice.
No use, distribution or reproduction is
permitted which does not comply with these
terms.

Enhancing computational scalability in Blockchain by leveraging improvement in consensus algorithm

Sohail Jabbar ^{1*}, Zain Ul Abideen ^{2*}, Shehzad Khalid ³,
Awais Ahmad ¹, Umar Raza ⁴ and Sheeraz Akram ¹

¹College of Computer and Information Sciences, Imam Mohammad Ibn Saud Islamic University (IMSIU), Riyadh, Saudi Arabia, ²Department of Computer Science, The University of Faisalabad, Faisalabad, Pakistan, ³Department of Computer Engineering, Bahria University, Islamabad, Pakistan, ⁴Department of Engineering, Manchester Metropolitan University, Manchester, United Kingdom

Accommodating an increasing number of users in the Blockchain network has moved to the forefront of discussion. It is also evident that without jeopardizing the data security in Blockchain, it is of indispensable need to devise an appropriate method for improving the scalability trait of Blockchain. In this article, we have proposed a consensus method that is having the potential to improve the scalability of the Private Blockchain. The system, at first, mitigates latency arising from kernel schedulers, ensuring that the application consistently has access to an available core for transaction processing. Secondly, the committee system alleviates the network's workload, preventing spurious transactions from monopolizing network resources and impeding its efficiency. Extensive experimentation is made by considering various scenarios of transaction with CPU isolation and application sticking to core 2 with varied priority. Based on the number of transactions performed per second, the proposed system is compared with different existing consensus mechanisms working in various types of Blockchains. Also, a detailed discussion is presented on the critical analysis of the adopted research mechanism. Overall, the proposed systems outperforms to other systems in various parameters of blockchain network scalability.

KEYWORDS

distributed ledger, transaction, algorithm, scalability, consensus protocol, Blockchain, Linux based distribution, supply chain

1 Introduction

In today's digital era, characterized by the rapid integration of new technologies into global networks, optimizing user management in financial transactions has become a paramount challenge for organizations. Blockchain technology has emerged as a transformative solution, functioning as a distributed database for data sharing among computer network nodes. This digital database, akin to its traditional counterparts, efficiently stores electronic data. Blockchain-based system is an amalgamation of cryptography, public key infrastructure, and economic modeling applied to peer-to-peer networking and decentralized consensus to achieve distributed database synchronization. Since the Blockchain is at its infant stage in comparison to the well-established systems of Visa, and PayPal. So is its transaction rate (Xu et al., 2022). This research gap makes this different. But the fact is that, by shifting an architecture, relying on centralized data servers to a trustless, and distributed peer-to-peer network, massive overhead costs, and concerns over centralized data control and single points of failure will be reduced significantly.

Blockchain with its salient features of decentralization, immutability, auditability, and fault-tolerant represents one of the most suitable candidate technologies able to support a better ecosystem (Jabbar et al., 2021). Extending it to the next step, although the technology itself is revolutionary, yet there are many limitations and challenges in its adoption in applications. From the long list of challenges, the mass adoption of Blockchain is largely limited by the issues of scalability, since it largely limits to cover the larger scale of any type of application. Understanding scalability requires a grasp of the underlying mechanisms: a distributed ledger system disseminates user records across multiple participants, triggering updates across all ledger instances upon any entry, fostering trust. Although, this is almost the same as the scalability issue is defined in general, yet, in common scenario, scalability is the ability of a system to continue to respond and function after increasing the size of the input in order to fulfill user demand. Further importance of scalability issue in Blockchain with respect to the confronting challenges and the available solutions is well discussed in recent publication Harshini Poojaa and Ganesh Kumar (2022) and Alshahrani et al. (2023).

Among many other factors that affect the scalability like block size, computation, communications, hardware and software limitations pose formidable challenges. In this context, each node continually adds transaction data, risking system stability from an ever-growing transaction history. Hardware upgrades become necessary as the network expands. Escalating transaction fees, a notable concern, stem from complex validation processes and varying user willingness to pay higher fees for faster verification, resulting in transaction backlogs. Block size also plays a critical role, with increasing transaction volumes elongating processing times (Yu et al., 2020). For instance, the Bitcoin Blockchain initially featured 1MB block sizes accommodating around 2020 transactions, but the surge in transactions necessitated larger block sizes, exacerbating scalability issues (Philippopoulos et al., 2019). Transaction validation, through the consensus process, encounters delays during peak hours, leading to higher fees and scalability challenges. Optimizing the network communication is also important not only for decreasing energy consumption but also for reducing the propagation delay. In the traditional Blockchain, each node is a relay node which broadcasts all transactions at least twice. When a transaction is generated, it first broadcasts to all nodes, and when a block containing the transaction is mined, it broadcasts to all nodes for the second time. This results in block propagation delay. Keeping in view this mechanism, the data transmission mode can not be scaled up to handle a large number of transactions due to the requirement for network bandwidth resources (Wang et al., 2022). Therefore, it is necessary to design more efficient data transmission mechanisms. More-over, the security concerns are not considered in the proposed system, since the system and the communication are assumed to be safe and may be considered in the Future work. So far as, the research work in the underlying article is concerned, Scalability issue confines the scope of the proposed system that is achieved by improving the computational aspect in Blockchain. The depiction of the core and usual process in blockchain transaction verification is given step by step in Figures 1–3 i.e., transaction initiation and its approval from leader node and hence by every other member node.

This paper provides a road map to improve the scalability of Blockchain using techniques and algorithms. Following enlisted are the key contributions of our research work.

- A real implementation of the proposed mechanism is made that gives a broader understanding of the Blockchain. This is unique in its kind. Since, It achieves the research thesis by mitigating the latency arising from kernel schedulers, ensuring that the application consistently has access to an available core for transaction processing. Moreover, the committee system alleviates the network's workload, preventing spurious transactions from monopolizing network resources and impeding its efficiency.
- A custom Linux-based distribution is performed to isolate the application to prevent hardware latency and to isolate the application's core.
- A comprehensive analysis of the current scaling solutions and the consensus algorithms with regard to their performance characteristics (such as latency and throughput) is made.

The subsequent sections of this paper are meticulously structured to facilitate comprehensive exploration of the research framework. Section 2 assesses existing supply chain traceability methodologies, highlighting limitations and providing insights into pharmaceutical supply chain organization. Section 3 outlines the research methodology, presenting algorithmic overviews. Section 4 reports empirical findings from rigorous experimentation, accompanied by in-depth discussions. Section 5 succinctly summarizes the study's insights and implications.

2 Literature review

Blockchain scalability has long been a prominent concern, prompting various algorithmic solutions and implementations. This paper examines existing solutions, revealing that the path to resolution lies not in a single algorithm or isolated approach but rather in the amalgamation of multiple solutions, particularly at the hardware level (Ali et al., 2022). While many solutions have scrutinized high-level programming aspects of Blockchain, there has been a conspicuous absence of exploration into lower-level approaches for addressing this issue.

One proposed solution is the Practical Byzantine Fault Tolerant (PBFT) algorithm (Feng et al., 2018), designed to resolve the Byzantine Generals Problem. This problem exemplifies the challenge faced by decentralized parties striving for consensus without a central trusted entity. In essence, it explores how decentralized parties can reach agreement when they lack the means to independently confirm each other's identities (Liu et al., 2022). This predicament is illustrated by generals surrounding Byzantium, needing to coordinate their attacks in unison rather than individually. The crux of the issue lies in their lack of communication channels to synchronize their efforts, along with the absence of trust in any delivered or deciphered communications. Importantly, decentralized systems are uniquely susceptible to the Byzantine Generals Problem due to their lack of a trustworthy information source and the inability to verify data

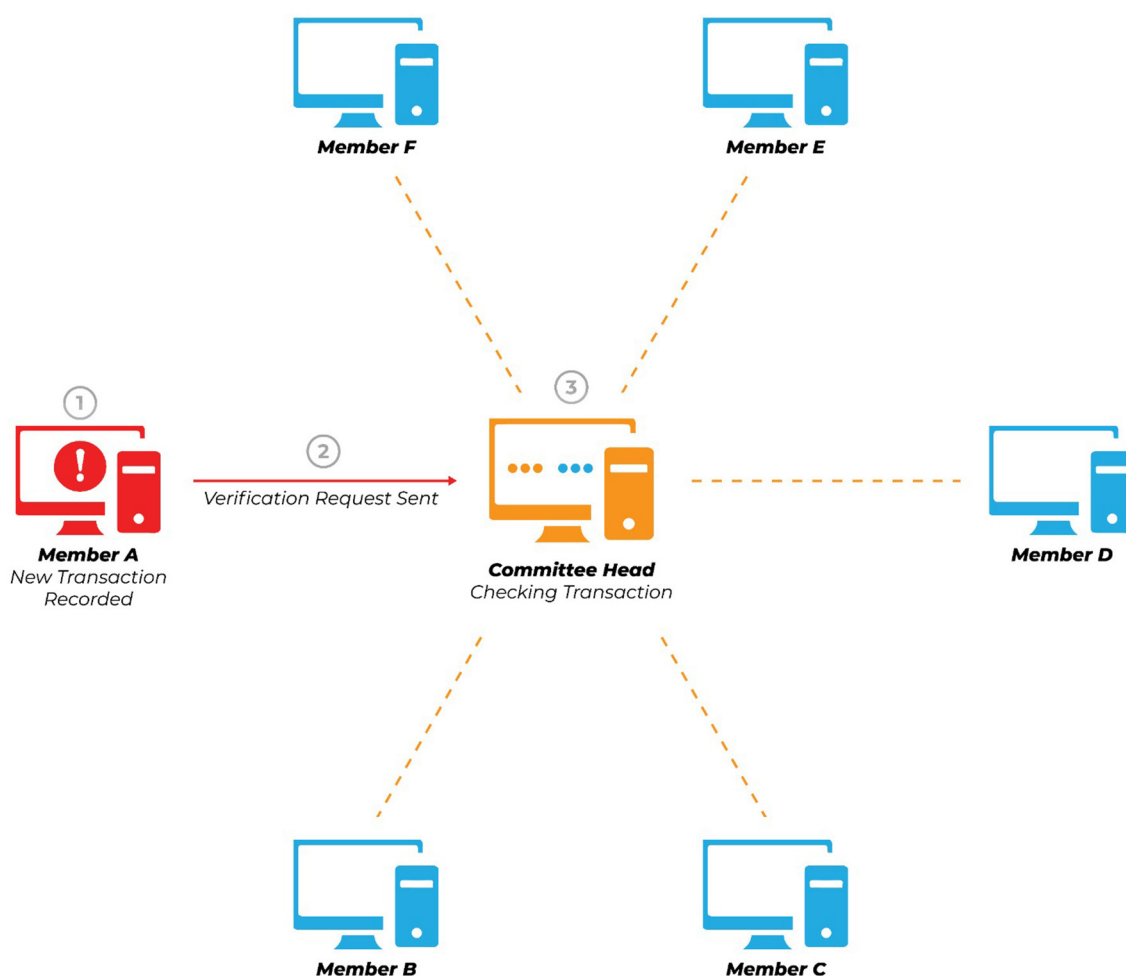


FIGURE 1

It shows that a committee member initiates a transaction, and the committee's leader has received it.

from other members, relying solely on a central authority (Bentov et al., 2014). Another pivotal contribution was the introduction of the Proof-of-Work (PoW) consensus algorithm, inspired by Adam Back's Hash Cash proposal from 2002, aimed initially at mitigating email spam and denial-of-service attacks (Seo et al., 2020). PoW served as the foundational mechanism for Bitcoin, predicated on solving computationally intensive hash problems, rendering block modifications practically infeasible without rehashing all subsequent blocks. PoW was envisioned as a peer-to-peer electronic currency, promising expedience, affordability, and accessibility while tackling the issue of double spending. Nevertheless, the solutions to double spending at the time demanded substantial hardware resources, outweighed by the advantages of enhanced security. The architecture entailed the creation of a complex and extensive peer-to-peer network, time stamping transactions using network timestamps and forming a Blockchain that rendered data modification highly improbable (Gucluturk, 2018). The decentralized nature of Blockchain, explored in this study, allows nodes to join and leave the network without constraints.

The Proof of Stake (PoS) consensus algorithm offers an alternative solution, where validators mine the next block

based on their economic stake in the network and the duration of their possession of network currency (Yang et al., 2019). Unlike PoW, PoS does not generate new coins or offer block rewards, relying solely on transaction fees. PoS, a permissioned public distributed ledger, minimizes the risk of a 51% attack, as attackers would undermine their own assets. This protocol is energy-efficient, requiring no specialized hardware and boasting good scalability due to minimal overhead. PoS draws inspiration from the concept of "coinage," a factor also influencing Bitcoin's transaction prioritization to some extent (Schedlbauer and Wagner, 2018). The concept of coinage is defined as "currency amount times holding period," essentially rewarding trust based on the age of the currency within the system. To protect against block-bloating attacks, transaction fees are enforced at the protocol level. Additionally, a third protocol, "proof-of-excellence," is briefly discussed, suggesting a tournament-based approach with intermittent rewards based on miner performance. However, power consumption concerns have tempered further exploration of this concept (Lone et al., 2019). The structure of Proof-of-Stake transaction is shown in Figure 4.

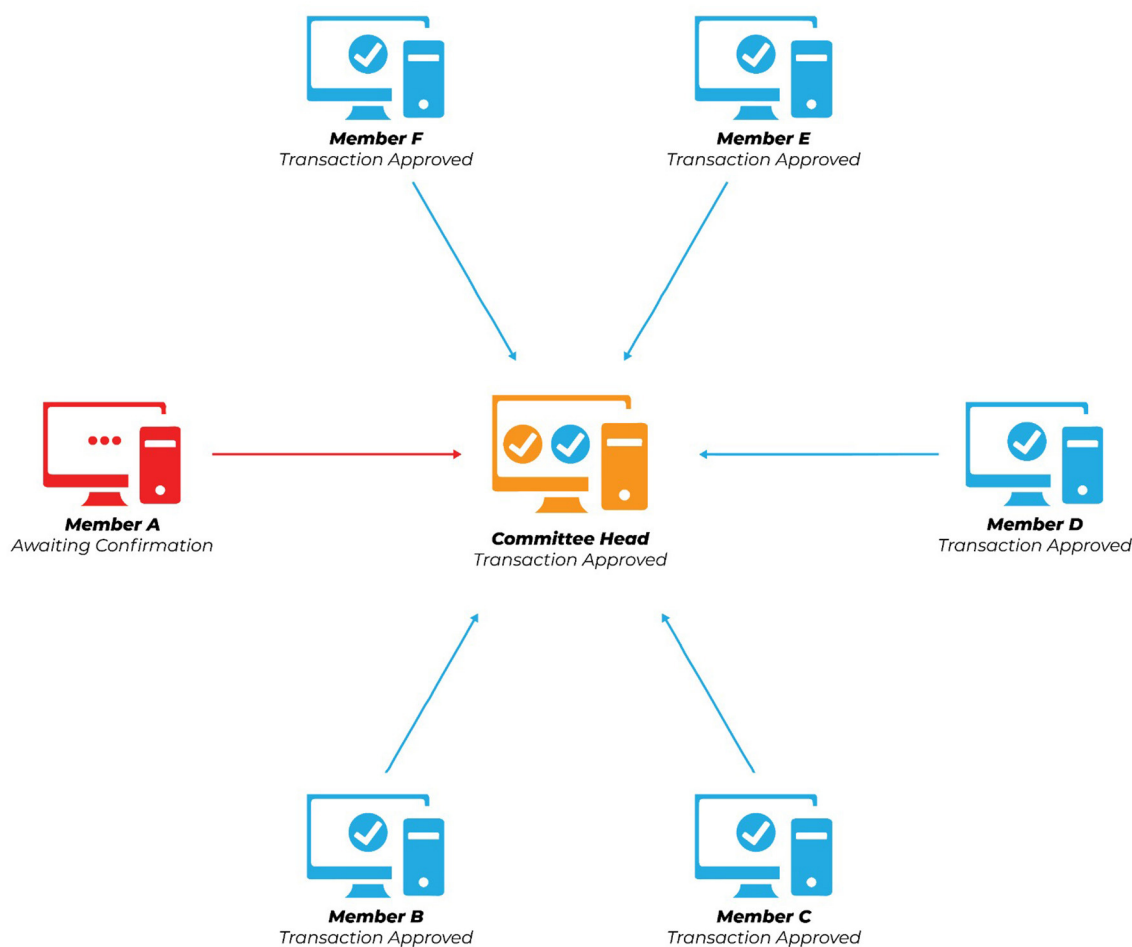


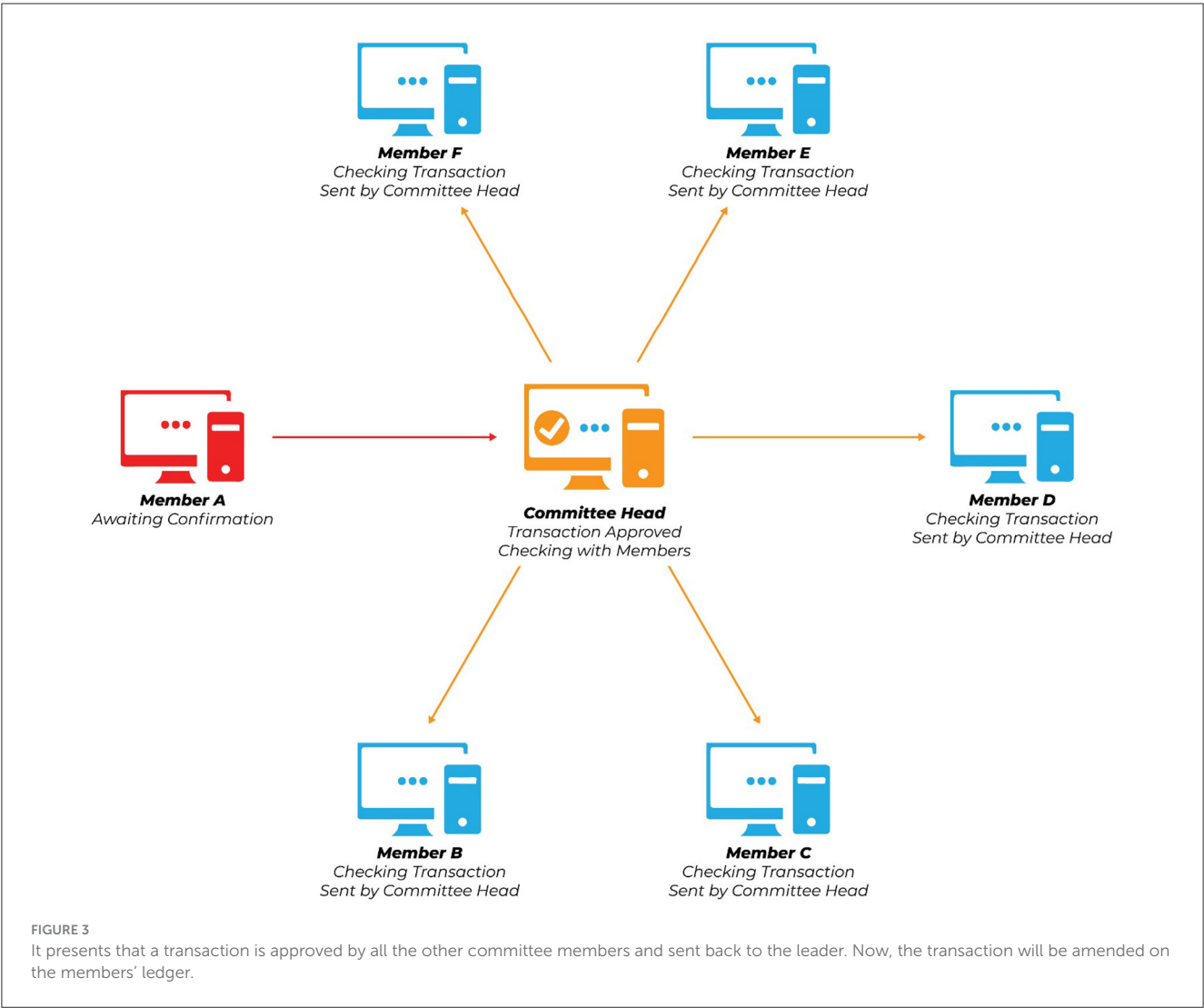
FIGURE 2

It reflects that a committee member initiates a transaction but is later approved by the leader by checking its ledger. Then the leader is forwarding that transaction to other members to check and report back.

A program known as “RAFT” was developed to achieve consensus, as described in Ongaro and Ousterhout (2014). The RAFT consensus algorithm functions by selecting a leader responsible for overseeing log replication. This leader’s duties encompass receiving log entries from nodes, replicating them across other servers, and notifying the servers when it’s safe to update their log entries to reflect the new data. If a leader experiences failure or disconnection, a replacement is promptly selected. This concept drew inspiration from Paxos, an effective but complex consensus algorithm. To better comprehend the differences between Paxos and Raft, a brief comparison has been included (Erdin et al., 2021). Raft streamlines the essential elements of network consensus, simplifying leader election, facilitating easy implementation of log replication protocols, and enhancing safety measures. This simplicity has been demonstrated through a study involving students, highlighting Raft’s superior ease of comprehension compared to Paxos. Raft was initially designed to aid newcomers, especially students, in grasping the fundamental challenge of achieving consensus in peer-to-peer networks. Moreover, Raft incorporates a mechanism to prevent the alteration of cluster ownership,

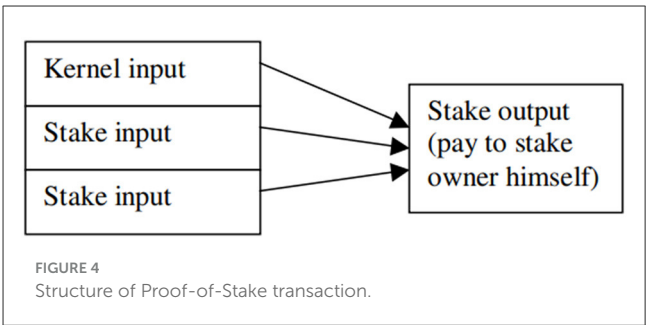
requiring majority agreement for enhanced security, as illustrated in Figure 5.

The introduction of a “technology-based democracy,” referred to as Delegated Proof of Stake (DPoS), has had a significant impact on Blockchain technology. DPoS safeguards Blockchain through a voting and election process, where active users cast their votes for “witnesses” and “delegates.” Witnesses play a crucial role in creating and validating blocks within the Blockchain. Notably, transaction fees are awarded to the top-tier witnesses. In the event that a witness misses a block due to node downtime or other reasons, the transaction is automatically redirected to the most recently active witness. Importantly, witnesses cannot manipulate transaction data in a DPoS-based Blockchain. To participate in a DPoS Blockchain, witnesses must maintain a reliable server with a 100% availability rate, as the system generates substantial network communication (Xie et al., 2019). The study also introduces an efficient consensus method known as DDPoS, designed to reduce resource consumption, enhance operational efficiency, and bolster security. This method addresses the challenge of “witness malevolent behavior” by merging the strengths of Proof of Work (PoW) with DPoS, improving the original DPoS



algorithm. Rather than relying on stakes, DDPoS employs the Proof of Work algorithm to determine nodes' processing capacity for participation in the election. Each node is limited to one vote, promoting fairness and decentralization in block production to prevent collusion attacks. Additionally, DDPoS implements a rapid node-downgrading mechanism to maintain system integrity and security. Performance evaluations indicate improvements in efficiency and security compared to conventional consensus methods (Philippopoulos et al., 2019).

Despite these advancements, certain constraints must be considered. The experimental environment does not replicate real-world scenarios, making immediate integration into existing business environments like Hyperledger or Ethereum challenging. For optimal results, the proposed consensus method necessitates validation within a production-like Blockchain setting, bridging the gap between experimentation and real-world applicability. These considerations are vital as we delve further into the study's details, comparing three protocols in Table 1. Similar ideas are extensively used in various research work focused on Pharmaceutical Supply Chain. Bandhu et al. (2023) worked on Blockchain and smart contract based implementation for making



drug supply chain secure, traceable and efficient. Rai et al. (2023) proposed Blockchain-Based traceability of counterfeit drugs and gave appreciable results on various parameters. Similar effort by Abdallah et al. on Blockchain-based solution for pharmaceutical supply chain industry. Their proposed solution is based on utilizing Ethereum smart contracts. This is to monitor the interaction between participants, trigger events that are logged to help the

participants to keep track and be informed about sale transactions, and ensure payment dispersal securely.

3 Materials and methods

3.1 Proposed solution—Overview

In our pursuit of addressing the Blockchain scalability challenge, we have explored various solutions during our literature review. While the solutions discussed here are among the best known, it's important to note that our proposed solutions may differ. One promising avenue of research has focused on enhancing consensus protocols, which play a pivotal role in the operation of Blockchain networks. Bitcoin, a renowned Blockchain network, currently employs the Proof-of-Work (PoW) consensus protocol (Yang et al., 2019). While PoW is recognized for its robust security, it is criticized for its sluggish transaction processing speed. Consequently, many Blockchain networks are exploring the adoption of the Proof-of-Stake (PoS) consensus mechanism as a potential solution to the scalability issue. PoS eliminates the need for miners to solve resource-intensive cryptographic algorithms, thereby conserving computational power. Instead, it ensures consensus reliability by selecting validators based on their stakes in the network. PoS holds the promise of delivering improved security, decentralization, and scalability (Yu et al., 2020).

Sharding, another compelling option, addresses the Blockchain scalability challenge by breaking down transactions into smaller data sets known as “shards” (Cai et al., 2022). These shards can be processed simultaneously by the network, allowing for

parallel transaction handling. Sharding facilitates data distribution among different nodes while maintaining data integrity. Shards serve as proofs of the main chain and interact with each other to share addresses, general state information, and common communication protocols (Ali et al., 2022). The concept of a nested Blockchain offers yet another innovative approach to scalability. This decentralized network infrastructure leverages the main Blockchain to establish a more extensive network of secondary chains, enabling transaction execution across interconnected chains. Nested Blockchains are considered a promising solution, particularly among layer-2 solutions (Philippopoulos et al., 2019).

Currently, all Blockchain networks operate on a peer-to-peer basis with a flat, egalitarian topology where nodes are considered equal and no central authority exists. This design promotes trustlessness, meaning participants need not trust one another but rely on the system. However, this structure can lead to slow transaction processing, especially as the number of nodes increases. Transaction fees further complicate matters, as some users can expedite their transactions by offering higher fees, while others experience significant delays (Liu et al., 2020). As we delve into our research, we aim to build upon these existing insights to contribute to the ongoing efforts to address Blockchain scalability effectively. When we discuss permissionless Blockchains, we're referring to open Blockchains where anyone can participate as a node and maintain anonymity if desired. In such Blockchains, nodes can modify previous transaction data before incorporating them into a new block, potentially leading to a “fork” in the chain. A fundamental goal of consensus algorithms is to prevent these forks, ensuring users reach a unanimous version of truth and maintain trust in the Blockchain. Even in permissioned Blockchains, where known and trusted nodes participate, consensus remains crucial. These Blockchains operate on a “trust but verify” principle, implying that transactions must be valid and trusted by all nodes to avoid the cascade of errors triggered by a single fraudulent transaction. This issue is akin to the Byzantine Generals' Problem when a node becomes unreliable or corrupted and the Two Army Problem when communication breaks down. Therefore, fault tolerance is a critical aspect of consensus methods.

Bitcoin, the most renowned application of Blockchain, employs the Proof of Work (PoW) consensus algorithm. In PoW, nodes must perform resource-intensive calculations to determine if a newly generated block has a valid hash before adding it to the chain. This block is then verified by all other nodes. PoW ensures that the winning node, often the one exerting the most

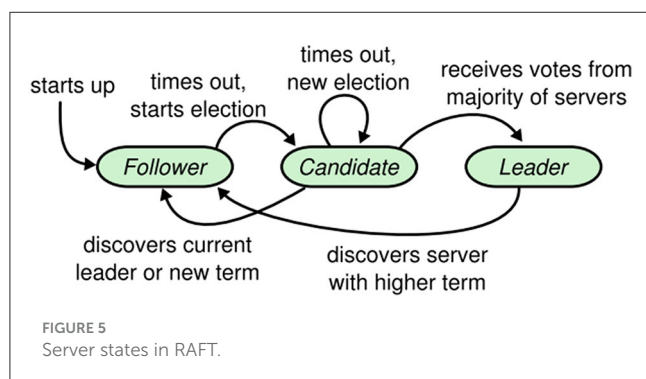


TABLE 1 Performance comparison of POW, POS, and DPoS (Yang et al., 2019).

	PoW	PoS	DPoS
The basis for assigning accounting rights	Computing power	Stakes	Stake votes
Threat to security	Concentration of power	Lack of active nodes	Destruction of the witnesses
Resource consumption	The highest	Lower than PoW, higher than DPoS	The lowest
Average time to generate blocks	10 min	64 s	3 s
Typical application	Bitcoin, Ethereum	Peercoin	Bitshare
Fairness	Relatively fair	Relatively unfair	Relatively unfair
Scalability	Good	Good	Good

effort, is awarded bitcoins as a prize, thus achieving consensus. If two blocks are mined simultaneously, a fork may occur, but PoW intentionally slows down the mining process to prevent excessive forks. Blockchain security faces various threats, including DoS attacks that flood the network with transactions, potentially disrupting legitimate transactions. More severe forms include distributed denial of service attacks. In a 51% attack, if an adversary controls over half of the network nodes, they can manipulate consensus decisions, insert transactions with altered hashes, and create the longest fork. Even with <51% control, certain attacks can succeed approximately half of the time. Double-Spend attacks are relevant in cryptocurrency contexts, involving the simultaneous use of the same currency for multiple transactions. Sybil Attacks occur when a node pretends to be multiple identities to deceive other nodes. Finally, Cryptographic Attacks with immense processing power can disrupt the balance of power among nodes with regular computing capabilities.

3.2 How proposed solution works?

The proposed system will operate as follows:

Firstly, users will be grouped into committees, each led by a committee leader. Transaction validation will involve all committee members, who will notify their leader once validation is complete. The leader will then conduct a final validation before forwarding the transaction.

Secondly, a new Linux distribution, inspired by Ubuntu, will be created. This distribution will embed the algorithm and program to run continuously in the background. The kernel scheduler within this distribution will be customized to prevent any process or tasks from running on a separate core. The recommended algorithm will run as the sole task in this core, and parallel processing will be employed to prevent conflicts.

Thirdly, the algorithm will be implemented using real-time programming, assigning priorities to all program tasks. When choosing a leader, the system will prioritize this task before permitting new transactions within the committee.

Finally, the system will utilize a device's real-time location or information from an ISP to establish a committee communication tree. Once one committee validates a transaction, it will transmit the transaction data to geographically neighboring committees.

This system will address two key issues:

- *Kernel scheduler latency*: By ensuring a dedicated core for transactions and implementing multi-threading with core isolation, latency will be reduced.
- *Workload reduction*: The committee system will prevent fake transactions from overwhelming the network and consuming resources.

Figure 6 depicts isolating the linux kernel and CPU cores for the application that is the core aspect of our proposed mechanism for enhanced computational scalability in Blockchain. As per the availability of the committee leader, the transaction processing will be started. The committee leader validates or invalidates the incoming transaction and forward it to other nodes for the further

decisions accordingly. Once the transaction valid flag is popped up from the other nodes, the ledger is updated at the ends of committee leader and others as well. After updating, the transaction is finally make available to the network for further processing. This cycle continues for every incoming transaction. Same is presented in Algorithm 1.

```

1 Check if committee leader == available
2 if committee leader == available then start
  processing transactions
3 if incoming transaction == valid, send to other
  nodes for validation
4 if incoming transaction != valid, send to others
  for rejection
5 if all nodes == transaction valid flag,
  committee leader amend its ledger and tell
  others to amend theirs too.
6 if transaction == valid from all nodes, the
  leader sends the transaction on the network.
7 if a transaction arrives on the network, the
  leader receives it, and validates it; if the
  received transaction == valid, repeat steps 2 to
  6.

```

Algorithm 1. Algorithm to improve scalability.

3.2.1 Leader selection algorithm

The initial application step involves leader availability verification for the node. If the leader is unavailable, the application proceeds to leader selection. The sole criterion for leader selection is superior hardware specifications. For instance, in a comparison between two computers with processors clocked at 3.0 and 3.1 GHz, the 3.1 GHz computer is designated as the new leader. This criterion is based on the additional responsibilities of a leader, necessitating slightly superior hardware capabilities compared to other nodes. If all nodes possess equivalent hardware specifications, the selection may be based on factors such as memory, storage, and Ethernet speed. Client application is only being started, if there is the committee leader available. In other case, the node with the highest specification in the network is selected as the committee leader. Algorithm 2 brief it in step by step process.

3.2.2 Client transaction algorithm

The client transaction initiates a continuous input loop from the client side. Upon the occurrence of a transaction at the client, it autonomously validates the transaction by applying predefined rules and regulations, as well as conducting format checks. Subsequently, the client cross-references the transaction with its internal transaction ledger to determine its validity. If the transaction is deemed valid, either through internal validation or the client's logical assessment, it is then transmitted to the leader of the respective committee. In cases where the internal ledger cannot validate the transaction, the node itself rejects the transaction and

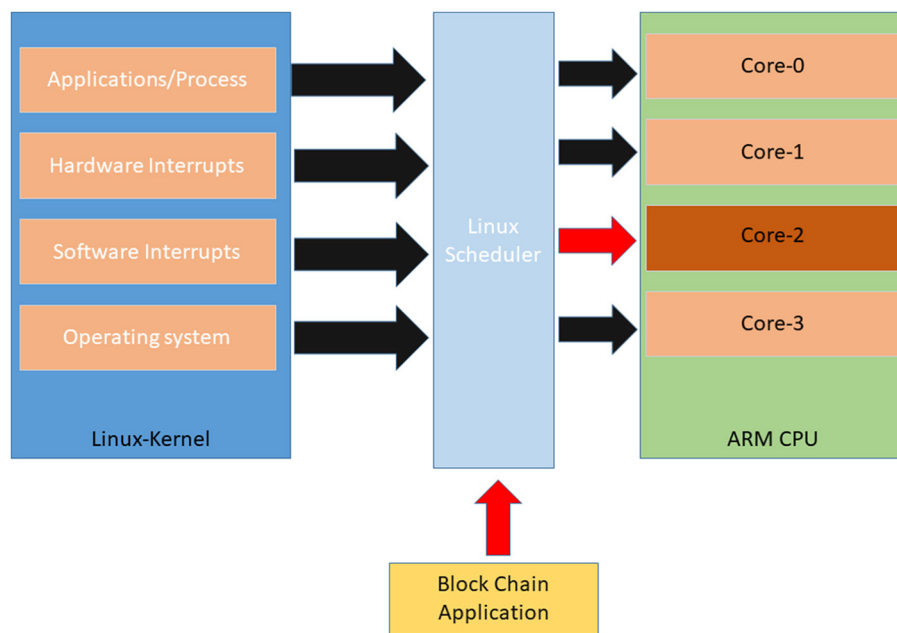


FIGURE 6
Linux Kernel and core isolation of the CPU for the application.

```

1 Check if committee leader == available, start
  the client application
2 if committee leader != available, then choose
  the leader.
3 check all the specifications of all the network
  in the system
4 compare all the specifications of all the
  system.
5 select the system with higher specifications as
  a leader and start the client.
6 In case, multiple nodes have equivalent high-end
  specifications, randomly select the leader among
  them. check if leader == available.

```

Algorithm 2. Leader selection algorithm.

notifies the user of the reason. In both scenarios, the client reenters the transaction input loop, continuously searching for input data or transactions. In short, Client has the role to finally validate the transaction and to share the decision with the leader. Once it is validated with internal ledger, its confirmation and the transaction ID is sent to the leader. Same is elaborated in Algorithm 3.

3.2.3 Algorithm for leader verification and updating

The leader verification and update algorithm exclusively operate in conjunction with the appointed committee leader. Upon the selection of a leader, the leader assumes the responsibility of processing each transaction received by nodes within the

```

1 starts the client and loop for incoming
  transaction
2 check the transaction with the internal ledger
3 if incoming transaction == valid, store the
  valid transaction in the ledger.
4 if incoming transaction != valid, discard the
  transaction
5 send the updated decision along with the
  transaction ID to the leader.

```

Algorithm 3. Client transaction algorithm.

committee. Subsequently, the leader meticulously assesses each transaction within its ledger, meticulously validating adherence to rules and regulations governing transaction formatting. Validated transactions are disseminated to all other nodes for ledger updates, thereby ensuring a synchronized record. Transactions undergo a two-tier verification process. Initially, the node client conducts an internal self-assessment. Subsequently, the leader conducts an independent verification. To further bolster transaction integrity, a redundant verification step is in place—receiving nodes autonomously cross-verify transactions against their record ledgers. If discrepancies arise during this cross-verification, the receiving node promptly notifies the leader. This comprehensive validation mechanism optimizes transaction security while conserving network bandwidth resources. Crux of its working is given in Algorithm 4.

```

1 If received transaction from the node clients ==
  validated from all nodes then amend the leader
  ledger
2 if ledger == amended, send transaction
  information on the network.
3 if received transaction from node clients !=
  valid, reject the transaction, note the sender.

```

Algorithm 4. Algorithm for leader verification and updating.

3.3 Implementation

A system comprising five Raspberry Pi 4 computers will be established, each running a customized Linux distribution. Modifications to the Linux Kernel have been applied through patches, guaranteeing the dedicated core's exclusivity for the application, while preventing the scheduling of other processes or applications on the same core. These five system-on-chip (SOC) units are interconnected via a Gigabit Ethernet switch to facilitate rapid communication. The primary objective of this setup is to assess latency, encompassing both the Linux Kernel scheduler and network latency during transaction processing. Additionally, the algorithm incorporates a leader-finding program, allowing subcommittees to change or select their leaders in accordance with the algorithm's criteria.

3.3.1 Hardware setup

Creating a Linux distribution from the ground up constitutes a resource-intensive endeavor. It necessitates a robust hardware configuration, encompassing a high-performance CPU with a maximal number of cores. Additionally, a minimum of 8 GB of RAM, 50 GB of disk space, and high-speed internet connectivity for package downloads are imperative prerequisites. The development system employed for designing and constructing the distribution is the Lenovo Legion 5 AMD, characterized by the specifications outlined in the following paragraph. AMD Ryzen 7 4800 Processor (2.90 GHz, Max Boost up to 4.20GHz, 8 Cores, 16 Threads, 8 MB Cache). NVIDIA GeForce RTX™2060 6 GB graphics card. 16GB DDR4 3,200 MHz RAM. 2TB Samsung SSD. 50 MBPS internet with Giga Ethernet. The distribution has been tailored for compatibility with the Raspberry Pi 4 Model B, necessitating the utilization of five such devices for evaluating the distribution and the application's performance. Among these devices, one assumes the role of the committee head, while the others operate as followers or nodes. To interconnect all the devices, a D-Link gigabit Ethernet switch has been deployed, with all network devices equipped with gigabit Ethernet ports, ensuring equitable networking conditions. The established configuration is centered around five Raspberry Pi 4 Model B (2GB) units, each running a specialized Linux distribution with core isolation. Specifically, Core 2 is dedicated to executing the real-time Blockchain application, with stringent restrictions in place to prevent the scheduling of any other tasks on this core.

3.3.2 Software setup

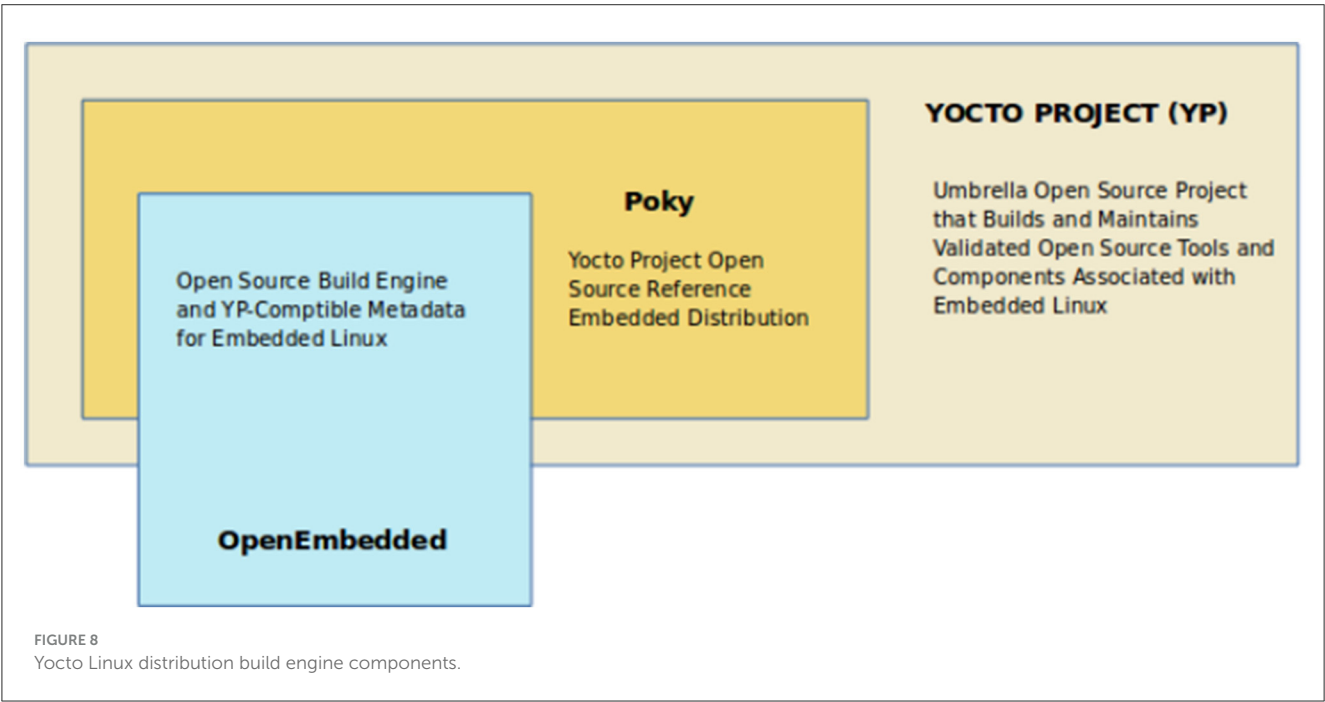
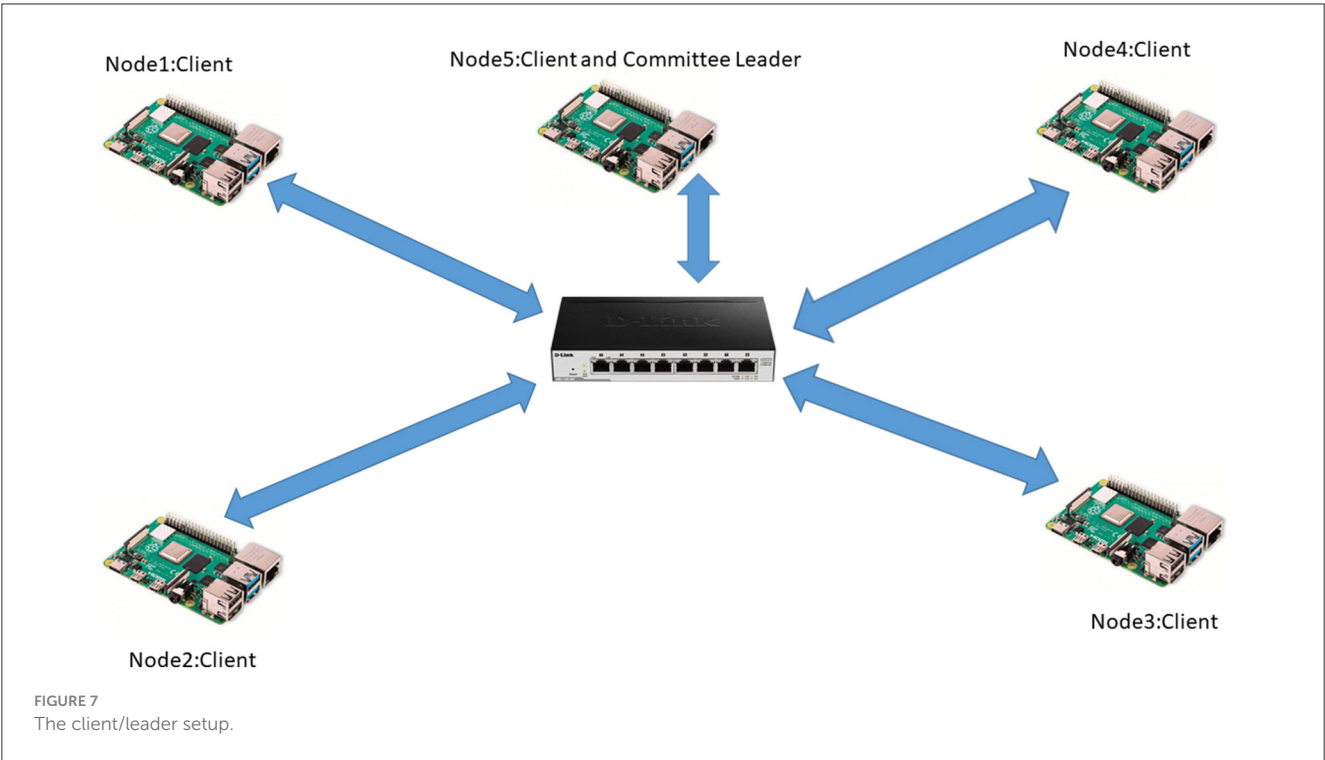
The initial steps involving the design and creation of the distribution were carried out within the Windows 10 environment, leveraging the capabilities of WSL 2 (Windows Subsystem for Linux) integration. Within this setup, Ubuntu 18.04 was selected and installed on WSL, utilizing the Hypervisor and enabling Virtual Memory support within the CPU. WSL stands as a dedicated subsystem capable of facilitating the operation of a GNU/Linux environment, encompassing a wide array of command-line tools, utilities, and applications. This subsystem operates as an autonomous realm, isolated from the host operating system. Once WSL 2 is successfully installed on a Windows PC, the utilization of software known as "Mobaxterm" becomes imperative to access WSL 2 and enable SSH and FTP data transfer capabilities. This software furnishes the essential command-line interface for interacting with the Linux environment. The client/leader setup is shown in Figure 7.

3.3.3 The Yocto Build framework

The Yocto Initiative represents a remarkable open-source community effort designed to simplify the construction of customizable Linux systems from the ground up and their deployment on embedded hardware, making the utilization of embedded Linux highly accessible. This initiative lends invaluable assistance to developers who aim to fashion tailored systems rooted in the Linux kernel. It provides a wealth of templates, tools, procedures, and extensive support for various hardware architectures, including ARM, PPC, MIPS, and x86, catering to both 32 and 64-bit systems. The Yocto Project is engineered with cross-platform tools and metadata, facilitating the expeditious creation of customized Linux distributions from source code. These custom systems, compared to conventional Linux installations, are tailored precisely to meet specific requirements. By harnessing the power of the Yocto Project, developers gain the capacity to construct and manage intricate systems and applications. Factors such as system speed, memory footprint, and memory consumption can all be fine-tuned to suit precise needs. This versatility extends across diverse hardware platforms, accommodating a vast software stack and enabling seamless software customization and construction interchange. For corporations, the Yocto Project offers the opportunity to leverage embedded engineers' expertise to craft bespoke Linux distributions, drawing upon the foundation of exceptional Linux systems. Implementing Yocto in operating system development simplifies cross-Linux Distro upgrades and cross-architecture migrations, reducing the complexities associated with these processes. The Components of the Yocto Linux Distribution Build Engine is shown in Figure 8.

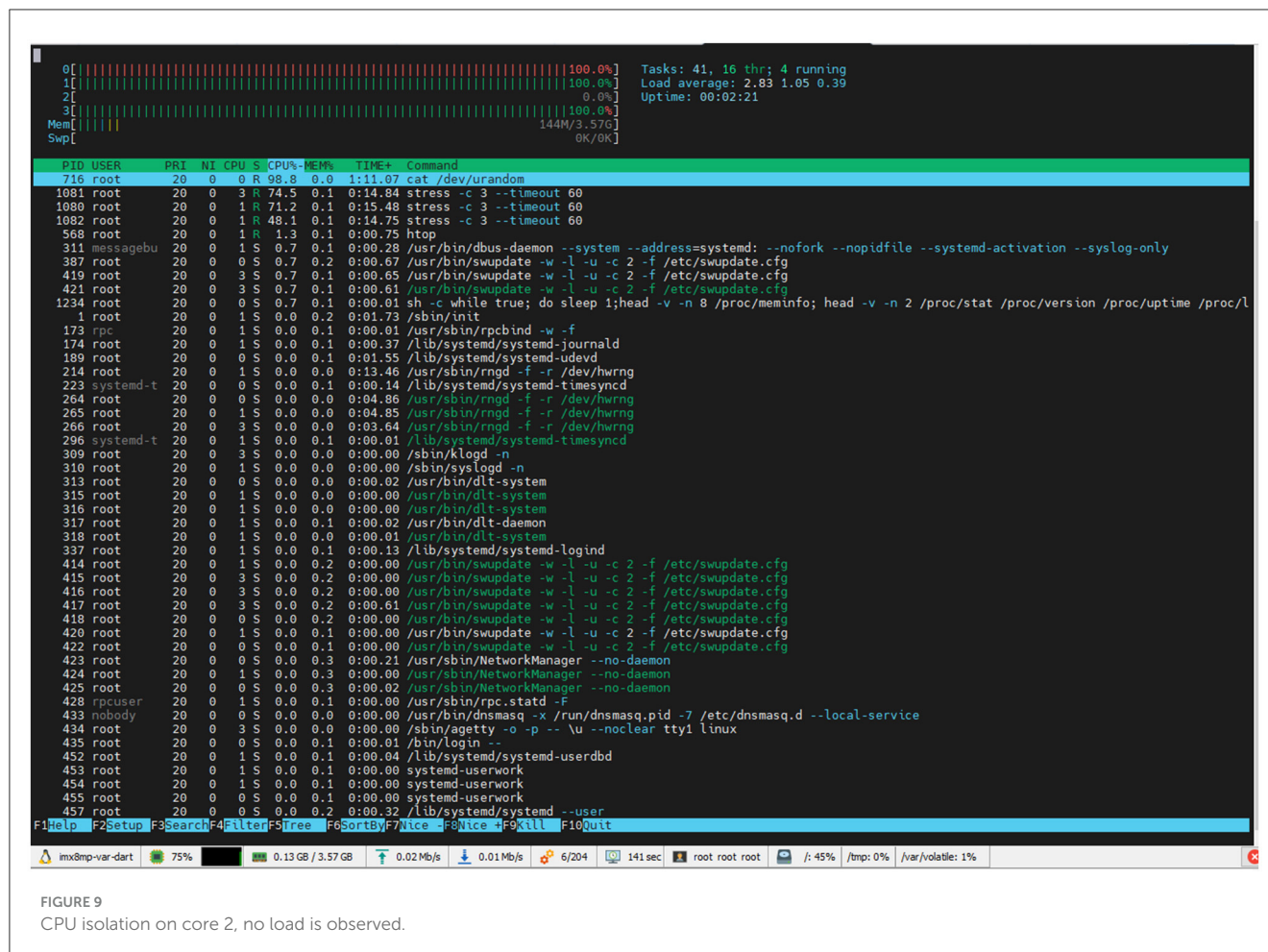
4 Results and discussion

The experimental setup consisted of a cluster of five system-on-chip (SOCs) devices, specifically Raspberry Pi 4B models equipped with 2 GB of RAM. These devices ran a tailored Linux distribution with an integrated Blockchain application that operated in the background, utilizing dedicated core isolation.



Interconnection between the five devices was established through a gigabit Ethernet switch. The application had already been seamlessly incorporated into the Linux distribution and was configured to initiate automatically at the startup of the Linux system. As depicted in Figure 9, it is evident that CPU 2 bore no computational load or scheduling, while CPUs 0, 1, and 3 were operating at full capacity. This test was conducted to apply stress to CPUs 0, 1, and 3, and assess

whether it had any discernible impact on the performance of CPU 2. A Linux Kernel scheduler, situated in the kernel space, is responsible for managing task and process scheduling across available CPU cores. This scheduling mechanism, however, often introduces delays as applications wait in a queue for their turn to be scheduled. To mitigate this performance bottleneck, modifications were made to the Kernel using the Yocto Build



system. These adjustments extended beyond the Kernel itself and also encompassed the bootloader, necessitating the editing of specific environment variables. These variables are necessary parts of the solution because these are isolating a core and only then we can be able to stick our application to the specific core, which cannot be missed. These are needed to be set only once, but they should be saved in the non-volatile memory of the computer.

• Transaction with CPU isolation and application sticking to core 2 with priority 80

A transaction is taking place on a CPU isolation (Table 2), a core, specifically, core 2 has been isolated from the Linux Kernel Scheduler. The Kernel is not going to schedule any task on the selected core. Our application will be the only application that will be running on the core. Transaction 0 is happening on Core 0, with a priority of 80, count number 69,810, with a minimum time of 2,750 ns, an actual time of 5,875 ns, an average of 4,953 ns, maximum of 48,375. This test shows that on Core 0, 69,810 transactions happened, while the minimum time that any transaction took was 2,750 ns. The actual time that a transaction took was 5,875 ns. The average time that a transaction took was 4,953 ns. The maximum time a transaction took was 48,375 ns.

• Transaction with CPU isolation and application sticking to core 2 with priority 90

Transaction 0 is happening on Core 0, with a priority of 90, count number 28,690 (Table 3), with a minimum time of 2,560

TABLE 2 Transaction with CPU isolation and application sticking to core 2 with priority 80.

T	P	C	Min	Act	Avg	Max
T0	80	69,810	2,750	5,875	4,953	48,375
T1	80	56,395	3,500	5,500	4,965	43,375
T2	80	32,524	3,625	5,125	5,277	50,750
T3	80	37,896	3,750	5,250	5,143	45,125

TABLE 3 Transaction with CPU isolation and application sticking to core 2 with priority 90.

T	P	C	Min	Act	Avg	Max
T0	90	28,690	2,560	5,474	4,563	47,365
T1	90	106,143	3,356	5,100	4,565	42,365
T2	90	31,935	3,610	5,176	4,977	49,740
T3	90	64,241	3,301	5,012	4,843	44,115

ns, an actual time of 5,474 ns, an average of 4,563 ns, maximum of 47,365. This test shows that on Core 0, 28,690 transactions happened, while the minimum time that any transaction took was 2,560 ns. The actual time that a transaction took was 5,474 ns. The average time that a transaction took was 4,563 ns. The maximum time a transaction took was 47,365 ns.

- **Transaction with CPU isolation and application sticking to core 2 with priority 99**

Transaction 0 is happening on Core 0, with a priority of 99, count number 102,123 (Table 4), with a minimum time of 2,240 ns, an actual time of 5,173 ns, an average of 4,462 ns, maximum of 43,355. This test shows that on Core 0, 102,123 transactions happened, while the minimum time that any transaction took was 2,240 ns. The actual time that a transaction took was 5,173 ns. The average time that a transaction took was 4,462 ns. The maximum time a transaction took was 43,355 ns. Due to the scheduler policy in Linux, no process of task can be set at priority more than 99 because some CPU power is needed for scheduler functioning.

- **Transaction with CPU isolation and application sticking to core 2 with priority 99 and Kernel performance mode**

Transaction 0 is happening on Core 0, with a priority of 99 and Kernel Performance Mode (Table 5), count number

17,642, with a minimum time of 1,841 ns, an actual time of 4,873 ns, and an average of 4,064 ns, maximum of 40,335. This test shows that on Core 0, 17,642 transactions happened, while the minimum time that any transaction took was 1,841 ns. The actual time that a transaction took was 4,873 ns. The average time that a transaction took was 4,064 ns. The maximum time a transaction took was 40,335 ns. Due to the scheduler policy in Linux, no process of task can be set at priority more than 99 because some CPU power is needed for scheduler functioning. The problem with Linux Kernel Performance mode is that it increases the power consumption of a device. It will continuously increase the frequency of that core. As seen from the result of four different scenarios, there is a difference in how much time a transaction takes.

4.1 Real time calculation

We will take the worst-case scenario here, meaning the Max value from results, which are between 50,000 ns to 39,000 ns. If we take 50,000 ns of worst case scenario for one transaction. There are 1,000 ms in one second, and 5,000 ns are equal to 0.05 ms. Meaning this system is capable of making 200,000 transactions per second. Keep in mind these are worst-case scenario calculations. The random results (in ms) from the application are shown in Figure 10.

4.2 Transaction comparison with other protocols

The worst-case scenario for the application is 200,000 transactions per second. Modern and latest Blockchain protocols are available in the market right now. Several key considerations come into play when analyzing our results in the context of modern Blockchains and their scalability. Despite the utilization of established Consensus methods, these Blockchains continue to make strides in enhancing their scalability. It is crucial to recognize that each method possesses its own set of advantages and disadvantages. Should we opt to retain these prior methods, we must be prepared to implement adjustments and refinements across various hardware layers. Our approach to addressing application and consensus challenges involved revisiting the fundamental elements of program execution and Blockchain operations. A comprehensive examination of Blockchain applications can be found in Abdallah et al. (2020), Jabbar et al. (2021), and Latif et al. (2021). It becomes evident that regardless of the algorithm's quality and the program's efficiency, their performance hinges on factors such as scheduling and latency within the system. In our case, testing was conducted on limited hardware resources, making it infeasible to execute a real-life program scenario, which would necessitate intricate hardware and network configurations. Consequently, numerous unanswered questions persist, particularly regarding real-time network behavior and user interactions. Comparison of TPS of different Blockchains is shown in Table 6.

TABLE 4 Transaction with CPU isolation and application sticking to core 2 with priority 99.

T	P	C	Min	Act	Avg	Max
T0	99	102,123	2,240	5,173	4,462	43,355
T1	99	100,724	3,136	5,000	4,464	40,355
T2	99	127,147	3,270	5,075	4,776	46,730
T3	99	147,587	3,101	5,011	4,745	42,105

TABLE 5 Transaction with CPU isolation and application sticking to core 2 with priority 99 and Kernel performance mode.

T	P	C	Min	Act	Avg	Max
T0	99	17,642	1,841	4,873	4,064	40,335
T1	99	115,568	2,734	4,804	4,068	37,335
T2	99	41,471	3,050	4,899	4,379	43,700
T3	99	128,898	2,801	4,791	4,349	39,175

TABLE 6 Comparison of TPS of different Blockchains (Yang et al., 2019).

Blockchain	TPS	Consensus mechanism
Ethereum	15–30 TPS	Proof of work
Cardano	250 TPS	Ouroboros proof of stake
Polkadot	1,000 TPS	Nominated proof of stake
Algorand	1,100 TPS	Pure proof of stake
Solana	50,000–85,000 TPS	Proof of stake
Bitcoin	7–10	Proof of work
Our application	200,000	Sharding method inspired by proof of stake and RAFT


```
# /dev/cpu_dma_latency set to 0us
policy: fifo: loadavg: 2.21 1.09 0.42 1/207 1217

T: 0 ( 816) P:80 I:100 C: 532368 Min:      3 Act:      3 Avg:      3 Max:      19
T: 1 ( 817) P:80 I:100 C: 532326 Min:      3 Act:      5 Avg:      3 Max:      17
T: 2 ( 818) P:80 I:100 C: 532225 Min:      3 Act:      4 Avg:      3 Max:      17
T: 3 ( 819) P:80 I:100 C: 532125 Min:      3 Act:      3 Avg:      3 Max:      30
```

FIGURE 10

Random result from the application, results in microseconds.

The development of a new Linux distribution is a complex undertaking, bearing in mind the distinct nature of Linux users compared to their Windows counterparts. Linux users typically possess a deeper understanding of hardware nuances and architectural limitations. A multitude of Linux distributions cater to specific user requirements; for instance, Ubuntu suits general users owing to its comprehensive software offerings, while Kali Linux caters to security professionals due to its extensive security tools. While integrating our application into the Linux distribution, various methods can be employed. It is worth noting that creating a distinct distribution was chosen because the process of user-managed Kernel integration is intricate and potentially risky, capable of destabilizing the operating system if executed incorrectly. Alternatively, developing a Kernel module poses challenges related to varying Kernel versions and user technical proficiency. Thus, our decision to create a dedicated Linux distribution with all requisite modifications and background applications was driven by practical considerations. Our approach, reminiscent of the political science strategy “divide and rule,” adapts this principle to the realm of Blockchain, where there is no ruling authority. Instead, we partition processes and establish committees for decision-making and consensus formation. Drawing inspiration from a variety of sources, this method showcases substantial potential for further latency-driven enhancements. Latency attributable to the operating system was a recurring impediment encountered in numerous methods implemented thus far, manifesting as delays in consensus and application processes. In addressing Blockchain scalability, it is imperative to scrutinize the foundational aspects of computation rather than exclusively addressing higher-level issues.

5 Critical analysis of research mechanism

The proposed system follows a structured sequence of operations, outlined as follows:

1. User organization into multiple committees, each led by an appointed committee leader. Within these committees, every member assumes the responsibility of validating a transaction before transmitting the information to their respective committee head. Subsequently, the committee leader performs a final validation of the transaction’s legitimacy before allowing further transmission.
2. Development of a novel Linux distribution, taking inspiration from Ubuntu, incorporating both the algorithm and program, ensuring they continuously operate in the background. The kernel scheduler configuration prevents any task or process allocation on a separate core within the distribution, with the designated algorithm being the sole task executing on that core. This parallel execution minimizes potential conflicts.
3. Implementation of the algorithm using real-time programming principles, assigning priorities to all tasks within the program. When selecting a leader, the system prioritizes this task, allowing new transactions within that committee to proceed only after leader designation.
4. Establishment of a committee communication tree, facilitated by utilizing real-time location data from devices or information provided by Internet service providers (ISPs). When one committee validates a transaction, it promptly communicates its decision regarding the transaction’s legitimacy to neighboring geographically affiliated committees.

The two step validation process makes a clear impact on the computation and the delay caused therefrom. Since, high load on one node minimizes the stability time of leader node and increases the switching frequency. This not only leads to more delay, and high computation but also decrease the throughput and consequently has its impact on the scalability. Similar approaches are adapted by [Goyal et al. \(2022\)](#), and [Aslam et al. \(2023\)](#) as well. But what makes the difference is that a custom Linux-based distribution is performed to isolate the application to prevent hardware latency and to isolate the application’s core. In addition, this system addresses two distinct challenges. First, it mitigates latency arising from kernel schedulers, ensuring that the application consistently has access to an available core for transaction processing. Even with core separation, multi-threading is necessary to further reduce latency. Second, the committee system alleviates the network’s workload, preventing spurious transactions from monopolizing network resources and impeding its efficiency. The system’s operation commences with an initial step: assessing the leader’s availability for the node. In cases where the leader is unavailable, the system promptly selects a new leader. The sole criterion for leader selection hinges on hardware specifications, specifically, the highest available hardware performance. For example, if two computers possess processors operating at different clock speeds, such as 3.0 and 3.1 GHz, the computer with the 3.1 GHz processor assumes the role of the new leader. This criterion stems from the leader’s augmented responsibilities, necessitating superior

processing power compared to other network nodes. In cases where hardware capabilities are equal, factors such as memory, storage capacity, and internet Ethernet speed may be considered as tiebreakers.

Client transactions are initiated by iteratively processing client inputs. Upon detecting a transaction, the system promptly subjects it to internal validation, verifying compliance with legal regulations and proper formatting. Subsequently, the transaction is cross-referenced with the system's internal transaction ledger to ascertain its legitimacy. If a transaction is deemed legitimate, either through internal validation or the client's reasoning, it is transmitted to the committee's leader. Transactions unverifiable by the internal ledger are rejected by the node, with the user being promptly notified of the reason for rejection. Regardless of the outcome, the client continuously cycles through the transaction input mode, actively searching for input data or additional transactions to process. The leader verification and update algorithm is exclusive to the selected committee leader, tasked with processing incoming transactions from committee nodes. Upon receiving a transaction, the leader conducts an internal ledger-based validation to ensure compliance with regulations and formatting standards. Validated transactions are broadcast to all nodes within the committee for ledger synchronization. Transactions unverified by the leader are not disseminated across the network, conserving network bandwidth. Transactions undergo dual verification processes: first by the node client and subsequently by the leader. The receiving node additionally cross-references the transaction with its internal record ledger, promptly alerting the leader if any issues arise. This secondary verification augments the primary check. The system's robustness is further demonstrated by its ability to thwart malicious transactions that lack validation from all committee members. Such transactions are prevented from entering the network, averting potential network congestion and delays. Despite its effectiveness, some limitations persist within the system. Addressing these limitations necessitates modifications at both the hardware and application levels. Additionally, the consensus algorithm must account for lower-level hardware intricacies, as these factors significantly influence system performance.

6 Conclusion, limitations, and future work

The scalability of the private Blockchain is enhanced by leveraging the improvement in consensus algorithm. The latency that is caused by the kernel scheduler is mitigated by confirming the access of application to an available core the transaction processing. The network workload is further alleviated by the committee system. This helped in avoiding from entering to the situation of spurious transactions from monopolizing network resources and impeding its efficiency. The two step validation process also made a clear impact on the computation and the delay caused therefrom. Since, high load on one node minimized the stability time of leader node and increased the switching frequency. This not only led more delay, and high computation but also decrease the throughput and consequently has its impact on the scalability. Conforming to the aforementioned scenarios ensured the Blockchain scalability that is further confirmed from the experimentation and the results

therefrom. At present, this system has certain limitations, primarily related to its compatibility. The Linux distribution developed for this system is tailored specifically for the Raspberry Pi 4 model, which is conducive to experimentation. However, it is important to note that not all Linux users possess Raspberry Pi devices. Fortunately, the Yocto Build system exhibits robust capabilities, enabling the creation of Linux distributions for a variety of systems, including those based on x86 and ARM architectures. To comprehensively evaluate the system's potential, further studies involving higher device counts and diverse hardware configurations are warranted. In particular, the inclusion of x86-based devices in the network would facilitate an assessment of their latency characteristics. The application's codebase, constructed in C with real-time POSIX libraries, constitutes a vast repository of functionalities. There exists untapped potential for optimizing its performance further. Multi-threading, while a complex subject, falls beyond the scope of our current research. Therefore, a deeper exploration and investigation into multi-threading techniques is necessary to unlock its benefits. It's important to acknowledge that not all users possess the capability or inclination to modify their operating systems. Moreover, the capacity to make Kernel-level changes is a privilege exclusive to Linux environments, precluding users of Windows operating systems from similar alterations. Consequently, the application remains limited to Linux distributions, given the impracticality of Kernel modifications in Windows.

Future prospects for this system are promising. The empirical performance comparison of this underlying work with the existing solutions is put forth in future work especially by considering the other factors affecting the scalability of Blockchain i.e. storage, communication and etc. The development of a standardized build system catering to PC, Embedded, and ARM-based devices could significantly expand its user base. Extending compatibility to PC and Windows platforms represents a viable avenue for further development. Moreover, the system can be adapted to operate without Kernel modifications, with the potential implementation of Linux Kernel modules as an alternative approach. Although Kernel options were modified in our research, such adjustments were not obligatory. Nonetheless, they served to underscore the performance and versatility of the Linux environment. Subsequent studies should explore scenarios with more devices, incorporating external network traffic into the network dynamics. By examining worst-case scenarios and quantifying maximum TPS latency, we arrived at a figure of 200,000 transactions per second. While external factors may cause TPS to fluctuate, this figure still surpasses existing market solutions. Delving deeper into hardware management at lower levels holds the promise of enhancing scalability. A comprehensive system-wide test involving a thousand or more devices could provide invaluable insights into network limitations. In lieu of creating new distributions, a Linux Kernel module could be developed, allowing for runtime insertion into the Kernel. Committee member selection could be refined based on geographical proximity and latency considerations. Furthermore, implementing a transaction validation reward system has the potential to expand the user base and increase the number of leaders available to verify transactions. These future endeavors hold the key to further enhancing the system's capabilities. The same idea could be customized to work in enhancing the scalability of Public

Blockchain environment. But the highly considerate factors in this particular scenario would be trust and security aspects.

Data availability statement

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

Author contributions

SJ: Conceptualization, Funding acquisition, Methodology, Supervision, Writing – original draft. ZA: Conceptualization, Data curation, Methodology, Writing – original draft. SK: Investigation, Project administration, Software, Writing – review & editing. AA: Writing – review & editing. UR: Resources, Visualization, Writing – original draft. SA: Project administration, Software, Validation, Writing – review & editing.

Funding

The author(s) declare financial support was received for the research, authorship, and/or publication of this article. This work

was supported and funded by the Deanship of Scientific Research at Imam Mohammad Ibn Saud Islamic University (IMSIU) grant number IMSIU-RG23157.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

The author(s) declared that they were an editorial board member of Frontiers, at the time of submission. This had no impact on the peer review process and the final decision.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

- Abdallah, M., Dobre, O. A., Ho, P.-H., Jabbar, S., Khabbaz, M. J., Rodrigues, J. J., et al. (2020). Blockchain-enabled industrial internet of things: advances, applications, and challenges. *IEEE Internet Things Mag.* 3, 16–18. doi: 10.1109/MIOT.2020.9125425
- Ali, A., Iqbal, M. M., Jabbar, S., Asghar, M. N., Raza, U., Al-Turjman, F., et al. (2022). Vablock: a blockchain-based secure communication in v2v network using icn network support technology. *Microprocess. Microsyst.* 93, 104569. doi: 10.1016/j.micpro.2022.104569
- Alshahrani, H., Islam, N., Syed, D., Sulaiman, A., Al Reshan, M. S., Rajab, K., et al. (2023). Sustainability in blockchain: a systematic literature review on scalability and power consumption issues. *Energies* 16, 1510. doi: 10.3390/en16031510
- Aslam, M., Jabbar, S., Abbas, Q., Albathan, M., Hussain, A., Raza, U., et al. (2023). Leveraging ethereum platform for development of efficient tractability system in pharmaceutical supply chain. *Systems* 11, 202. doi: 10.3390/systems11040202
- Bandhu, K. C., Litoriya, R., Lowanshi, P., Jindal, M., Chouhan, L., Jain, S., et al. (2023). Making drug supply chain secure traceable and efficient: a blockchain and smart contract based implementation. *Multimed. Tools. Appl.* 82, 23541–23568. doi: 10.1007/s11042-022-14238-4
- Bentov, I., Lee, C., Mizrahi, A., and Rosenfeld, M. (2014). Proof of activity: extending bitcoin's proof of work via proof of stake [extended abstract]. *ACM SIGMETRICS Perform. Eval. Rev.* 42, 34–37. doi: 10.1145/2695533.2695545
- Cai, Z., Liang, J., Chen, W., Hong, Z., Dai, H.-N., Zhang, J., et al. (2022). Benzene: scaling blockchain with cooperation-based sharding. *IEEE Trans. Parallel Distrib. Syst.* 34, 639–654. doi: 10.1109/TPDS.2022.3227198
- Erdin, E., Cebe, M., Akkaya, K., Bulut, E., and Uluagac, S. (2021). A scalable private bitcoin payment channel network with privacy guarantees. *J. Netw. Comput. Appl.* 180, 103021. doi: 10.1016/j.jnca.2021.103021
- Feng, L., Zhang, H., Chen, Y., and Lou, L. (2018). Scalable dynamic multi-agent practical byzantine fault-tolerant consensus in permissioned blockchain. *Appl. Sci.* 8, 1919. doi: 10.3390/app8101919
- Goyal, J., Ahmed, M., and Gopalani, D. (2022). A privacy preserving e-voting system with two-phase verification based on Ethereum blockchain. *Res. Sq.* doi: 10.21203/rs.3.rs-1729918/v1
- Guculturk, O. (2018). *Blockchain: A Trustless Network or a Technologically Disguised Shift of Trust?* Available at SSRN 3440044. doi: 10.2139/ssrn.3440044
- Harshini Poojaa, K., and Ganesh Kumar, S. (2022). "Scalability challenges and solutions in blockchain technology," in *Inventive Computation and Information Technologies: Proceedings of ICICIT 2021*, eds S. Smys, V. E. Balas, and R. Palanisamy (Cham: Springer), 595–606. doi: 10.1007/978-981-16-6723-7_44
- Jabbar, S., Lloyd, H., Hammoudeh, M., Adebisi, B., and Raza, U. (2021). Blockchain-enabled supply chain: analysis, challenges, and future directions. *Multimed. Syst.* 27, 787–806. doi: 10.1007/s00530-020-00687-0
- Latif, R. M. A., Farhan, M., Rizwan, O., Hussain, M., Jabbar, S., Khalid, S., et al. (2021). Retail level blockchain transformation for product supply chain using truffle development platform. *Clust. Comput.* 24, 1–16. doi: 10.1007/s10586-020-03165-4
- Liu, Y., Liu, J., Salles, M. A. V., Zhang, Z., Li, T., Hu, B., et al. (2022). Building blocks of sharding blockchain systems: concepts, approaches, and open problems. *Comput. Sci. Rev.* 46, 100513. doi: 10.1016/j.cosrev.2022.100513
- Liu, Y., Liu, J., Wu, Q., Yu, H., Hei, Y., Zhou, Z., et al. (2020). Sshc: a secure and scalable hybrid consensus protocol for sharding blockchains with a formal security framework. *IEEE Trans. Dependable Secure Comput.* 19, 2070–2088. doi: 10.1109/TDSC.2020.3047487
- Lone, A., Auqib, H., and Roohie, N. M. (2019). Consensus protocols as a model of trust in blockchains. *Int. J. Blockchains Cryptocurrencies* 1.1, 7–21. doi: 10.1504/IJBC.2019.101845
- Ongaro, D., and Ousterhout, J. (2014). "In search of an understandable consensus algorithm," in *2014 USENIX Annual Technical Conference (USENIX ATC 14)*, 305–319.
- Philippopoulos, P., Ricottone, A., and Oliver, C. G. (2019). Difficulty scaling in proof of work for decentralized problem solving. *arXiv [Preprint]*. doi: 10.48550/arXiv.1911.00435
- Rai, B. K., Srivastava, S., and Arora, S. (2023). Blockchain-based traceability of counterfeit drugs. *Int. J. Reliab. Qual. E-Healthc.* 12, 1–12. doi: 10.4018/IJRQEH.318129
- Schedlbauer, M., and Wagner, K. (2018). *Blockchain Beyond Digital Currencies-A Structured Literature Review on Blockchain Applications*. Available at SSRN 3298435. doi: 10.2139/ssrn.3298435
- Seo, J., Ko, D., Kim, S., and Park, S. (2020). A coordination technique for improving scalability of byzantine fault-tolerant consensus. *Appl. Sci.* 10, 7609. doi: 10.3390/app10217609
- Wang, X., Jiang, X., Liu, Y., Wang, J., and Sun, Y. (2022). Data propagation for low latency blockchain systems. *IEEE J. Sel. Areas Commun.* 40, 3631–3644. doi: 10.1109/JSAC.2022.3213330

Xie, J., Yu, F. R., Huang, T., Xie, R., Liu, J., Liu, Y., et al. (2019). A survey on the scalability of blockchain systems. *IEEE Netw.* 33, 166–173. doi: 10.1109/MNET.001.1800290

Xu, F., Bouri, E., and Cepni, O. (2022). Blockchain and crypto-exposed us companies and major cryptocurrencies: the role of jumps and co-jumps. *Fin. Res. Lett.* 50, 103201. doi: 10.1016/j.frl.2022.103201

Yang, F., Zhou, W., Wu, Q., Long, R., Xiong, N. N., Zhou, M., et al. (2019). Delegated proof of stake with downgrade: a secure and efficient blockchain consensus algorithm with downgrade mechanism. *IEEE Access* 7, 118541–118555. doi: 10.1109/ACCESS.2019.2935149

Yu, G., Wang, X., Yu, K., Ni, W., Zhang, J. A., Liu, R. P., et al. (2020). Survey: sharding in blockchains. *IEEE Access* 8, 14155–14181. doi: 10.1109/ACCESS.2020.2965147