Journal of Ambient Intelligence and Humanized Computing (2023) 14:7795–7811
https://doi.org/10.1007/s12652-023-04592-y

ORIGINAL RESEARCH

Reputation based proof of cooperation: an efficient and scalable consensus algorithm for supply chain applications

Aaliya Sarfaraz[1] · **Ripon K. Chakrabortty**[1] · **Daryl L. Essam**[1]

Received: 17 April 2022 / Accepted: 16 March 2023 / Published online: 15 April 2023

Abstract
The growing interest in blockchain technology has gained a lot of attention in Supply Chain Management (SCM) and sparked the quest for decentralized, scalable, efficient and trustworthy consensus schemes. Traditional blockchains rely on computationally expensive consensus mechanisms with low throughput and high latency. This paper conducts a performance evaluation of several existing consensus protocols to illustrate blockchain's shortcomings in terms of consensus and propose a new consensus algorithm: Reputation based proof of cooperation (RPoC). The RPoC algorithm uses a layered architecture to segment the nodes that participate in the consensus phase in order to improve scalability and efficiency while maintaining trust among peers. The layered design addresses the issues of flexibility and scalability and breaks down the extensive mining process into segments. Rather than choosing a few nodes for mining, the proposed consensus process involves all network nodes, making it more efficient, decentralized and scalable. Through extensive theoretical analysis and experimentation, the suitability of the proposed algorithm is established in terms of scalability and efficiency.

Keywords Blockchain · Supply chain · Consensus algorithm · Efficiency · Decentralization

1 Introduction

Blockchain has gained a lot of attention in the Supply Chain Management (SCM) domain recently, mostly due to the emergence of digitization and growth of the industry 4.0 context across industries. The emergence of Bitcoin Nakamoto and Bitcoin (2008) has further fueled this recognition. Over time, blockchain technology has evolved to meet a variety of applications, resulting in three types of blockchains.

- Public blockchains: Anyone can join and participate in the blockchain network. Examples include Bitcoin Nakamoto and Bitcoin (2008) and Ethereum.
- Private blockchains: Only selected transactions from authorized participants are allowed on a private blockchain, and the administrator has the authority to overrule, alter or delete any entries.
- Consortium blockchains: Instead of being governed by a single organization, the platform is governed by several organizations. An example is Hyperledger Fabric Androulaki et al. (2018).

Although cryptocurrencies have been the most well-known use of blockchain technology, several researchers have also identified the usage of blockchain and cryptocurrencies in different supply chain applications Longo et al. (2019) Sarfaraz et al. (2021b) Saberi et al. (2019). Private blockchains are ideal for supply chains Biswas et al. (2017) due to the nature of how private blockchains work. The proposed framework is based on a private blockchain solution. However, integrating blockchain technology into traditional SCM is a significant challenge, particularly with the absence of tailored consensus algorithms to tackle or embed within supply chain problems Xu et al. (2021).

The blockchain architecture validates information through a consensus mechanism among network nodes, removing the need for intermediaries. The consensus mechanism ensures a tamper-proof environment and ensures that the information stored is reliable and valid Mingxiao et al. (2017). In

✉ Aaliya Sarfaraz
a.sarfaraz@student.unsw.edu.au

Ripon K. Chakrabortty
r.chakrabortty@adfa.edu.au

Daryl L. Essam
d.essam@adfa.edu.au

[1] School of Engineering and Information Technology, University of New South Wales, Canberra, ACT, Australia

a blockchain, all nodes must agree on the current state of the ledger, making it difficult for adversaries to insert tampered blocks. Many challenges continue to affect blockchain technology, including insufficient transactions per second (TPS), transaction latency and decentralization Zheng et al. (2018). The throughput of existing blockchains is relatively low due to the complex consensus process; for instance, in a public blockchain with the proof-of-work (PoW) consensus algorithm, all nodes must perform hash calculations and are only allowed to broadcast their blocks after spending a great deal of energy for their computation Bach et al. (2018). Consequently, high consensus latency, low throughput and high energy consumption make it difficult to use existing algorithms in complex or large supply chain systems.

Considering all those shortcomings in existing consensus algorithms, a proper and customized consensus algorithm should be designed for typical SCM problems, particularly to resolve the TPS, latency and centralization issues. Most current consensus algorithm research focuses on improving mainstream consensus algorithms, even though only a few are relevant to SCM, which are highlighted in the literature review section. While the dynamic SC sector has enormous development potential, it is challenged by several other SCM issues. The Bullwhip Effect (BWE) Lee et al. (2004) is one of them and has been discussed in the literature in recent years. BWE occurs by order oscillations at each SC stage. Blockchain can mitigate BWE by providing real-time information and coordination among stakeholders. Sharing appropriate demand data throughout an SC is crucial because it may help the upstream echelons with resource and material scheduling. Furthermore, inventory requirements might be directly linked to inconsistencies between demand over time and actual demand fulfillment. In this research, we offset the traditional SCM phenomenon of BWE with BC by providing total visibility and exchanging demand data across all stakeholders. This would keep business transactions tamper-proof and available to stakeholders without the need for a centralized control body, as long as business practices and negotiated data processing contracts between firms were followed.

To address the aforementioned challenges, we first choose a few existing proofs-based (PoW, DPOS) and voting-based (PoI, PoC, Ripple, BPFT) consensus algorithms to test their performance in our proposed blockchain-based SCM framework. Based on the performance of those existing consensus algorithms, the second layer of this work proposes a reputation-based consensus mechanism by redesigning some existing approaches while complementing their strengths and eliminating some of their weaknesses. We name this proposed approach the reputation-based Proof-of-Coordination (RPoC) consensus mechanism. It reaches consensus by coordinating between two layers of nodes. The first layer consists of high authority nodes chosen based on a combination of their reputation score and verified identity. In contrast, the second layer comprises subordinate nodes selected using a random selection algorithm and grouped in clusters with a master node for each. By the performance evaluation (see section 5), each of the six existing consensus algorithms decreases blockchain efficiency by limiting blockchain throughput and increasing transaction latency. Whereas RPoC is made up of layers, each with its own set of nodes operating in parallel, thus increasing efficiency, decreasing latency and eliminating the centralization issue.

1.1 Problem motivation

While having numerous benefits, traditional blockchains are not immediately relevant in SCM. This is because they operate in a dynamic and unpredictable environment that creates millions of transactions per second Serdarasan (2013), whereas traditional blockchains have low throughput.

In a blockchain, all nodes must agree on the current state of the ledger, making it difficult for adversaries to insert tampered blocks. The consensus algorithm is the most significant component of a blockchain system as its efficiency significantly influences each blockchain's overall efficiency Mingxiao et al. (2017). Based on the diverse deployment types of blockchains, existing blockchain consensus algorithms may be divided into two categories: Proof-of-X (PoX) and Byzantine Fault Tolerant (BFT) consensus algorithms. PoX consensus algorithms, such as PoW and PoS, are appropriate for public blockchains with low efficiency and high processing power requirements. BFT consensus algorithms Yin et al. (2019); Kotla et al. (2010) necessitate significant communication resources and therefore have limited scalability. A further significant disadvantage is that the PBFT consensus algorithm's performance decreases drastically as the number of nodes in a network grows Yu et al. (2020). Furthermore, the entire consensus process is disrupted if the principal node fails. To overcome these challenges, an additional in-depth research is required. On the other hand, private blockchains are highly centralized and have fast processing speeds, making them ideal for adoption in SCM. Nevertheless, the consensus algorithms suggested in the literature are mostly intended for public crypto blockchains and so cannot be deployed for private networks, particularly SCM. In SCM, businesses can construct permissioned chains among themselves, and depending on their degree of decentralization and context, they often prefer to compromise the degree of decentralization and use algorithms with higher operating speeds and scalability. HoneyBadgerBFT Miller et al. (2016) has a greater cryptographic overhead than PBFT. Ripple Schwartz et al. (2014) requires more than 80% of nodes for transaction verification, resulting in low throughput and high latency. With the growing adoption of blockchain in the SCM domain, a number of

consensus algorithms have been developed to solve these issues. For example, Zhang et al. (2021), however, its shortcoming is that it compromises system decentralization by treating nodes differently depending on their trust scores.

To solve these problems, we propose a new scalable, decentralized consensus algorithm, known as RPoC, for permissioned blockchains that meets both performance and security requirements to boost blockchain adaption in SCM. We utilize a sharding technique and design a two-layer consensus protocol, where nodes are assigned to distinct consensus layers. Expanding the consensus groups allows both TPS and scalability to be linearly boosted while keeping the system decentralized.

1.2 Contributions

This paper provides a blockchain-enabled SCM framework to provide visibility and coordination along with blockchain consensus processes. We propose a two-layer consensus algorithm that combines a reputation and random selection algorithm appropriate for SCM. The Preprint publications of this study can be found at Sarfaraz et al. (2021a). The following are the paper's key contributions:

i An information-sharing framework is implemented based on a permissioned blockchain for a complex SC scenario. The use of BC technology in SCM is considered in terms of mitigating BWE.
ii A reputation-based consensus algorithm has been proposed by combining the advantages of existing algorithms, and throughput, scalability and latency were verified and validated for the improved algorithm.
iii The proposed algorithm is compared to the existing consensus algorithms and significantly improves TPS and scalability for SCM applications.

The rest of this article is laid out as follows. Section 2 offers a comprehensive literature review, and section 3 introduces the RPoC consensus algorithm. Section 4 presents the computational results and discussion along with different performance comparisons. In section 5, the security analysis is presented and the conclusion is in section 6.

2 Related work

Consensus algorithms are critical for improving and automating business and vendor customer logistics between various stakeholders in SCM. They do this by accelerating the delivery of manufactured products while also reducing costs.

According to the literature, most consensus algorithms are created specifically for cryptocurrency Bach et al. (2018). However, the trend is shifting and SCM is embracing blockchain for various reasons, including traceability, efficiency, security and trust. Following POW, one of the earliest consensus algorithms is Bitcoin-NG Eyal et al. (2016), a blockchain system based on the same trust paradigm as Bitcoin, but that improves latency and bandwidth over it. Delegated Proof of Stake (DPoS) Yang et al. (2019); Kang et al. (2019) is an improved and optimized version of Proof of Stake (PoS); however, there is a chance that delegated clients will be fraudulent, and there is no way to punish malicious nodes in the system. Practical Byzantine Fault Tolerance (PBFT) Li et al. (2020) is a scalable multi-layer consensus mechanism. It has been presented with hierarchically arranged nodes at different levels and restricted communication. However, ensuring data consistency among nodes requires significant communication resources. The PBFT algorithm is fast at processing transaction requests, but the overhead of communication limits its scalability Zamani et al. (2018). Following that, a great deal of work was done in improving BFT Guo et al. (2020); Crain et al. (2021). Another Byzantine-based consensus mechanism is HoneyBadgerBFT Miller et al. (2016), the first asynchronous BFT consensus system created specifically for a blockchain. However, it leads to a significant increase in communication complexity, and some financial scenarios are vulnerable to latency and scalability, demanding more in-depth analysis to resolve such situations. Moreover, HoneyBadgerBFT has a larger cryptographic overhead than PBFT. Hyperledger Fabric Androulaki et al. (2018), and Zilliqa Barrett (2017) are two projects presently employing PBFT. Yin et al. (2019) introduced HotStuff, which uses a three-phase commit mechanism to allow the protocol to establish agreement at the speed of actual network latency. Nevertheless, such techniques are difficult to scale up and suffer from trust difficulties created by botnets Dai et al. (2018).

Proof of Importance (PoI) Bozic et al. (2016) is an advanced consensus mechanism that eliminates the disadvantage of the wealthy being even wealthier. Each node's 'importance scale' decides which nodes are qualified to add a block to its blockchain. However, the concern with this is that nodes would accumulate as many coins as possible to reap the benefits of block formation. This behavior concentrates capital and reduces transaction activity. In Proof-of-Capacity (PoC) Zheng et al. (2018), a miner's storage is prioritized over hashing power. The aim of this mechanism is to reduce the amount of computing energy used. Instead of computing the hash in every block, PoC allows the list of potential solutions to be stored even before a block is mined. PoC is scalable, efficient and cost-effective, however, with the rise of cloud providers and large corporations, the mining process is becoming increasingly centralized and monopolized Sankar et al. (2017). Proof of Trust (PoT) Zou et al. (2018) calculates a node's trust based only on the total number of transactions it has completed, the number of

times it has participated in validation processes and the number of times it has received complaints from other nodes during those operations. Giving service coins to reward honest behavior and assigning a low trust value to dishonest activity are among its incentive and penalty mechanisms. However, such rewards and punishments are often excessively biased. Additionally, the procedure of PoT consensus algorithms is similar to that of classical techniques, besides the selection of trustworthy nodes. Work has been done to improve PoT Zhang et al. (2021) by classifying nodes as accounting, validating or propagating based on their trust values. However, because nodes are treated differently based on their trust scores, the system cannot lead to total decentralization.

Integrating a reputation system with blockchain has received much attention in the last few years and is still being investigated. A reputation mechanism is primarily used to facilitate delegated consensus, which reduces message complexity and resource usage by reducing the number of consensus participants Do et al. (2019). Gai et al. (2018) presented Proof-of-Reputation, a reputation-based consensus method for permissioned blockchain that relies only on reputation incentives and trustworthy registries for quick bootstrapping. Yu et al. (2019) calculated users' reputations based on their behavior and developed a reputation based consensus algorithm to reduce PoW computation costs. RepuCoin, the suggested algorithm, is still classified as PoW, which means it has all of the same shortcomings as PoW, such as power inefficiency, probabilistic consensus and low throughput Bou Abdo et al. (2021). Zhuang et al. (2019) proposed a reputation-based consensus mechanism that may be used in any setting, e.g. public or private network. Because it is a hybrid reputation/leader-based consensus algorithm, it is prone to all of the flaws of leader-based consensus algorithms, such as access fairness and denial of service attacks. A permissionless hybrid reputation/proof-of-reputation-X consensus mechanism was proposed by Bou Abdo et al. (2021). This approach substitutes the trusted identity database in proof-of-reputation-X with a new admission process to make it compatible with permissionless blockchain. However, the algorithm is centralized as a third party handles user registration.

Table 1 summarizes our discussion of the state-of-the-art consensus algorithm in the blockchain. Nevertheless, existing algorithms rely on resource-based or voting-based mechanisms, which increase communication costs by requiring several interactions. Moreover, some reputation-based algorithms achieve security and scalability while reducing fault tolerance or being semi-centralized. Even though our terminology is based on a reputation mechanism, we have significant distinctions. Accordingly, we present a consensus mechanism that preserves peer trust. Instead of picking a few nodes for mining, we propose that all network nodes participate and cooperate in the consensus mechanism, making it scalable, efficient and decentralized. Second, a high reputation score is not only a metric for consensus nodes; in order to be chosen for mining, they must put their identity at stake. Moreover, our proposed system relies on signature verification to minimize communication overhead rather than relying on a voting mechanism to verify new blocks.

3 System model and consensus scheme

This section demonstrates how we employ our Reputation-based Proof-of-coordination approach to build a supply chain architecture that minimizes the bullwhip effect. In addition, the selection of consensus nodes and the block confirmation mechanism in our RPoC are detailed.

Table 1 Comparison of state-of-the-art consensus algorithms in Blockchain

Article identifier	Mechanism	Use-case	Applicable blockchain type	Platform	Security	Scalability	Fault tolerance
Eyal et al. (2016)	Resource-based	BC network	Permissionless	Bitcoin core	Y	N	Y
Yang et al. (2019)	Voting based	BC network	Permissionless	Java	Y	Y	N
Li et al. (2020)	Voting based	IoT devices in BC	Permissioned	MATLAB	Y	Y	Y
Zhang et al. (2021)	Trust based	BC network	Consortium Blockchain	Ganache Ethereum	Y	Y	Y
Gai et al. (2018)	Reputation-based	BC network	Permissioned	Python	Y	Y	N
Yu et al. (2019)	Hybrid reputation/resource-based	BC network	Permissionless	BFT-SMaRt	Y	N	Y
Zhuang et al. (2019)	Hybrid reputation/leader-based	BC network	Permissionless/permissioned	Not specified	Y	N	Not specified
Bou Abdo et al. (2021)	Hybrid reputation/proof-of-reputation-X	BC network	Permissionless	Not specified	Y	Y	N

3.1 Blockchain based SCM framework

This section describes two models: network and threat models. The first is a blockchain-based information-sharing framework for a complex Supply Chain (SC) scenario that minimizes BWE and the second is a threat model that includes assumptions about the number and behavior of adversaries. The proposed system architecture is depicted in Fig. 1, where any stakeholder who wishes to join the network must first register with a Certificate Authority (CA) while using their original identity. After verification, the CA generates a certificate for each stakeholder. Since a CA is registered in BC, all of its operations are open to the public. A minimal trust score will be assigned to stakeholders joining the network for the first time (without a prior reputation score). Upon receiving their certificates, each stakeholder can start conducting transactions on BC. The detailed BC-coordinated SCM framework can be found in Sarfaraz et al. (2023).

In the proposed model, manufacturing and non-manufacturing stakeholders are part of a multi-tier supply network. In addition to vertical information sharing and cooperation, this method requires horizontal communication between stakeholders on the same SC tier. Suppliers and producers who use BC will collaborate by exchanging demand data and stock levels. The collaboration can be done through a permissioned BC, so only those members of the SC have access. It is easy to measure the effect of demand data because all supply chain layers share the same demand data and inventory policy. The system model and assumptions are given below:

3.1.1 Assumptions

The following assumptions are provided in the SC model.

- The retailer tracks the demand of the end consumer and places orders at the top tier (e.g., distributor or manufacturer) using the (Q, R) inventory policy.
- Any number demanded by the retailer can be generated indefinitely by the producer.
- Out-of-stock orders are not lost at any point; instead, they become backlogs that will be executed as soon as the inventory is replenished.
- The actual demand cannot be predicted ahead of time.
- Orders might be positive or negative, and cancellations are permitted.

The credentials are obtained from a CA, consisting of a set of public and private keys and a digital signature. If a situation occurs, a CA has access to individuals' identities and may disclose the true identities of the stakeholders and their relationships. The following is an overview of our blockchain-coordinated SCM framework.

i Stakeholders will be assigned a minimum reputation score after receiving keys and joining the network.

ii When an order arrives at the retailer, the stock inventory is initialized. The demand quantity is reported in the blockchain to calculate the demand deviation. The supplier then determines whether the demand quantity can be met based on the current inventory level. If the inventory is greater than or equal to the demand quantity, the demand quantity is then removed from the inventory. Alternatively, if demand exceeds supply, the order will

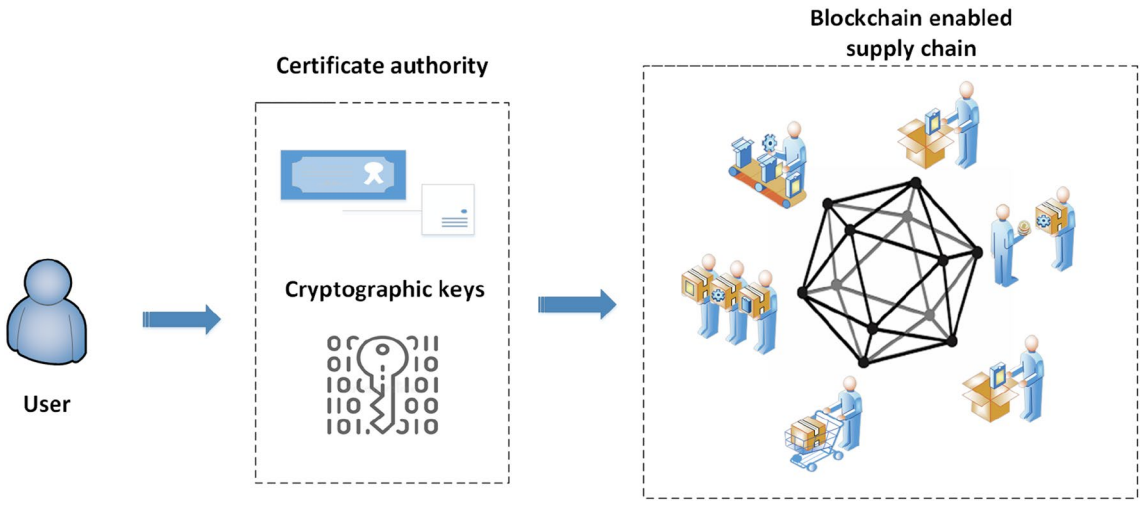

Fig. 1 Blockchain based SCM architecture

be sent to the upper echelon (e.g., distribution centre or manufacturer).

iii When the relative inventory amount is less than or equal to the reorder point (ROP), an order is placed at the next higher echelon. Request ID and order quantity are among the details sent to the next higher echelon. The next upper echelon sends back information on this order's lead time and information about the order, including order ID, date of release, lead time and order quantity, which are also documented on the BC.

iv An order is shipped when the delivery date arrives. The upper echelon would also receive information about the order ID and actual delivery date. The stock amount is adjusted, and the order is removed from the order receipt list.

v The replenishment quantity is added to the existing inventory as the lower echelon receives order delivery information from the next upper layer. If the estimated lead time differs from the order lead time, the estimated lead time is then updated.

vi The bullwhip effect (BWE) ratio is calculated by the difference between the order placed and the demand received and recorded on the blockchain.

vii After every order is received/shipped; inventory analysis is conducted to see if the relative amount of inventory is less than or equal to ROP.

viii All of the above steps are repeated when any stakeholders receive a demand quantity from their lower echelon.

This paper proposes a new consensus method, known as RPoC, for improving the throughput and scalability of a blockchain-based SCM architecture. A blockchain's consensus algorithm is at its core and significantly influences its security and efficiency. The essential features required for SCM applications are scalability, security and efficiency. RPoC utilizes a two-layer design that allows for quick consensus and scalability. By distributing the mining operations to all participating nodes, layering decreases the workload on individual nodes and increases consensus performance.

3.2 Network model

We assume the network is partially synchronous, which is the same assumption as Bitcoin Nakamoto and Bitcoin (2008). A distributed peer-to-peer network of authorized nodes communicates via the network and maintains a shared state update. The connectivity between the honest nodes is well established, and the transmission time t between them is well-defined and minimal. Once a user broadcasts a message, the rest of the honest nodes will receive the message within the specified delay Δ. Byzantine faults are also considered, as some network nodes may not be honest. The total number of nodes in the network is denoted by n, while the number of faulty nodes is denoted by f. We limit the adversary's control to a maximum of f faulty nodes where $3\mathsf{f} + 1 \leq \mathsf{n}$. Xiao et al. (2019) Xiao et al. (2020) provide detailed proofs for interested readers.

3.3 Security properties

In order to prove the security of the consensus algorithm, we must prove the safety and liveness of the algorithm. To begin with, RPoC is completely safe. Forks cannot occur as long as the number of Byzantine nodes is limited to f, even if no assumptions about network synchrony are made (i.e., there will not be a situation in which different nodes commit different blocks in the same round). The second point is that RPoC is live. RPoC achieves (eventual) liveness in a partially synchronous network, which means that new blocks are (eventually) added to the blockchain in a finite amount of time.

Given a blockchain network with a set of validators $\mathsf{V} = \{\mathsf{V}_1 \cdots \mathsf{V}_\mathsf{n}\}$, we define as pending transaction $T_x \in \mathsf{T}$ and a pending block $\mathsf{B} \in \Omega$ subject to these properties are valid:

Integrity (safety): If a T_x is confirmed to the blockchain, it has already been published by a legitimate V_n and T_x is only committed to the blockchain once, so there is no duplication.
Finality (safety): If a valid B has been appended to the blockchain at time T, it becomes definitive, and transactions within it cannot be reversed.
Validity (safety): If a valid B commits a transaction T_x in a block B, then T_x is committed, in the same block B, by every valid B.
Termination (liveness): For every transaction T_x, if a valid V_n commits T_x then all valid V eventually commit T_x.

3.3.1 Safety

Even with a slow and unstable communication network, RPoC is designed to offer safety. Once a block has been published to the blockchain by an honest node V, no other honest nodes V will ever append a different block for that round. The security of RPoC is dependent on the security of its underlying PoR protocol Gai et al. (2018).

Claim 1 (RPoC is safe): Assume that the nodes running RPoC are $\mathsf{V} = \{\mathsf{V}_1 \cdots \mathsf{V}_\mathsf{n}\}$. We take note of R_t, node $\{\mathsf{V}\}$'s reputation score, which gives it decision-making authority. Let B and B_0 be blocks appended to blockchain by honest nodes $I, j \in [n-1]$, respectively, in round k. Then $\mathsf{B} = \mathsf{B}_0$ in this case.

Proof: RPoC guarantees consensus safety If:

1. the adversary controls no more than f validators

2. or the validators compromised by the attacker have a total reputation score of R_t.

$$\mathbf{R}(t) = \frac{\sum_{i=1}^{|V|} R(\Delta_T)}{3} \tag{1}$$

Therefore, an attacker cannot violate the safety requirement unless one of the conditions is not true.

3.3.2 Liveness

Liveness is a key feature of a decentralized system that ensures that the algorithm runs correctly in time and that valid and honest transactions are eventually complete. Even if a conflict sometimes occurs, a liveness-favoring network will continue to run.

Claim 2 (RPoC is live): RPoC continues to proceed among n nodes, implying that regardless of the inner state of the nodes, some honest node will publish a new block to the blockchain within a finite time

Proof: We guarantee that all honest nodes' T_x will appear in some rounds and that all honest peers in the network will accept them. Assume that V_n has a high reputation and publishes T to the network, one of two things can happen: T will either be or not be, received by peer nodes.

1. T has been received: because of the asynchronous environment, liveness is achieved for the V_n node.
2. There has been no notification of T: This occurs when V_n is malicious or shut down during the transmission.

3.4 Encryption mechanism

Public key cryptography, such as elliptic curve cryptography, uses a public and a private key for each user. The mathematical operations of ECC are dispersed over an elliptic curve. A private key is a random number, whereas a public key is a point on the curve. By multiplying the private key in the curve by a generator point G, the public key is created. G is the starting point, also referred to as the generating point. The two parties that want to communicate information must first agree on using a curve and its parameters, such as the coefficients of a and b and the base point G to be used, before beginning the ECC process. The elliptic curve equation can be written as

$$Y^2 = X^3 + AX + B \tag{2}$$

where $4a^3 + 27b^2 \neq 0$.

Elliptic curve encryption algorithms are preferred because they demand fewer processing resources and use smaller key sizes. ECC has a reduced growth rate and time complexity of $(O\sqrt{X})$. It also has a higher resilience to attack, reduced CPU and content utilization, lower network consumption and faster encryption Sarfaraz et al. (2021b).

3.5 Threat model

The threat model describes the system's resilience to Byzantine behavior. There are two sorts of adversaries in a blockchain system developed for SCM applications. It could be external: participants may attempt to join the network or mimic an existing authorized entity. Or internal: malware or hacking can cause nodes that are correctly registered and have valid signatures to go renegade. In either case, an attacker's goal would be to get an invalid transaction approved and broadcast to the ledger Hassan et al. (2019). Any attempt to prevent a legitimate transaction or block from being recorded in a blockchain is known as a blockchain attack. We assumed that our protocol is used in permissioned blockchains, where participants can communicate in a secure environment, but that the reputation-based protocol is itself vulnerable to exploitation Gai et al. (2018). We use current public key infrastructure for key management and as a state-of-the-art secure encryption technique. A variety of threats can target the blockchain network, we consider the following attacks in our system:

Attack 1: The adversary attempts two simultaneous transactions with two different nodes in the network.

Attack 2: An attacker repeatedly engages in byzantine behavior.

Attack 3: An attacker creates numerous identities, offering network redundancy while lowering system security.

Attack 4: An attacker tries to destabilize the services of a targeted node by sending a large number of fake transactions and thence make it unavailable.

Attack 5: An attacker tries to control the network nodes to influence the consensus mechanism.

Attack 6: A malicious node pretends to be a legitimate node. It attacks the system only once its reputation score reaches a high threshold.

We assume the attacker is computationally limited, preventing it from exploiting cryptographic protocols. Furthermore, our proposed system does not consider terminal attacks or key hijacking.

3.6 Design of the proposed RPoC algorithm

The DPOS method is not decentralized, as the authority continues to be concentrated in the hands of a small group of users. For scalability, DPoS foregoes decentralization. Therefore, executing an attack is easier because fewer individuals are in charge of maintaining the network. Likewise, Ripple, PoC and PoI have decentralization issues. Therefore, they are not viable choices for SCM. On the other hand, the PBFT algorithm's consensus model only

works efficiently when the number of nodes in the distributed network is limited. PBFT does not scale efficiently because of its high communication cost that grows exponentially with each extra node in the network. The PoC protocol, from the proof-based consensus category, can be an adequate alternative for SCM because it does not require any resources or coins to invest. However, malware may have the ability to disrupt mining operations.

The algorithm has been designed considering the above evaluation. The acronyms and abbreviations used in this work are listed in Tables 2 and 3, respectively.

The proposed blockchain consensus algorithm has two steps, from the creation of a block to its confirmation: consensus node selection and transaction confirmation (block confirmation). A rigorous identity verification process must be completed before a node may join the network. If a node wishes to be a Val_a, it must confirm its true identity and agree to share it with the rest of the network. Second, the system generates the node's reputation value using a reputation algorithm and then analyses the node's credibility. Third, nodes that choose not to stake their real identification are pushed to the pool of Val_s.

As a result of the fair node selection method, the block addition procedure is optimized, and blocks can be added to the blockchain instantaneously after verification. Figure 2 depicts the algorithm's overall structure. There are two layers of N_i: Val_a and Val_s. To generate blocks, N_i are operating in parallel. Val_s is in charge of generating micro blocks and sending them to Val_a. These small blocks will be received by Val_a, who will verify them before combining them into a single block. The algorithm's fundamental feature is the ability to accurately and efficiently pick consensus nodes to work in parallel. Consensus nodes are chosen randomly from many nodes based on their reputation score, their willingness to stake their identity and the random selection algorithm that selects a subset of nodes for each cluster at random.

Table 2 Frequently used notations

Notion	Meaning
N_i	Participating nodes in blockchain
Val_a	Higher authority validators
Val_s	Subordinate validators
f_i	Faulty nodes
Tn_x	Transaction generated by Ni
Hat_n	total number of nodes in higher authority layer
Sat_n	total number of nodes in subordinate layer
At_k	Malicious nodes

Table 3 Frequently used abbreviations

Abbreviation	Definition
BC	Blockchain
SCM	Supply chain management
CA	Certification authority
BWE	Bullwhip effect
TPS	Transactions per second
PoW	Proof of work
PoA	Proof-of-authority
RPoC	Reputation-based proof-of cooperation
DPoS	Delegated proof of stake
PBFT	Practical byzantine fault tolerance
PoI	Proof of importance
PoC	Proof-of-capacity
PoT	Proof of trust
PBFT	Practical byzantine fault tolerance
ECC	Elliptic curve cryptography

3.6.1 Consensus node selection

A Blockchain network is characterized as a peer-to-peer network made up of N_i. In this algorithm, we are classifying validators into two layers: Val_a and Val_s. In order to determine node allocation into each layer, the layering setup requires the use of different methods. Therefore, to establish a consensus node selection mechanism, the algorithm combines a random number-generating approach with the node's reputation score system.

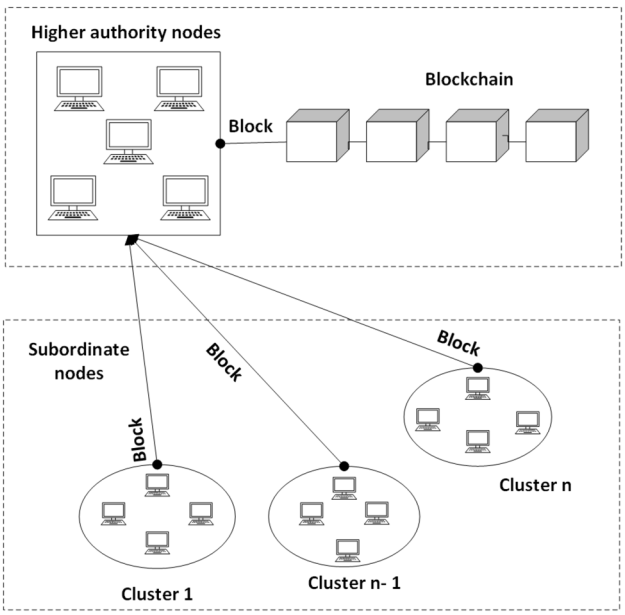

Fig. 2 Layer structure of the proposed mechanism

3.6.2 Transactions broadcasting

The stakeholder who provides a particular service during a transaction is known as the provider, whereas the stakeholder who assesses the service is known as the rater. During the transmission, the provider sends the requested service, which is signed using the provider's private key. The rater checks the data's integrity and prepares a transaction with the reputation score, which is broadcast to the rest of the network using digital signatures. The reputation score, denoted by R, can take values in the range $R_i^j \in (0, 1)$. For example, a manufacturer rates 1 to a supplier if it is satisfied with the service and 0 if it is not. In RPoC Val_a are chosen not only on the basis of their reputation (adopted from Gai et al. (2018)) but also based on their proven identity. When a stakeholder joins the network for the very first time, it has no previous reputation score. So, an initial reputation score Rep_{min} is assigned to the stakeholder, which is the minimal reputation score to continue operating in the network. $H \in (0, 1)$ represents the stakeholder's honesty, which is set to "1" for each new joiner and to "0" if a stakeholder has misbehaved. When a stakeholder is selected as a validator, it is considered to be misbehaving if it sends conflicting signed messages to other consensus group members or commits mini blocks with conflicting transactions. The stakeholder's aggregate reputation $R(\Delta_T)$ at time Δ_T is calculated by combining the stakeholder's current and past reputation score.

$$R(\Delta_T) = \sum_{t=0}^{t=i} Rep_{VAL}(T) \tag{3}$$

where $Rep_{VAL}(t_0)$ is the initial reputation score of the stakeholder and $Rep_{VAL}(t_i)$ is the current reputation score of the stakeholder, happening at Δ_T. The stakeholder's reputation can be calculated regularly, with the time determined by the system's administration. A stakeholder must stay in the system long enough and conduct themselves honestly to build a high reputation score.

We must classify nodes to perform different roles according to their reputation score. The node selection procedure is shown in Algorithm 1. The execution flow of Algorithm 1 is to select the nodes with the highest reputation scores (lines 2–8) for the higher authority layer. Along with reputation, we also use the value of identities in our algorithm, which implies that Val_a stake their real identities rather than any other resources. For Val_a, we will consider a small number of validators, so we can make a scalable system. The Hat_n layer contains N_i with higher reputation scores and verified identity. The remaining nodes are grouped into clusters using a separate random selection procedure (lines 9–14). Each sub-layer cluster has a master node, which is chosen based on its reputation score. The master node oversees validating transactions and forwards them in a small block. In the case that the primary validator goes down or becomes unresponsive, the cluster's next highest reputation score node serves as a replacement. It is important to remember those node roles are not fixed. A higher authority node's status changes as its reputation score change after its tenure. Consider a higher authority layer node, and it may become a validating or propagating node if its reputation score drops. A validating or propagating node may also become a higher authority node if its reputation score rises.

Algorithm 1 Algorithm of Selecting Consensus Nodes

```
1:  procedure ConsNodeSELT(NodesN, AuthSTD)
2:      AuthNodeLst(len(n)) ← 0
3:      ClusLst(len(n)) ← 0
4:      MasterNodesLst(len(n)) ← 0
5:      for all i, n ∈ N do
6:          AuthState ← AuthProcess(n)
7:          if (AuthState = T ∧ getRepScr(n) > Rep_min) then
8:              AuthNodesLst(i) ← n
9:          else
10:             ClusLst(i) = Random(n ∉ AuthNodeLst)
11:         end if
12:     end for
13:     for all i, nC ∈ ClusterLst do
14:         MasterNodesLst(i) = Random(nC)
15:     end for
16: end procedure
```

3.6.3 Block confirmation

The steps for block confirmation are as follows:

- A node initiates a transaction (Tn_x, Sig_c, T_s), where Tn_x is transaction, Sig_c is the client's signature and T_s indicates the timestamp.
- The cluster nodes receives and verifies Sig_c and T_s. If the verification is successful, the transaction $(Tn_x, Sig_c, T_s)_{\text{cls}}$ is forwarded to the master node in the cluster, where $_{\text{cls}}$ is the signature of the cluster node.
- The transaction must be verified by the master node. It verifies that the cluster node's signature is correct and that the transaction has not been registered in the blockchain. As soon as the verification is completed, the transaction is signed $(Tn_x, Sig_c, T_s)_{\text{cls, m}}$, where $_m$ is the master node signature.
- After signing, the transaction is pushed to the waiting pool. When there are a specific number of transactions in the pool, the master node packs each of them into a small block $(Smallblock_{TX})_{\text{m}}$ and broadcasts it to the same layer.
- After receiving $(Smallblock_{TX})_m$, other cluster nodes verify the transactions included in the block. Upon successful verification, the master node receives $CONSENT, (Smallblock_{TX})_{SL}$.
- The master node can send $(CONSENT, Smallblock_{TX}, \text{Sig}_{\text{SL}})_{\text{m}}$ to the higher authority consensus group. Where Sig_{SL} is all the signatures from subordinate nodes.
- There may still be some Tn_x left in the pool after packing a small block; these Tn_x will be verified first in the new round of consensus.
- After receiving a small block from Val_s, the nodes in the authority layer must validate the signatures and transactions of the small block.
- Once its verified, the higher authority nodes send an acknowledgment $(ACK_{accepted}, Smallblock_{TX})_{\text{auth}}$ to the subordinate nodes. In some cases, if verification fails, rejection $(ACK_{rejected}, Smallblock_{TX})_{\text{auth}}$ is sent back to subordinates.
- The small blocks are put in chronological order after the verification is successful. A large block will be packaged and added to the blockchain after receiving a minimum of 10 small blocks.

Algorithm 2 presents the module for reaching a consensus on the verification of a block. N_i in our system are equally responsible for confirming Tn_x throughout the entire blockchain and work for hand in hand to boost system throughput. By distributing the transaction verification process to every node in the network, our approach enhances consensus performance while lowering the workload on miners.

Algorithm 2 Algorithm of Reaching a Consensus

```
1:  procedure ReachCons(MasterNodesML, AuthNodeAN)
2:      SmallLst(len(SL)) ← 0
3:      AuthSmallLst(len(SL)) ← 0
4:      BlockChain ← 0
5:      Block ← 0
6:      for all i, m ∈ SL do
7:          SmallLst(i) ← generateSmallBlock(m)
8:          AuthSmallLst(i) ← SmallLst(i)
9:          if thenAuthState ← AuthProces(AuthSmallLst)
10:             AuthNodeLst(i) ← m
11:             Block ← AuthState
12:             BlockChain ← Block
13:         end if
14:     end for
15: end procedure
```

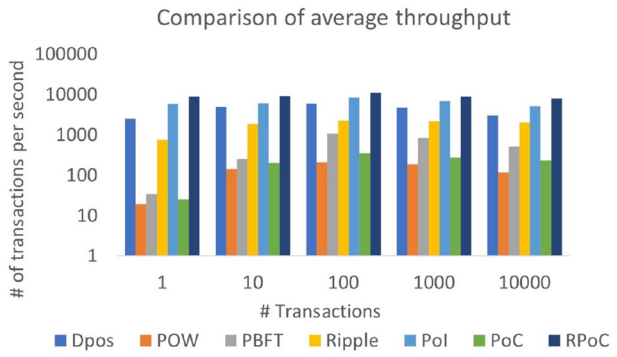

Fig. 3 Average Throughput with a varying number of transactions

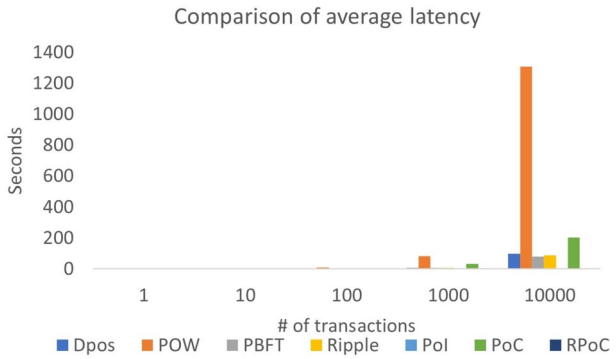

Fig. 5 Impact of the different number of transactions on latency

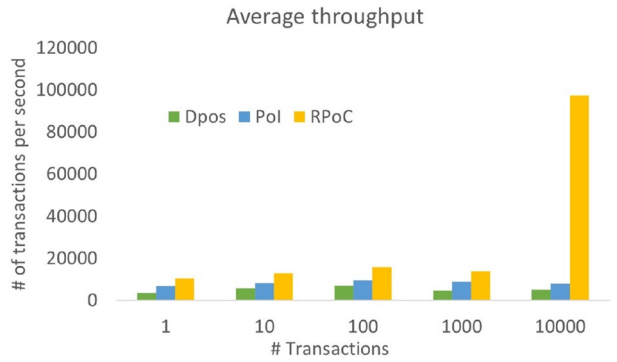

Fig. 4 Average Throughput of PoI, DPOS and RPoC with varying number of transactions

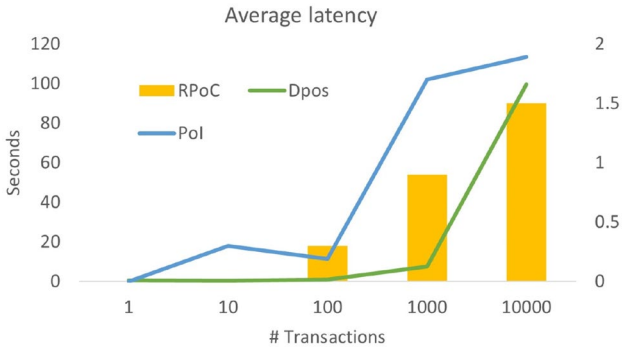

Fig. 6 Comparison of average latency among PoI, DPOS and RPoC

4 Simulation and result analysis

4.1 Simulation design

The development of the proposed framework and all the results were obtained using Python 3.9 on a Windows 10 computer with an Intel Core i7 processor running at 2.21 GHz and 16 GB of memory in Visual Studio Code. Traditional PoW, DPoS, Ripple, PBFT and PoI consensus algorithms were also simulated and their experimental results are compared with the RPoC algorithm in order to justify the experimental results. There are several options available that can be taken into consideration for implementing private and permissioned blockchains. Since high performance is a key need for our use case, we employ an x86–64 CPU system. Although it is technically possible to mimic the network using an i3 processor and 4GB of RAM, some components can be slower, but the ratio of benefits remains the same for our approach.

4.2 Result analysis

This section compares the experimental results of PoW, DPoS, Ripple, PBFT, PoI and RPoC consensus algorithms. The provided results are an average of 10 simulation runs. Throughput efficiency, latency and scalability are the three primary elements that we use to evaluate the performance of the RPoC consensus algorithm.

1. Throughput: Throughput efficiency is expressed by TPS (Transactions Per Second), which can be measured by calculating how many transactions are completed w.r.t. time. It is used to measure how much processing a blockchain network is doing and how much scalability it has.
2. Latency: This measure is used to calculate the time it takes for a transaction to go from being sent to the network to being written to the ledger. This metric is calculated by comparing the time transactions take from when they were submitted to the time they were validated and stored using their timestamps.
3. Scalability: This metric evaluates the algorithm's capacity to continue to perform properly when its size or vol-

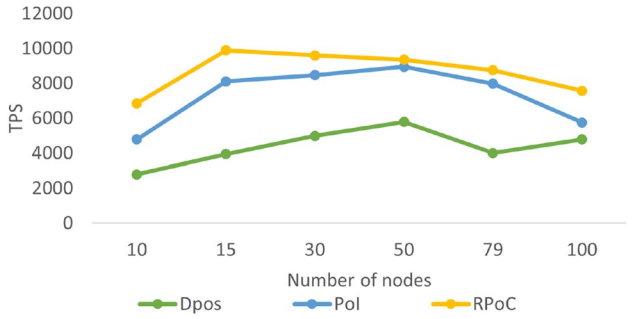

Fig. 7 Average throughput with the varying number of nodes

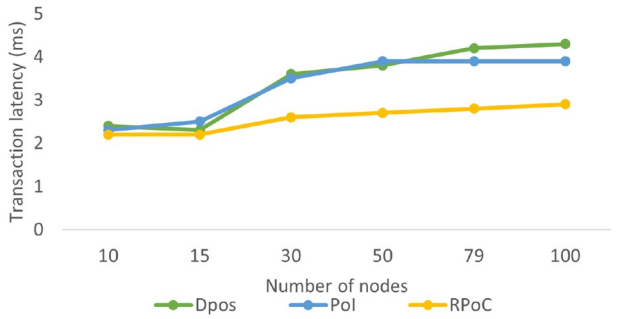

Fig. 8 Average latency with varying number of nodes among PoI, DPoS and RPoC

ume is modified. The re-scaling is usually to a bigger size or volume.

4.2.1 Throughput

For measuring a system's efficiency, throughput is an important performance indicator. Starting with the initial transaction deployment time, throughput is defined as the number of executed transactions per second, where the average throughput is the total throughput divided by the execution time. In this set of experiments, the TPS value of all algorithms was obtained and compared. The graph of the average throughput of consensus algorithms for various numbers of transactions is shown in Fig. 3. As the number of Tn_x grows from 1 to 100, the average throughput of all algorithms grows. PoI, DPOS and the RPoC algorithms have the highest throughput under 100 Tn_x, whereas PoW, PoC, PBFT and Ripple had the lowest. The average throughput of all algorithms drops after 1000 Tn_x as the number of Tn_x grows. Figure 4 presents a chart of the average throughput for DPOS, PoI and RPoC with varying numbers of Tn_x. As the number of Tn_x grows, RPoC's average throughput always exceeds that of DPOS and PoI. The PoI algorithm's TPS ranged between 6500 and 7000, while the DPOS algorithm's fluctuated between 3000 and 5000 and the RPoC algorithm's oscillated between 10400 and 95000.

4.2.2 Latency

Latency is a key metric for assessing a network's performance and determining an algorithm's delays between nodes. A system with minimal latency is advantageous since it can return transaction processing results more quickly. For instance, the block processing time frame for the PoW algorithm is around 10 min, which means that a transaction is successfully written to the blockchain after an average waiting period of 5 min Laurence (2019).

In this test, we investigate the average latency performance of all consensus algorithms with the same amount of Tn_xs, ranging from 10 to 10000. When dealing with a limited number of Tn_x, all algorithms have low latency. For instance, while there are 100 Tn_x in the system, all algorithms have a low transaction processing latency, but as the number of Tn_x grows, the latency increases, as shown in Fig. 5. In comparison to all other algorithms, PoW's average latency dramatically increased after 100 Tn_x. PoW's average latency when dealing with 10,000 Tn_x is 1307.56 s, which is 900 times higher than RPoC. The RPoC algorithm offers a more consistent transaction processing latency that does not vary significantly as the number of Tn_x grows. We compared the PoI, DOPS and RPoC algorithms in Fig. 6 to gain a clearer understanding. DPOS has a lower latency of 1.7 s at 1000 Tn_x, compared to 1.5 s at the same Tn_x for RPoC. In terms of latency performance, the DPOS and RPoC algorithms are competitive. In conclusion, the PoW and RPoC algorithms have significant latencies, whereas the DPOS and RPoC algorithms have shorter latencies.

4.2.3 Scalability

Scalability allows a system to respond dynamically depending on the latest settings. The scalability of a distributed system is determined by how the consensus mechanism allows for flexible joining and removal of nodes. The impact of increasing or decreasing the number of nodes during the operation of the consensus algorithm was investigated in the scalability test. Each system's TPS and transaction latency was investigated with various numbers of nodes. This analysis was applied to the PoI, DPOS and RPoC algorithms. Figure 7 shows the transaction processing performance of the system under varying numbers of nodes. It is obvious from the plot that the performance of the DPOS and PoI algorithms degrades with higher numbers of nodes. When there are 50, and 100 nodes in the system, the TPS values of the DPOS algorithm are around 4500 and 3500, respectively. PoI has a little better performance than DPOS. However,

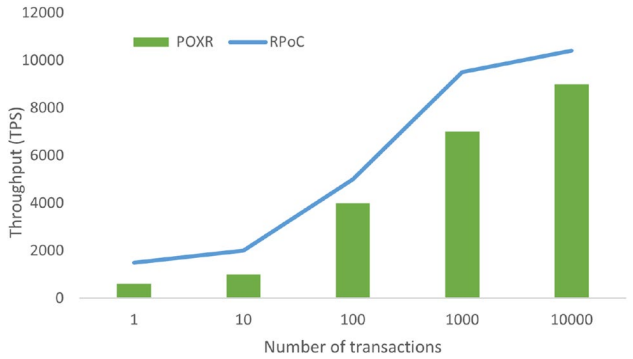

Fig. 9 Average latency comparison between PoXR and RPoC algorithms

Table 4 Attack resilience

Attacks	POXR	RPoC
Liveness	✓	✓
Selfish mining attack	✓	✓
Denial of services attack	✓	✓
Double spending attack	✓	✓
Sybil attack	✓	✓
51% attack	✓	✓

with 50 and 100 nodes, our proposed algorithm outperforms 8500 and 6900 TPS, respectively.

Figure 8 compares the average latency of the DPOS, PoI and RPoC algorithms with the same number of nodes. It is apparent that as the number of nodes increases, the average latency of each algorithm rapidly increases. The average latency for all three algorithms is identical for a set of 10 nodes. But as the number of nodes grows, the latency starts to increase for both the DOPS and PoI algorithms. On the other hand, when the number of nodes is increased from 50 to 100, the average latency of PoI and DPOS increases about 2 times faster than that of RPoC. In this set of experiments, PoI, DPOS and the proposed algorithm all perform reasonably well. The performance of the consensus algorithms is not greatly affected by the growth in the number of nodes. The DPOS algorithm was found to have low scalability, but the other two algorithms have relatively high scalability. In some cases, PoI and DPoS performance is close to RPoC. It should be noted that the PoI and DPoS protocols are lottery-based Consensus algorithms to encourage coin circulation. The two algorithms could be a perfect match for cryptocurrency use cases, where it is crucial to maintain coin circulation, instead of keeping them in a hoarded state. However, in the use case of the supply chain, trust in the network is highly desired.

4.3 Model validation

In the previous section, we compared our RPoC algorithm to the conventional consensus algorithm; however, to evaluate against proof of reputation consensus algorithms, we chose to test and validate the proposed RPoC against the Proof-of-X-Repute (PoXR) algorithm Wang et al. (2020). PoXR proposes a consensus mechanism that relies on the reputation of a system's nodes to lessen the difficulty of reaching PoX consensus in a public chain. In terms of how they work, the proposed RPoC and PoXR are polar opposites. RPoC is purely based on reputation scores, whereas POXR, like PoW, uses a mainstream protocol with a reputation layer. In PoXR, the likelihood of receiving the next honest block rises with an increase in reputation, making the process iterative. Furthermore, PoXR has issues with privacy preservation as each user protects their identity, which allows them to avoid being punished for malicious behavior.

In order to provide validation for the RPoC algorithm and an unbiased comparison, both algorithms are evaluated in the same setting (public network). We compared both models in terms of throughput and security. We compare the average throughput performance of both consensus algorithms with the same number of Tn_xs, ranging from 10 to 10000. Figure 9 shows the throughput performance comparison. Note that, unlike PoXR, RPoC does not require resources to mine a block.

Table 4 summarizes the important conclusions from the comparison of PoXR and the proposed RPoC in terms of attack resistance. In conclusion, our method works satisfactorily in terms of security and outperforms PoXR considerably in terms of throughput efficiency.

5 Security analysis

In this section, we investigate the security of RPoC against the variety of malicious attacks described in section 3.5. We assume that state-of-the-art secure encryption mechanisms are in place and that At_k will not be able to crack them. Below, we look at different types of threats:

5.1 Safety and liveness

To demonstrate the consensus algorithm's BFT characteristic, we should first prove the algorithm's safety and liveness. In RPoC, attackers cannot use their mining power to break the system; instead, they must develop a reputation and thereby contribute to the blockchain.

An attacker could never be among the top reputed miners in a network where there are trustworthy miners Val_a. For example, if the number of trustworthy miners is great enough, they all have a reputation score. However, an outside attacker who does not have a reputation score can never become a member of the consensus group. Therefore, the system's safety and liveness are always assured.

5.1.1 Double spending attack:

When At_k tries to do a second Tn_x with the same data that was already confirmed on the network, this is known as a double-spend attack. It assumes that At_k uses a double spending attack to transfer the same resource to two nodes in the network.

Defense: In RPoC, storing new blocks does not require solving a challenge or expending resources, it is predicted that a large number of validators will work in parallel. Since RPoC has two consensus layers, the network's large participating nodes will eventually recognize the double spending attack. Secondly, the blockchain's distributed nature itself prevents double spending attacks. Because all Tn_x are broadcast, validators will eventually receive blocks containing the double spend Tn_x and will be able to detect them during block verification. In this situation, At_k is removed from the validators list and node details are sent to CA, preventing them from rejoining the network.

5.1.2 Attacks in consensus groups

Assume those malicious nodes present across both layers that control the block generation and validation processes.

Defense: When the number of At_k in a cluster is less than 1/3, this consensus cluster has no effect on the generation of correct blocks. When the proportion of At_k in a subordinate cluster approaches 2/3, the At_k have the ability to package a fabricated mini-block. In this case, the higher authority nodes will create the correct large block, and then the fabricated mini-block will be recognized and excluded from large blocks, and the subordinate cluster will be eliminated from the cycle after a certain amount of time. Consider that the ratio of At_k in the higher authority layer, which is responsible for appending blocks to the blockchain, is greater than 1/3. No matter how many fake mini-blocks are received, our proposed consensus protocol will only include correct mini-blocks. This is supported by the fact that our protocol takes reputation into account when selecting block validators and creating blocks.

In addition to increasing liveness, RPoC is designed in such a way that it guarantees fairness by default, owing to its randomized validator selection process. Furthermore, RPoC distinguishes between safety, which is based on the reputation scores of the validators and liveness, which is determined by the framework.

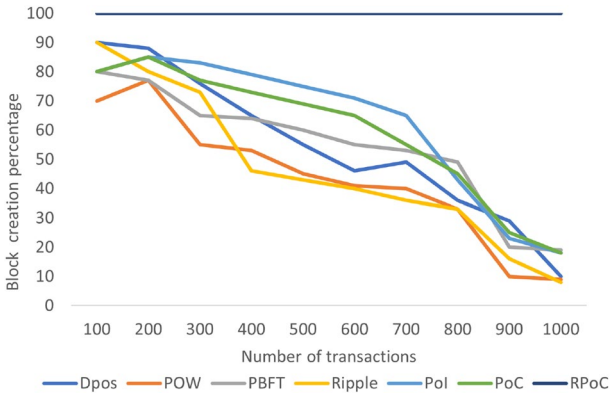

Fig. 10 Block creation with 20% Malicious nodes

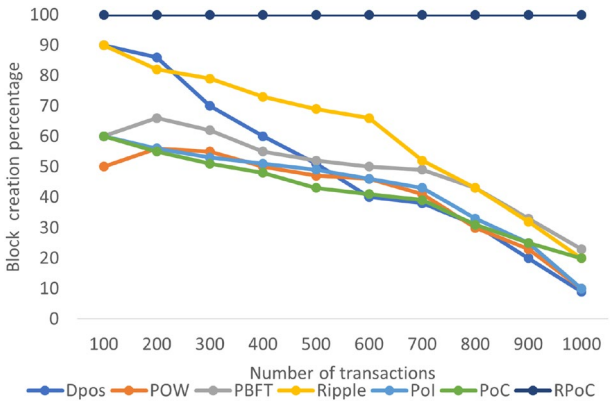

Fig. 11 Block creation with 45% Malicious nodes

5.1.3 Sybil attack:

Sybil Attack is a sort of threat in which a At_k in the network deliberately operates several identities to compromise the legitimacy of reputation systems.

Defense: As previously stated, RPoC is a two-layer consensus mechanism in which N_i work together with a CA. Every node that wishes to join the network requires a unique id issued by the CA. Furthermore, Val_a are required to provide documents in order to identify themselves, and their true identities are visible to the entire network and are at stake; if they engage in any malicious conduct and are exposed, they will be unable to rejoin the network and will lose their reputation in the business community. As a result, RPoC defends against this attack. Furthermore, let's assume that At_k has the ability to generate several accounts. However, each time the At_k starts a new account, it will be given

a low default reputation score. With a lower reputation score per account, the At_k becomes non-competitive.

5.1.4 Denial-of-service attacks:

In order to disrupt the operations of a targeted network node and make it unavailable, At_k sends a high number of Tn_x to block it.

Defense: It is feasible to protect against this attack using the RPoC mechanism: As block generation rights can only be assigned to nodes that can withstand DoS attacks since network nodes are pre-authenticated. In the case of when a validator is offline for an extended period of time, it can be removed from the validating node list. RPoC safeguards against this attack while also taking advantage of the blockchain's distributed nature.

5.1.5 Under 51% attack

The 51% attack demands that At_k gains control of 51% of nodes in the network.

Defense: Getting control of the nodes in a permissioned blockchain network is far more challenging than controlling nodes in a public blockchain network. In a permissioned setting, the adversary cannot control the majority of nodes. Hence the honest majority assumption holds.

To further evaluate our algorithm and analyze the behavior of existing protocols, we ran a set of experiments with a proportion of malicious nodes in the network. For the experiments, we have two scenarios: one with 20% malicious nodes and the other with 45% malicious nodes in the network. We did not take 51% proportion into account due to the fact that, in a permissioned ledger, malicious nodes can not control the majority of nodes, as described above. Figures 10 and 11 depict the presence of malicious nodes in the network. Existing consensus protocols focus on computational capacity, simple selection algorithms or voting for selecting a validator node without taking reputation into account. Therefore, if a malicious node is chosen as a validator, it will generate a block, solve the cryptographic puzzle and broadcast the block for validation to others. Other nodes will validate the hash values and keys of the produced block and validate the blocks, disregarding the block creator's reliability. These findings demonstrate that as the number of malicious nodes increases, all existing algorithms' resiliency declines. On the other hand, the results show that no matter how many malicious nodes are present in the network, the proposed RPoC will only publish valid blocks to the ledger. This occurs because RPoC takes reputation into account when selecting validators for both layers. Along with that, the generation of blocks in our protocol does not rely solely on a single validator. Consider the scenario where a malicious node gains a high reputation score by remaining honest for a long time and so is then able to become a master node in a subordinate layer. Further, assume that that node then generated incorrect/fake mini blocks; our higher authority layer' validators would then not allow that mini block to be included in the ledger.

Fault tolerance The capability of a design to resist the failure of one of its nodes is a part of what is referred to as fault tolerance. Since validators are in charge of storing new blocks, their failure might compromise an algorithm's fault tolerance. Multiple validators work together in our approach to append blocks at the higher authority node layer, increasing the process's fault tolerance. These validators are chosen at random based on their reputation scores and willingness to put their identities at stake, and they change over time to maintain the system's fairness. For the subordinate node layer, if any master node in a cluster fails, the algorithm chooses a high-reputation node in the same cluster to immediately resume the verification process, as mentioned in Sect. 3.6.1. As a result, a master node's failure has little influence on the transactions in that consensus cluster. This consistency, in terms of safety and liveness across the layers, leads to network reliability.

6 Conclusion

With the right consensus algorithm, blockchain technology can assist stakeholders in managing an SCM more effectively. Scalability, low latency, high throughput and decentralization are desirable characteristics of a successful consensus algorithm and directly impact a blockchain's performance. However, many existing blockchain consensus protocols are incompatible with SCMs. In this paper, a new consensus algorithm, namely Reputation based proof of cooperation (RPoC), is proposed for blockchain-based SCM that does not involve validators to solve any mathematical puzzle before storing a new block. The RPoC algorithm is an efficient and scalable consensus algorithm that dynamically selects the consensus node and permits many nodes to participate in the consensus process. The algorithm decreases the workload on individual nodes while increasing consensus performance by distributing the transaction verification process to every node. Furthermore, this paper highlights some current blockchain consensus algorithms and compares them to the proposed algorithm. Rigorous experiments against those existing consensus algorithms show the efficacy of the RPoC consensus algorithm in terms of TPS, latency and scalability.

However, the proposed methodology has the following limitations: According to the well-known "blockchain scalability trilemma," it is impossible to create consensus algorithms that simultaneously accomplish security, scalability

and decentralization. Due to the fact that we treat nodes differently based on their trust values, we cannot ensure complete decentralization. Our proposed framework also lacks detailed access control and identity management components, which are necessary to implement a practical reputation-based system effectively.

To sum up, we want to highlight that the content of this paper is not just limited to the RPoC incentive mechanism. In terms of economics, incentive design is a component that builds on the value proposition of a platform and constructs the system for which a platform's tokens will be built. In the future, we intend to work on the economics of blockchain-based SCM, to add a credit incentive mechanism to consensus nodes.

Funding Open Access funding enabled and organized by CAUL and its Member Institutions.

Declarations

Compliance with ethical standards: Mentioned authors have no conflict of interest in this article. This article does not contain any studies with human participants or animals performed by any of the authors.

References

Elli Androulaki, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo De Caro, David Enyeart, Christopher Ferris, Gennady Laventman, Yacov Manevich, et al. Hyperledger fabric: a distributed operating system for permissioned blockchains. In *Proceedings of the thirteenth EuroSys conference*, pages 1–15, 2018

Leo Maxim Bach, Branko Mihaljevic, and Mario Zagar. Comparative analysis of blockchain consensus algorithms. In *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 1545–1550. IEEE, 2018

P Barrett. Zilliqa technical whitepaper, 2017

Kamanashis Biswas, Vallipuram Muthukkumarasamy, and Wee Lum Tan. Blockchain based wine supply chain traceability system. In *Future Technologies Conference (FTC) 2017*, pages 56–62. The Science and Information Organization, 2017

Bou Abdo J, el Sibai R, Demerjian J, Jacques Bou Abdo (2021) Permissionless proof-of-reputation-x: a hybrid reputation-based consensus algorithm for permissionless blockchains. Trans Emerg Telecommun Technol 32:e4148

Nikola Bozic, Guy Pujolle, and Stefano Secci. A tutorial on blockchain and applications to secure network control-planes. In *2016 3rd Smart Cloud Networks & Systems (SCNS)*, pages 1–8. IEEE, 2016

Tyler Crain, Christopher Natoli, and Vincent Gramoli. Red belly: a secure, fair and scalable open blockchain. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 466–483. IEEE, 2021

Qianyi Dai, Kaiyong Xv, Song Guo, Leyu Dai, and Zhicheng Zhou. A private data protection scheme based on blockchain under pipeline model. In *2018 1st IEEE International Conference on Hot Information-Centric Networking (HotICN)*, pages 37–45. IEEE, 2018

Thuat Do, Thao Nguyen, and Hung Pham. Delegated proof of reputation: A novel blockchain consensus. In *Proceedings of the 2019 International Electronics Communication Conference*, pages 90–98, 2019

Ittay Eyal, Adem Efe Gencer, Emin Gün Sirer, and Robbert Van Renesse. Bitcoin-ng: A scalable blockchain protocol. In 13th {*USENIX*} symposium on networked systems design and implementation ({*NSDI*} 16), pages 45–59, 2016

Fangyu Gai, Baosheng Wang, Wenping Deng, and Wei Peng. Proof of reputation: A reputation-based consensus protocol for peer-to-peer network. In *International Conference on Database Systems for Advanced Applications*, pages 666–681. Springer, 2018

Bingyong Guo, Zhenliang Lu, Qiang Tang, Jing Xu, and Zhenfeng Zhang. Dumbo: Faster asynchronous bft protocols. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 803–818, 2020

Hassan Muneeb Ul, Rehmani Mubashir Husain, Chen Jinjun (2019) Privacy preservation in blockchain based iot systems: Integration issues, prospects, challenges, and future research directions. Futur Gener Comput Syst 97:512–529

Kang Jiawen, Xiong Zehui, Niyato Dusit, Ye Dongdong, Kim Dong In, Zhao Jun (2019) Toward secure blockchain-enabled internet of vehicles: optimizing consensus management using reputation and contract theory. IEEE Trans Vehicular Technol 68(3):2906–2920

Kotla Ramakrishna, Alvisi Lorenzo, Dahlin Mike, Clement Allen, Wong Edmund (2010) Zyzzyva: Speculative byzantine fault tolerance. ACM Transactions on Computer Systems (TOCS) 27(4):1–39

Tiana Laurence. *Introduction to Blockchain Technology: The many faces of blockchain technology in the 21st century*. Van Haren, 2019

Hau L Lee, Vineet Padmanabhan, and Seungjin Whang. Comments on information distortion in a supply chain: The bullwhip. *Management science*, 50(12_supplement): 1887–1893, 2004

Li Wenyu, Feng Chenglin, Zhang Lei, Hao Xu, Cao Bin, Imran Muhammad Ali (2020) A scalable multi-layer pbft consensus for blockchain. IEEE Transa Parallel Distributed Sys 32(5):1146–1160

Longo Francesco, Nicoletti Letizia, Padovano Antonio, d'Atri Gianfranco, Forte Marco (2019) Blockchain-enabled supply chain: an experimental study. Comput Ind Eng 136:57–69

Andrew Miller, Yu Xia, Kyle Croman, Elaine Shi, and Dawn Song. The honey badger of bft protocols. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 31–42, 2016

Du Mingxiao, Ma Xiaofeng, Zhang Zhe, Wang Xiangwei, and Chen Qijun. A review on consensus algorithm of blockchain. In *2017 IEEE international conference on systems, man, and cybernetics (SMC)*, pages 2567–2572. IEEE, 2017

Satoshi Nakamoto and A Bitcoin. A peer-to-peer electronic cash system. *Bitcoin.–URL: https://bitcoin. org/bitcoin. pdf*, 4, 2008

Saberi Sara, Kouhizadeh Mahtab, Sarkis Joseph, Shen Lejia (2019) Blockchain technology and its relationships to sustainable supply chain management. Int J Product Res 57(7):2117–2135

Lakshmi Siva Sankar, M Sindhu, and M Sethumadhavan. Survey of consensus protocols on blockchain applications. In *2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS)*, pages 1–5. IEEE, 2017

Aaliya Sarfaraz, Ripon Chakrabortty, and Daryl L Essam. Rpoc: An efficient and scalable consensus algorithm for scm applications, 2021a. URL https://doi.org/10.36227/techrxiv.16601546.v1

Sarfaraz Aaliya, Chakrabortty Ripon K, Essam Daryl L (2021) A tree structure-based improved blockchain framework for a secure online bidding system. Comput Secur 102:102147

Sarfaraz A, Chakrabortty RK, Essam DL (2023) The implications of blockchain-coordinated information sharing within a supply chain: a simulation study. Blockchain Res Appl 4(1):100110. https://doi.org/10.1016/j.bcra.2022.100110

Schwartz David, Youngs Noah, Britto Arthur et al (2014) The ripple protocol consensus algorithm. Ripple Labs Inc White Paper 5(8):151

Serdarasan Seyda (2013) A review of supply chain complexity drivers. Computers & Industrial Engineering 66(3):533–540

Wang Eric Ke, Sun RuiPei, Chen Chien-Ming, Liang Zuodong, Kumari Saru, Khan Muhammad Khurram (2020) Proof of x-repute blockchain consensus protocol for iot systems. Comput Secur 95:101871

Xiao Yang, Zhang Ning, Li Jin, Lou Wenjing, Thomas Hou Y (2019) Distributed consensus protocols and algorithms. Blockchain Distributed Syst Secur 25:40

Xiao Yang, Zhang Ning, Lou Wenjing, Thomas Hou Y (2020) A survey of distributed consensus protocols for blockchain networks. IEEE Commun Surv Tutor 22(2):1432–1465

Xiaolong Xu, Zhu Dawei, Yang Xiaoxian, Wang Shuo, Qi Lianyong, Dou Wanchun (2021) Concurrent practical byzantine fault tolerance for integration of blockchain and supply chain. ACM Trans Internet Technol (TOIT) 21(1):1–17

Yang Fan, Zhou Wei, QingQing Wu, Long Rui, Xiong Neal N, Zhou Meiqi (2019) Delegated proof of stake with downgrade: a secure and efficient blockchain consensus algorithm with downgrade mechanism. IEEE Access 7:118541–118555

Maofan Yin, Dahlia Malkhi, Michael K Reiter, Guy Golan Gueta, and Ittai Abraham. Hotstuff: Bft consensus with linearity and responsiveness. In *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing*, pages 347–356, 2019

Ge Yu, Bin Wu, and Xinxin Niu. Improved blockchain consensus mechanism based on pbft algorithm. In *2020 2nd International Conference on Advances in Computer Technology, Information Science and Communications (CTISC)*, pages 14–21. IEEE, 2020

Jiangshan Yu, Kozhaya David, Decouchant Jeremie, Esteves-Verissimo Paulo (2019) Repucoin: Your reputation is your power. IEEE Trans Comput 68(8):1225–1237

Mahdi Zamani, Mahnush Movahedi, and Mariana Raykova. Rapidchain: Scaling blockchain via full sharding. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 931–948, 2018

Peiyun Zhang, Mengchu Zhou, Qixi Zhao, Abdullah Abusorrah, and Omaimah Bamasak. A performance-optimized consensus mechanism for consortium blockchains consisting of trust-varying nodes. *IEEE Transactions on Network Science and Engineering*, 2021

Zheng Zibin, Xie Shaoan, Dai Hong-Ning, Chen Xiangping, Wang Huaimin (2018) Blockchain challenges and opportunities: a survey. Int J Web Grid Serv 14(4):352–375

Qianwei Zhuang, Yuan Liu, Lisi Chen, and Zhengpeng Ai. Proof of reputation: A reputation-based consensus protocol for blockchain based systems. In *Proceedings of the 2019 International Electronics Communication Conference*, pages 131–138, 2019

Zou Jun, Ye Bin, Lie Qu, Wang Yan, Orgun Mehmet A, Li Lei (2018) A proof-of-trust consensus protocol for enhancing accountability in crowdsourcing services. IEEE Trans Serv Comput 12(3):429–445