Int. j. inf. tecnol. (April 2023) 15(4):1845–1858
https://doi.org/10.1007/s41870-023-01214-w

ORIGINAL RESEARCH

# TISCMB: design of a highly efficient blockchain consensus model with trust integrated self-correcting miner selection

Shipra Ravi Kumar[1] · Mukta Goyal[1]

Received: 23 September 2022 / Accepted: 27 February 2023 / Published online: 1 April 2023

**Abstract** Blockchain is rapidly becoming the de facto standard for storage applications requiring high transparency, record traceability, immutable data, and distributed processing. Researchers have proposed a large number of such models, including Proof-of-Work (PoW), Proof-of-Stake (PoS), Proof-of-Authority (PoA), etc. Due to their respective limitations, each of these models is applied to context-specific blockchain deployments. In addition, the selection of the most efficient miners for hash calculation and verification is a complex task that must be executed with high efficiency in order to improve network performance. The novel contribution of this work is to design a hybrid consensus model which uses a combination of Proof-of-Work and Proof-of-Stake consensus methods for fast hash computations. Proposed model is supported by a highly efficient trust-based miner selection method that aids in choosing the most optimal miner nodes with low processing delay and high energy efficiency. In addition, this text proposes a self-correcting mechanism for the designed blockchain, which assists in the blockchain's correction in the event of any internal or external attacks. Due to these characteristics, it is observed that the proposed model has 20% less delay, 8% less energy consumption, and 15% higher levels of trust than standard PoS and PoW consensus models. The model was also subjected to various attack scenarios, and it was determined that it is capable of self-correcting the node's internal blockchain with a 99.9% success rate, thereby enhancing its real-time deployment capabilities.

**Keywords** Blockchain · Consensus · Delay · Trust · Energy · Self-correcting

✉ Shipra Ravi Kumar
shipra.chaudhary85@gmail.com

Mukta Goyal
mukta.goyal20@gmail.com

1 Department of Computer Science Engineering and Information Technology, Jaypee Institute of Information Technology, Noida, India

## 1 Introduction

Consensus protocols aid in resolving ambiguity for trivial decisions involving high-stakes transactional decisions. These decisions may be monetary, political, strategic, geographical, network-based, etc. These consensus models are utilized by blockchains to determine whether a new block can be added to the chain. Researchers have proposed a vast array of consensus models for this task. Proof-of-work (PoW) is a typical consensus model, depicted in Fig. 1 for the verification of financial transactions, with the application of cryptocurrencies depicted via peer-to-peer network validation [1].

Observable from the preceding model is that a transaction is initiated by a set of parties who wish to store a set of pre-validated data that must be usable in the event of a conflict. This information is transmitted to a network of trusted nodes (called miners), each of which verifies the transaction from each party involved. Each of these nodes, upon verification, attempts to generate a uniquely identifiable block using application-specific hash computations. A specified proportion of miner nodes must concur with the hash for it to be validated. A unique hash provided by the node with the longest chain length is selected and added to the main blockchain [2] for a PoW-based blockchain. This is because, in order to generate a block with a unique identifier, the miner node must perform the following operations:

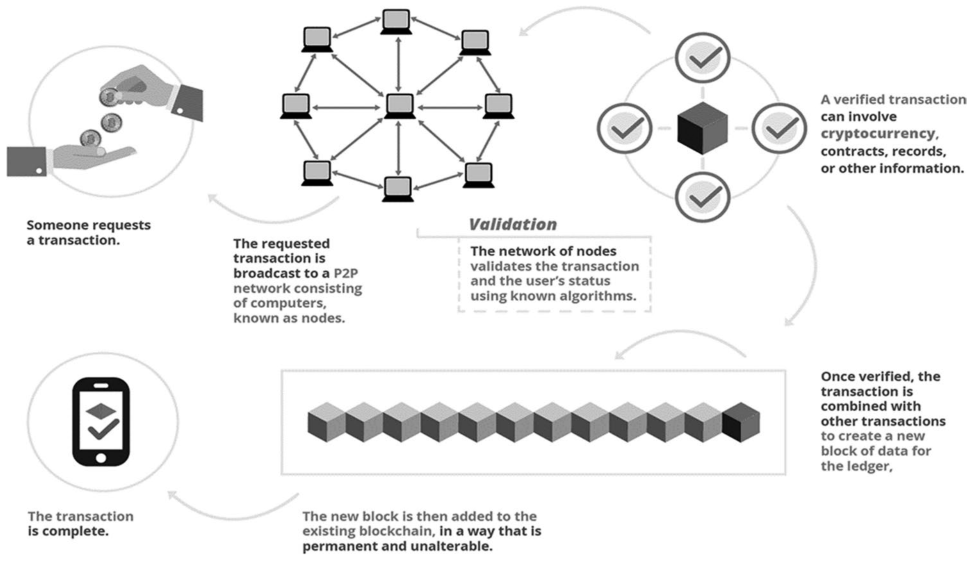
Someone requests a transaction.
The requested transaction is broadcast to a P2P network consisting of computers, known as nodes.
Validation
The network of nodes validates the transaction and the user's status using known algorithms.
A verified transaction can involve cryptocurrency, contracts, records, or other information.
Once verified, the transaction is combined with other transactions to create a new block of data for the ledger,
The new block is then added to the existing blockchain, in a way that is permanent and unalterable.
The transaction is complete.

**Fig. 1** A typical blockchain-based consensus model

- Compare the hashes of all blocks in the chain with the current hash.
- If the hash is repeated, it must be regenerated and this process must be repeated.
- If the hash is unique, add this block to the blockchain and broadcast it to the other nodes.

When designing blockchain-based consensus models, a large number of delays are encountered, including reading delay, writing delay, hashing delay, verification delay, etc. To reduce these delays, researchers proposed a vast array of consensus models, each of which varies in terms of computational complexity, speed of operation, network requirements, etc. [3]. In the following section, an analysis of these algorithms, including their limitations, benefits, subtleties, and future research opportunities, is presented. On the basis of this analysis, it has been determined that trust-based models offer superior privacy, security, and attack resistance than their non-trust counterparts. In light of this observation, the proposed hybrid model discusses the design of a novel trust-based method for miner selection. During the evaluation of trust, the computation of end-to-end delay is crucial, as nodes with shorter communication delays are preferable to those with longer delays. In addition, it was discovered that tampered blockchains can be rectified via self-correcting mechanisms, which would facilitate the reuse of compromised nodes. This reusability enhances network dependability and is thus incorporated into the proposed model designs.

The rest of this paper addressed the above issues and organized as follows: The literature review is described in Sect. 2. This is followed by Sect. 3, which describes the design of the proposed consensus model with self-correcting, trust-integrated miner selection capabilities. Various network conditions were utilized to test the proposed model, and parameters such as delay, energy consumption, and attack detection efficiency were evaluated. In Sect. 4, Result and Comparative analysis is described. These evaluations and comparisons with other reviewed models are discussed in order to estimate the scalability and adaptability of the proposed consensus model. Section 5 of this text concludes with some illuminating observations about the proposed hybrid model and also suggests ways to improve it further.

## 2 Literature review

Researchers have come up with a broad array of consensus models as potential solutions to increase the effectiveness of blockchain mining. The majority of these models are implemented into applications that are network-specific, which decreases the scalability and flexibility of such models. A blockchain network needed an authorized participant for the verification and validation of mining nodes, as well as to identify the malicious attacking nodes. The entire process of mining depends upon the consensus of the participants.

Yang et al. [4] presented Streamlined Consensus Protocols, Raft Consensus Algorithm, and Delegated Proof of Stake with Downgrade (DPoSD), each of which is used to a distinct network-specific application [5]. When applied to networks of a greater scale, these models do not perform well in terms of quality of service (QoS). Similarly, Huang et al. [6] the work in proposes a voting-based consensus model, which can be applied for securing moderate-scale applications that use both public and private blockchains [7].

Xiao et al. [8] discusses a survey of different consensus models and recommends that hybrid models are highly efficient when compared with their single consensus-based counterparts to improve this scalability. These models provide a wider hash search space to underlying consensus models, thereby improving their overall performance. Otsuki et al. [9] discusses that performance can be validated via the work in, wherein different attack scenarios are demonstrated for any underlying consensus model, and a case study of this performance evaluation on the Practical Byzantine Fault Tolerance (PBFT) model is described [10].

Wang et al. [11] performed the survey of these consensus models along with their characteristics that can also be observed in, wherein researchers have proposed the use of automatic & adaptive self-organization of miner nodes and cluster-based classification & analysis of consensus models [12]. Xiang et al. [13] discusses the application of these models, where the Proof of Previous Transactions (PoPT) based model is discussed. This model utilizes previous trust levels of miner nodes, in order to assist their selection/deselection from the network. Due to this, only the miner nodes that have high efficiency, and have showcased good temporal performance are used for consensus.

Liang et al. [14] discusses that models have good mining performance, but their security can be improved via the use of multiple hybrid encryption models, modification of hashing techniques, and improving key exchange behaviour. Pang et al. [15] discussed that such a work, where the security of blockchains is improved using multiple algorithmic augmentations, wherein Blockchain-Based Homomorphic Encryption, Multiple tokens-based Proof of Stake, and variants of proof of work (PoW) models are discussed [16].

Bhutta et al. [17] discussed that these models contribute to the enhancement of the system's overall security by including encryption, hashing, and key exchange layers into the architecture of the system. Experiments that are similar to these are reported in [18]. Hu et al. [19] demonstrated in their studies that integrated credit systems for consensus, improves both the quality of service and the efficiency of mining. The applications of these models are explored in [20, 21]. These references include discussions on the Byzantine Consensus Algorithm for information authentication, authentication of data in the internet of things (IoT) for healthcare applications, and large-scale network authentications. These apps help improve blockchain consistency for public, private, and consortium-based blockchains by reducing network overheads and computational redundancies.

Ngubo et al. [22] discussed that the consistency of blockchain can be improved in all three types of blockchains. Similar models are discussed in [23–25], where applications such as WiFi-dependent consensus, inference-based consensus, consortium blockchain-based trust-varying mining, and application of consensus to software guard extensions (SGX) are proposed. Feng et al. [26] discussed that these models are similar to the ones presented in this article. In addition, trust management has been described with the goal of determining the trustworthiness of miners, rotating several random Masters, and an error-correcting data storage system that is built on blockchain technology [27, 28].

Srividya et al. [29] discusses that mining delays are decreased even more, and the overall security performance of public, private, and consortium blockchains is improved as a result of the use of these models, which aid in identifying mining nodes based on trust-based modelling scenarios. In light of these data, it is possible to draw the conclusion that hybrid consensus models with trust-based miner selection perform better than other models with regard to security and quality of service performance [30, 31]. Motivated by this observation, the next section proposes the design of a blockchain consensus model with trust integrated self-correcting miner selection. The performance of the proposed model is evaluated in terms of mining delay, & energy requirements, and is compared with various reviewed approaches (Table 1).

## 3 Design of the proposed blockchain consensus model with trust integrated self-correcting miner selection

Based on the literature review, it is observed that researchers have proposed a wide variety of consensus models, and each of them utilizes different mining algorithms to optimize blockchain network performance. Based on these models, it can be observed that delay for block addition can be represented via reading, writing, & verification components, and can be evaluated via Eq. (1) as follows,

$$D_a = D_r * L + \left(D_h + D_c\right) * N + D_w \tag{1}$$

where $D_a, D_r, D_h, D_c, andD_w$ represent the delay needed to add a new block, delay needed to read existing blocks, delay needed for hashing, delay needed for checking hash values, and delay needed for writing a block, while $L\&N$ represent the current length of blockchain, and a number of times the block hash is found to be non-unique. It is observed in Table 2 and Fig. 2, that the result of total delay computations over POS, POW, and the proposed hybrid consensus algorithm. The Random number of blocks is taken to analyse the efficiency of the proposed algorithm in terms of delay.

It is observed that adding a new block, exponentially increases with an increase in blockchain length, and thus researchers must explore options to reduce internal delays for improving blockchain speed. But most of the models used for delay reduction have limited scalability, and reduced flexibility when applied to large-scale network deployments. In order to improve blockchain scalability, while ensuring high quality of service (QoS) mining performance, the upcoming section discusses the design of the proposed trust integrated self-correcting miner selection model.

**Table 1** Comparative analysis of related work

| S. no. | Literature survey | Method | Types of attack | Objective | Research gaps |
|---|---|---|---|---|---|
| 1 | Sarvar et. al [32] | Proof of work and proof of stake consensus methods | 51% attack, Long range attack, DPoS attack, Sybil attack, Balance attack | Analysed the impact of 51% attack on different consensus methods and reveals their weakness over it | Attacks were not analysed on simulator and results were also not computed |
| 2 | Wei et. al [33] | Hyperledger Fabric and Leader election in vote change mechanism for consensus methods | No | Improved Raft consensus algorithm for the Hyperledger Fabric platform for both log replication and leader election | Delay analysis on blockchain is not computed |
| 3 | Marcela et al. [34] | Blockchain reputation-based consensus method | 51% attack, Double Spending attack, DoS attack, Selfish Miner attack | It provides scalable access control, selection of miners using trust based threshold computation | During selection of miners, trust is not computed to identify the list of trusted miner nodes |
| 4 | Amani et al. [35] | Light Weight Mining (LWM) method | DoS attack | It provides the reliable data provenance using LWM and communicating sequential process (CSP) for identify delay attacks for secure transactions | End-to-end delay analysis is not done |
| 5 | Sivaganesan [36] | Data driven Trust based method for blockchain | Grey and black hole attacks | Data driven trust based mechanism and energy efficient solution to detect internal attacks to improve network efficiency | Self-correction mechanism for malicious nodes is not incorporated |
| 6 | Ping et al. [37] | Partial equilibrium framework | No | This paper proposed short term network hash rate shock affects amid transaction and rewards going in opposite direction using proof of work consensus and self-correction mechanism while generating rewards | Trust based methods is not incorporated during self- correction of mining nodes to issue reward income |

**Table 2** Comparative results of delay computation on different consensus algorithm

| No. of blocks | Total average delay using POW (s) | Total average delay using POS (s) | Total average delay using proposed consensus (s) |
|---|---|---|---|
| 1000 | 0.1396 | 0.171 | 0.1175 |
| 10,000 | 5.2311 | 5.0321 | 4.1102 |
| 20,000 | 18.1222 | 18.0001 | 16.9137 |
| 30,000 | 42.5612 | 41.1265 | 39.6552 |
| 40,000 | 98.0133 | 97.0012 | 92.6554 |
| 50,000 | 202.1341 | 201.7653 | 191.7811 |
| 100,000 | 563.0431 | 562.6712 | 554.1062 |

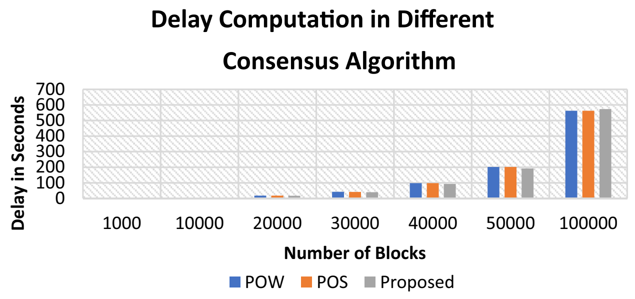

**Fig. 2** Delay analysis on different consensus algorithms

The comprehensive flow of the model is described in Fig. 3, wherein entire process of mining node selection, block creation & self-correction mining is visualized. The model initially uses a novel trust-factor evaluation method for selecting the best miner nodes. This trust factor is evaluated using node-to-node distance, energy levels, and historical trust values. Once miners are selected, then a hybrid consensus model is activated, which uses a combination of PoS, and PoW-based methods to improve mining efficiency. If the node is attacked, then the trust value is decreased and the self-correction process is called to update the hash of an affected node and validated the blockchain. Process of self-correction works simultaneously with the Trust Value computation of the node. Each selected miner node is validated using a self-correcting mechanism, which assists in the identification of hash mismatches and correcting them using a linked copy mechanism. The model design is categorised into 3 different parts, and each of which is discussed in detail in different sub-sections of this text.

## 3.1 Trust-based miner node selection model

In order to design trust-based miner node selection, the proposed model utilizes node-to-node distance, node residual energy levels, and temporal trust levels. These levels are iteratively evaluated for each node, and a relative node score is evaluated using the following process,

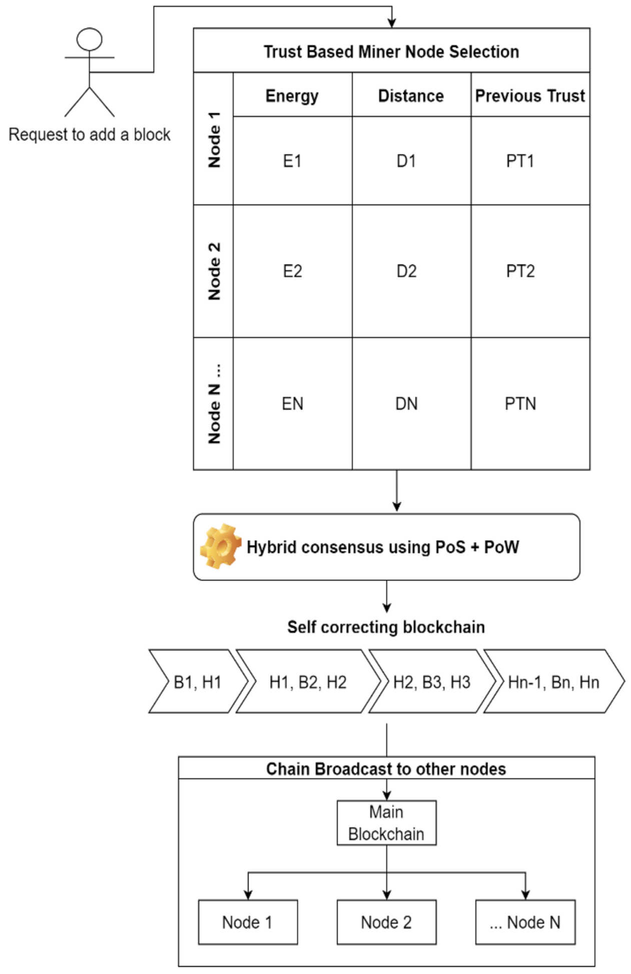

**Fig. 3** Overall flow for the proposed TISCMB model

- Request for block addition originates from a source node, located at position $x_s, y_s$ in the cartesian space.
- Let there by '$n$' other nodes in the space, with locations $[(x_1, y_1), (x_2, y_2), \ldots (x_n, y_n)]$
- Let the energy of these nodes be represented as $E_1, E_2, \ldots, E_n$
- Initialize a previous trust level for each node, and instantiate it with a constant value $t$

- For each node, evaluate its relative trust score ($RTS$) using Eq. (2),

$$RTS_{i,S} = \frac{E_i}{d_{i,S}} + \frac{TL_i}{TL_S} \tag{2}$$

where $d_{i,S}, andTL_i$ represent the distance between node $i\&S$, and the previous trust level of node $i$, which is updated after every evaluation.

- Find the average relative trust threshold using Eq. (3),

$$RTS_{th} = \frac{\sum_{i=1}^{n} RTS_{i,S}}{n} \tag{3}$$

- Discard all nodes with trust levels lower than $RTS_{th}$, and select other nodes as probable miner nodes.

Based on this process, a list of probable miner nodes is identified. Each of these nodes is checked for blockchain validation, and correction using the following process,

- For each node in the list of probable miner nodes, validate their local blockchain copy, using the following steps,

  - Let the number of blocks in this blockchain copy be $N$
  - For each block in the blockchain, check the identity given in Eq. (4),

$$Previous\,Hash(i) = Hash(i-1) \tag{4}$$

  - If this identity holds true for all blocks in the local copy, then mark this chain as validated, and reduce its energy level using Eq. (5),

$$New_E = Old_E - \sum_{i=1}^{N} E_{check_i} \tag{5}$$

where $New_E, Old_E, andE_{check}$ represent the new energy level of the recent node, old energy level of the current node, and the energy needed by a node to check & validate one block.

- Relative trust values for all probable miner nodes with validated blockchains are directly used, and their trust level is incremented while for miner nodes with invalid blockchains, Blockchains are self-corrected via the process given in Sect. 3.3 and their trust level of respective node is reduced using Eq. (6) as followswhere $N_{Inv}$ represents the number of invalidated blocks found in the blockchain.

$$TL = \begin{cases} TL + 1, & if\ node\ is\ validated \\ TL - N_{INV}, & Otherwise \end{cases} \tag{6}$$

- The energy level of this node is reduced using Eq. (7) as follows

$$E = E - N_{Inv} * E_{check} \tag{7}$$

- These new values of energy and trust levels are used for re-evaluation of relative trust score.
- Upon re-evaluation, the nodes are re-ranked according to $RTS$ values, and a final ranking list is evaluated.

**Algorithm 1: Selection of Trusted Node**

```
Input: Maximum number of nodes =N, Source = Xn, destination = Yn
Output: Miner nodes selected to perform consensus
Steps: For each node 1 to N
            Evaluate relative trust score $RTS_{i,s}$=( $(E_i/D_{i,s})$ + $(TL_i/TL_s)$)
            For (0<i<n)
                $RTS_{i,s}$= ($RTS_{i,s}$/n)
                Append all probable mining nodes having high trust level to the blockchain
            Evaluate validity via $Previous\ Hash(i) = Hash(i-1)$
            Evaluate Energy and Trust level using Equation 6 and 7
      Nodes Selected
```

**Table 3** The used block structure for storage

| | | | |
|---|---|---|---|
| Prev. hash | Source node | Destination node | Data |
| Time stamp | Meta data | *Nonce* | Hash value |

Based on this final ranking list, miner nodes are selected, and used for consensus by the hybrid consensus model. The complete design of the hybrid consensus model is discussed in the next sub-section of this text, wherein its internal working is discussed in detail.

## 3.2 Hybrid consensus model

Once miner nodes are selected by the trust-based model, then each of the nodes executes a hybrid consensus algorithm for hash evaluation. To perform this task, the following block structure as discussed in Table 3 is used.

In the current block structure, values of a source node, destination node, data, timestamp, and previous hash are prefilled, and cannot be modified. Thus, to enforce block-level uniqueness, the *Nonce* value must be modified. In order to generate a new nonce value, the following Eq. (8) that utilizes chain length ($C_{length}$) is used,

$$Nonce = random\left(RTS_{i,S} + Stake_S + Timestamp + C_{length}\right) \quad (8)$$

where *Stake* is initialized with unity value at node-level, and incremented for each source node. Due to the addition of $RTS, Stake\&Timestamp$ values, the model is able to generate unique hash values at faster speeds.

This is due to the fact that in PoW blockchains, chain lengths are used, while in PoS blockchains, node stakes are used to generate unique hash values, which restricts the random number feature space. While in the hybrid model, a combination of chain length, node stake, and relative trust score is used for enhancing random number generation, thereby reducing the overall delay needed for hash generation. Based on the generated *nonce* value, each node can find a unique hash value, and provide it for consensus. In order to a hash value to be accepted, conditions in Eq. (9) must be satisfied,

$$\begin{aligned} &Gen_{hash} \notin Previous\ Hashes \\ &T_{hash} = Min\left(T_i\right)\ where\ i \in M_{selected} \end{aligned} \quad (9)$$

where $Gen_{hash}, T_{hash}, and M_{selected}$ represent generated hash, time needed for hashing, and a list of selected miner nodes. Based on these conditions, new hashes are generated, and blocks are formed. The step-wise procedure of the proposed model is shown by algorithm 1, wherein process hybrid consensus is summarized. These blocks are validated using a self-correcting blockchain model, which is described in the next sub-section.

## 3.3 Self-correcting blockchain model

Any wireless node is prone to a wide variety of attacks, which include man in the middle, masquerading, spoofing, spying, etc. During attack, internal blockchains stored on

**Algorithm 2: Hybrid Consensus Model**

```
Input: Created Blockchain
Output: Hybrid Consensus Decision
Steps: Selected Miner Node, For each node 1 to N
            Evaluate, Nonce=random ($RTS_{i,S}$+$Stake_S$+Timestamp+$C_{length}$ )
            Generate Hash, $Gen_{hash} \in Previous\ Hashes$
                $T_{hash} = Min(T_i)\ where\ i \in M_{selected}$
            Utilize PoS & PoW for blockchain mining process
  Select miner outputs with faster results
```

**Table 4** Network parameters used for analysis

| Network parameter | Parameter value |
|---|---|
| Number of nodes | 100–1000 |
| Size of network | 300 m × 300 m |
| Idle power used | 1 mW |
| Power of reception | 1 mW |
| Power of transmission | 2 mW |
| Sleep power | 0.001 mW |
| Power for transition | 0.2 mW |
| Number of blocks | 1000 to 1 million |

nodes are exposed to hackers, which allows them to modify data, and hash values present in the chain. In order to self-correct such blockchains, the following process is used,

- For each block in the chain,Evaluate its hash value using
    - Evaluate its hash value using Eq. (10),

$$Hash = SHA256\begin{pmatrix} Prev.Hash, Src, Dest, \\ Data, Timestamp, \\ Metadata, Nonce \end{pmatrix} \tag{10}$$

    - Compare the current hash value, with the previous hash value from the next block, and verify whether they are equal.
    - If they are, then go to next block in the chain, else find a node with the highest RTS value, and copy the blockchain from that node.
- Once the blocks are copied, then perform chain correction using Eq. (4) as follows,

**Table 5** Average end-to-end delay for different consensus models (Sybil attack)

| Number of attack (%) | End-to end delay (ms) | | | |
|---|---|---|---|---|
| | PoS [14] | PoW [4] | PoPT [12] | Proposed |
| 1 | 0.80 | 0.86 | 0.93 | 0.70 |
| 1.5 | 0.88 | 0.94 | 1.00 | 0.77 |
| 2 | 0.95 | 0.99 | 1.05 | 0.81 |
| 2.5 | 0.98 | 1.02 | 1.10 | 0.85 |
| 3 | 1.03 | 1.09 | 1.16 | 0.90 |
| 3.5 | 1.10 | 1.17 | 1.28 | 0.97 |
| 4 | 1.19 | 1.33 | 1.50 | 1.10 |
| 4.5 | 1.42 | 1.69 | 1.91 | 1.38 |
| 5 | 1.91 | 2.19 | 2.40 | 1.77 |
| 5.5 | 2.40 | 2.52 | 2.71 | 2.09 |
| 6 | 2.56 | 2.72 | 2.95 | 2.24 |
| 6.5 | 2.78 | 3.04 | 3.31 | 2.50 |
| 7 | 3.18 | 3.45 | 3.75 | 2.84 |
| 8 | 3.61 | 3.85 | 4.18 | 3.18 |
| 9 | 3.95 | 4.28 | 4.61 | 3.51 |
| 10 | 4.45 | 4.65 | 4.99 | 3.85 |

$$Previous\,Hash(i) = Hash(i-1)$$

Once this process is followed, the entire chain of the current node is validated. For each block that is found to be non-validated, the chain's trust levels, and energy levels are modified using Eqs. (6) and (7), which assists in re-ranking nodes for better miner selection. Based on this process, new blocks are added to the blockchain, and their performance is evaluated. This performance evaluation is done in terms of transaction delay, and energy consumption, under a number of attacks that are mitigated by the system. A comparative

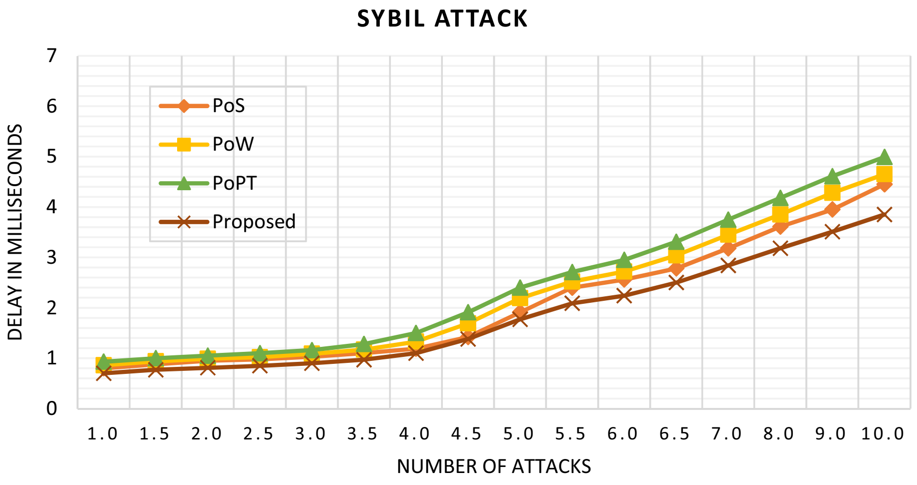

**Fig. 4** End-to-end delay analysis over Sybil attack

**Table 6** Average end-to-end delay for different consensus models (Spying attacking)

| Number of attack (%) | End-to end delay (ms) | | | |
|---|---|---|---|---|
| | PoS [14] | PoW [4] | PoPT [12] | Proposed |
| 1 | 0.92 | 0.98 | 1.06 | 0.81 |
| 1.5 | 1.01 | 1.07 | 1.15 | 0.88 |
| 2 | 1.09 | 1.13 | 1.21 | 0.95 |
| 2.5 | 1.14 | 1.19 | 1.26 | 0.98 |
| 3 | 1.19 | 1.25 | 1.34 | 1.03 |
| 3.5 | 1.26 | 1.34 | 1.46 | 1.11 |
| 4 | 1.37 | 1.53 | 1.73 | 1.27 |
| 4.5 | 1.62 | 1.95 | 2.20 | 1.57 |
| 5 | 2.20 | 2.52 | 2.76 | 2.05 |
| 5.5 | 2.75 | 2.90 | 3.11 | 2.40 |
| 6 | 2.94 | 3.12 | 3.39 | 2.59 |
| 6.5 | 3.19 | 3.49 | 3.81 | 2.87 |
| 7 | 3.66 | 3.98 | 4.31 | 3.27 |
| 8 | 4.15 | 4.42 | 4.80 | 3.66 |
| 9 | 4.54 | 4.92 | 5.30 | 4.04 |
| 10 | 5.12 | 5.35 | 5.74 | 4.43 |

**Table 7** Average end-to-end delay for different consensus models (Spoofing attack)

| Number of attack (%) | End-to end delay (ms) | | | |
|---|---|---|---|---|
| | PoS [14] | PoW [4] | PoPT [12] | Proposed |
| 1 | 0.79 | 0.84 | 0.90 | 0.69 |
| 1.5 | 0.85 | 0.91 | 0.98 | 0.75 |
| 2 | 0.94 | 0.97 | 1.03 | 0.80 |
| 2.5 | 0.96 | 1.00 | 1.08 | 0.84 |
| 3 | 1.01 | 1.07 | 1.14 | 0.88 |
| 3.5 | 1.08 | 1.14 | 1.25 | 0.95 |
| 4 | 1.16 | 1.30 | 1.46 | 1.08 |
| 4.5 | 1.38 | 1.65 | 1.88 | 1.34 |
| 5 | 1.86 | 2.15 | 2.35 | 1.74 |
| 5.5 | 2.34 | 2.46 | 2.65 | 2.04 |
| 6 | 2.50 | 2.65 | 2.89 | 2.20 |
| 6.5 | 2.72 | 2.97 | 3.24 | 2.44 |
| 7 | 3.11 | 3.38 | 3.66 | 2.78 |
| 8 | 3.53 | 3.76 | 4.08 | 3.11 |
| 9 | 3.85 | 4.18 | 4.50 | 3.43 |
| 10 | 4.36 | 4.54 | 4.88 | 3.77 |

analysis of these parameters can be discussed in the next section of this text.

## 4 Result analysis and comparison

In order to evaluate the performance of the proposed hybrid model, the blockchain network was tested under the large number of connection requests. All these requests were conducted on a standard network environment, which consists the following parameters (Table 4).

The number of nodes varied from 100 to 1000, and the following attacks (along with the reasons for selecting them) were injected into the system,

- Sybil attack (selected because blocks in the blockchain can be modified by attackers)
- Spying attack (selected because blocks in the blockchain can be spied upon by attackers via route disruptions)
- Spoofing attack (selected because false blocks can be added by attackers)

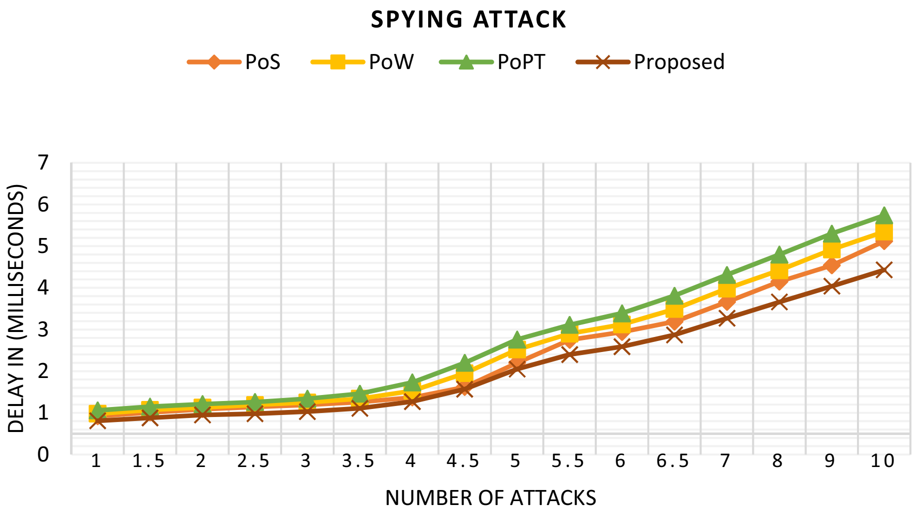

**Fig. 5** End-to-end delay analysis over Spying attack

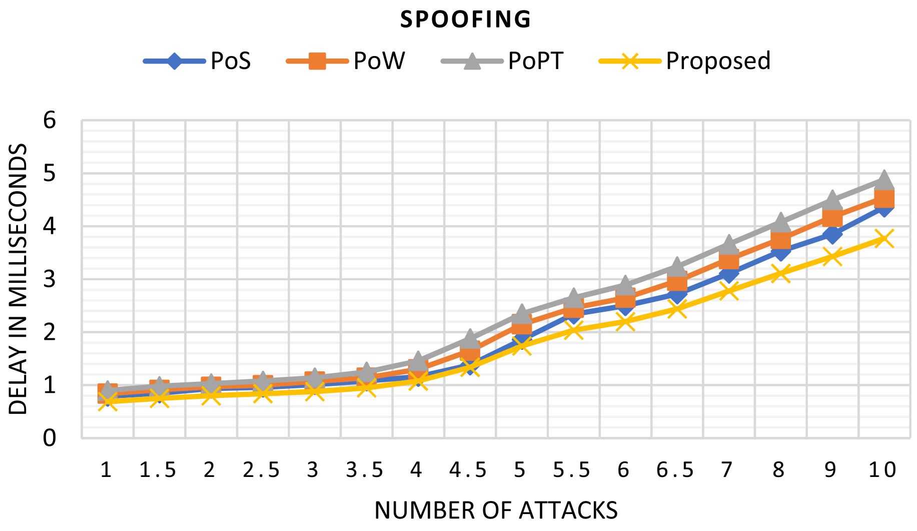

**Fig. 6** End-to-end delay analysis over Spoofing attack

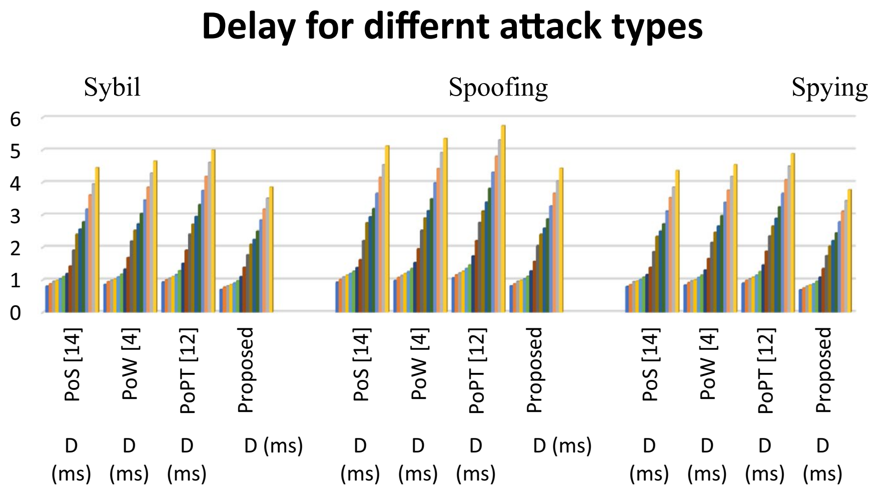

**Fig. 7** Delay for different models

Due to the inclusion of a self-correcting model, and hybrid consensus with trust-based miner selection, QoS of the proposed kernel is better when it is compared with PoS [14], PoW [4], and PoPT [12] under different attacks. This performance is evaluated by changing the number of attacker (NA) nodes between 1 and 10%, and by evaluating QoS values. Average QoS values were estimated by adding 100k blocks for a network of 1000 nodes. This gives the true estimation of the performance of the hybrid model and includes large-scale deployment scenarios. According to the strategy used for evaluation, the values for end-to-end delay (D) for different existing models with the proposed model can be observed under Sybil attack are tabulated in Table 5.

From this evaluation, and Fig. 4, it is estimated that the proposed hybrid model is 20% faster than PoS [14], 23% faster than PoW [4], and 28% faster than PoPT [12] under Sybil attack. Similar delay performance is obtained under different Spying attackers and compared over existing methods with the proposed method. This can be observed from Table 6 as follows.

From this evaluation and Fig. 5, it is estimated that the proposed hybrid model is 18% faster than PoS [14], 20% faster than PoW [4], and 22% faster than PoPT [12] under Spying attack. This delay is further estimated for Spoofing attack which can be analysed over existing methods with the proposed method, is observed in Table 7.

From this evaluation and Fig. 6, it is estimated that the proposed hybrid model is 14% faster than PoS [14], 16% faster than PoW [4], and 19% faster than PoPT [12] under Spoofing attack. The reason for this delay reduction as observed in Fig. 7, trust-based miner selection, the use of automatic correction of attacked nodes, which assists in reducing the delay needed for securely mining new blocks. The proposed hybrid consensus algorithm gives better results.

Similar estimations are performed for energy consumption, and this can be observed for Sybil attack and analysed

**Table 8** Average energy consumption for different consensus models (Sybil)

| Number of attack (%) | Energy consumption (mJ) | | | |
|---|---|---|---|---|
| | PoS [14] | PoW [4] | PoPT [12] | Proposed |
| 1 | 2.05 | 2.35 | 2.56 | 1.65 |
| 1.5 | 2.57 | 2.66 | 2.85 | 1.91 |
| 2 | 2.66 | 2.79 | 2.99 | 2.00 |
| 2.5 | 2.82 | 2.96 | 3.16 | 2.11 |
| 3 | 2.99 | 3.15 | 3.35 | 2.24 |
| 3.5 | 3.18 | 3.30 | 3.51 | 2.36 |
| 4 | 3.30 | 3.43 | 3.65 | 2.46 |
| 4.5 | 3.43 | 3.56 | 3.80 | 2.55 |
| 5 | 3.56 | 3.71 | 3.96 | 2.66 |
| 5.5 | 3.72 | 3.92 | 4.20 | 2.79 |
| 6 | 3.96 | 4.21 | 4.51 | 2.99 |
| 6.5 | 4.32 | 4.46 | 4.69 | 3.18 |
| 7 | 4.43 | 4.46 | 4.64 | 3.19 |
| 8 | 4.32 | 4.28 | 4.24 | 3.03 |
| 9 | 4.08 | 3.31 | 3.58 | 2.58 |
| 10 | 2.42 | 3.62 | 4.19 | 2.42 |

**Table 9** Average energy consumption for different consensus models (Spying attack)

| Number of attack (%) | Energy consumption (mJ) | | | |
|---|---|---|---|---|
| | PoS [14] | PoW [4] | PoPT [12] | Proposed |
| 1 | 2.36 | 2.71 | 2.95 | 1.89 |
| 1.5 | 2.95 | 3.07 | 3.28 | 2.19 |
| 2 | 3.07 | 3.21 | 3.44 | 2.30 |
| 2.5 | 3.25 | 3.40 | 3.64 | 2.43 |
| 3 | 3.43 | 3.62 | 3.86 | 2.57 |
| 3.5 | 3.66 | 3.80 | 4.04 | 2.72 |
| 4 | 3.80 | 3.95 | 4.20 | 2.82 |
| 4.5 | 3.95 | 4.10 | 4.36 | 2.93 |
| 5 | 4.10 | 4.27 | 4.56 | 3.06 |
| 5.5 | 4.27 | 4.50 | 4.84 | 3.21 |
| 6 | 4.56 | 4.85 | 5.19 | 3.44 |
| 6.5 | 4.96 | 5.12 | 5.39 | 3.65 |
| 7 | 5.09 | 5.13 | 5.34 | 3.68 |
| 8 | 4.97 | 4.92 | 4.88 | 3.49 |
| 9 | 4.70 | 3.81 | 4.11 | 2.97 |
| 10 | 2.78 | 4.16 | 4.81 | 2.77 |

over existing methods with the proposed method from the following Table 8.

From this evaluation and Fig. 8, it can be estimated that the proposed model is able to reduce energy consumption by 20% than PoS [14], 15% than PoW [4], and 18% than PoPT [12] under Sybil attack. Similar performance was observed for spying attacks and compared with the existing methods and the proposed method, this can be observed in Table 9 as follows.

From this evaluation and Fig. 9, it can be estimated that the proposed model is able to reduce energy consumption by 26% than PoS [14], 18% than PoW [4], and 22% than PoPT [12] under Spying attack. This energy consumption is further evaluated for Spoofing attack and compared with the existing methods and the proposed method, this can be observed in Table 10 as follows.

From this evaluation and Fig. 10, it is observed that the proposed model is able to reduce energy consumption by 41% than PoS [14], 29% than PoW [4], and 33% than PoPT [12] under Spoofing attack.

The consolidated average analysis for energy required is observed in Fig. 11, where proposed model depicts better

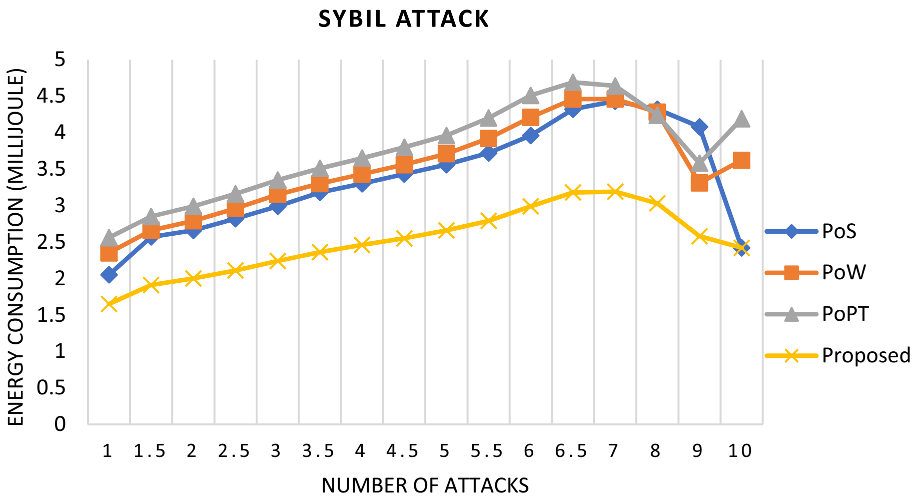

**Fig. 8** Energy consumption over Sybil attack

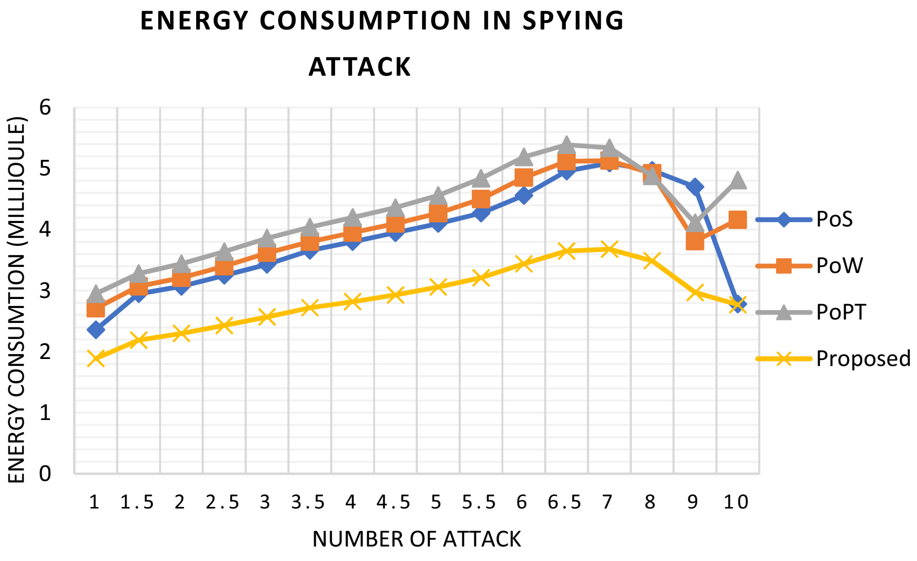

**Fig. 9** Energy consumption over Spying attack

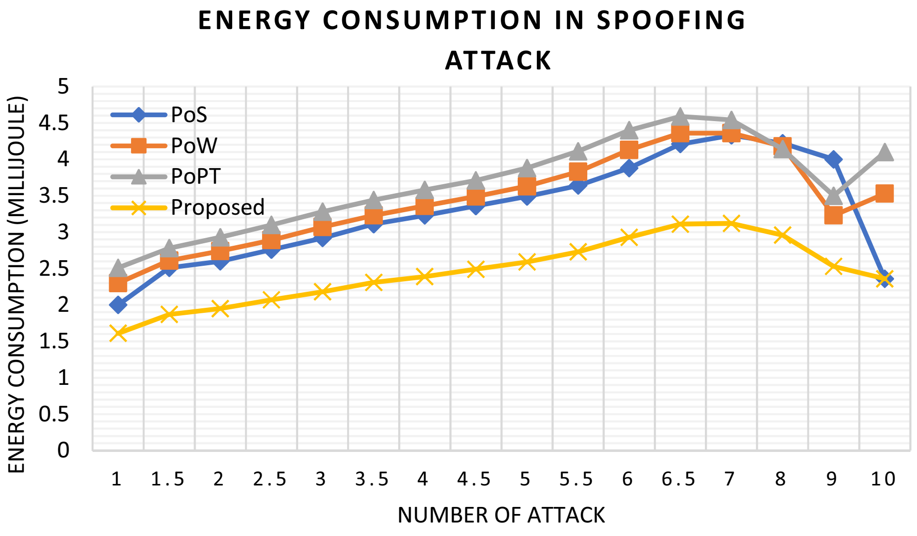

**Fig. 10** Energy consumption over Spoofing attack

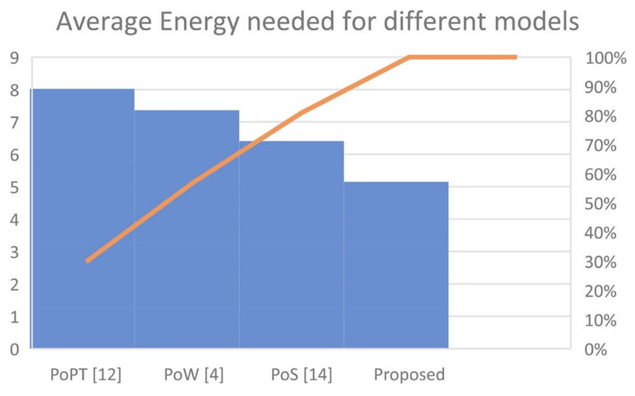

**Fig. 11** Energy consumption for different models

results. The reason behind this reduction in the above parameters, is the use of trust-based mining nodes, and hybrid consensus, which reduces the number of rounds needed for generating unique hash values, thereby reducing delay, and energy consumption when tested for different attack types.

## 5 Conclusion and future scope

In this paper, we proposed TISCMB model uses a combination of hybrid consensus, with trust-based miner selection which assists in improving the QoS performance of the blockchain network. Furthermore, due to the addition of self-correction, the network is able to reduce the probability of attacks, due to which the overall efficiency of

**Table 10** Average energy consumption for different consensus models (Spoofing attack)

| Number of attack (%) | Energy consumption (mJ) | | | |
|---|---|---|---|---|
| | PoS [14] | PoW [4] | PoPT [12] | Proposed |
| 1 | 2.00 | 2.30 | 2.51 | 1.61 |
| 1.5 | 2.51 | 2.61 | 2.78 | 1.87 |
| 2 | 2.60 | 2.74 | 2.93 | 1.95 |
| 2.5 | 2.76 | 2.89 | 3.10 | 2.07 |
| 3 | 2.92 | 3.07 | 3.28 | 2.18 |
| 3.5 | 3.11 | 3.23 | 3.44 | 2.31 |
| 4 | 3.23 | 3.36 | 3.58 | 2.39 |
| 4.5 | 3.36 | 3.49 | 3.71 | 2.49 |
| 5 | 3.49 | 3.63 | 3.88 | 2.59 |
| 5.5 | 3.64 | 3.83 | 4.11 | 2.73 |
| 6 | 3.88 | 4.13 | 4.40 | 2.93 |
| 6.5 | 4.21 | 4.36 | 4.59 | 3.11 |
| 7 | 4.33 | 4.36 | 4.54 | 3.12 |
| 8 | 4.22 | 4.18 | 4.14 | 2.96 |
| 9 | 4.00 | 3.23 | 3.50 | 2.53 |
| 10 | 2.36 | 3.53 | 4.10 | 2.36 |

blockchain storage is improved. This efficiency was evaluated in terms of transaction delay, and energy consumption under different attack types. It was observed that for Sybil attack, the proposed hybrid model is 20% faster than PoS [14], 23% faster than PoW [4], and 28% faster than PoPT [12]; and the proposed model is able to reduce energy consumption by 20% than PoS [14], 15% than PoW [4], and 18% than PoPT [12]. Similarly, for Spying attack, the proposed model is 18% faster than PoS [14], 20% faster than PoW [4], and 22% faster than PoPT [12], while the proposed hybrid model can reduce energy consumption by 26% than PoS [14], 18% than PoW [4], and 22% than PoPT [12]. Finally, for Spoofing attack, it is estimated that the proposed model is 14% faster than PoS [14], 16% faster than PoW [4], and 19% faster than PoPT [12], while, the proposed model can reduce energy consumption by 41% than PoS [14], 29% than PoW [4], and 33% than PoPT [12], which makes the proposed model highly useful and efficient under a wide variety of attacks. In future this work can be extended to different deep learning models for improving overall performance of miner node selection, which will further improve network lifetime, and speed of transactions, which will improve overall network scalability, flexibility, and effectiveness.

**Acknowledgements** I would like to thank my supervisor, my friends, and my institution for providing me wide research platform to carry out my research work in the right direction.

**Data availability** This article is not associated with any data set.

## Declarations

**Conflict of interest** The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## References

1. Hu M, Shen T, Men J, Yu Z, Liu Y (2020) CRSM: an effective blockchain consensus resource slicing model for real-time distributed energy trading. IEEE Access 8:206876–206887. https://doi.org/10.1109/ACCESS.2020.3037694
2. Yadav AS, Kushwaha DS (2021) Digitization of land record through blockchain-based consensus algorithm. IETE Tech Rev. https://doi.org/10.1080/02564602.2021.1908859
3. Anceaume E, Busnel Y, Sericola B (2021) Byzantine-tolerant uniform node sampling service in large-scale networks. Int J Parallel Emergent Distrib Syst 36(5):412–439. https://doi.org/10.1080/17445760.2021.1939873
4. Yang F, Zhou W, Wu Q, Long R, Xiong NN, Zhou M (2019) Delegated proof of stake with downgrade: a secure and efficient blockchain consensus algorithm with downgrade mechanism. IEEE Access 7:118541–118555. https://doi.org/10.1109/ACCESS.2019.2935149
5. Santiago C, Ren S, Lee C, Ryu M (2021) Concordia: a streamlined consensus protocol for blockchain networks. IEEE Access 9:13173–13185. https://doi.org/10.1109/ACCESS.2021.3051796
6. Huang D, Ma X, Zhang S (2019) Performance analysis of the raft consensus algorithm for private blockchains. IEEE Trans Syst Man Cybern Syst 50(1):172–181. https://doi.org/10.1109/TSMC.2019.2895471
7. Sun G, Dai M, Sun J, Yu H (2020) Voting-based decentralized consensus design for improving the efficiency and security of consortium blockchain. IEEE Internet Things J 8(8):6257–6272. https://doi.org/10.1109/JIOT.2020.3029781
8. Xiao Y, Zhang N, Lou W, Hou YT (2020) A survey of distributed consensus protocols for blockchain networks. IEEE Commun Surv Tutor 22(2):1432–1465. https://doi.org/10.1109/COMST.2020.2969706
9. Otsuki K, Nakamura R, Shudo K (2021) Impact of saving attacks on blockchain consensus. IEEE Access 9:133011–133022. https://doi.org/10.1109/ACCESS.2021.3115131
10. Meshcheryakov Y, Melman A, Evsutin O, Morozov V, Koucheryavy Y (2021) On performance of PBFT blockchain consensus algorithm for IoT-applications with constrained devices. IEEE Access 9:80559–80570. https://doi.org/10.1109/ACCESS.2021.3085405
11. Wang W, Hoang DT, Hu P, Xiong Z, Niyato D, Wang P, Wen Y, Kim DI (2019) A survey on consensus mechanisms and mining strategy management in blockchain networks. IEEE Access 7:22328–22370. https://doi.org/10.1109/ACCESS.2019.2896108
12. Aponte F, Gutierrez L, Pineda M, Merino I, Salazar A, Wightman P (2021) Cluster-based classification of blockchain consensus algorithms. IEEE Lat Am Trans 19(4):688–696. https://doi.org/10.1109/TLA.2021.9448552
13. Xiang F, Huaimin W, Peichang S (2019) Proof of previous transactions (PoPT): an efficient approach to consensus for JCLedger. IEEE Trans Syst Man Cybern Syst 51(4):2415–2424. https://doi.org/10.1109/TSMC.2019.2913007

14. Liang W, Zhang D, Lei X, Tang M, Li KC, Zomaya AY (2020) Circuit copyright blockchain: blockchain-based homomorphic encryption for IP circuit protection. IEEE Trans Emerg Top Comput 9(3):1410–1420. https://doi.org/10.1109/TETC.2020.2993032
15. Pang Y (2020) A new consensus protocol for blockchain interoperability architecture. IEEE Access 8:153719–153730. https://doi.org/10.1109/ACCESS.2020.3017549
16. Lashkari B, Musilek P (2021) A comprehensive review of blockchain consensus mechanisms. IEEE Access 9:43620–43652. https://doi.org/10.1109/ACCESS.2021.3065880
17. Bhutta MNM, Khwaja AA, Nadeem A, Ahmad HF, Khan MK, Hanif MA, Song H, Alshamari M, Cao Y (2021) A survey on blockchain technology: evolution, architecture and security. IEEE Access 9:61048–61073. https://doi.org/10.1109/ACCESS.2021.3072849
18. Wang Y, Cai S, Lin C, Chen Z, Wang T, Gao Z, Zhou C (2019) Study of blockchains's consensus mechanism based on credit. IEEE Access 7:10224–10231. https://doi.org/10.1109/ACCESS.2019.2891065
19. Hu W, Hu Y, Yao W, Li H (2019) A blockchain-based Byzantine consensus algorithm for information authentication of the Internet of vehicles. IEEE Access 7:139703–139711. https://doi.org/10.1109/ACCESS.2019.2941507
20. Ray PP, Dash D, Salah K, Kumar N (2020) Blockchain for IoT-based healthcare: background, consensus, platforms, and use cases. IEEE Syst J 15(1):85–94. https://doi.org/10.1109/JSYST.2020.2963840
21. Meng T, Zhao Y, Wolter K, Xu CZ (2021) On consortium blockchain consistency: a queueing network model approach. IEEE Trans Parallel Distrib Syst 32(6):1369–1382. https://doi.org/10.1109/TPDS.2021.3049915
22. Ngubo CE, Dohler M (2020) Wi-Fi-dependent consensus mechanism for constrained devices using blockchain technology. IEEE Access 8:143595–143606. https://doi.org/10.1109/ACCESS.2020.3014287
23. Liang Y, Lu C, Zhao Y, Sun C (2021) Interference-based consensus and transaction validation mechanisms for blockchain-based spectrum management. IEEE Access 9:90757–90766. https://doi.org/10.1109/ACCESS.2021.3091802
24. Zhang P, Zhou M, Zhao Q, Abusorrah A, Bamasag OO (2021) A performance-optimized consensus mechanism for consortium blockchains consisting of trust-varying nodes. IEEE Trans Netw Sci Eng 8(3):2147–2159. https://doi.org/10.1109/TNSE.2021.3079415
25. Bao Z, Wang Q, Shi W, Wang L, Lei H, Chen B (2020) When blockchain meets sgx: an overview, challenges, and open issues. IEEE Access 8:170404–170420. https://doi.org/10.1109/ACCESS.2020.3024254
26. Feng J, Zhao X, Chen K, Zhao F, Zhang G (2020) Towards random-honest miners selection and multi-blocks creation: proof-of-negotiation consensus mechanism in blockchain networks. Future Gener Comput Syst 105:248–258. https://doi.org/10.1016/j.future.2019.11.026
27. Fan Y, Zou J, Liu S, Yin Q, Guan X, Yuan X, Wu W, Du D (2020) A blockchain-based data storage framework: a rotating multiple random masters and error-correcting approach. Peer-to-Peer Netw Appl 13(5):1486–1504. https://doi.org/10.1007/s12083-020-00895-5
28. Patil RY, Patil YH, Kachhoria R, Lonare S (2022) A provably secure data sharing scheme for smart gas distribution grid using fog computing. Int J Inf Technol. https://doi.org/10.1007/s41870-022-01043-3
29. Srividya R, Vyshali Rao KP (2022) Light weight hash function using secured key distribution technique for MANET. Int J Inf Technol 14(6):3099–3108. https://doi.org/10.1007/s41870-022-00940-x
30. Chaudhary RRK, Chatterjee K (2022) A lightweight security framework for electronic healthcare system. Int J Inf Technol 14(6):3109–3121. https://doi.org/10.1007/s41870-022-01034-4
31. Maurya R, Rao GV, Rajitha B (2022) Visual cryptography for securing medical images using a combination of hyperchaotic-based pixel, bit scrambling, and DNA encoding. Int J Inf Technol 14(6):3227–3234. https://doi.org/10.1007/s41870-022-01029-1
32. Sayeed S, Marco-Gisbert H (2019) Assessing blockchain consensus and security mechanisms against the 51% attack. Appl Sci 9(9):1788. https://doi.org/10.3390/app9091788
33. Fu W, Wei X, Tong S (2021) An improved blockchain consensus algorithm based on raft. Arab J Sci Eng 46(9):8137–8149. https://doi.org/10.1007/s13369-021-05427-8
34. de Oliveira MT, Reis LH, Medeiros DS, Carrano RC, Olabarriaga SD, Mattos DM (2020) Blockchain reputation-based consensus: a scalable and resilient mechanism for distributed mistrusting applications. Comput Netw 179:107367
35. Altarawneh A, Sun F, Brooks RR, Hambolu O, Yu L, Skjellum A (2021) Availability analysis of a permissioned blockchain with a lightweight consensus protocol. Comput Secur 102:102098. https://doi.org/10.1016/j.cose.2020.102098
36. Sivaganesan D (2021) A data driven trust mechanism based on blockchain in IoT sensor networks for detection and mitigation of attacks. J Trends Comput Sci Smart Technol (TCSST) 3(01):59–69. https://doi.org/10.36548/jtcsst.2021.1.006
37. Tang D, He P, Fan Z (2020) PoW blockchain network's short-term self-correction mechanism. Available at SSRN 3757830. https://doi.org/10.2139/ssrn.3757830