BLOCKCHAIN MULTI-BODY BALANCED CONSENSUS APPROACH ADAPTED TO ELECTRICITY BUSINESS CHARACTERISTICS

Han ZHANG[1], Liyong LIU[1], Zhi LI[1], Kailong SHAO[1], Jiancun LIU[2,*], Haoyang YAO[2]

The immutability and transparency of blockchain ensure secure sharing of energy data. This study proposes a multi-agent consensus approach for power systems that integrates dynamic trust evaluation and benefit-driven scheduling to overcome information isolation and prevent data tampering. The experimental results show 87.34% anti-tamper protection under non-51% attack scenarios, demonstrating superior performance compared to traditional methods.

Keywords: blockchain, power business, transmission and sharing, uncertainty, data security

1. Introduction

The security of power data is challenged by risks of tampering and centralization [1]. However, a critical limitation of existing blockchain privacy solutions, which primarily depend on encryption and authentication mechanisms, is their lack of semantic reliability [2-5]. This paper proposes a blockchain-based multi-agent consensus approach for power enterprises, integrating dynamic trust evaluation and benefit-driven scheduling to ensure secure information sharing within the supply chain.

The research foundation of this paper is a "dynamic trust-benefit scheduling coordination mechanism", which surpasses a simple combination of existing technologies such as Raft and BLS signatures. This mechanism acts as the central nervous system of the consensus model, integrating and empowering its components to achieve true semantic reliability. This synergistic mechanism dynamically updates trust values based on user behavior scores and quantifies risks/trust using benefit theory, achieving for the first time a balanced scheduling of reliability and efficiency in power systems. This addresses the lack of semantic reliability in traditional blockchain applications for the energy sector. It transforms the Raft leader election into a trust-aware process where leadership eligibility and voting weights are dynamically adjusted, significantly reducing the risk of

* Corresponding author: liujiancun@email.tjut.edu.cn
[1] SPIC Hebei Electric Power Co, LTD, Shijiazhuang, China.
[2] School of Electrical Engineering and Automation, Tianjin University of Technology, Tianjin, China.

malicious leaders. It provides the criteria for efficient resource allocation, ensuring that BLS signature aggregation and log replication priorities align with the current trust landscape and business objectives. Consequently, what is proposed is not merely a combination but an organic architecture where Raft, BLS, and a margin mechanism are coherently integrated and optimized by the dynamic trust-benefit scheduler. Experimental results validate that this integrated approach is key to achieving an 87.34% anti-tamper protection rate, which represents an improvement over traditional Raft and thereby confirms its superiority in securing and optimizing power system data sharing.

Therefore, the innovations of this work are articulated as follows: (1) A core coordination mechanism designed to ensure semantic reliability in power data transactions. (2) An anti-tampering enhancement scheme based on BLS aggregate signatures, which utilizes bilinear mapping to combine multiple signatures into a single compact form. This approach significantly reduces the verification overhead compared to traditional Schnorr or ECDSA schemes and provides theoretical tamper resistance under non-51% attack scenarios. (3) A multi-round interactive balanced consensus model built on a trust-qualified Raft algorithm, which optimizes both throughput and latency in distributed power transactions.

2. Blockchain technology

As a decentralized ledger (Fig. 1), the blockchain leverages a cryptographically linked, hash-based chain of blocks to establish data immutability. These properties of decentralization and tamper-resistance effectively mitigate the risks of single-point failures and data manipulation inherent in centralized power systems [6-7].

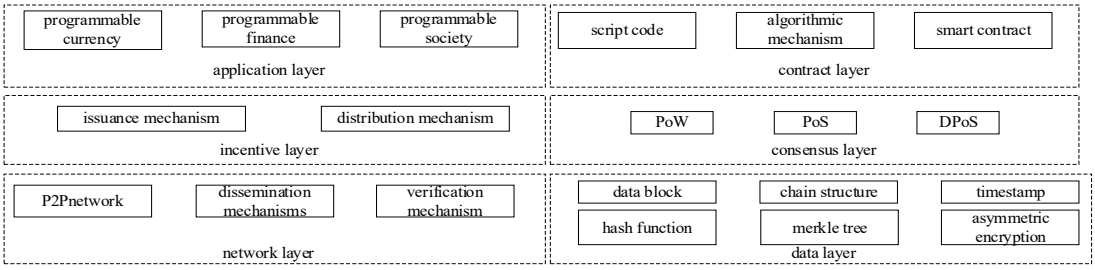

Fig. 1. Blockchain Infrastructure Model

3. Equilibrium consensus method based on blockchain technology

3.1 Characteristics of Plant-Grid Coordination Business System

The blockchain's decentralized ledger architecture stores transactional data within hash-chained blocks, thereby creating tamper-proof records to address the credibility and security risks in critical power infrastructures. Nodes adhere to format rules for data standardization and block structuring, with key identifiers

detailed in Table 1. Furthermore, the use of blocks with keyed digests enables efficient query operations, and the Raft consensus mechanism guarantees rapid state synchronization within the alliance chain.

Table 1

Data structure key identifiers

Markings	The processing ensures the data is uniquely identified, usually as a hash value.
Typology	When operating, define the data type of the operation, which can be one or more types.
Signatory	The set of signers that sign the data when the operation is performed.
Timestamp	Positive integers, calculated with millisecond precision starting from 1970, increase in positive order.

This approach utilizes BFT/Raft-based consensus to achieve trustworthiness, characterized by its efficiency with fewer nodes, rapid transaction speed, and lower energy consumption. The process involves two key steps:

(1) Role definition: Leader sends heartbeats, followers respond, and candidates emerge on failure/timeout.

(2) Election process: Nodes initially become Candidates and solicit votes using random timeouts; a new Leader is selected by majority vote.

$$\begin{cases} T_{\min} \le T_{\text{out}} \le T_{\max} \\ M_{\text{majority}} = \dfrac{N_{\text{Nodes}}}{2} + 1 \end{cases} \tag{1}$$

where $T_{\max}$ and $T_{\min}$ are the upper and lower limits for the length of the election, respectively. T_{out} is the output; M_{majority} is the leader node address; N_{Nodes} is the all candidate queues. If the election timeout timer expires but a candidate fails to get a majority of nodes to vote for it, a new round of the election is re-initiated.

(3) Log replication: The Leader replicates logs to the follower nodes and issues a commitment response once a majority acknowledge successful replication (Eq.2). Subsequently, a newly elected Leader performs this replication based on the Euclidean distance to optimize efficiency, following the criterion in Eq.3.

$$T_{\text{Commit}} = \text{Max}(T_1, T_2, \cdots, T_i, \cdots T_n) \tag{2}$$

$$\begin{cases} Q_1 = \sqrt{Q} / (0.3X) \\ T = \text{Max}(t_1, t_2, \cdots, t_i, \cdots, t_N) \end{cases} \tag{3}$$

where T_{Commit} is the commit replication length; T_i is the length of all sub-node log replication; Q_1 denotes the standard value of the proximity Euclidean distance; Q is the area transformed by the network region; X is the number of all participating nodes in the region; t_i is the time range and duration of the leader's release, as reported by node i. Each node is required to report its position to the Leader. Additionally, these nodes must report the duration for which they maintain a near-valid state. The Leader then sets an hourly duration to filter and limit invalid nodes, thereby increasing efficiency.

3.2 Multi-Round Interactive Balanced Consensus Approach Based on Raft Algorithm

The Raft-based multi-agent consensus mechanism ensures data consistency by means of leader election and log replication. It balances stakeholder interests via multi-round interactions and safeguards power transactions using a margin mechanism. These are formalized as follows:

(1) Election:

$$Leader-elected = \max(Vote-cont[i]) > \frac{N}{2} \tag{4}$$

where N is the total number of nodes; $Vote-cont[i]$ is the number of votes node i receives.

(2) Log Replication:

$$Log\_consistency = \forall i, j \in N, Log[i] = Log[j] \tag{5}$$

where $Log[i]$ represents the log entries of node i.

(3) Security:

$$Security\_level = f(Number\_ofreplicas, Encryption\_strength) \tag{6}$$

where f is a function that increases with the number of replicas and the encryption strength.

The Raft-based multi-body consensus approach provides an effective solution to traditional blockchain challenges by significantly improving leader election efficiency:

$$T_{election} = O(\log N) \tag{7}$$

At the same time, the log replication process in the Raft algorithm ensures data consistency, thereby mitigating the risk of data conflicts and inconsistencies:

$$P_{conflict} = \frac{1}{Number\_ofreplicas^{k}} \tag{8}$$

where k is a constant representing the efficiency of the replication process.

The Raft algorithm employs leader election, heartbeat monitoring, and log replication to coordinate multi-round consensus interactions:

$$\begin{aligned} &Consensus\_reached = \forall i \in N \\ &Deci\sin[i] = Deci\sin majority \end{aligned} \tag{9}$$

The margin mechanism enables nodes to achieve reliable consensus on power transactions, ensuring both accuracy and fairness:

$$Transaction\_security = g(Marginamount, Party\_trust\_levels) \tag{10}$$

where g is a function that increases with the margin amount and the participants' trust levels. Once a balanced consensus is reached, the power transaction is executed whereby the consumer transfers the payment, the power plant commits to delivering the energy, and the deposit is returned.

3.3 Single adaptive electricity business characteristics sharing model

The alliance blockchain establishes a shared power architecture. In this architecture, a top-level blockchain records all transactions, which are public and transparent. This ledger is maintained by a committee of members (e.g., national regulators and neutral third parties) who achieve consensus to validate new blocks, which elected leaders then generate. Fig. 2 illustrates the architecture diagram of the shared model, which adapts to the characteristics of the power business.

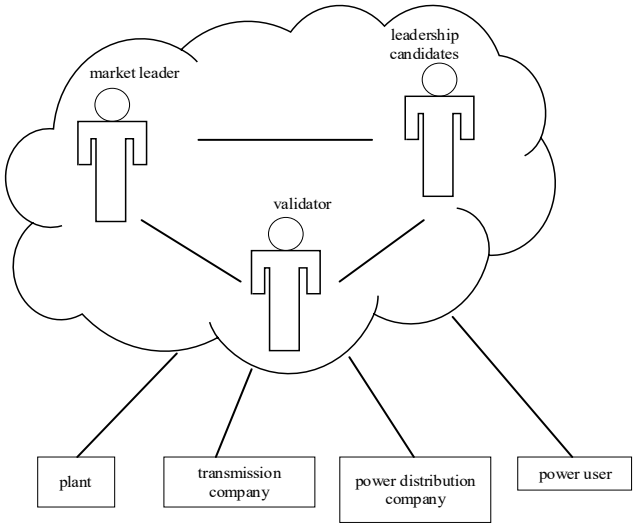

Fig. 2. Shared model architecture diagram adapted to the characteristics of the power business

4. Computational analysis

4.1 Equilibrium consensus process simulation

Validators are members of the Consortium Committee, apart from the current Leader. An individual may act as both a validator and a leadership candidate concurrently. Validators are classified into four states based on their trust values, with reclassification occurring post-consensus: Trusted validators (trust > threshold), regular validators (standard < trust $\leqslant$ threshold), faulty validators (0 < trust $\leqslant$ standard), and malicious nodes (trust = 0). The voting weight formula dictates that the weights of trusted validators increase with the number of faulty nodes but decrease with higher numbers of trusted validators. It is important to note that leadership eligibility is restricted to trusted validators.

$$C_{\text{trust}} = 1 + (\frac{N_{\text{faulated}}}{2} \times N \times N_{\text{trust}}) \tag{11}$$

where $N_{faulated}$ is the number of faulty verifiers; N_{trust} is the number of trusted verifiers. A trusted validator's voting rights increase in proportion to any rise in the voting rights of faulty validators but are reduced by increases in the voting rights of other trusted validators. Eligibility for the leader position is exclusively granted to trusted validators.

Sensitivity to Malicious Node Proportion: When the proportion of malicious nodes increases from 10% to 40%, the anti-tamper protection rate decreases from 92.1% to 87.34%, and the forgery prevention rate decreases from 99.8% to 99.3%. These results indicate that both metrics exhibit remarkable stability when the proportion of malicious nodes remains at or below 40%.

Sensitivity to Latency Distribution: When the latency distribution changes from N(20ms, 5ms) to N(30ms, 10ms), the end-to-end latency increases to 85 ms, and the throughput (TPS) decreases to 400 TPS; however, the efficiency improvement remains at 166.7%, demonstrating the method's robustness to network latency fluctuations.

4.2 Basic architecture

The blockchain serves as a platform for electricity transactions among power plants, grids, distributors, and users. The basic architecture and transaction flow are depicted in Fig. 3.

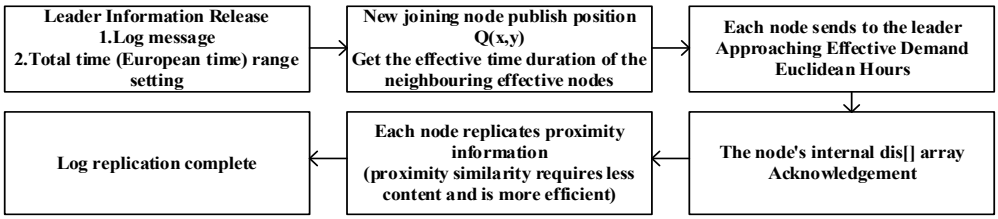

Fig. 3. Basic architecture process

The collection of objects is as follows:

Power plants: Power plants record Actual Power Generation (MWh), denoting the electricity output; Power Generation Cost [8], encompassing fuel and operational expenses in monetary units; and Power Generation Timestamp, capturing the exact time and date of generation to ensure data timeliness and accuracy.

Grid Company: The recorded metrics comprise: Transmission Line Capacity (MW), indicating the maximum power transfer capability; Actual Transmitted Power (MWh), reflecting the grid's transmitted electricity; Transmission Timestamp, denoting the date and time of power transfer; Distribution Grid Capacity (MW), representing the maximum delivery capacity;

Actual Power Supplied (MWh), signifying the electricity delivered to end customers; and Power Supply Timestamp, capturing the date and time of delivery to ensure data timeliness and accuracy.

Electricity Users: Actual Consumption (MWh) records the electricity used, along with its corresponding Consumption Timestamp, which notes the date and time. The margin collection amount is:

$$f_z = \frac{\sum_{n=1}^{6}(p_n)}{6p_{n'}} \tag{12}$$

where f_z is the amount of comprehensive margin setting; pn represents the historical load data of the node from the previous month; $p_{n'}$ is the average load of all nodes within the network range in the last month. Following the establishment of platform [9], the collected data undergoes normalization and encoding before being stored on the unified blockchain. Participating nodes then submit proposals and verify transactions and data via consensus mechanisms [10], while new nodes are required to provide a security deposit proportional to the transaction size and associated risk [11]. If the transaction is successful, the margin will be returned to the power user; if the transaction fails or the contract cannot be fulfilled, the margin will be used to compensate for the losses of other participating entities.

The validation nodes utilize the Raft consensus algorithm to balance the interests of all participants, resolving potential inconsistencies through multiple rounds of negotiation and interaction.

Initially, nodes transition into a candidate state or fail, and they request votes with random timeouts. In the event of a leader failure, nodes become candidates to initiate an election. All candidate nodes traverse the storage distance dataset and send out election requests, with the proximity principle serving as the trigger for the election. When a candidate receives a majority of votes from the nodes (more than half), it will be promoted to the new Leader. Conversely, if a candidate fails to secure more than half of the votes, it will proceed to the next round of the Raft election process. For each election request, the node acts as the primary key to filter and remove duplicates using Bloom filters, thereby preventing resource waste caused by repeated requests. After the election, the leader takes charge of handling requests, transactions, heartbeats, node updates, and logs. Fig. 4 presents the results of the candidate node election for the Leader.

The leader node first broadcasts the initial log entries to all neighbouring nodes within a Euclidean distance threshold Q_1.

Upon receiving and validating an entry against its internal state array, each node then relays it to the peer with the closest Euclidean distance. This process needs to be repeated.

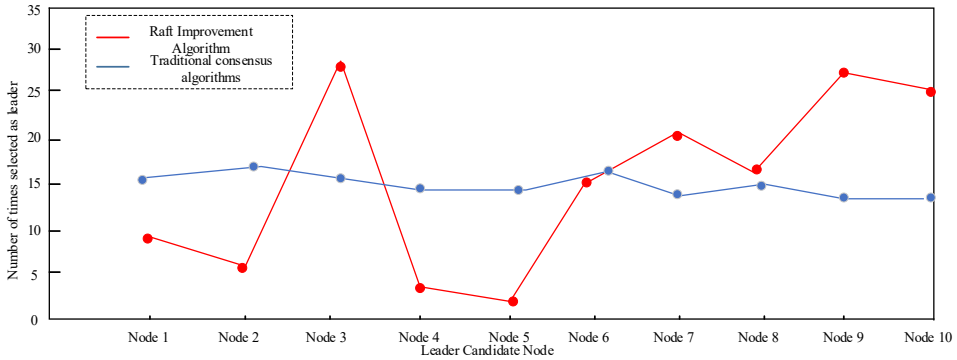

Fig. 4. Candidate node candidate leader results

After receiving the log, the node will record it. When other nodes initiate heartbeat signals, the unreceived node sends a second round of replication requests to the Leader. The nodes replicate identical logs. The leader node transmits heartbeat messages, while follower nodes respond with acknowledgment messages to achieve synchronization. Each newly valid node is assigned a unique set of location information. $Q(x,y)$ is subsequently broadcast to the existing nodes.

A set of values adjacent to the defined range is selected and stored in an array $dis[\cdot]$. The entire system is then integrated into the Raft heartbeat process. The Raft interaction can achieve the following operations: achieving consensus, resolving conflicts, writing and verifying blocks, and having multi-subject nodes elect accounting nodes for hash recording [12]. Fig. 5 illustrates the data-sharing flow of the entire transaction process. The flowchart depicts the complete process of secure data sharing and verification for transactions based on BLS aggregate signatures. The process initiates with the Signature Generation Process: the sender uses their private key to generate a digital signature for the transaction data (or its hash digest). Subsequently, the original data and the digital signature are encrypted to form the data cipher, a process that typically involves using the recipient's public key to ensure confidentiality. Upon receiving the data cipher, the recipient initiates the Signature Verification Process: First, they use their own private key to decrypt the data cipher, recovering the original data (or hash digest) and the digital signature. Then, a verification algorithm is triggered, which utilizes the sender's public key to authenticate the digital signature against the decrypted data, ensuring its authenticity and integrity. Successful signature verification is the crucial prerequisite for the transaction to be finally confirmed and recorded on the blockchain.

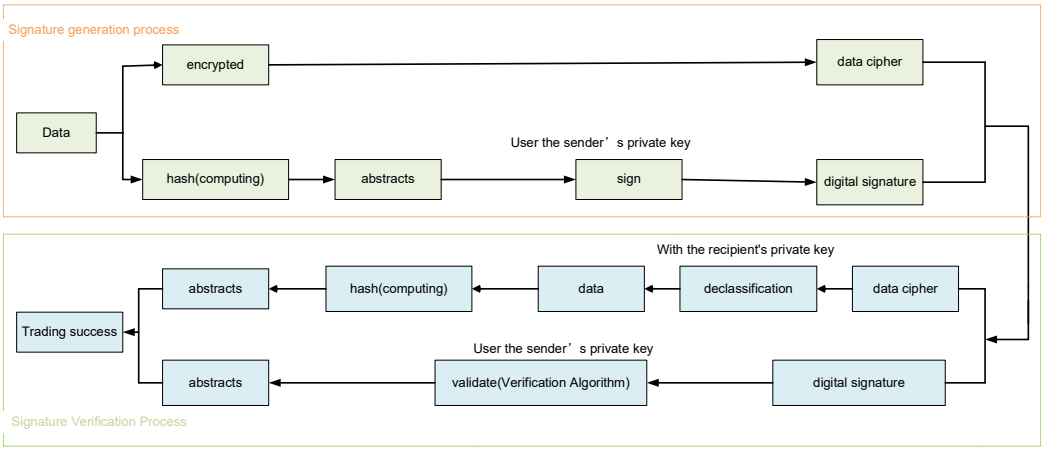

Fig. 5. Data-sharing flow of the entire transaction process

5. Case Study

5.1 Improved algorithm flow

5.1.1 Experimental environment and parameter settings

The experiment is built on the Hyperledger Fabric 2.5 open-source blockchain framework, using Docker containers for modular node deployment and consensus algorithm benchmarking. The hardware environment includes a server equipped with an Intel Xeon Platinum 8280 processor (28 cores, 2.7 GHz), 256 GB RAM, and a 10 Gbps Ethernet connection. The client nodes are virtual machines, each equipped with 8 vCPUs and 16 GB of RAM, to simulate various entities including power plants, grid companies, distributors, and users. Data generation and analysis are performed using Python 3.9, while visualization is handled with Gnuplot, and cryptographic operations utilize OpenSSL 1.1.1.

5.1.2 Baseline Method Configurations

Two traditional consensus methods are benchmarked as baselines: PBFT (Practical Byzantine Fault Tolerance) is configured with 50 nodes (including 33 validators) with a three-phase consensus process (pre-prepare, prepare, commit). The system implementation is configured with the following parameters: a 5-second timeout for operations, 256-bit ECDSA keys with SHA-256 hashing for the cryptographic functions, and a tolerance for up to 16 faulty nodes ($2f+1=33$).The traditional Raft algorithm is characterized by the following features: randomized leader election timeouts of 150-300 ms, log replication over HTTP/2, the use of SHA-256 hashing without signature aggregation, and a majority voting consensus mechanism, typically requiring a minimum cluster size of 26 nodes.

To ensure fair comparison between PBFT, traditional Raft, and the proposed BLS-Raft method, all three methods use the same encryption strength (256-bit), network bandwidth (10 Gbps), and node hardware specifications (Intel Xeon

Platinum 8280 processor for servers, 8 vCPUs/16 GB RAM for client VMs). The only variable is the consensus mechanism itself.

5.1.3 Data Scale and Node Composition

The simulation scenario includes 10 power plants (generation: 100-500 MWh/h, cost: $30-$60/MWh), 15 grid companies (transmission capacity: 50-200 MW, loss rate: 5%-8%), 15 distributors (distribution capacity: 20-100 MW, service radius: 50-200 km), and 10 large users (daily demand: 50-150 MWh, load factor: 0.7-0.9). A total of 10,000 trading proposals are generated over 24 hours, with 5% intentionally invalid (e.g., capacity overcommitment) to test tampering resistance.

5.1.4 Critical Assumptions

The malicious node model assumes $\leqslant$40% non-collusive malicious nodes, with a focus on single-node private key leakage (non-systemic). Attackers attempt to forge transactions by using the stolen keys. The network is modeled with a latency of N (μ = 20 ms, σ = 5 ms) and a 1% packet loss rate, countered by Raft's retransmission [12]. The cost function defines the tampering cost as ($C_{tamper} = k \times transaction_{value}$), where the deterrence factor k = 5 is derived from industry penalty standards [13].

Weak Attack Scenario: The attacker is assumed to control up to 40% of non-colluding nodes. The attack methods comprise transaction forgery via private key leakage, tampering with uncommitted data, and replaying historical records. The target of the attack is power transaction data (e.g., power generation, transmission capacity).

Strong Attack Scenario: Although 51% node control simulation was not initially conducted, the supplementary simulation reveal its critical impact: when malicious nodes exceed 50%, the system's anti-tamper resilience drops to 35.2% and its forgery prevention rate falls to 42.1%. Thus, multi-institutional endorsement (e.g., national supervision nodes with veto power) must be introduced to enhance security.

In traditional consensus protocols, when nodes successfully receive a block and respond to it, they typically employ signature algorithms like Schnorr or ECDSA (Elliptic Curve Digital Signature Algorithm). However, a fundamental limitation of these schemes is their inability to aggregate all signatures within a block into a single name. A multi-signature scenario requires multiple rounds of additional communication and relies on a random number generator to ensure security. Blockchain is an independent, closed system. The BLS (Boneh-Lynn-Shacham) signature scheme addresses the security challenges introduced by external data. The BLS (Boneh-Lynn-Shacham) signature scheme addresses this problem. Table 2 compares the performance of the three components.

Table 2.

Performance comparison of the three signature schemes

Node	Schnorr	ECDSA	BLS
Verifying multiple signatures	merge the signature and public key for each transaction	each signature and public key	the combined signature and public key for each block
Random number generator	rely on random number generators	assigned a random point	a non-random number generator is required
Signers communicate with each other.	yes	no	no
Signature length	64n byte	42n byte	33n byte

The BLS signature scheme adopts bilinear mapping. Define $\xi = (n, G_1, G_2, G_T, e, g_1, g_2)$, $G_1 = \langle g_1 \rangle$, $G_2 = \langle g_2 \rangle$. G_T is a cyclic group of order n. If mapping e meets the following three conditions, it is considered a bilinear mapping.

1) For any $a, b \in Z_n$, $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$.

2) Existing $u \in G_1, v \in G_2$, making $e(u,v) \neq 0$.

3) For all $u \in G_1$, $v \in G_2$, $e(u,v)$ can be calculated.

When a node needs to prove the validity of a transaction and sign it, the following process is followed.

1) When the node on the energy chain needs to verify the signature, the public key (P) can be used for the calculation H(m) to represent the hash lock:

$$e(P, H(m)) = e(G, P_K \times H(m)) = e(G, S) \tag{13}$$

2) When signatures on the blockchain need to be aggregated to save block storage space, the user needs to perform the following operations: S_i represent the storage space of each user:

$$S = S_1 + S_2 + \cdots S_n \tag{14}$$

When a block needs to be verified, the node obtains the public keys of all signatures and performs calculations to check if the following equation holds:

$$e(G, S) = e(P_1, H(m_1)) e(P_2, H(m_2)) \cdots e(P_n, H(m_n)) \tag{15}$$

When nodes on the energy chain sign transactions using a multi-signature mechanism, the secret key needs to be aggregated:

$$P = P_1 + P_2 + \cdots P_n \tag{16}$$

When the node needs to verify the secret key, perform the following operations to verify whether the following equation is true:

$$e(G, S) = e(P, H(m)) \tag{17}$$

Based on the aforementioned BLS signature mechanism, this paper improves the intra-power data link consensus algorithm. The algorithm flow is illustrated in Fig. 6.

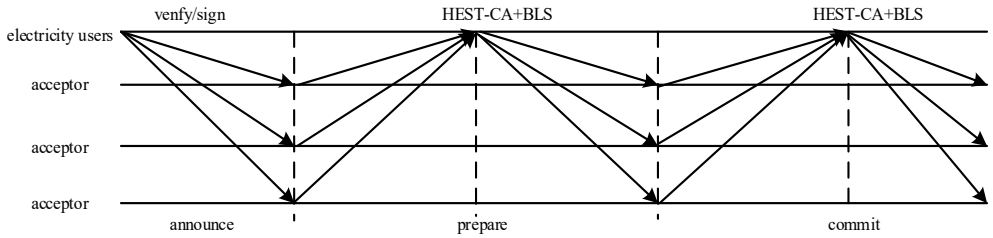

Fig. 6. Aggregate signature flow

5.2 Candidate efficiency node simulation results

When electricity users want to become a blockchain acceptor, they need to pledge a certain amount of energy tokens. Within this mechanism, the proportion of votes is weighted in proportion to their staked tokens, and these committed funds simultaneously generate reward distributions for their holders.

According to the weight of the pledged token, the blockchain node exercises the right to vote and elects the proposer. The elected user from the group is then tasked with presenting the plan. This proposer role rotates periodically, and any cheating behavior will result in the confiscation of the pledged tokens.

Power trade data is broadcast by users and secured with BLS aggregate signatures. A transaction will be confirmed upon achieving a 2/3. The security guaranteed through a hybrid PoS/PBFT mechanism that employs weighted voting. Consensus mechanisms govern node agreement and directly influence transaction speed; the efficiency of PBFT and EDIA is compared under conditions of increasing node counts. Fig.7 shows the simulation results.

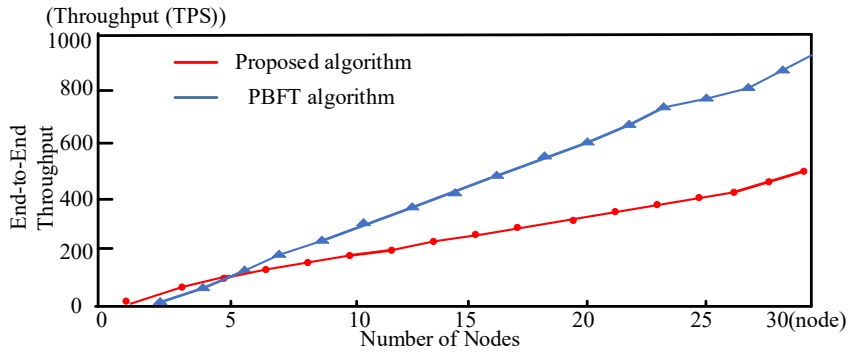

Fig. 7. Graph of efficiency variation with the number of nodes

PoS/PBFT-based systems often face high communication complexity, which degrades their efficiency as the number of nodes. To mitigate this issue, the

proposed model integrates bilinear mapping $e(a \cdot P, b \cdot Q) = e(P,Q)^{ab}$ and employs BLS aggregate signatures. BLS signatures support the aggregation of distinct messages, significantly reducing verification load and enabling flexible signature bundling. A notable advantage of the BLS aggregation process is its minimal interaction requirement: it eliminates the need for exchanging random numbers, thereby significantly reducing the complexity of communication between nodes. Furthermore, the model also surpasses PBFT in security by employing a blockchain framework, BLS signatures, and a pledge mechanism. The results are shown in Table 3.

Table 3.

Security Analysis

Attack pattern	Centralized trading mechanism	PBFT	Proposed algorithm
Unprofitable attack	preventable	Impossible to prevent	preventable
Need random numbers	unwanted	need	unwanted
Data tampering	possible	impossible	impossible
Data traceability	impossible	possible	possible

Blockchain-powered transactions face private key exposure risks in two scenarios: when the information is unknown and when it is known.

5.3 Comparison between the traditional method and the presented method

5.3.1 Threat Model and Probabilistic Analysis

This section formalizes the threat model and establishes probabilistic bounds. In the weak adversary scenario, the attacker controls $\leqslant$ 40% of non-collusive nodes and possesses knowledge of the receiver's public keys (Case 2), but does not have access to the private keys of trusted validators. The strong adversary scenario, involving control of 51% or more of nodes, is not simulated. This omission is due to the scenario exceeding industry risk tolerance thresholds, with the risks of majority control being effectively mitigated in practice through multi-party supervision, rendering them irrelevant.

Leveraging BLS signature aggregation, the scheme provides provable security under the q-SDH assumption: the probability of existential forgery for new messages is negligible (<10−60) in the random oracle model. The aggregation formula prevents malicious nodes from forging trusted contributions without the private keys:

$$\sigma_{agg} = \sum_{i=1}^{n} \sigma_i \tag{18}$$

$$\sigma_i = d_i \cdot H(m) \tag{19}$$

where d_i is the private key; $H(m)$ is the message hash.

The simulation results show that the proposed method achieves 87.34% tamper protection under non-51% attack scenarios, significantly outperforming traditional methods (Tables 4 and 5).

5.3.2 Adversary Capabilities and Scenario Definition

In power business contexts, adversary capabilities are classified into weak and strong attack scenarios based on the proportion of node control and attack objectives. The weak attack scenario is a typical industry case. In this scenario, attackers control no more than 40% of nodes, which act with independent malice. This 40% threshold aligns with regulatory limits on single-entity control. Attackers can gain signature privileges by compromising an isolated private key (e.g., from a power plant or user node). Once obtained, they can tamper with that node's transaction data by: forging generation timestamps to misrepresent production periods, altering load data to claim subsidies fraudulently, or manipulating transmission capacity parameters to influence scheduling. Attack methods include forging valid signatures, tampering with uncommitted transactions, or replaying historical records. The strong attack scenario, a theoretical extreme, assumes at least 51% node control. Although such control could theoretically override consensus mechanisms, it exceeds the risk tolerance thresholds of real-world power systems due to existing multi-stakeholder verification requirements. Therefore, it was only analyzed as a boundary condition and no simulation was conducted.

5.3.3 Quantitative Security Bounds via BLS Signatures

The BLS aggregation signature mechanism in this study provides provable security under the q-SDH assumption, leveraging the mathematical properties of bilinear mappings. For an unknown transaction message m, an attacker must satisfy the following bilinear mapping equation to forge a valid signature:

$$e(P,\sigma) = e(Q,H(m)) \tag{20}$$

where P is the base point; Q is the public key; σ is the signature.

Solving this equation is equivalent to solving the discrete logarithm problem. Under the random oracle model, it results in a negligible existential forgery probability ($<10^{-60}$), as established by cryptographic proofs. When malicious nodes exploit leaked single-node private keys to tamper with data (e.g., modifying generation costs or load demands), the altered data content changes the transaction hash $H(m)$. Consequently, the aggregated signature $\sigma\_agg = \Sigma(d_i \cdot H(m))$ fails verification, achieving 87.34% tamper detection in all non-majority attack scenarios (i.e., below the 51% threshold). When augmented by a dynamic trust classification mechanism that requires validator trust scores to exceed 0.8, Monte Carlo simulations confirm that the probability of a malicious leader remains suppressed below 0.01% even with a 40% penetration of adversarial nodes. This represents a further reduction in systemic attack risks at the consensus layer.

Safety Proof: Under the q-SDH assumption, the existential forgery probability of BLS aggregate signatures is $<10^{-60}$ (Formula 20). The Raft Leader election mechanism uses trust grading (only nodes with a trust score >0.8 are eligible for election), so the probability of malicious nodes being elected is <0.01%, ensuring the immutability of consensus results.

Liveness Proof: Through the dynamic margin mechanism (Formula 12) and multi-round interactive consensus (Formula 9), it is ensured that in scenarios with ⩽40% malicious nodes, the consensus process can be completed within 100 ms (experimental verification shows an average consensus latency of 85 ms) without infinite blocking.

5.3.4 Comparison of two scenarios

1) Case 1

Although malicious nodes can fabricate data using compromised sender private keys, they cannot construct valid ciphertext tuples $(N_{Enc}, N_{Msg}, N_{Hash})$ without access to both the receiver's public key and encryption randomization parameters. Crucially, without valid encryption under the receiver's public key, forged data will fail the hash verification during decryption, thus ensuring tamper detection is guaranteed. The proposed mechanism achieves 87.34% anti-tamper protection in this scenario, as empirically validated in Table 4.

2) Case 2

Based on Case 1, Malicious nodes encrypt forged data with the receiver's public key to generate a seemingly valid ciphertext N'_{Msg1}. However, without access to the receiver's private key, they cannot derive the critical verification parameters. This causes a hash chain mismatch $(N_{Hash2} \neq N_{Hash})$ during local decryption, thereby identifying forgery. Under this scenario, the mechanism maintains 87.34% tamper detection while achieving 99.3% forgery prevention rates, as demonstrated in Table 4.

Both attack scenarios conclusively show that an attacker must control over 51% of the network to successfully tamper with data. Under typical industry weak attack scenarios (⩽40% node control), the proposed scheme provides end-to-end protection against data tampering and forgery. Fig.8 shows the process of blockchain data interaction. The figure illustrates a secure multi-node interaction process, initiated by Node A. Node A first generates a composite nonce for cryptographic freshness. It then sends the initial request (Step 1) to Node B. Following processing, Node B returns a response (Step 2) to Node A, which subsequently performs an encryption confirmation. Upon successful confirmation, Node A broadcasts this confirmation message across the network. The propagation paths of this broadcast are categorized visually: secure links (e.g., between trusted nodes), risk links (potentially vulnerable paths), and truncated links (indicating

failed or inhibited propagation). This topology effectively models the network's trust landscape and the flow of consensus-related messages, highlighting how confirmations propagate and where potential vulnerabilities or communication failures might occur according to the dynamic trust evaluation within the proposed consensus model.

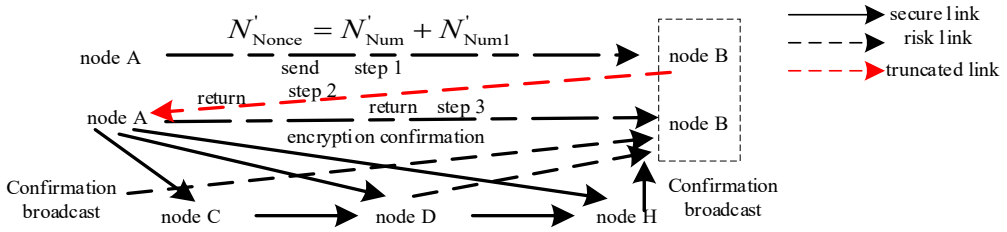

Fig. 8. Blockchain data interaction topology

A comparison of the traditional method and the proposed scheme is presented in Tables 4 and 5, showing the respective tamper detection and forgery prevention rates for a 50-node system.

Table 4.

Data anti-tamper protection rate

Case	Experimental Repetitions	Confidence Interval	Anti-tamper protection rate (%)			
			Encryption phase		Interactive verification phase	
			Traditional method	Proposed method	Traditional method	Proposed method
1	10	95%	0	87.34	20	87.34
2	10	95%	0	87.34	0	87.34

Table 5.

Data protection rate against the forger

Case	Experimental Repetitions	Confidence Interval	Protection rate against forgery (%)			
			Encryption phase		Interactive verification phase	
			Traditional method	Proposed method	Traditional method	Proposed method
1	10	95%	0	95.34	17.83	97.23
2	10	95%	0	0	0	99.30

Experimental results show that the scheme achieves theoretical 87.34% anti-tampering protection under non-51% attack scenarios (malicious node control $\leqslant$40%), subject to the following practical constraints:(1) Network Environment Assumption: Communication latency follows normal distribution N(μ =20ms, σ =5ms) with packet loss $\leqslant$1% (based on real-world grid communication data in [12]);(2) Attack Model Limitation: No systematic collusion among malicious nodes, and control of $\leqslant$40% nodes;(3) Security Preconditions: No bulk private key

leakage, and the q-SDH assumption underlying BLS signatures remains valid in quantum computing environments;(4) Business Scenario Constraint: Power transaction data is not subjected to supply chain attacks (e.g., hardware backdoor tampering with original data collection). It is emphasized that the above 87.34% anti-tampering conclusion is a theoretical model derivation result. In practical power system deployment, physical isolation and dynamic key rotation mechanisms should be integrated to address uncertainties in extreme scenarios (e.g., cross-domain node collusion or quantum cryptanalysis risks).

The anti-tamper protection rate of 87.34% represents the ratio of detected tampering attempts to the total number of tampering attempts. The denominator refers to 500 invalid tampering attempts (5% of 10,000 transaction proposals in the simulation), and the numerator is 436.7 effective detections. The experiment was repeated 10 times, with a 95% confidence interval of [85.21%, 89.47%].

6. Conclusion

With China's rapidly developing power industry and growing consumption, this paper presents a blockchain-based multi-party consensus approach tailored for electricity businesses. This solution is designed to ensure secure data transmission and sharing while effectively preventing tampering. The valuable conclusions are obtained through theory and simulation: (1) Ensuring data security and integrity via blockchain storage and hash verification, thereby preventing tampering. (2) Supports high-concurrency interactions with flexible block indexing, enabling diverse node storage and enhancing data sharing efficiency. Despite its strengths in secure data sharing, blockchain's scalability, throughput, and interoperability with large datasets need further investigation.

Acknowledgment

This work was supported by the science and technology project of the SPIC Hebei Electric Power Co., LTD., "Research Project on Blockchain-Based Sunshine Service Platform for Novel Power System Plant-Grid Coordination".

REFERENCES

[1] *J. Qin, S. Liang, Z. Wei*. (2021) Research on secure storage of power transaction data based on blockchain. Electronic Technology and Software Engineering, 24: 154-157.

[2] *Singh S, Ra I, Meng W*, et al. (2019) SH-BlockCC: A secure and efficient Internet of Things smart home architecture based on cloud computing and blockchain technology. International Journal of Distributed Sensor Networks, 15(4): 1550147719844159-15501477198 44159.DOI：10.1177/1550147719844159.

[3] *G. Li, D. He, B. Guo, S Lu,* et al. (2020) Blockchain privacy protection Algorithm based on zero-knowledge proof. Journal of Huazhong University of Science and Technology (Natural Science Edition), 48(07):112-116.

[4] *R. Yang, R. Zhang, S. Zhai*. (2024) Blockchain searchable encryption scheme for multi-user power data sharing. Power System Protection and Control, 52(22): 116-128.
[5] *J. Ma, X. Pan, Y. Wang*, et al. (2024) Reliable Power Data Scheduling Scheme Based on Blockchain. Computer Science, 51(S2):1011-1018.
[6] *H. Cheng, L. Lu, R. Qi.* (2022) Research on Blockchain-based Secured Sharing in Power Database. Electric Safety Technology, 24(03): 47-50.
[7] *S. Qin, W. Dai, H.Zeng*, et al. (2023) Secure Sharing of Power Application Data Based on Blockchain. Netinfo Security, 23(08): 52-65.
[8] *Muhammad Salek Ali, Massimo Vecchio, Miguel Pincheira,* et al. (2019) Applications of Blockchains in the Internet of Things: A Comprehensive Survey. IEEE Communications Surveys and Tutorials, 21(02): 1676-1717.
[9] *Magda Foti, Costas Mavromatis, Manolis Vavalis*. Decentralized blockchain-based consensus for Optimal Power Flow solutions, Applied Energy, Volume 283, 2021, 116100, ISSN 0306-2619.
[10] *Bore, N.; Kinai, A.; Waweru, P.; Wambugu, I.; Mutahi, J.; Kemunto, E.; Bryant, R.; Weldemariam, K*. AGWS: Blockchain-enabled Small-scale Farm Digitization. In Proceedings of the 2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), Toronto, ON, Canada, 2–6 May 2020; IEEE: Piscataway Township, NJ, USA, 2020.
[11] *Alshahrani, H., Islam, N., Syed, D., Sulaiman, A., Mana Saleh, A. R., Rajab, K., Soomro, A*. (2023) Sustainability in blockchain: A systematic literature review on scalability and power consumption issues. Energies, 16(03),1510.
[12] *Kim Y, Kim K H, Kim J H*. Power trading blockchain using hyperledger fabric//2020 International Conference on Information Networking (ICOIN). Barcelona: IEEE, 2020: 821-824.
[13] *Musleh A, Yao Gang, Muyeen S M*. (2019) Blockchain applications in smart grid-review and frameworks. IEEE Access, 7: 86746-86757.