Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 4314734, 13 pages
https://doi.org/10.1155/2022/4314734

Research Article

Blockchain Consensus Mechanism for Distributed Energy Transactions

Jiangyao Wu,[1] Ye Liu,[1] Jiefei Cai,[2] and Shuhui Su[2]

[1] *Guangdong Power Grid Co., Ltd., Guangzhou, Guangdong 510600, China*
[2] *China Southern Power Grid Digital Grid Research Institute Co., Ltd., Guangzhou, Guangdong 510630, China*

Correspondence should be addressed to Jiefei Cai; caijf1@iotosaas.com

Received 2 April 2022; Revised 10 June 2022; Accepted 29 June 2022; Published 15 July 2022

Academic Editor: Jun Ye

In order to reduce the cost of grid dispatching and increase the transparency of energy transactions, the distributed energy transaction model based on blockchain is constructed. At the same time, in order to improve the high communication overhead and low throughput of the traditional PBFT algorithm in the consortium blockchain, an efficient Byzantine fault-tolerant consensus mechanism (DE-BFT) for the energy blockchain is designed. The algorithm improves from two aspect: *node election* and *main chain consensus*. In the stage of *node election*, the model uses a health score evaluation and a verifiable random function to improve the security and randomness of node selection. In the stage of *main chain consensus*, the efficient data consistency interaction protocol decreases the complexity of the communications between nodes, down to a constant term level from exponential one. The result shows that, compared with other consensus algorithm, the DE-BFT algorithm performs better in terms of consensus delay, communication overhead, throughput, and consensus node reliability.

1. Introduction

Distributed energy transaction [1] has the characteristics of long business process, many participants, and wide distribution. These characteristics lead to serious data silos, difficult credit transfer, and prominent transaction risks in distributed energy transactions. Blockchain is famous for its openness, transparency, traceability, tamper resistance, and decentralization. Exploring blockchain-based energy-distributed transactions can help reduce grid dispatch costs and increase transaction transparency. According to the degree of decentralization, blockchain is divided into three modes: *public blockchain* [2], *consortium blockchain* [3], and *private blockchain* [4]. The multicenter characteristics of *consortium blockchain* are more suitable for the status quo of power reform in China. Therefore, most of the distributed energy transaction research based on blockchain currently adopts the application mode of *consortium blockchain* [5–8]. Consensus algorithm is the key technology that determines the performance and security of blockchain systems. Compared with the traditional *Byzantine fault tolerance (BFT) algorithm* [9], the *practical Byzantine fault tolerance (PBFT) algorithm* [10] reduces the overhead execution of network, which makes it becomes more practical. However, PBFT still has some problems, such as security loopholes in the selection of consensus nodes and excessive communication overhead in the case of multiple nodes, which have become important factors restricting the development of blockchain.

Distributed energy transaction platforms requires lower latency and higher security. Therefore, the performance of the consensus algorithm is an important factor affecting the development of energy blockchain. Literature [11] proposed an effective real-time distributed blockchain consensus algorithm for energy transactions. The algorithm manages a large number of transactions in partitions to ensure the real-time improvement of system transactions but manages multiple systems at the same time, so the security of the system cannot be guaranteed. Literature [12] proposed *delegation Byzantine fault tolerance (DBFT) consensus* *algorithm* to improve the efficiency of energy transactions. Compared with PoW and PoS, DBFT solved the efficiency problem. Although the algorithm verifies the security of transactions through entrustment, it increased the complexity of communication, and the complexity is $O(n^2)$. The security is not as high as PoW and PoS. Therefore, literature [13] proposes an energy trade deal algorithm based on the blockchain mechanism of consensus. First of all, according to PBFT, the lack of a dynamic problem in the VPBFT voting mechanism was introduced. The node system is divided into four types with different responsibilities and gives the number of relations between nodes. When the number of nodes is changed, it can be calculated according to the quantity relation, ensuring dynamic. Second, a data anonymous transaction and authentication protocol are designed. In the protocol, when the seller sells data, the mapping relationship between the real identity and the false identity of the data owner is blinded and sent to the buyer. When the buyer wants to verify their identity, the seller's identity can only be verified with the authentication of the blockchain. The complexity of the protocol is also $O(n^2)$, resulting in insufficient real-time performance of the protocol. Literature [14] proposed a secure energy transaction method based on blockchain consensus. The *Byzantine general problem (BGP)* protocol is used for developing transactions, by reducing the number of system attacks to ensure the safe operation of the system. However, the efficiency of the algorithm decreases rapidly with the increase of the number of nodes and the communication complexity is $O(n^2)$, so the real-time performance of the system is insufficient. Literature [15] proposed PBFT protocol, an energy-efficient consensus node selection mechanism is designed, and VRF is used to ensure the security of leader. In addition, in the case of multihop neighbor nodes, the authority of the node is evaluated by selecting the relay node by extending the centrality, which may lead to centralization of nodes' colluded interests, and communication complexity is the same as literature [14]. Literature [16] guarantees the fairness of transaction resource allocation by introducing the active reputation value of the entity. In the consensus process, the reputation is used to select the master node to reduce the traffic of duplicate nodes, and its communication complexity is $O(m * n)$ (where m is a constant). Since the selection of the master node must be very safe, the burden on the master node is increased. The above research found that the existing research results focus on the combination of blockchain and energy trading but ignore the real-time and security of energy trading. Research on an efficient consensus mechanism for distributed energy transactions is an urgent problem to be solved in energy blockchain research [17].

This paper comprehensively considers the security and efficiency of energy transaction scenarios based on blockchain and designs a distributed energy transaction mode based on blockchain. This mode improves the traditional PBFT algorithm from the two stages of block node election and main chain consensus. In the block node election stage, a health score evaluation mechanism is designed to reliably evaluate the consensus behavior of nodes. In order to improve the randomness of the node election process and the antiattack capability of the network, a verifiable random function is used to randomly elect candidate nodes and master nodes according to their health scores. In the main chain consensus stage, consensus is reached between nodes based on the efficient data interaction protocol provided by HSBFT [18], which further improves the transaction throughput of the distributed energy trading platform based on block chain and reduces the transaction delay.

2. Preliminary Knowledge

2.1. Verifiable Random Function. Verifiable random function (VRF) can generate specific outputs from specific inputs [19]. Its biggest feature is that it can verify that the output result is correct without knowing the input. So VRF is essentially a pseudorandom function with a verifiable function. If a specific value and private key are input, VRF outputs a random number and a proof by generating a function group. Combined with the public key, the verifier can use the proof function group to verify whether the random number is generated by the input. This process does not need to expose the private key of the input, so the VRF is safe. VRF contains the following two function groups [20].

2.1.1. Generating Function Group. Generating function: nodes use a generating function to generate a hash random output R and a hash proof P, respectively. SK is the private key of the node. M is a specific input value set by the system.

$$\begin{aligned} R &= \text{VRF\_Hash}(SK, M), \\ P &= \text{VRF\_Proof}(SK, M). \end{aligned} \tag{1}$$

2.1.2. Proof Function Group. PK is the public key of the verified node.

$$\begin{aligned} R &= \text{VRF\_P2H}(P), \\ &\text{VRF\_Verify}(PK, M, P). \end{aligned} \tag{2}$$

VRF satisfies three properties of verifiability, uniqueness, and randomness. Verifiability means that through the above steps, the verification node can still verify whether the R and P values are generated by the PK holder according to M without knowing the private key of the verified node. The uniqueness means that for any PK and M, there is a unique output R, and R can be verified. The randomness means that output R of VRF_Hash is distinguishable from the random number M.

Verifiable random functions have two characteristics:

(1) For different inputs, the output values are random and uniformly distributed within the range of values

(2) For the same input, the output it gets must be the same

The role of the VRF in the consensus mechanism of this paper is that even if the private key of the random number node is unknown, other nodes can verify that a certain random number is generated by the node that issued the random number. It can be verified that a random output is indeed generated by a specific node under the premise of not exposing the private key.

2.2. PBFT Consensus Algorithm. The PBFT consensus mechanism mainly achieves the consensus of all nodes through the consensus protocol and the view-change protocol. Among them, the view-change protocol is to replace the master node with a slave node when it cannot continue to perform its duties and to ensure that requests that have been executed by non-Byzantine servers will not be tampered with. The premise of PBFT to ensure security and activity is that the number of Byzantine nodes in the system does not exceed 1/3 of the total number of nodes in the system.

The PBFT algorithm is divided into three stages, namely, preprepare, prepare, and commit. Figure 1 is a flowchart of the execution of the PBFT algorithm. The detail is as follows:

(1) Request: the client sends a request to the master node

(2) Preprepare: after the master node receives the request sent by the client, it assigns a sequence number to the client's request and sends a preprepare message to all replica nodes

(3) Prepare: the replica node receives the preprepare message and verifies the authenticity of the message. After passing the verification, it broadcasts the prepare message to other replica nodes

(4) Commit: when the replica node receives $2f$ prepare messages as same as the above preprepare message, it steps into commit phase and broadcasts the commit message to all replica nodes

(5) Response: after the replica node receives $2f + 1$ the same commit messages, it replies with a corresponding message to the client. If the client receives $f + 1$ the same reply message, the request execution is completed

All nodes in the PBFT algorithm work under the same configuration information, which is called a view, and each view is uniquely determined by a master node. The master node has core capabilities such as serial number allocation and initiating proposals in the entire system. Other replica nodes participate in the voting process. When the master node fails, the view-change protocol will be triggered. The master node is going to be replaced and step into the next view stage. The PBFT consensus mechanism adopts the method of $p = v \bmod |N|$ for selecting master nodes. As the selection method is fixed, it is easily to be exposed at next round of master nodes and thus be attacked.

3. Blockchain-Based Distributed Power Transaction Model

Figure 2 shows a distributed power transaction model based on blockchain. The nodes is mainly classified as user nodes on the power consumption side, distributed energy nodes on the power generation side, power grid enterprises, data centers, and small power stations. The power consumption side and the power generation side calculate electricity data and release electricity purchase/selling plans to the blockchain trading platform rely on smart ammeter. After being matched, the transaction records are stored on the blockchain to ensure that each transaction is verifiable and traceable and cannot be tampered with.

The blockchain-based power distributed transaction is mainly divided into the following stages.

(1) Node initialization stage: each user or enterprise needs to go through the authorization and authentication of the central management node (the central management node is authorized by the power grid enterprise and is responsible for the identity authentication and authorization of the nodes in the blockchain) when it newly joins the blockchain network. At the same time, the central management node conducts authority and trust base evaluation and divides the trust base into three levels: A (high), B (medium), and C (low). Considering the advantages of high performance and high reliability of the data center compared with other nodes in the transaction model, the trust base of the data center is A. The trust base of small power stations is B as their performance are relatively weak even though endorsed by the power grid. For other distributed power generation nodes or user nodes newly joining the network, the trust base is set as C

(2) Smart contract initialization stage: all participating nodes in the power blockchain jointly build smart contracts. All nodes jointly agree on a certain smart contract including triggering conditions, response rules, and logical processes. The contract is signed by all parties with their private key, which ensures the validity, and uploaded to the blockchain network. Each node will receive a copy of the contract and save it in the memory at once, waiting for a new round of consensus phrase in the system, triggering the consensus execution of the contract. After reaching a consensus, the stage of the smart contract initialization is completed

(3) Transaction stage: before the transaction, the smart ammeter will conduct statistical analysis on the user's generation/consumption during this period and formulate an appropriate power purchase/sale plan for the user according to the analysis. This plan will release to a transaction platform. The smart contract carries out intelligent matching of this online transaction

(4) Transaction consensus on-chain stage: after the online transaction negotiation is completed, a consensus is reached among the distributed nodes and finally stored in the blockchain. A power transaction can be officially completed only after the consensus of the nodes

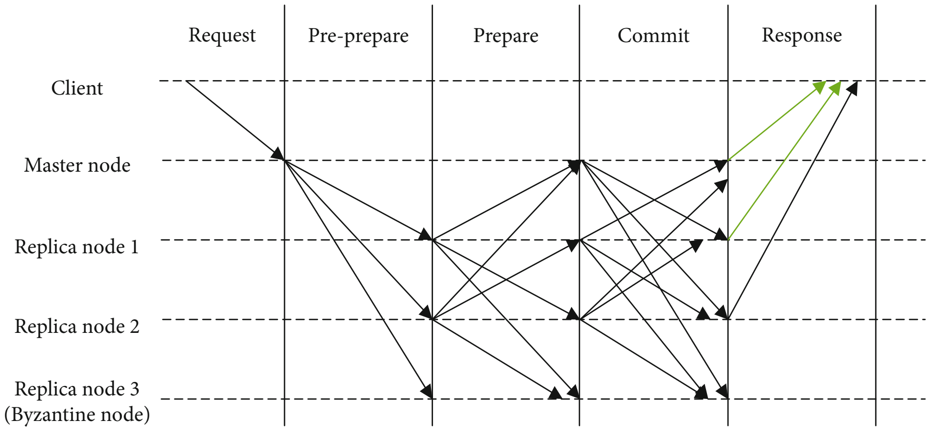

Figure 1: PBFT consensus consistency protocol.

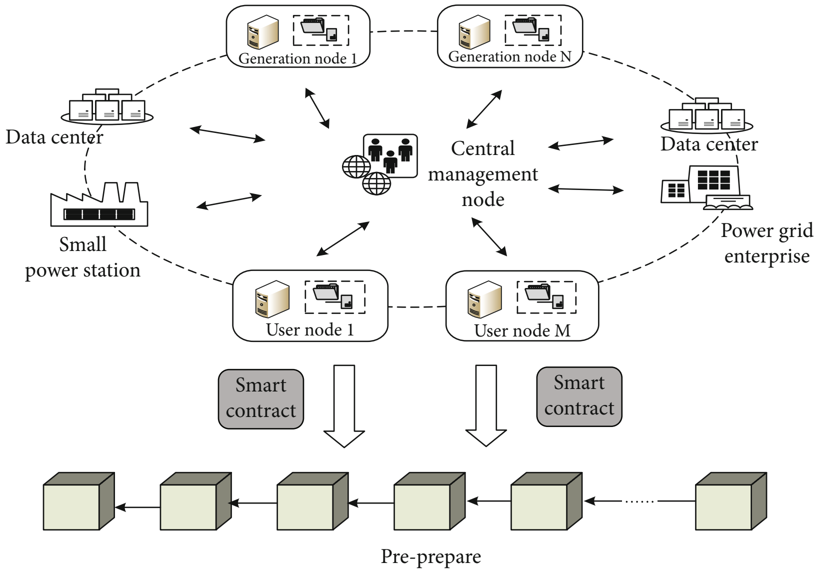

Figure 2: Blockchain-based distributed power transaction model.

The speed of the power transaction consensus on the chain directly affects the user's transaction experience and transaction efficiency. At the same time, in order to make the data finally stored on the blockchain reliable, the security and antiattack of the consensus process must be executed. Consensus failure will lead to transactions that cannot be carried out smoothly, hindering the normal operation of the power transaction model. Therefore, the consensus mechanism is crucial to the stability and safety of the blockchain-based power distributed transaction model.

4. DE-BFT Consensus Algorithm

4.1. Node Type and View Number. The distributed nodes participating in the consensus are divided into the following types according to their functions:

Master node: the master node is responsible for packaging transactions in the blockchain network and submitting new blocks

Candidate node: the candidate node is in charge of verifying and reaching consensus, the blocks of transactions generated during periods. It is also in charge of the election of the master node and consensus committee members

Follow node: follow node revolves the election of candidate nodes and copies the final block data on the blockchain

Byzantine node: it is a malicious node in the system, or a node exhibits malicious behavior due to being attacked. The malicious behavior of Byzantine nodes in the energy blockchain will cause transaction delays or untrustworthy transaction settlements, affect user experience, and hinder the development of power blockchain

Central management node: in power transaction model, as the joining and exiting status of each distributed power node changes dynamically, an authoritative and credible central node is required to manage identity authentication and to maintain the status of all nodes in the system. Based on the characteristics of the consortium chain, which ensures the reliability of the joining nodes. This paper

Table 1: Node status table.

Serial number	Trusted Base	Status	Health score	Acting as master node times
1	A	Master node	0.9	2
2	B	Candidate node	0.6	—
3	B	Candidate node	0.5	—
4	C	Follow node	0.4	—

introduces a new concept of a central management node. As an authority and leader in the power field, power grid companies are endorsed by national credit. Therefore, a branch office of power grid is generally in charge of the central management node. In order to prevent the centralization of power transaction system, the central management node does not revolve any power transactions such as transaction verification and block submission in the system

View number: it represents the position and status of the current master node. The view number is described as the term of $p-t$. p represents the number of the master node, and t represents that how many times that p is the master node. By this way, it ensures that no two views have the same view number throughout the entire period

4.2. Block Producer Election Mechanism

4.2.1. Health Score Evaluation Strategy. The health score evaluation strategy calculates the credibility of each node based on the behavior of the node in the process of participating in the block consensus. As part of the consensus protocol, the health score evaluation is deployed on the blockchain through smart contracts and can be performed in each node which participates in the consensus, and the evaluation results will be sent to the central management node. As shown in Table 1, the central management node maintains all nodes status in a node status table. Each time, the node status table is updated completely. The result will be synchronized to the blockchain, for all nodes to view. For each node newly added to the system, its initial health score value will be evaluated according to the analysis results of the trusted base of the CA certification center in the distributed power transaction model in this paper. If the trust base is B, the initial health is set to 0.5, and if the trust base is C, it is set to 0.4. For example, when a certified data center joins the network, considering its high performance and high reliability advantages compared to other nodes in the blockchain, the trust base of the data center is set as A; that is, the initial value of the health score is 0.6; if a small power station newly joins the network, since it is endorsed by the power grid, its trust base can be set as B; that is, the initial value of the health score is 0.5; similarly, when a user node joins the network, the trust base can be set as C, and the initial value of health score is set to 0.4 accordingly. The node health score evaluation algorithm is shown in Algorithm 1. The algorithm detail is described as below.

(1) Health Score Evaluation of Master Node. For the master node, if a new block is generated during the t^{th} round of consensus process of the current view which the current master node participates in, the health score of the master node will increase accordingly, but the health score value does not exceed the threshold of 1 set by the system. If the current view number of the master node has not changed, the growth rate of the master node's health score will be slower. In order to avoid centralized processing of the system, when the health score value is 1 and the number of blocks packaged by the master node reaches the threshold k, the view-change protocol will be triggered to select new master node.

$$H_i(t) = \begin{cases} \min\left\{1, H_i(t-1) + \dfrac{1}{k}[1 - H_i(t-1)]\right\} & \text{, Send consistent messages in a timely manner,} \\ H_i(t-1) \times \dfrac{1}{t} & \text{, Consistency messages not sent in time,} \\ 0 & \text{, Send inconsistency messages,} \end{cases} \quad (3)$$

$$H_i(t) = \begin{cases} \min\left\{1, H_i(t-1) + \dfrac{1}{k}[1 - H_i(t-1)]\right\} & \text{, Send consistent messages in time,} \\ H_i(t-1) \times \dfrac{1}{t} & \text{, Message not sent in time,} \\ H_i(t-1) \times \dfrac{1}{t+1} & \text{, The message sent is different from most,} \\ 0 & \text{, Send inconsistency messages.} \end{cases} \quad (4)$$

```
Algorithm: Health Score Evaluation Algorithm
Input (smart contract trigger condition): New node joins and a round of consensus ends.
Output: Node health score value.
1. When new nodes are added
        Trigger health score evaluation smart contract.
2.    If trust base = "A"
            health score =0.6
3.    If trust base = "B"
            health score =0.6
4.    If trust base = "C"
            health score =0.4
5.    End if
6. when a round of consensus ends / a new block is generated.
      Trigger health score evaluation smart contract
7.    If node type = primary
            If the master node sends messages consistently during the consensus phase
                Calculate the master node health score value according to formula (3) :
                    $H_i(t) = \min \{1, H_i(t-1) + (1/k)[1 - H_i(t-1)]\}$
            If the master node does not send the message in time.
              Calculate the master node health score value according to formula (3):
                    $H_i(t) = H_i(t-1) \times (1/t)$
            If the master node sends messages inconsistently
                    $\mathrm{H_i}(t) = 0$
                  Execute the view-change protocol
8.    If node type = candidate
            If candidate nodes send consistent messages
              Calculate the candidate node health score value according to formula (4):
                    $H_i(t) = \min \{1, H_i(t-1) + (1/k)[1 - H_i(t-1)]\}$
            If the candidate node did not send the message in time
                    $H_i(t) = H_i(t-1) \times (1/t)$
            If the candidate node sends a message different from other nodes
                    $H_i(t) = H_i(t-1) \times (1/t + 1)$
9.    End if
10. End
```

Algorithm 1: Algorithm description of node health score evaluation strategy.

The specific evaluation of the master node's health score is shown in formula (3), where $H_i(t)$ is the health score value of the t^{th} round of consensus and k is the threshold of the consensus round. If the master node does not send messages to the candidate node in time during the view process, resulting in no new block being generated, its health score will drop. If the master node sends inconsistent messages to other candidate nodes, its health score will drop directly to 0 and be kicked out of the candidate node set, triggering the view replacement protocol to replace the master node and view.

(2) Health Score Evaluation of Candidate Node. For a candidate node, if the same message is sent to other nodes during the t^{th} consensus process of the current view and it is consistent with the final consensus result, the node's health score will slowly increase, but again, it will not exceed the system setting threshold 1. As the number of consensuses in the same view increases, the rate of increase in health score also decreases. If a candidate node does not participate in the consensus process in a certain round, that is, does not send any messages to other nodes, its health score will be reduced in a certain proportion. If a candidate node participates in the consensus process but sends a message that is inconsistent with the final result, its health score will also decrease. The technical solution will reduce the health score value of nodes at different speeds according to the different behavior of nodes. If it is detected that the same consensus node has sent different information lists, the node will be regarded as a malicious node, its health score value will be reduced to 0, and it will be removed from the current candidate node set. The health score evaluation of candidate node is shown in formula (4).

4.2.2. Primary Node Election Protocol. The master node is generated in the candidate node set and follows the nodes whose health score exceeds 0.5 into the candidate node set. After the selection of candidate node is completed, each candidate node independently performs VRF calculation. If the calculated hash output meets the requirements for becoming the master node output threshold, then node broadcasts its own VRF output and proof and requests authentication from other nodes in the candidate node set. The specific verification process of the master node is shown in Figure 3.

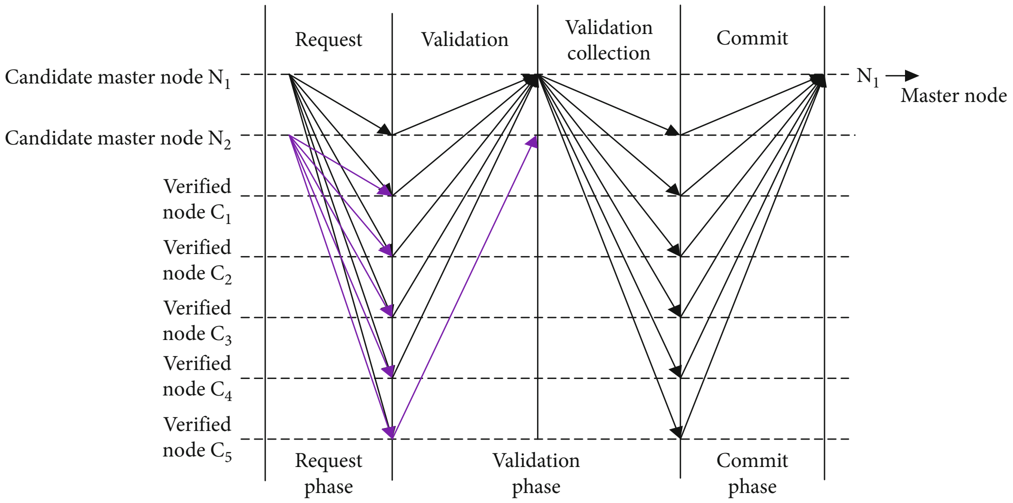

FIGURE 3: The process of the master node election protocol.

(1) Request Phase. The verifiable random function VRF hash outputs of candidate master nodes N_1 and N_2 are all within the threshold range set by the system, then these two nodes obtain their own proof values through the proof function, and then, N_1 and N_2, respectively, send the request verification message encrypted by digital signature to the consensus committee node. The format of the request message is

$$\langle \text{request}, R_i, P_i, T, H_i \rangle_{\text{sig}(N_i)(i=1,2)}. \tag{5}$$

(2) Validation Phase. In the validation phase, other candidate nodes start the timer when they receive the first validation request message. When the timer stops, they no longer receive any request validation messages and start to verify the received request validation messages. The validation process requires the following:

(a) Check whether the signatures of nodes N_1 and N_2 are correct; if not, delete the message directly

(b) Check whether the value sent by the node is correct; if not, delete the message directly

(c) Compare the health score values of all the verified nodes received during the timer period, and select the node with the highest health score as the master node

Assuming that in the above verification, the $C_1 - C_4$ verification nodes finally select N_1 as the master node, then a verification message will be returned to the node N_1. The format of the validation passed message is

$$\langle \text{verify}, T, m, \Phi \rangle_{\text{sig}(C_i)}, \tag{6}$$

where T represents the timestamp when the node verified the return message to the verified node, m represents the proof message digest that the node selects N_1, Φ represents the set of request messages received by the node, and sig(C_i) represents the digital signature of the consensus committee node, which is used to prove that the message is indeed sent by the node and cannot be tampered with by others.

(3) Validation Collection Phase. In this step, all candidate master nodes that meet the VRF output conditions and send the request verification message will receive the verification pass messages from other verification nodes and collect and package these messages. When the node receives a verification pass message from more than $f + 1$ different candidate nodes, it means that the node has obtained the approval of most nodes to be selected as the master node and records this status in its own local log, switches to the view of its elected master node, and modifies the number of view. Then, the master node returns a confirmation message to all other consensus nodes. The format of the message is

$$\langle \text{confirm}, T, \Psi \rangle_{\text{sig}(N_i)(i=1,2)}, \tag{7}$$

where Ψ is the set of verification pass messages received by the master node from other different nodes, as shown in Figure 3. N_2 only received the verification passing messages from C_5, so it cannot become the master node.

(4) Confirmation Stage. In the confirmation stage, after other consensus committee nodes receive the confirmation message from the N_1 node, they need to verify the correctness of all messages in the message set and verify whether the signature of the master node is correct. Consensus records the state change of the master node in the local log, switches to the view number where the current master node is located, and sends a confirmation message back to node N_1. After node N_1 receives the confirmation state change message from other nodes, it officially becomes the master node, assumes its role in the system at this stage, and begins to process incoming requests in the system. Algorithm 2 describes the algorithm of the master node election process.

```
Algorithm: Master Node Election Algorithm
Input: Candidate Node Set$\{N_1, N_2, \cdots, N_i\}$.
Output: Master Node Serial Number.
1. Trigger the smart contract to call a random verifiable function to generate a random number for this round of elections
2. The candidate node calls the random verifiable function to generate the random number of this round
                  $R_i = \text{VRF\_Hash}(SK, M)$
      Calculate:
                  $P_i = \text{VRF\_proof}(SK, M)$
3. If candidate node generates random numbers$R_i \leq v$.
      Broadcast the request verification message$\langle \text{request}, R_i, P_i, T, H_i \rangle_{sig(N_i)}$.
4. When the consensus node receives the first request verification message
      Timer start
      Verificatesig$(N_i)$
              If signature is incorrect
                Delete this message
      Verificate$R_i$, $P_\text{i}$
              If $R_i$, $P_\text{i}$is incorrect
                Delete this message
      End timer
      Compare < the health score value of all requesting verification nodes >
      Selectmax $\{H_i, i \in \{\text{A set of candidate nodes conforming to random numbers.}\}\}$
5. Broadcast master node serial number
6. End
```

Algorithm 2: Description of primary node election algorithm.

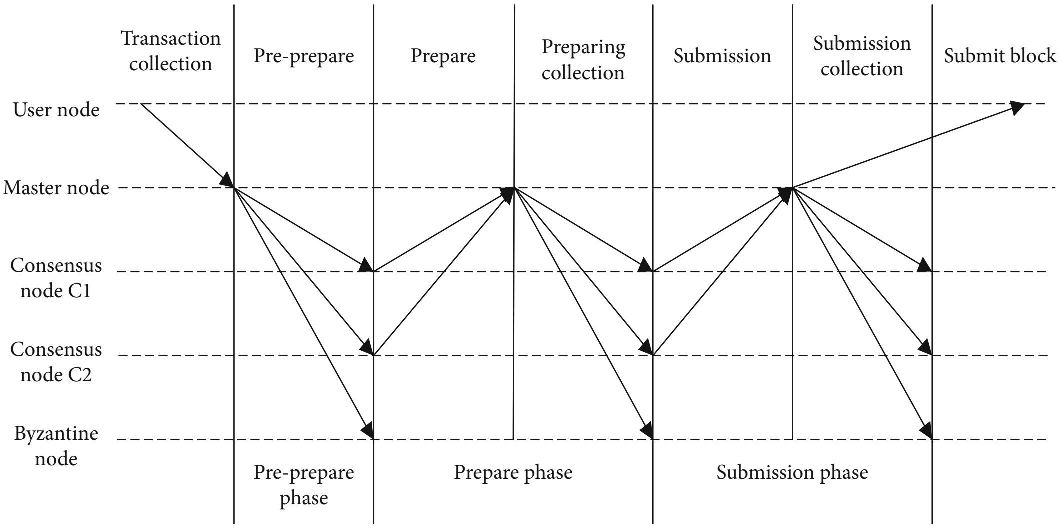

Figure 4: Data consistency interaction protocol.

4.3. Main Chain Consensus Protocol. In the main chain consensus stage of the traditional PBFT algorithm, the communication complexity between nodes is very high, which makes the Byzantine consensus algorithm difficult to apply in practical systems. HSBFT is an efficient data exchange protocol. In general, its communication complexity is at a constant level. Therefore, in our energy blockchain consensus mechanism, in order to improve the transaction confirmation speed, ensure the real-time information transmission, and provide a good user experience, this paper adopts HSBFT's efficient data consistency interaction protocol as the main chain consensus protocol. The specific process is shown in Figure 4.

The following describes the main stages of the consensus protocol operation process described above in detail:

(1) Preprepare stage: after the master node packages the transactions collected in the memory transaction pool into blocks, it multicasts a prepreparation message to the consensus node set, appends this message to its own local log, and starts timer $T1$. The consensus node verifies the message when it receives the prepreparation message. After the verification is passed, it officially receives the message, enters the preparation stage, and replies the preparation message to the master node

```
Algorithm: Main Chain Consensus Algorithm
Input: Transaction Collection
Output: New Block
1. The master node sends the packaged transaction set to the blockchain network
2. Broad coast pre-prepare message to consensus node
3. The master node starts the timer
4. When the consensus node receives the pre-preparation message
      Perform verification
      If verification passed
          Send prepare message back to master node
      If the number of messages received by the master node within the timer $\geq 2f$
          Then multicast a prepare message
      Else
          End this round of consensus
5. When the consensus node receives the prepare message
      Perform verification
        If verification passed
          Then send a commit message to the master
          When the number of commit messages received by the master node $\geq 2f$
            The master node sends a commit collection message to the non-master node
            When the non-master node receives the commit collection message
              Perform verification
              If verification passed
                  Then copy the new block content to the local
              End if
          End if
6. End
```

Algorithm 3: Description of the main chain consensus algorithm.

(2) Prepare stage: when the master node receives the preparation message sent by the nonmaster node, it will also verify the messages sent by each nonmaster node one by one. The verification process is similar to the verification process of the prepreparation message. When the master node receives more than $2f$ correct prepare messages from different nodes, the master node multicasts a prepare message to all active nonmaster nodes. The nonmaster node enters the submission phase after passing the verification of the prepare message

(3) Submission stage: the data interaction and verification process in the submission stage is similar to that in the preparation stage. When the nonmaster node enters the submission stage, it sends a message representing entering the submission stage to the master node. When the master node receives at least $2f$ messages from different nonmaster nodes, after the correct message is submitted by the master node, the master node sends a submission collection message to the nonmaster node, and the master node submits the block to the blockchain, and the nonmaster node verifies the message after receiving the submission collection message, after the message is authenticated successfully, copying the block data. The consensus process ends

Through the data interaction protocol in the above three stages, the final block consensus is realized to ensure the reliability and consistency of the data stored on the chain. It can be seen intuitively from Figure 3 that, compared with the traditional PBFT algorithm, the communication complexity of this protocol is reduced to a constant level. Therefore, the low complexity of the consensus protocol is more suitable for the real-time requirements of distributed energy transactions, and the transaction confirmation speed is faster. Algorithm 3 describes the main chain consensus algorithm in detail.

5. Experimental Results and Analysis

5.1. Experimental Configuration. This article uses the same test machine for experiments, the processor is AMD A4-Series A4-5000 quad-core, the operating system is 64-bit Windows 7 flagship SP1, and the memory is 4 GB. It is programmed using DVE-C++5.1 software.

5.2. Security Analysis. The traditional PBFT consensus mechanism uses the remainder of $p = v \bmod |N|$ to elect the master node, where v and N represent the current view number and the number of nodes, so the values of v and N are easy to know. The attacker can predict the next node in advance. The position of the main node of the round is attacked in advance, which destroys the consensus process of the system and easily leads to poor real-time energy transactions. The efficient and secure consensus mechanism proposed in this paper uses a verifiable random function to elect block-generating nodes and uses the node's private key as the input of the VRF function. And the result can be verified by the public key. The attacker does not know the private keys of other nodes in the network, so that the position of the next master node cannot be predicted in advance and an attack on a node cannot be launched in advance. At the same time, the consensus mechanism proposed in this paper will evaluate the health score of nodes according to the behavior of nodes in the consensus process. Nodes with low health scores will not be able to participate in the election of consensus nodes and master nodes, thereby enhancing the reliability and security of block producing nodes. Therefore, the consensus mechanism proposed in this paper can enhance the security and randomness of block node election, so as to effectively resist DDoS attacks.

5.3. Feasibility Analysis

5.3.1. Algorithms Based on Verifiable Random Functions. Most of the existing solutions use a fixed order to select the master in turn, which is the current mainstream master selection method. When the order in which the backup node becomes the master node is fixed, the master node is vulnerable to DDoS attacks by the adversary and destroys the system activity. In this paper, an algorithm is designed to improve the selection method of the consensus node set. First, a verifiable random function is used to randomly select the node set, and the health score is used as the basis for selection. For the selection of the master node, all nodes can be easily verified, and finally, the selection result is reached through consensus to ensure the randomness and reliability of node election. The main process of selecting a verifiable random function, due to its randomness and zero-knowledge proof characteristics, can effectively resist DDoS attacks and ensure the activity of the system.

5.3.2. Feasibility Analysis of View-Change. In PBFT, the view-change protocol is triggered by at least $2f+1$ consistent view-change messages from different nodes. If the current consensus fails, the system switches to a new view with the help of the view-change protocol for normal operation. However, in the PBFT optimized in this paper, the consensus mechanism follows the concept of PBFT. When the master node fails, the system will trigger the view-change protocol to conduct a new round of node election. According to the health score as the reference for selecting the master node, the selected master node is verified in combination with the verifiable random function. In this paper, a timeout mechanism is set to improve the efficiency of selecting the master node. If the master node is successfully verified within the specified time, replace the primary node. Otherwise, it fails, and the health score is used as a reference again, and the master node is verified by the verifiable random function again until the master node is successfully selected. Compared with the traditional PBFT, the selected master node has higher reliability, reduces the possibility of failure of the master node, reduces the communication overhead caused by view-change, and increases the algorithm throughput.

Table 2: Communication complexity comparison table.

Algorithm	Total number of communications	Communication complexity
PBFT	$f(n)=2n^2$	$O(n^2)$
ES-BFT [22]	$f(n)=2n^2+n+1$	$O(n^2)$
EPBFT [21]	$f(n)=n^2+3n+1$	$O(n^2)$
RBFT [23]	$f(n)=n^2+n$	$O(n^2)$
DE-BFT	$f(k)=5k+2(k\leq n)$	$O(n)$

Table 3: Transaction types table.

Transaction type	Number of test groups
Deploy smart contracts	4
Call smart contracts	4
Query the ledger status	4

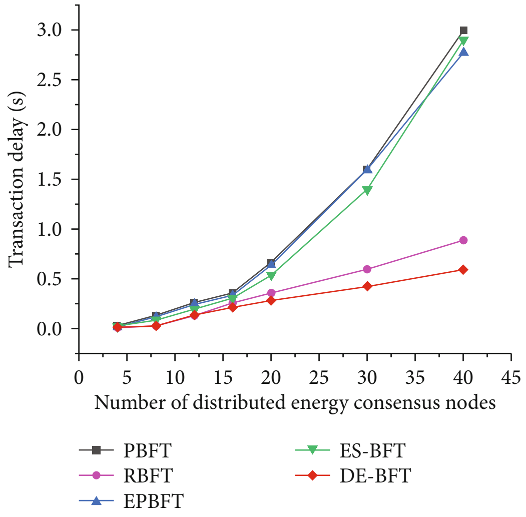

Figure 5: Comparison of transaction delays.

5.4. Stability Analysis of the System. The master node generated based on the VRF algorithm has randomness and unpredictability. The probability that the consensus node manipulated by malicious nodes is the master node is very small. When a malicious node operates on other slave nodes, it will not affect the correct consistency of the entire consensus result, because it is not the node responsible for producing blocks. Most nodes are honest. If a malicious node successfully deceives the master node, honest nodes in the consensus set can ensure that illegal blocks are not passed. At the same time, the health score set by the system can filter the behavior of malicious nodes, reduce their health score, and eliminate them from the consensus collective. Furthermore, the health score is

Table 4: Experimental parameter configuration table.

Parameter type	Parameter value
Test round	8
Number of transactions	1000
Number of concurrent processes	5

updated every n rounds, which also limits the ability of malicious nodes to manipulate security parameter choices to manipulate the identity of the master node. In contrast, the PBFT algorithm can neither identify Byzantine nodes nor guarantee the privacy of key nodes. The chosen master node is vulnerable to malicious nodes.

5.5. Communication Complexity Analysis. In this paper, the communication complexity is set as the number of times the system needs to communicate between nodes to complete a new block submission. In this section, the paper will select 4 similar algorithms for horizontal comparison and calculate the communication complexity in distributed energy trading, respectively. The statistics are shown in Table 2.

In the PBFT algorithm, the number of communications in the *request* phase is 1. The *preprepare* phase is $n-1$. The *prepare* and *commit* phases are n^2-n. The *response* phase is n. The total number of communications is $f(n)=1+n-1+2(n^2-n)+n=2n^2$. Therefore, the communication complexity of PBFT is $O(n^2)$.

The algorithm EPBFT [21] has a total of 5 stages. In *preprepare* and *prepare-1* phases, the number of communication is n. In *commit-1* phase, the primary only sends a *commit-1* message to backup 1. So the number of communication is 1. In *prepare-2* phase, the honest node will send messages to n nodes. The communication times in this stage is n^2, and the communication times in the final *commit-2* phase is n. In summary, the total communication time is $f(n)=n^2+3n+1$, and the communication complexity is $O(n^2)$, in distributed energy transaction.

The algorithm ES-BFT [22] also has 5 stages. Its total number of communications is $f(n)=2n^2+n+1$.

Compared with PBFT, the algorithm RBFT [23] lacks the *prepare* phase. So there are three stages in total, and the total communication time is $f(n)=n^2+n$. Communication complexity is both $O(n^2)$.

Due to the introduction of VRF, the algorithm DE-BFT communication times at each stage can be set as $k(k\le n)$. As can be seen from DE-BFT data consistency interaction protocol, the data interaction protocol goes through five stages. So the total communication time is $f(k)=5k+2(k\le n)$. Therefore, the communication complexity of DE-BFT is $O(n)$.

5.6. Delay. Delay indicates the time interval from the client initiating a transaction request to the request being confirmed and the chain being connected. The smaller the delay is, the faster the transaction is confirmed. In this paper, 7 groups of delay test experiments were set up according to different numbers of distributed energy consensus nodes. Each group of experiments tested the chain time required by 12 groups of the same type of transaction request under PBFT algorithm environment, EPBFT algorithm environment, ES-BFT algorithm environment, RBFT algorithm environment, and consensus protocol environment. The distribution of specific test transaction types is shown in Table 3, and its average value is taken as the final delay data.

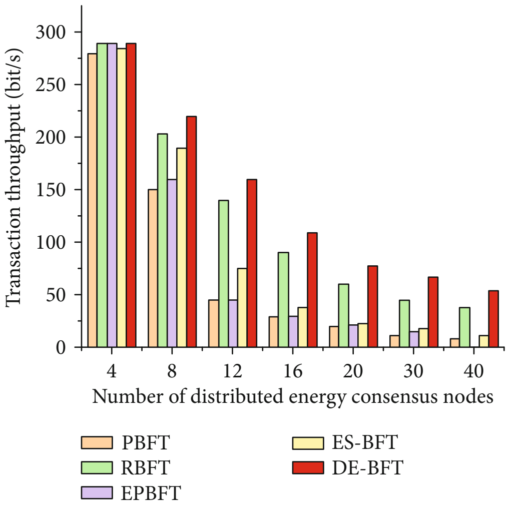

Figure 6: Comparison of transaction throughput.

Figure 5 shows the comparison of transaction delay results of various consensus algorithms in the application of distributed energy trading. With the increasing number of nodes, due to the differences in communication complexity and consensus mechanism among consensus algorithms, the delay of other consensus algorithms has an obvious linear relationship with the number of nodes. The more nodes, the greater the delay. In contrast, DE-BFT and RBFT proposed in this paper have relatively slow delay growth rate. When the number of nodes is small, RBFT and DE-BFT have similar delay, but with the increase of the number of nodes, the consensus algorithm proposed in this paper has more advantages. Distributed energy transaction will face a large number of distributed energy nodes, and the consensus algorithm proposed in this paper is more suitable for the increasing of distributed nodes.

Therefore, the consensus protocol of the energy blockchain solution proposed in this paper has a more stable performance with low latency when dealing with complex and flexible energy application scenarios.

5.7. Throughput. In this paper, 7 experiments of throughput have been set up according to different numbers of distributed energy consensus nodes. Each experiment has passed professional pressure test in PBFT algorithm environment, EPBFT algorithm environment, ES-BFT algorithm environment, RBFT algorithm environment, and consensus protocol environment in this paper with the same parameters by several professional pressure test tools. The parameter configuration is shown in Table 4, and the experimental results are shown in Figure 6.

When the number of distributed energy consensus nodes is greater than 12, the throughput of PBFT, ES-BFT, and EPBFT decreases significantly. This is because the dramatic increase in traffic during consensus puts pressure on network bandwidth, increasing the time required for consensus. Therefore, ES-BFT, PBFT, and EPBFT are not suitable for multinode blockchain environment.

RBFT and DE-BFT are in the same system environment of consensus nodes. When there are fewer consensus nodes, the throughput trends of RBFT and DE-BFT are consistent. Because the number of nodes is small, the number of communications is small. When there are more nodes, the throughput of DE-BFT is significantly higher than that of RBFT. Because the total number of communications in DE-BFT increases slowly, and the communication complexity is $O(n)$. In addition, DE-BFT uses VRF to select the master node, which can reduce the possibility of changing views and has higher reliability.

In general, the consensus protocol in this paper uses the health score reputation system to designate consensus nodes more reasonably and achieves good performance in the energy industry from the aspect of actual indicators.

6. Conclusion

Blockchain-based distributed energy transactions can effectively reduce grid dispatch costs and increase transaction transparency. This paper proposes a blockchain consensus mechanism for distributed energy transactions by analyzing the decentralization characteristics of blockchain and the noncentralized characteristics of distributed energy, aiming at the real-time and secure requirements of distributed energy trading platforms. This paper improves the traditional PBFT consensus algorithm of the consortium chain in the two stages of block node election and main chain consensus. First, in the block-generating node election stage, the health score combined with the verifiable random function is used to randomly select the master node from the candidate node set, and based on the health score, the node with the highest health score is selected as the block producer from the randomly selected candidate master nodes. In the consensus stage of the main chain, an efficient data consistency interaction protocol is adopted to reduce the communication complexity between nodes. Through experiments and analysis, it is proved that the improved consensus algorithm DE-BFT can reduce the communication complexity between nodes in the consensus stage down to $O(n)$. At the same time, compared with the traditional PBFT consensus algorithm and other improved PBFT algorithm, e.g., EPBFT, DE-BFT provides higher throughput and lower latency. The improved consensus algorithm ED-BFT not only improves the efficiency of distributed energy transactions but also ensures the security in the transaction.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this article.

Acknowledgments

This work was supported by the Research and Demonstration Project of Key Technologies of Energy Blockchain Application for Industrial Internet (037800KK52200005 and GDKJXM20201754).

References

[1] Y. He, W. Xiong, B. Y. Yang et al., "Distributed energy transaction model based on the alliance blockchain in case of China," *Journal of Web Engineering*, vol. 20, no. 2, pp. 359–386, 2021.

[2] Z. Yao, H. Pan, X. Si, and W. Zhu, "Decentralized access control encryption in public blockchain," in *International Conference on Blockchain and Trustworthy Systems*, pp. 240–257, Springer, Singapore, 2019.

[3] S. Guo, C. Huang, Y. Yan, L. Chen, and S. Shao, "Trusted digital asset copyright confirmation and transaction mechanism based on consortium blockchain," in *International Conference on Artificial Intelligence and Security*, pp. 703–714, Springer, Cham, 2021.

[4] P. B. Honnavalli, A. S. Cholin, A. Pai, A. D. Anekal, and A. D. Anekal, "A study on recent trends of consensus algorithms for private blockchain network," in *International Congress on Blockchain and Applications*, pp. 31–41, Springer, Cham, 2020.

[5] Y. F. Li, Y. L. Chen, T. Li, X. J. Ren, and C. M. Chen, "A regulatable data privacy protection scheme for energy transactions based on consortium blockchain," *Security and Communication Networks*, vol. 2021, Article ID 4840253, 11 pages, 2021.

[6] H. Zhao, M. Zhang, S. Wang, E. Li, Z. Guo, and D. Sun, "Security risk and response analysis of typical application architecture of information and communication blockchain," *Neural Computing and Applications*, vol. 33, no. 13, pp. 7661–7671, 2021.

[7] M. Saracevic and N. Wang, "New model of sustainable supply chain finance based on blockchain technology," *American Journal of Business and Operations Research*, vol. 3, no. 2, pp. 61–76, 2021.

[8] Z. Xu, "Computational intelligence based sustainable computing with classification model for big data visualization on map reduce environment," *Discover Internet of Things*, vol. 2, no. 1, p. 2, 2022.

[9] M. Castro and B. Liskov, "Practical byzantine fault tolerance," *OsDI*, vol. 1999, no. 99, pp. 173–186, 1999.

[10] K. Driscoll, B. Hall, H. Sivencrona, and P. Zumsteg, "Byzantine fault tolerance, from theory to reality," in *International Conference on Computer Safety, Reliability, and Security*, pp. 235–248, Springer, Berlin, Heidelberg, 2003.

[11] M. J. M. Chowdhury, M. Usman, M. S. Ferdous et al., "A cross-layer trust-based consensus protocol for peer-to-peer energy trading using fuzzy logic," *IEEE Internet of Things Journal*, vol. 3, no. 8, pp. 1–1, 2021.

[12] K. V. Amit, H. W. Zhong, and N. S. Yatindra, "Consensus mechanism for peer-to-peer energy trading," in *Recent Trends in Electronics and Communication*, pp. 355–364, Springer, Singapore, 2022.

[13] Y. Pan, "A novel trade transaction agreement algorithm using blockchain consensus mechanism," *Scientific Programming*, vol. 2021, Article ID 5343337, 9 pages, 2021.

[14] A. Sheikh, V. Kamuni, A. Urooj, S. Wagh, N. Singh, and D. Patel, "Secured energy trading using byzantine-based blockchain consensus," *IEEE Access*, vol. 8, pp. 8554–8571, 2020.

[15] X. Q. Xu, G. Sun, and H. F. Yu, "Au efficient blockchain PBFT consensus protocol in energy constrained IoT applications," in *2021 International Conference on UK-China Emerging Technologies (UCET)*, pp. 152–158, IEEE, Chengdu, China, 2021.

[16] J. W. Hu, Y. L. Chen, X. J. Ren, Y. Yang, X. Qian, and X. Yu, "Blockchain-enhanced fair and efficient energy trading in industrial internet of things," *Mobile Information Systems*, vol. 2021, Article ID 7397926, 13 pages, 2021.

[17] W. Hu and L. Huanhao, "A direct transaction model for energy blockchain mobile information system based on hybrid quotation strategy," in *International Conference on Human-Computer Interaction*, pp. 33–51, Springer, Cham, 2020.

[18] Y. Jiang and Z. Lian, "High performance and scalable byzantine fault tolerance," in *2019 IEEE 3rd information technology, networking, electronic and automation control conference (ITNEC)*, pp. 1195–1202, IEEE, Chengdu, China, 2019.

[19] G. L. Guo, Y. Zhu, E. Chen, G. Zhu, D. Ma, and W. C. C. Chu, "Continuous improvement of script-driven verifiable random functions for reducing computing power in blockchain consensus protocols," *Peer-to-Peer Networking and Applications*, vol. 15, no. 1, pp. 304–323, 2022.

[20] H. Wang and W. A. Tan, "Block proposer election method based on verifiable random function in consensus mechanism," in *2020 IEEE International Conference on Progress in Informatics and Computing (PIC)*, vol. 18, pp. 304–308, Shanghai, China, 2020.

[21] H. Tang, Y. Sun, and J. Ouyang, "Excellent practical byzantine fault tolerance," *Journal of Cybersecurity*, vol. 2, no. 4, p. 167, 2020.

[22] R. H. Wang, S. Y. Xing, and Q. Q. Xu, "Efficient byzantine fault-tolerant algorithm with supervision mechanism," *Computer Engineering and Applications*, vol. 57, no. 18, pp. 142–148, 2021.

[23] P. L. Aublin, S. B. Mokhtar, and V. Quéma, "Rbft: redundant byzantine fault tolerance," in *2013 IEEE 33rd International Conference on Distributed Computing Systems*, pp. 297–306, IEEE, Philadelphia, PA, USA, 2013.