Hindawi
Security and Communication Networks
Volume 2021, Article ID 6630291, 9 pages
https://doi.org/10.1155/2021/6630291

Research Article

Secure Data Collaborative Computing Scheme Based on Blockchain

Tao Feng, Xusheng Wang, Chunyan Liu, and Junli Fang

School of Computer and Communication, Lanzhou University of Technology, Lanzhou 730050, China

Correspondence should be addressed to Xusheng Wang; 1690042594@qq.com

Received 2 November 2020; Revised 11 December 2020; Accepted 22 December 2020; Published 13 January 2021

Academic Editor: Debiao He

With the rapid development of information technology, different organizations cooperate with each other to share data information and make full use of data value. Not only should the integrity and privacy of data be guaranteed but also the collaborative computing should be carried out on the basis of data sharing. In this paper, in order to achieve the fairness of data security sharing and collaborative computing, a security data collaborative computing scheme based on blockchain is proposed. A data storage query model based on Bloom filter is designed to improve the efficiency of data query sharing. The MPC contract is designed according to the specific requirements. The participants are rational, and the contract encourages the participants to implement the agreement honestly to achieve fair calculation. A secure multiparty computation based on secret sharing is introduced. The problem of identity and vote privacy in electronic voting is solved. The scheme is analyzed and discussed from storage expansion, anticollusion, verifiability, and privacy.

1. Introduction

Secure multiparty computing (SMPC) is a kind of privacy collaborative computing in which participants do not trust each other and have no trusted third party. It is applicable to solve the problem of mutual cooperation and mutual pursuit of common interests but not complete trust between organizations. Yao proposed a secure two-party computation to solve the "millionaire problem" in literature [1], which was extended by Goldreich et al. [2]. The basic model of secure multiparty computation is established theoretically.

Specifically, SMPC means that the participants have their own data by calculating the function $y_i = f(x_i)$ $(1 \le i \le n)$. The corresponding calculation results are obtained, y_i. In this calculation process, the calculation function used is f. Replace a trusted third party in an ideal situation. The basic requirement of secure multiparty computing protocol is to ensure the security of the protocol and the fairness of computation. However, some participants in the calculation conspire to disclose data information. The common collusion attacks can be divided into two types: semihonest participants (passive attack): participants perform computing tasks in accordance with the protocol and may leak their input data and calculation results to attackers, that is, attackers can obtain data. Malicious participants (active attack): the participants perform the calculation task according to the attacker's request. It not only discloses the input data and calculation results to the attacker but also tampers with the data or even terminates the protocol according to the attacker's intention.

At present, in the research of SMPC, literatures [3, 4] focus on how to prevent collusion in multiparty computing, focusing on anticollusion but not paying attention to privacy protection. At the same time, there are some problems in SMPC, such as frequent interaction between participants, which reduces the efficiency, and only part of the output of participants cannot achieve fairness. In order to solve the above problems, the works [5, 6] focus on the design of smart contracts for multiparty cooperation and propose solutions to solve specific problems such as collusion among participants and contract disputes. Based on bitcoin network [7–9], a fair SMPC protocol based on penalty mechanism is proposed. In order to solve the fairness and robustness problems in SMPC, BFR-MPC scheme is proposed [10]. A

kind of incentive mechanism is used to encourage all parties to cooperate, and those who do not cooperate will be punished economically. The fairness of the scheme is proved by game theory. In [11], the blocks are partitioned in MapReduce framework to realize data storage, and the improved homomorphic encryption is used to directly process the ciphertext and proxy reencryption is used for data sharing.

In order to solve the fairness and privacy problems of SMPC, blockchain has become an effective and feasible solution. It provides a trusted execution environment for SMPC and uses incentive mechanism to ensure computational fairness. However, there are also some problems: (1) SMPC based on bitcoin network has limitations in practical application scenarios because it cannot provide Turing complete implementation of complex functions. (2) In order to ensure the fairness of SMPC, incentive mechanism is introduced, but there is no in-depth study on the security and query efficiency of data storage. (3) In order to ensure the real-time performance of transaction calculation, it is necessary to improve the consensus algorithm to improve the consensus efficiency.

In response to the above problems, this paper combines the key technology of blockchain with SMPC. (1) According to the actual demand of computing transaction, the calculation contract is designed, which is convenient for complex practical scenarios. (2) It provides flexible data access for users by using Bloom filter and realizes efficient and feasible authorized access privacy data sharing. (3) The improved consensus algorithm is used to improve the efficiency of consensus and make the nodes consistent quickly. In the SMPC scheme based on blockchain, secret sharing is carried out to prevent the participants from conspiring to disclose the data information, thus ensuring the safety of the data. The incentive mechanism of blockchain promotes the fairness of computing.

The rest of this paper is organized as follows: in Section 2, we introduce key technologies such as secret sharing, data storage access control, and consensus algorithms. In Section 3, we introduce a system model and an SMPC protocol algorithm based on secret sharing. In Section 4, we perform security verification and performance analysis on the proposed architecture. Section 5 analyzes how our solution solves the problems in actual application scenarios. Finally, we come to the conclusions in Section 6.

2. Related Work

In this section, we introduce secret sharing, the data storage access structure in the blockchain system, and the consensus algorithm (RBFT) used in our architecture.

2.1. Password Sharing. Shamir's key sharing scheme based on gate trap is described in detail in [12]. Suppose that there are participants who do not trust each other but abide by a secret sharing protocol. The protocol is divided into two processes: secret distribution and reconstruction.

Secret distribution: choose a secret S to share among n participants. Each participant R_i has a secret S_i and a verifiable public identity $x_i\,(1 \le i \le n)$. Participant R_i randomly selects a polynomial $f_i(x) = S_i + a_{i1}x + \cdots + a_{it}x^t$ to encrypt its own secret S_i, where $n > t$, $t \in z^+$. Send the secret to other participants $R_j\,(j \ne i)\ (1 \le j \le n)$. Let $F(x) = \sum_{i-1}^{n} f_i(x)$; then, each participant R_j can calculate $F(x_j) = \sum_{i-1}^{n} f_i(x_j)$.

Secret reconstruction: when the participants negotiate to recover the secret S, a threshold t needs to be reached and only then can we reconstruct. Assuming that there are $t + 1$ participants who can calculate $F(x_1), \ldots, F(x_{t+1})$, the secret S is obtained by Lagrange interpolation formula. If the number of participants is less than the threshold t, it is impossible to get any information about S.

2.2. Blockchain

2.2.1. Data Storage Access. Manuskin et al. [13] used fragmented Ostraka nodes to solve the problem of block storage capacity limitation, which improves the query efficiency without affecting the security of the underlying consensus mechanism. Jia et al. [14] proposed an efficient query method ElasticQM for the scalable model of blockchain storage capacity. By extending the storage structure and improving the search algorithm, the query efficiency is improved. In [15], we used Bloom filter to generate keyword index and proved that it is secure against keyword search attack, which improves data query efficiency on the basis of reducing data storage space on the chain.

In the blockchain fragmentation storage model, the data is stored on the blockchain. Blockchain will cause all blocks to store data information synchronously, which increases the complexity of consensus algorithm and takes up a lot of storage space. This results in a great waste of resources on the chain and correspondingly increases the cost of data query. The scheme combines on-chain index and off-chain storage to solve the problem that blocks are difficult to store massive data and can be better compatible with traditional databases.

In the scheme, the out-of-chain database stores all the data information of the data owner. The data owner extracts the keywords and uses the Bloom filter to generate the keyword index. Then the public key of the specified data inquirer is used to encrypt the data information. The generated keyword index information is stored in the index block, which stores the key and the corresponding storage address value. If the data inquirer queries the corresponding data, he only needs to use his own private key to decrypt and obtain the data storage location according to the storage address value to get the complete original data. The specific process is shown in Figure 1.

2.2.2. Consensus Algorithm. Due to the decentralized characteristics of blockchain, it is difficult to reach consensus on the storage data information between nodes, such as easy loss and damage. Although PBFT algorithm has higher consistency than other consensus algorithms, its usability is facing challenges. If a single node fails due to failure and

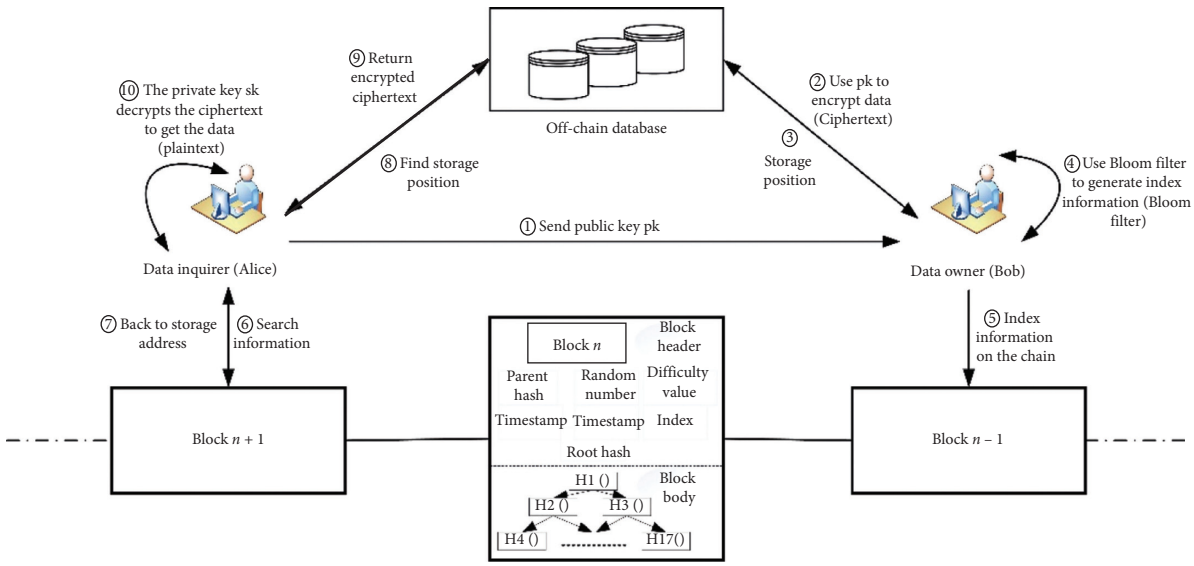

Figure 1: Data storage query model.

needs to wait for the view conversion, it is difficult to recover the failed node in time. The redundant Byzantine fault tolerance (RBFT) algorithm obtained by improving the PBFT algorithm in [16] has a complete disaster recovery mechanism, which ensures that the consensus process can be changed to dynamic nodes recovery when data inconsistency occurs. Compared with PBFT algorithm, it has higher TPS and lower delay. The scheme adopts timeout mechanism on RBFT view switching protocol, which can effectively identify and deal with the fault nodes in time.

RBFT retains the original (preprepare, prepare, and commit) processes of PBFT and has the same fault tolerance capability as PBFT. An important transaction calculation and verification link is added to ensure that a consensus is reached on the execution of transaction calculation sequence and the result of block verification. The specific process is shown in Figure 2.

RBFT consensus process steps are as follows:

(1) Transaction forwarding stage: the client sends the transaction to any node, and the node receives it and broadcasts it to other nodes.

(2) Preprepare phase: the master node packages the transaction into blocks according to the self-defined timeout mechanism and maximum block size strategy and verifies the transaction. Finally, the transaction information and verification results are written into the prepared message for broadcast.

(3) Prepare phase: after receiving the message sent by the master node, the slave node checks the current view and block number and other information and broadcasts after the check.

(4) Commit phase: the (quorum – 1) prepare messages are validated with the preprepare messages, and the results are compared with the verification results written by the master node in the preprepare messages. If it is consistent, it agrees to the verification and broadcast of the master node. Otherwise, it changes the view and declares that there is an exception in the master node.

(5) Write block: all nodes receive write commit messages to the block ledger.

The related variables of RBFT were as follows:

(1) RBFT limits the number of nodes N to at least 4 and can tolerate malicious f nodes at most $f = \lfloor (N-1)/3 \rfloor$

(2) The number of nodes needed to reach the consensus is as follows: quorum $= \lceil (N + f + 1)/2 \rceil$

3. SMPC Scheme Based on Blockchain

We introduce the system model structure, calculation contract, and the MPC protocol based on secret sharing and compare and analyze the calculation complexity of the protocol algorithm in this section.

3.1. System Model. This section introduces the blockchain network, computing network, and MPC contract in the scheme model, as shown in Figure 3. As a distributed ledger, blockchain provides a fair and reliable environment for computing. Calculation participants should register in MPC contract and pay corresponding deposit. In the process of calculation, if only some nodes get the calculation results, the deposit deduction penalty will be carried out. If each computing node complies with the protocol, it gets the corresponding output and pays a fee for it. Users are authorized to query the data stored in the chain, which realizes the data sharing and fairness of transaction calculation. Considering the actual application scenario requirements of electronic voting election, it is necessary to anonymize the identity of voters and candidates and encrypt the voting

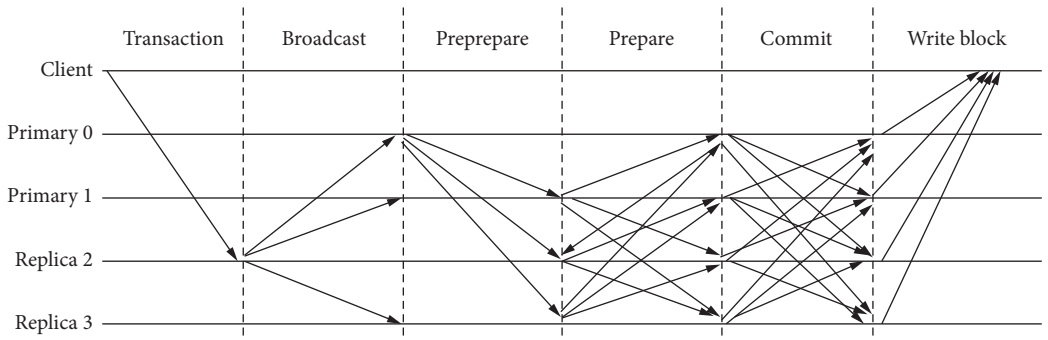

Figure 2: RBFT consensus process.

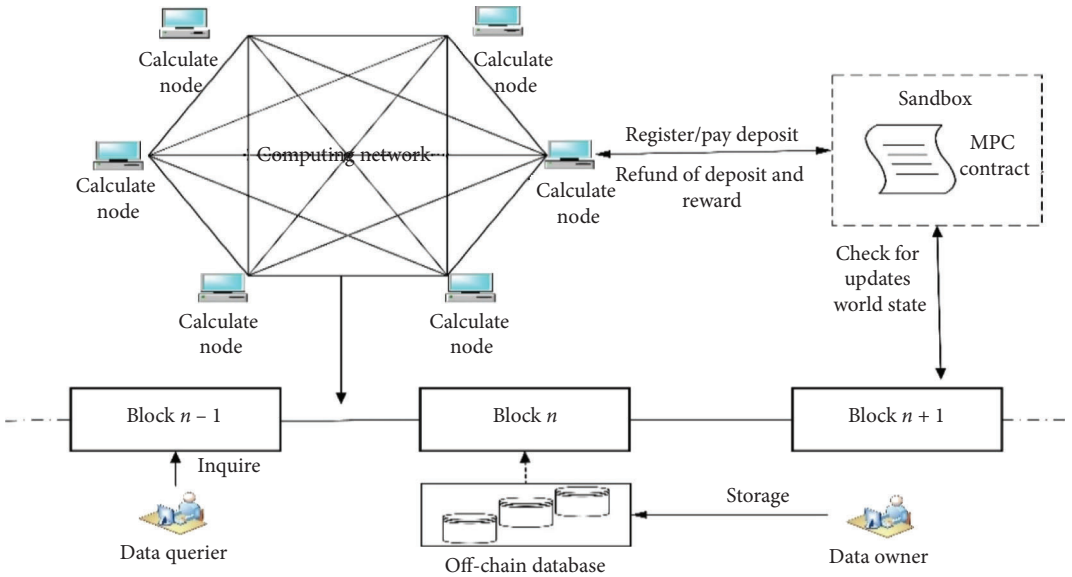

Figure 3: SMPC model based on blockchain.

results to publish on the blockchain network. The structure of the model is introduced as follows.

Firstly, the data owner extracts the keywords of the stored data, generates the key index using the Bloom filter, and encrypts the corresponding index information and the address value of the data storage with the corresponding public key of the inquirer. The index table is stored in the index block of the chain, and the query uses the private key to decrypt the data.

Secondly, when the calculation requester sends the transaction calculation request, it submits the public key to the blockchain and pays the deposit to the smart contract. The improved consensus mechanism is used to ensure the consistency of nodes. According to the request content, each node queries the original data for joint calculation. After that, the calculation result is encrypted by public key, and the requester receives the result and decrypts it with private key.

Finally, the transaction is verified by the verification node after the transaction calculation. If the transaction calculation is wrong or one of the participants does not get the calculation result, the malicious participant is traced through the time stamp and punished by the smart contract. At the same time, the honest participant is rewarded and the block status is changed accordingly. On the contrary, the deposit will be refunded.

3.2. Calculation Contract. Computing contract is a kind of smart contract in which participants negotiate and reach an agreement before participating in multiparty computing services. It is deployed on the blockchain platform in the form of code and automatically executed without human intervention trigger conditions. The contract includes the calculation node parameters, initial state, calculation process script, and execution conditions of the current calculation task.

The calculation can be divided into several rounds according to the calculation tasks and conditions. Each round of calculation in the contract has an initial judgment condition and time limit. In the initialization phase, participants (computing nodes or users) register on the MPC contract to pay the deposit. When the number of nodes involved in the calculation task and the correct input meet the initial judgment conditions, the calculation process script can be executed. If there is an error prompt in the calculation process, it may be caused by nondeterministic

factors (e.g., network congestion, code vulnerability, and other factors), so it is necessary to detect and modify and reexecute. If the calculation results are obtained, the submission must be sent correctly in this round. The verification node in the contract will verify whether the submitted information is correct. At the start of the next round of computing tasks, the contract will check the correctness of the messages submitted by all participants to determine whether the calculation continues. The execution of each round in the middle is completed in an integrated sandbox environment, which cannot be found and understood. Only in the last round, it is verified that the computing nodes publish their output results correctly and reward or deduct the deposit.

3.3. MPC Protocol. To ensure data integrity, privacy, and computing efficiency, combine homomorphic encryption and multiparty computing for transaction privacy calculations [17, 18]. For arbitrary data encryption, there is no distinction in any polynomial time algorithm, and a homomorphic encryption based on semantic security is proposed [19–21]. Threshold secret sharing solves the problem of data privacy leakage caused by the collusion of participants in SPMC [20]. Suppose that the public key cryptographic mechanism is (sk, pk, W, C, E_n, D_e); the encryption algorithm and decryption algorithm are $E_n(\,)$ and $D_e(\,)$, respectively. The encryption key and decryption key are pk and sk, respectively (where pk is public and sk is secret). Ciphertext space is C, and plaintext space is W. For any two pieces of data information $n_1, n_2 \in W$, $\forall k \in Z$ (k is a constant), if $n_1 + n_2 \in W$ and $kn_i \in W$ $(i = 1, 2)$, then

$$E_{n(pk)}(n_1) \oplus_h E_{n(pk)}(n_2) = E_n(n_1 + n_2), \tag{1}$$

$$k \otimes_h E_{n(pk)}(n_i) = E_n(kn_i), \tag{2}$$

where "$\oplus_h$" and "$\otimes_h$," respectively, represent the addition homomorphic operator and the multiplication homomorphic operator.

For the analysis of homomorphic encryption schemes [22], the homomorphic properties are applied to matrix operations in linear spaces. Assume that, in linear space $V^{n\times n}$, vector $v = (a_1, \ldots, a_n)$ is an n-dimensional row vector in linear space, and v^T is an n-dimensional column vector. Encryption algorithm E_n is used to encrypt each element in vector $v = (a_1, \ldots, a_n)$. Further extending it to matrix $B \in V^{n\times m}$, we can get

$$E_n(v) = (E_n(a_1), \ldots, E_n(a_n)) \Longrightarrow E_n(B)_{i\times j} = E_n(B_{i\times j}). \tag{3}$$

3.3.1. Protocol Design. Suppose that there is a set of computing participants $R = (R_1, \ldots, R_k)$. Each participant R_i has a matrix C_i and a corresponding vector α_i, and the corresponding relationship is R_i: $C_i \longrightarrow \alpha_i$ $(1 \le i \le k \text{ and } k \ge 4)$, where C_i is an $n \times n$-dimensional matrix and α_i is an n-dimensional vector. The value range of the number of participants k is related to the value of the number of nodes N in the above consensus mechanism. In order to make the input parameters have privacy, the invertible matrix mentioned in literature [20] is used to camouflage the cooperative calculation of linear equations as shown below:

$$\sum_{i=1}^{k} C_i x = \sum_{i=1}^{k} \alpha_i \Longrightarrow Q \sum_{i=1}^{k} C_i \hat{x} = Q \sum_{i=1}^{k} \alpha_i, \tag{4}$$

where Q is a randomly generated n-order invertible matrix and $x = \hat{x}$. The specific steps of the agreement are shown in Table 1.

3.3.2. Computational Complexity. In the above MPC protocol, it is obtained that $kn(n+1)$ encryption operations have been performed in Step 1. When calculating the encryption matrix $\sum_{i=1}^{k} E_{n(pk)}(C_i) = E_{n(pk)}(\sum_{i=1}^{k} C_i)$ and the corresponding encryption vectors $\sum_{i=1}^{k} E_{n(pk)}(\alpha_i) = E_{n(pk)}(\sum_{i=1}^{k} \alpha_i)$ in Step 2, there are a total of $(k-1)(n+1)n$ addition homomorphic operations. In Step 3, when calculating $E_{n(pk)}(Q\sum_{i=1}^{k} C_i)$, there are $kn^2(n-1)$ times of additive homomorphism and kn^3 times of multiplication homomorphism. A total of $kn(n-1)$ times of additive homomorphism and kn^2 times of multiplication homomorphism were performed when calculating $E_{n(pk)}(Q\sum_{i=1}^{k} \alpha_i)$. In Step 4, a total of $kn(n+1)$ times decryption operations were performed.

Compared with the calculation time $y1$ of the multiplication homomorphism and the addition homomorphism of the SMPC protocol in [23], the calculation time $y2$ of this protocol is reduced. Since the improved consensus mechanism requires at least 4 participants, the lower limit of the matrix dimensions is set to 4 dimensions, as shown in Figures 4 and 5.

4. Security Proof and Performance Analysis

In this section, we have played the adversary (attacker) and challenger (computing participant) game under the DBDH assumptions. It proves that the scheme is safe under specific ciphertext attacks, and we analyze the performance of the scheme in anticollusion, verifiability, scalability, and privacy.

4.1. Security Proof

Lemma 1. *Based on the DBDH assumption, our scheme can resist selected ciphertext attacks in the random oracle model [24], so our scheme is IND-CCA safe.*

Proof. Suppose that there is a probabilistic polynomial time PPT; given a public key encryption scheme $I = (\text{Gen}, \text{En}, \text{De})$, the adversary A uses the auxiliary input function ϕ in the polynomial time to play game with the challenger B as follows:

(1) Key generation: challenger B runs the key generation algorithm Gen( ) to obtain the public and private key pair (pk, sk) and send the public key pk to A.

Table 1: Specific steps of the calculation protocol.

MPC protocol
Input: for R_i , there is an $n \times n$-dimensional matrix C_i and a corresponding n-dimensional vector α_i, where $1 \le i \le k$ and $k \ge 4$
Initialization: randomly select a pair of public and private keys (pk, sk), decompose the private key to obtain subkey sk_l, $(1 \le l \le k)$, distribute the subkey to each participant $R = (R_1, \ldots, R_k)$, and randomly select a reversible matrix Q with the same dimension for participant R_1
Step:
1 Use public key pk to encrypt the matrix and corresponding vector owned by participant R_j $(2 \le j \le k)$ to obtain $E_{n(pk)}(C_j)$ and $E_{n(pk)}(\alpha_j)$; then send $E_{n(pk)}(C_j)$ and $E_{n(pk)}(\alpha_j)$ to R_1
2 R_1 uses the public key pk to encrypt the matrix C_1 and the vector α_1 to obtain $E_{n(pk)}(C_1)$ and $E_{n(pk)}(\alpha_1)$; calculate the encryption matrix $\sum_{i=1}^{k} E_{n(pk)}(C_i) = E_{n(pk)}(\sum_{i=1}^{k} C_i)$ and the phase encryption vectors $\sum_{i=1}^{k} E_{n(pk)}(\alpha_i) = E_{n(pk)}(\sum_{i=1}^{k} \alpha_i)$ at the same time
3 Use the invertible matrix Q to calculate $E_{n(pk)}(Q\sum_{i=1}^{k} C_i)$ and $E_{n(pk)}(Q\sum_{i=1}^{k} \alpha_i)$; then send them to the corresponding participant R
4 When the number of participants possessing subkey sk_i reaches the threshold, the master key sk is reconstructed by Lagrange interpolation formula, and the encryption matrix $D_{e(sk)}(E_{n(pk)}(Q\sum_{i=1}^{k} C_i)) = Q\sum_{i=1}^{k} C_i$ and vectors $D_{e(sk)}(E_{n(pk)}(Q\sum_{i=1}^{k} \alpha_i)) = Q\sum_{i=1}^{k} \alpha_i$ are computed by sk
5 Participant $R = (R_1, \ldots, R_k)$ collaboratively calculates the linear equation $Q\sum_{i=1}^{k} C_i\hat{x} = Q\sum_{i=1}^{k} \alpha_i$ to obtain $\hat{x}$
Output: since Q is an invertible matrix, $\hat{x} = x$

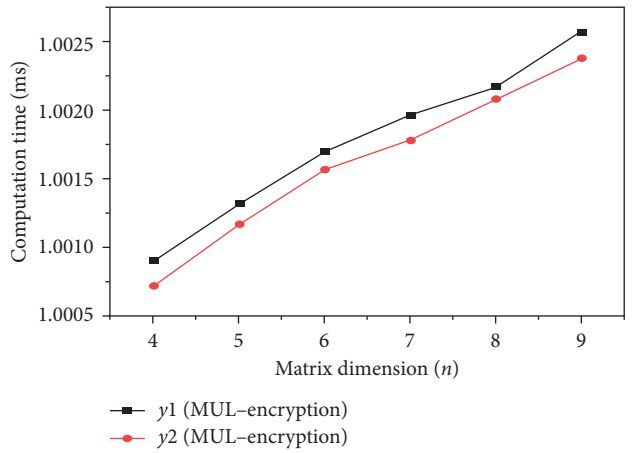

Figure 4: Comparison of homomorphic operation.

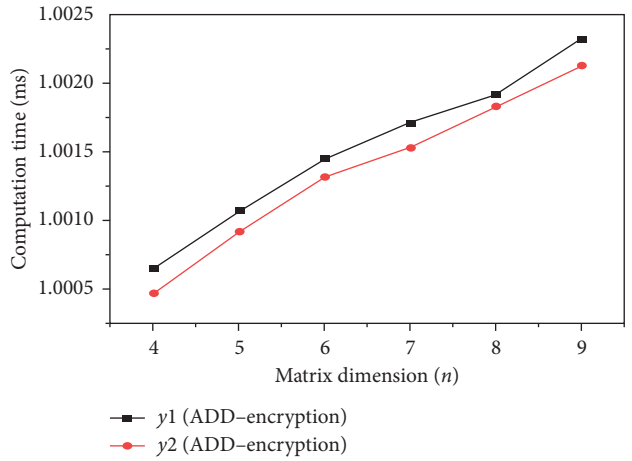

Figure 5: Comparison of matrix addition homomorphic time of matrix multiplication.

(2) Inquiry 1: ciphertext decryption oracle OracleDe( ): A submits ciphertext C to OracleDe( ), and B runs OracleDe( ) to decrypt C to obtain the corresponding information. Key leakage oracle OracleLe( ): for any auxiliary input function $\phi \in I$, the adversary A sends ϕ to B, and B sends $\phi(pk, sk)$ to A.

(3) Challenge phase: A selects two messages M_0, M_1 and sends them to B. Challenger B randomly selects a byte $\sigma \in \{0, 1\}$, tries to calculate ciphertext $C^* = E_n(pk, M_\sigma)$, and sends it to adversary A.

(4) Inquiry 2: repeat inquiry 1.

(5) Guess: adversary A submits σ's guess σ^*; when $\sigma = \sigma^*$, A wins the game; otherwise, A loses the game.

In the scheme, the advantage of opponent A to win the game is defined as $\mathrm{Adv}_A = |pr(\sigma = \sigma^*) - (1/2)|$, and the advantage of opponent A to break the encryption scheme in probabilistic polynomial time PPT is $\varepsilon = \varepsilon(n) = \mathrm{Adv}_A^{(0)}, f(n)$ (ε is a negligible small amount). The challenge ciphertext generated by $\mathrm{En}(pk, M_\sigma)$ and $\mathrm{En}^*(\phi(g, sk), M_\sigma)$ (g is a common parameter) is indistinguishable; that is, $\mathrm{Adv}_A^{(0)}, f(n) = \mathrm{Adv}_A^{(1)}, f(n) = \varepsilon$. Therefore, the advantage of adversary A is negligible in the game. No adversary can break our algorithm, so our solution is safe.

4.2. Performance Analysis

4.2.1. Prevention of Collusion. Collusion attack is a key problem in the real world, especially when the malicious people collude to obtain the data and information of other participants to seek benefits because of some intention (e.g., economic factors). Under the condition of semihonest model, our proposed SMPC scheme based on blockchain can avoid collusion and enable participants to execute the protocol honestly. Blockchain provides a trusted environment for collaborative computing through cryptography technology. Participants' data are encrypted and stored on the chain, and the master key needs to be reconstructed when executing the computing protocol, effectively preventing collusion among participants. At the same time, because the data is encrypted and stored on the chain, the

MPC protocol algorithm is encrypted by a random reversible matrix. After analyzing the game process of the adversary and the challenger, the generated challenge ciphertext is indistinguishable, and the advantage of the adversary to break the encryption can be ignored, which proves that our solution can resist the selective ciphertext attack. Secondly, the participants register and confirm their identity information, and the blocks of data storage have unique time stamp identification. At the same time, they pay a deposit as a guarantee to abide by the contract agreement, which makes noncollusion obtain higher benefits than collusion and eliminates the motives of participants' collusion.

4.2.2. Verifiable. Due to the lack of trusted third party in existing secure multiparty computing, there may be a potential security risk that participants conspire to tamper with data. The decentralization of blockchain provides a credible environment for multiparty computing, and the core of decentralization is consensus mechanism. For secure multiparty computing based on blockchain, before calculation, nodes and other nodes need to use consensus algorithm to generate blocks and verify the synchronization, so as to ensure the efficient consistency between nodes, and then homomorphic encryption calculation can be carried out to ensure the verifiability of transaction calculation.

4.2.3. Scalable. Blockchain is a decentralized ledger. Information stored on the chain through encryption cannot be tampered with. The consensus generation blocks on the chain all have time stamp traceability. However, due to the limited block storage, it is not possible to store all the information completely in the block. In order to ensure the security and fast query of the storage information, it is not universal to adopt the block partition method. In the scheme, we combine the blockchain storage and the on-chain index to store the original information file in the out-of-chain database, and the on-chain block storage address and keyword index are easy to find.

4.2.4. Privacy. For this scheme, there are three privacy protection measures. First, when data information is stored, the data information is encrypted by using password technology, the data file is encrypted and stored in the database outside the chain, and the storage address and index information are stored in the block. Only users with decryption key or authorized users can get the original data information. Second, before calculation, the data information stored in the block needs consensus mechanism to reach consensus chain. In the improved consensus algorithm, the information received from the node is verified with the information written by the master node. If the information is consistent, the stored data is correct. Third, threshold secret sharing is used to decompose the key and distribute it to each participant. In decryption, the secret reconstruction is needed to recover the master key. When the subkey share of the participant reaches the threshold, the master key can be recovered. Otherwise, the participant cannot reconstruct the key for decryption. Because the scheme is encrypted during the data storage and calculation phases, we performed a formal proof of the optional ciphertext attack on the entire scheme, which proved that our scheme is safe and can protect data privacy.

5. Application Scenarios

The scheme is suitable for applications with high privacy requirements and easy information sharing, for example, the typical scene of electronic election. In the traditional voting scheme, the privacy and correctness of voting depend on the credibility of the voter. Due to the great rights of the vote counter in the voting scheme, the ballot contains sensitive identity information and vote data information of the voters and voters, which may directly lead to the voters being unable to make a real choice, thus affecting the fairness and effectiveness of the voting results. The traditional voting scheme has the following limitations: (1) the voter knows the information of the vote; (2) the teller may know the identity of the voter; (3) the result of the ballot depends on the correct calculation of the voter; (4) the voter trusts the teller not to tamper with the ballot.

To solve the above problems, our proposed multiparty computing electronic voting election scheme based on blockchain ensures the anonymity of identity and the security of ballot data. The whole voting process is open and verifiable, and the security and privacy issues in the voting process are protected. If the voter is qualified to vote only after passing the registered identity authentication, once the voter's vote is submitted, it cannot be modified or deleted; except for the voting participants (i.e., voters and candidates), the third party cannot obtain the specific content of the vote; the voter can verify his or her own vote, including whether it is tampered with or not and whether it is included in the final result; all the people except the voter can check whether the vote is tampered with or not and whether it is included in the final result. Authorization can arbitrarily verify and supervise the voting process.

6. Scheme Comparison

Through the performance analysis, we compare and analyze the existing SMPC based on blockchain. In order to prevent collusion, Wang and Sen-ching and Luo and Li [4, 23] improved the use mechanism of previous work. The influence of privacy preference on preventing collusion attack is analyzed. We also find the limitation of the same preference in symmetric game. The research scheme of SMPC based on bitcoin network [9] can prevent collusion and improve fairness through punishment mechanism. Due to its limitations, it cannot be applied to complex functional scenarios. References [10, 11] are all based on blockchain for secure multiparty computing, without relying on a trusted third party. Reference [10] obtains the correct output under the premise of ensuring the privacy of the input. In order to solve the problem of fairness and robustness, a BFR-MPC scheme is proposed. The fairness is improved by encouraging the cooperation of all parties through incentive mechanism. The security data information sharing

Table 2: Performance comparison between this article and other SPMC solutions.

Program	Rely on trusted third parties	Verifiable	Privacy	Controllability	Scalability	Prevent collusion
[4]	✓	×	✓	−	−	✓
[9]	×	×	✓	−	−	✓
[10]	×	✓	✓	×	×	✓
[11]	×	✓	✓	✓	✓	−
[23]	✓	×	✓	−	−	✓
This paper	×	✓	✓	✓	✓	✓

and multiparty computing model are proposed in [11]. The data is partitioned to expand the storage capacity, and the consensus algorithm is improved to ensure the consistency between nodes. The homomorphic encryption algorithm directly uses ciphertext for data security calculation to ensure privacy. Finally, the performance analysis and simulation results show that the efficiency is improved significantly.

The scheme carries out multiparty security calculation on the blockchain and adopts the combination of on-chain index and off-chain index to expand the block storage, and the use of Bloom filter improves the efficiency of data query and verification. A calculation contract is designed by smart contract to pay deposit during transaction calculation to prevent collusion among participants. The scheme has high fairness and security. The specific performance comparison is shown in Table 2.

7. Concluding Remarks

Blockchain is a decentralized distributed storage structure, which will not cause data loss or damage due to single node failure. The scheme carries out multiparty collaborative computing on the blockchain. The data is encrypted to ensure the security. Secure multiparty computing can ensure that multiple participants who do not trust each other can perform the given computing tasks while protecting data privacy. In addition, in order to prevent the calculation participants from colluding to destroy the correctness of the calculation results, combined with the smart contract in the blockchain, the specific MPC contract is designed to encourage the participants to abide by the protocol to participate in the calculation honestly. The characteristics of blockchain, such as unforgeability, decentralization, and anonymity, combined with secure multiparty computing, can be applied in the fields of electronic auction, bidding, and medical sensitive information sharing. The next work is to apply the combination of blockchain and secure multiparty computing in specific actual scenarios to solve the existing key problems. At the same time, regulatory authorities need to formulate standards and improve regulatory policies according to the actual business needs.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant no. 61762060), the Educational Commission of Gansu Province, China (Grant no. 2017C-05), and the Foundation for the Key Research and Development Program of Gansu Province, China (Grant no. 20YF3GA016).

References

[1] A. C. C. Yao, "How to generate and exchange secrets," in *Proceedings of the 27th Annual Symposium on Foundations of Computer Science (SFCS 1986)*, pp. 162–167, IEEE, Toronto, Canada, October 1986.

[2] O. Goldreich, S. Micali, and A. Wigderson, "How to play any mental game," in *Proceedings of the 19th Symposium on Theory of Computing (STOC)*, pp. 218–229, ACM, New York, NY, USA, May 1987.

[3] O. Goldreich, *Foundations of Cryptography*, Vol. 2, Cambridge University Press, Cambridge, UK, 2004.

[4] Z. Wang and S. C. Sen-ching, "On privacy preference in collusion-deterrence games for secure multi-party computation," in *Proceedings of the 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2044–2048, IEEE, Shanghai, China, March 2016.

[5] C. D. Clack, V. A. Bakshi, and L. Braine, "Smart contract templates: foundations, design landscape and research directions," 2016, https://arxiv.org/abs/1608.00771.

[6] C. Dong, Y. Wang, and A. Aldweesh, "Betrayal, distrust, and rationality: smart counter-collusion contracts for verifiable cloud computing," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 211–227, ACM, Scottsdale, AZ, USA, November 2017.

[7] R. Kumaresan, V. Vaikuntanathan, and P. N. Vasudevan, "Improvements to secure computation with penalties," in *Proceedings of the 23rd ACM Conference on Computer and Communications Security*, pp. 406–417, ACM Press, Vienna, Austria, October 2016.

[8] A. Kiayias, H. S. Zhou, and V. Zikas, "Fair and robust multiparty computation using a global transaction ledger," in *Proceedings of the 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 705–734, Springer, Berlin, Germany, 2016.

[9] L. H. Song, T. Li, and Y. L. Wang, "A survey on applications of game theory in blockchain," *Journal of Cryptologic Research*, vol. 6, no. 1, pp. 100–111, 2019.

[10] H. Gao, Z. Ma, S. Luo, and Z. Wang, "BFR-MPC: a blockchain-based fair and robust multi-party computation scheme," *IEEE Access*, vol. 7, pp. 110439–110450, 2019.

[11] T. Wang, W. P. Ma, and W. Luo, "Information sharing and secure multi party computing model based on blockchain," *Journal of Computer Science*, vol. 46, no. 9, pp. 162–168, 2019.

[12] K. K. Phiri and H. Kim, "Linear secret sharing scheme with reduced number of polynomials," *Security and Communication Networks*, vol. 2019, 2019.

[13] A. Manuskin, M. Mirkin, and I. Eyal, "Ostraka: secure blockchain scaling by node sharding," 2019, https://arxiv.org/abs/1907.03331.

[14] D. Y. Jia, J. C. Xin, Z. Q. Wang, W. Guo, and G. R. Wang, "ElasticQM: a query model for storage capacity scalable blockchain system," *Ruan Jian Xue Bao/Journal of Software*, vol. 30, no. 9, pp. 2655–2670, 2019, in Chinese.

[15] F. Tao, X. Chen, and C. Liu, "Research on data encryption and retrieval privacy protection based on blockchain," in *Proceedings of the 5th International Symposium on Privacy Computing*, New York; NY, USA, February 2019.

[16] K. Lei, Q. Zhang, and L. Xu, "Reputation-based byzantine fault-tolerance for consortium blockchain," in *Proceedings of the 2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS)*, pp. 604–611, IEEE, Singapore, December 2018.

[17] L. M. Gong, S. D. Li, J. W. Dou, Y. M. Guo, and D. S. Wang, "Homomorphic encryption scheme and a protocol on secure computing a line by two private points," *Ruan Jian Xue Bao/Journal of Software*, vol. 28, no. 12, pp. 3274–3292, 2017.

[18] D. Das, "Secure cloud computing algorithm using homomorphic encryption and multi-party computation," in *Proceedings of the 2018 International Conference on Information Networking (ICOIN)*, pp. 391–396, IEEE, Chiang Mai, Thailand, January 2018.

[19] S. F. Zhou, J. W. Dou, Y. M. Guo, Q. Mao, and S. D. Li, "Secure multiparty vector computation," *Chinese Journal of Computers*, vol. 40, no. 5, pp. 1134–1150, 2017.

[20] L. Chen and B. G. Lin, "Secure protocols for resolving distributed system of linear equations," *Journal of Information Network Security*, vol. 9, pp. 2–5, 2013.

[21] J. W. Dou, X. H. Liu, S. F. Zhou, and S. D. Li, "Efficient secure multiparty set operations protocols and their application," *Chinese Journal of Computers*, vol. 41, no. 8, pp. 1844–1860, 2018.

[22] W. Xu, S. Xiang, and V. Sachnev, "A cryptograph domain image retrieval method based on paillier homomorphic block encryption," *Tech Science Press*, vol. 55, no. 2, pp. 285–295, 2018.

[23] W. J. Luo and X. Li, "The secure multi-party protocol of matrix product and its application," *Chinese Journal of Computers*, vol. 7, pp. 1230–1235, 2005.

[24] M. W. Zhang, M. W. Chen, D. B. He, and B. Yang, "An efficient leakage-resilient and CCA2-secure PKE system," *Chinese Journal of Computers*, vol. 39, no. 3, pp. 492–502, 2016.