Wireless Communications and Mobile Computing

Emerging Technologies towards Improving Confidentiality and Privacy in Blockchain

Lead Guest Editor: Jinguang Han
Guest Editors: Liqun Chen, Li-Quan Chen, and Willy Susilo

Emerging Technologies towards Improving Confidentiality and Privacy in Blockchain

Wireless Communications and Mobile Computing

Emerging Technologies towards Improving Confidentiality and Privacy in Blockchain

Lead Guest Editor: Jinguang Han
Guest Editors: Liqun Chen, Li-Quan Chen, and Willy Susilo

Contents

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 4467917, 17 pages
https://doi.org/10.1155/2022/4467917

Research Article

Formal Modelling of PBFT Consensus Algorithm in Event-B

Jie Li,[1,2] Kai Hu,[1,2] Jian Zhu,[3] Jean-Paul Bodeveix,[4] and Yafei Ye[5]

[1]*State Key Laboratory of Software Development Environment, Beihang University, Beijing, China*
[2]*Key Laboratory of Blockchain Application Technology of Yunnan Province, Yunnan, China*
[3]*The 14th Research Institute of CETC, Nanjing, Jiangsu, China*
[4]*Institut de Recherche en Informatique de Toulouse, Toulouse, France*
[5]*Yunnan Innovation Institute of Beihang University, Yunnan, China*

Correspondence should be addressed to Yafei Ye; yeyf@bhynii.com

Received 7 January 2022; Revised 10 July 2022; Accepted 29 August 2022; Published 14 October 2022

Academic Editor: Jinguang Han

The practical Byzantine Fault Tolerance (PBFT) is a classical consensus algorithm that has been widely applied in an alliance blockchain system to make all nodes agree to certain transactions under the assumption that the proportion of Byzantine nodes is no more than 1/3. It is prevalent due to its performance, simplicity, and claimed correctness. However, any vulnerability of the consensus algorithm can lead to a significant loss in finance because no one can change the transaction results after execution. This paper proposes a formal development method of the PBFT algorithm by horizontal refinement in Event-B, which allows us to manage the complexity of the proof process by factoring the proof of correctness into several refinement steps. During the development of PBFT, we have specified the core mechanism like parameterized message types, primary node change, and water-mark interval. Furthermore, we present a mechanical verification of the safety and liveness properties of the model in Rodin, which can be partially and widely used to check the blockchain consensus algorithm vulnerability using a refinement tree of algorithms.

1. Introduction

Blockchain technology is an emerging decentralized distributed system formed by using encryption algorithms, P2P technology, tree structure, consensus algorithm, reward mechanism, etc. It can be argued that blockchain is a distributed ledger with the characteristics of decentralization, immutability, traceability, trust, openness, and transparency. Due to these characteristics, blockchain technology has been widely applied in many industrial fields such as finance, cloud computing, and the Internet of Things. Bitcoin [1], as a representative of the first generation of blockchain technology, provides people with popular virtual currencies. It has successfully solved the double-spending problem and the Byzantine faults. Ethereum serves as the symbol of the second generation of blockchain while providing smart contract functionality. With the success of Bitcoin and Ethereum, there is more and more research on blockchain technology. As the core of blockchain technology, the consensus mechanism fundamentally determines the security, availability, and system performance of the entire blockchain system. Research on the consensus mechanism of the blockchain is of great significance to the expansion of the blockchain, the increase of transaction processing speed, and the improvement of security.

Currently, the consensus mechanism is mainly evaluated from four dimensions: security, scalability, efficiency, and resource consumption. Security is an essential point. A consensus algorithm ensures that all node operations are consistent with the service specifications and that no undesirable results occur during the process. Furthermore, we consider that it has good fault tolerance and can detect Byzantine errors, etc.

According to research [2], almost all the distributed systems which support strong consensus use Byzantine fault-tolerant protocols as their core algorithm. PBFT (Practical Byzantine fault tolerance) serves as one of their main building blocks or inspires them, such as [3–5]. Besides, compared to the proof of work (POW) [6] algorithm and the proof of stake (POS) [7] algorithm, the PBFT [8] is more prevalent in

Consortium Blockchain because it consumes less computation power. Even though the safety of the PBFT has been mathematically and manually proved, formal verification of safety and liveness is still concerned with confirming the distributed system consistently runs. Furthermore, if a typical alliance chain uses PBFT as the consensus protocol, some details may also be modified and need to be verified again. In this case, the model will still be instructive, and sometimes only a few configurations have to be redefined.

There are a lot of formal verification tools and research about verifying protocols, as shown in Section 2. Event-B [9] is a formal modeling language that supports specifying and implementing algorithms and systems as discrete transition systems based on a typed set theory. Since it integrates some tools which can verify the smart contracts automatically, and the syntax of Event-B. Event-B is one of the most popular theorem proving tools, and ProB is a model checker for the B-Method, which can be called to check specific errors.

We use horizontal refinement and ProB plug-in to build the protocol incrementally with the Event-B language and the tool Rodin platform [10]. Then these refinements are introduced to make Rodin produce the wanted proof obligations for stabilization. The stabilization property is a particular kind of liveness property. For any specific instance, all the messages sent from clients for execution will be confirmed by nodes ultimately, which means all nodes will reach a consensus. We rely on Event-B and Rodin to take advantage of the ability to write specifications in an expressive language that the built-in generator can translate and forward to SMT (Satisfiable Modulo Theories) solvers, which is logical first-order formulas of a decision problem. Our model illustrates the core mechanism of the PBFT, while different application scenarios may lead to tiny differences in details.

The main contributions of this paper are as follows:

(1) Detailed specification and formal modeling of PBFT by horizontal refinement implement critical features like weak synchrony, view change functions, and watermark interval. In addition, we provide a generic model, which can extend to implement various specific protocols

(2) Formal verification of the agreement property by proving the property that any message being effectively agreed on should have at least f + 1 identical execution result from different nodes

(3) We apply the weak fairness theory to verify the liveness property. A suitable variant is proposed to prove that all convergent events should remain globally enabled and decrease the variant. In contrast, the others should not increase it

In Section 2, we present the description of Event-B and the PBFT as a preliminary. In Section 3, some related work about formal verification of protocol algorithm is given. In Section 4 and 5, we address the development of the PBFT by stepwise refinement and verify the agreement property. In Section 6, we prove the expected liveness property under a weak fairness assumption. Finally, we give a short analysis and summary of the development as the conclusion and discuss future work in Section 6 and discuss future work in Section 7.

2. Preliminary

2.1. The Introduction of Event-B. An Event-B model contains two components: contexts and machines.

(1) A context comprises abstract sets that define data types and constants linked to some properties defined as axioms

(2) A machine contains variables, invariants, theorems, variants, and events. A machine has variables associated with invariants and events. An event consists of a guard and an action. The guard denotes the enabling condition of the event, and the action represents how the event modifies the state of the machine. Values of machine variables represent different machine states

As shown in Figure 1, a machine can refine another machine and inspect one or more contexts. Contexts can extend to other ones as well.

2.2. The Process of PBFT. As mentioned above, PBFT was proposed by Miguel Castro and Barbara Liskov in 1999 [8] to solve the Byzantine problem in asynchronous distributed systems. Their model adds the design of timeout for bypassing the impossibility of FLP [11], which proved no algorithm could guarantee the consensus in asynchronous and distributed systems, and hence it is called weak synchronization. For the default transmission of information, nodes use cryptographic technology to sign the information to prevent malicious behavior from tampering with the message. There are some restrictions on the number of nodes. We assume that most f nodes may exhibit Byzantine failure, and most other f nodes may exhibit crash failures. To ensure safety and liveness, responses from nonfaulty nodes must outnumber faulty nodes, which means $n - 2f > f$. Therefore $n > 3f$ (n represents the number of total nodes).

The standard process of the algorithm is mainly divided into five steps: Request, Pre-Prepare, Prepare, Commit and Reply stages. According to these steps, the messages have five categories: REQUEST, PRE-PREPARE, PREPARE, COMMIT, and REPLY. In addition, nodes in the system contain primary nodes and backup nodes (except the primary node). The five processes are as follows:

(1) Request stage

The client c sends a request message Request (o,t,c) to the primary node, where o is the request by client c and t is a timestamp marking the execution time of the client. If the client sends the message to a backup node, the node will forward it to the primary node.

(2) Pre-prepare stage

The primary node will check and broadcast the request message to all the other nodes. The nodes receive and verify

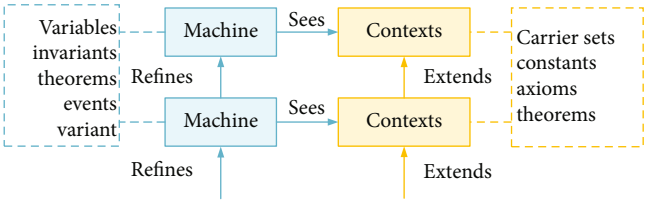


Figure 1: The relationship between machine and contexts.

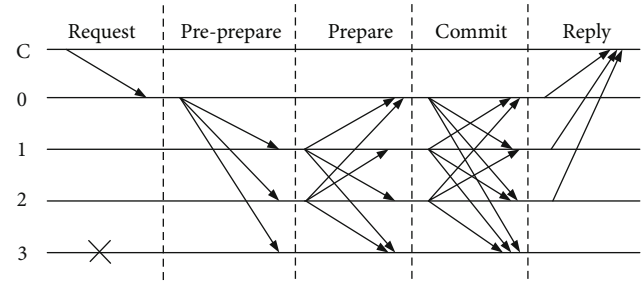


Figure 2: The normal process of PBFT (The figures and detailed description are in Practical Byzantine Fault Tolerance (OSDI, 1999).)

the message from the primary node. After the verification, they will broadcast the preprepare message (v,n,d,m). Here, v represents the view in which the message is being sent, n indicates the sequence number assigned by the primary node for ensuring the consistency of non-Byzantine nodes in the system, m is the client's request message, and d is m's digest;

(3) Prepare stage

The backup nodes check the soundness of the received Pre-Prepare message. After the verification, the message Prepare (v,n,d,i) is sent to all other nodes (including the primary node). Here, i is the identity of the node;

(4) Commit stage

The primary node and the backup node verify the soundness of the received Prepare message. If the node receives $2f + 1$ valid messages, it will send the Commit (v,n,d,i) message to the other nodes (including the primary node);

(5) Reply

The node verifies the soundness of the received COMMIT message. If the node receives $2f + 1$ valid messages, it allows the client's request to operate 'o', and returns Reply (v,t,c,i,r) to the client. In this format, r represents the result of the requested operation. If the client receives $f + 1$ identical Reply message, it means that the request initiated by the client has reached the consensus of the entire network.

Figure 2 shows the normal process where the primary node is nonfaulty [8]. In detail, C is the client, node 0 is the primary node (not faulty), and node 3 crashes and does not respond to the received messages.

When the primary node fails, a view-change is needed to ensure the liveness of the distributed system, as shown in Figure 3. The timer of the backup node ends view v due to a long wait. The view-change mechanism is triggered to change the system to view $v + 1$. The node will broadcast the message ViewChange ($v + 1$,n,C,P,i) to the entire system, and the node will only accept checkpoint, view-change, and new-view message types. Here, i is the node's identity, n equals the latest stable checkpoint s known to i. C is a set containing $2f + 1$ valid checkpoint messages for proving the correctness of checkpoint s. P is a set that includes the information of the Pre-prepare and Prepare messages with a sequence number higher than n.

When the primary node P of view $v + 1$ receives 2f valid view-change messages, it will broadcast the message New-View($v + 1$,V,O) to other nodes, where V is the set of received view-change messages (including the message sent by itself), and O contains Pre-prepare messages to be confirmed. After receiving the new-view type message, the backup node verifies the content of the view $v + 1$ and the correctness of the set O. If the message is valid, the node will enter view $v + 1$.

3. Related Work

Consensus mechanism serves as the fundamental and core technology of blockchain. It determines how the participating nodes agree on certain specific data. Consensus agreements have been studied for a long time, and the consensus mechanisms contain classic distributed and blockchain ones.

As early as 1975, Akkoyunlu et al. [12] put forward the "two armies' problem" in the computer science field, which opened the research on consensus mechanism. In 1978, Lamport defined a logical clock system to sequence events and introduced a state machine replication method in a distributed system to solve the mutual exclusion problem of resources in [13]. It is the first work involving consensus in distributed systems. In 1982, Leslie Lamport et al. [14] proposed the original presentation with the basic protocols "Oral Messages" (OM) and "Signed Messages" (SM) to solve the Byzantine General Problem. However, the cost of these solutions is very high and requires O(N^(f+1)) (f is the number of Byzantine nodes) in information exchange. In 1989, Lesile Lamport proposed the Paxos [15] algorithm based on the process of passing legislation in Congress without considering the Byzantine failure. This algorithm contains three roles: proposer, reviewer, and learner; it is divided into two stages: proposal preparation (consensus discussion) and proposal release (consensus release). But it does not support Byzantine Fault Tolerance. In 1999, Miguel Castro et al. proposed PBFT to the Byzantine Generals question [8]. In [11], Fischer, Lynch, and Paterson proposed the impossibility of FLP which proved that no algorithm could guarantee the consensus in an asynchronous system, even if only one process failed. For bypassing the impossibility of FLP, this algorithm used synchronization assumptions. And it reduced the cost of information exchange to O(n^2) (n is the number of nodes) as well.

In 2008, Satoshi Nakamoto used POW in Bitcoin and the consensus mechanism enters the era of blockchain consensus. POW indicated each node in the system conducts a round of computing power every ten minutes and the winner node

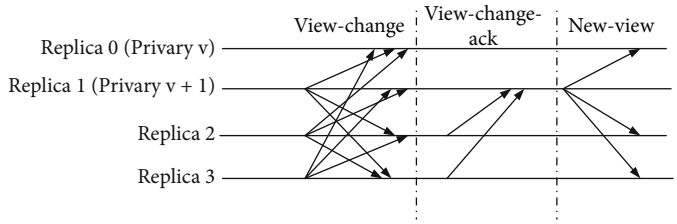


Figure 3: The process of view-change (the figures and detailed description are in Practical Byzantine Fault Tolerance (OSDI, 1999).)

obtains the right to keep accounts and synchronizes the new additions to other nodes. Due to the POW algorithm, the whole bitcoin system consumes excessive resources every year. In 2012, the Quantum Mechanic POS [7] consensus algorithm was proposed and used in the Peercoin system. POS indicated the node with the highest equity obtains the right to keep accounts. Compared with POW, POS has the advantage of less resource consumption. But it was easy to bring the centralization problem. In 2013, the Bitshares project proposed DPOS (Delegated proof-of-stake) [16] based on POW and POS, in which each node voted to form a board of directors for consensus. In 2014, Diego Ongaro [17] proposed the Raft algorithm. It inherited the advantages of POS and weakened the centralization problem. The algorithm contains three roles: leader, candidate, and follower. To achieve easy-to-understand purposes, the entire consensus process is simplified into four independent subproblems: leader election, log replication, safety and membership change, and resolved separately. However, the Raft algorithm cannot support a Byzantine system, which is likely more used in Blockchain.

According to [2], almost all distributed systems which support strong consensus use BFT protocols as the core algorithm, and PBFT is one of most classic, representative BFT algorithm. Compared with previous BFT algorithm, PBFT supports higher fault tolerance and it reduces the complexity of the algorithm to O(n^2).

PBFT is not only an excellent consensus algorithm which solves the traditional distributed consensus problems, but also becomes the subject of more and more formal methods for modelling and verification. For example, a new state-machine replication algorithm is used to prove PBFT's safety and liveness by describing asynchronous distributed systems [18] and stating their properties. Velisarios [19] is used to model and verify the safety of asynchronous Byzantine fault-tolerant protocols using Coq. An Event-B model [20] is used to prove the agreement and validity of a Byzantine agreement algorithms ZA and SM. Heard-Of model was implemented in Isabelle/HOL [21]. The work [22] investigates the current research on the use of formal methods to verify consensus protocols and believes model checking is used in the majority. However, most studies have focused on proving safety, sometimes with manual proofs only, while the correctness property for PBFT contains both safety and liveness properties in our work.

Other consensus mechanisms have been formally studied using "high-level" specification languages. For instance, TLA+ has been used to prove safety and liveness of Multi-Paxos [23] and safety of a variant of an abstract model of PBFT [24]. PSync [25] is a domain specific language based on the HO-model, and is used to implement crash fault-tolerant algorithms. ByMC is

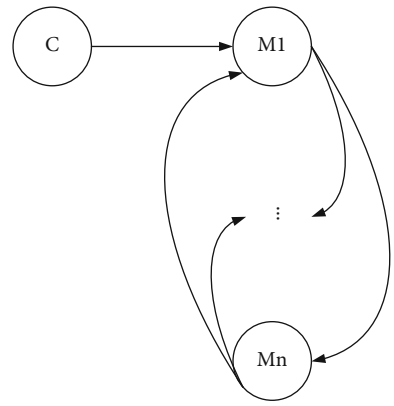


Figure 4: The communication between nodes and client.

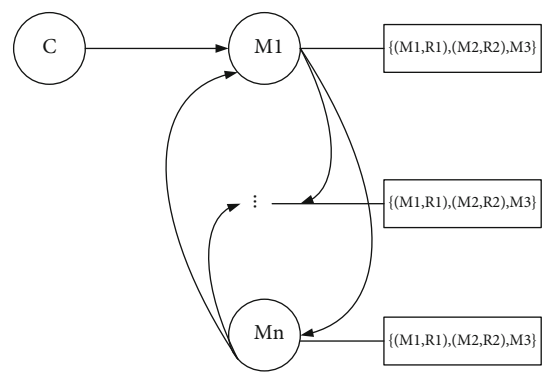


Figure 5: The buffers of each node.

counter-based model checker used to verify safety and liveness of fault-tolerant distributed algorithm [26–28].

Compared with the previous studies, we use horizontal refinement to build the PBFT incrementally. Then technical refinements are introduced to make Rodin produce the wanted proof obligations for correctness. Our work is the first to verify both the safety and liveness property of PBFT in a parameterized setting and use a much automated, mechanical proof.

4. Requirements for Models

Before developing a PBFT, we should consider the environment assumptions and the requirements for the constructed model. The protocol works in a network environment where the communications among the client, the primary node, and all other nodes are asynchronous and unbelievable. For example, messages can be dropped, altered, delayed, duplicated, or delivered out of order. Therefore, we need to

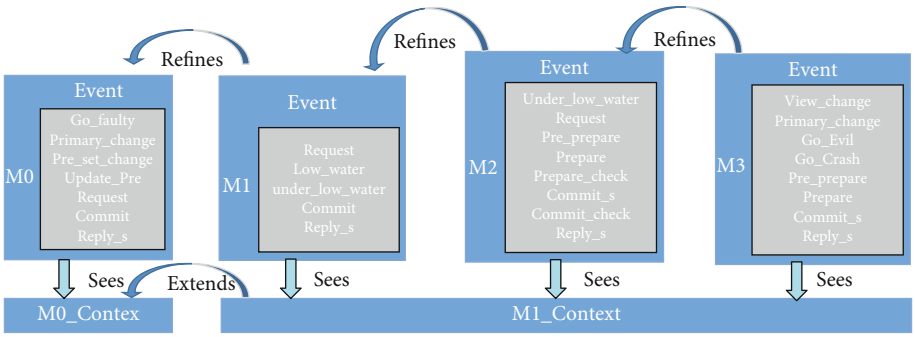


Figure 6: Model Development.

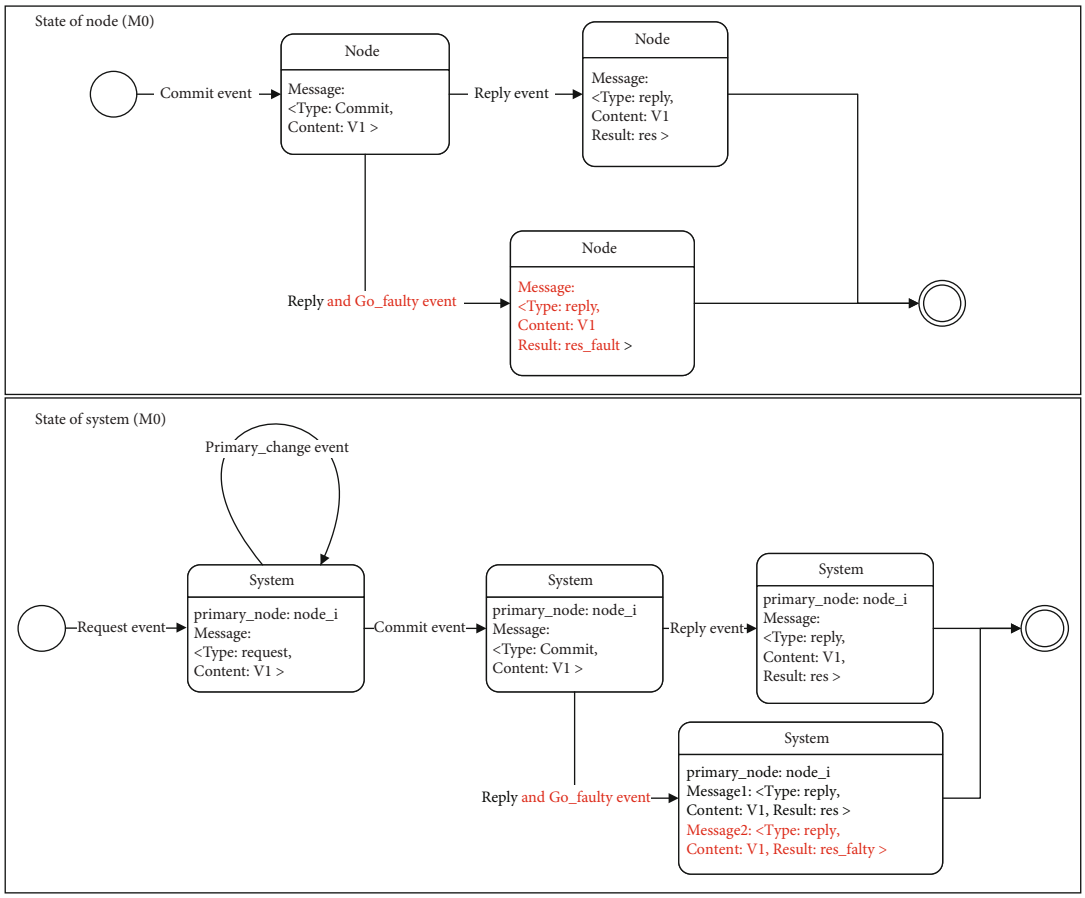


Figure 7: Initial model.

put forward more precise and appropriate requirements to ensure the correctness of the model. The refined model should safely and reliably run in such a complex environment. After carefully analyzing the algorithm, we propose the assumptions and the system requirements that we make about the various components of the algorithm:

(i) *A1*. The network consists of a finite set of nodes and clients forming a P2P architecture

The network is made up of a large but finite number of nodes, because the more distributed nodes participate in the consensus, the more reliable the system is. Considering the actual situation and the time complexity of the algorithm, the number of nodes cannot be infinite. These nodes are positioned on different sites that are connected in a distributed network architecture as indicated in Figure 4.

(ii) *A2*. Each node can send a message to all other nodes in the network and receive messages sent by any node

(iii) *A3*. Messages can be buffered in each node

As illustrated in Figure 5, all messages received by the node will be stored in a buffer which is associated with the node. In particular, messages that have been processed and

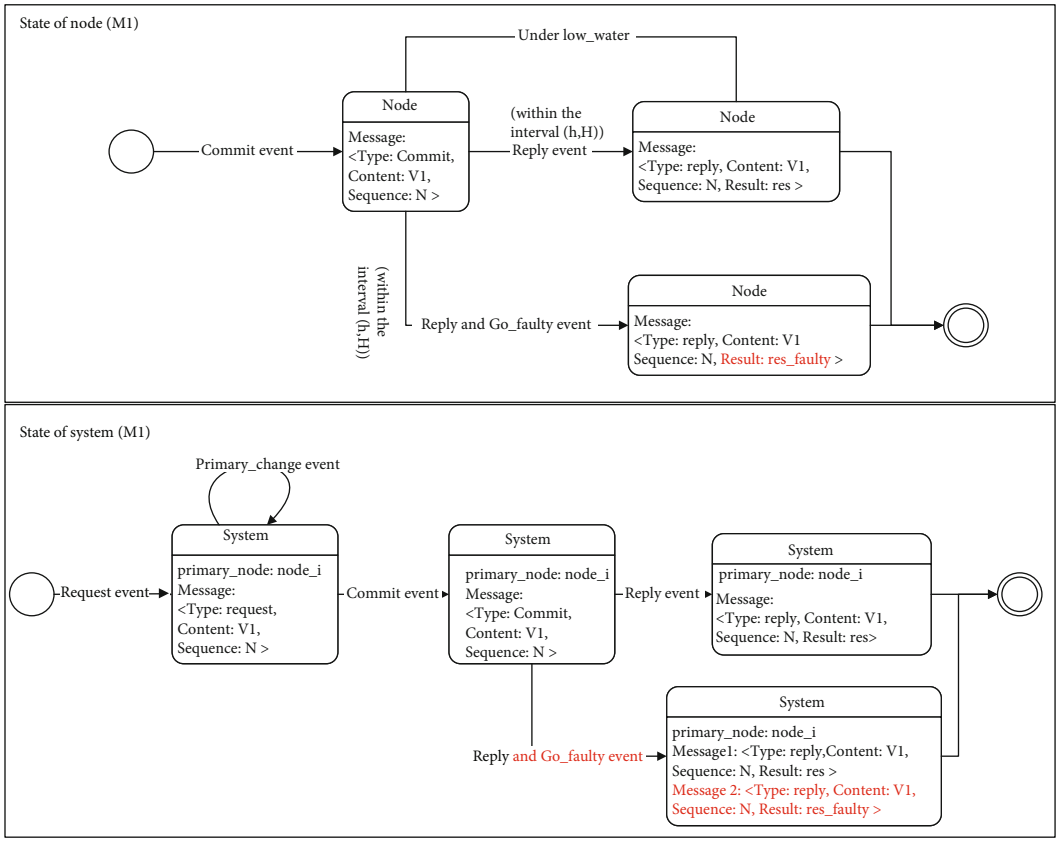


Figure 8: The first refinement model.

messages that are waiting for consensus will be placed in the buffers in order.

(iv) *A4*. Messages are reliable

A node (even a faulty one) cannot forge messages. It is up to the user signature and transfer methods to ensure that this is indeed the case. Usually, this means that forged messages can be created but are invalid (e.g. digital signatures violated) and, hence, forgery can always be detected.

(v) *R1*. For the same message, the results executed by all correct nodes are the same

If a message is executed by a CORRECT node, the result obtained should also be correct and unique. It guarantees the liveness and the safety of the system.

(vi) *R2*. If at most f nodes are faulty which means that they can send wrong execution results, and at most f nodes have crashed which means that they cannot forward messages, there should be at least $3f + 1$ nodes in total to ensure the reachability of a consensus

(vii) *A5*. The model is weak synchronous

We suppose that a peer node responds within a specific time, and this specific duration will not change. Messages may be discarded in the network, delayed, and arrive out of order.

(viii) *A6*. The number of messages waiting to be processed cannot exceed a certain threshold

This assumption reflects the setting of water mark interval. The client will stop sending new transactions when k messages are waiting to be processed, and it will continue to send messages when the number of messages waiting to be processed is less than k.

(ix) *R3*. All nodes may crash or go evil. When the primary node crashes, another correct node will be chosen to act as the primary node

This requirement guarantees the problem of view change in the PBFT and states that any node in the model can evolve from correct to faulty.

(x) *A7*. Each node has a unique number

We number all nodes, and if the primary node needs to be changed when it crashes, we can simply choose the next one as the new primary node.

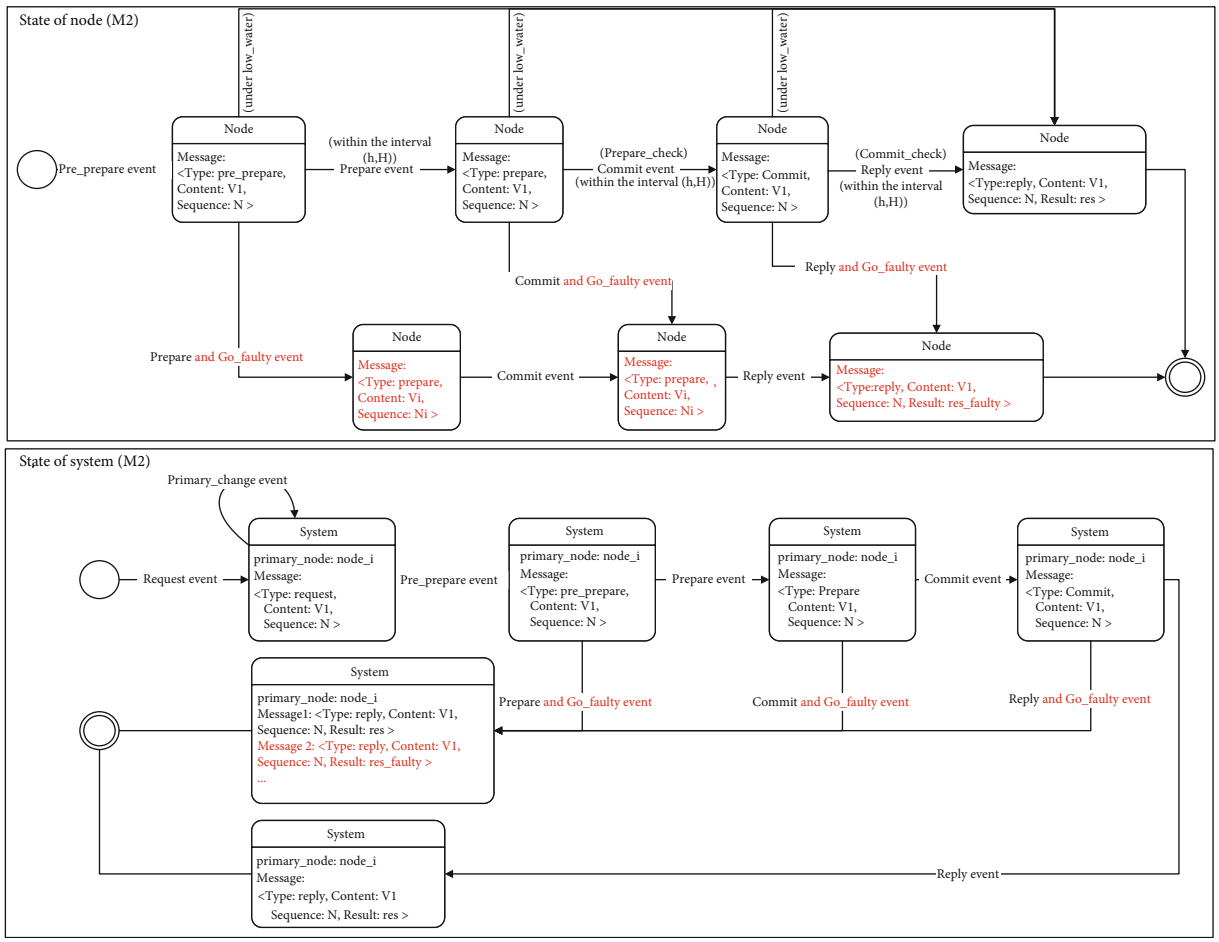


Figure 9: The second refinement model.

(xi) *R4*. A consensus process needs to go through five steps

5. Development

We now introduce the verification of the PBFT. As shown in Figure 6, we illustrate four steps of developing a formal model for the PBFT. Each model contains a machine and a context. The initial machine specifies the state of any successful run of the PBFT consensus algorithm where we prove the agreement and liveness properties, then M1 refines M0 with the concrete message type and the water mark interval definition. Next, M2 refines M1 with the complete consensus process and M3 refines M2 with more specific false behaviors of nodes.

Our initial model is very abstract compared to the PBFT. It only contains three stages: request, commit, and reply. And there are three kinds of messages. The request message and commit message has two items: Type and Content. Type is the set containing the three message types: request, commit, and reply. Content contains the effective contents of a message in PBFT. The reply message has an additional item: Result, which indicates the execution's result of the operation requested by the client. Figure 7 describes the state of the nodes and the system. The part of "State of node" focuses on the state after the primary node broadcasts the request message. And three events are needed: commit, reply and Go_faulty. In the commit event, the primary node broadcasts the message to all nodes in the distributed system. In the Reply event, a node executes the operation requested by the client. The event Go_faulty specifies the situation where the normal node becomes faulty during the commit and reply phases. The event Go_faulty only affects the final reply process on the contents of the message. In Figure 7, the message marked in red is caused by the Go_faulty event. The part of "State of system" additionally contains the request and primary_change event. In the request event, the client sends a request to the primary node. In the primary_change event, the system re-selects another node as the primary node.

The liveness property is guaranteed by the event primary_change which is fired when the primary node crashes. In this case, the system selects another node as the primary node so it will not enter a deadlock. The primary node in the system should belong to the set of correct nodes and behaves correctly. Otherwise, the event primary_change will be fired. The proof of the agreement property relies on the fact that the local views of correct nodes are identical at

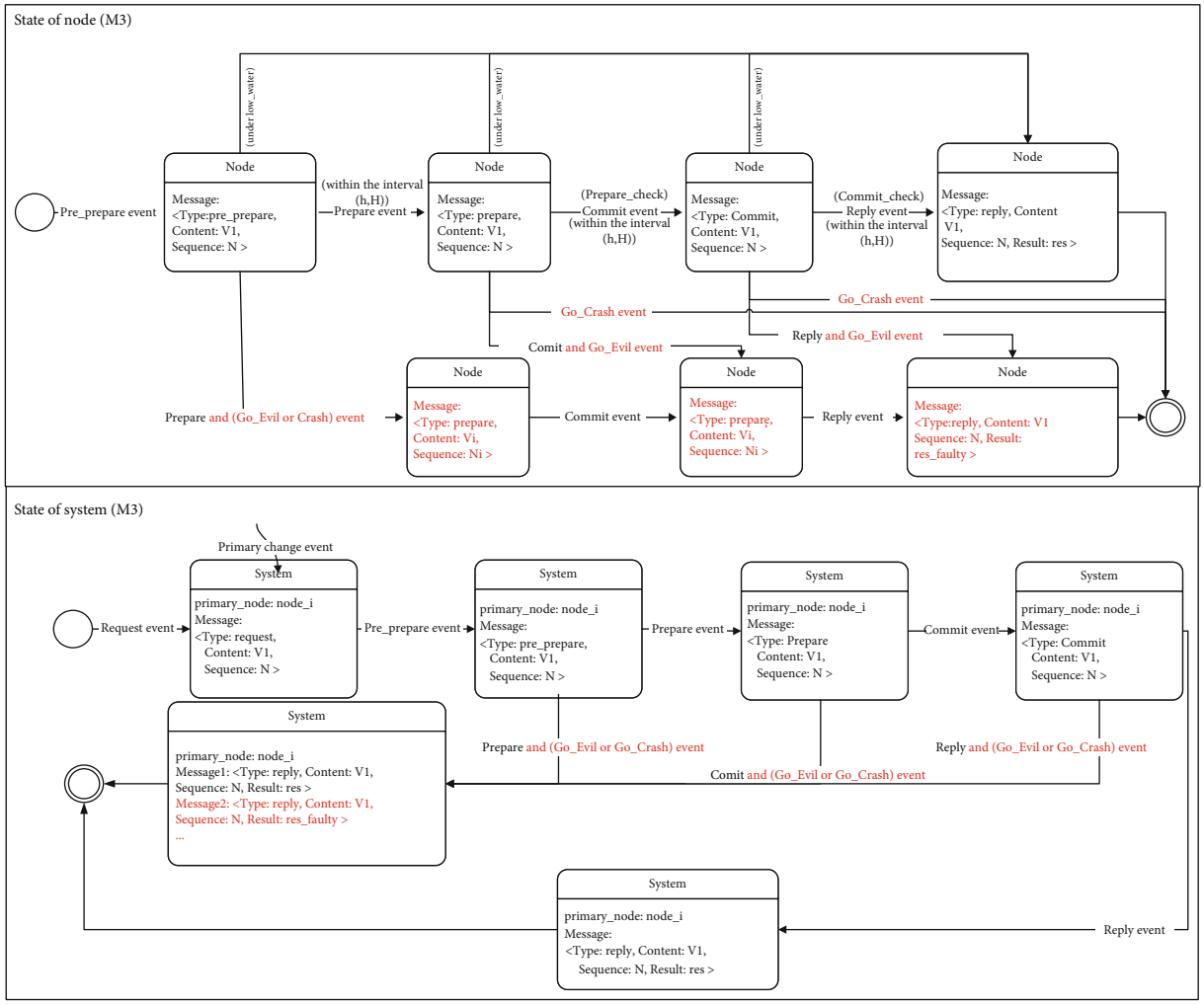


Figure 10: Diagram of the third refinement.

the end of any execution. To show this, we specify that more than $f + 1$ identical results of the same message from the system are regarded as valid responses to the client. Besides, another important thing is that the initial model emphasizes the asynchrony of the process where messages are sent by clients and the process of consensus between nodes.

As shown in Figure 8, the first refinement M1 mainly refines the message by adding the sequence number N which is assigned by the primary node. Besides, M1 imports the notion of low level of the water mark interval $[h, H]$. In Figure 8, we add the judging conditions: within the interval $[h, H]$ or under low-water, and we limit that all messages processed by nodes cannot exceed N by a specified value H which is the high level of the water mark interval. So that we can control the complexity of the entire model by modifying the high level of the water mark interval. When H is bigger, the model is more complicated for the asynchrony. Meanwhile, there are messages with the sequence number under low water level h, they will be updated automatically with the execution results stored in correct nodes' buffers.

As shown in Figure 9, The second refinement M2 refines the consensus process of the PBFT in M1. Compared with the previous model, it includes the complete consensus process of PBFT and adds two inspection conditions: In Prepare_check, each node checks if there are $2f + 1$ identical message have been received from different nodes; in

Sets
Message result
Constants
CORR NODES True_execute FAULTY
Axioms
@axm0 **NODES**⊆ℕ
@axm1 finite(**NODES**)
@axm2 **NODES**≠∅
@axm3 **CORR**⊆**NODES**
@axm4 **CORR**≠∅
@axm5 3∗card(**CORR**) ≥2∗card(**NODES**) +1
@axm6 **result**≠∅
@axm7 **FAULTY**∈**result**
@axm8 **True_execute**∈**message**↔**result**\{**FAULTY**}
End

Listing 1: Context of M0.

```
Invariants
@inv1 G∈NODES→(message↦result)
@inv2 partition(NODES,Faulty,corr)
@inv3 3∗card(corr)≥2∗card(NODES)+1
@inv4 cache∈NODES→ℙ(message)
@inv5 ∀i·i∈corr⇒dom(G(i))⊆cache(i)
@inv6 pre∈NODES
@inv7 Pre_set⊆NODES
@inv8 ∀n· n ∈ corr ⇒ G(n)⊆True_execute
@inv9 3∗card({i,j·i∈NODES∧j∈dom(G(i))∧(G(i))(j)=True_execute(j)|i})≥2∗card(NODES)+1
```

Listing 2: Invariants of M0.

```
Event Go_faulty
ANYnode
  Where
        @grd1 node∈corr
        @grd2 3∗card(corr\{node})≥2∗(NODES)+1
        @grd3 step1=Commit1∨step1=Reply1
  Then
        @act1 corr:=corr\{node}
        @act2 faulty:=faulty∪{node}
  End
```

Listing 3: Event 'Go_faulty'.

```
Event Primary_change
    Where
      @grd1 pre∉corr
      @grd2 NODES\Pre_set≠∅
    Then
      @act1 pre:=max(NODES\Pre_set)
      @act2 Pre_change:=TRUE
End
```

Listing 4: Event 'Primary_change'.

```
Event request
    Any m
    Where
      @grd1 m∈message
      @grd2 ∀i·(i∈NODES)⇒m∉cache(i)
    Then
      @act1 cache:=cache<+{pre↦cache(pre) ∪ {m}}
End
```

Listing 5: Event 'request'.

```
Event commit
    Anym node
    Where
      @grd1 m∈message
      @grd2 node∈NODES\{pre}
      @grd3 m∉cache(node)
      @grd4 m∈cache(pre)
    Then
      @act1 cache(node):=cache(node)∪{m}
End
```

Listing 6: Event 'Commit'.

Commit_check, each node checks if there are $2f + 1$ identical commit messages have been received from different nodes or itself. Because the event Go_faulty could fire during the events Prepare, Commit and Reply during the change of state of the node and system. Finally, the commits for execution could be faulty and the node reply with a wrong execution result of the message.

As shown in Figure 10, The third refinement M3 refines the M2 by importing concrete false behaviors of nodes through the events Go_Evil and Go_Crash. Any node in a crash cannot forward messages to other nodes or to the client, and the node will remain in its previous state. Any node that goes evil will forward false messages and reply with wrong execution results. In particular, the primary node cannot go evil; if it does, it will be switched. Which is formulated by requirement R3.

In the following sections, we will introduce the modeling details of the PBFT and explanations of the relevant codes, while some unimportant code details will not be presented.

5.1. The Initial Model. The purpose of the initial model is to define the success conditions for PBFT. This model should be as abstract as possible and specify the critical liveness and agreement properties to make them valid. We begin by defining the initial Event-B context as shown in Listing 1. In this context, we define a carrier set Message to represent

```
Event Reply_s
    Any node message
    Where
      @grd1 node∈NODES
      @grd2 node∉∅
      @grd3 cache[corr]≠∅
      @grd4 message∈message
      @grd5 message∈inter(cache[corr])
      @grd6 message∉dom(G(node))
    Then
      @act1 G:={TRUE↦G<+{node↦G(node)∪{message↦
True_execute(message)}},FALSE↦G<+{node↦G(node)∪ {message↦FAULTY}}}(bool(node∈corr))
  End
```

Listing 7:Event 'Reply_s'.

the different message formats introduced in Section 4. The carrier set Result represents the execution result produced by each node. It may contain many kinds of execution results in voting, calculation, election, etc., but they can all be regarded as elements of the set Result. To formalize the assumption A1, we define the constant NODES as the set of all network nodes and axiomatize that it is a finite and not empty subset of natural numbers. Besides, CORR is modeled as the set of all the correct initial nodes and it is a subset of NODES. To ensure that the initialization satisfies the premise of the PBFT, which bounds the number of faulty nodes, we add the axiom *@axm5* (3∗card(CORR) ≥2∗card (NODES) +1).

We define a relation True_execute from messages to results, which indicates the results that should be returned by a correct execution, and we consider the worst case, where the Byzantine node, noted as faulty nodes, return the special value FAULTY.

In our initial model, each node has a buffer where all the messages received and processed are stored. As shown in Listing 2, we formalize the final view of each node as G, which is a set of relations between each node to its result pairs. Since not all the messages contain their execution result, we define the partial function (Message⇸Result) from messages to results so that each node can save the messages being executed with their result in the buffer like {(M1, R1), (M2, R2) …}. At termination, each node i has its view $G(i)$. Besides, we define a variable Cache to serve as a buffer to store all the received messages for each node, which is a total function from nodes to messages. @inv5 specifies that the set of messages stored in G is a subset of that in Cache because some messages have been received but not yet executed.

The variable Pre contains the primary node and Pre_set is the set of all nodes which have served as a primary node in some rounds. The other important variables we have defined are corr and Faulty, the former is the set of all correct nodes and the latter contains all the faulty nodes. The union of these two sets constitutes the entire set NODES, and their intersection is the empty set. @inv8 specifies that all correct nodes will reply the correct execution results, @inv9 specifies the agreement property of the model. The agreement property emphasizes that the cardinality of the set of nodes reply-

```
Theorem @inv17 ∀i,j·i∈corr∧j∈dom(G(i))
                ⇒G(i)(j)=True_execute(j)
```

Listing 8: Invariant of M0.

ing with the correct execution results should exceed $f + 1$ (f being the number of faulty nodes), which can be specified by

$$\text{card}\left(\begin{array}{l} i, j \bullet i \in NODES \wedge j \in \text{dom}\ (G(i)) \wedge \\ (G(i))(j) = True_{execute(j)} \end{array} \middle| i\right) \geq \text{card}\ (NODES) - \text{card}\ (CORR) + 1. \quad (1)$$

With the invariant @inv3, we can simplify the formula (1) to the invariant @inv9:

$$3 * \text{card}\left(\begin{array}{l} i, j \bullet i \in NODES \wedge j \in \text{dom}\ (G(i)) \wedge \\ (G(i))(j) = True_{execute(j)} \end{array} \middle| i\right) \geq 2 * \text{card}\ (NODES) + 1. \quad (2)$$

We define the event Go_faulty in Listing 3 to convert a correct node to a faulty one as below.

The first guard @grd1 stipulates that the parameter should be an element of the set of correct nodes, the guard @grd2 specifies that the number of correct nodes should be greater than $(2^*\ \text{card}\ (\text{NODES}) + 1)/3$ after removing the node from the set corr. The third guard @grd3 specifies that the node can only be faulty during the phases Commit and Reply. Besides, we define the event Primary_change in Listing 4 which should be triggered before each client sends a message to the primary and the primary node is faulty. As shown, each node will be elected as the primary node in turn from high to low by taking the maximum value, and each node has the same chance of being elected as the master node. We define a variable Pre_set as the set of all the nodes selected as primary node. When all nodes have taken turns to become primary nodes, the variable Pre_set will be reset to an empty set. The primary node will be reelected from all nodes, and the variable Pre_set will continue to grow as

```
Context Ma1_ctx extends Ma0_ctx
Sets value
Constants contents Correct_value Faulty_value H
Axioms
  @axm1 value≠∅
  @axm2 finite(value)
  @axm3 partition(Value,Correct_value,Faulty_value)
  @axm4 card(Correct_value)=card(Faulty_value)
  @axm5 contents∈message⇸(ℕ×value)
  @axm6 ∀x,y·x∈message∧y∈message∧x≠y⇒
prj2(contents(x))≠prj2(contents(y))
  @axm7 ∀x·x∈message⇒(∃y·y∈message∧x≠y∧
        (prj1(contents(x))=prj1(contents(y))+1∨
          prj1(contents(x))=prj1(contents(y))−1))
  @axm8 ∀x,y·x∈message∧y∈message∧x≠y∧⇒
(prj2(contents(x))∈Correct_value∧prj2(contents(y))∈Faulty_value)∨
(prj2(contents(x))∈Faulty_value∧prj2(contents(y))∈Correct_value)
  @axm9 H=2
End
```

Listing 9: Context 'Ma1_ctx'.

new primary nodes are selected. This formalizes requirement 3.

For the event Request in Listing 5, the parameter m is the new message which will be added to the buffer, and @grd2 guarantees that m has not been received before. @grd3 guarantees that the primary node must be correct. Then we use a relational override to update the buffer of the primary node in @act1.

As shown in Listing 6, the event Commit describes the process of all nodes receiving and confirming messages from the primary node. In particular, it is an asynchronous process. The parameter m is the message sent by the primary node to all other nodes, and node is any node except the primary node. This event ensures the agreement property by requiring that the execution results of at least $f + 1$ different nodes are identical.

As shown in Listing 7, we define the event Reply, which will fire when the agreement property above holds. When fired, the event updates the global variable G with the new consensus result and the execution result. It is also modelled as an asynchronous process. According to the validity of the execution results, there should be at least $f + 1$ identical result from different nodes. Therefore, we propose the theorem @inv17 in Listing 8, which ensures the agreement property.

In our first refinement, we start to model the details of the protocol. To more specifically refine the serial number and content of the message defined in M0, we define a carrier set Value to represent different operations sent by the client in the context, and a constant called contents, which is an injection from *Message* to a set of pairs. In the set of pairs, the first element is a member of natural numbers and the second element is a member of Value. The natural number records the sequence number of the message sent by the client to the primary node. Together with the value, it ensures the uniqueness of each message.

```
  @inv1 n∈ℕ
  @inv2 view∈NODES→NODES
```

Listing 10: Invariant of M0.

```
Event Low_water
    Any S S1
    Where
      @grd1 S⊆dom(union(ran(G)))
      @grd2 S≠∅
      @grd3 S1⊆corr
      @grd4 3∗card(S1)≥2∗card(NODES)+1
      @grd5 ∀i,j·i∈S1∧j∈S⇒j∈dom(G(i))
    Then
      @act1 n:=max(dom(contents[S]))+1
  End
```

Listing 11: Event 'Low_water'.

```
Event under_low_water
    Any node m node1
    Where
      @grd1 node∈NODES
      @grd2 node1∈corr
      @grd3 m∈cache(node)
      @grd4 m∉dom(G(node))
      @grd5 prj1(contents(m))<n
      @grd6 m∈dom(G(node1))
    Then
      @act1 G:=G<+{node↦G(node)∪{m↦G(node1)(m)}}
End
```

Listing 12: Event 'under_low_water'.

```
Event request
  Refines request
    Any m1 m2
    Where
      @grd1 m1∈message
      @grd2 m2∈ℕ×value
      @grd3 m2=contents(m1)
      @grd4 m1∉cache(pre)
      @grd5 prj1(m2)=max(dom(contents[cache(pre)]))+1
      @grd6 prj1(m2)−n∈1..H
      @grd7 prj2(m2)∉ran(contents[cache(pre)])
    With
      @mm=m1
    Then
      @act1 cache:=cache<+{pre↦cache(pre) ∪ {m1}}
End
```

Listing 13: Event 'request'.

@inv7 specifies that it is a monotonically increasing function. Besides, we refine the Value type by two constants Correct_value and Faulty_value, which specify the condition where the faulty nodes forward the faulty value during the prepare and commit process (@inv6 and @inv8). @axm3 and @axm4 specify the relationship between the two constants. H is the high level of the water marks interval and we set it to 2 for tests in Listing 9.

For the machine, in Listing 10 we define a natural number variable n to represent the low level of the water mark interval, and a total function View from nodes to nodes, which represents the view of the primary node for each node.

As shown in Listing 11, the event Low_water is defined to change the variable n. S is the set of all messages which are under consensus and have at least $f + 1$ identical execution result from correct nodes(@grd1-@grd5). Then we assign the sum of 1 and the maximum value of the sequence number of messages in S to variable n.

Another relevant event we have defined is under_low_water. As shown in Listing 12, for any node in the network, if there is any message that has not been executed whose sequence number is less than n, then these messages will be updated with the execution results from other correct nodes(@act1).

The event Request, Commit, and Reply are all refined with the specific data types of messages and tighter restrictions. For Request in Listing 13, we define two parameters m1 and m2. m1 is any message sent to the primary node by the client and m2 is the content of the message. @grd3 guarantees that the message sent by the client is different from the messages which have been received by any node. @grd4 and @grd7 guarantee that the client cannot send duplicate messages to the primary node. @grd5 specifies that the sequence number of the messages sent is monotonically increasing by 1. @grd6 formalize the assumption 6, and it indicates that the client should wait when there is one more message than n in processing, therefore, this distributed algorithm can run stably without causing the system to crash due to weak synchrony. @act1 updates the cache of the primary node.

```
@grd1 prj1(contents(m1))−n∈0..H
@act1 view(node1):=pre
```

Listing 14: Guard and action of event commit.

For the event Commit, we add the constraint of water mark interval by @grd1 and @act1 as shown in Listing 14 to update each node's view of the primary node. It is similar for event Reply_s, we add the constraint of water mark interval.

5.2. The Second Refinement. The second refinement focuses on the consensus steps. We modelled the consensus phrase by these five steps: Request, Pre-prepare, Prepare, Commit, and Reply. Request refines Request, Pre_prepare refines Commit, and Reply_s refines Reply_s. Besides, event Prepare, Prepare_check, Commit_s, and Commit_check are defined to accomplish the whole process. These steps are all under weak synchrony corresponding to the real situation.

We introduce five new variables G_p, G_pre, G_r, G_pre_check, and G_r_check with the invariants given by Listing 15. G_p is a total function used to save the message information received during the request and pre-prepare process, which refines the variable cache in the previous machine, @inv6 specifies their relationship. G_pre is the total function from nodes to another partial function which maps nodes to a set of message information. It serves as a cache during the prepare process. G_pre_check is a total function from nodes to message information, which serves as a cache to store the received message under consensus. G_r and G_r_check are similar to G_pre and G_pre_check, and they are used during the event Commit and event Commit_check.

The event Prepare of Listing 16 models the process where all nodes in NODES\{Pre} send a message to nodes

```
@inv1 G_p∈NODES→ℙ(ℕ×(ℕ×value))
@inv2 G_pre∈NODES→(NODES⇸ℙ(ℕ×(ℕ×value)))
@inv3 G_r∈NODES→(NODES⇸ℙ(ℕ×(ℕ×value)))
@inv4 G_pre_check∈NODES→ℙ(ℕ×(ℕ×value))
@inv5 G_r_check∈NODES→ℙ(ℕ×(ℕ×value))
@inv6 ∀i,j·i∈dom(cache)∧j∈cache(i)⇔
i∈dom(G_p)∧contents(j)∈ran(G_p(i))
```

Listing 15: Invariants of M2.

```
Event prepare
    Any m_c m_f send rec
    Where
      @grd1 send∈NODES\{pre}
      @grd2 rec∈NODES
      @grd3 send≠rec
      @grd4 m_c∈ℕ×(ℕ×value)
      @grd5 m_c∈G_p(send)
      @grd6 prj1(prj2(m_c))−n∈0..H
      @grd7 m_f∈ℕ×(ℕ×value)
      @grd8 prj1(m_f)=prj1(m_c)
      @grd9 prj2(prj2(m_f))∈Faulty_value
      @grd10 prj1(prj2(m_f))−n∈0..H
      @grd11 send∈dom(G_pre(rec))
      @grd12 m_f∉G_pre(rec)(send)
      @grd13 m_c∉G_pre(rec)(send)
    Then
      @act1 G_pre:={TRUE↦G_pre<+{rec↦G_pre(rec)∪{send↦{m_c}}},
FALSE↦G_pre<+{rec↦G_pre(rec)∪{send↦{m_f}}}}(bool(send∈corr))
  End
```

Listing 16: Event 'Prepare'.

```
Event Prepare_check
    Any node m
    Where
      @grd1 node∈NODES
      @grd2 m∈ℕ×(ℕ×value)
      @grd3 prj1(prj2(m))−n∈0..H
      @grd4 m∈union(ran(G_pre(node)))
      @grd5 3 ∗ card({i|i∈dom(G_pre(node))∧m∈G_pre(node)(i)})≥2∗card(NODES) +1
      @grd6 prj1(m)=view(node)
    Then
      @act1 G_pre_check(node):=G_pre_check(node)∪{m}
End
```

Listing 17: Event 'Prepare_check'.

in *NODES*, any faulty node can forward false messages with the same sequence number, view, but the different message value (@grd7, @grd8, @grd9, @grd12, and @grd13). To make this process keep weak asynchronous, @grd6 is used to guarantee that all messages are sent with a sequence number that belongs to the water mark interval.

The event Prepare_check in Listing 17 is modelled to check the validity of the messages received during the event

```
@inv1 partition(Faulty,Crash,Evil)
@inv2 card(corr)≥card(crash)+1
@inv3 card(corr)≥card(evil)+1
@inv4 View_change∈NODES→(NODES⇸ℕ)
```

Listing 18: Invariants of M3.

```
Event view_change
  Any send rec
  Where
  @grd1 pre∈NODES
  @grd2 pre∈crash
  @grd3 send∈NODES
  @grd4 rec∈NODES
  @grd5 send≠rec
  @grd6 send≠pre∧rec≠pre
  Then
  @act1
View_change(rec):=View_change(rec)<+{send↦max(NODES\Pre_set)}
  End
```

Listing 19: Event 'view_change'.

Prepare: at least $f+1$ message received with the same sequence number should be identical. The message that passes the check is stored in variable G_pre_check. It is also a weak synchronous process by the guard @grd3.

The events Commit and Commit_check have a similar logic, except for the different stored data structure. Therefore, we do not detail them here.

5.3. The Third Refinement. Based on the former three models, we have achieved the agreement property of the PBFT. Therefore, the third refinement focuses on the view change function, which is very important to ensure the liveness property. We refine the Faulty variable with two variables Crash and Evil. Crash is the set of nodes that can receive messages but cannot forward them. Evil is the set of nodes that will forward false values during Prepare and Commit, and reply the incorrect execution results to the client during the Reply phase. As shown in Listing 18, @inv1 specifies the relation among Faulty, Crash, and Evil. @inv2 and @inv3 specify the agreement property, that is to say, the cardinality of the corr should be greater than or equal to the cardinality of Crash and cardinality of Evil, which is consistent with the @inv3 (3∗card (corr)≥2∗card(NODES)+1) in the initial model. In this refined model, once the primary node crashes, there should be an event to change the primary node, and all other nodes should change their view so that the system will not enter infinite waiting. Therefore, we define a variable View_change, which is a total function from nodes to another partial function, which maps nodes to natural numbers.

We defined the event view_change in Listing 19. It takes a sender, a receiver, and the primary node as parameters, @grd2 indicates that the primary node crashes. @act1 specifies that any node in NODES sends the message which declares the next primary node to all other nodes in NODES. It is an asynchronous process. This event will be fired until all guards in event Primary_change are met.

As shown in Listing 20, the event Primary_change models that the next primary node receives at least $f+1$ identical messages from different nodes, which indicates the next primary. We introduce a variable S, @grd1 specifies that S is a subset of the domain of the following primary node's View_change. @grd2 specifies that nodes in S send the message which indicates the primary node max(NODES\Pre_set), and we elect the master node in turn according to the node number(@act1).

We define another two events Go_Evil and Go_Crash to refine the event Go_faulty. In particular, they specify that the primary node will not become evil, but can crash. And we refine the consensus process like Pre_Prepare with the guard Pre∉Crash, which means that if the primary node crashes, it cannot forward messages. Then the event Primary_change may fire. As for event Prepare, we add the guard send∉Crash, which means that crashed nodes cannot send messages. In our system, we allow at most f nodes to crash as specified in R2.

6. Proof Engineering

The proofs of the PBFT model have been mechanized thanks to the Rodin framework. The framework in here was used as a proof obligation generator and as an environment to discharge generated proofs through user interaction. The framework contains built-in solvers and is also connected to external SMT solvers. The basic machinery available within Rodin allows for the automatic generation of proof obligations for invariants, event convergence, refinements, and theorems. An invariant is true initially and preserved by each event. Event convergence is established by introducing a variant, an expression yielding a natural number or a finite set. Each convergent event must decrease the variant strictly. Event-B also provides anticipated events which do not increase the variant. We use these features to generate proof obligations ensuring the stabilization property of the PBFT.

The main properties of PBFT are agreement and liveness. The former states that regardless of the view, results provided by correct nodes for a given message should be the same. The assumption R1 has been proved in Section 4.

The liveness property holds under the hypotheses that messages sent by clients will finally be processed and be weak fairness [29] over events E in a context H. Learning from the proof method proposed in [12], we propose proof obligations for establishing the stabilization of a given property Q:

(1) $\wedge_{e\in E\setminus C} H\wedge V=v \Rightarrow [e](V\preccurlyeq v)$ (Generated by Rodin for anticipated events): anticipated events do not increase the variant

(2) $\wedge_{e\in C} H\wedge V=v \Rightarrow [e](V\prec v)$ (Generated by Rodin for convergent events): convergent events make the variant decrease

(3) $H\wedge V\neq\varnothing \Rightarrow \bigvee_{e\in C} enabled(e)$(Manually added as a theorem to be proved): some of the convergent events are enables while the variant is not empty

(4) $H\wedge V=\varnothing \Rightarrow Q$(Manually added as a theorem to be proved): when the variant is empty, the targeted property is satisfied

$C\subseteq E$ is a set of convergent events, and V is a set expression called variant. Given an event e, $[e](p)$ denotes the

```
Event Primary_change refines Primary_change
  Any S
  Where
    @grd1 S⊆dom(View_change(max(NODES\Pre_set)))
    @grd2 ∀i·i∈S⇒union(ran(View_change))(i)=max(NODES\Pre_set)
    @grd3 3∗card(S)≥2∗card(NODES) +1
  Then
    @act1 pre:=max(NODES\Pre_set)
    @act2 Pre_change:=TRUE
End
```

Listing 20: Event 'Primary_change'.

```
Theorem
  @theorem 1: G∈NODES×{{MM↦RR}}∧
  CORR∈CORR∧
  Card({n|n∈corr∧G(n)(m)=∅})≠∅∧≠∅∧
      G_pre∈NODES→(NODES→ℙ(ℕ×(ℕ×value))) ∧
      G_r∈NODES→(NODES→ℙ(ℕ×(ℕ×value))) ∧
      G_pre_check∈NODES→ℙ(ℕ×(ℕ×value)) ∧
      G_r_check∈NODES→ℙ(ℕ×(ℕ×value))
      ⇒
      (node2∈NODES∧
      m2∈ℕ×(ℕ×value) ∧
      message2∈message∧ contents(message2)=prj2(m2) ∧
      m2∈G_r_check(node2) ∧
      Cache[corr] ≠ ∅∧
      message2∈inter(cache[corr]) ∧
      message2∉dom(G(node2)) ∧ prj1(prj2(m2))=n))
  @theorem2 G∈NODES×{{MM↦RR}}∧
    CORR∈CORR∧
    Card({n|n∈corr∧G(n)(m)=∅})≠∅∧
    G_pre∈NODES→(NODES⇸ℙ(ℕ×(ℕ×value))) ∧
    G_r∈NODES→(NODES→ℙ(ℕ×(ℕ×value))) ∧
    G_pre_check∈NODES→ℙ(ℕ×(ℕ×value)) ∧
    G_r_check∈NODES→ℙ(ℕ×(ℕ×value))
    ⇒
    (node∈NODES∧
    node1∈corr∧
    m∈cache(node) ∧
    m∉dom(G(node)) ∧
    prj1(contents(m))<n∧
    m∈dom(G(node1)))
  @theorem3 G∈NODES×{{MM↦RR}}∧
    CORR∈CORR ∧
    Card(n|n∈corr∧G(n)(m) = ∅ )=∅ ∧
    G_pre∈NODES→(NODES→ℙ(ℕ×(ℕ×value))) ∧
    G_r∈NODES→(NODES→ℙ(ℕ×(ℕ×value))) ∧
    G_pre_check∈NODES→ℙ(ℕ×(ℕ×value)) ∧
    G_r_check∈NODES→ℙ(ℕ×(ℕ×value))
    ⇒
      3∗card({i,j·i∈NODES∧j∈dom(G(i))∧(G(i))(j)=True_execute(j|i})≥2∗card(NODES)+1
```

Listing 21: Theorems to prove the liveness.

weakest precondition ensuring e terminates in a state satisfying the predicate p. The correctness of the liveness property relies on weak fairness between two classes of events: enabled convergent events should eventually be fired for the variant to decrease, and anticipated events do not increase the variant. A variation of this proof rule may be used when Q is reached before the variant V becomes empty. Obligation (3) can be changed as follows:

Table 1: Relation between models and requirements.

Models	Assumption or requirements
M0	A1, A2, A3, A4, R1, A5, A7
M1	A1, A2, A3, A4, R1, A5, A7, A6
M2	A1, A2, A3, A4, R1, A5, A7, A6, R4
M3	A1, A2, A3, A4, R1, A5, A7, A6, R4, R2, R3

Table 2: Proof statistics.

Model	Number of proof obligations	Automatically discharged	Interactively discharged
Context	7	3	4
Initial model	45	15	30
1st refinement	44	34	10
2nd refinement	87	52	35
3rd refinement	36	19	17
Total	219	123	96

3a. $H \wedge \neg Q \Rightarrow \bigvee_{e \in C}\text{enabled}(e)$(manually added as a theorem to be proved): some of the convergent events are enabled while the targeted property is not reached

3b. $H \wedge \neg Q \Rightarrow \bigwedge_{e \in C}[e](Q)$ (generated by Rodin if Q is declared as invariant) Q is stable

For a message m sent by the client, but not yet confirmed by agreement, we define the set V as follows:

$$\mathrm{V} = \mathrm{n} | \mathrm{n} \in \mathrm{corr} \bigwedge \mathrm{G}(\mathrm{n})(\mathrm{m}) = \varnothing. \tag{3}$$

$G(n)$ represents a partial function from the message to its executing results in Listing 21. If the node n has not executed the message m, $G(n)(m)$ should be an empty set. Otherwise, $G(n)(m)$ will get an execution value as the result sent to the client. Then we take the cardinality of V as the variant, and the theorems manually added are:

@theorem 1 specifies $H \wedge V \neq \varnothing \Rightarrow V_{e \in C}\text{enabled}(Reply\_s)$, which means that the event Reply_s must be enabled to decrease the variant. @theorem 2 specifies $\mathrm{H} \wedge \mathrm{V} \neq \varnothing \Rightarrow \mathrm{V}_{e \in C}\text{enabled}($under_low_water), which means that the event under_low_water must be enabled to decrease the variant. @theorem 3 specifies $H \wedge V = \varnothing \Rightarrow Q$, where Q is the agreement property.

7. Conclusion

We list each model with assumptions or preconditions that they have implemented in Table 1. The initial abstract model is significant in achieving main environment specifications so that the refined models can successfully enrich the details of requirements towards the final execution model or codes. It is important to point out that our final model can also be refined by adding the events of modifying the number of nodes, which are similar to the consensus process but with a change to the total number of nodes.

In Event-B, the system should satisfy the termination property, which means there should be a suitable variant to guarantee the convergence of the system. Our model uses the constant function contents to store the execution results for every node. Initially, we define a set that contains all messages that are not yet agreed on and executed at a specific time. Then, it will decrease until it becomes an empty set.

Another important thing is that our model ignores the difference in computing performance of each node, which is reflected in the model by the fact that message transmission and execution will be complete in one instant. Therefore, we do not need to consider the situation that often occurs in actual development. For example, a correct node may be slow due to poor performance, and it is not straightforward in specifying a waiting time.

In Table 2, we give proof statistics of the development of the Rodin tool. These statistics measure the model's size, the proof obligations generated and discharged by the Rodin platform, and those inter-a ctively proved. The table shows that 56% (123/219) of automatic proof is achieved. These results are acceptable since our formal model uses the card function to restrict the number of the relationship between nodes, which complicates the verification. Also, there are set comprehension, disjunctions, and strict subsets, so many proof obligations cannot be proved automatically.

In this paper, we proposed a mechanized correctness proof of the PBFT, widely used in blockchain systems. We formally developed the PBFT model by horizontal refinement by implementing core mechanisms like primary node change, water mark interval, and parameterized message types. The result shows that if the proportion of Byzantine nodes is under 1/3, the system can reach an agreement finally, and it will not fall into a deadlock for each message. This work provides a reusable model for developing consensus mechanisms with detailed message types rather than just values. Refactoring the development to use simpler datatypes may lead to improved levels of automatic proof and therefore improve the potential for reuse. However, there are still some limitations. For example, suppose the consensus protocol of the consortium chain makes significant changes to PBFT. In that case, there is no guarantee that the changed parts can also meet the verification conclusions in this article. Moreover, the manual proof effort required by this work may be too high to be reused in more complex developments.

In future work, we intend to build adversary models of distributed systems to test the reliability of PBFT, these are Sybil attacks, DDoS, etc. Another line of work is generating consensus protocol code for target systems and developing automated support for stabilization proofs.

Data Availability

This is because all functions except the repeated code are included in the lists, and tables show the results.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Project of Science and Technology Major Project of Yunnan Province (202103AN080001-001, 202002AA100007, 202002AD080003), Yunnan Key Laboratory of Blockchain Application Technology (202105AG070005), and State Key Laboratory of Software Development Environment (no. SKLSDE-2022ZX-11).

References

[1] N. Satoshi, "Bitcoin: a peer-to-peer electronic cash system," 2008, https://bitcoin.org/bitcoin.pdf.

[2] Y. Z. Liu, J. W. Liu, Z. Y. Zhang, T. G. Xu, and H. Yu, "Overview on blockchain consensus mechanisms," *Journal of Cryptologic Research*, vol. 6, no. 4, pp. 395–432, 2019.

[3] 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, Philadelphia, PA, USA, 2013.

[4] A. N. Bessani, J. Sousa, and E. A. P. Alchieri, "State machine replication for the masses with BFT-SMART," in *2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, pp. 355–362, Atlanta, GA, USA, 2014.

[5] T. Distler, C. Cachin, and R. Kapitza, "Resource-efficient Byzantine fault tolerance," *IEEE Transactions on Computers*, vol. 65, no. 9, pp. 2807–2819, 2016.

[6] M. Jakobsson and A. Juels, "Proofs of work and bread pudding protocols," in *Secure Information Networks: Communications and Multimedia Security*, pp. 258–272, Kluwer Academic Publishers, 1999.

[7] S. King and S. Nadal, "PPCoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake," 2012, https://bitcoin.peryaudo.org/vendor/peercoin-paper.pdf.

[8] M. Castro and B. Liskov, "Practical Byzantine fault tolerance and proactive recovery," *ACM Transactions on Computer Systems*, vol. 20, no. 4, pp. 398–461, 2002.

[9] J. R. Abrial, *Modeling in Event-B: System and Software Engineering*, Cambridge University Press, 2009.

[10] J. R. Abrial, M. Butler, S. Hallerstede, T. S. Hoang, F. Mehta, and L. Voisin, "Rodin: an open toolset for modelling and reasoning in event-B," *International Journal on Software Tools for Technology Transfer*, vol. 12, no. 6, pp. 447–466, 2010.

[11] M. J. Fischer, N. A. Lynch, and M. S. Paterson, "Impossibility of distributed consensus with one faulty process," *Journal of the ACM*, vol. 32, no. 2, pp. 374–382, 1985.

[12] J.-P. Bodeveix, J. Brunel, D. Chemouil, and M. Filali, "Mechanically verifying the fundamental liveness property of the chord protocol," in *23rd International Symposium on Formal Methods (FM 2019)*, Springer, Porto, Portugal, 2019.

[13] E. A. Akkoyunlu, K. Ekanadham, and R. V. Huber, "Some constraints and tradeoffs in the design of network communications," in *Proceedings of the Fifth Symposium on Operating System Principles (SOSP'75)*, pp. 67–74, Austin, Texas, USA, 1975.

[14] L. Lamport, R. Shostak, and M. Pease, "The Byzantine generals problem," in *Concurrency: The Works of Leslie Lamport*, pp. 203–226, 2019.

[15] L. Lamport, "Paxos made simple," *ACM SIGACT News*, vol. 32, no. 4, pp. 51–58, 2001.

[16] F. Yang, W. Zhou, Q. Wu, R. Long, N. N. Xiong, and M. Zhou, "Delegated proof of stake with downgrade: A secure and efficient blockchain consensus algorithm with downgrade mechanism," *IEEE Access*, vol. 7, pp. 118541–118555, 2019.

[17] D. Ongaro and J. Ousterhout, "In search of an understandable consensus algorithm," in *2014 USENIX Annual Technical Conference (USENIX ATC 14)*, pp. 305–319, USENIX Association, Philadelphia, PA, 2014.

[18] M. Castro, "Practical Byzantine fault tolerance," Technical Report MIT-LCS-TR-817, Laboratory for Computer Science, Cambridge, MA, 2001.

[19] V. Rahli, I. Vukotic, M. Völp, and P. Esteves-Verissimo, "Velisarios: Byzantine fault-tolerant protocols powered by Coq," in *Programming Languages and Systems ESOP 2018*, A. Ahmed, Ed., vol. 10801 of Lecture Notes in Computer Science, 2018Springer, 2018.

[20] R. Krenický and M. Ulbrich, "Deductive verification of a Byzantine agreement protocol, Tech. Rep. 2010-7 Karlsruhe Institute of Technology, Department of Computer Science," Institute for Theoretical Computer Science, Karlsruhe, Germany, 2010.

[21] B. Charron-Bost, H. Debrat, and S. Merz, "Formal verification of consensus algorithms tolerating malicious faults," in *Stabilization, Safety, and Security of Distributed Systems*, vol. 6976, pp. 120–134, Springer, 2011.

[22] N. Ge, Y. K. He, S. M. Zhai, X. Z. Li, and L. Zhang, "Formal verification of consensus protocol: a survey and perspective," *Journal of Software*.

[23] S. Chand, Y. A. Liu, and S. D. Stoller, "Formal verification of Multi-Paxos for distributed consensus," in *FM 2016: Formal Methods*, vol. 9995 of Lecture Notes in Computer Science, , pp. 119–136, Springer, 2016.

[24] L. Lamport, "Byzantizing Paxos by refinement," in *Proceedings of International symposium on distributed computing*, pp. 211–224, Berlin, Heidelberg, 2011.

[25] C. Dragoi, T. A. Henzinger, and D. Zufferey, "PSync: a partially synchronous language for fault-tolerant distributed algorithms," *ACM SIGPLAN Notices*, vol. 51, no. 1, pp. 400–415, 2016.

[26] I. Konnov, M. Lazić, H. Veith, and J. Widder, "A short counterexample property for safety and liveness verification of fault-tolerant distributed algorithms," in *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages*, pp. 719–734, Paris, France, 2017.

[27] I. V. Konnov, H. Veith, and J. Widder, "On the completeness of bounded model checking for threshold-based distributed algorithms: reachability," *Information and Computation*, vol. 252, pp. 95–109, 2017.

[28] I. Konnov, H. Veith, and J. Widder, "SMT and POR Beat Counter Abstraction: Parameterized Model Checking of Threshold-Based Distributed Algorithms," in *Computer Aided Verification. CAV 2015. Lecture Notes in Computer Science, vol 9206*, D. Kroening and C. Păsăreanu, Eds., pp. 85–102, Springer, Cham, 2015.

[29] L. Lamport, *Specifying Systems: The TLA+ Language and Tools for Hardware and Software Engineers*, Addison-Wesley Longman Publishing Co., Inc, 2002.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 8698009, 21 pages
https://doi.org/10.1155/2022/8698009

Research Article

A Consensus Algorithm Based on Risk Assessment Model for Permissioned Blockchain

Xiaohui Zhang,[1] **Mingying Xue**,[2] **and Xianghua Miao**[1,3]

[1]*Faculty of Information Engineering and Automation, Kunming University of Science and Technology, Kunming, China*
[2]*Faculty of Management and Economics, Kunming University of Science and Technology, Kunming, China*
[3]*Computer Technology Application Key Laboratory of Yunnan Province, Kunming, China*

Correspondence should be addressed to Mingying Xue; 20202209073@stu.kust.edu.cn

Received 1 December 2021; Revised 6 April 2022; Accepted 29 July 2022; Published 26 August 2022

Academic Editor: Liquan Chen

Blockchain is characterized by privacy, traceability, and security features as a novel framework of distributed ledger technologies. Blockchain technology enables stakeholders to conduct trusted data sharing and exchange without a trusted centralized institution. These features make blockchain applications attractive to enhance trustworthiness in very different contexts. Due to unique design concepts and outstanding performance, blockchain has become a popular research topic in industry and academia in recent years. Every participant is anonymous in a permissionless blockchain represented by cryptocurrency applications such as Bitcoin. In this situation, some special incentive mechanisms are applied to the permissionless blockchain, such as "mined" native cryptocurrency to solve the trust issues of the permissionless blockchain. In many use cases, permissionless blockchain has bottlenecks in transaction throughput performance, which restricts further application in the real world. A permissioned blockchain can reach a consensus among a group of entities that do not establish an entire trust relationship. Unlike permissionless blockchains, the participants must be identified in permissioned blockchains. By relying on the traditional crash fault-tolerant consensus protocols, permissioned blockchains can achieve high transaction throughput and low latency without sacrificing security. However, how to balance the security and consensus efficiency is still the issue that needs to be solved urgently in permissioned blockchains. As the core module of blockchain technology, the consensus algorithm plays a vital role in the performance of the blockchain system. Thus, this paper proposes a new consensus algorithm for permissioned blockchain, the Risk Assessment-based Consensus (RAC) protocol, combined with the decentralized design concept and the risk-node assessment mechanism to address the unbalance issues of performance in speed, scalability, and security.

1. Introduction

Bitcoin is a type of digital money that has taken the world by storm. A scholar first proposed Bitcoin in 2008 under the pseudonym of Satoshi Nakamoto, and then the Bitcoin system was released on the Internet [1]. There are no centralization management servers and third-party credit endorsement organizations. Bitcoin has been operating stably for more than ten years. These characteristics demonstrate the potential advantage of the blockchain technology behind the Bitcoin system. Blockchain is a decentralized distributed ledger that generates and stores data in blocks and constructs a chain structure in chronological order. The security and immutability of blockchain are grounded in cryptography, smart contract, P2P network, and consensus protocol [2]. Blockchain has brought hope to solve the issues such as privacy, security, and trustworthiness in distributed ledger technologies [3]. In recent years, more and more researchers have applied blockchain to other scenarios, such as supply chain [4], information security [5], data security [6], and Internet of things [7]. With the development of blockchain technology, a new business model is emerging based on blockchain to help a group of entities eliminate the dependence on centralization certification organizations [8]. It is foreseeable that blockchain will gradually become an indispensable part of the future Internet.

There is no central organization to undertake the data verification work. Every stakeholder must record the same correct data because a peer-to-peer architecture was used in the decentralized distributed system like blockchain. However, it is difficult to maintain the same content and sequence of transactions in all the participants due to the different status of the participants and the network environment in which they are located [9]. In addition, some participants may be attacked as Byzantine nodes to obstruct transmission. Therefore, blockchain technology also brings new problems to the system. The solution to these problems relies heavily on the consensus algorithm because the consensus algorithm plays a vital role in the performance of the blockchain system [10].

For example, permissionless blockchain such as Bitcoin employs the methods that "mined" the cryptocurrency using their computing powers to mitigate the absence of trust. PoW (Proof-of-Work) [1], PoS (Proof of Stake) [11], and DPoS (Delegated Proof of Stake) [12] are classified as permissionless blockchain consensus protocols. However, PoW has limitations in computing power consumption and small throughput. In addition, the PoW consensus may suffer the tailored attack behavior such as 51% attacks [13]. Although the PoS and DPoS solve the waste of resources in PoW, there are still problems such as low efficiency [14]. Permissioned blockchain is more suitable for high real-time applications due to its high transaction throughput performance and low transaction confirmation latency [15]. Permissioned blockchain can use classic consensus algorithms, such as Raft [16] and PBFT [17], to reach the consensus among entities because all participants must be identified. PBFT has not been widely used in real-world projects due to high consensus cost and poor scalability in many use cases. The raft can only be used in non-Byzantine environments with only honest network nodes [18]. The above examples all show that the many consensus algorithms unable to meet the Quality of Service (QoS) demands of specific scenarios are an important reason that hinders the broad application.

Therefore, to meet the requirements of high scalability and security as much as possible under the premise of decentralization performance, a new consensus algorithm for Permissioned blockchain put forward in this article is appropriable, called RAC (Risk Assessment-based Consensus protocol). The main contributions of our work are as follows.

(1) There is no centralized endorsement organization in the blockchain system. The participant with strong computing power or high-stake rights is often the accountant node that packs transactions into a block and sends the block to other participants, weakening the blockchain system's decentralized characteristics. We have designed a new decentralized consensus model to avoid monopolistic behavior caused by excessive concentration of authority. In our model, achieving consensus relies on cooperation between all roles. At the same time, different roles can achieve conversion under certain conditions

(2) It will immeasurably impact the blockchain system when the accountant node is maliciously controlled. We have designed an efficient accountant node election strategy to ensure that only honest participants can act as the accountant to generate new blocks, combined with the risk-node assessment mechanism to identify the malicious nodes that may exist in the network

(3) The distributed connectivity of the blockchain exposes the systems to Byzantine attacks. The compromised participants will further decrease the trust level among cooperative organizations by generating false data to obstruct consensus. We have designed an efficient block addition and transaction confirmation strategy to prevent possible Byzantine behavior and collusion attacks in the network. The process achieves a credible consensus by combining the reward and punishment mechanism and addressing the unbalance performance issues in speed, scalability, and security in the consensus algorithm proposed

The remainder of this article is organized as follows. Chapter 2 reviews the research related to the blockchain consensus algorithm. Chapter 3 introduces the basic definition and system model of the RAC algorithm. Chapter 4 describes the detailed implementation process of the RAC algorithm. Chapter 5 demonstrates the performance of the RAC algorithm through theoretical analysis and experiments. Chapter 6 summarizes the research work and provides an outlook for future work.

2. Related Work

There have been several works on blockchain and consensus algorithms in recent years, and some previous studies have motivated our work on the consensus model. In this section, we present a brief literature review from three aspects.

2.1. Overview of Blockchain. A blockchain is a tamper-evident ledger supported by a consensus algorithm. Peer to Peer (P2P) networks, cryptographic hash, digital signatures, and smart contracts are core blockchain technologies. Every node in the blockchain maintains a copy of the ledger verified by a consensus protocol, as shown in Figure 1. The ledger exists in blocks, each linked to the previous block by a hash [19] . Since each node in a P2P network has equal status, there is no need for a central server to exchange information in the blockchain system [20]

2.1.1. Blockchain Transaction Process. As shown in Figure 2, the transaction process of any blockchain system can be classified into three stages: accountant selection, block addition, and transaction confirmation [14].

Accountant selection is the first stage of the transaction process. Accountant nodes are responsible for collecting and generating blocks from the client then packaging transactions into blocks and sending blocks to other nodes in the blockchain network. Accountant nodes can be caused by

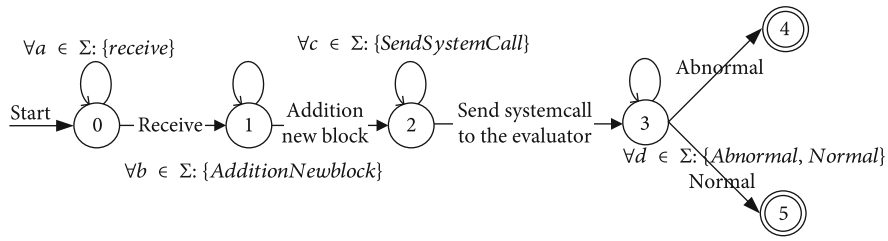


Figure 1: The structure of block in consensus algorithm.

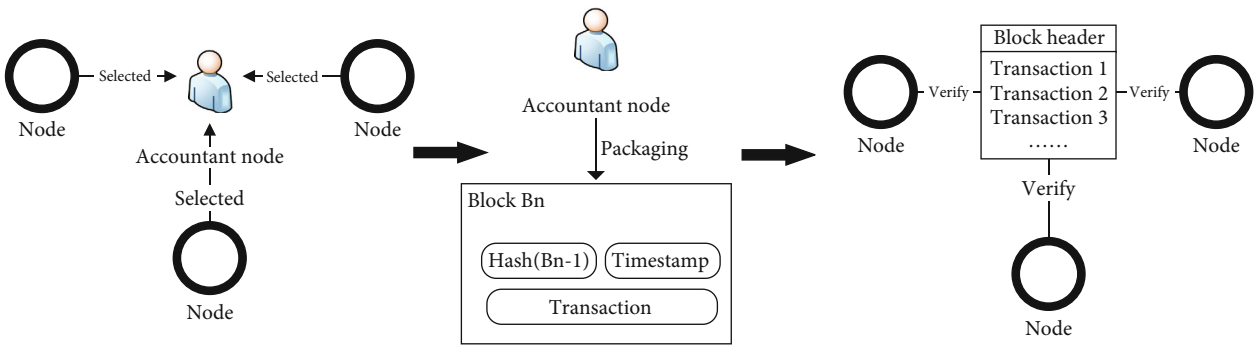


Figure 2: The transaction process of blockchain.

competing (e.g., PoW, PoS), polling (e.g., PBFT), or voting (e.g., Raft) approaches.

Block addition is the second stage of the transaction process. Since each node maintains a copied ledger locally, a node adds the block into the local ledger after the block verification is passed. In a few consensus algorithms, such as PBFT, block addition requires a majority vote of the nodes. Therefore, these consensus algorithms generally suffer from a lack of scalability and high consensus cost.

Transaction confirmation is the third stage of the transaction process. The purpose of transaction confirmation is to confirm the transaction's validity based on the blockchain held by each node in the blockchain network. However, most consensus algorithms do not require real-time voting by the nodes during the block addition phase. Hence, the confirmation efficiency relies on the consensus algorithm's characteristics used in the blockchain system. For example, confirming a transaction in the Raft algorithm requires that more than 50% of the nodes complete the block addition.

2.1.2. Types of Blockchains. Blockchain can be divided into permissionless and permissioned [21]. Bitcoin, Ethereum, and other cryptocurrencies issued to the public have been divided into permissionless blockchains. Any anonymous individual can freely join and maintain the blockchain network. There is no trust until the blockchain state reaches an immutable block depth. The measures of "mining" to provide financial incentives were used to compensate for the trust issues and offset Byzantine fault tolerance costs. So, the permissionless blockchain has weak concurrent transaction capability, high transaction fees, and a long confirmation time. In addition, participants in the permissionless blockchain are impossible to be identified.

However, some requirements must be considered in many cases, such as transaction throughput performance, transaction confirmation latency, the confidentiality of transactions, and participant identity [22]. These characteristics are the performance requirement of use cases and regulations that must be followed. Know-Your-Customer (KYC) is one of the essential principles in an actual financial transaction, which cannot be achieved in a permissionless blockchain.

Permissioned blockchains, on the other hand, provide a way to achieve the interactions between entities that do not establish an entire trust relationship. Since the participants' identities are known, permissioned blockchains can use a more efficient consensus algorithm to drive smart contracts, such as crash fault-tolerant (CFT) or Byzantine fault-tolerant (BFT) protocols. As a result, permissioned blockchain has lower transaction costs and higher efficiency than permissionless blockchain, which uses an "incentive mechanism" to reach a consensus.

Table 1compares permissionless blockchains and permissioned blockchains in terms of various parameters. To sum up, transactions are executed on every node in the permissionless blockchain network, which can be problematic for many use cases, such as supply chain, and the securities industry, because none of the partners would want their competitors to know confidentiality information in these businesses. On the contrary, permissioned blockchain systems have adopted various approaches to address the lack of privacy and further improve efficiency. These characteristics make permissioned blockchain acceptable in many enterprises' use cases.

2.1.3. The Challenge in Blockchain-Based Applications. Permissionless blockchains are more suitable for cryptocurrency applications in open and anonymous environments.

Table 1: Comparisons between permissionless blockchain and permissioned blockchain.

Sl. No.	Criteria	Permissionless	Permissioned
1	Decentralization	High	Low
2	Identity	Anonymous	Identifiable
3	Award	Mostly yes	Mostly no
4	User scale	Large	Small
5	Classic project	Bitcoin	Hyperledger

The number of users in such a case is enormous. On the other side of the coin, permissioned blockchains are more suitable for cooperation between a certain scale of organizations without a centralized authority party. All stakeholders of permissioned blockchain must be identified and identifiable.

Although blockchain has been applied in many aspects, including in the area of IoT, smart city, supply chain, vehicular ad-hoc networks, and so on, it is still a technology that is under development, which means there are some obstacles exist in blockchain-based applications that need to be improved [19]. From what has been analyzed in some previous work, the contradiction between decentralization, security, and scalability is the main reason that restricts the development of blockchain-based applications.

The three properties, consistency, availability, and partition tolerance, can only satisfy most two simultaneously in a distributed system, which was named the "CAP theorem" and proved in 2002 by Seth Gilbert and Nancy Lynch [23]. CAP provides a guiding principle for the design of consensus algorithms. From this, researchers no longer pursue that can satisfy all three properties simultaneously. Based on the CAP theorem, Vitalik Buterin proposed the DSS conjecture in the Ethereum system: the blockchain system cannot be enhanced in terms of decentralization, security, and scalability at the same time [24]. From the DSS conjecture, we can conclude that the transaction throughput of permissioned blockchains where total decentralization is not provided is much higher than permissionless blockchains. For example, a decentralized bitcoin system is much smaller than Hyperledger Fabric with partial decentralization in throughput performance.

Decentralization means that the accountant node is fairly generated throughout all nodes in the blockchain network rather than being centralized by a small number of users. The degree of decentralization and the control of the blockchain network are closely related. It is essential to ensure the fair participation of users.

Security means the main security issues and corresponding protection mechanisms in blockchain systems, including cryptographic security, network security, data security, and consensus algorithm security. These include Collusion, Double Spending, Sybil, and Targeted attack.

Scalability refers to the ability of the blockchain system to process the transaction. Three main aspects influence the performance of scalability. First, network latency of the distributed system; the network latency of a distributed system is more limited than that of a traditional distributed system due to an uncontrollable network environment. Second, the consistency of all users in blockchain requires necessary confirmation computation to maintain the consistency among nodes, resulting in additional computational cost. Third, constraints in computational performance of blockchain nodes, especially in IoT applications; many nodes cannot process some consensuses that needs a large amount of energy and computational consumption, such as PoW.

Although the DSS conjecture is only a way to analyze the blockchain performance of Ethereum, it is not theoretically rigorous. Still, it provides an entry point for researchers to improve the performance of the consensus algorithm. From what has been analyzed above, developing suitable consensus algorithms to balance the performance between decentralization, security, and scalability is the key to solving the challenge in blockchain-based applications [25].

2.2. Proof-Based Consensus Algorithm. PoW [1] algorithm is used in permissionless blockchain systems. Participants compete for accounting rights by solving a complex but easily verifiable mathematical problem with their computing power. The first node to solve such a problem is rewarded with a certain amount of cryptocurrency. This process is described as "mining," and the feature of open access makes PoW more scalable and decentralized. However, there is the problem of computing power consumption in PoW. It is unfair that the competition for accounting rights is closely related to the computing power of nodes because nodes with low CPU capacity are difficult to obtain accounting rights of PoW. In addition, a transaction in a block must wait for the confirmation of six additional blocks before consensus can be achieved to prevent Double Spending attacks in the consensus process. Studies have shown that the PoW algorithm can only reach consensus for seven transactions per second and may suffer tailored attack behavior such as 51% attacks. The low consensus efficiency limits the application of PoW in blockchain systems other than Bitcoin [26].

Researchers have proposed the PoS consensus to solve the problem of computing power consumption in the PoW algorithm, which has been applied in Ethereum [11]. In the PoS algorithm, each node is given a new metric called coinage. PoS algorithm assumes that rational high-asset nodes will not disrupt the consensus process because the potential loss of assets will outweigh the gain from their malicious deeds. The PoS algorithm achieves higher TPS performance compared to the PoW algorithm, but there are also shortcomings. The risk of monopoly is introduced in PoS. Low-asset nodes cannot compete fairly with high-asset nodes for accounting rights, and it can lead to a tendency to centralize the system. Researchers have proposed

the DPoS algorithm to mitigate this monopoly risk in PoS, which is currently used in projects such as EOSIO and Cosmos [12]. Each node can vote for a representative based on their stake, and the accounting rights will be given to the node that receives the highest number of votes. Although the PoS and DPoS solve the waste of resources in PoW, there are still problems such as low efficiency.

The proof-based consensus algorithm is designed for cryptocurrency systems like Bitcoin and Ethereum. Although the efficiency of the proof-based consensus algorithm is low, it is worth mentioning that these algorithms make some attack behavior for blockchain systems impractical. We provided a summary comparison in Table 2 about its pros and cons.

2.3. Voting-Based Consensus Algorithm. Permissioned blockchain has higher requirements for transaction throughput than permissionless blockchain, and it is more suitable to adopt the voting-based consensus mechanism. This section will introduce some voting-based consensus mechanisms and their improvement algorithms. The voting-based consensus mechanism can be divided into Byzantine Fault Tolerance algorithms that can tolerate Byzantine Failure and Crash Fault Tolerance algorithms that can only handle Fail-stop Failure. Byzantine General Problem proposed by Lamport et al. in 1982 is how the participants can complete consensus in the presence of malicious node interference in the network [27].

The differences between the two consensus algorithms are shown in Table 3. It should be noted that all the data of TPS in the table are the result under ideal experimental conditions.

2.3.1. Byzantine Fault Tolerance Algorithms. As a classical BFT algorithm based on the state machine replication mechanism, the PBFT algorithm was proposed by M. Castro and B. Liskov et al. [27]. PBFT algorithm can reach consensus with the number of malicious nodes accounting for less than 1/3. However, the PBFT algorithm requires mutual communication and confirmation of every two nodes in each consensus, and the algorithm's complexity is $O(n^2)$. Thus, the application scenario of PBFT is minimal due to its high consensus cost and poor scalability performance. PBFT has not been used in real-world engineering applications. For example, the famous open-source project of permissioned blockchain, Hyperledger Fabric, has only used the PBFT algorithm in the early v0.6 version.

Lei et al. proposed a reputation-based Byzantine fault-tolerant algorithm, RBFT, in 2018 [28]. RBFT evaluates the behavior of each node in the consensus process through a reputation model. If a malicious node is detected, its reputation value and voting weight will be reduced. In addition, the view change phase is improved by adding an incentive mechanism so that the nodes with higher reputation value have a higher chance of becoming the accountant node. The experimental results show that the RBFT algorithm can quickly and effectively identify malicious nodes through the reputation model. Compared with PBFT, the RBFT algorithm can eventually improve the transmission throughput by 15% and reduce the latency metric by 10%.

Rong et al. proposed an improved BFT algorithm, ERBFT, in 2019 [29]. ERBFT algorithm uses the backup nodes in the network to identify whether the accountant node is a malicious node through a new request ordering mechanism (Order-Match). Suppose the accountant node does not pass the verification of the Order-Match mechanism. In that case, the backup node will trigger the suspicious protocol (suspect protocol) to confirm further whether the accountant node is under malicious attack. The experimental results show that the ERBFT algorithm performs better than the PBFT algorithm in throughput and scalability and can improve the transaction throughput by 30%.

Tendermint is a blockchain project of the Cosmos network, known for simplicity, high performance, and fork accountability. As the consensus protocol of the Tendermint, Tendermint BFT is a simplified version of the PBFT algorithm. The relationship between Tendermint BFT and PBFT is similar to the relationship between Raft and Paxos. Tendermint BFT can be used in a Byzantine environment. Although Tendermint BFT can tolerate the failure of only 1/3 of the nodes in the system, it can tolerate any fault, including hacking and malicious attacks [30].

The core idea of the above study can be summarized as optimizing the traditional Byzantine fault-tolerant algorithm by improving the transaction throughput and scaling performance. Although the current research results have improved the performance and scalability of the traditional Byzantine fault-tolerant algorithm, the transaction efficiency is still far from that of the Crash Fault Tolerance algorithm.

2.3.2. Crash Fault Tolerance Algorithms. The Raft algorithm [16] is an improvement of the Paxos algorithm; the Raft algorithm completes the consensus through the leader election phase and logs replication phase. The nodes of the Raft cluster do not need to confirm each other for the transmitted data. The transmission throughput of the Raft algorithm in the experimental environment is more than five times that of the PBFT algorithm. The Raft algorithm has been used in many practical projects such as ETCD and BRAFT due to their easy-to-understand and excellent performance. However, the Raft algorithm can only be used in a non-Byzantine network environment without malicious nodes, which cannot restrict the behavior of malicious nodes, so it will bring incalculable impact to the whole consensus system once the leader node is maliciously controlled [18].

Chen et al. proposed a CRaft consensus algorithm based on the node trust mechanism based on the Raft algorithm in 2018 [31], which enables the algorithm to be used in the Byzantine network environment. CRaft is divided into the trust evaluation phase and consensus phase. It establishes trust evaluation criteria by the OC-SVM algorithm. The prediction accuracy of Craft for Byzantine nodes is up to 100% (there is still a 17.89% false-positive rate in CRaft). Compared with the PBFT algorithm, the throughput of CRaft can still be maintained at good performance when the number of nodes is expanded to 60. So, it is more suitable for a permissioned blockchain environment than PBFT.

Table 2: Comparisons between classic proof-based consensus algorithm.

Sl. No.	Consensus algorithms	Classic application	Decentralization level	Energy efficiency	Proof based on
1	PoW	Bitcoin	Decentralized	No	Work
2	PoS	Ethereum	Semi-centralized	Yes	Stake
3	DPoS	Ethereum	Semi-centralized	Yes	Vote

Table 3: Comparisons between two types of consensus algorithm in permissioned blockchain.

Sl. No.	Criteria	CFT algorithm	BFT algorithm
1	Crash fault tolerance	50%	33%
2	Byzantine fault tolerance	N/A	50%
3	Throughput (TPS)	>10 k	>1 k
4	Classic algorithms	RAFT	PBFT

Wang et al. developed the Beh-Raft algorithm in 2021, which divides blockchain nodes into groups as parallel shadings to improve scalability at the cost of increased communication and storage per node [32]. Beh-Raft introduces the Proof of Behavior (PoB) algorithm for incentivizing honest behavior, which ensures that the probability of an honest node being selected as an accountant node is greatly increased by combining the PoB with the Raft algorithm. The Beh-Raft algorithm is Byzantine fault-tolerant while maintaining better scalability.

The hhraft algorithm was proposed by Wang et al. in 2021 [33]. In response to the problem that the RAFT algorithm cannot use in a real-world network, hhraft introduces a new monitor mechanism to optimize the Raft consensus process, which is used to monitor the network for "Sybil nodes" that fake their identities maliciously and accountant nodes that tamper with the original data. The hhraft has been experimentally proven to outperform the Raft algorithm in terms of transaction throughput, consensus latency, and resistance to Byzantine failures. It is suitable for real network environments with high real-time and high adversarial performance.

Although the current research results can determine the malicious nodes in the network, there are still some shortcomings. Most of the present improved algorithms select malicious nodes in a "static" method, meaning they need to design adversarial node models or reputation value determination criteria in advance. They cannot be adjusted according to dynamically to network environment changes. However, the imbalance between normal and malicious nodes' behavior makes it challenging to construct an accurate classifier in a Byzantine network environment.

3. The System Model

3.1. Problem Description. In order to realize some business goals without centralized institutions and meet the demands of traceability and verifiability, such as data sharing or commercial paper exchange, a permissioned blockchain is usually established by several companies or organizations without a complete trust relationship based between them. In this blockchain network, everyone wants administrative privileges because everyone fears losing rights and profits in decentralized networks. It is an acceptable solution for all participants to monitor each other to prevent the organization that masters the power to dominate the system from gaining illegal benefits. In such a case, the participants do not subjectively destroy to cause Byzantine behavior to the permission blockchain unless they are maliciously controlled because everyone wants to benefit from the blockchain system. In addition, the participants are abstracted as nodes. Nodes are computational devices such as sensors, computers, and servers in the real environment. The lowest possible consumption of computational resources is what every stakeholder of the permission blockchain expects to achieve due to computational resources and economic costs being closely linked. Therefore, it is essential to find a suitable consensus algorithm to meet the demand of such a consortium blockchain.

From what has been analyzed in Section 2.1, the evaluation system of the consensus algorithm can be established based on decentralization, security, and scalability. More detailed, each evaluation criteria can be measured by several sub-indicators [25], so the main parameters of the evaluation consensus algorithm system can be summarized in Table 4.

Number of consensus nodes: In some consensus algorithms, not all nodes can participate in the election of accountant nodes because some consensus nodes need to undertake other tasks such as supervising the consensus process's realization. On the contrary, all nodes can be selected as accountant nodes due to no additional roles in the consensus algorithm. Number of consensus nodes is the Qualitative and Beneficial Indicator; we set $\{0, 1\}$ to indicate the $\{\text{Part, All}\}$, respectively.

Accountant selection method: In general, voting and polling are two of the most widely used in permissioned blockchain. Compared to polling, the voting is somewhat less decentralized because nodes may not have an equal probability of getting a vote under different network environments. Accountant selection method is the Qualitative and Beneficial Indicator; we set $\{0, 0.5, 1\}$ to indicate the $\{$ Competing, Voting, Polling$\}$, respectively.

Consensus nodes weight: The probability of a consensus node that is honest becoming an accountant node depends on the presence of constraints. For example, the credit evaluation standard that includes the node performance is set in CRAFT. Only the node that meets this credit evaluation standard can be selected. However, some consensus algorithm does not have constraints, which means that honest nodes have the same probability of being the accountant node. Consensus nodes weight is the Qualitative and

Table 4: The evaluation system of consensus algorithm.

Sl. No.	Criteria	Sub-indicators	Description
1		Number of consensus nodes	The number of nodes in the blockchain network that can become the accountant node.
2	Decentralization	Accountant selection method	The method that accountant node is generated in the blockchain network, competition, election, or polling.
3		Consensus nodes weight	Whether the consensus nodes have equal probability of becoming the accountant node in each consensus process.
4		Byzantine fault tolerance	The maximum percentage of Byzantine malicious nodes in the whole blockchain network that can be accepted.
5	Security	Byzantine node controllability	The ability of consensus algorithm to exclude malicious nodes from the consensus process
6		Attack behavior costs	The cost of an attack behavior by a malicious attacker in the blockchain network
7	Scalability	Resource consumption	Resources to be consumed in the consensus process include computation, storage, and network communication, which can be measured by the communication complexity of consensus.

Beneficial Indicator; we set $\{0, 1\}$ to indicate the {Not equal weighting of consensus nodes, Equal weighting of consensus nodes}, respectively.

Byzantine fault tolerance: The authors summarize this metric in their respective papers. Byzantine fault tolerance is the Qualitative and Beneficial Indicator. We assign a value to Byzantine fault tolerance with reference in Table 5.

Byzantine node controllability: In some consensus algorithms, identified malicious nodes can continue to compete for the accountant node. Reducing the impact of malicious nodes is one of the essential metrics for measuring security. In RAC, reducing the weight of malicious nodes can also prevent them from becoming accountant nodes because they cannot get more than half of the votes under the Risk-Node Assessment Mechanism. Byzantine node controllability is the Qualitative and Beneficial Indicator; we set $\{0, 1\}$ to indicate the {No controllability over Byzantine nodes, Having control over Byzantine nodes}, respectively.

Attack behavior costs: This metric indicates the cost consumed by attackers needed to malicious control the whole permissioned blockchain. Attack behavior costs include the following two areas: malicious control of the accountant node or enabling permissioned blockchain to reach consensus under the attackers' intentions. Attack behavior costs are the Qualitative and Beneficial Indicator. We assign a value to attack behavior costs according to Table 6.

Resource consumption: The communication complexity of consensus can measure this metric. The authors summarize the communication complexity of proposed consensus algorithms in their respective papers. Resource consumption is the Qualitative and Cost Indicator. We assign a value to resource consumption according to Table 7.

We can describe the problem that development of appropriate consensus algorithm based on some indicators as a mathematical model. Let eigenvector $S = \{S_1, S_2, \cdots, S_n\}$ indicate to the consensus algorithms that need to be compared. Assume that there $m(m = 7 \text{ in the case analyzed before})$ indicators in evaluation system, then $a_{ij}\,(i = 1, 2, \cdots, n\,; j = 1, 2, \cdots, m)$ is denoted as the value of corresponding indicator observation of decentralization, security, and scalability. The evaluation indicator is preprocessed by normalization and dimensionless method, and the evaluation matrix $\boldsymbol{B} = (b_{ij})_{n\times m}$ is constructed.

Positive ideal solution $C^+ = [c_1^+, c_2^+, \cdots, c_m^+]$ denotes an ideal consensus algorithm that not existed in real world (each indicator of C^+ is the best value among all the consensus algorithms). Set the value of the jth indicator in C^+ is c_j^+,

$$c_j^+ = \max\left(b_{ij}\right), 1 \le i \le n \text{ and } 1 \le j \le m. \tag{1}$$

On the contrary, negative ideal solution $C^- = [c_1^-, c_2^-, \cdots, c_m^-]$ denotes another ideal consensus algorithm (each indicator of C^- is the worst value among all the consensus algorithms). Set the value of the jth indicator in C^- is c_j^-,

$$c_j^- = \min\left(b_{ij}\right), 1 \le i \le n \text{ and } 1 \le j \le m. \tag{2}$$

Our objective is to find the consensus algorithm that has maximum proximity between ideal solution. Therefore, the objective problem is defined as

$$\max\left\{f_i = \frac{s_i^-}{s_i^- + s_i^+}\right\}, 1 \le i \le n, \tag{3}$$

where s_i^+ and s_i^- are calculated as follows

$$\begin{aligned} s_i^+ &= \sqrt{\sum_{j=1}^{m}\left(b_{ij} - c_j^+\right)^2}, 1 \le i \le n, \\ s_i^- &= \sqrt{\sum_{j=1}^{m}\left(b_{ij} - c_j^-\right)^2}, 1 \le i \le n. \end{aligned} \tag{4}$$

Table 5: Scoring basis for Byzantine fault tolerance.

Value	Range of Byzantine fault tolerance	Description
0	0%	Consensus algorithm that completes failure to recognize Byzantine behavior in the network.
0.3	1%-16%	Consensus algorithm that has ability to tolerate a small number of Byzantine nodes in the network.
0.5	17%-33%	Consensus algorithm that has ability to tolerate several numbers of Byzantine nodes in the network.
0.7	34%-51%	Consensus algorithm that has ability to tolerate no more than half of Byzantine nodes in the network.
1	>51%	Consensus algorithm that can withstand 51% of attacks.

Table 6: Scoring basis for attack behavior costs.

Value	Attack behavior costs	Description
0	None	An attacker can achieve malicious control of the blockchain system at no cost. Such as solo consensus of Hyperledger Fabric.
0.3	Low	The attacker only needs to spend a small amount of cost to control critical parts such as accountant or node with evaluation function to achieve malicious control of the blockchain system. Such as Raft.
0.6	Middle	The attacker only needs to spend many costs to achieve malicious control of the blockchain system because a certain malicious attack detection and prevention mechanism is set in the consensus algorithm.
1	High	It is almost impossible for an attacker to achieve complete control of the blockchain system due to the very strict confirmation mechanism, unless maliciously controlling more than half of the nodes, such as PoW and PBFT.

Table 7: Scoring basis for resource consumption.

Value	Resource consumption	Description
0	$> O(n^2)$	Extremely resource-intensive consensus algorithms.
0.3	$O(n^2)$	Consensus algorithms that require complex confirmation mechanisms, such as PBFT.
0.5	$O(n \log n)$	Consensus algorithms that require some additional confirmation mechanism, such as Beh-raft.
0.7	$O(n)$	Consensus algorithm with efficient confirmation mechanism, such as raft.
1	$O(1)$	Ideal-state consensus algorithm, typically used in experimental settings, such as solo consensus of Hyperledger Fabric.

3.2. Basic Definitions. The participants of permissioned blockchain are abstracted as nodes, either from the same organization or from different organizations, and will carry a certain amount of assets or stake interests. First, we need to introduce some basic definitions to help the reader better understand the proposed consensus algorithm.

(1) Byzantine node and honest node: Byzantine nodes are defined as nodes that malicious attackers may compromise to hinder consensus in participants. Byzantine nodes are also referred to as risk or malicious nodes in this paper. On the contrary, the honest node indicates the node making the right decision on the consensus. The definition of a right decision is determined by the behavior of nodes under different states. The introduction of the states and behaviors will be given in Sections 3.3 and 4.3

(2) RNL: The node in the risk node list (RNL) means that the node is not trusted anymore because the Byzantine General Problem or other malicious behavior has occurred in these nodes. RNL is continuously updated during the consensus process according to the behavior of every node in the network

(3) Term: RAC uses the term as the logical timestamp to ensure that the nodes have correct timestamps under their different physical environments. The term is numbered using consecutive integers, and the term

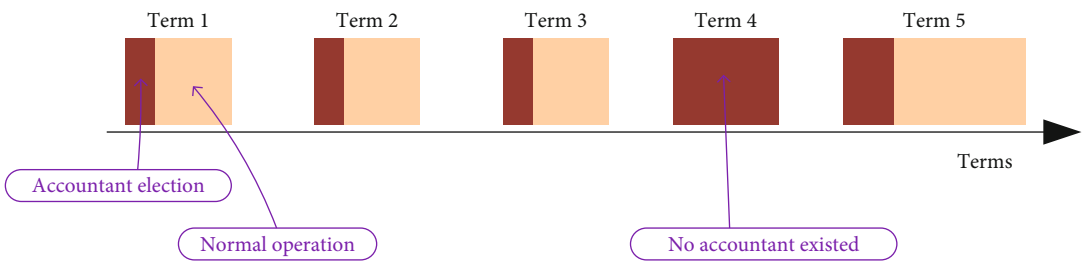


Figure 3: After a successful election, an accountant manages the cluster until the end of the term.

is updated to a larger term number when a new accountant is generated. The description of the term as shown in Figure 3

(4) Basic operations: This definition is mainly applied in the description of the RAC process in Section 4.1, which is divided into the sub-definition as follows

(i) $P_i = \{P_{i1}, P_{i2}, P_{i3}, \cdots, P_{\text{in}}\}$. The set of nodes in the participating organization i, where n denotes the total number of nodes in organization i

(ii) $E = \{E_1, E_2, E_3, \cdots, E_n\}$. The set of evaluator nodes in the permissioned blockchain network, i.e., the group of evaluator nodes

(iii) $O = \{O_1, O_2, O_3, \cdots, O_n\}$. The set of nodes that is participating of the RAC consensus in the permissioned blockchain network, that is, the consensus node group

(iv) Send(Message, A, B). Send the message quest Message from node A to node B

(v) Pick(O, Accountant). An accountant node Accountant is elected from the consensus node group O

(vi) Verify(RPC, A, B). The interaction between node A and node B through an interaction function RPC to complete the verification of a block, where node A is the client of RPC and node B is the server of RPC

(vii) Package(Message, new_Block, Accountant). The accountant node Accountant packages the message from client of permissioned blockchain Message into a new block new_Block that can be broadcasted and stored in the blockchain

(viii) Re_Blockchain(Accountant, RPC, new_Block). Block update function. The accountant node Accountant broadcasts the new block new_Block to all participating organizations in the network through the interactive function RPC.

3.3. The States in Consensus Algorithm. Permissioned blockchain is established by a certain number of companies or organizations. It is an acceptable solution that all participators monitor each other to prevent the organization from gaining illegal benefits. When joining the network, each node in the permissioned blockchain is verified and registered through the membership service provider and is assigned a unique ID and key pair to indicate its identity. All nodes may experience four different roles in RAC: follower, candidate, accountant, and evaluator.

(1) Follower: All nodes are labeled as followers after joining the permissioned blockchain. They can only passively receive the message about blocks that need to be added to the blockchain. When receiving requests from clients, followers need to forward these requests to the accountant for packaging because they do not have the authority to package blocks

(2) Candidate: The candidate is the intermediate state between the follower and accountant and will not exist in the network long. Any follower node can become a candidate node when it finds that the accountant node cannot serve properly. The candidate node can initiate votes to other follower nodes and need to get the voting support of enough follower nodes before it can be converted into an accountant node

(3) Accountant: Accountant nodes are elected from candidate nodes. Packing client requests into a block and sending them to the evaluator nodes for validation is the responsibility of the accountant node

(4) Evaluator: Evaluator node has two tasks in the RAC. First, based on the performance of follower nodes, the evaluator node can determine whether a node to be maliciously controlled through the risk-node assessment mechanism. Second, monitor the performance of accountant nodes and terminate the term of accountant node with abnormal behavior in time. They are high-asset nodes of each participant in the permissioned blockchain, which can primarily reduce the possibility of them actively causing damage to the blockchain system

The state of a node is temporary, and no one can work in a specific role permanently. Different roles can be converted

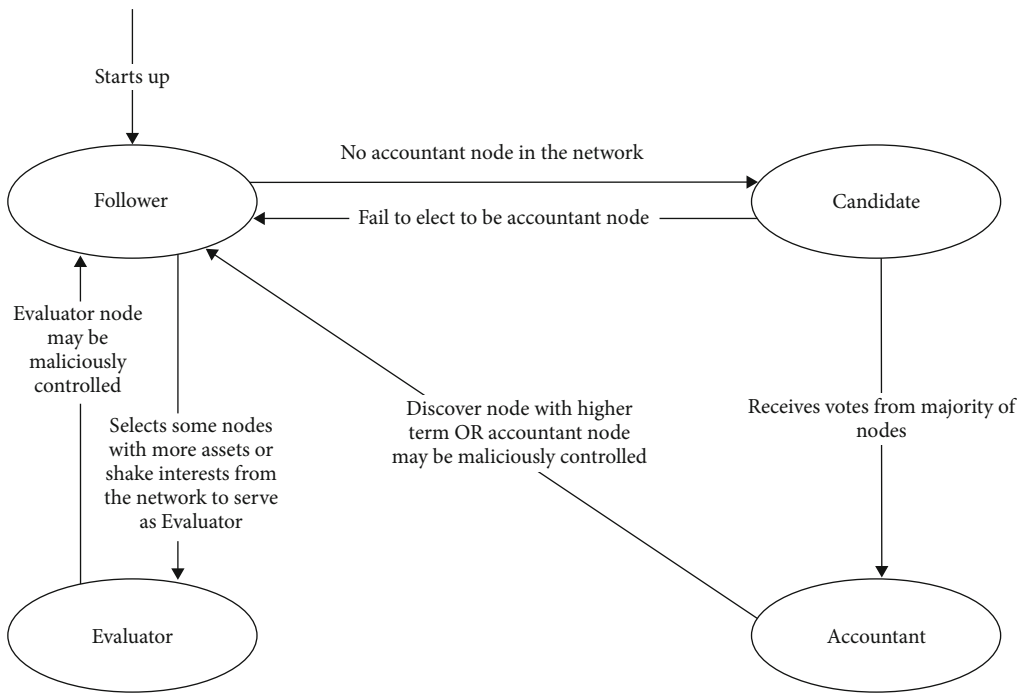


Figure 4: The conversion relationship in RAC.

under certain conditions, and the conversion relationship is shown in Figure 4.

4. Algorithm Design

4.1. The Description of Details in RAC Algorithm. The RAC is divided into six phases: Generation of Evaluator, Update of the RNL, Generation of Candidate and Election of Accountant, New Blocks addition, Judgment of New Blocks, New Blocks confirmation. Nodes use Remote Procedure Call (RPC) to communicate in the network. The consensus process of RAC is shown in Figure 5.

Each organization selects a certain number of nodes with more assets or computational power as evaluator nodes to represent the interests of the organization, and together they form the evaluator node group. To prevent possible collusion attacks in the subsequent consensus process, the evaluator node group needs to be composed from different organizations. $E = \{E_1, E_2, E_3, \cdots, E_n\}$. Initialize the evaluator nodes of each participating organization into evaluator node groups.

① Send(SystemCall, O, E).

All nodes send the sequence of system calls in the last term to the evaluator after the evaluator group is created to decide whether there are Byzantine nodes in the consensus node group that are not suitable for having accounting rights.

② Send(RiskComputeRPC, O, E)

The evaluator node calculates the potential risk nodes in the network through the Risk-Node Assessment Mechanism and generates and updates a list of risk nodes (RNL) to broadcast to all nodes in the network. RNL updates are implemented through RiskCompute RPC between the node and the evaluator. The description of RiskCompute RPC is shown in Table 8.

③ Pick(O, Accountant)

A follower node that is not selected as evaluator does not receive a heartbeat message from the accountant node in a period (randomly selected in the interval of 150-300 milliseconds), and it indicates that the accountant node of the current term can no longer provide adequate services, which means the accountant node may be offline due to network delay or system failure. At which time, the node that first discovers this situation becomes a candidate node and initiates an election for a new term accountant and sends a RequestVote RPC to all nodes in the network except itself. The nodes that receive a RequestVote RPC from the candidate node only respond to the vote request usually when the candidate node is not matched with RNL. The description of RequestVote RPC is shown in Table 9. When a candidate node gets half or more voting responses, the node changes from candidate state to accountant state and sends a heartbeat message to other nodes in the network. The purpose of this action is to prove that there is already an accountant node that can provide adequate services.

④ Package(Message, new_Block, Accountant)

The accountant node packages the requests from the client into a block with the structure shown in Figure 1. A complete block message can be represented as < BlockNum, prehash, TimeStamp, Merkle-root, Entries[ ]>, together with the previous hash value, the timestamp, and other necessary messages, where Entries[ ] including the client transaction request. The block is then broadcast to the evaluator group to validate the block information. It is important to note that the original transaction request from the client is sent to the accountant node as well as to evaluator group at the same time.

⑤ Verify(JudmentRPC, P_i, E)

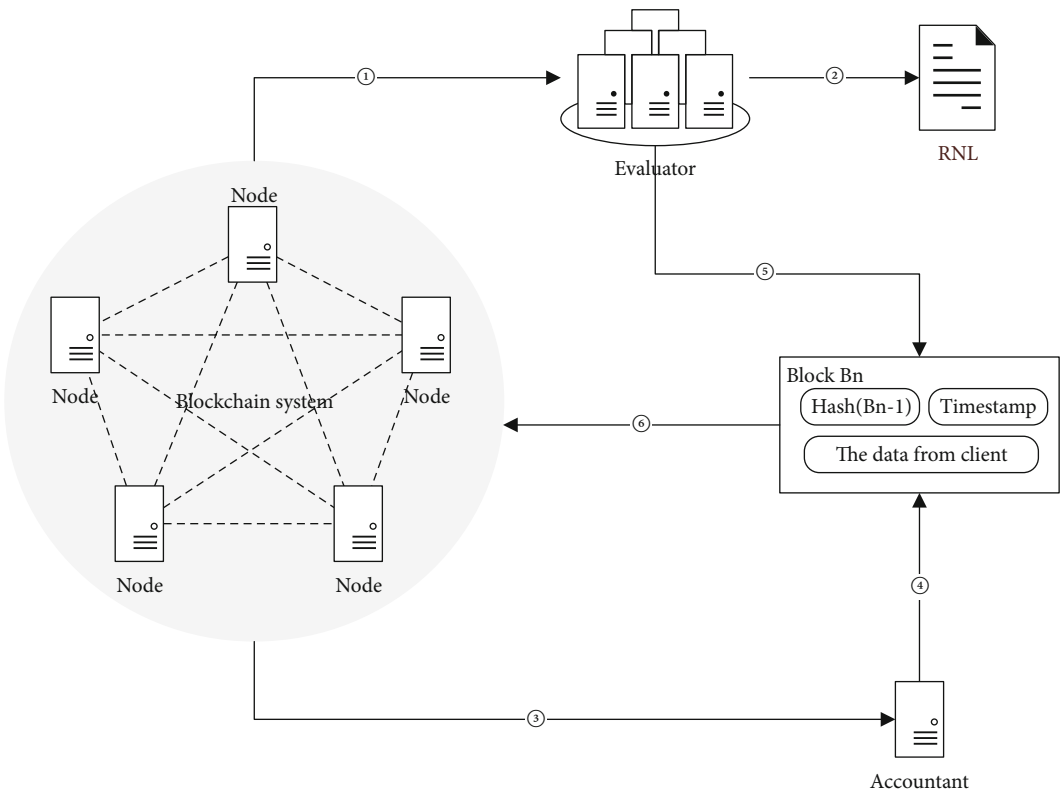


Figure 5: The consensus process of RAC.

Table 8: The description of RiskCompute RPC in the RAC algorithm.

RiskCompute RPC Parameter	Description
Term	The current term of the node.
NodeID	The id of the node.
SystemCall	The systemcall of the node in last term.
Return	**Description**
Term	The current term of the node.
RNL	The risk node list in current term.

Table 9: The description of RequestVote RPC in the RAC algorithm.

RequestVote RPC Parameter	Description
Term	The current term of the candidate node.
CandidateID	The id of the candidate node.
Return	**Description**
Term	The current term of the node.
VoteGranted	Set to true when the candidate won this vote.

The evaluator node verifies the block information from the accountant node. The purpose of this step is to determine whether a Byzantine event has occurred. The evaluator node first checks the block legitimacy information such as the hash value in the block. Secondly, it matches the Entries[ ] in the block to determine whether it is consistent with the transaction request from the client. Phase ⑤ is implemented through Judgment RPC between the node and the evaluator. The description of Judgment RPC is shown in Table 10.

⑥ Re_Blockchain(Accountant, AppendEntriesRPC, new _Block)

If the block is determined as invalid in phase ⑤, then the block information will be set to empty (more than 50% of the evaluator nodes respond to "Fail" to the accountant node in the Judgment RPC). The block information will be broadcast to all nodes after verification by the evaluator node. This process has interacted through AppendEntries RPC (the description of AppendEntries RPC is shown in Table 11). All nodes that receive the AppendEntries RPC will respond "success" to the accountant node, indicating that the transaction request has been correctly added. When the block has been securely copied to more than 50% of the nodes, the account node will return the execution result to the client, thus completing the consensus of the request. The follower node will decide that the current accountant node is a Byzantine node when they find an empty block added to the blockchain. Then, the node will add the current leader node into their RNL and restart a new term of accountant node election.

We now give a full version of RAC to help better understand their decentralized design concepts, and the pseudocodes are presented in Algorithm 1.

4.2. Risk-Node Assessment Mechanism. It is necessary to evaluate and update the Byzantine nodes in the network to prevent Byzantine nodes from becoming accountant nodes.

Table 10: The description of Judgment RPC in the RAC algorithm.

Judgment RPC Parameter	Description
Term	The current term of the accountant node.
Accountant ID	The id of the accountant node
Entries[ ]	The transaction request that added by accountant node in the block.
Return	Description
Term	The current term of the node.
Success	Set to true when the transaction information from the accountant and that from client are equal.
Fail	Set to true when the transaction information from the accountant and that from client are not equal.

Table 11: The description of AppendEntries RPC in the RAC algorithm.

Judgment RPC Parameter	Description
Term	The current term of the accountant node.
Accountant ID	The id of the accountant node
Entries[ ]	The transaction information that added by accountant node in the block.
Return	Description
Term	The current term of the node.
Success	Set to true when the transaction requests are successfully added to the local blockchain.

```
Input: Transaction request from client.
Output: void
begin
  if no evaluator group in the permissioned blockchain system then:
    generate evaluator group
  end if
  while Transaction request from client && evaluator group is existed do:
    if accountant node is not existed || the term of account node is less than current term || accountant node is Byzantine node
then:
      generate candidate
      communication between all follower nodes and evaluator group through RiskCompute RPC
      all node update RNL
        communication between candidate node and all follower nodes through RequestVote RPC
        if received vote from most follower nodes then:
          candidate node become accountant node
    else if accountant node existed then:
        add the transaction request into block
        communication between accountant node and evaluator group through Judgment RPC
        if the number of "fail" in all Judgment RPC>50% then:
          the block is set to empty block
        else then:
          the block is set to valid block
    communication between all follower nodes and accountant node through AppendEntries RPC
    if follower node i find empty block then:
          add accountant node into RNL
          the term of follower node i is increase
  end while
end
```

Algorithm 1: RAC

RAC describes the trustworthiness of a node by introducing the concepts of risk value and reliability. The risk value reflects the probability of Byzantine events occurring in a node during a specific term compared to other nodes in the network. Reliability reflects the probability of a Byzantine event occurring at the accountant node during their term. To improve the identification of malicious behavior, the Risk-Node Assessment Mechanism designed in this paper uses systemcall sequences as the source of data analysis. The system call represents the original interaction between the program and the host system, and there is no data abstraction in this process; the use of systemcall for intrusion detection system was first proposed by Forrest and assumes that program operation affects the system and that all exceptions leave traces in the system calls executed by the kernel [34] . Thus, the most significant advantage of our method is that it can find the occurrence of malicious behavior intuitively and locate malicious nodes with abnormal systemcall sequences quickly.

We offer some assumptions about the permissioned blockchain in order to set the stage for the discussion of the Risk-Node Assessment Mechanism.

Assumption 1. The number of honest nodes is always greater than that of Byzantine nodes in the network.

Assumption 2. The malicious behavior of Byzantine nodes cannot be predicted because attackers can use a variety of attack methods or tools to reach their goal of interfering with consensus individually or collusively.

Assumption 3. All nodes run on the same operating system, and the behavior of honest nodes is similar universally because they try to reach the consensus according to their responsibilities, which means that their systemcall sequence is also similar.

Based on the above assumptions, this section will use a short sequence-based technique to model the subsequence of system call sequences for honest nodes, and the node that deviates significantly from it will be considered malicious. The sequence of system calls for each node is first modeled. The sequence of system calls for all nodes is represented as a num $*$ k matrix, say N_r, where num indicates the number of nodes in the network and each row in N_r represents the entire sequence of system calls for a node in the previous term. Thus, the matrix N_r is the matrix consisting of the short sequence of system calls of all nodes in the network. Combining the sequence of system calls with their frequency of occurrence is also necessary to highlight the significant differences in behavior between malicious nodes and honest nodes. We set $S_{(i,j)}$ to indicate the specific index of a sequence of system calls, $S_{(i,j)}$ is derived from Equation (5), and f_s denotes the frequency of occurrence of $S_{(i,j)}$ in the matrix N_r.

$$S_{(i,j)} = T(i,j), 1 \le i \le \text{num and } 1 \le j \le k, \tag{5}$$

$$f_s = \frac{S_{(i,j)}}{\sum_m S_{(m,j)}}, 1 \le m \le \text{num and } 1 \le j \le k, \tag{6}$$

$$N_f(i,j) = S_{(i,j)} * f_s, 1 \le i \le \text{num and } 1 \le j \le k. \tag{7}$$

Second, based on Assumption 3, normal nodes behave very similarly in the consensus process. To further reflect the deviation between the short sequence matrix composed of frequently used system call sequences and the short sequence matrix composed of less regularly used system call sequences, it is also necessary to pay attention to some system call sequences with low frequency but tremendous value because the critical information such as malicious attack is likely to be hidden in them. Therefore, the process of inverse document frequency is also required, and the detail is shown in Equation (8).

$$idf_{(i,j)} = \log \frac{|\text{num}|}{\left|\left\{i : S_{(i,j)} \in N_i\right\} + 1\right|}, 1 \le i \le \text{num and } 1 \le j \le k. \tag{8}$$

In Equation (8), $|\text{num}|$ denotes the number of nodes; $|\{i : S_{(i,j)} \in N_i\} + 1|$ is the number of nodes containing the specific system call sequence $S_{(i,j)}$. Finally, the original matrix N_r is obtained as shown in Equation (9) after the processing of frequency and inverse document rate to obtain the matrix N_f.

$$N_f(i,j) = S_{(i,j)} * f_s(i,j) * idf_{(i,j)}, 1 \le i \le \text{num and } 1 \le j \le k. \tag{9}$$

Assumption 1 and Assumption 2 can analyze the risk value using an unsupervised outlier detection algorithm. In this paper, we use the isolated forest algorithm [35] for malicious node determination. The purpose of the isolation forest algorithm is to rank each node by its anomaly score. The isolation forest consists of many binary trees, each of which is called an isolation tree. The construction of an isolation tree is an entirely random process. Let there be num nodes in the network; the method of estimating the average path length of the leaf nodes of an isolation tree can be referred to as the unsuccessful search of a binary tree because they have a very similar structure, as shown in Equation (10).

$$c(n) = 2H(n-1) - 2\left(\frac{n-1}{n}\right), \tag{10}$$

where $H(x)$ is the Euler–Mascheroni constant, $H(x) = \ln(x) + 0.5772156649$). It can be used to normalize the path length because $c(n)$ is the average of the path length. The anomaly score of a node x can be calculated by Equation (11), where $h(x)$ is the path length of node x in the isolated tree.

$$s(x,n) = 2^{-E(h(x))/c(n)}. \tag{11}$$

Table 12: Comparisons between two types of consensus algorithm in blockchain.

Symbol	Description
Action	The set of operations performed by the follower, accountant, evaluator in different phase of the RAC. Action = {receive, generate new block, broadcast, verify, send system call ⋯ }.
Trace	A sequence on the set action. Such as {receive ⟶ generateNewBlock ⟶ broadcast ⟶ valid block}.
Behavior	The behavior of node 1 can be denoted as $<1, tr>$, the identification of the node is 1, and tr is a trace.

The risk-node assessment mechanism can calculate the risk values of each node by analyzing the sequence of system calls of the nodes in the previous term. The node will be judged as a Byzantine node when its risk value is significantly greater than other honest nodes.

4.3. Punishments Mechanism. This section will focus on the penalty mechanisms in the consensus algorithm. The symbol glossary is listed in Table 12 to facilitate expressing the penalty mechanisms. We use DFA (Deterministic Finite Automaton) to describe the Byzantine node. A DFA is a quintuple $<S, \Sigma, \delta, S_0, F>$, where S is a finite set of states, S_0 is the initial state, F is a set of acceptable states, Σ is a finite set of alphabets, and δ is conversion function; δ can be expressed as $\delta = s \times \Sigma \longrightarrow S$.

As shown in Figure 6(a), we define the Distinguishing Automaton for the behavior of an accountant node, in which the initial state is 0 ($S = 0$), acceptable states are [4, 5] ($F \in \{4, 5\}$), and Σ is the action, where "receive" indicates the receive the transaction request from the client, "generate new block" is the action of the convert transaction request to new block, "broadcast" means broadcast the new block to the evaluator group according to the description of phase ③, "valid block" and "empty block" represent the result of the block generated after verification of evaluator group ($\Sigma \in$ {receive, generate new block, broadcast, valid block, empty block}).

As shown in Figure 6(b), we define the Distinguishing Automaton for the behavior of an evaluator node, in which the initial state is 0 ($S = 0$), acceptable states are [3, 4] ($F \in \{3, 4\}$), and Σ is the action ($\Sigma \in$ {receive, verify, success, fail}), where "receive" indicates receive the transaction request from the client and the block from the accountant node in phase ⑤, "verify" means verify the block from the accountant node, "success" is thought that the evaluator has made a right decision, and "fail" represents the decision of a new block by an evaluator is wrong because their judgment is the opposite of that of the majority.

As shown in Figure 6(c), we define the Distinguishing Automaton for the behavior of a follower node, in which the initial state is 0 ($S = 0$), acceptable states are [4, 5] ($F \in \{4, 5\}$), and Σ is the action, where "receive" indicates receive the block that from the accountant node and verified by the evaluator group, "addition new block" means that add this block to the local blockchain of the follower node in phase ⑥ and the behavior of honest nodes is similar universally in this phase, "send systemcall" is the action that sends systemcall of previous to the evaluator group in phase ①, and "abnormal" and "normal" represent the result of risk-node assessment mechanism executed by the evaluator group in phase ①, where $\Sigma \in$ {receive, addition new block, send systemcall, abnormal, normal}.

Criterion 1: Byzantine behavior of accountant. Let $<1, tr>$ denote the behavior of accountant node 1. It will be considered that node 1 is a Byzantine node when node 1 reaches state 5 because the new block generated by the accountant node is judged to be invalid in phase ⑤, which may indicate that the transaction request from the client was maliciously tampered with by accountant.

Criterion 2: Byzantine behavior of evaluator. Let $<1, tr>$ denote the behavior of evaluator node 1. It will be considered that node 1 is a Byzantine node when node 1 reaches state 4 because their judgment is the opposite of that of the majority in phase ⑤, which may indicate that collusion attacks happened in the accountant node and the evaluator node.

Criterion 3: Byzantine behavior of follower. Let $<1, tr>$ denote the behavior of follower node 1. It will be considered that node 1 is a Byzantine node when node 1 reaches state 4 because their behavior of block addition is different from that of the majority in phase ⑥, which will be decided in the risk-node assessment mechanism.

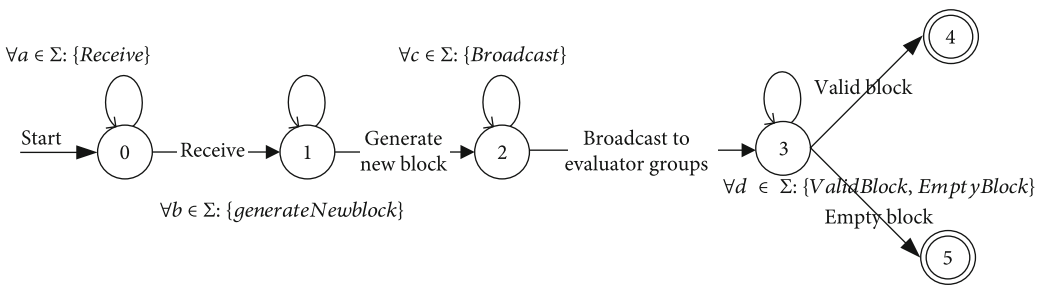
Node i was judged as a Byzantine node according to Criterion 1, Criterion 2, and Criterion 3. The Byzantine node will be added in RNL to prevent it from causing more significant losses to the blockchain system, which means that it can only serve as the follower node. It is an acceptable solution that the organization of the Byzantine nodes needs to pay a portion of stake interests or fines to reward those who had honest nodes. In addition, some compensation mechanisms can be added based on agreements reached between organizations in permissioned blockchain. For example, the organizations of the honest nodes can remove the Byzantine node from RNL through the smart contract after receiving the reward.

5. Performance and Evaluations

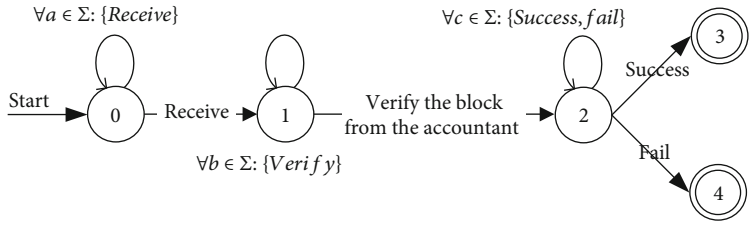
5.1. Theoretical Analysis

5.1.1. Complexity of Consensus. Every two nodes' communication between each other and a three-stage commit process was required in the PBFT algorithm before adding blocks to the blockchain, which leads to a high communication complexity of PBFT. However, the state classification mechanism of the RAC algorithm makes it possible to add blocks without high communication costs.

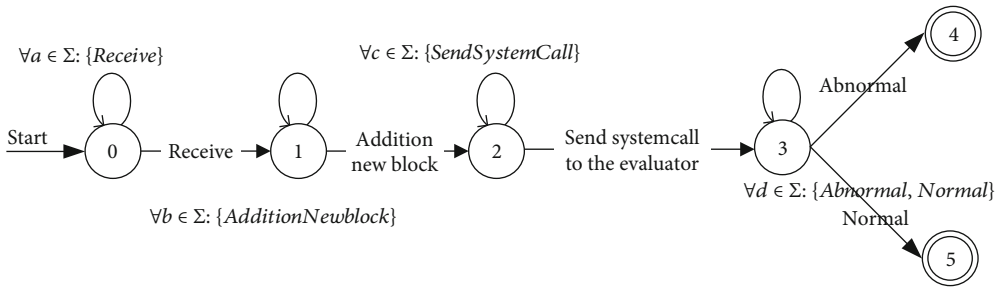
We divide the RAC algorithm into three main phases to facilitate the analysis: accountant selection stage (including



(a) The state graph of accountant node



(b) The state graph of evaluator node



(c) The state graph of follower node

Figure 6: The state graph of RAC.

phase ①, phase ②, and phase ③ of RAC algorithm), block addition stage (including phase ④ and phase ⑤ of RAC algorithm), and transaction confirmation stage (including phase ⑥ of RAC algorithm). Let n denote the number of nodes, n_e indicates the number of evaluator nodes, and n_f is the number of follower nodes.

In the accountant selection stage, the follower node needs to go through two steps of sending systemcall to the evaluator group and sending voting request to the rest of the follower nodes when it becomes an accountant node, so the communication complexity of this stage is $O(n + n_f)$. In the block addition stage, only the accountant node needs to send messages to the evaluator node and the follower node to reach consensus. No communication is needed between the evaluator nodes and follower nodes, so the communication complexity of the RAC algorithm at this stage is $O(n_e + n_f)$. The communication complexity of RAC algorithm in the transaction confirmation stage is $O(n1/2n)$ because only more than half of the follower nodes are required to complete the verification. Therefore, the communication complexity of the RAC algorithm is $O(2n_f + 3/2n + n_e)$, which is smaller than $O(n^2)$ in PBFT.

5.1.2. Security Analysis. Some malicious attacks can threaten the permissioned blockchain. In this section, we situate our discussion about the security ability of the RAC algorithm.

Collusion attack and Double Spending attack: It would have an unpredictable impact on the blockchain system if the consensus protocol was not protected against Double Spending attacks. The consensus result of the RAC algorithm is deterministic rather than probabilistic, which can avoid Double Spending attacks to a certain extent. However, a Double Spending attack occurs when the accountant and the malicious evaluator collude in the RAC algorithm. Thus, the defense mechanism for the Double Spending attack can also be equivalent to that for the collusion attack. Collusion and Double Spending attacks cannot occur if the percentage of malicious evaluator nodes does not exceed 50% in the RAC algorithm because blocks can only be added after more than 50% of the evaluator nodes have passed the checksum, which also means that the RAC algorithm can tolerate at most 49% of Byzantine evaluator nodes in the network. In the blockchain network, the attacker may use the 51% attack as a sub-attack of the double-spend or collusion attack.

Sybil attack: Sybil attack indicates that the attacker attempts to control the network by creating many fraudulent identities, which can be used to help the attacker gain voting power or broadcast false messages in the blockchain system. No node of the decentralized system can know how many nodes are involved in the peer network; they can only judge the global situation by the data they receive. The attackers can take the means to maliciously control the honest nodes

Table 13: Comparisons between two types of consensus algorithm in blockchain.

Consensus algorithms	Complexity of consensus	Crash fault tolerance	Byzantine fault tolerance	Sybil attack	Targeted attack
PBFT	$O(n^2)$	33%	33%	Vulnerable	Vulnerable
Tendermint BFT	$O(n^2)$	33%	33%	Vulnerable	Vulnerable
RAFT	$O(n)$	33%	/	Vulnerable	Vulnerable
CRAFT	$O(n)$	51%	51%	/	Vulnerable
Beh-Raft	$O(n)$	51%	51%	Safe	/
RAC	$O(n)$	51%	51%	Safe	Safe

by disguising themselves as multiple nodes and influencing the behavior of the honest nodes. Two approaches are used in the RAC algorithm to defend against Sybil attacks. On the one hand, nodes need to complete registration in the identity authentication mechanism and be assigned a unique identity before joining the permissioned blockchain. Registration needs to be endorsed by a specific organization, which can enhance the cost for attackers to register multiple identities in the network maliciously. On the other hand, the RAC algorithm uses Risk-Node Assessment Mechanism to defend against Sybil attack. When the attacker bypasses the identity authentication mechanism, the node where the abnormal behavior of forging identity occurs will be readily determined by the Risk-Node Assessment Mechanism because there will be a massive difference between the Byzantine node and the honest node in their system call record.

Targeted attack: Some consensus algorithms are vulnerable to Targeted attacks because the accountant node can be inferred. For example, an attacker launches a DOS attack on the current accountant node, causing it to be inoperative until a malicious node controlled by the attacker becomes the accountant. Thus, the periodic or irregular accountant replacement method used to defend against Targeted attacks is only valid under the assumption that the attacker cannot immediately damage the accountant node. RAC algorithm can achieve defense against Targeted attacks in an effective way due to the function of term.

The accountant node cannot be directly generated from follower nodes but needs to go through the candidate state and be voted by more than half of the nodes before becoming an accountant node. A term is divided into accountant election and regular operation. Only one accountant node existed in a certain term. The term can keep the real identity of the accountant uncertain in the accountant election stage until the accountant provides services for blockchain networks in the normal operation stage. But when the accountant node is elected successfully, it is too late to attack the accountant in this term because the accountant node has been able to serve normally without malicious control by the attacker. If an attacker wants to perform a Targeted attack on the permissioned blockchain, the only thing it can do is corrupt the node randomly before the next term. The randomness of different network environments and latency of each node makes it impossible for the attacker to predict which node will become the accountant node in the RAC algorithm. Therefore, it is a random behavior with a probability that can be calculated in Equation (12).

$$P_1(\text{Attack the accountant of next term}) = \frac{1}{N}. \tag{12}$$

In addition, an attacker who wants to take malicious control of a node will generate some traces of the attack chain in this node, such as Reconnaissance (use social engineering to understand the target), Weaponization (Targeted attack tool creation), Delivery (delivery of the attack tool to the target node), Exploitation (the attack tool is triggered on the node to realize control based on the system's application or operating system vulnerabilities), Installation (remote control program of the installation), and Command & Control (the C2 channel be established between the compromised node and internet controller server).

Therefore, an attacker's successful malicious control of a node will cause the system call of that node to be completely different from other nodes in the permissioned blockchain. Node assessment mechanism in the RAC algorithm can detect malicious behavior because the system call sequence of all nodes needs to be analyzed by the evaluator group before the accountant node election in the new term. The probability P_2 can be calculated by Equation (13), where Pr indicates the number of nodes that be maliciously controlled to becoming accountant nodes by attackers.

$$P_2 = \lim_{\text{pr}\longrightarrow 0} \frac{\text{Pr}}{N}. \tag{13}$$

5.1.3. Summary of Theoretical Analysis. We assume that there are n nodes in the network and $x_{i,j}$ indicates the j-th transaction in the i-th block. n_d is the number of failed nodes due to downtime or network errors. The function $f(x_{i,j}) \longrightarrow \{0, 1\}$ is used to denote the result of consensus, where 0 presenting invalid and 1 presenting valid of each transaction to reach a consensus. We can draw the following conclusions about the theoretical performance of the RAC algorithm and summarize them in Table 13 based on the previous analysis.

(i) Agreement. Consensus can be reached normally when the number of failed nodes is less than 50%

Table 14: Preliminary evaluation results.

Indicator	Algorithm				
	Beh-Raft	HHRAFT	CRAFT	Tendermint BFT	RAC
Number of consensus nodes	Part	Part	All	All	Part
Accountant selection method	Voting	Voting	Voting	Polling	Voting
Consensus nodes weight	No	Yes	No	Yes	Yes
Byzantine fault tolerance	51%	16%	51%	33%	51%
Byzantine node controllability	Yes	Yes	No	No	Yes
Attack behavior costs	Middle	Low	Low	High	Middle
Resource consumption	$O(n)$	$O(n)$	$O(n)$	$O(n^2)$	$O(n)$

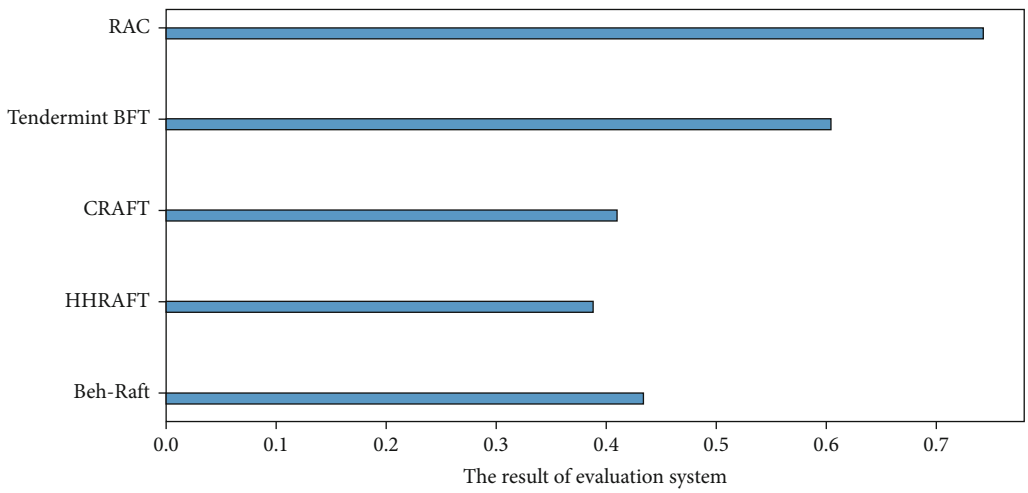


Figure 7: The experimental results of comparison with the improved consensus algorithms.

of the total number of nodes in the RAC algorithm. $f(x_{i,j}) = 1$, when $n_d/n < 50\%$.

(ii) Efficiency. The communication complexity required to reach consensus in the RAC algorithm is $O(n)$.

(iii) Security. The RAC algorithm can form an effective defense against Sybil attack, Targeted attack, Collusion attack, and Double Spending attack when the number of malicious nodes is satisfied to be smaller than 50%.

5.1.4. Comparison with the Improved Consensus Algorithms. This section compares RAC with some improved consensus algorithms of permissioned blockchains, such as Beh-Raft, HHRAFT, CRAFT, and Tendermint BFT, in terms of decentralization security and scalability under the evaluation system introduced in Section 3.1 to highlight the advantages of RAC.

We set there are n nodes in the permissioned block network. The preliminary evaluation results are shown in Table 14. An explanation of each indicator can be found in Section 2.1.

Based on the analysis results in Table 10, combined with the quantitative evaluation criteria introduced in Section 3.1, evaluation matrix $\boldsymbol{B} = (b_{ij})_{5\times7}$ is constructed as follows.

$$\boldsymbol{B} = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 0.5 & 0.5 & 0.5 & 1 & 0.5 \\ 0 & 1 & 0 & 1 & 1 \\ 0.7 & 0.3 & 0.7 & 0.5 & 0.7 \\ 1 & 1 & 0 & 0 & 1 \\ 0.6 & 0.3 & 0.3 & 1 & 0.6 \\ 0.7 & 0.7 & 0.7 & 0.3 & 0.7 \end{bmatrix}. \quad (14)$$

Positive ideal solution is $C^+ = [1, 1, 1, 0.7, 1, 1, 0.7]$ and negative ideal solution is $C^- = [0, 0.5, 0, 0.3, 0, 0.3, 0.3]$. We compared the proximity between ideal solution of Beh-Raft, HHRAFT, CRAFT, Tendermint BFT, and RAC. The experimental results are shown in Figure 7.

Under the mathematical model of blockchain evaluation proposed in Section 3.1, the RAC algorithm has a more significant advantage over other improved consensus algorithms.

5.2. Experiment Analysis. The performance of RAC algorithms can be experimentally compared and analyzed. The

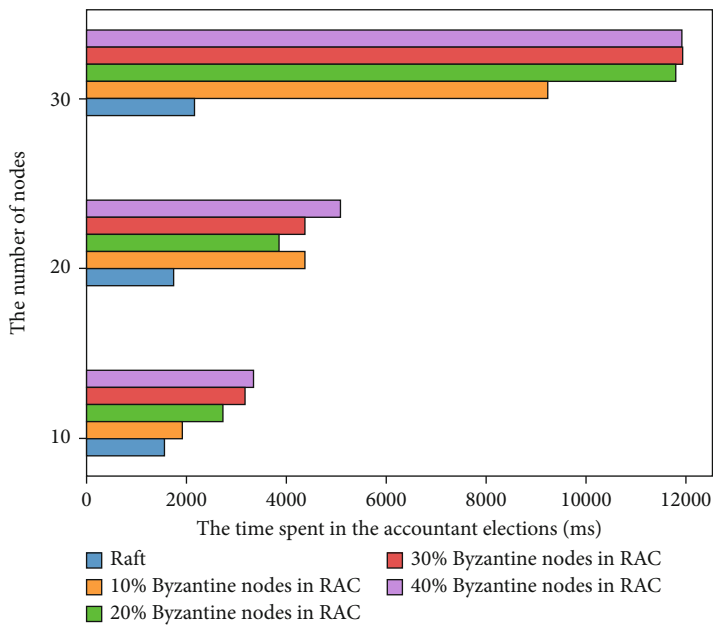


Figure 8: The cost of accountant election over different network size.

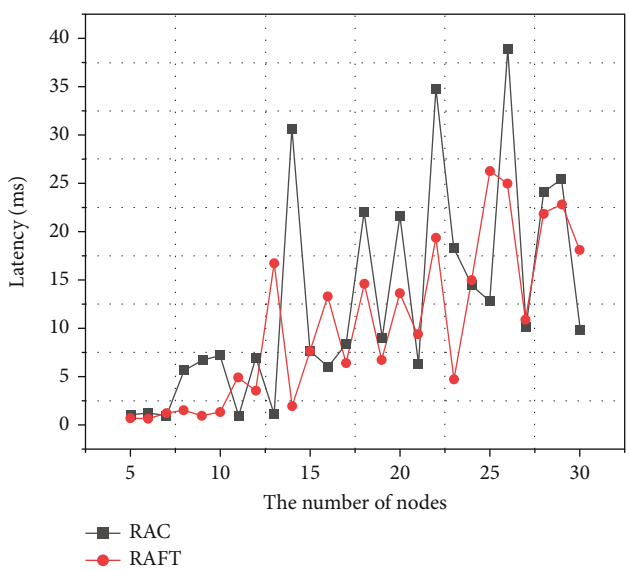


Figure 9: Comparison of latency between Raft and RAC for 5-30 nodes.

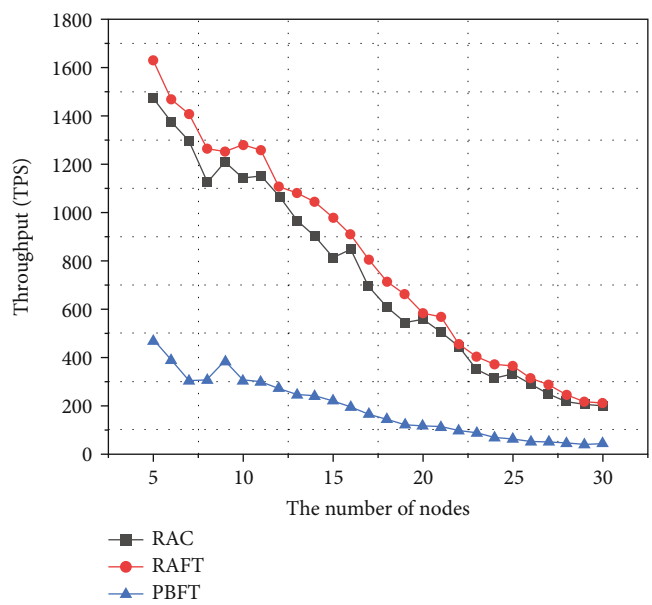


Figure 10: Comparison of throughput between Raft, PBFT, and RAC for 5-30 nodes.

RAC algorithm combines Byzantine fault tolerance and high transaction efficiency, making it applicable to real-world network environments. We conducted the experiments using an Intel Core i5 and 16 GB RAM running macOS operating system. The construction of the RAC algorithm is implemented using Golang1.14.7. To further test the performance of the proposed approach in a cluster environment, we use the technology of container, thread, and virtual machine to represent the different network nodes, and the technology of container, thread, and virtual machine is implemented with goreman0.3, docker18.09, and VMware fusion11.5.5. We compare the RAC with consensus algorithms such as Tendermint BFT and Raft, which are widely used in blockchain platforms available today in efficiency performance.

5.2.1. The Comparison of Election Cost. Accountant selection is the first phase in the consensus algorithm. The accountant selection of RAC and Raft is based on voting. Figure 8 depicts the comparison between Raft and RAC algorithms for different network sizes in the accountant election cost. The RAC algorithm consumes more cost of accountant election than the Raft algorithm due to the risk-node assessment mechanism. The cost of accountant selection has been

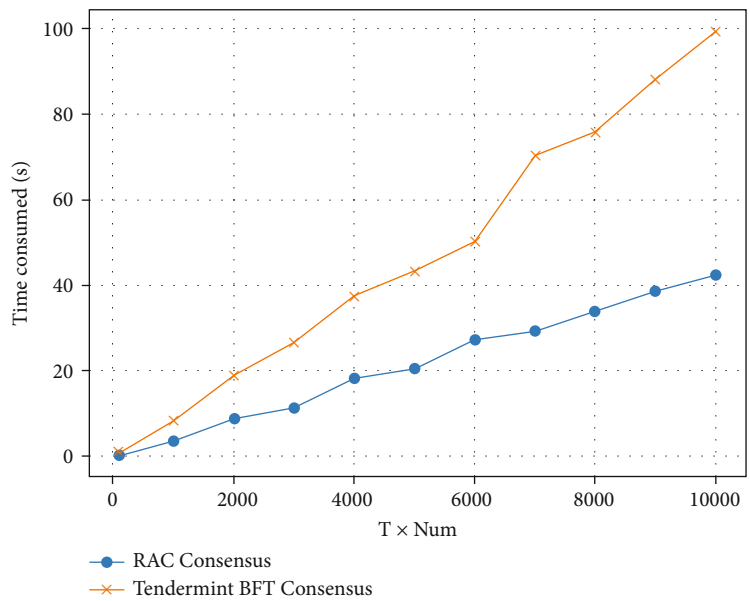


Figure 11: Comparison of time efficiency between Raft and Tendermint BFT.

Table 15: The application scenarios of RAC can be used.

Scenario	Participating methods	Characteristics	Node size
Supply chain	Internal cooperation of some companies	Medium time efficiency Many light nodes Node size fixed	<100
Smart cities	IoT application	Medium time efficiency Many light nodes Node size fixed	<1000
Vehicular ad-hoc networks	IoT application	Medium time efficiency Many light nodes Node size fixed	<1000
Smart home	IoT application	Medium time efficiency Some light nodes Node size fixed	<20
Healthcare system	Enterprises cooperate to provide external services	Medium time efficiency Some light nodes Node size fixed	<100

experimented within a network environment with 10, 20, and 30 nodes and different percentages of Byzantine nodes. The results show that as the network sizes and the proportion of Byzantine nodes increase, the cost of the RAC increases more than that of the Raft algorithm.

5.2.2. The Comparison of Transaction Confirmation Latency. Transaction confirmation latency is when a specific transaction reaches consensus in all nodes, which is one of the essential indicators of a consensus algorithm. It is calculated as shown in Equation (15).

$$\text{Latency} = \text{time}_{\text{block}} - \text{time}_{\text{request}}, \tag{15}$$

where $\text{time}_{\text{block}}$ denotes the timestamp when a transaction reach consensus, and $\text{time}_{\text{request}}$ indicates the timestamp when the transaction is initiated. The few slow nodes in both RAC and Raft algorithms do not affect the overall latency performance of the algorithm. However, the RAC algorithm needs to add a message forwarding process compared to the Raft algorithm. Due to their unique judgment mechanism of the new block, some latency performance was affected. Figure 9 depicts the latency performance of RAC and Raft algorithms with the different numbers of nodes. The overall latency performance of the RAC algorithm is reduced by about 50% in a network environment with 5-30 nodes.

5.2.3. The Comparison of Transaction Throughput. Throughput indicates the ability to process the number of client requests per unit time, and Equation (16) is the formula for calculating throughput, which can measure the efficiency

of the blockchain system.

$$\text{Throughput} = \frac{\text{number of transactions } T}{\text{the time of } T \text{ transactions reaching a consensus}}. \tag{16}$$

The value of T is taken as 1000, and the experimental results of throughput performance comparison of different consensus algorithms are shown in Figure 10. In a network environment with 5-30 nodes, the results show that the RAC algorithm loses about 10% of throughput performance due to its security mechanism compared to the Raft algorithm. However, compared with the Byzantine fault-tolerant consensus algorithm such as PBFT, the RAC algorithm still has an advantage in terms of throughput performance.

5.2.4. The Comparison of Time Efficiency. We compare the time efficiency of different consensus algorithms under the simulation environment that uses a 100-node to generate concurrency transactions, and the size of every block generated is 256 KB. As shown in Figure 11, the time efficiency of RAC at a better level than Tendermint BFT. Under the same size of blocks, RAC are superior to Tendermint BFT due to communicational complexity of Tendermint BFT is up to $O(n^2)$, while RAC have reduced it to a smaller value; as a result, the time consumed of Tendermint BFT is more than RAC with the large number of transactions.

5.2.5. Summary of Experiment Analysis. Compared to the Raft algorithm, the RAC requires an additional security mechanism, so there is a loss of efficiency in the simulation environment. However, RAC has advantages in latency and throughput performance compared to the Byzantine fault-tolerant consensus algorithm like Tendermint BFT. Thus, our results show that the RAC algorithm proposed in this paper can reach a better performance balance in efficiency and security.

6. Conclusions and Future Works

6.1. Conclusions. This paper proposes a consensus algorithm, RAC, for permissioned blockchain. First, we introduce the implementation process of the RAC algorithm, which focuses on the detailed phases, Risk-Node Assessment Mechanism, Reward and Punishments Mechanism of the algorithm. Second, we theoretically analyze the performance of the RAC algorithm and compare the RAC algorithm with other improvement consensus algorithms. The result shows that the RAC algorithm has high scalability (the communication complexity of the RAC algorithm is $O(n)$) and can provide better protection against blockchain attacks such as Sybil attacks and Targeted attacks. Finally, we experimentally compare the efficiency performance of the RAC algorithm with the consensus algorithms (Raft and Tendermint BFT) that are widely used in real-world blockchain systems. The results show that the RAC algorithm can achieve lower latency performance and higher throughput under a Byzantine network environment (the RAC algorithm loses about 10% of throughput performance compared to the Raft algorithm; however, Raft can be only used in a non-Byzantine environment; RAC algorithm still has an advantage in terms of throughput performance compared with Tendermint BFT under Byzantine network environment). In summary, RAC can reach a better balance in the performance of decentralization, security, and scalability.

6.2. Future Work. From what has been concluded in theoretical and experimental analyses, RAC can be well applied to the new business model based on blockchain to help a group of entities under cooperation and wants to eliminate the dependence on centralization certification organization. Table 15 lists some real-world applications of permissioned blockchain where RAC can be selected as the consensus algorithm. These scenarios are characterized by fixed node size and the presence of a certain number of light nodes but with certain security and time efficiency performance requirements. In the future, we will further use RAC in some complex scenarios and explore the performance of RAC on different real-world applications.

Even though RAC is an improved consensus solution in permissioned blockchain, it still has some limitations:

(1) RAC has a lower processing speed than traditional centralized systems, which may limit the application of the RAC algorithm

(2) Reputation-based incentive mechanism should be established and enforced to reduce the technical barriers and costs of the punishments mechanism we proposed in RAC

(3) The security of RAC needs to improve by developing some new technologies. The Risk-Node Assessment Mechanism of RAC introduces the threat of re-centralization, which destroys the distributed security features of the blockchain system to a certain extent, which is conducive to implementing 51% attacks and Double Spending attacks

Data Availability

All data included in this study are available upon request by contact with the corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research work was supported by the Ministry of Education Industry-University Cooperation Collaborative Education Project.

References

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008, https://bitcoin.org/bitcoin.pdf.

[2] Y. Yuan and F. Wang, "Blockchain and cryptocurrencies: model, techniques, and applications," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 48, no. 9, pp. 1421–1428, 2018.

[3] B. Bhushan, P. Sinha, K. M. Sagayam, and J. Andrew, "Untangling blockchain technology: a survey on state of the art, security threats, privacy services, applications and future research directions," *Computers & Electrical Engineering*, vol. 90, article 106897, 2021.

[4] K. Francisco and D. Swanson, "The supply chain has no clothes: technology adoption of blockchain for supply chain transparency," *Logistics*, vol. 2, no. 1, 2018.

[5] A. D. Liu, X. H. Du, N. Wang, and S. Z. Li, "Research progress of blockchain technology and its application in information security," *Journal of Software*, vol. 29, no. 7, pp. 2092–2115, 2018.

[6] M. D. Liu, Z. N. Chen, and Y. J. Shi, "Research progress of blockchain in data security," *Chinese Journal of Computers*, vol. 43, no. 1, pp. 1–28, 2020.

[7] O. Novo, "Blockchain meets IoT: an architecture for scalable access management in IoT," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1184–1195, 2018.

[8] A. I. Sanka, M. Irfan, I. Huang, and R. C. C. Cheung, "A survey of breakthrough in blockchain technology: adoptions, applications, challenges and future research," *Computer Communications*, vol. 169, pp. 179–201, 2021.

[9] Y. Z. Liu, J. W. Liu, Z. Y. Zhang, T. G. Xu, and H. Yu, "Overview on blockchain consensus mechanisms," *Journal of Cryptologic Research*, vol. 6, no. 4, pp. 395–432, 2019.

[10] G. Sun, M. Dai, J. Sun, and H. Yu, "Voting-based decentralized consensus design for improving the efficiency and security of consortium blockchain," *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6257–6272, 2021.

[11] S. King and S. Nadal, "PPCoin: peer-to-peer crypto-currency with proof-of-stake," 2012, https://www.chainwhy.com/upload/default/20180619/126a057fef926dc286accb372da46955.pdf.

[12] D. Larimer, "Delegated proof-of-stake (DPOS)," 2014, https://how.bitshares.works/en/master/technology/dpos.html.

[13] D. Mazieres, "The stellar consensus protocol: a federated model for internet-level consensus," *Stellar Development Foundation*, vol. 32, pp. 1–45, 2015.

[14] X. Fu, H. Wang, and P. Shi, "A survey of blockchain consensus algorithms: mechanism, design and applications," *SCIENCE CHINA Information Sciences*, vol. 64, no. 2, pp. 1–15, 2021.

[15] Q. F. Shao, Z. Zhang, Y. C. Zhu, and A. Y. Zhou, "Survey of enterprise blockchains[J]," *Journal of Software*, vol. 30, no. 9, pp. 2571–2592, 2019.

[16] D. Ongaro and J. Ousterhout, "In search of an understandable consensus algorithm," in *USENIX Annual Technical Conference*, pp. 305–319, Stanford University, Philadelphia, United States., 2014.

[17] M. Castro and B. Liskov, "Practical Byzantine fault tolerance and proactive recovery," *ACM Transactions on Computer Systems*, vol. 20, no. 4, pp. 398–461, 2002.

[18] D. Mingxiao, M. Xiaofeng, Z. Zhe, W. Xiangwei, and C. Qijun, "A review on consensus algorithm of blockchain," in *2017 IEEE international conference on systems, man, and cybernetics (SMC)*, pp. 2567–2572, Banff, AB, Canada, October 2017.

[19] S. Saxena, B. Bhushan, and M. A. Ahad, "Blockchain based solutions to secure IoT: background, integration trends and a way forward," *Journal of Network and Computer Applications*, vol. 181, article 103050, 2021.

[20] B. Bhushan, C. Sahoo, P. Sinha, and A. Khamparia, "Unification of Blockchain and Internet of Things (BIoT): requirements, working model, challenges and future directions," *Wireless Networks*, vol. 27, no. 1, pp. 55–90, 2021.

[21] D. Dasgupta, J. M. Shrein, and K. D. Gupta, "A survey of blockchain from security perspective," *Journal of Banking and Financial Technology*, vol. 3, no. 1, pp. 1–17, 2019.

[22] B. Bhushan, A. Khamparia, K. M. Sagayam, S. K. Sharma, M. A. Ahad, and N. C. Debnath, "Blockchain for smart cities: a review of architectures, integration trends and future research directions," *Sustainable Cities and Society*, vol. 61, article 102360, 2020.

[23] S. Gilbert and N. Lynch, "Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services," *ACM SIGACT News*, vol. 33, no. 2, pp. 51–59, 2002.

[24] V. Buterin, "A guide to 99% fault tolerant consensus," 2018, https://vitalik.ca/general/2018/08/07/99_fault_tolerant.html.

[25] J. Shixiong, Z. Xiaodan, G. Jingguo et al., "Overview of blockchain consensus algorithm," *Journal f Cyber Security*, vol. 6, 2021.

[26] G. T. Nguyen and K. Kim, "A survey about consensus algorithms used in blockchain," *Journal of Information Processing Systems*, vol. 14, no. 1, 2018.

[27] L. Lamport, R. Shostak, and M. Pease, "The Byzantine generals problem," *ACM Transactions on Programming Languages and Systems*, vol. 4, no. 3, pp. 382–401, 1982.

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

[29] Y. Rong, J. Zhang, J. Bian, and W. Wu, "ERBFT: efficient and robust byzantine fault tolerance," in *2019 IEEE 21st international conference on high performance computing and communications; IEEE 17th international conference on Smart City; IEEE 5th international conference on data science and systems (HPCC/SmartCity/DSS)*, pp. 265–272, Zhangjiajie, China, August 2019.

[30] E. Buchman, *Tendermint: Byzantine Fault Tolerance in the Age of Blockchains, [Ph.D. thesis]*, University of Guelph, Canada, 2016.

[31] Y. Chen, P. Liu, and W. Zhang, "Raft consensus algorithm based on credit model in consortium blockchain," *Wuhan University Journal of Natural Sciences*, vol. 2, no. 8, 2020.

[32] L. E. Wang, Y. Bai, Q. Jiang, V. C. Leung, W. Cai, and X. Li, "Beh-Raft-Chain: a behavior-based fast blockchain protocol for complex networks," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 2, pp. 1154–1166, 2021.

[33] Y. Wang, S. Li, L. Xu, and L. Xu, "Improved raft consensus algorithm in high real-time and highly adversarial environment," in *International Conference on Web Information Systems and Applications*, Springer, Cham, 2021.

[34] G. Creech and H. Jiankun, "A semantic approach to host-based intrusion detection systems using contiguousand discontiguous system call patterns," *IEEE Transactions on Computers*, vol. 63, no. 4, pp. 807–819, 2014.

[35] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 eighth ieee international conference on data mining*, pp. 412–422, Pisa, Italy, December 2008.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 6998046, 13 pages
https://doi.org/10.1155/2022/6998046

Research Article

Blockchain-Based Self-Auditing Scheme with Batch Verification for Decentralized Storage

Zhonghao Yuan, Jiaojiao Wu, Jianpeng Gong, Yao Liu, Guohua Tian, and Jianfeng Wang

School of Cyber Engineering, Xidian University, Xi'an 710126, China

Correspondence should be addressed to Jianfeng Wang; jfwang@xidian.edu.cn

Received 14 January 2022; Revised 22 March 2022; Accepted 24 May 2022; Published 26 June 2022

Academic Editor: Jinguang Han

Data owners outsource their data to remote storage providers without keeping local replicas to save their precious storage resources. However, the ownership and management of data are separated after outsourcing. How to ensure the integrity and recoverability of outsourced data becomes a significant problem. Provable Data Possession (PDP) and Proofs of Retrievability (POR) are two cryptographic protocols that enable users to verify the integrity of outsourced data. Nevertheless, the state-of-the-art PDP and POR schemes either need users to perform the complicated audit tasks by themselves or delegate these tasks to a Third-Party Auditor (TPA). Moreover, these schemes are constructed on a centralized storage framework which vulnerably suffers single-point-of-failure. In this paper, we propose a blockchain-based decentralized self-auditing scheme with batch verification. Firstly, data owners outsource their data to decentralized storage nodes, which can achieve self-auditing based on blockchain without TPA. Secondly, our scheme uses Pedersen-based polynomial commitment to significantly reduce the number of authenticators. Furthermore, we propose a batch verification algorithm, which can verify multiple proofs from different storage nodes to improve the verification efficiency. Finally, we analyze the security of our scheme and implement a gas-efficient system prototype using the smart contracts of the Ethereum Reposten test network. The results demonstrate that the scheme is practical.

1. Introduction

In recent years, cloud storage has become an essential application in our daily life. It is greatly convenient and flexible for users to store, update, and retrieve their data in the cloud [1]. Thus, more and more users outsource their data to cloud storage providers to save their local storage resources. However, the ownership and management of outsourced data are separated after users upload their data to the remote cloud. Therefore, there have arisen numerous security problems in the cloud storage. In 2015, a European data center of Google was affected by lighting strikes and permanently lost 100 GB data. In the same year, Tencent Cloud lost their customers' data, which caused significant losses for this company. Therefore, the cloud storage providers are not fully trusted. It is important for users to ensure the outsourced data are correct and intact.

Traditional cloud auditing schemes are mainly classified into two categories: Provable Data Possession (PDP) [2] and Proofs of Retrievability (POR) [3], which are two cryptographic protocols allowing users to verify the integrity of data without retrieving the data, and the POR protocol can also guarantee data recoverability. Concretely, the verifier can randomly sample a challenging set of original data, and the prover generates the corresponding integrity proof, which can be audited to ensure the integrity of the out-

sourced data. To avoid online and computational burden on data owners, existing schemes introduce a Third-Party Auditor (TPA) to help data owners to audit data [4]. However, TPA is a centralized entity that easily suffers from single-point-of-failure. In addition, many cloud auditing schemes assume that TPA will never collude with storage providers. This strong and impractical assumption may be easily broken driven by certain interests. Moreover, in the state-of-the-art auditing schemes, the data owner needs to generate homomorphic linear authenticators which grow linearly with the number of file blocks. These schemes cause an amount of computational overhead for data owners and storage redundancy for storage providers. In practical applications, we also desire to perform batch verification of multiple data auditing tasks to improve audit efficiency. Therefore, it is of great practical significance to design an efficient integrity auditing scheme without TPA.

With the rapid development of public blockchains, such as Ethereum [5]and Solana [6], more and more researchers focus on constructing a decentralized storage system by deploying blockchain. In these systems, data owners can outsource their sensitive data to decentralized storage nodes, which can arbitrarily join the network to contribute their idle storage resources. The existing blockchain-based auditing schemes [7–9] for decentralized storage can ensure the integrity of data without TPA and are naturally resistant to single point of failure due to the decentralized feature. However, these schemes delegate complex auditing tasks, including a lot of cryptographic operations, to the smart contract deployed in the blockchain, which cannot be efficiently implemented in Ethereum Virtual Machine (EVM) and may cause amounts of gas consumption. Thus, it is a challenging problem to handle the computation-intensive operations on smart contracts in blockchain-based auditing schemes for decentralized storage.

1.1. Our Contributions. In this paper, we propose a self-auditing scheme with batch verification based on blockchain for decentralized storage. Our scheme is based on blockchain technology, PDP protocol, Pedersen-based polynomial commitment, and batch opening polynomial commitment, to achieve efficient data self-auditing without TPA. In summary, our contributions can be listed as follows:

(i) We propose a blockchain-based self-auditing scheme with batch verification. We remove TPA and allow data owners to store their data to decentralized storage nodes, who can interact with the blockchain to achieve self-auditing

(ii) We adopt Pedersen-based polynomial commitment to construct the homomorphic linear authenticators, which significantly decreases storage overhead and slightly reduces authenticator computation time. In addition, we also propose a batch verification algorithm to verify multiple proofs simultaneously, which can improve verification efficiency

(iii) We implement a gas-efficient self-auditing system in the platform of the Ethereum test network and perform security analysis and performance evaluation. The results show that our scheme is efficient and practical

1.2. Related Work

1.2.1. Centralized Outsourced Storage. In 2007, Ateniese et al. [2] proposed the PDP protocol, which is the first public audit scheme to verify the data integrity in an untrusted server. Based on the homomorphic linear authenticators constructed by the RSA signature, data owners can probabilistically verify the integrity of data without retrieving original data. However, this scheme cannot guarantee data recoverability. In 2007, Juels and Kaliski [10] presented the POR protocol, which can achieve data recoverability by using the erasure coding, but their scheme only supports fixed number audits and cannot achieve public verifiability. In 2008, Shacham and Waters [3] improved POR scheme with provable security. Based on BLS signatures [11] and the bilinear pairing, the verifier can publicly verify the data integrity and recover the remote data at any time. Nevertheless, the above schemes cannot guarantee data privacy because the data is stored by plaintext in storage providers. At the same time, the data owners must always be online to audit the storage providers.

Then, there are numerous new protocols improving the PDP/POR system model, such as additional properties of privacy, multiple-replica, and batch auditing. Curtmola et al. [12] firstly proposed multiple-replica provable data possession (MR-PDP) to guarantee the data recoverability. MR-PDP scheme allows data owner that stores t replicas of a file to the cloud system to verify the multiple-file integrity. Then, there are numerous scheme attention on multiple-replica auditing [13–18]. Considering the data security, Wang et al. [4] proposed a privacy-preserving public auditing scheme that outsources the auditing tasks to TPA and supports batch auditing. Following, several schemes [19–21] concerned with privacy for data owners are proposed. To improve storage redundancy and communication overhead, Yuan and Yu [22] proposed a constant communication cost auditing scheme using the polynomial commitment for cloud storage. Besides, there are an increasing number of blockchain-based auditing schemes. Using the RSA signature, Wang et al. [23] proposed a blockchain-based private auditing scheme with small authenticators' redundancy. Their scheme can divide data into arbitrary blocks for generating authenticators which should be uploaded to the blockchain. Moreover, there are many schemes [7, 8, 24–26] that stored the verification proofs to blockchain to achieve undeniable verification interactions. Furthermore, Yuan et al. [7] and Wang et al. [8] used the smart contract to replace TPA to audit the storage providers. Recently, Su et al. [27] proposed a self-auditing scheme for multiple cloud servers without TPA. Their scheme stored the data to several cloud servers and achieved data integrity via the interactions of these servers. However, each server requires multiround interactions to acquire the other servers' proofs. Moreover, data owners need online to challenge servers.

1.2.2. Decentralized Outsourced Storage. All of the above schemes concern on centralized outsource storage framework, which has many obvious drawbacks. Firstly, centralized storage providers are vulnerable to single-point-of-failure making the users' data at risk. Secondly, it is too expensive to centralized storage compared with decentralized storage. In order to deal with the above problems, decentralized storage is becoming a hot spot.

Li et al. [28] proposed the notion of IntegrityChain, which is a decentralized storage system supporting MR-PDP. IntegrityChain is a blockchain that mainly stores the information of storage node registration, storage transactions, and auditing proofs. Based on the schemes [4, 22], Du et al. [9] also proposed a blockchain-based auditing scheme with privacy-assured in the decentralized storage network. However, due to the problem of Solidity language, currently, the smart contract cannot effectively support complex cryptographic primitives. At the same time, Francati et al. [29] utilized the blockchain nodes to store users' data. Miner audits the integrity of data when generating a new block. To ensure data recoverability, Chen et al. [30], respectively, distributed the data and its replica to cloud and decentralized storage providers. The cloud audits the replica stored in the decentralized storage nodes as TPA. With the development of blockchain technology, there are numerous blockchain-based decentralized storage projects. Both Swarm [31] and Storj [32] are decentralized storage networks that outsource audit services to centralized auditors. Unlike Swarm, based on an incentive system through the smart contract on Ethereum, Storj provides an incentive layer with cryptocurrency. Therefore, storage nodes may collude with auditors to deceive data owners. Sia [33] is a fully decentralized storage platform for uploading and downloading data between users and storage nodes. To verify the integrity of data, the storage node transfers Merkle proofs to blockchain and receives Siacoins as a reward. Unlike Sia, Filecoin [34] employs proof-of-spacetime and proof-of replication to guarantee that miners have correctly stored the committed data, which provides more robust storage security. However, the proof generation time and the computational overhead make it hard to be deployed.

1.3. Organization. This paper is organized as follows. Section 2 provides some notations and the cryptographic primitives used in our scheme. Section 3 introduces the system model and security goals of our scheme. We propose our main scheme in Section 4 and provide the security analysis in Section 5. Then, we evaluate our scheme performance and show the results in Section 6. Finally, we conclude this paper.

2. Preliminaries

In this section, we describe notations used in our scheme as Table 1. Then, some cryptographic primitives are introduced to construct our scheme.

2.1. Bilinear Map. There are three multiplicative cyclic groups, $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$, where the order of $\mathbb{G}_1$ is p. Let g_1 and g_2 be the generators of $\mathbb{G}_1$ and $\mathbb{G}_2$. We use $e : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$ to denote a bilinear map with the following properties.

Table 1: Notations and descriptions.

Notations	Description
p	Big prime
$\mathbb{Z}_p$	Cyclic group module safe prime p
$\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$	Three different multiplicative cyclic groups
g_1, h_1	The generators of $\mathbb{G}_1$
g_2	The generators of $\mathbb{G}_2$
m	The number of file blocks
d	The number of file chunks
id	The file identifier
H_1, H_2	The two hash functions
t	The number of challenged set
k	The number of storage nodes
$P_i(x)$	The challenged polynomial
$Q_i(x)$	The quotient of challenged polynomial

(i) Bilinear: for all $g \in \mathbb{G}_1$, $h \in \mathbb{G}_2$, and $a, b \in \mathbb{Z}_p$, $e(g^a, h^b) = e(g, h)^{ab}$

(ii) Computable: there exists a computable algorithm to compute the map e efficiently

(iii) Nondegenerate: $e(g_1, g_2) \neq 1$

2.2. Pedersen-Based Constant Size Polynomial Commitment. In a polynomial commitment scheme, the committer can commit a polynomial to a group element, and then, the committed polynomial can be opened at any point by a verifier. Based on an algebraic property of polynomial $f(x) \in Z[x]$: $(x - r)$ perfectly divides the polynomial $f(x) - f(r)$, $r \in \mathbb{Z}_p$, Kate et al. [35] proposed Pedersen-based polynomial commitment scheme which commits two polynomials simultaneously with the constant communication overhead. In their scheme, the proof generated by the committer proves that $\phi(r)$ is the evaluation of the committed polynomial at point r.

Firstly, we introduce the Pedersen commitment [36] that a value u with a random number r computes as:

$$C = \mathrm{com}(u, r) = g^u h^r, \tag{1}$$

where g and h are two generators of a cyclic group $\mathbb{G}$, whose order is p.

Concretely, Kate's scheme is described as below:

(i) Setup $(1^\lambda, s)$: given the security parameters λ and the degree of the polynomial s, a trusted entity generates private key $SK = \alpha$, selected randomly from $\mathbb{Z}_p$, and public key $PK = (\mathbb{G}, \mathbb{G}_1, g, g^\alpha, \cdots, g^{\alpha^{s-1}}, h, h^\alpha, \cdots, h^{\alpha^{s-1}})$. $\mathbb{G}$ and $\mathbb{G}_1$ are two multiplicative cyclic

groups, and the order of group $\mathbb{G}$ is prime p. g and h are two generators of $\mathbb{G}$, and $e : \mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_1$ is a symmetric bilinear pairing

(ii) Commit $(PK, \phi(x))$: given the PK and a polynomial $\phi(x)$, the committer chooses a polynomial $\widehat{\phi}(x)$ of degree s from $\mathbb{Z}_p[x]$. the commitment is computed as $C = g^{\phi(\alpha)} h^{\psi(\alpha)} \in \mathbb{G}$

(iii) CreateWitness $(PK, \phi(x), \widehat{\phi}(x), r)$: given the $r \xleftarrow{R} \mathbb{Z}_p$, the committer calculates $\psi(x) = (\phi(x) - \phi(r))/(x - r)$ and $\widehat{\psi}(x) = (\widehat{\phi}(x) - \widehat{\phi}(r))/(x - r)$. Then, the witness ω is calculated as $g^{\psi(\alpha)} h^{\widehat{\psi}(\alpha)}$ based on PK. Finally, the algorithm outputs $\{r, \phi(r), \widehat{\phi}(r), \omega\}$

(iv) VerifyEval $(PK, C, r, \phi(r), \widehat{\phi}(r), \omega)$: given the output of the algorithm CreateWitness, it is verifiable that $\phi(r)$ is the evaluation at the point r of polynomial which is committed by C as below:

$$e(C, g) = e\left(w_i, \frac{g^{\alpha}}{g^{i}}\right) e\left(g^{\phi(i)} h^{\widehat{\phi}(i)}, g\right). \tag{2}$$

2.3. Batch Opening Polynomial Commitment. To improve the verification efficiency of Kate's scheme [35], Boneh et al. [37] proposed two polynomial commitment schemes which can open proof for multiple points and polynomials at the same time. The first scheme is introduced to construct our scheme. The opening proof of their first scheme is constant size as same as Kate's scheme, but the verifier will have a large amount of computation if there are many distinct evaluation points. The concrete scheme is described as follows.

(i) Setup $(1^{\lambda}, s, t)$: s is the degree of the polynomial, and t is the max number of opening points. Then, a trusted authority uniformly chooses α as private key from $\mathbb{Z}_p$ denoted by SK and computes public key PK as $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_1^{\alpha}, \cdots, g_1^{\alpha^{s-1}}, g_2, g_2^{\alpha}, \cdots, g_2^{\alpha^t})$. Let $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ be multiplicative cyclic groups where the order of $\mathbb{G}_1$ is p. g_1 and g_2 are the generators, respectively, selected from $\mathbb{G}_1$ and $\mathbb{G}_2$. e is an asymmetric bilinear pairing: $\mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$

(ii) Commit $(PK, \phi(x))$: the algorithm outputs $C = g_1^{\phi(\alpha)}$

(iii) CreateWitness $(PK, r, \{\phi_i(x)\}_{i=1}^{n}, T = \{r_i\}_{i=1}^{t}, \{S_i \subset T\}_{i=1}^{n})$: given a random $\gamma \in \mathbb{Z}_p$ sent by the verifier, several polynomials $\{\phi_i(x)\}_{i=1}^{n}$, and their individual opening point subset $\{S_i \subset T\}_{i=1}^{n}$, a prover computes the polynomial $h(x)$ as $\sum_{i=1}^{n} \gamma^{i-1} \cdot (\phi_i(x) - \phi_i(r_i))/Z_{S_i(x)}$, where $Z_{s_i(x)}$ is the polynomial of $\prod_{r_i \in S_i} (x - r_i)$. The witness ω is the polynomial commitment of $h(x)$, computed as $g_1^{h(\alpha)}$

(iv) VerifyEval $(PK, C, r, \phi(r), \widehat{\phi}(r), \omega)$: verifier computes $F = \prod_{i=1}^{n} e(\gamma^{i-1} \cdot (\mathrm{cm}_i - [r_i(x)]_1), Z_i)$ and verifies the following equation $F \stackrel{?}{=} e(\omega, g_2^{Z_T(\alpha)})$, where Z_i is a polynomial commitment of $g_2^{Z_{T \setminus S_i}(x)}$

3. Problem Statement

3.1. System Model. We propose a blockchain-based public self-auditing scheme that ensures data integrity and recoverability for decentralized storage. The framework of our scheme is shown in Figure 1, which obtains three roles: data owner, storage node that belongs to a decentralized network, and blockchain.

(i) Data owner (DO): DO outsources his data to several distributed storage nodes to save storage space. Before uploading data to nodes, DO processes data in advance to guarantee data privacy and recoverability

(ii) Storage node (SN): SN, a peer of the decentralized network, wants to outsource his storage resources to gain more interest. Our scheme assumes that several nodes that store the same DO data cannot collude with each other. SN should generate proof for each auditing task and interact with the blockchain

(iii) Blockchain (BC): due to the transparency and tamper-proof property, blockchain servers as a trusted third party in our scheme. In the auditing stage, BC will challenge SN and store the auditing proof generated by the SN

As shown in Figure 2, our scheme is outlined in detail. In the setup stage, the DO divides the outsourcing file F into m blocks, and each $2n$ blocks form a chunk. For each chunk, DO generates an authenticator. Then, in the storage stage, DO will distribute file chunks and corresponding authenticators to several SNs. During the self-auditing stage, each SN firstly calculates the challenged set via the information of the blockchain header. Secondly, each SN will generate the proof according to the challenged data chunks and transfer the proof to the smart contract. Finally, each SN gets the other node's proofs from the smart contract, verifies the correctness of all proofs, and transfers the audit report to the smart contract.

3.2. Threat Model and Security Goals. Considering the fairness of the scheme, we describe malicious behaviors as follows. Besides, we assume that at least one of the nodes that stored DO's data is honest. Firstly, the SN may delete data rarely accessed by DO to save storage costs for more interest or directly leave the decentralized network. Secondly, due to various accidents, such as hardware and software failure, the outsourced data stored on the SNs may be tampered or deleted. In order to his reputation, the SN may hide the facts of data loss until the time of data retrieval. Finally, DO may generate erroneous metadata to SNs for gaining more

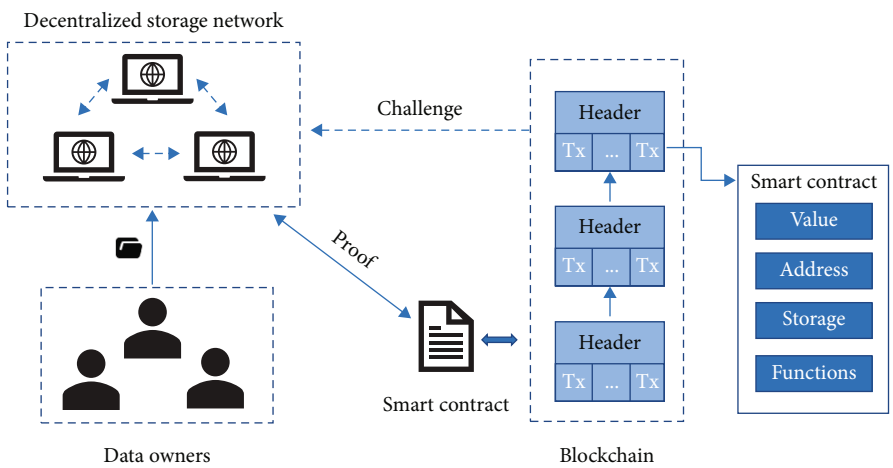


Figure 1: System model.

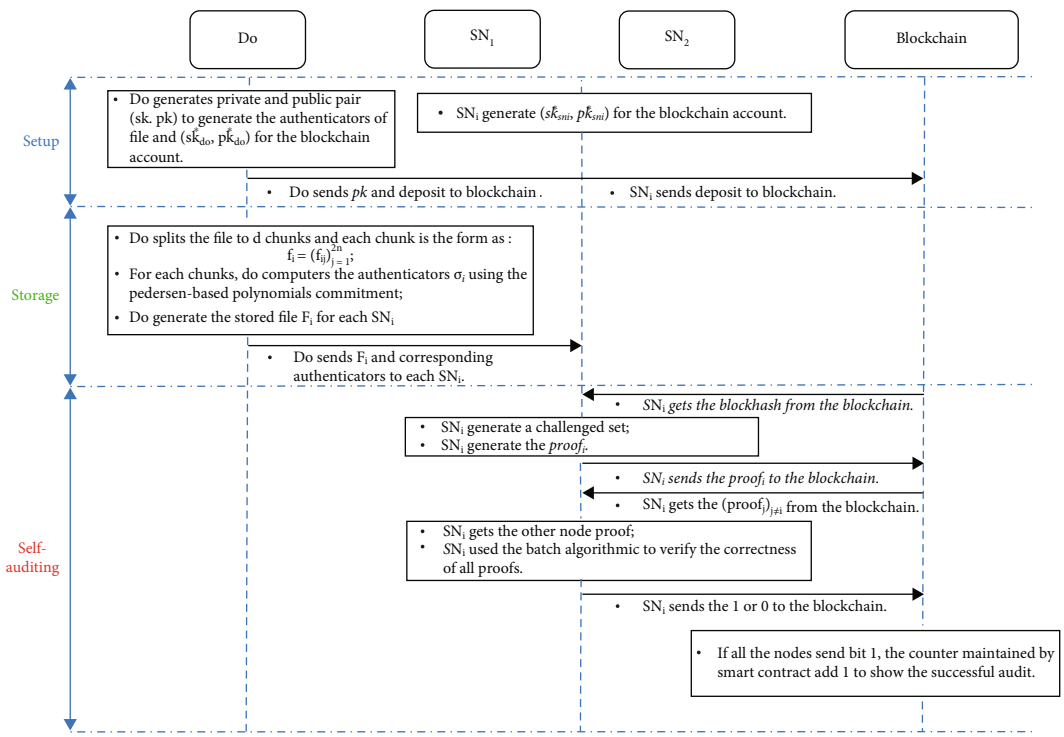


Figure 2: Overview of our proposed self-auditing scheme.

interests. In our scheme, we want to achieve the security goals as follows.

(i) Data privacy: we should protect DO's data privacy that SN and malicious adversaries cannot extract data contents

(ii) Storage correctness: we should guarantee that if SNs can pass each audit, they must correctly store the DO's data

(iii) Batch auditing: we require SNs to correctly verify multiple audit tasks at one time to improve the scheme's efficiency. If the batch auditing can be passed, SNs must correctly store the DO's data

(iv) Fairness: we should ensure the fairness of incentive mechanism, which will reward honest participants and punish malicious participants

4. Our Main Scheme

In this section, we first describe the formal definition of our scheme. Then, we propose the main idea of our scheme. In the end, we present our scheme in detail. The main algo-

rithms of our scheme are defined below. During the following presentation, we describe our scheme from the view of DO and SN_i.

(i) Setup $(1^\lambda, n, k)$: given the security parameters λ, the number of blocks in each chunk n and the number of opening points k. Then, this algorithm outputs private and public key pairs (SK, PK) for DO to preprocess the uploading file, and the blockchain accounts $\{sk^*_{sni}, pk^*_{sni}\}$ and $\{sk^*_{do}, pk^*_{do}\}$ for SN_i and DO

(ii) TagGen (SK, F): this algorithm inputs the DO's SK and uploading file F. It outputs the processed file $\widehat{F}$ and corresponding authenticators $\{\sigma_i\}^d_{i=1}$

(iii) FileDistribution(pk^*_{sni}, k): given the pk^*_{sni} and the number of storage nodes k, this algorithm outputs the file chunks and authenticators of each SN_i required for storage

(iv) ChallGen $(pk^*_{sni}, \text{blockhash}, t)$: given the pk^*_{sni}, the number of challenged set t and the blockhash on the blockchain, this algorithm outputs the challenged set of each SN_i, denoted by $C_i = \{\{c_j, d_j\}^t_{j=1}, r_i\}$

(v) Self-auditing $(PK, \{C_i\}^t_{i=1}, \widehat{F}, \{\sigma_i\}^d_{i=1})$: given the PK, the number of challenged set and metadata, this algorithm outputs 1 or 0, where 1 represented this round audit is successful and 0 failure

4.1. Main Idea. The overview of our scheme is shown in Figure 2. We propose a blockchain-based self-auditing scheme to solve the problem that the traditional schemes introduced TPA. The main idea is that the storage nodes act as verifiers and interact with the smart contract to complete each audit. Concretely, the data owner first encrypts and encodes the file for data privacy and retrievability. Then, selecting several storage nodes, the data owner distributes part of the file to each node. For each audit, every node generates the proof through the information of blockchain and transfers the proof to the blockchain. Each node serves as a verifier to verify the proofs of other nodes. If every proof is passed the verification, each node transfers the successful message to the smart contract, such as 1. The smart contract automatically maintains a counter recording the number of successful audits. When the data owner retrieves the file, the smart contract sends the data owner's deposit to the storage nodes according to the counter.

4.2. The Concrete Scheme. We use three different multiplicative cyclic groups in our protocol, $\mathbb{G}_1, \mathbb{G}_2$, and $\mathbb{G}_T$. And the order of the group $\mathbb{G}_1$ is prime p. Let three generators g_1, h_1, and g_2, respectively, selected from $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$, $e : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$ be a bilinear pairing. $H_1 : \{0,1\}^* \longrightarrow \mathbb{G}_1$ and $H_2 : \{0,1\}^* \longrightarrow \{0,1\}^\lambda$ are expressed as two collision-resistant hash functions. Besides, the stored file is denoted to F separated into m blocks. Each $2n$ blocks form a set of data for generating authenticators and saving storage space. It should be noted that each block is the element of group $\mathbb{Z}_p$ for the security of the audit. Furthermore, our scheme can support batch auditing where the number of audit proofs is k.

Setup$(1^\lambda, n, k)$. The data owner randomly selects two elements, α and υ from group $\mathbb{Z}_p$, as his private key and computes $\{g_1^{\alpha^j}, h_1^{\alpha^j}\}^{n-2}_{i=1}$, and $\{g_2^{\alpha^j}, g_2^{\upsilon(\alpha)^j}\}^t_{j=1}$ as part of the public key. Let $\varepsilon = g_2^\alpha$ and $\beta = g_2^{\alpha\upsilon}$ in order to conveniently express in a single audit. Therefore, the private key SK is (α, υ), and the public key PK is $(\varepsilon, \beta, H_1, H_2, g_1, g_2, h_1, \{g_1^{\alpha^j}, h_1^{\alpha^j}\}^{n-2}_{i=1}$ and $\{g_2^{\alpha^j}, g_2^{\upsilon(\alpha)^j}\}^t_{j=1})$. Meanwhile, data owner and each storage node SN_i generate the private and public keys of the blockchain account, respectively, denoted by $\{pk^*_{sni}, sk^*_{sni}\}$ and $\{pk^*_{do}, sk^*_{do}\}$.

TagGen(SK, F). Before uploading a file F to decentralized storage network, data owner should process F using symmetric encryption algorithms to protect data privacy, such as AES, and erasure-correcting code [38] to reinforce data recoverability. Then, the file F is split into m blocks $\{f_1, f_2, \cdots f_m\}$ and each $2n$ blocks form a chunk. Thus, the number of chunks is $\lceil m/2n \rceil$ represented by d. The file is the form of $\{F_1, F_2, \cdots F_d\}$, and each chunk F_i can be denoted as $\{f_{i,1}, f_{i,2}, \cdots f_{i,2n}\}$. We use $\widehat{F}$ to denote the processed file.

It is noteworthy for data owner that the last chunk may need padding. For each F_i, the data owner generates a corresponding homomorphic authenticator utilizing the Pedersen-based polynomial commitment [22, 35]. An authenticator is computed as the following operations, where id is the file identifier randomly selected from $\mathbb{Z}_p$ and $H_1(id\|i)$ is the index information of chunk bound in the authenticator.

$$\begin{aligned}\sigma_i &= \left(\prod_{j=1}^{n} g_1^{f_{i,j}\alpha^{j-1}} \cdot \prod_{j=n+1}^{2n} h_1^{f_{i,j}\alpha^{j-n-1}} \cdot H_1(id\|i)\right)^\upsilon \\ &= \left(g_1^{M_{i1}(\alpha)} \cdot h_1^{M_{i2}(\alpha)} \cdot H_1(id\|i)\right)^\upsilon.\end{aligned} \tag{3}$$

Using the polynomial commitment based on Pedersen, we can commit two polynomials at one time without increasing the amount of calculation, which greatly alleviates the storage redundancy and I/O overhead. As shown in Equation (3), $M_{i1}(x)$ and $M_{i2}(x)$ are bound to the authenticator in the form of polynomial commitment, and their parameters are the first and latter half part of each chunk F_i. Besides, considering the computational cost for SNs in the self-auditing stage, the number of each chunk has an upper limit of $2n$.

FileDistribution(pk^*_{sni}, k). In order to ensure data recoverability, the data owner selects several nodes in the decentralized distributed network, and each node stores a part of the file. In detail, there are k storage nodes selected to store file chunks and accompanying authenticators, $\{F_i, \sigma_i\}^d_{i=1}$.

Each storage node SN_i should store metadata pair set $\{\{F_j\},\{\sigma_j\}\}_{j\in I}$ whose index set is computed as $I=\{H_2(pk^*_{sni}\|j)==i \bmod k\}_{j=1}^{d}$. Then, the data owner transfers the divided metadata pairs to each node in a secure channel and sends deposit to smart contract compiling in advance on the blockchain. After nodes receive the corresponding metadata, they also send deposit to the smart contract.

ChallGen$(pk^*_{sni}, t, \text{blockhash})$. Utilizing the information on the blockchain, SN_i generates a challenged set using the hash function. The height of the blockchain B_{init} is taken as the initial audit point where DO's deposit is sent to the blockchain. And each audit is generated for every B_{interval} blocks. Concretely, SN_i generates a challenged set $C_i=\{\{c_j,d_j\}_{j=1}^{t}, r_i\}$, where $c_j=H_2(\text{blockhash}\|pk^*_{sni}\|j)$, $d_j=H_2(\text{blockchain}\|c_j)$, and $r_i=H_2(\text{blockhash}\|pk^*_{sni})$.

Self-Auditing$(PK, \{C_i\}_{i=1}^{t}, \widehat{F}, \{\sigma_i\}_{i=1}^{d})$. Based on the file and authenticators, the node SN_i computes:

$$\sigma=\prod_{i=1}^{t}\sigma_{c_i}^{d_i}. \tag{4}$$

SN_i generates two polynomials, $P_1(x)$ and $P_2(x)$, which are linear combinations of t challenged chunks. Note that $F_i=\{f_{i,1},\cdots f_{i,2n}\}$.

$$\begin{aligned} P_1(x)&=\sum_{j=1}^{t} d_j f_{c_j,1}\cdots+\sum_{j=1}^{t} d_j f_{c_j,n-1}\cdot x^{n-1},\\ P_2(x)&=\sum_{j=1}^{t} d_j f_{c_j,n+1}\cdots+\sum_{j=1}^{t} d_j f_{c_j,2n}\cdot x^{n-1}. \end{aligned} \tag{5}$$

Besides, computing quotient and remainder under the polynomial $(x-r_i)$, that is, $Q_i(x)=(P_i(x)-P_i(r_i))/x-r_i$, and we represent the coefficients vector of the quotient polynomial $Q_i(x)$ as $(w_{i,1}, w_{i2},\cdots,w_{i,n-2})$. In the end, SN_i produces:

$$\varphi=g_1^{Q_1(\sigma)}h_1^{Q_2(\sigma)}=\prod_{j=1}^{n-2}\left(g_1^{\alpha^j}\right)^{w_{1j}}\left(h_1^{\alpha^j}\right)^{w_{2j}}. \tag{6}$$

The form of proof is $\text{Proof}_i=\{\sigma,\varphi,P_1(r_i),P_2(r_i)\}$. Then, the node sends the Proof_i to the blockchain. After all nodes send proofs to the blockchain, every node verifies the correctness of other node's proofs through the following Equation (7). If each Proof_i is passed the verification equation, the node SN_i sends 1 to the smart contract; otherwise, 0. In the equation, χ, ϕ can, respectively, be acquired by calculating $\prod_{i=1}^{t} H(id\|c_i)^{d_i}$ and $g_1^{-P_1(r_i)}h_1^{-P_2(r_i)}$.

$$e(\sigma,g_2)e(\phi,\varepsilon)=e(\chi,\varepsilon)e(\varphi,\beta\cdot\varepsilon^{-r_i}). \tag{7}$$

The correctness of the above equation is proved as follows:

$$\begin{aligned} \text{LHS}&=e\left(g_1^{vP_1(\alpha)}h_1^{vP_2(\alpha)}\chi^v,g_2\right)e\left(g_1^{-P_1(r_i)}h_1^{-P_2(r_i)},g_2^v\right)\\ &=e\left(g_1^{(P_1(\alpha)-P_1(r_i))}h_1^{(P_2(\alpha)-P_2(r_i))},g_2^v\right)e(\chi^v,g_2)\\ &=e\left(g_1^{(\alpha-r_i)Q_1(\alpha)}h_1^{(\alpha-r_i)Q_2(\alpha)},g_2^v\right)e(\chi,g_2^v)=\text{RHS}. \end{aligned} \tag{8}$$

Remark. Compared with Su et al. [27] in the self-auditing stage, our scheme never needs interaction between SNs. At the same time, we use blockchain information to generate the challenge set for each SN. Blockchain assists SNs in completing each audit without DO online. Therefore, we reduce the number of interactions and some computational overhead between SNs and DO.

4.3. Batch Verification. Our scheme can support batch auditing for improving the efficiency of data audits via polynomial commitment aggregation [37]. Consequently, our scheme can verify multiple proofs at one time. Concretely, when the last proof is transferred to the blockchain, every SN generates a random number $r=H_2(\text{blockhash})$. Based on the proofs $\{\text{Proof}_i\}_{i=1}^{k}$ on the blockchain, every SN generates the aggregation of the proofs as shown in Algorithm 1.

Finally, each SN sends the audit results to the smart contract. The smart contract maintains a counter for all storage nodes denoted the number of successful audits. After archiving the data stored in the decentralized network, SNs are rewarded or punished according to the number of counter.

In the end, we show that Algorithm 1 can ensure the correctness of batch verification. There have k proofs to audit at one time and the correctness of batch verification is shown as follows:

$$\begin{aligned} \text{LHS}&=\prod_{i=1}^{k}\left(e(\sigma_i,Z_i)e\left(g_1^{-p_1(r_i)}h_1^{-p_2(r_i)},\widehat{Z}_i\right)\right)^{r^{i-1}}\\ &=\prod_{i=1}^{k}\left(e\left(g_1^{p_1(\alpha)-p_1(r_i)}h_1^{p_2(\alpha)-p_2(r_i)},\widehat{Z}_i\right)\right)^{r^{i-1}}\\ &=\prod_{i=1}^{k}e\left(g_1^{Q_1(\alpha)}h_1^{Q_2(\alpha)},Z_T\right)^{r^{i-1}}=\text{RHS}. \end{aligned} \tag{9}$$

5. Analysis of our Proposed Scheme

5.1. Security Analysis. In this section, we evaluate the security of our auditing scheme according to data privacy, storage correctness, and fairness listed in Section 3.2.

Theorem 1. *The scheme can guarantee that the correct proof can pass the verification, and the storage nodes cannot forge the authenticators and proofs when he does not maintain the entire data in the q-BSDH assumption.*

5.1.1. Data Privacy. We assume that the tools and cryptographic primitives used in our scheme are secure, such as the hash function, symmetric encryption algorithm, and polynomial commitment scheme. Therefore, we can ensure

Require:
The proofs shown on the blockchain:
$\{\text{Proof}_i = \sigma_i, \varphi_i, P_1(r_i), P_2(r_i)\}_{i=1}^{k}$;
The challenge r can compute by the information of blockchain, $r = H_2(\text{blockhash})$;
Ensure:
Result of integrity audit s;
1: Compute $\varphi = \prod_{i=1}^{k} \varphi_i^{r^{i-1}}$;
2: Compute $Z_i = g_2^{\Pi(\sigma - r_j)/(\sigma - r_i)}, \widehat{Z}_i = g_2^{\upsilon\Pi(\sigma - r_j)/(\sigma - r_i)}$;
3: Compute $Z_T = g_2^{\upsilon\Pi(\sigma - r_i)}$;
4: Compute A as: $\prod_{i=1}^{k} \left(e(\sigma_i, Z_i) e(g_1^{-p_1(r_i)} h^{-p_2(r_i)}, \widehat{Z}_i) e(\chi_i^{-1}, \widehat{Z}_i)\right)^{r^{i-1}}$
5: Compute $B = \mathrm{e}(\varphi, Z_T)$;
6: **If** $A = B$ **then**
7: Set $s = 1$;
8: **else**
9: Set $s = 0$;
10: **end if**
11: **return** s;

Algorithm 1: SN batch verification.

the privacy of the DO's data by using symmetric encryption algorithm before uploading data to the decentralized storage network.

5.1.2. Storage Correctness and Fairness. In Equation (8) and Equation (9), we first prove that the honest SN can always pass the integrity verification. Then, we give a proof sketch that the authenticator generated by DO and the proof generated by SN are unforgeable.

According to the description in [35], the Pedersen-based polynomial commitment scheme is security provided the t-SDH assumption in group $\mathbb{G}_1$. Therefore, if an existed probabilistic polynomial time adversary $\mathscr{A}$ can forge an authenticator, $\mathscr{A}$ can construct an algorithm to efficiently deal with the t-SDH problem. Specifically, assume that $\mathscr{A}$ can forge $f_1(x)$ and $g_1(x)$ such as $g^{f_1(\alpha)} h^{g_1(\alpha)} = g^{f_2(\alpha)} h^{g_2(\alpha)}$, where $f_1(x)$, $g_1(x)$, $f_2(x)$, and $g_2(x)$ are known to $\mathscr{A}$. $\mathscr{A}$ can construct polynomials $f(x) = f_1(x) - f_2(x)$ and $g(x) = g_1(x) - g_2(x)$ and gain $g^{f(x)} h^{g(x)}$. Therefore, to factor $f(x)$ and $g(x)$, $\mathscr{A}$ can achieve the private key α and break the t-SDH assumption in the system security parameters.

We show that malicious SN cannot generate valid proof if the entire data is never honestly stored. Given two valid proof responses $(\sigma_1, \varphi_1, P_1(r_1), P_2(r_1))$ and $(\sigma_2, \varphi_2, P_1(r_2), P_2(r_2))$. An adversary $\mathscr{A}$ is possible to extract the knowledge of linear combination of original data chunks with overwhelming probability unless $\mathscr{A}$ can deal with CDH and q-BSDH problems. Please refer to [3, 22, 39] for more details.

Moreover, the scheme generates a probabilistic proof of data possession like previous work. To guarantee a high confidence level is significant for DO. The probabilistic analysis given in [2] shows the relationship between the storage confidence level and the number of challenged chunks. Concretely, if 1% of data chunks have been tampered, only 300 chunks can give DO storage guarantee level of 95%. In the decentralized storage network, we think this number of challenges is sufficient to protect DO interest.

Remarks on Fairness. We further analyze the fairness in our scheme. First of all, we consider that the DO generates the incorrect metadata to gain more benefit. After the SN receives this erroneous metadata, it can implement the audit protocol locally in advance to ensure the correctness of the metadata with a tremendous probability. If the SN finds the wrong metadata, it can stop this transaction at negligible cost. Then, we consider that the malicious SN may generate wrong proof to gain more interest. The most honest nodes can send the malicious node's blockchain address to the smart contract when the audit fails. Then, the smart contract can transfer the deposit of the malicious node to other participants.

5.2. Comparison. As shown in Table 2, we compare four data auditing schemes, which consists of Yuan's scheme [22], Du's scheme [9], Su's scheme [27], and our scheme, in terms of audit mode, no interactions between SNs, data owner offline, decentralized storage, and batch auditing. Firstly, compared with Yuan's scheme, other schemes remove TPA from the traditional system model, which defends against single-point-of-failure and collusion between storage providers and TPA. Secondly, only Su's scheme has $4k$ interactions between SNs in the self-auditing stage because each node needs to obtain other nodes' proofs to complete each auditing task where the parameter k is the number of storage nodes.

Finally, Su's scheme and our scheme support decentralized storage and batch auditing, improving data recoverability and reducing the computation overhead for SNs.

Both Su's scheme and our scheme use self-auditing to remove TPA in traditional system model. However, our scheme uses blockchain information to generate the challenged set for each SN. Thus, data owners should never

Table 2: Comparison of data auditing schemes.

	Audit mode	No interactions between SNs	Data owner offline	Decentralized storage	Batch auditing
Scheme [22]	TPA	Yes	Yes	No	No
Scheme [9]	Smart contract	Yes	Yes	No	No
Scheme [27]	Self-auditing	No	No	Yes	Yes
Our scheme	Self-auditing	Yes	Yes	Yes	Yes

Table 3: Performance comparison.

	File information	Preprocess		Proof generation		Verification	Multiple proof verification
	Size	Time	Tag size	Time	Size	Time	Time
Scheme [9]	1 GB	86 s	13 MB	45 ms	288 byte	45 ms	88 ms
Our scheme	1 GB	82 s	6 MB	61 ms	320 byte	46 ms	51 ms

Note that we compare with scheme [9] in the parameter n to 150 and the size of challenged set t to 240.

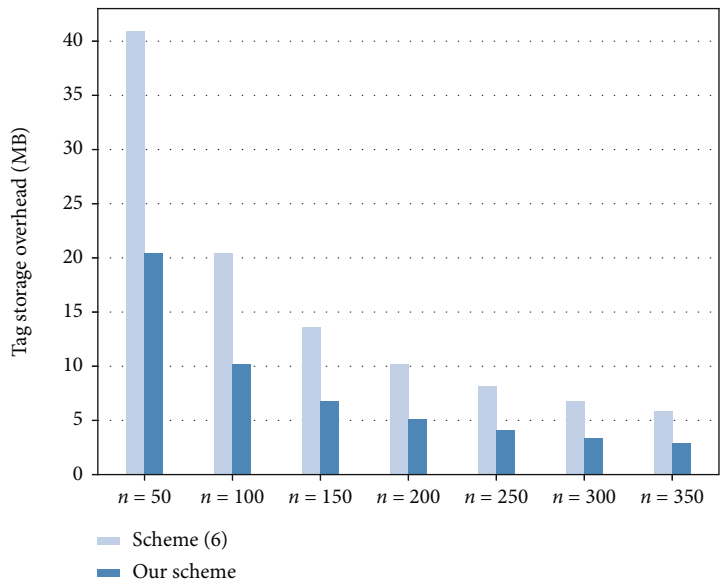


Figure 3: The size of authenticators of 1 GB data.

online to challenge each SN, and each SN should only interact with blockchain without amount communication overhead. However, Su's scheme should have $4k$ interactions between SNs which is unfeasible when k becomes larger.

6. Performance Evaluation

This section gives a performance evaluation of our auditing scheme in terms of off-chain storage and computational overhead and on-chain gas fee overhead. Besides, we mainly compare the experimental result with Du's scheme [9] which is the first auditing framework to consider on-chain privacy and efficiency.

6.1. Implementation and Experiment Setup. We leverage the Golang language to implement our off-chain scheme by the BN256 curve [40] and the KZG [41] library where the secure parameter is 256 bit. BN256 curve library implements the elliptic-curve-related operations with Golang language $((|p| = |\mathbb{G}_1| = 256\text{bit}, |\mathbb{G}_2| = 512\text{bit})$. KZG library implements addition, subtraction, multiplication, and division of polynomials with Golang language. On the part of the blockchain, we use the Ethereum test network Reposten and utilize the Solidity and Remix-IDE tool to employ our smart contract.

In order to simulate the decentralized storage network, we use three virtual machines as our storage nodes with Intel (R) Core (TM) i5-9500 CPU 3.00 GHz, 4 GB RAM and 20 GB SSD disks running on Ubuntu 18.04 LTS. The data owner uses a Desktop PC with windows 10 (AMD Ryzen5 4600 U CPU 2.1 GHz 16 GB RAM). Besides, we use the same configuration and parameters to implement the Du's scheme [9]. We set p as a 256 bit large prime number, the stored file size to 1 GB, and the number of challenged chunks from 240 to 440. The evaluation results are the average of 10 experiments.

6.2. Off-Chain Overhead. In the off-chain part, we test the authenticators' generation time, authenticators storage overhead, and proof generation time as shown in Table 3. To

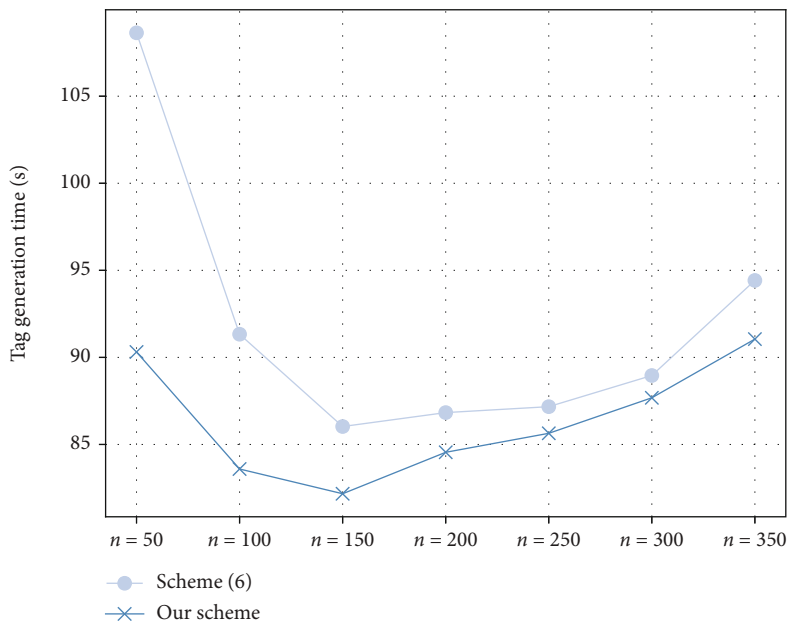


Figure 4: The computational time for DO to process 1 GB data.

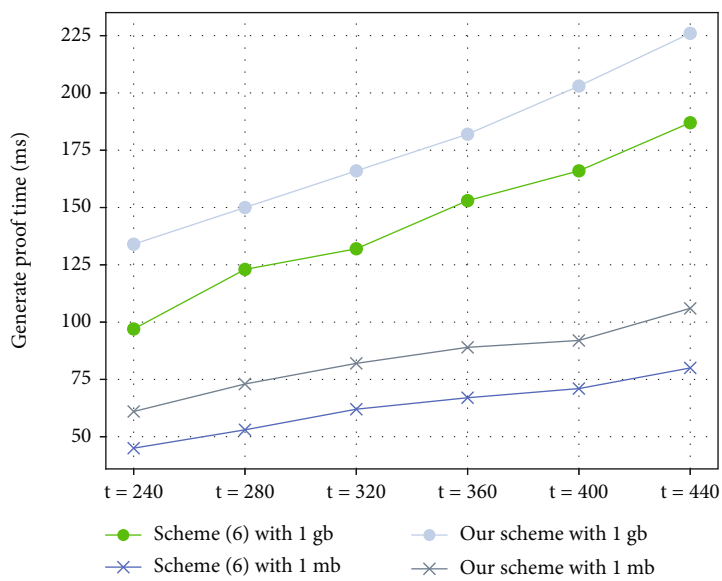


Figure 5: The proof generation time.

simulate data owner in the preprocessing stage, we use four Golang coroutines to process a 1 GB file size with the parameter n from 50 to 350. Note that our authenticators' generation time includes other factors such as key pairs' generation, I/O overhead, and polynomial coefficient transformation of data chunks.

The parameter n is negatively correlated with the number of chunks and authenticators. Thus, as illustrated in Figure 3, we greatly decrease the number of authenticators which is always half of Du's scheme [9] due to the use of Pedersen-based polynomial commitment. As shown in Figure 4, we slightly decrease the authenticators' generation time due to less frequency to access the original file. Therefore, we spend less time on I/O overhead with the increasing of parameter n compared with Du's scheme. However, $n-1$ is the order of two committed polynomials in the TagGen stage. The computational overhead for the polynomials becomes more expensive with the parameter n increases. We can see that the case of $n = 150$ is the least time for generating the auditing authenticators.

Then, we test the overhead of proof generation related to the number of the challenged chunks from 240 to 440. In the proof generation phase, our scheme needs to deal with an extrapolynomial with the order of $n-1$. Therefore, as shown

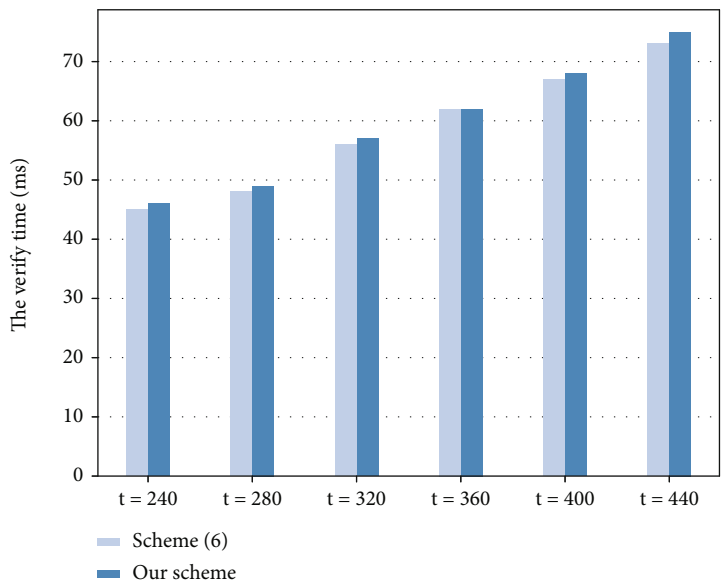


Figure 6: The single verification time in the parameter $n = 150$.

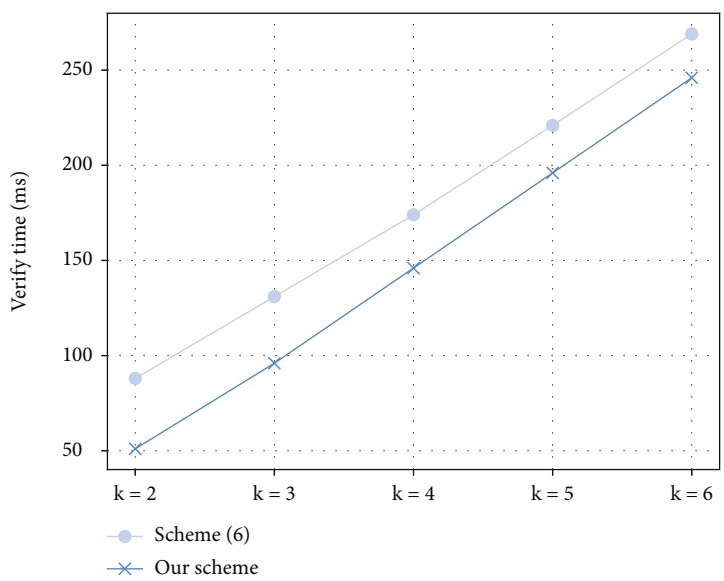


Figure 7: Multiple proof verification time.

Table 4: Gas consumption comparison (million).

	PK storage	Chal-generation	Proof storage	Proof verification
Scheme [9]	0.16	0.07	0.029	0.6
Our scheme	0.30	0	0.031	0

Note that we compare the gas consumption with scheme [9] in the parameter n to 50 and the size of challenged set t to 300.

in Figure 5, compared with Du's scheme, our scheme is about 20-30% slower in generating proof. Moreover, we can discover that the size of file dramatically affects the proof generation time, mainly dominated by the I/O cost. Due to the smaller authenticators' redundancy, our scheme has less I/O overhead. In the case of 1 MB, which can ignore the cost of I/O, we are about 30% slower than the comparison scheme. And for the case of 1 GB, we are about 20% slower than the comparison scheme. Thus, our scheme has the advantage of processing more larger file.

Then, setting the parameter n to 150, we evaluate the verification time by increasing the challenged chunks from 240 to 440. As shown in Figure 6, the time consumption increases linearly with the size of the challenged chunks. Compared with Du's scheme, our scheme requires additional overhead to calculate the multiplication and addition on elliptic curve $\mathbb{G}_1$, but this time can be ignored. As shown in the case of chunks of 240, the verification time of the two schemes is almost equal, about 45 ms.

Lastly, we test the time cost of batch verification with the increasing number of storage nodes. We set the number of blocks to 150, challenged chunks to 240, and storage nodes from 2 to 7. As shown in Figure 7, the test results of our batch verification are compared with Du's k-time single verification. In theory, we can decrease $k-1$ bilinear pairing operations, and other calculation overhead is unchanged when k is small. It can be seen from the figure that our batch verification scheme can improve 30% verification efficiency.

6.3. On-Chain Overhead. In the decentralized system model based on blockchain, the on-chain cost is concerned by the participants. We use a smart contract to manage all participants who use our scheme. All participants only need to interact with this smart contract rather than create their own smart contracts. The proofs and counters of SNs and the public keys of DOs are all stored in this smart contract. We use the Ethereum test network, Reposten, to deploy our smart contract written by Solidity. There are two main function in our smart contract, storage and auditCount.

As shown in Table 4, we compare the on-chain gas overhead in terms of the storage of public key and proofs, challenge generation, and proof verification.

After the setup phase, our scheme should send PK to the blockchain. Note that the size of a $\mathbb{G}_1$ element is 64 bytes and $\mathbb{G}_2$ is 128 bytes. To save gas consumption, we only send x -axis coordinates on the elliptic curve to the blockchain, and more 1 bit records the positive or negative information of the elliptic curve points. Through computing the y-axis coordinates on the off-chain, SNs can obtain public keys. Compared with Du' scheme, our scheme only adds $n-2$ group elements of $\mathbb{G}_1$ and $2k$ elements on $\mathbb{G}_2$ groups to improve the size of the authenticator and introduce the batch auditing. Thus, the consumption of PK storage of our scheme is 0.3 million, which is nearly a double increase compared with Du's scheme. However, note that this process is one-time storage cost for the whole auditing duration. Data owners can use these public keys in the future. Moreover, we compare the storage overhead of the proof in the blockchain. Compared with Du's scheme, our proof only has an extra 256 bit large number $P_2(r_i)$. We increase about 2000 gas consumption in each audit for the test in the Reposten.

However, compared with Du's scheme in the challenge generation and proof verification phase, our scheme never requires any overhead on the blockchain. We are only required to maintain a counter to record the number of successful audits without the amount of computation in the smart contract. To sum up, our scheme saves approximately 60% gas consumption.

7. Conclusion

In this paper, we proposed a blockchain-based self-auditing scheme with batch verification in a decentralized framework. Firstly, different from previous works, our scheme removes TPA through the interaction between storage nodes and blockchain to achieve self-auditing. The recoverability of data can be guaranteed due to the distribution to storage nodes. Secondly, using the Pedersen-based polynomial commitment to generate the authenticators, our scheme decreases the computational overhead for DO and the storage overhead for SNs. Moreover, the batch verification algorithm improves the verification efficiency by aggregating multiple polynomials and points. Lastly, security analysis and experiments show that our scheme achieves the security goals and is efficient and feasible to deploy in the practice blockchain environment.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the National Nature Science Foundation of China (no. 62072357), the Key Research and Development Program of Shaanxi (nos. 2022KWZ-01 and 2020ZDLGY08-03), the Shandong Provincial Key Research and Development Program of China (no. 2019JZZY020129), and the Fundamental Research Funds for the Central Universities (nos. JB211503 and YJS2212).

References

[1] J. Ning, X. Huang, W. Susilo, K. Liang, X. Liu, and Y. Zhang, "Dual access control for cloud-based data storage and sharing," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 2, pp. 1036–1048, 2022.

[2] G. Ateniese, R. Burns, R. Curtmola et al., "Provable data possession at untrusted stores," in *Proceedings of the 14th ACM conference on Computer and communications security*, pp. 598–609, Alexandria Virginia USA, 2007.

[3] H. Shacham and B. Waters, "Compact proofs of retrievability," in *International Conference on the Theory and Application of Cryptology and Information Security*, pp. 90–107, Springer, 2008.

[4] C. Wang, S. S. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for secure cloud storage," *IEEE Transactions on Computers*, vol. 62, no. 2, pp. 362–375, 2013.

[5] ethereum, "The golang version of etherum client," 2021, https://github.com/ethereum/go-ethereum.

[6] A. Yakovenko, "Solana: a blockchain network focused on fast transactions and high throughput," 2019, https://solana.com.

[7] H. Yuan, X. Chen, J. Wang, J. Yuan, H. Yan, and W. Susilo, "Blockchain-based public auditing and secure deduplication with fair arbitration," *Information Sciences*, vol. 541, pp. 409–425, 2020.

[8] H. Wang, H. Qin, M. Zhao, X. Wei, H. Shen, and W. Susilo, "Blockchain-based fair payment smart contract for public cloud storage auditing," *Information Sciences*, vol. 519, pp. 348–362, 2020.

[9] Y. Du, H. Duan, A. Zhou, C. Wang, M. H. Au, and Q. Wang, "Towards privacy-assured and lightweight on-chain auditing of decentralized storage," in *2020 IEEE 40th International*

Conference on Distributed Computing Systems (ICDCS), pp. 201–211, Singapore, Singapore, 2020.

[10] A. Juels and B. S. Kaliski Jr., "Pors: proofs of retrievability for large files," in *Proceedings of the 14th ACM conference on Computer and communications security*, pp. 584–597, Alexandria Virginia USA, 2007.

[11] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the Weil pairing," in *International Conference on the Theory and Application of Cryptology and information Security*, pp. 514–532, Springer, 2001.

[12] R. Curtmola, O. Khan, R. Burns, and G. Ateniese, "Mr-pdp: multiplereplica provable data possession," in *2008 the 28th international conference on distributed computing systems*, pp. 411–420, Beijing, China, 2008.

[13] H. Kai, H. Chuanhe, W. Jinhai et al., "An efficient public batch auditing protocol for data security in multi-cloud storage," in *2013 8th China Grid Annual Conference*, pp. 51–56, Los Alamitos, CA, USA, 2013.

[14] J. Chang, B. Shao, Y. Ji, and G. Bian, "Efficient identity-based provable multi-copy data possession in multi-cloud storage, revisited," *IEEE Communications Letters*, vol. 24, no. 12, pp. 2723–2727, 2020.

[15] I. Damgård, C. Ganesh, and C. Orlandi, "Proofs of replicated storage without timing assumptions," in *Annual International Cryptology Conference*, pp. 355–380, Springer, 2019.

[16] F. Armknecht, L. Barman, J.-M. Bohli, and G. O. Karame, "Mirror: enabling proofs of data replication and retrievability in the cloud," in *25th USENIX security symposium (USENIX security 16)*, pp. 1051–1068, Austin, TX, 2016.

[17] I. Leontiadis and R. Curtmola, "Secure storage with replication and transparent deduplication," in *Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy*, pp. 13–23, Tempe AZ USA, 2018.

[18] J. Ning, J. Chen, K. Liang, J. K. Liu, C. Su, and Q. Wu, "Efficient encrypted data search with expressive queries and flexible update," *IEEE Transactions on Services Computing*, vol. 15, no. 3, pp. 1619–1633, 2020.

[19] A. Yang, J. Xu, J. Weng, J. Zhou, and D. S. Wong, "Lightweight and privacy-preserving delegatable proofs of storage with data dynamics in cloud storage," *IEEE Transactions on Cloud Computing*, vol. 9, no. 1, pp. 212–225, 2021.

[20] X. Zhang, J. Zhao, C. Xu, H. Li, H. Wang, and Y. Zhang, "CIPPPA: conditional identity privacy-preserving public auditing for cloud-based wbans against malicious auditors," *IEEE Transactions on Cloud Computing*, vol. 9, no. 4, pp. 1362–1375, 2021.

[21] J. Wang, X. Chen, X. Huang, I. You, and Y. Xiang, "Verifiable auditing for outsourced database in cloud computing," *IEEE Transactions on Computers*, vol. 64, no. 11, pp. 3293–3303, 2015.

[22] J. Yuan and S. Yu, "Proofs of retrievability with public verifiability and constant communication cost in cloud," in *Proceedings of the 2013 international workshop on Security in cloud computing*, pp. 19–26, Hangzhou China, 2013.

[23] H. Wang, Q. Wang, and D. He, "Blockchain-based private provable data possession," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 5, pp. 2379–2389, 2021.

[24] Y. Zhang, C. Xu, X. Lin, and X. Shen, "Blockchain-based public integrity verification for cloud storage against procrastinating auditors," *IEEE Transactions on Cloud Computing*, vol. 9, no. 3, pp. 923–937, 2021.

[25] Y. Xu, J. Ren, Y. Zhang, C. Zhang, B. Shen, and Y. Zhang, "Blockchain empowered arbitrable data auditing scheme for network storage as a service," *IEEE Transactions on Services Computing*, vol. 13, no. 2, pp. 289–300, 2020.

[26] Y. Xu, C. Zhang, G. Wang, Z. Qin, and Q. Zeng, "A blockchain-enabled deduplicatable data auditing mechanism for network storage services," *IEEE Transactions on Emerging Topics in Computing*, vol. 9, no. 3, pp. 1421–1432, 2021.

[27] Y. Su, Y. Li, B. Yang, and Y. Ding, "Decentralized self-auditing scheme with errors localization for multi-cloud storage," *IEEE Transactions on Dependable and Secure Computing*, p. 1, 2021.

[28] Y. Li, Y. Yu, R. Chen, X. Du, and M. Guizani, "IntegrityChain: provable data possession for decentralized storage," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 6, pp. 1205–1217, 2020.

[29] D. Francati, G. Ateniese, A. Faye et al., "Audita: a blockchain-based auditing framework for off-chain storage," in *Proceedings of the Ninth International Workshop on Security in Blockchain and Cloud Computing*, pp. 5–10, Virtual Event Hong Kong, 2021.

[30] D. Chen, H. Yuan, S. Hu, Q. Wang, and C. Wang, "Bossa: a decentralized system for proofs of data retrievability and replication," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 4, pp. 786–798, 2021.

[31] S. fund, "Swarm: a decentralised storage and communication system for a sovereign digital society," https://www.ethswarm.org.

[32] S. Labs, "A decentralized cloud storage network framework," 2018, https://www.storj.io.

[33] D. Vorick and L. Champine, "Sia: simple decentralized storage," *Retrieved May*, vol. 8, p. 2018, 2014.

[34] P. Labs, "Filecoin: a decentralized storage network," 2017, https://filecoin.io/.

[35] A. Kate, G. M. Zaverucha, and I. Goldberg, "Constant-size commitments to polynomials and their applications," in *International Conference on the Theory and Application of Cryptology and Information Security*, pp. 177–194, Springer, 2010.

[36] T. P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Annual International Cryptology Conference*, pp. 129–140, Springer, 1992.

[37] D. Boneh, J. Drake, B. Fisch, and A. Gabizon, *Efficient Polynomial Commitment Schemes for Multiple Points and Polynomials*, Cryptology ePrint Archive, 2020.

[38] J. Bloemer, M. Kalfane, R. Karp, M. Karpinski, M. Luby, and D. Zuckerman, *An Xor-Based Erasure-Resilient Coding Scheme*, International Computer Science Institute (ICSI) Technical Report, 1995.

[39] D. Boneh and X. Boyen, "Short signatures without random oracles," in *International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 56–73, Springer, 2004.

[40] Cloudflare, "Package bn256 implements a particular bilinear group," 2019, https://github.com/ethereum/go-ethereum/tree/master/crypto/bn256/cloudflare.

[41] Arnaucube, "The kzg polynomials commitment," 2021, https://github.com/arnaucube/kzg-commitments-study.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 4618007, 18 pages
https://doi.org/10.1155/2022/4618007

Research Article

The Applications of Blockchain in the Covert Communication

Bangyao Du,[1,2] Debiao He,[3,4,5] Min Luo,[4,6] Cong Peng,[4,7] and Qi Feng[4]

[1]*Key Laboratory of Aerospace Information Security and Trusted Computing Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan 430072, China*
[2]*Hangzhou Innovation Institute, Beihang University, Hangzhou 310052, China*
[3]*Shandong Provincial Key Laboratory of Computer Networks, Qilu University of Technology (Shandong Academy of Sciences), Jinan 250014, China*
[4]*School of Cyber Science and Engineering, Wuhan University, Wuhan 430072, China*
[5]*Institute of Network System and Security, Peng Cheng Laboratory, Shenzhen 518000, China*
[6]*Guangxi Key Laboratory of Trusted Software, Guilin University of Electronic Technology, Guilin 541004, China*
[7]*Shanghai Key Laboratory of Privacy-Preserving Computation, MatrixElements Technologies, Shanghai 201204, China*

Correspondence should be addressed to Debiao He; hedebiao@163.com and Min Luo; mluo@whu.edu.cn

Received 11 January 2022; Revised 3 March 2022; Accepted 5 May 2022; Published 14 June 2022

Academic Editor: Antonio Guerrieri

Covert communication is designed for hiding the subliminal communication which takes place between both of the speakers and their relationship. The traditional covert communication utilizes the centralized channel and the third-party central node or authority to distribute messages which leads to a lack of undetectability, antitraceability, and robustness. In recent years, there have been attempts to apply the blockchain to covert communication solutions, for the characteristics of blockchain such as decentralization, openness, and trustworthiness. In this paper, based on the analysis of the literature and the classification according to the hiding position, we identify four kinds of covert communication: Address Channel, Value Channel, DSA (Digital Signature Algorithm) Channel, and Script Channel, which will help inform future research agenda.

1. Introduction

With the flourishing development of the Internet and computer science, network users communicate more and more on the Internet. The increasingly frequent interactions attract universal attention to the safety problem of information on the network, especially for some important sensitive information. So there is an expectation that covert communication can be realized to protect the safety, integrity, and secrecy in the process of information exchange.

The design goal of covert communication is to hide the relationship and the actual information exchange between the sender and the receiver beneath the superficial phenomenon. Where there is covert communication, there usually exists a subliminal channel. In 1983, Simmons [1] put forward the concept of the covert channel when solving the prisoners' problem that two prisoners want to construct a covert channel to plan an escape in an apparently normal conversation. A covert channel refers to a secret channel established in the digital signature, authentication, and other cryptosystems based on the public key infrastructure. Except for the sender and the designated receiver, no one knows whether there exists covert information in the transmitted cryptographic data content.

Traditional covert communication is served by a centralized channel generally, which results in the vulnerability to the interface of environment and other factors in the process. By monitoring the network and measuring the traffic, a covert channel can be distinguished by the attacker. Once the covert channel is detected by the attacker, participants are under threat of being exposed. In addition, the centralized nodes and devices are too weak to survive the attack, and the worst case may lead to the paralysis of the communication system.

Furthermore, the covert message is usually delivered directly to the receiver. It is sufficiently challenging to find a solution for group covert communication under the circumstance of the traditional method.

As a break of the mainstream mechanism that relies on the third parties in information and trade exchanges, the blockchain [2] is a new technique developing rapidly recently. The blockchain has the characteristics of decentralization, nontampering, nonforgery, openness, and security, which draw the attention of all professions and trades. Not only financial services but also almost every electronic services start exploring the benefits of using the blockchain in their infrastructure. With the entry of the blockchain, new protocols, new applications, and new solutions are published, such as smart contracts, proof of existing services, and covert channels.

For example, the facilities of Bitcoin like Mastercoin [3] and Colored Coins [4] are used in many projects to provide alternative currencies and other financial instruments such as stocks and bonds. All these applications profit from the ability of Bitcoin transactions where additional information can be embedded, so that many scholars manage to apply the blockchain to construct subliminal channels in the field of covert communication.

Compared with traditional covert communication, the applications of the blockchain in covert communication can be highly advantageous. To start with, nodes of a peer-to-peer network in the blockchain flood transactions and blocks. The identity of the receiver will be kept hidden in this mode of communication because there has no specific destination. Under this circumstance, it is more likely to achieve group communication. For another thing, the popularity of Bitcoin continues to grow as shown in Figure 1, so are other cryptocurrencies in the blockchain. As of February 2022, more than 81 million people had created unique Bitcoin wallets on http://Blockchain.com which makes purchasing Bitcoin possible. A huge amount of members lead to tremendous transactions, which provides benefits for covert communication. Thirdly, the blockchain is free of any government or organizational control. Users can send transactions anywhere in the world without banking infrastructure or exchange fees, which fosters the applications of blockchain in covert communication.

1.1. Our Contributions. In this paper we make the following contributions:

(i) We present researches about covert communication with and without blockchain. We concentrate on the blockchain-based covert communication and present some typical work in detail

(ii) According to the secret hiding place, we divide the subliminal channels used in the covert communication schemes on the blockchain into four categories: Address Channel, Value Channel, DSA Channel, and Script Channel. Their definitions and how to embed a secret in each kind of channel are given in Section 5

(iii) We perform an analysis of the proposed channels from three aspects: capacity, concealment, and efficiency. The summary of the related works is analyzed and some suggestions for further research are given in the end

1.2. Organization of This Paper. The remaining of this paper is structured as below. In Section 2, traditional covert communication schemes are briefly explained. In Section 3, we introduce the relevant background materials. The review of some typical schemes is presented in Section 4; then, we classify the existing schemes according to their hiding location and analyze them in Section 5. We discuss the weaknesses of existing protocols and some research directions in Section 6. Finally, we make a conclusion of this paper in Section 7.

2. Traditional Covert Communication

As an application of information hiding technology, covert communication adopts unconventional patterns to break through restrictions of a message and employs steganography or coding permutation as a carrier to transmit the message secretly.

The blockchain appeared in the military field [5] for the first time. Images, audios, and videos are used to build subliminal channels to exchange military information while avoiding the enemy's monitoring. At the military communication conference in 2010, Hijaz and Frost [6] studied the potential of covert communication within an orthogonal frequency division multiplexing (OFDM) waveform and realized secret communication by inserting a covert narrowband signal in a new subcarrier location of OFDM signal. Harvey et al. [7] emphasized the special requirements of specific military local area networks and talked about how higher band millimeter-wave technology can help to achieve high data rate and concealment at the same time.

As the Internet develops, covert communication has been extended to many different fields in which cryptography is taken as a carrier. As public-key cryptography develops, many covert channels are proposed such as DSA-based [8], RSA-based [9, 10], and ECDSA-based [11]. Nevertheless, due to the bandwidth limitation, only a little message can be embedded in these schemes. Hartl et al. [12] proved the existence of a broadband covert channel in the EdDSA signature scheme. In 2001, Rivest et al. [13] formalized the notion of a ring signature and proposed a provably secure ring signature scheme. In 2012, Dong et al. [14] proposed an anonymous covert channel based on RST (Rivest, Shamir, and Taumann) ring signature to keep the signer himself covert anonymous to the covert receiver. In 2019, Wang et al. [15] proposed covert channels in the code-based ring signature scheme in which the designated receiver did not need to know the private key of the signer.

Based on different data carriers, there are three fundamental kinds of covert channels [16] called covert timing channels (CTCs), covert storage channels (CSCs), and covert network channels (CNCs). CTCs mainly use time stamp, the transmission time interval of data packet [17], and

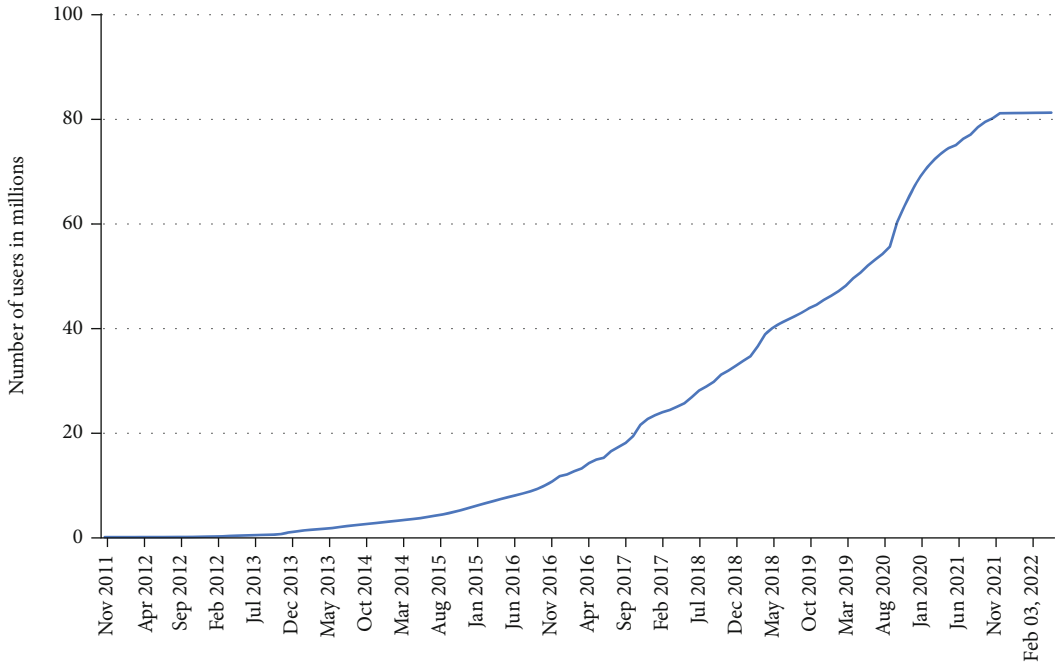


Figure 1: Number of blockchain wallet users worldwide from November 2011 to February 2022.

transmission quantity of data packet in unit time [18] to realize information transmission. Easy to be affected by network delay and jitter, CTCs have weak robustness [19]. And CSCs employ system variables or attributes except for time such as image [20, 21], protocol [22], and biological signal [23] to represent covert information. Different from CTCs, CSCs are not easy to be disturbed by the network environment and have better robustness, higher information embedding rate, and more types of carriers. CNCs include variations of CTCs, CSCs, and the properties of the network over which they operate [24, 25].

3. Preliminaries

In this section, we briefly review covert communication channel and the blockchain technology.

3.1. Covert Communication Channel. As a supplement to the traditional encryption technology, information hiding embeds information covertly into the media or carrier so as to transmit secret information without being noticed. At first, it only used static media like images as carriers to hide information. But in recent years, as communication technology and network media develop, the widespread adoption of information hiding in the field of communication has gradually developed into an emerging technology—covert communication channel.

Unlike the traditional encryption technology to hide the content of the message itself, covert communication is aimed at hiding the existence of a covert message. The information is transmitted secretly and the redundant part of the carrier serves to insert the processed information into communication and media messages. The model of a traditional covert communication channel with an omniscient knowledge of the covert channel is shown in Figure 2.

It is obvious that the covert/overt sender and receiver may not be the same node. So there are some requirements of the established channel to be covert and undetectable by the adversary.

(i) Subsurface; a covert channel ought to be hidden under an overt channel whose operation is not controlled by the adversary. If the overt channel gets closed, the covert channel will no longer exist.

(ii) Nonintervention and reasonable: there exist overt users, while covert users are using the communication channel. So users in the overt channel cannot be bothered or suspected, which requires the establishment of the covert channel to bring no damage to the existing channel.

(iii) Undifferentiated: in the communication channel, the covert data is supposed to be the same as the overt data. It is expected that the observer is not able to find out the difference between covert data and legitimate overt data.

3.2. Blockchain. As a data structure, blockchain consists of many blocks linked in the order of time from back to front which contain transactions submitted to it. The blockchain is also a tamper-resistant unforgeable distributed and decentralized ledger that can store sequential data which can be verified in the system. The addition of a consensus mechanism turns the blockchain into a trusted network. Block can be regarded as a container that aggregates inner transactions' information, whose structure is shown in Figure 3. TX refers to a transaction. It is composed of a block header

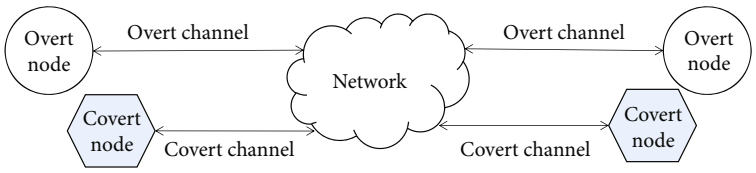


Figure 2: The model of traditional covert communication channel.

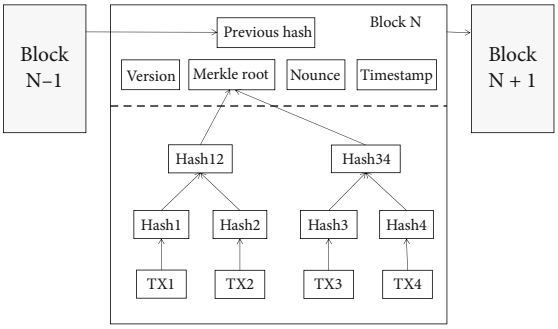


Figure 3: The structure of block.

followed by a series of transactions, which form the main body of the block. It has the following advantages.

3.2.1. Transparency. The openness and transparency of cryptography technology and data guarantee the security of the blockchain system. Anybody is able to join the blockchain as he wishes. Users can better supervise the data in the system and prevent dishonesty, denial, and disputes from happening. Cryptography guarantees that the exchanged message keeps privacy.

3.2.2. Decentralization. There are no intermediaries and trusted third parties in the blockchain. The recording, verification, restoration, and transmission of the blockchain data are all based on the distributed system architecture, rather than the traditional central structure. The whole network will not collapse due to the failure of several nodes or the attack on the central node, and the data will not be leaked due to the trust issues of the central node. So the blockchain has better fault tolerance and antiattack ability.

3.2.3. Traceability. Blockchain shows up as chain-like blocks with a timestamp attached, thus adding the time dimension and making it traceable and verifiable. Each node can track the changes in assets and transaction behavior according to the input and output of the transaction, which ensures the authenticity of data.

3.2.4. Immutability. Due to the special linked structure of blockchain, adjacent blocks keep in touch by means of hash values. Besides, the Merkle tree of each block guarantees that the transaction information cannot be distorted. Once the value of Merkle root is modified, the nonce in that block header is no longer legitimate. If an attacker wants to modify values in an existing block, he will have to modify the corresponding values in the latter block until the end of the chain to meet the needs of validation, which requires such tremendous computing power that it cannot be achieved within the time limit of consensus mechanism.

3.2.5. Collective Maintenance. A specific mechanism is adopted to make sure that all the nodes in the system take part in the validation procedure of data blocks during which a particular node is chosen through a consensus algorithm to append new blocks to the blockchain. All nodes jointly maintain the network while keeping their local ledger.

3.2.6. Anonymity. In the blockchain system, the addresses and accounts are generated by the user himself as long as they are legitimate. From an observer's point of view, the identity of the participants cannot be obtained directly from the transaction information.

3.3. Types of Cryptocurrency. On the basis of the open-resource blockchain technology, any developer has the access to the original source code which contributes to the upper creation. It should be noted that cryptocurrencies such as Bitcoin are built on the blockchain rather than take the place of it. Blockchain is merely used as a way to record purchases, payment information, etc. As is shown in Figure 4, it is estimated by http://statista.com that there are more than 10000 transactions as of the writing.

Although cryptocurrencies are virtual currencies, they can be traded or invested like any other real currencies and are particularly independent of banks and governments. The "crypto" part in cryptocurrencies indicates complex cryptographic algorithms used to deal with the processing of digital currencies and their exchange across decentralized users. There is not one "best" cryptocurrency because each developer's design is focused on a certain point to solve an existed problem.

Here is an overview of some of the most popular digital coins and how each is being used. Table 1 provides the methods of signature used in mainstream cryptocurrencies and their pros and cons. The circulating supply of each cryptocurrency is updated on Feb. 28, 2022. Type refers to whether the cryptocurrency is based on the UTXO model or not. The Signing Alg shows what kind of signing algorithm is adopted, and the elliptic curve used is shown in the column Curve.

Regarded as the first blockchain-based cryptocurrency launched by Satoshi in 2009, Bitcoin is the most popular and the biggest by market capitalization with the elimination of trusted third parties. Ether plays the role of the token on Ethereum to facilitate transactions. Designed on the basis of RippleNet, a digital payment platform, XRP was planned for financial institutions. In Litecoin, central processing

units (CPUs) can help the decoding process. Zero-knowledge proofs (zk-SNARKs) provide users in ZCash absolute privacy with no information leaked during validation. Taking advantage of the ring signature, Monero is able to hide the complete privacy of the sender. Unlike serving as a store of trading, Dash is devoted to becoming a real-life form of payment. In addition to payments, Lumens (XLM) can be used to fight spam.

3.4. Ethereum. Forbes mentioned that Ethereum is the first generic blockchain platform that allows users to create and deploy their decentralized and trustless applications easily. It has created incredible opportunities in the FinTech space. This section will introduce what Ethereum is and the Whisper protocol therein.

Ethereum is a decentralized application platform on the strength of blockchain technology. It allows users to establish and have distributed applications running on this platform in a decentralized manner. This means that applications running on Ethereum are available anywhere and anytime.

Compared with Bitcoin, Ethereum uses the account model rather than UTXO. This brings some advantages. Each transaction in Ethereum has only one input, one output, and one signature, thus saving much space and making it easier to understand. And the simple coding ability is required while there is no need to write complex scripts.

In the Whisper protocol, based on blockchain, Ethereum provides context both for the distributed applications (DAPP) and for the developers. In a DAPP where participants manage to reach an agreement, one-on-one communication between them is of great importance. That also accounts for the reason why Whisper takes a significant part under the circumstance of Ethereum [26]. Serving as the "decentralized communicate" component, Whisper works on a peer-to-peer framework with no server participating throughout the whole process and allows nodes to conceal the correlative information from unrelated parties while securely interacting with each other.

All the messages broadcast on the public network are routed according to the topic specified as shown in Figure 5 until they reach their destinations. Since there is a direct correspondence between every topic and the key to encrypt/decrypt data, that is to say, the covert message is encrypted with the public key of the recipient and is spread afterward. Nodes can apply for interested topics. Then, only messages with these topics will be received, the others will be abandoned.

The envelope acts as the basic data transfer unit in the Whisper protocol [27], whose structure is shown in Figure 6. There are two components in the envelope: enciphered data body and unencrypted metadata utilized for verification and data decryption in the envelope.

RLP (Recursive Length Prefix) encoding format is used while transmitting envelope.

(i) Version: 4 bytes at most (only 1 byte is being used now). During the transmission, if the envelope has a higher version than the node gets, it will not be decrypted and be forwarded only

(ii) Expiry: 4 bytes (UNIX timestamp in seconds) represents expiration time

(iii) TTL: 4 bytes, remaining survival time of the envelope in seconds

(iv) Topic: 4 bytes. The envelope is designed to have only one topic each. Besides, there is a consistent one-to-one match between each topic and the key required in the process of encryption and decryption

(v) AESNounce: 12-byte random digit, valid merely in symmetric encryption

(vi) Data: encrypted data body, whose role is to store message. It mainly consists of two parts. One is payload where the genuine information is located and is often enciphered in advance. The other is padding used to lower the risk of information exposure through the length

(vii) EnvNonce: 8-byte arbitrary data (for PoW calculation)

(viii) PoW: The aim of exploiting PoW is to reduce the quantity of junk mails and lighten network loads

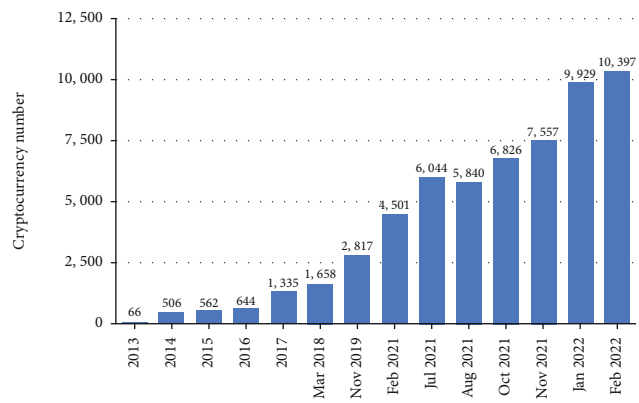


Figure 4: Number of cryptocurrencies worldwide from 2013 to February 2022.

The default rule to set data in version 5 is to keep the data length at multiples of 256 bytes. When obtaining an envelope whose topic is acknowledged, it will be decrypted by the appropriate key to gain the real information. Obviously, the identity of the sender in Whisper cannot be traced, let alone the location of the sender.

In the Whisper protocol, after each peer is connected successfully, two goroutines are generated to receive and broadcast messages.

3.5. Smart Contract. The concept of smart contract was first put forward in 1994. Smart contract was aimed at satisfying general needs, minimizing baleful and abnormal conditions, and lowering dependency on trusted intermediaries. The

Table 1: Mainstream digital currencies on blockchain (updated Feb. 28, 2022).

Name	Symbol	Type	Signing Alg	Curve	Pros	Cons	Circulating supply
Bitcoin	BTC	UTXO	ECDSA	secp256k1	Independence from central authorities; user anonymity and transparency.	No government regulations; limited use.	18,970,150 BTC
Ethereum	ETH	account	ECDSA	secp256k1	Second-biggest cryptocurrency; fast transaction speed.	Uncapped supply leads to inflation.	119,761,811 ETH
Ripple	XRP	account	ECDSA	secp256k1	Lightning fast transaction speed; cheap.	Less secure consensus protocol.	47,949,281,138 XRP
Litecoin	LTC	UTXO	ECDSA	secp256k1	Faster confirmation; cheap transaction fee.	Low market captalisation.	69,734,906 LTC
Zcash	ZEC	UTXO	ECDSA, zk-SNARKs	secp256k1	Prominent level of anonymity; fungible and interchangeable.	Restricted to CPU mining.	13,841,481 ZEC
Monero	XMR	UTXO	ECC, MLSAG	ed25519	Block limit flexibility; well security.	No mobile wallet; weak scalability.	18,085,509 XMR
Dash	DASH	UTXO	ECDSA	secp256k1	Faster confirmation speed; lower transaction fee.	Theoretical traceability.	10,603,264 DASH
Stellar	XLM	account	EdDSA	ed25519	Integrates with banks.	Not widely recognized.	24,943,914,340 XLM

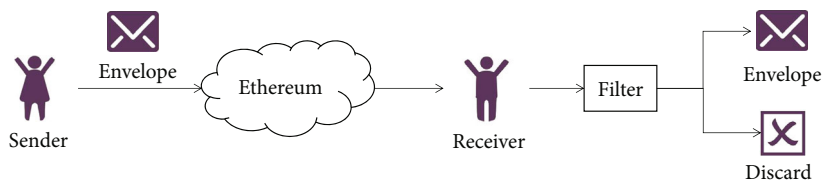


Figure 5: Message transmission in Whisper.

smart contract has been applied in many fields such as identity management [28], IoT [29], and medical privacy [30].

The smart contract makes the blockchain-based applications more convenient and expandable. While the contract is written into the blockchain in a digital form, the written data cannot be deleted but can be added or altered as a result of the characteristics of the blockchain. The whole process is transparent and trackable to ensure historical traceability. Because the behavior will be permanently recorded, the interference of malicious behavior on the normal execution of the contract can be avoided to a great extent. With the influence of centralization factors avoided, the cost efficiency of smart contract improves a lot.

4. Review of Blockchain-Based Covert Communication Protocols

Due to the immutable chain structure and distributed storage schema, a secure channel can be offered by blockchain. It requires extremely high cost that nearly nobody can afford to alter or fabricate the historical blocks. Because the blockchain network is on the basis of the peer-to-peer network, no third-party provider is needed while establishing the communication channel and delivering messages on it, thus making the blockchain-based covert communication protocols invulnerable to any availability attacks.

Version	Expiry	TTL
Topic	AESNounce	
Data	EnvNounce	
PoW		

Figure 6: The structure of envelope.

We investigated a number of recent papers related to blockchain-based covert communication protocols. Here, we make brief illustrations of some typical schemes by supposing a scene in which Alice, the sender, tries to dispatch a message denoted by M secretly to Bob, the receiver.

4.1. BLOCCE. Partala [31] provided an illustration of the method called $BLOCCE = (\mathrm{Gen}_{\mathrm{BLOCCE}}, \mathrm{Embed}, \mathrm{Extract})$ (Blockchain Covert Channel) in detail.

It is assumed that the latent message is of constant length and known to both Alice and Bob. Before transmission, a pretreatment of M, a symmetric encryption algorithm $SE = (\mathrm{Gen}_{\mathrm{SE}}, \mathrm{Enc}, \mathrm{Dec})$ whose secret key k is required. The protocol shown in Figure 7 proceeds as follows:

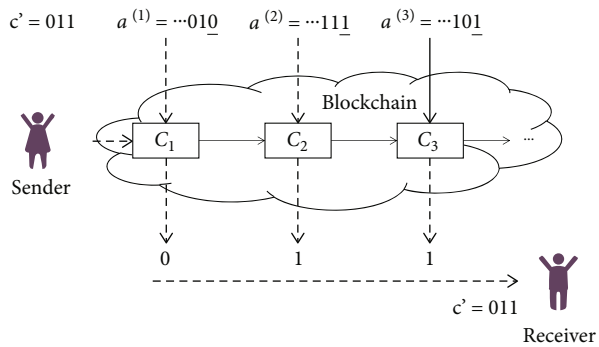


Figure 7: A simplified overview of BLOCCE.

Step 1 ($\text{Gen}_{\text{BLOCCE}}$): Alice produces a lot of private/public key pairs $(s_k^{(1)}, p_k^{(1)}), (s_k^{(2)}, p_k^{(2)}), \cdots, (s_k^{(n)}, p_k^{(n)})$ and calculates the billing addresses $a^{(1)}, a^{(2)}, \cdots, a^{(n)}$ correspondingly

Step 2 (Embed): a small quantity of transfer transactions are generated by Alice from her individual account to the newly generated addresses in the previous step. In advance, according to the secret text $c' \longleftarrow \lambda || c, c \longleftarrow \text{Enc}(k, M)$ of the plaintext M and a random message start indicator λ, Alice orders these addresses to use the least significant bits (LSBs) of the payment addresses to make up c'. Note that both Alice and Bob know n_λ (the length of λ) and $|c|$ (the length of c) denoted by $N = n_\lambda + |c|$, where N is the length of hidden text message c'.

Step 3: Alice submits payments to the blockchain in the correct order and for each block, there is a single payment from Alice, so that the message can be gathered properly.

Step 4 (Extract): after reading the blockchain and filtering out transactions made by Alice, Bob can concat the LSBs of the payment addresses to obtain the hidden text. The first n_λ bits generates λ, while the next $|c|$ generates ciphertext c. Eventually, the plaintext message $M \longleftarrow \text{Dec}(k, c)$ can be extracted.

This protocol is considered the first attempt to combine blockchain with covert communication. The size of data carried in each transaction is so small that it will not be used for real. Although it has extremely low efficiency, it contributes a lot to the following research.

4.2. V-BLOCCE. Improving on the BLOCCE, Lejun et al. [32] proposed the V-BLOCCE, a covert communication method. In this method, the covert message is coded in the form of Base58 and the addresses including embedded data are generated by Vanitygen. Figure 8 shows the integrated procedure of the system.

Step 1: the message M is encrypted into a provisional cipher and then completed with the Base58 encoding to get the ultimate ciphertext.

Step 2: All the different characters appeared in the final ciphertext are stitched into a string which is provided for the Vanitygen software to generate Bitcoin addresses.

Step 3: Alice goes through these addresses looking for matches with the characters of ciphertext. If a match is found, the index of the character in the ciphertext and the index of the addresses are recorded as a tuple in a list. In this way, the information is embedded with the index list generated. Then, the addresses are sorted in order of their hash value and the index lists are encrypted to be filled in the OP_RETURN field.

Step 4: Bob scans the transactions sent by Alice for the transaction information. Next, he picks up the order of the address used and the index information from OP_RETURN which are used to determine every character and its location to restore the ciphertext. It is decoded by Base58 and then decrypted to acquire the real message M.

Based on the protocol in Subsection 4.1, it improves the efficiency of data entry into special transactions. Furthermore, it leads the attention of researchers to the script field in the transaction, which brings more potential application to this domain.

4.3. DLchain. Tian et al. [33] proposed a new blockchain covert channel construction scheme implemented with dynamic labels which are generated on the basis of the distribution of a large amount of real transaction data to guarantee its dynamic and variety. Here, the OP_RETURN script is chosen to carry labels whose fixed length is 23. The scheme process is shown in Figure 9.

Step 1: dynamic labels are produced by both sides in the communication at the same time. So the receiver is able to recognize specific transactions coming from the sender.

Step 2: a prenegotiated key is employed to encrypt the message. Then, the encrypted messages take the place of the private key required in signature generation. At last, the sender signs two transactions with the particular private-key and packages the agreed label into them.

Step 3: after verification, the transactions spread through peer-to-peer connections online and the corresponding record is kept by the blockchain.

Step 4: in the light of the negotiated dynamic label, transactions with information hidden can be identified and filtered.

Step 5: the receiver extracts the secret message.

Taking the DGA algorithm Banjori [34] as a reference, the dynamic label generation algorithm is designed after analyzing the universal distribution of the OP_RETURN scripts in the practical transaction data statistically to make the label indiscernible.

This protocol focuses its attention on the verification phase. It is perceived that many special transactions with the covert message are attached to a fixed label so that receiver can make a distinguishment. This protocol puts forward a dynamic label generation algorithm which is frontline at that time.

4.4. Whispers on Ethereum: Blockchain-Based Covert Data Embedding Schemes. Liu et al. [35] used the VALUE field of a transaction in the Ethereum system to construct a one-bit embedding (OBE) scheme and an HMAC-based multiple-bit embedding (HMAC-based MBE) scheme. Furthermore, a Hash-based multiple-bit embedding (Hash-based MBE) scheme is proposed to enhance the covertness.

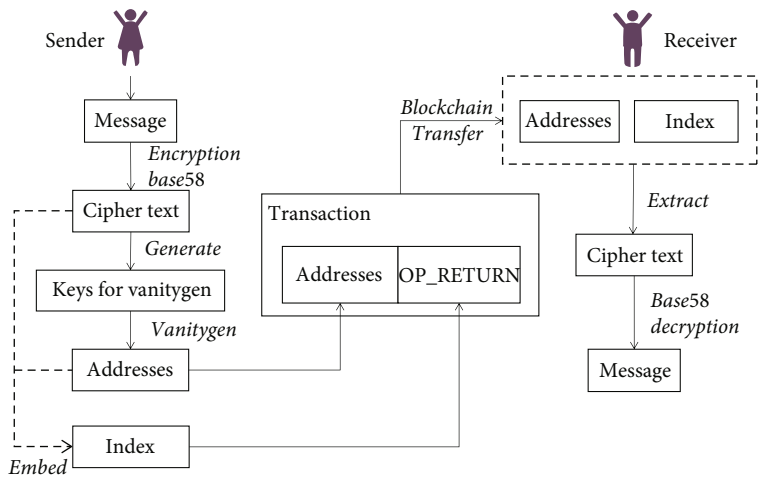


Figure 8: The procedure of the V-BLOCCE method.

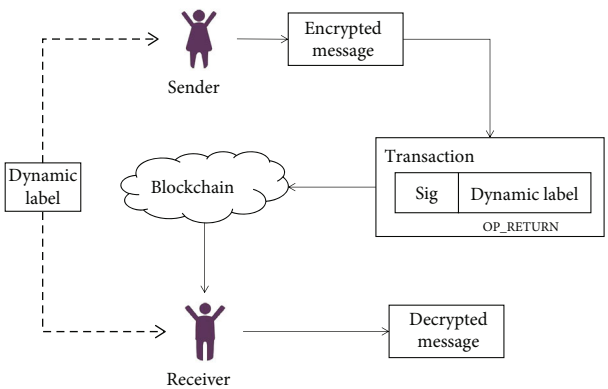


Figure 9: DLchain architecture.

The hidden message in all the proposed schemes is stored directly in the VALUE field.

Liu et al. [36] proposed a novel Monero-based covert communication system which provided higher security to fend off Eclipse attacks [37] and node crawling attacks [38]. The covert data is converted to decimal and then takes the place of the amount field in the transfer transaction. At the same time, the application of Stealth Address implements the anonymity of the receiver, while the application of the ring signature guarantees the privacy of the sender.

In these three schemes mentioned before, the AES encryption scheme is used to encrypt secret data M before it is embedded into the transactions which is defined as $c = r||\mathrm{AES}_{k_1}(r) \oplus M$. Here, c is the cipher text, k_1 is the encryption key and even if the same message M is encrypted for multiple times, the ciphertext will not be the same thanks to the random number r; thus, the chosen plaintext attack can be resisted.

4.4.1. The OBE Scheme. Transactions are constructed for each bit by filling in the VALUE field when traversing the ciphertext c bit-by-bit as is shown in Figure 10. There are two intervals V_0, V_1 representing the result of the HMAC. For every bit in c, if the bit $c[i]$ is 0, calculate a number VALUE to meet the demand that its HMAC result $\mathrm{HMAC}(k_2, \mathrm{VALUE})$ is in the set V_0. k_2 is the HMAC key. And when the bit $c[i]$ is 1, then calculate a number VALUE whose HMAC result $\mathrm{HMAC}(k_2, \mathrm{VALUE})$ stays in the interval V_1. In this scheme, only one bit is loaded in a transaction, which means the embedding rate is 1 bpt (bit per transaction).

Since the OBE scheme has low efficiency in the aspect of the embedding rate, a multiple-bit embedding scheme is further proposed as shown in Figures 11 and 12.

4.4.2. The HMAC-Based MBE Scheme. The first bit of the VALUE of the transaction is supposed to be 1. Then, we traverse the ciphertext c, for each bit $c[i]$, we select a number from 0 to 15 to make the HMAC result in the corresponding interval similarly to the OBE scheme. In this way, every 4 bits generated to represent 1 bit of the ciphertext will be placed in order together with the first bit 1 in the VALUE field of the constructed transaction.

4.4.3. The Hash-Based MBE Scheme. Apart from the covert data, the obfuscation data in this scheme is introduced to confuse the attacker. Before constructing the covert channel, a mixed hash root indicating the index of significant bits in the VALUE field of each transaction needs to be known to both the sender and the recipient. Valid ciphertext is divided into bits which are sequentially stored in the location of bit "1" in the binary representation of the mixed hash. Note that the second bit of VALUE and the first bit of the mixed hash should align. Moreover, the mixed hash of the first embedded transaction hashs from the mixed hash root while the mixed hash of the latter transaction hashs from that of the former transaction.

This protocol uses VALUE field in the transaction, which makes the embedding rate related to the length of value. Nevertheless, the requirements for special value in the special transaction may arise suspicion which needs be considered in the future work.

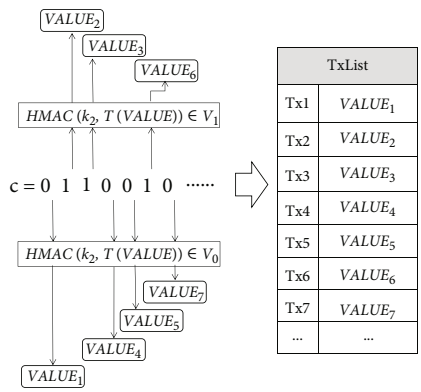


Figure 10: One-bit embedding (OBE) scheme.

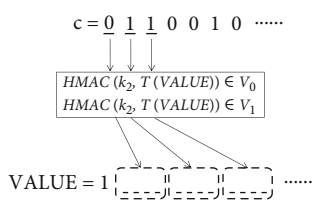


Figure 11: HMAC-based MBE scheme.

4.5. A Covert Communication Model Based on Ethereum's Whisper Protocol. Zhang et al. [39] designed a new covert communication model shown in Figure 13 founded on the Whisper protocol which is applicable to Ethereum.

In the architecture of Whisper network, envelope which consists of enciphered data and unencrypted data is the basic form of data transmission. Each envelope contains a topic introduced to scan for the target envelopes, and there is a one-to-one mapping between each topic and a encrypt/decrypt key. The protocol proceeds as follows:

Step 1: before transmission, each node participating in the covert communication needs to interact with Whisper to obtain the topic and key. Next, the secret message M will be encrypted to C to ensure its concealment.

Step 2: Alice sets the value of payload and padding to establish the envelope which is encrypted with the key obtained in step 1. The payload is either generated randomly or provided by Alice. For each bit of C, Alice compares it with each bit of the payload working as a carrier. If the match is successful, Alice will take down the indexes of the carrier and corresponding encrypted message.

Alice replaces the matched bits in C with * and repeats the former step until the message has only * or the payload does not contain any message characters. In the meanwhile, two index arrays are generated and stitched together as the first part of the padding P_1. Then, Alice sets a splitter R as the second part and a random redundant field P_2 as the last part of the padding. And P_1, R, P_2 are spliced into the padding.

Step 3: some attributes like TTL are packaged with the topic of the envelope set to construct the envelope.

Step 4: with the widespread of envelope, Bob is able to find out the particular envelope with the negotiated topic and obtains the payload and padding. After obtaining padding field, P_1 can be gained by deleting R and P_2 and then be decrypted to obtain the index arrays, according to which the ciphertext C can be restored and then decrypted to the original message M.

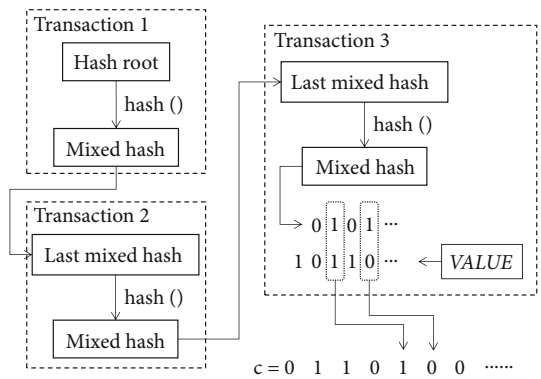


Figure 12: Hash-based MBE scheme.

This protocol takes advantage of Whisper protocol in Ethereum which is unique to other blockchain platforms. Focusing on the specific characteristics of current blockchain products provides innovation and alternative to the further research.

4.6. CCBRSN. Wang and Su [40] proposed a system called Covert Communication based on Bitcoin Regtest Self-built Network (CCBRSN), which takes blockchain as a covert communication channel.

The secret message is embedded into the blockchain's addresses in transmission. DES and Base58 are used to encrypt and code the secret message before embedding the encrypted message into a group of addresses which is employed to conduct a ciphertext-embedded transaction. As is shown in Figure 14, the system works as follows:

Step 1: Alice uses DES to encrypt the secret message M whose encryption key k is prenegotiated to get the result $\mathrm{DES}_k(M)$. Then, Base58 is called to encode $\mathrm{DES}_k(M)$ into $\mathrm{Base58}(b)$.

Step 2: Alice uses ECDSA while generating a private/public key pair $(\mathrm{sk}^{(1)}, \mathrm{pk}^{(1)})$. Then, $\mathrm{pk}^{(1)}$ needs to be computed using SHA256 hash, RIPEMD160 hash, and Base58 step by step to produce a corresponding address $a^{(1)}$.

Step 3: for every bit of $a^{(1)}$, Alice matches it with each bit of $\mathrm{Base58}(b)$. If succeed, Alice will record corresponding indexes of $a^{(1)}$ as set^a and indexes of $\mathrm{Base58}(b)$ as set^M which meet the equality that $a^{(1)}[set_k^a] = Base58(b)[set_k^M]$ (k means an arbitrary but the same place in the set).

Step 4: with the corresponding bits in $\mathrm{Base58}(b)$ replaced by *, $\mathrm{Base58}(b)$ is consequently transformed into $\mathrm{Base58}(b1)$. Then, Alice continues to repeat the same procedure for $\mathrm{Base58}(b1)$ until every bit in $\mathrm{Base58}(b1)$ is replaced.

Step 5: a transaction whose output addresses are $a^{(1)}$, $a^{(2)}, \cdots, a^{(n)}$ is submitted to the blockchain. And the earlier the address is generated, the more the transaction fee will be set to define the sequence. Then, the sets $\{\mathrm{set}^M\}_n$, $\{\mathrm{set}^a\}_n$ and transaction ID TxID are packed into a file *File* for the following extraction.

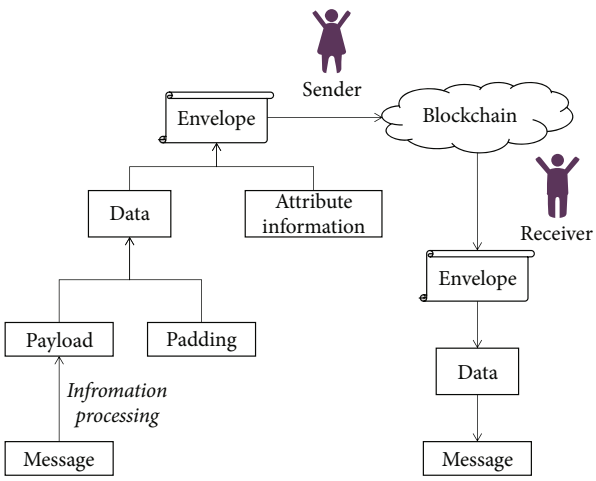


Figure 13: Covert communication model on Whisper.

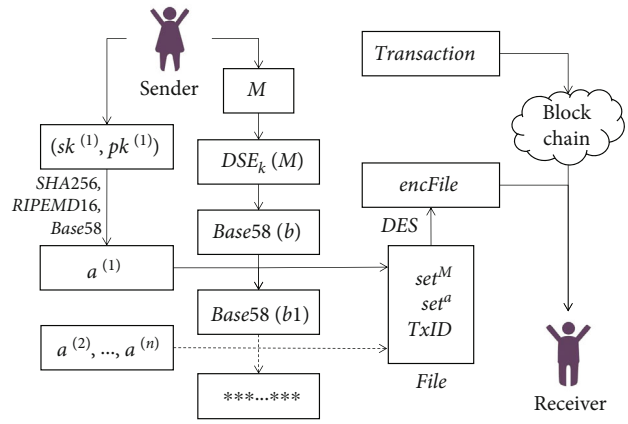


Figure 14: The process of CCBRSN execution.

Step 6: to protect the content of *File*, DES encryption is applied with a negotiated key before encrypting *File* into *encFile*.

Step 7: having received the *encFile*, Bob first decrypts it into *File* to obtain $\{\text{set}^M\}_n$, $\{set^a\}_n$ and TxID. And then, he finds the transaction in line with the TxID, gets the output addresses, and restores the ciphertext according to the correspondence.

Step 8: finally, after decoding with Base58 and decrypting with DES, Bob transforms the ciphertext into plaintext, so-called the secret message M.

With essential data for encryption/decryption recorded in an extra file and encoded data recorded in the addresses, this protocol achieves the cooperation between online transactions and offline transfers. However, when there are a lot of online transactions and offline files delivered to the receiver, it may lead to a mismatch.

4.7. A Covert Communication Model Realized by Using Smart Contracts. Zhang et al. [41] proposed a covert communication model combined with smart contracts to covertly transfer information under the blockchain circumstance. The parameters in the contract are used to carry the secret message, and the covert communication is completed by calling the smart contract.

Supposing that the secret message is encoded in ASCII format after the negotiation between Alice and Bob, a number ranging from 0 to 95 can represent any one of the 95 characters from 32nd to the 126th in the ASCII table. Take the biding contract as an example, the protocol shown in Figure 15 proceeds as follows:

Step 1: before communication, Alice and Bob need to reach a consensus on the effective price range interaction where quotations can be made. When bidding, the two decimal places are regarded as a carrier considering the number of characters used is 95.

Step 2: an actual price and the corresponding address of each bid are generated. Alice invokes the keccak(), a unidirectional cryptographic algorithm to produce an encrypted price of the bid with covert information embedded. Furthermore, the quoted price is exchanged between participants in enciphered form thus improving the tamper-proof ability when bidding.

Step 3: in the quotation announcement period, Alice provides both the special bids and the normal bids to the contract for comparison. If the bid matches the range, it is considered effective in the communication and will be used to restore the covert information.

Step 4: Bob extracts a price sequence from all the effective bids and selects all the prices within the valid price range. The number after the decimal point of the selected prices will be converted into a character according to the mapping relationship mentioned above. Then, the characters are ordered from low to high according to the number before the decimal point to generate the covert information.

Rather than common transfer transactions, this protocol makes use of smart contract where there are more application scenarios. Since the parameters of a customized contract are more flexible, this protocol provides better secrecy.

4.8. MRCC: A Practical Covert Channel over Monero. Guo et al. [42] proposed a practical and secure covert channel with no labels employed.

Before constructing a covert channel, both the sender and receiver have to reach an agreement on the data transfer algorithm and the necessary key information, including a public-key encryption scheme $\text{PKE} = (\text{Gen}, \text{Enc}, \text{Dec})$, a related key pair $(\text{PK}_e, \text{SK}_e)$ to encrypt/decrypt the message, and Bob's address (A, B). In the interest of brevity, let us suppose that both M and its corresponding ciphertext under PKE are of n-bit length. The protocol shown in Figure 16 proceeds as follows:

Step 1: the plaintext M is encrypted by PK_e, and C is the ciphertext.

Step 2: Alice establishes a particular transaction $tx = (PK[n], OTA, v, \sigma)$ which has the same format of common transactions. In the constructed transaction tx, $PK[n]$ is a public key set whose hash values' the least valid bits make up C, OTA is an one-time address of Bob, and σ is a significant ring signature generated by Alice's spending key.

Step 3: Alice submits the transaction tx to Monero's blockchain.

Figure 15: Data transmission method on bidding contract.

Step 4: while going through all of Monero's blockchain, Bob recognizes tx from the general transactions with the help of his tracking key.

Step 5: as $\mathrm{PK}[n]$ can be extracted from tx easily, Bob needs to pick up the least significant bit of each public key's hash value to reconstruct C. After the decryption of C using SK_e, the real message M will be the output.

This protocol pays attention to the technical characteristics of the blockchain platform, which makes the application of blockchain in the covert communication more customized than other protocols. However, there are some weaknesses that random value in elliptic curve (ed25519 in this protocol) cannot be replaced by any 256-bit number completely due to the range. So more details should be considered when pursuing research.

5. Comparison and Analysis

In this section, we will give some requirements for the blockchain-based covert communication channel and analysis based on the embedding position of the covert message.

There are some needs for the message embedding.

(i) Message Processing: in different kinds of protocols, the covert information is transformed into different forms depending on the way it splits. And for the concealment of the information, it is usually encrypted into ciphertext before transmission.

(ii) Indistinguishability: here, indistinguishability is defined from two aspects. On the one hand, indistinguishability in behavior is necessary. All the network behavior that happened during the phase of covert communication between the sender and receiver proceeds on the basis of universal blockchain protocols. So it is supposed to be indistinguishable from the behavior of ordinary blockchain users. On the other hand, the content and form of the blocks with a message are required to be indistinguishable. So that the malicious user observing the network will not have the ability to perceive hidden messages until the recognition method is known.

(iii) Compatible: the proposed covert communication system must be compatible with the existing public blockchains. It is not hard to figure out that the concealment of special transactions keeps improving with increasing normal transactions in a blockchain. Therefore, scenarios best suited for concealed data transmission are usually public blockchain. It is a must for the covert communication mechanism to be compatible with popular public blockchains without modifying their protocols

The comparison is shown in Table 2 and Table 3. It is assumed that there are 4 inputs in a Bytecoin transaction and 10 public keys in its ring signature.

5.1. Address Channel. Address Channel is a method using the address field of the blockchain when achieving covert communication. A Bitcoin transaction is a data structure consisting of inputs and outputs in which information of a fund flows from the initial place (inputs) to its destination (outputs). The inputs and outputs of the transaction have nothing to do with accounts or identity. They can be understood as a certain amount of Bitcoins with secret information locked. Only their owner or someone who knows the secret can unlock them. Transactions have multiple data fields as is shown in Table 4.

Blockchain address generation goes through a series of algorithms such as ECDSA, SHA-256, and RIPEMD-160, which means the generated address is random in some way under meeting standards. So as long as the addresses that occurred in the blockchain are legitimate, the address-based covert message will not be censored.

Partala [31] proposed an ideal covert communication protocol based on the blockchain. The protocol used a symmetric encryption scheme to deal with the covert message, attached a random tag to it, converted each character in it into an 8 bits binary number, and then embedded it into the generated address bit-by-bit. There are as many addresses needed as the bits of the processed information, which results in low efficiency.

Lejun et al. [32] chose Base58 rather than binarization to encode the message which directly increases the embedding bytes. With Vanitygen generating addresses which can be reused in information embedding and the OP_RETURN field utilized to save the index lists, the efficiency of information transmission increased greatly.

Wang and Su [40] encrypted and encoded plaintext, then embedded it into Bitcoin's output address. With these addresses as a carrier of the covert message, special transactions were constructed to transmit information. However, due to the size limit of the transaction, the message could be too large to be embedded. Meanwhile, the file for decryption is a necessity for restoration and is very important to the receiver, which requires an extrasecure transmission tunnel in the offline transfer.

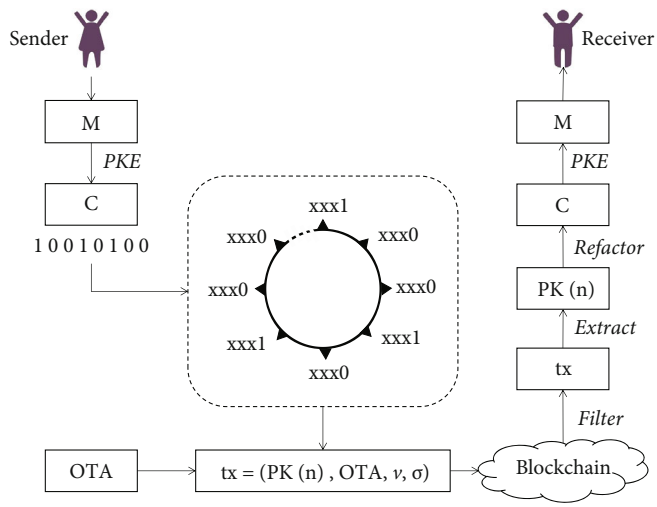


Figure 16: MRCC protocol.

Luo et al. [48] proposed a covert communication method based on Bitcoin transactions which designs the index matrix of the transaction address for the first time. The covert message is embedded in the address interaction relationship combined with the transaction amount. This embedding method decreases the quantity of transactions while increasing the embedding capacity of a single transaction. Besides, a recycled transaction index matrix of address is designed to save the intracorporeal relationship, thus making the address reused.

Minaei et al. [44] proposed R3C3 using the Insight API. It is widely acknowledged that the addresses in the Zcash are of two types: shielded address and unshielded address, the transactions fall into two kinds: minting transaction which creates new coins and pouring transaction which transfers the value of one coin to some new coins. Shielded outputs of a pouring transaction are tuples of the form $(\mathrm{rt}, \mathrm{sn}^{\mathrm{old}}, \mathrm{cm}_1^{\mathrm{new}}, \mathrm{ct}_1, \mathrm{com}_2^{\mathrm{new}}, \mathrm{ct}_2, h, \mathrm{vk}^*, \sigma^*, \pi_{\mathrm{pour}})$, in which rt, sn^{old}, and com_i^{new} are used to complete the zero-knowledge proof π_{pour}, ct_i is used to reconstruct the coin. But since the receiver is in alliance with the sender, ct_i can be reused to carry encrypted data whose portion constitutes 584 bytes.

Cao et al. [51] put forward a chain-based data embedding scheme. Rather than using one address once in the transaction, the proposed scheme uses the derivation from input address to output address to denote 0 or 1. In this way, one bit of binary data is embedded into a transaction.

With low transport volumes, Address Channel is favored when transferring short covert messages. Since each transaction is recorded according to the time sequence, there is no need to worry mess up covert transactions which makes Address Channel suitable for interactive communication.

5.2. Value Channel. Value Channel is a method using the VALUE field of the blockchain when achieving covert communication. It is noted that the VALUE field can be the amount of transfer, the parameters of the smart contract, or other flexible values.

In the Ethereum, its editable fields are described in Table 5 concerning the summary in [35]. Compared with the address field and the input field, the VALUE field is much more flexible. Whatever the value is, it complies with the specifications. So it can be used to store covert messages.

Liu et al. [35] proposed an OBE scheme and two MBE schemes, all of which take advantage of the VALUE field in transactions to construct a subliminal communication channel in Ethereum for the first time. Although the chosen location of the embedded message is exquisite, there lacks a method to recognize special transactions. The receiver has to consider all the transactions delivered from the sender as special transactions, which is bound to increase the processing burden of the receiver.

The smart contract is regarded as a covert communication carrier worth considering due to its diversity, redundancy, and programmability of data. The parameters submitted when calling a contract can be treated as a value.

Zhang et al. [41] proposed the scheme which first encrypted and encoded the data to be transferred in the form of ASCII. To call the smart contract, parameters need to be constructed accordingly. Besides, redundant parameters can be set to enhance the concealment of secret information and to defend against malicious attacks. When using hexadecimal as the format of information and containing eight prices in the bidding contract for each transaction, the loading capacity of a single transaction is 4 bytes.

Basuki et al. [45] proposed the joint use of image steganography and smart contract calling in the Ethereum to achieve a natural transactional model and high volume covert communication. The picture-based steganography is used to supply the scheme with a large storage capacity. By contrast, only the instruction manual is required to be recorded in the transactional steganography to retrieve the stegano-image, thus decreasing the amount of transactions. A smart contract for sensor gateways serves as the experimental platform in the proposed scheme. Pretending as one of the gateways, the sender will update sensor data regularly to the blockchain.

Table 2: Contrast among blockchain-based covert communication channel.

Scheme	Time	Type	Position	Environment	Hiding capacity (bit)/TX
Partala [31]	2018	Address Channel	Input address	Bitcoin	1
Frkat et al. [43]	2018	DSA Channel	Signature	Botnet	256
Minaei et al. [44]	2018	Address Channel	Output shielded address	Zcash	9344
Basuki and Rosiyadi [45]	2019	Value Channel	Contract parameter	Ethereum	29
Recabarren and Carbunar [46]	2019	DSA Channel	Signature	Bitcoin	13200
Alsalami and Zhang [47]	2019	DSA Channel	Signature	Bytecoin	16384
Tian et al. [33]	2019	Script Channel	OP_RETURN	Bitcoin	256
Lejun et al. [32]	2020	Address Channel	Input address	Bitcoin	36
Wang and Su [40]	2020	Address Channel	Output address	Bitcoin	34
Luo et al. [48]	2020	Address Channel	Interaction between address and transaction amount	Bitcoin	Uncertain
Liu et al. [36]	2020	Value Channel	VALUE	Ethereum	OBE:1
		Value Channel	VALUE	Ethereum	HMAC: 0.25* (ValueLength-1)
		Value Channel	VALUE	Ethereum	Hash: 0.5*(ValueLength-1)
Alsalami and Zhang [49]	2020	DSA Channel	Signature	Bytecoin	16384
Zhang et al. [39]	2020	Script Channel	Whisper payload	Ethereum	256
Cao et al. [38]	2020	Address Channel	Public key derivation	Bitcoin	1
Gao et al. [50]	2020	Script Channel	OP_RETURN	Bitcoin	640
Zhang et al. [41]	2021	Value Channel	Contract parameter	Ethereum	41.8
Liu et al. [36]	2022	Value Channel	Amount	Monero	39

Table 3: Comparison of transaction cost (updated Feb. 27, 2022).

Scheme	Time (year)	Environment	Capacity(/TX)	Tx fee(coin)	Price/coin	TX fee($)	Cost($)/1 MB
Minaei et al. [44]	2018	Zcash	1168 byte	0.0001 ZEC	109.64USD	0.011	9.4
Recabarren and Carbunar [46]	2019	Bitcoin	1650 byte	0.0000165 BTC	38763.7USD	0.64	387.89
Gao et al. [50]	2020	Bitcoin	80 byte	0.000039 BTC	38763.7USD	0.1512	1890
Alsalami and Zhang [49]	2020	Bytecoin	2 KB	0.01 BCN	0.00016USD	0.0000016	0.0000931
Liu et al. [36]	2022	Monero	39 bit	0.000009 XMR	155.51USD	0.0014	287

Compared with Address Channel, there is more flexibility in Value Channel. Value Channel is suitable for all blockchain platforms. But long communication which draws attackers' attention is not available, because the value of the special transaction is so intentionally designed that it may be exposed by statistical analysis.

5.3. Digital Signature Algorithm (DSA) Channel. DSA Channel is a method using the signature field of the blockchain when achieving covert communication. It is easy to find out that all the mainstream blockchain cryptocurrencies use original cryptographic primitives, for instance, digital signatures and noninteractive zero-knowledge proofs, so

Table 4: Transaction structure in Bitcoin.

Type	Name	Description	Length
Value	Version	Define the rules for this transaction	4 bytes
Value	Input counter	The number of inputs included	1-9 bytes
UTXO	Input	One or more transaction inputs	Uncertain
Value	Output counter	The number of inputs included	1-9 bytes
UTXO	Output	One or more transaction outputs	Uncertain
Time	Locktime	A block number or UNIX timestamp	4 bytes

Table 5: Transaction structure in Ethereum.

Type	Name	Description	Length
Address	From	The account of sender	Up to 20 bytes
Address	To	The account of receiver. If empty, create a contract.	Up to 20 bytes
Gas	Gas limit	Estimated maximum gas	Up to 32 bytes
Gas	Gas price	The price to cost ether	Up to 32 bytes
Value	VALUE	The amount of ether	Up to 32 bytes
Input	INPUT	An arbitrary message, or a code segment to create a contract, or a function call to a contract	0-about 700 KB

that the uncontrollability of random value in the blockchain can be harnessed under conscious control to complete covert communication.

Alsalami and Zhang [47] demonstrated how to embed covert messages in any subsistent public blockchain where there is enough redundant data. Covert information is embedded in the signatures of the transaction and then broadcast over the blockchain. To isolate and recognize transactions equipped with stegano-text, the receiver has to scan every new transaction appended to the blockchain. If desired transactions are detected, a secret message will be extracted. Otherwise, the receiver keeps searching.

Alsalami and Zhang [49] designed and implemented the first practical covert broadcast communication system which combines steganographic technique with Boneh et al. [52] broadcast encryption scheme. In the CryptoNote protocol, the random numbers in the ring signature are uncontrolled random group elements that can be employed. Considering that the highest significant bit of the random value is not uniformly distributed on 0 and 1, only the least 252 bits of the random number are used to embed covert messages to ensure indistinguishability.

Recabarren and Carbunar [46] introduced a Bitcoin-based framework, called Tithonus, which provides a censorship-resistant communication mechanism. Rather than employing a blockchain consensus mechanism of low speed and high expense, Tithonus makes use of the peer-to-peer gossip protocol in the blockchain as a straightforward agent to exchange covert messages. The encrypted information is embedded in the elliptic curve point generation procedure.

Ali et al. [53] proposed ZombieCoin 2.0 and validated Brenner's discussion [54] that the blockchain technology can be applied to the transmission of C&C instructions secretly. The C&C instruction within 32 bytes can be embedded in the 32-byte ECDSA private key. After the prototype implementation of ZombieCoin 2.0, Ali et al. deployed and controlled the Bitcoin network successfully.

Frkat et al. [43] presented ChanChannels which is a hidden botnet communication in which covert message is injected into the classical Elliptic Curve Digital Signature Algorithm (ECDSA) commonly used in the blockchain. In order to insert a covert message, the randomness in the signature is substituted with the hidden message. Using broadband secret channels, the proposed method could be distributed over multiple blockchains instead of a specific blockchain.

DSA Channel is for use in all scenarios where participants have the capabilities to sign and verify the customized signature. DSA Channel is preferred in the scenario relating to confidential information because outside users can only see on the face of it.

5.4. Script Channel. Script Channel is a method of calling other scripts when achieving covert communication. Bitcoin client calls a verifying script to distinguish transactions. A scriptPubkey is written into UTXO which simultaneously contains a scriptSig written in the same scripting language as the previous script. When a Bitcoin transaction is verified, the scriptSig in each input will be executed at the same time with the corresponding sigPubkey (without mutual interference), so as to check whether the transaction meets the applicable condition.

In the 0.9 version of the Bitcoin client, OP_RETURN operator has been used to compromise a situation that the blockchain is utilized to store data irrelevant to Bitcoin payment. OP_RETURN script allows developers to append nontransactional data of 40 bytes to the output of the transaction and create a remarkable nontransaction output with no need for storage in the UTXO set. The outputs of OP_RETURN are logged on the blockchain, which increases both the disk space consumption and the size of the blockchain. However, since they are not stored in UTXO, they will

not expand UTXO memory, nor will they overburden all nodes at the cost of consuming expensive memory.

Tian et al. [34] proposed DLchain, a mechanism which substitutes the private key with the covert message and presents a dynamic label generation algorithm. It can dynamically create tags that cannot be tracked statistically. Rather than on the stitching of characters in some fixed position, the generation scheme is founded on the statistical analysis of OP_RETURN distribution in plenty of actual transactions.

Gao et al. [50] proposed a mechanism to establish covert channels in public blockchain channels through the kleptography algorithm and OP_RETURN operator. In the submitted special transactions, the signature is dealt with using kleptography algorithm and OP_RETURN field is used to store the encrypted covert message. Then, the receiver can distinguish particular transactions by detecting the signature data.

As a communication protocol for information synchronization, Whisper allows nodes of distributed platforms to interact with each other securely and privately.

Abdulaziz et al. [55] put forward a secure and anonymous decentralized messaging application which can be divided into two phases. One is the temporary Topic and key distribution phase in which Whisper is used to transmit asymmetrically encrypted messages consisting of a random symmetric key for message processing and temporary Topic for message clarification. The other is the communication phase in which the sender constructs an envelope with an encrypted message and prenegotiated Topic. Although the information transferred is encrypted but it is stored directly in the envelope which is somewhat unsafe.

Zhang et al. [39] proposed a new kind of covert communication model for Ethereum. Communication Topic and data-encrypting key are also decided before communication. This paper takes a random string as the objective to refer to while recording the index information at the same time. Rather than transmit the encrypted message directly, it is separated into bytes mapping to another string and needs reorganization after being accepted. Considering that the character string is completely random and only lowercase English letters are contained, the amount of information can be calculated as $\log_2 26$ bits per character (bpc).

Zhang et al. [56] proposed a covert communication model with the Ethereum Whisper model. The payload field of envelope is used as a communicational unit, while the corresponding index information is logged in the padding field. In order to simplify the filtering procedure, this method sets the public key hash value of the recipient for the topic of the envelope.

Script Channel is able to achieve covert communication with high capacity which is suitable for some dense scenarios. With plenty of scripts used in the blockchain, Script Channel can be combined with other covert data transmission technologies. Furthermore, Script Channel is suitable for certain scenarios dedicated to certain blockchain platforms by using their unique scripts, which means the whole process can be customized.

5.5. Findings. As mentioned above, the blockchain-based covert communication channel is divided into four categories, Address Channel, Value Channel, Digital Signature Algorithm (DSA) Channel, and Script Channel. The analysis of the four kinds of channels is shown in Table 6.

Address Channel substitutes the input or output addresses in a transaction for an encoded covert message. The address used as a carrier has to be legitimate, so a covert message needs to go through several steps to the address, resulting in the low capacity and low efficiency of the Address Channel. However, thanks to the commonness of addresses, Address Channel has good invisibility. Address Channel is suitable for intercovert communication with low demand of transport volumes.

Similar to Address Channel, Value Channel replaces the value part of a transaction which also naturally exists in the transaction. It has the characteristic of low capacity. However, as the VALUE field is plain, the covert message may be detected by statistical methods.

Just as its name implies, DSA Channel utilizes mostly the randomness in a transaction to covertly transmit the information, which provides high capacity. Besides, information keeps being processed in the following procedures of the signature algorithm after being inserted, which leads to high concealment.

Script Channel requires extra fields or protocols which is not common or necessary in a transaction. The data fields employed to construct Script Channel may be a plain message or enciphered text. It is so easy to overlook the data fields when dealing with transactions that low concealment is gained. However, in Script Channel, these data fields are always dedicated storage parameters providing high capacity and efficiency.

Among the applications of blockchain in the covert communication, Address Channel and Value Channel are favored in short communication while DSA Channel and Script Channel are favored in long communication. When related to confidential information, DSA Channel is preferred since the ciphertext is not exposed. However, Script Channel is the most suitable for customized demand.

6. Discussion and Future Work

There are pros and cons in the applications of blockchain in the covert communication. On the one hand, the characteristics of high reliability, strong robustness, and decentralization of the blockchain bring about overwhelming change in the traditional covert communication. Blockchain has the capacity to hide both sender's and receiver's identities. With variable ways to embed data, blockchain is a natural alternative where stealth is required. On the other hand, the openness and tamper-resistance of blockchain make it simple for the potential adversary to fetch data which does no good to covertness. There are some drawbacks as follows.

(i) Uncommon flow: the size of data carried in a special transaction is limited. If someone wants to send a long text, so many transactions will be constructed that it is conspicuous in any way. Furthermore, if the amount of an irregular type of transaction soars,

Table 6: Analysis of channels.

Type	Capacity	Concealment	Efficiency
Address Channel	Low	High	Low
Value Channel	Low	Low	Medium
DSA Channel	High	High	Low
Script Channel	High	Low	High

reasonable suspicion exists there emerges covert communication.

(ii) Inefficient screening: in most cases, the receiver needs to scan all the received transactions and execute a recognition algorithm for special transactions, which leads to low efficiency

(iii) Vaporization of covertness. It is out of disputes that covert message stays hidden after the communication at first. But as time goes by, attackers can collect transactions sufficient enough to analyze, which increases the risk of exposure

A satisfactory covert communication protocol is supposed to hide its existence as much as possible while protecting the identities of participants in the communication at the same time especially when it has been detected. As is detailed below, some research directions are recommended in future work.

6.1. One-to-Many Solution. Construct an efficient scheme for one-to-many covert communication, that is, the sender can distribute the same secret message imperceptibly at the same time to many receivers whose communication keys are different. Current covert communication mainly allows people to be in touch one-to-one with the session key negotiated. If there requires one-to-many covert communication, it is more convenient to construct a special multikey protocol rather than embed the same message multiple times for each receiver; then, the receiver can extract the message with their own key. The existence of such a multikey covert communication scheme can be verified in future work.

6.2. Temporariness within Time Slice. Due to the immutability of blockchain, once the covert message is embedded and submitted, the probability of altering or withdrawing it is extremely low. Besides, the integrity guarantees are not supported by any centralized party, but by the consensus of the entire network. The embedding of covert messages is too fragile to stay undetected forever. Once the embedding is detected, the hidden message is supposed not to be extracted successfully. It is inevitable to consider the revocation and variation of covert communication by introducing an effective time slice. Considering that blockchain includes timestamp which provides a temporal feature by nature, how to embed a covert message into blockchain temporarily will become a trend of future work. The existence of such a scheme can be verified in further research for the future.

6.3. Key Compromise Impersonation Resilience. In cryptography, key disclosure is a serious problem. Once the communication key is leaked, the attackers may be able to forge a fake message sent to the legal receiver which disturbs the peace of the covert communication system. However, key-evolution and key-insulated schemes have been put forward to solve the problem mentioned above. Therefore, how to construct a provably secure antikey-disclosure scheme will be the following step of this paper.

7. Conclusion

In this paper, we provide an in-depth review of the applications of blockchain in the covert communication. In recent decades, blockchain technology has attracted a lot of interest in various applications, such as the Internet of Things, identity management, and covert communication. Covert communication uses information hiding techniques to embed secret messages into information carriers during the transmission, which protects the privacy of the secret message. With the underlying technology of cryptocurrencies in blockchain, there are a growing number of researches about covert communication on the blockchain. We investigate these schemes and classify them into certain types according to the hiding position. We also present the difference and comparisons among these types. With the analysis of current protocols, we aim to put forward some new schemes with higher capacity, lower cost, and better efficiency in the future work. With the same underlying architecture of blockchain, analyzing the applications of blockchain in the covert communication contributes to the following research.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

The work was supported the National Key Research and Development Program of China (No. 2019QY0800), the Shandong Provincial Key Research and Development Program (Nos. 2020CXGC010107 and 2021CXGC010107), the National Natural Science Foundation of China (Nos. U21A20466, 62172307, 61972294, and 61932016), the Blockchain Core Technology Strategic Research Program of Ministry of Education of China (No. 2020KJ010301), the Special Project on Science and Technology Program of Hubei Province (No. 2020AEA013), the Natural Science Foundation of Hubei Province (No. 2020CFA052), the Wuhan Municipal Science and Technology Project (No. 2020010601012187), the Foundation of Hangzhou Innovation Institute, Beihang University (No. 2020-Y10-A-019), the Peng Cheng Laboratory Project (Grant No. PCL2021A02), and the Foundation of Guangxi Key Laboratory of Trusted Software (No. kx202001).

References

[1] G. J. Simmons, "The prisoners' problem and the subliminal channel," in *Crypto*, Plenum Press, New York, 1984.

[2] S. Nakamoto, "Bitcoin: a peer-to-peer electronic cash system," *Decentralized Business Review*, p. 21260, 2008.

[3] J. Willett, M. Hidskes, D. Johnston, R. Gross, and M. Schneider, *Omni protocol specification (formerly mastercoin)*, vol. 28, 2016white paper.

[4] M. Rosenfeld, "Overview of colored coins," in *White paper, bitcoil. co. il*, vol. 41, p. 94, Coinprism, 2012.

[5] R. Roy and S. Changder, "Steganography with projection aided payload dimension reduction and reconstruction for military covert communication," *International Journal of Computer Applications*, vol. 139, no. 3, pp. 32–37, 2016.

[6] Z. Hijaz and V. S. Frost, "Exploiting OFDM systems for covert communication," in *2010-MILCOM 2010 MILITARY COMMUNICATIONS CONFERENCE*, pp. 2149–2155, San Jose, CA, USA, 2010.

[7] J. F. Harvey, M. B. Steer, and T. S. Rappaport, "Exploiting high millimeter wave bands for military communications, applications, and design," *IEEE Access*, vol. 7, pp. 52350–52359, 2019.

[8] G. J. Simmons, "Subliminal communication is easy using the DSA," in *Workshop on the Theory and Application of of Cryptographic Techniques*, pp. 218–232, Springer, 1993.

[9] J.-K. Jan and Y.-M. Tseng, "New digital signature with subliminal channels based on the discrete logarithm problem," in *Proceedings of the 1999 ICPP Workshops on Collaboration and Mobile Computing (CMC'99). Group Communications (IWGC). Internet '99 (IWI'99). Industrial Applications on Network Computing (INDAP). Multime*, pp. 198–203, Aizu-Wakamatsu, Japan, 1999.

[10] J.-M. Bohli and R. Steinwandt, "On subliminal channels in deterministic signature schemes," in *International Conference on Information Security and Cryptology*, pp. 182–194, Springer, 2004.

[11] J.-M. Bohli, M. I. G. Vasco, and R. Steinwandt, "A subliminal-free variant of ECDSA," in *International Workshop on Information Hiding*, pp. 375–387, Springer, 2006.

[12] A. Hartl, R. Annessi, and T. Zseby, "A subliminal channel in EDDSA: information leakage with high-speed signatures," in *Proceedings of the 2017 International Workshop on Managing Insider Security Threats*, pp. 67–78, Dallas, TX, USA, 2017.

[13] R. L. Rivest, A. Shamir, and Y. Tauman, "How to leak a secret," in *International Conference on the Theory and Application of Cryptology and Information Security*, pp. 552–565, Springer, 2001.

[14] Q. Dong, X. Li, and Y. Liu, "Two extensions of the ring signature scheme of Rivest-Shamir-Taumann," *Information Sciences*, vol. 188, pp. 338–345, 2012.

[15] B. Wang, Z. Zhang, and F. Zhang, "Subliminal channels in the code-based ring signature scheme," in *2019 14th Asia Joint Conference on Information Security (AsiaJCIS)*, Kobe, Japan, 2019.

[16] J. A. B. Dykstra, *A framework for network covert channel detection*, Iowa State University, 2004, PhD thesis.

[17] C. Wang, Y. Yuan, and L. Huang, "Base communication model of IP covert timing channels," *Frontiers of Computer Science*, vol. 10, no. 6, pp. 1130–1141, 2016.

[18] S. Gianvecchio, H. Wang, D. Wijesekera, and S. Jajodia, "Model-based covert timing channels: automated modeling and evasion," in *International Workshop on Recent Advances in Intrusion Detection*, pp. 211–230, Springer, 2008.

[19] J. Millen, "20 years of covert channel modeling and analysis," in *Proceedings of the 1999 IEEE Symposium on Security and Privacy (Cat. No. 99CB36344)*, pp. 113-114, Oakland, CA, USA, May 1999.

[20] S. M. R. Farschi and H. Farschi, "A novel chaotic approach for information hiding in image," *Nonlinear Dynamics*, vol. 69, no. 4, pp. 1525–1539, 2012.

[21] J. M. Blackledge and A. R. I. Al-Rawi, "Steganography using stochastic diffusion for the covert communication of digital images," *International Journal of Applied Mathematics*, vol. 41, no. 4, 2011.

[22] L. Ji, Y. Fan, and C. Ma, "Covert channel for local area network," in *2010 IEEE International Conference on Wireless Communications, Networking and Information Security*, pp. 316–319, Beijing, 2010.

[23] H. Dol, B. Quesson, and F. Benders, "Covert underwater communication with marine mammal sounds," in *Undersea Defence Technology-UDT Europe 2008*, ACM, Glasgow, UK, 2008.

[24] D. Kundur and K. Ahsan, "Practical internet steganography: data hiding in IP," *Proc. Texas wksp. security of information systems*, 2003.

[25] C. G. Girling, "Covert channels in LAN's," *IEEE Transactions on software engineering*, vol. SE-13, no. 2, pp. 292–296, 1987.

[26] D. Mohanty and D. Mohanty, "Ethereum Architecture," in *Ethereum for Architects and Developers*, Apress, Berkeley, CA, 2018.

[27] *Whisper poc 2 protocol spec*https://eth.wiki/concepts/whisper/poc-2-protocol-spec.

[28] M. Al-Bassam, "Scpki: a smart contract-based PKI and identity system," in *Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts*, pp. 35–40, Abu Dhabi United Arab Emirates, 2017.

[29] Y. Zhang and J. Wen, "The IoT electric business model: using blockchain technology for the Internet of Things," *Peer-to-Peer Networking and Applications*, vol. 10, no. 4, pp. 983–994, 2017.

[30] J. Cunningham and J. Ainsworth, "Enabling patient control of personal electronic health records through distributed ledger technology," *Studies in Health Technology and Informatics*, vol. 245, pp. 45–48, 2018.

[31] J. Partala, "Provably secure covert communication on blockchain," *Cryptography*, vol. 2, no. 3, p. 18, 2018.

[32] Z. Lejun, Z. Zhijie, W. Weizheng et al., "A covert communication method using special Bitcoin addresses generated by Vanitygen," *Cmc-computers Materials & Continua*, vol. 65, no. 1, pp. 597–616, 2020.

[33] J. Tian, G. Gou, C. Liu, Y. Chen, G. Xiong, and Z. Li, "DLchain: a covert channel over blockchain based on dynamic labels," *ICICS*, Springer, 2020.

[34] P. Barford, "The DGA of Banjori," February 10, 2015, https://blog.51cto.com/rude3knife/2910831.

[35] S. Liu, Z. Fang, F. Gao et al., "Whispers on ethereum: blockchain-based covert data embedding schemes," in *Proceedings of the 2nd ACM International Symposium on Blockchain and Secure Critical Infrastructure*, Taipei, Taiwan, China, 2020.

[36] L. Liu, L. Liu, B. Li, Y. Zhong, S. Liao, and L. Zhang, "MSCCS: a Monero-based security-enhanced covert communication system," *Computer Networks*, vol. 205, article 108759, 2022.

[37] E. Heilman, A. Kendler, A. Zohar, and S. Goldberg, "Eclipse attacks on fBitcoin'sgf Peer-to-Peerg network," in *24th USENIX Security Symposium (USENIX Security 15)*, pp. 129–144, Washington, D.C., USA, 2015.

[38] T. Cao, J. Yu, J. Decouchant, X. Luo, and P. Verissimo, "Exploring the Monero peer-to-peer network," in *International Conference on Financial Cryptography and Data Security*, pp. 578–594, Springer, 2020.

[39] Z. Zhang, L. Zhang, W. Rasheed et al., "The research on covert communication model based on blockchain: a case study of Ethereum's Whisper Protocol," Springer, Singapore, 2020.

[40] W. Wang and C. Su, "CCBRSN: a system with high embedding capacity for covert communication in Bitcoin," *ICT Systems Security and Privacy Protection*, vol. 580, pp. 324–337, 2020.

[41] L. Zhang, Z. Zhang, W. Wang, Z. Jin, Y. Su, and H. Chen, "Research on a covert communication model realized by using smart contracts in blockchain environment," *IEEE Systems Journal*, pp. 1–12, 2021.

[42] Z. Guo, L. Shi, M. Xu, and H. Yin, "MRCC: a practical covert channel over Monero with provable security," *IEEE Access*, vol. 9, pp. 31816–31825, 2021.

[43] D. Frkat, R. Annessi, and T. Zseby, "Chainchannels: Private botnet communication over public blockchains," in *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pp. 1244–1252, Halifax, NS, Canada, 2018.

[44] M. Minaei, P. Moreno-Sanchez, and A. Kate, "R3c3: cryptographically secure censorship resistant rendezvous using cryptocurrencies," *Cryptology ePrint Archive*, , Cryptology ePrint Archivep. 454, 2018.

[45] A. Basuki and D. Rosiyadi, "Joint transaction-image steganography for high capacity covert communication," in *2019 International Conference on Computer, Control, Informatics and its Applications (IC3INA)*, pp. 41–46, Tangerang, Indonesia, 2019.

[46] R. Recabarren and B. Carbunar, "Tithonus: a Bitcoin based censorship resilient system," *Proceedings on Privacy Enhancing Technologies*, vol. 2019, no. 1, pp. 68–86, 2019.

[47] N. Alsalami and B. Zhang, "Utilizing public blockchains for censorship-circumvention and IoT communication," in *2019 IEEE Conference on Dependable and Secure Computing (DSC)*, Hangzhou, China, 2019.

[48] X. Luo, P. Zhang, M. Zhang, H. Li, and Q. Cheng, "A novel covert communication method based on Bitcoin transaction," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 4, pp. 2830–2839, 2021.

[49] N. Alsalami and B. Zhang, "Uncontrolled randomness in blockchains: covert bulletin board for illicit activity," in *2020 IEEE/ACM 28th International Symposium on Quality of Service (IWQoS)*, Hang Zhou, China, 2020.

[50] F. Gao, L. Zhu, K. Gai, C. Zhang, and S. Liu, "Achieving a covert channel over an open blockchain network," *IEEE Network*, vol. 34, no. 2, pp. 6–13, 2020.

[51] H. Cao, H. Yin, F. Gao et al., "Chain-based covert data embedding schemes in blockchain," *IEEE Internet of Things Journal*, 2020.

[52] D. Boneh, C. Gentry, and B. Waters, "Collusion resistant broadcast encryption with short ciphertexts and private keys," in *Annual international cryptology conference*, pp. 258–275, Springer, 2005.

[53] S. T. Ali, P. McCorry, P. J. Lee, and F. Hao, "Zombiecoin 2.0: managing next-generation botnets using Bitcoin," *International Journal of Information Security*, vol. 17, no. 4, pp. 411–422, 2018.

[54] M. Brenner, T. Moore, and M. Smith, *Financial Cryptography and Data Security*, Springer, 2014.

[55] M. Abdulaziz, D. Çulha, and A. Yazici, "A decentralized application for secure messaging in a trustless environment," in *International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT)*, Ankara, Turkey, 2018.

[56] L. Zhang, Z. Zhang, Z. Jin, Y. Su, and Z. Wang, "An approach of covert communication based on the Ethereum whisper protocol in blockchain," *International Journal of Intelligent Systems*, vol. 36, no. 2, pp. 962–996, 2021.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 7125472, 13 pages
https://doi.org/10.1155/2022/7125472

Research Article

RBSmix: A Regulatable Privacy-Preserving Method for Cryptocurrency

Rongyu Xiao,[1] **Guozi Sun**,[1,2] **Jiale Yang**,[1] **Yao Wang**,[1] **and Puhe Hao**[1]

[1]*School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing 210023, China*
[2]*Key Laboratory of Urban Land Resources Monitoring and Simulation, MNR, Shenzhen 518000, China*

Correspondence should be addressed to Guozi Sun; sun@njupt.edu.cn

Received 14 January 2022; Accepted 3 May 2022; Published 8 June 2022

Academic Editor: Jinguang Han

Public chains represented by Bitcoin and Ethereum do not require users to use their real names, and transaction data are open to the whole network. Analysed based on this, researchers have achieved the deanonymization of blockchain transactions to a certain extent. Based on the existing blockchain transaction privacy protection scheme, the true link relationship between the transaction sender and receiver is hidden, which brings difficulties to regulation. In this paper, we propose a cryptocurrency mixing service RBSmix, which allows users to reestablish their financial privacy in Bitcoin and related cryptocurrencies. RBSmix, through blind signature to prevent attackers from linking input and output addresses, by the threshold secret sharing algorithm, encryption technology, and a regulation team, combined with the idea of voting, tracks the source of funds for illegal addresses. Experiments show that the scheme scales to large numbers of users and can provide users with better privacy protection.

1. Introduction

Satoshi Nakamoto published the Bitcoin white paper in 2008 [1] and launched the operation of the Bitcoin mainnet in 2009, ushering in a new era of cryptocurrency. Since then, more cryptocurrencies have appeared. The use of Bitcoin does not require personal information, and its anonymity is favoured by more and more people.

However, Bitcoin is not a completely anonymous system. Conti et al. [2] conducted a systematic survey covering Bitcoin's security and privacy and pointed out that the distributed nature of Bitcoin has problems with users' privacy and anonymity requirements. Attackers can obtain some valuable information by analysing transaction records. Reid and Harrigan [3] proposed the concepts of transaction graph and user graph by analysing the transaction characteristics of Bitcoin. Based on their work, researchers proposed some heuristic cluster analysis methods [4–6] to find different accounts of a user. Once a transaction is real named, the user identity information in the relevant transaction will face the risk of disclosure. The research work [7] analysed the possible risks of directly mapping Bitcoin addresses to IP addresses.

Therefore, there is an urgent need to provide users with better anonymous services. Mixing service creates a random mapping between input and output addresses by mixing funds of different users, thereby achieving complete anonymity and enhancing privacy. Bonneau et al. [8] proposed Mixcoin, a scheme based on confused nodes. Mixcoin used multiple nodes to provide mixing services, but if a node divulges privacy, the privacy of the whole protocol will also be destroyed. Liu et al. [9] proposed a scheme based on ring signature, which puts multiple users in the same transaction, and each output address corresponds to the same amount of currency, so that each user is in an anonymous set of fixed size to protect the privacy of user transaction data. However, the size of anonymous collection is limited, and the degree of anonymous protection obtained by users is not high. Ziegeldorf et al. [10] proposed Coinparty, which is based on decryption hybrid network and threshold signature scheme.

It simulates a trusted third party through secure multiparty computing [11] to realize secure and anonymous Bitcoin mixing. Combined the advantages of centralized and decentralized hybrid services in a single system, compared with the previous work, the anonymity has been greatly improved. However, these mixing services only design anonymous protection mechanism and no regulation mechanism, so that mixing services are often used for illegal activities, such as black-market trading and money laundering. On September 26, 2020, KuCoin issued an official announcement [12] that the platform was attacked by hackers. Information from the blockchain analysis company Elliptic showed that the hackers who stole from KuCoin used different mixing services to launder money. Unregulated mix technologies may cause harm to the society, and it is difficult for law enforcement agencies to trace down and punish it, so that many countries in the world are cautious about cryptocurrencies. In 2018, the U.S. Securities and Exchange Commission (SEC) issued the "Statement on Digital Asset Securities Issuance and Trading" [13], confirming that digital assets belong to the category of securities and must be included in the national regulatory system. Russia, Vietnam, etc. strictly restrict the circulation of cryptocurrencies. Therefore, it is of great significance to design a mixing scheme with both privacy protection and regulation functions.

Aiming at the problems existing in the existing schemes, this paper focuses on solving the problem of unregulated mixing server. This paper proposes a new protocol RBSmix (Regulatory, Blind, Shamir) with both privacy protection and regulation functions. Our contributions are summarized as follows:

(1) RBSmix designed a regulation team to solve the problem of unregulated mixing services. If more than half of the regulation members think it is illegal, they can link the address to its input address through calculation to trace the source of funds at the address

(2) RBSmix will encrypt and divide the transaction data and submit it to multiple institutions for storage. The data leakage of a few institutions will not lead to the destruction of users' transaction privacy and reduce the risk of data leakage

(3) RBSmix divides the user's request for mixing service into sending currency and withdrawing currency, and the two transactions of different users are interspersed. With the increase of the number of users requesting services, the anonymous set size of each user will also increase, realizing better anonymous protection services

The rest of this paper is organized as follows. We introduce some research work on protecting the privacy of blockchain transactions in Section 2. The technology related to this paper are presented in Section 3. The protocol and system design of RBSmix is laid out in Section 4. We provide a comprehensive analysis, evaluation, and discussion of correctness, performance, and security in Section 5. Section 6 concludes this paper.

2. Related Work

To protect the privacy of blockchain transactions and hide both parties and transaction amount, researchers have proposed many schemes.

Valenta and Rowan [14] improved the Mixcoin and proposed the Blindcoin, which protects users' privacy through a blind signature algorithm and a public log. Mix servers cannot obtain the real link relationship between the input and output addresses. However, due to the lack of regulatory mechanism, there is a risk of capital theft in both schemes. Bao et al. [15] proposed Lockmix based on Mixcoin and Blindcoin. The original scheme is further optimized through blind signature and multisignature, which can not only prevent the mix server from obtaining the real link relationship between the input and output addresses but also prevent the mix server from illegally stealing user funds.

Zerocoin [16] is a new type of side chain, which is an encrypted extension of Bitcoin. It uses zero-knowledge proof technology to convert users' Bitcoins into zero coins on the Bitcoin side chain to achieve stronger anonymity. Zerocoin once proposed a soft fork, but it was rejected by Bitcoin developers. Later, Sasson et al. [17] improved Zerocoin and proposed the Zerocash scheme; noninteractive zero-knowledge proofs are used to achieve stronger anonymity.

Coinjoin proposed by Maxwell [18] allows mixing service transactions without the intervention of a third party. In the Coinjoin transaction, multiple users take the same amount of Bitcoin as input to construct the transaction and check whether their address is written into the transaction. Only after obtaining the signatures of all users and merging, the transaction is considered legal and received by the network. However, Coinjoin also has some defects, such as being unable to resist DoS attacks, and users cannot deny that they have participated in mixing service. Aiming at the defects of Coinjoin mechanism, Ruffing et al. [19] proposed Coinshuffle, added the mechanism of shuffling the output addresses on the basis of Coinjoin, and realized the internal unlinkability, but it was vulnerable to DoS attack and cross attack [20]. Barber et al. [21] proposed fair exchange protocol, a bilateral Bitcoin exchange protocol, both parties use Bitcoin scripts, and three types of Bitcoin transactions (guarantee transaction, refund transaction, and claim transaction) realize Bitcoin exchange without mutual trust. Bissias et al. [22] proposed a method of anonymously finding hybrid nodes based on blockchain advertising. With the increase of the number of users, the cost of attack will increase, which can effectively avoid denial of service attack.

Sun et al. [23] proposed an MBDC model suitable for the central bank to regulate digital currency. The central bank can avoid the double flower problem and protect users' privacy by separating users' identity and transaction information. The establishment of DC (data center) and layers of supervision realized strong regulation on the model. Xue et al. [24] proposed a blockchain transaction model with both privacy and regulation functions. Probabilistic encryption is used to hide the true identity of blockchain transactions; commitment scheme and zero-knowledge proof

technology are used to protect privacy. Through encryption technology, regulators can store user information without storage regulating blockchain transactions without interest. Tu and Meredith [25] have studied the regulation of virtual currency and proposed to develop a comprehensive, cohesive, and appropriate scale virtual currency regulation model in combination with existing laws.

3. Preliminaries

This section will introduce blockchain transaction, blind signature, threshold secret sharing, Tor, etc., because they are related to the scheme in this paper.

3.1. Blockchain Transactions. Blockchain is a decentralized distributed ledger that records all transaction data, including transaction hash value, time, input and output address, and amount. The input address of each transaction is the output address of the previous transaction, and all funds can be traced back to the source. All transaction data in the blockchain are open to the whole network, and anyone can obtain complete transaction records. Through analysis, attackers may associate an anonymous address with a specific user, destroying user privacy. This is also the problem to be solved in this paper.

3.2. Blind Signature. Blind signature [26] is a special digital signature, which is a bilateral agreement. In blind signature, the signer does not know the specific content of the signed message, and the message owner can get the signer's digital signature of the original message. When the signature is made public, the signer cannot know when it was signed. An ideal blind signature protocol has the characteristics of unforgeability, nonrepudiation, unknown, and untraceable. Blind signature can achieve the purpose of protecting data privacy.

3.3. Threshold Secret Sharing. The concept of secret sharing was first independently proposed by famous cryptologists Shamir [27] and Blakley [28] in 1979. (k, n) threshold secret sharing means that the secret s is decomposed into n subkeys and distributed to n holders, in which no less than k secrets can recover the ciphertext, and no information of the ciphertext can be obtained if less than k secrets. Threshold secret sharing can distribute power to multiple independent users without relying on single point of management.

3.4. Tor Network. Tor [29, 30], known as onion routing anonymous proxy network, is a typical anonymous communication system. It is mainly used for privacy protection in IP layer to prevent attackers from attacking users' traffic and ensure online anonymity of users. Rerouting mechanism plays the role of confusing the actors of network behaviour, making it possible for all nodes in Tor network to participate in network behaviour. Hierarchical encryption erases the relationship between messages received and sent by nodes, that is, all nodes seem to be doing the same network behaviour. The combination of the two realizes the anonymity of network behaviour.

4. RBSmix Protocol

Based on blind signature algorithm, threshold secret sharing algorithm, and encryption technology, we propose a regulatable mixing service protocol RBSmix, which not only provides users with transaction privacy protection services but also realizes the regulation of illegal transactions. Next, we will introduce in detail.

4.1. System Model. The RBSmix model is shown in Figure 1, which mainly includes a central agency, a regulation team, mix servers, and users.

4.1.1. Central Agency. With certification and regulation functions, mix servers or users need to register with the central agency and take the lead in completing regulatory tasks when necessary.

4.1.2. Regulation Team. Assist the central agency to complete the regulation function, and prevent the central agency and mix servers from negotiating privately and undermining user privacy. Regulation members can also be used as ordinary mix server to provide users with mixing services. Team members are selected based on indicators such as the number of users of the mix server service and the amount of funds. If the number of members is too small, the degree of authority will be lower. Too many, it is difficult to guarantee the quality of members. It needs to be set according to the actual situation. This paper is for experimental testing only, and six members are tentatively scheduled.

4.1.3. Mix Servers. The number is variable, providing users with mixing services. It is required to register with the central agency and pay a certain deposit to ensure that it will not illegally violate the provisions of the protocol.

4.1.4. Users. Mainly divided into two categories: the first category is to use fiat currency to purchase cryptocurrency [31], and hope that anyone, including mix servers, cannot associate the receiving address with their own identity. The second category is to transfer cryptocurrency from an address (which may have a certain connection with his identity) to a new address with better anonymity.

As shown in Figure 2, usually, the mix server receives the user's funds, mixes them, and then, sends the funds to the address specified by the user. In the early days, the centralized mix server could know the correspondence between the input and output address. Users cannot judge whether the server will leak or sell their private information. To solve this problem, researchers later proposed some solutions based on blind signatures and ring signatures. Users have got better privacy protection, but they also bring difficulties to regulation.

RBSmix is based on a centralized server that uses a blind signature algorithm to cut the link between the input and output addresses. Adopt threshold secret sharing algorithm and encryption technology to realize the regulation of transactions. Tor protects user privacy at the IP layer.

The protocol is mainly divided into the following steps:

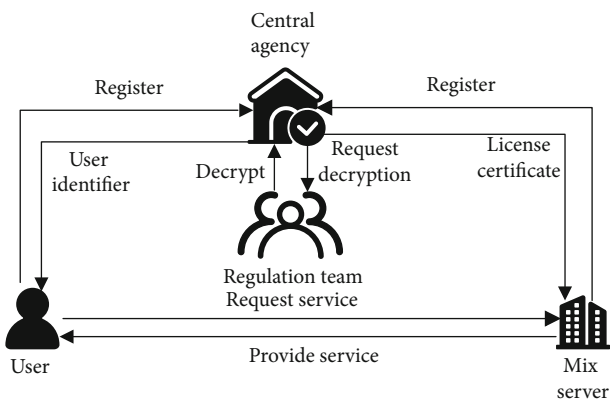


Figure 1: System model.

(1) User requests the central agency to sign his identity information

(2) User requests the mix server to sign the address and identification information through the signature of the central agency

(3) User relies on the signature of the mix server to request the withdrawal service from the mix server through the Tor anonymous network

(4) When necessary, the central agency requests transaction information from the mix server and submits it to the regulation team for voting to realize the regulation function

4.2. Protocol Goals. As a mixing service, we propose the following goals for RBSmix:

(1) *Anonymity Protection Service*. Users can transfer funds to addresses with better privacy in an unlinkable way through RBSmix

(2) *Better Anonymity*. The size of the anonymity set implemented by most of the current mixing services is limited. We hope to provide users with better privacy protection, so it is required that the anonymity set provided by RBSmix for users can become larger as the number of users increases

(3) *Antitheft*. RBSmix is implemented based on a centralized mix server, and we need to avoid mix server from stealing user funds

(4) *Defend against Malicious User Attacks*. If a malicious user attacks RBSmix, will he affect other users requesting the service, and whether the attacker can carry out a DDoS attack. This is the problem we need to solve

(5) *Deniability*. It is required that no one except the mix server can judge which transactions are mixed transactions from the transaction form. Because if the transaction form of the mixed transaction is significantly different from the normal transaction, even if the specific address correspondence is not known, the scope of the user anonymity set can be narrowed, and the risk of user privacy leakage can be increased

(6) *Monitorability*. Design a regulatory mechanism to avoid being used by illegal elements for illegal activities such as money laundering

The scheme in this paper is based on elliptic curve encryption algorithm, a blind signature algorithm based on elliptic curve [32] and Shamir secret sharing algorithm [27]. Therefore, the security of RBSmix is based on the security of these three algorithms. However, the RBSmix model may still face some other security threats. For this, we propose the following security goals:

(1) In RBSmix, the regulation function is realized through the signature information of the user identifier. We need to ensure that even if the attacker steals the signature information of the legitimate user, he cannot use the information to request the mixing service to evade regulation

(2) In RBSmix, the user needs to transfer money to the address specified by the mix service. We need to ensure that the attacker cannot tamper with the address to his own address during the communication process to deceive the user; the mix service also cannot deny that the address is its own after receiving the user's transfer

(3) In RBSmix, users need to use their own identifier-related information to request mixing service. We need to ensure that attackers cannot use this information for cluster analysis to destroy user privacy

4.3. Protocol Process. The specific process of the RBSmix protocol is shown in Figure 3. The parameter table is shown in Table 1. Next, each step will be described in detail.

(1) User **A** wants to request mixing service. First, find the cryptocurrency that meets his requirements and the mix server **M** that provides the service from the public data provided by the central agency. Then, **A** and **M** negotiate the amount v and service fee vp and confirm the number of blocks required for the transaction ω

(2) After the negotiation is completed, **M** sends a consent reply to **A**

(3) **A** receives the message from **M** and performs the following operations:

  (i) **A** uses a blind signature algorithm [32] based on elliptic curve to generate a blind factor r_1, which is used to blind the information later

  (ii) **A** chooses 5 of the 6 regulation team members as the regulation team members for this service. If the mix server of **A** requesting service is one of the 6 team members, he can only choose the remaining 5 as members of this

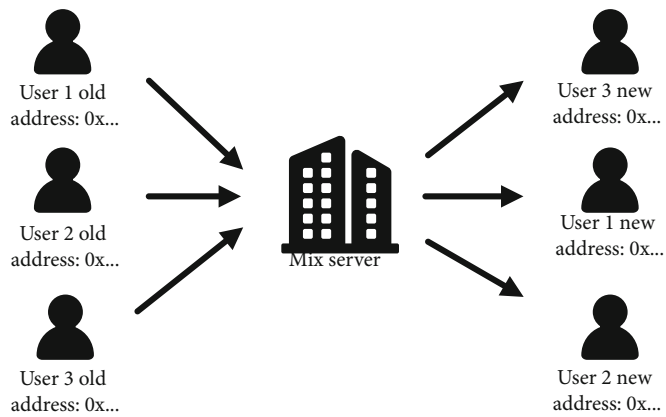


Figure 2: Centralized mix server model.

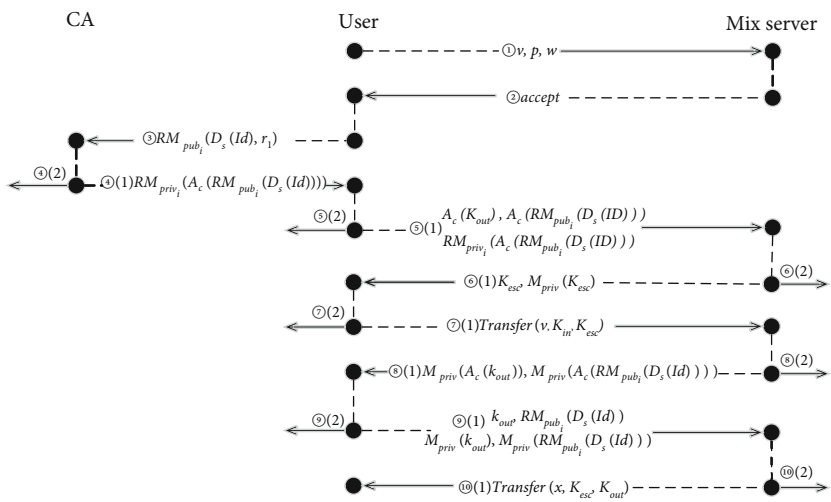


Figure 3: Protocol diagram.

service. The reasons will be explained in Section 4.3

(iii) **A** uses a threshold secret sharing algorithm to divide its identifier into 5 parts. Add KG_M, t_1, t_2 after each part, then the information is encrypted with the public keys of 5 team members and send them to the central agency

(4) The central agency receives the message from **A**, decrypts it, and then, sends the corresponding fragment information to the corresponding regulation member. The regulation member receives the message, decrypts it, and sends the obtained identifier information to the central agency. The central agency chooses 3 of the 5 fragments to verify whether the identifier is legal. All combinations need to be verified for a total of 10 rounds

(i) If all are successful, inform 5 regulation members that the verification is successful. After receiving the successful message, regulation members encrypt the identifier fragment with their own public keys and then blind the message with r_1. Then, sign the information with his own private key and send it to the central agency. After receiving it, the central agency will uniformly send it to user **A**

(ii) If there is a failure, inform the regulation member that there is an error and the protocol stops. Then, send a verification failure reply to **A**

(5) User **A** receives the message and verifies the signature. For the five parts of the identifier division in step 3 of process 3, the corresponding regulation member public key is used for encryption, and then, the same blind factor r_1 is used to blind the message. Five signatures are verified by using the public keys of five regulation members

(i) After verification, **A** sends a mixing service request to the mix server. First, generate the address k_{out} which **A** wants to receive cryptocurrency, and generate a new blind factor r_2 to blind the address. Finally, send the blinded address, blinded information containing the identifier generated in step 3 of process 3, the

Table 1: Parameter table.

Parameters	Description
p	The mixing fee rate that **A** will pay
v	Amount of **A** need to be mixed
x	Coin value withdrawal allowed by **M**
ω	The number of blocks which is required to confirm payment
Id	Identifier of **A**
r_i	Blinding factor (the general name of the factors used in blind processing)
k_{in}	The input address of **A**
k_{out}	The output address of **A**
k_{esc}	The escrow address of **M**
k'_{esc}	Another escrow address of **M**
CA_{pub}	The public key of CA
CA_{priv}	The private key of CA
RM_{pub_i}	The public key of RM_i
RM_{priv_i}	The private key of RM_i
M_{pub}	The public key of **M**
M_{priv}	The private key of **M**
A_c	Blinding function
A'_c	The inverse of A_c
D_s	Secret distribution
M_s	Secret reconstruction

signature information of the five regulation members received from the central agency, and which five members are chosen to the mix server

(ii) Verification failed: check whether there are any errors in your operation and restart process 3

(6) The mix server receives the request sent by **A** and verifies the signature. Get the blinded information containing the identifier and the signature information containing the identifier, and verify the signature by using the public key of five regulation members

(i) If the verification is successful, send the address k_{esc} that is going to receive **A**'s cryptocurrency and the signature $RM_{\text{priv}_i}(k_{\text{esc}})$ of the address to **A**

(ii) Failed to verify, reply to user **A** failed

(7) **A** receives the message from the mix server

(i) If successful, **A** needs to transfer the cryptocurrency amount v negotiated in the process 1 to the address k_{esc} designated by **M**. And send the transaction information to **M**

(ii) If it fails, **A** checks whether there is an error in his previous operation and reexecutes process 3 or process 5

(8) After receiving the reply from **A**, **M** checks whether it has received the negotiated amount of encrypted currency at the address specified by itself (for the second type of users, the mix server also needs to determine whether the received signature about the address is the signature of the central agency on the address sent by the user this time)

(i) Confirm receipt, wait for ω block confirmation, and sign the received blind address $A_c(k_{\text{out}})$ and $A_c(RM_{\text{pub}_i}(D_s(Id)))$ containing identifier. Send them to **A**

(ii) If it is not received or the amount is incorrect, reply with an error message

(9) **A** receives the information replied by **M**, verifies the signature, and withdraws the coin

(i) Reply as a signed message, and the signature is valid. Save the signature. **A** can anonymously send the original address k_{out}, $D_s(Id)$ containing the identifier generated in step 3 of process 3, and the received signature of the mix server to **M** through the Tor network at any time to initiate a withdrawal request

(ii) If the signature verification fails or an error message reply is received, check the reason. If it is **M**'s problem, you can appeal to the central agency by virtue of k_{esc} and $M_{\text{priv}}(k_{\text{esc}})$ received in process 6 and the transfer record to the address

(a) The central agency receives the appeal information and checks the correctness of the information

(b) If the information is correct, draw the transaction amount from the deposit paid by the mix server and transfer it to user **A**. At the same time, a part is taken as the punishment for the mix server's violation of the agreement

(c) If there is an error in the information, inform **A** that there is a problem with the appeal, and recheck the information

(10) The mix server receives an anonymous withdrawal request and verifies the validity of the signature

(i) Valid, agree to the request, and transfer coins to this address. At the same time, the request information $k_{\text{out}}, RM_{\text{pub}_i}(D_s(Id))$ is stored locally

(ii) Invalid, discard the request

(11) After a while, **A** checks whether the specified receiving addresses receives the specified number of currencies

(i) If a specified amount of currency is received, the entire mixing service is completed

(ii) Otherwise, **A** reinitiates the withdrawal request. If **A** still cannot receive the currency after multiple requests, he can appeal to the central agency with the signature information received in process 9. Same as step 2 of process 9

4.4. Regulation Process. In the future, if someone proposes that an address served by the mix server is involved in illegal and criminal activities, the central agency needs to organize regulation team to vote to decide whether to undermine the privacy of the address and trace the address of the source of funds. The regulation model of RBSmix is shown in Figure 4; the specific process is as follows:

(1) User **B** believes that an address K_{out} is involved in illegal activities and puts forward his views to the central agency

(2) The central agency accepts the request and requests the secret information containing the identifier corresponding to the address from the mix server who has served the address

(3) The mix server finds the corresponding information $k_{\text{out}}, RM_{\text{pub}_i}(D_s(Id))$ and sends it to the central agency

(4) The central agency distributes the relevant secret fragment information to the corresponding regulation members and organizes the regulation team to discuss whether the address behaviour is indeed illegal

(5) After receiving the information, the regulation members first judge whether the address is indeed involved in illegal and criminal activities, and if so, decrypt it to obtain $D_s(Id)$. The message is then sent to the central agency

If **A** chooses one of the six regulation members as the mix server and also chooses it as the regulation member to save one of the five subkeys that can decrypt the identifier information, the regulation member finds that one of the five signatures is his signature when serving the user and verifying the five signatures, and can directly pair and store the fragment information with the user locally. Then, the mix server can find the input-output link relationship of the service address alone. Therefore, in step 3 of process 3, users must choose mix servers and regulation members according to regulations.

(6) The central agency receives the reply from the regulation member

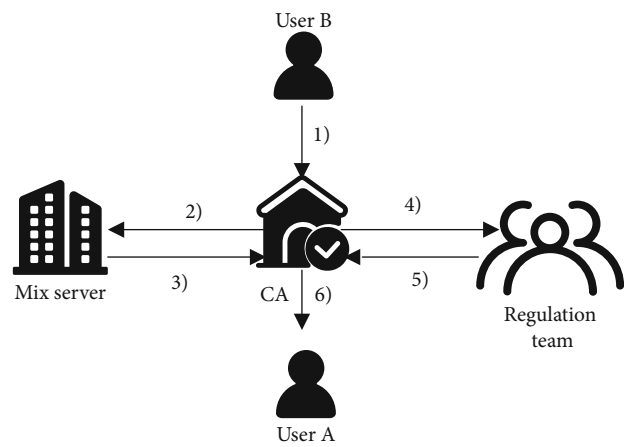


Figure 4: Regulation model.

(i) If three or more regulation members believe that the address is indeed illegal and agree to decrypt it, the central agency can use the information to calculate user identifier and find the source address of the address funds

(ii) If less than 3 regulation members believe that the address is illegal and do not agree to the decryption, the central agency will not be able to decrypt the user identifier information

5. Experiment and Analysis

In this section, the effectiveness of the scheme will be verified through experiments, and then, the various attributes of RBSmix will be discussed through comparison with other schemes. Finally, the scheme will be analyzed and the possible security threats will be discussed.

5.1. Experiment. The main purpose of our experiment was to measure the efficacy of RBSmix. The experimental environment is a virtual machine with 4-core 4G and running 64-bit Ubuntu 18 (the physical machine with an Intel Core i7-8750H CPU at 2.20 GHz 2021 GHz with 16 G of RAM, and running 64-bit Windows 10). The method proposed in this paper adopts Python 3 7.6 language design and implementation: the program designs a central agency and six mix servers. The mix servers act as members of the regulation team at the same time, and different institutions use different ports of the virtual machine for simulation implementation. Experiments were conducted on the Ethereum test chain Ropsten and the local Bitcoin test chain built using the open-source project bitcoin-testnet-box [33].

We have conducted many experiments. To make the experimental results easier to observe, we show from the perspective of a mix server the situation of mixing services for 6 users on the Bitcoin test chain built locally, involving a total of 13 transactions, as shown in Table 2.

People other than users and mix server cannot know who sent each transaction, let alone determine the relationship between the user's input addresses and the receiving

Table 2: Transaction records.

Transaction	Inputs	Outputs	Value [BTC]
Transaction 1	User 1	bcrt1qcmsjawreuk80as8g2fl7yaqs7kfrpm3dqyf8tf	1
Transaction 2	User 2	bcrt1qurndyuql97thg0ycfz8g9ecy29htktg2upzh7r	1
Transaction 3	Mix server	bcrt1q4yypg9amgzd2uzrr055em5frhwwj9j3pceyd7t	1
Transaction 4	Mix server	bcrt1q9jw04nx7gryrjhyud693zkkygyzp2t4ysv2chl	1
Transaction 5	User 3	bcrt1qpamtz0kpcdyza40pufjrkl0u2dn8dx45qv7w9p	1
Transaction 6	Mix server	bcrt1q7rls7z6kx5x98j2s0hjjk0j6r2skwpy64myxtm	1
Transaction 7	User 4	bcrt1qurndyuql97thg0ycfz8g9ecy29htktg2upzh7r	1
Transaction 8	User 5	bcrt1qpamtz0kpcdyza40pufjrkl0u2dn8dx45qv7w9p	2
Transaction 9	User 6	bcrt1qcmsjawreuk80as8g2fl7yaqs7kfrpm3dqyf8tf	1
Transaction10	Mix server	bcrt1qgam5smv0k2750mczczx4hhhjs0j23rpvrmj4hw	1
Transaction 11	Mix server	bcrt1qy2jdydcx565aekrd7n3m79fzpvw3zd7yrgwdc6	1
Transaction 12	Mix server	bcrt1q8weae9fsvsgepgaaakqcrtrt9qgexulu4x9std	1
Transaction 13	Mix server	bcrt1qv4gzdx25crdwerztqyjussjh608z2yusdtzmh4	1

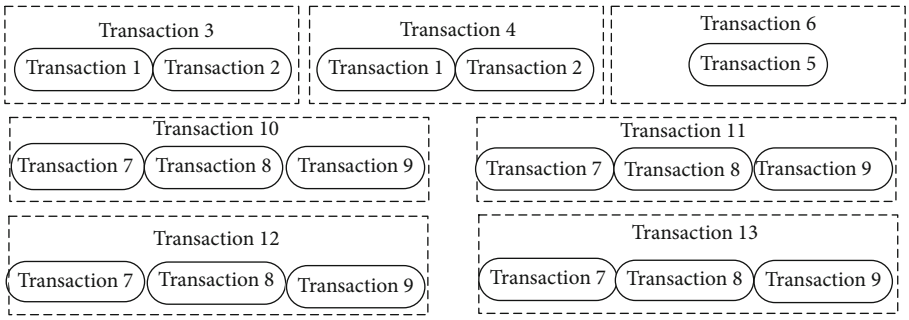


Figure 5: Anonymous set corresponding to each withdrawal transaction.

addresses, or even whether these transactions are mixed transactions. In this case, users can even deny that they have participated in a mixed transaction.

The mix server knows which are the addresses in its address pool and which are user's addresses, but cannot associate user's receiving addresses with the input addresses. As shown in the third transaction in Table 2, the mix server can determine that the address bcrt1q4yypg9amgzd2uzrr055em5frhwwj9j3pceyd7t is the receiving address of one of user 1 and user 2, but cannot determine which one it is. Because in its view, all withdrawal transactions are the same.

The size of the anonymity set of each withdrawal exchange in Table 2 is shown in Figure 5. It can be seen that the size of the anonymous set of the sixth transaction (belonging to the withdrawal transaction) is one. This is because the user who initiated the mixing service request has already withdrawn the coin. For the address bcrt1q7rls7z6kx5x98j2s0hjjk0j6r2skwpy64myxtm, the coin only may come from user 3. In this case, the user did not obtain the expected anonymity. But for other transactions, the anonymity set is greater than 1, for example, the 10th transaction, the coin of the address bcrt1qgam5smv0k2750mczczx4hhhjs0j23rpvrmj4hw may come from 3 addresses, and the anonymity set size is 3, achieving the purpose of mixing service.

The size of the anonymity set at the withdrawal exchange is the number of users who have not requested the withdrawal service before the withdrawal request is initiated. In the operation of the protocol, as the number of users requesting services gradually increases, the number of users that may correspond to each withdrawal exchange also increases, and the better the anonymity obtained. Therefore, when a user initiates a withdrawal request, he can pay attention to the number of users requesting mixing service, so as to put himself in a larger anonymous concentration and obtain better privacy protection.

Then, we used a parallel strategy to simulate multiple users' requests for mixing services to verify the effectiveness of the scheme. Test the time required for RBSmix to serve different numbers of users without considering transaction confirmation. Firstly, the time required for different number of mix servers to serve a large number of users is tested. As shown in Figure 6, it can be seen that it takes about 14 s for a mix server to serve 10 users, 144 s for 100 users, and 1470 s for 1000 users. There is a linear relationship between the number of users and the running time as a whole. When the number of mix servers increases, the service time decreases. The time required for two mix servers to serve users is about half that of one mix server, but when the number of mix servers becomes three, the time reduction is not obvious, because the service performance of the central agency has not been improved accordingly. Then, the program of the central agency is changed to dual-thread parallel processing, the number of mix servers is set to 4, and the experiment is carried out again. The experimental results

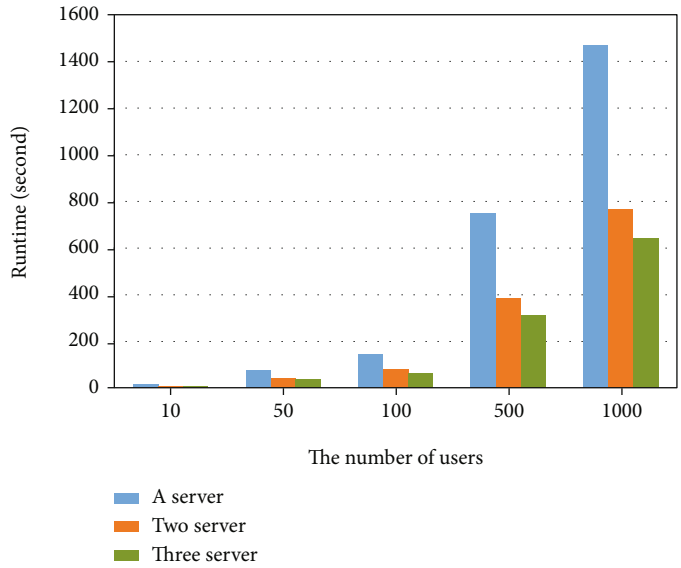


Figure 6: Running time in different numbers of users.

are shown in Figure 7. For the same number of users, compared with single thread processing, the performance of the central agency with dual thread parallel processing has been greatly improved.

Generally speaking, the running time of RBSmix increases with the increase of the number of users and decreases with the improvement of the number of mix servers and the service capacity of the central agency. Compared with the transaction confirmation time in the running process, the protocol process takes up very little time.

Then, we verified the regulation function. To realize the regulatory function in the protocol, the members of the regulation team need to vote. To verify the effectiveness of the regulatory function in the experiment, we assume that each time the central institution organizes a vote, most members will vote in favour, so that, each time, it can decrypt and complete the tracing of the source addresses of the addresses. Figure 8 shows the time required to complete different times of regulatory experiments. It can be seen that the running time of the program is very short after completing traceability of the source of address funds. In the actual operation of the agreement, most of the time required for regulation is the time when the central agency requests data from the mix server and the time for the members of the regulation team to vote. Code running time is very small.

It can be seen from the experimental results that compared with the transaction confirmation time, the time to complete encryption, decryption, signature, and verification is very short. In addition, multiple mix servers can serve a large number of users at the same time, and the scalability of the scheme is also relatively good.

5.2. Protocol Analysis. This section analyses the properties of the RBSmix protocol. The specific analysis is as follows:

5.2.1. Anonymity. Using the blind signature algorithm, the mix server cannot correspond to the user and the user's receiving address, but only knows which addresses it has served. In order to resist the damage to the privacy of transactions caused by the amount-based disambiguation method [34], the withdrawal amount of services provided by the same mix server to users must be the same. As shown in the experiment in Figure 5, each user is mapped to a different number of receiving addresses. Users can control the size of their anonymous set by adjusting the withdrawal time, because users who have requested services from the mix server before the withdrawal request is initiated may initiate the withdrawal request; the anonymous assembly gradually becomes larger with the increase of users. Ideally, the anonymous set can be as large as the total number of users.

5.2.2. Antitheft and Accountability. RBSmix requires mix servers to register with the central agency and pay a certain deposit. If the user requests the withdrawal service, the mix server refuses to transfer money in violation of the protocol and steals the user's funds; user can appeal to the central agency by the transfer records and the mix server's signature on the address; once it is determined that the appeal is successful, the central agency will send part of the mix server's deposit to the user to make up for the loss of the user and draw part as the punishment of the mix server. During the operation of the RBSmix, violations of the protocol may occur at every step. After the negotiation is completed, the service provider may refuse the service, affecting the user's service experience, and the user may refuse to request the service, resulting in the waste of service resources of the mix server. In these cases, they can appeal to the central agency. After the appeal is successful, part of the deposit will be deducted from the mix service's illegal agreement. Users who violate the protocol will not be able to request mixing services for a certain period. The loss of violating the protocol is far greater than the benefits obtained. In theory, neither side will do so.

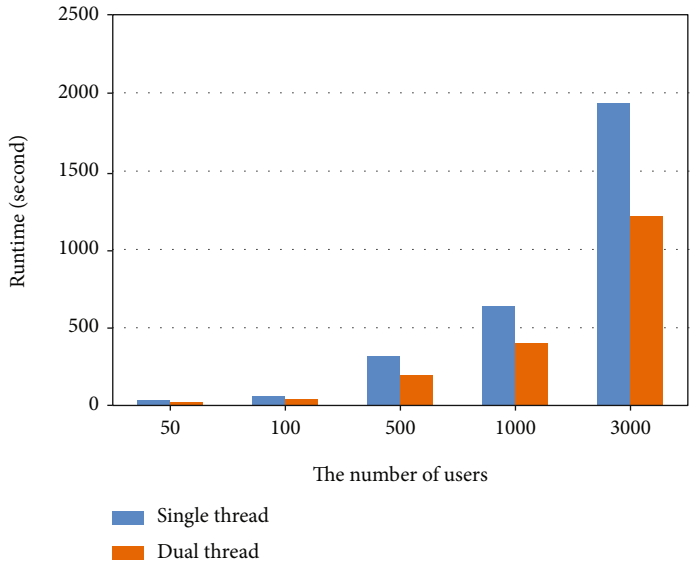


Figure 7: Running time in different CA.

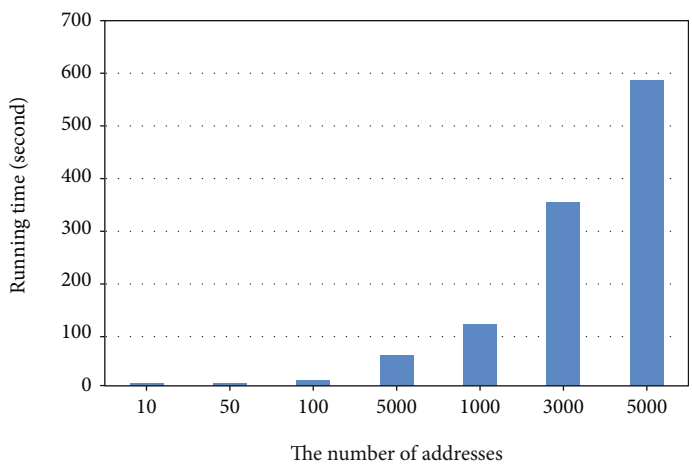


Figure 8: Running time of regulation function.

5.2.3. Scalability. According to the experimental results in Section 5.1, RBSmix can serve a large number of users at the same time, and with the increase of the number of users, the anonymity of users is better.

5.2.4. Costs. During the operation of RBSmix, a total of two transactions are required. Therefore, the cost is mixing service fee and the cost of two transactions.

5.2.5. Compatibility. The whole protocol runs independently of the cryptocurrency system and does not need to change the existing system. The transaction part of the scheme can be completed by automatic script or user manually. If an automated script is used, you need to add a call script related to the wallet in the scheme. To test the performance of RBSmix, in the experimental part of the local Bitcoin test chain, all transactions are completed through scripts. In the Ropsten experiment part of Ethereum test chain, users manually confirm the transaction. We recommend manual trading because it is more secure.

5.2.6. Resilience to DoS Attack. Attackers can carry out two kinds of attacks, malicious termination of the protocol or DoS attack. The mix server provides services for each user separately, and users do not affect each other. Even if the attacker maliciously terminates the agreement, it will not have any impact on other users. RBSmix is different from Coinjoin. Users who request mixing services need to pay corresponding service fees. If an attacker implements a DoS attack, it requires a great economic cost. Therefore, rational attackers would not use this attack method.

5.2.7. Deniability. The existing partial coin mixing schemes are to put the input and output addresses of multiple users in the same transaction; to resist the amount-based unmixing method, the output amount will generally be the same. This transaction form can be easily identified as mixed transaction. For example, Meiklejohn and Orlandi proposed an imprecise method [35] to identify Coinjoin [18] transactions, believing that transactions with more than 5 inputs

Table 3: Comparison of the properties with other schemes.

Approach	Accountability	Scalability	Costs	Compatibility	Resistant to DoS attacks	Deniability	Monitorability
RBSmix	√	√	$vp + 2tx$	√	√	√	√
Mixcoin [8]	√	√	$vp + 2tx$	√	√	√	×
Blindcoin [14]	√	√	$vp + 2tx$	√	√	√	×
Coinjoin [18]	×	√	tx	√	×	×	×
Coinswap [36]	×	×	$2tx$	√	√	×	×
Zerocoin [16]	×	√	$2tx$	×	√	√	×
Zerocash [17]	×	√	$2tx$	×	√	√	×

and more than outputs are mixed transactions. In this scheme, withdrawal transactions are transferred from one address (the address of the mix server) to one address (the receiving address specified by the user), which is no different from the normal transaction form. Therefore, it is impossible to judge whether the transaction is a mixed transaction from the transaction form alone.

5.2.8. Monitorability. The threshold secret sharing algorithm and the regulation team are used to regulate the transaction. Once someone proposes that an address is involved in illegal activities, the source of funds of the address can be found through the mix server, the central agency, and most regulation members. A voting idea is adopted here. When there is a problem, only the central agency or mix server cannot complete the regulation function alone. The central agency needs to send the secret fragments related to the address to the corresponding regulation members. The members first make their own judgment on the address and then decide whether to decrypt the information. Only when most regulation members agree to decrypt, the central agency can calculate the original secret information and complete the regulation function. This avoids the malicious destruction of user privacy to a certain extent.

There can be multiple mix servers in RBSmix, which can provide different services to meet users' different needs for cryptocurrency types, amount, etc. Compared with those of other mainstream schemes, this scheme has the basic characteristics of mixing service and realizes the regulation function, as shown in Table 3.

However, compared with the existing schemes, the scheme proposed in this paper has some limitations. The scheme needs an authoritative organization to cooperate with multiple institutions for implementation, so that the implementation cost is higher than the existing schemes (some existing schemes only need one mix server. Some solutions do not even need a mix server, and only need to negotiate between users to achieve the purpose of currency mixing). The premise for a user to reestablish financial privacy is that the user's request for services intersects with that of other users; otherwise, the user cannot reestablish privacy.

Chiu et al. [37] design a scheme that uses smart contract and blockchain to provide a secure data sharing and access environment. Smart contract can better control data, protect user data privacy, better ensure the legal operation of the protocol, and avoid many problems. For example, the user's behavior is controlled through the smart contract, and the request for withdrawal can be initiated only when the user's request intersects with that of other users, which can ensure that users can rebuild their financial privacy through this scheme. In the future, we will try to use smart contract for design and implementation to further improve our scheme.

5.3. Security Analysis. In Section 4.2, we propose the security threats that RBSmix may face, which we will analyse in this section.

Signature theft attack: if the attacker intercepts the signature $RM_{\text{priv}_i}(A_c(RM_{\text{pub}_i}(D_s(Id))))$ sent by mix server **M** to **A**, can the signature be used to request mixing service? First, RBSmix requires the communication process to use the other party's public key for encryption when sending information. When receiving the information, decrypt it with **A**'s private key. As long as **A** ensures the security of his private key, the communication information will not be stolen when the encryption algorithm is secure. If the attacker illegally and successfully obtains the decrypted signature in some way, but because he does not know the blind factor and all identifier fragment information, he cannot successfully construct the blind information $A_c(RM_{\text{pub}_i}(D_s(Id)))$ and still cannot legally initiate the request. If the attacker successfully obtains the signature $RM_{\text{priv}_i}(A_c(RM_{\text{pub}_i}(D_s(Id))))$ and blind information $A_c(RM_{\text{pub}_i}(D_s(Id)))$ after **A** initiates a complete mixing service request, in this case, the attacker can legally initiate the mixing service request and successfully obtain the signature of the mix server. However, because he does not know the blind factor, he cannot unblind the signature and cannot legally initiate the withdrawal request. Still unable to attack successfully. Therefore, as long as user guarantees the security of the private key and blind factor, on the premise of the security of the encryption algorithm, the attacker cannot steal the signature of other users to request mixing service.

Address tampering attack and address denial attack: in protocol process 6, the mix server sends an address k_{esc} to user and asks user to transfer the funds to the address. Consider two attack methods: (1) after intercepting the information, the attacker tampers the address into his address, making user transfer the funds to his own address to realize the attack; (2) after receiving the transfer, the mix server does not recognize that the address is its own address and

falsely claims that the address has been tampered with in the process of communication. These two attacks cannot be solved only by using encrypted communication. This scheme requires the mix server to add the mix server's signature $M_{\text{priv}}(k_{\text{esc}})$ to the address when sending this part of the information. After receiving the information and verifying the signature, user transfers the funds to the address. In this way, the mix server cannot deny its signature, and the attacker cannot forge the mix server's signature to ensure the security of user transfers.

User identifier information clustering attack: when users request mixing services from mix servers, they need to segment their personal identifiers to obtain $D_s(Id)$ and then request the signature $A_c(RM_{\text{pub}_i}(D_s(Id)))$ of the regulation member from the central agency. When withdrawing, they need to carry $RM_{\text{pub}_i}(D_s(Id))$. If user requests a signature from the central agency, and then frequently uses the signature to request mixing services, the signature information can be regarded as an account, and all mixed addresses related to the "account" can be found by clustering method. Once one of the addresses is real named, all mixed addresses related to it will face the risk of privacy disclosure. If user uses threshold secret sharing to divide the information and requests a signature from the central agency before requesting mixing service each time, it can ensure that the signature information used each time is different. For mix server, even if a user continuously requests mixing services, the personal information and address information used each time are different. Mix server can only regard them as different users to achieve stronger anonymity protection.

In summary, RBSmix can resist these potential attacks and provide users with safe and reliable mixing service.

6. Conclusion

This paper analyses the privacy issues of cryptocurrency transactions and proposes a regulatable mix server scheme. This scheme has the basic ideal properties of mixing service, realizes the accountability of the agreement when necessary, and provides a new solution to the conflict between the anonymity and regulation of cryptocurrency. In addition, the solution does not depend on a specific consensus mechanism and can be used as an independent module. Finally, experiments verify that multiple mix servers in the protocol can efficiently serve a large number of users at the same time, then several attack methods are proposed, and the security of this scheme is proved, and users can obtain better privacy protection through this scheme.

This paper also discusses some limitations of RBSmix and possible improvements, such as design and implementation through smart contract. In the future, we will continue to try to improve RBSmix to achieve better results.

Data Availability

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

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (no. 61906099), the Open Fund of Key Laboratory of Urban Land Resources Monitoring and Simulation, Ministry of Natural Resources (no. KF-2019-04-065).

References

[1] S. Nakamoto and A. Bitcoin, "A peer-to-peer electronic cash system," 2008, https://bitcoin.org/bitcoin.pdf.

[2] M. Conti, E. S. Kumar, C. Lal, and S. Ruj, "A survey on security and privacy issues of bitcoin," *IEEE Communication Surveys and Tutorials*, vol. 20, no. 4, pp. 3416–3452, 2018.

[3] F. Reid and M. Harrigan, "An analysis of anonymity in the bitcoin system," in *Security and Privacy in Social Networks*, pp. 197–223, Springer, New York, NY, 2013.

[4] S. Meiklejohn, M. Pomarole, G. Jordan et al., "A fistful of bitcoins: characterizing payments among men with no names," in *Proceedings of the 2013 conference on Internet measurement conference*, pp. 127–140, Barcelona, Spain, 2013.

[5] K. Liao, Z. Zhao, A. Doupé, and G. J. Ahn, "Behind closed doors: measurement and analysis of CryptoLocker ransoms in Bitcoin," in *2016 APWG symposium on electronic crime research (eCrime)*, pp. 1–13, Toronto, ON, Canada, 2016.

[6] B. Zheng, L. Zhu, M. Shen, X. Du, and M. Guizani, "Identifying the vulnerabilities of bitcoin anonymous mechanism based on address clustering," *Science China Information Sciences*, vol. 63, no. 3, pp. 1–15, 2020.

[7] P. Koshy, D. Koshy, and P. McDaniel, "An analysis of anonymity in bitcoin using p2p network traffic," in *International conference on financial cryptography and data security*, pp. 469–485, Springer, Berlin, Heidelberg, 2014.

[8] J. Bonneau, A. Narayanan, A. Miller, J. Clark, J. A. Kroll, and E. W. Felten, "Mixcoin: anonymity for bitcoin with accountable mixes," in *International Conference on Financial Cryptography and Data Security*, pp. 486–504, Springer, Berlin, Heidelberg, 2014.

[9] Y. Liu, X. Liu, C. Tang, J. Wang, and L. Zhang, "Unlinkable coin mixing scheme for transaction privacy enhancement of bitcoin," *IEEE Access*, vol. 6, pp. 23261–23270, 2018.

[10] J. H. Ziegeldorf, F. Grossmann, M. Henze, N. Inden, and K. Wehrle, "Coinparty: secure multi-party mixing of bitcoins," in *Proceedings of the 5th ACM Conference on Data and Application Security and Privacy*, pp. 75–86, San Antonio, Texas, USA, 2015.

[11] I. Damgård, M. Geisler, M. Krøigaard, and J. B. Nielsen, "Asynchronous multiparty computation: theory and implementation," in *International Workshop on Public Key Cryptography*, pp. 160–179, Springer, Berlin, Heidelberg, 2009.

[12] "Official announcement," http://Kucoin.comhttps://www.kucoin.com/news/kucoin-security-incident-update.

[13] US Securities And Exchange Commission, "Statement on digital asset securities issuance and trading," https://www.sec.gov/

news/public-statement/digital-asset-securites-issuuance-and-trading.

[14] L. Valenta and B. Rowan, "Blindcoin: blinded, accountable mixes for bitcoin," in *International Conference on Financial Cryptography and Data Security*, pp. 112–126, Springer, Berlin, Heidelberg, 2015.

[15] Z. Bao, W. Shi, S. Kumari, Z. Y. Kong, and C. M. Chen, "Lockmix: a secure and privacy-preserving mix service for Bitcoin anonymity," *International Journal of Information Security*, vol. 19, no. 3, pp. 311–321, 2020.

[16] I. Miers, C. Garman, M. Green, and A. D. Rubin, "Zerocoin: anonymous distributed e-cash from bitcoin," in *2013 IEEE Symposium on Security and Privacy*, pp. 397–411, Berkeley, CA, USA, 2013.

[17] E. B. Sasson, A. Chiesa, C. Garman et al., "Zerocash: decentralized anonymous payments from bitcoin," in *2014 IEEE Symposium on Security and Privacy*, pp. 459–474, Berkeley, CA, USA, 2014.

[18] G. Maxwell, "CoinJoin: bitcoin privacy for the real world," 2013, https://bitcointalk.org/index.php?topic=279249.0.

[19] T. Ruffing, P. Moreno-Sanchez, and A. Kate, "Coinshuffle: practical decentralized coin mixing for bitcoin," in *European Symposium on Research in Computer Security*, pp. 345–364, Springer, Cham, 2014.

[20] G. Danezis and A. Serjantov, "Statistical disclosure or intersection attacks on anonymity systems," in *International Workshop on Information Hiding*, pp. 293–308, Springer, Berlin, Heidelberg, 2004.

[21] S. Barber, X. Boyen, E. Shi, and E. Uzun, "Bitter to better—how to make bitcoin a better currency," in *International Conference on Financial Cryptography and Data Security*, pp. 399–414, Springer, Berlin, Heidelberg, 2012.

[22] G. Bissias, A. P. Ozisik, B. N. Levine, and M. Liberatore, "Sybil-resistant mixing for bitcoin," in *Proceedings of the 13th Workshop on Privacy in the Electronic Society*, pp. 149–158, Scottsdale, Arizona, USA, 2014.

[23] H. Sun, H. Mao, X. Bai, Z. Chen, K. Hu, and W. Yu, "Multi-blockchain model for central bank digital currency," in *2017 18th International conference on parallel and distributed computing, applications and technologies (PDCAT)*, pp. 360–367, Taipei, Taiwan, 2017.

[24] Z. Xue, M. Wang, Q. Zhang, Y. Zhang, and P. Liu, "A regulatable blockchain transaction model with privacy protection," *International Journal of Computational Intelligence Systems*, vol. 14, no. 1, p. 1642, 2021.

[25] K. V. Tu and M. W. Meredith, "Rethinking virtual currency regulation in the Bitcoin age," *Washington Law Review*, vol. 90, pp. 271–347, 2015.

[26] D. Chaum, "Blind signatures for untraceable payments," in *Advances in Cryptology*, pp. 199–203, Springer, Boston, MA, 1983.

[27] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612-613, 1979.

[28] G. R. Blakley, "Safeguarding cryptographic keys," in *1979 International Workshop on Managing Requirements Knowledge (MARK)*, pp. 313–313, New York, NY, USA, 1979.

[29] R. Dingledine, N. Mathewson, and P. Syverson, *Tor: The Second-Generation Onion Router*, Naval Research Lab, Washington DC, 2004.

[30] A. Chaabane, P. Manils, and M. A. Kaafar, "Digging into anonymous traffic: A deep analysis of the tor anonymizing network," in *2010 fourth international conference on network and system security*, pp. 167–174, Melbourne, VIC, Australia, 2010.

[31] X. Yi and K. Y. Lam, "A new blind ECDSA scheme for bitcoin transaction anonymity," in *Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security*, pp. 613–620, Auckland, New Zealand, 2019.

[32] F. G. Zhang, C. J. Wang, and Y. M. Wang, "Digital signature and blind signature based on elliptic curve," *Journal-China Institute of Communications*, vol. 22, no. 8, pp. 22–28, 2001.

[33] Freewill, "bitcoin-testnet-box," https://github.com/freewil/bitcoin-testnet-box.

[34] Y. Hong, H. Kwon, J. Lee, and J. Hur, "A practical de-mixing algorithm for bitcoin mixing services," in *Proceedings of the 2nd ACM Workshop on Blockchains, Cryptocurrencies, and Contracts - BCC '18*, pp. 15–20, Incheon, Republic of Korea, 2018.

[35] S. Meiklejohn and C. Orlandi, "Privacy-enhancing overlays in bitcoin," in *International Conference on Financial Cryptography and Data Security*, pp. 127–141, Springer, Berlin, Heidelberg, 2015.

[36] G. Maxwell, "CoinSwap: transaction graph disjoint trustless trading," 2013, https://bitcointalk.org/index.php?topic=321228.0.

[37] W. Y. Chiu, W. Meng, and C. D. Jensen, "My data, my control: a secure data sharing and access scheme over blockchain," *Journal of Information Security and Applications*, vol. 63, article 103020, 2021.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 6467866, 12 pages
https://doi.org/10.1155/2022/6467866

Research Article

An Anonymous Verifiable Random Function with Applications in Blockchain

Shuang Yao [1,2] **and Dawei Zhang** [1,2]

[1]*Department of Computer and Information Technology, Beijing Jiaotong University, Beijing 100044, China*
[2]*Beijing Key Laboratory of Security and Privacy in Intelligent Transportation, Beijing 100044, China*

Correspondence should be addressed to Dawei Zhang; dwzhang@bjtu.edu.cn

Received 15 November 2021; Revised 14 March 2022; Accepted 29 March 2022; Published 19 April 2022

Academic Editor: Liquan Chen

Verifiable random function is a powerful function that provides a noninteractively public verifiable proof for its output. Recently, verifiable random function has found essential applications in designing secure consensus protocols in blockchain. How to construct secure and practical verifiable random functions has also attracted more and more attention. In this paper, we propose a practical anonymous verifiable random function. Security proofs show that the proposed anonymous verifiable random function achieves correctness, anonymity, uniqueness, and pseudorandomness. In addition, we show a concrete application of our proposed anonymous verifiable random function in blockchain to improve the consensus mechanism for Hyperledger fabric. Finally, we implement the proposed anonymous verifiable random function and evaluate its performance. Test results show that the proposed anonymous verifiable random function supports faster computing operations and has a smaller proof size.

1. Introduction

The notion of verifiable random function was proposed by Micali et al. [1] in 1999. In a verifiable random function, the verifier can verify the function value y generated by the prover using a random message m and secret key sk via the proof π and public key pk by the prover in a noninteractive and public way as shown in Figure 1. A common verifiable random function should satisfy the following security properties. First, the verifier that receives the function value y and the corresponding proof π generated by the prover is able to verify that y is computed correctly on input m. Second, there is a unique function value y corresponding to each public key pk and the verifiable random function input m. Finally, there is no efficient adversary that can distinguish a function value y from a random element.

Since verifiable random function was proposed, it has been widely used in practice such as lottery systems [2], E-cash [3, 4], and many other situations [5]. It is also applied in Domain Name Security Extension (DNSSEC) protocol [6] to prevent offline zone enumeration attacks. Besides these traditional usages, one of the most attractive applications recently is that it is used in blockchain to improve consensus mechanisms. Consensus mechanisms play an important role in blockchain for the reason that they are responsible for achieving data consistency among untrusted nodes in blockchain. Applications utilizing verifiable random function include Algorand [7], Dfinity [8], and Ouroboros [9–11]. The main route that verifiable random function used in consensus mechanism can be roughly summarized as follows. Take Algorand for example; each user has a public key pk and a secret key sk. The user computes the function value y through verifiable random function on a random input m. If y lies in the preset range, the user will be a committee member. However, this relationship can also be publicly verified by all users in Algorand through the verification algorithm of verifiable random function. Furthermore, in public blockchain, current Proof-of-Stake (PoS) protocols inherently disclose both the identity and the wealth of the stakeholders and thus seem incompatible with privacy-preserving cryptocurrencies [12], so it is necessary to provide a construction a privacy-preserving PoS

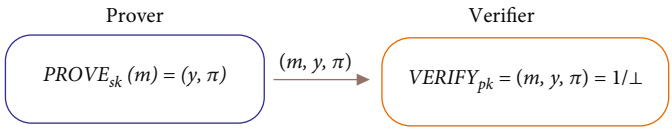


Figure 1: Verifiable random function.

protocol, that is, one where the identity of the lottery winner is kept secret by the protocol to satisfy the anonymity requirement. This also promotes the introduction of anonymous verifiable random function.

Numerous efforts have been invested in the pursuit of diversifying and simplifying constructions and underlying assumptions of verifiable random functions [13–21]. Most of them are based on RSA assumption or other complex assumptions. As verifiable random function has been gradually applied in various scenarios recently, how to construct a more efficient verifiable random function that satisfies different security properties attracts more and more attention. Ganesh et al.[12] put forward the first verifiable random function that is anonymous, which can be seen as an independent interest. There are also many other constructed verifiable random functions based on various theoretical assumptions. They all achieve the basic properties of verifiable random function except for anonymity. Therefore, in this paper, we aim to construct a more efficient anonymous verifiable random function (AVRF) that is applicable for building secure consensus mechanisms.

In addition, as one of the most popular consortium blockchains, Hyperledger fabric [22], has leveraged the benefits of both public and private blockchains. The consensus framework in Hyperledger fabric is different from public blockchain. In Hyperledger fabric, there are three types of nodes. They are endorsers, orderers, and validators. Endorsers endorse a transaction that is proposed by the transaction proposer; then, orderers block transactions and broadcast them. Validators validate transactions and update blocks on the chain. However, there is a drawback in Hyperledger fabric. Endorsing peers (endorsers) are quite essential in Hyperledger fabric as they are responsible for executing transactions proposals to ensure the transaction legality and they can directly process lots of sensitive transaction data, but these endorsing peers in consortium blockchain such as Hyperledger fabric are predetermined and endorser's identity is known to all participants. Thus, they are more likely to be attacked such as attacks possible on selective endorsers to block certain transactions, and it is necessary to construct an optimized consensus scheme with privacy properties such as randomness and anonymity for permissioned blockchain Hyperledger fabric. More precisely, endorsers should be chosen randomly, and their identities are anonymous to avoid possible attacks. Therefore, we also utilize our proposed anonymous verifiable random function to optimize the consensus mechanism in Hyperledger fabric for eliminating this drawback.

The main contributions of this paper are summarized as follows:

(i) We propose and construct an anonymous verifiable random function. The verifier can publicly verify the correctness of the function value that is the output of the prover. Meanwhile, the prover is anonymous to the verifier

(ii) We show a concrete application of the proposed verifiable random function. We use our anonymous verifiable random function to improve the consensus mechanism of Hyperledger fabric. We also analyze the security and the performance of the optimized consensus mechanism

(iii) We give theoretical analysis of the proposed anonymous verifiable random function. We also implement our anonymous verifiable random function, EC verifiable random function, Dodis verifiable random function, and the Ganesh et al. anonymous verifiable random function to evaluate their performance. Theoretical and experimental analysis results show that our anonymous verifiable random function has higher computation efficiency and a smaller proof size

1.1. Organization. We first introduce some related works in Section 2. We recall some necessary preliminaries in Section 3, and then, in Section 4, we give the concrete construction and detailed security proofs of our anonymous verifiable random function. In Section 5, we show an application of the proposed anonymous verifiable random function to improve consensus mechanism in blockchain. We then implement our anonymous verifiable random function and analyze its performance in Section 6. Conclusions are drawn in Section 7.

2. Related Works

The concept of verifiable random function was first put forth by Micali et al. [1]. Since then, verifiable random function based on elliptic curves [13] and pairing-based verifiable random function [13] have been gradually proposed. Dodis proposed a verifiable random function based on RSA[14]. In addition, some postquantum verifiable random functions [23, 24] are also proposed, but they do not have good performance in practice. Esgin et al.[25] put forward the first practical postquantum verifiable random function. It achieves significant increase in the communication size and was applied to Algorand. There are also many verifiable random functions [15, 21, 26] based on certain theoretical assumptions that have been constructed. Though these verifiable random functions all guarantee the security in pseudorandomness and uniqueness, they do not take the prover's privacy into consideration, and they cannot achieve anonymity. Ganesh et al.[12] constructed the first verifiable random function that is anonymous. In this anonymous

verifiable random function, the verifier can publicly verify the correctness of the function value as well as not reveal the public key in the verification.

In recent, verifiable random function has been widely used for consensus construction because of its randomness and public verifiability. Micali et al. proposed Algorand [7] that combines the verifiable random function and Practical Byzantine Fault Tolerance (PBFT) to select proposer and verifier committees. It avoids targeted attacks at chosen participants and achieves a high efficiency. Dfinity [8] is similar to Algorand. It actually uses Boneh–Lynn–Shacham (BLS) based verifiable random function to produce random seed which acts as the source of randomness for leader selection and leader ranking. Praos [9] is an optimized version of Ouroboros [9]. Instead of using a secure multiparty implementation of a coin-flipping protocol to produce the randomness for the leader election process, Praos uses verifiable random function for random selecting a slot leader from the stakeholder. It also prevents the adversary from learning the slot leader's identity ahead of time. Ganesh et al.[12] take the privacy properties of PoS protocol into consideration. They show that it is possible to add privacy to PoS protocols and give a privacy-preserving version of a popular PoS protocol. Most of these usages of verifiable random function mainly focus on the public blockchain. There are few researches that pay attention to providing privacy-preserving consensus scheme of the permissioned blockchain as their key nodes are predefined, and they are more likely to be attacked.

3. Preliminaries

In this section, we introduce the related hard problem and the complexity assumption, the detail of verifiable random function, and the anonymous random function. Notations used in the paper are summarized in Table 1.

3.1. Hard Problem and Complexity Assumption. Let $\mathbb{G}$ be a cyclic group of order q, where q is a prime number and it is λ bits. g is a generator of group $\mathbb{G}$. Given group elements $(\alpha = g^a, \beta = g^b, \gamma = g^{ab})$ as $a, b, c \in \mathbb{Z}_q^*$, the decisional Diffie-Hellman (DDH) problem is to distinguish between (α, β, γ) and (α, β, g^c).

We say that the DDH assumption holds if there is no probabilistic polynomial-time (PPT) algorithm$\mathscr{A}$that has advantage at leastεin solving the DDH problem in$\mathbb{G}$.

3.2. Verifiable Random Function. Let verifiable random function be a tuple of algorithms (Gen, Prove, and Verify) that are defined as follows:

(1) $\mathrm{Gen}(1^\lambda)$: the Gen algorithm takes a security parameter λ as input. It generates public key pk and secret key sk. It outputs a key pair (pk, sk)

(2) $\mathrm{Prove}(m, \mathrm{sk})$: the Prove algorithm takes $m \in \{0,1\}^{\mathrm{in}(\lambda)}$ and secret key sk as input. It generates a function value y and a proof π, then it outputs (y, π)

Table 1: Notations.

Symbol	Description
λ	Security parameter
$\mathbb{G}$	A cyclic group
g	A generator of group $\mathbb{G}$
H	Hash function
H_0	Hash function
H_1	Hash function
$\mathrm{In}(\lambda)$	input length
$\mathrm{Out}(\lambda)$	output length
pk	Public key
sk	Secret key

(3) $\mathrm{Verify}(\mathrm{pk}, m, y, \pi)$: the Verify algorithm takes a public key pk, m, a function value y, and a proof π as input. It outputs 1 or $\perp$

The verifiable random function satisfies correctness, pseudorandomness, and uniqueness as defined in the following:

(i) Correctness. For all (pk,sk) generated from the Gen algorithm and all update public key pk' generated by the KeyUpdate algorithm, and all $m \in \{0,1\}^{\mathrm{in}(\lambda)}$, if $(y, \pi) \longleftarrow \mathrm{Prove}(\mathrm{pk}', m, \mathrm{sk})$, then $\mathrm{Verify}(pk', y, \pi) = 1$

(ii) Pseudorandomness. For any pair of PPT $(\mathscr{A}_1, \mathscr{A}_2)$, the following probability is $\mathrm{negl}(\lambda)$:

$$\Pr\left[\eta = \eta' : \begin{array}{c} (\mathrm{pk}, \mathrm{sk}) \longleftarrow \mathrm{Gen}\left(1^\lambda\right); \\ (m, \mathrm{state}) \longleftarrow \mathscr{A}_1^{\mathrm{Prove}(\cdot)}(pk); \\ y_0 = F_{sk}(m); y_1 \longleftarrow \{0,1\}^{\mathrm{out}(\lambda)}; \\ \eta \longleftarrow \{0,1\}; \\ \eta' \longleftarrow \mathscr{A}_2^{\mathrm{Prove}(\cdot)}\left(y_\eta, \mathrm{state}\right) \end{array}\right] - \frac{1}{2} \tag{1}$$

Concretely, the definition means that no function value can be distinguished from random, even after seeing any other function values together with their corresponding proofs.

(i) Uniqueness. No PPT adversary $\mathscr{A}$ can output values $(p\mathrm{k}, m, y_1, y_2, \pi_1, \pi_2)$ such that $y_1 \neq y_2$ and

$$\mathrm{Verify}_{\mathrm{pk}}(m, y_1, \pi_1) = \mathrm{Verify}_{\mathrm{pk}}(m, y_2, \pi_2) = 1 \tag{2}$$

3.3. Anonymous Verifiable Random Function. We also briefly review the anonymous verifiable random function proposed in [12]. Let anonymous verifiable random function be a tuple of algorithms (Gen, Update, Prove, and Verify) as defined in the following:

(i) $\mathrm{Gen}(1^\lambda)$: it takes a security parameter λ as input. It generates public key pk and secret key sk. It outputs a key pair $(\mathrm{pk}, \mathrm{sk})$

(ii) $\mathrm{Update}(\mathrm{pk})$: it takes as input the public key pk and updates the public key pk. It outputs the updated public key pk'

(iii) $\mathrm{Prove}(m, \mathrm{pk}', \mathrm{sk})$: it takes as input $m \in \{0,1\}^{\mathrm{in}(\lambda)}$, the updated public key pk', and the secret key sk. It generates a function value Y and a proof π; then, it outputs (pk', Y, π)

(iv) $\mathrm{Verify}(\mathrm{pk}', m, Y, \pi)$: it takes as input the updated public key pk', m, a function value Y and a proof π. It outputs 1 or $\perp$

A function family $F(\cdot)$: $\{0,1\}^{\mathrm{in}(\lambda)} \longrightarrow \{0,1\}^{\mathrm{out}(\lambda)}$ is a family of anonymous verifiable random functions, if there is a tuple of algorithms (Gen, Update, Prove, and Verify) that satisfies the following properties [12]:

(i) Correctness. For all (pk, sk) generated from the Gen algorithm, all update public key pk' generated by the Update algorithm, and all $m \in \{0,1\}^{\mathrm{in}(\lambda)}$, if $(y, \pi) \longleftarrow \mathrm{Prove}(\mathrm{pk}', m, \mathrm{sk})$, then $\mathrm{Verify}(\mathrm{pk}', y, \pi) = 1$

(ii) Pseudorandomness. For any pair of PPT $(\mathscr{A}_1, \mathscr{A}_2)$, the following probability is $\mathrm{negl}(\lambda)$:

$$\Pr\left[\eta = \eta' \wedge m \notin Q_1 \cup Q_2 : \begin{array}{c} (\mathrm{pk}, m) \longleftarrow \mathrm{Gen}\left(1^\lambda\right); \\ (Q_1, m, \mathrm{state}) \longleftarrow \mathscr{A}_1^{\mathrm{Prove}(\cdot)}(\mathrm{pk}); \\ y_0 = F_{sk}(m); \\ y_1 \longleftarrow \{0,1\}^{\mathrm{out}(\lambda)}; \\ \eta \longleftarrow \{0,1\}; \\ \left(Q_2, \eta'\right) \longleftarrow \mathscr{A}_2^{\mathrm{Prove}(\cdot)}\left(y_\eta, \mathrm{state}\right) \end{array}\right] - \frac{1}{2} \tag{3}$$

The sets Q_1, Q_2 contain all the queries made to the Prove oracle. The random variable state stores information that $\mathscr{A}_1$ can save and pass on to $\mathscr{A}_2$.

(i) Uniqueness. No PPT adversary can output values $(\mathrm{pk}, m, y_0, y_0, \pi_0, \pi_1)$ such that $y_0 \neq y_1$ and

$$\mathrm{Verif}y_{\mathrm{pk}}(m, y_0, \pi_0) = \mathrm{Verify}_{\mathrm{pk}}(m, y_1, \pi_1) = 1 \tag{4}$$

(ii) Anonymity. For any PPT algorithm A, the following probability is $\mathrm{negl}(\lambda)$:

$$\Pr\left[\eta = \eta' : \begin{array}{c} (\mathrm{pk}_0, \mathrm{sk}_0) \longleftarrow \mathrm{Gen}\left(1^\lambda\right); \\ (\mathrm{pk}_1, \mathrm{sk}_1) \longleftarrow \mathrm{Gen}\left(1^\lambda\right); \\ m \longleftarrow A(\mathrm{pk}_0, \mathrm{pk}_1); \\ \mathrm{pk}'_0 \longleftarrow \mathrm{Update}(\mathrm{pk}_0); \\ (y_0, \pi_0) = \mathrm{Prove}_{\mathrm{sk}_0}\left(\mathrm{pk}'_0, m\right); \\ \mathrm{pk}'_1 \longleftarrow \mathrm{Update}(\mathrm{pk}_1); \\ (y_1, \pi_1) = \mathrm{Prove}_{\mathrm{sk}_1}\left(\mathrm{pk}'_1, m\right); \\ \eta \longleftarrow \{0,1\}; \\ \eta' \longleftarrow A\left(\mathrm{pk}'_\eta, y_\eta, \pi_\eta\right) \end{array}\right] - \frac{1}{2} \tag{5}$$

4. Construction of Our Anonymous Verifiable Random Function

In this section, we give the concrete construction of the proposed anonymous verifiable random function. The proposed anonymous verifiable random function contains a tuple of algorithms (Gen, Update, Prove, and Verify) as the following shows:

(1) $\mathrm{Gen}(1^\lambda)$: the Gen algorithm takes as input the security parameter λ. This algorithm randomly chooses $a, b \in \mathbb{Z}_q^*$; then, it computes $h_1 = g^a$ and $h_2 = g^b$. Thus, the public key is $\mathrm{pk} = (g, h_1, h_2)$ and the secret key is $\mathrm{sk} = (a, b)$. The Gen algorithm returns public key pk and secret key sk, where the secret key sk is kept secretly

(2) $\mathrm{Update}(\mathrm{pk})$: the Update algorithm takes as input the public key pk. This algorithm randomly chooses $r \in \mathbb{Z}_q^*$; then, it computes $g' = g^r$, $h'_1 = h_1^r$, and $h'_2 = h_2^r$. Therefore, the updated public key is set as $\mathrm{pk}' = (g', h'_1, h'_2)$. The Update algorithm returns the updated public key pk'

(3) $\mathrm{Prove}(\mathrm{pk}', m, \mathrm{sk})$: the Prove algorithm takes as input the updated public key pk', a random input m, and secret key sk. This algorithm generates the function value y and the corresponding proof π as the following shows:

(i) It calculates $\sigma = H_0(h'^{\,b}_1, m)$ and $s = b + \sigma/a$. So the function value can be computed as $y = H_1(s \oplus m)$

(ii) It sets the proof π as $\pi = (\sigma, s)$, and the function value is y. It returns the updated public key, the function value, and the proof (pk', y, π)

(4) $\mathrm{Verify}(\mathrm{pk}', m, y, \pi)$: the Verify algorithm takes as input the updated public key pk', a random input m, function value y, and the proof $\pi = (\sigma, s)$. It computes $w = h'^{\,s}_1 \cdot g'^{-\sigma}$; then, it determines whether equation (6) and equation (7) hold:

$$H_0(w, m) = \sigma, \tag{6}$$

$$H_1(s \oplus m) = y \tag{7}$$

If equation (6) and equation (7) all hold, the Verify algorithm outputs 1. Otherwise, the Verify algorithm outputs $\perp$.

We then prove the proposed anonymous verifiable random function satisfies correctness, anonymity, uniqueness, and pseudorandomness.

(i) Correctness. The correctness of the proposed anonymous verifiable random function represents that it can generate a function value y on any random input m with secret key through the Prove algorithm and also compute a proof π that y was computed correctly

For all public key and secret key $(\mathrm{pk}, \mathrm{sk})$ generated by the Gen algorithm, all updated public key pk' generated from the Update algorithm, all $m \in \{0, 1\}^{\mathrm{in}(\lambda)}$, proof π, and function value y generated by the Prove algorithm, we have

$$h'^{\,s}_1 \cdot g'^{-\sigma} = g^{ra(b+\sigma/a)} \cdot g^{-r\sigma} = g^{rab} \cdot g^{r\sigma} \cdot g^{-r\sigma} = h'^{\,b}_1. \tag{8}$$

So we get $H_0(h'^{\,s}_1 \cdot g'^{-\sigma}, m) = \sigma$ and $H_1(s \oplus m) = y$. Therefore, the function value y can be determined by secret key sk and m and can be verified by proof π and public key pk. The proposed anonymous verifiable random function satisfies correctness.

(ii) Anonymity. The anonymity of the proposed verifiable random function means that the verification does not reveal the public key. We adopt the idea about anonymity from the original anonymous verifiable random function that there are lots of public keys under the same secret key, and two different evaluations under the same secret key cannot be linked to a public key

We prove that the proposed anonymous verifiable random function is anonymous as the following shows.

Theorem 1. *If the DDH assumption holds in group $\mathbb{G}$, the proposed anonymous verifiable random function satisfies anonymity.*

Proof of Theorem 1. Let $\mathscr{A}$ be the adversary that wins the anonymity game. We can build an algorithm $\mathscr{B}$ to break the DDH assumption. $\mathscr{B}$ receives $(g, g_1 = g^a, g_2 = g^b, g_3 = g^c)$ and determines whether it is a DDH tuple or not. The algorithm $\mathscr{B}$ performs as the following shows:

(i) The algorithm $\mathscr{B}$ randomly selects $r \in \mathbb{Z}_q^*, \alpha \in \{0, 1\}$. It computes the public key pk_α as $\mathrm{pk}_\alpha = (g^r, g_1 = g^{ar}, g^{dr})$; then, it honestly executes the Gen algorithm to generate $\mathrm{pk}_{1-\alpha}$. The algorithm $\mathscr{B}$ returns pk_0 and pk_1 to the adversary $\mathscr{A}$

(ii) Once receiving the random input m, the algorithm $\mathscr{B}$ computes the updated public key as $\mathrm{pk}' = (g_2^r = g^{br}, g_3^r = g^{cr}, g^{dbr})$. It sets $w = g^{ad}$; then, it computes $\sigma = H_0(w, m)$, $s = d + \sigma/a$ and the function value $y = H_1(s \oplus m)$. It sets the proof π as $\pi = (\sigma, s)$. The algorithm $\mathscr{B}$ returns (pk', y, π) to the adversary $\mathscr{A}$

(iii) Let η be the output of the adversary $\mathscr{A}$. If $\eta = \alpha$, the algorithm $\mathscr{B}$ outputs "DDH tuple," otherwise the algorithm $\mathscr{B}$ outputs "not a DDH tuple."

□

Supposing the adversary $\mathscr{A}$ wins the anonymity game, then the probability $\eta = \eta'$ that we defined in the anonymity experiment is $\Pr[\eta = \eta'] \geq 1/2 + (\mathrm{Adv}_{\mathscr{A}}/2)$. So we get

$$\mathrm{Adv}_{\mathscr{B}} = |\Pr[\mathscr{B} \text{ outputs } 1 | \mathrm{DDH}] - \Pr \mathscr{B} \text{ outputs } 1 | \text{non} - \mathrm{DDH}|. \tag{9}$$

If the adversary $\mathscr{B}$ receives a non-DDH tuple, then the view of the adversary $\mathscr{B}$ is independent of η for the reason that $\mathrm{pk}' = (g^{br}, g^{abr}) = (g^{br}, (g^{br})^a)$. Thus, pk' is a correctly updated public key of pk_η. The probability of $\mathscr{B}$ outputs 1 is the same as the probability that the adversary $\mathscr{A}$ wins the anonymity game we defined. Therefore, we have

$$\Pr[\mathscr{B} \text{ outputs } 1 | \mathrm{DDH}] = \Pr\Big[\mathscr{A} \text{ outputs } \eta' = \eta \text{ in the real anonymity game}\Big]. \tag{10}$$

Then, if the algorithm $\mathscr{B}$ receives a non-DDH tuple, then the view of the adversary $\mathscr{A}$ is independent of η because $\mathrm{pk}' = (g^{br}, g^{cr})$ is independent of both pk_0 and pk_1. So the algorithm $\mathscr{B}$ cannot guess α with probability more than 1/2, so we have $\Pr[\mathscr{B} \text{ outputs} 1 | \text{non} - \mathrm{DDH}] = 1/2$. Thus, we have

$$\mathrm{Adv}_{\mathscr{B}} \geq \left|\frac{1}{2} + \frac{\mathrm{Adv}_A}{2} - \frac{1}{2}\right| \geq \mathrm{Adv}_A/2. \tag{11}$$

Therefore, the proposed anonymous verifiable random function satisfies anonymity as the DDH assumption holds.

(i) Uniqueness. The uniqueness of the proposed anonymous verifiable random function means that the function value y is uniquely determined by the corresponding secret key sk and a random input m, and accepting proofs only exist for this function value y

Suppose that for all $(\mathrm{pk}, \mathrm{sk})$ generated by the Gen algorithm, all updated public key pk' generated from the Update algorithm, and all $m \in \{0, 1\}^{\mathrm{in}(\lambda)}$, there exits the tuple (y_0, y_1, π_0, π_1) such that $\mathrm{Verify}(\mathrm{pk}', m, y_1, \pi_1) = \mathrm{Verify}(\mathrm{pk}',$

$m, y_0, \pi_0) = 1$, we can get $s_1 = b + \sigma_1/a = b + H_0((h'_1)^b, m)/a$ and $s_0 = b + \sigma_0/a = b + H_0((h'_1)^b, m)/a$, so

$$s_1 = s_2, \tag{12}$$

and according to the definition of y, we have $y_1 = H_1(s_1 \oplus m)$ and $y_2 = H_2(s_2 \oplus m)$. From equation (12), we have$y_1 = y_0$, as this is in contradiction with$y_1 \neq y_0$. Therefore, for all (pk, sk), all update pulic key pk$'$, and all $m \in \{0,1\}^{\text{in}(\lambda)}$, there does not exit any tuple (y_0, y_1, π_0, π_1) such that Verify $(\text{pk}', m, y_1, \pi_1)$= Verify$(\text{pk}', m, y_0, \pi_0) = 1$ and $y_1 \neq y_0$. The proposed anonymous verifiable random function satisfies uniqueness.

(ii) Pseudorandomness. We prove the pseudorandomness of the proposed anonymous verifiable random function as the following shows

Theorem 2. *If the DDH assumption holds in group $\mathbb{G}$, the proposed anonymous verifiable random function satisfies pseudorandomness.*

Proof of Theorem 2. Let $\mathbb{G}$ be a group and g is the generator of $\mathbb{G}$. Suppose that there is an adversary $\mathscr{A}$ that can break the pseudorandomness experiment we defined; then, we build a series of games as the following shows. Let W_0 be the probability that the adversary$\mathscr{A}$wins Game 0. Let $\text{Adv}_{\mathscr{A}}$ be the advantage of the adversary $\mathscr{A}$ in the pseudorandomness experiment. □

Game 0. This is the original pseudorandomness game we defined. The challenger $\mathscr{C}$ and the adversary $\mathscr{A}$ are interacted as the following shows:

(i) The challenger $\mathscr{C}$ computes public key and secret key $(\text{pk}, \text{sk}) \longleftarrow \text{Gen}(1^\lambda)$. It sends the generated public key pk to the adversary $\mathscr{A}$

(ii) The adversary $\mathscr{A}$ queries the oracle $\mathscr{O}_{\text{Prove}}$. The challenger $\mathscr{C}$ answers these queries

(iii) Once the challenger $\mathscr{C}$ receiving the message m^* that is sent by the adversary $\mathscr{A}$, the challenger $\mathscr{C}$ computes $(y_0, \pi_0) \longleftarrow \text{Prove}(\text{pk}, m^*, \text{sk})$ and randomly chooses $y_1 \in \{0,1\}^{\text{out}(\lambda)}$. It randomly chooses $\eta \in \{0,1\}$ and returns y_η to the adversary $\mathscr{A}$

(iv) The adversary $\mathscr{A}$ outputs η' which is the guess of η, and the adversary $\mathscr{A}$ wins the game if $\eta = \eta'$

So we get $\text{Adv}_{\mathscr{A}} = |\Pr[W_0] - 1/2|$.

Game 1. Game 1 is the same as Game 0 except that we make a change. We compute $g'^{\gamma\theta}$ for randomly chosen $\gamma, \theta \in \mathbb{Z}_q^*$ instead of computing $h'^{\,b}_1$. Let W_1 be the event that $\eta = \eta'$ in Game 1. The challenger $\mathscr{C}$ and the adversary $\mathscr{A}$ are interacted as the following shows:

(i) The challenger $\mathscr{A}$ computes public key and secret key $(\text{pk}, \text{sk}) \longleftarrow \text{Gen}(1^\lambda)$. It sends the generated pk to the adversary $\mathscr{A}$

(ii) The adversary $\mathscr{A}$ queries the oracle $\mathscr{O}_{\text{Prove}}$. The challenger $\mathscr{C}$ answers these queries

(iii) Once the challenger $\mathscr{C}$ receiving the message m^* that is sent by the adversary $\mathscr{C}$, it randomly chooses $\gamma, \theta \in \mathbb{Z}_q^*$ and computes $w = g'^{\gamma\theta}$. The challenger $\mathscr{C}$ computes $\sigma = H_0(w, m)$, $s = \gamma + \sigma/\theta$ and the function value $y_0 = H_1(s \oplus m)$. It sets $\pi = (\sigma, s)$; then, it obtains (y_0, π). The challenger $\mathscr{C}$ randomly chooses $y_1 \in \{0,1\}^{\text{out}(\lambda)}$. It randomly chooses $\eta \in \{0,1\}$ and returns y_η to the adversary $\mathscr{A}$

(iv) The adversary $\mathscr{A}$ outputs b', and it wins the game if $\eta = \eta'$.Since the adversary $\mathscr{A}$'s output of b' is independent of b, we have $\Pr[W_1] = 1/2$

Lemma 3. *We prove that $|\Pr[W_0] - \Pr[W_1]| = \varepsilon$, where ε is the advantage of some efficient algorithms to break the DDH advantage. It is negligible.*

Proof of Lemma 3. In Game 0, we have the tuple $(g', g'^a, g'^b, g'^{ab})$, while in Game 1, we have the tuple $(g', g'^a, g'^b, g'^{\gamma\theta})$. The adversary $\mathscr{A}$ cannot recognize the difference under the DDH assumption. We define a distinguishing algorithm $\mathscr{D}$. If the input to $\mathscr{D}$ is in the form of $(g', g'^a, g'^b, g'^{ab})$, the computation proceeds as in Game 0. So we have $\Pr[\mathscr{D}(g', g'^a, g'^b, g'^{ab}) = 1 : a, b \in \mathbb{Z}_q^*] = \Pr[W_0]$. If the input to $\mathscr{D}$ is in the form of $(g', g'^a, g'^b, g'^{\gamma\theta})$, the computation proceeds as in Game 1. So we have $\Pr[\mathscr{D}(g', g'^a, g'^b, g'^{\gamma\theta}) = 1 : a, b \in \mathbb{Z}_q^*] = \Pr[W_1]$. □

So the advantage to break the DDH assumption Adv_{DDH} is equal to $|\Pr[W_0] - \Pr[W_1]|$. As Adv_{DDH} is negligible, $|\Pr[W_0] - \Pr[W_1]| = \varepsilon$, ε is negligible.

According to the above proof, we have $\text{Adv}_{\mathscr{A}} = |\Pr[W_0] - (1/2)| = \varepsilon$, ε is negligible. Therefore, the proposed anonymous verifiable random function satisfies pseudorandomness.

5. Application of the Proposed AVRF in Blockchain

In this section, we show a specific application of the proposed anonymous verifiable random function in blockchain. As the key nodes in consortium blockchain such as endorsing peers in Hyperledger fabric are predetermined and fixed, they are more likely to be attacked. Therefore, we use the proposed anonymous verifiable random function to improve the consensus mechanism for Hyperledger fabric by randomly choosing endorsing peers instead of presetting

them. The improved consensus scheme is aimed at making the identity of endorsing peer random. It also provides identity privacy preservation of endorsing peers and reduces the risk attack of endorsing peers.

5.1. Hyperledger Fabric Consensus Mechanism Optimization Based on the Proposed AVRF. The consensus mechanism in Hyperledger fabric [22] is in the form of a more flexible trust model called "endorse-order-validate" which is different from consensus mechanism in public blockchain [27–29]. As we can see in Figure 2, in Hyperledger fabric, there are three types of nodes. They are endorsers, orderers, and validators. Firstly, in the endorsement phase, endorsers are predetermined and fixed. Endorsers execute transactions and record these results. Secondly, in the ordering step, it uses a pluggable consensus protocol to produce a totally ordered sequence of endorsed transactions grouped in blocks. These endorsed transactions are broadcasted to all peers via the gossip protocol. Next, in the validation step, validators validate the state changes from endorsed transactions with respect to the endorsement policy in the validation step.

In Hyperledger fabric, endorsing peers (endorsers) are quite essential as they are responsible for executing transactions proposals to ensure the transaction legality, and they can directly process lots of sensitive transaction data. However, as endorsers' identities are public and fixed, they are more likely to be attacked. Besides, the number of endorsers is small compared to other peers' numbers in general. It is even in single digits in some systems. This makes that there are many-to-one relationships between clients and endorsers, so it is difficult for endorsers to process transactions timely, which increases the transaction processing time.

In accordance with the above problems, we construct a noninteractive, verifiable, and optimized consensus scheme for randomly selecting endorsers based on the proposed anonymous verifiable random function. We use the candidate set of endorsing peers and randomly select endorsing peers in the candidate set through our anonymous verifiable random function. The usage of anonymous verifiable random function achieves the identity privacy of endorsers before endorsement, and this randomly expands the number of endorsing peers.

As we can see in Figure 3, the optimized consensus scheme based our anonymous verifiable random function is defined as follows:

(1) The client generates proposal proposal = <req, m, clientsig>. req is the transaction data which includes chaincode and its parameters. m is the random input that satisfies $m \in \{0,1\}^{\mathrm{in}(\lambda)}$. The client signs these data and generates clientsig. It sends the proposal to the candidate set of endorsing peers; then, the client starts a timer

(2) The candidate endorsing peer verifies the signature clientsig to check the integrity. If the verification fails, it aborts. Otherwise, the candidate endorsing peer performs as follows:

(i) The candidate endorsing peer executes the anonymous verifiable random function Update algorithm to generate the update public key pk'. It executes $\mathrm{Prove}(\mathrm{pk}', m, \mathrm{sk})$ to get the function value y and proof π

(ii) The candidate endorsing peer compares whether $(H(y)/2^{h\mathrm{len}}) > \eta$ holds. η is the predetermined threshold. H is a hash function and $h$len is the length of $H(y)$. If it holds, it means that the candidate endorsing peer is an endorser. It goes to the next step. Otherwise, it aborts

(iii) If a candidate endorsing peer has confirmed that it is an endorser, it executes the proposal to generate read and write set rw_set and the endorsing result ed; then, it computes the signature of $(\mathrm{rw\_set}, \mathrm{ed}, (\mathrm{pk}', y, \pi))$ as epsig, while si $\mathrm{g}_{\mathrm{sk}} = ra$ and $\mathrm{sig}_{\mathrm{pk}} = h'_1 = g^{ra}$. Therefore, the proposal response message as res_pro = <rw_set, ed, (pk', y, π), epsig>. It sends the proposal response message *res_pro t*o the client

(3) The client continuously receives proposal response messages res_pro from different endorsers before the timer runs out. It performs as the following shows:

(i) The client verifies the signature epsig for checking the integrity. If the verification fails, it aborts. Otherwise, it executes $\mathrm{Verify}(\mathrm{pk}', y, \pi)$ to verify the function value y. On one hand, if there is an adversary that replaces (pk', y, π) without secret key, which may lead to some malicious endorsing peers without endorsers qualifications to become logical endorsers. This will influence transaction endorsing results. However, when the client receives the replaced response message, it first verifies the signature epsig, then it verifies the function value y. For the reason that the signature satisfies unforgeability and the anonymous verifiable random function satisfies uniqueness, the replaced response message will not pass these verifications, and the malicious endorsing peer without endorser qualification will not become the logical endorser to influence transaction endorsing results. On the other hand, our anonymous verifiable random function can also be extended to provide some level of unpredictability under malicious key generation. In order to achieve this goal, in the Prove algorithm, it adds a computation $v = H_2(y, m)$, where H_2 is a hash function. Also, let $\pi = (\sigma, s, y)$. It outputs (π, v). In the Verify algorithm, it adds an verification to check whether $v = H_2(y, m)$ holds. In this case, our extended anonymous verifiable random function can provide unpredictability under malicious key generation. It means that an adversary that can maliciously choose the verifiable random keys cannot skew the output distribution, as long as the adversary has no information on the

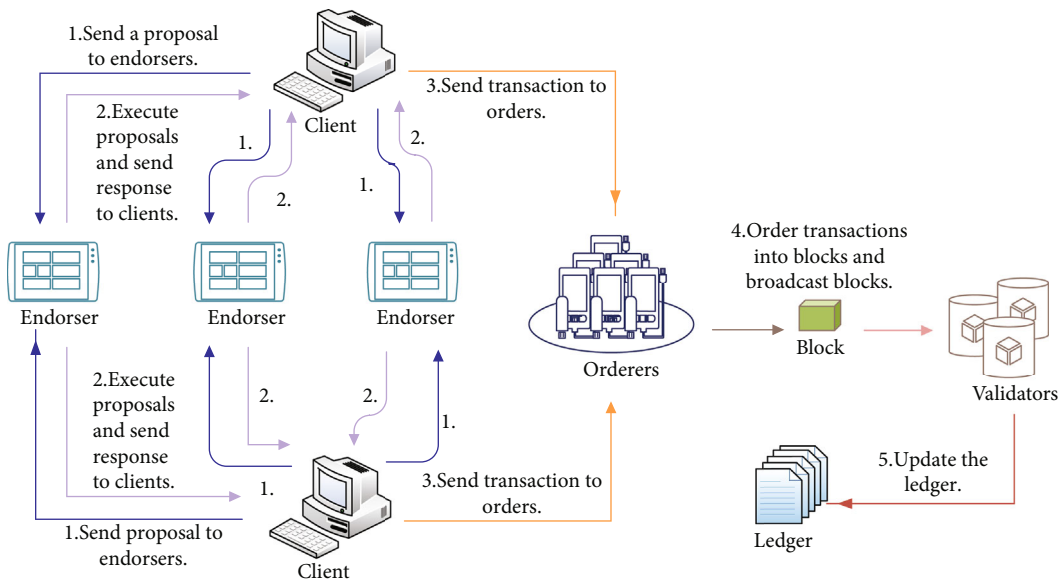


Figure 2: The consensus mechanism of Hyperledger fabric.

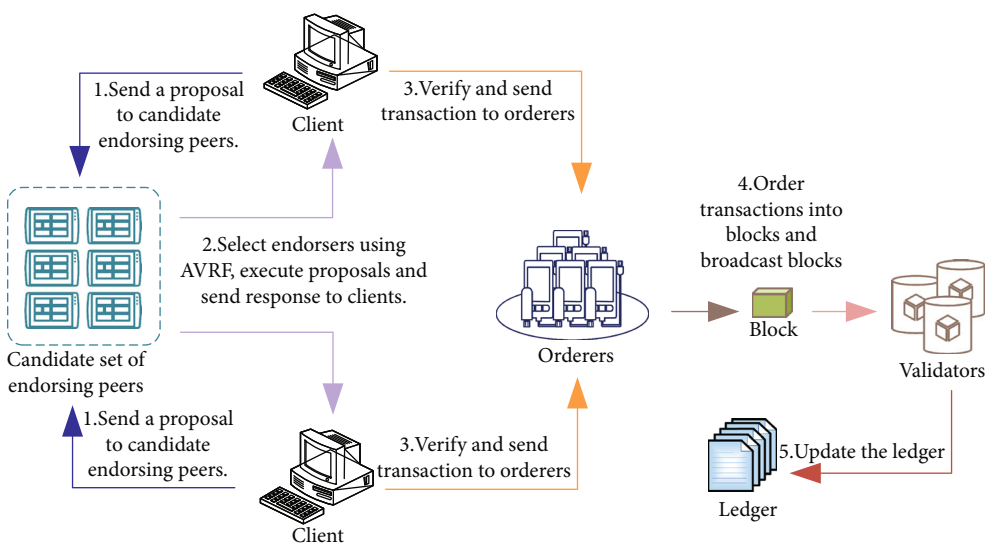


Figure 3: The optimized consensus scheme.

random input m when choosing its verifiable random function keys. We adopt the idea about unpredictability under malicious key generation from [9] and [25] that have given us detailed explanation and proof

(ii) The client computes $H(y)/2^{h\text{len}}$ and checks whether $(H(y)/2^{h\text{len}}) > \eta$ holds. If it holds, the proposal response message is from a logical endorser. The client sends these transactions to orderers

(4) Orderer monitors and receives all transactions and block transactions as block $< \text{tx}>^{n}_{\text{sign}}$. Let tx^n denote that there are n orderly transactions in a block. Orders sign blocks and broadcast them

(5) Validator receives the block, then it verifies the signature of the block the read and write set and updates the ledger. After all these steps, it represents that a consensus of transactions initiated by clients is gained

5.2. Analysis of the Optimized Consensus Scheme

5.2.1. Security Analysis

(1) Randomness. In our optimized consensus scheme, endorsers are randomly selected from the candidate set of

Table 2: Efficiency comparison.

	Dodis VRF	Ganesh et al. AVRF	The proposed AVRF
Prove time complexity	$P + E_T + E$	$5E + 2H$	$4E + 2H$
Verify time complexity	$E_T + 3P + M$	$4E + 2M + 2H$	$2E + M + H$
Proof size	$\lvert\mathbb{G}\rvert + \lvert\mathbb{G}_T\rvert$	$\lvert\mathbb{G}\rvert + 2\lvert\mathbb{Z}_q\rvert$	$2\lvert\mathbb{Z}_q\rvert$
Prove computation overhead (ms)	2.0	1.9	1.6
Verify computation overhead (ms)	3.3	1.2	0.9

endorsing peers via the proposed anonymous verifiable random function instead of predetermined. Whether a candidate endorsing peer is an endorser or not is determined by the function value y which is the random output of the anonymous verifiable random function's Prove algorithm. Only the function value y satisfies that $(H(y)/2^{h\text{len}}) > \eta$; a candidate endorsing peer is chosen as an endorser. As we can see from the definition of anonymous verifiable random function, y satisfies randomness, so the selection of endorsers is random. This reduces the centralization of endorsing peers.

At the same time, in the original consensus scheme, clients continue to process the transaction only after they have compared all endorsing results that were sent from endorsers and results are all same. So, the adversary can easily control endorsing results to destroy the correctness of transaction results if it has successfully attacked only one endorser to make endorsing results inconsistent and the client aborts the transaction. Furthermore, if the decision strategy is modified, the endorsing results are valid only if there are more than half of the results that are consistent. In this case, adversary can control the endorsing result to destroy the correctness of transaction results if more than half of endorsing peers are malicious. On the contrary, in our optimized consensus scheme, clients do not have to compare endorsing results from all predetermined endorsing peers. Endorsers are chosen randomly and dynamically; this will reduce the probability of adversary's influence on the transaction.

(2) Anonymity. In our optimized consensus scheme, endorser's identity is verifiable and anonymous to the client. As the proposed anonymous verifiable random function satisfies correctness, the endorsing peer's identity can be verified. Furthermore, observers cannot obtain the result about which candidate endorsing peers have been chosen if secret keys are not leaked. Moreover, the client can use the endorsing peer's update public keys to verify the identity validity of the endorsing peer, so the client cannot recognize the identity of endorsing peers for the reason that the proposed anonymous verifiable random function satisfies anonymity. Concretely, verification using update public keys will not reveal endorsing peers' public keys. The anonymity of our optimized consensus scheme provides privacy preservation of endorsing peers and reduces their risk of being attacked.

5.2.2. Performance Analysis of the Optimized Consensus Scheme. On one hand, for the same m, different secret key sk will generate different function value y by the Prove algorithm of the proposed anonymous verifiable random function. Therefore, different candidate endorsing peer will generate different function values y in the same transaction. Some of candidate endorsing peers will become endorsers for this transaction, while the rest of endorsing peers in candidate set will become other transactions' endorsers. The randomness of function value y ensures that the transaction is uniformly distributed to candidate endorsing peers. This reduces the workload of each endorsing peer and improves concurrent processing of transactions.

On the other hand, the transaction delay is the time it takes to initiate a proposal, endorse, validate, order, and commit transactions to the ledger. In our optimized consensus scheme, as the transaction flow is the same as the original consensus scheme except for the endorsing step, the main factor that increases the transaction processing time is that there is an extra endorser selection process and endorser identity's verification process. According to the prove time and the verify time of the proposed anonymous verifiable random function in Table 2, they are both milliseconds. It is negligible compared with the whole transaction processing time. Thus, the impact of the proposed anonymous verifiable random function on the transaction delay is negligible and there is no much difference on transaction delay between our optimized consensus scheme and the original consensus scheme.

6. Implementation and Evaluation

In this section, in order to give a better evaluation of the performance about our proposed anonymous verifiable random function, we give a reference implementation of our anonymous verifiable random function as well as the anonymous verifiable random function proposed in [12] in Python language. For convenience, we call it Ganesh et al. anonymous verifiable random function. We also implement another two representative verifiable random functions the Dodis verifiable random function [13] which is used in Algorand and the EC verifiable random function [6] that has been widely used in many scenarios such as in DNSSEC. We use the Charm [30] library to implement the elliptic curve group operations. We measure the prove time and the verify time of these verifiable random functions. Our tests are performed on a Linux desktop with an 8-core Intel Core i7-8550U 2.00 GHz processor and 8 GB of RAM. We also average the performance over 50 runs.

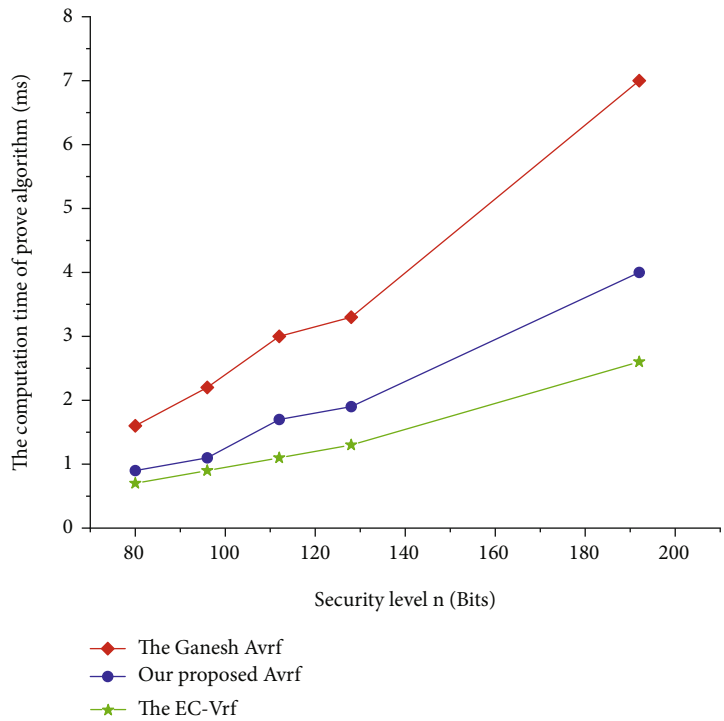


Figure 4: Prove time comparison.

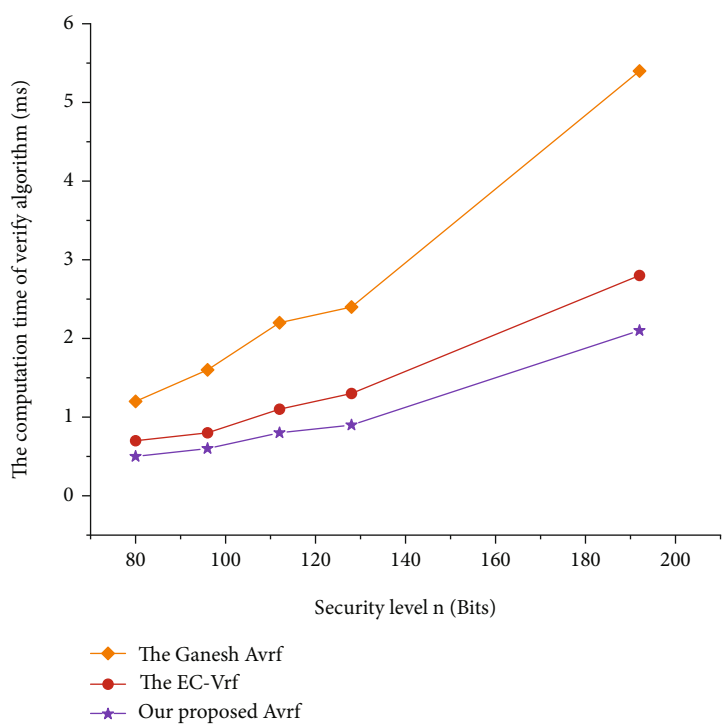


Figure 5: Verify time comparison.

In Table 2, we give the efficiency analysis by comparing our proposed anonymous verifiable random function, the Ganesh et al. anonymous verifiable random function, and the Dodis verifiable random function in terms of time complexity, computation overhead, and the size of proof π. We denote E as exponentiation operation in group $\mathbb{G}$, H as hash function, E as multiplication operation in group $\mathbb{G}_T$, P as pairing operation, M as multiplication operation in group $\mathbb{G}$, $|\mathbb{G}|$ as the size of elements in group $\mathbb{G}$, and $|\mathbb{Z}_q|$ as the size of elements in $\mathbb{Z}_q$. As we can see from Table 2, verify times

of the Dodis verifiable random function, Ganesh et al. anonymous verifiable random function, and our proposed anonymous verifiable random function are, respectively, 3.3 ms, 1.2 ms, and 0.9 ms with the 80-bit security level. It is obvious that our anonymous verifiable random function has the best performance in terms of the verify time and the proof size.

In Figure 4, we compare the computation of prove time among our anonymous verifiable random function, the Ganesh et al. anonymous verifiable random function, and the EC verifiable random function. As we set the security level as 80 bits, 96 bits, 112 bits, 128 bits, and 192 bits, respectively, the prove time of Ganesh et al. anonymous verifiable random function grows from 1.6 ms to 7.0 ms while our proposed anonymous verifiable random function increases from 0.9 ms to 4.0 ms. It is obvious that our anonymous verifiable random function has lower prove computation overhead compared with the Ganesh et al. anonymous verifiable random function. However, the prove computation overhead of our proposed anonymous verifiable random function is a little higher than the EC verifiable random function for the reason that there are extra exponentiation operations in our proposed verifiable random function to achieve anonymity, while the EC verifiable random function is not anonymous.

In Figure 5, we compare the computation of verify time among our anonymous verifiable random function, the Ganesh et al. anonymous verifiable random function and the EC verifiable random function. When security levels are set to be 80 bits, 96 bits, 112 bits, 128 bits, and 192 bits, respectively, the verify time of the Ganesh et al. anonymous verifiable random function grows from 1.2 ms to 5.4 ms and the EC verifiable random function grows from 0.7 ms to 2.9 ms. In our verifiable random function, it increases from 0.5 ms to 2.1 ms. Our anonymous verifiable random function also has the lowest verify computation overhead among these three verifiable random functions.

Therefore, the proposed anonymous verifiable random function is efficient according to the above analytical measurements and experimental evaluation as it has shorter prove and verify time as well as a smaller proof size.

7. Conclusions

In this paper, we construct an efficient anonymous verifiable random function which has a potential utilization in blockchain to build secure consensus protocols. Specially, our proposed verifiable random function is anonymous. It means that the verification will not reveal the public key of the prover. We also analyze and prove its security properties. Furthermore, we give a concrete utilization of our proposed anonymous verifiable random function to optimize the consensus mechanism of Hyperledger fabric. In addition, we implement and evaluate the proposed anonymous verifiable random function and another three representative verifiable random functions. Experimental results show that the proposed anonymous verifiable random function has lower computation overhead and a smaller proof size compared with other representative verifiable random functions. The proposed anonymous verifiable random function can also be applied to other permissioned blockchains as their transactions are processed by certain key nodes. However, to achieve a practical postquantum anonymous verifiable random function is still for future work.

Data Availability

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

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This paper was supported by National Natural Science Foundation of China (Grant no. U21A20463).

References

[1] S. Micali, M. Rabin, and S. Vadhan, "Verifiable random functions," in *40th Annual Symposium on Foundations of Computer Science (cat. No. 99CB37039)*, pp. 120–130, New York, NY, USA, 1999.

[2] S. Micali and R. L. Rivest, *Micropayments revisited. Cryptographers' Track at the RSA Conference*, Springer, 2002.

[3] M. H. Au, W. Susilo, and Y. Mu, "Practical compact e-cash," in *Australasian Conference on Information Security and Privacy*, pp. 431–445, Berlin, Heidelberg, 2007.

[4] M. Belenkiy, M. Chase, M. Kohlweiss, and A. Lysyanskaya, "Compact e-cash and simulatable VRFs revisited," in *International Conference on Pairing-Based Cryptography*, pp. 114–131, Berlin, Heidelberg, 2009.

[5] B. Mishra, D. Jena, and S. Patnaik, "Privacy preserving file auditing schemes for cloud storage using verifiable random function," *International Journal of Communication Networks and Distributed Systems*, vol. 26, pp. 50–75, 2021.

[6] D. Papadopoulos, D. Wessels, S. Huque, M. Naor, J. Vˇcelák, and S. Goldberg, *Making NSEC5 practical for DNSSEC*, Cryptology ePrint Archive, 2017.

[7] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich, "Algorand: scaling byzantine agreements for cryptocurrencies," in *Proceedings of the 26th symposium on operating systems principles*, pp. 51–68, Shanghai, China, 2017.

[8] T. Hanke, M. Movahedi, and D. Williams, "DFINITY technology overview series, consensus system," https://arxiv.org/abs/1805.04548.

[9] B. David, P. Gaži, A. Kiayias, and A. Russell, "Ouroboros Praos: an adaptively-secure, semisynchronous proof-of-stake blockchain," *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 2018, pp. 66–98, Cham, 2018.

[10] C. Badertscher, P. Gaži, A. Kiayias, A. Russell, and V. Zikas, "Ouroboros Genesis: composable proof-of-stake blockchains with dynamic availability," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 913–930, 2018.

[11] A. Kiayias, A. Russell, B. David, and R. Oliynykov, "Ouroboros: a provably secure proof-of-stake blockchain protocol,"

in *Annual International Cryptology Conference*, pp. 357–388, Cham, 2017.

[12] C. Ganesh, C. Orlandi, and D. Tschudi, "Proof-of-stake protocols for privacy-aware blockchains," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 690–719, Cham, 2019.

[13] Y. Dodis and A. Yampolskiy, *A verifiable random function with short proofs and keys. International Workshop on Public Key Cryptography*, Springer, 2005.

[14] Y. Dodis, *Efficient construction of (distributed) verifiable random functions. International Workshop on Public Key Cryptography*, Springer, 2003.

[15] N. Chandran, S. Raghuraman, and D. Vinayagamurthy, *Constrained pseudorandom functions: verifiable and delegatable*, Cryptology ePrint Archive, 2014.

[16] G. Guo, Y. Zhu, E. Chen, G. Zhu, D. Ma, and W. 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, pp. 304–323, 2021.

[17] G. Fuchsbauer, "Constrained verifiable random functions," in *International Conference on Security and Cryptography for Networks*, pp. 95–114, Cham, 2014.

[18] S. Hohenberger and B. Waters, "Constructing verifiable random functions with large input spaces," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 656–672, Berlin, Heidelberg, 2010.

[19] T. Jager, "Verifiable random functions from weaker assumptions," in *Theory of Cryptography Conference*, pp. 121–143, Berlin, Heidelberg, 2015.

[20] D. Hofheinz and T. Jager, "Verifiable random functions from standard assumptions," in *Theory of Cryptography Conference*, pp. 336–362, Berlin, Heidelberg, 2016.

[21] N. Bitansky, "Verifiable random functions from non-interactive witness-indistinguishable proofs," *Journal of Cryptology*, vol. 33, pp. 459–493, 2020.

[22] E. Androulaki, A. Barger, V. Bortnikov et al., "Hyperledger fabric: a distributed operating system for permissioned blockchains," in *Proceedings of the Thirteenth EuroSys Conference*, pp. 1–15, Porto, Portugal, 2018.

[23] S. Yamada, "Asymptotically compact adaptively secure lattice IBEs and verifiable random functions via generalized partitioning techniques," in *Annual International Cryptology Conference*, pp. 161–193, Cham, 2017.

[24] R. Goyal, S. Hohenberger, V. Koppula, and B. Waters, "A generic approach to constructing and proving verifiable random functions," in *Theory of Cryptography Conference*, pp. 537–566, Cham, 2017.

[25] M. F. Esgin, V. Kuchta, A. Sakzad et al., "Practical postquantum few-time verifiable random function with applications to Algorand," in *International Conference on Financial Cryptography and Data Security*, pp. 560–578, Berlin, Heidelberg, 2021.

[26] L. Kohl, "Hunting and gathering–verifiable random functions from standard assumptions with short proofs," in *IACR International Workshop on Public Key Cryptography*, pp. 408–437, Cham, 2019.

[27] S. Nakamoto, "Bitcoin: a peer-to-peer electronic cash system," *Decentralized Business Review*, p. 21260, 2008.

[28] G. Wood, "Ethereum: a secure decentralised generalised transaction ledger," *Ethereum project yellow paper*, vol. 151, pp. 1–32, 2014.

[29] E. B. Sasson, A. Chiesa, C. Garman et al., "Zerocash: decentralized anonymous payments from bitcoin," in *2014 IEEE Symposium on Security and Privacy*, pp. 459–474, Berkeley, CA, USA, 2014.

[30] J. A. Akinyele, C. Garman, I. Miers et al., "Charm: a framework for rapidly prototyping cryptosystems," *Journal of Cryptographic Engineering*, vol. 3, pp. 111–128, 2013.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 3923597, 16 pages
https://doi.org/10.1155/2022/3923597

Research Article

A Fine-Grained Medical Data Sharing Scheme with Ciphertext Reencryption

Jiahao Chen,[1] **Jingwei Wang**,[1] **Xinchun Yin**,[1,2] **and Jianting Ning**[3,4]

[1]*College of Information Engineering, Yangzhou University, Yangzhou, China*
[2]*College of Guangling, Yangzhou University, Yangzhou, China*
[3]*College of Computer and Cyberspace Security, Fujian Normal University, Fuzhou, China*
[4]*Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China*

Correspondence should be addressed to Xinchun Yin; xcyin@yzu.edu.cn

Received 9 January 2022; Revised 20 February 2022; Accepted 7 March 2022; Published 27 March 2022

Academic Editor: Liquan Chen

With the fantastic development of cloud computing technology, cloud storage has been widely applied in the field of medical data sharing due to its efficiency and flexibility. Attribute-based encryption (ABE), as a promising technology, plays an important role in the fine-grained sharing of medical data. However, there are two issues: (1) an access policy specified in the encryption phase may need to be updated after a period of time, and (2) the considerable decryption overhead in ABE is too heavy for resource-limited devices. To address these issues, a fine-grained medical data sharing scheme with ciphertext reencryption is proposed. Users with decryption permission can produce reencryption keys and cloud server can reencrypt initial ciphertext. The cloud server can predecrypt ciphertext, which reduces user's decryption overhead. Moreover, the traditional centralized server is replaced with a decentralized blockchain that performs system initialization, key management, and user revocation. We compared the proposed scheme with some existing schemes in terms of function and efficiency; the results show that the efficiency of the proposed scheme outperforms other related schemes. The security analysis shows that the proposed scheme is security and correctness.

1. Introduction

Since cloud computing was proposed, it has received increasing attention [1]. On the one hand, it is an Internet-based value-added service [2], which means that users can obtain cloud services through the network anywhere and anytime. On the other hand, with the continuous development of big data, cloud computing, as a novel and efficient computing paradigm [3], can store and manage huge amounts of data for users. As a result, an increasing number of individuals and enterprises outsource their data to the cloud, promoting the growth of cloud computing.

Cloud storage, a new storage mechanism evolved from cloud computing, can provide users with ubiquitous data storage and access services by centrally processing resources stored in its area. At the same time, by taking use of cloud storage's services, customers not only save costs on local data storage and maintenance but also get access to and share data with greater ease. Therefore, cloud storage is widely used in people's daily life due to its flexibility, convenience, and economy [4].

In the medical field, cloud storage has the advantages of low cost, convenient deployment and efficient storage, and the operations on its platform are basically automated, making it the most ideal environment for deploying standard data security access and sharing mechanisms [5]. On the one hand, the generation, collection, storage, integration, and application of data are decentralized, and cloud storage makes it possible to obtain high-quality medical services through the collection, management, and utilization of these data. On the other hand, with the explosive growth of data and the development of Internet of Things technology, users

can effectively save the cost of local data management and maintenance by outsourcing their massive medical data to the cloud for storage.

In the secure access and sharing mechanism of medical data stored in the cloud, as long as users are authorized, they can obtain safe, convenient, and high-quality medical services in a timely manner by sharing corresponding medical resources in the cloud. This not only greatly improves the storage and sharing efficiency of medical information but also effectively solves the problems of the traditional medical model, such as few medical channels for users, high medical costs for users, and unreasonable medical services for users. Therefore, the secure access and sharing mechanism of cloud storage medical data has become a hot topic in the field of medical and health research [6].

The basic model of secure accessing and sharing mechanism of medical data is shown in Figure 1. In the secure accessing and sharing mechanism of cloud storage medical data, data owners first encrypt the medical data, by paying a fee, the obtained ciphertext can be hosted to the cloud storage center, and the cloud storage center uses access control technology to manage users' permissions to the sharing data.

However, patients' medical data is sensitive and confidential; they may include social security numbers, credit card information, and treatment history. CSP cannot be fully trusted due to various attacks in an untrusted network environment. Once medical data leaked, it may cause serious medical accidents and economic losses, which restricts the application of medical data sharing [7–9]. Therefore, we need to address security and privacy issues when designing practical and secure EHR sharing systems adapted to the cloud environment.

Access control (AC) [10], as a part of the secure accessing and sharing mechanism of medical data, plays a significant role in medical data protection. As in an untrusted cloud storage environment, data owners encrypt the medical data before storing it in the cloud and ensuring that data requesters with decryption capabilities may access the data, therefore realizing the security of medical data cloud storage. Traditional ciphertext AC techniques, however, are no longer suited for cloud-based data sharing due to issues such as low work efficiency, high computing costs, and large bandwidth. This problem has been significantly improved by the proposal of the ciphertext-policy attribute-based encryption (CP-ABE) technique. In CP-ABE, data owners define the AC to ensure that only the data requesters who meet the access policy can decrypt the corresponding ciphertext. This provides not just fine-grained AC but also "one-to-many" encryption. Therefore, with the help of CP-ABE, safe access control and sharing of cloud-storage medical data can be implemented.

In secure accessing and sharing mechanism of medical data based on cloud storage, there are also a large number of scenarios where the ciphertext needs to be reencrypted. For example, ciphertext reencryption technology enables doctor Alice working in P city's hospital to share a patient's medical data with doctor Bob working in Q city's hospital. The ciphertext reencryption technology also enables the doctor Alice, who is on leave, to share her patient's medical data to another doctor Eve in the same hospital. The straightforward solution to reencryption the ciphertext is that Alice first downloads the ciphertext from the cloud, decrypts the ciphertext, and reencrypts the plaintext with the new access policy. After encryption, Alice uploads the new ciphertext to the cloud. When a large amount of patients' ciphertexts need to be reencrypted, this method not only incurs large computing overhead but also increases the communication cost between the cloud and Alice. In order to share data more effectively, proxy reencryption (PRE) [11] is introduced into CP-ABE, which is called ciphertext-policy attribute-based proxy reencryption scheme (CP-ABPRE). In response to the above situation, Alice generates a reencryption key and sends the key to the proxy server. The proxy server can use the key to encrypt the original ciphertexts, so that Bob(Eve) can decrypt the encrypted ciphertext data and realize the sharing of medical data.

In PRE, the ciphertext encrypted under A's public key can be transformed into ciphertext encrypted under B's public key by a semitrusted proxy, and the proxy will not get any information about the underlying encrypted medical data. By implementing the permission of decryption authority, CP-ABPRE can securely and effectively reencrypt ciphertexts and assure the secure access and sharing of medical data. Since the proxy only needs to verify that the attributes in the user-generated reencryption key satisfy the access policy of the original ciphertexts, the proxy server can reencrypt a batch of such ciphertexts. As a result of its great security and low computing overhead, this technology is suitable for a large number of ciphertext reencryption scenarios.

Overall, CP-ABPRE can not only perform one-to-many encryption but it can also safely reencrypt the relevant ciphertexts. It is suitable for the sharing of medical data. For one thing, the proxy can reencrypt the ciphertexts corresponding to access policy A into access policy B without changing the plaintexts. This not only allows for fine-grained access control for medical data users, but it also assures the security of medical data stored in the cloud. In other words, the proxy does not have to decrypt the ciphertext and then reencrypt the plaintext during the whole process. Instead, it may simply complete the ciphertext conversion by directly performing operations on the ciphertexts, which considerably decreases the proxy's computing cost. Therefore, CP-ABPRE is critical for the secure accessing and sharing mechanism of medical data in cloud storage.

However, most of the existing CP-ABPRE schemes must have a trusted center or key generation center (KGC), and the keys are generated and maintained through the KGC. In addition, centralized KGCs are vulnerable to attacks such as a single point of failure, once KGC is compromised by malicious organizations, and it will lead to serious information leakage. Blockchain technology has the potential to tackle this issue effectively. This technology was invented by Nakamoto [12] in 2008, and it has gained global notice. The benefits of blockchain include distributed storage, non-tampering, and traceability. As a result, it has a wide range of applications in a variety of industries, including finance, communications, and the Internet of Things. In this paper,

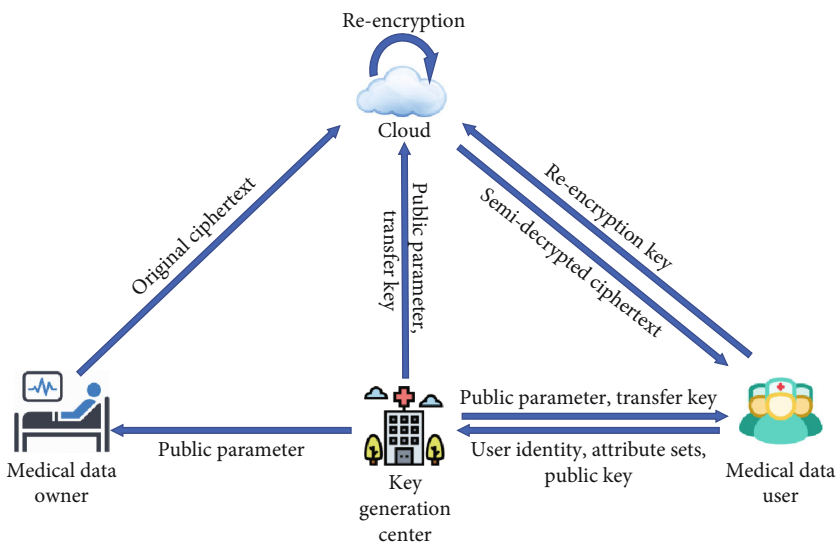


Figure 1: Basic model of secure accessing and sharing mechanism of medical data based on cloud storage.

blockchain is used to replace the traditional trusted center, and users' keys are generated by consensus nodes and users themselves, which solves the problem of centralization of key generation in traditional CP-ABPRE.

2. Related Work

In the traditional public key cryptosystem, the data owner can encrypt the medical data with the public key so that only the data user with the corresponding private key can decrypt ciphertexts, which effectively solves the problem of key distribution. However, due to issues with certificate administration and key escrow, this approach is inapplicable when large-scale medical data owners need to exchange their data. At the same time, in most cases, data owners may not know the particular receivers but aim to accomplish fine-grained access control through strategies created by data owners. As a result, the standard public key cryptosystem is incapable of addressing the issue of data access and sharing in cloud storage.

With the continuous development of cryptographic research, Sahai and Waters [13] prompted an ABE scheme in 2005. Subsequently, Goyal et al. [14] proposed a key-policy attribute-based encryption (KP-ABE) scheme, in which the access policy is bound to the user secret key, and the attribute set is related to the ciphertext. When the attribute set in the ciphertext satisfies the policy in the key, the ciphertext can be decrypted. Bethencourt et al. [15] proposed a ciphertext-policy attribute-based encryption (CP-ABE) scheme. In CP-ABE, the access policy is bound to the ciphertext while the attribute set is related to the user secret key. Only when the attribute set in the key satisfies the access policy in the ciphertext can the ciphertext be decrypted. Since the access policy can be defined by the data owner, CP-ABE is suitable for medical data sharing in cloud storage. In 2011, Waters [16] proposed a CP-ABE scheme that uses linear secret sharing scheme (LSSS). Compared with the scheme of Bethencourt et al. [15], the efficiency has been improved.

In order to address the problem of access policy updating, in 2009, Liang et al. [17] proposed a ciphertext-policy attribute-based proxy reencryption (CP-ABPRE) scheme, which allows a proxy to reencrypt a ciphertext encrypted by policy A to policy B and the proxy cannot obtain any information about the plaintext. However, their scheme cannot resist choose plaintext attacks. To solve this problem, Liang et al. [18] proposed a proxy reencryption scheme based on CP-ABE in 2013, which can resist choose plaintext attacks. However, both the generation of the reencryption key and the reencryption ciphertext require a large amount of calculation. In 2016, Yang et al. [19] employed a cloud server to implement dynamic attribute-based access control, which greatly improved the efficiency of reencryption and reduced the user's decryption overhead. However, the proposed scheme only supports tree access structure. In 2017, Feng et al. [20] prompted an attribute-based proxy reencryption (AB-PRE) scheme which supports LSSS, but it only supports AND gates. In the same year, Ma et al. [21] proposed a verifiable outsourced decryption scheme; the outsourced decryption operation was completed by a decryption server. In 2019, Xu et al. [22] proposed a novel healthcare IoT system fusing advantages of attribute-based encryption, cloud, and edge computing, which provides an efficient, flexible, secure fine-grained access control mechanism with data verification in healthcare IoT network without any secure channel and enables data users to enjoy the lightweight decryption. In 2020, Deng et al. [23] proposed an AB-PRE scheme that supports arbitrary monotonic access policies, but the reencrypted ciphertext only supports identity-based encryption (IBE), and the scalability of reencryption is not as good as ABE. In 2020, Paul et al. [24] proposed an efficient attribute-based proxy reencryption with constant size ciphertext, but his scheme did not support user revocation. In 2021, Xu et al. [25] proposed a practical and secure dynamic EHR sharing system via Cloud. The user

decryption keys of the works above are all generated by the KGC. Once the KGC is compromised, it will cause serious information leakage.

Blockchain was first proposed as the underlying technology of Bitcoin [12]. It is an open distributed public ledger. Due to the hash function and consensus protocol used in the blockchain, it can not only ensure the unforgeability of the data stored in it but also guarantees traceability. Any user can access all the data stored in the ledger, and it is impossible for a malicious user to tamper with this data. In 2017, Huh et al. [26] used public blockchain to manage IoT devices; all the public key of users are stored in blockchain. In 2017, Liu et al. [27] used private blockchain to ensure data integrity. In 2019, Hu et al. [28] proposed a blockchain-based data sharing scheme, a decentralized system is established based on smart contracts.

2.1. Contributions. In this paper, we proposed a blockchain-aided fine-grained medical data sharing scheme supporting ciphertext reencryption. The characteristics of the scheme are as follows.

2.1.1. Blockchain-Aided ABE. The proposed scheme allows users to generate private keys by themselves. This prevents the usage of users' private keys created by KGC in traditional schemes. The KGC in traditional ABE is replaced by the consensus nodes of the distributed consensus blockchain, and the immutable ledger can store transactions. The user's public key and system public parameters can be uploaded to the blockchain to ensure that the data is not tampered with.

2.1.2. Ciphertext Reencryption. The proposed scheme allows the cloud to reencrypt a ciphertext decryptable by user A into a ciphertext decryptable by user B without disclosing the plaintext or the authorizer's private key. The complete procedure does not necessitate decryption.

2.1.3. Collusion Resistance. The proposed scheme adopts the user's Global Identifier (GID) to bind attributes of users. Only attributes belonging to the same user can decrypt the ciphertext, thereby preventing the system from colluding attacks.

2.1.4. Lightweight Decryption and Verification. The proposed scheme allows the cloud to pre-decrypt the ciphertext, lowering the user's computational cost in the decryption process. Data user can decrypt the final ciphertext with only one exponentiation computation, and the user can verify if the cloud has made the correct transformation.

2.1.5. Efficient User Revocation. For situations such as the resignation of doctors and the recovery of patients, their access rights need to be revoked. The proposed scheme supports efficient user revocation.

2.2. Paper Organization. Section 3 introduces the background knowledge. The system architecture model is formally defined in Section 4. Details of the proposed scheme are depicted in Section 5. Section 6 introduces the performance analysis. Ultimately, the conclusion is given in Section 7.

3. Preliminaries

3.1. Notations. Related notations and definitions used in the proposed scheme are presented in Table 1.

3.2. Access Structures. Let $\{P_1, P_2, \cdots, P_n\}$ be the set of n participants. For set $A \subseteq 2^{\{P_1, P_2, \cdots P_n\}}$, if $\forall B, C \subseteq \{P_1, P_2, \cdots P_n\}$, $B \in C$, $B \subseteq C \Rightarrow C \in A$, then A is said to be monotonic. Among them, the set belonging to A is called authorized set; otherwise, it is called unauthorized set [29].

3.3. Bilinear Maps. Let G and G_T be two multiplicative cyclic groups of order p. g is a generator of G and $e : G \times G \longrightarrow G_T$ is a map with three properties: (1) bilinearity—for $\forall u, v \in G$ and $\forall a, b \in Z_p$, $e(u^a, v^b) = e(u, v)^{ab}$; (2) non-degeneracy—$e(g, g) \neq 1$; and (3) computability—for all $\forall u, v \in G$, there exists effective algorithms to calculate $e(u, v)$.

3.4. Linear Secret Sharing Schemes. Let A denote the attribute university and p denote a prime number. The secret sharing scheme Π on the secret space Z_p realizes the access policy on A. If Π satisfies the following two properties, then, Π is linear [30].

(1) Each part of the secret $s \in Z_p$ corresponds to an attribute in A, and each part constitutes a vector on Z_p

(2) For the access policy $S = (M, \rho)$, M is a $l \times n$ secret sharing matrix. The function ρ maps the $i_{th} \in \{1, 2, \cdots, l\}$ row of M to an attribute $\rho(i)$ of the attribute set A. For example, construct a column vector $u = (s, y_2, y_3, \cdots, y_n)^{\mathrm{T}}$, where $y_2, y_3, \cdots, y_n \in Z_p$ are random numbers used to hide the secret value s. Then, $M_u \in Z_p^{l \times 1}$ is a vector with l rows and 1 column. That is, the secret value s is divided into l parts according to Π. $(Mu)_i$ corresponds to the attribute $\rho(i)$, where $i \in [l]$

Every LSSS has linear reconstruction property. Suppose that Π is a LSSS for the access structure S. $P \in S$ is the set of arbitrary authorization, $I = \{i \in [l] \wedge \rho(i) \in P\}$ and $I \subseteq \{1, 2, \cdots, l\}$. For any legal sharing of $\{\lambda_i = (Mu)_i\}_{i \in I}$, there exists a set of constants $\{\sigma_i \in Z_p\}_{i \in I}$ that can be calculated in polynomial time. For the unauthorized set p', there is no such constant set as $\{\sigma_i\}_{i \in I}$.

3.5. Blockchain Technology. Blockchain is a specific data structure based on transparent and trusted consensus rules in a peer-to-peer network. It combines data blocks in a chronological order to form a specific data structure. Blockchain cryptographically guarantees a decentralized and credible distributed shared ledger system whose data cannot be tampered with, forged or traced. Blockchain is mainly divided into three types: public blockchain, consortium blockchain, and private blockchain. The consortium blockchain usually designates one or more preselected consensus nodes as the bookkeeper. The generation of each block is jointly determined by all consensus nodes. The connected nodes can participate in activities on the chain but do not care about the process of accounting. The public blockchain adopts a proof

Table 1: Notations and definitions used in the proposed scheme.

Notation	Definition
p	Large prime
G, G_T	Cyclic group of order p
g	Generator of G
Z_p	Integer domain not greater than p
e	Bilinear map
(M, ρ)	Access policy(M is a matrix of $l \times n$, ρ is a map)
CSP	Cloud server provider
MDO	Medical data owner
MDU	Medical data user
λ	Security parameter
A	Attribute universe
CN	Consensus nodes
H	Hash function
SE	Symmetric-key encryption algorithm
GID	User's Global Identifier

of work mechanism to exchange efficiency for fairness. The private blockchain sets a single central accounting node to exchange fairness for efficiency. Compared with the public blockchain and private blockchain, the participants in the consortium blockchain know each other's digital identities and can reach business consensus in advance. The proof-of-work mechanism is no longer needed; thus, this can reduce energy consumption and improve efficiency. Since the consortium blockchain takes into account both efficiency and fairness, it is considered by many researchers to be the most suitable blockchain for medical data sharing application scenarios.

3.6. Random Oracle Model. The random oracle model (ROM) was proposed in 1993 [31]; the ROM uses the character of a random oracle to show the security of the algorithm. The random oracle can be considered a black box in theory, and the black box is embedded with a certain algorithm so that the inquirer cannot obtain any information no matter how many inquiries are made. Among them, the returned value is collision resistant, the value is the same for the same query, and the value is unpredictable for different queries. The random oracle $H : \{0,1\}^* \longrightarrow \{0,1\}^n$ is a specific type of hash function, and the function satisfies the following characteristics:

(1) Uniformity: given the result y after H operation, the distribution of y on $\{0,1\}^n$ is uniform.

(2) Effectiveness: given a string x, H can do certain operations to get $H(x)$. $H(x)$ is the effective output of H in low-order polynomial time of x length.

(3) Certainty: if the two numbers input by the oracle H are the same, then the two corresponding outputs must be the same.

3.7. q-DBPBDHE2 Assumption. The system selects two multiplicative cyclic groups G and G_T whose order is prime p, g is the generator of group G, and $e : G \times G \longrightarrow G_T$ is a bilinear mapping. The system randomly selects $a, s, b_1, b_q \in Z_p^*$ and $R \in G_T$. The system defines a vector $D = (p, g, G, e, g^s, \{g^{a^i}\}_{i\in[2q], i\neq q+1}, \{g^{b_j a^i}\}_{(i,j)\in[2q,q], i\neq q+1}, \{g^{s/b_i}\}_{i\in[q]}, \{g^{s b_j a^i / b_{j'}}\}_{(i,j,j')\in[q+1,q,q], j\neq j'})$. If the adversary cannot distinguish a random factor $R \in G_T$ from $e(g,g)^{a^{q+1}s}$ with a nonnegligible advantage ε in probabilistic polynomial time (PPT), then q-DBPBDHE2 assumption establish. Advantage ε is defined as $|\Pr[A(D, e(g,g)^{a^{q+1}s}) = 0] - \Pr[A(D, R) = 0]| \geq \varepsilon$.

4. System Architecture

4.1. Functionalities of Each Entity. The proposed scheme includes 4 entities. As shown in Figure 2, they are consensus node (CN), cloud server provider (CSP), medical data owner (MDO), and medical data user (MDU).

4.1.1. Consensus Node. Consensus nodes are the maintainer of the consortium blockchain. Each consensus node manages the attributes of MDUs, initializing the system, generating transfer keys for MDU, and maintaining a revocation list.

4.1.2. Cloud Server Provider. CSP is responsible for storing ciphertext and managing the attribute transfer keys for each user. CSP receives the reencryption key and generates the corresponding reencryption ciphertext. CSP also performs outsourcing decryption for MDU. Furthermore, CSP needs to revoke illegal users in the system.

4.1.3. Medical Data Owner. MDO first encrypts medical data with a symmetric key encryption algorithm, then encrypts the symmetric-key with a CP-ABPRE, and finally uploads the ciphertext to the CSP.

4.1.4. Medical Data User. MDU with sufficient attributes can request the CSP to perform the semidecryption algorithm and then decrypt the semidecrypted ciphertext with their own private key efficiently. MDU also generates a reencryption key and uploads it to the CSP. It is worth noting that the MDU can only obtain the semidecrypted ciphertext after outsourcing decryption of the CSP but cannot obtain the original ciphertext in the CSP.

4.2. Formal Definition

(1) GlobalSetup$(\lambda, A) \longrightarrow (\mathrm{PP}, R)$: the algorithm inputs the security parameters λ according to the security requirements of the scheme and attribute universe A and outputs the system public parameter PP, an empty revocation list R.

(2) NodeSetup$(\mathrm{PP}, \eta) \longrightarrow (\mathrm{NPK}_\eta, \mathrm{NSK}_\eta)$: this algorithm takes the public parameters PP and CN's identity η as input and outputs CN's node public key NPK_η and node secret key NSK_η.

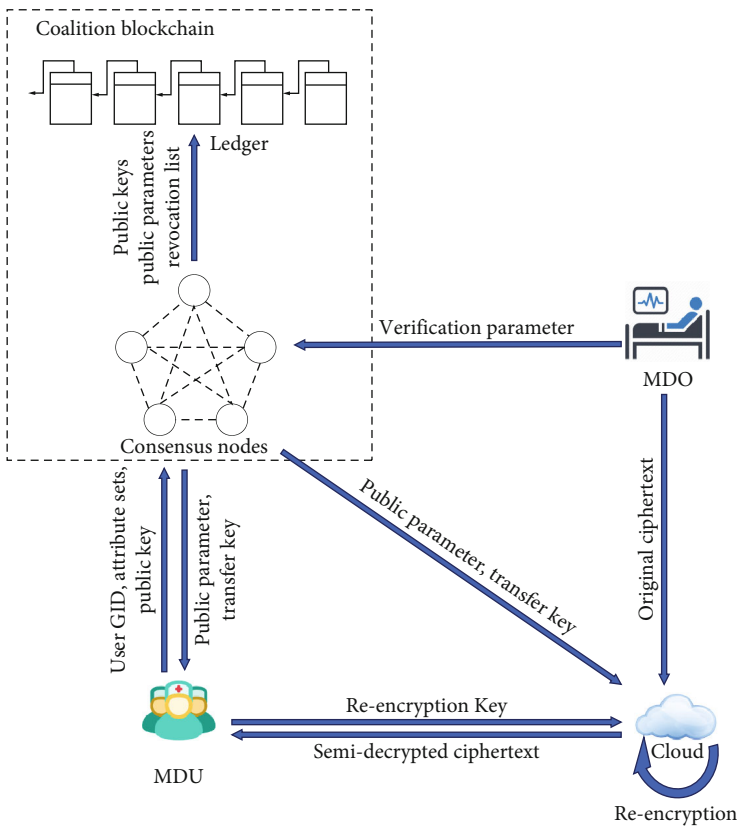


Figure 2: System model.

(3) $\mathrm{UserKeyGen}(\mathrm{PP}, \mathrm{GID}) \longrightarrow (\mathrm{UPK}, \mathrm{USK})$: the algorithm inputs the PP and user GID and outputs the user public key UPK and user private key USK.

(4) $\mathrm{TKGen}(\mathrm{PP}, \mathrm{GID}, S, \mathrm{NSK}_{\eta}, \mathrm{UPK}) \longrightarrow (\mathrm{TK}_{S,\mathrm{GID}})$: the algorithm inputs the public parameter PP, user GID, user attribute set S, node secret key NSK_{η} and user public key UPK and outputs the user's transfer key $\mathrm{TK}_{S,\mathrm{GID}}$. CN sends it to the user and CSP; CSP adds $(\mathrm{GID}, \mathrm{TK}_{S,\mathrm{GID}})$ to the key list KT.

(5) $\mathrm{Enc}(\mathrm{PP}, m, (M, \rho), \mathrm{NPK}_{\eta}) \longrightarrow (\mathrm{CT})$: this algorithm inputs the public parameter PP, a EHR plaintext m, an access policy (M, ρ), and node public key NPK_{η} and outputs the ciphertext CT.

(6) $\mathrm{ReKeyGen}(\mathrm{PP}, \mathrm{TK}_{S,}\ \mathrm{GID}, \mathrm{USK}, \mathrm{NPK}_{\eta}, S, (M', \rho')) \longrightarrow \mathrm{ReKey}_{(S,\mathrm{GID})\longrightarrow}\ (M', \rho')$: this algorithm inputs the public parameter PP, transfer key $\mathrm{TK}_{S,\mathrm{GID}}$, user private key USK, node public key NPK_{η}, user attribute set S, and a new access policy (M', ρ') and outputs the reencryption key $\mathrm{ReKey}_{(S,\mathrm{GID})\longrightarrow(M',\rho')}$.

(7) $\mathrm{ReEnc}(\mathrm{PP}, \mathrm{ReKey}_{(S,\mathrm{GID})\longrightarrow(} M', \rho'), \mathrm{CT}) \longrightarrow C^{R}_{(M',\rho')}$: this algorithm inputs the public parameter PP, reencryption key $\mathrm{ReKey}_{(S,\mathrm{GID})\longrightarrow(M',\rho')}$, and ciphertext CT and outputs the reencryption ciphertext $C^{R}_{(M',\rho')}$.

(8) $\mathrm{TransDec}(\mathrm{PP}, \mathrm{CT}, S, \mathrm{TK}_{S,\mathrm{GID}}, \mathrm{UPK}) \longrightarrow \mathrm{CT}_{\mathrm{GID}}$: this algorithm inputs the public parameter PP, ciphertext CT, user attribute set S, transform key $\mathrm{TK}_{S,\mathrm{GID}}$, and user public key UPK and outputs the semidecrypted ciphertext $\mathrm{CT}_{\mathrm{GID}}$.

(9) $\mathrm{UserDec}(\mathrm{USK}, \mathrm{CT}_{\mathrm{GID}}) \longrightarrow m$: this algorithm inputs the user private key USK and outputs the plaintext m.

(10) $\mathrm{TransDec}_{R}(\mathrm{PP}, C^{R}_{(M',\rho')}, S', \mathrm{TK}_{S',\mathrm{GID}}) \longrightarrow \mathrm{CT}_{\mathrm{GID}'}$: this algorithm inputs the public parameter PP, ciphertext $C^{R}_{(M',\rho')}$, user attribute set S', and transform key $\mathrm{TK}_{S',\mathrm{GID}}$ and outputs the semidecrypted ciphertext $\mathrm{CT}_{\mathrm{GID}'}$.

(11) $\mathrm{UserDec}_{R}(\mathrm{PP}, \mathrm{USK}, \mathrm{CT}_{\mathrm{GID}'}) \longrightarrow m$: this algorithm inputs the public parameter PP, user private key USK, and semidecrypted ciphertext $\mathrm{CT}_{\mathrm{GID}'}$ and outputs the plaintext m

(12) $\mathrm{Revoke}(\mathrm{GID}, R, \mathrm{KT}) \longrightarrow (R, \mathrm{KT}/(\mathrm{GID}, \mathrm{TK}_{S,\mathrm{GID}}))$: this algorithm inputs the identity of the revoked user GID, revocation list R, and key list KT and

outputs an updated revocation list R' and an updated key list KT/(GID, $\mathrm{TK}_{S,\mathrm{GID}}$).

4.3. Threat Model. In the system, consensus nodes and MDO are fully trusted. Each consensus node manages the attributes of MDUs, initializing the system, generating transfer keys for MDU, and maintaining a revocation list. MDO honestly encrypts medical data and generates ciphertext according to its own wishes and uploads it to CSP.

The CSP is semitrusted. CSP is semitrusted, and it follows our scheme but tries to learn sensitive data without any active attack. In addition to this, CSP may also dishonestly decrypt ciphertext for MDU to ease the burden of data storage or for any other reason.

MDU are untrusted. MDU try to learn about unauthorized data and even try to collude with other MDU to gain access to sensitive data. To prevent attacks from untrusted DUs, we apply cryptosystems to ensure data confidentiality.

From the above, the security of our proposed scheme depends on the blockchain and cryptosystem. As we all know, all data stored on the blockchain cannot be tampered with. The cryptosystems in our scheme are hash functions, symmetric encryption, and our scheme. We apply a collision-resistant hash function to prevent adversaries from compromising data integrity. Symmetric encryption is known to be resistant to all kinds of attacks. For the security of our scheme, we propose a formal security model in the next subsection to demonstrate the security of our scheme.

4.4. Security Model. The security model of the proposed scheme is a static security model. The static security model satisfies the confidentiality of messages; it can be defined by a security game between an adversary Adv and a challenger C. The game is performed in the form of query-answer between Adv and C. All inquiries must be completed before the challenge phase. Since static security model of the scheme is mainly aimed at the collusion attack of multiple legitimate MDUs, it is required that Adv can ask the MDU for the private key many times and ask other legitimate MDUs: GID to partially decrypt the ciphertext $\mathrm{CT}_{\mathrm{GID}}$. The implementation of the latter is based on that Adv can ask CSP for the corresponding transform key, so that Adv can obtain the GID's $\mathrm{CT}_{\mathrm{GID}}$ according to the transform key of CSP.

Definition 1. Given a probabilistic polynomial time, if there is no one adversary Adv can win the following game with a nonnegligible advantage, the proposed scheme is static security. In the game, A represents attribute universe of the protocol, and θ represents the consensus node set. Assume that each attribute is assigned to a consensus node (although each consensus node can manage multiple properties).

The game consists of the following 4 phases.

(1) Initialization phase: the initialization phase is performed by challenge C. C inputs λ according to security requirements. Then, the GlobalSetup algorithm is performed, and the public parameter PP is sent to Adv.

(2) Query phase: define consensus nodes as N_θ. Adv does the following query to C.

(i) Query 1: Adv chooses some N_θ; Adv asks C for their corresponding public key NPK_θ.

(ii) Query 2: Adv selects some MDUs' $\{\mathrm{GID}_i\}_{i=1}^m$ and asks for their key pairs $\{\mathrm{UPK}_i, \mathrm{USK}_i\}_{i=1}^m$.

(iii) Query 3: Adv asks C for the transfer key: input MDUs' $\mathrm{GID}_i (1 \le i \le n)$ and their attribute sets $S_i \in A$; C performs TKGen(PP, GID, S, NSK_θ, UPK) $\longrightarrow (\mathrm{TK}_{S,\mathrm{GID}})$ to generate transfer key $\mathrm{TK}_{S,\mathrm{GID}}$. Then, C returns $\mathrm{TK}_{S,\mathrm{GID}}$ to Adv. At the same time, the query made by Adv includes not only the query to obtain the transfer key corresponding to the MDUs' GID_i but also the query to obtain the transfer key corresponding to other MDUs (that is, if n is larger than m).

(iv) Query 4: Adv asks C for reencryption key: input user $\mathrm{GID}_i (1 \le i \le n)$, user attribute sets $S_i \in A$, and access policy (M', ρ'); C performs ReKeyGen(PP, $\mathrm{TK}_{S,\mathrm{GID}}$, USK, NPK_η, S, (M', ρ')) $\longrightarrow \mathrm{ReKey}_{(S,\mathrm{GID})\longrightarrow(M',\rho')}$ to generate reencryption key $\mathrm{ReKey}_{(S,\mathrm{GID})\longrightarrow(M',\rho')}$. Then, C returns $\mathrm{ReKey}_{(S,\mathrm{GID})\longrightarrow(M',\rho')}$ to Adv. $\mathrm{TK}_{S,\mathrm{GID}}$ is generated by TKGen() algorithm.

(3) Challenge phase: Adv submits two equal-length plaintexts $M_0{}^*$, $M_1{}^*$, and an access policy (M^*, ρ^*) to C. C randomly selects $b \in \{0, 1\}$ and executes Enc() algorithm to calculate challenge ciphertext and return it to *Adv*. The limitation is that the $\{\mathrm{GID}_i\} (1 \le i \le m)$ of the user who asked for the private key during the query; their attribute set S_i does not satisfy (M^*, ρ^*).

(4) Guess phase: Adv outputs guess $b' \in \{0, 1\}$ on b. If $b = b'$, Adv wins the game. The advantage of Adv winning is defined as $|\Pr[b = b'] - 1/2|$.

5. Proposed Scheme

5.1. Concrete Construction. This section will give the design of the protocol. To better understand the protocol, the access policy of the protocol is represented by a two-tuple (M, ρ), where M is a matrix of $l \times n$ and ρ maps each row of M to an attribute $\rho_{(x)}$. At the same time, define a function $F : A \longrightarrow A_\eta$; A represents attribute universe of the protocol, and A_η represents attributes managed by a consensus node (CN) with identity η: CN_η; this function can implement the mapping of the attribute $u \in A$ to CN_η. The

corresponding function $\delta(\bullet) = F(\rho(\bullet))$can realize the mapping of the rows of the matrix to a consensus node. The concrete construction is as follows:

(1) $\text{GlobalSetup}(\lambda, A) \longrightarrow (\text{PP}, R)$: the GlobalSetup algorithm is performed by all CNs. They take a security parameter λ and system attribute universe A as input and outputs system public parameter PP, an empty revocation list R. The algorithm randomly selects a security parameter λ. Let G and G_T be two multiplicative cyclic groups with prime order p, g be a generator of G, $\mathscr{K}$ be the key space of symmetric encryption, and $e : G \times G \longrightarrow G_T$ be a nondegenerate bilinear pair. The algorithm selects 4 safety hash functions: $H_1 : \{0,1\}^* \longrightarrow G$, $H_2 : G_T \longrightarrow Z_p^*$, $H_3 : A \longrightarrow G$, and $H_4 : \{0,1\}^* \longrightarrow \mathscr{K}$, and a symmetric-key encryption algorithm: $\text{SE} = (\text{SE.Enc}(\bullet), \text{SE.Dec}(\bullet))$. In SE, $\text{SE.Enc}(\bullet)$ is a symmetric encryption algorithm. At last, CNs output global public parameters $\text{PP} = \{p, G, G_T, g, e, H_1, H_2, H_3, H_4, \text{SE}, A, A_\eta\}$, an empty revocation list R, and sends them to the blockchain.

(2) $\text{NodeSetup}(\text{PP}, \eta) \longrightarrow (\text{NPK}_\eta, \text{NSK}_\eta)$: the NodeSetup algorithm is performed by all CNs. Each CN_η takes public parameters PP and identity η as input, output node public key NPK_η, and node secret key NSK_η. Each CN_η randomly selects α_η, $y_\eta \in Z_p^*$ and computes $e(g,g)^{\alpha_\eta}, g^{\alpha_\eta}, g^{y_\eta}$. They keep $\text{NSK}_\eta = \{\alpha_\eta, y_\eta\}$ confidential and publish $\text{NPK}_\eta = \{e(g,g)^{\alpha_\eta}, g^{\alpha_\eta}, g^{y_\eta}\}$.

(3) $\text{UserKeyGen}(\text{PP}, \text{GID}) \longrightarrow (\text{UPK}, \text{USK})$: MDU performs the UserKeyGen algorithm. The algorithm inputs PP and user GID and outputs MDU's user public key UPK and user private key USK. The algorithm randomly selects $x_{\text{GID}} \in Z_p^*$ and computes $K_1 = g^{x_{\text{GID}}}$, $K_2 = H_1(\text{GID})^{x_{\text{GID}}}$. It sets $\text{UPK} = (K_1, K_2)$ and $\text{USK} = x_{\text{GID}}$.

(4) $\text{TKGen}(\text{PP}, \text{GID}, S, \text{NSK}_\eta, \text{UPK}) \longrightarrow (\text{TK}_{S,\text{GID}})$: CN performs the TKGen algorithm. The algorithm inputs public parameter PP, user GID, user attribute set S, node secret key NSK_η, and user public key UPK and outputs transfer key $\text{TK}_{S,\text{GID}}$. For all attributes, $u \in A$, if u is managed by CN_η. Then CN_η randomly selects $t_u \in Z_p^*$ and computes $\text{CSK}_{u,\text{GID}} = g^{x_{\text{GID}}\alpha_\eta} \bullet H_1(\text{GID})^{x_{\text{GID}} y_\eta} \bullet H_3(u)^{t_u}$ and $\text{CSK}'_{u,\text{GID}} = g^{t_u}$. Then, CN_η sets $\text{TK}_{S,\text{GID}} = (\text{CSK}_{u,\text{GID}}, \text{CSK}'_{u,\text{GID}})_{u \in S}$ and sends it to MDU and CSP. CSP adds $(\text{GID}, \text{TK}_{S,\text{GID}})$ to the key list KT.

(5) $\text{Enc}(\text{PP}, m, (M, \rho), \text{NPK}_\eta) \longrightarrow (\text{CT})$: MDO performs the Enc algorithm. Let M be a $l \times n$ matrix and ρ be the function that maps each row of M to an attribute. The algorithm randomly selects $Y \in G_T$ and sets $\text{key} = H_4(Y)$, selects a positive integer ϖ, and concatenates ϖ bit 0 string after the message m, which is used for outsourced decryption verification. The algorithm uses the SE.Enc algorithm to obtain the ciphertext of the message m: $M_{\text{SE}} = \text{SE.Enc}_{\text{key}}(m \| 0^\varpi)$, where $\|$ denotes concatenation of a string. Then, it computes $\text{Ver} = H_1(m \| 0^\varpi)$.The algorithm randomly selects $s, y_2, \cdots, y_n, z_2, \cdots, z_n \in Z_p^*$ and sets vector $v = (s, y_2, \cdots y_n)^{\mathrm{T}}$, $\omega = (0, z_2, \cdots, z_n)^{\mathrm{T}}$. For $x \in [l]$, the algorithm randomly selects $Q_x, R_x, r_x \in Z_p^*$ and computes $C_0 = Y \bullet e(g,g)^s$, $C_{1,x} = g^{Q_x} g^{\alpha_{\delta(x)} r_x}, C_{2,x} = g^{-r_x}$, $C_{3,x} = g^{y_{\delta(x)} r_x} g^{R_x}$, $C_{4,x} = H_3(\rho(x))^{r_x}$, $\lambda_x = (Mv)_x$, $\omega_x = (M\omega)_x$, $C_{5,x} = \lambda_x - Q_x$, and $C_{6,x} = \omega_x - R_x$. The algorithm uploads $\text{CT} = ((M, \rho), C_0, \{C_{1,x}, C_{2,x}, C_{3,x}, C_{4,x}, C_{5,x}, C_{6,x}\}_{x \in [l]}, M_{\text{SE}})$ to CSP and uploads Ver to blockchain.

(6) $\text{ReKeyGen}(\text{PP}, \text{TK}_{S,\text{GID}}, \text{USK}, \text{NPK}_\eta, S, (M', \rho')) \longrightarrow \text{ReKey}_{(S,\text{GID}) \longrightarrow (M', \rho')}$: MDU performs the ReKeyGen algorithm. It inputs public parameter PP, transfer key $\text{TK}_{S,\text{GID}}$, user private key USK, node public key NPK_η, user attribute set S, and a new access policy (M', ρ') and outputs reencryption key $\text{ReKey}_{(S,\text{GID}) \longrightarrow (M', \rho')}$. The algorithm randomly selects $s', y'_2, \cdots, y'_n, z'_2, \cdots, z'_n, r'_x \in Z_p^*, \beta \in G_T$ and sets two vectors $v' = (s', y'_2, \cdots, y'_n)^{\mathrm{T}}$ and $\omega' = (0, z'_2, \cdots, z'_n)^{\mathrm{T}}$. Then, for $x \in [l']$, the algorithm sets $\lambda'_x = (M'v')_x$ and $\omega'_x = (M'\omega')_x$. The algorithm computes $C'_0 = \beta \bullet e(g,g)^{s'}$, $C'_{1,x} = g^{\lambda_x'} g^{\alpha_{\delta'(x)} r'_x}$, $C'_{2,x} = g^{-r_x'}$, $C'_{3,x} = g^{y_{\delta'(x)} r'_x} g^{\omega_x'}$, and $C'_{4,x} = H_3(\rho'(x))^{r'_x}$ and then sets $rk_1 = ((\text{CSK}_{u,\text{GID}})^{(x_{\text{GID}})^{-1}})^{H_2(\beta)}$, $rk_2 = ((\text{CSK}'_{u,\text{GID}})^{(x_{\text{GID}})^{-1}})^{H_2(\beta)}$, $rk_3 = g^{H_2(\beta)}$, $rk_4 = (H_1(\text{GID})^{x_{\text{GID}}})^{(x_{\text{GID}})^{-1} H_2(\beta)}$, and $rk_5 = ((M', \rho'), C'_0, \{C'_{1,x}, C'_{2,x}, C'_{3,x}, C'_{4,x}\}_{x \in [l']})$. Finally, the algorithm generates $\text{ReKey}_{(S,\text{GID}) \longrightarrow (M', \rho')} = (S, rk_1, rk_2, rk_3, rk_4, rk_5)$ and sends $\text{ReKey}_{(S,\text{GID}) \longrightarrow (M', \rho')}$ to CSP.

(7) $\text{ReEnc}(\text{PP}, \text{ReKey}_{(S,\text{GID}) \longrightarrow (M', \rho')}, \text{CT}) \longrightarrow C^R_{(M', \rho')}$: CSP performs the ReEnc algorithm. It inputs public parameter PP, reencryption key $\text{ReKey}_{(S,\text{GID}) \longrightarrow (M', \rho')}$, and ciphertext CT and outputs reencryption ciphertext $C^R_{(M', \rho')}$. After receiving $\text{ReKey}_{(S,\text{GID}) \longrightarrow (M', \rho')}$ sent by the MDU, the algorithm will check whether the attribute set S in $\text{ReKey}_{(S,\text{GID}) \longrightarrow (M', \rho')}$ matches the access policy (M, ρ) in CT. If S does not match (M, ρ), it indicates $\text{ReKey}_{(S,\text{GID}) \longrightarrow (M', \rho')}$ is illegal, and CSP terminates the reencryption operation. If S matches (M, ρ), the algorithm performs the following steps to compute the reencrypted ciphertext $C^R_{(M', \rho')}$.

The algorithm sets $I = \{x : \rho(x) \in S\} \subseteq (1, 2, \cdots, l)$ and computes constant $\{c_x \in Z_p^*\}$ to let $\sum_{x\in I} c_x M_x = (1, 0, .., 0)$. Then, the algorithm computes $C_4 = \prod_{x\in I}(e(C_{1,x} g^{C_{5,x}}, rk_3))^{c_x}$ and $C_5 = \prod_{x\in I}(e(rk_1, C_{2,x}) e(rk_4, C_{3,x} g^{C_{6,x}}) e(rk_2, C_{4,x}))^{c_x}$ and then computes $C_6 = C_4 \bullet C_5 = e(g, g)^{sH_2(\beta)}$. Finally, the algorithm outputs $C^R_{(M',\rho')} = ((M', \rho'), C_0, C_6, rk_5, M_{\mathrm{SE}})$.

(8) $\mathrm{TransDec}(\mathrm{PP}, \mathrm{CT}, S, \mathrm{TK}_{S,\mathrm{GID}}, \mathrm{UPK}) \longrightarrow \mathrm{CT}_{\mathrm{GID}}$: CSP performs the TransDec algorithm. It inputs public parameter PP, ciphertext CT, user attribute set S, transform key $\mathrm{TK}_{S,\mathrm{GID}}$, and user public key UPK and outputs semidecrypted ciphertext $\mathrm{CT}_{\mathrm{GID}}$. The algorithm checks whether S matches (M, ρ) in CT; if S does not match (M, ρ), it aborts; otherwise, the algorithm lets $I = \{x : \rho(x) \in S\} \subseteq \{1, 2, \cdots, l\}$ and computes $\{c_x \in Z_p^*\}$ to let $\sum_{x\in l} c_x M_x = (1, 0, .., 0)$. Thus, $\sum_{x\in l} \lambda_x c_x = s$, $\sum_{x\in l} \omega_x c_x = 0$. The algorithm computes $C_{1,\mathrm{GID}} = \prod_{x\in I} e(C_{1,x} g^{C_{5,x}}, g)^{c_x}$ and $C_{2,\mathrm{GID}} = \prod_{x\in I} (e(\mathrm{CSK}_{\rho(x),\mathrm{GID}}, C_{2,x}) \times e(H_1(\mathrm{GID})^{x_{\mathrm{GID}}}, C_{3,x} g^{C_{6,x}}) e(\mathrm{CSK}'_{\rho(x),\mathrm{GID}}, C_{4,x}))^{c_x}$. The algorithm sends semidecrypted ciphertext $\mathrm{CT}_{\mathrm{GID}} = (C_0, C_{1,\mathrm{GID}}, C_{2,\mathrm{GID}}, M_{SE})$ to MDU.

(9) $\mathrm{UserDec}(\mathrm{USK}, \mathrm{CT}_{\mathrm{GID}}) \longrightarrow m$: this algorithm is performed by MDU; MDU uses USK to decrypt $\mathrm{CT}_{\mathrm{GID}}$ and obtains m. The algorithm computes $C_{1,GID}(C_{2,GID})^{1/x_{\mathrm{GID}}} = e(g, g)^s$. It recovers $Y = C_0/e(g, g)^s$ and computes key $= H_4(Y)$, $m' = \mathrm{SE.Dec}_{\mathrm{key}}(M_{\mathrm{SE}})$. The algorithm checks if there is a redundancy 0^{ϖ} appended after m'; if it is, $m' = m \| 0^{\varpi}$. Then, it checks if $H_1(m') = \mathrm{Ver}$ (Ver can be obtained in blockchain); if it is, the message m can be obtained by truncating ϖ-bit 0 string. Otherwise, CSP is dishonest to return an incorrect semidecrypted ciphertext, and algorithm outputs $\perp$.

(10) $\mathrm{TransDec}_R(\mathrm{PP}, C^R_{(M',\rho')}, S', \mathrm{TK}_{S',\mathrm{GID}}) \longrightarrow \mathrm{CT}_{\mathrm{GID}'}$: CSP performs the $\mathrm{TransDec}_R$ algorithm. It inputs public parameter PP, ciphertext $C^R_{(M',\rho')}$, user attribute set S', and transform key $\mathrm{TK}_{S',\mathrm{GID}}$ and outputs semidecrypted ciphertext $\mathrm{CT}_{\mathrm{GID}'}$. The algorithm checks whether S' matches (M', ρ') in $C^R_{(M',\rho')}$; if S' does not match (M', ρ'), it aborts. Otherwise, the algorithm performs the following steps: CSP let $I' = \{x : \rho'(x) \in S'\} \subseteq \{1, 2, \cdots, l'\}$ and computes $\{c'_x \in Z_p^*\}$ to let $\sum_{x\in l'} c'_x M'_x = (1, 0, .., 0)$. It then computes $C_{1,\mathrm{GID}'} = \prod_{x\in I'} e(C'_{1,x}, g)^{c'_x}$ and $C_{2,\mathrm{GID}'} = \prod_{x\in I'} (e(\mathrm{CSK}_{\rho'(x),\mathrm{GID}'}, C'_{2,x}) e(H_1(\mathrm{GID}')^{x_{\mathrm{GID}'}}, C'_{3,x}) \times e(\mathrm{CSK}'_{\rho'(x),\mathrm{GID}}, C'_{4,x}))^{c'_x}$. The algorithm sends semidecrypted ciphertext $\mathrm{CT}_{\mathrm{GID}'} = (C_0, C'_0, C_6, C_{1,\mathrm{GID}'}, C_{2,\mathrm{GID}'}, M_{\mathrm{SE}})$ to MDU.

(11) $\mathrm{UserDec}_{\mathrm{R}}(\mathrm{PP}, \mathrm{USK}, \mathrm{CT}_{\mathrm{GID}'}) \longrightarrow m$: This algorithm is performed by MDU; it inputs PP, USK, $\mathrm{CT}_{\mathrm{GID}'}$ to decrypt $\mathrm{CT}_{\mathrm{GID}'}$ to obtain m. The algorithm uses USK to compute $\beta = \beta \bullet e(g, g)^{s'}/e(g, g)^{s'}$ and then recovers $Y = C_0/(C_6)^{1/H_2(\beta)}$. MDU computes key $= H_4(Y)$ and $m' = \mathrm{SE.Dec}_{\mathrm{key}}(M_{\mathrm{SE}})$. The algorithm then checks if there is a redundancy 0^{ϖ} appended after m', if it is, $m' = m \| 0^{\varpi}$. Then, it checks if $H_1(m') = \mathrm{Ver}$ (Ver can be obtained in blockchain), if it is, the message m can be acquired by omitting ϖ-bit 0 string. If not, CSP is dishonest to return an incorrect semidecrypted ciphertext and the algorithm outputs $\perp$.

(12) $\mathrm{Revoke}(\mathrm{GID}, R, \mathrm{KT}) \longrightarrow (R', \mathrm{KT}/(\mathrm{GID}, \mathrm{TK}_{S,\mathrm{GID}}))$: the algorithm is performed by CN and CSP; CN first updates R and uploads the latest R' to blockchain. CSP checks the latest R' in blockchain and inputs malicious user's GID in R' and KT and outputs an updated $\mathrm{KT}/(\mathrm{GID}, \mathrm{TK}_{S,\mathrm{GID}})$. If the user's GID needs to be revoked, CSP will find $\{\mathrm{GID}, \mathrm{TK}_{S,\mathrm{GID}}\}$ in KT and delete $\{\mathrm{GID}, \mathrm{TK}_{S,\mathrm{GID}}\}$ in KT.

5.2. Correctness

5.2.1. Correctness of Decryption of Original Ciphertext CT. The CSP checks whether S matches (M, ρ) in CT; if S does not match (M, ρ), it aborts; otherwise, the algorithm lets $I = \{x : \rho(x) \in S\} \subseteq \{1, 2, \cdots, l\}$ and computes $\{c_x \in Z_p^*\}$ to let $\sum_{x\in l} c_x M_x = (1, 0, .., 0)$. Thus, $\sum_{x\in l} \lambda_x c_x = s$, $\sum_{x\in l} \omega_x c_x = 0$. There is $\eta = \delta(x) = F(\rho(x))$. CSP computes formula (3) to acquire $C_{1,\mathrm{GID}}$.

$$\begin{aligned} C_{1,\mathrm{GID}} &= \prod_{x\in I} e\left(C_{1,x} g^{C_{5,x}}, g\right)^{c_x} = \prod_{x\in I} e\left(g^{Q_x} g^{\alpha_{\delta(x)} r_x} g^{\lambda_x - Q_x}, g\right)^{c_x} \\ &= \prod_{x\in I} e\left(g^{\alpha_{\delta(x)} r_x} g^{\lambda_x}, g\right)^{c_x} = \prod_{x\in I} e(g, g)^{\lambda_x c_x} e(g, g)^{\alpha_{\delta(x)} r_x c_x} \\ &= e(g, g)^{s + \sum_{x\in I} \alpha_{\delta(x)} r_x c_x}. \end{aligned} \tag{1}$$

CSP then computes formula (4) to acquire $C_{2,\mathrm{GID}}$.

$$\begin{aligned} C_{2,\mathrm{GID}} &= \prod_{x\in I} \left(e\left(\mathrm{CSK}_{\rho(x),\mathrm{GID}}, C_{2,x}\right) \times e(H_1(\mathrm{GID})^{x_{\mathrm{GID}}}, \right. \\ &\quad \left. C_{3,x} g^{C_{6,x}}) e\left(\mathrm{CSK}'_{\rho(x),\mathrm{GID}}, C_{4,x}\right)\right)^{c_x} \\ &= \prod_{x\in I} \left(e\left(g^{x_{\mathrm{GID}} \alpha_{\delta(x)}} H_1(\mathrm{GID})^{x_{\mathrm{GID}} y_{\delta(x)}} H_3(\rho(x))^{t_{\rho(x)}}, g^{-r_x}\right)\right. \\ &\quad \left. \times e(H_1(\mathrm{GID})^{x_{\mathrm{GID}}}, g^{y_{\delta(x)} r_x} g^{\omega_x}) e\left(g^{t_{\rho(x)}}, H_3(\rho(x))^{r_x}\right)\right)^{c_x} \\ &= \prod_{x\in I} \left(e(\mathrm{g}, \mathrm{g})^{x_{\mathrm{GID}} \alpha_{\delta(x)} (-r_x) c_x}\right) (e(H_1(\mathrm{GID}), g)^{x_{\mathrm{GID}} \omega_x c_x}) \\ &= e(g, g)^{\sum_{x\in I} x_{\mathrm{GID}} \alpha_{\delta(x)} (-r_x) c_x}. \end{aligned} \tag{2}$$

MDU computes formula (5) to acquire $e(g,g)^s$.

$$C_{1,GID}(C_{2,GID})^{1/x_{\mathrm{GID}}} = e(g,g)^{s+\sum_{x\in I}\alpha_{\delta(x)}r_x c_x} \times \left(e(g,g)^{\sum_{x\in I}x_{GID}\alpha_{\delta(x)}(-r_x)c_x}\right)^{1/x_{\mathrm{GID}}}$$
$$= e(g,g)^{s+\sum_{x\in I}\alpha_{\delta(x)}r_x c_x}\left(e(g,g)^{\sum_{x\in I}\alpha_{\delta(x)}(-r_x)c_x}\right) = e(g,g)^s. \tag{3}$$

It recovers $Y = C_0/e(g,g)^s$ and computes $\mathrm{key} = H_4(Y)$, $m' = \mathrm{SE.Dec}_{\mathrm{key}}(M_{\mathrm{SE}})$. MDU checks if there is a redundancy 0^{ϖ} appended after m'; if it is, $m' = m\|0^{\varpi}$. Then, it checks if $H_1(m') = \mathrm{Ver}$ (Ver can be obtained in blockchain); if it is, the message m can be obtained by truncating ϖ-bit 0 string. Otherwise, CSP is dishonest to return an incorrect semidecrypted ciphertext and algorithm outputs $\perp$.

5.2.2. Correctness of Decryption of Reencryption Ciphertext $ReKey_{(S,GID)\longrightarrow(M',\rho')}$. First, compute C_4 and C_5 separately.

$$C_4 = \prod_{x\in I}\left(e\left(C_{1,x}g^{C_{5,x}}, rk_3\right)\right)^{c_x} = \prod_{x\in I}\left(e\left(g^{\alpha_{\delta(x)}r_x}g^{\lambda_x}, g^{H_2(\beta)}\right)\right)^{c_x}$$
$$= \prod_{x\in I}\left(e(g,g)^{\lambda_x H_2(\beta)c_x}e(g,g)^{\alpha_{\delta(x)}r_x H_2(\beta)c_x}\right.$$
$$= e(g,g)^{sH_2(\beta)+\sum_{x\in I}\alpha_{\delta(x)}r_x H_2(\beta)c_x}, \tag{4}$$

$$C_5 = \prod_{x\in I}\left(e(rk_1, C_{2,x})e\left(rk_4, C_{3,x}g^{C_{6,x}}\right)e(rk_2, C_{4,x})\right)^{c_x}$$
$$= \prod_{x\in I}\left(e(\mathrm{g},\mathrm{g})^{\alpha_{\delta(x)}H_2(\beta)(-r_x)}\mathrm{e}(H_1(\mathrm{GID}),\mathrm{g})^{H_2(\beta)\omega_x}\right)^{c_x}$$
$$= \prod_{x\in I}e(\mathrm{g},\mathrm{g})^{\alpha_{\delta(x)}H_2(\beta)(-r_x)c_x}\mathrm{e}(H_1(\mathrm{GID}),\mathrm{g})^{H_2(\beta)\omega_x c_x} \tag{5}$$
$$= e(g,g)^{\sum_{x\in I}\alpha_{\delta(x)}H_2(\beta)(-r_x)c_x}.$$

Then, compute C_6.

$$C_6 = C_4 \bullet C_5 = e(g,g)^{sH_2(\beta)} \tag{6}$$

The CSP checks whether S' matches (M',ρ') in $C^R_{(M',\rho')}$; if S' does not match (M',ρ'), it aborts. Otherwise, CSP performs the following steps: CSP let $I' = \{x : \rho'(x) \in S'\} \subseteq \{1,2,\cdots,l'\}$ and computes $\{c'_x \in Z_p^*\}$ to let $\sum_{x\in I'}c'_x M'_x = (1,0,..,0)$. CSP computes formula (6) to acquire $C_{1,\mathrm{GID}'}$.

$$C_{1,\mathrm{GID}'} = \prod_{x\in I'}e\left(C'_{1,x}, g\right)^{c_x'} = \prod_{x\in I'}e\left(g^{\lambda_x'}g^{\alpha_{\delta'(x)}r'_x}, g\right)^{c'_x}$$
$$= \prod_{x\in I'}e(g,g)^{\lambda_x' c_x'}e(g,g)^{\alpha_{\delta'(x)}r'_x c'_x} = e(g,g)^{s'+\sum_{x\in I'}\alpha_{\delta'(x)}r'_x c'_x}. \tag{7}$$

CSP then computes formula (7) to acquire $C_{2,\mathrm{GID}'}$.

$$C_{2,\mathrm{GID}'} = \prod_{x\in I'}\left(e\left(CSK_{\rho'(x),\mathrm{GID}'}, C'_{2,x}\right)e\left(H_1\left(\mathrm{GID}'\right)^{x_{\mathrm{GID}'}}, C'_{3,x}\right)\right.$$
$$\left.\times e\left(\mathrm{CSK}'_{\rho'(x),\mathrm{GID}}, C'_{4,x}\right)\right)^{c'_x}$$
$$= \prod_{x\in I'}e\left(g^{x_{\mathrm{GID}}'\alpha_{\eta'}}H_1\left(\mathrm{GID}'\right)^{x_{\mathrm{GID}}'y_{\eta'}}H_3(\rho(x))^{t'_{\rho(x)}}, g^{-r_x'}\right.$$
$$\times e\left(H_1\left(\mathrm{GID}'\right)^{x_{\mathrm{GID}}'}, g^{y_{\delta'(x)}r'_x}g^{\omega_x'}\right)$$
$$\left.\cdot e\left(g^{t'_{\rho(x)}}, H_3(\rho(x))^{r_x'}\right)\right)^{c'_x} = e(g,g)^{\sum_{x\in I'}x_{\mathrm{GID}}'\alpha_{\eta'}(-r'_x)c'_x}. \tag{8}$$

MDU computes formula (8) to acquire β.

$$\beta == \frac{\beta\bullet e(g,g)^{s'}}{e(g,g)^{s'+\sum_{x\in I'}\alpha_{\delta'(x)}r'_x c'_x}\left(e(g,g)^{\sum_{x\in I'}x_{GID}'\alpha_{\eta'}(-r'_x)c'_x}\right)^{1/x_{\mathrm{GID}'}}}$$
$$= \frac{\beta\bullet e(g,g)^{s'}}{e(g,g)^{s'}}. \tag{9}$$

Then MDU can recover $Y = C_0/(C_6)^{1/H_2(\beta)}$. MDU computes $\mathrm{key} = H_4(Y)$ and $m' = \mathrm{SE.Dec}_{\mathrm{key}}(M_{\mathrm{SE}})$. The algorithm then checks if there is a redundancy 0^{ϖ} appended after m'; if it is, $m' = m\|0^{\varpi}$. Then it checks if $H_1(m') = \mathrm{Ver}$ (Ver can be obtained in blockchain), if it is, the message m can be acquired by omitting ϖ-bit 0 string. If not, CSP is dishonest to return an incorrect semidecrypted ciphertext and the algorithm outputs $\perp$.

5.3. Security Analysis

Theorem 2. *If the q-DBPBDHE2 assumption holds, then the proposed scheme is statically secure under ROM.*

The proof of Theorem 2 is based on the establishment of Lemmas 3 and 4. Therefore, the proofs of Lemmas 3 and 4 will be given as follows, respectively.

Lemma 3. *If the q-DBPBDHE2 assumption holds, then the RW scheme proposed in [32] is statically secure under ROM.*

Proof. Lemma 3 is proved in [32], so Lemma 3 is valid. □

Lemma 4. *If the RW scheme constructed in [32] is statically secure under ROM, then the proposed scheme is statically secure under ROM.*

Proof. Assuming there is a PPT adversary Adv that can break the proposed scheme with a non-negligible advantage ε, then there must exist a simulator B who can break the RW scheme with advantage ε. Among them, B can break the proposed scheme with Adv and the challenger C in RW. The relevant phases are as follows.

(1) Initialization phase: C sends the public parameter PP in the RW scheme to the simulator B. B performs $GlobalSetup(\lambda, A) \longrightarrow (PP, R)$ algorithm and sends the uploaded public parameter $PP = \{p, G, G_T, g, e, H_1, H_2, H_3, H_4, SE, A, A_\eta\}$ to Adv.

(2) Query phase: Adv does the following query to B.

□

Query 1: Adv chooses some N_θ, Adv asks B for the corresponding public key NPK_θ, B then sends NPK_θ to C. C performs $AuthoritySetup(GP, \theta) \longrightarrow (UPK_\theta, USK_\theta)$ in [32] to generate $UPK_\theta = \{e(g, g)^{\alpha_\theta}, g^{y_\theta}\}$ and sends it to B. After that, B performs $NodeSetup(PP, \theta) \longrightarrow (NPK_\theta, NSK_\theta)$ to generate $UPK_\theta = \{e(g, g)^{\alpha_\theta}, g^{y_\theta}\}$, then B updates public key and sends them to Adv.

Query 2: Adv selects some legitimate users $\{GID_i\}_{i=1}^m$ and asks for their key pairs $\{UPK_i, USK_i\}_{i=1}^m$. B performs $UserKeyGen(PP, GID) \longrightarrow (UPK, USK)$ to generate the corresponding $UPK_i = (g^{x_{GID_i}}, H_1(GID)^{x_{GID_i}})$ and $USK_i = x_{GID_i}$, then sends (UPK_i, USK_i) to Adv.

Query 3: Adv asks B for the transfer key: B first forwards the query to C to get the result output by scheme [32]. Then B performs $TKGen(PP, GID, S, NSK_\theta, UPK) \longrightarrow (TK_{S,GID})$ to generate transfer key $TK_{S,GID}$ and return it to Adv. During the inquiry process, B calculates the transfer key corresponding to $\{S_i, GID\}_{i=1}^n$ in two cases, specifically:

Case 1. For $1 \le i \le m$, attribute $j \in S_i$, B randomly selects $t_j \in Z_p^*$ and computes $CSK_{j,GID_i} = (g^{\alpha_\theta} H_1(GID_i)^{y_\theta} H_3(j)^{t_j})^{x_{GID_i}} = g^{x_{GID_i}\alpha_\theta} H_1(GID_i)^{x_{GID_i} y_\theta} H_3(j)^{t_j x_{GID_i}}$, $CSK'_{j,GID} = H_3(j)^{t_j x_{GID_i}}$. B then acquires the transfer key $TK_{S_i,GID_i} = (CSK_{j,GID_i}, CSK'_{j,GID})_{j \in S_i}$.

Case 2. For $m \le i \le n$, attribute $j \in S_i$, B randomly selects $t_j \in Z_p^*$, $g_j \in G$, computes $CSK_{j,GID_i} = g_j H_3(j)^{t_j}$ and $CSK'_{j,GID} = H_3(j)^{t_j}$. Since $g^{\alpha_\theta} H_1(GID_i)^{y_\theta}$ is in G, there must be an unknown $x_{GID_i} \in Z_p^*$ that can make $g_j = (g^{\alpha_\theta} H_1(GID_i)^{y_\theta})^{x_{GID_i}} = g^{x_{GID_i}\alpha_\theta} H_1(GID_i)^{x_{GID_i} y_\theta}$. Thus, the corresponding transfer key can be obtained as $CSK_{j,GID_i} = g_j H_3(j)^{t_j} = g^{x_{GID_i}\alpha_\theta} H_1(GID_i)^{x_{GID_i} y_\theta} H_3(j)^{t_j}$, $CSK'_{j,GID} = H_3(j)^{t_j}$. B then sends the transfer key $TK_{S_i,GID_i} = (CSK_{j,GID_i}, CSK'_{j,GID})_{j \in S_i}$ to Adv.

Table 2: System function comparison of different schemes.

Scheme	[28]	[24]	[23]	[20]	[19]	Ours
Support ciphertext conversion	×	√	√	√	√	√
Collusion resistant	√	√	√	√	√	√
Decryption outsourcing	√	√	√	×	×	√
No KGC	√	×	×	×	×	√
User revocation	√	×	×	√	√	√
Decryption verification	×	√	×	×	×	√

Query 4: Adv asks B for re-encryption key: B first forwards the query to C to get the result output by scheme [32]. Then, Adv sends users' $GID_i (1 \le i \le n)$, users' attribute sets $S_i \in A$ and a new access policy (M', ρ') to B. B performs $ReKeyGen(PP, TK_{S,GID}, USK, NPK_\eta, S, (M', \rho')) \longrightarrow ReKey_{(S,GID) \longrightarrow (M', \rho')}$ to generate re-encryption key $ReKey_{(S,GID) \longrightarrow (M', \rho')}$. Then B returns $ReKey_{(S,GID) \longrightarrow (M', \rho')}$ to Adv. $TK_{S,GID}$ is generated by $TKGen()$ algorithm.

(3) Challenge Phase: Adv submits two equal-length plaintexts $M_0{}^*$, $M_1{}^*$, and an access policy (M^*, ρ^*) to C. C randomly selects $b \in \{0, 1\}$, executes $Enc()$ algorithm to calculate challenge ciphertext and return it to Adv. The limitation is that the $\{GID_i\}$ $(1 \le i \le m)$ of the MDUs who asked for the private key during the query, their attribute set S_i does not satisfy (M^*, ρ^*).

(4) Guess Phase: Adv outputs guess $b' \in \{0, 1\}$ on b. At this time, B gives different guess values b' according to the different guesses of Adv.

In the above game, Adv can consider that B is the challenger C in scheme [32] The transfer key sent by C corresponds to the user private key generated by *UserKeyGen* algorithm in the scheme [32]. At the same time, B can determine the symmetric key *key* in the proposed scheme according to the value of b'. Mainly, the *key* corresponds to the message M input by the *Encrypt* algorithm in the scheme [32]. Since the RW scheme is statically secure, the proposed scheme is also statically secure. Lemma 4 is proved.

In summary, from the proofs of Lemmas 3 and 4, it can be proved that Theorem 2 is also valid. Therefore, the proposed scheme is statically secure under ROM.

6. Performance Analysis

In this section, the function and efficiency of the proposed scheme and other related schemes [19, 20, 23, 24, 28] are evaluated and compared. We compared the aforementioned 5 schemes with our scheme in terms of system functions. The specific comparison is shown in Tables 2 and 3.

Table 3: System efficiency comparison of different schemes.

Scheme	[28]	[24]	[23]	[20]	[19]	Ours
Encrypt	$(7l+4)E+P$	$(6l+2)E$	$(7l+2)E$	$(5l+1)E$	$(6l+1)E$	$(6l+1)E$
Original ciphertext decrypt	E	$(3\|I\|+1)P+\|I\|E$	E	$(3\|I\|+2)P+E$	$2\|I\|P+2\|I\|E$	E
Reencrypt key	–	$2\|S\|+5E$	$\|S\|+l+6E$	$(2\|S\|+2l+3)E$	$4\|S\|E$	$(2\|S\|+8l+8)E$
Reencrypt ciphertext decrypt	–	$4P+2E$	$2E$	$3\|I\|P+2P+E$	$2\|I\|P+2\|I\|E$	$2E$
Total	$(7l+5)E+P$	$(6l+\|I\|+9)E+(3\|I\|+5)P+2\|S\|$	$(7l+11)E+\|S\|+l$	$(5l+2\|S\|+6)E+(6\|I\|+4)P$	$4\|I\|P+(4\|S\|+4\|I\|+6l+1)E$	$(2\|S\|+14l+12)E$

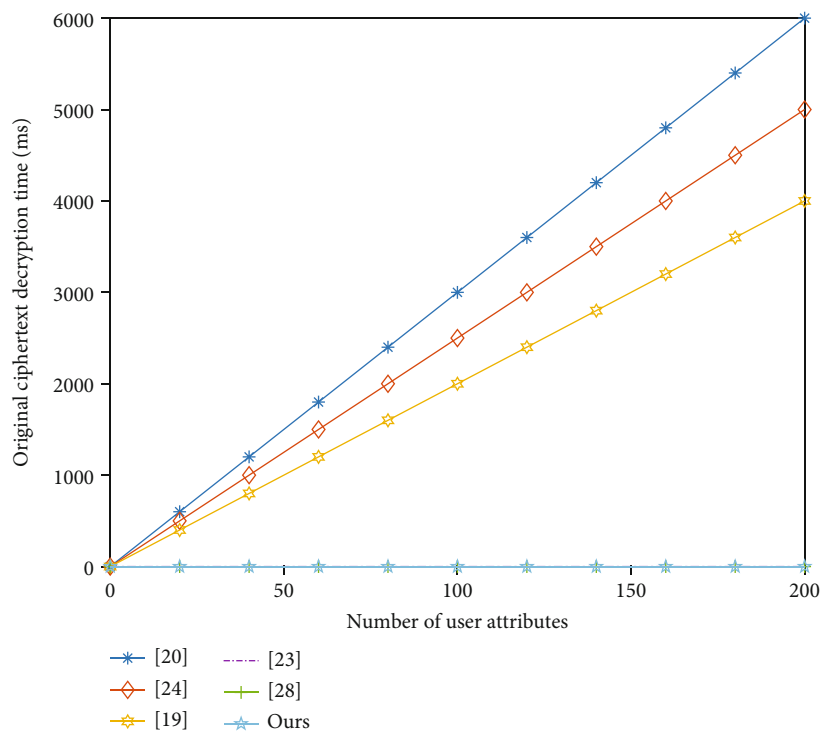


Figure 3: User original ciphertext decryption time.

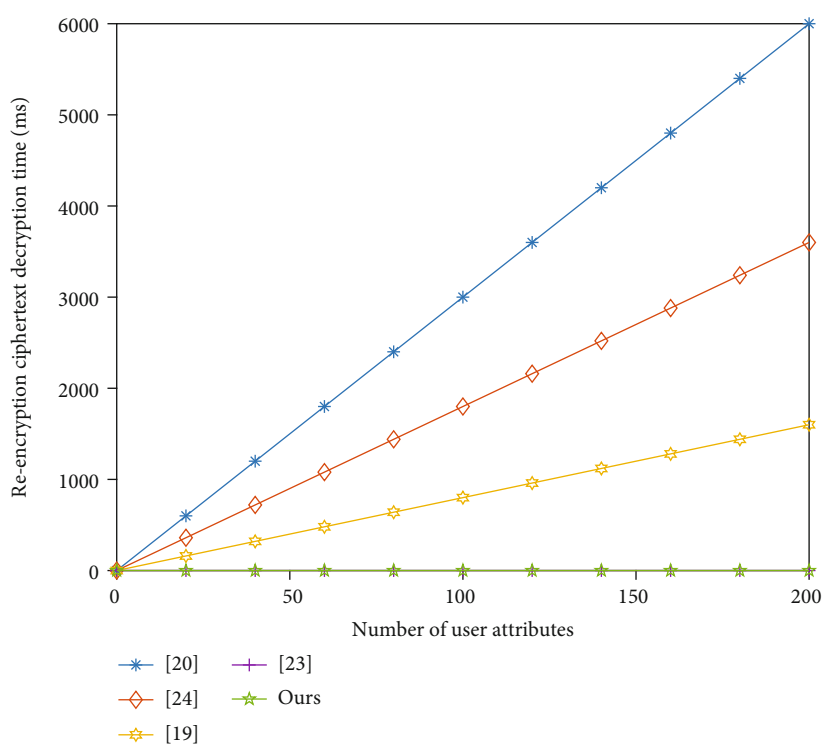


Figure 4: User reencrypted ciphertext decryption time.

Figures 3–5 shows the operating time of different algorithms. We use Java language in the Windows10 environment, and Java Pairing Based Cryptography (JPBC) is used as the cryptography library. G and G_T are selected from the elliptic curve $y2 = x3 + x$. $e(g, g)$ is a symmetric bilinear map. The length of the elements in G and G_T is 1024 bits. The AES encryption algorithm is used as a symmetric-key encryption algorithm. The hardware is AMD R5 4600u CPU, 2.1 GHz frequency, and 8GB memory.

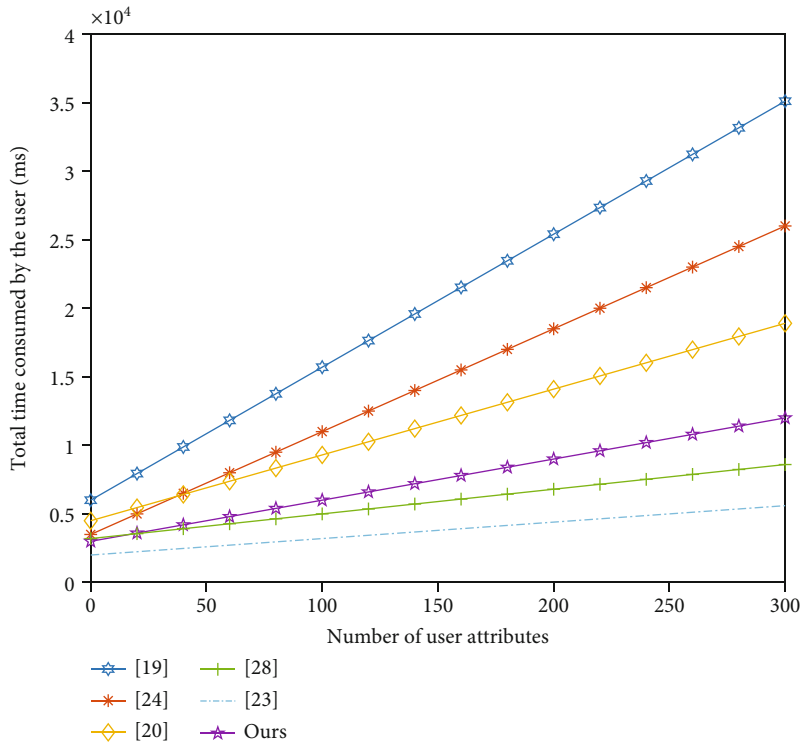


Figure 5: Total computational cost.

Table 2 shows the function comparison; schemes [19, 20, 23, 24] support ciphertext reencryption and collision resistance. Compared with the 4 schemes, the proposed scheme has the most comprehensive functions. Though scheme [19, 20] support ciphertext reencryption, they do not support outsourced decryption. Thus, MDU will face huge decryption costs. Although schemes [19, 20, 28] realize user revocation, the keys of unrevoked users need to be updated when the revocation occurs, which brings additional computational overhead. The user private keys in schemes [19–24] are all generated by KGC; once KGC is compromised, it will cause serious information security incidents. In fact, there exists no fully trusted KGC in the world. The proposed scheme does not need a KGC, and the user private keys are generated by users. Besides, consortium blockchain is introduced to ensure medical data will not be tampered with; consensus nodes work together to ensure blockchain operation. Schemes [20, 24] cannot realize user revocation. Even if a user leaves the system, his private key can still decrypt the ciphertext in the system, which may lead to abuse of key authority. Although schemes [20, 28] support decryption outsourcing, they lack decryption verification. This makes data users cannot distinguish whether the CSP has tampered with the result of outsourced decryption. In addition, the scheme [23] only realizes the reencryption from ABE to IBE, not the reencryption from ABE to ABE. ABE can realize "one-to-many" encryption, while IBE can only realize "one-to-one" encryption. In summary, the proposed scheme is practical in functions and is suitable for cloud storage environments.

Table 3 shows the system efficiency comparison of different schemes. For simplicity, $|S|$ denotes the number of user's attributes, l denotes the number of rows in M, $|I|$ denotes the number of rows used for decryption in M, P denotes the bilinear pairing operation in G, and P denotes the exponential operation in G. Since the calculation cost of the schemes is related to l, $|I|$, $|S|$, the variables are unified for the convenience of discussion. Let the user attribute be a subset of the policy attribute; the user attributes always satisfy the access policy so that $|I|$ is consistent with $|S|$. Thus, the computational cost of the solution is related to l and $|S|$.

It can be seen from the results given in Table 2 and Figures 3–5, schemes [23, 28] have a lower computational cost. The main reason is that in scheme [23], the ciphertext after reencryption is in IBE form, not ABE form, which makes its reencryption key cost is low. Scheme [28] does not support reencryption, so it has a low computational cost. Meanwhile, although schemes [20, 24] share the computational cost of the reencryption key to the trusted center, compared with the proposed scheme, the above two schemes have higher computational costs in the decryption stage. The computational cost of scheme [19] is the highest in all phases. Finally, from the perspective of total computational cost, the users in the proposed scheme have a lower computational overhead than users in schemes [19, 20, 24]. Therefore, the proposed scheme is efficient in computing.

7. Conclusion

This paper proposed a medical data sharing scheme supporting ciphertext reencryption. The proposed scheme can employ attribute-based proxy reencryption technology to allow the cloud to reencryption ciphertext. The reencryption key was provided by a user with decryption authority and

the cloud cannot obtain the plaintext. It can effectively reduce the bandwidth and time cost caused by the round-trip transmission of ciphertext. Simultaneously, the proposed scheme makes use of a consortium blockchain to store public keys and public parameters and update the revocation list. The data stored on the blockchain is secure and cannot be tampered with. Consensus nodes jointly maintain the consortium blockchain, which can prevent security and privacy issues caused by KGC's overcentralization. Ultimately, based on the q-DBPBDHE2 difficult problem assumption, the proposed scheme is proved to be statically secure. Hence, the proposed scheme is practical for secure access control and sharing of medical data in the cloud storage environment.

Data Availability

All data, models, and codes generated or used during the study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (Grant No. 61972094) and the Foundation of State Key Laboratory of Information Security (Grant No. 2021-ZD-02).

References

[1] T. Wood, K. Ramakrishnan, P. Shenoy, J. Merwe, and J. Hwang, "CloudNet: dynamic pooling of cloud resources by live WAN migration of virtual machines," *IEEE-ACM Transactions on Networking*, vol. 23, no. 5, pp. 1568–1583, 2015.

[2] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: state-of-the-art and research challenges," *Journal of Internet Services and Applications*, vol. 1, no. 1, pp. 7–18, 2010.

[3] B. Singh, S. Dhawan, A. Arora, and A. Patail, "A view of cloud computing," *International Journal of Computers & Technology*, vol. 4, no. 1, pp. 50–58, 2013.

[4] S. Xu, J. Ning, Y. Zhang, G. Xu, X. Huang, and R. Deng, "A secure EMR sharing system with tamper resistance and expressive access control," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 3, pp. 1–223, 2021.

[5] A. Abbas and S. Khan, "A review on the state-of-the-art privacy-preserving approaches in the e-health clouds," *IEEE Journal of Biomedical Health Informatics*, vol. 18, no. 4, pp. 1431–1441, 2014.

[6] G. Anthes, "Security in the cloud," *Communications of the ACM*, vol. 53, no. 11, pp. 16–18, 2010.

[7] C. Chu, W. Zhu, J. Han, J. Liu, J. Xu, and J. Zhou, "Security concerns in popular cloud storage services," *IEEE Pervasive Computing*, vol. 12, no. 4, pp. 50–57, 2013.

[8] N. Kaaniche and M. Laurent, "Data security and privacy preservation in cloud storage environments based on cryptographic mechanisms," *Computer Communications*, vol. 111, pp. 120–141, 2017.

[9] A. Sari, "A review of anomaly detection systems in cloud networks and survey of cloud security measures in cloud storage applications," *Journal of Information Security*, vol. 6, no. 2, pp. 142–154, 2015.

[10] A. Almutairi, M. Sarfraz, S. Basalamah, A. Walid, and A. Ghafoor, "A distributed access control architecture for cloud computing," *IEEE Software*, vol. 29, no. 2, pp. 36–44, 2012.

[11] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," in *International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT 1998)*, pp. 127–144, Nyberg, Kaisa, 1998.

[12] S. Nakamoto, "Bitcoin: a peer-to-peer electronic cash system," 2008, https://bitcoin.org/bitcoin.pdf.

[13] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Annual international conference on the theory and applications of cryptographic techniques (EUROCRYPT 2005)*, pp. 457–473, Aarhus, Denmark, 2005.

[14] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM conference on Computer and communications security (CCS 2006)*, pp. 89–98, Alexandria, VA, USA, 2006.

[15] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *2007 IEEE symposium on security and privacy (SP 2007)*, pp. 321–334, Berkeley, CA, USA, 2007.

[16] B. Waters, "Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization," in *International Workshop on Public Key Cryptography*, pp. 53–70, Springer, Berlin Heidelberg, 2008.

[17] X. Liang, Z. Cao, L. Huang, and J. Shao, "Attribute based proxy re-encryption with delegating capabilities," in *Proceedings of the 4th International Symposium on Information, Computer, and Communications Security (ASIACCS 2009)*, pp. 276–286, Sydney, Australia, 2009.

[18] K. Liang, L. Fang, W. Susilo, and D. Wong, "A ciphertext-policy attribute-based proxy re-encryption with chosen-ciphertext security," in *2013 5th International Conference on Intelligent Networking and Collaborative Systems (INCOS 2013)*, pp. 552–559, Xian, China, 2013.

[19] Y. Yang, H. Zhu, H. Lu, J. Weng, and R. Choo, "Cloud based data sharing with fine-grained proxy re-encryption," *Pervasive and Mobile Computering*, vol. 28, no. 2, pp. 122–134, 2016.

[20] X. Feng, C. Li, D. Li, Y. Fang, and Q. Shen, "Fully secure hidden ciphertext-policy attribute-based proxy re-encryption," in *International Conference on Information and Communications Security (ICIC 2017)*, pp. 192–204, Beijing, China, 2017.

[21] H. Ma, R. Zhang, Z. Wan, L. Yao, and S. Lin, "Verifiable and exculpable outsourced attribute-based encryption for access control in cloud computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 14, no. 6, pp. 679–692, 2017.

[22] S. Xu, Y. Li, and R. Deng, "Lightweight and expressive fine-grained access control for healthcare internet-of-things," *IEEE Transactions on Cloud Computing*, vol. 99, no. 6, pp. 190–207, 2019.

[23] H. Deng, Z. Qin, Q. Wu, Z. Guan, and Y. Zhou, "Flexible attribute-based proxy re-encryption for efficient data sharing," *Information Sciences*, vol. 511, no. 11, pp. 94–113, 2020.

[24] A. Paul, S. Selvi, and P. Rangan, "Efficient attribute-based proxy re-encryption with constant size ciphertexts," in

International Conference on Cryptology in India (INDOCRYPT), pp. 644–665, Bangalore, India, 2020.

[25] S. Xu, J. Ning, X. Huang, Y. Li, and G. Xu, "Untouchable once revoking: a practical and secure dynamic EHR sharing system via cloud," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 4, pp. 1–240, 2021.

[26] S. Huh, S. Cho, and S. Kim, "Managing IoT devices using blockchain platform," in *International Conference on Advanced Communication Technology (ICACT 2017)*, pp. 464–467, Pyeong Chang, Republic of Korea, 2017.

[27] B. Liu, X. L. Yu, S. Chen, X. Xu, and L. Zhu, "Blockchain based data integrity service framework for IoT data," in *IEEE International Conference on Web Services (ICWS 2017)*, pp. 468–475, Honolulu, HI, USA, 2017.

[28] S. Hu, C. Cai, Q. Wang, C. Wang, and D. Ye, "Augmenting encrypted search: a decentralized service realization with enforced execution," *IEEE Transactions of Dependable and Secure Computing*, vol. 16, no. 6, pp. 2569–2581, 2019.

[29] S. Xu, J. Ning, Y. Li, Y. Zhang, G. Xu, and X. Huang, "Match in my way: fine-grained bilateral access control for secure cloud-fog computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 17, no. 6, pp. 2193–2207, 2020.

[30] Q. Li, J. Ma, R. Li, X. Liu, J. Xiong, and D. Chen, "Secure, efficient and revocable multi-authority access control system in cloud storage," *Computers & Security*, vol. 59, no. 5, pp. 45–59, 2016.

[31] M. Bellare, "Random oracles are practical: a paradigm for designing efficient protocols," in *Proceedings of the 1st ACM conference on Computer and communications security (CCS 1993)*, pp. 62–73, Fairfax, USA, 1993.

[32] Y. Rouselakis and B. Waters, "Efficient statically-secure large-universe multi-authority attribute-based encryption," in *International Conference on Financial Cryptography and Data Security (FC 2015)*, pp. 315–332, Sanjuan, Rico, 2015.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 1137130, 10 pages
https://doi.org/10.1155/2022/1137130

Research Article

Housing Rental Scheme Based on Redactable Blockchain

Chunli Wang,[1,2] **Wensheng Jia**,[1] **and Yuling Chen**[1,3]

[1]*State Key Laboratory of Public Big Data, College of Mathematics and Statistics, Guizhou University Guizhou Guiyang, 550025, China*
[2]*Guangxi Key Laboratory of Cryptography and Information Security, Guilin University of Electronic Technology, 541000, China*
[3]*Blockchain Laboratory of Agricultural Vegetables, Weifang University of Science and Technology, Weifang, Shandong 262700, China*

Correspondence should be addressed to Wensheng Jia; wsjia@gzu.edu.cn

Received 19 December 2021; Revised 25 January 2022; Accepted 8 February 2022; Published 10 March 2022

Academic Editor: Jinguang Han

With the increase of the housing population, it is very important for renters to find credible housing information through rental platforms. However, due to the imperfection of relevant laws and insufficient supervision of the housing rental market, problems have emerged one after another. In this article, we propose a housing rental scheme based on redactable blockchain, introducing a trusted third party to verify the homeowner rental listings, and fundamentally preventing false listings from flowing into the housing rental market. Specifically, a trusted third party similar to the housing authority will issue a verifiable claim for the homeowners listing, and then the accounting node of the consortium blockchain will publish the verified listing information on the blockchain so as tenants to make inquiries according to their needs. At the same time, we use the new chameleon hash to build a redactable blockchain. After the homeowner proposes to change the housing price or housing information, the content of the historical block is modified as required. Thus, to a certain extent, the changes in housing prices can be monitored to curb chaos in the housing rental market.

1. Introduction

With the development of social economy and changes of people's concepts, not only the housing rental market's demand has increased significantly but also the number of renters has increased year by year. Nevertheless, the housing rental market is experiencing frequent occurrences. For example, the proliferation of false listings, the asymmetry of information in the leasing market, the serious leakage of customer information, and the low efficiency of industry transactions have caused chaos in the housing leasing market. The abovementioned problems have severely affected the vigorous development of the housing rental market and increased the additional burden of renters. Solving the various problems in the housing rental market plays a very important role in meeting the housing needs of residents and promoting stable economic and social development.

Therefore, we propose a housing leasing program based on blockchain. The blockchain is essentially a decentralized database. There is no centralized third-party organization in the blockchain system. Blockchain transactions are generated by participating entities. The transactions are packaged into blocks and added to the blockchain by miners in chronological order [1]. Participating entities in the blockchain need to update the block regularly and store it. Public chains are mature in Bitcoin and Ether, but private chains have emerged due to the real demand. There are selfish mining attacks in blockchain where attackers compromise the blockchain system by exploiting the vulnerability of the consensus mechanism [2]; while semiselfish mining attacks are proved by Li et al. through simulation that semiselfish mining is impossible in practice [3]. At present, blockchain has been used in many scenarios such as medical care, finance, and the Internet of Things. However, the use of blockchain in these scenarios cannot be directly applied to the housing rental market.

In the blockchain, information is disseminated according to the public key and private key, an asymmetric digital

encryption technology to achieve mutual trust between the two sides of the transaction. The public key and the key appear in pairs, and the public key is broadcasted for other users to know, while the private key is kept by the users themselves. In terms of public key setting, Chen et al. proposed a dynamic multikey FHE scheme based on the LWE assumption of public key setting [4], which uses the product of the public key and the product of a uniform random matrix to hide the secret key and improve the security of the public and private keys.

Accordingly, based upon the foregoing, we propose a plan to combine the blockchain with the housing leasing market to solve various problems in the leasing market. By introducing a trusted third-party verification agency, a weakly centralized blockchain is built to solve the problems of information asymmetry, illegal subletting, and tenant information leakage in the housing rental market. The homeowner submits the information of the house to be rented to a trusted third party for verification. The third-party agency verifies the housing information to ensure the authenticity and reliability of the rental housing and sends a verifiable certification claim to the verified homeowner. The homeowner sends the basic information and certification claim of the rented house to the accounting node. The accounting node verifies the certification claim and publishes the verified housing rental information on the blockchain for tenants to query. The tenant can query the rental information in the blockchain by paying a certain token RentCoin. When the homeowner and the tenant reach an agreement on renting a house, they automatically start the transaction by triggering a smart contract. After the lease transaction is generated, the homeowner and tenant pay a certain amount of RentCoin to the consortium blockchain nodes as a transaction fee. In this process, the tenant's information is only known to the homeowner. So, there is no leakage of the tenant's information.

The main contributions of this paper can be summarized as follows:

(1) We propose a scheme for house leasing based on blockchain technology. By introducing a trusted third party, we ensure the authenticity of rental information. Solve the problem of information asymmetry between homeowners and tenants in the housing rental market. In addition, our scheme has no housing intermediary, which saves the cost of housing intermediary

(2) The introduction of smart contract technology has improved the fairness of house leasing transactions. The conditions for triggering the smart contract are met, then the transaction is automatically executed, which improves the fairness of industry transactions

(3) Introduce a new chameleon hash to the consortium blockchain to reduce user interaction and realize the editability of the consortium blockchain

The rest of this paper is organized as follows. We review related work in Section 2. Section 3 introduces the relevant background knowledge that will be used in this article. We introduce the system model, threat model, and design goals in Section 4. Section 5 describes our proposed scheme in detail. Finally, Section 6 concludes the article.

2. Related Work

The chameleon hash and signature were proposed by Krawczyk and Rabin in 2000 to prohibit the recipient from freely disclosing the content of the signature information to any third party without the consent of the signatory [5]. But in the scheme proposed by Krawczyk and Rabin, there is the problem of key exposure. Therefore, some people have proposed whether it is possible to construct a chameleon hash and signature without key exposure. In 2004, Chen et al. first proposed an identity-based keyless exposure scheme. In this scheme, the author introduces a three-trapdoor mechanism, each transaction has its own trapdoor key, and no third party can create a trapdoor key that has not appeared before. Therefore, this scheme does not have the problem of key exposure [6].

In 2008, Nakamoto explained the blockchain technology in the article Bitcoin: A Peer-to-Peer Electronic Cash System [7]. Since blockchain technology was proposed, it has been applied to many fields such as vehicle transportation, securities finance, medical, and health care. The application of blockchain technology in specific scenarios is also becoming more mature. For example, Kang and others applied blockchain to vehicle edge computing and secure data sharing in 2019. Using consortium blockchain and smart contracts, the author built a secure vehicle data storage and sharing system [8]. Subramaniam et al. used blockchain technology to prevent credit fraud in 2020 and combined with Near Field Communication (NFC) to demonstrate the effect of this scheme [9]. Xu and others proposed a blockchain-based large-scale health data privacy protection scheme in 2019. The solution introduces the star file system (IPFS) to solve the storage problem of large-scale health data and realizes the privacy protection of health data based on blockchain technology. The author fully explained the program and gave the specific details of its implementation [10]. Chen et al. proposed a source location privacy (SLP) protection scheme (PSSPR) based on sector domain phantom routing in WSNs in 2021. The scheme has good performance in security [11].

In 2019, Ashritha et al. applied chameleon hash to blockchain and proposed redactable blockchain for the first time [12]. The introduction of chameleon hash can modify the content of the block without changing the block hash and other block contents. The master key in this scheme is dispersed among the master nodes based on secret sharing. When the content of a block is to be modified, the master node holding the master key share reconstructs the master key by secure multiparty computation to achieve the modification of the block content. In 2020, Xu et al. use editable blockchain for the management and authentication of mobile network identity. The article enables users to securely master their personal identity information by introducing self-sovereign identity. What is more, it empowers easier

and faster identity verification between users and network operators through editable blockchain and permits operators to dynamically revoke users through chameleon hashing [13]. The proposed redactable blockchain provides more scenarios for the practical application of blockchain.

In this paper, we introduce a new chameleon hash to achieve the editable character of the consortium blockchain as a way to build a housing rental platform. The method can solve the problems of information asymmetry and privacy leakage brought by the housing rental market to a certain extent.

3. Preliminaries

In this section, we introduce the background knowledge that will be used in this paper, including secret sharing, identity-based encryption, the new chameleon hash algorithm, and smart contracts.

3.1. Secret Sharing. Secret sharing was first proposed by Sharmir and Blakley in 1979 as the rational distribution of shared secrets among a group of users in order to achieve a shared ownership of the secret by all group members [14, 15]. Later in 1985, Chor et al. introduced the concept of verifiable secret sharing [16]. The concept was proposed considering the existence of dishonest participants in the secret sharing scheme, and it added some public commitment and verification algorithms to the secret sharing scheme as a way to detect dishonest users falsifying their secret shares [17–19]. Verifiable secret sharing assumes the existence of a secret S that is divided in a specific way into n shares $S_1, S_2, \cdots, S_n$ and securely given to n individuals for safekeeping, and the algorithmic process is the following two steps.

(1) *Secret Distribution Algorithm Share*$(S) = (S_1, S_2, \cdots, S_n)$. Given a secret S, n shares are randomly generated in a specific way

(2) *Secret Recovery Algorithm Rec*$(S_1, S_2, \cdots, S_n) = S \cup \perp$. Given the secret share $S_1, S_2, \cdots, S_n$, recover the secret S or return the outlier $\perp$

A (k, n)-verifiable secret sharing scheme needs to satisfy the following two requirements.

(1) *Verifiability*. A user can test whether a secret share is a valid share after receiving it. If the share is valid, the secret recovery algorithm can output a unique secret S. Any k valid shares out of n shares or more than k valid shares can recover the secret S

(2) *Unpredictability*. For polynomial-time algorithms, any less than k valid shares among n shares cannot fully recover the secret S

If k is larger, the higher the security of the secret sharing scheme. When $k = n$, all the secret sharers need to reconstruct the secret S together.

3.2. Self-Sovereign Identity. Self-sovereign identity (SSI) was proposed by Toth and Anderson-Priddy in 2019 [20], where each identity is fully owned, controlled, and managed by the owner of the identity. By virtue of a self-sovereign identity, users have full control over how their personal information is kept and used. Users with SSIs can store their data locally without relying on a central data repository. A service provider or organization can only access information about a user with the consent of SSI's owner. Thus, self-sovereign identity provides users with added security and flexibility.

3.3. Chameleon Hash. The chameleon hash was proposed by Krawczyk and Rabin in 2000 to prohibit the recipient from disclosing the contents of a signed message to any third party at will without the consent of the signer. The chameleon hash function is a cryptographic hash function that contains trapdoor information. The manager who has the trapdoor information can generate hash collisions based on the trapdoor information [1]. The chameleon hash satisfies the following security requirements.

(1) *Collision Resistance*. When inputting the public key hk, there is no effective algorithm to find the pair (m_1, r_1), (m_2, r_2) such that $\text{Hash}(hk, m_1, r_1) = \text{Hash}(hk, m_2, r_2)$, where $m_1 \neq m_2$

(2) *Trapdoor Collision*. At the input of trapdoor key tk, there exists a valid algorithm for any m_1, r_1, given m_2 can find r_2 such that $\text{Hash}(hk, m_1, r_1) = \text{Hash}(hk, m_2, r_2)$

(3) *Semantic Security*. For arbitrary messages m_1、m_2, the probability distributions of $\text{Hash}(hk, m_1, r_1)$ and $\text{Hash}(hk, m_2, r_2)$ are indistinguishable. In particular, for a randomly chosen r, no information about m can be inferred from $\text{Hash}(hk, m, r)$

Ateniese et al. replaced the hash of the block header in the blockchain with a chameleon hash to make the blockchain editable [21]. In 2018, Li et al. proposed a new chameleon hash for consortium blockchains, which gives each user of the consortium blockchain the right to modify the historical blocks without the need for multiparty interaction. The modification can be completed when the conditions for modification triggering are satisfied, i.e., a user is randomly selected according to the rules [22]. The random number $r = (r_1, r_2, \cdots, r_n)$ in the new chameleon hash function and the trapdoor key $(x_1, x_2, \cdots, x_n)$ are held by n users $P_1, P_2, \cdots, P_n$ in the consortium chain, respectively. The public keys of the n users are $(HK_1, HK_2, \cdots, HK_n)$, respectively. The new chameleon hash function is constructed as follows.

(1) *Setup*(λ). Input security parameter λ, construct large prime p, q satisfying security parameter λ, where $p = kq + 1$, select element g of order q in multiplicative cyclic group Z_p^*, output public parameter $pp = (p, q, g)$

(2) *KeyGen*(pp). Input the public parameter pp, randomly select the indices $x_1 \in Z_q^*, x_2 \in Z_q^*, \cdots, x_n \in Z_q^*$

, and calculate $h_1 = g^{x_1}, h_2 = g^{x_2}, \cdots, h_n = g^{x_n}$. Then the trapdoor private key $TK = (x_1, x_2, \cdots, x_n)$, the public key $HK = (g, h_1, h_2, \cdots, h_n)$

(3) $Hash(HK, m, r)$. Input hash public key $HK = (g, h_1, h_2, \cdots, h_n)$, message m and random number $r = (r_1, r_2, \cdots, r_n)$, output chameleon hash $CH = g^m h_1^{r_1} h_2^{r_2} \cdots h_n^{r_n} modp$

(4) $Forge(TK_i, m, r, m')$. Input trapdoor private key $TK_i = x_i$, message m, random number $r = (r_1, r_2, \cdots, r_n)$, message m'. Then according to $CH = g^{m'} h_1^{r_1} \cdots h_i^{r_i'} \cdots h_n^{r_n} modp$, we can get $m + x_1 r_1 + \cdots + x_i r_i + \cdots + x_n r_n = m' + x_1 r_1 + \cdots + x_i r_i' + \cdots + x_n r_n modq$, and we can calculate that $r_i' = (m - m' + x_i r_i) x_i^{-1} modq$

A user P_i with trapdoor key x_i can run the forge algorithm to find collisions such that $\text{Hash}(HK, m, r) = \text{Hash}(HK, m', r')$. Applying the new chameleon hash to the consortium blockchain can effectively reduce the interaction between users. Because each user in the consortium chain has the trapdoor key of the chameleon hash function, each can use their own trapdoor key to compute the new random number r' corresponding to the message m such that $\text{Hash}(m, r) = \text{Hash}(m', r')$. However, the modified block published by the user must satisfy the following two conditions to be accepted: (1) more than half of the users in the system agree to the modification, i.e., it contains the votes and signatures of more than half of the users; (2) the user who modifies the block is indeed the one corresponding to the smallest hash value obtained by computing the Lagrange interpolation formula.

3.4. n Noncooperative Game. Let $N = 1, \cdots, n$ be the set of insiders, $\forall i \in N$. The pure strategy set of insider i is the finite set $S_i = s_{i1}, \cdots, s_{im_i}$, and the mixed strategy set is $X_i = \{x_i = (x_{i1}, \cdots, x_{im_i}): x_{ik_i} \geq 0, k_i = 1, \cdots, m_i, \sum_{k_i=1}^{m_i} x_i k_i = 1\}$. When each inning i chooses the pure strategy $s_{ik} \in S_i$, $i = 1, \cdots, n$, the inning i gets paid as the real number $R_i(s_1 k_1, \cdots, s_n k_n)$, and note that $X = \prod_{i=1}^{n} X_i, \forall x = (x_1, \cdots, x_n) \in X$. When each inning i chooses a mixed strategy $x_i = (x_{i1}, \cdots, x_{im_i}) \in X_i$ (i.e., inning i chooses the pure strategy $s_{i1}, \cdots$, with probability x_{i1}, and the pure strategy s_{im_i} with probability x_{im_i}) with $i = 1, \cdots, n$, and assumes that their choices are independent, then inning i gets an expected payoff that is real

$$f_i(x_1, \cdots, x_n) = \sum_{k_1=1}^{m_1} \cdots \sum_{k_n=1}^{m_n} R_i(s_1 k_1, \cdots, s_n k_n) \prod_{i=1}^{n} x_i k_i. \quad (1)$$

$\forall i \in N$, noting $\hat{i} = N \setminus \{i\}$. Each inning is rational and wants to get the maximum benefit for itself. Therefore, if there exists $x^* = (x_1^*, \cdots, x_n^*) \in X$ such that $\forall i \in N$, there is $f_i(x_i^*, x_{\hat{i}}^*) = \max_{u_i \in X_i} f_i(u_i, x_{\hat{i}}^*)$. Then x^* the Nash equilibrium point of this n-person noncooperative game, at which point each inning cannot make itself more profitable by individually changing its strategy [23, 24].

3.5. Smart Contract. Smart contracts are an idea first proposed by Szabo in 1994 and published on the website of the Extropy Institute [25]. A smart contract is defined as an event-driven, stateful program that runs on top of a replicated and shared ledger that holds the assets on the ledger [26]. Smart contracts can improve the fairness of housing lease transactions by simplifying issues such as the signing of housing lease contracts and subsequent defaults.

4. System Model, Threat Model, and Design Goals

4.1. System Model. As shown in Figure 1 below, the specific description is shown as follows.

4.1.1. House Owner. The homeowner is the owner of the home. They certify the home for rent through a certification agency. A landlord may have more than one home to rent, so a landlord can have multiple verifiable certification statements at the same time. The innkeeper stores the relevant certification statements locally and presents them to the tenant as the tenant needs them.

4.1.2. Tenant. Tenants query the blockchain to get the desired property information. And sign a lease contract with the landlord based on smart contract.

4.1.3. Certification Body. Certification bodies are distributed trusted entities, such as housing authorities. They issue a verifiable certification statement against the homeowner's listing that includes the certification authority's signature for others to verify. There is a relationship of trust between the certification authority and the homeowner. All the certification bodies form a consortium to maintain the consortium blockchain. When the owner pays the appropriate number of tokens RentCoin, the certification body can modify the contents of the blockchain with trapdoor information according to the owner's needs while keeping the block hash value unchanged.

4.1.4. Blockchain. It is a consortium blockchain maintained by accounting nodes for publishing the listing information and verifiable authentication claims of the homeowner. Any tenant can read the information on the blockchain. The transactions in each block form the Merkle hash tree of the respective block, where the first level of the hash tree is a new chameleon hash. An accounting node can modify the content of a transaction while keeping the block header unchanged.

4.1.5. Accounting Node. This node is a special node in the consortium blockchain where a trusted third-party acts as the accounting node for the consortium chain. It verifies the validity of the signatures of verifiable claims issued by the certification authority. At each time period, the consortium blockchain consisting of certification authorities chooses a leader, which is rotated by the certification authorities. The leader needs to pack valid transactions from the homeowner, generate new blocks, and join them to the consortium blockchain.

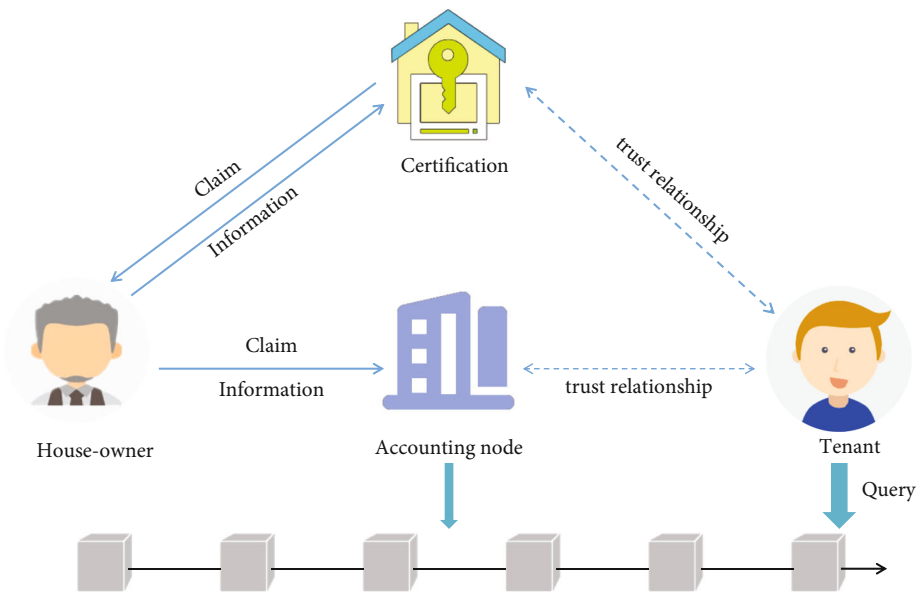


Figure 1: System model.

First, the homeowner sends the encrypted house information to the relevant certification authority according to the demand for rental. Second, the authentication agency decrypts and authenticates the information of the house. It also sends a verifiable claim to the authenticated owner. Finally, the owner sends the encrypted verifiable claims and the rental information of the house to the accounting node of the consortium blockchain. The leader of the consortium chain broadcasts the legitimate transaction to the other accounting nodes after verification. After they verify the signature, the leader adds the rental information to the consortium blockchain as a transaction for tenants to query and search.

4.2. Threat Model. We assume that a secure channel exists between the individual nodes. The certification authority strictly enforces the certification statement and authenticates truthfully. There are active and passive adversaries in the system. Passive adversaries obtain transaction data by eavesdropping on homeowners or analyzing leases between homeowners and tenants. Unlike passive adversaries, active adversaries may interrupt lease transactions between homeowners and tenants.

In addition, we assume that there are $3f + 1$ accounting nodes in the consortium chain to maintain the blockchain and that there are no more than f malicious nodes in the consortium chain.

4.3. Design Goals. Our goal is to achieve privacy and security, information symmetry, transaction fairness, and legal subletting in the rental market. Therefore, our design objectives are shown below.

4.3.1. Confidentiality of Property Information. The owner sends the ownership and other information of the house to the verification agency through a secure channel, and no third party can get the encrypted house information in the process, i.e., no agent can get the information of the rental property.

4.3.2. Authorized Home Ownership Certification. The certification claim issued by the certification agency must determine the match and authenticity of the homeowner's identity information and the housing information. It ensures the authenticity of the housing information as well as reduces the information asymmetry between homeowners and tenants.

4.3.3. Transaction Security. The rental transactions between landlords and tenants are only accessible to the house owner and dwellers themselves. No counterparty has access to the transaction information between them. There is also no third party to modify the transaction without the consent of the owner and tenant.

4.3.4. Privacy. Only the landlord and renter know their private information during the transaction. No third party has access to their private information without their permission.

4.3.5. Accountability. The rental contract signed by the leaseholder and landlord is based on a smart contract. They must be responsible for the contract they have signed and cannot break or deny the contents of the contract.

4.3.6. Legality of Subletting. If the landlord agrees to the tenant's subletting of an unexpired property, the house owner uses his or her private key to grant the boarder the legal right to sublet.

5. Proposed Scheme: Rentchain

5.1. Details of Our Proposed Scheme. Homeowners generate their own self-sovereign identity ID and corresponding public-private key pairs (pk, sk) according to their needs for rental properties, and get a legal verifiable certificate

statement through a trusted third-party organization. The owner sends the verifiable statement and ID to the accounting node of the consortium chain for verification. The accounting node publishes the information of rental properties with verifiable statements on the consortium chain for tenants to inquire according to their rental needs. If users in the consortium chain initiate a change request, they need to modify the content in the history block to initiate the voting phase. When more than half of the signatures of users in the consortium chain agree to the modification, the signatures of users who agree to the modification are broadcasted.

The user who changes the block is selected according to the distributed random generation (DRG) protocol and the Lagrange interpolation formula. The selected user modifies the content of the history block according to his own trapdoor key, and then broadcasts it to other users after the modification is completed. Only after all other users pass the verification, the changed history blocks are recorded and marked. The user who changes the block is selected according to the distributed random generation (DRG) protocol and the Lagrange interpolation formula. Selected user modifies the content of the history block according to his own trapdoor key and then broadcasts it to other users after the modification is completed. Only after all other users pass the verification, the changed history blocks are recorded and marked.

5.2. Rental Housing Information Up-Link Stage

(1) The homeowner H_i uses the RSA public key algorithm to generate his own identity ID_{H_i} and the corresponding public key pk_{H_i} and private key sk_{H_i}. Landlords H_i can generate different self-sovereign identity IDs and corresponding public-private key pairs according to their needs for renting out their properties. Tenant T_i uses RSA public key algorithm to generate their own public-private key pair (pk_{T_i}, sk_{T_i})

(2) The accounting node of the consortium chain (a trusted third-party verifier C_i) generates the corresponding public key hk_{C_i} and trapdoor private key tk_{C_i} based on the new chameleon hash Algorithm 1

(1) The landlord sends the basic information of the title deed, the self-sovereign identity ID, and the public key $(ID_{H_i}, pk_{H_i}, \inf_{H_i})$ of the house to be rented to the certification authority for authentication through a secure channel after encryption. The certification authority verifies the information after decrypting it using its own private key. If the information is incorrect, the certification authority rejects the request. Otherwise, the certification authority generates the verifiable statement claim and $\sigma_{H_i} = Sig_{sk_{C_i}}\{H(ID_{H_i}, pk_{H_i}, \inf_{H_i}), \text{claim}, t_1\}$, where t_1 is the verification period of the verifiable statement. Then, the certification authority C_i sends the verifiable statement and signature $\{ID_{H_i}, pk_{H_i}, \inf_{H_i}, claim, \sigma_{H_i}\}$ to the homeowner H_i through a secure channel. The homeowner needs to pay a certain amount of RentCoin to the certification authority as the certification fee

(2) After receiving a signed and authenticated statement from a certification authority, the owner requests the leader in the consortium chain to add the rental information to the chain. The owner encrypts a verifiable statement $E\text{claim} = Enc(ID_{C_i}, \{\text{claim}, t_1\})$ using the public key of the leader ID_{C_i} and sends it. After receiving the request from the homeowner, the leader first verifies whether the timestamp t_1 is within the allowed range compared to the current time. If not, the leader rejects the request; or else, the leader decrypts it with its own private key to get the verifiable statement claim

(3) The leader verifies the signature. If the signature is invalid, he rejects the request; contrarily the leader broadcasts the transactions for that period to other accounting nodes. After other accounting nodes verify the signature, the leader packages the signed transactions into blocks to join the consortium blockchain

Blockchain is a distributed system, which does not build upon a central authority. In our scheme, the consensus of the chosen consensus chain is Practical Byzantine Fault Tolerance (PBFT) [27]. We presuppose that there have $3f + 1$ accounting nodes in the consortium chain. There is only one leader for a time period, and the leader is rotated by the accounting nodes.

At regular intervals, the leader verifies the validity of the signature of the certification statement submitted by the homeowner. Before adding the transactions to the consortium blockchain, the leader broadcasts the results of the validation of the transactions. Only after getting the signatures of other accounting nodes, the leader packages the transactions for that period of time and adds them to the consensus blockchain. In this scenario, the hash of the transactions we use is the new chameleon hash. The new chameleon hash is used in order for the accounting node holding the trapdoor key to change the transaction content of the block on demand, while leaving the hash of the block unchanged. As seen in Figure 2, the first level of the Merkle tree is the new chameleon hash h, which is included in the transaction.

5.3. Smart Contract-Based Housing Rental Phase

(1) Tenants inquire and search for property information in the consensus chain according to their rental needs. If the landlord and tenant reach an agreement on the rental, the rental transaction is to be written into the consensus blockchain. The transaction includes rent, rental time, deposit, penalty for breach of contract, smart contract trigger conditions, and so on

Input: Security parameter κ;
Output: Secret trapdoor key $TK = (x_1, x_2, \cdots, x_n)$ and public hash key $HK = (g, h_1, h_2, \cdots, h_n)$;
1. Select prime p, q, where $p = kq + 1$
2. Select the element g of order q in the multiplicative cyclic group Z_p^*
3. Select a random value $x_1 \in Z_q^*, x_2 \in Z_q^*, \cdots, x_n \in Z_q^*$ as the secret trapdoor key $TK = (x_1, x_2, \cdots, x_n)$
4. Compute $h_1 = g^{x_1}, h_2 = g^{x_2}, \cdots, h_n = g^{x_n}$, and set public hash key $HK = (g, h_1, h_2, \cdots, h_n)$;
5. return $TK = (x_1, x_2, \cdots, x_n)$ and $HK = (g, h_1, h_2, \cdots, h_n)$;

Algorithm 1: New chameleon hash algorithm.

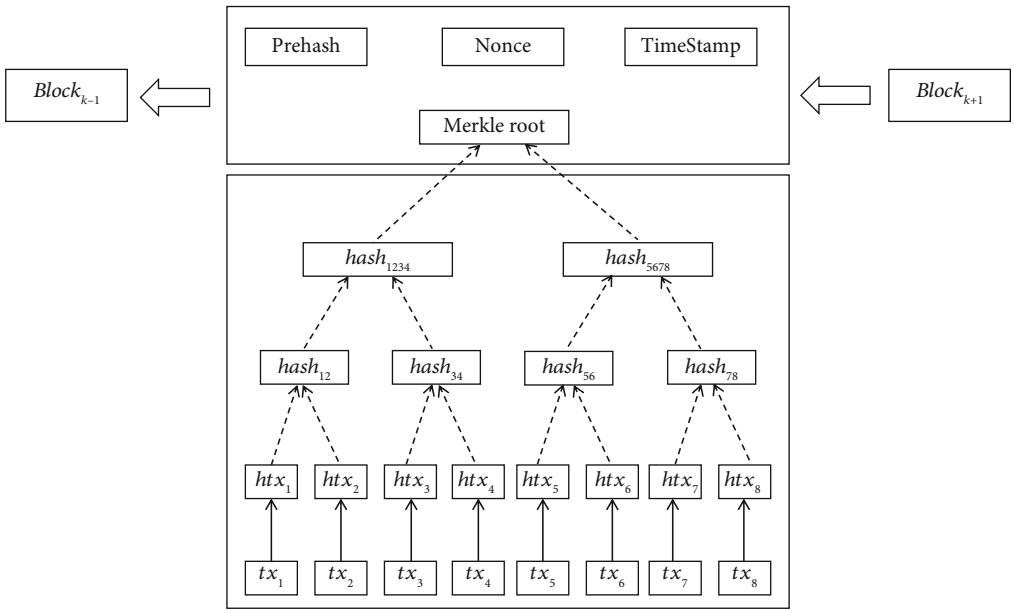


Figure 2: Block structure with new chameleon hash.

(2) The tenant submits their payment address to the smart contract server and the tenant deposits a certain amount of RentCoin to the payment address hinging on the rent

(3) The landlord and tenant create a specific lease contract for the housing lease and then send the finalized lease contract to the smart contract server. After the corresponding smart contract is generated, it will be sent to the owner as well as tenant for signature. Furthermore, both parties will use their own trapdoor private keys to sign after confirmation

(4) The signed smart contract is sent to the leader of the consortium blockchain, who verifies that the signature is correct and broadcasts the result to the other accounting nodes. After other nodes sign the contract, it is stored in the block. When triggering the smart contract, the automatic execution starts

(5) The process is shown in Figure 3 below

Before the smart contract starts to execute, the tenant transfers the deposit and the contracted rent in the form of RentCoin to the payment address, assuming that the tenant transfers RentCoin of b to the payment address. After the smart contract starts executing, the deposit and the contracted monthly (or several monthly) rent are transferred from the tenant's payment address to the landlord's payment address, supposing the monthly rent is b_0 and the deposit is b_1. After that the first smart contract implement, the remaining RentCoin in the tenant's payment account is $b - b_0 - b_1$. Each time the smart contract is hit, the lease is checked for expiration. If it has not expired, the rent continues to be transferred from the tenant's account to the landlord's payment account at b_0. If the tenant's lease has not expired and the landlord agrees to the tenant's subletting, the landlord uses his private key signature to authorize the tenant. Subsequently, the tenant can legally sublet after the landlord's signature.

When the lease between the landlord and tenant expires, the smart contract server generates a record to mark the termination of the smart contract.

At the same time, it is published to the consortium blockchain as a transaction and the contract is automatically terminated. Second, if the tenant's payment account is insufficient to pay the next month's rent after the smart contract is executed k times and the lease has not expired, i.e., $b - kb_0 - b_1 < b_0$ also performs the above operation. Therefore, the tenant must ensure that their payment account has

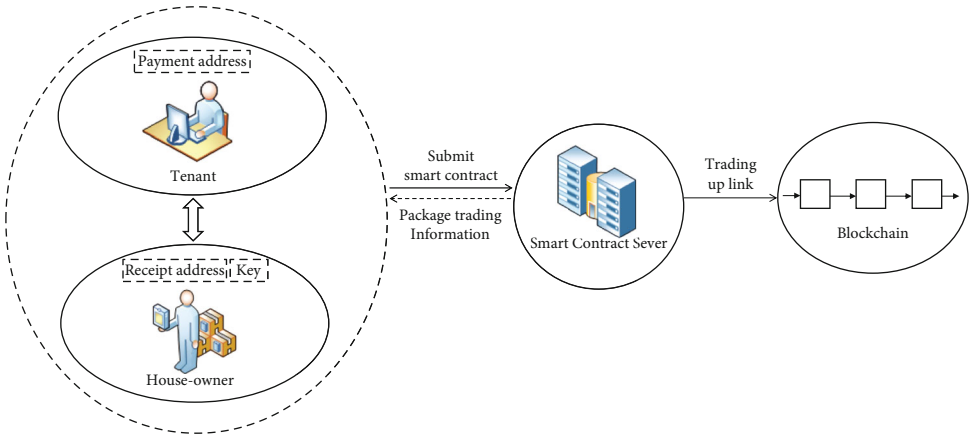


Figure 3: Lease contract generation process.

enough to pay the next month's RentCoin; otherwise, it is treated as a default. The defaulting tenant must pay the landlord a certain amount of default money, which can be deducted from the deposit. Finally, the landlord will return the deposit to the tenant's payment account after the lease expires normally. After the whole rental transaction is completed, the landlord and the tenant have to pay a certain amount of RentCoin to the consortium chain node as transaction fee. If the house has been rented, the status data of the property information update is written to the blockchain.

There is often a shortage of supply during the tenant rental phase, such as a significant increase in the number of rentals during the graduation season. At this time, the number of rented properties will be less than the number of tenants, and there is a noncooperative game among tenants during that period. We observe the healthy development of the housing rental market by solving the Nash equilibrium, through which we can calculate the Nash equilibrium of the housing rental market at different stages. The specific construction and solution process are shown below.

Let $N = 1, 2, \cdots, n$ be the set of tenants, $\forall i \in t$, and let T_i be the set of strategies (the set of listings) of tenant i, which is a nonempty set in R^{k_i}, $T = \prod_{i=1}^{n} T_i$, when tenant i chooses the listings$t_i \in T_i$, $i = 1, 2, \cdots, n$, the benefit that tenant i gets is $f_i(t_1, t_2, \cdots, t_n)$. $\forall i \in N$, let $\hat{i} = N \setminus \{i\}$, $T_{\hat{i}} = \prod_{j \neq i} T_j$, $f_i(t_1, t_2, \cdots, t_n) = f_i(t_i, t_{\hat{i}})$, where $t_{\hat{i}} \in T_{\hat{i}}$. If there exists $t^* = (t_1^*, t_2^*, \cdots, t_n^*) \in T$ such that $\forall i \in N$, with $f_i(t_i^*, t_{\hat{i}}) = \max_{u_i \in T_i} f_i(u_i, t_{\hat{i}}^*)$, then t^* at this point is the Nash equilibrium point of this n-player noncooperative game. At the equilibrium point, each tenant cannot make himself or herself more profitable by individually changing his or her chosen listing.

$\forall i \in N$, let T_i be a nonempty bounded closed convex set in R^{k_i}, $T = \prod_{i=1}^{n} T_i, f_i : T \longrightarrow R$ continuous, and $\forall t_{\hat{i}} \in T_{\hat{i}}, u_i \longrightarrow f_i(u_i, t_{\hat{i}})$ on T_i to be concave, then the Nash equilibrium point of the noncooperative game must exist. This is because, first, $T = \prod_{i=1}^{n} T_i$ must be a nonempty bounded closed convex set in R^k, where $k = k_1 + k_2 + \cdots k_n$. $\forall i \in N, \forall t_{\hat{i}} \in T_{\hat{i}}, F_i(t_{\hat{i}}) = \omega_i \in T_i : f_i(\omega_i, t_{\hat{i}}) = \max_{u_i \in T_i} f_i(u_i, t_{\hat{i}})$.

First, since f_i is continuous when $t_{\hat{i}}$ is fixed, $u_i \longrightarrow f_i(u_i, t_{\hat{i}})$ is continuous, and T_i is a bounded closed set in R^{k_i}, so $F_i(t_{\hat{i}}) \neq \varnothing$. T_i is a boundary, then $F_i(t_{\hat{i}})$ there must be a boundary. Next, note that $\max_{u_i \in T_i} f_i(u_i, t_{\hat{i}}) = c$, $\forall \omega_i^m \in F_i(t_{\hat{i}})$, $\omega_i^m \longrightarrow \omega_i$, then $\omega_i^m \in T_i$. Since T_i is a closed set, $\omega_i \in T_i$. $f_i(\omega_i^m, t_{\hat{i}}) = c$, and since f_i is continuous, $f_i(\omega_i, t_{\hat{i}}) = c, \omega_i \in F_i(t_{\hat{i}})$, $F_i(t_{\hat{i}})$ must be a closed set. $\forall \omega_i^1, \omega_i^2 \in F_i(t_{\hat{i}}), \forall \xi \in (0, 1)$, as $\omega_i^1, \omega_i^2 \in T_i$, T_i is a convex set, $\xi\omega_i^1 + (1 - \xi)\omega_i^2 \in T_i, f_i(\xi\omega_i^1 + (1 - \xi)\omega_i^2) \leq c$. $f_i(\omega_i^1, t_{\hat{i}}) = f_i(\omega_i^2, t_{\hat{i}}) = c$, when $t_{\hat{i}}$ is fixed, $u_i \longrightarrow f_i(u_i, t_{\hat{i}})$ is concave on T_i. So $f_i(\xi\omega_i^1 + (1 - \xi)\omega_i^2, t_{\hat{i}}) \geq \min f_i(\omega_i^1, t_{\hat{i}}) = f_i(\omega_i^2, t_{\hat{i}}) = c$ holds. Therefore, $f_i(\omega_i^1, t_{\hat{i}}) f_i(\xi\omega_i^1 + (1 - \xi)\omega_i^2, t_{\hat{i}}) = c$ $, f_i(\xi\omega_i^1 + (1 - \xi)\omega_i^2 \leq c\xi\omega_i^1 + (1 - \xi)\omega_i^2 \in F_i(t_{\hat{i}}))$, $F_i(t_{\hat{i}})$ must be a convex set. $\forall i \in N$, $F_i(t_{\hat{i}})$ is a nonempty bounded closed convex set in R^{k_i}. Because $F(t) = \prod_{i=1}^{n} F_i(t_{\hat{i}})$, it must be a nonempty bounded closed convex set in R^k, where $k = k_1 + k_2 + \cdots k_n$.

Finally, $\forall i \in N$, since $f_i(u_i, t_{\hat{i}})$ is continuous and T_i is a bounded closed set, $\forall t_{\hat{i}} \in T_{\hat{i}}$ the set-valued map from $G_i(t_{\hat{i}}) = T_i$ must be continuous. T_i is a bounded closed set, and by the great value theorem [28], the extremal map $F_i : T_{\hat{i}} \longrightarrow P_0(T_i)$ must be upper semicontinuous. $\forall t \in T$, since $F(t) = \prod_{i=1}^{n} F_i(t_{\hat{i}})$, best response mapping $F : T \longrightarrow P_0(T)$ on must be upper semicontinuous. Thus, by Kakutani fixed point theorem [29], there exists $t^* \in T$ such that, $t^* \in F(t^*)$, then t^* must be a Nash equilibrium point of the noncooperative game. In other words, if we can find the Nash equilibrium point during the peak rental period, we can prove to a certain extent that the housing rental market is healthy; if we cannot find the Nash equilibrium point, we cannot simply infer that the housing rental market is unhealthy, and we need to consider a number of factors.

5.4. Initiation of Modification Request Phase

(1) If the owner needs to modify the content of the transaction due to policy or rent change, he/she needs to submit a request to the accounting node of the consortium blockchain to change the content

of the historical block m to m'. The owner signs the request with his/her private key sk_{H_i} to get σ_s, and the leader of the consortium chain broadcasts (request, σ_s) to the accounting nodes of the consortium chain that starts the voting phase

(2) After receiving the request, the accounting nodes in the consortium chain sign and broadcast the initiated request if they agree to the modification

(3) After the leader collects signatures from greater than half of the consortium chain's accounting nodes (assuming that the number of signed users is $y > n/2$), he broadcasts these y signatures

5.5. Select the Change Block User Stage

(1) The y consortium chain accounting nodes participating in the voting are noted as $(P_1, P_2, \cdots, P_y)$. Each node P_i chooses a random number ρ_i, and according to (y, n) verifiable secret sharing shares ρ_i to other nodes. The sharing value of ρ_i is recorded as $(s_{i,1}, s_{i,2}, \cdots, s_{i,n})$

(2) Each consortium chain's accounting node P_i verifies the shared values after receiving. The shared values $s_{i,1}, s_{i,2}, \cdots, s_{i,n}$ after passing the verification are summed to $S_i = s_{i,1} + s_{i,2} + \cdots + s_{i,n}$, broadcasting S_i

(3) After each consortium chain accounting node receives at least y of S_i, the value of the random number ρ is calculated by Lagrange interpolation, $\rho = \rho_1 + \rho_2 + \cdots + \rho_y$

(4) Each consortium chain accounting node computes the hash value $h_i = \mathrm{Hash}(\rho, ID_{C_i}), i \in \{1, 2, \cdots, y\}$. The computed y hash values are sorted and the public key ID_{C_s} corresponding to the smallest hash value h_s is selected as the user P_s who modifies the block

5.6. Change the Block Content and Confirm Phase

(1) After selecting the consortium chain accounting node P_s that modifies the block, P_s uses its own private key sk_{C_s} to modify the message m of the historical block to m', and the calculated $r_s' = (m - m' + x_s r_s) x_s^{-1} mod q$. The content of the modified block header becomes $(m', (r_1, \cdots, r_s{}', \cdots, r_n))$, i.e. the rest remains the same except that m and r_s become m' and $r_s{}'$

(2) The node P_s will broadcast $(m', (r_1, \cdots, r_s{}', \cdots, r_n))$, the votes of other users on the modification requests, the obtained random numbers ρ and the signatures on these contents

(3) After the other nodes of the consortium chain receive the broadcast, they verify whether the node P_s that modifies the consortium chain block corresponds to the hash value $\mathrm{Hash}(\rho, ID_{C_s})$ is the smallest, and use the public key ID_{C_s} of the user to verify his signature and the signatures of other users at the voting stage. If all the above verifications pass, then verify the $\mathrm{Hash}(m', (r_1, \cdots, r_s{}', \cdots, r_n))$ and $\mathrm{Hash}(m, (r_1, \cdots, r_s, \cdots, r_n))$ are equal or not. If all of them are verified, the historical blocks after the changes are recorded and marked with all the information broadcast by P_s. The content of the tag contains all the information broadcast by P_s who modifies the consortium chain block

(4) When using trapdoor keys to modify the contents of historical blocks, signed consent from other nodes needs to be obtained. This approach has somewhat curbed the reckless rent inflation by landlords

6. Conclusion

Compared with the blockchain-based scheme proposed by Lin et al. to build a housing rental ecosystem [30], we introduce chameleon hashing to give each node of the consortium chain the right to modify historical blocks and use n noncooperative game to detect the health of the housing rental market in terms of undersupply to a certain extent. Both schemes apply blockchain technology to the housing rental market and propose solutions to various problems in the housing rental market from different perspectives, respectively.

Our proposal focuses on solving the problems of information asymmetry, illegal subletting, and tenant information leakage in the housing leasing market. The leasing contract based on smart contract enhances the fairness and convenience of transactions in the housing rental market to a certain extent. In the future, we will improve the evaluation and reward mechanism of the scheme as well as enrich the information of rental properties and other issues.

Data Availability

We did not use any external data.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (Grant no. 12061020), Foundation of National Natural Science Foundation of China (Grant no. 61962009), Major Scientific and Technological Special Project of Guizhou Province (20183001), Science and Technology Support Plan of Guizhou Province ([2020] 2Y011), and Foundation of Guangxi Key Laboratory of Cryptography and Information Security (GCIS202118).

References

[1] C. Xiaoqing, D. Yao, Z. Liang et al., "The principle and core technology of blockchain," *Chinese Journal of Computers*, vol. 44, no. 1, pp. 84–126, 2021.

[2] T. Li, Z. Wang, G. Yang, Y. Cui, Y. Chen, and X. Yu, "Semi-selfish mining based on hidden Markov decision process," *International Journal of Intelligent Systems*, vol. 36, no. 7, pp. 3596–3612, 2021.

[3] T. Li, Z. Wang, Y. Chen, C. Li, Y. Jia, and Y. Yang, "Is semi-selfish mining available without being detected?," *International Journal of Intelligent Systems*, 2021.

[4] Y. Chen, S. Dong, T. Li, Y. Wang, and H. Zhou, "Dynamic multi-key FHE in asymmetric key setting from LWE," *IEEE Transactions on Information Forensics and Security.*, vol. 16, pp. 5239–5249, 2021.

[5] H. M. Krawczyk and T. D. Rabin, "Chameleon hashing and signatures," 2000, US, US6108783 A.

[6] X. Chen, F. Zhang, W. Susilo, H. Tian, J. Li, and K. Kim, "Identity-based chameleon hashing and signatures without key exposure," *Information Sciences an International Journal*, vol. 265, pp. 198–210, 2014.

[7] S. Nakamotos, "Bitcoin: a peer-to-peer electronic cash system," 2009, https://bitcoin.org/bitcoin.pdf.

[8] J. Kang, R. Yu, X. Huang et al., "Blockchain for secure and efficient data sharing in vehicular edge computing and networks," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4660–4670, 2019.

[9] R. Subramaniam, S. R. Azzuhri, and T. Y. Wah, "Enhanced security approach powered by blockchain technology with NFC to prevent fraudulence in bank letter of credits," in *IECC 2020: 2020 2nd International Electronics Communication Conference*, New York, NY, USA, 2020.

[10] J. Xu, K. Xue, H. Tian, J. Hong, D. S. L. Wei, and P. Hong, "An identity management and authentication scheme based on redactable blockchain for mobile networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 6, pp. 6688–6698, 2020.

[11] Y. Chen, J. Sun, Y. Yang, T. Li, X. Niu, and H. Zhou, "PSSPR: a source location privacy protection scheme based on sector phantom routing in WSNs," *International Journal of Intelligent Systems*, vol. 37, no. 2, pp. 1204–1221, 2022.

[12] K. Ashritha, M. Sindhu, and K. V. Lakshmy, "Redactable blockchain using enhanced chameleon hash function," in *2019 5th International Conference on Advanced Computing and Communication Systems (ICACCS)*, Coimbatore, India, 2019.

[13] J. Xu, K. Xue, S. Li et al., "Healthchain: a blockchain-based privacy preserving scheme for large-scale health data," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8770–8781, 2019.

[14] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612-613, 1979.

[15] G. R. Blakley, "Safeguarding cryptographic keys," in *American Federation of Information Processing Societies*, USA, 1979.

[16] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch, "Verifiable secret sharing and achieving simultaneity in the presence of faults," in *26th Annual Symposium on Foundations of Computer Science*, Portland, Oregon, USA, 1985.

[17] P. Feldman, "A practical scheme for non-interactive verifiable secret sharing," in *Proceedings of 28th IEEE Symposium on Foundations of Computer Science*, Los Angeles, CA, USA, 1987.

[18] T. P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Advances in Cryptology — CRYPTO '91. CRYPTO 1991*, J. Feigenbaum, Ed., vol. 576 of Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 1991.

[19] M. Benor, S. Goldwasser, and A. Windgerson, "Completeness theorems for noncryptographic fault—tolerant distributed computation," in *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, Chicago, IL, USA, 1988.

[20] K. C. Toth and A. Anderson-Priddy, "Self-sovereign digital identity: a paradigm shift for identity," *IEEE Security and Privacy Magazine*, vol. 17, no. 3, pp. 17–27, 2019.

[21] G. Ateniese, B. Magri, D. Venturi, and E. Andrade, "Redactable blockchain-or-rewriting history in bitcoin and friends," in *2017 IEEE European symposium on security and privacy*, Paris, France, 2017.

[22] P. Li, H. Xu, T. Ma, and Y. Mu, "Reasearch on fault-correcting blockchain technology," *Journal of Cryptologic Research*, vol. 7, pp. 401–405, 2018.

[23] J. Nash, "Equilibrium points in n-person games," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 36, no. 1, pp. 48-49, 1950.

[24] J. Nash, "Non-cooperative games," *Annals of Mathematics*, vol. 54, no. 2, pp. 286–295, 1951.

[25] N. Szabo, "Smart contracts," 1994.

[26] R. Wang, W. T. Tsai, J. He, C. Liu, Q. Li, and E. Deng, "A medical data sharing platform based on permissioned blockchains," in *The 2018 International Conference*, New York, NY, USA, 2018.

[27] M. Castro and B. Liskov, "Practical byzantine fault tolerance," in *Operating Systems. Design Implementation*, USA, 1999.

[28] C. Berge, *Topological Spaces*, Mac Millan, New York, 1963.

[29] S. Kautani, "A generalization of Brouwer's fixed point theorem," *Duke Mathematical Journal*, vol. 8, no. 3, pp. 457–459, 1941.

[30] Y. Lin, M. Shi, and L. Chen, "Research on the application of blockchain technology in building housing rental information ecosystem," *Price: Theory & Practice*, vol. 10, pp. 56–59, 2020.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 1040662, 11 pages
https://doi.org/10.1155/2022/1040662

Research Article

Blockchain Data Privacy Protection and Sharing Scheme Based on Zero-Knowledge Proof

Tao Feng,[1] Pu Yang,[1] Chunyan Liu,[2] Junli Fang,[1] and Rong Ma[1]

[1]*School of Computer and Communication, Lanzhou of University of Technology, Lanzhou 730050, China*
[2]*School of Economics and Management, Lanzhou of University of Technology, Lanzhou 730050, China*

Correspondence should be addressed to Pu Yang; ypu97717@163.com

Received 3 November 2021; Accepted 21 January 2022; Published 23 February 2022

Academic Editor: Jinguang Han

The data generated in the Industrial Internet of Things (IIoT) has important research value. In the process of data sharing, data privacy, security, and data availability are important issues that cannot be ignored. This paper proposes a blockchain privacy protection scheme based on zero-knowledge proof to realize the secure sharing of data among data owners, cloud service providers, and semitrusted cloud servers. First, the method of combining zero-knowledge proof and smart contract is used to verify the availability of data between the data owner and the cloud service provider under the premise of protecting data privacy. Second, proxy reencryption technology is used to realize the secure sharing of data among authorized cloud service providers. In addition, data sharing transaction information between multiple parties and data hashes with digital signatures are stored on the blockchain to achieve public and verifiable data sharing information and data validity. Finally, the theoretical analysis of the scheme shows that the scheme meets the confidentiality requirements of security, integrity, and validity.

1. Introduction

Since the 21st century, the Internet has given traditional industries the explosive growth of data in the Industrial Internet of Things [1, 2]. The massive amount of data generated in different fields (such as smart home, smart city, and smart manufacturing) has extremely high research value, which has aroused research interest in industry and academia. How to share data safely and efficiently, use data to provide users with better and convenient services, and improve user experience has become a widespread concern today. However, most of the data generated by IIoT is the user's private data. In the process of data sharing, it is necessary to ensure the privacy, integrity, and validity of the data [3–5]. For example, sensitive and private data is tampered with or leaked during the sharing process. Data owners may provide irrelevant or false data to cloud service providers. Cloud service providers do not want data owners to provide information to other research institutions. Therefore, the following problems still exist in the data sharing of the Industrial Internet of Things. (1) There is lack of protection of data privacy and security in the data sharing process. (2) The data recipient cannot ensure that the data obtained is valid and relevant information. (3) Data integrity and data transaction records cannot be verified and traced during the data sharing process. Therefore, due to the above-mentioned problems, there is an urgent need for a solution to realize data sharing while protecting privacy and security.

Zero-knowledge proof is a cryptographic technology, which can make the verifier believe that a certain assertion is correct without providing any valuable information to the verifier. Zero-knowledge succinct noninteractive knowledge argumentation (zk-SNARKs) is one of the tools for generating zero-knowledge proofs. In the blockchain transaction platform, it is used in cryptocurrencies such as Zcash [6] and ZETH [7] to hide private information such as the address of the sender and receiver of the transaction and the transaction amount. In the data sharing between the cloud service provider and the data owner, zero-knowledge proof combined with smart contract technology can realize data availability verification between the two parties' data

transactions and ensure the provision of effective data information.

Blockchain is an effective method to solve verifiable and traceable transactions due to its decentralization, immutability, traceability, and executable smart contracts. Due to the characteristics of its distributed data ledger, it is widely used in multiple scenarios such as virtual currency, electronic bidding, and Industrial Internet of Things. In terms of addressing data privacy, blockchain can be combined with a variety of cryptographic methods, for example, attribute encryption [8], homomorphic encryption [9], searchable encryption, and proxy reencryption combined [10], to achieve the protection of data privacy and identity privacy on the blockchain.

In the data sharing scheme based on blockchain, some researchers have implemented data sharing schemes for individual users. However, these solutions focus on the aggregation of data and the balance between data privacy and data accessibility in the process of data sharing transactions, and data transmission between multiple entities cannot ensure user data privacy in the entire process. In response to these existing problems, this paper proposes a blockchain data privacy protection and sharing scheme based on zero-knowledge proof. It solves the problems of data privacy security, data availability and consistency, and data transaction traceability in data sharing.

The main research contributions of this paper are as follows.

(1) In multientity data sharing, a zero-knowledge proof-based blockchain data privacy protection and sharing scheme is proposed to achieve privacy protection. Use proxy reencryption technology to ensure data sharing between cloud service providers and data owners. Realize data sharing, traceability, and verifiability among multiple entities based on blockchain characteristics

(2) A method of combining zero-knowledge proof and smart contract is proposed. The data owner can prove that the data meets the requirements of the cloud service organization without revealing any data privacy, realize the consistency and availability of the data in the sharing process, and protect the interests of both parties. After the verification is passed, the improved consensus algorithm enables the nodes to reach consensus directly and faster

(3) Through security analysis and comparison with other solutions, this solution realizes the sharing of data among multiple entities under the premise of not revealing any data privacy, and the consistency, availability, and traceability, and verifiable characteristics of the sharing process during the sharing process. And it has better consensus efficiency

2. Related Work

In a data sharing scheme based on cloud services, it relies on some encryption methods to protect data privacy. However, the data is difficult to trace and verify, and the data is easy to be stolen and tampered. Blockchain can be used to solve some of the current problems in data sharing due to its decentralization, immutability, traceability, and other characteristics. In the data sharing scheme based on blockchain [11, 12], data privacy protection combined with data encryption mainly uses encryption methods such as attribute encryption and proxy reencryption.

In the research of data sharing based on cloud services, Muthusenthil et al. [13] proposed a new secure data sharing reencryption scheme based on trusted institutions, using proxy reencryption methods to ensure data privacy and security, with better performance. However, the solution cannot guarantee user identity privacy and does not have the traceability of transactions and data. Mahakalkar and Sahare [14] proposed SAPA, a privacy protection authentication protocol based on sharing authority, which uses reencryption to realize data sharing between multiple users. The use of an access request matching mechanism realizes the user's identity is private, but cannot guarantee the traceability of data and transactions. Wang et al. [15] proposed an identity-based data sharing audit scheme, which uses an information-hiding mechanism and a security mechanism that simplifies the signature algorithm to protect sensitive information and prevent malicious managers. However, the validity and consistency of the data cannot be guaranteed. Cheng et al. [16] proposed a reliable and efficient data sharing solution for the Industrial Internet of Things (IIoT). The scheme is based on an adaptive decentralized inadvertent transmission protocol, combined with zero-knowledge proof technology, so that the private key of the data recipient can be hidden from the data owner during the data sharing process. The traceability of data is realized, but the traceability of transactions cannot be realized.

In the research of blockchain-based data sharing solutions, Chowdhury et al. [17] proposed a notarization service framework based on blockchain-based personal data storage and sharing. This framework will ensure the authenticity of real-time shared data, and the transaction privacy is provided in the chain network. However, the complete traceability of the data is guaranteed in the process, but the privacy of the data cannot be guaranteed. Lu et al. [18] designed and implemented a blockchain-authorized secure data sharing architecture, combined with federal learning combined with privacy protection, transformed data sharing problems into machine learning problems, and maintained data privacy. However, the traceability of the transaction and the integrity of the data cannot be guaranteed. Wang et al. [19] proposed a blockchain-based security and privacy protection electronic medical record sharing protocol, which combines searchable encryption and conditional proxy reencryption to achieve data security, privacy protection, and access control. However, the validity of the data cannot be guaranteed. Sani et al. [20] proposed a high-performance, scalable blockchain that enhances the security and privacy of IIoT, using time-based zero-knowledge proof and authentication encryption to perform mutual authentication between multiple attributes. The evaluation from the three aspects of security, privacy, and performance shows that the scheme is safe, and the computational complexity and delay performance are significantly reduced. The privacy of

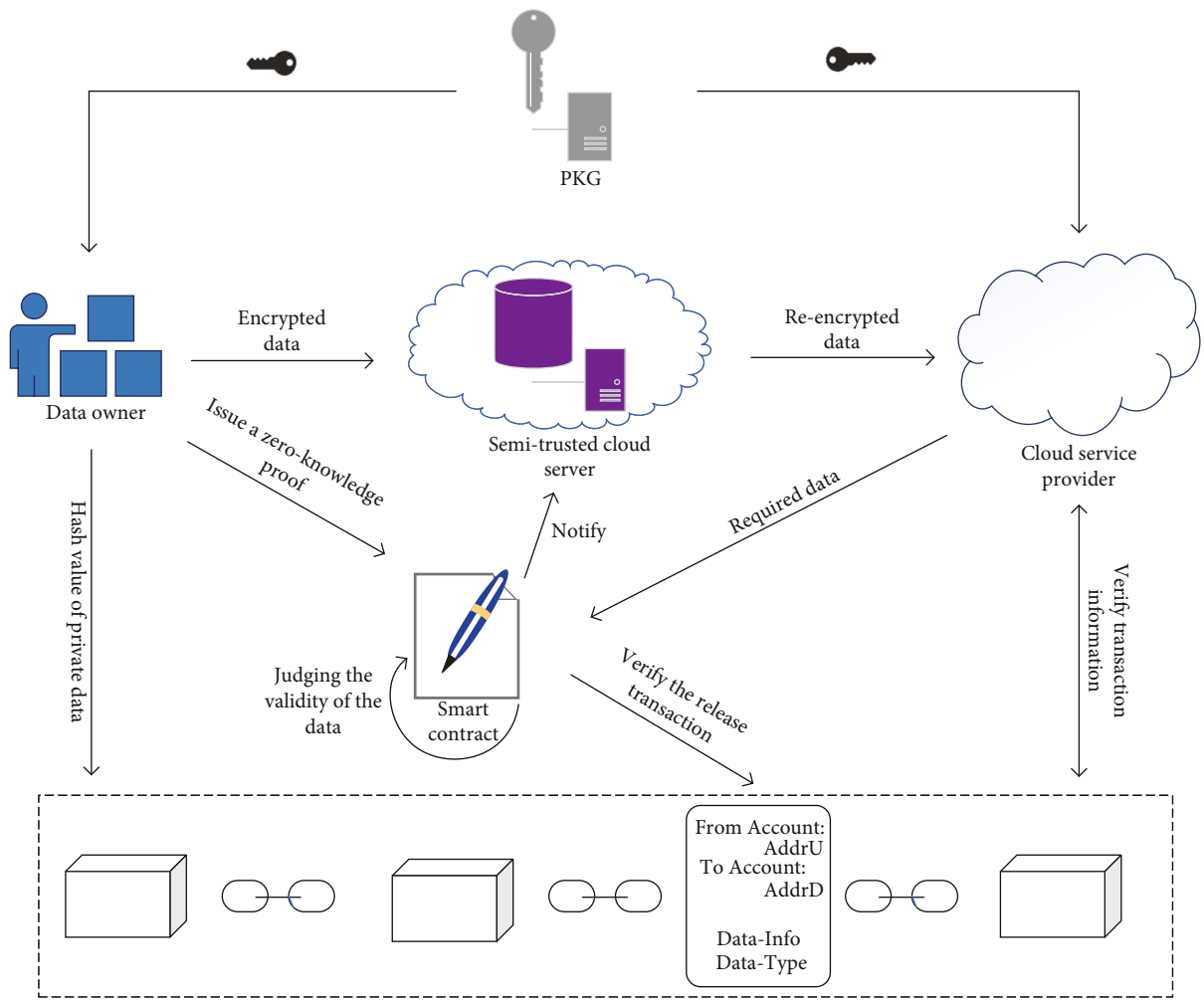


Figure 1: Blockchain data privacy protection and sharing scheme model based on zero-knowledge proof.

identity and data is guaranteed, but the traceability of transactions cannot be achieved. Shen et al. [21] proposed a reliable sharing and collaboration model based on blockchain. Data owners, miners, and third parties share data through blockchain and record through smart contracts. Participants can use private clouds or public clouds to obtain and store data sharing. The identity privacy of data participants is guaranteed, but the content privacy of data cannot be guaranteed. Kouicem et al. [22] proposed a decentralized and anonymous vehicle data sharing scheme, allowing each vehicle to anonymously verify each data record without revealing the identity of the vehicle sharing this data. Each vehicle sends a certificate to the data record, which uses zero-knowledge proof (ZKP) to anonymously combine the data record and the user's identity. Identity privacy is guaranteed, but the traceability of data transactions cannot be achieved. Manzoor et al. [23] proposed a blockchain-based IoT data sharing scheme. Use proxy reencryption to store and share Internet of Things data in a cloud proxy server, and establish smart contracts between sensors and data consumers without the involvement of a trusted third party. The privacy of the data is guaranteed, but the validity and consistency of the data cannot be guaranteed.

From the above scheme, we can see that the blockchain-based cloud data sharing scheme has achieved certain research results, and a variety of data sharing schemes have been proposed using blockchain technology and cryptographic methods. However, the consistency and availability of data in data sharing and the traceability and verifiability of data sharing transactions between multiple entities have not been effectively improved.

3. Problem Description

The solution proposed in this article combines blockchain, agent heavy intelligence, smart contracts, and zk-SNARK technology to achieve privacy protection and data security sharing among data owners, cloud service organizations, and semitrusted cloud servers. The system model of our proposal is shown in Figure 1.

It includes 6 participating entities: (1) data owner, (2) cloud service organization (CSP), (3) semitrusted cloud server (semitrusted CS), (4) private key generator (PKG), (5) smart contract, and (6) blockchain (Blockchain). Their functions are described as follows.

(i) Data owner: data owners have the right to securely own and conditionally share their information and data and can obtain corresponding benefits as remuneration during the sharing process

(ii) Cloud service organization (CSP): as a consumer of private data, cloud service organization needs to collect and analyze private data. They issue corresponding privacy data requirements by entrusting smart contracts. But at the same time, they do not believe that the data provided by the data owner meets their needs, so they use smart contracts to ensure the consistency and effectiveness of the data requirements

(iii) Semitrusted cloud server: as a semitrusted entity, it needs to store the original ciphertext of the data owner and is responsible for converting it into intermediate ciphertext, which will be handed over to the cloud service organization after verification and decrypted by its private key

(iv) Private key generator (PKG): it is a completely trusted entity that needs to generate master keys and system parameters and distributes public keys and keys to data owners and cloud service organizations

(v) Smart contract: smart contracts are responsible for predeclaring the requirements and the specific structure of private data and guaranteeing certain data benefits. Automatically judge the validity of the zero-knowledge proof without the participation of a third party

(vi) Blockchain: responsible for reaching a consensus on data transactions. Store the hash of private data in the blockchain to ensure the immutability and traceability of the data, which is the evidence for data disputes

4. Security Model

4.1. Definition. Based on the DBDH assumption, under the random oracle model, if there is a negligible function $\varepsilon(\kappa)$ for any polynomial time adversary A, such that $\mathrm{Adv}_{\prod A}^{\mathrm{CPA}}(\kappa) \leq \varepsilon(\kappa)$, then the encryption algorithm $\prod$ is indistinguishable under the selected plaintext attack, which is called IND-CPA (indistinguishability-chosen plaintext attack) security.

4.2. Initialization. Challenger B runs the initialization algorithm to generate public parameters and master key (PK, MSK) and sends the public parameters PK to the adversary A.

Stage 1: the adversary A sends a key generation request to the challenger, and the challenger B generates a key pair $(\mathrm{PK}_u, \mathrm{SK}_u)$ and sends it to A.

Challenge phase: the adversary A sends two messages of the same length, m_0 and m_1, to the challenger B. The challenger B randomly selects $\sigma \in (0, 1)$ and sends $\mathrm{Enc} = E_n(\mathrm{PK}, \mathrm{PK}_u, m_\sigma)$ to the adversary.

Stage 2: the adversary A repeats the request phase 1.

The adversary A outputs a guess value of $\sigma' \in (0, 1)$; if $\sigma = \sigma'$, the adversary A wins the game. The advantage of the adversary A can be defined as $\mathrm{Adv}_{\prod A}^{\mathrm{CPA}}(\kappa) = |\Pr[\sigma' = \sigma] - (1/2)|$.

5. Blockchain Data Privacy Protection Scheme Based on ZKP

5.1. Scheme Steps. As shown in Figure 2, after each entity is registered in the blockchain, the private key generation center assigns a common private key pair to the user. The cloud service provider generates a zero-knowledge proof π' of the required data through zk-SNARK, sends the calculation results R' and hash value h' to the smart contract, and records and publishes the required keywords in the blockchain. The data owner generates an encrypted ciphertext according to the needs of the cloud service provider, sends it to the semitrusted cloud server, and records the signed hash in the blockchain. At the same time, the zero-knowledge proof π generated by the private data, the calculation result R, and the hash value h are sent to the smart contract for automatic comparison. After passing the zero-knowledge proof verification, the data owner is notified to use the public key PK_d of the cloud service organization to execute the reencryption algorithm to generate the reencryption key $\mathrm{PK}_{u \longrightarrow d}$ and send it to the semitrusted proxy cloud server through the public key of the cloud service provider. The semitrusted proxy cloud server executes the reencryption algorithm to convert the ciphertext C_{PK_u} into the intermediate ciphertext $C_{\mathrm{PK}_{u \longrightarrow d}}$ and then sends the intermediate ciphertext to the cloud service provider. The cloud service provider uses the private key SK_d to execute the decryption algorithm to obtain the required private data for verification based on the information on the blockchain. After the smart contract is passed, the data sharing information transaction is submitted to the verification node, and the RBFT consensus algorithm is used to verify it and then publish it on the blockchain. The symbols used in this article are shown in Table 1.

6. Specific Structure

The specific construction process of the scheme is divided into the following ten stages.

6.1. Join the Network Phase. Equations should be provided in a text format, rather than as an image. Microsoft Word's equation tool is acceptable. Equations should be numbered consecutively, in round brackets, on the right-hand side of the page. They should be referred to as Equation 1, etc. in the main text.

6.2. Data Initialization Phase. First, PKG chooses to input a security parameter λ, selects prime number to generate multiplication cycles G_1 and G_1, and selects four hash function groups $H_1 : \{0, 1\}^* \longrightarrow \{0, 1\}^k$, $H_2 : \{0, 1\}^* \longrightarrow G_1$, $H_3 : G_2 \longrightarrow \{0, 1\}^k$, and $H_4 : \{0, 1\}^k \times \{0, 1\}^l \longrightarrow \mathbb{Z}_p^*$.

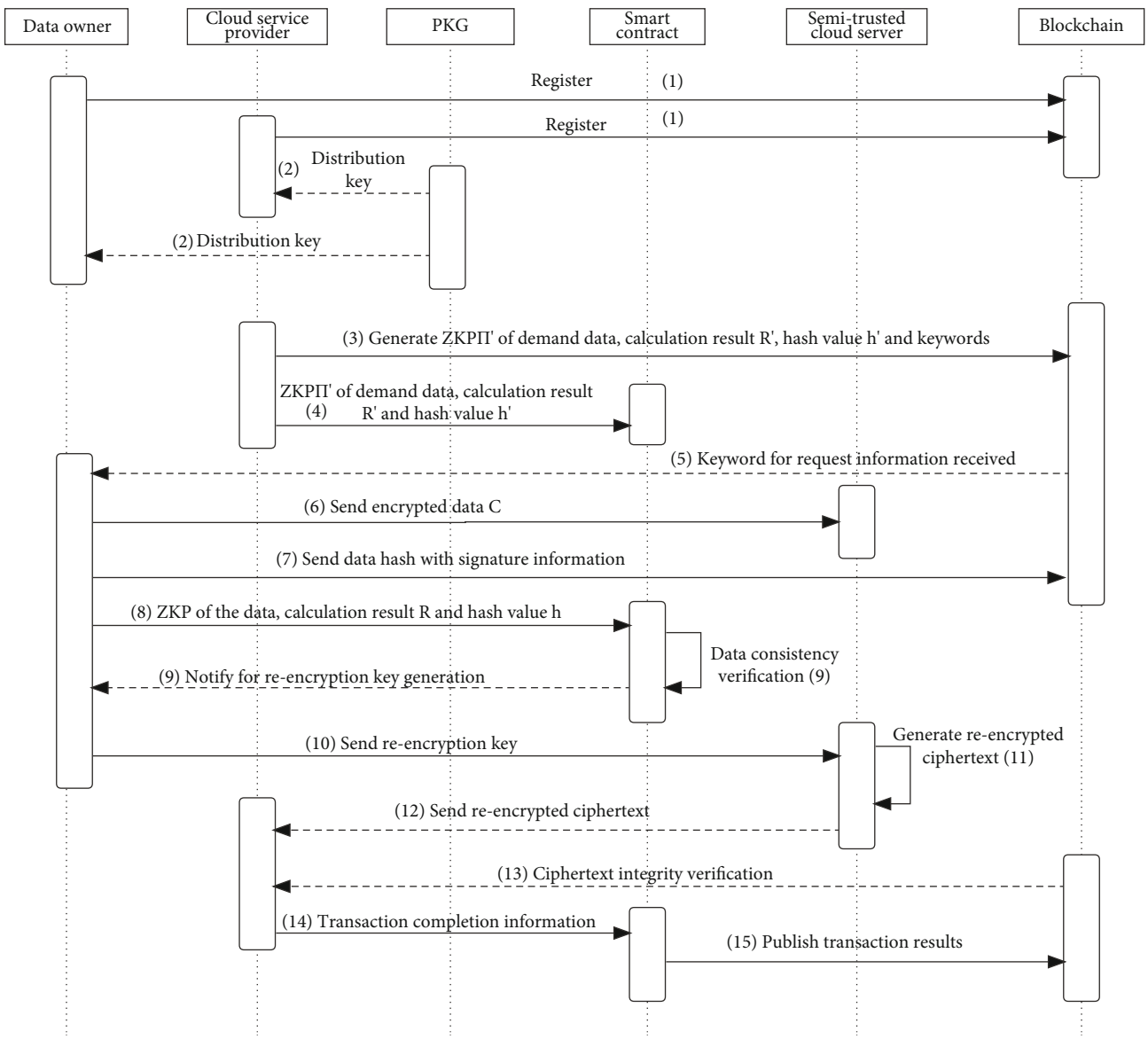


Figure 2: Timing diagram of blockchain data privacy protection and sharing scheme based on ZKP.

Table 1: Notations.

Symbols	Definitions
λ	Security parameter
PKG	Private key generation center
D	Data owner's private data
PK_u, PK_d	The public key of the data owner and cloud service provider
SK_u, SK_d	The private key of the data owner and cloud service provider
$RK_{u\longrightarrow d}$	Reencryption key
$C_{PK_{u\longrightarrow d}}$	Reencrypted ciphertext
σ_a	Digital signature
π	Zero-knowledge proof
EK_C	Generate the key for zero-knowledge proof
VK_C	Key to verify zero-knowledge proof

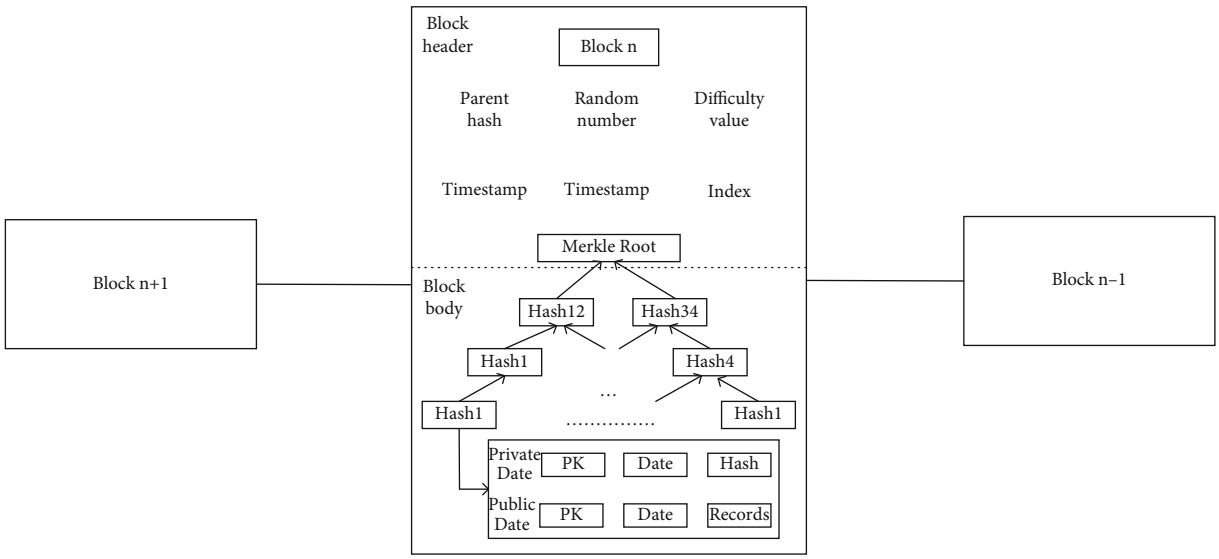


Figure 3: Transaction form.

Define the bilinear mapping $e : G_1 \times G_2 \longrightarrow G_2$. Then, PKG randomly selects $a, b, c \in \mathbb{Z}_p^*$, and $g, h \in G_1$ are two different generators of G_1. Generate public parameters and master key $\text{Setup}(1^\lambda) \longrightarrow (\text{PK}, \text{MSK})$, where $\text{PK} = (p, G_1, G_2, e, g, h, H_1, H_2, H_3 H_4)$, $\text{MSK} = (a, b, c)$.

PKG uses its identity ID_u provided by the data owner to generate its public and private key pair $\text{KeyGen}(\text{MSK}, \text{PK}, \text{ID}_u) \longrightarrow (\text{PK}_u, \text{SK}_u)$, and the cloud service provider obtains the key pair in the same way. PKG randomly selects parameters $t, x, y, z \in \mathbb{Z}_p^*$ to calculate the private key of the data owner as follows:

$$\begin{aligned} A_1 &= \frac{c+t}{a+b\cdot \text{ID}_u} A_2 = h^t A_3 = g^t, \\ B_1 &= \frac{a+x}{a+b\cdot \text{ID}_u} B_2 = \frac{b+x}{a+b\cdot \text{ID}_u} B_3 = \frac{z}{a+b\cdot \text{ID}_u}, \\ D_1 &= h^x D_2 = h^y D_3 = h^z. \end{aligned} \tag{1}$$

Among them, (A_1, A_2, A_3) is used to recover the ciphertext, and $(B_1, B_2, B_3, D_1, D_2, D_3)$ is used to generate the reencryption key.

6.3. Smart Contract Release Phase. The cloud service provider uses zk-SNARKs to generate a zero-knowledge proof π' that includes some of its attribute requirements, the calculation result R', and the hash value is recorded in the smart contract and at the same time publish some keywords for data requirements. The generation process of the zero-knowledge proof will be described below from the perspective of the data owner, and the zero-knowledge proof process of the cloud service organization is similar.

6.4. Encryption Phase. After the data owner generates the private data, it will encrypt private data$D = \langle d_1, d_2, \cdots, d_n \rangle$ by $\text{Encrypt}(\text{PK}, \text{PK}_u, \langle d_1, d_2, \cdots, d_n \rangle) \longrightarrow C_{\text{PK}_u}$, where $C_{\text{PK}_u} = (c_{pk_1}, c_{pk_2}, \cdots, c_{pk_n})$ is that PKG randomly selects $r, s \in \mathbb{Z}_p^*$ to calculate the following parameters.

$$\begin{aligned} c_{pk_1} &= D \cdot e(g, h)^{c(r+s)}, \\ c_{pk_2} &= g^r, \\ c_{pk_3} &= h^s, \\ c_{pk_1} &= e(g, h)^{(a+b\cdot ID_u)(r+s)}. \end{aligned} \tag{2}$$

Then, the data owner uploads the ciphertext C_{PK_u} to the semitrusted proxy cloud server for storage. We assume that the semitrusted proxy cloud server will not modify the data of the data owner without authorization, and it will perform the operations we set honestly.

6.5. Data Record On-Chain Phase. The data owner will store the hash value and digital signature of the data record on the blockchain platform, and the private data will be encrypted and stored on the proxy cloud server. The data owner will submit the hash value of his private data $D = \langle d_1, d_2, \cdots, d_n \rangle$ to generate the transaction form shown in Figure 3 and attach his digital signature $\sigma_a = \text{Authsign}(\text{SK}_u, H(\langle d_1, d_2, \cdots, d_n \rangle))$ to it. When the transaction is verified by the verification node, it is recorded in the blockchain.

6.6. Generate Zero-Knowledge Proof Phase. When the data owner's private data meets the keyword requirements provided by the cloud service provider, the data owner attaches his digital signature and local time information to the private data and submits it to zk-SNARKs to generate a zero-knowledge proof. The construction process is as follows.

Step 1. According to the private data $D = \langle d_1, d_2, \cdots, d_n \rangle$ of the data owner ID_u, the local time T generates auxiliary information $\delta = (D, T, \text{ID}_u)$.

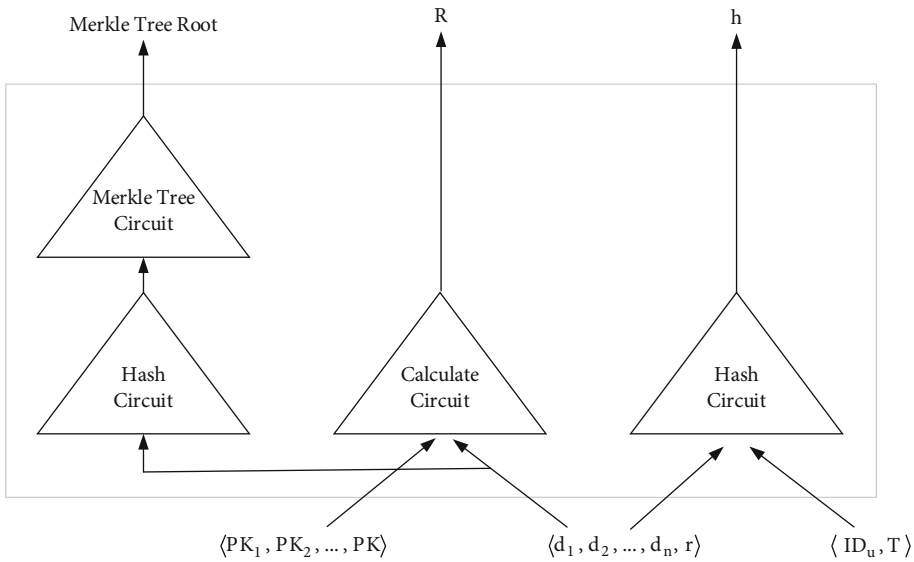


Figure 4: Circuit structure diagram.

Step 2. Select the random number r and the auxiliary information $\delta = (D, T, \mathrm{ID}_u)$ to calculate the hash value $H(\delta, r)$ and then generate the digital signature $\sigma_a = \mathrm{Authsign}(\mathrm{SK}_p, H(\delta, r))$ with the private key of the data owner.

Step 3. The data owner constructs the circuit $C : \mathbb{F}^n \times \mathbb{F}^h \longrightarrow \mathbb{F}^l$. Circuit input public parameters $\langle \mathrm{PK}_1, \mathrm{PK}_2, \cdots, \mathrm{PK}_n \rangle$, private data $D = \langle d_1, d_2, \cdots, d_n, r \rangle$, data owner identification information $\langle \mathrm{ID}_u, T \rangle$, where T and r are timestamps and random numbers, respectively. The output result R and the hash value h verify the authenticity and availability of the data.

$$C(\langle d_1, d_2, \cdots, d_n, r \rangle) \longrightarrow (R, h). \tag{3}$$

The circuit structure used in this paper is given in Figure 4.

Step 4. Enter the security parameter λ and the circuit C in the calculation task to calculate the key pair $(\mathrm{EK}_C, \mathrm{VK}_C)$, where EK_C is used to generate the zero-knowledge proof and VK_C is used to verify the zero-knowledge proof.

$$\mathrm{ZKPKeyGen}\left(1^\lambda, C\right) \longrightarrow (\mathrm{EK}_C, \mathrm{VK}_C). \tag{4}$$

Step 5. Prove algorithm consists of the generated key EK_C of the zero-knowledge proof, the private data D of the data owner, and the calculation result (R, h) in Step 3 together to generate the zero-knowledge proof π.

$$\mathrm{Prove}(\mathrm{EK}_C, D, R, h, \sigma_a) \longrightarrow \pi. \tag{5}$$

6.7. Zero-Knowledge Proof Verification Phase. The data owner submits the zero-knowledge proof to the smart contract. Then, the smart contract will automatically verify whether the zero-knowledge proof meets the requirements of the cloud service provider.

$$\mathrm{Verify}(\mathrm{VK}_C, \mathrm{PK}_u, \pi, R, h, \sigma_a) \longrightarrow (0, 1). \tag{6}$$

The smart contract first uses the public key of the data owner to verify its signature and then uses the verification key of zk-SNARKs to verify the zero-knowledge proof. After the verification is passed, the smart contract will automatically compare the zero-knowledge proof π of the data owner, the calculation result R, the hash value h, the zero-knowledge proof π' of the cloud service organization, the calculation result R', and the hash value h'. After the verification is completed, if the verification is correct, output 1; otherwise, output 0.

6.8. Reencryption Phase. After the verification is passed, the data owner uses the public key provided by the cloud service organization to generate the conversion key.

$$\mathrm{ReKeyGen}(\mathrm{PK}, \mathrm{SK}_u, \mathrm{PK}_d) \longrightarrow \mathrm{RK}_{u \longrightarrow d}. \tag{7}$$

PKG randomly selects the parameter $k_1, k_2 \in \mathbb{Z}_p^*$ and calculates as follows:

$$\begin{aligned} rk_1 &= (k_1 B_3 + B_1) + (k_2 B_3 + B_2) * \mathrm{ID}_u, \\ rk_2 &= \left(D_1 D_3^{k_1}\right)\left(D_2 D_3^{k_2}\right)^{\mathrm{ID}_u}, \\ \mathrm{RK}_{u \longrightarrow d} &= (rk_1, rk_2). \end{aligned} \tag{8}$$

The semitrusted proxy cloud server will convert the ciphertext into an intermediate ciphertext that can be decrypted by the cloud service organization after receiving the conversion key encrypted by the public key of the data owner. The proxy cloud server sends the intermediate

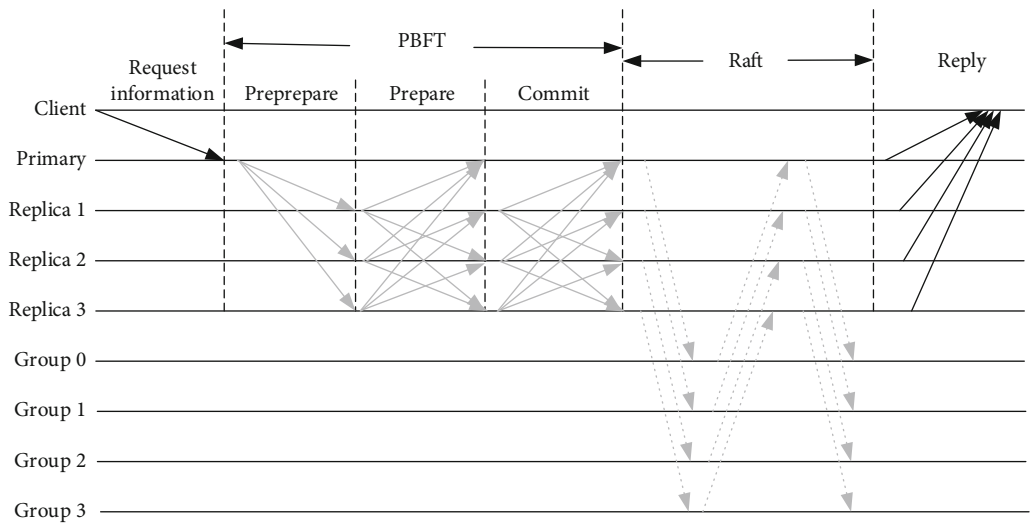


Figure 5: RBFT consensus process.

ciphertext $C_{\mathrm{PK}_{u\longrightarrow d}}$ to the cloud service organization.

$$\begin{aligned} c'_{pk_1} &= c_{pk_1}, \\ c'_{pk_2} &= c_{pk_2}, \\ c'_{pk_3} &= \frac{c_{pk_3}^{k_1}}{e\left(c_{pk_2}, rk_2\right)}, \\ C_{\mathrm{PK}_{u\longrightarrow d}} &= \left(c'_{pk_1}, c'_{pk_2}, c'_{pk_3}\right). \end{aligned} \tag{9}$$

6.9. Decryption Phase. When the cloud service provider receives the intermediate ciphertext obtained from the proxy cloud server, it decrypts the ciphertext with its private key. During the data sharing process, the semitrusted proxy cloud server cannot obtain any related information in cleartext. The cloud service provider uses the private key to decrypt the intermediate ciphertext to obtain the private data D.

$$\begin{aligned} &\mathrm{Decrypt}\left(\mathrm{PK}, C_{\mathrm{PK}_{u\longrightarrow d}}, \mathrm{SK}_d\right) \longrightarrow D, \\ &D = \frac{c_{pk_1} \cdot e\left(c_{pk_2}, A_2\right)}{C_{pk_1}^{A_1}}. \end{aligned} \tag{10}$$

6.10. Consensus Phase. The single consensus algorithm of the alliance chain cannot meet the environmental characteristics of low latency and high throughput in the Industrial Internet of Things environment. Combining the characteristics of PBFT [24] and Raft [25], a two-level mechanism is adopted to meet the environmental characteristics of the Industrial Internet of Things. The nodes are grouped, and the Raft consensus mechanism with supervisory nodes is used in the group, which has higher fault tolerance. The leadership committee elected by the Raft consensus mechanism uses the PBFT consensus mechanism, with reduced latency, improved throughput, and higher security. The specific process is shown in Figure 5. RBTF consensus process is as follows.

PBFT stage:

Step 1. After receiving client C's request, the master node (mian) will sort and sign the transactions and broadcast the prepacked message.

Step 2. After the secondary node (Replica) receives more than $2f$ messages, after verifying that the signature and other information are valid, it broadcasts a preparation message with an identity verification message.

Step 3. After receiving more than $2f + 1$ messages, the secondary node (Replica) judges whether the preparation phase is completed and enters the Raft consensus phase.

Raft stage:

Step 4. The leader in Raft broadcasts the message.

Step 5. After the follower nodes receive the message, they will verify the feedback.

Step 6. The leader node judges whether a consensus is reached according to the feedback result and submits the log.

Step 7. After completing the consensus, return the consensus result to the smart contract and write it into the blockchain ledger.

Related variables of RBFT:

(1) The RBFT consensus mechanism needs to meet the number of groups $k \geq 4$ and the number of nodes in the group $m \geq 3$

(2) The maximum fault tolerance range $f \leq \lfloor (k-1)/3 \rfloor$ in the PBFT phase and the maximum fault tolerance range in the Raft phase is $f \leq \lfloor (m-1)/2 \rfloor$

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

Scheme	Not rely on trusted third parties	Content privacy	Identity privacy	Data validity	Verifiability	Traceability
[16]	√	√	√	×	−	√
[20]	√	√	−	×	√	×
[21]	√	×	×	√	−	×
[22]	√	√	√	×	√	×
[23]	×	√	−	×	√	−
This paper	×	√	√	√	√	√

7. Specific Structure

7.1. Security Proof

Lemma 1. *Based on the DBDH assumption, our scheme can resist selected plaintext attacks under the random oracle model; our solution is IND-CPA secure, that is, it satisfies the indistinguishability under the selected plaintext attack.*

Proof. Assuming that A is an adversary of arbitrary polynomial time, it can access the reencryption key oracle PKGen() and the reencryption oracle ReEnc(). The games of adversary A and challenger B are as follows.

Stage 1:

(1) Preparation stage: B randomly selects $a, b, c \in \mathbb{Z}_p^*$ to generate $(\mathrm{PK}, \mathrm{MSK}) \longleftarrow \mathrm{Setup}(1^\lambda)$

(2) Key generation: A initiates a key inquiry; B calculates $(pk_A, sk_A) \longleftarrow \mathrm{KeyGen}(\mathrm{MSK}, \mathrm{PK}, \mathrm{id}_A)$ and sends (pk_A, sk_A) to$A$

Stage 2:

Reencryption key generation: when A initiates a query for generating a reencryption key, according to the query information $(sk_{A'}, pk_{A'})$, B randomly selects $k_1, k_2 \in \mathbb{Z}_p^*$ to calculate (rk_1, rk_2) and returns $\mathrm{RK}_{A\longrightarrow A'} \longleftarrow \mathrm{RKGen}(\mathrm{PK}, sk_{A'}, pk_{A'})$ to A.

Reencryption: when A initiates a reencryption query, B returns $C_{\mathrm{PK}_{A\longrightarrow A'}} \longleftarrow \mathrm{ReEnc}(\mathrm{PK}_{A\longrightarrow A'}, c)$ and sends $C_{\mathrm{PK}_{A\longrightarrow A'}}$ to A.

Challenge: let $m_0, m_1 \in M$. A initiates a challenge and inquiry; B randomly selects a bit $\sigma \in (0, 1)$ and sends $C^* \longleftarrow \mathrm{Enc}(pk_{A'}, m_\sigma)$ to the opponent A. □

The opponent A outputs a guess value $\sigma' \in (0, 1)$; if $\sigma = \sigma'$, the opponent wins the game. The advantage of the adversary A can be defined as $\mathrm{Adv}_{\prod_A}^{\mathrm{CPA}}(\kappa) = |\Pr[\sigma' = \sigma] - (1/2)|$. The probability that the adversary A guesses the correct σ' in the PPT is $(1 + \varepsilon)/2$, and ε is a nonnegligible quantity. The adversary A wins the game with at least the probability of $\varepsilon' = (1 + \varepsilon)/2 - 1/2 = \varepsilon/2$. Therefore, the advantage of the adversary A in the game can be ignored, and this solution is IND-CPA safe.

8. Performance Analysis

8.1. Content Privacy. In this solution, all private data is encrypted by the data owner using a sufficiently secure encryption algorithm and then uploaded to the proxy cloud server. We assume that the encryption algorithm used is sufficiently secure under the security model. If the key cannot be obtained, any internal adversary or external adversary cannot obtain the ciphertext. The data owner uses the private key of the cloud service provider to generate a reencryption key, which is reencrypted by the proxy cloud server and sent to the cloud service provider. Therefore, only authorized institutions can decrypt to obtain the ciphertext, and other entities in the process cannot obtain the ciphertext information.

8.2. Identity Privacy. When each participant entity registers, the identity certificate authority in the alliance chain will strictly examine the legality of the data owner or cloud service provider's identity and generate pseudoidentities for participants to ensure the privacy of their identities in transactions. In data sharing transactions, the real identity of the user cannot be obtained in the interaction between the participating entities and the smart contract. The cloud service provider only publishes the required keywords to achieve partial privacy protection and prevent the data owner from forging false data.

8.3. Data Validity. In this solution, only organizations authorized by the data owner can decrypt private data. The data owner generates a zero-knowledge proof π based on the private data. After submitting it to the smart contract, it can automatically verify whether the data meets the data requirements of the cloud service provider. Ensure the validity of the data.

8.4. Verifiability. The data owner sends the digital signature together with the generated zero-knowledge proof, and other entities can verify the validity of the signature, ensuring the validity of the zero-knowledge proof. The hash of the data is stored on the blockchain. The characteristics of the blockchain ensure that the data cannot be tampered with, and other entities receiving the data can verify the integrity of the data.

8.5. Traceability. In this solution, when the data owner and the cloud service provider reach a transaction on the premise that the data is complete and valid, the transaction is stored

in the blockchain. If the data owner fails to abide by the promise of no longer selling information to others and sells the information multiple times, the transaction history can be traced back in the blockchain to impose punishment.

9. Performance Analysis

Through comparative analysis of existing data sharing schemes, literature [20] uses time-based zero-knowledge proof and authentication encryption to perform mutual authentication between multiple attributes, ensuring security and privacy in data sharing. Literature [16] combined with zero-knowledge proof technology to achieve confidentiality and correctness in the data sharing process. Literature [21] realizes the reliability of data in data sharing among participants through an incentive mechanism, but the shared data is neither anonymous nor encrypted. Literature [22] realizes the sharing of vehicle data, but when there is an error in the data transaction, the traceability of the data source cannot be realized. Literature [23] uses proxy reencryption based on blockchain to store and share Internet of Things data in a cloud proxy server to achieve confidentiality and integrity in the data sharing process, but it cannot guarantee the validity and consistency of the data.

This article realizes the validity and consistency of the data sharing process under the premise of protecting data privacy and the privacy of the data owner's identity. Utilize the immutability and traceability of the blockchain to realize data integrity and traceability of data transactions. The comparison between this article and other programs is shown in Table 2.

10. Conclusions

Blockchain combined with zero-knowledge proof provides a new solution to the data sharing model. A large amount of data in the Industrial Internet of Things is the basis for promoting better development of services. How to maintain data privacy as much as possible on the premise of effective use of data is an important issue facing now. In response to these problems, this article combines zero-knowledge proof and smart contracts to achieve data validity and consistency between data owners and cloud service providers. Use proxy reencryption technology to realize the safe sharing of data among multiple participants. And combined with the non-tamperable and traceable characteristics of the blockchain, the data can be verified and the transaction can be traced. Future work will study the realization of the secure sharing of data without a third-party server and the realization of a completely decentralized data sharing scheme.

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 is supported by the National Natural Science Foundation of China (Grant Nos. 62162039 and 61762060) and the Foundation for the Key Research and Development Program of Gansu Province, China (Grant No. 20YF3GA016).

References

[1] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund, "Industrial Internet of things: challenges, opportunities, and directions," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 11, pp. 4724–4734, 2018.

[2] Q. Li, B. Xia, H. Huang, Y. Zhang, and T. Zhang, "TRAC: traceable and revocable access control scheme for mHealth in 5G-enabled IIoT," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 5, pp. 3437–3448, 2022.

[3] Z. Li, J. Kang, R. Yu, D. Ye, Q. Deng, and Y. Zhang, "Consortium blockchain for secure energy trading in industrial Internet of things," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 8, pp. 3690–3700, 2017.

[4] J. Xiong, R. Bi, M. Zhao, J. Guo, and Q. Yang, "Edge-assisted privacy-preserving raw data sharing framework for connected autonomous vehicles," *IEEE Wireless Communications*, vol. 27, no. 3, pp. 24–30, 2020.

[5] J. Xiong, R. Ma, L. Chen et al., "A personalized privacy protection framework for mobile crowdsensing in IIoT," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 4231–4241, 2020.

[6] E. B. Sasson, A. Chiesa, C. Garman et al., "Zerocash: decentralized anonymous payments from bitcoin," in *2014 IEEE symposium on security and privacy*, pp. 459–474, Berkeley, CA, USA, 2014.

[7] A. Rondelet and M. Zajac, *ZETH: On Integrating Zerocash on Ethereum*, 2019.

[8] J. Li, S. Wu, Y. Yang, F. Duan, H. Lu, and Y. Lu, "Controlled sharing mechanism of data based on the consortium blockchain," *Security and Communication Networks*, vol. 2021, 10 pages, 2021.

[9] F. Loukil, C. Ghedira-Guegan, K. Boukadi, and A. N. Benharkat, "Privacy-preserving IoT data aggregation based on blockchain and homomorphic encryption," *Sensors*, vol. 21, no. 7, p. 2452, 2021.

[10] X. Qin, Y. Huang, Z. Yang, and X. Li, "LBAC: a lightweight blockchain-based access control scheme for the Internet of things," *Information Sciences*, vol. 554, pp. 222–235, 2021.

[11] Y. Tian, Z. Wang, J. Xiong, and J. Ma, "A blockchain-based secure key management scheme with trustworthiness in DWSNs," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 9, pp. 6193–6202, 2020.

[12] Y. Tian, T. Li, J. Xiong, M. Z. Bhuiyan, J. Ma, and C. Peng, "A blockchain-based machine learning framework for edge services in IIoT," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 3, pp. 1918–1929, 2022.

[13] B. Muthusenthil, D. Nivetha, and H. Kim, "Reencryption scheme for secure data sharing," in *2016 International Conference on Communication and Signal Processing (ICCSP)*, pp. 1170–1174, Melmaruvathur, India, 2016.

[14] N. Mahakalkar and V. Sahare, "Implementation of reencryption based security mechanism to authenticate shared access in cloud computing," in *2017 International Conference*

on Trends in Electronics and Informatics (ICEI), pp. 547–550, Tirunelveli, India, 2017.

[15] Y. Fan, Y. Liao, F. Li, S. Zhou, and G. Zhang, "Identity-based auditing for shared cloud data with efficient and secure sensitive information hiding," *IEEE Access*, vol. 7, pp. 114246–114260, 2019.

[16] C. Huang, D. Liu, J. Ni, R. Lu, and X. Shen, "Achieving accountable and efficient data sharing in industrial Internet of things," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 2, pp. 1416–1427, 2021.

[17] M. J. M. Chowdhury, A. Colman, M. A. Kabir, J. Han, and P. Sarda, "Blockchain as a notarization service for data sharing with personal data store," in *2018 17th ieee international conference on trust, security and privacy in computing and communications/12th ieee international conference on big data science and engineering (TrustCom/BigDataSE)*, pp. 1330–1335, New York, NY, USA, 2018.

[18] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Blockchain and federated learning for privacy-preserved data sharing in industrial IoT," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 4177–4186, 2020.

[19] Y. Wang, A. Zhang, P. Zhang, and H. Wang, "Cloud-assisted EHR sharing with security and privacy preservation via consortium blockchain," *IEEE Access*, vol. 7, pp. 136704–136719, 2019.

[20] A. S. Sani, D. Yuan, W. Bao et al., "Xyreum: a high-performance and scalable blockchain for IIoT security and privacy," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pp. 1920–1930, Dallas, TX, USA, 2019.

[21] M. Shen, J. Duan, L. Zhu, J. Zhang, X. Du, and M. Guizani, "Blockchain-based incentives for secure and collaborative data sharing in multiple clouds," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 6, pp. 1229–1241, 2020.

[22] D. E. Kouicem, A. Bouabdallah, and H. Lakhlef, "An efficient and anonymous blockchain-based data sharing scheme for vehicular networks," in *2020 IEEE Symposium on Computers and Communications (ISCC)*, pp. 1–6, Rennes, France, 2020.

[23] A. Manzoor, A. Braeken, S. S. Kanhere, M. Ylianttila, and M. Liyanage, "Proxy re-encryption enabled secure and anonymous IoT data sharing platform based on blockchain," *Journal of Network and Computer Applications*, vol. 176, article 102917, 2021.

[24] R. Kashyap, K. Arora, M. Sharma, and A. Aazam, "Security-aware GA based practical byzantine fault tolerance for permissioned blockchain," in *2019 4th International Conference on Control, Robotics and Cybernetics (CRC)*, Tokyo, Japan, 2019.

[25] L. Hou, X. Xu, K. Zheng, and X. Wang, "An intelligent transaction migration scheme for RAFT-based private blockchain in Internet of things applications," *IEEE Communications Letters*, vol. 25, no. 8, pp. 2753–2757, 2021.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 5721212, 11 pages
https://doi.org/10.1155/2022/5721212

Research Article

An Immunity Passport Scheme Based on the Dual-Blockchain Architecture for International Travel

Hancheng Gao,[1,2] Haoyu Ji,[1,2] Haiping Huang,[1,2] Fu Xiao,[1,2] and Luo Jian[1]

[1]*College of Computer, Nanjing University of Posts and Telecommunications, Nanjing 210023, China*
[2]*High Technology Research Key Laboratory of Wireless Sensor Network of Jiangsu Province, Nanjing 210023, China*

Correspondence should be addressed to Haiping Huang; hhp@njupt.edu.cn

Received 18 October 2021; Revised 9 December 2021; Accepted 20 December 2021; Published 10 January 2022

Academic Editor: Jinguang Han

The implementation of immunity passport has been hampered by the controversies over vaccines in various countries, the privacy of vaccinators, and the forgery of passports. While some existing schemes have been devoted to accelerating this effort, the problems above are not well solved in existing schemes. In this paper, we present an immunity passport scheme based on the dual-blockchain architecture, which frees people from the cumbersome epidemic prevention process while traveling abroad. Specially, the dual-blockchain architecture is established to fit with the scenarios of immunity passport. Searchable encryption and anonymous authentication are utilized to ensure users' privacy. In addition, the performance and security evaluations show that our scheme achieves the proposed security goals and surpasses other authentication schemes in communicational and computational overheads.

1. Introduction

The Coronavirus Disease 2019 (COVID-19) pandemic is undoubtedly an unprecedented disaster for human society [1–3]. The pandemic is rapidly spreading and getting worse in many countries and regions of the world, which has caused a large number of infections and deaths. Countries around the world are doing their utmost to curb the spread of the pandemic, enacting strict policies such as quarantine for infected people, prohibitions on mass gatherings, and restrictions on entry-exit and so on.

Vaccination, in combination with personal protection, is the most effective measure to prevent the COVID-19 [4]. However, the effectiveness of some vaccines remains controversial in countries because of differences in policies, technical standards, and religions. As shown by recent publications, not everyone holds a positive attitude towards the COVID-19 vaccine [5, 6]. There are even discriminations against unvaccinated people in some areas, which is called stigmatization of vaccination [7, 8].

Restoring the order of human society in the postepidemic era is one of the most important issues, among which lifting restrictions on people's entry-exit is particularly significant. The restrictions on the people who have been vaccinated could be relaxed [9]. Therefore, a number of countries and organizations have launched the immunity passport that allows them to work and travel abroad without compromising personal or public health [8, 10]. However, some serious issues remain unresolved: (1) traditional passports are easy to falsify. (2) There are controversies about the effectiveness of some vaccines among different countries. (3) Under the premise of stigmatization, vaccinators' privacy is still at risk.

To effectively ensure the privacy of people traveling during the COVID-19 pandemic, we propose an immunity passport scheme in this paper. In our scheme, vaccinated people can show their passports to a staff of customs without compromising their privacy for entry and exit. Our contributions are summarized as follows:

(1) In order to adapt our scheme to the international travel scenarios, we designed a dual-blockchain architecture with two different types of blockchain, domestic and international. Different countries participate in the consensus of the international

blockchain, which is conducive to solving the controversies about vaccines.

(2) We leveraged the use of the inherent characteristics of blockchain to make the immunity passport traceable and nonrepudiable. And for the purpose that users can have control over their data, we combined searchable encryption and anonymous authentication with blockchain.

(3) Our scheme allows users to participate in vaccination, authentication, and other processes using legitimate pseudonyms, which can well solve the stigmatization of vaccination.

(4) To prove the feasibility and reliability of our scheme, we conducted a complete security analysis and simulation experiments, including computational overheads, communication overheads, and energy overheads.

The rest of this paper is organized as follows. Section 2 discusses some related research achievements. Section 3 describes the preliminary knowledge and introduces the design details of the system model. The immunity passport scheme is proposed in Section 4. Section 5 presents the correctness and security analysis. Section 6 presents the performance evaluation, and Section 7 concludes this paper.

2. Related Work

Due to its outstanding characteristics, blockchain technology has attracted widespread attention in many fields including medical care, identity authentication, and finance [11–13]. Recently, there have been some studies applying blockchain technology to meet the challenges of COVID-19. Xu et al. [14] proposed a blockchain-enabled privacy preserving contact tracing scheme, in which users' privacy is ensured by the pseudonym. However, their scheme has a high demand for the intensive computation of blockchain nodes. In order to control the spread of COVID-19, a privacy anonymous IoT model using blockchain was presented in [15]. In this scheme, people who wear RFID tags will be notified if they are near to the possible or confirmed "hotspot" area. But the authors did not give a security analysis of the scheme in this paper. Song et al. [16] using Bluetooth technology designed a tracing and notification system based on blockchain and smart contract to ensure users' privacy. However, there is an unreasonable assumption that people always honestly upload their health status to the blockchain. Jacob and Lawarée [17] pointed out that apps such as to StopCovid (France), NHS Covid-19 (UK), and Coronalert (Belgium) have security, political, and other issues. Although these schemes and applications are focused on addressing the issues of privacy, the public is still reluctant to disclose their personal data for privacy reasons [18, 19]. Moreover, contact tracing is a passive defense against the COVID-19 pandemic.

Hasan et al. [20] proposed a digital health passport system combining blockchain, proxy reencryption, and smart contracts. In this system, the data owner grants access to other entities so that the user has control over his data. Based on blockchain, a framework was proposed in [21] to ensure users' privacy, which uses a locality-sensitive hash function to generate a secure identifier. The identifier can only be derived if the user provides his biometric and personal information, whereas, although the authors give details of the pseudoidentity generation, the description of the vaccination certificate is very brief. Angelopoulos et al. [22] presented a framework that used a private blockchain to store the digital health passport. But the authors did not give details about how to ensure users' privacy, and the characteristics of private blockchain did not apply to the scenarios where people travel among multiple countries.

None of the above researches [20–22] addressed how the passport holder can verify the legality of inspectors, which is extremely important for users. Some existing authentication schemes are designed for scenarios such as the smart grid, the Internet of Things, and the smart medical [23–26]. Mahmood et al. [23] proposed an anonymous key agreement protocol for the smart grid infrastructure by using the identity-based signature. This protocol empowers the smart meters for anonymous information exchange with utility, which is proved secure under the random oracle model. A mutual authentication scheme focusing on mobile edge computing is proposed by Jia et al. [24], which only needs one massage exchange round to achieve mutual authentication. However, their scheme cannot achieve some security properties. Almadhoun et al. [25] proposed a decentralized and scalable authentication mechanism that utilizes blockchain-enabled fog nodes with connectivity to Ethereum smart contracts, which gives details of smart contracts involved. Although all the above schemes have advantages and highlights, these authentication schemes are not suitable for the scenarios of immunity passport.

It is noteworthy that the above schemes have some shortcomings when applied to epidemic prevention scenarios, which makes the privacy of users cannot be guaranteed well. Therefore, it is meaningful to design a secure, reliable, and efficient immunization passport scheme for the COVID-19 epidemic.

3. System Model and Security Goals

In this section, we give a brief introduction to the basic theoretical knowledge involved in this paper, such as blockchain, searchable encryption, and bilinear mapping. Subsequently, the system model and security goals are presented. The system model is depicted in Figure 1, and the main notations that appear in the scheme are listed in Table 1.

3.1. Preliminaries. Blockchain. Blockchain is a special kind of data structure that arranges a large number of blocks into a chain in chronological order, where each block is composed of certain data [27]. Blockchain is categorized roughly into public blockchain, consortium blockchain, and private blockchain according to the degree of decentralization. Our scheme adopts the consortium blockchain because of the specific advantages: (1) it can be jointly controlled by multiple organizations or countries, which is suitable for the

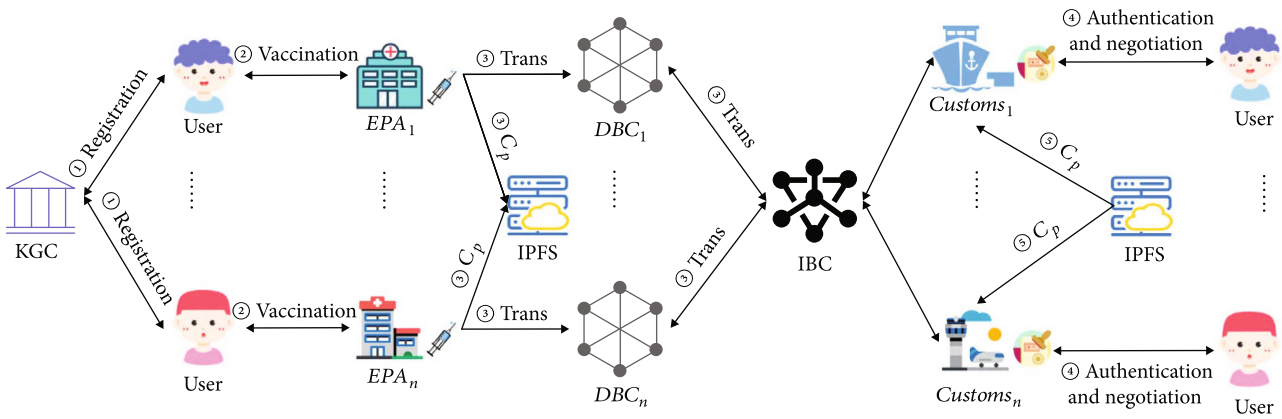


Figure 1: System model of the immunity passport scheme.

Table 1: Notations in the scheme.

Notations	Description
U	User
$\mathbb{G}_1, \mathbb{G}_2$	Multiplicative cyclic groups
$H_i (i = 1, \cdots, 6)$	Secure hash function
PK, SK	Public and private key pair of KGC
PK_c, SK_c	Public and private key pair of IBC
pk_i, sk_i	Public and private key pair of user
D_i	Partial-private key
ID_i'	Pseudoidentity of user
ID_{ctry}	Identity of the country
ID_{DB}	ID of the DBC-block
idx	Index of keyword
C_p	Ciphertext of the passport
Confir	Key confirmation message
K_p	Decryption key
T, T'	Trapdoor

scenarios of our scheme. (2) Only the members of the consortium participate in the consensus, so it has high efficiency. (3) Not everyone can access the data on the consortium blockchain.

Searchable Encryption. Searchable encryption is a cryptographic primitive that supports users to conduct keyword search on encrypted data. It mainly solves how to complete the search for encrypted data when the data is encrypted and stored in the cloud, under the premise that the cloud server is not completely trusted. Similar to searching for plaintext data, a common method for searchable encryption is to establish a secure index for the entire dataset and then use the secure index to complete a secure search for encrypted data on the cloud server. Searchable encryption enhances the scalability of search while saving users a lot of network and computing overhead.

Bilinear Pairings. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two multiplicative cyclic groups with the prime order as p. Let g be the generator of $\mathbb{G}_1$, which means $\mathbb{G}_1 = \langle g \rangle$. We accept e as bilinear pairing if $e : \mathbb{G}_1 \times \mathbb{G}_1 \longrightarrow \mathbb{G}_2$ satisfies the following properties [28]:

(1) Bilinearity. For all $g_1, g_2 \in \mathbb{G}_1$, $a, b \in \mathbb{Z}_p^*$, there is $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$.

(2) Nondegeneracy. There exists $g_1, g_2 \in \mathbb{G}_1$, such that $e(g_1, g_2) \neq 1$.

(3) Computability. For all $g_1, g_2 \in \mathbb{G}_1$, there exists an efficient algorithm to compute $e(g_1, g_2)$.

3.2. System Model. In the model of the immunity passport scheme, it is assumed that various epidemic prevention agencies (EPAs) in each country form an alliance and jointly maintain a domestic consortium blockchain, that is the "Domestic Blockchain (DBC)." Every country selects an institution with high credibility on behalf of the country to maintain an international consortium blockchain, that is the "International Blockchain (IBC)." Since we use consortium blockchains to design the system model, popular consensus mechanisms adapted to consortium blockchains can be run on our scheme, such as Practical Byzantine Fault Tolerance (PBFT) and Delegated Proof of Stake (DPoS) [29, 30]. Thus, our scheme focuses on how to efficiently authenticate the identity and verify the validity of the passport. The seven entities and two structures of transaction in this model are described in detail as follows:

Key Generate Center (KGC). KGC is an organization with high credibility in this system, which is responsible for generating system parameters and distributing partial-private keys for all users.

Users. The user is vaccinated at EPA by virtue of the legal pseudoidentity. The user generates a trapdoor and a decryption key for the staff when he needs the immunity passport; the ciphertext of passport is then searched by the IBC node and returned by the IPFS.

Inter-Planetary File System (IPFS). IPFS is a decentralized file storage network used to store the ciphertext of passports generated by the EPAs.

Epidemic Prevention Agency (EPA). EPAs maintain a DBC in each country, responsible for vaccinating, generating

Table 2: Structure of transaction on the DBC.

Block header	Timestamp t		Block ID ID_{DB}	Size size	Prehash hash
Transaction	Producer ID_{EPA}	User ID ID'	Keyword-index $\{idx, w\}$	Hash of Ciphertext $\mathrm{hash}(C_p)$	Signature $\mathrm{sig}_{\mathrm{EPA}}$

Table 3: Structure of transaction on the IBC.

Block header	Timestamp t	Block ID ID_{IB}	Size size	Prehash hash
Transaction	Producer ID_{ctry}	Search-index $\left(ID_{DB}, ID', \{idx, w\}, \mathrm{hash}(C_p)\right)$		Signature $\mathrm{sig}_{\mathrm{ctry}}$

immunity passports, and uploading the ciphertext of passports to the IPFS. And EPAs participate in the consensus of DBC to generate new blocks.

Domestic Blockchain (*DBC*). There are many DBCs in our model. The role of DBC nodes is played by EPAs of each country and the transaction on DBC is broadcast by EPAs.

International Blockchain (*IBC*). Only one IBC exists in our model. The role of IBC nodes is played by institutions on behalf of countries, such as the ministry of health.

Customs. The staff of customs gets the ciphertext of passport and decrypt it after achieving mutual authentication with the user, where a session key is negotiated for transferring the trapdoor and the decryption key.

Structure of Transaction. We deployed two types of blockchain in our scheme, thus we designed different structures of transaction.

The structure of transactions on DBCs is shown in Table 2, including identity of EPA ID_{EPA} that generates the DBC-transaction, pseudoidentity of the inoculator ID', the keyword-index $\{idx, w\}$, hash of the ciphertext of the passport $\mathrm{hash}(C_p)$, and signature of the EPA $\mathrm{sig}_{\mathrm{EPA}}$.

The structure of transactions on the IBC is shown in Table 3, including identity of the country ID_{ctry} that generates the IBC-transaction, signature of the country $\mathrm{sig}_{\mathrm{ctry}}$, and search-index $(ID_{DB}, ID', \{idx, w\}, \mathrm{hash}(C_p))$. The search-index is composed of ID of the DBC-block, pseudoidentity of the inoculator, the keyword-index, and hash of the ciphertext of the passport.

3.3. Security Goals. We assumed that all blockchain nodes and customs staffs are semihonest, and attackers can eavesdrop on messages while users are communicating with other entities. Based on the assumption, we propose the following security goals.

Confidentiality and Privacy. Our scheme is based on the blockchain, and data stored on the blockchain is shared and transparent. The scheme needs to satisfy user's personal privacy and the confidentiality of immunity passports.

Mutual Authentication. In the proposed scheme, users need to communicate with customs staff. In order to ensure the legitimacy of two parties, they need to achieve mutual authentication before communication.

Traceability and Nonrepudiation. The EPA is responsible for user's health after vaccination. Accordingly, the goals of traceability and nonrepudiation should be achieved in our scheme.

Other Attacks. Furthermore, our scheme should also be able to resist other attacks, such as impersonation attack and insider attack.

4. The Proposed Scheme

In order to facilitate readers to better understand the application scenario, we have made a brief overview of the scheme before describing the details. For the convenience of presentation, it is assumed that the entire process takes user U_1 as an example, referring to Figure 1.

(1) Firstly, U_1 will get his legal pseudoidentity and his full public-private key pair by interacting with KGC.

(2) Then, U_1 is vaccinated at the EPA_1 after mutual authentication with the EPA_1.

(3) Subsequently, EPA_1 generates an immunity passport for U_1 and stores the ciphertext of the passport in IPFS, and two different transactions will be uploaded to DBC_1 and IBC, respectively.

(4) When U_1 travels through the customs, send the decryption key and trapdoor to the staff through the negotiated session key after mutual authentication. The staff will issue a request to the IBC node to search for the corresponding transaction.

(5) Finally, IPFS sends the ciphertext of the passport to the staff, who can verify user's vaccination information.

The detail scheme mainly contains the following phases.

4.1. System Setup and User-Registration. In this phase, KGC generates system parameters and its public-private key pair. The user obtains a legal pseudoidentity and generates his full public-private key pair through the partial-private key generated by KGC (as shown in Figure 2).

System-Setup. To generate system parameters, KGC chooses two multiplicative cyclic groups $\mathbb{G}_1$ and $\mathbb{G}_2$ with a prime order p, an element g, which is the generator of $\mathbb{G}_1$, and a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \longrightarrow \mathbb{G}_2$. KGC chooses a secret value $SK = x \in \mathbb{Z}_P^*$ and calculates PK as

$$PK = g^x. \tag{1}$$

The IBC node chooses a secret value $SK_c = x_c$ and calculates PK_c as

$$PK_c = g^{x_c}. \tag{2}$$

Then, KGC selects some secure hash functions $H_1 : \{0,1\}^* \times \mathbb{G}_1 \longrightarrow \{0,1\}^{n_a}$, $H_2 : \{0,1\}^* \times \mathbb{G}_1 \longrightarrow \mathbb{Z}_p^*$, $H_3 : \{0,1\}^* \times \mathbb{G}_1^2 \longrightarrow \mathbb{Z}_p^*$, $H_4 : \{0,1\}^* \longrightarrow \{0,1\}^{n_b}$, $H_5 : \mathbb{G}_2^2 \times \{0,1\}^* \longrightarrow \{0,1\}^{n_c}$, $H_6 : \{0,1\}^* \times \{0,1\}^* \times \mathbb{G}_1^3 \longrightarrow \{0,1\}^{n_d}$. The system parameters $params = <\mathbb{G}_1, \mathbb{G}_2, p, e, g, PK, H_1, H_2, H_3, H_4, H_5, H_6>$ are published.

User-Registration. KGC randomly picks $\mu \in \mathbb{Z}_P^*$, calculates g^μ, and sends g^μ to the user. The user chooses a secret value $x_i \in \mathbb{Z}_P^*$ and calculates X_i and his pseudo-identity ID_i' as

$$X_i = g^{x_i}, \tag{3}$$

$$ID_i' = H_1(ID_i \| (g^\mu)^{x_i}), \tag{4}$$

then sends ID_i' and X_i to KGC. KGC checks whether formula (5) is valid.

$$ID_i' = H_1\left(ID_i \| X_i^\mu\right). \tag{5}$$

If the equality holds, KGC picks $r_i \in \mathbb{Z}_P^*$ and calculates R_i, k_i, and d_i according to

$$R_i = g^{r_i}, \tag{6}$$

$$k_i = H_2\left(ID_i' \| R_i\right), \tag{7}$$

$$d_i = g^{r_i + k_i x}. \tag{8}$$

KGC sends the partial-private key $D_i = (R_i, d_i)$ to the user through a secure channel. The user sets his full public-private key pair to: $pk_i = (R_i, X_i)$ and $sk_i = (x_i, d_i)$.

Once the user has his legal pseudoidentity and full public-private key pair, he will use the pseudoidentity to participate in the next phases.

4.2. Passport Generation and Storage. In this phase, the EPA vaccinates the user and generates an immunity passport after authenticating user's pseudoidentity, then stores the ciphertext of the passport on IPFS. Subsequently, different types of transaction will be uploaded to IBC and DBC.

User-Authentication. The user chooses a secret value $u_i \in Z_q^*$, calculates U_i, h_i, and V_i according to

$$U_i = g^{u_i}, \tag{9}$$

$$h_i = H_3\left(ID_i' \| U_i \| X_i\right), \tag{10}$$

$$V_i = g^{h_i x_i + h_i u_i} \cdot d_i, \tag{11}$$

and sends a signature $\mathrm{sig}_i(U_i, V_i)$ to EPA. EPA calculates k_i, h_i as

$$k_i = H_2\left(ID_i' \| R_i\right), \tag{12}$$

$$h_i = H_3\left(ID_i' \| U_i \| X_i\right), \tag{13}$$

and checks Equation (14). If the equation holds, the user is considered legitimate.

$$e(V_i, g) = e\left(R_i \cdot PK^{k_i} \cdot X_i^{h_i}, g\right) e\left(U_i^{h_i}, g\right). \tag{14}$$

Passport-Storage. EPA generates the immunity passport passpt $\in \{0,1\}^*$ for the user, randomly picks $l_i \in Z_q^*$, and calculates L_i, the ciphertext of the passport C_p according to

$$L_i = X_i^{l_i}, \tag{15}$$

$$C_p = \left\{e\left(R_i \cdot PK^{k_i} \cdot X_i^{h_i}, g^{l_i}\right) \times \mathrm{passpt}, L_i\right\}, \tag{16}$$

and the hash of the ciphertext $\mathrm{hash}(C_p) = H_4(C_p)$. EPA gets sig_{EPA} by signing the $\mathrm{hash}(C_p)$, extracts the ID of vaccine $ID_v \in \{0,1\}^*$ as the keyword $w = ID_v$, and calculates the index of keyword idx as

$$idx = H_5\left(e\left(g^{l_i}, g\right) \| w\right). \tag{17}$$

EPA stores C_p in IPFS and broadcasts $\{ID_{EPA}, ID_i', \{idx, w\}, \mathrm{hash}(C_p), \mathrm{sig}_{\mathrm{EPA}}\}$ as a new transaction. Then, the transaction is uploaded to DBC after being verified by other EPAs. After a new block is generated on the DBC, the IBC node sets $(ID_{DB}, ID', \{idx, w\}, \mathrm{hash}(C_p))$ as the search-index and broadcasts $\{ID_{\mathrm{ctry}}, (ID_{DB}, ID', \{idx, w\}, \mathrm{hash}(C_p)), \mathrm{sig}_{\mathrm{ctry}}\}$ as a new transaction. After being verified by other countries, the transaction is uploaded to IBC.

After the end of this phase, the ciphertext of user's passport is stored in IPFS, the corresponding keyword-index and search-index are also uploaded to the blockchain as transaction information.

4.3. Identity Authentication and Key Agreement. In this phase, the user and customs staff perform identity authentication to confirm both of them are legitimate, and a secure session key is negotiated for subsequent data transmission, as depicted in Figure 3.

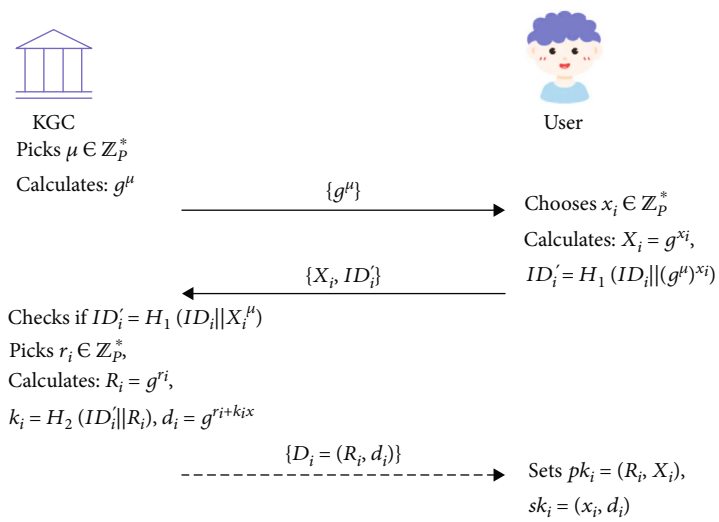


Figure 2: Generation of pseudoidentity and public-private key pair.

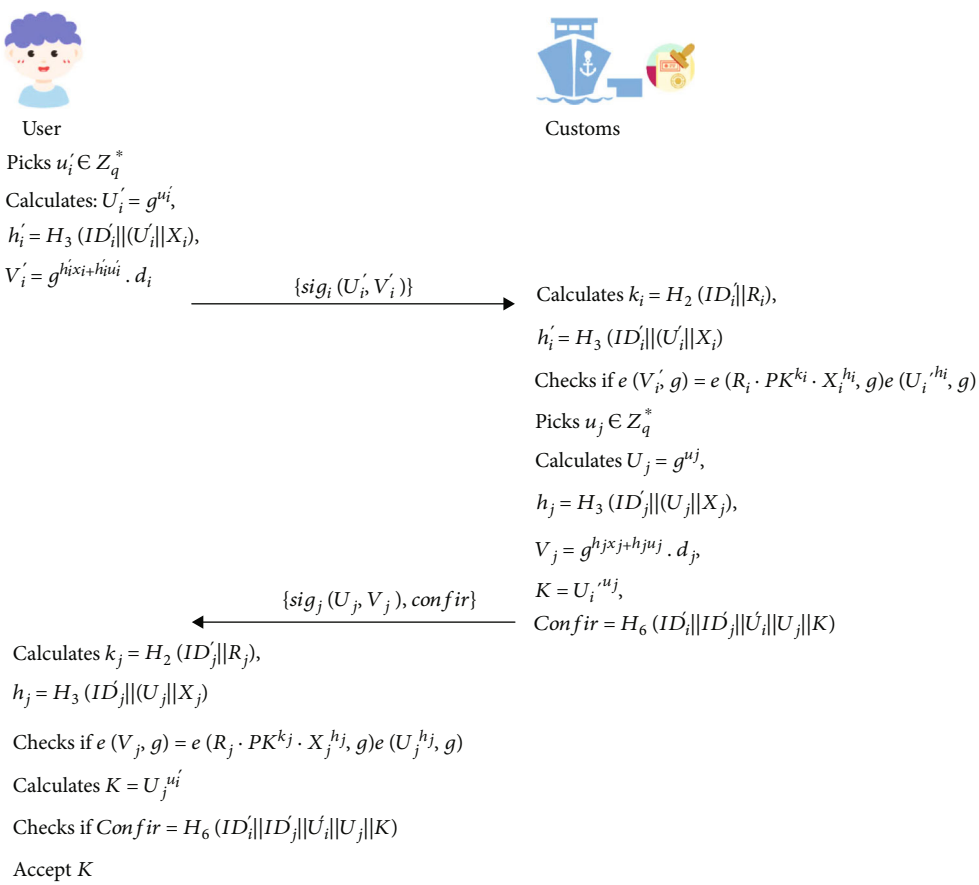


Figure 3: Identity authentication and key agreement for user and customs staff.

Authentication and Negotiation. The user randomly picks $u_i' \in Z_q^*$ and calculates U_i', h_i', V_i' to authenticate with the customs staff as in the "User-Authentication". After authenticating user's identity, the staff picks $u_j \in Z_q^*$ and calculates U_j, h_j, and V_j to obtain the signature $sig_j(U_j, V_j)$ according to

$$U_j = g^{u_j}, \tag{18}$$

$$h_j = H_3\left(ID_j' || U_j || X_j\right), \tag{19}$$

$$V_j = g^{h_j x_j + h_j u_j} \cdot d_j. \tag{20}$$

Table 4: Comparisons of functional properties.

	[20]	[21]	[22]	Ours
Anonymity	×	√	√	√
Universal	×	×	√	√
Mutual authentication	×	×	×	√
Access control	√	√	√	√

Then, the staff calculates the session key K, the key confirmation message Confir, and sends $\{sig_j(U_j, V_j), Confir\}$ to the user. K and Confir can be computed as

$$K = U_i^{'\mu_j}, \tag{21}$$

$$\text{Confir} = H_6\left(ID_i' \| ID_j' \| U_i' \| U_j \| K\right). \tag{22}$$

The user calculates k_j, h_j according to

$$k_j = H_2\left(ID_j' \| R_j\right), \tag{23}$$

$$h_j = H_3\left(ID_j' \| U_j \| X_j\right), \tag{24}$$

checks Equation (25). If the equation holds, the staff is considered legitimate.

$$\left(V_j, g\right) = e\left(R_j \cdot PK^{k_j} \cdot X_j^{h_j}, g\right) e\left(U_j^{h_j}, g\right). \tag{25}$$

The user then calculates K and verifies whether Equation (22) is established. And the session key K is accepted if the equation is established. The K can be computed as

$$K = U_j^{u_i'}. \tag{26}$$

The user has completed the mutual authentication with the staff, and both of them have obtained the same session key for the transmission of important information.

4.4. Passport Search and Access. In this phase, the staff gets a trapdoor and a decryption key from the user through the session key and uses the trapdoor to get the ciphertext of passport from IPFS. Then, with the decryption key, the staff decrypts the ciphertext to obtain the passport.

Passport-Search. The user calculates the trapdoor T, and the decryption key K_p according to

$$T = g^{1/x_i} \cdot PK_c^{u_i}, \tag{27}$$

$$K_p = e\left(R_i \cdot PK^{k_i} \cdot X_i^{h_i}, L_i\right)^{1/x_i}, \tag{28}$$

sends $\{T, K_p\}$ to the staff with the support of K. The staff sends $\{T, U_i\}$ to an IBC node. The IBC node calculates T' as

$$T' = \frac{T}{U_i^{SK_c}}, \tag{29}$$

and checks Equation (30), and locates the specific block on the DBC according to the ID_{DB} if the equation holds. Then, IPFS searches corresponding C_p according to the hash(C_p) and sends it to the staff.

$$idx = H_5\left(e\left(L_i, T'\right) \| w\right). \tag{30}$$

Passport-Access. The staff decrypts the C_p with the decryption key K_p, where passport $= C_p/K_p$.

At this point, the staff uses user's trapdoor to search for the ciphertext of the passport. Through the decryption key, user's passport is finally obtained by the staff.

5. Correctness and Security Analysis

5.1. Correctness and Security Analysis. In this section, we analyse the correctness of critical steps in our scheme.

Authentication-Correctness:

$$\begin{aligned} e(V_i, g) &= e\left(R_i \cdot PK^{k_i} \cdot X_i^{h_i}, g\right) e\left(U_i^{h_i}, g\right) \\ &= e\left(g^{h_i x_i + h_i u_i} \cdot d_i, g\right) = e\left(g^{h_i x_i + h_i u_i} \cdot g^{r_i + k_i x}, g\right) \\ &= e\left(g^{r_i + k_i x + h_i x_i + h_i u_i}, g\right) = e\left(R_i \cdot PK^{k_i} \cdot X_i^{h_i} \cdot U_i^{h_i}, g\right) \\ &= e\left(R_i \cdot PK^{k_i} \cdot X_i^{h_i}, g\right) e\left(U_i^{h_i}, g\right). \end{aligned} \tag{31}$$

Decryption-Correctness:

$$\begin{aligned} \text{passport} = \frac{C_p}{K_p} &= \frac{C_p}{e\left(R_i \cdot PK^{k_i} \cdot X_i^{h_i}, L_i\right)^{1/x_i}} \\ &= \frac{e\left(R_i \cdot PK^{k_i} \cdot X_i^{h_i}, g^{l_i}\right)}{e\left(R_i \cdot PK^{k_i} \cdot X_i^{h_i}, L_i\right)^{1/x_i}} \times \text{passpt} \\ &= \frac{e\left(R_i \cdot PK^{k_i} \cdot X_i^{h_i}, g^{l_i}\right)}{e\left(R_i \cdot PK^{k_i} \cdot X_i^{h_i}, X_i^{l_i}\right)^{1/x_i}} \times \text{passpt} \\ &= \frac{e\left(R_i \cdot PK^{k_i} \cdot X_i^{h_i}, g^{l_i}\right)}{e\left(R_i \cdot PK^{k_i} \cdot X_i^{h_i}, g^{x_i l_i}\right)^{1/x_i}} \times \text{passpt} = \text{passpt}. \end{aligned} \tag{32}$$

Table 5: Comparison of computational complexity.

Scheme	Users	Other devices	Total
[23]	$3T_h + 2T_m + 1T_e + T_p$	$4T_h + 2T_m + 1T_e + 2T_p$	$7T_h + 4T_m + 2T_e + 3T_p$
[24]	$5T_h + 4T_m + 1T_e + T_p$	$5T_h + 5T_m + 3T_a + T_p$	$10T_h + 9T_m + 1T_e + 3T_a + 2T_p$
Ours	$4T_h + 3T_m + 3T_e + T_p$	$4T_h + 3T_m + 3T_e + T_p$	$8T_h + 6T_m + 6T_e + 2T_p$

Search-Correctness:

$$\begin{aligned} idx &= H_5\left(e\left(L_i, T'\right)\|w\right) = H_5\left(e\left(X_i^{l_i}, \frac{T}{U_i^{SK_c}}\right)\|w\right) \\ &= H_5\left(e\left(g^{x_i l_i}, \frac{g^{1/x_i}\cdot PK_c^{u_i}}{g^{u_i SK_c}}\right)\|w\right) \\ &= H_5\left(e\left(g^{x_i l_i}, \frac{g^{1/x_i}\cdot g^{SK_c u_i}}{g^{u_i SK_c}}\right)\|w\right) \\ &= H_5\left(e\left(g^{l_i}, g\right)\|w\right) = idx. \end{aligned} \tag{33}$$

5.2. Security Analysis. Confidentiality and Privacy. In our scheme, the user interacts with other entities by virtue of a legal pseudoidentity. The attacker cannot infer user's real identity through the ID' unless he cracks user's secret key x_i or the random number μ picked by the KGC. The attacker also cannot obtain effective data even if the IPFS is hacked, because the IPFS stores the ciphertext of passport. In the step of "Passport-Search," only the user can generate a trapdoor and send it to the staff for searching, and then, IPFS returns the corresponding C_p to the staff. Thus, users have full control over their data.

Mutual Authentication. In the phase of "Authentication and Negotiation," the user signs his identity information with the private key $sk_i = (x_i, d_i)$ to get $\mathrm{sig}_i(U_i, V_i)$, where $V_i = g^{h_i x_i + h_i u_i}\cdot d_i$. The customs staff verifies V_i with user's public key $pk_i = (R_i, X_i)$. The correctness of this step has been given above. Therefore, the scheme achieves the goal of mutual authentication.

Traceability and Nonrepudiation. In our scheme, the information of each user's vaccination is uploaded to DBC and IBC. Each transaction contains the identity of the producer, known as ID_{EPA} or ID_{ctry}. Once the user has a health problem due to the vaccine, it can be traced back to the corresponding country or EPA, and the corresponding $\mathrm{sig}_{\mathrm{EPA}}$ and $\mathrm{sig}_{\mathrm{ctry}}$ can avoid producer repudiation.

Impersonation Attack. It is impossible for an attacker to pose as a legitimate user unless he cracks user's private key sk_i, and the attacker cannot impersonate the staff as well. Assume that an attacker wants to impersonate a legitimate entity, he must sign with user's private key in the "Authentication and Negotiation" phase, which is hard because only the user knows the secret value x_i.

Insider Attack. KGC cannot reveal the private key sk_i of users because it is only responsible for generating partial-private keys in the phase of "User-Registration." In addition, all the vaccination records will be uploaded to blockchain, and the traceability and nonrepudiation characteristics ensure that blockchain nodes will not upload fake information.

6. Performance Evaluation

In this section, we make a functional property comparison between the proposed scheme and the existing immunity passport schemes [20–22]. Then, the proposed scheme is compared with the existing authentication schemes [23, 24] in terms of computational overheads, communicational overheads, and energy overheads.

6.1. Functional Comparison. Table 4 shows the comparison of the functional properties of our scheme with other immunity passport schemes. From Table 4, we can see that all four schemes achieve access control of user data. Hasan et al.'s scheme [20] cannot provide anonymity, although blockchain is used in their scheme. Schemes in [21, 22] did not consider the issue of coordination between different departments in multiple countries in the scenarios of immunity passport. Moreover, schemes in [20–22] all cannot provide mutual authentication between the user and the passport inspector. Our scheme achieves these functions well.

6.2. Overheads Comparison. The computational complexity comparison of our scheme and schemes [23, 24] in the phase of authentication is shown in Table 5. Among them, T_h, T_m, T_e, T_a, and T_p, respectively, represents the time of hash function, point multiplication, modular exponentiation, point addition, and bilinear mappings.

For comparing the computational overheads, we conducted simulations on a PC with an Intel Core i5-7300HQ CPU at 2.50 GHz and 8 GB RAM, running Windows 10 Home (64 bit). Simulations show that the operation time of T_h, T_m, T_e, T_a, and T_p, which are about 0.0018 ms, 0.0012 ms, 0.0021 ms, 0.0127 ms, and 2.7737 ms, respectively. The computational overhead comparison of the user, other devices, and the total are shown in Figures 4, 5, and 6.

As for the computation of users, user in our scheme requires to calculate $\{U_i', h_i', V_i', k_j, h_j, e(V_j, g), K\}$, that is $4T_h + 3T_m + 3T_e + T_p$ (2.7908 ms). Similarly, Mahmood et al.'s scheme [23] requires $3T_h + 2T_m + 1T_e + T_p$ (2.7838 ms), and Jia et al.'s scheme [24] requires $5T_h + 4T_m + 1T_e + T_p$ (2.7908 ms). Figure 4 shows that our scheme is similar as other schemes in terms of users' computational overheads. Comparing the computational overheads of other devices, our scheme requires to calculate $\{k_i, h_i', e(V_i', g), U_j, h_j, V_j, K, \mathrm{Confir}\}$, that is $4T_h + 3T_m + 3T_e + T_p$ (2.7908 ms). Similarly, scheme [23] requires $4T_h + 2T_m + 1$

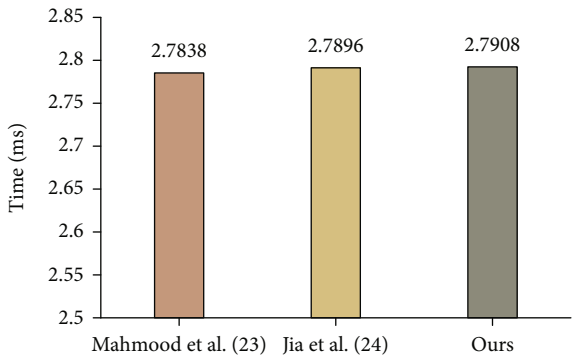


Figure 4: Computational overheads comparison: users.

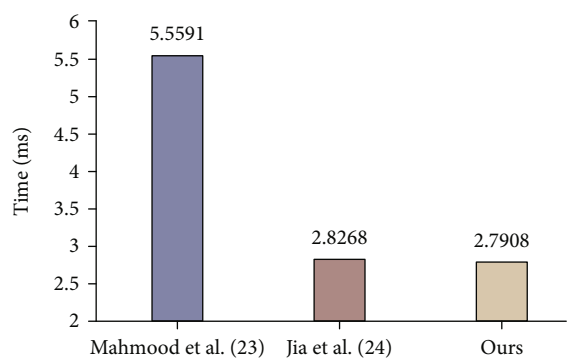


Figure 5: Computational overheads comparison: other devices.

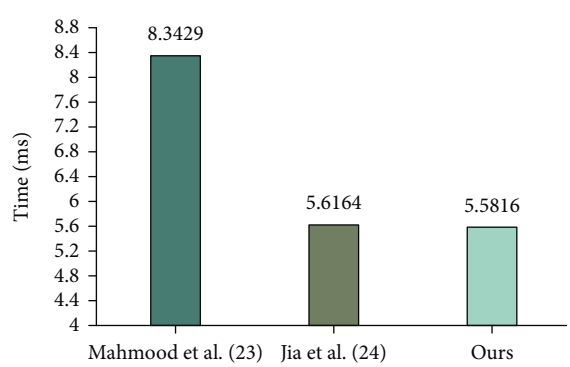


Figure 6: Computational overheads comparison: total.

$T_e + 2T_p$ (5.5591 ms), and scheme [24] requires $5T_h + 5T_m + 3T_a + T_p$ (22.8268 ms). As can be seen from Figure 5, our scheme and scheme [24] are significantly better than scheme [23], because the number of bilinear mappings operation is reduced, which is time-consuming. Furthermore, it can be seen that the computational overheads of our scheme are equal between the users and other devices. As for the total computational overheads, our scheme performs similarly to scheme [24], with a 33.10% reduction compared to scheme [23], which can be seen from Figure 6.

The bit length of a signature, a public key pair, and the hash values are assumed 256 bits. The identity and the timestamp are, respectively, assumed 128 bits and 32 bits, respectively. Our scheme needs to transmit $\{|U_i'|, |V_i'|, |U_j|, |V_j|, |\text{Confir}|\}$,

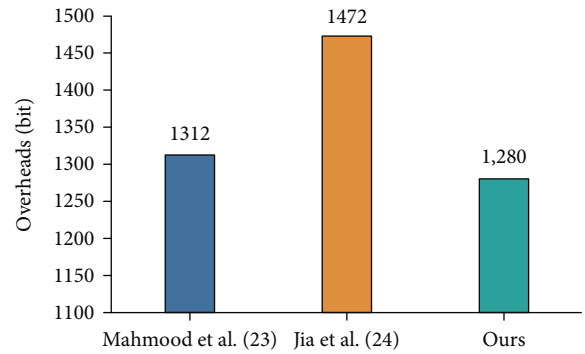


Figure 7: Communicational overheads comparison: total.

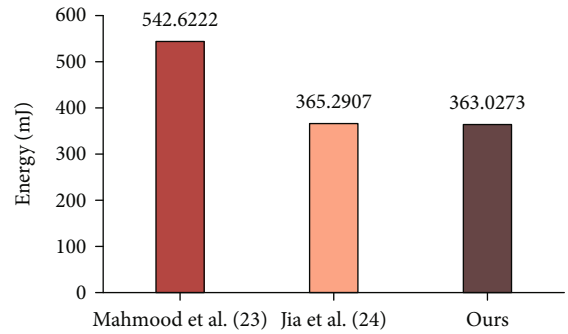


Figure 8: Energy overheads comparison: total.

that is 1280 bits. Similarly, scheme [23] needs to transmit 1312 bits during authentication; scheme [24] needs to transmit 1472 bits. We can see from Figure 7 that the performance of the communicational overhead of our scheme is a little different from scheme [23]. However, our scheme only requires two rounds of massage exchange, whereas scheme [23] requires three rounds. And our scheme reduced 13.04% compared to scheme [24] because the transmission of unnecessary information is reduced in our scheme, such as timestamps.

Energy overheads is also an important evaluation indicator. We use the voltage and current of the PC used in the simulations for comparing energy overheads, which are 1.2 V and 54.2 A, respectively. A hash function consumes $1.2\ \text{V} * 54.2\ \text{A} * 0.0018\ \text{ms} = 0.1171\ \text{mj}$, a point multiplication consumes $1.2\ \text{V} * 54.2\ \text{A} * 0.0012\ \text{ms} = 0.0780\ \text{mj}$, a modular exponentiation consumes $1.2\ \text{V} * 54.2\ \text{A} * 0.0021\ \text{ms} = 0.1366\ \text{mj}$, a point addition consumes $1.2\ \text{V} * 54.2\ \text{A} * 0.0127\ \text{ms} = 0.8260\ \text{mj}$, and a bilinear mappings consumes $1.2\ \text{V} * 54.2\ \text{A} * 2.7737\ \text{ms} = 180.4014\ \text{mj}$. The total energy overheads comparison can be seen in Figure 8, which shows that the energy overheads of our scheme is almost equal to that of scheme [24] and still better than that of scheme [23].

7. Conclusion

In this paper, we propose an immunity passport scheme to mitigate the impact of COVID-19. This scheme helps people travel between different countries without going through tedious epidemic prevention procedures in this era of postepidemic. The highlight of this scheme is that it combines

searchable encryption and authentication with blockchain, which ensures users' privacy and allows them to have control over their data. According to the security analysis, our scheme can well meet the security requirements of the immunity passport scenarios. Furthermore, the evaluation results show that compared with other schemes, our scheme has better communication and computing performance while achieving the functional properties. In the next, designing an efficient consensus mechanism and detailed smart contracts for this scheme is our future research direction.

Data Availability

No data were used to support this study.

Conflicts of Interest

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

Authors' Contributions

Hancheng Gao and Haoyu Ji are the co-first author.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (grant numbers 62072252 and 61872194).

References

[1] O. E. Awosusi and E. Shaib, "COVID-19 induced changes on lifestyles education and socio-economic activities in West African states: recovery strategies for post Pandemic Era," *International Journal of World Policy and Development Studies*, vol. 6, no. 64, pp. 38–43, 2020.

[2] T. P. Velavan and C. G. Meyer, "The COVID-19 epidemic," *Tropical Medicine & International Health*, vol. 25, no. 3, pp. 278–280, 2020.

[3] R. Padhan and K. P. Prabheesh, "The economics of COVID-19 pandemic: a survey," *Economic Analysis and Policy*, vol. 70, pp. 220–237, 2021.

[4] J. S. Tregoning, K. E. Flight, S. L. Higham, Z. Wang, and B. F. Pierce, "Progress of the COVID-19 vaccine effort: viruses, vaccines and variants versus efficacy, effectiveness and escape," *Nature Reviews Immunology*, vol. 21, no. 10, pp. 626–636, 2021.

[5] J. V. Lazarus, S. C. Ratzan, A. Palayew et al., "A global survey of potential acceptance of a COVID-19 vaccine," *Nature Medicine*, vol. 27, no. 2, pp. 225–228, 2021.

[6] M. Sallam, "COVID-19 vaccine hesitancy worldwide: a concise systematic review of vaccine acceptance rates," *Vaccines*, vol. 9, no. 2, p. 160, 2021.

[7] M. Ansari, A. Mohammad Aghaei, Y. Rezaie, and Y. Rostam-Abadi, "Discrimination in COVID-19 vaccination programs - a possible risk for mental health," *Asian Journal of Psychiatry*, vol. 63, article 102758, 2021.

[8] C. Dye and M. C. Mills, "COVID-19 vaccination passports," *Science*, vol. 371, no. 6535, p. 1184, 2021.

[9] I. de Miguel Beriain and J. Rueda, "Immunity passports, fundamental rights and public health hazards: a reply to Brown et al.," *Journal of Medical Ethics*, vol. 46, no. 10, pp. 660-661, 2020.

[10] J. Pang, Y. Huang, Z. Xie, J. Li, and Z. Cai, "Collaborative city digital twin for the COVID-19 pandemic: a federated learning solution," *Tsinghua Science and Technology*, vol. 26, no. 5, pp. 759–771, 2021.

[11] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, "Blockchain challenges and opportunities: a survey," *International Journal of Web and Grid Services*, vol. 14, no. 4, pp. 352–375, 2018.

[12] C. C. Agbo, Q. H. Mahmoud, and J. M. Eklund, "Blockchain technology in healthcare: a systematic review," *Healthcare*, vol. 7, no. 2, p. 56, 2019.

[13] S. Xu, X. Chen, and Y. He, "EVchain: an anonymous blockchain-based system for charging-connected electric vehicles," *Tsinghua Science and Technology*, vol. 26, no. 6, pp. 845–856, 2021.

[14] H. Xu, L. Zhang, O. Onireti, Y. Fang, W. J. Buchanan, and M. A. Imran, "BeepTrace: blockchain-enabled privacy-preserving contact tracing for COVID-19 pandemic and beyond," *IEEE Internet of Things Journal*, vol. 8, no. 5, pp. 3915–3929, 2021.

[15] L. Garg, E. Chukwu, N. Nasser, C. Chakraborty, and G. Garg, "Anonymity preserving IoT-based COVID-19 and other infectious disease contact tracing model," *IEEE Access*, vol. 8, pp. 159402–159414, 2020.

[16] J. Song, T. Gu, X. Feng, Y. Ge, and P. Mohapatra, "Blockchain meets COVID-19: a framework for contact information sharing and risk notification system," 2020, http://arxiv.org/abs/2007.10529.

[17] S. Jacob and J. Lawarée, "The adoption of contact tracing applications of COVID-19 by European governments," *Policy Design and Practice*, vol. 4, pp. 1–15, 2020.

[18] S. Abuhammad, O. F. Khabour, and K. H. Alzoubi, "COVID-19 contact-tracing technology: acceptability and ethical issues of use," *Patient Preference and Adherence*, vol. Volume 14, pp. 1639–1647, 2020.

[19] S. M. Idrees, M. Nowostawski, and R. Jameel, "Blockchain-based digital contact tracing apps for COVID-19 pandemic management: issues, challenges, solutions, and future directions," *JMIR Medical Informatics*, vol. 9, no. 2, article e25245, 2021.

[20] H. R. Hasan, K. Salah, R. Jayaraman et al., "Blockchain-based solution for COVID-19 digital medical passports and immunity certificates," *IEEE Access*, vol. 8, pp. 222093–222108, 2020.

[21] S. Chaudhari, M. Clear, P. Bradish, and H. Tewari, "Framework for a DLT based COVID-19 passport," *Intelligent Computing*, pp. 108–123, 2021.

[22] C. M. Angelopoulos, A. Damianou, and V. Katos, "DHP framework: digital health passports using blockchain–use case on international tourism during the COVID-19 pandemic," 2020, http://arxiv.org/abs/2005.08922.

[23] K. Mahmood, X. Li, S. A. Chaudhry et al., "Pairing based anonymous and secure key agreement protocol for smart grid edge computing infrastructure," *Future Generation Computer Systems*, vol. 88, pp. 491–500, 2018.

[24] X. Jia, D. He, N. Kumar, and K.-K. R. Choo, "A provably secure and efficient identity-based anonymous authentication scheme for mobile edge computing," *IEEE Systems Journal*, vol. 14, no. 1, pp. 560–571, 2020.

[25] R. Almadhoun, M. Kadadha, M. Alhemeiri, M. Alshehhi, and K. Salah, "A user authentication scheme of IoT devices using blockchain-enabled fog nodes," in *2018 IEEE/ACS 15th International Conference on Computer Systems and Applications (AICCSA)*, pp. 1–8, Aqaba, Jordan, 2018.

[26] A. Yazdinejad, G. Srivastava, R. M. Parizi, A. Dehghantanha, K. K. R. Choo, and M. Aledhari, "Decentralized authentication of distributed patients in hospital networks using blockchain," *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 8, pp. 2146–2156, 2020.

[27] M. Pilkington, "Blockchain technology: principles and applications," in *Research Handbook on Digital Transformations*, pp. 225–253, Edward Elgar Publishing, 2016.

[28] F. Zhang, R. Safavi-Naini, and W. Susilo, "An efficient signature scheme from bilinear pairings and its applications," *International Workshop on Public Key Cryptography*, vol. 2947, pp. 277–290, 2004.

[29] S. M. H. Bamakan, A. Motavali, and A. B. Bondarti, "A survey of blockchain consensus algorithms performance evaluation criteria," *Expert Systems with Applications*, vol. 154, p. 113385, 2020.

[30] Y. Xiao, N. Zhang, W. Lou, and Y. T. Hou, "A survey of distributed consensus protocols for blockchain networks," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 2, pp. 1432–1465, 2020.